Noida Institute of Engineering and Technology, Greater Noida

Microprocessor

(ACSE0405)

Unit: 2

8085 Instructions And Programming Techniques

B.Tech 4th Semester

Manish Kumar
Assistant Professor
ECE

Faculty Introduction

I received my Master of Technology from DBIT Banglore and Bachelor of Technology from Bharath Institute of Science and Technology, Bharath University, Chennai. Presently I am working as an Assistant Professor in Electronics & Communication Engineering department of Noida Institute of Engineering & Technology, Greater Noida, India. My research interest includes Antennas, VLSI and Embedded System. I published National & International Journals in repute. I attended various training programmes in the area of interest. I am a Life member of International Association of Engineers (**IAENG**). I attended various training programs in the area of Mechatronics (Pneumatics, Electro pneumatics & PLC) & Embedded System.

Evaluation Scheme

Sl. No.	Subject Codes	Subject Name	Periods			Evaluation Scheme				End Semester		Total	Credit
			L	T	P	CT	TA	TOTAL	PS	TE	PE		
1	AAS0402	Engineering Mathematics-IV	3	1	0	30	20	50		100		150	4
2	AASL0401	Technical Communication	2	1	0	30	20	50		100		150	3
3	ACSE0405	Microprocessor	3	0	0	30	20	50		100		150	3
4	ACSE0403A	Operating Systems	3	0	0	30	20	50		100		150	3
5	ACSE0404	Theory of Automata and Formal Languages	3	0	0	30	20	50		100		150	3
6	ACSE0401	Design and Analysis of Algorithm	3	1	0	30	20	50		100		150	4
7	ACSE0455	Microprocessor Lab	0	0	2				25		25	50	1
8	ACSE0453A	Operating Systems Lab	0	0	2				25		25	50	1
9	ACSE0451	Design and Analysis of Algorithm Lab	0	0	2				25		25	50	1
10	ACSE0459	Mini Project using Open Technology	0	0	2				50			50	1
11	ANC0402 / ANC0401	Environmental Science*/ Cyber Security*(Non Credit)	2	0	0	30	20	50		50		100	0
12		MOOCs** (For B.Tech. Hons. Degree)											
		GRAND TOTAL										**1100**	**24**

Subject Syllabus

Course Contents / Syllabus		
UNIT-I	**8085 Microprocessor**	**8 Hours**
Introduction to Microprocessor, Microprocessor evolutionandtypes, Microprocessorarchitectureand its operation, Logic devices for interfacing,Pin diagram and internal architecture of 8085 Microprocessor,Example of an 8085 based computer, Instructionanddataflow,timerandtimingdiagram,interruptandmachinecycle, Addressingmodes.		
UNIT-II	**8085 Instructions and Programming Techniques**	**8 Hours**
Instructionsets,InstructionClassification:datatransfer operations, arithmeticoperations,logicaloperations,branchingoperations,machinecontrolandassemblerdirectives, writing assembly language programs, Programming techniques: looping, counting and indexing		
UNIT-III	**Code Conversion and BCD Arithmetic**	**8 Hours**
Counter and time delays, Illustrative program: Hexadecimal counter, zero-to-nine, (module ten) counter, generating pulse waveforms, Stack, Subroutine, Restart, Conditional call and return instructions, Advance subroutine concepts, Program: BCD-to-Binary conversion, Binary-to-BCD conversion, BCD-to-Seven segment code converter, Binary-to-ASCII and ASCII-to-Binary code conversion, BCD Addition, BCD Subtraction, Introduction to Advance instructions and Application, Multiplication		
UNIT-IV	**Interfacing of I/O devices**	**8 Hours**
Basic interfacing concepts,Memoryinterfacing,Interfacing output displays, Interfacing input devices, Memory mapped I/O,Interfacing keyboard and seven segment displays, The 8085 Interrupts, 8085 vector interrupts,8259 programmable interrupt controller,		
UNIT-V	**Programmable Peripheral IC's and 8086 Microprocessor**	**8 Hours**
Peripheral Devices: 8255 programmable peripheral interface,8253/8254 programmable timer/counter, 8237 DMA Controller, 8251 USART and RS232C.Introduction to 8086 microprocessors: Architecture of 8086 (Pin diagram, Functional block diagram, register organization), Addressing Modes		

