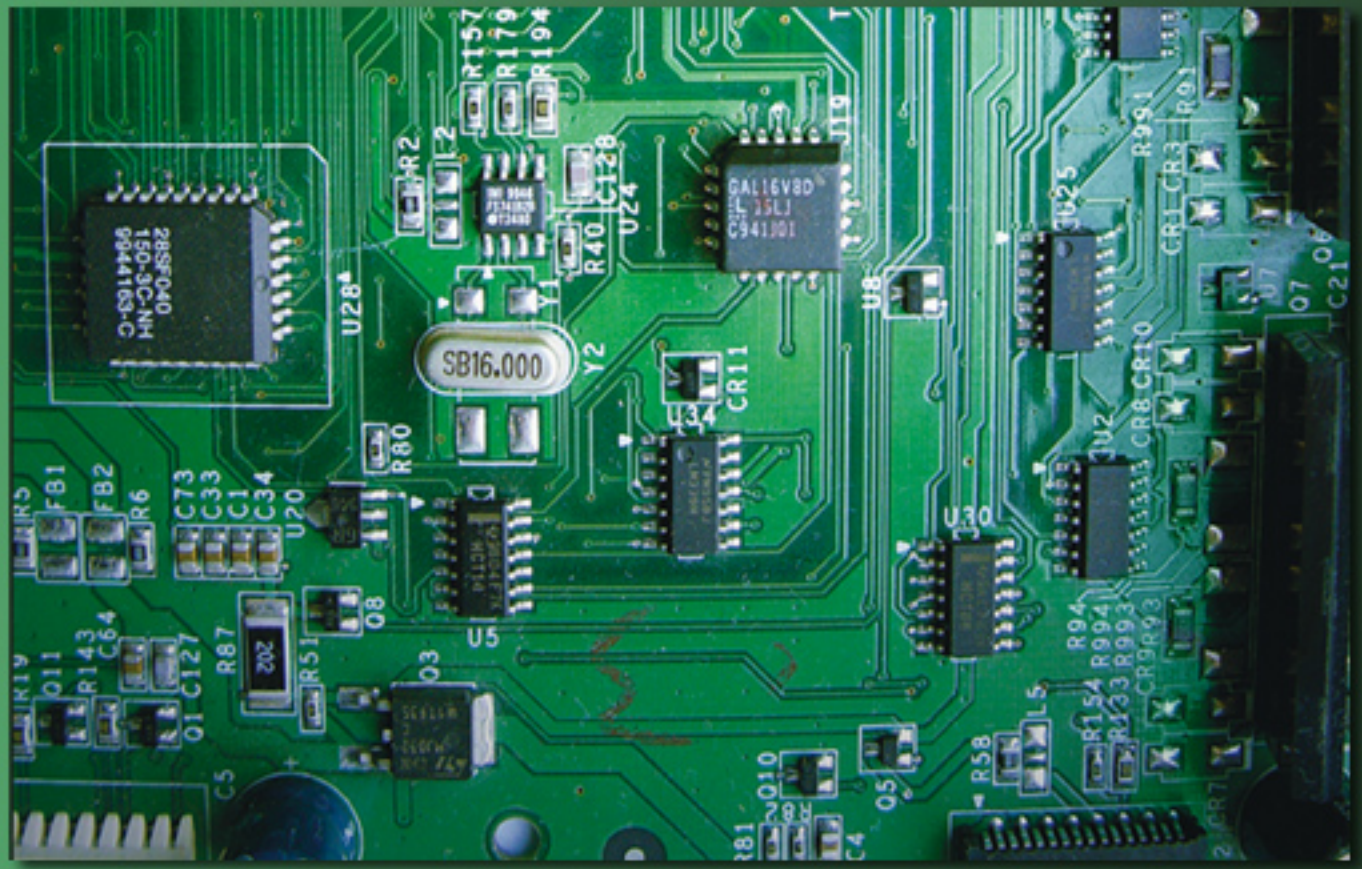
The Intel Microprocessors

8086/8088, 80186/80188, 80286, 80386, 80486 Pentium, Pentium Pro Processor, Pentium II, Pentium 4, and Core2 with 64-bit Extensions

Architecture, Programming, and Interfacing



EIGHTH EDITION

Barry B. Brey

PEARSON

Chapter 12: Interrupts

Introduction

- In this chapter, the coverage of basic I/O and programmable peripheral interfaces is expanded by examining a technique called interrupt-processed I/O.
- An interrupt is a hardware-initiated procedure that interrupts whatever program is currently executing.
- This chapter provides examples and a detailed explanation of the interrupt structure of the entire Intel family of microprocessors.

Chapter Objectives

Upon completion of this chapter, you will be able to:

- Explain the interrupt structure of the Intel family of microprocessors.
- Explain the operation of software interrupt instructions INT, INTO, INT 3, and BOUND.
- Explain how the interrupt enable flag bit (IF) modifies the interrupt structure.
- Describe the function of the trap interrupt flag bit (TF) and the operation of trap-generated tracing.

Chapter Objectives (*cont.*)

Upon completion of this chapter, you will be able to:

- Develop interrupt-service procedures that control lower-speed, external peripheral devices.
- Expand the interrupt structure of the microprocessor by using the 82S9A programmable interrupt controller and other techniques.
- Explain the purpose and operation of a real-time clock.

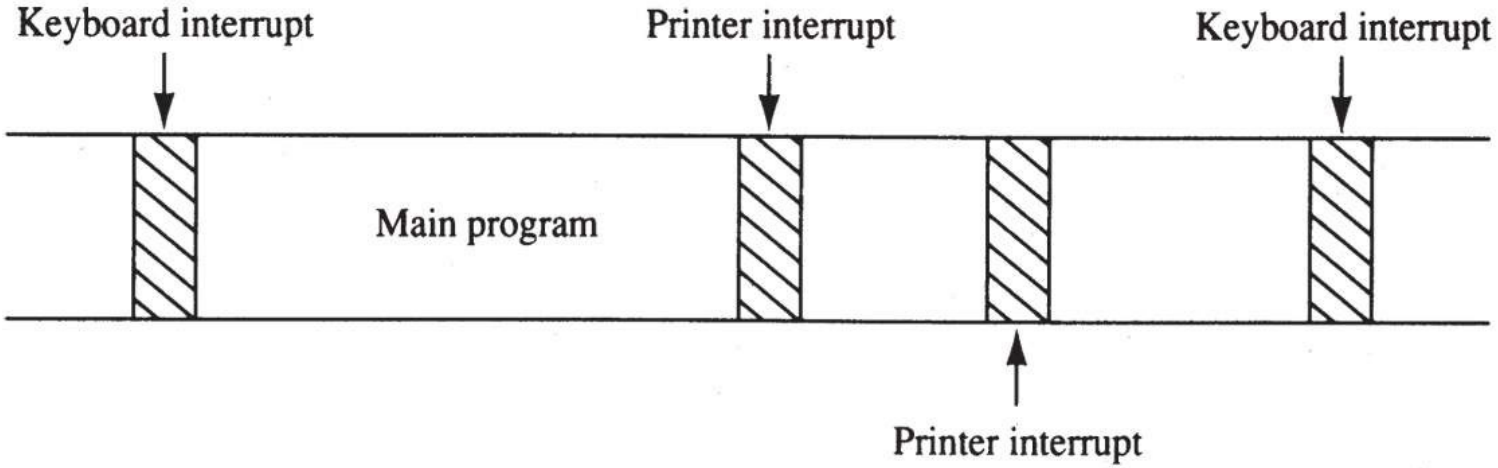
12–1 BASIC INTERRUPT PROCESSING

- This section discusses the function of an interrupt in a microprocessor-based system.
- Structure and features of interrupts available to Intel microprocessors.

