

Am29331

16-Bit Microprogram Sequencer

PRELIMINARY

DISTINCTIVE CHARACTERISTICS

- **16-Bits Address up to 64K Words**
Supports 80-90 ns microcycle time for a 32-bit high-performance system when used with the other members of the Am29300 Family.
- **Real-Time Interrupt Support**
Micro-trap and interrupts are handled transparently at any microinstruction boundary.
- **Built-In Conditional Test Logic**
Generates a full set of branch conditions from four ALU status bits. Has eight external test inputs plus a polarity input. Test multiplexer selects one out of 16 test inputs.
- **Break-Point Logic**
Built-in address comparator allows break-points in the microcode for debugging and statistics collection.
- **Master/Slave Error Checking**
Two sequencers can operate in parallel as a master and a slave. The slave generates a fault flag for unequal results.
- **33-Level Stack**
Provides support for interrupts, loops, and subroutine nesting. It can be accessed through the D-bus to support diagnostics.

GENERAL DESCRIPTION

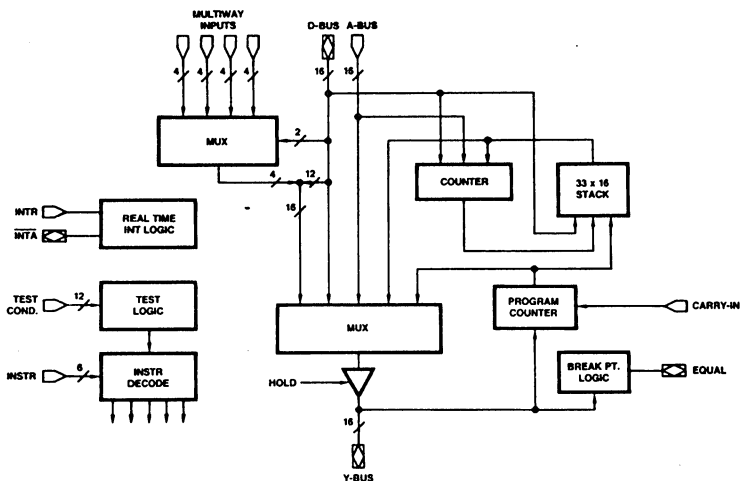
The Am29331 is a 16-bit wide, high-speed single-chip sequencer designed to control the execution sequence of microinstructions stored in the microprogram memory. The instruction set is designed to resemble high-level language constructs, thereby bringing high-level language programming to the micro level.

The Am29331 is interruptible at any microinstruction boundary to support real-time interrupts. Interrupts are handled transparently to the microprogrammer as an unexpected procedure call. Traps are also handled transparently at any microinstruction boundary. This feature allows re-execution of the prior microinstruction. Two separate buses are provided to bring a branch address directly into the chip from two sources to avoid slow turn-on and turn-off times

for different sources connected to the data-input bus. Four sets of multiway inputs are also provided to avoid slow turn-on and turn-off times for different branch-address sources. This feature allows implementation of table look-up or use of external conditions as part of a branch address. The thirty-three-deep stack provides the ability to support interrupts, loops, and subroutine nesting. The stack can be read through the D-bus to support diagnostics or to implement multitasking at the micro-architecture level. The master/slave mode provides a complete function check capability for the device.

The Am29331 is designed with the IMOX™ process which allows internal ECL circuits with TTL-compatible I/O. It is housed in a 120-lead pin-grid-array package.

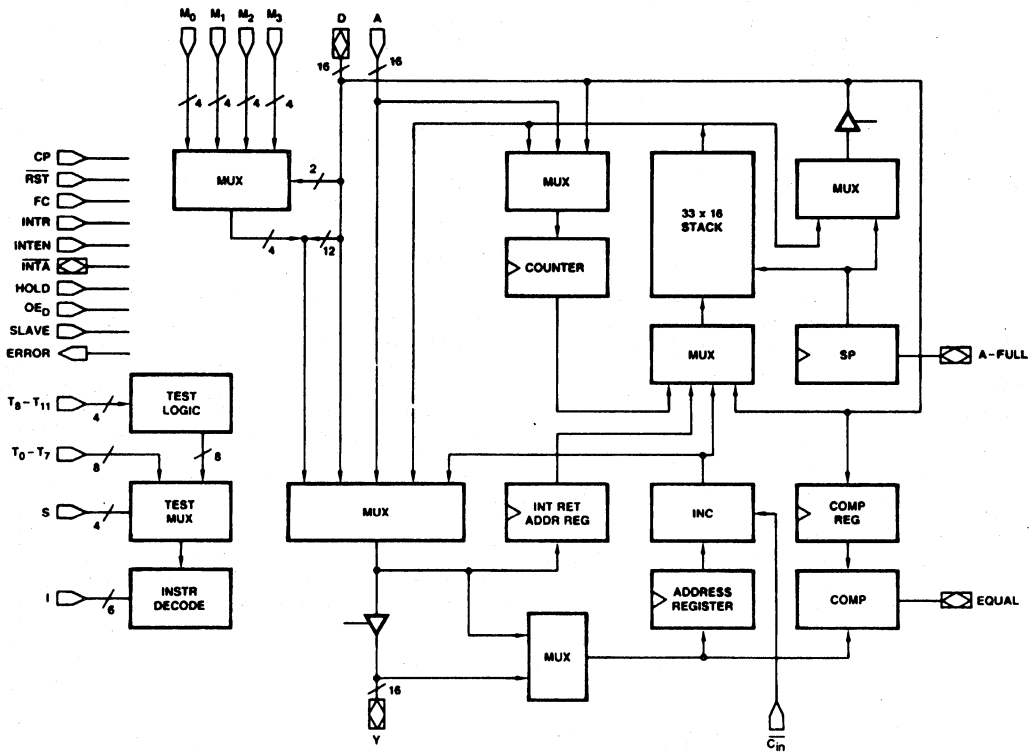
SIMPLIFIED BLOCK DIAGRAM



BD006090

RELATED PRODUCTS

Part No.	Description
Am29323	32 x 32 Parallel Multiplier
Am29325	32-Bit Floating Point Processor
Am29332	32-Bit Extended-Function ALU
Am29334	64 x 18 Four-Port, Dual-Access Register File

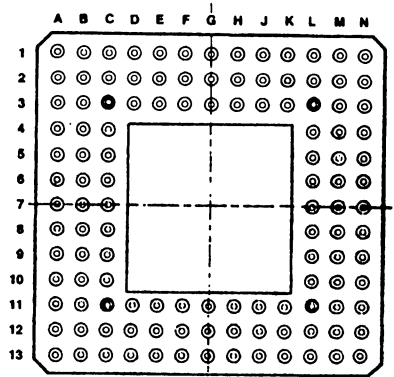


BD006101

Figure 1. Am29331 Detailed Block Diagram

CONNECTION DIAGRAM

120-Lead PGA*



CD009170

*Pinout observed from pin side of package.

PIN DESIGNATIONS (Sorted by Pin No.)

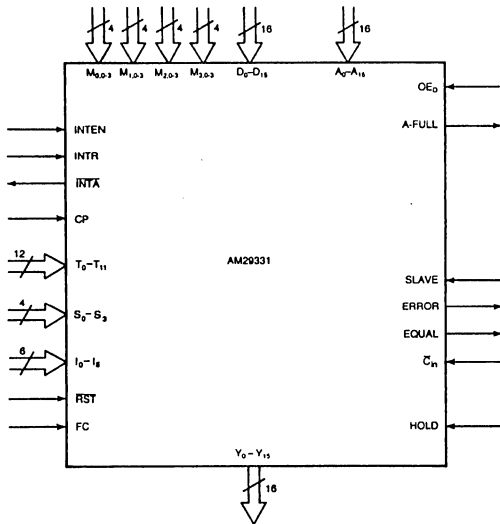
PIN NO.	PIN NAME	PAD NO.	PIN NO.	PIN NAME	PAD NO.	PIN NO.	PIN NAME	PAD NO.	PIN NO.	PIN NAME	PAD NO.
-	-	99	C-5	Y ₂	115	H-2	M _{3, 3}	10	M-5	A ₁₃	80
-	-	97	C-6	ECL GND	113	H-3	ECL VCC	68	M-6	D ₁₂	81
-	-	39	C-7	A ₄	52	H-11	I ₀	34	M-7	Y ₁₂	82
-	-	37	C-8	ECL VCC	53	H-12	S ₁	95	M-8	Y ₁₁	25
A-1	M _{0, 0}	1	C-9	Y ₅	109	H-13	S ₃	94	M-9	A ₁₀	86
A-2	D ₀	120	C-10	Y ₆	48	J-1	TTL GND	11	M-10	D ₉	87
A-3	TTL VCC	59	C-11	T ₃	44	J-2	EQUAL	71	M-11	D ₈	89
A-4	A ₁	58	C-12	T ₂	104	J-3	A-FULL	70	M-12	A ₈	30
A-5	TTL GND	56	C-13	T ₉	41	J-11	ECL VCC	38	M-13	I ₅	91
A-6	A ₃	114	D-1	M _{2, 1}	4	J-12	ECL VCC	38	N-1	D ₁₅	16
A-7	Y ₃	54	D-2	M _{1, 1}	63	J-13	ECL VCC	38	N-2	A ₁₅	76
A-8	D ₅	51	D-3	M _{0, 1}	3	K-1	RST	13	N-3	TTL VCC	17
A-9	TTL GND	50	D-11	T ₆	102	K-2	OED	72	N-4	Y ₁₄	19
A-10	D ₆	49	D-12	T ₅	43	K-3	ERROR	12	N-5	TTL GND	20
A-11	TTL VCC	47	D-13	T ₄	103	K-11	I ₃	92	N-6	Y ₁₃	21
A-12	A ₇	106	E-1	C _{in}	5	K-12	I ₂	33	N-7	D ₁₁	24
A-13	Y ₇	46	E-2	M _{0, 2}	65	K-13	I ₁	93	N-8	A ₁₁	84
B-1	M _{1, 0}	61	E-3	M _{3, 1}	64	L-1	INTR	14	N-9	TTL GND	26
B-2	A ₀	60	E-11	ECL GND	98	L-2	INTEN	74	N-10	A ₉	28
B-3	Y ₀	119	E-12	ECL GND	98	L-3	INTA	73	N-11	TTL VCC	29
B-4	Y ₁	117	E-13	ECL GND	98	L-4	D ₁₄	18	N-12	Y ₈	90
B-5	A ₂	116	F-1	M _{1, 2}	6	L-5	D ₁₃	79	N-13	FC	31
B-6	D ₃	55	F-2	M _{2, 2}	66	L-6	ECL GND	23			
B-7	D ₄	112	F-3	ECL GND	8	L-7	A ₁₂	22			
B-8	Y ₄	111	F-11	T ₁₀	100	L-8	ECL VCC	83			
B-9	A ₅	110	F-12	T ₇	42	L-9	D ₁₀	85			
B-10	A ₆	108	F-13	T ₈	101	L-10	Y ₁₀	27			
B-11	D ₇	107	G-1	M _{1, 3}	9	L-11	Y ₉	88			
B-12	T ₁	45	G-2	M _{0, 3}	67	L-12	I ₄	32			
B-13	T ₀	105	G-3	M _{3, 2}	7	L-13	S ₂	35			
C-1	M _{2, 0}	2	G-11	T ₁₁	40	M-1	SLAVE	75			
C-2	M _{3, 0}	62	G-12	S ₀	36	M-2	HOLD	15			
C-3	D ₁	118	G-13	CP	96	M-3	Y ₁₅	77			
C-4	D ₂	57	H-1	M _{2, 3}	69	M-4	A ₁₄	78			

