The 8085/8080A based Microcomputer Systems

The 8085 microprocessor:

→ It is an 8 bit General purpose processor capable of addressing 64K of memory.

→ It has 40 pins, requires a +5V single power supply and operates with a 3 MHz single phase clock.

→ 8080A is its predecessor.

→ The instruction set of 8085 is compatible with that of 8080A. which means programs written for 8080A will be executed by 8085 but 8085 programs cannot be executed on 8080A.

→ 8080A and 8085 IC are not pin compatible.

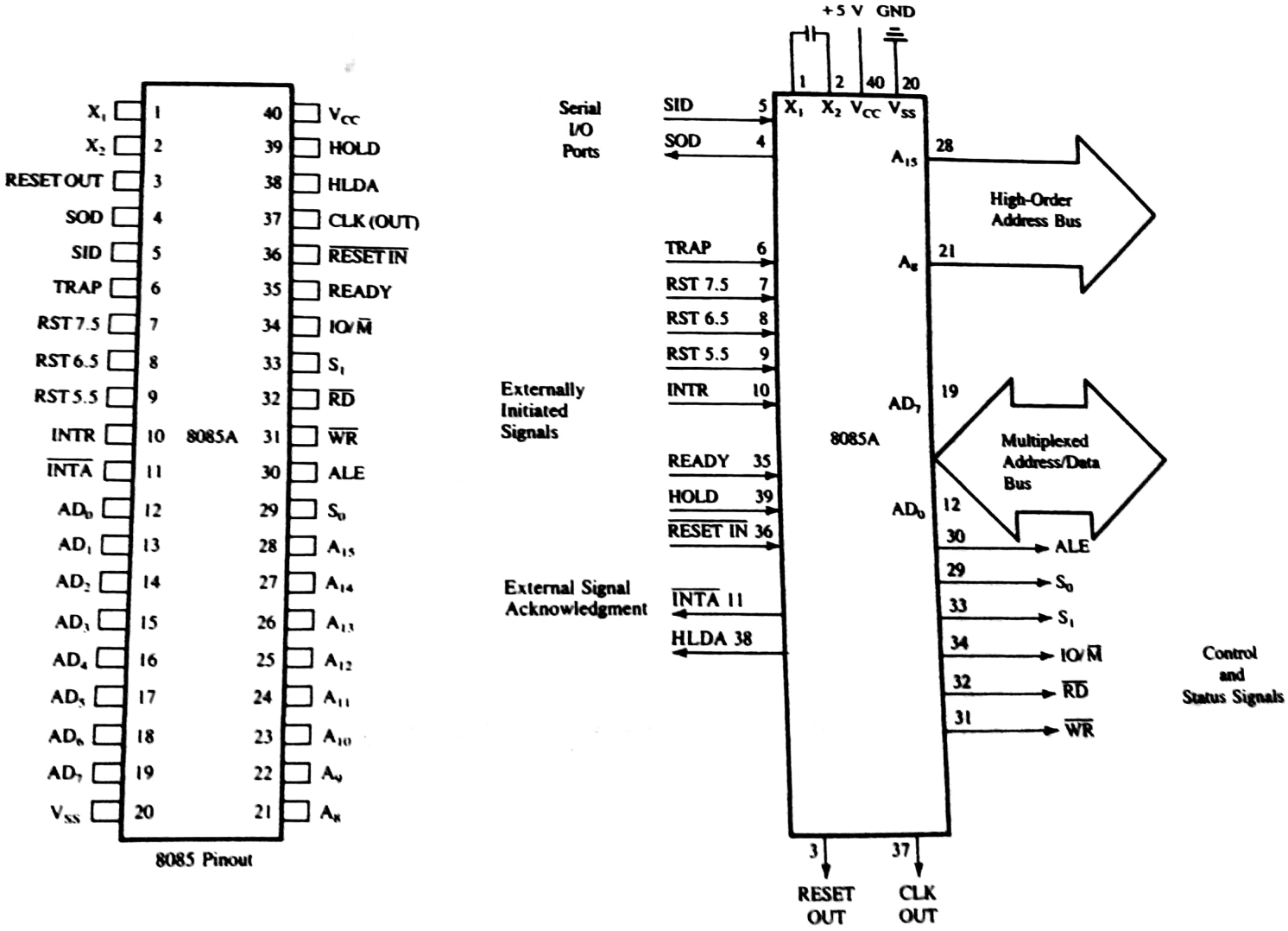

FIGURE 4.1
The 8085 Microprocessor Pinout and Signals

→ All the signals can be classified into six groups :

(1) address bus

(2) data bus

(3) control and status bus

(4) power supply and frequency signals

(5) interrupts and peripheral initiated signals

(6) serial I/O ports.

→ Address Bus

The 8085 has eight signal lines A_{15} - A_8, which are unidirectional and used as high order address bus

→ Multiplexed Address / Data Bus :

The single lines AD_7 to AD_0 are bidirectionals. They serve a dual purpose.

They are used as low order address bus as well as data bus.

In executing an instruction, during the earlier part of the cycle, these lines are used as the low-order address bus.

During the later part of the cycle, they are used as data bus.

→ Control and Status Signals.

This group of signals includes two control signals ($\overline{RD}$, $\overline{WR}$), three status signals ($IO/\overline{M}$, S_1, S_0) to identify the nature of operation and one special signal (ALE) to indicate the beginning of the operation.

ALE – Address latch enable –

This is a positive going pulse generated every time the 8085 begins an operation (machine cycle).

This indicates the bits on AD_7 - AD_0 are address bits

This signal is used primarily to latch the low order address from the multiplexed bus and generate a seperate set of eight address lines A_7 to A_0.

$\overline{RD}$ → Read : This is a read control signal (active low).
This signal indicates that the selected I/O or memory device is to be read and data are available on data bus

$\overline{WR}$ – Write:

This is a write control signal (active low). This signal indicates that the data on the data bus are to be written into a selected memory or I/O location.

$IO/\overline{M}$:

This is a status signal used to differentiate between I/O and memory operations. When it is high, it indicate I/O operation when it is low, it indicates a memory operation. This signal is combined with $\overline{RD}$ (Read) and $\overline{WR}$ (write) to generate I/O and memory control signals.

S_1 and S_0 :

These status signals, similar to $IO/\overline{M}$ can identify various operations, but these are rarely used in small systems.

