68000 Interrupt Controller

Danesh Tavana

Commercial and industrial microprocessor-based systems consist of the basic block units of CPU, memory, and I/O devices. While executing instructions in memory, the CPU must somehow be interrupted to service requests from various I/O devices. The 68000 microprocessor is a powerful 16-bit processor which makes provisions for 256 different interrupt routines. A simple and cost-effective way of interfacing a peripheral's interrupt request signal to the CPU is through Programmable Array Logic. This paper introduces two ways of designing such an interface with PAL devices.

2175 Mission College Blvd. Santa Clara, CA 95054-1592 Tel: (408) 970-9700 TWX: 910-338-2376 TWX: 910-338-2374
Monolithic Memories

68000 Interrupt Controller

Danesh Tavana

Introduction

Commercial and industrial microprocessor-based systems typically consist of a processor interfaced with many peripherals which randomly require service. To fully utilize the processor's computing potential, sources of hardware interrupts must be used to free the processor from software routines. The 68000 16-bit microprocessor has 256 different exception-processing vectors which point to predetermined locations in the program memory space. There are 192 external user interrupt vectors and seven autovector interrupts. Table 1 shows the exception vector assignments. The interrupt structure of the 68000 will be discussed in more detail along with two methods of designing an interrupt controller using PAL devices (Programmable Array Logic).

Vector	Address			Assignment
Number(s)	Dec	Hex	Space	
0	0	000	SP	Reset: Initial SSP
–	4	004	SP	Reset: Initial PC
2	8	008	SD	Bus Error
3	12	00C	SD	Address Error
4	16	010	SD	Illegal Instruction
5	20	014	SD	Zero Divide
6	24	018	SD	CHK Instruction
7	28	01C	SD	TRAPV Instruction
8	32	020	SD	Privilege Violation
9	36	024	SD	Trace
10	40	028	SD	Line 1010 Emulator
11	44	02C	SD	Line 1111 Emulator
12*	48	030	SD	(Unassigned, reserved)
13*	52	034	SD	(Unassigned, reserved)
14*	56	038	SD	(Unassigned, reserved)
15	60	03C	SD	Uninitialized Interrupt Vector
16-23*	64	04C	SD	(Unassigned, reserved)
	95	05F		–
24	96	060	SD	Spurious Interrupt
25	100	064	SD	Level 1 Interrupt Autovector
26	104	068	SD	Level 2 Interrupt Autovector
27	108	06C	SD	Level 3 Interrupt Autovector
28	112	070	SD	Level 4 Interrupt Autovector
29	116	074	SD	Level 5 Interrupt Autovector
30	120	078	SD	Level 6 Interrupt Autovector
31	124	07C	SD	Level 7 Interrupt Autovector
32-47	128	080	SD	TRAP Instruction Vectors
	191	0BF		–
48-63*	192	0C0	SD	(Unassigned, reserved)
	255	0FF		–
64-255	256	100	SD	User Interrupt Vectors
	1023	3FF		–

*Vector numbers 12, 13, 14, 16 through 23 and 48 through 63 are reserved for future enhancements by Motorola. No user peripheral devices should be assigned these numbers.

Table 1. Exception Vector Assignment

68000 Exception Processing and Pin Description

The interrupt structure of the 68000 can grant up to 256 types of interrupt requests. These interrupts, or exceptions, are generated either by external or internal causes and are serviced by directing the flow of program to an exception processing routine. Table 1 lists these exceptions and their respective address in memory. Exception vectors are locations in memory where the processor fetches the address of a routine or subprogram which will service that exception. The exception vector is either internally or externally generated depending on the cause of the interrupt. The externally generated exceptions are interrupts which are placed on the interrupt control pins ($\overline{IPL2}$, $\overline{IPL1}$ and $\overline{IPL0}$), bus errors, and reset requests. The interrupts placed on control pins $\overline{IPL2}$-$\overline{IPL0}$ are requests from peripheral devices. The internally-generated exceptions come from instructions, address errors, or tracing. These different types of exceptions and their priorities are shown in Table 2. We will focus on group 1 interrupt exceptions.

Group	Exception	Processing
0	Reset Bus Error Address Error	Exception processing begins within two clock cycles.
1	Trace Interrupt Illegal Privilege	Exception processing begins before the next instruction.
2	TRAP, TRAPV, CHK, Zero Divide	Exception processing is started by normal instruction execution.

Table 2. Exception Grouping and Priority

The pinout of the 68000 is shown in Figure 1. The three interrupt control pins ($\overline{IPL2}$, $\overline{IPL1}$, $\overline{IPL0}$) are asynchronous active low inputs which indicate the encoded priority level of

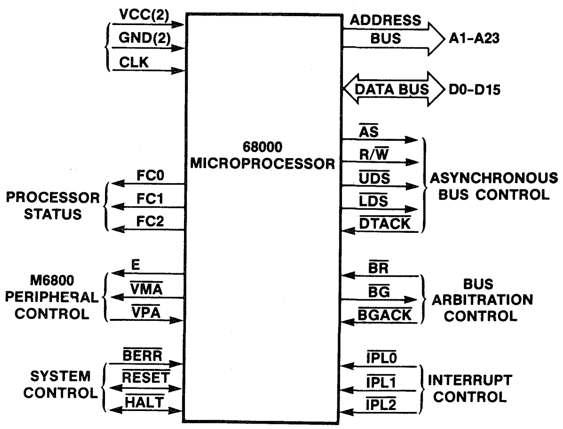


Figure 1.