PIN DESIGNATIONS

(Sorted by Pin Name)

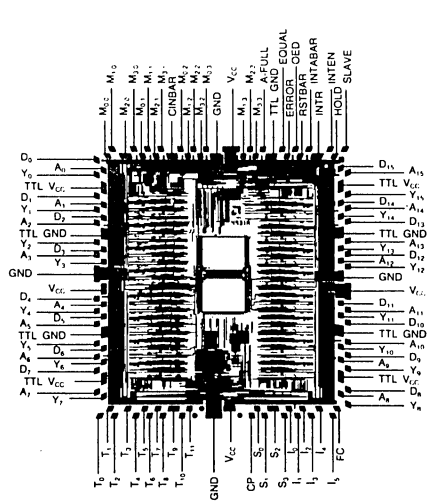
PIN NAME	PIN NO.	PAD NO.	PIN NAME	PIN NO.	PAD NO.	PIN NAME	PIN NO.	PAD NO.	PIN NAME	PIN NO.	PAD NO.
-	-	37	D ₈	M-11	89	INTEN	L-2	74	T ₆	D-11	102
-	-	39	D ₉	M-10	87	INTR	L-1	14	T ₇	F-12	42
-	-	97	D ₁₀	L-9	85	M _{0, 0}	A-1	1	T ₈	F-13	101
-	-	99	D ₁₁	N-7	24	M _{0, 1}	D-3	3	T ₉	C-13	41
A-FULL	J-3	70	D ₁₂	M-6	81	M _{0, 2}	E-2	65	T ₁₀	F-11	100
A ₀	B-2	60	D ₁₃	L-5	79	M _{0, 3}	G-2	67	T ₁₁	G-11	40
A ₁	A-4	58	D ₁₄	L-4	18	M _{1, 0}	B-1	61	TTL GND	J-1	11
A ₂	B-5	116	D ₁₅	N-1	16	M _{1, 1}	D-2	63	TTL GND	N-5	20
A ₃	A-6	114	ECL GND	E-12	98	M _{1, 2}	F-1	6	TTL GND	A-9	50
A ₄	C-7	52	ECL GND	E-13	98	M _{1, 3}	G-1	9	TTL GND	N-9	26
A ₅	B-9	110	ECL GND	E-11	98	M _{2, 0}	C-1	2	TTL GND	A-5	56
A ₆	B-10	108	ECL GND	F-3	8	M _{2, 1}	D-1	4	TTL Vcc	N-3	17
A ₇	A-12	106	ECL GND	L-6	23	M _{2, 2}	F-2	66	TTL Vcc	N-11	29
A ₈	M-12	30	ECL Vcc	C-6	113	M _{2, 3}	H-1	69	TTL Vcc	A-3	59
A ₉	N-10	28	ECL Vcc	J-13	38	M _{3, 0}	C-2	62	TTL Vcc	A-11	47
A ₁₀	M-9	86	ECL Vcc	H-3	68	M _{3, 1}	E-3	64	Y ₀	B-3	119
A ₁₁	N-8	84	ECL Vcc	C-8	53	M _{3, 2}	G-3	7	Y ₁	B-4	117
A ₁₂	L-7	22	ECL Vcc	L-8	83	M _{3, 3}	H-2	10	Y ₂	C-5	115
A ₁₃	M-5	80	ECL Vcc	J-12	38	OED	K-2	72	Y ₃	A-7	54
A ₁₄	M-4	78	ECL Vcc	J-11	38	RST	K-1	13	Y ₄	B-8	111
A ₁₅	N-2	76	EQUAL	J-2	71	S ₀	G-12	36	Y ₅	C-9	109
C _{in}	E-1	5	ERROR	K-3	12	S ₁	H-12	95	Y ₆	C-10	48
CP	G-13	96	FC	N-13	31	S ₂	L-13	35	Y ₇	A-13	46
D ₀	A-2	120	HOLD	M-2	15	S ₃	H-13	94	Y ₈	N-12	90
D ₁	C-3	118	I ₀	H-11	34	SLAVE	M-1	75	Y ₉	L-11	88
D ₂	C-4	57	I ₁	K-13	93	T ₀	B-13	105	Y ₁₀	L-10	27
D ₃	B-6	55	I ₂	K-12	33	T ₁	B-12	45	Y ₁₁	M-8	25
D ₄	B-7	112	I ₃	K-11	92	T ₂	C-12	104	Y ₁₂	M-7	82
D ₅	A-8	51	I ₄	L-12	32	T ₃	C-11	44	Y ₁₃	N-6	21
D ₆	A-10	49	I ₅	M-13	91	T ₄	D-13	103	Y ₁₄	N-4	19
D ₇	B-11	107	INTA	L-3	73	T ₅	D-12	43	Y ₁₅	M-3	77

LOGIC SYMBOL



LS002350

METALLIZATION AND PAD LAYOUT

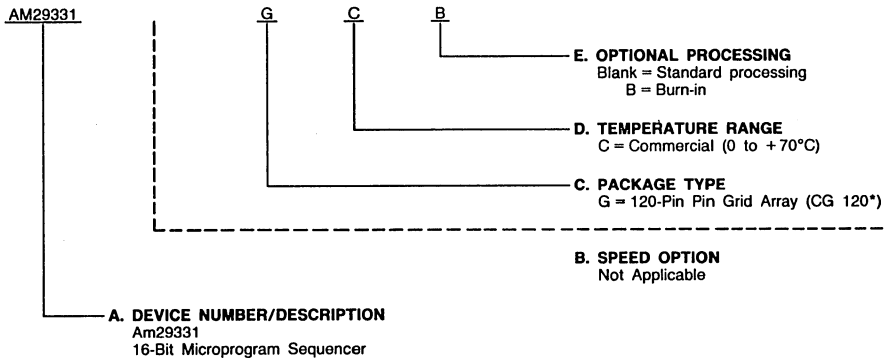


ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

- A. Device Number
- B. Speed Option (if applicable)
- C. Package Type
- D. Temperature Range
- E. Optional Processing



* Preliminary. Subject to Change.

Valid Combinations

Valid Combinations	
AM29331	GC, GCB

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released valid combinations, and to obtain additional data on AMD's standard military grade products.

PIN DESCRIPTION

D₀ - D₁₅ Data (Bidirectional, Three-State)

Input to address multiplexer, counter, stack, and comparator register. Output for stack and stack pointer.

A₀ - A₁₅ Alternate Data (Input)

Input to address multiplexer and counter.

M₀₋₃, 0-3 Multiway (Input)

Four sets of multiway inputs providing 16-way branches. The first index refers to the set number.

Y₀ - Y₁₅ Address (Bidirectional, Three-State)

Output of microcode address. Input for interrupt address.

I₀ - I₅ Instruction (Input)

Selects one of sixty-four instructions.

T₀ - T₁₁ Test (Input)

Provides external test inputs.

S₀ - S₃ Select (Input)

Selects one of sixteen test conditions.

CP Clock Pulse (Input)

Clocks sequencer at the LOW-to-HIGH transition.

RST Reset (Input, Active LOW)

Resets the sequencer.

FC Force Continue (Input)

Overrides instruction with CONTINUE.

INTR Interrupt Request (Input)

Requests the sequencer to interrupt execution.

