Am29C331

CMOS 16-Bit Microprogram Sequencer

PRELIMINARY

DISTINCTIVE CHARACTERISTICS

- **16-Bits Address up to 64K Words**
  Supports 110-ns microcycle time for a 32-bit high-performance system when used with the other members of the Am29C300 Family.
- **Speed Select**
  Supports 80-ns system cycle time.
- **Real-Time Interrupt Support**
  Micro-trap and interrupts are handled transparently at any microinstruction boundary.
- **Built-In Conditional Test Logic**
  Has twelve external test inputs, four of which are used to internally generate an additional four test conditions. 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.

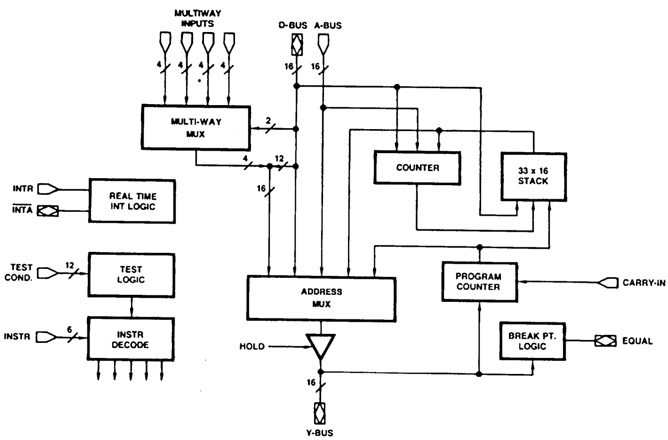
GENERAL DESCRIPTION

The Am29C331 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 Am29C331 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 33-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.

Fabricated using Advanced Micro Devices' 1.6 micron CMOS process, the Am29C331 is powered by a single 5-volt supply. The device is housed in a 120-terminal pin-grid array package.

SIMPLIFIED BLOCK DIAGRAM

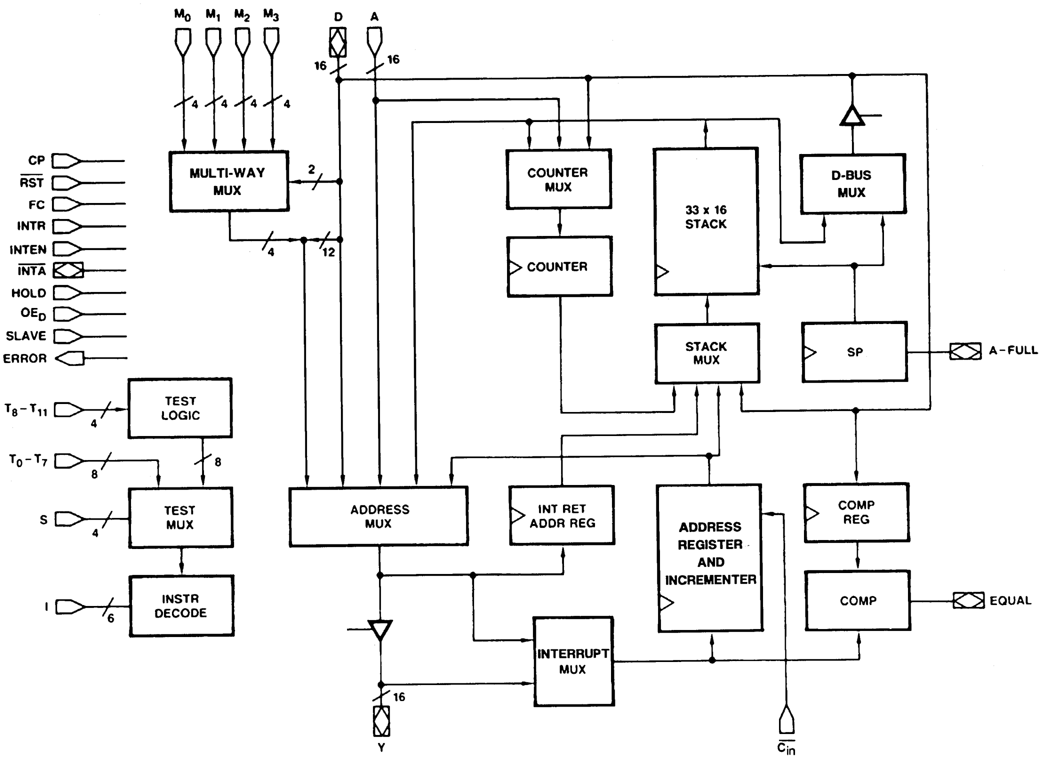


BD006091

Publication # 08823 Rev. B Amendment /0
Issue Date: September 1987

RELATED AMD PRODUCTS

Part No.	Description
Am29114	Vectored Priority Interrupt Controller
Am29116	High-Performance Bipolar 16-Bit Microprocessor
Am29C116	High-Performance CMOS 16-Bit Microprocessor
Am29PL141	Field-Programmable Controller
Am29C323	CMOS 32-Bit Parallel Multiplier
Am29325	32-Bit Floating-Point Processor
Am29C325	CMOS 32-Bit Floating-Point Processor
Am29332	32-Bit Extended Function ALU
Am29C332	CMOS 32-Bit Extended Function ALU
Am29334	64 x 18 Four-Port, Dual-Access Register File
Am29C334	CMOS 64 x 18 Four-Port Dual-Access Register File
Am29337	16-Bit Bounds Checker
Am29338	Byte Queue



BD006102

Figure 1. Am29C331 Detailed Block Diagram

CONNECTION DIAGRAM

120-Lead PGA*

	A	B	C	D	E	F	G	H	J	K	L	M	N
1	M0,0	M1,0	M2,0	M2,1	$\overline{\text{CIN}}$	M1,2	M1,3	M2,3	GND	$\overline{\text{RST}}$	INTR	SLAVE	D15
2	D0	A0	M3,0	M1,1	M0,2	M2,2	M0,3	M3,3	EQUAL	OED	INTEN	HOLD	A15
3	VCC	Y0	D1	M0,1	M3,1	GND	M3,2	VCC	A-FULL	ERROR	$\overline{\text{INTA}}$	Y15	VCC
4	A1	Y1	D2								D14	A14	Y14
5	GND	A2	Y2								D13	A13	GND
6	A3	D3	GND								GND	D12	Y13
7	Y3	D4	A4								A12	Y12	D11
8	D5	Y4	VCC								VCC	Y11	A11
9	GND	A5	Y5								D10	A10	GND
10	D6	A6	Y6								Y10	D9	A9
11	VCC	D7	T3	T6	GND	T10	T11	I0	VCC	I3	Y9	D8	VCC
12	A7	T1	T2	T5	GND	T7	S0	S1	VCC	I2	I4	A8	Y8
13	Y7	T0	T9	T4	GND	T8	CP	S3	VCC	I1	S2	I5	FC

CD010380

*Pins facing up.

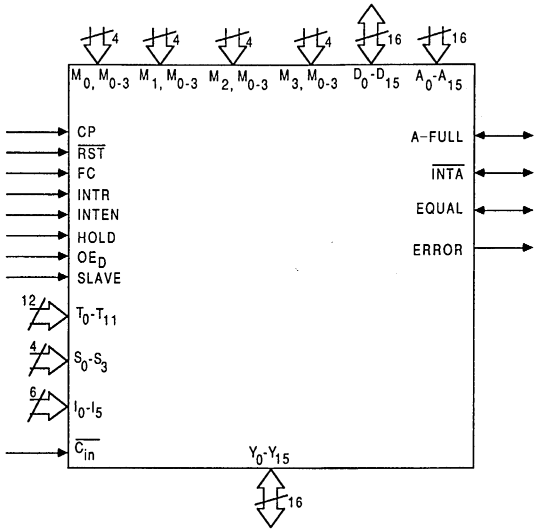
PIN DESIGNATIONS
(Sorted by Pin No.)

PIN NO.	PIN NAME	PAD NO.
A-1	$M_{0,0}$	1
A-2	D_0	120
A-3	V_{CC}	59
A-4	A_1	58
A-5	GND	56
A-6	A_3	114
A-7	Y_3	54
A-8	D_5	51
A-9	GND	50
A-10	D_6	49
A-11	V_{CC}	47
A-12	A_7	106
A-13	Y_7	46
B-1	$M_{1,0}$	61
B-2	A_0	60
B-3	Y_0	119
B-4	Y_1	117
B-5	A_2	116
B-6	D_3	55
B-7	D_4	112
B-8	Y_4	111
B-9	A_5	110
B-10	A_6	108
B-11	D_7	107
B-12	T_1	45
B-13	T_0	105
C-1	$M_{2,0}$	2
C-2	$M_{3,0}$	62
C-3	D_1	118
C-4	D_2	57
C-5	Y_2	115
C-6	GND	113
C-7	A_4	52
C-8	V_{CC}	53
C-9	Y_5	109
C-10	Y_6	48
C-11	T_3	44
C-12	T_2	104
C-13	T_9	41
D-1	$M_{2,1}$	4
D-2	$M_{1,1}$	63
D-3	$M_{0,1}$	3
D-11	T_6	102
D-12	T_5	43
D-13	T_4	103
E-1	$\overline{C_{in}}$	5
E-2	$M_{0,2}$	65
E-3	$M_{3,1}$	64
E-11	GND	97
E-12	GND	98
E-13	GND	99
F-1	$M_{1,2}$	6
F-2	$M_{2,2}$	66
F-3	GND	8
F-11	T_{10}	100
F-12	T_7	42
F-13	T_8	101
G-1	$M_{1,3}$	9
G-2	$M_{0,3}$	67
G-3	$M_{3,2}$	7
G-11	T_{11}	40
G-12	S_0	36
G-13	CP	96
H-1	$M_{2,3}$	69
H-2	$M_{3,3}$	10
H-3	VCC	68
H-11	I_0	34
H-12	S_1	95
H-13	S_3	94
J-1	GND	11
J-2	EQUAL	71
J-3	A–FULL	70
J-11	V_{CC}	37
J-12	V_{CC}	38
J-13	V_{CC}	39
K-1	$\overline{RST}$	13
K-2	OE_D	72
K-3	ERROR	12
K-11	I_3	92
K-12	I_2	33
K-13	I_1	93
L-1	INTR	14
L-2	INTEN	74
L-3	$\overline{INTA}$	73
L-4	D_{14}	18
L-5	D_{13}	79
L-6	GND	23
L-7	A_{12}	22
L-8	V_{CC}	83
L-9	D_{10}	85
L-10	Y_{10}	27
L-11	Y_9	88
L-12	I_4	32
L-13	S_2	35
M-1	SLAVE	75
M-2	HOLD	15
M-3	Y_{15}	77
M-4	A_{14}	78
M-5	A_{13}	80
M-6	D_{12}	81
M-7	Y_{12}	82
M-8	Y_{11}	25
M-9	A_{10}	86
M-10	D_9	87
M-11	D_8	89
M-12	A_8	30
M-13	I_5	91
N-1	D_{15}	16
N-2	A_{15}	76
N-3	V_{CC}	17
N-4	Y_{14}	19
N-5	GND	20
N-6	Y_{13}	21
N-7	D_{11}	24
N-8	A_{11}	84
N-9	GND	26
N-10	A_9	28
N-11	V_{CC}	29
N-12	Y_8	90
N-13	FC	31

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	GND	J-1	11
A_2	B-5	116	D_{15}	N-1	16	$M_{1,1}$	D-2	63	GND	N-5	20
A_3	A-6	114	GND	E-12	97	$M_{1,2}$	F-1	6	GND	A-9	50
A_4	C-7	52	GND	E-13	98	$M_{1,3}$	G-1	9	GND	N-9	26
A_5	B-9	110	GND	E-11	99	$M_{2,0}$	C-1	2	GND	A-5	56
A_6	B-10	108	GND	F-3	8	$M_{2,1}$	D-1	4	V_{CC}	N-3	17
A_7	A-12	106	GND	L-6	23	$M_{2,2}$	F-2	66	V_{CC}	N-11	29
A_8	M-12	30	GND	C-6	113	$M_{2,3}$	H-1	69	V_{CC}	A-3	59
A_9	N-10	28	V_{CC}	J-13	38	$M_{3,0}$	C-2	62	V_{CC}	A-11	47
A_{10}	M-9	86	V_{CC}	H-3	68	$M_{3,1}$	E-3	64	Y_0	B-3	119
A_{11}	N-8	84	V_{CC}	C-8	53	$M_{3,2}$	G-3	7	Y_1	B-4	117
A_{12}	L-7	22	V_{CC}	L-8	83	$M_{3,3}$	H-2	10	Y_2	C-5	115
A_{13}	M-5	80	V_{CC}	J-12	37	$\overline{OE_D}$	K-2	72	Y_3	A-7	54
A_{14}	M-4	78	V_{CC}	J-11	39	$\overline{RST}$	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
$\overline{C_{in}}$	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	$\overline{INTA}$	L-3	73	T_5	D-12	43	Y_{15}	M-3	77