the device requesting an interrupt. The least significant bit is $\overline{\mathrm{IPL0}}$ and the most significant bit is $\overline{\mathrm{IPL2}}$. Level seven, ($\overline{\mathrm{IPL2}}$, $\overline{\mathrm{IPL1}}$, $\overline{\mathrm{IPL0}}$) = (000), has the highest priority, while level zero, (111), indicates that no interrupts are requested. All lower or equal priority interrupts are masked during the interrupt service routine of the present interrupting device. There are two ways by which a peripheral device can interrupt the normal flow of program: autovectoring or external vector number generation. The function code pins (FC2, FC1, and FC0) all become high during an interrupt acknowledge cycle. The interrupt acknowledge cycle always follows an interrupt request only after the present instruction cycle is completed. Thus interrupt acknowlege cycles come in between the instruction cycles and no information is lost.

The signals which are used during an external interrupt request are listed below with a description of their behavior and function.

ADDRESS BUS: The 24-bit address bus holds the address of the data to be accessed. During an interrupt acknowledge cycle the lower three bits (A3 A2 A1) hold the encoded level of the interrupt being serviced. If a level 5 interrupt is being serviced then (A3 A2 A1) will be (101) respectively.

DATA BUS: The lower eight bits of this 16-bit data bus must contain the vector number during a user interrupt acknowledge cycle. If an autovector routine is in process then the data bus is ignored.

$\overline{\mathbf{AS}}$: The processor asserts address strobe anytime there is valid data on the address bus. It remains asserted for as long as the address is valid.

R/$\overline{\mathbf{W}}$: This signal defines the data bus transfer as a read or a write cycle. During an interrupt acknowledge cycle the processor is in a read mode.

$\overline{\mathbf{UDS}}$, $\overline{\mathbf{LDS}}$: The upper and lower data strobes are asserted when the processor is in a read or write instruction cycle. Upper stobe enables the most significant byte of data while the lower stobe enables the least significant.

$\overline{\mathbf{DTACK}}$: Data transfer acknowledge is an externally generated signal which tells the processor that valid data is present on the data bus. If $\overline{\mathrm{DTACK}}$ is not asserted before the falling edge of S4 (S4 is the fourth CPU clock state in a seven-state instruction cycle) then wait states are introduced.

$\overline{\mathbf{IPL2}}$-$\overline{\mathbf{IPL0}}$: The three interrupt control pins are inputs which contain the encoded priority level of the external interrupting device. Note that these are active low pins where (000) is level seven encoded.

FC2-FC0: Function code output pins indicate the processor's status. When they are all high the processor is in an interrupt acknowledge cycle.

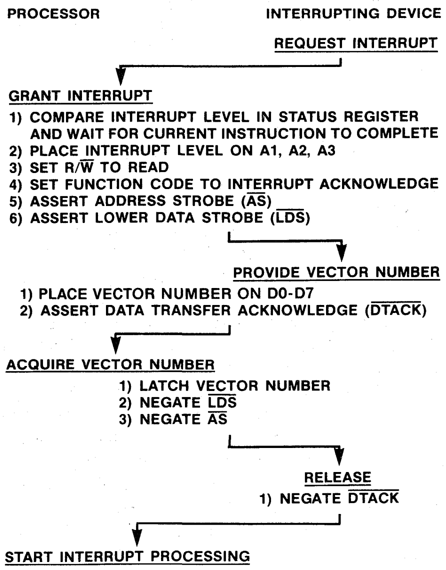
E, $\overline{\mathbf{VMA}}$, $\overline{\mathbf{VPA}}$: Enable, valid memory address, and valid peripheral address are the standard 6800 family signals. $\overline{\mathrm{VPA}}$ is also used when autovectoring.

The following two design examples use PAL units as a simple and cost-effective interrupt controller interface to the 68000. The first introduces external vector generation while the second shows how autovectoring can be done on the 68000.

Prioritized Individually Vectored Interrupt (Method 1)

As shown in the interrupt acknowledge sequence flow chart and timing diagram (Figure 2), a peripheral device must interrupt the processor by placing the encoded level of priority on the interrupt control pins ($\overline{\mathrm{IPL2}}$-$\overline{\mathrm{IPL0}}$). The processor then grants the interrupt after the completion of the current instruction cycle. The encoded priority level of the interrupt is placed on address bus bits A3-A1 during the interrupt acknowledge cycle. The status code lines (FC2-FC0) become high, indicating an interrupt acknowledge cycle. Once the lower data strobe ($\overline{\mathrm{LDS}}$) is asserted the external device must place an 8-bit vector on the least significant byte of the data bus. This data is latched in state S7.

