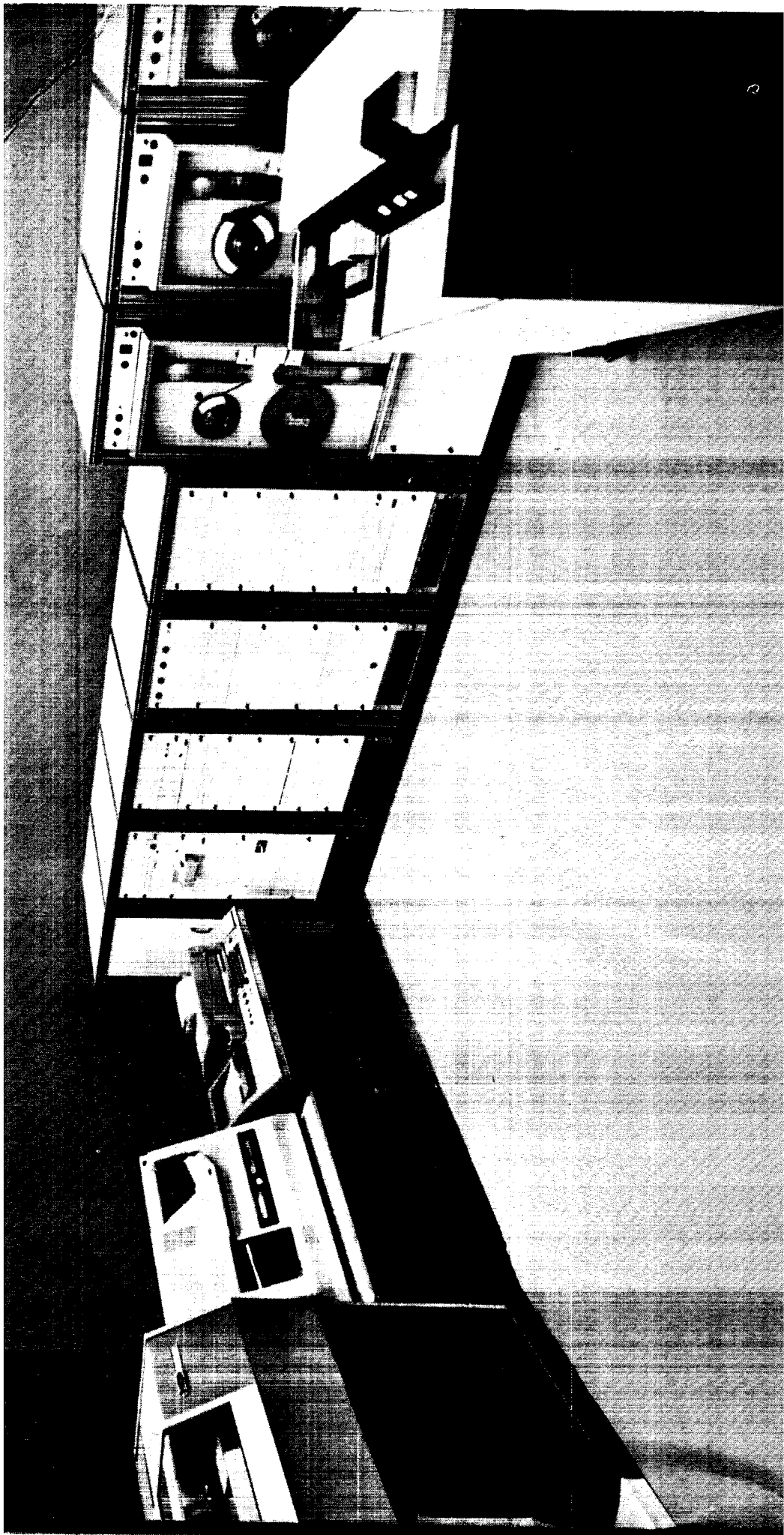


RAYTHEON
706
COMPUTER

USER'S MANUAL



RAYTHEON
706
COMPUTER
USER'S MANUAL

SP-328 A

RAYTHEON

RAYTHEON COMPUTER

2700 SOUTH FAIRVIEW ST., SANTA ANA, CALIFORNIA 92704

FEBRUARY 1970

This manual contains proprietary information that shall not be reproduced or transferred to other documents of disclosed to others, or used for manufacturing or any other purpose without prior written permission of Raytheon Computer. Where related to a U.S. Government Contract, such information may be disclosed as provided in such contract and may also be used by the U.S. Government for maintenance, servicing, and repair purposes or as otherwise provided by such prior written permission.

INSERT LATEST CHANGED PAGES, DESTROY SUPERSEDED PAGES.

LIST OF EFFECTIVE PAGES

NOTE: The portion of the text affected by the changes is indicated by a vertical line in the outer margins of the page.

Page No.	Issue	Page No.	Issue
Title	Original	Appendices	Original
ii thru xxxvi	Original	Blank	Original
1.1 thru 1.38	Original	A.1 thru A.3	Original
2.1 thru 2.31	Original	A.4 Blank	Original
2.32 Blank	Original	B.1 thru B.2	Original
3.1 thru 3.3	Original	C.1	Original
3.4 Blank	Original	C.2 Blank	Original
4.1 thru 4.14	Original	D.1 thru D.3	Original
5.1 thru 5.11	Original	D.4 Blank	Original
5.12 Blank	Original	E.1 thru E.15	Original
5.13 thru 5.21	Original	E.16 Blank	Original
5.22 Blank	Original	F.1 thru F.3	Original
5.23	Original	F.4 Blank	Original
5.24 Blank	Original	G.1 thru G.2	Original
5.25 thru 5.47	Original	H.1 thru H.15	Original
5.48 Blank	Original	H.16 Blank	Original
5.49 thru 5.77	Original	I.1	Original
5.78 Blank	Original	I.2 Blank	Original
5.79 thru 5.93	Original	J.1 thru J.37	Original
5.94 Blank	Original	J.38 Blank	Original
5.95 thru 5.133	Original	K.1 thru K.18	Original
5.134 Blank	Original	L.1 thru L.4	Original
5.135 thru 5.155	Original	M.1 thru M.45	Original
5.156 Blank	Original	M.46 Blank	Original
5.157 thru 5.162	Original	N.1 thru N.15	Original
6.1 thru 6.5	Original	N.16 Blank	Original
6.6 Blank	Original	O.1 thru O.3	Original
7.1 thru 7.7	Original	O.4 Blank	Original
7.8 Blank	Original	P.1 thru P.3	Original
8.1 thru 8.17	Original	P.4 Blank	Original
8.18 Blank	Original	Q.1 thru Q.3	Original
8.19 thru 8.29	Original	Q.4 Blank	Original
8.30 Blank	Original	R.1	Original
8.31 thru 8.92	Original	R.2 Blank	Original
9.1 thru 9.9	Original	Reply Card	Original
9.10 Blank	Original	Index-1 thru Index-12	Original
9.11 thru 9.51	Original	Reader Comment	Original
9.52 Blank	Original		
9.53 thru 9.63	Original		
9.64 Blank	Original		

*The asterisk indicates pages changed, added or deleted by the current change.

ADDITIONAL COPIES OF THIS MANUAL MAY BE OBTAINED FROM:

Raytheon Computer
 Order Services Dept.
 2700 Fairview St.
 Santa Ana, Calif. 92704

TABLE OF CONTENTS

Section	Page
1 GENERAL SYSTEM DESCRIPTION	
1-1 INTRODUCTION	1.1
1-1.1 System Description and Configuration	1.1
1-2 CENTRAL PROCESSING UNIT	1.6
1-2.1 Register Description and Operation	1.6
1-2.1.1 Program Counter	1.6
1-2.1.2 Instruction Register	1.6
1-2.1.3 Accumulator	1.6
1-2.1.4 Memory Buffer Register	1.6
1-2.1.5 Index Register	1.6
1-2.1.6 Extension Register	1.9
1-2.1.7 Overflow Indicator	1.9
1-2.1.8 Comparison Indicators	1.9
1-2.1.9 Local/Global Indicator	1.9
1-2.2 Internal Number Representation	1.9
1-2.3 Timing	1.9
1-3 WORD FORMATS	1.10
1-3.1 Instruction Word Formats	1.10
1-3.2 Data Word Formats	1.10
1-3.2.1 Logical Word and Single Precision Fixed Point Data	1.10
1-3.2.2 Double Precision Integer and Multiprecision Floating Point Data	1.10
1-3.3 Status Word Formats	1.11
1-3.4 Address Word Formats	1.11
1-3.5 Assembler Source Statement Formats	1.11
1-3.5.1 Title	1.11
1-3.5.2 Comment	1.11
1-3.5.3 Instruction	1.11
1-4 ADDRESSING	1.17
1-4.1 Page Numbering	1.17
1-4.2 Word and Byte Addresses	1.17
1-4.3 Extension Register	1.17
1-4.3.1 Loading the Extension Register	1.19
1-4.4 Direct Addressing	1.19
1-4.4.1 Set Memory Bank (SMB)	1.19
1-4.4.2 Direct Addressing Example	1.20

TABLE OF CONTENTS (Cont)

Section		
1	1-4.5	Index Register
	1-4.5.1	Decrementing and Incrementing
	1-4.5.2	Subroutine Linkage
	1-4.5.3	General Purpose Use
	1-4.6	Indexed Addressing
	1-4.6.1	Local Mode Indexing
	1-4.6.2	Local Mode Indexing Example.
	1-4.6.3	Global Mode Indexing
	1-4.6.4	Global Mode Indexing Example
	1-4.6.5	Using Global Mode to Return from Subroutines
	1-4.7	Indirect Addressing
	1-4.7.1	Multilevel Indirect Addressing
	1-5	INPUT/OUTPUT
	1-5.1	Direct Input/Output.
	1-5.2	Direct Memory Access
	1-5.3	Automatic Priority Interrupt System
	1-6	SOFTWARE
	1-6.1	Software Systems
	1-6.1.1	BASIC Software System
	1-6.1.2	STANDARD Software System.
	1-6.1.3	EXTENDED Software System
	1-6.1.4	ADVANCED Software System.
	1-6.2	Monitors and Executives
	1-6.2.1	X-RAY BASIC
	1-6.2.2	X-RAY STANDARD.
	1-6.2.3	Magnetic Tape Operating System
	1-6.2.4	Real-Time Operating System.
	1-6.2.5	Multi-Programming System.
	1-6.3	Program Generation
	1-6.3.1	Symbolic Program Preparation.
	1-6.3.2	FORTRAN PREP
	1-6.3.3	Symbolic Debugging
	1-6.3.4	Symbolic Program Editor
	1-6.4	Assemblers and Compilers.
	1-6.4.1	SYM I
	1-6.4.2	SYM II

TABLE OF CONTENTS (Cont)

tion		Page
1-6.4.3	Conversational FORTRAN	1.35
1-6.4.4	Real-Time FORTRAN IV	1.35
1-6.5	Library Programs	1.36
1-6.5.1	System Generation	1.36
1-6.5.2	Loaders	1.36
1-6.5.3	Disc Loaders	1.36
1-6.5.4	Conversational FORTRAN Library	1.36
1-6.5.5	Real-Time FORTRAN IV Library	1.37
1-6.5.6	Math Library	1.37
1-6.6	Sensor Diagnostics	1.37
2	INSTRUCTIONS	
2-1	HARDWARE	2.1
2-1.1	General	2.1
2-1.2	Class 1 - Word Reference	2.1
2-1.3	Class 2 - Byte Reference	2.6
2-1.4	Class 3 - Generic with 4-bit Operand	2.7
2-1.5	Class 4 - Shift	2.8
2-1.6	Class 5 - Halt	2.14
2-1.7	Class 6 - Generic with No Address	2.15
2-1.8	Class 7 - Generic with Byte Operand	2.21
2-1.9	Class 8 - Input/Output	2.23
2-1.10	Class 9 - Optional Hardware Double Word Instructions	2.24
2-1.11	Class 10 - Set Memory Bank	2.25
2-2	PSEUDO INSTRUCTIONS	2.26
2-2.1	General	2.26
2-2.2	Symbol Data and Definition	2.26
2-2.3	Assembler Control Directives	2.29
3	INTERRUPT	
3-1	GENERAL	3.1
3-2	PROGRAMMING WITH THE INTERRUPT SYSTEM	3.1
3-3	OPERATION OF THE INTERRUPT SYSTEM	3.2
3-4	INTERRUPT RANKING AND ASSIGNMENT	3.3

TABLE OF CONTENTS (Cont)

Section		
4	INPUT/OUTPUT	
	4-1	GENERAL 4
	4-2	DIRECT INPUT/OUTPUT (DIO) CHANNEL. 4
	4-2.1	General Description 4
	4-2.2	Channel Operation. 4
	4-2.2.1	DIN Instruction 4
	4-2.2.2	DOT Instruction 4
	4-3	DIRECT MEMORY ACCESS (DMA) CHANNEL. 4
	4-3.1	General Description 4
	4-3.2	Control 4
	4-4	CABLING AND INTERFACE. 4
	4-4.1	DIO Cable 4
	4-4.2	DMA Cable 4
	4-4.3	Cabling of Peripheral Devices 4
5	PERIPHERAL EQUIPMENT OPERATION AND PROGRAMMING	
	5-1	GENERAL 5
	5-1.1	Device Selection 5
	5-1.2	Data Bit Assignment for DIO Transfer 5
	5-1.3	Status Bit Assignments 5
	5-2	DIO PROGRAMMING. 5
	5-2.1	Status Test Method 5
	5-2.1.1	Protection 5
	5-2.1.2	Selection 5
	5-2.1.3	Wait for Data Ready 5
	5-2.1.4	Service Data 5
	5-2.1.5	End-of-Transmission & Disconnect 5
	5-2.2	Burst Method 5
	5-2.3	Interrupt Method 5
	5-2.3.1	Linkage Address Set-up 5
	5-2.3.2	I/O Device Selection 5
	5-2.3.3	Interrupt & Mask Control 5
	5-2.3.4	Service First Character 5
	5-2.3.5	Continue Processing 5
	5-2.3.6	I/O Service Routine(s) 5
	5-3	DMA PROGRAMMING 5

TABLE OF CONTENTS (Cont)

Section		Page
5	5-4 INPUT/OUTPUT SOFTWARE (IOS)	5.25
	5-4.1 IOS Functions	5.25
	5-4.1.1 IOS Calls	5.25
	5-4.1.2 IOS Drivers.	5.25
	5-4.1.3 IOS Interrupt Service.	5.25
	5-4.2 Peripheral Equipment Assignment Table (PEAT)	5.26
	5-4.2.1 Logical Unit Definitions	5.26
	5-4.2.2 PEAT Table Description	5.30
	5-4.3 File Input/Output Table (FIOT)	5.31
	5-4.4 Basic IOS Calls	5.32
	5-4.4.1 OPEN.	5.33
	5-4.4.2 DOIO.	5.33
	5-4.4.3 STAT.	5.33
	5-4.5 Special IOS Calls.	5.33
	5-4.5.1 Device Status.	5.37
	5-4.5.2 Write End-of-File.	5.37
	5-4.5.3 Seek End-of-File	5.37
	5-4.5.4 Backspace.	5.37
	5-4.5.5 Write-Skip	5.37
	5-4.5.6 Rewind	5.37
	5-4.6 End Action	5.37
	5-4.7 Device Driver References	5.38
	5-5 TELETYPE ASR-33.	5.39
	5-5.1 Speed	5.39
	5-5.2 Data Type	5.39
	5-5.3 Data Format.	5.39
	5-5.4 Modes	5.39
	5-5.4.1 Disconnect	5.39
	5-5.4.2 Paper Tape Read	5.39
	5-5.4.3 Print/Punch.	5.39
	5-5.4.4 Keyboard.	5.39
	5-5.5 Function Codes	5.39
	5-5.6 Status Codes	5.39
	5-5.7 Interrupt Meaning	5.42
	5-5.8 Timing	5.42
	5-5.9 Operation Notes	5.42

TABLE OF CONTENTS (Cont)

Section		Page
5	5-6 HIGH SPEED PAPER TAPE READER AND PUNCH	5.43
	5-6.1 Speed	5.43
	5-6.2 Data Type	5.43
	5-6.3 Data Format	5.43
	5-6.4 Reader Functions	5.43
	5-6.4.1 Disconnect	5.43
	5-6.4.2 Read Forward	5.43
	5-6.4.3 Transfer Character and Read Forward	5.43
	5-6.4.4 Transfer Character and Disconnect	5.43
	5-6.5 Punch Functions.	5.43
	5-6.5.1 Disconnect	5.43
	5-6.5.2 Turn on Punch Power	5.43
	5-6.5.3 Output Character and Turn on Punch Power	5.43
	5-6.5.4 Turn Off Punch Power.	5.44
	5-6.6 Function Codes	5.44
	5-6.7 Status Codes.	5.44
	5-6.8 Interrupt Meaning	5.44
	5-6.9 Timing	5.44
	5-6.10 Operation Notes	5.44
	5-6.10.1 Tape Spooler (Model No. 75603)	5.44
	5-6.10.2 Punching Leader	5.47
	5-6.10.3 Reading Leader.	5.47
	5-6.11 High Speed Paper Tape Reader (Model No. 75601) and Punch (Model No. 75602) Operation.	5.47
	5-6.11.1 Controls and Indicators	5.47
	5-6.11.2 Loading.	5.47
	5-6.11.3 Operation.	5.47
	5-7 CARD READER AND PUNCH	5.49
	5-7.1 Speed	5.49
	5-7.2 Data Type	5.49
	5-7.3 Data Format.	5.49
	5-7.3.1 Read Data Format	5.49
	5-7.3.2 Punch Data Format	5.49
	5-7.4 Card Reader Functions	5.49
	5-7.4.1 Disconnect	5.49
	5-7.4.2 Disconnect After Current Card is Read	5.49

TABLE OF CONTENTS (Cont)

Section		Page
5	5-7.4.3 Select Card Reader	5.49
	5-7.4.4 Stop Interrupts for Current Card	5.49
	5-7.4.5 Offset Card in Process in Output Stacker	5.49
	5-7.4.6 Transfer Data	5.50
	5-7.5 Card Punch Functions	5.50
	5-7.5.1 Disconnect	5.50
	5-7.5.2 Stop Punch	5.50
	5-7.5.3 Punch Cards	5.50
	5-7.5.4 Last Punch	5.50
	5-7.5.5 Offset Card in Process in Output Stacker	5.50
	5-7.5.6 Load Data	5.50
	5-7.6 Function Codes	5.50
	5-7.7 Status Codes	5.50
	5-7.8 Interrupt Meaning	5.50
	5-7.9 Timing	5.52
	5-7.9.1 Card Punch High Speed Skip Feature	5.52
	5-7.10 Card Reader Operation (Model No. 75611, 1100 cpm).	5.54
	5-7.10.1 Controls and Indicators	5.54
	5-7.10.2 Loading.	5.54
	5-7.10.3 Operation.	5.54
	5-7.11 Card Punch Operation (Model No. 75613, 400 cpm).	5.54
	5-7.12 Card Reader Operation (Model No. 75612, 400 cpm)	5.59
	5-7.12.1 Controls and Indicators	5.59
	5-7.12.2 Loading.	5.59
	5-7.12.3 Operation.	5.59
	5-7.12.4 Operational Troubles.	5.59
	5-8 LINE PRINTER	5.61
	5-8.1 Speed	5.61
	5-8.2 Data Type	5.61
	5-8.3 Data Format.	5.61
	5-8.4 Line Printer Functions.	5.61
	5-8.4.1 Disconnect	5.61
	5-8.4.2 Load Slew Register	5.64
	5-8.4.3 Load (Transfer) One Character.	5.64
	5-8.4.4 Print Line and Move Paper	5.64

TABLE OF CONTENTS (Cont)

Section		Page
5	5-8.4.5 Move Paper	5.6
	5-8.5 Function Codes	5.6
	5-8.6 Status Codes	5.6
	5-8.7 Interrupt Meaning	5.6
	5-8.8 Timing	5.6
	5-8.8.1 360 LPM Printer	5.6
	5-8.8.2 600 LPM Printer	5.6
	5-8.8.3 1000 LPM Printer	5.6
	5-8.9 Vertical Format Tape	5.6
	5-8.10 Operation Notes	5.6
	5-8.10.1 Data Buffer Overspill	5.6
	5-8.10.2 Overprinting	5.6
	5-8.11 Line Printer Operation (Model Nos. 75622, 75623) 360 LPM and 600 LPM	5.6
	5-8.11.1 Controls and Indicators	5.6
	5-8.11.2 Paper Loading	5.6
	5-8.11.3 Ribbon Changing	5.7
	5-8.11.4 Cleaning, Filter, Adjustments, and Lubrication	5.7
	5-8.12 Line Printer Operation (Model No. 75624-1000 LPM)	5.7
	5-8.12.1 Controls and Indicators	5.7
	5-8.12.2 Paper Loading	5.7
	5-8.12.3 Ribbon Changing	5.7
	5-8.12.4 Cleaning, Filter, Adjustment and Lubrication	5.7
	5-9 DISC MEMORY	5.73
	5-9.1 Speed	5.73
	5-9.2 Data Type	5.73
	5-9.3 Data Format	5.73
	5-9.4 Sectors per Track and Words per Sector Selection	5.73
	5-9.5 Disc Functions	5.74
	5-9.5.1 Disconnect	5.74
	5-9.5.2 Set Memory Address	5.74
	5-9.5.3 Set Track and Sector	5.74
	5-9.5.4 Set Unit Number, Number of Words, and Write	5.74
	5-9.5.5 Set Unit Number, Number of Words and Read	5.74
	5-9.5.6 Set Unit Number, Number of Words and Verify	5.75
	5-9.6 Function Codes	5.75
	5-9.7 Status Codes	5.75

TABLE OF CONTENTS (Cont)

Section	Page
5	5-9.8 Interrupt Meaning 5.75
	5-9.9 Timing 5.75
	5-9.10 Operation Notes 5.75
	5-9.10.1 Rate Errors 5.75
	5-9.10.2 Track Protection 5.75
	5-9.10.3 Bootstrap Load. 5.75
	5-9.11 Disc Operation (Model No. 74601, 74602) 5.75
	5-10 DMA 9-TRACK MAGNETIC TAPE. 5.79
	5-10.1 Speed 5.79
	5-10.2 Data Type 5.79
	5-10.3 Tape Unit Number Assignment 5.79
	5-10.4 DMA 9-Track Magnetic Tape Functions. 5.79
	5-10.4.1 Disconnect 5.79
	5-10.4.2 Set Memory Address. 5.79
	5-10.4.3 Set Word Count and Write a Record. 5.79
	5-10.4.4 Set Word Count and Read a Record. 5.82
	5-10.4.5 Continue Last Operation. 5.82
	5-10.4.6 Write End-of-File. 5.83
	5-10.4.7 Advance to End-of-File. 5.83
	5-10.4.8 Write Skip 5.84
	5-10.4.9 Rewind 5.84
	5-10.4.10 Read Memory Address. 5.84
	5-10.4.11 Backspace One Record. 5.84
	5-10.5 Data Chain Option 5.85
	5-10.5.1 Load Data Chain Address 5.85
	5-10.5.2 Load Data Chain Mode and Word Count 5.85
	5-10.5.3 Stop Data Chain 5.86
	5-10.6 Function Codes and Command Words 5.86
	5-10.7 Status Codes. 5.86
	5-10.8 Interrupt Meaning 5.88
	5-10.9 Timing 5.88
	5-10.10 Operation Notes. 5.88
	5-10.10.1 Rate Errors. 5.88
	5-10.10.2 File Protection 5.88
	5-10.10.3 Bootstrap Load. 5.89
	5-10.11 DMA 9-Track Magnetic Tape Operation (Model Nos. 73691, 73692, 73693, 73694, 73695) 5.89

TABLE OF CONTENTS (Cont)

Section	Page
5	
5-10.11.1	Controls and Indicators 5.8
5-10.11.2	Power ON and OFF Procedure 5.9
5-10.11.3	Loading Tape 5.9
5-10.11.4	Manual Operation 5.9
5-10.11.5	Automatic Operation. 5.9
5-10.11.6	Unloading Tape. 5.9
5-10.11.7	Manual Memory Load 5.9
5-11	DMA 7-TRACK MAGNETIC TAPE. 5.9
5-11.1	Speed 5.9
5-11.2	Data Type 5.9
5-11.2.1	3-Character/Word Binary. 5.9
5-11.2.2	2-Character/Word Binary. 5.9
5-11.2.3	2-Character/Word Decimal 5.9
5-11.3	Tape Unit Number Assignment 5.9
5-11.4	DMA 7-Track Magnetic Tape Functions. 5.9
5-11.4.1	Disconnect 5.9
5-11.4.2	Set Memory Address 5.9
5-11.4.3	Set Word Count and Write a Record. 5.9
5-11.4.4	Set Word Count and Read a Record. 5.9
5-11.4.5	Continue Last Operation. 5.9
5-11.4.6	Write End-of-File. 5.9
5-11.4.7	Advance to End-of-File. 5.9
5-11.4.8	Write Skip 5.9
5-11.4.9	Rewind. 5.9
5-11.4.10	Rewind and Lockout. 5.9
5-11.4.11	Read Memory Address. 5.9
5-11.4.12	Backspace One Record. 5.9
5-11.5	Data Chain Option. 5.10
5-11.5.1	Load Data Chain Address 5.10
5-11.5.2	Load Data Chain Mode and Word Count 5.10
5-11.5.3	Stop Data Chain 5.10
5-11.6	Function Codes and Command Words. 5.10
5-11.7	Status Codes. 5.10
5-11.8	Interrupt Meaning 5.10
5-11.9	Timing 5.10
5-11.10	Operation Notes 5.10

TABLE OF CONTENTS (Cont)

Section		Page	
5	5-11.10.1	Rate Errors	5.103
	5-11.10.2	File Protection	5.103
	5-11.10.3	Bootstrap Load	5.103
	5-11.11	DMA 7-Track Magnetic Tape Operation (Model Nos. 73671, 73672, 73673)	5.103
	5-11.11.1	Controls and Indicators	5.103
	5-11.11.2	Power ON and OFF Procedure.	5.106
	5-11.11.3	Loading Tape.	5.106
	5-11.11.4	Manual Operation	5.106
	5-11.11.5	Automatic Operation.	5.107
	5-11.11.6	Unloading Tape.	5.107
	5-11.11.7	Manual Memory Load	5.107
	5-12	DIO 7 AND 9-TRACK MAGNETIC TAPE	5.109
	5-12.1	Speed and Density.	5.109
	5-12.2	Data Type	5.109
	5-12.3	Tape Unit Number Assignment	5.109
	5-12.4	DIO 7 and 9-Track Magnetic Tape Functions.	5.109
	5-12.4.1	Disconnect	5.109
	5-12.4.2	Write One Record	5.109
	5-12.4.3	Transfer Data Out	5.112
	5-12.4.4	Read One Record	5.112
	5-12.4.5	Transfer Data In	5.112
	5-12.4.6	Write End-of-File.	5.112
	5-12.4.7	Backspace One Record.	5.112
	5-12.4.8	Rewind	5.112
	5-12.4.9	Write Skip	5.113
	5-12.4.10	Search for End-of-File	5.113
	5-12.5	Continuous Functions	5.113
	5-12.6	Function Codes and Command Words	5.114
	5-12.7	Status Codes.	5.114
	5-12.8	Parity and Error Detection	5.114
	5-12.9	Interrupt Meanings.	5.114
	5-12.10	Timing	5.115
	5-12.11	Operation Notes	5.116
	5-12.11.1	Skip Record Operation.	5.116
	5-12.11.2	Change of Direction	5.116

TABLE OF CONTENTS (Cont)

Section		Page
5	5-12.11.3 Simultaneous Unit Operation	5.116
	5-12.11.4 File Protection	5.116
	5-12.12 DIO Magnetic Tape Operation (Model Nos. 73696, 73697, 73674, 73675).	5.116
	5-12.12.1 Controls and Indicators	5.116
	5-12.12.2 Loading Tape on Transport	5.116
	5-12.12.3 Reel Loading	5.116
	5-12.12.4 Tape Threading.	5.117
	5-12.12.5 Bringing Tape to Load Point.	5.117
	5-12.12.6 Bootstrap Loading	5.117
	5-12.12.7 Unloading Tape.	5.117
	5-13 DIGITAL PLOTTER	5.123
	5-13.1 Speed	5.123
	5-13.2 Principles of Operation (Drum Type)	5.123
	5-13.3 Digital Plotter Functions	5.125
	5-13.13.1 Disconnect	5.125
	5-13.13.2 Initialize	5.125
	5-13.13.3 Transfer Data.	5.125
	5-13.4 Function Codes and Command Words	5.126
	5-13.5 Status Codes.	5.126
	5-13.6 Interrupt Meaning	5.126
	5-13.7 Timing	5.126
	5-13.8 Digital Plotter Operation (Model No. 75631 (565))	5.126
	5-13.8.1 Controls and Indicators	5.126
	5-13.8.2 Installation of Pen	5.126
	5-13.8.3 Installation Precautions.	5.128
	5-13.8.4 Installation of Chart Roll	5.128
	5-13.8.5 Installation of Single Sheet Chart Paper	5.130
	5-13.8.6 Operational Checkout	5.130
	5-13.8.7 Reticles.	5.130
	5-13.8.8 Automatic Operation.	5.131
	5-13.8.9 Removal of Chart Paper	5.131
	5-14 TELETYPE MULTIPLEXER	5.135
	5-14.1 Speed	5.135
	5-14.2 Data Type	5.135
	5-14.3 Multiplexer Functions	5.135
	5-14.3.1 Disconnect Channel	5.135

TABLE OF CONTENTS (Cont)

Section		Page
5	5-14.3.2	Select Channel for Input Mode 5.135
	5-14.3.3	Select Channel for Output Mode. 5.135
	5-14.3.4	Output Character. 5.136
	5-14.3.5	Input Character 5.136
	5-14.4	Teletype Multiplexer Function Codes. 5.136
	5-14.5	Status Codes. 5.136
	5-14.6	Interrupt Meaning 5.136
	5-14.7	Timing 5.136
	5-14.8	Operation Notes 5.138
	5-14.9	Teletype Multiplexer Operation (Model No. 76601 and 76602) 5.138
	5-15	BUFFERED INPUT/OUTPUT CHANNELS. 5.139
	5-15.1	Speed 5.139
	5-15.2	Data Format. 5.139
	5-15.3	Buffered Channel Input Functions 5.139
	5-15.4	Buffered Channel Output Functions 5.139
	5-15.4.1	Transfer Data to Channel 5.139
	5-15.4.2	Transfer Data to Channel and Disconnect 5.139
	5-15.5	Function Codes 5.140
	5-15.6	Status Codes. 5.140
	5-15.7	Interrupt Meaning 5.140
	5-15.8	Timing 5.142
	5-15.9	Buffered Input Channel or Buffered Output Channel. 5.142
	5-16	ANALOG TO DIGITAL CONVERTERS 5.143
	5-16.1	Output Codes 5.143
	5-16.2	Analog to Digital Function Codes. 5.143
	5-16.2.1	Disconnect 5.143
	5-16.2.2	Reset Error. 5.143
	5-16.2.3	Set Random Mode and Address Unclocked 5.143
	5-16.2.4	Set Sequential Mode Unclocked 5.143
	5-16.2.5	Set Random Mode and Address Clocked 5.143
	5-16.2.6	Set Sequential Mode Clocked 5.147
	5-16.2.7	Transfer Data and Start 5.147
	5-16.2.8	Transfer Data. 5.147
	5-16.3	Function Codes 5.147
	5-16.4	Status Codes. 5.147
	5-16.5	Interrupt Meaning. 5.147

TABLE OF CONTENTS (Cont)

Section		Page
5	5-16.6 Timing	5.15
	5-16.7 Miniverter Operation (Model Nos. 77631, 77632, 77633, 77634, 77635)	5.15
	5-16.7.1 Controls and Indicators	5.15
	5-16.8 Multiverter Operation (Model No. 77636)	5.15
	5-16.8.1 Controls and Indicators	5.15
	5-17 DIGITAL TO ANALOG CONVERTER.	5.15
	5-17.1 Speed	5.15
	5-17.2 Data Word Format	5.15
	5-17.3 Digital to Analog Function Code	5.15
	5-17.4 Converter Input Code	5.15
	5-17.5 Status and Interrupt.	5.15
	5-17.6 Timing	5.15
	5-17.7 Digital to Analog Converter Operation (Model Nos. 77641, 77642, 77643, 77644)	5.15
	5-18 TIME OF DAY CLOCK.	5.15
	5-18.1 Speed	5.15
	5-18.2 Data Word Formats	5.15
	5-18.3 Time of Day Clock Functions.	5.15
	5-18.3.1 Stop Clock	5.15
	5-18.3.2 Reset Clock	5.15
	5-18.3.3 Start Clock	5.15
	5-18.3.4 Write Word 1, Write Word 2	5.15
	5-18.3.5 Hold Clock, Read Word 1, Read Word 2, Read Word 3	5.15
	5-18.4 Function Codes	5.15
	5-18.5 Status Codes.	5.15
	5-18.6 Interval Timing	5.15
	5-18.7 Interrupt Meaning	5.16
	5-18.8 Time of Day Clock Operation (Model Nos. 77651, 77652, 77653)	5.16
	5-18.8.1 Power ON and OFF Procedures	5.16
	5-18.8.2 Manual Start, Stop and Reset	5.16
	5-18.8.3 Manual Time Set.	5.16

TABLE OF CONTENTS (Cont)

Section		Page
6	OPTIONS	
	6-1 GENERAL	6.1
	6-2 DIRECT MEMORY ACCESS	6.1
	6-3 INTERRUPT EXPANSION	6.1
	6-4 HARDWARE MULTIPLY/DIVIDE	6.1
	6-5 MEMORY PARITY CHECK	6.1
	6-6 POWER FAIL SAFE	6.2
	6-6.1 Purpose	6.2
	6-6.2 Functional Characteristics	6.2
	6-6.3 Programming	6.2
	6-6.4 Operation Note	6.2
	6-7 MEMORY PROTECT	6.2
	6-7.1 General	6.2
	6-7.2 Setup and Entry to the User State	6.3
	6-7.3 User State	6.5
	6-7.4 Leaving the User State	6.5
	6-8 MEMORY EXPANSION	6.5
7	OPERATION	
	7-1 POWER ON AND OFF PROCEDURE	7.1
	7-2 MANUAL DATA DISPLAY AND ENTRY	7.1
	7-2.1 Program Counter Register	7.1
	7-2.2 Selected Display	7.3
	7-2.3 Memory Display and Entry	7.3
	7-3 PROGRAM LOADING	7.4
	7-3.1 Manual Program Entry	7.4
	7-4 BOOTSTRAP PROGRAM LOADING	7.4
	7-4.1 Bootstrap Loading Procedure	7.4
	7-4.2 Bootstrap Device Operation	7.5
	7-4.3 Teletype Bootstrap Operation	7.5
	7-4.4 High Speed Paper Tape Bootstrap Operation	7.6
	7-4.5 DIO Magnetic Tape Bootstrap Operation	7.6
	7-4.6 Card Reader Bootstrap Operation	7.6
	7-5 DIRECT PROGRAM LOADING	7.6
	7-6 PROGRAM EXECUTION	7.6
	7-6.1 Run Mode	7.6

TABLE OF CONTENTS (Cont.)

Section		Page
7	7-6.2 Single Command Mode	7.
	7-6.3 Single-Step Mode	7.
	7-7 Manual Lockout Switch	7.
8	PROGRAMMING SYSTEMS DESCRIPTION	
	8-1 INTRODUCTION	8.
	8-2 HARDWARE/SOFTWARE CONFIGURATIONS	8.1
	8-2.1 Hardware System Configurations	8.1
	8-2.2 Non-Mass Storage vs. Mass Storage Systems	8.4
	8-2.3 Software System Configurations	8.4
	8-3 ASSEMBLERS	8.4
	8-3.1 One Pass, Two Pass Assemblers	8.6
	8-3.2 Coding	8.6
	8-3.3 Pseudo Instructions	8.8
	8-3.4 Procedures	8.8
	8-3.4.1 Using the Procedure	8.8
	8-3.4.2 Conditional Processing Within Procedures	8.1
	8-3.4.3 Procedures Within Procedures	8.1
	8-3.5 Subroutines	8.1
	8-3.6 SYM-I and SYM-II Assembler Review	8.1
	8-3.6.1 SYM-I	8.1
	8-3.6.2 SYM-II	8.1
	8-4 COMPILERS	8.1
	8-4.1 Introduction	8.1
	8-4.2 Real-Time FORTRAN-IV	8.1
	8-4.2.1 General	8.1
	8-4.2.2 Program Segmentation	8.1
	8-4.3 Conversational FORTRAN	8.1
	8-4.3.1 General	8.1
	8-4.3.2 Conversational FORTRAN Compiler	8.1
	8-4.3.3 Run-Time System	8.1
	8-4.3.4 System Design	8.1
	8-5 LOADERS	8.1
	8-5.1 Absolute Loaders	8.1
	8-5.1.1 Absolute Program Format	8.1

TABLE OF CONTENTS (Cont.)

tion

8

		Page
8-5.1.2	Absolute Load Routine	8.22
8-5.1.3	Absolute Load Routine – BASIC	8.22
8-5.1.4	Initial Loader	8.22
8-5.1.5	Absolute Bootstrap	8.23
8-5.2	Relocatable Loaders	8.23
8-5.2.1	Relocatable Loader Text	8.23
8-5.2.2	External Strings	8.24
8-5.2.3	Record Blocking of Loader Text	8.24
8-5.2.4	BASIC Relocating Loader	8.24
8-5.2.5	STANDARD Relocating Loader	8.26
8-5.2.6	Disc Relocating Loader	8.27
8-5.2.7	Linking Absoluter	8.27
8-6	INPUT/OUTPUT ROUTINES (DRIVERS)	8.31
8-6.1	Teletype, High-Speed Paper Tape I/O Driver	8.31
8-6.1.1	Purpose	8.31
8-6.1.2	Record Formats	8.31
8-6.1.3	Console Teletype	8.32
8-6.1.4	High-Speed Paper Tape Reader	8.33
8-6.1.5	High-Speed Paper Tape Punch	8.33
8-6.2	Teletype Multiplexer Driver	8.34
8-6.2.1	Purpose	8.34
8-6.2.2	Usage	8.34
8-6.2.3	Message	8.34
8-6.2.4	Error Conditions	8.34
8-6.2.5	Restrictions	8.34
8-6.3	Card Reader Driver	8.34
8-6.3.1	Purpose	8.34
8-6.3.2	Usage	8.35
8-6.3.3	Special Data Format	8.35
8-6.3.4	Binary Data Format	8.35
8-6.3.5	Hollerith (Alpha) Data Format	8.35
8-6.3.6	Messages	8.35
8-6.4	Card Punch Driver	8.37
8-6.4.1	Purpose	8.37
8-6.4.2	Usage	8.37
8-6.4.3	Method	8.37

TABLE OF CONTENTS (Cont.)

Section		Page
8	8-6.4.4 Write Special Format	8.3
	8-6.4.5 Write Binary	8.3
	8-6.4.6 Write Hollerith	8.3
	8-6.4.7 Write End-of-File	8.3
	8-6.4.8 Punch Leader	8.3
	8-6.5 Line Printer Driver	8.3
	8-6.5.1 Purpose	8.3
	8-6.5.2 Usage	8.3
	8-6.5.3 Driver Functions	8.3
	8-6.5.4 Error Conditions	8.3
	8-6.5.5 Timing	8.3
	8-6.6 Disc and DMA Magnetic Tape Driver	8.3
	8-6.6.1 Purpose	8.3
	8-6.6.2 Usage	8.3
	8-6.6.3 Error Recovery	8.4
	8-6.6.4 Disc General	8.4
	8-6.6.5 Disc FIOT	8.4
	8-6.6.6 Disc Files	8.4
	8-6.6.7 Disc Status Response	8.4
	8-6.6.8 DMA Magnetic Tape – General	8.4
	8-6.6.9 DMA Magnetic Tape FIOT	8.4
	8-6.6.10 DMA Magnetic Tape Status	8.4
	8-6.6.11 DMA Magnetic Tape Record Chaining Feature	8.4
	8-6.7 DIO Magnetic Tape Driver	8.4
	8-6.7.1 Purpose	8.4
	8-6.7.2 Usage	8.4
	8-6.7.3 OPEN	8.4
	8-6.7.4 DOIO – Read or Write a Record	8.4
	8-6.7.5 STAT – Status	8.4
	8-6.7.6 STUS – Device Status	8.4
	8-6.7.7 WEOF – Write End-of-File	8.4
	8-6.7.8 SEOF – Seek End-of-File	8.4
	8-6.7.9 BKSP – Backspace a Record	8.4
	8-6.7.10 WKSP – Write Skip	8.4
	8-6.7.11 REWD – Rewind	8.4

TABLE OF CONTENTS (Cont.)

		Page
8-6.7.12	Special Functions	8.46
8-6.7.13	Device Status Word	8.46
8-6.7.14	Data Format	8.46
8-6.7.15	Error Processing	8.46
8-6.7.16	Method	8.47
8-6.7.17	Program Restrictions and Space Used	8.47
8-6.8	Digital Plotter Driver	8.47
8-6.8.1	Purpose	8.47
8-6.8.2	Usage	8.47
8-6.8.3	File Input/Output Table	8.48
8-6.8.4	Method	8.48
8-6.8.5	Space Used	8.49
8-6.9	Plotter Interface Routine	8.49
8-6.9.1	Purpose	8.49
8-6.9.2	Usage	8.49
8-6.9.3	Method	8.49
8-6.9.4	Space Used	8.49
8-6.10	Adding User Created I/O Drivers	8.50
8-6.10.1	Method 1	8.50
8-6.10.2	Method 2	8.50
8-6.10.3	Method 3	8.51
8-6.10.4	User Driver Residency	8.53
8-6.10.5	User Driver Stack Information	8.54
8-6.10.6	Primary User Driver Functions	8.54
8-7	OPERATING SYSTEMS	8.59
8-7.1	Introduction	8.59
8-7.1.1	Program Development Control	8.59
8-7.1.2	Job Control	8.59
8-7.1.3	Data Control	8.62
8-7.2	BASIC Operating System	8.62
8-7.2.1	Paper Tape Input/Output System (PTIOS)	8.62
8-7.3	STANDARD Operating System	8.63
8-7.3.1	SOS Resident Monitor	8.64
8-7.3.2	SOS System Library	8.64
8-7.3.3	SOS XRAY Executive	8.65

TABLE OF CONTENTS (Cont.)

Section	Pa		
8	8-7.4	Magnetic Tape Operating System	8.6
	8-7.4.1	MTOS Resident Monitor	8.6
	8-7.4.2	MTOS System Library	8.6
	8-7.4.3	MTOS Queue/Loader Processor	8.6
	8-7.4.4	MTOS XRAY Executive	8.6
	8-7.4.5	MTOS System Configurations	8.6
	8-7.5	Real-Time Operating System	8.6
	8-7.5.1	RTOS Resident Monitor	8.6
	8-7.5.2	RTOS Disc Based Library	8.6
	8-7.5.3	RTOS Queue/Loader Processor	8.6
	8-7.5.4	RTOS XRAY Executive	8.7
	8-7.5.5	RTOS Residence and Interrupt Connection	8.7
	8-7.6	Multiprogramming System (MPS)	8.7
	8-7.6.1	Concurrent Programming	8.7
	8-7.6.2	System Control	8.7
	8-7.6.3	System Protection	8.7
	8-7.6.4	Input/Output System	8.7
	8-7.6.5	Batch Processor	8.7
	8-7.6.6	MPS-RTOS Comparison	8.7
	8-8	UTILITY PROGRAMS	8.7
	8-8.1	Math Library	8.7
	8-8.1.1	Ward Formats	8.7
	8-8.1.2	Math Library Storage Pool	8.7
	8-8.1.3	Calling Math Subroutines	8.7
	8-8.1.4	Loading the Math Subroutines	8.7
	8-8.2	Editors	8.7
	8-8.2.1	Symbolic Program Editor-BASIC	8.7
	8-8.2.2	Symbolic Program Editor	8.7
	8-8.2.3	System Editor	8.7
	8-8.3	Debug and Trace Programs	8.8
	8-8.3.1	TRACE/DEBUG	8.8
	8-8.3.2	MPS DEBUG	8.8
	8-8.4	SORT/MERGE Program	8.8
	8-8.4.1	Method of Operation	8.8
	8-8.4.2	SORT/MERGE Rates	8.8

TABLE OF CONTENTS (Cont.)

Section		Page
8	8-8.5 System Generation	8.81
	8-8.5.1 General	8.81
	8-8.5.2 Disc Organization	8.81
	8-8.5.3 Processor Library	8.82
	8-8.5.4 Relocatable Library	8.82
	8-8.5.5 SYSGEN 1 and SYSGEN 2 Operation	8.82
	8-8.5.6 Relocatable Library Extension	8.84
	8-8.6 SENSOR Diagnostic Package	8.84
	8-8.6.1 General	8.84
	8-8.6.2 Contents of SENSOR	8.84
	8-8.6.3 The Diagnostic Process	8.85
	8-8.6.4 Example Diagnostic Procedure	8.85
PROGRAMMING SYSTEMS OPERATION		
9	9-1 GENERAL	9.1
	9-1.1 Software Shipping Manifest	9.1
	9-1.2 Paper Tapes and Card Decks	9.1
	9-1.2.1 Part Numbers	9.1
	9-1.2.2 Revision Letters	9.1
	9-1.3 Part Number by Configuration	9.2
	9-2 BASIC SYSTEM SOFTWARE	9.5
	9-2.1 Initial Loader-Paper Tape	9.5
	9-2.1.1 Initial Loader Operation	9.5
	9-2.1.2 Loading Errors	9.5
	9-2.2 Initial Loader - Cards	9.5
	9-2.2.1 Initial Loader Operation	9.5
	9-2.2.2 Loading Errors	9.5
	9-2.3 Initial Loader Bootstrap	9.6
	9-2.4 XRAY EXEC-BASIC	9.7
	9-2.5 Relocating Loader - BASIC	9.7
	9-2.5.1 Initialize Load	9.7
	9-2.5.2 Continue Load	9.7
	9-2.5.3 Entry Names Table Printout	9.7
	9-2.5.4 Execute	9.8

TABLE OF CONTENTS (Cont.)

Section	Page		
9	9-2.5.5	Termination of Loading	9.8
	9-2.5.6	Relocatable Loader Error Messages	9.8
	9-2.6	SYM-I/PREP Operation	9.9
	9-2.6.1	SYM-I/PREP Initialization	9.9
	9-2.6.2	PREP Mode Operation	9.9
	9-2.6.3	Assemble From Paper Tape	9.9
	9-2.6.4	Assemble From Keyboard into Memory	9.9
	9-2.7	Symbolic Program Editor - BASIC	9.21
	9-2.7.1	Operation of the Editor	9.21
	9-2.7.2	Description of Editor Directives	9.22
	9-2.7.3	Paper Tape Format	9.23
	9-2.7.4	Operating Instructions	9.23
	9-2.7.5	Example of Program Modification	9.24
	9-2.8	Conversational FORTRAN Operation	9.27
	9-2.8.1	Conversational FORTRAN Initialization	9.27
	9-2.8.2	Conversational FORTRAN Prepare Paper Tape	9.27
	9-2.8.3	Conversational FORTRAN Edit	9.27
	9-2.8.4	Conversational FORTRAN Compile and Go	9.27
	9-2.8.5	Conversational FORTRAN Run-Time Execution	9.27
	9-3	STANDARD OPERATING SYSTEM OPERATION	9.43
	9-3.1	General	9.43
	9-3.2	Bootstrap Loading the System	9.43
	9-3.3	System Processor Loading	9.43
	9-3.4	Loading the System Executive	9.43
	9-3.5	Loading a Relocatable Program	9.44
	9-3.6	Manual Intervention	9.44
	9-3.6.1	System Recovery	9.44
	9-3.6.2	Exit Current Task	9.44
	9-3.6.3	Relocatable Program Restart	9.44
	9-3.6.4	Absolute Load Routine	9.44
	9-3.7	System Messages	9.45
	9-3.7.1	I/O Not Ready Message	9.45
	9-3.7.2	SYM-II Read/Write Error Message	9.45
	9-3.7.3	Processor Pause Message	9.45
	9-3.7.4	FORTRAN Pause Message	9.46
	9-3.7.5	System Ready Message	9.46
	9-3.7.6	Assembler in Manual Mode Message	9.46

TABLE OF CONTENTS (Cont.)

Section	Page		
9	9-4	MAGNETIC TAPE OPERATING SYSTEM OPERATION	9.49
	9-4.1	General	9.49
	9-4.2	Bootstrap Loading the System	9.49
	9-4.3	Operator Interrupt	9.49
	9-4.3.1	Operator Interrupt Directives	9.49
	9-4.4	Manual Intervention	9.50
	9-4.4.1	System Recovery	9.50
	9-4.4.2	Exit Current Task	9.50
	9-4.4.3	Relocatable Program Restart	9.50
	9-4.4.4	Absolute Load Routine	9.50
	9-4.5	System Messages	9.50
	9-4.5.1	I/O Not Ready Message	9.50
	9-4.5.2	Error Message	9.51
	9-4.5.3	SYM-II Read/Write Error Messages	9.51
	9-4.5.4	Processor Pause Message	9.51
	9-4.5.5	FORTTRAN Pause Messages	9.51
	9-4.5.6	System Ready Messages	9.51
	9-4.5.7	XRAY Ready Message	9.51
	9-4.5.8	Assembler in Manual Mode Message	9.51
	9-5	REAL-TIME OPERATING SYSTEM OPERATION	9.55
	9-5.1	General	9.55
	9-5.2	Bootstrap Loading the System	9.55
	9-5.3	Operator Interrupt	9.55
	9-5.3.1	Operator Interrupt Directives	9.55
	9-5.4	Manual Intervention	9.56
	9-5.4.1	System Recovery	9.56
	9-5.4.2	Exit Current Task	9.56
	9-5.4.3	Relocatable Program Restart	9.56
	9-5.4.4	Absolute Load Routine	9.56
	9-5.5	System Messages	9.56
	9-5.5.1	I/O Not Ready Message	9.57
	9-5.5.2	Error Message	9.57
	9-5.5.3	System Load Problem Message	9.57
	9-5.5.4	SYM-II Read/Write	9.57
	9-5.5.5	Processor Pause Message	9.57
	9-5.5.6	FORTTRAN Pause Message	9.57

TABLE OF CONTENTS (Cont.)

Section	Page
9	<ul style="list-style-type: none"> 9-5.5.7 System Ready Messages 9.58 9-5.5.8 Real-Time XRAY Ready Message 9.58 9-5.5.9 Queue Filled Message 9.58 9-5.5.10 Assembler in Manual Mode Message 9.58 9-5.5.11 System Installation Message 9.58
9-6	MULTIPROGRAMMING SYSTEM OPERATION 9.61
9-6.1	General 9.61
9-6.2	Bootstrap Loading the System 9.61
9-6.2.1	MPS Initialization 9.61
9-6.3	System Control 9.61
9-6.4	System Messages 9.61
9-6.4.1	I/O Not Ready Message 9.62
9-6.4.2	Error Message 9.62
9-6.4.3	System Load Problem Message 9.62
9-6.4.4	SYM-II Read/Write Error Message 9.62
9-6.4.5	Processor Pause Message 9.62
9-6.4.6	FORTRAN Pause Message 9.63
9-6.4.7	System Ready Message 9.63
9-6.4.8	Assembler in Manual Mode Message 9.63
9-6.4.9	Initialization Error Message 9.63
9-6.4.10	Processor Map Read Error Message 9.63
9-6.4.11	Swap Levels Too Large Message 9.63

TABLE OF CONTENTS (Cont)

APPENDICES

Section		Page
	K-6 BASIC XRAY Executive Program and I/O Monitor Characteristics	K.15
	K-7 Paper Tape Input/Output System (PTIOS) Characteristics	K.16
	K-8 Symbolic and System Editor Characteristics	K.17
	K-9 TRACE/DEBUG and MPS DEBUG Characteristics	K.18
L	I/O OPERATING STATUS AND SELECTION SUMMARY	L.1
	Device I/O Selection Codes	L.1
	Device I/O Operating Status	L.3
M	USER'S SOFTWARE OPERATING SUMMARY AND FUNCTIONAL CHARACTERISTICS	M.1
	M-1A BASIC System Software Part Numbers	M.2
	M-1B STANDARD System Software Part Numbers	M.3
	M-1C Magnetic Tape Operating System Software Part Numbers	M.4
	M-1D EXTENDED Operating System Software Part Numbers	M.5
	M-1E ADVANCED Operating System Software Part Numbers	M.6
	M-2 XRAY Directive Dictionary	M.7
	M-2A XRAY-BASIC System Directives	M.9
	M-2B XRAY Operating System Directives	M.16
N	ASSEMBLER/COMPILER ERROR REVIEW	N.1
	N-1 SYM-I Error Messages	N.1
	N-2 SYM-II Error Messages	N.2
	N-3 Real-Time FORTRAN IV Compile Time Error Messages	N.3
	N-4 Real-Time FORTRAN IV Run-Time Error Messages	N.6
	N-5 Conversational FORTRAN Compile Time Error Messages	N.8
	N-6 Conversational FORTRAN Run-Time Error Messages	N.9
	N-7 BASIC Relocatable Loader Error Messages	N.9
	N-8 STANDARD and Disc Relocatable Loader Error Messages	N.10
	N-9 RTOS Error Messages	N.11
	N-10 SYSGEN 1 Error Messages	N.12
	N-11 SYSGEN 2 Error Messages	N.14
O	MNEMONIC INDEX OF MACHINE INSTRUCTIONS	O.1
P	OP CODE INDEX OF MACHINE INSTRUCTIONS	P.1
Q	FUNCTIONAL INDEX OF MACHINE INSTRUCTIONS	Q.1
R	INDEX OF PSEUDO INSTRUCTIONS	R.1
INDEX	Index - 1

TABLE OF CONTENTS (Cont)

APPENDICES

Section	Page
A	GLOSSARY A.1
B	APPLICABLE PUBLICATIONS INDEX B.1
C	POWERS OF 2 AND 16 C.1
D	MISCELLANEOUS REFERENCE DATA D.1
	Common Constants D.1
	Factorials D.1
	Reciprocals D.2
	Square Roots D.3
	Miscellaneous Constants D.3
E	HEXADECIMAL-TO-DECIMAL CONVERSION E.1
	Hexadecimal Number System E.2
	Integer Table Extension E.3
	Hexadecimal-to-Positive Integer Conversion Table E.4
	Hexadecimal-to-Negative Integer Conversion Table E.8
	Hexadecimal-to-Decimal Fraction Table E.12
F	TWO'S COMPLEMENT ARITHMETIC F.1
G	CHARACTER CODES G.1
	ASCII (Teletypewriter) Char. Codes, Hexadecimal and Hollerith Punch Equivalents G.1
	ASCII-8 (IBM) Character Codes and Hexadecimal Equivalents G.2
	Line Printer Character Set G.3
H	IMPLEMENTATION AND INSTALLATION INFORMATION H.1
I	TIMING CONVERSION I.1
J	MATH LIBRARY SUMMARY J.1
	Timing, Accuracy, and Storage J.1
	Math Routine, Part No., and Purpose J.4
	Calling Sequence, Argument Description and Error Conditions J.4
	FORTRAN IV Library Functions J.14
	FORTRAN IV Run-Time Library J.17
	Conversational FORTRAN Library Functions J.35
	FORTRAN ATP Function Calls J.36
K	SOFTWARE CHARACTERISTICS SUMMARY K.1
	K-1 SYM-I/PREP Characteristics K.1
	K-2 SYM-II Characteristics K.4
	K-3 Raytheon 706 FORTRAN vs USASI STANDARD K.8
	K-4 Real-Time FORTRAN IV Characteristics K.10
	K-5 Conversational FORTRAN Characteristics K.13

TABLES

Table		Page
1-1	706 Computer Specifications	1.4
1-2	CPU Register Description.	1.8
2-1	Definition of Instruction Equation Terms and Symbols	2.2
3-1	Interrupt Ranking and Assignment.	3.3
4-1	Direct Input/Output Channel Signals.	4.3
4-2	DMA Channel Signals	4.9
4-3	DIO/DMA Wire, Electrical, and Cable Specifications	4.10
4-4	Direct I/O Connectors J1, J2	4.11
4-5	Direct Memory Access Connector J5	4.12
5-1	Device/Function Codes and Operations	5.1
5-2	Peripheral Data Bit Assignment	5.8
5-3	Peripheral Status Bit Assignment and Meaning	5.9
5-4	DIO Status Test Method Coding Example.	5.15
5-5	DIO Burst Method Coding Example	5.15
5-6	DIO Interrupt Method Coding Example	5.19
5-7	DMA Read/Write Selection Instruction Sequence	5.23
5-8	IOS Function Table	5.27
5-9	PEAT Table	5.28
5-10	System Linkage Table	5.34
5-11	SYM I Sample Program	5.35
5-12	SYM II Sample Program	5.36
5-13	ASR-33 ASCII & ASCII-8 Character Codes	5.40
5-14	ASR-33 Commands and Function Codes.	5.41
5-15	ASR-33 Status	5.42
5-15.1	Teletype Console Operation.	5.42
5-16	High Speed Paper Tape Reader and Punch Commands and Function Codes	5.45
5-17	High Speed Paper Tape Reader and Punch Status	5.46
5-18	Card Row-Accumulator Bit Correspondence	5.49
5-19	Card Reader and Card Punch Commands and Function Codes	5.51
5-20	Card Reader and Card Punch Status	5.52
5-21	Punching Speeds Using High Speed Skip Feature	5.53
5-22	Card Reader (1000 CPM) Controls and Indicators	5.56
5-23	Card Punch Controls and Indicators	5.57
5-23.1	Card Reader (400 CPM) Controls and Indicators	5.60
5-23.2	Line Printer Specifications	5.61
5-24	Line Printer Character Set	5.62

TABLES (Cont)

Table		Page
5-25	Line Printer Commands and Function Codes	5.65
5-26	Line Printer Status	5.64
5-26.1	Drum, Paper and Print Speeds	5.66
5-27	Line Printer Operating Controls and Indicators (360 and 600 lpm).	5.69
5-27.1	Line Printer Operating Controls and Indicators (1000 lpm).	5.71
5-28	Disc Commands and Function Codes	5.76
5-29	Disc Status	5.76
5-30	Disc Controller Controls	5.77
5-31	DMA 9-Track Magnetic Tape System Characteristics	5.80
5-32	DMA 9-Track Magnetic Tape Function Codes.	5.87
5-33	DMA 9-Track Magnetic Tape Status	5.88
5-34	DMA 9-Track Magnetic Tape Interrupt Meaning.	5.89
5-35	DMA 9-Track Tape Unit Control Panel Controls and Indicators	5.90
5-36	DMA 7-Track Magnetic Tape System Characteristics	5.95
5-37	DMA 7-Track Magnetic Tape Function Codes.	5.102
5-38	DMA 7-Track Magnetic Tape Status	5.103
5-39	DMA 7-Track Magnetic Tape Interrupt Meaning.	5.104
5-40	DMA 7-Track Tape Unit Control Panel Controls and Indicators	5.105
5-41	DIO Magnetic Tape Characteristics	5.110
5-42	DIO Magnetic Tape Speed-Density Review	5.111
5-43	DIO Magnetic Tape Device Assignment Codes.	5.111
5-44	DIO Magnetic Tape Commands and Function Codes	5.115
5-45	DIO Magnetic Tape Status	5.116
5-46	DIO Magnetic Tape Unit Controls and Indicators	5.119
5-47	Digital Plotter System Characteristics	5.123
5-48	Carriage and Drum Execution Time	5.125
5-49	Commands and Function Codes for Digital Plotter	5.127
5-50	Digital Plotter Status.	5.128
5-51	Digital Plotter (Model 565) Controls and Indicators.	5.132
5-52	Teletype Multiplexer Function Codes	5.137
5-53	Teletype Multiplexer Status	5.137
5-54	Buffered Digital Input Channel Characteristics	5.139
5-55	Buffered Digital Output Channel Characteristics.	5.140
5-56	BIC and BOC Commands and Function Codes	5.141
5-57	BIC and BOC Status	5.142
5-58	Miniverter Specifications	5.144

TABLES (Cont.)

Table	Page
5-59	Multiverter Specifications 5.145
5-59.1	Multiverter Throughput Rates 5.145
5-60	Available Output Codes 5.146
5-61	Analog to Digital Converter Function Codes 5.149
5-62	Analog to Digital Converter Status 5.149
5-63	Miniverter Controls and Indicators 5.151
5-64	Multiverter Controls and Indicators 5.152
5-65	Digital to Analog Converter Specifications 5.153
5-66	Digital to Analog Input Codes, Analog Values, Output Voltages 5.155
5-67	Time-of-Day Clock Specifications 5.157
5-68	Time-of-Day Clock Commands and Function Codes 5.160
5-69	Interval Timer Input Jumper Connections 5.161
5-70	Time-of-Day Clock Controls and Indicators 5.162
6-1	Exits from User State 6.5
8-1	706 Hardware System Configurations 8.2
8-2	706 Software System Configurations 8.3
8-3	Pseudo Operation Summary 8.9
8-4	Relocatable Loader Text Codes and Formats 8.25
8-5	MPS - RTOS Comparison Chart 8.74
8-6	Math Library Functions and Subroutines Part Numbers 8.76
8-7	External/Internal Word Formats 8.77
8-8	Math Library Example Program 8.79
9-1	BASIC System Software 9.4
9-2	STANDARD Operating System Software 9.42
9-3	Magnetic Tape Operating System Software 9.48
9-4	Real-Time Operating System Software 9.54
9-5	Multiprogramming System Software 9.61

ILLUSTRATIONS

Figure		Page
1-1	706 Hardware Block Diagram	1.2
1-2	706 Software Block Diagram	1.3
1-3	Raytheon 706 Central Processor Unit (CPU)	1.7
1-4	Instruction Word Formats	1.1
1-5	Hardware Data Word Formats	1.1
1-6	Software Data Word Formats	1.1
1-7	Status Word Formats	1.1
1-8	Address Word Formats	1.1
1-9	Assembler Word Formats and Syntax	1.1
1-10	Memory Layout	1.1
1-11	Word and Byte Page Number Conversion	1.1
1-12	Word vs. Byte Addressing	1.1
1-13	Direct Word Addressing Example	1.1
1-14	Direct Byte Addressing Example	1.1
1-15	Extension Register Operation	1.2
1-16	Direct Word Addressing Operation	1.2
1-17	Direct Byte Addressing Operation	1.2
1-18	Direct Memory Addressing Example	1.2
1-19	Index Register Operation	1.2
1-20	Local Mode Word Indexing Operation	1.2
1-21	Local Mode Byte Indexing Operation	1.2
1-22	Local Mode Indexing Example	1.2
1-23	Global Mode Word Indexing Operation	1.2
1-24	Global Mode Byte Indexing Operation	1.2
1-25	Global Mode Indexing Example	1.2
1-26	Single Level Indirect Addressing	1.2
1-27	Multi-Level Indirect Addressing	1.2
1-28	BASIC Hardware Configuration	1.3
1-29	STANDARD Hardware Configuration	1.3
1-30	EXTENDED Hardware Configuration	1.3
1-31	ADVANCED Hardware Configuration	1.3
1-32	Fully Expanded Hardware Configuration	1.3
3-1	Interrupt Operation for Level X	3.2
4-1	Direct Input/Output System	4.2
4-2	Direct Input Operation	4.4
4-3	DIO Interface Timing	4.5

ILLUSTRATIONS (Cont)

Figure		Page
4	Direct Output Operation	4.6
5	Direct Memory Access System	4.8
6	DMA Interface Timing	4.10
7	Cable Termination Scheme	4.13
8	Cabling of Peripheral Devices	4.14
1	DIO Status Test Input Routine Flow	5.14
2	DIO Interrupt Method Block Flow Diagrams	5.16
3	DIO Interrupt Main Flow Diagram	5.17
4	DIO Interrupt Service Routines Flow Diagram	5.18
5	Eight Channel Paper Tape Data Format	5.40
6	Teletype Interrupt Timing	5.42
7	High Speed Paper Tape Reader and Punch Interrupt Timing	5.46
8	Card Reader and Card Punch Timing	5.53
9	Card Reader (1000 CPM) and Card Punch Controls and Indicators.	5.55
9.1	Card Reader (400 CPM) Control Panel	5.60
10	Slew Characters for Line Printer.	5.63
10.1	Line Printer Timing	5.67
11	Printer Control Panel (360 and 660 lpm)	5.69
11.1	Printer Control Panel (1000 lpm)	5.71
12	Disc Controller Controls	5.77
13	DMA 9-Track Magnetic Tape Unit Control Panel	5.90
14	DMA 7-Track Word Formats.	5.96
15	DMA 7-Track Magnetic Tape Unit Control Panel	5.105
16	DIO Magnetic Tape Data Words	5.113
17	DIO Magnetic Tape Timing for Reading.	5.118
17.1	DIO Magnetic Tape Timing for Writing	5.118
18	Tape Path and Control Panel (5000 Series)	5.121
19	Tape Path and Control Panel (6000 Series)	5.122
20	8-Vector Format	5.125
21	Pen, Drum, and Carriage Control Codes	5.126
22	Vector Command Codes	5.127
23	Digital Plotter Timing	5.129
24	Digital Plotter (Model 565) Control Panel.	5.131
25	Teletype Multiplexer Input Word Formats.	5.136
26	Teletype Multiplexer Timing.	5.138
27	BIC and BOC Timing	5.142

ILLUSTRATIONS (Cont.)

Figure		Page
5-28	Analog to Digital Word Formats	5.14
5-29	Analog to Digital Timing	5.15
5-30	Miniverter Control Panel	5.15
5-31	Multiverter Control Panel	5.15
5-32	Digital to Analog Converter Data Word Format	5.15
5-33	Time-of-Day Clock Data Word Format	5.15
5-34	Time-of-Day Clock Control Panel	5.16
6-1	Memory Protect Option	6.4
7-1	706 Console	7.2
7-2	Bootstrap Memory Map	7.5
8-1	The Assembly Process	8.5
8-2	Assembly Language Coding Form	8.7
8-3	FORTRAN IV Compilation	8.14
8-4	Executing a FORTRAN IV Object Deck	8.15
8-5	FORTRAN IV Compile and Execute (GO)	8.15
8-6	Conversational FORTRAN in Interactive Use	8.17
8-7	Conversational FORTRAN in Batch Process Use	8.17
8-8	Paper Tape Absolute Program Format — SYMI, II	8.20
8-9	Card Absolute Program Format	8.21
8-10	Absolute Bootstrap Program	8.23
8-11	BASIC System Loader Map	8.27
8-12	STANDARD and MTOS System Loader Map	8.28
8-13	EXTENDED and ADVANCED System Loader Map	8.29
8-14	Teletype Driver Record Formats	8.31
8-15	Card Reader Driver Record Formats	8.36
8-16	Adding User's Driver — Method 2	8.51
8-17	Adding User's Driver — Method 3	8.52
8-18	SAVE Flowchart	8.55
8-19	REST Flowchart	8.55
8-20	INTENB Flowchart	8.56
8-21	DRIVEX Flowchart	8.57
8-22	M.DDR Flowchart	8.58
8-23	Job Control Under a Basic Batch Processing Operating System	8.60
8-24	Job Control Under a Queued Sequential Batch Processing Operating System	8.61
8-25	MTOS Minimum Configuration	8.67

ILLUSTRATIONS (Cont.)

Figure		Page
26	MTOS Expanded Configuration	8.67
27	MPS Memory Layout	8.72
28	RTOS Disc Layout	8.83
29	MPS Disc Layout	8.83
30	System Checkout Flow Diagram	
	Sheet 1 of 6	8.86
	Sheet 2 of 6	8.87
	Sheet 3 of 6	8.88
	Sheet 4 of 6	8.89
	Sheet 5 of 6	8.90
	Sheet 6 of 6	8.91
31	Sensor Diagnostic Detail Example	8.92
-1	Software Shipping Manifest Example	9.2
-2	Initial Loader Bootstrap	9.6
-3	SYM-I/PREP Initialization	9.11
-4	SYM-I/PREP Mode Operation	9.12
-5	SYM-I Assemble From Paper Tape	9.14
-6	SYM-I Assemble From Keyboard into Memory	9.16
-7	SYM-I/PREP Input Listing	9.18
-8	SYM-I/PREP Listing	9.19
-9	SYM-I Object Listing	9.20
-10	Symbolic Program Editor Example	9.25
-11	Conversational FORTRAN Initialization	9.28
-12	Conversational FORTRAN Prepare Paper Tape	9.30
-13	Conversational FORTRAN Edit Paper Tape	9.32
-14	Conversational FORTRAN Compile and Go	9.34
-15	Conversational FORTRAN Run-Time Execution	9.36
-16	Conversational FORTRAN Input Listing	9.38
-17	Conversational FORTRAN PREP Listing	9.38
-18	Conversational FORTRAN Compiler Listing	9.39
-19	Conversational FORTRAN Run-Time and Trace	9.40

INTRODUCTION TO THE MANUAL

The reader of this manual should be familiar with the operation of small digital computers. The manual presents an overall view of the 706 computer hardware and software. Hardware and software descriptions have been integrated throughout the manual wherever possible. Categories are specified hardware or software when the possibility of an ambiguity exists. This manual does not replace either the *Technical Operations and Maintenance Manual* or detailed software manuals.

Specific reference to other Raytheon Computer collateral material has been avoided, however. Appendix B, *Applicable Publications Index*, lists support documentation that can be obtained from Raytheon Computer.

This manual is a teaching aid for students of the 706 computer as well as a reference manual for persons already familiar with its operation and use.

NOTE

There may be changes to either the hardware or software not reflected in this manual. Errata and addendum sheets are issued periodically between edition printings.

Please use the form in the back of the manual to notify Raytheon Computer of any errors you have detected in the manual.

Section 1 – General Systems Description should be read by everyone prior to reading the rest of the manual. It is particularly important to understand the sections on Word Formats (1-3), Addressing (1-4), Input/Output (1-5) and the software synopsis given in Section 1-6.

Section 2 – Instructions describes the operation of the machine hardware instructions and the assembler pseudo instructions. The instruction descriptions are given in assembler breakdown order. Appendices O, P, Q, and R provide indices to Section 2.

Section 3 – Interrupt, *Section 4 – Input/Output* are prerequisites to understanding Section 5 on peripheral equipment operation. These two sections describe the operation and architecture of the 706 Input/Output System. The hardware interface is also discussed in Section 4.

Section 5 – Peripheral Equipment Operation and Programming reviews Direct Input/Output and Direct Memory Access programming techniques. The 706 Monitor Input/Output Software system is described here rather than in Section 8 on programming systems. The remainder of the section is devoted to describing the characteristics of standard 706 peripheral controllers and devices.

Section 6 – Options explains the optional features of the 706 Central Processing Unit and Memory.

Section 7 – Operation describes the functions of the Central Processing Unit Control Panel. The operations of turning power on and off, displaying and changing the contents of registers and memory, and program execution including the bootstrap are discussed.

Section 8 – Programming Systems Description familiarizes the user with the four hardware and five operational systems available with the 706 computer. The majority of this section describes the characteristics of Raytheon Computer assemblers, compilers, loaders, operating systems, and utility programs. This section represents a short course in small computer software orientation

The operation of the I/O Drivers are described in detail. This description in conjunction with the Input/Output Software description (Section 5) eliminates the coding of I/O operations in machine language.

Section 9 – Programming Systems Operation is a review of the operating procedures for major 706 programs. The section is divided into four areas that correspond to the four hardware system configurations.

The Appendices provide quick reference information. Of particular interest are the conversion tables of Appendices C, D, E, and F. Appendix N gives a comprehensive *error message* review of nearly all Raytheon Computer software. The *Math Library Summary* of Appendix J shows all pertinent characteristics of the *Math Library*, the *FORTRAN IV Library*, and the *Conversational FORTRAN Library*

Section 1

GENERAL SYSTEM DESCRIPTION

1 INTRODUCTION

The Raytheon 706 computer system is a general purpose digital system designed primarily as the central element for data acquisition processing and control systems. The important hardware characteristics of the 706 are:

ARDWARE FEATURES

Standard

- 16-bit word length.
- Memory expandable to 32K
- Byte and word addressing and instructions.
- 2048 word memory pages.
- Direct and indexed addressing.
- 74 hardware instructions.
- 900 nanosecond cycle time.
- Automatic priority interrupt.
- Operator console with lockout.
- Hardware bootstrap.
- Direct input/output to central processor.
- T²L integrated circuits.

Optional

- Direct memory access – word transfer rates to 1.1 MHz.
- Power fail-safe with automatic restart.
- Hardware Multiply/Divide.
- Memory parity.
- Memory protect – User/Executive mode, privileged instructions.
- Full line of peripheral options.

The Raytheon 706 software is tailored to the complete range of hardware configurations. The 706 program library includes over 300 programs and subroutines, containing more than 200,000 instructions. The important software characteristics of the 706 are:

SOFTWARE FEATURES

- X-Ray Executives.
- Real-time operating system.
- Multi-programming system.
- Conversational FORTRAN.
- Real-time FORTRAN IV (ASA superset).
- File oriented, device independent input/output system.
- 1-pass and 2-pass relocatable assemblers.
- True-isolation hardware diagnostics.
- Batch processing.
- Extensive utility programs.
- Foreground/background processor.
- Fixed and floating point math routines.
- Multi-precision word formats.
- Modularly expandable software.

1-1.1 System Description and Configuration

The 706 basic hardware configuration (Figure 1-1) includes a central processor unit (CPU), a 4K memory system, an ASR-33 teletype, and a Direct Input/Output (DIO) bus. Up to 16 peripheral controllers may be attached to the DIO bus including high speed paper tape, card reader/punch, line printer, X-Y plotter, disc memory, magnetic tape, time-of-day clock, etc. Up to six Direct Memory Access (DMA) devices may transfer data to and from memory directly without

CPU intervention. Table 1-1 gives the computer hardware specifications.

The 706 software capability (Figure 1-2) allows the user to compile and/or assemble source application programs to produce object programs. The user may then load and execute these object programs using the trace/debug capability to detect any errors. The object and source programs can be edited and the error-free program inserted into the system library. All of the aforementioned

functions are under control of an executive program that automatically controls the system operation according to user directives entered the ASR-33 teletype keyboard on paper tape or punched cards.

The remainder of this manual is dedicated explaining and illustrating all of the facts necessary for successful use of the Raytheon 706 computer system.

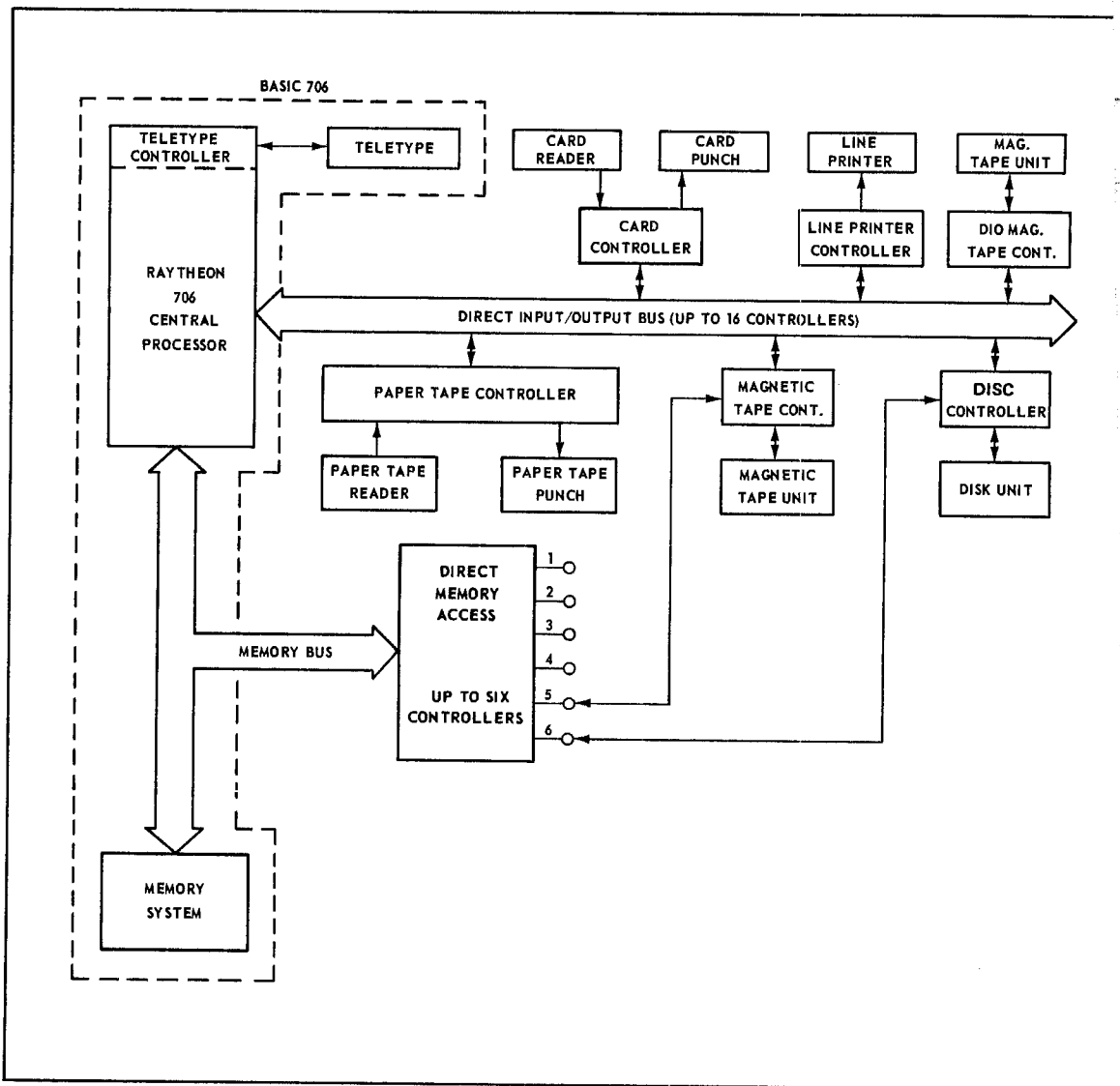


Figure 1-1. 706 Hardware Block Diagram

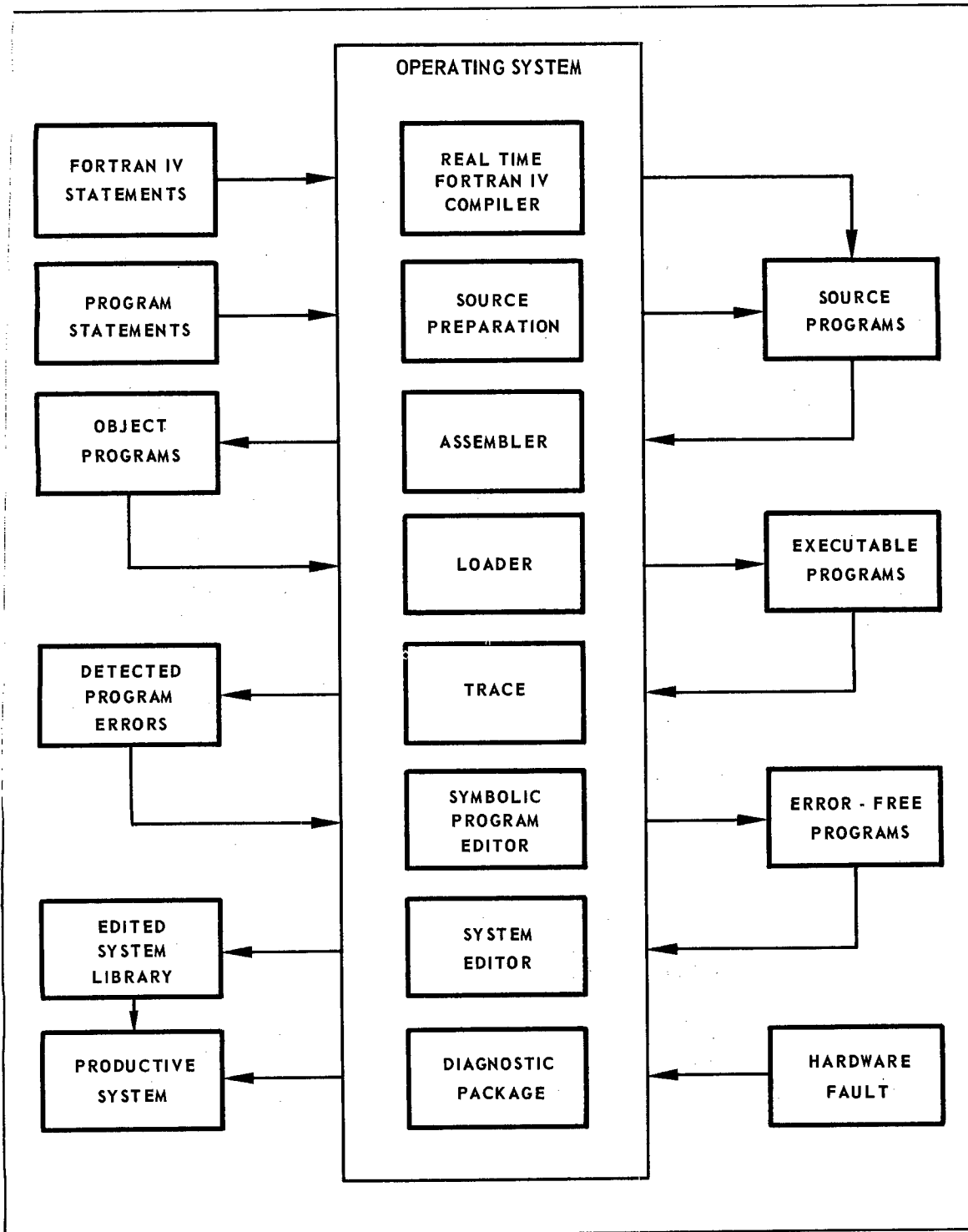


Figure 1-2. 706 Software Block Diagram

Table 1-1. 706 Computer Specifications

<p>FUNCTIONAL</p> <p>Cycle Time 900 nanoseconds</p> <p>Number System Binary</p> <p>Word Length 16 bits</p> <p>Byte Length 8 bits</p> <p>Instructions 74 standard</p> <p> Memory Reference ... 13</p> <p> Index 4</p> <p> Shift 20</p> <p> Test/Jump 18</p> <p> Control, I/O,</p> <p> Data Generic 19</p> <p>Registers 6</p> <p> Addressable 3</p> <p> Index 1</p> <p>Memory</p> <p> Type Magnetic Core</p> <p> Size 4096 words expandable to 32,768 in 4K, 8K, or 16K modules.</p> <p> Organization Page</p> <p> Addressing Modes ... 3</p> <p> Direct Addressing ... 2048 words</p> <p> Page Addressing Extension Register</p> <p>Input/Output 256 device functions are available on standard DIO bus; 6 devices in addition to the processor can be on line to the Direct Memory Access.</p> <p>Interrupt Up to 16-levels of priority interrupt.</p>	<p>Peripheral Equipment</p> <p>Standard ASR-33 Teletype, paper tape reader and punch, 10 cps.</p> <p>Optional High Speed Paper Tape Reader, 300 cps. High Speed Paper Tape Punch, 110 cps Card Reader, 400 or 1000 cards/min. Card Punch, 100-400 cards/min. Line Printer, 315, 600, 1000 lines per minute. DMA Magnetic Tape, 9- or 7-track, 800 BPI, 36 and 75 ips. DIO Magnetic Tape, 9- or 7-track, 25 ips. Disc Memory, 385,024 words 143 KHz. Incremental Digital Plotter Real-Time Clocks. Analog Instruments. Buffered Input/Output Channels. Teletype Multiplexer.</p> <p>Options Direct Memory Access (DMA). Priority Interrupt Expansion. Memory Protect. Multiply/Divide. Memory Expansion. Memory Parity. Power Fail-Safe.</p>
---	--

Table 1-1. 706 Computer Specifications (Cont.)

<u>MECHANICAL</u>	
System Cabinet	
Size	52 inches high, 25 inches wide, 30 inches deep; 22.75 inches of spare front panel space are available for options and user requirements.
Weight	175 pounds.
Mainframe Components	
Size	CPU—5.25 inches high, 19 inches wide, 26.0 inches deep; Memory—7 inches high, 19.0 inches wide, 20.75 inches deep; Power Supply—5.25 inches high, 19.0 inches wide, 25.0 inches deep
Weight	155 pounds
Teletype	
Size	33.0 inches high, 5 21.5 inches wide, 19.0 inches deep
Weight	56 pounds
Power	
Source Power	115 volts, 60 cps, single phase, (50 cps optional).
Power Consumption	1200 watts.
Environmental	
Storage	Temperature: -20° to +65°C. Humidity: 0 to 95% RH.
Operating	Temperature: 0° to 40°C. Humidity: 0 to 89% RH.
Air Conditioning	Not required. Filtered air is forced throughout the mainframe by self-contained blowers.

1-2 Central Processing Unit

The Raytheon 706 is a fully parallel 16-bit, 900 nanosecond computer that is capable of performing both general purpose and real-time computations. The advanced building block design of the 706 supports simple, flexible expansion of the system. The basic configuration provides a word and byte oriented repertoire of 74 instructions. Included in the minimal central processor configuration are a 4K core memory, an ASR-33 teletype, a Direct Input/Output bus, one level of Automatic Priority Interrupt, an automatic 4-device hardware bootstrap, and a cabinet complete with an operator's console.

All registers are connected through a high speed adder. Major registers include the accumulator, index register, program counter, extension register, memory buffer register, and the instruction register. See Figure 1-3.

1-2.1 Register Description and Operation

A brief description of the CPU registers is given in Table 1-2. The operation of each register is described more fully in the following paragraphs.

1-2.1.1 Program Counter

The program counter (PCR) is a 15-bit register that is used to hold the address of the next instruction to be executed. During normal program execution, the PCR is incremented by one during every machine instruction fetch from memory. This causes the PCR to be pointing one location ahead of the location of the instruction about to be executed. Provided that the instruction to be executed is not a jump or skip and an interrupt does not occur, the PCR will continue to sequence one location at a time until a halt instruction is executed or until the HALT button on the control panel is pressed.

A jump instruction (JMP, JSX) causes the PCR to be set to the effective address created by the contents of the M field (bits 5-15) of the instruction, the index register and the extension register (see Section 1-3, Addressing). Depending on the condition tested, a skip type of instruction may or may not cause the PCR to be incremented by an additional one. Normally, if the condition tested for is satisfied (ie, skip on sense switch false, and the sense switch is false or off), the PCR will

be incremented. If the condition tested for is not satisfied, the PCR will not be incremented and the next instruction in sequence will be fetched.

An interrupt that is to be processed, will not alter the PCR. The PCR is saved in a fixed location (see Section 3) and a pre-stored address is loaded into the PCR causing the next instruction to come from the beginning of an interrupt subroutine.

The PCR may also be set by using the 706 console. See Section 7.

1-2.1.1 Instruction Register

The instruction register (INR) stores the eight most significant bits of the instruction for decoding. Instruction codes consist of one, two, or three hexadecimal characters (4, 8, or 12 bits). The first two characters of the instruction word (bits 0-7) are stored in the INR. The third character, if required by the instruction, is decoded from the corresponding bits of the contents of the memory buffer register.

If the first character of the instruction is anything other than a zero, it is a *memory reference* type of instruction. If the first character is a zero, the instruction is a *generic* type of instruction. See Section 1-3.1

1-2.1.3 Accumulator

The accumulator register (ACR) is a 16-bit register that operates in conjunction with the adder to form a conventional accumulator. It is also used as a shift register, either alone in single-length shifts or in conjunction with the index register in double length shifts. Input data from peripheral devices is received in the ACR, and output data is transmitted from the ACR to peripheral devices.

1-2.1.4 Memory Buffer Register

The memory buffer register (MBR) is a 16-bit register used to hold data and operand addresses that are transferred between the CPU and the memory. The MBR also holds the device address during a DIO operation.

1-2.1.5 Index Register

The index register is a 16-bit register used primarily for indexing operand addresses. See Section 1-4.5

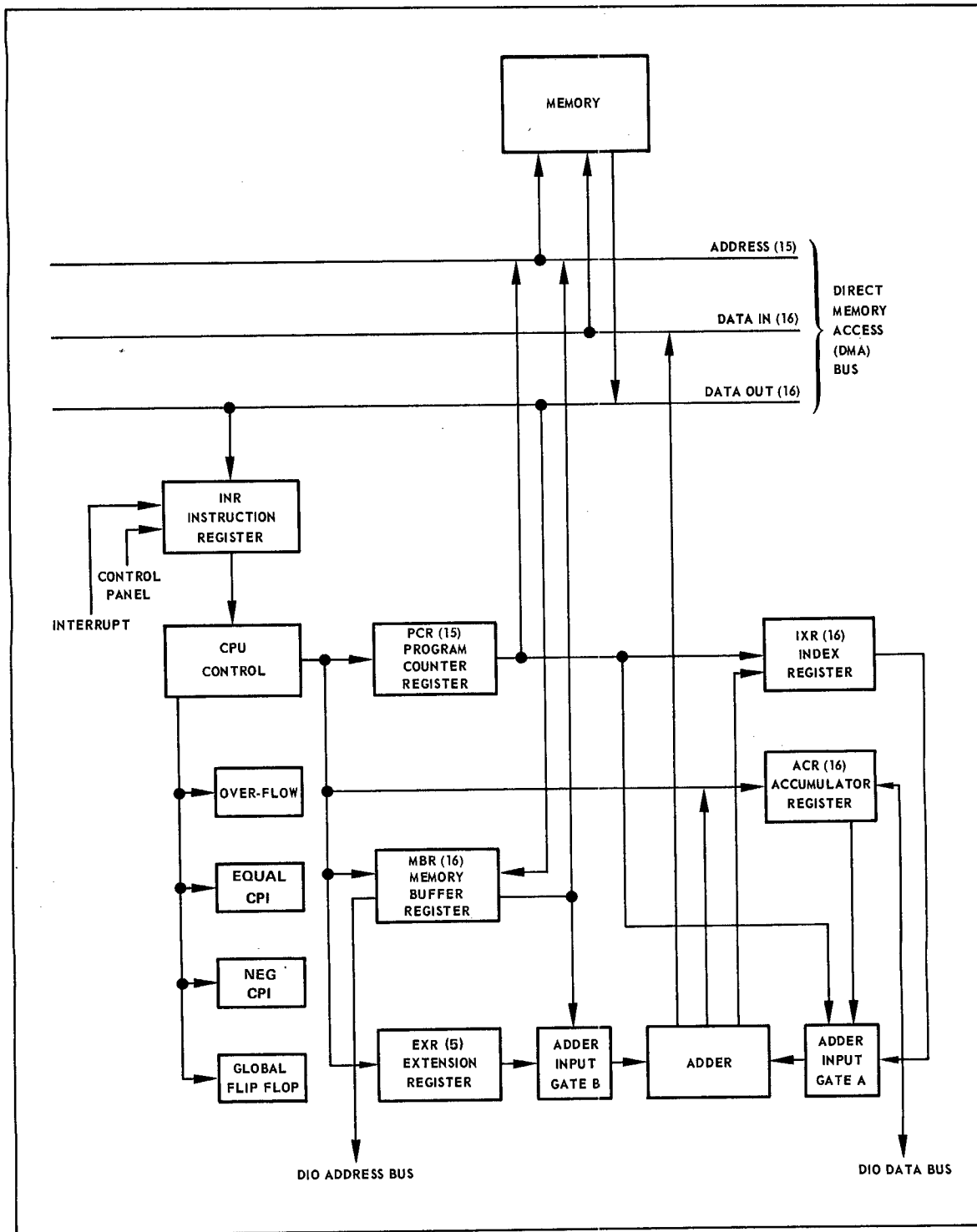


Figure 1-3. Raytheon 706 Central Processor Unit (CPU)

Table 1-2. CPU Register Description

<u>Name</u>	<u>Function</u>
Instruction Register (INR)	Stores the eight most significant bits of an instruction for decoding.
Memory Buffer Register (MBR)	A 16-bit buffer for transfer of data, instructions and operand addresses.
Accumulator ((ACR)	Temporary storage of results of all full word and byte mathematical/logical operations including Direct Input/Output.
Program Counter (PCR)	Contains 15-bit address of next instruction to be executed.
Index Register (IXR)	The 15-bit address is indexed with byte and word memory references. Also used for double precision operations and by the multiply/divide option.
Extension Register(EXR)	The 5-bit address is concatenated with the instruction operand field for extended addressing. Points to selected page.
Overflow (OVF)	Arithmetic operations and shifts with overflows or underflows set the 1-bit OVF.
Comparison Indicators (CPI)	Two bits denote results of word and byte compares (equal to, greater than, less than).
Local/Global Indicator (GLB)	1-Bit GLB indicates Local or Global addressing mode.

1-2.1.6 Extension Register

The extension register (EXR) is a 5-bit register that is concatenated with the instruction operand field to point to the selected word or byte page. See Section 1-4.3

1-2.1.7 Overflow Indicator

The overflow indicator (OVF) is set on overflow or on underflow arising as the result of executing an arithmetic operation. The OVF is reset either by testing the OVF for being "on" or performing an arithmetic operation that results in no overflow or underflow. The largest signed integer that can be held in the ACR is $+32,767_{10}$ ($2^{15}-1$). The smallest signed integer is $-32,768_{10}$ (-2^{15}).

1-2.1.8 Comparison Indicators

The comparison indicators (CPI) store the results of executing the compare ACR and memory instructions and the compare literal byte (CLB) instruction. The contents of the CPI can be tested for less than, equal to, and greater than values. The CPI are also part of the machine status word (see Section 1-3.3).

1-2.1.9 Local/Global Indicator

The local/global indicator (GLB) or the global flip-flop, as it is technically known, stores the index addressing state of the machine. The global state can be entered by executing a Set Global Mode (SGM) or a Jump and Store Return in Index (JSX) instruction or by processing an interrupt. The local mode of indexing is selected with a Set Local Mode (SLM) instruction or by pressing the RESET button on the CPU control panel. See Section 1-4.6.1 and Section 1-4.6.3.

1-2.2 Internal Number Representation

The 706 uses two's complement form arithmetic.

Unless otherwise specified, all internal fixed-point positive numbers must have a binary zero for the sign bit followed by the normal binary representation of the magnitude portion of the number. All internal fixed-point negative numbers consist of a binary one for the sign bit followed by the two's complement of the magnitude portion of the equivalent positive number.

For example:

$+12FC_{16}$ is represented internally in binary as 000100101111100. A $-12FC_{16}$ is represented internally in binary as 1110110100000100.

To convert any negative internal number to its positive external counterpart, take the two's complement (re-complement) of the negative number. See Appendix F.

1-2.3 Timing

The basic time required for performing one operation in the CPU is called a *machine cycle*. The time required to complete one machine cycle is 0.9 μ sec. Instructions that reference memory (i.e., one of the arguments for the completion of the instruction resides in core memory), take two machine cycles; one to fetch the instruction and one to fetch the argument and complete the operation. Instructions that do not reference memory (generic instructions) normally take only one machine cycle: one cycle to fetch the instruction and execute the operation. Shift operations may take up to four machine cycles while the Interrupt Return (INR) instruction takes three cycles. See the individual description for each instruction in Section 2 for the exact timing of every instruction.

1-3 WORD FORMATS

The 706 Computer Word is 16 bits in length. Bit positions within a word are numbered from left to right, bit 0 being the most significant bit and bit 15 being the least significant bit. There are four types of computer word formats: instruction words, data words, status words, and an address word. Also, of interest to the user are the assembler source statement formats.

1-3.1 Instruction Word Formats

Instruction words for the 706 are either memory type instructions or generic type instructions. See Figure 1-4. The memory reference instructions use bits 5 through 15 to directly address any one of 2048 words or bytes of a word or byte page specified in the Extension Register (EXR). Bits 0 through 3 specify the operation code (OP Code) and bit 4 indicates either direct addressing or indexed addressing. If bit 4 is a one, indexed addressing is specified; if bit 4 is a zero, direct addressing is specified.

The generic instruction has an operation code (bits 0-3) of zero and does not reference memory. The six types of generic instructions are: 4-bit operand, shift, halt, no operand, literal byte operand, and input/output.

The optional double word instructions (multiply and divide) reference memory with the second word of the instruction. Bits 1 through 15 specify the effective address of either the multiplicand or the divisor.

All instructions are further defined in Section 2 of this manual.

1-3.2 Data Word Formats

The 706 hardware treats data words as either logical words, single precision fixed point full word data, or single precision fixed point byte data. Raytheon software uses data word formats for double precision integer and multiprecision floating point data. See Figure 1-5.

1-3.2.1 Logical Word and Single Precision Fixed Point Data

A logical data word consists of 16 bits (0-15) that are treated as an unsigned word by the logical instructions (i.e., ORI, ORE, AND). Single

precision fixed point full word data is held in a 16-bit two's complement number format. Bit-0, the sign bit, is a zero for positive numbers and a one for negative numbers. The fractional or integer portion of a single precision fixed point data word is held in bits 1 through 15. The number range for full word data is

$$\text{Fractional: } -1 \leq N < +1$$

$$\text{Integer: } -2^{15} \leq N \leq +2^{15} - 1$$

Byte data is contained in two 8-bit bytes per 706 computer word. Single precision fixed point byte data is held in an 8-bit two's complement number format. Bit-0 and bit-8, the sign bits, are zero for positive numbers and ones for negative numbers. The fractional or integer portions of single precision fixed point byte data words are held in bits 1 through 7 and bits 9 through 15. The number range for byte data is

$$\text{Fractional: } -1 \leq N < +1$$

$$\text{Integer: } -2^7 \leq N \leq +2^7 - 1$$

1-3.2.2 Double Precision Integer and Multiprecision Floating Point Data

Raytheon 706 software provides the capability for using double precision data words and floating point data words (Figure 1-6). The double precision integer data is held in two words with the sign of the number contained in bit-0 of the first word. Bits 1-15 of word 1 contain the most significant half of the integer and bits 1-15 of the second word contain the least significant half of the integer. The range for double precision integer data is

$$\text{Integer: } -2^{30} \leq N \leq +2^{30} - 1$$

Floating point data consists of a sign, a mantissa and an exponent. All exponents for this type of data are in excess 80_{16} notation and contained in 8 bits (i.e., an exponent greater than 80_{16} indicates a positive exponent; and exponent less than 80_{16} indicates a negative exponent). Both double precision integer and floating point data is fully described in Section 2-2.2 of this manual.

1-3.3 Status Word Formats

The machine status word (Figure 1-7) is saved in memory during the processing of an interrupt. Upon execution of an interrupt return (INR) instruction the machine status is restored. The machine status word contains the contents of EXR, the state of the comparison, overflow, global-local mode, user state, memory protect, and memory parity flip-flops. See operation of INR instruction in Section 2-1.1 of this manual.

The I/O status word meanings vary according to the particular device being used. Normally, the status word from an I/O device is returned to the accumulator after issuing a DIN X, 0 instruction with 'X' being the individual device code. The status word is useful in determining whether a device is operable, has had an error on the last operation and most important, if an interrupt has been generated and has not been serviced. Status word bit meanings are found in Section 5 of this manual.

1-3.4 Address Word Format

The address word format (Figure 1-8) uses bits 1-15 to indicate any one of 32,768 words or bits 0-15 to indicate any one of 65,536 bytes. If indirect addressing is desired, bit-0 is set to a one otherwise, bit-0 is zero. Only word addresses or the first 32,768 byte addresses may be indirectly addressed (See Indirect Addressing, Section 1-4.7).

1-3.5 Assembler Source Statement Formats

Three types of source statements are acceptable to the 706 assemblers (SYM I, SYM II): Title, Comment and Instruction. Source statements may exist on 80 column cards, paper tape, magnetic tape, or disc. See Figure 1-9.

1-3.5.1 Title

A title card is designated by an apostrophe (') in column one. Columns 2 through 19 replace the previous listing title. The assembler spaces the listing output to a new page and puts the title at the top of the page. Any of the characters listed in Appendix G, Character Codes, are permitted with the exception of the carriage return, replacement arrow (←), and rub-out. No object text is produced by this statement.

1-3.5.2 Comment

A comment card is designated by an asterisk (*) in column one. The remainder of the card is used for comments and will appear in the listing output. Any of the characters listed in Appendix G, Character Codes, are permitted with the exception of the carriage return, replacement arrow (←), and rub-out. No object text is produced by this statement.

1-3.5.3 Instruction

An instruction statement consists of four fields: label (name of location), operation, operand, and comments. The label field begins in column one; however, the remainder of the statement is free field (i.e., fields are separated by at least one blank). An instruction statement must contain an operation field.

Label Field

The label is the symbolic name of an instruction, constant, or storage location. It allows symbolic referencing of this location elsewhere in the program. A label begins in column one and is composed of one or more alphanumeric characters, the first of which must be alphabetic. There is no limit to the number of characters in a label. If column one is blank, the label field is assumed to be blank. Some statements do not require a label field.

Operation Field

Operation mnemonics are three or four letter codes that specify a machine operation, procedure, or assembler directive. More than four alphanumeric characters may be given with a four letter code, but only the first four letters are used. The one column modification subfield is separated from the preceding operation field by one or more blanks. It consists of one of the following four characters:

Character	Modification Specified
*	Indexing
D	Double Shift
L	Left Byte Shift
R	Right Byte Shift

The modification characters may be used only with certain instructions. If they are used improperly, the assembler will flag the instruction. For example:

Operation	Correct
LDW *	Yes
LDW L	No (L specifies left byte , LDW load word)
SLC L	Yes
SLL R	Yes
SLL *	No (Shift cannot be indexed)

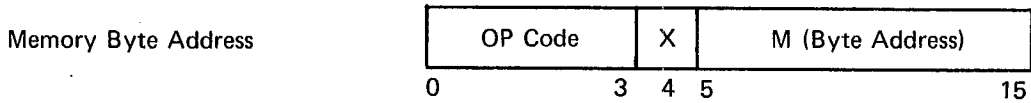
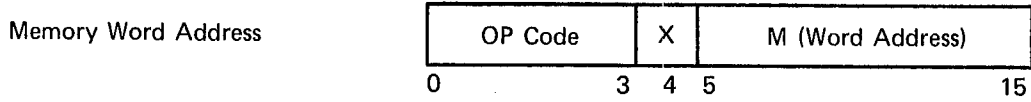
Operand Field

Use of the operand field varies according to the operation. Hence, an explanation is given with each instruction. The operand field may contain any of the following: a name, an expression, a constant, a series of operands, or may be left blank.

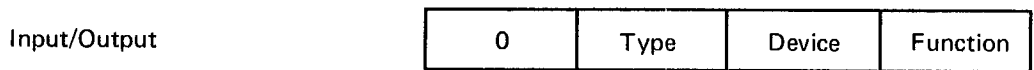
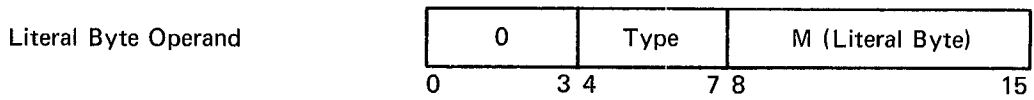
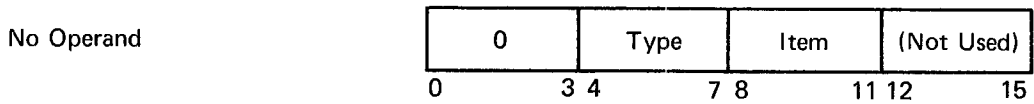
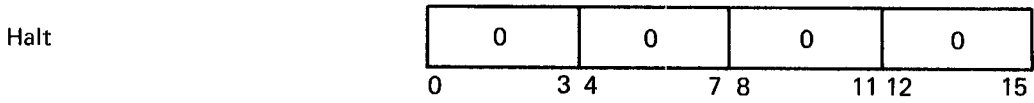
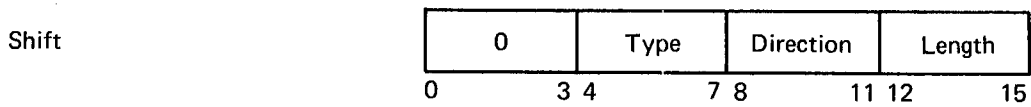
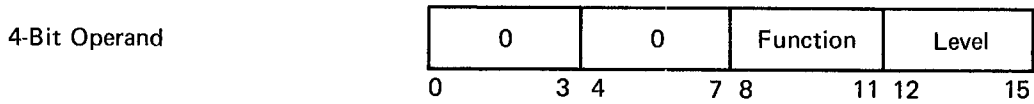
Comment Field

The remaining columns following the first blank after the operand are available for comments. If no operand is required, comments may start after the first blank after the operation field.

MEMORY REFERENCE



GENERIC



DOUBLE WORD (Optional)

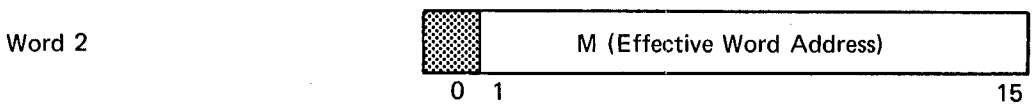
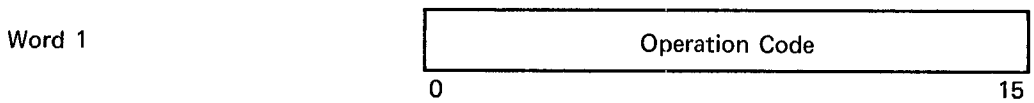


Figure 1-4. Instruction Word Formats

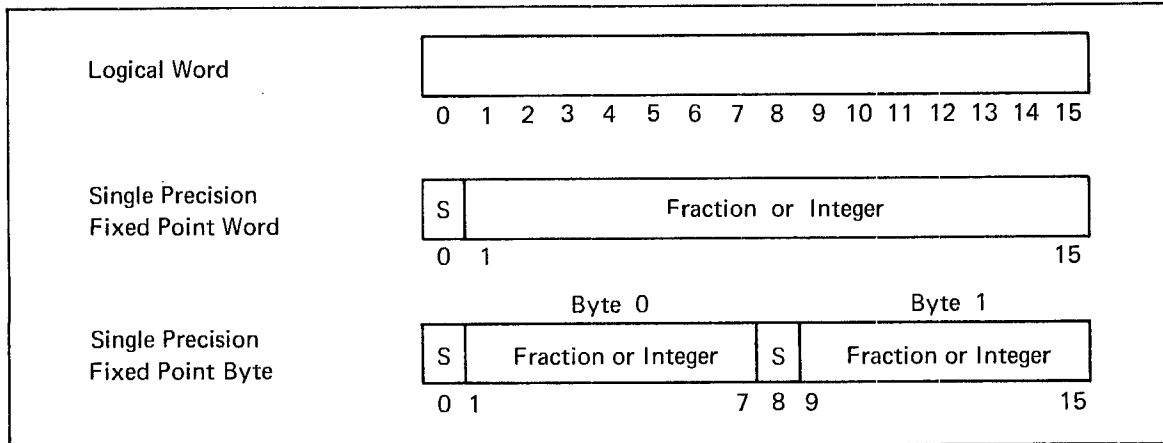


Figure 1-5. Hardware Data Word Formats

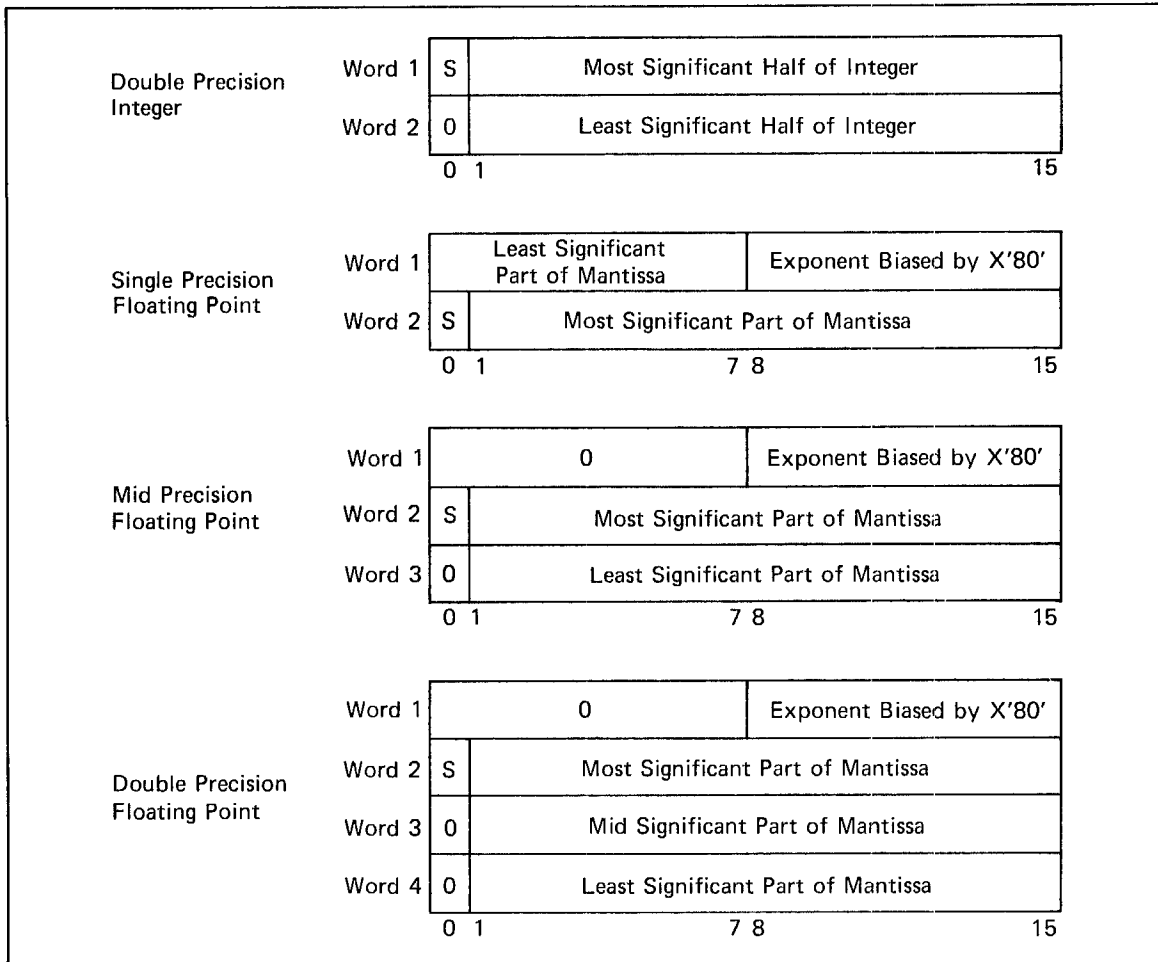
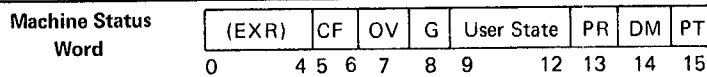
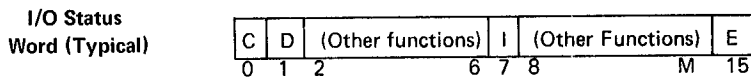


Figure 1-6. Software Data Word Formats



BIT(S)	SYMBOL	MEANING
0-4	(EXR)	Contents of Extension Register
5,6	CF	Comparison Flip-Flops, 00 – Last Comparison Greater Than 01 – Last Comparison Equal 10 – Last Comparison Less Than 11 – Not Allowed
7	OV	Overflow Flip-Flop
8	G	Global/Local Flip-Flop 1 – Global State 0 – Local State
9-12		User State Flip-Flops (See Memory Protect, Section 6-7)
13	PR	CPU Memory Parity Error Flip-Flop
14	DM	DMA Memory Parity Error Flip-Flop
15	PT	Protect Violation Flip-Flop



BIT	SYMBOL	MEANING WHEN TRUE
0	C	I/O Controller Not Ready
1	D	I/O Device Not Ready
7	I	Interrupt Present, has not been serviced
15	E	Error detected on last operation
2-6		Other Functions (See Section 5)
8-14		Other Functions (See Section 5)

Figure 1-7. Status Word Formats

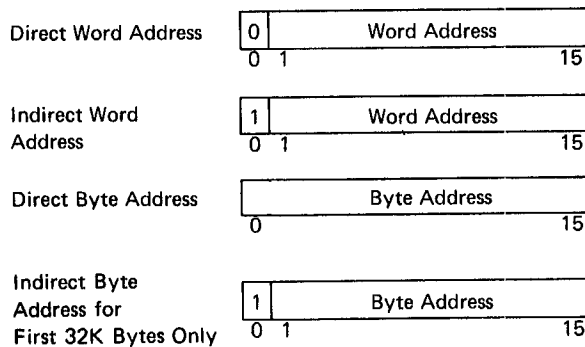
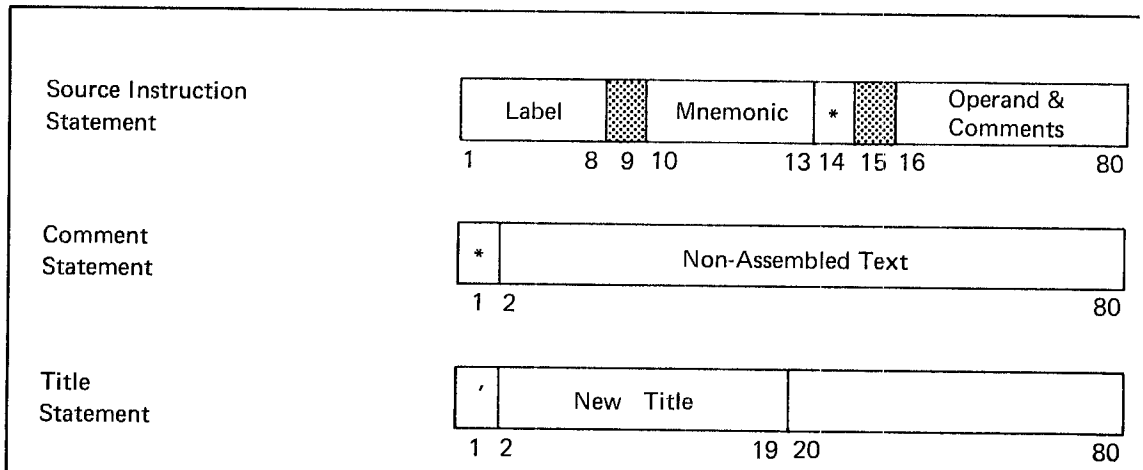


Figure 1-8. Address Word Formats



ASSEMBLER SYNTAX

Character Set	Alphabetic	Numeric	Special		
	A through Z	0 through 9	+ - , . / \$ blank / = : * ()		
Constants	Decimal Integer	Hex Integer	Alphanumeric		
	32119 39 -706	X '3F90' X 'A' X 'FFFF'	'AB' 'B'' '\$'		
	Concatenated Integer		Hex Equivalent		
	[3:4, X'FF':8, Six:3] [Six + Six: 7, Six-1:3] Where Six Equ 6		X'3FFC' X'1940'		
	Dbl. Precision Decimal Integer		Sgl. Precision Floating Point		
	1 -392457381 512		.375 37.5 E-2 (37.5 X 10 ⁻²) -512		
	Mid Precision Floating Point		Dbl. Precision Floating Point		
	1 - 37.S E-2 -47.1		.375 1.23456789 E37 5E-20		
	Literals	= 3 = SIN = 'SL' = JOE + 47 = X'A7'			
	Expressions	Name	Factor	Designator	Operator
SAM A23		\$ SAM 424	-623 /ALPHA	+ -	

Figure 1-9. Assembler Word Formats and Syntax

ADDRESSING

Fourteen of the 706's instructions reference core memory. In addition, the optional hardware divided multiply instructions also reference memory. The remainder of the instructions are considered to be generic and do not require a memory access other than the fetching of the instruction. By using the memory reference format without indexing, only one of 32,768 words or 65,536 bytes may be directly addressed provided that the Extension Register (EXR) has been pre-set to point to any one of 16 possible 2048-word pages or 32 possible 2048-byte pages. See Figure 1-4.

Using the memory reference format with indexing, any memory location may be accessed regardless of page selection (Global Mode) or indexing can be used relative to a given word or byte page (Local Mode).

1-4.1 Page Numbering

Figure 1-10 shows the memory layout including the byte and word page boundary addresses in both decimal and hexadecimal notation. The 32,768 possible word locations of the 706 memory are divided into 32 byte pages that are numbered from $0-1F_{16}$ and 16 word pages that are numbered $0-F_{16}$. The 706 memory reference instruction uses eleven bits (5-15) to contain a memory address therefore, only 2048 (2^{11}) separate addresses can be specified directly. The page in which the address resides must be held in the Extension Register. Figure 1-11 gives the conversion factors for calculating the word page number (WPN) to a byte page number (BPN).

1-4.2 Word and Byte Addresses

One 16-bit word corresponds to two 8-bit bytes. Figure 1-12 shows the relationship between word and byte addresses. For instance, word 3 corresponds to bytes 6 and 7 while byte 3 corresponds to the right half of word 1.

Figure 1-12 also shows the formulas for converting word addresses to byte addresses and vice-versa. All left bytes (bits 0-7) have even byte addresses and all right bytes (bits 8-15) have odd byte addresses.

1-4.3 Extension Register

The Extension Register (EXR) is a 5-bit register that is used to either hold the operand's word page

number during the execution of a memory word type instruction (e.g., LDW, ADD) or hold the operand's byte page number during the execution of a memory byte type instruction (e.g., LDB, STB, CMB). During the execution of a non-indexed instruction, the contents of EXR are concatenated to the 11-bit address M, to form a 16-bit byte address or a 15-bit word address (by using only the first 4 bits of EXR).

For example, Figure 1-13 shows an effective word address $(08E4)_{16}$ being formed by the concatenation of EXR $(02)_{16}$ to the memory operand word address $(OE4)_{16}$.

Figure 1-14 shows an effective byte address $(1241)_{16}$ or the right byte of word address $(920)_{16}$, being formed by the concatenation of EXR $(02)_{16}$ to the memory operand byte address $(241)_{16}$.

DECIMAL	HEX	BYTE PAGE NO.	WORD PAGE NO.
0000	0000	0	0
1023	03FF	1	
1024	0400		
2047	07FF		
2048	0800	2	1
3071	0BFF	3	
3072	0C00		
4095	0FFF		
4096	1000	4	2
5119	13FF	5	
5120	1400		
6143	17FF		
6144	1800	6	3
7167	1BFF	7	
7168	1C00		
8191	1FFF		
8192	2000	8	4
9215	23FF	9	
9216	2400		
10239	27FF		
10240	2800	A	5
11263	2BFF	B	
11264	2C00		
12287	2FFF		
12288	3000	C	6
13311	33FF	D	
13312	3400		
14335	37FF		
14336	3800	E	7
15359	3BFF	F	
15360	3C00		
16383	3FFF		
16384	4000	10	8
17407	43FF	11	
17408	4400		
18431	47FF		
18432	4800	12	9
19455	4BFF	13	
19456	4C00		
20479	4FFF		
20480	5000	14	A
21503	53FF	15	
21504	5400		
22527	57FF		
22528	5800	16	B
23551	5BFF	17	
23552	5C00		
24575	5FFF		
24576	6000	18	C
25599	63FF	19	
25600	6400		
26623	67FF		
26624	6800	1A	D
27647	6BFF	1B	
27648	6C00		
28671	6FFF		
28672	7000	1C	E
29695	73FF	1D	
29696	7400		
30719	77FF		
30720	7800	1E	F
31743	7BFF	1F	
31744	7C00		
32767	7FFF		

Figure 1-10. Memory Layout

$WPN = M_W / 2048_{10}$ $WPN = M_B / 4096_{10}$ $WPN = BPN / 2$	Integer Value Only
$BPN = M_W / 1024_{10}$ $BPN = M_B / 2048_{10}$ BPN Cannot be determined from WPN above	
WPN – Word Page Number BPN – Byte Page Number M_W – Memory Word Address M_B – Memory Byte Address	

Figure 1-11. Word and Byte Page Number Conversion.

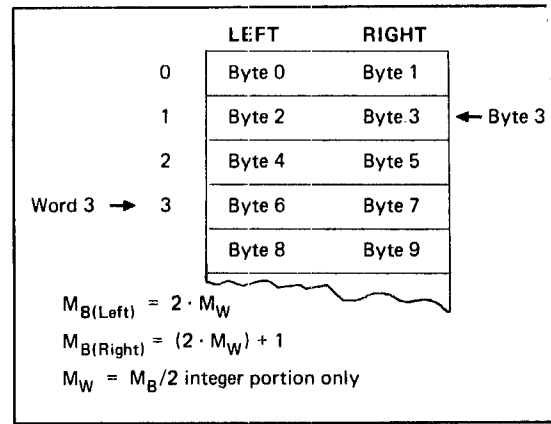


Figure 1-12. Word vs. Byte Addressing

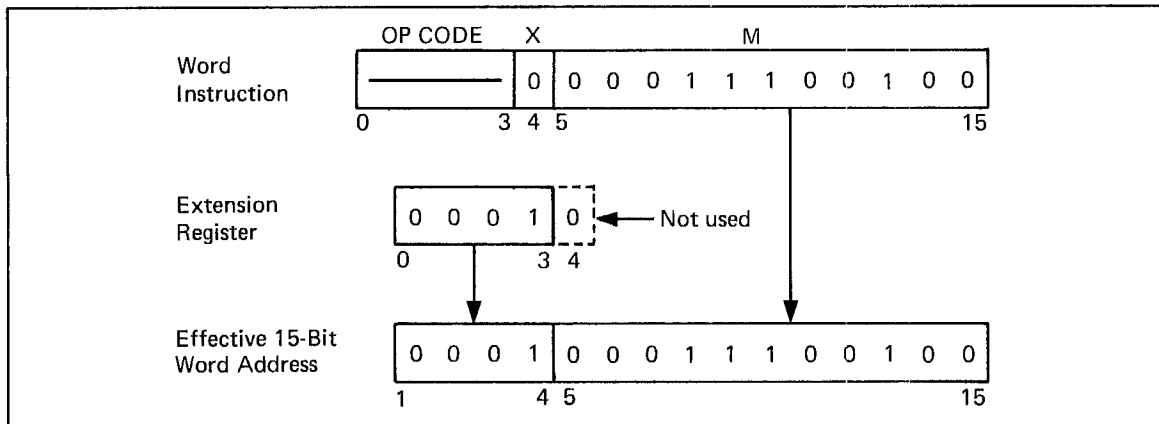


Figure 1-13. Direct Word Addressing Example

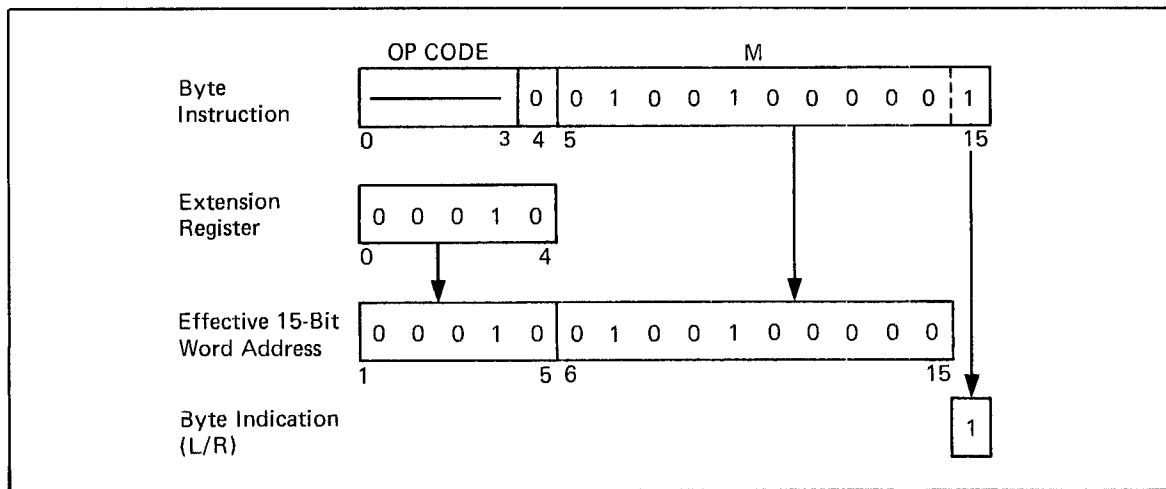


Figure 1-14. Direct Byte Addressing Example

1-4.3.1 Loading the Extension Register

The Extension Register (EXR) is loaded by executing one of three hardware instructions or it is loaded automatically during every memory fetch of an operand.

The Copy Index to Extension Register (CXE) copies bits 0-4 of the index register into bits 0-4 of the extension register. The Set Memory Upper (SMU) and Set Memory Lower (SML) instructions copy bits 12-15 of the instruction word into bits 1-4 of the extension register while copying a binary 1 or 0, respectively, into bit-0 of the extension register.

During the operand fetch phase of a memory type instruction, the current word page number or byte page number located in the program counter (PCR)₁₋₅ is automatically copied into (EXR)₀₋₄. When a Jump (JMP) or Jump and Store Return in Index (JSX) instruction is executed, the page number of the next instruction (instruction being "jumped to") is copied into EXR. Therefore, unless altered by a Set Memory Upper (SMU), Set Memory Lower (SML), or a Copy Index to Extension Register (CXE), the EXR is "pointing" to the word or byte page plus one memory word location used in the last memory fetch or store type of instruction (including JMP and JSX). Figure 1-15 shows a pictorial representation of every method used in loading the EXR.

1-4.4 Direct Addressing

Bit-4 of the instruction word specifies either direct or indexed addressing. If bit-4 of the instruction word is a zero or column 14 of the source code is a blank, direct addressing is specified. The base address formed by the concatenation of the contents of EXR to the address field of the instruction (M) is used directly as the effective address. See Figure 1-16 and Figure 1-17.

Care must be exercised when using instructions that directly address word or byte locations that are outside of the *local* page, (i.e., the page from which instructions are being executed). In particular, a Set Memory Bank (SMB) should be executed just prior to executing a memory type of instruction whose effective address lies outside of the local page.

1-4.4.1 Set Memory Bank (SMB)

The Set Memory Upper (SMU) and Set Memory Lower (SML) instructions establish in the extension register the word or byte page for the next memory access instruction. The SML instruction selects pages in the lower 16 K of memory while the SMU instruction selects pages in the upper 16 K of memory (if available).

The SMB instruction (Ref. 2-1.11) is not really a hardware instruction. It is an instruction to the assembler to assemble either a Set Memory Upper (SMU) or a Set Memory Lower (SML) instruction using the first five most significant bits of the operand address as the literal value (byte page) for the SMU or SML instruction. If the programmer were to code an SMU or SML instruction, he would have to know from which page the operand was coming from in order to fill in the operand field. In most cases (especially with relocatable programs) the programmer does not know where the operand is located or it is too much trouble to calculate the page location of the operand. By using the SMB instruction with the same symbolic operand address as the memory reference instruction, the programmer can be assured of having the EXR set properly before referencing another word or byte page. If the programmer is positive that the operand is located in the local page there is no need to execute a SMB instruction.

As good practice, an SMB instruction should be used before the first memory reference instruction is executed, even if the first operand is known to be in the local page. The reason for this is that the EXR has not been initialized and may contain an unknown page number. After initializing the EXR it is only necessary to use SMB when referencing operands outside of the local page or when executing a Jump (JMP) or Jump and Store Return in Index (JSX) instruction that transfers to a location outside of the local page. It is not necessary to use an SMB instruction when transferring execution control (JMP, JSX) within the local page.

1-4.4.2 Direct Addressing Example

Figure 1-18 shows the assembled coding of a routine that simply moves the contents of three words into the contents of three other words and then jumps to the end of the routine and halts. Two of the words lie outside of the local page in word page 1. Lines 08 and 12 demonstrate the proper use of the SMB instruction when referencing effective addresses outside of word page 0. After executing the Load Word (LDW) instruction on line 09, it was not necessary to reset the EXR back to zero. This was done automatically when the location 'ED' was accessed;

at that time the program counter (PCR) had bits 1-5 (equal to 0) copied into the EXR, thereby resetting EXR to the local page.

Note that the routine starts on line 05 with an SMB instruction that references some address (e.g., BILL) in the local page. This insures that the EXR has been initialized for the first memory reference instruction located on line 06. In addition, there is an SMB instruction that immediately precedes the JMP instruction on line 15. The EXR must be set to word page 1 in this case. At the end of executing the JMP instruction, the EXR remains set to word page 1.

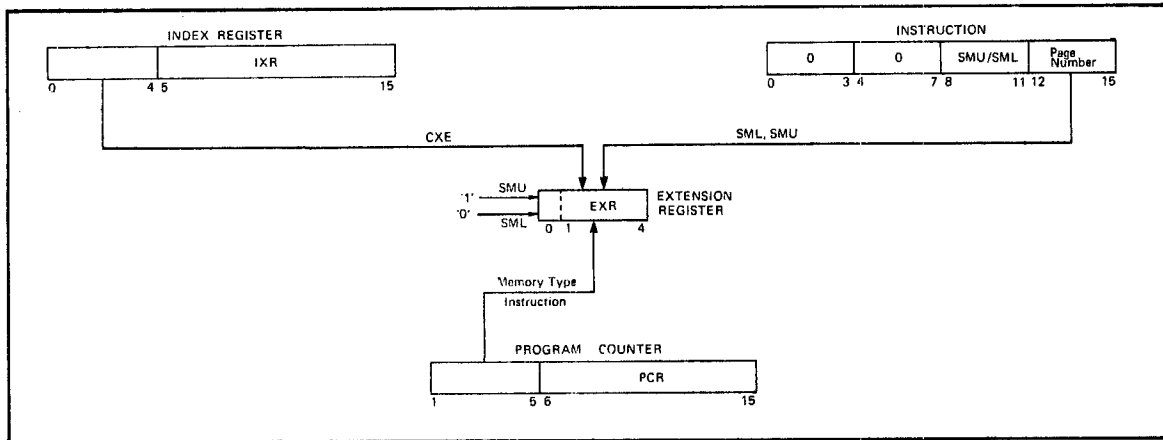


Figure 1-15. Extension Register Operation

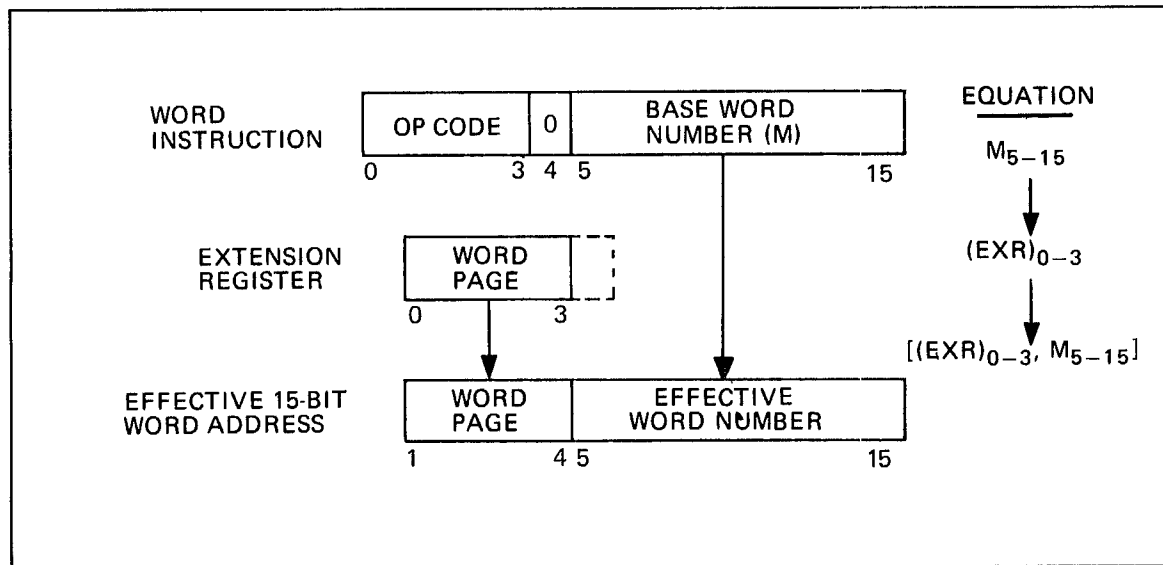


Figure 1-16. Direct Word Addressing Operation

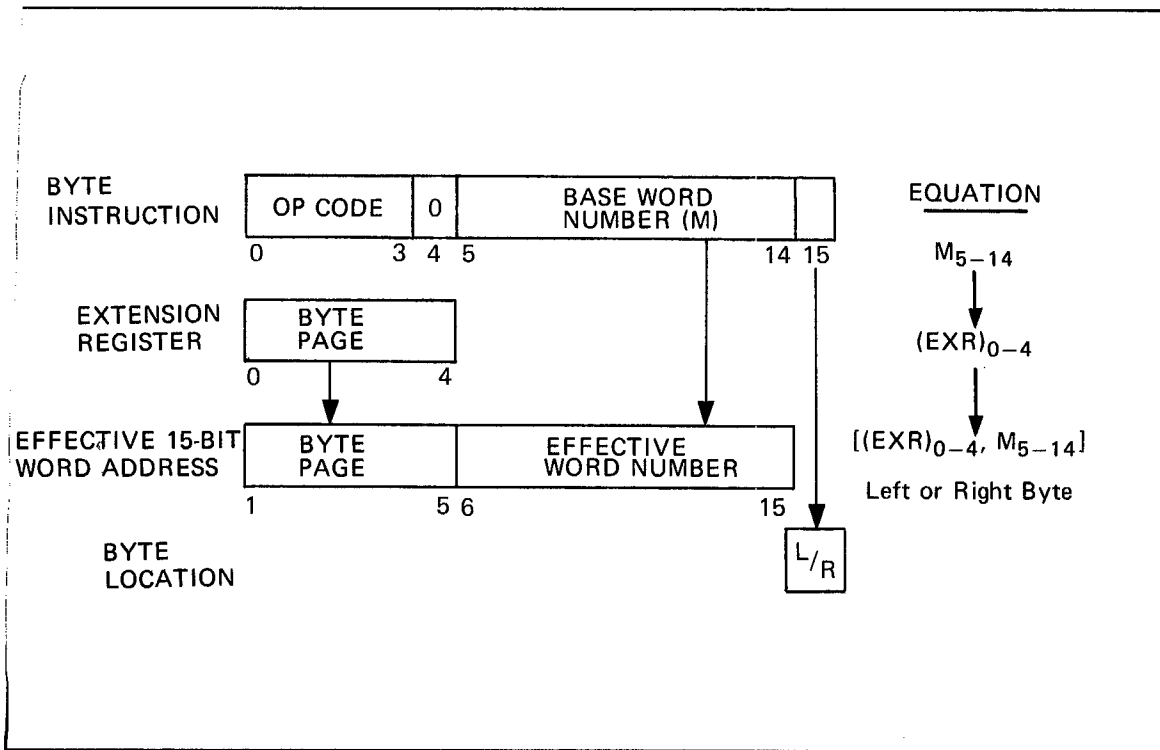


Figure 1-17. Direct Byte Addressing Operation

```

1 *   DIRECT MEMORY ADDRESSING EXAMPLE
2 *
3 *
4
5 START   ORIG 0
6         SMB  BILL          SET EXR TO WORD PAGE 0
7         LDW  BILL          (ACR) ← (BILL)
8         STW  TOM           (TOM) ← (BILL)
9         SMB  ED            SET EXR TO WORD PAGE 1
10        LDW  ED           (ACR) ← (ED)
11        STW  DAVE         (DAVE) ← (ED)
12        LDW  MIKE        (ACR) ← (MIKE)
13        SMB  BARBARA     SET EXR TO WORD PAGE 1
14        STW  BARBARA    (BARBARA) ← (MIKE)
15        SMB  ED+2       SET EXR TO WORD PAGE 1
16        JMP  ED+2       GO TO HALT IN WORD PAGE 1
17 BILL   DATA 1
18 TOM    DATA 0
19 DAVE   DATA 0
20 MIKE   DATA 3
21        ORIG X'800'     WORD PAGE BOUNDRY
22 ED     DATA 2
23 BARBARA DATA 0
24        HLT
25        END

```

Figure 1-18. Direct Memory Addressing Example

1-4.5 Index Register

The index register (IXR) is a 16-bit register that is used for a number of purposes. Its primary function is to hold an address modifier for use in forming effective addresses in indexed addressing modes.

1-4.5.1 Decrementing and Incrementing

The number in IXR may be decremented or incremented by means of DXS or IXS instructions. This keeps track of the number of iterations of an operation and causes a program to skip after a preselected number of iterations. This function may also involve address modification, but can be performed without address modification as well.

1-4.5.2 Subroutine Linkage

The IXR is used to store the return address in subroutine linkages. The JSX instruction is used to effect the program jump to the subroutine and to store the return address in IXR.

1-4.5.3 General Purpose Use

The IXR is also used as a general-purpose register in data manipulation operations as an extension of the accumulator register in double length shift operations and in the double word operations of multiply and divide. Besides the shift operations and JSX instruction, the three commands for altering the contents of the IXR are: Copy Accumulator to Index (CAX), Copy Extension Register to Index (CEX) and Load Index Register (LDX). Figure 1-19 gives a pictorial representation of every method of loading the IXR.

1-4.6 Indexed Addressing

A one in bit-4 of the instruction word or an asterisk in column-14 of the source code specifies indexing. Two modes of indexing are provided, local and global. The machine may be set to either the local or global mode using SLM or SGM instructions. Execution of a JSX instruction automatically selects the global mode.

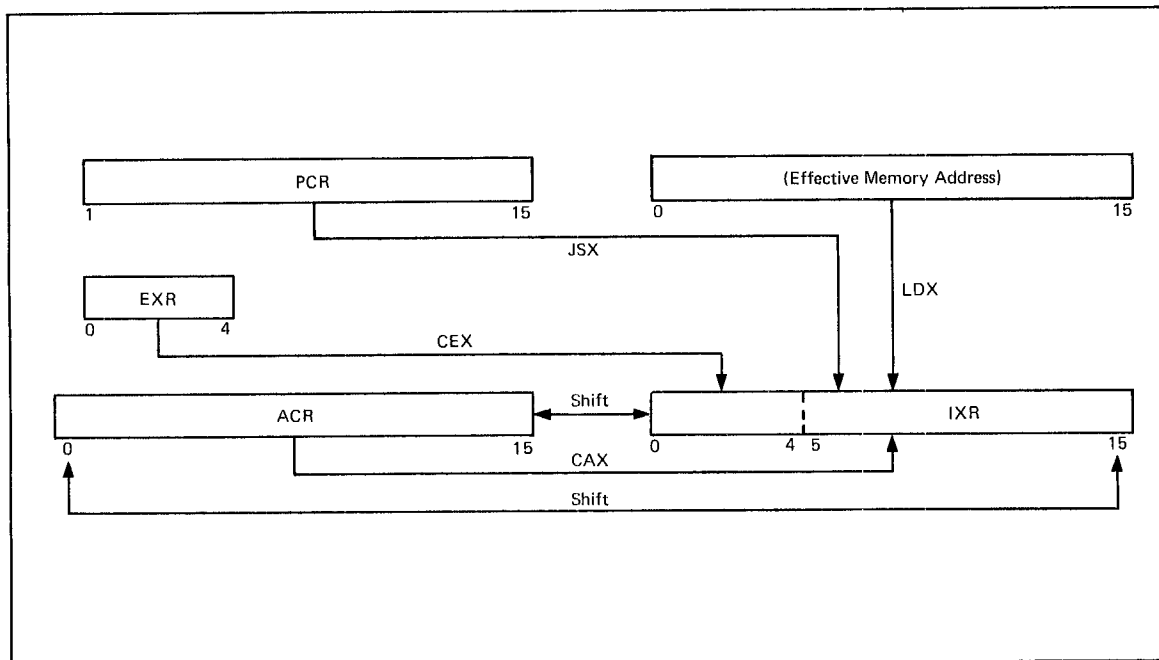


Figure 1-19. Index Register Operation

1-4.6.1 Local Mode Indexing

In the local mode of indexing, a base address is formed using the extension register. The contents of the index register (IXR), treated as a 16-bit two's complement number, are added to this base address to form the effective address. See Figure 1-20 and 1-21.

In this mode, indexing is accomplished by specifying the starting address of a table (or series of values to be accessed by indexing) in the

operand field of the load or store instruction. If the start of the table is located in the local page, the EXR may be left alone. If the start of the table is in a page other than the local page, the EXR must be set with an SMB instruction prior to the execution of the indexed instruction. Usually, the limit of the table length is put into the IXR. IXR is decremented or incremented after each iteration of indexing by a DXS or IXS command.

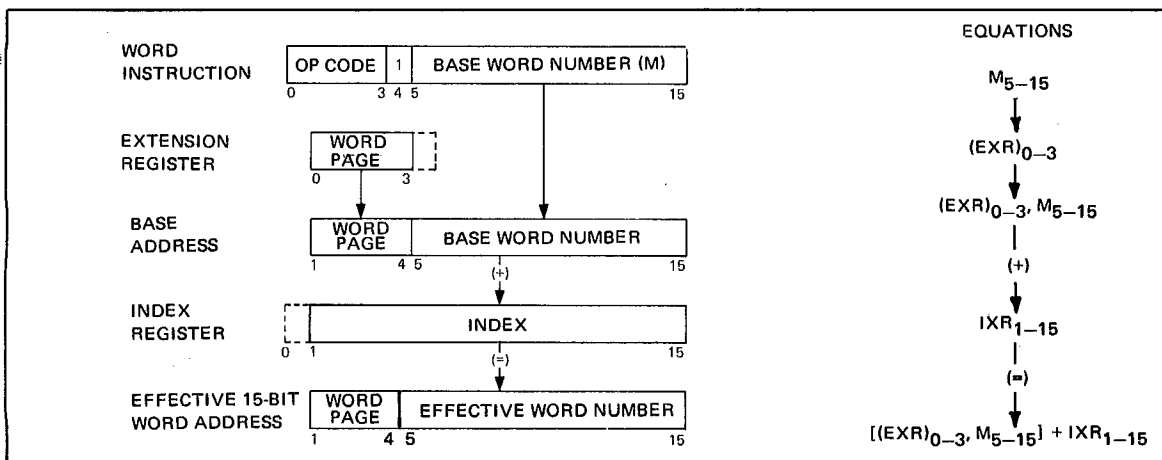


Figure 1-20. Local Mode Word Indexing Operation

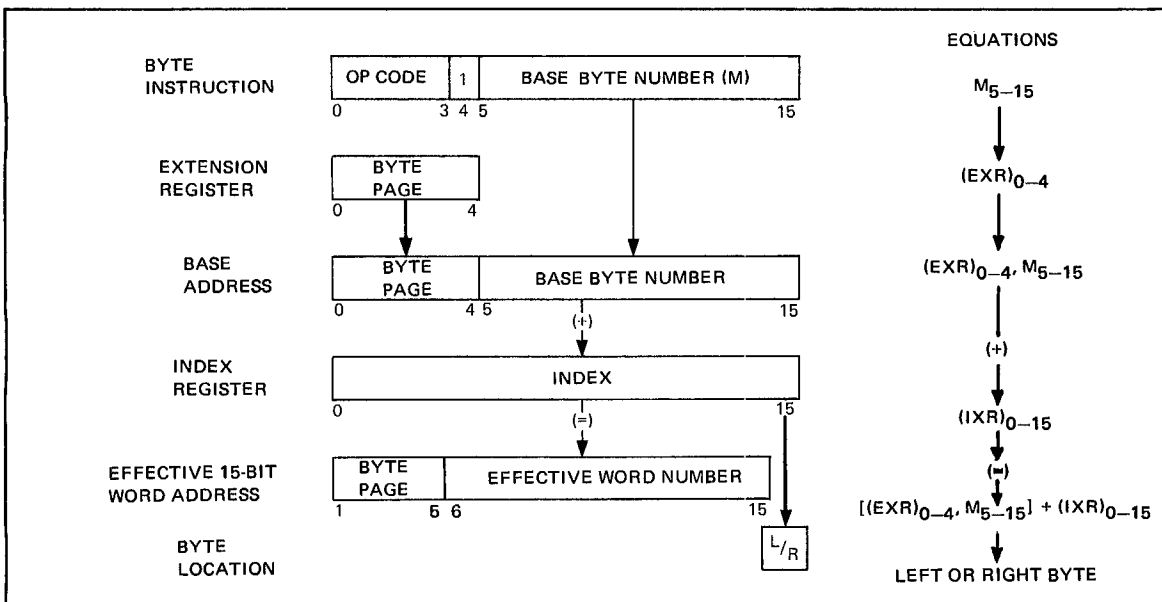


Figure 1-21. Local Mode Byte Indexing Operation

1-4.6.2 Local Mode Indexing Example

Figure 1-22 shows an example of indexing where a table is searched for a given VALUE. If VALUE is in the table the program halts with one less than the number of places accessed before VALUE was found. The end of the table search is detected by sensing a word of all binary ones. The table starts in the local word page and extends into the next word page, therefore, an SMB command is not needed before the indexed instruction.

1-4.6.3 Global Mode Indexing

In the global mode of indexing, zeros are substituted for the contents of the EXR when the base address is formed. The contents of IXR are added to this base address to form the effective address. See Figures 1-23 and 1-24.

In the global mode of indexing, the starting address of a table (or series of values to be accessed by indexing) is normally loaded into the index register and a zero put into the operand field of the indexed instruction. Whether the table starts in the local page or not is of no consequence, since the absolute address (including the page number) of the table starting address must reside in IXR (16 bits).

1-4.6.4 Global Mode Indexing Example

Figure 1-25 shows an example of indexing where a table is searched for a given VALUE. If VALUE is in the table the program halts with the address in IXR, where VALUE was found in the table. The end of the table search is detected by sensing a word of all binary ones. The table starts in word page 1 while the routine resides in word page 0. An SMB instruction is not necessary before the indexed instruction because in the global mode the EXR is not concatenated to the base address.

1-4.6.5 Using Global Mode to Return From Subroutines

Executing a Jump and Store Return in Index (JSX) instruction causes the index register (IXR) to be loaded with the address of the instruction following the JSX (return address), the program counter (PCR) to be loaded with the effective address, and the local/global flip-flop to be set in the global state. Being in the global state allows the programmer to return from the subroutine by executing an indexed Jump (JSX * n) with a base address of n (number of arguments in subroutine) and the return address in IXR. See Section 2-2.3, SUBR Directive. Care must be taken to restore the computer to the local mode, if necessary, after returning from the subroutine.

	1 *	LOCAL MODE INDEXING EXAMPLE	
	2 *		
	3 *		
	4	ORIG 0	
07FD 0040	5 START	SLM	SET LOCAL INDEXING MODE
07FE 0081	6	SMB ZERO	SET EXR TO WORD PAGE 0
07FF 97FA	7	LDM ZERO	(IXR) = 0
07F0 8FFD	8 SEARCH	LDM *TABLE	(ACR) = (TABLE + IXR)
07F1 F7FH	9	CMW ALLONES	DOES (ACR) = FFFF
07F2 0870	10	SAE	
07F3 0000	11	HLT	YES, HALT, VALUE NOT FOUND
07F4 F7FC	12	CMW VALUE	DOES (ACR) = (VALUE)
07F5 0870	13	SAE	
07F6 0000	14	HLT	YES, HALT, VALUE FOUND, COUNT IN IXR,
07F7 0401	15	IXS 1	NO, (IXR) = (IXR) + 1
07F8 17F0	16	JMP SEARCH	RETURN TO SEARCH
07F9 17F0	17	JMP SEARCH	
	18 *		
	19 *		
07FA 0000	20 ZERO	DATA 0	
07FB FFFF	21 ALLONES	DATA X'FFFF'	
07FC 0000	22 VALUE	DATA 0	VALUE INITIALIZED BY SOME
	23 *		OTHER ROUTINE
	24 *		
07FD 0001	25 TABLE	DATA 1	
07FE 0002	26	DATA 2	
07FF 0003	27	DATA 3	
	28	ORIG X'800'	WORD PAGE BOUNDARY
0800 0004	29	DATA 4	
0801 0005	30	DATA 5	
0802 0006	31	DATA 6	
0803 0007	32	DATA 7	
0804 0008	33	DATA 8	
0805 FFFF	34	DATA X'FFFF'	
	35	END	

Figure 1-22. Local Mode Indexing Example

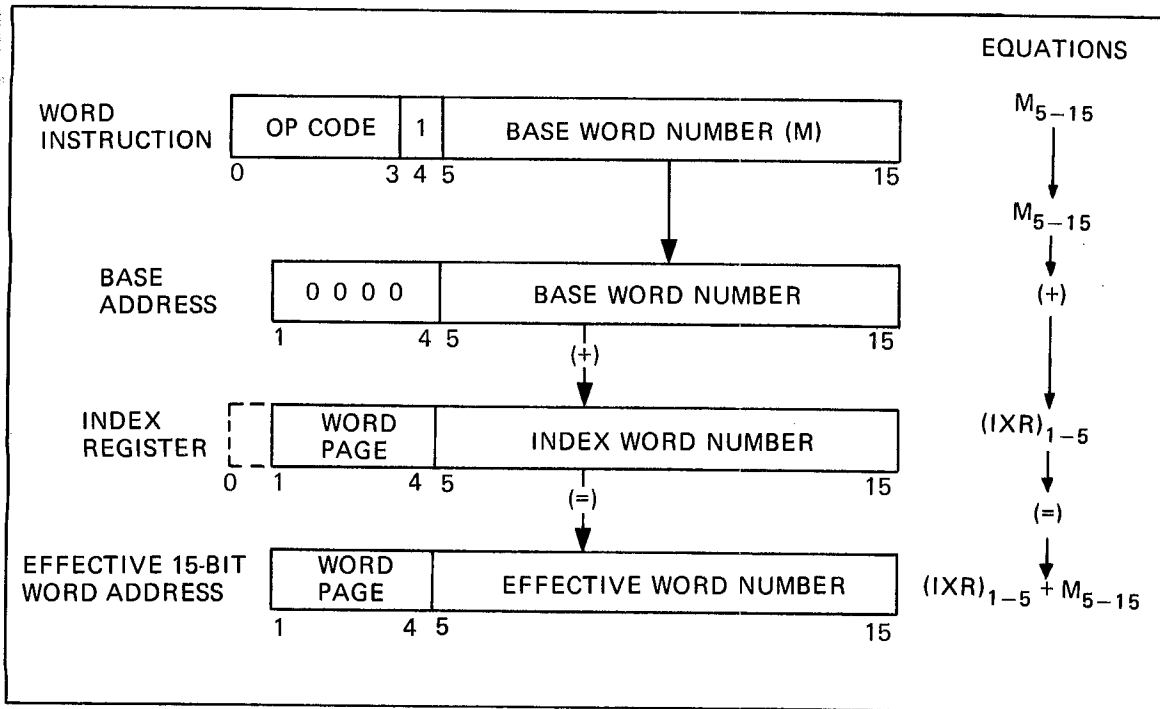


Figure 1-23. Global Mode Word Indexing Operation

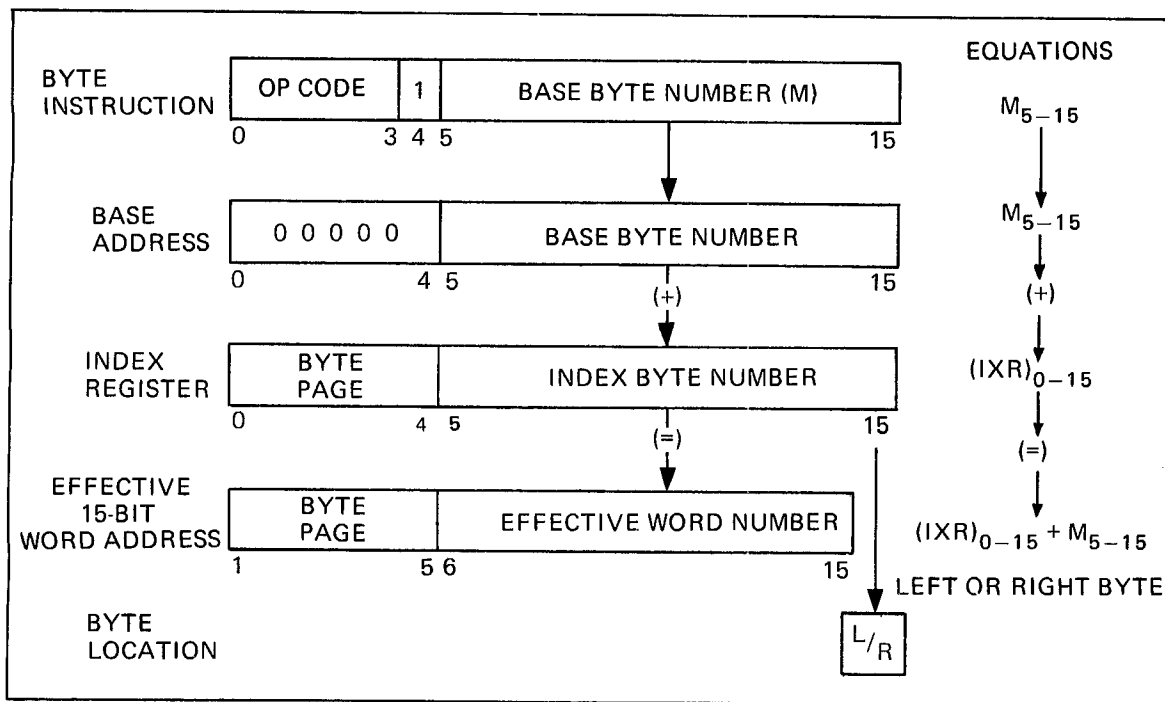


Figure 1-24. Global Mode Byte Indexing Operation

```

1 *   GLOBAL MODE INDEXING EXAMPLE
2 *
3 *
4     ORIG 0
07F0 0050 5 START  SGM          SET GLOBAL INDEXING MODE
07F1 0081 6       SMH  TABLEDR SET EXR TO WORD PAGE 0
07F2 97FD 7       LDX  TABLEDR (IXR) + TABLE
07F3 8800 8 SEARCH LDW  * 0     (ACR) + ((IXR)) OR (TABLE)
07F4 F7FE 9       CMW  ALLONES  DOES (ACR) = FFFF
07F5 0870 10      SNE
07F6 0000 11      HLT          YES, HALT, VALUE NOT FOUND
07F7 F7FF 12      CMW  VALUE   DOES (ACR) = VALUE
07F8 0870 13      SNE
07F9 0000 14      HLT          YES, HALT, VALUE FOUND, ADDR IN IXR
07FA 0401 15      IXS  1       NO, (IXR) + (IXR) + 1
07FB 17F3 16      JMP  SEARCH  RETURN TO SEARCH
07FC 17F3 17      JMP  SEARCH
18 *
19 *
07FD 0800 20 TARLEADR DATA TABLE
07FE FFFF 21 ALLONES DATA X'FFFF'
07FF 0000 22 VALUE  DATA 0       VALUE INITIALIZED BY SOME
23 *                                       OTHER ROUTINE
24 *
25      ORIG X'800'   WORD PAGE BOUNDRY
0800 0001 26 TARLE  DATA 1
0801 0002 27       DATA 2
0802 0003 28       DATA 3
0803 0004 29       DATA 4
0804 0005 30       DATA 5
0805 0006 31       DATA 6
0806 0007 32       DATA 7
0807 0008 33       DATA 8
0808 FFFF 34       DATA X'FFFF'
35      END

```

Figure 1-25. Global Mode Indexing Example

4.7 Indirect Addressing

Indirect addressing means that instead of the operand field of an instruction specifying an effective address, the operand field specifies an address where the effective address is to be found. See Figure 1-26. The 706 performs indirect addressing by doing an indexed instruction in the global mode with the base address of the instruction set equal to zero and the indirect address contained in the index register.

If single level indirect addressing is desired, two instructions, LDX and LDW should be used and they require 4 cycles for execution.

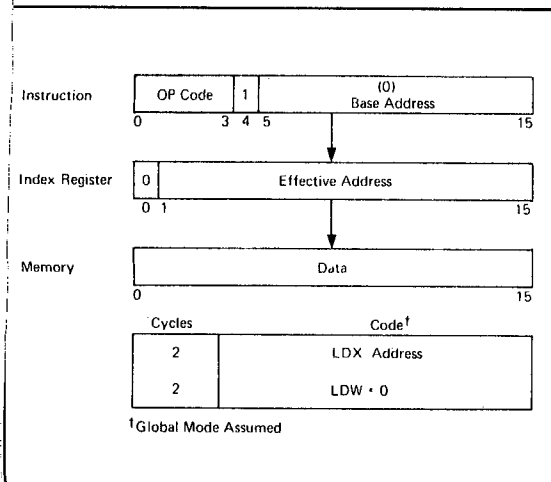


Figure 1-26. Single Level Indirect Addressing.

1-4.7.1 Multilevel Indirect Addressing

Multilevel indirect addressing means that the address that supposedly contains the effective address for single level indirect addressing contains another address where either the effective address or more indirect addresses can be found.

Multilevel indirect addressing is specified by having the "indirect bit" set true for each address that points to another address. Figure 1-27 shows the 706 method of multilevel indirect addressing.

Bits 1-15 directly specify any address in the 706 memory. Bit-0 is used to specify multilevel indirect addressing. When bit-0 of the address word is true, multilevel indirect addressing is specified, when bit-0 is false the final level of indirect addressing has been reached.

If multilevel indirect addressing is desired, the routine shown in Figure 1-27 must be coded (it may be a macro). If it is known that there is more than 1 level of indirect addressing, the first SXP instruction may be omitted.

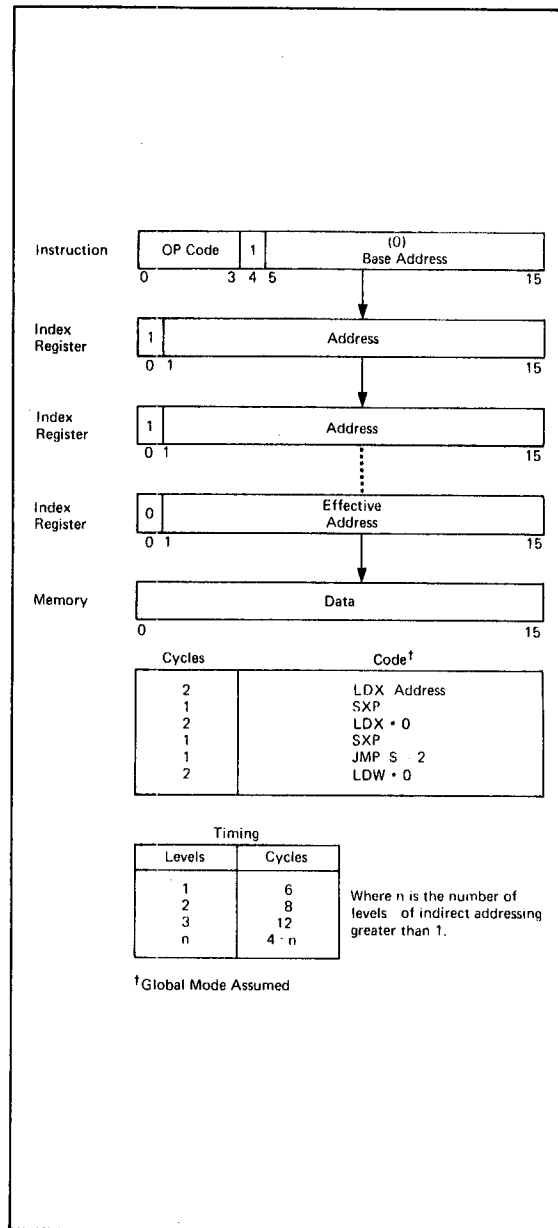


Figure 1-27. Multi-Level Indirect Addressing

1-5. INPUT OUTPUT

1-5.1 Direct Input/Output

The DIO channel, included in the basic 706 configuration, is used for program transfers between the accumulator and up to 16 peripheral controllers. Two instructions, DIN (data input to the accumulator) and DOT (data output from the accumulator), are provided for programmed transfers. The DIO channel is used for data transfers with slow speed devices and also for control of peripheral devices which operate on the DMA channel. This control is mainly used for initialization; i.e., transmission of starting address, block length, and select information to the peripheral controller, for testing the current status of peripheral devices on the DMA channel.

The parallel DIO structure incorporates a 16-bit data output path, a 16-bit data input path, an 8-bit device and function address field, computer clocking and strobe lines, sense line, and the Automatic Priority Interrupt System. Any size data up to 16 bits, including status information, can be transferred via the 16-bit data output path or the 16-bit data input path.

The modular design of the 706 input/output system makes it possible to field expand or reconfigure at any time. Simplified interface requirements facilitate addition of user-designed or special purpose equipment. Flexibility and the power of concurrent multiple device operation are the results of each device having its own control logic and data buffer.

Data transfers, device commands, and status requests are executed in two machine cycles, 1.8 microseconds, by either the DIN or DOT instruction. Data and status are transferred between the accumulator and the selected device by either of two methods, program controlled or interrupt controlled. In the program controlled mode, the program determines when to request or send data by device status inspection or internal program-developed timing. In the interrupt controlled mode, transfer of data is made within an interrupt service routine which is automatically keyed upon hardware controlled program breaks. Whenever a device is ready for a data transfer an interrupt is generated which initiates a program break. Timing of data transmission is thereby determined by the speed of the device. This mode is most useful where data is volatile or dependent upon irregular conditions. See Section 4-2.

1-5.2 Direct Memory Access

The DMA channel, available as an option, is used for direct access to memory without programmed intervention. Once initialized, the transfer of data between the memory system and high-speed peripheral devices is coordinated by the DMA channel on a fixed priority basis. The DMA channel can accommodate up to 6 high-speed peripheral controllers, each having its own level of priority. The priority is established according to the transfer rates and characteristics of the peripheral devices and is completely separate from the Automatic Priority Interrupt System of the central processor. The DMA channel operates in a cycle stealing mode; i. e., transfer of data with the memory steals one memory cycle from the central processor. When the central processor does not require the use of memory during the execution of an instruction, computation and memory access by the DMA channel can occur simultaneously. Maximum transfer through the DMA channel can be up to the speed of the memory, i.e., 1.1 megawords per second.

The DMA is most effectively used with devices such as disc, magnetic tape, and high-speed data acquisition instruments where fast response to a data transfer request is required. Maximum wait time for the highest active input/output request is 900 nanoseconds. Cycle stealing never requires more than one machine cycle per memory access. DMA devices are also interfaced to the DIO bus for the transmission of commands, memory data transfer parameters and status information. Memory address and data transfer block length are functions of the device controller. When a DMA peripheral completes a block transfer, the processor is notified by an interrupt via the Automatic Priority Interrupt System. The program can then determine whether to initialize the next block transfer or discontinue channel operation.

See Section 4-3.

1-5.3 Automatic Priority Interrupt System

The Automatic Priority Interrupt System has access to the central processor via the DIO channel and is used to provide rapid response by the central processor to external conditions in the Input/Output System. The external conditions can typically be the end of a block transfer by a

peripheral device on the DMA channel, a request for service by a peripheral device on the DIO channel, a time marker from either the real-time clock or the interval timer, or a condition unique to a custom interface. The basic 706 configuration is one level of priority interrupt as standard and can be expanded to 16 levels. Each level can be assigned to a specific peripheral device or it can be shared by a number of devices. Each level is allocated three unique memory locations in the lower address portion of memory for storage of the program counter, an interrupt linkage address and machine status.

The Raytheon 706 has five instructions which allow the programmer to achieve full utilization of the interrupt system. The instructions are Mask Interrupt (MSK), Unmask Interrupt (UNM),

Enable Interrupt (ENB), Disable Interrupt (DSB), and Interrupt Return (INR.)

Full interrupt response time is 2.7 microseconds following receipt of an interrupt with a duration of at least 900 nanoseconds. An automatic hardware sequence verifies that no higher interrupt level is pending or active and stores the program counter, machine status, and jumps to the linkage address for that particular level. No software polling of devices is required to determine the source of the interrupt. A higher priority level request can immediately interrupt the service routine of an active lower level. Whether a service routine or the main program is interrupted, pertinent information is efficiently saved under hardware control and an automatic return can be made upon completion of the service of the higher level. See Section 3.

1-6 SOFTWARE

Included with the purchase or lease of a 706 computer system is a full complement of Raytheon developed software. Section 8 and Section 9 of this manual are devoted to a detailed description and use of the 706 software. The remainder of this section gives a brief overview of the programs available to the 706 user.

1-6.1 Software Systems

There are essentially four different configurations of peripheral equipment that can be added to the 706 CPU. The Raytheon software systems are designed to support these four system configurations.

1-6.1.1 BASIC Software System

The BASIC software system emphasizes compactness of program components and is provided for users of the minimum hardware configuration (Figure 1-28). BASIC includes an X-RAY executive for system control, a real time device-independent input/output system. Conversational FORTRAN, SYM I assembler and relocating loader, symbolic editor, SENSOR diagnostics math library, and utility routines. FORTRAN or SYM I statements can be entered at the teletype keyboard and executed immediately.

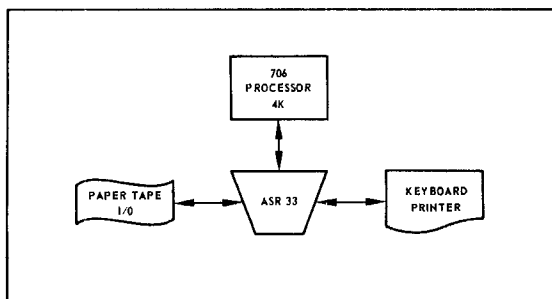


Figure 1-28. BASIC Hardware Configuration

1-6.1.2 STANDARD Software System

If memory is expanded to 8,192 words and a high speed paper tape reader or card reader is added to the basic system, it becomes a STANDARD system (Figure 1-29). In addition to the BASIC software, SYM II (2-pass assembler) and X-RAY Exec. STANDARD can be used. Real-Time FORTRAN IV is available in its entirety in 12K systems, while a subset version is available for 8K systems.

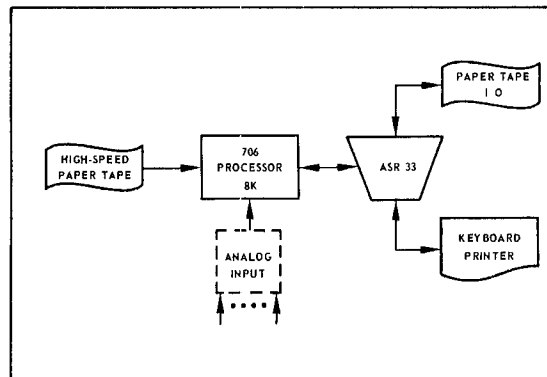


Figure 1-29. STANDARD Hardware Configuration

1-6.1.3 EXTENDED Software System

With the addition of a disc storage device or magnetic tapes and 4 priority interrupts, the STANDARD system becomes the EXTENDED System (figure 1-30). EXTENDED includes the Real-Time Operating System (RTOS) and/or the Magnetic Tape Operating System (MTOS) in addition to the STANDARD system software. With RTOS/MTOS one can have several high priority real-time tasks executing in the foreground, while a batch of computations, assemblies, and other lower priority programs are automatically executed in the background on a time-available basis. In order to take advantage of the automatic batch capabilities of RTOS/MTOS, a card reader should be substituted for the high speed paper tape reader.

1-6.1.4 ADVANCED Software System

ADVANCED systems require a 16K computer with memory protect, 8 priority interrupts, ASR 33/35, card reader, and disc (Figure 1-31). The ADVANCED system includes software of the type usually found only in large scale computer systems; such as a Multi-Programming System (MPS) and Real-Time FORTRAN IV. MPS permits multiple real-time tasks to be run concurrent with automatic batch processing. System features include dynamic task swapping and hardware intertask protection.

Figures 1-28 to 1-31 define minimum hardware requirements for the four software systems. Additional peripherals may be added or device substitutes made in order to improve performance characteristics for particular applications. For

example, in systems which use FORTRAN extensively or otherwise produce large volumes of data output, it is desirable to add a medium or high speed printer to improve system throughput. Additional disc devices or magnetic tapes may be required in systems with large data storage requirements. A fully expanded configuration is shown in Figure 1-32.

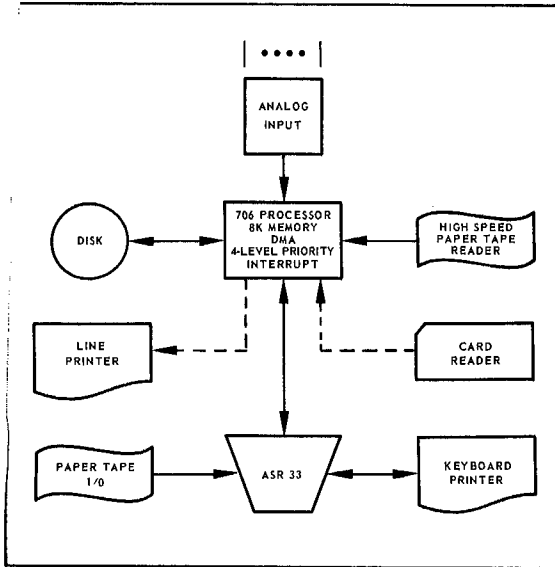


Figure 1-30. EXTENDED Hardware Configuration

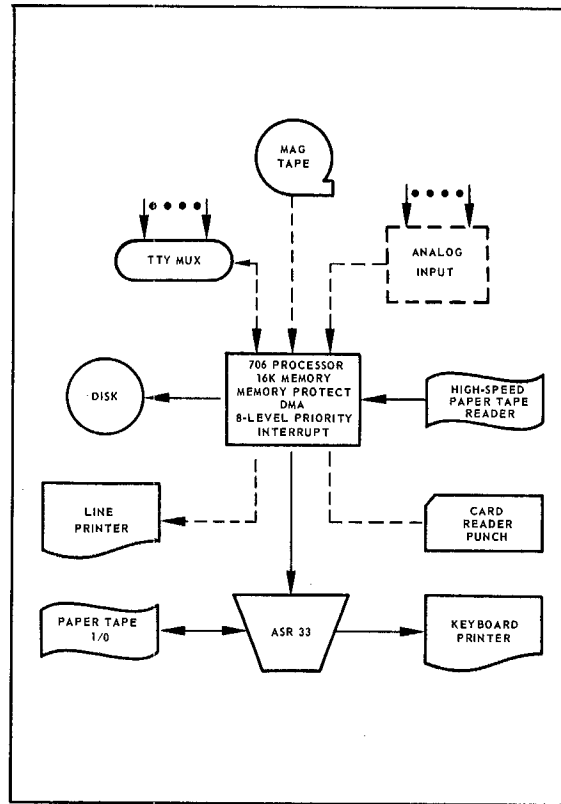


Figure 1-31. ADVANCED Hardware Configuration

1-6.2 Monitors and Executives

Each 706 operating system has a resident monitor and an X-RAY Executive tailored to the hardware configuration. Included in all resident monitors is a file-oriented, device independent input/output system which is designed to allow the computer to be time-shared in a real-time interrupt environment.

Features of the 706 Input/Output Software (IOS) include device reassignment, simultaneous I/O capability, centralized I/O monitor, end-action, and IOS macros. IOS provides a common I/O system for all configurations and alleviates the programmer's need to understand the machine language I/O operation of the 706, while still providing full utilization of the 706 capabilities. The user need only become acquainted with the I/O macros OPEN, DOIO, and STAT to communicate with any standard 706 peripheral. See Section 5-4.

Device independence allows different physical devices to be substituted for I/O operation at execution time without requiring re-assembly or compilation of the program. For example, a program could be debugged by using a small set of test data input from the teletype. During production runs the input can be changed to magnetic tape to process the active data files.

The End Action feature permits program operation to be synchronized to I/O device operation. It allows sharing of time, otherwise wasted during I/O operations, with other tasks. A core resident program can initiate an I/O operation and then exit the system to allow it to initiate another task. Then, when the I/O operation is complete, the system will return to the resident program at its specified end-action address. It can start another I/O operation and then exit. The system will restore the interrupted task. End action may also be used within a single task; for example, to operate a double buffered I/O function.

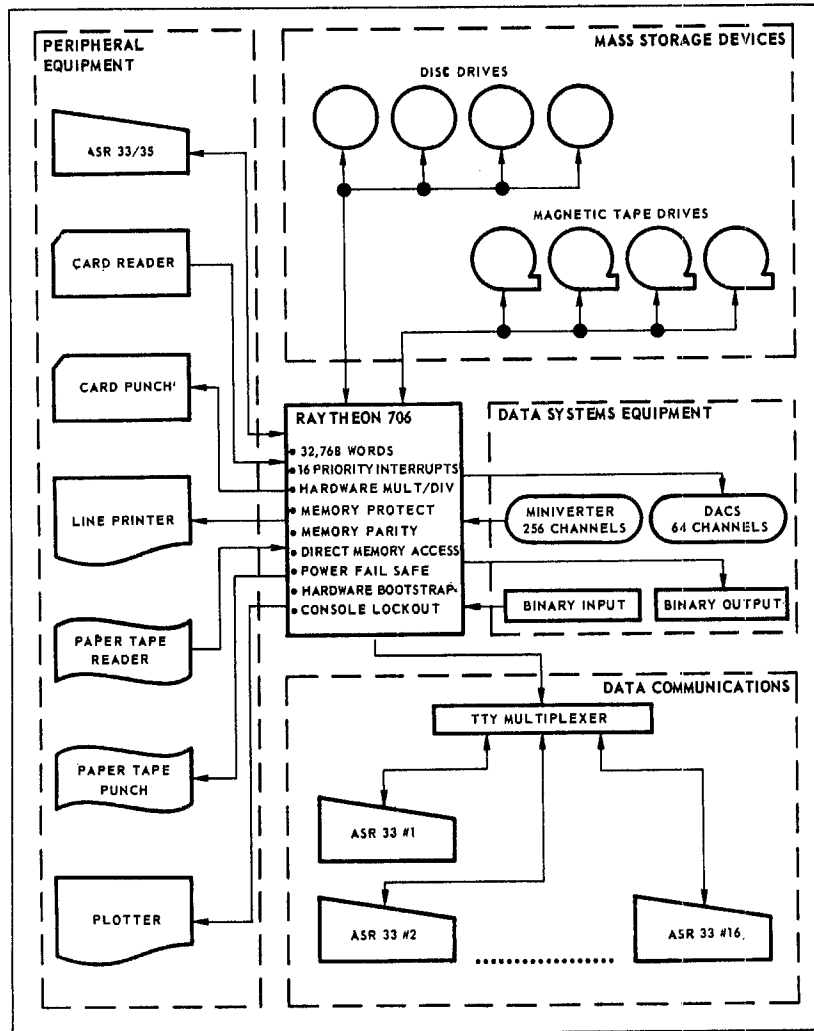


Figure 1-32. Fully Expanded Hardware Configuration

1-6.2.1 X-RAY BASIC

In BASIC systems the Resident Monitor and X-RAY EXEC are combined into a single program and made resident at the low end of core. The BASIC X-RAY EXEC has 12 directives to assist the programmer in preparation, debugging, and execution of his application programs. Directives control I/O device reassignment, program loading,

data initialization, data dump, alteration of memory, punching memory, program transfer, etc. Program dumps can be initiated dynamically through program calls. The entire system occupies the lower 930 memory locations, while the I/O monitor uses the lower 620 locations. Users may choose to overlay the X-RAY EXEC with applications programs or data.

3.2.2 X-RAY STANDARD

In STANDARD systems the Resident Monitor resides at the top of memory and is separate from the X-RAY EXEC which is loaded as a relocatable program. System directives allow the X-RAY EXEC to define itself as resident or non-resident. The size of the Resident Monitor ranges from 750 to 1750 locations depending on the number and type of I/O drivers required.

6.2.3 Magnetic Tape Operating System

The Magnetic Tape Operating System (MTOS) for the 706 is a 1, 2, 3, or 4 magnetic tape based system, which provides for rapid response to real-time events or external requests concurrent with automatic batch processing on a time available basis. All features of the RTOS apply to the MTOS except that the MTOS is operationally much slower than the RTOS due to the random access feature of discs versus the sequential access feature of magnetic tapes.

6.2.4 Real-Time Operating System

The Real-Time Operating System for the 706 is a disc-based operating system, which provides for rapid response to real-time events or external requests concurrent with automatic batch processing on a time-available basis. RTOS is an integral hardware/software system which has been designed to fully utilize a fast, fixed head disc storage, the automatic priority interrupt system, and the powerful input/output structure of the 706.

The center of the system is a compact resident monitor whose size ranges from 1000 to 2048 words depending on the number and type of peripherals in the system. The resident monitor is located at the top of memory and includes a file-oriented, device independent input/output software system, as well as service routines for queue maintenance, interrupt processing, operator interrupt and operator recovery. A disc-based library is organized for rapid response loading of both absolute programs (about 50 ms total load time) and relocatable programs (between 1300 and 4000 words per second).

A non-resident Real-Time X-RAY Executive controls the system configuration and manages the execution of program tasks through a system

queue with 256 priority levels. With over 40 control directives the operator or user may make I/O device reassignments, add or delete tasks in the system queue, reconfigure the resident portion of the system, connect or disconnect tasks to interrupts, and control the execution of programs. The ability to connect programs to interrupts and define resident programs, subroutines, and data blocks provides the user with a flexible means of configuring his real-time systems.

1-6.2.5 Multi-Programming System

The Multi-Programming System (MPS) provides the ADVANCED system user all the capabilities of RTOS plus facilities for dynamic task swapping and hardware inter-task protection. Dynamic task swapping is initiated whenever a higher priority non-resident task is entered into the system queue and insufficient memory is available in which to load the task. At this time the current task in execution is saved on the disc and the higher priority task is loaded and executed. High priority tasks can be entered into the queue by an external interrupt, the operator, or through dynamic call by the program currently in execution. The MPS resident monitor, in conjunction with the hardware memory protect option, protects the Resident Monitor and other programs and data in memory from inadvertent destruction by the current task. The MPS Resident Monitor occupies between 3000 and 4096 words of memory, depending on the number of I/O drivers and the amount of memory allocated to the system queue.

1-6.3 Program Generation

1-6.3.1 Symbolic Program Preparation

Symbolic assembly programs can be prepared on-line under computer supervision with PREP. Free form source statements are accepted from the ASR keyboard and dynamic syntax diagnostics are performed. Immediate corrections to indicated errors can be made. Statements are buffered in memory and output onto the selected device. PREP automatically inserts statement numbers in order to facilitate subsequent editing. The corrected source program can be processed by either assembler — SYM I or SYM II.

1-6.3.2 FORTRAN PREP

Free form source statements for Conversational FORTRAN can be composed on-line from the ASR keyboard with compiler assistance. Statements are diagnosed on-line and errors can be corrected immediately. Error-free statements are retained in memory and output to the selected device. Statement numbers are automatically inserted in the output to facilitate subsequent editing. The product, a diagnosed FORTRAN program, can be compiled with minimum turn-around time, since compiler detected errors have already been removed. Subsequent debugging of the executable FORTRAN programs is made easy by a TRACE option in the FORTRAN run-time system.

1-6.3.3 Symbolic Debugging

TRACE/DEBUG is a comprehensive utility program which provides 15 functions for assisting the user in the checkout of his programs. Functions include provisions to list selected portions of memory, modify selected memory locations, punch or load absolute programs, perform hexadecimal arithmetic, initialize data arrays, search memory, and trace a program. Functions can be selected directly via input directive or dynamically through subroutine calls.

The trace may be complete or selective; the trace range may be set within specified limits; and the user may optionally halt or continue after each trace output. Instructions may be traced selectively. For example, the user may choose to trace only JSX instructions. A masked memory search permits the user to search memory for a specific data value or set of values.

1-6.3.4 Symbolic Program Editor

The Symbolic Program Editor provides a convenient means of revising SYM I, SYM II, FORTRAN, and Real-Time FORTRAN IV programs under computer supervision. Editor directives provide for the insertion, deletion, or replacement of symbolic program statements. Only changes need be specified—unchanged statements are reproduced exactly as they appeared in the old program. The program operates in two phases; in the directive phase all revisions and new statements are input; during the edit phase the old source text is read, revised according to the directives previously input, and a new source

program is output. The editor automatically inserts statement numbers in the new program. As an option a listing of the new program can be obtained. Since the editor operates under control of the X-RAY EXEC, the user may assign input and output devices to any desired configuration.

1-6.4 Assemblers and Compilers

1-6.4.1 SYM I

For BASIC systems with only 4K memory and ASR 33 the one-pass SYM I assembler provides unparalleled capabilities. The assembler is truly one-pass in that it only requires one pass of the source program to produce a source listing, object listing, and object tapes. Many so-called "one-pass" assemblers require one pass for each output. Object program output may be either relocatable or absolute. Advanced assembler features include forward defined symbols, 11 pseudo operations including TRUE/FALSE conditional directives, operand expressions, and MACRO subroutine calls.

Prior to assembly source input and object output devices can be assigned by control directives. The assembler includes a PREP option which allows the user to prepare his program in the conversational mode with assembler assistance. Another option permits programs to be assembled directly into memory for immediate execution without requiring a load phase. Multiple assemblies or program PREP's can be run without reloading SYM I. The entire SYMI/PREP program, including all I/O, occupies only 2460 locations. Thus, the user can assemble absolute programs with as many as 1000 instructions and 100 symbols and relocatable programs with 500 instructions and 100 symbols.

1-6.4.2. SYM II

The SYM II assembler is a fast and powerful two-pass system with procedural capabilities. SYM II was designed for efficient operations in larger system configurations; STANDARD, EXTENDED, and ADVANCED. The assembler processes statements internally at rates in excess of 3000 statements per minute for each pass and its throughput rate is only limited by the speed of the system peripherals. All input/output is double buffered for efficient throughput. In STANDARD systems the source statements are recirculated for

the second pass; thus eliminating the time consuming output of intermediate text by pass 1. In EXTENDED and ADVANCED systems the disc is used to provide automatic assemble and go operation. Options permit selective output of binary text, assembly listing, and errors only listing. The latter is particularly effective in systems which do not include a medium or high speed printer.

SYM II produces either absolute or relocatable output. Procedures, subroutine macros, and twenty two pseudo operations simplify programming of the 706. Debugging aids, include extensive documentation; such as, alphabetical symbol dictionary, literal list, and external names list. Conversion routines for all 706 math formats are included: integer, double integer, real, mid-precision, and double precision. Multiple assemblies can be run without reloading SYM II. All input/output device assignments are controlled by the X-RAY EXEC. The entire SYM II occupies 4600 locations.

1-6.4.3 Conversational FORTRAN

Conversational FORTRAN is designed specifically for users of the small system where effective programs must be developed without knowledge of the computer's internal structure. Conversational FORTRAN is a one-pass compiler which compiles programs directly into memory providing rapid compile and go capabilities. In BASIC and STANDARD systems programs can be prepared and executed in four simple steps: load the compiler, input source statements, load the run-time library (the library overlays the compiler), and execute.

In EXTENDED and ADVANCED systems the entire process is automated by batch control directives. The compiler includes a PREP option which allows the user to prepare his program on-line in the conversational mode. The run-time system includes a TRACE option to facilitate program checkout. Provisions are also included for interfacing machine language routines for data acquisition and real-time applications. The compiler is compact, requiring only 3024 locations including I/O. A proprietary code generation technique optimizes the amount of memory used by the object program. As a result, the user can compile and execute large programs (about 125 statements in a 4K computer and 500 statements in an 8K computer).

1-6.4.4 Real-Time FORTRAN IV

Real-Time FORTRAN IV is an extended superset of ASA Standard FORTRAN. Extended features include:

- Relaxation of syntax restrictions.

- Real-time capabilities.

- Mixed expressions and compound statements.

- N-dimensional arrays and generalized subscripts.

- Multiple replacements in statements.

- Mid-precision floating point numbers.

- Double-precision floating point numbers. Extended relational operators and expressions.

- In-line symbolic machine instructions.

The compiler in one-pass produces symbolic text for input to the SYM II assembler at internal rates in excess of 1200 statements per minute. All inputs and outputs are double buffered so that compilation speeds are only limited by the user's peripheral devices.

Thorough diagnostics provide 68 unique error messages. Each message is accompanied by an arrow which points to the most probable character in error within the statement. A proprietary method of code generation minimizes object code produced by the compiler.

Two versions of the compiler are available. A subset, which includes all capabilities except the DATA and EQUIVALENCE statements, occupies 6400 memory locations and operates in 8K STANDARD and EXTENDED systems. The complete compiler uses 7300 memory locations and operates in ADVANCED systems, as well as 12K STANDARD and EXTENDED systems. Both versions provide automatic compile and go operations in EXTENDED and ADVANCED systems.

For real-time applications 706 FORTRAN IV provides re-entrant and recursive operation and allows in-line symbolic code for ease of interface to

data acquisition or other system equipment. Further extension of real-time capabilities is provided by RTOS and MPS; such as, assignment of resident subroutines and data, program segmentation, connection of tasks to priority interrupts, and dynamic queuing of programs.

1-6.5 Library Programs

1-6.5.1 System Generation

Several system processors are provided for preparing and editing paper tape, magnetic tape and disc libraries. Two System Generation programs, SYSGEN 1 and SYSGEN 2, provide automatic generation of the absolute and relocatable disc libraries in EXTENDED and ADVANCED systems. A library extension processor (EXTEND) permits programs to be replaced or added to the disc library after system generation. A System Editor maintains libraries on magnetic tape or paper tape.

1-6.5.2 Loaders

Relocatable and absolute loaders are provided for all 706 software systems, even BASIC. The absolute loaders input core image programs directly into memory. Relocatable programs are input in relocatable loader text format. The loader processes the text at load time in order to convert the program into core image format in the locations in memory. In this way programs and subroutines are relocating in memory at load time with the loader automatically allocating memory for optimum utilization of word and byte pages.

The BASIC loader provides a convenient means of selectively loading relocatable programs and subroutines in BASIC hardware configurations which contain no system library facilities. Loader options are controlled via X-RAY EXEC directives. The STANDARD loader features automatic library search, augmented memory allocation, and relocatable correction records. Options include printing a load map, loading TRACE/DEBUG, and executing despite missing subroutines. The loader first loads a complete program in the primary or non-selective phase and then enters the selective library phase in which only those programs referenced are loaded from the library. The library is read from a high speed paper tape reader, card reader, or magnetic tape.

1-6.5.3 Disc Loaders

EXTENDED and ADVANCED systems utilize absolute and relocatable disc loaders. RTOS and MPS both contain two libraries on the disc; the processor library and the relocatable library. The processor library contains programs which are in core image format, while the relocatable library contains those programs and subroutines whose locations in memory are determined by the system loader at load time. Each library has its own directory which contains a list of the programs and their location on the disc. In this way programs are located directly without a time-consuming serial search process.

The processor library contains frequently used programs which require very rapid loading. Only two disc accesses are required to load a processor. Once a processor entry is found in the directory, loading is accomplished with a single direct memory access operation. This is possible because the program is in core image format. Thus, absolute programs are loaded in approximately 50 milliseconds. Most major system programs; such as, the Resident Monitor, the X-RAY Executive, FORTRAN IV, SYM II, etc. are located in the processor library. The user may choose to locate various application programs which require rapid loading in the processor library.

In the relocatable library programs and subroutines are stored on the disc in relocatable loader text format. The system loader processes the text at load time, in order to convert the program to core image format in memory. The programs are found by means of a directory. The loader text records for each program are spread around each track so that access time is minimized and loader text processing is efficiently overlapped with disc input time. As a result, the system loader loads between 1300 and 4000 words per second on 706 systems. Actual load times will vary somewhat depending on the size of the program units and the number of external references. Small subroutines will tend to load at the smaller figure because of initial disc access time, while large programs will tend to load at the higher figure.

1-6.5.4 Conversational FORTRAN Library

The library for Conversational FORTRAN contains 16 intrinsic and external functions including all those specified in ASA Basic FORTRAN. In addition, logical functions have been added to facilitate internal operations on integer control

data. The entire library, including formatted and unformatted I/O, occupies 2750 locations of memory. The floating point math routines and functions provide nine decimal digits of accuracy. All routines have been machine language coded for fast execution speeds.

A run-time trace option makes program checkout easy. The trace has three options: data only, control only, or both. In the data mode an integer or floating point value is printed for each replacement statement which is executed. The control mode traces the GO TO, IF, CALL, RETURN, DO, and CONTINUE statements. The FORTRAN program can be interrupted during execution in order to initiate, terminate, or change trace modes.

1-6.5.5 Real-Time FORTRAN IV Library

The Real-Time FORTRAN IV Library is both comprehensive and fast. The library contains 107 subroutines which perform 74 intrinsic and external functions, including all functions specified in ASA Standard FORTRAN and 19 additional functions. Three floating point formats are available: Real with 6 decimal digit accuracy, Mid-precision with 9 decimal digit accuracy and Double-precision with 12 decimal digit accuracy. Exponents for all floating formats include complex, integer, double integer, logical, and hollerith.

All functions are designed for fast execution in real-time environments. The routine can be made re-entrant and recursive as a run-time option by an X-RAY EXEC control directive. As a result only one set of library routines is required. Execution times in the recursive mode are almost as fast as in the non-recursive mode, since temporary data is saved in a dynamic list only after it has been determined that a routine is being re-entered. The entire FORTRAN IV library has been coded in machine language with algorithms chosen for fast execution. For example, the mid-precision square root and sine execute in .24 and 2.03 milliseconds, respectively. All routines can be used in SYM II assembly language programs, as well as FORTRAN IV programs.

Formatted and unformatted output is double buffered so that computation is overlapped with data outputs. This is particularly important if the users list device is a teletype or low speed printer. The unformatted input/output routines provide for

simulation of magnetic tape I/O on disc in disc-based systems. A thorough set of run-time error messages identify argument errors, overflow, computed subscripts outside dimension range, data format incompatibilities, data transmission errors, etc.

1-6.5.6 Math Library

A comprehensive set of 37 mathematical operations for SYM I and SYM II users are included in the Math Library. The library includes Double Precision Fixed Point and Floating Point Packages, in addition to Fixed Point multiply/divide subroutines and Binary/BCD conversion routines. No additional hardware is required for operation of the above routines, although allowance is made for the hardware MPY/DIV option for significant increased speed of associated double precision/floating point operations. A precision of 9 decimal digits is provided by both double precision fixed point and floating point formats. The problems of fixed point scaling are eliminated with floating point operation and the programmer can take advantage of its wide numerical range, $10^{\pm 38}$.

A complete set of very fast floating point trigonometric and transcendental functions are also provided. Scaling factors associated with fixed point functions are eliminated and the full range of $10^{\pm 38}$ is incorporated with a precision of 9 decimal digits. Functions include sine, cosine, square root, arctangent, exponential, hyperbolic, tangent, and natural logarithm.

1-6.6 Sensor Diagnostics

Unique in the 706 class of machines, SENSOR hardware diagnostics test and verify 706 operational integrity. SENSOR programs and procedures are organized and documented to simplify the job of maintenance personnel and reduce system down time to minutes.

The SENSOR package includes technical programs and procedures designed to detect and isolate malfunctions in the central processing unit. When a malfunction occurs, whether it's major or minor, or in the event of routine system testing, diagnostic programs are bootstrap loaded into the CPU. SENSOR procedures provide fault detection and isolation instructions in the event bootstrap loading is unsuccessful. Once loaded, error

messages typed by the diagnostic program, and use of technically annotated program listings with fault isolation information, enable rapid repair of faulty components. Typically, CPU faults are isolated to six or fewer integrated circuits and, in many instances to an individual IC element. A simple light probe, available in all 706 computers, in conjunction with annotated program listings, assists maintenance personnel with detailed fault isolation.

The fourth generation packaging concept of the 706 processor with dual in-line monolithic TTL integrated circuits plugged directly into a single master module provides a unique capability. A faulty component can be quickly removed from its individual receptacle and replaced without soldering or wiring changes. Any signal checking of the processor is limited to the standardized signals of the IC's themselves. The hundreds of unique signals associated with plug-in circuit cards and the tedious matching of test point to signal list is eliminated.

Also included in the SENSOR package are numerous performance test programs designed to test the proper operation of 706 peripherals. These

programs provide software performance test capability as they exercise the X-RAY EXEC, I/O drivers and other basic software elements in the same way operational software utilizes the system. These programs are listed below:

- Interrupt System
- ASR 33/35
- High-speed Paper Tape Reader
- High-Speed Paper Tape Punch
- Disc
- 9-Track Magnetic Tape
- 7-Track Magnetic Tape
- Line Printer
- Miniverter/Multiverter
- Teletype Multiplexer
- Card Reader
- Card Punch
- Plotter

Section 2

INSTRUCTIONS

2-1 HARDWARE

This section defines the function of each hardware instruction and its effect on the various registers and memory of the machine. Unless a register or the memory is specifically mentioned, it is unaffected by the instruction. The equation terms and symbols are defined in Table 2-1.

2-1.1 General

The instructions for the 706 processor are divided into ten classes. The groups are divided based upon the instruction formats and not their functions. The classes from one to ten are: Word Reference, Byte Reference, 4-bit Operand, Shift, Halt, No Operand, Byte Literal, Input/Output, Multiply and Divide and Set Memory Bank. Appendices O, P, and Q list the instructions alphabetically, numerically, and functionally.

Included with every mnemonic description is the timing (in cycles) of the instruction, the equation representing the operation of the instruction, the values of the variable fields, special conditions (i.e., affect on overflow), a verbal description of the instruction, and an example of the use of the instruction.

The example following each verbal description has the following format:

$AAAA_{16}^*$	$BBBB_{16}$	CCC	DDD
(instruction location)	(Hex code)	(mnemonic)	(operand)

	(ACR)	(IXR)	$(DDD) \dots$
Initial	ACR_i	IXR_i	$DDD_i \dots$
Final	ACR_f	IXR_f	$DDD_f \dots$

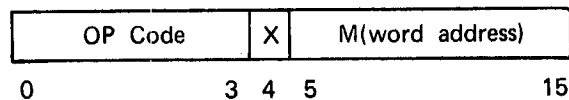
*Instruction locations are shown for JUMP and SKIP type of instructions only.

NOTE: Except for the operand, all numbers shown are in hexadecimal notation unless otherwise specified.

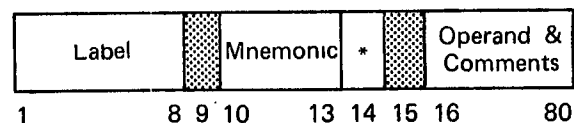
2-1.2 Class 1 - Word Reference

Class 1 instructions reference words by building an effective address, M, from the extension register, index register, and the 11-bit operand address.

