Chapter 3 Z80 Microprocessor Architecture

The Z 80 is one of the most talented 8 bit microprocessors, and many microprocessor-based systems are designed around the Z80. The Z80 microprocessor needs an external oscillator circuit to provide the operating frequency and appropriate control signals to communicate with memory and I/O.

3.1 Z80 Hardware Model

The Z80 is a general-purpose 8 bit microprocessor with 16 address lines and requires a single +5V power supply. It is 40pin dual-in-line (DIP) package. The different versions of Z80 microprocessors such as Z80, Z80A, Z80B and Z80H are rated to operate at various frequencies ranging from 2.5MHz to 8MHz.

Figure 3.1 shows the pin configuration of the Z80 microprocessor and its hardware model with logic signals. All the signals can be classified into six groups.

- 1. address bus
- 2. data bus
- 3. control signals
- 4. external requests
- 5. request acknowledge and special signals
- 6. power and frequency signals

Address Bus

The Z80 has 16 tri-state signal lines, A15 - A0 known as the address bus. These lines are unidirectional and capable of addressing 64K (2¹⁶) memory registers. The address bus is used to send (or place) the addresses of memory registers and I/O devices.

Data Bus

The data bus consists of eight tri-state bidirectional lines D7 - D0 and is used for data transfer. On these lines, data can flow in either direction-from the microprocessor to memory and I/Os or vice versa.



FIGURE

Z80 Microprocessor Pinout and Logic Signals SOURCE: Courtesy of Zilog Corporation

Control and Status Signals

This group consists of five individual output lines: three can be classified as status signals indicating the nature of the operation being performed, and two as control signals to read from and write into memory or I/Os.

- M1- Machine Cycle One: This is and active low signal indicating that an opcode is being fetched from memory. This signal is also used in an interrupt operation to generate an interrupt acknowledge signal.
- MREQ Memory Request: This is an active low tri-state signal. This signal indicates that the address bus holds a valid address for a memory read or writes operation.
- \overline{IORQ} I/O Request: This is an active low tri-state line. This signal indicates that the low-order address bus $(A_7 A_0)$ holds a valid address for an I/O read or writes operation. This signal is also generated for an interrupt operation.
- RD Read: This is an active low tri-state line. This signal indicates that the microprocessor is ready to read data from memory or an I/O device. This signal should be used in conjunction with MREQ for the Memory Read (MEMRD) operation and with IORQ for the I/O Read (IORD) operation.
- WR Write: This is an active low tri-state line. This signal indicates that the microprocessor has already placed a data byte on the data bus and is ready to write into memory or an I/O device. This signal should be used in conjunction with MREQ for the Memory Write (MEMWR) operation and with IORQ for the I/O Write (IOWR) operation.

External Requests

This group includes five different input signals to the microprocessor from external sources. These signals are used to interrupt an ongoing process and to request the microprocessor to do something else. At this point, you should not try to go into the details of these signals; this information may be overwhelming. The descriptions of these signals are included here for completeness.

- RESET Reset: This is an active low signal used to reset the microprocessor.
 When RESET is activated, the program counter (PC), the interrupt register (I), and the memory refresh register (R) are all cleared to 0. During the reset time, the address bus and the data bus are in high impedance state, and all control signals become inactive. This signal also disables interrupt and refresh. The RESET signal can be initiated by an external key or switch and must be active for at least three clock periods to complete the reset operation.
- INT Interrupt Request: This is an active low signal, initiated by an external I/O device to interrupt the microprocessor operation. When the microprocessor accepts the interrupt request, it acknowledges by activating the IORQ signal during the M₁ cycle. The INT signal is maskable, meaning it can be disabled through a software instruction.
- NMI- Nonmaskable Interrupt: This is a nonmaskable interrupt; it cannot be disabled. It is activated by a negative edge-triggered signal from an external source. This signal is used primarily for implementing emergency procedures. There is no signal or pin to acknowledge this signal; it is accepted provided the Bus Request signal is inactive.
- BUSRQ Bus Request: This is an active low signal initiated by external I/O devices such as the DMA (Direct Memory Access) controller. An I/O device can send a low signal to BUSRQ to request the use of the address bus, the data bus, and the control signals. The external device can use the buses, and when its operations are complete, it returns the control to the microprocessor. This signal is used primarily for the direct memory access technique.
- WAIT Wait: This is an active low signal and can be used memory or I/O devices to add clock cycles to extend the Z80 operations. This signal is used when the response time of memory or I/O devices is slower than that of the Z80. When this signal goes low, it indicates to the microprocessor that the addressed memory or I/O device is not yet ready for data transfer. As long as this signal is low, the Z80 keeps adding cycles to its operation.