INTERRUPT ACKNOWLEDGE SEQUENCE
FLOW CHART


PROCESSOR
INTERRUPTING DEVICE
REQUEST INTERRUPT
GRANT INTERRUPT
1) COMPARE INTERRUPT LEVEL IN STATUS REGISTER AND WAIT FOR CURRENT INSTRUCTION TO COMPLETE
2) PLACE INTERRUPT LEVEL ON A1, A2, A3
3) SET R/W̅ TO READ
4) SET FUNCTION CODE TO INTERRUPT ACKNOWLEDGE
5) ASSERT ADDRESS STROBE (AS̅)
6) ASSERT LOWER DATA STROBE (LDS̅)
PROVIDE VECTOR NUMBER
1) PLACE VECTOR NUMBER ON D0-D7
2) ASSERT DATA TRANSFER ACKNOWLEDGE (DTACK̅)
ACQUIRE VECTOR NUMBER
1) LATCH VECTOR NUMBER
2) NEGATE LDS̅
3) NEGATE AS̅
RELEASE
1) NEGATE DTACK̅
START INTERRUPT PROCESSING

INTERRUPT ACKNOWLEDGE SEQUENCE
TIMING DIAGRAM

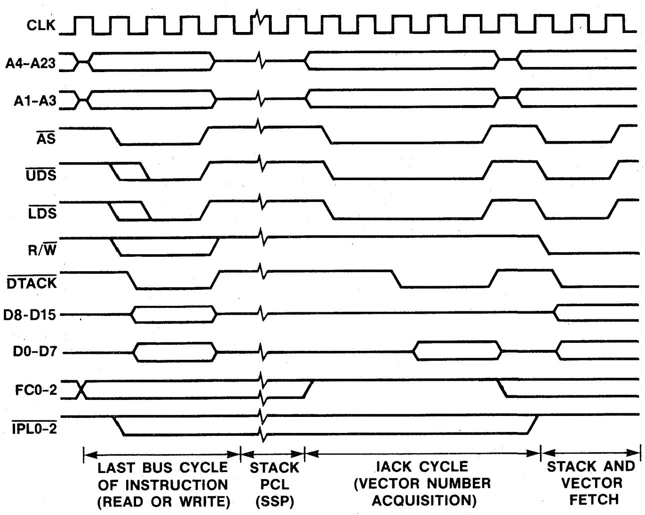


Figure 2.

As shown on Table 1 vectors 64-255 are assigned to the user interrupt vector numbers. These vectors must be generated externally and placed on the data bus during an interrupt acknowledge cycle. In our interrupt controller we arbitrarily choose vectors 249-255 as the vectors assigned to the interrupting peripheral devices. Table 3a shows this assignment and how the vector can be decoded from the address bits A1-A3.

Priority level	Vector assigned	Data								Address		
Device #	Decimal #	D7	D6	D5	D4	D3	D2	D1	D0	A3	A2	A1
1 (low priority)	249	1	1	1	1	1	0	0	1	0	0	1
2	250	1	1	1	1	1	0	1	0	0	1	0
3	251	1	1	1	1	1	0	1	1	0	1	1
4	252	1	1	1	1	1	1	0	0	1	0	0
5	253	1	1	1	1	1	1	0	1	1	0	1
6	254	1	1	1	1	1	1	1	0	1	1	0
7 (high priority)	255	1	1	1	1	1	1	1	1	1	1	1

Table 3a.

The two PALs used on this design example generate the interrupt vectors and all the necessary control signals. The various signals and their implementation on the PALs are explained below.

$\overline{\text{INT7}}$-$\overline{\text{INT0}}$: Any of the seven peripheral devices can request an interrupt by asserting one of these inputs. The interrupt must remain asserted until acknowledged by the CPU.

FC2-FC0 and $\overline{\text{AS}}$: Function code or processor status code become logical high during an interrupt acknowledge cycle. Address strobe is asserted anytime valid address is on the bus. $\overline{\text{DTACK}}$ and Data output control are decoded from these four outputs of the 68000.

A1-A3: The three least significant bits of the address bus contain the encoded level of the interrupting device. These signals are used in generating the vector number which is to be put on the data bus.

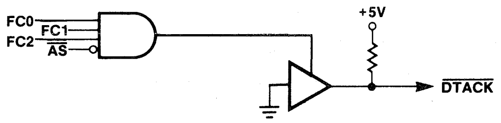
$\overline{\text{RESET}}$: Reset is an input which clears all outputs, and is used for system initialization.

$\overline{\text{IPL2}}$-$\overline{\text{IPL0}}$: These PAL outputs are active low synchronous signals which interface directly to the CPU. They represent the encoded level of the highest priority interrupt that is requested. Table 3b shows the truth table of our priority encoder implemented on a PAL.

Interrupt request input from peripheral							Encoded int. level		
$\overline{\text{INT7}}$	$\overline{\text{INT6}}$	$\overline{\text{INT5}}$	$\overline{\text{INT4}}$	$\overline{\text{INT3}}$	$\overline{\text{INT2}}$	$\overline{\text{INT1}}$	$\overline{\text{IPL2}}$	$\overline{\text{IPL1}}$	$\overline{\text{IPL0}}$
0	X	X	X	X	X	X	0	0	0
1	0	X	X	X	X	X	0	0	1
1	1	0	X	X	X	X	0	1	0
1	1	1	0	X	X	X	0	1	1
1	1	1	1	0	X	X	1	0	0
1	1	1	1	1	0	X	1	0	1
1	1	1	1	1	1	0	1	1	0
1	1	1	1	1	1	1	1	1	1

Table 3b.

$\overline{\text{DTACK}}$: Data transfer acknowledge must be asserted by outside circuitry during a data transfer operation. The logic diagram shown below illustrates how $\overline{\text{DTACK}}$ is derived from address strobe and processor status signals.