Object Format:



Source Format:



*Specifies indexing

JMP Unconditional Jump
 T: 1
 Equation: $(PCR) \leftarrow M_w$
 OP Code: 1
 Indexing: Permitted

The effective word address replaces the contents of the program counter. Execution of this instruction copies the local page address to the extension register.

Table 2-1. Definition of Instruction Equation Terms and Symbols

Terms and Symbols	Definition	Terms and Symbols	Definition
T: n	Timing: n cycles	{Expression}	The logical value of the expression
(ACR)	The contents of the accumulator	(ADFOVF)	The contents of the overflow flip flop
(IXR)	The contents of the index register	(CCFGLB)	The contents of the global flip flop
(M _W)	The contents of the memory word location M	(CCFUSR _i)	The contents of the user mode flip flops; where i = 0, 1, 2 or 3
(PCR)	The contents of the program counter	(DMFPCR)	The contents of the DMA parity error flip flop
(M _B)	The contents of the memory byte location M	(MMFPCR)	The contents of the CPU parity error flip flop
(EXR)	The contents of the extension register	(MPFPRV)	The contents of the privileged instruction flip flop
M	The memory address field of the instruction	(ITR _{ij} N)	The contents of the interrupt enable flip-flop; where ij ranges from 00-15
(X) _{n-m}	The n th through m th bits of the contents of the register or memory location X	(ITR _{ij} A)	The contents of the interrupt active flip flop; where ij ranges from 00-15
(X) _{n-m}	The absolute value of the n th through m th bits of the contents of the register or memory location X	(ITFINH)	The contents of the interrupt inhibit flip flop
A ← B	A is replaced by B	SSE	External Sense Line
+	Arithmetic sum	SSi	Sense Switch i, i = 0-3
-	Arithmetic difference	(MPRL)	The contents of the Memory Protect Lower Register
X	Arithmetic Multiplication	(MPRU)	The contents of the Memory Protect Upper Register
÷	Arithmetic Division	(DIN)	The contents of the Direct Input Bus
∧	Logical AND	(DOT)	The contents of the Direct Output Bus
∨	Logical OR	\$	Location counter designation
⊕	Logical EXCLUSIVE OR (logical difference)	*	Indexing when in column 14, Comment when in column 1
(\bar{X})	Logical inversion of the contents of X	X''	Hexadecimal integer
(ADFNEG)	The contents of the "negative" comparison flip flop	"	Alphanumeric constant
(ADFEQL)	The contents of the "equals" comparison flip flop	/	Byte location
[Expression]	The arithmetic value of the expression	?	Variable field for DO directive

Example:

0100 1104 JMP X '104'

	(PCR)
Initial	0100
Final	0104

JSX Jump and Store Return in Index

T: 1

Equation:

$$(IXR) \leftarrow (PCR) + 1; (PCR) \leftarrow M_W$$

OP Code: 2

Indexing: Permitted

The contents of the index register are replaced by the contents of the program counter. The contents of the program counter are replaced by the effective word address. Execution of this instruction copies the local page address from the memory address register to the extension register after the generation of the effective word address. Execution of this instruction forces the computer into global addressing mode.

Example:

0200 2104 JSX 260

	(PCR)	(IXR)
Initial	0200	0
Final	0104	0201

STX Store Index

T: 2

$$Equation: (M_W) \leftarrow (IXR)$$

OP Code: 6

Indexing: Permitted

The contents of the index register replace the contents of the memory word location specified by the effective address. The contents of the index register remain unchanged. Execution of this instruction copies the local page address to the extension register.

Example:

6300 STX X '300'

	(IXR)	(300)
Initial	+ 43	-20
Final	+ 43	+ 43

STW Store Word

T: 2

$$Equation: (M_W) \leftarrow (ACR)$$

OP Code: 7

Indexing: Permitted

The contents of the accumulator replace the contents of the memory word location specified by the effective address. The contents of the accumulator remain unchanged. Execution of this instruction copies the local page address to the extension register.

Example:

7300 STW BILL

	(ACR)	(300)
Initial	+ 1234	+ 5678
Final	+ 1234	+ 1234

Note: The expression "BILL" is equivalent to hex location 300 in this example.

LDW Load Word

T: 2

$$Equation: (ACR) \leftarrow (M_W)$$

OP Code: 8

Indexing: Permitted

The contents of the memory word location specified by the effective address replace the contents of the accumulator. The contents of memory remain unchanged. Execution of this instruction copies the local page address to the extension register.

Example:

8300 LDW 768

	(ACR)	(300)
Initial	FFFF	0
Final	0	0

LDX Load Index

T: 2

Equation: $(IXR) \leftarrow (M_W)$

OP Code: 9

Indexing: Permitted

The contents of the memory word location specified by the effective address replace the contents of the index register. The contents of memory remain unchanged. Execution of this instruction copies the local page address to the extension register.

Example:

9302 LDX BILL + 2

	(IXR)	(302)
Initial	0	+ 4
Final	+ 4	+ 4

Note: The expression "BILL" is equivalent to hex location 300 in this example.

ADD Add

T: 2

Equation $(ACR) \leftarrow (ACR) + (M_W)$

OP Code: A

Indexing: Permitted

The arithmetic sum of the original contents of the accumulator plus the contents of the memory word location specified by the effective address replaces the contents of the accumulator. The contents of memory remain unchanged. Execution of this instruction copies the local page address to the extension register. The overflow flip flop is set if the result of an addition is less than -2^{15} or greater than $2^{15}-1$.

Example:

A9FE ADD * X'01FE'

	(ACR)	(IXR)	(200)
Initial	+ 327	2	+ 20
Final	+ 347	2	+ 20

SUB Subtract

T: 2

Equation: $(ACR) \leftarrow (ACR) - (M_W)$

OP Code: B

Indexing: Permitted

The arithmetic difference between the original contents of the accumulator minus the contents of the memory word location specified by the effective address replaces the contents of the accumulator. The contents of memory remain unchanged. Execution of this instruction copies the local page address to the extension register. The overflow flip flop is set if the result of a subtraction is less than -2^{15} or greater than $2^{15}-1$.

Example:

B100 SUB X '100'

	(ACR)	(100)
Initial	+004F	+001C
Final	+0033	+001C

ORI Inclusive OR

T: 2

Equation: $(ACR) \leftarrow (ACR) \vee (M_W)$

OP Code: C

Indexing: Permitted

The logical sum of the contents of the accumulator and the contents of memory word location specified by the effective address replaces the contents of the accumulator. The contents of memory remain unchanged. Execution of this instruction copies the local page address to the extension register.

Example:

C200 ORI 512

	(ACR)	(200)
Initial	041C	740F
Final	741F	740F

ORE Exclusive OR
 T: 2
 Equation: $(ACR) \leftarrow (ACR) \oplus (M_W)$
 OP Code: D
 Indexing: Permitted

The logical Exclusive OR between the contents of the accumulator and the contents of the memory word location specified by the effective address replaces the contents of the accumulator. The contents of memory remain unchanged. Execution of this instruction copies the local page address to the extension register.

Example:

D700 ORE X '700'

	(ACR)	(700)
Initial	1234	5F6C
Final	4D58	5F6C

CMW Compare Word
 T: 2
 Equations:

$$(ADFNEG) \leftarrow \{ (ACR) - (M_W) < 0 \}$$

$$(ADFEQL) \leftarrow \{ (ACR) - (M_W) = 0 \}$$

OP Code: F
 Indexing: Permitted

The contents of the memory word location specified by the effective address are compared to the contents of the accumulator. The result of the comparison is stored in the comparison register specifying whether the contents of the accumulator are less than, equal to, or greater than the contents of the memory location specified by the effective address. Neither the contents of the accumulator nor the contents of memory are affected. Execution of this instruction copies the local page address to the extension register. Words are treated as signed two's complement 16-bit numbers.

Example:

F020 CMW X '020'

	(ADFNEG)	(ADFEQL)	(ACR)	(020)
Initial	0	0	1234	1234
Final	0	1	1234	1234

AND Logical AND
 T: 2
 Equation: $(ACR) \leftarrow (ACR) \wedge (M_W)$
 OP Code: E
 Indexing: Permitted

The logical product of the original contents of the accumulator and the contents of the memory word location specified by the effective address replaces the contents of the accumulator. The contents of memory remain unchanged. Execution of this instruction copies the local page to the extension register.

Example:

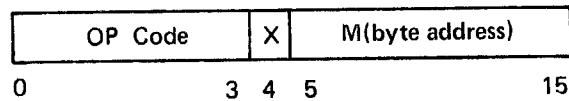
E200 AND X '200'

	(ACR)	(200)
Initial	1234	5F6C
Final	1224	5F6C

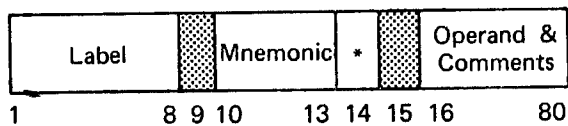
2-1.3 Class 2 - Byte Reference

Class 2 instructions reference bytes by building an effective address, M, from the extension register, index register, and the 11-bit operand address.

Object Format:



Source Format:



*Specifies indexing

STB Store Byte

T: 2
Equation: $(M_B) \leftarrow (ACR)_{8-15}$
OP Code: 3
Indexing: Permitted

The contents of bits 8-15 of the accumulator replace the contents of the memory byte location specified by the effective address. The contents of the accumulator remain unchanged. Execution of this instruction copies the local page address to the extension register.

Example:

3020 STB X '020'

	(ACR)	(010)
Initial	1234	5678
Final	1234	3478

CMB Compare Byte

T: 2
Equation: $(ADFNEG) \leftarrow \{(ACR)_{8-15} - (M_B) < C$
 $(ADFEQL) \leftarrow \{(ACR)_{8-15} - (M_B) = 0$
OP Code: 4
Indexing: Permitted

The contents of the memory byte location specified by the effective address are compared to the contents of bits 8-15 of the accumulator. The result of the comparison is stored in a comparison register specifying whether the contents of bits 8-15 of the accumulator were less than, equal to, or greater than the contents of the memory byte position specified by the effective address. Neither the contents of the register or memory are affected. Execution of this instruction copies the local page address to the extension register. Bytes are treated as signed two's complement 8-bit numbers.

Example:

4021 CMB X '021'

	(ADFNEG)	(ADFEQL)	(ACR)	(010)
Initial	0	0	1234	5634
Final	0	1	1234	5634

LDB Load Byte

T: 2
Equation: $(ACR)_{8-15} \leftarrow (M_B)$
OP Code: 5
Indexing: Permitted

The contents of the memory byte location specified by the effective address replace the contents of bits 8-15 of the accumulator. The contents of memory remain unchanged. Execution of this instruction copies the local page address to the extension register.

Example:

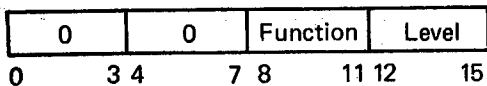
5800 LDB * 0

	(ACR)	(IXR)	(0010)
Initial	1234	0020	5678
Final	1256	0020	5678

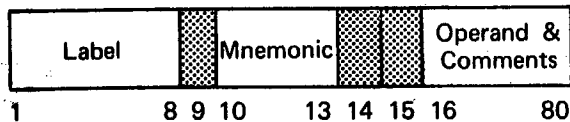
2-1.4 Class 3 - Generic with 4-bit Operand

Class 3 instructions control the interrupt system, set memory protect state and set the extension register. The operands are limited to values that do not exceed 15_{10} .

Object Format:



Source Format:



No Indexing Permitted

INR Interrupt Return

T: 3

Equations: $(PCR) \leftarrow ((Level \times 4))$
 $(EXR) \leftarrow (((Level \times 4) + 2))_{0-4}$
 $(ADFN EG) \leftarrow (((Level \times 4) + 2))_5$
 $(ADFEQL) \leftarrow (((Level \times 4) + 2))_6$
 $(ADFOVF) \leftarrow (((Level \times 4) + 2))_7$
 $(CCFGLB) \leftarrow (((Level \times 4) + 2))_8$
 $(CCFUSRO) \leftarrow (((Level \times 4) + 2))_9$
 $(CCFUSR1) \leftarrow (((Level \times 4) + 2))_{10}$
 $(CCFUSI 2) \leftarrow (((Level \times 4) + 2))_{11}$
 $(CCFUSR3) \leftarrow (((Level \times 4) + 2))_{12}$
 $(MMFPCER) \leftarrow (((Level \times 4) + 2))_{13}$
 $(DMFPCER) \leftarrow (((Level \times 4) + 2))_{14}$
 $(MPFPRV) \leftarrow (((Level \times 4) + 2))_{15}$
 $(ITR_{[Level]N}) \leftarrow 1$ } Idle State (Only
 $(ITR_{[Level]A}) \leftarrow 0$ } if interrupt is in
 active state)

Function: 1

The contents of the program counter are replaced by the contents of the memory location specified by bits 12-15 of the instruction word multiplied by 4. The machine status is restored (except for MMFPCER, DMFPCER, and MPFPRV which are reset upon entering the interrupt service routine and remain reset until another error occurs) and the interrupt level specified in bits 12-15 of the instruction is returned from the active to the idle state. (See Section 4.)

Example:

0100 0012 INR 2

	(PCR) (008)	Level 2 Int.	Status
Initial	0100 0200	Active	
Final	0200 0200	Idle	Restored

ENB Enable Interrupt

T: 1

Equations:

$(ITR_{[Level]N}) \leftarrow 1$ } Idle State (Only if
 $(ITR_{[Level]A}) \leftarrow 0$ } interrupt is in disabled
 state).

Function: 2

The interrupt level specified in bits 12-15 of the instruction word is permitted to respond to interrupts by putting the interrupt flip-flops into the idle state.

Example:

0021 ENB 1

	Level 1 Int.
Initial	Disabled
Final	Idle

DSB Disable Interrupt

T: 1

Equations:

$(ITR_{[Level]N}) \leftarrow 0$ } Disabled State
 $(ITR_{[Level]A}) \leftarrow 0$ }

Function: 3

The interrupt level specified in bits 12-15 of the instruction word is prevented from responding to interrupts by putting the interrupt flip-flops into the disabled state.

Example:

0030 DSB 0

	Level 0 Int.
Initial	Any State
Final	Disabled

SML Select Memory Lower
T: 1

Equations:

$$\begin{aligned} (\text{EXR})_0 &\leftarrow 0 \\ (\text{EXR})_{1-4} &\leftarrow [\text{Level}] \end{aligned}$$

Function: 8

The contents of bits 1 to 4 of the extension register are replaced by bits 12-15 of the instruction. Bit 0 of the extension register is set to zero.

Example:

0082 SML 2

	(EXR)
Initial	13
Final	02

SMU Select Memory Upper
T: 1

Equations:

$$\begin{aligned} (\text{EXR})_0 &\leftarrow 1 \\ (\text{EXR})_{1-4} &\leftarrow [\text{Level}] \end{aligned}$$

Function: 9

The contents of bits 1 to 4 of the extension register are replaced by bits 12-15 of the instruction. Bit 0 of the extension register is set to one.

Example:

0095 SMU 5

	(EXR)
Initial	00
Final	15

SUS Select User State (Memory Protect Option;
See Section 6-7)

T: 1

Equations:

$$\begin{aligned} (\text{CCFUSR0}) &\leftarrow [\text{Level}]_{14} \\ (\text{CCFUSR1}) &\leftarrow [\text{Level}]_{15} \end{aligned}$$

After performing SUS and then executing JMP or JSX:

$$\begin{aligned} (\text{CCFUSR2}) &\leftarrow (\text{CCFUSR0}) \\ (\text{CCFUSR3}) &\leftarrow (\text{CCFUSR1}) \end{aligned}$$

Function: C

The contents of bits 14 and 15 of the instruction are copied into the User Pre-State flags (CCFUSR0,1). Then upon execution of a JMP or JSX instruction the User Pre-State flags are copied into the User State Flags (CCFUSR 2, 3). All four flags are part of the machine status word and are stored and restored during interrupt processing.

Example:

00C3 SUS 3

	User Pre-State Flags	
Initial	0	0
Final	1	1

2-1.5 Class 4 - Shift

Class 4 instructions shift the accumulator and index register, the latter is treated as a 16-bit right extension of the former. Two classes of shift instructions, arithmetic and logical are provided in which the shift type is specified by bits 4-7 of the instruction word. Open and circular shifts, as well as their direction, and single or double precision are specified by bits 8-11. All single length shifts affect only the accumulator while double length shifts affect both the accumulator and index register. The shift length is designated by bits 12-15 of the instruction word. The timing of the shift operation involves 1 cycle for access of the instruction and the shifting of up to 5 bits for each additional cycle.

Object Format:

0	Type	Direction	L
3 4	7 8	11 12	15

Type: Bits 4-7 = 9 or A specifies a shift instruction
 Bits 6, 7 = 10 specifies logical/circular shift
 Bits 6, 7 = 01 specifies an arithmetic shift

Direction:

Bit 8 = 1 specifies Byte shift
 Bit 8 = 0 specifies single/double shift
 Bit 9 = 1 specifies circular shift
 Bit 9 = 0 specifies logical shift
 Bit 10 = 1 specifies double shift
 Bit 10 = 0 specifies single shift
 Bit 11 = 1 specifies left shift
 Bit 11 = 0 specifies right shift

L: 0-F (Length of shift)

Note: A shift length of zero (0) results in an effective NO-OP that takes 1 cycle.

Source Format:

Label	Mnemonic	D/L/R	Length & Comments
1	8 9 10	13 14	15 16 80

D: Double Length Shift
 L: Left Byte
 R: Right Byte
 No indexing permitted

SRA Shift Right Arithmetic

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$(ACR)_{0-15} \leftarrow (ACR)_{0-15} \div 2^{[L]}$$

$$(ACR)_0 \leftarrow (ACR)_0$$

Type: 9

Direction: 0

The contents of the accumulator are shifted L positions to the right, as specified by bits 12-15 of the instruction word. The sign bit of the accumulator is unchanged by this operation and is shifted to bit 1 of the accumulator. Bits shifted off the right end of the register are lost.

Example:

0904 SRA 4

	(ACR)
Initial	8246
Final	F824

SLA Shift Left Arithmetic

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$(ACR)_{0-15} \leftarrow (ACR)_{0-15} \times 2^{[L]}$$

$$(ACR)_{15} \leftarrow 0$$

$$(ADFOVF) \leftarrow 1 \text{ if } (ACR)_{0[Final]} \neq (ACR)_{0[Initial]}$$

$$(ADFOVF) \leftarrow 0 \text{ if } (ACR)_{0[Initial]} = (ACR)_{0[Initial]}$$

Type: 9

Direction: 1

The contents of the accumulator are shifted L positions to the left, as specified by bit positions 12-15 of the instruction word. Bits shifted from position 0 are lost and zeros replace the vacated bit positions on the right end of the accumulator. If the value of the sign bit of the accumulator after the shift is different than the value of the sign bit before the shift, the overflow storage flip-flop is set.

Example:

0911 SLA 1

	(ACR)	(ADFOVF)
Initial	8001	0
Final	0002	1

SRA D Shift Right Arithmetic Double

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$\begin{aligned} (ACR, IXR)_{0-15, 0-15} &\leftarrow \\ (ACR, IXR)_{0-15, 0-15} &\div 2^{[L]} \\ (ACR)_0 &\leftarrow (ACR)_0 \end{aligned}$$

Type: 9

Direction: 2

The contents of the accumulator and index register are shifted L positions to the right as specified by bits 12-15 of the instruction word. The sign bit of the accumulator is unchanged by this operation, and is shifted to bit 1 of the accumulator. Bit 15 of the accumulator is shifted to bit 0 of the index register. Bits shifted off the right end of the index register are lost.

Example:

0923 SRA D 3

	(ACR)	(IXR)
Initial	0200	8008
Final	0040	1001

SLA D Shift Left Arithmetic Double

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$\begin{aligned} (ACR, IXR)_{0-15, 0-15} &\leftarrow \\ (ACR, IXR)_{0-15, 0-15} &\times 2^{[L]} \\ (IXR)_{15} &\leftarrow 0 \\ (ADFOVF) &\leftarrow \\ &1 \text{ if } (ACR)_{0[Final]} \neq (ACR)_{0[Initial]} \\ (ADFOVF) &\leftarrow \\ &0 \text{ if } (ACR)_{0[Final]} = (ACR)_{0[Initial]} \end{aligned}$$

Type: 9

Direction: 3

The contents of the accumulator and index register are shifted L positions to the left, as specified by bit positions 12-15 of the instruction word. Bits shifted from bit position 0 of the accumulator are lost and zeros replace vacated bit positions on the right end of the index register. Bit 0 of the

index register is shifted to bit 15 of the accumulator. If the value of the sign bit of the accumulator after the shift is different from the value of the sign bit before the shift, the overflow storage flip flop is set.

Example:

0935 SLA D 5

	(ACR)	(IXR)	(ADFOVF)
Initial	0023	5002	0
Final	046A	0040	0

SRL Shift Right Logical

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$\begin{aligned} (ACR)_{0-15} &\leftarrow (ACR)_{0-15} \div 2^{[L]} \\ (ACR)_0 &\leftarrow 0 \end{aligned}$$

Type: A

Direction: 0

The contents of the accumulator are shifted L positions to the right, specified by bits 12-15 of the instruction word. Bits shifted off the right end of the accumulator are lost and zeros replace the vacated bit positions at the left end of the accumulator.

Example:

0A0F SRL 15

	(ACR)
Initial	FFFF
Final	0001

SLL Shift Left Logical

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$\begin{aligned} (ACR)_{0-15} &\leftarrow (ACR)_{0-15} \times 2^{[L]} \\ (ACR)_{15} &\leftarrow 0 \end{aligned}$$

Type: A

Direction: 1

The contents of the accumulator are

shifted L positions to the left, as specified by bits 12-15 of the instruction word. Bits shifted off the left end of the accumulator are lost and zeros replace the vacated bit positions on the right end of the accumulator.

Example:

0A10 SLL 0

	(ACR)
Initial	1234
Final	1234

SRL D Shift Right Logical Double

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$\begin{aligned} (ACR, IXR)_{0-15, 0-15} &\leftarrow \\ (ACR, IXR)_{0-15, 0-15} &\div 2^{[L]} \\ (ACR)_0 &\leftarrow 0 \end{aligned}$$

Type: A

Direction: 2

The contents of the accumulator and index register are shifted L positions to the right, as specified by bits 12-15 of the instruction word. Bit 15 of the accumulator is shifted into bit 0 of the index register. Bits shifted off the right end of the index register are lost and zeros replace the vacated bit positions on the left end of the accumulator.

Example:

0A2A SRL D X'A'

	(ACR)	(IXR)
Initial	8000	8000
Final	0020	0020

SLL D Shift Left Logical Double

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$\begin{aligned} (ACR, IXR)_{0-15, 0-15} &\leftarrow \\ (ACR, IXR)_{0-15, 0-15} &\times 2^{[L]} \\ (IXR)_{15} &\leftarrow 0 \end{aligned}$$

Type: A

Direction: 3

The contents of the accumulator and index register are shifted L positions to the left, as specified by bits 12-15 of the instruction word. Bits shifted off the left end of the accumulator are lost and zeros replace the vacated bit positions on the right end of the index register. Bit 0 of the index register is shifted to bit 15 of the accumulator.

Example:

0A32 SLL D 2

	(ACR)	(IXR)
Initial	8000	8000
Final	0002	0000

SRC Shift Right Circular

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$\begin{aligned} (ACR)_{0-15} &\leftarrow (ACR)_{0-15} \div 2^{[L]} \\ \vee (ACR)_{[16-L]-15} &\times 2^{[16-L]} \end{aligned}$$

Type: A

Direction: 4

The contents of the accumulator are shifted L positions to the right as specified by bits 12-15 of the instruction word. During this operation bit 15 of the accumulator is shifted to bit 0 in the accumulator.

Example:

0A43 SRC 3

	(ACR)
Initial	8003
Final	7000

SLC Shift Left Circular

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$(ACR)_{0-15} \leftarrow (ACR)_{0-15} \times 2^{[L]}$$

$$\vee (ACR)_{0-[L-1]} \div 2^{[16-L]}$$

Type: A

Direction: 5

The contents of the accumulator are shifted L positions to the left, as specified by bits 12-15 of the instruction word. During this operation bit 0 of the accumulator is shifted to bit 15 of the accumulator.

Example:

0A54 SLC 4

	(ACR)
Initial	8003
Final	0038

SLC D Shift Left Circular Double

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$(ACR, IXR)_{0-15, 0-15} \leftarrow$$

$$(ACR, IXR)_{0-15, 0-15} \times 2^{[L]} \vee$$

$$(ACR)_{0-[L-1]} \div 2^{[32-L]}$$

Type: A

Direction: 7

The contents of the accumulator and index register are shifted L positions to the left, as specified by bits 12-15 of the instruction word. During this operation bit 0 of the index register is shifted to bit 15 of the accumulator and bit 0 of the accumulator is shifted to bit 15 of the index register.

Example:

0A7C SLC D 12

	(ACR)	(IXR)
Initial	1234	5678
Final	4567	8123

SRC D Shift Right Circular Double

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$(ACR, IXR)_{0-15, 0-15} \leftarrow$$

$$(ACR, IXR)_{0-15, 0-15} \div 2^{[L]} \vee$$

$$(IXR)_{[16-L]-15} \times 2^{[32-L]}$$

Type: A

Direction: 6

The contents of the accumulator and index register are shifted L positions to the right, as specified by bits 12-15 of the instruction word. During this operation bit 15 of the index register is shifted to bit zero of the accumulator. Bit 15 of the accumulator is shifted to bit 0 of the index register.

Example:

0A68 SRC D 8

	(ACR)	(IXR)
Initial	8003	4500
Final	0080	0345

SRL L Shift Right Logical Left Byte

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$(ACR)_{0-7} \leftarrow (ACR)_{0-7} \div 2^{[L]}$$

$$(ACR)_0 \leftarrow 0$$

Type: A

Direction: 8

The contents of bits 0-7 of the accumulator are shifted L positions to the right as specified by bits 12-15 of the instruction word. Bits shifted out of bit position 7 are lost and zeros are inserted into bit position 0. The contents of bit positions 8-15 of the accumulator are not affected by this instruction.

Example:

0A82 SRL L 2

	(ACR)
Initial	8145
Final	2045

SLL L Shift Left Logical Left Byte

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$(ACR)_{0-7} \leftarrow (ACR)_{0-7} \times 2^{[L]}$$

$$(ACR)_7 \leftarrow 0$$

Type: A

Direction: 9

The contents of bit positions 0-7 of the accumulator are shifted L positions to the left as specified by bits 12-15 of the instruction word. Bits shifted out of bit position 0 are lost and zeros are inserted into bit position 7. The contents of bit positions 8-15 of the accumulator are not affected by this instruction.

Example:

0A98 SLL L 8

	(ACR)
Initial	FFFF
Final	00FF

SLL R Shift Left Logical Right Byte

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$(ACR)_{8-15} \leftarrow (ACR)_{8-15} \times 2^{[L]}$$

$$(ACR)_{15} \leftarrow 0$$

Type: A

Direction: B

The contents of bit positions 8-15 of the accumulator are shifted L positions to the left as specified by bits 12-15 of the instruction word. Bits shifted out of bit position 8 are lost and zeros are inserted into bit position 15. The contents of bit positions 0-7 of the accumulator are not affected by this instruction.

Example:

0AB1 SLL R 1

	(ACR)
Initial	8201
Final	8202

SRL R Shift Right Logical Right Byte

T: $[L \div 5] + 1$ (Round to largest integer)

Equations:

$$(ACR)_{8-15} \leftarrow (ACR)_{8-15} \div 2^{[L]}$$

$$(ACR)_8 \leftarrow 0$$

Type: A

Direction: A

The contents of bit positions 8-15 of the accumulator are shifted L positions to the right, as specified by bits 12-15 of the instruction word. Bits shifted out of bit position 15 are lost and zeros are inserted into bit position 8. The contents of bit positions 0-7 of the accumulator are not affected by this instruction.

Example:

0AA1 SRL R 1

	(ACR)
Initial	8201
Final	8200

SRC L Shift Right Circular Left Byte

T: $[L \div 5] + 1$ (Round to largest integer)

Equation:

$$(ACR)_{0-7} \leftarrow (ACR)_{0-7} \div 2^{[L]}$$

$$\vee (ACR)_{[8-L]-7}^{[8-L]}$$

Type: A

Direction: C

The contents of bit positions 0-7 of the accumulator are shifted L positions to the right as specified by bits 12-15 of the instruction word. During this operation bit 7 of the accumulator is shifted into bit 0 of the accumulator. The contents of bit positions 8-15 of the accumulator are not affected by this instruction.

Example:

0AC2 SRC L 2

	(ACR)
Initial	A455
Final	2955

SLC L Shift Left Circular Left Byte

T: $[L \div 5] + 1$ (Round to largest integer)

Equation:

$$(ACR)_{0-7} \leftarrow (ACR)_{0-7} \times 2^{[L]}$$

$$V(ACR) \div 2_{0-[L-1]}^{[8-L]}$$

Type: A

Direction: D

The contents of bit positions 0-7 of the accumulator are shifted L positions to the left as specified by bits 12-15 of the instruction word. During this operation bit 0 of the accumulator is shifted to bit 7 of the accumulator. The contents of bit positions 8-15 of the accumulator are not affected by this instruction.

Example:

0AD4 SLC L 4

	(ACR)
Initial	A455
Final	4A55

SLC R Shift Left Circular Right Byte

T: $[L \div 5] + 1$

Equation:

$$(ACR)_{8-15} \leftarrow (ACR)_{8-15} \times 2^{[L]}$$

$$V(ACR) \div 2_{8-[16-L]}^{[8-L]}$$

Type: A

Direction: F

The contents of bit positions 8-15 of the accumulator are shifted L positions to the left as specified by bits 12-15 of the instruction word. During this operation bit 8 of the accumulator is shifted to bit 15 of the accumulator. The contents of bit positions 0-7 of the accumulator are not affected by this instruction.

Example:

0AF5 SLC R 5

	(ACR)
Initial	0436
Final	04C6

SRC R Shift Right Circular Right Byte

T: $[L \div 5] + 1$ (Round to largest integer)

Equation:

$$(ACR)_{8-15} \leftarrow (ACR)_{8-15} \div 2^{[L]}$$

$$V(ACR)_{[16-L]-15}^{[8-L]}$$

Type: A

Direction: E

The contents of bit positions 8-15 of the accumulator are shifted L positions to the right as specified by bits 12-15 of the instruction word. During this operation bit 15 of the accumulator is shifted to bit 8 of the accumulator. The contents of bit positions 0-7 of the accumulator are not affected by this instruction.

Example:

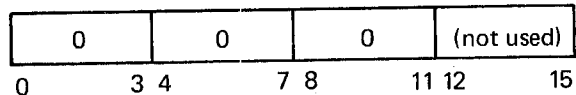
0AE5 SRC R 5

	(ACR)
Initial	0436
Final	04B1

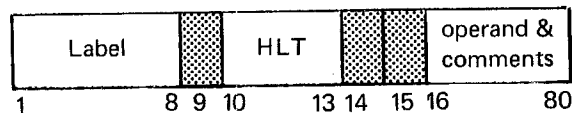
2-1.6 Class 5 - Halt

Class 5 has one instruction—the halt instruction. It has a 4-bit operand field which is not used by the instruction; however, it is useful to the programmer for identifying various halts. The operand field, if used must contain a decimal constant.

Object Format:



Source Format:



Operand may be decimal constant only.

HLT Halt
 T: 1
 Equation: None

The central processing unit is placed in the Halt state. To resume computation the run switch on the control panel must be activated.

2-1.7 Class 6 - Generic with No Address

Class 6 instructions do not use any addresses in their operand fields.

Object Format:

0	Type	Item	(not used)
0	3 4	7 8	11 12 15

when Type = 0; Control
 Type = 1; Accumulator or Index Operation
 Type = 8; Skip
 Type = A; No Operation

Source Format:

Label		Mnemonic			Comments
1	8	9 10	13	14 15	16 80

SLM Set Local Mode
 T: 1
 Equation: (CCFGLB) ← 0
 Type: 0
 Item: 4

The central processing unit is placed in the local addressing mode causing the contents of the extension register (EXR) to be used in forming the base address of indexed instructions.

Example:
 0040 SLM

	(CCFGLB)
Initial	0 or 1
Final	0

SGM Set Global Mode
 T: 1
 Equation: (CCFGLB) ← 1
 Type: 0
 Item: 5

The central processing unit is placed in the global addressing mode. The contents of the extension register (EXR) are not used in forming the base address of indexed instructions. The bit positions normally provided by the EXR are set to 0.

Example:

0050 SGM

	(CCFGLB)
Initial	0 or 1
Final	1

CEX Copy Extension to Index
 T: 1
 Equation: (IXR)₀₋₄ ← (EXR)
 Type: 0
 Item: 6

The contents of bits 0-4 of the index register are replaced by the contents of the extension register. The contents of the extension register and the contents of the remaining bits of the index register remain unchanged.

Example

0060 CEX

	(IXR)	(EXR)
Initial	FFFF	00
Final	07FF	00

CXE Copy Index to Extension
 T: 1
 Equation: (EXR) ← (IXR)₀₋₄
 Type: 0
 Item: 7

The contents of the extension register are replaced by the contents of bits 0-4 of the index register. The contents of the index register remain unchanged.

Example:

0070 CXE

	(EXR)	(IXR)
Initial	00	FFFF
Final	IF	FFFF

MSK Mask Interrupts (See Section 4)

T: 1

Equation: (ITFINH) ← 1

Type: 0

Item: A

The interrupt system is inhibited from processing any interrupts that may occur. The interrupt condition of each level will remain pending (WAIT state) and will be processed when the inhibit condition is removed with a UNM instruction. An interrupt sub-routine in process is not affected by execution of this instruction.

Example:

00A0 MSK

	(ITFINH)
Initial	0 or 1
Final	1

UNM Unmask Interrupts (See Section 4)

T: 1

Equation: (ITFINH) ← 0

Type: 0

Item: B

The interrupt system is unmasked to allow enabled interrupts to be serviced as required.

Example:

00B0 UNM

	(ITFINH)
Initial	0 or 1
Final	0

CLR Clear Accumulator

T: 1

Equation: (ACR) ← 0

Type: 1

Item: 0

The contents of the accumulator are replaced by 0.

Example:

0100 CLR

	(ACR)
Initial	1234
Final	0000

CMP Complement Accumulator

T: 1

Equations: (ACR) ← $[(\overline{ACR}) + 1]$
(ADFOVF) ← $\{(ACR) = 8000_{16}\}$

Type: 1

Item: 1

The contents of the accumulator are replaced by the two's complement of the contents of the accumulator. The overflow flip flop will be set if the number -215 is complemented.

Example:

0110 CMP

	(ACR)
Initial	1234
Final	EDCC

INV Invert Accumulator

T: 1

Equation: (ACR) ← (\overline{ACR})

Type: 1

Item: 2

The contents of the accumulator are replaced by the one's complement of the contents of the accumulator.

Example:

0120 INV

	(ACR)
Initial	1234
Final	EDCB

CAX Copy Accumulator to Index

T: 1

Equation: (IXR) ← (ACR)

Type: 1

Item: 3

The contents of the index register are replaced by the contents of the accumulator. The contents of the accumulator are not affected.

Example:

0130 CAX

	(IXR)	(ACR)
Initial	5678	1234
Final	1234	1234

CXA Copy Index to Accumulator

T: 1

Equation: (ACR) ← (IXR)

Type: 1

Item: 4

The contents of the accumulator are replaced by the contents of the index register. The contents of the index register are not affected

Example:

0140 CXA

	(ACR)	(IXR)
Initial	5678	1234
Final	1234	1234

SAZ Skip on Accumulator Zero

T: 1

Equation: (PCR) ← (PCR) + 1 + {(ACR) = 0}

Type: 8

Item: 0

If the contents of the accumulator are zero, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed. The contents of the accumulator are not affected.

Example:

0100 0800 SAZ

	(PCR)	(ACR)
Initial	0100	0000
Final	0102	0000

SAP Skip on Accumulator Plus

T: 1

Equation:

(PCR) ← (PCR) + 1 + {(ACR) ≥ 0}

Type: 8

Item: 1

If the contents of the accumulator are greater than or equal to zero, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed. The contents of the accumulator are not affected.

Example:

0100 0810 SAP

	(PCR)	(ACR)
Initial	0100	8004
Final	0101	8004

SAM Skip on Accumulator Minus

T: 1

Equation:

(PCR) ← (PCR) + 1 + {(ACR) < 0}

Type: 8

Item: 2

If the contents of the accumulator are less than 0, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed. The contents of the accumulator are not affected.

Example:

0200 0820 SAM

	(PCR)	(ACR)
Initial	0200	8004
Final	0202	8004

SAO Skip on Accumulator Odd

T: 1

Equation: $(PCR) \leftarrow (PCR) + 1 + \{(ACR)_{15} = 1\}$

Type: 8

Item: 3

If the contents of bit 15 of the accumulator is equal to 1, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed. The contents of the accumulator are not affected.

Example:

0200 0830 SAO

	(PCR)	(ACR)
Initial	0200	8004
Final	0201	8004

SLS Skip on Compare Less

T: 1

Equation:

$(PCR) \leftarrow (PCR) + 1 + \{(ADFNEG) = 1\}$

Type: 8

Item: 4

If the contents of the negative comparison flip flop specify that the contents of the accumulator were less than the operand of the last previous compare instruction, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed. Neither the contents of the accumulator nor the contents of the negative comparison flip flop are affected.

Example:

0100 0840 SLS

	(PCR)	(ADFNEG)
Initial	0100	1
Final	0102	1

SXE Skip on Index Even

T: 1

Equation:

$(PCR) \leftarrow (PCR) + 1 + \{(IXR)_{15} = 0\}$

Type: 8

Item: 5

If the contents of bit 15 of the index is equal to 0, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed. The contents of the index register are not affected.

Example:

0100 0850 SXE

	(PCR)	(IXR)
Initial	0100	432F
Final	0101	432F

SXP Skip on Index Positive*

T: $1 + \{(IXR) \geq 0\}$

Equation:

$(PCR) \leftarrow (PCR) + 1 + \{(IXR) \geq 0\}$

Type: 4

Item: 0

Bits 12-15: 0

If the contents of the index register are greater than or equal to 0, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed. The contents of the index register are not affected.

*Skip on Index Positive is a special case of the instruction: IXS, increment the index and skip with the literal value added always equal to 0.

Example:

0300 0400 SXP

	(PCR)	(IXR)
Initial	0300	0004
Final	0302	0004

Example:

0100 0860 SEQ

	(PCR)	(ADFEQL)
Initial	0100	1
Final	0102	1

SXM Skip on Index Minus (Negative)**

T: 1 + {(IXR) < 0}

Equation:

$$(PCR) \leftarrow (PCR) + 1 + \{(IXR) < 0\}$$

Type: 5

Item: 0

Bits 12-15: 0

If the contents of the index register are less than 0, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed. The contents of the index register are not affected.

**Skip on Index Minus is a special case of the instruction: DXS, Decrement the Index and Skip with the literal value always equal to 0.

Example:

0100 0500 SXM

	(PCR)	(IXR)
Initial	0100	0001
Final	0101	0001

SEQ Skip on Compare Equal

T: 1

Equation:

$$(PCR) \leftarrow (PCR) + 1 + \{(ADFEQL) = 1\}$$

Type: 8

Item: 6

If the contents of the comparison storage register specify that the contents of the accumulator were equal to the operand of the last previous compare instruction, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed. The contents of the accumulator are not affected.

SNE Skip on Compare Not Equal

T: 1

Equation:

$$(PCR) \leftarrow (PCR) + 1 + \{(ADFEQL) = 0\}$$

Type: 8

Item: 7

If the contents of the comparison storage register specify that the contents of the accumulator were not equal to the operand of the last previous compare instruction, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed. The contents of the accumulator are not affected.

Example:

0100 0870 SNE

	(PCR)	(ADFEQL)
Initial	0100	0
Final	0102	0

SGR Skip on Compare Greater

T: 1

Equation:

$$(PCR) \leftarrow (PCR) + 1 + \{ \{(ADFEQL) = 0\} \wedge \{(ADFNEG) = 0\} \}$$

Type: 8

Item: 8

If the contents of the comparison storage register specify that the contents of the accumulator were greater than the operand of the last previous compare instruction, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed. The contents of the accumulator are not affected.

Example:

0200 0880 SGR

	(PCR)	(ADFEQL)	(ADFNEG)
Initial	0200	0	0
Final	0202	0	0

Example:

0100 08A0 SNO

	(PCR)	(ADFOVF)
Initial	0100	1
Final	0101	0

SLE Skip on Compare Less Than or Equal

T: 1

Equation:

$$(PCR) \leftarrow (PCR) + 1 + \{ \{ (ADFNEG) = 1 \} \vee \{ (ADFEQL) = 1 \} \}$$

Type: 8

Item: 9

If the contents of the comparison storage register specify that the contents of the accumulator were less than or equal to the operand for the last previous compare instruction, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed. The contents of the accumulator are not affected.

Example:

0200 0890 SLE

	(PCR)	(ADFEQL)	(ADFNEG)
Initial	0200	0	0
Final	0201	0	0

SNO Skip on No Overflow

T: 1

Equations:

$$(PCR) \leftarrow (PCR) + 1 + \{ (ADFOVF) = 0 \}$$

$$(ADFOVF) \leftarrow 0$$

Type: 8

Item: A

If the contents of the overflow storage flip flop is zero the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed. The contents of the overflow storage flip flop is set to zero.

SSE Skip on Sense External

T: 1

$$\text{Equation: } (PCR) \leftarrow (PCR) + 1 + \{ SSE = 0 \}$$

Type: 8

Item: B

If the external sense line is false, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed.

Example:

0100 08B0 SSE

	(PCR)	SSE
Initial	0100	1
Final	0102	1

SSO Skip on Sense Switch 0 False

T: 1

$$\text{Equation: } (PCR) \leftarrow (PCR) + 1 + \{ SSO = 0 \}$$

Type: 8

Item: C

If sense switch zero is false, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed.

Example:

0100 08C0 SSO

	(PCR)	SSO
Initial	0100	0
Final	0102	0

SS1 Skip on Sense Switch 1 False
 T: 1
 Equation: $(PCR) \leftarrow (PCR) + 1 + \{SS1 = 0\}$
 Type: 8
 Item: D

If sense switch one is false, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed.

Example:
 0200 08D0 SS1

	(PCR)	SS1
Initial	0200	1
Final	0201	1

SS2 Skip on Sense Switch 2 False
 T: 1
 Equation: $(PCR) \leftarrow (PCR) + 1 + \{SS2 = 0\}$
 Type: 8
 Item: E

If sense switch two is false, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed.

Example:
 0200 08E0 SS2

	(PCR)	SS2
Initial	0200	0
Final	0202	0

SS3 Skip on Sense Switch 3 False
 T: 1
 Equation: $(PCR) \leftarrow (PCR) + 1 + \{SS3 = 0\}$
 Type: 8
 Item: F

If sense switch three is false, the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed.

Example:
 0100 08F0 SS3

	(PCR)	SS3
Initial	0100	1
Final	0101	1

NOP No operation*
 T: 1
 Equation: $(PCR) \leftarrow (PCR) + 1$
 Type: A
 Item: 0
 Bits 12-15: 0

This instruction does not do anything but cause the next instruction in sequence to be executed.

*No operation is a special case of the instruction: SRL, Shift Right Logical with the shift length equal to 0.

Example:
 0A00 NOP

	(ACR)
Initial	1234
Final	1234

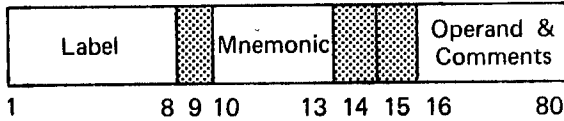
2-1.8 Class 7 – Generic with Byte Operand
 Class 7 instructions operate on their least significant byte literal.

Object Format:

0	Type	M (Literal Byte)
0	3 4	7 8 15

Literal Byte varies from 00 - FF₁₆ (0 - 255₁₀)

Source Format:



IXS Increment Index and Skip

T: $1 + \{(IXR) \geq 0\}$

Equations:

$$(IXR) \leftarrow (IXR) + [M]_{8-15}$$

Type: 4 (PCR) $\leftarrow (PCR) + 1 + \{(IXR) \geq 0\}$

The algebraic sum of the contents of the index register plus the unsigned literal, bits 8-15 of the instruction word, replace the contents of the index register. If the sum is greater than or equal to zero the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed.

Example:

0100 0404 IXS 04

	(PCR)	(IXR)
Initial	0100	FFFE
Final	0102	0002

DXS Decrement Index and Skip

T: $1 + \{(IXR) < 0\}$

Equations:

$$(IXR) \leftarrow (IXR) - [M]_{8-15}$$

$$(PCR) \leftarrow (PCR) + 1 + \{(IXR) < 0\}$$

Type: 5

The algebraic difference between the contents of the index register and the unsigned literal, bits 8-15 of the instruction word replace the contents of the index register. If the contents of the index register are less than zero the next instruction in sequence is skipped; otherwise, the next instruction in sequence is executed.

Example:

0100 0504 DXS 04

	(PCR)	(IXR)
Initial	0100	0004
Final	0101	0000

LLB Load Literal Byte

T: 1

$$\text{Equation: } (ACR)_{8-15} \leftarrow [M]_{8-15}$$

Type: 6

The contents of bits 8-15 of the accumulator are replaced by the literal, bits 8-15 of the instruction word. The content of bits 0-7 of the accumulator are unaffected.

Example:

06FC LLB - 4

	(ACR)
Initial	2300
Final	23FC

CLB Compare Literal Byte

T: 1

Equations:

$$(ADFNEG) \leftarrow \{(ACR)_{8-15} - [M]_{8-15} < 0\}$$

$$(ADFEQL) \leftarrow \{(ACR)_{8-15} - [M]_{8-15} = 0\}$$

Type: 7

The contents of bits 8-15 of the accumulator are compared to the literal, bits 8-15 of the instruction word, and the result stored in the comparison register specifying whether the contents of bits 8-15 of the accumulator were less than, equal to, or greater than the literal. Both the literal and the contents of bits 8-15 of the accumulator are treated as 8-bit two's complement numbers for the purpose of this comparison.

Example:

070C CLB X 'C'

	(ACR)	(ADFEQL)	(ADFNEG)
Initial	230C	0	0
Final	230C	1	0

LPL Load Protect Lower
(Memory Protect Option; See Section 6-7)

T: 1

Equations: $(MPRL)_{1-8} \leftarrow [M]_{8-15}$
 $(MPRL)_{9-15} \leftarrow 0$

Type: D

The contents of bits 8-15 of the instruction word are loaded into bits 1-8 of the Memory Protect Lower Register. Bits 9-15 are forced to zero.

Example:

0D40 LPL X '40'

	(MPRL)
Initial	0000
Final	2000

LPU Load Protect Upper
(Memory Protect Option: See Section 6-7)

T: 1

Equations: $(MPRU)_{1-8} \leftarrow [M]_{8-15}$
 $(MPRU)_{9-15} \leftarrow 1$

Type: E

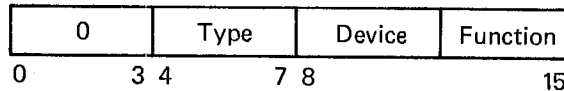
The contents of bits 8-15 of the instruction word are loaded into bits 1-8 of the Memory Protect Upper Register. Bits 9-15 are forced to all ones.

Example:

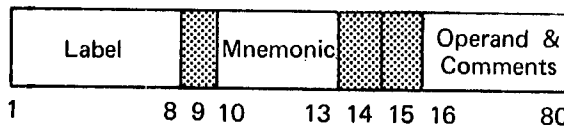
0E40 LPU X '40'

	(MPRU)
Initial	007F
Final	207F

Object Format:



Source Format:



DIN Direct Input

T: 2

Equation: $(ACR)_{0-15} \leftarrow (DIN)_{0-15}$

Type: 2

Device and Function:

(See Section 5 or Appendix L.)

Bits 8-15 of the instruction are transferred to the DIO address bus and an input strobe is generated. At the trailing edge of the input strobe, the contents of the accumulator are replaced by the data applied to the DIO input data bus. Bits 8-11 of the instruction word designate the device selected for input and bits 12-15 of the instruction designate the selected function.

Example:

02ED DIN X'E', X'D'

	(Address Bus)	DIO Input Bus	(ACR)
Initial	00	0002	FFFF
Final	ED	0000	0002

2-1.9 Class 8 - Input/Output (See Section 4)

Class 8 instructions communicate with the peripheral equipment. Two operands, 4 bits each, are required to specify the device and function, respectively. The two operands must be separated by a comma.

DOT Direct Output

T: 2

Equation $(DOT)_{0-15} \leftarrow (ACR)_{0-15}$

Type: 3

Device and Function: (See Section 5 or Appendix L.)

Bits 8-15 of the instruction are transferred to the DIO address bus, the contents of the accumulator are transferred to the DIO output data bus, and an output strobe is generated. The contents of the accumulator are not affected. Bits 8-11 of the instruction word designate the device selected for putput and bits 12-15 of the instruction word designate the selected function.

Example:

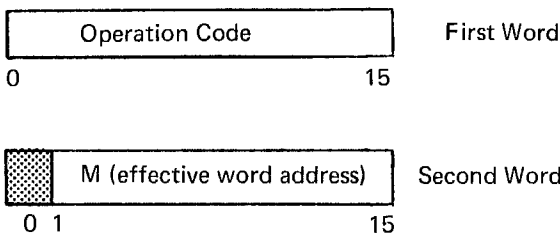
0300 DOT X 'E', X 'F'

	Address Bus	(ACR)	DIO Out Bus
Initial	00	0020	0000
Final	E \bar{F}	0020	0020

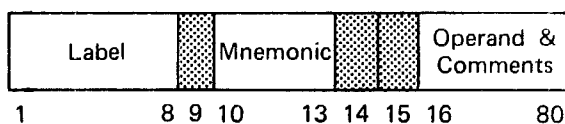
2-1.10 Class 9 - Optional Hardware Double Word Instructions

Class 9 instructions are these that require two words for their operation while using the accumulator and index register as a double register. The first word of the instruction contains the operation code and the second word contains the effective address, M.

Object Format:



Source Format:



MPY Multiply (See Section 6-4)

T: $6 \frac{1}{5} + \frac{N_{1-15}}{5} + \frac{2N_0}{5}$ (Round upward to whole in multiplier where N is the number of "ones" in the multiplier)

Equations:

$$\begin{aligned} (IXR, ACR)_{1-15, 1-15} &\leftarrow (ACR)_{1-15} \times (M)_{1-15} \\ (ACR)_0 &\leftarrow (IXR)_0 \oplus \{(ACR)_0 \oplus (M)_0 = 1\} \\ (ADFOVF) &\leftarrow \{(ACR)_{0-15} = 8000\} \vee \{(M)_{0-15} = 8000\} \end{aligned}$$

OP Code: OBOF

The multiply instruction utilizes a 16-bit multiplier which resides in the accumulator and a 16-bit multiplicand which resides in the memory location specified by the instruction. All negative numbers are expressed in 2's complement form.

Execution of the multiply instruction produces a 31-bit algebraic product in the index register and the accumulator. The most significant 16 bits of the product reside in the index register. The least significant 15 bits of the product reside in bits 1-15 of the accumulator. The most significant bit (sign bit) of the accumulator is set to the most significant bit of the index register. The product is expressed in 2's complement form. The overflow flip flop will be set if both the multiplier and multiplicand equal 8000₁₆, the most negative member.

Example:

0100 OBOF MPY X '200'
0101 0200

	(PCR)	(IXR)	(ACR)	(200)
Initial	0100	0000	4000	0040
Final	0102	0020	0000	0040

DIV Divide (See Section 6-4)

T: $10 + \{(IXR)_0 \wedge ((M)_0)\}$
Equations:

$$\begin{aligned} (ACR)_{1-15} &\leftarrow (IXR, ACR)_{1-15, 1-15} \div (M)_{1-15} \\ (IXR)_{1-15} &\leftarrow \text{Remainder of} \\ &\quad [(IXR, ACR) \div (M)_{1-15}]_{1-15, 1-15} \\ (IXR)_0 &\leftarrow (IXR)_0 \\ (ACR)_0 &\leftarrow \{(IXR)_0 \oplus (M)_0 = 1\} \end{aligned}$$

Special Conditions: Overflow

T: 4

Equations:

$$(ADFOVF) \leftarrow \{I(M)_{0-15} \leq I(IXR)_{0-15}\}$$

$$(IXR, ACR)_{0-15, 0-15} \leftarrow (IXR, ACR)_{0-15, 0-15}$$

$$(ACR)_0 \leftarrow (IXR)_0$$

OP Code: 0C0F

The divide instruction utilizes a 31-bit dividend which resides in the accumulator and the index register. The most significant 16 bits of the dividend reside in the index register and the least significant 15 bits are located in bits 1-15 of the accumulator. The most significant bit of the accumulator has no effect on the divide instruction. The 16-bit divisor resides in the memory location specified by the instructions. All negative numbers are expressed in 1's complement form.

Execution of the divide instruction produces a 16-bit algebraic quotient in the accumulator and a 16-bit remainder in the index register. The sign of the remainder is the same as that of the dividend. The sign of the quotient is in bit 0 of the accumulator. Both the quotient and the remainder are in two's complement form.

The execution time of the divide instruction is 10 cycles except when the dividend is negative and the divisor is positive, it is then 11 cycles.

Care must be exercised in the set-up of the divide instruction to avoid an overflow condition. An overflow condition exists when the absolute value of the divisor is equal to or less than the absolute value of the most significant half (Index Register) of the dividend. If this condition exists, the overflow indicator will be turned on and the dividend contained in the index register and accumulator will remain unchanged; except bit 0 of the index register is copied into bit 0 of the accumulator. The execution time of the divide instruction for overflow is 4 cycles.

Example:

0100 0C0F DIV X '200'

0101 0200

	(PCR)	(IXR)	(ACR)	(200)
Initial	0100	0001	0000	0002
Final	0102	0000	4000	0002

2-1.11 Class 10- Set Memory Bank

The SMB, set memory bank instruction constructs a 15-bit word address from the expression in the operand field. This expression may consist of any legal constant or variable with the exception of a byte address, or an external if it is an absolute program.

From the constructed 15-bit word address the five most significant bits are copied into the extension register. This will set the extension register to point to the byte page in which the created 15-bit address resides.

Object Format:

0	0	100	Effective Byte Page No.
0	3 4	7 8	10 11
			15

Source Format:

Label	SMB	Address & Comments
1	8 9 10	13 14 15 16
		80

SMB Set Memory Bank*
T: 1

Equations:

$$(EXR)_{0-4} \leftarrow \text{Effective Byte Page Number}$$

Type: 8 or 9

The contents of bits 0 to 4 of the extension register are replaced by bits 12-15 of the instruction.

See Section 1-4.4.1.

*Set Memory Bank is assembled as either a Set Memory Lower or Set Memory Upper instruction.

Example:
0080 SMB BILL

	(EXR)
Initial	05
Final	00

NOTE: The expression BILL represents an address that resides in byte page 0 in this example.

2-2 PSEUDO INSTRUCTIONS

2-2.1 General

Pseudo instructions (or operations) are assembly directives that allow the user to specify program parameters. The directives provided are classified as 1) Symbol and Data Definitions and 2) Assembler Control. Types of addressing, data, symbols, memory usage, and subroutine names may be defined.

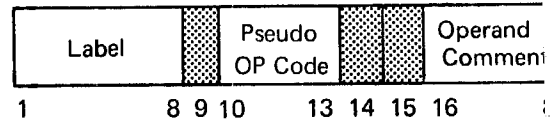
Not all pseudo instructions are compatible with every 706 assembler. Therefore, the usage of each directive will be indicated by assembler title (e.g., SYM I, SYM II).

2-2.2 Symbol Data and Definition

The directives in this category provide easy definitions of the following types of constants:

1. Data for one computer word.
2. Data for one computer byte.
3. Alphanumeric data, two characters per word, called a text string.
4. Two word (double precision) integer.
5. Multiword floating point real numbers.
6. Symbol definitions may be accomplished by equating symbols to expressions.
7. Reserving core storage areas.

Source Format:



DATA Data

Usage: SYM I, SYM II

The DATA directive defines data to fill one (1) word. Its operand may be an expression. To specify data for more than one word, separate each expression with delimiting commas. The pseudo op D may be used as an abbreviation of DATA. Repeated commas indicate operands with the value zero.

Examples:

Label	Operation	Operand
SOCK	DATA	R, S, /R, /S
TAP	D	7 + ZERO, , 0
ELE	DATA	A, 'BC', X'ABCD'

BYTE Byte

Usage: SYM I, SYM II

SYM I:

The BYTE directive defines data to fill two bytes. The two operands are delimited by comma. If more than two operands are given, only the first two are used.

SYM II:

The BYTE directive defines data to fill n bytes—two bytes per word. The operands are delimited by a comma. If n is an odd number of bytes, one more byte will be added by the assembler to make the string end on a word boundary. The additional byte will be zero (0).

Examples:

Label	Operation	Operand
SQRT	BYTE	1, 'E'
	BYTE	X'BB', JOE, SAM, 0

ignored fo.
SYM 1

XT Text

Usage: SYM II

The TEXT directive defines alphanumeric data to fill as many words as necessary to accommodate the given data, two characters per word. The text string is initiated and terminated with an apostrophe. If an apostrophe is desired as part of the text string, it must be indicated by two successive apostrophes. Only one apostrophe will be inserted into the data. If the number of text bytes is odd, an extra byte will be added by the assembler to make the string end on a word boundary. The additional byte will be a blank.

Example:

Label	Operation	Operand
KARL	TEXT	'ABCDEF12F3F5'
DAVE	TEXT	'ISN'T IT GREAT'
	TEXT	'A'

DPI Double Precision Decimal Integer

Usage: SYM II

The DPI directive is used to define a double precision decimal integer data type. More than one double precision integer may be specified by separating each integer definition with delimiting commas. Repeated commas indicate operands with the value zero. The defined integer is assembled into the next two words of the programs. These words have the following format:

Word 1	Sign	Most significant half of decimal integer.
Word 2	0	Least significant half of decimal integer.

0 1 15

The sign bit (bit 0) of word one signifies the algebraic sign of the decimal integer. Bit 0 of word 2 is always 0.

Label	Operation	Operand
ME	DPI	-123456789
	DPI	999999999
	DPI	2, -8878876,,9

REAL Two Word Floating Point Number

Usage: SYM II

The REAL directive defines a single precision floating point real number. More than one single precision real number may be specified by separating the number definitions with delimiting commas. Repeated commas indicate an operand with a value zero. The defined real number is assembled into next two words of the program. These words have the following format:

Word 1	least significant part of mantissa	exponent biased by X'80'
Word 2	sign	most significant part of mantissa

0 7 8 15

The sign bit (bit 0) of word 2 indicates the algebraic sign of the mantissa of the single precision real number. This data type is used only in FORTRAN IV.

Example:

Label	Operation	Operand
MY	REAL	-78546.23E+20,,27.4E2
	REAL	+2487.333E+2

EPRL Mid Precision Real Number

Usage: SYM II

The EPRL directive defines a mid-precision floating point real number. More than one mid precision real number may be specified by separating the number definitions with delimiting commas. Repeated commas indicate an operand with value zero. The defined real number is assembled into the next three words of the program. These words have the following format:

Word 1	0	exponent biased by X'80'
Word 2	sign	most significant part of mantissa
Word 3	0	least significant part of mantissa
	0	7 8 15

The sign bit (bit 0) of word 2 indicates the algebraic sign of the extended precision real number. Bit 0 of word 3 is always zero. This data type is used in FORTRAN, FORTRAN IV, and by the math package.

Example:

Label	Operation	Operand
SELF	EPRL	+2,776432.389E20
	EPRL	-123456.789E-9

DPRL Double Precision Real Number
Usage: SYM II

The DPRL directive defines a double precision floating point real number. More than one double precision real number may be specified by separating the number definitions with delimiting commas. Repeated commas indicate an operand with value zero. The defined real number is assembled into the next four words of the program. These words have the following format:

Word 1	0	exponent biased by X'80'
Word 2	sign	most significant part of mantissa
Word 3	0	mid significant part of mantissa
Word 4	0	least significant part of mantissa
	0	7 8 15

The sign bit (bit 0) of word 2 indicates the algebraic sign of the double precision real number. Bit 0 of words 3 and 4 is always zero.

This data type is used only in FORTRAN IV.

Example:

Label	Operation	Operand
1	DPRL	-123456789.9873E-1
	DPRL	-2,999.9999876E12

EQU Equate
Usage: SYM I, SYM II

The EQU directive equates a symbol to an expression. The expression must be self-defining or completely defined by the preceding source statements (i.e., not forward defined).

Examples:

Label	Operation	Operand
JOE	EQU	X'3A' SELF-DEFINED
A	EQU	\$+JOE PREDEFINED
KARL	EQU	'NO' SELF-DEFINED
HERE	EQU	\$ SELF-DEFINED

IS Is
Usage: SYM II

The IS directive equates a symbol to an expression. The only difference between the IS directive and the EQU directive is that if a label appears in the label field of more than one IS directive its value will be redefined each time rather than being treated as multiply defined.

Examples:

JOE	IS	2
JOE	IS	X'4'
JOE	IS	JOE + 3

If a label appears in more than one IS directive its value at any time during the assembly is determined by the last IS directive assembled up to that point.

ES Reserve
Usage: SYM I, SYM II

The RES directive reserves a block of words whose size is determined by the value in the operand field. The value must be a constant or a previously defined name.

Examples:

Label	Operation	Operand
VIC	RES	10
TOBY	RES	A
TOP	RES	2048-\$

2.3 Assembler Control Directives

The directives in this category control the assembler's location counter, conditional processing of statements, external definitions and loader directives. The label field of a control directive is not used.

ORIG Origin
Usage: SYM I, SYM II

SYM I:

The ORIG directive sets the location counter to the specified value of the operand field in absolute assemblies. Object programs produced from such an assembly are loadable by the bootstrap loader or the XRAY executive. ORIG must precede source statements that cause object code to be produced by the assembler. An assembly must reside in one 2048 word page.

SYM II

The ORIG directive sets the location counter to the specified value of the operand field in absolute assemblies. Object programs produced from such an assembly are loadable by the bootstrap loader. Only one ORIG statement is allowed in an assembly; however, it may be placed anywhere in the program to fix the origin of the next cell.

Examples:

Label	Operation	Operand
	ORIG	300
	ORIG	X'7AB'

END End
Usage: SYM I, SYM II

END causes the assembler to terminate processing of input statements. It also allows the option of specifying a transfer address (a name or constant) in its operand field on relocatable programs.

Examples:

Label	Operation	Operand
	END	
	END	START

LOAD Load
Usage: SYM I, SYM II

LOAD directs the system loader to load unconditionally the specified routines along with the program at object time.

Examples:

Label	Operation	Operand
	LOAD	DUMP, DEBUG

NTRY Entry
Usage: SYM I, SYM II

An NTRY statement identifies an entry point to the program. The label is defined to the loader which links it to routines in which the same label was undefined and external. The NTRY directive must not precede the definition of the entry point.

Examples: (Where ENT1, ENT2, and HERE have been defined).

Label	Operation	Operand
	NTRY	ENT1, ENT2
	NTRY	HERE

LIBR Library
Usage: SYM I, SYM II

The LIBR directive is used to generate identification labels for routines being added to the system library. The LIBR directive must precede all other source statements. Names in the LIBR list must also appear in an NTRY list.

Examples:

Label	Operation	Operand
	LIBR	SIN, COS
	LIBR	DOIO, STAT

DO

Do
Usage: SYM II

The DO directive provides for repetitive code generation based upon the value of the DO variable, a question mark (?). The "range" of the DO directive is one statement. When used with the PROC capability, however, the range is extended to multiple statements. The format of the DO directive is:

Label	Operation	Operand
	DO	E ₁ , E ₂ , E ₃

Where E₁ is the initial value of the DO variable, E₂ is the terminal value, and E₃ is the increment for each repetition.

The operands may be expressions. However, they must always be positive values. If E₁ ≤ E₂, the next statement is assembled with ? = E₁. The statement will be repeated adding E₃ to the value of ? each time. The repetition will terminate when a value of ? is encountered which is larger than E₂ (that value will not be included). The repeated statement must not have a label.

Example: Generate a table of numbers from -100 to -200 in increments of 5.

Label	Operation	Operand	Comment
TABLE	RES	0	LABEL
	DO	100,200,5	
	DATA	?	

TRUE Conditional Statement Processing
FALS Usage: SYM I, SYM II
ENDC

The TRUE or FALS directives permit the programmer to specify the conditional processing of statements by the assembler. The effect of the TRUE or FALS is terminated by the appearance of the ENDC directive (end of conditional processing). The label field of the conditional directive is unused.

The operand field of TRUE or FALS is composed of two expressions separated by a relational operator. Relational operators equal (=), greater than (>) and less than (<) are allowed.

Examples:

Label	Operation	Operand
	TRUE	X = Y + 3
W	DATA	1
	ENDC	
	FALS	W = 1
	TRUE	Y < W + 2
Z	DATA	A
	ENDC	
B	DATA	Z = W + 1
	ENDC	

In the first example, if X is equal to Y plus 3, the statement before the ENDC will be assembled. If the condition is not true, the DATA statement will not be processed by the assembler.

In the second example both conditions must be met, i.e., W ≠ 1 and Y < W + 2, for the statement labeled Z to be processed. The statement labeled B will be processed, if one condition is satisfied, i.e., W ≠ 1.

PROC Procedure Control
ENDP Usage: SYM II

The user is provided with the capability to create procedures through the use of the PROC and ENDP directives. The procedure capability permits the creation of operation codes to represent frequently employed instruction sequences. By using procedures the user can reduce both the effort and time involved in developing debugged programs.

To use a procedure the user defines an instruction sequence in terms of the dummy operands P(1), . . . , P(N).

The assembler retains this *procedure definition* in memory. When the name of the procedure is encountered in the operation code field of a statement, the instruction sequence from the definition is assembled with the dummy operands replaced by the values of the arguments in the *reference line*. The procedure definition must precede its reference lines.

The name of the procedure is defined in the label field of the PROC statement. Only the first four characters are used as the procedure name. Labels should not be used in procedure definitions with the exception of the IS directive. A definition is terminated by the ENDP statement.

The dummy parameter P(0) has a special meaning when used in a procedure definition. P(0) is assigned a value equal to the number of parameters in the procedure reference line. This feature allows conditional processing of the procedure text, based on the number of parameters given in the reference line. A reference line to a previously defined procedure may appear in a procedure definition (i.e., procedures may be nested to any level).

Example:

Label	Operation	Operand
SUM	PROC	
	LDW	P(1)
	ADD	P(2)
	STW	P(3)
	ENDP	

PROCEDURE REFERENCE

If the procedure reference line is:

SUM A1, A2, . . . , AN

then the correspondence between the dummy operands P(n)'s and the reference line argument An's is the following:

P(1) has the value A1
P(2) has the value A2

.

.

.

P(N) has the value AN

If fewer reference line arguments are given than are required in the definition, the absent parameters are assigned the value of zero.

Example:

Reference line for the previously defined SUM procedure.

Operation	Operand
SUM	A, B, TOTAL

The equivalent symbolic code would be:

Operation	Operand
LDW	A
ADD	B
STW	TOTAL

SUBR EXIT

Usage: SYM II

Two pseudo operations are convenient for subroutine entry and exit. If the subroutine has been reached via the JSX calling sequence, the index register contains the address of the cell following JSX. To preserve the return address SUBR pseudo-op is used.

Label	Mnemonic	Operand
JOE	SUBR	
	.	
	.	
	.	
	EXIT	JOE, n

Where JOE is the entry point and n is the number of arguments in the calling sequence. The equivalent code is:

Label	Mnemonic	Operand
	DATA	0
JOE	STX	\$-1
	.	
	.	
	.	
	LDX	JOE-1
	JSX	* n

If n = 0, the return can be given as:

Mnemonic	Operand
EXIT	JOE

Section 3 INTERRUPT

3-1 GENERAL

The priority interrupt system of the Raytheon 706 permits rapid response by the computer to events occurring external to the central processing unit (CPU). One level of interrupt is included with the basic system; however, expansion up to 16 levels is optionally available. Each level in the interrupt system may assume one of four states.

Disabled: The interrupt level is unable to respond to an interrupt. An interrupt signal sent from an external device is ignored.

Idle: The interrupt level is able to respond to an interrupt signal, but none has been received. The interrupt signal must have a pulse width greater than 900 nanoseconds.

Wait: An interrupt signal has been received, accepted, and the level is awaiting processing.

Active: The CPU is processing the interrupt by execution of a fixed hardware instruction sequence. The interrupt remains in the active state until the program executes either an Interrupt Return or a Disable Interrupt instruction.

Interrupt levels are numbered from 0 to 15. The lowest enabled interrupt level has the lowest priority. Level 0 is lowest. Level 15 is highest. Each interrupt level is allocated three unique words in the lower address portion of memory for storage of the program counter, an interrupt linkage address and machine status. Memory interrupt locations are assigned as follows:

0000	Interrupt Level 0	PCR Save
0001		Linkage Address
0002		Machine Status
0003		Unused

0004	Interrupt Level 1	PCR Save
0005		Linkage Address
0006		Machine Status
0007		Unused
.		.
.		.
003C	Interrupt Level 15	PCR Save
003D		Linkage Address
003E		Machine Status
003F		Unused

3-2 PROGRAMMING WITH THE INTERRUPT SYSTEM

The Raytheon 706 has five instructions which allow the user to achieve full utilization of the priority interrupt system. These instructions are Mask Interrupts (MSK), Unmask Interrupts (UNM), Enable Interrupt (ENB), Disable Interrupt (DSB), and Interrupt Return (INR).

To allow an interrupt subroutine to complete at least a few essential operations before it is itself interrupted, the Mask Interrupts (MSK) instruction may be employed. Execution of the MSK instruction inhibits all levels from causing interruptions to the current program. However, the interrupt conditions remain pending and will be serviced by the 706 CPU when the interrupt inhibit mask is removed. The interrupt inhibit mask is removed by the execution of the Unmask Interrupts (UNM) instruction.

Initialization of the Raytheon 706 (power ON or RESET) sets all interrupt levels to the Disabled state and removes the interrupt inhibit mask. Each interrupt level may advance from the Disabled state to the Idle state by execution of an Enable Interrupt (ENB) instruction which specifies the particular level.

The Disable Interrupt (DSB) instruction changes the referenced interrupt level from its present state (Idle, Wait, or Active) to the Disabled state.

When the interrupt subroutine completes its operation, the interrupt level can be returned to the Idle state and control returned to the interrupted program by execution of the Interrupt Return (INR) instruction. The INR instruction specifies the interrupt level being returned to the Idle state and restores the program counter and machine status to what they were at time of the interrupt.

3-3 OPERATION OF THE INTERRUPT SYSTEM

The operation of an interrupt level is shown in the flow chart, Figure 3-1. When power is turned on, or when the RESET switch on the control panel is activated, all interrupt levels are set to the Disabled state and the inhibit interrupt mask is set off. Interrupt levels, on an individual basis, may be transferred from the Disabled state to the Idle state by the ENB instruction.

When an interrupt level is in the Idle state, any interrupt signal to that level with duration greater than 900 nanoseconds causes an advance to the Wait state. An interrupt level in the Wait state is advanced to the Active state when there is no higher priority interrupt level in the Active or the Wait state, the inhibit interrupt mask is off, and the execution of the current instruction is completed by the 706 CPU.

When an interrupt level advances to the Active state a fixed hardware sequence stores the contents of the program counter, stores the machine status, places the central processor in global mode, and transfers to the interrupt linkage address. Machine status consists of the contents of the extension register, the overflow indicator, the comparison indicators, and the memory addressing mode (local/global) at the time of interrupt. An interrupt level remains in the Active state until the interrupt subroutine is completed by execution of the INR instruction. The INR instruction returns the interrupt to the Idle state and restores the program counter and the machine status to their previous condition at time of interrupt.

An interrupt level in the Active state does not necessarily imply that the 706 CPU is still processing this particular level of interrupt. The 706 CPU can be under control of a higher priority interrupt subroutine. Priority control allows the highest level in the Wait or Active state to postpone lower priority interrupts that are pending. For example, if interrupt level 7 is in the Active state, interrupt levels 0 to 6 that are in the Wait state will not be serviced by the 706 CPU. The pending lower priority interrupts will not be serviced until interrupt level 7 is changed to the Idle or the Disabled state by either the INR or the DSB instruction. However, if any interrupt level from 8 to 15 should change to Wait state while interrupt level 7 is in the Active state, an interrupt would occur and control transferred to the higher priority subroutine.

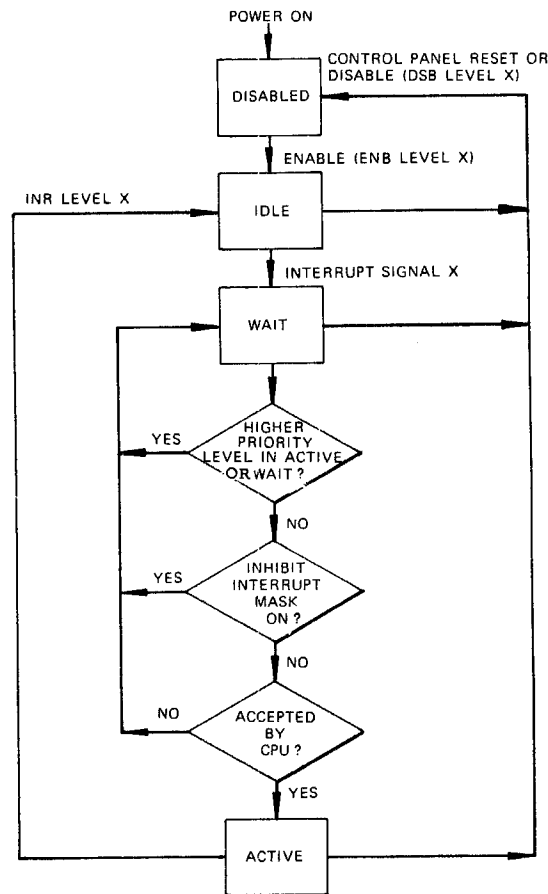


FIGURE 3-1.
INTERRUPT OPERATION FOR LEVEL X
(X = 0, 1, 2, 15)

3-4 INTERRUPT RANKING AND ASSIGNMENT

Table 3-1 is a list of 706 options and peripheral devices arranged in descending order of relative priority for interrupt assignment. In any system consisting of a set of peripheral devices and options selected from the list, the interrupt level for each may be assigned according to this ranking.

When a system does not have sufficient number of interrupt levels to permit each option or peripheral device to have unique interrupts, then the available interrupt levels will be assigned according to the following rules:

1. The higher interrupt levels are first assigned to those devices or options which must have unique separate interrupts. Level assignment within this group is according to priority ranking.

2. The lower interrupt levels will be shared by the remaining devices.

A maximum of sixteen interrupt lines are provided in the DIO channel. Each line may be used individually by a peripheral device or may be shared by various devices. An interrupt line is used individually when it is desired to minimize service response time for a given peripheral device. An interrupt on a shared interrupt line requires the I/O service routine to individually interrogate all active peripheral devices by means of DIN instructions for status since more than one device may have caused the interrupt.

Table 3-1 Interrupt Ranking and Assignment

Peripheral Option	Interrupt Levels Assignable
Power Fail Safe	1-15*
Memory Protect	1-15*
Memory Parity	1-15*
Interval Timers	1-15
DIO Magnetic Tape	1-15
Card Reader	1-15
Card Punch	1-15
DMA Magnetic Tape – Data Chaining	1-15
DMA Magnetic Tape	1-15
Disc	1-15
Line Printer	1-15
Digital Plotter	1-15
High Speed Paper Tape Reader	} Shares One interrupt Level
High Speed Paper Tape Punch	
Teletype	0
User Decision Required	
A/D Converter	0-15
Buffered Input Channel(s)	0-15
Buffered Output Channel(s)	0-15
Time of Day Clock (Normally does not use an interrupt)	
Teletype Multiplexer	0-15

*Interrupt operation mandatory

NOTE:

Any peripheral may actually be assigned to any interrupt level, but, the Raytheon 706 software only supports the teletype and high speed paper tape on level 0. Any other device may be assigned to a level other than zero.

SECTION 4

INPUT/OUTPUT

4 - 1 GENERAL

The Raytheon 706 has two I/O communication channels, the Direct Input/Output (DIO) channel and the Direct Memory Access (DMA) channel (Figure 2 - 1).

The DIO channel transfers data between a peripheral device and the 706 CPU accumulator through direct program control. Instructions within the CPU initiate the transfer of data between the peripheral device and the accumulator, and between the accumulator and memory. The DIO channel can service data transfer rates up to 278 kilowords per second and is especially suited to peripheral controllers of slow and medium speed devices such as teletypes, card readers, paper tape readers and punches. An interrupt system provided with the DIO channel assumes that the CPU responds rapidly to word input/output requests.

The DMA channel allows peripheral devices to by-pass the CPU by transferring data *directly* between themselves and memory. Instructions within the CPU initialize DMA data transfers. The DMA channel can service data transfer rates up to 1.11 megawords per second and is especially suited to peripheral controllers of high speed devices such as disc memory and magnetic tape. An interrupt system provided with the DMA channel assures rapid response by the CPU to block input/output requests.

4 - 2 DIRECT INPUT/OUTPUT (DIO) CHANNEL

4 - 2.1 General Description

The DIO channel exchanges data between external devices and the 706 CPU accumulator. The channel consists of an 8-bit address bus, a 16-bit input bus, a 16-bit output bus, two strobe lines, timing pulses, up to 16 interrupt signals, an external sense and system reset line, and various terminator voltages. These buses and signals are

shown in Figure 4 - 1 and described in Table 4 - 1.

4 - 2.2 Channel Operation

The DIO channel is controlled by two instructions, Direct Input (DIN) and Direct Output (DOT). The DIN instruction reads data from an input device (e.g., card reader) and reports status from both input and output devices. The DOT instruction starts peripheral motion for both input and output devices, writes data on an output device (e.g., card punch) and terminates peripheral operations.

4 - 2.2.1 DIN Instruction

The DIN instruction places bits 8-15 of the instruction word on the 8-bit address bus and generates an input strobe (DISB) signal (Figure 4 - 2). The 8-bit address bus selects one of sixteen possible device controllers on the DIO channel and designates one of sixteen possible functions of each device controller. The selected device then places data on the 16-bit input bus. This data is transferred to the accumulator at the trailing edge of the input strobe signal (Figure 4 - 3). Each word transferred requires the execution of a DIN instruction.

4 - 2.2.2 DOT Instruction

The DOT instruction places bits 8-15 of the instruction word on the 8-bit address bus and generates an output strobe (DOSB) signal (Figure 4 - 4). The 8-bit address bus selects one of sixteen possible device controllers on the DIO channel and designates one of sixteen possible functions of each device controller. Output data must be loaded into the accumulator before the DOT instruction is executed so that data is present on the output bus when the output strobe is generated (Figure 4 - 3). Each word transferred requires the execution of a DOT instruction.

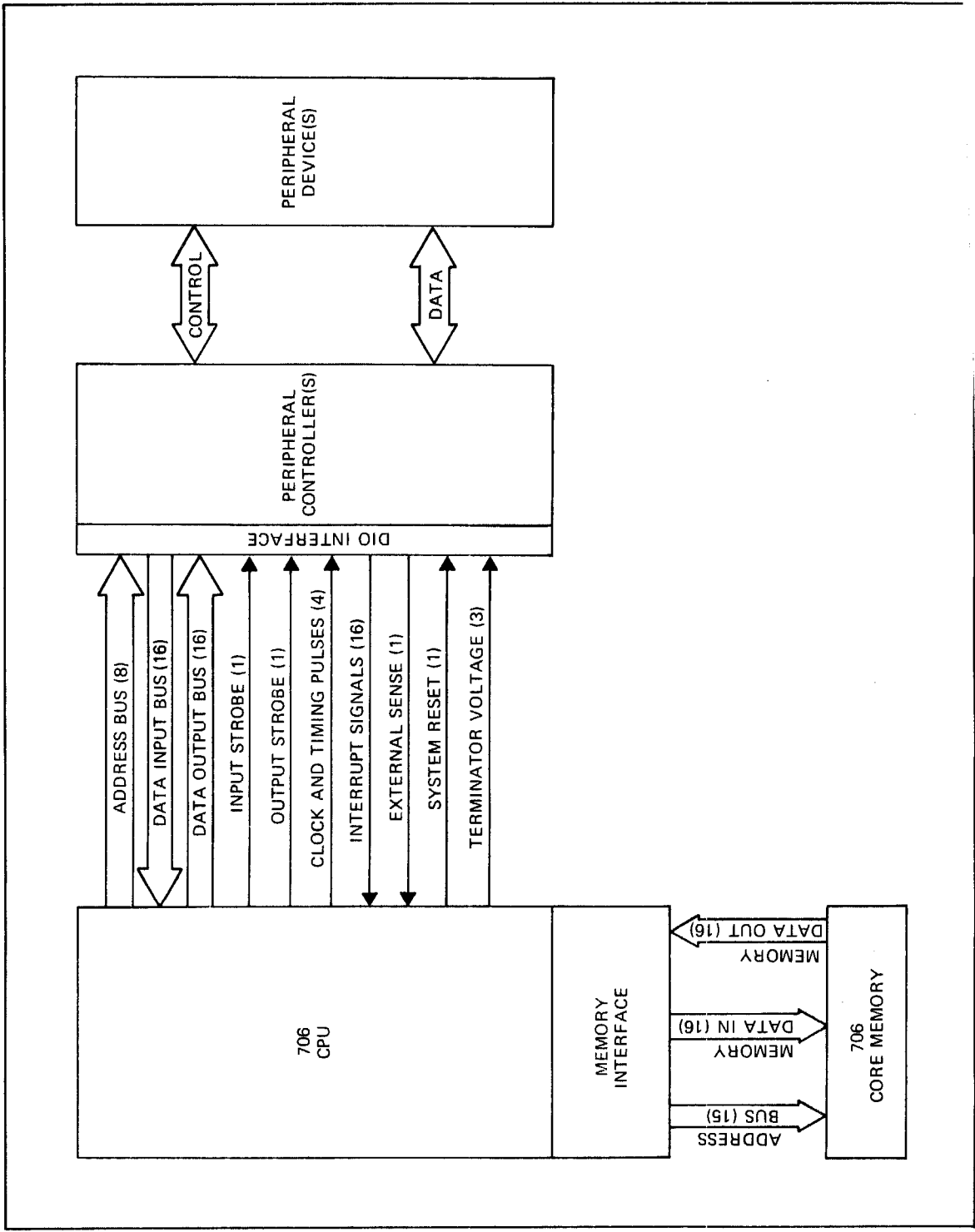


Figure 4 - 1. Direct Input/Output System

Table 4 - 1. Direct Input/Output Channel Signals

SIGNAL (No. of Lines)	PURPOSE
<p>Address Bus (8) DAD08–DAD15</p>	<p>An 8-bit address bus that transmits selection information to the input/output system. This information is contained in the input/output instruction being executed and is transmitted from the memory buffer register. DAD08–DAD11 defines the device being selected while DAD12–DAD15 defines the function that the selected peripheral device is to perform.</p>
<p>Data Input Bus (16) DIN00–DIN15</p>	<p>A 16-bit input bus that transmits data from the selected peripheral device to the accumulator. It is not necessary for a device to use all of the 16 lines. For example, the teletype controller uses only the lower 8 positions (DIN08–DIN15) of the bus.</p>
<p>Data Output Bus (16) DOT00–DOT15</p>	<p>A 16-bit output bus that transmits data from the accumulator to a selected peripheral device.</p>
<p>Input Strobe (1) DISB</p>	<p>A single line that notifies all the peripheral controllers that there is an address on the DIO Address Bus and that an input (to the computer) function is being requested.</p>
<p>Output Strobe (1) DOSB</p>	<p>A single line that notifies all the peripheral controllers that there is an address in the DIO Address Bus and that an output is being sent from the computer.</p>
<p>Clock and Timing Pulses (4) KEXT, MT0, MT2, MT4</p>	<p>These four lines enable the peripheral devices to synchronize with the memory. MT0, MT2, and MT4 are pulses with a period of 900 nsec. and a width of 180 nsec. KEXT is a pulse with a period of 180 nsec. and a width of 50 nsec.</p>
<p>Interrupt Signals (16) IRPT00–IRPT15</p>	<p>A maximum of 16 interrupt signals are optionally available within the DIO channel. Each signal corresponds to one level of interrupt that alerts the 706 CPU of events external to the CPU. The interrupt signals have a minimum duration of 900 nsec.</p>
<p>External Sense (1) EXSENS</p>	<p>The external sense line informs the CPU when a peripheral device reaches a given state. The CPU may then test this line by using the Skip on Sense External (SSE) instruction. The inclusion of the external sense line in the peripheral is normally specified by the user.</p>

Table 4 - 1. Direct Input/Output Channel Signals (Cont)

SIGNAL (No. of Lines)	PURPOSE
System Reset (1) REXT	The system is reset when the RESET pushbutton (Figure 7 - 1) on the CPU control panel is pressed, or when an initial power-on sequence is performed. The reset line initializes the peripheral device controller, i.e., reset registers and halt the device operation. When a reset occurs, the CPU is placed in the HALT state, all pending interrupts are eliminated, and all the CPU registers are reset.
Terminator Voltage (3) V + 3.0	This voltage is required for the resistor networks used to terminate the DIO lines.

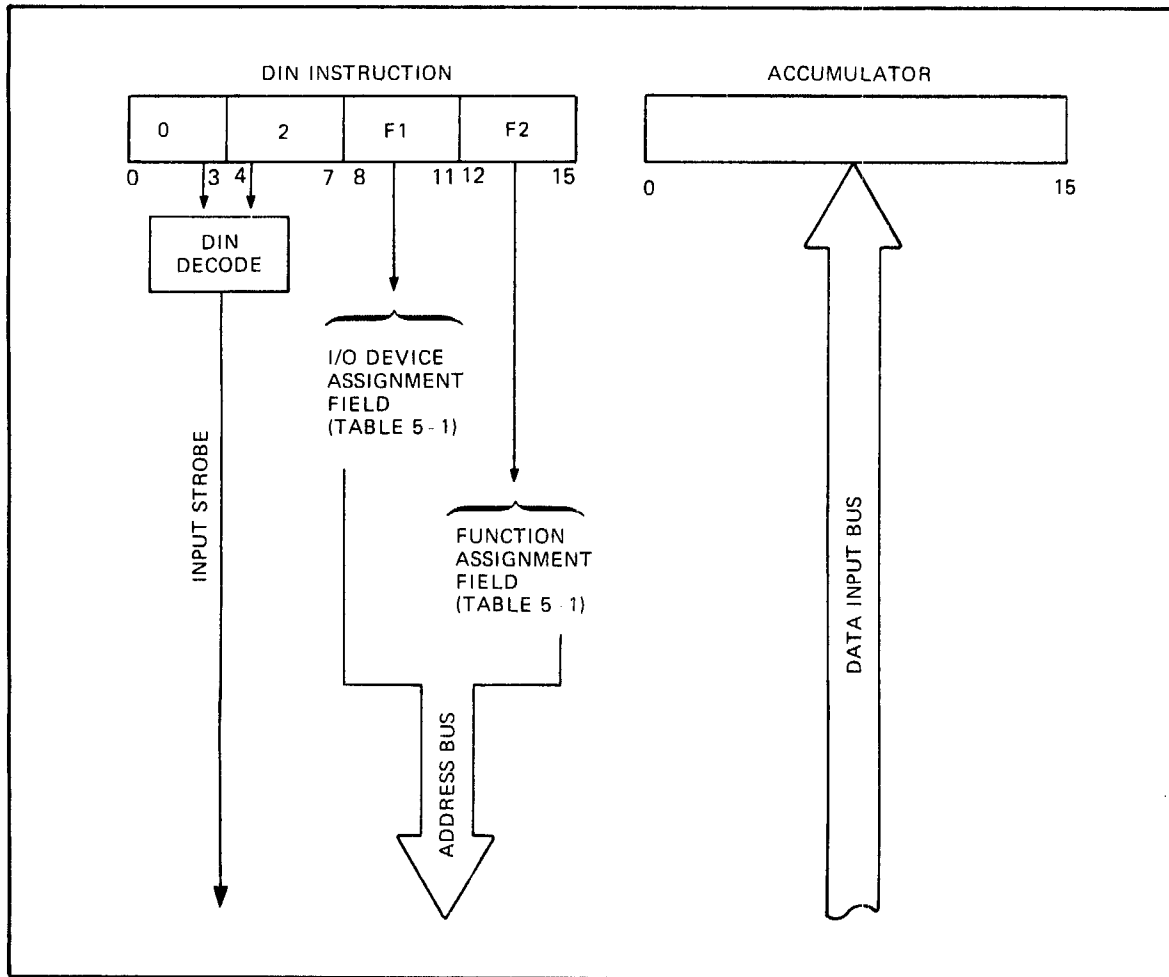


Figure 4 - 2. Direct Input Operation

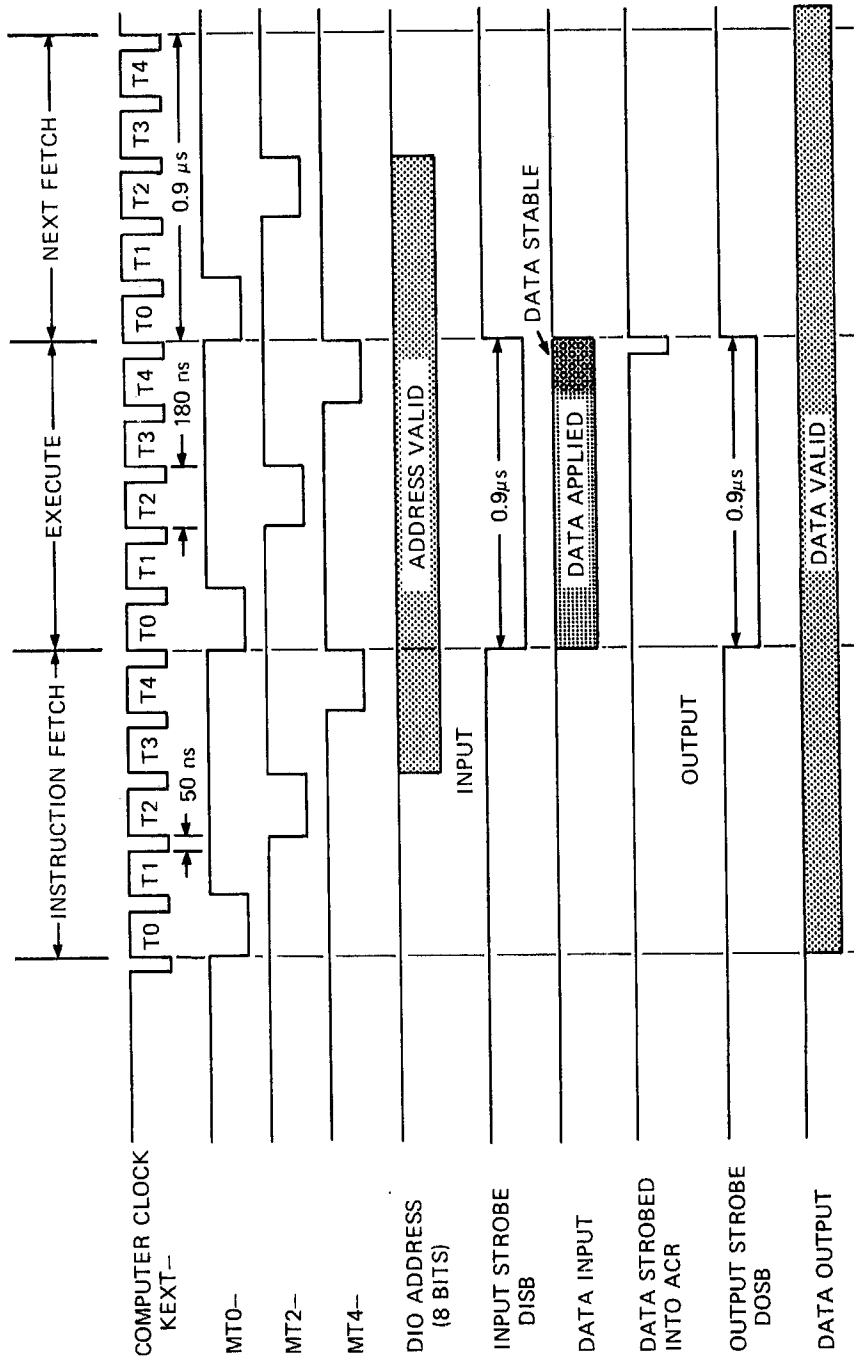


Figure 4-3. DIO Interface Timing

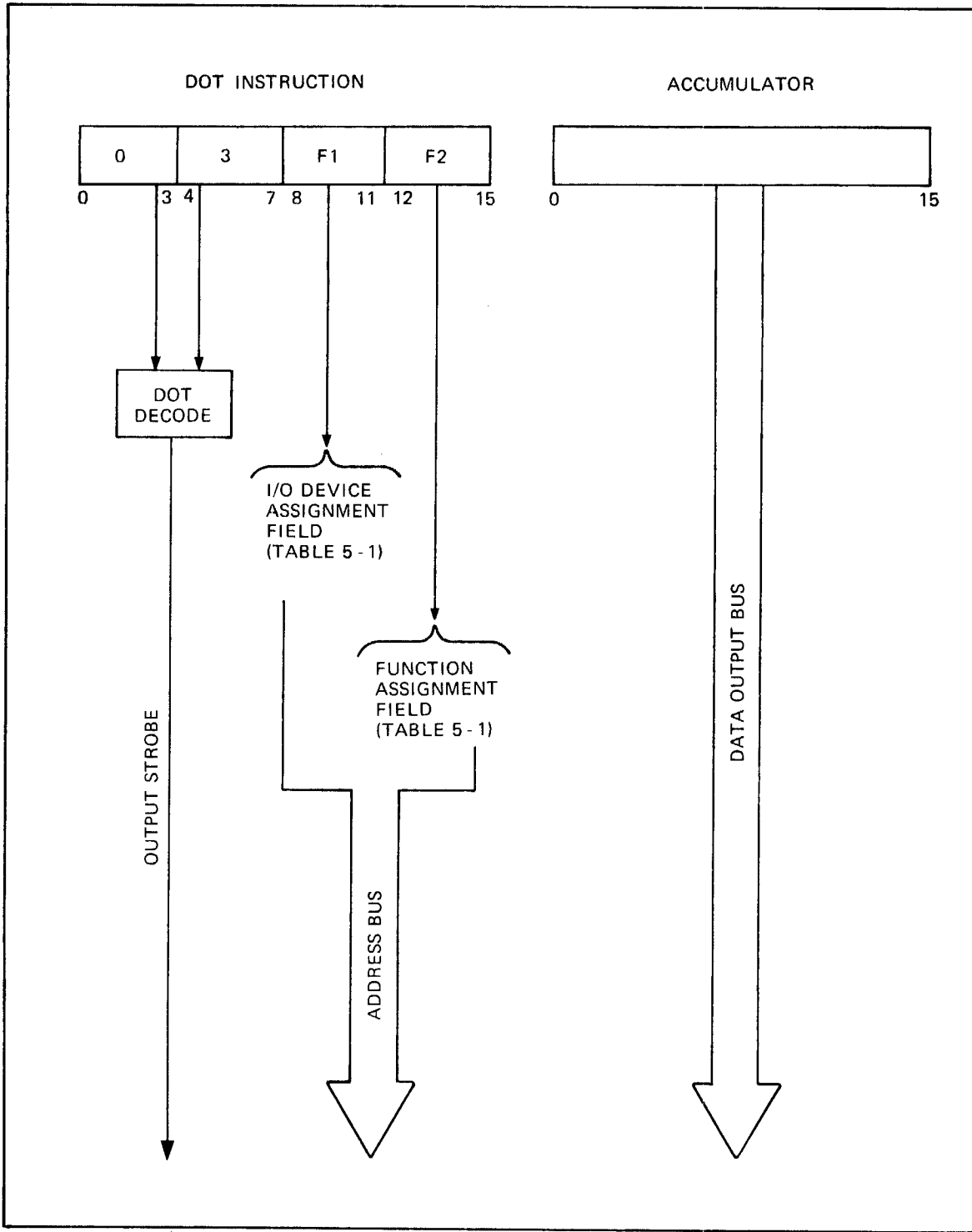


Figure 4 - 4. Direct Output Operation

4-3 DIRECT MEMORY ACCESS (DMA) CHANNEL

4-3.1 General Description

The direct memory access option permits up to six external devices to communicate directly with the memory without going through the CPU data channels. Direct memory accessing does not interfere with normal CPU operation unless both require access to memory during the same machine cycle.

Program execution and direct-memory accessing proceed independently at the same time, until both require access to memory simultaneously. At this point, the DMA is given precedence over the CPU and program execution is suspended until a machine cycle is available to the CPU. Operation then resumes until both require access simultaneously again. Program execution is suspended by suppressing the distribution of clock pulses in the CPU.

The DMA channel consists of a 15-bit address bus, a 16-bit input bus, a 16-bit output bus, 6 memory request lines, 6 memory write lines, 6 memory acknowledge lines, a system reset line and 4 terminator voltage lines. These buses and signals are shown in Figure 4-5 and described in table 4-2.

4-3.2 Control

The DMA system requires a DIO interface in the peripheral controller in addition to the DMA interface. The DIO (DOT and DIN) commands set the memory starting address for data transfer, set device selection, set the word count to be transferred, and initiate the data transfer (either read or write). In addition, the DIO system receives status and disconnects any of the peripheral controllers.

After issuing the command to initiate a data transfer, the transfer of data between the peripheral device and the 706 memory proceeds automatically and without further intervention by the program. Data is transferred starting with the word coming from or going to the initial memory address specified by a previous DIO command. The data transfer continues until the initial word count has been decremented to zero.

When the word count finally reaches zero, indicating that the data transfer has been completed, the peripheral controller interrupts the 706 CPU. The 706 program then reacts the same as when data has been completely transferred via the DIO method.

The main advantage of transferring data via the DMA system instead of the DIO system is that the overhead (memory space and execution time) of handling every data word under program control is eliminated. After the DMA peripheral device has been initially set up and selected, the 706 user can proceed with additional processing while awaiting the transfer-complete interrupt (Figure 4-6).

The device addresses for identifying DMA peripheral controllers are identical to those for DIO (see Table 5-1). The function codes for each device are defined in Section 5.

4-4 CABLING AND INTERFACE

Both the DIO and DMA cables use twisted pair wire for maximum noise rejection. All wires are TFE coated, 22 gage, silver plated copper (Table 4-3).

The circuits used in the 706, including cable drivers and receivers, belong to the DTL integrated circuit family. Logic levels for both DIO and DMA bus signals are: True "1" is 0 to 0.6V and False "0" is nominally +3.0V.

4-4.1 DIO Cable

All DIO signals are available on two Elco 90 pin connectors which are connected in parallel. Each connector carries a full set of the DIO signals. These connectors are labeled J1 and J2, and are mounted on the back panel of the CPU chassis. Table 4-4 gives the pin assignment for the DIO connector.

Cables used for the DIO channel may have a total length of up to 50 feet.

The two ends of the DIO cable must be terminated with resistive networks as shown in Figure 4-7. These networks are packaged in standard DIO termination caps. This facilitates changing location of these terminations when the relative position of the peripheral devices is changed.

4 - 4.2 DMA Cable

All DMA signals are available on an Elco 120-Pin Connector which is mounted on the back panel of the CPU chassis. This connector is labeled J5. Table 4-5 gives the pin assignment for the DMA connector.

The DMA cables are connected directly to the memory cables which are terminated at the last memory module. The DMA memory cables may have a maximum total length of 24 feet.

The last DMA connector in a chain must be terminated with resistive networks as shown in Figure 4-7. These networks are packaged in a DMA termination cap.

4 - 4.3 Cabling of Peripheral Devices

Figure 4-8 illustrates cabling of peripheral devices to the CPU. These are interconnected as a daisy chain, which requires two connectors for each DIO device and two connectors for each DMA device. Note that a DMA device also requires the DIO channel.

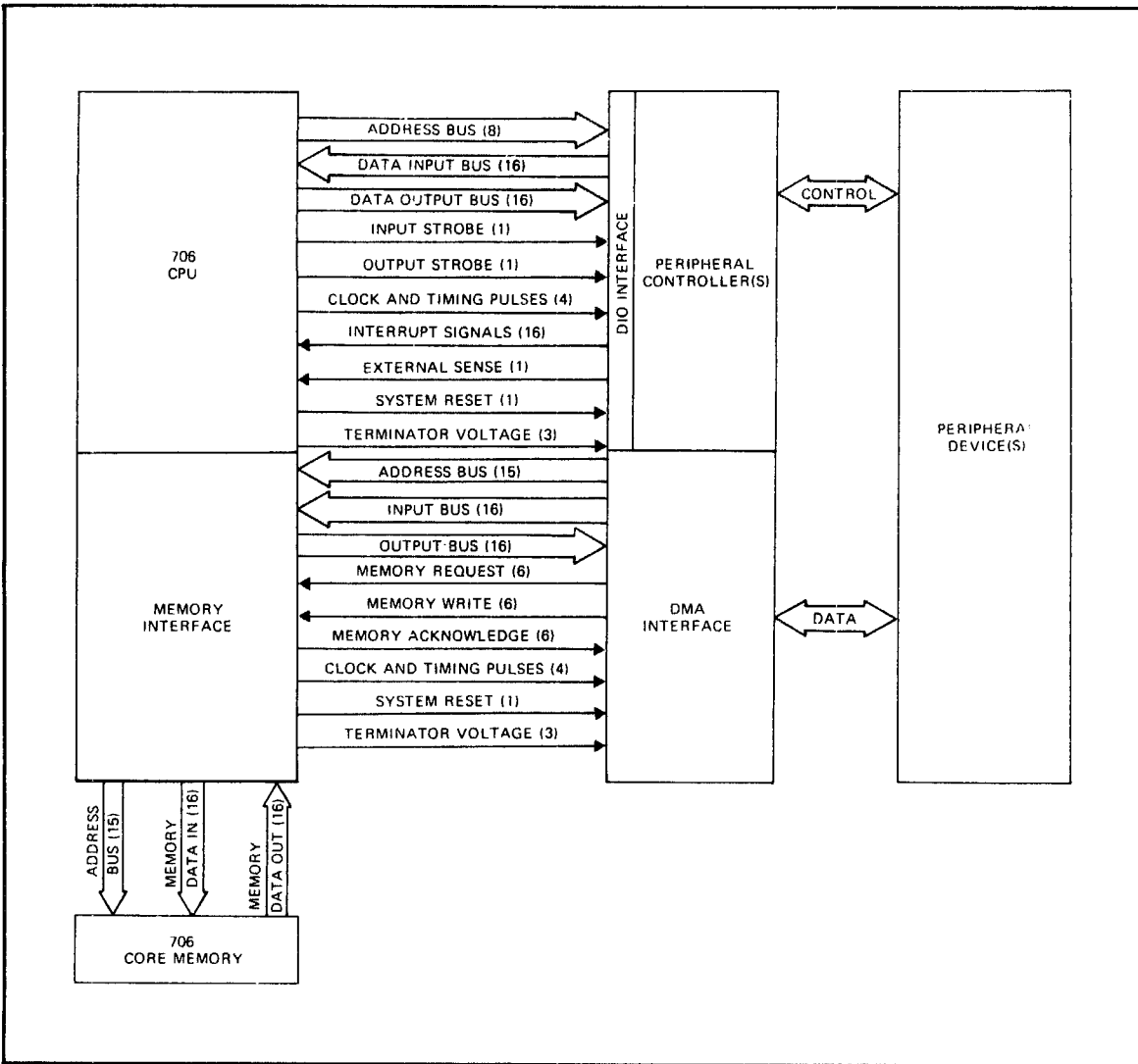


Figure 4 - 5. Direct Memory Access System

Table 4-2. DMA Channel Signals

SIGNAL (No. of Lines)	PURPOSE
Address Bus (15) MMAD01–MMAD15	A 15-bit bus carrying the address of the memory location to be read from or written into by a peripheral device.
Input Bus (16) MMDI00–MMDI15	A 16-bit input bus carrying the data to be written into the memory location specified by the DMA Address Bus.
Output Bus (16) MMDO00–MMDO15	A 16-bit output bus carrying the data read from the memory location specified by the Address Bus.
Memory Request (6) MRQ2–MRQ7	A DMA device uses one of the six request lines to request a memory cycle from the CPU. Since it is not possible for the CPU to grant a memory cycle to all the requesting devices simultaneously, each request line has a fixed priority. These request lines are assigned numbers from 2 to 7 with the lowest numbered line having the lowest priority.
Memory Write (6) MWT2–MWT7	A DMA device utilizes a memory write line with its corresponding Memory Request line to denote whether the request is for a read or a write cycle. If the write line is true, the request is for a write cycle; if false, the request is for a read cycle.
Memory Acknowledge (6) MAK2–MAK7	The CPU responds to a memory request by setting the correspond Memory Acknowledge line true. The Acknowledge line informs the peripheral device that the data transfer is enabled.
Clock and Timing Pulses (4) KEXT, DMT0, DMT2, DMT4	These four lines enable the peripheral devices to synchronize with the memory. DMT0, DMT2, and DMT4 are pulses with a period of 900 nsec. and a width of 180 nsec. KEXT is a pulse with a period of 180 nsec. and a width of 50 nsec.
System Reset (1) REXT	System reset is performed when the RESET pushbutton (Figure 7-1) on the CPU control panel is pressed, or when an initial power-on sequence is performed. The reset line initializes a peripheral device controller, i.e., reset registers and halt the device operation. When a reset occurs the CPU is in the HALT state, all pending interrupts are eliminated, and all CPU registers are reset.
Terminator Voltage (3) V + 3.0	This voltage is required for the resistor networks used to terminate the DMA lines.

Table 4 -3. DIO/DMA Wire, Electrical and Cable Specifications

<u>WIRE TYPE</u>		<u>DIO CABLE</u>	
Wire	Silver plated copper	Connector Type	Elco 90-pin
Size	19 strands of 34 gage	Pin Assignment	See Table 4 - 4
Twists	12-16 twists per foot	Maximum Length	50 feet
Coating	TFE, 0.010 thick	Part No. for 72-inch cable	Raytheon 279558-072
Insulation	600 volt	Termination	Resistor Network (Figure 4 -7)
Specification Dwg. No.	531420	Termination Cap Part No.	Raytheon 279767
<u>CIRCUITS</u>		<u>DMA CABLE</u>	
IC Type	TTL	Connector Type	Elco 120-pin
Supply Voltage	+5 volts \pm 10%	Pin Assignment	See Table 4 - 5
Specification Dwg. No.	531593	Maximum Length	24 feet
		Part No. for 72-inch cable	Raytheon 279557-072
		Termination	Resistor Network (Figure 4 -7)
		Termination Cap Part No.	Raytheon 279768

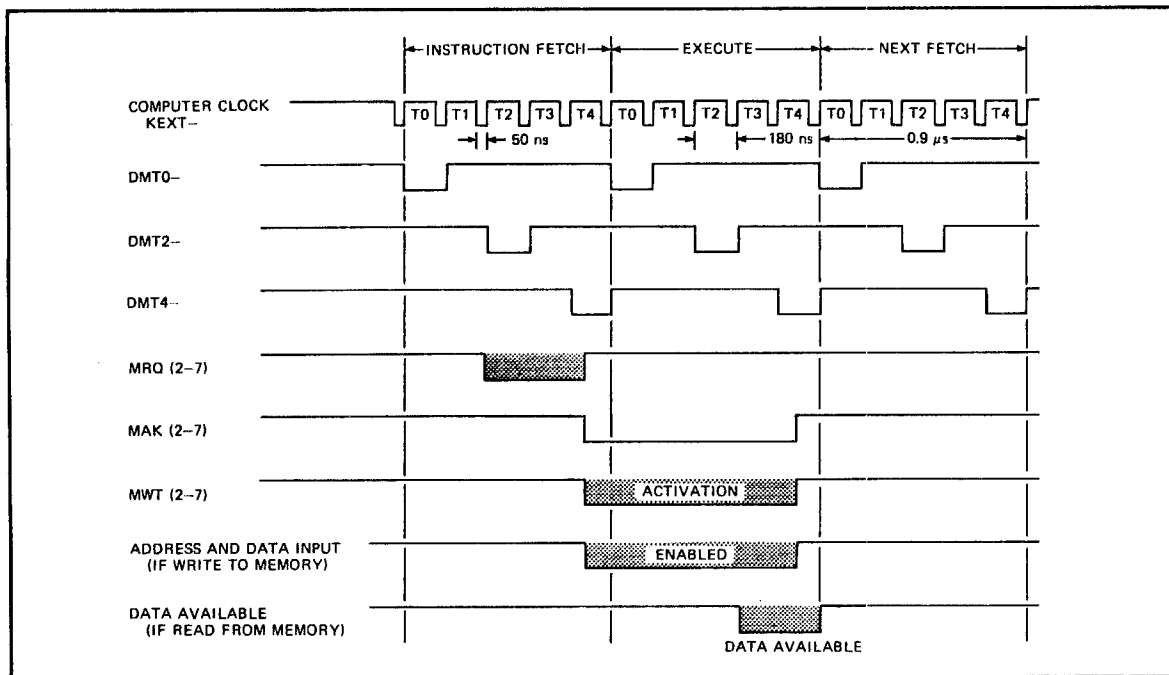


Figure 4 - 6. DMA Interface Timing

Table 4 - 4. Direct I/O Connectors J1, J2

Signal	Signal Pin	Ground Pin	Signal	Signal Pin	Ground Pin
DIN00-	A	J	DAD10-	BB	BC
DIN01-	B	J	DAD11-	BH	BC
DIN02-	H	J	DAD12-	BD	BK
DIN03-	C	L	DAD13-	BE	BK
DIN04-	D	L	DAD14-	BJ	BK
DIN05-	K	L	DAD15-	BL	BN
DIN06-	E	N	DOSB-	BM	BN
DIN07-	F	N	DISB-	BF	BN
DIN08-	M	N	KEXT-	BP	BX
DIN09-	R	AE	EXSENS-	BR	BX
DIN10-	X	AE	REXT-	BS	BZ
DIN11-	AM	AE	V+3.0A	BT	BZ
DIN12-	S	Z	V+3.0B	CZ	DA
DIN13-	T	Z	V+3.0C	DB	DA
DIN14-	Y	Z	MT0-	BY	BZ
DIN15-	U	AB	MT2-	BU	CB
DOT00-	V	AB	MT4-	BV	CB
DOT01-	AA	AB	IRPT00-	BW	BX
DOT02-	P	W	IRPT01-	CA	CB
DOT03-	AC	W	IRPT02-	CC	CE
DOT04-	AD	W	IRPT03-	CD	CE
DOT05-	AF	AP	IRPT04-	CV	CE
DOT06-	AH	AP	IRPT05-	CF	CP
DOT07-	AN	AP	IRPT06-	CH	CP
DOT08-	AJ	AS	IRPT07-	CN	CP
DOT09-	AK	AS	IRPT08-	CJ	CS
DOT10-	AR	AS	IRPT09-	CK	CS
DOT11-	AL	AU	IRPT10-	CR	CS
DOT12-	AT	AU	IRPT11-	CL	CU
DOT13-	AY	AU	IRPT12-	CM	CU
DOT14-	AV	BA	IRPT13-	CT	CU
DOT15-	AW	BA	IRPT14-	CW	CX
DAD08-	AZ	BA	IRPT15-	CY	CX
DAD09-	AX	BC			

Table 4 - 5. Direct Memory Access Connector J5

Signal	Signal Pin	Ground Pin	Signal	Signal Pin	Ground Pin
MRQ2-	A	L	MMDI04-	BH	BW
MRQ3-	B	L	MMDI05-	BP	BW
MRQ4-	K	L	MMDI06-	CC	BW
MRQ5-	C	N	MMDI07-	BJ	BS
MRQ6-	D	N	MMDI08-	BK	BS
MRQ7-	M	N	MMDI09-	BR	BS
MAK2-	E	R	MMDI10-	BL	BY
MAK3-	F	R	MMDI11-	BT	BY
MAK4-	P	R	MMDI12-	BX	BY
MAK5-	H	T	MMDI13-	BU	CB
MAK6-	J	T	MMDI14-	BV	CB
MAK7-	S	T	MMDI15-	CA	CB
MWT2-	V	AE	MMDO00-	BZ	CF
MWT3-	W	AE	MMDO01-	CD	CF
MWT4-	AD	AE	MMDO02-	CE	CF
MWT5-	X	AH	MMDO03-	CH	CK
MWT6-	Y	AH	MMDO04-	CJ	CK
MWT7-	AF	AH	MMDO05-	CL	CK
MMAD01-	AJ	AK	MMDO06-	CM	CX
MMAD02-	AB	AM	MMDO07-	CN	CX
MMAD03-	AC	AM	MMDO08-	CW	CX
MMAD04-	AL	AM	MMDO09-	CP	CZ
MMAD05-	U	AN	MMDO10-	CR	CZ
MMAD06-	AW	AN	MMDO11-	CY	CZ
MMAD07-	AX	AN	MMDO12-	CS	DB
MMAD08-	AP	AZ	MMDO13-	CT	DB
MMAD09-	AR	AZ	MMDO14-	DA	DB
MMAD10-	AY	AZ	MMDO15-	CU	DD
MMAD11-	AS	BB	MT0-	DJ	DU
MMAD12-	AT	BB	MT2-	DT	DU
MMAD13-	BA	BB	MT4-	DM	DW
MMAD14-	AU	BD	KEXT-	DV	DW
MMAD15-	AV	BD	REXT-	DN	DY
MMDI00-	BC	BD	V+3.0G	EF	EE
MMDI01-	BE	BN	V+3.0H	EH	EK
MMDI02-	BF	BN	V+3.0J	EJ	EK
MMDI03-	BM	BN	V+3.0K	EL	EK

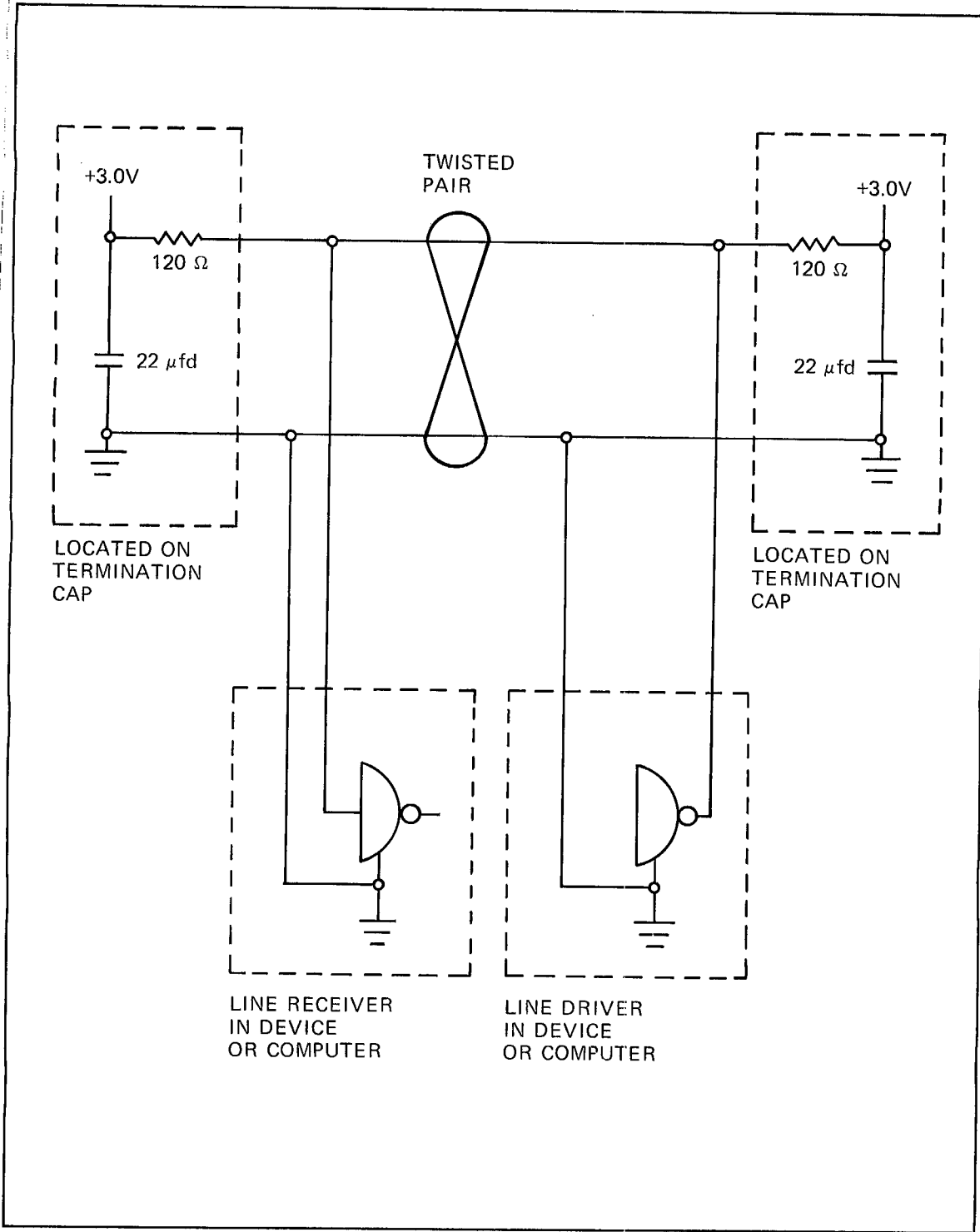


Figure 4 - 7. Cable Termination Scheme

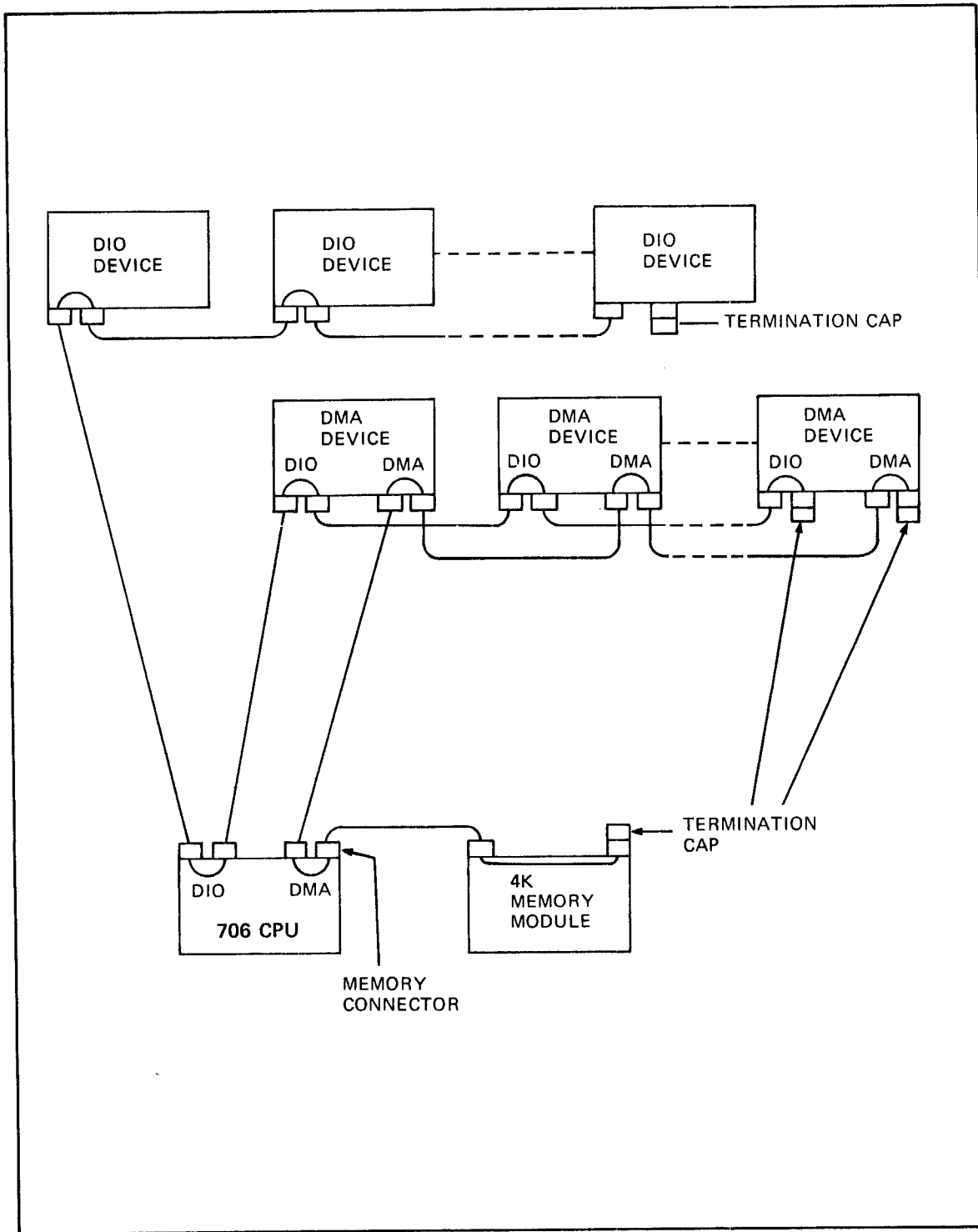


Figure 4 - 8. Cabling of Peripheral Devices

SECTION 5
PERIPHERAL EQUIPMENT
OPERATION AND PROGRAMMING

5-1 GENERAL

This section contains programming requirements for data transfer between the 706 and peripheral devices.

5-1.1 Device Selection

All peripheral devices are selected for operation by the issuance of a DIN or DOT instruction. Bits 8-11 and 12-15 (F2) of these instructions specify the device and function selected respectively. Table 5-1 depicts the assigned device/function codes in hexadecimal along with their associated operations. The function codes are listed for DOT (Direct Output) instructions, unless the codes are followed by "(I)", in which case the assignments are associated with DIN (Direct Input) instructions. Note: The "(I)" is not part of the function code and is shown in the table for clarification only.

NOTE: Throughout Section 5 the abbreviated notation for specifying device and function codes in hexadecimal omit the hexadecimal identifier (X") - do not omit this identifier (without equating the symbols A-E) when coding for assembler input.

For example:

DOT C, D as shown in text should be coded as

- a. DOT X'C', X'D' or
- b. DOT 12, 13 or
- c. C EQU 12
D EQU 13
DOT C, D

Table 5-1. Device/Function Codes & Operations

Peripheral	Codes		Operation
	Device (F1)	Function (F2)	
Digital Plotter	0	0	Disconnect controller
		1	Select plotter and plot with data and command in accumulator.
		5	Plotter reset.
Disc(s)	1	0 (I)	Return status to accumulator.
		0	Disconnect controller.
		1	Set memory address.
		2	Set track and sector.
		4	Set unit number, number of words and write. The cyclic code is not checked during a write (see code No. 7).
		6	Set unit number, number of words and read.

Table 5-1. Device/Function Codes & Operations (Cont.)

Peripheral	Codes		Operation
	Device (F1)	Function (F2)	
DMA Magnetic Tape	2	7	Set unit number, number of words and verify data (no data is transferred to the computer during the verify operation; however, the cyclic code is checked).
		0 (I)	Return the status of disc unit 0 into the accumulator.
		1 (I)	Return the status of disc unit 1 into the accumulator.
		2 (I)	Return the status of disc unit 2 into the accumulator.
		3 (I)	Return the status of disc unit 3 into the accumulator.
		0	Disconnect controller.
		1	Set memory address.
		3	Continue last operation; i.e., read or write.
		4	Set word count and write a record.
		5	Write an end-of-file.
	6	Set word count and read a record.	
	7	Write 3" of blank tape.	
	8	Optional — set data chain address.	
	9	Optional — set data chain word count.	
	A	Optional — stop data chain.	
	B	Rewind.	
	C	Backspace one record.	
	D	Advance tape to next end-of-file.	
	0 (I)	Return status of unit 0 in the accumulator.	
	1 (I)	Return status of unit 1 in the accumulator.	

Table 5-1. Device/Function Codes & Operations (Cont.)

Peripheral	Codes		Operation
	Device (F1)	Function (F2)	
ADC (s)	3	2 (I)	Return status of unit 2 in the accumulator.
		3 (I)	Return status of unit 3 in the accumulator.
		4 (I)	Return current memory address to the accumulator.
		0	Disconnect
		1	Reset error
		2	Set random mode unlocked.
		6	Set sequential mode unlocked.
		A	Set random mode clocked.
		E	Set sequential mode clocked.
		0 (I)	Return status to the accumulator.
DAC (s)	4	1 (I)	Transfer data.
		3 (I)	Transfer data and start.
		0-F	Transfer data.
Line Printer	5	0	Disconnect controller.
		2	Move paper one space.
		3	Move paper to top of form.
		4	Load one character.
		8	Print.
		9	Print and move paper.
		0 (I)	Return status to accumulator.
Buffered Input Channel (s)	6	0 (I)	Return status to accumulator.
		1 (I)	Input data on channel 1.
		2 (I)	Input data on channel 2.
		3 (I)	Input data on channel 3.
		4 (I)	Input data on channel 4.
Buffered Output Channel (s)	7	1	Transfer data and disconnect channel 1.

Table 5-1. Device/Function Codes & Operations (Cont.)

Peripheral	Codes		Operation
	Device (F1)	Function (F2)	
Card Reader	8	2	Transfer data and disconnect channel 2.
		3	Transfer data and disconnect channel 3.
		4	Transfer data and disconnect channel 4.
		9	Transfer data on channel 1.
		A	Transfer data on channel 2.
		B	Transfer data on channel 3.
		C	Transfer data on channel 4.
		0 (I)	Return status to a accumulator.
		0	Disconnect controller.
		1	Read cards continuously and send interrupts for each column.
		2	Stop interrupts for the remainder of the card in process. Interrupts resume with next card.
		3	Do not select card after card in process has been read.
		8	Offset the card in process in output stacker.
DIO Magnetic Tape (Synchronous Read/ Synchronous Write)	9	0 (I)	Return status into the accumulator.
		4 or 5 (I)	Transfer 12 bits of data from card reader into the accumulator.
		0	Disconnect controller.
		2	Write EOF.
		3	Write 1 Record.
		9	Read 1 Record.
		A	Rewind.
B	Backspace		

Table 5-1. Device/Function Codes & Operations (Cont.)

Peripheral	Codes		Operation
	Device (F1)	Function (F2)	
DIO Magnetic Tape (Synchronous Read/ Incremental Write)	9	F	Transfer data out.
		0 (I)	Return status device 0 to accumulator.
		1 (I)	Return status device 1 to accumulator.
		2 (I)	Return status device 2 to accumulator.
		3 (I)	Return status device 3 to accumulator.
		F (I)	Transfer data in.
		0	Disconnect controller.
		2	Write EOF.
		3	Write 1 Record.
		9	Read 1 Record
		A	Rewind.
Teletype Multiplexer	A	B	Backspace
		7 or F	Transfer data out.
		0 (I)	Return status unit 0 to accumulator.
		7 or F (I)	Transfer data in.
		0	Disconnect controller.
		1	Set input mode per station.
Time-of-Day Clock	B	2	Set output mode per station.
		3	Output Character.
		0 (I)	Return status to accumulator.
		1 (I)	Input Character.
		0	Hold clock.
		1	Write clock word 1 (option)
		2	Write clock word 2 (option)
		4	Stop clock.
		5	Reset clock.

Table 5-1. Device/Function Codes & Operations (Cont.)

Peripheral	Codes		Operation
	Device (F1)	Function (F2)	
High Speed Paper Tape Punch	C	6	Start clock.
		1 (I)	Read clock word 1.
		2 (I)	Read clock word 2.
		3 (I)	Read clock word 3. (option).
		0	Disconnect controller.
		2	Turn on punch power.
		3	Turn off punch power.
High Speed Paper Tape Reader	D	6	Output character and turn on punch power.
		0 (I)	Return status to accumulator.
		0	Disconnect controller.
		1	Read forward.
		3	Read reverse.
		0 (I)	Return status to accumulator.
ASR Teletype	E	4 (I)	Transfer character & disconnect.
		5 (I)	Transfer character & read forward.
		0	Disconnect controller.
		1	Select paper tape reader (ASCII-8).
		2	Select printer/punch (ASCII-8).
		3	Select keyboard (ASCII-8).
		4	Output character and disconnect (ASCII-8).
		6	Output character (print/punch ASCII-8).
		8	Disconnect controller.
		9	Select paper tape reader (ASCII).
A	Select printer/punch (ASCII).		

Table 5-1. Device/Function Codes & Operations (Cont.)

Peripheral	Codes		Operation
	Device (F1)	Function (F2)	
Card Punch	F	B	Select keyboard (ASCII).
		E	Output character (print/punch ASCII).
		0 (I)	Return status to accumulator.
		4 (I)	Input character & disconnect (ASCII-8).
		5 (I)	Input character (paper tape reader - ASCII-8).
		7 (I)	Input character (keyboard ASCII-8).
		C (I)	Input character & disconnect (ASCII).
		D (I)	Input character (paper tape reader - ASCII).
		F (I)	Input character (keyboard - ASCII).
		0	Disconnect controller.
		1	Punch cards continuously and send interrupt for every two columns punched.
		2	Last punch.
		3	Stop punch.
		4 or 5	Load data.
		8	Offset card.
		9	Punch & offset card.
		A	Last punch & offset card.
B	Stop punch & offset card.		
C or D	Load data & offset card.		

5-1.2 Data Bit Assignment for DIO Transfer

All character transfers with DIO devices are made to and from the 706 accumulator (ACR). Since each device does not use the same bits of the ACR, Table 5-2 shows the standard peripheral devices

with their respective ACR bit assignments including the unit of transfer these bits represent. For inputting (DIN) unused bit positions are set to zero. For outputting (DOT), unused bit positions are unaffected.

Table 5-2. Peripheral Data Bit Assignment

Device	Accumulator Bits (msb-lsb)	Unit
ASR Teletype	8-15	Byte
High Speed Paper Tape Reader	8-15	Byte
High Speed Paper Tape Punch	8-15	Byte
Card Reader	4-15	One Card Column
Card Punch	4-15	One Card Column
DIO Magnetic Tape 9 track	0-15	Word } 2 Frames
7 track	0-11	
Plotter	2-7, 10-15	Byte
Real Time Clock	0-15	Word
Line Printer	10-15	Byte
Buffered Input Channels	0-15	Word
Buffered Output Channels	0-15	Word
Miniverter	0-12	Word
Multiverter	0-14	Word
Digital-Analog Converter	0-15	Word

DMA 7 and 9-track Magnetic Tape and Disc do not use accumulator but are full word (0-15) transfer devices.

5-1.3 Status Bit Assignments

All peripheral devices (except Time-of-Day Clock and DAC) return a status word to the accumulator within 1.8 μ seconds after issuing a DIN instruction with a function code of "0". This status word can be used to determine the readiness of the controller and/or device to transmit data or accept a new command. Error information (e.g., parity error,

hopper empty), terminal conditions (e.g., end of tape) and general information (e.g., write enabled) are also available via the status word. The normal case is usually indicated by a status response word of all binary zeros. Table 5-3 gives the status bit assignment and meaning when the bit is true or present.

Table 5-3. Peripheral Status Bit Assignment and Meaning

Device	Bit	Meaning When True ("on" or "1")
Plotter	0	Controller has been disconnected.
	1	Controller power on.
	7	Plotter controller busy (no interrupt).
Disc	0	Controller or device not ready.
	13	Write command issued to a protected track.
	14	Rate error.
	15	Rate or CRC error
DMA Magnetic Tape	0	Controller or device not ready.
	1	Device not ready.
	2	Beginning of tape or load point.
	3	End of tape.
	4	Rewinding.
	5	Density selected is 200 BPI, 7-track only.
	6	Density selected is 556 BPI, 7-track only
	7	Format switch is 1, 7-track only.
	8	Format switch is 2, 7-track only.
	9	Format switch is 3, 7-track only.
	10	Magnetic tape continue was last operation.
	11	End of file detected.
	12	Transferred byte count (0 = even; 1 = odd).
	13	Write enabled (write ring in).
	14	Rate error.
15	Parity or CRC error, or rate error.	
ADC	0	Device busy.
	14	Frame Sync.
DAC	15	Error.
		No status necessary

Table 5-3. Peripheral Status Bit Assignment and Meaning (Cont.)

Device	Bit	Meaning When True ("on" or "1")
Line Printer	0	Controller or device not ready
	1	Device not ready.
	7	Controller sending interrupt
Buffered Input Channel (s)	0	Channel 1 interrupting.
	1	Channel 2 interrupting.
	2	Channel 3 interrupting.
	3	Channel 4 interrupting.
Buffered Output Channel (s)	0	Channel 1 interrupting.
	1	Channel 2 interrupting.
	2	Channel 3 interrupting.
	3	Channel 4 interrupting.
Card Reader	0	Controller or device not ready.
	1	Controller not ready.
	2	Card moving through reader.
	3	Reader input hopper empty.
	7	Interrupt, no DIN received.
	14	Response error. Late DIN from computer.
	15	Response error or card reader malfunction.
DIO Magnetic Tape (Synchronous & Incremental)	0	Controller or device not ready.
	1	Device not ready.
	2	Beginning of tape or load point.
	3	End of tape.
	4	Rewinding.
	6	Function finished interrupt true.
	7	Data transfer request interrupt true.
	11	End of file detected.
	13	Write enabled (write ring in)
	14	Rate error.
15	Parity or CRC error or rate error.	
Teletype Multiplexer	0	Channel waiting to be serviced.

Table 5-3. Peripheral Status Bit Assignment and Meaning (Cont.)

Device	Bit	Meaning When True ("on" or "1")
	1	Unwanted message on remote TTY.
	15	Controller not ready.
Time-of-Day Clock		No status necessary.
High Speed Paper Tape Punch	0	Controller not ready.
	1	Device not ready.
	4	Tape low.
	7	Interrupt has not been serviced.
High Speed Paper Tape Reader	0	Controller not ready.
	1	Device not ready.
	7	Interrupt has not been serviced.
ASR Teletype	0	Teletype has been selected.
	7	Interrupt has not been serviced.
Card Punch	0	Controller or device not ready.
	1	Controller not ready.
	7	Interrupt has not been serviced.
	13	Punch check error.
	14	Response error – DOT late.
15	Response error, card punch malfunction, or punch error.	

5-2 DIO PROGRAMMING

There are three methods of transmitting data via the DIO channel: Status Testing, Burst, and Interrupt.

5-2.1 Status Testing Method

The status testing method of programming the DIO channel involves requesting status with a DIN DEV, 0 instruction. The response is then tested to determine if the device is ready to send or receive the next character. Interrupts are not used and the interrupt level corresponding to the particular device should either be disabled, with a DSB instruction, or all interrupts should be masked with an MSK instruction, prior to selection of the device. Figure 5-1 and Table 5-4 show a sample Status Test Method of programming for any device that uses status bit 7 as an indicator of "device ready to transmit or receive data".

The MSK and UNM instructions in the routine shown in Table 5-4 are optional. As shown, no other peripheral device will be serviced until the device being serviced by the status method has completed its data transfer and has been disconnected. It may be that the user wants the above routine to be interruptable, in which case the MSK and UNM instructions may be omitted.

There are five major sections that comprise a routine using the status testing method of programming: Protection, Selection, Wait for Data, Ready, Service Data, and Test for End of Transmission & Disconnect.

5-2.1.1 Protection

Protection controls the ability of any device (including the device being serviced) to interrupt the status testing routine. In the status test mode, it is imperative that the device being serviced have its corresponding interrupt level disabled and/or all interrupts masked. If other devices are to be allowed to interrupt during the execution of the status test routine, the MSK instruction must not be used. (See Interrupt Method, Sec. 5-2.3).

5-2.1.2 Selection

Selection of the I/O device enables its controller to set the proper status bits, indicating that the device is ready to send or receive data. For devices that require motion (e.g., card reader), the data medium (e.g., card) is fed into the read or write station.

5-2.1.3 Wait for Data Ready

Wait for data ready is implemented by requesting status with a DIN DEV, 0 instruction and then examining status bit-7 (see Table 5-3). If the data is not ready to be transferred, the routine normally returns to the status request instruction.

5-2.1.4 Service Data

Servicing data is accomplished by issuing the proper DOT or DIN instruction after it has been determined the data is ready for transmittal. If the 706 is sending data, the word must have been first loaded into the accumulator. If the 706 is receiving data the previous word must have been processed and the accumulator made ready for this word.

5-2.1.5 End of Transmission & Disconnect

Testing for end-of-transmission is a technique that must be employed in order to determine when the data transfer is complete. Normally, two methods are used: Special Character and End-of-Function.

The Special Character method (Logical Record) requires the routine to check for the transfer of a "key" or "end" character or count the number of characters transmitted.

The End-of-Function method (Physical Record) requires that the routine examines a status bit that signifies that the physical record has ended (e.g., one card completely processed or one magnetic tape record read). After the last word has been transmitted a DOT DEV, 0 instruction will de-select the peripheral device and prepare it to be selected at another time.

Note: There are certain cases where the I/O device should not be disconnected and these cases are described later in this section under individual device descriptions.

5-2.2 Burst Method

The burst method of programming the DIO channel uses the external sense feature of the 706 to notify the central processor when a device is ready to be serviced. The burst method is very similar to the status test method of programming; the major difference being that the external sense line is interrogated with an SSE instruction instead of asking for and checking the status response word (Table 5-5).

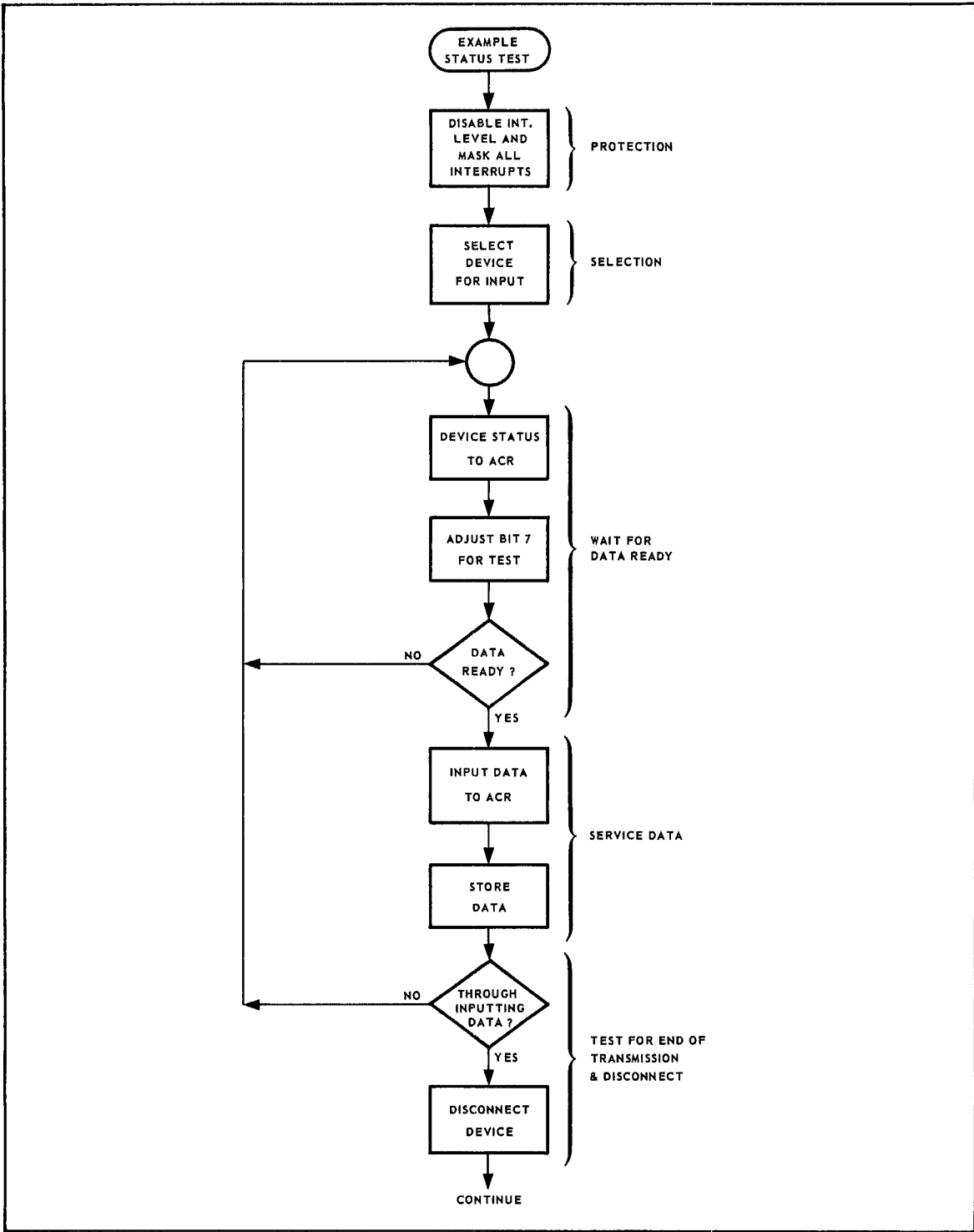


Figure 5-1. DIO Status Test Input Routine Flow

Table 5-4. DIO Status Test Method Coding Example

```

1 *   DIO STATUS TEST METHOD CODING EXAMPLE (INPUT)
2 *
0000 0040   3   SLM           SET LOCAL INDEXING MODE
0001 0012   4   SMR   COUNT   SET INDEX TO COUNT
0002 9012   5   LDX   COUNT   INHIBIT DEVICE INTERRUPT
0003 0030   6   DSB   LEVEL   INHIBIT ALL INTERRUPTS (OPTIONAL)
0004 00A0   7   MSK
0005 0300   8   DOT   DEV,FUNCT1 SELECT DEVICE
0006 0200   9 GETSTAT  DIN   DEV,0   RETURN STATUS TO ACCUMULATOR
0007 0A17  10   SLL           WAIT FOR DATA READY
0008 0820  11   SAM
0009 1006  12   JMP   GETSTAT   DATA NOT READY, GET STATUS AGAIN.
000A 0200  13   DIN   DEV,FUNCT2 SERVICE DATA
000B 7877  14   STW * BUFFER+100 STORE DATA
000C 0401  15   IXS   1         INCREMENT COUNTER
000D 1006  16   JMP   GETSTAT
000E 0300  17 DISCDEV  DOT   DEV,0   END-OF-TRANSMISSION
000F 0080  18   UNM
0010 0050  19   SGM           ALLOW INTERRUPTS (OPTIONAL)
0011 0000  20   HLT           SET GLOBAL MODE
0012 FF9C  21 COUNT   DATA  -100   TWO'S COMPLEMENT OF BLOCK LENGTH
      0000  22 DEV     EQU    0       DEVICE CODE GOES HERE
      0000  23 FUNCT1 EQU    0       FUNCTION CODES GO HERE
      0000  24 FUNCT2 EQU    0       (SEE TABLE 5-1)
      0000  25 LEVEL  EQU    0       INTERRUPT LEVEL
26 *
0013 0064  27 BUFFER  RES    100      STORAGE AREA
28   END

```

Table 5-5. DIO Burst Method Coding Example

```

1 *   DIO BURST METHOD CODING EXAMPLE (INPUT)
2 *
0000 0040   3   SLM           SET LOCAL INDEXING MODE
0001 0010   4   SMR   COUNT   SET INDEX TO COUNT
0002 9010   5   LDX   COUNT   INHIBIT DEVICE INTERRUPT
0003 0030   6   DSB   LEVEL   INHIBIT ALL INTERRUPTS (OPTIONAL)
0004 00A0   7   MSK
0005 0300   8   DOT   DEV,FUNCT1 SELECT DEVICE
0006 0880   9 TESTEXTR SSE     TEST EXTERNAL SIGNAL
0007 1006  10   JMP   TESTEXTR  DATA NOT READY
0008 0200  11   DIN   DEV,FUNCT2 SERVICE DATA
0009 7875  12   STW * BUFFER+100 STORE DATA
000A 0401  13   IXS   1         INCREMENT COUNTER
000B 1006  14   JMP   TESTEXTR
000C 0300  15 DISCDEV  DOT   DEV,0   END-OF-TRANSMISSION
000D 0080  16   UNM           ALLOW INTERRUPTS (OPTIONAL)
000E 0050  17   SGM           SET GLOBAL INDEXING MODE
000F 0000  18   HLT
19 *
0010 FF9C  20 COUNT   DATA  -100   TWO'S COMPLEMENT BLOCK LENGTH
      0000  21 DEV     EQU    0       DEVICE CODE
      0000  22 FUNCT1 EQU    0       FUNCTION CODES
      0000  23 FUNCT2 EQU    0
      0000  24 LEVEL  EQU    0       INTERRUPT LEVEL
25 *
0011 0064  26 BUFFER  RES    100      STORAGE AREA
27   END

```

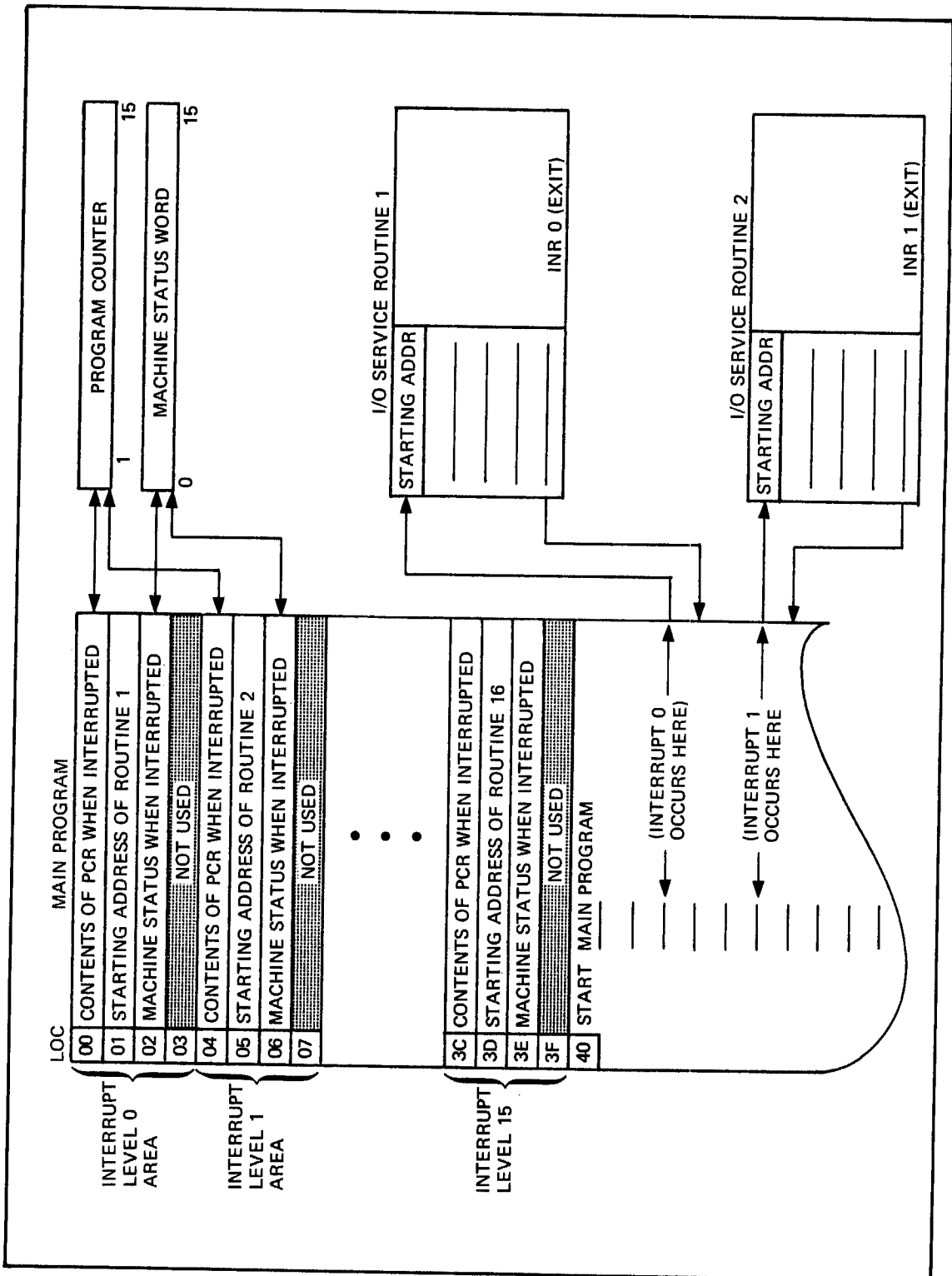


Figure 5-2. DIO Interrupt Method Block Flow Diagram

5-2.3 Interrupt Method

The interrupt method of programming the DIO channel relies on the automatic priority hardware of the 706 CPU to automatically transfer to the routine that services the I/O device. The interrupt method of DIO programming is normally employed when two or more I/O devices must operate concurrently and on a priority basis. In addition, the central processor can use the time between interrupts to process the data transmitted or to process other portions of the program.

Note: The user should be familiar with the principles of operation of the Automatic Priority Interrupt and I/O systems (Sections 3 and 4) before continuing with this section.

Figure 5-2 shows a simplified block flow diagram of a typical DIO interrupt sequence. Shown are two interrupt service routines (Routine 1 & 2), a main program (including interrupt working area 0-3F₁₆), the Program Counter and the Machine Status word. The diagram implies that the second interrupt (level 1) is serviced after the first interrupt (level 0) has been serviced. This need not necessarily be the case. The second interrupt could be serviced while the first interrupt is being serviced provided that the interrupts were not masked, the second interrupt was enabled, and the higher level interrupt occurred at some time during I/O Service Routine 1 time.

Figure 5-3 and 5-4 are flow diagrams of a typical DIO interrupt method program using two interrupt levels. Table 5-6 is the coding for the DIO interrupt method. Two levels of interrupt do not necessarily imply that two I/O devices are being used. For instance, the DIO magnetic tape has two interrupt levels available, "data ready" and "end-of-function".

The main program of any DIO interrupt method of programming must perform the following tasks:

5-2.3.1 Linkage Address Setup

The starting addresses of the I/O routines must be stored in the proper interrupt area locations. The formula for computing the interrupt area location for each level of interrupt is:

$$\text{Interrupt Linkage Address} = [4 \times \text{Level}] + 1$$

5-2.3.2 I/O Device Selection

Selection of an I/O device enables its controller to initiate motion (if required), generate an interrupt (read type device) or accept the first data transfer from the CPU (write type device).

5-2.3.3 Interrupt and Mask Control

Before a device may interrupt the CPU it must not only be selected but the appropriate interrupt level must be enabled (ENB) and the interrupt system must be unmasked. Only those interrupts that are to be used should be enabled.

5-2.3.4 Service First Character

If the I/O device(s) just selected is an output device (e.g., Line Printer) the first characters must normally be sent before the first interrupt can occur. Some devices allow the selection and sending of the first character at the same time, while other devices (e.g., DIO mag tape, card punch) require that the interrupt be processed before the first character is sent.

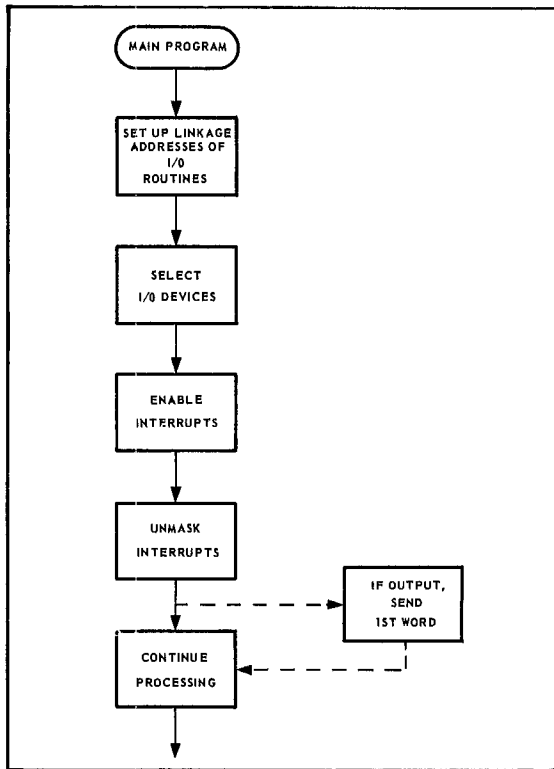


Figure 5-3. DIO Interrupt Main Flow Diagram

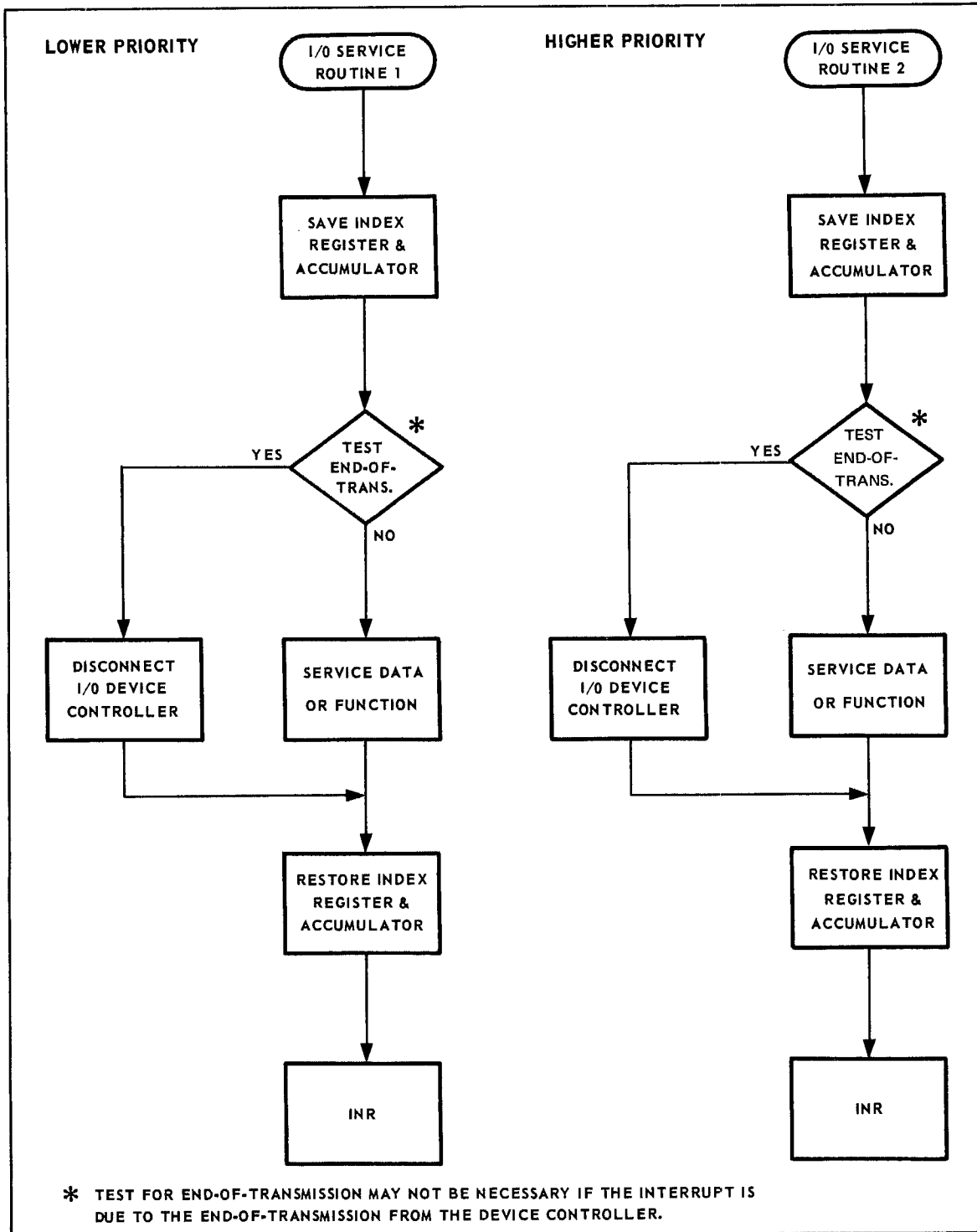


Figure 5-4. DIO Interrupt Service Routines Flow Diagram

Table 5-6. DIO Interrupt Method Coding Example

```

1 *   DIO INTERRUPT METHOD CODING EXAMPLE (OUTPUT)
2 *   RELOCATABLE PROGRAM (ASSUMES PROGRAM IS NOT LOADED ACROSS
3 *   WORD PAGE BOUNDRY)
4 *
5 *   MAIN PROGRAM
6 *
0000 8049   7       LDW   =ROUTINE1   SETUP LINKAGE ADDRESS
0001 0080   8       SML   0
0002 7001   9       STW   1
0003 804A  10       LDW   =ROUTINE2
0004 0080  11       SML   0
0005 7005  12       STW   5
0006 8048  13       LDW   =1
0007 703C  14       STW   COUNT1     INITIALIZE TABLE COUNTERS
0008 703D  15       STW   COUNT2
0009 0300  16       DOT   DEV1,FUNCT10  SELECT I/O DEVICES
000A 0300  17       DOT   DEV2,FUNCT20
000B 0020  18       ENB   0           ENABLE INTERRUPT LEVELS
000C 0021  19       ENB   1           0 AND 1
000D 0080  20       UNM   0           UNMASK ALL INTERRUPTS
000E 803E  21       LDW   TABLE1
000F 0300  22       DOT   DEV1,FUNCT11  OUTPUT 1ST WORD TO DEVICE 1,
0010 8044  23       LDW   TABLE2
0011 0300  24       DOT   DEV2,FUNCT21  OUTPUT 1ST WORD TO DEVICE 2,
25 *
26 *   CONTINUE PROCESSING
27 *
28 *
29 *
30 *   I/O SERVICE ROUTINE 1
31 *
0012 7038  32 ROUTINE1 STW   TEMP1A     SAVE ACCUMULATOR
0013 603A  33       STX   TEMP1X     SAVE INDEX
0014 803C  34       LDW   COUNT1     TEST FOR END OF
0015 F043  35       CMW   ENDCNT1    TRANSMISSION
0016 0870  36       SNE
0017 1021  37       JMP   DISCDEV1
0018 0130  38       CAX
0019 0040  39       SLM
001A 883E  40       LDW   * TABLE1     LOAD DATA
001B 0050  41       SGM
001C 0300  42       DOT   DEV1,FUNCT11  OUTPUT DATA
001D 0401  43       IXS   1
001E 0A10  44       NOP
001F 603C  45       STX   COUNT1     INCREMENT TABLE
0020 1022  46       JMP   RESTORE1    COUNTER
0021 0300  47 DISCDEV1 DOT   DEV1,0    DISCONNECT DEVICE 1
0022 8038  48 RESTORE1 LDW   TEMP1A     RESTORE ACCUMULATOR
0023 903A  49       LDX   TEMP1X     RESTORE INDEX
0024 0010  50       INR   0           EXIT
51 *
52 *

```

Table 5-6. DIO Interrupt Method Coding Example (Cont.)

```

>3 *
>4 *   I/O SERVICE ROUTINE 2
>5 *
>6 *
0025 7039 >7 ROUTINE2 STW  TEMP2A   SAVE ACCUMULATOR
0026 6038 >8          STX  TEMP2X   SAVE INDEX
0027 803D >9          LDW  COUNT2   TEST FOR END OF
0028 0870 60          SNE                    TRANSMISSION
0029 F048 61          CMW  ENDCNT2
002A 1034 62          JMP  DISCDEV2
002B 0130 63          CAX                    SERVICE DATA
002C J040 64          SLM                    LOAD DATA
002D 8844 65          LDW  * TABLE2
002E 0050 66          SGM                    OUTPUT DATA
002F 0300 67          UOT  DEV2,FUNCT21
0030 0401 68          IXS  1
0031 0A10 69          NOP                    INCREMENT TABLE
0032 603D 70          STX  COUNT2   COUNTER
0033 1035 71          JMP  RESTORE2
0034 0300 72 DISCDEV2 UOT  DEV2,0   DISCONNECT DEVICE 2
0035 8039 73 RESTORE2 LDW  TEMP2A   RESTORE ACCUMULATOR
0036 9038 74          LDX  TEMP2X   RESTORE INDEX
0037 0011 75          INR  1           EXIT
76 *
77 *
78 *
0038 0000 79 TEMP1A  DATA  0
0039 0000 80 TEMP2A  DATA  0
003A 0000 81 TEMP1X  DATA  0
003B 0000 82 TEMP2X  DATA  0
003C 0000 83 COUNT1  DATA  0
003D 0000 84 COUNT2  DATA  0
85 *
003E 0001 86 TABLE1 DATA  1
003F 0002 87          DATA  2
0040 0003 88          DATA  3
0041 0004 89          DATA  4
0042 0005 90          DATA  5
0043 0005 91 ENDCNT1 DATA  5
92 *
0044 00C1 93 TABLE2 DATA  'A'
0045 00C2 94          DATA  'B'
0046 00C3 95          DATA  'C'
0047 00C4 96          DATA  'D'
0048 0004 97 ENDCNT2 DATA  4
98 *
0000 99 DEV1   EQU  0   DEVICE CODES HERE
0000 100 DEV2 EQU  0   (SEE TABLE 5-1)
0000 101 FUNCT10 EQU  0 FUNCTION CODES HERE
0000 102 FUNCT11 EQU  0 (SEE TABLE 5-1)
0000 103 FUNCT20 EQU  0
0000 104 FUNCT21 EQU  0
105          END

0049 J012
004A 0025
004B J001

```

5-2.3.5 Continue Processing

Before or between interrupts, the program may continue to execute instructions. If no other processing is required, the main program can simply execute a JMP instruction to its own location (JMP \$) while waiting for the interrupt to occur.

If a JMP \$ instruction was being executed at the time an interrupt occurs and the user wishes to continue program execution from the point \$ + 1 after servicing the interrupt, the cell containing the program counter must be incremented by one via the service routine.

5-2.3.6 I/O Service Routine(s)

The I/O service routines generally follow the format:

1. Save accumulator and index register.
2. Test for end-of-transmission and disconnect. (Interrupt may be due to end-of-function from device controller, in which case the test may be omitted, the device controller disconnected, and the service data/function omitted.)
3. Service data/function.
4. Restore accumulator and index register.
5. Exit from I/O service routine via INR.

Higher priority (larger interrupt level number) devices may interrupt during the execution of lower priority service routines. The priority interrupt system can be inhibited by use of the MSK and UNM instructions.

5-3 DMA PROGRAMMING

The main programming difference between using a DMA device or a DIO device is that of word versus block transfer. A DIO device interrupts and provides a status indication when each data word is ready while a DMA device interrupts and provides a status indication when the block data transfer is complete. Both I/O systems use the DIO bus for control but only the DIO system uses the DIO bus for data transfer. The DMA system transfers blocks of data directly between memory and the device controller without the CPU having to issue a separate command for each data word transferred.

DMA device operation (Table 5-7) is initialized by executing a DOT DEV, 1 instruction with the starting memory address of the data transfer in the accumulator (ACR₁₋₁₅). If the DMA device is addressable (e.g., disc track and sector), issuing a DOT DEV, 2 instruction with the device address

residing in the accumulator, (ACR₀₋₅ track; ACR₇₋₁₅ Sector), initializes the beginning address for the device.

The block data transfer is started with the next DOT DEV, 4 instruction. The accumulator specifies the unit number in bits 00-01 and the number of words to be transferred (word count) in bits 02-15.

When the block data transfer is complete or an error condition exists (i.e., no write ring on magnetic tape or disc track protected and a write command issued), an interrupt is sent to the 706 CPU. After the CPU has been interrupted the I/O service routine usually interrogates the device(s) for status. Information about read errors controller and device ready, end-of-tape, are available from the status response.

Table 5-7. DMA Read/Write Selection Instruction Sequence

DISC										
FUNCTION	INSTRUCTION	ACCUMULATOR								
SEND MEMORY STARTING ADDRESS	DOT 1, 1	<table border="1"> <tr><td colspan="2">ADDRESS</td></tr> <tr><td>0</td><td>15</td></tr> </table>	ADDRESS		0	15				
ADDRESS										
0	15									
SEND TRACK AND SECTOR	DOT 1, 2	<table border="1"> <tr><td colspan="2">TRACK</td><td colspan="2">SECTOR</td></tr> <tr><td>0</td><td>5 6 7</td><td></td><td>15</td></tr> </table>	TRACK		SECTOR		0	5 6 7		15
TRACK		SECTOR								
0	5 6 7		15							
SEND WRITE COMMAND, UNIT NO. AND NUMBER OF WORDS IN TRANSFER	DOT 1, 4	<table border="1"> <tr><td colspan="2">WORD COUNT</td></tr> <tr><td>0</td><td>1 2</td></tr> <tr><td></td><td>15</td></tr> </table>	WORD COUNT		0	1 2		15		
WORD COUNT										
0	1 2									
	15									
SEND READ COMMAND, UNIT NO. AND NUMBER OF WORDS IN TRANSFER	DOT 1, 6	<table border="1"> <tr><td colspan="2">WORD COUNT</td></tr> <tr><td>0</td><td>1 2</td></tr> <tr><td></td><td>15</td></tr> </table>	WORD COUNT		0	1 2		15		
WORD COUNT										
0	1 2									
	15									

DMA MAGNETIC TAPE								
FUNCTION	INSTRUCTION	ACCUMULATOR						
SEND MEMORY STARTING ADDRESS	DOT 2, 1	<table border="1"> <tr><td colspan="2">ADDRESS</td></tr> <tr><td>0</td><td>15</td></tr> </table>	ADDRESS		0	15		
ADDRESS								
0	15							
SEND WRITE ONE RECORD, UNIT NO. AND WORD COUNT	DOT 2, 4	<table border="1"> <tr><td colspan="2">WORD COUNT</td></tr> <tr><td>0</td><td>1 2</td></tr> <tr><td></td><td>15</td></tr> </table>	WORD COUNT		0	1 2		15
WORD COUNT								
0	1 2							
	15							
SEND READ ONE RECORD, UNIT NO. AND WORD COUNT	DOT 2, 6	<table border="1"> <tr><td colspan="2">WORD COUNT</td></tr> <tr><td>0</td><td>1 2</td></tr> <tr><td></td><td>15</td></tr> </table>	WORD COUNT		0	1 2		15
WORD COUNT								
0	1 2							
	15							

5-4 INPUT/OUTPUT SOFTWARE (IOS)

The Raytheon 706 Input/Output Software (IOS) provides the user with a powerful and flexible file oriented input/output system. Features of the system include device reassignment, simultaneous I/O capability, centralized I/O monitor end-action, and I/O macros. IOS provides a common I/O system for all users and alleviates the user's need to fully understand the machine language I/O operations of the 706 computer (Sections 5-1, 5-2, 5-3).

IOS is able to perform I/O operations with the devices listed in Table 5-8. By using the 706 automatic priority interrupt system, IOS allows a rapid response to I/O interrupts. To further minimize time spent doing I/O, the software is file oriented, i.e., all information necessary to perform the I/O operation is contained in a File Input/Output Table. (FIOT)

Several I/O macros are available to IOS users. These macros reduce the effort required to program efficient input/output. The user need only become acquainted with the macros OPEN, DOIO, and STAT to communicate with any standard 706 device.

5-4.1 IOS Functions

Not all functions are available for every type of I/O device. For example, one cannot rewind the teletype. Refer to Table 5-8 for applicable functions.

IOS is an input/output control system which provides a user the facility to perform I/O operations which are effectively independent of the actual physical device being used. This is achieved by allowing different physical devices to be substituted for I/O operations without requiring re-assembly or recompilation of the program. For example, a program can be debugged using a small set of test data punched on cards and read from the card reader. After debugging, during production runs, the input can be changed to magnetic tape to process the actual data files.

A user performs I/O by requesting data transfers on logical units, which can be related to any physical device. In a typical I/O request to IOS, the user specifies a Logical Unit Number (LUN), the address of the I/O data buffer and the number of words to be transferred.

The logical unit is associated with a particular driver which performs a data transfer on a specific I/O device. The user can, at load time, cause this association to change so that a different driver and, therefore, a different physical device is referenced by the I/O request. This driver/device association is kept in a Peripheral Equipment Assignment Table (PEAT, see 5-4.2).

The device number or LUN assignment is made at system creation time. The IOS user can use the established LUNs and is free to reassign a given LUN to one of a set of devices, through X-RAY control directives. (Ref. Sec. 8 & 9)

All information pertaining to an I/O operation is kept in an 8-word File Input/Output Table, (FIOT, see 5-4.3). This table details the specific operation for the I/O request, as well as the LUN, I/O buffer, and the number of words. FIOT also provides temporary storage area as needed by the device driver.

5-4.1.1 IOS Calls

Macros are communication links between the programmer and IOS. All calls go through IOS subroutines to the appropriate device drivers. They translate information from an argument calling sequence to a FIOT, perform the I/O operation, provide error checking and end-action.

5-4.1.2 IOS Drivers

An I/O driver is a subroutine designed to perform data transfers between a specific peripheral device and the 706. An I/O driver is never directly used by a programmer, but only indirectly via IOS macros. Thus, for example, a programmer requests data to be read from some logical unit. IOS macros pass the FIOT address to the drivers, and the driver then extracts necessary information to start the I/O operation. The driver then exits back to the user program.

5-4.1.3 IOS Interrupt Service

Once initiated, I/O operations are processed independent of the user's program. Data transfers for devices on the Direct Memory Access (DMA) channel (such as disc and magnetic tape) are performed automatically by the hardware. The

interrupt service routine portion of each driver performs all data transfer operations for devices on the Direct Input/Output (DIO) channel (such as; paper tape reader, teletype, card reader, etc.). When the I/O operation is finished, the interrupt service routine for DIO and DMA devices perform all operations required to conclude the I/O operation. Device status and ending buffer address are stored in the FIOT. At this time an end-action subroutine may be executed, which permits any arbitrary operation to be performed as a part of the interrupt service function.

5-4.2 Peripheral Equipment Assignment Table (PEAT)

IOS processes input and output operations via logical equipment. Logical equipment is a functional classification of I/O devices; such as, symbolic input devices, listing devices, binary output devices, etc. Therefore, since program reference to I/O are by function, different devices may be assigned to a logical unit without requiring a compensating change in the program reference.

For example, the binary input device may be the high speed paper tape reader for one job and the teletype for the next, but neither the I/O monitor nor the users program need be reassembled. The device may simply be reassigned by an X-RAY directive. (Ref. Sec. 8 & 9)

The peripheral equipment assignment table contains the information about the correspondence between logical and physical units. IOS uses this table to select the appropriate drivers when input or output requests are made.

5-4.2.1 Logical Unit (LUN) Definitions

The PEAT shown in Table 5-9 contains more equipment than is included in a typical system, but it is needed to show how logical units function. The first 7 units are system logical units that are referred to by name in the remainder of this manual and in other manuals.

The definition of each of these names is listed below:

LUN	Description	
0	SYSF	SYStem File. This is the unit that contains the system library. In Table 5-9 SYSF points to the disc.
1	SYSI	SYStem Input. SYSI is the unit for all directives input to system programs; such as the X-RAY EXEC or SYM II Assembler. In Table 5-9 SYSI points to the teletype.
2	PRIN	PRImary INPUT. PRIN is the unit for input to system programs where the input is neither a directive nor binary; i.e., SYM II accepts the input source text from PRIN. In Table 5-9 PRIN points to the card reader.
3	LIST	LISTing. System programs use LIST for tabular output other than error messages. SYM II outputs the assembly listing on the LIST device. In Table 5-9 LIST points to the line printer.
4	BIN	Binary INput. BIN is the system binary input unit. In Table 5-9 BIN points to the card reader. Absolute or relocatable binary programs are input from this unit.
5	BOUT	Binary OUTput. BOUT is the system binary output unit. In Table 5-9 BOUT points to the card punch. SYM II outputs absolute or relocatable binary text to the BOUT unit.
6	SCR	SCRatch. System programs use SCR as temporary storage. SYM II uses SCR to store intermediate text. In Table 5-9 SCR points to the disc.

<p>7 LOGA, 8 LOGB, 9 LOGC, 10 LOGD, 11 LOGE</p>	<p>The next 5 units, LOGA-E, are intended for users. Devices are divided into two classes; mass and non-mass. A mass device is one that can perform both read and write operations; i.e., disc, magnetic tapes, etc. Non-mass devices can perform only a read or write; i.e., card reader, card punch, etc. LOGA is the systems highest level non-mass input device. LOGB is the systems highest level non-mass</p>	<p>output device. LOGC is the system mass unit. LOGD and E are assigned according to the assignment of other devices within the system. The user should look at the PEAT in his I/O monitor listing to determine how these units are assigned.</p>
	<p>12 DUMMY</p>	<p>This logical unit performs no I/O but allows program to be checked out without a peripheral device being available.</p>

Table 5-8. IOS Function Table

DEVICE	FUNCTIONS
Teletype Keyboard	Read, Write, Status
Teletype PTR	Read, Status
Teletype PTP	Write, Punch Leader, Status, Write EOF
High Speed PTR	Read, Status
High Speed PTP	Write, Punch Leader, Status
Card Reader	Read, Status
Card Punch	Write, Status, Write EOF, Punch Leader
Line Printer	Write, Status
Teletype Multiplexer	Read, Write, Status, Write EOF
Plotter	Write, Status
Disc	Read, Write, Verify, Status, Device Status
DMA Mag Tape	Read, Write, Rewind, Backspace, Write EOF, Search EOF, Write Skip, Status, Device Status
DIO Mag Tape	Read, Write, Rewind, Backspace, Write EOF, Write Skip, Status, Device Status

Figure 5-9. Card Reader (1000 CPM) and Card Punch Controls and Indicators

```

PEAT TABLE

3796 *          PEAT TABLE
3797 * THE PHYSICAL UNIT ASSIGNMENT TABLE IS GENERATED
3798 * UP TO THE HIGHEST NUMBERED UNIT PRESENT IN THE
3799 * SYSTEM
3800 * PEAT TABLE FORMAT IS AS FOLLOWS
3801 *          *****
3802 *          PETE(0) *P  INR* DEV* UN *      WORD 1
3803 *          *****
3804 *          PETE(1) * DRIVER ADDRESS *      WORD 2
3805 *          *****

7FD0 1200 3810 PEAT  BYTE  ASYSF,U          0  SYSF
7FD1 1300 3811  BYTE  ASYSI,U          1  SYSI
7FD2 1500 3812  BYTE  APRIN,U          2  PRIN
7FD3 1700 3813  BYTE  ALIST,U          3  LIST
7FD4 1500 3814  BYTE  ABIN,U           4  BIN
7FD5 1600 3815  BYTE  AROUT,U          5  ROUT
7FD6 1200 3816  BYTE  ASCR,U           6  SCR
7FD7 1500 3817  BYTE  ALOGA,U          7  LOG A
7FD8 1600 3818  BYTE  ALOGB,U          8  LOG B
7FD9 1900 3819  BYTE  ALOGC,U          9  LOG C
7FDA 1A00 3820  BYTE  ALOGD,U         10  LOG D
7FDB 0F00 3821  BYTE  ALOGE,U         11  LOG E
7FDC 4000 3822  BYTE  X'80',U         12  DUMMY PHYSICAL UNIT
7FDD 799C 3823  DATA M,DUM
          5638 3824 TTYU1 EQU TTYU+TTYU+TTYU+TTYU
          56E0 3825 TTYUX EQU TTYU1+TTYU1+TTYU1+TTYU1
7FE0 80E0 3826  BYTE  X'80'+TTYU1,TTYUX 13  TELETYPE
7FE1 7D4F 3827  DATA M,THSPT
          3828  FALS  NUNITS=0
          3829  TRUF  NMAG>0
          5008 3830 MAGU1 EQU MAGU+MAGU+MAGU+MAGU
          5020 3831 MAGUX EQU MAGU1+MAGU1+MAGU1+MAGU1
7FE2 4420 3832  BYTE  X'80'+MAGU1,MAGUX 14  MAG TAPE 0
7FE3 7C57 3833  DATA M,DMD
7FE4 5421 3834  BYTE  X'80'+MAGU1,MAGUX+1 15  MAG TAPE 1
7FE5 7C57 3835  DATA M,DMD
7FE6 5422 3836  BYTE  X'80'+MAGU1,MAGUX+2 16  MAG TAPE 2
7FE7 7D57 3837  DATA M,DMD
7FE8 5423 3838  BYTE  X'80'+MAGU1,MAGUX+3 17  MAG TAPE 3
7FE9 7C57 3839  DATA M,DMD
          3840  ENDC
          3851  FALS  NUNITS=NMAG=0
          3852  TRUF  NDSK>0
          5004 3853 DSKU1 EQU DSKU+DSKU+DSKU+DSKU
          5010 3854 DSKUX EQU DSKU1+DSKU1+DSKU1+DSKU1
7FEA 5310 3855  BYTE  X'80'+DSKU1,DSKUX 18  DISK 0
7FE9 7C57 3856  DATA M,DMD
          3857  ENDC
          3862  FALS  NUNITS=NMAG=NDSKU=0
          3863  FALS  NPCH=0
          5030 3864 PCHU1 EQU PCHU+PCHU+PCHU+PCHU
          50C0 3865 PCHUX EQU PCHU1+PCHU1+PCHU1+PCHU1
7FEA 80C0 3866  BYTE  X'80'+PCHU1,PCHUX 19  HIGH SPEED TAPE PUNCH
7FEB 7D4F 3867  DATA M,THSPT
          3868  ENDC
          3873  FALS  NUNITS=NMAG=NDSKU=NPCH=0

```

Figure 5-9. Card Reader (1000 CPM) and Card Punch Controls and Indicators (Cont.)

PFAT TABLE			
	3874	FALS	NPTR=0
0034	3875 PTRU1	EQU	PTRU+PTRU+PTRU+PTRU
00D0	3876 PTRUX	EQU	PTRU1+PTRU1+PTRU1+PTRU1
7FEC 80D0	3877	BYTE	X'80'+PTRI,PTRUX 20 HIGH SPEED TAPE READER
7FFD 7D4F	3878	DATA	M,THSPT
	3879	ENDC	
	3884	FALS	NUNITS=NMAG-NDSK0-NPCH-NPTR=0
	3885	FALS	NCDR=0
0020	3886 CDRU1	EQU	CDRU+CDRU+CDRU+CDRU
0080	3887 CDRUX	EQU	CDRU1+CDRU1+CDRU1+CDRU1
7FFE 8980	3888	BYTE	X'80'+CDRI,CDRUX 21 CARD READER
7FEF 7H4B	3889	D	CARDR
	3890	ENDC	
	3895	FALS	NUNITS=NMAG-NDSK0-NPTR-NCDR-NPCH=0
	3896	FALS	NCPC=0
003C	3897 CPCU1	EQU	CPCU+CPCU+CPCU+CPCU
00F0	3898 CPCUX	EQU	CPCU1+CPCU1+CPCU1+CPCU1
7FF0 88F0	3899	BYTE	X'80'+CPCI,CPCUX 22 CARD PUNCH
7FF1 7A33	3900	D	CARDP
	3901	ENDC	
	3906	FALS	NUNITS=NMAG-NDSK0-NPTR-NCDR-NCPC-NPCH=0
	3907	FALS	NLPR=0
0014	3908 LPRU1	EQU	LPRU+LPRU+LPRU+LPRU
0050	3909 LPRUX	EQU	LPRU1+LPRU1+LPRU1+LPRU1
7FF2 8250	3910	BYTE	X'80'+LPRI,LPRUX 23 LINE PRINTER
7FF3 79FF	3911	D	M,LPR
	3912	ENDC	
	3917	FALS	NUNITS=NMAG-NDSK0-NPTR-NCDR-NCPC-NPCH-NLPR=0
	3924	TRUE	NMXT=0
7FF4 8000	3925	D	X'8000' 24 MULTIPLEX CHANNEL ABSENT
7FF5 799C	3926	D	M,DUM
	3927	ENDC	
	3928	FALS	NUNITS=NMAG-NDSK0-NPTR-NCDR-NCPC-NPCH-NLPR-MUXCH=0
	3929	TRUE	NDSK>1
7FF6 8311	3930	BYTE	X'80'+DSK1,DSKUX+1 25 DISK 1
7FF7 7C57	3931	D	M,DMD
	3932	ENDC	
	3937	TRUE	NDSK>2
7FF8 8312	3938	BYTE	X'80'+DSK1,DSKUX+2 26 DISK 2
7FF9 7C57	3939	D	M,DMD
	3940	ENDC	
	3949	FALS	NDSK>3
7FFA 8000	3950	D	X'8000' 27 DISK 3 ABSENT
7FFB 799C	3951	D	M,DUM
	3952	ENDC	
	3977	ENDC	
	3978	ENDC	
	3979	ENDC	
	3980	ENDC	
	3981	ENDC	
	3982	ENDC	
	3983	ENDC	
	3984	ENDC	
	3985	ENDC	
	3986	ENDC	
7FFC	3987 ENDP	EQU	

The remainder of the logical units are assigned to the various physical devices contained in the system. The following are currently assigned:

LUN	Assignment
13	Console Teletype
14	Magnetic Tape Unit 0
15	Magnetic Tape Unit 1
16	Magnetic Tape Unit 2
17	Magnetic Tape Unit 3
18	Disc Unit 0
19	High Speed Paper Tape Punch
20	High Speed Paper Tape Reader
21	Card Reader
22	Card Punch
23	Line Printer
24	Multiplexer Teletype
25	Disc Unit 1
26	Disc Unit 2
27	Disc Unit 3
28	Digital Plotter

Physical assignments for all logical units are made when the monitor is assembled for each system. These assignments are determined by what units are physically present in the system. The user can determine what these initial (or standard) assignments are by referring to the PEAT table as it appears in the listing for his monitor. Following is an example of how the first 12 words of the PEAT might appear:

LOC	Contents	Source Text	Comments Field
3FD9	1200 3461	PEAT	BYTE ASYSF,0 0 SYSF
3FDA	0D00 3462		BYTE ASYSI,0 1 SYSI
3FDB	1500 3463		BYTE APRIN,0 2 PRIN
3FDC	1700 3464		BYTE ALIST,0 3 LIST
3FDD	1500 3465		BYTE ABIN,0 4 BIN
3FDE	1600 3466		BYTE ABOUT,0 5 BOUT
3FDF	1200 3467		BYTE ASCR,0 6 SCR
3FEO	1500 3468		BYTE ALOGA,0 7 LOG A
3FE1	1600 3469		BYTE ALOGB,0 8 LOG B
3FE2	1200 3470		BYTE ALOGC,0 9 LOG C
3FE3	0E00 3471		BYTE ALOGD,0 10 LOG D
3FE4	0F00 3472		BYTE ALOGE,0 11 LOG E

To determine the assignment of a particular unit, find the unit name (or number) in the comment field. The left byte of the contents field contains the hex number of the assignment. Convert it to decimal and look further down the table for that number in the comments field. For example, the device assigned to BIN (logical unit 4) has unit number 15₁₆. (15₁₆ is the left byte of the pointer word corresponding to BIN). 15₁₆ converts to decimal 21 which is the unit number of the card reader.

5-4.2.2 PEAT Table Description

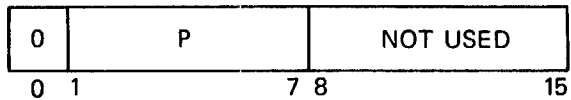
There are two types of entries in the PEAT. There are one word entries which define the correspondence between a logical unit and another logical unit, and there are two word entries which define the correspondence between a logical unit and a physical unit.

Entries within the PEAT are in logical unit number order; i.e., the first entry is for LUN 0, the second entry is for LUN 1, etc.

The table is in two sections. The first eleven entries, section 1, are always pointers to other LUNs. The remaining entries, section 2, are pointers to physical devices.

Numbers 0-11 may be considered logical unit numbers whereas units 12-31 may be considered physical unit numbers, since they are not normally reassigned.

In section 1 there is one word per LUN, formatted as follows:

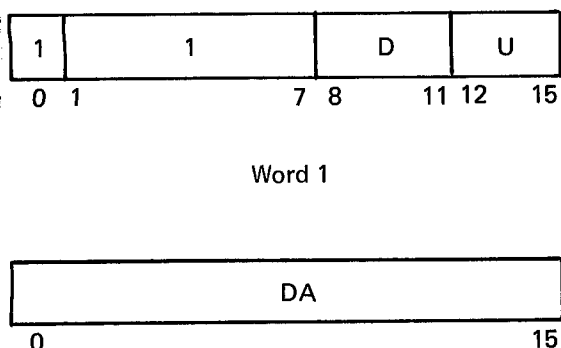


where:

The 0 in bit 0 denotes a pointer to another unit.

P is a 7-bit field pointing to the location, relative to the top of the PEAT table, for the entry of the next LUN.

In section 2 there are two words per entry. If the unit has been reassigned, word one will have the same format as section 1, and word two will be unused. If the LUN has not been reassigned, the format is as follows:



where:

The 1 denotes a pointer to a physical unit.

1 is a 7-bit field used to denote the priority interrupt level for the device.

D is a 4-bit field used to denote the device code.

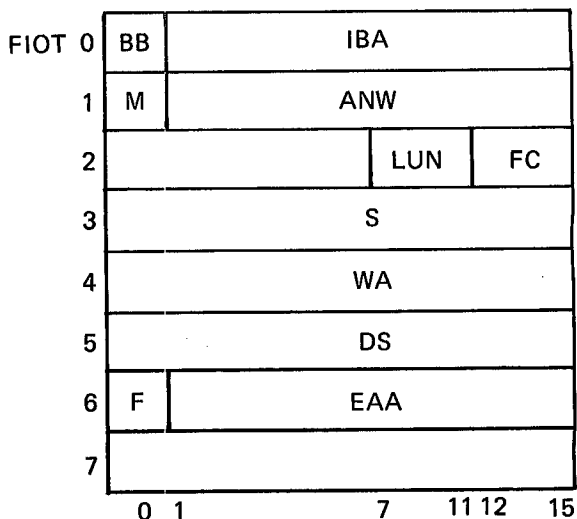
U is a 4-bit field used to denote the unit number of the device number; i.e., magnetic tape may be units 0, 1, 2, 3, etc.

DA is the device driver address for the LUN.

When IOS is seeking the physical unit corresponding to a logical unit, it will go from logical unit to a physical unit (a 1 in bit position 0).

5-4.3 File Input/Output Table (FIOT)

The purpose of the FIOT is to provide user area for I/O operation description and temporary storage for device drivers. How each driver uses the temporary storage is described in the individual driver documentation. Each FIOT is 8 words in length and formatted as follows:



Word 0:

The Busy Bit (BB) is a one-bit field which is set to 1 while the I/O operation is being performed. It will be reset to 0 when the operation has completed. The Initial Buffer Address (IBA) is a 15-bit field which contains the word address of the first location of the data buffer to be transferred.

Word 1:

Mode (M) is a one-bit field which has different meanings for various physical devices and is documented with each device driver (see Appendix K). ANW is a 15-bit field which is the Address of a location which contains the Number of Words to be input or output.

Word 2:

Bits 0 through 6 are temporary storage. The Logical Unit Number (LUN) is a 5-bit field contained in bits 7 through 11. The Function Code (FC) is a 4-bit field contained in bits 12 through 15, and defines the type of operation that is to be performed. Certain function codes perform special functions on specific devices; these are described in the applicable driver documentation. Function codes 9 and B₁₆ will perform a read operation for every device capable of a read. Function code E₁₆ will perform a write operation on every device capable of a write. Codes 9 and B₁₆ are equivalent for every device except the teletype. To preserve device independence codes 9, B, and E should be used for all read and write operations.

Word 3:

Status is the status response word obtained from the device upon completion of each I/O operation. The response differs from device to device and is documented separately for each device. Refer to the appropriate driver documentation. For all devices bit 15 true means that an error has occurred during the I/O operation.

Word 4:

Working Address. Upon completion of the I/O operation word 4 will contain the byte address one greater than the buffer address of the last byte of data transferred. For certain devices this word contains the address of the next byte to be transferred throughout the I/O operation and may be monitored to determine the progress of the operation. Refer to individual driver documentation for details.

Word 5:

Word 5 serves as temporary storage for most drivers. For the disc driver, this word contains the Sector Address to be read or written. When performing device independent I/O, where the disc is a legitimate assignment for the unit, word 5 should be set to the desired sector. Then, if some other device is assigned, the disc sector specification will be ignored. The format for this word as a disc sector is described in the disc driver documentation.

Word 6:

Bit 0 of word six is used for record Format control. When bit 0 is false, those drivers which generate or read formatted records will do so. When bit 0 is true, all drivers generate or read unformatted records. For unformatted records the data in the external medium will be an exact image of the internal buffer. For formatted records certain drivers expect or supply additional formatting data. In most discussions bit 0 is referenced as "the special format bit" and unformatted records are called "special format records". Details of record formatting are described in the applicable driver documentation.

Bits 1 to 15 of word 6 are the End-Action Address (EAA). This address is the address of an end-action subroutine to which the I/O system will transfer when a particular I/O is completed. If this address is 0, no end action is specified. For a discussion of end action, refer to section 5-4.6.

Word 7:

Word 7 is used by most drivers as temporary storage. It has a special use when the magnetic tape chain feature is present in the system, as described in the magnetic tape driver documentation.

5-4.4 Basic IOS Calls

IOS provides the user with 3 basic calls.

- a. OPEN
- b. DOIO
- c. STAT

With these three calls the user can perform an I/O operation on any standard 706 peripheral device.

In the discussion that follows the calling sequence for the various I/O subroutine, calls are stated in SYM II assembly language format.

CALL ARG1, ARG2 . . . ARGN

This will be assembled as though it were written:

```

SMB      CALL
JSX      CALL
DATA     ARG1
DATA     ARG2
. . . . .
DATA     ARGN + X'8000'
```

Note that the sign bit of the last argument is set. For more detail refer to the SYM II manual for an explanation of this form.

Entry points to the I/O subroutines are through fixed link points in the System Linkage Table located in cells X'40' to X'7F'. These points must be defined to the assembler using equate cards. Table 5-10 is a listing of the System Linkage Table.

For example, the programmer who writes

DOIO INFIOT

must include the definition

DOIO EQU X'44'

in this program. Programs written to execute on page 0 of the computer (location 80₁₆ through 7FF₁₆) may omit the SMB instruction by using the form:

JSX DOIO, INFIOT

Table 5-11 illustrates a sample input/output program which is written in the SYM I assembly language. The same program written in the SYM II assembly language is shown in Table 5-12.

5-4.4.1 OPEN

The first call for any I/O operation is to subroutine OPEN. OPEN translates the arguments to the proper format and inserts them into the FIOT table.

OPEN has a fixed length calling sequence as follows:

OPEN FIOT, BUF, WC, UNIT, OPER, MODE

where: FIOT is the first location of an 8-word array for the file description table.

BUF is the data buffer starting address.

WC is the address of the number of words.

UNIT is the logical unit number.

OPER is the type of operation (read, write, etc.).

MODE defines binary or alpha operation; 1= binary; 0= alpha.

It should be noted that the disc sector and the end-action address are not inserted into the FIOT by OPEN. These are less frequently used arguments and were eliminated to shorten the OPEN calling sequence. The user can assemble the arguments into the FIOT or store them at execution time.

The use of the OPEN call is optional, insofar as the FIOT may be assembled in the correct form. However, the OPEN call will initialize the FIOT, setting the busy bit off and clearing the status response word and may provide a convenient way of reinitializing. In addition, OPEN converts the specified logical unit into the actual physical unit number and stores it into bits 7-11 of word 2 of the FIOT, so that the using program may determine this assignment before beginning an I/O operation, if required.

5-4.4.2 DOIO

Actual I/O operations are initiated with calls to DOIO. DOIO changes the FIOT table as required, calls a driver to initiate the I/O operation, and returns to the user.

The calling sequence is as follows:

DOIO FIOT, BUF, WC

where: BUF is the data buffer starting address.

WC is the address of the number of words.

DOIO has a variable length calling sequence. If BUF or WC do not change from the previous call or have been set by OPEN, they need not be included. However, if WC is to be changed, BUF must be included.

5-4.4.3 STAT

STAT provides the user with a means of waiting until an I/O operation is complete. STAT has a variable length calling sequence as follows:

STAT FIOT, ERR

When a call to STAT is made, it will loop until the FIOT busy bit is off. If the ERR argument is not included, STAT will make a return to the next instruction following the calling sequence. If argument ERR is in the calling sequence, STAT will load the accumulator with the proper error processor return, and transfer to the ERR address. If the user wished to return from his ERR routine to the program sequence, he can do this by saving the accumulator as a return, then returning by loading the return into the index and executing a JMP * 2 (see sample program in Tables 5-11 and 5-12). If there were no errors, STAT returns to the next instruction following the ERR argument.

5-4.5 Special IOS Calls

The following calls perform special functions; such as Rewind, Write-End-of-File, and Backspace. Not all devices can perform the specified operation; for instance, the teletype will not rewind. If the device cannot perform the specified operation, the subroutine will return with the sign of the accumulator negative. The program can test this condition and determine an appropriate course of action.