LOGIC SYMBOL



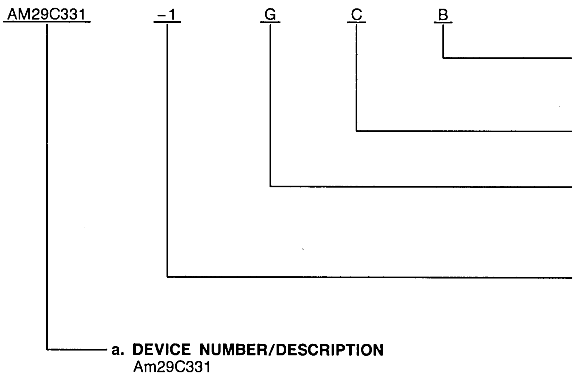
LS002872

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**



e. OPTIONAL PROCESSING
Blank = Standard processing
B = Burn-in

d. TEMPERATURE RANGE
C = Commercial (0 to +70°C)

c. PACKAGE TYPE
G = 120-Lead Pin Grid Array without Heatsink (CGX120)

b. SPEED OPTION
-1 = Speed Select
-2 = Speed Select (TBD)

a. DEVICE NUMBER/DESCRIPTION
Am29C331
CMOS 16-Bit Microprogram Sequencer

Valid Combinations	
AM29C331	GC, GCB
AM29C331-1	

Valid Combinations

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.

MILITARY ORDERING INFORMATION

APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. The order number (Valid Combination) for APL products is formed by a combination of:

- **a. Device Number**
- **b. Speed Option** (if applicable)
- **c. Device Class**
- **d. Package Type**
- **e. Lead Finish**

AM29C331 /B Z C

e. LEAD FINISH
C = Gold

d. PACKAGE TYPE
Z = 120-Lead Pin Grid Array without Heatsink (CGX120)

c. DEVICE CLASS
/B = Class B

b. SPEED OPTION
Not Applicable

a. DEVICE NUMBER/DESCRIPTION
Am29C331
CMOS 16-Bit Microprogram Sequencer

Valid Combinations	
AM29C331	/BZC

Valid Combinations

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 or to check for newly released valid combinations.

Group A Tests

Group A tests consist of Subgroups 1, 2, 3, 7, 8, 9, 10, 11.

PIN DESCRIPTION

$A_0 - A_{15}$ Alternate Data (Input)
Input to address multiplexer and counter.

A-FULL Almost Full (Bidirectional; Three-State)
Indicates that $28 \leqslant SP \leqslant 63$ (meaning there are five or less empty locations left on stack). Also active during stack underflow.

$\overline{C_{in}}$ $\overline{\text{Carry In}}$ (Input, Active LOW)
Carry-in to the incrementer.

CP Clock Pulse (Input)
Clocks sequencer at the LOW-to-HIGH transition.

$D_0 - D_{15}$ Data (Bidirectional, Three-State)
Input to address multiplexer, counter, stack, and comparator register. Output for stack and stack pointer.

EQUAL Equal (Bidirectional, Three-State)
Indicates that the address comparator is enabled and has found a match.

ERROR Error (Output)
Indicates a master/slave error in the slave mode. Indicates a malfunctioning driver or contention of any output in the master mode.

FC Force Continue (Input)
Overrides instruction with CONTINUE.

HOLD Hold (Input)
Stops the sequencer and three-states the outputs.

$I_0 - I_5$ Instruction (Input)
Selects one of 64 instructions.

$\overline{\text{INTA}}$ $\overline{\text{Interrupt Acknowledge}}$ (Bidirectional; Three-State, Active LOW)
Indicates that an interrupt is accepted.

INTEN Interrupt Enable (Input)
Enables interrupts.

INTR Interrupt Request (Input)
Requests the sequencer to interrupt execution.

$M_{0\text{-}3,\ 0\text{-}3}$ Multiway (Input)
Four sets of multiway inputs providing 16-way branches. The first index refers to the set number.

OE_D Output Enable — D-Bus (Input)
Enables the D-bus driver, provided that the sequencer is not in the hold or slave mode.

$\overline{\text{RST}}$ $\overline{\text{Reset}}$ (Input; Active LOW)
Resets the sequencer.

$S_0 - S_3$ Select (Input)
Selects one of 16 test conditions.

SLAVE Slave (Input)
Makes the sequencer a slave.

$T_0 - T_{11}$ Test (Input)
Provides external test inputs.

$Y_0 - Y_{15}$ Address (Bidirectional; Three-State)
Output of microcode address. Input for interrupt address.

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

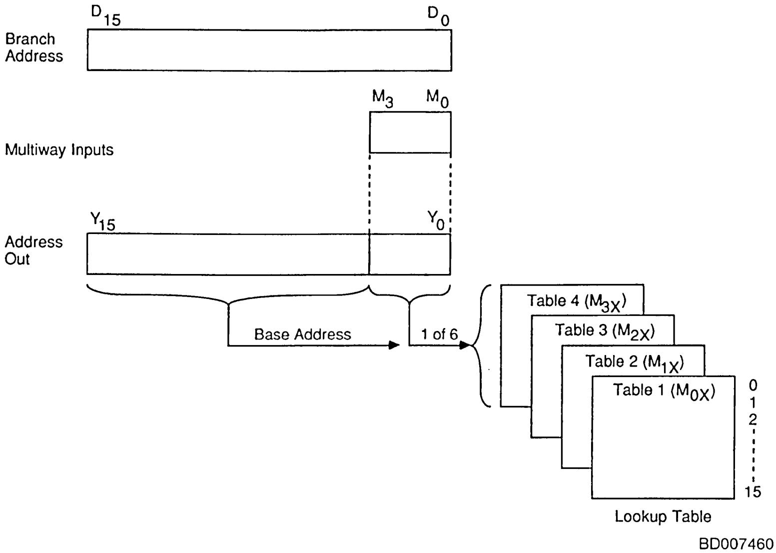
The address multiplexer can select an address from any of five sources:

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 (D_3, D_2, D_1, D_0) with one of the four sets (M_{0X}, M_{1X}, M_{2X}, or M_{3X}) of four-bit multiway-branch addresses. The multiway-branch set is selected by the number D_1D_0, while the bits D_3 and D_2 are "don't cares" (see Figure 2).

D_1	D_0	Multiway Set Selected
0	0	M_{0X}
0	1	M_{1X}
1	0	M_{2X}
1	1	M_{3X}



Notes: 1. D_1 and D_0 select one out of four multiway sets. D_3 and D_2 are "don't cares."
2. Each set of $M_{3X} - M_{0X}$ can select one of sixteen locations. The multiway-branch address is the concatenation of $D_{15} - D_4$ (base address) and $M_{X3} - M_{X0}$.
3. For a given base address, there can be four look-up tables, each sixteen deep.

Figure 2. Multiway Branch

Address Register and Incrementer

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.

In the case of stack overflow, the SP is incremented for every push after overflow. Thus, immediately after the first occurence of stack overflow, the SP will be equal to 34. Subsequent pushes will increment the SP to 35, 36 ... 61, 62, 63, 0, 1, etc. In the case of stack underflow, the SP is decremented for every pop after underflow. Thus, immediately after the first occurrence of stack underflow, the SP will be equal to 63. Subsequent pops will decrement the SP to 62, 61, ... 2, 1, 0, 63, etc.

The contents of the stack pointer are present on the D-bus for all instructions except POP D, provided the driver is turned on. The output signal, A–FULL, is active under the following condition: $28 \leqslant SP \leqslant 63$.

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.

$(S_0 - S_3)_H$	Test	Intended Use
0 – 7	$T_0 - T_7$	General
8	T_8	C (Carry)
9	T_9	N (Negative)
A	T_{10}	V (Overflow)
B	T_{11}	Z (Zero or equal)
C	$T_8 + T_{11}$	C + Z (Unsigned less than or equal, borrow mode)
D	$\overline{T_8} + T_{11}$	$\overline{C}$ + Z (Unsigned less than or equal)
E	$T_9 \oplus T_{10}$	$N \oplus V$ (Signed less than)
F	$(T_9 \oplus T_{10}) + T_{11}$	$(N \oplus V) + Z$ (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. The contents of the stack are invalid after a reset.

TABLE 1. INSTRUCTION SET for $I_5I_4 = 00, 01, 10$

$I_5 - I_0$	Instruction	Cond.: Fail Y	Cond.: Fail Stack	Cond.: Pass Y	Cond.: Pass Stack	Counter	Comp.	D-Mux
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	Pop & Push 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 I_5) XOR I_4
: = Concatination
C = Counter
INC = Output of Incrementer = AR + 1 (if $\overline{C_{in}}$ = LOW)

Note: For unconditional instructions, the action marked under "Cond: Pass" is taken.

TABLE 2. INSTRUCTION SET for $I_5I_4 = 11$

$I_5 - I_0$	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, $\overline{RST}$ = HIGH, and HOLD = LOW). The interrupt-acknowledge output ($\overline{INTA}$) 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 4). 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 5).

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 interrrupt, 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 ignore the current microinstruction, select the trap return address at the address multiplexer, and initiate an interrupt. This will save the trap return address on the stack and issue the trap address from an external source (Figure 6). 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 ($\overline{C_{IN}}$ = 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 7), 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.

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

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 the 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 Am29C331 instructions is given in Figure 3 (3-1, 3-2, 3-3, and 3-4).

```
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 3-1. Loops with Unknown Number of Iterations

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

Figure 3-2. Loop with Known Number of Iterations

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

Figure 3-3. Case Statement (with $D = A_{15} \ldots A_4XX00$ and $M_{0,\,0-3} = A_3I_1I_00$ during the GOTO M instruction. A_1A_0 must be 00, and X signifies a don't care.)

```
               PUSH D C
IF X THEN      IF NOT X THEN GOTO A
 IF Y THEN      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 3-4. Double-Nested If Statement