D7-D0: The three least significant bits of the data output can be decoded straight from the address bits A3, A2 and A1. That is D2=A3, D1=A2 and D0=A1. The other five bits of data can be held high with pull-up resistors. Outputs of the three data bits become enabled by using the same scheme which enables the $\overline{\text{DTACK}}$ output.

$\overline{\text{INTACK7}}$-$\overline{\text{INTACK1}}$: Only one of these signals will be asserted during the interrupt acknowledge cycle. This signal feeds back into the interrupting device to tell that device that its interrupt has been acknowledged. We can use the 3-bit addresses to decode these signals as shown in Table 3a.

			$\overline{\text{INTACK}}$						
A3	A2	A1	7	6	5	4	3	2	1
0	0	0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	1	1	0
0	1	0	1	1	1	1	1	0	1
0	1	1	1	1	1	1	0	1	1
1	0	0	1	1	1	0	1	1	1
1	0	1	1	1	0	1	1	1	1
1	1	0	1	0	1	1	1	1	1
1	1	1	0	1	1	1	1	1	1

Table 3c.

The logic diagram of the controller is shown in Figure 3. The controller can operate without any wait states at the fastest CPU clock rate of 12.5 MHz as shown in the timing diagram of Figure 4. The appendix contains the fuse plot which translates into a logic schematic for these two PALs.

Auto-vectored Interrupts (Method 2)

The seven autovector interrupts are assigned vector numbers 25 through 31. Interrupts are requested by placing the encoded level of the request on the interrupt control pins ($\overline{\text{IPL2}}$-$\overline{\text{IPL0}}$). The processor responds to this request by placing the requested level in the processor's status register and by inhibiting all requests of lower priority. When the current instruction cycle is completed an interrupt acknowledge cycle takes place. If Valid Peripheral Address ($\overline{\text{VPA}}$) is asserted before the falling edge of S4 then an autovector routine takes place. The data or the vector number is generated internally depending on the priority level of the interrupt request; the vector assigned is shown on the table at the beginning of this report. The autovector timing diagram and a very simple and practical interrupt controller implemented on a PAL is shown in Figure 5. The PAL design specifications are included in the appendix. Note that $\overline{\text{VPA}}$ is generated by enabling the PAL output when FC2-FC1 are high and $\overline{\text{AS}}$ is asserted. The appendix contains the fuse plots for this PAL design. Note that the fuse plots translate directly to a logic diagram.

Interrupt Controller Logic Diagram External Vector Generation

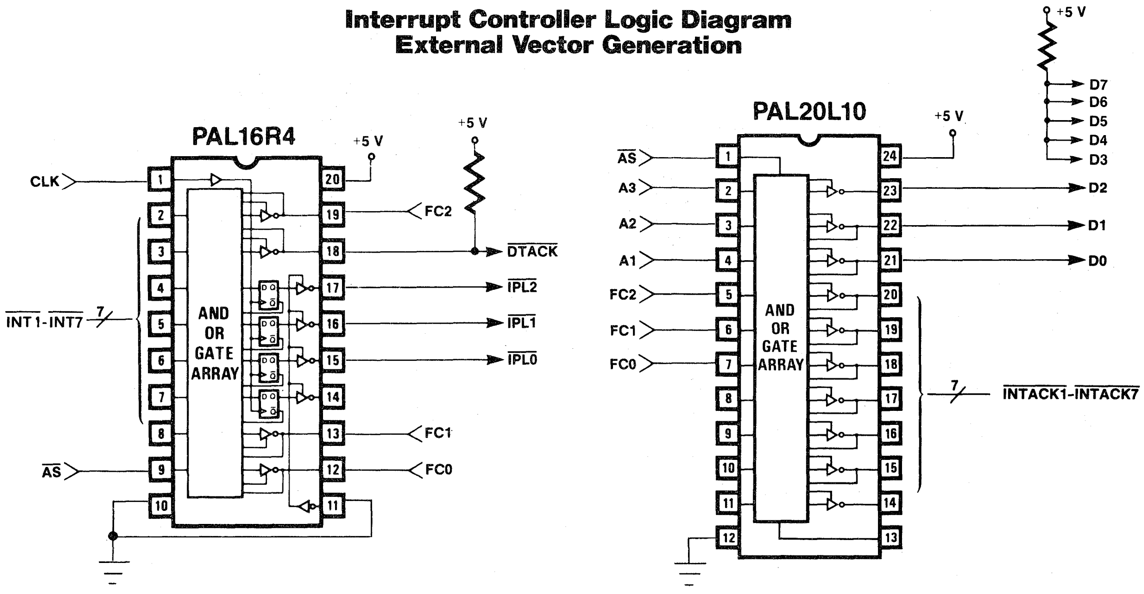
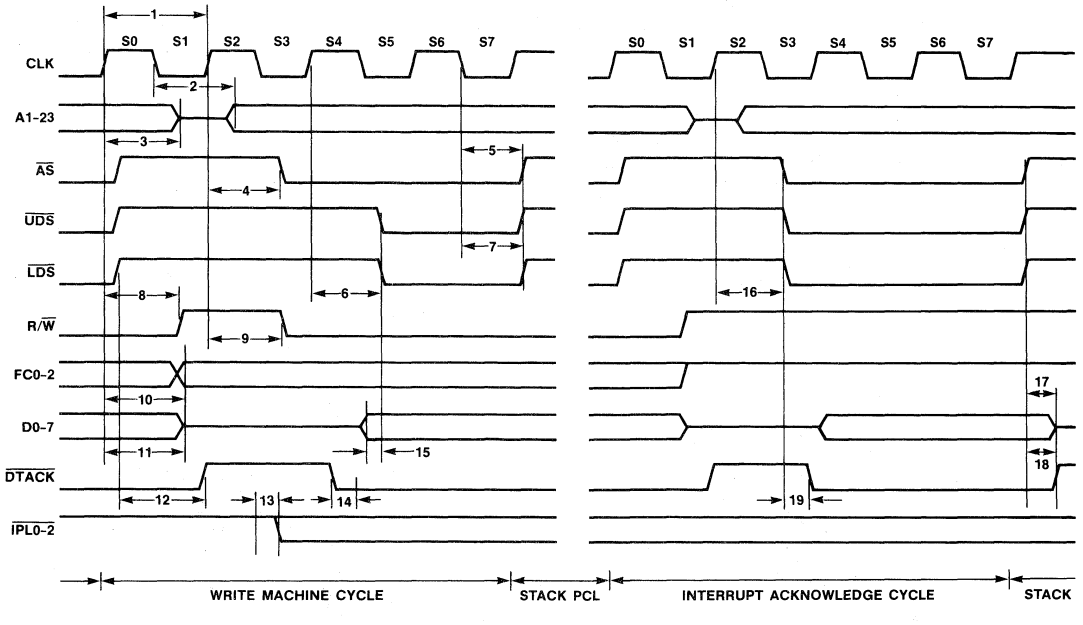
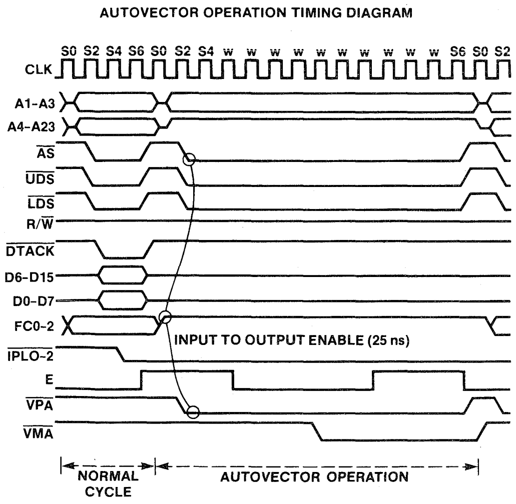


