Am29PL131

Field-Programmable Controller (FPC)

ADVANCE INFORMATION

DISTINCTIVE CHARACTERISTICS

- Implements complex state machines
- 7 conditional registered inputs, 12 outputs
- 64-word by 28-bit PROM
- Up to 15-MHz clock rate, 24-pin (0.3" wide) DIP
- Instruction set compatible with Am29PL141
- 29 instructions
  - Conditional branching
  - Conditional looping
  - Conditional subroutine call
  - Multiway branch

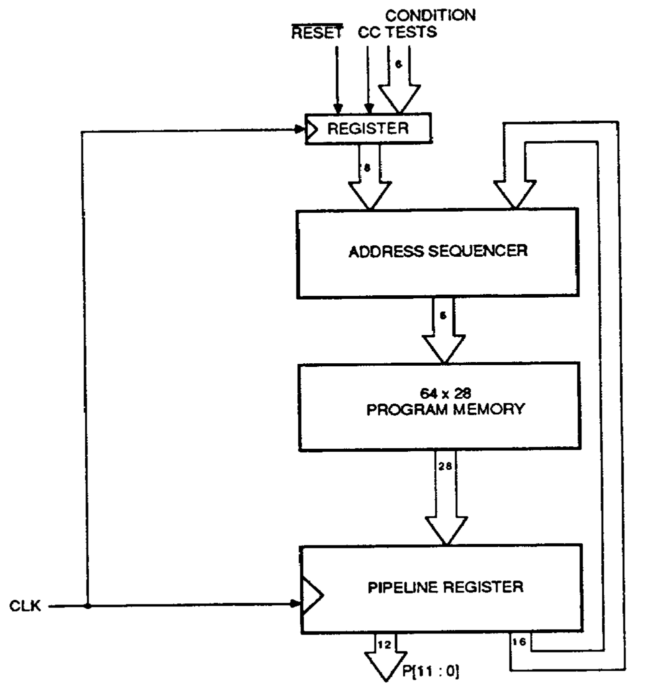
GENERAL DESCRIPTION

The Am29PL131 is a single-chip Field-Programmable Controller (FPC) that allows implementation of complex state machines and controllers by programming the appropriate sequence of instructions. Jumps, loops, and subroutine calls, conditionally executed based on the test inputs, provide the designer with powerful control flow primitives.

Intelligent control may be distributed throughout the system by using FPCs to control the various self-contained functional units, such as register file/ALU, I/O, interrupt, diagnostic, and bus control units.

An address sequencer, the heart of the FPC, provides the address to an internal 64-word by 28-bit PROM. The fuse programming algorithm is almost identical to that used for AMD's Programmable Array Logic family.

SIMPLIFIED BLOCK DIAGRAM



BD007580

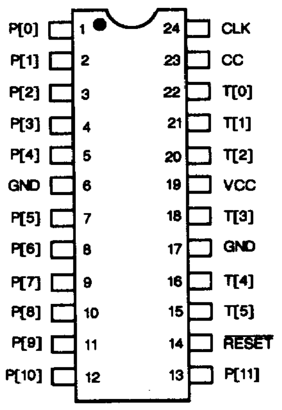
Publication # 09463 Rev. A Amendment /0
Issue Date: April 1988

RELATED AMD PRODUCTS

Part No.	Description
Am29114	8-Level Real-Time Interrupt Controller (Expandable)
Am29116	16-Bit Bipolar Microprocessor (Supports 100 ns System Cycle Time)
Am29116A	High-Performance Version of the Am29116
Am29C116/-1/-2	CMOS Version of Am29116, Speed Selects
Am29117	2-Port Version of the Am29116
Am29C117	CMOS Version of the Am29117
Am29118	8-Bit Am29116 I/O Support
Am29130	16-Bit Barrel Shifter (Expandable)
Am2914	Vectored Priority Interrupt Controller
Am29PL141	64-Word PROM Field-Programmable Controller
Am29LPL141	Low-Power Version of the Am29PL141
Am29CPL141	CMOS 64-Word EPROM Version of the Am29PL141
Am29PL142	128-Word PROM Field-Programmable Controller
Am29CPL144	CMOS 512-Word EPROM Field-Programmable Controller
Am2940	DMA Address Generator

CONNECTION DIAGRAM
Top View

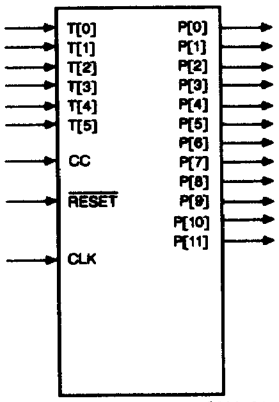
Slim DIP



CD011210

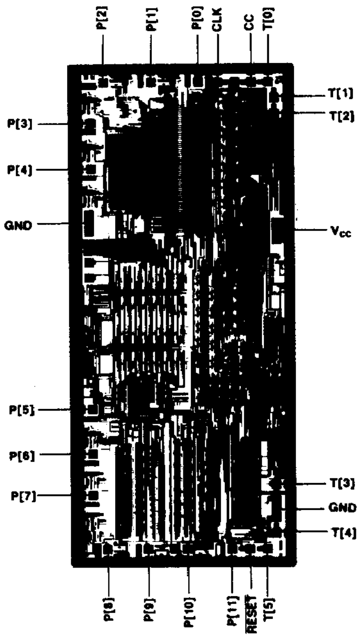
Note: Pin 1 is marked for orientation.

LOGIC SYMBOL



LS003101

METALLIZATION AND PAD LAYOUT



Die Size: 0.325" x 0.140"
Gate Count: 600 Equivalent Gates and 64 x 28 of PROM

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

AM29PL131 D C B

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

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

c. PACKAGE TYPE
D = 24-Pin Slim Ceramic DIP (CD3024)

b. SPEED OPTION
Not Applicable

a. DEVICE NUMBER/DESCRIPTION
Am29PL131
Field-Programmable Controller (FPC)

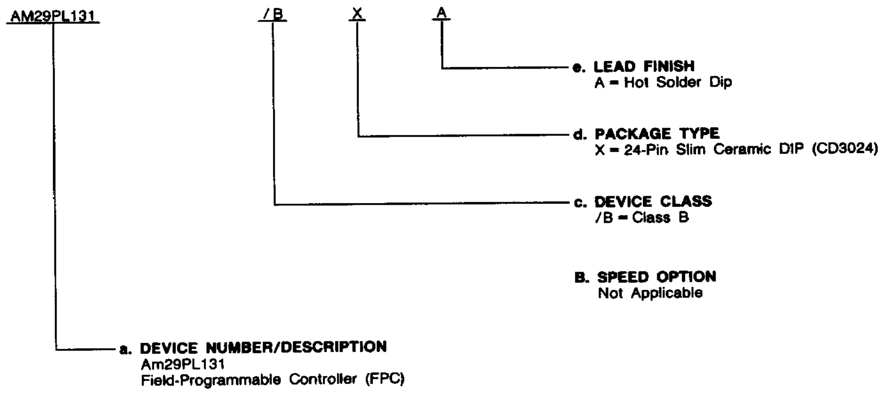
Valid Combinations	
AM29PL131	DC, DCB

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) is formed by a combination of: **a. Device Number**
b. Speed Option (if applicable)
c. Device Class
d. Package Type
e. Lead Finish



Valid Combinations	
AM29PL131	/BXA

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 consists of Subgroups
1, 2, 3, 7, 8, 9, 10, 11.

PIN DESCRIPTION

CC Condition Code-TEST (Input)
The internally synchronized condition-code test input is selected through the 3-bit test-condition-select field.

CLK Clock (Input)
The rising edge of the clock latches the program counter, count register, subroutine register, pipeline register, test-input register, CC register, and EQ flag.

P[11:8] (Outputs)
The upper four general-purpose control outputs are permanently enabled.

P[7:0] (Outputs)
The lower eight general-purpose control outputs are enabled by the OE bit from the instruction pipeline register. When OE is HIGH, these outputs are enabled; when LOW, they are three-stated.

$\overline{\text{RESET}}$ Internally Synchronized Reset (Input; Active LOW)
$\overline{\text{RESET}}$ is latched internally on the first rising edge of the clock after it goes LOW. During the first clock cycle, the PC MUX is set to all ones (address 63). On the next rising edge of the clock, the program-memory contents at location 63 are loaded into the pipeline register. The EQ flag is also cleared at this time.

T[5:0] Internally Synchronized Test (Inputs)
In conditional instructions, these inputs are selected according to the 3-bit test-condition-select field. Inputs T(5:0) can also be used either as a branch address or as a value to be loaded into CREG, depending on the instruction.

FUNCTIONAL DESCRIPTION

Figure 1, the detailed block diagram of the Am29PL131 FPC, shows logic blocks and interconnecting buses that permit parallel performance of different operations in a single instruction. The FPC consists of four main logic blocks: the program memory, address control logic, condition code selection logic, and instruction decode.

The program memory contains the user-defined instruction flow and output sequence. The address control logic addresses the program memory. This control logic supports high-level instruction functions including conditional branches, subroutine calls and returns, loops, and multiway branches. The condition code selection logic selects the condition code input to be tested when a conditional instruction is executed. The polarity of the selected condition code input is controlled by the POL bit in the microword. The instruction decode generates the control signals necessary to perform the instruction specified by the instruction part (P[27:12]) of the microword.

Program Memory

The FPC program memory is a 64-word by 28-bit PROM with a 28-bit pipeline register at its output. The upper 16 bits (P[27:12]) of the pipeline register are internal to the FPC and form the instruction to control address sequencing. The format for instructions is: a 1-bit synchronous Output Enable OE, a 5-bit OPCODE, a 1-bit test polarity select POL, a 3-bit TEST condition select field, and a 6-bit immediate DATA field. The DATA field is used to provide branch addresses, test input masks, and counter values.

