Burroughs

B5500

Information Processing Systems

SYSTEM SOFTWARE OPERATIONAL GUIDE

Burroughs

B 5500

INFORMATION PROCESSING SYSTEMS

SYSTEM SOFTWARE

OPERATIONAL GUIDE

Burroughs Corporation
Detroit, Michigan 48232

$2.00

Printed in U.S. America 9-69 1043825

Correspondence regarding this document should be forwarded using the Remarks Form at the back of the manual, or may be addressed directly to Systems Documentation, Sales Technical Services, Burroughs Corporation, 6071 Second Avenue, Detroit, Michigan 48232.

TABLE OF CONTENTS

TABLE OF CONTENTS (cont)

INTRODUCTION

This document is intended to provide the user with information pertaining to the initialization and maintenance of the Burroughs B 5500 Software Package. Section 1, Loading and Maintaining the System, describes the procedures necessary to initialize the system. Section 2, MARK IX Programing System for the B 5500, contains information pertaining to the B 5500 Software. Examples are given illustrating the control cards necessary to compile a majority of the programs on the MARK IX system tape. Since it is intended primarily for systems personnel, the reader is expected to be familiar with the B 5500 Operation Manual.

SECTION 1
LOADING AND MAINTAINING THE SYSTEM

GENERAL.

This section describes the procedures for loading and maintaining the B 5500 System.

SYSTEM PROGRAMS.

When a user obtains a B 5500 System, he is supplied with the following items which are described in more detail in subsequent paragraphs.

a. System tape.

b. MCP Loader.

c. Cold Start deck.

d. Disk Halt/Load Button card.

e. LOAD control cards for the system programs.

f. SYMBOL tape - this is a multi-file tape containing the source programs for the system software.

g. BASIC change decks for all programs on the SYMBOL tape.

SYSTEM TAPE.

This system tape is a library tape, created through the use of a DUMP control card which has the library tape name SYSTEM and contains the B 5500 compilers, the system programs, and any other library programs which may be desired by the user. Programs which are at the present time considered system programs are the following:

Symbol File Name	Object File Name	Program Function
SYMBOL/DCESPSY	MCP/DISK	Master Control Program.
SYMBOL/ALGOLSY	ALGOL/DISK	ALGOL Compiler programing system.
SYMBOL/COBOLSY	COBOL/DISK	COBOL Compiler programing system.

Symbol File Name	Object File Name	Program Function
SYMBOL/EXPOLSY	ESPOL/DISK	ESPOL programing system.
SYMBOL/FORTSY	FORTRAN/DISK	FORTRAN Compiler programing system.
SYMBOL/INTRNSY	INTRNSC/DISK	MCP Subroutine library.
SYMBOL/COLDSY	*	Routine to generate Cold Start deck.
SYMBOL/DILDSY	*	Routine to generate Disk to Disk Loader deck.
SYMBOL/TALDSY	*	Routine to generate MCP tape Loader deck.
SYMBOL/MSTSTSY	MASTER/TEST	System Maintenance Routine.
SYMBOL/TEXTSY	TEXT/EDITOR	Routine to create and update disk files remotely.
SYMBOL/MKASTSY	MAKCAST/DISK	Routine to create symbolic library files.
SYMBOL/XLATFSY	FORTRAN/TRANS	FORTRAN Translator programing system.
SYMBOL/INTRPSY	INTERP/DISK	Routine permitting a remote console to be used for rapid computational ability.
SYMBOL/LOGSY	LOGOUT/DISK	Routine to print system log.
SYMBOL/UPDATSY	UPDATE/USERS	Routine to create and maintain the REMOTE/USERS file.
SYMBOL/DCLOGSY	LOGOUTR/DISK	Routine to print system remote log.

*No object file name applicable.

Because of the importance of the system tape, each installation should make a backup copy of it. This can be done by means of a DUMP control card. This card can also be used to add or delete items on the tape when making a new tape.

MCP LOADER.

The MCP Loader is a CARD LOAD SELECT program (i.e., a program loaded with the CARD LOAD SELECT button on) which searches for the magnetic tape unit which has the system tape containing the file MCP/DISK. The program then loads the MCP from the tape to its reserved area at the beginning of the system disk. The MCP Loader consists of a deck of punched cards of a fixed size and requires no parameter cards.

COLD START DECK.

The Cold Start deck is a CARD LOAD SELECT program (i.e., program loaded with the CARD LOAD SELECT button on), which has the primary function of creating an initial disk directory and setting initial operating conditions. This routine must be run before the MCP is used for the first time since the MCP expects to find a disk directory on the disk. It is generally considered a one-time routine. The Cold Start routine is explained in greater detail on page 1-8.

IMPROVED COLD START PROGRAM.

This program incorporates the following features:

a. Disk file headers are now created in the format described in Change MCP VIII.41.

b. The BACKUP card is no longer accepted.

c. A FENCE card may be included. This card initializes DIRECTORYTOP [19] which is used by the Time Sharing MCP. The format of the FENCE card is:

FENCE = ⟨integer⟩

Any ⟨integer⟩ less than 8192 will be changed to 8192. Any ⟨integer⟩ greater than 28582 will be changed to 28582. Any ⟨integer⟩ will be changed to the nearest number that is a multiple of 1024.

NOTE

Although this COLD START program is being released on the MARK IX.0 System, it must not be used on any standard system below MARK IX level 8 MCP. You will be notified at the time of release of level 8, to begin use of this COLD START program.

COOL START PROGRAM.

This program is essentially the same as the improved COLD START program described on the previous page, with the exception that, instead of rebuilding the directory, the COOL START program searches the directory and removes each file that has the following characteristics:

a. The first character of the ⟨multiple-file-identification⟩ or ⟨file-identification⟩ is non-zero.

b. Any word in the file header that has the flag bit on.

c. The maximum number of areas declared is greater than 20.

d. The disk address of an area is less than DISKTOP or greater than that possible, based on the number of disk electronics units declared.

e. A disk address is present for an area that is outside the maximum number of areas declared.

NOTE

Although this COOL START program is being released on the MARK IX.0 System, it must not be used on any standard system below MARK IX, Level 8 MCP. You will be notified at the time of release of level 8, to begin use of this COOL START Program.

DISK HALT/LOAD BUTTON.

The Disk Halt/Load Button is a one-card CARD LOAD SELECT program which causes the MCP to be initiated by reading a portion of the MCP code from disk into core memory and then branching to that code.

LOAD CONTROL CARDS.

The LOAD cards which cause programs to be read from the system tape and entered as library files on the user disk are identical in constructs to the LOAD cards described in section 4 of the B 5500 Operation Manual. For example, a LOAD card to load ALGOL onto disk could appear as follows:

```
? LOAD FROM SYSTEM ALGOL/DISK
```

It should be noted that with the exception of the MCP, system programs are executed from the user disk. (The user disk is that portion of the disk available for user files. It includes all disks except the system disk which is an area on the first disk module which is reserved for the MCP, special MCP tables, and overlay storage.) The MCP is executed from the system disk.

SYMBOL TAPE.

