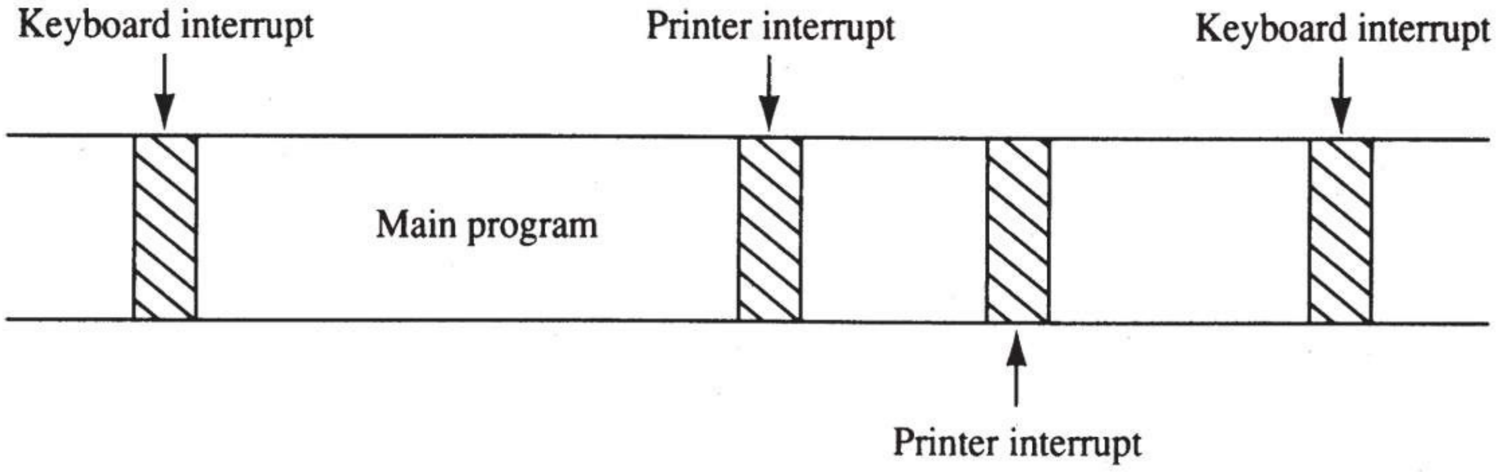
# Interrupts

## I/O Data Transfer

- Data transfer involves three basic techniques
  - Programmed I/O
  - DMA
  - Interrupt-driven I/O
- A hardware interrupt is a hardware-initiated procedure that interrupts whatever program is currently executing.
- Basic I/O
- Programmable peripheral interfaces

Expanded by interrupt-processed I/O.

# BASIC INTERRUPT PROCESSING



# Interrupts

- Two hardware pins : INTR and NMI
- Hardware pin $\overline{\text{INTA}}$: to acknowledge the interrupt requested through INTR.
- The processor also has software interrupts INT, INTO, INT 3
- Flag bits IF (interrupt flag) and TF (trap flag), are also used with the interrupt structure and  special return instruction IRET