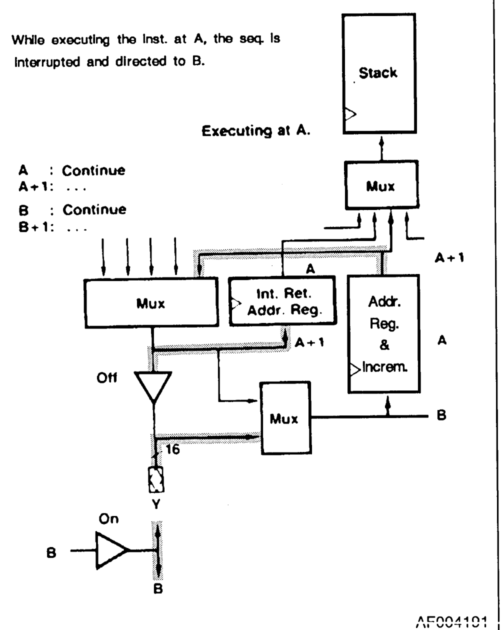


Figure 4. Am29C331 Interrupt Cycle 1

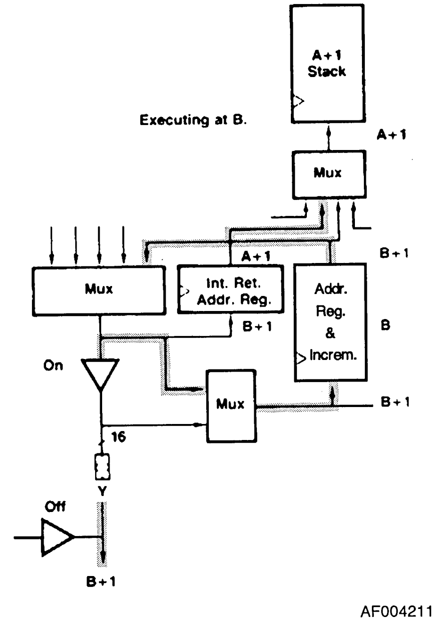


Figure 5. Am29C331 Interrupt Cycle 2

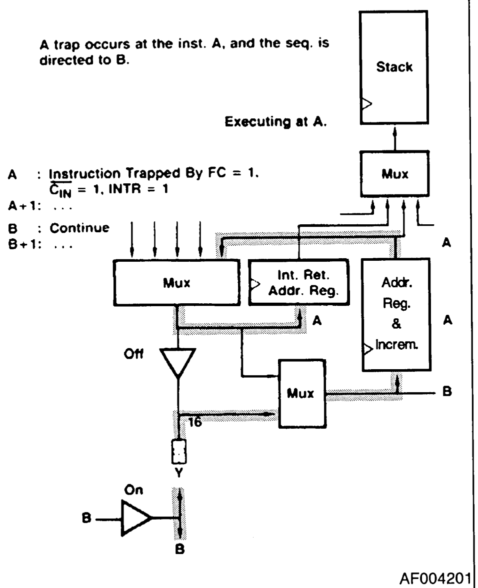


Figure 6. Am29C331 Traps Cycle 1

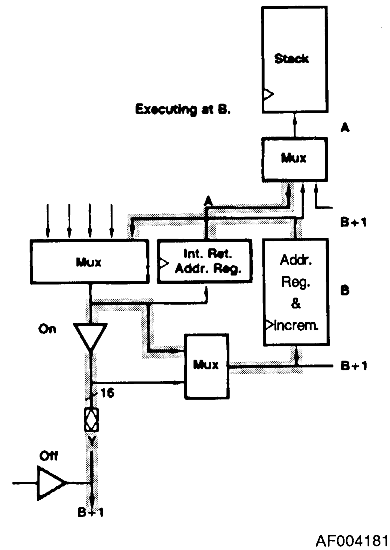
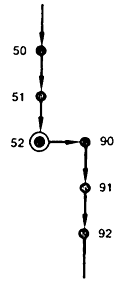
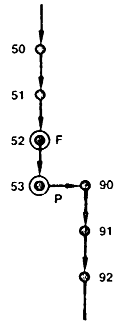


Figure 7. Am29C331 Traps Cycle 2

Instruction Set Definition

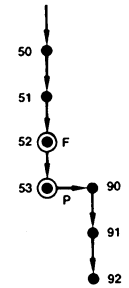
Legend: ● = Other instruction
⊙ = Instruction being described
CC = (Test [$S_3 - S_0$] OR I_5) XOR I_4

P = Test pass
F = Test fail
○ = Register in part

Opcode ($I_5 - I_0$)	Mnemonics	Description	Execution Example
20_H	BRA_D	GOTO D Unconditional branch to the address specified by the D inputs. The D port must be disabled to avoid bus contention.	
24_H	BRA_A	GOTO A Unconditional branch to the address specified by the A inputs.	
28_H	BRA_M	GOTO Multiway ($D_{15} - D_4$ $M_{X3} - M_{X0}$) Unconditional branch to the address specified by the M inputs concatenated with the D input. The lower four bits on the D bus ($D_3 - D_0$) are replaced by one of the four sets of the four-bit multiway branch addresses. The multiway branch set is selected by bits D_1 and D_0 while bits D_3 and D_2 are "don't cares."	
$2C_H$	BRA_S	GOTO TOS Unconditional branch to the address on the top of the stack.	PF001730
00_H	BRCC_D	IF CC THEN GOTO D ELSE CONTINUE If CC is HIGH (pass), branch to the address specified by D. If CC is LOW (fail), continue. The D port must be disabled to avoid bus contention.	
04_H	BRCC_A	IF CC THEN GOTO A ELSE CONTINUE If CC is HIGH (pass), branch to the address specified by A. If CC is LOW (fail), continue.	
08_H	BRCC_M	IF CC THEN GOTO Multiway ($D_{15} - D_4$ $M_{X3} - M_{X0}$) ELSE CONTINUE If CC is HIGH (pass), branch to the address specified by D inputs concatenated with the M inputs. If CC is LOW (fail) continue. The lower four bits on the D bus ($D_3 - D_0$) are replaced by one of the four sets of the 4-bit multiway branch addresses. The multiway branch set is selected by bits D_1 and D_0 while bits D_3 and D_2 are "don't cares."	PF001740
$0C_H$	BRCC_S	IF CC THEN GOTO TOS ELSE POP STACK CONTINUE If CC is HIGH (pass), branch to the address on the top of the stack. If CC is LOW (fail), pop the stack and continue.	

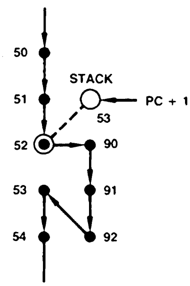
Note: Opcode numbers are in hexadecimal notation.

Opcode $(I_5 - I_0)$	Mnemonics	Description	Execution Example
10_H	BRNC_D	IF NOT CC THEN GOTO D ELSE CONTINUE If CC is LOW (pass), branch to the address specified by D. If CC is HIGH (fail), continue. The D Port must be disabled to avoid Bus contention.	
14_H	BRNC_A	IF NOT CC THEN GOTO A ELSE CONTINUE If CC is LOW (pass), branch to the address specified by A. If CC is HIGH (fail), continue.	
18_H	BRNC_M	IF NOT CC THEN GOTO Multiway $(D_{15} - D_4\ M_{X3} - M_{X0})$ ELSE CONTINUE If CC is LOW (pass), branch to the address specified by D inputs concatenated with the M inputs. If CC is HIGH (fail), continue. The lower four bits on the D bus $(D_3 - D_0)$ are replaced by one of the four sets of the 4-bit multiway branch addresses. The multiway branch set is selected by bits D_1 and D_0 while bits D_3 and D_2 are "don't cares."	
$1C_H$	BRNC_S	IF NOT CC THEN GOTO TOS ELSE POP STACK CONTINUE If CC is LOW (pass), branch to the address on the top of the stack. If CC is HIGH (fail), pop the stack and continue.	



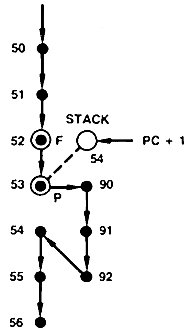
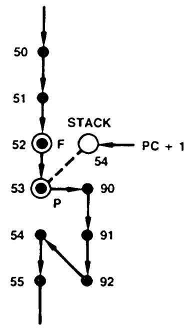
PF001750

Opcode $(I_5 - I_0)$	Mnemonics	Description	Execution Example
21_H	CALL_D	CALL D Unconditional branch to the subroutine specified by the D inputs. Push the return address (address Reg. + 1) on the stack. The D port must be disabled to avoid bus contention.	
25_H	CALL_A	CALL A Unconditional branch to the subroutine specified by the A inputs. Push the return address (Address Reg. + 1) on the stack.	
29_H	CALL_M	CALL Multiway $(D_{15} - D_4\ M_{X3} - M_{X0})$ Unconditional branch to the subroutine specified by the D inputs concatenated with the multiway inputs. Push the return address (Address Reg. + 1) on the stack. The lower four bits on the D bus $(D_3 - D_0)$ are replaced by one of the four sets of the 4-bit multiway branch addresses. The multiway branch set is selected by bits D_1 and D_0 while bits D_3 and D_2 are "don't cares."	
$2D_H$	CALL_S	CALL TOS Unconditional branch to the subroutine specified by the address on the top of the stack. The stack is popped and the return address (Address Reg. + 1) is then pushed onto the stack.	

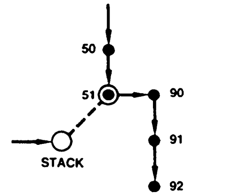
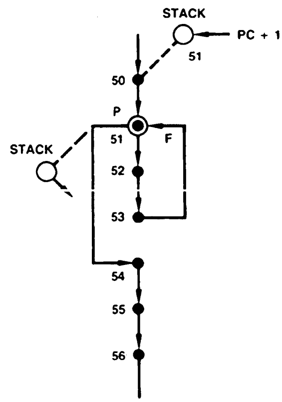


PF001760

Note: Opcode numbers are in hexadecimal notation.

Opcode ($I_5 - I_0$)	Mnemonics	Description	Execution Example
01_H	CCC_D	IF CC, THEN CALL D ELSE CONTINUE If CC is HIGH (pass), call the subroutine specified by the D inputs. Push the return address (Address Reg. + 1) on the stack. If CC is LOW (fail), continue. The D port must be disabled to avoid bus contention.	
05_H	CCC_A	IF CC, THEN CALL A ELSE CONTINUE If CC is HIGH (pass), call the subroutine specified by the A inputs. Push the return address (Address Reg. + 1) on the stack. If CC is LOW (fail), continue.	
09_H	CCC_M	IF CC, THEN CALL Multiway ($D_{15} - D_4$ $M_{X3} - M_{X0}$) ELSE CONTINUE If CC is HIGH (pass), call the subroutine specified by the D inputs concatenated with the M inputs. Push the return address (Address Reg. + 1) on the stack. The lower four bits on the D bus ($D_3 - D_0$) are replaced by one of the four sets of the 4-bit multiway branch addresses. The multiway branch set is selected by bits D_1 and D_0 while bits D_3 and D_2 are "don't cares."	PF001770
$0D_H$	CCC_S	IF CC, THEN CALL TOS ELSE CONTINUE If CC is HIGH (pass), call the subroutine specified by the address on the top of the stack. The stack is popped and the return address (Address Reg. + 1) is pushed onto the stack. If CC is LOW (fail), continue.	
11_H	CNC_D	IF NOT CC, THEN CALL D ELSE CONTINUE If CC is LOW (pass), call the subroutine specified by the D inputs. Push the return address (Address Reg. + 1) on the stack. If CC is HIGH (fail), continue. The D port must be disabled to avoid bus contention.	
15_H	CNC_A	IF NOT CC, THEN CALL A ELSE CONTINUE If CC is LOW (pass), call the subroutine specified by the A inputs. Push the return address (Address Reg. + 1) on the stack. If CC is HIGH (fail), continue.	
19_H	CNC_M	IF NOT CC, THEN CALL Multiway ($D_{15} - D_4$ $M_{X3} - M_{X0}$) ELSE CONTINUE If CC is LOW (pass), call the subroutine specified by the D inputs concatenated with the M inputs. Push the return address (Address Reg. + 1) on the stack. The lower four bits on the D bus ($D_3 - D_0$) are replaced by one of the four sets of the 4-bit multiway branch addresses. The multiway branch set is selected by bits D_1 and D_0 while bits D_3 and D_2 are "don't cares."	PF001780
$1D_H$	CNC_S	IF NOT CC, THEN CALL TOS ELSE CONTINUE If CC is LOW (pass), call the subroutine specified by the address on the top of the stack. The stack is popped and the return address (Address Reg. + 1) is pushed onto the stack.	

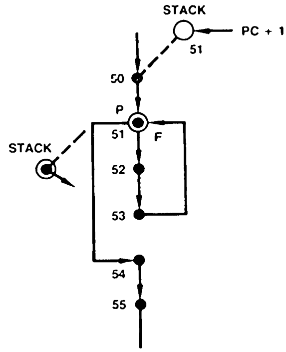
Note: Opcode numbers are in hexadecimal notation.

Opcode ($I_5 - I_0$)	Mnemonics	Description	Execution Example
22_H	EXIT_D	EXIT TO D Unconditional branch to the address specified by the D inputs and pop the stack. The D port must be disabled to avoid bus contention.	
26_H	EXIT_A	EXIT TO A Unconditional branch to the address specified by the A inputs and pop the stack.	
$2A_H$	EXIT_M	EXIT TO Multiway ($D_{15} - D_4$ $M_{X3} - M_{X0}$) Unconditional branch to the address specified by the D inputs concatenated with the M inputs and pop the stack. The lower four bits on the D bus ($D_3 - D_0$) are replaced by one of the four sets of the 4-bit multiway branch addresses. The multiway branch set is selected by bits D_1 and D_0 while D_3 and D_2 are "don't cares."	PF001790
$2E_H$	EXIT_S	EXIT TO TOS Unconditional branch to the address on the top of the stack and pop the stack. Also used for unconditional returns.	
02_H	XTCC_D	IF CC, THEN EXIT TO D ELSE CONTINUE If CC is HIGH (pass), exit to the address specified by the D inputs and pop the stack. If CC is LOW (fail), continue with no pop. The D port must be disabled to avoid bus contention.	
06_H	XTCC_A	IF CC, THEN EXIT TO A ELSE CONTINUE If CC is HIGH (pass), exit to the address specified by the A inputs and pop the stack. If CC is LOW (fail), continue with no pop.	
$0A_H$	XTCC_M	IF CC, THEN EXIT TO Multiway ($D_{15} - D_4$ $M_{X3} - M_{X0}$) ELSE CONTINUE If CC is HIGH (pass), exit to the address specified by the D inputs concatenated with the M inputs and pop the stack. The lower four bits on the D bus ($D_3 - D_0$) are replaced by one of the four sets of the 4-bit multiway branch addresses. The multiway branch set is selected by bits D_1 and D_0 while bits D_3 and D_2 are "don't cares."	PF001800
$0E_H$	XTCC_S	IF CC, THEN EXIT TO TOS ELSE CONTINUE If CC is HIGH (pass), exit to the address on the top of the stack and pop the stack. If CC is LOW (fail), continue with no pop. Also used for conditional returns.	

