15.61 The 8237 DMA Controller

The 8237 is a programmable Direct Memory Access controller (DMA) housed in a 40-pin package. It has four independent channels with each channel capable of transferring 64K bytes. It must interface with two types of devices: the MPU and peripherals such as floppy disks. As mentioned earlier, the DMA plays two roles in a given system: It is an I/O to the microprocessor (slave mode) and it is a data transfer processor to peripherals such as floppy disks (master mode). Many of its signals that are input in the I/O mode become outputs in the processor mode. It also needs additional signal lines to communicate with the addresses of 64K data bytes, and these signals must be generated externally by using latches and buffers. The 8237 is a complex device. To maintain clarity, the following discussion is divided into five segments: DMA channels and interfacing, DMA signals, system interface; programming, and DMA execution. The specification details of the 8237 are included in Appendix D.

DMA CHANNELS AND INTERFACING

Figure 15.33 shows a logical pin out and internal registers of the 8237. It also shows the interface with the 8085 using a 3-to-8 decoder.

The 8237 has four independent channels, CH0 to CH3. Internally, two 16-bit registers are associated with each channel: One is used to load a starting address of the byte to be copied and the second is used to load a count of the number of bytes to be copied. Figure 15.33 shows eight such registers that can be accessed by the MPU. The addresses of these registers are determined by four address lines, A_3 to A_0, and the chip select ($\overline{CS}$) signal. Address 0000 on lines A_3–A_0 selects CH0 Memory Address Register (MAR) and address 0001 selects the next register, CH0 Count. Similarly, all the remaining registers are selected in sequential order. The last eight registers are used to write commands or read status as shown. In Figure 15.33, the MPU accesses the DMA controller by asserting the signal Y_0 of the decoder. Therefore, the addresses of these internal registers range from 00 to 0FH as follows:

A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0		
0	0	0	0	0	0	0	0	=	00 CH0 MAR
				1	1	1	1	=	0F Mask Register
Chip Select				Register Select					

DMA SIGNALS

In Figure 15.33, signals are divided into two groups: (1) one group of signals shown on the left of the 8237 is used for interfacing with the MPU; (2) the second group shown on the right-hand side of the 8237 is for communicating with peripherals. Some of these signals are bidirectional and their functions are determined by the DMA mode of operation (I/O or processor). The signals that are necessary to understand the DMA operation are explained as follows; the remaining signals are listed in Appendix D.

□ **DREQ0–DREQ3—DMA Request:** These are four independent, asynchronous input signals to the DMA channels from peripherals such as floppy disks and the hard disk. To obtain DMA service, a request is generated by activating the DREQ line of the channel.

□ **DACK0–DACK3—DMA Acknowledge:** These are output lines to inform the individual peripherals that a DMA is granted. DREQ and DACK are equivalent to handshake signals in I/O devices.

□ **AEN and ADSTB—Address Enable and Address Strobe:** These are active high output signals that are used to latch a high-order address byte to generate a 16-bit address.

□ **MEMR and MEMR—Memory Read and Memory Write:** These are output signals used during the DMA cycle to write and read from memory.

□ **A_3–A_0 and A_7–A_4—Address:** A_3–A_0 are bidirectional address lines. They are used as inputs to access control registers as shown in the previous section. During the DMA cycle, these lines are used as output lines to generate a low-order address that is combined with the remaining address lines A_7–A_4.

□ **HRQ and HLDA—Hold Request and Hold Acknowledge:** HRQ is an output signal used to request the MPU control of the system bus. After receiving the HRQ, the MPU completes the bus cycle in process and issues the HLDA signal.

SYSTEM INTERFACE

The DMA is used to transfer data bytes between I/O (such as a floppy disk) and system memory (or from memory to memory) at high speed. It includes eight data lines, four control signals ($\overline{\text{IOR}}$, $\overline{\text{IOW}}$, MEMR, and $\overline{\text{MEMW}}$), and eight address lines (A_7–A_0). However, it needs 16 address lines to access 64K bytes. Therefore, an additional eight lines must be generated as shown in Figure 15.34.

When a transfer begins, the DMA places the low-order byte on the address bus and the high-order byte on the data bus and asserts AEN (Address Enable) and ADSTB (Address Strobe). These two signals are used to latch the high-order byte from the data bus; thus, it places the 16-bit address on the system bus. After the transfer of the first byte, the latch is updated when the lower byte generates a carry (or borrow). Figure 15.34 shows two latches: one latch (373 #1) to latch a high-order address from the data bus by using the AEN and ADSTB signals, and the second latch (373 #2) to demultiplex the 8085 bus and generate the low-order address bus by using the ALE (Address Latch Enable from the 8085) signal. The AEN signal is connected to the $\overline{\text{OE}}$ signal of the second latch to disable the low-order address bus from the 8085 when the first latch is enabled to latch the high-order byte of the address.