Branch Wise Application

- **Microprocessor**, any of a type of miniature electronic device that contains the arithmetic, logic, and control circuitry necessary to perform the functions of a digital computer's central processing unit.
- This kind of integrated circuit can interpret and execute program instructions as well as handle arithmetic operations.

Course Objective

- The objective of this course is to understand basic concepts of Microprocessor based systems and able to do programming in Assembly Language of 8085. They will be able to learn and program Peripheral IC's.

Course Outcome

At the end of this course students will able to:

- Apply a basic concept of digital fundamentals to Microprocessor based personal computer system.
- Analyze a detailed s/w & h/w structure of the Microprocessor.
- Illustrate how the different peripherals (8085/8086) are interfaced with Microprocessor.
- Analyze the properties of Microprocessors (8085/8086).
- Evaluate the data transfer information through serial & parallel ports.

Program Outcomes

- **Program Outcomes** are narrow statements that describe what the students are expected to know and would be able to do upon the graduation.

- These relate to the skills, knowledge, and behavior that students acquire through the programmed.

1. Engineering knowledge
2. Problem analysis
3. Design/development of solutions
4. Conduct investigations of complex problems
5. Modern tool usage
6. The engineer and society
7. Environment and sustainability
8. Ethics
9. Individual and team work
10. Communication
11. Project management and finance
12. Life-long learning

CO-PO Mapping

Course Outcome	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
ACSE0405.1	3	1	2		1	-	-	-	-	-	-	3
ACSE0405.2	3	1	-		1	-	-	-	-	-	-	2
ACSE0405.3	3	1	3		1	-	-	-	-	-	-	2
ACSE0405.4	3	1	-		1	-	-	-	-	-	-	2
ACSE0405.5	3	1	2		1	-	-	-	-	-	-	2
Average	3	1	2.3		1	-	-	-	-	-	-	2.1

PSOs Mapping

Sr. No.	Course Outcome	PSO1	PSO2	PSO3
1	**ACSE0405.1**	3	2	1
2	**ACSE0405.2**	3	2	1
3	**ACSE0405.3**	3	2	1
4	**ACSE0405.4**	3	2	1
5	**ACSE0405.5**	3	2	2
	Average	**3**	**2**	**1.2**

End Semester Question Paper Template

B.Tech (Semester IV Theory Examination 2021-22)			**Total Marks : 100**
Note: Attempt all sections. If require any missing data, then choose suitably. **Time: 3 hours**			
Section A			
1. Attempt all questions in brief.			2 X 10 = 20
Q. No.	Question	Marks	CO
a. to j		2	
Section B			
2. Attempt any three of the following			3 X 10 = 30
Q. No.	Question	Marks	CO
a to e		10	
Section C			
Question no. 3,4,5,6,7. Attempt any one of the following			1 X 10 = 10
a		10	
b		10	

Prerequisites

- Basic knowledge of 8085 microprocessor
- Knowledge of its architecture and operations

Introduction to microprocessor

- 8085 is an **8-bit microprocessor** as it operates on 8 bits at a time and is created with **N-MOS technology**.
  This microprocessor exhibits some unique characteristics and this is the reason it still holds popularity among the microprocessors.
- Basically, 8085 was the **first commercially successful microprocessor by Intel**.