Figure 3.



1	CLOCK PERIOD (80 ns)
2	CLOCK LOW TO ADDRESS (55 ns)
3	CLOCK HIGH TO ADDRESS Hi-Z (60 ns)
4	CLOCK HIGH TO AS LOW (55 ns)
5	CLOCK LOW TO AS HIGH (50 ns)
6	CLOCK HIGH TO DS LOW (55 ns)
7	CLOCK LOW TO DS HIGH (50 ns)
8	CLOCK HIGH TO R/W HIGH (60 ns)
9	CLOCK HIGH TO R/W LOW (60 ns)
10	CLOCK HIGH TO FC VALID (55 ns)
11	CLOCK HIGH TO DATA Hi-Z (60 ns)
12	AS HIGH TO DTACK HIGH (70 ns)
13	PAL's CLOCK TO OUT (25 ns)
14	MINIMUM SETUP TIME (20 ns)
15	DATA VALID TO DS LOW (15 ns)
16	CLOCK HIGH TO AS, DS LOW (55 ns)
17	PAL's INPUT TO Hi-Z (25 ns)
18	PAL's INPUT TO Hi-Z (25 ns)
19	PAL's INPUT TO Hi-Z (25 ns)

Figure 4.


AUTOVECTOR OPERATION TIMING DIAGRAM
S0 S2 S4 S6 S0 S2 S4 w w w w w w w w w w S6 S0 S2
CLK
A1-A3
A4-A23
AS
UDS
LDS
R/W
DTACK
D6-D15
D0-D7
FC0-2
INPUT TO OUTPUT ENABLE (25 ns)
IPL0-2
E
VPA
VMA
NORMAL CYCLE
AUTOVECTOR OPERATION

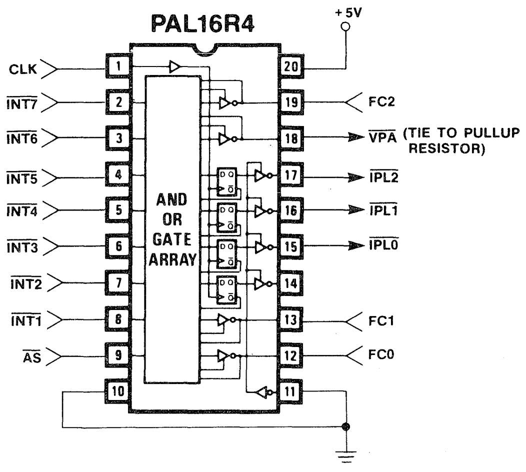


Figure 5.