The Purpose of Interrupts

- Interrupts are useful when interfacing I/O devices at relatively low data transfer rates, such as keyboard inputs, as discussed in Chapter 11.
- Interrupt processing allows the processor to execute other software while the keyboard operator is thinking about what to type next.
- When a key is pressed, the keyboard encoder debounces the switch and puts out one pulse that interrupts the microprocessor.

Figure 12–1 A time line that indicates interrupt usage in a typical system.



- a time line shows typing on a keyboard, a printer removing data from memory, and a program executing
- the keyboard interrupt service procedure, called by the keyboard interrupt, and the printer interrupt service procedure each take little time to execute

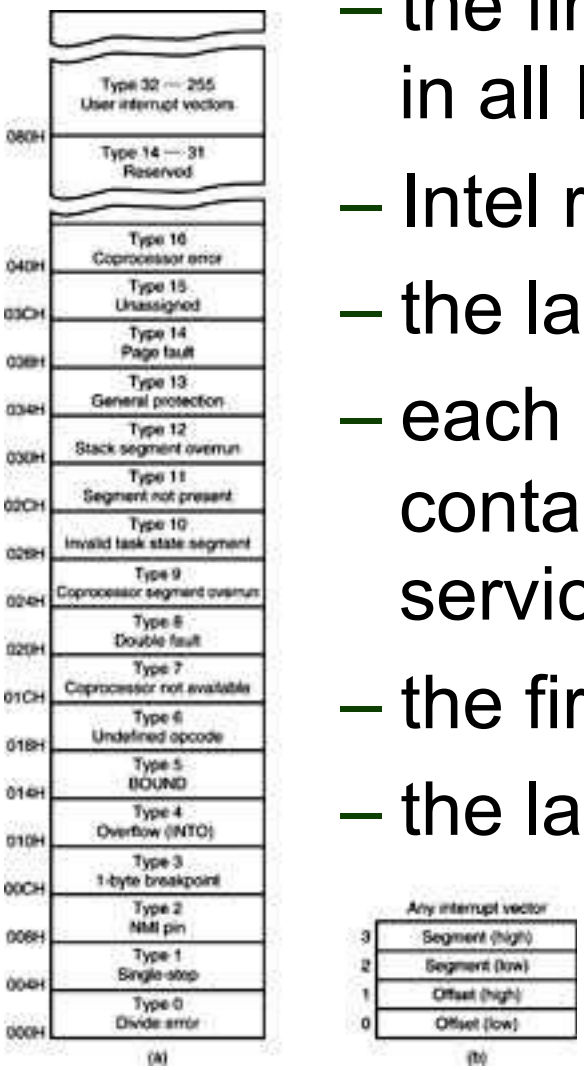
Interrupts

- Intel processors include two hardware pins (INTR and NMI) that request interrupts…
- And one hardware pin ($\overline{\text{INTA}}$) to acknowledge the interrupt requested through INTR.
- The processor also has software interrupts INT, INTO, INT 3, and BOUND.
- Flag bits IF (interrupt flag) and TF (trap flag), are also used with the interrupt structure and special return instruction IRET
  - IRETD in the 80386, 80486, or Pentium

Interrupt Vectors

- Interrupt vectors and the vector table are crucial to an understanding of hardware and software interrupts.
- The **interrupt vector table** is located in the first 1024 bytes of memory at addresses 000000H–0003FFH.
  - contains 256 different four-byte interrupt vectors
- An interrupt vector contains the address (segment and offset) of the interrupt service procedure.

Figure 12–2 (a) The interrupt vector table for the microprocessor and (b) the contents of an interrupt vector.



- the first five interrupt vectors are identical in all Intel processors
- Intel reserves the first 32 interrupt vectors
- the last 224 vectors are user-available
- each is four bytes long in real mode and contains the starting address of the interrupt service procedure.
- the first two bytes contain the offset address
- the last two contain the segment address

Intel Dedicated Interrupts

- **Type 0**
  The **divide error** whenever the result from a division overflows or an attempt is made to divide by zero.
- **Type 1**
  Single-step or trap occurs after execution of each instruction if the trap (TF) flag bit is set.
  - upon accepting this interrupt, TF bit is cleared so the interrupt service procedure executes at full speed

- **Type 2**
  The **non-maskable interrupt** occurs when a logic 1 is placed on the NMI input pin to the microprocessor.
  - non-maskable—it cannot be disabled
- **Type 3**
  A special one-byte instruction (INT 3) that uses this vector to access its interrupt-service procedure.
  - often used to store a breakpoint in a program for debugging

- **Type 4**
  Overflow is a special vector used with the INTO instruction. The INTO instruction interrupts the program if an overflow condition exists.
  - as reflected by the overflow flag (OF)

- **Type 13**
  The **general protection fault** occurs for most protection violations in 80286–Core2 in protected mode system.

  These errors occur in Windows as general protection faults.

  A list of these protection violations follows.

- **Type 13** protection violations (*cont.*)
  - (a) Descriptor table limit exceeded
  - (b) Privilege rules violated
  - (c) Invalid descriptor segment type loaded
  - (d) Write to code segment that is protected
  - (e) Read from execute-only code segment
  - (f) Write to read-only data segment
  - (g) Segment limit exceeded
  - (h) CPL = IOPL when executing CTS, HLT, LGDT, LIDT, LLDT, LMSW, or LTR
  - (i) CPL > IOPL when executing CLI, IN, INS, LOCK, OUT, OUTS, and STI

- **Type 14**
  Page fault interrupts occur for any page fault memory or code access in 80386, 80486, and Pentium–Core2 processors.
- **Type 16**
  Coprocessor error takes effect when a coprocessor error (ERROR = 0) occurs for ESCape or WAIT instructions for 80386, 80486, and Pentium–Core2 only.

Interrupt Instructions: BOUND, INTO, INT, INT 3, and IRET

- Five software interrupt instructions are available to the microprocessor:
- INT and INT 3 are very similar.
- BOUND and INTO are conditional.
- IRET is a special interrupt return instruction.

- BOUND has two operands, and compares a register with two words of memory data.
- INTO checks or tests the overflow flag (O).
  - If O = 1, INTO calls the procedure whose address is stored in interrupt vector type 4
  - If O = 0, INTO performs no operation and the next sequential program instruction executes
- The INT *n* instruction calls the interrupt service procedure at the address represented in vector number *n*.