Unit Content

- Course Objective
- Unit Objective
- Course Outcome
- CO and PO Mapping
- Topic Objective
- Prerequisite
- Instruction sets
- Instructions classification
- Data transfer operations
- Arithmetic operations
- Logical operations
- Branching operations
- Machine control and assembler directives
- Writing assembly language programs
- Programming techniques
- Daily quiz & MCQs
- Old Question Papers
- Recap
- Video Links
- Weekly Assignments
- References

Unit Objective

1. To understand the instructions sets of 8085 microprocessor
2. To learn writing assembly language programs
3. To discuss concept of classification of instructions
4. To acquire the knowledge of programming techniques

Instruction Set of 8085

Name of the Topic	Objective of the topic	Mapping with CO
Instruction Set of 8085	To understand the instruction of 8085 microprocessor.	CO2

Prerequisite

- Basic concept of Digital Logic.
- Understanding of memory unit & data size.

Instruction Set of 8085

- An instruction is a binary pattern designed inside a microprocessor to perform a specific function.
- The entire group of instructions that a microprocessor supports is called Instruction Set.
- 8085 has 246 instructions.
- Each instruction is represented by an 8-bit binary value.
- These 8-bits of binary value is called Op-Code or Instruction Byte.

Instruction Set of 8085

Classification of Instruction Set

- Data Transfer Instruction
- Arithmetic Instructions
- Logical Instructions
- Branching Instructions
- Control Instructions

Video Link

https://www.youtube.com/watch?v=ekkoIeonyzg

Daily Quiz

- What is true about microprocessor?

  A. Microprocessor is a controlling unit of a micro-computer

  B. It is fabricated on a small chip capable of performing ALU (Arithmetic Logical Unit) operations

  C. It also communicate with the other devices connected to it.

  D. All of the above

- Microprocessor consists of?

  A. ALU

  B. register array

  C. control unit

  D. All of the above

Daily Quiz

- Which control instruction is followed by an un-conditional branch instructions so as to branch to a single location from the double ones with respect to specified status-bit condition?

  A. jump instruction

  B. branch instruction

  C. skip instruction

  D. return- from-subroutine instructions

- Which category of microprocessor instructions detect the status conditions in registers and accordingly exhibit the variations in program sequence on the basis of detected results?

  A. Transfer Instructions

  B. Operation Instructions

  C. Control Instructions

  D. All of the above

Recap

- Detail of various types of instruction set is discussed.

Data Transfer Operation

Name of the Topic	Objective of the topic	Mapping with CO
Data Transfer Operation	To learn different types of data transfer instructions.	CO2