Note: Opcode numbers are in hexadecimal notation.

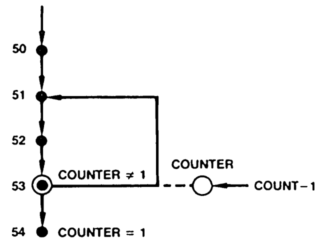
Opcode ($I_5 - I_0$)	Mnemonics	Description
12_H	XTNC_D	IF NOT CC, THEN EXIT TO D ELSE CONTINUE If CC is LOW (pass), exit to the address specified by the D inputs and pop the stack. If CC is HIGH (fail), continue with no pop. The D port must be disabled to avoid bus contention.
16_H	XTNC_A	IF NOT CC, THEN EXIT TO A ELSE CONTINUE If CC is LOW (pass), exit to the address specified by the A inputs and pop the stack. If CC is HIGH (fail), continue with no pop.
$1A_H$	XTNC_M	IF NOT CC, THEN EXIT TO Multiway ($D_{15} - D_4$ $M_{X3} - M_{X0}$) ELSE CONTINUE If CC is LOW (pass), exit to the address specified by the D inputs concatenated with the M inputs and pop the stack. The lower four bits on the D bus ($D_3 - D_0$) are replaced by one of the four sets of the 4-bit multiply branch addresses. The multiway branch set is selected by bits D_1 and D_0 while bits D_3 and D_2 are "don't cares."
$1E_H$	XTNC_S	IF NOT CC, THEN EXIT TO TOS ELSE CONTINUE If CC is LOW (pass), exit to the address on the top of the stack and pop the stack. If CC is HIGH (fail), continue with no pop. Also used for conditional returns.

Execution Example



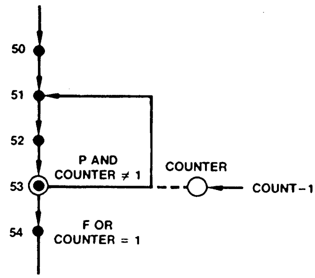
PF001810

Opcode ($I_5 - I_0$)	Mnemonics	Description
23_H	DJMP_D	IF CNT = 1 THEN CNT: = CNT − 1 GOTO D ELSE CNT: = CNT − 1 CONTINUE If the counter is not equal to one, decrement the counter and branch to the address specified by the D inputs. If the counter is equal to one, then decrement the counter and continue. The D port must be disabled to avoid bus contention.
27_H	DJMP_A	IF CNT = 1 THEN CNT: = CNT − 1 GOTO A ELSE CNT: = CNT − 1 CONTINUE If the counter is not equal to one, decrement the counter and branch to the address specified by the A inputs. If the counter is equal to one, then decrement the counter and continue.
$2B_H$	DJMP_M	IF CNT = 1 THEN CNT: = CNT − 1 GOTO Multiway ($D_{15} - D_4$ $M_{X3} - M_{X0}$) ELSE CNT: = CNT − 1 CONTINUE If the counter is not equal to one, decrement the counter and branch to the address specified by the D inputs concatenated with the M inputs. The lower four bits on the D bus ($D_3 - D_0$) are replaced by one of the four sets of the 4-bit multiway branch addresses. The multiway branch set is selected by bits D_1 and D_0 while bits D_3 and D_2 are "don't cares."
$2F_H$	DJMP_S	IF CNT = 1 THEN CNT: = CNT − 1 GOTO TOS ELSE CNT: = CNT − 1 POP STACK CONTINUE If the counter is not equal to one, decrement the counter and branch to the address on the top of the stack. If the counter is equal to one, then decrement the counter, pop the stack and continue.

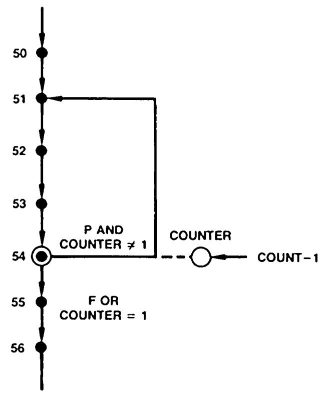
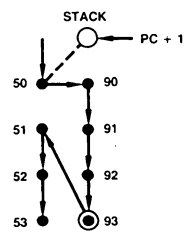


PF001820

Note: Opcode numbers are in hexadecimal notation.

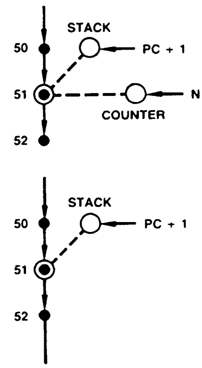
Opcode ($I_5 - I_0$)	Mnemonics	Description	Execution Example
03_H	DJCC_D	IF CC AND CNT = 1 THEN CNT: = CNT − 1 GOTO D ELSE CNT: = CNT − 1 CONTINUE If CC is HIGH (pass) and the counter is not equal to one, decrement the counter and branch to the address specified by the D inputs. If CC is LOW (fail) or the counter is equal to one, then decrement the counter and continue. The D port must be disabled to avoid bus contention.	 PF001830
07_H	DJCC_A	IF CC AND CNT = 1 THEN CNT: = CNT − 1 GOTO A ELSE CNT: = CNT − 1 CONTINUE If CC is HIGH (pass) and the counter is not equal to one, decrement the counter and branch to the address specified by the A inputs. If CC is LOW (fail) or the counter is equal to one, then decrement the counter and continue.	
$0B_H$	DJCC_M	IF CC AND CNT = 1 THEN CNT: = CNT − 1 GOTO Multiway ($D_{15} - D_4$ $M_{X3} - M_{X0}$) ELSE CNT: = CNT − 1 CONTINUE If CC is HIGH (pass) and the counter is not equal to one, decrement the counter and branch to the address specified by the D inputs concatenated with the M inputs. The lower four bits on the D bus ($D_3 - D_0$) are replaced by one of the four sets of the 4-bit multiway branch addresses. The multiway branch set is selected by bits D_1 and D_0 while bits D_3 and D_2 are "don't cares."	
$0F_H$	DJCC_S	IF CC AND CNT = 1 THEN CNT: = CNT − 1 GOTO TOS ELSE CNT: = CNT − 1 POP STACK CONTINUE If CC is HIGH (pass) and the counter is not equal to one, decrement the counter and branch to the address on the top of the stack. If CC is LOW (fail) or the counter is equal to one, then decrement the counter, pop the stack and continue.	

Note: Opcode numbers are in hexadecimal notation.

Opcode ($I_5 - I_0$)	Mnemonics	Description	Execution Example
13_H	DJNCC_D	IF NOT CC AND CNT = 1 THEN CNT: = CNT − 1 GOTO D ELSE CNT: = CNT − 1 CONTINUE If CC is LOW (pass) and the counter is not equal to one, decrement the counter and branch to the address specified by the D inputs. If CC is HIGH (fail) or the counter is equal to one, then decrement the counter and continue. The D port must be disabled to avoid bus contention.	 PF001840
17_H	DJNCC_A	IF NOT CC AND CNT = 1 THEN CNT: = CNT − 1 GOTO A ELSE CNT: = CNT − 1 CONTINUE If CC is LOW (pass) and the counter is not equal to one, decrement the counter and branch to the address specified by the A inputs. The content of the interrupt return address register and the address register is replaced by the A address in this case. If CC is HIGH (fail) or the counter is equal to one, the current address is incremented, appears on the bus for continue, and is stored into the above two registers.	
$1B_H$	DJNCC_M	IF NOT CC AND CNT = 1 THEN CNT: = CNT − 1 GOTO Multiway ($D_{15} - D_4$ $M_3 - M_0$) ELSE CONTINUE If CC is LOW (pass) and the counter is not equal to one, decrement the counter and branch to the address specified by the D inputs concatenated with the M inputs. The lower four bits on the D bus ($D_3 - D_0$) are replaced by one of the four sets of the 4-bit multiway branch addresses. The multiway branch set is selected by bits D_1 and D_0 while bits D_3 and D_2 are "don't cares."	
$1F_H$	DJNCC_S	IF NOT CC AND CNT = 1 THEN CNT: = CNT − 1 GOTO TOS ELSE CNT: = CNT − 1 POP STACK CONTINUE If CC is LOW (pass) and the counter is not equal to one, decrement the counter and branch to the address on the top of the stack. If CC is HIGH (fail) or the counter is equal to one, then decrement the counter, pop the stack and continue.	
$2E_H$	RET	RETURN Unconditional return from subroutine. The return address is popped from the stack.	 PF001850
$0E_H$	RETCC	IF CC THEN RETURN ELSE CONTINUE If CC is HIGH (pass), return from subroutine. The return address is popped from the stack. If CC is LOW (fail), continue.	
$1E_H$	RETNC	IF NOT CC THEN RETURN ELSE CONTINUE If CC is LOW (pass), return from subroutine. The return address is popped from the stack. If CC is HIGH (fail), continue.	

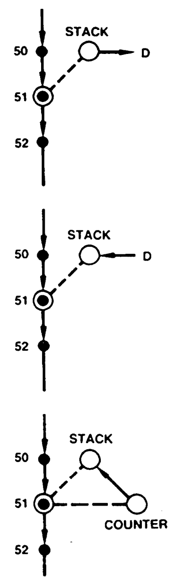
Note: Opcode numbers are in hexadecimal notation.

Opcode $(I_5 - I_0)$	Mnemonics	Description	Execution Example
31_H	FOR_D	INITIALIZE LOOP Push the Address Reg. + 1 on the stack, load the counter from the D inputs and continue. Use with DJUMP_S for FOR . . . NEXT loops. The D port must be disabled to avoid bus contention.	
37_H	FOR_A	INITIALIZE LOOP Push the Address Reg. + 1 on the stack, load the counter from the A inputs and continue. Use with DJUMP_S for FOR . . . NEXT loops.	
33_H	LOOP	INITIALIZE LOOP Push the Address Reg. + 1 on the stack and continue. Use with BRCC_S for REPEAT . . . UNTIL loops, or with XTCC_D and BRA_S for WHILE . . . END WHILE loops.	



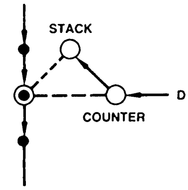
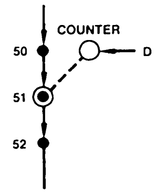
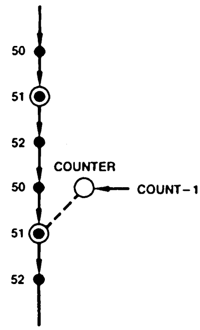
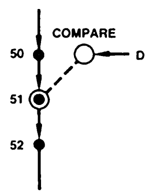
PF001860

Opcode $(I_5 - I_0)$	Mnemonics	Description	Execution Example
34_H	POP_D	Pop the stack and output the value on the D outputs and continue. The D port must be enabled.	
38_H	POP_C	Pop the stack and store the value in the counter and continue.	
35_H	PUSH_D	Push the D inputs on the stack and continue. The D port must be disabled to avoid bus contention.	
39_H	PUSH_C	Push the counter on the stack and continue.	
$3A_H$	SWAP	Exchange the counter and the top of stack and continue.	



PF001870

Note: Opcode numbers are in hexadecimal notation.