TABLE 4.1
8085 Machine Cycle Status and Control Signals

Machine Cycle	Status			Control Signals
	$IO/\overline{M}$	S_1	S_0	
Opcode Fetch	0	1	1	$\overline{RD} = 0$
Memory Read	0	1	0	$\overline{RD} = 0$
Memory Write	0	0	1	$\overline{WR} = 0$
I/O Read	1	1	0	$\overline{RD} = 0$
I/O Write	1	0	1	$\overline{WR} = 0$
Interrupt Acknowledge	1	1	1	$\overline{INTA} = 0$
Halt	Z	0	0	$\overline{RD}$, $\overline{WR}$ = Z and $\overline{INTA} = 1$
Hold	Z	X	X	
Reset	Z	X	X	

NOTE: Z = Tri-state (high impedance)
X = Unspecified

TABLE 4.2
8085 Interrupts and Externally Initiated Signals

☐ INTR (Input)	Interrupt Request: This is used as a general-purpose interrupt; it is similar to the INT signal of the 8080A.
☐ $\overline{\text{INTA}}$ (Output)	Interrupt Acknowledge: This is used to acknowledge an interrupt.
☐ RST 7.5 (Inputs) RST 6.5 RST 5.5	Restart Interrupts: These are vectored interrupts that transfer the program control to specific memory locations. They have higher priorities than the INTR interrupt. Among these three, the priority order is 7.5, 6.5, and 5.5.
☐ TRAP (Input)	This is a nonmaskable interrupt and has the highest priority.
☐ HOLD (Input)	This signal indicates that a peripheral such as a DMA (Direct Memory Access) controller is requesting the use of the address and data buses.
☐ HLDA (Output)	Hold Acknowledge: This signal acknowledges the HOLD request.
☐ READY (Input)	This signal is used to delay the microprocessor Read or Write cycles until a slow-responding peripheral is ready to send or accept data. When this signal goes low, the microprocessor waits for an integral number of clock cycles until it goes high.

- $\overline{RESET}$ $\overline{IN}$: When the signal on this pin goes low, the program counter is set to zero, the buses are tri-stated, and the MPU is reset.
- RESET OUT: This signal indicates that the MPU is being reset. The signal can be used to reset other devices.

Serial I/O ports:

→ The 8085 has two signals to implement serial transmission: SID (Serial Input Data) and SOD (Serial Output Data)

Bus Timings:

→ To understand the functions of various signals of 8085, we must examine the timings of these signals in relation to the system clock.

Ex:-
Illustrate the timing of data flow when the instruction code 0100 1111 (4FH – MOV C, A), stored in memory location 2005H is being fetched.

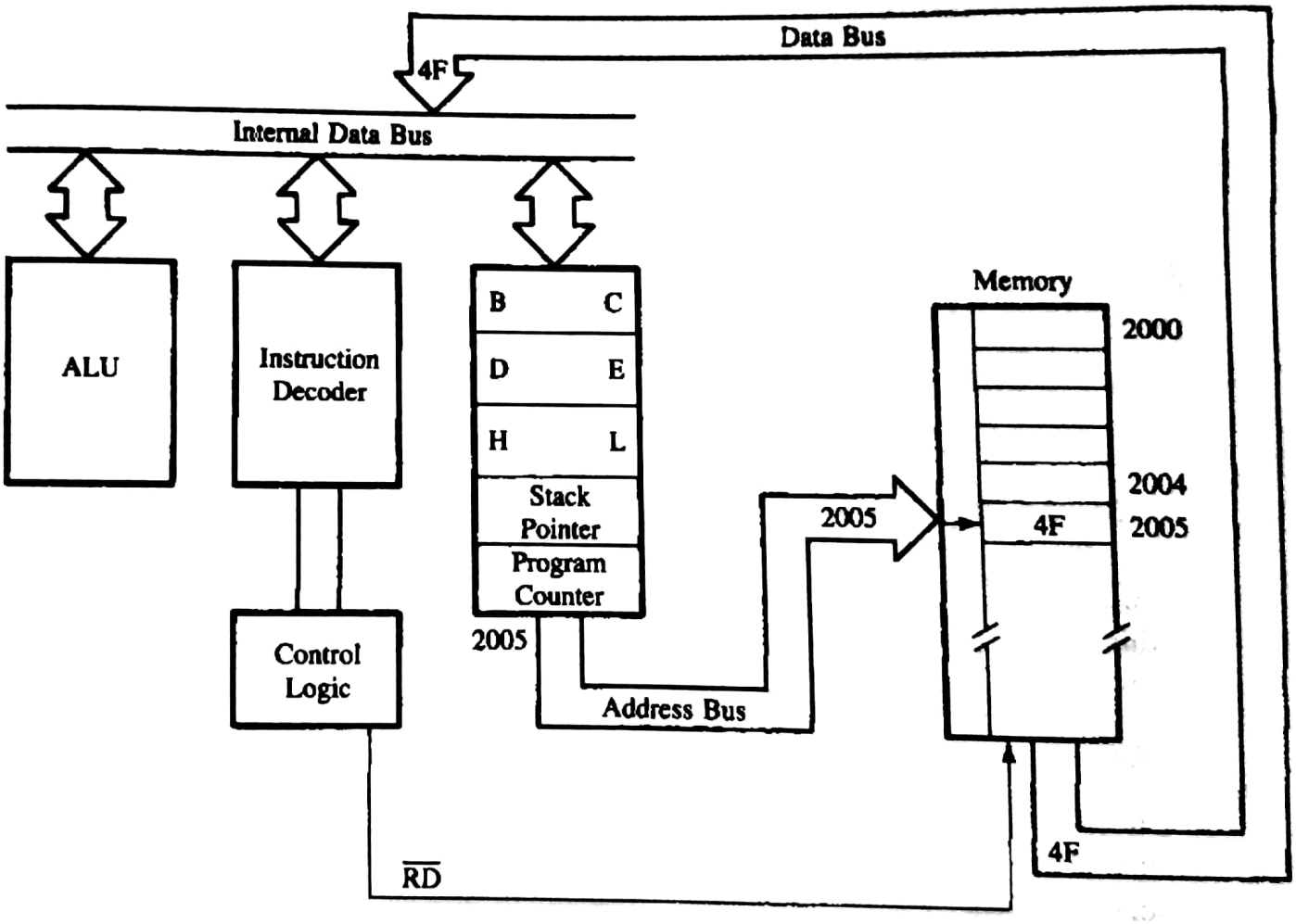

FIGURE 4.2
Data Flow from Memory to the MPU

To fetch the byte, the MPU performs the following steps:

Step 1: The PC places the 16 bit memory address on the address bus.

At T_1 → the higher order memory address 20H is placed on the address lines A_{15} - A_8

→ the low order memory address 05H is placed on the bus AD_7 - AD_0

→ ALE goes high.

→ IO/$\overline{M}$ goes low, indicating this is a memory related operation.

memory read cycle

Step 2 : The control unit sends the control signal $\overline{RD}$ to enable memory chip.