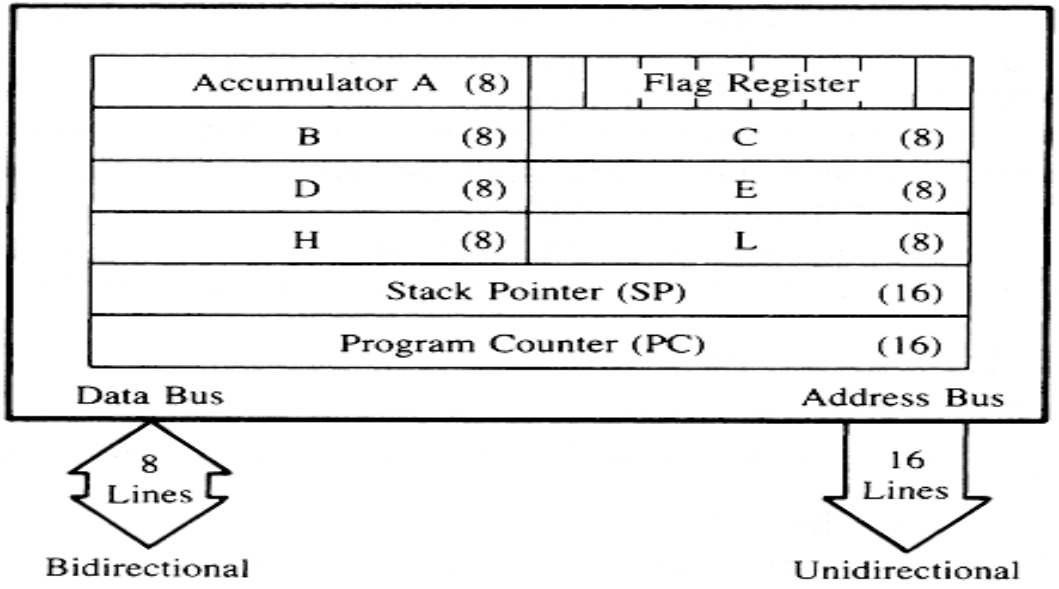
Prerequisite

- Basic concept of Digital Logic.
- Overview of register set of 8085 microprocessor.

Data Transfer Operation

Data Transfer Instructions

- These instructions move data between registers, or between memory and registers.
- These instructions copy data from source to destination.
- While copying, the contents of source are not modified.



Data Transfer Operation

Data Transfer Instructions

- This instruction perform following six operation.
  - Load an 8 bit number in a register.
  - Load 16 bit number in a register pair.
  - Copy from register to register.
  - Copy between register & memory.
  - Copy between I/O & accumulator.
  - Copy between registers & stack memory.

Data Transfer Operation

Data Transfer Instructions

Opcode	Operand	Description
MOV	Rd, Rs M, Rs Rd, M	Copy from source to destination.

- This instruction copies the contents of the source register into the destination register.
- The contents of the source register are not altered.
- If one of the operands is a memory location, its location is specified by the contents of the HL registers.
- Example: MOV B, C or MOV B, M.

Data Transfer Operation

Data Transfer Instructions

Opcode	Operand	Description
MVI	Rd, Data M, Data	Move immediate 8-bit

- The 8-bit data is stored in the destination register or memory.
- If the operand is a memory location, its location is specified by the contents of the H-L registers.
- **Example:** MVI B, 57H or MVI M, 57H

Data Transfer Operation

Data Transfer Instructions

Opcode	Operand	Description
LDA	16-bit address	Load Accumulator

- The contents of a memory location, specified by a 16-bit address in the operand, are copied to the accumulator.
- The contents of the source are not altered.
- **Example:** LDA 2034H

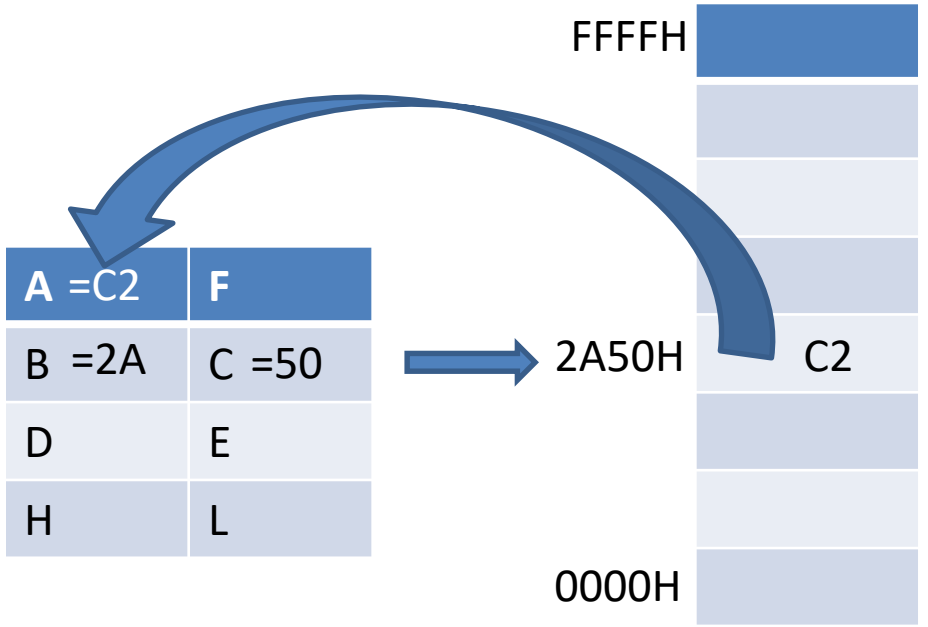
Data Transfer Operation

Data Transfer Instructions

Opcode	Operand	Description
LDAX	B/D Register Pair	Load accumulator indirect

- The contents of the designated register pair point to a memory location. This instruction copies the contents of that memory location into the accumulator. The contents of either the register pair or the memory location are not altered.