Opcode ($I_5 - I_0$)	Mnemonics	Description	Execution Example
$3B_H$	STACK__C	Push the counter on the stack and load the counter with the value of the D inputs and continue.	
$3C_H$	LOAD__D	Load the counter with the value of the D inputs and continue. The D port must be disabled to avoid bus contention.	
$3D_H$	LOAD__A	Load the counter with the value of the A inputs and continue.	PF001880
30_H	CONT	Continue.	
32_H	DECR	Decrement the counter and continue.	
36_H	RESET__SP	Reset the stack pointer and continue.	PF001890
$3E_H$	SET	Load the comparison register with the value of the D inputs, enable the comparator and continue.	
$3F_H$	CLEAR	Disable the comparator and continue.	PF001900

Note: Opcode numbers are in hexadecimal notation.

APPLICATIONS

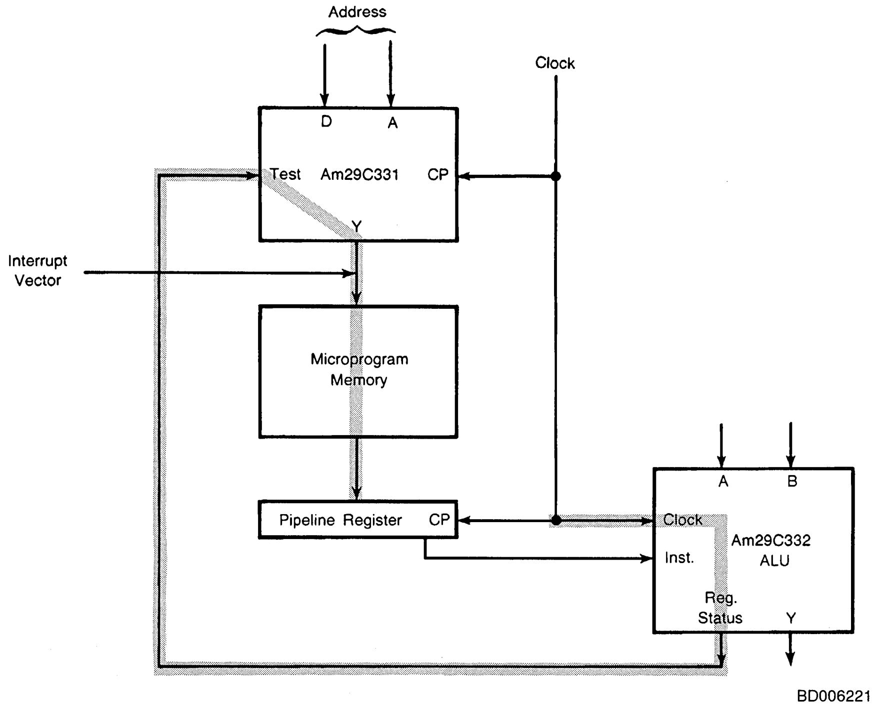


Figure 8. Typical Control-Path Architecture For Am29C300 Family

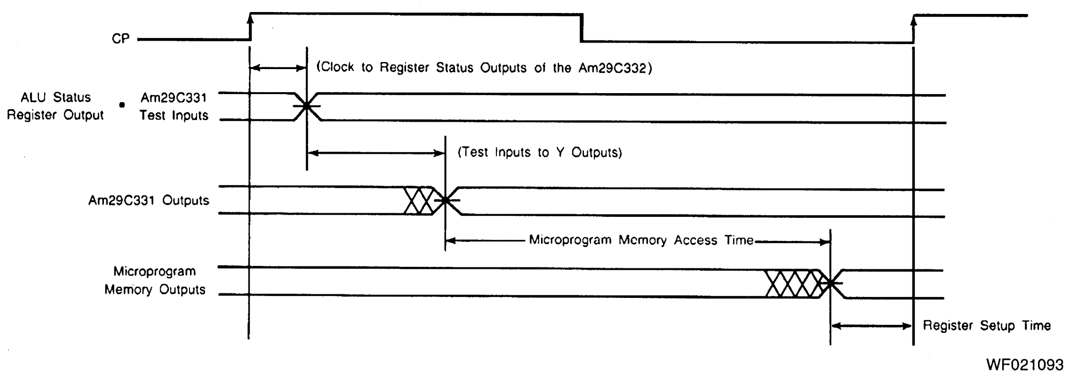


Figure 9. Cycle Timing Waveform*

*This waveform shows the timing relationship for the configuration shown in Figure 8.

ABSOLUTE MAXIMUM RATINGS

Storage Temperature .... −65 to +150°C
(Case) Temperature Under Bias .... −55 to +125°C
Supply Voltage to
Ground Potential Continuous .... −0.3 V to +7.0 V
DC Voltage Applied to Outputs For
High Output State .... −0.3 V to $+V_{CC}$ +0.3 V
DC Input Voltage .... −0.3 V to $+V_{CC}$ +0.3 V
DC Output Current, Into LOW Outputs .... 30 mA
DC Input Current .... −10 mA to +10 mA

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 V to +5.25 V

Military* (M) Devices
Temperature (T_A) .... −55 to +125°C
Supply Voltage (V_{CC}) .... +4.5 V to +5.5 V

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

*Military Product 100% tested at T_A = +25°C, +125°C, and −55°C.

DC CHARACTERISTICS over operating range unless otherwise specified (for APL Products, Group A, Subgroups 1, 2, 3 are tested unless otherwise noted)

Parameter Symbol	Parameter Description	Test Conditions (Note 1)			Min.	Max.	Unit
V_{OH}	Output HIGH Voltage	V_{CC} = Min. $V_{IN} = V_{IH}$ or V_{IL}	I_{OH} = 0.4 mA		2.4		Volts
V_{OL}	Output LOW Voltage	V_{CC} = Min. $V_{IN} = V_{IH}$ or V_{IL}	I_{OL} = 8 mA for Y-BUS = 4 mA for All Other Pins			0.5	Volts
V_{IH}	Guaranteed Input Logical HIGH Voltage (Note 2)				2.0		Volts
V_{IL}	Guaranteed Input Logical LOW Voltage (Note 2)					0.8	Volts
I_{IL}	Input LOW Current	V_{CC} = Max. V_{IN} = 0.5 Volts				−10	μA
I_{IH}	Input HIGH Current	V_{CC} = Max. $V_{IN} = V_{CC} - 0.5$ V				10	μA
I_{OZH}	Off-State (HIGH Impedance) Output Current	V_{CC} = Max. V_O = 2.4 Volts				10	μA
I_{OZL}	Off-State (HIGH Impedance) Output Current	V_{CC} = Max. V_O = 0.5 Volts				−10	μA
I_{CC}	Static Power Supply Current (Note 3)	V_{CC} = Max., $V_{IN} = V_{CC}$ or GND, I_O = 0 μA	COM'L	29C331		40	mA
				29C331-1/-2		50	
			MIL	29C331 only		50	
C_{PD}	Power Dissipation Capacitance (Note 4)	V_{CC} = 5.0 V T_A = 25°C No Load				pF Typical	

Notes: 1. V_{CC} conditions shown as Min. or Max. refer to the commercial and military V_{CC} limits.
2. These input levels provide zero-noise immunity and should only be statically tested in a noise-free environment (not functionally tested).
3. Worst-case I_{CC} is measured at the lowest temperature in the specified operating range.
4. C_{PD} determines the no-load dynamic current consumption:
I_{CC} (Total) = I_{CC} (Static) + C_{PD} V_{CC} f, where f is the switching frequency of the majority of the internal nodes, normally one-half of the clock frequency. This specification is not tested.

SWITCHING CHARACTERISTICS over COMMERCIAL operating range

A. COMBINATIONAL PROPAGATION DELAYS

No.	From	To	29C331 Max. Delay	29C331-1 Max. Delay	29C331-2 Max. Delay	Unit
1	D_{15-0}	Y_{15-0}	22*	20*	18	ns
	D_{15-0}	EQUAL	32	28	23	ns
	D_{15-0}	ERROR	36	32	26	ns
2	A_{15-0}	Y_{15-0}	20	18	16	ns
	A_{15-0}	EQUAL	31	27	22	ns
	A_{15-0}	ERROR	33	29	24	ns
3	M_{X3-X0}	Y_{15-0}	19	18	16	ns
	M_{X3-X0}	EQUAL	29	26	21	ns
	M_{X3-X0}	ERROR	33	29	24	ns
	Y_{15-0}	EQUAL	31	28	23	ns
	Y_{15-0}	ERROR	26	23	19	ns
4	I_{5-0}	Y_{31-0}	24	22	18	ns
5	I_{5-0}	D_{15-0}	29	26	21	ns
	I_{5-0}	EQUAL	36	33	27	ns
	I_{5-0}	ERROR	40	35	28	ns
6	T_{11-0}	Y_{15-0}	24	22	18	ns
	T_{11-0}	EQUAL	35	32	26	ns
	T_{11-0}	ERROR	37	33	27	ns
	S_{3-0}	Y_{15-0}	24	22	18	ns
	S_{3-0}	EQUAL	35	32	26	ns
	S_{3-0}	ERROR	37	33	27	ns
7	CP	Y_{15-0}	28	25	20	ns
8	CP	D_{15-0}	27/Z	25/Z	20/Z	ns
9	CP	A-FULL	27	24	20	ns
	CP	EQUAL	36	32	26	ns
	CP	ERROR	50	45	36	ns
10	$\overline{RST}$	Y_{15-0}	26/Z	24/Z	20/Z	ns
	$\overline{RST}$	D_{15-0}	Z	Z	Z	ns
11	$\overline{RST}$	$\overline{INTA}$	22	19	17	ns
	$\overline{RST}$	EQUAL	35	31	25	ns
	$\overline{RST}$	ERROR	38	34	28	ns
12	FC	Y_{15-0}	24	22	18	ns
13	FC	D_{15-0}	28	25	20	ns
	FC	EQUAL	33	30	24	ns
	FC	ERROR	35	31	25	ns
	INTR	Y_{15-0}	Z	Z	Z	ns
14	INTR	$\overline{INTA}$	17	16	9	ns
	INTR	EQUAL	(Note 1)	(Note 1)	(Note 1)	ns
	INTR	ERROR	46	21	18	ns
	INTEN	Y_{15-0}	Z	Z	Z	ns
15	INTEN	$\overline{INTA}$	16	15	9	ns
	INTEN	EQUAL	(Note 1)	(Note 1)	(Note 1)	ns
	INTEN	ERROR	46	21	18	ns
	HOLD	Y_{15-0}	Z	Z	Z	ns
	HOLD	INTA	Z	Z	Z	ns
	HOLD	A-FULL	Z	Z	Z	ns
	HOLD	EQUAL	34/Z	31/Z	17/Z	ns
	HOLD	ERROR	46	18	17	ns
	OED	D_{15-0}	Z	17	Z	ns
	OED	ERROR	19	Z	17	ns
	INTA	ERROR	19**	17**	17	ns
	A-FULL	ERROR	21**	20**	17	ns
	EQUAL	ERROR	19**	17**	17	ns
16	$\overline{C_{in}}$	Y_{15-0}	24	21	18	ns
	$\overline{C_{in}}$	EQUAL	36	33	20	ns
	$\overline{C_{in}}$	ERROR	37	33	21	ns
	SLAVE	Y_{15-0}	Z	Z	Z	ns
	SLAVE	D_{15-0}	Z	Z	Z	ns
	SLAVE	$\overline{INTA}$	Z	Z	Z	ns
	SLAVE	A-FULL	Z	Z	Z	ns
	SLAVE	EQUAL	Z	Z	Z	ns

Notes: See notes following Table D.

*This includes using D as select lines for multiway sets.
**In the slave mode.

SWITCHING CHARACTERISTICS over COMMERCIAL operating range (Cont'd.)