## 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 = 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

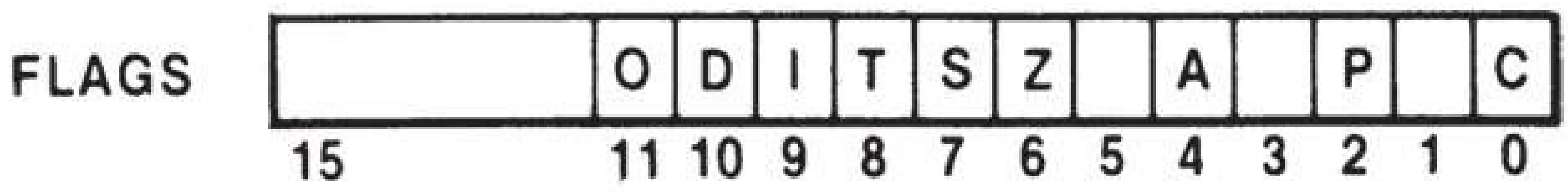


**Figure 12–4** The flag register. (Courtesy of Intel Corporation.)

## *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.

## Storing an Interrupt Vector in the Vector Table

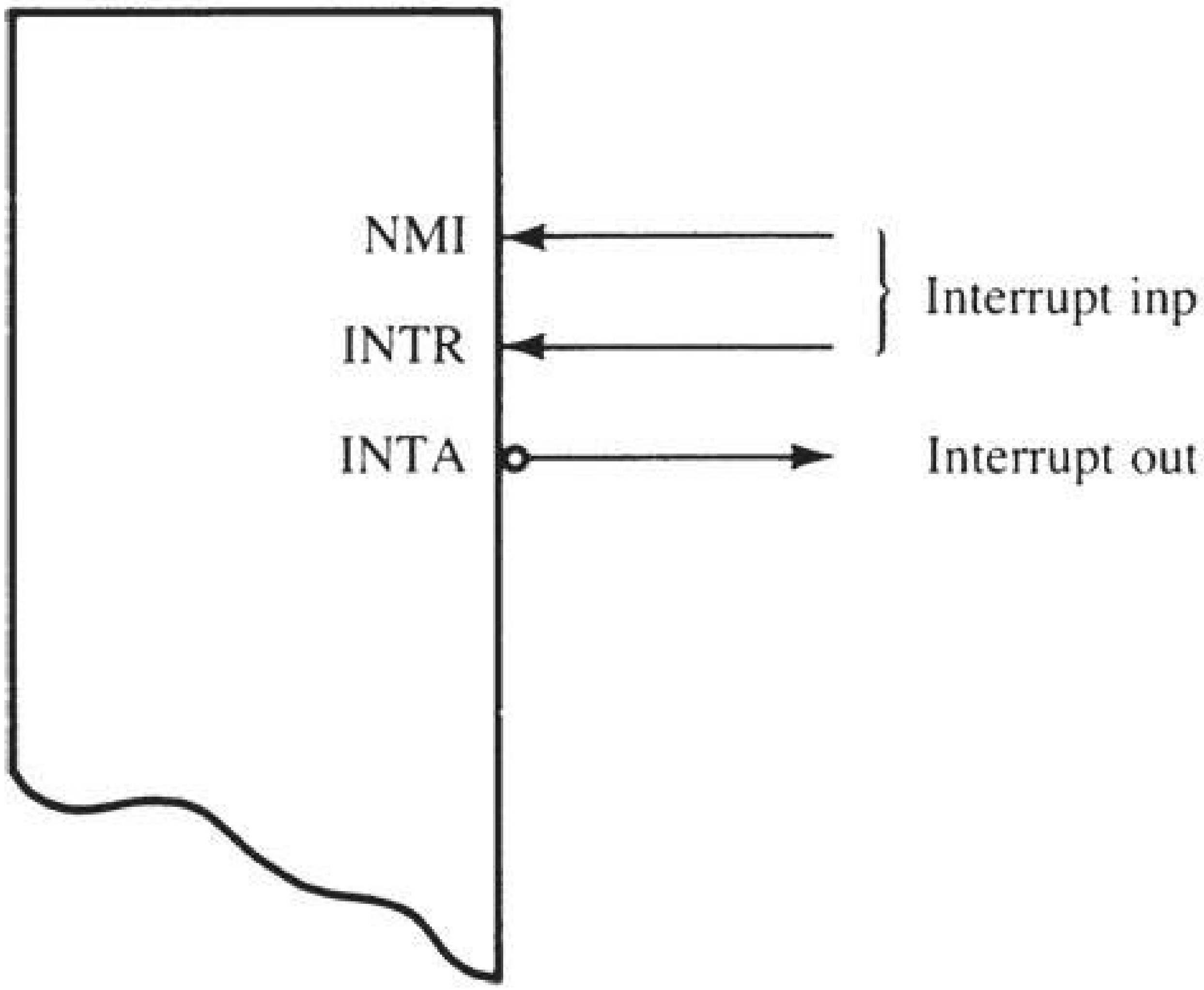
```
.MODEL TINY
.CODE
.STARTUP
                JMP START
OLD  DD         ? ; SPACE FOR OLD VECTOR
        NEW40 PROC FAR
        .
        .
        ;INT SOFT FOR INT 40H
        .
        IRET
        NEW40 ENDP

START:
MOV AX.0
MOV DS,AX
MOV AX,DS:[100H] ; GET INT40H OFFSET
MOV WORD PTR CS:OLD,AX
MOV AX,DS:[102H] ; GET INT 40 H SEGMENT
MOV WORD PTR CS:OLD+2,AX
MOV DS:[100H],OFFSET NEW40
MOV DS:[102H],CS
MOV DX,OFFSET START
SHR DX,4
INC DX
MOV AX,3100H
INT 21H
END
```