Table 5-10 System Linkage Table

Cell (Base 16)	Cell Use
40-41	Linkage to Monitor Subroutine EXIT
42-43	Linkage to Monitor Subroutine OPEN
44-45	Linkage to Monitor Subroutine DOIO
46-47	Linkage to Monitor Subroutine STAT
48	Linkage to Monitor Subroutine BKSP
49	Not Used
4A	Linkage to Monitor Subroutine WSKP
4B	Not Used
4C	Linkage to Monitor Subroutine SEOF
4D	Not Used
4E-4F	Linkage to Monitor Subroutine RWND
50-51	Linkage to Monitor Subroutine WEOF
52-53	Linkage to Monitor Subroutine STUS
54	Upper Bound of Low End Resident Memory
55	Save Cell for Relo Program Start Address
56-57	Special Links for Trace Program
58	Peripheral Equipment Indicator Word (STYPE)
59	Address of PEAT Table
5A	Lower Limit of High End Resident Memory (ULIM)
5B	Lower Limit of Resident Monitor
5C	Address of Highest Location in Physical Core Memory
5D-60	Save Cells for DATE
61-64	Save Cells for ID Name
65	Cell Contains JMP * 0, Self Relocating Programs may Locate Themselves by doing a JSX X'65'
66	Lower Limit of Loaded Program, set by Loader
67	RTOS Systems Job Control Word
68	Upper Limit of COMMON Storage, Set by Loader
69-6D	Not Used
6A-6B	Linkage to Monitor Subroutine QUEUE
6C-6D	Linkage to Monitor Subroutine FETCH
6E-6F	Linkage to Monitor Subroutine TYPE
70-71	Linkage to Monitor System Flag Test Subroutine (FLAG)
72-73	Linkage to Monitor LOAD Subroutine
74	Points to DISC Vector
75	Upper Bound of Memory Available to Disc Loaders
76	DISC Sector Address of Processor Map
77	Queue Processor Control Word
78-79	Linkage to Monitor Absolute Load Routine
7A-7B	7A is the Relocatable Program Restart Location
7C	Resident Task String Point
7D-7E	Power Fail Save Restart Linkage
7F	Resident Entry Name Table Size

Table 5-11. SYM I Sample Program

```

1 *      TEST PROGRAM TO READ A RECORD FROM
2 *      THE BINT UNIT AND TYPE OUT DONE,
3      ORIG 2048
0009    4 READ EQU 9
000E    5 WRIT EQU X'F'
0001    6 BINT EQU 1
0001    7 BIN EQU 1
0000    8 ALPH EQU 0
0003    9 LIST EQU 3
0040   10 EXIT EQU 64
0042   11 OPEN EQU 66
0044   12 DDIO EQU 68
0046   13 STAT EQU 70
0800 0080 14 STRT SMR OPEN
0801 2042 15      JSX OPEN,RFIT,RBUF,RWC,BINT,READ,BIN
0802 882B
0803 8821
0804 8845
0805 0001
0806 0009
0807 8001
0808 0080 16      SMR OPEN
0809 2042 17      JSX OPEN,TFIT,RBUF,RWC,LIST,WRIT,ALPH
080A 883B
080B 8821
080C 8845
080D 0003
080E 000E
080F 8000
0810 0080 18      SMR DDIO
0811 2044 19      JSX DDIO,RFIT
0812 882B
0813 0080 20      SMR STAT
0814 2046 21      JSX STAT,RFIT,RERR
0815 882B
0816 8834
0817 0080 22      SMR DDIO
0818 2044 23      JSX DDIO,TFIT,FMES,TWO
0819 883B
081A 8843
081B 8833
081C 0080 24      SMR STAT
081D 2046 25      JSX STAT,TFIT
081E 883B
081F 0080 26      SMR EXIT
0820 2040 27      JSX EXIT
0821      28 RBUF RES 10
082B      29 RFIT RES 8
0833 0002 30 TWO D 2
0834 703A 31 RERR STW RER
0835 0080 32      SMR DDIO,TFIT
0836 2044 33      JSX DDIO,TFIT
0837 883B
0838 903A 34      LDX RER
0839 1802 35      JMP * 2
083A 0000 36 RER DATA 0
083B      37 TFIT RES 8
0843 C4CF 38 FMES D 'DDIO','NEI'
0844 GEC5
0845 000A 39 RWC DATA 10
40      END

```

Table 5-12. SYM II Sample Program

```

1 *      TEST PROGRAM TO READ A RECORD FROM
2 *      THE BIN UNIT AND TYPE OUT DONE.
3        ORIG 2048
4 READ   EQU 9
5 WRIT   EQU X'E'
6 HINT   EQU 1
7 RIN    EQU 1
8 ALPH   EQU 0
9 LIST   EQU 3
10 DONE  EQU 64
11 OPEN  EQU 66
12 D010  EQU 68
13 STAT  EQU 70
14 STRT  OPEN  RFIT,RBUF,RWC,BINT,READ,BIN

0800 0080
$ 0801 2042
0802 0828
0803 0821
0804 0844
0805 0001
0806 0009
0807 8001
0808 0080 15      OPEN  TFIT,RBUF,RWC,LIST,WRIT,ALPH
$ 0809 2042
080A 083A
080B 0821
080C 0844
080D 0003
080E 000E
080F 8000
0810 0080 16      D010  RFIT
$ 0811 2044
0812 8828
0813 0080 17      STAT  RFIT,RERR
$ 0814 2046
0815 0828
0816 8833
0817 0080 18      D010  TFIT,FMES,#?
$ 0818 2044
0819 083A
081A 0842
081B 8845
081C 0080 19      STAT  TFIT
$ 081D 2046
081E 883A
081F 0080 20      DONE
$ 0820 2040
0821
082B 21 RBUF  RES 10
082B 22 RFIT  RFS 8
0833 7039 23 RERR  STW RER
0834 0080 24      D010  TFIT
$ 0835 2044
0836 883A
0837 9039 25      LDX  RER
0838 1802 26      JMP  * 2
0839 0000 27 RER   0 0
083A      28 TFIT  RFS 8
0842 C4CF 29 FMES  TEXT 'DONE'
0843 CEC5
0844 000A 30 RWC   D 10
0800      31      END  STRT
0845 0002

```

For example, the Loader accepts input from the IIN device (logical unit 4). If an input error occurs, the Loader backspaces the unit. If the unit was a mag tape, it backspaces and the Loader re-reads the record. If the unit is not mag tape and cannot backspace, the Loader requests operator intervention to reposition the record.

The disc unit cannot perform the special functions; however, special calls for the disc return with the accumulator positive, since such functions as backspace and write end-of-file may be simulated by adjusting the sector address in the FIOT and adjusting the disc logical file pointers, which may be done arbitrarily and then allowing the program to drop through the special calls. Refer to the disc driver documentation.

The only information which must be in the FIOT for special operation is the logical unit number (bits 7-11 of word 2).

5-4.5.1 Device Status

The physical status of any I/O device may be interrogated at any time. The call is

STUS FIOT

The device's physical status response is returned in the accumulator. The FIOT is not disturbed by this operation, except to substitute the actual physical device number for the logical unit, if this has not already been done. An I/O operation in progress will not be disturbed.

5-4.5.2 Write End-of-File

A file mark is defined for most I/O devices and may be written on all output devices except the disc, line printer, and plotter. For magnetic tape the file mark is hardware supplied. For paper tape an ASCII "Bell" (X'87') character is the file mark. For cards a 2-7-8-9 punch is the file mark. All input devices upon sensing a file mark conclude the I/O operation and return End-of-File status (bit 10 of the status word true). The write end-of-file call is

WEOF FIOT

The FIOT will be busy until the operation is complete.

5-4.5.3 Seek End-of-File

A file mark is sought on the specified unit. Only the magnetic tape can perform this function. Refer to the magnetic tape driver documentation. The call is

SEOF FIOT

5-4.5.4 Backspace

The specified unit is backspaced one record. Only the magnetic tape can perform this operation. The call is

BKSP FIOT

5-4.5.5 Write-Skip

The specified magnetic tape unit will erase forward three inches of tape. The call is

WSKP FIOT

5-4.5.6 Rewind

The specified magnetic tape unit is rewound. The call is

REWD FIOT

5-4.6 End Action

The End Action feature permits program operation to be synchronized to I/O device operation. It allows sharing of time, otherwise wasted during I/O operations, with other tasks. A core resident program can initiate an I/O operation and then exit to allow the system to initiate another task. Then, when the I/O operation is complete, the system will return to the resident program at the end-action address. It can start another I/O operation and then exit. The system will restore the interrupted task.

End action may also be used within a single task; for example, to operate a double buffered I/O function.

End action is requested by specifying a non-zero service address in word 6 of the FIOT. Upon completion of the I/O operation, the I/O system will transfer to that address, after servicing but before restoring the final interrupt of the I/O operation. Then, when the end action subroutine

exits, the system automatically restores the interrupted task.

End action subroutines are largely arbitrary in form and function. Any legal operation, including I/O, may be performed as part of an end-action subroutine. The system transfers to the end action address with the return on the index, so that the end action service routine may be in the form of a subroutine:

```

STX  RETSAVE
.
.
.
.
LDX  RETSAVE
JSX * 0

```

Of course, the SUBR and EXIT pseudo operations of SYM II may be used.

The end action service routine may also simply exit through the system EXIT link, cell X'40'. The system intercepts this exit and restores the interrupt. Thus, the service routine may have the form of a main program.

There are two restrictions on an end action service routine which affects the initiation of new I/O

operations. When the new I/O operation is on the same device or on a device with the same interrupt level, as the device causing the end action, DOIO does not return after initiating the I/O, but rather exits the end action. Practically, this means that the last thing which should be done in an end action service routine is to initiate new I/O for the same device. It is emphasized that this restriction does not apply to I/O for device on other interrupt levels, which return normally from the DOIO call.

The second restriction applies when the end action service routine is interrupting the interrupt service routine for the device for which the new I/O request is being made. When this occurs, the I/O in progress on that device cannot complete and therefore, the new I/O operation cannot be initiated. In this event DOIO does not set the FIO busy bit, but returns a status response of X'00FF in word 3 of the FIOT. A STAT call with an error return will take the error return for this condition. This condition is virtually unrecoverable; however it can occur normally only when two users are simultaneously attempting to use the same device. Cautious programming should avoid this situation. A possible solution to this situation, should it occur, would be to initiate a dummy I/O operation on an available device for example, the console typewriter whose end action would attempt to initiate the I/O on the unavailable device.

5-4.7 Device Driver Reference

I/O Code	Title	Drawing No.
BPU	Disc and Magnetic Tape Driver	391040
BSU	Card Punch Driver	390018
BSV	Card Reader Driver	390019
BSW	Line Printer Driver	390020
BUU	Teletype Multiplexer Driver	391909
BYM	Teletype - High Speed Paper Tape I/O Driver	392292
BYP	Plotter Driver	392295
See Section 8-6.		

5-5 TELETYPE ASR-33

The ASR-33 provides 8-channel paper tape input/output, keyboard input and printer output for the Raytheon 706 computer.

5-5.1 Speed

Characters are read, printed or punched at 10 cps.

5-5.2 Data Type

Data are transferred to and from paper tape or the keyboard/printer in one of two 8-level forms, ASCII or ASCII-8 (American Standard Code for Information Interchange). Raytheon 706 Programming Systems utilize only the ASCII mode. Table 5-13 depicts the codes of the 706 ASR-33 in hexadecimal notation.

5-5.3 Data Format

The 8-bit data are transferred between the ASR and bits 8-15 of the accumulator. The most significant 8 bits (0-7) of the accumulator are unaffected by an output; however, they are replaced by zeros upon each input. Data transferred to and from the keyboard/printer are in the form of characters illustrated in Table 5-13. The data format of paper tape is depicted in Figure 5-5 where "T" Channel Number represents the frame timing (sprocket). Paper tape characters are shown in Table 5-13.

5-5.4 Modes

The ASR-33 teletype functions in either of two basic modes, Local or Line, as selected by the position of its manual local/line switch. Local mode causes the ASR to be off-line and no data can be transferred between the teletype and the 706 processor. The device may then be used as a local tape perforator in the preparation and editing of programs and data. Line mode allows the processor to select the device on-line permitting data flow between the teletype and the 706. Once the ASR is on-line and the manual paper tape control switches are in the proper state, any of the following operations may be performed under program control.

5-5.4.1 Disconnect (DOT E, 0)

The ASR is disconnected and put into an off-line state. Its interrupt capability is inhibited. The operator may use the teletype as though it is in Local mode.

5-5.4.2 Paper Tape Read (DOT E, 1; DOT E, 9; DIN E, 5; DIN E, D)

This operation is primarily intended for reading paper tape; however, data are also accepted from the keyboard. The paper tape reader must be manually turned ON. Erroneous data will result if a key is depressed while reading paper tape.

5-5.4.3 Print/Punch (DOT E, 2; DOT E, A; DOT E, 6; DOT E, E)

Data are transferred to the printer. Paper tape is punched if the paper tape punch has been manually turned ON. Printing can be manually inhibited in this operation on the ASR-35 only.

5-5.4.4 Keyboard (DOT E, 3; DOT E, B; DIN E, 7; DIN E, F)

This operation is primarily intended for data input from the keyboard; however, data will also be accepted from the paper tape reader if it is manually turned ON.

5-5.5 Function Codes

The various 8-bit control codes are illustrated in hexadecimal and binary formats in Table 5-14. The ASR is put on-line by an initial select whereupon it will immediately interrupt to request service. Thereafter a service routine can transfer the data and execute I/O housekeeping. Each character request (interrupt) requires a previous select. Therefore, continuous data flow is accomplished by executing input/output instructions (DIN or DOT) which simultaneously specify Data Transfer and select.

Whenever only one character is to be transferred, there is no need for an initial select. The character may be transferred by a combined select and transfer command. No interrupt will follow since the data was transferred along with the single select.

5-5.6 Status Codes

Status is returned to the accumulator from the teletype after a DIN E, 0 instruction is issued. The status word consists of two bits (see Table 5-15). Bit 0 true means that a select command has been

Table 5-13. ASR-33 ASCII and ASCII-8 Character Codes

Character	ASCII	ASCII-8	Character	ASCII	ASCII-8
BELL	87	07	?	BF	5F
LF	8A	0A	@	C0	A0
CR	8D	0D	A	C1	A1
BLANK	A0	40	B	C2	A2
!	A1	41	C	C3	A3
"	A2	42	D	C4	A4
#	A3	43	E	C5	A5
\$	A4	44	F	C6	A6
%	A5	45	G	C7	A7
&	A6	46	H	C8	A8
'	A7	47	I	C9	A9
(A8	48	J	CA	AA
)	A9	49	K	CB	AB
*	AA	4A	L	CC	AC
+	AB	4B	M	CD	AD
,	AC	4C	N	CE	AE
-	AD	4D	O	CF	AF
.	AE	4E	P	D0	B0
/	AF	4F	Q	D1	B1
0	BO	50	R	D2	B2
1	B1	51	S	D3	B3
2	B2	52	T	D4	B4
3	B3	53	U	D5	B5
4	B4	54	V	D6	B6
5	B5	55	W	D7	B7
6	B6	56	X	D8	B8
7	B7	57	Y	D9	B9
8	B8	58	Z	DA	BA
9	B9	59	[DB	BB
:	BA	5A	\	DC	BC
;	BB	5B]	DD	BD
<	BC	5C	↑	DE	BE
=	BD	5D	←	DF	BF
>	BE	5E			

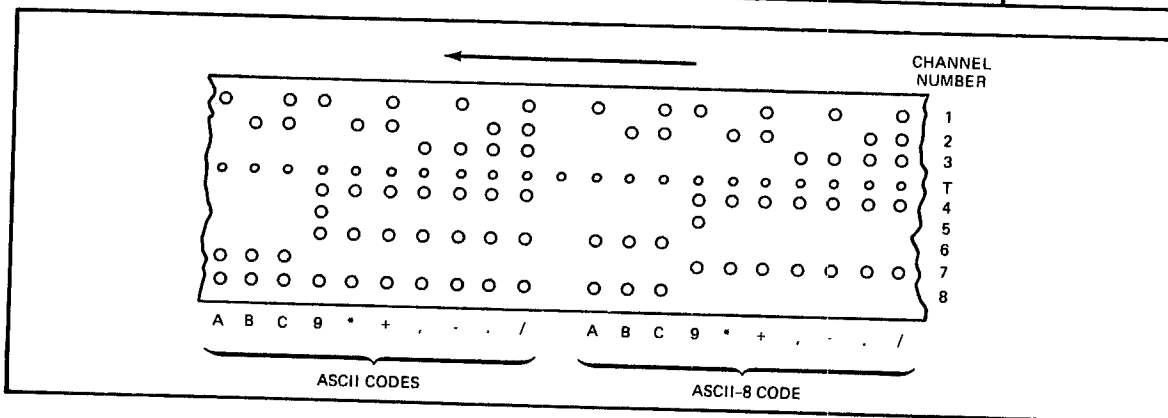


Figure 5-5. Eight Channel Paper Tape Data Format

Table 5-14. ASR-33 Commands and Function Codes

Command	Function Codes								
	Hexadecimal Notation	Binary Notation							
		8	9	10	11	12	13	14	15
DOT									
Disconnect Controller	E0	1	1	1	0	0	0	0	0
Select PTR (ASCII-8)	E1	1	1	1	0	0	0	0	1
Select Printer/Punch (ASCII-8)	E2	1	1	1	0	0	0	1	0
Select Keyboard (ASCII-8)	E3	1	1	1	0	0	0	1	1
Transfer Data Select Printer/Punch (ASCII-8)	E6	1	1	1	0	0	1	1	0
Disconnect Controller	E8	1	1	1	0	1	0	0	0
Select PTR (ASCII)	E9	1	1	1	0	1	0	0	1
Select Printer/Punch (ASCII)	EA	1	1	1	0	1	0	1	0
Select Keyboard (ASCII)	EB	1	1	1	0	1	0	1	1
Transfer Data Select Printer/Punch (ASCII)	EE	1	1	1	0	1	1	1	0
DIN									
Return Status to Accumulator	E0	1	1	1	0	0	0	0	0
Transfer Data Select PTR (ASCII-8)	E5	1	1	1	0	0	1	0	1
Transfer Data Select Keyboard (ASCII-8)	E7	1	1	1	0	0	1	1	1
Transfer Data & Select PTR (ASCII)	ED	1	1	1	0	1	1	0	1
Transfer Data & Select Keyboard (ASCII)	EF	1	1	1	0	1	1	1	1

issued to the teletype and bit-7 true means that the interrupt is true and has not been serviced by the transmission of a character.

TABLE 5-15 ASR-33 Status

Meaning When Bit True	Bit No.
Teletype has been selected.	0
Interrupt true and has not been serviced.	7

5-5.7 Interrupt Meaning

An interrupt caused from the selection of the printer/punch means that the teletype controller's character buffer is ready to receive a new character. An interrupt caused from the selection of the paper tape reader or keyboard means that the controller's character buffer is full and ready to transmit the new character to the CPU.

5-5.8 Timing

When selecting the printer/punch a character must be transmitted before the first interrupt can occur. The first interrupt and all succeeding interrupts occur at intervals of 100 milliseconds. In order to

keep the teletype options operating at the maximum rate of 10 characters/second, the interrupt must be serviced within 17 milliseconds. See Figure 5-6.

5-5.9 Operation Notes

Before using the teletype on-line with the computer, turn the LINE-OFF-LOCAL switch to LINE. The paper tape punch must be turned on manually prior to its selection. There will be no paper tape movement until a Transfer Data command has been issued.

The paper tape reader must *not* be turned on before it is selected. After the Select command has been issued, turn the START-STOP-FREE lever to START. Thereafter the reader interrupts the processor whenever data is ready to be transferred to the accumulator.

Whenever a punching a reading operation has been completed, turn the device off to prevent erroneous data from being transferred.

Two indicator lights, Keyboard (KBD) and Paper Tape Reader (PTR), on the console of the ASR signify selected operation. The four operational states are given in Table 5-15.

Table 5-15. Teletype Console Operation

Keyboard (KBD) Light	Paper Tape Reader (PTR) Light	Operation
OFF	OFF	Local Mode
OFF	ON	Paper Tape Read
OFF	OFF	Print/Punch
ON	OFF	Keyboard

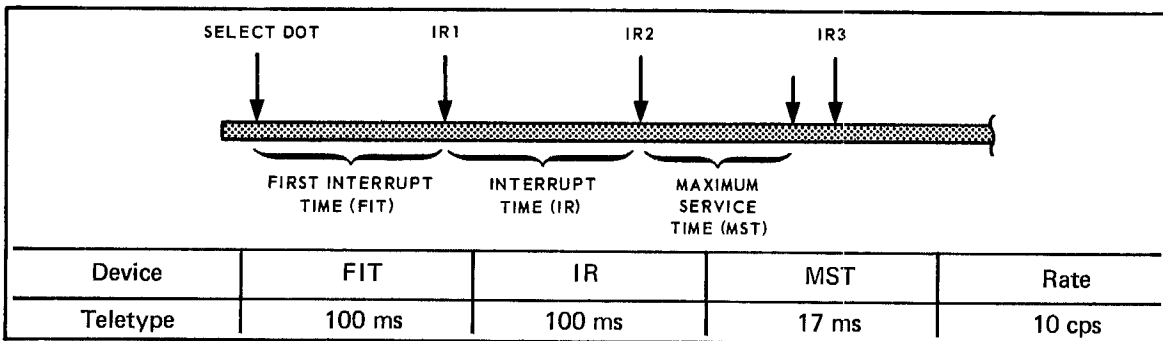


Figure 5-6. Teletype Interrupt Timing

5-6 HIGH SPEED PAPER TAPE READER AND PUNCH

The high speed paper tape reader and punch provides 8-channel paper tape input/output for the IBM 606 computer. The photoelectric reader is a bidirectional unit augmented by an optional paper tape spooler.

5-6.1 Speed

Characters are read at 300 cps, and characters are punched at 110 cps.

5-6.2 Data Type

All characters either read or punched are in binary form.

5-6.3 Data Format

The 8-bit data are transferred between the high speed reader/punch controller and bits 8-15 of the accumulator. The most significant 8 bits (0-7) of the accumulator are unaffected by an output; however they are replaced by zeros upon each input. The data format for paper tape is depicted in Figure 5-5 (ASCII codes), where "T" channel number represents the frame timing sprocket. All paper tape characters are shown in Table 5-13.

5-6.4 Reader Functions

Although the high speed paper tape reader and high speed paper tape punch share the same controller, they function as two separate devices and may be selected for concurrent reading and punching.

5-6.4.1 Disconnect (DOT D, 0)

The high speed paper tape reader is disconnected and put into an off-line state. Its interrupt capability is inhibited. Status bits 0 and 1 become true.

5-6.4.2 Read Forward (DOT D, 1)

The read forward command initiates tape motion in the forward direction. When the first character is read, an interrupt is sent to the CPU and status bit-7 becomes true. If a character is over the photocell block when the operation is started, no interrupt is sent until the next character is read.

5-6.4.3 Transfer Character and Read Forward (DOT D, 5)

After the interrupt is received or status bit-7 has been tested and found true, the transfer character and read forward command inputs the character into the accumulator (bits 8-15), turns off the interrupt and status bit-7, and causes tape motion to continue forward. This instruction must be issued within 250 μ seconds after the interrupt to keep the tape moving continuously. If the character is not transferred within this time, the tape stops "on character" and waits until the character is transferred into the CPU.

5-6.4.4 Transfer Character and Disconnect (DOT D, 4)

The transfer character and disconnect is normally used to input the last character to be read from a tape. The character is transferred, motion stops, no more interrupts are allowed, and status bits 0 and 1 are set true.

5-6.5 Punch Functions

5-6.5.1 Disconnect (DOT C, 0)

The high speed paper tape punch is disconnected and put into an off-line state. Its interrupt capability is inhibited. Status bit 0 becomes true. Punch motor power is not turned off by this function.

5-6.5.2 Turn on Punch Power (DOT C, 2)

Power to the punch motor is not turned on when the computer power is turned on. Punch power must be turned on by a separate power on command prior to transferring the first character to the controller or at the same time as transferring the first character to the controller. A 1-second delay is initiated when punch power is turned on before status bit-1 is allowed to become true. No interrupt is returned to the CPU because punch power has been turned on.

5-6.5.3 Output Character and Turn On Punch Power (DOT C, 6)

The punch power may be turned on with the same instruction used to output a character. If the punch has not been turned on previously, there is a 1-second power on delay before the character is

punched and the interrupt sent to the CPU. The maximum service time to keep the punch operating at full speed is 4 milliseconds.

5-6.5.4 Turn Off Punch Power (DOT C, 3)

Punch power is turned off by a separate instruction. Disconnecting the punch does not turn off punch power. When the punch power is turned off status bit-1 becomes true.

5-6.6 Function Codes

The various 8 bit control codes are illustrated in hexadecimal and binary formats in Table 5-16.

When reading, the high speed paper tape reader is put on-line by an initial read forward. Upon reading the first character into the controller buffer, the high speed reader controller interrupts the CPU to request service. Thereafter, a service routine can transfer the data and execute I/O housekeeping. Each character request (interrupt) requires a previous read. Therefore, continuous data flow is accomplished by executing input instructions (DIN) which simultaneously specify data transfer and read.

When punching, the high speed paper tape punch is put on-line and the punch motor power turned on by executing either a Turn Punch Power On or Output Character and Turn Punch Power On. Each character request (interrupt) requires a previous Turn Punch Power On. Therefore, continuous data flow is accomplished by executing output instructions (DOT) which simultaneously specify Output Characters and Turn Punch Power On. After the last character of a series is transferred, the punch power may be turned off by issuing a Turn Punch Power Off command. The interrupt may be inhibited by executing a disconnect.

5-6.7 Status Codes

Status is returned to the accumulator from the high speed paper tape reader or punch after a DIN D, 0 or a DIN C, 0 instruction is issued, respectively.

The high speed paper tape reader status word consists of three bits (Table 5-17). Bit-0 true means

that the reader controller has not been selected, bit-1 true means that the reader is not ready to transfer data, and bit-7 true means that an interrupt from the reader has not been serviced.

The high speed paper tape punch status word consists of four bits (Table 5-17). Bit-0 true means that the punch controller has not been selected, bit-1 true means that the punch motor is not up to speed, bit 4 true means that the tape supply is low and bit-7 true means that the interrupt from the punch has not been serviced.

5-6.8 Interrupt Meaning

The interrupt from the High Speed Paper Tape Reader means that a character has been read into the reader buffer and is ready to be transferred to the CPU.

The interrupt from the High Speed Paper Tape Punch means that the previous character sent is being punched and that a new character may be sent from the CPU.

5-6.9 Timing

Figure 5-7 gives the interrupt timing for the High Speed Paper Tape Reader and Punch.

After selecting the HSPT Reader, the first interrupt will occur within 3.3 milliseconds. All other interrupts will occur at a 3.3 millisecond rate provided that each interrupt is serviced within 250 microseconds after being sent.

The High Speed Paper Tape Punch will send the first interrupt only after the first data word is transmitted. Interrupts will occur thereafter at a 9.1 millisecond rate provided that each interrupt is serviced within 4 milliseconds after being sent.

5-6.10 Operation Notes

5-6.10.1 Tape Spooler (Model No. 75603)

The tape spooler operates in conjunction with the tape reader and responds to tape motion through the reader. Forward or reverse rewind is controlled by a switch on the front panel of the spooler. No program control of the spooler is provided.

Table 5-16. High Speed Paper Tape Reader and Punch Commands and Function Codes

High Speed Paper Tape Reader Commands	Function Codes							
	Hexadecimal Notation	Binary Notation						
		8	9	10	11	12	13	14
DOT								
Disconnect Controller Read Forward	D0	1	1	0	1	0	0	0
	D1	1	1	0	1	0	0	1
DIN								
Return Status to Accumulator Transfer Character & Disconnect Transfer Character & Read Forward	D0	1	1	0	1	0	0	0
	D4	1	1	0	1	0	1	0
	D5	1	1	0	1	0	1	1

High Speed Paper Tape Punch Commands	Function Codes							
	Hexadecimal Notation	Binary Notation						
		8	9	10	11	12	13	14
DOT								
Disconnect Turn On Punch Power Turn Off Punch Power Output Character & Turn On Punch Power	C0	1	1	0	0	0	0	0
	C2	1	1	0	0	0	0	1
	C3	1	1	0	0	0	0	1
	C6	1	1	0	0	0	1	1
DIN								
Return Status to Accumulator	C0	1	1	0	0	0	0	0

Table 5-17. Status Response from High Speed Paper Tape Reader and Punch.

HSPT Reader Status	
Meaning When Bit True	Bit No.
Controller Not Ready	0
Device Not Ready	1
Interrupt has not been serviced	7
HSPT Punch Status	
Controller Not Ready	0
Device Not Ready – Punch Not up to speed	1
Paper Tape Supply Low	4
Interrupt has not been serviced	7

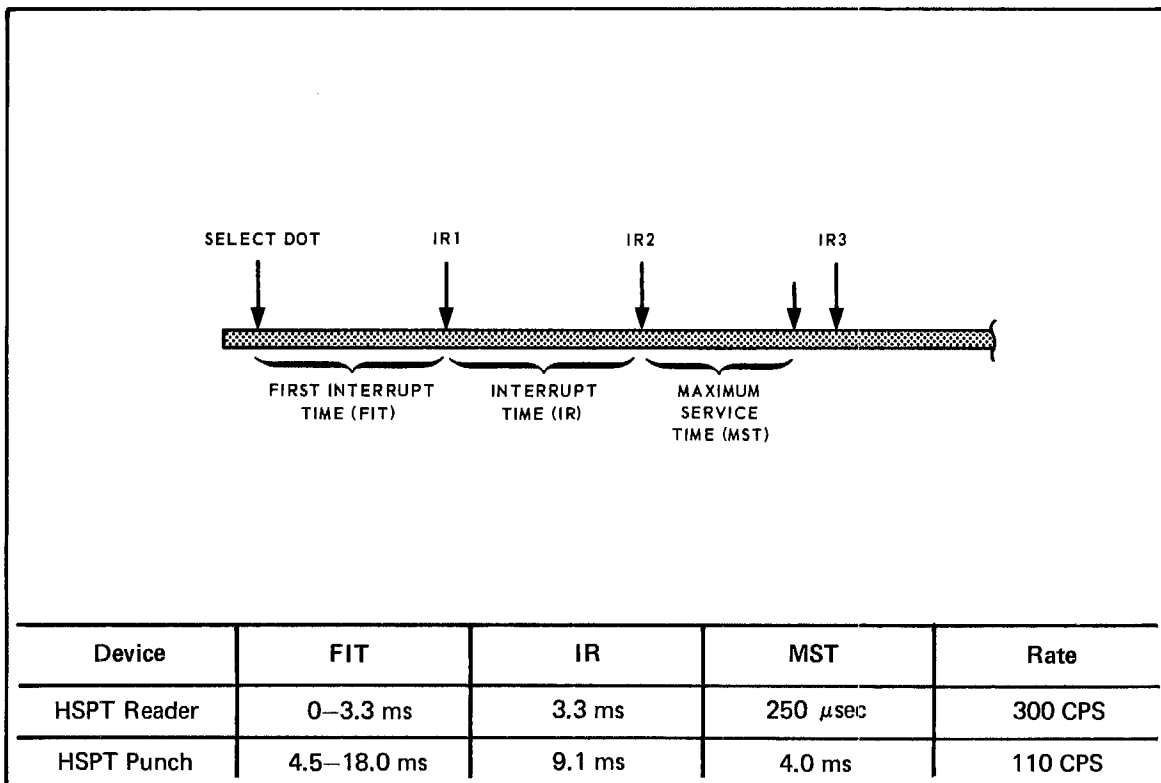


Figure 5-7. High Speed Paper Tape Reader and Punch Interrupt Timing

5-6.10.2 Punching Leader

Leader (sprocket holes only) can be punched manually by pressing the TAPE FEED switch on the front panel of the punch. Leader can be punched under program control by issuing Output Character commands with all zeros in accumulator bits 8-15. There are 10 sprocket holes per inch.

5-6.10.3 Reading Leader

The first characters transferred from the reader will normally be leader characters (sprocket hole only). These characters are not normally stored in memory and an SAZ instruction is used to detect and reject them. When the first non-zero character is read, the SAZ test instruction should be changed to a store byte (STB) or store word (STW) instruction as required by the application. The disposition of the first non-zero character and subsequent characters is determined by the specific tape format used in the application.

5-6.11 High Speed Paper Tape Reader (Model No. 75601) and Punch (Model No. 75602) Operation

5-6.11.1 Controls and Indicators

The 706 Tape Reader has two switches on its front panel; POWER ON and RUN-LOAD. The POWER ON switch may be left on so that power is supplied to the reader whenever computer power is on, or it may be turned on only when this unit is to be

used. The RUN-LOAD switch must be in the LOAD position to manually insert or withdraw paper tape and it must be in the RUN position before tape can be read under program control.

The 706 Tape Punch, which contains the punch set, has one operating control: a TAPE FEED switch on the front panel. When pressed, this switch causes the punch set to feed out tape with sprocket holes punched but otherwise blank. This operation continues until the switch is released.

No operating controls or indicators are provided on the paper tape controller.

5-6.11.2 Loading

Refer to the OEM manuals supplied with the reader, spooler and punch for the proper loading procedure. When loading the reader, the sprocket holes go towards the cabinet.

5-6.11.3 Operation

The paper tape reader requires that the RUN-LOAD switch be in the RUN position before tape can be read. Tape motion is initiated via program control.

The paper tape punch is turned on and off via program control.

5-7. CARD READER AND PUNCH

The card reader and card punch provide column punched card input/output for the Raytheon 706 computer.

5-7.1 Speed

Cards are read at 1100 or 400 cpm; cards are punched up to 400 cpm.

5-7.2 Data Type

Data are transferred to and from the card reader/card punch in binary form.

5-7.3 Data Format

5-7.3.1 Read Data Format

The 12-bit data are transferred between the card reader and bits 4-15 of the accumulator. The most significant bits (0-3) of the accumulator are set to zero. Table 5-18 shows the correspondence between the card rows and the accumulator.

5-7.3.2 Punch Data Format

The 12-bit data are transferred between bits 4-15 of the accumulator and rows 12-9 of the card punch. Two 12-bit data words (columns) are punched at a time. The most significant bit (0-3) of the accumulator are not punched. Table 5-18 shows the correspondence between the card rows and the accumulator.

5-7.4 Card Reader Functions

5-7.4.1 Disconnect (DOT 8,0)

The card reader is disconnected and put into an off-line state. Its interrupt capability is inhibited and all data transfer comes to an immediate stop. Status bit-0 becomes true.

5-7.4.2 Disconnect After Current Card is Read (DOT 8, 3)

Same as Disconnect except that the interrupt capability is not inhibited until the end of card has been reached.

5-7.4.3 Select Card Reader (DOT 8, 1)

Card reading is initiated and an interrupt is issued when data are available. The interrupt is terminated when the data are transferred to the accumulator (bits 4 through 15). Card reading continues automatically until there are no more cards in the stacker or until a Disconnect is issued.

Table 5-18. Card Row-Accumulator Bit Correspondence

Card Row	Accumulator Bit
No correspondence	0-3
12	4
11	5
0	6
1	7
2	8
3	9
4	10
5	11
6	12
7	13
8	14
9	15

5-7.4.4 Stop Interrupts for Current Card (DOT 8, 2)

Interrupts are inhibited for the remainder of the card and are resumed on the following card.

5-7.4.5 Offset Card in Process in Output Stacker (DOT 8, 8)

The card in process is offset stacked in the output hopper (card stacked with left edge offset approximately one inch). This operation can be combined with any other instruction, i.e. a DOT 8, A (8 + 2) would stop interrupts for the card in process while offset stacking that card in the hopper.

5-7.4.6 Transfer Data (DIN 8, 4 or DIN 8, 5)

Bits 4-15 of the accumulator receive individual column rows 12-9, respectively, from the card in process.

5-7.5 Card Punch Functions

5-7.5.1 Disconnect (DOT F, 0)

The card punch is disconnected and put into an off-line state. Its interrupt capability is inhibited and no more data can be accepted by the controller.

5-7.5.2 Stop Punch (DOT F, 3)

Same as Disconnect except that the interrupt capability is not inhibited until the end-of-card has been reached.

5-7.5.3 Punch Cards (DOT F, 1)

Card punching is initiated and an interrupt is issued approximately two seconds after the command to punch cards is given. The card punch punches two 12-bit columns of data at a time.

5-7.5.4 Last Punch (DOT F, 2)

Interrupts are inhibited for the remainder of the card in process and resume beginning with the next card. See High Speed Skip Feature, Section 5-7.9.1.

5-7.5.5 Offset Card In Process In Output Stacker (DOT F, 8)

The card in process is offset stacked in the output hopper (card stacked with left edge offset approximately one inch). This operation can be combined with any other instruction, i.e., a DOT F, A (8 + 2) would stop interrupts for the card in process while offset stacking that card in the hopper.

5-7.5.6 Load Data (DOT F, 4 or DOT F, 5)

Data to be punched is transferred from the accumulator (bits 4-15) to the card punch (rows 12-9). Two Load Data commands must be issued with each card punch interrupt.

5-7.6 Function Codes

The various 8 bit control codes are illustrated in hexadecimal and binary formats in Table 5-19.

When reading or punching cards, the initial select command starts card motion. The cards continue to feed until there are no more cards in the stacker or a Disconnect command is issued.

5-7.7 Status Codes.

Status is returned to the accumulator from the card reader or card punch after a DIN 8, 0 or a DIN F, 0 instruction is issued, respectively.

The card reader status word consists of seven bits (see Table 5-20). An all zero status word indicates that the card reader is ready to read. Bit-0 true means that the controller or device is not ready. Bit-1 true means that the controller is not ready to read. Bit-2 true means that there is a card in the read station. Bit-3 true indicates that the card reader hopper is empty. Bit-7 true means that the card reader has issued an interrupt that has not been serviced. Bit-14 true means that the DIN instruction to read data was late being issued by the CPU. Bit-15 indicates either a card reader malfunction or a late DIN from the CPU.

The card punch status word consists of six bits (see Table 5-20). An all zero status word indicates that the card punch is ready to punch. Bit-0 true means that the controller or device is not ready. Bit-1 true means that the controller is not ready to punch. Bit-7 true means that the card punch has issued an interrupt that has not been serviced. Bit-13 indicates there was a punch check error. Bit-14 means that the DOT instruction to transfer data was late. Bit-15 means that there was either a punch error or a late data transfer.

5-7.8 Interrupt Meaning

The interrupt from the card reader means that a card column has been read and is ready to be transferred to the CPU.

The interrupt from the card punch means that the punch is ready for data. Two data transfers are necessary to turn off the interrupt and status bit 7.

Table 5—19. Card Reader and Card Punch Command and Function Codes

Card Reader Commands	Function Codes								
	Hexadecimal Notation	Binary Notation							
		Bit Position							
DOT		8	9	10	11	12	13	14	15
Disconnect Controller	80	1	0	0	0	0	0	0	0
Read Cards	81	1	0	0	0	0	0	0	1
Stop Interrupts on Current Card	82	1	0	0	0	0	0	1	0
Disconnect After Current Card	83	1	0	0	0	0	0	1	1
Offset Card In Process	88	1	0	0	0	1	0	0	0
DIN									
Return Status to Accumulator	80	1	0	0	0	0	0	0	0
Transfer Data	84	1	0	0	0	0	1	0	0

Card Punch Commands	Function Codes								
	Hexadecimal Notation	Binary Notation							
		Bit Position							
DOT		8	9	10	11	12	13	14	15
Disconnect Controller	F0	1	1	1	1	0	0	0	0
Punch Cards	F1	1	1	1	1	0	0	0	1
Last Punch	F2	1	1	1	1	0	0	1	0
Stop Punch	F3	1	1	1	1	0	0	1	1
Offset Card In Process	F8	1	1	1	1	1	0	0	0
Transfer Data	F4	1	1	1	1	0	1	0	0
DIN									
Return Status to Accumulator	F0	1	1	1	1	0	0	0	0

Table 5-20. Card Reader and Card Punch Status Response

Card Reader Status	
Meaning When Bit True	Bit No.
Controller or Device Not Ready	0
Controller Not Ready	1
Card In Read Station	2
Hopper is Empty	3
Interrupt Has Not Been Serviced	7
Late Data Transfer	14
Card Reader Malfunction on Late Data Transfer	15
Card Punch Status	
Meaning When Bit True	Bit No.
Controller or Device Not Ready	0
Controller Not Ready	1
Interrupt Has Not Been Serviced	7
Punch Check Error	13
Late Data Transfer	14
Card Punch Malfunction, or Late Data Transfer	15

5-7.9 Timing

Figure 5-8 gives the interrupt timing for the card reader and card punch.

For the 1100 cpm card reader the time between initiation of the reader operation and the first interrupt is approximately 32 milliseconds. The interval between interrupts during the reading of a card is approximately 370 microseconds. The maximum time to service an interrupt is 300 microseconds. Failure to service an interrupt within 300 microseconds after it occurs causes a response error. Approximately 25 milliseconds are required between the interrupt for column 80 of one card and the interrupt for column 1 of the next card.

The time between initiation of card punch operation and the first interrupt is approximately 2 seconds if the card punch has been disconnected (DOT F, 0) or not previously selected. This time is required to start the card transport motor. If the

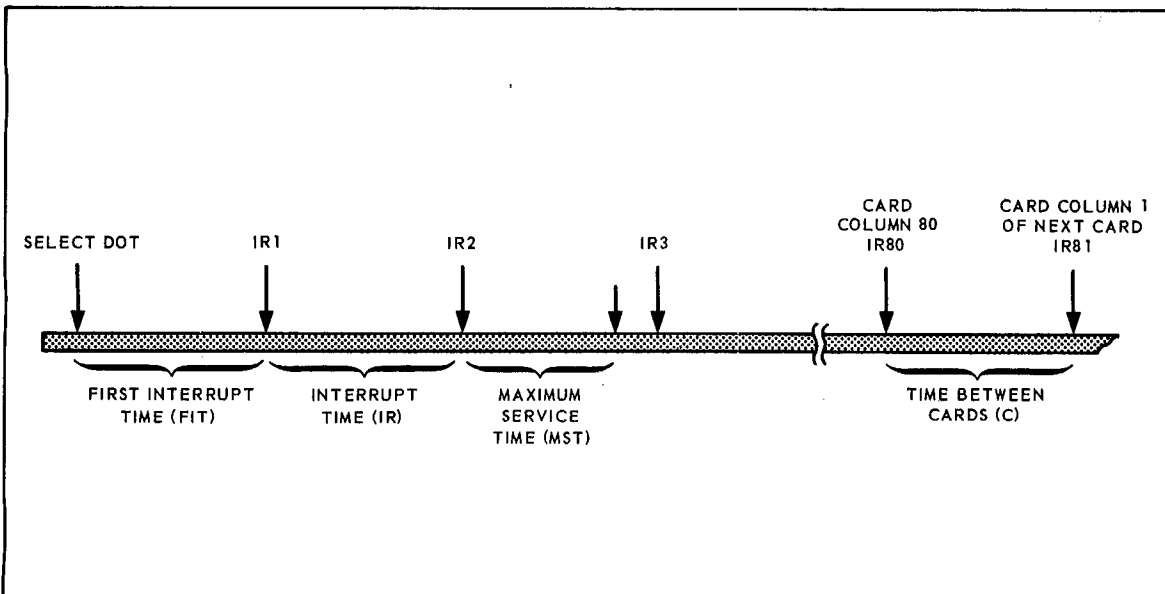
card transport motor is running the first interrupt is issued immediately. The interval between interrupts during the punching of a card is approximately 12.5 milliseconds. The maximum time to service an interrupt is 1 millisecond. Failure to service an interrupt within that time results in a response error. Approximately 100 milliseconds are required between the last interrupt of one card and the first interrupt of the next card.

5-7.9.1 Card Punch High Speed Skip Feature

The high speed skip feature, while not affecting the maximum or minimum punching rate, provides a significant increase in punching throughput on applications that involve a gap between fields. This increase is achieved by greatly accelerating card movement for all card columns not punched. For many punch patterns the speed increases 100% (see

Table 5-21. Punching Speeds Using High Speed Skip Feature

Punch Columns	Skip Columns	Punch Columns	Throughput Cards Per minute
1-80	- -	- -	100
1-10	11-70	71-80	215
1-10	11-50	51-60	228
1-10	11-30	31-40	246
1-20	21-80	- -	266
1-8	9-80	- -	400



Device	FIT	IR	MST	C	Rate
Card Reader	32 ms	370 us	300 us	25ms	1100 CPM
Card Punch*	0-2 sec	12.5 ms	1.0 ms	100 ms	100 CPM
Card Punch**	0-2 sec	12.5 ms	1.0 ms	100 ms	400 CPM
Card Reader	90 ms	1.0 ms	825 μ s	70 ms	400 CPM

*Punches in all 80 columns (See Section 5-7.10.1)

**Punches in 8 columns or less (See Section 5-7.10.1)

Figure 5-8. Card Reader. & Card Punch Timing

Table 5-21. In order to program this feature, the Last Punch (ref. 5-7.5.4) instruction must be issued.

5-7.10 Card Reader Operation (Model No. 75611, 1100 CPM)

5-7.10.1 Controls and Indicators

The controls and indicators for the card reader are shown in Figure 5-9 and described in Table 5-22.

5-7.10.2 Loading

The reader input hopper holds a maximum of 1000 cards. Load cards with the 9 edge toward panel, 12 edge toward operator, and column 1 toward read station (operator's left). Before loading a deck of cards riffle both ends and flex the deck, then square up the deck by jogging it on a smooth surface. Place a one inch portion of the deck in the hopper, holding the right side of the deck higher so that the leading edge of the bottom card rests against the picker throat block. Allow the cards to fall into place in this position. Add the rest of the deck. Additional cards may be added to a partially full hopper during reading without stopping the reader.

5-7.10.3 Operation

Press POWER switch-indicator if it is lighted white. (If it is not lighted turn on the circuit breaker at the rear of the cabinet.) It should light green, START should light white, STOP should light

yellow, and CLEAR should light white. If CLEAR is red check the indicators to determine the nature of the trouble. Correct the trouble and press the CLEAR switch before proceeding.

Press the START switch. START lights green, and STOP lights white. The card reader is now 'on line' and ready to be operated under program control.

Card reading may be stopped by pressing the STOP switch or allowed to stop automatically when the input hopper is empty, the output hopper full, or by a command from the CPU.

To remove cards from the stacker press down on the elevator and lift the deck straight out. The stacker holds 1000 cards. Do not remove cards from the stacker while cards are being read.

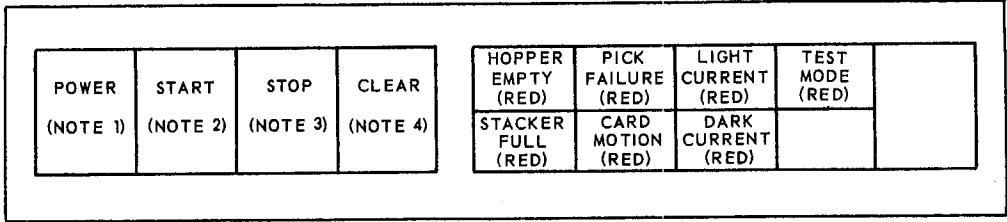
5-7.11 Card Punch Operation (Model No. 75613, 400 CPM)

The controls and indicators for the card punch are shown in Figure 5-9 and described in Table 5-23.

To punch cards press the POWER ON switch and then press the RUNOUT switch to clear the card path.

Load the card input hopper with the cards face down and with the nine edges first. Place the card weight on the cards.

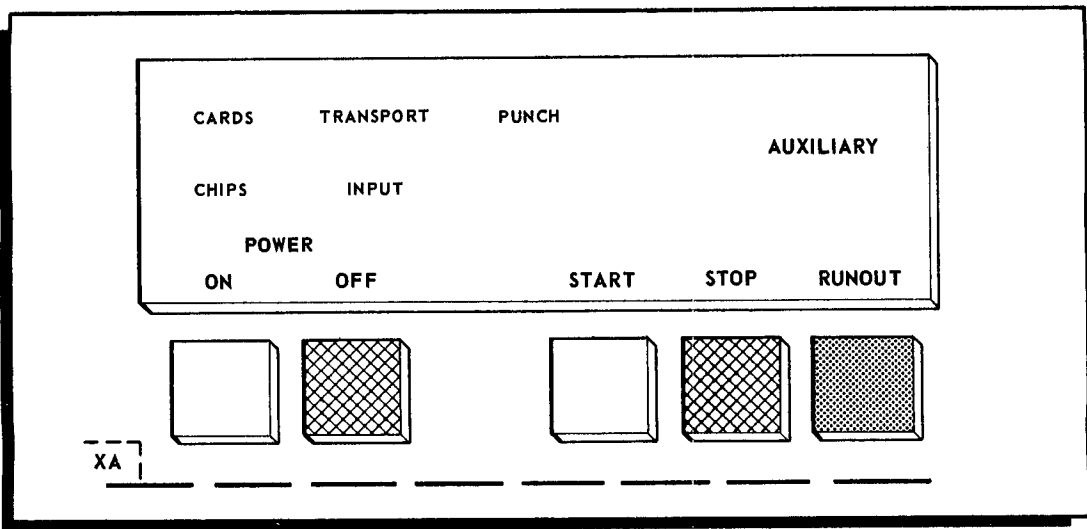
Press the START switch. The card punch is now on-line and ready to be operated under program control.



READER

NOTES:

- 1 GREEN-ON, WHITE-OFF
- 2 GREEN-START, WHITE-STOP
- 3 YELLOW-STOP, WHITE-START
- 4 WHITE-NO TROUBLE, RED-TROUBLE



PUNCH MAIN PANEL

Figure 5-9. Card Reader (1000 CPM) and Card Punch Controls and Indicators

Table 5-22. Card Reader (1000 CPM) Controls and Indicators

Control or Indicator	Function
POWER Switch-indicator	Applies operating power and causes power on reset to be generated which establishes the initial condition of the reader logic. Lights green when on, red when off (red in off condition indicates reader is connected to AC source and circuit breaker is on).
START Switch-indicator	Places reader logic in start condition (a requirement for ready condition). Causes picking and reading to begin when in TEST mode. Lights green in Start condition. White in stop condition.
STOP Switch-indicator	Places the reader logic in stop condition and inhibits the ready condition. Stops picking and reading in TEST mode. Lights yellow in stop condition. White in start condition.
CLEAR Switch-indicator	Resets monitoring circuits after a trouble condition has been corrected. Lights red when trouble. White when no trouble.
*HOPPER EMPTY indicator	Lights red when there is no card in the input hopper.
*STACKER FULL Indicator	Lights red when output stacker is full.
*PICK FAILURE Indicator	Lights red when a card does not reach the read station within a set time after the picker is energized
*CARD MOTION Indicator	Lights red when a card fails to complete passage through the read station within a set time or when a card is more than approximately 1/3 column out of synchronization with the timing signals.
*LIGHT CURRENT Indicator	Lights red when the read station fails to detect light after a card is picked and before the leading edge of the card reaches the read station.
*DARK CURRENT Indicator	Lights red when read station is not darkened by the leading edge of the card before column 1 is sensed.
TEST MODE Indicator	Lights red when TEST/NORMAL switch is in TEST position.
*Once lighted these indicators remain on until trouble is corrected and CLEAR switch is pressed.	

Table 5-22. Card Reader (1000 CPM) Controls and Indicators (Cont)

Control or Indicator	Function
TEST/NORMAL Switch	Causes continuous pick and read for test purposes. (Located on reader logic assembly)
Circuit Breaker	Applies primary AC power. (Located on rear of reader.)

Table 5-23. Card Punch Controls and Indicators

Control or Indicator	Function
*POWER ON Switch-Indicator	Applies primary power to punch. Lights TRANSPORT and AUXILIARY indicators.
POWER OFF Switch	Removes primary power.
*START Switch-Indicator	Functions only when STOP switch/indicator is lighted. Places punch in ready condition and inhibits busy signal. Turns off CARDS and INPUT indicators.
*STOP Switch-Indicator	Causes busy signal to be generated. Stops punch operation at end of card-in process. Turns off START switch-indicator.
*RUNOUT Switch-indicator	Functions only when STOP switch-indicator is lighted. Causes card to be removed from card path (used to clear a jam).
**PUNCH Indicator	Lights when the holes punched in the card fail to compare with the input data or when a jam occurs in the card transport mechanism.
**CARDS Indicator	Lights when input hopper is empty, output stacker is full, the stacker pressure switch is released, a card is jammed in the output stacker rollers, or the chip box is either full or removed from the machine.
**CHIPS Indicator	Lights when the chip box is either full or removed from the machine.

Table 5-23. Card Punch Controls and Indicators (Cont)

Control or Indicator	Function
**TRANSPORT Indicator	Lights when a jam occurs in the card transport mechanism. The card transport motors are turned off when a jam is detected.
**INPUT Indicator	Lights when the wait station is not filled after the START switch-indicator has been activated or is not refilled following a punch instruction.
**AUXILIARY Indicator	Lights when a condition occurs which the operator does not normally correct. Either the nature of the trouble is indicated on the auxiliary control panel or the unit is in test mode.
Auxiliary Control Panel TEST Switch	Selects normal operation or one of nine test modes.
INTERLOCK Indicator	Lights when one of the covers is open or the punch head is not locked in position.
BREAKER Indicator	Lights when a power circuit breaker opens.
PICKER Indicator	Lights when the picker motor drive circuits fail in such a manner as to cause the picker motor to run at a speed injurious to the mechanism.
HEAT Indicator	Lights when one of the cooling fans fail.
<p>* Lights when activated. Activated upon release after being pressed. **Visible only when activated.</p>	

5-7.12 Card Reader Operation (Model No. 75612, 400 CPM)

5-7.12.1 Controls and Indicators

The controls and indicators for the 400 CPM card reader are shown in Figure 5-9.1 and described in Table 5-23.1.

5-7.12.2 Loading

Check that both the stacker bin and hopper are empty of cards. Load up to 500 perforated cards in the supply hopper, printed side down and column 1 toward the read station.

5-7.12.3 Operation

Press and release the POWER switch. The switch indicator will be illuminated green. This indicates the d-c power supply is activated and the a-c circuits are enabled.

Press and release the MOTOR switch. This will cause the drive motor to start and the indicator will be illuminated green.

Press and release the START switch. The indicator will be illuminated green. The motor will continue to operate but card processing will await a read command from the computer.

When the read command is sent to the card reader, cards will be fed to the stacker in the same rotation they were in the supply hopper. When no trouble occurs, cards will continue to be read until the hopper is empty or the STOP switch is pressed.

5-7.12.4 Operational Troubles

HOPPER EMPTY

To correct a hopper empty condition, proceed as follows:

- (a) Reload the supply hopper.
- (b) Press and release the MOTOR push switch.
- (c) Press and release the START push switch.
- (d) When the external equipment sends a read command, card reading will continue.

PICK FAIL

A pick failure is normally the result of the leading edge of a card being damaged or the card thickness being out of tolerance. The pneumatic gate of the picker assembly will pass cards up to 0.010 inch thick and not pass cards 0.012 inch thick. Where the leading edge of a card is damaged so its thickness is greater than 0.010 inch, a picker failure will result and stop the reader operation. To correct the condition, proceed as follows:

- (a) Remove the cards from the supply hopper and examine the leading edge of the bottom card for damage.
- (b) Smooth any burr on the leading edge of the card and try to manually insert the leading edge of the card in the picker pneumatic gate.
- (c) Where the leading edge of the card entered the pneumatic gate, leave just the edge of the card entered in the gate and reload the supply hopper.
- (d) Press and release the MOTOR push switch.
- (e) Press and release the START push switch.
- (f) When the external equipment sends a read command (IRC), card reading will continue.

LIGHT CURRENT OR DARK CURRENT ERRORS

A light current or dark current error is normally the result of a damaged card. The damaged card will be the last card fed to the stacker. (Top card in the stacker bin.) When either of these failures occur, proceed as follows:

- (a) Remove the top card from the stacker bin and examine the ends of the card for damage.
- (b) Place the card in the supply hopper as the bottom card in the hopper.
- (c) Press and release the MOTOR push switch.
- (d) Press and release the START push switch.
- (e) Where the same error occurs, remove the card from the stacker bin, repeat steps (c) and (d).
- (f) Where card reading continues, the card removed from the stacker bin contains damage not noticed.
- (g) Replace card with a new card.

Table 5-23.1. Card Reader (400 CPM) Controls and Indicators

Control or Indicator	Function		
POWER (Switch/ Indicator)	Controls AC power to DC power supply. When ON, indicator is illuminated GREEN.	CARD MOTION (Indicator)	Indicator is illuminated RED when time interval between a card entering and leaving read station does not correspond to prearranged card column counts.
MOTOR (Switch/ Indicator)	Clears all trouble signals and starts drive motor. When motor activated, indicator is illuminated GREEN.	LIGHT CURRENT (Indicator)	Indicator is illuminated RED when all phototransistors are not conducting when a card is not in the read station.
START (Switch/ Indicator)	Puts card reader ON-LINE with card controller. When ON-LINE, indicator is illuminated GREEN.	DARK CURRENT (Indicator)	Indicator is illuminated RED when all phototransistors do not stop conducting (go dark) between the leading edge of a card and column 1, or between column 80 and the trailing edge of a card.
STOP (Switch/ Indicator)	Puts card reader OFF-LINE with card controller. When OFF-LINE, indicator is illuminated YELLOW.		
PICK FAILURE (Indicator)	Indicator is illuminated RED when card fails to reach read station.		

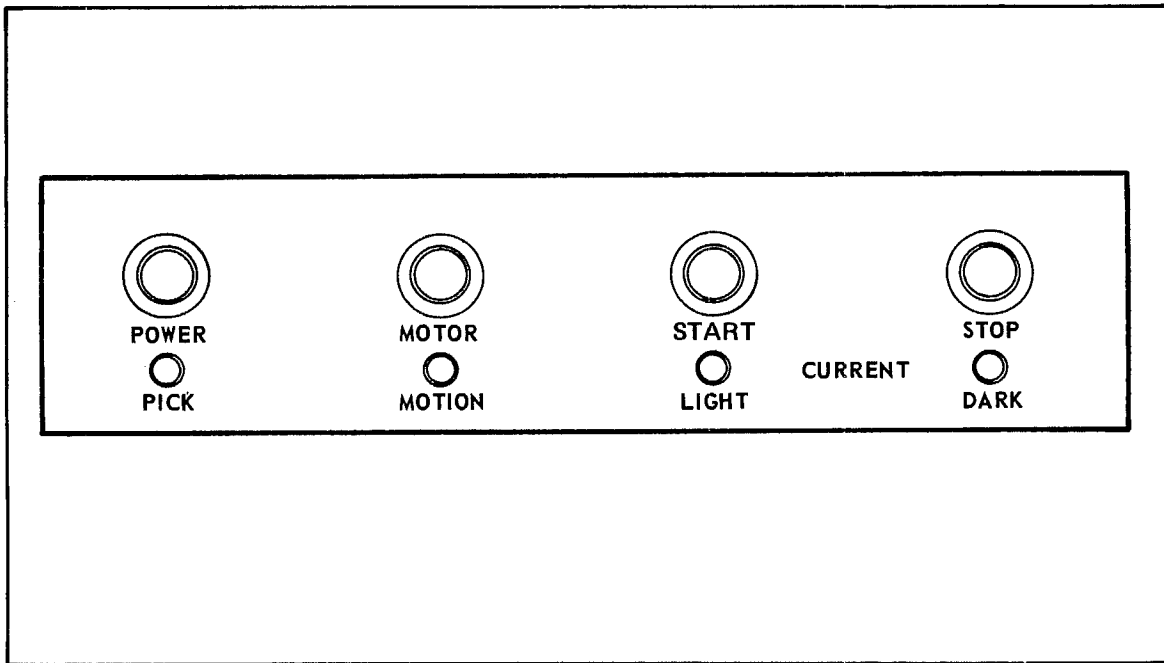


Figure 5-9.1. Card Reader (400 CPM) Control Panel.

5-8 LINE PRINTER

The line printer provides a high speed printing capability for the 706 computer. Specifications are given in Table 5-23.2.

5-8.1 Speed

Data are printed at rates of 360, 600 and 1,000 lines per minute; 132 characters per line. Data transfer rate is asynchronous to 125 kHz for the 360 and 600 lpm printer and asynchronous to 167 kHz for the 1,000 lpm printer.

5-8.2 Data Type

The character set for the printer is listed in Table 5-24. The hexadecimal codes that represent the

characters are a subset of the eight-bit ASCII codes. The printer uses only six of the eight bits. The two most significant bits are not used.

5-8.3 Data Format

The 6-bit data are transferred between bits 10-15 of the accumulator to the printer buffer. The most significant 10 bits (0-9) of the accumulator do not affect the character to be printed.

5-8.4 Line Printer Functions

5-8.4.1 Disconnect (DOT 5,0)

The line printer is disconnected and put into an off-line state. Its interrupt capability is inhibited and no more data can be accepted by the printer buffer. Status bit - 0 becomes true.

Table 5-23.2. Line Printer Specifications

Type	Drum		Vertical Paper Adjustments	Vertical paper tension, dynamic vertical paper position
Speeds	360, 600, and 1,000 lpm		Ribbon	Vertical Feed roll type, 14 in. wide x 20 yds. long
No. of Characters	64 standard		Vertical Format Tape	8 channels
Characters per line	132		Step Count	1 - 7 lines
Character Size	Height	Width	Power Specifications	
Upper Case (typical)	0.095 in.	± .003 in.	Power	1.4 KW
Lower Case (maximum)	0.065 in.	± .003 in.	Input Voltage	115 VAC ± 10%
Horizontal Character Spacing	10 characters per inch or 0.100 in. ± .005 in. center to center		Frequency	60 Hz ± 1 Hz
Vertical Spacing	6 characters per inch or 0.167 in. ± .010 in. line to line		Operating Environment	
Line Straightness	Deviation from mean horizontal line does not exceed ± .010 in.		Operating Temperature	50° F - 100° F
Paper Specifications	Standard, edge-punched (½ in. hole center) fanfold paper up to 19 in. wide		Nonoperating Temperature	0° F - 125° F
Single copy min. weight	15 lb. bond		Operating Humidity	20% - 80%
Six part with carbon	12 lb. bond		Nonoperating Humidity	5% - 95%
Tabulation card	0.007 in. thickness		Physical Characteristics	
Adjustments			Dimensions	Height: 48 inches Width: 47 inches Depth: 26 inches
Horizontal Paper Adjustments	Left margin detents, form width lock, horizontal paper tension, horizontal paper position		Weight	838 lbs.

Table 5-24. Line Printer Character Set

Char.	Hex. Code	Char.	Hex. Code	Char.	Hex. Code	Char.	Hex. Code
@	C0	P	D0	Blank	A0	0	B0
A	C1	Q	D1		A1	1	B1
B	C2	R	D2	"	A2	2	B2
C	C3	S	D3	No.	A3	3	B3
D	C4	T	D4	\$	A4	4	B4
E	C5	U	D5	%	A5	5	B5
F	C6	V	D6	&	A6	6	B6
G	C7	W	D7	'	A7	7	B7
H	C8	X	D8	(A8	8	B8
I	C9	Y	D9)	A9	9	B9
J	CA	Z	DA	*	AA	:	BA
K	CB	[DB	+	AB	;	BB
L	CC	\	DC	,	AC	<	BC
M	CD]	DD	-	AD	=	BD
N	CE	↑	DE	.	AE	>	BE
O	CF	←	DF	/	AF	?	BF

Hex Code	Accumulator Bit					Function
	10	11	12	13	14 15	
X0	X	X	0	0	0 0	Slew to Channel 0 (1 line)
X1	X	X	0	0	0 1	1 (TOF)
X2	X	X	0	0	1 0	2
X3	X	X	0	0	1 1	3
X4	X	X	0	1	0 0	4
X5	X	X	0	1	0 1	5
X6	X	X	0	1	1 0	6
X7	X	X	0	1	1 1	Slew to Channel 7
X8	X	X	1	0	0 0	Move Paper 1 Line
X9	X	X	1	0	0 1	Move Paper 1 Line(s)
XA	X	X	1	0	1 0	2
XB	X	X	1	0	1 1	3
XC	X	X	1	1	0 0	4
XD	X	X	1	1	0 1	5
XE	X	X	1	1	1 0	6
XF	X	X	1	1	1 1	Move Paper 7 Lines

Note: At print time if Bit-10 is zero, paper is moved according to bits 12–15.
If Bit-10 is a one, paper motion is inhibited.

Figure 5-10. Slew Characters for Line Printer.

5-8.4.2 Load Slew Register (DOT 5,1)

Bits 10–15 of the accumulator are loaded into the slew register. No data are transferred and paper motion is not initiated by this command. See Figure 5-10 for slew character definitions. This command may be issued any time before the Print Line and Move Paper command is given.

5-8.4.3 Load (Transfer) One Character (DOT 5,4)

One character is transferred from the 706 accumulator (bits 10–15) to the line printer data buffer with this instruction.

No interrupt is generated because of a character transfer. The line printer buffer accepts characters as fast as the 706 can transmit them, provided that the controller is not busy printing.

The line printer data buffer stores the characters up to a maximum of 133 (132 data characters + 1 slew character). See Section 5-8.10.1 for overspill. Unused character spaces are printed as blanks.

5-8.4.4 Print Line and Move Paper (DOT 5,8)

The Print Line and Move Paper command causes bits 10–15 of the accumulator to be transferred (last character), the line to be printed and paper moved according to the slew character.

If bit-10 of the slew character is a one, paper motion is inhibited. If bit-10 of the slew character is a zero, paper is moved according to the slew character (see Figure 5-10).

During the print cycle, status bits 0 and 1 are set true. Upon completion of printing, an interrupt is sent to the CPU and status bits 0 and 1 are set false. The printer buffer is cleared. The slew character remains and can be used for the next line.

5-8.4.5 Move Paper (Dot 5,9)

This command causes bits 12–15 of the accumulator to be loaded into the slew register. Bit-10 of the slew character is forced to a zero.

Paper motion only takes place according to the slew character just transferred. With bit-12 equal to zero, paper may be slewed to any one of eight channels to a punch in the control tape. If bit-12 equals a one, paper may be spaced from one to seven lines (see Figure 5-10). The data buffer is cleared and the slew character buffer is cleared. Status bit-0 is false while paper is in motion.

5-8.5 Function Codes

The various 8-bit control codes are illustrated in hexadecimal and binary formats in Table 5-25.

5-8.6 Status Codes

Status is returned to the accumulator from the line printer after a DIN 5,0 instruction is issued.

The line printer status word consists of three bits (Table 5-26). An all-zero status word indicates that the line printer is ready to receive characters, move paper command, or a print command. Bit-0 true means that the controller device is not ready. Bit-1 true indicates that the device is not ready. Bit-7 true means that the line printer is interrupting the CPU and has not been serviced.

5-8.7 Interrupt Meaning

The interrupt from the line printer means that printing has been completed. If the line has been printed, the line printer data buffer has been cleared and a new line of data may be transmitted.

5-8.8 Timing

The operational speed of the line printer is a function of the drum rotational speed, the paper speeds and the information being printed. Figure 5-10.1 gives the line printer timing based on the drum and line paper speeds listed in Table 5-26.1.

Table 5–25. Line Printer Commands and Function Codes

Command	Function Codes								
	Hexadecimal Notation	Binary Notation							
		Bit Position							
DOT		8	9	10	11	12	13	14	15
Disconnect Controller	50	0	1	0	1	0	0	0	0
Load Slew Register	51	0	1	0	1	0	0	0	1
Load One Character	54	0	1	0	1	0	1	0	0
Print Line and Move Paper	58	0	1	0	1	1	0	0	0
Move Paper	59	0	1	0	1	1	0	0	1
DIN									
Return Status to Accumulator	50	0	1	0	1	0	0	0	0

Table 5-26.1. Drum, Paper and Print Speeds

360 Line Per Minute Printer	Drum Rotational Speed*		360 rpm Nominal	
	Drum Rotational Time		160 msec/Revolution	
	Paper Speed			
	One-Line Advance	Two-Line Advance	Three-Line Advance	Skip Feed
	25 msec. max.	35 msec. max.	44 msec. max.	20 in./sec min. or 8.4 msec/line after 3rd line.
	Print and Slew Time			
	Up to 54 Adjacent Characters		Over 54 Characters	
	Print Time	Slew Time	Print Time	Slew Time
142 msec.	25 msec.	167 msec.	25 msec.	
600 Line Per Minute Printer	Drum Rotational Speed		600 rpm Nominal	
	Drum Rotational Time		100 msec/Revolution	
	Paper Speed			
	One-Line Advance	Two-Line Advance	Three-Line Advance	Skip Feed
	25 msec. max.	35 msec. max.	44 msec. max.	20 in./sec min. or 8.4 msec/line after 3rd line.
	Print and Slew Time			
	Up to 48 Adjacent Characters		Over 48 Characters	
	Print Time	Slew Time	Print Time	Slew Time
75 msec.	25 msec.	100 msec.	25 msec.	
1,000 Line Per Minute Printer	Drum Rotational Speed		1,000 rpm Nominal	
	Drum Rotational Time		60 msec/Revolution	
	Paper Speed			
	One-Line Advance	Two-Line Advance	Three-Line Advance	Skip Feed
	15 msec.	25 msec.	30 msec.	35 in./sec max or 4.8 msec/line after 3rd line
	Print and Slew Time			
	Up to 48 Adjacent Characters		Over 48 Characters	
	Print Time	Slew Time	Print Time	Slew Time
45 msec.	15 msec.	60 msec.	15 msec.	

*64 character drum

5-8.8.1 360 LPM Printer

Using a 64-character drum, the line printer prints at 360 lines per minute when using any combination of up to 54 physically adjacent characters plus space code assuming single-line paper stepping between print operations. In this case, up to 142 msec of each 167 msec rotational time is used for printing. During the remaining 25 msec, the next line to be printed is stored and the paper is advanced while the 10 unused characters are passing the print position. When printing more than 54 physically adjacent characters, an entire drum revolution may be required to scan all desired characters. Thus, using a 64-character drum, the printer prints at 300 lines per minute when using any combination of between 54 and 64 physically adjacent characters plus space code, assuming single line stepping between print operations. In this case, 167 msec is used for printing and an additional 25 msec is required to step the paper.

Alphabetic, numerics and commonly used punctuation marks are usually located in adjacent positions on the drum so that the probability is high that on most lines of print, the character set which is used is 54 characters or less and the line printer

will operate at 360 lines per minute. It should be noted that the print operation is asynchronous, i.e., character scanning is initiated as soon as all data is loaded without waiting for any index point.

5-8.8.2 600 LPM Printer

Using a 64-character drum, the line printer prints at 600 lines per minute when using any combination of up to 48 physically adjacent characters plus space code assuming single-line paper stepping between print operations. In this case, up to 75 msec of each 100 msec rotational time is used for printing. During the remaining 25 msec, the next line to be printed is stored and the paper is advanced while the 16 unused characters are passing the print position.

When printing more than 48 physically adjacent characters, an entire drum revolution may be required to scan all desired characters. Thus, using a 64-character drum, the printer prints at 480 lines per minute when using any combination of between 49 and 64 physically adjacent characters plus space code, assuming single line stepping between print operations. In this case, a maximum of 100 msec is used for printing and an additional 25 msec is required to step the paper.

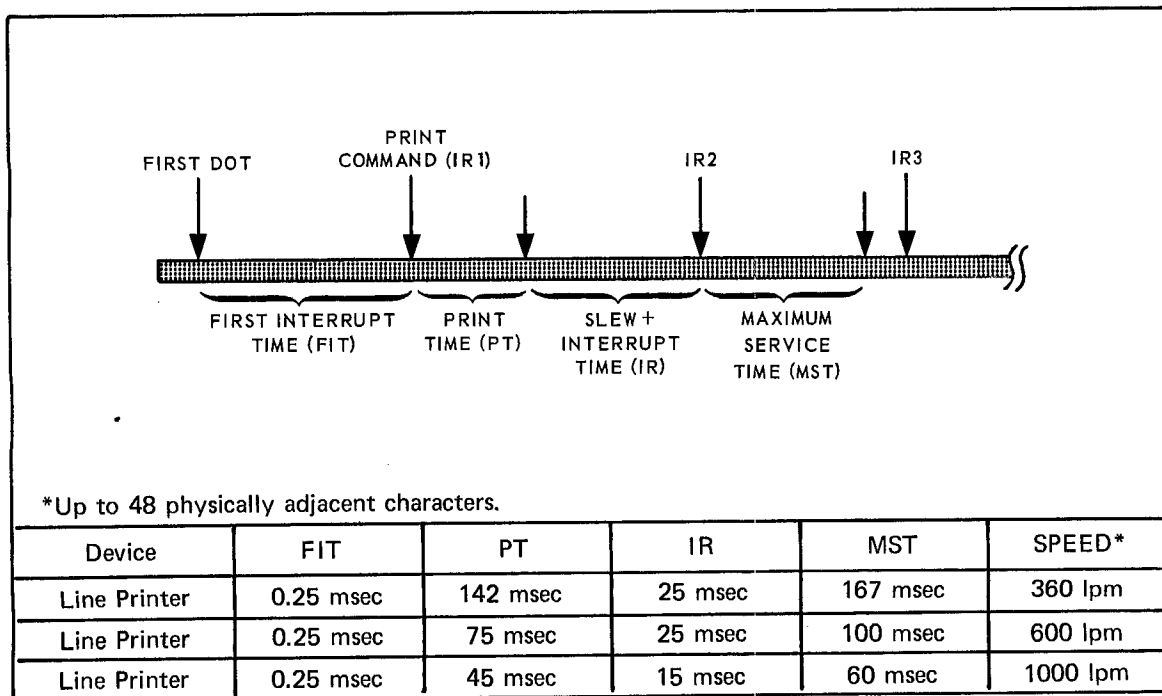


Figure 5-10.1. Line Printer Timing

Alphabetic, numerals and commonly used punctuation marks are usually located in adjacent positions on the drum so that the probability is high that on most lines of print, the character set which is used is 48 characters or less, and the line printer will operate at 600 lines per minute. It should be noted that the printing operation is asynchronous, i.e., character scanning is initiated as soon as all data is loaded without waiting for any index point.

In no case should the printer be operated continuously at greater than 750 lines per minute.

5-8.8.3 1,000 LPM Printer

Using a 64-character drum, the printer prints at not less than 1,000 lines per minute when using any combination of up to 48 physically adjacent characters plus space code assuming single-line paper stepping between print operations. In this case, up to 45 msec of each 60 msec rotational time is used for printing. During the remaining 15 msec, the next line to be printed is stored and the paper is advanced while the 16 unused characters are passing the print position.

When printing more than 48 physically adjacent characters, an entire drum revolution may be required to scan all desired characters. Thus, using a 64-character drum, the printer prints at not less than 800 lines per minute when using any combination of between 49 and 64 physically adjacent characters plus space code, assuming single line stepping between print operations. In this case, a maximum of 60 msec is used for printing and an additional 15 msec is required to step the paper.

Alphabetic, numerals and commonly used punctuation marks are usually located in adjacent positions on the drum so that the probability is high that on most lines of print, the character set which is used is 48 characters or less and the line printer will operate at 1,000 lines per minute. It should be noted that the print operation is asynchronous, i.e., character scanning is initiated as soon as all data is loaded without waiting for any index point.

5-8.9 Vertical Format Tape

The vertical format tape is an 8-channel (0-7) paper tape that controls page and line formatting for the line printer output. Tapes come with prepunched sprocket and feed strobe holes. The user must punch the eight control channels according to his specific requirement (see line printer OEM manual

for punching instructions). The eight channels on the format tape are normally punched as follows (not required):

Channel 0	Hole provided for each line to be printed
Channel 1	Top of Form
Channels 2-7	As required

5-8.10 Operation Notes

5-8.10.1 Data Buffer Overspill

If 134 or more characters are transmitted before a print command is issued, the line printer will automatically replace the second character with the 134th character (1st or slow character unaffected) and continue to replace characters on a one for one basis.

5-8.10.2 Overprinting

The line printer output may be overprinted (to darken type) by issuing a print command without a paper motion slow character and then issuing another print command.

5-8.11 Line Printer Operation (Model Nos. 75622, 75623 - 360 LPM and 600 LPM)

5-8.11.1 Controls and Indicators

The controls and indicators for the 360 and/or 600 lpm line printer are shown in Figure 5-11. and described in Table 5-27.

5-8.11.2 Paper Loading

The paper feed mechanism is accessed by raising the printer top cover and swinging the drum gate forward. When the form to be loaded is identical to that for the previous printout, the paper is simply inserted up between the drum gate and the hammer bank and the paper drive holes aligned with the tractor paper drive teeth. Pressure plates then secure the paper in the mechanism.

When the mechanism is not adjusted for the particular sized paper, the paper feed tractors must first be positioned to obtain an approximate adjustment for the form. The paper is then loaded and the tractors are statically adjusted horizontally and vertically until the paper is taut on the mechanism. Vernier controls are provided to allow dynamic adjustment of the mechanism while printing. Addi-

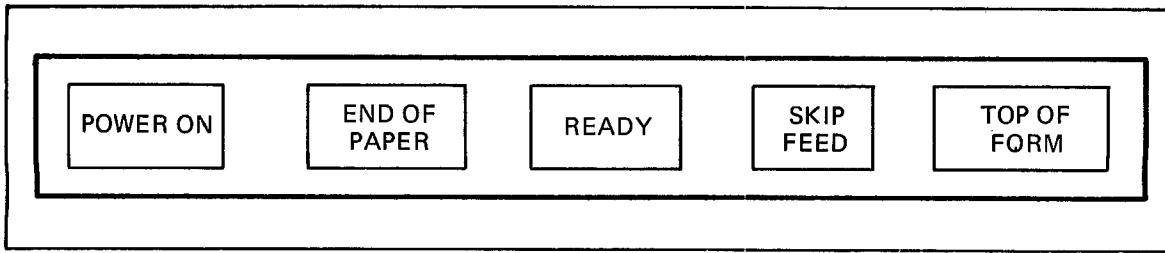


Figure 5-11.1 Line Printer Control Panel (360 and 600 lpm)

Table 5-27. Line Printer Operating Controls and Indicators (360 and 600 lpm)

Control or Indicator	Function
<p>POWER ON pushbutton/ indicator</p> <p>END OF PAPER indicator</p>	<p>Used to control application and removal or primary AC power to printer. Lights when power is on.</p> <p>When lighted, indicates that the paper has broken between the upper and lower tractors or that the paper supply has been exhausted. When this condition occurs, the READY indicator will be extinguished and the printer will be inhibited from accepting any additional character information; however, any line being printed will be completed.</p>
<p>READY pushbutton/ indicator</p>	<p>Lights to indicate the status of the line printer (ready or standby) after AC power is applied. In the ready condition, data may be sent to the line printer from the controller. Depressing the lighted READY pushbutton-indicator will place the printer in the standby condition.</p>
<p>SKIP FEED pushbutton</p>	<p>When depressed, the paper slews at 20 inches per second until the switch is released. If a tape loop is in the tape reader, the paper will hold at the first channel 7 entry after the switch is released.</p>
<p>TOP OF FORM pushbutton</p>	<p>When depressed and released, with a tape loop in the tape reader, the paper is advanced to the next top-of-form position. When a tape loop is not in the tape reader, the paper will advance one line and stop each time the switch is depressed and released.</p>

tional controls are provided to ensure optimum print quality for various part paper.

The line printer should not be operated without paper and ribbon loaded. The only time the hammers are fired without paper or ribbon is when adjusting a newly installed hammer bank. Extended operation of this kind will damage the hammers.

5-8.11.3 Ribbon Changing

The ribbon assembly is accessed by lifting the cover and opening the drum gate. The ribbon is easily removed or inserted. The ribbon is rolled on two spools which are secured to the frame by spring loaded bearings.

5-8.11.4 Cleaning, Filter, Adjustments, and Lubrication

Cleaning should be performed periodically, the cleaning intervals being dependent upon operating usage. The drum gate and hammer bank should be vacuumed to remove accumulated paper dust. When required, the drum should be cleaned to remove ink and any paper dust.

The printer chassis has two aluminum mesh filters which should be checked regularly and cleaned when dirty. The electronics gate uses two fiber glass filters which should be checked regularly and replaced when dirty.

The only adjustments incorporated are those required to ensure optimum printing quality and operational flexibility. No periodic adjustments are required to compensate for any deterioration or variation in component performance.

The only operational adjustments are those required when using paper of various dimensions and thickness. Hammer adjustment is required only at the time of installation or replacement. The paper drive belt should be checked quarterly and requires less than two minutes to adjust.

Lubrication is normally a semi-annual procedure. The lubrication points are the ribbon motors, tape reader bearings, vertical positioning and tension controls, chassis blower motors and muffin fans.

5-8.12 Line Printer Operation (Model No. 75624 - 1,000 LPM)

5-8.12.1 Controls and Indicators

The controls and indicators for the 1,000 lpm line printer are shown in Figure 5-11.1 and described in Table 5-27.1.

5-8.12.2 Paper Loading

The paper feed mechanism is accessed by raising the printer top cover and swinging the drum gate forward. When the form to be loaded is identical to that for the previous printout, the paper is simply inserted up between the drum gate and the hammer bank and the paper drive holes aligned with the tractor paper drive teeth. Pressure plates then secure the paper in the mechanism.

When the mechanism is not adjusted for the particular sized paper, the paper feed tractors must first be positioned to obtain an approximate adjustment for the form. The paper is then loaded and the tractors are statically adjusted horizontally and vertically until the paper is taut on the mechanism. With the manual vertical paper positioner option, controls are provided to conveniently orient the vertical position of the initial entry. Vernier controls are provided to allow dynamic adjustment of the mechanism while printing. Additional controls are provided to ensure optimum print quality for various thickness of paper.

5-8.12.3 Ribbon Changing

The ribbon assembly is also accessed by lifting the cover and opening the drum gate. The ribbon is easily removed or inserted. The ribbon is rolled on two spools which are secured to the frame by spring loaded bearings.

5-8.12.4 Cleaning, Filters, Adjustments and Lubrication

Cleaning should be performed periodically, the cleaning intervals being dependent upon operating usage. The drum gate and hammer bank should be vacuumed to remove accumulated paper dust. When required, the drum should be cleaned to remove ink and any paper dust.

The printer chassis has two aluminum mesh filters which should be checked regularly and cleaned when dirty. The electronics gate uses two fiber glass

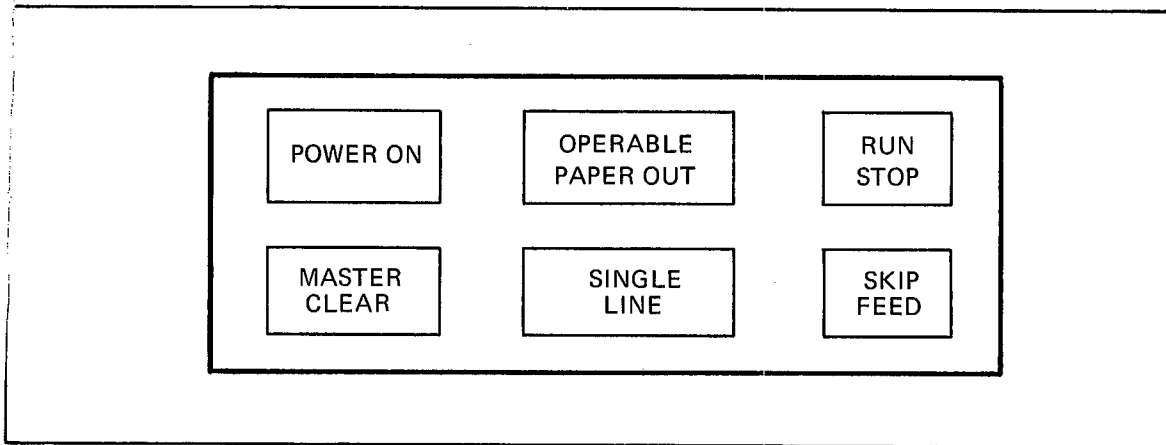


Figure 5-11.1 Line Printer Control Panel (1000 lpm)

Table 5-27.1 Line Printer Operating Controls and Indicators (1000 lpm)

Control or Indicator	Function
POWER ON pushbutton/	Used to control application and removal of AC power to the printer. Lights when power is on.
OPERABLE/ PAPER OUT dual indicator	Operable portion lights when printer is ready and on line to accept data. Paper out portion lights when paper is torn or supply is exhausted.
RUN/STOP pushbutton/ indicator	Alternately puts printer on-line and off-line with the controller.
MASTER CLEAR pushbutton	Resets all logic circuits, clears one-line buffer and initializes printer.
SINGLE LINE pushbutton	Advances paper one line. Paper motion is initiated upon release of the pushbutton.
SKIP FEED pushbutton (Top of Form)	When no tape loop option is used, depressing this pushbutton causes paper to slew at 15 inches per second until pushbutton is released. When tape loop option is exercised, this pushbutton causes paper to advance according to paper tape channel 1.

filters which should be checked regularly and replaced when dirty.

The only adjustments incorporated are those required to ensure optimum printing quality and operational flexibility. No periodic adjustments are required to compensate for any deterioration or variation in component performance.

The only operational adjustments are those required

when using paper of various dimensions and thickness. Hammer adjustment is required only at the time of installation or replacement. The paper drive belt should be checked quarterly and requires less than 20 minutes to adjust.

Lubrication is normally a semi-annual procedure. The lubrication points are the ribbon motors, tape reader bearings, vertical positioning and tension controls, chassis blower motors and muffin fans.

5-9 DISC MEMORY

The disc memory provides up to 385,024 words of auxiliary storage for the 706 computer. Up to four disc drives may be attached to the disc controller. Data is transferred directly to and from memory via the Direct Memory Access (DMA) Channel.

5-9.1 Speed

Disc units are available with disc rotation rates of 1800 RPM. Data may be transferred between the CPU and core memory at the rate of 1,110,000 words per second. The disc provides an average access time of 16.7 milliseconds and a data transfer rate of 187,000 words per second and requires approximately one-sixth of the core memory access time.

5-9.2 Data Type

Data are transferred between the disc and memory in binary form.

5-9.3 Data Format

Full 16-bit data words are transferred directly between the disc controller and the 706 core memory via the DMA bus.

5-9.4 Sectors Per Track and Words Per Sector Selection

The number of sectors per track and the number of words per sector are interdependent. There may be 24 to 512 sectors per track and 10 to 256 words per sector and with 512 sectors there are 10 words per sector. Most units use 128 sectors per track and 47 words per sector so that programming may be standardized.

The sector counter and word counter in the controller must count to values determined by the prerecorded sector marks. The count for the sector counter is selected by installing jumpers on master module connectors. The count for the word counter is selected by installing jumpers on the master module connectors. See Disc Memory Manual RC2024, P/N 391346 for jumper installation instructions.

To determine the number of sectors per track when the number of words per sector is known, multiply the number of words per sector by 16 and add 21. This result is the number of bits per sector. Next, divide the number of bits per track (100,000) by the number of bits per sector. The quotient equals the number of sectors per track, and the remainder equals the fraction of a sector at the end of the track.

If the disc controller reaches the end of a track in the read mode, and there are still words to be transferred, the track register is incremented at the end of the last sector. The read operation then resumes at the first sector of the next track. Since 200 microseconds are required to stabilize the read amplifier after a track change, the unused part at the end of the track must be at least 600 bits for an 1800 RPM disc.

For 47 words per sector the number of sectors is determined as follows:

$$\begin{array}{r} 47 \text{ words per sector} \\ \times 16 \text{ bits per word} \\ \hline 752 \\ +21 \text{ (13 error bits + 8 sector gap bits)} \\ \hline 773 \text{ bits per sector} \end{array}$$

$$\frac{100,000 \text{ sectors per track}}{773 \text{ bits per sector}} = 129 \text{ sectors per track with } 283 \text{ bits remaining}$$

Since the remainder in the example above is less than the minimum number of bits required to stabilize the read amplifier, the disc is sectored with 128 sectors of 47 words each. This allows the 129th sector and the remainder, or a total of 1056 bit times to stabilize the read amplifier.

5-9.5 Disc Functions

5-9.5.1 Disconnect (DOT 1, 0)

The disc controller is disconnected and put into an off-line state and its interrupt capability is inhibited.

5-9.5.2 Set Memory Address (DOT 1, 1)

Both the read and write operations require that the controller address the core memory location into or out of which data are to be transferred. The Set Memory Address command causes the core memory address of the first word to be transferred and loaded into the memory address counter from bit-1 through bit-15 of the accumulator. Bit-0 of the accumulator is not used. See Table 5-7.

5-9.5.3 Set Track and Sector (DOT 1, 2)

The location of data in the disc memory is defined by unit number, track number, and sector number. The Set Track and Sector command causes the starting sector number to be loaded into the first sector register and the track number to be loaded into the track number counter. The track number is contained in bit-0 through bit-5 of the accumulator. The sector number is contained in bit-7 through bit-15 of the accumulator. Bit-6 of the accumulator is not used. See Table 5-7.

5-9.5.4 Set Unit Number, Number of Words, and Write (DOT 1, 4)

This command causes the number of words that are to be transferred from core memory into disc memory to be loaded into the number of words counter, the unit number to be loaded into the disc unit selector, and a write cycle to be initiated. The word count is contained in bit-2 through bit-15 of the accumulator. The unit number is contained in bit-0 and bit-1. See Table 5-7.

Data are transferred from core memory to disc memory starting with the data from the core memory address specified by the Set Memory Address instruction. The ending core address is computed by adding the number of words to be transferred to the starting core memory less one. At the end of each section the cyclic redundancy check code is added. See Section 5-9.5.5 for description and checking.

The data transfer is handled automatically by the DMA channel. There is no need for any user intervention once the write command has been given. At the completion of the data transfer, the disc controller will interrupt the CPU. Status bit-0 and bit-1 will be set false.

5-9.5.5 Set Unit Number, Number of Words, and Read (DOT 1, 6)

This command causes the number of words that are to be transferred from disc memory into core memory to be loaded into the number of words counter, the unit number to be loaded into the disc unit selector and a read cycle to be initiated. The word count is contained in bit-2 through bit-15 of the accumulator. The unit number is contained in bit-0 and bit-1. See Table 5-7.

Data are transferred from disc memory to core memory starting with the data from the track and sector location specified by the Set Track & Sector instruction. The core memory starting address, as specified by the Set Memory Address instruction, receives the first word transferred from the disc memory. The ending core address is computed by adding the number of words to be transferred to the starting core memory address less one.

The data transfer is handled automatically by the DMA channel. There is no need for any user intervention once the read command has been given.

Simultaneously with its transfer to core memory during write, the data is applied to a cyclic encoder. The cyclic encoder is part of an error detecting loop that treats data as a binary number which it divides (module two) by the polynomial $1 + X + X^3 + X^4 + X^{13}$. The remainder produced by this operation is stored at the end of each sector in the write operation. When the sector is read the division is again performed and the remainder is compared with the stored remainder. Failure to compare causes status bit-15 to be set.

At the completion of the data transfer, the disc controller interrupts the CPU. Status bit-0 and bit-1 are set false.

5-9.5.6 Set Unit Number, Number of Words, and Verify (DOT 1, 7)

This command causes the same operation to be performed as the read operation (Ref. 5-9.5.5) except that no data is transferred into the core memory.

5-9.6 Function Codes

The various 8 bit control codes are illustrated in hexadecimal and binary formats in Table 5-28.

The disc system performs three basic operations; write, read, and verify. In the write operation data is transferred from the disc into the core memory. In the verify operation data is read from the disc and the accuracy of the read operation is verified but no data is transferred.

Each of the basic operations require that the disc unit, track, starting sector, and the number of words involved in the operation be specified. In the read and write operations the starting core-memory address must also be specified.

5-9.7 Status Codes

Status is returned to the accumulator from one of four possible disc units after issuing a DIN 1, 0; DIN 1, 1; DIN 1, 2; or a DIN 1, 3 command.

The disc status word consists of four bits (Table 5-29). An all zero status word indicates that the disc unit and controller are ready to receive a new command. Bit-0 true indicates that either the disc controller or device are not ready to receive a new command.

Bit-13 true means that the previous command (other than status) issued was a write command to a protected track. Bit-14 true means there was a rate error in the last read or write command. Bit-15 true indicates a rate error or a cyclic redundancy check (CRC) error on the previous read or verify commands (see 5-9.5.5 for CRC description).

5-9.8 Interrupt Meaning

Since the disc is a DMA type of device, there is no need for the disc controller to interrupt the CPU

for each word transfer. The interrupt from the disc signifies that the data transfer (read, write) or verification has been completed.

5-9.9 Timing

Unlike DIO devices (e.g., card reader, line printer, etc.), all interrupt times depend entirely on the number of words being transferred and the speed (Ref. Sec. 5-9.1) of the disc drive. There is no maximum service time because the disc controller will save the results (i.e., rate or CRC error, write to protected track) of the previous operation indefinitely.

5-9.10 Operation Notes

5-9.10.1 Rate Error

A rate error occurs whenever there is an interruption in the flow of data during a transfer between core memory and the disc controller. Rate errors do not normally occur and they are usually the result of a hardware malfunction. Note that a rate error causes both bit-14 and bit-15 of the status word to be true.

5-9.10.2 Track Protection

The track protect feature of the disc allows the user, by using the WRITE INHIBIT switches on the disc controller, to "lockout" or write protect selected tracks. If the program specifies one of the protected tracks with Set Track & Sector and Write commands, the protected track will not be written on and an interrupt will be returned immediately to the CPU. Status bit-13 will also be set true.

5-9.10.3 Bootstrap Load

The contents of sector 0, track 0 of disc 0 can be read into core memory, starting at location 0, by pressing the LOAD button on the disc controller operating panel.

5-9.11 Disc Operation (Model No. 74601, 74602)

Disc controller controls are listed in Table 5-30 and shown in Figure 5-12. The disc units have no operator controls or indicators.

Table 5-28. Disc Commands and Function Codes

Disc Commands	Function Codes								
	Hexadecimal Notation	Binary Notation							
		Bit Position							
DOT		8	9	10	11	12	13	14	15
Disconnect Controller	10	0	0	0	1	0	0	0	0
Set Memory Address	11	0	0	0	1	0	0	0	1
Set Track & Sector	12	0	0	0	1	0	0	1	0
Set Unit No., Number of Words and Write	14	0	0	0	1	0	1	0	0
Set Unit No., Number of Words and Read	16	0	0	0	1	0	1	1	0
Set Unit No., Number of Words and Verify	17	0	0	0	1	0	1	1	1
DIN									
Return Status of Disc Unit 0 to Accumulator	10	0	0	0	1	0	0	0	0
Return Status of Disc Unit 1 to Accumulator	11	0	0	0	1	0	0	0	1
Return Status of Disc Unit 2 to Accumulator	12	0	0	0	1	0	0	1	0
Return Status of Disc Unit 3 to Accumulator	13	0	0	0	1	0	0	1	1

Table 5-29 Disc Status

Meaning When Bit True*	Bit No.
Controller or Device Not Ready	0
Write Command Issued to Protected Track	13
Rate Error	14
Rate or CRC Error	15
*Response is for one of four possible units. Unit number must be specified in bits 12-15 of the DIN instruction for status.	

Table 5-30. Disc Controller Controls

Control	Function																				
<p>WRITE INHIBIT Switches</p> <p>PROTECTED TRACK DISC 0 - 3</p> <p>LOAD Switch</p>	<p>Protects data stored on selected tracks of corresponding disc by inhibiting writing on these tracks as follows:</p> <table border="0"> <tr> <td>OFF</td> <td></td> </tr> <tr> <td>Position</td> <td>Tracks Inhibited</td> </tr> <tr> <td>OFF</td> <td>None</td> </tr> <tr> <td>0</td> <td>0</td> </tr> <tr> <td>0-1</td> <td>0 and 1</td> </tr> <tr> <td>0-3</td> <td>0 through 3</td> </tr> <tr> <td>0-7</td> <td>0 through 7</td> </tr> <tr> <td>0-15</td> <td>0 through 15</td> </tr> <tr> <td>0-31</td> <td>0 through 31</td> </tr> <tr> <td>0-63</td> <td>0 through 63</td> </tr> </table> <p>Lloads contents of sector 0, track 0 of disc 0 into core memory starting at location 0.</p>	OFF		Position	Tracks Inhibited	OFF	None	0	0	0-1	0 and 1	0-3	0 through 3	0-7	0 through 7	0-15	0 through 15	0-31	0 through 31	0-63	0 through 63
OFF																					
Position	Tracks Inhibited																				
OFF	None																				
0	0																				
0-1	0 and 1																				
0-3	0 through 3																				
0-7	0 through 7																				
0-15	0 through 15																				
0-31	0 through 31																				
0-63	0 through 63																				

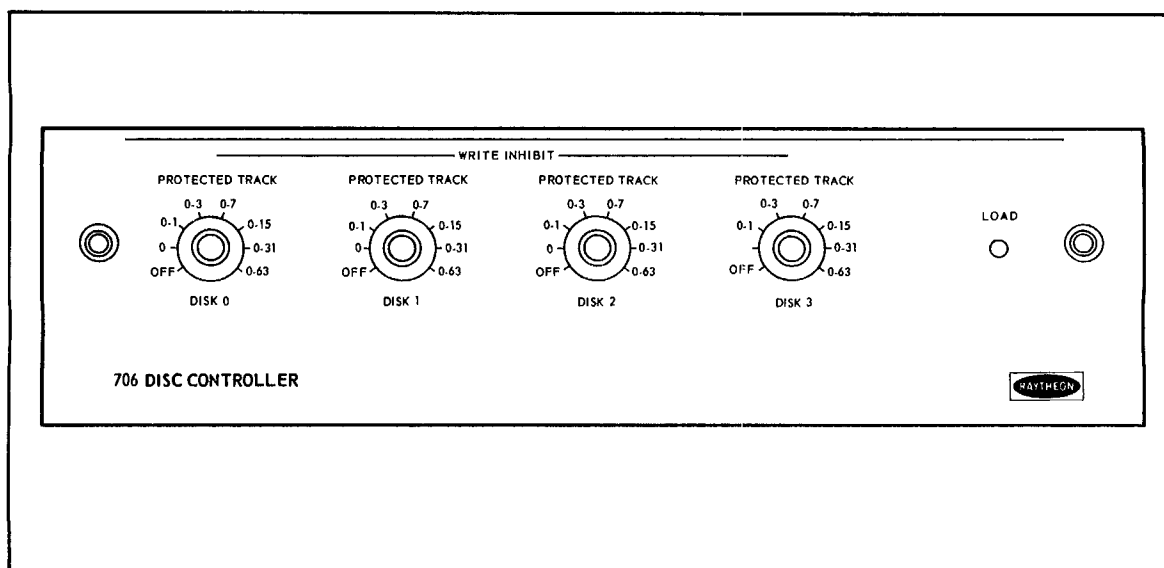


Figure 5-12. Disc Controller Controls

5-10 DMA 9-TRACK MAGNETIC TAPE

The DMA 9-track magnetic tape controller and unit(s) provide mass program and/or data storage for the 706 computer. Up to four magnetic tape drives may be attached to the magnetic tape controller. Data is transferred directly to and from memory via the Direct Memory Access (DMA Channel). Table 5-31 shows the system and unit characteristics for the 9-track magnetic tape controller, 36 ips, and 75 ips tape transports.

A data chaining option permits automatic transfer to the start of another memory buffer at the end of a buffer without stopping tape motion or inserting end-of-record gaps. The option thus facilitates maximum data transfer rates and the handling of blocks of data larger than the normal maximum record length (16,383 words; 32,766 characters).

5-10.1 Speed

Magnetic tape transports are available with tape speeds of either 36 ips or 75 ips. Both units have a packing density of 800 bpi or byte transfer rates of 28.8 kc and 60 kc respectively. Data may be transferred between the CPU and core memory at a rate of 1,110,000 words per second. Therefore, the magnetic tape requires approximately 1/80 of the core memory access time when the 36 ips unit is used. When the 75 ips unit is used it requires 1/36 of the core memory access time.

5-10.2 Data Type

Data transfer between the magnetic tape controller and DMA is full word, 16-bit binary. Data transfer between the magnetic tape controller and the tape unit is 9-bit binary (8 data bits plus parity). It takes two tape frames (characters) to equal one computer word.

5-10.3 Tape Unit Number Assignment

Each of the magnetic tape units in the system may be assigned any number from 0 through 3, or may be turned off. The UNIT SELECT switch (Table 5-35) allows the operator to make the assignment manually.

5-10.4 DMA 9-Track Magnetic Tape Functions

5-10.4.1 Disconnect (DOT 2, 0)

The disconnect instruction is used to reset the magnetic tape controller and is not addressed to a

particular tape unit. Therefore, no command word is placed into the CPU accumulator register prior to executing the instruction.

The effect of the Disconnect instruction is the same as the system reset signal received from the CPU. When the instruction is executed, all mode control, tape motion control, (except rewind) and word assembly register flip-flops are reset. The status, tape error, and interrupt flip-flops in the controller and the EOT latches in each of the switchers are also reset, and the read inhibit flip-flop is set. The memory address and word count registers are not reset because they are normally reset just prior to loading the address or word count into them.

The reset does not affect the rewind control logic, and any rewind that may be in progress when the reset instruction is executed will be completed normally. The rewind control logic for each tape unit is reset at the completion of each rewind operation.

No interrupt is returned to the CPU when the Disconnect Controller instruction is executed.

5-10.4.2 Set Memory Address (DOT 2, 1)

The Set Memory Address instruction is used to load the core memory address of the first data word to be transferred into the memory address register of the controller. In magnetic tape systems with dual-speed tape units, (special option) this instruction also sets the tape speed.

Bit-0 of the accumulator specifies low tape speed when it is true and remains false to select high speed. This bit has no significance in systems with single speed tape units. Bit-1 through Bit-15 of the accumulator specify the starting memory address.

5-10.4.3 Set Word Count and Write a Record (DOT 2, 4)

This instruction loads the number of words to be transferred from memory to magnetic tape into the word count register of the controller and then initiates a write operation. Bit-2 through bit-15 of the accumulator specify the word count, while bit-0 and bit-1 specify the tape unit number.

Table 5-31. DMA 9-Track Magnetic Tape System Characteristics

Controller		
Tape Units Controlled	1 to 4	
Operating Modes	Write forward	
	Read forward	
	Write end-of-file	
	Advance to end-of-file	
	Write skip	
	Back space	
	Rewind	
	Data chain (option)	
	Data Word (DMA bus)	16-bit binary, two 8-bit characters per word
	Tape Character	9-bit binary (8 data bits plus parity)
Tape Format	IBM 9-track, System 360/2400 compatible	
Tape Speed Selection	Single speed	
Parity Checks	Lateral and longitudinal (read and read-after-write)	
Parity Conventions	Lateral – odd	
	Longitudinal – even	
Error Checking and Correcting Code	Cyclic redundancy check character, IBM System 360/2400 compatible	
Parity Gap Length	3 characters	
Record Gap Length	0.6 in., nominal	
Word Transfer Rate	One-half transport character transfer rate	
Power Requirement	+5vdc, 6 amp	
Power Supply Input Requirements	115 vac \pm 10%, 60 cps, 1.4 amp	
Operating Temperature	0 to 40°C	
Storage Temperature	-20 to 65° C	
Relative Humidity	90% (max) without condensation)	

Table 5-31. DMA 9-Track Magnetic Tape System Characteristics (Cont.)

Tape Unit		
Tape Speed	36 ips	75 ips
Transport Model	MT36	MT75
No. of Tracks	9	9
Packing Density	800 bpi	800 bpi
Writing Mode	NRZ Mod	NRZ Mod
Character Transfer	28.8 kc	60 kc
Tape Width	½ in.	½ in.
Reel Size (Max)	10½ in.	10½ in.
Tape Speed Accuracy	2%	2%
Wos and Flutter	2%	1%
Start Time	5 ms	3 ms
Start Distance (in.)	0.100 ± 0.020	0.100 ± 0.035
Stop Time	3 ms	1.75 ms
Stop Distance (in.)	0.054 ± 0.020	0.090 ± 0.020
Skew, any 2 channels		
Static (max)	8 μsec	4 μsec
Dynamic	6 μsec	3 μsec
Start-Stop Command	5 ms	5 ms
Interval (Min)		
Rewind Time (2400 ft)	3 min	3 min
Operating Temperature	0–50°C (within tape limitations)	
Power Requirements	115 vac ± 10%, 60 cps, 6.5 amp cont. 9 amp surge	

If the write enable ring is not installed on the file reel when a write instruction is executed, the write operation will not be performed, and an interrupt will be returned to the CPU. This would constitute a programming error since absence of the write enable ring should have been detected in the status check, and this situation should not occur. If it does occur, the interrupt allows a means of program recovery.

If the unit is at the BOT marker when the operation is started, a 3.5-inch BOT gap is written before data transfer commences.

The number of words specified are transferred from memory, starting at the specified memory location, and are written onto the tape in two characters (bytes) per word. When the specified number of words has been written on tape, a three-character gap is generated, and the cyclic redundancy check (CRC) is written. After another three-character gap, the longitudinal redundancy check (LRC) character is written.

The write operation is terminated when the read-after-write electronics has detected an "end-of-block" condition. The end-of-block condition occurs fourteen blank characters after the LRC character has been read and causes the continue status flip-flop to be set and an interrupt to be sent to the CPU.

Upon processing the interrupt, the program should check status to determine if there was a rate or tape error or if the end-of-tape (EOT) marker has been sensed.

If the "continue" status bit is true, a continue command may be issued. If a continue command is not issued during the allowed period (i.e., a few milliseconds after detecting end-of-block), tape motion will be halted and the controller will initiate a 5-millisecond duty cycle delay that must expire before another write operation can be initiated. No interrupt is sent to the CPU to indicate the end of the continue period or the end of the duty cycle delay.

5-10.4.4 Set Word Count and Read a Record (DOT 2, 6)

This instruction loads the number of words to be transferred from magnetic tape to memory into the word count register of the controller and then initiates a read operation. Bit-2 through bit-15 of

the accumulator specify the word count while bit-0 and bit-1 specify the tape unit number.

If the unit is at the BOT marker when the operation is started, a 3.5-inch BOT gap is skipped before data transfer commences.

In a read operation, the number of words specified may be equal to, greater than, or less than the number of words in the record to be read. If the number of words specified is less than the number of words in the record, only that number of words will be transferred to memory, but the entire record will be read. If the number of words specified is equal to or greater than the number of words in the record, the entire record will be read into memory, and the operation will be terminated regardless of whether the word count has reached zero or not.

The read operation is terminated by reading the data on the tape and detecting an "end-of-block" condition. The end-of-block condition occurs fourteen blank characters after the LRC character has been read and causes the continue status flip-flop to be set and an interrupt to be sent to the CPU.

Upon processing the interrupt, the program should check status to determine if there was a rate or tape error or if the end-of-tape (EOT) marker has been sensed.

If the continue status bit is true, a continue command may be issued. If a continue command is not issued during the allowed period (i.e., a few milliseconds after detecting end-of-block), tape motion will be halted and the controller will initiate a 5-millisecond duty cycle delay that must expire before another read operation can be initiated. No interrupt is sent to the CPU to indicate the end of the continue period or the end of the duty cycle delay.

5-10.4.5 Continue Last Operation (DOT 2, 3)

The continue instruction enables a read or write operation to be repeated on the same tape unit without stopping tape motion and waiting for the expiration of the 5-millisecond duty cycle delay. This instruction can only be used to continue a read or a write operation.

The period during which the continue instruction may be executed is indicated by the continue

status bit (bit-10) of the status response word. However, the status response word does not indicate which of the operations was just completed, and it is necessary to keep track in the programs whether a read or write operation was just performed in order to know whether the same mode of operation should be continued or a new mode selected.

The continued read or write operation may start at the memory address remaining in the memory address register at the conclusion of the previous operation, or a new memory address may be transmitted to the controller by means of a set memory address instruction prior to executing the continue instruction.

A new word count must be placed into bit-2 through bit-15 of the accumulator prior to executing the continue instruction. The unit number is not significant in this command word because the operating mode is continued on the same tape unit.

When the continue instruction is executed, another record is read or written without stopping tape motion.

5-10.4.6 Write End-of-File (DOT 2, 5)

The write end-of-file (EOF) instruction is used to write an EOF character on tape and must be addressed to a particular tape unit by means of a command word that is placed into bit-0 and bit-1 of the accumulator. Bit-2 through bit-15 of the accumulator are not used.

When the write EOF instruction is executed, the rate error and tape error flip-flops in the controller are reset, and the tape on the addressed unit is driven forward. After a normal end-of-record gap, an EOF character is written onto the tape, followed by a seven-character gap and an LRC character. (No CRC character is written in this operation, which accounts for the single seven-character gap rather than a three-character gap, a CRC character, and another three-character gap.)

Lateral and longitudinal parity, but not the CRC, are checked during the read-after-write operation and the tape error flip-flop is set if an error is detected. Tape motion is halted when the end-of-block condition is detected (fourteen characters after the LRC character is read), but no

interrupt is generated at this time as in a write operation. After a delay (which varies with tape unit speed) to allow tape motion to stop, a 5-millisecond duty cycle delay is initiated. At the end of the duty cycle delay, an interrupt is sent to the CPU signifying that the operation has been completed.

The EOF character is normally written after one or more records have been written, and operation would not normally progress to this point if the write enable ring were not installed on the file reel. However, if the write EOF instruction is executed and is addressed to a unit that is not enabled to write, the write EOF operation will not be performed, and an interrupt will be returned to the CPU immediately.

5-10.4.7 Advance to End-of-File (DOT 2, D)

The advance to end-of-file (EOF) instruction is used to advance the tape to the next EOF mark on the tape and must be addressed to a particular tape unit by means of a command word that is placed into bit-0 and bit-1 of the accumulator. Bit-2 through bit-15 of the accumulator are not used.

When the advance to EOF instruction is executed, the rate error and tape error flip-flops are reset, and the addressed tape unit starts to read forward. No data is transferred in this operation, and the rate error flip-flop remains reset. The data on the tape is read and the tape error flip-flop will be set if a lateral parity, longitudinal parity, or CRC error is made.

At the end of each record, a check for the EOF character is made. If no EOF character is found, the next record is read. This process is repeated until an EOF character is found. If no EOF character is found on the tape, the unit will read off the end of the tape and stop due to a "tape break" being detected. The end-of-tape (EOT) status bit will be set if the EOT marker is passed, but this will not stop the EOF search operation.

When an EOF mark is detected, the EOF-character-detected status flip-flop is set, and the read operation continues until the LRC character associated with the EOF mark is read. When the end-of-block condition is detected (fourteen blank characters after the LRC character is read), tape is driven approximately 0.2 inch before the run command is terminated.

When the run command is terminated, a 5-millisecond duty cycle delay is initiated to allow the tape to stop and the tape unit to become ready for another motion command. At the end of the duty cycle delay, an interrupt is sent to the CPU signifying that the operation has been completed. The status request in response to the interrupt should check the state of the EOF character status bit (bit 11) to verify that an EOF character has been found. This status bit will remain true until a new operation is initiated.

5-10.4.8 Write Skip (DOT 2, 7)

The write skip instruction is used to advance and erase approximately 3.5 inches of tape and must be addressed to a particular tape unit by means of a command word that is placed into bit-0 and bit-1 of the accumulator. Bit-2 through bit-15 of the accumulator are not used.

When the write skip instruction is executed, the rate error and tape error flip-flops in the controller are reset, the unit is enabled to write (which turns on the erase head), and the tape is driven forward. Since no data is transmitted by this instruction, the erased tape is left blank. A write-skip delay terminates the erase operation after approximately 3.5 inches of tape have been erased.

After a delay (which varies with tape unit speed) to allow tape motion to stop, a 5-millisecond duty cycle delay is initiated. At the end of the duty cycle delay, an interrupt is sent to the CPU signifying that the operation has been completed.

The write skip instruction is normally used only to erase a section of tape during a write operation. There is therefore no provision for not performing the operation and interrupting if the write enable ring is not installed on the file reel. If the instruction is executed when the write enable ring is not installed, the tape will be moved the normal distance, but will not be erased because absence of the write enable ring disables the erase and write circuits in the record/playback amplifier.

5-10.4.9 Rewind (DOT 2, B)

The rewind instruction is used to rewind a tape to the beginning-of-tape (BOT) marker and must be addressed to a particular tape unit by means of a command word that is placed into bit-0 and bit-1 of the accumulator. Bit-2 through bit-15 of the accumulator are not used.

Separate unit select and control logic is used for the rewind operation, and execution of a rewind instruction has no effect on the remainder of the controller. One or more tape units may be rewinding at the same time that other operations are being performed on other tape units.

When the rewind instruction is executed, a rewind command is transmitted to the addressed tape unit. If the unit is already sensing the BOT marker, the rewind command is terminated immediately. If the unit is not sensing the BOT marker, a rewind operation is initiated and continues until the BOT marker is passed. The tape is then automatically stopped, loaded into the vacuum columns, and advanced at capstan speed to the BOT marker. When the BOT marker is sensed as the tape is being advanced, the rewind command from the controller is terminated. No interrupt is returned to the CPU at the completion of the operation.

Execution of the rewind instruction does not reset the rate error and tape error flip-flops and does not affect the status of the controller. The status of the tape unit will be "not ready" during the rewind and will return to "ready" when the tape stops at the BOT marker.

5-10.4.10 Read Memory Address (DIN 2, 4)

This instruction causes the current contents of the magnetic tape controller's memory address register to be returned to bit-1 through bit-15 of the accumulator. Bit-0 of the accumulator is set to zero. This instruction allows the program to determine just where in the DMA data transfer the read or write process is at any given moment.

5-10.4.11 Backspace One Record (DOT 2, C)

The back space instruction is used to back up the tape one record and must be addressed to a particular tape unit by means of a command word that is placed into bit-0 and bit-1 of the accumulator. Bit-2 through bit-15 of the accumulator are not used.

When the back space instruction is executed, the rate error and tape error flip-flops are reset and remain reset at the end of the operation. If the addressed tape unit is sensing the BOT marker the operation is terminated and an interrupt is sent to the CPU immediately. If the unit is not sensing the BOT marker, the backspace operation is performed. The manner in which the backspace

operation is performed depends on the last previous operation performed.

If this operation was write a record, write EOF, or write skip, the write-enabled condition would still exist. If the write enable were turned off and the tape backspaced from this point, a mark would be left on the tape that could cause reading errors later. For this reason, the tape is advanced and erased approximately 3.2 inches before the write enable is turned off. The mark on tape thus made will cause no trouble in subsequent read operations and will be erased if a write operation is performed over that portion of the tape.

If the tape is advanced as described above, a 5-millisecond duty cycle delay is initiated when the tape is stopped. At the end of a duty cycle delay, the reverse mode is entered, which turns off the write enable.

If write is not enabled on the addressed tape unit when the backspace instruction is executed, the reverse mode is entered directly. Operation in the reverse mode is the same if it is entered directly or after advancing and erasing the tape. The tape is driven in reverse until a character is detected (normally the LRC character of the record to be backspace over), then continues until an end-of-block condition is detected (fourteen blank characters after the first character of the record is read).

Tape motion is halted approximately 0.45 inch after the end-of-block condition is detected, and a 5-millisecond duty cycle delay is initiated. At the end of the delay, an interrupt is returned to the CPU signifying that the operation has been completed.

If no data is detected on the tape, the tape will continue to be driven in reverse until the BOT marker is sensed. The operation will then be terminated in the same manner as if an end-of-block condition had been detected, and the interrupt will be sent to the CPU at the end of the duty cycle delay.

5-10.5 Data Chain Option

The data chain option permits records longer than 16,383 words (32,766) characters) to be read or written. It also permits the same buffer in memory to be used repeatedly in continuous data acquisition or retrieval applications without

introducing inter-record gaps after each repetition.

Two data chain modes of operation are available. In ordinary data chain operation, an interrupt is returned to the CPU each time that the word count in the controller reaches zero and also at the completion of the data chain operation. In the "data chain special" mode no interrupts are returned to the CPU except at the completion of the operation.

The data chain option employs two auxiliary storage registers to contain a memory address and a word count. When the count in the controller word count register reaches zero, the contents of these auxiliary registers are automatically transferred to the memory address and word count registers, and the read or write operation continues uninterrupted. Before executing any of the data chain instructions a read or write operation must first be initiated.

5-10.4.1 Load Data Chain Address (DOT 2, 8)

The data chain address is loaded into the data chain address register from bit-1 through bit-15 of the accumulator. Bit-0 of the accumulator is not used. This is the address of the first word of the next block of data that will be read or written and may specify the same buffer in memory as the first operation or a different buffer.

5-10.5.2 Load Data Chain Mode and Word Count (DOT 2, 9)

The data chain mode is selected in the controller, and the data chain word count is loaded into the data chain word count register by means of a load data chain mode instruction. This may be the same word count used in the first operation or a different word count. Bit-2 through bit-15 of the accumulator contain the new word count. Bit-0 and bit-1 contain the mode selection code.

The first two bits of the command word (bits-0 and -1) specifying the data chain mode and are copied into two data chain mode flip-flops. Data chain mode is not selected when both mode flip-flops are reset.

Therefore, mode code 00 is an improper code and cannot be used. Code 01 specifies data chain special mode, and code 10 specifies ordinary data chain mode. Code 11 also specifies ordinary data chain mode and is redundant.

When the count in the word count register reaches zero, the contents of the data chain address register is automatically transferred to the memory address register, and the contents of the data chain word count register is transferred to the word count register. In the ordinary data chain mode, an interrupt is sent to the CPU at this time to signify that the transfer has occurred and that a new data chain address and word count may be sent to the controller. No interrupt is sent in the data chain special mode.

The contents of the data chain address and word count registers are not altered by the transfer to the memory address and word count registers. If new data is not placed into these registers before the word count reaches zero again, the same address and word count will be transferred again. This will cause the same buffer in memory to be used again. The data chain special mode is thus best adapted to reading or writing the same buffer repeatedly where no interrupts are required to transfer new addresses and word counts. The ordinary data chain mode is best adapted to reading or writing consecutive or separate buffers where the interrupt is used to transfer the address and word count of the next buffer.

5-10.5.3 Stop Data Chain (DOT 2, A)

Data chain operation will continue until stopped by a stop data chain instruction (or a reset). Failure to transmit a new data chain address and word count will not stop the operation because the previous address and word count will be used repeatedly as explained above. When the stop data chain instruction is executed, both data chain mode flip-flops are reset, and the contents of the data chain address and word count registers will not be transferred when the word count reaches zero the next time. The operation is terminated in the same manner as a normal read or write operation.

The controller is no longer in either data chain mode during the termination of the operation, and the CRC and LRC characters are read or written as in an ordinary read or write operation. The interrupt normally sent during the termination of a read or write operation is also sent at this time regardless of the previous data chain mode.

5-10.6 Function Codes and Command Words

Table 5-32 is a summary of the instruction, function and accumulator format for DMA 9-track magnetic tape.

5-10.7 Status Codes

Status is returned to the accumulator from one of four possible magnetic tape units after issuing a DIN 2, 0; DIN 2, 1; DIN 2, 2, or a DIN 2, 3 command.

The DMA 9-track magnetic tape status word consists of 11 bits (Table 5-33). An all zero status word has no significance as to the readiness of a magnetic tape unit.

Bit-0 and Bit-1 indicate if the controller and/or tape unit are busy. The tape unit is busy if it is in the process of rewinding; it is not ready if it is in manual mode, the vacuum lights are out, tape is broken or the power is off.

Bit-2 and bit-3 indicate beginning-of-tape (BOT) and end-of-tape (EOT) markers sensed respectively. The status word should be tested for EOT after each read or write operation because no indication other than the status bit is given when the EOT marker is passed; it is possible to read or write off the end of the tape if the EOT status bit is ignored. Bit-4 remains true while the selected unit is in the rewind process.

Bit-10 remains true during the period that a continue command may be issued. The length of this period depends on the speed of the tape units and varies from less than a millisecond to a few milliseconds.

When an end-of-file (EOF) mark is detected, bit-11 of the status word is set true. The EOF mark may be detected because the tape was advanced to EOF, or an EOF mark was read as a normal record via a read command.

The byte count bit may be of interest if a tape not written on the 706 is being read. Such tapes may contain an odd or even number of character; all tapes written on the 706 system have an even number of characters. Bit-12 is true when the number of characters in the record is odd.

If a write operation is to be performed, the state of bit-13 of the status response word should be

checked. If the write enable ring is not installed on the file reel, bit-13 will be false.

indicate whether a rate error (bit-14) or a tape error (bit-15) has occurred. Tape errors are lateral parity, longitudinal parity (LRC), and cyclic redundancy check (CRC) errors.

Bit-14 and bit-15 of the status response word

Table 5-32. DMA 9-Track Magnetic Tape Function Codes

Instruction	Function	Accumulator Format
DOT 2, 0	Reset controller (disconnect)	None
DOT 2, 1	Set tape speed and memory address	00 01 15 Spd Memory Address
DOT 2, 3	Continue read or write	00 01 02 15 Word Count
DOT 2, 4	Write one record	00 01 02 15 Unit Word Count
DOT 2, 5	Write end-of-file (EOF)	00 01 02 15 Unit
DOT 2, 6	Read one record	00 01 02 15 Unit Word Count
DOT 2, 7	Write skip	00 01 02 15 Unit
DOT 2, 8	Load data chain address	00 01 15 Data Chain Address
DOT 2, 9	Load data chain mode and word count	00 01 02 15 Mode Data Chain Word Count
DOT 2, A	Stop data chain	None
DOT 2, B	Rewind	00 01 02 15 Unit
DOT 2, C	Backspace	00 01 02 15 Unit
DOT 2, D	Advance to end-of-file (EOF)	00 01 02 15 Unit
DIN 2, 0	Read controller and unit 0 status	See Table 5-33
DIN 2, 1	Read controller and unit 1 status	See Table 5-33
DIN 2, 2	Read controller and unit 2 status	See Table 5-33
DIN 2, 3	Read controller and unit 3 status	See Table 5-33
DIN 2, 4	Read memory address register	00 01 15 Memory Address

Table 5-33. DMA 9-Track Magnetic Tape Status

Meaning When Bit True*	Bit No.
Controller busy or tape unit not ready	0
Tape unit not ready (rewind, manual mode, vacuum lights out, tape break, power off).	1
Tape at beginning-of-tape (BOT) marker	2
Unit has reached or passed end-of-tape (EOT) marker. (Remains true until tape rewound or backspaced past marker.)	3
Unit is in process of rewind	4
Continue command may be issued.	10
End-of-File (EOF) character has been found	11
Byte count odd	12
Write enabled (write enable ring installed)	13
Rate error on last read or write operation	14
Tape error (parity or CRC on last read or write operation)	15

*Response is for one of four possible units. Unit number must be specified in bits 12-15 of the DIN instruction for status.

5-10.8 Interrupt Meaning

Table 5-34 gives a summary of the interrupt meaning for each instruction.

5-10.9 Timing

Unlike DIO devices (e.g., card reader, line printer, etc.), all interrupt times depend entirely on the number of words being transferred, and the speed (Ref. 5-10.1) of the magnetic tape units. There is no maximum service time because the DMA magnetic tape controller will save the results (i.e., rate or tape error, etc.) of the previous operation until a new command other than status is issued.

5-10.10 Operation Notes

5-10.10.1 Rate Errors

A rate error occurs whenever there is an interruption in the flow of data during a transfer between core memory and the DMA magnetic tape controller. Rate errors do not normally occur and they are usually the result of a hardware malfunction. Note that a rate error causes bit-14 of the status word to be true.

5-10.10.2 File Protection

In order to write on magnetic tape, the write enable ring must be inserted in the back of a magnetic tape file reel before mounting the reel on the unit. Without this ring, the magnetic tape file cannot be destroyed or written over. If a write command is issued with the ring out, the controller interrupts the CPU immediately.

Table 5-34. DMA 9-Track Magnetic Tape Interrupt Meaning

Instruction	Function	Meaning
DOT 2, 0	Reset Controller	No Interrupt
DOT 2, 1	Set tape speed and memory address	No Interrupt
DOT 2, 3	Continue read or write	(See Read or Write)
DOT 2, 4	Write one record	Record written, LRC and CRC have been read
DOT 2, 5	Write end-of file	EOF written, unit stopped
DOT 2, 6	Read one record	Record read, LRC and CRC have been read
DOT 2, 7	Write skip	Tape erased, unit stopped
DOT 2, 8	Load data chain address	No interrupt
DOT 2, 9	Load data chain mode	Ordinary mode – after data block read or written Special mode—No Interrupt
DOT 2, A	Stop data chain	Record read or written, LRC and CRC have been read
DOT 2, B	Rewind	No Interrupt
DOT 2, C	Backspace	Record Backspaced, Unit Stopped
DOT 2, D	Advance to EOF	EOF detected, unit stopped

5-10.10.3 Bootstrap Load

The contents of the first record on a magnetic tape can be read into core memory by pressing the LOAD button on the magnetic tape controller. The record is stored starting at location 0. See Section 5-10.11.7.

5-10.11 DMA 9-Track Magnetic Tape Operation (Model Nos. 73691, 73692, 73693, 73694, 73695)

5-10.11.1 Controls and Indicators

The majority of the magnetic tape system controls and indicators are located on control panels at the top-front of each of the magnetic tape units. Their controls and indicators are illustrated in Figure 5-13 and their functions are given in Table 5-35.

The magnetic tape controller has one switch-indicator on the front panel. This LOAD switch is used to load one record into memory from tape unit 0 starting at memory location 0000. The maximum length of the record that may be

loaded with this switch is 8192 words (16,384 tape characters).

The indicator in the LOAD switch lights if a parity error or cyclic redundancy check error occurs during a read or write operation. It remains lit until the next tape-motion command is issued to the controller.

The record/playback amplifier power supply in each magnetic tape unit has one switch-indicator on the front panel. The switch controls application of ac power to the supply and is normally left on at all times because ac power in the magnetic tape unit is controlled by an AC controller.

Three indicators are incorporated into the power supply switch-indicator. The POWER ON segment indicates that an ac power is applied to the power supply. The -15v segment indicates that the -15v output voltage is present, and the +15v segment indicates that the +15v output voltage is present.

Table 5-35. Tape Unit Control Panel Controls and Indicators

Control/Indicator	Function
MODE Switch	<p>Selects tape unit operating mode:</p> <p>LOAD position removes ac power from vacuum blower, capstan motor, and reel motors; enables tape to be loaded.</p> <p>MANUAL position applies ac power to vacuum blower and capstan motor; enables manual control at tape unit</p> <p>AUTO position applies ac power to vacuum blower and capstan motor, enables computer control of tape unit</p>
DIRECTION Switch	<p>Selects direction of tape motion in manual mode of operation.</p>
SPEED Switch	<p>Selects tape speed in manual mode of operation:</p> <p>NORMAL SPEED position selects forward or reverse drive at capstan drive speed</p> <p>HIGH SPEED position selects forward or reverse drive at rewind speed</p>
UNLOAD Switch	<p>Used to rewind tape completely in manual mode with tape drive stopped. Pressing switch turns off vacuum blower and unloads tape from vacuum columns. After delay of 5 seconds reverse drive at rewind speed is initiated and continues as long as switch is held.</p>
LOAD POINT Switch/ BOT Indicator	<p>Used to advance tape to load point in manual mode of operation. Tape advanced at capstan drive speed as long as switch is pressed until BOT marker strip is sensed. BOT indicator lights whenever BOT marker strip is sensed.</p>

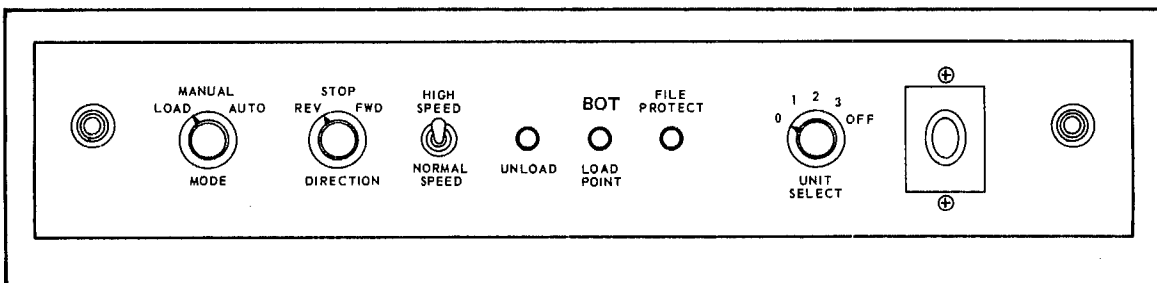


Figure 5-13. DMA 9-Track Magnetic Tape Unit Control Panel

Table 5-35. Tape Unit Control Panel Controls and Indicators

Control/Indicator	Function
FILE PROTECT Indicator	Indicates no write enable ring installed on file reel.
UNIT SELECT Switch	Selects tape unit no. (0-3) for communication with computer or disables tape unit (OFF)
UNIT SELECT Indicator	Displays unit no. selected by UNIT SELECT switch.

A circuit breaker (CB1) and a switch are located on the front panel of each ac controller in the magnetic tape system. The circuit breaker acts as the master power switch for all units in the cabinet in which the ac controller is installed. The switch allows remote control of the ac controller in the NORMAL OPERATION position and holds power on continuously in the POWER-OFF OVERRIDE position.

No operating controls or indicators are provided on the remaining magnetic tape system components.

5-10.11.2 Power ON and OFF Procedure

Application of ac power to the magnetic tape system is controlled remotely by the computer ac controller in normal operation. Circuit breaker CB1 should remain on and switch S1 should remain in the NORMAL OPERATION position on all ac controllers in the magnetic tape system except during maintenance procedures requiring power to the system to be controlled independently of computer power.

The power switch on the record/playback amplifier power supply in each magnetic tape unit should remain on at all times because ac power to the power supply is controlled by the tape unit ac controller.

5-10.11.3 Loading Tape

Note: The following instructions are for operating one magnetic tape unit. However, all magnetic tape units in the system (up to four) may be operated under program control simultaneously,

and the operating instructions for each of the units is the same. These instructions differ from those in the manufacturer's manual for the tape transport and electronic chassis because a Raytheon control panel is used in the unit rather than the control panel normally supplied for the electronic chassis.

With 706 System power on, load tape onto transport as follows:

- a. Set UNIT SELECT switch to unit number required by program: selected unit number will be displayed by adjacent digital indicator.
- b. Set speed switch to NORMAL SPEED.
- c. Set DIRECTION switch to STOP.
- d. Set MODE switch to LOAD.
- e. Install a file reel of tape on the upper reel hub and an empty reel on the lower reel hub.

NOTE: If a write operation is to be performed, install a write-enable ring on the back of the file reel before installing reel on hub. If it is desired to protect data on the tape from being lost, make sure that write-enable ring is not installed on file reel. The FILE PROTECT indicator will light when the MODE switch is set to the MANUAL or AUTO positions if no write-enable ring is installed on the reel.

f. Unwind approximately three feet of tape from file reel.

CAUTION: Handle only the end of the tape. Avoid touching the tape at any other point.

g. Insert tape under upper vacuum column fixed post and roller.

h. Insert tape between mirror and window of EOT/BOT assembly.

i. Insert tape between upper capstan and pinch roller.

j. Open head pad assembly and place tape over magnetic read/write head and upper and lower trough guides; close head pad assembly.

CAUTION: To prevent damage to ferrite shield, do not allow head pad assembly to slam closed.

k. Insert tape between lower capstan and pinch roller.

l. Place end of tape around take-up reel and wind approximately four turns of tape onto reel.

CAUTION: Do not place end of tape into slot of take-up reel or in any way fasten end of tape to reel.

m. Set MODE switch to MANUAL position; vacuum blower will start, and tape will be loaded into vacuum columns. If no write-enable ring is installed on file reel, FILE PROTECT indicator will light.

n. Press and hold LOAD POINT switch. Tape will advance until beginning-of-tape (BOT) marker is detected. Tape will halt on BOT marker and BOT indicator will light.

o. Release LOAD POINT switch when tape halts and BOT indicator lights.

p. Close glass door on transport.

CAUTION: To avoid damage to ferrite shield, make sure that head pad assembly is closed before closing glass door.

5-10.11.4 Manual Operation

With the MODE switch in the MANUAL position, the transport may be operated in the forward or reverse direction at normal (capstan) speed or high (rewind) speed as follows:

a. Set speed switch to NORMAL SPEED to select capstan speed or to HIGH SPEED to select rewind speed.

b. Set DIRECTION switch to FWD or REV as desired. If normal speed is selected, tape will be driven in the selected direction immediately. If high speed is selected, vacuum blower will be turned off and the tape unloaded from the vacuum columns. After a delay of approximately 5 seconds tape will be wound in the selected direction.

c. To halt tape motion, set DIRECTION switch to STOP.

NOTE: When operating manually in the forward direction at either speed, tape drive will continue until DIRECTION switch is set to STOP. If allowed to continue, tape will be completely removed from file reel: it will not stop when the end-of-tape (EOT) marker is sensed.

When operating manually in the reverse direction, the tape will halt on the BOT marker. If rewinding, the BOT marker will be passed, the tape will be halted and loaded into the vacuum columns, then will be driven forward to the BOT marker.

5-10.11.5 Automatic Operation

With the MODE switch set to AUTO, operation of the magnetic tape unit is controlled by the computer.

5-10.11.6 Unloading Tape

Several methods may be used to unload tape, depending on the amount of tape on the take-up reel and operator preference.

If the tape is at the BOT point, tape may be unloaded by pressing the UNLOAD switch or by setting the MODE switch to LOAD and winding the tape onto file reel manually.

When the UNLOAD switch is pressed, the vacuum blower is turned off and the tape is unloaded from the vacuum columns. After a delay of 5 seconds, the tape is driven in reverse at high (rewind) speed. Drive continues until the UNLOAD switch is released.

Setting the MODE switch to LOAD avoids the 5-second delay encountered when using the UNLOAD switch. The vacuum blower and the capstan and reel motors are turned off when the switch is set to LOAD, and the file reel may be turned manually a few turns to completely rewind the tape. This method also avoids whipping the end of the tape and throwing off oxide particles due to the reel motor driving the reel after the tape is completely rewound.

If only a moderate amount of tape is on the take-up reel, the UNLOAD switch may be pressed and held until the tape is rewound completely. This involves only one 5-second delay, but the UNLOAD switch must be released promptly when the tape is completely rewound to avoid unnecessary whipping of the tape end and oxide scatter.

If considerable tape is to be rewound, a manual mode rewind should be used to rewind the tape to the BOT marker. After the tape halts on the BOT marker, either method discussed above may be used to complete the rewind operation.

After the tape is completely rewound, remove the file reel from the upper reel hub.

5-10.11.7 Manual Memory Load

The LOAD switch on the magnetic tape controller front panel is normally used to load a bootstrap program from tape unit 0 into memory to enable loading more complex programs. When used for this purpose, the bootstrap program is normally the first record on the tape, and the tape is positioned at the load point prior to pressing the switch.

When the LOAD switch is pressed one record from the tape on unit 0 is read into memory, starting at memory location 0000. The maximum number of words that may be transferred to memory is 16,384.

The LOAD switch may also be used to transfer successive records (for instance, test programs) to memory because the next record on the tape is read each time the switch is pressed. However, each record is always written into memory starting at location 0000, so previously written records are destroyed by the new record.

5-11 DMA 7-TRACK MAGNETIC TAPE

The DMA 7-track magnetic tape controller and unit(s) provide mass data storage for the 706 computer. Up to four magnetic tape drives may be attached to the magnetic tape controller. Data is transferred directly to and from memory via the Direct Memory Access (DMA) Channel. Table 5-36 shows the system and unit characteristics for the 7-track magnetic tape controller and transport.

A data chaining option permits automatic transfer to the start of another memory buffer without stopping tape motion or inserting end-of-record gaps. The option thus facilitates maximum data transfer rates and the handling of blocks of data larger than the normal maximum record length (16,384 words; 32,768 characters).

5-11.1 Speed

The magnetic tape transport has a speed of 75 ips. The packing density is selectable on the control

panel for either 200, 556, or 800 bpi. This represents character transfer rates of 15, 41.7, and 60 kHz, respectively. Data may be transferred between the cpu and core memory at a rate of 1,110,000 words per second. Therefore, the 7-track tape unit requires as little as 1/36 of the core memory's available cycles.

5-11.2 Data Type

Data transfer between the magnetic tape controller and the DMA is full word, 16-bit binary. Data transfer between the magnetic tape controller and the tape unit is 7-bit binary (6 data bits plus parity). There are three tape formats (Figure 5-14) for transferring data between the tape unit and the controller; 3-character per word binary, 2-character per word binary, and 2-character per word decimal. All formats are selectable by the Format switch on the tape unit control panel.

Table 5-36. DMA 7-Track Magnetic Tape System Characteristics

Controller		Storage Temperature	-20 to 65° C
Tape Units Controlled	1 to 4	Relative Humidity	90% (max) without condensation
Operating Modes	Write Forward	Tape Unit	
	Read Forward	Tape Speed	75 ips
	Write end-of-file	Transport Model	MT 75
	Advance to end-of-file	No. of Tracks	7
	Write skip	Packing Density	800, 556, 200 bpi
	Backspace	Writing Mode	NRZ Mod.
	Rewind, Rewind & Lock-out	Character Transfer	60 KHZ
	Data Chain (option)	Tape Width	1/2 in.
Data Word (DMA bus)	16-bit word, 2 or 3 characters per word	Reel Size (max)	10½ in.
Tape Character	7-bit binary or decimal (6 data bits plus parity)	Tape Speed Accuracy	2%
Tape Format	IBM 7-track	Wow and Flutter	1%
Tape Speed Selection	Single speed	Start Time	3 ms.
Parity Checks	Lateral and longitudinal (read and read-after write)	Start Distance (in)	0.100 ± 0.035
		Stop Time	1.75 ms
Parity Conventions	Lateral: odd binary, even decimal Longitudinal: even	Stop Distance (in)	0.090 ± 0.020
		Skew, any 2 channels	
		Static (max)	4 u sec.
Parity Gap Length	3 characters	Dynamic	3 u sec.
		Start-Stop Command Interval (min.)	5 ms
Record Gap Length	0.75 in., nominal	Rewind Time (2400 ft.)	3 min.
Word Transfer Rate	30 KHZ maximum	Operating Temperature	0-50° C (within tape limitations)
Power Requirement	+5 vdc, 6 amp	Power Requirements	115 Vac ± 10%, 60 cps, 6.5 amp cont., 9 amps surge
Power Supply Input Requirements	115 vac ± 10%, 60 cps, 1.4 amp		
Operating Temperature	0-40° C		

5-11.2.1 3-Character/Word Binary

In this format the 16-bit word is recorded or read as three successive tape characters. The first and second characters are each six bits long (0-5, 6-11). The third character is four bits long (12-15) with track positions 1 and 2 of the third character recorded as zeros. When reading these positions are ignored. The parity bit will be such that the lateral bit total will be odd.

5-11.2.2 2-Character/Word Binary

In this format the twelve most significant bit positions are recorded or read as two successive 6-bit characters or as a 12-bit word (0-5, 6-11). The least significant four bit positions (12-15) are ignored when writing and are read as zeros. Lateral parity is odd.

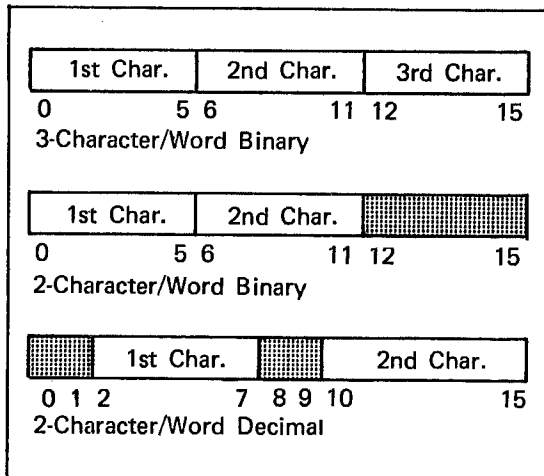
5-11.2.3 2-Character/Word Decimal

This format (BCD) records and reads the 706 word on a two character basis. Bits 2-7 and 10-15 are recorded with bits 0-1 and 8-9 being ignored. During a read operation bits 0-1 and 8-9 are set to zero. Lateral parity is even.

5-11.3 Tape Unit Number Assignment

Each of the magnetic tape units in the system may be assigned any number from 0 through 3, or may be turned off. The UNIT SELECT Switch allows the operator to make the assignment manually.

Figure 5-14. DMA 7-Track Word Formats



5-11.4 DMA 7-Track Magnetic Tape Functions

5-11.4.1 Disconnect (DOT 2,0)

The disconnect instruction is used to reset the magnetic tape controller and is not addressed to a particular tape unit. Therefore, no command word is placed into the CPU accumulator register prior to executing the instruction.

The effect of the Disconnect instruction is the same as the system reset signal received from the CPU. When the instruction is executed, all mode control, tape motion control (except rewind), and word assembly register flip-flops are reset. The status, tape error, and interrupt flip-flops in the controller and the EOT latches in each of the switches are also reset, and the read inhibit flip-flop is set. The memory address and word count registers are not reset because they are normally reset just prior to loading the address or word count into them.

The reset does not affect the rewind control logic, and any rewind that may be in progress when the reset instruction is executed will be completed normally. The rewind control logic for each tape unit is reset at the completion of each rewind operation. No interrupt is returned to the CPU when the disconnect controller instruction is executed.

5-11.4.2 Set Memory Address (DOT 2, 1)

The Set Memory Address instruction is used to load the core memory address of the first data word to be transferred into the memory address register of the controller. Bit-1 through Bit-15 of the accumulator specify the starting memory address.

5-11.4.3 Set Word Count and Write a Record (DOT 2,4)

This instruction loads the number of words to be transferred from memory to magnetic tape into the word count register of the controller and then initiates a write operation. Bit-2 through bit-15 of the accumulator specify the word count, while bit-0 and bit-1 specify the tape unit number.

If the write enable ring is not installed on the file reel when a write instruction is executed, the write operation will not be performed, and an interrupt will be returned to the CPU. This would constitute a programming error since absence of the write enable ring should have been detected in the status

check, and this situation should not occur. If it does occur, the interrupt allows a means of program recovery.

If the unit is at the BOT marker when a write operation is started, a 3.0 inch BOT gap is written before data transfer commences.

The number of words specified are transferred from memory, starting at the specified memory location, and are written onto the tape in two or three characters (bytes) per word. When the specified number of words has been written on tape, a three-character gap is generated, and the longitudinal redundancy check (LRC) character is written.

The write operation is terminated when the read-after-write electronics has detected an "end-of-block" condition. The end-of-block condition occurs after the LRC character has been read and causes the continue status flip-flop to be set and an interrupt to be sent to the CPU.

Upon processing the interrupt, the program should check status to determine if there was a rate or tape error or if the end-of-tape (EOT) marker has been sensed.

If the "continue" status bit is true, a continue command may be issued. If a continue command is not issued during the allowed period (i.e., a few milliseconds after detecting end-of-block) tape motion will be halted and the controller will initiate a 5-millisecond duty cycle delay that must expire before another operation can be initiated. No interrupt is sent to the CPU to indicate the end of the continue period or the end of the duty cycle delay.

5-11.4.4 Set Word Count and Read a Record (DOT 2, 6)

This instruction loads the number of words to be transferred from magnetic tape to memory into the word count register of the controller and then initiates a read operation. Bit-2 through bit-15 of the accumulator specify the word count while bit-0 and bit-1 specify the tape unit number.

If the unit is at the BOT marker when the operation is started, a 3.0 inch BOT gap is skipped before data transfer commences.

In a read operation, the number of words specified may be equal to, greater than, or less than the number of words in the record to be read. If the number of words specified is less than the number of words in the record, only that number of words will be transferred to memory, but the entire record will be read. If the number of words specified is equal to or greater than the number of words in the record, the entire record will be read into memory, and the operation will be terminated regardless of whether the word count has reached zero or not.

The read operation is terminated by reading the data on the tape and detecting an "end-of-block" condition. The end-of-block condition occurs after the LRC character has been read and causes the continue status flip-flop to be set and an interrupt to be sent to the CPU.

Upon processing the interrupt, the program should check status to determine if there was a rate or tape error or if the end-of-tape (EOT) marker has been sensed.

If the continue status bit is true, a continue command may be issued. If a continue command is not issued during the allowed period (i.e., a few milliseconds after detecting end-of-block), tape motion will be halted and the controller will initiate a 5-millisecond duty cycle delay that must expire before another read operation can be initiated. No interrupt is sent to the CPU to indicate the end of the continue period for the end of the duty cycle delay.

5-11.4.5 Continue Last Operation (DOT 2, 3)

The continue instruction enables a read or write operation to be repeated on the same tape unit without stopping tape motion and waiting for the expiration of the 5-millisecond duty cycle delay. This instruction can only be used to continue a read or a write operation.

The period during which the continue instruction may be executed is indicated by the continue status bit (bit-10) of the status response word. However, the status response word does not indicate which of the operations was just completed, and it is necessary to keep track in the program whether a write operation was just performed in order to know whether the same mode of operation should be continued or a new mode selected.

The continued read or write operation may start at the memory address remaining in the memory address register at the conclusion of the previous operation, or a new memory address may be transmitted to the controller by means of a set memory address instruction prior to executing the continue instruction.

A new word count must be placed into bit-2 through bit-15 of the accumulator prior to executing the continue instruction. The unit number is not significant in this command word because the operating mode is continued on the same tape unit.

When the continue instruction is executed, another record is read or written without stopping tape motion.

5-11.4.6 Write End-of-File (DOT 2, 5)

The write end-of-file (EOF) instruction is used to write an EOF character on tape and must be addressed to a particular tape unit by means of a command word that is placed into bit-0 and bit-1 of the accumulator. Bit-2 through bit-15 of the accumulator are not used.

When the write EOF instruction is executed, the rate error and tape error flip-flops in the controller are reset, and the tape on the addressed unit is driven forward. After a normal end-of-record gap, an EOF character is written onto the tape, followed by a three-character gap and an LRC character.

Lateral and longitudinal parity are checked during the read-after-write operation and the tape error flip-flop is set if an error is detected. Tape motion is halted when the end-of-block condition is detected (seven character times after the LRC character is read), but no interrupt is generated at this time as in a write operation. After a delay to allow tape motion to stop, a 5-millisecond duty cycle delay is initiated. At the end of the duty cycle delay, an interrupt is sent to the CPU signifying that the operation has been completed.

The EOF character is normally written after one or more records have been written, and operation would not normally progress to this point if the write enable ring were not installed on the file reel. However, if the write EOF instruction is executed and is addressed to a unit that is not enabled to write, the EOF operation will not be performed, and an interrupt will be returned to the CPU immediately.

5-11.4.7 Advance to End-of-File (DOT 2, D)

The advance to end-of-file (EOF) instruction is used to advance the tape to the next EOF mark on the tape and must be addressed to a particular tape unit by means of a command word that is placed into bit-0 and bit-1 of the accumulator. Bit-2 through bit-15 of the accumulator are not used.

When the advance to EOF instruction is executed, the rate error and tape error flip-flops are reset, and the addressed tape unit starts to read forward. No data is transferred in this operation, and the tape error flip-flop will be set if a lateral parity or longitudinal parity error is made.

At the end of each record, a check for the EOF character is made. If no EOF character is found, the next record is read. This process is repeated until an EOF character is found. If no EOF character is found on the tape, the unit will read off the end of the tape and stop due to a "tape break" being detected. The end-of-tape (EOT) status bit will be set if the EOT marker is passed, but this will not stop the EOF search operation.

When an EOF mark is detected, the EOF-character-detected status flip-flop is set, and the read operation continues until the LRC character associated with the EOF mark is read. When the end-of-block condition is detected (fourteen blank characters after the LRC character is read), tape is driven approximately 0.4 inch before the run command is terminated.

When the run command is terminated, a 5-millisecond duty cycle delay is initiated to allow the tape to stop and the tape unit to become ready for another motion command. At the end of the duty cycle delay, an interrupt is sent to the CPU signifying that the operation has been completed. The status request in response to the interrupt should check the state of the EOF character status bit (bit 11) to verify that an EOF character has been found. This status bit will remain true until a new operation is initiated.

5-11.4.8 Write Skip (DOT 2, 7)

The write skip instruction is used to advance and erase approximately 3.8 inches of tape and must be addressed to a particular tape unit by means of a command word that is placed into bit-0 and bit-1 of the accumulator. Bit-2 through bit-15 of the accumulator are not used.

When the write skip instruction is executed, the rate error and tape error flip-flops in the controller are reset, the unit is enabled to write (which turns on the erase head), and the tape is driven forward. Since no data is transmitted by this instruction, the erased tape is left blank. A write-skip delay terminates the erase operation after approximately 3.8 inches of tape have been erased.

After a delay to allow tape motion to stop, a 5-millisecond duty cycle delay is initiated. At the end of the duty cycle delay, an interrupt is sent to the CPU signifying that the operation has been completed.

The write skip instruction is normally used only to erase a section of tape during a write operation. If the instruction is executed when the write enable ring is not installed, the tape will not be moved and an interrupt will be generated immediately.

5-11.4.9 Rewind (DOT 2, B)

The rewind instruction is used to rewind a tape to the beginning-of-tape (BOT) marker and must be addressed to a particular tape unit by means of a command word that is placed into bit-0 and bit-1 of the accumulator. Bit-2 through bit-15 of the accumulator are not used.

Separate unit select and control logic is used for the rewind operation, and execution of a rewind instruction has no effect on the remainder of the controller. One or more tape units may be rewinding at the same time that other operations are being performed on other tape units.

When the rewind instruction is executed, a rewind command is transmitted to the addressed tape unit. If the unit is already sensing the BOT marker, the rewind command is terminated immediately. If the unit is not sensing the BOT marker, a rewind operation is initiated and continues until the BOT marker is passed. The tape is then automatically stopped, loaded into the vacuum columns, and advanced at capstan speed to the BOT marker. When the BOT marker is sensed as the tape is being advanced, the rewind command from the controller is terminated. No interrupt is returned to the CPU at the completion of the operation.

Execution of the rewind instruction does not reset the rate error and tape error flip-flops and does not affect the status of the controller. The status of the tape unit will be "not ready" during the rewind and will return to "ready" when the tape stops at the BOT marker.

5-11.4.10 Rewind and Lockout (DOT 2, 2)

The rewind and lockout instruction is used to rewind a tape to the beginning-of-tape (BOT) marker and to provide a measure of safety in preventing erroneous destruction of the written data.

Operation of the instruction is identical to the rewind operation described in section 5-11.4.9 except that the unit will not return to "ready" when the tape is repositioned at the BOT marker. The operator must return the unit to MANUAL before the operation can be completed.

5-11.4.11 Read Memory Address (DIN 2, 4)

This instruction causes the current contents of the magnetic tape controller's memory address register to be returned to bit-1 through bit-15 of the accumulator. Bit-0 of the accumulator is set to zero. This instruction allows the program to determine just where in the DMA data transfer the read or write process is at any given moment.

5-11.4.12 Backspace One Record (DOT 2, C)

The backspace instruction is used to back up the tape one record and must be addressed to a particular tape unit by means of a command word that is placed into bit-0 and bit-1 of the accumulator. Bit-2 through bit-15 of the accumulator are not used.

When the backspace instruction is executed, the rate error and tape error flip-flops are reset and remain reset at the end of the operation. If the addressed tape unit is sensing the BOT marker the operation is terminated and an interrupt is sent to the CPU immediately. If the unit is not sensing the BOT marker, the backspace operation is performed. The manner in which the backspace operation is performed depends on the last previous operation performed.

If this operation was write a record, write EOF, or write skip, the write-enabled condition would still exist. If the write enable were turned off and the tape backspaced from this point, a mark would be left on the tape that could cause reading errors later. For this reason, the tape is advanced and erased approximately 3.8 inches before the write enable is turned off. The mark on tape thus made will cause no trouble in subsequent read operations and will be erased if a write operation is performed over that portion of the tape.

When the tape is advanced as described above, a 5-

millisecond duty cycle delay is initiated when the tape is stopped. At the end of a duty cycle delay, the reverse mode is entered, which turns off the write enable. If write is not enabled on the addressed tape unit when the backspace instruction is executed, the reverse mode is entered directly. Operation in the reverse mode is the same if it is entered directly or after advancing and erasing the tape. The tape is driven in reverse until a character is detected (normally the LRC character of the record to be backspaced over), then continues until an end-of-block condition is detected (seven blank characters after the first character of the record is read).

Tape motion is halted approximately 0.65 inch after the end-of-block condition is detected, and a 5-millisecond duty cycle delay is initiated. At the end of the delay, an interrupt is returned to the CPU signifying that the operation has been completed.

If no data is detected on the tape, the tape will continue to be driven in reverse until the BOT marker is sensed. The operation will then be terminated in the same manner as if an end-of-block condition had been detected, and the interrupt will be sent to the CPU at the end of the duty cycle delay.

5-11.5 Data Chain Option

The data chain option permits records longer than 16,384 words (32,768 characters) to be read or written. It also permits the same buffer in memory to be used repeatedly in continuous data acquisition or retrieval applications without introducing inter-record gaps after each repetition.

Two data chain modes of operation are available. In ordinary data chain operation, an interrupt is returned to the CPU each time that the word count in the controller reaches zero and also at the completion of the data chain operation. In the "data chain special" mode no interrupts are returned to the CPU except at the completion of the operation.

The data chain option employs two auxiliary storage registers to contain a memory address and a word count. When the count in the controller word count register reaches zero, the contents of these auxiliary registers are automatically transferred to the memory address and word count registers, and the read or write operation continues uninterrupted. Before executing any of the data chain instructions, a read or write operation must first be initiated.

5-11.5.1 Load Data Chain Address (DOT 2, 8)

The data chain address is loaded into the data chain address register from bit-1 through bit-15 of the accumulator. Bit-0 of the accumulator is not used. This is the core address of the first word of the next block of data that will be read or written and may specify the same buffer in memory as the first operation or a different buffer.

5-11.5.2 Load Data Chain Mode and Word Count (DOT 2, 9)

The data chain mode is selected in the controller, and the data chain word count is loaded into the data chain word count register by means of a load data chain mode instruction. This may be the same word count used in the first operation or a different word count. Bit-2 through bit-15 of the accumulator contain the new word count. Bit-0 and bit-1 contain the mode selection code.

The first two bits of the command word (bits 00 and 01) specifying the data chain mode and are copied into two data chain mode flip-flops. Data chain mode is not selected when both mode flip-flops are reset. Therefore, mode code 00 is an improper code and cannot be used. Code 01 specifies data chain special mode, and code 10 specifies ordinary data chain mode. Code 11 also specifies ordinary data chain mode and is redundant.

When the count in the word count register reaches zero, the contents of the data chain address register is automatically transferred to the memory address register, and the contents of the data chain word count register is transferred to the word count register. In the ordinary data chain mode, an interrupt is sent to the CPU at this time to signify that the transfer has occurred and that a new data chain address and word count may be sent to the controller. No interrupt is sent in the data chain special mode.

The contents of the data chain address and word count registers are not altered by the transfer to the memory address and word count registers. If new data is not placed into these registers before the word count reaches zero again, the same address and word count will be transferred again. This will cause the same buffer in memory to be used again. The data chain special mode is thus best adapted to reading or writing the same buffer repeatedly where no interrupts are required to transfer new addresses and word counts. The ordinary data chain mode is best adapted to reading or

writing consecutive or separate buffers where the interrupt is used to transfer the address and word count of the next buffer.

5-11.5.3 Stop Data Chain (DOT 2, A)

Data chain operation will continue until stopped by a stop data chain instruction (or a reset). Failure to transmit a new data chain address and word count will not stop the operation because the previous address and word count will be used repeatedly as explained above. When the stop data chain instruction is executed, both data chain mode flip-flops are reset, and the contents of the data chain address and word count registers will not be transferred when the word count reaches zero the next time. The operation is terminated in the same manner as a normal read or write operation.

The controller is no longer in either data chain mode during the termination of the operation, and the data and LRC characters are read or written as in an ordinary read or write operation. The interrupt normally sent during the termination of a read or write operation is also sent at this time regardless of the previous data chain mode.

5-11.6 Function Codes and Command Words

Table 5-37 is a summary of the instruction, function and accumulator format for DMA 7-track magnetic tape.

5-11.7 Status Codes

Status is returned to the accumulator from one of four possible magnetic tape units after issuing a DIN 2, 0; DIN 2, 1; DIN 2, 2, or a DIN 2, 3 command.

The DMA 7-track magnetic tape status word consists of 16 bits (Table 5-38). An all zero status word has no significance as to the readiness of a magnetic tape unit.

Bit-0 and Bit-1 indicate if the controller and/or tape unit are busy. The tape unit is busy if it is in the process of rewinding; it is not ready if it is in manual mode, the vacuum lights are out, tape is broken or the power is off.

Bit-2 and bit-3 indicate beginning-of-tape (BOT) and end-of-tape (EOT) markers sensed respectively. The status word should be tested for EOT after each read or write operation because no indication other than the status bit is given when the EOT

marker is passed; it is possible to read or write off the end of the tape if the EOT status bit is ignored. Bit-4 remains true while the selected unit is in the rewind process.

Bit-5 and Bit-6 indicate if either 200 bpi or 556 bpi density is selected respectively. If both bit-5 and bit-6 are false, 800 bpi is selected.

Bit-7, bit-8, and bit-9 indicate the setting of the format switch. Bit-7 means 3-character/word binary format is selected. Bit-8 means 2-character/word decimal is selected. Bit-9 means 2-character/word binary is selected.

Bit-10 remains true during the period that a continue command may be issued. The length of this period is approximately 1 millisecond.

When an end-of-file (EOF) mark is detected, bit-11 of the status word is set true. The EOF mark may be detected because the tape was advanced to EOF, or an EOF mark was read as a normal record via a read command.

The byte count bit may be of interest if a tape not written on the 706 is being read. Such tapes may contain an odd or even number of characters; all tapes written on the 706 system have an even number of characters.

Bit-12 is true when the number of characters in the record is odd.

If a write operation is to be performed, the state of bit-13 of the status response word should be checked. If the write enable ring is not installed on the file reel, bit-13 will be false.

Bit-14 and bit-15 of the status response word indicate whether a rate error (bit-14) or a tape error (bit-15) has occurred. Tape errors are lateral parity and longitudinal parity (LRC) errors.

5-11.8 Interrupt Meaning

Table 5-39 gives a summary of the interrupt meaning for each instructions.

5-11.9 Timing

Unlike DIO devices (e.g., card reader, line printer, etc.), all interrupt times depend entirely on the number of words being transferred, and the speed (Ref. 5-11.1) of the magnetic tape units. There is no maximum service time because the DMA magnetic

Table 5-37 DMA 7-Track Magnetic Tape Function Codes

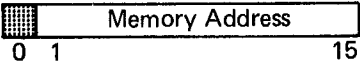

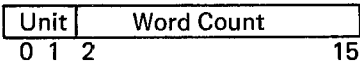
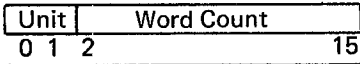

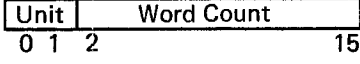

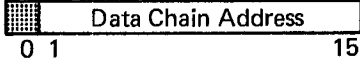
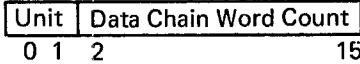



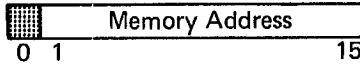
Instruction	Function	Accumulator Format
DOT 2, 0	Reset controller (disconnect)	None
DOT 2, 1	Set memory address	
DOT 2, 2	Rewind and lockout	
DOT 2, 3	Continue read or write	
DOT 2, 4	Write one record	
DOT 2, 5	Write end-of-file (EOF)	
DOT 2, 6	Read one record	
DOT 2, 7	Write skip	
DOT 2, 8	Load data chain address	
DOT 2, 9	Load data chain mode and word count	
DOT 2, A	Stop data chain	None
DOT 2, B	Rewind	
DOT 2, C	Backspace	
DOT 2, D	Advance to end-of-file (EOF)	
DIN 2, 0	Read controller and unit 0 status	See Table 5-38
DIN 2, 1	Read controller and unit 1 status	See Table 5-38
DIN 2, 2	Read controller and unit 2 status	See Table 5-38
DIN 2, 3	Read controller and unit 3 status	See Table 5-38
DIN 2, 4	Read memory address register	

Table 5-38. DMA 7-Track Magnetic Tape Status

Meaning When Bit True *	Bit No.
Controller busy or tape unit not ready	0
Tape unit not ready (rewind, manual mode, vacuum lights out, tape break, power off)	1
Tape at beginning-of-tape (BOT) marker	2
Unit has reached or passed end-of-tape (EOT) marker. (Remains true until tape rewound or backspaced past marker).	3
Unit is in process of rewind or locked out	4
Density selected is 200 bpi	5
Density selected is 556 bpi	6
Format Switch is 1. (3-character/word binary)	7
Format Switch is 2. (2-character/word decimal)	8
Format Switch is 3. (2-character/word binary)	9
Continue command may be issued	10
End-of-file (EOF) character has been found	11
Byte count odd	12
Write enabled (write enable ring installed)	13
Rate error on last read or write operation	14
Tape error. (Parity error on last read or write operation).	15

*Response is for one of four possible units. Unit number must be specified in bits 12-15 of the DIN instruction for status.

tape controller will save the results (i.e., rate or tape error, etc.) of the previous operation until a new command other than status is issued.

5-11.10 Operation Notes

5-11.10.1 Rate Errors

A rate error occurs whenever there is an interruption in the flow of data during a transfer between core memory and the DMA magnetic tape controller.

Note that a rate error causes bit-14 of the status word to be true.

5-11.10.2 File Protection

In order to write on magnetic tape, the write enable ring must be inserted in the back of a magnetic tape file reel before mounting the reel on the unit. Without this ring, the magnetic tape file cannot be destroyed or written over. If a write command is issued with the ring out, the controller interrupts the CPU immediately.

5-11.10.3 Bootstrap Load

The contents of the first record on a magnetic tape can be read into core memory by pressing the LOAD button on the magnetic tape controller. The record is stored starting at location 0. See Section 5-11.11.7.

5-11.11 DMA 7-Track Magnetic Tape Operation (Model Nos. 73671, 73672, 73673)

5-11.11.1 Controls and Indicators

The majority of the magnetic tape system controls and indicators are located on control panels at the top-front of each of the magnetic tape units. These controls and indicators are illustrated in Figure 5-15 and their functions are given in Table 5-40.

The magnetic tape controller has one switch-indicator on the front panel. This LOAD switch is used to load one record into memory from tape unit 0 starting at memory location 0000. The maximum length of the record that may be loaded with this switch is 16,384 words (32,768 tape characters).

Table 5-39 DMA 7-Track Magnetic Tape Interrupt Meaning

Instruction	Function	Meaning
DOT 2, 0	Reset Controller	No Interrupt
DOT 2, 1	Set Memory Address	No Interrupt
DOT 2, 2	Rewind & Lockout	No Interrupt
DOT 2, 3	Continue read or write	(See Read or Write)
DOT 2, 4	Write one record	Record written, LRC has been read
DOT 2, 5	Write end-of-file	EOF written, unit stopped
DOT 2, 6	Read one record	Record read, LRC has been read
DOT 2, 7	Write skip	Tape erased, unit stopped
DOT 2, 8	Load data chain address	No Interrupt
DOT 2, 9	Load data chain mode	Ordinary mode - after data block read or written. Special mode-No Interrupt
DOT 2, A	Stop data chain	Record read or written, LRC has been read
DOT 2, B	Rewind	No Interrupt
DOT 2, C	Backspace	Record Backspaced, Unit Stopped
DOT 2, D	Advance to EOF	EOF detected, unit stopped.

The record/playback amplifier power supply in each magnetic tape unit has one switch-indicator on the front panel. The switch controls application of ac power to the supply and is normally left on at all times because ac power in the magnetic tape unit is controlled by an ac controller.

Three indicators are incorporated into the power supply switch-indicator. The POWER ON segment indicates that an ac power is applied to the power supply. The -15 v segment indicates that the -15 v output voltage is present, and the +15 v segment indicates that the +15 v output voltage is present.

A circuit breaker (CB1) and a switch are located on the front panel of each ac controller in the magnetic tape system. The circuit breaker acts as the master power switch for all units in the cabinet in which the ac controller is installed. The switch allows remote control of the ac controller in the NORMAL OPERATION position and holds power on continuously in the POWER-OFF OVERRIDE position.

No operating controls or indicators are provided on the remaining magnetic tape system components.

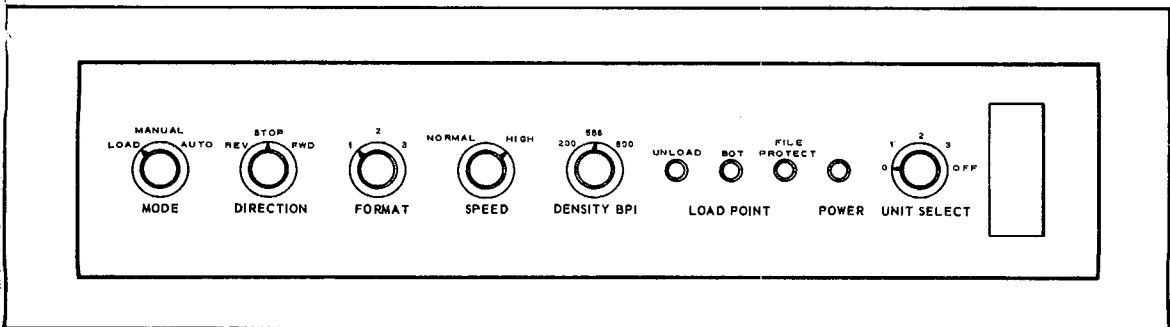


Figure 5-15. DMA 7-Track Magnetic Tape Unit Control Panel

Table 5-40. Tape Unit Control Panel Controls and Indicators

Control/Indicator	Function	Control/Indicator	Functions
MODE Switch	<p>Selects tape unit operating mode: LOAD position removes ac power from vacuum blower, capstan motor, and reel motors, enables tape to be loaded.</p> <p>MANUAL position applies ac power to vacuum blower and capstan motor; enables manual control at tape unit.</p> <p>AUTO position applies ac power to vacuum blower and capstan motor, enables computer control of tape unit.</p>	LOAD POINT Switch/BOT Indicator	Used to advance tape to load point in manual mode of operation. Tape advanced at capstan drive speed as long as switch is pressed until BOT marker strip is sensed. BOT indicator lights whenever BOT marker strip is sensed.
DIRECTION Switch	Selects direction of tape motion in manual mode of operation.	FILE PROTECT Indicator	Indicates no write enable ring installed on file reel.
SPEED Switch	<p>Selects tape speed in manual mode of operation: NORMAL SPEED position selects forward or reverse drive at capstan drive speed.</p> <p>HIGH SPEED position selects forward or reverse drive at rewind speed.</p>	UNIT SELECT Switch	Selects tape unit no. (0-3) for communication with computer or disables tape unit (OFF).
UNLOAD Switch	Used to rewind tape completely in manual mode with tape drive stopped. Pressing switch turns off vacuum blower and unloadstape from vacuum columns. After delay of 5 seconds reverse drive at rewind speed is initiated and continues as long as switch is held.	Unit Select Indicator	Displays unit no. selected by UNIT SELECT switch.
		POWER ON-OFF Switch	Controls primary AC to tape cabinet.
		FORMAT 1, 2, 3, Switch	Selects tape format as: 1. 3-character/word binary. 2. 2-character/word decimal. 3. 2-character/word binary.
		DENSITY 800, 555, 200 Switch	Selects tape packing density in bits per inch.

5-11.11.2 Power ON and OFF Procedure

Application of ac power to the magnetic tape system is controlled by the tape control panel in normal operation. Circuit breaker CB1 should remain on and switch S1 should remain in the NORMAL OPERATION position on all ac controllers in the magnetic tape system except during maintenance procedures requiring power to the system to be controlled independently of computer power.

The power switch on the record/playback amplifier power supply in each magnetic tape unit should remain on at all times because ac power to the power supply is controlled by the tape unit ac controller.

5-11.11.3 Loading Tape

NOTE: The following instructions are for operating one magnetic tape unit. However, all magnetic tape units in the system (up to four) may be operated under program control concurrently, and the operating instructions for each of the units is the same. These instructions differ from those in the manufacturer's manual for the tape transport and electronic chassis because a Raytheon control panel is used in the unit rather than the control panel normally supplied for the electronic chassis.

With 706 System power on, load tape onto transport as follows:

- a. Set UNIT SELECT switch to unit number required by program: selected unit number will be displayed by adjacent digital indicator. Set FORMAT switch and DENSITY switch.
- b. Set speed switch to NORMAL SPEED.
- c. Set DIRECTION switch to STOP.
- d. Set MODE switch to LOAD.
- e. Open glass door on transport.
- f. Install a file reel of tape on the upper reel hub.

NOTE: If a write operation is to be performed, install a write-enable ring on the back of the file reel. If it is desired to protect data on the tape from being lost, make sure that write-enable ring is not installed on file reel. The FILE PROTECT indicator will light when the MODE switch is set to the MANUAL or AUTO positions if no write-enable ring is installed on the reel.

- g. Unwind approximately three feet of tape from file reel.

CAUTION: Handle only the end of the tape. Avoid touching the tape at any other point.

- h. Insert tape under upper vacuum column fixed post and roller.
- i. Insert tape between mirror and window of EOT/BOT assembly.
- j. Insert tape between upper capstan and pinch roller.
- k. Open head pad assembly and place tape over magnetic read/write head and upper and lower trough guides; close head pad assembly.

CAUTION: To prevent damage to ferrite shield, do not allow head pad assembly to slam closed.

- l. Insert tape between lower capstan and pinch roller.
- m. Place end of tape around take-up reel and wind approximately four clockwise turns of tape onto reel.

CAUTION: Do not place end of tape into slot of take-up reel or in any way fasten end of tape to reel.

- n. Set MODE switch to MANUAL position; vacuum blower will start, and tape will be loaded into vacuum columns. If no write-enable ring is installed on file reel, FILE PROTECT indicator will light.
- o. Press and hold LOAD POINT switch. Tape will advance until beginning-of-tape (BOT) marker is detected. Tape will halt on BOT marker and BOT indicator will light.
- p. Release LOAD POINT switch when tape halts and BOT indicator lights.
- q. Close glass door on transport.

CAUTION: To avoid damage to ferrite shield, make sure that head pad assembly is closed before closing glass door.

5-11.11.4 Manual Operation

With the MODE switch in the MANUAL position, the transport may be operated in the forward or reverse direction at normal (capstan) speed or high (rewind) speed as follows:

- a. Set speed switch to NORMAL SPEED to select capstan speed or to HIGH SPEED to select rewind speed.
- b. Set DIRECTION switch to FWD or REV as desired. If normal speed is selected, tape will be driven in the selected direction immediately. If high speed is selected, vacuum blower will be turned off and the tape unloaded from the vacuum columns; after a delay of approximately 5

seconds, tape will be wound in the selected direction.

- c. To halt tape motion, set DIRECTION switch to STOP.

NOTE: When operating manually in the forward direction at either speed, tape drive will continue until DIRECTION switch is set to STOP. If allowed to continue, tape will be completely removed from file reel: it will not stop when the end-of-tape (EOT) marker is sensed.

When operating manually in the reverse direction, the tape will halt on the BOT marker. If rewinding, the BOT marker will be passed, the tape will be halted and loaded into the vacuum columns, then will be driven forward to the BOT marker.

5-11.11.5 Automatic Operation

With the MODE switch set to AUTO, operation of the magnetic tape unit is controlled by the computer.

5-11.11.6 Unloading Tape

Several methods may be used to unload tape, depending on the amount of tape on the take-up reel and operator preference.

If the tape is at the BOT point, tape may be unloaded by pressing the UNLOAD switch or by setting the MODE switch to LOAD and winding the tape onto file reel manually.

When the UNLOAD switch is pressed, the vacuum blower is turned off and the tape is unloaded from the vacuum columns. After a delay of 5 seconds, the tape is driven in reverse at high (rewind) speed. Drive continues until the UNLOAD switch is released.

Setting the MODE switch to LOAD avoids the 5-second delay encountered when using the UNLOAD switch. The vacuum blower and the capstan and reel motors are turned off when the switch is set to LOAD, and the file reel may be turned manually a few turns to completely rewind the tape. This method also avoids whipping the end of the tape and throwing off oxide particles due to the reel

motor driving the reel after the tape is completely rewound.

If only a moderate amount of tape is on the take-up reel, the UNLOAD switch may be pressed and held until the tape is rewound completely. This involves only one 5-second delay, but the UNLOAD switch must be released promptly when the tape is completely rewound to avoid unnecessary whipping of the tape end and oxide scatter.

If considerable tape is to be rewound, a manual mode rewind should be used to rewind the tape to the BOT marker. After the tape halts on the BOT marker, either method discussed above may be used to complete the rewind operation.

After the tape is completely rewound, remove the file reel from the upper reel hub.

5-11.11.7 Manual Memory Load

The LOAD switch on the magnetic tape controller front panel is normally used to bootstrap a loader program from tape unit 0 into memory to enable loading more complex programs. When used for this purpose, the loader program is normally the first record on the tape, and the tape is positioned at the load point prior to pressing the switch.

When the LOAD switch is pressed one record from the tape on unit 0 is read into memory, starting at memory location 0000. The maximum number of words that may be transferred to memory is 16,384.

The LOAD switch may also be used to transfer successive records (for instance, test programs) to memory because the next record on the tape is read each time the switch is pressed. However, each record is always written into memory starting at location 0000, so previously written records are destroyed by the new record.

CAUTION

No two tape units may have the same unit assignment number even if one of the units is off-line and is powered down. Unpredictable operation will occur if units have not been uniquely defined.

Table 5-40. Tape Unit Control Panel Controls and Indicators

Control/Indicator	Functions	Control/Indicator	Functions
MODE Switch	<p>Selects tape unit operating mode:</p> <p>LOAD position removes ac power from vacuum blower, capstan motor, and reel motors; enables tape to be loaded.</p> <p>MANUAL position applies ac power to vacuum blower and capstan motor; enables manual control at tape unit.</p> <p>AUTO position applies ac power to vacuum blower and capstan motor, enables computer control of tape unit.</p>		<p>vacuum blower and unloads tape from vacuum columns. After delay of 5 seconds reverse drive at rewind speed is initiated and continues as long as switch is held.</p>
DIRECTION Switch	<p>Selects direction of tape motion in manual mode of operation.</p>	LOAD POINT Switch/BOT Indicator	<p>Used to advance tape to load point in manual mode of operation. Tape advanced at capstan drive speed as long as switch is pressed until BOT marker strip is sensed. BOT indicator lights whenever BOT marker strip is sensed.</p>
SPEED Switch	<p>Selects tape speed in manual mode of operation:</p> <p>NORMAL SPEED position selects forward or reverse drive at capstan drive speed.</p> <p>HIGH SPEED position selects forward or reverse drive at rewind speed.</p>	FILE PROTECT Indicator	<p>Indicates no write enable ring installed on file reel.</p>
		UNIT SELECT Switch	<p>Selects tape unit No. (0-3) for communication with computer or disables tape unit (OFF).</p>
		UNIT SELECT Indicator	<p>Displays unit No. selected by UNIT SELECT switch.</p>
		POWER ON-OFF Switch	<p>Controls primary AC to tape cabinet.</p>
		FORMAT 1, 2, 3 Switch	<p>Selects tape format as:</p> <ol style="list-style-type: none"> 1. 3-character/word binary. 2. 2-character/word decimal. 3. 2-character/word binary.
UNLOAD Switch	<p>Used to rewind tape completely in manual mode with tape drive stopped. Pressing switch turns off</p>	DENSITY 800, 556, 200 Switch	<p>Selects tape packing density in bits per inch.</p>

5-12 DIO 7 AND 9-TRACK MAGNETIC TAPE

The DIO 9-track magnetic tape controller and tape unit(s) provide mass program and/or data storage for the 706 computer. This controller can be adapted for a 7-track drive with dual density and it can be adapted to function with either a synchronous or incremental tape drive. The controller is capable of handling up to four synchronous tape drives or one incremental tape drive. Table 5-41 shows the system and unit characteristics for the 7 and 9-track synchronous and incremental tape controller and transports.

5-12.1 Speed and Density

The standard product is the 9 track-synchronous read/synchronous write magnetic tape transport that operates at 25 inches per second (ips). There are three other 25 ips optional transports available:

- 7-track synchronous read/synchronous write;
- 9-track synchronous read/incremental write;
- 7-track synchronous read/incremental write.

Seven track units with the synchronous read/synchronous write have the capability of dual density selection—either 200/556 bits per inch (bpi) or 556/800 bpi.

The incremental write units can have only one of the densities 200, 556, or 800 bpi and can operate at one of the following stepping rates: 300, 500, 700 or 1000 steps/second. The maximum word transfer rate is equal to 10 KHz (800 bpi x 25 ips/2). Table 5-42 reviews the read and write rates for 7 and 9 track, synchronous and incremental tape units.

5-12.2 Data Type

Data transfer between the magnetic tape controller and the 706 is via the DIO bus. Either 16-bit binary data (9-track) or 12-bit binary data (7-track) is transferred. The lateral parity bit is "stripped-off" at the controller and is not transferred when reading magnetic tape. When writing, the lateral parity bit is determined and added automatically by the controller. The 9-track data word (Figure 5-16a) consists of two 8-bit bytes and is transmitted between the DIO magnetic tape controller and the accumulator (ACR 0-7, ACR 8-15). The 7-track data word (Figure 5-16b) consists of two 6-bit bytes and is transmitted between the DIO magnetic magnetic tape controller and the first 12 bits of the accumulator (ACR 0-5, ACR 6-11). ACR 12-15 are set to zero when reading and are ignored by the controller when writing.

5-12.3 Tape Unit Number Assignment

Each of the synchronous magnetic tape units may be assigned any number from 0 to 3 by making the appropriate connection of the unit cable on the rear of the controller. The incremental tape unit is always assigned as device number 0. When selecting the DIO magnetic tape to perform an operation, bits-14 and 15 of the accumulator are used to designate the device number. Table 5-43 shows the device assignment code.

5-12.4 DIO 7 and 9 Track Magnetic Tape Functions

Both 7-track and 9-track tape units are capable of performing the following functions.

5-12.4.1 Disconnect (DOT 9, 0)

The disconnect instruction is used to reset the individual magnetic tape unit. The disconnect command applies only to the device indicated by bits 14 and 15 of the accumulator (Table 5-43).

If the selected tape unit is reading when the disconnect command is issued, the controller discontinues transferring to the 706; however, the tape drive will continue to move tape until the next interrecord gap is detected and the end-of-function interrupt is generated.

If the selected tape unit is writing when the disconnect command is issued, the record is terminated and the end-of-function interrupt is generated. If a new command is issued within 1.4 msec the tape will not stop but continue moving.

If any command, other than read or write, is being performed while the disconnect is issued, the disconnect is ignored.

Issuing a disconnect command turns off both the data-ready and end-of-function interrupt.

5-12.4.2 Write One Record (DOT 9, 3)

This instruction selects a specific tape unit to initiate forward tape motion and send the data-ready interrupt to the CPU. The write-one-record instruction need only be given once for every record to be written. Data is transferred via the transfer-data-out command.

Table 5-41. DIO Magnetic Tape System Characteristics

Controller	9-Track	7-Track
Tape Units Controlled	1-4 Synchronous 1 incremental	1-4 Synchronous 1 incremental
Operating Modes	Read Write Backspace Rewind Write EOF Write Skip	Read Write Backspace Rewind Write EOF Write Skip
Data Word	16-bit binary, two 8-bit characters/word	12-bit binary, two 6-bit characters/word
Tape Character	9-bit binary (8 data bits plus parity)	7-bit binary (6 data bits plus parity)
Tape Format	IBM Compatible	IBM Compatible
Parity Checks	Lateral	Lateral
Parity Conventions	Lateral - odd	Lateral - odd/even
Error Checking & Correcting Code	Cyclic Redundancy check character, IBM compatible	None
Parity Gap Length	3 characters	3 characters
Inter-record Gap Length	0.6 in.	0.75 in.
Word Transfer Rate	One-half transport character byte transfer rate 10 KHz (max)	One-half transport character byte transfer rate 10 KHz (max)
Power Requirement	5 vdc, 1.2 amp	5 vdc, 1.2 amp
Operating Temperature	0 to 40°C	0 to 40°C

	9-Track	7-Track
Storage Temperature	-20 to 65°C	-20 to 65°C
Relative Humidity	90% (max)	90% (max)
Tape Unit		
Synchronous Tape Speed	25 ips	25 ips
Incremental Tape Speed	300, 500, 700 1000 steps/sec	300, 500, 700, 1000 steps/sec
No. of Tracks	9	7
Packing Density	800 bpi	200/556/800 bpi
Writing Mode	NRZ 1	NRZ 1
Character Byte Transfer	20 KHZ (max)	20 KHZ (max)
Tape Width	1/2 in.	1/2 in.
Reel Size (max)	10-1/2 in.	10-1/2 in.
Start Time	17.25 ± .75 ms	17.25 ± .75 ms
Start Distance (in)	0.215 ± .01	0.215 ± .01
Stop Time	17.25 ± .75 ms	17.25 ± .75 ms
Stop Distance (in)	0.215 ± .01	0.215 ± .01
Start-Stop Command Interval (min)	1.4 ms	4.3 ms
Rewind Time (2400 ft)	6 min. @ 75 ips	6 min @ 75 ips
Operating Temperature	0 to 40°C	0 to 40°C
Power Requirements	115 vac ± 10%, 60 cps, 300- 400 watts	115 vac ± 10% 60 cps, 300- 400 watts

Table 5-42. DIO Magnetic Tape Speed-Density Review

Tracks	Synchr./Incr. Version	Density bpi	Read Speed ips	Write Rate ips or Steps/sec	Lateral Parity	Basic/option	
9	Synchr. Read/Synchr. Write	800	25	25 ips	odd	basic	
	Synchr. Read/Incr. Write	800	25	300 steps/sec	odd	option	
				500 steps/sec			
				700 steps/sec			
				1000 steps/sec			
7	Synchr. Read/Synchr. Write	200/556 556/800	25	25 ips	odd/even	option	
	Synchr. Read/Incr. Write	200	25	300 steps/sec	odd/even	option	
				500 steps/sec			
				700 steps/sec			
				1000 steps/sec			
		556	25	300 steps/sec	odd/even	option	
							500 steps/sec
							700 steps/sec
							1000 steps/sec
	800	25	300 steps/sec	odd/even	option		
500 steps/sec							
700 steps/sec							
1000 steps/sec							

Table 5-43. DIO Magnetic Tape Device Assignment Codes

Device Number	ACR 14	ACR 15
0	0	0
1	0	1
2	1	0
3	1	1

5-12.4.3 Transfer Data Out (DOT 9, F or DOT 9, D)

Once the first data-ready interrupt is received due to the write-one-record command, data is written by issuing the transfer-data-out command. Twelve or sixteen bit data words (7 or 9-track) must reside in the accumulator while this command is being issued (see Figure 5-16). After the first data word has been transferred, the controller will interrupt the CPU and continue to interrupt after each data transfer. This command has to be given no later than 95 μ sec (800 bpi) following the data-ready interrupt.

If the transfer-data-out command does not arrive in time (95 μ sec) or if a disconnect command is sent, the CRC and LRC check characters are written on the tape and the end-of-function interrupt is generated.

5-12.4.4 Read One Record (DOT 9, 9)

This instruction selects a specific tape unit to initiate forward tape motion. When the first data word (2 bytes) has been read, the controller notifies the CPU via the data-ready interrupt. The read-one-record instruction need only be given once for every record to be read. Data is transferred by the transfer-data-in command.

5-12.4.5 Transfer Data In (DIN 9, F or DIN 9, D)

Once the first data-ready interrupt is received due to the read-one-record command, data is read by issuing the transfer-data-in command. Twelve or sixteen bit data words (7 or 9-track) are read into the accumulator as a result of executing this command (see Figure 5-16). After the first data word has been transferred, the controller will continue to interrupt the CPU with the data-ready interrupt after each data-transfer-in command. When the end-of-record is detected the end-of-function interrupt notifies the CPU that the record has been read completely. At this time status should be checked for a late condition or a parity error.

A late condition exists if the transfer-data-in command is not issued within 95 μ sec (800 bpi) after the data-ready interrupt is sent.

5-12.4.6 Write End-of-File (DOT 9, 2)

This command causes an EOF record (mark) to be written on the selected tape. The 9-track EOF

record consists of one frame of data; a binary one on track 3, 6 and 7. The data frame is followed by an 8 character gap and an LRC character identical to the data frame. There is no CRC. The 7-track EOF record consists of one frame of data; a binary one on track 1, 2, 4 and 8. The LRC is identical to and follows the data frame after a 3-character gap. The end-of-function interrupt is issued at the completion of the write end-of-file operation.

5-12.4.7 Backspace One Record (DOT 9, B)

This command causes the selected tape drive to move in the reverse direction for one record length and then stop in the preceding record gap. When the controller detects the record gap, the end-of-function interrupt is generated.

If the tape drive had previously been writing when the backspace one record command is issued, the tape will advance approximately 1/2 inch and then turn off the write enable circuitry before backspacing.

An exceptional case exists for the synchronous read/incremental write units only:

If the computer sends a backspace command while the tape is in the very first gap (but not at the load point after a load forward or rewind operation), the tape moves in the reverse direction and stops at the load point (BOT). A new command cannot be accepted because the transportation stays *not ready*. To put the transport into operation, press the LOAD FORW switch.

5-12.4.8 Rewind (DOT 9, A)

This command causes the selected tape to be driven in the reverse direction at 75 ips. When the BOT is sensed, the tape drive halts and advances to load point. The end-of-function interrupt is generated provided there has been no other DOT command issued to the DIO magnetic tape.

After a rewind command is accepted by the controller, it takes several milliseconds for the rewind operation to start. The rewinding status bit 4 also goes true after several milliseconds delay. Status bit 1 (device not ready) goes true immediately and stays true until the rewind has finished and the tape stops at the Load Point. Both status bits 1 and 4 go false at the same time.

If the tape drive had previously been writing when the rewind command is issued, the tape will advance approximately 1/2 inch and turn off the write enable circuitry before rewinding.

5-12.4.9 Write Skip (DOT 9, 1)

This command causes tape to be advanced approximately 4 inches with the write current on, thus erasing 4 inches of tape. At the conclusion of the write skip operation the end-of-function interrupt is generated.

5-12.4.10 Search for End-of-File

There is no direct command that causes the tape to be advanced or returned to the next end-of-file record (mark). If an EOF mark is detected after the issuance of the read-one-record command, bit-11 of the status response word is set true. There are no data-ready interrupts generated and the file mark is not transferred to the CPU. After detecting the EOF mark and LRC character the end-of-function interrupt is generated.

5-12.5 Continuous Functions

If the appropriate command is issued within 1.4 msec. following the end-of-function interrupt (i.e., write after a write operation, read after a read operation, or backspace after a backspace operation) on the same tape unit, the tape does not come to a stop in the gap but its motion continues. Therefore, for write commands (one record, EOF, or skip), continuous records may be written provided that one of the three write operations is selected in time and to the same transport. For read the read-one-record command, the tape may be driven in continuous motion provided that a new read command is issued within 1.4 milliseconds after detecting the end-of-record. The tape may be backspaced continuously in a like manner, except that the backspace command must be issued within 5 m sec. If a disconnect command is issued after the end-of-function interrupt has occurred, the continuous function is reset. A disconnect command issued before the end-of-function interrupt has no influence on the continuous mode.

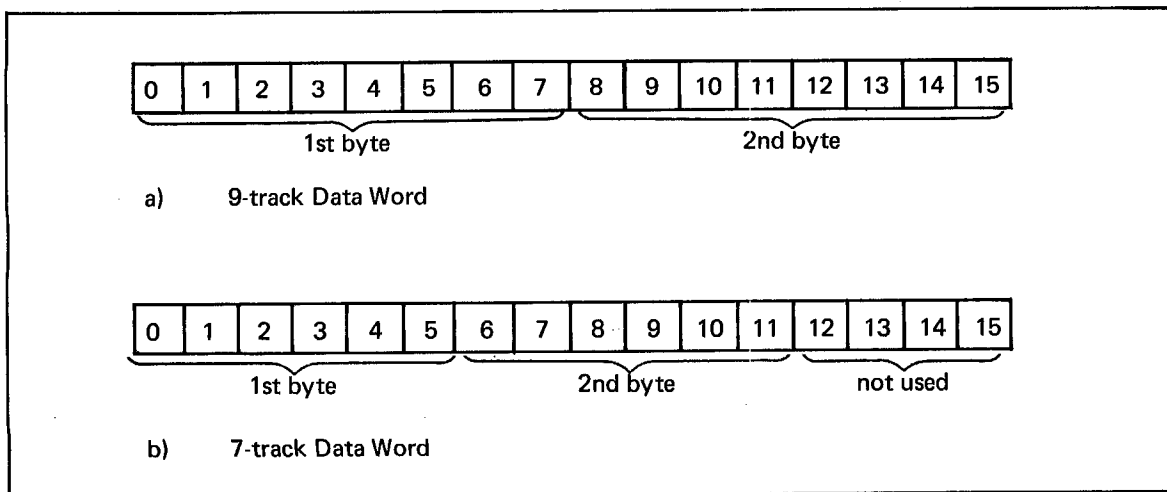


Figure 5-16. DIO Magnetic Tape Data Words

Interrupts are issued in the normal manner during continuous operation.

5-12.6 Function Codes and Command Words

Table 5-44 is a summary of the function codes for DIO 7 and 9-track magnetic tape.

5-12.7 Status Codes

Status is returned to the accumulator from the controller and from one of four possible magnetic tape units after issuing a DIN 9, 0; DIN 9, 1; DIN 9, 2; or a DIN 9, 3 command.

The DIO 7 and 9-track status word consists of 11 bits (Table 5-45).

Bit-0 indicates if the controller is busy. If the controller is busy it is not able to accept a new command. Bit-1 is true when the tape transport is not ready. Only that device indicated by the status request can set bit-1. The tape unit is not ready when rewinding, is in the manual mode, tape is broken, or the power is turned off.

Bit-2 and bit-3 of the status response word indicate BOT and EOT found respectively. When the EOT is passed, bit-3 stays true until the tape comes to the EOT during a rewind or backspace operation.

Note: The BOT and EOT status bit are indicators for the tape unit currently selected for operation and not the tape unit selected by the status command as for bit-1 and bit-4.

Bit-4 is true when the tape transport selected by the status command is rewinding. There is a several millisecond delay between the issuance of a rewind command and the setting of the rewind status bit 4. The incremental write tape unit does not set the rewind status bit.

Bit-6 and bit-7 of the status response word indicate the end-of-function and data-ready interrupt as being true, respectively.

Bit-11 is true when an end-of-file record has been detected. This is the only means by which file marks can be identified. The EOF detected bit remains true until a new DOT command is issued.

Bit-13 of the status response word is true when the write protect ring is in place on the tape transport currently selected for operation. The incremental write tape unit does not use the write protect status bit.

Bit-14 is set when a rate error occurs in the transferring of data during a read or write operation. At 800 bpi, data must be transferred within 95 μ sec. after the data-ready interrupt is generated.

Bit-15 is set when a parity error, CRC error or a rate error occurs during the reading or writing of a record.

5-12.8 Parity and Error Detection

Lateral parity is always recorded on the 7th or 9th track of the magnetic tape. Odd lateral parity is always recorded for 9-track units and odd or even parity (switch on front panel of controller) can be recorded for 7-track magnetic tapes. Lateral parity is checked during all read and write operations, but none of the parity data are transferred to the computer.

The longitudinal redundancy check character (LRC) is recorded such that longitudinal parity is always even. The LRC character is not checked after a read operation and is therefore not instrumental in setting status bit-15. The controller does load the LRC character into its assembly register thereby allowing a DIN 9, 7 command to transfer the LRC to the accumulator bits 0-7. The transfer of the LRC must take place after the end-of-function interrupt is generated.

The cyclic redundancy check character (CRC) is written at the end of every word (except EOF) and is checked when reading the record.

5-12.9 Interrupt Meanings

The DIO magnetic tape is somewhat unique in that two interrupts are generated. Both interrupts are levels (not pulses) and they stay true until reset with any new command except status or data transfer. Both interrupts are part of the status response word and both interrupts can be connected to the same or different interrupt levels.

The data-ready interrupt, when true, indicates that data is ready to be transmitted to or from the magnetic tape controller during a write or read operation.

The end-of-function interrupt, when true, indicates that any function has terminated. Normally, the data-ready interrupt is connected to a higher prior-

Table 5-44. Commands and Function Codes for DIO Magnetic Tape

Command	Hexadecimal Notation	Function Codes							
		Binary Notation							
		Bit Position							
		8	9	10	11	12	13	14	15
Disconnect Controller and Tape Unit*	90	1	0	0	1	0	0	0	0
Write Skip*	91	1	0	0	1	0	0	0	1
Write EOF*	92	1	0	0	1	0	0	1	0
Write One Record*	93	1	0	0	1	0	0	1	1
Transfer Data Out	9F or 9D	1	0	0	1	1	1	1	1
Read One Record*	99	1	0	0	1	1	0	0	1
Rewind*	9A	1	0	0	1	1	0	1	0
Backspace*	9B	1	0	0	1	1	0	1	1
DIN									
Return Status Device 0 to Accumulator	90	1	0	0	1	0	0	0	0
Return Status Device 1 to Accumulator	91	1	0	0	1	0	0	0	1
Return Status Device 3 to Accumulator	92	1	0	0	1	0	0	1	0
Return Status Device 3 to Accumulator	93	1	0	0	1	0	0	1	1
Transfer Data In	9F or 9D	1	0	0	1	1	1	1	1

*Device number must reside in Accumulator bits 14-15.

ity interrupt level than the end-of-function interrupt.

5-12.10 Timing

Figure 5-17 gives the interrupt timing for the DIO synchronous 9 and 7-track magnetic tape systems for reading. Write interrupt timing is shown in Figure 5-17.1. The time between the initiation of tape motion and the first interrupt is 18 m sec. for read operations only. When selecting for a write operation, the data-ready interrupt is issued immediately.

The word transfer rate for 25 ips at 800 bpi is 10 KHz, therefore, the time between interrupts is 100 u sec. Data transfer commands must be issued within 95 u sec after receiving the data-ready interrupt. For continuous operation, the new select DOT must be issued within 1.4 m sec after receiving the end-of-function interrupt or else there is a 270 m sec. delay before the next function can be started. However, the command will be accepted and stored for the remainder of the 270 msec. Interrupt times and maximum service times for 556 and 200 bpi units are proportionately longer than for the 800 bpi unit.

Table 5-45. DIO Magnetic Tape Status

Meaning When Bit True	Bit No.
Controller not ready	0
Device not ready*	1
Tape at BOT or Load Point	2
Tape has passed EOT	3
Device rewinding*	4**
End-of-function interrupt has not been serviced	6
Data-ready interrupt has not been serviced	7
End-of-file has been detected	11
Write-protect ring in place	13**
Rate error	14
Lateral parity error, CRC error, or rate error	15

* Response is for one of four possible units. Unit number must be specified in bits 12-15 of the DIN instruction for status. All other responses are for either the controller or the last tape unit to have been selected for an operation.

** These responses not available with incremental write tape units.

5-12.11 Operation Notes

5-12.11.1 Skip Record Operation

Records may be skipped in the forward direction by issuing a read command followed immediately by a disconnect command. All data-ready interrupts are inhibited and only the end-of-function interrupt will be generated when the inter-record gap is reached.

5-12.11.2 Change of Direction

Obviously, there cannot be continuous operation when a change of direction is involved with a new command issued to the DIO magnetic tape. A backspace or rewind command, given immediately after a read or write operation is completed, will be accepted but not performed until 250 m sec. after the completion of the forward operation. The same condition will occur when the completion of a backspace operation is followed by a read or write command or if a read command follows a write command or a write command follows a read command.

5-12.11.3 Simultaneous Unit Operation

Two or more units **cannot** be operated simultaneously except when they are rewinding. While one tape unit is rewinding, the other units may still accept commands; however, after deselection, the first unit will never cause the end-of-function interrupt to become true.

5-12.11.4 File Protection

In order to write on magnetic tape, the write enable ring must be inserted in the back of a magnetic tape file reel before mounting the reel in the unit. Without this ring, the magnetic tape file cannot be destroyed or written over. If a write command is issued with the ring out, tape motion will not be initiated and no interrupts will respond. Therefore, the status bit-13 should be interrogated before a write command is issued.