- INT 3 instruction is often used as a breakpoint-interrupt because it is easy to insert a one-byte instruction into a program.
  - breakpoints are often used to debug software
- The IRET instruction is a special return instruction used to return for both software and hardware interrupts.
  - much like a far RET, it retrieves the return address from the stack

Operation of a Real Mode Interrupt

- When the processor completes executing the current instruction, it determines whether an interrupt is active by checking:
  - (1) instruction executions
  - (2) single-step
  - (3) NMI
  - (4) coprocessor segment overrun
  - (5) INTR
  - (6) INT instructions in the order presented

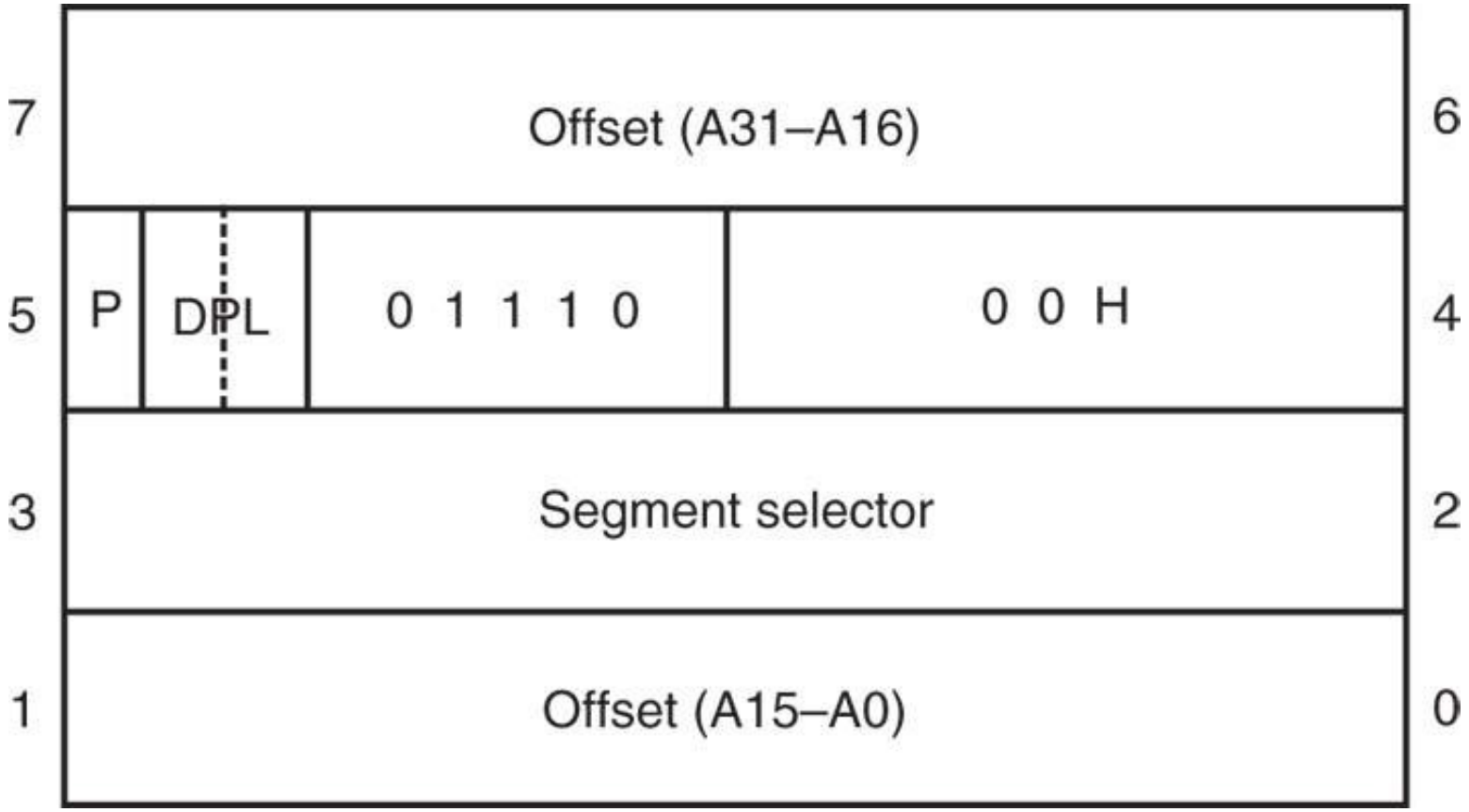
- If one or more are present:
  - 1. Flag register contents are pushed on the stack
  - 2. Interrupt (IF) & trap (TF) flags clear, disabling the INTR pin and trap or single-step feature
  - 3. Contents of the code segment register (CS) are pushed onto the stack
  - 4. Contents of the instruction pointer (IP) are pushed onto the stack
  - 5. Interrupt vector contents are fetched and placed into IP and CS so the next instruction executes at the interrupt service procedure addressed by the vector

Operation of a Protected Mode Interrupt

- In protected mode, interrupts have the same assignments as real mode.
  - the interrupt vector table is different
- In place of interrupt vectors, protected mode uses a set of 256 interrupt descriptors stored in an interrupt descriptor table (IDT).
  - the table is 256 × 8 (2K) bytes long
  - each descriptor contains eight bytes

- The interrupt descriptor table is located at any memory location in the system by the interrupt descriptor table address register (IDTR).
- Each IDT entry contains the address of the interrupt service procedure
  - in the form of a segment selector and a 32-bit offset address
  - also contains the P bit (present) and DPL bits to describe the privilege level of the interrupt
- Fig 12–3 shows interrupt descriptor contents.

Figure 12–3 The protected mode interrupt descriptor.



Interrupt Flag Bits

- The interrupt flag (IF) and the trap flag (TF) are both cleared after the contents of the flag register are stacked during an interrupt.
- the contents of the flag register and the location of IF and TF are shown here
  - when IF is set, it *allows* the INTR pin to cause an interrupt
  - when IF is cleared, it *prevents* the INTR pin from causing an interrupt

– when TF = 1, it causes a trap interrupt (type 1) to occur after each instruction executes
– Trap is often called a *single-step*
– when TF = 0, normal program execution occurs
– the interrupt flag is set and cleared by the STI and CLI instructions, respectively
– the contents of the flag register and the location of IF and TF are shown here