The lower 12 bits (P[11:0]) of the pipeline register are brought out as user-defined, general purpose control outputs. The lower eight control outputs (P[7:0]) are three-stated when OE is programmed as a LOW. The upper four control bits (P[11:8]) are always enabled.

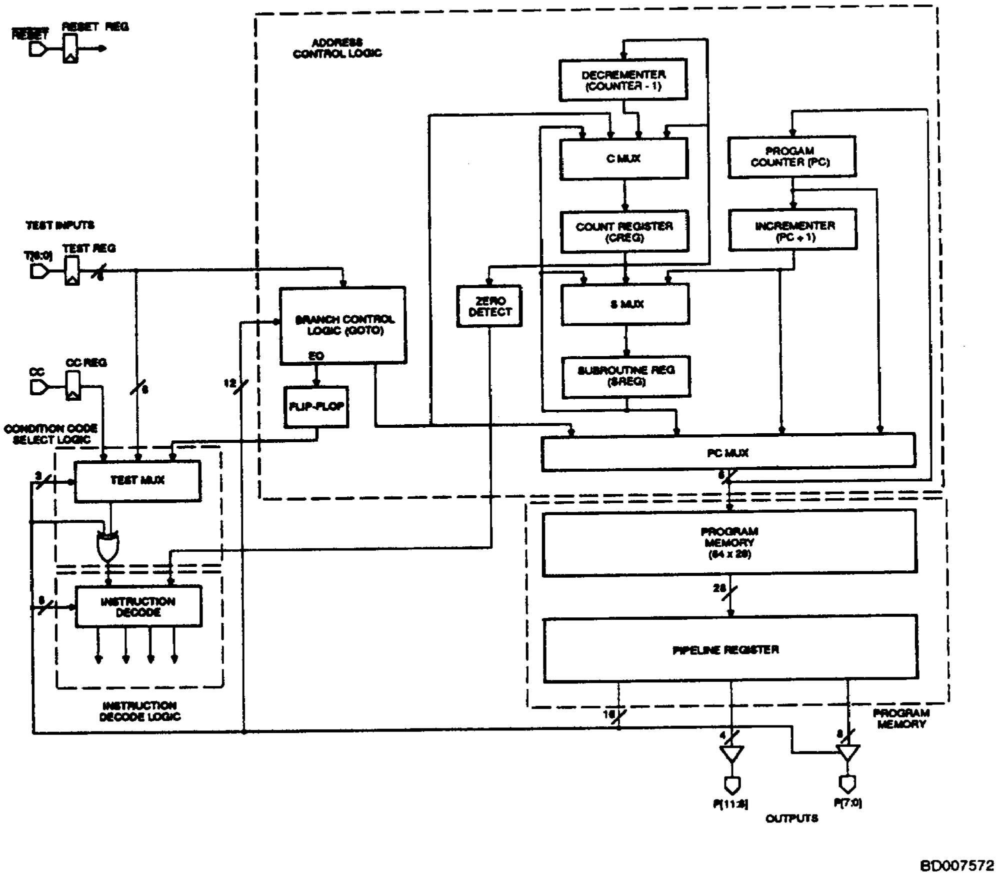


Figure 1. Am29PL131 Detailed Block Diagram

Address Control Logic

The address control logic consists of five smaller logic blocks. These are:

PC MUX	– Program counter multiplexer
P CNTR	– Program counter (PC) and incrementer (PC + 1)
SUBREG	– Subroutine register (SREG) with subroutine mux (S MUX)
CNTR	– Count register (CREG) with counter mux (C MUX), decrementer (COUNTER-1), and zero detect
GOTO	– Specialized branch control logic

The PC MUX is a 6-bit, 4-to-1 multiplexer. It selects either the PC, PC+1, SREG, or GOTO output as the next microaddress input to the Program Memory and to the PC. The PC thus always contains the address of the instruction in the pipeline register. On the rising edge of the clock after $\overline{\text{RESET}}$ goes LOW, the PC MUX output is forced to all ones, selecting location 63 of the Program Memory.

The P CNTR block consists of a 6-bit register (PC) driving a 6-bit combinatorial incrementer (PC+1). Either the present or the incremented values of PC can address the Program Memory. The incremented value of PC can be saved as a subroutine return address. The present PC value can address the Program Memory when waiting for a condition to become valid. PC+1 addresses the Program Memory for sequential program flow and for unconditional instructions.

The SUBREG block consists of a 6-bit, 3-to-1 multiplexer (S MUX) driving a 6-bit register (SREG). The three possible SREG inputs are PC+1, CREG, and SREG. SREG normally operates as a 1-deep stack to save subroutine return addresses. PC+1 is the input source when performing subroutine calls, and PC MUX is the output destination when performing return from subroutine.

The CNTR block consists of a 6-bit, 4-to-1 multiplexer (C MUX); driving a 6-bit register (CREG); a 6-bit, combinatorial decrementer (COUNTER-1); and a zero-detection circuit. The CNTR logic block is typically used for timing functions and iterative loop counting.

The SUBREG and CNTR can be considered as one logic block because of their unique interaction. Both have the other as an additional input source and output destination. The CREG can therefore be an additional stack location when not used for counting, and the SREG can be a nested-count location when not used as a stack location. Thus the SREG and CREG can operate in three different modes:

1) As a separate 1-deep stack and counter.
2) As a 2-deep stack.
3) As a 2-deep nested counter.

The GOTO logic block serves three functions:

1) It provides a 6-bit count value from the DATA field in the pipeline register (P[17:12]) or from the TEST inputs (T[5:0]) masked by the DATA field (P[17:12]). This is represented by T*M.

2) It provides a branch address from the DATA field in the pipeline register (P[17:12]) or from the TEST inputs (T[5:0]) masked by the DATA field (P[17:12]). This is represented by T*M.

3) It compares the TEST inputs: T[5:0] masked by the DATA field (P[17:12]), called T*M, to the CONSTANT field from the pipeline register (P[23:18]). If a match occurs, the EQ flip-flop is set. EQ remains unchanged if there is no match. Constant field bits that correspond to masked test bits must be zero.

The EQ flag can be tested by the condition code selection logic. Multiple tests of any group of T inputs in a manner analogous to sum-of-products can be performed since a no-match comparison does not reset the EQ flag. Any conditional branch on EQ will reset the EQ flag. Conditional returns on EQ will not change the EQ flag. $\overline{\text{RESET}}$ input LOW will reset the EQ flag.

NOTE: A zero in the DATA field blocks the corresponding bit in the TEST field; a one activates the corresponding bit.

The constant field bits that correspond to masked test field bits must be zero. A zero is substituted for masked test field bits. The 'POL' bit is a "don't care" when using test inputs to load registers.

Condition Code Selection Logic

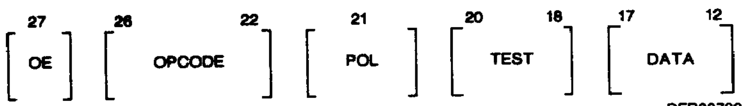
The condition code selection logic consists of an 8-to-1 multiplexer. The eight test conditon inputs are the device inputs, CC, T[5:0], and the EQ flag. The TEST field (P[20:18]) selects one of the eight conditions to test.

The polarity bit POL in the instructions allows the user to test for either a true or false conditon. Refer to Table 1 for details.

Instruction Decode

The instruction decoder is a PLA that generates the control for 29 different instructions. The decoder inputs include the OPCODE field (P[26:22]), the zero detection output from the CNTR, and the selected test condition code from the condition code selection logic.

Am29PL131 General Instruction Format



DFR00732

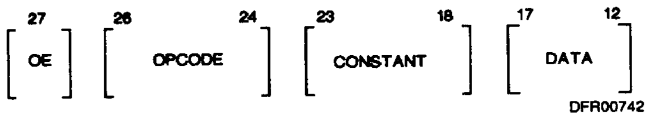
WHERE:

OE	= Synchronous Output Enable for P[7:0].
OPCODE	= A 5-bit opcode field for selecting one of the 28 single-data-field instructions.
POL	= A 1-bit test condition polarity select. 0 = Test for true (HIGH) condition. 1 = Test for false (LOW) condition.
TEST	= A 3-bit test condition select.

TEST[2:0]	UNDER TEST
000	T[0]
001	T[1]
010	T[2]
011	T[3]
100	T[4]
101	T[5]
110	CC
111	EQ

DATA	= A 6-bit conditional branch address, test input mask, or counter value field designated as PL in instruction mnemonics.

Am29PL131 Comparison Instruction Format



DFR00742

WHERE:

OE	= Synchronous Output Enable for P[7:0].
OPCODE	= Compare instruction (binary 100).
CONSTANT	= A 6-bit constant for equal to comparison with T*M.
DATA	= A 6-bit mask field for masking the incoming T[5:0] inputs.

TABLE 1.