$\overline{RD}$ is sent out during clock period T_2, thus enabling memory chip

$\overline{RD}$ signal is active for two clock periods

Step 3 : The byte from memory location is placed on data bus ($AD_7 - AD_0$).

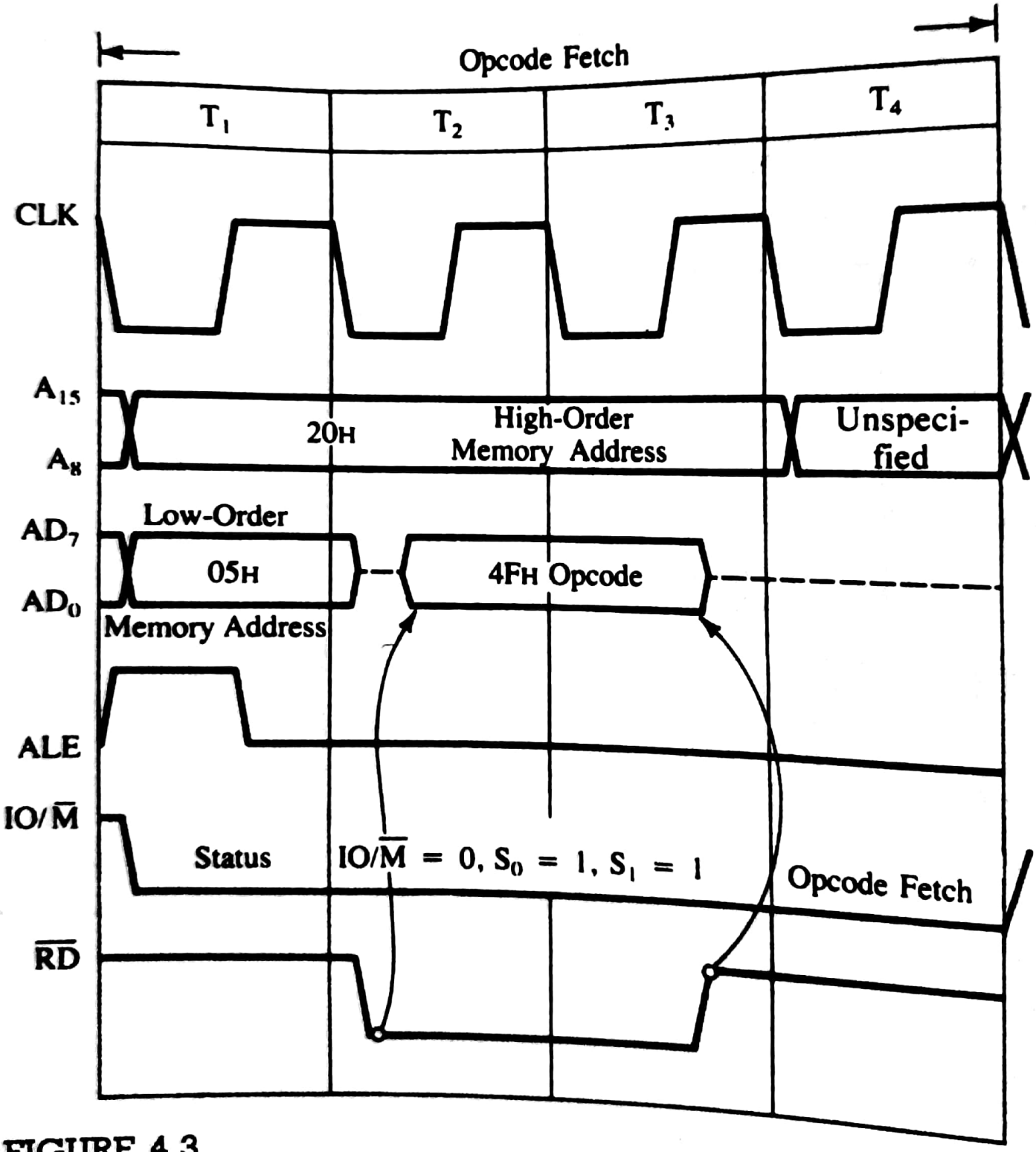

FIGURE 4.3
Timing: Transfer of Byte from Memory to MPU

Demultiplexing the Bus $AD_7 - AD_0$:

→ The need for demultiplexing the bus $AD_7 - AD_0$ becomes easier to understand after examining the previous timing diagram.

→ The address on the high order bus (20H) remains on the bus for three clock periods. The low order address (05H) is lost after the first clock period.

→ This address needs to be latched and used for identifying the memory location.

If the bus $AD_7 - AD_0$ is used to identify the memory location (2005H), the address will change to 204FH after the first clock period.