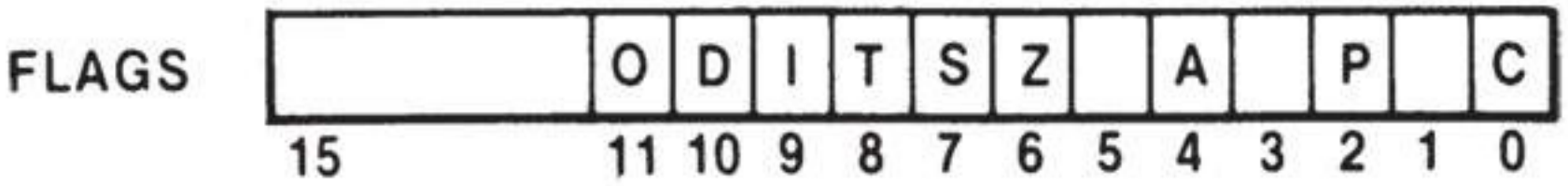


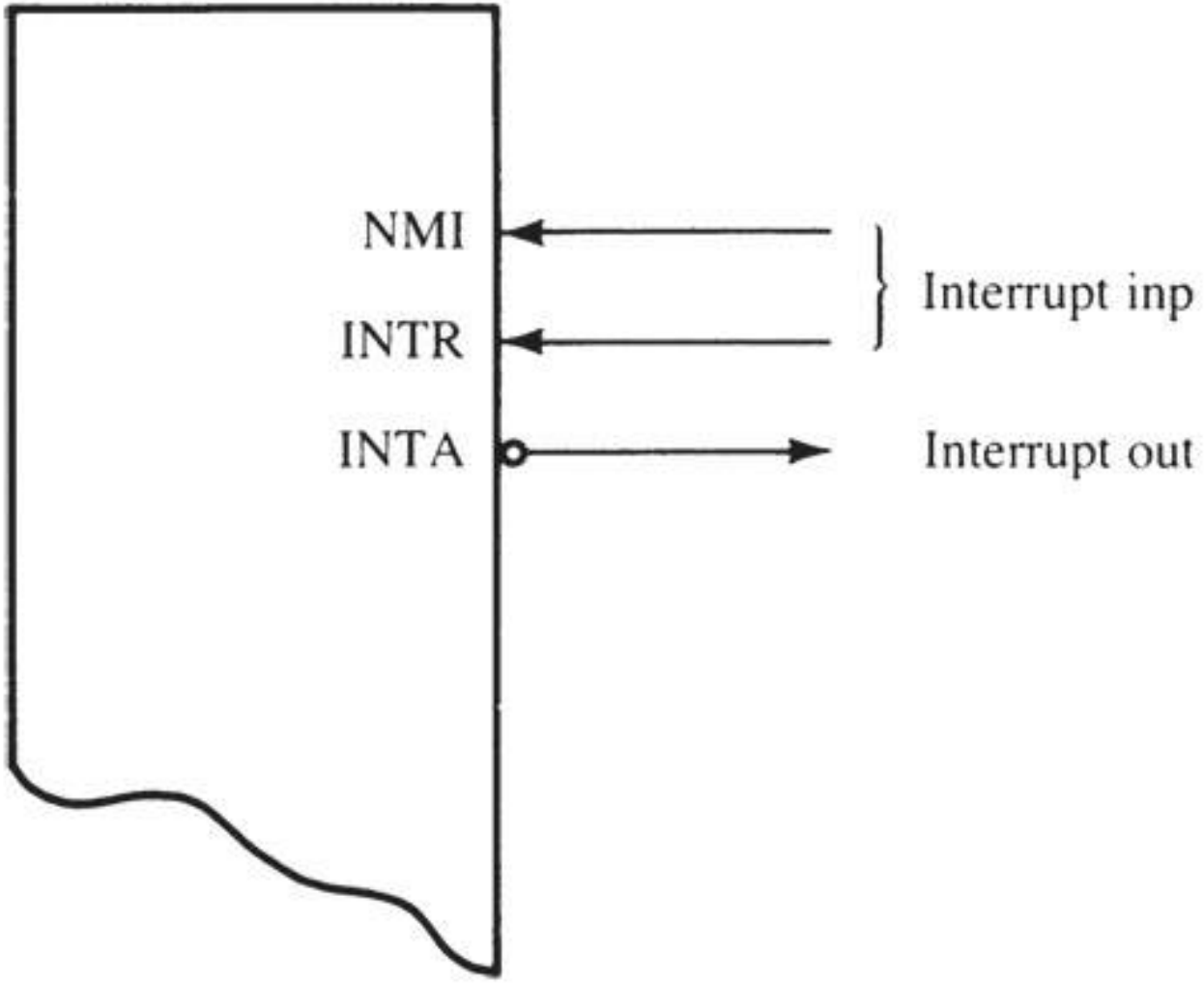
Figure 12–4 The flag register. (Courtesy of Intel Corporation.)

12–2 HARDWARE INTERRUPTS

- The two processor hardware interrupt inputs:
  - non-maskable interrupt (NMI)
  - interrupt request (INTR)
- When NMI input is activated, a type 2 interrupt occurs
  - because NMI is internally decoded
- The INTR input must be externally decoded to select a vector.

- Any interrupt vector can be chosen for the INTR pin, but we usually use an interrupt type number between 20H and FFH.
- Intel has reserved interrupts 00H - 1FH for internal and future expansion.
- $\overline{\text{INTA}}$ is also an interrupt pin on the processor.
  - it is an output used in response to INTR input to apply a vector type number to the data bus connections D_7–D_0
- Figure 12–5 shows the three user interrupt connections on the microprocessor.

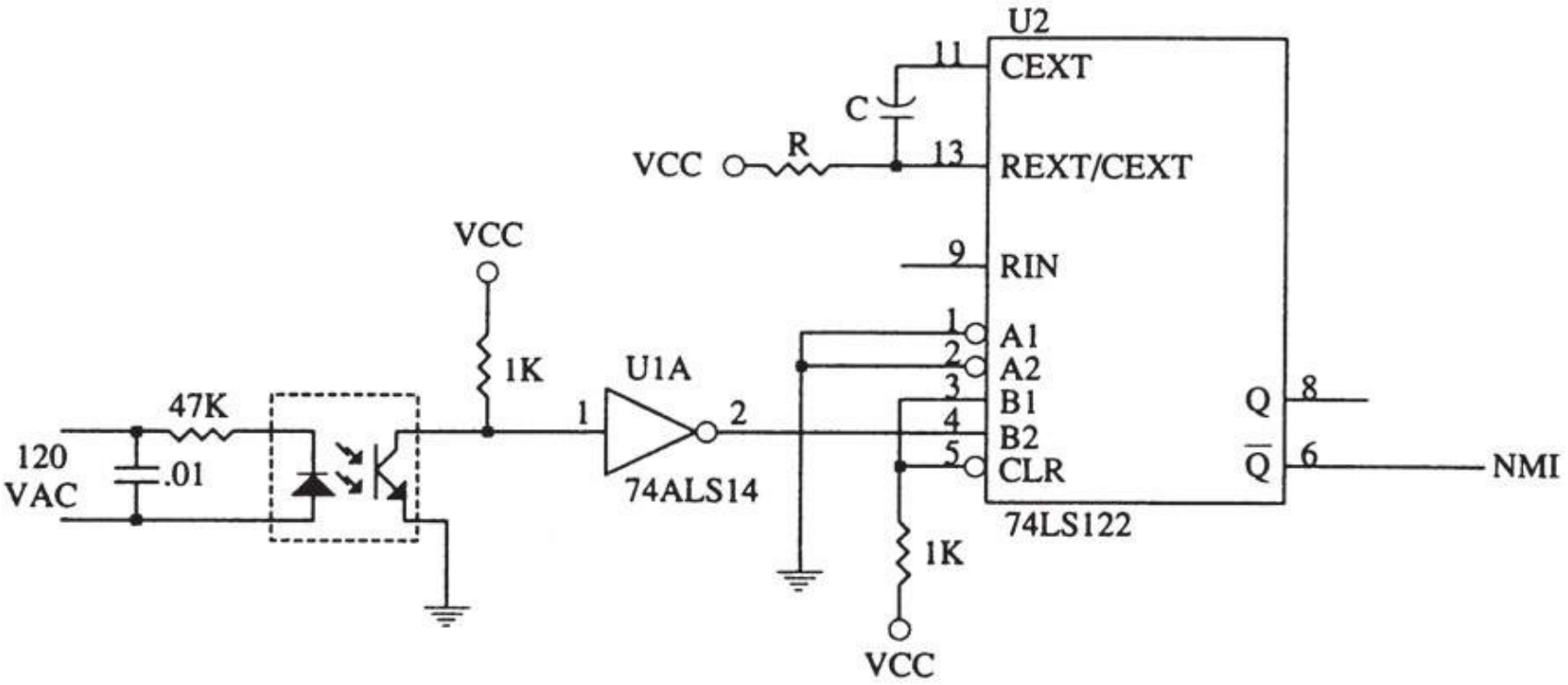
Figure 12–5 The interrupt pins on all versions of the Intel microprocessor.