Request Acknowledge and Special Signals

Among the five external requests described above, only two of the requests need acknowledgement: Bus Request and Interrupt. The interrupt is acknowledged by the \overline{IORQ} signal in conjunction with the \overline{M} Isignal. The Bus Request is acknowledged by a \overline{BUSAK} (Bus Acknowledge). In addition, the Z80 has two special signals: \overline{HALT} and \overline{RFSH} .

• BUSAK – Bus Acknowledge: This is an active low output signal initiated by the Z80 in response to the Bus Request signal. This signal indicates to the requesting device that the address bus, the data bus, and the control signals

 $(\overline{RD}, \overline{WR}, \overline{MREQ}, \text{ and } \overline{IORQ})$ have entered into the high impedance state and can be used by the requesting device.

- HALT Halt: This is an active low output signal used to indicate that the MPU has executed the HALT instruction.
- RFSH Refresh: This is an active low signal indicating that the address bus A₆-A₀ (low-order seven bits) holds a refresh address of dynamic memory; it should be used in conjunction with MREQ to refresh memory contents.

Power and Frequency Signals

This group includes three signals as follows:

- φ Clock: This pin is used to connect a single phase frequency source. The Z80 does not include a clock circuit on its chip; the circuit must be built separately.
- +5V and GND These pins are for a power supply and ground reference; the Z80 requires one +5V power source.

3.1.2 Z80 Programming Model

In the last chapter, we developed a model to represent the internal structure of the MPU to process data, shown in Figure 2.3. Now, we will describe a similar model of the Z80 microprocessor as shown in Figure 3.2. The model includes an accumulator and a flag register, general-purpose register arrays, registers as memory pointers, and special-purpose registers.

General-Purpose Registers

The Z80 microprocessor has six programmable general-purpose registers named B, C, D, E, H, and L, as shown in Figure 3.2. These are 8-bit registers used for storing data during the program execution. They can be combined as register pairs – BC, DE, and HL – to perform 16-bit operations or to hold memory addresses. The programmer can use these registers to load or copy data.

Accumulator

The accumulator is an 8-bit register that is part of the arithmetic logic unit (ALU) and is also identified as register A. This register is used to store 8-bit data and to perform arithmetic and logic operations. The result of an operation performed in the ALU is stored in the accumulator.

Flag Register

The ALU includes six flip-flops that are set or reset according to data conditions after an ALU operation, and the status of these flip-flops, also known as flags, is stores in the 8-bit flag register. For example, after an addition in which the result generates a carry, the carry flip-flop will be set and bit D_0 in the flag register will show logic 1. The bit position of each flag is shown in Figure 3.2(b); bits D_5 and D_3 are unused.

Among the six flags, the H (Half-Carry) and N (Add or Subtract) flags are used internally by the microprocessor for BCD (Binary Coded Decimal) operations. Each of the remaining four flags – S (Sign), Z (Zero), P/V (Parity or Overflow), and C (Carry) – has two Jump or Call instructions associated with it: one when the flag is set and the other when the flag is reset.



Figure 3.2 (a) The Z80 Programming Model; (b) Expanded Flag Register with Bit Positions

Alternate Register Set

In addition to the general-purpose registers, the accumulator, and the flag register, the Z80 includes a similar set of alternate registers designated as B', C', D', E', H', L', the accumulator A', and the flag register F'. These registers are not directly available to the programmer; however, the exchange instructions can exchange information of register pairs with the respective alternate register pairs.

16-Bit Registers As memory Pointers

The Z80 microprocessor includes four 16-bit registers, and these registers are used to hold memory addresses; thus, they are classified here as memory pointers. The primary function of memory is to store instructions and data, and the microprocessor needs to access memory registers to read these instructions and data. To access a byte in a memory location, the microprocessor identifies the memory location by using the addresses in these memory pointers.