- **Example:** LDAX B



Data Transfer Operation

Data Transfer Instructions

Opcode	Operand	Description
LXI	Reg. pair, 16-bit data	Load register pair immediate

- This instruction loads 16-bit data in the register pair.
- **Example:** LXI H, 2034 H

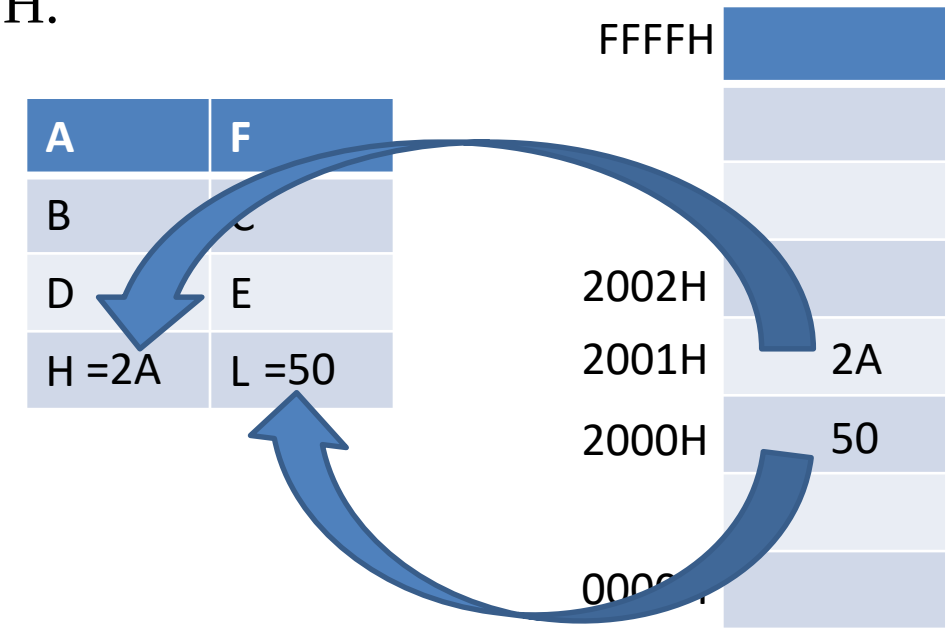
Data Transfer Operation

Data Transfer Instructions

Opcode	Operand	Description
LHLD	16-bit address	Load H-L registers direct

- This instruction copies the contents of memory location pointed out by 16-bit address into register L. It copies the contents of next memory location into register H.