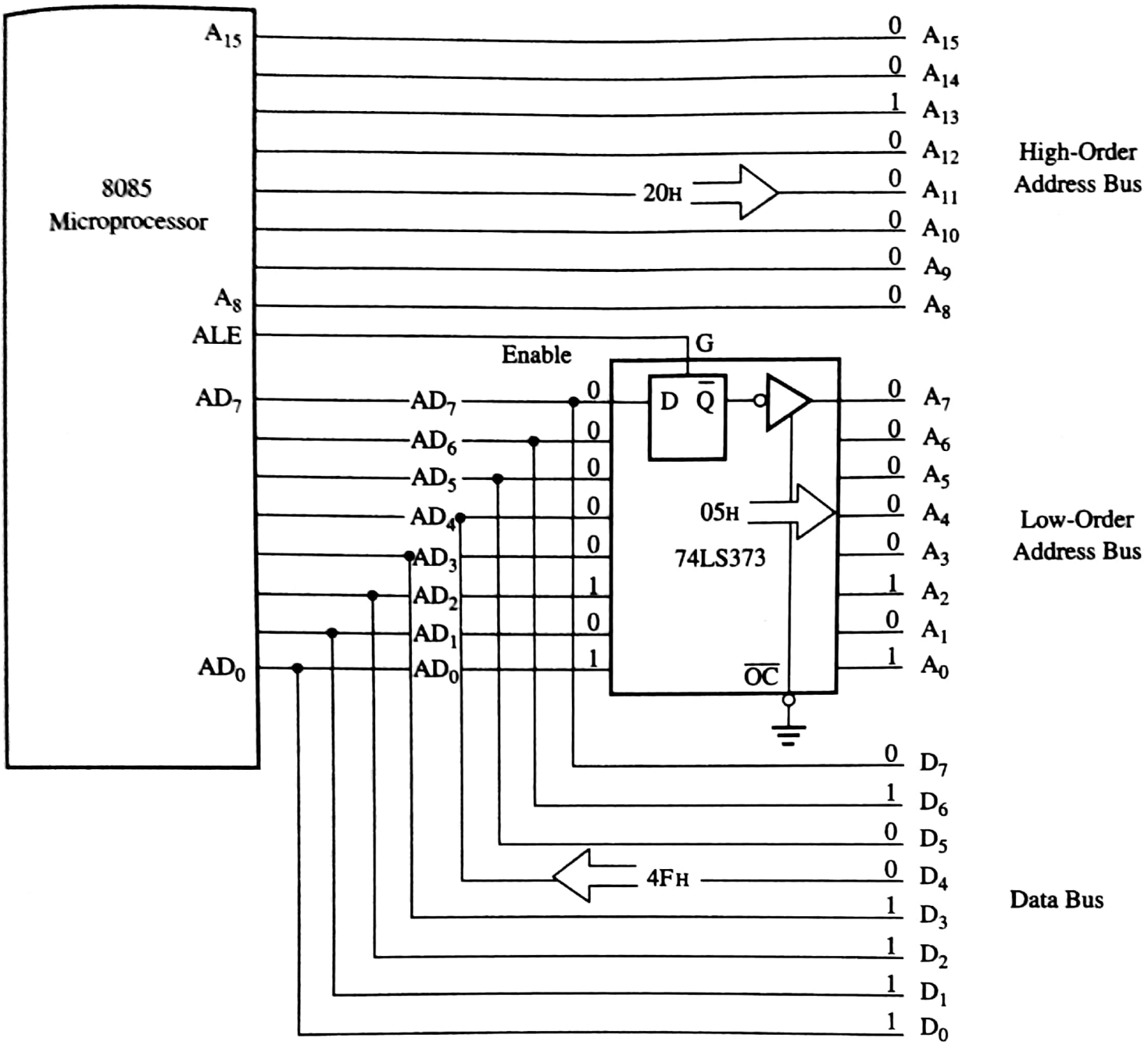

FIGURE 4.4
Schematic of Latching Low-Order Address Bus

The figure shows a schematic that uses a latch and ALE signal to demultiplex the bus.

→ Three bus $AD_7 - AD_0$ is connected as the input to the latch 74LS 373.

The ALE signal is connected to the Enable (G) pin of the latch and the output control ($\overline{OC}$) signal of latch is grounded.

→ When ALE goes high, the latch is transparent; that means the output changes according to input data.

During T_1, the output of latch is 05H.

When the ALE goes low, the low data byte 05H is latched until the next ALE and the O/P of latch represents the low order address bus $A_7 - A_0$, after the latching operation.

After examining the timing diagram, we can make the following observations.

1. The machine code 4FH (0100 1111) is a one byte instruction that copies contents of accumulator A into register C.

2. The 8085 microprocessor requires one external operation, fetching a machine code (opcode) from memory location 2005H.

3. The entire operation - fetching, decoding and execution - requires four clock periods.

We define three terms instruction cycle, machine cycle and T-state:

Instruction Cycle: It is defined as the time required to complete the execution of an instruction. In 8085, instruction cycle consists of one to six machine cycles or one to six operations.

Machine Cycle: It is defined as the time required to complete one operation of accessing memory, I/O, or acknowledging an external request. This consists of three to six T-states.

T-State : T-state is defined as one subdivision of the operation performed in one clock period.

These subdivisions are internal states synchronized with the system clock. Each T-state is precisely equal to one clock period. The terms T-state and clock period are often used synonymously.

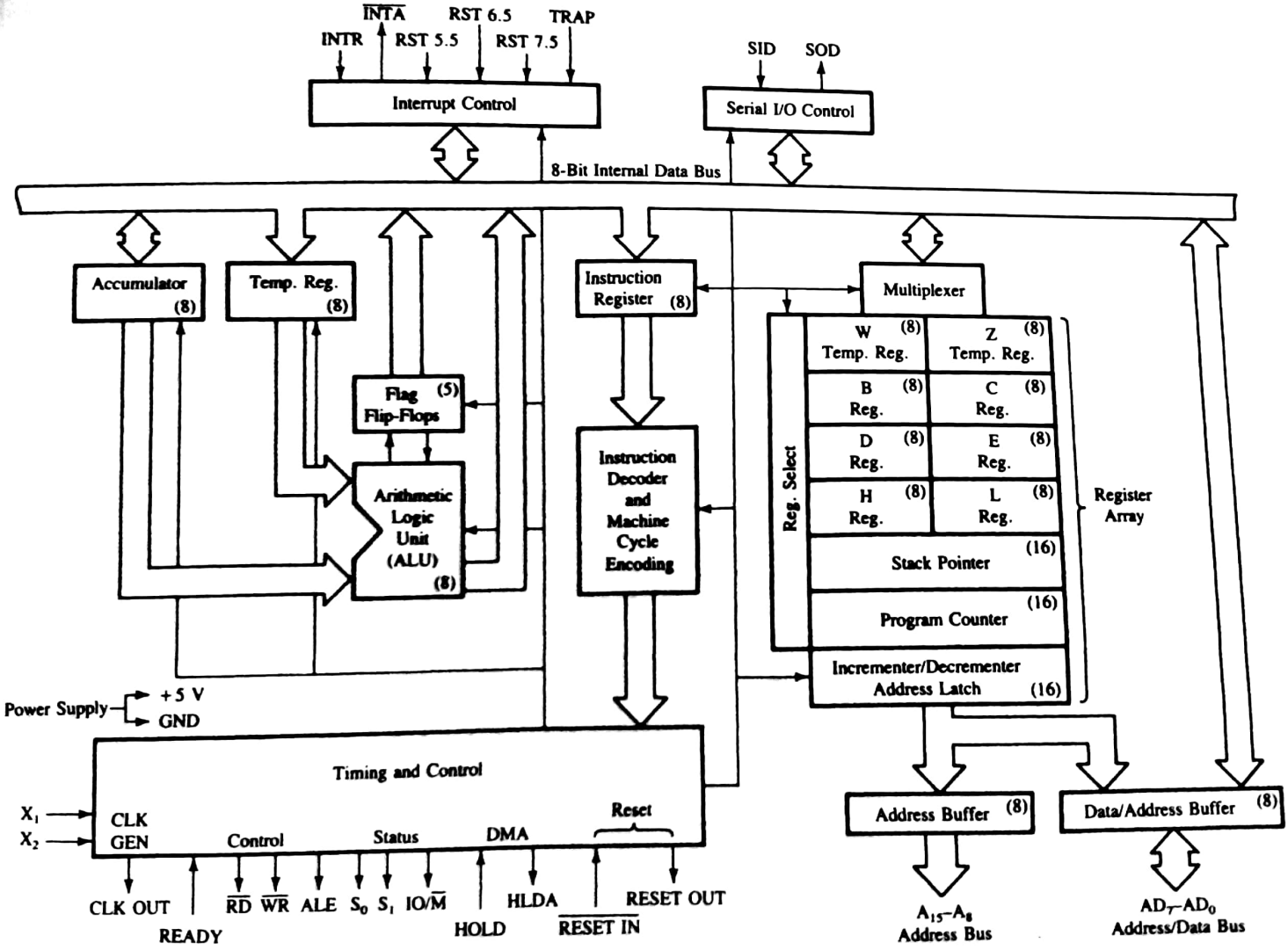

FIGURE 4.7
The 8085A Microprocessor: Functional Block Diagram

NOTE: The 8085A microprocessor is commonly known as the 8085.