- The **non-maskable interrupt** (NMI) is an edge-triggered input that requests an interrupt on the positive edge (0-to-1 transition).
  - after a positive edge, the NMI pin must remain logic 1 until recognized by the microprocessor
  - before the positive edge is recognized, NMI pin must be logic 0 for at least two clocking periods
- The NMI input is often used for parity errors and other major faults, such as power failures.
  - power failures are easily detected by monitoring the AC power line and causing an NMI interrupt whenever AC power drops out

- Figure 12–6 shows a power failure detection circuit that provides logic 1 to the NMI input whenever AC power is interrupted.
- In this circuit, an optical isolator provides isolation from the AC power line.
- The interrupt service procedure stores the contents of all internal registers and other data into a battery-backed-up memory.
- This assumes the PC power supply has a large enough filter capacitor to provide energy for at least 75 ms after the AC power ceases.

Figure 12–6 A power failure detection circuit.



INTR and $\overline{\text{INTA}}$

- The interrupt request input (INTR) is level-sensitive, which means that it must be held at a logic 1 level until it is recognized.
  - INTR is set by an external event and cleared inside the interrupt service procedure
- INTR is automatically disabled once accepted.
  - re-enabled by IRET at the end of the interrupt service procedure
- 80386–Core2 use IRETD in protected mode.
  - in 64-bit protected mode, IRETQ is used

- The processor responds to INTR by pulsing $\overline{\text{INTA}}$ output in anticipation of receiving an interrupt vector type number on data bus connections D_7–D_0.
- Fig 12–8 shows the timing diagram for the INTR and pins of the microprocessor.
- Two $\overline{\text{INTA}}$ pulses generated by the system insert the vector type number on the data bus.
- Fig12–9 shows a circuit to appy interrupt vector type number FFH to the data bus in response to an INTR.

Figure 12–8 The timing of the INTR input and $\overline{\text{INTA}}$ output. *This portion of the data bus is ignored and usually contains the vector number.

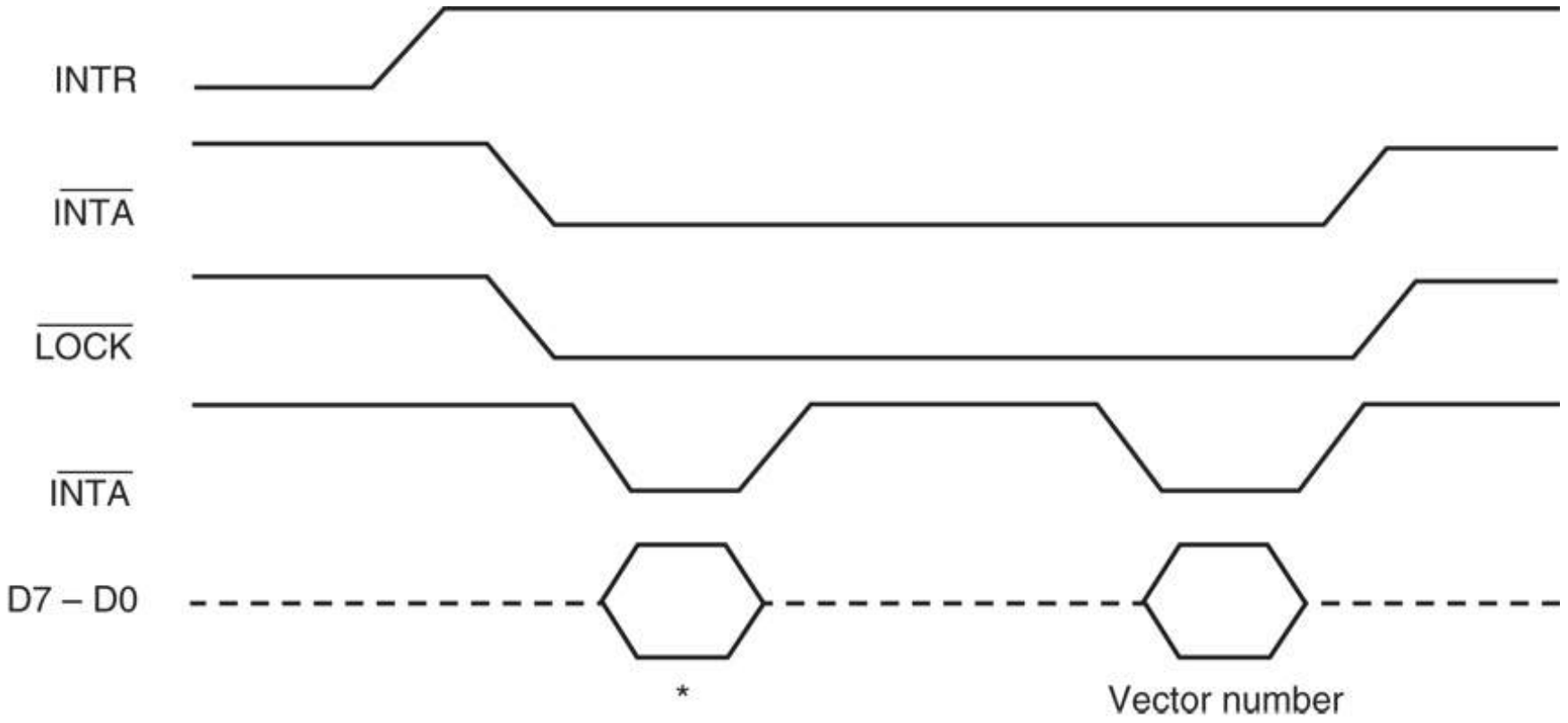
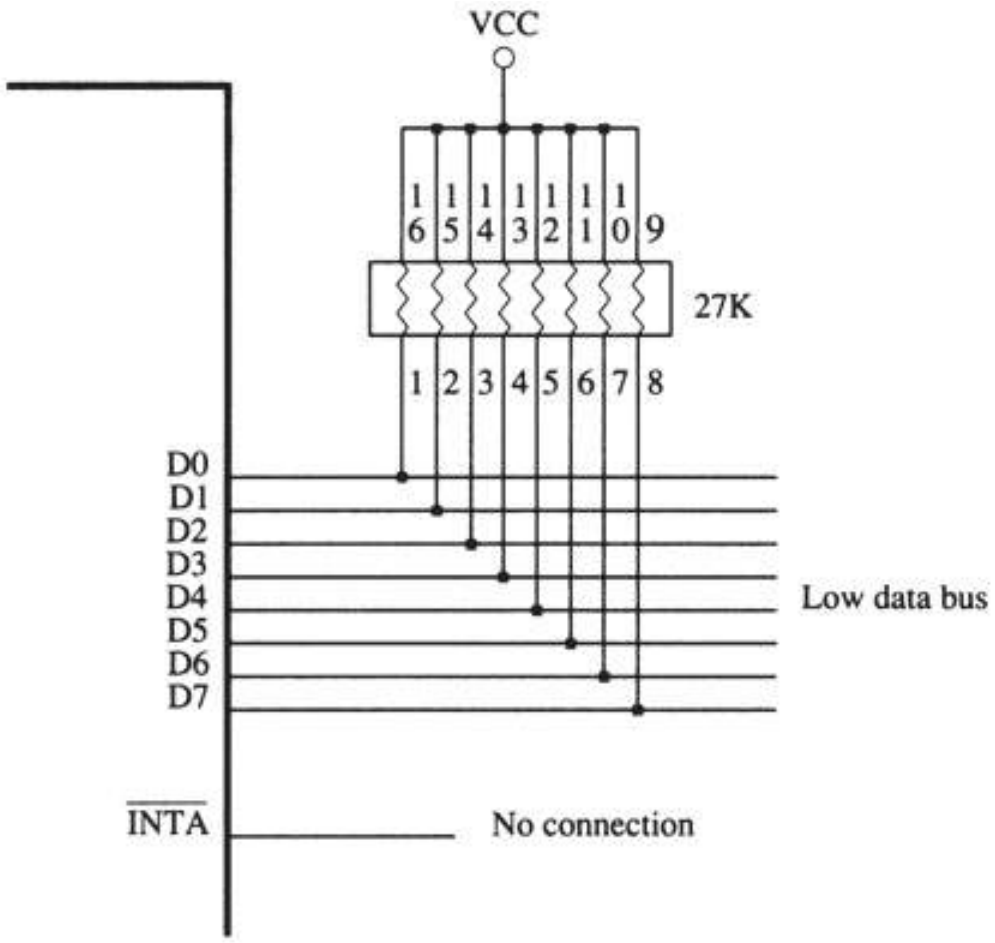