- **Example:** LHLD 2000 H

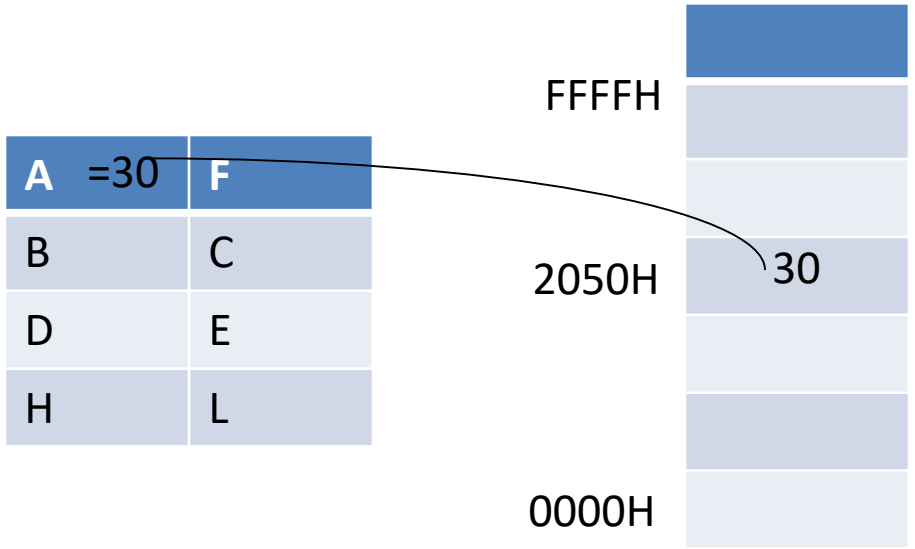


Data Transfer Operation

Data Transfer Instructions

Opcode	Operand	Description
STA	16-bit address	Store accumulator direct

- The contents of accumulator are copied into the memory location specified by the operand.
- **Example:** STA 2050 H

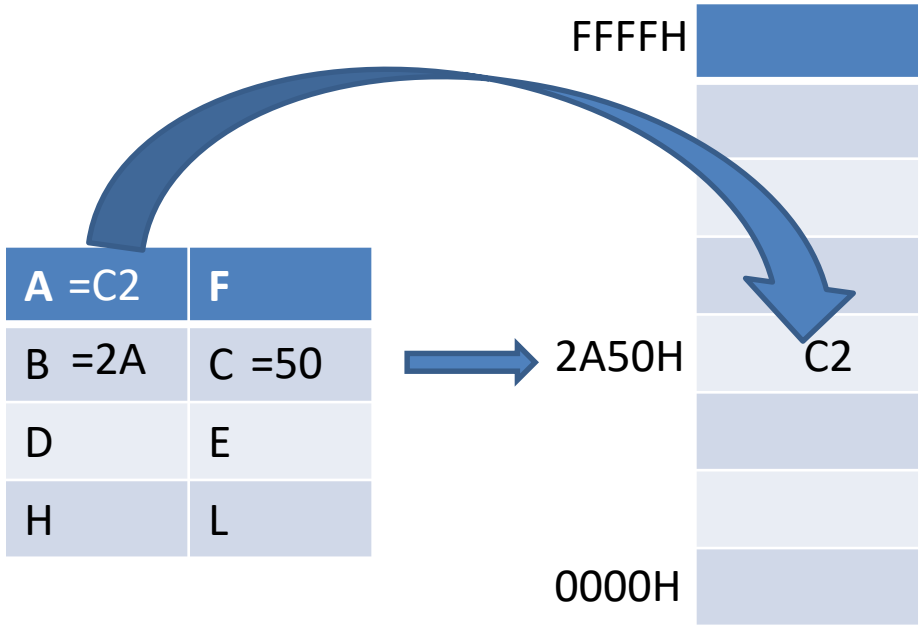


Data Transfer Operation

Data Transfer Instructions

Opcode	Operand	Description
STAX	Reg. pair	Store accumulator indirect

- The contents of accumulator are copied into the memory location specified by the contents of the register pair.
- **Example:** STAX B