INTEN Interrupt Enable (Input)

Enables interrupts.

INTA Interrupt Acknowledge (Bidirectional, Three-State, Active LOW)

Indicates that an interrupt is accepted.

HOLD Hold (Input)

Stops the sequencer and three-states the outputs.

OE_D Output Enable - D-Bus (Input)

Enables the D-bus driver, provided that the sequencer is not in the hold or slave mode.

SLAVE Slave (Input)

Makes the sequencer a slave.

ERROR Error (Output, Three-State)

Indicates a master/slave error in the slave mode. Indicates a malfunctioning driver or contention of any output in the master mode.

C_{in} Carry In (Input, Active LOW)

Carry-in to the incrementer.

A-FULL Almost Full (Bidirectional, Three-State)

Indicates that $SP \geq 28$ (meaning there are five or less empty locations left on stack).

EQUAL Equal (Bidirectional, Three-State)

Indicates that the address comparator is enabled and has found a match.

FUNCTIONAL DESCRIPTION

Architecture

The major blocks of the sequencer are the address multiplexer, the address register (AR), the stack (with the top of stack denoted TOS), the counter (C), the test multiplexer with logic, and the address comparison register (R) (Figure 1). The bidirectional D-bus provides branch addresses and iteration counts; it also allows access to the stack from the outside. The A-bus may be used for map addresses. There are four sets of four-bit multiway branch inputs (M). The bidirectional Y-bus either outputs microprogram addresses or inputs interrupt addresses. The buses are all 16 bits wide. Figure 1 shows a detailed block diagram of the sequencer.

Address Multiplexer

The address multiplexer can select an address from any of five sources:

- 1) A branch address supplied by the D-bus
- 2) A branch address supplied by the A-bus
- 3) A multiway-branch address
- 4) A return or loop address from the top of stack
- 5) The next sequential address from the incrementer

Multiway Branch Address

A multiway-branch address is formed by substituting the lower four bits of the address on the D-bus (D₃D₂D₁D₀) with one of the four sets (M₀, M₁, M₂ or M₃) of four-bit multiway-branch addresses. The multiway-branch set is selected by the number D₁D₀, while the bits D₃ and D₂ are "don't cares."

Address Register

The address register contains the current address. It is loaded from the interrupt multiplexer and feeds the incrementer. The incrementer is inhibited if $\overline{C_{in}}$ is taken HIGH.

Stack

A 33-word-deep and 16 bit-wide stack provides first-in last-out storage for return addresses, loop addresses, and counter values. Items to be pushed come from the incrementer, the interrupt-return-address register, the counter, or the D-bus. Items popped go to the address multiplexer, the counter, or the D-bus.

The access to the stack via the D-bus may be used for context switching, stack extension, or diagnostics. As the stack is only accessible from the top, stack extension is done by temporarily storing the whole or some lower part of the stack outside the sequencer. The save and the later restore are done with pop and push operations respectively, at balanced points in the microprogram, for example, points with the same stack depth. The internal D-bus driver must be turned on when popping an item to the D-bus; if the driver is off, the item will be unstacked instead. The driver is normally turned on when the Output Enable signal is asserted and the sequencer is not being reset (OE_D = 1, \overline{RST} = 1).

The stack pointer is a modulo 64 counter, which is incremented on each push and decremented on each pop. The stack pointer is reset to zero when the sequencer is reset, but the pointer may also be reset by instruction. Thus, the stack pointer indicates the number of items on the stack as long as stack overflow or underflow has not occurred. Overflow happens when an item is pushed onto a full stack, whereby the item at the bottom of the stack is overwritten. Underflow happens when an item is popped from an empty stack, in this case the item is undefined.

The contents of the stack pointer is present on the D-bus for all instructions except POP D, provided the driver is turned on. The output signal, A-FULL, is defined as $SP \geq 28$.

Counter

The counter may be used as a loop counter. It may be loaded from the D-bus, the A-bus, or via a pop from the stack. Its contents may also be pushed onto the stack.

A normal for-loop is set up by a FOR instruction, which loads the counter from the D- or A-bus with the desired number of iterations; the instruction also pushes onto the stack a loop address, that points to the next sequential instruction. The end of the loop is given by an unconditional END FOR instruction, which tests the counter value against the value one and then decrements the counter. If the values differ, the loop is repeated by selecting the address at the stack as the next address. If the values are equal, the loop is terminated by popping the stack, thereby removing the loop address, and selecting the address from the incrementer as the next address. The number of iterations is a 16-bit unsigned number, except that the number zero corresponds to 65,536 iterations. By pushing and popping counter values it is possible to handle nested loops.

Address Comparison

The sequencer is able to compare the address from the interrupt multiplexer with the contents of the comparator register. The instruction SET loads the comparator register with the address on the D-bus and enables the comparison, while CLEAR disables it. The comparison is disabled at reset. A HIGH is present at the output EQUAL if the comparison is enabled and the two addresses are equal. The comparison is useful for detection of a break point or counting the number of times a microinstruction at a specific address is executed.

Instruction Set

The sequencer has 64 instructions that are divided into four classes of 16 instructions each. The instruction lines $I_0 - I_5$ use I_5 and I_4 to select a class and $I_0 - I_3$ to select an instruction within a class. The classes are:

I_5	I_4	Classes
0	0	Conditional sequence control,
0	1	Conditional sequence control with inverted polarity,
1	0	Unconditional sequence control, and
1	1	Special function with implicit continue.

Note that for the first three classes I_5 forces the condition to be true and I_4 inverts the condition. The basic instructions of

the first three classes are shown in Table 1 and the instructions of the fourth class in Table 2.

Structured microprogramming is supported by sequencer instructions that singly or in pairs correspond to high-level-language control constructs. Examples are FOR I: = D DOWN TO 1 DO . . . END FOR and CASE N OF . . . END CASE. The instructions have been given high-level language names where appropriate. Figure 2 shows how to microprogram important control constructs; the high-level language is on the left and the microcode on the right.

Test Conditions

The condition for a conditional instruction is supplied by a test multiplexer, which selects one out of sixteen tests with the select lines $S_0 - S_3$. Twelve of these are supplied directly by the inputs $T_0 - T_{11}$, while the remaining four tests are generated by the test logic from the inputs $T_8 - T_{11}$. The following table shows the assignments.

$S_0 - S_3$	Test	Intended Use
0-7	$T_0 - T_7$	General
8	T_8	C (Carry)
9	T_9	N (Negative)
10	T_{10}	V (Overflow)
11	T_{11}	Z (Zero or equal)
12	$T_8 + T_{11}$	C + Z (Unsigned less than or equal, borrow mode)
13	$\overline{T_8} + T_{11}$	$\overline{C} + Z$ (Unsigned less than or equal)
14	$T_9 \oplus T_{10}$	$N \oplus V$ (Signed less than)
15	$(T_9 \oplus T_{10}) + T_{11}$	$(N \oplus V) + Z$ (Signed less than or equal)

Force Continue

The sequencer has a force continue (FC) input, which overrides the instruction inputs $I_0 - I_5$ with a CONTINUE instruction. This makes it possible to share the microinstruction field for the sequencer instruction with some other control or to initialize a writable control store.

Reset

In order to start a microprogram properly the sequencer must be reset. The reset works like an instruction overriding both the instruction input and the force continue input. The reset selects the address 0 at the address multiplexer, forces the EQUAL output to LOW, and disregards a potential interrupt request. It synchronously disables the address comparison and initializes the stack pointer to 0.

TABLE 1. INSTRUCTION SET for I₅I₄ = 00, 01, 10

I ₅ -I ₀	Instruction	Cond.: False*		Cond.: True*		Counter	Comp.	D-Mux
		Y	Stack	Y	Stack			
00, 10, 20	Goto D	INC	-	D	-	-	-	SP
01, 11, 21	Call D	INC	-	D	Push INC	-	-	SP
02, 12, 22	Exit D	INC	-	D	Pop	-	-	SP
03, 13, 23	End for D, C ≠ 1	INC	-	D	-	C←C-1	-	SP
	End for D, C = 1	INC	-	INC	-	C←C-1	-	SP
04, 14, 24	Goto A	INC	-	A	-	-	-	SP
05, 15, 25	Call A	INC	-	A	Push INC	-	-	SP
06, 16, 26	Exit A	INC	-	A	Pop	-	-	SP
07, 17, 27	End for A, C ≠ 1	INC	-	A	-	C←C-1	-	SP
	End for A, C = 1	INC	-	INC	-	C←C-1	-	SP
08, 18, 28	Goto M	INC	-	D:M	-	-	-	SP
09, 19, 29	Call M	INC	-	D:M	Push INC	-	-	SP
0A, 1A, 2A	Exit M	INC	-	D:M	Pop	-	-	SP
0B, 1B, 2B	End for M, C ≠ 1	INC	-	D:M	-	C←C-1	-	SP
	End for M, C = 1	INC	-	INC	-	C←C-1	-	SP
0C, 1C, 2C	End Loop	INC	Pop	TOS	-	-	-	SP
0D, 1D, 2D	Call Coroutine	INC	-	TOS	TOS←INC	-	-	SP
0E, 1E, 2E	Return	INC	-	TOS	Pop	-	-	SP
0F, 1F, 2F	End for, C ≠ 1	INC	Pop	TOS	-	C←C-1	-	SP
	End for, C = 1	INC	Pop	INC	Pop	C←C-1	-	SP

Cond. = (Test [S] OR I₅) XOR I₄

: = Concatination

C = Counter

INC = Output of Incrementer = AR + 1 (if $\overline{C_{in}}$ = LOW)

* For instructions 00 to 0F, these two columns are as marked. For instructions 10 to 1F, these two columns have inverted polarity; i.e., left column is test true, right column is test false. For instructions 20 to 2F, execution is unconditional and the right column is always used.