Index Registers (IX and IY) The Z80 has two 16-bit registers called index registers that are used to store 16-bit addresses. These registers are used to identify locations for data transfer.

Stack Pointer (SP) This is also a 16-bit register that is used to point to the memory location called the stack. The stack is a defined area of memory location in R/W memory, and the beginning of the stack is defined by loading a 16-bit address in the stack pointer.

Program Counter (PC) This register functions as a 16-bit counter. The microprocessor uses this register to sequence the execution of instructions. The function of the program counter is to point to the memory address from which the next byte is to be fetched. When the microprocessor places an address on the address bus to fetch the byte from memory, it then increments the program counter by one to point to the next memory location.

Special-Purpose Registers The Z80 microprocessor includes two special-purpose registers that are generally absent in other 8-bit microprocessors. These registers are shown in Figure 3.2 as the Interrupt Vector Register (I) and the Memory Register (R). The functions of these registers will be described in later chapters.

3.2 Machine Cycles and Bus Timings

The Z80 microprocessor is designed to execute 158 different instruction types. Each instruction has two parts: Operation code (known as opcode) and operand. The opcode is a command such as Add, and operand is an object to be operand on, such as a byte or the contents of Register. Some instructions are 1-byte instructions, and some are multibyte instructions. To execute an instruction, the Z80 needs to perform various operations such as Memory Read/Write and I/O Read/Write.



The Z80 has to perform three operations:

- 1. Read Byte 1 from the first memory location
- 2. Read Byte 2 from the next memory location
- 3. Send Data to port 10H

TABLE

The Z80 Machine Cycles and Control Signals

Machine Cycle	(M ₁)	MREQ	IORQ	RD	WR
Opcode Fetch (M ₁)	0	0	1	0	1
Memory Read	1	0	1	0	I
Memory Write	1	0	1	1	0
I/O Read	1	1	0	0	1
I/O Write	1	1	0	1	0
Interrupt Acknowledge	0	1	0	1	1
Nonmaskable Interrupt	0	0	1	0	1
Bus Acknowledge ($\overline{BUSAK} = 0$)	1	Z	Z	Z	Z

NOTE: Logic 0 = Active, Logic 1 = Inactive, Z = High Impedance

The microprocessor's external communication functions can be divided into three basic categories:

- 1. Memory Read and Write
- 2. I/O Read and Write
- 3. Request Acknowledge

These functions are further divided into various operations (machine cycles) as shown in Table 3.1. Each instruction consists of one or more of these machine cycles, and each machine cycle is divided into T-states.

Instruction cycle is defined as the time required to complete the execution of an instruction. The Z80 instruction cycle consists of one to six machine cycles or one to six operations.

Machine Cycle is defined as the time required to complete one operation of accessing memory, accessing I/O, or acknowledging an external request. This cycle may consist

of three to six T-states. The machine cycle is an external operation (such as Memory Read or I/O Write) performed by the processor.

T-state is defined as one subdivision of operation performed in one clock period. These subdivisions are internal states synchronized with the system clock and each T-state is equal to one clock period.

3.2.1 Opcode Fetch Machine Cycle (M1)

The first operation in any instruction is opcode fetch. The microprocessor needs to get (fetch) this machine code from the memory register where it is stored before the microprocessor can begin to execute the instruction.

List the sequence when microprocessor executes the instruction.

Address	Machine Code	Instruction	Comment
2002	$01000111_2 \longrightarrow 47H$	LD B, A	Copy A into B

To fetch the operate, the Z80 performs the following steps:

- 1. The Z80 places the contents of the program counter (2002H) on the address bus, and increments counter to the next address, 2003H. The program counter always points to the next byte to be executed.
- 2. The address is decoded by the external decoding circuit, and the register 2002H is identified.
- 3. The Z80 sends the control signals ($\overline{\text{MREQ}}$ and $\overline{\text{RD}}$) to enable the memory output buffer.
- 4. The contents of the memory register (opcode 47H) are placed on the data bus and brought into the instruction decoder of the microprocessor.
- 5. The Z80 decodes the opcode and execute the instruction, meaning it copies the contents of the accumulator into register B.
- 6. The last operation, copying A into B, is a write operation, but it is an internal operation to the processor. This is a Memory Read operation, but it is identified as an opcode fetch because the processor is reading opcode.