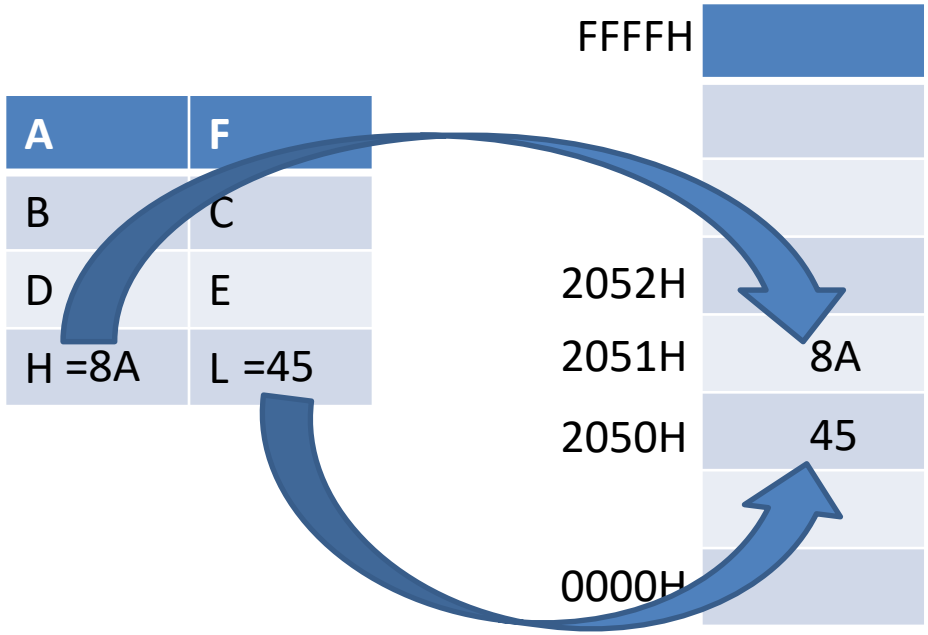


Data Transfer Operation

Data Transfer Instructions

Opcode	Operand	Description
SHLD	16-bit address	Store H-L registers direct

- The contents of register L are stored into memory location specified by the 16-bit address.
- The contents of register H are stored into the next memory location.
- **Example:** SHLD 2050 H

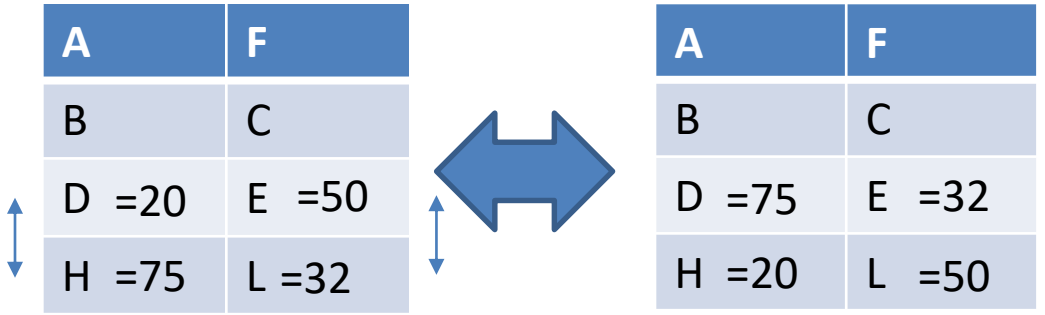


Data Transfer Operation

Data Transfer Instructions

Opcode	Operand	Description
XCHG	None	Exchange H-L with D-E

- The contents of register H are exchanged with the contents of register D.
- The contents of register L are exchanged with the contents of register E.
- **Example:** XCHG



Data Transfer Operation

Data Transfer Instructions

Opcode	Operand	Description
SPHL	None	Copy H-L pair to the Stack Pointer (SP)

- This instruction loads the contents of H-L pair into SP.
- **Example:** SPHL

Data Transfer Operation

Data Transfer Instructions

Opcode	Operand	Description
XTHL	None	Exchange H–L with top of stack

- The contents of L register are exchanged with the location pointed out by the contents of the SP.
- The contents of H register are exchanged with the next location (SP + 1).
- **Example:** XTHL

Data Transfer Operation

Data Transfer Instructions

Opcode	Operand	Description
PCHL	None	Load program counter with H-L contents

- The contents of registers H and L are copied into the program counter (PC).
- The contents of H are placed as the high-order byte and the contents of L as the low-order byte.
- **Example:** PCHL