SOURCE: Intel Corporation, *Embedded Microprocessors* (Santa Clara, Calif.: Author, 1994), pp. 1-11.

THE ALU

The arithmetic/logic unit performs the computing functions; it includes the accumulator, the temporary register, the arithmetic and logic circuits, and five flags. The temporary register is used to hold data during an arithmetic/logic operation. The result is stored in the accumulator, and the flags (flip-flops) are set or reset according to the result of the operation.

The flags are affected by the arithmetic and logic operations in the ALU. In most of these operations, the result is stored in the accumulator. Therefore, the flags generally reflect data conditions in the accumulator—with some exceptions. The descriptions and conditions of the flags are as follows:

- **S—Sign flag:** After the execution of an arithmetic or logic operation, if bit D_7 of the result (usually in the accumulator) is 1, the Sign flag is set. This flag is used with signed numbers. In a given byte, if D_7 is 1, the number will be viewed as a negative number; if it is 0, the number will be considered positive. In arithmetic operations with signed numbers, bit D_7 is reserved for indicating the sign, and the remaining seven bits are used to represent the magnitude of a number. However, this flag is irrelevant for the operations of unsigned numbers. Therefore, for unsigned numbers, even if bit D_7 of a result is 1 and the flag is set, it does not mean the result is negative. (See Appendix A2 for a discussion of signed numbers.)
- **Z—Zero flag:** The Zero flag is set if the ALU operation results in 0, and the flag is reset if the result is not 0. This flag is modified by the results in the accumulator as well as in the other registers.
- **AC—Auxiliary Carry flag:** In an arithmetic operation, when a carry is generated by digit D_3 and passed on to digit D_4, the AC flag is set. The flag is used only internally for BCD (binary-coded decimal) operations and is not available for the programmer to change the sequence of a program with a jump instruction.
- **P—Parity flag:** After an arithmetic or logical operation, if the result has an even number of 1s, the flag is set. If it has an odd number of 1s, the flag is reset. (For example, the data byte 0000 0011 has even parity even if the magnitude of the number is odd.)
- **CY—Carry flag:** If an arithmetic operation results in a carry, the Carry flag is set; otherwise it is reset. The Carry flag also serves as a borrow flag for subtraction.

The bit positions reserved for these flags in the flag register are as follows:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S	Z		AC		P		CY

Among the five flags, the AC flag is used internally for BCD arithmetic; the instruction set does not include any conditional jump instructions based on the AC flag. Of the remaining four flags, the Z and CY flags are those most commonly used.

TIMING AND CONTROL UNIT

This unit synchronizes all the microprocessor operations with the clock and generates the control signals necessary for communication between the microprocessor and peripherals. The control signals are similar to a sync pulse in an oscilloscope. The $\overline{RD}$ and $\overline{WR}$ signals are sync pulses indicating the availability of data on the data bus.

INSTRUCTION REGISTER AND DECODER

The instruction register and the decoder are part of the ALU. When an instruction is fetched from memory, it is loaded in the instruction register. The decoder decodes the instruction and establishes the sequence of events to follow. The instruction register is not programmable and cannot be accessed through any instruction.

REGISTER ARRAY

The programmable registers were discussed in the last chapter. Two additional registers, called temporary registers W and Z, are included in the register array. These registers are used to hold 8-bit data during the execution of some instructions. However, because they are used internally, they are not available to the programmer.

Instruction classification

→ An instruction is a binary pattern designed inside the microprocessor to perform a specific function.

→ The entire group of instructions, called the instruction set, determines what functions the microprocessor can perform.

→ 8085 microprocessor includes the instruction set of its predecessor, 8080A, plus two additional instructions.

The 8085 Instruction Set:

→ five functional categories:

1. Data transfer (copy) operations
2. Arithmetic operations
3. logical operations
4. branching operations
5. machine-control operations.

Data Transfer (Copy) operations :

This group of instructions copies data from a location called a source to another location, called destination, without modifying the contents of the source.

→	Types	Example
	Between registers →	Copy contents of Reg E into Reg D
	Specific data byte to a register or memory location →	Load reg. E with data byte 32H
	between a memory location and a register →	From memory location 2000H to a reg B
	between I/O device and Accumulator →	From an input keyboard into Acc A.

Arithmetic operations

→ These instructions perform operations such as addition, subtraction, increment, decrement.

(1) Addition:

Any 8 bit number, or the contents of a register, or the contents of a memory location can be added to the contents of Accumulator, and the sum is stored in the accumulator.

Note: No two 8 bit registers can be added directly.

(DAD is an exception: it adds 16 bit data directly in register pairs)

(2) Subtraction:

Any 8 bit number, or contents of a register or the contents of a memory location can be subtracted from the contents of accumulator and the results stored in accumulator

The subtraction is performed in 2's complement and the result if negative are expressed in 2's complement.

No two registers can be subtracted directly.

(3) Increment / Decrement :

The 8 bit contents of a register or a memory location can be incremented or decremented by 1.

Similarly, the 16 bit contents of a register pair (such as BC) can be incremented or decremented by 1.

The increment and decrement operations can be performed in any one of the registers or in a memory location.

(6) Logical operations

(1) AND, OR, Ex-OR:

Any 8 bit number, or the contents of a register or of a memory location can be logically ANDed, ORed, or Ex-ORed with the contents of ACC. and result is stored in ACC.

(2) Rotate:

Each bit in the ACC. can be shifted either left or right to the next position.

(3) Compare:

Any 8 bit number, the contents of a register or a memory location can be compared for equality, greater than, less than with the contents of ACC.

(4) complements :

The contents of Acc. can be complemented ; all 0's are replaced by 1's and all 1's are replaced by 0's.

and

Branching Operations

→ This group of instructions alter the sequence of program execution either conditionally or unconditionally.

(1) Jump:

Conditional jumps are very important in decision making process in programming. These instructions test for a certain condition (eg Z or CY flag) and alter the program sequence when the condition is met.