The SYMBOL tape contains source programs for system programs. The source programs for ALGOL/DISK, COBOL/DISK, FORTRAN/DISK, ESPOL/DISK, MAKCAS1/DISK, FORTRAN/TRANS, and LOGOUT/DISK are in Extended ALGOL. The source program for MCP/DISK is in ESPOL. The routines PRNPBT/DISK and LDCNTRL/DISK were coded in part by hand and do not exist in a source language. These routines are, in fact, partly contained in their respective program files and partly within communicate routines in the MCP. Consequently, they are not on the SYMBOL tape.

The file identifications for the files on the SYMBOL tape are as follows:

File Identification	Program Function
DCESPSY	MCP.
INTRNSY	MCP subroutine library augmentation.
ALGOLSY	ALGOL programing system.
COBOLSY	COBOL programing system.
FORTRAN	FORTRAN programing system.
ESPOLSY	ESPOL programing system.
LOGSY	Routine to print system log.
DCLOGSY	Routine to print system remote log.
MKASTSY	Routine to create symbolic library files.

BASIC CHANGE DECKS.

The basic change decks for the programs on the SYMBOL tape are, with the exception of the deck for the MCP, ALGOL patch decks. That is, they each contain a COMPILE control card for ALGOL, label equation cards, a DATA or LABEL control card, $ TAPE cards, patch cards if any, a 9's card, and an END control card. The basic change deck for the MCP contains an EXECUTE control card to call out ESPOL, label equation cards, a DATA or LABEL control card and, as it is with ALGOL, $ TAPE cards, patch cards if any, a 9's card, and an END control card.

UPDATING SYSTEM PROGRAMS.

If changes are made in system programs, users are supplied with symbolic patch cards which reflect these changes. It is then the responsibility of the user to:

a. Place the patch cards, according to the sequence numbers in columns 73 through 80, in the symbolic patch deck for which they were intended.

b. Recompile the subject program in order to obtain the updated version.

Programs compiled on the B 5500 System are compiled onto disk. If the user wishes to obtain an updated version of the system tape, he must create that tape through the use of a DUMP control card.

It should be noted that when compiling system programs onto user disk, the system program being compiled is generally given a different name than that by which it must be called when used. This is done to avoid a duplicate library condition. However, once the new version of the system program has been successfully compiled onto disk, the old version of the program can be removed. Then the name of the new version can be changed through the use of a CHANGE control card.

There are two MCP Loader programs contained in symbolic form on the multi-file SYMBOL Tape. They have the file identifications of TALDSY and DILDSY. These are the Tape MCP Loader and the Disk to Disk MCP Loader respectively. These are ESPOL programs for which basic change decks are furnished which will allow the production of the load decks themselves.

When the load decks have been punched, the front and back label cards should be discarded, and an ESPOL loader card placed on the front of each deck.

A HALT/LOAD card should be placed on the back of the Tape loader. This deck is now ready for use in conjunction with the COLD START deck.

The Disk to Disk loader assumes that the disk directory already exists; i.e., it is not a replacement for the tape-to-disk loader. It is useful, however, for loading experimental and/or new MCP's after compilation.

The program must be loaded with an ESPOL disk loader. Immediately following the program is an input card containing the file name of the MCP file to be loaded. After reading this card, the program searches the disk directory, copies the file into the MCP resident area on disk, and simulates a HALT/LOAD operation.

The input card contains the file id. prefix in columns 2-8 and the file id. suffix in columns 10-16. If the specified file is not in the directory, a message will be printed and another card will be read. If the specified file is in the directory, the SPO will skip one line and the file will be loaded. Then, the message "⟨file specified⟩ LOADED" will be typed and a HALT/LOAD sequence initiated.

SYSTEM LOADER.

The two prerequisites to operation on a B 5500 System are:

a. The MCP must be on disk.
b. The disk directory must be on disk.

The System Loader is used to establish the above two conditions. In addition, the System Loader provides for the following:

a. A means for initializing the current date word and option codes.

b. A means for specifying the disk file configuration and the amount of disk to be used for data overlay storage.

c. Loading of the B 5500 compilers.

The System Loader consists of the following:

a. MCP Loader (refer to page 1-3).
b. Cold Start routine.
c. DISK HALT/LOAD BUTTON card.
d. Control cards used to load other software onto disk.

The above are discussed in detail in the following paragraphs.

COLD START ROUTINE.

A COLD START symbolic program is contained on the multi-file SYMBOL tape with the file identification of COLDSY. When the COLD START deck is punched, remove the front and back label cards and discard them. An ESPOL loader card must be placed on the front of the COLD START deck and a HALT/LOAD card must be the final card of this deck.

The Cold Start routine constructs the initial disk directory, and initializes the current date word and option codes.

The Cold Start routine deck is a deck of punched cards, the last of which must be a STOP card, i.e., a card containing the word STOP. Three or more parameter cards must immediately precede the STOP card.

The parameter cards for the Cold Start routine have a free-field format and are described in the following paragraphs.

DRCTRYTP CARD.
The DRCTRYTP card (Directory Top) provides an integer which specifies the absolute disk address of the segment known as DIRECTORYTOP. A DRCTRYTP card must appear in the COLD START deck. It must be the first parameter card in the deck.

The DRCTRYTP card must have the following information:

DRCTRYTP ⟨integer⟩

Example:

DRCTRYTP 999

DIRECT CARD.
The DIRECT card provides an integer which specifies the address of the highest addressed disk segment which should be used for the disk directory. A DIRECT card must appear in the Cold Start deck.

When determining the figure to be specified as the upper boundary, it should be realized that every 15 files on the disk use 16 segments in the directory.

The DIRECT card must have the following information:

DIRECT ⟨integer⟩

Example:

DIRECT 2500

ESU CARD.

The ESU (electronic storage unit) card supplies an integer which indicates the number of electronics units connected to the system. If there are no electronics units on the system which are unique to DKB (i.e., Disk File Control Unit 2) then this card should contain an integer which reflects the total number of electronics units on the system. (This is generally the case when an exchange is present.) If there are electronics units which are unique to DKB, then the integer on this card must have a value which expresses the sum of the number of electronics units on DKA (i.e., Disk File Control Unit 1) plus 100 times the number of electronics units on DKB. For example, if a B 5500 System had three electronics units on DKA and two on DKB, then the integer on the ESU card should be 203. An ESU card must appear in the Cold Start deck.

The ESU card must contain the following information:

ESU ⟨integer⟩

Examples:

ESU 1
ESU 202

DATE CARD.

The DATE card is optional, but should precede all FILE cards in the Cold Start deck, if any. A date supplied in a DT keyboard message, entered subsequently to the use of the DATE card, would supersede the information on the DATE card.

The DATE card provides three integers separated by the character /. The first integer specifies the two digits of the month, the second integer specifies the day, and the third specifies the last two digits of the year. This card causes the date-word on the disk to be set to the date specified. (This date-word contains the current date used, e.g., in tape labels.)

The DATE card must contain the following information:

DATE ⟨integer⟩ / ⟨integer⟩ / ⟨integer⟩

Example:

DATE 12/29/69

FILE CARD GROUP.

The function of a FILE card group is to define a user file which is to be listed in the disk directory. This method of defining a file, as opposed to defining a file through the use of a file declaration in a program, allows an installation to explicitly assign specific disk addresses for files.

A FILE card group consists of a FILE card and one or more file address cards. There may be as many FILE card groups as desired in the Cold Start deck.