Conclusions

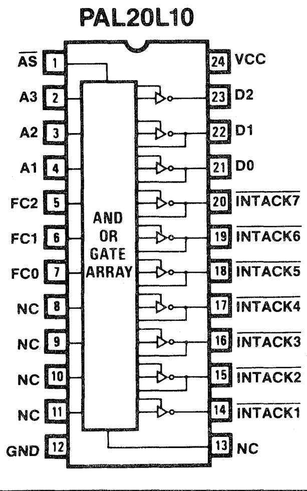
We have just seen how to implement an interrupt controller circuit using one or two PAL devices. This circuit can operate at the maximum operating frequency of the 68000 which is 12.5 MHz. Since the most critical timing of the circuit is the PAL unit's input to Hi-Z, we see that we can operate the circuit with a wide spectrum of frequencies and with slower PAL devices. To guarantee operation, timing spec. No. 16 and No. 19 on Figure 4 must add up to assert DTACK 20 ns prior to falling edge of S4. Thus 55 ns + (PAL input delay to Hi-Z) should be less than, or equal to, 100 ns for a 12.5 MHz clock. We see that this qualifies fast, regular and half-power PAL devices for this circuit. At slower CPU clock rates even the quarter-power PAL devices may be used.

References

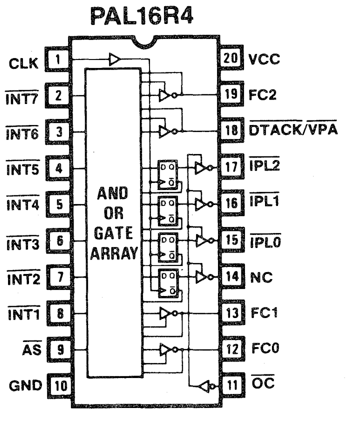
1. J. Birkner, V. Coli, *"PAL® Programmable Array Logic Handbook,"* Monolithic Memories Inc.
2. MC68000 Data Sheet, Motorola Semiconductors. Austin, Texas 78721.
3. Rex Davis, *"Prioritized Individually Vectored Interrupts for Multiple Peripheral Systems with the MC68000."* Motorola Application Note AN-819