The instruction set also includes an instruction called unconditional Jump.

(2) Call, Return, Restart:

These instructions change the sequence of a program either by calling a subroutine or returning from a subroutine.

The conditional call and Return instructions can also test condition flags.

Instruction, Data format, and Storage

→ An instruction is a command to the microprocessor to perform a given task on a specified data.

Each instruction has two parts:

① operation code (opcode): task to be performed
② operands : data to be operated on.

The operand can be specified in various ways: It may include 8bit (or 16bit) data, an internal register, a memory location, or an 8bit or 16bit address. In some instructions, the operand is implicit.

Instruction word size:

→ 8085 instruction set is classified into three groups according to word size or byte size

① 1-byte instructions
② 2-byte instructions
③ 3-byte instructions.

One-byte instructions:

A 1-byte instruction includes opcode and operand in same byte.

Two byte instructions:

In 2-byte instruction, the first byte specifies the op code and the second byte specifies the operands.

These instructions would require two memory locations to store binary code.

Three Byte Instructions:

In 3-byte instructions, the first byte specifies the opcode, the following two bytes specify the 16 bit address.

These instructions would require three memory locations to store the binary code.

Machine Control Operations:

These instructions control machine functions such as Halt, Interrupt or do nothing.

Opcode format:

→ To understand the opcodes, we need to examine how an instruction is designed into the microprocessor

→ In the design of 8085 microprocessor chip, all operations, registers, status flags are identified with a specific code.

Ex:- Internal Registers

Code	Register
000	B
001	C
010	D
011	E
100	H
101	L
111	A
110	Reserved for memory related operations

Code	Register Pairs
00	BC
01	DE
10	HL
11	AF.OR SP

Clues on how to recognize the no. of bytes in an instruction of the 8085 microprocessor

1. One-byte instruction:

A mnemonic followed by letter (or two letters) representing the registers is a one byte instruction

Ex:- MOV A, B, DCX SP, RRC

Note: Instruction with implicit registers are also one byte.

2. Two byte instructions:

A mnemonic followed by 8 bit (byte) is a two byte instruction.

Ex:- MVI A, 8bit ; ADI 8bit

3. Three byte instruction:

A mnemonic followed by 16 bit (also terms such as addr, or dble) is a three byte instruction.

Ex:- LXI B, 16 bit (dble)

JNZ 16 bit (addr)

CALL 16 bit (addr)

How does the microprocessor differentiate between data and instruction code?

Ans:

The microprocenor interprets the first byte it fetches as an opcode.

When 8085 is reset, its PC is cleared to 0000H, and it fetches the first code from the location 0000H.

Important Note: In the 8085 processor, data transfer instructions do not affect the flags.

Opcode	Operand	Description
MOV	Rd,Rs*	Move ☐ This is a 1-byte instruction ☐ Copies data from source register Rs to destination register Rd
MVI	R,8-bit*	Move Immediate ☐ This is a 2-byte instruction ☐ Loads the 8 bits of the second byte into the register specified
OUT	8-bit port address	Output to Port ☐ This is a 2-byte instruction ☐ Sends (copies) the contents of the accumulator (A) to the output port specified in the second byte
IN	8-bit port address	Input from Port ☐ This is a 2-byte instruction ☐ Accepts (reads) data from the input port specified in the second byte, and loads into the accumulator
HLT		Halt ☐ This is a 1-byte instruction ☐ The processor stops executing and enters wait state ☐ The address bus and data bus are placed in high impedance state. No register contents are affected
NOP		No Operation ☐ This is a 1-byte instruction ☐ No operation is performed ☐ Generally used to increase processing time or substitute in place of an instruction. When an error occurs in a program and an instruction needs to be eliminated, it is more convenient to substitute NOP than to reassemble the whole program

ADD : Add	Add the contents of a register.*
ADI : Add Immediate	Add 8-bit data.
SUB : Subtract	Subtract the contents of a register.
SUI : Subtract Immediate	Subtract 8-bit data.
INR : Increment	Increase the contents of a register by 1.
DCR : Decrement	Decrease the contents of a register by 1.

The arithmetic operations Add and Subtract are performed in relation to the contents of the accumulator. However, the Increment or the Decrement operations can be performed in any register. The instructions for these operations are explained below.

INSTRUCTIONS

These arithmetic instructions (except INR and DCR)

1. assume implicitly that the accumulator is one of the operands.
2. modify all the flags according to the data conditions of the result.
3. place the result in the accumulator.
4. do not affect the contents of the operand register.

The instructions INR and DCR

1. affect the contents of the specified register.
2. affect all flags except the CY flag.

The descriptions of the instructions (including INR and DCR) are as follows:

Opcode	Operand	Description
ADD	R†	Add ☐ This is a 1-byte instruction ☐ Adds the contents of register R to the contents of the accumulator
ADI	8-bit	Add Immediate ☐ This is a 2-byte instruction ☐ Adds the second byte to the contents of the accumulator
SUB	R†	Subtract ☐ This is a 1-byte instruction ☐ Subtracts the contents of register R from the contents of the accumulator
SUI	8-bit	Subtract Immediate ☐ This is a 2-byte instruction

- ☐ Subtracts the second byte from the contents of the ac
  mulator

INR **R***

Increment

- ☐ This is a 1-byte instruction
- ☐ Increases the contents of register R by 1
  Caution: All flags except the CY are affected

DCR **R***

Decrement

- ☐ This is a 1-byte instruction
- ☐ Decreases the contents of register R by 1
  Caution: All flags except the CY are affected

LOGIC OPERATIONS

A microprocessor is basically a programmable logic chip. It can perform all the logic functions of the hard-wired logic through its instruction set. The 8085 instruction set includes such logic functions as AND, OR, Ex OR, and NOT (complement). The opcodes of these operations are as follows:*

ANA:	AND	Logically AND the contents of a register.
ANI :	AND Immediate	Logically AND 8-bit data.
ORA:	OR	Logically OR the contents of a register.
ORI :	OR Immediate	Logically OR 8-bit data.
XRA:	X-OR	Exclusive-OR the contents of a register.
XRI :	X-OR Immediate	Exclusive-OR 8-bit data.

All logic operations are performed in relation to the contents of the accumulator. The instructions of these logic operations are described below.

INSTRUCTIONS

The logic instructions

1. implicitly assume that the accumulator is one of the operands.
2. reset (clear) the CY flag. The instruction CMA is an exception; it does not affect any flags.
3. modify the Z, P, and S flags according to the data conditions of the result.
4. place the result in the accumulator.
5. do not affect the contents of the operand register.