B. OUTPUT DISABLE TIME

No.	From	To	Description	29C331 Max. Value	29C331-1 Max. Value	29C331-2 Max. Value	Unit
	$\overline{RST}$	Y_{15-0}	Reset-to-Address Enable	29	25	25	ns
	$\overline{RST}$	Y_{15-0}	Reset-to-Address Disable	29	25	25	ns
43	INTR	Y_{15-0}	INTR-to-Address Enable	24	21	21	ns
44	INTR	Y_{15-0}	INTR-to-Address Disable	24	21	21	ns
	INTEN	Y_{15-0}	INTEN-to-Address Enable	24	21	21	ns
	INTEN	Y_{15-0}	INTEN-to-Address Disable	24	21	21	ns
	HOLD	Y_{15-0}	HOLD-to-Address Enable	23	20	20	ns
	HOLD	Y_{15-0}	HOLD-to-Address Disable	23	20	20	ns
	SLAVE	Y_{15-0}	SLAVE-to-Address Enable	24	21	21	ns
	SLAVE	Y_{15-0}	SLAVE-to-Address Disable	24	21	21	ns
	OED	Y_{15-0}	OED-to-Data Enable	26	22	22	ns
	OED	D_{15-0}	OED-to-Data Disable	26	22	22	ns
	$\overline{RST}$	D_{15-0}	Reset-to-Data Enable	27	23	23	ns
	$\overline{RST}$	D_{15-0}	Reset-to-Data Disable	27	23	23	ns
	SLAVE	D_{15-0}	SLAVE-to-Data Enable	26	22	22	ns
	SLAVE	D_{15-0}	SLAVE-to-Data Disable	26	22	22	ns
	CP	D_{15-0}	Clock-to-Data Enable	35	24	24	ns
	CP	D_{15-0}	Clock-to-Data Disable	35	24	24	ns
	HOLD	$\overline{INTA}$	HOLD-to-$\overline{INTA}$ Enable	22	19	19	ns
	HOLD	$\overline{INTA}$	HOLD-to-$\overline{INTA}$ Disable	22	19	19	ns
	HOLD	A-FULL	HOLD-to-A-FULL Enable	21	18	18	ns
	HOLD	A-FULL	HOLD-to-A-FULL Disable	21	18	18	ns
	HOLD	EQUAL	HOLD-to-EQUAL Enable	21	18	18	ns
	HOLD	EQUAL	HOLD-to-EQUAL Disable	21	18	18	ns
	SLAVE	$\overline{INTA}$	SLAVE-to-$\overline{INTA}$ Enable	22	19	19	ns
	SLAVE	$\overline{INTA}$	SLAVE-to-$\overline{INTA}$ Disable	22	19	19	ns
	SLAVE	A-FULL	SLAVE-to-A-FULL Enable	22	19	19	ns
	SLAVE	A-FULL	SLAVE-to-A-FULL Disable	22	19	19	ns
	SLAVE	EQUAL	SLAVE-to-EQUAL Enable	22	19	19	ns
	SLAVE	EQUAL	SLAVE-to-EQUAL Disable	22	19	19	ns

Notes: See notes following Table D.

SWITCHING CHARACTERISTICS over COMMERCIAL over operating range (Cont'd.)

C. SETUP AND HOLD TIMES

No.	Parameter	For	With Respect To	29C331 Max. Value	29C331-1 Max. Value	29C331-2 Max. Value	Unit
17	Data Setup	D_{15-0}	CP ↑	21	19	19	ns
18	Data Hold	D_{15-0}	CP ↑	0	0	0	ns
19	Alternate Data Setup	A_{15-0}	CP ↑	23	21	21	ns
20	Alternate Data Hold	A_{15-0}	CP ↑	0	0	0	ns
21	Multiway Setup	M_{X3-X0}	CP ↑	23	21	21	ns
22	Multiway Hold	M_{X3-X0}	CP ↑	0	0	0	ns
23	Address Setup	Y_{15-0}	CP ↑	18	17	17	ns
24	Address Hold	Y_{15-0}	CP ↑	0	0	0	ns
25	Instruction Setup	I_{5-0}	CP ↑	24	21	21	ns
26	Instruction Hold	I_{5-0}	CP ↑	0	0	0	ns
27	Forced Continue Setup	FC	CP ↑	21	19	19	ns
28	Forced Continue Hold	FC	CP ↑	0	0	0	ns
29	Test Setup	T_{11-0}	CP ↑	21	20	20	ns
30	Test Hold	T_{11-0}	CP ↑	0	0	0	ns
31	Select Setup	S_{3-0}	CP ↑	22	20	20	ns
32	Select Hold	S_{3-0}	CP ↑	0	0	0	ns
33	Reset Setup	$\overline{RST}$	CP ↑	22	20	20	ns
34	Reset Hold	$\overline{RST}$	CP ↑	0	0	0	ns
35	Interrupt Request Setup	INTR	CP ↑	20	18	18	ns
36	Interrupt Request Hold	INTR	CP ↑	0	0	0	ns
37	Interrupt Enable Setup	INTEN	CP ↑	18	16	16	ns
38	Interrupt Enable Hold	INTEN	CP ↑	0	0	0	ns
39	Hold Mode Setup	HOLD	CP ↑	21	19	19	ns
40	Hold Mode Hold	HOLD	CP ↑	0	0	0	ns
41	Carry-In Setup	$\overline{C_{in}}$	CP ↑	22	20	20	ns
42	Carry-In Hold	$\overline{C_{in}}$	CP ↑	0	0	0	ns

D. MINIMUM CLOCK REQUIREMENT

No.	Description	29C331 Max. Value	29C331-1 Max. Value	29C331-2 Max. Value	Unit
53	Minimum Clock LOW Time	23	22	22	ns
54	Minimum Clock HIGH Time	19	16	16	ns

Notes: 1. (INTR, INTEN)-to-EQUAL is the sum of (INTR, INTEN)-to-Y disable time and Y-to-EQUAL delay time.

2. $C_L = 50$ pF; $C_L = 5$ pF for Disable Time only.

3. The status of $I_5 - I_0$ and FC must not be changed during the clock LOW time.

SWITCHING CHARACTERISTICS over **MILITARY** operating range (for APL Products, Group A, Subgroups 9, 10, 11 are tested unless otherwise noted)

A. COMBINATIONAL PROPAGATION DELAYS

No.	From	To	29C331 Max. Delay	Unit
1	D_{15-0}	Y_{15-0}	30*	ns
	D_{15-0}	EQUAL	48	ns
	D_{15-0}	ERROR	29**	ns
2	A_{15-0}	Y_{15-0}	27	ns
	A_{15-0}	EQUAL	44	ns
	A_{15-0}	ERROR	50	ns
3	M_{X3-X0}	Y_{15-0}	30	ns
	M_{X3-X0}	EQUAL	48	ns
	M_{X3-X0}	ERROR	55	ns
	Y_{15-0}	EQUAL	41	ns
	Y_{15-0}	ERROR	29**	ns
4	I_{5-0}	Y_{31-0}	32	ns
5	I_{5-0}	D_{15-0}	37	ns
	I_{5-0}	EQUAL	48	ns
	I_{5-0}	ERROR	55	ns
6	T_{11-0}	Y_{15-0}	32	ns
	T_{11-0}	EQUAL	48	ns
	T_{11-0}	ERROR	55	ns
	S_{3-0}	Y_{15-0}	32	ns
	S_{3-0}	EQUAL	48	ns
	S_{3-0}	ERROR	55	ns
7	CP	Y_{15-0}	37	ns
8	CP	D_{15-0}	37/Z	ns
9	CP	A-FULL	32	ns
	CP	EQUAL	54	ns
	CP	ERROR	60	ns
10	$\overline{RST}$	Y_{15-0}	32/Z	ns
	$\overline{RST}$	D_{15-0}	Z	ns
11	$\overline{RST}$	$\overline{INTA}$	22	ns
	$\overline{RST}$	EQUAL	48	ns
	$\overline{RST}$	ERROR	55	ns
12	FC	Y_{15-0}	32	ns
13	FC	D_{15-0}	37	ns
	FC	EQUAL	48	ns
	FC	ERROR	55	ns
	INTR	Y_{15-0}	Z	ns
14	INTR	$\overline{INTA}$	21	ns
	INTR	EQUAL	(Note 1)	ns
	INTR	ERROR	49	ns
	INTEN	Y_{15-0}	Z	ns
15	INTEN	$\overline{INTA}$	21	ns
	INTEN	EQUAL	(Note 1)	ns
	INTEN	ERROR	49	ns
	HOLD	Y_{15-0}	Z	ns
	HOLD	INTA	Z	ns
	HOLD	A-FULL	21/Z	ns
	HOLD	EQUAL	43/Z	ns
	HOLD	ERROR	49	ns
	OED	D_{15-0}	26	ns
	OED	ERROR	Z	ns
	INTA	ERROR	29**	ns
	A-FULL	ERROR	29**	ns
	EQUAL	ERROR	29**	ns
16	$\overline{C}_{in}$	Y_{15-0}	32	ns
	$\overline{C}_{in}$	EQUAL	48	ns
	$\overline{C}_{in}$	ERROR	55	ns
	SLAVE	Y_{15-0}	Z	ns
	SLAVE	D_{15-0}	Z	ns
	SLAVE	$\overline{INTA}$	Z	ns
	SLAVE	A-FULL	Z	ns
	SLAVE	EQUAL	Z	ns

Notes: See notes following Table D.

*This includes using D as select lines for multiway sets.
**In the slave mode.

SWITCHING CHARACTERISTICS over MILITARY operating range (Cont'd.)

B. OUTPUT DISABLE TIME

No.	From	To	Description	29C331 Max. Value	Unit
	$\overline{RST}$	Y_{15-0}	Reset-to-Address Enable	26	ns
	$\overline{RST}$	Y_{15-0}	Reset-to-Address Disable	26	ns
43	INTR	Y_{15-0}	INTR-to-Address Enable	26	ns
44	INTR	Y_{15-0}	INTR-to-Address Disable	26	ns
	INTEN	Y_{15-0}	INTEN-to-Address Enable	26	ns
	INTEN	Y_{15-0}	INTEN-to-Address Disable	26	ns
	HOLD	Y_{15-0}	HOLD-to-Address Enable	26	ns
	HOLD	Y_{15-0}	HOLD-to-Address Disable	26	ns
	SLAVE	Y_{15-0}	SLAVE-to-Address Enable	26	ns
	SLAVE	Y_{15-0}	SLAVE-to-Address Disable	26	ns
	OED	Y_{15-0}	OED-to-Data Enable	26	ns
	$\overline{OED}$	D_{15-0}	OED-to-Data Disable	26	ns
	$\overline{RST}$	D_{15-0}	Reset-to-Data Enable	26	ns
	$\overline{RST}$	D_{15-0}	Reset-to-Data Disable	26	ns
	SLAVE	D_{15-0}	SLAVE-to-Data Enable	26	ns
	SLAVE	D_{15-0}	SLAVE-to-Data Disable	26	ns
	CP	D_{15-0}	Clock-to-Data Enable	23	ns
	CP	D_{15-0}	Clock-to-Data Disable	23	ns
	HOLD	$\overline{INTA}$	HOLD-to-$\overline{INTA}$ Enable	21	ns
	HOLD	$\overline{INTA}$	HOLD-to-$\overline{INTA}$ Disable	21	ns
	HOLD	A-FULL	HOLD-to-A-FULL Enable	21	ns
	HOLD	A-FULL	HOLD-to-A-FULL Disable	21	ns
	HOLD	EQUAL	HOLD-to-EQUAL Enable	21	ns
	HOLD	EQUAL	HOLD-to-EQUAL Disable	21	ns
	SLAVE	$\overline{INTA}$	SLAVE-to-$\overline{INTA}$ Enable	21	ns
	SLAVE	$\overline{INTA}$	SLAVE-to-$\overline{INTA}$ Disable	21	ns
	SLAVE	A-FULL	SLAVE-to-A-FULL Enable	21	ns
	SLAVE	A-FULL	SLAVE-to-A-FULL Disable	21	ns
	SLAVE	EQUAL	SLAVE-to-EQUAL Enable	21	ns
	SLAVE	EQUAL	SLAVE-to-EQUAL Disable	21	ns

Notes: See notes following Table D.

SWITCHING CHARACTERISTICS over MILITARY operating range (Cont'd.)