The FILE card contains the word FILE, which identifies the FILE card group, and the card supplies the following information in the order listed and is separated by commas:

a. Data file specifier which provides the data file name to be listed in the disk directory.

b. ⟨Integer⟩ x ⟨integer⟩ construct, where the first integer specifies the number of areas on disk to be used by the file, and the second integer specifies the number of 30-word disk segments in each area.

c. Integer which specifies the purge factor for the file (i.e., the number of days past the date of last access that the file is to be retained on disk).

In total then, a FILE card must contain the following information:

FILE ⟨data file specifier⟩, ⟨integer⟩ x ⟨integer⟩, ⟨integer⟩

Example:

```
FILE PREFIX/NAME, 2x1000, 30
```

A file address card contains a single integer, which is different from zero, and specifies the absolute disk address of the first word in an area to be used by the file whose name was supplied by the preceding FILE card. There must be one file address card for each area specified for the file. The first file address card in a FILE card group provides the beginning address of the first area to be used by the file; the second file address card provides the beginning address of the second area to be used, and so on. A zero integer for an address denotes that the MCP is to assign the address for that area.

As noted above, a file address card must contain the following information:

⟨integer⟩

Example:

```
2000
0
```

Three examples of a FILE card group are as follows:

a.
```
FILE SYSTEM/LOG, 1x1000, 2
0
```

b.
```
FILE SAVES/AREA, 3x9000, 15
2500
3400
0
```

c.
```
FILE B 280/RESERVE, 2x1500, 365
4300
8000
```

It should be noted that if log information for the system is to be recorded, a file with the file identification prefix SYSTEM and the file identification LOG must be defined on the disk. Also, this file must be limited to one area on the disk. Consequently, if system log information is to be retained, a FILE card group (such as the first example above) should appear in the Cold Start deck.

OPTION CARDS.

If the option cards described in the following paragraphs are not used, the corresponding options are automatically set to off.

The USE DRA or OPTN 47 card specifies that a system is equipped with a drum memory unit designated DRUM A. Consequently, this option can be used only when the system is equipped. When this option is specified, the MCP will use DRUM A for overlay storage, and DRUM A will be used in preference to disk. When the USE DRA option is specified, a Loader which loads the MCP from disk to core at HALT-LOAD time is placed on the drum so that a HALT-LOAD from the drum can be used to initiate the operation of the system. However, <u>in this case only,</u> a HALT-LOAD card must physically be the last card in the Cold Start deck, therby making the STOP card the next-to-the-last card (see below). If the USE DRA option is not used, a CARD LOAD SELECT HALT-LOAD must be performed to initiate the system. This is done by using the DISK LOAD BUTTON card which is described later in this section.

The USE DRA card must contain either USE DRA or OPTN 47.

<u>Examples</u>:

```
USE DRA
OPTN 47
```

The USE DRB or OPTN 46 card specifies that a system is equipped with a drum memory unit designated DRUM B. Consequently, this option can only be used when the system is so equipped. When this option is

specified, the MCP will use DRUM B for data overlay storage and, when available, drum memory is used in preference to disk memory.

The USE DRB card must contain either USE DRB or OPTN 46.

Examples:

```
USE DRB
OPTN 46
```

The TYPE BOJ or OPTN 45 card specifies that a BOJ message is to be typed each time the MCP initiates a compiler or an object program.

The TYPE BOJ card must contain either TYPE BOJ or OPTN 45.

Examples:

```
TYPE BOJ
OPTN 45
```

The TYPE EOJ or OPTN 44 card specifies that an EOJ message is to be typed by the MCP when a compiler or an object program has come to a normal completion.

The TYPE EOJ card must contain either TYPE EOJ or OPTN 44.

Examples:

```
TYPE EOJ
OPTN 44
```

The TYPE OPEN or OPTN 43 card specifies that a message is to be typed by the MCP whenever an object program opens a file other than a disk file.

The TYPE OPEN card must contain either TYPE OPEN or OPTN 43.

Examples:

```
TYPE OPEN
OPTN 43
```

The USE TERMNATE or OPTN 42 card specifies that the TERMINATE procedure of the MCP is to be called if the MCP must discontinue processing of a program because of an error condition.

Since it is the function of the TERMINATE procedure to clear the system of all information pertaining to a discontinued program, the USE TERMNATE option generally should always be specified. However, if an error condition should occur where it is necessary to obtain a memory dump that reflects core conditions at error time, the USE TERMNATE option should not be used.

The USE TERMNATE card must contain either USE TERMNATE or OPTN 42.

Examples:

```
USE TERMNATE
OPTN 42
```

The TYPE DATE or OPTN 41 card specifies that DT PLEASE is to be typed by the MCP at HALT-LOAD time and that the system operator must enter a DT keyboard input message before processing can commence.

The TYPE DATE card must contain either TYPE DATE or OPTN 41.

Examples:

```
TYPE DATE
OPTN 41
```

The TYPE TIME or OPTN 40 card specifies that TR PLEASE is to be typed by the MCP at HALT-LOAD time and that the system operator must enter a TR keyboard input message before processing can commence.

The TYPE TIME card must contain either TYPE TIME or OPTN 40.

Examples:

```
TYPE TIME
OPTN 40
```

The USE ONEBREAK or OPTN 39 card specifies that all programs performing BREAKOUT will use only one BREAKOUT tape. If the option is not specified, each such program will be assigned its own BREAKOUT tape.

The USE ONEBREAK card must contain either USE ONEBREAK or OPTN 39.

Examples:

```
USE ONEBREAK
OPTN 39
```

The USE AUTOPRNT or OPTN 38 card specifies that printer backup tapes (not including those created previous to the latest HALT-LOAD) are to be automatically printed when a backup tape and a line printer are not in use at the same time. If the option is not specified, printer backup tapes will be printed only if the system operator enters a PB keyboard input message.

The USE AUTOPRNT card must contain either USE AUTOPRNT or OPTN 38.

Examples:

```
USE AUTOPRNT
OPTN 38
```

The USE CLEARWRS or OPTN 37 card specifies that the MCP will attempt to keep remote data communications stations (which have SPO capabilities) from remaining in a Write-Ready condition if object program output is not available when the condition occurs. (The Write-Ready condition occurs when an output message which does not not end in a group mark is sent to a data communications station. When a data communications station is Write-Ready, no input can be received.) If a Write-Ready condition occurs when no more output is queued for a station and if the station has SPO capabilities and if the USE CLEARWRS card is used, the MCP will send a group mark to the station, thus clearing the Write-Ready status.

NOTE

This option is applicable to Data Communications Systems only.

The USE CLEARWRS card must contain either USE CLEARWRS or OPTN 37.

Examples:

```
USE CLEARWRS
OPTN 37
```

The TYPE DISCONDC or OPTN 36 card specifies that the MCP will write a disconnect code on any data communications station which is not logged in when the station is disconnected from a program. This option should not be set if any data communications equipment on the system is connected to any telephone company equipment not wired to handle disconnect codes.

NOTE

This option is applicable to Data Communications Systems only.

The TYPE DISCONDC card must contain either TYPE DISCONDC or OPTN 36.