TABLE 2. INSTRUCTION SET for I₅I₄ = 11

I ₅ -I ₀	Instruction	Y	Stack	Counter	Comp.	D-Mux
30	Continue	INC	-	-	-	SP
31	For D	INC	Push INC	C←D	-	SP
32	Decrement	INC	-	C←C-1	-	SP
33	Loop	INC	Push INC	-	-	SP
34	Pop D	INC	Pop	-	-	TOS
35	Push D	INC	Push D	-	-	SP
36	Reset SP	INC	SP←0	-	-	SP
37	For A	INC	Push INC	C←A	-	SP
38	Pop C	INC	Pop	C←TOS	-	SP
39	Push C	INC	Push C	-	-	SP
3A	Swap	INC	TOS←C	C←TOS	-	SP
3B	Push C Load D	INC	Push C	C←D	-	SP
3C	Load D	INC	-	C←D	-	SP
3D	Load A	INC	-	C←A	-	SP
3E	Set	INC	-	-	R←D, Enable	SP
3F	Clear	INC	-	-	Disable	SP

R = Comp. Register

Interrupts

The sequencer may be interrupted at the completion of the current microcycle by asserting the interrupt request input INTR. The return address of the interrupted routine is saved on the stack, so that nested interrupts can be easily implemented. An interrupt is accepted if interrupts are enabled and the sequencer is not being reset or held (INTEN = HIGH, RST = HIGH, and HOLD = LOW). The interrupt-acknowledge output (INTA) goes LOW when an interrupt is accepted.

When there is no interrupt, addresses go from the address multiplexer to the Y-bus via the driver and to the address register and the comparator via the interrupt multiplexer. When there is an interrupt, the driver of the sequencer is turned off, an external driver is turned on, and the interrupt multiplexer is switched. The interrupt address is supplied via the external driver to the Y-bus, the address register and the comparator (Figure 3). In order to save the address from the address multiplexer, the address is stored in the interrupt return address register, which for simplicity is clocked every cycle. The next microinstruction is the first microinstruction of the interrupt routine (Figure 4).

In this cycle the address in the interrupt return address register is automatically pushed onto the stack. Therefore the microinstruction in this cycle must not use the stack; if a stack operation is programmed, the result is undefined. The instructions that do not use the stack are GOTO D, GOTO A, GOTO M, CONTINUE, DECREMENT, LOAD D, LOAD A, SET and CLEAR. A RETURN instruction terminates the interrupt routine and the interrupted routine is resumed. Interrupts only work with a single-level control path.

Traps

A trap is an unexpected situation linked to current microinstruction, that must be handled before the microinstruction completes and changes the state of the system. An example of such a situation is an attempt to read a word from memory across a word boundary in a single cycle. When a trap occurs, the current microinstruction must be aborted and re-executed after the execution of a trap routine, which in the meantime will take corrective measures. An interrupt, on the other hand, is not linked directly to the current microinstruction that can complete safely before an interrupt routine is executed.

Execution of a trap requires that the sequencer ignores the current microinstruction, selects the trap return address at the address multiplexer, and initiates an interrupt. This will save the trap return address on the stack and issue the trap address from an external source (Figure 5). The address register contains the address of the microinstruction in the

pipeline register, thus the address register already contains the trap return address when a trap occurs. This address can be selected by the address multiplexer by disabling the incrementer ($\overline{CIN} = 1$), and using the force continue mode (FC = 1). In this mode the sequencer ignores the current microinstruction. The remaining part of the trap handling is done by the interrupt (Figure 6). Thus the section on interrupts also applies to traps. There is one exception, however. The interrupt enable cannot be used as a trap enable as it does not control the force continue mode and the carry-in to the incrementer.

Hold Mode

The sequencer has a hold mode in which the operation is suspended.

When the HOLD signal goes active, the outputs (Y, INTA, A-FULL & EQUAL) are disabled and the sequencer enters the hold mode after the current cycle. While the sequencer is in this mode, the internal state is left unchanged and the D-bus is disabled. When the HOLD signal goes inactive, the outputs (Y, INTA, A-FULL & EQUAL) are enabled again and the sequencer leaves the hold mode after the cycle.

In a time multiplexed multi-microprocess system there may be one sequencer for all processes with microprogrammed context save and restore, or there may be one sequencer per microprocess permitting fast process switch. In the latter case the Y-buses of the sequencers are tied together and connected to a single microprogram store. A control unit decides on a cycle by cycle basis, what sequencer should be running and activates the HOLD signal to the remaining sequencers. The hold mode has higher priority than interrupts, and works independently of the reset. The hold mode can only be used with a single-level control path.

Master/Slave Configuration

In some systems reliability is very important. The master/slave configuration, that consists of two sequencers operated in parallel, is able to detect faults in both the interconnect and the internal function of the sequencers. One sequencer is the master and operates normally. The other is slave, i.e., all outputs except the signal ERROR are turned into inputs and connected to the outputs of the master. Since the slave is operated in parallel with the master, it can compare its result with the result of the master and signal an error if they differ. The error signal from the master indicates a malfunctioning driver or contention. Because a TTL output goes HIGH when power is missing, the ERROR signal also indicates power failure.

High-Level Language Constructs

An example of high-level language constructs using Am29331 instructions is given in Figure 2 (2-1, 2-2, 2-3, and 2-4).

```

REPEAT          LOOP
-              -
-              -
UNTIL CC       END LOOP NOT CC

WHILE CC DO    LOOP
-              IF NOT CC THEN EXIT L
-              -
-              -
END WHILE      END LOOP
                L:

LOOP           LOOP
-             -
IF CC THEN EXIT IF CC THEN EXIT L
-             -
END LOOP      END LOOP
                L:
    
```

Figure 2-1. Loops with Unknown Number of Iterations

```

FOR CNT: = 10 DOWN TO 1 DO  FOR D 10
-                             -
-                             -
END FOR                      END FOR
    
```

Figure 2-2. Loop with Known Number of Iterations

```

CASE I OF      PUSH D B
0: -          GOTO M
-            A: -
-            -, RETURN (TO B)
1: -          A + 2: -
-            -, RETURN (TO B)
2: -          A + 4: -
-            -, RETURN (TO B)
3: -          A + 6: -
-            -, RETURN
END CASE B:
    
```

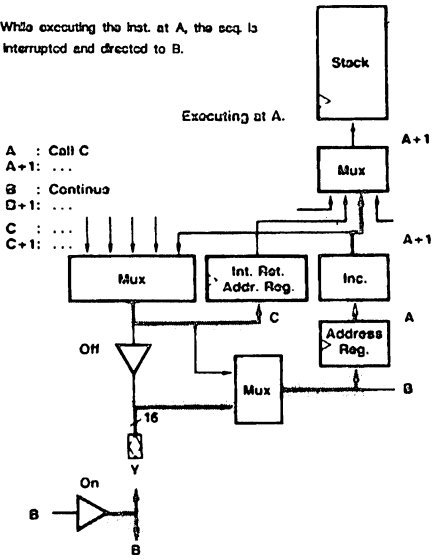
Figure 2-3. Case Statement
(with $D = A_{15} \dots A_4XX00$ and $M_{0, 0-3} = A_{31}1_00$ during the GOTO M instruction. A_1A_0 must be 00, and X signifies a don't care.)

```

IF X THEN     PUSH D C
IF Y THEN     IF NOT X THEN GOTO A
-             IF NOT Y THEN GOTO B
-             -
-             -, RETURN (TO C)
ELSE          B:
-             -
-             -, RETURN (TO C)
END IF
ELSE          A:
IF Z THEN     IF NOT Z THEN GOTO D
-             -
-             -, RETURN (TO D)
ELSE          D:
-             -
-             -, RETURN (TO C)
END IF
END IF       C:
    
```

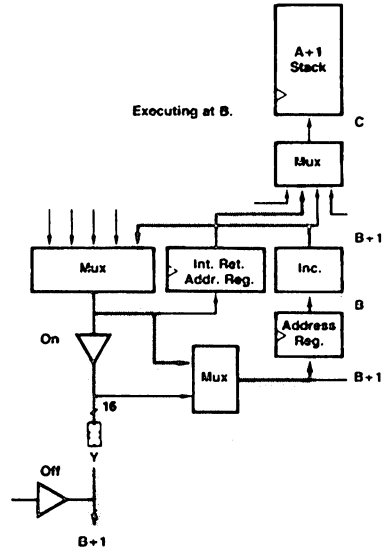
Figure 2-4. Double-Nested If Statement

While executing the inst. at A, the seq. is interrupted and directed to B.