The timing of the opcode fetch machine cycle is relation to the system's clock.



1. Figure 3.4 shows that the Opcode Fetch cycle is completed in four clock periods or T-states. This machine cycle is also identified as the M_1 cycle.

- 2. At the beginning of the first clock period T_1 , the control signal $\overline{M1}$ goes low and the contents of the program counter (2002_H) are placed on the address bus.
- 3. After the falling edge of T_1 , the Z80 asserts two control signals MREQ and \overline{RD} , both active low. The \overline{MREQ} indicates that it is a memory-related operation, and \overline{RD} suggests that it is a Read operation. Both signals are necessary to read from memory.
- 4. The internal decoder of the memory and the Chip Select circuit (not shown in Figure 3.4) decode the address and identify register 2002_{H} . The control signals $\overline{\text{MREQ}}$ and $\overline{\text{RD}}$ are used to enable the memory output buffer. The data bus, which was in high impedance state, is activated as an input bus (to the microprocessor) shortly after the leading edge of T₂. After the falling edge of T₂, memory places its register contents (47_H) on the data bus.
- 5. At the leading edge of T_3 , the data on the bus are read, and the control signals become inactive.
- 6. During T_3 and T_4 , the instruction decoder in the microprocessor decodes and executes the opcode. These are internal operations and cannot be observed on the data bus.

The following two steps are irrelevant to the present problem; however, they are included here as part of the M_1 cycle.

- 7. During T₃ and T₄, when the Z80 is performing internal operations, the low-order address bus is used to supply a 7-bit address for refreshing dynamic memory. If the system includes dynamic memory, this operation simplifies its interfacing hardware. However, this feature is unique to the Z80 processor. In systems based on other processors, the function of refreshing dynamic memory is generally performed by special-purpose peripheral chips.
- 8. Figure 3.4 shows the signal called WAIT. The Z80 samples the Wait line during T₂, and if it is forced low by an external device (such as memory or I/O), the Z80 adds Wait states (clock cycles) to extend the machine cycle an continues to add clock cycles until the Wait signal goes high again. This technique is used to interface memories with slow response time.

3.2.2 Memory Read Machine Cycle

This cycle is quite similar to the Opcode fetch cycle.

Address	Machine Code	Instruction	Comment
2000H	0011 1110 → 3E	LD A, 9FH	Load 9FH in the accumulator
2001H	1001 1111 → 9F		

The instruction consists of two bytes; the first is the opcode, the second is the data byte. The Z80 must first read these bytes from memory and thus requires at least two machine cycles. The machine cycle is opcode fetch, and the second machine cycle is Memory Read, as shown in Figure 3.5. These cycle are described in the following list



- 1. The first machine cycle (Opcode Fetch) is identical in bus timing with the machine cycle illustrated in Example 3.1, except for the bus contents. The address bus contains 2000_{H} , and the data bus contains the opcode $3E_{\text{H}}$. When the Z80 decodes the opcode during the T₃ state, it realizes that a second byte must be read.
- 2. After the completion of the Opcode Fetch cycle, the Z80 places the address $2001_{\rm H}$ on the address bus and increments the program counter to the next address, $2002_{\rm H}$. To differentiate the second cycle from the Opcode Fetch cycle, the M₁ signal remains inactive (high).
- 3. After the falling edge of T_1 of the Memory Read cycle, the control signals $\overline{\text{MREQ}}$ and $\overline{\text{RD}}$ are asserted. These signals, along with the memory address, are used to identify the register 2001_H and enable the memory chip.
- 4. After the leading edge of T_3 , the Z80 activates the data bus as an input bus; memory places the data byte $9F_H$ on the data bus, and the Z80 reads and stores the byte in the accumulator during T_3 .
- 5. After the falling edge of T_3 , both control signals become inactive (high), and at the end of T_3 , the next machine cycle begins.
- 6. In the second machine cycle, the processor writes (copies) $9F_H$ into A; however, this is an internal execution. This internal operation should not be confused with the second machine cycle whereby the processor reads $9F_H$ from memory.

3.2.3 Memory Write Cycle

Memory write is the machine cycle writes or stores data in a specified memory register.