5-12.12 DIO Magnetic Tape Operation (Model No. 73696, 73697, 73674, 73675, 25 ips, 800 bpi)

5-12.12.1 Controls and Indicators

The controls and indicators are illustrated in Figure 5-18 and their functions are given in Table 5-46.

5-12.12.2 Loading Tape on Transport

The 5000 Series transport, in the position shown in Figure 5-18 has the take-up reel on the left side (as one faces the transport) located below the manual controls. The supply reel (reel to be recorded) is located on the right-hand side.

The 6000 Series transport, in the position shown in Figure 5-19 has the take-up reel on the top as one faces the transport, located to the left of the manual controls. The supply reel (reel to be recorded) is located below.

CAUTION: Be sure that the supply reel is such that tape will unwind from the reel when the reel is turned clockwise while facing the transport. This is shown in Figure 6-18 for 8½ inch transports and Figure 5-19 for 10½ inch transports.

5-12.12.3 Reel Loading

To load the supply reel, position the reel over the reel retainer hub, then depress the center plunger.

This will allow the reel to slip over the rubber ring on the reel retainer. Press the reel evenly and firmly against the back of the reel retainer hub while the center plunger is depressed. While holding the reel in this position, release the center plunger. The reel is now properly aligned in the tape path and ready for tape threading.

5-12.12.4 Tape Threading

Thread the tape through the transport as shown in Figure 5-18 and 5-19. Wrap the tape leader on the take-up reel so that the tape will be wound on the reel when the reel is rotated clockwise. Wind several turns on the take-up reel, then turn the supply reel counter clockwise until slack tape has been taken up.

CAUTION: If slack is left in the tape, tape damage may result when power is applied; after the tape has been manually tensioned, check to be sure that tape is properly located on all guides or tape damage may result.

5-12.12.5 Bringing Tape to Load Point

After the tape has been loaded on the transport, the following steps will bring the tape to LOAD POINT.

1. Turn power on with the POWER control.
2. Momentarily depress the LOAD FORWARD control. This will apply power to the capstan motor and reel motors. This allows the reel servos to operate and brings the tape storage arms to the nominal operating position.

CAUTION: Check to see that the tape is positioned properly on all guides or tape damage may result.

3. Momentarily depress the LOAD FORWARD control a second time. This will cause the tape to move forward at a velocity of 12 ips. When the

BOT tab is detected, the tape will stop with the front edge of the tab approximately 1.2 inches from the magnetic head gap.

When the above sequence has been followed and the transport has been switched to the on-line mode by momentarily depressing the ON-LINE control, the transport is ready to receive remote commands.

CAUTION: Close the dust cover door and keep it closed except when exchanging tape reels. Data reliability due to tape contamination will be impaired if the door is left open.

5-12.12.6 Bootstrap Loading

Records may be bootstrapped into the 706 memory by positioning the tape at the BOT and depressing the MAG TAPE BOOTSTRAP button on the 706 control panel. See Section 7-4.5 for detailed description of operation.

5-12.12.7 Unloading Tape

To unload a recorded tape, complete the following procedure if the power has been switched off (if power has not been switched off, start the procedure at step (3)).

1. Turn the transport power on.
2. Momentarily depress the LOAD FORWARD control to apply capstan-motor and reel-motor power.
3. Momentarily depress the REWIND control. When the tape has rewound to BOT, it will come to a controlled stop. The tape will then enter the load sequence and eventually come to rest at the BOT tab.
4. Momentarily depress the REWIND control a second time. This will initiate a further rewind action that will continue until tape tension is lost.
5. Open the dust cover door, depress the reel hub retainer plunger, and remove the supply reel. Close the dust cover door.

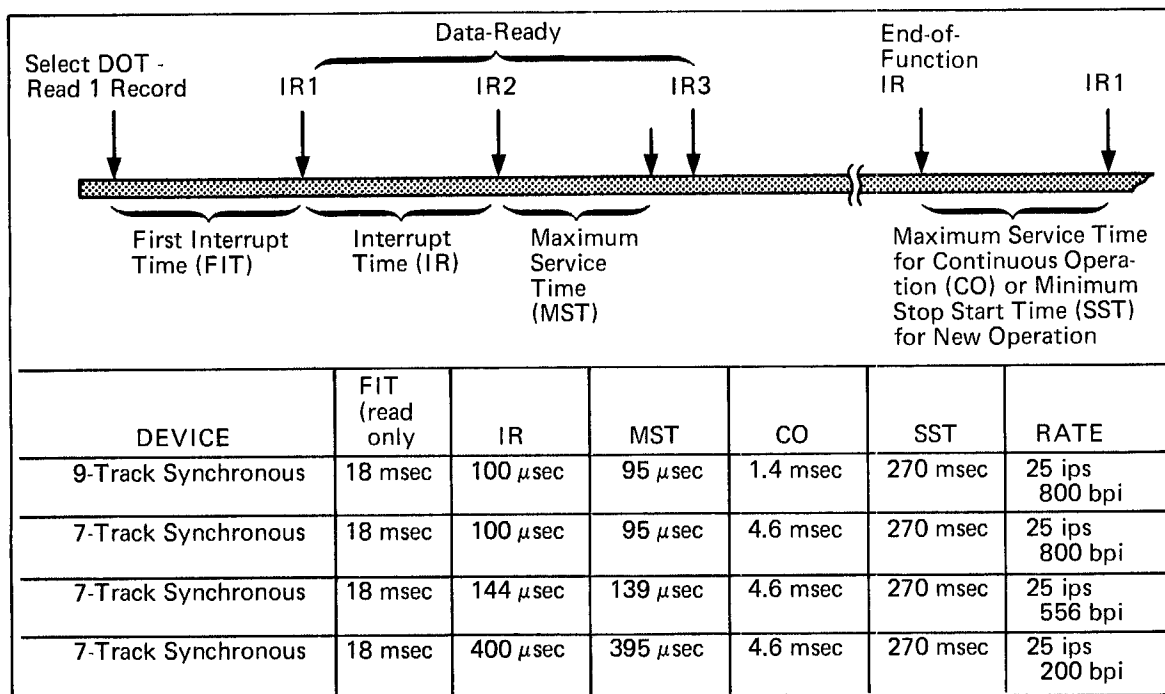
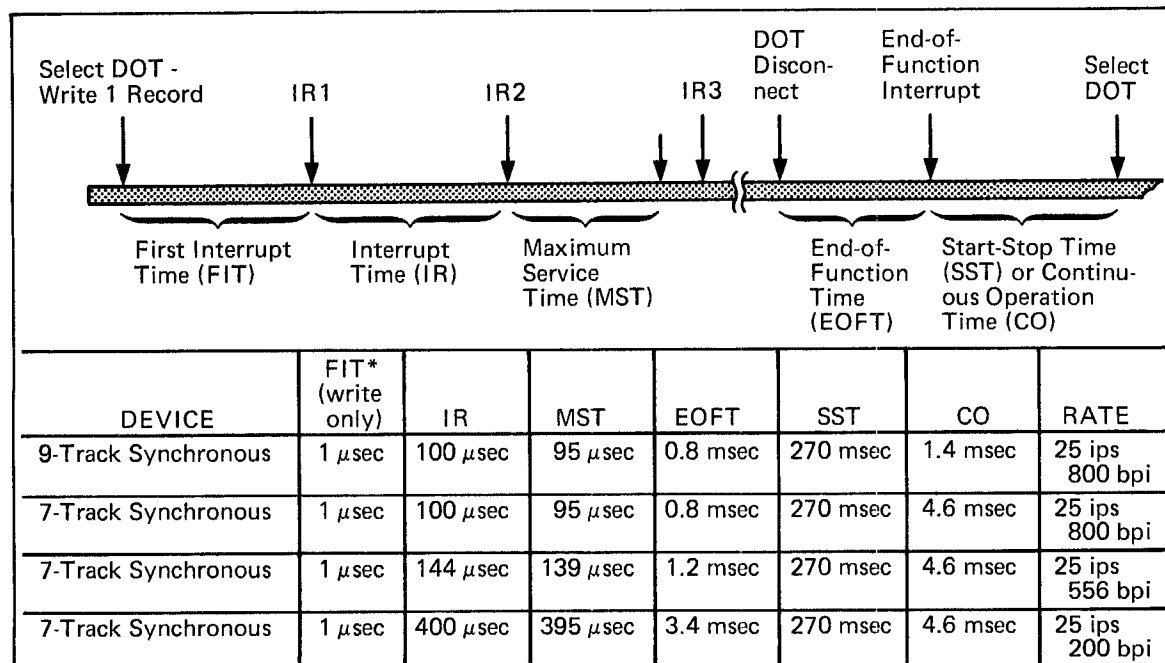


Figure 5-17. DIO Magnetic Tape Timing for Reading

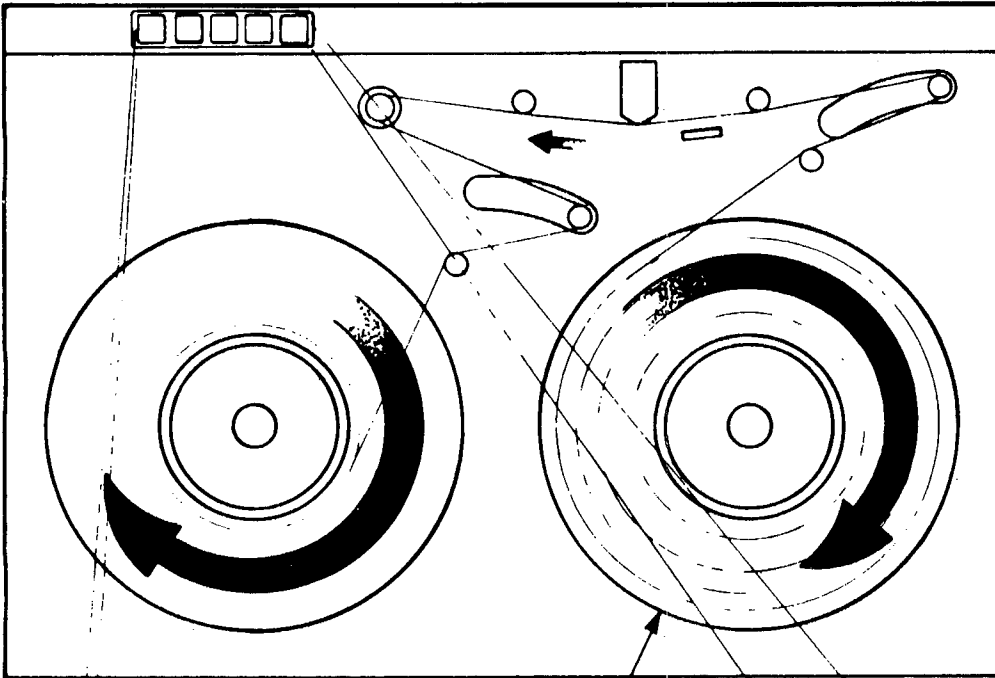


*FIT must be serviced within 23 msec (off BOT) or 150 msec (on BOT).

Figure 5-17.1. DIO Magnetic Tape Timing for Writing

Table 5-46. DIO Magnetic Tape Unit Controls and Indicators (Cont.)

Control/Indicator	Function
<p data-bbox="394 457 581 485">ON-LINE Switch</p> <p data-bbox="394 993 586 1045">WRITE ENABLE Indicator</p> <p data-bbox="394 1140 586 1167">DATA DENSITY</p>	<p data-bbox="951 457 1203 804">When depressed for the first time after a power-on sequence, transport is switched to an on-line mode and the indicator is illuminated. In this condition, all manual controls except the POWER and ON-LINE are disabled and the transport can now recognize a SELECT signal from the controller.</p> <p data-bbox="951 821 1203 978">Transport may be switched to an off-line mode, regardless of its SELECT status, by depressing the ON-LINE control a second time.</p> <p data-bbox="951 993 1203 1125">Indicator lights when power is on, and a tape reel with the Write Enable ring installed is on transport.</p> <p data-bbox="951 1140 1203 1350">Selects the character packing density at which the tape system operates. Two positions are available: High Density and Low Density. Combinations available are 800-556, 556-200.</p>



SUPPLY REEL

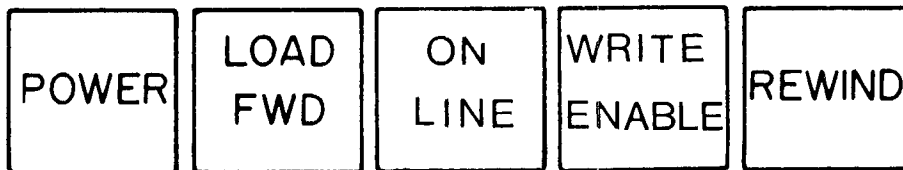


Figure 5-18. Tape Path and Control Panel (5000 Series)

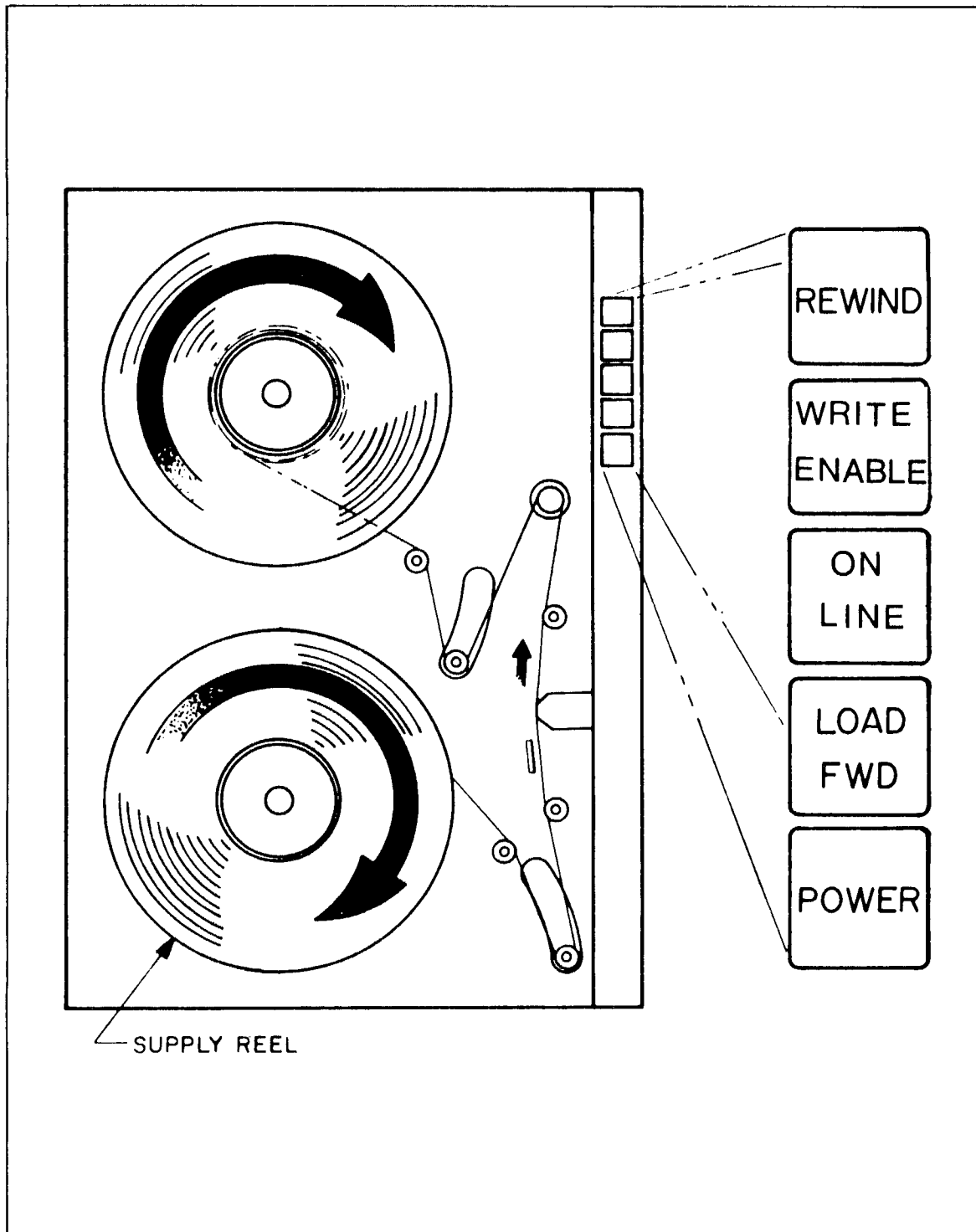


Figure 5-19. Tape Path and Control Panel (6000 Series)

5-13. DIGITAL PLOTTER

The 706 digital plotter controller and plotters provide a flexible means of producing high-quality ink-on-paper graphic presentations of 706 computer output data. Both flatbed and drum-type plotters may be attached to the 706 via the DIO bus. Table 5-47 shows the system and unit characteristics of the controller and digital plotters.

5-13.1 Speed

There are three types of movements associated with the drum-type digital plotter: pen, carriage and drum. Pen-up time is 10 msec. and pen-down time is 60 msec. Carriage and drum movement times are listed in Table 5-48.

5-13.2 Principles of Operation (Drum Type)

All of the drum type plotters listed in Table 5-47 operate on the same principle – digital commands activate step motors to produce a plot or drawing (a recording of the pen movement relative to the

surface of the paper). A separate motor controls the movement along each axis.

Because it is usually desirable for the X axis to be the larger of the two, this axis extends in the direction of the paper feed. The unit of movement is an increment, and the basic increment size is optional and determined by the gear ratio in the plotter (see Table 5-48).

The 8-vector diagram (Figure 5-20) illustrates the direction and increment values possible with each of the incremental input command formats. In 8-vector incremental input format, each plot command to a step motor causes movement relative to the axis controlled by that motor. A Y-axis command causes the pen to move one increment to the left (+) or right (-); an X-axis command causes the drum to rotate one increment, either up (-) or down (+). When the motors controlling both axes are activated at the same time, a combined X and Y movement results.

Table 5-47. Digital Plotter System Characteristics

<u>Controller</u>		
Plotter Supplier	CalComp	
Models		
Flat bed	502, 602, 618	
Drum type	565, 563, 665, 663	
Power Supply	<u>+5 Volts</u>	<u>+15 Volts</u>
Adjustment	± 1/2 V	± 1 V
Output Current	4.0 amps	0.75 amps
Load Regulation	0.5 V	0.15 V
Line Regulation	.05 V	.05 V
Ripple	1 mv RMS	1 mv RMS
Input	110VAC, 60 Hz, 5 amps	
Number of Plotters per Controller	1	
Commands	Pen up, pen down, drum up, drum down, carriage right, carriage left, increment	
Vector Format	8	
Number of Commands/Word	2	

Table 5-47. Digital Plotter System Characteristics (Cont.)

Model Maximum plot size: Y-Axis X-Axis	Flatbed Plotter		
	602	618	502
	31" 34"	54" 72"	31" 34"
Optional increment sizes available with corresponding maximum operating speeds	.005/.0025" at 450/900 increments/second .002/.001" at 450/900 increments/second .1/.05mm at 450/900 increments/second .05/.025mm at 450/900 increments/second	.005/.0025" at 200/400 increments/second .002/.001" at 450/900 increments/second .1/.05mm at 200/400 increments/second .05/.025mm at 450/900 increments/second	.010" at 300 increments/second .005" at 300 increments/second .002" at 300 increments/second 0.1mm at 300 increments/second 0.05mm at 300 increments/second
Maximum Speed: Inches-per-Second Incremental Mode 1) Horizontal/ Vertical Axis 2) Diagonal Axis	.005/.0025" at (1) 2.25 or (2) 3.1 inches/second .002/.001" at (1) 0.9 or (2) 1.2 inches/second .1/.05mm at (1) 1.8 or (2) 2.5 inches/second .05/.025mm at (1) 0.9 or (2) inches/second	.005/.0025" at (1) 1.0 or (2) 1.4 inches/second .002/.001" at (1) 0.9 or (2) 1.2 inches/second .1/.05mm at (1) 0.8 or (2) 1.1 inches/second .05/.025mm at (1) 0.9 or (2) 1.2 inches/second	.010" at (1) 3.0 or (2) 4.2 inches/sec .005" at (1) 1.5 or (2) 2.1 inches/sec .002" at (1) 0.6 or (2) 0.8 inches/sec 0.1mm at (1) 1.2 or (2) 1.7 inches/sec 0.05mm at (1) 0.6 or (2) 0.8 inches/sec
Plotter Dimensions	D: 45.40" W: 45.64" H: 37.75" (With stand; can be raised additional 3")	D: 74" (With control panel) W: 113" H: 35.26" (With stand; horizontal position)	D: 45.40" W: 45.64" H: 37.75" (With stand; can be raised additional 3")
Plotter Stand	Matching plotter stand for horizontal operation	Matching floor-mount plotter stand with variable, manual tilt provision to approx. 45" angle or beyond	Matching plotter stand for horizontal operation
Inputs: 500 Series Format 700 Series Format	Positive or negative polarity pulses, amplitude greater than 10V, rise time less than 10 usec, min. pulse width 4 usec, source impedance less than 500 ohms. Binary coded five-bit command signals at logic levels of 0 volts (false) and +8 volts (true); clock pulses at nominal incremental speed; plot enable signal at 0 volts (false) and +8 volts (true).		Positive or negative polarity pulses, amplitude greater than 10V, rise time less than 10 usec, minimum pulse width 4 usec, source impedance less than 500 ohms.
Power Requirements	115 vac ± 10%, single phase, 60 Hz, Also available to operate at 208 or 230V, 50/60 Hz.		115 vac ± 10%, single phase, 60 Hz. Also available to operate at 230V ± 10%, 50 Hz.

Model Nominal Plot Size: Y-Axis X-Axis	Drum Type Plotter			
	565	563	665	663
	11" 120"	28.55" 120"	11" 120"	28.55" 120"
Optional increment sizes available, with corresponding maximum operating speeds	.010" at 300 increments/second .005" at 300 increments/second 0.1mm at 300 increments/second	.010" at 200 increments/second .005" at 300 increments/second 0.1mm at 300 increments/second	.010/.005" at 450/900 increments/second .005/.0025" at 450/900 increments/second .0025/.00125" at 450/900 increments/second	.010/.005" at 350/700 increments/second .005/.0025" at 450/900 increments/second .0025/.00125" at 450/900 increments/second
Maximum Speed: Inches-per-second 1) Horizontal/ Vertical Axis 2) Diagonal Axis	.010" at (1) 3.0 or (2) 4.2 inches/sec .005" at (1) 1.5 or (2) 2.1 inches/sec 0.1mm at (1) 1.2 or (2) 1.7 inches/sec	.010" at (1) 2.0 or (2) 2.8 inches/sec .005" at (1) 1.5 or (2) 2.1 inches/sec 0.1mm at (1) 1.2 or (2) 1.7 inches/sec	.010/.005" at (1) 4.5 or (2) 6.3 inches/second .005/.0025" at (1) 2.25 or (2) 3.1 inches/second .0025/.00125" at (1) 1.125 or (2) 1.5 inches/second	.010/.005" at (1) 3.5 or (2) 4.9 inches/second .005/.0025" at (1) 2.25 or (2) 3.1 inches/second .0025/.00125" at (1) 1.125 or (2) 1.5 inches/second
Plotter Dimensions	D 14.7" W 18.0" H 9.8"	D 14.7" W 39.5" H 9.8"	Cabinet D 15.5" D 21" W 20.6" W 48" H 10.2" H 29"	Cabinet D 15.5" D 21" W 40.12" W 48" H 10.2" H 29"
Inputs: 500 Series Format	Positive or negative polarity pulses, amplitude greater than 10V, rise time less than 10 usec, minimum pulse width 4 usec, source impedance less than 500 ohms.		Positive or negative polarity pulses, amplitude greater than 10V, rise time less than 10 usec, minimum pulse width 4 usec, source impedance less than 500 ohms. Binary-coded five-bit command signals at logic levels of 0 volt (false) and +8V (true).	
Power Requirements	105 to 125 vac, single phase, 50/60 Hz, 1.5 amp at 115 vac.		115 vac ± 10%, single phase, 60 Hz.	

Table 5-48. Carriage and Drum Execution Time.

Model	Incr/Sec	Incr. Length	Execution Time
502	300	.010"	3.4 ms
502	300	.005"	3.4 ms
502	300	.002"	3.4 ms
502	300	.1 mm	3.4 ms
502	300	.05 mm	3.4 ms
602	450	.005"	2.2 ms
602	900	.0025"	1.1 ms
602	450	.002"	2.2 ms
602	900	.001"	1.1 ms
602	450	.1 mm	2.2 ms
602	900	.05 mm	1.1 ms
602	900	.025 mm	1.1 ms
618	200	.005 mm	5.0 ms
618	400	.0025"	2.5 ms
618	450	.002"	2.2 ms
618	900	.001"	1.1 ms
618	200	.1 mm	5.0 ms
618	400	.05 mm	2.5 ms
618	450	.05 mm	2.2 ms
618	900	.025 mm	1.1 ms
565	300	.010"	3.4 ms
565	300	.005"	3.4 ms
565	300	.1 mm	3.4 ms
563	200	.010"	5.0 ms
563	300	.005"	3.4 ms
563	300	.1 mm	3.4 ms
665	450	.010"	2.2 ms
665	900	.005"	1.1 ms
665	900	.0025"	1.1 ms
665	900	.00125"	1.1 ms
663	350	.010"	2.9 ms
663	700	.005"	1.5 ms
663	900	.0025"	1.1 ms
663	900	.00125	1.1 ms

5-13.3 Digital Plotter Functions

5-13.3.1 Disconnect (DOT 0, 0)

The disconnect instruction resets the plotter interrupt and sets bit-0 of the status response word. No more interrupts can be sent until the plotter is initialized as more data is transmitted.

5-13.3.2 Initialize (DOT 0, 5)

This instruction conditions the plotter controller by resetting the control flip-flops. The initialize instruction performs the same function as the manual reset signal from the CPU.

5-13.3.3 Transfer Data (DOT 0, 1)

This instruction loads two 6-bit execution bytes from the accumulator of the CPU into the execution register of the plotter controller. The two most significant bits of each byte control the pen movement (ACR 2-3, 10-11); the next two control the drum movement (ACR 4-5, 12-13); and the next two control the carriage movement (ACR 6-7, 14-15). See Figure 5-21. Bits 0-1 and 8-9 of the accumulator are not used.

The plotter controller looks at each byte separately and generates the appropriate pulses to the plotter; first for the left byte and then for the right byte. After completion of the second byte command the plotter controller generates an interrupt to the CPU.

Figure 5-22 is a summary of all the possible vectors that can be drawn using the combination of pen, drum, and carriage control bits.

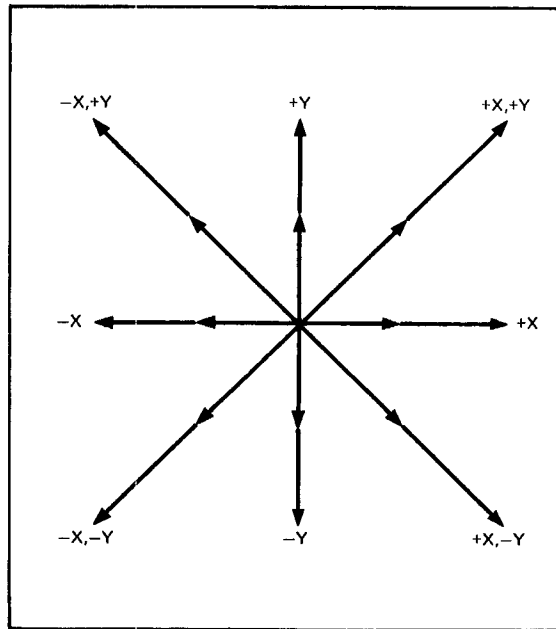


Figure 5-20. 8-Vector Format

5-13.4 Function Codes and Command Words

Table 5-49 is a summary of the function codes in hexadecimal and binary formats.

5-13.5 Status Codes

Status is returned to the accumulator after issuing a DIN 0, 0 instruction. The digital plotter status word consists of three bits (Table 5-50).

Bit-0 true means that the controller has been disconnected. Bit-1 indicates that the controller is busy performing an operation. Bit-7 may be used to indicate that an interrupt is present when the bit is false.

5-13.6 Interrupt Meaning

The interrupt from the digital plotter means that two bytes have been plotted and that pen, drum, and carriage motion has stopped.

5-13.7 Timing

The timing for the various digital plotter models is shown in Figure 5-23. The maximum interrupt service time (MST) is zero because the digital plotter controller is not double buffered. All times shown include no pen motion.

5-13.8 Digital Plotter Operation (Model No. 75631 (565))

5-13.8.1 Controls and Indicators

The controls and indicators for the model 565 digital plotter are shown in Figure 5-24 and described in Table 5-51.

5-13.8.2 Installation of Pen

Ballpoint pens normally supplied with the recorder are black, blue, red and green. To assemble the pen, insert the desired color pen into the plunger, then

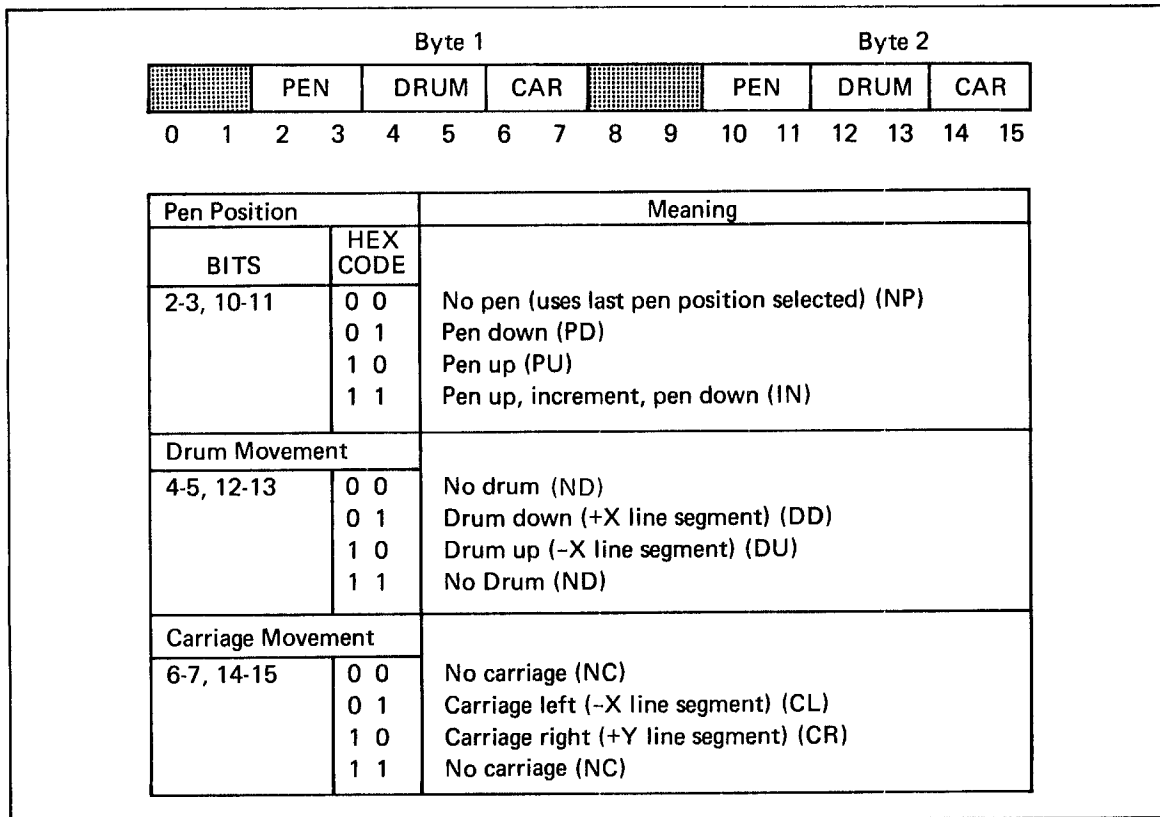


Figure 5-21. Pen, Drum and Carriage Control Codes.

Function	DRAW LINE SEGMENT				NO LINE SEGMENT, MOTION ONLY			
Initial Pen Position	PEN UP		PEN DOWN		PEN UP		PEN DOWN	
Final Pen Position	Pen Up	Pen Down	Pen Up	Pen Down	Pen Up	Pen Down	Pen Up	Pen Down
↑	PD, DD, NC (14) PU, ND, NC (20)	PD, DD, NC (14)	NP, DD, NC (04) PU, ND, NC (20)	NP, DD, NC (04)	NP, DD, NC (04)	IN, DD, NC (34)	PU, DD, NC (24)	IN, DD, NC (34)
↓	PD, DU, NC (18) PU, ND, NC (20)	PD, DU, NC (18)	NP, DU, NC (08) PU, ND, NC (20)	NP, DU, NC (08)	NP, DU, NC (08)	IN, DU, NC (38)	PU, DU, NC (28)	IN, DU, NC (38)
→	PD, ND, CR (12) PU, ND, NC (20)	PD, ND, CR (12)	NP, ND, CR (02) PU, ND, NC (20)	NP, ND, CR (02)	NP, ND, CR (02)	IN, ND, CR (32)	PU, ND, CR (22)	IN, ND, CR (32)
←	PD, ND, CL (11) PU, ND, NC (20)	PD, ND, CL (11)	NP, ND, CL (01) PU, ND, NC (20)	NP, ND, CL (01)	NP, ND, CL (01)	IN, ND, CL (31)	PU, ND, CL (21)	IN, ND, CL (31)
↗	PD, DD, CR (16) PU, ND, NC (20)	PD, DD, CR (16)	NP, DD, CR (06) PU, ND, NC (20)	NP, DD, CR (06)	NP, DD, CR (06)	IN, DD, CR (36)	PU, DD, CR (26)	IN, DD, CR (36)
↘	PD, DU, CL (19) PU, ND, NC (20)	PD, DU, CL (19)	NP, DU, CL (09) PU, ND, NC (20)	NP, DU, CL (09)	NP, DU, CL (09)	IN, DU, CL (39)	PU, DU, CL (29)	IN, DU, CL (39)
↖	PD, DD, CL (15) PU, ND, NC (20)	PD, DD, CL (15)	NP, DD, CL (05) PU, ND, NC (20)	NP, DD, CL (05)	NP, DD, CL (05)	IN, DD, CL (35)	PU, DD, CL (25)	IN, DD, CL (35)
↙	PD, DU, CR (1A) PU, ND, NC (20)	PD, DU, CR (1A)	NP, DU, CR (0A) PU, ND, NC (20)	NP, DU, CR (0A)	NP, DU, CR (0A)	IN, DU, CR (3A)	PU, DU, CR (2A)	IN, DU, CR (3A)

NOTATION: PU-Pen Up; PD-Pen Down; DU-Drum Up; DD-Drum Down; CR-Carriage Right; CL-Carriage Left; IN-Pen Up, increment Pen Down; NP-No Pen; ND-No Drum; NC-No Carriage. Hex Codes in parenthesis.

Figure 5-22. Vector Command Codes.

Table 5-49. Digital Plotter Commands and Function Codes

Digital Plotter Commands	Hexadecimal Notation	Function Codes							
		Binary Notation							
		8	9	10	11	12	13	14	15
Disconnect Controller	00	0	0	0	0	0	0	0	0
Transfer Data (2 bytes)	01	0	0	0	0	0	0	0	1
Initialize Controller DIN	05	0	0	0	0	0	1	0	1
Return Status to Accumulator	00	0	0	0	0	0	0	0	0

Table 5-50. Digital Plotter Status

Meaning When Bit True	Bit No.
Controller has been disconnected	0
Controller power on	1
Controller busy (no interrupt)	7

insert the pen and plunger into the holder and install the threaded cap. Align the key on the holder with the key slots in the carriage and press the pen assembly into the pen mounting. Tighten the knurled nut on the bottom of the pen assembly.

5-13.8.3 Installation Precautions

The Model 565 depends upon free circulation of air under the base plate for proper cooling. Do not place the unit on top of any loose papers or cloth. Loose materials on this type can block the ventilating louvres in the base plate and cause over-heating. In addition, the unit should not be placed on top of any other heat-producing equipment.

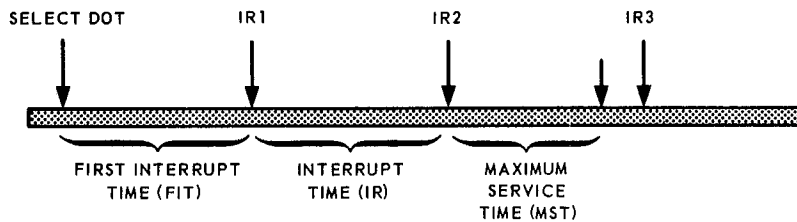
5-13.8.4 Installation of Chart Roll

To install a roll of chart paper in the instrument, first make sure that it is evenly wound on the core; i.e., that the ends are not "coned." Straighten coned rolls by tamping the end with the protruding core against a flat surface. Then observe the following points:

- a. Set POWER switch to OFF.
- b. Remove the pen assembly from the carriage by loosening the knurled nut at the bottom of the pen holder and lifting the assembly out of the carriage.

CAUTION: Use care not to drop the pen assembly or any of its parts. The assembly is constructed of soft steel to close tolerances, and can be rendered inoperative by nicks or dents.

- c. Rotate the right rear paper spool by hand until the drive key is pointing upward.
- d. Hold the new roll of chart paper so that the key slot in the core is pointing upward. Using your thumb, push the idler spool (left-hand end) to the left, and slip the left-hand end of the roll on the spool. Do not force the idler spool aside with the paper roll, as this tends to cone the end and thus cause misalignment.
- e. Lower the paper roll into the paper well and slide the right end onto the drive spool. Make certain the drive key engages the key slot in the core.
- f. Install a paper roll core on the two front spools below the drum, in the same manner as the paper roll.
- g. Pull the end of the paper over the drum so that the sprocket holes on both edges of the paper engage the sprockets on the drum. Guide the chart paper under the carriage rods and behind the tear bar. The chart paper winds on the take-up spool from the back; fasten the end on the spool with scotch tape provided in the accessory kit. Using the DRUM FAST RUN switch, wind a few turns onto the take-up spool.
- h. Scales on the inside faces of the rear paper spools indicate the approximate amount in feet of paper remaining on the roll. The black scale is for chart papers 0.003 inch thick; the blue scale is for 0.002-inch paper:



Model	FIT (ms)*	IR (ms)*	MST (ms)*	Transfer Rate
502	6.8	6.8	0	300 bytes/sec
602	4.4	4.4	0	450 bytes/sec
602	2.2	2.2	0	900 bytes/sec
618	10.0	10.0	0	200 bytes/sec
618	5.0	5.0	0	400 bytes/sec
618	4.4	4.4	0	450 bytes/sec
618	2.2	2.2	0	900 bytes/sec
565	6.8	6.8	0	300 bytes/sec
563	10.0	10.0	0	200 bytes/sec
563	6.8	6.8	0	300 bytes/sec
665	4.4	4.4	0	450 bytes/sec
665	2.2	2.2	0	900 bytes/sec
663	5.8	5.8	0	350 bytes/sec
663	3.0	3.0	0	700 bytes/sec
663	2.2	2.2	0	900 bytes/sec

* Assumes no pen-up time (10 msec) or pen-down time (60 msec)

Figure 5-23. Digital Plotter Timing.

5-13.8.5 Installation of Single Sheet Chart Paper

Single sheets of chart paper may be used for plotting in place of the chart paper roll. To install a single sheet of chart paper, proceed as follows:

- a. Set POWER and CHART DRIVE switches to OFF.
- b. Remove the pen assembly from the carriage.
- c. Slide the chart paper sheet under the carriage rods onto the drum surface.
- d. Fasten the top edge of the paper to the drum with two or three short pieces of tape. Rotate the drum by hand, keeping the paper smooth and flat against the drum surface. Fasten the bottom edge of the paper in the same manner as the top.

5-13.8.6 Operational Checkout

The following procedure is intended to provide an overall check of the operation of the Model 565 prior to the start of automatic recording. If a malfunction is encountered at any point in the checkout procedure, refer to Section 5 of the Cal Comp 565 Reference Manual.

- a. Install the pen assembly in its carriage.
- b. Set POWER and CHART DRIVE switches to ON.
- c. Set DRUM FAST RUN to UP position. Check that the pen traces a vertical line on the chart paper.
- d. Turn the PEN switch to DOWN, then UP. Check that the pen lifts off the drum surface.
- e. Set the PEN switch to DOWN position, then set the DRUM FAST RUN to DOWN position. Check that the pen again traces a vertical line on the chart paper.
- f. Set the CARRIAGE FAST RUN switch to the left position. Check that the pen traces a horizontal line on the chart and that the carriage step motor stops when the carriage reaches its limit of travel. Repeat with the CARRIAGE FAST RUN switch in the right-hand position.

g. Alternately operate the CARRIAGE SINGLE STEP and DRUM SINGLE STEP switches. Check that both the carriage and the drum move one step only each time one of the switches is operated.

h. Move the carriage near the left margin of the chart paper. Set CARRIAGE FAST RUN switch to the right position and DRUM FAST RUN to down position. Allow the instrument to run until the carriage reaches the right side of its track, then return both switches to off (center) position. Check that the pen traces a 45-degree line on the chart. Check the line carefully for any evidence of discontinuity.

i. Operate the DRUM SINGLE STEP switch several times to reposition the pen either above or below its position at the end of step h.

j. Set CARRIAGE FAST RUN switch to the left position and DRUM FAST RUN switch to the up position. Allow the instrument to run until the carriage reaches the left side of the track, then return both switches to OFF position. Check that the pen again traces a 45-degree line on the chart and that this line is exactly parallel to the line traced in step h.

k. Repeat steps h through j, changing the switch positions to produce two 45-degree lines at right angles to the first two. Again check for discontinuities and make certain the two lines are parallel.

5-13.8.7 Reticles

Two alignment reticles are provided to permit manual alignment of the carriage to the desired zero point. One of these replaces the pen assembly and shows the exact point on which the pen will fall; the other occupies a special receptacle in the carriage, and indicates a point exactly one inch from the pen measured along the X axis in the positive direction. The latter reticle can be left in the carriage at all times. When installing either reticle, rotate it as it is inserted to prevent damage to the O-ring.

5-13.8.8 Automatic Operation

After the installation of chart paper, the operational checkout, and the reticle adjustment described in preceding paragraphs, the Model 565, is ready for use. Connect the controller to P5 on the rear panel of the recorder and set the POWER switch to ON.

NOTE

Do not turn power off, then on again, during a re-

cording; this can introduce a plotting error of one step in any direction.

5-13.8.9 Removal of Chart Paper

The roll of chart paper or single sheet of chart paper should be removed in the reverse sequence to the installation procedure described above. If a single sheet of chart paper is used, any remaining tape adhesive should be cleaned from the drum surface with acetone or a good commercial grade of cleaning solvent.

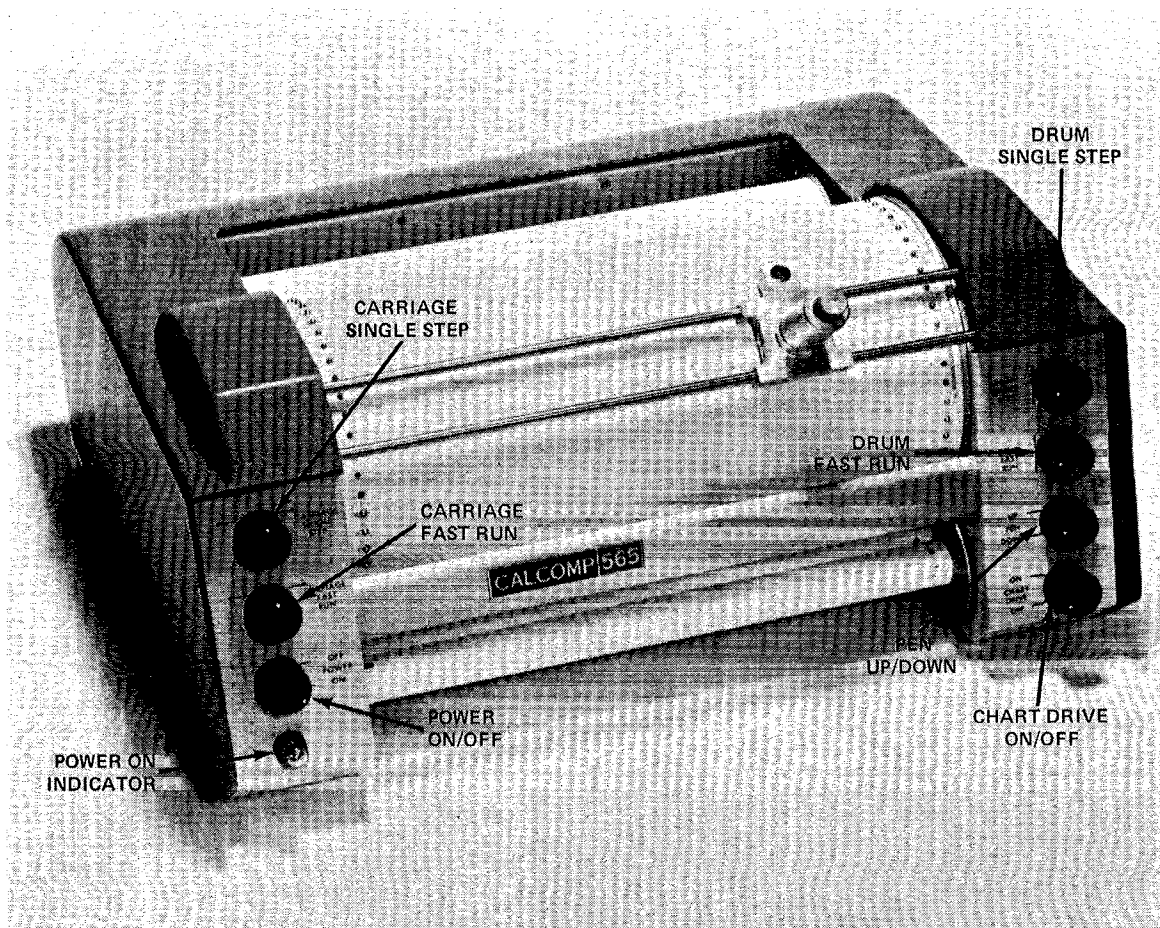


Figure 5-24. Digital Plotter (Model 565) Control Panel.

Table 5-51. Digital Plotter (Model 565) Controls and Indicators.

Control	Function
<p>POWER ON/OFF Switch</p>	<p>The POWER ON/OFF switch connects 115-volt a-c power from connector J7 on the rear panel of the recorder to the cooling fan and the power supply transformer. A neon indicator, located directly below the switch, is lighted when the switch is ON.</p>
<p>CARRIAGE FAST RUN Switch</p>	<p>The CARRIAGE FAST RUN switch allows the pen carriage to be stepped rapidly to the left or right at the rate of 120 steps per second where using 60 cps power, or 100 steps per second where using 50 cps power. The switch may be used to move the carriage to any desired area of the graph, or for operational checkout of the carriage control circuits and the carriage step motor.</p>
<p>CARRIAGE SINGLE STEP Switch</p>	<p>THE CARRIAGE SINGLE STEP switch allows the pen carriage to be moved in single-step increments either to the left or right. This control, in combination with the DRUM SINGLE STEP control, permits the operator to accurately align the carriage on a point or fixed coordinate on the paper.</p>
<p>CHART DRIVE ON/OFF Switch</p>	<p>The CHART DRIVE/ON/OFF switch allows the operator to disable the front and rear chart takeup motors. This permits the use of single sheets of chart paper in place of the paper rolls supplied with the instrument.</p>

Table 5-51. Digital Plotter (Model 565) Controls and Indicators (cont.)

Control	Function
<p>PEN UP/DOWN Switch</p>	<p>The PEN UP/DOWN switch provides a means of manually raising and lowering the pen from the surface of the drum.</p> <p>NOTE: When the instrument is first turned on, or if the pen is removed and replaced when the plunger is in the PEN-UP position, the pen may remain down even when the PEN switch is turned to UP. When this occurs turn the PEN switch first to DOWN, then to UP.</p>
<p>DRUM FAST RUN Switch</p>	<p>The DRUM FAST RUN switch allows the drum to be stepped rapidly up or down at the rate of 120 steps per second. The switch is used in the same manner as the CARRIAGE FAST RUN control to move the pen to any desired area of the chart, or for operational checkout of the drum control circuits and the drum step motor.</p>

5-14. TELETYPE MULTIPLEXER

The teletype multiplexer provides the means for the 706 to communicate with up to 16 teletypes concurrently.

5-14.1 Speed

The maximum transfer rate of an ASR-33 or ASR-35 teletype is 10 characters per second (cps). Therefore, the maximum transfer rate of the teletype multiplexer is 160 cps.

5-14.2 Data Type

Both the Model 33 and the Model 35 teletype sets employ the standard 8-level ASCII code (Appendix G). The eighth code element, which is the parity bit, is always transmitting a "one."

The character is transmitted between the multiplexer and the teletype as a series of eight spacing (0) or marking (1) pulses preceded by a start pulse (spacing) of the same duration as the character pulses and followed by a stop pulse (marking) of twice the duration of the character pulses. Thus the actual signal transmitted contains 11 units, based on the duration of one character pulse as one unit. At 10 cps, the character is transmitted in 100 msec. and each unit is transmitted in 9.09 msec.

Transmission and reception of character codes is timed mechanically in the teletype sets by rotating shafts. When tripped, the shafts make one revolution in 100 msec and actuate contacts that generate or receive the coding pulses. This action is simulated in the teletype multiplexer controller by a shift register that is clocked at the same rate (9.09 ms period) and is used to transmit and receive the serial characters.

5-14.3 Multiplexer Functions

Each teletype channel (0-15) has its own assembly register. The register transmits data to and from the teletype serially and transmits data to and from the computer in parallel.

NOTE

The assembly register communicates only with the

teletype send and receive circuits, not with individual components of the set (keyboard, paper tape punch, printer, and paper tape reader). The controller has no control over the destination of data sent to the set (printer and/or paper tape punch) nor the source of data received from the set (keyboard or paper tape reader). These are determined solely by the teletype set controls and the manner in which the set is operated.

The controller has a scanner that sequentially interrogates each channel. If a channel has a character for input or desires a character to be output, the scan is stopped and an interrupt is generated. The scanner provides the address information necessary for input or output and will resume scanning when the data transfer has been completed.

Unwanted message alert may be accomplished by depressing the BREAK key on the teletype. Status bit-1 will be set and transmitted to the computer on the next interrupt when the channel requests another character from the CPU. The program should normally disconnect the channel at this time.

5-14.3.1 Disconnect Channel (DOT A, 0)

The channel indicated by bits 12-15 of the accumulator is disconnected. The scanner will not stop on the disconnected channel. Operation of the teletype in this mode is identical to operation in the LOCAL mode.

5-14.3.2 Select Channel for Input Mode (DOT A, 1)

The channel indicated by bits 12-15 of the accumulator is selected for the input mode. When connected to the selected channel, inputting a character will cause the scanner to stop and an interrupt will be generated.

5-14.3.3 Select Channel for Output Mode (DOT A, 2)

The channel indicated by bits 12-15 of the accumulator is selected for the output mode. This command causes the scanner to stop at the selected channel forcing the controller to generate an interrupt to the CPU.

5-14.3.4 Output Character (DOT A, 3)

This command transmits data residing in bits 8-15 of the accumulator to the teletype channel designated by the scanner. After completion of the data transfer, the scanner proceeds to search and lock onto the next selected channel.

5-14.3.5 Input Character (DIN A, 1)

This command causes a word to be transmitted to the accumulator. There are two formats for this word (Figure 5-25), depending on whether the channel interrupting the CPU is in the input or output mode.

The method of determining whether an interrupting channel is in the input or output mode is to input a word and examine bit-0. If bit-0 is a binary one, the channel is selected for input and bits 8-15 of the accumulator contain the character from the teletype. If bit-1 is a binary one, the CPU was late taking the character from the controller register, thereby losing some data. Bits 4-7 contain the channel number.

If bit-0 is a binary 0, the channel is selected for output with bits 12-15 indicating the channel causing the interrupt. If bit-1 is a binary one, the channel should be disconnected. If bit-1 is a binary 0, the next command to the multiplexer should be Output Character.

When a channel is selected for input and an Input Character Command is issued, the interrupt is cleared and the scanner is released to search for the next selected channel. When a channel is selected for output and an Input Character command is issued, an Output Character command must be given before the scanner is released.

5-14.4 Teletype Multiplexer Function Codes

The various 8-bit control codes are illustrated in hexadecimal and binary formats in Table 5-52.

5-14.5 Status Codes

Status is returned to the accumulator from the teletype multiplexer after a DIN A, 0 command is issued.

The teletype multiplexer status word consists of three bits (Table 5-53). Bit-0 true means that there is an interrupt waiting to be serviced by the CPU.

Bit-1 true means that there is an unwanted message on a remote teletype. This status bit is the same as bit-1 that is received after doing an Input Character command. Bit-15 of the status response reflects if any of the channels have been selected for either input or output. If all channels have been disconnected, bit-15 is a binary 0.

5-14.6 Interrupt Meaning

An interrupt is generated when a character is transferred (serially) between the multiplexer controller and the teletype. When inputting, the interrupt signifies that a character is waiting to be transferred from the assembly register. When outputting, the interrupt signifies that the character has been sent to the teletype and the controller is ready to receive another character from the CPU.

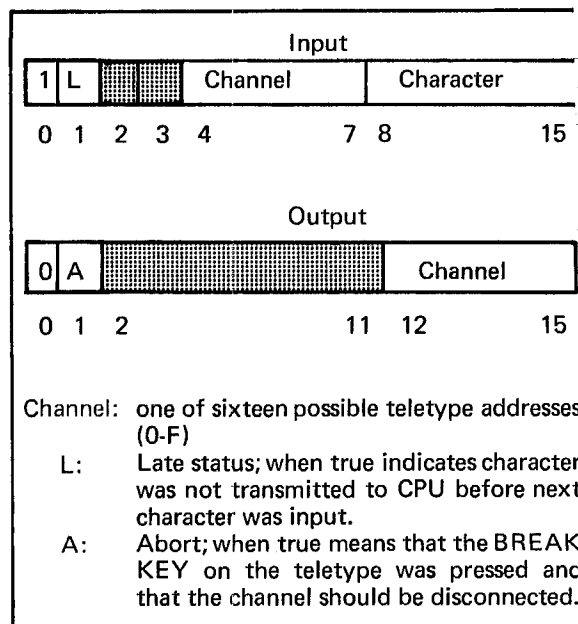


Figure 5-25. Teletype Multiplexer Input word Formats

5-14.7 Timing

In order to transfer data at the maximum rate of 10 characters per second, the maximum service time for processing interrupts from the teletype multiplexer is equal to 100 milliseconds divided by the number of selected teletypes (Figure 5-26).

Table 5-52. Teletype Multiplexer Function Codes

Command	Hexadecimal Notation	Function Codes							
DOT		Binary Notation							
		Bit Position							
		8	9	10	11	12	13	14	15
Disconnect Channel	A0	1	0	1	0	0	0	0	0
Select Channel for Input	A1	1	0	1	0	0	0	0	1
Select Channel for Output	A2	1	0	1	0	0	0	1	0
Output Character	A3	1	0	1	0	0	0	1	1
DIN									
Return Status to Accumulator	A0	1	0	1	0	0	0	0	0
Input Character	A1	1	0	1	0	0	0	0	1

Table 5-53. Teletype Multiplexer Status

Meaning When Bit True	Bit No.
Interrupt waiting to be serviced.	0
Unwanted message on remote teletype.	1
At least one channel has been selected.	15

5-14.8 Operation Notes

The data transfer functions only have significance when the teletype set is operating in the ON LINE mode. When the set is operating in the LOCAL mode, the send and receive circuits are connected together internally and are disconnected from the controller input and output lines.

5-14.9 Teletype Multiplexer Operation (Model No. 76601 and 76602)

Refer to Section 5-5 for the operating procedure of the ASR-33 teletype. There are no controls or indicators in the multiplexer.

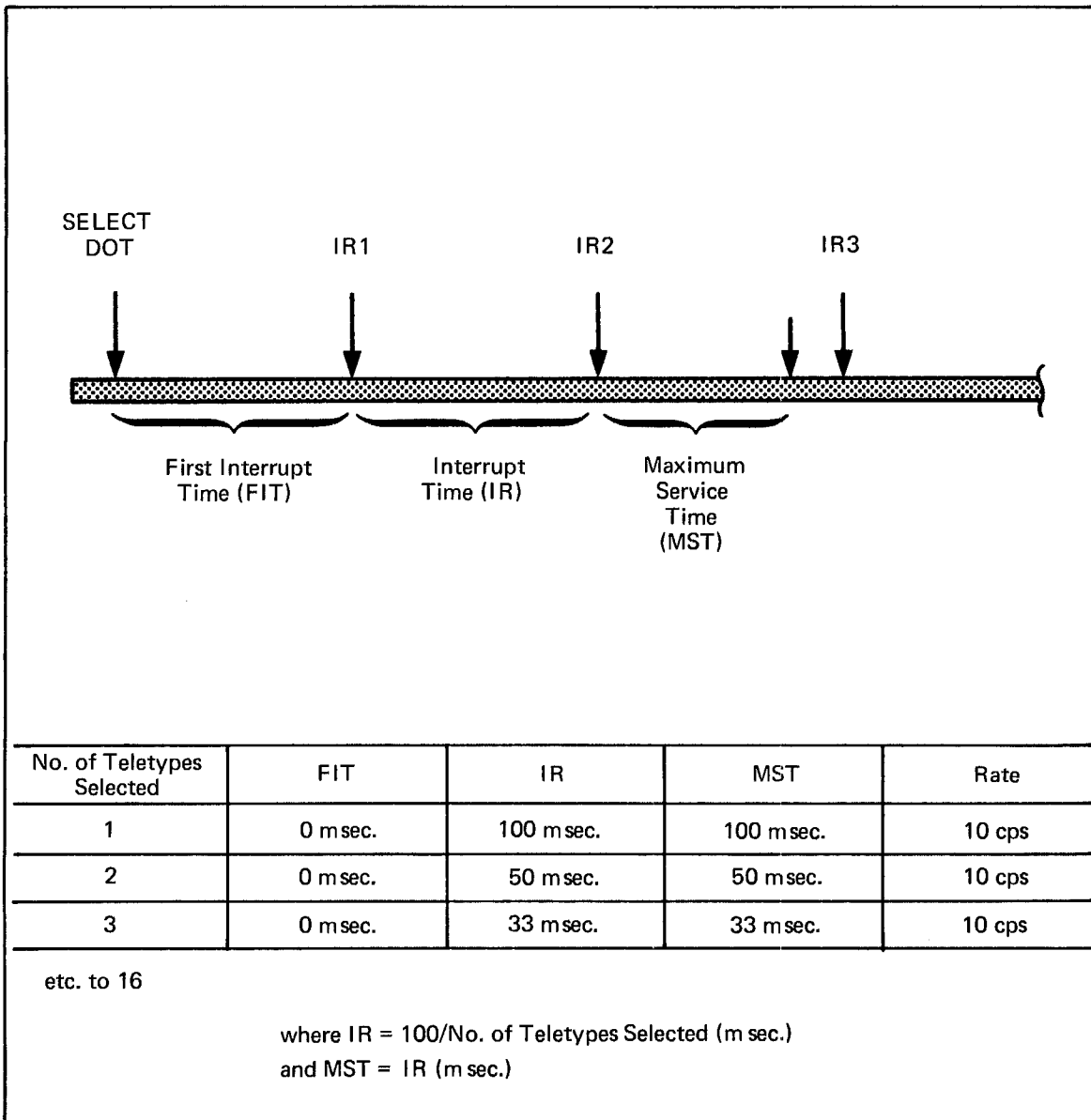


Figure 5-26. Teletype Multiplex Timing

5-15 BUFFERED INPUT/OUTPUT CHANNELS

The buffered digital input channels provide an interface for transfer of digital data from external devices to the Raytheon 706 Computer. All channels share the direct input bus, with each channel being capable of completely independent operation. The computer is able to use interrupts or a tight sense loop to determine when data has been entered into one of its three channels.

The buffered digital output channels provide an interface for transfer of digital data from the Raytheon 706 Computer to external system devices. All channels share the direct output bus, with each channel being capable of completely independent operation. The computer is able to use interrupts or a tight sense loop to determine when data has been read from any channel.

The buffered input and output characteristics are summarized in Tables 5-54 and 5-55.

5-15.1 Speed

The buffered digital input/output channel can operate at the speed of the routine used to transmit or receive data. Table 5-5 shows a burst method coding example that could serve to input data from the buffered input channel. The burst method shown inputs data at a rate of 159 KHz.

5-15.2 Data Format

The 16-bit data are transferred between any of the three input channels or three output channels and the accumulator.

5-15.3 Buffered Channel Input Functions (DIN 6, 1; DIN 6, 2; DIN 6, 3)

This command causes 16-bits of data from the input channel buffer (indicated in bits 12-15 of the DIN instruction) to be transferred to bits-0 through 15 of the accumulator. Execution of this command causes the interrupt from the Input Buffer Channel to be dropped.

5-15.4 Buffered Channel Output Functions

5-15.4.1 Transfer Data to Channel (DOT 7, 9; DOT 7, A; DOT 7, B)

This command causes 16-bits of data to be transferred from the accumulator to the output channel

buffer (indicated in bits 12-15 of the DOT instruction). Execution of this command causes the interrupt from the output buffer channel to be dropped.

5-15.4.2 Transfer Data to Channel and Disconnect (DOT 7, 1; DOT 7, 2; DOT 7, 3)

The main purpose of this command is to terminate a data block transfer. The contents of the accumulator are gated into the buffer channel (indicated in bits 12-15 of the DOT instruction). Execution of this command resets the interrupt from the output buffer channel. This command immediately resets the external data ready line between the buffered output channel and the external device. This causes the data not to be gated from the output channel to the external device. In order to transfer the data another transfer data command must be given using the same data in the accumulator.

Table 5-54. Buffered Digital Input Channel Characteristics

Types of Operation	Wait for interrupt to signify data has been entered
	Perform test to determine when data has been entered
Number of Addressable Channels	Up to 3 independently operable channels
Number of Bits per Word	16 parallel bits
Output Level of Buffer-Full Signal (to external system)	One: 0 to +0.5 volt Zero: +2.5 to 5.0 volts
Input Requirements (external system)	Both data and control signals are to be ground, or 0 volt, true and +2.5 to 5.0 volts false. Normal operation requires the input gate signal be a ground pulse. Data should be stable 100 nsec prior to and after the negative transition.
Input Loads	Two standard DTL loads
Internal Logic Levels	+2.5 to 5.0 volts, true 0 to 0.5 volt, false
Primary Power	110 volts, 60 cps, 0.5 amp

Table 5-55. Buffered Digital Output Channel Characteristics

Types of Operation	Output data and wait for interrupt to signify data has been read Output data and perform test to determine when data has been read Output data without any testing or qualification
Writing Method	In any of the above cases the data ready line to the external device may be made to be pulsed or dc
Number of Addressable Channels	Up to 3 independently operable channels
Number of Bits per Word	16 parallel bits
Output Levels (to external system)	One: 0 to 0.5 volt Zero: +2.5 to 5.0 volts
Drive Capability	Two driver outputs are available to the customer are are: 1. Unterminated output drivers capable of driving 25 standard loads when terminated with 120 ohms to +5 volts at the external device. (Specify MTG 3 drivers.) 2. Terminated output drivers sufficient for lower speed and shorter cable applications. (Specify MTG4 drivers.)
Input Requirements (external system)	Both data and control signals are to be ground, or 0 volt, true and +2.5 to 5.0 volts false.
Input Loads	Two standard DTL loads
Internal Logic Levels	+2.5 to 5.0 volts, true 0 to 0.5 volt, false
Primary Power	110 volts, 60 cps, 0.5 amp

The Transfer Data and Disconnect command in conjunction with the Transfer Data Command can be used to pulse the data ready line. The procedure is as follows:

- a. Load the data into the accumulator register and disconnect that channel. This places the data onto the data output line.
- b. Next a delay subroutine is entered to provide the output delay needed.
- c. With the same data in the accumulator register issue the fill buffer command. This sets the data ready line.
- d. If the data ready pulse need be stretched, another delay subroutine is entered at this point.
- e. Again, with the same data in the accumulator register, issue another disconnect command. This resets the data ready line making the data ready signal appear as a pulse.

A brief programming example for the above sequence for channel 1 follows:

```
LDW DATA      Load data word
DOT DEV, DISC1 DOT device with function
                code of 0001
JSX DELAY      Delay is a delay subroutine
DOT DEV, CH1   DOT device with function
                code of 1001
JSX DELAY
DOT DEV, DISC1
```

5-15.5 Function Codes

The various 8-bit control codes are illustrated in hexadecimal and binary formats in Table 5-56.

5-15.6 Status Codes

Status is returned to the accumulator from the buffered input/output channel after a DIN 6, 0 or a DIN 7, 0 command is issued, respectively. There are three interrupts associated with both the input channel and the output channel (Table 5-57). All six interrupts, when true, denote that a given channel is interrupting and is ready for a data transfer.

5-15.7 Interrupt Meaning

An interrupt from the buffered input channel means that there is data in one or more of the three

Table 5-56. BIC and BOC Commands and Function Codes

Command	Hexadecimal Notation	Function Codes							
		Binary Notation							
		Bit Position							
DOT		8	9	10	11	12	13	14	15
Transfer Data & Disconnect Channel 1	71	0	1	1	1	0	0	0	1
Transfer Data & Disconnect Channel 2	72	0	1	1	1	0	0	1	0
Transfer Data & Disconnect Channel 3	73	0	1	1	1	0	0	1	1
Transfer Data & Disconnect Channel 4*	74	0	1	1	1	0	1	0	0
Transfer Data on Channel 1	79	0	1	1	1	1	0	0	1
Transfer Data on Channel 2	7A	0	1	1	1	1	0	1	0
Transfer Data on Channel 3	7B	0	1	1	1	1	0	1	1
Transfer Data on Channel 4*	7C	0	1	1	1	1	1	0	0
DIN									
Return Status BOC to Accumulator	70	0	1	1	1	0	0	0	0
Return Status BIC to Accumulator	60	0	1	1	0	0	0	0	0
Input Data on Channel 1	61	0	1	1	0	0	0	0	1
Input Data on Channel 2	62	0	1	1	0	0	0	1	0
Input Data on Channel 3	63	0	1	1	0	0	0	1	1
Input Data on Channel 4*	64	0	1	1	0	0	1	0	0

*For 4th channel operation see Section 5-15.9.

buffers awaiting transfer to the accumulator. Status must be tested to determine which buffers (1, 2 or 3) are causing the interrupt.

An interrupt from the buffered output channel means that there is no data in one or more of the three buffers and the buffer(s) are awaiting a data transfer from the accumulator. Status must be tested to determine which buffers (1, 2 or 3) are causing the interrupt.

5-15.8 Timing

Theoretically the buffered input/output channels can transmit data as fast as the computer can send and receive data (3.6 u sec.). But practically, the computer program must load/store the data and it must interrogate the status, process the interrupt, or interrogate the external sense line in addition to performing a DOT or DIN instruction. Therefore, the maximum practical transfer rate is approximately 159 KHz. (See Figure 5-27).

5-15.9 Buffered Input Channel or Buffered Output Channel

If the Buffered Input/Output channels (Models 77621-2-3-4-5) are ordered, there is the capability

of having up to three input channels and three output channels.

If the Buffered Input Channel (Models 77601-2-3-4) or the Buffered Output Channels (Models 77611-2-3-4-5-6-7-8) are ordered separately, there is the capability of having up to four input channels and four output channels.

The Buffered Output Channels may be ordered with relay contact outputs instead of DTL circuitry outputs. Channels must be either all relay or all DTL outputs.

Table 5-57. BIC and BOC Status

Meaning when Bit True	Bit No.
Channel 1 interrupting	0
Channel 2 interrupting	1
Channel 3 interrupting	2
Channel 4 interrupting*	3

*For 4th channel operation see Section 5-15.9.

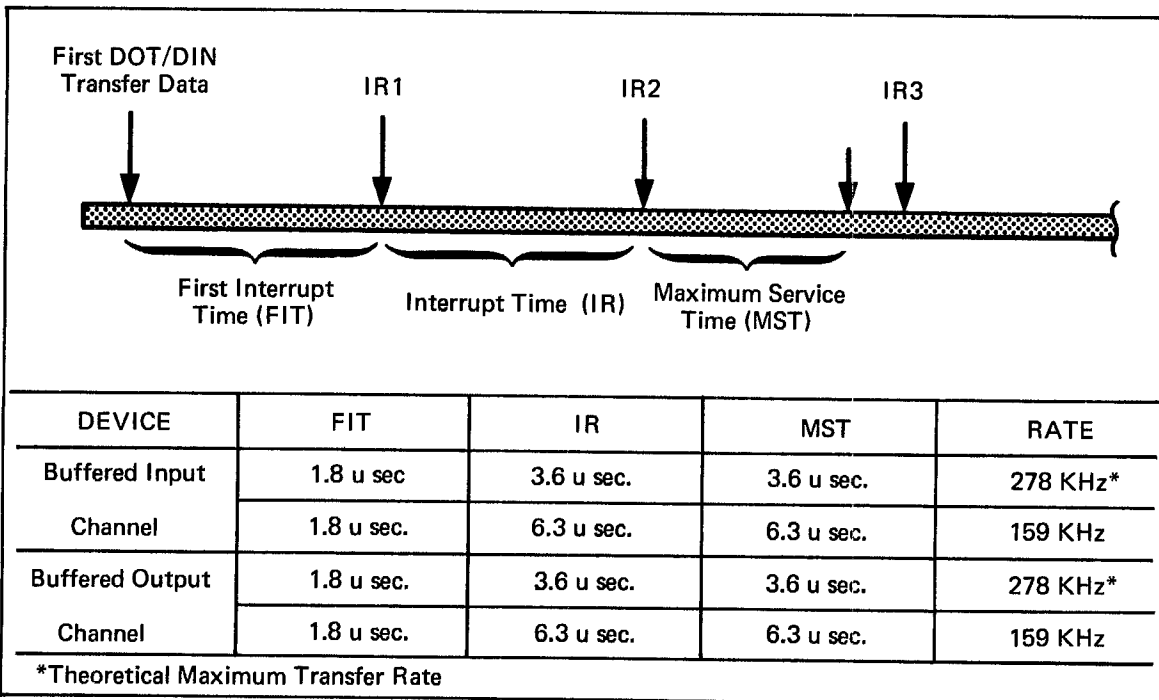


Figure 5-27. BIC & BOC Timing

5-16 ANALOG TO DIGITAL CONVERTERS

Raytheon's exclusive MINIVERter™ and MULTIVERter^R are provided as standard analog "front end" instruments to the 706 computer. Each includes a multiplexer, sample-and-hold amplifier and an analog-to-digital converter in a single compact unit. Conversion rates range from 15 to 100 KHz and resolution from 10 to 15 bits in binary or BCD modes. Up to 256 input channels are provided. Accuracy varies from 0.1% to 0.01% ± 1/2 LSB, depending upon device used. See Tables 5-58 and 5-59 for specifications.

5-16.1 Output Codes

All A/D converters can be supplied with any of the output codes shown in Table 5-60. Except for sign and magnitude all of the codes shown can be achieved by adjustments or simple component changes. Sign and magnitude codes require installation of an additional pair of circuit cards.

5-16.2 Analog to Digital Function Codes

Both the miniverter and multiverter use the same function codes and both converters are programmed in an identical manner.

5-16.2.1 Disconnect (DOT 3, 0)

The disconnect command performs the same functions in initializing the interface as pressing the reset switch. The interface clock is turned off, the error flip-flop resets, and no more interrupts occur.

5-16.2.2 Reset Error (DOT 3, 1)

The reset error command is included in the instruction set to allow more versatility in methods of monitoring and resetting rate errors. This command functions the same as the set mode command, except for two significant differences. The first is that the error flip-flop is reset. The second, and more important, difference is that the reset error command has no effect on the state of the multiverter or miniverter start pulse logic. Thus, the mode, clock, and address lines may be modified without allowing a start pulse to be generated.

5-16.2.3 Set Random Mode and Address Unlocked (DOT 3, 2)

Whenever the miniverter or multiverter is set to the

random mode, bits 08 through 15 of the accumulator specify the address (channel 0 - 255). The random mode causes digitized data to be input from one channel only. When the random mode is selected, it can only be changed by executing another DOT instruction.

The unlocked mode causes interrupts to occur immediately after analog to digital conversion. Rate errors can occur when data is requested too fast (before the interrupt) in the unlocked mode.

The start pulse given with this command causes an interrupt to occur when the selected channel has digitized the analog signal.

5-16.2.4 Set Sequential Mode Unlocked (DOT 3, 6)

Whenever the miniverter or multiverter is set to the sequential mode, digitized data is input from each channel sequentially. A start pulse causes the analog signal from the channel previously set by the Set Random Mode and Address Command to be digitized. All subsequent Transfer Data and Start commands cause the analog signals to be digitized from the next higher channel. This process continues until the last channel has been read; with the next start pulse forcing the address back to channel 0 or stopping when a Transfer Data (no start pulse) command is issued.

The unlocked mode causes interrupts to occur immediately after analog to digital conversion.

Rate errors can occur when data is requested too fast (before the interrupt) in the unlocked mode.

The start pulse given with this command causes an interrupt to occur when the previously selected channel has digitized the analog signal.

5-16.2.5 Set Random Mode and Address Clocked (DOT 3, A)

Whenever the miniverter or multiverter is set to the random mode, bits 08 through 15 of the accumulator specify the address (channel 0 - 255). The random mode causes digitized data to be input from one channel only. When the random mode is selected, it can only be changed by executing another DOT instruction.

Table 5-58. Miniverter Specifications

	10-Bit Binary Model MADC 10-06		12-Bit Binary Model MADC 12-06		3-Digit BCD Model MADC 13-06	
	With MSH1	With MSH2	With MSH1	With MSH2	With MSH1	With MSH2
ANALOG INPUT						
Input Range	±10V	±10V	±10V	±10V	±10V	±10V
Maximum Overvoltage (without damage)	±15V	±15V	±15V	±15V	±15V	±15V
Channel Capacity (8 Chls/card)	128	128	128	128	128	128
INPUT IMPEDANCE						
"ON" Channel	25 megohms	100 megohms	25 megohms	100 megohms	25 megohms	100 megohms
"OFF" Channel	1000 megohms	1000 megohms	1000 megohms	1000 megohms	1000 megohms	1000 megohms
Offset Current	4 uA	200 nA	4uA	200 nA	4 uA	200 nA
Source Impedance (Max.)	1000 ohms	1000 ohms	1000 ohms	1000 ohms	1000 ohms	1000 ohms
Aperture Time	100 nsec	50 nsec	100 nsec	50 nsec	100 nsec	50 nsec
Crosstalk (1K source at 400 Hz, 16 chls.)	70 db	70 db	70 db	70 db	70 db	70 db
Adjustment Range	320 mV	320 mV	80 mV	80 mV	160 mV	160 mV
PERFORMANCE						
Resolution	9 Bits + Sign (Approx. 20 mV/Count)		11 Bits + Sign (Approx. 5 mV/Count)		3 BCD + Sign (Approx. 10 mV/Count)	
Conversion Time	9.8 usec	9.8 usec	18.5 usec	18.5 usec	12 usec	12 usec
Sample and Settle Time	20 usec	12.5 usec	28.5 usec	22.5 usec	22 usec	15 usec
Max. Throughput Rate	50 KHz	80 KHz	35 KHz	45 KHz	45 KHz	65 KHz
Accuracy at 25°C (Includes linearity, noise, gain errors, and dynamic effects)	0.05% ± 1 LSB		0.065% ± 1 LSB		0.065% ± 1 LSB	
Temperature Range	0° to 50° C		0° to 50° C		0° to 50° C	
Temperature Coefficient	40 PPM/° C (0.8 mV/° C)		40 PPM/° C (0.8 mV/° C)		40 PPM/° C (0.8 mV/° C)	
Max. Long Term Drift	0.1% for 30 days		0.1% for 30 days		0.1% for 30 days	
DIGITAL						
Output Code	Binary, 2's Complement		Binary, 2's Complement		BCD, 10's Complement	
POWER REQUIRED						
Voltage	115 V ±10%		115 V ±10%		115 V ±10%	
Frequency	50 to 60 Hz		50 to 60 Hz		50 to 60 Hz	
Power	Approx. 30 watts (Fused - 1 Amp S.B.)		Approx. 30 watts (Fused - 1 Amp. S.B.)		Approx. 30 watts (Fused - 1 Amp. S.B.)	

Table 5-59. Multiverter Specifications

<p>Analog Input Channel Capacity (16 channels/card) 96 channels w/o attenuators 48 channels w/attenuators Crosstalk 80 db with 1K source at 400 Hz Aperture Time 50 nanoseconds Input Range ±10 volts full scale w/o attenuators ± 100 volts full scale w/attenuators Input Impedance (w/o attenuators) 100 meg. ohms min. for selected channel 1000 meg. ohms min. for unselected channel (w/attenuators) 20K ohms (±0.02%) for either selected or unselected channels Max. Source Impedance 1000 ohms for specified performance w/o attenuators Max. Voltage Overload⁽²⁾ 100% full scale w/o attenuators 25% of full scale w/attenuators Performance Throughput Rate See Table 5-59.1 Accuracy at DC Linearity 0.01 ± ½ LSB Long Term Drift 0.01% Typical Temp. Coefficient 10 ppm/°C max. Crosstalk 80db with 1K source at 400 cps⁽¹⁾ Voltage Ref. Stability ... 0.001% Typical Accuracy at 25°C will be within 0.02% (± ½ LSB) of full scale.</p>	<p>Temperature Temperature Range 0 to 50° C Warm up Time 10 min. to rated accuracy Cooling Internal-muffin fan Temperature Coefficient ... 10 ppm/°C max. Mechanical Analog Inputs Malco Connector Panel (wired for 64 channels). Digital Inputs (mate) Amphenol Type 17-20500-1 or Cannon Type DD-50P Display Amperex Indicators for A/D Register Size (See Page 15) 19" slide mounted drawer 22" deep 5½" high Weight 60 pounds (typ.) Power 105 to 125 vac 60 to 400 cps (Fused at 3 Amp S.B.)</p> <p>GENERAL NOTES ON MULTIVERTER SPECIFICATIONS (1) Crosstalk is measured as follows: A 400 cps full scale input voltage is applied through a 1000 ohm source impedance to all channels except the one being monitored. The 400 cps noise is measured on the selected channel which has 0 volts applied through a 1000 ohms source impedance. (2) No more than 2 channels can simultaneously be overloaded (by 100%) for the multiverter to operate to rated specs.</p>
---	--

Table 5-59.1. Multiverter Throughput Rates

Resolution	Model Number (Binary Instruments)	Bit Rate	Nominal A/D Conversion Rate	Nominal Multiverter Throughput ** Rate
10 Bits*	ADC21-10B-B	1.2 Microsec/bit	75KC	55KC
	ADC23-10B-B	2.0 Microsec/bit	45KC	38KC
11 Bits*	ADC21-11B-B	1.2 Microsec/bit	69KC	52KC
	ADC23-11B-B	2.0 Microsec/bit	41KC	35KC
12 Bits*	ADC21-12B-B	1.2 Microsec/bit	64KC	50KC
	ADC23-12B-B	2.0 Microsec/bit	38KC	33KC
	ADC25-12B-B	3.5 Microsec/bit	22KC	21KC
13 Bits*	ADC22-13B-B	2.0 Microsec/bit	35KC	31KC
	ADC24-13B-B	3.5 Microsec/bit	20KC	19KC
14 Bits*	ADC22-14B-B	2.0 Microsec/bit	33KC	30KC
	ADC24-14B-B	3.5 Microsec/bit	19KC	18KC
15 Bits*	ADC20-15B-B	2.0 Microsec/bit	31KC	27KC
	ADC24-15B-B	3.5 Microsec/bit	17KC	17KC
15 Bits*	ADC21-15B	1.0 Microsec/bit	75KC	50KC
3 BCD Plus Sign (8-4-2-1 Code)	ADC22-13D-B	2.0 Microsec/bit	35KC	31KC
	ADC24-13D-B	3.5 Microsec/bit	20KC	19KC

*Includes Sign
 **Throughput Rate for entire Multiverter when using the A/D Converter shown.

Table 5-60. Available Output Codes

Binary								
	2's Complement		1's Complement		*Sign and Magnitude		Unipolar	
	Sign		Sign		Sign		Sign	
+ Full Scale (-1 Count)	0	1111111111	0	1111111111	0	1111111111	1	1111111111
+ ½ Full Scale	0	1000000000	0	1000000000	0	1000000000	1	0000000000
Zero (+1 Count)	0	0000000001	0	0000000001	0	0000000001	0	0000000001
Zero	0	0000000000	0	0000000000	0	0000000000	0	0000000000
Zero (-1 Count)	1	1111111111	1	1111111110	1	0000000001	Not	
- ½ Full Scale	1	1000000000	1	0111111111	1	1000000000		
-Full Scale (+1 Count)	1	0000000001	1	0000000000	1	1111111111	Applicable	

Binary Coded Decimal							
	10's Complement		9's Complement		*Sign and Magnitude		Unipolar
	Sign		Sign		Sign		Sign
+Full Scale (-1 Count)	0	1001 1001 1001	0	1001 1001 1001	0	1001 1001 1001	1001 1001 1001
+ ½ Scale	0	0101 0000 0000	0	0101 0000 0000	0	0101 0000 0000	0101 0000 0000
Zero (+1 Count)	0	0000 0000 0001	0	0000 0000 0001	0	0000 0000 0001	0000 0000 0001
Zero	0	0000 0000 0000	0	0000 0000 0000	0	0000 0000 0000	0000 0000 0000
Zero (-1 Count)	1	1001 1001 1001	1	1001 1001 1000	1	0000 0000 0001	Not
-½ Full Scale	1	0101 0000 0000	1	0100 1001 1001	1	0101 0000 0000	⋮
Full Scale (+1 Count)	1	0000 0000 0001	1	0000 0000 0000	1	1001 1001 1001	Applicable

*Extra Cost Option

The clocked mode causes interrupts to occur after analog to digital conversion and at clock time only.

The start pulse given with this command causes an interrupt to occur when the selected channel has digitized the analog signal and clock time has occurred.

5-16.2.6 Set Sequential Mode Clocked (DOT 3, E)

Whenever the miniverter or multiverter is set to the sequential mode, digitized data is input from each channel sequentially. A start pulse causes the analog signal from the channel previously set by the set Random Mode and Address command to be digitized. All subsequent Transfer Data and Start commands cause the analog signal to be digitized from the next higher channel. This process continues until the last channel has been read; with the next start pulse forcing the address back to channel 0 or stopping when a Transfer Data (no start pulse) command is issued.

The clocked mode causes interrupts to occur after analog to digital conversion and at the next clock time.

Rate errors can occur when data is requested too fast (before the interrupt) in the clocked mode.

The start pulse given with this command causes an interrupt to occur when the previously selected channel has digitized the analog signal and the next clock pulse occurs.

5-16.2.7 Transfer Data and Start (DIN 3, 3)

When this code is received, data is transferred to the accumulator and a new start pulse is generated. Data is transferred to selected bits of the accumulator according to the precision of the miniverter or multiverter (Figure 5-28).

After the data has been transferred, the start pulse causes the signal on the same channel (random mode) or the next higher numbered channel (sequential mode) to be digitized. An interrupt is generated either immediately (unclocked mode) or at the next clock time after digitizing has been completed (clocked mode).

5-16.2.8 Transfer Data (DOT 3, 1)

When this code is received, data is transferred to

the accumulator but no new start pulse is given. Data is transferred to selected bits of the accumulator according to the precision of the miniverter or multiverter (Figure 5-28). This command is used in the sequential mode when the last channel has been read and immediate digitizing of the first channel is not desired, or in the random mode when the next desired channel is different than the presently addressed channel.

5-16.3 Function Codes

Table 5-61 is a summary of the function codes for both the multiverter and miniverter.

5-16.4 Status Codes

Status is returned to the accumulator from the miniverter or multiverter after a DIN 3, 0 instruction is issued.

The status word from the analog to digital equipment consists of three bits (Table 5-62). Bit-0, the busy bit, goes true at the leading edge of the start pulse and remains true until the specified channel has been digitized and is ready to be read. It is by monitoring this bit that maximum burst rates may be obtained.

Bit 14 is the frame sync bit. This bit goes true and remains true while channel 0 is being digitized.

Bit-15 shows the condition of the error flip-flop.

This flip-flop is set when two consecutive start pulses have occurred and data has not been accepted from the miniverter or multiverter. It is merely an indicator. The data is still available at the converter and may still be read. The error flip-flop is reset with a DOT 3, 1 command.

5-16.5 Interrupt Meaning

In the unlocked mode, an interrupt means that data has been digitized and is ready for transfer to the accumulator. In the clocked mode, an interrupt means that the data has been digitized and the clock signal has gone high.

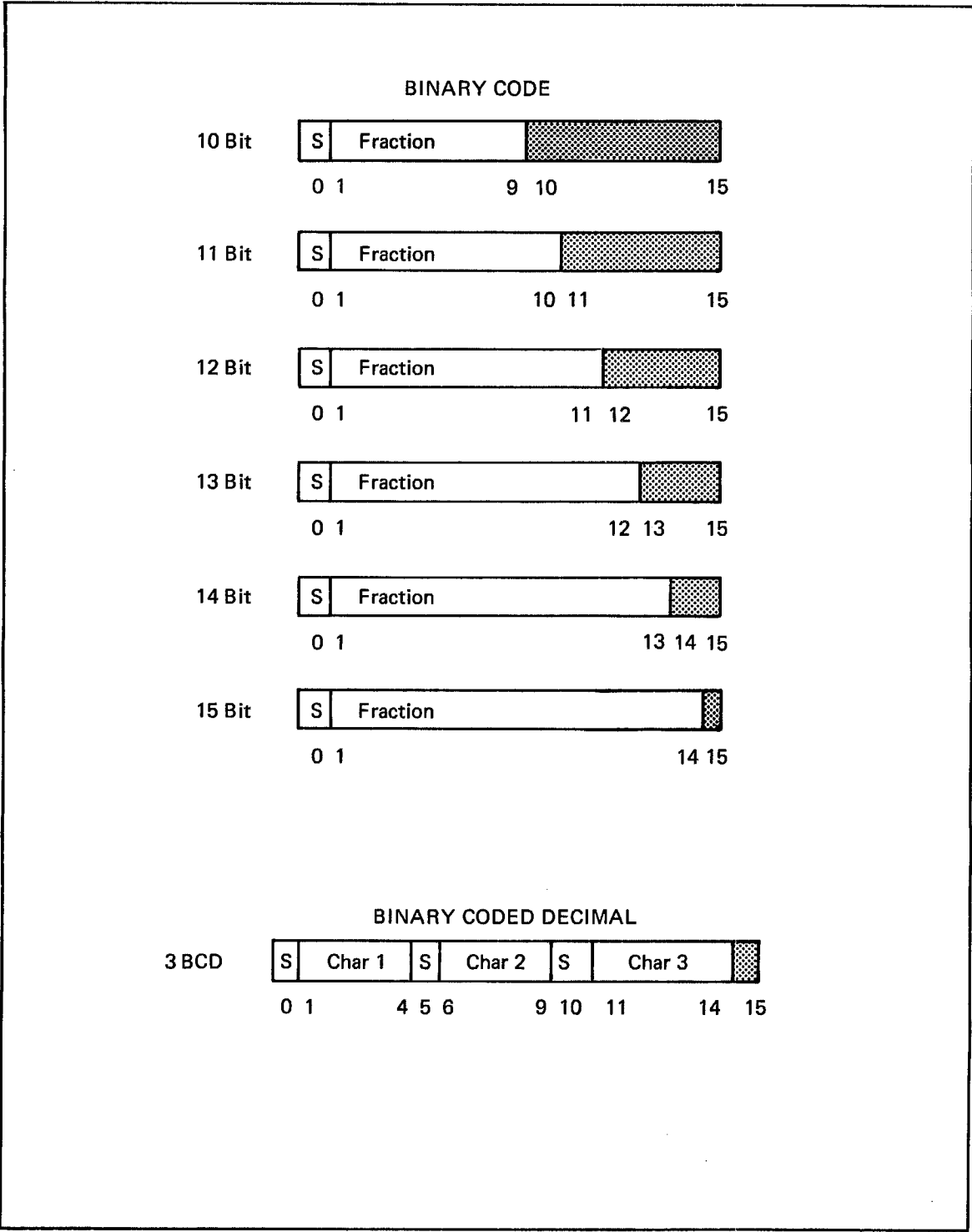


Figure 5-28. Analog to Digital Word Formats

Table 5-61. Analog to Digital Converter Function Codes

COMMAND		FUNCTION CODES							
DOT	Hexadecimal Notation	Binary Notation							
		8	9	10	11	12	13	14	15
Disconnect Controller	30	0	0	1	1	0	0	0	0
Reset Error	31	0	0	1	1	0	0	0	1
Set Random Mode & Address Unlocked	32	0	0	1	1	0	0	1	0
Set Sequential Mode Unlocked	36	0	0	1	1	0	1	1	0
Set Random Mode & Address Clocked	3A	0	0	1	1	1	0	1	0
Set Sequential Mode Clocked	3E	0	0	1	1	1	1	1	0
DIN									
Return Status to Accumulator	30	0	0	1	1	0	0	0	0
Transfer Data	31	0	0	1	1	0	0	0	1
Transfer Data and Start	33	0	0	1	1	0	0	1	1

Table 5-62. Analog to Digital Converter Status

Meaning When Bit True	Bit No.
Controller Busy (Channel being digitized)	0
Frame Sync	14
Rate Error	15

5-16.6 Timing

Figure 5-29 shows some sample analog to digital converter timing characteristics. Those shown are for the unlocked mode only. Clocked mode timing will depend on the clock rate selected for the miniverter or multiverter.

Internal miniverter and multiverter timing can be found in bulletins SP-286 and SP-205, respectively.

5-16.7 Miniverter Operation (Model No. 77631, 77632, 77633, 77634, 77635)

The miniverter can be operated in either a manual

mode or a normal mode. In the manual mode, the unit is under control from the optional front panel switches. Any channel can be manually selected and a conversion initiated on that channel. Indicators display the results of each conversion and additional front panel controls provide means for short-cycling the ADC to fewer bits of resolution to achieve higher throughput rates. Other front panel controls allow the user to establish the last channel in any given sequence.

5-16.7.1 Controls and Indicators

The controls and indicators for the miniverter are shown in Figure 5-30 and described in Table 5-63.

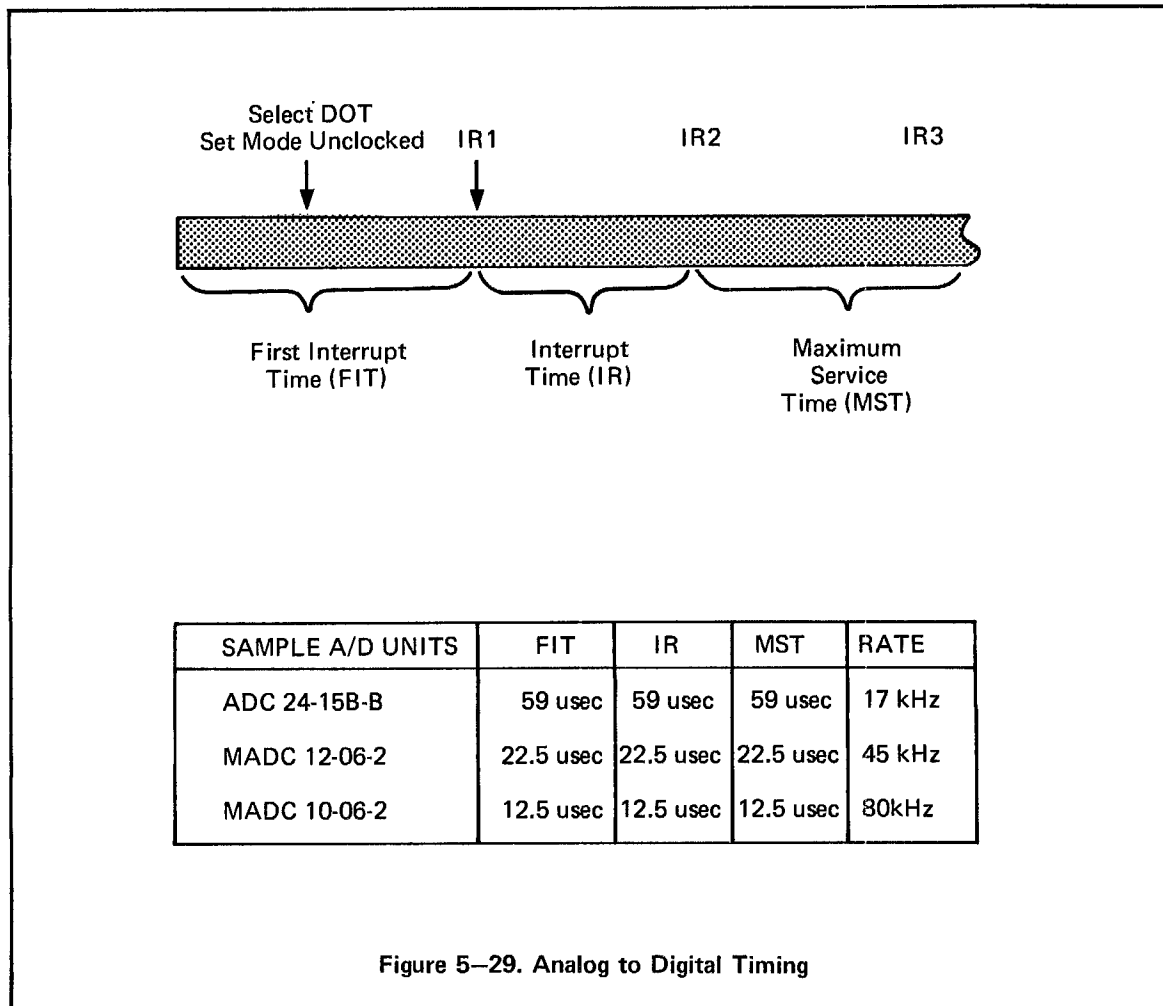


Figure 5-29. Analog to Digital Timing

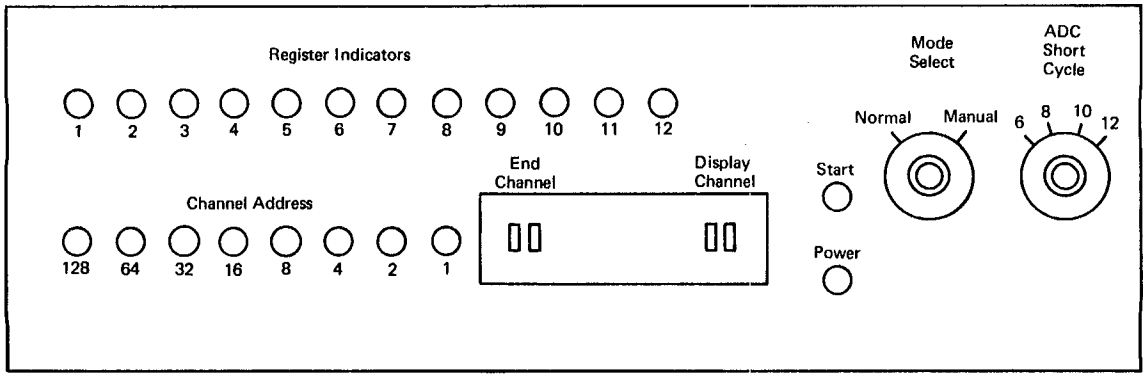


Figure 5-30. Miniverter Control Panel

Table 5-63. Miniverter Controls and Indicators

Control or	Function
DISPLAY CHANNEL thumbwheel switches	Selects one of 256 channels for digitizing and display.
REGISTER indicators	Displays digitized data of channel indicated in CHANNEL ADDRESS lights.
CHANNEL ADDRESS indicators	Indicates channel number being digitized.
END CHANNEL thumbwheel switches	Selects last channel to be digitized when in the sequence mode.
POWER switch/indicator	Supplies power to miniverter. Illuminated red when on.
START push-button switch	Generates a start pulse when in the manual mode only.
MODE SELECT switch	Selects either the normal or manual mode.
ADC SHORT CYCLE switch	Selects the number of significant bits that the analog signal is to be converted to. Decreasing the number of bits increases the throughput rate.

5-16.8 Multiverter Operation (Model No. 77636)

The multiverter can be operated in either a calibrate mode or a normal mode. In the calibrate mode, the unit is under control from the optional front panel switches. Any channel can be manually selected and a conversion initiated on that channel. Indicators

display the results of each conversion.

5-16.8.1 Controls and Indicators

The controls and indicators for the multiverter are shown in Figure 5-31 and described in Table 5-64.

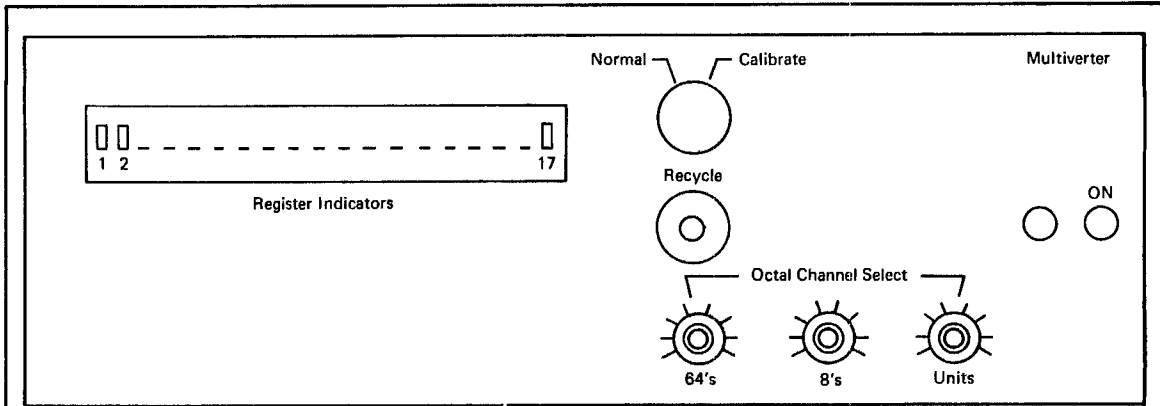


Figure 5-31. Multiverter Control Panel

Table 5-64. Multiverter Controls and Indicators

Control or Indicator	Function
OCTAL CHANNEL SELECT switches	Select the channel to be digitized in the calibrate mode.
REGISTER indicators	Display digitized data of channel indicated by OCTAL CHANNEL SELECT switches.
NORMAL/CALIBRATE switch	Selects either the on-line or off-line mode.
RECYCLE button	Generates a start pulse when in the calibrate mode.
ON switch and indicator	Supplies power to the multiverter. Indicator is illuminated white when power is on.

5-17 DIGITAL TO ANALOG CONVERTER

The 706 DAC may contain from one to 64 separately addressable digital-to-analog converters (which are referred to as converter channels to distinguish them from the DAC that they are a part of and the DAC module that they contain). The converter channels have a resolution of 10 binary bits (9 magnitude bits plus sign) and an output voltage range of -10.000v to +9.980v. Accuracy is 0.1% at 25°C. (Table 5-65).

Each converter channel contains its own 10-bit data storage register, and the analog output voltage continuously represents the contents of the register.

Digital data is transmitted to the channel along with the channel address via the computer data input-output (DIO) bus.

The 706 DAC is made up of standard Raytheon M-Series integrated-circuit modules and is housed in one or more MC40 module cases. The basic DAC chassis contains the computer interface logic and up to eight converter channels. Additional converter channels are housed in DAC expansion chassis with capacities of up to eight channels each. The maximum configuration (64 channels) is thus housed in one basic DAC chassis and seven DAC expansion chassis.

Table 5-65. Digital to Analog Converter Specifications

Number of channels	1 to 64
Resolution	10 bits (9 bits + sign)
Output voltage	-10.000 to +9.980v
Output current	5 ma at 10v
Output impedance (dc)	0.1 ohm
Output noise	2 mv p-p (max)
Settling time	25 usec for 10v excursion to 0.1%
Accuracy	0.1% at 25°C
Temperature coefficient	50 ppm/°C
Temperature range	15 to 50°C
Input code	Binary

5-17.1 Speed

The throughput rate of the DAC is approximately 40 KHz. The settling time for a 10 v excursion to 0.1% is 25 u sec.

5-17.2 Data Word Format

A 16-bit data word (Figure 5-32) specifying the 10-bit binary-coded data and the 6-bit address of converter channel that is to receive the data must be placed into the CPU accumulator prior to executing the DOT instruction to transfer the word to the 706 DAC.

5-17.3 Digital to Analog Function Code (DOT 4, 0-F)

One program instruction is used to communicate with the DAC. This is a DOT instruction used to transmit a channel address and data word to the DAC. The function code field is not required or used by the DAC.

The DAC provides no interrupts and no status response.

5-17.4 Converter Input Code

The converter input code is a nonsignificant binary code derived from the 2's complement binary number n in the relationship:

$$E_{ANAL} = \left(\frac{n}{2} \right) 10V$$

The functional relationship between the binary input code to the converter (N) and the number n is:

$$N = (2^9 - 1) - n$$

Since the number n is a 10-bit signed binary number (sign plus nine magnitude bits) with values from -2^9 to $+(2^9 - 1)$, input code N is an unsigned 10-bit binary number with values from 0 to $(2^{10} - 1)$. Examples of input codes for various values of the number n and the corresponding analog output voltages are given in Table 5-66.

The computer program must make provision for converting 2's complement binary data to the converter input code prior to combining it with the channel number and transmitting it to the DAC. Any mathematical manipulations that must be performed on the data must be performed before it is encoded for use in the DAC.

5-17.5 Status and Interrupt

None available for DAC.

5-17.6 Timing

See Section 5-17.1.

5-17.7 Digital to Analog Converter Operation (Model Nos. 77641, 77642, 77643, 77644)

There are no panel controls or indicators for the 706 DAC. Power is turned on and off with the computer power. The 706 DAC operates automatically under control of the computer, and no operating instructions are required.

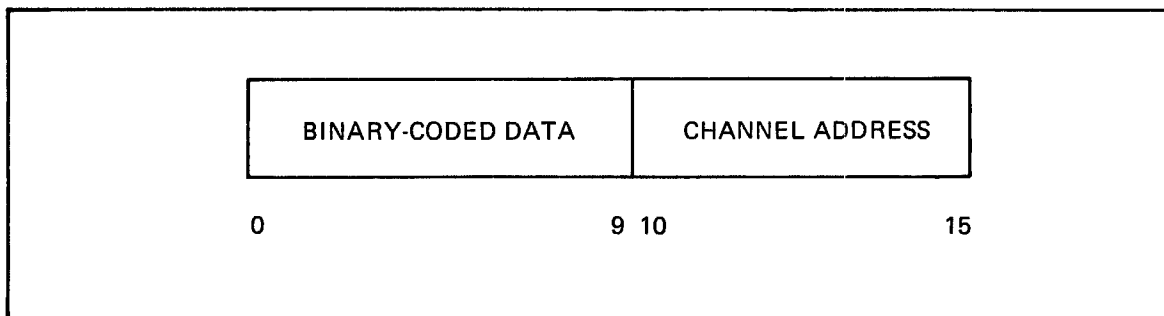


Figure 5-32. Digital to Analog Converter Data Word Format

Table 5-66. Digital to Analog Converter Input Codes, Analog Values, Output Voltages

Input Code (N) (= 511-n)	Analog Value (n) (= 511-N)	Analog Output Voltage (vdc)
0 000 000 000	0 111 111 111	+9.980
0 000 000 000	0 111 111 111	+9.980
0 011 111 111	0 100 000 000	+5.000
0 101 111 111	0 010 000 000	+2.500
0 110 111 111	0 001 000 000	+1.250
0 111 011 111	0 000 100 000	+0.625
0 111 101 111	0 000 010 000	+0.3125
0 111 110 111	0 000 001 000	+0.156
0 111 111 011	0 000 000 100	+0.078
0 111 111 101	0 000 000 010	+0.039
0 111 111 110	0 000 000 001	+0.0195
0 111 111 111	0 000 000 000	0.000
1 000 000 000	1 111 111 111	-0.0195
1 000 000 001	1 111 111 110	-0.039
1 000 000 011	1 111 111 100	-0.078
1 000 000 111	1 111 111 000	-0.156
1 000 001 111	1 111 110 000	-0.3125
1 000 011 111	1 111 100 000	-0.625
1 000 111 111	1 111 000 000	-1.250
1 001 111 111	1 110 000 000	-2.500
1 011 111 111	1 100 000 000	-5.000
1 111 111 111	1 000 000 000	-10.000

5-18 TIME OF DAY CLOCK

The Time-of-Day Clock provides the time-of-day in 32 bits of BCD information with resolution (lsb) of one millisecond plus an optional extended resolution to 100 microseconds by increasing the clock output to 36 bits.

The Computer Time Set option allows 32 bits of clock indication to be set directly by the central

processor. When used with the overflow interrupt, this feature provides an interval timer function. The time of day clock specifications are listed in Table 5-67.

5-18.1 Speed

The real-time clock can be operated at the memory speed of the 706 computer.

Table 5-67. Time-of-Day Clock Specifications

Type	Crystal-controlled
Accuracy	0.001 percent at 25° C
Stability	0.003 percent (-30° to +60° C)
Resolution	1 ms (std) 100 usec (opt)
Range	0 to 23 hr, 59 min, 59.999 sec (std) 0 to 23 hr, 59 min, 59.9999 sec (opt)
Clock output	32 BCD bits (std) 36 BCD bits (opt)
Time set	
Manual	0 to 23 hr, 59 min, 59.999 sec (std)
Computer	0 to 23 hr, 59 min, 59.999 sec (opt)
Time display	32 BCD indicators
Control (start, stop, reset)	Manual or computer
Clock hold limitations	
Maximum hold period for read operation without loss of accuracy	60 usec
Minimum period between end of read operation and start of next hold	20 usec
Logic levels (nominal)	+5v, true (1) 0v, false (0)
Power requirements	115 vac, 60 cps, 1 amp

5-18.2 Data Word Formats

The formats of the data words used to read or set the time are shown in Figure 5-33.

5-18.3 Time-of-Day Clock Functions

The computer may start, stop, and reset the clock, set the time (if the direct computer time set option is installed), and read the time.

Except when used as an interval counter, the clock sends no interrupts to the computer, and there is no provision for selecting or disconnecting the clock. There is also no provision for response to a status request because the clock is considered always ready.

5-18.3.1 Stop Clock (DOT B, 4)

This command stops the clock from counting. The clock must be stopped in order to set the time into the clock. This command performs the same function under CPU control as the HALT button does on the control panel.

5-18.3.2 Reset Clock (DOT B, 5)

The reset command does not stop the clock. If the clock is stopped when the reset command is given the counter will be cleared to 0. If the clock is not stopped, the counter will be cleared to 0 but will continue to count.

The clock must be reset in order to set the time into the clock. This command performs the same function as the RESET button on the control panel.

5-18.3.3 Start Clock (DOT B, 6)

This command starts the clock counter counting up. The clock counter consists of 10 BCD stages that express time in hours, minutes, seconds, and decimal fractions of a second from 0 to 23 hr., 59 min., 59.9999 sec. Those stages that are not required to count to 9 (10 sec., 10 min., and 10 hr.) are *short counted* so that they return to zero after reaching the maximum count (5 in the case of 10 sec. and 10 min., and 2 in the case of 10 hrs.). The remaining stages count from 0 to 9 and repeat. The number of bits per stage, therefore, varies from two to four, depending on the maximum count of the stage.

This command performs the same function as the RUN button on the control panel.

5-18.3.4 Write Word 1, Write Word 2 (DOT B, 1; DOT B, 2)

If the direct computer time set option is installed in the clock, the 32 most significant bits of the clock time may be set by transmitting two 16-bit data words from the computer. The clock must be stopped and reset prior to setting the time.

The data words from the computer (Figure 5-33) are designated word 1 and word 2. However, either word or both words may be transmitted, and the order in which they are transmitted is immaterial.

The write word 1 command gates the contents of the accumulator into the following counters: 10 hr., 1 hr., 10 min., 1 min., and 10 sec. The write word 2 command gates the contents of the accumulator into the following counters: 1 sec., 100 msec., 10 msec., and 1 msec. The four least significant bits of the counter (0.1 msec. stage) cannot be set by the computer and is initially zero after issuing a reset command.

The clock may thus be set to anytime from 0 hr., 0 min., 0.000 sec. to 23 hr., 59 min., 59.999 sec.

5-18.3.5 Hold Clock, Read Word 1, Read Word 2, Read Word 3 (DOT B, 0; DIN B, 1; DIN B, 2; DIN B, 3)

The outputs of all counter stages except the 0.1-msec. stage are continuously displayed on the front panel of the clock and may also be read by the computer.

Unlike the computer time set operation, the computer time read operation must be performed in the correct sequence to avoid introducing errors in the time or holding the clock indefinitely. The read operation must be started by executing a hold clock instruction, which must be followed by one or more read word instructions. The least significant word must be read in each read operation in order to end the hold condition established by the hold clock instruction. The other word (or words) may or may not be read, depending upon the application. If they are read, they must be read before the least significant word is read. Failure to read the least significant word will cause the time to be held indefinitely.

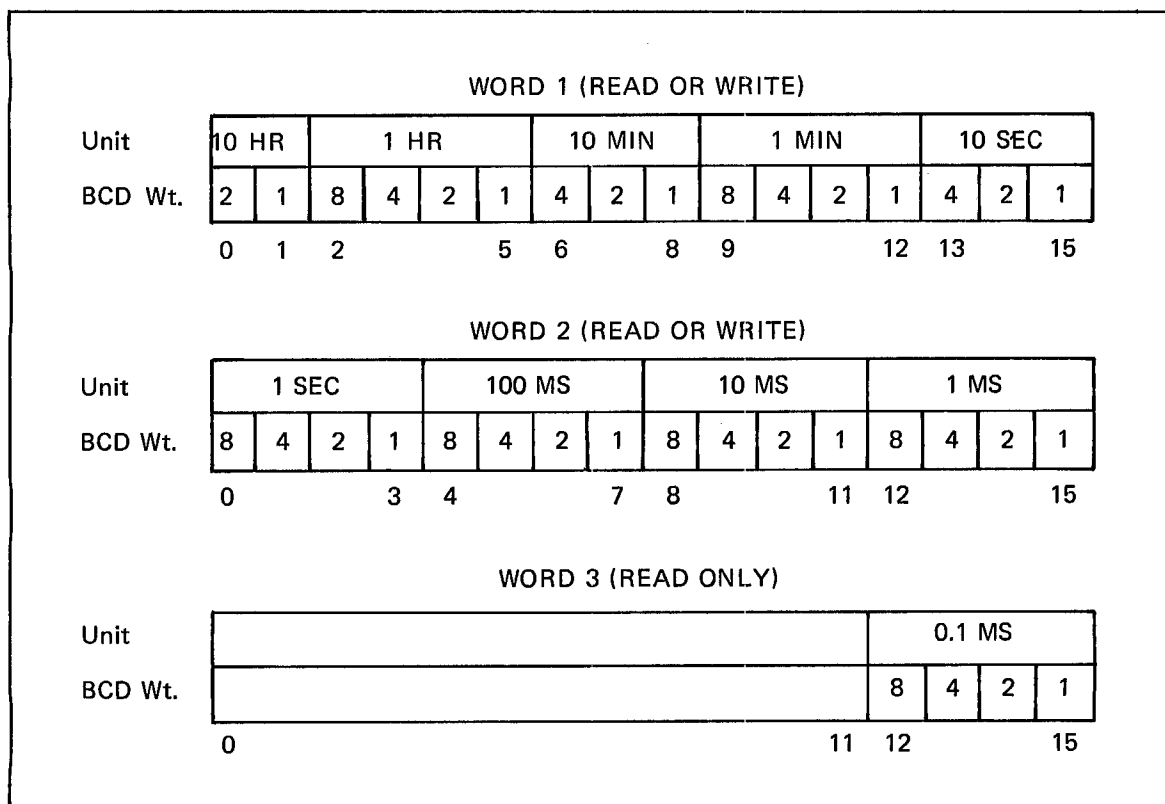


Figure 5-33. Time-of-Day Clock Data Word Formats

In order to not affect the clock accuracy during a read operation (the clock is held from counting), the entire two (or three) word read operation (including the hold clock command) must be completed within 60 usec. An entire 3-word read operation, including two STW instructions for the first two words can be completed within 10.8 u sec. if no DMA memory requests intervene.

The read word 1 command transfers the contents of the following counters into the accumulator (Figure 5-33): 10 hr., 1 hr., 10 min., 1 min., and 10 sec. The read word 2 command transfers the contents of the following counters into the accumulator: 1 sec., 100 msec., 10 msec., and 1 msec. The read word 3 command (if installed) transfers the 0.1 msec. counter to bits 12-15 of the accumulator, with bits 0-11 being set to zero.

After the termination of a computer time read operation, a new read operation should not be initiated before 20 u sec. has elapsed.

5-18.4 Function Codes

The various 8-bit control codes are illustrated in hexadecimal and binary formats in Table 5-68.

5-18.5 Status Codes

Since the Time-of-Day Clock is always considered ready, there is no status response.

5-18.6 Interval Timing

The clock may be wired to provide interrupts to the computer at fixed intervals. The intervals that may be selected and the required jumper connections are covered in Table 5-69. Other than starting, stopping, and resetting the clock, the computer has no control over this operation.

By combining the clock control and time set functions with the interval timing function, the clock

may be used to time nonrepetitively any interval less than a selected fixed interval. For this function, the clock must be stopped, reset, and preset to the difference between the desired interval and the fixed interval for which it is wired.

Timing of the desired interval is initiated by a clock start command from the computer, and the interrupt is sent to the computer when the clock advances from the preset time to the fixed interval time. Consecutive intervals may not be timed in this

manner because the clock must be stopped, reset, preset, and started for each such interval.

5-18.7 Interrupt Meaning

When the Time-of-Day Clock is used for recording the time-of-day, there is normally no interrupt associated with its operation. When the time-of-day clock is used as an interval timer, the interrupt signals that the prewired time interval has elapsed.

Table 5-68. Time-of-Day Clock Commands and Function Codes

Time-of-Day Clock Commands	Function Codes								
	Hexadecimal Notation	Binary Notation							
		Bit Position							
DOT		8	9	10	11	12	13	14	15
Hold Clock	B 0	1	0	1	1	0	0	0	0
Write Word 1*	B 1	1	0	1	1	0	0	0	1
Write Word 2*	B 2	1	0	1	1	0	0	1	0
Stock Clock	B 4	1	0	1	1	0	1	0	0
Reset Clock	B 5	1	0	1	1	0	1	0	1
Start Clock	B 6	1	0	1	1	0	1	1	0
DIN									
Read Word 1	B 1	1	0	1	1	0	0	0	1
Read Word 2	B 2	1	0	1	1	0	0	1	0
Read Word 3**	B 3	1	0	1	1	0	0	1	1

* Optional with time set option (Model No. 77653)

**Optional with extended clock resolution (Model No. 77652)

Table 5-69. Interval Timer Input Jumper Connections

Interval	Jumper to 26X13	
	Signal	Pin
10 usec	C08-	26X04
20 usec	C10-	25X34
100 usec	C80-	25X04
200 usec	.1MS1-	24X34
1 ms	.1MS8-	24X04
2 ms	MS1-	23X34
10 ms	MS8-	23X04
20 ms	10MS1-	22X34
100 ms	10 MS8-	22X04
200 ms	HDMS1-	21X34
1 sec	HDMS8-	21X04
2 sec	S1-	20X34

Interval	Jumper to 26X13	
	Signal	Pin
10 sec	S8-	20X04
20 sec	10S1-	16X04
1 min	10S4-	16X15
2 min	M1-	19X34
10 min	M8-	19X04
20 min	10M1-	16X32
1 hr	10M4-	15X12
2 hr	H1-	15X15
10 hr	H8-	14X12
20 hr	10H1-	14X15
1 day	H23-	29X28

**5-18.8 Time-of-Day Clock Operation
(Model Nos. 77651, 77652, 77653)**

All clock operating controls and indicators are located on the front panel of the clock and are illustrated in Figure 5-34 and described in Table 5-70.

5-18.8.1 Power ON and OFF Procedure

Application of power to the clock is remotely controlled by the computer. No provision is made for turning on and off at the clock.

5-18.8.2 Manual START, STOP, and RESET

The FUNCTION switches are used to start, stop, and reset the clock manually. The RUN and HALT switches are pressed to start and stop the clock, and the RESET switch is pressed to clear the clock counter to 0 and to enable setting the time.

The RESET switch does not stop the clock. If the clock is not stopped when the RESET switch is pressed, the counter will be cleared to 0 when the switch is pressed, but will continue to run when the switch is released.

5-18.8.3 Manual Time Set

The procedure for manually setting the clock is as follows:

- a. Press HALT switch to stop clock.
- b. Press RESET switch to clear clock counter and to enable manually setting the count in the counter stages.
- c. Press least-significant TIME switch-indicators (engraved "1") to increment the counter stages. The count displayed in a column of indicators will be incremented by one each time the switch-indicator at the bottom of the column is pressed and released. No carries are propagated between stages when the clock has been reset, and the stages may be set in any order.
- d. If a mistake is made in setting a stage, the count may be incremented past zero to the correct value. The carries are suppressed, and the next-higher stage will not be incremented when the count passes zero.

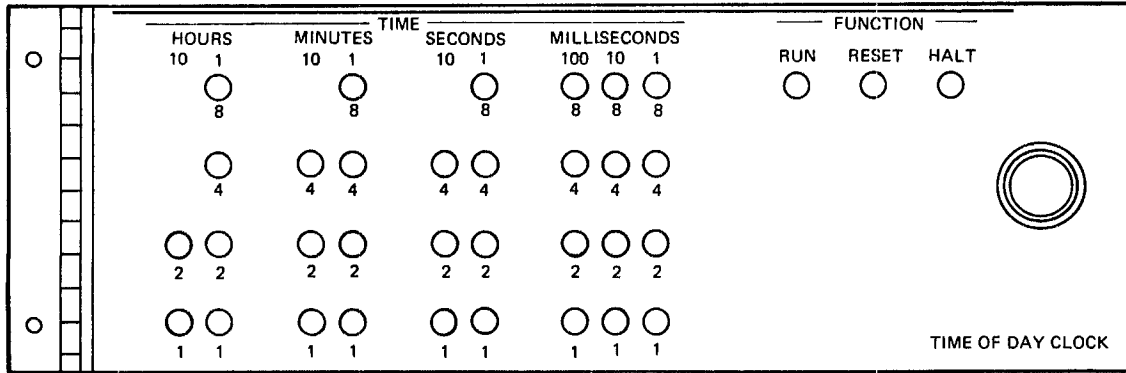


Figure 5-34. Time-of-Day Clock Control Panel

Table 5-70. Time-of-Day Clock Controls and Indicators

Control/Indicator	Function
TIME Indicators (2, 4, 8)	Display contents of clock counter stages except LSB.
TIME Switch-indicators (1)	Display LSB of clock counter stages. Permit manual entry of count into stages when clock is stopped and reset.
FUNCTION	
RUN Switch-indicator	Starts clock. Indicates clock is running.
RESET Switch-indicator	Resets clock counter and divider and hold logic. Indicates clock is stopped and reset.
HALT Switch-indicator	Stops clock. Indicates clock is stopped.

Section 6

OPTIONS

6-1 GENERAL

This section describes options that can be installed in the 706. Some of these options have been described in the three preceding sections and have been noted as options. Peripheral options are described in the sales literature for the 706.

6-2. DIRECT MEMORY ACCESS

A Direct Memory Access channel (DMA) can be provided in the 706. The DMA allows up to six peripheral devices to communicate directly with the 706 memory. A detailed explanation of the operation of the DMA channel is given in Section 4-3.

6-3. INTERRUPT EXPANSION

The basic 706 computer is equipped with one interrupt level. Interrupt options include expanding to four, eight, twelve, or sixteen levels of priority interrupt. A detailed explanation of the operation of the interrupts is given in Section 3.

6-4 HARDWARE MULTIPLY/DIVIDE

Hardware multiply/divide is an option that performs a 16-bit 2's complement multiply or divide. A multiply operation produces a 31-bit algebraic product. A divide operation produces a 16-bit algebraic quotient and a 16-bit remainder.

The multiply/divide instruction format requires two consecutive memory locations. The first location defines the instruction, multiply or divide, and the second location defines a 15-bit address which contains the multiplicand for a multiply or the divisor for a divide. The 15-bit address allows either instruction to reference any location in memory without using the index register.

The index register is not available for address modification during the execution of multiply and divide instructions. It is utilized with the accumulator (ACR) to form the double precision register required by the operation. See Section 2-1.10. for a detailed description of the MPY/DIV instructions.

The execution time of an MPY instruction is 7-10 cycles depending upon the number of "ones" in the multiplier. The execution time of a DIV instruction is 10 or 11 cycles depending upon the signs of the divisor and dividend.

If the hardware multiply/divide option is not included in the 706, software must be used to perform multiply/divide operations. The single precision multiply and divide routines in the 706 Math Library (see Appendix J) performs effective multiplication and division in 70 and 155 cycles (average) respectively.

6-5. MEMORY PARITY CHECK

The memory parity check option provides error detection in data transfers to and from memory. A parity bit is generated for each byte stored into memory. On reading out of memory, each byte is checked for odd parity. An eighteen bit memory is provided with the memory parity option. The extra two bits are used for storing the parity bits for each word. If the DMA option is provided, parity is generated for all bytes stored by DMA, and odd parity is checked for all bytes read out of memory by DMA.

If a parity error is detected, an error interrupt is generated and the CPU Parity Error Flag (MMFPCER) will be set for a CPU memory fetch or the DMA Parity Error Flag (DMFPCER) will be set for a DMA memory fetch.

If a parity error is detected during the processing of an interrupt (while reading the link address) the 706 will halt.

6-6 POWER FAIL SAFE

6-6.1 Purpose

The power fail safe (PFS) option provides an orderly power-down sequence for the 706 when primary AC power fails and an orderly power-up sequence when AC power is restored.

The PFS ensures that no memory data is lost from power failure and permits the CPU to restart operation at the point where power was lost.

6-6.2 Functional Characteristics

Data in memory are not destroyed if the CPU is not executing an instruction at the time DC power goes down. By sensing the loss of AC power, the power fail safe feature initiates a signal which forces Unmask Interrupt and interrupts the CPU. The CPU may then execute an interrupt routine to terminate instruction execution before the DC power is actually degraded to an inoperable level. Unless a valid DC power level is available a reset signal is generated by PFS which disables logic circuits in the CPU and peripheral equipment. On a power-up sequence the reset signal is released when DC power reaches the valid level. PFS then generates a restart and the CPU resumes program execution.

Interrupt Execution Time: The CPU has 1.0 millisecond to execute a subroutine which stores all pertinent parameters and halts.

Adjustable Sense Point: The point at which power failure is detected is adjustable from 100 to 115 VAC. Minimum recommended setting is 107 VAC. Lower settings will reduce the time available to execute the interrupt routine.

Restart: The restart signal occurs after release of the reset signal and consequently, after power is stabilized at the operational level.

6.6.3 Programming

The power-down interrupt routine should mask all interrupts, save the index register and accumulator, and store a JMP instruction in location zero. Upon restarting, the JMP instruction should transfer to a power-up restoring routine. If restart is not desired, the power-down interrupt routine must store a HLT instruction in location zero.

The PFS restart signal causes the CPU to automatically begin execution on any power-up sequence. The CPU is in the reset state when restart occurs. Execution starts from memory location zero with all registers cleared, all interrupts disabled, overflow and compare flip-flops reset, and the local/global flip-flop in the local state.

Execution of the power-down interrupt routine must be completed within one millisecond to guarantee that the CPU is halted before expiration of Interrupt Execution Time.

6-6.4 Operation Note

The PFS feature may be used either for its intended power protect function or for an automatic restart function. A POWER-OFF OVERRIDE switch is located on the AC Controller which, when in the override position, causes the PFS to ignore power-down by the CPU power switch. If not in the override position, PFS will react to any power-down, including the CPU power switch. For automatic restart the JUMP instruction stored in memory location zero (by the PFS interrupt routine) must transfer execution to the beginning of the main program. Under these conditions the program will repeat from the beginning when power is restored from any power-down situation. When programming, PFS attention must be paid to the effects of data changed by any previous processing in the event of a power failure.

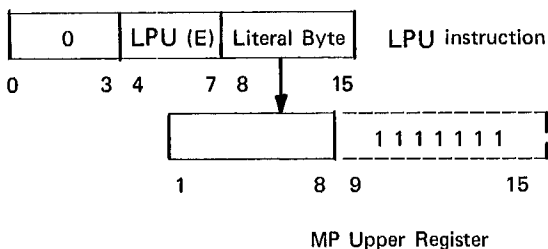
6-7 MEMORY PROTECT

6-7.1 General

The memory protect option allows operation of the background/foreground monitor system to be used in applications requiring real-time responses to interrupts while concurrently performing batch background processing.

When the foreground, a real-time program, is in operation the 706 operates in the EXECUTIVE mode. This means that there are no restrictions regarding the use of memory or the use of certain privileged instructions. When the foreground program is not servicing an interrupt the background program can become operational by the execution of the Select User State (SUS), Load Memory Protect Upper (LPU) and the Load Memory Protect Lower (LPL) commands.

The background program operates in one of three user states. In any of these user states, if an attempt is made to fetch an instruction from or store data into a protected area of memory, an interrupt is generated and the fetch or store operation is inhibited. In addition, execution of a privileged instruction generates an interrupt and the privileged instruction is treated as a No Operation. The privileged instructions involve input/output, interrupt, halt, and user state types of commands.

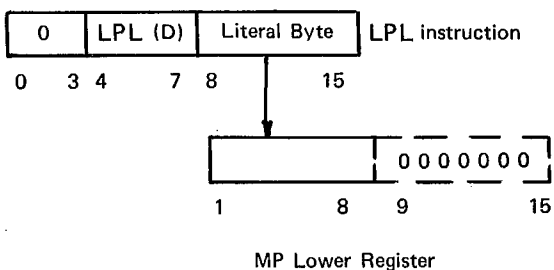


The Memory Protect Lower and Upper Registers each form the most significant 8 bits (1-8) of the lower and upper bounds of memory that is *not* protected inclusively. The least significant 7 bits (9-15) of the lower address are always considered to be zero. The least significant 7 bits (9-15) of the upper address are always considered to be ones. For example: executing a (LPU X '23') followed by a (LPL X 'OE') would allow program execution and data storage from 0700₁₆ to 11FF₁₆ inclusive. An instruction fetch or a data store into any other memory location causes an interrupt and exit from the user state.

6-7.2 Setup and Entry to the User State

There are three instructions that allow the user to initialize the user state or background mode of operation: Load Memory Protect Upper (LPU), Load Memory Protect Lower (LPL) and Select User State (SUS). Figure 6-1 shows the effect of executing the three memory protect instructions.

Executing the Load Memory Protect Lower (LPL) instruction puts the contents of bits 8-15 of the instruction word into the Memory Protect Lower Register. Executing the Load Memory Protect Upper (LPU) instruction puts the contents of bits 8-15 of the instruction word into the Memory Protect Upper Register.



To force the program into one of the three user states a Select User State (SUS) instruction must be executed with bits 14 and 15 of the instruction defining the user state accordingly:

Bit 14	Bit 15	User State
0	0	Executive Mode (if in Executive Mode already do nothing)
0	1	User State 1
1	0	User State 2
1	1	User State 3

The SUS instruction, by itself, does not force the 706 into the User State. SUS only preconditions the user state circuitry such that when a JMP or JSX instruction is subsequently executed the user state is then entered. (INR restores all user flip-flops but does not transfer the User Pre-State Flags to the User State Flags.)

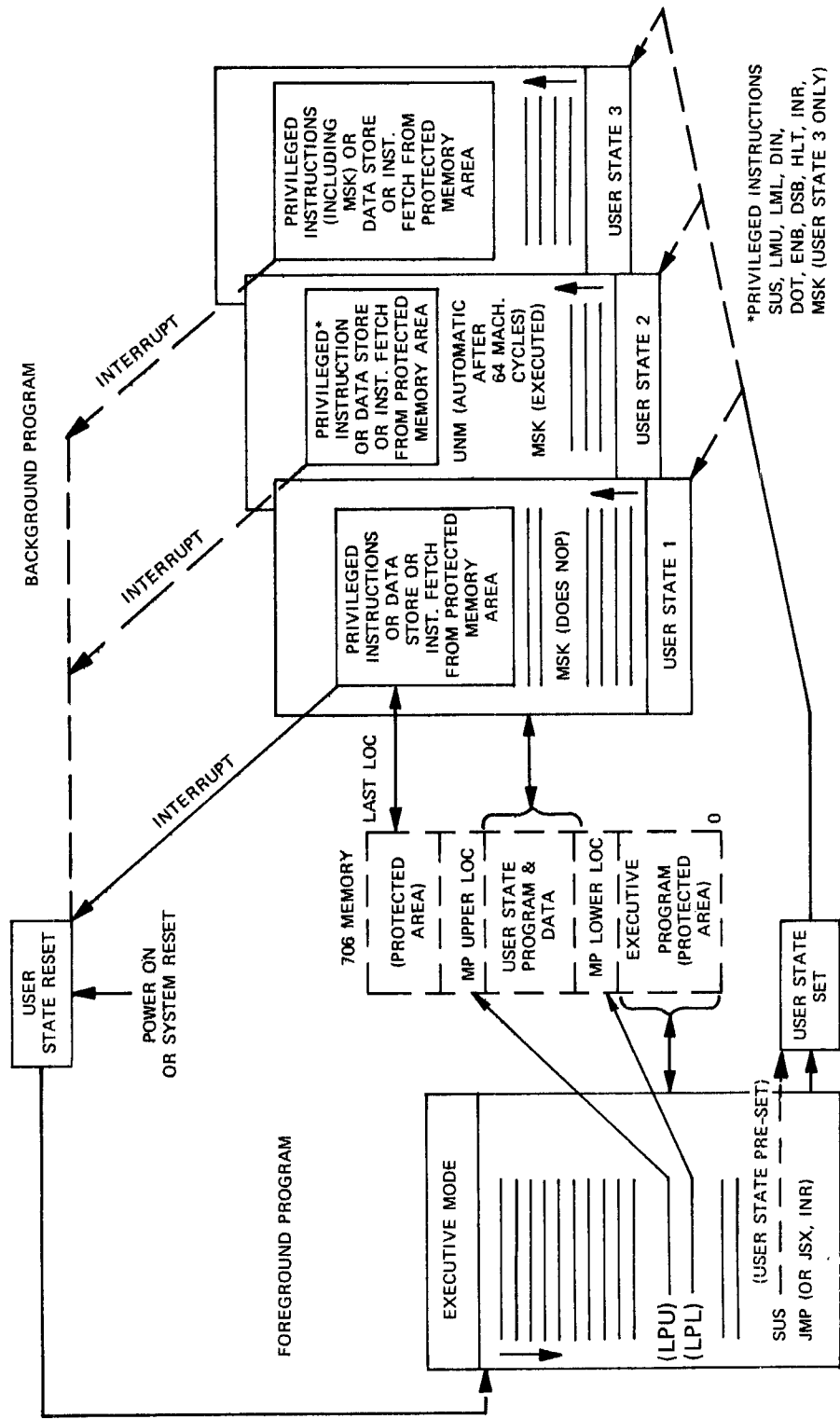


FIGURE 6-1 MEMORY PROTECT OPTION

6-7.3 User States

All three User States have the following set of privileged instructions:

SUS	Select User State (See Section 2-1.4)
LPU	Load Memory Protect Upper (See Section 2-1.8)
LPL	Load Memory Protect Lower (See Section 2-1.8)
DIN	Direct Input
DOT	Direct Output
ENB	Enable Interrupt
DSB	Disable Interrupt
HLT	Halt
INR	Interrupt Return

If an attempt is made to execute any of these instructions in the User mode, an interrupt will be generated, the instruction will not be executed, and the user program will be terminated. If the interrupt level for the privileged instructions is disabled at the time of generating the interrupt, the 706 processor will halt. The preceding discussion also applies to either trying to do an instruction fetch or data store outside the bounds of the memory protect registers. One interrupt serves both the memory protect violation and the privileged instruction violation.

The User states differ in the treatment of the MSK instruction as follows:

User State	MSK Instruction
1	Treated as a NO-OP
2	Interrupts are masked and 64 machine cycles later they are automatically unmasked, or interrupts are immediately unmasked if a privileged instruction is executed
3	Privileged instruction. Causes interrupt and instruction is not executed.

6-7.4 Leaving the User State

There are three reasons for leaving the User State. These reasons along with the method of handling the User state exit are shown in table 6-1.

6-8 MEMORY EXPANSION

The memory can be expanded in 4096, 8192, or 16,384 word modules to a maximum capacity of 32,768 words. These modules can be intermixed. No memory matching with CPU modification is required for expansion of the memory system. The maximum memory size is 32,768 words. See Section 1-3 for addressing schemes.

Table 6-1. Exits From User State

Reason for Leaving User State	Method
1. Normal Return - Processing completed or the user desires to actually execute a privileged instruction in the user state.	1. Store a "key" or "identifier" word in the cell following the privileged instruction. Upon processing the error interrupt examine the identifier word for the reason the user state was left.
2. Error Return - Unintentional attempted execution of a privileged instruction or an instruction fetch/data store to protected memory.	2. Upon processing the error interrupt, examine the identifier word. It should be an illegal key word.
3. Foreground Peripheral Interrupt - A foreground device interrupts the 706 while in the User state.	3. Process the interrupt and return to the User state by issuing INR command.

Section 7

OPERATION

7-1 POWER ON AND OFF PROCEDURE

Computer power is turned on and off by pressing the power ON and OFF switches. No special procedures or sequences are involved, but the following precaution should be observed:

The computer should be halted before turning power off. Turning off power while the computer is running may introduce errors into the program, the memory, and data transmitted over the DIO and DMA buses unless the Power Fail Safe option is installed.

7-2 MANUAL DATA DISPLAY AND ENTRY

The controls and indicators on the CPU control panel permit the contents of CPU registers, machine status, the contents of memory locations to be displayed. They also permit entry of data into four CPU registers and into memory locations.

The indicators are lit when the corresponding register flip-flop contains a 1 (set) and are not lit when the flip-flop contains a 0 (reset). The switches may be used to enter data into the program counter, memory buffer, index and accumulator registers and to enter data from the memory buffer register into memory or read data from memory into the memory buffer register. The

CLEAR switches reset all flip flops in the register, and the numbered switches (1-15 or 0-15) set the corresponding flip-flops directly when pressed. No provision is made for resetting the individual flip-flops.

The entire register must be cleared and the entire word entered if 0's are to be entered into any positions already containing 1's.

The contents of the program counter register and the selected register (or machine status) are displayed continuously whether the computer is running or halted. However, the computer must be halted when data are entered into registers or the contents of a memory location are displayed or entered.

7-2.1 Program Counter Register

The Program Counter indicators display the contents of the program counter register at all times. The PROGRAM COUNTER switches are not capable of clearing or entering data into the register when the computer is running. When the computer is halted the Program Counter is always displayed. The procedure for entering data is as follows:

- a. Press HALT switch
- b. Press CLEAR switch.
- c. Press appropriate switch-indicators

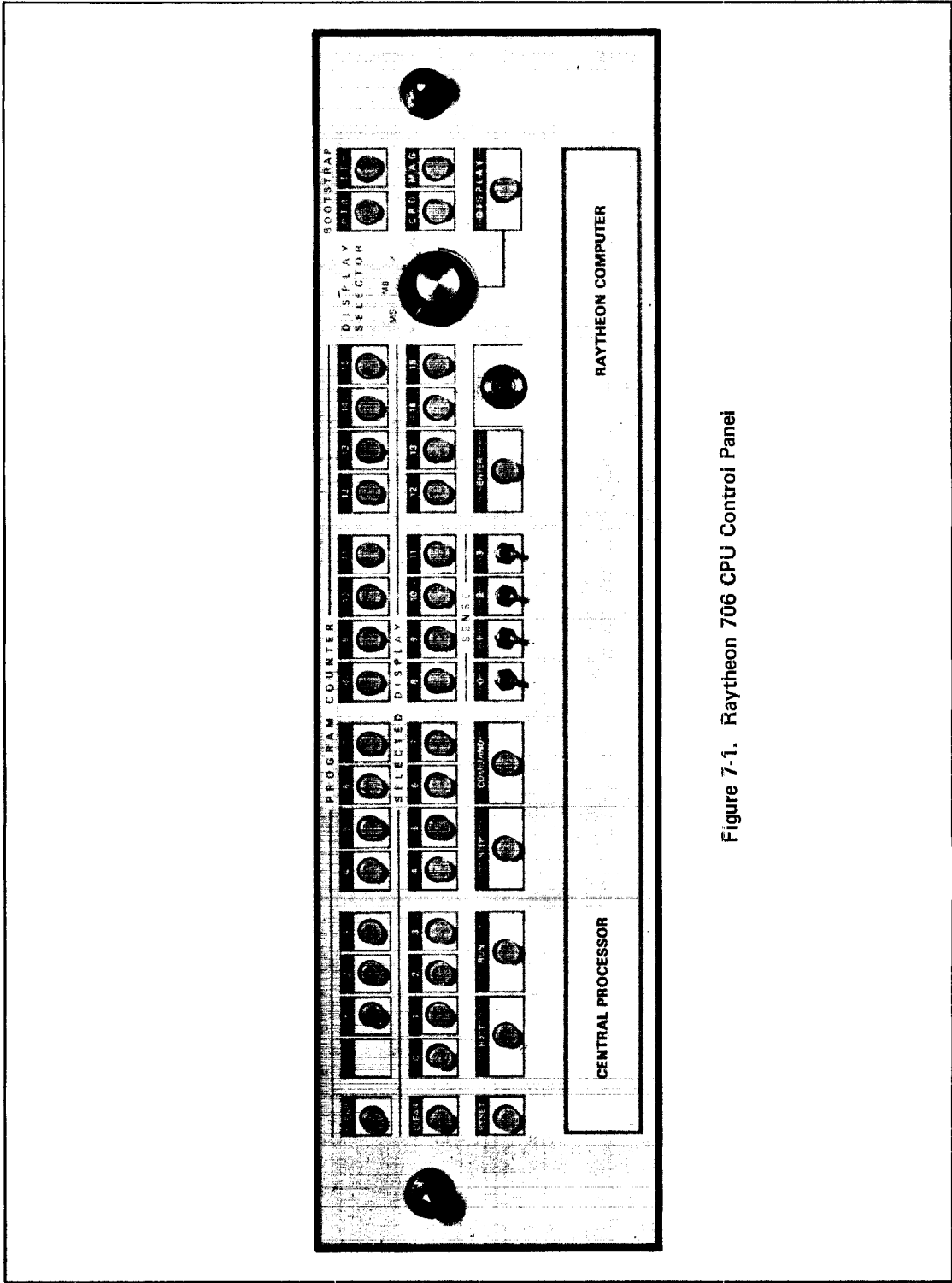


Figure 7-1. Raytheon 706 CPU Control Panel

7-2.2 Selected Display

The SELECTED DISPLAY indicators display a machine status word (MS) or the contents of a selected register, depending on the setting of the DISPLAY SELECTOR switch. The registers available are the accumulator (AC), index register (IX), memory buffer register (MB) and instruction register (IN). The machine status word or contents of the selected register are displayed continuously when the DISPLAY SELECTOR switch is set to the corresponding position. No other action is required to initiate the display.

The machine status word contains the contents of the extension register (bits 00-04), adder negative flip-flop ADFNEG (bit 05), adder equal flip-flop ADFEQL (bit 06), adder overflow flip-flop ADFOVF (bit 07), global control flip-flop CCFGLB (bit 08), user state flip-flops CCFUSRO-CCFUSR3 (bits 9-12) CPU parity error flip-flop MMFPCER (bit 13), the DMA parity error flip-flop DMFPCER (bit 14) and the privileged instruction flip-flop MPFPRV (bit 15).

The SELECTED DISPLAY switches are enabled only in the MB, IX and AC positions of the DISPLAY SELECTOR switch. They are capable of clearing or entering data into the memory buffer, index, and accumulator registers at all times that the selector switch is set to these positions, but must be used only when the computer is halted. The procedure for entering data into these registers is as follows:

- a. Set DISPLAY SELECTOR switch to position corresponding to register into which data is to be entered.
- b. Press HALT switch.
- c. Press CLEAR switch.
- d. Press appropriate switch-indicators.

7-2.3 Memory Display and Entry

The DISPLAY switch is used to read a word from the memory location specified by the program counter register and to load it into the memory buffer register. The word is displayed by the SELECTED DISPLAY indicators. The ENTER switch is used to write a word from the memory buffer register into the memory location specified by the program counter register. Either switch increments the program counter by one each time it is pressed. The DISPLAY SELECTOR switch must be set to MB and the computer must be halted during memory display or entry.

Displaying Memory

The procedure for displaying a memory location or a series of locations is as follows:

- a. Press HALT switch.
- b. Set DISPLAY SELECTOR switch to MB.
- c. Enter address of location or first location of a series to be displayed into program counter register.
- d. Press DISPLAY switch. The contents of the memory location will be loaded into the memory buffer register and displayed by the SELECTED DISPLAY indicators. The program counter will be incremented to the next location in sequence.
- e. If a series of words are to be displayed, press DISPLAY switch once for each location.

Entering Memory

The procedure for entering a memory location or series of locations is as follows:

- a. Press HALT switch.
- b. Set DISPLAY SELECTOR switch to MB.
- c. Enter address of location to be entered into program counter register.
- d. Enter word to be entered into Selected Display indicators.
- e. Press ENTER switch. The word in the memory buffer register will be entered into memory location, and the program counter register will be incremented to the next location in sequence.
- f. If a series of words are to be entered, enter next word into memory buffer register and press ENTER switch. Repeat until all words are entered.

7-3 PROGRAM LOADING

Programs may be loaded into the 706 Computer manually by means of the control panel, from paper tape by means of a manually-entered bootstrap program, directly from magnetic tape or disc storage via the direct memory access bus or from a number of sources under control of the X-RAY executive program.

Absolute or relocatable object programs may be loaded under control of the X-RAY executive program. This program is covered in Section 8 and Section 9. Absolute programs may also be loaded manually, by paper tape bootstrap, or via the direct memory access bus.

7-3.1 Manual Program Entry

The CPU front panel controls are used to manually load programs into memory. These controls are generally used to load brief programs such as the bootstrap programs.

The procedure for manually loading a program is as follows:

- a. Press computer RESET switch.
- b. Set program counter register to starting location of program.
- c. Set DISPLAY SELECTOR switch to MB position.
- d. Enter first word of program into memory buffer register.
- e. Press ENTER switch. The word will be entered into the location contained in the program counter register and the program counter will be incremented by one.
- f. Clear memory buffer register, enter next word into register, and press ENTER switch. Repeat until entire program is loaded.

- g. After the program is loaded, correct loading can be verified by setting the program counter register to the starting location, pressing the DISPLAY switch, and comparing the displayed contents of the location with the program listing. The program counter is incremented to the next location each time the DISPLAY switch is pressed, and succeeding locations are displayed each time the switch is pressed.

7-4 BOOTSTRAP PROGRAM LOADING

A set of four switch-indicators are used in the 706 to select one of the following four I/O devices and its associated hardware bootstrap mode.

TTY Paper Tape Reader (TTY Mode)

High Speed Paper Tape Reader (PTR Mode)

DIO Mag Tape (MT Mode)

Card Reader (CR Mode)

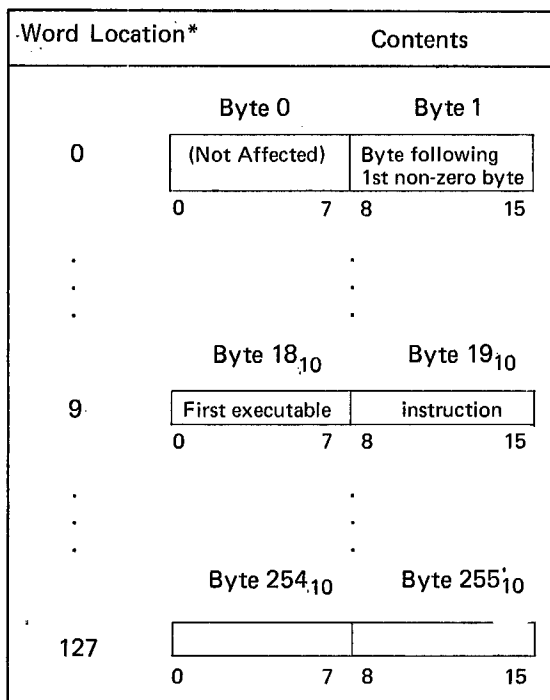
7-4.1 Bootstrap Loading Procedure

The system RESET switch must be pressed prior to pressing the BOOTSTRAP switch. When a BOOTSTRAP switch is pressed, the CPU is placed in the bootstrap mode and its associated indicator will light. Only one of the four switch-indicators can be pressed at a time, and only one of the four bootstrap modes can be on at a time. Once set to the bootstrap mode, the CPU will ignore the action of any of the four bootstrap switches until it returns to the Halt Mode and the RESET switch depressed again.

The CPU leaves the Halt Mode and enters the Run Mode simultaneously with the setting of Bootstrap Mode. During the instruction fetch cycle, the memory buffer register (MBR) is inhibited by the Bootstrap Mode from getting data from the memory; instead, it receives data from the hardware bootstrap logic.

In the hardware Bootstrap Mode or Initial Program Load Mode, the following sequence of instructions is decoded and executed:

PCR DECODE	BOOTSTRAP PROGRAM	COMMENT
0	DOT X, 9	Select Device/Initiate Reading
1	DIN X, 0	Obtain Status
2	SLL 7	Move "Data Ready!" Bit to Sign Bit
3	SAM	Is Device Ready?
4	JMP 1	No, Wait
5	DIN X, D	Yes, Get Data
6	SAZ/STB *0	Data Zero Test/Store Data Byte
7	IXS 1	Data Not Zero
8	JMP 1	Data Zero Return
9	JMP 1	



*Assumes IXR initially 0.

Figure 7-2. Bootstrap Memory Map

The data, decoded from PCR by the hardware bootstrap logic, is gated into MBR during the instruction fetch phase and executed the same as instructions coming from memory.

When PCR counts up to 6, the bootstrap instruction is decoded as SAZ until the *second* non-zero byte has been read. After reading the first non-zero byte an SAZ instruction is executed, and the index is incremented, but the first byte is not stored. All bytes read, after the first byte is read, are stored according to the contents of IXR.

If Sense Switch 3 is off, bytes are stored until the IXR has been incremented to 100₁₆. Program execution then begins at word location 9. If Sense Switch 3 is on, bytes are read and stored until either the HALT switch has been depressed or no more data is available from the peripheral device.

Figure 7-2 depicts the results of bootstrapping 256 bytes (128 words) from an I/O device with the IXR initially set to zero and Sense Switch 3 off.

7-4.2 Bootstrap Device Operation

The four I/O devices associated with hardware bootstrap logic have been assigned the following device codes:

	Bit 8	Bit 9	Bit 10	Bit 11
TTY Paper Tape Reader	1	1	1	0
High Speed Paper Tape Reader	1	1	0	1
DIO Mag Tape	1	0	0	1
Card Reader	1	0	0	0

7-4.3 Teletype Bootstrap Operation

Only the TTY paper tape reader is involved in the bootstrap operation, not the TTY keyboard. When the TTY paper tape reader is selected, DOT E, 9 does not initiate reading, it only lights up the PTR mode indicator on the TTY set. The TTY paper tape reader must be started manually. One byte is transferred at a time, strobed into ACR 08 through 15 first, then stored into memory.

7-4.4 High-Speed Paper Tape Bootstrap Operation

If the high speed paper tape reader is selected, the instruction DOT D,9 initiates reading. One byte is transferred at a time, strobed into ACR 08 through 15 first, then stored into memory.

7-4.5 DIO Magnetic Tape Bootstrap Operation

If the DIO 9-Track Magnetic Tape is selected, the instruction DOT 9, 9 initiates the reading of one record. Two 9-track characters are assembled into a 16-bit word. One 9-track word is transferred at a time and strobed into ACRO-15, then bits 8-15 of ACR are stored. Bits 0-7 are lost in the bootstrap operation only, due to the STB instruction.

Since the 7-track unit transfers zeros for bits 12-15, it is impractical to bootstrap a program directly from this unit.

If the record being bootstrapped is less than 256 words long, the computer will hang-up in the status loop waiting for bit 7 of the status word to go true.

7-4.6 Card Reader Bootstrap Operation

When the card reader is selected, DOT 8, 9 initiates reading. A 12-bit word is strobed into ACRO4 through 15, after which the least significant byte will be loaded into the effective memory byte location. Data stored in ACRO4 through 07 is lost and should not contain any useful information. Bit 4 corresponds to Row 12 and Bit 15 corresponds to Row 9 of the IBM card. Because one select instruction to the card reader causes cards to read continuously, the bootstrap program must be contained on 4 cards: 80 columns each of the first three cards and the first 16 columns of the fourth card.

7-5 DIRECT PROGRAM LOADING

Programs may be loaded directly into memory from DMA magnetic tape or disc storage via the direct memory access bus. Program loading is initiated by pressing the controller LOAD switch.

The program is loaded automatically and, except for use of the DMA logic, is completely independent of the central processor unit. No commands or status are exchanged on the data input-output bus and no interrupts are involved.

Operating instructions for the device controllers are contained in the manuals covering the controllers and Section 5 of this manual.

7-6 PROGRAM EXECUTION

Programs may be executed in three modes: run, single-command, and single-step. The program may be modified by the use of the four SENSE switches on the CPU control panel and an external sense line (EXSENS-) on the data input-output bus. The use of the SENSE switches and the external sense line is a function of the individual program and is covered in the program descriptions. The operating modes are discussed in the paragraphs that follow.

7-6.1 Run Mode

In the run mode, the program is executed continuously until a program halt occurs or the HALT switch is pressed. Execution of the program may be interrupted by devices on the data input-output bus, but will be resumed when the interrupt is serviced. The initial conditions that must be set up before the program is executed are given in the program description. The procedure for executing a program in the run mode is as follows:

- a. After program is loaded, press computer HALT and RESET switches.
- b. Enter data into index and accumulator registers if called for in program description.
- c. Enter data into memory locations if called for in program description.
- d. Enter address of first program instruction into program counter register.

- e. Set SENSE switches as specified in program description.
- f. Prepare peripheral equipment as specified in program description.
- g. Press computer RUN switch.

Each instruction requires the switch to be pressed once for each number of phases required to execute the instruction. The first instruction requires that the switch be pressed an extra time to enter the fetch phase.

The single-step mode can only be terminated by pressing the computer HALT switch (or by a system reset). This resets the single-step mode flip-flop and allows instructions to be executed in the single-command or run mode. Of the computer is not in the halt state when the HALT switch is pressed, the remaining phases of instruction being executed are completed without stopping, and the computer returns to the halt state.

7-6.2 Single-Command Mode

In the single-command mode, one complete program instruction is executed each time that the SINGLE COMMAND switch is pressed, regardless of the number of phases (machine cycles) required to execute the instruction. This mode is normally used for "debugging" programs and as a maintenance aid when it is desired to observe the execution of complete instructions.

The computer enters the single-command mode from the halt state each time the SINGLE COMMAND switch is pressed and returns to the halt state after executing each instruction. Program execution may be resumed by pressing the computer RUN switch.

7-6.3 Single-Step Mode

In the single-step mode, one instruction phase is executed each time the SINGLE STEP switch is pressed. This mode is normally used as a maintenance aid when it is desired to observe the progression from phase to phase of instructions and the execution of each phase.

The single-step mode must be entered from the halt state. The first time the SINGLE STEP switch is pressed, a mode flip-flop (CCFSTP) is set, and the computer advances from the halt state to the instruction fetch phase. If an interrupt is in the WAIT state, the instruction fetched is the first one of the interrupt service routine. The next operation of the switch causes the fetch phase of the instruction to be executed; after which the computer advances to the next phase, but does not execute the next phase until the switch is pressed again.

7-7 CONSOLE LOCKOUT SWITCH

By inserting the key and turning the CONSOLE LOCKOUT switch to the right, all console functions are protected against intrusion except the SENSE switches and the POWER ON-OFF switch.

Section 8

PROGRAMMING SYSTEMS DESCRIPTION

8-1 INTRODUCTION

A full range of Raytheon Computer developed software is offered to 706 users. Four major configurations of peripheral equipment can be assembled to support five different operating systems.

Two assemblers can be used with the 706: SYM I, a one-pass conversational assembler for use with 4K systems and SYM II, a fast and powerful two-pass assembler with procedural capabilities for 8K and larger systems.

Both Conversational FORTRAN and Real-Time FORTRAN IV provide high-level language capability for the 706. Conversational FORTRAN may be used with only 4K of core memory while Real-Time FORTRAN IV provides an extended superset capability of ASA FORTRAN.

Relocatable and absolute loaders are provided for all 706 software systems. The absolute loaders input core image programs directly into memory. Relocatable programs are input in relocatable loader text format.

There are five separate operating systems associated with the 706:

- *BASIC System (manual)*
- *STANDARD Operating System (High-Speed Paper Tape)*
- Magnetic Tape Operating System
- Disc Operating System (Real-Time)
- *Multiprogramming System (MPS)*

Each operating system can be implemented using a 706 CPU and a minimum set of options and peripheral equipment.

Each 706 operating system has a resident monitor and an X-RAY Executive tailored to the hardware configuration. Included in all resident monitors is a file-oriented, device independent input/output system that allows the computer to be time-shared in a real-time interrupt environment.

The 706 comes complete with a comprehensive set of utility programs, including editors, system generators, math library, and trace/debug routine.

8-2 HARDWARE/SOFTWARE CONFIGURATIONS

All Raytheon 706 hardware and software falls into one or more of four categories:

- *BASIC*
- *STANDARD*
- *EXTENDED*
- *ADVANCED*

These categories imply certain minimal hardware configurations. Section 1-6.1 of this manual briefly explains these categories by showing block diagrams of each configuration (Figures 1-28, 1-29, 1-30, 1-31).

8-2.1 Hardware System Configurations

Table 8-1 is a hardware system configurator. It shows all standard product devices arranged in two sections: *Minimum Requirements* and *Other Peripheral Equipment and Options*. Vertical columns indicate the hardware systems, showing which devices *can* be included and which devices are *software supported*. Physically, there is no reason why any piece of peripheral equipment or option cannot be attached to a BASIC 706 configuration. But system software support is predicated on

Table 8-1. 706 Hardware System Configurator

HARDWARE \ SYSTEM	NON MASS STORAGE		MASS STORAGE	
	BASIC	STANDARD	EXTENDED	ADVANCED
MINIMUM REQUIREMENTS				
706 CPU (INCL 1-LEVEL AUTOMATIC PRIORITY INTERRUPT)	●	●	●	●
ASR 33 TELETYPE	●	●	●	●
4K CORE MEMORY	●	●	●	●
8K CORE MEMORY		●	●	●
16K CORE MEMORY		●	●	●
HIGH SPEED PAPER TAPE READER OR CARD READER		●	●	●
DISC OR MAGNETIC TAPE		●	●	●
DIRECT MEMORY ACCESS		●	●	●
4 LEVELS AUTO PRIORITY INTERRUPT		●	●	●
MEMORY PROTECT, OPERATER INTERRUPT & TIME OF DAY CLOCK		●	●	●
OTHER PERIPHERAL EQUIPMENT AND OPTIONS				
ASR 35 TELETYPE	●	●	●	●
4K MEMORY EXPANSION	●	●	●	●
8K MEMORY EXPANSION	●	●	●	●
16K MEMORY EXPANSION	●	●	●	●
HIGH SPEED MULTIPLY/DIVIDE	●	●	●	●
POWER FAIL SAFE	●	●	●	●
4-LEVEL P.I. EXPANSION	●	●	●	●
8-LEVEL P.I. EXPANSION	●	●	●	●
12 LEVEL P.I. EXPANSION	●	●	●	●
16-LEVEL P.I. EXPANSION	●	●	●	●
MEMORY PARITY	●	●	●	●
MEMORY PROTECT	●	●	●	●
DIRECT MEMORY ACCESS	●	●	●	●
HIGH SPEED PAPER TAPE READER	●	●	●	●
CARD READER	●	●	●	●
DISC	●	●	●	●
CARD PUNCH	●	●	●	●
HIGH SPEED PAPER TAPE PUNCH	●	●	●	●
LINE PRINTER	●	●	●	●
DIGITAL PLOTTER	●	●	●	●
DMA MAGNETIC TAPE	●	●	●	●
DIO MAGNETIC TAPE	●	●	●	●
TELETYPE MULTIPLEXER	●	●	●	●
BUFFERED I/O CHANNELS	●	●	●	●
ANALOG TO DIGITAL CONVERTERS	●	●	●	●
DIGITAL TO ANALOG CONVERTER	●	●	●	●
TIME OF DAY CLOCK	●	●	●	●
OPERATER INTERRUPT	●	●	●	●

- ALLOWABLE AS MINIMUM REQUIREMENT OR OPTION AND SUPPORTED BY RAYTHEON SOFTWARE
- ALLOWABLE AS MINIMUM REQUIREMENT OR OPTION BUT NOT SUPPORTED BY RAYTHEON SOFTWARE

- 1 - IF CARD READER IS USED INSTEAD OF OR IN ADDITION TO HIGH SPEED PAPER TAPE READER, 8-LEVELS OF PRIORITY INTERRUPT REQUIRED
- 2 - MAGNETIC TAPE IS NOT PART OF ADVANCED OPERATING SYSTEM. THREE TAPE UNITS REQUIRED.
- 3 - CARD READER SHOULD BE SUBSTITUTED FOR HIGH SPEED PAPER TAPE READER.
- 4 - DMA NOT REQUIRED WITH DIO MAGNETIC TAPE

Table 8-2. 706 Software System Configurator

SOFTWARE	SYSTEM	NON-MASS STORAGE		MASS STORAGE	
		BASIC	STANDARD	EXTENDED	ADVANCED
ASSEMBLERS AND COMPILERS ¹					
SYM I/PREP ASSEMBLER		•	•	•	•
SYM II ASSEMBLER			•	•	•
CONVERSATIONAL FORTRAN ²		•	•	•	•
REAL-TIME FORTRAN IV			•	•	•
PROGRAM GENERATION					
SYMBOLIC PROGRAM EDITOR		•	•	•	•
TRACE/DEBUG			•	•	•
SYMBOLIC PROGRAM PREPARATION ¹		•	•	•	•
FORTRAN PROGRAM PREPARATION ²		•	•	•	•
MONITORS AND EXECUTIVES					
X-RAY EXEC		•	•	•	•
RESIDENT MONITOR ³		•	•	•	•
PAPER TAPE INPUT/OUTPUT SYSTEM (PTIOS)		•			
INPUT/OUTPUT SOFTWARE (IOS)		•	•	•	•
STANDARD OPERATING SYSTEM (SOS)			•		
REAL-TIME OPERATING SYSTEM (RTOS)				•	•
MAGNETIC TAPE OPERATING SYSTEM (MTOS)			•	•	
MULTI-PROGRAMMING SYSTEM (MPS)					•
LIBRARY PROGRAMS					
BASIC LOADER		•			
STANDARD LOADER			•		
DISC LOADER				•	•
SYSTEM GENERATORS				•	•
SYSTEM EDITOR			•	•	•
CONVERSATIONAL FORTRAN LIBRARY		•	•	•	•
REAL-TIME FORTRAN IV LIBRARY			•	•	•
MATH LIBRARY		•	•	•	•
SORT/MERGE				•	•
DIAGNOSTICS					
SENSOR		•	•	•	•

1 - SYM I ASSEMBLER ALSO INCLUDES PREP PROGRAM.
 2 - FORTRAN PROGRAM PREP IS PART OF CONVERSATIONAL FORTRAN PACKAGE.
 3 - X RAY EXEC AND RESIDENT MONITOR ARE COMBINED FOR BASIC SYSTEM.
 4 - OPERATES WITH STANDARD SOFTWARE

certain minimum core memory capacity and interrupt level requirements. For instance, a disc memory may be attached to a BASIC 706 system (4K core memory, no high-speed input device) but, Raytheon Computer supplied disc software will not operate in this minimum configuration*. In order for Raytheon Computer supplied disc software to operate, an EXTENDED system (high-speed input device, 8K memory, and disc memory) is required.

The Minimum Requirements section of Table 8-1 shows the absolute minimum peripheral equipment and options needed to create any one of the four system configurations. The Peripheral Equipment and Options section indicates additional hardware that can be added to each of the four system configurations.

Also shown are the equipment that must be added to each system configuration to convert to the next higher configuration.

8-2.2 Non-Mass Storage vs. Mass Storage Systems

The BASIC and STANDARD systems are non-mass storage types of configurations. A non-mass storage system implies that there are no mass storage peripheral devices (disc and/or magnetic tape) available for use by the operating system. The EXTENDED and ADVANCED systems are mass storage types of configurations. They not only consist of mass storage peripheral devices (disc and/or magnetic tape) but have the appropriate number of automatic priority interrupt levels and Direct Memory Access channels to function efficiently with the operating system.

The mass storage systems allow the use of the Real-Time Operating System (RTOS), the Magnetic Tape Operating System (MTOS) or the Multiprogramming System (MPS). The non-mass storage systems may use mass-storage devices, but normally these devices are only for dedicated programs that operate outside of the operating system.

8-2.3 Software System Configurations

Raytheon supplied software operates within one or more of the hardware configurations. Table 8-2 shows the software items that may be used with any given hardware system.

As a general rule, software is upward compatible. This means that a program that operates within a BASIC system will operate within the STANDARD, EXTENDED and ADVANCED system also.

8-3 ASSEMBLERS

Assemblers are programs which translate (or "assemble") symbolically coded programs into numeric codes used by the computer. Assemblers relieve the programmer of many of the clerical tasks associated with the preparation of a program. Assembly languages permit programs to be coded using symbolic instructions and labels instead of numeric machine codes. For example, the symbolic instruction to jump to location START

```
JMP    START
```

is more easily remembered and more meaningful to the programmer than '1965' which is the numerical machine equivalent. Symbolic code makes programs easier to debug because the assembly program assigns the proper numeric values to the symbol making clerical errors less likely.

Figure 8-1 illustrates the assembly process. A symbolically coded program which is referred to as the *source program* is input from the PRincipal INput (PRIN) device. The PRIN device is usually a card reader or paper tape reader. If the source code has been first stored in disc or magnetic tape, these units may serve as the PRIN device. The conversational assembler, SYM 1, allows source input from the ASR-33 keyboard.

The assembler (that has been previously loaded) accepts the source statements and translates them

*The term "software supported" does not apply to hardware diagnostic programs.

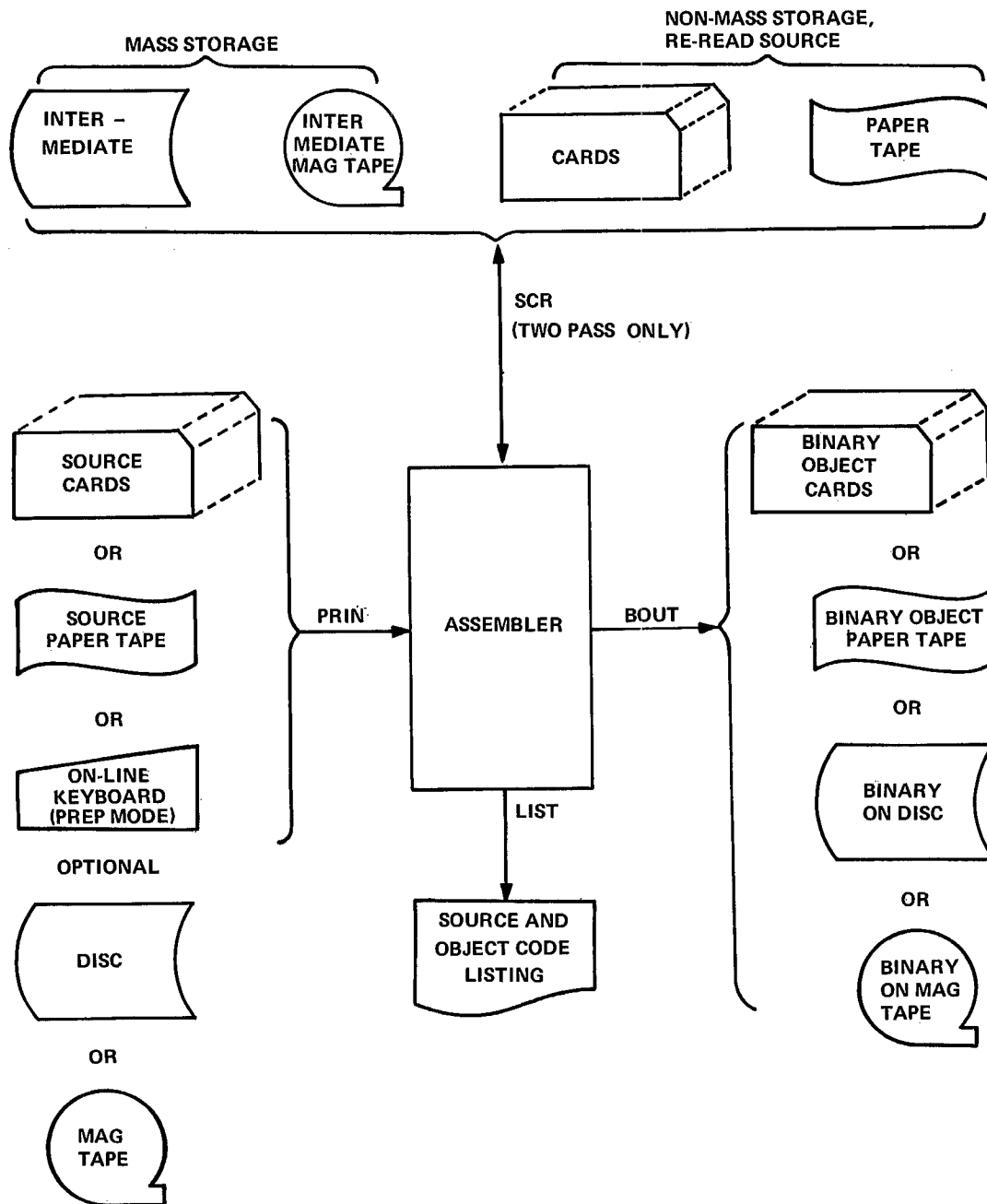


Figure 8-1. The Assembly Process

into either relocatable or absolute loader text (see Section 8-5). The loader text is commonly referred to as the *binary program* or *object program*. The binary program is output on the Binary OUTput (BOUT) device. The BOUT device can be a card punch, paper tape punch, disc, or magnetic tape. A card or paper tape binary program is normally removed from the device for loading and execution at some later time. Programs assembled onto the disc or magnetic tape are normally loaded and executed immediately without the handling problem involved with paper tape or cards (Go Mode).

The LISTing (LIST) device is normally the line printer or ASR-33 keyboard. A typical program listing duplicates images of the source statements and adds the assembled hexadecimal equivalents and their relative or absolute locations. The listing provides a *hard copy* of the source and object program for the purpose of debugging.

8-3.1 One Pass, Two Pass Assemblers

Assemblers are designated *one-pass* or *multi-pass*, according to the method of implementation. A *pass* is defined as one scan of the source program by the assembler. When the source program is on cards, this means a single reading of the card deck. In a two-pass assembler, the first pass allocates storage and sets up tables of all symbols and labels addressed by the source program. The second pass generates code into the previously allocated storage from data in the tables. If certain restrictions are imposed (e.g., storage allocation at the head of a program and a method of providing for forward references), an assembler that makes only one pass can be implemented.

Figure 8-1 shows the *intermediate* output devices for a two-pass assembler. The intermediate program is output on the SCRatch (SCR) device. If there are no mass-storage devices on the system, there is no intermediate output. In this case, the source program is re-positioned and re-read for the second pass. When a mass storage device is available (disc and magnetic tape), pass-one copies the source onto the device after which the source is re-read for the second pass. The advantage of the mass storage method is that the source program is handled only once.

8-3.2 Coding

The assembly process allows the programmer to write *code* in symbolic rather than numeric terms. This means that program coding begins to resemble the English language rather than a list of obscure numbers. Figure 8-2 shows an assembly language coding form. There are three major parts to each assembly language statement: the *label field*, the *operation field*, and the *operand list*. Any location in core memory may be given a symbolic name or label, (e.g., Sam or S1), provided that the name conforms to certain rules (i.e., names must begin with a letter and no spaces allowed). Instructions or data values are identified by this name in the label field.

If an instruction is to be placed in any location in memory, the three letter mnemonic (as specified in Section 2 and Appendices O, P, and Q) is used instead of a hexadecimal operation code. The *operand list* can be symbolic names or expressions or literal values. Comments may be added after the operand list.

The following example demonstrates how the solution to the equation

$$A = | B - C |$$

might be coded in both symbolic and machine language.

	Symbolic	Location	Machine Language
START	LDW B	0	8007
	SUB C	1	B008
	SAP	2	0810
	CMP	3	0110
	STW A	4	7006
	HLT	5	0000
A	DATA 0	6	0000
B	DATA	7	
C	DATA	8	

Note: A, B, and C represent the memory location containing the data and not the data themselves.



700 CODING FORM

PREPARED BY _____ DATE _____
 CHECKED BY _____ PHONE _____ PROBLEM NO. _____ COST ACCT. NO. _____
 PAGE _____ OF _____

LABEL	OPERATION	700 STATEMENT																				MODULE IDENTIFICATION	SEQUENCE NUMBER								
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20										
1		OPERAND LIST. " BLANKFIELD "	COMMENTS																			11	12	13	14	15	16	17	18	19	20
2																						21	22	23	24	25	26	27	28	29	30
3																						31	32	33	34	35	36	37	38	39	40
4																						41	42	43	44	45	46	47	48	49	50
5																						51	52	53	54	55	56	57	58	59	60
6																						61	62	63	64	65	66	67	68	69	70
7																						71	72	73	74	75	76	77	78	79	80
8																						81	82	83	84	85	86	87	88	89	90
9																						91	92	93	94	95	96	97	98	99	100
10																						101	102	103	104	105	106	107	108	109	110
11																						111	112	113	114	115	116	117	118	119	120
12																						121	122	123	124	125	126	127	128	129	130
13																						131	132	133	134	135	136	137	138	139	140
14																						141	142	143	144	145	146	147	148	149	150
15																						151	152	153	154	155	156	157	158	159	160
16																						161	162	163	164	165	166	167	168	169	170
17																						171	172	173	174	175	176	177	178	179	180
18																						181	182	183	184	185	186	187	188	189	190
19																						191	192	193	194	195	196	197	198	199	200
20																						201	202	203	204	205	206	207	208	209	210
21																						211	212	213	214	215	216	217	218	219	220
22																						221	222	223	224	225	226	227	228	229	230
23																						231	232	233	234	235	236	237	238	239	240
24																						241	242	243	244	245	246	247	248	249	250
25																						251	252	253	254	255	256	257	258	259	260

Figure 8-2. Assembly Language Coding Form

Obviously it is easier to code and subsequently understand what is intended by the code when the symbolic rather than the machine language coding is used.

The last three lines of symbolic code in the preceding example show the use of a *pseudo instruction* in defining space for data storage.

8-3.3 Pseudo Instructions

Pseudo instructions (or operations) are assembly directives that allow the user to specify program parameters at assembly and load time. Pseudo instructions are divided into two types: *symbol and data definition*, and *assembler control* (see Table 8-3).

The symbol and data definition types of pseudo instructions facilitate the insertion of data into the program and they also reserve blocks of storage and equate symbols to expressions.

The assembler control directives signify absolute or relocatable loader text, end of assembly, library control, conditional processing, repetitive code, subroutine calls, and procedure definitions. Refer to Section 2-2 of this manual for a detailed description of the operation of each pseudo instruction and assembler control directive.

8-3.4 Procedures

While coding a program, the programmer often finds that he uses certain sequences of instructions many times with only minor differences, such as the names of variables. For example, a program may require frequent repetition of the absolute value equation:

$$A = |B - C|$$

where the variables A, B, and C are different each time. The general sequence of code required to solve the equation is:

```
LDW    B
SUB    C
SAP
CMP
STW    A
```

The SYM II assembler permits the programmer to define an operation code which represents a sequence of instructions or a *procedure*. The programmer can cause the assembler to generate a previously defined sequence of code or procedure by writing a single operation code. For example, the above sequence of code for the absolute value could be defined as the procedure ABSV (absolute value). Then whenever the statement ABSV A, B, C is encountered, the assembler will automatically create the correct sequence of instructions and insert these instructions *in-line* with the new variables placed in the proper position.

SYM II lets the programmer create procedures through the use of the PROC and ENDP directives.

8-3.4.1 Using the Procedure

To use a procedure the programmer defines an instruction sequence in terms of the dummy operands P(1), . . . , P(N). The assembler retains this *procedure definition* in memory. When the name of the procedure is encountered in the operation code field of a statement, the instruction sequence from the definition is assembled with the dummy operands replaced by the values of the arguments in the *reference line*. The procedure definition must precede its reference lines.

The name of the procedure is defined in the label field of the PROC statement. Only the first four characters are used as the procedure name. Labels should not be used in procedure definitions with the exception of the PROC statement. A definition is terminated by the ENDP statement.

Example:

Label	Operation	Operand
ABSV	PROC	
	LDW	P(2)
	SUB	P(3)
	SAP	
	CMP	
	STW	P(1)
	ENDP	

Table 8-3. Pseudo Operation Summary (See Section 2.2)

<u>Symbol and Data Definition</u>			
Operation Code	Function	SYM I	SYM II
DATA	Defines Word Data	Yes	Yes
BYTE	Defines Byte Data	Yes	Yes
TEXT	Defines Alphanumeric Data	Yes	Yes
DPI	Defines Double Precision Integer Data	No	Yes
REAL	Defines Two Word Floating Point Data	No	Yes
EPRL	Defines Three Word Floating Point Data	No	Yes
DPRL	Defines Four Word Floating Point Data	No	Yes
EQU	Equates a Symbol to An Expression Permanently	Yes	Yes
IS	Equates a Symbol to An Expression Temporarily	No	Yes
RES	Reserves a Block of Storage	Yes	Yes
<u>Assembler Control Directives</u>			
Operation Code	Function	SYM I	SYM II
ORIG	Sets the origin of a program	Yes	Yes
END	Defines end of program	Yes	Yes
LOAD	Directs the relocatable loader to load specified routines	Yes	Yes
NTRY	Defines the entry point to a program	Yes	Yes
LIBR	Generates ID labels for system library	Yes	Yes
DO	Provide repetitive code generation	No	Yes
TRUE, FALSE, ENDC	Provides conditional processing	Yes	Yes
PROC, ENDP	Provides Procedure Capability	No	Yes
SUBR, EXIT	Generates linkage to subroutines	No	Yes

If fewer reference line arguments are given than are required in the definition, the absent parameters are assigned the value of zero.

Examples:

Programmer Codes	Operation	Operand
	ABSV	A, B, C

Assembler Inserts:	Operation	Operand
	LDW	B
	SUB	C
	SAP	
	CMP	
	STW	A

8-3.4.2 Conditional Processing Within Procedures

The dummy parameter P(0) has a special meaning when used in a procedure definition. P(0) is assigned a value equal to the number of parameters in the procedure reference line. This feature allows conditional processing of the procedure text, based on the number of parameters given in the reference line.

Example:

STZ	PROC	
	CLR	
	DO	1, P(0), 1
	STW	P(7)
	ENDP	

If reference lines for the previously defined STZ procedure are:

STZ	A,B,C
STZ	Q

The equivalent symbolic code will be:

CLR	
STW	A
STW	B
STW	C
CLR	
STW	Q

8-3.4.3 Procedures Within Procedures

A reference line to a previously defined procedure may appear in a procedure definition (i.e., procedures may be nested to any level). For example, consider the equation

$$E = D - |B - C|$$

The absolute value procedure may be used in defining a new procedure SUBA.

Example:

Label	Operation	Operand
SUBA	PROC	P
	ABSV	P(3),P(4),P(5)
	LDW	P(2)
	SUB	P(3)
	STW	P(1)
	ENDP	

The reference and subsequent code generation for SUBA would be:

Reference:

Operation	Operand
SUBA	E,D,A,B,C

Generates:

Operation	Operand
LDW	B
SUB	C
SAP	
CMP	
STW	A
LDW	D
SUB	A
STW	E

8-3.5 Subroutines

The use of a *subroutine* is another method to solve the problem of repetitive code generation. A subroutine is a specific set of instructions that solve some generalized problems by using parameters from a main program *calling sequence*. A subroutine differs from a procedure in that the coding for a given problem (e.g., the *Store Zero* problem defined in Section 8-3.4.2) appears only once and not in-line as it does with a procedure. Subroutines normally require less space (since they

appear only once) and take more execution time than procedures.

Consider the Store Zero problem again. Instead of clearing three independent memory locations (A, B, C), it is desired to clear a consecutive number of locations beginning with location A and ending with location B.

The coding to solve this problem with a subroutine is as follows:

Main Program Call:

	Operation	Operand	Comment
Call →	SMB	CLEAR	Global Mode Set
	JSX	CLEAR	
	DATA	A	
	DATA	B	
Return →			

where A is the starting address and
B is the ending address inclusive.

CLEAR Subroutine:

Label	Operation	Operand	Comments
CLEAR	DATA	0	Return - 2 stored here
	STX	CLEAR-1	Save return
	STW	ASAV	Save accumulator
	LDW *	1	
	STW	ENDADD	Save End Address
	LDX *	0	IXR ← Start Address
REPEAT	CLR		(ACCUM) ← 0
	STW *	0	
	CXA		(ACCUM) ← Current Address
	CMW	ENDADD	
ASAV	SNE		
	JMP	GETOUT	Clear Complete
	IXS	1	Always Skips
GETOUT	DATA	0	
	JMP	REPEAT	Return to clear next address
ENDADD	LDW	ASAV	Restore accumulator
	LDX	CLEAR-1	
	JSX *	2	Exit to Main Program
	DATA	0	

The preceding example shows that the subroutine takes fewer *calling* locations than a procedure designed for the same result.

Subroutines may be put on the system library and called from different main programs at different times, while procedures must be defined in a main program each time they are used.

8-3.6 SYM I and SYM II Assembler Review

Raytheon 706 software includes two assemblers. A one-pass assembler, SYM I, is available for BASIC systems without mass storage. It provides direct on-line assembly using the ASR 33/35 as the primary input/output device. A more powerful two-pass assembler, SYM II, is available for STANDARD, EXTENDED and ADVANCED systems. It includes procedure capabilities and other advanced features such as conditional processing.

8-3.6.1 SYM I

SYM I (Appendix K-1) accepts symbolic paper-tape programs prepared by PREP or source statements entered at the ASR typewriter keyboard. In the latter case, errors are diagnosed on-line. The programmer can immediately correct the statement at which time the assembler will accept and translate it to relocatable or absolute object code. SYM I is a one-pass assembler with a language subset of SYM II. Forward definitions are allowed. Statement error diagnostics are printed on the ASR 33/35.

Multiple assemblies can be run without reloading SYM I after each assembly. A special feature of SYM I permits the program to be assembled directly into memory and executed immediately. Any combination of input/output assignments for ASR paper tape reader, ASR keyboard, ASR paper tape punch, High Speed paper tape punch or High Speed paper tape reader may be made symbolically at the ASR keyboard. This versatility is provided during the initial phase of each assembly.

8-3.6.2 SYM II

SYM II (Appendix K-2) features a large scale assembly for small machines. The assembler incorporates symbolic representation of machine instructions and the facility for user-generated procedure. In the STANDARD configuration, SYM II is assembled by reading the source program twice from any media--paper tape, cards magnetic tape or disc. Slow punching of intermediate text, which is normally found with two-pass assemblers, is eliminated. On the first pass of the source program, a diagnostic test is performed with

programming error print-out. Correction of errors is thus possible with only one-pass and is done at the source level. A symbolic table is also generated. The second pass of the source program replaces the symbols with numeric equivalents, provides a listing, and produces an object program.

Pseudo operations in the assembler permit conditional processing of statements. As a result, one source program may be altered slightly to yield a variety of unique object programs.

Multiple assemblies can be run without reloading SYM II after each assembly. Either pass can be repeated or restarted without limitations.

Standard procedures provide extended instruction capability. The user may define additional procedures peculiar to his application. Floating point and mathematical operations are symbolically referenced at a macro level. An automatic call to the routine resident on the library is part of the macro sequence. A powerful, simplified input/output repertoire, especially developed for ease of communication, is provided by the assembler. The two-pass operation produces literals and symbols lists. Source data definition and transfer points labeled later in the program sequence (forward-defined equates) are permitted.

Object binary text is absolute or relocatable and can be output to paper tape, cards, magnetic tape, or disc. Full flexibility of input/output is obtained by logical unit identification of peripherals and assignment under program control.

In the EXTENDED and ADVANCED configuration, the assembly process is further automated by reading the source program only once and recirculating its image via the mass storage device--disc or magnetic tape.

Symbolic data labeling is easily accomplished by directives that identify alphanumeric characters, special characters, decimal and hexadecimal constants. The assembler also exploits the unusual byte manipulation instructions in the 706 hardware. Linkages to and from user-generated and library routines are quickly established. Memory is allocated and the position of the program in absolute memory is established with additional directives.

8-4 COMPILERS

8-4.1 Introduction

The previous section discussed assemblers which translate source statements into machine instructions on a one-for-one basis. This section discusses language processors that translate source statements into multiple instruction sequences. These processors are called compilers. The Raytheon 706 has two compilers: FORTRAN IV and Conversational FORTRAN. FORTRAN (FORmula TRANslation) is oriented toward mathematical problems and is easy to learn by non-programmers. Since FORTRAN is problem oriented rather than machine oriented, programs written in FORTRAN may be compiled on different machines. This means that FORTRAN programs are largely machine independent and may be transferred from one computer to another with minor modifications. This is not true of assembly language programs.

FORTTRAN has fewer statement types than the SYM assembly language, but the statement formats are more complex. Since the statements do more, fewer statements are required. This improves program readability and reduces the chances of error.

Usually, equivalent assembly language and FORTRAN programs result in the FORTRAN program executing slower and requiring more core storage. This fact is largely irrelevant in many cases since much less proficiency and check-out time is required to obtain the debugged FORTRAN program.

FORTTRAN has two language capabilities which are clearly superior to assembly language: algebraic calculations and formatted input/output. The translation of algebraic equations is efficiently done by a compiler and closely approaches the capability of a good assembly language programmer. Similarly, the conversion of data values and character strings is rather tedious in assembly language, but quite simple in FORTRAN.

Assembly language programs can be used effectively to do bit manipulation, high speed integer computation loops and in general reduce the

storage requirement. Raytheon FORTRAN IV allows the user to gain back many of the assembly language advantages by accepting SYM II statements intermixed with the FORTRAN statements. This capability allows the user to select the features most advantageous to his program.

8-4.2 Real-Time FORTRAN IV

8-4.2.1 General

The Raytheon 706 is fully capable of dealing with real-time systems requirements through a comprehensive expanded FORTRAN IV compiler.

The American Standards Association has established a FORTRAN specification to encourage computer manufacturers to standardize their FORTRAN systems. Raytheon Computer observes this standard and has added capabilities found in no other small machine FORTRAN compilers. In addition to object programs running in a real-time priority interrupt environment, Raytheon FORTRAN IV includes:

- Intermixing FORTRAN statements and symbolic assembly language. This includes the use SYM II procedures.
- Two additional data types handle double precision integers and 9-digit floating point numbers.
- Expressions may be composed of items of any data type.
- Expressions are allowed in output lists.
- Arrays are not limited in number of dimensions.

Real-time FORTRAN IV makes optimum use of storage, both during compilation and for the compiled programs. Tables generated during the compiling process are dynamically allocated and readjusted to maximize the use of working storage. Intermediate output from the compiler is packed by utilizing the free-format of SYM II. The advanced compiler techniques and speed of the 706 result in very rapid compilation and execution

of user programs. The compiler translates the FORTRAN source statements into SYM II statements in one pass at the rate of 1200 statements per minute. The output from the compiler is called the intermediate text. The SYM II assembler reads the intermediate text two times and produces a binary object program. The process is shown in Figure 8-3. The object program may be loaded and executed as shown in Figure 8-4.

Compile and execute capability is provided if a disc or three magnetic tapes are available. If a mass storage device is not available, the intermediate, text is punched out and assembled as a symbolic program. The compile and execute (GO) process is shown in Figure 8-5. The real-time FORTRAN IV library is extensive and includes over 30 external functions and 35 intrinsic functions. See Appendix K-4.

8-4.2.2 Program Segmentation

Real-Time FORTRAN IV in conjunction with the Real-Time Operating System (RTOS), or the Multi-programming System (MPS) allows a unique method of program segmentation. FORTRAN IV programs may request the system to load and execute additional main programs by the use of a CALL QUEUE statement. Thus, very large programs may be fragmented into smaller overlaying segments (called tasks) with each segment queuing its successor.

The form of the queuing statement is:

```
CALL QUEUE ('SEGMENT', I, J, K)
```

The name SEGMENT must be an eight-character name. Trailing blanks may be used to fill up to eight characters for shorter names. The program SEGMENT must contain the following coding:

```
S      LIBR  SEGMENT
S      NTRY  SEGMENT
S SEGMENT EQU $
      CALL  FETCH (II, JJ, KK)
      .
      .
      .
      END
```

By executing a CALL FETCH (II, JJ, KK), up to five integer arguments may be passed to the program SEGMENT from the program that contains the CALL QUEUE statement. The CALL FETCH causes II = I, JJ = J, KK = K. Executing more than one CALL QUEUE statement will cause the system to execute the tasks in a chronological order of first in, first out. The total number of segments queued at one time is determined at system generation. Communication between segments is accomplished by initially requesting a resident labeled COMMON data block by the X-RAY directive, :CB.

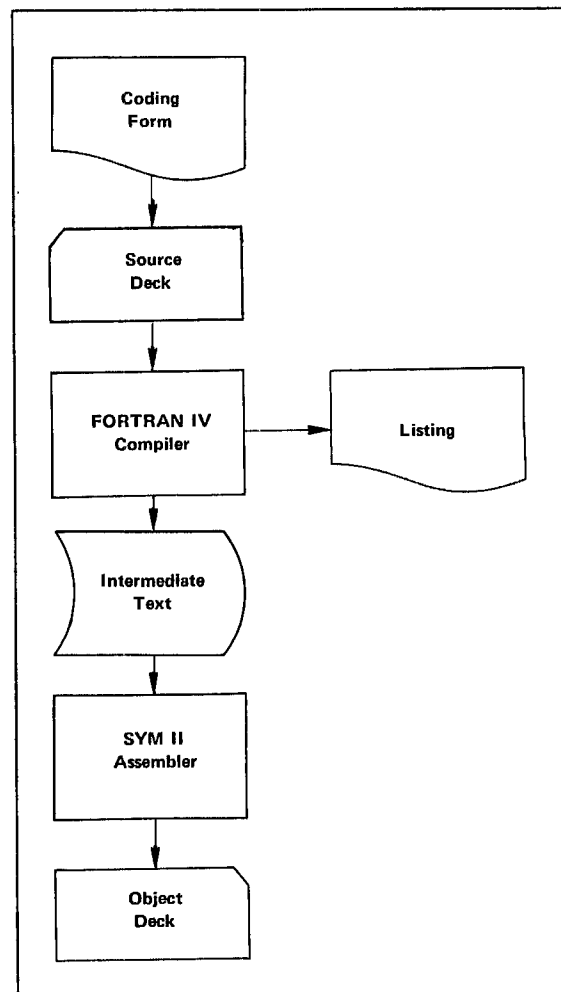


Figure 8-3. FORTRAN IV Compilation

8-4.3 Conversational FORTRAN

8-4.3.1 General

Since many users do not need the extensive capabilities of FORTRAN IV to solve their applications problems, a useful and convenient Conversational FORTRAN system was designed for the 706 user with a minimal configuration. Capabilities have been added to aid the real-time user in his special applications.

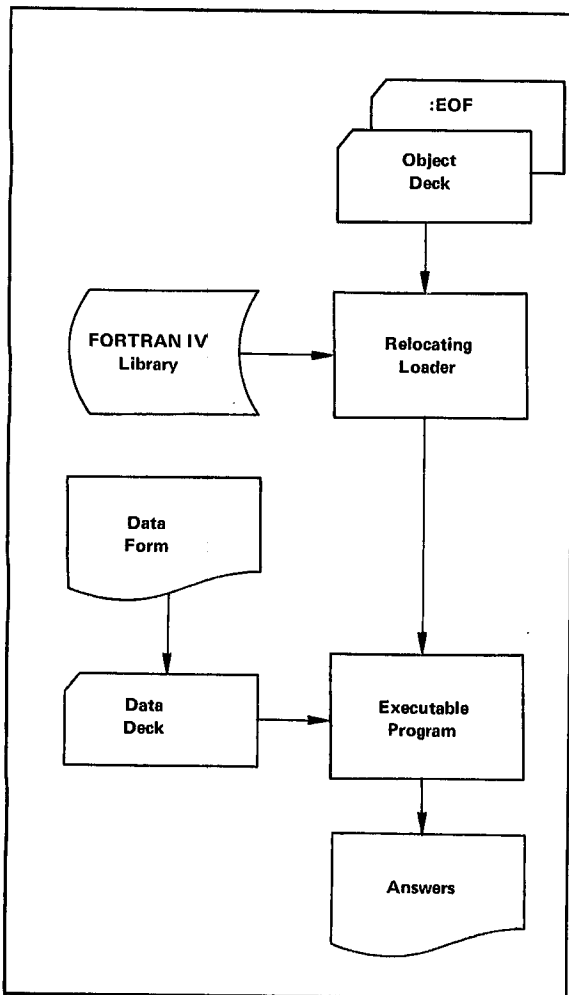


Figure 8-4. Executing a FORTRAN IV Object Deck

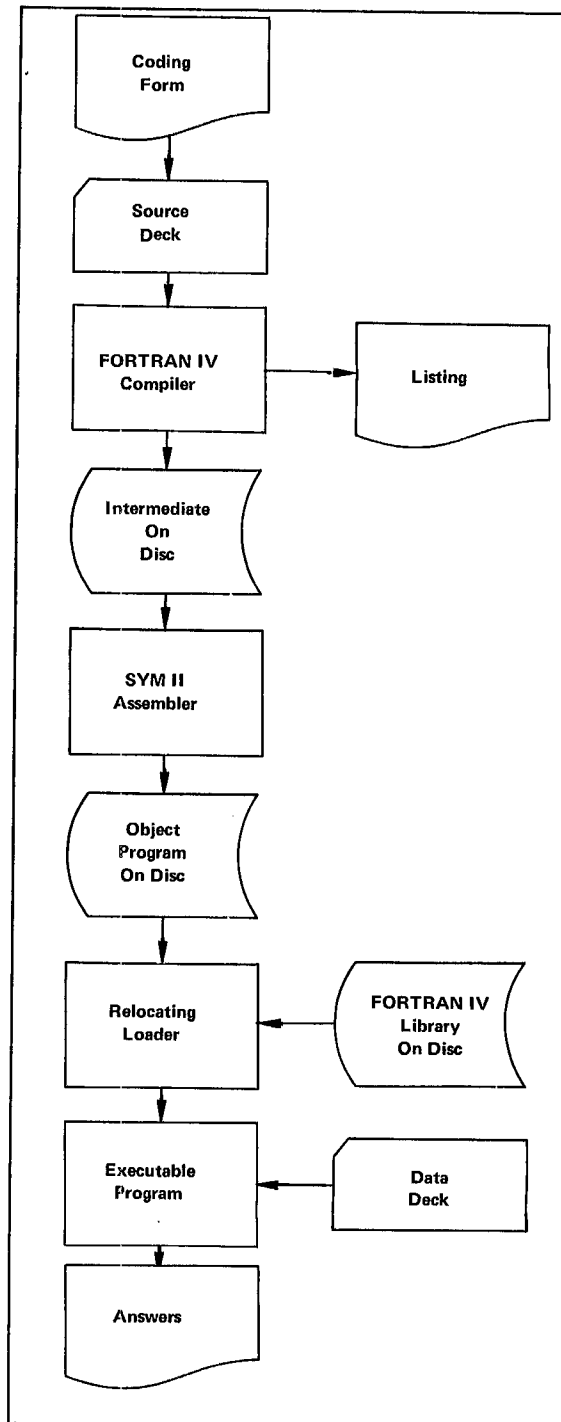


Figure 8-5. FORTRAN IV Compile and Execute (GO)

Conversational FORTRAN offers the advantages of minimal tape handling and loading, rapid compile and go operation, and program preparation in a conversational mode. Provisions for interfacing assembly language programs are included for special purpose subroutines and arrays. Free form statement formats eliminate spacing problems.

Conversational FORTRAN consists of two subsystems; the compiler and the run-time system.

8-4.3.2 Conversational FORTRAN Compiler

The one-pass compiler translates the FORTRAN (Appendix K-5) version of the problem into the language of the run-time system. When errors are detected by the compiler, they are listed as diagnostic messages. The compiler may be used in either of two modes: program preparation (PREP) or compile and execute (GO). See Figures 8-6 and 8-7.

In the PREP mode, the compiler diagnoses each statement for syntax errors. The error-free statements are retained in memory and punched out after the END statement is encountered. Erroneous statements are not retained in memory and may be corrected and re-read by the compiler. The resultant program may be compiled in the GO mode to construct the executable program.

Programs compiled in the GO mode may be output in absolute binary format or executed directly by the run-time system.

8-4.3.3 Run-Time System

The run-time system consists of the library functions and input/output routines for executing a FORTRAN program. After compilation of the main program and additional subprograms, the run-time system is loaded. Assembly language programs may be loaded and the FORTRAN main program executed.

The run-time system may be operated in either normal or trace modes. The normal mode executes the program and all printed results must be output by WRITE statements. In the trace mode all values on the left side of replacement statements and/or program branches are listed. The trace mode provides the user with the means to easily debug his FORTRAN program.

8-4.3.4 System Design

The prime design criterion for the Conversational FORTRAN system is for the convenience of the user with a teletype and 4K of memory. Since the teletype is a slow input device, minimizing the amount of paper tape that must be read is of major importance. To compile and execute a FORTRAN program, only three paper tapes need be loaded: The FORTRAN compiler, the FORTRAN source program, and the run-time library. This contrasts with up to twelve tapes on some computers. To accelerate the production of useful programs the on-line diagnostic and program preparation compiler mode is used. This eliminates alternatively loading a compiler to find errors and an editor to remove them.