Appendix

Interrupt Controller



Interrupt Controller

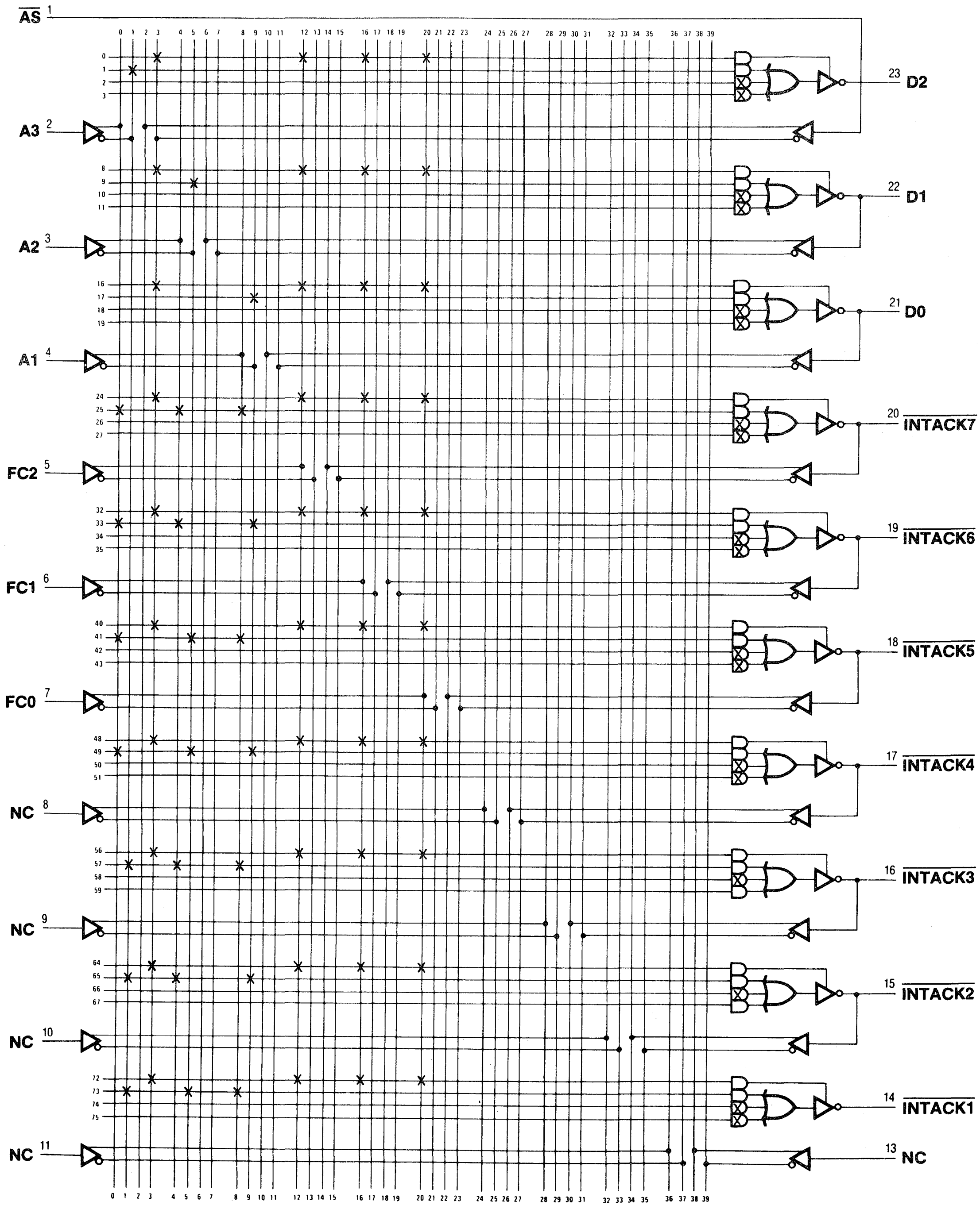


```
PAL20L10                                                  PAL DESIGN SPECIFICATIONS
INTC.DAT                                                  DANESH TAVANA     9/25/82
INTERRUPT CONTROLLER
MMI, SUNNYVALE
/AS    A3        A2        A1        FC2       FC1       FC0       NC     NC NC NC GND
 NC /INTACK1 /INTACK2 /INTACK3 /INTACK4 /INTACK5 /INTACK6 /INTACK7 D0 D1 D2 VCC


IF (FC2* FC1* FC0* AS) /D2 =/A3

IF (FC2* FC1* FC0* AS) /D1 =/A2

IF (FC2* FC1* FC0* AS) /D0 =/A1

IF (FC2* FC1* FC0* AS) INTACK7 = A3* A2* A1

IF (FC2* FC1* FC0* AS) INTACK6 = A3* A2*/A1

IF (FC2* FC1* FC0* AS) INTACK5 = A3*/A2* A1

IF (FC2* FC1* FC0* AS) INTACK4 = A3*/A2*/A1

IF (FC2* FC1* FC0* AS) INTACK3 =/A3* A2* A1

IF (FC2* FC1* FC0* AS) INTACK2 =/A3* A2*/A1

IF (FC2* FC1* FC0* AS) INTACK1 =/A3*/A2* A1



FUNCTION TABLE
/AS FC2 FC1 FC0 A3 A2 A1 D2 D1 D0
/INTACK7 /INTACK6 /INTACK5 /INTACK4 /INTACK3 /INTACK2 /INTACK1
;                                         /  /  /  /  /  /  /
;                                         I  I  I  I  I  I  I
;                                         N  N  N  N  N  N  N
;                                         T  T  T  T  T  T  T
;                                         A  A  A  A  A  A  A
;/  F  F  F                               C  C  C  C  C  C  C
;A  C  C  C    A  A  A    D  D  D         K  K  K  K  K  K  K
;S  2  1  0    3  2  1    2  1  0         7  6  5  4  3  2  1       COMMENTS
-------------------------------------------------------------------------------
 L  H  H  H    L  L  L    L  L  L         H  H  H  H  H  H  H       NO INTERRUPT
 L  H  H  H    L  L  H    L  L  H         H  H  H  H  H  H  L       LEVEL 1
 L  H  H  H    L  H  L    L  H  L         H  H  H  H  H  L  H       LEVEL 2
 L  H  H  H    L  H  H    L  H  H         H  H  H  H  L  H  H       LEVEL 3
 L  H  H  H    H  L  L    H  L  L         H  H  H  L  H  H  H       LEVEL 4
 L  H  H  H    H  L  H    H  L  H         H  H  L  H  H  H  H       LEVEL 5
 L  H  H  H    H  H  L    H  H  L         H  L  H  H  H  H  H       LEVEL 6
 L  H  H  H    H  H  H    H  H  H         L  H  H  H  H  H  H       LEVEL 7
 H  X  X  X    X  X  X    Z  Z  Z         Z  Z  Z  Z  Z  Z  Z       OUTPUT HI-Z
 L  H  H  L    X  X  X    Z  Z  Z         Z  Z  Z  Z  Z  Z  Z       OUTPUT HI-Z
 L  H  L  H    X  X  X    Z  Z  Z         Z  Z  Z  Z  Z  Z  Z       OUTPUT HI-Z
 L  L  H  H    X  X  X    Z  Z  Z         Z  Z  Z  Z  Z  Z  Z       OUTPUT HI-Z
-------------------------------------------------------------------------------
```