Examples:

```
TYPE DISCONDC
OPTN 36
```

The TYPE CMPLFILE or OPTN 35 card specifies that file-open and file-close messages are to be typed for compiler files according to the respective settings of the TYPE OPEN and TYPE CLOSE options. If this option is not specified, messages will not be typed because of the opening and/or closing of files used by compilers.

The TYPE CMPLFILE card must contain either TYPE CMPLFILE or OPTN 35.

Examples:

```
TYPE CMPLFILE
OPTN 35
```

The TYPE CLOSE or OPTN 34 card specifies that a message is to be typed by the MCP whenever an object program closes a file other than a disk file.

The TYPE CLOSE card must contain either TYPE CLOSE or OPTN 34.

Examples:

```
TYPE CLOSE
OPTN 34
```

The TYPE ERRORMSG or OPTN 33 card specifies that object time error messages are to be typed by the MCP if errors are encountered during the running of an object program and if programmatic recovery is used in the program.

The TYPE ERRORMSG card must contain either TYPE ERRORMSG or OPTN 33.

Examples:

```
TYPE ERRORMSG
OPTN 33
```

The TYPE RET or OPTN 32 card specifies that magnetic tape retention messages are to be typed by the MCP.

The TYPE RET card must contain either TYPE RET or OPTN 32.

Examples:

```
TYPE RET
OPTN 32
```

The TYPE LIBMSG or OPTN 31 card specifies that the following messages are to be typed by the MCP when appropriate:

```
⟨program ID⟩ LOADED
⟨program ID⟩ DUMPED
⟨program ID⟩ REMOVED
⟨program ID⟩ CHANGED TO ⟨program ID⟩
```

The TYPE LIBMSG card must contain either TYPE LIBMSG or OPTN 31.

Examples:

```
TYPE LIBMSG
OPTN 31
```

The TYPE SCHEDMSG or OPTN 30 card specifies that a message is to be typed by the MCP whenever a job is placed in the schedule.

The TYPE SCHEDMSG card must contain either TYPE SCHEDMSG or OPTN 30.

Examples:

```
TYPE SCHEDMSG
OPTN 30
```

The TYPE SECMSG or OPTN 29 card specifies that File Security maintenance messages are to be typed by the MCP.

NOTE

This option is applicable to Data Communications Systems only.

The TYPE SECMSG card must contain either TYPE SECMSG or OPTN 29.

Examples:

```
TYPE SECMSG
OPTN 29
```

The USE DSKTOG or OPTN 28 card specifies that any object program attempting input or output at any absolute disk address below the user disk area is to be discontinued by the MCP and an INVALID PRL message typed.

NOTE

This option is applicable to Data Communications Systems only.

The USE DSKTOG card must contain either USE DSKTOG or OPTN 28.

Examples:

```
USE DSKTOG
OPTN 28
```

The USE RELTOG or OPTN 27 card specifies that all object programs running under File Security which attempt to perform an ALGOL RELEASE statement referencing disk are to be automatically discontinued by the MCP.

NOTE

This option is applicable to Data Communications Systems only.

The USE RELTOG card must contain either USE RELTOG or OPTN 27.

Examples:

```
USE RELTOG
OPTN 27
```

The TYPE PBDREL or OPTN 26 card specifies that printer backup disk-release messages are to be typed by the MCP when option 38, USE AUTOPRNT, is not set.

The TYPE PBDREL card must contain either TYPE PBDREL or OPTN 26.

Examples:

```
TYPE PBDREL
OPTN 26
```

The USE CHECK or OPTN 25 card specifies that the MCP is to check all memory links for validity whenever allocation or deallocation of memory is performed. If an invalid link is found, the message INVALD LINK is printed on the SPO. A Halt/Load must then be performed. This option is available only when the MCP is compiled with:

```
$ DEBUGGING=TRUE
```

If this option is not set (reset) or if the MCP is compiled with DEBUGGING=FALSE, then the MCP will only check adjoining links when allocating or deallocating memory.

The USE CHECK card must contain either USE CHECK or OPTN 25.

Examples:

```
USE CHECK
OPTN 25
```

The STOP card must physically be the last card of the Cold Start deck and must contain STOP.

Example:

```
STOP
```

DISK LOAD BUTTON CARD.

The Disk Load Button card is a binary card containing code that causes the MCP to be inititated. This card must be used in conjunction with a "CARD LOAD SELECT" HALT-LOAD operation to initiate the MCP. The only exception to this would occur when the system is equipped with a drum memory unit and the DRA option is specified.

CONTROL CARDS USED TO LOAD COMPILERS ONTO DISK.

The control cards used in the System Loader to load compilers onto disk are:

a. A LOAD card specifying the library tape name SYSTEM and the names of the compilers to be loaded.

b. An END card.

The following example shows a possible choice of such control cards:

```
? LOAD FROM SYSTEM ALGOL/DISK, COBOL/DISK
? END
```

SYSTEM START UP PROCEDURE.

Before beginning any operation, the system must be turned on by pressing the POWER ON switch on the operator console.

LOADING THE SYSTEM FROM THE SYSTEM TAPE.

The entire software package (MCP, compilers, system programs, etc.) is contained on the system tape. Any or all of the above items can be loaded onto the disk by means of the System Loader. This loading can be accomplished all at one time, or at different times. Once the desired items have been placed on the disk, they remain there until a subsequent loading operation is called for.

To load the system with all items on the system tape at the same time, the operator must perform the following steps:

a. Press the CARD LOAD SELECT and HALT switches on the operator console. (These switches will be lit when properly actuated.)

b. Place the system tape on a tape unit.

c. Place the System Loader in the card reader.

d. Press RESET and START on the card reader.

e. Press LOAD on the operator console.

Since the MCP Loader and the Cold Start are independent routines, they may be loaded onto disk separately or together by placing the respective card decks in the card reader and performing a "CARD LOAD SELECT" HALT-LOAD operation. Also, the DISK LOAD BUTTON card can be used anytime the MCP and the disk directory are on the disk. The control cards used to load the compilers may be used any time the MCP is in operation. Therefore, the MCP LOADER, the Cold Start, the HALT LOAD BUTTON or the software load cards may be used separately. For example, if a compiler that was not initially loaded is required on the disk, a LOAD card alone can be used to place that compiler on the disk. If a modified MCP is to replace the MCP already on the disk, the MCP LOADER alone can be initiated with a "CARD LOAD SELECT" HALT-LOAD operation.

There are messages which will be typed out while loading the system. They are as follows:

Message	Description
MCP FILE LOADED	The MCP Loader has successfully loaded the MCP onto the disk.
DIRECTORY BUILT	Cold Start has constructed a disk directory.
-HL-	The MCP has assumed control of the system.

Message	Description
INCORRECT CARD	An erroneous card is read during the operation of the Cold Start. This situation can be remedied by placing the correct card in the card reader, pressing RESET, then START on the card reader.
⟨unit mnemonic⟩ ERROR	

PROGRAM SCHEDULING INFORMATION.

After the desired information has been loaded onto the disk, the MCP is ready to begin processing. The MCP is told what is to be processed by means of control information. This information, made available to the MCP via punched cards, is placed in an available card reader by the operator.

The MCP scans the available input/output units, reading a record from each input file. During this scanning operation, the MCP recognizes the card it reads as containing the needed control information. The card is then analyzed and the indicated operation is performed.