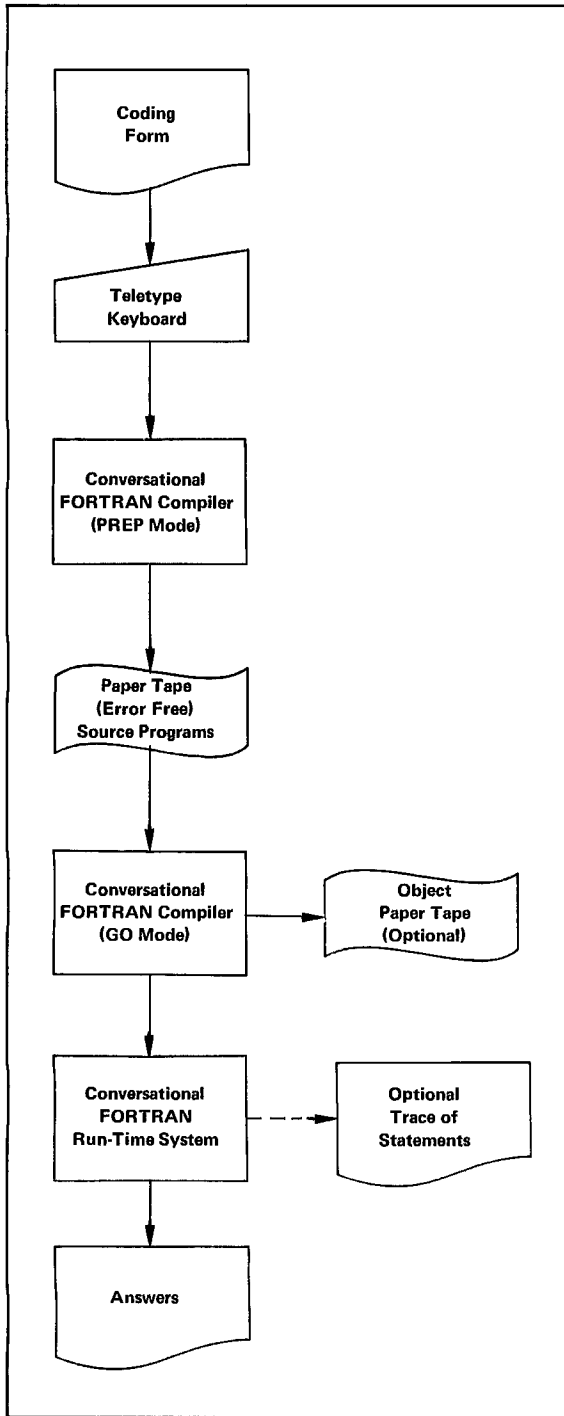


Figure 8-6. Conversational FORTRAN In Interactive Use

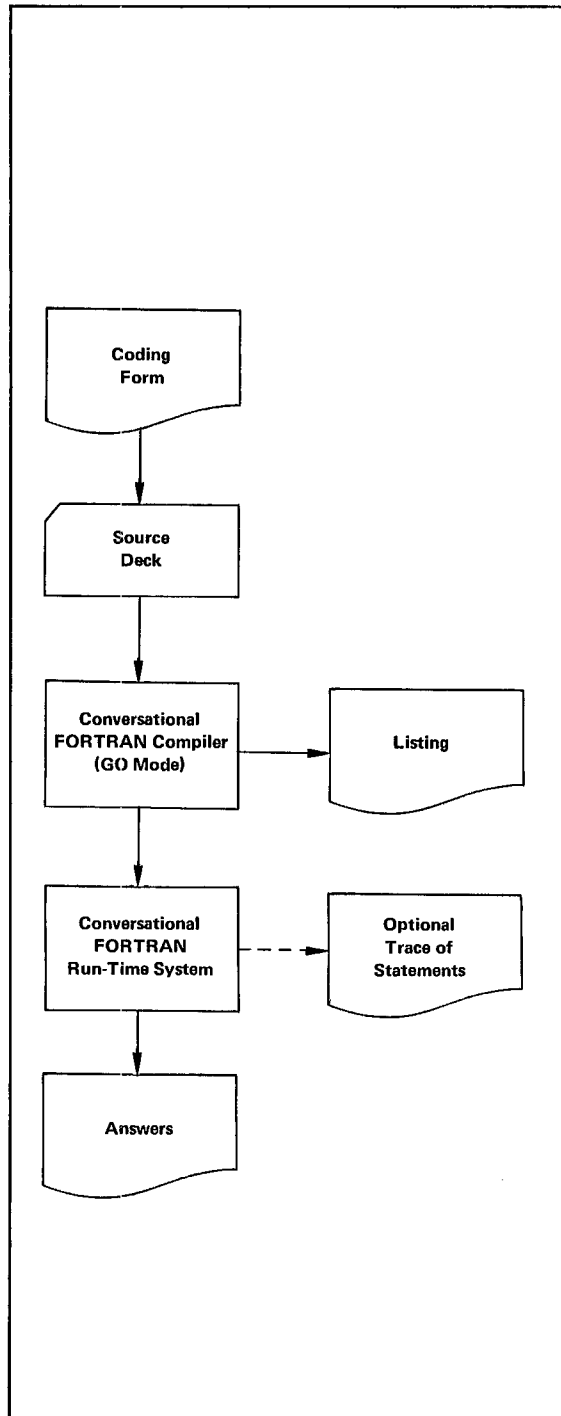


Figure 8-7. Conversational FORTRAN In Batch Process Use

8-5 LOADERS

A loader is a program which inputs or *loads* binary programs into the memory of the computer. In general, there are two types of loaders, absolute and relocatable.

An absolute loader inputs binary programs directly into fixed or absolute memory locations. The absolute locations into which the program is loaded were specified when the program was assembled.

A relocatable loader permits the program location in memory to be moved or *relocated* at load time without requiring the program to be reassembled. A relocatable loader will also *link* together several program subroutines which have been assembled individually. Thus, sometimes relocatable loaders are referred to as *linking loaders*.

In order to relocate a program or subroutine in memory, the relocatable loader must know whether each word loaded into the computer is an invariant data value or an instruction whose address value will vary depending on the location into which the program is loaded. Therefore, loader codes are inserted between each word to identify for the relocatable loader the type of word that follows.

Absolute and relocatable loaders each have their advantages and disadvantages. Absolute loaders are usually very simple and compact programs because absolute programs are in a format which permits them to be read directly into memory, with a minimum of computer processing. Thus, absolute loaders load programs faster and allow more space for loading larger programs. Also, programs in absolute format occupy less space on paper tape, cards, magnetic tape or disc. However, if it is desired to change a location of a program or revise one small section of a program, the entire program must be reassembled. It is also difficult to link together independently assembled absolute programs since the user must allocate memory.

On the other hand, the relocatable loader permits the user to assemble program subroutines independently. The relocatable loader automatically

allocates memory and links together the users subroutines. General subroutines which are frequently used can be stored in a relocatable library. The loader automatically scans the library to load any required subroutines.

If desired the relocatable loader will print a *load map* to tell where it has loaded the subroutines. If one subroutine in a program is changed, only that one subroutine need be reassembled. The relocatable loader will automatically link the new subroutine to the others and allocate memory accordingly. However, relocatable loaders are much larger than absolute loaders and less space is available for the users programs. Loading takes longer because more computer processing is required and relocatable text uses more space on paper tape, cards, magnetic tapes, or disc. A description of 700 series loaders follows.

8-5.1 Absolute Loaders

Absolute loaders are usually very simple and compact programs because absolute programs are in a format which permits them to be input directly into memory.

8-5.1.1 Absolute Program Format

A program in absolute format consists of two binary records. The first record is a one-word record which contains the word origin or starting location of the program. This location record tells the absolute loader at which word location in memory to start loading the program. The second binary record contains an exact binary image of the program to be loaded.

The program record contains an extra word which is not part of the program as the last word of the record. This word contains a check-sum in the right byte portion. The check-sum is similar to parity and is used to verify that the program was read correctly. Check-sums are created by SYM I, SYM II, and the Linking Absoluter. However, Trace/Debug does not create a check-sum.

Figures 8-8 and 8-9 show the general format of an absolute program. The end-of-file record is used to

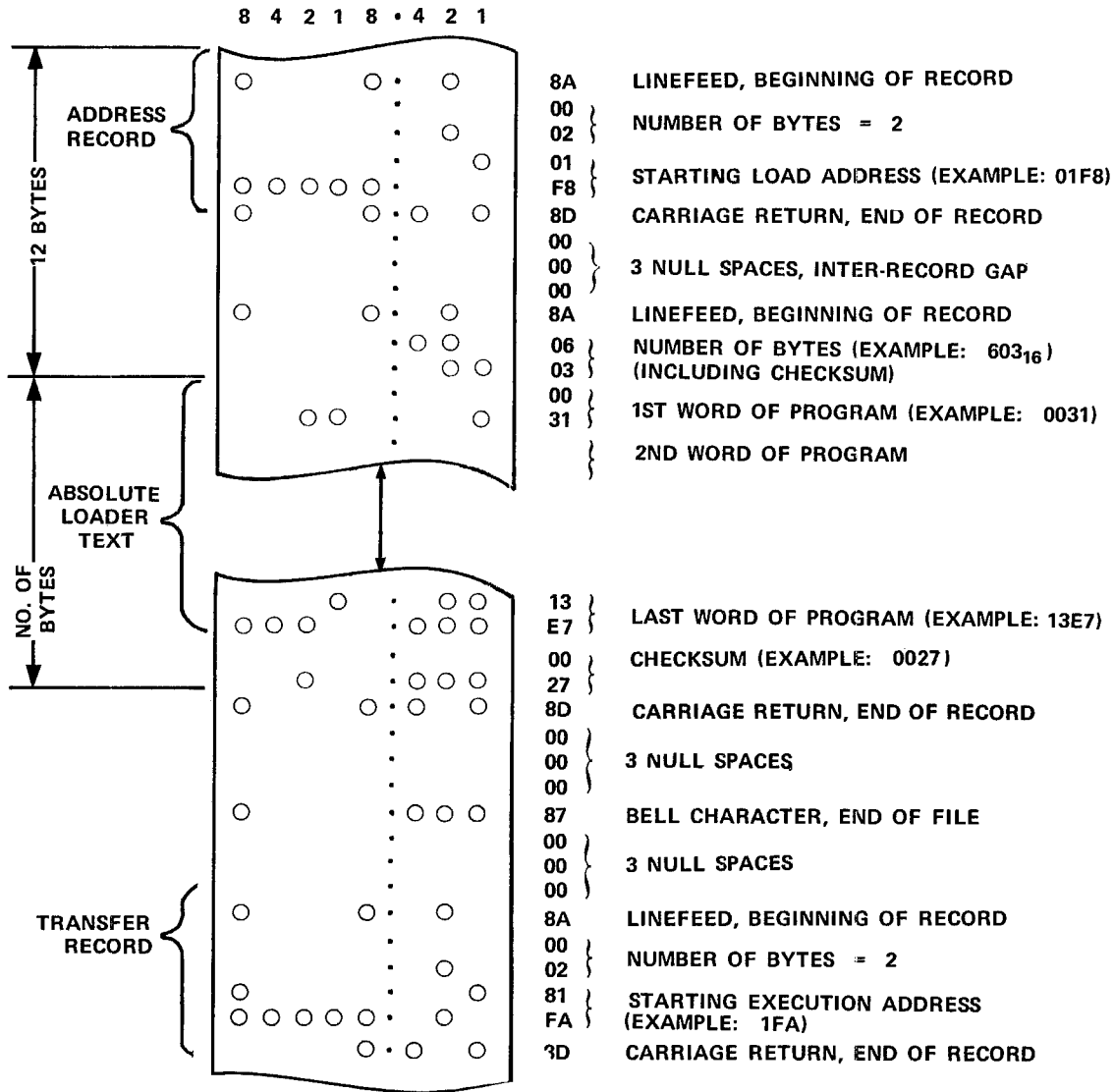


Figure 8-8. Paper Tape Absolute Program Format — SYM I, II

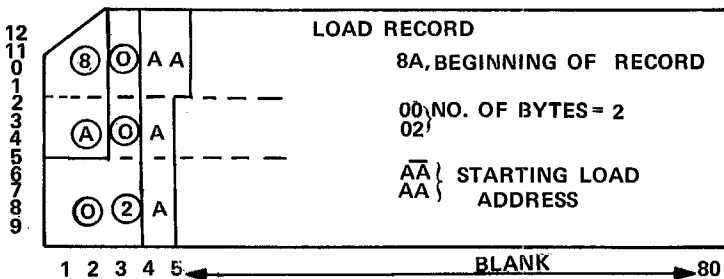
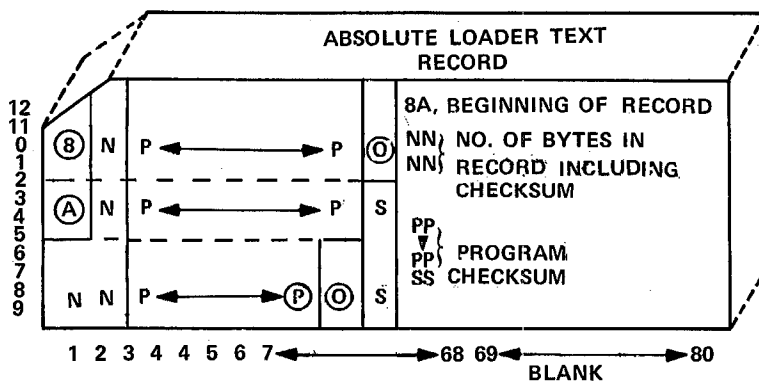
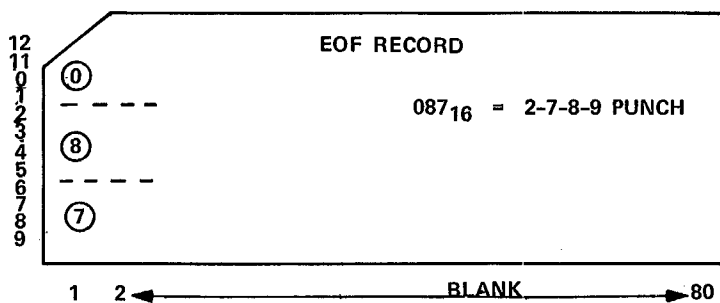
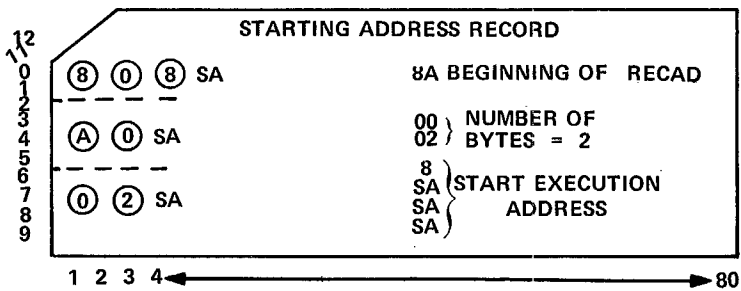


Figure 8-9. Card Absolute Program Format

signify the end of a string of programs to the absolute loader. The end-of-file record is a single ASCII BELL character (X' 87') for paper tape or a 2-7-8-9 punch in column one for cards. The end-of-file record must be followed by an execution address record or a second end-of-file record. The execution address record is a one-word record which contains the starting address of the program.

Note that the exact form of binary records depend upon the peripheral type. For example, binary paper tape records always begin with the line feed character (X' 8A'). The next two frames make up a word which contain a count of the number of bytes that make up the actual data in the binary record. The record is terminated by a carriage return (X' 8D') followed by three blank frames.

Figure 8-8 and Figure 8-9 show the paper tape and card formats of an absolute program that begins at location 01F8. Refer to Section 8-6 on device drivers which describe the exact form of binary records for various devices.

8-5.1.2 Absolute Load Routine

In STANDARD, EXTENDED and ADVANCED systems the absolute load routine is included as an integral part of the resident I/O monitor. Control may be transferred to the absolute loader by the :AL directive or by transferring to location X'78' in the monitor link cells.

The absolute load routine inputs programs in absolute format from the Binary Input (BIN) device.

The absolute load routine continues loading successive programs until an end-of-file record is read. The absolute load routine verifies the checksum and will output on the teletype the message "ACK!" when an error is detected. If a single end-of-file mark is followed by a one-word record containing an execution address, the absolute load routine automatically transfers to that execution address. If successive end-of-file marks are read following the program records, the absolute load routine halts. At this time the programmer may enter manual program connections and manually

transfer to his program. However, if RUN is depressed the absolute loader will continue loading.

If end-of-file marks and execution address are missing completely, the absolute load routines will continue to select the BIN device for more records. At this time the user may HALT and RESET the computer, set the program counter to the starting location of his program, and begin execution by depressing RUN.

8-5.1.3 Absolute Load Routine - BASIC

The absolute load routine is an integral part of the X-RAY EXEC BASIC in BASIC systems. The X-RAY directive AL causes X-RAY to jump to the absolute load routine which is located at symbolic location S.FILL. The absolute load routine will read a binary paper tape which is in absolute format from the Binary INput (BIN) device. The BASIC absolute load routine does *not* verify the check-sum; however, the check-sum is read into the cell immediately following the program.

If BIN is assigned to the teletype, the paper tape reader must be turned on manually. This should be done after the AL directive has been typed. After the program has been loaded the computer will halt to allow the operator to turn off the reader. When the Run button is pushed, the SYStem INput (SYSIN) unit will be selected for another X-RAY directive. At this time the user may transfer to the starting location of the program with the Transfer (T) directive. Or another program can be loaded with the AL directive. Note that the Absolute Load Routine in BASIC systems only loads one program at a time and does not recognize the end-of-file character and subsequent transfer location.

8-5.1.4 Initial Loader

The Initial Loader (INITLOAD) is a stand-alone absolute loader which can load absolute programs without requiring a resident I/O monitor. Three versions of the program are available. One version loads absolute programs from paper tape (either

high speed reader or teletype), a second version from cards, and the third from DIO magnetic tape.

Each version of the initial loader can be loaded by depressing the appropriate Bootstrap switch on the 706 console. The initial loader then automatically loads any absolute programs which follow on the selected device. Thus, INITLOAD can be appended to the front of any absolute program tape, and the combined tape will load and execute automatically by depressing the Bootstrap and Run switches. For example, it is normal practice to append the initial loader to the front of the resident I/O monitor program. Once the monitor has been loaded, any other absolute programs can be loaded with the absolute load routine which is imbedded as the monitor. Check-sum and absolute program format are verified by INITLOAD. Refer to Section 7-4 and Section 9-2 for a description of the initial loader bootstrap program (hardware bootstrap).

<u>Location (Hex)</u>	<u>Contents (Hex)</u>
0	0020
1	8004
2	03E9 (03D9 for HSPTR)
3	1003
4	02ED (02DD for HSPTR)
5	0800
6	0401
7	0010
8	0638
9	300A
A	0010

Figure 8-10. Absolute Bootstrap Program

8-5.1.5 Absolute Bootstrap

The Absolute Bootstrap program (Figure 8-10) is a highly specialized form of the absolute loader which has been written to utilize as few instructions as possible. This is so that absolute programs can be loaded manually through the computer console by hand-entering the absolute bootstrap program.

In order to keep the bootstrap program as small as possible, it is assumed that the index register of the computer has been manually initialized. When loading absolute paper tape programs using the absolute bootstrap, first place the starting byte address minus 12_{10} bytes locations into the index register. Checking Figure 8-8 it can be seen that absolute paper tape formats have 12_{10} leader bytes before the first word of the program appears. For example, if you wish to bootstrap an absolute paper tape that starts at word address 01F8, the address 03E4 must be placed in the index register:

$$01F8 \text{ word address} = 03F0 \text{ byte address}$$

$$03F0 - 000C = 03E4$$

Place the tape in the teletype or high-speed reader and press RUN. The absolute program will load with the check-sum not verified.

8-5.2 Relocatable Loader

In order to relocate programs and link external references to external definitions, the relocatable loader must be able to distinguish between different types of data words. For example, the loader must be able to differentiate between a data word whose value remains constant regardless of location and an instruction whose address field changes value depending upon where the program is located in memory.

Loader codes are placed in front of each data word or string of words so that the loader can identify the type of data which follows. A loader code consists of a full byte.

8-5.2.1 Relocatable Loader Text

Loader text appears to the loader as a continuous string of bytes, made up of loader code bytes and data alternating in the text stream. This text stream is referred to as relocatable loader text.

The codes have been defined as shown in Table 8-4. The first column is the code in hexadecimal. The second is a mnemonic which has been assigned to that code. The third describes the format of data which follows the code in the text stream.

Repeatable loader codes are included in order to conserve space in the relocatable text. A repeatable loader code tells the loader that the next $n + 1$ words are of the type specified, where n is a number between 0 and 15. For example, if six consecutive words are absolute program words, the single loader code C5 would specify this. Thus, it is not necessary to have a loader code precede every computer word. Table 8-4 gives all of the repeatable code definitions.

8-5.2.2 External Strings

In relocatable assemblies, it is permissible to reference symbolic locations which are external to the program being assembled. These references are referred to as *externals* or *external labels*. The programmer may link together several independently assembled programs at load time using external references.

One of the primary functions of a relocatable loader is to *fix up* external references to the correct addresses. A given program may reference a particular external label in several places so it is necessary for the loader to be able to find and fix up all of these references to the correct address. The assembler does this by stringing external labels from word-to-word. That is, the first reference points to the location of the second, the second to the third, etc. The last reference contains a special code to identify the end of the string. When following these strings, to fill them with the correct address, the following conventions are observed. If the end of the string is specified by loader code 7 (EXTN15), the string is a full 15-bit string and each word of the string is filled with a 15-bit address value. If it is specified by code 2, the string is an 11-bit string and bits 0-4 contain an op code and index bit. Byte externals are not permitted by the assembler, so the byte reference op codes are used to flag special requirements in the string as follows:

- 3- 15-bit address. This word will be filled with the full 15-bit value of the external.
- 3X- 15-bit address with the sign bit true. This code is used to set the argument list ending bit.

- 0- SMB. This word will be filled with an SMU or SML instruction, which will set the bank to the one containing the referenced external.

All other words are filled with the 11-bit value of the referenced external with bits 0-4 preserved unchanged.

8-5.2.3 Record Blocking of Loader Text

Loader text appears as a continuous string of bytes to the loader. For ease of validity checking, it is blocked into records of 94 or fewer bytes. The first data byte of each loader text record is a zero. This byte helps to identify relocatable loader text. The last byte of each record is the check-sum of the record (the folded arithmetic sum of the data bytes in the record). Paper tape requires four additional bytes to specify a binary text record. The first byte, a line feed (X '8A'), identifies the start of a record. The succeeding two bytes specify the number of bytes in the record. At the end of the record there are the terminating bytes, a carriage return (X'8D') followed by three unpunched frames (X'00'). Refer to Section 8-6 for a description of the binary record format for the various device drivers.

8-5.2.4 BASIC Relocating Loader

The RElocating LOADER-Basic (RELOADB) is used to load relocatable programs which were assembled using the SYM-I assembler.

This Basic Loader provides a convenient operating environment for loading and executing medium sized programs on a Raytheon 706 with 4K words of core and no system library facilities.

Figure 8-11 illustrates the operation of RELOADB. Programs which are in relocatable loader text format on paper tape are input from the Binary INput (BIN) device. This device may be assigned to either the teletype or the high speed paper tape reader.

All control of the Basic Relocating Loader is exercised by using directives through the X-RAY EXEC-BASIC which in turn calls RELOADB.

Table 8-4. Relocatable Loader Text Codes and Formats

Hex	Mnemonic	Data Format Following Hex Code
00	Not used	Ignored by the loader.
01	LIBR	Eight bytes of symbolic library name, output by the LIBR pseudo op.
02	EXTN	Eight bytes of symbolic name, followed by a word containing the end address of an external string. See External Strings, below.
03	SMB	A word containing a 15-bit relocatable address. The loader will construct and store an SMB instruction for the memory bank containing the specified address.
04	ILOC	A word containing the number of locations by which the location counter is to be incremented, output by the RES pseudo op.
05	NTRY	Eight bytes of symbolic name, followed by a word which contains the value (i.e., address) assigned to that symbol. If the sign bit of this word is set, the address is relocatable.
06	END	A word containing the starting execution address. The address is always relocatable. If the word is zero, the loader interprets this as no execution address specified. Code 6 marks the end of a program module.
07	EXTN 15	Eight bytes of symbolic name followed by a word containing the end of a 15-bit external string. See External Strings, below.
08	NBLK	Eight bytes of symbolic name. Output by BLK pseudo op.
09	SIZE	A word containing the program size in words.
0A	SIZEW	A word containing the program size in words. This code specifies that the program must be loaded on one word page.
0B	SIZEW	A word containing the program size in words. This code specifies that the program must be loaded on one byte page.
0C	DBLOCF	Eight bytes of symbolic name followed by one word containing the block size. This code declares the data block.
0D	DBLOCR	Eight bytes of symbolic name followed by one program word containing a relative reference in the block.
8n	RELW11	n + 1 16-bit program words, whose 11-bit address fields are 11-bit relocatable word addresses.
9n	RELW15	n + 1 16-bit program words, which contains 15-bit relocatable word addresses.

Table 8-4. Relocatable Loader Text Codes and Formats (Cont)

Hex	Mnemonic	Data Format Following Hex Code
An	RELB16	n + 1 16-bit program words which contain 16-bit relocatable byte addresses.
Bn	RELB1	n + 1 16-bit program words whose 11-bit address fields are 11-bit relocatable byte addresses.
Cn	ABSO	n + 1 16-bit absolute program words.
Dn	DFILL	A relative (relo) address followed by n + 1 words of absolute data.

RELOADB has the ability to load multiple program modules and print an entry names table.

All programs loaded by this loader are relocated into the upper word page in memory. The X-RAY EXEC-BASIC and the Basic Relocating Loader (RELOADB) all reside in the lower word page in memory. Since the loader's entry names table (ENT) uses the space remaining in the lower word page, the entire upper word page is available for programs to be loaded and the required data areas.

If more data area is needed, the space in the lower word page occupied by the loader can be used; however, if the loader is undisturbed it is re-executable and need not be reloaded between jobs.

Although this loader is designed to accept SYM I assembler output it will accept programs which have been written in SYM I but assembled on the SYM II assembler.

It will also load SYM II programs with two restrictions on SYM II assemblies. First, the

pseudo-ops DFLL, and LABL must not appear. Second, the program must have at least one reference to a relocatable address. This reference will ensure that the SYM II assembler will output 11-bit addresses for external strings rather than 15-bit addresses (the BASIC loader cannot process 15-bit external strings).

8-5.2.5 STANDARD Relocating Loader

The RElocating LOADER-STANDARD (RELOADS) is used to load relocatable loader text produced by either the SYM I or SYM II assemblers (Figure 8-12) from any medium, except disc, containing programs in relocatable object text. The loader first loads a complete program in a primary or nonselective phase and then enters a selective library phase in which only programs referenced during the primary phase are loaded from the library.

The Relocating Loader performs memory allocations for any size computer, arranging programs in core for optimal fit on word or byte pages as

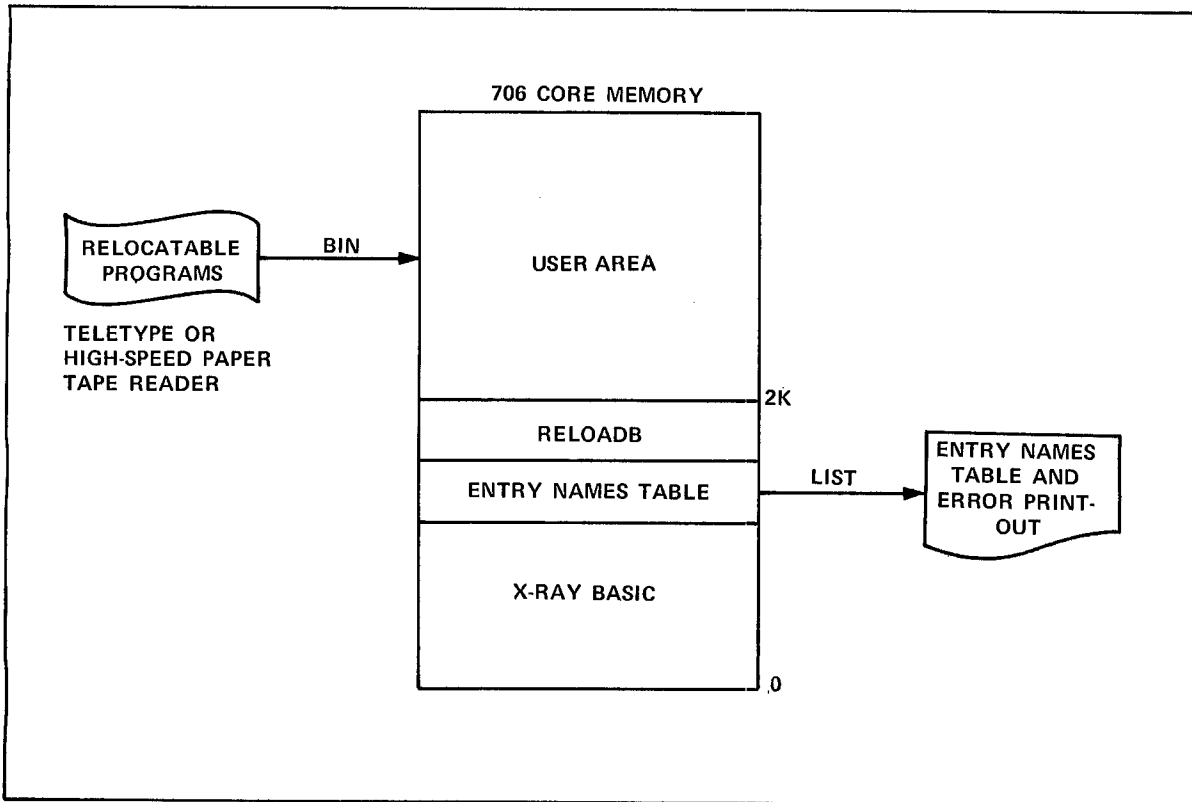


Figure 8-11. BASIC System Loader Map

required. All programs written in SYM I, SYM II, and FORTRAN IV source languages may be loaded by this loader. FORTRAN Unlabeled Common is allocated in the space occupied by the loader. If the space occupied by the loader is not used during object program execution, the loader will be intact and re-executable.

8-5-2.6 Disc Relocating Loader

The Disc Relocating Loader serves the 706 Real-Time Operating System (Figure 8-13). The loader is used to load relocatable object text produced by either the SYM I or SYM II assembly language processors. Input is from any medium containing programs in relocatable object text. The loader first loads a complete program in a primary or non-selective phase and then enters a selective

library phase in which only programs referenced during the primary phase are loaded from the library.

The Disc Relocating Loader performs memory allocations for any size computer, arranging programs in core for optimal fit on word or byte pages as required. All programs written in SYM I, SYM II, and FORTRAN IV source languages may be loaded by this loader. FORTRAN Unlabeled Common is allocated in the space occupied by the loader. If the space occupied by the loader is not used during object program execution, the loader will be intact and re-executable.

8-5.2.7 Linking Absoluter

The Linking Absoluter (LINKABS) is a program

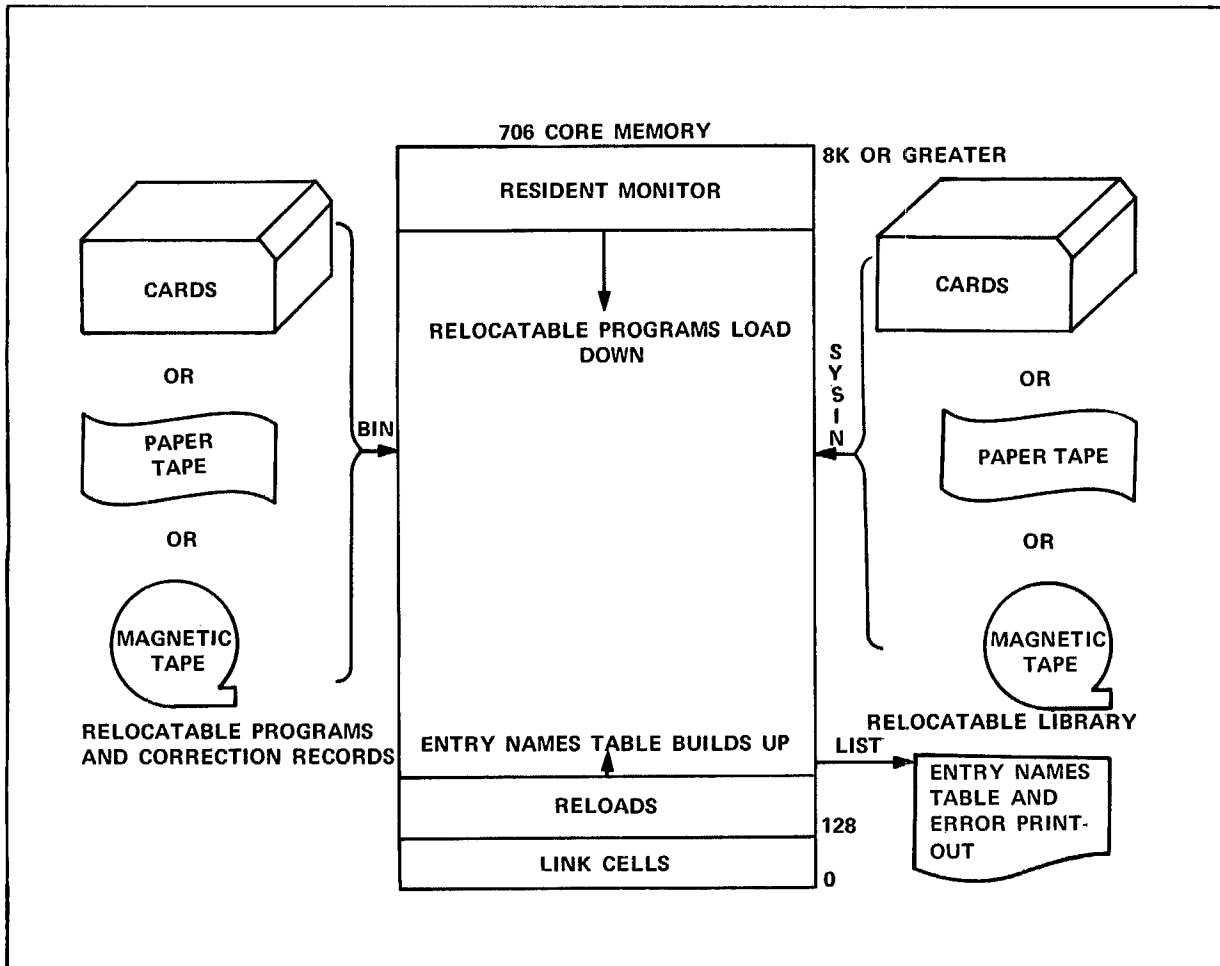


Figure 8-12. STANDARD and MTOS System Loader Map

that converts relocatable loader text into absolute program text, which can be loaded by the absolute loader or by the bootstrappable Initial Loader (INITLOAD).

Using the Linking Absoluter it is possible to develop and completely check out an application program as a series of independent, relocatable

sections then link these sections to form one absolute object program.

Linked absolute programs require less space in core memory than relocatable programs but relocatable programs are easier to code and check out. The Linking Absoluter is a means to obtain the advantages of both relocatable and absolute programs.

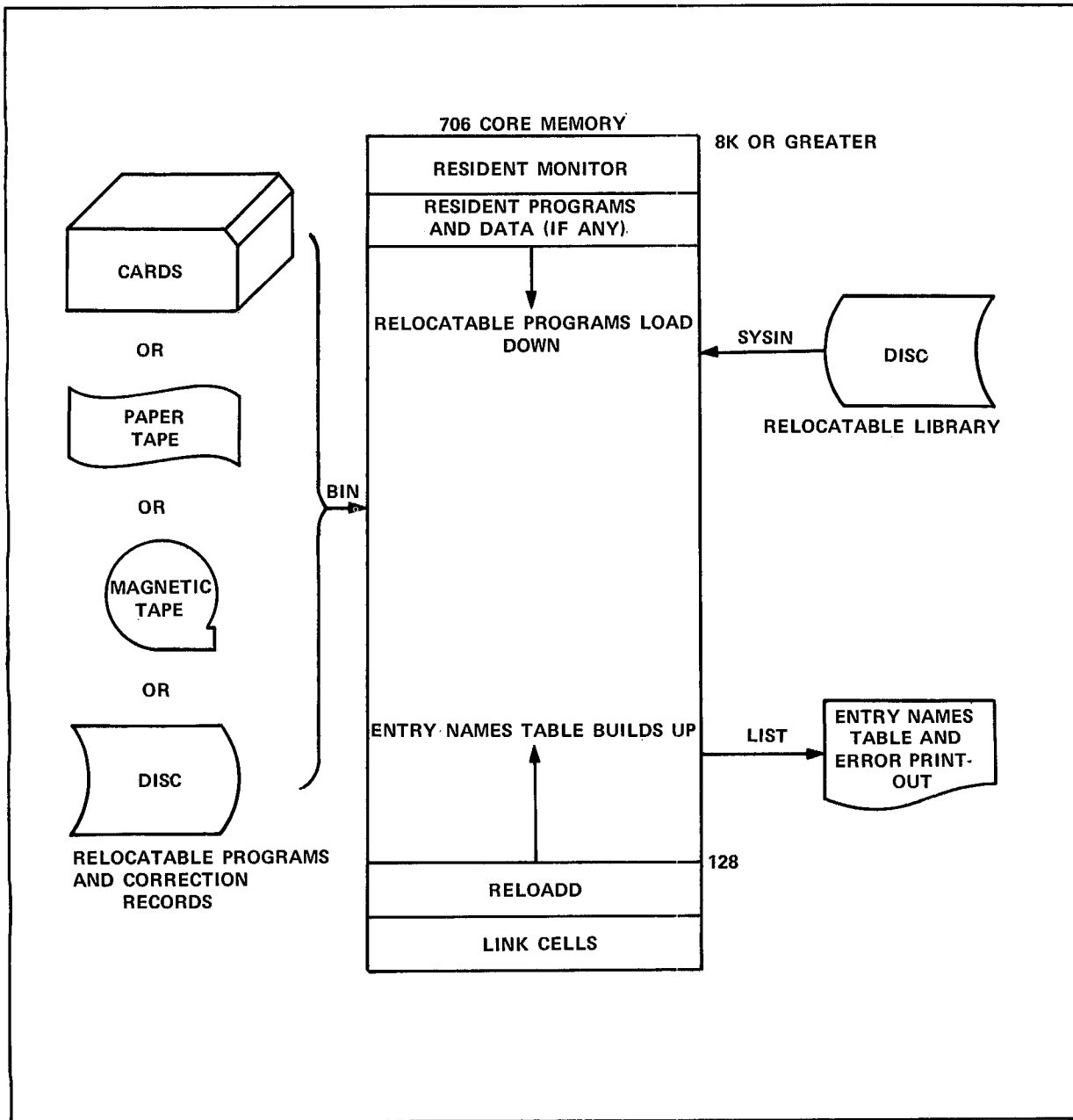


Figure 8-13. EXTENDED and ADVANCED System Loader Map

8-6. INPUT/OUTPUT ROUTINES (DRIVERS)

Since the 706 is seldomly used without doing input/output operations, Raytheon Computer has created and standardized input/output subroutines (drivers) that control the common peripheral equipment. This eliminates the need for the programmer to concern himself with the machine language detail of I/O selection, data transfer, interrupt scheduling, and end-action. He only has to code three macros OPEN, DOIO, and STAT to communicate with any standard 706 device.

The resident I/O monitor system processes all three macros and calls upon drivers to handle the input/output function designated by the program. Table 5-8 lists all of the devices that currently have drivers associated with them.

Section 5-4 of this manual describes the operation of the Input/Output software while the remainder of this section describes the individual I/O drivers.

8-6.1 Teletype, High-Speed Paper Tape I/O Driver

8-6.1.1 Purpose

The Teletype High-Speed Paper Tape Driver services I/O operations on the Console Teletype, the High Speed Paper Tape Reader, and the High Speed Paper Tape Punch. The driver is fully re-enterable and can service simultaneous I/O on all three devices.

This driver can perform the following functions:

1. Read a record — Teletype and High Speed Reader
2. Write a record — Teletype and High Speed Punch
3. Write a file mark — Teletype and High Speed Punch
4. Punch leader — Teletype and High Speed Punch

8-6.1.2 Record Formats

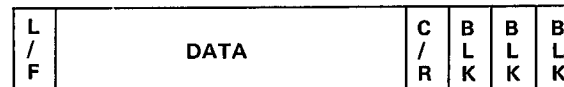
This driver reads and writes formatted or unformatted records. Two formats, alpha and binary, are used for formatted records. The binary format

includes a record size, in bytes, at the beginning of the record.

The binary record format is shown in Figure 8-14a, and the alpha format in Figure 8-14b.



a) Binary Format



b) Alpha Format

Figure 8-14. Teletype Driver Record Formats

As shown in the diagrams, a line feed character (L/F, 8A₁₆) will be the first character in each record. The binary record follows with two characters of byte count. This is the total number of data bytes contained in the binary record.

The alpha record does not have a byte count. The area shown as data in the diagrams is the information to be moved during the I/O operation. Each record closes with a carriage return character (C/R, 8D₁₆). There are three blank characters (BLK, A0₁₆) after each record on tape. These will allow an operator to position the tape between records. The blanks are not necessary when input is from a keyboard.

When the requested operation is alpha mode input, the first C/R following the initial L/F terminates the record regardless of how many input words had been requested. This means there can be no C/R

characters within an alpha input record. When the requested number of words have been input, the device will be disconnected even if there has been no C/R. The number of bytes in a binary record is a similar delimiter. If a user requests 100 words input and the binary record contains 25, only 25 words would be input. If a record is larger than the number of characters requested to input, the remainder is skipped before the input is terminated.

For output operations, bit 0 of FIOT word 2 determines whether the record will be output as a binary or alpha record. For input operations the driver determines whether the incoming record is alpha or binary and sets bit 0 of word 2 accordingly.

For alpha mode input, two control characters are recognized. These are primarily to permit easy operator correction of mis-typed teletype input records; however, they function for paper tape records as well. The first character is RUBOUT (FF₁₆). Instead of storing this character, the driver backs up the buffer pointer one character for each RUBOUT character, thereby permitting mis-typed characters to be retyped. For example, if the record

```
:Qu, AKPH
```

were typed, when the operator intended to type

```
:QU, ALPH
```

he may correct it by continuing

```
(RUBOUT) (RUBOUT) RUBOUT) LPH
```

Since the RUBOUT doesn't print, the resulting record will appear on the printout as

```
:QU, AKPHLPH
```

However, it will be corrected internally. The second character is the horizontal arrow (←). This will delete the entire record, which should then be restarted, including the initial line feed. Both of these characters must be input before the closing carriage return. Once the carriage return has been

input, the record is processed and no correction is possible.

When reading formatted records, if an ASCII Bell (X'87') character is encountered between records, prior to the Line Feed character, the I/O operation is terminated and end-of-file status (bit 10 true) is returned in word 3 of the FIOT. The driver accepts a WEOF call and outputs this character to the device.

Unformatted records, some time referred to as special format, contain no added formatting characters. The contents of the number of words specified are transferred to or from the external medium. Unformatted records are specified by setting bit 0 of FIOT word 6 true.

Notice that no positioning of paper tape input records occur for unformatted record; data transferring begins with the first frame of tape read and continues until the requested number of frames has been read.

8-6.1.3 Console Teletype

The standard Teletype Model 33/35 (Automatic Send-Receive) can be used to type in or print out information at a rate of up to ten characters per second, or to read in or punch out perforated paper tape at a ten characters per second.

The teletype reader is not controllable by command from the computer. It must be turned on and off manually. If the teletype keyboard is selected for input, the reader may be turned on and allowed to drive the keyboard. Similarly, when the reader is selected, the keyboard may be used instead, but what is typed will not be printed.

The teletype unit can perform the following functions through the driver:

1. Read a Record from Keyboard — Function Code B

The keyboard is enabled, the keyboard light lights, and characters read are printed.

2. Read a Record From Reader – Function Code 9

The reader light lights and characters read are not typed. However, the keyboard is enabled

3. Write a Record – Function Code E

A record is output to the receive section of the teletype and will print if the ASR 35 teletype mode switch is in the PRINT position, punched if the switch is in the PUNCH position, and printed and punched if the switch is in the PRINT and PUNCH position. The ASR 33 record is always printed and will be printed and punched if the punch is turned on. These modes must be selected manually.

4. Write a File Mark – Call WEOF

The ACSII Bell character is output followed by approximately 60 blank frames. If the punch is on, it will appear as leader. Bit 0 of FIOT word 6 must be false (not special format).

5. Punch Leader – Function Code E

If the buffer address is given as X'7FFF', no data is output; instead 60 blank frames are output.

8-6.1.4 High Speed Paper Tape Reader

The 706 paper tape reader is an unidirectional photoelectric reader capable of reading 300 characters per second continuously; or up to 100 characters per second intermittently.

The reader has an adjustable tape guide which adapts it to read 8-channel, 7-channel, or 5-channel tapes.

Reader power is controlled by an ON/OFF switch located on the reader front panel. The RUN/LOAD switch is an integral part of the tape guide mechanism.

The high speed reader can perform the following function through the driver.

1. Read a Record – Function Codes 8 or 9.

A formatted or unformatted record is read from the reader device.

8-6.1.5 High Speed Paper Tape Punch

The 706 paper tape punch is capable of punching up to 110 characters per second. Each character consists of eight data bits plus a sprocket hole. A hole in a data bit position represents a *one*; no hole represents a *zero*.

The controller contains a one-character buffer register that accepts data from the 706 in the form of eight-bit binary numbers.

Data is transferred to the controller under control of the central processor. There are no punch controls or indicators located on the computer console. Only a manual tape feed push-button is located on the punch assembly front panel. When the TAPE FEED push-button is pressed, power is applied to the punch mechanism whenever a character is transferred from the computer. If the punch is off, there is a one second delay which allows the punch to come up to speed before data is transferred to the punch. If the punch is already on, there is no delay. Once started, the punch remains on until a command from the computer turns it off.

The punch can perform the following functions through the driver.

1. Write a Record – Function Code E

A formatted or unformatted record is output to the paper tape punch. This record will be identical to the same record punched on the teletype punch.

2. Write a File Mark – Call WEOF

The WEOF call will cause a file mark (X'87') character to be punched followed by approximately 60 blank frames. Bit 0 of FIOT word 6 must be false (not special format).

3. Punch Leader — Function Code E

If the buffer address is X'7FFF', 60 blank frames are punched as leader.

After any punch operation, the punch motor remains on which will eliminate the one second delay when more than one record is to be output. When the user is entirely finished using the punch, he should turn off the motor by executing the instruction

DOT PCHU, 3

When PCHU is the unit code for the punch, normally X'C'. This instruction may be given even when the device might not be the punch. If another user is using the punch, in a real-time system, the motor will be turned back on when the next character is transferred and the turn-off instruction is harmless.

8-6.2 Teletype Multiplexer Driver

8-6.2.1 Purpose

The Teletype Multiplexer Driver performs data input-output operations to a teletype system configuration consisting of a maximum of sixteen teletype units all of which are under a channel scanning scheme in the controller device.

8-6.2.2 Usage

The OPEN, DOIO, and STAT calls are used as described in Section 5-4. The teletype description given in the I/O monitor with regard to the alpha format is applicable to this driver. Each teletype unit is physically connected to one of sixteen channels, numbered from 0 through 15 decimal. The user must store a channel number in the least significant hexadecimal digit position in FIOT (5) before executing a DOIO call.

8-6.2.3 Message

The message TPLX is given when the driver detects a channel number which is greater than the total number of channels in the system, as denoted by the literal NMXT. This error may be due to an

incorrect I/O mode (bit 0) of the controller word, which would cause the driver to extract an incorrect channel number from the controller word. This error may be due also to an incorrect channel number being received from the controller.

This message is given also when the driver fetches an FIOT address from the FIOT table (FTBL) which is not a currently active FIOT address. The pointer to the FIOT addresses in the table is a channel number; therefore, an incorrect channel number would cause this error.

8-6.2.4 Error Conditions

If the user requests an I/O operation on a channel which does not exist, the driver will set the no-channel bit 9 and the error bit 15 of status word FIOT (3). For the case of an unwanted message abort, the driver will set the abort bit 8 and error bit 15 of status word FIOT (3) and terminate the I/O operation.

If a data late error condition is detected, the driver will set the abort bit 14 and error bit 15 of status word FIOT (3) at end-of-record time.

The above error conditions may be detected by including an ERR argument in the STAT call.

8-6.2.5 Restrictions

The I/O monitor is structured such that a single logical unit number is associated with the teletype multiplexer, rather than a logical unit for each teletype. Accordingly FORTRAN READ and WRITE statements cannot be used to specify a particular teletype. FORTRAN programs which are to communicate with the teletype multiplexer should contain subroutines which communicate directly with the driver.

8-6.3 Card Reader Driver

8-6.3.1 Purpose

The card reader driver provides a method for the Raytheon 706 IOS user to input data from the card reader, Soroban Model ERD-1 (400 and 1000 cpm).

8-6.3.2 Usage

The OPEN, DOIO, and STAT calls are used as described in Section 5-4. The card reader driver performs the read function only, function code X'B' or X'9'. Any other call is ignored. The driver does not alter the device status or the FIOT for calls ignored.

Information is read from cards in one of three data formats:

1. Special format – Figure 8-15a.
2. Binary – Figure 8-15b.
3. Alpha (Hollerith) – Figure 8-15c.

The information is processed in one of two modes:

1. Binary
2. Alpha

If bit position 0 of word 6 of the FIOT is one the information is read as special format (binary) mode.

If bit position 0 of word 6 of the FIOT is zero, the driver will specify the mode (binary or alpha) after the first column is read. Bit position 0 of the FIOT (2) is set to one if the input is in binary format, or to zero if the input is in alpha format.

For any call upon this driver, if a 2-7-8-9 punch is read from the first column of the first card, the driver sets bit position 11 (end-of-file) of the status word and terminates the read operation. The driver stores the status word in word 3 of the FIOT: The card reader reads the entire card.

8-6.3.3 Special Data Format

A maximum of 120, eight bit bytes of data (80 columns) is read from a card. If the number of bytes to be read is greater than 120, eighty columns of data are read from each card except possibly the last. The column data is read as binary information and is packed to form 16-bit words. The driver only reads the number of words specified by the user. The card reader always reads an entire card.

8-6.3.4 Binary Data Format

If bit position 0 of the FIOT is OFF and the first column read contains a line feed character (12-2-4 punch), the information is read in the binary mode. The first two columns contain the binary record length (byte count) and the line feed character. They are not stored into the users buffer. See Figure 8-15b for the organization of the first two columns.

There are a maximum of 102 bytes of data (68 columns) read from a card. If the number of bytes to be read is greater than 102, sixty-eight columns of data are read from each card, except possibly the last. The 12-bit column information is packed to form 16-bit words.

If the byte count is less than the number of bytes specified by the user, input will be terminated when the number of bytes read is equal to the byte count. If the number of words specified by the user is less than the record length, input to the user's buffer is terminated when the number of words requested have been read. The driver always stops reading cards after the last card of the binary record has been read. The card reader always reads an entire card.

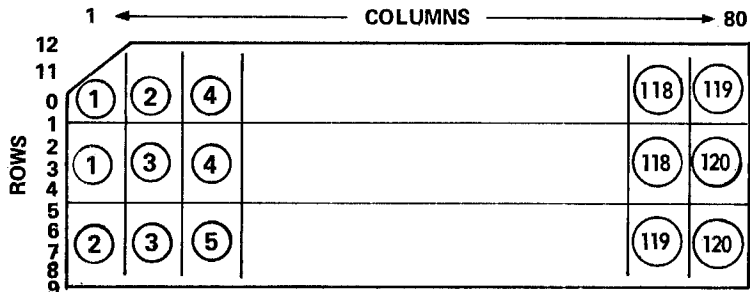
8-6.3.5 Hollerith (Alpha) Data Format

If bit position 0 of word 6 of the FIOT is OFF and the first column read does not contain a line feed character, the driver reads Hollerith information from the card.

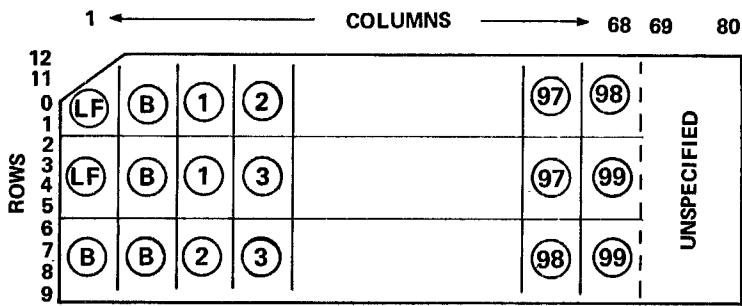
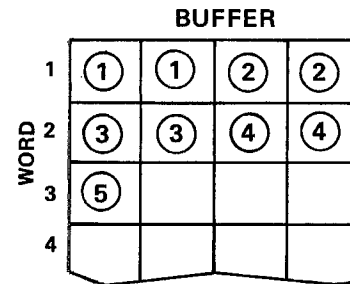
Each card column (Hollerith character) is converted to an ASCII character. Two ASCII characters will be stored as one 16-bit word. A maximum of 80 columns (40 16-bit words) of information may be read by the driver per call, although the entire card is always read by the card reader.

8-6.3.6 Messages

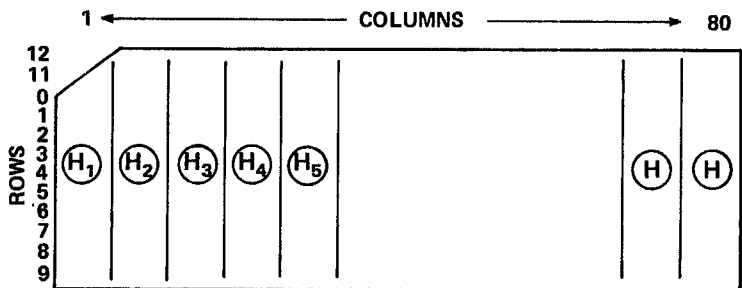
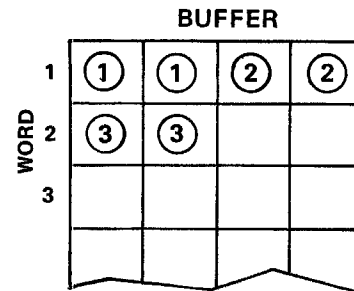
For each call the driver checks for device not ready and trouble conditions. If any of these conditions exists, the operator is notified by the message, on the teletype, CR and two bells. These checks are



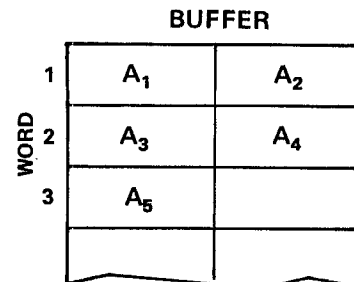
A) SPECIAL FORMAT



B) BINARY FORMAT



C) ALPHA (HOLLERITH) FORMAT



(n) Specifies The 4 Bit Digit of Data.

(H_n) Specifies a Hollerith Character.

(LF) Specifies a Line Feed Byte (Punches in 12-2-4).

(A_n) Specifies an ASCII Character.

(B) Specifies the Byte Count.

Figure 8-15. Card Reader Driver Record Formats

performed at the time the DOIO is executed.

NOTE: The driver always returns a status word to the FIOT with the first three bit positions ON. These bits should be ignored by the user.

If the driver is required to read more than one card per call, the buffer working address is stored in word 4 of the FIOT after each card is processed.

Word 3 and 7 of the FIOT are used as temporary storage locations by the driver.

8-6.3.7 Space and Timing

This routine uses 253 decimal words and also shares numerical constants with the I/O monitor.

To process an interrupt for:

- | | | |
|--------------|-----------|--|
| 1. Hollerith | — Maximum | 98 cycles
(non-blank 1st
column) |
| | Minimum | 34 cycles
(blank column) |
| 2. Binary | — Maximum | 64 cycles |
| | Minimum | 41 cycles |

The I/O monitor uses an additional 70 cycles to service an interrupt.

8-6.4 Card Punch Driver

8-6.4.1 Purpose

The Card Punch Driver program provides the Raytheon 703 user with a method for outputting data to the Honeywell Model 214-1 Card Punch, using existing 706 input/output software.

8-6.4.2 Usage

The driver is designed to interact with the 706 IOS. It uses the normal macros in I/O monitor and several of the IOS subroutines.

The following I/O functions are recognized by the program:

1. Write (3 formats: Hollerith, Binary, Special)
2. Write End-of-File
3. Punch Leader

All other functions are ignored by the driver.

8-6.4.3 Method

The Card Punch Driver program is called by the I/O Monitor macro, DOIO. On each call the driver checks device status (device not ready, hopper empty, stacker full, etc.) and notifies the user if any of these conditions exist by typing CP followed by two bells on the system teleprinter. When the card punch becomes available for use, its interrupt level is enabled, the driver starts the punch, the program returns to the user and waits for controller interrupts.

Cards are punched at the rate of two columns per interrupt. After each interrupt, the driver returns to I/O Monitor and is not active until the next interrupt is received.

Upon receiving an interrupt from the controller, the driver responds with two columns of data within one millisecond to prevent response errors. If at least one hole is to be punched for each pair of columns, the time between interrupts will be 12.5 milliseconds.

For each data output command to the card punch, a 12-bit data word (right justified in the accumulator) is sent to one of the two buffer registers. Two such commands are required per interrupt. When the buffers have been loaded, two columns are punched and the driver returns to I/O Monitor. Bits 0 through 3 of the accumulator are ignored during the data output.

8-6.4.4 Write Special Format

If bit 0 of word 6 in the user's FIOT is set to one, the data is output to the card punch in special

format. The driver extracts 12-bit data words from the user's buffer on command and re-packs the data for output in the format shown in Figure 8-15a. A maximum of sixty 16-bit words (80 columns) may be punched per card. If record length exceeds 60 words, only the last card punched may contain less than 60 words.

8-6.4.5 Write Binary

If bit 0 of word 6 of the user's FIOT is reset (zero), the driver then examines bit 0 of word 1 of the FIOT. A one in this bit position tells the driver to punch data in the binary format.

In the first 2 columns of the first card in the record, the driver punches a *line feed* (12-4-2 punch) and the number of bytes to be output. A maximum of 102 bytes may be punched per card. If the record exceeds 102 bytes, only the last card punched may contain less than 102 bytes. The driver extracts 16-bit data words from the user's buffer and re-packs them into 12-bit words (right justified) for output as shown in Figure 8-15b.

8-6.4.6 Write Hollerith

If bit 0, word 1 of the FIOT is reset (zero), data will be output by the driver in Hollerith format. In this format data is extracted from the user's storage by 8-bit bytes and converted to 12-bit right-justified Hollerith words for output to the card punch. The data must be packed 2 bytes per word in the storage area and each byte must represent an ASCII character to obtain a meaningful output. One Hollerith character (one column) is output for each ASCII character (byte) in the record (Figure 8-15c). A maximum of 80 columns (forty 16-bit words) are punched by the driver per call.

8-6.4.7 Write End-of-File

If the sign bit of M. DR is on, X'870' is output to the card punch and the 2-7-8-9 punch appears in the first column of the card in process. No other information appears on the card. Only one EOF is punched per call.

8-6.4.8 Punch Leader

If the buffer address in word 1 of the user's FIOT is X'7FFF', the driver feeds one blank card and offsets it in the card stacker.

8-6.5 Line Printer Driver

8-6.5.1 Purpose

The line printer driver, a part of IOS, transfers data to the line printer and controls paper movement. It obviates the need for the user to understand the detailed line printer programming procedure.

8-6.5.2 Usage

IOS calls, OPEN, DOIO, and STAT, may be used to initialize, initiate, and await the completion of printer operations.

Not all of the OPEN arguments are relevant. The OPER and MODE arguments are ignored by the driver. However, the FIOT, BUF, WC and UNIT arguments are required to initialize the FIOT prior to initiating a printer operation.

Each of the line printer functions described below is initiated by a DOIO call of the form:

```
DOIO  FIOT,  BUF,  WC
```

The STAT statement should be of the form:

```
STAT  FIOT
```

The ERR argument, normally associated with the STAT statement, may be omitted since there is no error associated with the line printer.

8-6.5.3 Driver Functions

The functions performed by the line printer driver are summarized below:

1. Move paper to top of form.
2. Move paper one space.

3. Print a line and move paper one space.
4. Print a line.
5. No printer operation.

The function performed by the driver upon execution of a DOIO depends on the content of location BUF and the special format bit, which is contained in bit position 0 of FIOT (6).

Function 1, move paper to top of form, is performed whenever BUF contains line feed codes, 8A8A₁₆. Top of form is channel 3 on the line printer carriage control tape.

Function 2, move paper one line, is performed whenever BUF contains line feed and carriage return codes, 8A8D₁₆.

Function 3, print and move paper one line, is performed whenever FIOT (6) bit 0 is equal to 0 and location BUF does not contain 8A8A₁₆, 8A8D₁₆ or 8D8D₁₆. (See Function 5). In this case the content of the user's buffer is printed and paper advanced to the next print line. The driver expects two character codes per word in the user's buffer. The character corresponding to the code in the left byte of location BUF will appear at the left most print position.

Function 4, print a line, is performed whenever FIOT (6) bit 0 is equal to 1 and BUF does not contain 8A8A₁₆, 8A8D₁₆, 8D8D₁₆. In this case the content of the user's buffer is printed. Paper movement is controlled by the right byte of FIOT (4). In order to slew to any particular channel, the right byte of FIOT (4) must contain X'10' + channel number. For example: slew to channel 5 would be X'15'.

Function 5, no printer operation, occurs when BUF contains carriage return codes, 8D8D₁₆. This function is provided for programs that overprint on the system list device (logical unit 3) and further, makes possible the assignment of the line printer or the teletype to logical unit 3.

8-6.5.4 Error Conditions

In the event the line printer is off-line when a call to the driver occurs, LP is typed on the teleprinter.

At this point the operator should ready the printer and toggle sense switch 3, signaling IOS to continue.

8-6.5.5 Timing

The driver requires 1.5 ms to fill the printer buffer. The printer requires 45 ms to print a line and 15 ms to space paper 1 line (1000 LPM printer). The interrupt system is masked by the driver for a period of 22 microseconds.

8-6.6 Disc and DMA Magnetic Tape Driver

8-6.6.1 Purpose

The disc and DMA Magnetic Tape Driver transfers information between the 706 computer and the magnetic tape or disc units. This information is transferred using the 706 direct memory access capability. Therefore, all information is binary and requires no formatting by the driver. Certain similarities of the disc and magnetic tape are exploited in the sharing of a large portion of this driver with both devices. Those portions which are not common are assembled into the monitor only for those devices actually present.

8-6.6.2 Usage

Section 5-4 describes the IOS calls and the general FIOT configuration used to operate this driver. Certain uses of the FIOT are specific to this driver and are described below. This driver can perform the following functions:

- | | |
|-----------------------|--------------------------|
| 1. Read a record | — Magnetic tape and Disc |
| 2. Write a record | — Magnetic Tape and Disc |
| 3. Verify a record | — Disc only |
| 4. Write a file mark | — Magnetic tape only |
| 5. Backspace a record | — Magnetic tape only |
| 6. Seek a file mark | — Magnetic tape only |

- 7. Rewind — Magnetic tape only
- 8. Erase forward (write skip) — Magnetic tape only

This driver cannot punch leader, and the leader call is ignored.

8-6.6.3 Error Recovery

All magnetic tape and disc records are written with cyclic redundancy check characters. If an error is detected in this character while reading or by the read after write feature of the magnetic tape, or if the magnetic tape detects a character parity error, the driver will automatically attempt recovery of the error by retransmitting the offending record, up to a maximum of 16 times. If after 16 attempts the record has not been successfully transmitted, the operation is terminated and error status is returned. The tape is positioned past the offending record. The tape retry procedure may be by-passed by stuffing bits 0-6 of FIOT (2) with a zero after DOIO is initiated.

The following section details considerations applicable individually to the two devices serviced by this driver.

8-6.6.4 Disc General

The disc is a high speed random access magnetic storage device upon which data is stored in individually addressable sectors, each holding forty-seven 16-bit computer words. The disc is organized into 8,192 sectors for a total storage capacity of 385,034 words. A record must begin at the first word of any sector, and may be of any length up to 16,383 words, since the disc hardware automatically sequences to the next sector when the current section has been transmitted.

Since the disc is used in the Real-Time Operating System and Multi-Programming System as the permanent storage for the resident system, a special disc addressing structure is super-imposed by the driver, which allows the disc to be addressed as four relatively addressed, relocatable logical files of limited size. This feature relieves the programmer from the responsibility of storage

allocation on the disc which can be allocated dynamically at execution time as required.

Since certain system programs, as well as some users, must directly address specific sectors of the actual disc, an additional addressing mode is supplied which permits this. This circumvents the system protection features of the driver, and must be used knowledgeably.

The disc can perform the following functions through the driver:

1. **Read a record — function code 9 or B**
A record is read from the disc.
2. **Write a record — function code E16**
A record is written on the disc.
3. **Verify a record — function code 7.**

No data is transmitted to or from memory, but the cyclic redundancy check characters are computed and checked for as many sectors as are specified by the record length. An unrecoverable error results in error status being returned.

8-6.6.5 Disc FIOT

The disc FIOT is of standard form, except as follows. Bit 0 of word 2 is not used, since no formatting is done for disc records. The special format bit, bit 0 of word 6 must be a 0. If it is a 1, the requested operation is not performed. Word 5 provides sector address for the disc.

FIOT (5) is divided into three fields as follows:

- Field 1 Bit 0 is a one-bit field designating the format of fields 2 and 3. Also, if bit 0 is a one, field 2 is ignored.
- Field 2 Bits 1 and 2 form a two-bit field designating which disc file is being used for the I/O request. When bit 0 is zero, field 2 is used as a binary count referring to files 0 through 3.

Field 3 Field 3 consists of bits 3 through 15 inclusive. If bit 0 is a zero, field 3 is a sector address to be converted to track and sector by the disc driver. Field 3 is added to the file base sector and checked to be less than or equal to the top limit. The new sector address is then converted to track and sector and used by the driver. Also, the next available sector for the user is calculated by adding to the initial sector the number of sectors involved in the I/O request, and the result is stored in FIOT (5).

If bit 0 is a 1, the addressing is said to be in track and sector mode. In this mode, the right byte is a track address between 0 and 127. The left byte (bits 1–7) is a track address between 0 through 63. In this mode all addresses are absolute disc addresses, and the system protection features are entirely circumvented.

8-6.6.6 Disc Files

Disc File allocation is performed by the driver on the basis of pointers stored in an area of the resident monitor called the Disc Vector (DVEC). Cell X'74' of the System Linkage Table points to the first cell of the Disc Vector. The Disc Vector is organized as follows:

DVEC+0	First sector of file 0
+1	First sector of file 1
+2	First sector of file 2
+3	First sector of file 3
+4	Number of sectors in file 0
+5	Number of sectors in file 1
+6	Number of sectors in file 2
+7	Number of sectors in file 3
+8	
+9	Various system disc pointers
+10	
+11	First sector after system

When a user accesses a sector in a file, the driver constructs the actual address as follows:

1. The total number of sectors to be transmitted is computed and added to the specified starting sector address. This result is compared with the size of the file. If it is larger than the file, the requested operation is not performed. Instead, if it is a read operation, end-of-file status (bit 11 of the status response word) is returned. If it is a write operation, protected track and error status (bits 13 and 15) are returned.
2. If the referenced address lies within the file, the actual disc starting address is computed by adding to the requested sector address the starting address of the file (from DVEC, words 0–3) plus the first sector after the system (DVEC, word 11). Thus, the first sector of each file is specified relative to the end of the system. For example, suppose the starting address of file 0 were given as 50 and the size of file 0 were given as 200 sectors; then sectors 0 to 199 would be available to the user of file 0. Writing in sector 0 would cause the driver to write in the 50th sector following the system area of the disc. That is, if the system ended at actual disc sector 199, writing in sector 0 of file 0 would write in actual disc sector 250, and the file would extend on the actual disc through sector 399.

Disc file pointers are normally set by the Real-Time X-RAY executive's :DF directive. Operating programs may modify the disc file pointers if necessary. At each new job (whenever NEXT is listed by X-RAY), they will be reset to the configuration previously specified using the :DF directive.

Certain system processors use the disc files, whenever the disc is used as intermediate storage: FORTRAN IV and SYM II use file 1 as intermediate text storage. SYM II uses file 0 as binary output for assembly and go, and compile and go operations. When SYM II exits, it sets the size of file 0 to as many sectors of binary text as it has actually output to the file. This will not be reset by the system until Real-Time X-RAY resets it at Next Job time.

Disc file pointers may be reset by any operating program. However, this should be done cautiously to avoid interfering with other users. The user may modify cells 0–7 of the disc vector. Cells 8–11 must not be disturbed.

8-6.6.7 Disc Status Response

At the conclusion of disc I/O operation, the status is stored in word 3 of the FIOT. The bits of this word have the following meanings:

Bit Set	Use
0	Controller or device not ready
1	Device not ready
11	End of logical file
13	Write command issued to a protected track or sector outside of logical file.
14	Rate error
15	Rate or CRC error or bit 13 true.

If the status is returned with bit 15 true, an unrecoverable error has occurred.

8-6.6.8 DMA Magnetic Tape – General

The DMA magnetic tape driver will handle all magnetic tape operations for the 9-track tape unit, and for the 7-track unit in the 3 character binary mode. It will also operate the 7-track tape in the 2-character binary and 2-character alpha modes. However, it does no formatting of the data buffers and, if any formatting is required, it must be performed outside of the I/O system.

The DMA magnetic tape can perform the following functions through the driver:

1. **Read a Record – Function Code 9 or B**
One record is read from the magnetic tape unit.

2. **Write a Record – Function Code E16**
One record is written on the magnetic tape.
3. **Read a Gapless Tape (Chained I/O) – Function Code 1**
The driver initiates reading of gapless records. This operation can be performed only when the DMA record chaining feature is included in the magnetic tape controller.
4. **Write a Gapless Tape – Function Code 2**
The driver initiates writing of gapless records. This operation can be performed only when the DMA record-chaining feature is included in the magnetic tape controller.
5. **Write a File Mark – WEOF Call**
A hardware detectable file mark is written on the tape.
6. **Backspace a Record – BKSP Call**
If the previous command was a read, the tape backspaces one record. If the previous command was a write, the tape erases forward 1/2 inch, then backspaces one record.
7. **Seek A File Mark – SEOF Call**
The tape advances until a file mark is detected, then halts just past the file mark.
8. **Rewind – REWD Call**
The specified tape deck is rewound. This is an off-line operation for the magnetic tape controller. Other decks are available for use while the specified deck is rewinding.
9. **Erase Forward (Write-Skip) – WSKP Call**
Three inches of blank tape are written.

8-6.6.9 DMA Magnetic Tape FIOT

The DMA magnetic tape FIOT conforms to the standard FIOT configuration described in the I/O Monitor documentation with the following exceptions.

The alpha-binary bit, bit 0 of word 1, is not used since the tape is not formatted. However, if the tape unit is a two-speed unit, this bit serves as the speed select bit. A zero is normal speed, a 1 is low speed.

The special format bit, bit 0 of word 6, must be a zero. If it is a one, the requested operation is not performed.

If the DMA magnetic tape record chaining feature is used, certain other variations in the FIOT structure must be observed, as described in paragraph 8-6.6.11 under *DMA Magnetic Tape Record Chaining Feature*.

8-6.6.10 DMA Magnetic Tape Status

The DMA magnetic tape status response word is stored in word 3 of the FIOT at the conclusion of each read or write operation. It is not returned for special operations. The word has the following configuration.

Bit Set	Use
0	Controller or device not ready
1	Device not ready
2	Beginning of tape
3	End of tape
4	Rewinding
5-9	Not used
10	Magnetic tape continue last operation
11	End-of-file detected
12	Transferred byte count (0 = even; 1 = odd)
13	Write enabled (write ring in)
14	Rate error
15	Parity, rate or CRC error

If bit 15 is set in this word, an unrecoverable read or write error has occurred. If an unrecoverable write error occurs, the user may lay down blank tape by backspacing, then writ-skipping, and then re-try the operation.

9-6.6.11 DMA Magnetic Tape Record Chaining Feature

The optional record chaining feature permits very long magnetic tape records to be read or written with no end-of-record gap following each buffer transfer. This is normally a multiple buffered operation. The tape controller generates an interrupt signal as each buffer transfer is completed and proceeds automatically to the next buffer. At this time the driver may set up to continue the chain when that buffer is complete.

The chaining feature is implemented in the driver whenever the supporting hardware is included in the system. Separate function codes are used to request chain operations. Code 1 requests a chained read. Code 2 requests a chained write operation. In order for the operation to proceed automatically, a sequence of linked FIOT's provides the driver with the necessary information as to buffer location, word counts, etc. Word 7 of each FIOT in the chain points the next FIOT. The chain of FIOT's may be of any length desired, and may be (and usually is) circular. If it is circular, there must be at least two FIOT's in the circle; i.e., a FIOT may not point to itself.

Only the first FIOT in the string must be stuffed with the unit number and function code. The driver will copy this information into the other FIOT's in the chain.

The chain operation is initiated by a standard DOIO call on the first FIOT in the chain. As the associated buffer for each FIOT is being filled, the FIOT's busy bit will be true and a STAT call on the appropriate FIOT may be used to test that bit, and to detect hardware errors.

In a circular chain, it is possible for the controller to attempt to initiate a transfer to a buffer containing data previously transferred but not yet used. This situation would result in a loss of data; therefore, a feature has been added to detect this error. Bit 0 of word 7 of the FIOT is a buffer protect bit. The driver sets this bit true when a buffer has been filled and at the same time sets the busy bit *Off*. If later, the driver returns to that

FIOT and finds the buffer protect bit still on, it assumes that it is now filling a buffer with which the user has not finished. This is called a software rate error. The chain operation is stopped immediately, the tape is allowed to proceed to end-of-record, and an error status is returned to the next FIOT in the chain. The user must set the buffer protect bit off before a new transfer into the buffer is initiated. Normally, this is whenever he has finished processing that buffer. A software rate error is flagged by the status response found in word 3 of the FIOT for which the error is detected. The above sequence is described in terms of tape read operation; however, the same considerations apply to a write operation.

Chained I/O will continue until the pointer word (7) to the next FIOT contains a zero. For a circular chain, a write operation will continue indefinitely; a read operation will continue until an end-of-record gap is encountered. When this occurs, status and ending byte address will be returned to the current FIOT. The byte address represents one more than the address of the last byte actually transferred. The appearance of this data in word 4 signals the user that the end-of-record has been encountered. At all other times, this word contains zero.

For a write operation with a circular chain, the operation is stopped by setting word 7 to zero. Of course, the pointer must be restored before the next write operation is begun.

If an end-action is desired at the conclusion of the operation, the address of the end-action routine must appear in word 6 of each FIOT in the chain; the last FIOT used supplies the end-action address.

A typical pair of FIOTs to sustain a chained read operation might appear in assembly text as follows:

FIOT1	DATA	BUF1	Word 0
	DATA	WCT1	Word 1
	DATA	X'81'	Word 2
	RES	3	
	DATA	ENDACT	Word 6
	DATA	FIOT2	Word 7

FIOT2	DATA	BUF2	Word 0
	DATA	WCT2	Word 1
	RES	4	
	DATA	ENDACT	Word 6
	DATA	FIOT1	Word 7

The first three words of each FIOT may also be set by a call to OPEN. Word 2 of FIOT1 requests a chain read on logical unit 8. This word is copied into FIOT2 by the driver.

8-6.7 DIO Magnetic Tape Driver

8-6.7.1 Purpose

The DIO Magnetic Tape Driver provides a method for the Raytheon 706 IOS user to transfer data between the DIO Magnetic Tape units and the computer CPU.

8-6.7.2 Usage

This driver performs the following functions:

1. Read a record
2. Write a record
3. Write an end-of-file
4. Backspace a record
5. Seek for end-of-file
6. Rewind
7. Write skip (erase forward)

The I/O calls available to the IOS user are discussed in the following paragraphs.

8-6.7.3 OPEN

The first call for any I/O operation is to subroutine OPEN. OPEN translates the arguments to the proper format and inserts them into the FIOT table.

OPEN has a fixed length calling sequence as follows:

OPEN FIOT, BUF, WC, UNIT, OPER, MODE

where:

FIOT is the location of an 8-word array for the file description table.

BUF is the data buffer address.

WC is the address of the number of words.

UNIT is the logical unit number.

OPER is the type of I/O operation (OPER = X'9' for read and X'E' for write).

MODE is not used by this driver and should be zero (0).

8-6.7.4 DOIO – Read or Write a Record

Actual I/O operations are initiated with calls to DOIO. DOIO changes the FIOT table as required, calls a driver to initiate the I/O operation and returns to the user.

The calling sequence is as follows:

DOIO FIOT, BUF, WC

where:

BUF is the data buffer address.

WC is the address of the number of words.

DOIO has a variable length calling sequence. If BUF and WC do not change from the previous call or have been set by OPEN, they need not be included. However, if WC is to be changed, BUF must be included also.

8-6.7.5 STAT – Status

STAT provides the user with a means of waiting until an I/O operation is complete. STAT has a variable length calling sequence as follows:

STAT FIOT, ERR

When STAT is called, it will loop until the FIOT busy bit is off. If the ERR argument is not

included, STAT will return to the next cell following the STAT call. If argument ERR is in the calling sequence, STAT will check bit 15 of FIOT (3). If there was an error, STAT will load the accumulator with the normal return address and transfer to the ERR address. If there were no errors, STAT returns to the next cell following the ERR argument.

8-6.7.6 STUS – Device Status

The physical status of any I/O device may be interrogated at any time. The call is

STUS FIOT

The device's physical status response is returned in the accumulator. The FIOT is not disturbed by this operation, except to substitute the actual physical device number for the logical unit, if this has not already been done. An I/O operation in progress will not be disturbed.

8-6.7.7 WEOF – Write End-of-File

The write end-of-file call is

WEOF FIOT

A hardware detectable file mark is written on the tape.

8-6.7.8 SEOF – Seek End-of-File

The call is

SEOF FIOT

The tape advances until a file mark is detected. The tape is positioned in the gap just past the file mark.

8-6.7.9 BKSP – Backspace a Record

The selected tape unit backspaces one record. The call is:

BKSP FIOT

8-6.7.10 WKSP – Write Skip

The specified magnetic tape unit erases forward three inches of tape. The call is:

WSKP FIOT

8-6.7.11 REWD – Rewind

The call is:

REWD FIOT

The selected tape drive is rewound. Other tape units are available for I/O operations while a tape drive is rewinding.

Refer to Section 5-4 for a detailed description of the IOS calls, FIOT configuration and the IOS call linkage addresses.

NOTE: User must ensure that word 6 of the FIOT is zero. Information in word 6 does not apply to this driver. Skip one record forward can be done by reading with the word count equal to zero.

8-6.7.12 Special Functions

The special functions include continuous writing, continuous reading and continuous backspacing. If the same command is issued to the same tape unit within 1.4 ms following the end of function interrupt (i.e., write after write, read after read, or backspace after backspace), the tape does not come to a stop in the gap, but its motion continues.

8-6.7.13 Device Status Word

Bit Position	Meaning
0	True if controller is busy
1	True if device is busy.
2	True if at BOT (beginning of tape, load point).
3	True if at EOT (end of tape).
4	True if device is rewinding.
5	Not used.

Bit Position	Meaning
6	True if end of function interrupt occurred.
7	True if data transfers interrupt occurred.
8–10	Not used.
11	True if EOF (end of file) was detected.
12	Not used.
13	True if device is not write protected.
14	True if rate error occurred.
15	True if parity error, CRC error, LRC error, or rate error occurred.

8-6.7.14 Data Format

9-Track Data Word

0 1 2 3 4 5 6 7	8 9 10 11 12 13 14 15
-----------------	-----------------------

1st byte

2nd byte

7-Track Data Word

0 1 2 3 4 5	6 7 8 9 10 11	12 13 14 15
-------------	---------------	-------------

1st char.

2nd char.

not used

The user is responsible for data formatting on the 7-track magnetic tape.

8.6.7.15 Error Processing

Input operations are repeated a maximum of five times, if an error occurs. If the error persists, the driver returns to the user with the current device status in word 3 of the FIOT. The tape is positioned in the gap following the record in error.

8-6.7.16 Method

Initially the driver makes certain that no other I/O operation is active before initiating the requested operation for the selected DIO magnetic tape. When read, write, and write end-of-file operation have been initiated, the driver does not relinquish control until the particular operation is complete.

The write skip, write end-of-file, rewind, and backspace functions are processed by the I/O monitor. These functions are just initiated when control is returned to the user. The device status should be tested if a device busy condition is to be determined.

A special subroutine has been written for the I/O monitor to handle the seek for end-of-file operation on the DIO magnetic tape unit. A continuous read (skip record forward) operation is performed until an end-of-file mark is detected. Program control is not returned to the user until the file mark has been detected.

The driver does not use the priority interrupt system. No interrupts should occur while a DIO magnetic tape operation is being performed. Bits 6 and 7 of the device status word sense the end of operation and the data transfer time.

8-6.7.17 Program Restrictions and Space Used

The I/O operation on the DIO magnetic tape unit cannot occur simultaneously with I/O operations of other devices.

The space used is 144 decimal words.

8-6.8 Digital Plotter Driver

8-6.8.1 Purpose

The Digital Plotter Driver provides a method for the Raytheon 706 IOS user to transfer data between the Calcomp Digital Plotter and the computer CPU.

8-6.8.2 Usage

Calling Sequence

OPEN FIOT, BUF, WC, UNIT, OPER, M

FIOT	Address of the file input/output table as defined in "XRAY EXEC and System Programs - BASIC" 1-3
BUF	Beginning buffer address of input buffer
WC	Address of number of words
UNIT	Logical unit number designating plotter to be used
OPER	Plotter number
M	M is ignored and should be zero.

DOIO FIOT, BUF, WC

FIOT, BUF, WC have the same definitions as above and are used to change the FIOT table upon call.

The input buffer consists of data entries each of which represent one or two plotter commands. The first plotter command to be transmitted to the plotter control should be stored in the left byte of the first entry. An entry has the following format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	PEN	DRUM	CAR	0	0	PEN	DRUM	CAR						

Bits 2-3 and 10-11 determine pen position as follows:

- 00 no pen (uses the last mode selected)
- 01 pen down,
- 10 pen up,
- 11 pen up, increment, pen down

Bits 4-5 and 12-13 determine drum movement direction as follows:

- 00 no drum,
- 01 drum down (+Y line segment),
- 10 drum up (-Y line segment),
- 11 no drum.

Bits 6-7 and 14-15 determine carriage movement direction

- 00 no carriage
- 01 carriage left (-X line segment)
- 10 carriage right (+X line segment)
- 11 no carriage

8-6.8.3 File Input Output Table (FIOT)

Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Word 0	B	BBA														
1	ANW															
2									LUN				UNIT			
3	STATUS															
4	D	WBA														
5	EBA															
6	EAA															
7	PBA															

- B busy bit, indicates driver busy.
- BBA beginning buffer address.
- ANW address of number of entries in buffer.
- LUN logical unit number, used by the I/O monitor to determine drivers entry location in the PEAT Table.
- UNIT physical plotter number (1-4)
- D job done bit, indicates previous task initiated through this FIOT was completed.

- WBA If job done bit is on, WBA will contain the address from which the driver will select the next entry.
- EBA end buffer address, computed by driver.
- EAA end action address, specified if end action is desired upon completion.
- PBA plotter buffer address, completed and maintained by driver, address of next command to be executed.

8-6.8.4 Method

The driver performs data transmission to the plotter controller on an interrupt basis. When the DOIO call is executed, the driver selects the proper plotter (1-4) then exits to the calling program.

With each interrupt that occurs, the driver loads an entry and outputs it to the plotter. Then the driver replaces that entry with a word of zeroes and increments the buffer pointer. The current buffer pointer is saved in word 7 of the FIOT. When the end of the buffer is reached, the buffer pointer is set to the initial buffer address.

The plotter driver continues to output plotter commands when interrupted, until a word of zeroes is loaded from the buffer. When a zero entry is sensed, plotting is terminated, the current buffer pointed (word 7 of the FIOT) is stored in word 4 of the FIOT with the sign bit on and the driver exits.

On subsequent DOIO calls to the plotter driver, the sign bit of word 4 of the FIOT is checked. If the sign bit is on, the content of word 4 is used as the current buffer pointer. Otherwise, the content of word 0 of the FIOT (beginning buffer address) is used. In this manner plotting can be continuous providing the calling program stores entries into the buffer as the entries are output by the driver.

If data is to be output from the beginning buffer address for each DOIO call, word 4 of the FIOT must be cleared prior to each DOIO call.

If end action is specified, control will be transferred to the given end action address upon completion.

8-6.8.5 Space Used

The space used is 73 decimal words.

8-6.9 Plotter Interface Routine

8-6.9.1 Purpose

This subroutine produces the plotter commands necessary to plot from the current position of the pen to a point, or through a series of points, specified by the calling program.

8-6.9.2 Usage

Calling Sequence

SMB	WPLLOT1	
JSX	SPLOT1	
DATA	PLNPEN	
DATA	IXARRAY	X direction buffer
DATA	IYARRAY	Y direction buffer
DATA	NUMPTS	Number of points

PLNPEN is the address of a word describing the desired plotter number (1–4) in the left byte and the pen position, as described below, right oriented in the right byte.

1	PEN DOWN	(solid line)
2	PEN UP	(no line)
3	PEN DOWN, PEN UP	(dashed line)

IXARRAY and IYARRAY are the beginning addresses of the buffers containing the coordinates through which a plot is to be constructed. NUMPTS is the address of the number of points contained in the buffers. The user must maintain a one-to-one correspondence between coordinates in IXARRAY and IYARRAY. That is, the point (X₁

Y₁) is defined by the first entry in IXARRAY and the first entry in IYARRAY, respectively. These coordinates must be integer values indicating the number of machine increments and direction to move in order to reach the specified point.

8-6.9.3 Method

The algorithm implemented in this subroutine provides plotting of an approximate line between two points that is as close as possible to the desired line. There are two keyvalues computed and maintained within the subroutine, a ratio and a sum. A ratio is obtained for each point from the expression:

$$\text{RATIO} = \frac{\text{Larger Coordinate}}{\text{Smaller Coordinate}}$$

The sum, initially biased by 0.5, is used to select the appropriate plotter command. For each plotter command generated the ratio is added to the sum, the new sum replaces the previous sum and the overflow condition is tested. If an overflow condition occurred in attaining the new sum, a diagonal plotter command is stored in the output buffer. If no overflow occurred a straight line in the appropriate X or Y direction will be used. This procedure continues until number of commands, as specified by the larger coordinate, have been produced, at which time the plotter driver is called.

The *continuous plotting* method is used in order to keep the plotter as busy as possible. A 100 word buffer is filled then the driver is called. The program monitors the progress of the driver (cleared buffer positions) and continues to fill this buffer until the specified number of points have been plotted.

8-6.9.4 Space Used

The space used is 352 decimal words (including 100 word buffer).

8-6.10 Adding User Created I/O Drivers

There are three methods for adding user-created I/O drivers to 706 Standard, MTOS, and RTOS operating systems. The driver interface devices are not provided for in standard Raytheon software.

8-6.10.1 Method 1

Method 1 requires that the driver be written as an I/O service routine. The starting address of the routine is *stuffed* into the proper linkage cell ($4N + 1$, where N is the interrupt level) after the monitor has been initialized. Section 5-2.3 of this manual describes Method 1 in detail. The major requirements of Method 1 are:

Requirements	Where Accomplished
1. Original device selection	User's main program
2. Enable interrupt level	User's main program
3. Save and restore registers	User's Driver
4. Processing of data after interrupt.	User's Driver
5. Selection of device for next operation (if necessary).	User's Driver
6. Return to main program	User's Driver (Issue INR to proper interrupt level)

Method 1 does not allow the use of SAVE or RESTORE routines, logical unit numbers, DOIO calls, or an FIOT table.

8-6.10.2 Method 2

In Method 2 the user must write a driver that interfaces to the STAK, SAVE, and REST and DRIVEX routines of the I/O monitor (Figure 8-16). Interfacing is accomplished by inserting the starting address of the driver into the proper place in the *interrupt stack* (STAK).

Requirements of Method 2 are:

Requirements	Where Accomplished
1. Original device selection	User's main program
2. Enable Interrupt level	User's main program
3. Save registers	Done by SAVE
4. Process data after interrupt	User's Driver
5. Select device for next operation (if necessary)	User's Driver
6. Restore registers	Done by REST
7. Return to main program	Done by REST or DRIVEX

The location of the interrupt stack is in the interrupt linkage cell for a given interrupt level. A sample method of stuffing the interrupt stack with the starting address of the driver is:

```
SMB      0
LDX      N+N+N+N+1
LDW      STARTADD
STW      * 8
```

Where: N is the interrupt level to be processed and STARTADD contains the starting address of the driver.

When bit 0 of the STARTADD is true, the SAVE routine disables the interrupt level before transferring to the user's driver. All interrupt level stuffing should be done after the monitor is initialized.

The user's driver should exit through REST or DRIVEX. If the driver takes advantage of the *disable interrupt* feature of the SAVE routine, the DRIVEX return should be used, otherwise the RESTORE routine should be used.

Method 2 does not permit logical unit numbers, DOIO calls nor FIOT tables, but does allow interrupt stringing and the optimal priority override features of the I/O monitor.

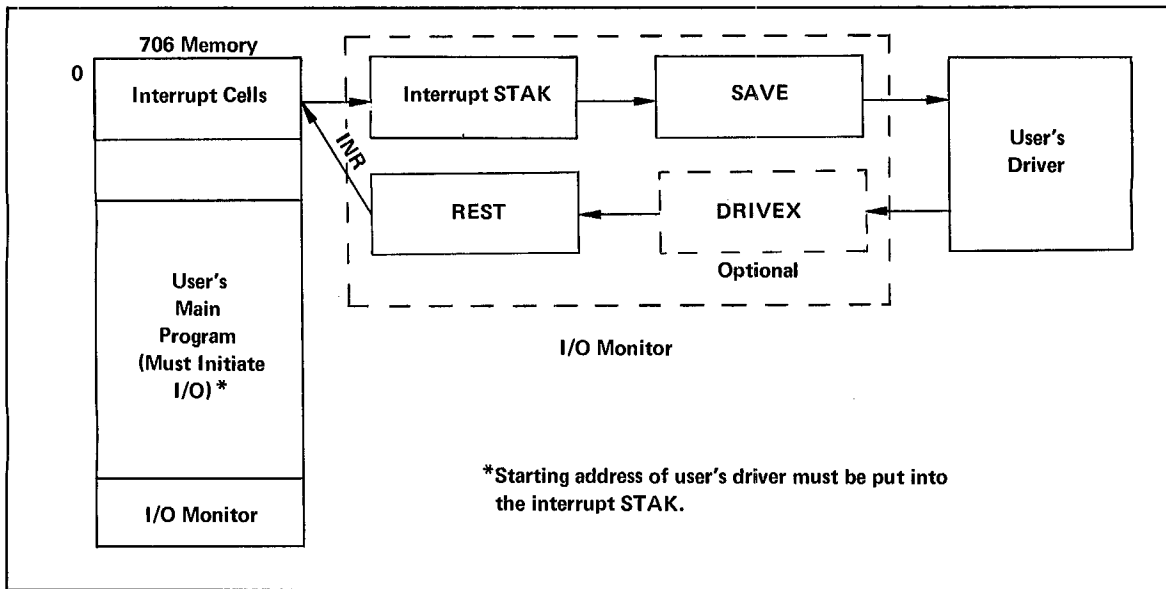


Figure 8-16. Adding User's Driver – Method 2

8-6.10.3 Method 3

Method 3 (Figure 8-17) is similar to Method 2 in that the routines SAVE, REST, STAK, and DRIVEX are used, but additionally the logical unit reassignment capabilities of the PEAT table and the generalized I/O calls to OPEN, DOIO, and STAT are permitted. These additional features are accomplished through the use of M.DDR, INTENB, and DOIO routines.

In order to write an I/O driver for Method 3, the user must write two routines – a *Setup* routine and an *Interrupt Handling* routine.

The requirements of Method 3 are:

Requirement	Where Accomplished
1. Original device selection	User's setup routine
2. Enable Interrupt Level	Done by DRIVEX (transferred to by setup routine)
3. Save registers	Done by SAVE
4. Process data after interrupt	User's Interrupt Handling routine

Requirement	Where Accomplished
5. Select device for next operation	User's Interrupt Handling Routine
6. Restore registers	Done by REST
7. Return to main program	If last interrupt, done by M.DDR which exits through REST

When a user calls DOIO, the DOIO routine transfers to the user's setup routine with the following information:

1. Cell M.DF will contain the address of the user's FIOT table (bit 0 of M.DF will be on if it was a special format call).
2. Buffer address located in word 0 of the FIOT.
3. Word count address located in word 1 of the FIOT.
4. Bit 0 of the word 0 of the FIOT set to 1 to indicate FIOT busy.

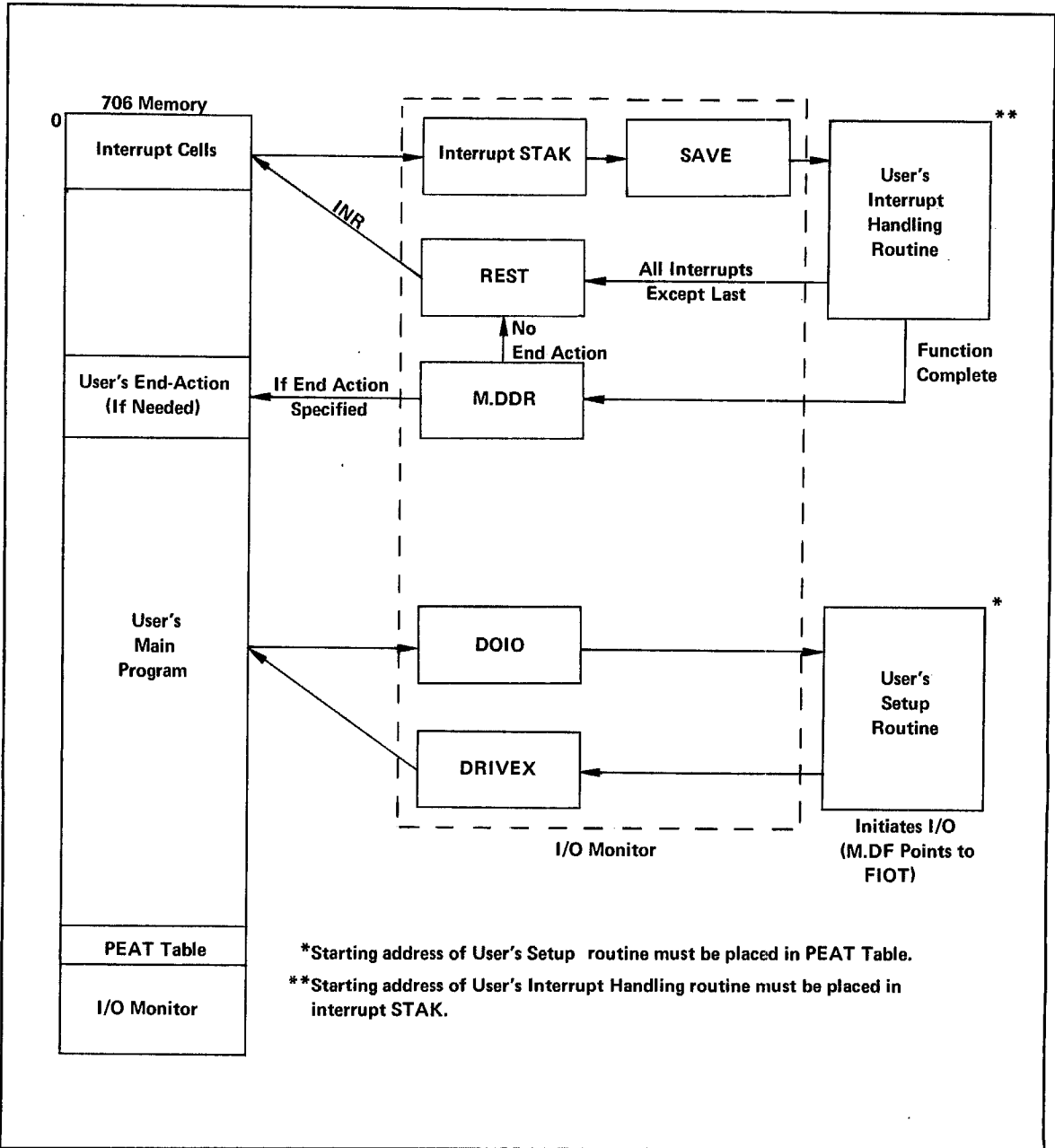


Figure 8-17. Adding User's Driver – Method 3

5. The accumulator and M.DC both contain the first of the two words for the unit in the PEAT table. (LUN & FC)
6. Word 2 of the FIOT will contain the function code in bits 12–15.

The setup routine should do the following:

1. Call INTENB to determine if initiation of I/O is allowed at this time (if it is not, INTENB will stuff 00FF₁₆ into the status word of the user's FIOT and return to setup without returning to the user's Interrupt Handling routine.
2. Do whatever setup and housekeeping the user wishes.
3. Issue the DIN's and/or DOT's necessary to get the device into operation.
4. If the device is to operate off of interrupts, the setup routine should exit to DRIVEX which will enable the interrupt and return to the user.
5. If no interrupts are to be used, all processing should be done within the setup routine.

After the entire operation that was requested by the call to DOIO is completed, the setup routine should turn off the busy bit in the FIOT and return to the main program by picking up M.DR (not to be confused with M.DDR):

```
SMB    0
LDX    X'49'      get recursive cells
                    pointer
LDX    * 4        get M.DR
JMP    * 1        return to DOIO caller
```

The address of the setup routine must be placed into the PEAT table. A sample method of doing this is:

```
SMB    0
LDX    X'59'      pick up PEAT pointer
LDW    ADDSETUP  address of setup routine
STW    * L+L-11
```

where L is the logical unit for handling the data and ADDSETUP contains starting address of setup routine. L must be greater than 11, i.e., only entries in the physical portion of the PEAT table are processed this way (all entries in the logical portion ultimately point to an entry in the physical portion).

If data is to be handled by interrupt processing, an interrupt handling driver must be written exactly as in Method 2. The only difference is that when the operation is complete, the driver should exit through M.DDR which will turn off the busy bit and transfer to an end-action address if one is specified. All exits from the driver prior to the last or *function complete* exit, should be made through REST just as in Method 2.

8-6.10.4 User Driver Residency

User drivers (and setup routines if Method 3 is being used) can be operated either as subroutines which are loaded each time they are used, or can be made resident so that they need not be loaded with each program that needs them.

If residency is not required, the proper linkages to the user's routines (interrupt cells, interrupt STAK, and/or PEAT table) must be established before the routines are used. Since the routines are not resident, the values of these linkages prior to the changes should be saved, then restored prior to exiting from the main program.

If the routines are to be made resident, they should be written as an executable main program that consists of two parts. The first part initializes, establishes links, and establishes residency. The second part is the actual processing of the driver, the part which is to be made resident. (If Method 3 is being used, this part can contain both the interrupt handling and setup portions.)

The first part need not be made resident as its function is a one-time job. It should establish the linkages to the processing portion (interrupt cell 41 + 1 if Method 1; interrupt STAK if method 2 or 3; and PEAT table if Method 3). Residency of the second part is accomplished by putting the address of the lowest cell used into ULIM, ENDM, and LOWLOAD (X,5A', X'5B', and X'66').

Routines that are to be made resident in this manner can be written in absolute or relocatable. However, relocatable is recommended since all locations from the beginning of the driver up to and including the monitor will be made resident, therefore unavailable for loading. The relocating loader guarantees the tightest packing of routines.

8-6.10.5 User Driver Stack Information

Associated with each interrupt level are 11 locations in the monitor called the stack. The general form of a stack is:

Word	Operation	Operand	Comment
0	STX	\$+3	Saves IXR when interrupt occurs
1	JSX	SAVE	Transfers control to SAVE
2	INR	N	Interrupt Return Instruction
3	D	0	Saves contents of IXR
4	D	0	Saves contents of ACR
5	D	0	Saves contents of temporary cell M.DR
6	D	0	Saves contents of temporary cell M.DF
7	D	0	Saves contents of temporary cell M.DC
8	D	Variable	Interrupt Service Address
9	D	0	Address of FIOT used by interrupt level.
10	D	0	Saves INTPOINT

8-6.10.6 Primary User Driver Routines

The most used driver routines are each described in the subsequent text.

SAVE Routine (Figure 8-18)

Purpose — To save ACR, IXR temporary cells upon an interrupt, string this interrupt with previous interrupts, disable the interrupt level if requested, and transfer to driver which handles this level.

Calling Sequence — Called (with JSX) by interrupt stack for level which has interrupted.

Timing — 37 cycles if disable requested.
30 cycles if disable not requested.

Core Requirements — 19 locations.

REST Routine (Figure 8-19)

Purpose — To do the housekeeping associated with leaving the driver for an interrupt level by restoring M.DR, M.DF, M.DC, setting the software inactive bit in the machine status word, restoring INTPOINT, ACR, IXR and executing an INR for the level.

Calling Sequence — JSX REST.

Access — The address of REST is the 13th word of the DVEC table

```
SMB 0
LDX X'74'    get DVEC pointer
LDW * 13     get REST address
```

Timing — 42 cycles

Core Requirements — 22 locations

INTENB Routine (Figure 8-20)

Purpose — To check for the situation where an I/O request is being made to a device whose level is currently active.

Calling Sequence — JSX INTENB
D STAKINR

Return if interrupt level is available.

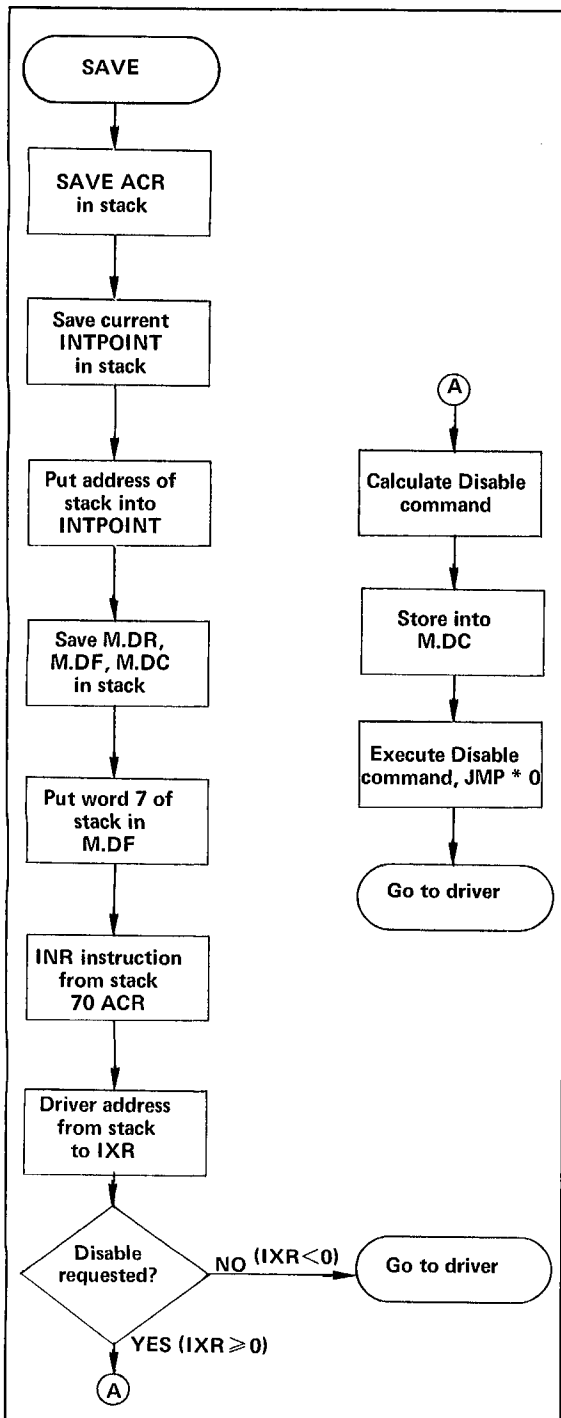


Figure 8-18. SAVE Flowchart

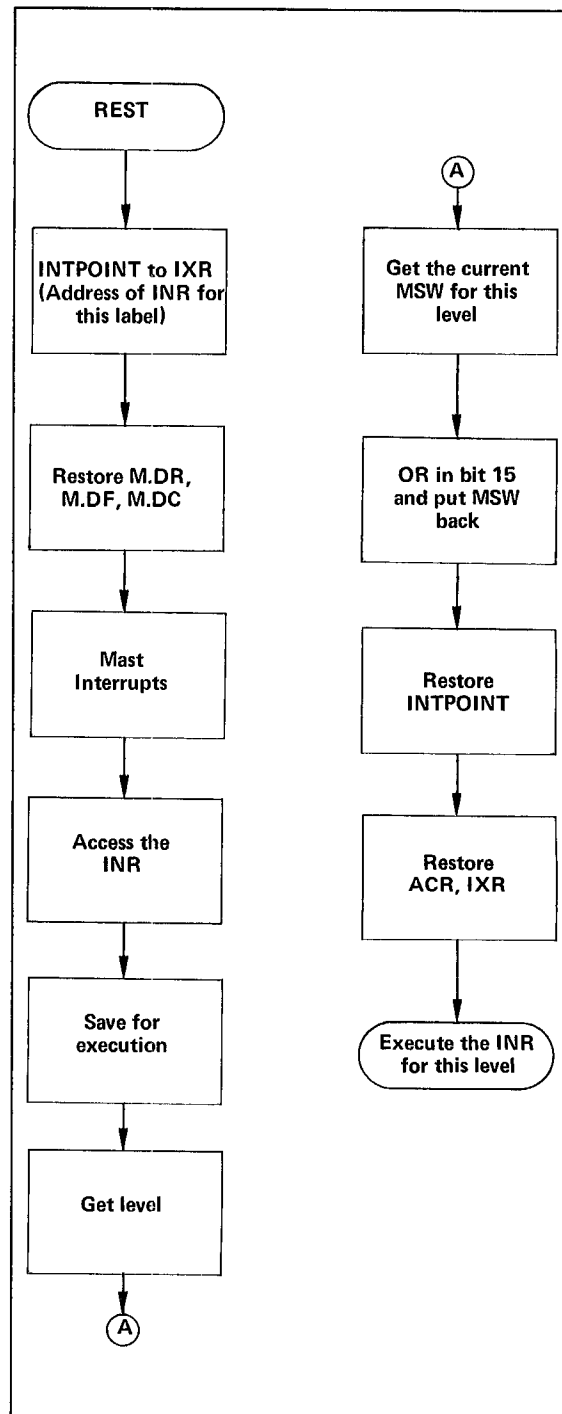


Figure 8-19. REST Flowchart

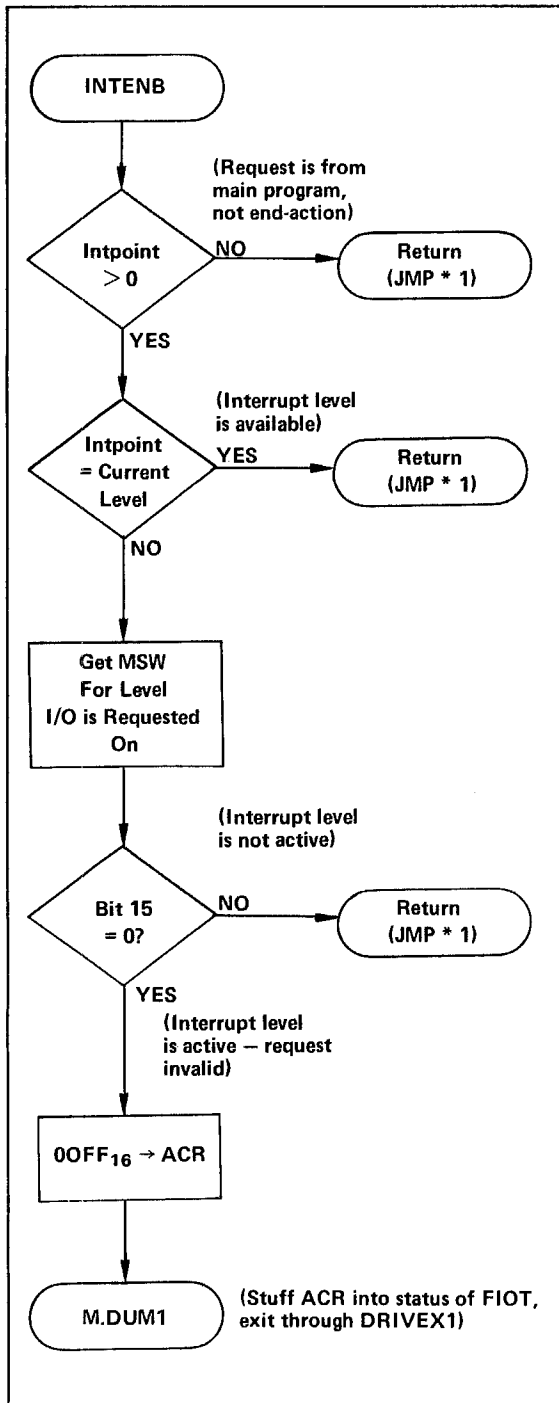


Figure 8-20. INTENB Flowchart

Access – The address of INTENB is the 14th word of the DVEC table.

```

SMB  0
LDX  X'74'  get DVEC pointer
LDW  * 14   get INTENB pointer
  
```

Timing – 4 cycles if called from machine rather than end-action

7 cycles if end-action is requesting currently active level.

23 cycles if request is to a level other than current which is not active.

37 cycles if level is in use (+ DRIVEX)

Core Requirements – 20 locations.

DRIVEX Routine (Figure 8-21)

Purpose – To enable the interrupt level for an I/O operation and either exit through M.DR or restore interrupt level, depending on whether I/O request was generated by main program or by end-action.

Calling Sequence – JSX DRIVEX
D STAKINR

where STAKINR is the address of the INR instruction within the interrupt stack for level to be enabled.

No return is made from DRIVEX to the calling program. Exit is either made through M.DR or the restore routine REST.

Access – The address of DRIVEX is the 15th word of the DVEC table.

```

SMB  0
LDX  X'74'  get DVEC pointer
LDW  * 15   get DRIVEX pointer
  
```

Timing – 14 cycles if request was made from end action.

17 cycles if request was made from main program.

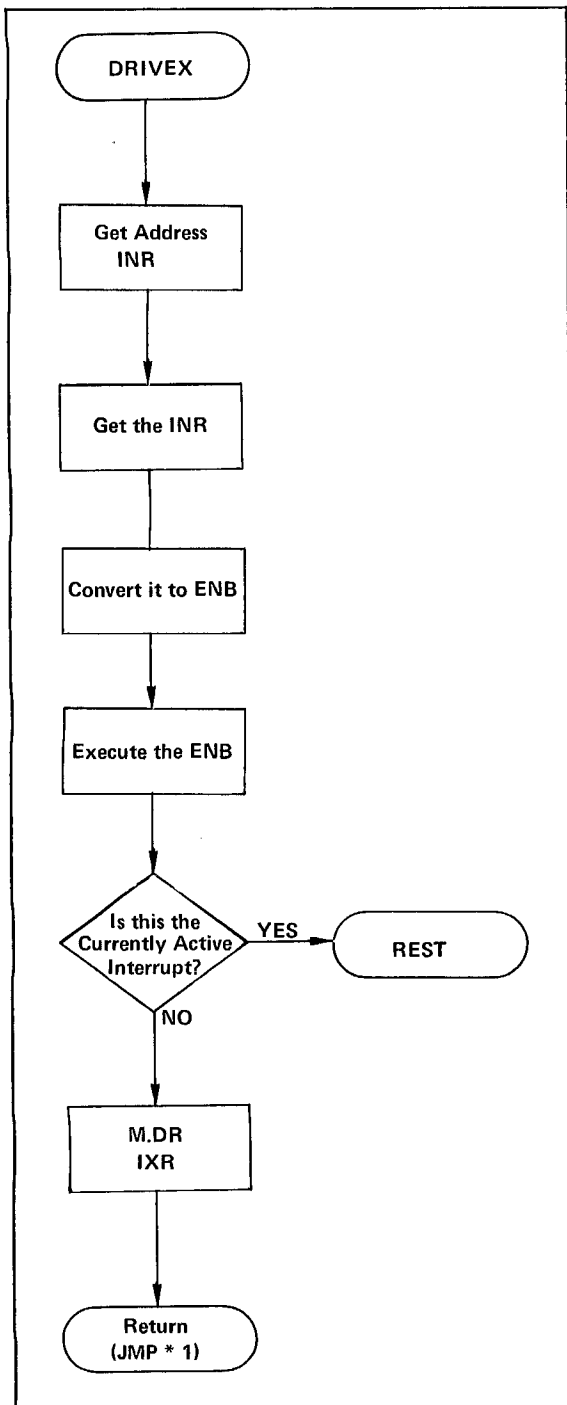


Figure 8-21. DRIVEX Flowchart

Core Requirements – 12 locations.

M.DDR Routine (Figure 8-22)

Purpose – To turn off busy bit of FIOT, M.DDR will transfer to end-action subroutine if one is specified, restore through REST of end-action not specified.

Calling Sequence – JSX M.DDR

No return is made from M.DDR.

Access – The address of M.DDR is the sixteenth word of the DVEC table.

SMB	0	
LDX	X'74'	get DVEC pointer
LDW	* 16	get M.DDR pointer

Timing – 15 cycles if end-action specified
 15 cycles if end-action not specified

+2 if HSR or HSP or TTY MUX
 +4 if HSR or HSP
 +4 if TTY Multiplexer

Core Requirements – 9 locations.

- add 1 cell if either
 - a) HSR or HSP on system
 - b) Teletype Mux on system
- add 9 cells if either HSR or HSP
- add 2 if both HSR and HSP
- add 8 if TTY MUX on system

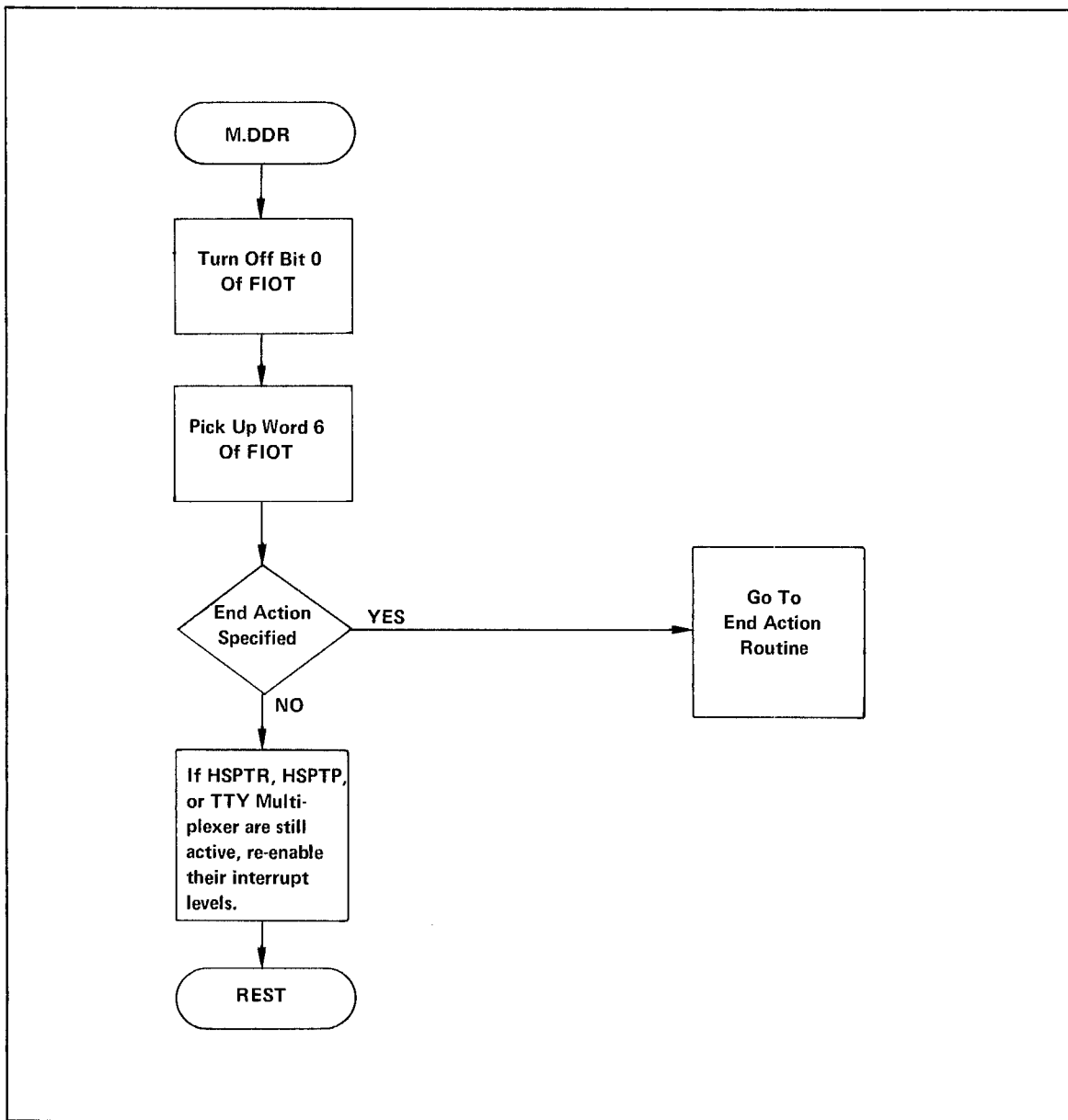


Figure 8-22. M.DDR Flowchart

8-7. OPERATING SYSTEMS

8-7.1 Introduction

Operating systems are software systems used to control the flow of jobs through a computer center. These systems maximize the productivity of the computer center by minimizing idle time between jobs, allocating system facilities efficiently, and scheduling jobs wisely.

Modern operating systems require a very high degree of organization and standardization of the programs and data they control.

In an operating system all system programs, such as language processors (FORTRAN), utility routines, debugging aids, and applications programs written by users, interact with the operating system rather than the computer itself. This assures that programs written by different persons will be compatible. Note however, that the flexibility and compatibility associated with standardization is achieved at the expense of overhead such as machine time, memory requirements, software development, and maintenance cost.

Most operating systems assist in performing three major tasks: program development control, job control, and data control.

8-7.1.1 Program Development Control

The 706 Operating systems accept properly identified programs in several source languages and call on the proper translator to reduce the statements to some form of machine language. Each translator produces modules of *standard* format, relocatable object code, that can be combined with other modules of code. Once programs are reduced to this standard format, the source language is immaterial. This technique of putting together several modules of code to make a program aids the user by allowing the use of various, previously-coded, *standardized* routines (i.e., data handling and mathematical routines). For example, at load time, routines originally written in FORTRAN, and assembler language can be linked together to provide a combined system. This use of standardized routines saves the programming costs of such routines being rewritten each time they are needed.

8-7.1.1 Job Control

The basic form of job control implemented under an operating system is that of *batch processing* (see Figures 8-23 and 8-24). Batch processing is the sequential processing of jobs. RTOS, a more complicated batch operating system, takes its input stream from a job stack. The stack may be a deck of cards, a magnetic tape, or on disc. RTOS uses a table to tell the system which job is the next in sequence. The table, called the system queue, is composed of program names and their associated priorities. In RTOS when a job is completed, the table is searched for the highest priority number. The program corresponding to this priority number (not necessarily the next in sequence) is then fetched from the disc library and executed. Improvements on this batch processing system with priority control have led to the rise of the multiprogramming system (MPS). See Section 8-7.6.

In examining various forms of job control, the hardware must be considered. For instance, it is impractical to implement a batch processing system with paper tape as the basic form of system input/output because it is too slow and difficult to maintain a continuous sequential string of jobs. Such a system requires at least a card reader. Another hardware consideration employs the queued system that uses a mass storage device. One outgrowth of simple batch processing is the ability to do *translate, load and go* (commonly called "load and go") processing. Such a system requires the use of rapid access intermediate storage (i.e., disc or tape) for storing the object module output of the translation step. The next step is to load the object module from intermediate storage, do any necessary linking, then execute it. The Real-Time Operating System and Multi-Programming System have a load and go capability for both FORTRAN IV and SYM II source programs.

The flexibility of an effective operating system increases operational efficiency. With a job-to-job transition automatically provided, the programmer can decide the next processing step based on object time conditions and cause the appropriate program to be called. Previously, this would have necessitated elaborate operating instructions and

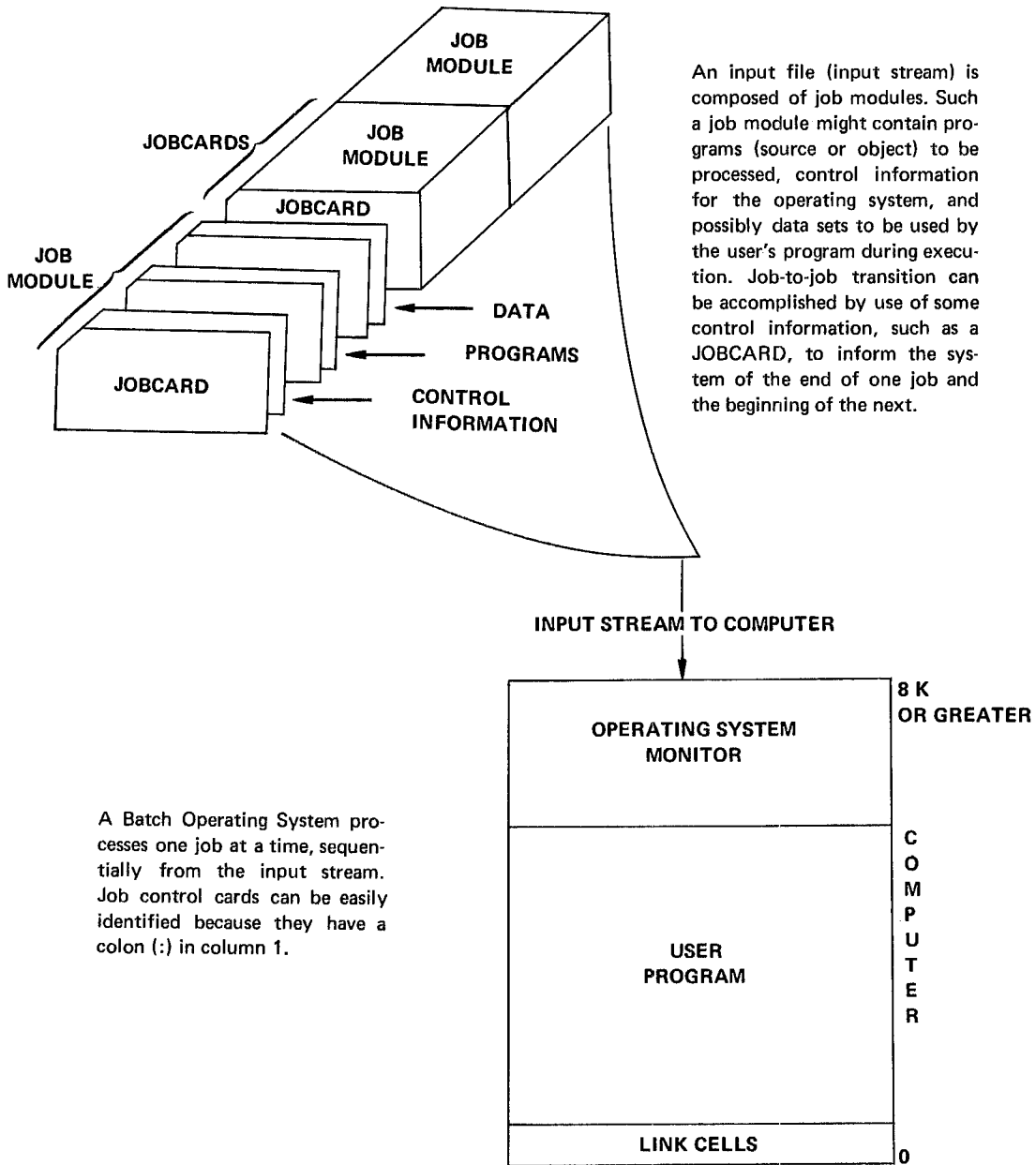


Figure 8-23. Job Control Under A Basic Batch Processing Operating System

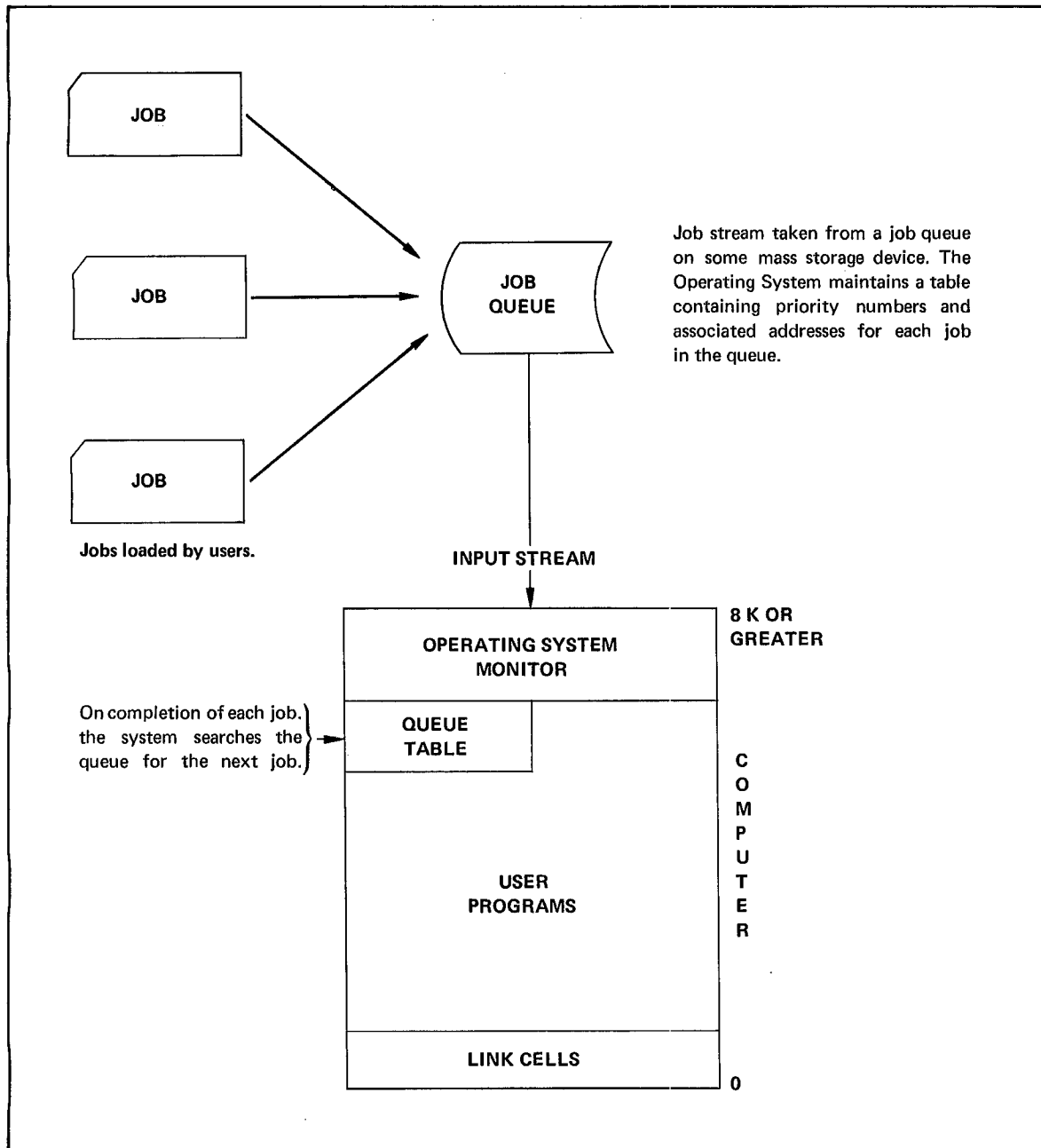


Figure 8-24. Job Control Under A Queued Sequential Batch Processing Operating System

constant operator action. In a similar way, a program too large for the computer's memory can be constructed as a series of overlays which can be transferred into and out of the machine as needed.

The operating system provides a uniform language and procedure for communicating with the computer operator. Since all programs are controlled by the operating system, the system informs the operator of program status. For instance, whenever an I/O device is not on line and is being used, the operator will receive a standard message. Without an operating system, each programmer might compose his own message, or none at all, thereby confusing and delaying the operator.

This standard operating environment also helps the programmer. He does not have to decide when each new program overlay is needed. Also, operating standards are maintained with a minimum of programming indoctrination and supervision.

8-7.1.3 Data Control

The goal of complete data management facilities in an operating system is to eliminate the need for programmer attention to the physical characteristics of the data storage and data organization, and to allow him to concentrate on the logical properties of the data. To accomplish this, Raytheon provides several standard forms of data organization and standard data access methods. The programmer selects that method which is best suited to his application and uses pseudo-instructions which perform logical operations upon the data set. Some of the functions this type of software offers are:

1. Control of the physical movement of data between central storage and auxiliary storage.
2. Detection and correction of errors associated with data transfer.
3. Data buffering, blocking and deblocking, and overlapping data operations with computing.

4. Dynamic scheduling of input/output devices.
5. Logical handling of data sets without reference to physical characteristics of the storage devices: thus data sets stored on discs, cards, tape, or other media can be handled similarly and independently of device type.

8-7.2 BASIC Operating System

The BASIC hardware configuration, with *no* high-speed input device precludes the development of a true operating system as defined in the preceding paragraphs. The BASIC configuration does have an XRAY Executive program (Appendix K-6) and an Input/Output Monitor (complete with I/O drivers and relocatable loader) to assist in the preparation, execution and debugging of user programs.

8-7.2.1 Paper Tape Input/Output System (PTIOS)

A limited simulation of the 706 Monitor I/O interface is available with the Paper Tape Input/Output System (PTIOS). Programs written to operate with PTIOS will also operate in the 706 Monitor environment. PTIOS was developed to support the operation of Conversational FORTRAN. Appendix K-7 gives the pertinent characteristics of PTIOS.

PTIOS Calls

The following linkages are used for PTIOS calls:

EXIT	EQU	X'40'
DOIO	EQU	X'44'
STAT	EQU	X'46'
WEOF	EQU	X'50'
ULIM	EQU	X'5A'
ABSLOAD	EQU	X'78'

EXIT

A call to exit will transfer to ABSLOAD to load the next binary processor.

DOIO

The DOIO call may have only one argument: FIOT. PTIOS DOIO cannot stuff a new buffer or word count address.

STAT

The STAT call is a no-op, since PTIOS I/O is performed on a hang up and wait basis. The call may be included for compatibility when running in the 706 monitor environment. The call may have the form:

```
STAT FIOT
or
STAT FIOT,ERR
or
JSX STAT,FIOT If SMB is not needed
or
JSX STAT,FIOT,ERR
```

Do not use the form:

```
STAT
D FIOT
D ERR
```

Since PTIOT depends upon the sign bit or ERR being true.

WEOF

The WEOF call will write a file mark on the punch or teletype. The file mark is an ASCII BELL character (X'87'). PTIOS will precede it with the logical unit number punched in binary and follow it with a C/R. These extraneous characters will have no effect upon the reading of this file mark, either by PTIOS or the 706 Monitor I/O Driver. The call is:

```
WEOF FIOT
or
JSX WEOF,FIOT
```

ULIM

Cell X'5A' contains the address of the last cell of core memory. This is computed and stuffed

by the PTIOS initialization routine. If it is desired for any reason to protect some portion of upper core from CONFORT or other processors, this cell may be stuffed with the last available cell. Note that in the 706 monitor environment cell X'5A' contains the highest available cell of non-resident memory.

ABSLOAD

Cell X'78' is the link point to the ABSOLUTE loader. Any program acceptable to the 706 monitor absolute loader is acceptable to the PTIOS absolute loader. The program will be loaded, and the loader will transfer to the specified execution address.

INTERRUPT LINKAGES

PTIOS occupies cells 0-7F, exclusive of the interrupt linkage cells for interrupts 1, 2, and 3. Thus PTIOS operates in the area reserved for system linkages in the 706 operating systems. These interrupt cells may be stuffed with interrupt linkages, at the users discretion.

Interrupt 1	Cell 4	PCR save
	5	Interrupt link
	6	Status save
Interrupt 2	Cell 8	PCR save
	9	Interrupt link
	A	Status save
Interrupt 3	Cell C	PCR save
	D	Interrupt link
	E	Status save

Every other location in the range 0-7F is used by PTIOS.

8-7.3 STANDARD Operating System

The STANDARD Operating System (SOS) for the Raytheon 706 computer provides standard configuration system users with an expandable operating system, compatible with other 706 operating systems. See Figure 1-29.

SOS is an integrated hardware/software system which uses the automatic priority interrupt system

and powerful input/output structure of the 706 computer. The center of the system is a compact Resident Monitor, whose size ranges from 750 to 1750 words, depending on the number and type of peripherals in the system.

The other major components of the standard system are a relocatable X-RAY Executive which contains those functions necessary to control the system, the System Processors which are absolute core image programs that perform various service functions, and the System Library which includes relocatable programs and subroutines such as the Math Library.

The degree of automaticity with which the SOS operates depends upon the number and type of peripheral devices in the system.

8-7.3.1 SOS Resident Monitor

Each STANDARD Operating System has a Resident Monitor tailored to the hardware configuration. The Resident Monitor is primarily a file-oriented, device independent, input/output system which allows the computer to be time-shared in a real-time interrupt environment. The input/output Software (IOS) features device reassignment, simultaneous I/O capability, centralized I/O monitor, end-action, and IOS macros. IOS provides a common I/O system for all system configurations and eliminates the programmer's need to understand the machine language I/O operation of the computer, still providing full utilization of the computer's I/O structure. The user need only become acquainted with the I/O macros OPEN, DOIO, and STAT to communicate with any standard peripheral.

Device independence allows different devices to be substituted for I/O operation at execution time without requiring reassembly or compilation of the program. For example, a program could be debugged by using a small setup test data input from the teletype. During production runs, the input can be changed to magnetic tapes to process the active files.

The end-action feature permits program operation to be synchronized to I/O device operation. It

allows sharing of time, otherwise wasted during I/O operations, with other tasks. A program can initiate an I/O operation, continue computing, and request other I/O operations. Then, when the first I/O operation is complete, the system will return to the program at its specified end-action address. It can start another I/O operation then exit. The system will restore the interrupt. End action may also be used to operate a double buffered I/O function.

The Resident Monitor resides at the top of memory and occupies the upper 750 to 1750 locations depending on the number and type of peripherals in the system. In addition to IOS, service routines for interrupt processing and operator recovery are provided within the Resident Monitor and are completely re-entrant.

8-7.3.2 SOS System Library

Since the SOS is essentially a manually operated system, the System Library is maintained as separate decks or paper tapes. If there is one magnetic tape in the system (except 7-track DIO tape unit), the library of relocatable programs may be kept on a magnetic tape, since the loader is equipped to perform a serial library scan. If desired, a paper tape or card library can also be scanned for library loading. However, it may be more convenient to maintain the library as individual decks or paper tapes, and load only those which are actually required to execute a particular program. The decision as to library maintenance must be made at each installation and will depend upon such criteria as operator experience, whether the operation is open or closed shop, and the type of programming task being implemented.

The 706 Program Library contains programs which may be in absolute or relocatable format. Most major system programs, such as; the Resident Monitor, FORTRAN IV, SYM II, etc., are loaded as absolute programs.

The relocatable library contains subroutines and programs whose locations in memory are determined by the system loader at load time. The system loader processes the text at load time in order to convert the program to core image format in the locations in which it is loaded.

8-7.3.3 SOS X-RAY Executive

The STANDARD X-RAY Executive is a non-resident processor which controls the system configuration and manages the execution of program tasks. A comprehensive set of control directives permit the operator to make I/O device assignments, load programs, and specify system parameters. STANDARD X-RAY is a relocatable program loaded by the loader.

If there is sufficient core space, STANDARD X-RAY may be made resident, which provides the user a convenient teletype operated control package. On the other hand, most of the system functions can be accomplished by manual operations. For instance any absolute program can be loaded by setting the program counter to X'78', and I/O assignments can be made by manually storing data directly into the Peripheral Equipment Assignment Table (PEAT). STANDARD X-RAY, however, provides a systematic way by which the operator or user may control the system.

8-7.4 Magnetic Tape Operating System

The Magnetic Tape Operating System (MTOS) economically assembles, compiles, and executes programs or a group of programs automatically. Jobs to be processed are stacked and separated by control records that direct system operation. Jobs may be processed individually under operator control.

MTOS consists of a Resident Monitor, an Executive (X-RAY), a Queue Processor Loader (MQLP) and an open ended set of assemblers, compilers, and subroutine libraries. The center of the system is a compact Resident Monitor, whose size ranges from 1024 to 2048 words of core memory, depending on the number and type of peripherals in the system.

To utilize the powerful input/output structure of the 700 series computer, resident, re-entrant, device independent input/output software (IOS) eliminates the user's need to understand the machine language I/O operation of the computer, while still providing the simultaneous I/O-computation capability.

The Queue/Loader Processor maintains a system queue with 256 priority levels whereby tasks may be injected into the queue in random order but executed in strict priority sequence. The X-RAY Executive controls the system configuration and manages the execution of program tasks. X-RAY also facilitates automatic batch processing of jobs. Over 30 control directives allow the operator to assign I/O devices, add or delete tasks in the system queue, and control the execution of programs.

The organization and operation of the Magnetic Tape Operating System are described in detail in this manual.

8-7.4.1 MTOS Resident Monitor

The Resident Monitor is tailored to the hardware configuration. The Resident Monitor is primarily a file-oriented, device-independent input/output system. The Input/Output Software (IOS) features device reassignment, simultaneous I/O capability, centralized I/O monitor, and IOS macros. IOS provides a common I/O system for all system configurations and eliminates the programmer's need to understand the machine language I/O operation of the computer, still providing full utilization of the computer's I/O structure. The user need only become acquainted with the I/O macros OPEN, DOIO, and STAT to communicate with any standard peripheral.

Device independence allows different physical devices to be substituted for I/O operation at execution time without requiring re-assembly or compilation of the program. For example, a program could be debugged by using a small setup test data input card reader. During production runs, the input can be changed to magnetic tape to process the active data files.

The Resident Monitor resides at the top of memory and occupies the upper 1000 to 2048 locations depending on the number and type of peripherals in the system. In addition to IOS, service routines for queue maintenance, interrupt processing, operator interrupt, and operator recovery are provided within the Resident Monitor.

8-7.4.2 MTOS System Library

The Magnetic Tape Operating System consists of two libraries maintained on the system tape: the processor library and the relocatable library. The processor library contains only absolute programs in core image format, while the relocatable library contains programs and subroutines whose locations in memory are determined by the system loader at load time.

The processor library contains frequently used programs requiring rapid loading. Once a processor entry is found on the system tape, the entry is loaded with a single read operation. This is possible because the program is in core image format. Most major programs, such as: Resident Monitor, X-RAY Executive, FORTRAN IV, SYM II, etc., are loaded in the processor library. Various application programs requiring rapid loading may also be located in the processor library.

The relocatable library contains subroutines and programs whose locations in memory are determined by the system loader at load time. These programs and subroutines are stored on tape in relocatable loader text format. The system loader processes the text at load time, in order to convert the program to core image format in the locations in which it is loaded. The FORTRAN IV library and MATH Package are in the relocatable library.

8-7.4.3 MTOS Queue/Loader Processor

The MTOS system sequences automatically from task to task under control of a list of tasks known as the system queue. The queue is organized around a 256-level priority ranking scheme whereby tasks may be injected into the queue in random order, but executed in strict priority sequence. If two or more tasks are queued at the same priority level, they will be executed in chronological order (first-in, first-out).

The queue processor program, and the relocating loader are combined to form the Queue/Loader Processor (MQLP). MQLP is a non-resident processor which constructs and maintains the system queue in memory and loads all programs from the library tape. Between tasks the Queue/Loader

Processor updates the system queue and initiates execution of the highest priority task.

8-7.4.5 MTOS X-RAY Executive

The X-RAY Executive is a non-resident processor which controls the system configuration and manages the execution of program tasks through a system queue with 256 priority levels.

X-RAY is operated the same as any other task in the system queue. When the system is bootstrapped from tape, X-RAY is automatically injected into the queue. At all other times it is maintained in the queue by itself, scheduling its own appearance at the completion of each job request.

8-7.4.6 MTOS System Configurations

The minimum configuration for MTOS operation is shown in Figure 8-25. Conversational FORTRAN may be operated in the compile and go mode in this configuration. Note that a high speed paper tape reader may be substituted for the card reader. However, batch operations are required to punch the control directives on paper tape. Note also that Real-Time FORTRAN IV *cannot* normally operate in the minimum MTOS system, because the MTOS monitor is about 500 locations larger than a Standard System Monitor.

The expanded configuration shown in Figure 8-26 is recommended for batch FORTRAN IV compile and go operations.

FORTRAN IV will operate with fewer tape drives, but only with manual intervention. For example with only two tape drives, the MTOS system library tape must be dismounted to output the program on magnetic tape. Otherwise, it would be output on the ASR33 paper-tape punch.

8-7.5 Real-Time Operating System

The Real-Time Operating System (RTOS) for the Raytheon 706 computer is a disc based operating system that permits rapid response to real-time events or external requests, provides greater system

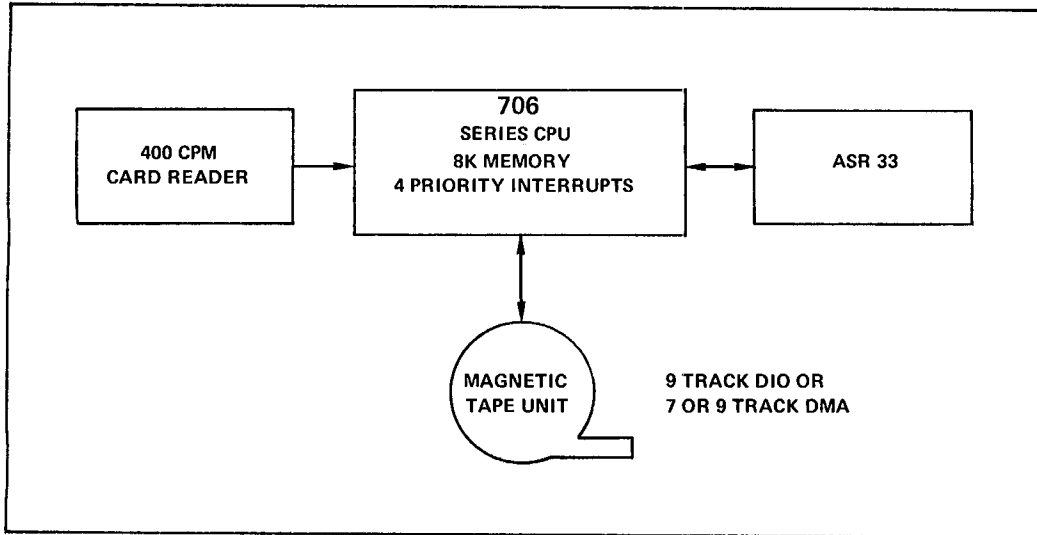


Figure 8-25. MTOS Minimum Configuration

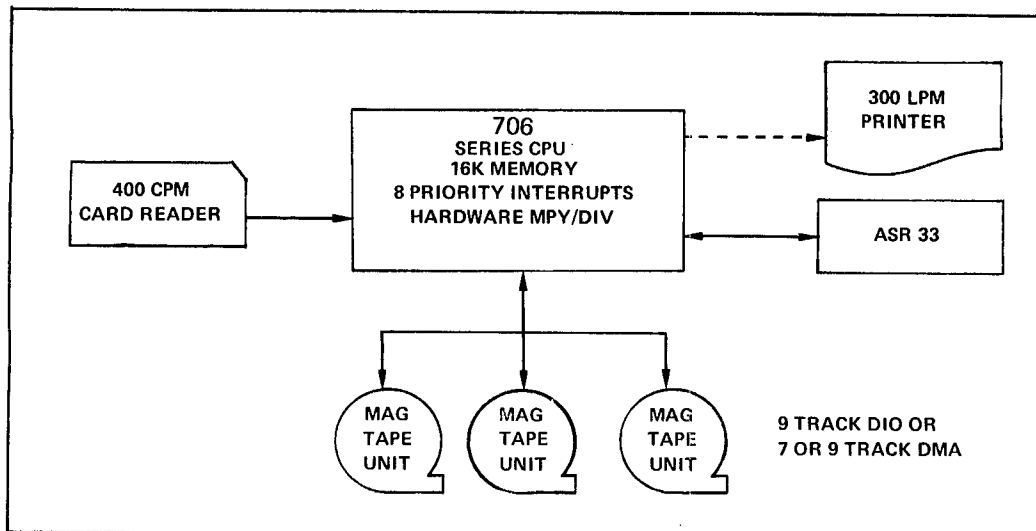


Figure 8-26. MTOS Expanded Configuration

throughput, and better utilization of machine time. With RTOS one can have several high priority real-time tasks executing in the foreground while a batch of compilations, assemblies, and other lower priority programs are automatically executed in the background on a time-available basis. For example, the computer system may be acquiring data from several scientific experiments in the foreground, while a FORTRAN IV program is performing calculations on the previously acquired data in the background.

RTOS is an integrated hardware/software system using a fast, fixed head, disc storage; the automatic priority interrupt system; and the powerful input/output structure of the 706 computer. The center of the system is a compact Resident Monitor, ranging in size from 1000 to 2048 words, depending on the number and type of peripherals in the system. The Resident Monitor includes a file-oriented, device independent, input/output software; as well as service routines for queue maintenance, interrupt processing, operator interrupt, and operator recovery.

RTOS includes a disc-based library organized for rapid response loading of both absolute programs (about 50 MS) and relocatable programs (up to 3000 words per second on 706 systems). System Generator programs provide automatic system generation while a library extension processor permits programs to be replaced or added to the library. The system also contains a disc user area which is divided into four logical files whose sizes and relative locations can be dynamically allocated by X-RAY directives.

The Queue/Loader Processor maintains a system queue with 256 priority levels whereby tasks may be injected into the queue in random order but executed in strict priority sequence. The Real-Time X-RAY Executive controls the system configuration and manages the execution of program tasks. Real-Time X-RAY facilitates automatic batch processing of jobs, concurrent with priority interrupt connected real-time tasks. With over 40 control directives, the operator may make I/O device assignments, add or delete tasks in the system queue, reconfigure the resident portion of the

system, connect or disconnect tasks to interrupts, and control the execution of programs. The ability to connect programs to interrupts and define resident programs, subroutines, and data blocks provides the user with a flexible means of configuring his real-time system.

8-7.5.1 RTOS Resident Monitor

Each Real-Time Operating System has a Resident Monitor tailored to the hardware configuration. The Resident Monitor is primarily a file-oriented, device independent, input/output system which allows the computer to be time-shared in a real-time interrupt environment. The Input/Output Software (IOS) features device reassignment, simultaneous I/O capability, centralized I/O monitor, end-action, and IOS macros. IOS provides a common I/O system for all system configurations and eliminates the programmer's need to understand the machine language I/O operation of the computer, still providing full utilization of the computer's I/O structure. The user need only become acquainted with the I/O macros OPEN, DOIO, and STAT to communicate with any standard peripheral.

Device independence allows different physical devices to be substituted for I/O operation at execution time without requiring re-assembly or compilation of the program. For example, a program could be debugged by using a small set up test data input from the teletype. During production runs, the input can be changed to the disc to process the active data files.

The end-action feature permits program operation to be synchronized to I/O device operation. It allows sharing of time, otherwise wasted during I/O operations, with other tasks. A core resident program can initiate an I/O operation then exit the system to allow it to initiate another task. Then, when the I/O operation is complete, the system will return to the resident program at its specified end-action address. It can start another I/O operation then exit. The system will restore the interrupted task. End-action may also be used within a single task; for example, to operate a double buffered I/O function.

The Resident Monitor resides at the top of memory and occupies the upper 1000 to 2048 locations depending on the number and type of peripherals in the system. In addition to IOS, service routines for queue maintenance, interrupt processing, operator interrupt, and operator recovery are provided within the Resident Monitor.

8-7.5.2 RTOS Disc Based Library

The Real-Time Operating System uses two libraries maintained on the disc; the processor library and the relocatable library. The processor library contains only absolute programs which are in core image format, while the relocatable library contains those programs and subroutines whose locations in memory are determined by the system loader at load time. Each library has its own directory which contains a list of the programs and their location on the disc. In this way programs are located directly without a time-consuming serial search process. Libraries are constructed automatically by two system generation programs; SYSGEN 1 generates the processor library, while SYSGEN 2 generates the relocatable library. The relocatable library, once created by SYSGEN 2 can be extended at any time by the library extension processor (EXTEND) which can also replace any program in the library.

The processor library contains frequently used programs which require very rapid loading. Only two disc accesses are required to load a processor. Once a processor entry is found in the directory, the program is loaded with a single direct memory access operation. This is possible because the program is in core image format. Absolute programs are loaded in about 50 milliseconds. Most major system programs, such as the Resident Monitor, the X-RAY Executive, FORTRAN IV, SYM II, are loaded in the processor library. Various application programs requiring rapid loading may also be located in the processor library.

The relocatable library contains subroutines and programs whose locations in memory are determined by the system loader at load time. These programs and subroutines are stored on the disc in relocatable loader text format. The system loader processes the text at load time, in order to convert

the program to core image format in the locations in which it is loaded. The programs are found by means of a directory. The loader text records for each program are spread around each track so that access time is minimized and loader text processing is efficiently overlapped with the disc input time. As a result, the system loader loads between 700 and 4000 words per second. Actual load times vary somewhat depending on the size of the program units and the number of externals. Small subroutines tend to load at the lower rate because of initial disc access time, while large programs tend to load at the higher figure.

Approximately one-fourth of a disc capacity is used by the Raytheon-provided software library. The remainder of the disc is divided into four relatively addressed logical files (0-3) which are available to the user. Files 0 and 1, which are used for FORTRAN IV and SYM II processors as binary object and intermediate storage, but may be used as temporary data storage. Files 2 and 3 may be used for temporary or permanent data storage. The size and relative location of the files can be dynamically allocated through X-RAY directives.

8-7.5.3 RTOS Queue/Loader Processor

The RTOS system sequences automatically from task to task under control of a list of tasks known as the system queue. The queue is organized around a 256 level priority ranking scheme whereby tasks may be injected into the queue in random order but executed in strict priority sequence. If two or more tasks are queued at the same priority level, they will be executed in chronological order (first in-first out).

The system queue, the queue processor program, and the relocating disc loader are combined to form the Queue/Loader Processor (QLP). QLP is a non-resident processor which constructs and maintains the system queue on the disc and loads all programs from the disc library system. The Resident Monitor also contains a queue buffer which contains the current task and any other queue entries made during execution of the current task. Between tasks the Queue/Loader Processor updates the system queue, resets the resident monitor queue buffer and initiates execution of the highest priority task.

8-7.5.4 RTOS X-RAY Executive

The Real-Time X-RAY Executive is a non-resident processor which controls the system configuration and manages the execution of program tasks through a system queue with 256 priority levels. A comprehensive set of control directives permit the operator or user to make I/O device assignments, demand task execution, reconfigure the resident portions of the system, and connect or disconnect tasks to interrupts.

Real-Time X-RAY is operated as is any other job in the system queue. When the system is bootstrapped from the disc, Real-Time X-RAY is automatically injected into the queue. At all other times it is maintained in the queue by itself, scheduling its own appearance at the completion of each requested job.

8-7.5.5 RTOS Residence and Interrupt Connection

In typical real-time applications in which the computer must respond to external stimuli, some tasks and subroutines are required frequently

enough to justify their retention in core at all times. Such routines are called resident routines. Four types of resident entries may be created in RTOS: resident data blocks, resident common blocks, resident subroutines, and resident tasks.

RTOS also permits the user to connect resident tasks to the hardware priority interrupt system so that the execution of a task can be initiated by an external stimulus. For example, a data acquisition program may be made resident and connected to the interrupt to which the analog-to-digital converter is connected. Thus, the data acquisition process can proceed independent of a program operating in the background, stealing time only when it is required to input another data value.

The resident data block and resident common block provide a convenient means of transmitting data between programs which may not be in memory at the same time. By defining resident subroutines, the user can eliminate repetitive loading of subroutines which may be shared by many programs. Through Real-Time X-RAY directives, the operator can dynamically connect and disconnect interrupts and reconfigure the resident area.

8-7.6 Multiprogramming System (MPS)

In its early development, the digital computer was a purely sequential mechanism in which information was acted on in sequence by the various computer parts, with each part waiting for the completion of the task of the previous part before it began its own. Consequently, the over-all utilization for many components in the system was very low, since most of their time was spent waiting rather than computing. The effective problem-solving speed of the system became largely a function of the speed of the slowest component, so that on problems where the slow components were frequently used, the achievable computing speeds were disturbingly low.

One way to increase both efficiency and effective problem-solving speed is to *time-share* the faster portions of the computer among several slower ones and to permit the latter to operate concurrently. This technique was first exploited with respect to the input-output sections, since the most glaring disparities lay between the characteristically slow speeds of the input/output equipment and the megacycle rates of the computer arithmetic unit.

Multiprogramming means executing multiple routines or programs concurrently, by an interleaving method, on a machine with one processor. This form of processing system is, in general, much more complex than a simple batch processing system.

Even a simple batch processing system requires some form of operating system. In a multiprogramming environment, it becomes imperative that all processing be controlled by a comprehensive operating system. All control operations such as job scheduling, interrupt handling, and I/O servicing of the numerous demands of several programs is likely to be completely chaotic.

The primary function of the multiprogramming system is to increase system throughput, and make better use of available machine time. For example, when a program running in a batch operating system must perform an input/output operation, the computer is idle until the completion of that input/output operation. This waste can be a

significant portion of the available machine time if any great amount of input/output is done. Delays in input/output operations can be because of manual intervention, such as the operator having to mount tapes, or simply because input/output devices are generally very slow in relation to internal machine speeds. A simple way to avoid this loss is to shift control to another program while the first one is awaiting completion of input/output, then return control to the first program.

The multiprogramming system enables the computer to respond to external requests in a rapid manner. Such rapid response is necessary in real-time and time-sharing applications. In any batch processing system, such rapid response is not required because jobs are processed one after another, and a job is not normally halted for a more important job. However, in the MPS, if one program is running and an external request for a more important program is received, control can be transferred to the more important one, suspending execution of the preceding program. Control can then be returned to the suspended program.

8-7.6.1 Concurrent Programming

MPS makes an already powerful CPU more efficient and more self sufficient by letting it work on several tasks at the same time. System resources of core storage, CPU time, and disc storage are allocated on a priority basis so that the most important tasks are completed first. While doing lower priority tasks, the system can still respond rapidly to higher priority tasks as they are initiated.

MPS enables a programmer to segment large programs into smaller tasks that are executed at different priority levels. Each program module can be written as a simple, independent program and may be checked out separately from the total problem. As a result, complex system programs are easier to specify, write, and debug. Data acquisition can be separated from lower priority data processing. Multiple data acquisition and processing subsystems are possible because MPS provides the automatic sequencing of unrelated tasks.

8-7.6.2 System Control

The MPS Resident Monitor is a package of routines that organize and allocate the system's resources. All jobs entering the system are treated as an ordered set of smaller tasks which may be executed at one or more priority levels. Tasks may be scheduled by the operator, by another task, by an event, or on a periodic basis. Once scheduled, tasks are arranged into a priority list called the system Queue.

The highest priority task in the Queue is executed first. If a higher priority task is scheduled, the operating environment of the uncompleted task is saved. Resident programs are executed immediately. Non-resident, or transient programs are loaded from the disc. A separate load area can be allocated for foreground transient tasks. If the transient core area contains an uncompleted program, it is saved on the disc to be restored later. *Swapping out* the lower priority task is done on a fixed head disc with an average access time of 16.7 milliseconds. Swapping is done only if insufficient core memory is available to load the new task.

Two hundred and fifty six priority levels are available to both the background and the foreground. Tasks scheduled at the same priority level are executed chronologically.

All programs, including both the batch processor and foreground executives, are serviced through the system Queue. This arrangement allows the user to modify or replace the system executives and completely restructure the system's operating characteristics while still utilizing the scheduling, input/output, and system generation features of MPS.

8-7.6.3 System Protection

Dynamic protection for both core and disc resident programs is accomplished by hardware memory protect. In addition the monitor checks I/O requests for interference and dedication of magnetic tapes, line printers, etc. This feature insures the integrity of concurrent data acquisition and processing.

Core memory (Figure 8-27) is protected by two limit registers which bracket the active task area to prevent inadvertent transfer into unrelated task and monitor areas. Execution of any input/output, halt, or undecoded instruction in a user task initiates the protect interrupt. The limit registers are adjusted by the monitor according to the changing conditions in the system. The resident programs and the monitor are intact regardless of what happens in the background.

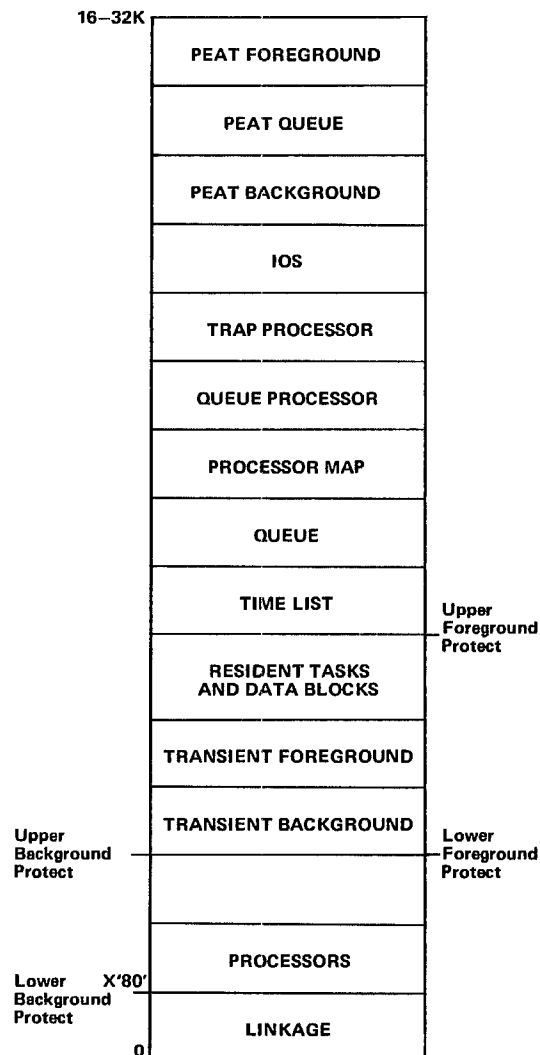


Figure 8-27. MPS Memory Layout

The disc is protected at two levels. First, the lower tracks of the disc which contain the system can be hardware protected. Secondly, the monitor prevents attempts to write on system program and data areas even if the hardware protect is not set. Similarly, background and foreground disc areas are individually protected.

8-7.6.4 Input/Output System

Working with large amounts of data on magnetic tape, paper tape, cards, etc., the programmer need not concern himself with the physical characteristics of input/output devices and the complex coding required to control them. MPS provides all necessary routines to operate all devices supported by the system. More important, a standard method of communication to operate each device allows the programmer to reassign devices without modifying his program. Input/output is accomplished concurrently with computing to increase total system efficiency.

The Input/Output System uses minimum core space and is tailored to each users system configuration. Integrated hardware/software design has enabled the I/O device to share control routines. For example, the disc and magnetic tape share identical I/O routines. Low speed I/O devices are resident monitor buffered to allow faster swapping.

8-7.6.5 Batch Processor

To realize the full potential of the 706 and

minimize idle time, a rapid means of communicating operator and programmer instructions to the computer is provided. Through a comprehensive job control language, the programmer writes his instructions to the computer, not the operator. The batch executive, EXECB, translates the instructions into action automatically and reduces the chances of error and the number of re-runs.

The batch executive enables the computer to process a series of jobs with little or no operator intervention. At the same time, high priority foreground tasks are protected and allowed to execute on a demand basis. The time not required by foreground tasks is made available to the background batch stack. If the core memory containing the background job is required by the foreground, the background environment is saved on the disc and restored when the foreground releases the core space.

The background may be controlled manually, via the teletype, if a card reader or magnetic tape unit is not available for the batch stack.

8-7.6.6 MPS-RTOS Comparison

The MPS is essentially a superset of the RTOS. Table 8-5. gives a comparison of some of the features of MPS and RTOS. MPS has the features of RTOS plus features for solving time-sharing problems.

Table 8-5. MPS-RTOS Comparison Chart

Feature	RTOS	MPS
<p>General</p> <p>Disc Based Library</p> <p>Foreground/Background Operation</p> <p>Dynamic Task Swapping</p> <p>Resident I/O Monitor</p> <p>Device Independence</p> <p>Device Dedication</p> <p>Disc System Generators</p> <p>Task Queuing</p> <p>Periodic Task Queuing</p> <p>Hardware Core Protection</p> <p>Hardware Disc Protection</p> <p>Disc Software Protection</p> <p>Batch Processing</p>	<p>Yes</p> <p>Yes</p> <p>No</p> <p>Yes</p> <p>Yes</p> <p>No</p> <p>Yes</p> <p>Yes</p> <p>Yes</p> <p>No</p> <p>No</p> <p>No</p> <p>Yes</p> <p>Yes</p>	<p>Yes</p> <p>Yes</p> <p>Yes</p> <p>Yes</p> <p>Yes</p> <p>Yes</p> <p>Yes</p> <p>Yes</p> <p>Yes</p> <p>Yes</p> <p>Yes</p> <p>Yes</p> <p>Yes</p>
<p>Resident Monitor</p> <p>Core Required</p> <p>PEAT Tables</p> <p>Trap Processor, Time List</p> <p>Transient Foreground/Background Areas</p>	<p>Less than 2K</p> <p>1</p> <p>No</p> <p>No</p>	<p>Less than 4K</p> <p>2</p> <p>Yes</p> <p>Yes</p>

8-8 UTILITY PROGRAMS

Utility programs assist the application programmer in editing and debugging his programs. They also provide for math functions, system generation, sort/merge of files, plus hardware test and diagnostic programs. Loader programs are normally classified as utility programs but are described separately in Section 8-5 of this manual.

8-8.1 Math Library

The Raytheon 706 Math Library is a comprehensive set of mathematical functions and subroutines available to users of all four hardware systems – BASIC, STANDARD, EXTENDED and ADVANCED.

A list of the functions and subroutines available are shown in Table 8-6. A complete summary of each routine including purpose, calling sequence, timing, and space required may be found in Appendix J.

8-8.1.1 Word Formats

The math package uses the following word formats: Single Precision Fixed Point, Double Precision Fixed Point, Single Precision Floating Point and Mid Precision Floating Point. See Figure 1-6.

NOTE: The Double Precision Floating Point format is used in FORTRAN-IV only.

The math package routines treat the single and double precision fixed point words as fractions with the binary point considered to be just to the right of the sign bit. The user must scale his calculations accordingly.

NOTE: FORTRAN-IV assumes all words stored in the fixed point formats to be integers. In FORTRAN-IV numbers with fractional values are stored in one of the three floating point formats.

Table 8-7 shows examples of various word types and how they are stored internally. All floating point numbers are stored with normalized mantissas except when the number is a negative or positive power of two. The reason for this is to allow the complemented mantissas to be re-complemented in the math routines, thereby

producing the original number. An example of the non-normalized internal number representation is shown in Table 8-7 under the DPRL second example (-2), and the first example (+2) under mid precision floating point. All other floating point examples are normalized.

8-8.1.2 Math Library Storage Pool

There are two non-executable modules that are part of the math subroutines. One of these is a block of 19 locations used as storage by the various routines. Each location has a label. There are four cells used as temporary storage for the index register return locations: RET1, RET2, RET3 and RET4. There are ten locations used as temporary storage by the executable routines: TMP1, . . . , TMP9, TMP0. The three locations labelled MNT1, MNT2, MNT3 are used as a software register for Double Precision (MNT2, MNT3) and for floating point (MNT1, MNT2, MNT3).

The label MNT0 refers to the same location as MNT1. There is one word labelled OVFL that is used as a flag to indicate overflow or underflow conditions.

The second non-executable module consists of a set of labelled constants commonly used by various math subroutines. Examples are:

D1 which is a decimal 1,
BO which is bit zero of a 16-bit word
M15R which is a 15-bit mask, right adjusted

8-8.1.3 Calling Math Subroutines

The calling sequence for any of the math subroutines or functions follows the typical following format:

Location	OP Code	Operand
L-1	SMB	Subroutine
L	JSX	Subroutine
L+1	D	Argument (Optional depending on routine)
L+2	Return	

Table 8-6. Math Library Functions and Subroutine Part Numbers

Multiply		Divide	
SP Multiply	(390663)	SP Divide	(390665)
SP Cumulative Multiply	(391101)	MP Floating Divide	(391096)
MP Floating Multiply	(391096)	DP Fixed Divide	(391096)
DP Fixed Multiply	(391096)		
Load, Store		Add, Subtract	
DP Fixed Point Load	(391079)	DP Fixed Point Add	(391083)
DP Fixed Point Store	(391081)	DP Fixed Point Subtract	(391083)
MP Floating Load	(391075)	MP Floating Add	(391090)
MP Floating Store	(391077)	MP Floating Subtract	(391090)
Shift, Normalize		Overflow, Underflow	
Double Shift Arithmetic	(391085)	MP Floating Underflow	(390014)
Double Shift Magnitude	(390017)	MP Floating Overflow	(390015)
MP Floating Normalize	(391090)		
Conversion, Compare		Math (MP Floating Point)	
DP Fixed Point Two's Complement	(390664)	Square Root (39006)	(390006)
Convert Fixed Point to MP Floating Point	(391094)	Sin and Cos	(390007)
Convert MP Floating Point to DP Fixed Point	(392338)	Arc Tangent	(390010)
Convert MP Floating Point to SP Fixed Point	(392339)	Exponential	(390009)
MP Floating Compare	(391088)	Natural Log	(390008)
DP Fixed Compare	(391088)	Hyperbolic Tangent	(392279)
		Polynomial	(390016)
SP — Single precision			
MP — Mid-Precision			
DP — Double Precision			

Table 8-7 External/Internal Word Formats

		Internal (Base 16)	
Single Precision Fixed Point (1 Word)			
DATA	+1	0001	
DATA	-100	FF9C	
DATA	32767 (max. value)	7FFF	
DATA	X'7FFF'	7FFF	
D	-32767	8001	
Double Precision Fixed Point (2 Words)			
DPI	-123456789	F148	32EB
DPI	1073741823 (max. value)	7FFF	7FFF
DPI	+32768	0001	0000
Single Precision Floating Point (2 Words)			
REAL	+2487.333E+2	A992	7973
REAL	+2	0082	4000
REAL	27.4E2	008C	55A0
Mid Precision Floating Point (3 Words)			
EPRL	+2	0082	4000 0000
EPRL	-123456.789E-9	0074	BF45 7483
Double Precision Floating Point (4 Words)			
DPRL	-15	0084	8800 0000 0000
DPRL	-2	0082	C000 0000 0000
DPRL	999.9999876E12	00B2	71AF 6A40 4A54

Where SUBROUTINE is a three or four letter mnemonic identifying the math routine, ARGUMENT is an optional cell(s) containing an operand(s) followed by the RETURN location. Normally, the operand(s) must be placed in the software registers (MNT1, MNT2, MNT3) before calling the math routine. The result is normally found in the software registers.

Table 8-8 gives an example using several math library routines to add two mid precision floating point numbers, take the square root of the sum and convert the answer to a single precision fixed point number.

8-8.1.4 Loading the Math Subroutines

When a SYM I or SYM II program references one of the math subroutines, an external reference links the object text to the math library routine. Only those routines called are stored at load time along with the math pool.

8-8.2 Editors

There are three file management and update programs (editors) offered with the 706 computer. The Symbolic Program Editor – BASIC is used with paper tape systems to update *source language* paper tapes. The Symbolic Program Editor is used with STANDARD, EXTENDED, and ADVANCED systems to update *source language* statements on magnetic tape or disc. The System Editor is used to generate and modify RTOS, MTOS, and MPS system tapes. The System Editor acts on binary object text rather than source statements.

8-8.2.1 Symbolic Program Editor – BASIC

The Symbolic Program Editor – BASIC (Appendix K-8) provides a means for generating a modified version of a Raytheon 706 symbolic language program without entering the entire source program.

Once a user has determined what changes need to be made to a program he relates those changes to the editor in the form of specially formatted instructions called directives. Only the changes need to be specified – unchanged instructions will

be punched exactly as they appear in the old version of the program.

Three types of changes may be made to a program.

Statements may be inserted between existing lines.

Existing lines may be deleted.

Statements may be input to replace existing lines.

A program is modified in two phases. During the first phase, called *directive time*, all directives and new statements are input. During the second (edit) phase the old source text is read in, it is changed as specified by the directives which have been input, and a new source program is then punched.

The Symbolic Program Editor is designed to edit SYM I, SYM II or FORTRAN IV source paper tapes. Off-line tapes must be prepared per the alphabetic format prepared by SYM I/PREP.

8-8.2.2 Symbolic Program Editor

The Symbolic Program Editor does everything the BASIC version does and it will also modify source statements on the disc memory and magnetic tape.

8-8.2.3 System Editor

The System Editor (Appendix K-8) is a highly flexible editing program for generating and modifying RTOS, MTOS, and MPS system tapes. The System Editor also has the capability to copy source programs and records from one device to another, write end-of-file markers, rewind certain peripherals, duplicate binary programs and records, and generate four types of bootstraps.

Once the user determines his objective, he communicates with the editor through specially formatted instructions called directives.

The System Editor resides in the processor library section of the RTOS or MPS system and is executed by queueing.

Table 8-8. Math Library Example Program

```

1 *           MATH LIBRARY EXAMPLE PROGRAM TO
2 *           ADD TWO MID PRECISION FLOATING
3 *           POINT NUMBERS, TAKE THE SQUARE
4 *           ROOT OF THE SUM, AND CONVERT
5 *           THE ANSWER TO A SINGLE PRECISION
6 *           FIXED POINT NUMBER UNSCALED,
7 *
8 *
9 *
0000 0085 10 ALPHA   EPRL 16,0
0001 4000
0002 0000
0003 0084 11 BETA   EPRL  9,0
0004 4800
0005 0000
0006 0000 12 ANSWER  D    0
13 *
14 *
* 0007 07FF 15 START  SMB   FLD
* 0008 2007 16        JSX   FLD      MNT1,2,3 * (ALPHA)
0009 0000 17        D     ALPHA
* 000A 07FF 18        SMB   FAD
000B 200A 19        JSX   FAD      MNT1,2,3 * (ALPHA) + (BETA)
000C 0003 20        D     BETA
* 000D 07FF 21        SMB   MSQR
* 000E 200D 22        JSX   MSQR    MNT1,2,3 * SQRT((ALPHA) + (BETA))
* 000F 37FF 23        D     MNT1
* 0010 07FF 24        SMB   FIX
* 0011 2010 25        JSX   FIX      FP RESULT UNSCALED TO ANSWER
0012 0006 26        D     ANSWER
0013 0000 27        HLT
      0007 28        END   START

```

```

000B FAD
0011 FIX
0008 FLD
000F MNT1
000E MSQR

```

NO ERRORS

8-8.3 DEBUG and TRACE Programs

The 706 has two debugging packages: TRACE/DEBUG and MPS/DEBUG. TRACE/DEBUG operates with STANDARD and EXTENDED systems. It allows selective tracing of programs. MPS DEBUG operates with the ADVANCED configuration, allowing only debugging features.

8-8.3.1 TRACE/DEBUG

TRACE/DEBUG (Appendix K-9) is a comprehensive utility package centralizing functions which are useful for the checkout of users' programs. It includes functions to list and modify the contents of memory, punch out absolute programs, trace programs, and otherwise aid the user in the preparation, checkout, and debugging of programs.

TRACE/DEBUG is a relocatable program on the system library. The user may direct the Relocating Loader to load it by setting System Flag 5 or by referencing DEBUG as an external name in any of his programs.

Trace is a debugging aid which executes the users program, printing each step as it does. The trace may be complete, or selective (partial); the trace range may be set to within specified limits; and the user may optionally halt or continue after each trace print-out. Input/output operations within the traced program proceed normally without tracing the I/O drivers themselves. All trace options are specified by the input directives and all hardware operations except DIN and DOT instructions, may be traced to the teletype. MSK or DSBO will stop tracing, and will not begin again until UNM or ENBO, respectively, are executed. If the instruction following the DXS or IXS instruction is not skipped, it is not traced. It is executed as though it were a part of the IXS or DXS instruction. Of course, programs involving critical timing will be disrupted by the trace, since execution speed drops greatly, even when the trace is not printing each instruction.

8-8.3.2 MPS DEBUG

MPS DEBUG package performs all the features of TRACE/DEBUG package except tracing.

8-8.4 SORT/MERGE Program

The SORT/MERGE program fulfills various user requirements for sorting and merging data files. The system handles fixed length records on any size in blocked and unblocked formats. Any number of sort or merge keys may be specified. The user may specify special collating sequences as well as ascending or descending sequence. Record formats, sort or merge keys, collating sequence, and input/output device assignments are specified by control card inputs.

The SORT/MERGE program is a disc based system which operates under the Real-Time Operating System (RTOS) or the Multiprogramming system (MPS). The SORT/MERGE program is organized into five separate subprograms: initialization, control, primary sort, intermediate merge, and merge.

The minimum equipment configuration for the SORT/MERGE program is as follows:

1. 8K Core Memory
2. Disc Memory (2 Files)
3. Magnetic Tape Units
 - a. 0, if the input file equals the available disc space and the output unit is the card punch.
 - b. 1, if the input file equals the available disc space and the output unit is magnetic tape.
 - c. 2, if the input file is less than twice the disc capacity and the output unit is the card punch.
 - d. 3, all other cases.

8-8.4.1 Method of Operation

An information table containing SORT/MERGE parameters and other computed control data is constructed by SORT. TABLE is used by all of the programs, and TABLE resides in file 0 of the disc.

Once initialization has been completed, SORT transfers control to CONTROL. All of the

programs, except SORT and CONTROL, are called from the library by CONTROL.

When completing its function, each program transfers control back to CONTROL.

If the MERGE option is specified, the merge program (MERGE) is called and executed.

If the SORT directive is entered, control is transferred to the primary sort program (PSORT) which reads the input file, sets up the internal list, constructs and outputs sorted *record strings* to the disc. Record strings are written onto the disc until the disc is filled.

After the sorted record strings have been recorded on the disc, the intermediate merge program (IMERGE) is called by CONTROL. If the number of strings recorded on the disc is larger than the number that can be merged per pass, IMERGE does a *disc-to-disc* merge to reduce the number of strings. The disc-to-disc merge is repeated until all of the strings can be merged with one pass. Then the remaining strings are merged and recorded onto the specified device as one sorted record string.

If the input data file has not been exhausted, the primary sort program is called back and the entire operation is repeated. If two or more secondary output magnetic tapes exist when the input file is exhausted, control is transferred to MERGE which produces one final sorted data file.

Character sort fields are encoded according to the collating sequence when the data records are read. The records are decoded as they are written onto magnetic tape or the final output device. Encoding/decoding is performed only when COLLATE has been entered.

8-8.4.2 SORT/MERGE Rates

Card records can be sorted internally as fast as cards can be read (1100 cpm). Records can be merged as fast as magnetic tape can be moved (equivalent to 750 cpm).

8-8.5 System Generation

8-8.5.1 General

The Real-Time Operating System and Multiprogramming System each have two libraries on the disc; processor and relocatable. The system generation has two phases; System Generation Phase (SYSGEN 1 and MPS SYSGEN 1) which constructs the processor library and System Generation Phase 2 (SYSGEN 2 and MPS SYSGEN 2) which constructs the relocatable library. If the system is being generated from scratch (with a blank disc), an RTOS or MPS System Installation program precedes System Generation.

Absolute programs in core image format are called *Processors* and must be put into the processor library. These programs could have been assembled absolute or processed through the linking absolute program. Most major system programs; such as, the Resident Monitor, Real-Time X-RAY, SYM II, FORTRAN IV, etc., are located in the processor library. Processors offer the advantage of very rapid loading (generally less than 50 ms) because they are in core image format and require no processing of loader text.

SYSGEN 2 constructs the relocatable program library on the disc. SYSGEN 2 accepts loader text from any input medium, writes it on the disc and constructs the disc directory which will be used by the disc loader to locate program modules to be loaded. All object text is scanned and validity checked. If desired, a listing is produced of all library names, entry names, names defined by an assembler BLK pseudo op, external names, and FORTRAN common block names referenced or defined.

8-8.5.2 Disc Organization

The disc is a high speed random access magnetic storage device upon which data is stored in individually addressable sectors, each holding 47 16-bit computer words. Physically, the disc consists of 64 tracks, each containing 128 sectors. Each track has its own fixed read/write head. The disc rotates at 1,800 rpm and, hence, the maximum access time is 33.3 ms while average access time is 16.7 ms. The disc contains 8,192

sectors for a total storage capacity of 385,034 words.

The disc is divided into two main sections, the System Library and the User Area. The System Library consists of the Processor Library and the Relocatable Library. The size of the System Library depends upon the number and size of programs which are contained within the library. If the library is generated to include only Raytheon-provided software, it will require approximately 14 tracks (1535 sectors).

A detail layout of the disc is illustrated in Figure 8-28 and Figure 8-29. Sector 0 contains the disc bootstrap program and Sector 1 contains image of the disc vector and permanent PEAT Table. The Queue Loader Processor (QLP) is the first processor in the processor library for RTOS. The QLP for MPS is core resident. The order of other processors within the library is immaterial. The resident programs symbol table immediately follows the processor library and is followed by the relocatable programs.

The disc user area is divided into four relatively addressed logical files, numbered 0–3. File 0 is used by SYM II to accumulate binary output whenever BOUT is assigned to the disc. File 1 is used as FORTRAN IV and SYM II intermediate whenever the scratch unit (SCR) is assigned to the disc. The logical files are available to the user of data storage. The size and relative location of the files can be dynamically allocated by the DF directive.

8-8.5.3 Processor Library

The processor library consists of two parts; a group of processors and a processor directory. A processor is a frequently used program which requires extremely rapid loading. These processors are stored on the disc in core image format. They are independent of each other and contain all subroutines required for operation.

Once a processor entry is found in the directory, loading is accomplished with a single direct memory access input operation by the Resident Monitor. Thus, absolute programs can be loaded in a matter of milliseconds.

8-8.5.4 Relocatable Library

The relocatable library consists of three parts; the resident program symbol table, the relocatable programs, and the relocatable program directory. The amount of space reserved for the resident program symbol table is specified during SYSGEN 2. The resident loader builds the resident program symbol table in the allocated space. The relocatable program library includes a directory which contains the names and location of the relocatable program. In this way the programs are randomly accessed and loaded, thus eliminating time-consuming serial search procedures. Also, the loader text records for each program are spread around each track so that access time is minimized and loader text processing and input are effectively overlapped.

8-8.5.5 SYSGEN 1 and SYSGEN 2 Operation

The RTOS and MPS disc processor libraries may be generated from a *cold start* (blank disc) condition by using the Initial Loader, RTOS or MPS Monitor, System Installation Routine and SYSGEN 1 in that order, and called from some other peripheral besides the disc. All of these routines, except the Initial Loader become part of the processor library so that for a future system regeneration, these four routines need not be input from a peripheral other than the disc. An exception exists when the disc processor library is destroyed and the RTOS or MPS Monitor is non-operational.

A convenient method for generating the processor library is to store the library on magnetic tape or cards and use the hardware bootstrap feature of the 706.

The relocatable library is generated by SYSGEN 2, SYSGEN 2 exists in the processor library so it can be queued to execute. At this time the remainder of the disc library is generated.

SYSGEN 1 accepts a unique set of operator directives that identify the processor name, the number of sectors to be used for the System Queue list, the maximum number of processors to be installed, and termination of library generation.

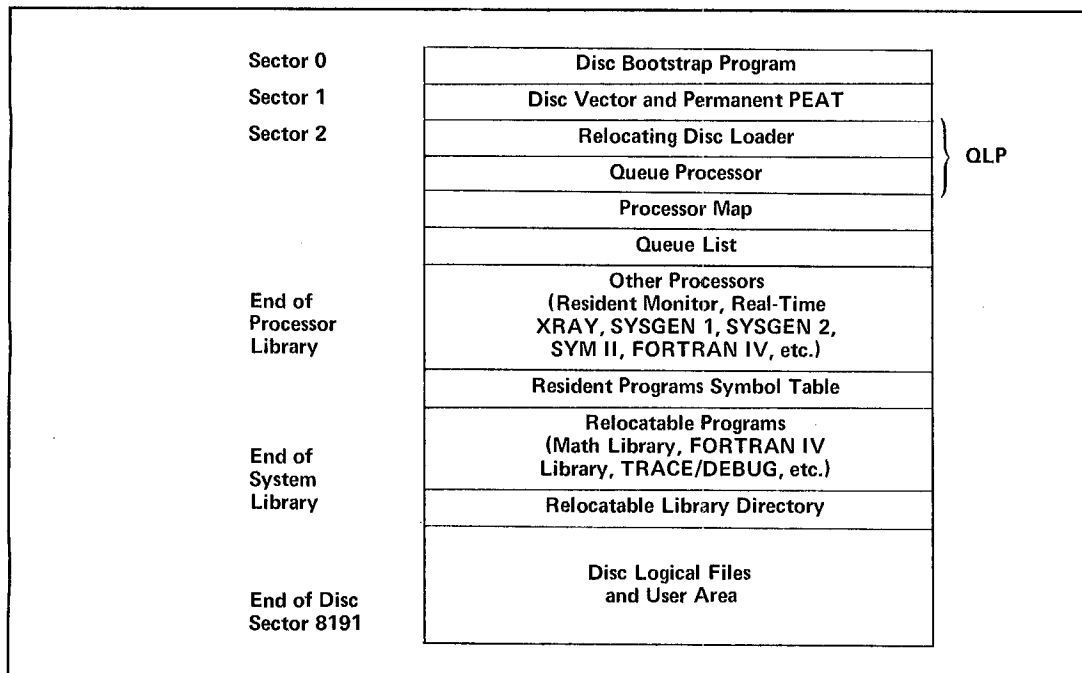


Figure 8-28. RTOS Disc Layout

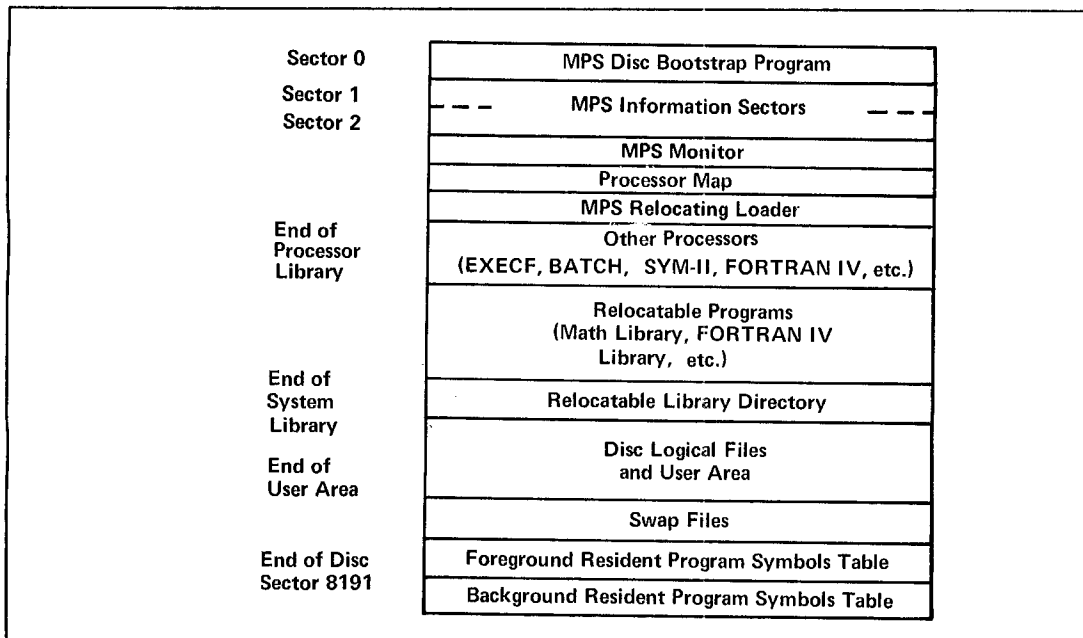


Figure 8-29. MPS Disc Layout

SYSGEN 2 options are controlled by setting system flags with an X-RAY SF directive. Options for SYSGEN 2 include: listing the library directory, list program names, list ordering errors and duplicate library names, and correct ordering errors.

Both SYSGEN 1 and SYSGEN 2 have error detection features. SYSGEN 1 can identify: seven types of ordering errors, checksum errors, I/O errors, and disc protected condition. SYSGEN 2 can identify: checksum and device error, directory overflow, invalid records, and unrecognizable loader codes.

8-8.5.6 Relocatable Library Extension

The relocatable library, once created by SYSGEN 2, can be expanded at any time by the library extension processor (EXTEND) which can also replace any program in the relocatable library. Extensions and replacements are permanent.

The extension processor replaces a program by copying the new version of the program to the disc and replacing the old directory entries with new ones. If there are new library names, they will be added. The disc storage occupied by the old version of the program is not reclaimed. For this reason replacement is used as a temporary expedient until SYSGEN 2 can be rerun with the updates included in the SYSGEN 2 input media.

8-8.6 SENSOR Diagnostic Package

8-8.6.1 General

The SENSOR Diagnostic Package is an integral part of Raytheon Computer's 706 hardware/software system and provides the user with programs and procedures for maintaining the hardware system. SENSOR enables the user to quickly determine the operational status of the system, to detect hardware failures, and isolate and repair the failures.

Because the 706 and 703 computers are software compatible, many 703 programs are used in the 706 SENSOR Diagnostic Package. However, since the 706 uses more sophisticated integrated circuits than the 703, and thus has a different hardware

configuration, additional programs are supplied to test the 706 central processor logic.

8-8.6.2 Contents of SENSOR

The number of programs contained in the SENSOR Diagnostic Package varies according to the number of peripheral devices in the system. The package includes the information required to use and interpret the results of the programs plus appropriate repair/replacement data.

SENSOR documentation is divided into 7 sections; each is briefly described below.

Section 1 familiarizes the user with the capability of the SENSOR Diagnostic Package and shows flowcharts of the overall maintenance scheme.

Section 2 includes procedures for testing and diagnosing the problems basic to computer hardware operation.

Section 3 explains the operation of the SENSOR bootstrap loading and diagnostic procedure. Included are detailed instructions on the use of bootstraps and the diagnostic information needed to remedy possible failures.

Section 4 contains error detection programs, including program descriptions and user instructions. These programs diagnose hardware problems and define preventative maintenance procedures.

Section 5 is the Basic SENSOR Program. This program contains a step-by-step diagnostic of most machine instructions and gives the user the information for isolating and repairing computer failures.

Section 6 is BRAINWASH, the memory diagnostic program. It exercises comparison techniques to find the cause of any dynamic memory failure.

Section 7 contains programs for testing the operation of peripheral devices and to assist in isolating possible malfunctions.

8-8.6.3 The Diagnostic Process

The complete diagnostic and maintenance procedure is flow charted in Figure 8-30, the system checkout flow diagram. There are three entry points in the diagram: (1) computer installation, (2) computer failure, (3) routine systems check.

When the computer system is installed, computer hardware is given a preliminary checkout prior to executing SENSOR. The hardcore checkout outlined in section 2 is then performed. If the system passes all the hardcore tests continue to section 3 and execute the SENSOR bootstrap procedure. Upon successful completion of bootstrap, proceed to section 4 and run the error detection programs. Once all the programs run without error, initiate BRAINWASH, section 6. When BRAINWASH has been successfully executed, the applicable

Peripheral Equipment Test Programs can be run to complete the execution of SENSOR.

If the system has been running and experiences a computer failure, SENSOR can be entered at the abbreviated hardcore checkout, section 2. This procedure quickly verifies the correct functioning of hardcore. The rest of SENSOR is then executed as above.

For routine maintenance system checks, SENSOR can be entered at section 5.

8-8.6.4 Example Diagnostic Procedure

Figure 8-31 is a reproduction of the diagnostic detail information as it appears in the SENSOR write-up. Since the SENSOR write-up is more voluminous than this book, the detailed information is available in separate volumes.