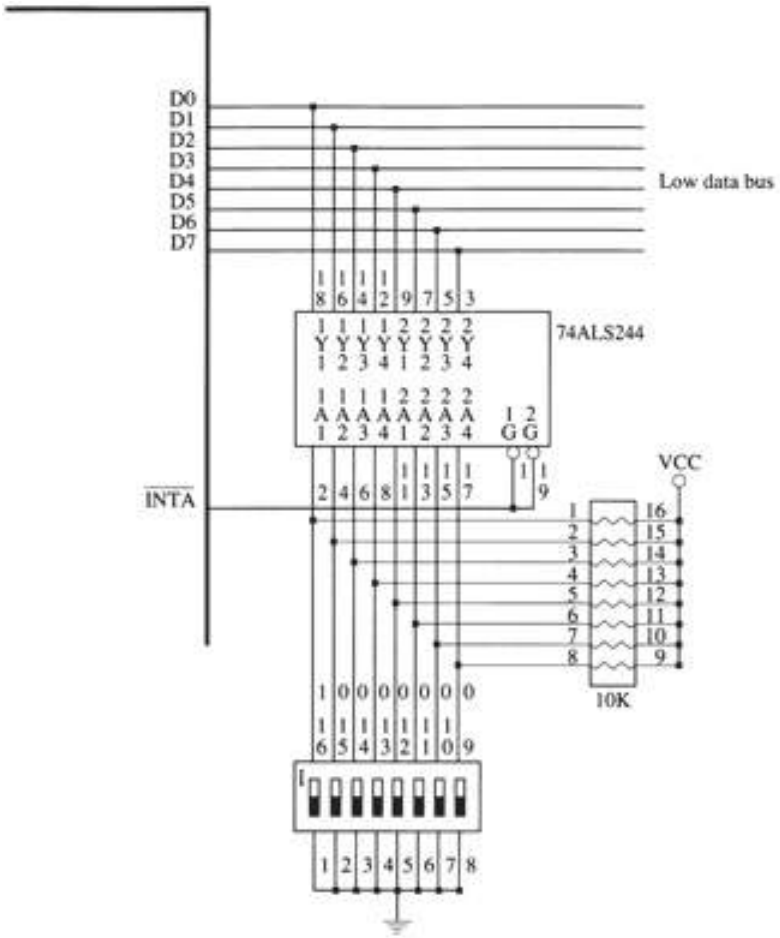
Figure 12–9 A simple method for generating interrupt vector type number FFH in response to INTR.



Using a Three-State Buffer for INTA

- Fig 12–10 shows how interrupt vector type number 80H is applied to the data bus (D_0–D_7) in response to an INTR.
- In response to INTR, the processor outputs the $\overline{\text{INTA}}$ to enable a 74ALS244 three-state octal buffer.
- The octal buffer applies the interrupt vector type number to the data bus in response.
- The vector type number is easily changed with DIP switches shown in this illustration.

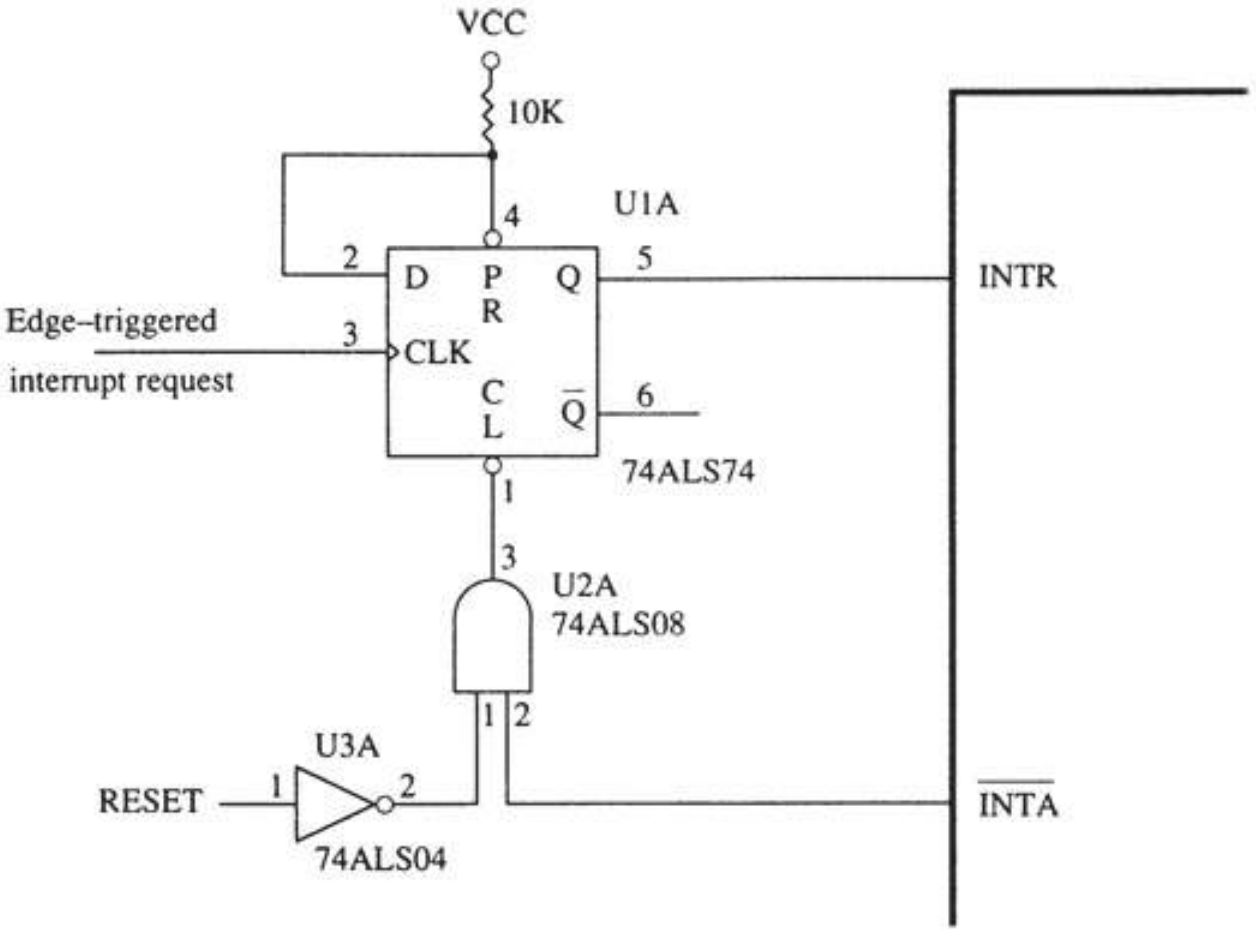
Figure 12–10 A circuit that applies any interrupt vector type number in response to $\overline{\text{INTA}}$. Here the circuit is applying type number 80H.