## 12–2 HARDWARE INTERRUPTS

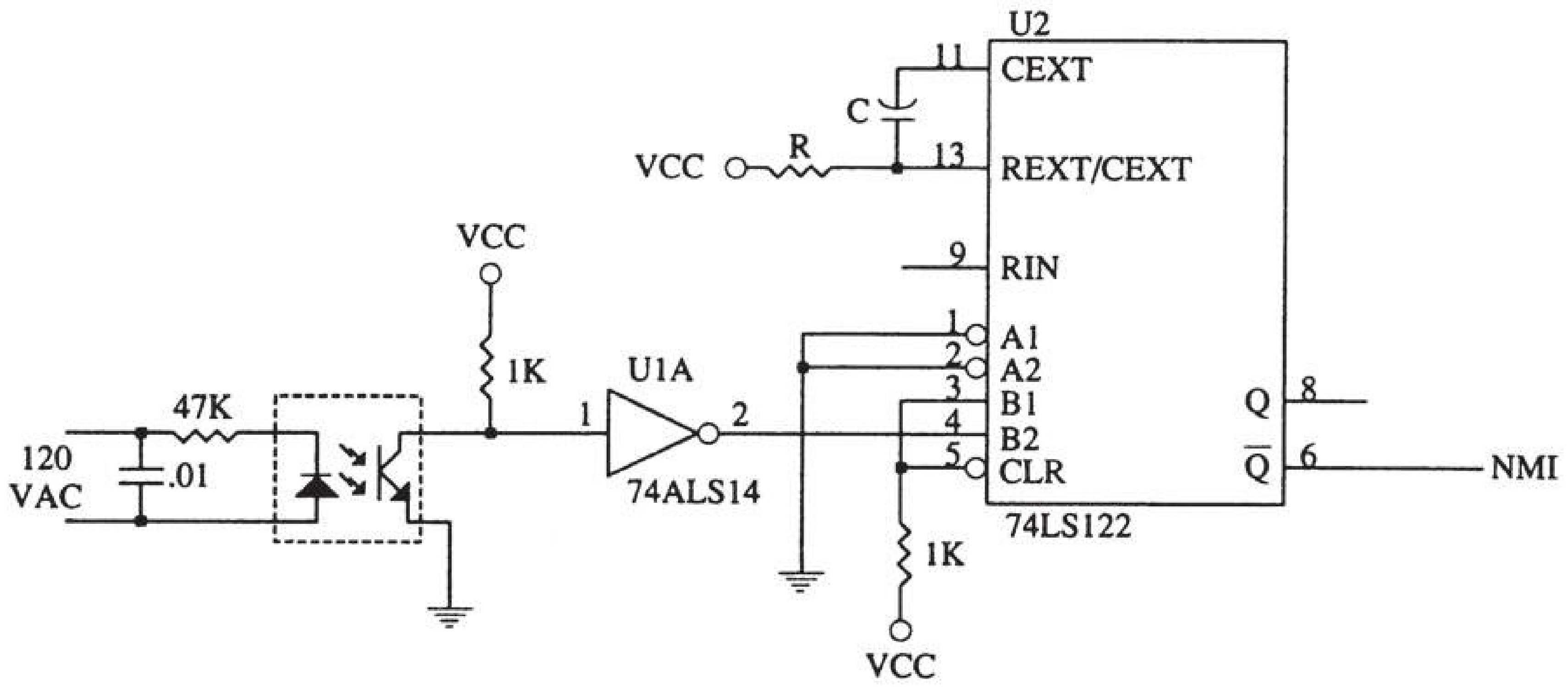
- The two processor hardware interrupt inputs:
  - non-maskable interrupt (NMI)
  - interrupt request (INTR)
- When NMI input is activated, a type 2 interrupt occurs
  - because NMI is internally decoded
- The INTR input must be externally decoded to select a vector.