AF004190

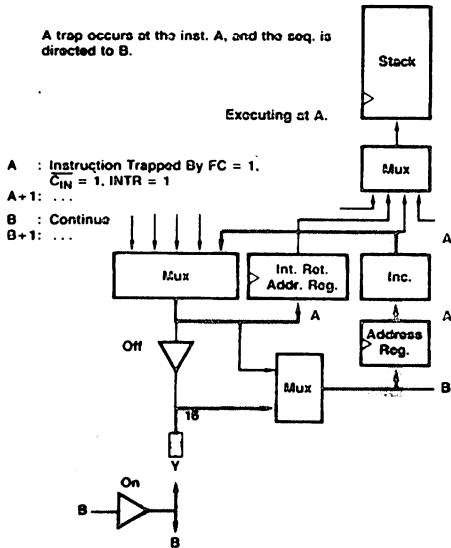
Figure 3. Am29331 Interrupt Cycle 1



AF004210

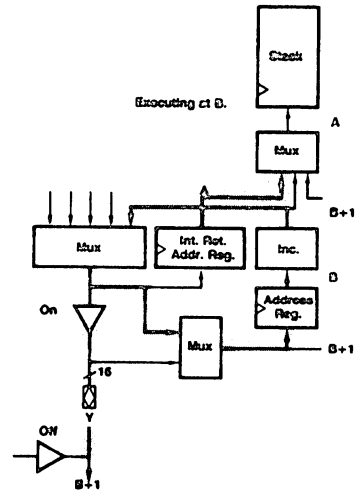
Figure 4. Am29331 Interrupt Cycle 2

A trap occurs at the inst. A, and the seq. is directed to B.



AF004200

Figure 5. Am29331 Traps Cycle 1



AF004180

Figure 6. Am29331 Traps Cycle 2

Instruction Set Definition

Legend: ● = Other instruction

⊙ = Instruction being described

CC = (Test [S₃ - S₀] OR I₅) XOR I₄

P = Test pass

F = Test fail

○ = Register in part

Opcode (I ₅ - I ₀)	Mnemonics	Description	Execution Example
20 24 28 2C	BRA_D BRA_A BRA_M BRA_S	Go to D. Unconditional branch to the address specified by the D inputs. Go to A. Unconditional branch to the address specified by the A inputs. Go to M. Unconditional branch to the address specified by the D inputs catenated with the multiway M inputs. Go to TOS. Unconditional branch to the address on the top of the stack. Also End Loop when used to terminate WHILE ... ENDWHILE loops.	<p style="text-align: right;">PF001730</p>
00 04 08 0C	BRCC_D BRCC_A BRCC_M BRCC_S	If CC is HIGH then branch to the address specified by the D inputs else continue. If CC is HIGH then branch to the address specified by the A inputs else continue. If CC is HIGH then branch to the address specified by the D inputs catenated with the multiway M inputs else continue. If CC is HIGH then branch to the address on the top of the stack else pop the stack and continue. Also End Loop when used to terminate REPEAT ... UNTIL loops.	<p style="text-align: right;">PF001810</p>
10 14 18 1C	BRNC_D BRNC_A BRNC_M BRNC_S	If CC is LOW then branch to the address specified by the D inputs else continue. If CC is LOW then branch to the address specified by the A inputs else continue. If CC is LOW then branch to the address specified by the D inputs catenated with the multiway M inputs else continue. If CC is LOW then branch to the address on the top of the stack else pop the stack and continue. Also End Loop when used to terminate REPEAT ... UNTIL loops.	<p style="text-align: right;">PF001750</p>

Note: Opcode numbers are in hexadecimal notation.

Opcode (I ₅ - I ₀)	Mnemonics	Description	Execution Example
21	CALL_D	Call D. Unconditional branch to the subroutine address specified by the D inputs and push the PC on the stack.	
25	CALL_A	Call A. Unconditional branch to the subroutine address specified by the A inputs and push the PC on the stack.	
29	CALL_M	Call M. Unconditional branch to the subroutine address specified by the D inputs catenated with the multiway M inputs and push the PC on the stack.	
2D	CALL_S	Call TOS. Exchange PC and TOS. Also call coroutine.	
01	CCC_D	If CC is HIGH then call the subroutine address specified by the D inputs else continue.	
05	CCC_A	If CC is HIGH then call the subroutine address specified by the A inputs else continue.	
09	CCC_M	If CC is HIGH then call the subroutine address specified by the D inputs catenated with the multiway M inputs else continue.	
0D	CCC_S	If CC is HIGH then call the address on the top of the stack else continue. Also used for conditional coroutine calls.	
11	CNC_D	If CC is LOW then call the address specified by the D inputs else continue.	
15	CNC_A	If CC is LOW then call the address specified by the A inputs else continue.	
19	CNC_M	If CC is LOW then call the address specified by the D inputs catenated with the multiway M inputs else continue.	
1D	CNC_S	If CC is LOW then call the address on the top of the stack else continue. Also a conditional coroutine call.	
22	EXIT_D	Exit to D. Unconditional branch to the address specified by the D inputs and pop the stack.	
26	EXIT_A	Exit to A. Unconditional branch to the address specified by the A inputs and pop the stack.	
2A	EXIT_M	Exit to M. Unconditional branch to the address specified by the D inputs catenated with the multiway M inputs and pop the stack.	
2E	EXIT_S	Exit to TOS. Unconditional branch to the address specified by the TOS and pop the stack. Also a return.	

Note: Opcode numbers are in hexadecimal notation.

Opcode (I ₅ - I ₀)	Mnemonics	Description	Execution Example
02	XTCC_D	If CC is HIGH then exit to the address specified by the D inputs and pop the stack else continue with no pop.	
06	XTCC_A	If CC is HIGH then exit to the address specified by the A inputs and pop the stack else continue with no pop.	
0A	XTCC_M	If CC is HIGH then exit to the address specified by the D inputs catenated to the multiway M inputs and pop the stack else continue with no pop.	
0E	XTCC_S	If CC is HIGH then exit to the address specified by the TOS and pop the stack else continue with no pop. Also RETCC.	
PF001800			
12	XTNC_D	If CC is LOW then exit to the address specified by the D inputs and pop the stack else continue.	
16	XTNC_A	If CC is LOW then exit to the address specified by the A inputs and pop the stack else continue.	
1A	XTNC_M	If CC is LOW then exit to the address specified by the D inputs catenated with the multiway M inputs and pop the stack else continue.	
1E	XTNC_S	If CC is LOW then exit to the address specified by the TOS and pop the stack else continue. Also RETNC.	
PF001810			
23	DJMP_D	If the counter is not equal to one then decrement the counter and branch to the address specified by the D inputs else continue.	
27	DJMP_A	If the counter is not equal to one then decrement the counter and branch to the address specified by the A inputs else continue.	
2B	DJMP_M	If the counter is not equal to one then decrement the counter and branch to the address specified by the D inputs catenated with the multiway M inputs else continue.	
2F	DJMP_S	If the counter is not equal to one then decrement the counter and branch to the address on the top of the stack else decrement the counter, pop the stack and continue.	
PF001820			

Note: Opcode numbers are in hexadecimal notation.

Opcode (I ₅ - I ₀)	Mnemonics	Description	Execution Example
03	DJCC_D	If CC is HIGH and the counter is not equal to one then decrement the counter and branch to the address specified by the D inputs else decrement the counter and continue.	<p style="text-align: right;">PF001830</p>
07	DJCC_A	If CC is HIGH and the counter is not equal to one then decrement the counter and branch to the address specified by the A inputs else decrement the counter and continue.	
0B	DJCC_M	If CC is HIGH and the counter is not equal to one then decrement the counter and branch to the address specified by the D inputs catenated with the multiway M inputs else decrement the counter and continue.	
0F	DJCC_S	If CC is HIGH and the counter is not equal to one then decrement the counter and branch to the address on the top of the stack else decrement the counter, pop the stack and continue.	

13	DJNCC_D	If CC is LOW and the counter is not equal to one then decrement the counter and branch to the address specified by the D inputs else decrement the counter and continue.	<p style="text-align: right;">PF001840</p>
17	DJNCC_A	If CC is LOW and the counter is not equal to one then decrement the counter and branch to the address specified by the A inputs else decrement the counter and continue.	
1B	DJNCC_M	If CC is LOW and the counter is not equal to one then decrement the counter and branch to the address specified by the D inputs catenated with the multiway M inputs else decrement the counter and continue.	
1F	DJNCC_S	If CC is LOW and the counter is not equal to one then decrement the counter and branch to the address on the top of the stack else decrement the counter, pop the stack and continue.	

2E	RET	Unconditional return from subroutine.	<p style="text-align: right;">PF001850</p>
0E	RETCC	If CC is HIGH then return from subroutine else continue.	
1E	RETNC	If CC is LOW then return from subroutine else continue.	

Note: Opcode numbers are in hexadecimal notation.

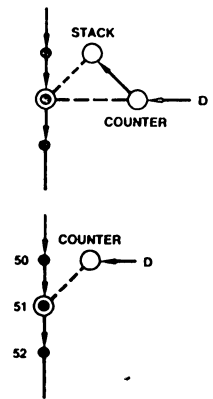
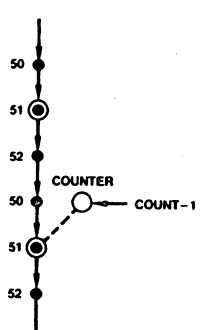
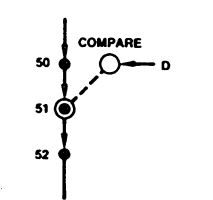
Opcode (I ₅ - I ₀)	Mnemonics	Description	Execution Example
31	FOR_D	Initialize loop. Push the PC on the stack, load the counter with the value of the D inputs and continue. Use with DJMP_S for FOR...NEXT loops.	
37	FOR_A	Initialize loop. Push the PC on the stack, load the counter with the value of the A inputs and continue. Use with DJMP_S for FOR...NEXT loops.	
33	LOOP	Initialize loop. Push the PC and continue. Use with BRCC_S for REPEAT...UNTIL loops or with XTCC_D and BRA_S for WHILE...ENDWHILE loops.	