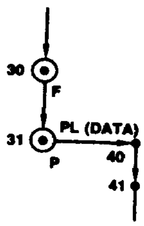
Input Condition Being Tested	POL	Condition
0	0	Fail
0	1	Pass
1	0	Pass
1	1	Fail

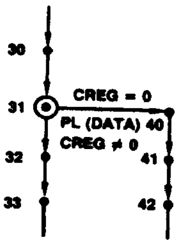
Am29PL131 INSTRUCTION SET DEFINITION

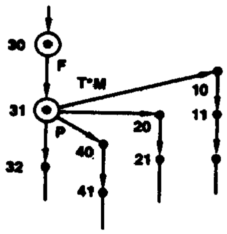
● = Other instruction
⊙ = Instruction being described
O = Register in part

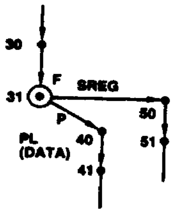
P = Test Pass
F = Test Fail
X, Y are arbitrary values in the CREG or SREG

Opcode	Mnemonic	Description	Execution Example	Register Transfer Description

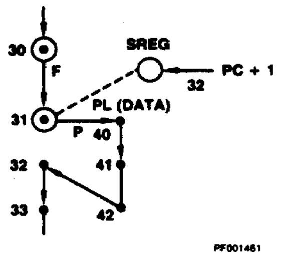
19	GOTOPL	**IF (cond) THEN GOTO PL (data)** Conditional branch to the address in the PL (DATA field). The EQ flag will be reset if the test field selects it and the condition passes.	 PF001420	If (cond = true) Then
				PC = PL(data)
				Else
				PC = PC + 1

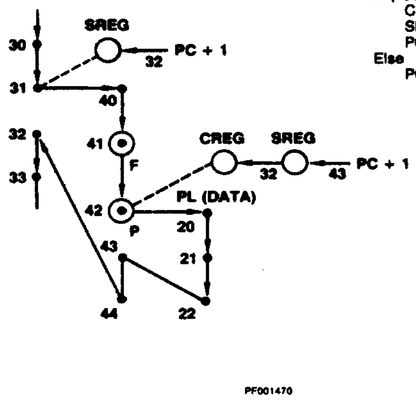
0B	GOTOPLZ	**IF (CREG = O) THEN GOTO PL (data)** Conditional branch to the address in the PL (DATA field) when CREG is equal to zero. This instruction does not depend on the pass/fail condition. The EQ flag will be reset if the test field selects it and the CREG is equal to zero.	 30
			CREG = 0
			31
			PL (DATA) 40
			CREG ≠ 0
			32
			41
			33
			42 PF001430
				If (CREG = 0) Then
				PC = PL(data)
				Else
				PC = PC + 1

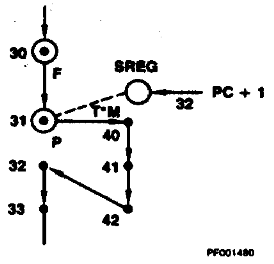
0F	GOTOTM	**IF (cond) THEN GOTO TM (data)** Conditional branch to the address defined by the T*M (T[5:0] under bitwise mask from the DATA field). This instruction is intended for multiway branches. The EQ flag will be reset if the test field selects it and the condition passes.	 30
			F
			T*M
			10
			31
			11
			20
			P
			40
			21
			32
			41 PF001440
				If (cond = true) Then
				PC = T*M
				Else
				PC = PC + 1

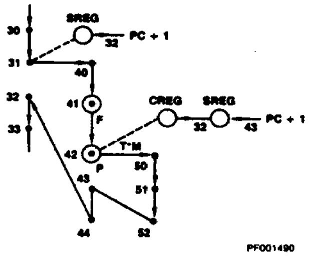
18	FORK	**IF (cond) THEN GOTO PL (data) ELSE GOTO (SREG)** Conditional branch to the address in the PL (DATA field) or the SREG. A branch to PL is taken if the condition is true and a branch to SREG if false. The EQ flag will be reset if the test field selects it and the condition passes.	 30
			F SREG
			31
			P
			50
			PL (DATA)
			40
			51
			41 PF001451
				If (cond = true) Then
				PC = PL(data)
				Else
				PC = SREG

Opcode	Mnemonic	Description	Execution Example	Register Transfer Description

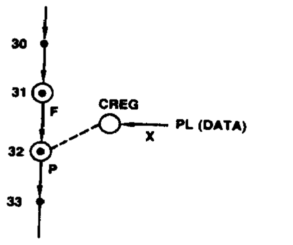
1C	CALPL	**IF (cond) THEN CALL PL (data)**
		Conditional jump to subroutine at the address in the PL (DATA field). The PC + 1 is pushed into the SREG as the return address. The EQ flag will be reset if the test field selects it and the condition passes.
			 SREG
			PC + 1
			30
			31
			32
			33
			40
			41
			42
			F
			P
			PL (DATA)
			PF001461
				If (cond = true) Then
				SREG = PC + 1
				PC = PL(data)
				Else
				PC = PC + 1

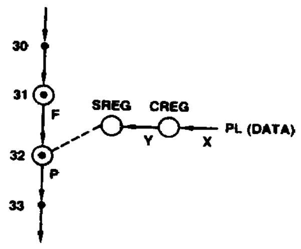
1D	CALPLN	**IF (cond) THEN CALL PL (data), NESTED**
		Conditional jump to subroutine at the address in the PL (DATA field) nested. The SREG and CREG are treated as a two-deep stack, the PC + 1 is pushed into the SREG as the return address, and the previous SREG value is transferred into the CREG as a nested return address. The EQ flag will be reset if the test field selects it and the condition passes.
			 SREG
			PC + 1
			CREG
			SREG
			PC + 1
			30
			31
			32
			33
			40
			41
			42
			43
			44
			20
			21
			22
			F
			P
			PL (DATA)
			PF001470
				If (cond = true) Then
				CREG = SREG
				SREG = PC + 1
				PC = PL(data)
				Else
				PC = PC + 1

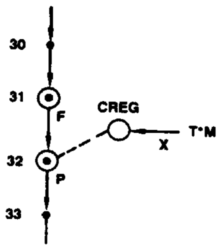
1E	CALTM	**IF (cond) THEN CALL TM (data)**
		Conditional jump to subroutine at the address specified by the T*M (T[5:0] under bitwise mask from the DATA field). The PC + 1 is pushed into the SREG as the return address. The EQ flag will be reset if the test field selects it and the condition passes.
			 SREG
			PC + 1
			30
			31
			32
			33
			40
			41
			42
			F
			P
			T*M
			PF001480
				If (cond = true) Then
				SREG = PC + 1
				PC = T*M
				Else
				PC = PC + 1

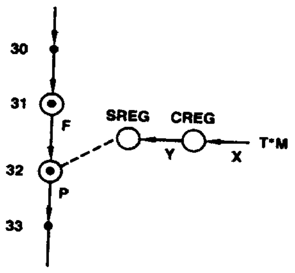
1F	CALTMN	**IF (cond) THEN CALL TM (data), NESTED**
		Conditional jump to subroutine at the address specified by the T*M (T[5:0] under bitwise mask from the DATA field) nested. The PC + 1 is pushed into the SREG as the return address and the previous SREG value is transferred into the CREG as a nested return address. The EQ flag will be reset if the test field selects it and the condition passes.
			 SREG
			PC + 1
			CREG
			SREG
			PC + 1
			30
			31
			32
			33
			40
			41
			42
			43
			44
			50
			51
			52
			F
			P
			T*M
			PF001490
				If (cond = true) Then
				CREG = SREG
				SREG = PC + 1
				PC = T*M
				Else
				PC = PC + 1

Opcode	Mnemonic	Description	Execution Example	Register Transfer Description

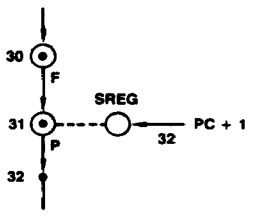
04	LDPL	**IF (cond) THEN LOAD PL (data)** Conditional Load the CREG from the PL (DATA field).	 PF001512	If (cond = true) Then
				CREG = PL(data)
				PC = PC + 1
				Else
				PC = PC + 1

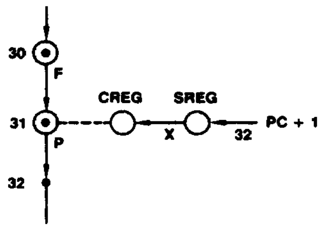
05	LDPLN	**IF (cond) THEN LOAD PL (data), NESTED** Conditional load the CREG from the PL (DATA field) nested. The CREG and SREG are treated as a two-deep nested count register, the previous CREG value is pushed into the SREG as a nested count, and the CREG is loaded from PL.	 30
			31
			F
			SREG
			CREG
			PL (DATA)
			Y
			X
			32
			P
			33 PF001501
				If (cond = true) Then
				SREG = CREG
				CREG = PL(data)
				PC = PC + 1
				Else
				PC = PC + 1

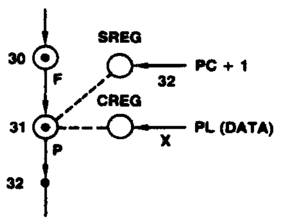
06	LDTM	**IF (cond) THEN LOAD TM (data)** Conditional load the CREG from the T*M (T[5:0] inputs under bitwise mask from the DATA field).	 PF001522	If (cond = true) Then
				CREG = T*M
				PC = PC + 1
				Else
				PC = PC + 1

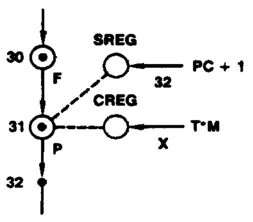
07	LDTMN	**IF (cond) THEN LOAD TM (data), NESTED** Conditional load the CREG from the T*M (T[5:0] inputs under bitwise mask from the DATA field) nested. The SREG and CREG are treated as a two-deep nested count register, the previous CREG value is transferred into the SREG, and the CREG is loaded from T*M.	 30
			31
			F
			SREG
			CREG
			T*M
			Y
			X
			32
			P
			33 PF001531
				If (cond = true) Then
				SREG = CREG
				CREG = T*M
				PC = PC + 1
				Else
				PC = PC + 1

Opcode	Mnemonic	Description	Execution Example	Register Transfer Description

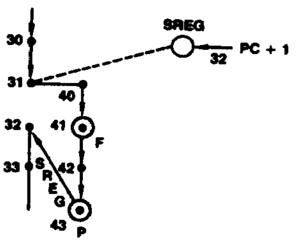
15	PSH	**IF (cond) THEN PUSH**
		Conditional push the PC + 1 into the SREG.
			 PF0001540	If (cond = true) Then
				SREG = PC + 1
				PC = PC + 1
				Else
				PC = PC + 1