**Figure** 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** A power failure detection circuit.

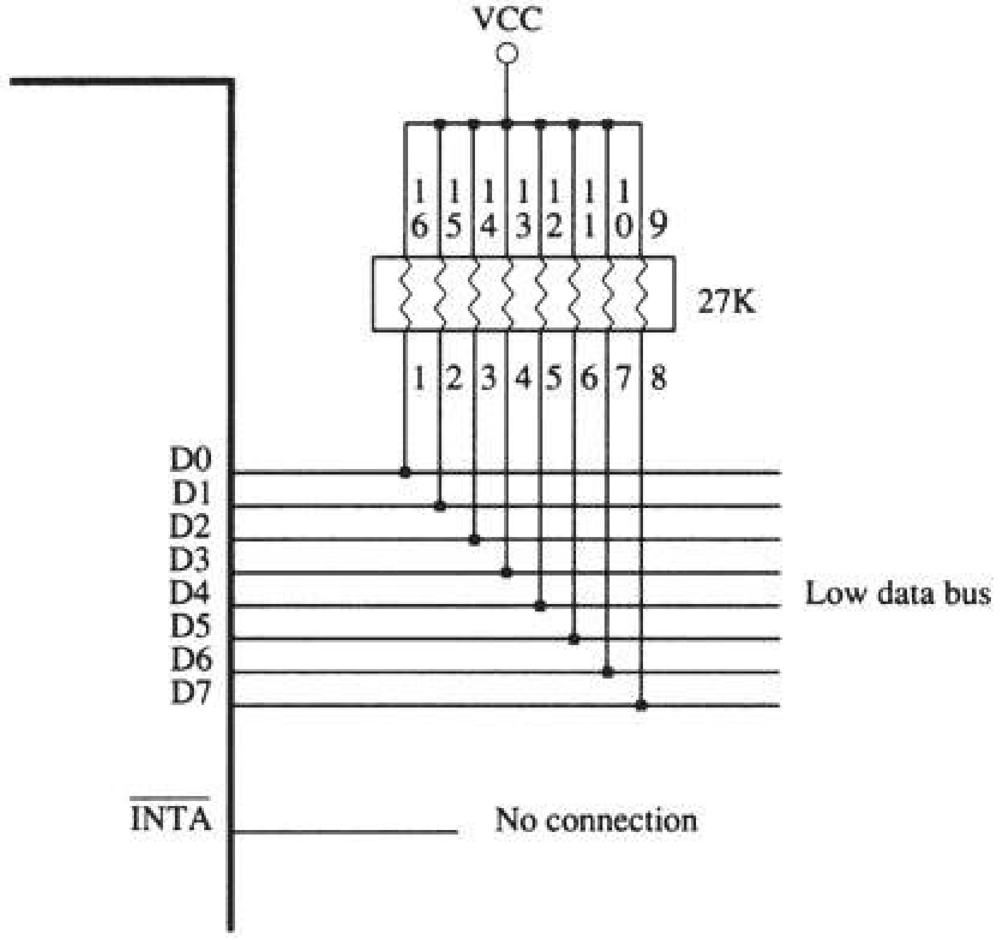


## INTR and $\overline{\text{INTA}}$

- 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

- The processor responds to INTR by pulsing $\overline{\text{INTA}}$ output in anticipation of receiving an interrupt vector type number on data bus connections $D_7$–$D_0$.
- Two $\overline{\text{INTA}}$ pulses generated by the system insert the vector type number on the data bus.
- Fig shows a circuit to appLy interrupt vector type number FFH to the data bus in response to an INTR.

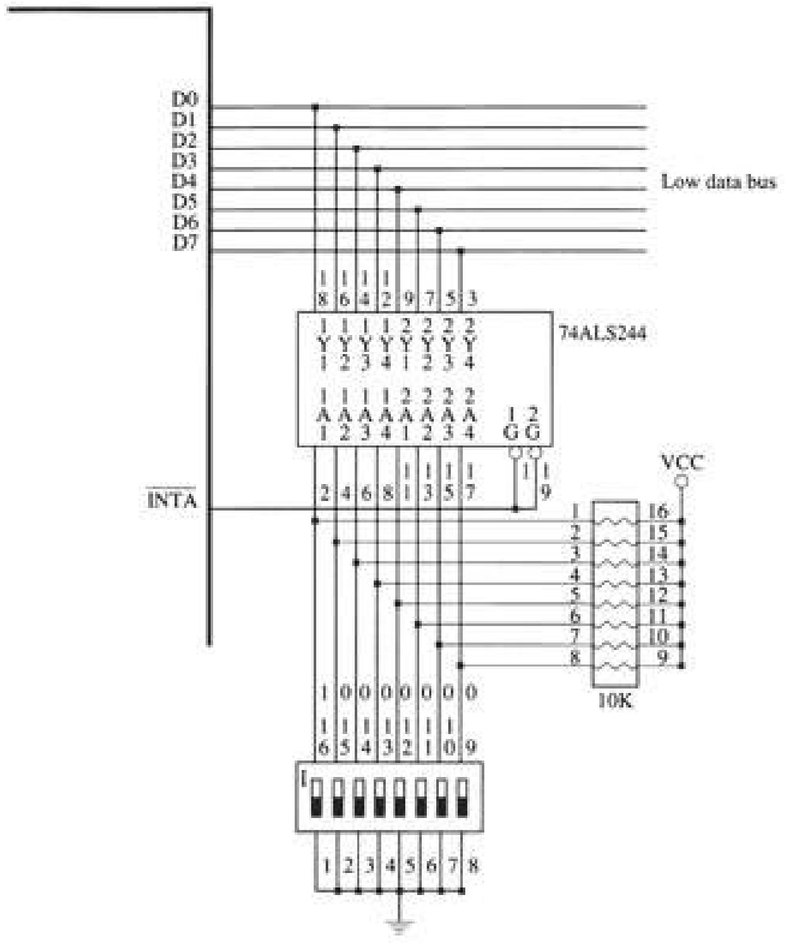
**Figure** A simple method for generating interrupt vector type number FFH in response to INTR.