PF001860

34	POP_D	Pop the stack, output the value on the D outputs and continue.	
38	POP_C	Pop the stack, place the value in the counter and continue.	
35	PUSH_D	Push the D inputs on the stack and continue.	
39	PUSH_C	Push the counter on the stack and continue.	
3A	SWAP	Exchange the counter and the top of stack and continue.	

PF001870

Note: Opcode numbers are in hexadecimal notation.

Opcode (15 - 10)	Mnemonics	Description	Execution Example
3B 3C 3D	STACK_C LOAD_D LOAD_A	Push the counter on the stack, load the counter with the value of the D inputs and continue. Load the counter with the value of the D inputs and continue. Load the counter with the value of the A inputs and continue.	 <p>The diagram illustrates the execution of three opcodes. It shows a vertical stack with addresses 50, 51, and 52. A counter is shown being pushed onto the stack at address 50, then loaded with the value of the D input, and finally loaded with the value of the A input.</p> <p style="text-align: right;">PF001880</p>
30 32 36	CONT DECR RESET_SP	Continue. Decrement the counter and continue. Reset the stack pointer and continue.	 <p>The diagram illustrates the execution of three opcodes. It shows a vertical stack with addresses 50, 51, and 52. The counter is shown being decremented and the stack pointer being reset.</p> <p style="text-align: right;">PF001890</p>
3E 3F	SET CLEAR	Load the comparison register with the value of the D inputs, enable the comparator and continue. Disable the comparator and continue.	 <p>The diagram illustrates the execution of two opcodes. It shows a vertical stack with addresses 50, 51, and 52. The comparison register is shown being loaded with the value of the D input and then disabled.</p> <p style="text-align: right;">PF001900</p>

Note: Opcode numbers are in hexadecimal notation.

APPLICATIONS

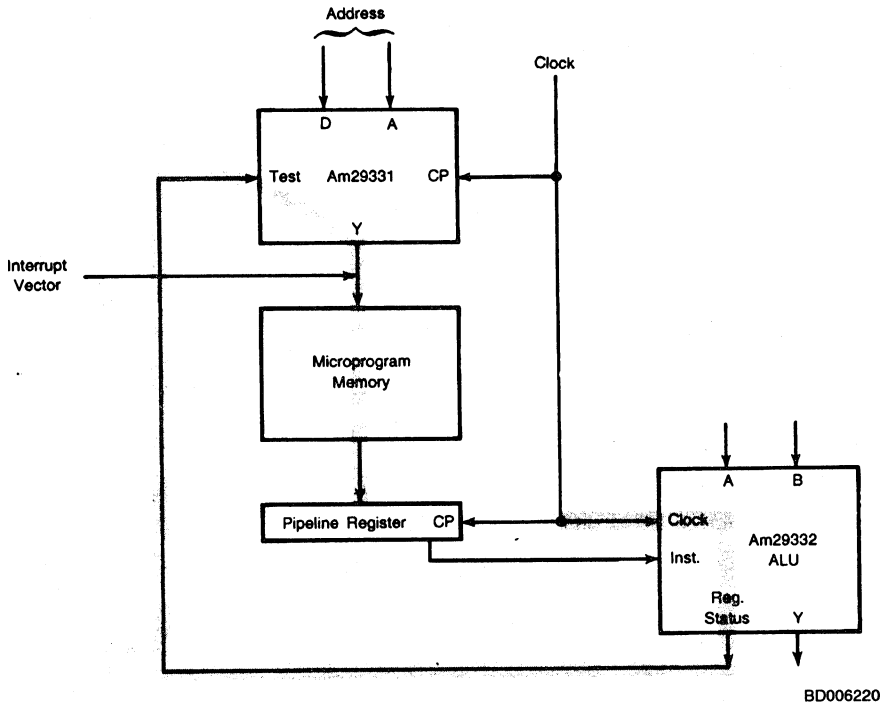


Figure 7. Typical Control-Path Architecture For Am29300 Family

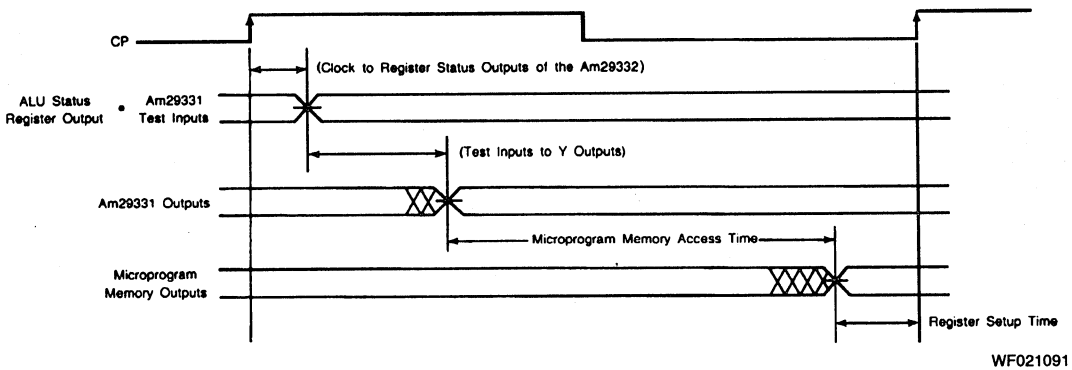
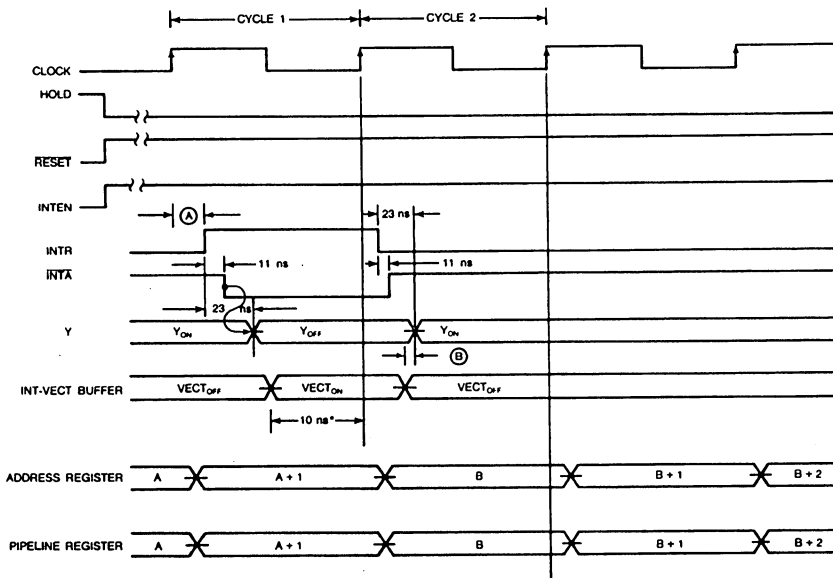


Figure 8. Cycle Timing Waveform*

* This waveform shows the timing relationship for the configuration shown in Figure 7.



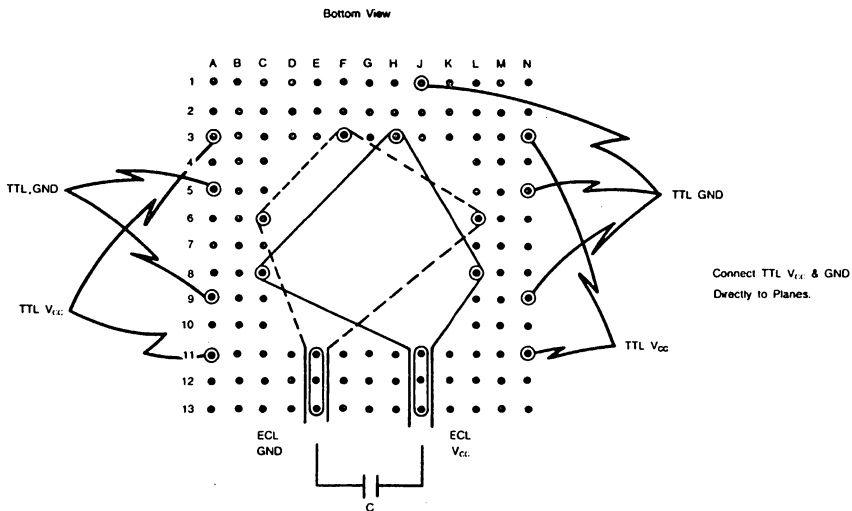
WF021101

Figure 9. Interrupt Timing Waveform*

Notes: (A) Interrupt Request comes from an interrupt-controller register. It reflects the CP \uparrow to INTR time of the interrupt controller.

(B) During Cycle 2, there may be contention on the Y-bus if Y-bus is turned ON before INT-VECT buffer is turned OFF.

*10 ns is the Y setup time with respect to CP \uparrow of the Am29331. Since the VECT address also goes to the microprogram memory and the pipeline register, minimum setup time will depend on the memory access time plus pipeline register data setup time.



Connect TTL V_{cc} & GND Directly to Planes.

AF004231

Figure 10. Suggested Printed-Circuit-Board Layout

Note: Connect ECL V_{cc} & GND directly to planes from E-13 and J-13.