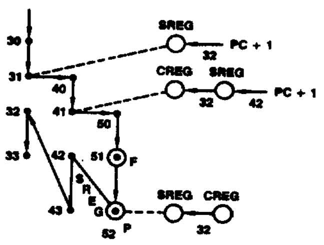
17	PSHN	**IF (cond) THEN PUSH, NESTED**
		Conditional push the PC + 1 into the SREG nested. This microinstruction treats the SREG and CREG as a two-deep stack, PC + 1 is pushed into SREG, and the previous value in SREG is transferred into the CREG.
			 30
			F
			CREG
			SREG
			31
			X
			32
			PC + 1
			P
			32 PF001551
				If (cond = true) Then
				CREG = SREG
				SREG = PC + 1
				PC = PC + 1
				Else
				PC = PC + 1

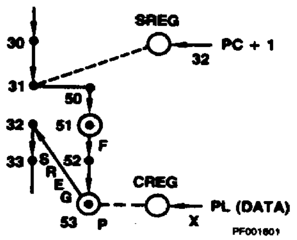
14	PSHPL	**IF (cond) THEN PUSH, LOAD PL (data)**
		Conditional push the PC + 1 into the SREG and load the CREG from the PL (DATA field).
			 SREG
			30
			PC + 1
			F
			32
			CREG
			31
			PL (DATA)
			X
			P
			32 PF001561
				If (cond = true) Then
				CREG = PL(data)
				SREG = PC + 1
				PC = PC + 1
				Else
				PC = PC + 1

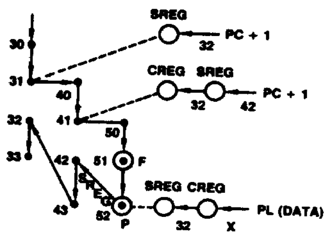
16	PSHTM	**IF (cond) THEN PUSH, LOAD TM (data)**
		Conditional push the PC + 1 into the SREG and load the CREG from the T*M (T[5:0] under bitwise mask from the DATA field).
			 SREG
			30
			PC + 1
			F
			32
			CREG
			31
			T*M
			X
			P
			32 PF001571
				If (cond = true) Then
				CREG = T*M
				SREG = PC + 1
				PC = PC + 1
				Else
				PC = PC + 1

Opcode	Mnemonic	Description	Execution Example	Register Transfer Description

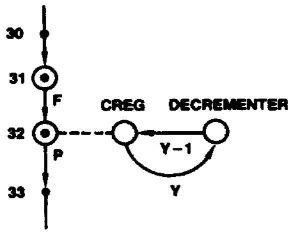
02	RET	**IF (cond) THEN RET**
		Conditional return from subroutine. The SREG provides the return from subroutine address.
			
			PF001580
				If (cond = true) Then
				PC = SREG
				Else
				PC = PC + 1

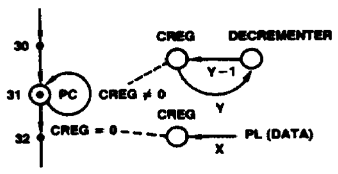
03	RETN	**IF (cond) THEN RET, NESTED**
		Conditional return from nested subroutine. This instruction treats the SREG and CREG as a two-deep stack providing the SREG value as a return address, and the CREG value as a nested return address that is transferred into the SREG.
			 SREG CREG SREG SREG CREG 30 31 32 33 40 41 42 43 50 51 52 F P PC + 1 S R E G
			PF001590
				If (cond = true) Then
				PC = SREG
				SREG = CREG
				Else
				PC = PC + 1

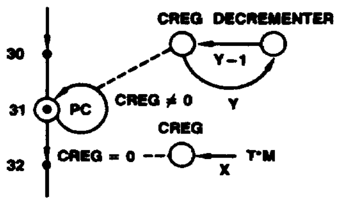
00	RETPL	**IF (cond) THEN RET, LOAD PL (data)**
		Conditional return from subroutine and load the CREG from the PL (DATA field). The SREG provides the return from subroutine address.
			 SREG CREG 30 31 32 33 50 51 52 53 F P X PC + 1 PL (DATA) S R E G
			PF001601
				If (cond = true) Then
				CREG = PL(data)
				PC = SREG
				Else
				PC = PC + 1

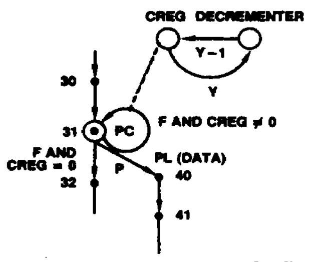
01	RETPLN	**IF (cond) THEN RET NESTED, LOAD PL (data)**
		Conditional return from nested subroutine and load the CREG from the PL (DATA field). This instruction treats the SREG and CREG as a two-deep stack providing the SREG value as a return address, and the CREG value as a nested return address that is transferred into the SREG. The CREG is loaded from the PL (DATA) field.
			 SREG CREG SREG SREG CREG 30 31 32 33 40 41 42 43 50 51 52 F P X PC + 1 PL (DATA) S R E G
			PF001611
				If (cond = true) Then
				PC = SREG
				SREG = CREG
				CREG = PL(data)
				Else
				PC = PC + 1

Opcode	Mnemonic	Description	Execution Example	Register Transfer Description

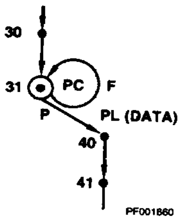
09	DEC	**IF (cond) THEN DEC**
		Conditional decrement of the CREG.
			 30
			31
			F
			CREG
			DECREMENTER
			32
			Y−1
			P
			33
			Y
			PF001622
				If (cond = true) Then
				  CREG = CREG − 1
				  PC = PC + 1
				Else
				  PC = PC + 1

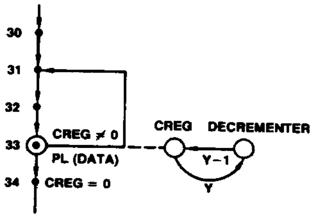
0C	DECPL	**WHILE (CREG <> 0) WAIT ELSE LOAD PL (data)**
		Conditional Hold until the counter is equal to zero, then load CREG from the PL (DATA field). This instruction is intended for timing waveform generation. If the CREG is not equal to zero, the same instruction is refetched while CREG is decremented. Timing is complete when the CREG is equal to zero, causing the next instruction to be fetched and the CREG to be reloaded from PL. This instruction does not depend on the pass/fail condition.
			 CREG
			DECREMENTER
			30
			Y−1
			31
			PC
			CREG ≠ 0
			Y
			CREG
			32
			CREG = 0
			PL (DATA)
			X
			PF001633
				While (CREG < > 0)
				  CREG = CREG − 1
				  PC = PC
				  End While
				CREG = PL(data)
				PC = PC + 1

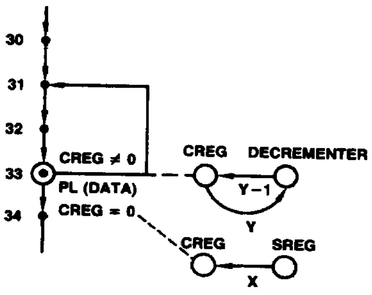
0E	DECTM	**WHILE (CREG <> 0) WAIT ELSE LOAD TM (data)**
		Conditional Hold until the counter is equal to zero, then load CREG from the T*M (T[5:0] under bitwise mask from the DATA field). This instruction is intended for timing waveform generation. If the CREG is not equal to zero, the same instruction is refetched while the CREG is decremented. Timing is complete when the CREG is equal to zero, causing the next instruction to be fetched and the CREG to be reloaded from T*M. This instruction does not depend on the pass/fail condition.
			 CREG DECREMENTER
			30
			Y−1
			31
			PC
			CREG ≠ 0
			Y
			CREG
			32
			CREG = 0
			T*M
			X
			PF001642
				While (CREG < > 0)
				  CREG = CREG − 1
				  PC = PC
				  End While
				CREG = T*M
				PC = PC + 1

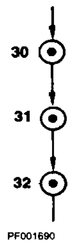
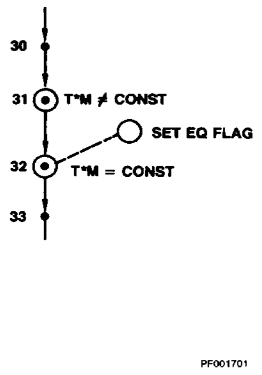
1B	DECGOPL	**IF (cond) THEN GOTO PL (data) ELSE WHILE (CREG <> 0) WAIT**
		Conditional Hold/Count. The current instruction will be refetched and the CREG decremented until the condition under test becomes true or the counter is equal to zero. If the condition becomes true, a branch to the address in the PL (DATA field) is executed. If the counter becomes zero without the condition becoming true, a CONTINUE is executed. The EQ flag will be reset if the test field selects it and the condition passes.
			 CREG DECREMENTER
			Y−1
			30
			Y
			31
			PC
			F AND CREG ≠ 0
			F AND CREG = 0
			P
			PL (DATA)
			40
			32
			41
			PF001652
				While (cond = false)
				  If (CREG < > 0)
				    CREG = CREG − 1
				    PC = PC
				  Else
				    PC = PC + 1
				  End While
				PC = PL(data)

Opcode	Mnemonic	Description	Execution Example	Register Transfer Description

1A	WAIT	**IF (cond) THEN GOTO PL (data) ELSE WAIT**
		Conditional Hold. The current instruction will be refetched and executed until the condition under test becomes true. When true, a branch to the address in the PL (DATA field) is executed. The EQ flag will be reset if the test field selects it and the condition passes.
			 PF001660	If (cond = true) Then
				PC = PL(data)
				Else
				PC= PC

08	LPPL	**WHILE (CREG < > 0) LOOP TO PL (data)**
		Conditional loop to the address in the PL (DATA field). This instruction is intended to be placed at the bottom of an iterative loop. If the CREG is not equal to zero, it is decremented (signifying completion of an iteration), and a branch to the PL address (top of the loop) is executed. If the CREG is equal to zero, looping is complete and the next sequential instruction is executed. This instruction does not depend on the pass/fail condition. The EQ flag will be reset if the test field selects it and CREG is not equal to zero.
			 30
			31
			32
			33
			34
			CREG ≠ 0
			PL (DATA)
			CREG = 0
			CREG
			DECREMENTER
			Y−1
			Y PF001672
				While (CREG < > 0)
				CREG = CREG − 1
				PC = PL (data)
				End While
				PC = PC + 1