Making INTR Input Edge-Triggered

- INTR input can be converted to an edge-triggered input by using a D-type flip-flop, as illustrated in Figure 12–11.
- Clock input becomes an edge-triggered interrupt request input, and the clear input is used to clear the request when the INTA signal is output by the microprocessor.
- The RESET signal initially clears the flip-flop so that no interrupt is requested when the system is first powered.

Figure 12–11 Converting INTR into an edge-triggered interrupt request input.



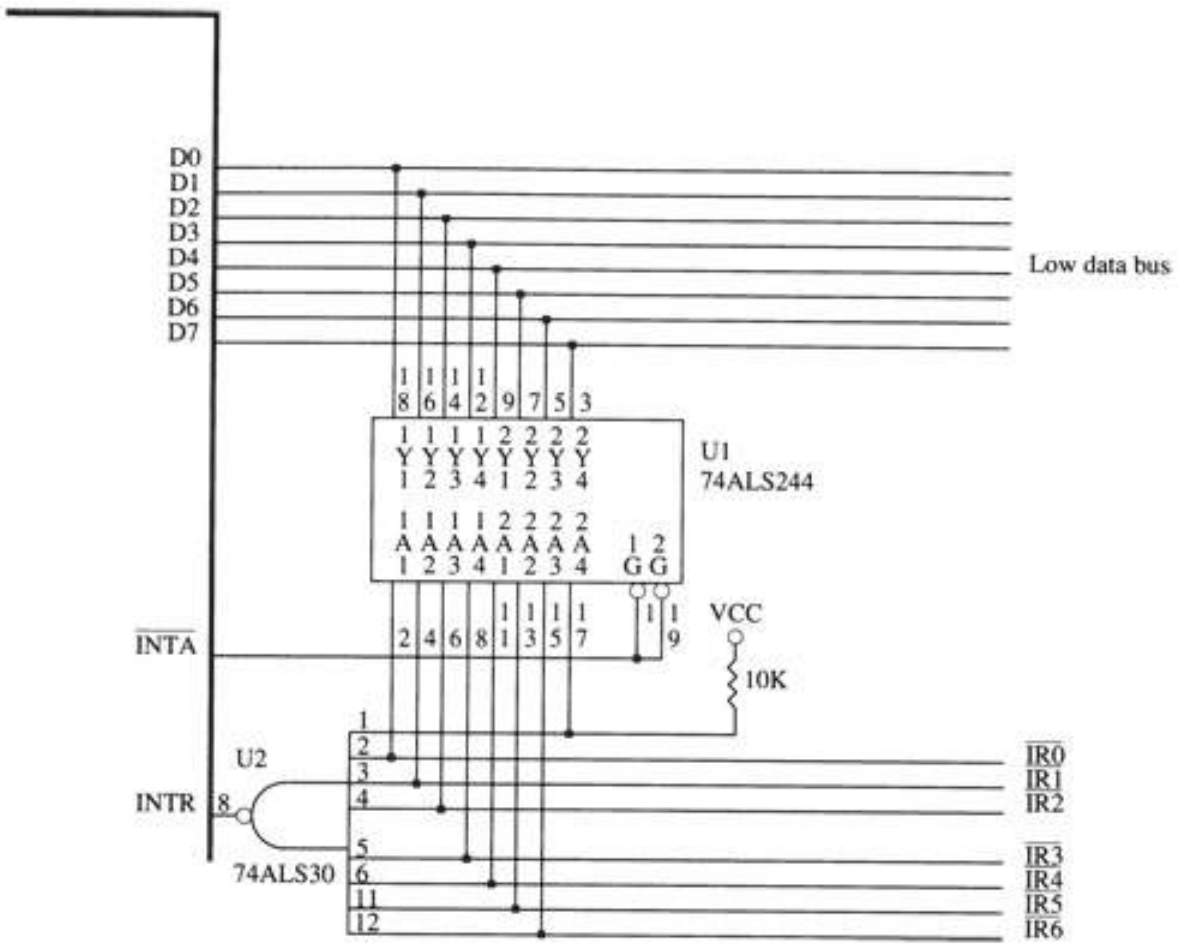
12–3 EXPANDING THE INTERRUPT STRUCTURE

- This covers three common methods of expanding the interrupt structure of the processor.
- It is possible to expand the INTR input so it accepts seven interrupt inputs.
- Also explained is how to “daisy-chain” interrupts by software polling.

Using the 74ALS244 to Expand Interrupts

- The modification shown in Fig 12–13 allows the circuit of Fig 12–10 to accommodate up to seven additional interrupt inputs.
- The only hardware change is the addition of an eight-input NAND gate, which provides the INTR signal to the microprocessor when any of the $\overline{IR}$ inputs becomes active.

Figure 12–13 Expanding the INTR input from one to seven interrupt request lines.