7

PAL1 — Interrupt Controller

Logic Diagram PAL20L10

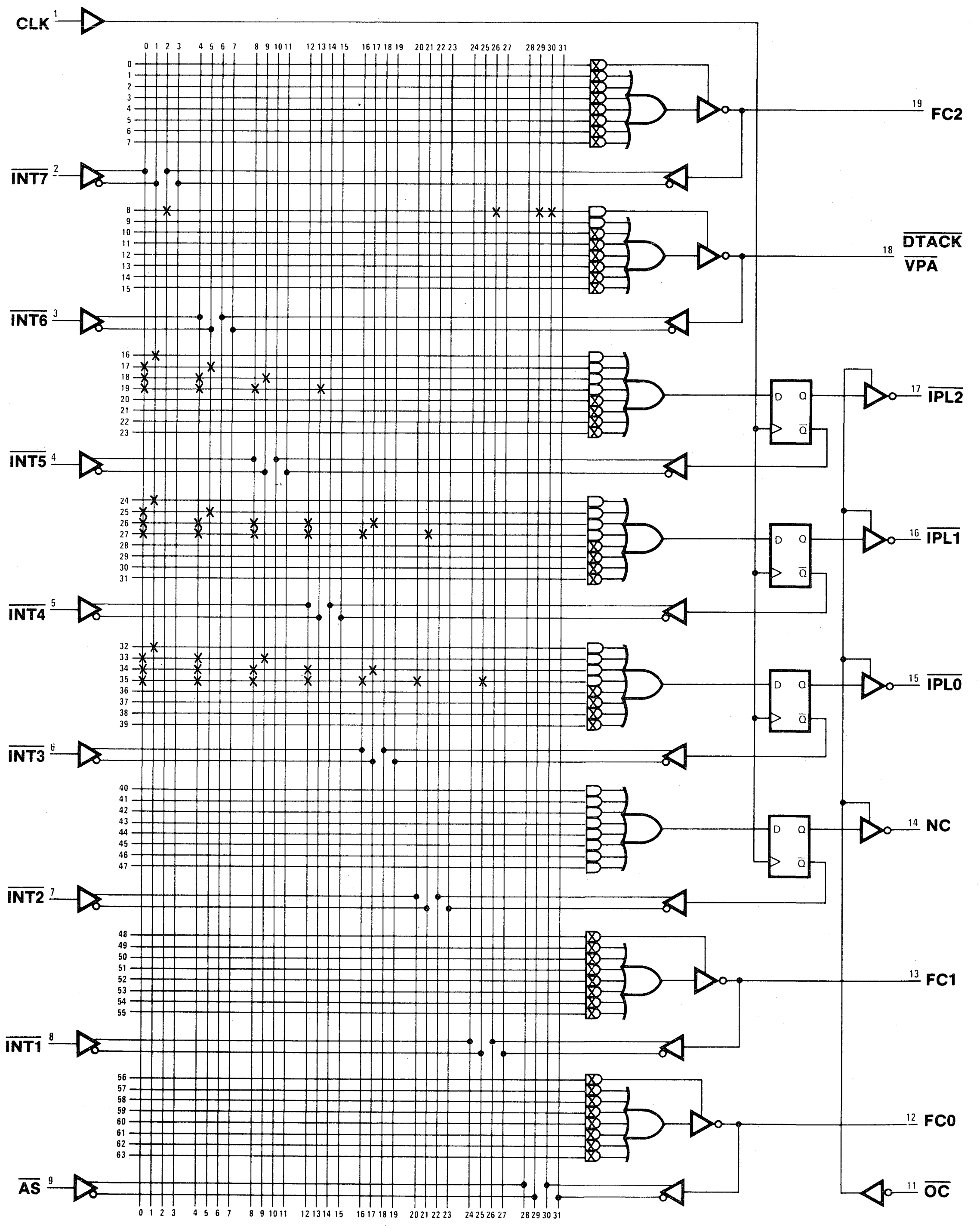


```
PAL16R4                                          PAL DESIGN SPECIFICATIONS
INTA.DAT                                         DANESH TAVANA
INTERRUPT CONTROLLER
MMI SUNNYVALE, CA
CLK /INT7 /INT6 /INT5 /INT4 /INT3 /INT2 /INT1  /AS GND
/OC  FC0   FC1    NC  /IPL0 /IPL1 /IPL2 /DTACK FC2 VCC

IF (FC2* FC1* FC0* AS) DTACK = VCC                    ;ASSERT IF OUTPUT ENABLE

IPL2 :=  INT7                                         ;PRIORITY ENCODED BIT
      + /INT7* INT6
      + /INT7*/INT6* INT5
      + /INT7*/INT6*/INT5* INT4

IPL1 :=  INT7                                         ;PRIORITY ENCODED BIT
      + /INT7* INT6
      + /INT7*/INT6*/INT5*/INT4* INT3
      + /INT7*/INT6*/INT5*/INT4*/INT3* INT2

IPL0 :=  INT7                                         ;PRIORITY ENCODED BIT
      + /INT7*/INT6* INT5
      + /INT7*/INT6*/INT5*/INT4* INT3
      + /INT7*/INT6*/INT5*/INT4*/INT3*/INT2* INT1
```

```
FUNCTION TABLE

CLK /INT7 /INT6 /INT5 /INT4 /INT3 /INT2 /INT1
FC2 FC1 FC0 /AS /IPL2 /IPL1 /IPL0 /DTACK
;                                       /
;   / / / / / / /               / / /   D
;   I I I I I I I               I I I   T
;C  N N N N N N N    F F F /    P P P   A
;L  T T T T T T T    C C C A    L L L   C
;K  7 6 5 4 3 2 1    2 1 0 S    2 1 0   K           COMMENTS
--------------------------------------------------------------------------------
 C  L X X X X X X    H H H L    L L L   L           ASSERT /DTACK
 C  H L X X X X X    H H H H    L L H   Z           INTERRUPT LEVEL 7
 C  H H L X X X X    L L H L    L H L   Z           INTERRUPT LEVEL 6
 C  H H H L X X X    L H L L    L H H   Z           INTERRUPT LEVEL 5
 C  H H H H L X X    L H H L    H L L   Z           INTERRUPT LEVEL 4
 C  H H H H H L X    H L L L    H L H   Z           INTERRUPT LEVEL 3
 C  H H H H H H L    H L H L    H H L   Z           INTERRUPT LEVEL 2
 C  H H H H H H H    H H L L    H H H   Z           INTERRUPT LEVEL 1
--------------------------------------------------------------------------------
```

PAL2 — Interrupt Controller

Logic Diagram PAL16R4


CLK
INT7
INT6
INT5
INT4
INT3
INT2
INT1
AS
FC2
DTACK
VPA
IPL2
IPL1
IPL0
NC
FC1
FC0
OC
D
Q