0A	LPPLN	**WHILE (CREG < > 0) LOOP TO PL (data) ELSE NEST**
		Conditional loop to the address in the PL (DATA field) nested. The SREG and CREG are treated as a two-deep nested count register, and the instruction is intended to be placed at the bottom of an "inner-nested" iterative loop. If the CREG is not equal to zero, the CREG is decremented (signifying completion of an iteration), and a branch to the PL address (top of the loop) is executed. If the CREG is equal to zero, the inner loop is complete, and the count value for the outer loop is transferred from the SREG into the CREG. This instruction does not depend on the pass/fail condition. The EQ flag will be reset if the test field selects it and CREG is not equal to zero.
			 30
			31
			32
			33
			34
			CREG ≠ 0
			PL (DATA)
			CREG = 0
			CREG
			DECREMENTER
			Y−1
			Y
			CREG
			SREG
			X PF001681
				While (CREG < > 0)
				CREG = CREG − 1
				PC = PL(data)
				End While
				CREG = SREG
				PC = PC + 1

Opcode	Mnemonic	Description	Execution Example	Register Transfer Description
0D	CONT	**CONTINUE** The next sequential instruction is fetched unconditionally.		PC = PC + 1
10 – 13 (100XX binary)	CMP	**CMP TM (data) TO PL (data)** This instruction performs bitwise exclusive-or of T*M (T[5:0] under bitwise mask from the DATA field) with CONSTANT (P[23:18]). If T*M equals CONSTANT, the EQ flag is set to one, which may be branched on in a following instruction. If not equal, the EQ flag is unaffected. This allows sequences of compares, in a manner analogous to sum-of-products, to be performed which can be followed by a single conditional branch if one or more of the comparisons are true. Note: The EQ flag is set to zero on reset or when EQ is selected as the test condition in a branch. Conditional returns on EQ leave the flag unchanged. **Constant field bits that correspond to masked test field bits must be zero.** This instruction does not depend on the pass/fail condition.	 30 31 T*M ≠ CONST SET EQ FLAG 32 T*M = CONST 33 PF001701	Compare T*M and PL(data) EQ = ((T [5:0] .AND. DATA) .XNOR. CONSTANT) .OR. EQ

INSTRUCTIONS BASED ON TEST CONDITIONS

Opcode	Mnemonic	Assembler Statement	Condition Pass				Condition Fail				Notes
			PC MUX	STACK	CREG	EQ FLAG	PC MUX	STACK	CREG	EQ FLAG	
00	RETPL	IF (cond) THEN RET, LOAD PL (data)	SREG	Hold	Load PL	NC	PC + 1	Hold	Hold	NC	
01	RETPLN	IF (cond) THEN RET NESTED, LOAD PL (data)	SREG	Load CREG	Load PL	NC	PC + 1	Hold	Hold	NC	
02	RET	IF (cond) THEN RET	SREG	Hold	Hold	NC	PC + 1	Hold	Hold	NC	
03	RETN	IF (cond) THEN RET, NESTED	SREG	Load CREG	Hold	NC	PC + 1	Hold	Hold	NC	
04	LDPL	IF (cond) THEN LOAD PL (data)	PC + 1	Hold	Load PL	NC	PC + 1	Hold	Hold	NC	
05	LDPLN	IF (cond) THEN LOAD PL (data), NESTED	PC + 1	Load CREG	Load PL	NC	PC + 1	Hold	Hold	NC	
06	LDTM	IF (cond) THEN LOAD TM (data)	PC + 1	Hold	Load TM	NC	PC + 1	Hold	Hold	NC	
07	LDTMN	IF (cond) THEN LOAD TM (data), NESTED	PC + 1	Load CREG	Load TM	NC	PC + 1	Hold	Hold	NC	
09	DEC	IF (cond) THEN DEC	PC + 1	Hold	DEC	NC	PC + 1	Hold	Hold	NC	
0F	GOTOTM	IF (cond) THEN GOTO TM (data)	TM	Hold	Hold	Reset	PC + 1	Hold	Hold	NC	1
14	PSHPL	IF (cond) THEN PUSH, LOAD PL (data)	PC + 1	PC + 1	Load PL	NC	PC + 1	Hold	Hold	NC	
15	PSH	IF (cond) THEN PUSH	PC + 1	PC + 1	Hold	NC	PC + 1	Hold	Hold	NC	
16	PSHTM	IF (cond) THEN PUSH, LOAD TM (data)	PC + 1	PC + 1	Load TM	NC	PC + 1	Hold	Hold	NC	
17	PSHN	IF (cond) THEN PUSH, NESTED	PC + 1	PC + 1	Load SREG	NC	PC + 1	Hold	Hold	NC	
18	FORK	IF (cond) THEN GOTO PL (data) ELSE GOTO (SREG)	PL	Hold	Hold	Reset	SREG	Hold	Hold	NC	1
19	GOTOPL	IF (cond) THEN GOTO PL (data)	PL	Hold	Hold	Reset	PC + 1	Hold	Hold	NC	1
1A	WAIT	IF (cond) THEN GOTO PL (data) ELSE WAIT	PL	Hold	Hold	Reset	PC	Hold	Hold	NC	1
1C	CALPL	IF (cond) THEN CALL PL (data)	PL	PC + 1	Hold	Reset	PC + 1	Hold	Hold	NC	1
1D	CALPLN	IF (cond) THEN CALL PL (data), NESTED	PL	PC + 1	Load SREG	Reset	PC + 1	Hold	Hold	NC	1
1E	CALTM	IF (cond) THEN CALL TM (data)	TM	PC + 1	Hold	Reset	PC + 1	Hold	Hold	NC	1
1F	CALTMN	IF (cond) THEN CALL TM (data), NESTED	TM	PC + 1	Load SREG	Reset	PC + 1	Hold	Hold	NC	1

INSTRUCTIONS DEPENDENT ON CREG

Opcode	Mnemonic	Assembler Statement	CREG = 0				CREG ≠ 0				Notes
			PC MUX	STACK	CREG	EQ FLAG	PC MUX	STACK	CREG	EQ FLAG	
08	LPPL	WHILE (CREG < > 0) LOOP TO PL (data)	PC + 1	Hold	Hold	NC	PL	Hold	DEC	Reset	2
0A	LPPLN	WHILE (CREG < > 0) LOOP TO PL (data), ELSE NEST	PC + 1	Hold	Load SREG	NC	PL	Hold	DEC	Reset	2
0B	GOTOPLZ	IF (CREG = 0) THEN GOTO PL (data)	PL	Hold	Hold	Reset	PC + 1	Hold	Hold	NC	3
0C	DECPL	WHILE (CREG < > 0) WAIT ELSE LOAD PL (data)	PC + 1	Hold	Load PL	NC	PC	Hold	DEC	NC	
0E	DECTM	WHILE (CREG < > 0) WAIT ELSE LOAD TM (data)	PC + 1	Hold	Load TM	NC	PC	Hold	DEC	NC	

INSTRUCTIONS DEPENDENT ON TEST CONDITION AND CREG VALUE

				Condition Pass				Condition Fail				
Opcode	Mnemonic	Assembler Statement	CREG Content	PC MUX	STACK	CREG	EQ FLAG	PC MUX	STACK	CREG	EQ FLAG	Notes
1B	DECGOPL	IF (cond) THEN GOTO PL (data) ELSE WHILE (CREG < > 0) WAIT	≠0	PL	Hold	Hold	Reset	PC	Hold	DEC	NC	1
			= 0	PL	Hold	Hold	Reset	PC + 1	Hold	Hold	NC	

UNCONDITIONAL INSTRUCTIONS

Opcode	Mnemonic	Assembler Statement	PC MUX	STACK	CREG	EQ FLAG	Notes
0D	CONT	CONTINUE	PC + 1	Hold	Hold	NC	
10-13 (Binary 100XX)	CMP	CMP TM (data) TO PL (data)	PC + 1	Hold	Hold	Set	4

Key:		
	PC	= Program Counter
	SREG	= Stack Register
	CREG	= Counter Register
	PL	= Pipeline (data) Field
	TM	= Test Inputs Masked by PL (data) Field
	DEC	= Decrement
	NC	= No Change

Notes: 1. If COND = EQ and condition PASSES, reset EQ flag.
2. If COND = EQ and CREG ≠ 0, reset EQ flag.
3. If COND = EQ and CREG = 0, reset EQ flag.
4. Set EQ flag if CONST field = T*M.

PROGRAMMING

The Am29PL131 FPC is programmed and verified using a simple algorithm that is almost identical to that used for AMD's Programmable Array Logic family. The internal programmable array of the Am29PL131 is organized as a 64-word by 28-bit PROM. The fuse to be programmed is selected by its address (1 of 64), the byte at that address (1 of 4), and the bit in the byte (1 of 8, 1 of 4 for byte 3). Control of programming and verifying is accomplished by applying a simple sequence of voltages on two control pins (CLK and CC).