Н	L	Accumulator
23	50	9F

Address	Machine Code	Instruction	Comment
2003	0111 0111 ₂ → 77H	LD (HL), A	Copy contents of A into memory location, the address of which is stored in HL

This is a one-byte instruction with two machine cycles: Opcode Fetch and Memory Write. In the first machine cycle, the Z80 fetches the code (77H) and in the second machine cycle, it copies the byte 9FH from the accumulator into the memory location 2350H.



- 1. In the Opcode Fetch machine cycle, the Z80 places the address $2003_{\rm H}$ on the address bus and gets the code $77_{\rm H}$ by using the control signals $\overline{\rm MREQ}$ and $\overline{\rm RD}$ as in the previous examples. The program counter is also incremented to the next address, $2004_{\rm H}$.
- 2. During the T_3 and T_4 states, the Z80 decodes the machine code 77_H and prepares for the memory write operation.
- 3. At the beginning of the next machine cycle (Memory Write), it places the contents (2350_H) of the HL register on the address bus. At the falling edge of T_1 , \overline{MREQ} goes low and the data byte $9F_H$ from the accumulator is placed on the data bus.
- 4. After allowing one T-state (after $\overline{\text{MREQ}}$) to stabilize the address, the Z80 asserts the control signal Write ($\overline{\text{WR}}$), which is used to write the data byte at the address shown on the address bus.
- 5. After the falling edge of T_3 , both control signals become inactive, and one-half T-state later, the data bus goes into high impedance state.

3.2.5 Review of Important Concepts

- 1. In each instruction cycle, the first operation is always Opcode Fetch, and it is indicated by the active low $\overline{M1}$ signal. This cycle can be four to six T-states in duration.
- 2. The Memory Read cycle is in many ways similar to the Opcode Fetch cycle. Both use the same control signals ($\overline{\text{MREQ}}$ and RD) and read contents from memory. However, the Opcode Fetch reads opcodes and the Memory Read reads 8-bit data or addresses; the two machine cycles are differentiated by the $\overline{\text{M1}}$ signal.
- 3. The control signals, $\overline{\text{MREQ}}$ and RD, are both necessary to read from memory.
- 4. In the Memory Write cycle, the Z80 writes (stores) data in memory using the control signals MREQ and WR.
- 5. In the Memory Read cycle, the Z80 asserts the MREQ and RD signals to enable memory, and then the addressed memory places the data on the data bus; on the other hand, in the Memory Write cycle, the Z80 asserts the MREQ, places the data byte on the data bus, and then asserts the WR signal to write into the addressed memory.
- 6. Generally, the Memory Read and Write cycles consist of three T-states; however, they can take four T-states in some instructions. The Memory Read and Write cycles will not be asserted simultaneously; the microprocessor cannot read and write at the same time.



Generating Control Signals

- Figure 3.7
- To read from memory, the MREQ and the RD signals are necessary, and to read from an input device, the IORQ and the RD are necessary; all these signals are active low. As a design practice, the MREQ is generally combined with a decoded address, and RD is connected directly to the memory chip. However, control signals RD and WR can also be combined with MREQ

and IORQ to generate additional signals. We can generate the active low Memory Read ($\overline{\text{MEMRD}}$) signal either by ANDing these signals in a negative NAND gate as shown in Figure 3.7(a) or by using a 2-to-4 decoder as shown in Figure 3.7(b). The decoder is enabled by the $\overline{\text{MREQ}}$ and has $\overline{\text{RD}}$ and $\overline{\text{WR}}$ signals as input. (See Example 3.4 for additional explanation.) Similarly, we can generate an active low I/O Read ($\overline{\text{IORD}}$) control signal by ANDing $\overline{\text{IORQ}}$ and $\overline{\text{RD}}$ as shown in Figure 3.7(a).

2. To write into memory, the MREQ and the WR signals are necessary, and to write a data byte to an output device; the IORQ and WR signals are necessary; all these signals are active low. We can generate an active low Memory Write (MEMWR) signal by ANDing MREQ and WR signals in a negative NAND pate as shown in Figure 3.7 (a) or by using the decoder as shown in Figure 3.7 (b). Similarly, we can generate an IOWR control signal by ANDing IORQ and WR signals in a negative NAND gate as shown in Figure 3.7 (s). We need to generate all four control signals (MEMRD, MEMWR, IORD, and IOWR) to interface memory and I/O devices (fan, heater, and LCD) in our MCTS project.