## *Using a Three-State Buffer for INTA*

- Fig 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.

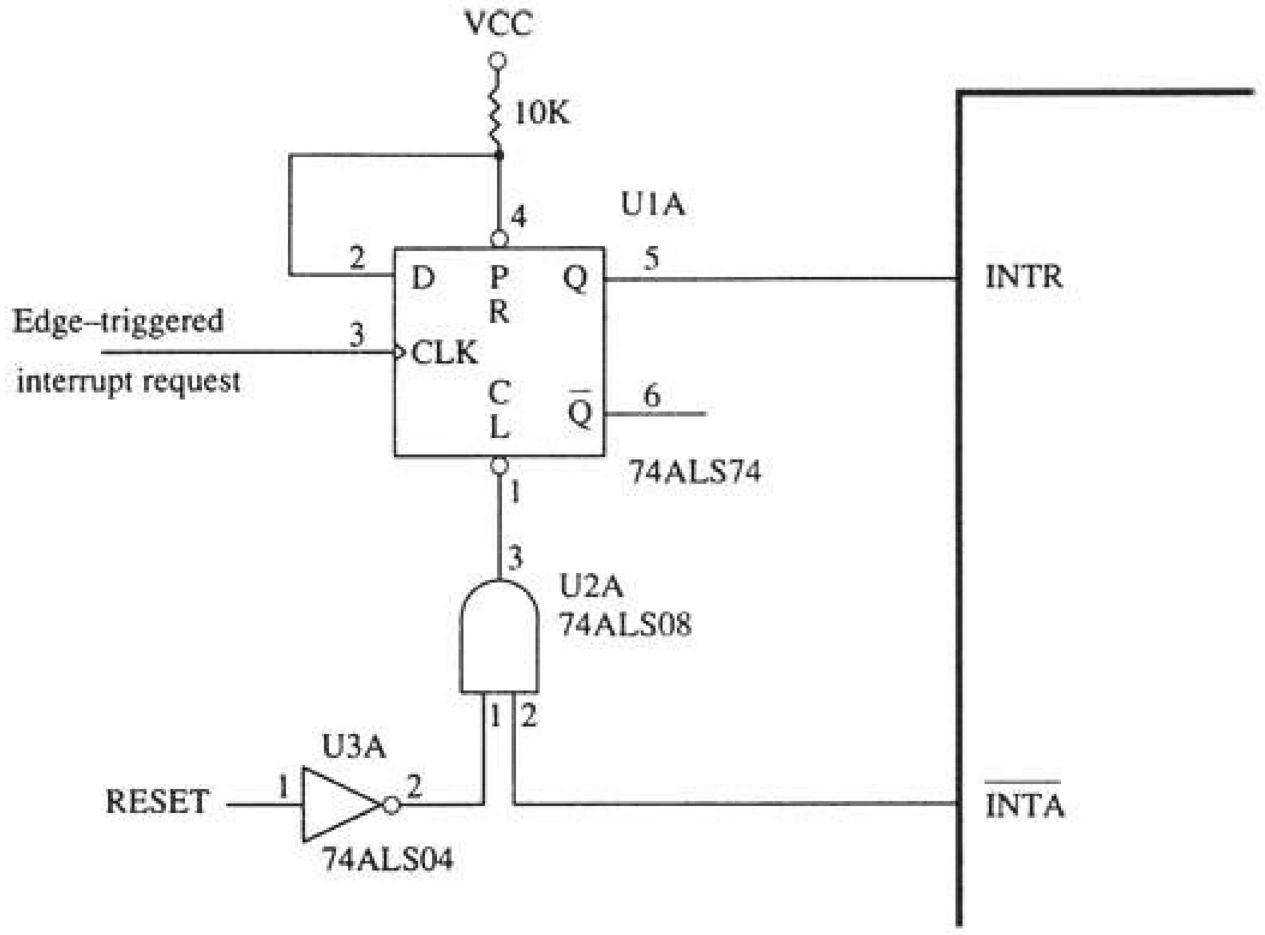
**Figure** A circuit that applies any interrupt vector type number in response to $\overline{\text{INTA}}$. Here the circuit is applying type number 80H.