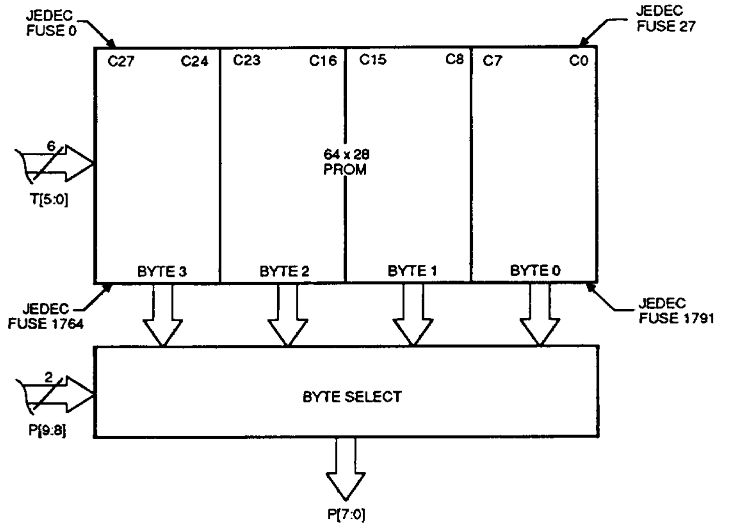
The fuse address is selected using a full decode of the T[5 : 0] inputs, where T[5] is the MSB and T[0] the LSB. The 1-of-4 byte addressing is done on the P[9] (MSB) and P[8] (LSB) outputs. The bit selection is done one output at a time by applying the programming voltage (V_{OP}) to the respective output pin P[7:0]. A graphic representation of the fuse array organization for programming, with fuse numbering compatible to the JEDEC standard programmable logic transfer format, is shown in Figure 2.

The complete program and verify cycle timing is shown in the programming waveform. A programming sequence is initiated by raising the CLK pin to V_{HH}. This places the device in the program mode and disables the output pins so that they may be used as fuse addressing inputs. The next step is to address the fuse to be blown as previously stated. Note that bit selection, with V_{OP}, should follow address and byte selection. Raising the CC pin to V_{HH} initiates programming and lowering V_{OP} terminates programming. Lowering the CLK pin to a TTL LOW level places the device in the fuse verification mode by enabling the programming outputs, P[7 : 0]. Following a clock pulse, the fuse may be verified on the same output that the bit selection was performed. Using this scheme, fuses can be verified in parallel as a byte if desired. The verification mode is terminated by lowering the CC pin back to a normal TTL level.

Programming Yield

AMD programmable logic devices have been designed to ensure extremely high programming yields (> 98%). To help ensure that a part was correctly programmed once the programming sequence is completed, the entire fuse array should be reverified at both LOW and HIGH V_{CC} (V_{CCL} and V_{CCH}). Reverification can be accomplished in a verification-only mode (CC at V_{HH}) by reading the outputs in parallel. This verification cycle checks that the array fuses have been blown correctly and can be sensed under varying conditions by the outputs.

AMD programmable logic devices contain extra fuses and circuitry so that before shipping it can be verified that all PROM fuses can be programmed. Additional circuitry is included for pre-shipment testing of the logic portion of the device. These added features assure high programming yields and correct logic operation.



JEDEC FUSE NUMBER = 28 (FUSE ADDRESS) + 8 (2 − BYTE) + (7 − BIT) + 4 FOR BYTE 0 − 2
= 28 (FUSE ADDRESS) + (3 − BIT) FOR BYTE 3, BIT 0 − 3 ONLY

Byte Select		
Byte	**P[9]**	**P[8]**
0	H	L
1	H	H
2	L	L
3	L	H

Bit Select for Byte 0-2									Bit Select for Byte 3								
Bit	**P[7]**	**P[6]**	**P[5]**	**P[4]**	**P[3]**	**P[2]**	**P[1]**	**P[0]**	**Bit**	**P[7]**	**P[6]**	**P[5]**	**P[4]**	**P[3]**	**P[2]**	**P[1]**	**P[0]**
0	L	L	L	L	L	L	L	H	0	L	L	L	L	L	L	L	H
1	L	L	L	L	L	L	H	L	1	L	L	L	L	L	L	H	L
2	L	L	L	L	L	H	L	L	2	L	L	L	L	L	H	L	L
3	L	L	L	L	H	L	L	L	3	L	L	L	L	H	L	L	L
4	L	L	L	H	L	L	L	L									
5	L	L	H	L	L	L	L	L									
6	L	H	L	L	L	L	L	L									
7	H	L	L	L	L	L	L	L									

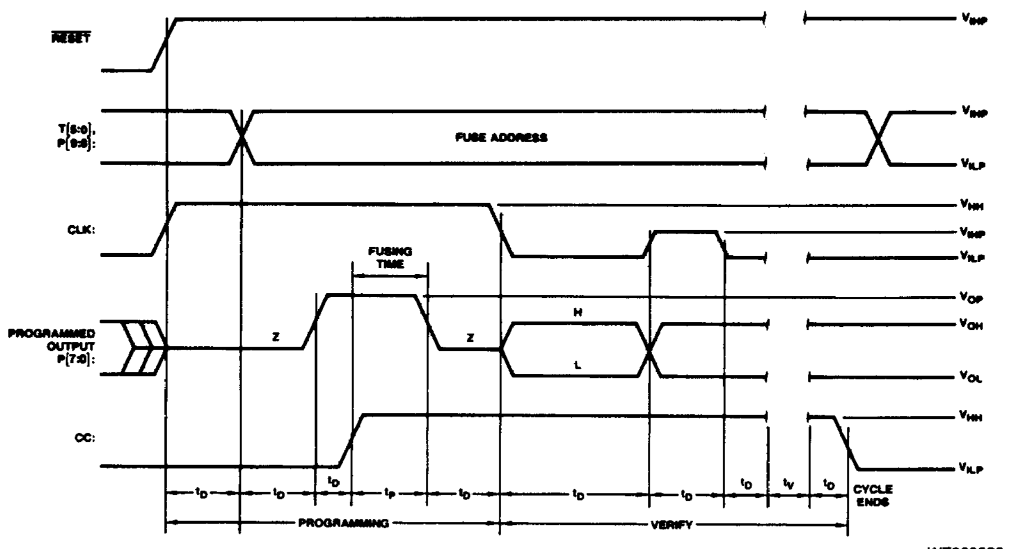
Column Decode				
0	C0	C8	C16	C24
1	C1	C9	C17	C25
2	C2	C10	C18	C26
3	C3	C11	C19	C27
4	C4	C12	C20	
5	C5	C13	C21	
6	C6	C14	C22	
7	C7	C15	C23	

Figure 2. Programming Configuration

Fuse Address Decode						
Fuse Address	T[5]	T[4]	T[3]	T[2]	T[1]	T[0]
0	L	L	L	L	L	L
1	L	L	L	L	L	H
2	L	L	L	L	H	L
3	L	L	L	L	H	H
4	L	L	L	H	L	L
5	L	L	L	H	L	H
6	L	L	L	H	H	L
7	L	L	L	H	H	H
8	L	L	H	L	L	L
9	L	L	H	L	L	H
10	L	L	H	L	H	L
11	L	L	H	L	H	H
12	L	L	H	H	L	L
13	L	L	H	H	L	H
14	L	L	H	H	H	L
15	L	L	H	H	H	H
16	L	H	L	L	L	L
17	L	H	L	L	L	H
18	L	H	L	L	H	L
19	L	H	L	L	H	H
20	L	H	L	H	L	L
21	L	H	L	H	L	H
22	L	H	L	H	H	L
23	L	H	L	H	H	H
24	L	H	H	L	L	L
25	L	H	H	L	L	H
26	L	H	H	L	H	L
27	L	H	H	L	H	H
28	L	H	H	H	L	L
29	L	H	H	H	L	H
30	L	H	H	H	H	L
31	L	H	H	H	H	H
32	H	L	L	L	L	L
33	H	L	L	L	L	H
34	H	L	L	L	H	L
35	H	L	L	L	H	H
36	H	L	L	H	L	L
37	H	L	L	H	L	H
38	H	L	L	H	H	L
39	H	L	L	H	H	H
40	H	L	H	L	L	L
41	H	L	H	L	L	H
42	H	L	H	L	H	L
43	H	L	H	L	H	H
44	H	L	H	H	L	L
45	H	L	H	H	L	H
46	H	L	H	H	H	L
47	H	L	H	H	H	H
48	H	H	L	L	L	L
49	H	H	L	L	L	H
50	H	H	L	L	H	L
51	H	H	L	L	H	H
52	H	H	L	H	L	L
53	H	H	L	H	L	H
54	H	H	L	H	H	L
55	H	H	L	H	H	H
56	H	H	H	L	L	L
57	H	H	H	L	L	H
58	H	H	H	L	H	L
59	H	H	H	L	H	H
60	H	H	H	H	L	L
61	H	H	H	H	L	H
62	H	H	H	H	H	L
63	H	H	H	H	H	H

PROGRAMMING PARAMETERS $T_A = 25°C$

Parameter Symbol	Parameter Description		Min.	Typ.	Max.	Unit
V_{HH}	Control Pin Extra-HIGH Level	CC @ 5 – 10 mA	15.5	16	16.5	V
		CLK@ 5 – 10 mA	15.5	16	16.5	
V_{OP}	Program Voltage, P[7:0] @ 15 – 200 mA		19.5	20	20.5	V
V_{IHP}	Input HIGH Level During Programming and Verify		2.4	5	5.5	V
V_{ILP}	Input LOW Level During Programming and Verify		0.0	0.2	0.5	V
V_{CCP}	V_{CC} During Programming @ I_{CC} = 425 mA		5	5.2	5.5	V
V_{CCL}	V_{CC} During First Pass Verification @ I_{CC} = 425 mA		4.5	4.7	5.0	V
V_{CCH}	V_{CC} During Second Pass Verification @ I_{CC} = 425 mA		5.4	5.7	6.0	V
V_{Blown}	Successful Blown Fuse Sense Level @ Output			0.3	0.5	V
dV_{OP}/dt	Rate of Output Voltage Change		20		250	V/µs
dV_{FE}/dt	Rate of Fusing Enable Voltage Change (CC Rising Edge)		100		1000	V/µs
t_P	Fusing Time First Attempt		40	50	100	µs
	Subsequent Attempts		4	5	10	ms
t_D	Delays Between Various Level Changes		100	200		ns
t_V	Period During which Output is Sensed for V_{Blown} Level		500			ns
V_{ONP}	Pull-Up Voltage on Outputs Not Being Programmed		$V_{CCP} - 0.3$	V_{CCP}	$V_{CCP} + 0.3$	V
R	Pull-Up Resistor on Outputs Not Being Programmed		1.9	2	2.1	kΩ
dV_{CLK}/dt	Rate of CLK Voltage Change (PROG)		40	50	250	V/µs
	Rate of CLK Voltage Change (VERIFY)		250	300	1000	