C. SETUP AND HOLD TIMES

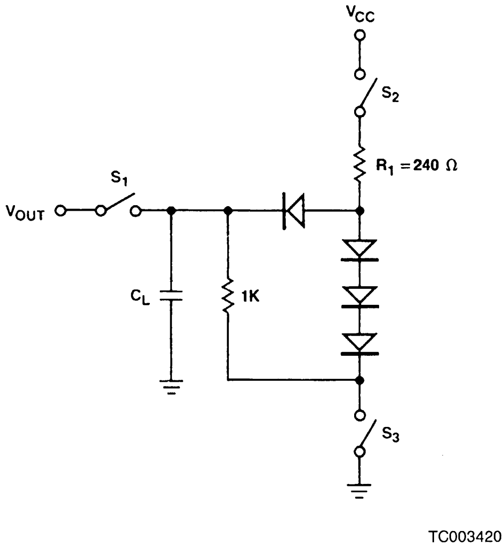
No.	Parameter	For	With Respect To	29C331 Max. Value	Unit
17	Data Setup	D_{15-0}	CP ↑	32	ns
18	Data Hold	D_{15-0}	CP ↑	1	ns
19	Alternate Data Setup	A_{15-0}	CP ↑	32	ns
20	Alternate Data Hold	A_{15-0}	CP ↑	1	ns
21	Multiway Setup	M_{X3-X0}	CP ↑	32	ns
22	Multiway Hold	M_{X3-X0}	CP ↑	1	ns
23	Address Setup	Y_{15-0}	CP ↑	27	ns
24	Address Hold	Y_{15-0}	CP ↑	2	ns
25	Instruction Setup	I_{5-0}	CP ↑	32	ns
26	Instruction Hold	I_{5-0}	CP ↑	0	ns
27	Forced Continue Setup	FC	CP ↑	32	ns
28	Forced Continue Hold	FC	CP ↑	1	ns
29	Test Setup	T_{11-0}	CP ↑	32	ns
30	Test Hold	T_{11-0}	CP ↑	0	ns
31	Select Setup	S_{3-0}	CP ↑	32	ns
32	Select Hold	S_{3-0}	CP ↑	0	ns
33	Reset Setup	$\overline{RST}$	CP ↑	32	ns
34	Reset Hold	$\overline{RST}$	CP ↑	1	ns
35	Interrupt Request Setup	INTR	CP ↑	27	ns
36	Interrupt Request Hold	INTR	CP ↑	1	ns
37	Interrupt Enable Setup	INTEN	CP ↑	27	ns
38	Interrupt Enable Hold	INTEN	CP ↑	1	ns
39	Hold Mode Setup	HOLD	CP ↑	27	ns
40	Hold Mode Hold	$\overline{HOLD}$	CP ↑	1	ns
41	Carry-In Setup	$\overline{C_{in}}$	CP ↑	30	ns
42	Carry-In Hold	$\overline{C_{in}}$	CP ↑	1	ns

D. MINIMUM CLOCK REQUIREMENTS

No.		29C331 Max. Value	Unit
53	Minimum Clock LOW Time	33	ns
54	Minimum Clock HIGH Time	28	ns

Notes: 1. (INTR, INTEN)-to-EQUAL is the sum of (INTR, INTEN)-to-Y disable time and Y-to-EQUAL delay time.
2. C_L = 50 pF; C_L = 5 pF for Disable Time only.
3. The status of $I_5 - I_0$ and FC must not be changed during the clock LOW time.

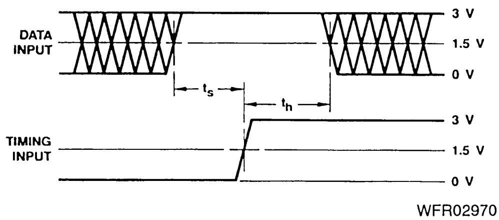
SWITCHING TEST CIRCUIT



A. Three-State Outputs

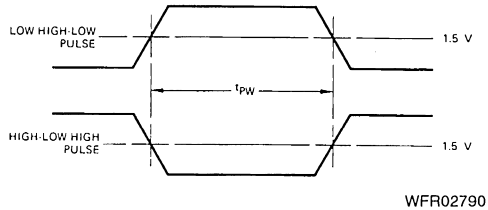
Notes: 1. $C_L = 50$ pF includes scope probe, wiring, and stray capacitances without device in test fixture.
2. S_1, S_2, S_3 are closed during function tests and all AC tests except output enable tests.
3. S_1 and S_3 are closed while S_2 is open for t_{PZH} test.
S_1 and S_2 are closed while S_3 is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

SWITCHING TEST WAVEFORMS

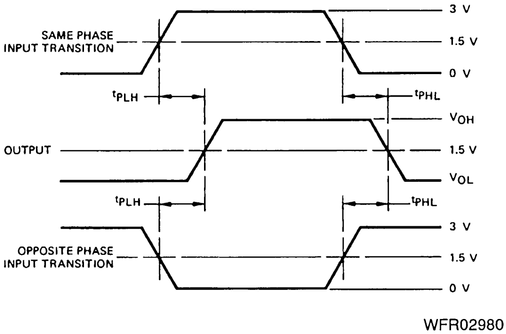


Notes: 1. Diagram shown for HIGH data only. Output transition may be opposite sense.
2. Cross hatched area is don't care condition.

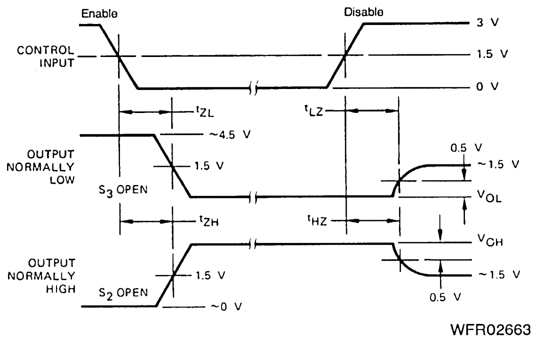
Setup, Hold, and Release Times



Pulse Width



Propagation Delay



Notes: 1. Diagram shown for Input Control Enable-LOW and Input Control Disable-HIGH.
2. S_1, S_2, and S_3 of Load Circuit are closed except where shown.

Enable and Disable Times

Test Philosophy and Methods

The following points give the general philosophy that we apply to tests that 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. Current level may vary from product to product.

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 $V_{IL} \leqslant 0$ V and $V_{IH} \geqslant 3$ 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. Capacitive Loading for AC 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 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 setup 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 setup and the knowledge that certain DC measurements (I_{OH}, I_{OL}, for example) have already been taken and are within specification. In some cases, special DC tests are performed in order to facilitate this correlation.

7. 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 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 AC testing are performed at "hard" input levels rather than at V_{IL} max. and V_{IH} min.

8. AC Testing

   Occasionally parameters are specified that 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 AC tests that have been performed. These correlations are arrived at by the cognizant engineer using data from precise bench measurements in conjunction with the knowledge that certain DC parameters have already been measured and are within specification.

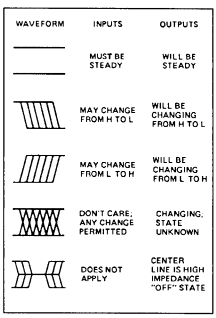
   In some cases, certain AC tests are redundant since they can be shown to be predicted by other tests that have already been performed. In these cases, the redundant tests are not performed.

9. Output Short-Circuit Current Testing

   When performing I_{OS} tests on devices containing RAM or registers, great care must be taken that undershoot caused by grounding the high-state output does not trigger parasitic elements which in turn cause the device to change state. In order to avoid this effect, it is common to make the measurement at a voltage (V_{output}) that is slightly above ground. The V_{CC} is raised by the same amount so that the result (as confirmed by Ohm's law and precise bench testing) is identical to the $V_{OUT} = 0$, V_{CC} = Max. case.

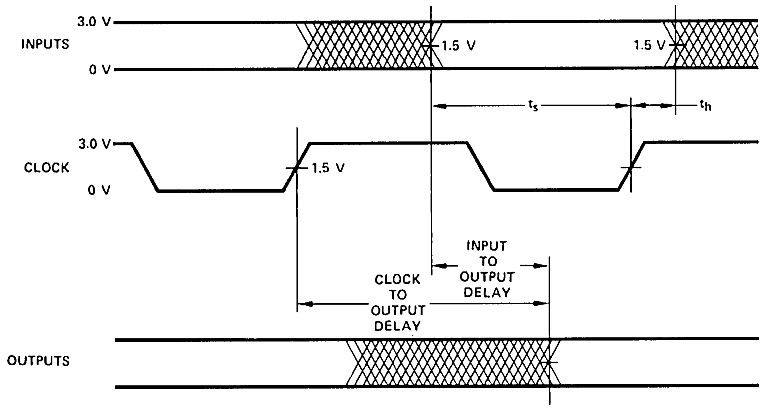
SWITCHING WAVEFORMS

KEY TO SWITCHING WAVEFORMS

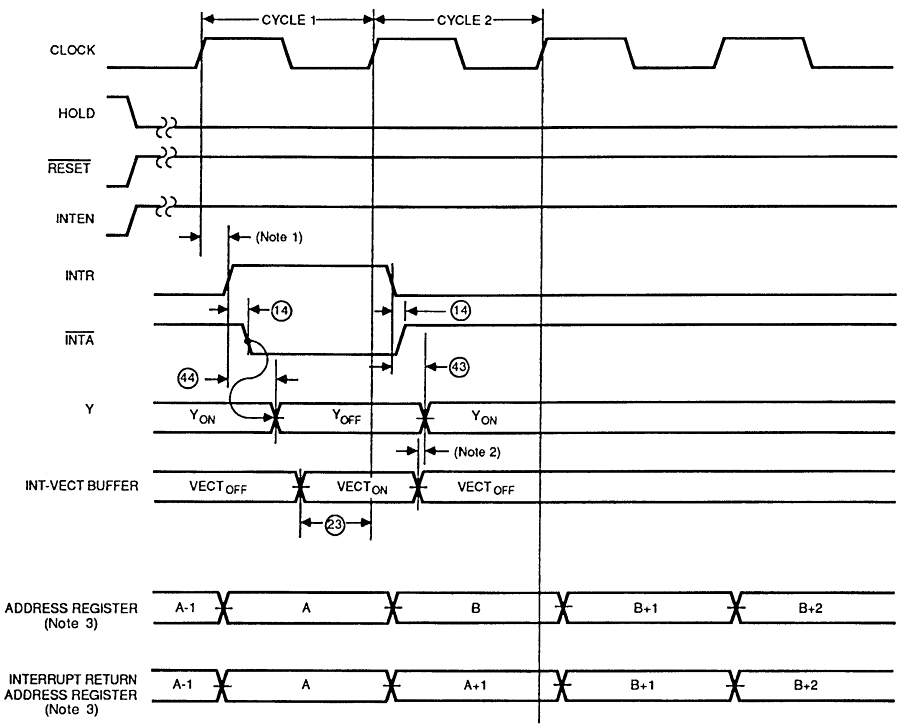


KS000010

SWITCHING WAVEFORMS (Cont'd.)



WFR02990



WF025100

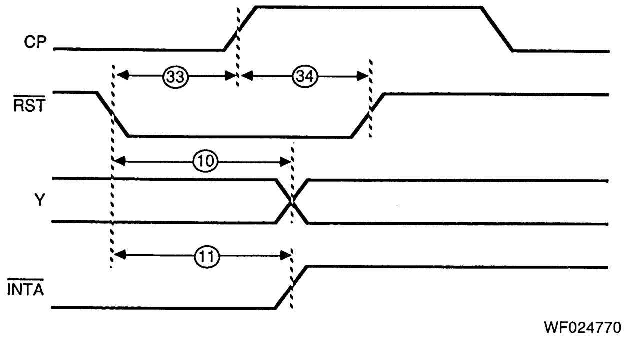
Interrupt Timing

Notes: 1. Interrupt Request comes from an interrupt-controller register. If reflects the CP ↑ to INTR time of the interrupt controller.

