the highest priority first, assuming that both inputs are unmasked (enabled) in the 8259A.

Now let's look again at the block diagram of the 8259A in Figure 8-27 so we can explain in more detail how the device will respond to multiple interrupt signals. In the block diagram note the four boxes labeled *interrupt request register* (IRR), *interrupt mask register* (IMR), *in-service register* (ISR), and *priority resolver*.

The interrupt mask register is used to disable (mask) or enable (unmask) individual interrupt inputs. Each bit in this register corresponds to the interrupt input with the same number. You unmask an interrupt input by sending a command word with a 0 in the bit position that corresponds to that input.

The interrupt request register keeps track of which interrupt inputs are asking for service. If an interrupt input has an interrupt signal on it, then the corresponding bit in the interrupt request register will be set.

NOTE: An interrupt signal must remain high on an IR input until after the falling edge of the first $\overline{\text{INTA}}$ pulse.

The in-service register keeps track of which interrupt inputs are currently being serviced. For each input that is currently being serviced, the corresponding bit will be set in the in-service register.

The priority resolver acts as a "judge" that determines if and when an interrupt request on one of the IR inputs gets serviced.

As an example of how this works, suppose that IR2 and IR4 are unmasked and that an interrupt signal comes in on the IR4 input. The interrupt request on the IR4 input will set bit 4 in the interrupt request register. The priority resolver will detect that this bit is set and check the bits in the in-service register (ISR) to see if a higher-priority input is being serviced. If a higher-priority input is being serviced, as indicated by a bit being set for that input in the ISR, then the priority resolver will take no action. If no higher-priority interrupt is being serviced, then the priority resolver will activate the circuitry which sends an interrupt signal to the 8086. When the 8086 responds with $\overline{\text{INTA}}$ pulses, the 8259A will send the interrupt type that was specified for the IR4 input when the 8259A was initialized. As we said before, the 8086 will use the type number it receives from the 8259A to find and execute the interrupt-service procedure written for the IR4 interrupt.

Now, suppose that while the 8086 is executing the IR4 service procedure, an interrupt signal arrives at the IR2 input of the 8259A. This will set bit 2 of the interrupt request register. Since we assumed for this example that IR2 was unmasked, the priority resolver will detect that this bit in the IRR is set and make a decision whether to send another interrupt signal to the 8086. To make the decision, the priority resolver looks at the in-service register. If a higher-priority bit in the ISR is set, then a higher-priority interrupt is being serviced. The priority resolver will wait until the higher-priority bit in the ISR is reset before sending an interrupt signal to the 8086 for the new interrupt input. If the priority resolver finds that the new interrupt has a higher priority than the highest-priority interrupt currently being serviced, it will set the appropriate bit in the ISR and activate the circuitry which sends a new INT signal to the 8086. For our example here, IR2 has a higher priority than IR4, so the priority resolver will set bit 2 of the ISR and activate the circuitry which sends a new INT signal to the 8086. If the 8086 INTR input was reenabled with an STI instruction at the start of the IR4 service procedure, as shown in Figure 8-28*a*, then this new INT signal will interrupt the 8086 again. When the 8086 sends out a second $\overline{\text{INTA}}$ pulse in response to this interrupt, the 8259A will send it the type number for the IR2 service procedure. The 8086 will use the received type number to find and execute the IR2 service procedure.

At the end of the IR2 procedure, we send the 8259A a command word that resets bit 2 of the in-service register so that lower-priority interrupts can be serviced. After that, an IRET instruction at the end of the IR2 procedure sends execution back to the interrupted IR4 procedure. At the end of the IR4 procedure, we send the 8259A a command word which resets bit 4 of the in-service register so that lower-priority interrupts can be

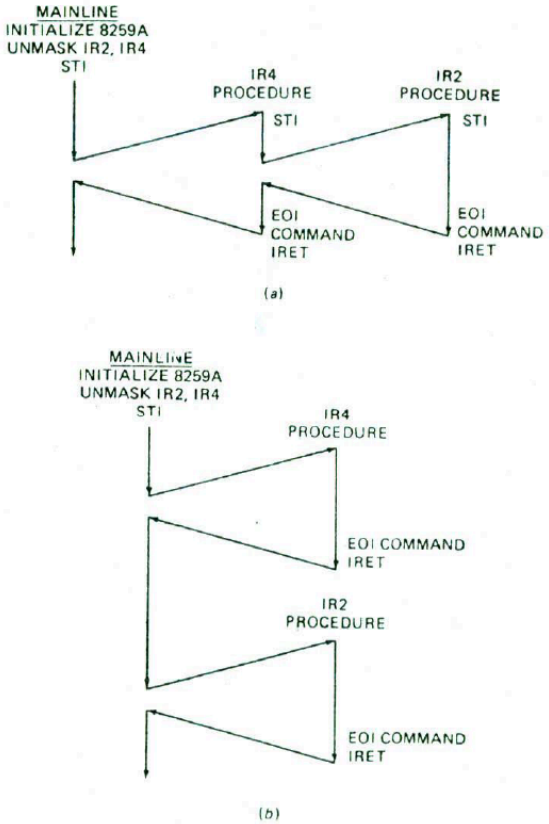

FIGURE 8-28 8259A and 8086 program flow for IR4 interrupt followed by IR2 interrupt. (*a*) Response with INTR enabled in IR4 procedure. (*b*) Response with INTR not enabled in IR4 procedure.