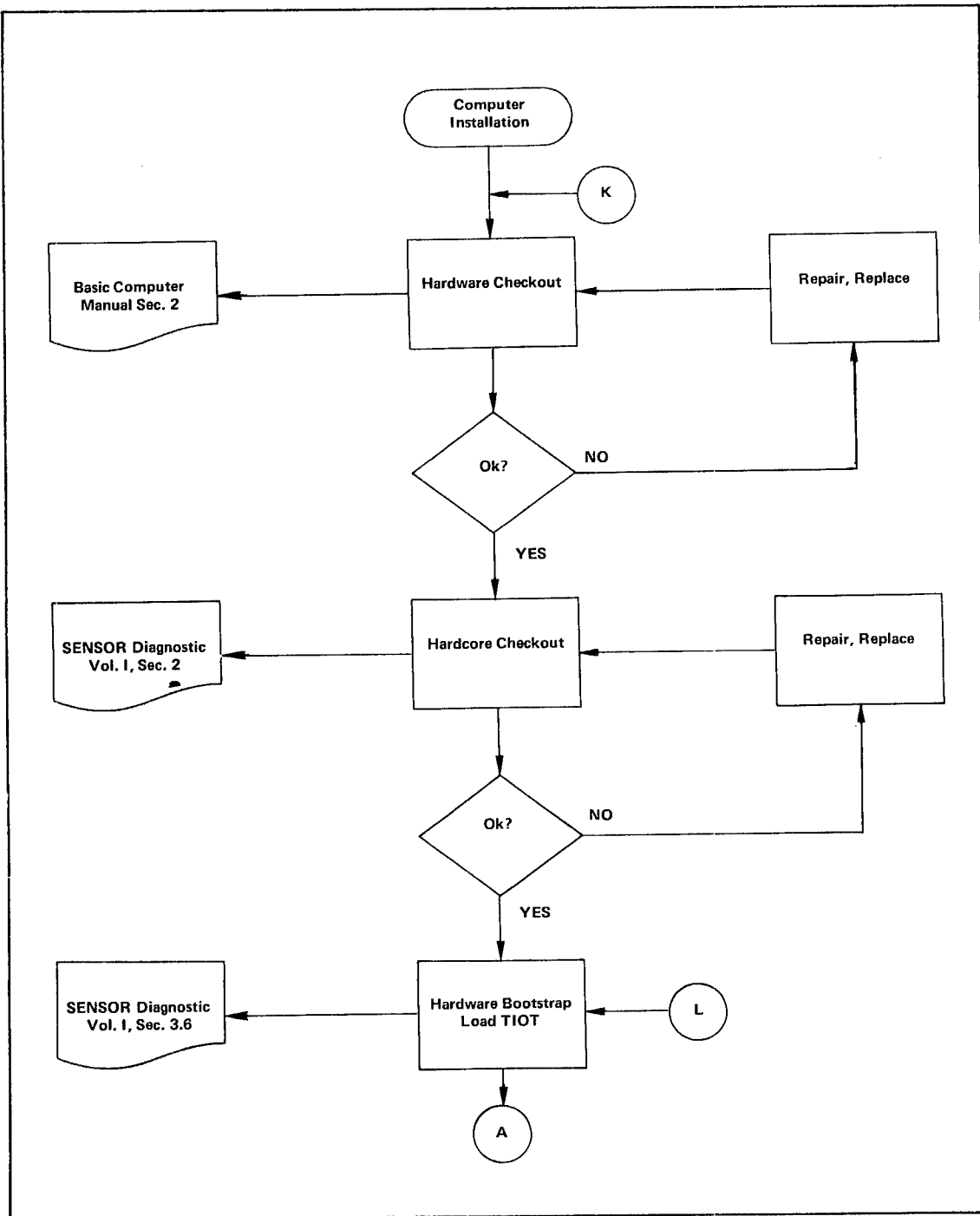


Figure 8-30. System Checkout Flow Diagram (Sheet 1 of 6)

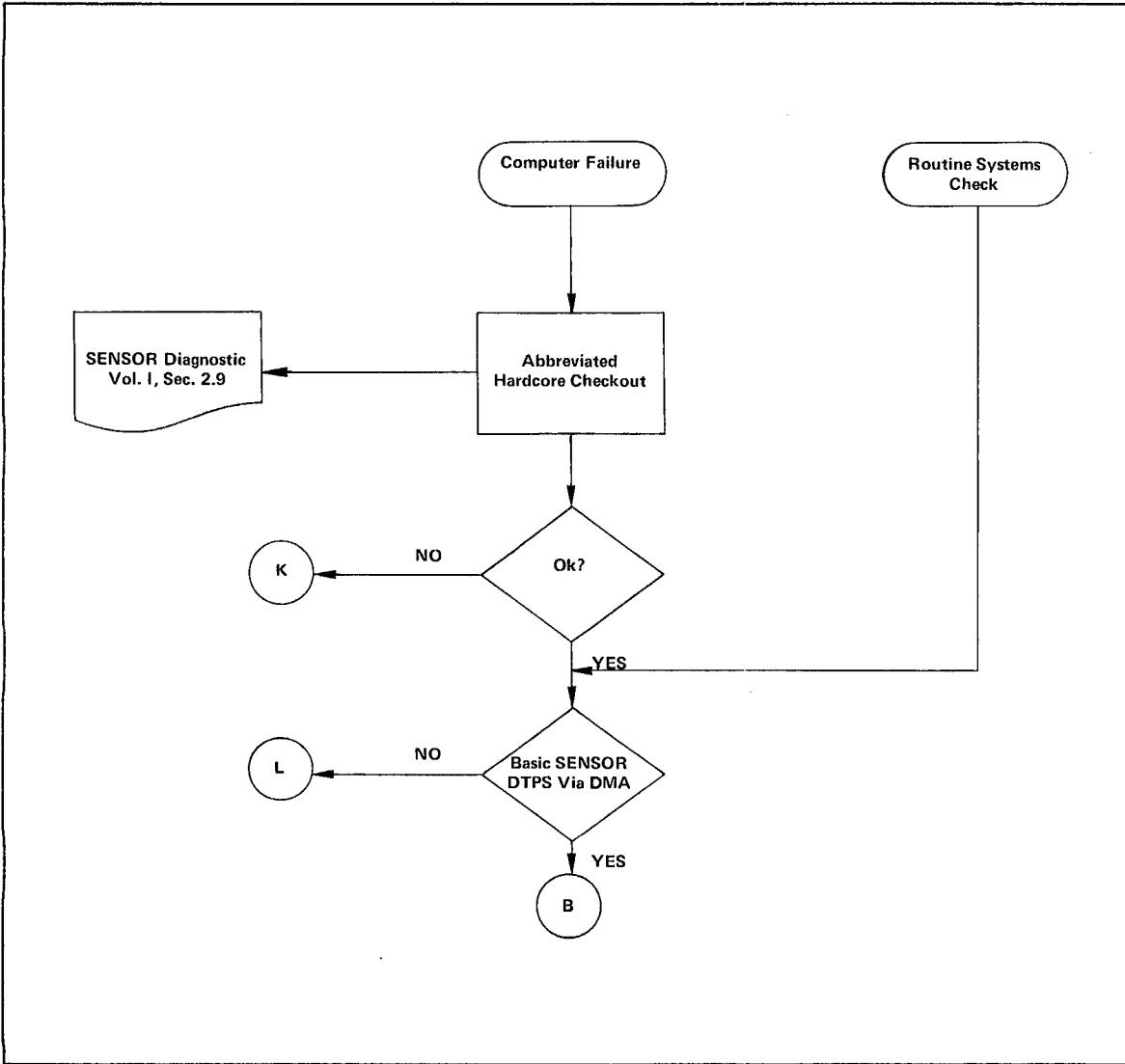


Figure 8-30. System Checkout Flow Diagram (Sheet 2 of 6)

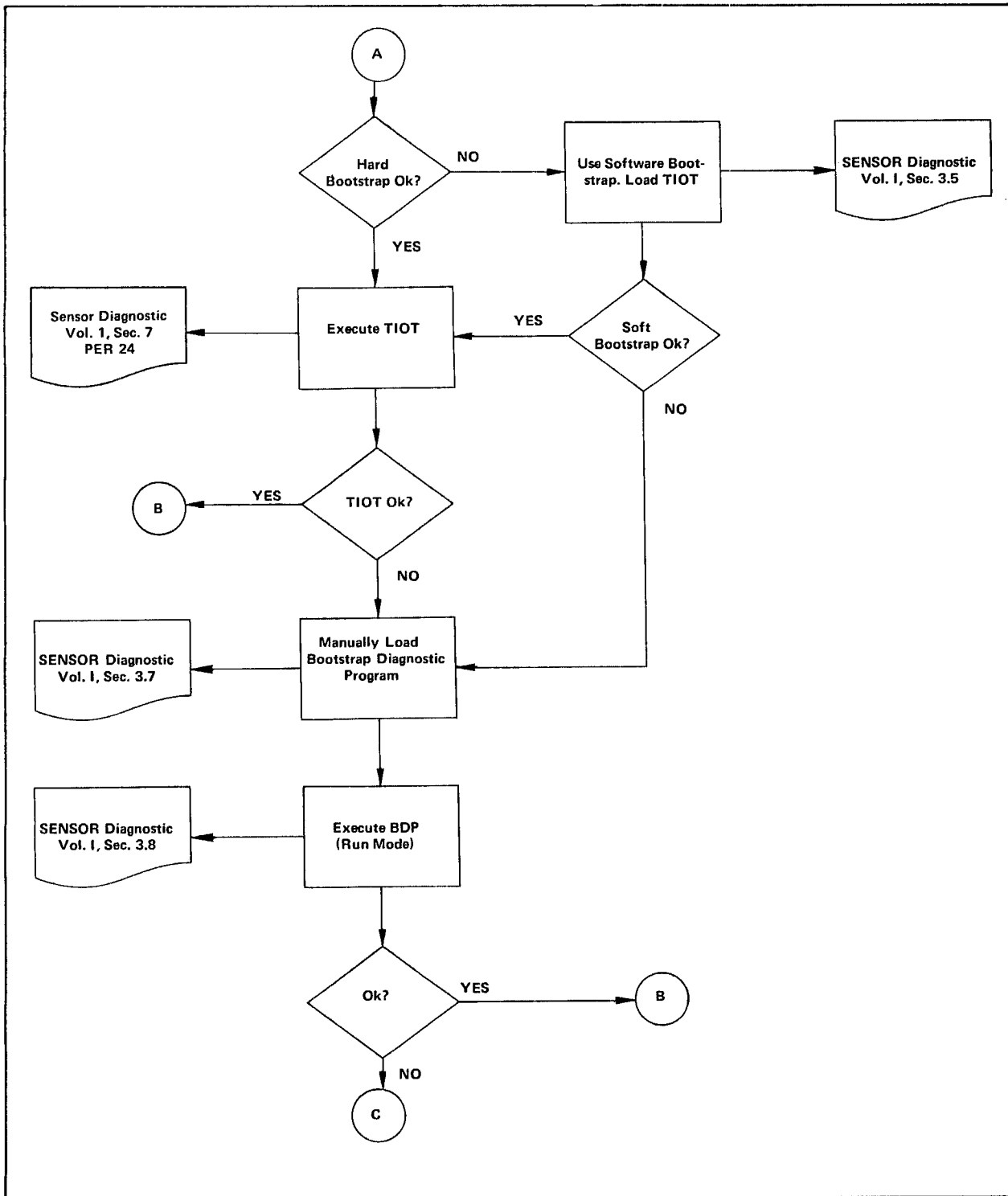


Figure 8-30. System Checkout Flow Diagram (Sheet 3 of 6)

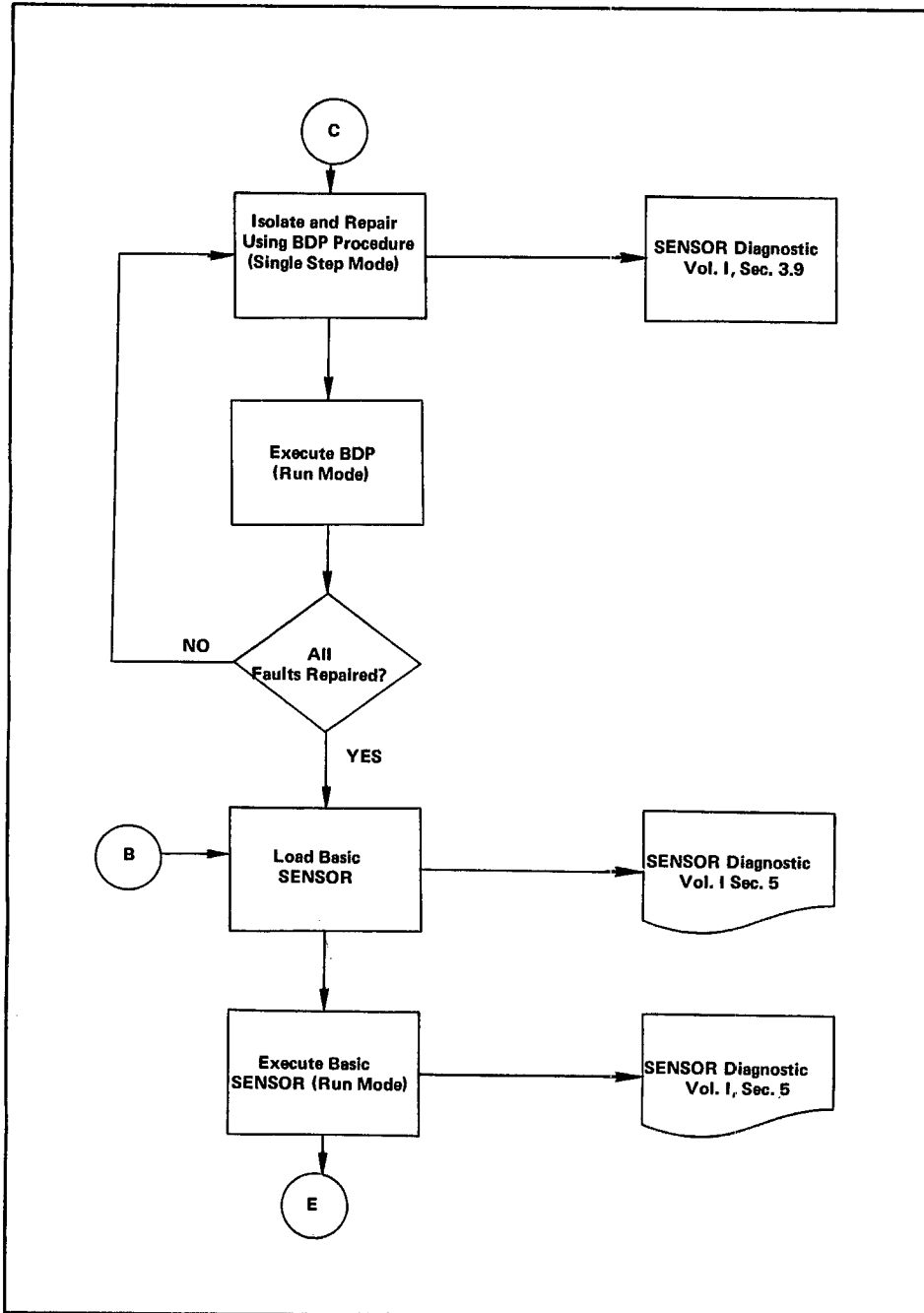


Figure 8-30. System Checkout Flow Diagram (Sheet 4 of 6)

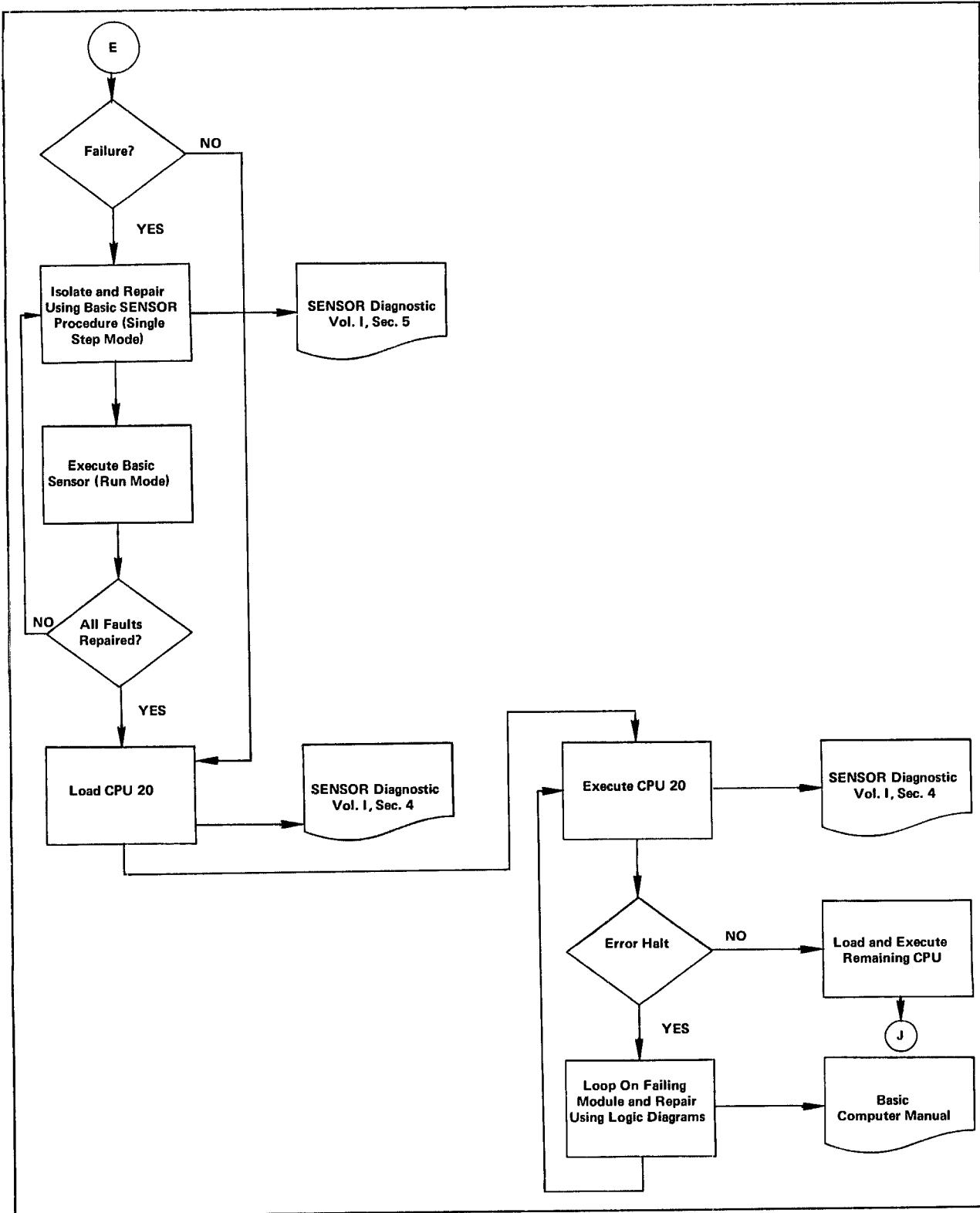


Figure 8-30. System Checkout Flow Diagram (Sheet 5 of 6)

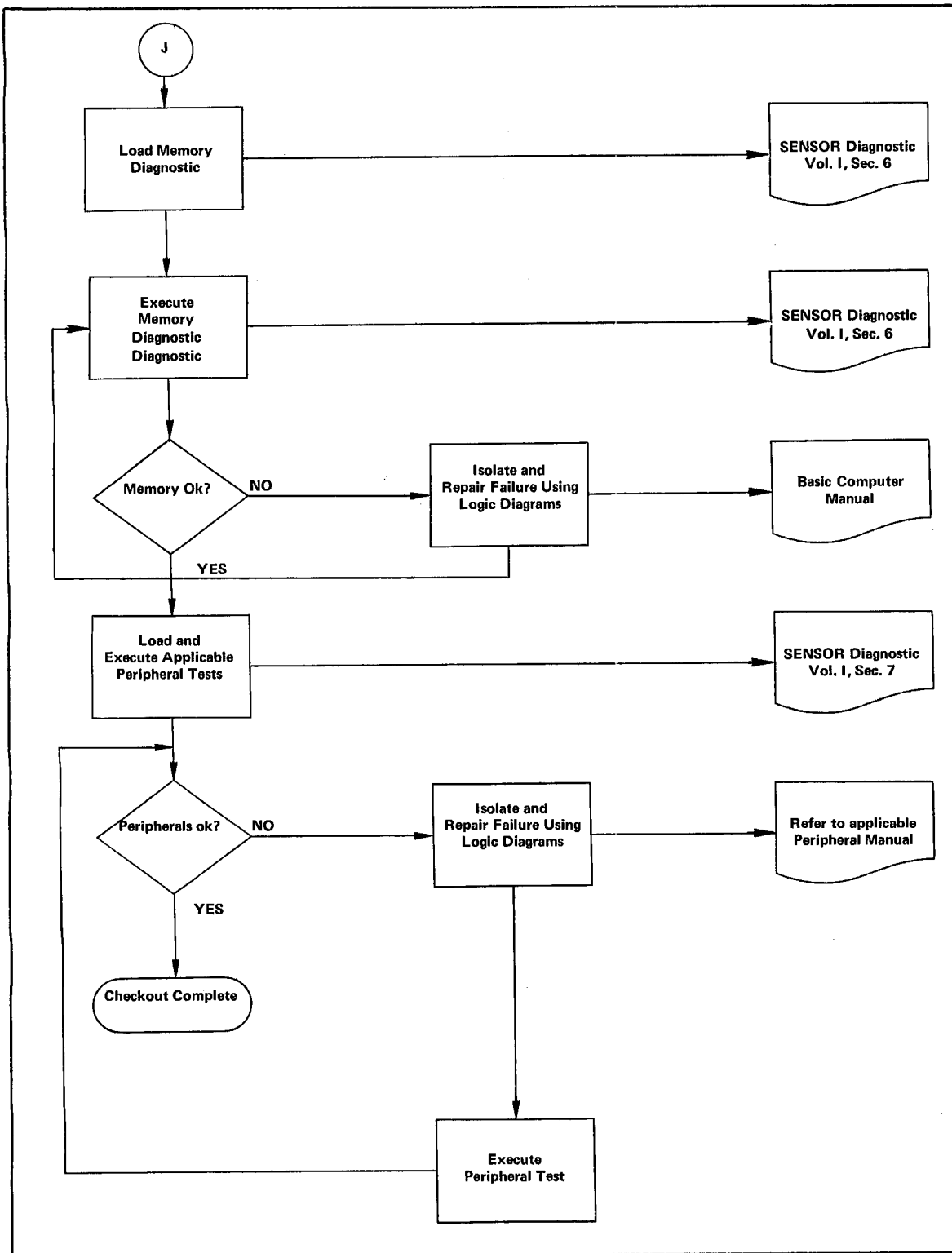


Figure 8-30. System Checkout Flow Diagram (Sheet 6 of 6)

2.2.1.3 BOOTSTRAP RESET

Bootstrap TTY light is on.	Replace (24L) (23L) (switch)
Bootstrap PTR light is on.	Replace (24L) (23L) (switch)
Bootstrap MAG light is on.	Replace (24K) (23L) (switch)
Bootstrap CRD light is on.	Replace (24K) (23L) (switch)

2.2.1.4 PROGRAM COUNTER RESET

PC will not reset.	Hold RESET button depressed, if PC still will not reset; replace (21V).
--------------------	---

Any bit in PC fails to reset.	Hold RESET button depressed, if any bit in PC is on, replace as follows:
-------------------------------	--

PC	01	(12Y)	(20W)	(10Z)
	02	(12Y)	(20U)	(10Z)
	03	(12Y)	(20U)	(15Z)
	04	(12Y)	(20S)	(15Z)
	05	(13Y)	(20S)	(16Z)
	06	(13Y)	(20R)	(16Z)
	07	(13Y)	(20R)	(21Z)
	08	(13Y)	(20K)	(13F)
	09	(14Y)	(20K)	(23Z)
	10	(14Y)	(20J)	(24T)
	11	(14Y)	(20J)	(24T)
	12	(14Y)	(20G)	(24T)
	13	(15Y)	(20G)	(24T)
	14	(15Y)	(20F)	(22Y)
	15	(15Y)	(20F)	(22Y)

2.2.2 Register Reset Tests

2.2.2.1 ACCUMULATOR REGISTER

<u>Problem</u>	<u>Solution</u>
AC will not reset	Hold RESET button depressed, if AC still will not reset, replace (21V) (26Y)
Any bit in AC fails to reset.	Hold RESET button depressed. If any bit in AC is on, replace as follows:
AC	00 (3Z) (8W) (7Z)
	01 (3Z) (8W) (4H)
	02 (3Z) (8V) (9Z)

Figure 8-31. Sensor Diagnostic Detail Example

SECTION 9

PROGRAMMING SYSTEMS OPERATION

9-1 GENERAL

This section presents the operating instructions for the five Raytheon 706 operating systems. Particular attention is paid to the programs for the BASIC hardware system configuration. The instructions are not described in detail in this section, however, the associated software manuals (see Appendix B) give complete operating instructions for system programs. This section, in conjunction with Appendix M, provides enough information to operate the system.

Raytheon Computer presently offers more than 300 programs and each program has its own part number. An explanation of the numbering scheme is included in the subsequent text.

9-1.1 Software Shipping Manifest

A *software shipping manifest* accompanies each 706 computer system. The manifest gives the part number of every piece of software shipped with the 706 computer hardware. Upon receipt of the software, the user should verify that he has received the correct documentation, paper tapes and card decks.

The remainder of this section contains partial lists of software that should be delivered with each type of computer system. The user will receive at least the minimum software for a given operating system. A sample first page of a software shipping manifest is shown in Figure 9-1.

9-1.2 Paper Tapes and Card Decks

Binary paper tape programs are shipped with every system (other than BASIC) that has a high-speed paper tape reader. Binary card decks are shipped with every system that has a card reader. BASIC systems always receive a full complement of binary paper tape programs.

9-1.2.1 Part Numbers

Tables 9-1 through 9-5 list the titles and part numbers of programs supplied by Raytheon Computer for each of the five operating systems: BASIC, STANDARD, Magnetic Tape, RTOS and MPS. Each part number is a six digit number beginning with 39 (i.e., 391234 or 390682).

On the actual software, there is a three digit *dash* number following each part number. The dash number specifies the type of software as follows:

- 006 Looseleaf documentation
- 008 Change; errata, bulletin, etc.
- 010 Binary paper tape
- 012 Binary card deck

For example: The part number 390470-010 specifies the object paper tape program for SYM-1/PREP.

9-1.2.2 Revision Letters

Normally, a revision letter is associated with every piece of software. The initial release or non-amended version of any program is assigned letter "A". Succeeding letters indicate changes to the program. The remainder of this section ignores revision level identification due to its high volatility. The operating instructions that follow are based on the revision level of each program at the time of printing of this book.

Check the software shipping manifest for the revision level of each program at the time of delivery. Any questions about program revision levels should be directed to:

Raytheon Computer
Software Support Services
2700 South Fairview Street
Santa Ana, Calif. 92704
Phone (714) 546-7160

9-1.3 Part Number by Configuration

All Raytheon Computer 706-software is divided into three categories: (See Appendix M.)

1. System
2. Core Memory
3. Software or Hardware Multiply/Divide

For instance: A Conversational FORTRAN compiler for a BASIC 4K core memory system has a different part number (393295) than the Conversational FORTRAN compiler for a BASIC 8K core memory system (394005).

In general the FORTRAN compilers are libraries and the Math Library routines are separate programs for each system determined by core memory size and the availability of the hardware multiply/divide option.

In almost all cases, the Monitor part number for a given hardware configuration is indicated on the software shipping manifest and is not shown in this manual. Each Monitor is a custom mode program that is constructed on the basis of a unique set of peripheral equipment and options.

CUSTOMER NAME:	RAYTHEON COMPANY	
ADDRESS:	BOOTH 1000 F.J.C.C. LAS VEGAS CONVENTION CENTER LAS VEGAS, NEVADA	
SALES ORDER NUMBER:	26046	HARDWARE CONFIGURATION 26046
SYSTEM NUMBER	347	CPU NO. 246
COMPILED BY:	S.M.K.	
CHECKED BY:	E.J.K.	
SHIPPED VIA:	SHIPPING	
SHIPPING DATE:	11-15-69	
REMARKS:	FIRST SIX NUMBERS IN PART NUMBER ARE THE BASIC PART NUMBER. 3 NUMBERS BEHIND DASH ARE CLASSIFICATION NUMBERS. CLASSIFICATION NUMBERS: -006 LOOSELEAF DOCUMENTATION - 3 RING NOTEBOOK TYPE -008 CHANGE: ERRATA, BULLETIN, ETC. -010 BINARY PAPER TAPES -012 BINARY CARDS	

Figure 9-1. Software Shipping Manifest Example (First Page)

**BASIC SYSTEM SOFTWARE OPERATION
SUMMARY**

Table 9-1. Basic System Software

Description	Label	Part No.	Origin/End (HEX)	Start (HEX)	Size (DEC)	Loaded-By
XRAY EXEC-BASIC ¹	XRAY	390779	0018/03B9	0040	0930	INITLOAD
Relocating Loader-BASIC	RELOADB	390682	0545/07FF	0545	0699	XRAY
SYM-I/PREP	SYM1	390470	05B0/0FF1	0800	2626	INITLOAD
Symbolic Prog. Editor-BASIC	SYMEDB	390941	0450/0858	04B5	1033	XRAY
Initial Loader-Paper Tape	INITLOAD	393260	0000/007F	0009	0128	BOOTSTRAP
Initial Loader-Cards	INITLOAD	393259	0000/007F	0009	0128	BOOTSTRAP
Paper Tape I/O System	PTIOS	393954	0080/0169	0080	0234	INITLOAD
Math Library						
Soft. MPY/DIV	(See	393325	Relocatable	(See		RELOADB
Hard. MPY/DIV	Appendix J)	393977	Relocatable	Appendix J)		RELOADB
Conver. FORTRAN Compiler						
4K Core Memory	CF	393295	0080/0C4F	0080	3024	PTIOS
8K-32K Core Memory	CF	394005	0080/0CEA	0080	3179	PTIOS
Conver. FORTRAN Library						
4K, Soft. MPY/DIV	CFR	394002	0080/0C46	0080	3015	PTIOS
4K, Hard. MPY/DIV	CFR	394001	0080/0C34	0080	2741	PTIOS
8K-32K, Soft MPY/DIV	CFR	394004	0080/0CB6	0080	2871	PTIOS
8K-32K, Hard MPY/DIV	CFR	394003	0080/0CA4	0080	3109	PTIOS
<p>1. The monitor for the BASIC system is included as part of the XRAY-EXEC program.</p>						

9-2 BASIC SYSTEM SOFTWARE

The BASIC System Software operating instructions are described in detail in this section and Appendices J, K, and M.

9-2.1 Initial Loader-Paper Tape (P/N 393260)

Initial Loader (INITLOAD) is a bootstrappable loader, which can load tapes punched in the format acceptable to the 706 monitor absolute loader. Such tapes are produced by an absolute assembly by SYM I or SYM II or by the Linking Absoluter or the Trace/Debug punch option. Tapes punched by sum error. This error should be ignored (see Loading Errors below). INITLOAD can be appended to the front end of any absolute loadable tape, including the 706 monitor tape. This combined tape will load and execute automatically under control of the hardware bootstrap switch. Loading locations and transfer address are read from the absolute tape. Checksums and tape format validity are checked by INITLOAD.

9-2.1.1 Initial Loader Operation

1. Place INITLOAD in the paper tape reader. For high-speed paper tape reader, make sure that the sprocket hole is towards the panel (away from the operator), the LOAD gate is up, and the reader power is turned on.
2. Depress Computer RESET.
3. Depress the appropriate bootstrap switch, depending on whether the device is the tele-type or high speed reader.
4. Depress RUN. INITLOAD *reads in*, occupying cells 0-7F and executes automatically, loading subsequent absolute loadable tapes.

9-2.1.2 Loading Errors

Three possible conditions can cause INITLOAD to halt: checksum error, tape found error, and successive file marks.

When a checksum error causes a halt, the record may be re-read by backspacing the program record, beginning with the LOC record and depressing

RUN. The error may be bypassed by depressing RUN.

A tape format error occurs if INITLOAD expects a one-word record (a LOC or a transfer address), and receives a multi-word record. Depressing RUN causes INITLOAD to look for another LOC.

INITLOAD halts if successive file marks are read, indicating no transfer address specified. Depressing RUN causes INITLOAD to continue loading.

9-2.2 Initial Loader – Cards (PN 393259)

Initial Loader (INITLOAD) is a bootstrappable loader, which can load card decks punched in the format acceptable to the 706 monitor's absolute loader. Such decks are produced by an absolute assembly by SYM I or SYM II or by the Linking Absoluter or the Trace/Debug punch option. Card decks punched by Trace/Debug lack checksums and will cause INITLOAD to halt for the checksum error. This error should be ignored (see Loading Errors below). INITLOAD can be appended to the front end of any absolute loadable deck, including the 706 monitor deck. This combined will load and execute automatically under control of the hardware bootstrap switch. Loading locations and transfer address are read from the absolute deck. Checksums and card format validity are checked by INITLOAD.

9-2.2.1 Initial Loader Operation

1. Place the INITLOAD in the card reader.
2. Depress Computer RESET
3. Depress RUN. INITLOAD *reads in* occupying cells 0-7F and executes automatically, loading subsequent absolute loadable decks.

9-2.2.2 Loading Errors

Three possible conditions can cause INITLOAD to halt: checksum, error, card format error and successive file marks.

When checksum error causes a halt, the record may be re-read by backspacing the program record, beginning with the LOC record and depressing

RUN. The error may be ignored by depressing RUN.

A card format error occurs if INITLOAD expects a one-word record (A LOC or a transfer address), and receives a multi-word. Depressing RUN causes INITLOAD to look for another LOC.

INITLOAD halts if successive file marks are read, indicating no transfer address specified. Depressing RUN causes INITLOAD to continue loading.

9-2.3 Initial Loader Bootstrap

With the 706 hardware bootstrap feature (see Section 7-4), there is normally no need to hand-enter an Initial Loader Bootstrap. However, with other 700 series computers or with a hardware bootstrap malfunction, it is still possible to load the Initial Loader. Execution of the Initial Loader Bootstrap program (Figure 9-2) for teletype paper tape, high-speed paper tape, or cards causes the Initial Loader to be loaded in lieu of using the hardware bootstrap feature of the 706.

LOCATION (HEX)	CONTENTS (HEX)
7F	1081
80	03E9 (03D9 for HSR, 0381 for Cards)
81	0230 (02D0 for HSR, 0280 for Cards)
82	OAC1
83	0820
84	1081
85	02ED (02DD for HSR, 0284 for Cards)
86	3800
87	0501
88	0402
89	0130
8A	107F

Press RESET, set PCR to X '80' and Press RUN

Figure 9-2. Initial Loader Bootstrap

9-2.4 XRAY EXEC – BASIC (PN 390779)

The XRAY EXEC – BASIC program is loaded by the Initial Loader (Section 9-2.1 and 9-2.2). The XRAY EXEC – BASIC includes the I/O Monitor, but not the Relocating Loader (PN 390682). Instructions for using XRAY EXEC – BASIC are in Appendix M of this manual.

9-2.5 Relocating Loader – BASIC (PN 390682)

RELOADB is used to load relocatable programs written in the SYM I programming language. Input is from paper tapes containing programs in relocatable object text.

This loader provides a convenient operating environment for loading and executing medium sized programs on a Raytheon 706 with 4K words of core and no system library facilities.

All programs loaded by this loader are relocated into the upper word page in memory. The XRAY EXEC; I/O monitor and RELOADB all reside in the lower word page in memory. Since the loader's entry names table (ENT) uses the space remaining in the lower word page, the entire upper word page is available for programs and data areas.

If more data area is needed, the space in the lower word page occupied by the loader can be used; however, if the loader is undisturbed, it is re-executable and need not be reloaded between jobs.

Although this loader is designed to accept SYM I assembler output, it will accept programs which have been written in SYM I but assembled on the SYM II assembler.

There are two restrictions on SYM II assemblies. First, the pseudo-ops DFLL, and LABL must not appear. Second, the program must have at least one reference to a relocatable address. This reference will ensure that the SYM II assembler will output 11-bit addresses for external strings rather than 15-bit addresses (RELOADB cannot process 15-bit external strings).

The relocating loader is controlled by issuing directives to the XRAY monitor which then calls RELOADB. The user gives no instructions directly to the loader.

Before any of these directives involving the relocating loader can be given, RELOADB must be loaded using the "AL" directive (absolute load).

9-2.5.1 Initialize Load

This directive begins a relocatable loading operation. The directive input format is as follows:

IL

There are no input arguments for IL.

The operator places the text to be loaded in the BIN device, types the directive IL, then turns the reader on. The loader clears the entry name table, resets the storage map and loads the text. As entry names are encountered in loading, they are placed into the entry names table. Loading ceases when an END statement is detected in the text. If Sense Switch 0 is false, the loader will load another program from BIN. If Sense Switch 0 is true, control will be returned to XRAY for further directives. (See Section 9-2.5.5 Termination of Loading.)

9-2.5.2 Continue Load

This directive is used to continue loading relocatable programs without destroying the entry names table or programs already loaded.

The directive input format is as follows:

CL

There are no input arguments for CL.

The operation and functions of the CL directive are identical to those of the IL directive except that the entry names table is not cleared and the storage map is not reset.

9-2.5.3 Entry Names Table Printout

This directive prints the contents of the entry names table generated by the loading of programs.

The input format is as follows:

ET

There are no input arguments for ET.

Each entry name which has been encountered since the last IL directive will be listed along with its location in core. Undefined names will be used to fill all references to that name. All subsequent definitions of that name will be printed, with their location, but flagged by a D.

9-2.5.4 Execute

This directive executes a program loaded into core using RELOADB.

The directive input format is as follows:

T^

where ^ is a space

There are no input arguments for T.

The loader scans the entry names table (ENT) and prints all undefined or multiply defined entries. If there are any undefined values the loader will print "MS" (meaning missing) and halt.

At this time execution may be forced by setting Sense Switch 1 true (jp) and depressing the RUN button. Setting Sense Switch 1 false and depressing the RUN button causes the loader to return to XRAY.

If there are no undefined symbols (or if execution has been forced as above) the loader will check to see if an execution address has been specified. If not, it will type "NX" (meaning no execution address) and reurn to XRAY.

If an execution address has been specified execution of the program will begin. The program may be restarted at any time by using the XRAY "T" directive.

9-2.5.5 Termination of Loading

The loading operation proceeds either automatically or under manual control. Each time an END statement is read from the text, i.e., at the end of each program module, Sense Switch 0 is examined. If it is true, control will be returned to XRAY, from which loading of the next module can be

initiated by using directive CL. If the switch is false, loading will proceed to the next module, and will continue from module to module until the switch is set true, or until a record consisting of the single character BELL (ASCII 87), the end-of-file character, is read from the paper tape. This character may be manually punched at the end of a loadable tape. It must not be preceded by a line feed character. When the end-of-file is read, the loader will return unconditionally to XRAY, and other directives may be typed.

9-2.5.6 Relocatable Loader Error Messages

When errors are detected by the loader, one of the following error codes will be output:

Code	Error	Action to be Taken
CK	Checksum error or non-loader text record.	Re-position the tape in front of the record, if desired to re-attempt reading it, and push the RUN button to continue.
LC	An unrecognized loader code has been encountered.	No recovery
MX	Program being loaded is larger than available memory, or insufficient memory for the entry names table (approximately 100 symbols can be held in this table).	No recovery
MS	Undefined names in the entry names table.	Load missing program using the CL directive.
NX	No execution address has been specified (no main line program has been loaded).	Load main line using the CL directive.

When any of these errors occur the loader will halt to allow the operator to turn off the reader before any messages are output. When the operator pushes the RUN button, the loader will continue.

9-2.6 SYM-I/PREP Operation

SYM-I/PREP consists of only one paper tape that must be loaded before a source paper tape may be prepared or a program can be assembled.

Figures 9-3 through 9-6 describe the four phases of preparing and assembling a program using SYM-I/PREP. The four phases of SYM-I/PREP are:

- Initialization
- PREP Mode Operation
- Assembly from Paper Tape/Output Binary Paper Tape
- Assemble from Keyboard into Memory.

9-2.6.1 SYM-I/PREP Initialization

SYM-I/PREP is initialized by loading the SYM-I/PREP absolute binary paper tape into memory using either the AL directive of XRAY or the hardware bootstrap and the Initial Loader.

Device assignment can be made from the keyboard with either the teletype paper tape reader (TTYPTR), the teletype keyboard (Keyboard), or the High-Speed Paper Tape Reader (HSPTTR) designated as the *source input unit*. The *source output unit* is designated as either the Teletype Paper Tape Punch (TTYPCH) or the High-Speed Paper Tape Punch (HSPTP). All assignments are made by typing two-character directives preceded by a line-feed and followed by a carriage return.

After assigning the devices, either the PREP Mode of Operation or the SYM-I assembly mode can be selected.

9-2.6.2 PREP Mode Operation

The PREP mode of operation allows the operator to prepare source paper tapes using the teletype keyboard. In this mode, diagnostic messages are output at the time of initial program statement

entry. An *input listing* (Figure 9-7) is produced simultaneously with program statement entry.

Upon encountering an END statement or when the memory buffer area has been exceeded the operator may elect to output a *source paper tape* and a *PREP Listing* (Figure 9-8) including line numbers. Error on the source paper tape may be edited (statements inserted, replaced or deleted) by loading and using the Symbolic Program EDITOR-BASIC (see Section 9-2.7).

9-2.6.3 Assemble From Paper Tape

Once a relatively syntax error-free source tape exists, the program may be assembled using the SYM-I option with the TTYPTR or HSPTTR designated as the source input unit.

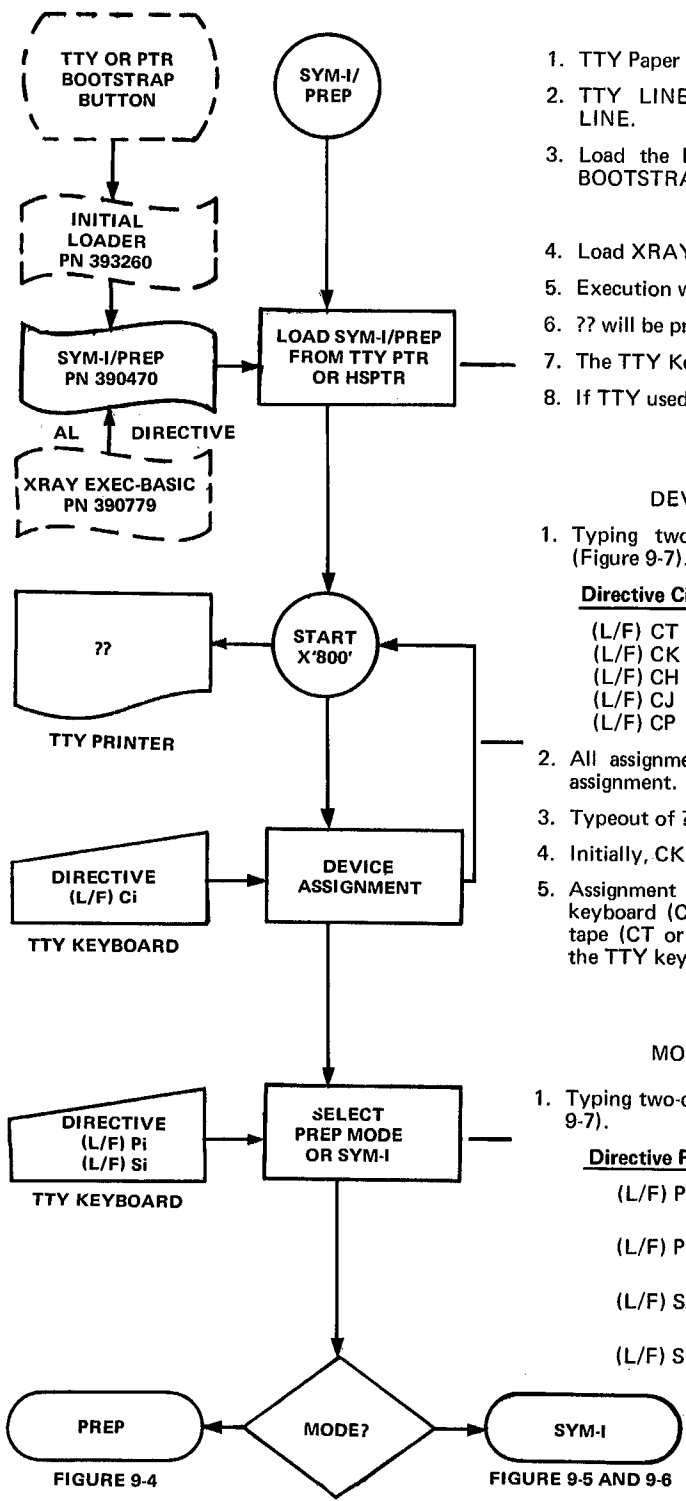
The output of this operation is an *object listing* (Figure 9-9) and an *object paper tape*. The object listing shows the hexadecimal locations (absolute or relative) and contents for the source program just assembled. The object paper tape (binary) is ready to be loaded by the absolute or relocatable loader of XRAY.

9-2.6.4 Assemble From Keyboard into Memory

To bypass using a source program paper tape and assembling directly into memory, the SYM-I option must be selected with the TTY keyboard as the source input unit. An optional object listing and object paper tape may also be output in this mode.

After entering the END statement, the assembled program resides completely in memory and may be executed by manually transferring the program counter to the starting location and processing RUN.

Program assembled directly into memory reside starting at location X'16'.



SYM-I/PREP IS LOADED BY:

1. TTY Paper Tape Punch should be turned off
 2. TTY LINE-OFF-LOCAL switch should be positioned to LINE.
 3. Load the Initial Loader using the TTY or PTR Hardware BOOTSTRAP button (Section 9-2.1).
- OR
4. Load XRAY and use the AL directive (Section 9-2.4.)
 5. Execution will automatically be transferred to X'800'.
 6. ?? will be printed on the TTY keyboard.
 7. The TTY Keyboard light is turned on.
 8. If TTY used for loading, turn off reader.

DEVICE OPTIONS ARE ASSIGNED BY:

1. Typing two-character directives preceded by a line-feed (Figure 9-7).

<u>Directive Ci</u>	<u>Assignment</u>
(L/F) CT	Assign TTY paper tape reader as source input unit.
(L/F) CK	Assign TTY keyboard as source input unit.
(L/F) CH	Assign HSPTR as source input unit.
(L/F) CJ	Assign HSPTR as output unit
(L/F) CP	Assign TTY paper tape punch as output unit.

2. All assignments remain in effect until changed by another assignment.
3. Timeout of ?? follows each assignment.
4. Initially, CK and CP have been assigned.
5. Assignment for PREP input unit is normally the TTY keyboard (CK), assignments for SYM-I input unit are paper tape (CT or CH) when the source is generated by PREP and the TTY keyboard (CK) for assembling directly into core.

MODE SELECTION IS SPECIFIED BY:

1. Typing two-character directive preceded by a line-feed (Figure 9-7).

<u>Directive Pi or Si</u>	<u>Mode</u>
(L/F) PA	PREP MODE specified for an absolute program
(L/F) PR	PREP MODE specified for a relocatable program
(L/F) SA	SYM-I ASSEMBLY MODE specified for an absolute program.
(L/F) SR	SYM-I ASSEMBLY MODE specified for a relocatable program.

FIGURE 9-4

FIGURE 9-5 AND 9-6

Figure 9-3. SYM-I/PREP Initialization

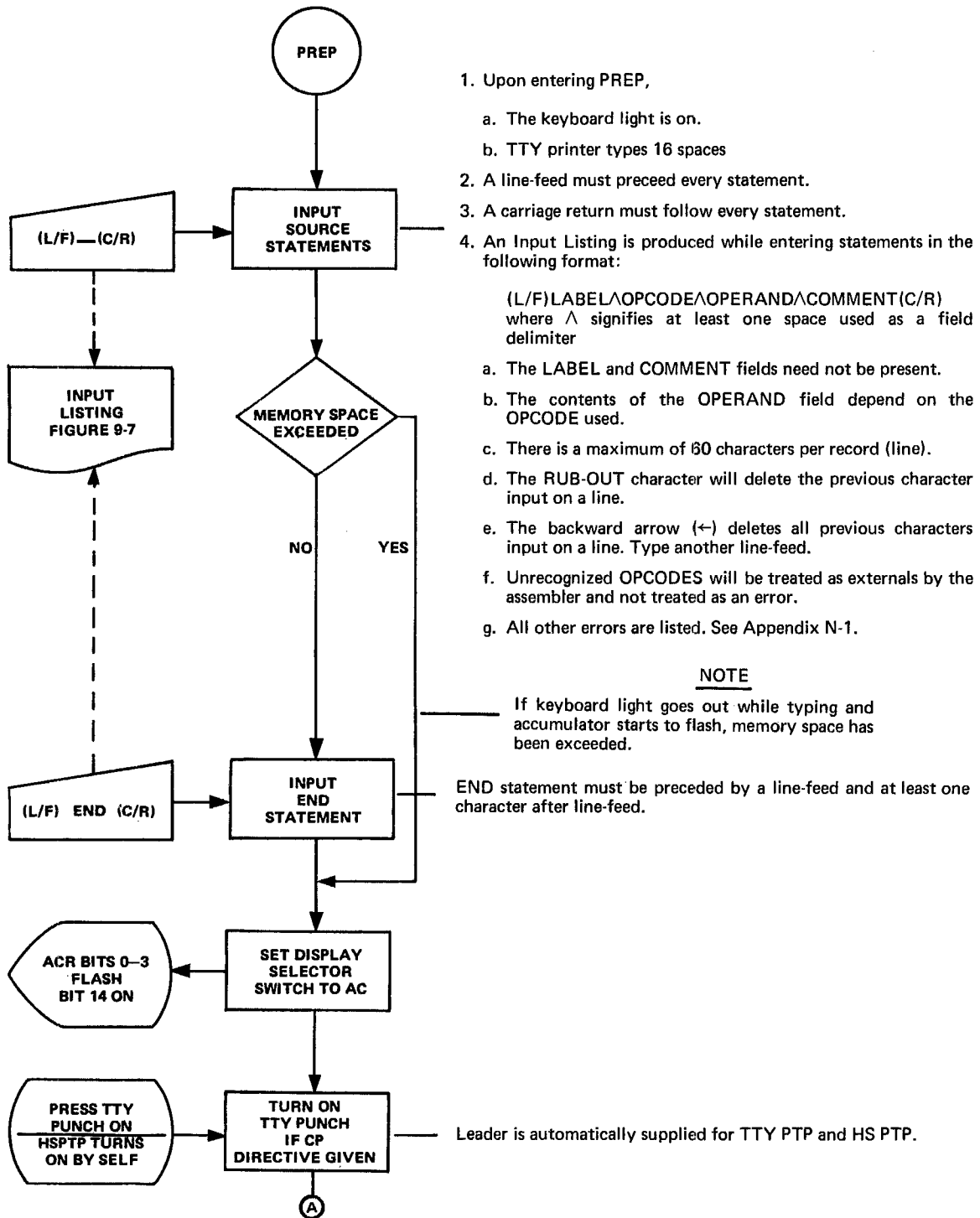


Figure 9-4. SYM-I PREP Mode Operation (Sheet 1 of 2)

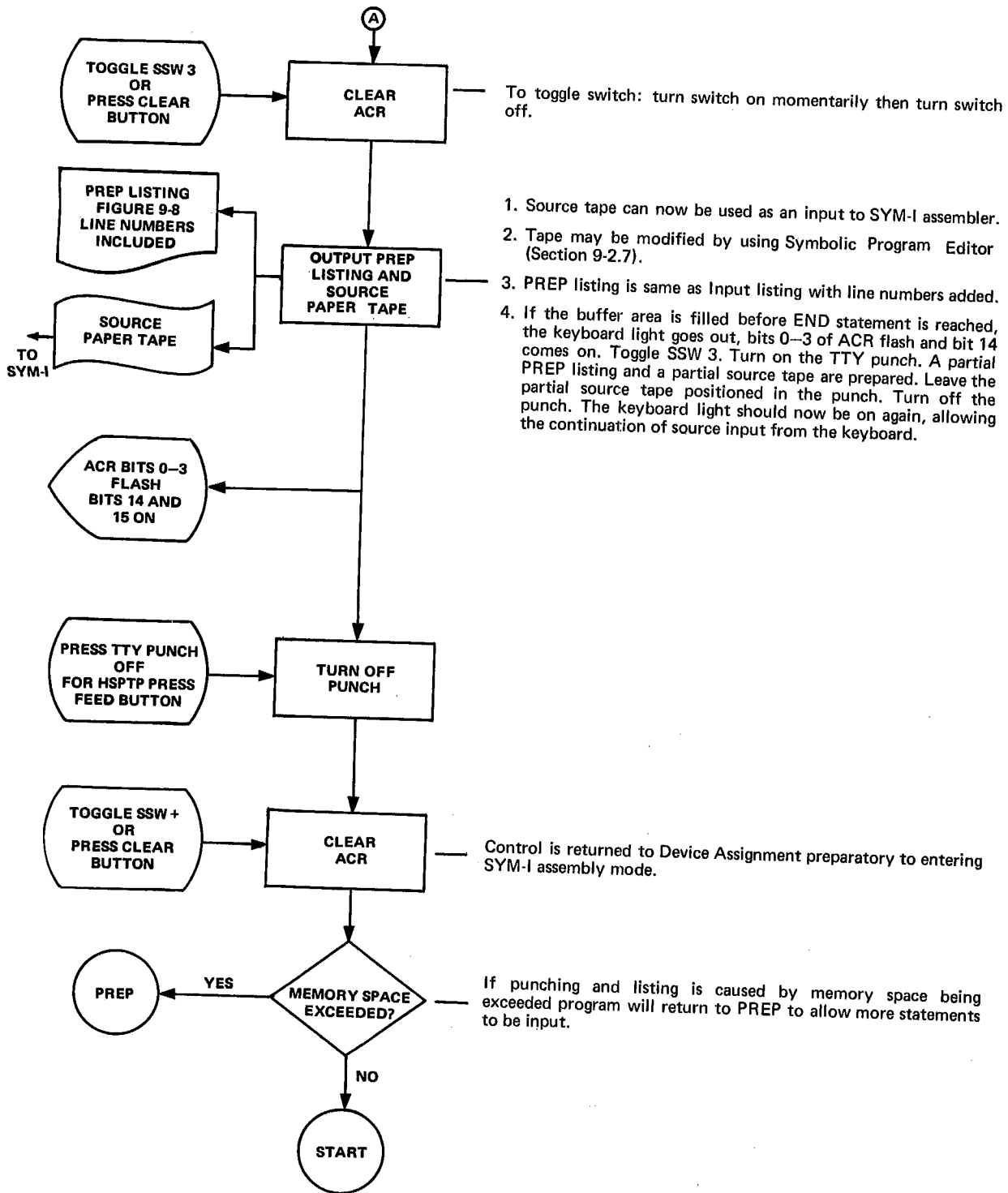
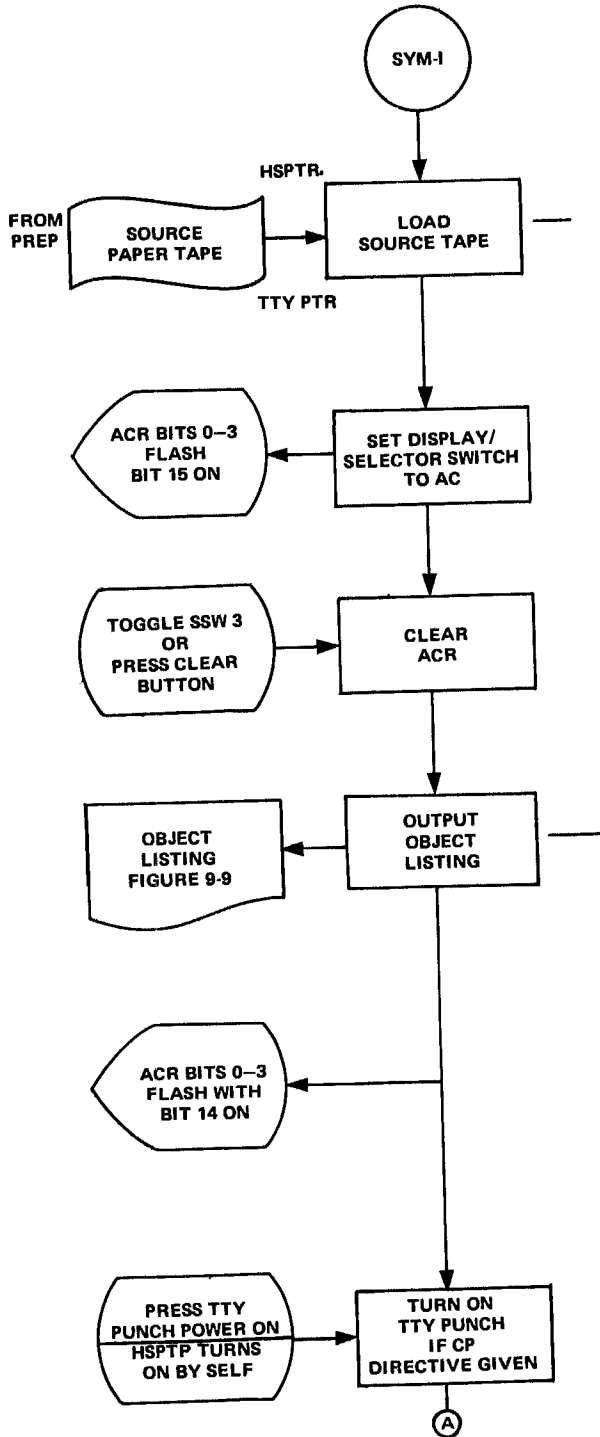


Figure 9-4. SYM-I/PREP Mode Operation (Sheet 2 of 2)

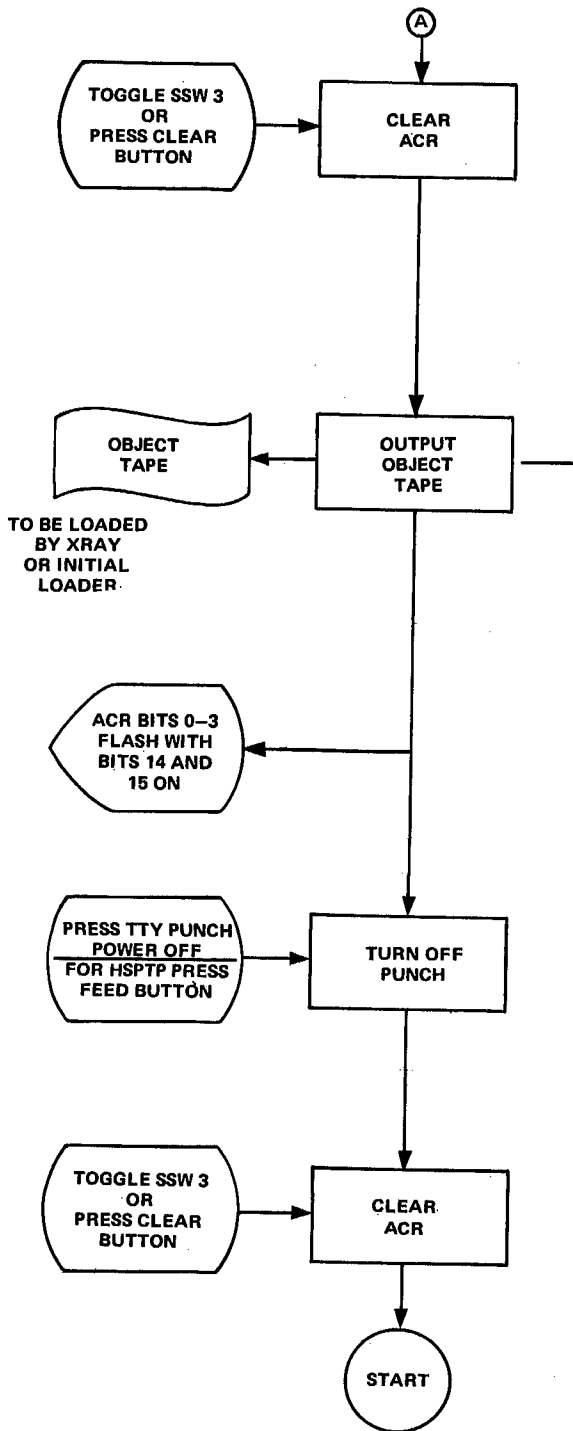
ASSEMBLE FROM PAPER TAPE



1. When using HSPTR (CH), make sure that RUN-LOAD switch is placed in RUN position.
2. When using TTY paper tape reader (CT), make sure STOP-START switch is placed in START position after PTR light comes on.
3. External names are given at the end of reading.
4. If TTY used, turn off reader at end of load.
5. Memory does not contain executable program.
6. If memory is exceeded, error message 'v' is indicated. Program must be segmented.

The OBJECT listing gives hexadecimal locations and contents plus all labels that are in source program.

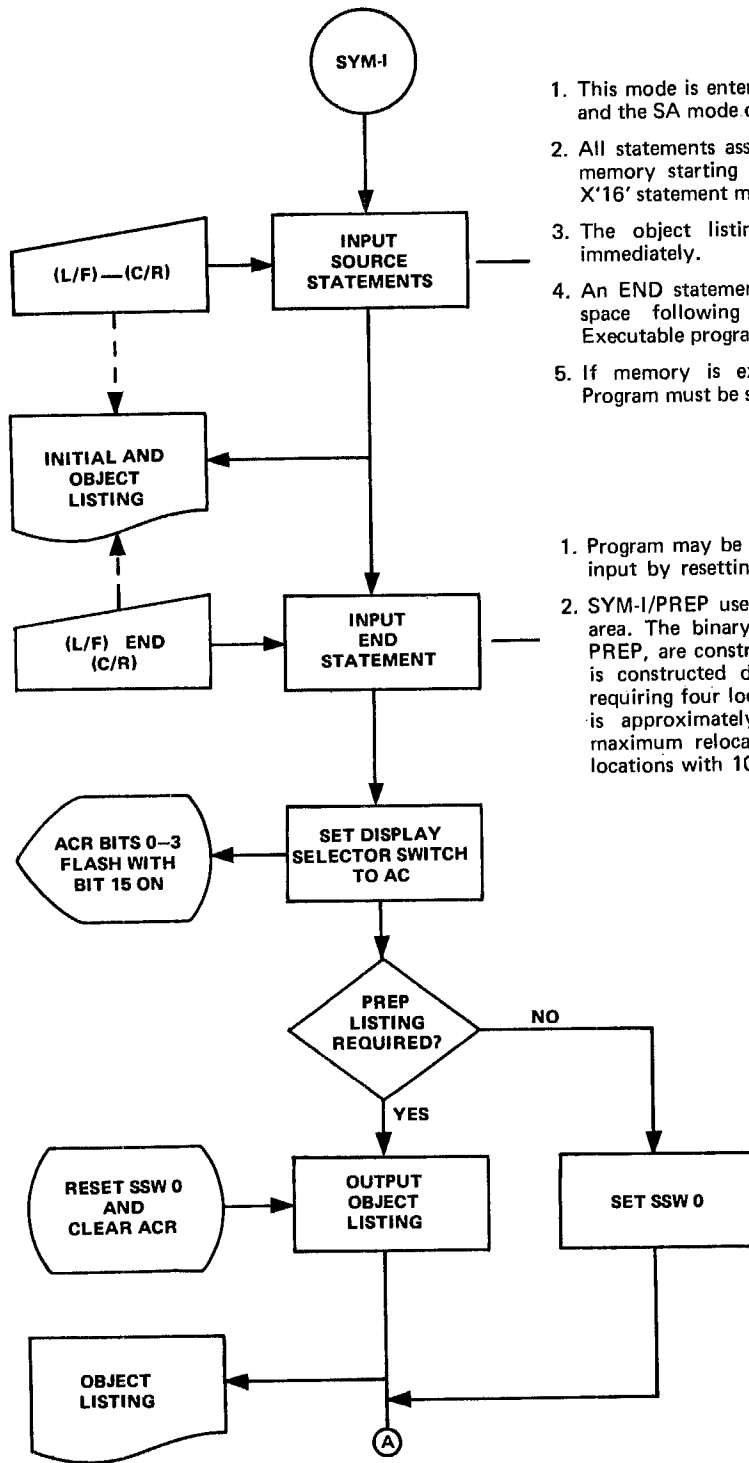
Figure 9-5. SYM-I Assemble From Paper Tape (Sheet 1 of 2)



1. Object tape is either in absolute or relocatable format according to whether SA or SR was selected.
2. The object tape may be loaded by reloading XRAY and/or the relocatable loader and using the AL or IL directives respectively.
3. The object tape may also be loaded by using the initial loader.
4. The object tape may also be loaded by using the absolute bootstrap (see Section 8-5.1.5). By typing the control directives (L/F) XT or (L/F) XH, the teletype or high-speed paper tape absolute bootstrap will be copied into cells 0 through A16. These directives must be typed after ?? appears on the printer.

Figure 9-5. SYM-I Assemble From Paper Tape (Sheet 2 of 2)

ASSEMBLE FROM KEYBOARD



1. This mode is entered by entering the CK assignment directive and the SA mode directive.
2. All statements assembled from the keyboard are placed into memory starting at location X'16'. An ORIG 22 or ORIG X'16' statement must begin every program.
3. The object listing (hex) is added to the initial listing immediately.
4. An END statement preceded by a line-feed and at least one space following the line-feed terminates the assembly. Executable program is now in memory.
5. If memory is exceeded, error message "V" is indicated. Program must be segmented.

1. Program may be executed at any time after END statement is input by resetting computer and starting execution at X'16'.
2. SYM-I/PREP uses the locations X'16' to X'5AF' for a work area. The binary text for SYM-I, or source statements for PREP, are constructed upward from X'16'. The symbol table is constructed downward from X'5AF' with each symbol requiring four locations. The maximum absolute program size is approximately 1000₁₀ words with 100 symbols. The maximum relocatable program size is approximately 500₁₀ locations with 100 symbols.

Figure 9-6. SYM-I Assemble From Keyboard Into Memory

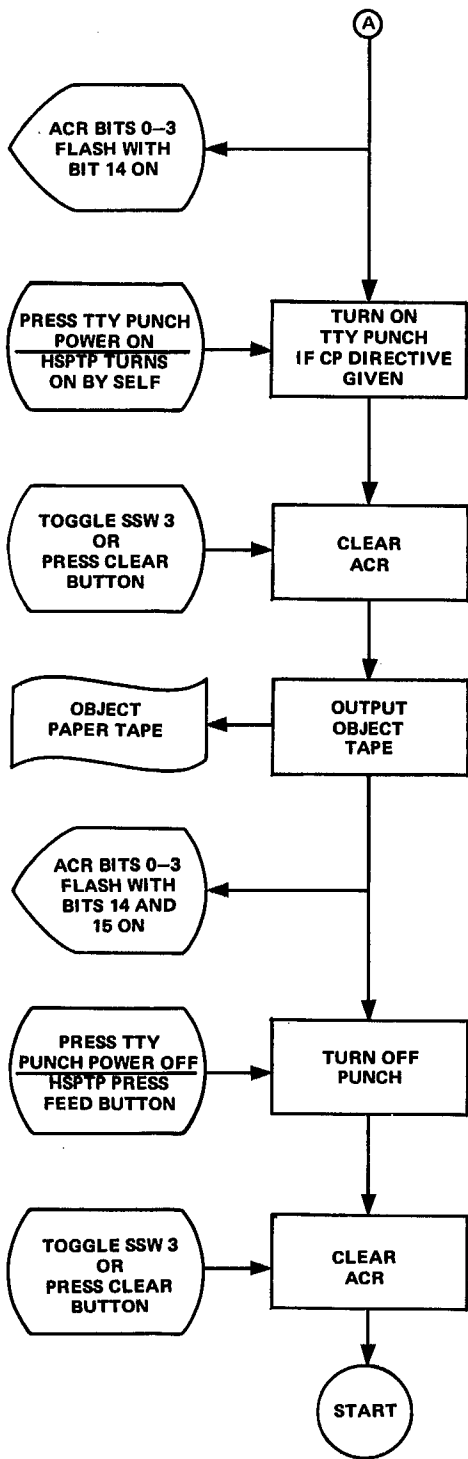


Figure 9-6. SYM-I Assemble From Keyboard Into Memory (Sheet 2 of 2)

??
CK

??
PA

```
*      SAMPLE SYM-1 PROGRAM
*
      ORIG X'600'      ABSO ORIGIN
DOIO   ←
DOIO   EQU    68
TEN    DATA 10,X'10',TEN,/TEN,'10', $ SAMPLE DATA
P      BUF    RES X'62000'-$ VARIABLE RESERVE
      BUF    RES X'620'-$ VARIABLE RESERVE
WC     D      $-BUF   SHORT FORM OF DATA
START  LDW    MESSAGE
      STW * 2
      SMB    DOIO    SET EXR FOR JUMP
      JSX    DOIO,FIOT,BUF,WC FIOT IS FORWARD DEFINE
      MPY    TEN     HARDWARE MULTIPLY
      STB    TEN+3   BYTE ADDRESS 2*TEN+3
      TRUE   TEN>500 CONDITION
      CLB    'S'     COMPARE A(8-15) WITH 'S'
      SEQ    'S'     SKIP IF A(8-15) EQUALS 'S'
      SLC R 3      CIRCULAR SHIFT RIGHT BYTE
      ENDC      TERMINATE RANGE OF CONDITION
      LONGLABELFORMYCONVENIENCE BYTE 3,DOIO FILL BYTES
LISA   LDW    TEN
      FAD    TEN     EXTERNAL CALL
D      WC     STW    TEN
      FIOT   RES    8      FILLS REFERENCE IN DOIO CALL
      DATA 'ER','RO','R' CHARACTER STRING
      HLT   3      HALT WITH 3 ON MEMORY BUS
      END
```

Figure 9-7. SYM-1/PREP Input Listing

```

1 *   SAMPLE SYM-1 PROGRAM
2 *
3     ORIG X'600'   ABSO ORIGEN
4 DOIO EQU 68
5 TEN  DATA 10,X'10',TEN,/TEN,'10',S SAMPLE DATA
6 BUF  RES X'620'-S VARIABLE RESERVE
7 WC   D $-BUF   SHORT FORM OF DATA
8 START LDW MESSAGE
9      STW * 2
10     SMB DOIO   SET EXR FOR JUMP
11     JSX DOIO,FIOT,BUF,W C FIOT IS FORWARD DEFINED
12     MPY TEN    HARDWARE MULTIPLY
13     STB TEN+3  BYTE ADDRESS 2*TEN+3
14     TRUE TEN>500 CONDITION
15     CLB 'S'    COMPARE A(8-15) WITH 'S'
16     SEQ       SKIP IF A(8-15) EQUALS 'S'
17     SLC R 3    CIRCULAR SHIFT RIGHT BYTE
18     ENDC      TERMINATE RANGE OF CONDITION
19 LONGLABELFORMYCONVENIENCE BYTE 3,DOIO FILL BYTES
20 LISA LDW TEN
21     FAD TEN    EXTERNAL CALL
22 FIOT RES 8     FILLS REFERENCE IN DOIO CALL
23     DATA 'ER','RO','R' CHARACTER STRING
24     HLT 3      HALT WITH 3 ON MEMORY BUS
      END

```

Figure 9-8. SYM-1/PREP Listing

??			
CT			
SA	??		
MESS	0621	}	EXTERNALS, ILLEGAL LABELS, OR ILLEGAL OP CODES
FAD	0631		
0600	000A	TEN	
0601	0010		
0602	0600		
0603	0C00		
0604	B1B0		
0605	0605		
0606	0000	BUF	
0607	0000		
0608	0000		
0609	0000		
060A	0000		
060B	0000		
060C	0000		
060D	0000		
060E	0000		
060F	0000		
0610	0000		
0611	0000		
0612	0000		
0613	0000		
0614	0000		
0615	0000		
0616	0000		
0617	0000		
0618	0000		
0619	0000		
061A	0000		
061B	0000		
061C	0000		
061D	0000		
061E	0000		
061F	0000		
0620	001A	WC	
0621	87FF	STAR	EXTERNAL OPERAND
0622	7802		
0623	0080		
0624	2044		
0625	0633		
0626	0606		
0627	8620		
0628	0B0F		
0629	0600		
062A	3403		
062B	0703		
062C	0860		
062D	0AF3		
062E	0344	LONG	
062F	8600	LISA	
0630	07FF	}	EXTERNAL OPERATION CALL : SMB (LEGAL FOR RELOCATABLE DATA (SIGN BIT SET ON))
0631	2630		
0632	8600		
0633	0000	FIOT	
0634	0000		
0635	0000		
0636	0000		
0637	0000		
0638	0000		
0639	0000		
063A	0000		
063B	C5D2		
063C	D2CF		
063D	D2A0		
063E	0003		

Figure 9-9. SYM-I Object Listing

9-2.7 Symbolic Program Editor-BASIC

The Symbolic Program Editor BASIC (SYMEDB) generates a modified version of a Raytheon 706 symbolic language program without repunching the entire program.

Changes are related to the editor in the form of specially formatted instructions called directives. Only the changes need to be specified — unchanged instructions will be punched exactly as they appear in the old version of the program.

Three types of changes may be made to a program.

1. Statements may be inserted between existing lines.
2. Existing lines may be deleted.
3. Statements may be input to replace existing lines.

A program is modified in two phases. During the first phase, which is called "Directive time" all directives and new statements are input. During the second (edit) phase the old source text is read in, it is changed as specified by the directives which have been input, and a new source program is then punched.

SYMEDB edits SYM I, SYM II or 706 FORTRAN source paper tapes. Off-line tapes must be prepared per the alphabetic format set forth in Section 8-6. Tapes prepared by SYM I/PREP are in the correct format for editing.

9-2.7.1 Operation of The Editor

The Symbolic Program Editor operates under the control of the XRAY EXEC monitor. The editor is distributed in the form of an absolute object tape and is loaded using the XRAY directive AL.

Once loaded, control is transferred to the editor by the XRAY directive "ET". The editor will respond by printing 'BE' (begin editor) on the LIST device and will wait for directives and new statements to be input on the SYSI device. These two logical units (LIST AND SYSI) are normally assigned to the teletype, however, they may be reassigned through XRAY directives (see Section 9-2.4).

If core is exhausted at directive time, the editor will print 'OV' (overflow) on the teletype and control will be returned to XRAY. If an incorrect editor directive is input or if an incorrect sequence of directives and statements is used, the editor will respond by printing 'Q?' on the teletype. All input back to and including the last directive will be deleted and the editor will wait for another directive.

After all directives for insertions, deletions, and replacements have been input, the directive "+E" is issued. This directive signifies the end of directive time and the start of the edit phase. The editor will then go to the PRIN device to begin inputting the symbolic program being edited.

Normally, reading of the source language text will continue until the END statement is read; however, if core is exhausted before the END statement is reached, the editor will stop inputting and begin editing the portion of the program which has been read.

If the input device (PRIN) is the teletype reader, it must be stopped manually. Stopping the reader quickly is essential because after editing is complete the tape must be restarted without missing any statements (see Section 9-2.7.3).

When reading stops (whether because core was exhausted or because an END statement was reached) editing will begin. When editing is complete the new source text is punched on the BOUT unit. If the BOUT device is the teletype, the computer pauses with bit 14 on in the ACR. This is the signal to turn the punch on. Clearing the ACR or toggling Sense Switch 3 causes the source text to be punched. When punching is complete the computer pauses again. This time bits 14 and 15 are on in the ACR. This is the signal to turn off the punch. Clearing the ACR or toggling Sense Switch 3 allows the computer to continue.

If editing was initiated by exhausting core, the editor will go back to read more source data. If the input device (PRIN) is the teletype, as soon as the reader is turned on, the editor will again input source text.

If editing was initiated by reading an END statement in the source text, the editor will return ('BE' will be printed on the teletype) to look for more directives. At this time the editor directive "+X" may be input to return to the XRAY EXEC monitor, or additional insertions, deletions, and replacements may be specified for the edit of another source program.

The Symbolic Program Editor can use for its input devices (PRIN) all paper tape devices, magnetic tape and a card reader. For output devices it can use all paper tape devices, magnetic tape, and a card punch. If the output device is not the teletype, (i.e., if the BOUT and LIST device are not the same) listing will occur on the list device and may be suppressed by setting Sense Switch 2 true.

9-2.7.2 Description of Editor Directives

The following general statements apply to all Symbolic Program Editor directives:

1. They must be preceded by a line feed (LF,X'8A') and followed by a carriage return (C/R,X'8D')
2. They must begin with a plus sign (+,X'AB').
3. All characters in a directive line are ignored after the first blank encountered. (Thus, the directive line may contain a comments field if it is separated from the directive by a blank).

These general statements apply to the symbolic source statements which are typed in with the directives:

1. They must be preceded by a line feed and followed by a carriage return.
2. They must not begin with a plus sign. (The editor uses + to identify an editor directive.)
3. They must be no more than 54 characters long. (All additional characters will be ignored.)

Insertion

This directive is used to insert statements into the program. The format is:

```
+M
S1
S2
.
.
Si
```

where S₁,S₂ . . . S_i are the statements to be inserted following line M.

To insert a data instruction between lines 23 and 24:

```
+23
WENDY DATA 0
```

Deletion

This directive deletes the specified lines from the old source.

The format is:

```
+M,N
```

where lines M to N, inclusively, are to be deleted. To delete lines 20 and 21:

```
+20,21
```

Replacement

This directive replaces the specified lines in the old source text with the statements typed in.

The format is:

```
+M,N
S1
S2
.
.
Si
```


where: M,N are the numbers of the first and last lines to be replaced
 $S_1, S_1 \dots S_i$ are the statements which will replace the lines specified.

To replace lines 1 and 2 by three comment statements:

```
+1,2
SAMPLE SYM-I PROGRAM EDITED
```

Note that deletion is actually a special case of replacement. Deleting is actually replacing the specified lines with no instructions.

Edit

This directive signals the Symbolic Program Editor that all insertions, deletions and replacements have been completed and editing should being.

The format is:

```
+E
```

After this directive is input the editor will begin reading the source text which is to be modified.

Return to XRAY

This directive is used to signify that all symbolic editing is complete.

The format is:

```
+X
```

The editor will return control to the XRAY EXEC monitor

9-2.7.3 Paper Tape Format

The paper tape containing source text to be edited by the Symbolic Editor should be in the standard SYM I/PREP format. The output tape generated by the editor will also be in this format.

The format is as follows:

State- ment i-1	C R	Null	Line No. (i-1)	b l a n k	L F	State- ment i	C R	Null	Line No. (i)	b l a n k	L F	State- ment i+1
-----------------------	--------	------	----------------------	-----------------------	--------	---------------------	--------	------	--------------------	-----------------------	--------	-----------------------

Since the editor uses the I/O MONITOR for input and output, each line must start with a line feed. Care must be taken when the editor stops reading paper tape on the teletype as a result of core being full. If the editor stops inputting with line i-1, it will already have read the carriage return that follows line i-1. If the tape is being read from the teletype, the reader will continue to run, and the teletype keyboard will begin to chatter. This is a signal that the information is not being read into core. At this point the reader must be turned off manually.

If possible the reader should be turned off within the next seven character (3 nulls, 3 line Nos., and 1 blank). If this is done, when the reader is turned on again to read statement i, it will be able to read the line feed preceding statement i.

If the reader is allowed to go more than 7 characters, when it is turned on again, it will not read a line feed until the one preceding statement i+1. Thus, statement i will have been totally ignored. To read statement i in this case, the tape must be manually repositioned into the seven character *gap* preceding statement i, before the reader is turned on.

9-2.7.4 Operating Instructions

1. Load Symbolic Program EDITOR-BASIC (PN 390941) with XRAY "AL" directive.
2. Type ET; Editor will type BE.
3. Input directives and statements for addition, deletion, and replacement.
4. Type +E to signify end of directive time.

5. Load tape to be edited into TTY reader and turn on reader.
6. When teletype chatters turn reader off immediately.
7. When computer halts (with bit 14 on) turn punch on.
8. Clear ACR or toggle Sense Switch 3 to continue.
9. When computer pauses (with bits 14 and 15 on) turn teletype punch off.
10. If END statement has not been read, check to see paper tape is positioned correctly.
11. Clear ACR or toggle Sense Switch 3 to continue.
12. a. Return to instruction 5 if END statement has not been reached.
b. Return to instruction 3 if another program is to be edited.

- c. Type "+X" if return to XRAY is desired.

If the teletype is not used for source input and output, all steps in the operating instructions concerning the ACR and the turning on and off of the reader and punch may be ignored.

9-2.7.5 Example of Program Modification

Figure 9-10 is a sample program before and after modification, along with the Symbolic Editor directives to make the modifications.

Figure 9-10a is a listing of the program to be modified. The changes to be made have been typed in the right hand column (similar to the way a programmer would pencil changes onto a listing).

Figure 4-10b is a copy of the directives that were input on the teletype to make the modifications.

Figure 9-10c is a listing of the new version of the program with the changes incorporated.

```

1 *      SAMPLE SYM-1 PROGRAM }
2 *
3          ORIG X'600'      ABSO ORIGIN          ADD WORDS,
4 DOIO     EQU 68          "EDITED" AND "*"
5 TEN      DATA 10,X'10',TEN,/TEN,'10', $ SAMPLE DATA
6 BUF      RES X'620'-$ VARIABLE RESERVE
7 WC       D $-BUF      SHORT FORM OF DATA
8 START    LDW MESSAGE
9          STW * 2
10         SMB DOIO      SET EXR FOR JUMP
11         JSX DOIO,FIOT,BUF,WC FIOT IS FORWARD DEFINED
12         MPY TEN      HARDWARE MULTIPLY
13         STB TEN+3    BYTE ADDRESS 2*TEN+3
14         TRUE TEN>500 CONDITION
15         CLB 'S'     COMPARE A(8-15) WITH 'S'
16         SEQ         SKIP IF A(8-15) EQUALS 'S'
17         SLC R 3     CIRCULAR SHIFT RIGHT BYTE
18         ENDC       TERMINATE RANGE OF CONDITION
19 LONGLABELFORMYCONVENIENCE BYTE 3,DOIO FILL BYTES
20 LISA LDW TEN
21 PAD TEN      EXTERNAL CALL } DELETED LINES 20&21
22 FIOT    RES 8        FILLS REFERENCE IN DOIO CALL
23         DATA 'ER','RO','R' CHARACTER STRING ADD WENDY DATA 0
24         HLT 3        HALT WITH 3 ON MEMORY BUS
          END

```

Figure 9-10a

```

BE
+1,2
*
*      SAMPLE SYM-1 PROGRAM EDITED
*
*
+20,21
+23
WENDY DATA 0
+E

```

Figure 9-10b

Figure 9-10. Symbolic Program Editor Sample

```

1 *      SAMPLE SYM-1 PROGRAM EDITED
2 *
3 *
4          ORIG X'600'      ABSO ORIGIN
5 DOIO    EQU      68
6 TEN     DATA   10,X'10',TEN,/TEN,'10',$ SAMPLE DATA
7 BUF     RES     X'620'-$ VARIABLE RESERVE
8 WC      D       $-BUF    SHORT FORM OF DATA
9 START   LDW     MESSAGE
10        STW * 2
11        SMB     DOIO      SET EXR FOR JUMP
12        JSX     DOIO,FIOT,BUF,WC FIOT IS FORWARD DEFINED
13        MPY     TEN      HARDWARE MULTIPLY
14        STB     TEN+3    BYTE ADDRESS 2*TEN+3
15        TRUE    TEN>500  CONDITION
16        CLB     'S'      COMPARE A(8-15) WITH 'S'
17        SEQ     SKIP IF A(8-15) EQUALS 'S'
18        SLC R 3    CIRCULAR SHIFT RIGHT BYTE
19        ENDC    TERMINATE RANGE OF CONDITION
20 LONGLABELFORMYCONVENIENCE BYTE 3,DOIO FILL BYTES
21 FIOT    RES     8       FILLS REFERENCE IN DOIO CALL
22        DATA   'ER','RO','R ' CHARACTER STRING
23 WENDY   DATA   0
24        HLT     3        HALT WITH 3 ON MEMORY BUS
                END

```

Figure 9-10c

Figure 9-10. Symbolic Program Editor Sample (Cont)

9-2.8 Conversational FORTRAN Operation

Figures 9-11 through 9-15 describe the five phases of preparing, compiling, and executing a Conversational FORTRAN program. The five phases of Conversational FORTRAN are:

- Initialization
- Prepare Paper Tape
- Edit Paper Tape
- Compile and Go
- Run-Time Execution

9-2.8.1 Conversational FORTRAN Initialization

The program is Initialized by loading the Paper Tape Input/Output System (PTIOS) tape into memory using the Initial Loader and the appropriate hardware bootstrap switch (Figure 9-11). Logical units are assigned to physical units by entering directives from the teletype keyboard. The compiler is now loaded by the PTIOS. The mode (e.g. compile and go, execute, edit and prepare paper tape) is selected by entering directives from the teletype keyboard

9-2.8.2 Conversational FORTRAN Prepare Paper Tape

In the Prepare Paper Tape mode (Figure 9-12), FORTRAN statements are entered from the keyboard, creating an input listing (Figure 9-16). Syntax errors, if any, are output on the teletype. If the fill-form mode was selected statements do not have to begin in any certain column, thereby disallowing the use of continuation lines.

When memory is full or an END statement is typed, the computer will halt. Upon pressing the RUN button a PREP Listing (Figure 9-17) is output along with a FORTRAN source paper tape.

If "memory full" was the cause of the computer halting, the program will return to the Prepare Paper Tape mode, otherwise the program will be ready for new mode directives.

9-2.8.3 Conversational FORTRAN Edit

Once a source paper tape has been prepared, it is likely that since statements must be edited (Figure 9-13). Editing involves replacing, deleting and adding statements to the source paper tape. Unlike SYM-I/PREP that requires the use of the Symbolic Program Editor, Conversational FORTRAN allows the editing of a paper tape within the compiler.

9-2.8.4 Conversational FORTRAN Compile and Go

Once an error-free source paper tape has been prepared, the Compile and Go mode can be selected to translate source statements directly into executable machine code stored in core memory (Figure 9-14). FORTRAN source statements may also be entered directly from the key board. In either method of entry, a compiler listing (Figure 9-18) is generated, complete with the display of any syntax errors (up to 10 shown at the end of the listing). An optional, object paper tape may be output for use later.

9-2.8.5 Conversational FORTRAN Run-Time Execution

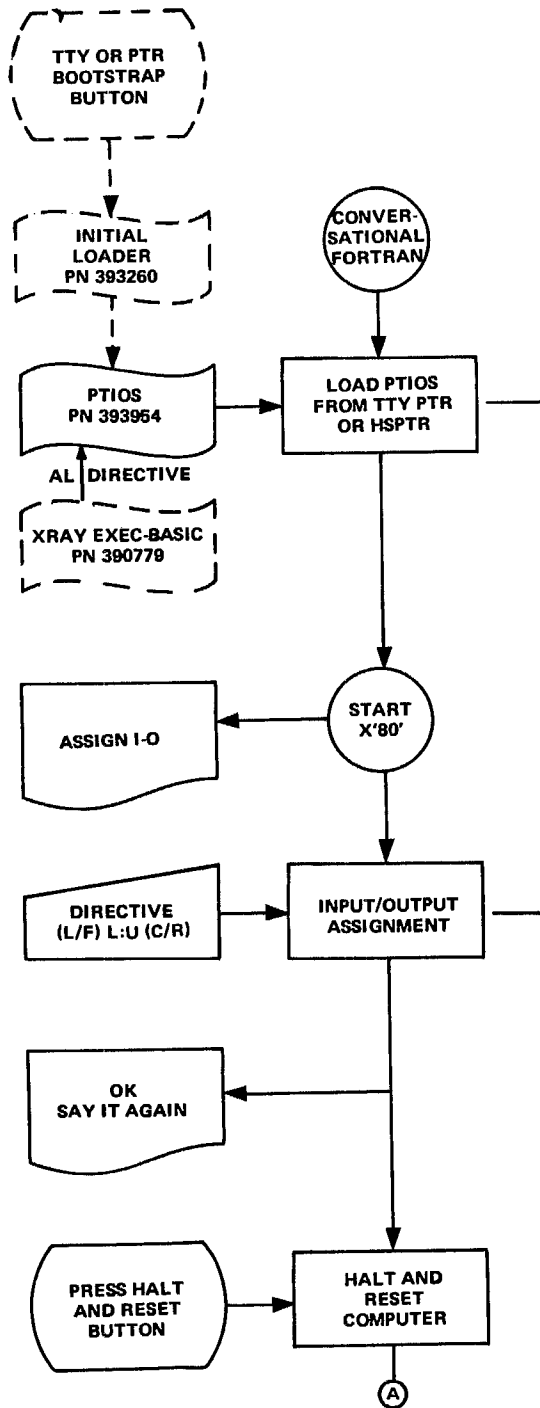
Once a FORTRAN program has been compiled, it may be executed by first loading the run-time system and library (Figure 9-15). Optional machine language programs may be loaded also.

The *trace* feature of the run-time package eases the troubleshooting of logical program errors. Each of the following statements may be traced:

- GO TO
- IF
- CALL
- RETURN
- DO
- Integer Replacement
- Real Variable Replacement

Trace statements are output on the run-time listing (Figure 9-19).

The trace feature may be interrupted and changed *during* the execution of a program.



PTIOS IS LOADED BY:

1. TTY paper tape punch should be turned off.
2. TTY LINE-OFF-LOCAL switch should be positioned to LINE.
3. Load LOADER using the TTY or PTR hardware BOOTSTRAP button (Section 9-2.1).

OR

4. Load XRAY and use the AL directive (Section 9-2.4).
5. Execution will automatically be transferred to X'80'.
6. ASSIGN I-O will be printed on the TTY keyboard.
7. The TTY keyboard light is turned on.
8. If TTY used for loading, turn off reader.

INPUT/OUTPUT IS ASSIGNED BY:

1. Logical units 0-7 are available and may be assigned to teletype, HSPTR or HSPTP.
2. Device assignment directive must be preceded by line-feed and followed by carriage return:

(L/F) L:U (C/R) where L = 0, 1, 2, 3, 4, 5, 6, or 7
and U = T for Teletype
U = R for high-speed paper tape reader
U = P for high-speed paper tape punch

All units are initially assigned to teletype.

3. System uses logical units as follows:

Unit	Use
1	Directive Input
2	Source Statement Input
4	Absolute loader binary input (used to load compiler in next step)
5	Binary output and PREP output

4. If the assignment is accepted the message OK will be output, if the assignment is not accepted the message SAY IT AGAIN will be output. Rubout or backward arrow will not delete line. To clear line type (L/F) (C/R).

Figure 9-11. Conversational FORTRAN Initialization (Sheet 1 of 2)

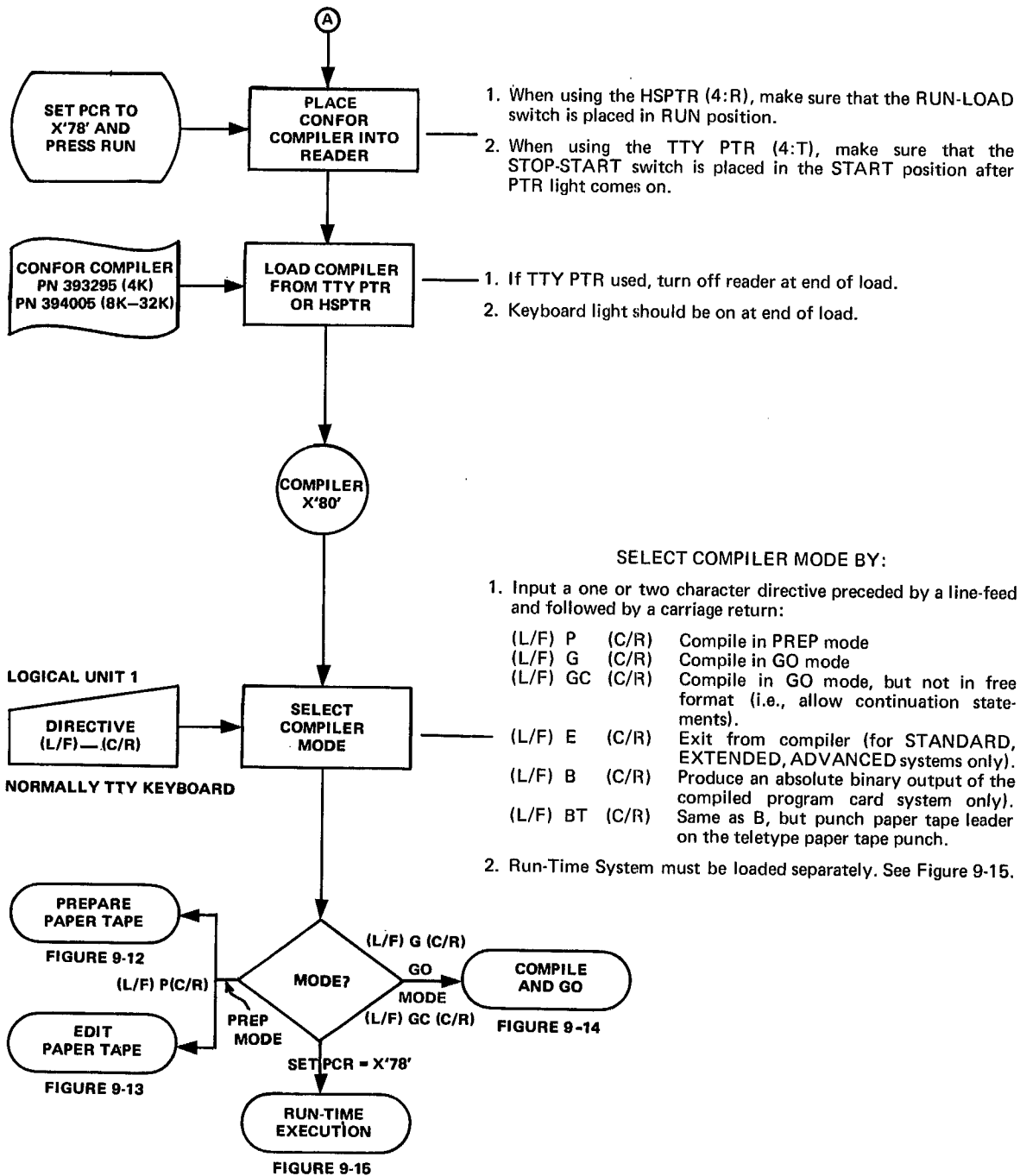
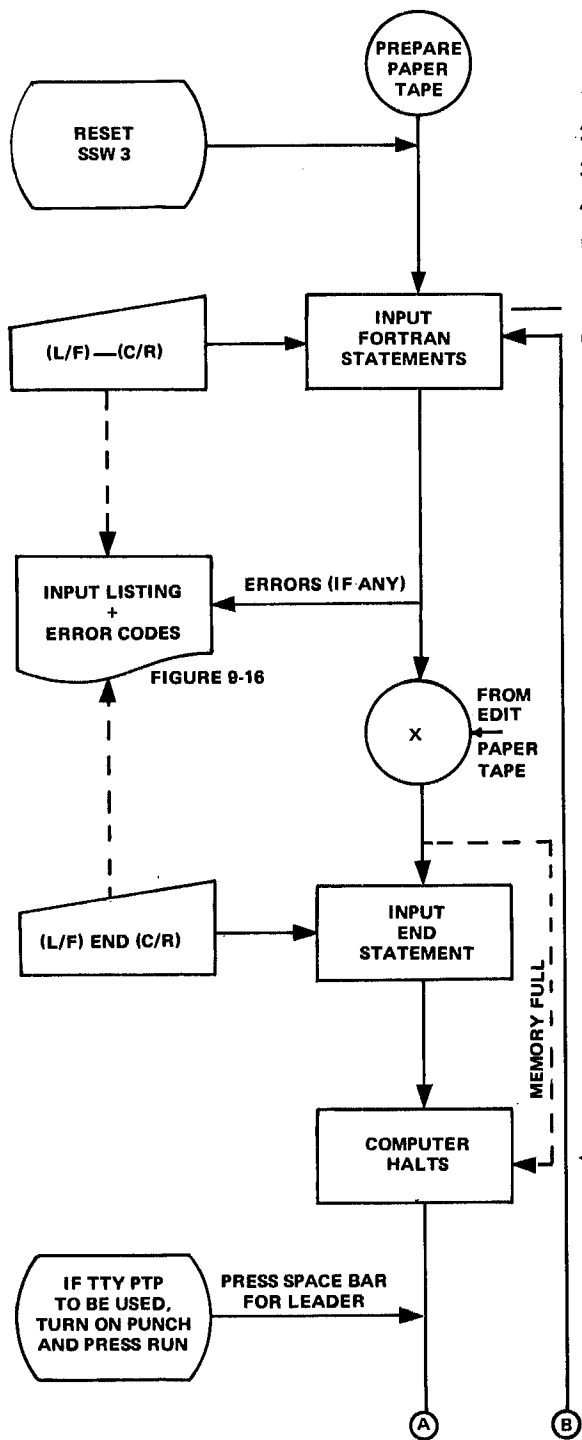


Figure 9-11. Conversational FORTRAN Initialization (Sheet 2 of 2)



1. Upon entering prepare paper tape, the keyboard light is on.
2. Sense Switch 3 should be turned off (down).
3. A line-feed must precede every statement.
4. A carriage return must follow every statement.
5. An input listing is produced while entering statements in the following format:
(L/F) FORTRAN statement (C/R)
6. All spaces are ignored and no continuation statements are allowed (Free Format).
7. Maximum of 72 characters on teletype. PREP will accept more than 72 characters but listing will be unintelligible.
8. Erroneous statements are indicated by a two character code followed by a string of periods and terminated by an up-arrow pointing to the offending character. Error codes are listed in Appendix N-5.

The computer will halt when:

- a. The statement (L/F) END (C/R) is input or
- b. Available memory has been filled.

Figure 9-12. Conversational FORTRAN Prepare Paper Tape (Sheet 1 of 2)

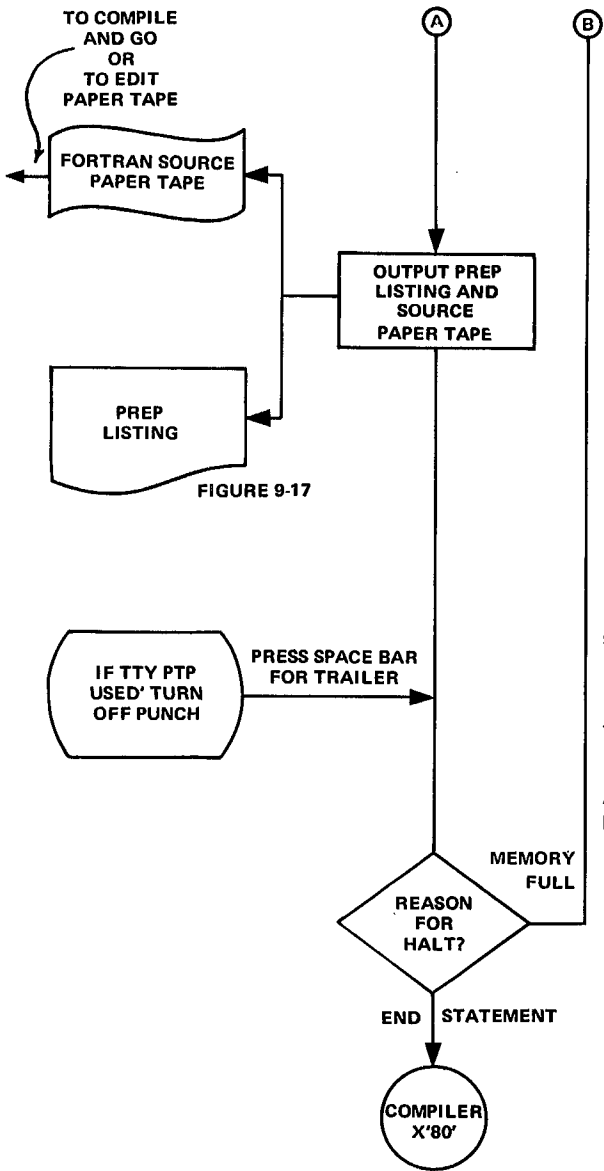


FIGURE 9-17

If memory full caused halt and subsequent punching, do not press space bar for trailer.

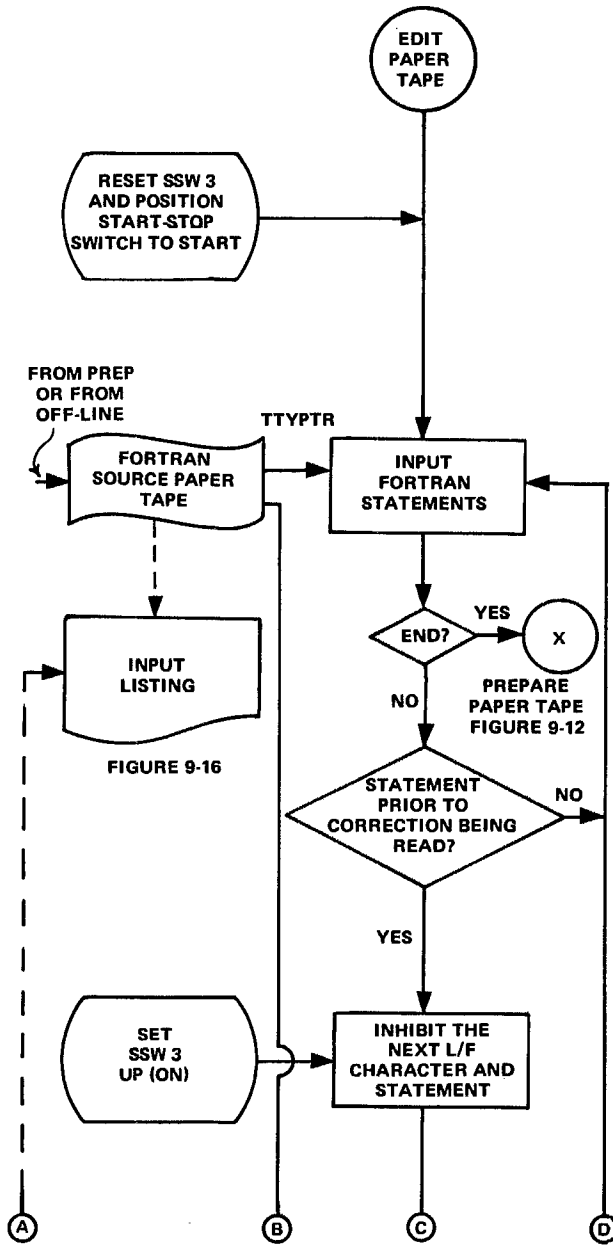
If memory space has caused HALT and subsequent punching, the teletype is connected for remaining statements.

Additional programs that are to be executed together may be prepped by typing (L/F) P (C/R) before each program.

Paper tape may now be edited or compiled.

NOTE: Prepare Paper Tape and Edit Paper Tape are the same routine with the Sense Switch 3 option used in editing paper tape.

Figure 9-12. Conversational FORTRAN Prepare Paper Tape (Sheet 2 of 2)



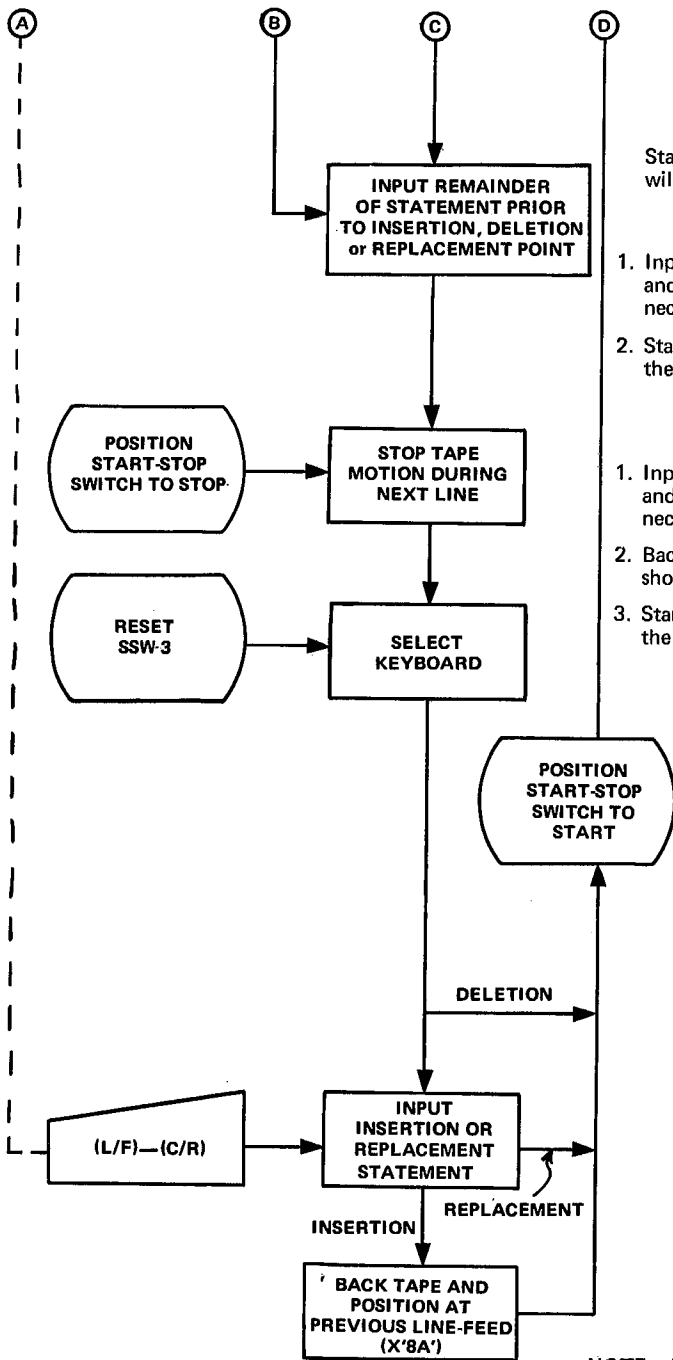
1. Upon entering Edit Paper Tape the keyboard light is on if TTY is selected as source input device (2:T).
2. Reset Sense Switch 3 and load FORTRAN source paper tape into reader.
3. Position TTY START-STOP switch to START.
4. Even though the keyboard light is on, input statements will be accepted from the TTY PTR.

Remainder of this routine is same as FORTRAN Prepare Paper Tape.

PROCEDURE FOR DELETING, REPLACING, OR INSERTING STATEMENTS

1. When the statement prior to the correction point is reached, Sense Switch 3 must be turned on. Simultaneously, STOP paper tape motion on the teletype.
2. Start the teletype paper tape reader and finish inputting the statement prior to the correction point. Stop the tape motion during the line to be corrected.
3. Turn Sense Switch 3 off (down). The keyboard light should now be on.

Figure 9-13. Conversational FORTRAN Edit Paper Tape (Sheet 1 of 2)



TO DELETE THE CURRENT STATEMENT:

Start the teletype paper tape reader. The current statement will be ignored.

TO REPLACE THE CURRENT STATEMENT:

1. Input source statements from keyboard preceded by line-feeds and followed by carriage returns. As many statements as necessary may be inserted.
2. Start the teletype paper tape reader. The current statement in the reader will be ignored.

TO INSERT STATEMENTS:

1. Input source statements from keyboard preceded by line-feeds and followed by carriage returns. As many statements as necessary may be inserted.
2. Back paper tape to preceding line-feed character (X'8A'). Tape should be positioned directly over reader pin mechanism.
3. Start the teletype paper tape reader. The current statement in the reader will be accepted.

NOTE: Prepare Paper Tape and Edit Paper Tape are the same routine with the Sense Switch 3 option used in editing paper tape. The High-Speed Paper Tape Reader should not be used for editing paper tape.

Figure 9-13. Conversational FORTRAN Edit Paper Tape (Sheet 2 of 2)

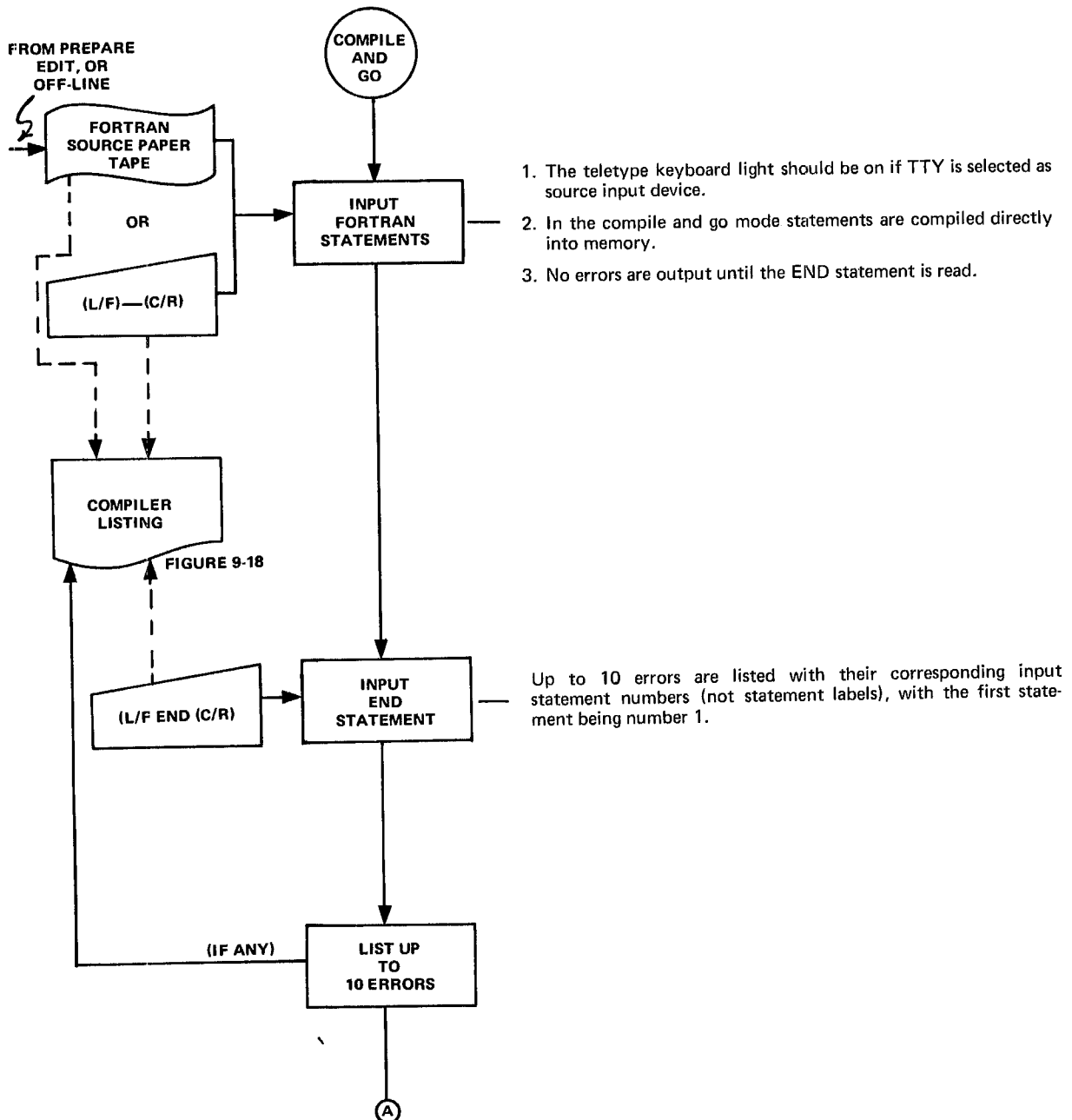
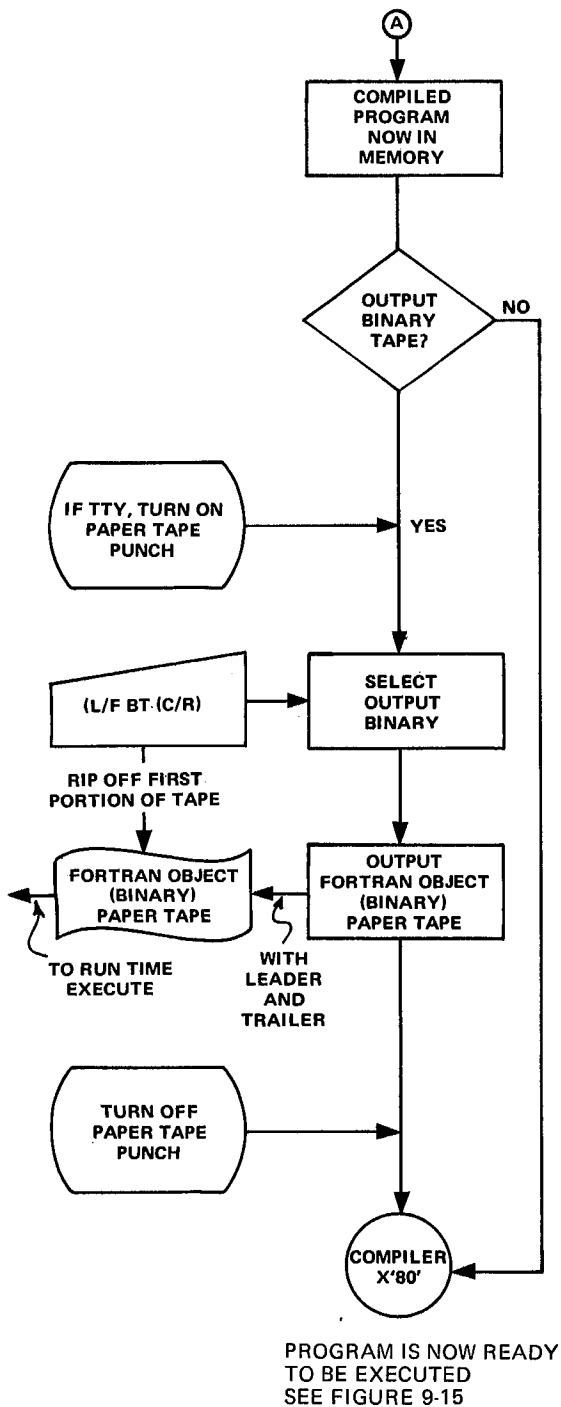
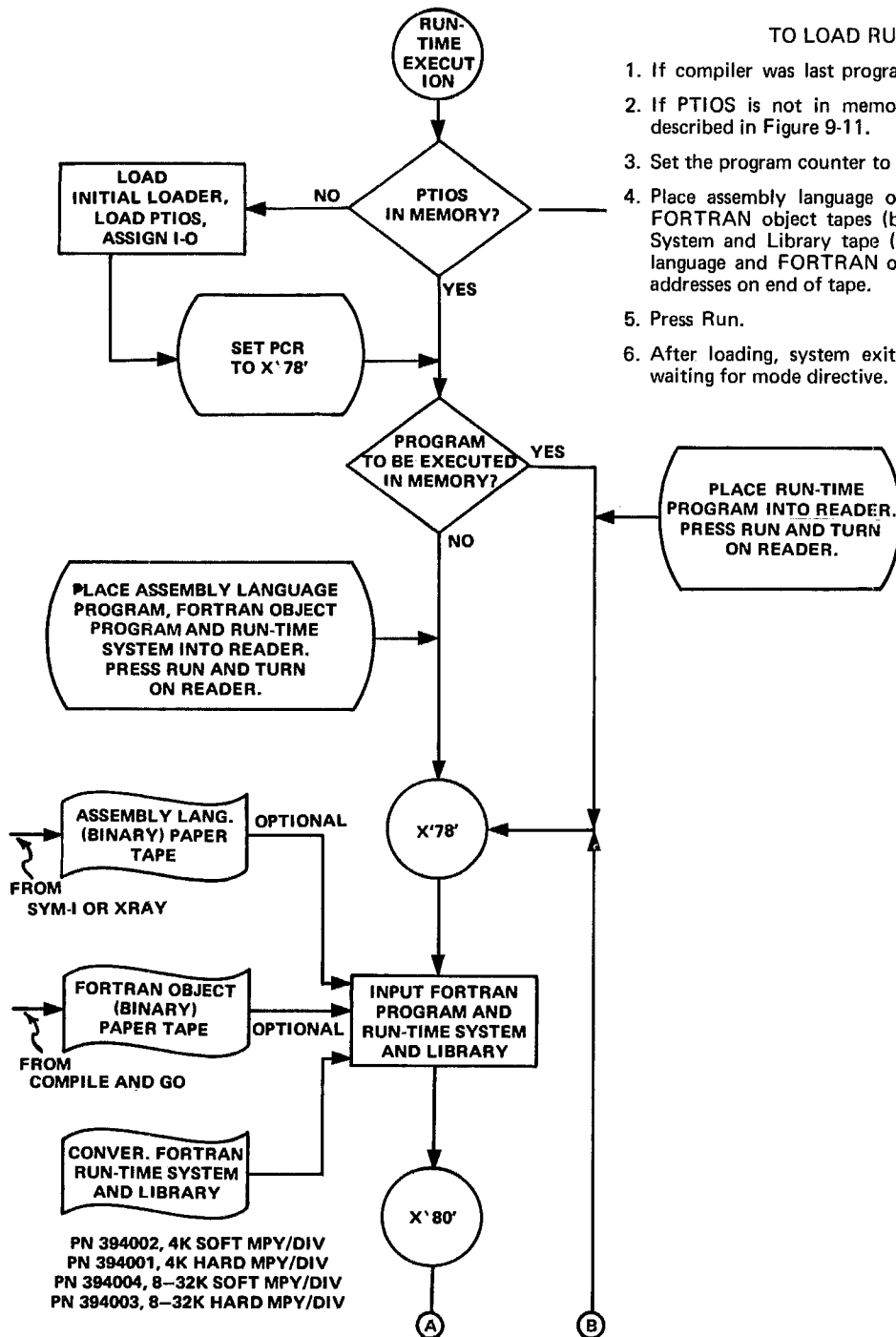


Figure 9-14. Conversational FORTRAN Compile and Go (Sheet 1 of 2)



The FORTRAN object paper tape will have the characters (L/F) BT (CR) on the front followed by leader, the binary program and trailer. Tear off the first characters.

Figure 9-14. Conversational FORTRAN Compile and Go (Sheet 2 of 2)



TO LOAD RUN-TIME SYSTEM:

1. If compiler was last program executed, PTIOS is in memory.
2. If PTIOS is not in memory, it must be loaded by method described in Figure 9-11.
3. Set the program counter to X'78'.
4. Place assembly language object tapes (binary), followed by FORTRAN object tapes (binary), followed by the Run-Time System and Library tape (binary) into the reader. Assembly language and FORTRAN object tapes must not have transfer addresses on end of tape.
5. Press Run.
6. After loading, system exits to X'80' and types two blanks waiting for mode directive.

Figure 9-15. Conversational FORTRAN Run-Time Execution (Sheet 1 of 2)

TO INPUT MODE DIRECTIVES:

1. Type line-feed followed by one character directive followed by a carriage return.

Directive	Response
C	Trace only control statements
V	Trace only values of variables on the left side of replacement statement
B	Trace both control and values
N	No trace (run-time system normally set here)
G	Go execute program
E	Exit to absolute load routine (X'78')

3. Each trace output is contained on a single line. Trace output format is:

Output	Statement Executed
GO XXXXX	GO to XXXXX or Computed GO TO
IF XXXXX	IF statement, control is transferred to statement XXXXX.
CALL NNNNN	CALL NNNNN(a1, . . . ,aN)
RETURN	RETURN
DO XXXXX	Executing a DO loop, XXXXX is the value of the DO index.
**	The DO index has equaled or exceeded its terminal value.

The trace output for the results of replacement statements depends on the data type.

Output	Data Type
±XXXXX	Integer variable
±.XXXXXXXXXX±EE	Real variable, EE is the decimal exponent.

1. The FORTRAN object program and any optional assembly language programs will execute with results output on the logical unit specified in the FORTRAN Program.

2. Run-time error messages (see Appendix N-6) are output on the teletype printer.

1. Upon executing a stop statement, the message STOP will be printed on the run-time listing.

2. The program exits to the absolute load routine (X'78') of PTIOS at which time more FORTRAN object program tapes may be loaded and executed without re-loading the run-time system and library.

3. The compiler may also be loaded in order to prepare another FORTRAN program.

1. By throwing Sense Switch 3 up and then down, the trace mode may be altered during execution time.

2. The run-time system returns control to the point in the FORTRAN Program where execution was interrupted.

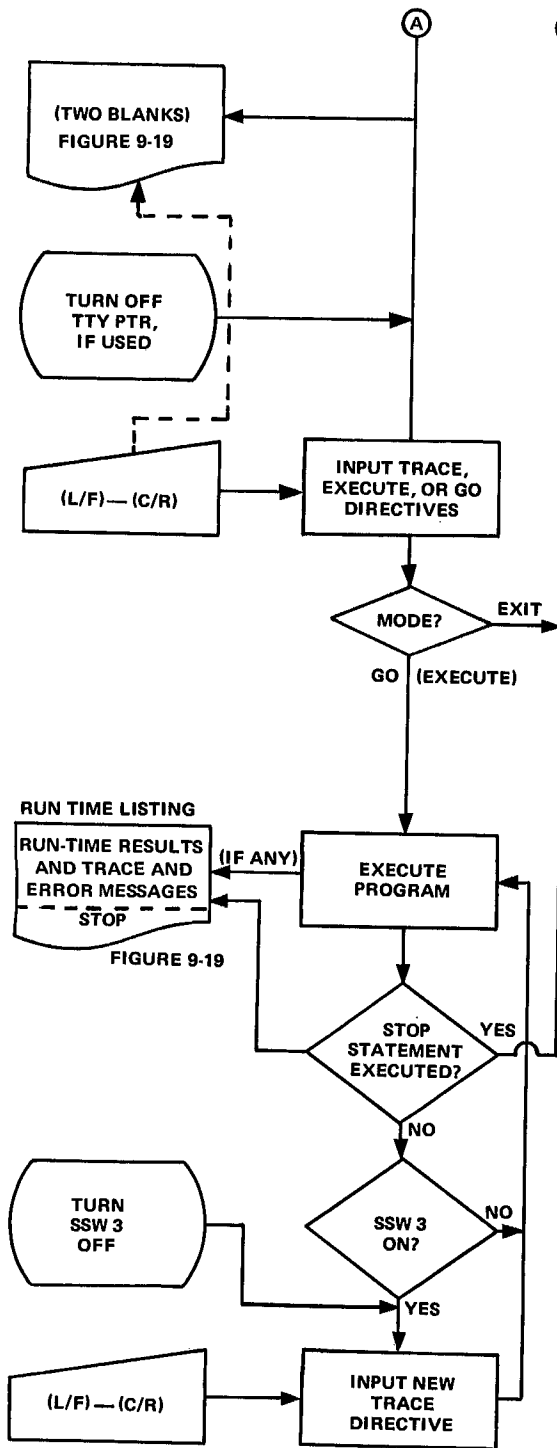


Figure 9-15. Conversational FORTRAN Run-Time Execution (Sheet 2 of 2)

```

C      CONVERSATIONAL FORTRAN SAMPLE PROGRAM
C
C      AMORTIZATION OF MORTGAGE LOAN
C
C
7      PAUSE
1      WRITE (13,1)
1      FORMAT (16HINTEREST RATE = )
DD.....
      READ (13,2) AINT
2      FORMAT (F6.4)
9      AINT = AINT/12.
10     WRITE (13,3)
3      FORMAT (7HTIME = )
      READ (13,2) TIME
      WRITE (13,4)
4      FORMAT (12HPRINCIPAL = )
      READ (13,2) P
      AM=(AINT*(1.+AINT)**TIME)/(((1.+AINT)**TIME)-1)
MX.....
      S = AM * P
      WRITE (13,5)
5      FORMAT (34HNUMBER TOTAL INTEREST BALANCE )
      AINTOT = 0.
      STOT = 0
      STOT = 0.
      ITIME = TIME
      DO 6 N=1,ITIME,1
      STOT = STOT + S
      AINTOT = AINT * P + AINTOT
      P = P - S + AINT * P
6      WRITE (13,8) N,STOT,AINTOT,P
8      FORMAT (15,F9.2,F9.2,F9.2)
      GO TO 10
      GO TO 7
      END
LA     70

```

Figure 9-16. Conversational FORTRAN Input Listing

```

C      CONVERSATIONAL FORTRAN SAMPLE PROGRAM
C
C      AMORTIZATION OF MORTGAGE LOAN
C
C
7      PAUSE
1      WRITE (13,1)
      HEAD (13,2) AINT
2      FORMAT (F6.4)
9      AINT = AINT/12.
10     WRITE (13,3)
3      FORMAT (7HTIME = )
      READ (13,2) TIME
      WRITE (13,4)
4      FORMAT (12HPRINCIPAL = )
      READ (13,2) P
      S = AM * P
      WRITE (13,5)
5      FORMAT (34HNUMBER TOTAL INTEREST BALANCE )
      AINTOT = 0.
      STOT = 0
      STOT = 0.
      ITIME = TIME
      DO 6 N=1,ITIME,1
      STOT = STOT + S
      AINTOT = AINT * P + AINTOT
      P = P - S + AINT * P
6      WRITE (13,8) N,STOT,AINTOT,P
8      FORMAT (15,F9.2,F9.2,F9.2)
      GO TO 10
      GO TO 7
      END
LA     70

```

Figure 9-17. Conversational FORTRAN PREP Listing


```

C      CONVERSATIONAL FORTRAN SAMPLE PROGRAM
C
C      AMORTIZATION OF MORTGAGE LOAN
C
7      PAUSE
      WRITE (13,1)
1      FORMAT (16HINTEREST RATE = )
      READ (13,2) AINT
2      FORMAT (F6.4)
9      AINT = AINT/12.
10     WRITE (13,3)
3      FORMAT (7HTIME = )
      READ (13,2) TIME
      WRITE (13,4)
4      FORMAT (12HPRINCIPAL = )
      READ (13,2) P
      AM=(AINT*(1.+AINT)**TIME)/(((1.+AINT)**TIME)-1.)
      S = AM * P
      WRITE (13,5)
5      FORMAT (34HNUMBER TOTAL INTEREST BALANCE )
      AINTOT = 0.
      STOT = 0.
      ITIME = TIME
      DO 6 N=1,ITIME,1
      STOT = STOT + S
      AINTOT = AINT * P + AINTOT
      P = P - S + AINT * P
6      WRITE (13,8) N,STOT,AINTOT,P
8      FORMAT (15,F9.2,F9.2,F9.2)
      GO TO 7
      END

```

Figure 9-18. Conversational FORTRAN Compiler Listing

```

NO TRACE

PAUS
INTEREST RATE =
.1000
TIME =
12.0000
PRINCIPAL =
3000.00
NUMBER TOTAL INTEREST BALANCE
1 263.74 24.99 2761.25
2 527.49 48.01 2520.51
3 791.24 69.01 2277.77
4 1054.99 87.99 2033.00
5 1318.73 104.93 1786.19
6 1582.48 119.82 1537.33
7 1846.23 132.63 1286.40
8 2109.98 143.35 1033.37
9 2373.72 151.96 778.23
10 2637.47 158.45 520.97
11 2901.22 162.79 261.56
12 3164.97 164.97 -.000
PAUS

TRACE MODE

PAUS
INTEREST RATE =
.1000
.833333331 -2
TIME =
12.0000
PRINCIPAL =
3000.00
.879158940 -1
.263747682 3
NUMBER TOTAL INTEREST BALANCE
.000000000 0
.000000000 0
12
1
DO 1
.263747682 3
.249999999 2
.276125232 4
1 263.74 24.99 2761.25
DO 2
.527495365 3
.480104359 2
.252051506 4
2 527.49 48.01 2520.51

DO 3
.791243048 3
.690147280 2
.227777168 4
3 791.24 69.01 2277.77
DO 4
.105499073 4
.879961587 2
.203300542 4
4 1054.99 87.99 2033.00
DO 5
.131873841 4
.104937870 3
.178619945 4
5 1318.73 104.93 1786.19
DO 6
.158248609 4
.119822866 3
.153733676 4
6 1582.48 119.82 1537.33
DO 7
.184623377 4
.132634005 3
.128640022 4
7 1846.23 132.63 1286.40
DO 8
.210998146 4
.143354007 3
.103337254 4
8 2109.98 143.35 1033.37
DO 9
.237372913 4
.151965444 3
.778236298 3
9 2373.72 151.96 778.23
DO 10
.263747682 4
.158450747 3
.520973918 3
10 2637.47 158.45 520.97
DO 11
.290122450 4
.162792196 3
.261567684 3
11 2901.22 162.79 261.56
DO 12
.316497218 4
.164971927 3
-.267215073 -3
12 3164.97 164.97 -.000
**
GO 7
PAUS

```

Figure 9-19. Conversational FORTRAN Run-Time and Trace

**STANDARD SYSTEM SOFTWARE
OPERATION SUMMARY**

Table 9-2. Standard Operating System Software

Description	Label	Part No.	Origin/End	Start	Size	Loaded-By
			(HEX)	(HEX)	(DEC)	
Monitor	SYSMON	(1)	(2)	(2)	(2)	INITLOAD
XRAY EXEC-STANDARD	XRAY	391305	Relocatable		0475	MONITOR
Relocating Loader-Standard	RELOADS	390013	0080/05C0	0080	1345	MONITOR
SYM-I/PREP	SYM1	390470	0580/0FF1	0800	2626	INITLOAD
SYM-II	SYM2	391878				MONITOR
Trace/Debug Package	DEBUG	391082	Relocatable		0768	RELOADS
Symbolic Program Editor	SYMED	394873	0080/0474	00E5	1013	MONITOR
Absoluter, Linking-Standard	ABSOL	393254	0080/0709	0080	1674	MONITOR
Initial Loader-Paper Tape	INITLOAD	393260	0000/007F	0009	0128	BOOTSTRAP
Initial Loader-Cards	INITLOAD	393259	0000/007F	0009	0128	BOOTSTRAP
Math Library						
Soft, MPY/DIV	(See	393325	Relocatable	(See		RELOADS
Hard, MPY/DIV	Appendix J)	393977	Relocatable	Appendix J)		RELOADS
Conver. FORTRAN Compiler	CF	394005	0080/0CEA	0080	3179	MONITOR
Conver. FORTRAN Library						
8K-32K, Soft, MPY/DIV	CFR	394004	0080/0CB6	0080	2871	MONITOR
8K-32K, Hard, MPY/DIV	CFR	394003	0080/0CA4	0080	3109	MONITOR
FORTRAN IV Compiler						
8K	FORTRAN 4	393297	0080/17F7	0080	6008	MONITOR
12K-32K	FORTRAN 4	392957	0080/1DDC	0080	7517	MONITOR
FORTRAN IV Run-Time Library						
Soft, MPY/DIV	(See	393945	Relocatable	(See		RELOADS
Hard, MPY/DIV	Appendix J)	393944	Relocatable	Appendix J)		RELOADS
<p>(1) Each monitor is custom prepared for each installation according to the peripheral device configuration. See shipping manifest for part number.</p> <p>(2) A typical STANDARD system monitor for a machine with 16K memory, 16 interrupt levels, an ASR-33 teletype, and a high-speed paper tape reader requires 1077 decimal locations (3BCB to 3FFF).</p>						

9-3 STANDARD OPERATING SYSTEM OPERATION

9-3.1 General

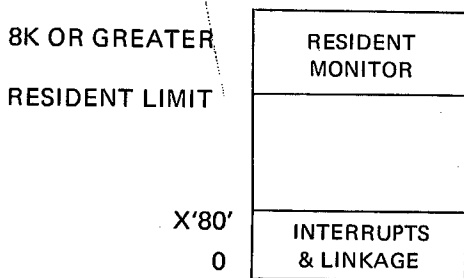
The Standard Operating System is primarily a "hands on" operating system, requiring manual operations to sequence the system through the tasks. Thus, the user must load the proper programs at the proper time, and perform those functions which would be performed automatically if the system were supported by automatic devices; such as, disc and magnetic tape. This section details some of the manual operations. It is not possible to describe the sequence of every possible task, but it is necessary that the operator become familiar with the general principals of the operating system, and realize that a full understanding of the system requires study and experience.

9-3.2 Bootstrap Loading The System

SOS first loads the system monitor into core using the Bootstrap Loading procedure. Follow the instructions for loading and executing Initial Loader - Card (PN 393259) or Initial Loader - Paper Tape (PN 393260). These instructions are found in Section 9-2. If the input device is cards, the Initial Loader and monitor decks should be combined. For paper tape, the two may be separate, or may be combined by copying the two into a single paper tape. Then the bootstrap procedure described in Section 9-2 will load the monitor into core and execute the monitor initialization routine. The monitor will type:

LOAD

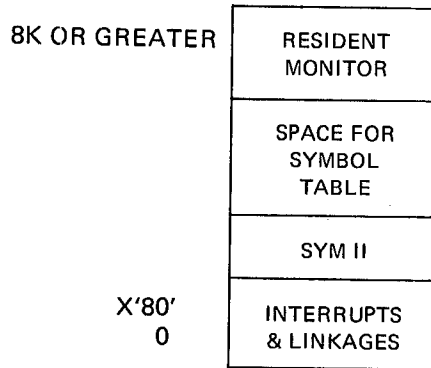
on the console teletype, and the computer will halt. At this time core memory will be configured as shown below:



9-3.3 System Processor Loading

After the computer halts, the operator may load any absolute program into the computer and execute it by placing the binary deck or tape in the reader and depressing the RUN switch. The program will be loaded and, if an execution address was specified by the original source language program, will begin execution.

Refer to Section 8-5.1 for a description of absolute program text, and of the behavior of the absolute loader for programs having checksum errors or deviations from the standard text format. Among the system processors which could be loaded at this time are SYM I, the basic assembler, SYM II, the two-pass assembler, FORTRAN IV, SYMED, the symbolic editor, and LINKABS, the linking absolute and Conversational FORTRAN. Any user created absolute program may be loaded and executed also. Refer to the documentation of the associated processor for operating instructions. For example, if the user loads SYM II, core memory will be configured as shown below:



9.3.4 Loading The System Executive

To use Standard XRAY, or to load a relocatable object program, the user must load the system processor called:

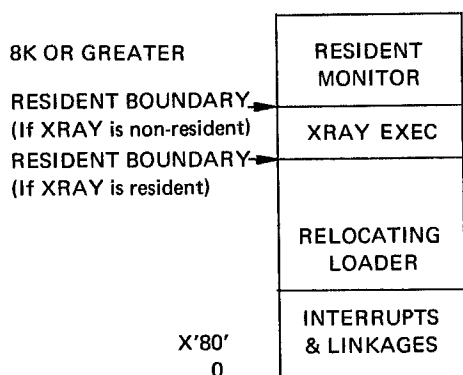
RELOADS - XRAYS

This program combines the relocating loader and the Standard XRAY EXEC on a single paper tape. For card systems, the user constricts this processor by stacking the Standard XRAY deck behind the Relocating Loader - Standard deck. The absolute

loader loads the relocating loader which loads XRAY. XRAY begins execution by listing

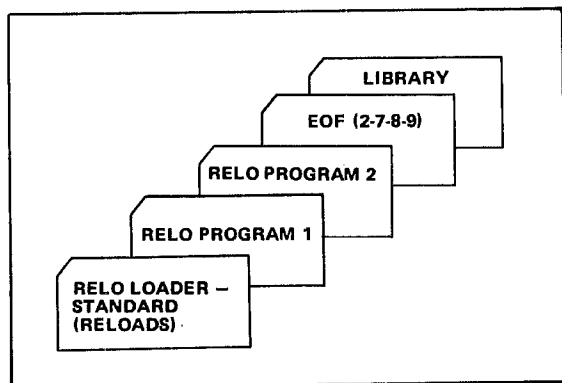
NEXT

on the teletype. The directives described in Appendix M are then available to the user. With experience XRAY Executive functions can be supplanted by simple manual operations. For example, an absolute load operation can always be obtained by setting the computer program counter to X'78'. The core configuration after loading XRAY is shown below:



9.3.5 Loading A Relocatable Program

A relocatable program may be loaded in either of two ways. The first is to load XRAY and use the :IL directive. The second does not require XRAY, and may be used whenever no XRAY facilities (I/O reassignment, loader option, etc.) are required. The program, visualized below as a deck sequence, is loaded by the absolute loader.



9-3.6 Manual Intervention

The system includes certain manual intervention features. These consist of fixed system link points which may be set in the program counter after halting and re-setting the computer. When RUN is depressed, the requested function will be accomplished. The following functions can be requested: system recovery, exit current task, relocatable program restart, and absolute load.

9-3.6.1 System Recovery

The operator may restart the system by setting the program counter to the last cell of memory minus 2. This would be X'1FFD' for an 8K system, X'3FFD' for a 16K system. When RUN is depressed, the interrupt processing system is reset, XRAY is made non-resident, and the monitor types LOAD and halts.

9-3.6.2 Exit Current Task

To EXIT from the current task or interrupt level, the operator may set the program counter to X'40'. This is the entry point to the system EXIT routine. If XRAY is resident, control will transfer there; if XRAY is not resident, the monitor will type LOAD and halt.

9-3.6.3 Relocatable Program Restart

Any program loaded by the relocating loader may be restarted by setting the program counter to X'7A'. This is primarily a "hands on" debugging aid and is only operational if the program has not exited to the system. Once the program has exited, it cannot be reliably restarted.

9-3.6.4 Absolute Load Routine

As a convenience for manual operation, the system absolute load routine is available as an operator function. This routine loads programs in the absolute format from the BIN device (logical unit 4). It is executed by setting the program counter to X'78' and depressing RUN. Absolute programs have the following format, representing the paper tape records:

Absolute programs have the following format, representing the paper tape records:

Loc 1 Record	Program 1 Record	//	Loc n Record	Program n Record	E O F	Execution Address Record
-----------------	---------------------	----	-----------------	---------------------	-------------	--------------------------------

Program 1 is read into core at location Loc i. The end-of-file record is a single ASCII BELL character for paper tape, or a 2-7-8-9 punch in column 1 for cards. Loc and execution addresses are one word binary records. Absolute programs in this format are produced by SYM I, SYM II, the linking Absoluter, and by the program punch-out option of the Trace/Debug package. The absolute loader assumes that the last byte loaded is a checksum byte. It recomputes this checksum, and if there is a mis-match, types

ACK!

(Absolute Checksum!). Checksums are supplied by SYM I, SYM II, and the Absoluter. Trace/Debug does not supply checksums, however, and programs punched from core by Trace/Debug will indicate a checksum error, which should be ignored.

If successive file marks are read following a program record, the absolute load routine halts. If RUN is depressed, the absolute load routine will continue program loading. If a single file mark followed by a one-word record containing an execution address is encountered, the absolute load routine will transfer to that execution address.

If file marks and execution address are missing entirely, the loader will continue to select the input device for more records. The user may halt and reset the computer, and set the program counter to the program entry point and manually begin execution by depressing RUN.

9-3.7 System Messages

Certain messages for the operator are output on the console teletype. Typically these messages signal trouble or error conditions. This group of messages is output by the Resident Monitor, by XRAY, and by certain system processors.

The following more common messages are briefly explained with regard to operator response.

Further clarification can be found in the documentation relevant to the processor which outputs the message.

9-3.7.1 I/O Not Ready Message

Most messages signal that a required I/O device is not on line. This group outputs a two-letter mnemonic identifying the device, and sounds the teletype bell twice to attract the operator's attention, typically:

LP (Bell) (Bell)	-	Line Printer
CR (Bell) (Bell)	-	Card Reader
CP (Bell) (Bell)	-	Card Punch
Mn (Bell) (Bell)	-	Magnetic Tape Unit n
MnW (Bell)	-	Magnetic Tape Unit n is write protected and writing.

These signals are repeated at five second intervals until the requested device is placed on line.

9.3.7.2 SYM-II Read/Write Error Message

The messages:

IOE1

and

IOE2

signal read and write errors in SYM II, respectively. If IOE2 occurs during Pass 1 and the scratch device is the disc, this indicates that the size of disc file 1 is insufficient to hold the source text. The file should be increased in size and the assembly re-run. IOE1 during Pass 1 means that the source text has been misread from the PRIN device. During Pass 2, the error has occurred reading from the scratch device. This error signals a hardware malfunction.

9-3.7.3 Processor Pause Message

The message

TOG3

is a pause message output by the system loader and by other system processors. It may be accomplished by error messages on the system list device; such as, CK or RD, in which instance the pause is

to permit the operator to manually backspace on record to allow the processor to retry. Refer to the documentation for the processor which actually outputs the message for additional clarification of the operation expected. In addition, the system loader uses the pause between loading program modules, if Sense Switch 0 is set, and at the end of load, if system flag 5 is set. The operator may cause the system to continue by Setting Switch 3 to the opposite state, and then returning to the Present state. This procedure is called *toggling switch 3*.

9-3.7.4 FORTRAN Pause Message

The messages

STOP

and

PAUS(E)

are output by the FORTRAN IV Run-Time package. PAUSE will require toggling switch 3 to continue.

9-3.7.5 System Ready Message

The message

NEXT

is output by Standard XRAY whenever a program exits to XRAY. It signals the beginning of a new job and is output for job logging purposes, and serves to request a directive.

9-3.7.6 Assembler In Manual Mode Message

The message

PAS

is output by the SYM II assembler when the assembler is not running in the automatic mode. The operator may type 1, 2, or X, unless the system is in batch mode, in which instance these would be read from the card reader.

**MAGNETIC TAPE OPERATING SYSTEM
OPERATION SUMMARY**

Table 9-3. Magnetic Tape Operating System Software

Description	Label	Part No.	Origin/End	Start	Size	Loaded-By
			(HEX)	(HEX)	(DEC)	
Monitor		(1)	(2)	(2)	(2)	BOOTSTRAP
Mag Tape XRAY EXEC	MTXRAY	393979	0080/055D	0080	1246	MONITOR
Mag Tape SYSGEN	MTSYSG	393992	0100/05D5	0100	1238	MONITOR
Mag Tape QPROC	MQLP	393980	0600/06EF	0600	0240	MONITOR
Relocating Loader-Standard	RELOADS	390013	0080/05C0	0080	1345	MONITOR
SYM-I/PREP	SYM1	390470	05B0/0FF1	0800	2626	INITLOAD
SYM-II	SYM2	391878	0080/1201	0080	4482	MONITOR
Trace/Debug Package	DEBUG	391082	Relocatable		0768	RELOADS
Symbolic Program Editor	SYMED	394873	0080/0474	00E5	1013	MONITOR
System Editor	EDITOR	391915	0080/09FC	0080	2429	MONITOR
Absoluter, Linking-Standard	ABSOL	393254	0080/0709	0080	1674	MONITOR
Initial Loader-Paper Tape	INITLOAD	393260	0000/007F	0009	0128	BOOTSTRAP
Initial Loader-Cards	INITLOAD	393259	0000/007F	0009	0128	BOOTSTRAP
Magnetic Tape Dump	MAGDMP	390540	0800/0A14	0868	0533	MONITOR
Conver. FORTRAN Compiler	CF	394005	0080/0CEA	0080	3179	MONITOR
Conver. FORTRAN Library						
8K-32K, Soft, MPY/DIV	CFR	394004	0080/0CB6	0080	2871	MONITOR
8K-32K, Hard, MPY/DIV	CFR	394003	0080/0CA4	0080	3109	MONITOR
FORTRAN IV Compiler						
8K	FORTRAN4	393297	0080/17F7	01CE	6008	MONITOR
12K-32K	FORTRAN4	392957	0080/1DDC	01CE	7517	MONITOR
FORTRAN IV Run-Time Library						
Soft, MPY/DIV	(See	393945	Relocatable		(See	RELOADS
Hard, MPY/DIV	Appendix J)	393944	Relocatable		Appendix J)	RELOADS
Math Library						
Soft, MPY/DIV	(See	393325	Relocatable		(See	RELOADS
Hard, MPY/DIV	Appendix J)	393977	Relocatable		Appendix J)	RELOADS
<p>(1) Each monitor is custom prepared for each installation according to the peripheral device configuration. See shipping manifest for part number.</p> <p>(2) A typical Magnetic Tape Operating System monitor for a machine with 8K memory, 4 interrupt levels, a DIO magnetic tape and an ASR-33 teletype requires 1225 decimal locations (1B37-1FFF)</p>						

9-4 MAGNETIC TAPE OPERATING SYSTEM OPERATION

9-4.1 General

The MTOS system control directives permit the user to set up a batch of many jobs to be executed without manual intervention. However, at times manual intervention may be required or the most convenient method of operation. The MTOS system includes facilities for operator alert and intervention. This section describes the manual operations required to operate the MTOS system.

9-4.2 Bootstrap Loading The System

The MTOS system resides permanently on the system tape (magnetic tape 0) after system generation. It may be re-loaded from the tape at any time by placing the tape at load point, depressing computer RESET, the tape controller LOAD (DMA), or the MAG TAPE BOOTSTRAP (DIO), and computer RUN switches respectively. At this time, the monitor is completely re-initialized. As an operator recovery method, bootstrapping the system should be considered only when other recoveries have failed, since all I/O device assignments revert to their standard assignments. For example, if a program has inadvertently destroyed the monitor, it will be necessary to bootstrap the system.

9-4.3 Operator Interrupt

The operator interrupt service routine is included in the Resident Monitor *only* when the manual interrupt hardware is included on the teletype console.

When the manual interrupt button is depressed, the message

OPRI

is typed to acknowledge the interrupt. If the teletype is busy doing an input or an output, the operator interrupt will wait for it to become available. For an input operation, this means that the current input request must be completed before the operator interrupt will acknowledge. During this waiting period, the interrupted computer operations are suspended. As soon as the

operator interrupt has acknowledged, the teletype is selected to input two characters and the interrupted task is restored and continues execution until the operator has typed the two character directive.

The operator interrupt cannot acquire the teletype when the operator interrupt has interrupted the teletype's interrupt service routine. However, the operator interrupt is acknowledged by depressing the switch a second time.

9-4.3.1 Operator Interrupt Directives

Operator interrupt directives consist of two characters typed *without preceding line feed nor following carriage return*. The following directives are acceptable:

XX — This transfers control of the system to XRAY after the current task exits. The system configuration may be changed through XRAY directives. The insertion of XRAY in the system queue at the highest priority does not affect the execution of any tasks in progress. See Appendix M.

TT — The current task is terminated after a one second delay to allow the I/O in progress to complete, and transfer control to the next task in the queue.

TX — The current task is terminated after a one second delay to allow the I/O in progress to complete, and transfer control to XRAY.

Character pairs not of the form *X or T*, where * is any character, are ignored. This feature allows the operator to control the I/O system. Teletype output operations may be suspended by depressing the manual interrupt. Then if an invalid directive is input (e.g. space-space), the operation will continue. This may be used to stop the repetitive device-not-ready message output by the I/O system, while the operator readies the device.

A program in progress may be suspended by depressing the manual interrupt switch a second

time after 'OPRI' is typed. This will cause the system to hang up in the operator interrupt service routine waiting for the teletype. To continue, the operator enters no-op directives (i.e., space-space).

9-4.4 Manual Intervention

If the operator interrupt is not a part of the hardware system, or if the operator interrupt has been disabled, perhaps inadvertently by some user program, or if the system is being operated through the console switches, as in hands-on single step debugging, manual intervention will be required. These consist of fixed system link points which may be set in the program counter after halting and resetting the computer. Depressing RUN initiates the requested function. The following functions can be requested: system recovery, exit current task, relocatable program restart, and absolute load.

9-4.4.1 System Recovery

The operator may restart the system by setting the program counter to the (last cell of memory -2). This would be X'1FFD' for an 8K system, X'3FFD' for a 16K system. When RUN is depressed, the interrupt processing system is reset, and the operator and power fail safe interrupts are re-enabled if present in the system, and the system proceeds to the next task in the queue. If no tasks are pending and if the system has an operator interrupt, the system will wait in an idle loop. The operator interrupt may then be used to inject XRAY into the queue. If there is no operator interrupt, the system will not idle, but will instead automatically inject XRAY into the queue.

9-4.4.2 Exit Current Task

To EXIT from the current task or interrupt level, the operator sets the program counter to X'40'. This is the entry point to the system EXIT routine. If the operator has halted the system while it was servicing an interrupt, that interrupt level will be immediately restored and processing will continue.

9-4.4.3 Relocatable Program Restart

Any program loaded by the relocating loader may

be restarted by setting the program counter to X'7A'. This is primarily a hands-on debugging aid and only operates if the program has not exited to the system. Once the program has exited, it cannot be reliably restarted.

9-4.4.4 Absolute Load Routine

The system absolute load routine facilitates manual operation. This routine loads programs in the absolute format from the BIN device (logical unit 4). It is executed by setting the program counter to X'78' and depressing RUN. This is a strictly manual mode of system operation. The format of absolute programs is described in Section 8-5.1.

If successive file marks are read following a program record, the absolute load routine halts. If RUN is depressed, the absolute load routine continues program loading. If a single file mark followed by a one word record containing an execution address is encountered, the absolute load routine transfers to that execution address.

9-4.5 System Messages

Certain messages for the operator are output on the console teletype. Typically the messages signal error conditions. These messages are output by the Resident Monitor, XRAY, the Queue Processor, and certain system processors.

Each of the messages are briefly described below. Further clarification can be found in the documentation relevant to the processor outputting the message.

9-4.5.1 I/O Not Ready Message

Messages which signal that a required I/O device is not on line are the most common. These messages output a two-letter mnemonic identifying the device, and sounding the teletype bell twice to attract the operator's attention. The following may be encountered:

LP (Bell) (Bell)	-	Line Printer
CR (Bell) (Bell)	-	Care Reader
CP (Bell) (Bell)	-	Card Punch

Mn (Bell) (Bell) - Magnetic Tape Unit n
MnW (Bell) (Bell) - Write protected unit is selected for output.

These signals are repeated at five second intervals until the requested device is placed on line.

9-4.5.2 Error Message

The message

EROR

is output by XRAY to alert the operator that an error has occurred in the processing of the previous phase of a job, and that he should examine the listed output to ascertain that the error was a programmer's error and not a system malfunction or operator error.

9-4.5.3 SYM-II Read/Write Error Messages

Messages

IOE1

and

IOE2

signal read and write errors in SYM II, respectively. If 'IOE2' occurs during Pass 1 and the scratch device is a tape unit, this indicates that the size of the scratch tape is insufficient to hold the source text. A larger tape should be mounted and the assembly re-run. 'IOE1' during pass 1 means that the source text has been misread from the PRIN device. During pass 2, the error has occurred reading from the scratch device. This error signals a hardware malfunction.

9-4.5.4 Processor Pause Message

A pause message

TOG3

output by the system loader and other system processors is initiated by error messages on the system list device; such as, "CK" or "RD," in which instance the pause permits the operator to manually backspace on the record to allow the processor to re-try. For additional information, refer to the documentation for the processor outputting the message. The system loader also uses this pause between loading program modules, if Sense Switch 0 is set, and at the end of load, if system flag 5 is set. The operator may cause the

system to continue by setting switch 3 to the opposite state, and then returning to the present state. This procedure is called "toggling switch 3."

9-4.5.5 FORTRAN Pause Messages

The messages

STOP

and

PAUS(E)

are output by the FORTRAN Run-Time Package. "PAUSE" will require toggling switch 3 or pressing RUN to continue.

9-4.5.6 System Ready Messages

The message

NEXT

is output by XRAY whenever a :NJ or :EOJ directive is encountered. It signals the beginning of a new job and is output for job logging.

The message

REDY

is output by XRAY whenever the executive comes in to execution during the normal operating sequence and expects a directive from the system teletype.

9-4.5.7 XRAY Ready Message

The message

XRAY

is output by XRAY when it makes an extraordinary appearance in the operating sequence, for instance, having been queued by the operator interrupt. In this event the teletype is always the input device, even if the system is in batch mode and directives and messages are not listed on the LIST device.

9-4.5.8 Assembler in Manual Mode Message

The message

PAS?

is output by the SYM II assembler when the assembler is not running in the automatic mode. The operator may type 1, 2, or X, unless the system is in batch mode, in which instance these would be read from the card reader.

**REAL-TIME OPERATING SYSTEM
SOFTWARE OPERATION SUMMARY**

Table 9-4. Real-Time Operating System Software

Description	Label	Part No.	Origin/End	Start	Size	Loaded-By
Monitor-RTOS	SYSMON	(1)	(HEX)	(HEX)	(DEC)	DBST
RTOS-Real Time XRAY EXEC	RTXRAY	391881	0080/07F8	0080	1913	MONITOR
Relocating Loader-Disc	RELOADD	390012	0080/0626	0080	1447	MONITOR
Initial Loader-Paper Tape	INITLOAD	393260	0000/007F	0009	0128	BOOTSTRAP
Initial Loader-Cards	INITLOAD	393259	0000/007F	0009	0128	BOOTSTRAP
Queue Processor	QLP	391879	Relocatable		0518	MONITOR
Trace/Debug Package	DEBUG	391082	Relocatable		0768	RELOADD
SYM-I/Prep	SYM1	390470	05B0/0FF1	0800	2626	INITLOAD
SYM-II	SYM2	391878	0080/1201	0080	4482	MONITOR
RTOS Disc Bootstrap	DBST	391917	0000/002E	0000	0047	DISC-BOOT ³
Symbolic Program Editor	SYMED	394873	0080/0474	00E5	1013	MONITOR
System Editor	EDITOR	391915	0080/09FC	0080	2429	MONITOR
RTOS SYSGEN 1	SYSG1	391876	0080/0461	0080	0994	MONITOR
RTOS SYSGEN 2	SYSG2	391877	0080/0438	016A	0953	MONITOR
Resident Loader	RESLOAD	391916	0080/06D2	0080	1619	MONITOR
Library Extension Processor	EXTEND	391914	0080/043F	0080	0959	MONITOR
RTOS Installation Program	SYSINST	392341	0080/00D9	0080	0090	MONITOR
Disc Dump	DSKDMP	390539	0800/0989	087D	0442	MONITOR
Absoluter, Linking-Disc	LINKABS	393255	0080/0756	0080	1751	MONITOR
Magnetic Tape Dump	MAGDMP	390540	0800/0A14	0868	0533	MONITOR
Card Sequencer	SEQ	392920	Relocatable		0276	RELOADD
Conver. FORTRAN Compiler	CF	394005	0080/0CEA	0080	3179	MONITOR
Conver. FORTRAN Library						
8K-32K, Soft, MPY/DIV	CFR	394004	0080/0CB6	0080	2871	MONITOR
8K-32K, Hard, MPY/DIV	CFR	394003	0080/0CA4	0080	3109	MONITOR
FORTRAN IV Compiler						
8K	FORTRAN4	393297	0080/17F7	01CE	6008	MONITOR
12K-32K	FORTRAN4	392957	0080/1DDC	01CE	7517	MONITOR
FORTRAN IV Run-Time Library						
Soft, MPY/DIV	(See	393945	Relocatable	(See		RELOADD
Hard, MPY/DIV	Appendix J)	393944	Relocatable	Appendix J)		RELOADD
Math Library						
Soft, MPY/DIV	(See	392325	Relocatable	(See		RELOADD
Hard, MPY/DIV	Appendix J)	393977	Relocatable	Appendix J)		RELOADD
<p>(1) Each monitor is custom prepared for each installation according to the peripheral device configuration. See shipping manifest for part number.</p> <p>(2) A typical RTOS monitor for a machine with 16K memory, 16 interrupt levels, a high-speed paper tape reader and punch, 2 disc memories, 3 DMA magnetic tapes, a card reader and punch, an operator interrupt, a line printer and power fail-safe requires 2328 decimal locations (36E8-3FFF).</p> <p>(3) Disc-Boot is the LOAD button on the disc controller.</p>						

9-5 REAL-TIME OPERATING SYSTEM OPERATION

9-5.1 General

The RTOS system operates automatically. The system control directives permit the operator to set up and execute a batch of jobs without operator intervention. However, at times manual intervention may be required or may be the most convenient method of operation. The RTOS system includes the necessary facilities for operator alert and intervention. This section details the manual operations required to operate the RTOS system.

9-5.2 Bootstrap Loading The System

The RTOS system resides permanently on the disk after system generation. It may be re-loaded from the disk at any time by depressing computer RESET, the disk controller LOAD button, and computer RUN. At this time the monitor is completely re-initialized. Any resident tasks sub-routines previously defined are lost. However, any tasks in the system queue are preserved through the bootstrap operation and, after the system is bootstrapped, it will initiate execution of the highest task in the queue. As an operator recovery method, bootstrapping the system should be considered only when other recoveries have failed, since resident areas are lost and all I/O device assignments revert to their standard assignments. For example, if a program has inadvertently destroyed the monitor, it will be necessary to bootstrap the system.

9-5.3 Operator Interrupt

The operator interrupt service routine is included in the Resident Monitor only when the manual interrupt hardware is included on the teletype console.

When the manual interrupt button is depressed, the message

'OPRI'

is typed to acknowledge the interrupt. If the teletype is busy doing an input or an output, the

operator interrupt will wait for it to become available. For an input operation this means that the current input request must be completed before the operator interrupt will acknowledge. During this waiting period, the interrupted computer operations are suspended. As soon as the operator interrupt has acknowledged, the teletype is selected to input two characters and the interrupted task is restored and continues execution until the operator has typed the two character directive. See Appendix M.

There is one instance at which the operator interrupt cannot acquire the teletype. This occurs when the operator interrupt has interrupted the teletype's interrupt service routine. The operator interrupt is ignored then, but depressing the switch a second time will result in acknowledgement.

9-5.3.1 Operator Interrupt Directives

Operator interrupt directives consist of two characters typed *without preceding line feed nor following carriage return*. The following directives are acceptable:

XX - This transfer control of the system to Real-Time XRAY after the current task exits. System configuration may be changed through the RTXRAY directives. The insertion of RTXRAY in the system queue at the highest priority does not affect the execution of any tasks in progress.

TT - The current task is terminated after a one second delay to allow the I/O in progress to complete, and control is transferred to the next task in the queue.

TX - The current task is terminated after a one second delay to allow the I/O in progress to complete, and control is transferred to Real-Time XRAY.

Any other character pair not of the form *X or T*, where * is any character, is ignored entirely. This feature may be used to allow the operator to control the I/O system. Teletype output operations may be suspended by depressing the manual

interrupt. Then if an invalid directive is input (e.g. *space-space*), the operation will continue. This may be used to stop the repetitive device-not-ready message output by the I/O system, while the operator readies the device.

A program in progress may be suspended by depressing the manual interrupt switch a second time after 'OPRI' is typed. This causes the system to hang up in the operator interrupt service routine waiting for the teletype. To allow the system to continue, the operator enters no-op directives.

9-5.4 Manual Intervention

If the operator interrupt is not a part of the hardware system, or if the operator interrupt has been disabled, perhaps inadvertently by some user program, or if the system is being operated through the console switches, as in hands-on single step debugging, it will be necessary to utilize the manual intervention features of the system. These consist of fixed system link points which may be set in the program counter after halting and resetting the computer. When RUN is depressed, the requested function will be accomplished. The following functions can be requested: system recovery, exit current task, relocatable program restart, and absolute load.

9-5.4.1 System Recovery

The operator may restart the system by setting the program counter to the (last cell of memory minus A2). This would be X'1FFD' for an 8K system, X'3FFD' for a 16K system. When RUN is depressed, the interrupt processing system is reset, and the operator and power fail safe interrupts are re-enabled if they are present in the system, and the system proceeds to the next task in the queue. If there are not tasks pending and if the system has an operator interrupt, the system will wait in an idle loop. The operator interrupt may then be used to inject Real-Time XRAY into the queue. If there is not operator interrupt, the system will not idle, but will instead automatically inject XRAY into the queue.

9-5.4.2 Exit Current Task

To EXIT from the current task or interrupt level

the operator may set the program counter to X'40'. This is the entry point to the system EXIT routine. If the operator has halted the system while it was servicing an interrupt, that interrupt level will be immediately restored and processing will continue. This feature can be used in the event that the system is looping in an interrupt service routine, as it might in a defective interrupt connected task. To utilize this feature, the computer should be halted *but not reset*, the program counter set to X'40' and RUN depressed. If the system is not on an interrupt level at the time the computer is halted, the current task is exited and the system proceeds exactly as described for operator recovery.

9-5.4.3 Relocatable Program Restart

Any program loaded by the relocating loader may be restarted by setting the program counter to X'7A'. This is primarily a hands-on debugging aid and is only operational if the program has not exited to the system. Once the program has exited, it cannot be reliably restarted.

9-5.4.4 Absolute Load Routine

As a convenience for manual operation, the system absolute load routine is available as an operator function. This routine loads programs in the absolute format from the BIN device (logical unit 4). It is executed by setting the program counter to X'78' and depressing RUN. Note that this is a strictly manual mode of system operation. The format of absolute programs is described in Section 8-5.1.

If successive file marks are read following a program record, the absolute load routine halts. If RUN is depressed, the absolute load routine will continue program loading. If a single file mark followed by a one word record containing an execution address is encountered, the absolute load routine will transfer to that execution address.

9-5.5 System Message

Certain messages for the operator are output on the console teletype. Typically these messages signal trouble or error conditions. This group of

messages is output by the Resident Monitor, by Real-Time XRAY, by the Queue Processor, and by certain system processors.

The following messages are cataloged here with a brief explanation of what the operator may be expected to assume when they occur. Further clarification can be found in the documentation relevant to the processor which outputs the message.

9-5.5.1 I/O Not Ready Message

The most frequent messages signal that a required I/O device is not on line. This group outputs a two-letter mnemonic identifying the device, and sounds the teletype bell twice to attract the operator's attention. The following may be encountered:

LP (Bell) (Bell)	-	Line Printer
CR (Bell) (Bell)	-	Card Reader
CP (Bell) (Bell)	-	Card Punch
Mn (Bell) (Bell)	-	Magnetic Tape Unit n

These signals are repeated at five second intervals until the requested device is placed on line.

9-5.5.2 Error Message

The message

EROR

is output by XRAY to alert the operator that some error has occurred in the processing of the previous phase of a job, and that he should examine the listed output to ascertain that the error was a programmer's error and not a system malfunction or operator error.

9-5.5.3 System Load Problem Message

The message

HELP

announces that the system is unable to correctly load a system processor from the disc. This error is catastrophic, since the system cannot proceed. It

normally indicates a hardware malfunction of the disc. Since the disc load routine retries indefinitely, the message will be output repeatedly until the processor is correctly loaded or until the operator terminates it.

9-5.5.4 SYM II Read/Write Error Messages

Messages

IOE1

and

IOE2

signal read and write errors in SYM II, respectively. If 'IOE2' occurs during Pass 1 and the scratch device is the disc, this indicates that the size of disc file 1 is insufficient to hold the source text. The file should be increased in size and the assembly re-run. 'IOE1' during pass 1 means that the source text has been misread from the PRIN device. During pass 2 the error has occurred while reading from the scratch device. This error signals a hardware malfunction.

9-5.5.5 Processor Pause Message

The message

TOG3

is a pause message output by the system leader and other system processors. It may be accompanied by error messages on the system list device; such as, CK or RD, in which instance the pause permits the operator to manually backspace one record to allow the processor to retry. The documentation for the processor which outputs the message clarifies this operation. The system loader also uses this pause between loading program modules, if Sense Switch 0 is set, and at the end-of-load, if system flag 5 is set. The operator may cause the system to continue by Setting Switch 3 to the opposite state, then returning to the present state. This procedure is called *Toggling Switch 3*.

9-5.5.6 FORTRAN Pause Message

The messages

STOP

and .

PAUS(E)

are output by the FORTRAN Run-Time package. PAUSE will require toggling Switch 3 to continue.

9-5.5.7 System Ready Messages

The message

NEXT

is output by Real-Time XRAY whenever a :NJ or :EOJ directive is encountered. It signals the beginning of a new job and is output for job logging purposes.

The message

REDY

is output by RTXRAY whenever the executive comes in to execution during the normal operating sequence and expects a directive from the system teletype.

9-5.5.8 Real-Time XRAY Ready Message

The message

XRAY

is output by RTXRAY when it makes an extraordinary appearance in the operating sequence, for instance having been queued by the operator interrupt. In this event the teletype is always the input device, even if the system is in batch mode, and directives and messages are not listed on the LIST device.

9-5.5.9 Queue Filled Message

The message

QMO

is output by the system Queue processor whenever

there is insufficient room remaining on the disc queue list area to allow the monitor queue buffer to be unloaded. The operator may ignore this error, unless further errors (resulting from monitor queue buffer overflow) occur, since the system will proceed with task execution until the disc queue is sufficiently emptied to allow the queue buffer to unload. Tasks may be executed in other than the expected priority order and the QMO message serves to diagnose any abnormalities which may occur. The proper solution to this problem is to re-generate the system, enlarging the queue space as specified to SYSGEN 1 by the :NPQ card.

9-5.5.10 Assembler in Manual Mode Message

The message

PAS?

is output by the SYMII assembler when the assembler is not running in the automatic mode. The operator may type 1, 2, or X, unless the system is in batch mode, in which instance these would be read from the card reader.

9-5.5.11 System Installation Message

The message

DATE

is output by the system installation routine whenever a listing is requested for system installation (Sense Switch 1 is set false). The operator may input a heading, for example, the date which will be copied to the program list device.

**MULTIPROGRAMMING SYSTEM
SOFTWARE OPERATION SUMMARY**

Table 9-5. Multiprogramming System Software

Description	Label	Part No.	Origin/End (HEX)	Start (HEX)	Size (DEC)	Loaded-By
Monitor-MPS	SYSMON	(1)	(2)	(2)	(2)	MPSDBST
MPS Foreground Exec	EXECF	394862	0080/0763	0080	1764	MONITOR
MPS Background Exec	BATCH	394863	0080/0A65	0080	2534	MOINTOR
MPS SYSGEN	MPSSYSG	394864	0080/0CC1	0080	3138	MONITOR
MPS Relocating Loader	MPRELD	394867	0080/06C1	0080	1610	MONITOR
MPS Debug Package	MPSDEBUG	394868	Relocatable			MPRELD
MPS Disc Bootstrap	MPSDBST	394864*	0000/002E	0000	0047	DISC-BOOT
MPS Resident Loader	MPSRESLD	394870	0080/07A9	0080	1834	MONITOR
MPS Absolute Loader	MPSAL	394869	Relocatable		0208	MPRELD
MPS Error Processor	MPSERORP	394929	0080/014B	0080	0204	MONITOR
Initial Loader-Paper Tape	INITLOAD	393260	0000/007F	0009	0128	BOOTSTRAP
Initial Loader-Cards	INITLOAD	393259	0000/007F	0009	0128	BOOTSTRAP
Disc Dump	DSKDMP	390539	0800/09B9	087D	0442	MONITOR
Absoluter, Linking-Disc	LINKABS	393255	0080/0756	0080	1751	MONITOR
Symbolic Program Editor	SYMED	394873	0080/0474	00E5	1013	MONITOR
Card Sequencer	SEQ	392920	Relocatable		0276	MPRELD
System Editor	EDITOR	391915	0080/09FC	0080	2429	MONITOR
SYM-I/PREP	SYM1	390470	05B0/0FF1	0800	2626	INITLOAD
SYM-II	SYM2	391878	0080/1201	0080	4482	MONITOR
Conver. FORTRAN Compiler	CF	394005	0080/0CEA	0080	3179	MONITOR
Conver. FORTRAN Library						
Soft. MPY/DIV	CFR	394004	0080/0CB6	0080	2871	MONITOR
Hard. MYP/DIV	CFR	394003	0080/OCA4	0080	3109	MONITOR
FORTRAN IV Compiler	FORTRAN 4	392957	0080/1DDC	01CE	7517	MONITOR
FORTRAN IV Run-Time Library						
Soft. MYP/DIV	(See	393945	Relocatable	(See		MPRELD
Hard. MPY/DIV	Appendix J)	393944	Relocatable	Appendix J)		MPRELD
Math Library						
Soft. MYP/DIV	(See	392325	Relocatable	(See		MPRELD
Hard. MYP/DIV	Appendix J)	393977	Relocatable	Appendix J)		MPRELD
<p>(1) The MPS monitor is custom prepared for each installation according to the peripheral device configuration. See software shipping manifest for monitor part number.</p> <p>(2) A typical MPS monitor for a machine with 16K memory, memory protect, operator interrupt, hardware multiply/divide, 16 levels of priority interrupt, card reader, line printer, DMA magnetic tape, disc memory, teletype multiplexer, high-speed paper tape reader and punch, requires 4020 decimal locations (304A-3FFF).</p> <p>* Documented in MPSSYSG</p>						

9-6 MULTIPROGRAMMING SYSTEM OPERATION

9-6.1 General

The Multiprogramming System (MPS) automatically sequences from task to task providing CPU time, core storage, input/output functions, interrupt control, and protection from other users. By dividing core storage into sections and allocating these sections to specific jobs, MPS can process real-time programs concurrently with background batch processing. Tasks are scheduled on a time or event basis. This section describes the MPS initialization and manual intervention procedures required to operate the MPS system.

9-6.2 Bootstrap Loading The System

The MPS system resides permanently on the disc after system generation. It may be re-loaded from the disc by depressing computer RESET, the disc controller LOAD button, then computer RUN. At this time the monitor starts the system initialization.

As an operator recovery method, bootstrapping the system should be considered only when other recovery methods have failed, since the resident and system Queue areas are lost, and all I/O device and disc vector assignments revert to their standard assignments.

9-6.2.1 MPS Initialization

After bootstrapping the system, MPS outputs the message,

Q SIZE?

The operator must input a two-digit decimal number which specifies the maximum number of entries allowed in the system Queue. There are 20 words reserved for each entry. MPS does not allow the number of Queue entries to be less than 15.

MPS, then requests the number of swap levels required by the message

SIZE?

Up to 2 decimal digits may be input specifying the number of swap levels. The number of swap levels must be 2 or greater for MPS. This number also specifies the size of the swap file reserved on the disc.

When MPS initialization is completed without any errors, the foreground executive program (EXECF) is made active in the Queue at priority level 170 and executed.

EXECF types the message

EXEC, when it is ready

to accept control directives. EXECF never inserts itself back into the Queue, since it operates in a real-time environment and does not want to tie up the system input device.

9-6.3 System Control

The operator can control the system by depressing the operator interrupt button which activates EXECF at priority 170. If another task with a higher priority than foreground 170 is currently active, EXECF will wait till the task terminates.

By depressing the operator interrupt button again, EXECF is made active at priority zero and EXECF is executed immediately.

The background execution (BATC0) is initially activated by the EXECF directive :BX.

For any EXECF intervention, the current operating environment is suspended and saved for continuance.

9-6.4 System Messages

Certain messages for the operator are output on the console teletype. Typically these messages signal trouble or error conditions. This group of messages is output by the Resident Monitor, by EXECF, by BATCH, and by certain system processors.

The following messages are cataloged here with a brief explanation of what the operator may be

expected to assume when they occur. Further clarification can be found in the documentation relevant to the processor which output the message. Appendix B is a cross reference list of these messages.

9-6.4.1 I/O Not Ready Message

The most commonly encountered messages are those which signal that a required I/O device is not on line. This group outputs a two-letter mnemonic identifying the device, and sounds the teletype bell twice to attract the operator's attention. The following may be encountered:

LP (Bell) (Bell)	-	Line Printer
CR (Bell) (Bell)	-	Card Reader
CP (Bell) (Bell)	-	Card Punch
Mn (Bell) (Bell)	-	Magnetic Tape Unit n
Mn W (Bell)	-	MT unit n is selected for write and the unit is write protected.

These signals are repeated at five second intervals until such time as the requested device is placed on line.

9-6.4.2 Error Message

The message

EROR

is output by BATCH to alert the operator that an error has occurred in the processing of the previous phase of a background job, and that he should examine the listed output to ascertain that error was a programmer's error and not a system malfunction or operator error.

9-6.4.3 System Load Problem Message

The message

HELP

announces that the system is unable to correctly load a system processor from the disc. This error is

catastrophic, since the system cannot proceed. It normally indicates a hardware malfunction of the disc. Since the disc load routine retires indefinitely, the message will be output repeatedly until the processor is correctly loaded or until the operator terminates it.

9-6.4.4 SYM-II Read/Write Errors Message

The messages

IOE1

and

IOE2

signal read and write errors in SYM II, respectively. If IOE2 occurs during Pass 1 and the scratch device is the disc, this indicates that the size of disc file 1 is insufficient to hold the source text. The file should be increased in size and the assembly re-run. IOE1 during Pass 1 means that the source text has been misread from the PRIN device. If during Pass 2 the error has occurred while reading from the scratch device, this message signals a hardware malfunction.

9-6.4.5 Processor Pause Message

The message

TOG3

is a pause message output by the system loader and by other system processors. It may be accompanied by error messages on the system list device; such as, CK or RD, in which instance the pause is to permit the operator to manually backspace one record to allow the processor to retry. Refer to the documentation for the processor which actually outputs the message for additional clarification of the operator expected. The system loader also uses this pause between loading program modules if Sense Switch 0 is set, and at the end of load, if system flag 5 is set. The operator may cause the system to continue by setting Switch 3 to the opposite state, then returning to the present state. This procedure is called *tooggling Switch 3*.

9-6.4.6 FORTRAN Pause Message

The message

STOP

and

PAUS(E)

are output by the FORTRAN-IV Run-Time package. PAUSE will require toggling Switch 3 to continue.

9-6.4.7 System Ready Messages

The message

NEXT

is output by BATCH whenever a :NJ or :EOJ directive is encountered. It signals the beginning of a new job and is output for job logging purposes.

The message

REDY

is output by BATCH whenever the background executive comes in to execution during the normal operating sequence and expects a directive from the system teletype.

The message

BATCH

is output by BATCH when it makes an extraordinary appearance in the operating sequence, for instance having been queued by the foreground executive. If the teletype is always the input device, even if the system is in the batch mode, directive messages are not listed on the LIST device.

The message

EXEC

is output by the foreground executive (EXECF)

when it is ready to accept directives. The teletype is always the input device.

9-6.4.8 Assembler in Manual Mode Message

The message

PAS?

is output by the SYM II assembler when the assembler is not running in the automatic mode. The operator may type 1, 2, or X, unless the system is in the batch mode, in which instance these would be read from the card reader. Refer to the SYM II Manual for additional details.

9-6.4.9 Initialization Error Message

The message

SEC1

is output by MPS during the initialization phase. If a read error has occurred while reading sector 1 of disc 0 (MPS information sector), MPS re-reads sector 1 continuously until sector 1 is read correctly or until it is terminated by the operator.

9-6.4.10 Processor Map Read Error Message

The message

PMAP

is output by MPS during the initialization phase. If MPS was unable to read the processor map from the disc correctly, MPS retries indefinitely until the map is read correctly or until the operator terminates it.

9-6.4.11 Swap Levels Too Large Message

The message

SFMX

is output by MPS during the initialization phase. If the number of swap levels specified is too large for the system configuration, MPS requests another number.

APPENDICES

APPENDIX A

GLOSSARY

ACRONYMS AND ABBREVIATIONS

ACR	Accumulator Register	OVF	Overflow Indicator
CP	Card Punch	PCR	Program Counter
CR	Card Reader	PEAT	Peripheral Equipment Assignment Table
DIO	Direct Input/Output	PFS	Power Fail-Safe
DMA	Direct Memory Access	PI	Priority Interrupt
EXR	Extension Register	PTIOS	Paper Tape Input/Output System
FIOT	File Input/Output Table	PTP	Paper Tape Punch
HSPTP	High-Speed Paper Tape Punch	PTR	Paper Tape Reader
HSPTR	High-Speed Paper Tape Reader	RTOS	Real-Time Operating System
IOS	Input/Output Software	SOS	Standard Operating System
IXR	Index Register	SSW	Sense Switch
LP	Line Printer	SYM I	Symbolic Assembler No. 1
LUN	Logical Unit Number	SYM II	Symbolic Assembler No. 2
MPS	Multiprogramming System	TTY	Teletype
MT	Magnetic Tape	X-RAY	Executive Program for an Operating System
MTOS	Magnetic Tape Operating System		

A

absolute address, an address which indicates the exact storage location where the referenced operand is to be found or stored in the actual machine code address numbering system.

assembler, a computer program which translates symbolic instructions into machine language.

assembly language, the source language used as input to an assembler and translated by the assembler into machine language.

B

background processing, lower priority processing, often of a batch type, that is permitted to go on in a nominally real-time system during the intervals between real-time tasks.

blocking, the combining of computer words into a physical unit of data.

breakpoint, a point in a computer program at which conditional interruption is made to permit visual check, print-outs, or other analysis.

byte, a term to indicate a measurable portion of consecutive binary digits, e.g., an 8-bit byte, a group of binary digits operated upon as a unit.

C

checkpoint, a point in time during a machine run at which processing is momentarily halted to make a record on an external storage medium of the condition of the variables of the program being executed, such as position of input and output tapes, and a copy of working storage.

concatenate, literally 'to form a chain', the placing of one register before or after another register.

D

data set, a unit of data storage and retrieval in an operating system consisting of a collection of data in some prescribed arrangement and described by control information (label) that the system has access to.

E

external subroutine, a sequence of instructions, coded once and translated as a separate program, structured to allow other programs to use its function by relinquishing control to it.

F

foreground processing, higher priority processing that takes precedence over background processing and can interrupt such processing. It often results from real-time entries or enquiries.

forward referencing, the act of referring to a symbol in a program prior to its definition (i.e., trying to assemble the instruction jump, place, where place is the location where the symbol is first defined further in the program).

G

generic instruction, an instruction or set of instructions that make no memory reference and therefore can use certain bits of the would-be address field to identify additional instructions.

APPENDIX A GLOSSARY (Cont)

I

input stream, a sequence of jobs with appropriate control statements entering the system

interleaving, the execution of more than one program at a time in a single processor by switching control back and forth between programs.

J

job, an externally specified unit of work (translation or execution of a program) for the computing system from the standpoint of operating system control.

job queue, a Line-up of jobs to be processed under operating system control.

K

key field, a portion of a record (set of characters) by which the record is identified and ordered.

L

linking, that function of the loader program which establishes connections between programs and any subroutines with which they need to communicate.

literal, a datum, usually preceded by a special character such as '=' to distinguish it from addresses, which the assembler allows the programmer to address immediately for use (e.g., add = 10 means add a value of 10; 10 is the literal).

loader, a system program for placing in memory a previously-generated machine language program and providing any necessary inter-connections with other programs and subroutines.

load time, that time at which an assembled program is placed in the computer and readied for execution.

logical record, a record from the standpoint of its content, function, and use rather than its physical attributes; i.e., one that is defined in terms of the information it contains.

M

machine language, a language designed for interpretation and use by a machine (computer) without translation, a language consisting of numeric machine instructions.

macro instruction, an instruction which causes the generation of a set of machine instructions which are inserted into machine language program output from assembler.

mass storage, usually a random access device of very large capacity in comparison to magnetic core memories (e.g., magnetic disc).

module, the input to, or output from, a single execution of an assembler, a compiler, or a loader; hence, a program unit that is discrete and identifiable with respect to compiling, combining with other units (linking), and loading.

O

on-line, a system in which input data is processed as it is received and output data is transmitted to the point where it is needed on an immediate or real-time basis.

operating system, an integrated collection of service routines for supervising the execution of programs by a computer.

optimum code, a set of machine language instructions which is particularly efficient with regard to a particular aspect; e.g., minimum time to execute or minimum or efficient use of storage space.

overlapping, the act of reading information into or writing it from a portion of memory while computing operations continue in other portions; input/output processor and central processor share memory.

overlay, a load module which is to be placed in main storage locations occupied by another module; several modules may occupy the same storage area at different times.

P

page, a consecutive portion of core memory that provides for easy memory reference when the address portion of a computer word is not long enough to directly address all of memory.

PL/1, programming language one, also called new programming language; a higher level language designed for use with some third-generation equipment (IBM S360).

procedure-oriented language, a programming language which describes the process of solving a problem in machine independent terms.

pseudo-operations, those instructions required for controlling the functions of an assembler.

pseudo-instruction, instructions used to give data definition information to the operating system and to control the system's action.

R

recursive, pertaining to a process which is repetitive within itself, the result of each repetition is usually dependent upon the result of the previous repetition.

re-entrant routine, a routine that may be entered a successive number of times without having to complete each entry. Normally used for an interrupt environment.

relocatable code, code generated by assemblers or other translocation programs that provides for address assignment at load time and thus allows the loader program to place the code at any convenient locations in core.

real-time basis, a processing mode in which the processing of information or data occurs in a sufficiently rapid manner so that the results of the processing are available in time to influence the process being controlled.

APPENDIX A

GLOSSARY (Cont)

S

- snapshot dump**, a dynamic printout during execution, at breakpoints and checkpoints, of selected areas in storage.
- source language**, the original form in which a program is prepared prior to processing by a language translation program such as an assembler or a compiler.
- symbolic address**, a symbolic name used in the location field of a symbolic instruction to identify a data element or memory location without prior knowledge of its actual absolute location, the symbolic reference is easier to read and write than an absolute reference.
- symbolic instructions**, an instruction which is the basic component on an assembly language (input to assembler) and is directly translatable into machine language.
- symbolic name**, a string of one or more alphabetic or numeric characters; the first portion is usually alphabetic.
- symbol**, a substitute or representation of characteristics, relationships, or transformations of ideas or things.
- syntax**, the rules governing the ordering of symbols in a language and the meanings associated with these symbols in that given context.
- system macro library**, a set of macro prototypes available to the assembler so that the programmer may call them without having to define them in his program.

T

- time sharing**, a special kind of multiprogramming where emphasis is no longer on achieving maximum throughput, but rather on providing service to many remote users, by using such a job scheduling method as time slicing, a time-sharing system gives each user the impression that the computer is under his exclusive control.
- time-slicing**, a method of job scheduling in a multiprogramming system. This refers to the allocation of fixed amounts of computing time among users on a round-robin basis. Interrupts are generated by a fixed interval timer causing control to pass to the next waiting service request.
- translate, load, and go capability**, the ability of a system, by use of intermediate storage, to allow the translation, linkage, loading, and execution of a program as one continuous job without break.

U

- unit record**, a record from the standpoint of the manner or form in which it is stored, retrieved, and moved; i.e., one that is defined in terms of physical qualities, e.g., a card.

APPENDIX B

APPLICABLE PUBLICATIONS INDEX

706 Products

Part No. or Dwg. No.	Description
SP-328	706 User's Manual
SP-325	706 Brochure
PL-321	706 Price List
392325-003 (SP-301,302,303)	Math Library
392325-002 (SP-301,302,303)	Math Library ¹
392316-003	X-RAY EXEC and Systems Program – Basic
392316-002	X-RAY EXEC and Systems Program – Basic ¹
392317-003	X-RAY EXEC and Systems Program – Standard
392317-002	X-RAY EXEC and Systems Program – Standard ¹
392318-003	X-RAY EXEC and Systems Program – Extended
392318-002	X-RAY EXEC and Systems Program – Extended ¹
392319-003	Utility Programs – Basic
392320-003	Utility Programs – Standard
392321-003	Utility Programs – Extended
392931-003 (SP-307)	FORTTRAN IV Library – Volume I
392932-003 (SP-308)	FORTTRAN IV Library – Volume II
390470-002	SYM I/PREP Reference Manual ¹
391919-002 (SP-274)	SYM II Reference Manual ¹
392284-002 (SP-304)	Real-Time FORTRAN IV ¹
392936-002	Conversational FORTRAN ¹
392243-002	Real-Time Operating System ¹
RC-2060	Central Processor – Technical & Operations Manual
RC-2024	Disc Memory – Technical & Operations Manual
RC-2022	Paper Tape Reader & Punch – Technical and Operations Manual
RC-2026	Line Printer Controller – Technical and Operations Manual
RC-1057	Teletypewriter Multiplexer Controller – Technical & Operations Manual
RC-2033	Buffered Digital Output Channel – Technical & Operations Manual
RC-2034	Buffered Digital Input Channel – Technical & Operations Manual
RC-2025	Card Reader and Punch – Technical & Operations Manual
RC-2031	Multiverter Interface – Technical & Operations Manual

¹Does not include program listing

APPENDIX B

APPLICABLE PUBLICATIONS INDEX (Cont)

RC-2032	Miniverter Interface – Technical & Operations Manual
RC-2035	Time-of-Day Clock – Technical & Operations Manual

Other Products

392934-002 (SP-248C)	703 Reference and Interface Manual
SP-311	Semiautomatic Wire Wrap Service
SP-327	Applications Interface Device (AID)
SP-330	Array Transform Processor (ATP)
SP-230C	M-Series IC Digital Modules
PL-323	703 Price List
SP-205C	Multiverter Brochure
SP-288A	Analog Instruments
SP-283	Differential Miniverter
SP-327A	AID (Application Interface Device)
SP-2048A	300 Core Memory System

¹Does not include program listing

2ⁿ and 16^m

m n

APPENDIX C
POWERS OF 2 AND 16

2⁻ⁿ and 16^{-m}

1	0	0	1.0	
2		1	0.5	
4		2	0.25	
8		3	0.125	
16	1	4	0.062 5	
32		5	0.031 25	
64		6	0.015 625	
128		7	0.007 812 5	
256	2	8	0.003 906 25	
512		9	0.001 953 125	
1 024		10	0.000 976 562 5	
2 048		11	0.000 488 281 25	
4 096	3	12	0.000 244 140 625	
8 192		13	0.000 122 070 312 5	
16 384		14	0.000 061 035 156 25	
32 768		15	0.000 030 517 578 125	
65 536	4	16	0.000 015 258 789 062 5	
131 072		17	0.000 007 629 394 531 25	
262 144		18	0.000 003 814 697 265 625	
524 288		19	0.000 001 907 348 632 812 5	
1 048 576	5	20	0.000 000 953 674 316 406 25	
2 097 152		21	0.000 000 476 837 158 203 125	
4 194 304		22	0.000 000 238 418 579 101 562 5	
8 388 608		23	0.000 000 119 209 289 550 781 25	
16 777 216	6	24	0.000 000 059 604 644 775 390 625	
33 554 432		25	0.000 000 029 802 322 387 695 312 5	
67 108 864		26	0.000 000 014 901 161 193 847 656 25	
134 217 728		27	0.000 000 007 450 580 596 923 828 125	
268 435 456	7	28	0.000 000 003 725 290 298 461 914 062 5	
536 870 912		29	0.000 000 001 862 645 149 230 957 031 25	
1 073 741 824		30	0.000 000 000 931 322 574 615 478 515 625	
2 147 483 648		31	0.000 000 000 465 661 287 307 739 257 812 5	
4 294 967 296	8	32	0.000 000 000 232 830 643 653 869 628 906 25	
8 589 934 592		33	0.000 000 000 116 415 321 826 934 814 453 125	
17 179 869 184		34	0.000 000 000 058 207 660 913 467 407 226 562 5	
34 359 738 368		35	0.000 000 000 029 103 830 456 733 703 613 281 25	
68 719 476 736	9	36	0.000 000 000 014 551 915 228 366 851 806 640 625	
137 438 953 472		37	0.000 000 000 007 275 957 614 183 425 903 320 312 5	
274 877 906 944		38	0.000 000 000 003 637 978 807 091 712 951 660 156 25	
549 755 813 888		39	0.000 000 000 001 818 989 403 545 856 475 830 078 125	
1 099 511 627 776	10	40	0.000 000 000 000 909 494 701 772 928 237 915 039 062 5	
2 199 023 255 552		41	0.000 000 000 000 454 747 350 886 464 118 957 519 531 25	
4 398 046 511 104		42	0.000 000 000 000 227 373 675 443 232 059 478 759 765 625	
8 796 093 022 208		43	0.000 000 000 000 113 686 837 721 616 029 739 379 882 812 5	
17 592 186 044 416	11	44	0.000 000 000 000 056 843 418 860 808 014 869 689 941 406 25	
35 184 372 088 832		45	0.000 000 000 000 028 421 709 430 404 007 434 844 970 703 125	
70 368 744 177 664		46	0.000 000 000 000 014 210 854 715 202 003 717 422 485 351 562 5	
140 737 488 355 328		47	0.000 000 000 000 007 105 427 357 601 001 858 711 242 675 781 25	
281 474 976 710 656	12	48	0.000 000 000 000 003 552 713 678 800 500 929 355 621 337 890 625	

APPENDIX D
MISCELLANEOUS REFERENCE DATA

COMMON CONSTANTS		
	DECIMAL	HEX
e	2.718 281 828 5	2.B7E1 5163
$1/e$.367 879 441 2	0.5E2D 58D9
$\log_{10}e$.434 294 481 9	0.6F2D EC55
$\log_e 10$	2.302 585 093 0	2.4D76 3777
$\log_e 2$.693 147 180 6	0.B172 17F8
$\log_{10}\pi$.497 149 872 7	0.7F45 36CC
$\log_e\pi$	1.144 729 885 8	1.250D 048 E
π	3.141 592 653 6	3.243F 6A89
$1/\pi$.318 309 886 2	0.517C C1B7
$\sqrt{\pi}$	1.7 724 538 509	1.C5BF 891C
γ	0.5 772 156 649	0.93C4 67E4
$\log_e \gamma$	-0.5 495 393 129	-0.8CAE 9BC3

FACTORIALS		
	DECIMAL	HEX
1!	1	001
2!	2	002
3!	6	006
4!	24	018
5!	120	078
6!	720	2D0
7!	5 040	1 3B0
8!	40 320	9 D80
9!	362 880	58 980
10!	3 628 800	375 EBA
11!	39 916 800	2 611 500
12!	479 001 600	1B 8CF C00
13!	6 227 020 800	173 291 A20
14!	87 178 291 200	1 44C 3B2 800

APPENDIX . D

RECIPROCAL S		
n	1/n (DECIMAL)	1/n (HEXADECIMAL)
2	+.500000000	+.800000000
3	+.333333333	+.555555555
4	+.250000000	+.400000000
5	+.200000000	+.333333333
6	+.166666667	+.2AAAAAAAA
7	+.1428571428	+.249249249
8	+.125000000	+.200000000
9	+.111111111	+.1C71C71C7
10	+.100000000	+.199999999
11	+.0909090909	+.1745D1745
12	+.083333333	+.155555555
13	+.0769230769	+.13B13B13B
14	+.0714285714	+.124924924
15	+.066666667	+.111111111
16	+.062500000	+.100000000
17	+.0588235294	+.0F0F0F0F0
18	+.055555555	+.0E38E38E3
19	+.0526315789	+.0D79435E5
20	+.050000000	+.0CCCCCCCC
21	+.0476190476	+.0C30C30C3
22	+.0454545454	+.0BA2E8BA2
23	+.0434782609	+.0B21642C8
24	+.041666667	+.0AAAAAAAA
25	+.040000000	+.0A3D70A3D
26	+.0384615384	+.09D89D89D
27	+.0370370370	+.097B425ED
28	+.0357142857	+.092492492
29	+.0344827586	+.08D3DCB08
30	+.033333333	+.088888888

APPENDIX D

SQUARE ROOTS		
	DECIMAL	HEX
$\sqrt{2}$	1.414 213 562 4	1.6A0 9E6 681
$\sqrt{3}$	1.732 050 807 6	1.BB6 7AE 85A
$\sqrt{5}$	2.236 067 977 5	2.3C6 EF3 730
$\sqrt{6}$	2.449 489 742 8	2.731 1C2 813
$\sqrt{7}$	2.645 751 311 1	2.A54 FF5 3A8
$\sqrt{8}$	2.828 427 124 8	2.D41 3CC D02
$\sqrt{10}$	3.162 277 660 2	3.298 B07 5B7

MISCELLANEOUS CONSTANTS		
	DECIMAL	HEX
1° = 1/360 of a circle	0.002 777 777 8	0.00B60B
1° = .017 453 292 5 radians	0.017 453 292 5	0.0477D1
Euler's Constant $\gamma =$	0.577 215 664 9	0.93C467

APPENDIX E

HEXADECIMAL - TO - DECIMAL CONVERSION

GENERAL

This appendix contains Hexadecimal-To-Decimal conversion tables for positive numbers, negative numbers, and fractions. Conversion of binary numbers to decimal or hexadecimal and conversion of hexadecimal numbers to decimal are also discussed.

BINARY NUMBER CONVERSION TO DECIMAL

Prior to explaining the specifics of the number systems, observe the following basics of number systems.

Any positional number system can be defined by its base (radix). The base is the number of unique symbols that can occupy a position in the number system. As an example, the decimal system has a base of 10 because symbols (numbers in this case) 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 can occupy a position in the number system. In the binary system, there are two symbols, 0 and 1, that can occupy a position. Therefore, a binary system is base 2. In the hexadecimal system, symbols 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F can occupy a position in the number system. Therefore, it is a base 16 system. In converting from binary or hexadecimal to decimal, the symbol represents a coefficient, and, its position in the number, the exponent of the base -- examples of this are given below.

DECIMAL NUMBER SYSTEM

Given a decimal number of 159, state it as a function of its base.

The number 159 can be restated as 100 plus 50 plus 9 or:

$$1(10^2) + 5(10^1) + 9(10^0) = 159$$

BINARY NUMBER SYSTEM

Given the binary number 1011, convert it to decimal.

$$1(2^3) + 0(2^2) + 1(2^1) + 1(2^0) = 11$$

APPENDIX E

Given the binary numbers 0 through 15, convert them to hexadecimal numbers.

Binary-to-Hexadecimal Conversion Table		
Binary	Hexadecimal	Decimal
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	A	10
1011	B	11
1100	C	12
1101	D	13
1110	E	14
1111	F	15

HEXADECIMAL NUMBER SYSTEM

Given the hexadecimal number 4F2, convert it to decimal. (For the conversion of F refer to the table under the Binary Number System).

$$4 (16^2) + F (16^1) + 2 (16^0) = 4 (256) + 15 (16) + 2 (1) = 1266$$

Note that the exponent for the base was determined (in all cases) by counting from the right, the symbols position in the number.

APPENDIX E

INTEGER TABLE EXTENSION

For numbers outside the range of the integer table add the following values to those of the Hexadecimal to Positive Decimal Integer table.