WF020636

Programming Waveforms

ABSOLUTE MAXIMUM RATINGS

Storage Temperature	−65 to +150°C
(Ambient) Temperature Under Bias	−55 to +125°C
Supply Voltage to Ground Potential (Pin 19 to Pin 6 and Pin 17) Continuous	−0.5 V to +7.0 V
DC Voltage Applied to Outputs (Except During Programming)	−0.5 V to +V_{CC} Max.
DC Voltage Applied to Outputs During Programming	21 V
DC Output Current, Into Outputs During Programming (Max Duration of 1 s)	200 mA
DC Input Voltage	−0.5 V to +5.5 V
DC Input Current	−30 mA to +5.0 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	
Ambient Temperature (T_A)	0 to +70°C
Supply Voltage (V_{CC})	+4.75 to +5.25 V
Military* (M) Devices	
Case Temperature (T_C)	−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_C = +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			Min.	Max.	Unit
V_{OH}	Output HIGH Voltage	V_{CC} = Min., V_{IN} = V_{IH} or V_{IL}	COM'L	I_{OH} = −3.0 mA	2.4		V
			MIL	I_{OH} = −1.0 mA			
V_{OL}	Output LOW Voltage	V_{CC} = Min., V_{IN} = V_{IL} or V_{IH}	COM'L	I_{OL} = 16 mA		0.50	V
			MIL	I_{OL} = 12 [illegible]			
V_{IH} (Note 1)	Input HIGH Level	Guaranteed Input Logical HIGH Voltage for [illegible] In[illegible]ts			[illegible]		V
V_{IL} (Note 1)	Input LOW Level	Guaranteed Input Logical L[illegible] [illegible]ge fo[illegible] All Inpu[illegible]				0.8	V
I_{IL}	Input LOW Current	V_{CC} = Max. V_{IN} = 0.5 V	CLK			−2.0	mA
			All O[illegible] Inpu[illegible]			−0.50	
I_{IH}	Input HIGH Current	V_{C}[illegible] Ma[illegible] V_{IN} [illegible]	CLK			150	μA
			All [illegible]ther Inputs			25	
I_I	Input HIGH Current	[illegible] Max., V_{IN} = 5.5 V				1.0	mA
I_{SC}	Output Shor[illegible]	V_{CC} [illegible] Max + 0.5 V V_{OUT} = 0.5 V (Note 2)			−20	−80	mA
I_{CC}	Powe[illegible] Current	V_{CC} = Max.	COM'L	T_A = 0 to +70°C		330	mA
				T_A = +70°C		290	
			MIL	T_C = −55 to +125°C		360	
				T_C = +125°C		310	
V_I	Input Clamp Voltage	V_{CC} = Min, I_{IN} = −18 mA				−1.2	V
I_{OZH}	Output Leakage Current (Note 3)	V_{CC} = Max., V_{IL} = 0.8 V V_{IH} = 2.0 V	V_O = 2.4 V			100	μA
I_{OZL}			V_O = 0.5 V			−550	

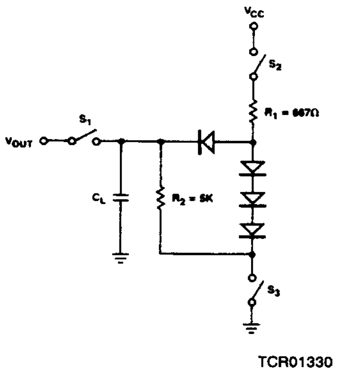
Notes: 1. These are absolute values with respect to device ground; and all overshoots due to system or tester noise are included.
2. Not more than one output should be tested at a time. Duration of the short-circuit test should not exceed one second. V_{OUT} = 0.5 V has been chosen to avoid test problems caused by tester ground degradation.
3. I/O pin leakage is the worst-case of I_{OZX} or I_{IX} (where X = H or L).

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

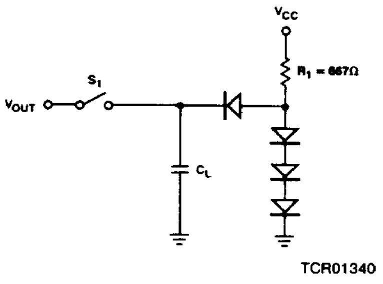
No.	Parameter Symbol	Parameter Description	Test Conditions	COM'L Min.	COM'L Max.	MIL Min.	MIL Max.	Unit
1	t_{PD}	CLK to P[11:0]	See Test Output Load Conditions		20		25	ns
2	t_S	T[5:0] to CLK		15		20		ns
3		CC to CLK		15		20		ns
4		$\overline{\text{RESET}}$ to CLK		[illegible]		35		ns
5	t_H	T[5:0] to CLK		[illegible]		0		ns
6		CC to CLK		0		0		ns
7		$\overline{\text{RESET}}$ to CLK		3		3		ns
8	t_{PZX}	CLK to P[15:8] Enable			30		35	ns
9	t_{PXZ}	CLK to P[15:8] Disable			30		35	ns
10	t_{PW}	CLK Pulse Width (HIGH and LOW)		25		30		ns
11	t_P	CLK Period (Note 1)		66.7		72		ns

Notes: 1. These parameters are measured indirectly on unprogrammed devices. They are determined as follows:
a. Measure delay from input CLK to PROM address out in test mode. This will measure the delay through the sequence logic.
b. Measure setup time from T[5:0] input through PROM test columns to pipeline register in verify test column mode. This will measure the delay through the PROM and register setup.
c. Measure delay from T[5:0] input to PROM address out in verify test column mode. This will measure the delay through the logic and P[11:0] outputs.
To calculate the desired parameter measurement, the following formula is used:
Measurement (a) + Measurement (b) – Measurement (c)
CLK PERIOD:
CLK (a) + (b) – (c) = CLK PERIOD
$\overline{\text{RESET}}$ to CLK Setup time:
$\overline{\text{RESET}}$ (a) + (b) – (c) = $\overline{\text{RESET}}$ to CLK Setup time

SWITCHING TEST CIRCUITS



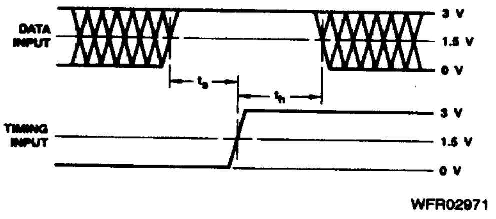
A. Three-State Outputs



B. Normal Outputs

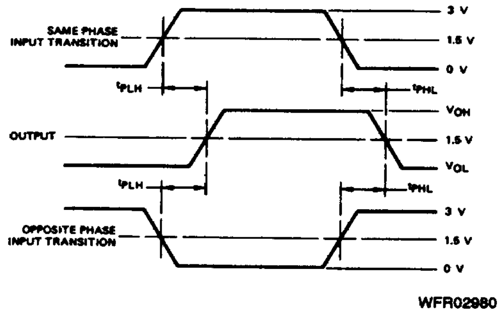
Notes: 1. CL = 50 pF includes scope probe, wiring, and stray capacitances without device in test fixture.
2. S_1, S_2, and S_3 are closed during function tests and all AC tests except output enable tests.
3. S_1 and S_3 are closed while S_2 is open for t_{PZH} test.
4. C_L = 5.0 pF for output disable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

SWITCHING TEST WAVEFORMS

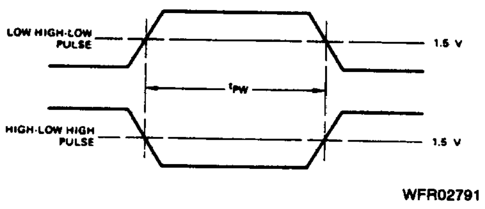


Setup, Hold, and Release Times

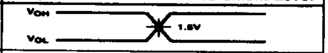
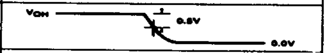
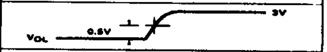
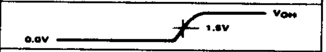
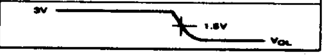
Notes: 1. Diagram shown for HIGH data only. Output transition may be opposite sense.
2. Cross-hatched area is don't care condition.



Propagation Delay



Pulse Width

Test	V_X	Output Waveform – Measurement Level
All t_{PD}s	5.0V	
t_{PHZ}	0.0V	
t_{PLZ}	5.0V	
t_{PZH}	0.0V	
t_{PZL}	5.0V	

WFR02680

Enable and Disable Times

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.

Note: Pulse generator for all pulses: Rate $\leq$ 1.0 MHz; $Z_0 = 50\ \Omega$: $t_r \leq 2.5$ ns.

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.
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 that 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 that varies from one type of tester to another, but is generally around 50 pF. This makes it impossible to make direct measurements of parameters that 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

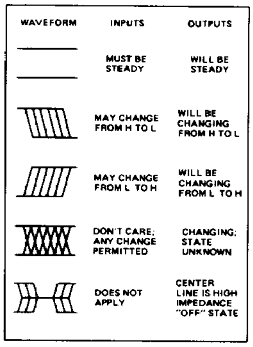
   Occasionally parameters are specified that cannot be measured directly on automatic testers because of tester limitations. Data input hold times often fall into this category. In these cases, the parameter in question is guaranteed by correlating these tests with other AC tests that have been performed. 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 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. 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 $V_O = 0$, $V_{CC} =$ Max., case.