THE SELECTION ALGORITHM.

The selection algorithm in the MCP prevents too many programs from being executed at the same time; otherwise, the number of programs which are multiprocessed together would be limited only by MIXMAX. Presented with many jobs at the same time, the system would load all of the jobs into core memory with the following results:

a. The system could run out of core memory.

b. The system could become overlay-bound, with the total elapsed time for the multiprocessed mix greater than that required to run the jobs serially.

The selection algorithm runs only as many programs as will fit together in core. Estimates of core requirements for programs are constructed by the ALGOL, FORTRAN and COBOL Compilers. The MCP determines how much core memory is available for the processing of object programs. As each program is started, its core requirements are added to the total. A program will not be started if its core requirement, plus the sum of the core requirements for jobs already running, exceeds the total amount of core storage available for the processing of object programs.

The selection algorithm contains several variants which make it more useful. These are as follows:

a. Any program will be loaded immediately, regardless of its core requirement, if there are no other jobs actually running.

b. The core estimates created by the compilers are reasonably accurate. It should be understood, however, that it is logically impossible for a compiler to determine how much core space a program will need in order to run efficiently. (The compiler could easily determine a maximum core requirement, but program logic determines which program segments, arrays, and files should coexist in core storage for efficient operation.) A program-parameter card, called a CORE card, is provided to allow the programmer or operator to override the core requirement provided by the compiler.

c. Programs which have been presented to the MCP to be run, but which have not been started because of the lack of core space, are said to be in the "schedule".

d. The operator may choose to cause a program which is in the schedule to be loaded in spite of the fact that the MCP does not think the program will run efficiently with the jobs already in the mix.

e. The operator may choose to terminate a program which is still in the schedule.

THE MULTIPROCESSING FACTOR.

At Halt/Load time, the MCP determines the total amount of core storage available in the system, and subtracts from this the amount of core used for non-overlayable MCP program segments and tables. This number is then multiplied by the multiprocessing factor (nominally set to one) to determine how much core space should be considered to be available for scheduling purposes.

This multiprocessing factor can be changed by the operator so that the MCP will think it has more, or less, core storage to use for running object programs. On a system containing eight memory modules, for example, the amount of core available for object programs is approximately 28,800 words. If the operator chose to set the factor to 0.8, the MCP would consider that only 23,040 words were available to object programs. Thus the jobs which would run together would have core requirements which totaled less than 23,040 words, and any other jobs would be left in the schedule until one of the jobs in the mix came to End-of-Job.

If, on the other hand, the factor had been set to 1.2, the MCP would consider that 34,560 words are available to user programs, and would continue to introduce jobs into the mix until the sum of their core requirements approached 34,000 words.

In effect, the multiprocessing factor allows the operator to increase or decrease the MCP's tendency to multiprocess. By increasing the factor, the MCP will load more programs to be multiprocessed together; by decreasing the factor, the MCP will not load as many programs to be multiprocessed.

Experience has shown that a factor of 1 causes the MCP to make quite reasonable decisions as to which jobs should be multiprocessed. It is conceivable that the factor might be set as high as 1.5, or as low as 0.8, to effect better performance at some particular installa-

tion. Factors outside of this range are not suggested. In particular, a factor greater than 2 will almost certainly cause the MCP to load every job presented to it, causing the system to run out of memory if job requests are presented indiscriminately. A factor of 0 will cause only one job to be run at a time, with the features that ZIPped programs will not be loaded until the calling program reached End-of-Job.

Changing the factor will cause the MCP to remember the new value until it is changed. A Halt/Load will not reset the factor.

SECTION 2
MARK IX PROGRAMMING SYSTEM FOR THE B 5500

The MARK IX Programming System consists of two symbolic tapes, each being a multi-file reel. Their contents are:

Tape 1	SYMBOL/DCESPSY	MARK IX MCP Symbolic
	SYMBOL/ALGOLSY	ALGOL Compiler Symbolic
	SYMBOL/XALGLSY	XALGOL Compiler Symbolic
	SYMBOL/COBOLSY	COBOL Compiler Symbolic
Tape 2	SYMBOL/INTRNSY	MARK IX INTRINSICS Symbolic
	SYMBOL/DCLOGSY	REMOTE/LOG File Analyzer Symbolic
	SYMBOL/UPDATSY	UPDATE/USERS Symbolic
	SYMBOL/LOGSY	SYSTEM/LOG File Analyzer Symbolic
	SYMBOL/XLATFSY	FORTRAN Translator Symbolic
	SYMBOL/CBL6500	B 6500 COBOL Compiler Symbolic
	SYMBOL/ALGFLTR	XALGOL - B 6500 ALGOL Filter Symbolic
	SYMBOL/ESPOLSY	ESPOL Compiler Symbolic
	SYMBOL/TEXTSY	TEST/EDITOR Symbolic
	SYMBOL/BASICSY	BASIC Compiler Symbolic
	SYMBOL/COBFLTR	B 6500 Cobol Filter Symbolic
	SYMBOL/FORTSY	FORTRAN Compiler Symbolic
	SYMBOL/DUMPANL	Memory Dump Analyzer Symbolic
	SYMBOL/MAKCAST	MAKCAST Utility Symbolic
	SYMBOL/COLDSY	COLD START Program Symbolic
	SYMBOL/PMERGSY	PATCH/MERGE Symbolic
	SYMBOL/COOLSY	COOL START Program SYMBOLIC
	SYMBOL/DUMPTAP	ESPOL Dump Core to Tape Program Symbolic

With the release of this system, a new method of updating symbolic tapes is being introduced. The program PATCH/MERGE, included in the MARK IX system tapes, should be used to incorporate changes as patch decks are released. The patch decks and current symbolic tapes will be used as input to the PATCH/MERGE program to create new symbolic tapes. The use of PATCH/MERGE is discussed on page 2-2.

PATCH/MERGE PROGRAM.

This program is to be used to create a master patch deck from the individual patch decks released by Burroughs. This master patch deck is then used to compile the program for which the patches were released.

The first card of data to the program must have the characters $. in columns one and two. The rest of the card must contain the number of patches, the identification of the program which the patches are for, and any options desired of PATCH/MERGE. The format of this card is:

$. ⟨number of patches⟩ FOR ⟨program id⟩ ⟨comments⟩ ⟨options⟩

The option currently available is CONFLICTS which will cause PATCH/MERGE to bring out any conflicts among patches and indicate how they are resolved.

Individual patch decks must begin with a card which has $# in columns one and two and the patch identification in columns three through 72. The standard format for this card is:

$# PATCH NUMBER ⟨number⟩ FOR ⟨program id⟩ CONTAINS ⟨number⟩ CARDS

An error will be indicated if the patch decks are not in ascending order, if the program identification does not agree with that of the first data card, or if the number of cards present does not agree with the number of cards given on this card. All patches released by Burroughs will have the proper header card as the first card of the deck. Future patch releases will assume that this program is being used to compile the program being patched.

PATCH/MERGE creates and ZIPs a control deck containing the merged patches. The control cards used in this control deck must be specified by cards which have the characters $* in columns one and two. The remaining 70 columns can contain any control card information desired. This type of card must precede all PATCH/MERGE control cards but the $. card.