PROGRAMMING THE 8237

To implement the DMA transfer, the 8237 should be initialized by writing into various control registers discussed earlier in the DMA channels and interfacing section. To initialize the 8237, the following steps are necessary.

1. Write a control word in the Mode register that selects the channel and specifies the type of transfer (Read, Write, or Verify) and the DMA mode (block, single-byte, etc.).

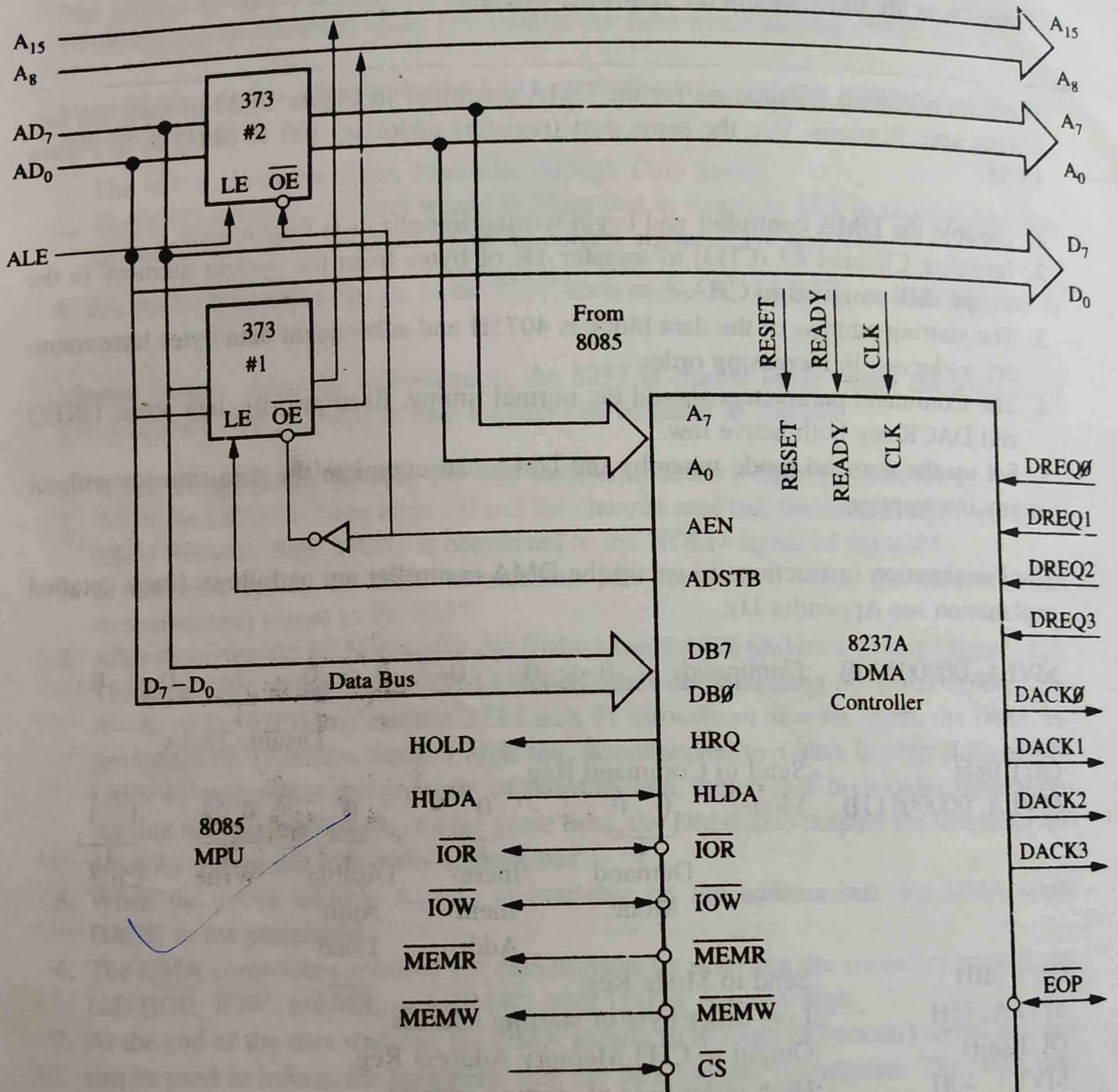

FIGURE 15.34
Interfacing 8237A—DMA Controller with the 8085

Ans

2. Write a control word in the Command register that specifies parameters such as priority among four channels, DREQ and DACK active levels, and timing, and enables the 8237.
3. Write the starting address of the data block to be transferred in the channel Memory Address Register (MAR).
4. Write the count (the number of the bytes in the data block) in the channel Count register.