## *Making INTR Input Edge-Triggered*

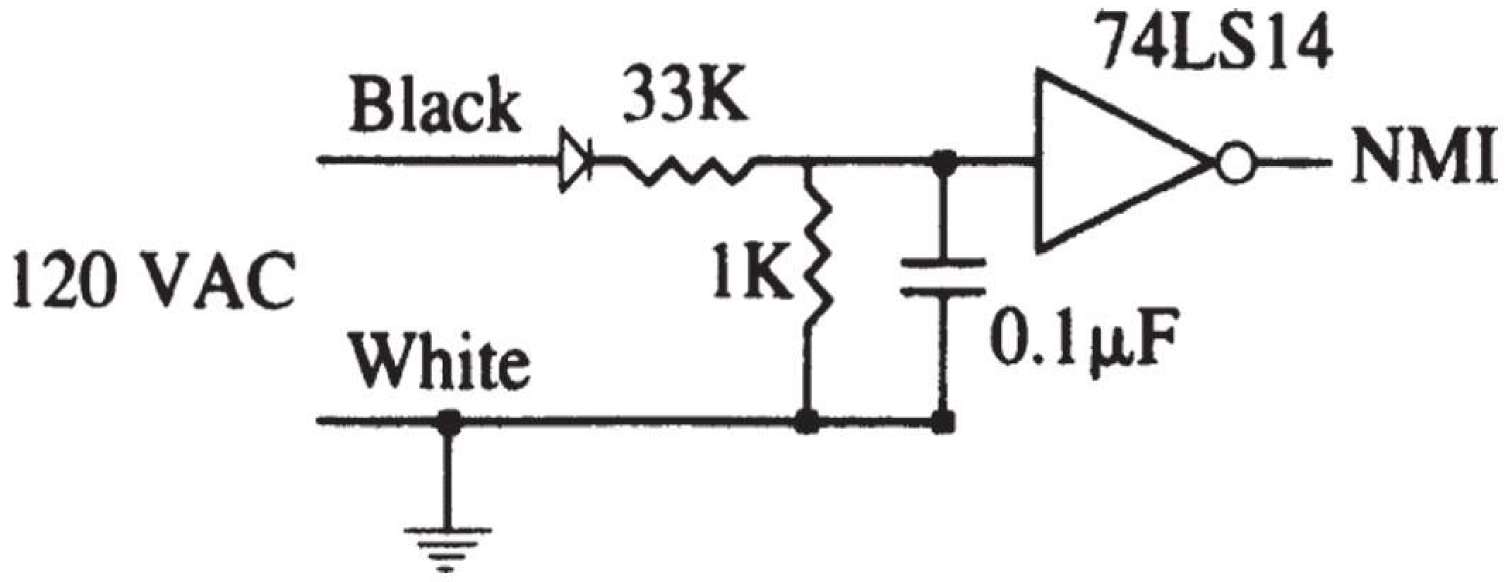
- INTR input can be converted to an edge-triggered input by using a D-type flip-flop, as illustrated in Figure .
- Clock input becomes an edge-triggered interrupt request input, and the clear input is used to clear the request when the $\overline{\text{INTA}}$ signal is output by the microprocessor.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

**Figure** Converting INTR into an edge-triggered interrupt request input.



## EX1: Real-Time Clock

- Fig shows a circuit using the 60 Hz AC power line to generate a periodic interrupt request signal for the NMI interrupt input pin.



```
TIME DB ?
         DB?
         DB?
         DB?

LOOK  DB   60H, 60H, 60H, 24H

TIMEP PROC FAR USES AX BX DS
       MOV AX,SEGMENT
       MOV DS,AX
       MOV BX,0
.REPEAT
       MOV AL,DS:TIME[BX]
       ADD AL,1
       DAA
                  .IF AL==BYTE PTR CS: LOOK[BX]
                                  MOV AL,0
                  .ENDIF
                  MOV DS:TIME[BX],AL
                  INC BX
.UNTIL AL!=0 || BX==4
IRET
TIMP ENDP
```