NOTE

There must be at least one PSEUDOREADER in use or the control deck will not be scheduled by the MCP.

Patches which are not to be merged (e.g., dollar cards) may be included as the first patch deck. The header card for this deck must have the characters $- in columns one and two. The remainder of the card may contain comments. The cards in this deck will be the first cards of master patch deck.

The work files used by PATCH/MERGE have twenty rows of 300 records each. The number of rows may be changed by specifying the desired value on a COMMON control card when executing PATCH/MERGE.

Besides merging patch decks, PATCH/MERGE resolves conflicts among patch decks. If two or more patch decks have identical sequence numbers, then PATCH/MERGE will discard all but the one from the deck with the highest patch number. (If a void card is to be discarded, special action is taken so that the voiding is accomplished.) When a void card is encountered, cards in the voided range are discarded from the patch deck in which the void was found, and in all decks of lower patch number. Void cards in the decks of lower patch number are also so handled, when they are encountered while discarding. If there are cards in decks with higher patch numbers than the void card which fall into the voiding range, the necessary void cards are generated so that the tape area is voided but the patch cards from those higher numbered decks are not.

NOTE

For the purposes of this program, void cards must have their $ in column one, and the voiding sequence must be eight digits. This is a good practice anyway.

The following is an example of a deck which would compile the MCP from the two patch decks given:

```
? EXECUTE PATCH/MERGE
? DATA CARD
$.2 patches for MCP
$*EXECUTE ESPOL/DISK .compile the MCP
$*FILE DISK-MCP/DISK
$*FILE TAPE=SYMBOL/DCESPSY
$*FILE LINE=LINE BACK UP DISK
$*FILE CARD=MCPDECK
$*DATA MCPDECK
$-dollar cards necessary for MCP compilation
$SET DATACOM=FALSE, DEBUGGING=FALSE,DUMP=FALSE,DFX=FALSE INQUIRY=
TRUE
$SET CHECKLINK=TRUE,BREAKOUT=TRUE,DISKLOG=FALSE
$TAPE LIST PRT
$#PATCH NUMBER 1 FOR MCP CONTAINS 40 CARDS
                .
                .
                .
      〈SORTED PATCH DECK〉
                .
                .
                .
$#PATCH NUMBER 2 FOR MCP CONTAINS 10 CARDS
                .
                .
                .
     〈SORTED PATCH DECK〉
                .
                .
                .
? END
```

Following the preceding instructions, and being sure that the patch decks are placed in order will insure a compilation process which replaces the individual patch processing procedure. Maintaining a merged master patch deck is no longer necessary, in fact it is in-

advisable. Pulling a patch now involves only taking that deck out and rerunning the program. The features provided by the program are:

a. Duplicate sequence numbers in different patch decks are resolved by retaining only that card from the most recent deck (the deck furthest from the front of the input). If any void cards would have been replaced in this manner, their sequence numbers are adjusted so that they will remain in the deck.

b. If a void card exists in one patch deck, and more recent patch decks have cards in the area voided, the program will create the necessary void cards to void out the entire area except for the patches from these more previous decks.

c. Control card information is included with the patches, and this information is used to make a control deck. At the end of its processing, PATCH/MERGE zips this control deck. Since the information is specified as part of the data deck, any compilation options may be specified.

NOTE

In its released version, the program now requires at least one patch in the change deck in order to operate correctly. Because of this temporary requirement, PATCH/MERGE should not be used to compile Level 0 software. Once the first changes are released to a software subsystem, PATCH/MERGE should be used. This problem will be corrected, and you will be notified of the correction.

SYSTEM INITIALIZATION SUMMARY

Listed below are five basic operations to be performed as a newly installed system, or if a HALT/LOAD will not remedy the problem.

a. COLD START. See pages 1-3 and 1-8.

b. Load MCP. One of two options:

1) COLD START: Use CARD LOAD SELECT program TALDY (MCP tape to disk loader).

2) NOT a COLD START: Use either TALDY or the CARD LOAD SELECT program DILDY (MCP disk to disk loader). The latter is usually used to reload the currently running MCP or to replace it with a new version. The last card of the DILDY deck contains the name of the file which should be loaded. Example:

```
2       10          Card Column
NEW     MCP
```

This will cause DILDY to find NEW/MCP in the disk directory and place it into the area on disk for the running MCP.

c. HALT/LOAD.

d. LOAD SYSTEM FILES. One of two options:

1) COLD START: Load from SYSTEM =/=.

2) NOT a COLD START: Load any new files or continue to step e.

e. If this is not a COLD START and a new intrinsic file has been created, then key in the following:

CI ⟨File Identification Prefix⟩ Separator ⟨File Identification⟩

This step can be done anytime the system is operating.

APPENDIX A
DECK FORMAT FOR MASTER CONTROL PROGRAM

GENERAL.

This appendix contains information pertaining to the deck necessary to compile the MCP and an explanation of its option cards.

```
? EXECUTE PATCH/MERGE
?DATA CARD
$.XX PATCHES FOR MCP.IX CONFLICTS
$*EXECUTE ESPOL/DISK .COMPILE STANDARD MK IX MCP
$*PROCESS = 600; IO = 600
$*FILE STUFF = OOOOOOO/MCP TAPE
$*FILE TAPE = SYMBOL/DCESPSY
$*FILE NEWTAPE = MCP/SOURCE TAPE
$*FILE LINE = MCP/LISTING BACK UP DISK
$*DATA CARD
$-CARDS NEEDED FOR MCP COMPILATION
$ SET BREAKOUT = FALSE
$ SET BREAKOUT = FALSE
$ SET DEBUGGING = FALSE
$ SET DEBUGGING = TRUE
$ SET DUMP = FALSE
$ SET DUMP = TRUE
$ SET DFX = FALSE
$ SET DFX = TRUE
$ SET INQUIRY = FALSE
$ SET INQUIRY = TRUE
$ SET DATACOM = FALSE
$ SET DATACOM = TRUE
$ SET DCSPO = FALSE
$ SET DCSPO = TRUE
$ SET DCLOG = FALSE
$ SET DCLOG = TRUE
$ SET CHECKLINK = FALSE
$ SET CHECKLINK = TRUE
$ SET DISKLOG = FALSE
$ SET DISKLOG = TRUE
$ TAPE LIST PRT STUFF NEW TAPE
```

NOTES

See following pages for explanation.

Since the last card for each option determines its setting they may be changed by changing their position.

```
                    .
                    .
                    .
$#PATCH NUMBER 1 FOR MCP.IX CONTAINS N CARDS
            ⟨N SORTED PATCHES⟩
$#PATCH NUMBER 2 FOR MCP.IX CONTAINS N CARDS
            ⟨N SORTED PATCHES⟩
$#PATCH NUMBER XX FOR MCP.IX CONTAINS N CARDS
            ⟨N SORTED PATCHES⟩
? END
```

NOTE

When the MCP is compiled it is necessary to execute the program FILL/PRT which builds the disk file MCP/PRT used by DUMP/ANALYZE. If the MCP is compiled in the manner shown then FILL/PRT should be executed as follows:

```
? EXECUTE FILL/PRT
? FILE MCP = OOOOOOO/MCP TAPE
? FILE INT = OOOOOOO/INT TAPE
? END
```

MCP MODULARITY.