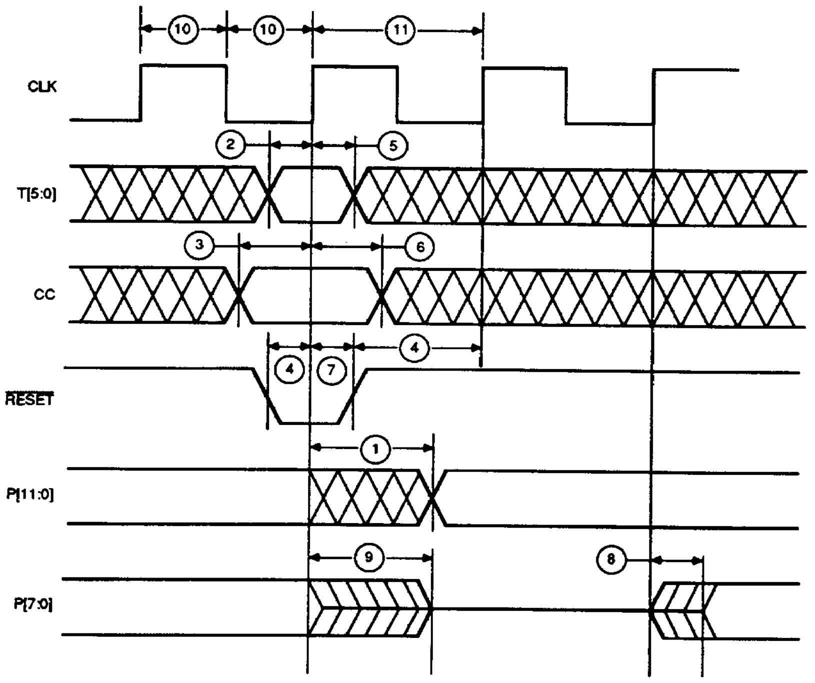
SWITCHING WAVEFORMS

KEY TO SWITCHING WAVEFORMS



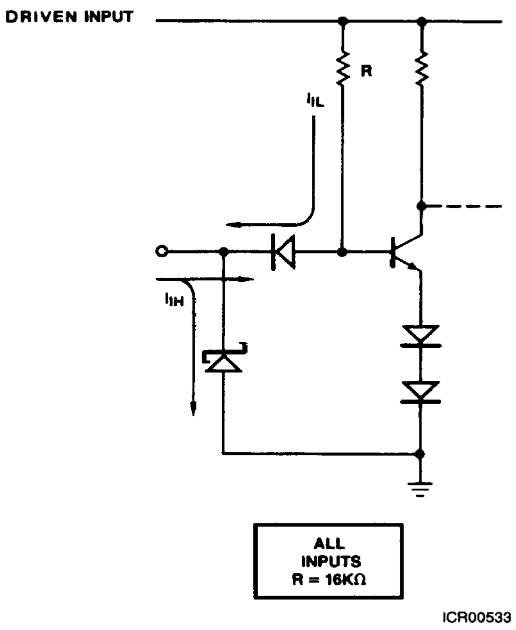
WAVEFORM	INPUTS	OUTPUTS
	MUST BE STEADY	WILL BE STEADY
	MAY CHANGE FROM H TO L	WILL BE CHANGING FROM H TO L
	MAY CHANGE FROM L TO H	WILL BE CHANGING FROM L TO H
	DON'T CARE; ANY CHANGE PERMITTED	CHANGING; STATE UNKNOWN
	DOES NOT APPLY	CENTER LINE IS HIGH IMPEDANCE "OFF" STATE

KS000010

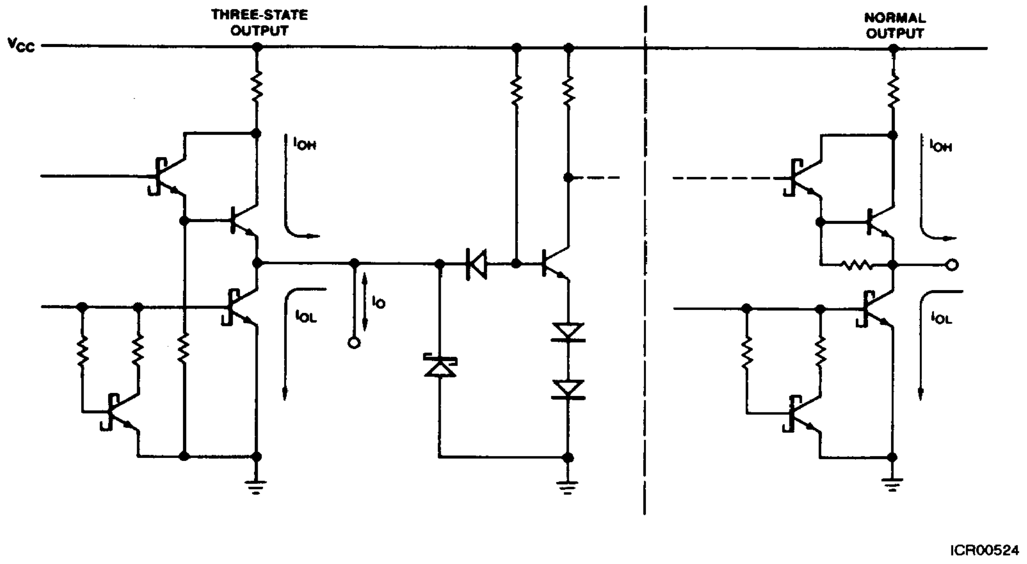


WF025471

INPUT/OUTPUT CIRCUIT DIAGRAMS



$C_O \cong 5.0$ pF, all inputs

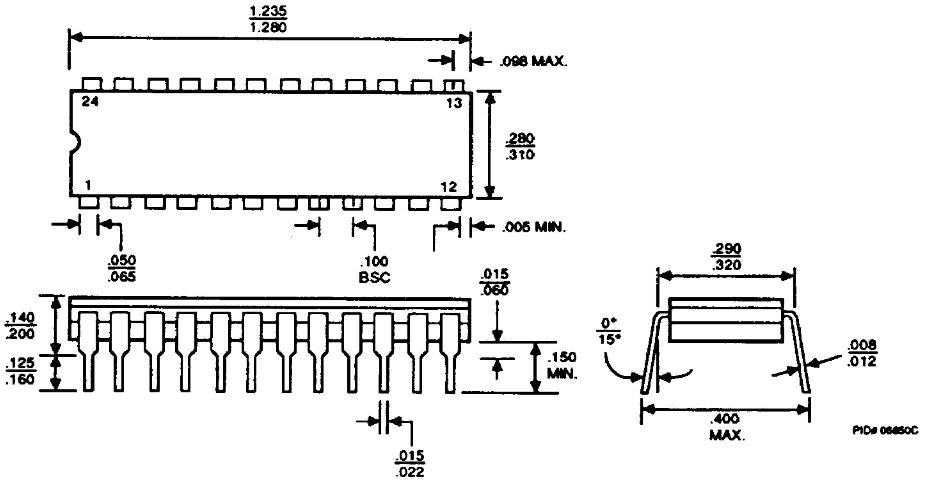


$C_O \cong 5.0$ pF, all outputs

NOTE: Actual current flow direction shown.

PHYSICAL DIMENSIONS*

CD3024



*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
San Jose	(408) 249-7766
Santa Clara	(408) 727-3270
Woodland Hills	(818) 992-4155
CANADA, Ontario,	
Kanata	(613) 592-0060
Willowdale	(416) 224-5193
COLORADO	(303) 741-2900
CONNECTICUT	(203) 264-7800
FLORIDA,	
Clearwater	(813) 530-9971
Ft Lauderdale	(305) 776-2001
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	(913) 451-3115
NEW JERSEY	(201) 299-0002
NEW YORK,	
Liverpool	(315) 457-5400
Poughkeepsie	(914) 471-8180
Woodbury	(516) 364-8020
NORTH CAROLINA	(919) 878-8111
OHIO	(614) 891-6455
Columbus	(614) 891-6455
Dayton	(513) 439-0470
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	(1) 49-75-10-10
	FAX	(1) 49-75-10-13
	TLX	263282
WEST 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	852-123-4276
	TLX	504260AMDAPHX
ITALY, Milano	TEL	(02) 3390541
		(02) 3533241
	FAX	(02) 3498000
	TLX	315286
JAPAN,		
Kanagawa	TEL	462-47-2911
	FAX	462-47-1729
Tokyo	TEL	(03) 345-8241
	FAX	(03) 342-5196
	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
NORWAY,		
Hovik	TEL	(02) 537810
	FAX	(02) 591959
	TLX	79079
SINGAPORE	TEL	65-2257544
	FAX	2246113
	TLX	RS55650 MMI RS
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	(0483) 756196
	TLX	859103

NORTH AMERICAN REPRESENTATIVES

CALIFORNIA	
I² INC	OEM (408) 988-3400
	DISTI (408) 498-6868
CANADA	
Burnaby, B.C.	
DAVETEK MARKETING	(604) 430-3680
Calgary, Alberta	
VITEL ELECTRONICS	(403) 278-5833
Kanata, Ontario	
VITEL ELECTRONICS	(613) 592-0090
Mississauga, Ontario	
VITAL ELECTRONICS	(416) 676-9720
Quebec	
VITEL ELECTRONICS	(514) 636-5951
IDAHO	
INTERMOUNTAIN TECH MKGT	(208) 888-6071
INDIANA	
ELECTRONIC MARKETING CONSULTANTS, INC.	(317) 253-1668
IOWA	
LORENZ SALES	(319) 377-4666
KANSAS	
LORENZ SALES	(913) 384-6556
KENTUCKY	
ELECTRONIC MARKETING CONSULTANTS, INC.	(317) 253-1668
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	
Centerville	
DOLFUSS ROOT & CO	(513) 433-6776
Columbus	
DOLFUSS ROOT & CO	(614) 885-4844
Strongsville	
DOLFUSS ROOT & CO	(216) 238-0300
PENNSYLVANIA	
DOLFUSS ROOT & CO	(412) 221-4420
UTAH	
R² 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
APPLICATIONS HOTLINE TOLL FREE: (800) 222-9323

© 1988 Advanced Micro Devices, Inc.
Printed in U.S.A. AIS-CP-15M-4/88-0

016252 ✓