Opcode	Operand	Description
ANA	R	Logical AND with Accumulator ☐ This is a 1-byte instruction ☐ Logically ANDs the contents of the register R with the contents of the accumulator ☐ 8085: CY is reset and AC is set
ANI	8-bit	AND Immediate with Accumulator ☐ This is a 2-byte instruction ☐ Logically ANDs the second byte with the contents of the accumulator ☐ 8085: CY is reset and AC is set
ORA	R	Logically OR with Accumulator ☐ This is a 1-byte instruction ☐ Logically ORs the contents of the register R with the contents of the accumulator
ORI	8-bit	OR Immediate with Accumulator ☐ This is a 2-byte instruction ☐ Logically ORs the second byte with the contents of the accumulator
XRA	R	Logically Exclusive-OR with Accumulator ☐ This is a 1-byte instruction ☐ Exclusive-ORs the contents of register R with the contents of the accumulator
XRI	8-bit	Exclusive-OR Immediate with Accumulator ☐ This is a 2-byte instruction ☐ Exclusive-ORs the second byte with the contents of the accumulator
CMA		Complement Accumulator ☐ This is a 1-byte instruction that complements the contents of the accumulator ☐ No flags are affected

BRANCH OPERATIONS

The **branch instructions** are the most powerful instructions because they allow the microprocessor to change the sequence of a program, either unconditionally or under certain test conditions. These instructions are the key to the flexibility and versatility of a computer.

The microprocessor is a sequential machine; it executes machine codes from one memory location to the next. Branch instructions instruct the microprocessor to go to a different memory location, and the microprocessor continues executing machine codes from that new location. The address of the new memory location is either specified explicitly or supplied by the microprocessor or by extra hardware. The branch instructions are classified in three categories:

1. Jump instructions
2. Call and Return instructions
3. Restart instructions

This section is concerned with applications of Jump instructions. The Call and Return instructions are associated with the subroutine technique and will be discussed in Chapter 9; Restart instructions are associated with the interrupt technique and will be discussed in Chapter 12.

The Jump instructions specify the memory location explicitly. They are 3-byte instructions: one byte for the operation code, followed by a 16-bit memory address. Jump instructions are classified into two categories: Unconditional Jump and Conditional Jump.

6.4.1 Unconditional Jump

The 8085 instruction set includes one unconditional Jump instruction. The unconditional Jump instruction enables the programmer to set up continuous loops.

INSTRUCTION

Opcode	Operand	Description
JMP	16-bit	Jump ☐ This is a 3-byte instruction ☐ The second and third bytes specify the 16-bit memory address. However, the second byte specifies the low-order and the third byte specifies the high-order memory address

6.4.3 Conditional Jumps

Conditional Jump instructions allow the microprocessor to make decisions based on certain conditions indicated by the flags. After logic and arithmetic operations, flip-flops (flags) are set or reset to reflect data conditions. The conditional Jump instructions check the flag conditions and make decisions to change or not to change the sequence of a program.

FLAGS

The 8085 flag register has five flags, one of which (Auxiliary Carry) is used internally. The other four flags used by the Jump instructions are

1. Carry flag
2. Zero flag
3. Sign flag
4. Parity flag

Two Jump instructions are associated with each flag. The sequence of a program can be changed either because the condition is present or because the condition is absent. For example, while adding the numbers you can change the program sequence either because the carry is present (JC = Jump On Carry) or because the carry is absent (JNC = Jump On No Carry).

INSTRUCTIONS

All conditional Jump instructions in the 8085 are 3-byte instructions; the second byte specifies the low-order (line number) memory address, and the third byte specifies the high-order (page number) memory address. The following instructions transfer the program sequence to the memory location specified under the given conditions:

Opcode	Operand	Description
JC	16-bit	Jump On Carry (if result generates carry and CY = 1)
JNC	16-bit	Jump On No Carry (CY = 0)
JZ	16-bit	Jump On Zero (if result is zero and Z = 1)
JNZ	16-bit	Jump On No Zero (Z = 0)
JP	16-bit	Jump On Plus (if $D_7 = 0$, and S = 0)
JM	16-bit	Jump On Minus (if $D_7 = 1$, and S = 1)
JPE	16-bit	Jump On Even Parity (P = 1)
JPO	16-bit	Jump On Odd Parity (P = 0)

Addressing Modes:

→ The above instructions are commands to the microprocessor to copy 8 bit data from source to destination.

→ Source can be a register, an input port or an 8 bit number (00H to FFH).

→ Destination can be a register, an output port

→ Sources and destination are infact operands

→ The various formats of specifying the operands are called addressing modes.
8085 has the following addressing modes

1. Immediate Addressing — MVI R, Data
2. Register Addressing — MOV Rd, Rs
3. Direct Addressing — IN/OUT Port #
4. Indirect Addressing — (To be discussed later)

Instructions	Tasks	Addressing Mode
Data transfer (Copy) Instructions		
1. MOV Rd,Rs	Copy (Rs) into (Rd).	Register
2. MVI R,8-bit	Load register R with the 8-bit data.	Immediate
3. IN 8-bit port address	Read data from the input port.	Direct
4. OUT 8-bit port address	Write data in the output port.	Direct
Arithmetic Instructions		
1. ADD R	Add (R) to (A).	Register
2. ADI 8-bit	Add 8-bit data to (A).	Immediate
3. SUB R	Subtract (R) from (A).	Register
4. SUI 8-bit	Subtract 8-bit data from (A).	Immediate

*R, Rs, and Rd represent any one of the 8-bit registers—A, B, C, D, E, H, and L.

		Register
5. INR R	Increment (R).	Register
6. DCR R	Decrement (R).	
Logic Instructions		
1. ANA R	Logically AND (R) with (A).	Register
2. ANI 8-bit	Logically AND 8-bit data with (A).	Immediate
3. ORA R	Logically OR (R) with (A).	Register
4. ORI 8-bit	Logically OR 8-bit data with (A).	Immediate
5. XRA R	Logically Exclusive-OR (R) with (A).	Register
6. XRI 8-bit	Logically Exclusive-OR 8-bit data with (A).	Immediate
7. CMA	Complement (A).	
Branch Instructions		
1. JMP 16-bit	Jump to 16-bit address unconditionally.	Immediate
2. JC 16-bit	Jump to 16-bit address if the CY flag is set.	Immediate
3. JNC 16-bit	Jump to 16-bit address if the CY flag is reset.	Immediate
4. JZ 16-bit	Jump to 16-bit address if the Zero flag is set.	Immediate
5. JNZ 16-bit	Jump to 16-bit address if the Zero flag is reset.	Immediate
6. JP 16-bit	Jump to 16-bit address if the Sign flag is reset.	Immediate
7. JM 16-bit	Jump to 16-bit address if the Sign flag is set.	Immediate
8. JPE 16-bit	Jump to 16-bit address if the Parity flag is set.	Immediate
9. JPO 16-bit	Jump to 16-bit address if the Parity flag is reset.	Immediate
Machine Control Instructions		
1. NOP	No operation.	
2. HLT	Stop processing and wait.	