<u>Hexadecimal</u>	<u>Decimal</u>
1000	4096
2000	8192
3000	12288
4000	16384
5000	20480
6000	24576
7000	28672
8000	32768
9000	36864
A000	40960
B000	45056
C000	49152
D000	53248
E000	57344
F000	61440

APPENDIX E

HEXADECIMAL-TO-POSITIVE INTEGER CONVERSION TABLE

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
01	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
02	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
03	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
04	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
05	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
06	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
07	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
08	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
09	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
0A	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
0B	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
0C	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
0D	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
0E	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
0F	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255
10	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271
11	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287
12	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303
13	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319
14	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335
15	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351
16	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367
17	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383
18	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399
19	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415
1A	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431
1B	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447
1C	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463
1D	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479
1E	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495
1F	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511
20	512	513	514	515	516	517	518	519	520	521	522	523	524	525	526	527
21	528	529	530	531	532	533	534	535	536	537	538	539	540	541	542	543
22	544	545	546	547	548	549	550	551	552	553	554	555	556	557	558	559
23	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575
24	576	577	578	579	580	581	582	583	584	585	586	587	588	589	590	591
25	592	593	594	595	596	597	598	599	600	601	602	603	604	605	606	607
26	608	609	610	611	612	613	614	615	616	617	618	619	620	621	622	623
27	624	625	626	627	628	629	630	631	632	633	634	635	636	637	638	639
28	640	641	642	643	644	645	646	647	648	649	650	651	652	653	654	655
29	656	657	658	659	660	661	662	663	664	665	666	667	668	669	670	671
2A	672	673	674	675	676	677	678	679	680	681	682	683	684	685	686	687
2B	688	689	690	691	692	693	694	695	696	697	698	699	700	701	702	703
2C	704	705	706	707	708	709	710	711	712	713	714	715	716	717	718	719
2D	720	721	722	723	724	725	726	727	728	729	730	731	732	733	734	735
2E	736	737	738	739	740	741	742	743	744	745	746	747	748	749	750	751
2F	752	753	754	755	756	757	758	759	760	761	762	763	764	765	766	767
30	768	769	770	771	772	773	774	775	776	777	778	779	780	781	782	783
31	784	785	786	787	788	789	790	791	792	793	794	795	796	797	798	799
32	800	801	802	803	804	805	806	807	808	809	810	811	812	813	814	815
33	816	817	818	819	820	821	822	823	824	825	826	827	828	829	830	831
34	832	833	834	835	836	837	838	839	840	841	842	843	844	845	846	847
35	848	849	850	851	852	853	854	855	856	857	858	859	860	861	862	863
36	864	865	866	867	868	869	870	871	872	873	874	875	876	877	878	879
37	880	881	882	883	884	885	886	887	888	889	890	891	892	893	894	895
38	896	897	898	899	900	901	902	903	904	905	906	907	908	909	910	911
39	912	913	914	915	916	917	918	919	920	921	922	923	924	925	926	927
3A	928	929	930	931	932	933	934	935	936	937	938	939	940	941	942	943
3B	944	945	946	947	948	949	950	951	952	953	954	955	956	957	958	959
3C	960	961	962	963	964	965	966	967	968	969	970	971	972	973	974	975
3D	976	977	978	979	980	981	982	983	984	985	986	987	988	989	990	991
3E	992	993	994	995	996	997	998	999	1000	1001	1002	1003	1004	1005	1006	1007
3F	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1020	1021	1022	1023

APPENDIX E

HEXADECIMAL-TO-POSITIVE INTEGER CONVERSION TABLE (Cont)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
40	1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039
41	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1050	1051	1052	1053	1054	1055
42	1056	1057	1058	1059	1060	1061	1062	1063	1064	1065	1066	1067	1068	1069	1070	1071
43	1072	1073	1074	1075	1076	1077	1078	1079	1080	1081	1082	1083	1084	1085	1086	1087
44	1088	1089	1090	1091	1092	1093	1094	1095	1096	1097	1098	1099	1100	1101	1102	1103
45	1104	1105	1106	1107	1108	1109	1110	1111	1112	1113	1114	1115	1116	1117	1118	1119
46	1120	1121	1122	1123	1124	1125	1126	1127	1128	1129	1130	1131	1132	1133	1134	1135
47	1136	1137	1138	1139	1140	1141	1142	1143	1144	1145	1146	1147	1148	1149	1150	1151
48	1152	1153	1154	1155	1156	1157	1158	1159	1160	1161	1162	1163	1164	1165	1166	1167
49	1168	1169	1170	1171	1172	1173	1174	1175	1176	1177	1178	1179	1180	1181	1182	1183
4A	1184	1185	1186	1187	1188	1189	1190	1191	1192	1193	1194	1195	1196	1197	1198	1199
4B	1200	1201	1202	1203	1204	1205	1206	1207	1208	1209	1210	1211	1212	1213	1214	1215
4C	1216	1217	1218	1219	1220	1221	1222	1223	1224	1225	1226	1227	1228	1229	1230	1231
4D	1232	1233	1234	1235	1236	1237	1238	1239	1240	1241	1242	1243	1244	1245	1246	1247
4E	1248	1249	1250	1251	1252	1253	1254	1255	1256	1257	1258	1259	1260	1261	1262	1263
4F	1264	1265	1266	1267	1268	1269	1270	1271	1272	1273	1274	1275	1276	1277	1278	1279
50	1280	1281	1282	1283	1284	1285	1286	1287	1288	1289	1290	1291	1292	1293	1294	1295
51	1296	1297	1298	1299	1300	1301	1302	1303	1304	1305	1306	1307	1308	1309	1310	1311
52	1312	1313	1314	1315	1316	1317	1318	1319	1320	1321	1322	1323	1324	1325	1326	1327
53	1328	1329	1330	1331	1332	1333	1334	1335	1336	1337	1338	1339	1340	1341	1342	1343
54	1344	1345	1346	1347	1348	1349	1350	1351	1352	1353	1354	1355	1356	1357	1358	1359
55	1360	1361	1362	1363	1364	1365	1366	1367	1368	1369	1370	1371	1372	1373	1374	1375
56	1376	1377	1378	1379	1380	1381	1382	1383	1384	1385	1386	1387	1388	1389	1390	1391
57	1392	1393	1394	1395	1396	1397	1398	1399	1400	1401	1402	1403	1404	1405	1406	1407
58	1408	1409	1410	1411	1412	1413	1414	1415	1416	1417	1418	1419	1420	1421	1422	1423
59	1424	1425	1426	1427	1428	1429	1430	1431	1432	1433	1434	1435	1436	1437	1438	1439
5A	1440	1441	1442	1443	1444	1445	1446	1447	1448	1449	1450	1451	1452	1453	1454	1455
5B	1456	1457	1458	1459	1460	1461	1462	1463	1464	1465	1466	1467	1468	1469	1470	1471
5C	1472	1473	1474	1475	1476	1477	1478	1479	1480	1481	1482	1483	1484	1485	1486	1487
5D	1488	1489	1490	1491	1492	1493	1494	1495	1496	1497	1498	1499	1500	1501	1502	1503
5E	1504	1505	1506	1507	1508	1509	1510	1511	1512	1513	1514	1515	1516	1517	1518	1519
5F	1520	1521	1522	1523	1524	1525	1526	1527	1528	1529	1530	1531	1532	1533	1534	1535
60	1536	1537	1538	1539	1540	1541	1542	1543	1544	1545	1546	1547	1548	1549	1550	1551
61	1552	1553	1554	1555	1556	1557	1558	1559	1560	1561	1562	1563	1564	1565	1566	1567
62	1568	1569	1570	1571	1572	1573	1574	1575	1576	1577	1578	1579	1580	1581	1582	1583
63	1584	1585	1586	1587	1588	1589	1590	1591	1592	1593	1594	1595	1596	1597	1598	1599
64	1600	1601	1602	1603	1604	1605	1606	1607	1608	1609	1610	1611	1612	1613	1614	1615
65	1616	1617	1618	1619	1620	1621	1622	1623	1624	1625	1626	1627	1628	1629	1630	1631
66	1632	1633	1634	1635	1636	1637	1638	1639	1640	1641	1642	1643	1644	1645	1646	1647
67	1648	1649	1650	1651	1652	1653	1654	1655	1656	1657	1658	1659	1660	1661	1662	1663
68	1664	1665	1666	1667	1668	1669	1670	1671	1672	1673	1674	1675	1676	1677	1678	1679
69	1680	1681	1682	1683	1684	1685	1686	1687	1688	1689	1690	1691	1692	1693	1694	1695
6A	1696	1697	1698	1699	1700	1701	1702	1703	1704	1705	1706	1707	1708	1709	1710	1711
6B	1712	1713	1714	1715	1716	1717	1718	1719	1720	1721	1722	1723	1724	1725	1726	1727
6C	1728	1729	1730	1731	1732	1733	1734	1735	1736	1737	1738	1739	1740	1741	1742	1743
6D	1744	1745	1746	1747	1748	1749	1750	1751	1752	1753	1754	1755	1756	1757	1758	1759
6E	1760	1761	1762	1763	1764	1765	1766	1767	1768	1769	1770	1771	1772	1773	1774	1775
6F	1776	1777	1778	1779	1780	1781	1782	1783	1784	1785	1786	1787	1788	1789	1790	1791
70	1792	1793	1794	1795	1796	1797	1798	1799	1800	1801	1802	1803	1804	1805	1806	1807
71	1808	1809	1810	1811	1812	1813	1814	1815	1816	1817	1818	1819	1820	1821	1822	1823
72	1824	1825	1826	1827	1828	1829	1830	1831	1832	1833	1834	1835	1836	1837	1838	1839
73	1840	1841	1842	1843	1844	1845	1846	1847	1848	1849	1850	1851	1852	1853	1854	1855
74	1856	1857	1858	1859	1860	1861	1862	1863	1864	1865	1866	1867	1868	1869	1870	1871
75	1872	1873	1874	1875	1876	1877	1878	1879	1880	1881	1882	1883	1884	1885	1886	1887
76	1888	1889	1890	1891	1892	1893	1894	1895	1896	1897	1898	1899	1900	1901	1902	1903
77	1904	1905	1906	1907	1908	1909	1910	1911	1912	1913	1914	1915	1916	1917	1918	1919
78	1920	1921	1922	1923	1924	1925	1926	1927	1928	1929	1930	1931	1932	1933	1934	1935
79	1936	1937	1938	1939	1940	1941	1942	1943	1944	1945	1946	1947	1948	1949	1950	1951
7A	1952	1953	1954	1955	1956	1957	1958	1959	1960	1961	1962	1963	1964	1965	1966	1967
7B	1968	1969	1970	1971	1972	1973	1974	1975	1976	1977	1978	1979	1980	1981	1982	1983
7C	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999
7D	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
7E	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031
7F	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047

APPENDIX E

HEXADECIMAL-TO-POSITIVE INTEGER CONVERSION TABLE (Cont)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
80	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063
81	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079
82	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095
83	2096	2097	2098	2099	2100	2101	2102	2103	2104	2105	2106	2107	2108	2109	2110	2111
84	2112	2113	2114	2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127
85	2128	2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143
86	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156	2157	2158	2159
87	2160	2161	2162	2163	2164	2165	2166	2167	2168	2169	2170	2171	2172	2173	2174	2175
88	2176	2177	2178	2179	2180	2181	2182	2183	2184	2185	2186	2187	2188	2189	2190	2191
89	2192	2193	2194	2195	2196	2197	2198	2199	2200	2201	2202	2203	2204	2205	2206	2207
8A	2208	2209	2210	2211	2212	2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223
8B	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239
8C	2240	2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254	2255
8D	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271
8E	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2287
8F	2288	2289	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303
90	2304	2305	2306	2307	2308	2309	2310	2311	2312	2313	2314	2315	2316	2317	2318	2319
91	2320	2321	2322	2323	2324	2325	2326	2327	2328	2329	2330	2331	2332	2333	2334	2335
92	2336	2337	2338	2339	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351
93	2352	2353	2354	2355	2356	2357	2358	2359	2360	2361	2362	2363	2364	2365	2366	2367
94	2368	2369	2370	2371	2372	2373	2374	2375	2376	2377	2378	2379	2380	2381	2382	2383
95	2384	2385	2386	2387	2388	2389	2390	2391	2392	2393	2394	2395	2396	2397	2398	2399
96	2400	2401	2402	2403	2404	2405	2406	2407	2408	2409	2410	2411	2412	2413	2414	2415
97	2416	2417	2418	2419	2420	2421	2422	2423	2424	2425	2426	2427	2428	2429	2430	2431
98	2432	2433	2434	2435	2436	2437	2438	2439	2440	2441	2442	2443	2444	2445	2446	2447
99	2448	2449	2450	2451	2452	2453	2454	2455	2456	2457	2458	2459	2460	2461	2462	2463
9A	2464	2465	2466	2467	2468	2469	2470	2471	2472	2473	2474	2475	2476	2477	2478	2479
9B	2480	2481	2482	2483	2484	2485	2486	2487	2488	2489	2490	2491	2492	2493	2494	2495
9C	2496	2497	2498	2499	2500	2501	2502	2503	2504	2505	2506	2507	2508	2509	2510	2511
9D	2512	2513	2514	2515	2516	2517	2518	2519	2520	2521	2522	2523	2524	2525	2526	2527
9E	2528	2529	2530	2531	2532	2533	2534	2535	2536	2537	2538	2539	2540	2541	2542	2543
9F	2544	2545	2546	2547	2548	2549	2550	2551	2552	2553	2554	2555	2556	2557	2558	2559
A0	2560	2561	2562	2563	2564	2565	2566	2567	2568	2569	2570	2571	2572	2573	2574	2575
A1	2576	2577	2578	2579	2580	2581	2582	2583	2584	2585	2586	2587	2588	2589	2590	2591
A2	2592	2593	2594	2595	2596	2597	2598	2599	2600	2601	2602	2603	2604	2605	2606	2607
A3	2608	2609	2610	2611	2612	2613	2614	2615	2616	2617	2618	2619	2620	2621	2622	2623
A4	2624	2625	2626	2627	2628	2629	2630	2631	2632	2633	2634	2635	2636	2637	2638	2639
A5	2640	2641	2642	2643	2644	2645	2646	2647	2648	2649	2650	2651	2652	2653	2654	2655
A6	2656	2657	2658	2659	2660	2661	2662	2663	2664	2665	2666	2667	2668	2669	2670	2671
A7	2672	2673	2674	2675	2676	2677	2678	2679	2680	2681	2682	2683	2684	2685	2686	2687
A8	2688	2689	2690	2691	2692	2693	2694	2695	2696	2697	2698	2699	2700	2701	2702	2703
A9	2704	2705	2706	2707	2708	2709	2710	2711	2712	2713	2714	2715	2716	2717	2718	2719
AA	2720	2721	2722	2723	2724	2725	2726	2727	2728	2729	2730	2731	2732	2733	2734	2735
AB	2736	2737	2738	2739	2740	2741	2742	2743	2744	2745	2746	2747	2748	2749	2750	2751
AC	2752	2753	2754	2755	2756	2757	2758	2759	2760	2761	2762	2763	2764	2765	2766	2767
AD	2768	2769	2770	2771	2772	2773	2774	2775	2776	2777	2778	2779	2780	2781	2782	2783
AE	2784	2785	2786	2787	2788	2789	2790	2791	2792	2793	2794	2795	2796	2797	2798	2799
AF	2800	2801	2802	2803	2804	2805	2806	2807	2808	2809	2810	2811	2812	2813	2814	2815
B0	2816	2817	2818	2819	2820	2821	2822	2823	2824	2825	2826	2827	2828	2829	2830	2831
B1	2832	2833	2834	2835	2836	2837	2838	2839	2840	2841	2842	2843	2844	2845	2846	2847
B2	2848	2849	2850	2851	2852	2853	2854	2855	2856	2857	2858	2859	2860	2861	2862	2863
B3	2864	2865	2866	2867	2868	2869	2870	2871	2872	2873	2874	2875	2876	2877	2878	2879
B4	2880	2881	2882	2883	2884	2885	2886	2887	2888	2889	2890	2891	2892	2893	2894	2895
B5	2896	2897	2898	2899	2900	2901	2902	2903	2904	2905	2906	2907	2908	2909	2910	2911
B6	2912	2913	2914	2915	2916	2917	2918	2919	2920	2921	2922	2923	2924	2925	2926	2927
B7	2928	2929	2930	2931	2932	2933	2934	2935	2936	2937	2938	2939	2940	2941	2942	2943
B8	2944	2945	2946	2947	2948	2949	2950	2951	2952	2953	2954	2955	2956	2957	2958	2959
B9	2960	2961	2962	2963	2964	2965	2966	2967	2968	2969	2970	2971	2972	2973	2974	2975
BA	2976	2977	2978	2979	2980	2981	2982	2983	2984	2985	2986	2987	2988	2989	2990	2991
BB	2992	2993	2994	2995	2996	2997	2998	2999	3000	3001	3002	3003	3004	3005	3006	3007
BC	3008	3009	3010	3011	3012	3013	3014	3015	3016	3017	3018	3019	3020	3021	3022	3023
BD	3024	3025	3026	3027	3028	3029	3030	3031	3032	3033	3034	3035	3036	3037	3038	3039
BE	3040	3041	3042	3043	3044	3045	3046	3047	3048	3049	3050	3051	3052	3053	3054	3055
BF	3056	3057	3058	3059	3060	3061	3062	3063	3064	3065	3066	3067	3068	3069	3070	3071

APPENDIX E

HEXADECIMAL-TO-POSITIVE INTEGER CONVERSION TABLE (Cont)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
C0	3072	3073	3074	3075	3076	3077	3078	3079	3080	3081	3082	3083	3084	3085	3086	3087
C1	3088	3089	3090	3091	3092	3093	3094	3095	3096	3097	3098	3099	3100	3101	3102	3103
C2	3104	3105	3106	3107	3108	3109	3110	3111	3112	3113	3114	3115	3116	3117	3118	3119
C3	3120	3121	3122	3123	3124	3125	3126	3127	3128	3129	3130	3131	3132	3133	3134	3135
C4	3136	3137	3138	3139	3140	3141	3142	3143	3144	3145	3146	3147	3148	3149	3150	3151
C5	3152	3153	3154	3155	3156	3157	3158	3159	3160	3161	3162	3163	3164	3165	3166	3167
C6	3168	3169	3170	3171	3172	3173	3174	3175	3176	3177	3178	3179	3180	3181	3182	3183
C7	3184	3185	3186	3187	3188	3189	3190	3191	3192	3193	3194	3195	3196	3197	3198	3199
C8	3200	3201	3202	3203	3204	3205	3206	3207	3208	3209	3210	3211	3212	3213	3214	3215
C9	3216	3217	3218	3219	3220	3221	3222	3223	3224	3225	3226	3227	3228	3229	3230	3231
CA	3232	3233	3234	3235	3236	3237	3238	3239	3240	3241	3242	3243	3244	3245	3246	3247
CB	3248	3249	3250	3251	3252	3253	3254	3255	3256	3257	3258	3259	3260	3261	3262	3263
CC	3264	3265	3266	3267	3268	3269	3270	3271	3272	3273	3274	3275	3276	3277	3278	3279
CD	3280	3281	3282	3283	3284	3285	3286	3287	3288	3289	3290	3291	3292	3293	3294	3295
CE	3296	3297	3298	3299	3300	3301	3302	3303	3304	3305	3306	3307	3308	3309	3310	3311
CF	3312	3313	3314	3315	3316	3317	3318	3319	3320	3321	3322	3323	3324	3325	3326	3327
D0	3328	3329	3330	3331	3332	3333	3334	3335	3336	3337	3338	3339	3340	3341	3342	3343
D1	3344	3345	3346	3347	3348	3349	3350	3351	3352	3353	3354	3355	3356	3357	3358	3359
D2	3360	3361	3362	3363	3364	3365	3366	3367	3368	3369	3370	3371	3372	3373	3374	3375
D3	3376	3377	3378	3379	3380	3381	3382	3383	3384	3385	3386	3387	3388	3389	3390	3391
D4	3392	3393	3394	3395	3396	3397	3398	3399	3400	3401	3402	3403	3404	3405	3406	3407
D5	3408	3409	3410	3411	3412	3413	3414	3415	3416	3417	3418	3419	3420	3421	3422	3423
D6	3424	3425	3426	3427	3428	3429	3430	3431	3432	3433	3434	3435	3436	3437	3438	3439
D7	3440	3441	3442	3443	3444	3445	3446	3447	3448	3449	3450	3451	3452	3453	3454	3455
D8	3456	3457	3458	3459	3460	3461	3462	3463	3464	3465	3466	3467	3468	3469	3470	3471
D9	3472	3473	3474	3475	3476	3477	3478	3479	3480	3481	3482	3483	3484	3485	3486	3487
DA	3488	3489	3490	3491	3492	3493	3494	3495	3496	3497	3498	3499	3500	3501	3502	3503
DB	3504	3505	3506	3507	3508	3509	3510	3511	3512	3513	3514	3515	3516	3517	3518	3519
DC	3520	3521	3522	3523	3524	3525	3526	3527	3528	3529	3530	3531	3532	3533	3534	3535
DD	3536	3537	3538	3539	3540	3541	3542	3543	3544	3545	3546	3547	3548	3549	3550	3551
DE	3552	3553	3554	3555	3556	3557	3558	3559	3560	3561	3562	3563	3564	3565	3566	3567
DF	3568	3569	3570	3571	3572	3573	3574	3575	3576	3577	3578	3579	3580	3581	3582	3583
E0	3584	3585	3586	3587	3588	3589	3590	3591	3592	3593	3594	3595	3596	3597	3598	3599
E1	3600	3601	3602	3603	3604	3605	3606	3607	3608	3609	3610	3611	3612	3613	3614	3615
E2	3616	3617	3618	3619	3620	3621	3622	3623	3624	3625	3626	3627	3628	3629	3630	3631
E3	3632	3633	3634	3635	3636	3637	3638	3639	3640	3641	3642	3643	3644	3645	3646	3647
E4	3648	3649	3650	3651	3652	3653	3654	3655	3656	3657	3658	3659	3660	3661	3662	3663
E5	3664	3665	3666	3667	3668	3669	3670	3671	3672	3673	3674	3675	3676	3677	3678	3679
E6	3680	3681	3682	3683	3684	3685	3686	3687	3688	3689	3690	3691	3692	3693	3694	3695
E7	3696	3697	3698	3699	3700	3701	3702	3703	3704	3705	3706	3707	3708	3709	3710	3711
E8	3712	3713	3714	3715	3716	3717	3718	3719	3720	3721	3722	3723	3724	3725	3726	3727
E9	3728	3729	3730	3731	3732	3733	3734	3735	3736	3737	3738	3739	3740	3741	3742	3743
EA	3744	3745	3746	3747	3748	3749	3750	3751	3752	3753	3754	3755	3756	3757	3758	3759
EB	3760	3761	3762	3763	3764	3765	3766	3767	3768	3769	3770	3771	3772	3773	3774	3775
EC	3776	3777	3778	3779	3780	3781	3782	3783	3784	3785	3786	3787	3788	3789	3790	3791
ED	3792	3793	3794	3795	3796	3797	3798	3799	3800	3801	3802	3803	3804	3805	3806	3807
EE	3808	3809	3810	3811	3812	3813	3814	3815	3816	3817	3818	3819	3820	3821	3822	3823
EF	3824	3825	3826	3827	3828	3829	3830	3831	3832	3833	3834	3835	3836	3837	3838	3839
F0	3840	3841	3842	3843	3844	3845	3846	3847	3848	3849	3850	3851	3852	3853	3854	3855
F1	3856	3857	3858	3859	3860	3861	3862	3863	3864	3865	3866	3867	3868	3869	3870	3871
F2	3872	3873	3874	3875	3876	3877	3878	3879	3880	3881	3882	3883	3884	3885	3886	3887
F3	3888	3889	3890	3891	3892	3893	3894	3895	3896	3897	3898	3899	3900	3901	3902	3903
F4	3904	3905	3906	3907	3908	3909	3910	3911	3912	3913	3914	3915	3916	3917	3918	3919
F5	3920	3921	3922	3923	3924	3925	3926	3927	3928	3929	3930	3931	3932	3933	3934	3935
F6	3936	3937	3938	3939	3940	3941	3942	3943	3944	3945	3946	3947	3948	3949	3950	3951
F7	3952	3953	3954	3955	3956	3957	3958	3959	3960	3961	3962	3963	3964	3965	3966	3967
F8	3968	3969	3970	3971	3972	3973	3974	3975	3976	3977	3978	3979	3980	3981	3982	3983
F9	3984	3985	3986	3987	3988	3989	3990	3991	3992	3993	3994	3995	3996	3997	3998	3999
FA	4000	4001	4002	4003	4004	4005	4006	4007	4008	4009	4010	4011	4012	4013	4014	4015
FB	4016	4017	4018	4019	4020	4021	4022	4023	4024	4025	4026	4027	4028	4029	4030	4031
FC	4032	4033	4034	4035	4036	4037	4038	4039	4040	4041	4042	4043	4044	4045	4046	4047
FD	4048	4049	4050	4051	4052	4053	4054	4055	4056	4057	4058	4059	4060	4061	4062	4063
FE	4064	4065	4066	4067	4068	4069	4070	4071	4072	4073	4074	4075	4076	4077	4078	4079
FF	4080	4081	4082	4083	4084	4085	4086	4087	4088	4089	4090	4091	4092	4093	4094	4095

APPENDIX E

HEXADECIMAL-TO-NEGATIVE INTEGER CONVERSION TABLE

	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
FFF	-1	-2	-3	-4	-5	-6	-7	-8	-9	-10	-11	-12	-13	-14	-15	-16
FFE	-17	-18	-19	-20	-21	-22	-23	-24	-25	-26	-27	-28	-29	-30	-31	-32
FFD	-33	-34	-35	-36	-37	-38	-39	-40	-41	-42	-43	-44	-45	-46	-47	-48
FFC	-49	-50	-51	-52	-53	-54	-55	-56	-57	-58	-59	-60	-61	-62	-63	-64
FFB	-65	-66	-67	-68	-69	-70	-71	-72	-73	-74	-75	-76	-77	-78	-79	-80
FFA	-81	-82	-83	-84	-85	-86	-87	-88	-89	-90	-91	-92	-93	-94	-95	-96
FF9	-97	-98	-99	-100	-101	-102	-103	-104	-105	-106	-107	-108	-109	-110	-111	-112
FF8	-113	-114	-115	-116	-117	-118	-119	-120	-121	-122	-123	-124	-125	-126	-127	-128
FF7	-129	-130	-131	-132	-133	-134	-135	-136	-137	-138	-139	-140	-141	-142	-143	-144
FF6	-145	-146	-147	-148	-149	-150	-151	-152	-153	-154	-155	-156	-157	-158	-159	-160
FF5	-161	-162	-163	-164	-165	-166	-167	-168	-169	-170	-171	-172	-173	-174	-175	-176
FF4	-177	-178	-179	-180	-181	-182	-183	-184	-185	-186	-187	-188	-189	-190	-191	-192
FF3	-193	-194	-195	-196	-197	-198	-199	-200	-201	-202	-203	-204	-205	-206	-207	-208
FF2	-209	-210	-211	-212	-213	-214	-215	-216	-217	-218	-219	-220	-221	-222	-223	-224
FF1	-225	-226	-227	-228	-229	-230	-231	-232	-233	-234	-235	-236	-237	-238	-239	-240
FF0	-241	-242	-243	-244	-245	-246	-247	-248	-249	-250	-251	-252	-253	-254	-255	-256
FEF	-257	-258	-259	-260	-261	-262	-263	-264	-265	-266	-267	-268	-269	-270	-271	-272
FEE	-273	-274	-275	-276	-277	-278	-279	-280	-281	-282	-283	-284	-285	-286	-287	-288
FED	-289	-290	-291	-292	-293	-294	-295	-296	-297	-298	-299	-300	-301	-302	-303	-304
FEC	-305	-306	-307	-308	-309	-310	-311	-312	-313	-314	-315	-316	-317	-318	-319	-320
FEB	-321	-322	-323	-324	-325	-326	-327	-328	-329	-330	-331	-332	-333	-334	-335	-336
FEA	-337	-338	-339	-340	-341	-342	-343	-344	-345	-346	-347	-348	-349	-350	-351	-352
FE9	-353	-354	-355	-356	-357	-358	-359	-360	-361	-362	-363	-364	-365	-366	-367	-368
FE8	-369	-370	-371	-372	-373	-374	-375	-376	-377	-378	-379	-380	-381	-382	-383	-384
FE7	-385	-386	-387	-388	-389	-390	-391	-392	-393	-394	-395	-396	-397	-398	-399	-400
FE6	-401	-402	-403	-404	-405	-406	-407	-408	-409	-410	-411	-412	-413	-414	-415	-416
FE5	-417	-418	-419	-420	-421	-422	-423	-424	-425	-426	-427	-428	-429	-430	-431	-432
FE4	-433	-434	-435	-436	-437	-438	-439	-440	-441	-442	-443	-444	-445	-446	-447	-448
FE3	-449	-450	-451	-452	-453	-454	-455	-456	-457	-458	-459	-460	-461	-462	-463	-464
FE2	-465	-466	-467	-468	-469	-470	-471	-472	-473	-474	-475	-476	-477	-478	-479	-480
FE1	-481	-482	-483	-484	-485	-486	-487	-488	-489	-490	-491	-492	-493	-494	-495	-496
FE0	-497	-498	-499	-500	-501	-502	-503	-504	-505	-506	-507	-508	-509	-510	-511	-512
FDF	-513	-514	-515	-516	-517	-518	-519	-520	-521	-522	-523	-524	-525	-526	-527	-528
FDE	-529	-530	-531	-532	-533	-534	-535	-536	-537	-538	-539	-540	-541	-542	-543	-544
FDD	-545	-546	-547	-548	-549	-550	-551	-552	-553	-554	-555	-556	-557	-558	-559	-560
FDC	-561	-562	-563	-564	-565	-566	-567	-568	-569	-570	-571	-572	-573	-574	-575	-576
FDB	-577	-578	-579	-580	-581	-582	-583	-584	-585	-586	-587	-588	-589	-590	-591	-592
FDA	-593	-594	-595	-596	-597	-598	-599	-600	-601	-602	-603	-604	-605	-606	-607	-608
FD9	-609	-610	-611	-612	-613	-614	-615	-616	-617	-618	-619	-620	-621	-622	-623	-624
FDB	-625	-626	-627	-628	-629	-630	-631	-632	-633	-634	-635	-636	-637	-638	-639	-640
FD7	-641	-642	-643	-644	-645	-646	-647	-648	-649	-650	-651	-652	-653	-654	-655	-656
FD6	-657	-658	-659	-660	-661	-662	-663	-664	-665	-666	-667	-668	-669	-670	-671	-672
FD5	-673	-674	-675	-676	-677	-678	-679	-680	-681	-682	-683	-684	-685	-686	-687	-688
FD4	-689	-690	-691	-692	-693	-694	-695	-696	-697	-698	-699	-700	-701	-702	-703	-704
FD3	-705	-706	-707	-708	-709	-710	-711	-712	-713	-714	-715	-716	-717	-718	-719	-720
FD2	-721	-722	-723	-724	-725	-726	-727	-728	-729	-730	-731	-732	-733	-734	-735	-736
FD1	-737	-738	-739	-740	-741	-742	-743	-744	-745	-746	-747	-748	-749	-750	-751	-752
FD0	-753	-754	-755	-756	-757	-758	-759	-760	-761	-762	-763	-764	-765	-766	-767	-768
FCF	-769	-770	-771	-772	-773	-774	-775	-776	-777	-778	-779	-780	-781	-782	-783	-784
FCE	-785	-786	-787	-788	-789	-790	-791	-792	-793	-794	-795	-796	-797	-798	-799	-800
FCD	-801	-802	-803	-804	-805	-806	-807	-808	-809	-810	-811	-812	-813	-814	-815	-816
FCC	-817	-818	-819	-820	-821	-822	-823	-824	-825	-826	-827	-828	-829	-830	-831	-832
FCB	-833	-834	-835	-836	-837	-838	-839	-840	-841	-842	-843	-844	-845	-846	-847	-848
FCA	-849	-850	-851	-852	-853	-854	-855	-856	-857	-858	-859	-860	-861	-862	-863	-864
FC9	-865	-866	-867	-868	-869	-870	-871	-872	-873	-874	-875	-876	-877	-878	-879	-880
FC8	-881	-882	-883	-884	-885	-886	-887	-888	-889	-890	-891	-892	-893	-894	-895	-896
FC7	-897	-898	-899	-900	-901	-902	-903	-904	-905	-906	-907	-908	-909	-910	-911	-912
FC6	-913	-914	-915	-916	-917	-918	-919	-920	-921	-922	-923	-924	-925	-926	-927	-928
FC5	-929	-930	-931	-932	-933	-934	-935	-936	-937	-938	-939	-940	-941	-942	-943	-944
FC4	-945	-946	-947	-948	-949	-950	-951	-952	-953	-954	-955	-956	-957	-958	-959	-960
FC3	-961	-962	-963	-964	-965	-966	-967	-968	-969	-970	-971	-972	-973	-974	-975	-976
FC2	-977	-978	-979	-980	-981	-982	-983	-984	-985	-986	-987	-988	-989	-990	-991	-992
FC1	-993	-994	-995	-996	-997	-998	-999	-1000	-1001	-1002	-1003	-1004	-1005	-1006	-1007	-1008
FC0	-1009	-1010	-1011	-1012	-1013	-1014	-1015	-1016	-1017	-1018	-1019	-1020	-1021	-1022	-1023	-1024

APPENDIX E

HEXADECIMAL-TO-NEGATIVE INTEGER CONVERSION TABLE (Cont)

	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
FBF	-1025	-1026	-1027	-1028	-1029	-1030	-1031	-1032	-1033	-1034	-1035	-1036	-1037	-1038	-1039	-1040
FBE	-1041	-1042	-1043	-1044	-1045	-1046	-1047	-1048	-1049	-1050	-1051	-1052	-1053	-1054	-1055	-1056
FBD	-1057	-1058	-1059	-1060	-1061	-1062	-1063	-1064	-1065	-1066	-1067	-1068	-1069	-1070	-1071	-1072
FBC	-1073	-1074	-1075	-1076	-1077	-1078	-1079	-1080	-1081	-1082	-1083	-1084	-1085	-1086	-1087	-1088
FBB	-1089	-1090	-1091	-1092	-1093	-1094	-1095	-1096	-1097	-1098	-1099	-1100	-1101	-1102	-1103	-1104
FBA	-1105	-1106	-1107	-1108	-1109	-1110	-1111	-1112	-1113	-1114	-1115	-1116	-1117	-1118	-1119	-1120
FB9	-1121	-1122	-1123	-1124	-1125	-1126	-1127	-1128	-1129	-1130	-1131	-1132	-1133	-1134	-1135	-1136
FBB	-1137	-1138	-1139	-1140	-1141	-1142	-1143	-1144	-1145	-1146	-1147	-1148	-1149	-1150	-1151	-1152
FB7	-1153	-1154	-1155	-1156	-1157	-1158	-1159	-1160	-1161	-1162	-1163	-1164	-1165	-1166	-1167	-1168
FB6	-1169	-1170	-1171	-1172	-1173	-1174	-1175	-1176	-1177	-1178	-1179	-1180	-1181	-1182	-1183	-1184
FB5	-1185	-1186	-1187	-1188	-1189	-1190	-1191	-1192	-1193	-1194	-1195	-1196	-1197	-1198	-1199	-1200
FB4	-1201	-1202	-1203	-1204	-1205	-1206	-1207	-1208	-1209	-1210	-1211	-1212	-1213	-1214	-1215	-1216
FB3	-1217	-1218	-1219	-1220	-1221	-1222	-1223	-1224	-1225	-1226	-1227	-1228	-1229	-1230	-1231	-1232
FB2	-1233	-1234	-1235	-1236	-1237	-1238	-1239	-1240	-1241	-1242	-1243	-1244	-1245	-1246	-1247	-1248
FB1	-1249	-1250	-1251	-1252	-1253	-1254	-1255	-1256	-1257	-1258	-1259	-1260	-1261	-1262	-1263	-1264
FB0	-1265	-1266	-1267	-1268	-1269	-1270	-1271	-1272	-1273	-1274	-1275	-1276	-1277	-1278	-1279	-1280
FAF	-1281	-1282	-1283	-1284	-1285	-1286	-1287	-1288	-1289	-1290	-1291	-1292	-1293	-1294	-1295	-1296
FAE	-1297	-1298	-1299	-1300	-1301	-1302	-1303	-1304	-1305	-1306	-1307	-1308	-1309	-1310	-1311	-1312
FAD	-1313	-1314	-1315	-1316	-1317	-1318	-1319	-1320	-1321	-1322	-1323	-1324	-1325	-1326	-1327	-1328
FAC	-1329	-1330	-1331	-1332	-1333	-1334	-1335	-1336	-1337	-1338	-1339	-1340	-1341	-1342	-1343	-1344
FAB	-1345	-1346	-1347	-1348	-1349	-1350	-1351	-1352	-1353	-1354	-1355	-1356	-1357	-1358	-1359	-1360
FAA	-1361	-1362	-1363	-1364	-1365	-1366	-1367	-1368	-1369	-1370	-1371	-1372	-1373	-1374	-1375	-1376
FA9	-1377	-1378	-1379	-1380	-1381	-1382	-1383	-1384	-1385	-1386	-1387	-1388	-1389	-1390	-1391	-1392
FAB	-1393	-1394	-1395	-1396	-1397	-1398	-1399	-1400	-1401	-1402	-1403	-1404	-1405	-1406	-1407	-1408
FA7	-1409	-1410	-1411	-1412	-1413	-1414	-1415	-1416	-1417	-1418	-1419	-1420	-1421	-1422	-1423	-1424
FA6	-1425	-1426	-1427	-1428	-1429	-1430	-1431	-1432	-1433	-1434	-1435	-1436	-1437	-1438	-1439	-1440
FA5	-1441	-1442	-1443	-1444	-1445	-1446	-1447	-1448	-1449	-1450	-1451	-1452	-1453	-1454	-1455	-1456
FA4	-1457	-1458	-1459	-1460	-1461	-1462	-1463	-1464	-1465	-1466	-1467	-1468	-1469	-1470	-1471	-1472
FA3	-1473	-1474	-1475	-1476	-1477	-1478	-1479	-1480	-1481	-1482	-1483	-1484	-1485	-1486	-1487	-1488
FA2	-1489	-1490	-1491	-1492	-1493	-1494	-1495	-1496	-1497	-1498	-1499	-1500	-1501	-1502	-1503	-1504
FA1	-1505	-1506	-1507	-1508	-1509	-1510	-1511	-1512	-1513	-1514	-1515	-1516	-1517	-1518	-1519	-1520
FA0	-1521	-1522	-1523	-1524	-1525	-1526	-1527	-1528	-1529	-1530	-1531	-1532	-1533	-1534	-1535	-1536
F9F	-1537	-1538	-1539	-1540	-1541	-1542	-1543	-1544	-1545	-1546	-1547	-1548	-1549	-1550	-1551	-1552
F9E	-1553	-1554	-1555	-1556	-1557	-1558	-1559	-1560	-1561	-1562	-1563	-1564	-1565	-1566	-1567	-1568
F9D	-1569	-1570	-1571	-1572	-1573	-1574	-1575	-1576	-1577	-1578	-1579	-1580	-1581	-1582	-1583	-1584
F9C	-1585	-1586	-1587	-1588	-1589	-1590	-1591	-1592	-1593	-1594	-1595	-1596	-1597	-1598	-1599	-1600
F9B	-1601	-1602	-1603	-1604	-1605	-1606	-1607	-1608	-1609	-1610	-1611	-1612	-1613	-1614	-1615	-1616
F9A	-1617	-1618	-1619	-1620	-1621	-1622	-1623	-1624	-1625	-1626	-1627	-1628	-1629	-1630	-1631	-1632
F99	-1633	-1634	-1635	-1636	-1637	-1638	-1639	-1640	-1641	-1642	-1643	-1644	-1645	-1646	-1647	-1648
F98	-1649	-1650	-1651	-1652	-1653	-1654	-1655	-1656	-1657	-1658	-1659	-1660	-1661	-1662	-1663	-1664
F97	-1665	-1666	-1667	-1668	-1669	-1670	-1671	-1672	-1673	-1674	-1675	-1676	-1677	-1678	-1679	-1680
F96	-1681	-1682	-1683	-1684	-1685	-1686	-1687	-1688	-1689	-1690	-1691	-1692	-1693	-1694	-1695	-1696
F95	-1697	-1698	-1699	-1700	-1701	-1702	-1703	-1704	-1705	-1706	-1707	-1708	-1709	-1710	-1711	-1712
F94	-1713	-1714	-1715	-1716	-1717	-1718	-1719	-1720	-1721	-1722	-1723	-1724	-1725	-1726	-1727	-1728
F93	-1729	-1730	-1731	-1732	-1733	-1734	-1735	-1736	-1737	-1738	-1739	-1740	-1741	-1742	-1743	-1744
F92	-1745	-1746	-1747	-1748	-1749	-1750	-1751	-1752	-1753	-1754	-1755	-1756	-1757	-1758	-1759	-1760
F91	-1761	-1762	-1763	-1764	-1765	-1766	-1767	-1768	-1769	-1770	-1771	-1772	-1773	-1774	-1775	-1776
F90	-1777	-1778	-1779	-1780	-1781	-1782	-1783	-1784	-1785	-1786	-1787	-1788	-1789	-1790	-1791	-1792
FBF	-1793	-1794	-1795	-1796	-1797	-1798	-1799	-1800	-1801	-1802	-1803	-1804	-1805	-1806	-1807	-1808
FBE	-1809	-1810	-1811	-1812	-1813	-1814	-1815	-1816	-1817	-1818	-1819	-1820	-1821	-1822	-1823	-1824
FBD	-1825	-1826	-1827	-1828	-1829	-1830	-1831	-1832	-1833	-1834	-1835	-1836	-1837	-1838	-1839	-1840
FBC	-1841	-1842	-1843	-1844	-1845	-1846	-1847	-1848	-1849	-1850	-1851	-1852	-1853	-1854	-1855	-1856
FBB	-1857	-1858	-1859	-1860	-1861	-1862	-1863	-1864	-1865	-1866	-1867	-1868	-1869	-1870	-1871	-1872
FBA	-1873	-1874	-1875	-1876	-1877	-1878	-1879	-1880	-1881	-1882	-1883	-1884	-1885	-1886	-1887	-1888
FB9	-1889	-1890	-1891	-1892	-1893	-1894	-1895	-1896	-1897	-1898	-1899	-1900	-1901	-1902	-1903	-1904
FBB	-1905	-1906	-1907	-1908	-1909	-1910	-1911	-1912	-1913	-1914	-1915	-1916	-1917	-1918	-1919	-1920
FB7	-1921	-1922	-1923	-1924	-1925	-1926	-1927	-1928	-1929	-1930	-1931	-1932	-1933	-1934	-1935	-1936
FB6	-1937	-1938	-1939	-1940	-1941	-1942	-1943	-1944	-1945	-1946	-1947	-1948	-1949	-1950	-1951	-1952
FB5	-1953	-1954	-1955	-1956	-1957	-1958	-1959	-1960	-1961	-1962	-1963	-1964	-1965	-1966	-1967	-1968
FB4	-1969	-1970	-1971	-1972	-1973	-1974	-1975	-1976	-1977	-1978	-1979	-1980	-1981	-1982	-1983	-1984
FB3	-1985	-1986	-1987	-1988	-1989	-1990	-1991	-1992	-1993	-1994	-1995	-1996	-1997	-1998	-1999	-2000
FB2	-2001	-2002	-2003	-2004	-2005	-2006	-2007	-2008	-2009	-2010	-2011	-2012	-2013	-2014	-2015	-2016
FB1	-2017	-2018	-2019	-2020	-2021	-2022	-2023	-2024	-2025	-2026	-2027	-2028	-2029	-2030	-2031	-2032
FB0	-2033	-2034	-2035	-2036	-2037	-2038	-2039	-2040	-2041	-2042	-2043	-2044	-2045	-2046	-2047	-2048

APPENDIX E

HEXADECIMAL-TO-NEGATIVE INTEGER CONVERSION TABLE (Cont)

	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
F7F	-2049	-2050	-2051	-2052	-2053	-2054	-2055	-2056	-2057	-2058	-2059	-2060	-2061	-2062	-2063	-2064
F7E	-2065	-2066	-2067	-2068	-2069	-2070	-2071	-2072	-2073	-2074	-2075	-2076	-2077	-2078	-2079	-2080
F7D	-2081	-2082	-2083	-2084	-2085	-2086	-2087	-2088	-2089	-2090	-2091	-2092	-2093	-2094	-2095	-2096
F7C	-2097	-2098	-2099	-2100	-2101	-2102	-2103	-2104	-2105	-2106	-2107	-2108	-2109	-2110	-2111	-2112
F7B	-2113	-2114	-2115	-2116	-2117	-2118	-2119	-2120	-2121	-2122	-2123	-2124	-2125	-2126	-2127	-2128
F7A	-2129	-2130	-2131	-2132	-2133	-2134	-2135	-2136	-2137	-2138	-2139	-2140	-2141	-2142	-2143	-2144
F79	-2145	-2146	-2147	-2148	-2149	-2150	-2151	-2152	-2153	-2154	-2155	-2156	-2157	-2158	-2159	-2160
F78	-2161	-2162	-2163	-2164	-2165	-2166	-2167	-2168	-2169	-2170	-2171	-2172	-2173	-2174	-2175	-2176
F77	-2177	-2178	-2179	-2180	-2181	-2182	-2183	-2184	-2185	-2186	-2187	-2188	-2189	-2190	-2191	-2192
F76	-2193	-2194	-2195	-2196	-2197	-2198	-2199	-2200	-2201	-2202	-2203	-2204	-2205	-2206	-2207	-2208
F75	-2209	-2210	-2211	-2212	-2213	-2214	-2215	-2216	-2217	-2218	-2219	-2220	-2221	-2222	-2223	-2224
F74	-2225	-2226	-2227	-2228	-2229	-2230	-2231	-2232	-2233	-2234	-2235	-2236	-2237	-2238	-2239	-2240
F73	-2241	-2242	-2243	-2244	-2245	-2246	-2247	-2248	-2249	-2250	-2251	-2252	-2253	-2254	-2255	-2256
F72	-2257	-2258	-2259	-2260	-2261	-2262	-2263	-2264	-2265	-2266	-2267	-2268	-2269	-2270	-2271	-2272
F71	-2273	-2274	-2275	-2276	-2277	-2278	-2279	-2280	-2281	-2282	-2283	-2284	-2285	-2286	-2287	-2288
F70	-2289	-2290	-2291	-2292	-2293	-2294	-2295	-2296	-2297	-2298	-2299	-2300	-2301	-2302	-2303	-2304
F6F	-2305	-2306	-2307	-2308	-2309	-2310	-2311	-2312	-2313	-2314	-2315	-2316	-2317	-2318	-2319	-2320
F6E	-2321	-2322	-2323	-2324	-2325	-2326	-2327	-2328	-2329	-2330	-2331	-2332	-2333	-2334	-2335	-2336
F6D	-2337	-2338	-2339	-2340	-2341	-2342	-2343	-2344	-2345	-2346	-2347	-2348	-2349	-2350	-2351	-2352
F6C	-2353	-2354	-2355	-2356	-2357	-2358	-2359	-2360	-2361	-2362	-2363	-2364	-2365	-2366	-2367	-2368
F6B	-2369	-2370	-2371	-2372	-2373	-2374	-2375	-2376	-2377	-2378	-2379	-2380	-2381	-2382	-2383	-2384
F6A	-2385	-2386	-2387	-2388	-2389	-2390	-2391	-2392	-2393	-2394	-2395	-2396	-2397	-2398	-2399	-2400
F69	-2401	-2402	-2403	-2404	-2405	-2406	-2407	-2408	-2409	-2410	-2411	-2412	-2413	-2414	-2415	-2416
F68	-2417	-2418	-2419	-2420	-2421	-2422	-2423	-2424	-2425	-2426	-2427	-2428	-2429	-2430	-2431	-2432
F67	-2433	-2434	-2435	-2436	-2437	-2438	-2439	-2440	-2441	-2442	-2443	-2444	-2445	-2446	-2447	-2448
F66	-2449	-2450	-2451	-2452	-2453	-2454	-2455	-2456	-2457	-2458	-2459	-2460	-2461	-2462	-2463	-2464
F65	-2465	-2466	-2467	-2468	-2469	-2470	-2471	-2472	-2473	-2474	-2475	-2476	-2477	-2478	-2479	-2480
F64	-2481	-2482	-2483	-2484	-2485	-2486	-2487	-2488	-2489	-2490	-2491	-2492	-2493	-2494	-2495	-2496
F63	-2497	-2498	-2499	-2500	-2501	-2502	-2503	-2504	-2505	-2506	-2507	-2508	-2509	-2510	-2511	-2512
F62	-2513	-2514	-2515	-2516	-2517	-2518	-2519	-2520	-2521	-2522	-2523	-2524	-2525	-2526	-2527	-2528
F61	-2529	-2530	-2531	-2532	-2533	-2534	-2535	-2536	-2537	-2538	-2539	-2540	-2541	-2542	-2543	-2544
F60	-2545	-2546	-2547	-2548	-2549	-2550	-2551	-2552	-2553	-2554	-2555	-2556	-2557	-2558	-2559	-2560
F5F	-2561	-2562	-2563	-2564	-2565	-2566	-2567	-2568	-2569	-2570	-2571	-2572	-2573	-2574	-2575	-2576
F5E	-2577	-2578	-2579	-2580	-2581	-2582	-2583	-2584	-2585	-2586	-2587	-2588	-2589	-2590	-2591	-2592
F5D	-2593	-2594	-2595	-2596	-2597	-2598	-2599	-2600	-2601	-2602	-2603	-2604	-2605	-2606	-2607	-2608
F5C	-2609	-2610	-2611	-2612	-2613	-2614	-2615	-2616	-2617	-2618	-2619	-2620	-2621	-2622	-2623	-2624
F5B	-2625	-2626	-2627	-2628	-2629	-2630	-2631	-2632	-2633	-2634	-2635	-2636	-2637	-2638	-2639	-2640
F5A	-2641	-2642	-2643	-2644	-2645	-2646	-2647	-2648	-2649	-2650	-2651	-2652	-2653	-2654	-2655	-2656
F59	-2657	-2658	-2659	-2660	-2661	-2662	-2663	-2664	-2665	-2666	-2667	-2668	-2669	-2670	-2671	-2672
F58	-2673	-2674	-2675	-2676	-2677	-2678	-2679	-2680	-2681	-2682	-2683	-2684	-2685	-2686	-2687	-2688
F57	-2689	-2690	-2691	-2692	-2693	-2694	-2695	-2696	-2697	-2698	-2699	-2700	-2701	-2702	-2703	-2704
F56	-2705	-2706	-2707	-2708	-2709	-2710	-2711	-2712	-2713	-2714	-2715	-2716	-2717	-2718	-2719	-2720
F55	-2721	-2722	-2723	-2724	-2725	-2726	-2727	-2728	-2729	-2730	-2731	-2732	-2733	-2734	-2735	-2736
F54	-2737	-2738	-2739	-2740	-2741	-2742	-2743	-2744	-2745	-2746	-2747	-2748	-2749	-2750	-2751	-2752
F53	-2753	-2754	-2755	-2756	-2757	-2758	-2759	-2760	-2761	-2762	-2763	-2764	-2765	-2766	-2767	-2768
F52	-2769	-2770	-2771	-2772	-2773	-2774	-2775	-2776	-2777	-2778	-2779	-2780	-2781	-2782	-2783	-2784
F51	-2785	-2786	-2787	-2788	-2789	-2790	-2791	-2792	-2793	-2794	-2795	-2796	-2797	-2798	-2799	-2800
F50	-2801	-2802	-2803	-2804	-2805	-2806	-2807	-2808	-2809	-2810	-2811	-2812	-2813	-2814	-2815	-2816
F4F	-2817	-2818	-2819	-2820	-2821	-2822	-2823	-2824	-2825	-2826	-2827	-2828	-2829	-2830	-2831	-2832
F4E	-2833	-2834	-2835	-2836	-2837	-2838	-2839	-2840	-2841	-2842	-2843	-2844	-2845	-2846	-2847	-2848
F4D	-2849	-2850	-2851	-2852	-2853	-2854	-2855	-2856	-2857	-2858	-2859	-2860	-2861	-2862	-2863	-2864
F4C	-2865	-2866	-2867	-2868	-2869	-2870	-2871	-2872	-2873	-2874	-2875	-2876	-2877	-2878	-2879	-2880
F4B	-2881	-2882	-2883	-2884	-2885	-2886	-2887	-2888	-2889	-2890	-2891	-2892	-2893	-2894	-2895	-2896
F4A	-2897	-2898	-2899	-2900	-2901	-2902	-2903	-2904	-2905	-2906	-2907	-2908	-2909	-2910	-2911	-2912
F49	-2913	-2914	-2915	-2916	-2917	-2918	-2919	-2920	-2921	-2922	-2923	-2924	-2925	-2926	-2927	-2928
F48	-2929	-2930	-2931	-2932	-2933	-2934	-2935	-2936	-2937	-2938	-2939	-2940	-2941	-2942	-2943	-2944
F47	-2945	-2946	-2947	-2948	-2949	-2950	-2951	-2952	-2953	-2954	-2955	-2956	-2957	-2958	-2959	-2960
F46	-2961	-2962	-2963	-2964	-2965	-2966	-2967	-2968	-2969	-2970	-2971	-2972	-2973	-2974	-2975	-2976
F45	-2977	-2978	-2979	-2980	-2981	-2982	-2983	-2984	-2985	-2986	-2987	-2988	-2989	-2990	-2991	-2992
F44	-2993	-2994	-2995	-2996	-2997	-2998	-2999	-3000	-3001	-3002	-3003	-3004	-3005	-3006	-3007	-3008
F43	-3009	-3010	-3011	-3012	-3013	-3014	-3015	-3016	-3017	-3018	-3019	-3020	-3021	-3022	-3023	-3024
F42	-3025	-3026	-3027	-3028	-3029	-3030	-3031	-3032	-3033	-3034	-3035	-3036	-3037	-3038	-3039	-3040
F41	-3041	-3042	-3043	-3044	-3045	-3046	-3047	-3048	-3049	-3050	-3051	-3052	-3053	-3054	-3055	-3056
F40	-3057	-3058	-3059	-3060	-3061	-3062	-3063	-3064	-3065	-3066	-3067	-3068	-3069	-3070	-3071	-3072

APPENDIX E

HEXADECIMAL-TO-NEGATIVE INTEGER CONVERSION TABLE (Cont)

	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
F3F	-3073	-3074	-3075	-3076	-3077	-3078	-3079	-3080	-3081	-3082	-3083	-3084	-3085	-3086	-3087	-3088
F3E	-3069	-3090	-3091	-3092	-3093	-3094	-3095	-3096	-3097	-3098	-3099	-3100	-3101	-3102	-3103	-3104
F3D	-3105	-3106	-3107	-3108	-3109	-3110	-3111	-3112	-3113	-3114	-3115	-3116	-3117	-3118	-3119	-3120
F3C	-3121	-3122	-3123	-3124	-3125	-3126	-3127	-3128	-3129	-3130	-3131	-3132	-3133	-3134	-3135	-3136
F3B	-3137	-3138	-3139	-3140	-3141	-3142	-3143	-3144	-3145	-3146	-3147	-3148	-3149	-3150	-3151	-3152
F3A	-3153	-3154	-3155	-3156	-3157	-3158	-3159	-3160	-3161	-3162	-3163	-3164	-3165	-3166	-3167	-3168
F39	-3169	-3170	-3171	-3172	-3173	-3174	-3175	-3176	-3177	-3178	-3179	-3180	-3181	-3182	-3183	-3184
F38	-3185	-3186	-3187	-3188	-3189	-3190	-3191	-3192	-3193	-3194	-3195	-3196	-3197	-3198	-3199	-3200
F37	-3201	-3202	-3203	-3204	-3205	-3206	-3207	-3208	-3209	-3210	-3211	-3212	-3213	-3214	-3215	-3216
F36	-3217	-3218	-3219	-3220	-3221	-3222	-3223	-3224	-3225	-3226	-3227	-3228	-3229	-3230	-3231	-3232
F35	-3233	-3234	-3235	-3236	-3237	-3238	-3239	-3240	-3241	-3242	-3243	-3244	-3245	-3246	-3247	-3248
F34	-3249	-3250	-3251	-3252	-3253	-3254	-3255	-3256	-3257	-3258	-3259	-3260	-3261	-3262	-3263	-3264
F33	-3265	-3266	-3267	-3268	-3269	-3270	-3271	-3272	-3273	-3274	-3275	-3276	-3277	-3278	-3279	-3280
F32	-3281	-3282	-3283	-3284	-3285	-3286	-3287	-3288	-3289	-3290	-3291	-3292	-3293	-3294	-3295	-3296
F31	-3297	-3298	-3299	-3300	-3301	-3302	-3303	-3304	-3305	-3306	-3307	-3308	-3309	-3310	-3311	-3312
F30	-3313	-3314	-3315	-3316	-3317	-3318	-3319	-3320	-3321	-3322	-3323	-3324	-3325	-3326	-3327	-3328
F2F	-3329	-3330	-3331	-3332	-3333	-3334	-3335	-3336	-3337	-3338	-3339	-3340	-3341	-3342	-3343	-3344
F2E	-3345	-3346	-3347	-3348	-3349	-3350	-3351	-3352	-3353	-3354	-3355	-3356	-3357	-3358	-3359	-3360
F2D	-3361	-3362	-3363	-3364	-3365	-3366	-3367	-3368	-3369	-3370	-3371	-3372	-3373	-3374	-3375	-3376
F2C	-3377	-3378	-3379	-3380	-3381	-3382	-3383	-3384	-3385	-3386	-3387	-3388	-3389	-3390	-3391	-3392
F2B	-3393	-3394	-3395	-3396	-3397	-3398	-3399	-3400	-3401	-3402	-3403	-3404	-3405	-3406	-3407	-3408
F2A	-3409	-3410	-3411	-3412	-3413	-3414	-3415	-3416	-3417	-3418	-3419	-3420	-3421	-3422	-3423	-3424
F29	-3425	-3426	-3427	-3428	-3429	-3430	-3431	-3432	-3433	-3434	-3435	-3436	-3437	-3438	-3439	-3440
F28	-3441	-3442	-3443	-3444	-3445	-3446	-3447	-3448	-3449	-3450	-3451	-3452	-3453	-3454	-3455	-3456
F27	-3457	-3458	-3459	-3460	-3461	-3462	-3463	-3464	-3465	-3466	-3467	-3468	-3469	-3470	-3471	-3472
F26	-3473	-3474	-3475	-3476	-3477	-3478	-3479	-3480	-3481	-3482	-3483	-3484	-3485	-3486	-3487	-3488
F25	-3489	-3490	-3491	-3492	-3493	-3494	-3495	-3496	-3497	-3498	-3499	-3500	-3501	-3502	-3503	-3504
F24	-3505	-3506	-3507	-3508	-3509	-3510	-3511	-3512	-3513	-3514	-3515	-3516	-3517	-3518	-3519	-3520
F23	-3521	-3522	-3523	-3524	-3525	-3526	-3527	-3528	-3529	-3530	-3531	-3532	-3533	-3534	-3535	-3536
F22	-3537	-3538	-3539	-3540	-3541	-3542	-3543	-3544	-3545	-3546	-3547	-3548	-3549	-3550	-3551	-3552
F21	-3553	-3554	-3555	-3556	-3557	-3558	-3559	-3560	-3561	-3562	-3563	-3564	-3565	-3566	-3567	-3568
F20	-3569	-3570	-3571	-3572	-3573	-3574	-3575	-3576	-3577	-3578	-3579	-3580	-3581	-3582	-3583	-3584
F1F	-3585	-3586	-3587	-3588	-3589	-3590	-3591	-3592	-3593	-3594	-3595	-3596	-3597	-3598	-3599	-3600
F1E	-3601	-3602	-3603	-3604	-3605	-3606	-3607	-3608	-3609	-3610	-3611	-3612	-3613	-3614	-3615	-3616
F1D	-3617	-3618	-3619	-3620	-3621	-3622	-3623	-3624	-3625	-3626	-3627	-3628	-3629	-3630	-3631	-3632
F1C	-3633	-3634	-3635	-3636	-3637	-3638	-3639	-3640	-3641	-3642	-3643	-3644	-3645	-3646	-3647	-3648
F1B	-3649	-3650	-3651	-3652	-3653	-3654	-3655	-3656	-3657	-3658	-3659	-3660	-3661	-3662	-3663	-3664
F1A	-3665	-3666	-3667	-3668	-3669	-3670	-3671	-3672	-3673	-3674	-3675	-3676	-3677	-3678	-3679	-3680
F19	-3681	-3682	-3683	-3684	-3685	-3686	-3687	-3688	-3689	-3690	-3691	-3692	-3693	-3694	-3695	-3696
F18	-3697	-3698	-3699	-3700	-3701	-3702	-3703	-3704	-3705	-3706	-3707	-3708	-3709	-3710	-3711	-3712
F17	-3713	-3714	-3715	-3716	-3717	-3718	-3719	-3720	-3721	-3722	-3723	-3724	-3725	-3726	-3727	-3728
F16	-3729	-3730	-3731	-3732	-3733	-3734	-3735	-3736	-3737	-3738	-3739	-3740	-3741	-3742	-3743	-3744
F15	-3745	-3746	-3747	-3748	-3749	-3750	-3751	-3752	-3753	-3754	-3755	-3756	-3757	-3758	-3759	-3760
F14	-3761	-3762	-3763	-3764	-3765	-3766	-3767	-3768	-3769	-3770	-3771	-3772	-3773	-3774	-3775	-3776
F13	-3777	-3778	-3779	-3780	-3781	-3782	-3783	-3784	-3785	-3786	-3787	-3788	-3789	-3790	-3791	-3792
F12	-3793	-3794	-3795	-3796	-3797	-3798	-3799	-3800	-3801	-3802	-3803	-3804	-3805	-3806	-3807	-3808
F11	-3809	-3810	-3811	-3812	-3813	-3814	-3815	-3816	-3817	-3818	-3819	-3820	-3821	-3822	-3823	-3824
F10	-3825	-3826	-3827	-3828	-3829	-3830	-3831	-3832	-3833	-3834	-3835	-3836	-3837	-3838	-3839	-3840
F0F	-3841	-3842	-3843	-3844	-3845	-3846	-3847	-3848	-3849	-3850	-3851	-3852	-3853	-3854	-3855	-3856
F0E	-3857	-3858	-3859	-3860	-3861	-3862	-3863	-3864	-3865	-3866	-3867	-3868	-3869	-3870	-3871	-3872
F0D	-3873	-3874	-3875	-3876	-3877	-3878	-3879	-3880	-3881	-3882	-3883	-3884	-3885	-3886	-3887	-3888
F0C	-3889	-3890	-3891	-3892	-3893	-3894	-3895	-3896	-3897	-3898	-3899	-3900	-3901	-3902	-3903	-3904
F0B	-3905	-3906	-3907	-3908	-3909	-3910	-3911	-3912	-3913	-3914	-3915	-3916	-3917	-3918	-3919	-3920
F0A	-3921	-3922	-3923	-3924	-3925	-3926	-3927	-3928	-3929	-3930	-3931	-3932	-3933	-3934	-3935	-3936
F09	-3937	-3938	-3939	-3940	-3941	-3942	-3943	-3944	-3945	-3946	-3947	-3948	-3949	-3950	-3951	-3952
F08	-3953	-3954	-3955	-3956	-3957	-3958	-3959	-3960	-3961	-3962	-3963	-3964	-3965	-3966	-3967	-3968
F07	-3969	-3970	-3971	-3972	-3973	-3974	-3975	-3976	-3977	-3978	-3979	-3980	-3981	-3982	-3983	-3984
F06	-3985	-3986	-3987	-3988	-3989	-3990	-3991	-3992	-3993	-3994	-3995	-3996	-3997	-3998	-3999	-4000
F05	-4001	-4002	-4003	-4004	-4005	-4006	-4007	-4008	-4009	-4010	-4011	-4012	-4013	-4014	-4015	-4016
F04	-4017	-4018	-4019	-4020	-4021	-4022	-4023	-4024	-4025	-4026	-4027	-4028	-4029	-4030	-4031	-4032
F03	-4033	-4034	-4035	-4036	-4037	-4038	-4039	-4040	-4041	-4042	-4043	-4044	-4045	-4046	-4047	-4048
F02	-4049	-4050	-4051	-4052	-4053	-4054	-4055	-4056	-4057	-4058	-4059	-4060	-4061	-4062	-4063	-4064
F01	-4065	-4066	-4067	-4068	-4069	-4070	-4071	-4072	-4073	-4074	-4075	-4076	-4077	-4078	-4079	-4080
F00	-4081	-4082	-4083	-4084	-4085	-4086	-4087	-4088	-4089	-4090	-4091	-4092	-4093	-4094	-4095	-4096

APPENDIX E

HEXADECIMAL-TO-DECIMAL FRACTION TABLE (Cont)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
.00	.00000	.00024	.00049	.00073	.00098	.00122	.00146	.00171	.00195	.00220	.00244	.00269	.00293	.00317	.00342	.00366
.01	.00391	.00415	.00439	.00464	.00488	.00513	.00537	.00562	.00586	.00610	.00635	.00659	.00684	.00708	.00732	.00757
.02	.00781	.00806	.00830	.00854	.00879	.00903	.00928	.00952	.00977	.01001	.01025	.01050	.01074	.01099	.01123	.01147
.03	.01172	.01196	.01221	.01245	.01270	.01294	.01318	.01343	.01367	.01392	.01416	.01440	.01465	.01489	.01514	.01538
.04	.01562	.01587	.01611	.01636	.01660	.01685	.01709	.01733	.01758	.01782	.01807	.01831	.01855	.01880	.01904	.01929
.05	.01953	.01978	.02002	.02026	.02051	.02075	.02100	.02124	.02148	.02173	.02197	.02222	.02246	.02271	.02295	.02319
.06	.02344	.02368	.02393	.02417	.02441	.02466	.02490	.02515	.02539	.02563	.02588	.02612	.02637	.02661	.02686	.02710
.07	.02734	.02759	.02783	.02808	.02832	.02856	.02881	.02905	.02930	.02954	.02979	.03003	.03027	.03052	.03076	.03101
.08	.03125	.03149	.03174	.03198	.03223	.03247	.03271	.03296	.03320	.03345	.03369	.03394	.03418	.03442	.03467	.03491
.09	.03516	.03540	.03564	.03589	.03613	.03638	.03662	.03687	.03711	.03735	.03760	.03784	.03808	.03833	.03857	.03882
.0A	.03906	.03931	.03955	.03979	.04004	.04028	.04053	.04077	.04102	.04126	.04150	.04175	.04199	.04224	.04248	.04272
.0B	.04297	.04321	.04346	.04370	.04395	.04419	.04443	.04468	.04492	.04517	.04541	.04565	.04590	.04614	.04639	.04663
.0C	.04687	.04712	.04736	.04761	.04785	.04810	.04834	.04858	.04883	.04907	.04932	.04956	.04980	.05005	.05029	.05054
.0D	.05078	.05103	.05127	.05151	.05176	.05200	.05225	.05249	.05273	.05298	.05322	.05347	.05371	.05396	.05420	.05444
.0E	.05469	.05493	.05518	.05542	.05566	.05591	.05615	.05640	.05664	.05689	.05713	.05737	.05761	.05786	.05810	.05835
.0F	.05859	.05884	.05908	.05933	.05957	.05981	.06006	.06030	.06055	.06079	.06104	.06128	.06152	.06177	.06201	.06226
.10	.06250	.06274	.06299	.06323	.06348	.06372	.06396	.06421	.06445	.06470	.06494	.06519	.06543	.06567	.06592	.06616
.11	.06641	.06665	.06689	.06714	.06738	.06763	.06787	.06812	.06836	.06860	.06885	.06909	.06934	.06958	.06982	.07007
.12	.07031	.07055	.07080	.07104	.07129	.07153	.07178	.07202	.07227	.07251	.07275	.07300	.07324	.07349	.07373	.07397
.13	.07422	.07446	.07471	.07495	.07520	.07544	.07568	.07593	.07617	.07642	.07666	.07690	.07715	.07739	.07764	.07788
.14	.07812	.07837	.07861	.07886	.07910	.07935	.07959	.07983	.08008	.08032	.08057	.08081	.08105	.08130	.08154	.08179
.15	.08203	.08228	.08252	.08276	.08301	.08325	.08350	.08374	.08398	.08423	.08447	.08472	.08496	.08521	.08545	.08569
.16	.08594	.08618	.08643	.08667	.08691	.08716	.08740	.08765	.08789	.08813	.08838	.08862	.08887	.08911	.08936	.08960
.17	.08984	.09009	.09033	.09058	.09082	.09106	.09131	.09155	.09180	.09204	.09229	.09253	.09277	.09302	.09326	.09351
.18	.09375	.09399	.09424	.09448	.09473	.09497	.09521	.09546	.09570	.09595	.09619	.09644	.09668	.09692	.09717	.09741
.19	.09766	.09790	.09814	.09839	.09863	.09888	.09912	.09937	.09961	.09985	.10010	.10034	.10059	.10083	.10107	.10132
.1A	.10156	.10181	.10205	.10229	.10254	.10278	.10303	.10327	.10352	.10376	.10400	.10425	.10449	.10474	.10498	.10522
.1B	.10547	.10571	.10596	.10620	.10645	.10669	.10693	.10718	.10742	.10767	.10791	.10815	.10840	.10864	.10889	.10913
.1C	.10937	.10962	.10986	.11011	.11035	.11060	.11084	.11108	.11133	.11157	.11182	.11206	.11230	.11255	.11279	.11304
.1D	.11328	.11353	.11377	.11401	.11426	.11450	.11475	.11499	.11523	.11548	.11572	.11597	.11621	.11646	.11670	.11694
.1E	.11719	.11743	.11768	.11792	.11816	.11841	.11865	.11890	.11914	.11938	.11963	.11987	.12012	.12036	.12061	.12085
.1F	.12109	.12134	.12158	.12183	.12207	.12231	.12256	.12280	.12305	.12329	.12354	.12378	.12402	.12427	.12451	.12476
.20	.12500	.12524	.12549	.12573	.12598	.12622	.12646	.12671	.12695	.12720	.12744	.12769	.12793	.12817	.12842	.12866
.21	.12891	.12915	.12939	.12964	.12988	.13013	.13037	.13062	.13086	.13110	.13135	.13159	.13184	.13208	.13232	.13257
.22	.13281	.13306	.13330	.13354	.13379	.13403	.13428	.13452	.13477	.13501	.13525	.13550	.13574	.13599	.13623	.13647
.23	.13672	.13696	.13721	.13745	.13770	.13794	.13818	.13843	.13867	.13892	.13916	.13940	.13965	.13989	.14014	.14038
.24	.14062	.14087	.14111	.14136	.14160	.14185	.14209	.14233	.14258	.14282	.14307	.14331	.14355	.14380	.14404	.14429
.25	.14453	.14478	.14502	.14526	.14551	.14575	.14600	.14624	.14648	.14673	.14697	.14722	.14746	.14771	.14795	.14819
.26	.14844	.14868	.14893	.14917	.14941	.14966	.14990	.15015	.15039	.15063	.15088	.15112	.15137	.15161	.15186	.15210
.27	.15234	.15259	.15283	.15308	.15332	.15356	.15381	.15405	.15430	.15454	.15479	.15503	.15527	.15552	.15576	.15601
.28	.15625	.15649	.15674	.15698	.15723	.15747	.15771	.15796	.15820	.15845	.15869	.15894	.15918	.15942	.15967	.15991
.29	.16016	.16040	.16064	.16089	.16113	.16138	.16162	.16187	.16211	.16235	.16260	.16284	.16309	.16333	.16357	.16382
.2A	.16406	.16431	.16455	.16479	.16504	.16528	.16553	.16577	.16602	.16626	.16650	.16675	.16699	.16724	.16748	.16772
.2B	.16797	.16821	.16846	.16870	.16895	.16919	.16943	.16968	.16992	.17017	.17041	.17065	.17090	.17114	.17139	.17163
.2C	.17187	.17212	.17236	.17261	.17285	.17310	.17334	.17358	.17383	.17407	.17432	.17456	.17480	.17505	.17529	.17554
.2D	.17578	.17603	.17627	.17651	.17676	.17700	.17725	.17749	.17773	.17798	.17822	.17847	.17871	.17896	.17920	.17944
.2E	.17969	.17993	.18018	.18042	.18066	.18091	.18115	.18140	.18164	.18188	.18213	.18237	.18262	.18286	.18311	.18335
.2F	.18359	.18384	.18408	.18433	.18457	.18481	.18506	.18530	.18555	.18579	.18604	.18628	.18652	.18677	.18701	.18726
.30	.18750	.18774	.18799	.18823	.18848	.18872	.18896	.18921	.18945	.18970	.18994	.19019	.19043	.19067	.19092	.19116
.31	.19141	.19165	.19189	.19214	.19238	.19263	.19287	.19312	.19336	.19360	.19385	.19409	.19434	.19458	.19482	.19507
.32	.19531	.19556	.19580	.19604	.19629	.19653	.19678	.19702	.19727	.19751	.19775	.19800	.19824	.19849	.19873	.19897
.33	.19922	.19946	.19971	.19995	.20020	.20044	.20068	.20093	.20117	.20142	.20166	.20190	.20215	.20239	.20264	.20288
.34	.20312	.20337	.20361	.20386	.20410	.20435	.20459	.20483	.20508	.20532	.20557	.20581	.20605	.20630	.20654	.20679
.35	.20703	.20728	.20752	.20776	.20801	.20825	.20850	.20874	.20898	.20923	.20947	.20972	.20996	.21021	.21045	.21069
.36	.21094	.21118	.21143	.21167	.21191	.21216	.21240	.21265	.21289	.21313	.21338	.21362	.21387	.21411	.21436	.21460
.37	.21484	.21509	.21533	.21558	.21582	.21606	.21631	.21655	.21680	.21704	.21729	.21753	.21777	.21802	.21826	.21851
.38	.21875	.21899	.21924	.21948	.21973	.21997	.22021	.22046	.22070	.22095	.22119	.22144	.22168	.22192	.22217	.22241
.39	.22266	.22290	.22314	.22339	.22363	.22388	.22412	.22437	.22461	.22485	.22510	.22534	.22559	.22583	.22607	.22632
.3A	.22656	.22681	.22705	.22729	.22754	.22778	.22803	.22827	.22852	.22876	.22900	.22925	.22949	.22974	.22998	.23022
.3B	.23047	.23071	.23096	.23120	.23145	.23169	.23193	.23218	.23242	.23267	.23291	.23315	.23340	.23364	.23389	.23413
.3C	.23437	.23462	.23486	.23511	.23535	.23560	.23584	.23608	.23633	.23657	.23682	.23706	.23730	.23755	.23779	.23804
.3D	.23828	.23853	.23877	.23901	.23926	.23950	.23975	.23999	.24023	.24048	.24072	.24097	.24121	.24146	.24170	.24194
.3E	.24219	.24243	.24268	.24292	.24316	.24341	.24365	.24390	.24414	.24438	.24463	.24487	.24512	.24536	.24561	.24585
.3F	.24609	.24634	.24658	.24683	.24707	.24731	.24756	.24780	.24805	.24829	.24854	.24878	.24902	.24927	.24951	.24976

APPENDIX E

HEXADECIMAL-TO-DECIMAL FRACTION TABLE (Cont)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
.40	.25000	.25024	.25049	.25073	.25098	.25122	.25146	.25171	.25195	.25220	.25244	.25269	.25293	.25317	.25342	.25366
.41	.25391	.25415	.25439	.25464	.25488	.25513	.25537	.25562	.25586	.25610	.25635	.25659	.25684	.25708	.25732	.25757
.42	.25781	.25806	.25830	.25854	.25879	.25903	.25928	.25952	.25977	.26001	.26025	.26050	.26074	.26099	.26123	.26147
.43	.26172	.26196	.26221	.26245	.26270	.26294	.26318	.26343	.26367	.26392	.26416	.26440	.26465	.26489	.26514	.26538
.44	.26562	.26587	.26611	.26636	.26660	.26685	.26709	.26733	.26758	.26782	.26807	.26831	.26855	.26880	.26904	.26929
.45	.26953	.26978	.27002	.27026	.27051	.27075	.27100	.27124	.27148	.27173	.27197	.27222	.27246	.27271	.27295	.27319
.46	.27344	.27368	.27393	.27417	.27441	.27466	.27490	.27515	.27539	.27563	.27588	.27612	.27637	.27661	.27686	.27710
.47	.27734	.27759	.27783	.27808	.27832	.27856	.27881	.27905	.27930	.27954	.27979	.28003	.28027	.28052	.28076	.28101
.48	.28125	.28149	.28174	.28198	.28223	.28247	.28271	.28296	.28320	.28345	.28369	.28394	.28418	.28442	.28467	.28491
.49	.28516	.28540	.28564	.28589	.28613	.28638	.28662	.28687	.28711	.28735	.28760	.28784	.28809	.28833	.28857	.28882
.4A	.28906	.28931	.28955	.28979	.29004	.29028	.29053	.29077	.29102	.29126	.29150	.29175	.29199	.29224	.29248	.29272
.4B	.29297	.29321	.29346	.29370	.29395	.29419	.29443	.29468	.29492	.29517	.29541	.29565	.29590	.29614	.29639	.29663
.4C	.29687	.29712	.29736	.29761	.29785	.29810	.29834	.29858	.29883	.29907	.29932	.29956	.29980	.30005	.30029	.30054
.4D	.30078	.30103	.30127	.30151	.30176	.30200	.30225	.30249	.30273	.30298	.30322	.30347	.30371	.30396	.30420	.30444
.4E	.30469	.30493	.30518	.30542	.30566	.30591	.30615	.30640	.30664	.30688	.30713	.30737	.30762	.30786	.30810	.30835
.4F	.30859	.30884	.30908	.30933	.30957	.30981	.31006	.31030	.31054	.31079	.31103	.31128	.31152	.31177	.31201	.31226
.50	.31250	.31274	.31299	.31323	.31348	.31372	.31396	.31421	.31445	.31470	.31494	.31519	.31543	.31567	.31592	.31616
.51	.31641	.31665	.31689	.31714	.31738	.31763	.31787	.31812	.31836	.31860	.31885	.31909	.31934	.31958	.31982	.32007
.52	.32031	.32056	.32080	.32104	.32129	.32153	.32178	.32202	.32227	.32251	.32275	.32300	.32324	.32349	.32373	.32397
.53	.32422	.32446	.32471	.32495	.32520	.32544	.32568	.32593	.32617	.32642	.32666	.32690	.32715	.32739	.32764	.32788
.54	.32812	.32837	.32861	.32886	.32910	.32935	.32959	.32983	.33008	.33032	.33057	.33081	.33105	.33130	.33154	.33179
.55	.33203	.33228	.33252	.33276	.33301	.33325	.33350	.33374	.33398	.33423	.33447	.33472	.33496	.33521	.33545	.33569
.56	.33594	.33618	.33643	.33667	.33691	.33716	.33740	.33765	.33789	.33813	.33838	.33862	.33887	.33911	.33936	.33960
.57	.33984	.34009	.34033	.34058	.34082	.34106	.34131	.34155	.34180	.34204	.34229	.34253	.34277	.34302	.34326	.34351
.58	.34375	.34399	.34424	.34448	.34473	.34497	.34521	.34546	.34570	.34595	.34619	.34644	.34668	.34692	.34717	.34741
.59	.34766	.34790	.34814	.34839	.34863	.34888	.34912	.34937	.34961	.34985	.35010	.35034	.35059	.35083	.35107	.35132
.5A	.35156	.35181	.35205	.35229	.35254	.35278	.35303	.35327	.35352	.35376	.35400	.35425	.35449	.35473	.35498	.35522
.5B	.35547	.35571	.35596	.35620	.35645	.35669	.35693	.35718	.35742	.35767	.35791	.35815	.35840	.35864	.35889	.35913
.5C	.35937	.35962	.35986	.36011	.36035	.36060	.36084	.36108	.36133	.36157	.36182	.36206	.36230	.36255	.36279	.36304
.5D	.36328	.36353	.36377	.36401	.36426	.36450	.36475	.36499	.36523	.36548	.36572	.36597	.36621	.36646	.36670	.36694
.5E	.36719	.36743	.36768	.36792	.36816	.36841	.36865	.36890	.36914	.36938	.36963	.36987	.37012	.37036	.37061	.37085
.5F	.37109	.37134	.37158	.37183	.37207	.37231	.37256	.37280	.37305	.37329	.37354	.37378	.37402	.37427	.37451	.37476
.60	.37500	.37524	.37549	.37573	.37598	.37622	.37646	.37671	.37695	.37720	.37744	.37769	.37793	.37817	.37842	.37866
.61	.37891	.37915	.37939	.37964	.37988	.38008	.38037	.38062	.38086	.38110	.38135	.38159	.38184	.38208	.38232	.38257
.62	.38281	.38306	.38330	.38354	.38379	.38403	.38428	.38452	.38477	.38501	.38525	.38550	.38574	.38599	.38623	.38647
.63	.38672	.38696	.38721	.38745	.38770	.38794	.38818	.38843	.38867	.38892	.38916	.38940	.38965	.38989	.39014	.39038
.64	.39062	.39087	.39111	.39136	.39160	.39185	.39209	.39233	.39258	.39282	.39307	.39331	.39355	.39380	.39404	.39429
.65	.39453	.39478	.39502	.39526	.39551	.39575	.39600	.39624	.39648	.39673	.39697	.39722	.39746	.39771	.39795	.39819
.66	.39844	.39868	.39893	.39917	.39941	.39966	.39990	.40015	.40039	.40063	.40088	.40112	.40137	.40161	.40186	.40210
.67	.40234	.40259	.40283	.40308	.40332	.40356	.40381	.40405	.40430	.40454	.40479	.40503	.40527	.40552	.40576	.40601
.68	.40625	.40649	.40674	.40698	.40723	.40747	.40771	.40796	.40820	.40845	.40869	.40894	.40918	.40942	.40967	.40991
.69	.41016	.41040	.41064	.41089	.41113	.41138	.41162	.41187	.41211	.41235	.41260	.41284	.41309	.41333	.41357	.41382
.6A	.41406	.41431	.41455	.41479	.41504	.41528	.41553	.41577	.41602	.41626	.41650	.41675	.41699	.41724	.41748	.41772
.6B	.41797	.41821	.41846	.41870	.41895	.41919	.41943	.41968	.41992	.42017	.42041	.42065	.42090	.42114	.42139	.42163
.6C	.42187	.42212	.42236	.42261	.42285	.42310	.42334	.42358	.42383	.42407	.42432	.42456	.42480	.42505	.42529	.42554
.6D	.42578	.42603	.42627	.42651	.42676	.42700	.42725	.42749	.42773	.42798	.42822	.42847	.42871	.42896	.42920	.42944
.6E	.42969	.42993	.43018	.43042	.43066	.43091	.43115	.43140	.43164	.43188	.43213	.43237	.43262	.43286	.43310	.43335
.6F	.43359	.43384	.43408	.43433	.43457	.43481	.43506	.43530	.43554	.43579	.43604	.43628	.43652	.43677	.43701	.43726
.70	.43750	.43774	.43799	.43823	.43848	.43872	.43896	.43921	.43945	.43970	.43994	.44019	.44043	.44067	.44092	.44116
.71	.44141	.44165	.44189	.44214	.44238	.44263	.44287	.44312	.44336	.44360	.44385	.44409	.44434	.44458	.44482	.44507
.72	.44531	.44556	.44580	.44604	.44629	.44653	.44678	.44702	.44727	.44751	.44775	.44800	.44824	.44849	.44873	.44897
.73	.44922	.44946	.44971	.44995	.45020	.45044	.45068	.45093	.45117	.45142	.45166	.45190	.45215	.45239	.45264	.45288
.74	.45312	.45337	.45361	.45386	.45410	.45435	.45459	.45483	.45508	.45532	.45557	.45581	.45605	.45630	.45654	.45679
.75	.45703	.45728	.45752	.45776	.45801	.45825	.45850	.45874	.45898	.45923	.45947	.45972	.45996	.46021	.46045	.46069
.76	.46094	.46118	.46143	.46167	.46191	.46216	.46240	.46265	.46289	.46313	.46338	.46362	.46387	.46411	.46436	.46460
.77	.46484	.46509	.46533	.46558	.46582	.46606	.46631	.46655	.46680	.46704	.46729	.46753	.46777	.46802	.46826	.46851
.78	.46875	.46899	.46924	.46948	.46973	.46997	.47021	.47046	.47070	.47095	.47119	.47144	.47168	.47192	.47217	.47241
.79	.47266	.47290	.47314	.47339	.47363	.47388	.47412	.47437	.47461	.47485	.47510	.47534	.47559	.47583	.47607	.47632
.7A	.47656	.47681	.47705	.47729	.47754	.47778	.47803	.47827	.47852	.47876	.47900	.47925	.47949	.47974	.47998	.48022
.7B	.48047	.48071	.48096	.48120	.48145	.48169	.48193	.48218	.48242	.48267	.48291	.48315	.48340	.48364	.48389	.48413
.7C	.48437	.48462	.48486	.48511	.48535	.48560	.48584	.48608	.48633	.48657	.48682	.48706	.48730	.48755	.48779	.48804
.7D	.48828	.48853	.48877	.48901	.48926	.48950	.48975	.48999	.49023	.49048	.49072	.49097	.49121	.49146	.49170	.49194
.7E	.49219	.49243	.49268	.49292	.49316	.49341	.49365	.49390	.49414	.49438	.49463	.49487	.49512	.49536	.49561	.49585
.7F	.49609	.49634	.49658	.49683	.49707	.49731	.49756	.49780	.49805	.49829	.49854	.49878	.49902	.49927	.49951	.49976

APPENDIX E

HEXADECIMAL-TO-DECIMAL FRACTION TABLE (Cont)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
.80	.50000	.50024	.50049	.50073	.50098	.50122	.50146	.50171	.50195	.50220	.50244	.50269	.50293	.50317	.50342	.50366
.81	.50391	.50415	.50439	.50464	.50488	.50513	.50537	.50562	.50586	.50610	.50635	.50659	.50684	.50708	.50732	.50757
.82	.50781	.50806	.50830	.50854	.50879	.50903	.50928	.50952	.50977	.51001	.51025	.51050	.51074	.51099	.51123	.51147
.83	.51172	.51196	.51221	.51245	.51270	.51294	.51318	.51343	.51367	.51392	.51416	.51440	.51465	.51489	.51514	.51538
.84	.51562	.51587	.51611	.51636	.51660	.51685	.51709	.51733	.51758	.51782	.51807	.51831	.51855	.51880	.51904	.51929
.85	.51953	.51978	.52002	.52026	.52051	.52075	.52100	.52124	.52148	.52173	.52197	.52222	.52246	.52271	.52295	.52319
.86	.52344	.52368	.52393	.52417	.52441	.52466	.52490	.52515	.52539	.52563	.52588	.52612	.52637	.52661	.52686	.52710
.87	.52734	.52759	.52783	.52808	.52832	.52856	.52881	.52905	.52930	.52954	.52979	.53003	.53027	.53052	.53076	.53101
.88	.53125	.53149	.53174	.53198	.53223	.53247	.53271	.53296	.53320	.53345	.53369	.53394	.53418	.53442	.53467	.53491
.89	.53516	.53540	.53564	.53589	.53613	.53638	.53662	.53687	.53711	.53735	.53760	.53784	.53809	.53833	.53857	.53882
.8A	.53906	.53931	.53955	.53979	.54004	.54028	.54053	.54077	.54101	.54126	.54150	.54175	.54199	.54224	.54248	.54272
.8B	.54297	.54321	.54346	.54370	.54395	.54419	.54443	.54468	.54492	.54517	.54541	.54565	.54590	.54614	.54639	.54663
.8C	.54687	.54712	.54736	.54761	.54785	.54810	.54834	.54858	.54883	.54907	.54932	.54956	.54980	.55005	.55029	.55054
.8D	.55078	.55103	.55127	.55151	.55176	.55200	.55225	.55249	.55273	.55298	.55322	.55347	.55371	.55396	.55420	.55444
.8E	.55469	.55493	.55518	.55542	.55566	.55591	.55615	.55640	.55664	.55688	.55713	.55737	.55762	.55786	.55811	.55835
.8F	.55859	.55884	.55908	.55933	.55957	.55981	.56006	.56030	.56055	.56079	.56104	.56128	.56152	.56177	.56201	.56226
.90	.56250	.56274	.56299	.56323	.56348	.56372	.56396	.56421	.56445	.56470	.56494	.56519	.56543	.56567	.56592	.56616
.91	.56641	.56665	.56690	.56714	.56738	.56763	.56787	.56812	.56836	.56860	.56885	.56909	.56934	.56958	.56982	.57007
.92	.57031	.57055	.57080	.57104	.57129	.57153	.57177	.57202	.57226	.57251	.57275	.57300	.57324	.57348	.57373	.57397
.93	.57422	.57446	.57471	.57495	.57520	.57544	.57568	.57593	.57617	.57642	.57666	.57690	.57715	.57739	.57764	.57788
.94	.57812	.57837	.57861	.57886	.57910	.57935	.57959	.57983	.58008	.58032	.58057	.58081	.58105	.58130	.58154	.58179
.95	.58203	.58228	.58252	.58276	.58301	.58325	.58350	.58374	.58398	.58423	.58447	.58472	.58496	.58521	.58545	.58569
.96	.58594	.58618	.58643	.58667	.58691	.58716	.58740	.58765	.58789	.58813	.58838	.58862	.58887	.58911	.58936	.58960
.97	.58984	.59009	.59033	.59058	.59082	.59106	.59131	.59155	.59180	.59204	.59229	.59253	.59277	.59302	.59326	.59351
.98	.59375	.59399	.59424	.59448	.59473	.59497	.59521	.59546	.59570	.59595	.59619	.59644	.59668	.59692	.59717	.59741
.99	.59766	.59790	.59814	.59839	.59863	.59888	.59912	.59937	.59961	.59985	.60010	.60034	.60059	.60083	.60107	.60132
.9A	.60156	.60181	.60205	.60229	.60254	.60278	.60303	.60327	.60351	.60376	.60400	.60425	.60449	.60474	.60498	.60522
.9B	.60547	.60571	.60596	.60620	.60644	.60669	.60693	.60718	.60742	.60767	.60791	.60815	.60840	.60864	.60889	.60913
.9C	.60937	.60962	.60986	.61011	.61035	.61060	.61084	.61108	.61133	.61157	.61182	.61206	.61230	.61255	.61279	.61304
.9D	.61328	.61353	.61377	.61401	.61426	.61450	.61475	.61499	.61523	.61548	.61572	.61597	.61621	.61646	.61670	.61694
.9E	.61719	.61743	.61768	.61792	.61816	.61841	.61865	.61890	.61914	.61938	.61963	.61987	.62012	.62036	.62061	.62085
.9F	.62109	.62134	.62158	.62183	.62207	.62231	.62256	.62280	.62305	.62329	.62354	.62378	.62402	.62427	.62451	.62476
.A0	.62500	.62524	.62549	.62573	.62598	.62622	.62646	.62671	.62695	.62720	.62744	.62769	.62793	.62817	.62842	.62866
.A1	.62891	.62915	.62939	.62964	.62988	.63013	.63037	.63062	.63086	.63110	.63135	.63159	.63184	.63208	.63232	.63257
.A2	.63281	.63306	.63330	.63354	.63379	.63403	.63428	.63452	.63477	.63501	.63525	.63550	.63574	.63599	.63623	.63647
.A3	.63672	.63696	.63721	.63745	.63770	.63794	.63818	.63843	.63867	.63892	.63916	.63940	.63965	.63989	.64014	.64038
.A4	.64062	.64087	.64111	.64136	.64160	.64185	.64209	.64233	.64258	.64282	.64307	.64331	.64355	.64380	.64404	.64429
.A5	.64453	.64478	.64502	.64526	.64551	.64575	.64600	.64624	.64648	.64673	.64697	.64722	.64746	.64771	.64795	.64819
.A6	.64844	.64868	.64893	.64917	.64941	.64966	.64990	.65015	.65039	.65063	.65088	.65112	.65137	.65161	.65186	.65210
.A7	.65234	.65259	.65283	.65308	.65332	.65356	.65381	.65405	.65430	.65454	.65479	.65503	.65527	.65552	.65576	.65601
.A8	.65625	.65649	.65674	.65698	.65723	.65747	.65771	.65796	.65820	.65845	.65869	.65894	.65918	.65942	.65967	.65991
.A9	.66016	.66040	.66065	.66089	.66113	.66138	.66162	.66187	.66211	.66235	.66260	.66284	.66309	.66333	.66357	.66382
.AA	.66406	.66431	.66455	.66479	.66504	.66528	.66553	.66577	.66602	.66626	.66650	.66675	.66699	.66724	.66748	.66772
.AB	.66797	.66821	.66846	.66870	.66895	.66919	.66943	.66968	.66992	.67017	.67041	.67065	.67090	.67114	.67139	.67163
.AC	.67187	.67212	.67236	.67261	.67285	.67310	.67334	.67358	.67383	.67407	.67432	.67456	.67480	.67505	.67529	.67554
.AD	.67578	.67603	.67627	.67651	.67676	.67700	.67725	.67749	.67773	.67798	.67822	.67847	.67871	.67896	.67920	.67944
.AE	.67969	.67993	.68018	.68042	.68066	.68091	.68115	.68140	.68164	.68188	.68213	.68237	.68262	.68286	.68311	.68335
.AF	.68359	.68384	.68408	.68433	.68457	.68481	.68506	.68530	.68555	.68579	.68604	.68628	.68652	.68677	.68701	.68726
.B0	.68750	.68774	.68799	.68823	.68848	.68872	.68896	.68921	.68945	.68970	.68994	.69019	.69043	.69067	.69092	.69116
.B1	.69141	.69165	.69189	.69214	.69238	.69263	.69287	.69312	.69336	.69360	.69385	.69409	.69434	.69458	.69482	.69507
.B2	.69531	.69555	.69580	.69604	.69629	.69653	.69678	.69702	.69727	.69751	.69775	.69800	.69824	.69849	.69873	.69897
.B3	.69922	.69946	.69971	.69995	.70020	.70044	.70068	.70093	.70117	.70142	.70166	.70190	.70215	.70239	.70264	.70288
.B4	.70312	.70337	.70361	.70386	.70410	.70435	.70459	.70483	.70508	.70532	.70557	.70581	.70605	.70630	.70654	.70679
.B5	.70703	.70728	.70752	.70776	.70801	.70825	.70850	.70874	.70899	.70923	.70948	.70972	.70996	.71021	.71045	.71070
.B6	.71094	.71118	.71143	.71167	.71191	.71216	.71240	.71265	.71289	.71313	.71338	.71362	.71387	.71411	.71436	.71460
.B7	.71484	.71509	.71533	.71558	.71582	.71606	.71631	.71655	.71680	.71704	.71729	.71753	.71777	.71802	.71826	.71851
.B8	.71875	.71899	.71924	.71948	.71973	.71997	.72021	.72046	.72070	.72095	.72119	.72144	.72168	.72192	.72217	.72241
.B9	.72266	.72290	.72314	.72339	.72363	.72388	.72412	.72437	.72461	.72485	.72510	.72534	.72559	.72583	.72607	.72632
.BA	.72656	.72681	.72705	.72729	.72754	.72778	.72803	.72827	.72852	.72876	.72900	.72925	.72949	.72974	.72998	.73022
.BB	.73047	.73071	.73096	.73120	.73144	.73169	.73193	.73218	.73242	.73267	.73291	.73315	.73340	.73364	.73389	.73413
.BC	.73437	.73462	.73486	.73511	.73535	.73560	.73584	.73608	.73633	.73657	.73682	.73706	.73730	.73755	.73779	.73804
.BD	.73828	.73853	.73877	.73901	.73926	.73950	.73975	.73999	.74023	.74048	.74072	.74097	.74121	.74146	.74170	.74194
.BE	.74219	.74243	.74268	.74292	.74316	.74341	.74365	.74390	.74414	.74438	.74463	.74487	.74512	.74536	.74561	.74585
.BF	.74609	.74634	.74658	.74683	.74707	.74731	.74756	.74780	.74805	.74829	.74854	.74878	.74902	.74927	.74951	.74976

APPENDIX E

HEXADECIMAL-TO-DECIMAL FRACTION TABLE (Cont)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
.C0	.75000	.75024	.75049	.75073	.75098	.75122	.75146	.75171	.75195	.75220	.75244	.75269	.75293	.75317	.75342	.75366
.C1	.75391	.75415	.75439	.75464	.75488	.75513	.75537	.75562	.75586	.75610	.75635	.75659	.75684	.75708	.75732	.75757
.C2	.75781	.75806	.75830	.75854	.75879	.75903	.75928	.75952	.75977	.76001	.76025	.76050	.76074	.76099	.76123	.76147
.C3	.76172	.76196	.76221	.76245	.76270	.76294	.76318	.76343	.76367	.76392	.76416	.76440	.76465	.76489	.76514	.76538
.C4	.76562	.76587	.76611	.76636	.76660	.76685	.76709	.76733	.76758	.76782	.76807	.76831	.76855	.76880	.76904	.76929
.C5	.76953	.76978	.77002	.77026	.77051	.77075	.77100	.77124	.77148	.77173	.77197	.77222	.77246	.77271	.77295	.77319
.C6	.77344	.77368	.77393	.77417	.77441	.77466	.77490	.77515	.77539	.77563	.77588	.77612	.77637	.77661	.77686	.77710
.C7	.77734	.77759	.77783	.77808	.77832	.77856	.77881	.77905	.77930	.77954	.77979	.78003	.78027	.78052	.78076	.78101
.C8	.78125	.78149	.78174	.78198	.78223	.78247	.78271	.78296	.78320	.78345	.78369	.78394	.78418	.78442	.78467	.78491
.C9	.78516	.78540	.78564	.78589	.78613	.78638	.78662	.78687	.78711	.78735	.78760	.78784	.78809	.78833	.78857	.78882
.CA	.78906	.78931	.78955	.78979	.79004	.79028	.79053	.79077	.79102	.79126	.79150	.79175	.79199	.79224	.79248	.79272
.CB	.79297	.79321	.79346	.79370	.79395	.79419	.79443	.79468	.79492	.79517	.79541	.79565	.79590	.79614	.79639	.79663
.CC	.79687	.79712	.79736	.79761	.79785	.79810	.79834	.79858	.79883	.79907	.79932	.79956	.79980	.80005	.80029	.80054
.CD	.80078	.80103	.80127	.80151	.80176	.80200	.80225	.80249	.80273	.80298	.80322	.80347	.80371	.80395	.80420	.80444
.CE	.80469	.80493	.80518	.80542	.80566	.80591	.80615	.80640	.80664	.80688	.80713	.80737	.80762	.80786	.80811	.80835
.CF	.80859	.80884	.80908	.80933	.80957	.80981	.81006	.81030	.81055	.81079	.81104	.81128	.81152	.81177	.81201	.81226
.D0	.81250	.81274	.81299	.81323	.81348	.81372	.81396	.81421	.81445	.81470	.81494	.81519	.81543	.81567	.81592	.81616
.D1	.81641	.81665	.81689	.81714	.81738	.81763	.81787	.81812	.81836	.81860	.81885	.81909	.81934	.81958	.81982	.82007
.D2	.82031	.82056	.82080	.82104	.82129	.82153	.82178	.82202	.82227	.82251	.82275	.82300	.82324	.82349	.82373	.82397
.D3	.82422	.82446	.82471	.82495	.82520	.82544	.82568	.82593	.82617	.82642	.82666	.82690	.82715	.82739	.82764	.82788
.D4	.82812	.82837	.82861	.82886	.82910	.82934	.82959	.82983	.83008	.83032	.83057	.83081	.83105	.83130	.83154	.83179
.D5	.83203	.83228	.83252	.83276	.83301	.83325	.83350	.83374	.83398	.83423	.83447	.83472	.83496	.83521	.83545	.83569
.D6	.83594	.83618	.83643	.83667	.83691	.83716	.83740	.83765	.83789	.83813	.83838	.83862	.83887	.83911	.83936	.83960
.D7	.83984	.84009	.84033	.84058	.84082	.84106	.84131	.84155	.84180	.84204	.84229	.84253	.84277	.84302	.84326	.84351
.D8	.84375	.84399	.84424	.84448	.84473	.84497	.84521	.84546	.84570	.84595	.84619	.84644	.84668	.84692	.84717	.84741
.D9	.84766	.84790	.84814	.84839	.84863	.84888	.84912	.84937	.84961	.84985	.85010	.85034	.85059	.85083	.85107	.85132
.DA	.85156	.85181	.85205	.85229	.85254	.85278	.85303	.85327	.85351	.85376	.85400	.85425	.85449	.85473	.85498	.85522
.DB	.85547	.85571	.85596	.85620	.85644	.85669	.85693	.85718	.85742	.85767	.85791	.85815	.85840	.85864	.85889	.85913
.DC	.85937	.85962	.85986	.86011	.86035	.86060	.86084	.86108	.86133	.86157	.86182	.86206	.86230	.86255	.86279	.86304
.DD	.86328	.86353	.86377	.86401	.86426	.86450	.86475	.86499	.86523	.86548	.86572	.86597	.86621	.86645	.86670	.86694
.DE	.86719	.86743	.86768	.86792	.86816	.86841	.86865	.86890	.86914	.86938	.86963	.86987	.87011	.87036	.87060	.87085
.DF	.87109	.87134	.87158	.87183	.87207	.87231	.87256	.87280	.87305	.87329	.87354	.87378	.87402	.87427	.87451	.87476
.E0	.87500	.87524	.87549	.87573	.87598	.87622	.87646	.87671	.87695	.87720	.87744	.87769	.87793	.87817	.87842	.87866
.E1	.87891	.87915	.87939	.87964	.87988	.88013	.88037	.88062	.88086	.88110	.88135	.88159	.88184	.88208	.88232	.88257
.E2	.88281	.88306	.88330	.88354	.88379	.88403	.88428	.88452	.88477	.88501	.88525	.88550	.88574	.88599	.88623	.88647
.E3	.88672	.88696	.88721	.88745	.88770	.88794	.88818	.88843	.88867	.88892	.88916	.88940	.88965	.88989	.89014	.89038
.E4	.89062	.89087	.89111	.89136	.89160	.89185	.89209	.89233	.89258	.89282	.89307	.89331	.89355	.89380	.89404	.89429
.E5	.89453	.89478	.89502	.89526	.89551	.89575	.89600	.89624	.89648	.89673	.89697	.89722	.89746	.89771	.89795	.89819
.E6	.89844	.89868	.89893	.89917	.89941	.89966	.89990	.90015	.90039	.90063	.90088	.90112	.90137	.90161	.90186	.90210
.E7	.90234	.90259	.90283	.90308	.90332	.90356	.90381	.90405	.90430	.90454	.90479	.90503	.90527	.90552	.90576	.90601
.E8	.90625	.90649	.90674	.90698	.90723	.90747	.90771	.90796	.90820	.90845	.90869	.90894	.90918	.90942	.90967	.90991
.E9	.91016	.91040	.91064	.91089	.91113	.91138	.91162	.91187	.91211	.91235	.91260	.91284	.91309	.91333	.91357	.91382
.EA	.91406	.91431	.91455	.91479	.91504	.91528	.91553	.91577	.91602	.91626	.91650	.91675	.91699	.91724	.91748	.91772
.EB	.91797	.91821	.91846	.91870	.91895	.91919	.91943	.91968	.91992	.92017	.92041	.92065	.92090	.92114	.92139	.92163
.EC	.92187	.92212	.92236	.92261	.92285	.92310	.92334	.92358	.92383	.92407	.92432	.92456	.92480	.92505	.92529	.92554
.ED	.92578	.92603	.92627	.92651	.92676	.92700	.92725	.92749	.92773	.92798	.92822	.92847	.92871	.92896	.92920	.92944
.EE	.92969	.92993	.93018	.93042	.93066	.93091	.93115	.93140	.93164	.93188	.93213	.93237	.93262	.93286	.93311	.93335
.EF	.93359	.93384	.93408	.93433	.93457	.93481	.93506	.93530	.93555	.93579	.93604	.93628	.93652	.93677	.93701	.93726
.F0	.93750	.93774	.93799	.93823	.93848	.93872	.93896	.93921	.93945	.93970	.93994	.94019	.94043	.94067	.94092	.94116
.F1	.94141	.94165	.94189	.94214	.94238	.94263	.94287	.94312	.94336	.94360	.94385	.94409	.94434	.94458	.94482	.94507
.F2	.94531	.94556	.94580	.94604	.94629	.94653	.94678	.94702	.94727	.94751	.94775	.94800	.94824	.94849	.94873	.94897
.F3	.94922	.94946	.94971	.94995	.95020	.95044	.95068	.95093	.95117	.95142	.95166	.95190	.95215	.95239	.95264	.95288
.F4	.95312	.95337	.95361	.95386	.95410	.95435	.95459	.95483	.95508	.95532	.95557	.95581	.95605	.95630	.95654	.95679
.F5	.95703	.95728	.95752	.95776	.95801	.95825	.95850	.95874	.95898	.95923	.95947	.95972	.95996	.96021	.96045	.96069
.F6	.96094	.96118	.96143	.96167	.96191	.96216	.96240	.96265	.96289	.96313	.96338	.96362	.96387	.96411	.96436	.96460
.F7	.96484	.96509	.96533	.96558	.96582	.96606	.96631	.96655	.96680	.96704	.96729	.96753	.96777	.96802	.96826	.96851
.F8	.96875	.96899	.96924	.96948	.96973	.96997	.97021	.97046	.97070	.97095	.97119	.97144	.97168	.97192	.97217	.97241
.F9	.97266	.97290	.97314	.97339	.97363	.97388	.97412	.97437	.97461	.97485	.97510	.97534	.97559	.97583	.97607	.97632
.FA	.97656	.97681	.97705	.97729	.97754	.97778	.97803	.97827	.97852	.97876	.97900	.97925	.97949	.97974	.97998	.98022
.FB	.98047	.98071	.98096	.98120	.98144	.98169	.98193	.98218	.98242	.98266	.98291	.98315	.98340	.98364	.98389	.98413
.FC	.98437	.98462	.98486	.98511	.98535	.98560	.98584	.98608	.98633	.98657	.98682	.98706	.98730	.98755	.98779	.98804
.FD	.98828	.98853	.98877	.98901	.98926	.98950	.98975	.98999	.99023	.99048	.99072	.99097	.99121	.99146	.99170	.99194
.FE	.99219	.99243	.99268	.99292	.99316	.99341	.99365	.99390	.99414	.99438	.99463	.99487	.99512	.99536	.99561	.99585
.FF	.99609	.99634	.99658	.99683	.99707	.99731	.99756	.99780	.99805	.99829	.99854	.99878	.99902	.99927	.99951	.99976

APPENDIX F

TWO'S COMPLEMENT ARITHMETIC

The Raytheon 706 Computer systems hold numbers internally in *two's complement form*. Two's complement form* is defined to mean:

All fixed-point positive numbers must have a binary zero for the sign bit followed by the normal binary representation of the magnitude portion of the number. All internal fixed-point negative numbers consist of a binary one for the sign bit followed by the two's complement of the magnitude portion of the equivalent positive number.

Rules for Putting all Binary Numbers into Two's Complement Form

Positive Number: Retain the sign bit as a binary "0" and retain the magnitude portion of the number as is.

Negative Number: Retain the sign bit as a binary "1" and replace the magnitude portion of the number with the two's complement of itself.

The two's complement of any binary negative number can be calculated by subtracting the number from all binary ones and then adding one. For example:

Original Number	Complement Form	Comment
+1010	+1010	No Calculation necessary
- 1010	- 0110	Minus sign retained;

$$\begin{array}{r} 1111 \\ - 1010 \\ \hline 0101 \\ + \quad 1 \\ \hline 0110 \end{array}$$

*Frequent references are made in this manual to: "all numbers are in two's complement form." This does *not* mean that positive numbers have had the two's complement operation applied to their mantissa. The word "form" as used in this sense means that both positive and negative numbers are represented as stated in the definition."

APPENDIX F

TWO'S COMPLEMENT ARITHMETIC (Cont)

The reason for using two's complement arithmetic is that the additive and subtractive operations are simplified in that the sign of the result is always correct without special manipulation of the numbers. Example:

Add the following numbers:

Binary	Decimal	
+1010	+10	}
- 1010	- 10	
<u>+0010</u>	<u>+2</u>	

Converting to two's complement form and adding: (the signs will be shown on binary digits)

Binary	Decimal	
01010	+10	}
10110	- 10	
<u>00010</u>	<u>+2</u>	
1 00010	+2	

↑
(throw away carry)

If the carry is ignored the answer of +2 is correct.

Recomplementing

If the final result of a computation is negative (sign bit is a one), then the external representation of the number can be obtained by again taking the two's complement of the magnitude portion of the number. Example:

+1010	}	
- 1010		external
- 0010		

APPENDIX F

TWO'S COMPLEMENT ARITHMETIC (Cont)

Putting into two's complement form and adding:

$$\begin{array}{r} 01010 \\ 10110 \\ 11110 \\ \hline 1\ 11110 \\ \text{(throw away carry)} \end{array} \left. \vphantom{\begin{array}{r} 01010 \\ 10110 \\ 11110 \\ 1\ 11110 \end{array}} \right\} \text{internal}$$

Recomplementing the magnitude portion of the result while retaining the minus sign:

$$\begin{array}{r} 1111 \\ - 1110 \\ \hline 0001 \\ + \quad 1 \\ \hline 1\ 0010 \end{array} \left. \vphantom{\begin{array}{r} 1111 \\ - 1110 \\ 0001 \\ + \quad 1 \\ 1\ 0010 \end{array}} \right\} \text{external}$$

The correct external number of (-2) results.

Action of the Assembler

When negative decimal numbers are encountered by the assembler, they are automatically converted to two's complement form by the assembler for internal processing. Hexadecimal numbers specified by (X'---') must be put into two's complement form by the programmer.

APPENDIX G
CHARACTER CODES

**ASCII (TELETYPEWRITER) CHARACTER CODES,
HEXADECIMAL AND HOLLERITH PUNCH EQUIVALENTS**

Character	ASCII (HEX)	Hollerith Punch	Character	ASCII (HEX)	Hollerith Punch	Character	ASCII (HEX)	Hollerith Punch	Character	ASCII (HEX)
@	C0	0-2-8	X	D8	0-7	0	B0	0	FE ₀	88
A	C1	12-1	Y	D9	0-8	1	B1	1	TAB	87
B	C2	12-2	Z	DA	0-9	2	B2	2	LF	8A
C	C3	12-3	[DB	12-5-8	3	B3	3	VT	8B
D	C4	12-4	\	DC	0-6-8	4	B4	4	FORM	8C
E	C5	12-5]	DD	11-5-8	5	B5	5	RETURN	8D
F	C6	12-6	↑	DE	7-8	6	B6	6	SO	8E
G	C7	12-7	←	DF	2-8	7	B7	7	SI	8F
H	C8	12-8	Blank	A0	Blank	8	B8	8	DC ₀	90
I	C9	12-9		A1	11-2-8	9	B9	9	X ON	91
J	CA	11-1	"	A2	0-5-8	:	BA	5-8	R1 ON	92
K	CB	11-2	#	A3	0-7-8	;	BB	11-6-8	X OFF	93
L	CC	11-3	\$	A4	11-3-8	<	BC	12-6-8	R1 OFF	94
M	CD	11-4	%	A5	11-7-8	=	BD	3-8	ERR	95
N	CE	11-5	&	A6	12-7-8	>	BE	6-8	SYNC	96
O	CF	11-6	'	A7	4-8	?	BF	12-2-8	LEM	97
P	D0	11-7	(A8	0-4-8	NULL	80		S ₀	98
Q	D1	11-8)	A9	12-4-8	SOM	81		•	
R	D2	11-9	*	AA	11-4-8	EOA	82		•	
S	D3	0-2	+	AB	12	EOM	83		•	
T	D4	0-3	,	AC	0-3-8	EOT	84		•	
U	D5	0-4	-	AD	11	WRU	85		•	
V	D6	0-5	.	AE	12-3-8	RU	86		•	
W	D7	0-6	/	AF	0-1	BELL	87		S ₇	9F

\ Reverse slant ↑ Up Arrow (Exponentiation) ← Left Arrow (Implies/Replaced by) SP Space NULL Blank SOM Start of message EOA End of address EOM End of message EOT End of transmission WRU Who are you? RU Are you ? BELL Audible signal FE ₀ Format effector TAB Horizontal tabulation LF Line feed VT Vertical tabulation FORM Form feed	RETURN C Carriage return SO Shift out SI Shift in DC ₀ Device control (reserved for data link escape) X ON Transmitter on R1 ON Receiver on (TAPE) X OFF Transmitter off R1 OFF Receiver off (TAPE) ERR Error SYNC Synchronizing character LEM Logical end of medium S _n Information separators ACK Acknowledge ALT MODE Escape (for communicators) ESC Escape (for data processing) RUB OUT Delete
--	--

APPENDIX G

CHARACTER CODES (Cont)

ASCII-8 (IBM) CHARACTER CODES AND HEXADECIMAL EQUIVALENTS

Char.	Hex.	Char.	Hex.	Char.	Hex.	Char.	Hex.
	A0	blank	40	NULL	00		E0
A	A1	!	41	SOM	01	a	E1
B	A2	"	42	EOA	02	b	E2
C	A3	#	43	EOM	03	c	E3
D	A4	\$	44	EOT	04	d	E4
E	A5	%	45	WRU	05	e	E5
F	A6	&	46	RU	06	f	E6
G	A7	'	47	BELL	07	g	E7
H	A8	(48	BKSP	08	h	E8
I	A9)	49	HT	09	i	E9
J	AA	*	4A	LF	0A	j	EA
K	AB	+	4B	VT	0B	k	EB
L	AC	,	4C	FF	0C	l	EC
M	AD	-	4D	CR	0D	m	ED
N	AE	.	4E	SO	0E	n	EE
O	AF	/	4F	SI	0F	o	EF
P	B0	0	50	DC ₀	10	p	F0
Q	B1	1	51	DC ₁	11	q	F1
R	B2	2	52	DC ₂	12	r	F2
W	B3	3	53	DC ₃	13	s	F3
T	B4	4	54	DC ₄	14	t	F4
U	B5	5	55	ERR	15	u	F5
V	B6	6	56	SYNC	16	v	F6
W	B7	7	57	LEM	17	w	F7
X	B8	8	58	S ₀	18	x	F8
Y	B9	9	59	S ₁	19	y	F9
Z	BA	:	5A	S ₂	1A	z	FA
[BB	;	5B	S ₃	1B		FB
\	BC	<	5C	S ₄	1C		FC
]	BD	=	5D	S ₅	1D		FD
↑	BE	>	5E	S ₆	1E	ESC	FE
←	BF	?	5F	S ₇	1F	DEL	FF

LINE PRINTER CHARACTER SET

Char.	Hex. Code	Char.	Hex. Code	Char.	Hex. Code	Char.	Hex. Code
@	C0	P	D0	Blank	A0	0	B0
A	C1	Q	D1	!	A1	1	B1
B	C2	R	D2	"	A2	2	B2
C	C3	S	D3	#	A3	3	B3
D	C4	T	D4	\$	A4	4	B4
E	C5	U	D5	%	A5	5	B5
F	C6	V	D6	&	A6	6	B6
G	C7	W	D7	'	A7	7	B7
H	C8	X	D8	(A8	8	B8
I	C9	Y	D9)	A9	9	B9
J	CA	Z	DA	*	AA	:	BA
K	CB	[DB	+	AB	;	BB
L	CC	\	DC	,	AC	<	BC
M	CD]	DD	-	AD	=	BD
N	CE	↑	DE	.	AE	>	BE
O	CF	←	DF	/	AF	?	BF

Note: The line printer interprets only the six least significant bits of the hexadecimal code.

APPENDIX H
IMPLEMENTATION AND INSTALLATION INFORMATION

PHYSICAL CHARACTERISTICS

Model Number	Description	Dimensions (Inches)			Weight Pounds	Comments
		Height	Width	Depth		
70610	Central Processor (CPU) Model 706, with Display and Control panel.	5.25	19.0	26.0	25	Includes 5.5 inches of depth allowance for connectors and cables.
	4096 Word Memory Module.	7.00	19.0	20.75	40	Includes 5.5 inches of depth allowance for connectors and cables.
	Power Supply.	5.25	19.0	25.0	55	Includes 5.0 inches of depth allowance for connectors and cables
	Blower Pack.	5.25	19.0	6.5	10	
	AC Controller.	5.25	19.0	6.25	10	No front panel space required.
	ASR 33.	33.0	21.5	19.0	56	Stand alone unit.
	Cabinet.	52.0	25.0	30.0	175	22.75 inches of front panel space are available for addition of options. Power for 70610 System: 115 Vac, 60 Hz, 1 ϕ , 10.0 amps
70611	Central Processor (CPU) Model 706, with Display and Control Panel.	5.25	19.0	26.0	25	Includes 5.5 inches of depth allowance for connectors and cables.
	4096 Word Memory Module	7.00	19.0	20.75	40	Includes 5.5 inches of depth allowance for connectors and cables.
	Power Supply	5.25	19.0	25.0	55	Includes 5.0 inches of depth allowance for connectors and cables.
	Blower Pack	5.25	19.0	6.5	10	
	AC Controller	5.25	19.0	6.25	10	No front panel space required.

APPENDIX H

IMPLEMENTATION AND INSTALLATION INFORMATION (Cont)

PHYSICAL CHARACTERISTICS (Cont)

Model Number	Description	Dimensions (Inches)			Weight Pounds	Comments
		Height	Width	Depth		
70630	ASR 35	41.5	41.5	24.5	225	Stand alone unit.
	Cabinet	52.0	25.0	30.0	175	22.75 inches of front panel space are available for addition of options. Power for 70611 System: 115 Vac, 60 Hz, 1 ϕ , 11.5 amps.
	Central Processor (CPU) Model 706 with Display and Control Panel.	5.25	19.0	26.0	25	Includes 5.5 inches of depth allowance for connectors and cables.
	16,384 Word Memory	52.0	25.0	30.0	320	The 16,384 word memory module is housed in a separate cabinet with power supply. This cabinet provides space and power for a second 16,384 word memory module.
	Power Supply	5.25	19.0	25.0	55	Includes 5.0 inches of depth allowance for connectors and cables.
	Blower Pack	5.25	19.0	6.5	10	
	AC Controller	5.25	19.0	6.25	10	No front panel space required.
70631	ASR 33	33.0	21.5	19.0	56	Stand alone unit.
	Cabinet	52.0	25.0	30.0	175	29.75 inches of front panel space are available for addition of options. Power for 70630 System: 115 Vac, 60 Hz, 1 ϕ , 10 amps.
	Central Processor (CPU) Model 706 with Display and Control Panel	5.25	19.0	26.0	25	Includes 5.5 inches of depth allowance for connectors and cables.

APPENDIX H

IMPLEMENTATION AND INSTALLATION INFORMATION (Cont)

PHYSICAL CHARACTERISTICS (Cont)

Model Number	Description	Dimensions (Inches)			Weight Pounds	Comments
		Height	Width	Depth		
	16,384 Word Memory	52.0	25.0	30.0	320	The 16,384 word memory module is housed in a separate cabinet with power supply. This cabinet provides space and power for a second 16,384 word memory module.
	Power Supply	5.25	19.0	25.0	55	Includes 5.0 inches of depth allowance for connectors and cables.
	Blower Pack	5.25	19.0	6.5	10	
	AC Controller	5.25	19.0	6.25	10	No front panel space required.
	ASR35	41.5	41.5	24.5	225	Stand alone unit.
	Cabinet	52.0	25.0	30.0	175	29.75 inches of front panel space are available for addition of options. Power for 70631 System: 115 Vac, 60 Hz, 1 ϕ , 11.5 amps.
	Memory Expansion Options					
71601	Memory Expansion, 4096 Words	7.00	19.0	20.75	40	Includes 5.5 inches of depth allowance for connectors and cables.
71602	Memory Expansion, 8192 Words	7.00	19.0	20.75	41	Includes 5.5 inches of depth allowance for connectors and cables.
71603	Memory Expansion, 16,384 Words				115	For addition to 706 with 16,384 word memory system (70630 or 70631). Does not require additional memory cabinet.
	a) Module	39.5	4.25	25.75	60	Power for 71603 Expansion: 115 Vac, 60 Hz, 1 ϕ , 5 amps.
	b) Power Supply	5.25	19.0	25.0	55	

APPENDIX H

IMPLEMENTATION AND INSTALLATION INFORMATION (Cont)

PHYSICAL CHARACTERISTICS (Cont)

Model Number	Description	Dimensions (Inches)			Weight Pounds	Comments
		Height	Width	Depth		
71604	Memory System, 16,384 Words				320	
	a) Cabinet	52.0	25.0	30.0	260	Provides space for two 16,384 word memory modules and power supply.
	b) 16,384 word memory module	39.5	4.25	25.75	60	Includes 10.0 inches of depth allowance for connectors and cables. Uses CPU power.
	Central Processor Options					
72601	Direct Memory Access (DMA).	--	--	--	--	Housed in Central Processor drawer.
72602	High-Speed Multiply and Divide	--	--	--	--	Housed in Central Processor drawer.
72603	Power Fail Safe	5.25	19.0	2.75	5	No Front Panel Space required.
72604	Priority Interrupt Expansion to 4 levels.	--	--	--	--	Housed in Central Processor drawer.
72605	Priority Interrupt Expansion from 4 to 8 levels.	--	--	--	--	Housed in Central Processor drawer.
72606	Priority Interrupt Expansion from 8 to 12 levels.	--	--	--	--	Housed in Central Processor drawer.
72607	Priority Interrupt Expansion from 12 to 16 levels.	--	--	--	--	Housed in Central Processor drawer.
72608	Memory Parity Option	--	--	--	--	Housed in Central Processor drawer.
72609	Memory Protect Option	--	--	--	--	Housed in Central Processor drawer.
72610	Operator Interrupt Button	--	--	--	--	Located on CPU teletype.

APPENDIX H

IMPLEMENTATION AND INSTALLATION INFORMATION (Cont)

PHYSICAL CHARACTERISTICS (Cont)

Model Number	Description	Dimensions (Inches)			Weight Pounds	Comments
		Height	Width	Depth		
	Accessories					
72651	Optional Equipment Cabinet. Includes blower, AC outlet strip, and AC power control.	52.0	25.0	30.0	215	Provides 45.50 inches of front panel space. Requires 5.25 in. for blower.
72652	Optional Equipment Cabinet. Includes blower, AC outlet strip, and AC power control.	65.0	25.0	30.0	250	Provides 57.75 inches of front panel space. Requires 5.25 inches for blower.
	Magnetic Tape Storage: 7-Track					
73671	Magnetic Tape System.					
	a) Controller.	5.25	19.0	26.0	25	Includes 5.5 inches of depth allowance for connectors and cables.
	b) Controller Power Supply.	5.25	19.0	19.0	15	
	c) Magnetic Tape Drive - 75 ips	64.0	25.0	30.0	512	
73672	Magnetic Tape Drive - 75 ips.	64.0	25.0	30.0	512	Stand Alone Unit. Does not provide space for controller or controller power supply.
73673	Hardware Chaining Option.	--	--	--	--	Housed in Controller drawer.
73674	Magnetic Tape System.					
	a) Controller.	--	--	--	5	Housed in Peripheral Equipment Drawer*
	b) Magnetic Tape Drive - 25 ips.	12.5	19.0	21.0	60	Includes 6.0 inches of depth allowance for connectors and cables. Can be accommodated by Processor Cabinet or Optional Equipment Cabinet.
73675	Magnetic Tape Drive - 25 ips.	12.5	19.0	21.0	60	

APPENDIX H

IMPLEMENTATION AND INSTALLATION INFORMATION (Cont)

PHYSICAL CHARACTERISTICS (Cont)

Model Number	Description	Dimensions (Inches)			Weight Pounds	Comments
		Height	Width	Depth		
73691	Magnetic Tape Storage: 9-Track Magnetic Tape Controller.					
	a) Controller	5.25	19.0	26.0	25	Includes 5.5 inches of depth allowance for connectors and cables.
	b) Controller Power Supply.	5.25	19.0	19.0	15	
73692	Magnetic Tape Drive - 36 ips.	52.0	25.0	30.0	502	Stand Alone Unit. Does not provide space for controller or controller power supply.
73693	Magnetic Tape Controller.					
	a) Controller.	5.25	19.0	26.0	25	Includes 5.5 inches of depth allowance for connectors and cables.
	b) Controller Power Supply	5.25	19.0	19.0	15	
73694	Magnetic Tape Drive - 75 ips.	64.0	25.0	30.0	520	Stand Alone Unit. Does not provide space for controller or controller power supply.
73695	Hardware Chaining Option	--	--	--	--	Housed in controller drawer.
73696	Magnetic Tape System					
	a) Controller	--	--	--	5	Housed in Peripheral Equipment Drawer.*
	b) Magnetic Tape Drive - 25 ips.	12.5	19.0	21.0	60	Includes 6.0 inches of depth allowance for connectors and cables. Can be accommodated by Processor Cabinet or Optional Equipment Cabinet.
73697	Magnetic Tape Drive - 25 ips.	12.5	19.0	21.0	60	

APPENDIX H

IMPLEMENTATION AND INSTALLATION INFORMATION (Cont)

PHYSICAL CHARACTERISTICS (Cont)

Model Number	Description	Dimensions (Inches)			Weight Pounds	Comments
		Height	Width	Depth		
74602	Disc Storage					
	Disc Controller.					
	a) Controller	--	--	--	5	Housed in Peripheral Equipment Drawer.*
	b) Controller Power Supply.	5.25	19.0	16.0	30	
74602	Disc Storage Drive.	14.00	19.0	20.0	62	Includes 5.25 inches of height for disc drive power supply.
	Paper Tape Equipment					
75601	Paper Tape Reader and Controller.					
	a) Controller.	--	--	--	5	Housed in Peripheral Equipment Drawer.*
	b) Paper Tape Reader - 300 cps.	7.0	19.0	14.6	26	Includes 4.0 inches of depth allowance for connectors and cables and 2.35 inches in front of panel.
75602	Paper Tape Punch and Controller.					
	a) Controller.	--	--	--	--	Housed in Peripheral Equipment Drawer.*
	b) Paper Tape Punch - 110 cps.	15.75	19.0	16.5	28	
75603	High-Speed Paper Tape Reader spooler.	10.5	19.0	13.75	54	Depth does not include 3.0 inches in front of panel.
	Punched Card Equipment					
75611	Card Reader and Controller.					
	a) Controller.	--	--	--	5	Housed in Peripheral Equipment Drawer.*
	b) Card Reader - 1000 cpm.	35.0	35.0	32.0	250	Requires table.

APPENDIX H

IMPLEMENTATION AND INSTALLATION INFORMATION (Cont)

PHYSICAL CHARACTERISTICS (Cont)

Model Number	Description	Dimensions (Inches)			Weight Pounds	Comments
		Height	Width	Depth		
75612	Card Reader and Controller.					
	a) Controller.	--	--	--	5	Housed in Peripheral Equipment Drawer.*
	b) Card Reader - 400 cpm.	12.5	23.0	12.5	75	Requires table.
75613	Card Punch and Controller.					
	a) Controller.	--	--	--	--	Housed in Peripheral Equipment Drawer.*
	b) Card Punch - 100 to 400 cpm.	42.0	44.0	30.0	475	Stand Alone Unit.
	Line Printer					
	Line Printer and Controller.					
	a) Controller.	--	--	--	5	Housed in Peripheral Equipment Drawer.*
75621	b) Line Printer - 132 column, 245 lpm (minimum)	45.75	48.5	24.0	400	Stand Alone Unit
75622	c) Line Printer - 132 column, 360 lpm.	48.0	47.0	26.0	838	Stand Alone Unit.
75623	d) Line Printer - 132 column, 600 lpm.	48.0	47.0	26.0	838	Stand Alone Unit.
75624	e) Line Printer - 132 column, 1000 lpm.	48.0	47.0	26.0	828	Stand Alone Unit.
	Graph Plotters					
75631	Digital Plotter and Controller.					
	a) Controller.	5.25	19.0	15.75	26	Includes 2.0 inches of depth allowance for connectors and cables.
	b) Digital Plotter	9.8	18.0	14.7	50	
75632	Digital Plotter Controller.	5.25	19.0	15.75	26	Includes 2.0 inches of depth allowance for connectors and cables.

APPENDIX H

IMPLEMENTATION AND INSTALLATION INFORMATION (Cont)

PHYSICAL CHARACTERISTICS (Cont)

Model Number	Description	Dimensions (Inches)			Weight Pounds	Comments
		Height	Width	Depth		
	Teleprinters					
75641	Off-Line ASR-33 Teleprinter.	32.9	22.0	18.5	56	Stand Alone Unit. Power: 115 Vac, 60 Hz, 1 ϕ , 2.5 amps.
75642	Off-Line ASR-35 Teleprinter.	38.5	40.0	24.0	225	Stand Alone Unit. Power: 115 Vac, 60 Hz, 1 ϕ , 4.0 amps.
	Data Communications Equipment					
76601	Teletype Multiplexer					
	a) Multiplexer.	5.25	19.0	23.5	24	Includes 3.5 inches of depth allowance for connectors and cables.
	b) Power Supply.	5.25	19.0	12.0	29	Can be mounted at rear of cabinet.
76602	Teletype Multiplexer Expansion.	--	--	--	5	Housed in Teletype Multiplexer chassis (76601).
	Buffered Digital Channels					
77601						Up to four (4) Buffered Input channels can be accommodated in one chassis.
77602	Input Channels.	5.25	19.0	15.75	25	
77603						Includes 2.0 inches of depth allowance for connectors and cables.
77604						
77611						Up to four (4) Buffered Output Channels can be accommodated in one chassis.
77612	Output Channels	5.25	19.0	15.75	26	
77313						Includes 1.0 inches of depth allowance for connectors and cables.
77614						

APPENDIX H

IMPLEMENTATION AND INSTALLATION INFORMATION (Cont)

PHYSICAL CHARACTERISTICS (Cont)

Model Number	Description	Dimensions (Inches)			Weight Pounds	Comments
		Height	Width	Depth		
77615	Output Channels with Relay Contact Outputs	5.25	19.0	15.75	27	Up to four (4) Buffered Output Channels with Relay Contact Outputs can be accommodated in one chassis. Includes 2.0 inches of depth allowance for connectors and cables.
77616						
77617						
77618						
77621	Input/Output Channels.	5.25	19.0	15.75	26	Up to three (3) Buffered Input Channels and three (3) Buffered Output Channels can be accommodated in one chassis. Includes 2.0 inches of depth allowance for connectors and cables.
77622						
77623						
77624						
77625						
	Analog Input Equipment					
77631	10-bit Miniverter and Controller.	5.25	19.0	15.75	24	Up to 16 channels of Multiplexer can be accommodated in one chassis with Miniverter and Controller. Includes 2.0 inches of depth allowance for connectors and cables.
77632	12-bit Miniverter and Controller.	5.25	19.0	15.75	24	Up to 16 channels of Multiplexer can be accommodated in one chassis with Miniverter and Controller. Includes 2.0 inches of depth allowance for connectors and cables.

APPENDIX H

IMPLEMENTATION AND INSTALLATION INFORMATION (Cont)

PHYSICAL CHARACTERISTICS (Cont)

Model Number	Description	Dimensions (Inches)			Weight Pounds	Comments
		Height	Width	Depth		
77633	13-bit (BCD) Miniverter and Controller	5.25	19.0	15.75	24	Up to 16 channels of Multiplexer can be accommodated in one chassis with Miniverter and Controller. Includes 2.0 inches of depth allowance for connectors and cables.
77634	8-Channel Multiplexer Expansion.	--	--	--	--	Housed in Miniverter chassis (77631, 77632, or 77633).
77635	Miniverter Controller.	--	--	--	10	Housed in Miniverter System chassis.
77636	Multiverter Controller.	5.25	19.0	15.75	25	Includes 2.0 inches of depth allowance for connectors and cables.
Analog Output Equipment						
77641	Digital-to-Analog Converter and Controller.	5.25	19.0	15.75	22	Includes 2.0 inches of depth allowance for connectors and cables.
77642	Additional Digital-to-Analog Converters.	--	--	--	1	Housed in 77641 or 77643.
77643	Digital-to-Analog Converter and Expansion Chassis.	5.25	19.0	15.75	22	Includes 2.0 inches of depth allowance for connectors and cables.
77644	Digital-to-Analog Converter Controller.	--	--	--	1	Housed in MDAC system chassis.

APPENDIX H

IMPLEMENTATION AND INSTALLATION INFORMATION (Cont)

PHYSICAL CHARACTERISTICS (Cont)

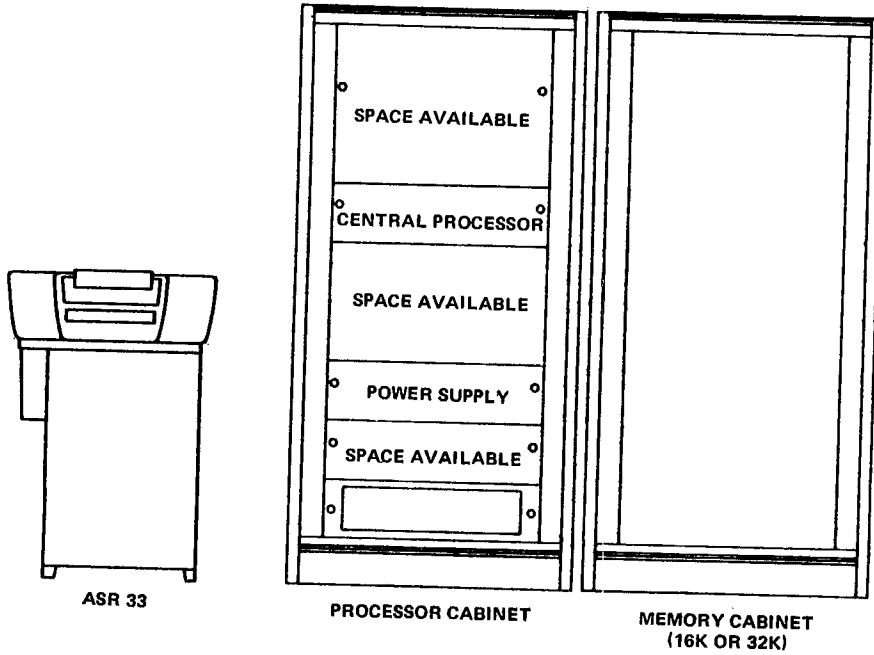
Model Number	Description	Dimensions (Inches)			Weight Pounds	Comments
		Height	Width	Depth		
77651	Clocks and Timers Time-of-Day Clock.	5.25	19.0	15.75	24	Includes 2.0 inches of depth allowance for connectors and cables.
77652	Extended Clock Resolution.	--	--	--	1	Housed in Time-of-Day Clock chassis (77651).
77653	Computer Time Set.	--	--	--	1	Housed in Time-of-Day Clock chassis (77651).
*	Peripheral Equipment Drawer.	5.25	19.0	26.0	15	Includes 5.5 inches of depth depth allowance for connectors and cables.

***Peripheral Equipment Drawer**

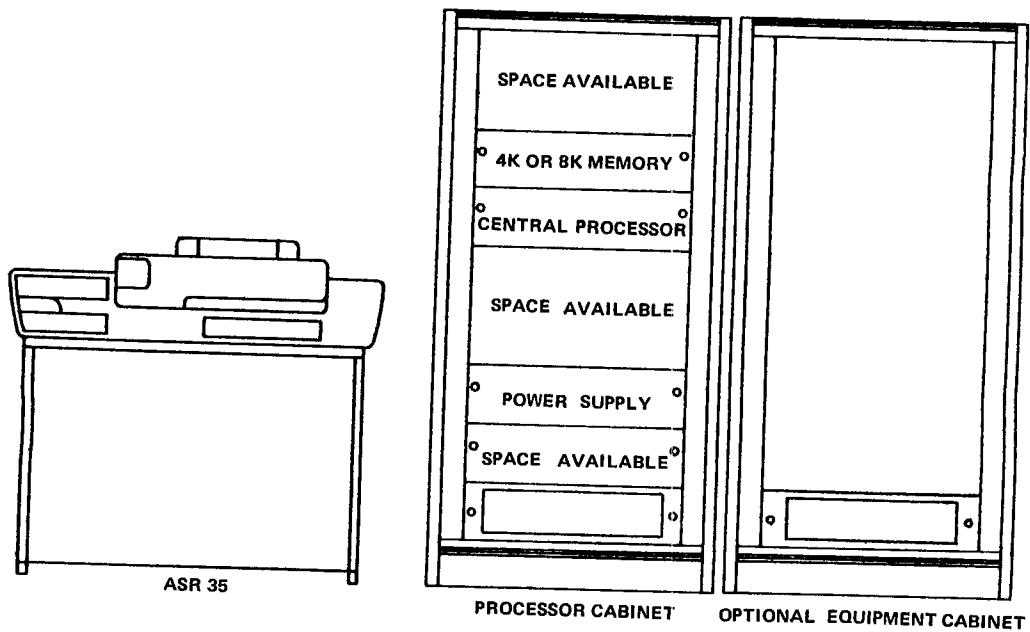
The Peripheral Equipment Drawer can accommodate a number of peripheral equipment controllers. These controllers are mechanized on IC motherboards and two such boards can be accommodated in a drawer. The peripheral equipment controllers have space requirements as follows:

Controller	Comments
Paper Tape (75601 and 75602).	One IC motherboard accommodates both paper tape reader and punch controllers.
Card Equipment (75611, 75612, and 75613).	One IC motherboard accommodates both card reader and punch controllers.
Line Printer (75621, 75622, 75623, and 75624)	One IC motherboard accommodates the line printer controller.
Magnetic Tape (73674 and 73696).	One IC motherboard accommodates the magnetic tape controller (7 or 9 track).
Disc (74601)	One IC motherboard accommodates the disc controller.

APPENDIX H
PHYSICAL CONFIGURATIONS

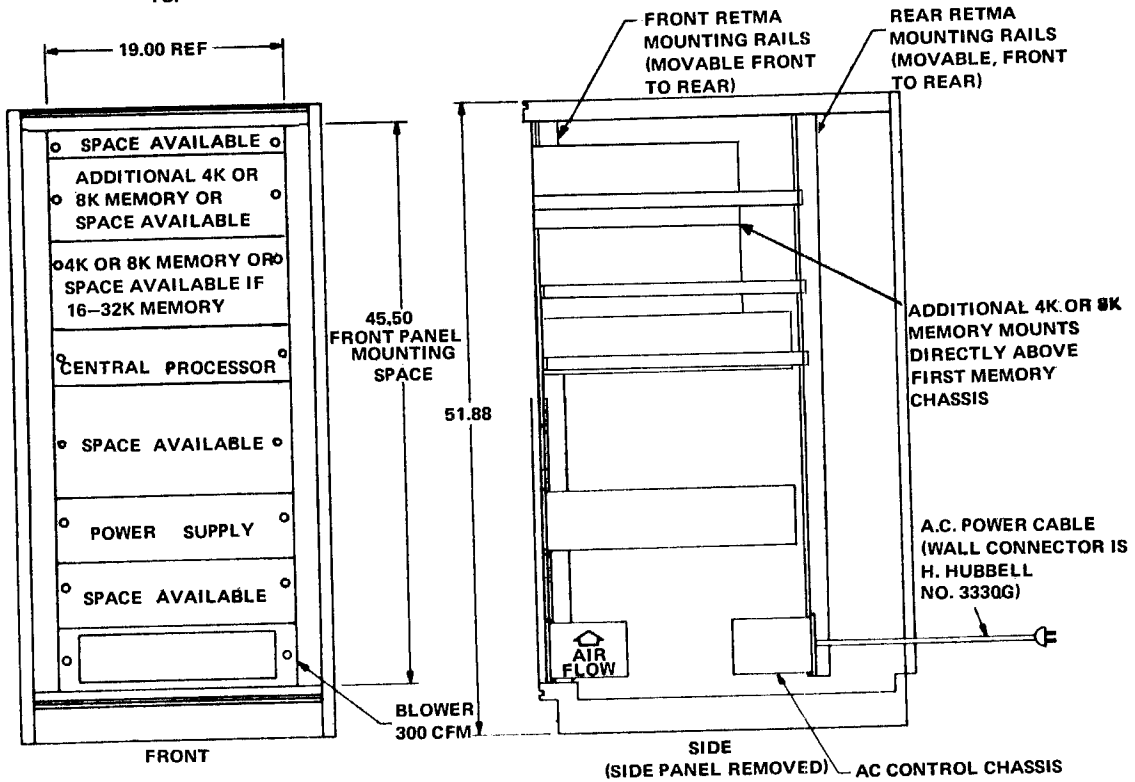
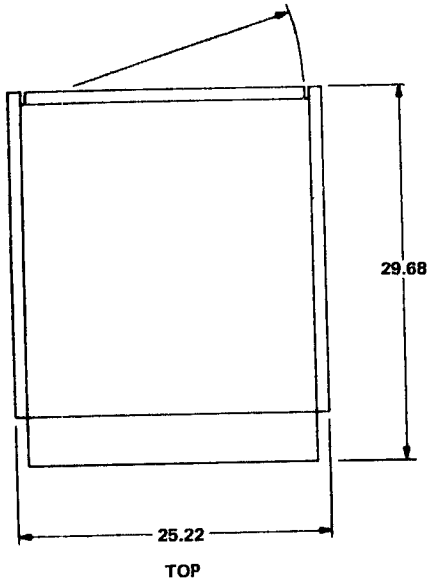


Typical 16K or 32K 706 Computer System Physical Configuration



Typical 4K or 8K 706 Computer System

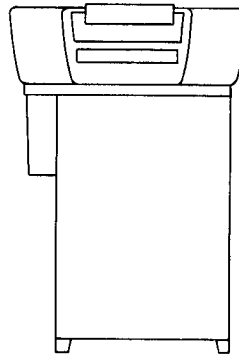
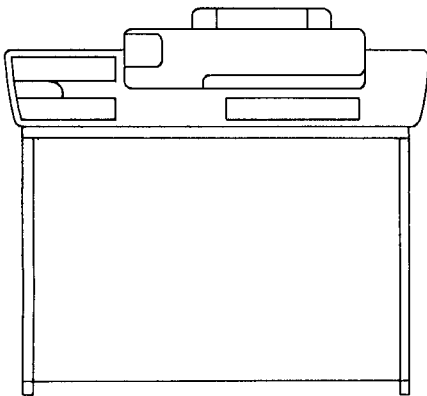
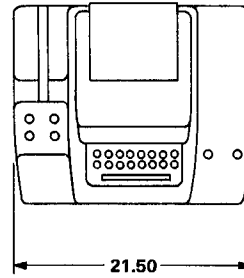
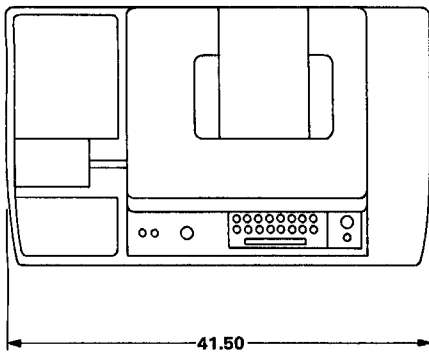
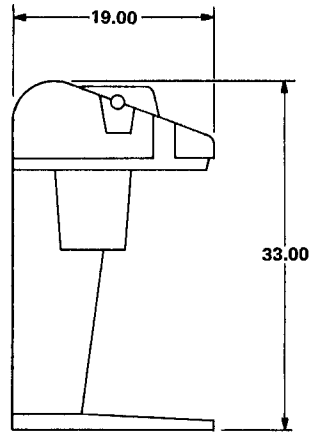
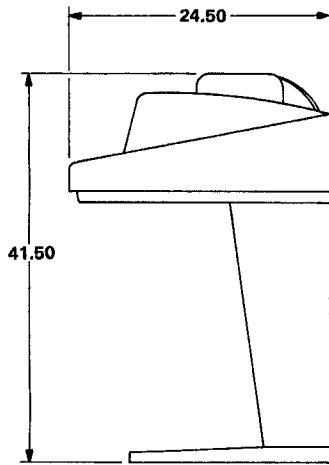
APPENDIX H



ALL DIMENSIONS IN INCHES

706 CPU CHASSIS

APPENDIX H



ASR 35 TELETYPE
WEIGHT: 225 LBS.

ASR 33 TELETYPE
WEIGHT: 56 LBS.

APPENDIX I TIMING CONVERSION

General

This appendix contains information for converting 706 program times to other 700 series compatible computer program times. The following table lists differences between the 706 and other 700 series computers.

Feature	706	703	704
Cycle Time	0.9 μ sec.	1.75 μ sec.	1.00 μ sec.
STB (Cycles)	yes (2)	yes (3)	yes (2)
Shifts (Cycles)	yes (1-4)	yes (2-5)	yes (1-5)
MPY* (Cycles)	yes (7-10)	yes (7-10)	yes (7-10)
DIV* (Cycles)	yes (10-11)	yes (14)	yes (10-11)
SUS* (Cycles)	yes (1)	no	no
LPL* (Cycles)	yes (1)	no	no
LPU* (Cycles)	yes (1)	no	no
Memory Parity	yes (optional)	no	yes (optional)
Memory Protect	yes (optional)	no	no

*Optional

In order to calculate the amount of time a 700 series compatible program will take on the 706, perform the following operations:

1. Total the number of cycles in routine using 700 series machine (C_{7xx}).
2. Make the appropriate cycle changes for the special instructions shown above, and calculate the total number of cycles the 706 takes to execute the routine (C_{706}).
3. The following formulas can be used to determine the routine execution time using a 706 computer (T_{706}) or another 700 series computer (T_{7xx}):

$$T_{706} = (C_{706}) (0.9) \mu \text{ sec.}$$

$$T_{703} = (C_{703}) (1.75) \mu \text{ sec.}$$

$$T_{7xx} = (C_{7xx}) (\text{cycle time for } 7xx) \mu \text{ sec.}$$

APPENDIX J

MATH LIBRARY SUMMARY

TIMING, ACCURACY, AND STORAGE

Math Routine	Entry Names	Cycles-With Software Multiply/Divide			Cycles-With Hardware Multiply/Divide			Storage Required (Words)	Accuracy
		Min	Max	Avg	Min	Max	Avg		
SP Multiply	MPYS	065	105	84			20	71/8*	
SP Divide	DIV	149	193	171	25	30	30	73/16*	
SP Cum. Multiply	ACMY			83			24	6/12*	
DP Fixed Point Load	DLD			15				8	
DP Fixed Point Store	DST			15				8	
DP Fixed Point Add	DAD			24				19	
DP Fixed Point Sub	DSUB			23				15	
DP Fixed Point Two's Complement	D2C			14				13	
Double Shift Arithmetic Left	DSL	55	71	63				40	
Double Shift Arithmetic Right	DSR	54	70	62				E	
Double Shift Magnitude	DSM	23	26	24				12	
MP Floating Underflow	M,ZE			8				5	
MP Floating Overflow	M,OV			21				11	
MP Floating Mutiply	FMP			370			203	209	29 bits
MP Floating Divide	FDV			584			247	A	29 bits
DP Fixed Multiply	CMP			332			165	A	29 bits
DP Fixed Divide	DDV			578			241	A	29 bits
MP Floating Compare	FCM			26				28	

APPENDIX J

MATH LIBRARY SUMMARY (Cont)

TIMING, ACCURACY, STORAGE

Math Routine	Entry Names	Cycles-With Software Multiply/Divide			Cycles-With Hardware Multiply/Divide			Storage Required (Words)	Accuracy
		Min	Max	Avg	Min	Max	Avg		
DP Floating Compare	DCM			15				B	
MP Floating Load	FLD			19				10	
MP Floating Store	FST			19				10	
Convert SP Fixed Point to MP Floating	FLT	41	161	116				29	
Convert DP Fixed Point to MP Floating	FLT2	42	180	127				F	
Convert MP Floating to SP Fixed Point	FIX	29	51	45				39	
Convert MP Floating to DP Fixed Point	FIX2	33	105	98				38	
MP Floating Add	FAD	107	208	120				149	30 bits
MP Floating Sub	FSB	107	208	120				C	30 bits
MP Floating Normalize	FNRM	18	119	50				C	30 bits
MP Floating Point Square Root	MSQR	31	730	676	31	232	215	82	29 bits
MP Floating Point Sine	MSIN	2980	3800	3110	1940	2350	2110	119	29 to 16 bits
MP Floating Point Cosine	MCOS	2730	3800	3110	1010	2350	2110	D	29 to 16 bits
MP Hyperbolic Tangent	MTNH	35	4800	4550	35	3100	2950	91	28 bits
MP Floating Point Arc Tangent	MATN	2700	4800	4120	2000	2900	2690	128	28 bits
MP Floating Point If (Tan < 1)				3820			2470		
MP Floating Point Exponential	MEXP	42	4750		42	2760	2530	138	29 bits

APPENDIX J
MATH LIBRARY SUMMARY (Cont)

TIMING, ACCURACY, STORAGE

Math Routine	Entry Names	Cycles-With Software Multiply/Divide			Cycles-With Hardware Multiply/Divide			Storage Required (Words)	Accuracy
		Min	Max	Avg	Min	Max	Avg		
MP Floating Point Natural Log	MLOG	4000	4600	4230	2700	2850	2740	83	29 bits
MP Floating Point Polynomial	M.P	798	G	G	233	G	G	40	29 bits
703 cycle time is 1.75 μ sec									
704 cycle time is 1.00 μ sec									
706 cycle time is 900 ns									

* Software/Hardware

- A Contained in MP Floating Multiply
- B Contained in MP Floating Compare
- C Contained in MP Floating Add
- D Contained in Sine
- E Contained in Double Shift Arithmetic, Left
- F Contained in FLT
- G Depends on order of polynomial

APPENDIX J

MATH LIBRARY SUMMARY (Cont)

MATH ROUTINE, PART NO., AND PURPOSE

Math Routine	Part No.	Purpose
Math SUBR Storage Pool	391098	<p>There are two non-executable modules which are parts of the 706 MATH SUBROUTINES. One of these is a block of 19 locations used as storage by the various routines. Each location has a label. There are four locations labeled RET1, RET2, RET3 and RET4. These locations are used as temporary storage for the index register return location. There are ten locations labeled TMP1, . . . , TMP9, TMPO. These are used as temporary storage by the executable routines. There are four locations labeled MNT1, . . . , MNT4. These are used as a software register for Double Precision (MNT2, MNT3), for Floating (MNT1, MNT2, MNT3) and for Double Floating (MNT1, MNT3, MNT4) arguments.</p> <p>The label MNT0 refers to the same location as MNT1. There is one word labeled OVFL which is used as a flag to indicate overflow or underflow conditions.</p>
SP Multiply	390663	<p>To multiply two single-word arguments and set their two-word product in the software pseudo-registers MNT2, MNT3. The high word of the product is stored in MNT1. The low word, always given a zero sign bit, is stored in MNT3 and is also left in the accumulator.</p>
SP Divide	390665	<p>To divide a two-word dividend by a single-word divisor, giving a single-word quotient in the software register MNT3, and a single-word remainder in the software register MNT2. The quotient is also duplicated in the accumulator.</p>
SP Cumulative Multiply	391101	<p>To add a one-word number to the product of two one-word numbers</p>
DP Fixed Point Load	391079	<p>To load a double precision fixed point argument from memory into the software registers MNT2, MNT3.</p>
DP Fixed Point Store	391081	<p>To store in two consecutive memory words a double precision fixed point number fetched from the software registers MNT2, MNT3.</p>

APPENDIX J

MATH LIBRARY SUMMARY (Cont)

CALLING SEQUENCE, ARGUMENT DESCRIPTION, AND ERROR CONDITIONS

Calling Sequence	Argument Description and Error Condition
<p>L SMB MPYS L+1 JSX MPYS Return</p>	<p>The two arguments must be in the hardware accumulator and the least significant half of the Double Precision Register, i.e., in MNT3. Error Condition: Squaring -2^{15} will turn on the overflow.</p>
<p>L SMB DIV L+1 JSX DIV Return</p>	<p>The divisor must be placed in the hardware accumulator. The dividend must be set in the software registers MNT2, MNT3. In order to conform with the hardware divide, the routine accepts any setting in the sign bit of the lower word (the standard double precision format requires the second sign bit to be zero). Both arguments are in two's complement form. Error Condition: Any time the divisor is not greater in magnitude than the dividend high word, the overflow is set. Arguments are left unaltered.</p>
<p>L SMB ACMY L+1 JSX ACMY Return</p>	<p>The argument to be added to the product is in the software register MNT2. The two factors of the product are in the software register MNT3 and in the hardware accumulator. All three arguments must be positive.</p>
<p>L SMB ACMY L+1 JSX ACMY L+2 D DARG Return</p>	<p>Where DARG is the first of two words containing the number to be stored into the software registers. The routine will return to L + 2 with the contents of MNT2 in the hardware accumulator.</p>
<p>L SMB DST L+1 JSX DST L+2 D DARG Return</p>	<p>Where DARG is the first of two words into which the contents of the software registers will be stored. The routine will return to L + 2 with the contents of MNT2 in the hardware accumulator.</p>

APPENDIX J

MATH LIBRARY SUMMARY (Cont)

MATH ROUTINE, PART NO., AND PURPOSE

Math Routine	Part No.	Purpose
DP Fixed Point Add, Subtract	391083	To form the algebraic sum or difference of two double precision fixed point numbers set in the software registers MNT2, MNT3 and in two consecutive words of memory.
DP Fixed Point Two's Complement	390664	To form the two's complement of a two-word number set in the software Registers MNT2, MNT3.
Double Shift Arithmetic	391085	To perform the arithmetic shift operation on a two-word argument.
Double Shift Magnitude	390017	To assist the mathematical function subroutines in the double arithmetic shift operations.
MP Floating Underflow	390014	To clear the three software registers MNT1, MNT2, MNT3
MP Floating Overflow	390015	To flag an overflow condition in the overflow flag word OVFL, and to set the software registers MNT1, MNT2, MNT3 to the maximum magnitude of the mid-precision floating point format, keeping the sign as found in MNT2.

APPENDIX J

MATH LIBRARY SUMMARY (Cont)

CALLING SEQUENCE, ARGUMENT DESCRIPTION, AND ERROR CONDITIONS

Calling Sequence			Argument Description and Error Condition
L	SMB	DAD (or DSUB)	<p>DARG is the first of two consecutive memory locations containing a fixed point double precision number. Both arguments must be in double precision format: the sign bit of the second word both arguments must be set to zero.</p> <p>Error Condition: Overflow possible.</p> <p>The argument is in the software registers MNT2, MNT3.</p> <p>Where COUNT is any integer and ARGUMENT is the address of the first word of the two-word argument to be shifted.</p> <p>The first argument must be an integer. The second argument will be treated as a double precision integer.</p> <p>The shift is performed on the contents of the software registers MNT2, MNT3. The sign bit of MNT3 is left out of the shift. On return both sign bits are cleaned off and MNT2 is duplicated in the accumulator register.</p> <p>Place the right byte of the shift instruction (left or right, double, shift count up to 15) in the accumulator register, before calling routine.</p> <p>Error Condition: No check is made on the right half of the shift instruction provided by the caller.</p> <p>M.ZE is called by the library routines dealing with the mid-precision floating point format, when the software registers must be set to zero, as in the case of an underflow condition.</p> <p>M.OV is called by the library routines dealing with the mid-precision floating point format, when an overflow condition occurs.</p>
L+1	JSX	DAD (or DSUB)	
L+2	D	DARG	
	Return		
L	SMB	D2C	
L+1	JSX	D2C	
	Return		
L	SMB	DSR	
L+1	JSX	DSR, Count, Argument	
	Return	(for a shift right)	
	or		
L	SMB	DSL	
L+1	JSX	DSL, Count, Argument	
	Return	(for a shift left)	
L	SMB	DSM	
L+1	JSX	DSM	
	Return		
L	SMB	M.ZE	
L+1	JSX	M.ZE	
	Return		
L	SMB	M.OV	
L+1	JSX	M.OV	
	Return		

APPENDIX J

MATH LIBRARY SUMMARY (Cont)

MATH ROUTINE, PART NO., AND PURPOSE

Math Routine	Part No.	Purpose
MP Floating Multiply	391096	DMP: To multiply two double precision fixed point numbers and return a double precision fixed point product.
DP Fixed Multiply		DDV: To divide a double precision fixed point number by another and return a double precision fixed point quotient.
MP Floating Divide		FMP: To multiply two mid-precision floating point numbers and return a mid-precision floating point.
DP Fixed Divide		FDV: To divide a mid-precision floating point number by another and return a mid-precision floating point quotient.
MP Floating, DP Fixed Compare	391088	To compare an argument in memory, set either in mid-precision floating point (FCM), or in double precision fixed point (DCM) form, to the number set, in the same format, in the pseudo-registers MNT1, MNT2, MNT3, (FCM) or MNT2, MNT3 (DCM).
MP Floating Load	391076	To load into the software registers MNT1, MNT2, MNT3, a mid-precision floating point number set in three consecutive words of memory.
MP Floating Store	391077	To store into three consecutive words of memory a mid-precision floating point number set in the software registers MNT1, MNT2, MNT3.
Convert Fixed Point to MP Floating Point	391094	<p>This subroutine converts data given in the single or double precision fixed point formats to a normalized mid-precision floating point number. The result is stored in the software register MNT1, MNT2, MNT3.</p> <p>The binary point in a fixed point number is normally considered to be between bits 0 and 1, i.e., immediately in front of the first magnitude bit. The use of an algebraic integer N as a scale factor permits to shift the point N bit positions from its normal location, either to the right if the factor is positive, or to the left if it is negative. That is equivalent to multiplying the fixed point number by 2^N prior to conversion, but without altering the number.</p>

APPENDIX J

MATH LIBRARY SUMMARY (Cont)

CALLING SEQUENCE, ARGUMENT DESCRIPTION, AND ERROR CONDITION

Calling Sequence			Argument Description and Error Condition
L	SMB JSX DATA Return	DMP(DDV,FMP, FDV) DMP(DDV,FMP, FDV) ARG	<p>These routines will return to L + 2 with the result in the software registers MNT1, MNT2, MNT3.</p> <p>Two arguments are necessary, one in the software registers and the other starting at the location ARG.</p>
L L+1 L+2	SMB JSX DATA Return	FCM (DCM) FCM (DCM) ARG	<p>DCM: compare most significant words and if equal compare least significant words.</p> <p>FCM: if signs are different call DCM, otherwise compare exponents, if they are the same call DCM.</p> <p>Both routines exit with the machine comparison flip-flops set to indicate the results of the compare. Skip instructions (SEQ, SLE, SGR, etc.) should be used to test the results of the compare.</p>
L L+1 L+2	SMB JSX D Return	FLD FLD FARG	<p>The routine will return to L + 2 with the contents of MNT2 in the hardware accumulator.</p> <p>FARG is the first of three consecutive words of memory containing a floating point number. The first will be an exponent; the remaining two will be a normalized two-word mantissa.</p>
L L+1 L+2	SMB JSX D Return	FST FST FARG	<p>The routine will return to L + 2 with the contents of MNT2 in the hardware accumulator.</p> <p>FARG is the first of three consecutive words of memory. The first will be an exponent; the remaining two will be a Double Precision Normalized word.</p>
L L+1 L+2 L+3	SMB JSX DATA DATA Return	FLT (or FLT2) FLT (or FLT2) ARG N + X'8000'	<p>FLT refers to a single precision fixed point number. FLT2 refers to a double precision fixed point number.</p> <p>Where ARG is the address of the fixed point number, and 2^N is the scale.</p>
L L+1	SMB JSX DATA	FLT (or FLT2) FLT (or FLT2) ARG + X'8000'	<p>To convert from an unscaled integer, the scale factor N = 15 (FLT) or N = 30 (FLT2) need not be given.</p>

APPENDIX J

MATH LIBRARY SUMMARY (Cont)

MATH ROUTINE, PART NO., AND PURPOSE

Math Routine	Part No.	Purpose
Convert MP Floating Point to DP Fixed Point	392338	<p>This subroutine converts data from the mid-precision floating point format to the double precision fixed point format. The latter consists in two consecutive words in two's complement form, where in the first word bit 0 is reserved to the sign and bits 1–15 are the most significant half, while in the second word bit 0 is always zero and bits 1–15 are the least significant half.</p> <p>The binary point is normally located between bits 0 and 1 of the most significant word. However, the use of a binary scale, extrinsic to the format, permits to consider the point to the left or right of its normal location.</p> <p>A fixed point number can be considered as an integer by shifting the binary point to the right of the least significant bit.</p> <p>The use of an algebraic integer N as a scale factor permits to shift the binary point, in the fixed point number, N bit positions from its normal location, either to the right if the factor is positive, or to the left if it is negative. That is equivalent to scaling the floating point number up or down by 2^N prior to the conversion, but without upsetting the given number.</p>
Convert MP Floating Point to SP Fixed Point	392339	<p>This subroutine converts data from the mid-precision floating point format to the single precision fixed point format. A single precision fixed point number consists in one word in two's complement form, where bit 0 and bits 1–15 are reserved respectively to sign and magnitude.</p> <p>The binary point is normally located between bits 0 and 1. However, the use of a binary scale, extrinsic to the format, permits to consider the point at any position to the left or right of its normal location.</p> <p>A fixed point number can be considered as an integer by shifting the binary point to the right of the magnitude bit.</p> <p>The use of an algebraic integer N as a scale factor permits to shift the binary point, in the fixed point number, N bit positions from its normal location, either to the right if the factor is positive, or to the left if it is negative. That is equivalent to scaling the floating point number up or down by 2^N prior to the conversion, but without upsetting the given number.</p>

APPENDIX J

MATH LIBRARY SUMMARY (Cont)

CALLING SEQUENCE, ARGUMENT DESCRIPTION, AND ERROR CONDITIONS

Calling Sequence	Argument Description and Error Condition
L SMB FIX2 L+1 JSX FIX2 L+2 DATA STORE L+3 DATA N + X'8000' L+3 Return	The floating point number to convert is set in the software registers MNT1, MNT2, MNT3, where STORE is the address of the converted number and 2^N is the scale. To convert to an unscaled integer, N = 30 need not be given. Error Condition: An overflow condition occurs when the exponent of the floating point number is greater than the scale factor. The result is then set to $\pm (-2^{-30})$.
L SMB FIX2 L+1 JSX FIX2 L+2 DATA STORE + X'8000' L+2 Return	
L SMB FIX L+1 JSX FIX L+2 DATA STORE L+3 DATA N + X'8000' L+3 Return	The floating point number to convert is set in the software registers, MNT1, MNT2, MNT3, where STORE is the address of the converted number and $2N$ is the scale.
L SMB FIX L+1 JSX FIX L+2 DATA STORE + X'8000' L+2 Return	To convert to an unscaled integer, N = 15 need not be given. Error Condition: An overflow occurs when the exponent of the floating point number is greater than the scale factor. The result is then set to $\pm (1-2^{-15})$.

APPENDIX J

MATH LIBRARY SUMMARY (Cont)

MATH ROUTINE, PART NO., AND PURPOSE

Math Routine	Part No.	Purpose
MP Floating Add, Subtract, Normalize	391090	FAD, FSB: To add or subtract two mid-precision floating point numbers and leave the normalized mid-precision sum or difference in the software registers MNT1, MNT2, MNT3. FNRM: To normalize a floating point number set in the software registers MNT1, MNT2, MNT3.
Square Root MP Floating Point	390006	To extract the square root of a mid-precision floating point number and leave the result in the same format in the software registers MNT1, MNT2, MNT3.
SIN and COS MP Floating Point	390007	To find the sine or the cosine of a mid-precision floating point number and leave the result in the same format in the software registers MNT1, MNT2, MNT3.
Arc Tangent MP Floating Point	390010	To obtain the arc tangent of an argument given in the mid-precision floating point format. The result is set in the same format in the software registers MNT1, MNT2, MNT3.
Exponential MP Floating Point	390009	To find the exponential value of a mid-precision floating point number and leave the result in the same format in the software registers MNT1, MNT2, MNT3.
Natural Logarithm MP Floating Point	390008	To find the natural logarithm of a mid-precision floating point number and leave the result in the same format in the software registers MNT1, MNT2, MNT3.
Hyperbolic Tangent MP Floating Point	392279	To find the hyperbolic tangent of a mid-precision floating point number and leave the result in the same format in the software registers MNT1, MNT2, MNT3.
Polynomial MP Floating Point	390016	To assist the mathematical function subroutines in the expansion of series in the mid-precision floating point format. This subroutine can sum up: $P_1 = X + A_2 \cdot X^2 \dots + A_n \cdot X^n$ or $P_2 = A_1 \cdot X^2 + A_2 \cdot X^4 \dots + A_n \cdot X^{2n}$ or $P_3 = X + A_1 \cdot X^3 \dots + A_n \cdot X^{2n+1}$

APPENDIX J

MATH LIBRARY SUMMARY (Cont)

CALLING SEQUENCE, ARGUMENT DESCRIPTION, AND ERROR CONDITIONS

Calling Sequence			Argument Description and Error Condition	
L	SMBI	FAD (FSB)	<p>All three routines use the Floating Point Register as one argument. FAD and FSB need a search argument, FARG.</p> <p>Error Condition: An overflow or an underflow gives and exponent off by ± 256 and causes the flag word 'OVFL' to be set to non-zero. No error message is given.</p>	
L+1	JSX	FAD (FSB)		
L+2	DATA	FARG		
L+2	Return			
or				
L	SMBI	FNRM		
L+1	JSX	FNRM		
L+1	Return			
L	SMB	MSQR		
L+1	JSX	MSQR		
L+2	D	ARG		
L+2	Return		<p>The argument is a mid-precision floating point number occupying three consecutive words in memory, beginning at address "ARG", as pointed out in the calling sequence.</p> <p>Error Condition: A negative argument is treated as zero. No error message is given.</p>	
L	SMB	MSIN (MCOS)	<p>The argument is given in radians, and is a mid-precision floating point number occupying three consecutive words in memory, beginning at address "ARG" as pointed out in the calling sequence.</p> <p>Error Condition: For argument values greater than or equal to 2^{13}, the function is set to zero.</p>	
L+1	JSX	MSIN (MCOS)		
L+2	D	ARG		
L+2	Return			
L	SMB	MATN		
L+1	JSX	MATN		
L+2	D	ARG		
L+2	Return			
L	SMB	MEXP		
L+1	JSX	MEXP		
L+2	D	ARG		
L+2	Return		<p>The argument is a mid-precision floating point number occupying three consecutive words in memory, beginning at address "ARG" as pointed out in the calling sequence.</p> <p>Error Condition: Overflow is set when the argument is greater than +88.0296918 or less than -89.4159863.</p>	
L	SMB	MLOG	<p>The argument is a mid-precision floating point number occupying three consecutive words in memory, beginning at address "ARG" as pointed out in the calling sequence.</p> <p>Error Condition: Overflow is set for an argument equal to or less than zero.</p>	
L+1	JSX	MLOG		
L+2	D	ARG		
L+2	Return			
L	SMB	MTNH		
L+1	JSX	MTNH		
L+2	D	ARG		
L+2	Return			
<p>The user is referred to the assembly listings of MEXP, MLOG, MSIN, MATN for examples of calling sequences and coefficient tables.</p>				

APPENDIX J

FORTRAN IV LIBRARY FUNCTIONS

Function	Definition	Number of Arguments	Symbolic Name	Argument Data Type	Function Data Type
Absolute Value	$ a $	1	IABS ABS EABS DABS	Integer Real Mid Double	Integer Real Mid Double
Truncation	Sign of a times largest integer $\leq a $	1	INT AINT IDINT EINT	Real Real Double Mid	Integer Real Integer Mid
Choosing Largest Value	Max (a_1, a_2, \dots)	≥ 2	MAX0 MAX1 AMAX0 AMAX1 EMAX1 DMAX1	Integer Real Integer Real Mid Double	Integer Integer Real Real Mid Double
Choosing Smallest Value	Min (a_1, a_2, \dots)	≥ 2	MIN0 MIN1 AMIN0 AMIN1 EMIN1 DMIN1	Integer Real Integer Real Mid Double	Integer Integer Real Real Mid Double
Positive Difference	$a_1 - \text{Min}(a_1, a_2)$	2	IDIM DIM	Integer Real	Integer Real
Obtain most significant part of Double Precision Argument		1	SNGL	Double	Real
Obtain real part of Complex Argument		1	REAL	Complex	Real
Obtain imaginary part of Complex Argument		1	AIMAG	Complex	Real
Express single precision argument in Double precision form		1	DBLE	Real	Double

APPENDIX J

FORTRAN IV LIBRARY FUNCTIONS (Cont)

Function	Definition	Number of Arguments	Symbolic Name	Argument Data Type	Function Data Type
Express two real arguments in complex form	$a_1 + a_2i$	2	CMLX	Real	Complex
Obtain conjugate of complex argument		1	CONJG	Complex	Complex
Exponential	e^a	1	EXP EEXP DEXP CEXP	Real Mid Double Complex	Real Mid Double Complex
Natural Logarithm	$\log_e(a)$	1	ALOG ELOG DLOG CLOG	Real Mid Double Complex	Real Mid Double Complex
Common Logarithm	$\log_{10}(a)$	1	ALOG10 ELOG10 DLOG10	Real Mid Double	Real Mid Double
Trigonometric Sine	$\sin(a)$	1	SIN ESIN DSIN CSIN	Real Mid Double Complex	Real Mid Double Complex
Trigonometric Cosine	$\cos(a)$	1	COS ECOS DCOS CCOS	Real Mid Double Complex	Real Mid Double Complex
Hyperbolic Tangent	$\tanh(a)$	1	TANH ETANH	Real Mid	Real Mid
Square Root	$(a)^{1/2}$	1	SQRT ESQRT DSQRT CSQRT	Real Mid Double Complex	Real Mid Double Complex
Arctangent	Arctan(a)	1 1 1 2 2 2	ATAN EATAN DATAN ATAN2 EATAN2 DATAN2	Real Mid Double Real Mid Double	Real Mid Double Real Mid Double

APPENDIX J

FORTRAN IV LIBRARY FUNCTIONS (Cont)

Function	Definition	Number of Arguments	Symbolic Name	Argument Data Type	Function Data Type
Complex Modulus		1	CABS	Complex	Real
Remaindering*	$a_1 \pmod{a_2}$	2	MOD	Integer	Integer
		2	AMOD	Real	Real
		2	EMOD	Mid	Mid
		2	DMOD	Double	Double
Float	Convert from integer to real	1	FLOAT	Integer	Real
Fix	Convert from real to integer	1	IFIX	Real	Integer
Transfer of Sign	Sign of a_2 times $ a_1 $	2	ISIGN	Integer	Integer
			SIGN	Real	Real
			ESIGN	Mid	Mid
			DSIGN	Double	Double

*The functions MOD, AMOD, EMOD or DMOD (a_1, a_2) is defined as $a_1 - [a_1/a_2] a_2$ where $[X]$ is the whole number (or integer) whose magnitude does not exceed the magnitude of X and whose sign is the same as X .

APPENDIX J
FORTRAN IV RUN-TIME LIBRARY

BASIC COMMANDS AND UTILITY FUNCTIONS

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
RUN-TIME EXEC (NON-RECURSIVE)	R.EXEC R.RET R.RETP R.MACH	391912	0100	02 TO 18
RUN-TIME EXEC (RECURSIVE)	R.EXEC	393251	0190	02 TO VARIABLE
OVERFLOW FLAG SET	R.OV	391913	0007	09
RELATIONAL OPERATORS	R.LT R.GE R.NE R.EQ R.LE R.GT	392925	0100	33 33 33 33 36 36
GO TO FUNCTION	R.GO	390039	0010	10
ERROR MESSAGES	R.ERA R.ER	391890	0075	35 44 OR 11
STOP AND EXIT	R.ST EXIT R.XT	391911	0036	NA
PAUSE	R.PA	390055	0033	NA
COMPUTED GO TO	R.CG	390059	0041	40
DO LOOP TERMINATOR	R.DO	390048	0029	27 TO 30
I/O DUMMY	R.BF R.B	392988	0001	NA
ASSIGN COMMAND	R.AS	390047	0022	28
IF COMMAND	R.IF	390043	0050	36 TO 49
ARRAY ELEMENT ADDRESS CALCU.	R.ARY	392314		
HARD MULT/DIV			0120	43+57N N IS NUMB.
SOFT MULT/DIV			0129	288N-188 OF ELEMENTS

APPENDIX J
FORTRAN IV RUN-TIME LIBRARY (Cont.)

LIBRARY SUBPROGRAMS

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
SENSE SWITCH TEST	SSWTCH	390054	0032	30
OVERFLOW AND DIVIDE CHECK	OVERFL DVCHK	391891	0027	29

INPUT/OUTPUT PROGRAMS

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
RUN-TIME I/O PROCESSOR		391905	2033	NA
FORMATTED I/O	R.OP			
UNFORMATTED I/O	R.IO			
TERMINATE I/O PROCESS	R.CL			
REPETITIVE I/O PROCESSOR	R.RIO	392921	0101	NA
MAGNETIC TAPE SIMULATOR	R.MSIM	392330	0377	NA

INTEGER AND LOGICAL OPERATIONS

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
ONE WORD STORE		390060	0021	28
STORE INTEGER	R.6			
STORE LOGICAL	R.7			
CONVERT LOGICAL TO INTEGER	R.76			
TWO WORD STORE		390027	0026	36
STORE REAL	R.4			
STORE DOUBLE INTEGER	R.5			
CONVERT COMPLEX TO REAL	R.14			

APPENDIX J
FORTRAN IV RUN-TIME LIBRARY (Cont.)

INTEGER AND LOGICAL OPERATIONS (Cont.)

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
INTEGER TO LOGICAL	R.67	390021	0023	30
INTEGER AND LOGICAL TO DOUBLE PRECISION, MID-PRECISION AND REAL		390024	0044	
CONVERT INTEGER TO DP	R.62			27 TO 116
CONVERT INTEGER TO MP	R.63			24 TO 113
CONVERT INTEGER TO REAL	R.64			18 TO 107
CONVERT LOGICAL TO DP	R.72			27 TO 116
CONVERT LOGICAL TO MP	R.73			24 TO 113
CONVERT LOGICAL TO REAL	R.74			18 TO 107
INTEGER NEGATIVE STORE	R.N6	390040	0022	29
INTEGER ADD,SUB,AND,OR,EOR		390037	0057	46
ADD TWO INTEGERS	R.A6			
SUBTRACT TWO INTEGERS	R.S6			
AND TWO LOGICAL VALUES	R.AND			
OR TWO LOGICAL VALUES	R.OR			
EXCLUSIVE OR TWO LOGICAL	R.EOR			
SP INTEGER MULTIPLY		391901		
SOFT, MPY/DIV	R.M6		0047	29 TO 58 (42 AVE.)
HARD, MPY/DIV	R.MPY		0047	16 TO 69 (17 AVE.)
SP INTEGER QUOTIENT AND REM.		391902		
SOFT, MYP/DIV	R.D6		0052	43 TO 55 (44 AVE.)
HARD, MYP/DIV	R.DIV		0052	18 TO 27 (27 AVE.)
REMAINDER	MOD			35 TO 47 (45 AVE.)
LOGICAL NOT	R.NOT	390038	0022	30
EXPONENTIATION I**J	R.E66	391896	0080	
SOFT, MYP/DIV				29 TO 5600
HARD, MYP/DIV				29 TO 5600

APPENDIX J

FORTRAN IV RUN-TIME LIBRARY (Cont.)

INTEGER LIBRARY FUNCTIONS

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
INTEGER SIGN TRANSFER	ISIGN	390034	0028	53
INTEGER ABSOLUTE VALUE	IABS	390031	0024	30
INTEGER POSITIVE DIFFERENCE	IDIM	390030	0031	40
INTEGER MINIMUM VALUE	MINO AMINO	390032	0070	38+21(N-1) N IS NUMB. 37+21(N-1) OF ARGUM.
INTEGER MAXIMUM VALUE	MAXO AMAXO	390033	0070	38+21(N-1) 37+21(N-1)

REAL AND MID-PRECISION OPERATIONS

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
FOUR WORD STORE		390028	0035	
STORE COMPLEX IN COMPLEX	R.1			52
STORE DP IN DP	R.2			52
STORE MP IN MP	R.3			52
DOUBLE AND MID-PRECIS. TO REAL		390025	0047	
CONVERT DP TO REAL	R.24			60
CONVERT MP TO REAL	R.34			60
COMPLEX AND REAL TO DP AND MP		390026	0032	
CONVERT COMPLEX TO DP	R.12			49
CONVERT COMPLEX TO MP	R.13			49
CONVERT REAL TO DP	R.42			49
CONVERT REAL TO MP	R.43			49

APPENDIX J
FORTRAN IV RUN-TIME LIBRARY (Cont.)

REAL AND MID-PRECISION OPERATIONS (Cont.)

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
EXPONENTIATION X**Y		391895	0155	
MID-PRECISION	R.E33			
SOFT, MYP/DIV				218 TO 12100 (10600 AVE.)
HARD, MYP/DIV				218 TO 7050 (6280 AVE.)
REAL	R.E44			
SOFT, MYP/DIV				239 TO 12130 (10626 AVE.)
HARD, MYP/DIV				239 TO 7080 (6306 AVE.)
EXPONENTIATION X(MIO)**Y(REAL)	R.E34	391897	0023	
SOFT, MPY/DIV				267 TO 12149 (10649 AVE.)
HARD, MPY/DIV				267 TO 7099 (6329 AVE.)
EXPONENTIATION X**J		391898	0128	
MID-PRECISION	R.E36			
SOFT, MPY/DIV				217 TO 10980 (9700 AVE.)
HARD, MPY/DIV				217 TO 6310 (5730 AVE.)
REAL	R.E46			
SOFT, MYP/DIV				230 TO 11002 (9718 AVE.)
HARD, MPY6DIV				230 TO 6332 (5748 AVE.)
POLYNOMIAL	R.PM	391892	0067	
SOFT, MPY/DIV				325 TO 2385 (4TH DEG.)
HARD, MPY/DIV				130 TO 1410 (4TH DEG.)
DOUBLE SHIFT MAGNITUDE	R.DSH	391893	0024	40 TO 44
REAL NEGATIVE STORE	R.N4	390041	0032	37
MP NEGATIVE STORE	R.N3	390042	0042	51

APPENDIX J
FORTRAN IV RUN-TIME LIBRARY (Cont.)

REAL AND MID-PRECISION OPERATIONS (Cont.)

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
COMPLEX,DP,MP,REAL TO INTEGER		390023	0040	46
CONVERT COMPLEX TO INTEGER	R.16			
CONVERT DP TO INTEGER	R.26			
CONVERT MP TO INTEGER	R.36			
CONVERT REAL TO INTEGER	R.46			
COMPLEX,DP,MP,REAL TO LOGICAL		390022	0028	218
CONVERT COMPLEX TO LOGICAL	R.17			
CONVERT DP TO LOGICAL	R.27			
CONVERT MP TO LOGICAL	R.37			
CONVERT REAL TO LOGICAL	R.47			
FLOATING POINT				
SOFT, MPY DIV		392311	0627	
REAL ADD	R.A4			186 TO 332 (203 AVE.)
REAL SUBTRACT	R.S4			186 TO 332 (203 AVE.)
REAL MULTIPLY	R.M4			352 TO 507 (536 AVE.)
REAL DIVIDE	R.D4			303 TO 793 (641 AVE.)
MP ADD	R.A3			169 TO 306 (182 AVE.)
MP SUBTRACT	R.S3			169 TO 306 (182 AVE.)
MP MULTIPLY	R.M3			335 TO 481 (415 AVE.)
MP DIVIDE	R.D3			286 TO 767 (620 AVE.)
HARD, MPY/DIV		391903	0551	
REAL ADD	R.A4			186 TO 332 (203 AVE.)
REAL SUBTRACT	R.S4			186 TO 332 (203 AVE.)
REAL MULTIPLY	R.M4			200 TO 275 (241 AVE.)
REAL DIVIDE	R.D4			200 TO 364 (281 AVE.)

APPENDIX J

FORTRAN IV RUN-TIME LIBRARY (Cont.)

REAL AND MID-PRECISION OPERATIONS (Cont.)

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
MP ADD	R.A3			169 TO 306 (182 AVE.)
MP SUBTRACT	R.S3			169 TO 306 (182 AVE.)
MP MULTIPLY	R.M3			183 TO 249 (220 AVE.)
MP DIVIDE	R.D3			183 TO 338 (260 AVE.)

REAL AND MID-PRECISION LIBRARY FUNCTIONS

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
REAL,MP,AND DP ABSOLUTE VALUE		390045	0041	
REAL	ABS			163
MP	EABS			77
DP	DABS			90
REAL POSITIVE DIFFERENCE	DIM	390046	0037	505
REAL,MP,AND INTEGER MIN VALUE		390044	0145	
REAL	MIN1			213 + 116(N-1) WHERE N IS
INTEGER	AMIN1			226 + 115(N-1) THE NUMBER
MP	EMIN1			112 + 84(N-1) OF ARGUMENTS
REAL,MP AND INTEGER MAX VALUE		391889	0145	
REAL	MAX1			213 + 116(N-1) WHERE N IS
INTEGER	AMAX1			226 + 115(N-1) THE NUMBER
MP	EMAX1			112 + 84(N-1) OF ARGUMENTS
EXPONENTIAL		390052	0188	
SOFT, MPY/DIV				
REAL	EXP			243 TO 5295 (4371 AVE.)
MP	EEXP			230 TO 5273 (4353 AVE.)
SPECIAL	R.EXM			27 TO 5070 (4150 AVE.)

APPENDIX J
FORTRAN IV RUN-TIME LIBRARY (Cont.)

REAL AND MID-PRECISION LIBRARY FUNCTIONS (Cont.)

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
HARD, MPY/DIV				
REAL	EXP			243 TO 2755 (2371 AVE.)
MP	EEXP			230 TO 2733 (2353 AVE.)
SPECIAL	R.EXM			27 TO 2530 (2150 AVE.)
SINE AND COSINE		390050	0161	
SOFT, MPY/DIV				
REAL SINE	SIN			237 TO 4180 (3514 AVE.)
REAL COSINE	COS			238 TO 4179 (3515 AVE.)
MP SINE	ESIN			224 TO 4158 (3496 AVE.)
MP COSINE	ECOS			225 TO 4157 (3497 AVE.)
HARD, MPY/DIV				
REAL SINE	SIN			237 TO 2746 (2258 AVE.)
REAL COSINE	COS			238 TO 2745 (2259 AVE.)
MP SINE	ESIN			224 TO 2724 (2238 AVE.)
MP COSINE	ECOS			225 TO 2723 (2239 AVE.)
ARC TANGENT		390051	0242	
SOFT, MPY/DIV				
1 REAL VARIABLE	ATAN			850 TO 5349 (4653 AVE.)
2 REAL VARIABLES	ATAN2			1420 TO 5806 (4953 AVE.)
1 MP VARIABLE	EATAN			833 TO 5323 (4633 AVE.)
2 MP VARIABLES	EATAN2			1403 TO 6053 (5023 AVE.)
1 VARIABLE SPECIAL CALL	R,ATN			630 TO 5120 (4430 AVE.)
2 VARIABLES SPECIAL CALL	R,ATN2			1200 TO 5850 (4820 AVE.)

APPENDIX J

FORTRAN IV RUN-TIME LIBRARY (Cont.)

REAL AND MID-PRECISION LIBRARY FUNCTIONS (Cont.)

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
HARD, MPY/DIV				
1 REAL VARIABLE	ATAN			520 TO 3279 (2893 AVE.)
2 REAL VARIABLES	ATAN2			767 TO 3726 (3107 AVE.)
1 MP VARIABLE	EATAN			503 TO 3253 (2873 AVE.)
2 MP VARIABLES	EATAN2			743 TO 3693 (3083 AVE.)
1 VARIABLE SPECIAL CALL	R,ATN			300 TO 3050 (2670 AVE.)
2 VARIABLES SPECIAL CALL	R,ATN2			540 TO 3490 (2880 AVE.)
SQUARE ROOT		390049	0116	
SOFT, MPY/DIV				
REAL	SQRT			101 TO 851 (767 AVE.)
MP	ESQRT			86 TO 836 (752 AVE.)
MACHINE LEVEL	R.SQM			26 TO 776 (692 AVE.)
HARD, MPY/DIV				
REAL	SQRT			101 TO 351 (327 AVE.)
MP	ESQRT			86 TO 336 (312 AVE.)
MACHINE LEVEL	R.SQM			26 TO 276 (252 AVE.)
HYPERBOLIC TANGENT		392278	0136	
SOFT, MPY/DIV				
REAL	TANH			224 TO 5320 (5120 AVE.)
MP	ETANH			212 TO 5300 (5100 AVE.)
HARD, MPY/DIV				
REAL	TANH			224 TO 3020 (2920 AVE.)
MP	ETANH			212 TO 3000 (2900 AVE.)
REAL AND MP SIGN TRANSFER		390035	0056	
REAL	SIGN			172
MP	ESIGN			82

APPENDIX J
 FORTRAN IV RUN-TIME LIBRARY (Cont.)

REAL AND MID-PRECISION LIBRARY FUNCTIONS (Cont.)

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
X MODULO Y		391899	0086	
REAL	AMOD			231 TO 9525
MP	EMOD			218 TO 9500
TRUNCATION		391900	0052	
REAL	AINT			89 TO 168 (145 AVE.)
MP	EINT			78 TO 148 (129 AVE.)
NATURAL LOGARITHM		390053	0133	
SOFT, MPY/DIV				
REAL	ALOG			4626 TO 4955 (4701 AVE.)
MP	ELOG			4613 TO 4933 (4683 AVE.)
SPECIAL CALL	R.LGM			4410 TO 4730 (4480 AVE.)
HARD, MPY/DIV				
REAL	ALOG			2946 TO 3125 (2946 AVE.)
MP	ELOG			2933 TO 3103 (2963 AVE.)
SPECIAL CALL	R.LGM			2730 TO 2900 (2760 AVE.)
DECIMAL LOGARITHM		391894	0021	
SOFT, MPY/DIV				
REAL	ALOG10			4823 TO 5292 (4978 AVE.)
MP	ELOG10			4800 TO 5260 (4950 AVE.)
HARD, MPY/DIV				
REAL	ALOG10			2983 TO 3232 (3058 AVE.)
MP	ELOG10			2960 TO 3200 (3030 AVE.)

APPENDIX J
 FORTRAN IV RUN-TIME LIBRARY (Cont.)

DOUBLE PRECISION INTEGER OPERATIONS

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
DOUBLE INTEGER TO INTEGER	R.56	391885	0027	37
INTEGER AND LOGICAL TO DI		391887	0029	36
INTEGER	R.65			
LOGICAL	R.75			
DOUBLE INTEGER NEGATIVE STORE	R.N5	391872	0035	41
DOUBLE INTEGER TO REAL AND MP		392247	0024	
REAL	R.53			154
MP	R.54			160
DOUBLE INTEGER ADD	R.A5	392254	0054	57 TO 74 (58 AVE.)
DOUBLE INTEGER SUBTRACT	R.S5	392255	0052	65 TO 78 (67 AVE.)
DOUBLE INTEGER MULTIPLY	R.M5	392256	0101	
SOFT, MPY/DIV				133 TO 358 (303 AVE.)
HARD, MPY/DIV				133 TO 220 (190 AVE.)
DOUBLE INTEGER DIVIDE	R.D5	392257	0141	
SOFT, MPY/DIV				144 TO 1030 (460 AVE.)
HARD, MPY/DIV				144 TO 1030 (340 AVE.)
EXPONENTIATION X**JJ		392955	0156	
SOFT, MPY/DIV				
REAL	R.E45			232 TO 11000 (9700 AVE.)
MP	R.E35			219 TO 11000 (9700 AVE.)
HARD, MPY/DIV				
REAL	R.E45			232 TO 6350 (5750 AVE.)
MP	R.E35			219 TO 6350 (5750 AVE.)

APPENDIX J

FORTTRAN IV RUN-TIME LIBRARY (Cont.)

DOUBLE PRECISION INTEGER OPERATIONS (Cont.)

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
DP INTEGER EXPONENTIATION		392954	0100	
SOFT, MPY/DIV				
DOUBLE INTEGER EXPONENT	R.E55			50 TO 13000 (1446 AVE.)
SINGLE INTEGER EXPONENT	R.E56			41 TO 13000 (1437 AVE.)
HARD, MPY/DIV				
DOUBLE INTEGER EXPONENT	R.E55			50 TO 10000 (1116 AVE.)
SINGLE INTEGER EXPONENT	R.E56			41 TO 10000 (1107 AVE.)
THREE WORD SHIFT	R.TSH	392953	0036	59 TO 67
DOUBLE INTEGER TO LOGICAL	R.57	391886	0023	30
COMPLEX,REAL,MP,DP TO DBLE.	INTEGER	392246	0096	
COMPLEX	R.15			39 TO 75 (61 AVE.)
DP	R.25			36 TO 75 (58 AVE.)
MP	R.35			37 TO 73 (59 AVE.)
REAL	R.45			39 TO 75 (61 AVE.)
DOUBLE INTEGER TO DP	R.52	392280	0018	194

COMPLEX OPERATIONS

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
NON-COMPLEX DATA TYPES TO COMPLEX		391888	0059	
DOUBLE PRECISION	R.21			168
MP	R.31			168
DOUBLE INTEGER	R.41			71
REAL	R.51			281
INTEGER	R.61			105
LOGICAL	R.71			105

APPENDIX J
FORTRAN IV RUN-TIME LIBRARY (Cont.)

COMPLEX OPERATIONS (Cont.)

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
COMPLEX ADD AND SUBTRACT	R.A1,R.S1	392926	0152	
SOFT, MPY/DIV				1582
HARD, MPY/DIV				1582
COMPLEX MULTIPLY AND DIVIDE		392927	0297	
SOFT, MPY/DIV				
MULTIPLY	R.M1			4128
DIVIDE	R.D1			7085
HARD, MPY/DIV				
MULTIPLY	R.M1			3348
DIVIDE	R.D1			5195
COMPLEX** MID-PRECISION	R.E13	392266		
SOFT, MPY/DIV				28645
HARD, MPY/DIV				18579
COMPLEX ** INT,DI,REAL,DP		392928	0074	
SOFT, MPY/DIV				
INTEGER	R.E16			28874
DOUBLE INTEGER	R.E15			28057
REAL	R.E14			28847
DOUBLE PRECISION	R.E12			28798
HARD, MPY/DIV				
INTEGER	R.E16			18808
DOUBLE INTEGER	R.E15			18991
REAL	R.E14			18781
DOUBLE PRECISION	R.E12			18732

APPENDIX J
FORTTRAN IV RUN-TIME LIBRARY (Cont.)

COMPLEX OPERATIONS (Cont.)

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
COMPLEX ** COMPLEX	R.E11	392268	0042	
SOFT, MPY/DIV				32044
HARD, MPY/DIV				21588
COMPLEX NEGATIVE STORE	R.N1	391908	0044	198

COMPLEX LIBRARY FUNCTIONS

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
COMPLEX IMAGINARY PART	AIMAG	392269	0026	
SOFT, MPY/DIV				197
HARD, MPY/DIV				197
TWO REALS TO COMPLEX	CMPLX	392270	0025	160
COMPLEX CONJUGATE	CONJG	392271	0059	
SOFT, MPY/DIV				843
HARD, MPY/DIV				843
COMPLEX MODULUS	CABS	392272	0082	
SOFT, MPY/DIV				2787
HARD, MPY/DIV				1951
COMPLEX EXPONENTIAL	CEXP	392273	0107	
SOFT, MPY/DIV				13702
HARD, MPY/DIV				8792

APPENDIX J
FORTRAN IV RUN-TIME LIBRARY (Cont.)

COMPLEX LIBRARY FUNCTIONS (Cont.)

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
COMPLEX NATURAL LOG	CLOG	392274	0126	
SOFT, MPY/DIV				13717
HARD, MPY/DIV				8947
COMPLEX SINE AND COSINE	CSIN,CCOS	392929	0276	
SOFT, MPY/DIV				16631
HARD, MPY/DIV				10645
COMPLEX SQUARE ROOT	CSQRT	392277	0176	
SOFT, MPY/DIV				7299 TO 7796
HARD, MPY/DIV				4869 TO 5366

DOUBLE PRECISION OPERATIONS

Description	Label	Part No.	Size (Dec)	Cycles (Timing)
DP NEGATIVE STORE	R.N2	391873	0056	64
MP TO DP CONVERSION	R.32	390029	0032	47
DP POLYNOMIAL	R.POD	392949	0068	
SOFT, MPY/DIV				758 TO 4702 (4TH DEGREE)
HARD, MPY/DIV				724 TO 2152 (4TH DEGREE)
DP EXPONENTIATION X ** JJ	R.E25	392945	0151	
SOFT, MPY/DIV				227 TO 22900 (21800 AVE.)
HARD, MPY/DIV				227 TO 11500 (10800 AVE.)
DP EXPONENTIATION X**J	R.E26	392944	0121	
SOFT, MPY/DIV				224 TO 22900 (21800 AVE.)
HARD, MPY/DIV				224 TO 11500 (10800 AVE.)

APPENDIX J

FORTRAN IV RUN-TIME LIBRARY (Cont.)

DOUBLE PRECISION OPERATIONS (Cont.)

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
DP EXPONENTIATION X**Y	R.E22	392943	0155	
SOFT, MPY/DIV				226 TO 25300 (23400 AVE.)
HARD, MPY/DIV				226 TO 12200 (11500 AVE.)
DP TO MID-PRECISION	R.23	391874	0036	39
DOUBLE PRECISION COMPARE	R.DCMP	392283	0056	70
DP FLOATING POINT				
SOFT, MPY/DIV		392952	0745	
ADD	R.AD			158 TO 393 (247 AVE.)
SUBTRACT	R.SD			158 TO 393 (247 AVE.)
MULTIPLY	R.MD			642 TO 1040 (812 AVE.)
DIVIDE	R.DD			100 TO 1484 (1160 AVE.)
HARD, MPY/DIV		392951	0717	
ADD	R.AD			158 TO 393 (247 AVE.)
SUBTRACT	R.SD			158 TO 393 (247 AVE.)
MULTIPLY	R.MD			243 TO 402 (302 AVE.)
DIVIDE	R.DD			100 TO 643 (426 AVE.)

DOUBLE PRECISION LIBRARY FUNCTIONS

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
DOUBLE PRECISION X(MOD Y)	DMOD	392950	0060	134 TO 21000
DOUBLE PRECISION EXPONENTIAL		392948	0223	

APPENDIX J

FORTRAN IV RUN-TIME LIBRARY (Cont.)

DOUBLE PRECISION LIBRARY FUNCTIONS (Cont.)

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
SOFT, MPY/DIV				
NORMAL ENTRY	DEXP			241 TO 10600 (10100 AVE.)
SPECIAL ENTRY	R.EXD			30 TO 10400 (9900 AVE.)
HARD, MPY/DIV				
NORMAL ENTRY	DEXP			241 TO 5400 (5000 AVE.)
SPECIAL ENTRY	R.EXD			30 TO 5200 (4800 AVE.)
DOUBLE PRECISION NATURAL LOG		392947	0151	
SOFT, MPY/DIV				
NORMAL ENTRY	DLOG			10400 TO 10900 (10600 AVE.)
SPECIAL ENTRY	R.LGD			10200 TO 10700 (10400 AVE.)
HARD, MPY/DIV				
NORMAL ENTRY	DLOG			5200 TO 5500 (5300 AVE.)
SPECIAL ENTRY	R.LGD			5000 TO 5300 (5100 AVE.)
DOUBLE PRECISION COMMON LOG	DLOG10	392946	0019	
SOFT, MPY/DIV				11100 TO 11800 (11400 AVE.)
HARD, MPY/DIV				5450 TO 5900 (5600 AVE.)
DOUBLE PRECISION ARC TANGENT		392941	0238	
SOFT, MPY/DIV				
ONE VARIABLE CALL	DATAN			1430 TO 12600 (10950 AVE.)
TWO VARIABLE CALL	DATAN2			1430 TO 13500 (11550 AVE.)
HARD, MPY/DIV				
ONE VARIABLE CALL	DATAN			710 TO 6500 (5400 AVE.)
TWO VARIABLE CALL	DATAN2			710 TO 6700 (5650 AVE.)

APPENDIX J

FORTRAN IV RUN-TIME LIBRARY (Cont.)

DOUBLE PRECISION LIBRARY FUNCTIONS (Cont.)

Description	Label	Part No.	Size (Dec)	Timing (Cycles)
DOUBLE PRECISION SINE/COSINE		392940	0195	
SOFT, MPY/DIV				
SINE	DSIN			239 TO 8700 (7950 AVE.)
COSINE	DCOS			241 TO 8700 (7950 AVE.)
HARD, MPY/DIV				
SINE	DSIN			239 TO 4450 (4040 AVE.)
COSINE	DCOS			241 TO 4450 (4040 AVE.)
DOUBLE PRECISION SQUARE ROOT	DSQRT	392939	0133	
SOFT, MPY/DIV				89 TO 2450 (2070 AVE.)
HARD, MPY/DIV				89 TO 1150 (900 AVE.)
DOUBLE PRECISION SIGN TRANS.	DSIGN	391910	0033	88
MAXIMUM VALUE DOUBLE PREC.	DMAX1	392281	0078	88 + 81(N-1) WHERE N IS
MINIMUM VALUE DOUBLE PREC.	DMIN1	392282	0080	87 + 81(N-1) NO. ARGUME.