ABSOLUTE MAXIMUM RATINGS

Storage Temperature -65 to +150°C
 Temperature Under Bias - T_C -55 to +125°C
 Supply Voltage to Ground Potential
 Continuous -0.5 to +7.0 V
 DC Voltage Applied to Outputs
 for High State -0.5 V to + V_{CC} Max
 DC Input Voltage -0.5 to +5.5 V

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices
 Temperature (T_A) 0 to +70°C
 Supply Voltage (V_{CC}) +4.75 to +5.25 V
 Air Velocity 200 linear feet per minute

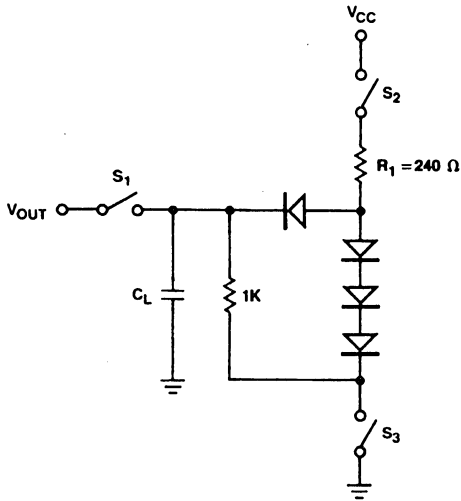
Operating ranges define those limits between which the functionality of the device is guaranteed.

DC CHARACTERISTICS over operating range unless otherwise specified

Parameters	Description	Test Conditions (Note 1)		Min.	Typ. (Note 2)	Max.	Units
V_{OH}	Output HIGH Voltage	$V_{CC} = \text{Min.}$ $V_{IN} = V_{IL}$ or V_{IH}	$I_{OH} = -1.6$ mA for $Y_0 - Y_{15}$, INTA	2.4			Volts
			$I_{OH} = -1.2$ mA for All Others				
V_{OL}	Output LOW Voltage	$V_{CC} = \text{Min.}$ $V_{IN} = V_{IL}$ or V_{IH}	$I_{OL} = 16$ mA for $Y_0 - Y_{15}$, INTA			0.5	Volts
			$I_{OL} = 12$ mA for All Others				
V_{IH}	Input HIGH Level	Guaranteed Input Logical HIGH Voltage for All Inputs		2.0			Volts
V_{IL}	Input LOW Level	Guaranteed Input Logical LOW Voltage for All Inputs				0.8	Volts
V_I	Input Clamp Voltage	$V_{CC} = \text{Min.}$, $I_{IN} = -18$ mA				-1.5	Volts
I_{IL}	Input LOW Current	$V_{CC} = \text{Max.}$, $V_{IN} = 0.5$ V	$Y_0 - Y_{15}$, $D_0 - D_{15}$, INTA, A-FULL, EQUAL			-0.55	mA
			$A_0 - A_{15}$, $M_0 - 3$, $0 - 3$, $I_0 - I_5$, $T_0 - T_{11}$, $S_0 - S_3$, FC, C_{in}			-0.50	
			OED			-1.0	
			SLAVE, HOLD			-1.5	
			CP, INTR, INTEN			-2.5	
			RST			-3.0	
I_{IH}	Input HIGH Current	$V_{CC} = \text{Max.}$, $V_{IN} = 2.4$ V	$Y_0 - Y_{15}$, $D_0 - D_{15}$, INTA, A-FULL, EQUAL			100	μ A
			$A_0 - A_{15}$, $M_0 - 3$, $0 - 3$, $I_0 - I_5$, $T_0 - T_{11}$, $S_0 - S_3$, FC, C_{in}			50	
			OED			100	
			SLAVE, HOLD			150	
			CP, INTR, INTEN			250	
			RST			300	
I_I	Input HIGH Current	$V_{CC} = \text{Max.}$, $V_{IN} = 5.5$ V				1.0	mA
I_{OZH} I_{OZL}	Off State (High-Impedance) Output Current	$V_{CC} = \text{Max.}$		$V_O = 2.4$ V		100	μ A
				$V_O = 0.5$ V		-550	
I_{SC}	Output Short Circuit Current (Note 3)	$V_{CC} = \text{Max.}$, +0.5 V $V_{OUT} = +0.5$ V		-15		-50	mA
I_{CC}	Power Supply Current (Note 4)	$V_{CC} = \text{Max.}$	COM'L Only	$T_A = 0$ to +70°C		1,300	mA
				$T_A = +70^\circ\text{C}$		1,000	
			MIL Only	$T_A = -55$ to +125°C			
				$T_A = +125^\circ\text{C}$			

- Notes: 1. For conditions shown as Min. or Max., use the appropriate value specified under Operating Ranges for the applicable device type.
 2. Typical values are for $V_{CC} = 5.0$ V, +25°C ambient and maximum loading.
 3. Not more than one output should be shorted at a time. Duration of the short circuit test should not exceed one second.
 4. Measured with all inputs LOW and outputs disabled.

SWITCHING TEST CIRCUIT THREE STATE OUTPUTS



KEY TO SWITCHING WAVEFORMS

WAVEFORM	INPUTS	OUTPUTS
	MUST BE STEADY	WILL BE STEADY
	MAY CHANGE FROM H TO L	WILL BE CHANGING FROM H TO L
	MAY CHANGE FROM L TO H	WILL BE CHANGING FROM L TO H
	DON'T CARE: ANY CHANGE PERMITTED	CHANGING: STATE UNKNOWN
	DOES NOT APPLY	CENTER LINE IS HIGH IMPEDANCE "OFF" STATE

KS000010

TC003420

- Notes:
1. $C_L = 50$ pF includes scope probe, wiring and stray capacitances without device in test fixture.
 2. S_1 , S_2 , S_3 are closed during function tests and all AC tests except output enable tests.
 3. S_1 and S_3 are closed while S_2 is open for t_{pZH} test.
 S_1 and S_2 are closed while S_3 is open for t_{pZL} test.
 4. $C_L = 5.0$ pF for output disable tests.

SWITCHING CHARACTERISTICS over operating range unless otherwise specified (Note 1)

A. Combinational Propagation Delays

From Input	To Output (Note 2)					
	Y	D	INTA	A-FULL	EQUAL	ERROR
D	20*	-	-	-	26	28
A	19	-	-	-	26	28
M	19	-	-	-	26	28
Y	-	-	-	-	25	27
I	28	31	-	-	29	29
T	30	-	-	-	30	32
S	30	-	-	-	30	32
CP	24	20/Z	-	20	31	33
RST	27	Z	16	-	27	29
FC	21	23	-	-	28	30
INTR	Z	-	12	-	35	31
INTEN	Z	-	12	-	35	31
HOLD	Z	-	Z	Z	25/Z	-
OED	-	Z	-	-	-	19
INTA	-	-	-	-	-	21
A-FULL	-	-	-	-	-	21
EQUAL	-	-	-	-	-	21
C _{in}	21	-	-	-	28	30
SLAVE	Z	Z	Z	Z	Z	-

Notes: 1. C_L = 50 pF; C_L = 5 pF for D and T.
 2. Z = Three-state output plus bus enable.

*This includes using D as Select lines for 16-way sets.

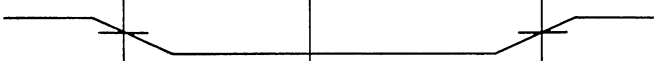
B. Output Enable/Disable Times

Input	Outputs	Enable	Disable
RST	Y	30	30
INTR			
INTEN			
HOLD			
SLAVE			
OED	D	30	30
RST			
SLAVE			
CP	D	30	30
HOLD	INTA	30	30
	A-FULL		
SLAVE	EQUAL		

C. Minimum Clock Requirement

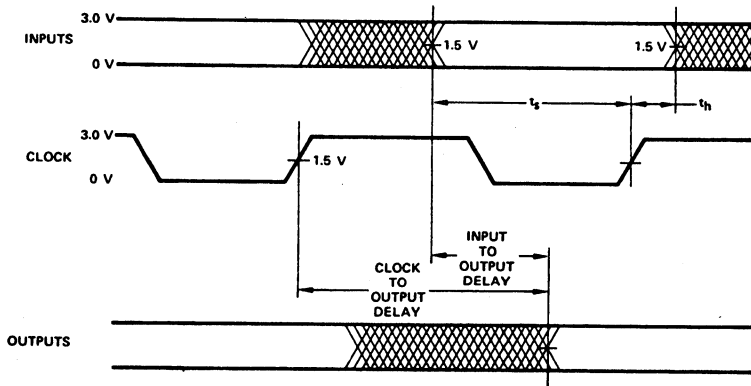
Min. LOW Time	Min. HIGH Time
20	20

D. Setup and Hold Times

Input	CP: 			
	Setup Time Before H → L	Hold Time After H → L	Setup Time Before L → H	Hold Time After L → H
D	-	-	15	4
A	-	-	14	4
M	-	-	14	3
Y	-	-	10	4
I	15	Do Not Change		2
FC	15	Do Not Change		1
T	-	-	20	1
S	-	-	20	1
$\overline{\text{RST}}$	-	-	15	2
INTR	-	-	12	3
INTEN	-	-	-	3
HOLD	-	-	12	4
$\overline{\text{C}}_{\text{in}}$	-	-	16	3

PRELIMINARY

SWITCHING WAVEFORMS



WFR02990

Notes on Test Methods

The following points give the general philosophy which we apply to tests which must be properly engineered if they are to be implemented in an automatic environment. The specifics of what philosophies applied to which test are shown.