DATA TRANSFER GROUP

Move

MOV	A,A	7F
	A,B	78
	A,C	79
	A,D	7A
	A,E	7B
	A,H	7C
	A,L	7D
	A,M	7E
MOV	B,A	47
	B,B	40
	B,C	41
	B,D	42
	B,E	43
	B,H	44
	B,L	45
	B,M	46
MOV	C,A	4F
	C,B	48
	C,C	49
	C,D	4A
	C,E	4B
	C,H	4C
	C,L	4D
	C,M	4E
MOV	D,A	57
	D,B	50
	D,C	51
	D,D	52
	D,E	53
	D,H	54
	D,L	55
	D,M	56

Move (cont)

MOV	E,A	5F
	E,B	58
	E,C	59
	E,D	5A
	E,E	5B
	E,H	5C
	E,L	5D
	E,M	5E
MOV	H,A	67
	H,B	60
	H,C	61
	H,D	62
	H,E	63
	H,H	64
	H,L	65
	H,M	66
MOV	L,A	6F
	L,B	68
	L,C	69
	L,D	6A
	L,E	6B
	L,H	6C
	L,L	6D
	L,M	6E
MOV	M,A	77
	M,B	70
	M,C	71
	M,D	72
	M,E	73
	M,H	74
	M,L	75
XCHG		EB

Move Immediate

MVI	A, byte	3E
	B, byte	06
	C, byte	0E
	D, byte	16
	E, byte	1E
	H, byte	26
	L, byte	2E
	M, byte	36

Load Immediate

LXI	B, dble	01
	D, dble	11
	H, dble	21
	SP, dble	31

Load/Store

LDAX B	0A
LDAX D	1A
LHLD adr	2A
LDA adr	3A
STAX B	02
STAX D	12
SHLD adr	22
STA adr	32

byte = constant, or logical/arithmetic expression that evaluates to an 8-bit data quantity (Second byte of 2-byte instructions)

dble = constant, or logical/arithmetic expression that evaluates to a 16-bit data quantity (Second and Third bytes of 3-byte instructions)

adr = 16-bit address (Second and Third bytes of 3-byte instructions)

* = all flags (C, Z, S, P, AC) affected

** = all flags except CARRY affected (exception: INX and DCX affect no flags)

† = only CARRY affected

ARITHMETIC AND LOGICAL GROUP

Add*

ADD	A	87
	B	80
	C	81
	D	82
	E	83
	H	84
	L	85
	M	86
ADC	A	8F
	B	88
	C	89
	D	8A
	E	8B
	H	8C
	L	8D
	M	8E

Subtract*

SUB	A	97
	B	90
	C	91
	D	92
	E	93
	H	94
	L	95
	M	96
SBB	A	9F
	B	98
	C	99
	D	9A
	E	9B
	H	9C
	L	9D
	M	9E

Double Add †

DAD	B	09
	D	19
	H	29
	SP	39

Increment**

INR	A	3C
	B	04
	C	0C
	D	14
	E	1C
	H	24
	L	2C
	M	34
INX	B	03
	D	13
	H	23
	SP	33

Decrement**

DCR	A	3D
	B	05
	C	0D
	D	15
	E	1D
	H	25
	L	2D
	M	35
DCX	B	0B
	D	1B
	H	2B
	SP	3B

Specials

DAA*	27
CMA	2F
STC†	37
CMC†	3F

Rotate †

RLC	07
RRC	0F
RAL	17
RAR	1F

Logical*

ANA	A	A7
	B	A0
	C	A1
	D	A2
	E	A3
	H	A4
	L	A5
	M	A6
XRA	A	AF
	B	A8
	C	A9
	D	AA
	E	AB
	H	AC
	L	AD
	M	AE
ORA	A	B7
	B	B0
	C	B1
	D	B2
	E	B3
	H	B4
	L	B5
	M	B6
CMP	A	BF
	B	B8
	C	B9
	D	BA
	E	BB
	H	BC
	L	BD
	M	BE

Arith & Logical Immediate

ADI byte	C6
ACI byte	CE
SUI byte	D6
SBI byte	DE
ANI byte	E6
XRI byte	EE
ORI byte	F6
CPI byte	FE

BRANCH CONTROL GROUP

Jump

JMP adr	C3
JNZ adr	C2
JZ adr	CA
JNC adr	D2
JC adr	DA
JPO adr	E2
JPE adr	EA
JP adr	F2
JM adr	FA
PCHL	E9

Call

CALL adr	CD
CNZ adr	C4
CZ adr	CC
CNC adr	D4
CC adr	DC
CPO adr	E4
CPE adr	EC
CP adr	F4
CM adr	FC

Return

RET	C9
RNZ	C0
RZ	C8
RNC	D0
RC	D8
RPO	E0
RPE	E8
RP	F0
RM	F8

Restart

RST	0	C7
	1	CF
	2	D7
	3	DF
	4	E7
	5	EF
	6	F7
	7	FF

I/O AND MACHINE CONTROL

Stack Ops

PUSH	B	C5
	D	D5
	H	E5
	PSW	F5
POP	B	C1
	D	D1
	H	E1
	PSW*	F1
XTHL		E3
SPHL		F9

Input/Output

OUT byte	D3
IN byte	DB

Control

DI	F3
EI	FB
NOP	00
HLT	76

New Instructions (8085 Only)

RIM	20
SIM	30

ASSEMBLER REFERENCE (Cont.)

Pseudo Instruction

General:

ORG
END
EQU
SET
DS
DB
DW

Macros:

MACRO
ENDM
LOCAL
REPT
IRP
IRPC
EXITM

Relocation:

ASEG	NAME
DSEG	STKLN
CSEG	STACK
PUBLIC	MEMORY
EXTRN	

Conditional Assembly:

IF
ELSE
ENDIF

RESTART TABLE

Name	Code	Restart Address
RST 0	C7	0000_{16}
RST 1	CF	0008_{16}
RST 2	D7	0010_{16}
RST 3	DF	0018_{16}
RST 4	E7	0020_{16}
TRAP	Hardware* Function	0024_{16}
RST 5	EF	0028_{16}
RST 5 5	Hardware* Function	$002C_{16}$
RST 6	F7	0030_{16}
RST 6 5	Hardware* Function	0034_{16}
RST 7	FF	0038_{16}
RST 7 5	Hardware* Function	$003C_{16}$

*NOTE The hardware functions refer to the on-chip interrupt feature of the 8085 only