APPENDIX J

CONVERSATIONAL FORTRAN LIBRARY FUNCTIONS

Function	Definition	Number of Arguments	Symbolic Name	Argument Data Type	Function Data Type
Logical AND	$a \cap b$	2	IAND	Integer	Integer
Logical OR	$a \cup b$	2	IOR	Integer	Integer
Logical EOR	$a \oplus b$	2	IEOR	Integer	Integer
Logical NOT	a	1	NOT	Integer	Integer
Absolute Value	$ a $	1	ABS IABS	Real Integer	Real Integer
Transfer of sign	Sign of a_2 times a_1	2	SIGN ISIGN	Real Integer	Real Integer
Exponential	e^a	1	EXP	Real	Real
Natural Logarithm	$\log_e(a)$	1	ALOG	Real	Real
Trigonometric Sine	Sine (a)	1	SIN	Real	Real
Trigonometric Cosine	Cosine (a)	1	COS	Real	Real
Hyperbolic Tangent	$\tanh(a)$	1	TANH	Real	Real
Square root	(a)	1	SQRT	Real	Real
Arctangent	Arctan (a)	1	ATAN	Real	Real
Test sense switch (0-2)	Set, function = 1 Reset, function = 0	1	ISWCH	Integer	Integer
Float	Conversion from integer to real	1	FLOAT	Integer	Real
Fix	Conversion from real to integer	1	IFIX	Real	Integer

APPENDIX J
FORTRAN ATP FUNCTION CALLS

Array Operation	Function Name	Argument List					Data Formats			Overflow Flag
		Array 1	Array 2	Array 3	Scale	Flag	Array 1	Array 2	Array 3	
<u>LOGISTICS</u>										
Clear	RCLR			C, N, K		0 1			SP DP, CX	
Relocate	RLOC	A, I		C, N, K		0 1	SP DP		SP DP	
Relocate and Convert						2 3	SP1 SP2		SP SP	
Reorder	RORD	B, J	A, I	C, N		0 1	SP SP	SP DP	SP DP	
Reorder and Swap						2 3	SP SP	SP DP	SP DP	
Restore FFT Table	RFUT			C, N					CX	
Memory Cycle Control	RCYC					0, 1				
<u>SCALE</u>										
Normalize	RNRM	A, I	S*	C, N, K	S	0 1 2 3	SP DP SP DP		SP SP(R) DP DP	
Scan Magnitude	RMAG	A, M, I		S			SP		SP	
Scan Maximum	RMAX	A, M, I		S			SP		SP	
Scan Minimum	RMIN	A, M, I		S			SP		SP	
Shift (Arithmetic)	RSFT	A, I	S	C, N, K		0 1 2 3	SP DP SP DP		SP SP(R) DP DP	No No No No
<u>REAL ARITHMETIC</u>										
Real Add	RAD RDAD	A, I A, I	B, J B, J	C, N, K C, N, K		0 1 2 3	SP SP DP SP DP	SP SP DP DP DP	SP SP DP DP DP	Yes Yes Yes Yes Yes
Real Subtract	RSB RDSB	A, I A, I	B, J B, J	C, N, K C, N, K		0 1 2 3	SP SP DP SP DP	SP SP DP DP DP	SP SP DP DP DP	Yes Yes Yes Yes Yes

**APPENDIX J
FORTRAN ATP FUNCTION CALLS (Cont.)**

Array Operation	Function Name	Argument List					Data Formats			Overflow Flag	
		Array 1	Array 2	Array 3	Scale	Flag	Array 1	Array 2	Array 3		
Real Multiply	RMP	A, I	B, J	C, N, K			SP	SP	SP(R)		
	RDMP	A, I	B, J	C, N, K		1	SP	SP	SP(L)		
						2	SP	SP	SP(R)		
						3	SP	SP	DP		
						5	DP	SP	DP(L)		
						6	DP	SP	DP(R)		
COMPLEX ARITHMETIC											
Complex Multiply	RCMP	A, I	B, J	C, N, K		0.1	CX	CX	CX(R)	Yes	
Magnitude Spectrum	RCSM	A, I		C, N, K		1	CX		SP(L)	Yes	
							CX		SP(R)	Yes	
							CX		DP	Yes	
MATH FUNCTIONS											
Convolution Integral	RCON	B, M, J	A, I	C, N, K		1	SP	SP	SP(L)		
							SP	SP	SP(LR)		
							SP	SP	DP((L)		
							1 4	SP	SP	SP(R)	
							6	SP	SP	DP(R)	
							7	SP	SP	TP	
Fast Fourier Transform	RFFT	A	B	N	S*	0,1*	CX	CX	CX	***	
FFT Inverst	RFTI	A	B	N	S*	0,1*	CX	CX	CX	***	

SYMBOL EXPLANATION

Argument Type

A, B, C = array base of operand, operator, destination

I, J, K = index increment for operand, operator, destination

N = primary count

M = secondary count

S = scalar

Observations

* = assigned by ATP

** = reassigned by ATP

*** = overflow flagged in Flag variable

Data Format

SP = single precision

SP1 = single precision (1's complement)

SP2 = single precision (sign-magnitude)

DP = double precision

TP = triple precision

CX = complex

(L) = left truncated

(R) = right truncated

APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY

K-1. SYM I/PREP CHARACTERISTICS

SYM I

Operates Under System	BASIC
Label	SYM I
Part Number	390470
No. of Passes	1
Symbols	Forward defined
Pseudo-Operations (directives)	12
Conditional Directives	Yes (TRUE,FALS,ENDC)
Conversational Mode	Yes
Procedure Capability	No
Total Memory Required	2460 locations
Absolute Assembled Programs	1000 instructions, 100 symbols (4 K computer)
Relocatable Assembled Programs	500 instructions, 100 symbols (4 K computer)
Source Input Units	ASR-33 PTR, ASR-33 keyboard, HSPTTR
Binary (object) Output Units	ASR-33 PTP, HSPTP

STATEMENT FORMATS

Types of Statements	Comment, instruction
Coding Form Fields	4; Label, Operation, Operand, Comment
Label (name of location)	1-4 characters; 1st must be alphabetic, additional characters ignored, forward defined labels allowed.
Operation	3 or 4 letter codes (see Section 2)
Operand	Names, expressions, constants (see Figure 1-9)
Comment	
Delimiter	1 or more blank characters

SYNTAX

Character Set	
Alphabetic	A through Z

**APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY (Cont)**

K-1. SYM I/PREP CHARACTERISTICS (Cont)

Numeric	0 through 9
Special	+ - , . \$ "blank" / = *
Other	May appear in alphanumeric constants or comments
Constants	
Decimal Integer	1–5 digits; 32,767 ($2^{15} - 1$) maximum value
Hexadecimal Integer	1–4 Hex digits (X'3F90',X'A')
Alphanumeric	1 or 2 Alpha or Special Characters ('AB', 'B', '\$')
Literals	Not acceptable
Expressions	
Name	1–4 characters
Factor	Name, integer, or location counter designator (\$)
Designator	Two's complement (-), byte variable (/)
Operator	An addition (+) or subtraction (-) sign
Expression Evaluation	Absolute and Relocatable
<u>PSEUDO OPERATIONS (See Section 2-2)</u>	
<u>Symbol</u>	<u>Data Definition</u>
EQU	Equates a symbol to an expression
BYTE	Defines data to fill two bytes
DATA	Defines data to fill one word
RES	Reserves block of storage
Assembler Control Directives	
ORIG	Sets location counter to specified value
END	Terminates assembly process
LOAD	Directs system loader to load specified programs at object time.

**APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY (Cont)**

K-1. SYM I/PREP CHARACTERISTICS (Cont)

<u>Symbol</u>	<u>Data Definition</u>
NTRY	Identifies entry point to program
LIBR	Generates ID labels for library routines
TRUE,FALS,ENDC	Specifies conditional processing of statements by assembler.
<u>ASSEMBLY LISTING FORMAT (hexadecimal)</u>	
Statement Diagnostic	Column 1 (see Appendix N-1)
Undefined Operands	Column 2 (Asterisk)
Location Counter	Columns 3–6
Memory Image of Assembled Word	Columns 9–12
Source Statement	Columns 16–72
<u>LOADER TEXT</u>	
Absolute	
Output	Paper tape core image only
Organization	1 byte per frame
Loaded By	XRAY EXEC or Initial Loader
Relocatable	
Output	Paper Tape, cards, mag tape, disc.
Record Blocking	94 bytes, check sum last byte
Loaded by	Relocatable Loader
<u>PREP MODE</u>	
I/O Device Assignment	
Source Input Units	Teletype PTR, HSPTTR, Keyboard
Binary Output Units	Teletype PTP, HSPTP
Operation Mode	
Assemble	Absolute, relocatable
Prep	Absolute, relocatable

**APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY (Cont)**

K-2. SYM II CHARACTERISTICS

SYM II

Operates Under System	STANDARD, EXTENDED, ADVANCED
Label	SYM II
Part Number	391878
No. of Passes	2
Intermediate Text	None
Pseudo-Operations (directives)	23
Conditional Directives	Yes (TRUE,FALS,ENDC)
Conversational Mode	No
Procedure Capability	Yes (PROC,ENDP)
Total Memory Required	4600 locations
Absolute Assembled Programs	450 symbols (8 K computer)
Relocatable Assembled Programs	450 symbols (8 K computer)
Source Input Units	Any Processor Logical Input Unit (PRIN)
Binary Output Units	Any Binary Logical Output Unit (BOUT)
Intermediate Units	
Mass Storage (source re-written)	Any Scratch Unit (SCRATCH)
Non-mass Storage	Any Processor Logical Input Unit (PRIN)
Symbolic Listing Suppression	Yes, Sense Switch 0 up or System Flag 0 Set
Binary Object Text Suppression	Yes, Sense Switch 1 up or System Flag 1 Set
Error list only	Yes, Sense Switch 0 and 2 up

STATEMENT FORMATS

Types of Statements	Title, Comment, Instruction
Coding Form Fields	4; Label, Operation, Operand, Comment
Label (name of location)	Any number of characters; 1st must be alphabetic

**APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY (Cont)**

K-2. SYM II CHARACTERISTICS (Cont)

Operation	3 or 4 letter codes (see Section 2)
Operand	Names, expressions, constants (see Figure 1-9).
Comment	
Delimiter	1 or more blank characters

SYNTAX

Character Set

Alphabetic	A through Z
Numeric	0 through 9
Special	+ - , . ' \$ "blank" / > < = *
Other	May appear in alphanumeric constants or comments

Constants

Decimal Integer	1–5 digits; 32,767 ($2^{15} - 1$) maximum value
Hexadecimal Integer	1–4 Hex digits (X'3F90',X'A')
Alphanumeric	1 or 2 alpha or special characters ('AB','B','\$')
Concatenated Integer	1 or more constants packed into sub-fields (3:4,X'FF': 8, six:3)
Double Precision Decimal Integer	1–10 digits; 1,073,741,823 ($2^{30} - 1$) max value
Real-Two Word Floating Point	1–7 digits; limits $\pm 10^{38}$
Mid-Precision Real-Three Word Floating Point	1–9 digits; limits $\pm 10^{38}$
Double Precision Real-Four Word Floating Point	1–13 digits; limits $\pm 10^{38}$

Literals Allowed (ED + 47)

Expressions

Name	1–4 characters
Factor	Name, integer, or location counter designator, (\$)

**APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY (Cont)**

K-2. SYM II CHARACTERISTICS (Cont)

Designator	Two's complement (-), byte variable (/)
Operator	An addition (+) or subtraction (-) sign
Expression Evaluation	Absolute and Relocatable
<u>PSEUDO OPERATIONS (See Section 2-2)</u>	
<u>Symbol</u>	<u>Data Definition</u>
DATA	Defines data to fill one word
BYTE	Defines data to fill two bytes
TEXT	Defines alphanumeric data to fill variable number of words
DPI	Defines double precision decimal integer
REAL	Defines single precision floating point real number
EPRL	Defines mid-precision floating point real number
DPRL	Defines double precision floating point real number
EQU	Permanently equates a symbol to an expression
IS	Temporarily equates a symbol to an expression
RES	Reserves a block of storage
Assembler Control Directives	
ORIG	Sets location counter to specified value
END	Terminates assembly process
LOAD	Directs system loader to load specified programs at object time
NTRY	Identifies entry point to program
LIBR	Generates ID labels for library routines
TRUE,FALS,ENDC	Specifies conditional processing
DO	Provides for repetitive code generation

**APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY (Cont)**

K-2. SYM II CHARACTERISTICS (Cont)

PROC, ENDP	Defines procedure (macro)
SUBR, EXIT	Defines entry and exit from subroutines.

ASSEMBLY LISTING FORMAT (hexadecimal)

Statement Diagnostic	Column 1 (see Appendix N-2)
Undefined Operands	Column 2 (asterisk)
Location Counter	Columns 3-6
Memory Image of Assembled Word	Columns 9-12
Source Statement	Columns 16-72

LOADER TEXT

Absolute	
Output	Paper tape or card core image
Organization	1 byte per frame, column binary
Loaded by	X-Ray EXEC (AL directive)
Relocatable	
Output	Paper tape, cards, mag tape, disc
Record Blocking	94 bytes, checksum last byte
Loaded by	Relocatable Loader

FORTTRAN LABELED DATA BLOCKS

Purpose	Allows creation, initialization and referencing of FORTRAN labeled data blocks within SYM II
Additional Pseudo Ops	LABL, DFLI, DFLD, DFLR, DFLM, DFLH

APPENDIX K

USERS SOFTWARE CHARACTERISTICS SUMMARY

K-3. RAYTHEON 706 FORTRAN VS USASI STANDARD

USASI Standard Section Number	Feature	USASI BASIC FORTRAN	USASI FORTRAN	CONV FORTRAN	Real-Time FORTRAN IV
.3 Program Form	Character set	Yes	Yes	Yes	Yes
	A-Z, 0-9	Yes	Yes	Yes	Yes
	Blank = + - * / 0 , .	No	Yes	No	Yes
	\$	72	72	72	72
	Line size, characters	5	19	19	unlimited
	Continuation lines	1-4	1-5	1-5	1-5
	Statement label size, characters	1-5	1-6	1-5	unlimited
.4 Data Types	Variable name size, characters				
	Integer	Yes	Yes	Yes	Yes
	Real	Yes	Yes	Yes	Yes
	Double Precision	No	Yes	No	Yes
	Complex	No	Yes	No	Yes
	Logical	No	Yes	No	Yes
.6 Expressions	Hollerith	No	Yes	Yes	Yes
	Mixed mode arithmetic:				
	Integer exponents with any data type	Yes	Yes	Yes	Yes
	Real with Double Precision	No	Yes	No	Yes
	Real with Complex (except exponentiation)	No	Yes	No	Yes
	Other	No	No	No	Yes
	Relational	No	Yes	No	Yes
.7 Statements	Logical	No	Yes	No	Yes
	Executable	3	3	3	3
	Assignment	1	3	2	3
	Arithmetic	Yes	Yes	Yes	Yes
	Logical	No	Yes	No	Yes
	GO TO Assignment	No	Yes	No	Yes
	Control	7	8	7	8
	GO TO	2	3	2	3
	Unconditional	Yes	Yes	Yes	Yes
	Assigned	No	Yes	No	Yes
	Computed	Yes	Yes	Yes	Yes
	Arithmetic If	Yes	Yes	Yes	Yes
	Logical If	No	Yes	No	Yes
	CALL Yes	Yes	Yes	Yes	Yes
	RETURN	Yes	Yes	Yes	Yes
	CONTINUE	Yes	Yes	Yes	Yes
	Program Control	Yes	Yes	Yes	Yes
	DO	Yes	Yes	Yes	Yes
	Extended DO	No	Yes	No	Yes
	Input/Output	2	2	2	2
	Read/Write (formatted and unformatted)	Yes	Yes	Yes	Yes
	Auxiliary I/O				
	REWIND	Yes	Yes	Yes	Yes
BACKSPACE	Yes	Yes	Yes	Yes	
END FILE	Yes	Yes	Yes	Yes	

APPENDIX K

USERS SOFTWARE CHARACTERISTICS SUMMARY (Cont)

K-3. RAYTHEON 706 FORTRAN VS USASI STANDARD (Cont)

USASI Standard Section Number	Feature	USASI BASIC FORTRAN	USASI FORTRAN	CONV FORTRAN	Real-Time FORTRAN IV
	Non-executable	4	5	4	5
	Specification	3	5	4	5
	Array declarator				
	Subscripts, number and type	2 integers or constants	3 integers, constants, or expressions	2 integers, constants, or expressions	N-expressions of any data type
	Adjustable dimension	No	Yes	No	Yes
	DIMENSION	Yes	Yes	Yes	Yes
	COMMON	Yes	Yes	Yes	Yes
	Blank	Yes	Yes	Yes	Yes
	Named	No	Yes	No	Yes
	Array Size Declared	No	Yes	No	Yes
	EQUIVALENCE	Yes	Yes	No	Yes
	EXTERNAL	No	Yes	No	Yes
	Type	No	Yes	No	Yes
	DATA	No	Yes	No	Yes
	FORMAT				
	Field descriptors	5	9	5	11
	Repeat extended grouping	No	Yes	No	Yes
	Scale factor	No	Yes	No	Yes
	Numeric conversions	2	4	2	42
	Integer	Yes	Yes	Yes	Yes
	Real	Yes	Yes	Yes	Yes
	Double Precision	No	Yes	No	Yes
	Complex	No	Yes	No	Yes
	Logical conversion	No	Yes	No	Yes
	Hollerith field descriptor	Yes	Yes	Yes	Yes
	Blank field descriptor	Yes	Yes	Yes	Yes
	Format specification in arrays	No	Yes	No	Yes
.8 Procedures and Subprograms	Intrinsic Functions	4	15	9	15
	External Functions	7	9	7	9
	CALL statement (user subroutines)	Yes	Yes	Yes	Yes
	Block Data subprogram	No	Yes	No	Yes

APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY (Cont.)

K-4. REAL-TIME FORTRAN IV CHARACTERISTICS

GENERAL

Operates under System	BASIC, STANDARD, EXTENDED, ADVANCED
Label	CF, CFR
Part Number	
Compiler	393295, 394005
Run-Time Library	394001, 394002, 394003, 394004
PTIOS (Basic System)	393954
No. of Passes	1 + 2 SYM II passes
Memory Required	
For 8 K Systems	6400 locations (no DATA or EQUIVALENCE)
For 12 K or Larger Systems	7500 locations
Automatic Compile and Go	Yes
Mixed Mode Expressions	Allowed
Array Dimensions	Any Number (N) allowed
Expressions in Output Lists	Allowed
Relational Operators	6 (LT, LE, EQ, NE, GE, GT)
Logical Operators	4 (NOT, AND, OR, EOR)
Real-Time Features	
Re-entrant and recursive Operation	Yes
In-Line Symbolic Code	Allowed
Program Segmentation	Allowed (with RTOS or MPS)
Connection of tasks to priority Interrupts	Allowed (with RTOS or MPS)
Dynamic Queuing of Programs	Allowed (with RTOS or MPS)

APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY (Cont)

K-4. REAL-TIME FORTRAN IV CHARACTERISTICS (Cont)

Statements	
Executable	15
Non-Executable	17
Library Functions	24 + 13 special array functions
Error Messages	68 compile time, 13 run-time (see appendices N-3, N-4)

STATEMENT NAMES

<u>Non-Executable</u>	<u>Executable</u>
COMPLEX	
DOUBLE-PRECISION	RETURN
MID-PRECISION	GO TO
REAL	IF
DOUBLE-INTEGER	ASSIGN
INTEGER	CONTINUE
LOGICAL	CALL
COMMON	DO
DIMENSION	PAUSE
EQUIVALENCE (12 K version)	STOP
EXTERNAL	REWIND
FUNCTION	BACKSPACE
SUBROUTINE	END FILE
BLOCK DATA	READ
DATA (12 K version)	WRITE
FORMAT	PRINT
END	

APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY (Cont.)
K-4. REAL-TIME FORTRAN IV CHARACTERISTICS (Cont.)

LIBRARY FUNCTIONS

Absolute value	Exponential
Truncation	Natural Logarithm
Choosing Largest Value	Common Logarithm
Choosing Smallest Value	Trigonometric Sine
Positive Difference	Trigonometric Cosine
Complex Modules	Hyperbolic Tangent
Remaindering	Square Root
Convert Integer to Real	Arc Tangent
Convert Real to Integer	Transfer of Sign
Obtain Most Significant Part of Double Precision Argument	
Obtain Real Part of Complex Argument	
Obtain Imaginary Part of Complex Argument	
Express Single Precision Argument in Double Precision Form	
Express Two Real Arguments in Complex Form	
Obtain Conjugate of Complex Argument	

SPECIAL ARRAY FUNCTIONS

Fast Fourier Transform	Array Add
Convolution Integral	Array Subtract
Array Complex Multiply	Array Shift (single, double)
Array Complex Spectral Magnitude	Clear Array
Array Real Multiply	Move and Convert Array
Order Array	

STORAGE ALLOCATION

Data Type	Words	Date Type	Words
Logical	2	Mid-Precision	4
Integer	2	Double-Precision	4
Double Integer	2	Complex	4
Real	2		

**APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY (Cont)**

K-5. CONVERSATIONAL FORTRAN CHARACTERISTICS

GENERAL

Operates under System	BASIC, STANDARD, EXTENDED, ADVANCED
Label	CF. CFR
Part Number	
Compiler	393295, 294005
Run-Time Library	394001, 394002, 394003, 394004
PTIOS (Basic System)	393954
No. of Passes	1
Memory Required	3050 locations
No. of Statements Compiled	
4 K Computer	100
8 K Computer	500
12 K Computer	900
Automatic Compile and Go	Yes
Array Dimensions	1 or 2
Conversational Mode	Yes (PREP)
Conversational Syntax Error Identification	Yes
Trace Mode	Yes
Real-Time Features	
In-Line Symbolic Code	Allowed in Subroutines
Connection of Tasks to Priority Interrupts	Allowed by Subroutines
Statements	
Executable	14
Non-Executable	5
Library Functions	16
Error Messages	
Compile Time	20 (see Appendix N-5)
Run-Time	7 (see Appendix N-6)

APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY (Cont)

K-5. CONVERSATIONAL FORTRAN CHARACTERISTICS (Cont.)

STATEMENT NAMES

Non-Executable

COMMON
DIMENSION
LOCATE
FORMAT
END

Executable

RETURN	STOP
GO TO	REWIND
IF	BACKSPACE
CONTINUE	END FILE
CALL	READ
DO	WRITE
PAUSE	PRINT

LIBRARY FUNCTIONS

Logical AND
Logical OR
Logical EOR
Logical NOT
Absolute Value
Transfer of Sign
Exponential
Natural Logarithm

Trigonometric Sine
Trigonometric Cosine
Hyperbolic Tangent (not 4 K Systems)
Square Root
Arc Tangent
Test Sense Switch
Convert Integer to Real
Convert Real to Integer

**APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY (Cont.)**

K-6.BASIC XRAY EXECUTIVE PROGRAM & I/O MONITOR CHARACTERISTICS

<u>XRAY EXEC - BASIC</u>	
Purpose	To aid the user in preparation, loading, execution and debugging of programs.
Core Memory Required	930 decimal locations (includes both XRAY and I/O Monitor).
Part Number	390779
Directives	12
Punch	Outputs on paper tape consecutive memory locations.
Absolute Load	Loads an absolute paper tape into memory.
Dump	Prints contents of an area of memory.
Hex Correction	Stores different hexadecimal numbers into consecutive memory cells.
Blank Buffer	Stores single data value into area of memory.
Date	Stores 8 characters into monitor date locations.
Identification	Stores 8 characters into monitor name locations.
Transfer	Transfers program execution to a location in memory.
Reassign I/O Device	Reassigns devices in the Peripheral Equipment Assignment Table (PEAT).
Initialize Load	Begins relocatable load from BIN device.
Continue Load	Continues loading relocatable program without destroying Entry Names table.
Entry Names Table Printout	Prints contents of Entry Names table.
<u>I/O MONITOR</u>	
Purpose	To provide the user with a flexible, file-oriented input/output system (see Sections 5-4 and 8-4. 11)
Features	
Device Reassignment	Yes
Simultaneous I/O Operation	Yes
End-Action	Yes
I/O Macros	Yes; OPEN,STAT,DOIO
Core Memory Required	Part of XRAY (930 decimal locations total – Drivers require additional memory).

APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY (Cont.)

K-7. PAPER TAPE INPUT/OUTPUT SYSTEM (PTIOS) CHARACTERISTICS

Purpose	To facilitate the use of Conversational FORTRAN in 4K of memory without having to use the BASIC Monitor System.
Label	PTIOS
Part Number	393954
Loaded By	Initial Loader
Restrictions	No end action is supported No special format (unformatted I/O) FIOT must specify binary or alpha record format for input as well as output. Record will be read in specified mode. No "OPEN" call may be made. Punch Leader is not available. "Write End-of-File" only magnetic tape call. The characters whose ASCII codes are X'80'-X'8D' may not appear in an alpha output buffer. There is no "RUBOUT" or reverse arrow for alpha input.
Logical Units Available	0 - 7
Logical Units Assignment	
Directive Input	1
Source Statement Input	2
Absolute Loader Binary Input	4
Binary Output/Prep Output	5
I/O Assignments	
Initial	Teletype
Optional	HSPTR,HSPTP
Directives	
Absolute Load	AL
Assign I/O	N:T, N:R, N:P where N = 0,...,7

APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY (Cont.)

K-8. SYMBOLIC AND SYSTEM EDITOR CHARACTERISTICS

Characteristics	Symbolic Program Editor-BASIC	Symbolic Program Editor	System Editor
Operates Under System	BASIC	STANDARD, EXTENDED, ADVANCED	EXTENDED, ADVANCED
Source Code	SYM-II	SYM-II	SYM-II
Object Code	Absolute Binary	Absolute Binary	Absolute Binary
Label	SYMEDB	SYMED	EDITOR
Part Number	390941	394873	391915
Locations	1033 decimal words	1013 decimal words	2500 decimal words
Input Devices	All paper tape, mag tape, card reader	All paper tape, mag tape, card reader, disc	Any input device (LUN 8)
Output Devices	All paper tape, mag tape, card reader	All paper tape, mag tape, card reader, disc	Any output device (LUN 9)
Edit Control	By line number	By line number	By files, lines, characters
Modes	Operator only (all directives input through XRAY Monitor)	Operator only (all directives input through XRAY Monitor)	Operator Mode (TTY keyboard) Batch Mode (through LUN 7)
Error Conditions Detected	Core Exhausted ('OV') Incorrect Directive Input ('Q?')	Core Exhausted ('OV') Incorrect Directive Input ('Q?')	20 Total
Directives	5; Insertion, detection, replacement, begin edit, return to XRAY	5; Insertion, detection, replacement, begin edit, return to XRAY	20 total; device assignment, rewind, write end-of-file, copy program, copy record, skip system program, skip program, skip record, plus misc. directives.

APPENDIX K
SOFTWARE CHARACTERISTICS SUMMARY (Cont.)

K-9. TRACE/DEBUG AND MPS DEBUG PACKAGE CHARACTERISTICS

Operates Under System	STANDARD, EXTENDED, ADVANCED
Source Code	SYM II
Object Code	Relocatable binary
Label	DEBUG
Part No.	391082, 394868
Locations	662 decimal words
Directives	12
Punch	Consecutive memory output on paper tape
Punch Execution Address	Terminates punching of absolute format tape
Punch Leader	Punches 5 inches of blank tape
Absolute Load	Loads absolute program from paper tape
Dump	Prints contents of an area of memory
Hexadecimal Correction	Alters memory
Blank Buffer	Stores single data value into area of memory
Transfer	Transfers execution to a location in memory
Reassign I/O Device	Reassigns devices in PEAT
Hexadecimal Adder	Adds and subtracts hexadecimal numbers
Masked Search Memory	Data word memory search
Trace (not with MPS)	Executes user's program and prints each step
Range	Complete or partial
Control	Optional HALT after each print out and selective instruction tracing

APPENDIX L

I/O OPERATING STATUS AND SELECTION SUMMARY

DEVICE I/O SELECTION CODES

Device	Digital Plotter	Disc Memory	DMA Magnetic Tape	ADC	DAC	Line Printer	Buffered Input Channel	Buffered Output Channel
y, x	0, x	1, x	2, x	3, x	4, x	5, x	6, x	7, x
DOT y, 0	Disconnect	Disconnect	Disconnect	Disconnect	Trans. Data	Disconnect		
↑ y, 1	Select	Mem. Addr.	Mem. Addr.	Reset Error	Trans. Data	Load Skew		Tr. and Disc. No. 1
y, 2		Tr. and Sector		Random, Unck.	Trans. Data			Tr. and Disc. No. 2
y, 3			Continue		Trans. Data			Tr. and Disc. No. 3
y, 4		No. Wds., Write	No. Wds., Write		Trans. Data	Load Char.		Tr. and Disc. No. 4
y, 5	Reset		Write EOF		Trans. Data			
y, 6		No. Wds., Read	No. Wds., Read	Seq., Unck.	Trans. Data			
y, 7		No. Wds., Verify	Erase		Trans. Data			
y, 8			Chain Addr.		Trans. Data	Print and Move		
y, 9			Chain Wd. Ct.		Trans. Data	Move Pap.		Tr. Ch. No. 1
y, A			Stop Chain	Random, Clk	Trans. Data			Tr. Ch. No. 2
y, B			Rewind		Trans. Data			Tr. Ch. No. 3
y, C			Backspace		Trans. Data			Tr. Ch. No. 4
y, D			Search EOF		Trans. Data			
y, E				Seq., Clk.	Trans. Data			
DOT y, F					Trans. Data			
DIN y, 0	Status	Status, No. 0	Status, No. 0	Status		Status	Status	Status
↑ y, 1		Status, No. 1	Status, No. 1	Trans. Data			In. Ch. 1	
y, 2		Status, No. 2	Status, No. 2				In. Ch. 2	
y, 3		Status, No. 3	Status, No. 3	Trans. and Start			In. Ch. 3	
↓ DIN y, 4			Read Mem. Addr.				In. Ch. 4	

APPENDIX L

I/O OPERATING STATUS AND SELECTION SUMMARY (Cont)

DEVICE I/O SELECTION CODES

Device	Card Reader	DIO Magnetic Tape	Teletype Multiplexer	Time-of-Day Clock	High Speed PT Punch	High-Speed PT Reader	ASR Teletype	Card Punch
y, x	8, x	9, x	A, x	B, x	C, x	D, x	E, x	F, x
DOT y, 0	Disconnect	Disconnect	Disconnect	Hold	Disconnect	Disconnect	Disconnect	Disconnect
y, 1	Read		Set In. Mode	Wr. Wd. 1		Read Fwd	Sel. PTR	Punch Cos.
y, 2	Stop	Write EOF	Set Out. Mode	Wr. Wd. 2	Pwr. On		Sel. PR/PU	Last Punch
y, 3	Stop Aft. Cd.	Write	Out. Data		Pwr. Off	Read Rev	Sel. KEYB.	Stop Punch
y, 4				Stop			Out. Char. and Disc	Load Data
y, 5				Reset				
y, 6				Start	Out. Char. and Pwr. On		Out. Char. and Print	
y, 7								
y, 8	Offset						Disconnect	Offset Card
y, 9		Read					Sel. PTR	Punch and Offset
y, A		Rewind					Sel. PR/PU	Last Punch and Offset
y, B		Backspace					Sel. KEYB	Stop Punch and Offset
y, C								Load Data and Offset
y, D								
y, E							Out. Char. (PR/PU)	
DOT y, F		Tr. Data						
DIN y, 0	Status	Status, No. 0	Status	Rd. Wd. 1	Status	Status	Status	
y, 1		Status, No. 1	In. Data	Rd. Wd. 2				
y, 2		Status, No. 2		Rd. Wd. 3				
y, 3		Status, No. 3				Tr. Char. and Disconnect		
y, 4	Tr. Data					Tr. Char and Read Fwd	In. Char. and Disc.	
y, 5	Tr. Data						In. Char. (PTR)	
y, 6								
y, 7							In. Char. (KEYB)	
y, 8								
y, 9								
y, A								
y, B								
y, C							In. Char. and Disc.	
y, D							In. Char. (PTR)	
y, E								
DIN y, F		Tr. Data					In. Char. (KEYB)	

APPENDIX L

I/O OPERATING STATUS AND SELECTION SUMMARY

DEVICE I/O OPERATING STATUS¹

Accumulator Bit	Card Reader	DIO Magnetic Tape	Teletype Multiplexer	Time-of-Day Clock	Paper Tape Punch	Paper Tape Reader	ASR Teletype	Card Punch
0	Controller or Device Not Ready	Controller or Device Not Ready	Channel Waiting		Controller Not Ready	Controller Not Ready	Teletype Selected	Controller or Device Not Ready
1	Controller Not Ready	Device Not Ready	Unwanted Message		Device Not Ready	Device Not Ready		Controller Not Ready
2	Card in Motion	BOT						
3	Hopper Empty	EOT						
4		Rewinding						
5								
6		Function Interrupt						
7	Interrupt	Data Interrupt			Interrupt	Interrupt	Interrupt	Interrupt
8								
9								
10								
11		EOF						
12								
13	Write Enabled							Punch Check Error
14	Response Error	Rate Error						Response Error
15	Response Error or Malfunction	Parity CRC or Rate Rate Error	Controller Not Ready					Response Error or Malfunction

¹Meaning when Bit "True," "ON," or a Binary "1."

APPENDIX L

I/O OPERATING STATUS, AND SELECTION SUMMARY (Cont)

DEVICE I/O OPERATING STATUS¹

Accumulator Bit	Digital Plotter	Disc Memory	DMA Magnetic Tape	ADC	DAC	Line Printer	Buffered Input Channel	Buffered Output Channel
0	Controller Disconnected	Controller or Device Not Ready	Controller or Device Not Ready	Device Busy		Controller or Device Not Ready	Channel 1 Interrupt	Channel 1 Interrupt
1	Controller Power On		Device Not Ready			Device Not Ready	Channel 2 Interrupt	Channel 2 Interrupt
2			BOT				Channel 3 Interrupt	Channel 3 Interrupt
3			EOT				Channel 4 Interrupt	Channel 4 Interrupt
4			Rewinding					
5			Density 200 BPI					
6			Density 556 BPI					
7	No Interrupt		Format 1		No Status Req'd	Interrupt		
8			Format 2					
9			Format 3					
10			Cont. Was Last Op.					
11			EOF					
12			Byte Count Odd					
13		Write on Protected Trade	Write Enabled					
14		Rate Error	Rate Error	Frame Sync.				
15		Rate or CRC Error	Parity, CRC or Rate Error	Error				

¹Meaning when Bit "True," "ON," or a Binary "1."

APPENDIX M

USER'S SOFTWARE OPERATING SUMMARY AND FUNCTIONAL CHARACTERISTICS

Introduction

Appendix M-1 identifies by part number the primary programs provided with the various hardware configurations. Hardware configurations are further defined as to memory size (i.e., 4K, 8K–32K) and software or hardware multiply/divide options.

Appendix M-2 is a review of all XRAY operating

system directives. Preceding this appendix is a dictionary arranged alphabetically by directive and indicating the applicable hardware system, the function, and the associated page number. Appendix M-2A is for XRAY BASIC. Appendix M-2B includes STANDARD, EXTENDED, and ADVANCED system directives. Appendix M-2B is arranged by directive type: job specification, I/O device management, assembly and compilation, resident program, interrupt system control and list directives.

APPENDIX M-1A
BASIC SYSTEM SOFTWARE PART NUMBERS

Description	4K Core Memory		8K-32K Core Memory	
	Soft MPY/DIV	Hard MPY/DIV	Soft MPY/DIV	Hard MPY/DIV
XRAY EXEC-BASIC ¹	390779	390779	390779	390779
Relocating Loader-BASIC	390682	390682	390682	390682
SYM-I/PREP	390470	390470	390470	390470
Symbolic Prog. Editor-BASIC	390941	390941	390941	390941
Initial Loader-Paper Tape	393260	393260	393260	393260
Initial Loader-Cards	393259	393259	(2)	(2)
Conver. FORTRAN Library	394002	394001	394004	394003
Conver. FORTRAN Compiler	393295	393295	394005	394005
Paper Tape I/O System ³	393954	393954	393954	393954
Math Library	392325	393977	392325	393977

Notes: BASIC Configuration –
No High Speed Input Device

1. Monitor Included
2. No card version available because card reader with 8K-32K core memory implies STANDARD operating system.
3. Includes Initial Loader

APPENDIX M-1B
STANDARD OPERATING SYSTEM SOFTWARE PART NUMBERS

Description	8K Core Memory		12K-32K Core Memory	
	Soft MPY/DIV	Hard MPY/DIV	Soft MPY/DIV	Hard MPY/DIV
Monitor	(1)	(1)	(1)	(1)
XRAY EXEC-STANDARD	391305	391305	391305	391305
Relocating Loader-Standard	390013	390013	390013	390013
SYM-I/PREP	390470	390470	390470	390470
SYM-II	391878	391878	391878	391878
Trace/Debug Package	391082	391082	391082	391082
Symbolic Program Editor	394873	394873	394873	394873
Absoluter, Linking-STANDARD	393254	393254	393254	393254
Conver. FORTRAN Library	394004	394003	394004	394003
Conver. FORTRAN Compiler	394005	394005	394005	394005
FORTRAN IV Run-Time Library	393945	393944	393945	393944
FORTRAN IV Compiler	393297	393297	392957	392957
Math Library	392325	393977	392325	393977
Initial Loader-Paper Tape	393260	393260	393260	393260
Initial Loader-Cards	393259	393259	393259	393259

Notes: STANDARD Configuration -- High Speed
Input Device and 8K-32K Core Memory

1. Monitor is custom prepared for each installation according to peripheral device configuration. See shipping manifest for part number.

APPENDIX M-1C

MAGNETIC TAPE OPERATING SYSTEM SOFTWARE PART NUMBERS

Description	8K Core Memory		12K-32K Core Memory	
	Soft MPY/DIV	Hard MPY/DIV	Soft MPY/DIV	Hard MPY/DIV
Mag Tape XRAY EXEC	393979	393979	393979	393979
Mag Tape SYSGEN	393992	393992	393992	393992
Mag Tape QPROC	393980	393980	393980	393980
Monitor	(1)	(1)	(1)	(1)
Relocating Loader-Standard	390013	390013	390013	390013
SYM-I/PREP	390470	390470	390470	390470
SYM-II	391878	391878	391878	391878
Trace/Debug Package	391082	391082	391082	391082
Symbolic Program Editor	394873	394873	394873	394873
System Editor	391915	391915	391915	391915
Absoluter, Linking-Standard	393254	393254	393254	393254
Conver. FORTRAN Library	394004	394003	394004	394003
Conver. FORTRAN Compiler	394005	394005	394005	394005
FORTRAN IV Run-Time Library	393945	393944	393945	393944
FORTRAN IV Compiler	393297	393297	392957	392957
Math Library	392325	393977	392325	393977
Initial Loader-Paper Tape	393260	393260	393260	393260
Initial Loader-Cards	393259	393259	393259	393259
Magnetic Tape Dump	390540	390540	390540	390540

Notes: Magnetic Tape Operating System –
 High Speed Input Device, 8K-32K
 Core Memory, minimum 1 magnetic
 tape unit.

1. Monitor is custom prepared for each installation according to peripheral device configuration. See shipping manifest for part number.

APPENDIX M-1D
EXTENDED OPERATING SYSTEM SOFTWARE PART NUMBERS

Description	8K Core Memory		12K-32K Core Memory	
	Soft MPY/DIV	Hard MPY/DIV	Soft MPY/DIV	Hard MPY/DIV
Monitor	(1)	(1)	(1)	(1)
RTOS-Real Time XRAY	391881	391881	391881	391881
Relocating Loader-Disc	390012	390012	390012	390012
Initial Loader-Paper Tape	393260	393260	393260	393260
Initial Loader-Cards	393259	393259	393259	393259
Queue Processor	391879	391879	391879	391879
Trace/Debug Package	391082	391082	381082	381082
SYM-I/PREP	390470	390470	390470	390470
SYM-II	391878	391878	391878	391878
RTOS Disc Bootstrap	391917	391917	391917	391917
Symbolic Program Editor	394873	394873	394873	394873
System Control	391915	391915	391915	391915
RTOS Sysgen 1	391876	391876	391876	391876
RTOS Sysgen 2	391877	391877	391877	391877
Resident Loader	391916	391916	391916	391916
Library Extension Processor	391914	391914	391914	391914
RTOS Installation Program	392341	392341	392341	392341
Disc Dump	390539	390539	390539	390539
Absoluter, Linking-Disc	393255	393255	393255	393255
Magnetic Tape Dump	390540	390540	390540	390540
Card Sequencer	392920	392920	392920	392920
FORTTRAN IV Run-Time Library	393945	393944	393945	393944
FORTTRAN IV Compiler ²	393297	393297	392957	392957
Conver. FORTTRAN Library	394004	394003	394004	394003
Conver. FORTTRAN Compiler	394005	394005	394005	394005
Math Library	392325	393977	392325	393977

Notes: Extended Configuration – High Speed Input Device, 8K-32K Core Memory and Disc Memory.

1. Monitor is custom prepared for each installation according to peripheral device configuration. See shipping manifest for part number.
2. 2-Piece FORTTRAN IV Compiler for 8K system.

APPENDIX M-1E

ADVANCED OPERATING SYSTEM PART NUMBERS

Description	16K-32K Core Memory	
	Soft MPY/DIV	Hard MPY/DIV
Monitor	(1)	(1)
MPS Foreground EXEC	394862	394862
MPS Background EXEC	394863	394863
MPS SYSGEN	394864	394864
MPS Absolute Loader	394869	394869
MPS Error Processor	394929	394929
MPS Relocating Loader	394867	394867
Initial Loader-Paper Tape	393260	393260
Initial Loader-Cards	393259	393259
MPS Debug Package	394868	394868
MPS Disc Bootstrap	394864	394864
MPS Resident Loader	394870	394870
Disc Dump	390539	390539
Absoluter, Linking-Disc	393255	393255
Magnetic Tape Dump	390540	390540
Symbolic Program Editor	394873	394873
Card Sequencer	392920	392920
System Editor	391915	391915
SYM-I/PREP	390470	390470
SYM-II	391878	391878
FORTTRAN IV Run-Time Library	393945	393945
FORTTRAN IV Compiler	392957	392957
Conver. FORTRAN Library	394004	394003
Conver. FORTRAN Compiler	394005	394005
Math Library	392325	393977

Notes: Advanced Configuration — High Speed Input Device, 16K-32K Core Memory, Disc Memory, Time-of-Day Clock, and Operator Interrupt.

1. Monitor is custom prepared for each installation according to peripheral device configuration. See shipping manifest for part number.

APPENDIX M-2
 USER'S SOFTWARE OPERATING SUMMARY
 XRAY DIRECTIVES DICTIONARY

Directive	Available In System				MPS		Function	Page
	BASIC	STAND	MTOS	RTOS	Back	Fore		
A			X	X	X		Assemble with SYM II	M.32
AG			X	X	X		Assemble, Load, and Execute	M.36
AI				X	X		Arm Interrupts	M.43
AL	X	X	X	X	X		Load with Absolute Loader	M.10, M.19
BA			X	X	X		Enter Batch Mode	M.19
BB	X						Blank Buffer (Memory)	M.12
BM					X		Set Background Core Size	M.23
BT			X	X	X		Enter Temporary Batch Mode	M.19
CB				X	X	X	Create a Resident Common Block	M.42
CL	X						Continue Loading Relocatable Program	M.14
CO			X	X	X		Comment Record	M.20
CQ						X	Connect Queue	M.43
D	X						Dump Contents of Memory	M.10
DA	X	X	X	X	X	X	Set Date for Duration of Job	M.12, M.16
DB				X	X	X	Create a Resident Data Block	M.41
DC				X	X	X	Create Combined Resident Data and Common	M.42
DF				X	X	X	Set Disc File Pointers	M.30
DI				X	X	X	Disarm Interrupts	M.43
DJ			X	X	X	X	Delete Job	M.19
DL			X	X	X	X	Delete Priority Interrupt Level	M.18
DO					X	X	Dedicate I/O	M.31
DQ			X	X	X	X	Delete Task from Queue	M.17
EO(J)			X	X	X		End of Job	M.16
ET	X						Print Entry Names Table	M.14
EX			X	X	X	X	Queue XRAY and Exit	M.18
F			X	X	X		Compile with FORTRAN IV	M.35
FG			X	X	X		Compile, Load, and Execute	M.36
FO			X	X	X		Compile Only, Intermediate not Assemble	M.35
GO			X	X	X		Begin Execution for Assemble and Go And Compile and Go	M.36
H	X						Hexadecimal Correction	M.11
IC				X	X		Connect Task to Interrupt	M.43
ID	X	X	X	X	X		Set ID Data for Duration of Job	M.13, M.16
IL	X	X	X	X	X		Load with Relocating Loader	M.14, M.19

APPENDIX M-2

USER'S SOFTWARE OPERATING SUMMARY

XRAY DIRECTIVES DICTIONARY (Cont.)

Directive	Available In System				MPS		Function	Page
	BASIC	STAND	MTOS	RTOS	Back	Fore		
IO	X	X	X	X	X	X	Make I/O Assignment for Job Duration	M.13, M.28
LD					X	X	List Disc Vector	M.44
LL		X					Re-enter Loader in Library Phase	M.21
LP					X	X	List Peat Table and Mark Dedicated Devices	M.44
LQ					X		List Queue	M.44
LR						X	List Resident Tasks	M.44
MA			X	X	X		Return System to Operate (Manual) Mode	M.19
MP				X	X	X	Print Resident Entry Names Table	M.45
NJ			X	X	X		Next Job	M.16
OB			X	X	X		Load Object Text To or From Go Device	M.36
OP			X	X	X		Operator Message	M.20
OT					X	X	Output Time of Day	M.44
P	X						Punch Memory on Paper Tape	M.9
PC				X	X	X	Purge Resident Common Block	M.42
PD				X	X	X	Purge Resident Data Block	M.42
PF				X	X	X	Set Permanent Disc File Pointers	M.30
PI			X	X	X		Permanent I/O Directives	M.28
PL						X	List Periodic Task Table	M.45
PR					X		Pseudo Resident List	M.45
QP					X	X	Queue Periodic Tasks	M.23
QU			X	X	X	X	Queue Specified Task	M.17
RF				X	X		Set Recursive FORTRAN Run-Time Mode	M.21
RO					X	X	Release I/O	M.31
RS				X		X	Create Resident Subroutine	M.41
RT				X		X	Create Resident Task	M.41
RW			X	X	X	X	Rewind Magnetic Tapes	M.29
SF		X	X	X	X		Set System Flags	M.20
ST					X	X	Set Time-of-Day	M.26
T	X						Transfer and Execute	M.13
TD				X	X	X	Set Today's Date (Permanent)	M.20
UT	X						Load Specified Program from Library	M.21
WE		X	X	X	X	X	Write End-Of-File	M.29
XN		X					Make XRAY Non-Resident	M.22
XR		X					Make XRAY Resident	M.21
XX			X	X	X	X	Exit XRAY	M.18

APPENDIX M-2A

XRAY-BASIC SYSTEM DIRECTIVES

GENERAL

XRAY (PN 390779) is loaded by bootstrapping the Initial Loader (PN 393260 or PN 393259). XRAY is then placed in the paper tape reader or card reader and is loaded automatically. XRAY is entered by starting execution at location 4016. XRAY consists of a group of subroutines which process operator directives. Each directive is defined separately in the following sections; however, below is a list of characteristics common to all directives:

1. Directives are always two characters in length; i.e., DA, AL, etc.
2. Directives are received through logical unit

SYIN (the system input unit) by the I/O Monitor system; therefore, all directives must be preceded by a line feed and followed by a carriage return; e.g., L/F AL C/R.

3. All directive arguments are hexadecimal.
4. Leading zeroes need not be typed. Only the last 4 hex digits of any argument are used as input.
5. If the operator makes a mistake and types in something other than a legal directive, two question marks will be typed out (? ?) and device SYIN selected for another input.

EXECUTIVE DIRECTIVES

Form and Argument String	Use
<p>PUNCH P^NNNN,XXXX</p> <p>Argument String: Hexadecimal System: BASIC</p>	<p>The punch directive outputs onto paper tape the contents of a series of consecutive memory locations. The directive input format is:</p> <p style="text-align: center;">P^NNNN,XXXX</p> <p>where: NNNN is the beginning memory location, inclusive XXXX is one greater than the last location punched</p> <p style="text-align: center;">^ is a space</p> <p>The space follows the P because each XRAY directive must be two characters in length.</p> <p>The punch directive produces a tape format exactly as that output by an absolute assembly, except no program checksum is output by punch. See Section 8-5.1.1 for the paper tape format.</p>

APPENDIX M-2A

XRAY-BASIC SYSTEM DIRECTIVES

EXECUTIVE DIRECTIVES (Cont.)

Form and Argument String	Use
<p>HEX CORRECTION H^NNNN,XXXX,YYYY,---</p> <p>Argument String: Hexadecimal System: BASIC</p>	<p>The dump directive produces a tabulation on the system logical list unit (LIST). Each line consists of an address A, and the contents, in hexadecimal, of locations A through A+7. A complete line is always output. Below is an example of a dump beginning at location 300₁₆.</p> <pre>D 300, 310 0300 0000 90D6 7804 80D6 22E7 90D6 8800 E381 0308 7800 8806 0800 130F 903E 00B0 2800 803E 0310 9806 00B0 2800 903E 9800 1303 731D 631E</pre> <p>The DUMP subroutine also provides the user with a dynamic dump capability. The dump output format is exactly as described above, but the dump subroutine which is resident in XRAY is called from the users program. The calling sequence is as follows:</p> <pre>JSX DUMP DATA NNNN DATA XXXX</pre> <p>The program sequence causes the contents of memory location NNNN through XXXX to output on LIST. The users program then continues execution following the DUMP calling sequence. I/O Monitor cell 56₁₆ is the linkage to DUMP. The user must equivalence DUMP to cell 56₁₆ in his program.</p> <p>The hex correction directive stores different hex numbers into consecutive memory cells.</p> <p>The directive input format is as follows:</p> <pre>H^NNNN,XXXX,YYYY,ZZZ,---</pre> <p>where: NNNN is the beginning location XXXX is data to be stored YYYY is data to be stored ZZZZ is data to be stored etc.</p> <p>^ is a space</p>

APPENDIX M-2A
XRAY-BASIC SYSTEM DIRECTIVES

EXECUTIVE DIRECTIVES (Cont.)

Form and Argument String	Use
<p>BLANK BUFFER BBNNNN,XXXX,YYYY</p> <p>Argument String: Hexadecimal System: BASIC</p>	<p>XXXX is stored into location NNNN, YYYY is stored into location NNNN+1, etc. When NNNN is negative (a hex number from 8000 to FFFF) the program will begin storing the new data in the location following the last data word in the previous H directive. The directive cannot be longer than 54 characters, including the H, spaces, and commas. Below is an example of a 53 character input to store data starting at cell 400₁₆. The input character count starts with the H and ends with the last number typed.</p> <p style="text-align: center;">H 400,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19</p> <p>The blank buffer directive stores a single data value into an area of memory. The input format is as follows:</p> <p style="text-align: center;">BBNNNN,XXXX,YYYY</p> <p>where: NNNN is the beginning memory location XXXX is the data to be stored YYYY is the number of times to store the data</p> <p>Below is an example to store 0003 into 100₁₆ cells beginning at location 300₁₆.</p> <p style="text-align: center;">BB300,3,100</p>
<p>DATE DAXXXXXXXXXX</p> <p>Argument String: Alphanumeric System: BASIC</p>	<p>The data directive stores the 8 characters following DA into the monitor data locations. See the monitor documentation for the exact location of DATE. The directive format is as follows:</p> <p style="text-align: center;">DAXXXXXXXXXX</p> <p>where: X's will be stored into cells DATE through DATE+3</p> <p>Below is an example to store 11/28/67 into DATE to DATE+3:</p> <p style="text-align: center;">DA11/28/67</p>

APPENDIX M-2A

XRAY-BASIC SYSTEM DIRECTIVES

EXECUTIVE DIRECTIVES (Cont.)

Form and Argument String	Use
<p>IDENTIFICATION IDXXXXXXXX</p> <p>Argument String: Alphanumeric System: BASIC</p>	<p>The identification directive stores the first 8 characters following ID into the monitor program name locations. The directive input format is as follows:</p> <p style="text-align: center;">IDXXXXXXXX</p> <p>where: X's will be stored in NAME to NAME+3</p> <p>Below is an example to store COMP PRO into cells NAME to NAME+3:</p> <p style="text-align: center;">IDCOMP PROG</p>
<p>TRANSFER T^XXXX</p> <p>Argument String: Hexadecimal System: BASIC</p>	<p>The transfer directive transfers execution to a location in memory. The directive input format is as follows:</p> <p style="text-align: center;">T^XXXX</p> <p>where: XXXX is the location to begin execution</p> <p style="text-align: center;">^ is a space</p> <p>Below is an example to start execution at cell 300₁₆ :</p> <p style="text-align: center;">T 300</p>
<p>REASSIGN I/O DEVICE I/OXX,YY</p> <p>Argument String: Decimal System: BASIC</p>	<p>The I/O directive reassigns devices in the Peripheral Equipment Assignment Table (PEAT). The directive input format is as follows:</p> <p style="text-align: center;">IOXX,YY</p> <p>where: XX is the first Logical Unit Number (LUN) YY is the second Logical Unit Number</p> <p>XX must always be less than 12₁₀ . The I/O directive assigns XX to YY. Below is an example assigning LUN 2 to the same device as LUN 12:</p> <p style="text-align: center;">I02,12</p>

APPENDIX M-2A

XRAY-BASIC SYSTEM DIRECTIVES

RELOCATABLE LOADER DIRECTIVES

Form and Argument String	Use
<p>INITIALIZE LOAD IL</p> <p>Argument String: None System: BASIC</p>	<p>This directive begins a relocatable loading operation. The directive input format is as follows:</p> <p style="text-align: center;">IL</p> <p>There are no input arguments for IL.</p> <p>The operator places the text to be loaded in the BIN device, types the directive IL, then turns the reader on. The loader clears the entry name table, resets the storage map, and loads the text. As entry names are encountered in loading, they are placed into the entry names table. Loading ceases when an END statement is detected in the text. If Sense Switch 0 is true, the loader will load another program from BIN. If Sense Switch 0 is false, control will be returned to XRAY for further directives.</p>
<p>CONTINUE LOAD CL</p> <p>Argument String: None System: BASIC</p>	<p>This directive continues loading relocatable programs without destroying the entry names table or programs already loaded.</p> <p>The directive input format is as follows:</p> <p style="text-align: center;">CL</p> <p>There are no input arguments for CL.</p> <p>The operation and functions of the CL directive are identical to those of the IL directive with the exceptions that the entry names table is not cleared and the storage map is not reset.</p>
<p>ENTRY NAMES TABLE PRINTOUT ET</p> <p>Argument String: None System: BASIC</p>	<p>This directive prints the contents of the entry names table generated by the loading of programs.</p> <p>The input format is as follows:</p> <p style="text-align: center;">ET</p> <p>There are no input arguments for ET.</p>

APPENDIX M-2A

XRAY-BASIC SYSTEM DIRECTIVES

RELOCATABLE LOADER DIRECTIVES (Cont.)

Form and Argument String	Use
	<p>Each entry name which has been encountered since the last IL directive will be listed along with its location in core.</p> <p>Undefined names will be given a location of zero. Names referenced only by a LOAD pseudo-op but not loaded will be printed with a location value of 3FFF and flagged by an arrow.</p> <p>If a name is defined more than once, the location value of the first encounter will fill all references to that name. All subsequent definitions of that name will be printed, with their locations, but flagged by an arrow.</p> <p>The directive ET also initiates execution of the Symbolic Program Editor and the Trace package. The directive is used according to the program (Loader, Editor, or Trace) which has been loaded.</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

JOB SPECIFICATION

Form and Argument String and Operating System	Use
<p>NEXT JOB :NJ OR :EOJ</p> <p>Argument String: None Systems: MTOS, RTOS, MPS Background</p>	<p>The NJ directive introduces a new job. The EOJ directive declares the end of the current job. These directives cause a top of form record to be written on the LIST device, and the message</p> <p style="text-align: center;">NEXT</p> <p>will appear on the list device and on the console teletype for job logging purposes. In addition, the PEAT table and disk file pointers will be reset to their permanent configuration. "today's date" will be reset into the monitor DATE cells, and the monitor ID cells will be set to blanks. This assures the user a standard configuration for his job. If the system is in the temporary batch mode, it will return to the operator mode when the EOJ or NJ directive is encountered. The NJ directive should always introduce the job, as it will protect the job against errors caused by the previous jobs neglecting to supply an EOJ, and will aid the operator in stacking and unstacking the job stack.</p>
<p>IDENTIFICATION :ID,XXXXXXXX</p> <p>Argument String: Fixed-length Alphabetic Systems: STANDARD, MTOS, RTOS, MPS Background</p>	<p>The ID directive stores the eight or fewer character alphabetic argument into the monitor cells reserved for ID names. This name is printed on listing heading output by certain processors. The specified identification must begin with an alphabetic character. If the number of characters is less than 8, the name is stored left justified with trailing blanks supplied.</p>
<p>DATE :DA,XXXXXXXX</p> <p>Argument String: Fixed-length Alphabetic Systems: STANDARD, MTOS, RTOS, MPS Background, MPS Foreground</p>	<p>The date directive stores the eight or fewer character alphabetic into the monitor cells reserved for the date. This is printed on listings output by certain processors. To use a numeric date of the form 11/25/68, it must be input with a leading /, thus</p> <p style="text-align: center;">:DA,/11/25/68</p> <p>A comma cannot be used as part of the date, since comma is the string delimiter. Thus</p> <p style="text-align: center;">:DA,APR.3,68</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

JOB SPECIFICATION (Cont.)

Form and Argument String and Operating System	Use
<p>TASK QUEUE :QU,PROGNAME,PRIORITY, ARG1,ARG10</p> <p>Argument String: Variable length alpha-numeric</p> <p>System: MTOS, RTOS, MPS Background, MPS Foreground</p>	<p>is <i>not</i> a valid date directive.</p> <p>In MPS, :DA is equivalent to :TD directive.</p> <p>Task PROGNAME is placed in the queue at priority level PRIORITY. Up to 10 arguments may be specified. If no priority is given, the task will be executed at level 128. However, priority must be included if ARG1 is present.</p> <p>ARG1, . . . , ARG10 may be alphabetic or numeric strings. If an argument is numeric, it is converted to an integer occupying a single word of the queue entry. If an argument is alphabetic, it occupies 4 words of the queue entry and is left justified with trailing blanks supplied. This directive may be used to place any task, located in the processor library or in the relocatable library, into the system queue thereby requesting its execution. The QU directive must be the last directive on its line, because of the variable length argument string. The last argument <i>must</i> be followed by a new line.</p>
<p>TASK DELETION :DQ,PROGNAME,PRIORITY, ARG1ARG10</p> <p>Argument String: Variable length alphanumeric</p> <p>System: MTOS, RTOS, MPS Background, MPS Foreground</p>	<p>Task PROGNAME is matched up to the last argument given with tasks in the queue, all matching tasks are deleted.</p> <p>This directive deletes a specific task or tasks from the queue. For example,</p> <p style="text-align: center;">:DQ,PROGNAME</p> <p>would delete all entries of PROGNAME from the queue. While</p> <p style="text-align: center;">DQ,PROGNAME,10</p> <p>would delete all entries of PROGNAME which are queued at priority 10.</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

JOB SPECIFICATION (Cont.)

Form and Argument String and Operating System	Use
<p>PRIORITY LEVEL DELETION :DL,N1,N2,...NN</p> <p>Argument String: Variable length Numeric</p> <p>System: MTOS, RTOS, MPS Background, MPS Foreground</p>	<p>N1...NN are positive integers <256. All tasks at priority levels N1,N2,...NN are deleted from the job queue. Since the delete requests are queued tasks, only a limited number of deletions can be made at a time, depending on the size of the queue area in the monitor. Observe that it is necessary to exit Real-Time XRAY before the task deletion task is executed, then any tasks queued at a level for which a delete is pending will also be deleted. For example the following sequence, intended to execute a program called DLIST, will <i>not</i> work.</p> <p style="padding-left: 40px;">:DL,5 :QU,DLIST,5 :EX</p> <p>instead the sequence must be</p> <p style="padding-left: 40px;">:DL,5 :EX :QU,DLIST,5 :EX</p>
<p>EXIT & QUEUE REAL-TIME XRAY</p> <p>:EX</p> <p>No Argument String</p> <p>System: MTOS, RTOS, MPS Background, MPS Foreground</p>	<p>The EX directive causes Real-Time XRAY to exit, after first queuing itself (RTXRAY) at level 128. This returns control of the system to the queue processor which will execute any tasks which may be present in the queue. Eventually, control will return to RTXRAY, when it becomes the highest entry in the queue. This directive is normally used to exit RTXRAY after the desired tasks have been set up.</p>
<p>EXIT REAL-TIME XRAY</p> <p>:XX</p> <p>No Argument String</p> <p>System: MTOS, RTOS, MPS Background, MPS Foreground</p>	<p>The XX directive exits RTXRAY when RTXRAY is not to queue itself. For example, this directive may be used when the responsibility for queuing RTXRAY is given to another program in the queue on some discretionary basis. Batch mode users must use this directive cautiously, since failure to re-queue RTXRAY will stop the processing of the batch stack.</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

JOB SPECIFICATION (Cont.)

Form and Argument String and Operating System	Use
<p>DELETE JOB :DJ</p> <p>No Argument String</p> <p>System: MTOS, RTOS, MPS Background, MPS Foreground</p>	<p>The DJ directive deletes level 128 and queues RTXRAY at priority level 0. After the directive DJ is entered, RTXRAY will type</p> <p style="text-align: center;">XRAY</p> <p>indicating that the current job has been deleted and control returned to the operator. Since all system tasks queue at level 128, this directive may be used to prevent the execution of queued but unexecuted system tasks.</p>
<p>RELOCATABLE LOAD :IL</p> <p>No Argument String</p> <p>System: STANDARD, MTOS, RTOS, MPS Background</p>	<p>This directive requests the system loader to load a program from the Binary Input (BIN) device (logical unit 4). Additional details of this operation are found in Section 8-5.2.6.</p>
<p>ABSOLUTE LOAD :AL</p> <p>No Argument String</p> <p>Systems: STANDARD, MTOS, RTOS, MPS Background</p>	<p>Control is transferred to the absolute loader in the monitor, to load an absolute program from the BIN device. The absolute loader computes and verifies the checksum of the program loaded. If a checksum error is detected, the message</p> <p style="text-align: center;">ACK!</p> <p>is output on the teletype and loading continues.</p>
<p>REAL-TIME XRAY MODE CONTROL</p> <p>:BA :BT :MA</p> <p>No Argument String</p> <p>Systems: MTOS, RTOS, MPS Background</p>	<p>These directives provide a means of switching the system input unit to an automatic input unit for batch mode operation, or switching it to the teletype for operator mode operation. The mode switch directive must be the last directive on its line. The BA and BT directives copy the assignment of the PRIN unit (logical unit 2) to the SYSI unit (logical unit 1). To specify from which physical unit the batch stack is to be read, the operator may assign logical unit 2 before giving the BA or BT directives. Thus, to batch from magnetic tape 1, the following directives are used:</p> <p style="text-align: center;">:IO,2,15,:BA</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

JOB SPECIFICATION (Cont.)

Form and Argument String and Operating System	Use
<p>SET SYSTEM FLAGS :SF,n1,n2,n3...</p> <p>Argument String: Variable length Numeric</p> <p>Systems: STANDARD, MTOS, RTOS, MPS Background</p>	<p>Batch mode specified by the BA directive remains in force until an MA directive is read or the system is bootstrapped from the disc. Batch mode specified by the BT directive remains in force until an EOJ or NJ directive is encountered.</p> <p>System Flags, used to transmit special options to system processors, are set by this directive. These flags are automatically reset at each entrance to XRAY. There are 12 such flags available. See Assembly and Compilation in this table.</p>
<p>TODAY'S DATE :TD,XXXXXXXX</p> <p>Argument String: Fixed Length Alphabetic</p> <p>Systems: RTOS, MPS Background, MPS Foreground</p>	<p>The TD directive stores the eight or fewer character alphabetic string on the disc, from which the :EOJ or :NJ directives will copy it into the monitor cells reserved for the date.</p> <p>The TD directive should be used by the system operator to set the current date each morning.</p>
<p>COMMENT CARD :CO, any string characters</p> <p>Argument String: Indeterminate</p> <p>Systems: MTOS, RTOS, MPS Background</p>	<p>The :CO card is simply printed on the LIST device and may be used to insert arbitrary identifying information in the output listing.</p>
<p>OPERATOR MESSAGE :OP, any string of characters</p> <p>Argument String: Indeterminate</p> <p>Systems: MTOS, RTOS, MPS Background</p>	<p>The :OP card is output on the teletype, and the teletype bell is sounded four times to attract the operator's attention. This directive permits batch mode users to communicate instructions to the machine operator.</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

JOB SPECIFICATION (Cont.)

Form and Argument String and Operating System	Use
<p>RECURSIVE FORTRAN :RF</p> <p>No Argument String</p> <p>Systems: RTOS, MPS Background</p>	<p>This directive sets a flag which directs the FORTRAN Run-Time Executive to operate in the recursive mode. In this mode subroutine arguments and temporaries are stacked in a dynamic pool, allowing FORTRAN subroutines to be called recursively. Once selected, this mode remains selected until NEXT job. A resident copy of the Run-Time Executive will operate recursively if the :RF directive was in force when the Run-Time Executive was first executed, even though the flag may be later reset by some batch mode :EOJ.</p>
<p>LIBRARY LOAD :LL</p> <p>Argument String: None</p> <p>System: STANDARD</p>	<p>This directive enters the relocatable loader in the binary phase. It may be used to force the loader to scan a second library tape to staisfy undefined references formed at the end of the first scan.</p>
<p>LOAD SPECIFIED PROGRAM FROM LIBRARY :UT,PROGNAME</p> <p>Argument String: Variable-length Alphabetic</p> <p>System: STANDARD</p>	<p>The relocating loader is directed to scan the system library for the program PROGNAME. If the requested program is found, it is loaded by the loader and executed. Of course, only main line programs in the library may be called in this way. Sub-programs have no execution address specified.</p>
<p>MAKE XRAY RESIDENT :XR</p> <p>Argument String: None</p> <p>System: STANDARD</p>	<p>XRAY is made resident, and the core space it occupies is unavailable. When a task exits through cell X'40', control returns to XRAY, which outputs</p>
	<p>NEXT</p> <p>and selects the system input device for a control directive. If the loader has been damaged — for instance, if an absolute processor such as SYM II has been loaded — it will be necessary to reload the loader before giving the IL, LL, or UT directives. If the loader is reloaded, XRAY should first be made non-resident. Any other XRAY directive may be given without reloading the loader.</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

JOB SPECIFICATION (Cont.)

Form and Argument String and Operating System	Use
<p>MAKE XRAY NON-RESIDENT :XN</p> <p>Argument String: None</p> <p>System: STANDARD</p>	<p>To load a program which requires the space occupied by XRAY or by the loader, XRAY must be made non-resident, the monitor load routine will type:</p> <p style="text-align: center;">LOAD</p> <p>and halt to allow the next processor to be placed in the reader.</p> <p style="text-align: center;">NOTE</p> <p>The XRAY and Relocating Loader are normally non-resident. In this way the loader can load programs into the space occupied by XRAY and programs may store data into the locations occupied by the loader. This means that XRAY and the loader must be reloaded at the end of each job.</p> <p>This is not convenient in non-mass systems. Therefore, the user may specify that the XRAY and the loader are to remain resident throughout the execution of the task. When the task exits, through cell X'40', it will return to XRAY, at which time another operation may be initiated. In this situation it is the responsibility of the user to avoid damage to the loader or to XRAY during the execution of his program.</p> <p>In this mode of operation the space occupied by XRAY is not available to the loader for program loading, and a program being executed may not use the space occupied by the loader for data storage. Preserving XRAY even though the loader is lost may be a convenience, since retaining XRAY provides keyboard control of the system at all times.</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

JOB SPECIFICATION (Cont.)

Form and Argument String and Operating System	Use
<p>SET BACKGROUND CORE SIZE</p> <p>:BM,0HHHH</p> <p>:BM,XXXXX</p> <p>Argument String: Hexadecimal or Decimal</p> <p>System: MPS Background</p>	<p>The BM directive sets the upper limit of the available background core size. The argument string may be either hexadecimal (preceded by the number "0") or decimal. The upper limit must be larger than 80_{16} and less than the value (total core size minus Resident Monitor size). If a value of less than 80_{16} is entered, the upper limit is not reset and the message</p> <p style="text-align: center;">BCOR</p> <p>will appear on the console teletype. If the number entered is greater than (total core size minus Resident Monitor size) the background core size is set to (total core size minus Resident Monitor size minus 1).</p>
<p>QUEUE PERIODIC TASKS</p> <p>:QP, PROGRAM, PRIORITY, START, INTERVAL, REPEAT COUNT</p> <p>Argument String: Alphanumeric</p> <p>System: MPS Background and MPS Foreground</p>	<p>The task PROGRAM is placed in the periodic task table at the indicated PRIORITY. The task is queued at the START time to operate at the given INTERVAL for REPEAT COUNT number of times.</p> <p>The START and INTERVAL times are input in the form:</p> <p style="text-align: center;">HH\$MM*SS.SS</p> <p>where: HH is number of hours MM is number of minutes SS.SS is number of seconds.</p> <p>All times not indicated are assumed to be zero.</p> <p>For example:</p> <p style="text-align: center;">23\$20*</p> <p>indicates the task is to begin at 11:20 p.m. or that the interval is 23 hours, 20 minutes.</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

JOB SPECIFICATION (Cont.)

Form and Argument String and Operating System	Use
	<p>The delimiters \$ and * must follow hours and minutes respectively or the time will be interpreted as seconds.</p> <p>For example:</p> <p style="text-align: center;">23\$20</p> <p>indicates the task is to begin at 11:00 and 20 seconds or that the interval is 23 hours and 20 seconds.</p> <p>The start time is on a 24 hour basis. The time 1:00 a.m. in the morning is represented by 1\$00*00.00 while 1:00 p.m. in the afternoon is represented by 13\$00*00.00.</p> <p>For example:</p> <p>It is Thursday, 1:00 p.m. in the afternoon and a task is periodically queued to start execution at 12\$20*00.00. The task will begin to execute at 12:20 a.m. on Friday.</p> <p>The REPEAT COUNT is any number less than or equal to 32,767. A negative REPEAT COUNT means to cycle the task indefinitely or until the task is deleted from the periodic table with a DQ directive.</p> <p>Care must be taken to insure that the task to be queued periodically does not have a start time differential (ΔT) (start time minus current time) exceeding 65,535.</p> <p>For Example:</p> <p>At 1:00 p.m. in the afternoon the following is entered:</p> <p style="text-align: center;">:QP PROGRAM,2,23\$20,0.02,3</p> <p>The intention was at 11:20 p.m., the task PROGRAM is to be queued and executed at priority level 2 every 0.02 seconds for 3 times including the first time. The above is <i>not</i> acceptable since the starting time is $3,720,000 \times 10^{-2}$ seconds away, which is</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

JOB SPECIFICATION (Cont.)

Form and Argument String and Operating System	Use								
	<p>larger than 65,535. The smallest INTERVAL that could be used is .01 second. The following table lists the start time differential (ΔT) versus the smallest INTERVAL that can be used.</p> <table data-bbox="617 735 1218 913"> <thead> <tr> <th>ΔT (Start Time – Current Time)</th> <th>Smallest Interval</th> </tr> </thead> <tbody> <tr> <td>23 Hr., 59 Min.,</td> <td>1 Min.</td> </tr> <tr> <td>18 Hr., 12 Min., 15 Sec.</td> <td>1 Sec.</td> </tr> <tr> <td>10 Min., 55.35 Sec.</td> <td>0.01 Sec.</td> </tr> </tbody> </table> <p>The task in the preceding example may be accomplished by queuing a routine which will dynamically queue the task PROGNAME:</p> <p>At 1:00 p.m. in the afternoon the following is entered:</p> <pre>:QP,SET,2,23\$20,23\$20,1</pre> <p>SET is the following routine:</p> <pre> SET SMB X'6A' JSX X'6A' D PERTABLE SMB X'40' JSX X'40' PERTABLE D 12 D 'PR' D 'OG' D 'NA' D 'ME' D X'0801' D 2 D 0 D 0 D 0 D 2 D 3 </pre> <p>8 character name</p> <p>Time Base (0,1,2,3) * ΔT (Should be 0) Interval Repetitions</p> <p>* Time-Base 0 = hundredths of seconds 1 = seconds 2 = minutes 3 = hours</p>	ΔT (Start Time – Current Time)	Smallest Interval	23 Hr., 59 Min.,	1 Min.	18 Hr., 12 Min., 15 Sec.	1 Sec.	10 Min., 55.35 Sec.	0.01 Sec.
ΔT (Start Time – Current Time)	Smallest Interval								
23 Hr., 59 Min.,	1 Min.								
18 Hr., 12 Min., 15 Sec.	1 Sec.								
10 Min., 55.35 Sec.	0.01 Sec.								

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

JOB SPECIFICATION (Cont.)

Form and Argument String and Operating System	Use
<p>SET TIME-OF-DAY :ST,HH\$MM*SS.SS</p> <p>Argument String: Alphanumeric</p> <p>Systems: MPS Background MPS Foreground</p>	<p>The time-of-day clock may be set by the ST directive. The delimiter \$ signifies hours and the delimiter * signifies minutes. No delimiter will be interpreted as seconds. The maximum value that can be set is:</p> <p style="text-align: center;">:ST, 23\$59*59.99</p> <p>The minimum value is</p> <p style="text-align: center;">:ST, 0</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

I/O DEVICE MANAGEMENT

System I/O in the operating systems is largely device independent. I/O is performed on logical units which are assigned to physical units at executive time. XRAY directives facilitate this operation. The assignments are stored in the Resident Monitor in a table called the Physical Equipment Assignment Table (PEAT). In addition, an image of this table is stored on the disc (RTOS) and used to reset the core PEAT at each NEXT job. A directive permits this image to be modified.

Certain of the logical units are used by system processors, and to systematize their use, particular functions have been assigned to the first 7 logical units. The following list explains these functions and lists the mnemonics by which these units are often described in the documentation.

These units may be assigned by operator, or by directives read in the batch mode of operation to any physical device present in the system.

LOGICAL UNIT NUMBERS

The following list gives the numbers of the 13 logical units, their names and the function of these units in system programs.

LUN	Name	Description
0	SYSF	SY stem F ile. This is the unit from which the system library is read and written.
1	SYSI	SY stem I nput. SYSI is the unit for all directives input to system programs; such as the XRAY EXEC or SYM II Assembler.

LUN	Name	Description
2	PRIN	PR imary I Nput. PRIN is the unit for input to system programs where the input is neither a directive nor binary; i.e., SYM II accepts the input source text from PRIN.
3	LIST	LI STing. System programs use LIST for tabular output other than error messages. SYM II outputs the assembly listing on the LIST device.
4	BIN	B inary I Nput. BIN is the system binary input unit. Absolute or relocatable binary programs are input from this unit.
5	BOUT	B inary O Utput. BOUT is the system binary output unit. SYM II outputs absolute or relocatable binary text to the BOUT unit.
6	SCR	SCR atch. System programs use SCR as temporary storage. SYM II uses SCR to store intermediate text.
7-11	LOGA, LOGB, LOGC, LOGD, LOGE	The next 5 units, LOGA-E, are intended for users. Devices are divided into two classes; mass and non-mass. A mass device is one that can perform both read and write operations; i.e., disc, magnetic tapes, etc. Non-mass devices can perform only a read or write; i.e., card reader, card punch, etc. LOGA is the system's

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

I/O DEVICE MANAGEMENT (Cont.)

LUN Name	Description	UNIT	Assignment
12 DUMMY	This logical unit performs no I/O but allows programs to be checked out without a peripheral device being available.	13	Console Teletype
		14	Magnetic Tape Unit 0
		15	Magnetic Tape Unit 1
		16	Magnetic Tape Unit 2
		17	Magnetic Tape Unit 2
		18	Disc Unit 0
		19	High Speed Paper Tape Punch
		20	High Speed Paper Tape Reader
		21	Card Reader
		22	Card Punch
		23	Line Printer
		24	Multiplex Teletype
<p>PHYSICAL UNIT NUMBERS</p> <p>The remaining unit numbers correspond to actual physical units which may be present in the system. These are not usually re-assigned, and may be used as the unit number in the FIOT whenever the user wishes to operate a particular unit without the possibility of re-assignment.</p>		25	Disc Unit 1
		26	Disc Unit 2
		27	Disc Unit 3
		28	Plotter
		29	DIDS Channel (CRT Display)

Form and Argument String and Operating System	Use
<p>DEVICE REASSIGNMENT</p> <p>:IO,L1,P1,L2,P2,...</p> <p>:PI,L1,P1,L2,P2,...</p> <p>Argument String: Variable length Numeric</p> <p>Systems: (:IO), STANDARD, MTOS, RTOS, MPS Background, MPS Foreground</p> <p>(:PI) MTOS, RTOS, MPS Background.</p>	<p>Logical unit number Pn is assigned to logical unit number Ln. Logical units greater than 11 cannot be given as an L. Thus, physical units cannot be reassigned by this directive.</p> <p>:IO,3,13,4,7</p> <p>Logical unit 3 is assigned to physical unit 13 (the teletype). Logical unit 4 is assigned to the same physical unit to which logical unit 7 is assigned.</p> <p>If an Ln greater than 11 is given, the message</p> <p>"ICD?"</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

I/O DEVICE MANAGEMENT (Cont.)

Form and Argument String and Operating System	Use
<p>REWIND :RW, N1, N2,</p> <p>Argument String: Variable length Numeric</p> <p>Systems: MTOS, RTOS, MPS Background, MPS Foreground</p> <p>WRITE END-OF-FILE :WE, N1, N2, . . .</p> <p>Argument String: Variable length Numeric</p> <p>Systems: STANDARD, MTOS, RTOS, MPS Background, MPS Foreground</p>	<p>is listed and the offending assignment and those following it in the argument string are ignored.</p> <p>Assignments made by the :IO directive are temporary changes to the PEAT which is reset to the permanent configuration at each NEXT job.</p> <p>Permanent changes are made using the directive</p> <p style="text-align: center;">:P1,L1,P1,L2,P2...</p> <p>where the form and meaning of this argument string are identical to that for the :IO directive. This permanent change is stored on the disc, and will be changed only if the system is regenerated on the disc, or changed by another :PI directive. Whenever a :PI directive is given, the PEAT is immediately reset to its permanent configuration, then the new assignments are made and the new configuration is stored on the disc. Any assignments made to a device not physically present will be automatically switched to the dummy device (unit 12).</p> <p>Units numbers N1,N2,... are rewound if the units are, in fact, magnetic tapes. If the units specified are not magnetic tapes, no operation occurs. The physical unit numbers for magnetic tapes 0-3 are 14-17. Thus</p> <p style="text-align: center;">:RW,OF,16</p> <p>will rewind units 1 and 2.</p> <p>A file mark is written on units N1,N2,... A file mark is defined for every output device except the disc, and this directive is provided to permit a file mark to be added to a loadable binary program, in order to specify end-of-load.</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

I/O DEVICE MANAGEMENT (Cont.)

Form and Argument String and Operating System	Use
<p>DISC FILE MANAGEMENT :DF,N1,BA1,SZ1,N2,BA2,SZ2 :PF,N1,BA1,SZ1,N2,BA2,SZ2</p> <p>Argument String: Variable length Numeric</p> <p>Systems: RTOS, MPS Back- ground, MPS Foreground</p>	<p>The disc user area is divided into four relatively addressed logical files, numbered 0-3. File 0 is used by SYM II to accumulate binary output whenever BOUT is assigned to the disc. File 1 is used as FORTRAN and SYM II intermediate whenever the Scratch Unit (SCR) is assigned to the disc. The DF directive permits the operator to allocate storage dynamically on the disc. N1 is the logical file number. BA1 is the starting sector number of the file relative to the end of the disc library system. SZ1 is the number of sectors assigned in the file. If BA and SZ are given, such that the file will not fit on the disc, the message</p> <p style="text-align: center;">FIL?</p> <p>is output. The special value END may be given for SZ. This will cause the file to extend to the end of the disc on which it begins. BA may be given as E0, E1, E2, or E3, meaning file begins at the end of the specified file.</p> <p>Thus, the directive</p> <p style="text-align: center;">:DF,0,0,500,1,E0,500,2,E1,2000,3,E2,END</p> <p>will divide the disc into four files, non-overlapping. If the system contains more than one disc, the directive</p> <p style="text-align: center;">:DF,0,0,END,1,E0,END,2,E1,END,3,E2,END</p> <p>will place file 0 on the remainder of disc 0 and files 1, 2, and 3 on discs 1, 2, and 3, respectively. On the other hand, to create file 0 starting at sector 1000 relative to the end of the system and makes it 1500 sectors long, the directive</p> <p style="text-align: center;">:DF,0,1000,1500</p> <p>is given. During System Generation logical files 0, 2, and 3 are initialized to a starting sector of 0 and a size such that each file extends to the end of the disc. Logical file 1 has a starting sector of 15 and also extends to the end of the file. For additional information regarding disc files and their use, refer to <i>"Disc and Magnetic Tape Driver"</i>, Section 8-6.6.</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

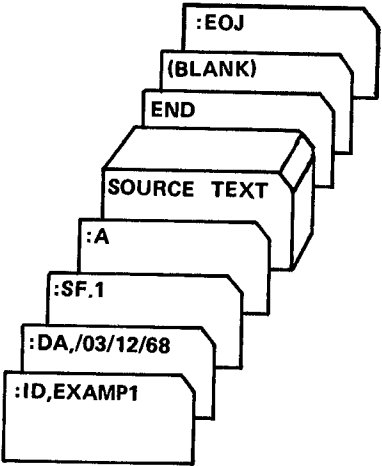
I/O DEVICE MANAGEMENT (Cont.)

Form and Argument String and Operating System	Use
<p>DEDICATE I/O :DO,P1,P2,</p> <p>Argument String: Numeric</p> <p>Systems: MPS Background, MPS Foreground</p> <p>RELEASE I/O :RO,P1,P2, . . .</p> <p>Argument String: Numeric</p> <p>Systems: MPS Background MPS Foreground</p>	<p>File allocations made by the :DR directive are for the duration of a job only; that is, they will be reset to the permanent configurations at each :NJ or :EOJ directive. Permanent changes in the disc file allocation can be made by using the directive</p> <p style="text-align: center;">:PF,N1,BA1,SZ1,N2,BA2,SZ2</p> <p>where the argument string has the same structure and meaning as that for the :DF directive. Whenever the :PF directive is given, the file allocation is immediately reset to the permanent configuration.</p> <p>Each installation must decide how the disc files are to be allocated and utilized by the system users, observing the restrictions on files 0 and 1 by major Raytheon-supplied system processors; such as SYM II and FORTRAN IV.</p> <p>The DO directive dedicates or reserves peripheral equipment for either the MPS background or MPS foreground system. Logical devices may not be dedicated. Peripheral devices remain dedicated for the duration of a job or until the RO directive is given. If the device to be dedicated is already dedicated in a different system (i.e., foreground tries to dedicate device but background already has dedicated device), the message:</p> <p style="text-align: center;">USED</p> <p>is output on the teletype.</p> <p>The RO directive releases or undedicates peripheral devices for either the MPS background or foreground respectively.</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

ASSEMBLY AND COMPILATION

Form and Argument String and Operating System	Use																						
<p>ASSEMBLY :A</p> <p>No Argument String</p> <p>Systems: MTOS, RTOS, MPS Background</p> 	<p>Assembly is specified by the RTOS directive</p> <p>:A</p> <p>When this directive is given, a single program is assembled with SYM II, after which control returns to RTXRAY.</p> <p>Assembly options available are shown below. If the user wishes to use the manual operating mode of the assembler in which the assembler types</p> <p>PAS?</p> <p>at the end of every assembly pass and reads the system input device for a 1, 2, or X card (refer to SYM II Operating Instructions in the SYM II Manual, SP-274), he should queue the assembler and not use the A directive.</p> <p>For example:</p> <p>:QU,SYM2 :EX</p> <p>Following are two examples of deck stacking for assembly. Of course, in the operator mode the directives would be input from the teletype.</p> <p>1. Assembly of a single program with no binary output</p> <table border="0"> <tr> <td>:BA</td> <td>Optional Batch Mode</td> </tr> <tr> <td>:EX</td> <td>input by operator</td> </tr> <tr> <td>:ID,EXAMP1,DA,/03/12/68</td> <td>sets the ID and date</td> </tr> <tr> <td>:SF,1</td> <td>request no binary</td> </tr> <tr> <td>:A</td> <td>request assembly</td> </tr> <tr> <td>source text</td> <td></td> </tr> <tr> <td>.....</td> <td></td> </tr> <tr> <td>END</td> <td></td> </tr> <tr> <td>blank card</td> <td>SYM II double buffers</td> </tr> <tr> <td>:EOJ</td> <td></td> </tr> <tr> <td>(:MA)</td> <td>return to operator mode</td> </tr> </table>	:BA	Optional Batch Mode	:EX	input by operator	:ID,EXAMP1,DA,/03/12/68	sets the ID and date	:SF,1	request no binary	:A	request assembly	source text			END		blank card	SYM II double buffers	:EOJ		(:MA)	return to operator mode
:BA	Optional Batch Mode																						
:EX	input by operator																						
:ID,EXAMP1,DA,/03/12/68	sets the ID and date																						
:SF,1	request no binary																						
:A	request assembly																						
source text																							
.....																							
END																							
blank card	SYM II double buffers																						
:EOJ																							
(:MA)	return to operator mode																						

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

ASSEMBLY AND COMPILATION (Cont.)

Form and Argument String and Operating System	Use
	<p>2. Assembly of three programs in Batch Mode; the first from a magnetic tape source input with no binary output, the second from cards with no listing, the third from cards with binary and listing.</p> <p>:BA input by operator</p> <p>:EX input by operator</p> <p>:NJ introduce the job</p> <p>:IO,2,14 assign mag tape 0 to PRIN</p> <p>:SF,1 no binary</p> <p>:ID,EXAMP2 source text</p> <p>:A mag tape source</p> <p>:EOJ resets PRIN unit</p> <p>:SF,0 request no listing</p> <p>:ID,EXAMP2A source text</p> <p>:A</p> <p>END blank</p> <p>:EOJ reset system flags</p> <p>:ID,EXAMP2B source text</p> <p>:A</p> <p>END Blank</p> <p>:EOJ</p> <p>:MA return to operator mode</p> <p>In the preceding example the specification of the first assembly might have been given on a single line</p> <p>:IO,2,14,:SF,1,:ID,EXAMP2,A</p>

Table M-1. SYM II Options

System Flag	Sense Switch	Function
0	0	No source listing if True.
1	1	No binary output if True.
	2	List errors even if no listing.
11		Automatic operation. SYM II does Pass 1, then Pass 2, then exits. Set by :A directive.

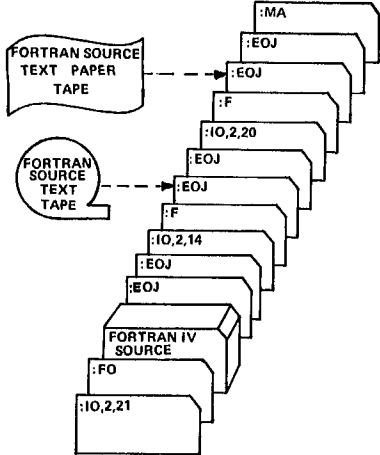
Table M-2. FORTRAN Options

System Flag	Sense Switch	Function
0		No compiler list.
	0	Suppresses compiler and assembler listings unconditionally.
1	1	No binary output from assembly phase.
2		Allow SYM II to list during assembly phase. Note that sense switch 0 overrides this request.

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

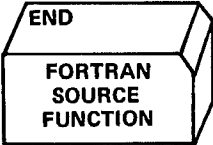
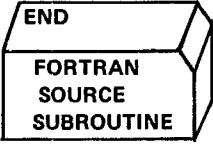
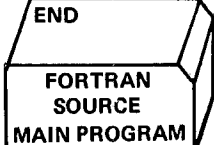
ASSEMBLY AND COMPILATION (Cont.)

Form and Argument String and Operating System	Use
<p>COMPILE WITH FORTRAN IV</p> <p>:F :FO</p> <p>No Argument String</p> <p>Systems: MTOS, RTOS, MPS Background</p> 	<p>Compilation with FORTRAN IV is specified by the :F directive. Options available are specified above. When the :F directive is used, XRAY assumes responsibility for sequencing the job through the compilation and assembly of the job. Multiple compilations may be stacked and performed sequentially. If any compilation error occurs, the assembly phase is automatically bypassed. If the user wishes to perform only the compilation phase, he may do so with the :FO directive. Once the :FO directive has been given, all compilation performed will be 'Compile Only', until after the next :EOJ directive. For example:</p> <ol style="list-style-type: none"> 1. Compile 3 programs, one from card source, one from magnetic tape source and one from paper tape source. The card source should have no binary output. <pre> :BA input by operator :EX input by operator :IO,2,21 :FO FORTRAN Source :EOJ :EOJ Required to get system out of compile only mode Sets PRIN to MT 0 :IO,2,14 :F Magnetic tape source END Read from magnetic tape source :EOJ :EOJ :IO,2,20 Sets PRIN to paper tape :F Paper tape source END Read from paper tape source :EOJ :EOJ :MA End of job read from SYSI Return to operate mode </pre>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

ASSEMBLY AND COMPILATION (Cont.)

Form and Argument String and Operating System	Use
<div style="text-align: center;">    </div> <p>ASSEMBLE AND GO :AG</p> <p>COMPILE AND GO :FG</p> <p>OUTPUT BINARY :OB</p> <p>EXECUTE :GO</p> <p>Argument String: None</p> <p>Systems: MTOS, RTOS, MPS Background</p>	<p>2. Compile 1 program including subroutines and functions from card source. Binary output is cards. Listing is in System List Device.</p> <p>:NJ Input by operator :IO,2,21,5,22 Sets PRIN to CR and BOUT to CP</p> <p>:ID,EXAMP3 Input by operator :F Input by operator FORTRAN Source-Main Program END :EOJ Input by operator :F Input by operator FORTRAN Source-Subroutine END :EOJ Input by operator :F Input by operator FORTRAN Source-Function END :EOJ Input by operator</p> <p>Assemble and Go is specified with the directive :AG. Compile and Go with the directive :FG. Any number of compilations and assemblies may be intermixed and the output accumulated on the Binary Output (BOUT) unit for loading. The following rules must be observed.</p> <ol style="list-style-type: none"> 1. The BOUT unit must be assigned to a mass unit – disc or magnetic tape. 2. Sense Switch 1 must be off. If it is on when the :FG or :AG directive is received <p style="text-align: center;">SW1</p> <p>will be listed on the teletype until the switch is turned off. See Tables M-1 and M-2.</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

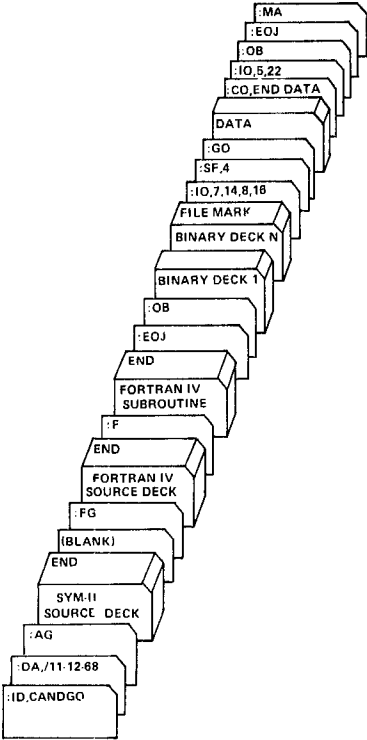
ASSEMBLY AND COMPILATION (Cont.)

Form and Argument String and Operating System	Use
	<p>3. All programs for Assemble and Go must be relocatable. No absolute binary program can be loaded from the disc.</p> <p>When the desired number of binary programs has been stacked on the BOUT device, the directive :GO will cause the stack to be loaded and executed normally. If the conditions for GO operation are not met, the message</p> <p style="text-align: center;">NO GO</p> <p>is output, and the directive is ignored. This can occur if the BOUT unit is not correctly assigned or if no programs have been compiled or assembled for go operation when the :GO directive is given.</p> <p>Binary object decks may also be intermixed into the input stream, and accumulated along with the compiler and assembler output on the Binary Output Device (BOUT).</p> <p>A stack of relocatable binary programs in loadable format is introduced to XRAY by the directive</p> <p style="text-align: center;">:OB</p> <p>Program after program is copied on the disc until a file mark is encountered. Incoming records are checksum verified and checked for text validity. Programs are copied from the Binary Input (BIN) device. After a file mark is read, control returns to XRAY and processing the job stream continues.</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

ASSEMBLY AND COMPILATION (Cont.)

Form and Argument String and Operating System	Use
	<p>The :OB directive may also be used to output the accumulated binary from the GO device, after execution. This is accomplished by re-assigning the Binary Output device to a hard copy device and giving the :OB directive. If a system detected error has occurred during execution, the object program dump will be omitted automatically, in the batch mode.</p> <p>The following example will clarify the use of these directive since it includes a mixed compilation, assembly, and object program stream, followed by execution and output of hard copy binary programs.</p> <p>Example:</p> <pre> :NJ input by operator :BA input by operator :ID,CANDGO,DA,/11-12-68 :AG assemble and go Source END blank :FG now compile and Go Source END :F do another compilation (Go is automatically understood) Source Subroutine END :EOJ any card (except BLANK) will do to stop compiling. :OB introduce a group of object programs Binary deck 1 </pre>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

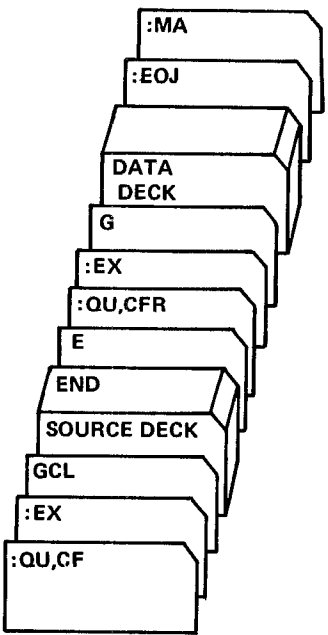
ASSEMBLY AND COMPILATION (Cont.)

Form and Argument String and Operating System	Use
	<p>Binary deck n File mark :IO,7,14,8,16 :SF,4 :GO Data :CO,END DATA :IO,5,22 :OB :EOJ :MA</p> <p>end of binary set up I/O for execution (arbitrary) request a load map execute the program . comment card marks end of data deck reassign BOUT to the card punch dumps the binary object programs to the card punch end of job Returns control to XRAY</p> <p>The comment card :CO,END DATA terminates a FORTRAN data deck. If it is read by the FORTRAN I/O processor, the I/O processor will exit, whereas XRAY will ignore the comment card.</p>
<p>COMPILE WITH CONVERSATIONAL FORTRAN :QU,CF</p> <p>No argument string</p> <p>Systems: MTOS, RTOS, MPS Background</p>	<p>Compilation with Conversational FORTRAN is specified by queuing the CF processor. The following example will compile a source program from magnetic tape 0 and produce a binary paper tape object.</p> <p>:NJ :QU,CF :IO,2,15,5,19 :EX G Program source B E</p> <p>Compiles statements in GO mode Input from Magnetic Tape Produces absolute binary Exits from compiler to XRAY</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

ASSEMBLY AND COMPILATION (Cont.)

Form and Argument String and Operating System	Use
<p>EXECUTE CONVERSATIONAL FORTRAN :QU,CFR</p> <p>No Argument String Systems: MTOS, RTOS, MPS Background</p> <p>COMPILE AND GO WITH CONVERSATIONAL FORTRAN</p> 	<p>The following directives will load the Conversational FORTRAN Run-time library, and execute the program that has just been compiled in the previous example.</p> <p>:QU,CFR :EX G</p> <p>To compile and execute a Conversational FORTRAN program with a card reader, the following directives must be input:</p> <p>:NJ } :BA } Input by operator :EX }</p> <p>:QU,CF :EX GCL Compile and go mode, allow continuation cards and list the source program</p> <p>Source Deck END E :QU,CFR Load the run-time library :EX G Execute the program</p> <p>Data deck :EOJ :MA Return control to XRAY</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

RESIDENT PROGRAM DIRECTIVES

Form and Argument String and Operating System	Use
<p>CREATE A RESIDENT TASK :RT,TASKNAME, LIBENAME</p> <p>Argument String: Variable length Alphabetic</p> <p>Systems: RTOS, MPS Foreground</p>	<p>The argument TASKNAME pre-empts any entry names found in the program which is loaded and made resident. This name will appear in the resident task list, and is the name by which the task may be placed in the queue and executed. The argument LIBENAME is optional. If it is absent, the resident loaded will load from the BIN device, then satisfy any undefined references from the library. If LIBENAME is present, the resident loader will proceed directly to the library phase, and load the program called LIBENAME from the library.</p>
<p>CREATE A RESIDENT SUBROUTINE :RS,LIBENAME</p> <p>Argument String: Fixed length Alphabetic</p> <p>Systems: RTOS, MPS Foreground</p>	<p>A resident task cannot be linked to a loaded program, but can only be connected to an interrupt or placed in the queue to be executed. A resident subroutine on the other hand is linked to any program loaded by the loader, just as though it had been loaded from the library. The argument LIBENAME is a library name of a subroutine to be loaded and made resident. Any other entry name in this program is also placed in the resident subroutine names table. However, entry names in subroutines called by this subroutine are not placed in the resident names table, although they are loaded and linked to the first subroutine. Any resident subroutine called by the first subroutine will be linked to it. Thus, residence must be declared starting at the lowest level at which the linkage names are desired to appear in the resident subroutines names table. At lower levels multiple copies are automatically loaded. This feature allows control of subroutine re-entry, allowing the user to limit re-entry to only those which are actually re-enterable (recursive).</p>
<p>CREATE A RESIDENT DATA BLOCK :DB,NAME,SIZE</p> <p>Fixed Length Mixed Argument String</p> <p>Systems: RTOS, MPS Background MPS Foreground</p>	<p>NAME is the eight or fewer character name of the block and SIZE is the number of memory locations required.</p> <p>A resident data block is created, and the name is placed in the block link table. Such a block may be referenced by any program module as an external.</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

RESIDENT PROGRAM DIRECTIVES (Cont.)

Form and Argument String and Operating System	Use
<p>CREATE A RESIDENT COMMON BLOCK :CB,NAME,SIZE :DC,NAME,SIZE</p> <p>Fixed Length Mixed Argument String Systems: RTOS, MPS Background, MPS Foreground</p>	<p>A resident labeled common block is created and the NAME placed in the link table. This type of block must be referenced as a labeled common block, and is subject to the rules governing FORTRAN block common.</p> <p>To create a block which can be referenced either as a common block or as an ordinary external, the directive</p> <p style="text-align: center;">:DC,NAME,SIZE</p> <p>is used.</p>
<p>PURGE DATA OR COMMON BLOCKS :PD,NAME :PC,NAME</p> <p>Argument String: Fixed Length Alphabetic Systems: RTOS, MPS Background, MPS Foreground</p>	<p>PD purges data block NAME; PC purges common block NAME. If a block has been created using the DC directive, both purge directives should be given to completely delete the block.</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

INTERRUPT SYSTEM CONTROL

Form and Argument String and Operating System	Use
<p>ARMING & DISARMING INTERRUPTS :AI,N :DI,N</p> <p>Argument String: Fixed length Numeric</p> <p>Systems: RTOS, MPS Backgrnd.</p>	<p>Interrupt N is armed or disarmed. Interrupt connection, arming and disarming directives have no affect upon I/O interrupt levels connected to devices controlled by the I/O Monitor. An attempt to arm or disarm an I/O level controlled by the I/O Monitor will cause the message</p> <p>ICD?</p> <p>to be output, signifying an illegal control directive.</p>
<p>INTERRUPT CONNECTION :IC,N,TASKNAME</p> <p>Argument String: Fixed Length Numeric and Variable Length Al- phabetic</p> <p>Systems: RTOS, MPS Background</p>	<p>The task TASKNAME is connected to interrupt level N. This connection supercedes any previous connection which may have existed. If TASKNAME is blank, the interrupt is disconnected from any task.</p> <p>TASKNAME must be a resident task. If it is not, the message</p> <p>NRES</p> <p>is output and no connection is made.</p>
<p>CONNECT QUEUE :CQ,TASKNAME,PRIORITY, LEVEL,ARG1,ARG2,ARG3</p> <p>Argument String: Fixed Length Numeric and Variable Length Al- phabetic</p> <p>System: MPS Foreground</p>	<p>The task TASKNAME is inserted into the queue at PRIORITY. At execution interrupt, LEVEL is connected to TASKNAME.</p> <p>If TASKNAME is blank, the interrupt is disconnected from any task.</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

LIST

Form and Argument String and Operating System	Use
<p>LIST DISC VECTOR :LD</p> <p>Argument String: None</p> <p>Systems: MPS Background MPS Foreground</p>	<p>The LD directive causes all disc vectors to be listed on the system list device.</p>
<p>LIST PEAT TABLE AND MARK DEDICATED DEVICES :LP</p> <p>Argument String: None</p> <p>Systems: MPS Background MPS Foreground</p>	<p>The LP directive causes both the MPS Background and MPS Foreground PEAT tables to be listed on the system list device. Dedicated devices are marked.</p>
<p>LIST QUEUE :LQ</p> <p>Argument String: None</p> <p>System: MPS Background</p>	<p>The LQ directive causes a snapshot dump of the queue on the system list device. Included in the list are the tasknames, priorities, background or foreground, and connected or periodic information.</p>
<p>LIST RESIDENT TASKS :LR</p> <p>Argument String: None</p> <p>System: MPS Foreground</p>	<p>The LR directive outputs a snapshot of all resident tasks on the system list device.</p>
<p>OUTPUT TIME-OF-DAY :OT</p> <p>Argument String: None</p> <p>Systems: MPS Background MPS Foreground</p>	<p>The OT directive outputs the date and time-of-day on the system list device and the teletype. The format is:</p> <p style="text-align: center;">04/18/69/12:34:02</p>

APPENDIX M-2B

XRAY OPERATING SYSTEM DIRECTIVES

LIST (Cont.)

Form and Argument String and Operating System	Use
<p>LIST PERIODIC TASK TABLE :PL</p> <p>Argument String: None</p> <p>System: MPS Foreground</p>	<p>The PL directive outputs a snapshot of the periodic task table on the system list device. See QP directive for format.</p>
<p>PSEUDO RESIDENT LIST :PR</p> <p>Argument String: None</p> <p>System: MPS Background</p>	<p>The PR directive outputs a snapshot dump of the names and locations of the Resident Data Blocks and Common on the system list device.</p>
<p>MAP RESIDENT MEMORY :MP</p> <p>No Argument String</p> <p>Systems: RTOS, MPS Back- ground, MPS Foreground</p>	<p>A load map of entry names currently in resident memory is output on the LIST device.</p>

APPENDIX N

ASSEMBLER/COMPILER ERROR REVIEW

N-1 SYM I ERROR MESSAGES

The following is a list of all possible error indicators from SYM-1 and their meaning. These indicators are displayed to the extreme left of a source image and are always one character in length.

If the input unit is the teletype keyboard, the erroneous source statement is ignored. If the input unit is not the teletype keyboard the erroneous statement and error indicator are printed on the teletype and assembly is terminated.

Some source statements are not completely deleted when an error is detected.

For example:

```
A DATA 1, 2, X'A00FF', 3
```

Let us assume that the extra 'F' in the hex string was a typographical error. The assembler considers the whole statement in error but has put the label 'A' in the symbol table. This also happens using the JSX command. If an attempt is now made to re-type the label 'A', it will be double-defined. Any reference to 'A' will not 'PREP' correctly.

Indicator	Reason for Occurrence
B	Usage error. Improper use of byte addressing.
C	Operator missing in a conditional statement.
D	Doubly defined symbol.
E	Equate statement was given without a label.
L	Illegal character in label field. Library names (LIBR) not first directive in relocatable program.
M	Magnitude error in operand field. Operand is too large or operand is illegally designated as negative (-).
P	Parameter error. Illegal operand. Error in expression in operand field. Missing or illegal operator or terminator in operand. More than one operand required and one operand required and one is missing.
V	Block overflow. Length of program more than one byte page (1024 words). Available core is exhausted due to unusually large symbol table. Operand address or value of operand used as an address not within one word page (2048 words).

N-2 SYM II ERROR MESSAGES

The following single character error codes are those which appear in columns 1 or 3 or both of a SYM II assembly listing. These characters are all the error codes which SYM II displays for assembly errors.

Indicator	Reason for Occurrence
B	A byte operand was used with a word instruction (class i).
C	Relational operator missing in TRUE or FALSE pseudo op.
D	This symbol has previously been defined causing a double definition. First definition prevails.
E	Equate error. No label with equate or double defined equate label.
L	PROC pseudo op used in PROC definition without a label.
M	a. More than 16 bits in a hexadecimal constant. b. Decimal integer constant larger than 32,767.

APPENDIX N

ASSEMBLER/COMPILER ERROR REVIEW

N-2 SYM II ERROR MESSAGES (Cont)

- c. An expression in the operand field whose calculated value is larger than $\pm 32,767$.
 - d. Operand larger than 4 bits used with a 4 bit operand generic (class 4,5).
 - e. Operand larger than one byte used with a byte operand generic (class 4,5).
 - f. Operand larger than 4 bits used with I/O generic (class 8).
 - a. Unrecognized parameter in operand field.
 - b. Terminating apostrophe missing for hexadecimal constant.
 - c. Missing separator in concatenation option of DATA pseudo op. (:)
 - d. Illegal use of parenthesis.
 - e. Leading or termination quote missing in the alpha character string of a TEXT pseudo op.
 - f.. Multiple parameters in operand field not separated by a comma.
 - g. A byte label appears as an externally defined label.
 - h. Negative value given as operand in ORIG pseudo op.
 - i. Unrecognizable literal operand in Pass 2.
 - j. Operand of an RES pseudo op not an absolute constant. Only one word is reserved.
 - k. Operand of RES pseudo op a negative number. Only one word is reserved.
 - l. The operand of the END statement is not a relocatable name.
 - m. The value given to fill the specified field in a concatenation in the DATA pseudo op is not an absolute number.
 - n. The size specified for the field to be concatenated in a DATA pseudo op is larger than 16 bits.
 - o. An argument number, N, designated by a "(N)" in a PROC definition was not assigned an absolute number in the PROC reference line.
 - p. Argument number given in this PROC is too large.
 - q. Argument number given in this PROC is negative.
 - r. The sum of relocatable addresses is different at assembly time and load time; hence expression is meaningless.
 - s. An absolute address used with a word op code (class 1) whose value is larger than 11 bits.
- R Data block initialization outside named datablock (i.e., bias larger than block size).
- V Address overflow. Address used with a word or byte op code (class 1 or 2) is outside the presently used word or byte page respectively.
- X Available core exhausted by symbol table. This is the only error causing termination of the assembly.
- Y Symbol error. First character of a symbol or operator code not a letter.

APPENDIX N
ASSEMBLER/COMPILER ERROR REVIEW

N-2 SYM II ERROR MESSAGES (Cont)

The following errors may appear on the teletype:

- IOE1 Input/Output error during reading of source during pass 1 or read error of source from scratch unit during pass 2. Processing continued.
- IOE2 Input/Output error during write of source onto scratch unit. Processing is terminated and Assembler is restarted.

N-3 REAL-TIME FORTRAN IV COMPILE TIME ERROR MESSAGES

- AD Adjustable dimension declaration not a dummy name
- AM Argument missing in function declaration
- BL DO loop termination label not forward
- BP Not enough branch points in arithmetic IF
- BX Executable statement within BLOCK DATA sub-program
- CE Initialization of blank COMMON illegal
- CG Closing parenthesis missing in computed GO TO
- CM Comma missing
- DA Declaration statement incomplete
- DE Maximum loop value in DO missing
- DL DO termin label not a number
- DM Dimension declarations missing with array name
- DR Reference to an array with improper number subscripts
- DV Dimension allocation greater than 16K
- E1 Missing parenthesis in EQUIVALENCE statement
- E1 Subscripted variable not previously dimensioned
- E1 Missing comma in EQUIVALENCE statement
- E2 Dummy variable in EQUIVALENCE statement
- E3 Conflicting EQUIVALENCE statement
- E4 EQUIVALENCE conflict with COMMON statement
- EE Executable statement before statement function or COMMON
- FM FORMAT statement error
- GN GO with nothing following
- HO Hollerith error
- IA Adjustable dimensions not allowed in main program
- IB Illegal use of BLOCK statement name

APPENDIX N
ASSEMBLER/COMPILER ERROR REVIEW

N-3 REAL-TIME FORTRAN IV COMPILE TIME ERROR MESSAGES (Cont)

IC	Illegal character in DATA statement
IF	Illegal function name
IN	Name in CALL not a subroutine name
IO	Input-Output error
IS	Illegal SUBROUTINE statement.
IS	Illegal dummy argument
IT	ASSIGN statement target illegal, not a name
LE	Label error
LO	Logical error
LU	Literal constant used in other than CALL or DATA statement
MR	Missing right parenthesis
NC	First character of name not letter
NI	ASSIGN source not an integer
NR	No RETURN statement in sub-program
NU	Number error
OO	Not a 1-1 correspondence in DATA
OP	Illegal Operator
OV	Integer for subscript on dimension size too large
PA	Left parenthesis, but no arguments
PD	Name in external previously defined
PM	Closing parenthesis missing
RN	No SUBROUTINE or FUNCTION with return
RT	Re-typing by multiple type statements not allowed
RW	Input/Output statement error
SF	Statement function argument delimiters are wrong; s or ()
SI	Subscript in array element illegal in DATA
SN	Syntax error - next item and no valid first character
SO	Compiler working area exhausted
TC	DATA type conflict
TE	Termination error, or unbalanced parenthesis
TM	Target missing in ASSIGN statement
TN	Number is too long
TO	TO missing in ASSIGN statement
TY	Illegal name in type statement
UP	Unbalanced parenthesis in IF expression
UT	Unknown data type in initialization

APPENDIX N
ASSEMBLER/COMPILER ERROR REVIEW

N-3 REAL-TIME FORTRAN IV COMPILE TIME ERROR MESSAGES (Cont)

VA	Variable previously dimensioned or assigned in COMMON
XA	Illegal array element
ZC	Zero digits in name or label
ZZ	Function name missing after FUNCTION statement

MISSING LABEL – The label is missing or is contained on a non-executable statement. Only the first 6 characters are printed.

DO LOOP NESTING ERROR – Improper arrangement of DO loops.

The two character error codes appear on the line following the erroneous statement with an up arrow pointing to the character where the compiler abandoned the statement. Very often the arrow is pointing to the offending character. If the compiler scans off the end of a statement, the next statement may be falsely accused of having a label or termination error.

Examples:

```
10      A = * B
ERSN .....↑.....
          SUBROUTINE JOE (1, A)
ERIS.....↑.....
```

APPENDIX N

N-4 REAL-TIME FORTRAN IV RUN-TIME ERROR MESSAGES

GENERAL FORMAT: ERR XXXX

ERROR MESSAGES

Message	Routine	Condition	Result	Flag Overflow
NSQR	SQRT	ARG 0	Zero	No
X**X	EXP	Overflow	Positive Maximum	Yes
LOGN	LOG	ARG 0	Positive Maximum	Yes
LOGZ	LOG	ARG = 0	Negative Maximum	Yes
X**Y	XY	Overflow	Positive or Negative Maximum	Yes*
X**J	X ^j	Overflow	Positive or Negative Maximum	Yes*
I**J	I ^j	Overflow	Positive or Negative Maximum	Yes**
ATN2	ATAN2	Zero	Zero	No
SUBS	R.ARRY	1. Subscript ≤ 0 2. Subscript > 0 Dimension	Selects First Array Element Selects First Array Element	N.A. N.A.
DIMS	R.ARRY	Dimension of ARRAY Exceeds Core	Selects First Array Element	N.A.
DIMS	R.RIO	Dimension of Array Exceeds Core	Aborts	N.A.
FORM	R.IO	1. No opening left parenthesis or left parenthesis is not the first character 2. Illegal character in format 3. No width specified 4. Digits to the right of the decimal point unspecified 5. Parenthesis group nested more than five levels deep.	Aborts Aborts Aborts Aborts Aborts	N.A. N.A. N.A. N.A. N.A.
DATA	R.IO	1. First non blank character in a logical input string is other than T or F. 2. More characters have been requested than are present in the input buffer 3. Data transmission error occurred between the peripheral device and the processor	Aborts Aborts Aborts	N.A. N.A. N.A.

APPENDIX N

ASSEMBLER/COMPILER ERROR REVIEW

N-4 REAL-TIME FORTRAN IV RUN-TIME ERROR MESSAGES (Cont)

4. Length of the output buffer has been exceeded.	Aborts	N.A.
5. Illegal character in the input string	Aborts	N.A.
6. Magnitude of integer data in the input string exceeds the maximum allowed	Aborts	N.A.
7. Magnitude of exponent of real constants in the input string exceeds the maximum or minimum allowed.	Aborts	N.A.
8. The list items requested exceed the number of items in the logical record while processing unformatted data.	Aborts	N.A.

NOTE:

I, F, E, D and G output conversions

If the magnitude of the data requires more positions than provided by the value of the width w , only w digits appear in the external string, and the digits lost are the right most or least significant digits. In addition, the sign will be output as a \$ character if the internal value was positive or an * character if the external value was negative. This is not a terminating error condition. Thus, for the specification F 5.3:

25.125	is converted to	\$25.1
200.3	is converted to	\$200
-1.250	is converted to	*1.25

To insure that such a loss of digits does not occur, the following relation must be satisfied:

$$W \geq d+2+n$$

where n is the number of digits to the left of the decimal point.

* Some overflow may be detected by EXP, in which case the proper routine might add its own message or not.

** If $|^2$ does not give an overflow no message is released in case of an overflow of $|j$.

APPENDIX N
ASSEMBLER/COMPILER ERROR REVIEW

N-5 CONVERSATIONAL FORTRAN COMPILE TIME ERROR MESSAGE

CM	Second COMMON statement in program
CN	Integer too large
CN	No trailing apostrophe on hexadecimal constant
CN	Number error
CN	Real exponent too large
DC	Declaration statements after executable statements
DD	Doubly defined label or identifier
DN	Improper nesting of DO statements
DS	Array dimensioned size is too large
ID	Identifier does not begin with a letter
LA	Missing statement level
LL	Locate hex address too low
MC	Missing comma
ML	Missing left parenthesis
MR	Missing right parenthesis
MX	Mixed mode, integer and real, arithmetic
NA	Not an array name
NI	Not integer expression
OP	Unrecognized operator
RT	RETURN statement not in subroutine
SL	Incorrect statement level reference or definition
SZ	Program too large for memory
TR	Statement not properly terminated

APPENDIX N

ASSEMBLER/COMPILER ERROR REVIEW

N-6 CONVERSATIONAL FORTRAN RUN-TIME ERROR MESSAGES

FD	Unrecognized descriptor in FORMAT statement
FM	Input data magnitude error.
FT	Data type of I/O list item does not match FORMAT statement
FW	Insufficient field width specification for I/O list item.
MS	Called subprogram or function is missing.
SO	Subscript overflow. Array subscript is larger than dimensioned size. Selects first element of array.
SU	Subscript underflow. Array subscript is zero or negative. Selects first element of array.

N-7 BASIC RELOCATABLE LOADER ERROR MESSAGES

<u>Code</u>	<u>Error</u>	<u>Action to be Taken</u>
CK	Checksum error, or non-loader text record.	Re-position the tape in front of the record, if desired to re-attempt reading it, and push the 'RUN' button to continue.
LC	An unrecognized loader code has been encountered.	No recovery
MX	Program being loaded is larger than available memory, or insufficient memory for the entry names table (approximately 100 symbols can be held in this table).	No recovery
MS	Undefined names in the entry names table.	Load missing program using the CL directive.
NX	No execution address has been specified (no main line program has been loaded).	Load main line using the CL directive.

When any of these errors occur the loader will halt to allow the operator to turn off the reader before any messages are output. When the operator pushes the run button, the loader will continue.

APPENDIX N
ASSEMBLER/COMPILER ERROR REVIEW

N-8 STANDARD AND DISC RELOCATABLE LOADER ERROR MESSAGES

<u>Code</u>	<u>Error</u>	<u>Action to be Taken</u>
CK	Checksum error.	Reposition the record and toggle switch 3 to re-read.
LC	An unrecognized loader code has been encountered.	No recovery.
MX	Program being loaded is larger than available memory.	No recovery.
MS	Undefined names in the entry names table.	No recovery, unless system flag 6 has been set.
NX	No execution address has been specified (no main line program has been loaded).	No recovery.
RD	Input device error.	Reposition the record and toggle switch 3 to re-read.
BK	FORTTRAN labelled block error. A data block reference has been made to a cell lying outside the block.	No recovery.

If an RD error or a CK error occurs, the loader will attempt to re-read the record from a mass device. For a non-mass device (paper tape or card reader) the loader will type

"TOG3'

and wait for the operator to toggle switch 3. He must manually reposition the record, and when the switch is toggled, the record is re-read. If a second failure occurs on the same record, the loader aborts the job, as it does for any unrecoverable error.

APPENDIX N
ASSEMBLER/COMPILER ERROR REVIEW

N-9 RTOS ERROR MESSAGES

Real-Time X-Ray detects certain errors in syntax and in intent. These errors result in four-character error codes being listed on the list device, and on the teletype. Each code is listed here along with the possible events which may lead to its occurrence.

ICD? – Illegal Control Directives

1. : missing from beginning of directive line.
2. Unrecognized directive.
3. Invalid queue argument in QU and DQ directives.
4. Attempt to arm or disarm an I/O interrupt level with AL or DI directive.
5. Attempt to link task to I/O or non-existent interrupt with IC directive.
6. Attempt to connect to non-resident task (see NRES below).
7. Illegal input argument for ID or DA directive.
8. Attempts to assign a logical unit greater than 11 with I/O directive.

NUM? – Numeric Argument Error

Alphabetic argument given when numeric argument was expected.

ALF? – Alphabetic Argument Error

Numeric argument given when alphabetic argument expected.

I/O? – I/O Device Error

An input error has occurred on the System Input (SYSI) device.

FIL? – Disk File Specification Error

A file number larger than 3 or a file size too large for the disk has been given.

QFUL – Monitor Queue is Full

The tasks in the monitor queue buffer have filled it to capacity. Exit RTXRAY with the :XX directive to allow the queue to unload.

EROR – An Error Detected by the System

Compilation or assembly errors, FORTRAN Run-Time errors, and certain other system detectable errors are flagged by this message. Reference to the system list device will identify the error type.

APPENDIX N
ASSEMBLER/COMPILER ERROR REVIEW

N-9 RTOS ERROR MESSAGES (Cont)

HELP — A Disk Read Error in RTXRAY or in the System Load Routine

If an unrecoverable error occurs reading system processors from the disk, this message is output and the system re-tries the operation. In the event of repeated failure, the message will be output repeatedly. In this event, no system processing can occur until the disk has been repaired. If data has actually been destroyed on the disk, it will be necessary to re-install the system. This message is listed only on the teletype.

NRES — Attempt to Connect an Interrupt to a Task Which Has Not Been Made Resident

A task which is to be connected to an interrupt must first be made resident. If an attempt is made to connect to a non-resident task, this message is output followed by ICD?. The error is corrected by supplying the directives to make the task resident followed by the interrupt connection directive.

NOGO — Conditions for Load and GO Operation Are Not Met

This message will be listed if the .AG, :OB, or :FG directives are given with the binary output unit not assigned to a mass storage unit, or if the :GO directive is given when load and go operation has not been set up or has been aborted on account of assembly or compilation errors.

LTXT — Invalid Loader Text Encountered When Using :OB Directive

This message is listed if loader text read while copying from one medium to another using the :OB directive is not valid text.

SW1 — Sense Switch 1 is True

This switch suppresses binary output from system processors and must be switched false for all GO operations. This message reminds the operator to switch it off.

N-10 SYSGEN 1 ERROR MESSAGES

After each type-out of an error message, the computer halts. This gives the operator time to correct the error. He can then continue SYSGEN 1 by pushing the run button.

Below is a listing of errors typed out by SYSGEN 1, what each type-out means, and the operator's corrective procedure. In the event that the SYSGEN 1 input device is the teletype, the error message may be masked by data being read from the paper tape and printed at the same time that the error printout is occurring. To alleviate this inconvenience the error message is also duplicated as ASCII characters in the accumulator, when the computer halts.

APPENDIX N
ASSEMBLER/COMPILER ERROR REVIEW

N-10 SYSGEN 1 ERROR MESSAGES (Cont)

Errors

<u>Type Out</u>	<u>Meaning</u>	<u>Corrective Procedure</u>
FP	First processor was not 'DBST'	Load 'DBST'
SP	Second processor was not 'QLP'	Load 'QLP'
PQ	Third directive was not :NPQ	Input the :NPQ directive
FE	Format error. A binary record was encountered when a directive was expected.	Correct directive
IE	I/O error while reading last directive.	Re-read last directive
DE	There was an error on five consecutive disc reads or writes.	Stop, fix the disc, and start SYSGEN 1 over
UD	Disc is write protected.	Unprotect the disc.
CS	The checksum is wrong on last processor read.	If sense switch 0 is true when run button is pushed, the processor is ignored. If sense switch 0 is false when run button is pushed, the processor is included in the library with the checksum as calculated by SYSGEN 1.
ER	There was an I/O error while reading the last processor.	The processor is ignored. Load another one or re-try loading the bad one.
BA	Processor section 1 was not loaded first.	Load processor in correct sequence.

APPENDIX N
ASSEMBLER/COMPILER ERROR REVIEW

N-10 SYSGEN 1 ERROR MESSAGES (Cont)

Errors

<u>Type Out</u>	<u>Meaning</u>	<u>Corrective Procedure</u>
EF	Processor segments out of sequence. End-of-file was found while reading a program section.	Correct processor.
LR	The RTOS Monitor has not been loaded.	Load RTOS, then terminate job with the next directive.

N-11 SYSGEN 2 ERROR MESSAGES

Certain errors are detected by the system generator. In addition to the ordering error already noted, duplicate library names are optionally listed. These are not necessarily errors. Nonetheless, duplications may be inadvertent, and so they may be optionally flagged.

Ordering errors are listed as OE LIBENAME
Duplicates are listed as DL LIBENAME

where:

LIBENAME is the offending library name.

The remaining errors are catastrophic, and if unrecoverable, will cause the program in which they occur to be deleted from the library. When such an error occurs, one of the following error codes will be listed.

<u>Code</u>	<u>Error</u>	<u>Recovery</u>
LC	An unrecognized loaded code has been encountered.	None. The program is deleted.
CK	A checksum error has occurred.	The record may be re-read by re-positioning it. If the input is magnetic tape, this is done automatically. Otherwise, TOG3 will be listed. Re-position the record and toggle sense switch 3. If the error recurs, the program is deleted.
RD	or An input device error has occurred.	

APPENDIX N
ASSEMBLER/COMPILER ERROR REVIEW

N-11 SYSGEN 2 ERROR MESSAGES (Cont)

<u>Code</u>	<u>Error</u>	<u>Recovery</u>
DO	Directory overflow.	The directory is constructed in core. In an 8K computer there is space for approximately 1100 names. If this is exceeded, the "DO" message is listed, and the program which caused the overflow is deleted. System generation proceeds until the end of the input library is reached, accepting any program whose LIBR name will fit in the space vacated by the deleted programs, and deleting all others. The message "DO" will be repeated for each deleted program.
IR	Invalid record.	Only loader text records, hex correction (C in column 1) block equivalence (E in column 1), SYMBOLS card, and comment (blank in column 1) records are accepted by the system generator. These records are listed, if the listing option is chosen, and the hex corrections and block equivalence records are copied to the disk. Any other record causes the error message IR to be output, the record is listed if the listing option is chosen, and the record is ignored entirely.

APPENDIX O

MNEMONIC INDEX OF MACHINE INSTRUCTIONS

Mnemonic	OP Code	Operation	Time (μ s)	Cycles	Page
ADD	A	Add	1.8	2	2.4
AND	E	Logical AND	1.8	2	2.5
CAX	013	Copy Accumulaotr to Index	0.9	1	2.17
CEX	006	Copy Extension to Index	0.9	1	2.15
CLB	07	Compare Literal Byte	0.9	1	2.22
CLR	010	Clear Accumulator	0.9	1	2.16
CMB	4	Compare Byte	1.8	2	2.6
CMP	011	Complement Accumulator	0.9	1	2.16
CMW	F	Compare Word	1.8	2	2.5
CXA	014	Copy Index to Accumulator	0.9	1	2.17
CXE	007	Copy Index to Extension	0.9	1	2.15
DIN	02	Direct Input	1.8	2	2.23
DIV	0C0F	Divide*	9.0-9.9	10-11	2.24
DOT	03	Direct Output	1.8	2	2.23
DSB	003	Disable Interrupt	0.9	1	2.7
DXS	05	Decrement Index and Skip if < 0	0.9-1.8	1-2	2.22
ENB	002	Enable Interrupt	0.9	1	2.7
HLT	000	Halt	0.9	1	2.14
INR	001	Interrupt Return	2.7	3	2.7
INV	012	Invert Accumulator	0.9	1	2.16
IXS	04	Increment Index and Skip if ≥ 0	0.9-1.8	1-2	2.22
JMP	1	Unconditional Jump	0.9	1	2.1
JSX	2	Jump and Store Return in Index	0.9	1	2.3
LDB	5	Load Byte	1.8	2	2.6
LDW	8	Load Word	1.8	2	2.3
LDX	9	Load Index	1.8	2	2.4
LLB	06	Load Literal Byte	0.9	1	2.22
LPL	0D	Load Protect Lower*	0.9	1	2.23
LPU	0E	Load Protect Upper*	0.9	1	2.23
MSK	00A	Mask Interrupts	0.9	1	2.16
MPY	0B0F	Multiply*	6.3-9.0	7-10	2.24
NOP	0A00	No Operation**	0.9	1	2.21
ORE	D	Exclusive OR	1.8	2	2.5

*Optional

**Quasi Instruction

APPENDIX O

MNEMONIC INDEX OF MACHINE INSTRUCTIONS (Cont)

Mnemonic	OP Code	Operation	Time (μ s)	Cycles	Page
ORI	D	Inclusive OR	1.8	2	2.4
SAM	082	Skip on Accumulator Minus	0.9	1	2.17
SAO	083	Skip on Accumulator Odd	0.9	1	2.18
SAP	081	Skip on Accumulator Plus	0.9	1	2.17
SAZ	080	Skip on Accumulator Zero	0.9	1	2.17
SEQ	086	Skip on Compare Equal	0.9	1	2.19
SGM	005	Set Global Mode	0.9	1	2.15
SGR	088	Skip on Compare Greater	0.9	1	2.19
SLA	091	Shift Left Arithmetic	0.9-3.6	1-4	2.9
SLA D	093	Shift Left Arithmetic Double	0.9-3.6	1-4	2.10
SLC	0A5	Shift Left Circular	0.9-2.6	1-4	2.12
SLC D	0A7	Shift Left Circular Double	0.9-3.6	1-4	2.12
SLC L	0AD	Shift Left Circular Left Byte	0.9-3.6	1-4	2.14
SLC R	0AF	Shift Left Circular Right Byte	0.9-3.6	1-4	2.14
SLE	089	Skip on Compare Less Than or Equal	0.9	1	2.20
SLL	0A1	Shift Left Logical	0.9-3.6	1-4	2.10
SLL D	0A3	Shift Left Logical Double	0.9-3.6	1-4	2.11
SLL L	0A9	Shift Left Logical Left Byte	0.9-3.6	1-4	2.13
SLL R	0AB	Shift Left Logical Right Byte	0.9-3.6	1-4	2.13
SLM	004	Set Local Mode	0.9	1	2.15
SLS	084	Skip on Compare Less	0.9	1	2.18
SMB	008 or 009	Set Memory Bank**	0.9	1	2.15
SML	008	Select Memory Lower	0.9	1	2.8
SMU	009	Select Memory Upper	0.9	1	2.8
SNE	087	Skip on Compare Not Equal	0.9	1	2.19
SNO	08A	Skip on No Overflow	0.9	1	2.20
SRA	090	Shift Right Arithmetic	0.9-3.6	1-4	2.9
SRA D	092	Shift Right Arithmetic Double	0.9-3.6	1-4	2.10
SRC	0A4	Shift Right Circular	0.9-3.6	1-4	2.11
SRC D	0A6	Shift Right Circular Double	0.9-3.6	1-4	2.12
SRC L	0AC	Shift Right Circular Left Byte	0.9-3.6	1-4	2.13
SRC R	0AE	Shift Right Circular Right Byte	0.9-3.6	1-4	2.14

*Optional

**Quasi Instruction

APPENDIX O

MNEMONIC INDEX OF MACHINE INSTRUCTIONS (Cont)

Mnemonic	OP Code	Operation	Time (μ s)	Cycles	Page
SRC R	0AE	Shift Right Circular Right Byte	0.9-3.6	1-4	2.14
SRL	0A0	Shift Right Logical	0.9-3.6	1-4	2.10
SRL D	0A2	Shift Right Logical Double	0.9-3.6	1-4	2.11
SRL L	0A8	Shift Right Logical Left Byte	0.9-3.6	1-4	2.12
SRL R	0AA	Shift Right Logical Right Byte	0.9-3.6	1-4	2.13
SSE	08B	Skip on Sense External	0.9	1	2.20
SS0	08C	Skip on Sense Switch Zero False	0.9	1	2.20
SS1	08D	Skip on Sense Switch One False	0.9	1	2.21
SS2	08E	Skip on Sense Switch Two False	0.9	1	2.21
SS3	08F	Skip on Sense Switch Three False	0.9	1	2.21
STB	3	Store Byte	1.8	2	2.6
STW	7	Store Word	1.8	2	2.3
STX	6	Store Index	1.8	2	2.3
SUB	B	Subtract	1.8	2	2.4
SUS	00C	Select User State*	0.9	1	2.8
SXE	085	Skip on Index Even	0.9	1	2.18
SXM	0500	Skip on Index Minus**	0.9-1.8	1-2	2.19
SXP	0400	Skip on Index Positive**	0.9-1.8	1-2	2.18
UNM	00B	Unmask Interrupts	0.9	1	2.16

*Optional

**Quasi Instructions

APPENDIX P

OP CODE INDEX OF MACHINE INSTRUCTIONS

OP Code	Mnemonic	Operation	Time (μ s)	Cycles	Page
1	JMP	Jump	0.9	1	2.1
2	JSX	Jump and Store Return in Index	0.9	1	2.3
3	STB	Store Byte	1.8	2	2.6
4	CMB	Compare Byte	1.8	2	2.6
5	LDB	Load Byte	1.8	2	2.6
6	STX	Store Index	1.8	2	2.3
7	STW	Store Word	1.8	2	2.3
8	LDW	Load Word	1.8	2	2.3
9	LDX	Load Index	1.8	2	2.4
A	ADD	Add	1.8	2	2.4
B	SUB	Subtract	1.8	2	2.4
C	ORI	Inclusive OR	1.9	2	2.4
D	ORE	Exclusive OR	1.9	2	2.5
E	AND	Logical AND	1.9	2	2.5
F	CMW	Compare Word	1.8	2	2.5
000	HLT	Halt	0.9	1	2.14
001	INR	Interrupt Return	2.7	3	2.7
002	ENB	Enable Interrupt	0.9	1	2.7
003	DSB	Disable Interrupt	0.9	1	2.7
004	SLM	Set Local Mode	0.9	1	2.15
005	SGM	Set Global Mode	0.9	1	2.15
006	CEX	Copy Extension to Index	0.9	1	2.15
007	CXE	Copy Index to Extension	0.9	1	2.15
008 or 009	SMB	Set Memory Bank**	0.9	1	2.25
008	SML	Select Memory Lower	0.9	1	2.8
009	SMU	Select Memory Upper	0.9	1	2.8
00A	MSK	Mask Interrupts	0.9	1	2.16
00B	UNM	Unmask Interrupts	0.9	1	2.16
00C	SUS	Select User State*	0.9	1	2.8
010	CLR	Clear Accumulator	0.9	1	2.16
011	CMP	Complement Accumulator	0.9	1	2.16
012	INV	Invert Accumulator	0.9	1	2.16

*Optional

**Quasi Instruction

APPENDIX P

OP CODE INDEX OF MACHINE INSTRUCTIONS (Cont)

OP Code	Mnemonic	Operation	Time (μ s)	Cycles	Page
013	CAX	Copy Accumulator to Index	0.9	1	2.17
014	CXA	Copy Index to Accumulator	0.9	1	2.17
02	DIN	Direct Input	1.8	2	2.23
03	DOT	Direct Output	1.8	2	2.23
04	IXS	Increment Index and Skip if ≥ 0	0.9-1.8	1-2	2.22
0400	SXP	Skip if Index Positive**	0.9-1.8	1-2	2.19
05	DXS	Decrement Index and Skip if < 0	0.9-1.8	1-2	2.22
0500	SXM	Skip if Index Minus**	0.9-1.8	1-2	2.22
06	LLB	Load Literal Byte	0.9	1	2.17
07	CLB	Compare Literal Byte	0.9	1	2.17
080	SAZ	Skip on Accumulator Zero	0.9	1	2.17
081	SAP	Skip on Accumulator Plus	0.9	1	2.18
082	SAM	Skip on Accumulator Minus	0.9	1	2.18
083	SAO	Skip on Compare Accumulator Odd	0.9	1	2.18
084	SLS	Skip on Compare Less	0.9	1	2.19
085	SXE	Skip on Index Even	0.9	1	2.19
086	SEQ	Skip on Compare Equal	0.9	1	2.19
087	SNE	Skip on Compare Not Equal	0.9	1	2.19
088	SGR	Skip on Compare Greater	0.9	1	2.19
089	SLE	Skip on Compare Less or Equal	0.9	1	2.20
08A	SNO	Skip on No Overflow	0.9	1	2.20
08B	SSE	Skip on Sense External False	0.9	1	2.20
08C	SSO	Skip on Sense Switch 0 False	0.9	1	2.20
08D	SS1	Skip on Sense Switch 1 False	0.9	1	2.21
08E	SS2	Skip on Sense Switch 2 False	0.9	1	2.21
08F	SS3	Skip on Sense Switch 3 False	0.9	1	2.21
090	SRA	Shift Right Arithmetic	0.9-3.6	1-4	2.9
091	SLA	Shift Left Arithmetic	0.9-3.6	1-4	2.9
092	SRA D	Shift Right Arithmetic Double	0.9-3.6	1-4	2.10
093	SLA D	Shift Left Arithmetic Double	0.9-3.6	1-4	2.10
0A00	NOP	No Operation	0.9	1	2.21
0A0	SRL	Shift Right Logical	0.9-3.6	1-4	2.10

*Optional

**Quasi Instruction

APPENDIX P

OP CODE INDEX OF MACHINE INSTRUCTION (Cont)

OP Code	Mnemonic	Operation	Time (μ s)	Cycles	Page
0A1	SLL	Shift Left Logical	0.9-3.6	1-4	2.10
0A2	SRL D	Shift Right Logical Double	0.9-3.6	1-4	2.11
0A3	SLL D	Shift Left Logical Double	0.9-3.6	1-4	2.11
0A4	SRC	Shift Right Circular	0.9-3.6	1-4	2.11
0A5	SLC	Shift Left Circular	0.9-3.6	1-4	2.12
0A6	SRC D	Shift Right Circular Double	0.9-3.6	1-4	2.12
0A7	SLC D	Shift Left Circular Double	0.9-3.6	1-4	2.12
0A8	SRL L	Shift Right Logical Left Byte	0.9-3.6	1-4	2.12
0A9	SLL L	Shift Left Logical Left Byte	0.9-3.6	1-4	2.13
0AA	SRL R	Shift Right Logical Right Byte	0.9-3.6	1-4	2.13
0AB	SLL R	Shift Left Logical Right Byte	0.9-3.6	1-4	2.13
0AC	SRC L	Shift Right circular Left Byte	0.9-3.6	1-4	2.13
0AD	SLC L	Shift Left Circular Left Byte	0.9-3.6	1-4	2.14
0AE	SRC R	Shift Right Circular Right Byte	0.9-3.6	1-4	2.14
0AF	SLC R	Shift Left Circular Right Byte	0.9-3.6	1-4	2.14
0B0F	MPY	Multiply*	6.3-9.0	7-10	2.24
0C0F	DIV	Divide*	9.0-9.9	10-11	2.24
OD	LPL	Load Protect Lower*	0.9	1	2.23
OE	LPU	Load Protect Upper*	0.9	1	2.23

*Optional

**Quasi Instruction

APPENDIX Q
FUNCTIONAL INDEX OF MACHINE INSTRUCTION

Class	OP Code	Mnemonic	Operation	Time (μ s)	Cycles	Page
Load/Store	5	LDB	Load Byte	1.8	2	2.6
	8	LDW	Load Word	1.8	2	2.3
	9	LDX	Load Index	1.8	2	2.4
	3	STB	Store Byte	1.8	2	2.6
	7	STW	Store Word	1.8	2	2.3
	6	STX	Store Index	1.8	2	2.3
Arithmetic	A	ADD	Add	1.8	2	2.4
	B	SUB	Subtract	1.8	2	2.4
	OB0F	MPY	Multiply*	6.3-9.0	7-10	2.24
	OC0F	DIV	Divide*	9.0-9.9	10-11	2.24
Logical	C	ORI	Inclusive OR	1.8	2	2.3
	D	ORE	Exclusive OR	1.8	2	2.5
	E	AND	Logical AND	1.8	2	2.5
Compare	4	CMB	Compare Byte	1.8	2	2.6
	F	CMW	Compare Word	1.8	2	2.5
Jump	1	JMP	Unconditional Jump	0.9	1	2.1
	2	JSX	Jump and Store	0.9	1	2.3
Shift	090	SRA	Shift Right Arithmetic	0.9-3.6	1-4	2.9
	091	SLA	Shift Left Arithmetic	0.9-3.6	1-4	2.9
	092	SRA D	Shift Right Arithmetic Double	0.9-3.6	1-4	2.10
	093	SLA D	Shift Left Arithmetic Double	0.9-3.6	1-4	2.10
	0A0	SRL	Shift Right Logical	0.9-3.6	1-4	2.10
	0A1	SLL	Shift Left Logical	0.9-3.6	1-4	2.10
	0A2	SRL D	Shift Right Logical Double	0.9-3.6	1-4	2.11
	0A3	SLL D	Shift Left Logical Double	0.9-3.6	1-4	2.11
	0A4	SRC	Shift Right Circular	0.9-3.6	1-4	2.11
	0A5	SLC	Shift Left Circular	0.9-3.6	1-4	2.12

*Optional

**Quasi Instruction

APPENDIX Q

FUNCTIONAL INDEX OF MACHINE INSTRUCTION (Cont)

Class	OP Code	Mnemonic	Operation	Time (μ s)	Cycle	Page
	0A6	SRC D	Shift Right Circular Double	0.9-3.6	1-4	2.12
	0A7	SLC D	Shift Left Circular Double	0.9-3.6	1-4	2.12
	0A8	SRL L	Shift Right Logical Left Byte	0.9-3.6	1-4	2.12
	0A9	SLL L	Shift Left Logical Left Byte	0.9-3.6	1-4	2.13
	0AA	SRL R	Shift Right Logical Right Byte	0.9-3.6	1-4	2.13
	0AB	SLL R	Shift Left Logical Right Byte	0.9-3.6	1-4	2.13
	0AC	SRC L	Shift Right Circular Left Byte	0.9-3.6	1-4	2.13
	0AD	SLC L	Shift Left Circular Left Byte	0.9-3.6	1-4	2.14
	0AE	SRC R	Shift Right Circular	0.9-3.6	1-4	2.14
	0AF	SLC R	Shift Left Circular Right Byte	0.9-3.6	1-4	2.14
Control	000	HLT	Halt	0.9	1	2.14
	001	INR	Interrupt Return	2.7	3	2.7
	002	ENB	Enable Interrupt	0.9	1	2.7
	003	DSB	Disable Interrupt	0.9	1	2.7
	004	SLM	Set Local Mode	0.9	1	2.15
	005	SGM	Set Global Mode	0.9	1	2.15
	006	CEX	Copy Extension to Index	0.9	1	2.15
	007	CXE	Copy Index to Extension	0.9	1	2.15
	008	SML	Select Memory Lower	0.9	1	2.8
	009	SMU	Select Memory Upper	0.9	1	2.8
	00A	MSK	Mask Interrupts	0.9	1	2.16
	00B	UNM	Unmask Interrupts	0.9	1	2.16
	00C	SUS	Select User State*	0.9	1	2.8
	0D	LPL	Load Protect Lower*	0.9	1	2.23
	0E	LPU	Load Protect Upper*	0.9	1	2.23
	008 or 009	SMB	Set Memory Bank**	0.9	1	2.25
	0A00	NOP	No Operation**	0.9	1	2.21
Data	010	CLR	Clear Accumulator	0.9	1	2.15
	011	CMP	Complement Accumulator	0.9	1	2.16
	012	INV	Invert Accumulator	0.9	1	2.16
	014	CXA	Copy Index to Accumulator	0.9	1	2.17
	013	CAX	Copy Accumulator	0.9	1	2.17

* Optional

**Quasi Instruction

APPENDIX Q

FUNCTIONAL INDEX OF MACHINE INSTRUCTION (Cont)

Class	OP Code	Mnemonic	Operation	Time (μ s)	Cycle	Page
I/O	02	DIN	Direct Input	1.8	2	2.23
	03	DOT	Direct Output	1.8	2	2.23
Literal	04	IXS	Incremental Index and Skip if ≥ 0	0.9-1.8	1-2	2.22
	05	DXS	Decrement Index and Skip if < 0	0.9-1.8	1-2	2.22
	06	LLB	Load Literal Byte	0.9	1	2.22
	07	CLB	Compare Literal Byte	0.9	1	2.22
Skip	080	SAZ	Skip on Accumulator Zero	0.9	1	2.17
	081	SAP	Skip on Accumulator Plus	0.9	1	2.17
	082	SAM	Skip on Accumulator Minus	0.9	1	2.17
	083	SAO	Skip on Accumulator Odd	0.9	1	2.18
	084	SLS	Skip on Compare Less	0.9	1	2.18
	085	SXE	Skip on Index Even	0.9	1	2.18
	086	SEQ	Skip on Compare Equal	0.9	1	2.19
	087	SNE	Skip on Compare Not Equal	0.9	1	2.19
	088	SGR	Skip on Compare Greater	0.9	1	2.19
	089	SLE	Skip on Compare Less Than or Equal	0.9	1	2.20
	08A	SNO	Skip on No Overflow	0.9	1	2.20
	08B	SSE	Skip on Sense External	0.9	1	2.20
	08C	SSO	Skip on Sense Switch Zero False	0.9	1	2.20
	08D	SS1	Skip on Sense Switch One False	0.9	1	2.21
	08E	SS2	Skip on Sense Switch Two False	0.9	1	2.21
	08F	SS3	Skip on Sense Switch	0.9	1	2.21
	0400	SXP	Skip on Index Positive**	0.9-1.8	1-2	2.18
0500	SXM	Skip on Index Minus	0.9-1.8	1-2	2.19	

* Optional

**Quasi Instruction

**APPENDIX R
INDEX OF PSEUDO INSTRUCTIONS**

Mnemonic	Operation	Usage (SYM 1 or 2)	Page
BYTE	Byte Data	1,2	2.26
DATA	Full Word Data	1,2	2.26
DO	Do	2	2.30
DPI	Double Precision Decimal Integer	2	2.27
DPRL	Double Precision Real Number	2	2.28
ENDC	End of Conditional Processing	1,2	2.30
END	End	1,2	2.29
ENDP	End Processing	2	2.30
EPRL	Mid Precision Real Number	2	2.27
EQU	Equate	1,2	2.28
EXIT	Exit From Subroutine	2	2.31
FALS	False	1,2	2.30
IS	Is	2	2.28
LIBR	Library	1,2	2.29
LOAD	Load	1,2	2.29
NTRY	Entry	1,2	2.29
ORIG	Origin	1,2	2.29
PROC	Procedure Control	2	2.30
REAL	Two-word Floating Point Number	2	2.27
RES	Reserve	1,2	2.29
SUBR	Subroutine	2	2.31
TEXT	Text	2	2.27
TRUE	True	1,2	2.30

ABSOLUTE LOAD ROUTINE	
BASIC	8.22
FORMAT	8.19
STANDARD	8.22
ABSOLUTE LOADERS	8.19
ABSOLUTER, LINKING	8.27
ACCESS, DIRECT MEMORY	
CHANNEL	4.7
CONNECTOR	4.12
SYSTEM	1.28/4.7/6.1
ACCUMULATOR, (ACR)	1.6
ACCURACY, MATH LIBRARY	AP J
ACTION, END	8.37
ADDING USER CREATED I/O DRIVERS	8.80
ADDRESS	
GENERIC INSTRUCTION WITH NO.	2.15
LINKAGE	5.17
LOAD DATA CHAIN	5.85/5.100
READ MEMORY	5.84/5.99
SET MEMORY	5.74/5.79/5.96
SET RANDOM MODE	5.143
WORD FORMATS	1.11/1.15
ADDRESSES, WORD AND BYTE	1.17
ADDRESSING	
DIRECT, WORD AND BYTE	1.17-1.21
INDEXED	1.22
INDIRECT	1.27
ADVANCE TO END-OF-FILE	5.83/5.98
ADVANCED	
HARDWARE SYSTEM CONFIGURATION	1.31
LOADER MAP	8.25
SOFTWARE SYSTEM	1.30
ANALOG TO DIGITAL CONVERTER	5.143-5.152
FUNCTION CODES	5.143
STATUS	5.149
TIMING	5.150
WORD FORMATS	5.148
APPLICABLE PUBLICATIONS INDEX	AP B
ARITHMETIC, TWOS COMPLEMENT	1.9/AP F
ASCII, ASCII-8 CHARACTER CODES	5.40/AP G
ASR-33/39 TELETYPE	5.39-5.42
COMMANDS AND FUNCTIONS	5.41
STATUS	5.42
TIMING	5.42
ASSEMBLE	
FROM PAPER TAPE	9.9
KEYBOARD TO MEMORY	9.9
ASSEMBLER	
CONTROL DIRECTIVES	2.29/8.9
ERROR REVIEW	AP N
MANUAL MODE MESSAGE	9.46/9.51/9.58/9.63
REVIEW, SYM-I, SYM-II	8.12
STATEMENT FORMATS AND SYNTAX	1.11/1.16
XRAY EXECUTIVE DIRECTIVES	AP M-2
ASSEMBLERS	1.34/8.4-8.12
ASSIGNMENT	
DATA BIT, PERIPHERAL	5.8
DIO MAGNETIC TAPE DEVICE	5.111
INTERRUPT	3.3
PEAT	5.26
STATUS BIT, PERIPHERAL	5.9
TAPE UNIT NUMBER	5.79/5.96/5.109
ATP, FORTRAN FUNCTION CALLS	AP J
AUTOMATIC PRIORITY INTERRUPT	1.28/3.1
AUTOMATIC OPERATION, MAGNETIC TAPE	
DMA 9-TRACK	5.92
DMA 7-TRACK	5.107
AUTOMATIC OPERATION, DIGITAL PLOTTER	5.131
BACKSPACE	
INPUT/OUTPUT SOFTWARE	5.37
ONE RECORD, MAGNETIC TAPE	
DIO	5.112
DMA 9-TRACK	5.84
DMA 7-TRACK	5.99
BANK, SET MEMORY	1.19/2.25
BASIC	
ABSOLUTE LOAD ROUTINE	8.22
HARDWARE CONFIGURATION	1.30
INPUT/OUTPUT SOFTWARE CALLS	5.32
OPERATING SYSTEM	8.62
RELOCATABLE LOADER	8.24/9.7
ERROR MESSAGES	AP N-7
SYMBOLIC PROGRAM EDITOR	9.21
SYSTEM LOADER MAP	8.27
XRAY	1.33/8.62/9.7
DIRECTIVES	AP M-2
BATCH PROCESS	
CONVERSATIONAL FORTRAN	8.15
MULTIPROGRAMMING	8.73
BINARY	
CARD PUNCH DRIVER, WRITE	8.37
CARD READER DRIVER, READ	8.34
FORMAT, MAGNETIC TAPE	5.96
NUMBER SYSTEM	AP E

BIT
 CARD ROW-ACCUMULATOR CORRES. 5.49
 DATA, ASSIGNMENT 5.8
 STATUS, ASSIGNMENT 5.9

 BLOCK DIAGRAM
 HARDWARE 1.2
 SOFTWARE 1.3

 BOOTSTRAP
 INITIAL LOADER 9.6
 LOADERS 8.23
 LOADING 7.4
 CARD READER 7.6
 DIO MAGNETIC TAPE 5.117/7.6
 DISC 5.75
 DMA MAGNETIC TAPE 5.89/5.103
 PROGRAM 7.4
 TELETYPE 7.5
 SYSTEM LOADING
 MPS 9.61
 MTS 9.49
 RTS 9.55
 STANDARD 9.43

 BUFFER
 DATA, OVERSPILL, LINE PRINTER, . . . 5.68
 MEMORY REGISTER 1.6

 BUFFERED
 DIGITAL INPUT CHANNEL 5.139-5.142
 DIGITAL OUTPUT CHANNEL 5.139-5.142

 BURST METHOD, INPUT/OUTPUT 5.13

 BYTE
 ADDRESSING 1.18-1.25
 INSTRUCTIONS WITH BYTE OPERAND 2.21
 PAGE NUMBER CONVERSION 1.18
 REFERENCE INSTRUCTION 2.6

 CABLE
 DIO 4.7
 DMA 4.8

 CABLING 4.7-4.14

 CALLING MATH SUBROUTINES 8.75

 CALLS
 ATP FUNCTION AP J
 INPUT/OUTPUT SOFTWARE (IOS) 5.25-5.33

 CARD PUNCH 5.43-5.58
 CONTROLS AND INDICATORS 5.57
 DRIVER 8.37
 WRITE BINARY 8.38
 WRITE END-OF-FILE 8.38
 WRITE HOLLERITH 8.38
 FUNCTIONS 5.50
 HIGH-SPEED SKIP 5.52
 OPERATION 5.54
 ROW-ACCUMULATOR BIT CORRESPONDENCE . . 5.49
 STATUS 5.52
 TIMING 5.53

CARD READER 5.43-5.60
 BOOTSTRAP OPERATION 7.6
 CONTROLS AND INDICATORS
 400 CPM 5.60
 1000 CPM 5.56
 DRIVER 8.34
 READ BINARY 8.35
 READ HOLLERITH 8.35
 READ SPECIAL 8.35
 FUNCTIONS 5.49
 OPERATION 5.59
 ROW-ACCUMULATOR BIT CORRESPONDENCE . . 5.49
 STATUS 5.52
 STOP INTERRUPTS FOR CURRENT CARD 5.49
 TIMING 5.53

 CARDS
 DECKS 9.1
 INITIAL LOADER 9.5

 CARRIAGE CONTROL, PLOTTER 5.125

 CENTRAL PROCESSOR, (CPU) 1.6

 CHAIN, DATA
 DMA 9-TRACK 5.85
 DMA 7-TRACK 5.100

 CHAINING, DMA MAGNETIC TAPE DRIVER . . . 8.43

 CHANGE OF DIRECTION, DIO MAG TAPE . . . 5.116

 CHANGING RIBBON, LINE PRINTER 5.70

 CHANNEL
 BUFFERED DIGITAL INPUT/OUTPUT 5.139
 DIRECT MEMORY ACCESS (DMA) 1,28/4,1-4,8
 TELETYPE MULTIPLEXER 5.135

 CHARACTER
 ASCII, ASCII-8 CODES 5.40/AP G
 INTERRUPT METHOD, SERVICE 5.17
 LINE PRINTER
 CHARACTER SET 5.62/AP G
 LOAD (TRANSFER) ONE 5.64
 SLEW 5.63
 PAPER TAPE READER/PUNCH
 OUTPUT, PUNCH 5.43
 TRANSFER, READER 5.43
 SYM-II AP K-2
 TELETYPE MULTIPLEXER, OUTPUT 5.136

 CHARACTERISTICS
 BUFFERED DIGITAL INPUT/OUTPUT 5.139/5.140
 CONVERSATIONAL FORTRAN AP K-5
 DIGITAL PLOTTER 5.123
 DIO MAGNETIC TAPE 5.110
 DMA 7-TRACK MAGNETIC TAPE 5.95
 DMA 9-TRACK MAGNETIC TAPE 5.80
 FORTRAN ASA AP K-3
 POWER FAIL-SAFE 6.2
 REAL-TIME FORTRAN IV AP K-4
 SYM-I AP K-1
 SYM-II AP K-2

CHART PAPER, DIGITAL PLOTTER, 5.128-5.130
 CHECK, MEMORY PARITY, 6.1
 CHECKOUT, DIGITAL PLOTTER, 5.130
 CLASSES OF INSTRUCTIONS, 2.1
 CLEANING, LINE PRINTER, 5.70
 CLOCK
 MACHINE TIMING, 1.9
 TIME-OF-DAY, 5.157-5.162
 CODES
 CONVERTER INPUT, 5.154
 FUNCTION (SEE
 INDIVIDUAL DEVICES), 5.1-5.7/AP L
 PEN, DRUM, CARRIAGE CONTROL, 5.126
 RELOCATABLE LOADER TEXT, 8.25
 STATUS (SEE
 INDIVIDUAL DEVICES), 5.9/AP L
 CODING
 DIO, 5.13-5.21
 GENERAL, 8.6
 COMMANDS (SEE INDIVIDUAL DEVICES
 OR INSTRUCTIONS)
 COMMENT, 1.11
 COMPARISON INDICATORS, 1.9
 COMPARISON, RTOS=MPS, 8.73
 COMPILERS, 1.34/8.13-8.18
 CONVERSATIONAL FORTRAN 8.15/9.27/AP M=2
 ERROR REVIEW, AP N
 FORTRAN IV, REAL-TIME, 8.13/AP M=2
 COMPLEMENT, TWOS, AP F
 COMPLEX, FORTRAN LIBRARY, AP J
 COMPUTER SPECIFICATION, 706, 1.4
 CONCURRENT PROGRAMMING, MPS, 8.71
 CONDITIONAL PROCESSING, 2.30
 WITHIN PROCEDURE, 8.10
 CONFIGURATION
 BY PART NUMBER, 9.2
 HARDWARE, 1.30-1.32/8.1
 SOFTWARE SYSTEM, 8.1
 CONNECTION
 INTERVAL TIMER INPUT JUMPER, 5.161
 RTOS RESIDENCE AND INTERRUPT, 8.70
 CONNECTORS
 DIO, 4.11
 DMA, 4.12

CONSOLE
 TELETYPE, 5.42
 706, 7.2
 CONSTANTS, REFERENCE AND DATA, AP D
 CONTINUE
 LAST OPERATION
 DMA 9-TRACK, 5.82
 DMA 7-TRACK, 5.97
 LOAD, 9.7
 PROCESSING, INTERRUPT METHOD, 5.21
 CONTINUOUS FUNCTIONS, DIO MAG TAPE, 5.113
 CONTROL
 ASSEMBLER, DIRECTIVES, 2.29/8.8
 DATA, OPERATING SYSTEM, 8.62
 FUNCTIONS (SEE INDIVIDUAL DEVICES)
 INTERRUPT, XRAY, AP M=2
 INTERRUPT AND MASK, 5.17
 JOB, OPERATING SYSTEM, 8.59
 MPS SYSTEM, 8.72
 PANELS (SEE INDIVIDUAL DEVICES)
 CPU, 7.2
 PROGRAM DEVELOPMENT, 8.59
 CONTROLLERS (SEE INDIVIDUAL DEVICES)
 CONVERSATIONAL FORTRAN
 BATCH PROCESS USE, 8.17
 CHARACTERISTICS, AP K=5
 COMPILE AND GO, 9.27
 COMPILER, 8.16
 EDIT, 9.27
 ERROR MESSAGES, AP N=5/AP N=6
 INITIALIZATION, 9.27
 INTRODUCTION, 1.35-1.36
 IN INTERACTIVE USE, 8.17
 LIBRARY, AP J
 OPERATION, 9.27
 PREPARE PAPER TAPE, 9.27
 RUN-TIME
 ERROR MESSAGES, AP N=6
 EXECUTION, 9.27
 SYSTEM, 8.16
 CONVERSION
 HEXADECIMAL TO
 DECIMAL FRACTION, AP E
 NEGATIVE INTEGER, AP E
 POSITIVE INTEGER, AP E
 TIMING, AP I
 WORD AND BYTE PAGE NUMBER, 1.18
 CONVERTERS
 ANALOG TO DIGITAL, 5.143-5.152
 DIGITAL TO ANALOG, 5.153-5.156
 COUNTER, PROGRAM, 1.6/7.1
 CURRENT, STOP INTERRUPT FOR CARD, 5.49

DATA (SEE INDIVIDUAL DEVICES)
 CONTROL OPERATING SYSTEM 8.62
 DISPLAY AND ENTRY, MANUAL 7.1
 REFERENCE, MISCELLANEOUS AP D
 WORD FORMATS 1.10-1.14

DAY, TIME-OF-, CLOCK 5.157-5.161

DEBUG, TRACE/ 8.80

DEBUGGING, SYMBOLIC 1.34

DECIMAL
 CONVERSION TO HEXADECIMAL, AP E
 NUMBER SYSTEM, AP E

DECK, EXECUTING FORTRAN IV OBJECT 8.15

DECKS, PAPER TAPES AND CARD 9.1

DECREMENTING INDEX REGISTER 1.22

DEFINITION
 LOGICAL UNIT 5.26
 SYMBOL AND DATA, 2.26/8.9

DENSITY, DIO MAGNETIC TAPE, 5.109

DESCRIPTION
 CPU REGISTERS, 1.5
 GENERAL SYSTEM, 1.1-1.37
 PROGRAMMING SYSTEMS, 8.1-8.92

DESIGN, CONVERSATIONAL FORTRAN SYSTEM 8.16

DETECTION, PARITY AND ERROR 5.114

DEVELOPMENT, PROGRAM CONTROL, 8.59

DEVICE
 CABLING OF PERIPHERAL, 4.8/4.14
 DRIVER REFERENCES, 5.38
 I/O SELECTION, 5.17
 STATUS 5.37

DIAGNOSTICS, SENSOR 1.37/8.84

DIAGRAM, HARDWARE/SOFTWARE BLOCK, 1.2/1.3

DIGITAL
 ANALOG TO, CONVERTER 5.143-5.152
 BUFFERED I/O CHANNEL 5.139-5.142
 PLOTTER, 5.123-5.134
 TO ANALOG CONVERTER, 5.153-5.156

DIRECT
 ADDRESSING 1.18-1.21
 INPUT/OUTPUT (DIO) 1.28/4.1-4.6
 MEMORY ACCESS (DMA), 1.28/4.7-4.12/8.1
 PROGRAM LOADING, 7.6

DIRECTION, CHANGE OF, 5.116

DIRECTIVES
 ASSEMBLER CONTROL, 2.29/8.9
 MTO5, RTOS OPERATOR INTERRUPT 9.49/9.55
 SYMBOLIC PROGRAM EDITOR, 9.22
 XRAY EXECUTIVE, SUMMARY, AP M-2

DISC MEMORY 5.73-5.78
 DRIVER 8.39
 FILES, 8.41
 FUNCTIONS, 5.74
 LOADERS, 1.36/8.27
 SYSTEMS GENERATION AND ORGAN 8.81

DISC-BASED LIBRARY, 8.81

DISCONNECT (SEE INDIVIDUAL DEVICES) AP L

DISPLAY, MEMORY 7.1-7.3

DMA (SEE DIRECT MEMORY ACCESS)

DMA 7-TRACK MAGNETIC TAPE (SEE
 MAGNETIC TAPE)

DIO 9-TRACK MAGNETIC TAPE (SEE
 MAGNETIC TAPE)

DIO10, 5.33

DOT INSTRUCTION 4.1

DOUBLE
 PRECISION
 FORTRAN IV LIBRARY, AP J
 INTEGER 1.10
 WORD INSTRUCTIONS, 2.24

DRIVER(S) 5.25/8.31-8.58
 ADDING USER-CREATED, 8.50
 CARD PUNCH 8.37
 CARD READER, 8.34
 DEVICE REFERENCES, 5.38

DISC 8.39
 INPUT/OUTPUT SOFTWARE, 5.25
 LINE PRINTER, 8.38
 MAGNETIC TAPE
 DIO 8.44
 DMA 8.39
 PAPER TAPE, HIGH-SPEED 8.31
 PLOTTER, DIGITAL 8.47
 TELETYPE 8.31

DRUM
 LINE PRINTER, SPEEDS 5.66
 PLOTTER, DIGITAL 5.125

EDIT, CONVERSATIONAL FORTRAN, 9.27

EDITORS 1.34/8.78
 SYMBOLIC PROGRAM 8.78/9.21

EIGHT CHANNEL PAPER TAPE FORMAT . . . 5.40
ELECTRICAL SPECIFICATIONS . . . 1,5/4.10
END ACTION. 4.37
END-OF-FILE
ADVANCE TO
DIO MAGNETIC TAPE (SEARCH) . . . 5.113
DMA 9-TRACK MAGNETIC TAPE . . . 5.83
DMA 7-TRACK MAGNETIC TAPE . . . 5.98
IOS(SEEK) 5.37
CARD PUNCH, DRIVER 8.38
WRITE
DIO MAGNETIC TAPE 5.112
DMA 9-TRACK MAGNETIC TAPE . . . 5.83
DMA 7-TRACK MAGNETIC TAPE . . . 5.98
IOS 5.37
END-OF-TRANSMISSION 5.13
ENTRY
MANUAL 7.1-7.4
NAMES TABLE. 9.7
EQUATION, INSTRUCTION TERMS AND SYMBOL 2.2
EQUIPMENT
PERIPHERAL ASSIGNMENT TABLE (PEAT) 5.26
PERIPHERAL, OPERATION, . . . 5.1-5.161
EQUIVALENTS, CHARACTER CODES, . . . AP G
LOADING. 9.5
MESSAGE, REVIEW, AP N
ASSEMBLER/COMPILER, AP N=1 THRU N=6
RELOCATABLE LOADER, . . . AP N=7/N=8
RTOS, AP N=9
SYSGEN1,SYSGEN2 AP N=10/N=11
MESSAGE, SYSTEM
MPS, SYM=II 9.62
RTOS, SYM=II, 9.51
RTOS, SYM=II, 9.57
PARITY, DIOMAGNETIC TAPE 5.114
RATE
DISC MEMORY 5.75
DMA 9-TRACK MAGNETIC TAPE . . . 5.88
DMA 7-TRACK MAGNETIC TAPE . . . 5.103
EXECUTE, FORTRAN IV 8.13
EXECUTION
CONVERSATIONAL FORTRAN 9.27
PLOTTER DRUM AND CARRIAGE TIMES 5.125
PROGRAM. 7.6
EXECUTIVE(S)
MONITORS AND 1.31
RTOS OPERATING SYSTEM, 8.66
RTOS OPERATING SYSTEM, 8.70
STANDARD OPERATING SYSTEM, . . . 8.65
XRAY, DIRECTIVES REVIEW, AP M=2

EXIT
CURRENT TASK
RTOS. 9.50
RTOS. 9.56
STANDARD, 9.44
FROM USER STATE, 6.5
EXPANSION
INTERRUPT, 6.1
MEMORY 6.5
EXTENDED SYSTEM
HARDWARE CONFIGURATION 1,31/8.1
LOADER MAP 8.29
SOFTWARE 1,30/8.1
EXTENSION
REGISTER 1,9/1.17-1.20
RELOCATABLE LIBRARY, 8.84
EXTERNAL
STRINGS. 8.24
WORD FORMATS 8.77
FACTORIALS. AP D
FAIL, POWER=FAIL SAFE 6.2
FILE(S)
DISC 8.41
INPUT/OUTPUT TABLE (SEE DRIVERS) . 5.31
PROTECTION
DIO MAGNETIC TAPE 5.116
DMA 9-TRACK MAGNETIC TAPE . . . 5.88
DMA 7-TRACK MAGNETIC TAPE . . . 5.103
FILLED, RTOS QUEUE, 9.58
FILTER, LINE PRINTER, 5.70
FIOT (SEE FILE INPUT/OUTPUT TABLE)
FIXED POINT DATA, 1.10
FLOATING POINT DATA 1.10
FORMAT(S) (SEE INDIVIDUAL DEVICES)
ABSOLUTE PROGRAM 8.19
ADDRESS WORD 1.11
ASSEMBLER SOURCE STATEMENT 1.11
DATA WORD, 1.14
INSTRUCTION WORD 1.13/1.15
PAPER TAPE, SYM=I, 9.23
FORTRAN
ATP FUNCTION CALLS AP J
CONVERSATIONAL 1.34-1.36/8.15
OPERATION 9.27
PAUSE MESSAGES 9.46/9.51/9.57/9.63
REAL-TIME FORTRAN IV 1.35-1.37/8.13
FUNCTION (SEE INDIVIDUAL DEVICES) AP L
IOS, TABLE 5.27

FUNCTIONAL INDEX, MACH. INSTRUCTIONS AP Q
 FUNCTIONS
 FORTRAN IV LIBRARY AP J
 GENERAL SYSTEM DESCRIPTION, . . . 1.1-1.37
 GENERATION, SYSTEM, 1.33/1.36/8.81
 GENERIC INSTRUCTIONS, 2.7/2.15/2.21
 GLOBAL MODE INDEXING, 1.24-1.26
 GLOSSARY, AP A
 GO
 COMPILE AND, CONVER, FORTRAN . . . 9.27
 EXECUTE AND GO, FORTRAN IV 8.15
 HALT INSTRUCTION, 2.14
 HARDWARE
 BLOCK DIAGRAM, 1.2
 CONFIGURATIONS, 1.30-1.32
 DATA WORD FORMATS, 1.14
 INSTRUCTIONS, 2.1-2.24
 MULTIPLY/DIVIDE, 2.24/6.1
 HEXADECIMAL
 CHARACTER CODES, AP G
 CONVERSION TABLES, AP E
 HIGH-SPEED PAPER TAPE PUNCH, . . . 5.43-5.48
 DRIVER, 8.31
 HIGH-SPEED PAPER TAPE READER, . . . 5.43-5.48
 DRIVER, 8.31
 HOLD CLOCK, 5.158
 HOLLERITH CHARACTER CODES, AP G
 INCREMENTING, INDEX REGISTER, 1.22
 INDEX
 APPLICABLE PUBLICATIONS, AP B
 MACHINE INSTRUCTIONS
 FUNCTIONAL, AP Q
 MNEMONIC, AP G
 OP CODE, AP P
 PSEUDO INSTRUCTIONS, AP R
 REGISTER, 1.6/1.22
 INDEXED ADDRESSING, 1.22
 INDEXING
 GLOBAL MODE, 1.22-1.25
 LOCAL MODE, 1.22-1.25
 INDICATOR(S)
 COMPARISON, 1.9
 CONTROLS (SEE INDIVIDUAL DEVICES)
 LOCAL/GLOBAL, 1.9
 OVERFLOW, 1.9

INDIRECT ADDRESSING, 1.27
 INITIAL LOADER, 8.22
 BOOTSTRAP, 9.6
 CARDS, 9.5
 OPERATION, 9.5
 PAPER TAPE, 9.5
 INITIALIZATION
 CONVERSATIONAL FORTRAN, 9.27
 SYM-I/PREP, 9.9
 INPUT/OUTPUT
 DIO CHANNEL, 4.1
 DRIVERS, 8.31-8.58
 FILE TABLE (F10T), 5.31
 MPS SYSTEM, 8.73
 SOFTWARE, 5.25-5.38
 INSTALLATION
 HARDWARE, AP H
 RTOS SYSTEM MESSAGE, 9.98
 INSTRUCTION(S)
 DIN/DOT, 4.1
 HARDWARE, 2.1-2.29
 INDEX OF, AP G/AP P/AP Q
 PSEUDO
 DEFINITIONS, 2.26/8.8
 INDEX, AP R
 INTEGER
 DOUBLE PRECISION, 1.10
 FORTRAN LIBRARY FUNCTIONS, AP E
 HEXADECIMAL CONVERSION, AP E
 INTERACTIVE USE, CONVER, FORTRAN, . . . 8.17
 INTERFACE
 CABLING, 4.7
 DIO/DMA TIMING, 4.5/4.10
 PLOTTER ROUTINE, 8.49
 INTERNAL
 NUMBER REPRESENTATION, 1.9
 WORD FORMATS, 8.77
 INTERRUPT
 AUTOMATIC PRIORITY, 1.28/3.1-3.3
 OPERATION, 3.2
 PROGRAMMING, 3.1/5.17-5.22
 EXPANSION, 6.1
 MEANING FOR DEVICES (SEE
 INDIVIDUAL DEVICES)
 RTOS OPERATOR INTERRUPT, 9.49
 RANKING AND ASSIGNMENT, 3.3
 RTOS OPERATOR INTERRUPT, 9.55
 XRAY DIRECTIVE CONTROL, AP M-2
 INTERVAL TIMING, 5.159-5.160

INTERVENTION, MANUAL
 MTOS 9,50
 RTOS 9,56
 STANDARD 9,44
 SPEED-DENSITY 5,111

I/O (SEE INPUT/OUTPUT SOFTWARE)

JOB CONTROL OPERATING SYSTEM. 8,59

JOB DIRECTIVES, XRAY. AP M=2

JUMPER CONNECTIONS, INTERVAL TIMER. 5,161

KEYBOARD. 5,39
 ASSEMBLE FROM. 9,9

LAST PUNCH. 5,50

LAYOUT, MEMORY. 1,17

LEADER, READING/PUNCHING. 5,47

LEAVING THE USER STATE. 6,5

LETTERS, REVISION 9,1

LEVEL, INTERRUPT OPERATION. 3,2

LIBRARY
 CONVERSATIONAL FORTRAN 1,36/AP J
 FORTRAN-IV 1,37/AP J
 MATH 1,37/8,75/AP J
 RELOCATABLE EXTENSION. 8,84
 SYSTEM
 MTOS. 8,66
 STANDARD. 8,64
 PROCESSOR 8,82
 RTOS DISC-BASED 8,69

LINE PRINTER. 5,61-5,72
 CHARACTER SET. 5,62/AP Q
 COMMANDS AND FUNCTION CODES. 5,65
 CONTROLS AND INDICATORS
 1000 LPM. 5,71
 360 AND 600 LPM 5,69
 DRIVER 8,38
 OPERATION. 5,68
 SLEW CHARACTERS. 5,63
 SPECIFICATION. 5,61
 STATUS 5,64
 TIMING 5,67

LINKAGE
 ADDRESS SET-UP 5,17
 SUBROUTINE 1,22
 SYSTEM TABLE 5,34

ERROR(S)
 DMA MAGNETIC TAPE, RECOVERY. 8,40

LINKING ABSOLUTER 8,27

LIST DIRECTIVE, XRAY EXECUTIVE. AP M=2

LOAD
 ABSOLUTE, ROUTINE. 8,22
 BASIC 8,22
 MTOS. 9,50
 RTOS. 9,56
 STANDARD. 9,44

BOOTSTRAP
 DATA CHAIN ADDRESS AND MODE. 5,85/5,100
 DISC MEMORY 5,75
 DMA 9-TRACK MAGNETIC TAPE 5,89
 DMA 7-TRACK MAGNETIC TAPE 5,103
 MANUAL MEMORY. 5,93/5,107
 PROBLEM MESSAGE, RTOS. 9,57

LOADER
 INITIAL. 8,22
 BOOTSTRAP 9,6
 CARDS 9,5
 PAPER TAPE. 9,5

MAP
 BASIC 8,27
 EXTENDED-ADVANCED 8,29
 STANDARD-MTOS 8,28
 RECORD BLOCKING, TEXT. 8,24

RELOCATING
 BASIC 8,24/9,7
 DISC. 8,27
 STANDARD. 8,26
 TEXT CODES. 8,23

LOADERS 1,36/8,19-8,30
 ABSOLUTE 8,19
 BOOTSTRAP. 8,23

LOADING
 BOOTSTRAP. 7,4
 DIRECT PROGRAM 7,6
 EXTENSION REGISTER 1,19
 MATH SUBROUTINES 8,78
 TAPE (SEE INDIVIDUAL DEVICES 5,91/5,106

LOCAL MODE INDEXING 1,23/1,24

LOCAL/GLOBAL INDICATOR. 1,9

LOCKOUT SWITCH. 7,7

LOGICAL
 OPERATIONS, FORTRAN IV AP J
 UNIT DEFINITIONS 5,26
 WORD 1,10

MACHINE INSTRUCTIONS INDEX, AP Q/AP P/AP Q

MAGNETIC TAPE
 DISC. 5,109-5,122
 BOOTSTRAP 7,6
 CHARACTERISTICS 5,110
 COMMANDS AND FUNCTIONS. 5,115
 CONTROLS AND INDICATORS 5,119
 DATA WORDS. 5,113
 DEVICE ASSIGNMENT 5,111
 DRIVER. 8,44
 SPEED-DENSITY 5,111

STATUS,	5,116	MINIVERTER,	5.44/5,150/5,151
TIMING,	5,118	MISCELLANEOUS REFERENCE DATA,	AP D
DMA 9-TRACK,	5,79-5,94	MNEMONIC INDEX OF MACHINE INSTRUCT, .	AP O
CHARACTERISTICS	5,80	MODE,	
CONTROL PANEL	5,90	DATA CHAIN, MAGNETIC TAPE,	5,85/5,100
FUNCTIONS	5,79	GLOBAL INDEXING,	1,24
INTERRUPT MEANING	5,89	LOCAL INDEXING	1,23
STATUS,	5,88	PREP	9,9/9,27
DMA 7-TRACK,	5,95-5,108	RANDOM, A-D,	5,143
CHARACTERISTICS	5,95	RUN,	7,6
CONTROL PANEL	5,105	SINGLE COMMAND	7,7
FUNCTIONS	5,96	MODIFICATION, PROGRAM	9,24
INTERRUPT MEANING	5,104	MONITORS AND EXECUTIVES	1,31
STATUS,	5,103	RESIDENT	
DRIVER, DMA,	8,39	MTOS,	8,65
OPERATING SYSTEM	1,33/8,65/9,49	RTOS,	8,68
MANIFEST, SOFTWARE SHIPPING	9,1	STANDARD,	8,64
MANUAL		MPS (SEE MULTIPROGRAMMING SYSTEM)	
DATA DISPLAY AND ENTRY	7,1	MTOS (SEE MAGNETIC TAPE	
INTERVENTION, SYSTEM		OPERATING SYSTEM)	
MTOS,	9,50	MULTI-LEVEL INDIRECT ADDRESSING . . .	1,27
RTOS,	9,56	MULTIPROGRAMMING SYSTEM	1,33/8,71/9,61
STANDARD,	9,44	BATCH PROCESSOR,	8,73
LOCKOUT SWITCH	7,7	CONCURRENT PROGRAMMING	8,71
PROGRAM ENTRY,	7,4	CONTROL,	8,72
MAP, MEMORY		INPUT/OUTPUT	8,73
BOOTSTRAP,	7,5	RTOS COMPARISON,	8,73
EXTENDED-ADVANCED	8,29	SOFTWARE SUMMARY	9,61
LOADER		SYSTEM CONTROL	9,61
BASIC	8,27	SYSTEM MESSAGES,	9,61
STANDARD-MTOS	8,28	SYSTEM PROTECTION,	8,72
MASK, INTERRUPT	5,17	MULTIPLEXER, TELETYPE	5,135-5,138
MASS STORAGE,	8,4	MULTIPLY/DIVIDE	
MATH LIBRARY,	1,37/8,75/AP J	HARDWARE	2,24/6,1
MEMORY		SOFTWARE	AP J
ADDRESSING	1,17-1,27	MULTIPRECISION FLOATING POINT DATA, .	1,10
BANK, SET,	1,19/2,25	MULTIVERTER,	5,145/5,152
BUFFER REGISTER,	1,6	NAMES, ENTRY PRINTOUT	9,7
DISPLAY AND ENTRY,	7,3	NEGATIVE NUMBERS,	1,9/AP E/AP F
KEYBOARD, ASSEMBLE TO,	9,9	NON-MASS STORAGE,	8,4
MAP (SEE MAP, MEMORY)		NUMBERS	
OPTIONS		INTERNAL	1,9
DIRECT ACCESS (DMA)	1,28/4,7/6,1	PAGE	1,17
EXPANSION	6,5	SYSTEMS,	AP D/AP E
PARITY CHECK,	6,1	OFFSET CARD IN PROCESS,	5,49/5,50
PROTECT	6,2	ONE PASS ASSEMBLER,	8,6
MERGE, SORT PROGRAM	8,80		
MESSAGES			
ERROR,	AP N		
MPS,	9,62		
MTOS	9,51		
RTOS	9,57		
STANDARD	9,45		
MID-PRECISION, FORTRAN IV LIBRARY, .	AP J		

OP CODE INDEX OF MACHINE INSTRUCTIONS AP P	
OPEN, IOS	5.33
OPERAND, 4-BIT AND BYTE	2,7/2,21
OPERATING SYSTEMS	8.59-8.74
BASIC	8.62
DATA CONTROL	8.62
JOB CONTROL	8.59
MAGNETIC TAPE (MTOS)	1,33/8.65
REAL-TIME (RTOS)(DISC)	1,33/8.66
STANDARD	8.63
OPERATION	
DEVICE (SEE INDIVIDUAL DEVICES)	
SYSTEM (SEE INDIVIDUAL SYSTEMS)	
OPERATOR INTERRUPT	
MTOS	9.49
RTOS	9.55
OPTIONS	6.1-6.5
DIRECT MEMORY ACCESS	6.1
HARDWARE MULTIPLY/DIVIDE	6.1
INTERRUPT EXPANSION	6.1
MEMORY EXPANSION	6.5
MEMORY PARITY CHECK	6.1
MEMORY PROTECT	6.2
POWER FAIL-SAFE	6.2
ORGANIZATION, DISC	8.81
OUTPUT (SEE INPUT/OUTPUT)	
OVERFLOW INDICATOR	1.9
OVERPRINTING	5.68
OVERSPILL, LINE PRINTER DATA BUFFER	5.68
PAGE NUMBERING	1,17
PANELS, CONTROL (SEE INDIVIDUAL DEVICES)	
PAPER TAPE	
ASSEMBLE FROM	9.9
DRIVER	8.31-8.33
EIGHT CHANNEL DATA FORMAT	5.40
HIGH-SPEED READER	5.43-5.48
HIGH-SPEED PUNCH	5.43-5.48
PREPARE, CONVERSATIONAL FORTRAN	9.27
SYM-I FORMAT	9.23
TAPES AND CARD DECKS	9.1
PARITY	
MAGNETIC TAPE, DIO	5.114
MEMORY CHECK	6.1
PART NUMBERS	9.1/AP M=1
PASS, ONE AND TWO, ASSEMBLER	8.6
PAUSE MESSAGE	
MTOS	9.51
RTOS	9.57
STANDARD	9.45
PEAT (SEE PERIPHERAL EQUIPMENT ASSIGNMENT TABLE)	
PEN, CONTROL CODES, PLOTTER	5.126
PERIPHERAL	
CABLING OF DEVICES	4.8
DATA BIT ASSIGNMENT	5.8
EQUIPMENT (SEE INDIVIDUAL DEVICES) ASSIGNMENT TABLE (PEAT)	5.26
OPERATION	5.1-5.161
STATUS BIT ASSIGNMENT	5.9
PLOTTER, DIGITAL	5.123-5.134
CHARACTERISTICS	5.123
CONTROL PANEL	5.131
DRIVER	8.47
FUNCTIONS	5.125
INTERFACE ROUTINE	8.49
OPERATION	5.123
SPEED	5.123
STATUS CODES	5.126
TIMING	5.126
POOL, MATH STORAGE	8.75
POWER ON AND OFF PROCEDURES	7.1
POWER FAIL-SAFE	6.2
POWERS OF 2 AND 16	AP C
PREP	
FORTRAN	1.34/9.27
SYM-I	9.9
PRINTER, LINE (SEE LINE PRINTER)	
PRIORITY INTERRUPT SYSTEM	1,28/3.1
PROCEDURE (CONDITIONAL PROCESSING 8.8-8.10)	
PROCESSING, CONDITIONAL	8.10
PROCESSOR	
CPU, 706	1.7
MPS BATCH	8.73
MTOS	8.66
RTOS	8.69
STANDARD	8.64
PROGRAM	
ABSOLUTE, FORMAT	8.20/8.21
COUNTER	1.6
DEVELOPMENT CONTROL	8.59
DIRECT LOADING	7.4
EDITOR, SYMBOLIC	1.34/8.78/9.
EXECUTION	7.6

GENERATION 1,33
 MANUAL ENTRY 7.4
 MODIFICATION 9.24
 PREPARATION 1,33/8,78/9,21
 SORT/MERGE 8.80

 PROGRAMMING
 DIO 5,13
 DMA 5,23
 MPS CONCURRENT 8,71
 PERIPHERAL EQUIPMENT 5,1-5,161
 SYSTEMS DESCRIPTION 8,1-8,92
 SYSTEMS OPERATION 9,1-9,63

 PROTECT, MEMORY 6.2

 PROTECTION
 DIO PROGRAMMING 5,13
 FILE 5,88/5,103/5,116
 MPS SYSTEM 8,72
 TRACK, DISC 5,75

 PSEUDO INSTRUCTIONS 2,26/8,8/AP R

 PUBLICATION INDEX AP B

 PUNCH
 CARD 5,49-5,54
 PAPER TAPE
 HIGH-SPEED 5,43-5,48
 TELETYPE 5,39-5,42

 QUEUE/LOADER PROCESSOR 8,66/8,69

 RANDOM MODE ADDRESSING, A=D 5,143

 RANKING, INTERRUPTS 3,3

 RATE ERRORS
 DISC 5,75
 MAGNETIC TAPE 5,88/5,103

 READ (SEE INDIVIDUAL DEVICES
 AND DRIVERS)

 READER
 CARD 5,49-5,60
 PAPER TAPE
 HIGH-SPEED 5,43-5,48
 TELETYPE 5,39-5,42

 REAL OPERATIONS, FORTRAN IV LIBRARY AP J

 REAL-TIME FORTRAN IV 1,35/8,13
 CHARACTERISTICS AP K=4
 COMPILE-TIME ERROR MESSAGES AP N=3
 LIBRARY 1,37/ AP J
 RUN-TIME ERROR MESSAGES AP N=4
 USAS1 COMPARISON AP K=3

REAL-TIME OPERATING SYSTEM, 1,33/8,66/9,55
 ABSOLUTE LOAD ROUTINE 9,56
 BOOTSTRAP LOADING SYSTEM 9,55
 DISC-BASED LIBRARY 8,69
 ERROR MESSAGES AP N=9
 EXIT CURRENT TASK 9,56
 MESSAGES 9,56
 MANUAL INTERVENTION 9,56
 MPS COMPARISON 8,73
 OPERATOR INTERRUPT DIRECTIVES 9,55
 RELOCATABLE PROGRAM RESTART 9,56
 RESIDENT MONITOR 8,68
 SOFTWARE SUMMARY 9,54
 SYSTEM RECOVERY 9,56
 XRAY EXECUTIVE 8,70

 RECORD BLOCKING, LOADER TEXT 8,24

 RECORD FORMATS (SEE INDIVIDUAL
 DEVICES AND DRIVERS)

 RECOVERY
 ERROR, MAGNETIC TAPE 8,40/8,46
 SYSTEM
 MTOS 9,50
 RTOS 9,56
 STANDARD 9,44

 REFERENCE DATA AP D

 REGISTER
 ACCUMULATOR 1,6
 DESCRIPTION AND OPERATION 1,6
 EXTENSION 1,9/1,17
 INDEX 1,6/1,22
 INSTRUCTION 1,6
 MEMORY BUFFER 1,6
 PROGRAM COUNTER 1,6/7,1
 SLEW 5,64

 RELOCATABLE
 LIBRARY 8,82/8,84
 LOADER
 BASIC 8,24/9,7
 DISC 8,27
 ERROR MESSAGES AP N=7/AP N=8
 STANDARD 8,26

 RESIDENCY, USER DRIVER 8,53

 RESIDENT MONITOR
 MTOS 8,65
 RTOS 8,68
 STANDARD 8,64

 RESIDENT PROGRAM DIRECTIVES AP M=2

 RESTART, RELOCATABLE PROGRAM
 MTOS 9,50
 RTOS 9,56
 STANDARD 9,44

REVICLES, PLOTTER	5.130
RETURN FROM SUBROUTINES	1.24
REVISION LETTERS,	9.1
REWIND MAGNETIC TAPE,	5.37/5.99/5.112
ROOTS, SQUARE	AP D
ROUTINES	
INPUT/OUTPUT SERVICE	5.18/8.31
INTERRUPT,	8.50
ROW-ACR BIT CORRESPONDENCE, CARDS	5.49
RUN-TIME	
CONVERSATIONAL FORTRAN	
ERROR MESSAGES,	AP N=6
EXECUTION	9.27
LIBRARY	AP J
SYSTEM,	8.16
FORTRAN IV, REAL-TIME	
ERROR MESSAGES,	AP N=4
LIBRARY	AP J
SYSTEM,	8.16
SEARCH FOR END-OF-FILE (SEE END-OF-FILE)	
SECTORS, DISC	5.73
SEEK END-OF-FILE (SEE END-OF-FILE)	
SEGMENTATION, PROGRAM, FORTRAN,	8.14
SELECTION SUMMARY, I/O,	AP L
SENSOR DIAGNOSTICS,	1.37/8.84-8.92
SET-UP	
ENTRY TO USER STATE,	6.3
LINKAGE ADDRESS,	5.17
SHIFT INSTRUCTIONS,	2.8
SHIPPING MANIFEST, SOFTWARE	9.1
SIMULTANEOUS UNIT OPERATION,	
MAGNETIC TAPE	5.116
SINGLE COMMAND AND STEP OPERATION, CPU	7.7
SKIP RECORDS,	5.37/5.84/5.98
SET MEMORY BANK	1.19/2.25
SLEW, LINE PRINTER,	5.63
SMB (SEE SET MEMORY BANK)	

SOFTWARE	
BLOCK DIAGRAM,	1.3
DATA WORD FORMATS,	1.14
INPUT/OUTPUT (IOS)	5.25
OPERATION SUMMARY	
BASIC	9.5-9.42
MPS	9.61-9.63
MTOS,	9.49-9.54
RTOS,	9.55-9.60
STANDARD,	9.43-9.48
SYSTEM CONFIGURATION	8.1
ADVANCED,	1.30
BASIC	1.30
EXTENDED,	1.30
STANDARD,	1.30
SORT/MERGE PROGRAM,	8.80
SOURCE STATEMENT FORMATS, ASSEMBLER	1.11
SPECIAL IOS CALLS	5.33
SPECIFICATIONS	
706,	1.4
PERIPHERAL DEVICES (SEE INDIVIDUAL DEVICES)	
SPOOLER, TAPE	5.44
SQUARE ROOTS,	AP D
STACKER, CARD	5.49/5.50
STANDARD OPERATING SYSTEM	1.30/8.63-8.65
STANDARDS, USASI FORTRAN,	AP K=3
STATE, USER	6.3-6.5
STATEMENT FORMATS, ASSEMBLER,	1.11
STATUS	
PERIPHERAL (SEE INDIVIDUAL DEVICES)	
PROGRAMMING,	9.13
WORD FORMATS	1.15/5.9/AP L
STORAGE	
MASS	8.4
MATH LIBRARY	8.75/ AP J
STRINGS, EXTERNAL	8.24
SUBPROGRAMS, FORTRAN IV	AP J
SUBROUTINES	8.11
LINKAGE,	1.22
MATH	8.75
RETURN FROM,	1.24

SYM-I/PREP. 1,34/8,12/9,9-9,20
 CHARACTERISTICS. AP K=1
 ERROR MESSAGES AP N=1
 OPERATION. 0,9
 PAPER TAPE FORMAT. 9,23
 SAMPLE PROGRAM 5,35/9,18-9,20

 SYM-II. 1,34/8,12
 CHARACTERISTICS. AP K=2
 ERROR MESSAGES AP N=2
 SAMPLE PROGRAM 5,36

 SYMBOL AND DATA DEFINITION. 2,26/8,77

 SYMBOLS, EQUATION TERMS 2,2

 SYMBOLIC DEBUGGING. 1,34

 SYMBOLIC PROGRAM EDITOR 1,34/8,78/9,21

 SYNTAX, ASSEMBLER WORD FORMAT 1,16

 SYSGEN1,SYSGEN2 8,82
 ERROR MESSAGES AP N=10/AP N=11

 SYSTEM(S)
 AUTO, PRIORITY INTERRUPT 1,28/3,1-3,3
 DIRECTIVES, XRAY AP M=2
 GENERAL 706 DESCRIPTION. 1,1-1,37
 GENERATION 8,81
 HARDWARE 8,1
 NUMBER AP E
 OPERATING
 BASIC 1,30/8,62/9,7
 MULTIPROGRAMMING. 1,33/8,71/9,61
 MAGNETIC TAPE 1,33/8,65/9,49
 REAL-TIME 1,33/8,66/9,55
 STANDARD. 1,33/8,63/9,43
 PAPER TAPE INPUT/OUTPUT (PTIOS). 8,62

 TABLE
 CONVERSION AP E
 ENTRY NAMES. 9,7
 FILE INPUT/OUTPUT (FIOT) 5,31
 IOS FUNCTION 5,27
 PERIPHERAL EQUIPMENT ASSIGNMENT. 5,30
 SYSTEM LINKAGE 5,34

 TAPE(SEE MAGNETIC, HIGH-SPEED PAPER)

 TASK, EXIT CURRENT
 MTOS 9,50
 RTOS 9,56
 STANDARD 9,44

 TELETYPE (SEE ASR-33 AND MULTIPLEXER)

 TEST, DIO STATUS METHOD 5,13

 TEXT, LOADER
 CODES. 8,20-8,25
 RECORD BLOCKING. 8,24

TIME-OF-DAY CLOCK 5,157-5,162

 TIMING (SEE INDIVIDUAL DEVICES)
 CENTRAL PROCESSOR. 1,9
 CONVERSION, COMPUTER-TO-COMPUTER AP I
 INTERFACE
 DIO 4,5
 DMA 4,10
 MATH LIBRARY AP J

 TITLE LINE, ASSEMBLER 1,11

 TRACE/DEBUG PROGRAM 8,80

 TRACK, DISC 5,73

 TWO PASS ASSEMBLER. 8,6

 TWO'S COMPLEMENT ARITHMETIC AP F

 UNLOCKED, A=D. 5,143

 UNIT
 CENTRAL PROCESSING 1,6
 LOGICAL, NUMBER. 5,26
 TAPE, ASSIGNMENT NUMBER 5,79/5,96/5,109

 USASI, FORTRAN STANDARDS. AP K=3

 USER
 CREATED DRIVERS. 8,50-8,58
 STATE. 6,3-6,5

 UTILITY PROGRAMS. 8,75-8,92
 DIAGNOSTICS, SENSOR. 8,84
 EDITORS. 8,78/AP K
 MATH LIBRARY 8,75/AP J
 SORT/MERGE 8,80
 SYSTEM GENERATION. 8,81
 TRACE/DEBUG. 8,80/AP K

 VECTOR COMMAND CODES, PLOTTER 5,127

 VERIFY, DISC. 5,75

 VERTICAL FORMAT TAPE, LINE PRINTER. 5,68

 WORD
 FORMATS. 1,10/8,77
 REFERENCE INSTRUCTIONS 2,1

 WRITE (SEE INDIVIDUAL DEVICES
 AND DRIVERS)

 XRAY EXECUTIVE
 DIRECTIVES AP M=2
 SYSTEMS
 BASIC 1,33/8,62/9,7
 MTOS. 1,33/8,65/9,49
 RTOS. 1,33/8,66/9,55
 STANDARD. 1,33/8,63/9,43

**WORLD-WIDE SALES AND SERVICE
UNITED STATES**

HOME OFFICE - Raytheon Computer
2700 South Fairview Street, Santa Ana, California 92704
Phone: (714) 546-7160 From Los Angeles phone: 625-7645
TWX: 910 595-1570

CALIFORNIA

Western District
Raytheon Computer
9550 Flair Drive, Suite 207
Flair Industrial Park
El Monte, California 91731
Phone: (213) 443-7181

Raytheon Computer
363 S. Taaffe, Suite 201
Sunnyvale, California 94086
Phone: (408) 736-2286
TWX: 910-339-9205

GEORGIA

Southern District
Raytheon Computer
1600 Tullie Circle, N.E.
Atlanta, Georgia 30329
Phone: (404) 631-0091
TWX: 810-751-8447

ILLINOIS

Central District
Raytheon Computer
710 Higgins Road
Park Ridge, Illinois 60068
Phone: (312) 692-3325
TWX: 910-687-2266

MASSACHUSETTS

Northeastern District
Raytheon Computer
Box 184 - Fourth Avenue
Burlington, Massachusetts 01803
Phone: (617) 272-6400
TWX: 710-332-0313

TEXAS

West Central District
Raytheon Computer
6420 Hillcroft, Suite 215
Houston, Texas 77036
Phone: (713) 771-4449; - 5851
TWX: 910-881-1795

WASHINGTON, D.C.

Mid-Atlantic District
Raytheon Computer
1316 Fenwick Lane, Suite A
Silver Spring, Maryland 20910
Phone: (301) 587-9110
TWX: 710-825-9762

INTERNATIONAL

ARGENTINA

Rayo Electronica
Belgrano 990
Buenos Aires, Argentina
Phone: 38-1779; 37-9476
Cable: RAYOTRONICA

AUSTRALIA

Datamatic Pty, Ltd.
90 Alexander Street
P.O. Box 171
Crow's Nest, N.S.W., 2065, Australia
Phone: 43-3333; - 7125

BELGIUM

Koopman & Co. Electronica N.V.
13 Avenue des Gaulois
Brussels, Belgium
Phone: 02-35-80-62
Telex: 846-23616

DENMARK

Aage Hempel Teknik A/S
Longangstraede 16
DK 1468 Copenhagen K, Denmark
Phone: 01-156801
Telex No.: 855/5231

ENGLAND

Raytheon Overseas, Ltd.
Shelley House-Noble Street
London, E.C. 2, England
Phone: 01-606-8991
Telex: 851-25251

FRANCE

Gamsi
3 Rue du Faubourg Saint-Honore
Paris 8, France
Telex: 842-22489 F

WEST GERMANY

Dynatec (Datenverarbeitung-Elektrotechnik
Vertriebs GmbH)
4 Dusseldorf, West Germany Hansa Allee
159
Phone: Dusseldorf 5910-71
Telex: 858-4389

HOLLAND

Koopman & Co. Electronica N.V.
Post Box 6049 (Stadhouderskade 6)
Amsterdam, Holland
Phone: 020-182821
Telex: 844-11273
Cable: KJOEBMAND

INDIA

Greaves & Cotton & Co., Ltd.
2 Palace Road, P.B. No. 13
Bangalore 1, India
Phone: 28373/26773
Telex: 043-365
Cable: GREAVES

ISRAEL

Agentex, Ltd.
3 Bograshov Street T.A. P.O.B. 3308
Tel-Aviv, Israel
Phone: 233968
Cable: AGENTEX TELAVIV

ITALY

Elettronucleonica
Pia de Angeli 7
20146 Milano, Italy
Phone: 463 520/286

JAPAN

Japan Radio Co., Ltd.
25, Sakuragawa Cho, Shiba
Mori Building Fifth
Nishikubo, Minato-ku
Tokyo, Japan
Phone: 591-3451
Telex: 781/1222-3068

KOREA

Korea Computers Corp.
Kurodong Industrial Estates
I.P.O. Box 3320
Seoul, Korea
Phone: 69-5187-7
Cable: KORAND

LEBANON

Triad International Marketing Co.
P.O. Box 4984
Beirut, Lebanon
Phone: 230091, 230093
Telex: TRICORP
Cable: TRIMARK

NORWAY

Metric A.S.
P.O. Box 80 Bekkelagshgd
Oslo, Norway
Phone: (02) 28-26-24
Telex: 851-18461

SOUTH AFRICA

Aerosignals Pty. Ltd.
P.O. Box 957
Johannesburg, South Africa
Phone: 836-6828/9
Telex: 43-0209JH
Cable: AEROSIGNALS

SPAIN & PORTUGAL

Neotecnica, S.A.E.
Marques de Urquijo 44
Madrid 8, Spain
Phone: 248-96-02
Cable: NEOTECNICA

SWEDEN

Scandia Metric AB
Dalvagen 12
Fack 171 03
Solna 3, Sweden
Phone: 08/82 04 10
Telex: 854/10766 METRICADE,
STOCKHOLM

SWITZERLAND

Computer AG Zurich
Badenerstrasse 551
8048 Zurich Postfach 145
Switzerland
Phone: (051) 277-325
Telex: 845/52 039 PULP CH

YUGOSLAVIA

Jugo Auto
Borisa Kidrica 6-1
Beograd, Yugoslavia
Phone: 336-503 or 342-394
Telex: 862/11564

From other countries contact:

Raytheon International
141 Spring Street
Lexington, Massachusetts 02173
Phone: (617) 862-6600
TWX: 710-324-6568