The current MCP is modular in design, and it incorporates all functions available in previous standard MCP's.

Modularity is achieved by the inclusion of $INCLUDE and $OMIT cards. The parameters defined on these cards must be given values of TRUE or FALSE by using $SET cards in the change deck at compile time. The compiled MCP is a function of the values given the parameters.

Parameters for which $SET cards must be included when compiling DCESPSY are:

PARAMETER	DESCRIPTION
BREAKOUT	If BREAKOUT is set TRUE, then code for the break-out-restart facility is included in the MCP. If set FALSE, the breakout-restart facility is not available.
DEBUGGING	If Debugging is set TRUE, then all debugging facilities of the MCP are available including memory dump, console access to core memory, access to disk via the console, and trace. If DEBUGGING is set FALSE, then debugging facilities are unavailable with the exception noted below by the DUMP parameter.
DUMP	If DUMP is set TRUE, then the memory dump facility is provided via the console. If DUMP is set FALSE, this facility is not provided.

NOTE

DEBUGGING set TRUE overrides the setting of the DUMP option since a system with full debugging facilities has memory dump capabilities via the console.

DFX	DFX set TRUE indicates two DISK FILE Control Units are on the system. Code is compiled into the MCP optimizing DISK I/O for this system configuration.
INQUIRY	If INQUIRY is set TRUE, code is included in the MCP for the data-communication INQUIRY system. If INQUIRY is set FALSE this code is omitted.

PARAMETER	DESCRIPTION
INQUIRY (cont)	NOTE An INQUIRY system precludes a DATACOM system and vice versa. The parameters INQUIRY and DATACOM should not both be set TRUE.
DATACOM	If DATACOM is set TRUE code is included in the MCP for the data-communication DATACOM II facilitities. If DATACOM is set FALSE, this code is omitted. NOTE An INQUIRY system precludes a DATACOM system and vice versa. The parameters INQUIRY and DATACOM should not both be set TRUE.
DCSPO	If DCSPO is set TRUE, SPO facilities are available to the user at remote stations. If DCSPO is set FALSE, remote stations have no SPO capabilities. NOTE The DCSPO and DCLOG facilities are functions of a DATACOM system. DCSPO and DCLOG should be set FALSE if DATACOM is set FALSE.
DCLOG	If DCLOG is set TRUE, logging facilities for remote stations are compiled into the MCP. If DCLOG is set FALSE this facility is not provided.

PARAMETER	DESCRIPTION
DCLOG (cont)	NOTE The DCSPO and DCLOG facilities are functions of a DATACOM system. DCSPO and DCLOG should be set FALSE if DATACOM is set FALSE.
CHECKLINK	The CHECKLINK option is now set TRUE when DEBUGGINE is set TRUE. This causes the MCP to perform extensive memory link checking prior to every GETSPACE and after every FORGETSPACE, if CHECK, option 25 is set. If an improper memory link is discovered, the message INVALID LINK is printed on the SPO and the system ceases to function.
DISKLOG	Disk files will be logged at the time the files are removed from the disk (e.g., after a CC REMOVE) under the following conditions: a. When a scratch file is CLOSEd. b. When a file is CLOSEd after obtaining more space. c. When a file is LOADed from a library tape with the same ⟨multi-file identification⟩ / ⟨file identification⟩ as a file on disk. d. When the operator enters a log-out instruction, LNDK. The LNDK message logs out all disk files and resets their creation date (H[3].[30:18]) and the creation time (H[1].[25:23]).

The format of the disk file log entry is as follows (MARK VIII, MCP #80):

DISK FILE LOG ENTRY FORMAT

WORD NO.	USE	REMARKS
1	TYPE CODE	=8
2	MFID	FIRST NAME OF FILE (7 CHR)
3	FID	SECOND NAME OF FILE (7 CHR)
4	USER CODE	(7 CHR)
5	CREATION DATE	IN FORM YYDDD (BCD)
6	CREATION TIME	1/60 SECOND (OCT)
7	DATE LOGGED	IN FORM YYDDD (BCD)
8	TIME LOGGED	1/60 SECOND (OCT)
9	NUMBER OF SEGMENTS	(OCT)
10	0	RESERVED FOR EXPANSION

APPENDIX B
DECK FORMAT FOR INTRINSICS

GENERAL.

This appendix contains the card deck necessary to compile the intrinsics.

```
? EXECUTE PATCH/MERGE
? DATA CARD
$.XX PATCHES FOR INTRINSICS.IX CONFLICTS
$*EXECUTE ESPOL/DISK .COMPILE STANDARD INTRINSICS
$*FILE STUFF = 0000000/INT TAPE
$*FILE NEWTAPE = INT/SOURCE TAPE
$*FILE DISK = INT/DISK
$*FILE TAPE = SYMBOL/INTRNSY
$*PROCESS = 60; IO = 60
$*DATA CARD
$-CARDS NEEDED FOR INTRINSICS COMPILATION
$ SET TIMESHARING = TRUE
$ SET TIMESHARING = FALSE
$ SET INQUIRY = TRUE
$ SET INQUIRY = FALSE
$ TAPE INTRINSIC LIST PRT STUFF
$#PATCH NUMBER 1 FOR INTRINSICS.IX CONTAINS N CARDS.
    ⟨N SORTED PATCHES⟩
$#PATCH NUMBER XX FOR INTRINSICS.IX CONTAINS N CARDS
    ⟨N SORTED PATCHES⟩
? END
```

It is noted that the $INTRINSIC option is required when compiling the symbolic file for the intrinsics of the B 5500 Programing System. When compiling these intrinsics, the ESPOL file DISK is equated to INT/DISK.

NOTE

When the INTRINSICS are compiled it is necessary to execute the program FILL/PRT which builds the disk file MCP/PRT used by DUMP/ANALYZE. If the INTRINSICS are compiled in the manner shown then FILL/PRT should be executed as follows:

```
? EXECUTE FILL/PRT
? FILE MCP = OOOOOOO/MCP TAPE
? FILE INT = OOOOOOO/INT TAPE
? END
```

APPENDIX C
COMPILER FILE IDENTIFIERS

Identifiers which may be required by control cards or program-parameter cards for modification to any of the programs contained in the B 5500 System are as follows:

ALGOL Compiler and XALGOL Compiler

Program Identifier	ALGOL
Program Identifier Suffix	DISK
File ID's	
Source program card input file ID	CARD*
Source program tape input file ID	TAPE
Source program tape output file ID	NEWTAPE
Source program printer output file ID	LINE
Source program punch output file ID	PNCH

COBOL Compiler

Program Identifier	COBOL
Program Identifier Suffix	DISK
File ID's	
Source program card input file ID	CARD
Source program tape input file ID	TAPE
Source program tape output file ID	NEWTAPE
Source program printer output file ID	LINE

FORTRAN Compiler

Program Identifier	FORTRAN
Program Identifier Suffix	DISK
File ID's	
Source program card input file ID	CARD
Source program tape input file ID	TAPE
Source program tape output file ID	NEWTAPE
Source program printer output file ID	LINE

ESPOL Compiler

Program Identifier	ESPOL
Program Identifier Suffix	DISK
File ID's	
Source program card input file ID	CARD
Source program tape input file ID	TAPE
Source program tape output file ID	NEWTAPE
Source program printer output file ID	LINE

---

* It should be noted that the ALGOL, XALGOL, COBOL, FORTRAN, and BASIC Compilers will use the card file immediately following the COMPILE card, regardless of file ID.