1. Ensure the part is adequately decoupled at the test head. Large changes in supply current when the device switches may cause function failures due to V_{CC} changes.
2. Do not leave inputs floating during any tests, as they may oscillate at high frequency.
3. Do not attempt to perform threshold tests at high speed. Following an input transition, ground current may change by as much as 400 mA in 5 - 8 ns. Inductance in the ground cable may allow the ground pin at the device to rise by hundreds of millivolts momentarily.
4. Use extreme care in defining input levels for AC tests. Many inputs may be changed at once, so there will be significant noise at the device pins which may not actually reach V_{IL} or V_{IH} until the noise has settled. AMD recommends using $V_{IL} \leq 0$ V and $V_{IH} \geq 3$ V for AC tests.
5. To simplify failure analysis, programs should be designed to perform DC, Function, and AC tests as three distinct groups of tests.
6. To assist in testing, AMD offers complete documentation on our test procedures and, in most cases, can provide actual Genrad programs, under license from AMD.
7. Capacitive Loading for A.C. Testing

Automatic testers and their associated hardware have stray capacitance which varies from one type of tester to another, but is generally around 50 pF. This, of course, makes it impossible to make direct measurements of parameters which call for a smaller capacitive load than the associated stray capacitance. Typical examples of this are the so-called "float delays" which measure the propagation delays into and out of the high impedance state and are usually specified at a load capacitance of 5.0 pF. In these cases, the test is performed at the higher load capacitance (typically 50 pF) and engineering correlations based on data taken with a bench set up are used to

predict the result at the lower capacitance.

Similarly, a product may be specified at more than one capacitive load. Since the typical automatic tester is not capable of switching loads in mid-test, it is impossible to make measurements at both capacitances even though they may both be greater than the stray capacitance. In these cases, a measurement is made at one of the two capacitances. The result at the other capacitance is predicted from engineering correlations based on data taken with a bench set up and the knowledge that certain D.C. measurements (I_{OH} , I_{OL} , for example) have already been taken and are within specification. In some cases, special D.C. tests are performed in order to facilitate this correlation.

8. Threshold Testing

The noise associated with automatic testing, the long, inductive cables, and the high gain of bipolar devices when in the vicinity of the actual device threshold, frequently give rise to oscillations when testing high-speed speed circuits. These oscillations are not indicative of a reject device, but instead, of an overtaxed test system. To minimize this problem, thresholds are tested at least once for each input pin. Thereafter, "hard" high and low levels are used for other tests. Generally this means that function and A.C. testing are performed at "hard" input levels rather than at V_{IL} max and V_{IH} min.

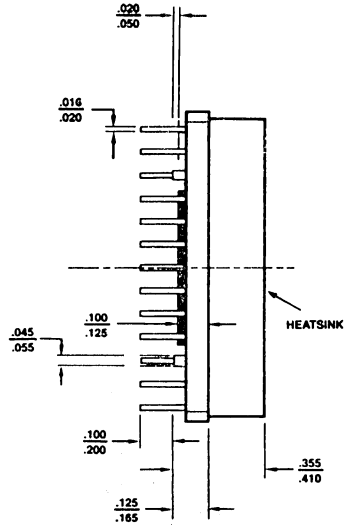
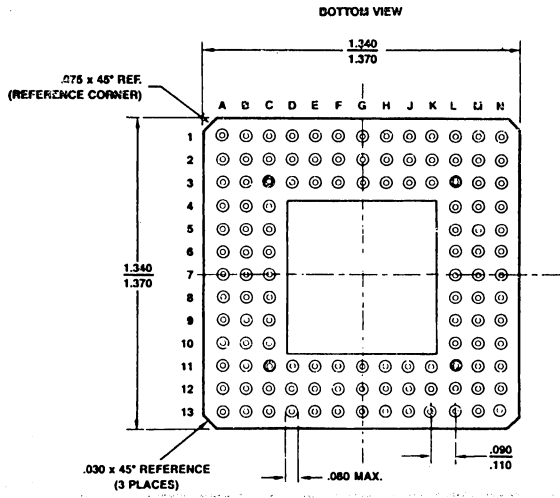
9. A.C. Testing

Occasionally, parameters are specified which cannot be measured directly on automatic testers because of tester limitations. Data input hold times often fall into this category. In these cases, the parameter in question is guaranteed by correlating these tests with other A.C. tests which have been performed. These correlations are arrived at by the cognizant engineer by using data from precise bench measurements in conjunction with the knowledge that certain D.C. parameters have already been measured and are within specification.

In some cases, certain A.C. tests are redundant since they can be shown to be predicted by other tests which have already been performed. In these cases, the redundant tests are not performed.

PHYSICAL DIMENSIONS

CG 120*



PD # 07429A

* Preliminary. Subject to Change.

The International Standard of
Quality guarantees a 0.05% AQL on all
electrical parameters, AC and DC,
over the entire operating range.

INT·STD·500



ADVANCED MICRO DEVICES DOMESTIC SALES OFFICES

ALABAMA	(205) 882-9122	MARYLAND	(301) 796-9310
ARIZONA,		MASSACHUSETTS	(617) 273-3970
Tempe	(602) 242-4400	MINNESOTA	(612) 938-0001
Tucson	(602) 792-1200	NEW JERSEY	(201) 299-0002
CALIFORNIA,		NEW YORK,	
El Segundo	(213) 640-3210	Liverpool	(315) 457-5400
Newport Beach	(714) 752-6262	Poughkeepsie (IBM only)	(914) 471-8180
San Diego	(619) 560-7030	Woodbury	(516) 364-8020
Sunnyvale	(408) 720-8811	NORTH CAROLINA,	
Woodland Hills	(818) 992-4155	Raleigh	(919) 847-8471
COLORADO	(303) 741-2900	OREGON	(503) 245-0080
CONNECTICUT,		OHIO,	
Southbury	(203) 264-7800	Columbus	(614) 891-6455
FLORIDA,		PENNSYLVANIA,	
Altamonte Springs	(305) 339-5022	Allentown (AT&T only)	(215) 398-8006
Clearwater	(813) 530-9971	Willow Grove	(215) 657-3101
Ft. Lauderdale	(305) 484-8600	TEXAS,	
Melbourne	(305) 254-2915	Austin	(512) 346-7830
GEORGIA	(404) 449-7920	Dallas	(214) 934-9099
ILLINOIS	(312) 773-4422	Houston	(713) 785-9001
INDIANA	(317) 244-7207	WASHINGTON	(206) 455-3600
KANSAS	(913) 451-3115	WISCONSIN	(414) 782-7748

INTERNATIONAL SALES OFFICES

BELGIUM,		HONG KONG,	
Bruxelles	TEL:.....(02) 771 99 93	Kowloon	TEL:.....3-695377
	FAX:.....762-3716		FAX:.....1234276
	TLX:.....61028		TLX:.....50426
CANADA, Ontario,		ITALY, Milano	TEL:.....(02) 3390541
Kanata	TEL:.....(613) 592-0090		FAX:.....3498000
Willowdale	TEL:.....(416) 224-5193		TLX:.....315286
	FAX:.....(416) 224-0056	JAPAN, Tokyo	TEL:.....(03) 345-8241
FRANCE,			FAX:.....3425196
Paris	TEL:.....(01) 46 87 36 66		TLX:.....J24064 AMDTKOJ
	FAX:.....(01) 46 86 21 85	LATIN AMERICA,	
	TLX:.....202053F	Ft. Lauderdale,	TEL:.....(305) 484-8600
GERMANY,			FAX:.....(305) 485-9736
Hannover area	TEL:.....(05143) 50 55		TLX:.....5109554261 AMDFTL
	FAX:.....5553	SWEDEN, Stockholm	TEL:.....(08) 733 03 50
	TLX:.....925287		FAX:.....7332285
München	TEL:.....(089) 41 14-0		TLX:.....11602
	FAX:.....406490	UNITED KINGDOM,	
	TLX:.....523883	Manchester area	TEL:.....(0925) 828008
Stuttgart	TEL:.....(0711) 62 33 77		FAX:.....827693
	FAX:.....625187		TLX:.....628524
	TLX:.....721882	London area	TEL:.....(04862) 22121
			FAX:.....22179
			TLX:.....859103

NORTH AMERICAN REPRESENTATIVES

CALIFORNIA		NEW MEXICO	
I ² INC	OEM (408) 988-3400	THORSON DESERT STATES	(505) 293-8555
	DISTI (408) 496-6868	NEW YORK	
IDAHO		NYCOM, INC	(315) 437-8343
INTERMOUNTAIN TECH MKGT	(208) 322-5022	OHIO	
INDIANA		Dayton	
SAI MARKETING CORP	(317) 241-9276	DOLFUSS ROOT & CO	(513) 433-6776
IOWA		Strongsville	
LORENZ SALES	(319) 377-4666	DOLFUSS ROOT & CO	(216) 238-0300
MICHIGAN		PENNSYLVANIA	
SAI MARKETING CORP	(313) 227-1786	DOLFUSS ROOT & CO	(412) 221-4420
NEBRASKA		UTAH	
LORENZ SALES	(402) 475-4660	R ² MARKETING	(801) 595-0631
NEW JERSEY			
TAI CORPORATION	(609) 933-2600		

Advanced Micro Devices reserves the right to make changes in its product without notice in order to improve design or performance characteristics. The performance characteristics listed in this document are guaranteed by specific tests, correlated testing, guard banding, design and other practices common to the industry. For specific testing details, contact your local AMD sales representative. The company assumes no responsibility for the use of any circuits described herein.