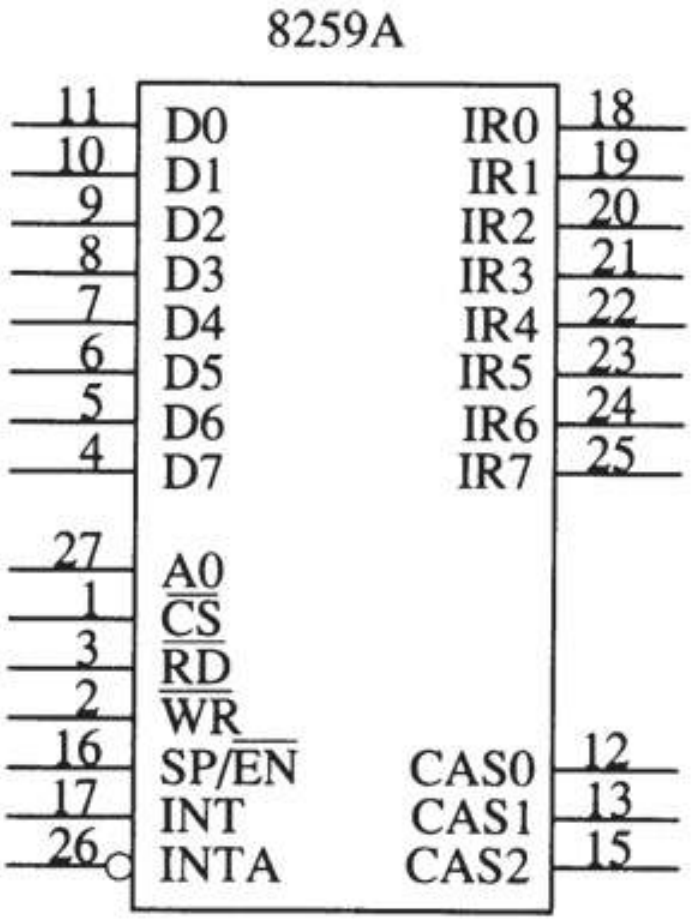
Operation

- If any of the $\overline{IR}$ inputs becomes logic 0, the output of the NAND gate goes to logic 1 and requests an interrupt through the INTR input.
- The interrupt vector that is fetched during the pulse depends on which interrupt request line becomes active.
  - Table 12–1 shows the interrupt vectors used by a single interrupt request input
- If two or more interrupt requests are active active, a new interrupt vector is generated.

12–4 8259A PROGRAMMABLE INTERRUPT CONTROLLER

- 8259A (PIC) adds eight vectored priority encoded interrupts to the microprocessor.
- Expandable, without additional hardware, to accept up to 64 interrupt requests.
  - requires a master 8259A & eight 8259A slaves
- A pair of these controllers still resides and is programmed as explained here in the latest chip sets from Intel and other manufacturers.

General Description of the 8259A



- 8259A is easy to connect to the microprocessor
- all of its pins are direct connections except the $\overline{CS}$ pin, which must be decoded, and the $\overline{WR}$ pin, which must have an I/O bank write pulse

Figure 12–15 The pin-out of the 8259A programmable interrupt controller (PIC).

8259A Pin-Outs

D_0–D_7

- The bidirectional **data connections** are normally connected to the data bus on the microprocessor.

IR_0–IR_7

- **Interrupt request inputs** are used to request an interrupt and to connect to a slave in a system with multiple 8259As.

$\overline{\text{WR}}$

- The **write input** connects to write strobe signal ($\overline{\text{IOWC}}$) on the microprocessor.

$\overline{\text{RD}}$

- The **read input** connects to the $\overline{\text{IORC}}$ signal.

INT

- The **interrupt output** connects to the INTR pin on the processor from the master and is connected to a master IR pin on a slave.

$\overline{\text{INTA}}$

- **Interrupt acknowledge** is an input that connects to the $\overline{\text{INTA}}$ signal on the system. In a system with a master and slaves, only the master $\overline{\text{INTA}}$ signal is connected.

A_0

- The **A_0 address input** selects different command words within the 8259A.

$\overline{CS}$

- **Chip select** enables the 8259A for programming and control.

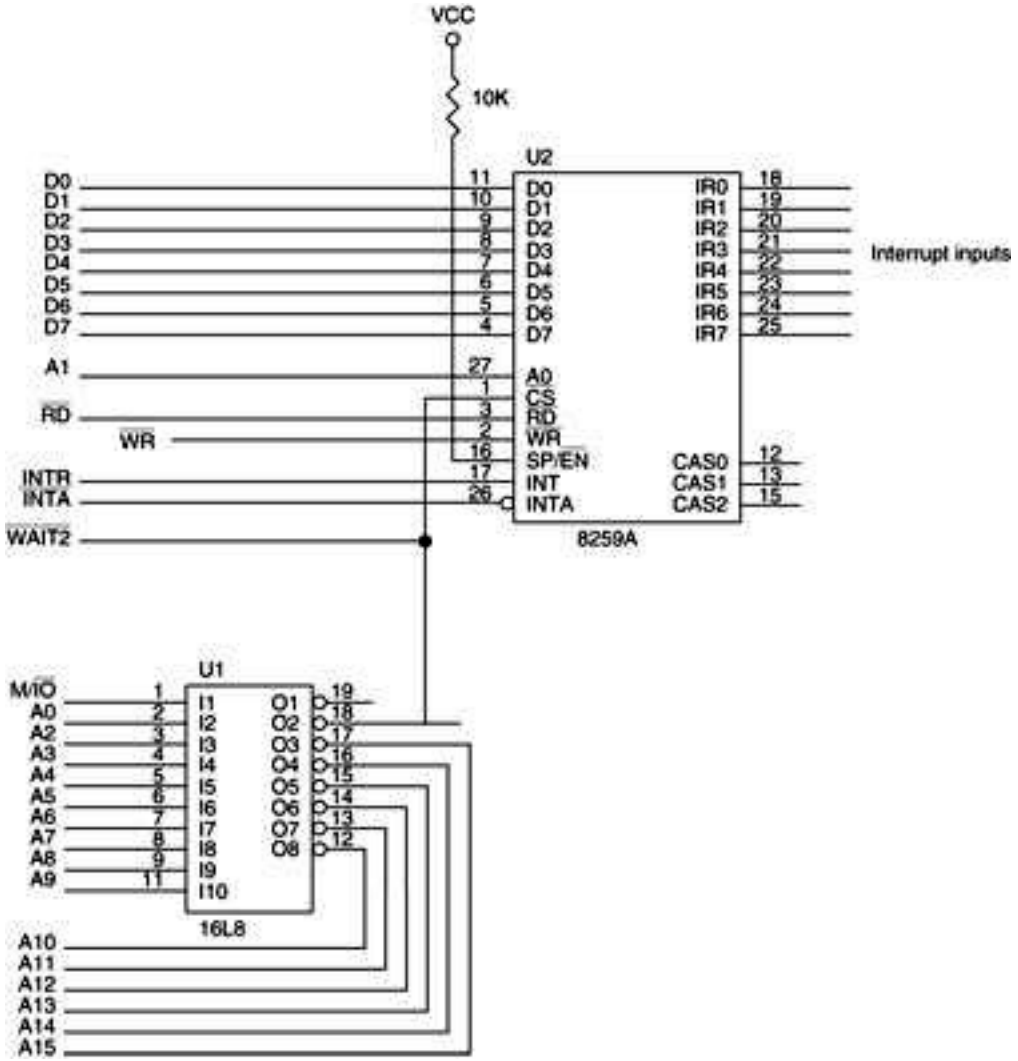
CAS_0–CAS_2

- The **cascade lines** are used as outputs from the master to the slaves for cascading multiple 8259As in a system.

Connecting a Single 8259A

- Fig 12–16 shows a single 8259A connected to the microprocessor.
- The 8259A is decoded at I/O ports 0400H and 0401H by the PLD.
- The 8259A requires four wait states for it to function properly with a 16 MHz 80386SX
  - more for some other versions of the Intel microprocessor family

Figure 12–16 An 8259A interfaced to the 8086 microprocessor.



SUMMARY

- An interrupt is a hardware- or software-initiated call that interrupts the currently executing program at any point and calls a procedure.
- The procedure is called by the interrupt handler or an interrupt service procedure.
- Interrupts are useful when an I/O device needs to be serviced only occasionally at low data transfer rates.