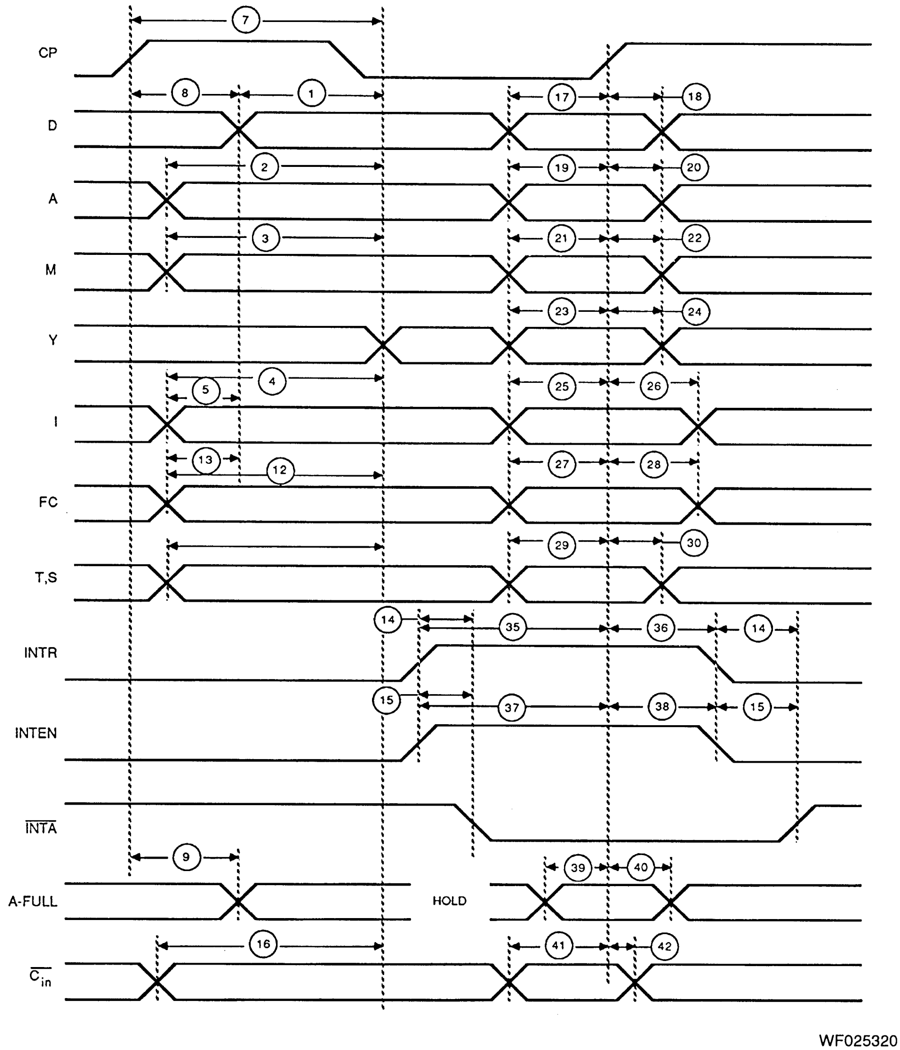
2. During Cycle 2, there may be contention on the Y-bus if the Y-bus is turned ON before the INT-VECT buffer is turned OFF.

3. Refer to Figures 4 and 5 for definition of A and B.

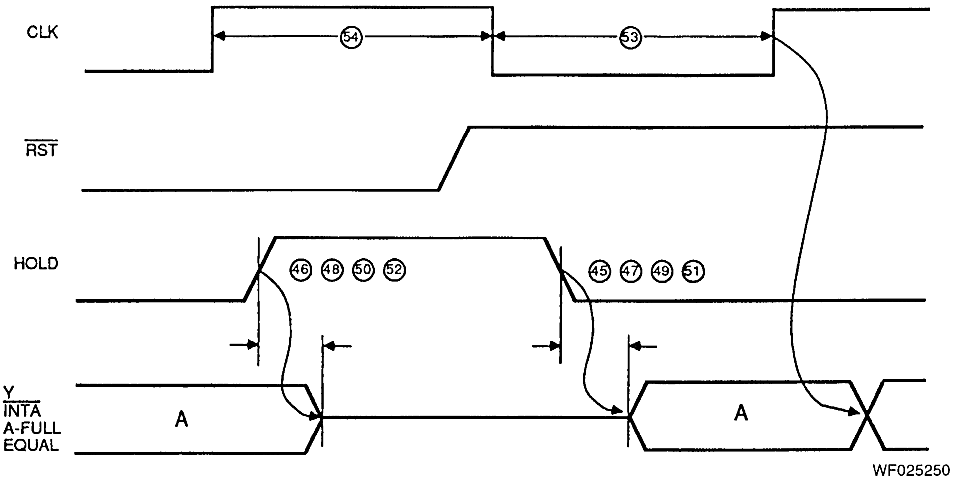
SWITCHING WAVEFORMS (Cont'd.)



Reset Timing



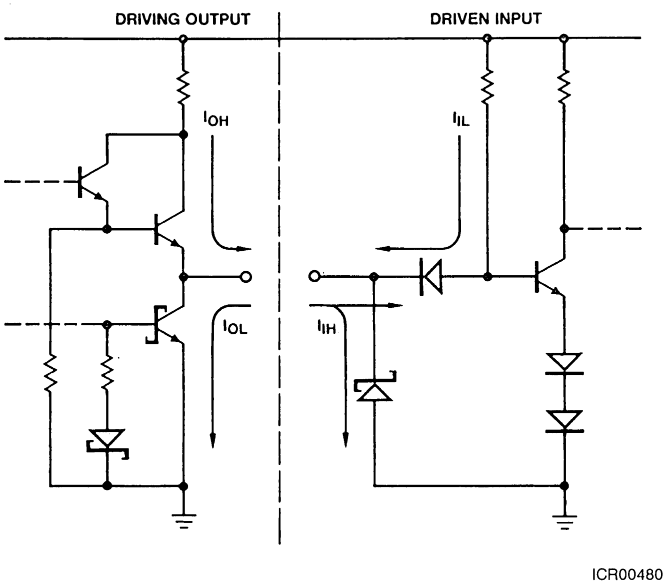
SWITCHING WAVEFORMS (Cont'd.)



Am29331 Hold Timing

INPUT/OUTPUT CIRCUIT DIAGRAM

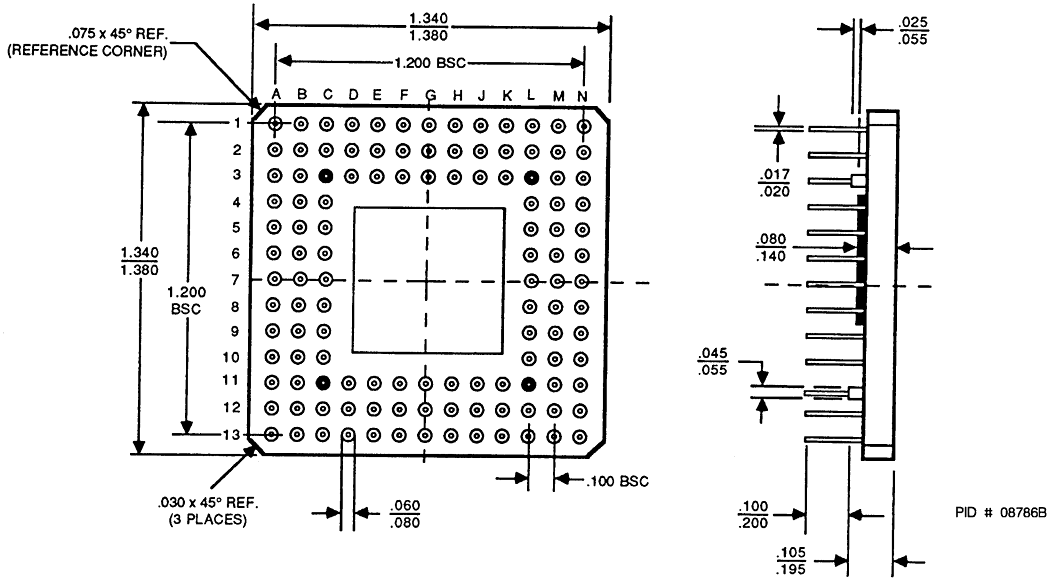
(All Devices)



PHYSICAL DIMENSIONS*

CGX120

BOTTOM VIEW



For reference only.

ADVANCED MICRO DEVICES' NORTH AMERICAN SALES OFFICES

ALABAMA	(205) 882-9122
ARIZONA	(602) 242-4400
CALIFORNIA,	
Culver City	(213) 645-1524
Newport Beach	(714) 752-6262
San Diego	(619) 560-7030
Santa Clara	(408) 727-3270
Woodland Hills	(818) 992-4155
CANADA, Ontario,	
Kanata	(613) 592-0060
Willowdale	(416) 224-5193
FAX	(416) 224-5193
COLORADO	(303) 741-2900
CONNECTICUT	(203) 264-7800
FLORIDA,	
Clearwater	(813) 530-9971
Ft Lauderdale	(305) 484-8600
Melbourne	(305) 729-0496
Orlando	(305) 859-0831
GEORGIA	(404) 449-7920
ILLINOIS,	
Chicago	(312) 773-4422
Naperville	(312) 505-9517
INDIANA	(317) 244-7207
KANSAS	(913) 451-3115
MARYLAND	(301) 796-9310
MASSACHUSETTS	(617) 273-3970
MINNESOTA	(612) 938-0001
MISSOURI	(314) 275-4415
NEW JERSEY	(201) 299-0002
NEW YORK,	
Liverpool	(315) 457-5400
Poughkeepsie	(914) 471-8180
Woodbury	(516) 364-8020
NORTH CAROLINA	(919) 847-8471
OHIO	(614) 891-6455
OREGON	(503) 245-0080
PENNSYLVANIA,	
Allentown	(215) 398-8006
Willow Grove	(215) 657-3101
TEXAS,	
Austin	(512) 346-7830
Dallas	(214) 934-9099
Houston	(713) 785-9001
WASHINGTON	(206) 455-3600
WISCONSIN	(414) 792-0590

ADVANCED MICRO DEVICES' INTERNATIONAL SALES OFFICES

BELGIUM,		
Bruxelles	TEL	(02) 771 91 42
	FAX	(02) 762 37 12
	TLX	61028
FRANCE,		
Paris	TEL	(01) 49-75-10-10
	FAX	(01) 49-75-10-13
	TLX	263282F
GERMANY,		
Hannover area	TEL	(05143) 50 55
	FAX	(05143) 55 53
	TLX	925287
München	TEL	(089) 41 14-0
	FAX	(089) 406490
	TLX	523883
Stuttgart	TEL	(0711) 62 33 77
	FAX	(0711) 625187
	TLX	721882
HONG KONG,		
Kowloon	TEL	852-3-695377
	FAX	1234276
	TLX	504260AMDAPHX
ITALY, Milano	TEL	(02) 3390541
	FAX	(02) 3498000
	TLX	315286
JAPAN,		
Tokyo	TEL	(03) 345-8241
	FAX	3425196
	TLX	J24064AMDTKOJ
Osaka	TEL	06-243-3250
	FAX	06-243-3253
KOREA, Seoul	TEL	82-2-784-7598
	FAX	82-2-784-8014
LATIN AMERICA,		
Ft. Lauderdale	TEL	(305) 484-8600
	FAX	(305) 485-9736
	TLX	5109554261 AMDFTL
SWEDEN, Stockholm	TEL	(08) 733 03 50
	FAX	(08) 733 22 85
	TLX	11602
TAIWAN	TLX	886-2-7122066
	FAX	886-2-7122017
UNITED KINGDOM,		
Manchester area	TEL	(0925) 828008
	FAX	(0925) 827693
	TLX	628524
London area	TEL	(04862) 22121
	FAX	(04862) 22179
	TLX	859103

NORTH AMERICAN REPRESENTATIVES

CALIFORNIA	
I^2 INC	OEM (408) 988-3400
	DISTI (408) 498-6868
CANADA	
Calgary, Alberta	
VITEL ELECTRONICS	(403) 278-5833
Kanata, Ontario	
VITEL ELECTRONICS	(613) 592-0090
Mississauga, Ontario	
VITEL ELECTRONICS	(416) 676-9720
Quebec	
VITEL ELECTRONICS	(514) 636-5951
IDAHO	
INTERMOUNTAIN TECH MKGT	(208) 888-6071
INDIANA	
ELECTRONIC MARKETING CONSULTANTS	(317) 253-1668
IOWA	
LORENZ SALES	(319) 377-4666
KANSAS	
LORENZ SALES	(913) 384-6556
MICHIGAN	
SAI MARKETING CORP	(313) 750-1922
MISSOURI	
LORENZ SALES	(314) 997-4558
NEBRASKA	
LORENZ SALES	(402) 475-4660
NEW MEXICO	
THORSON DESERT STATES	(505) 293-8555
NEW YORK	
NYCOM, INC	(315) 437-8343
OHIO	
Columbus	
DOLFUSS ROOT & CO	(614) 885-4844
Dayton	
DOLFUSS ROOT & CO	(513) 433-6776
Strongsville	
DOLFUSS ROOT & CO	(216) 238-0300
PENNSYLVANIA	
DOLFUSS ROOT & CO	(412) 221-4420
UTAH	
R^2 MARKETING	(801) 595-0631

ADVANCED MICRO DEVICES 901 Thompson Pl., P.O. Box 3453, Sunnyvale, CA 94088, USA
TEL: (408) 732-2400 • TWX: 910-339-9280 • TELEX: 34-6306 • TOLL FREE: (800) 538-8450

Printed in U.S.A. **AIS-WCP-13.5M-10/87-0**