Card Output file for compiled code	DECK
Disk output file for compiled code	DISK
Card output file for stuff cards	STUFF

BASIC Compiler

Program Identifier	BASIC
Program Identifier Suffix	DISK
File ID's	
Source program card input file ID	CARD
Source program tape input file ID	TAPE
Source program tape output file ID	NEWTAPE
Source program printer output file ID	LINE

APPENDIX D
DECK FORMATS FOR COMPILERS

GENERAL.

This appendix contains the decks necessary to compile the ESPOL, ALGOL, COBOL, FORTRAN and BASIC Compilers.

ESPOL.

```
? EXECUTE PATCH/MERGE
? DATA CARD
$.XX PATCHES FOR ESPOL.IX CONFLICTS
$*COMPILE ESPOL/DISC ALGOL LIBRARY
$*ALGOL PROCESS = 60; ALGOL IO = 60
$*ALGOL FILE TAPE = SYMBOL/ESPOLSY
$*DATA CARD
$-CARDS NEEDED FOR ESPOL COMPILATION
$ TAPE LIST PRT CHECK
$#PATCH NUMBER 1 FOR ESPOL.IX CONTAINS N CARDS
    <N SORTED PATCHES>
$#PATCH NUMBER XX FOR EXPOL.IX CONTAINS N CARDS
    <N SORTED PATCHES>
? END
```

ALGOL.

```
? EXECUTE PATCH/MERGE
? DATA CARD
$.XX PATCHES FOR ALGOL.IX CONFLICTS
$*COMPILE ALGOL/DISC ALGOL LIBRARY
$*ALGOL PROCESS = 60; ALGOL IO = 60
$*ALGOL FILE TAPE = SYMBOL/ALGOLSY
$*FILE LINE = LINE BACK UP DISK
$*FILE NEWTAPE = "OCRDIMG" TAPE
$*FILE PNCH = PNCH PUNCH
$*DATA CARD
$-CARDS NEEDED FOR ALGOL COMPILATION
$ TAPE LIST PRT CHECK
$#PATCH NUMBER 1 FOR ALGOL.IX CONTAINS N CARDS
    <N SORTED PATCHES>
$#PATCH NUMBER XX FOR ALGOL.IX CONTAINS N CARDS
    <N SORTED PATCHES>
? END
```

NOTE

The same setup is used to compile XALGOL (SYMBOL/XALGLSY).

COBOL.

```
? EXECUTE PATCH/MERGE
? DATA CARD
$.XX PATCHES FOR COBOL.IX CONFLICTS
$*COMPILE COBOL/DISC ALGOL LIBRARY
$*ALGOL STACK = 750
$*ALGOL PROCESS = 60; ALGOL IO = 60
$*ALGOL FILE TAPE = SYMBOL/COBOLSY
$*FILE NEWTAPE = SOLT TAPE
$*FILE LINE = LINE BACK UP DISK
$*DATA CARD
$-CARDS NEEDED FOR COBOL COMPILATION
$ TAPE LIST PRT CHECK
$#PATCH NUMBER 1 FOR COBOL.IX CONTAINS N CARDS
    ⟨N SORTED PATCHES⟩
$#PATCH NUMBER XX FOR COBOL.IX CONTAINS N CARDS
    ⟨N SORTED PATCHES⟩
? END
```

FORTRAN.

```
? EXECUTE PATCH/MERGE
? DATA CARD
$.XX PATCHES FOR FORTRAN.IX CONFLICTS
$*COMPILE FORTRAN/DISC ALGOL LIBRARY
$*ALGOL PROCESS = 60; ALGOL IO = 60
$*ALGOL FILE TAPE = SYMBOL/FORTSY
$*FILE LINE = LINE BACK UP DISK
$*FILE NEWTAPE = FORTSYN/TAPE
$*DATA CARD
$-CARDS NEEDED FOR FORTRAN COMPILATION
$ TAPE LIST PRT CHECK
$#PATCH NUMBER 1 FOR FORTRAN.IX CONTAINS N CARDS
    ⟨N SORTED PATCHES⟩
$#PATCH NUMBER XX FOR FORTRAN.IX CONTAINS N CARDS
    ⟨N SORTED PATCHES⟩
? END
```

BASIC.

```
? EXECUTE PATCH/MERGE
? DATA CARD
$.XX PATCHES FOR BASIC.IX CONFLICTS
$*COMPILE BASIC/DISC ALGOL LIBRARY
$*ALGOL PROCESS = 60; ALGOL IO = 60
$*ALGOL FILE TAPE = SYMBOL/BASICSY TAPE
$*DATA CARD
```

```
                    .
                    .
                    .
$-CARDS NEEDED FOR BASIC COMPILATION
$ TAPE LIST PRT CHECK
$#PATCH NUMBER 1 FOR BASIC.IX CONTAINS N CARDS
    ⟨N SORTED PATCHES⟩
$#PATCH NUMBER XX FOR BASIC.IX CONTAINS N CARDS
    ⟨N SORTED PATCHES⟩
? END
```

APPENDIX E
CARD LOAD SELECT PROGRAMS

GENERAL.

The following three deck structures are used to produce the cold start, cool start and memory dump decks, and appear in that order.

```
? EXECUTE ESPOL/DISK .COMPILE MKIX COLD START
? FILE LINE = LINE BACK UP DISK
? FILE TAPE = SYMBOL/COLDSY
? DATA CARD
$ TAPE LIST PRT DECK
                                              99999999
? END

? EXECUTE ESPOL/DISK .COMPILE MKIX COOL START
? FILE TAPE = SYMBOL/COOLSY
? FILE LINE = LINE BACK UP DISK
? DATA CARD
$ TAPE LIST PRT DECK
                                              99999999
? END

? EXECUTE EPSOP/DISK .COMPILE DUMP CORE TO TAPE
? FILE TAPE = SYMBOL/DUMPTAP
? FILE LINE = LINE BACK UP DISK
? DATA CARD
$ TAPE LIST PRT DECK
                                              99999999
? END
```

BURROUGHS CORPORATION
DATA PROCESSING PUBLICATIONS
REMARKS FORM

TITLE: _____ FORM: _____

_____ DATE: _____

_____

CHECK TYPE OF SUGGESTION:

☐ ADDITION ☐ DELETION ☐ REVISION ☐ ERROR

GENERAL COMMENTS AND/OR SUGGESTIONS FOR IMPROVEMENT OF PUBLICATION:

FROM: NAME _____ DATE _____

TITLE _____

COMPANY _____

ADDRESS _____

_____

tear along dotted line

STAPLE

FOLD DOWN SECOND FOLD DOWN

BUSINESS REPLY MAIL
First Class Permit No. 817, Detroit, Mich. 48232

Burroughs Corporation
6071 Second Avenue
Detroit, Michigan 48232

attn: Sales Technical Services
Systems Documentation

FOLD UP FIRST FOLD UP

1043825 9-69 Printed in U.S. America