Data Transfer Operation

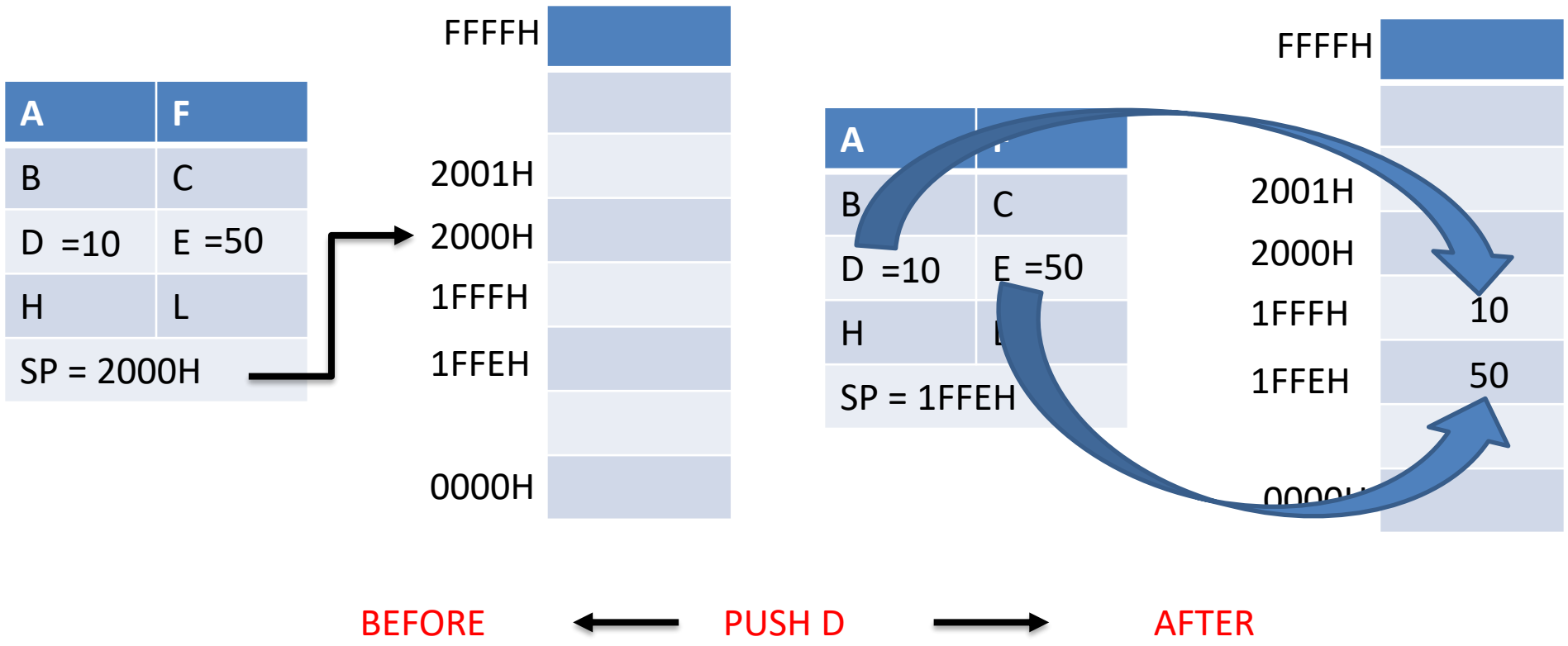
Data Transfer Instructions

Opcode	Operand	Description
PUSH	Reg. pair	Push register pair onto stack

- The contents of register pair are copied onto stack.
- SP is decremented and the contents of high-order registers (B, D, H, A) are copied into stack.
- SP is again decremented and the contents of low-order registers (C, E, L, Flags) are copied into stack.
- **Example:** PUSH D

Data Transfer Operation

Example : PUSH D



Data Transfer Operation

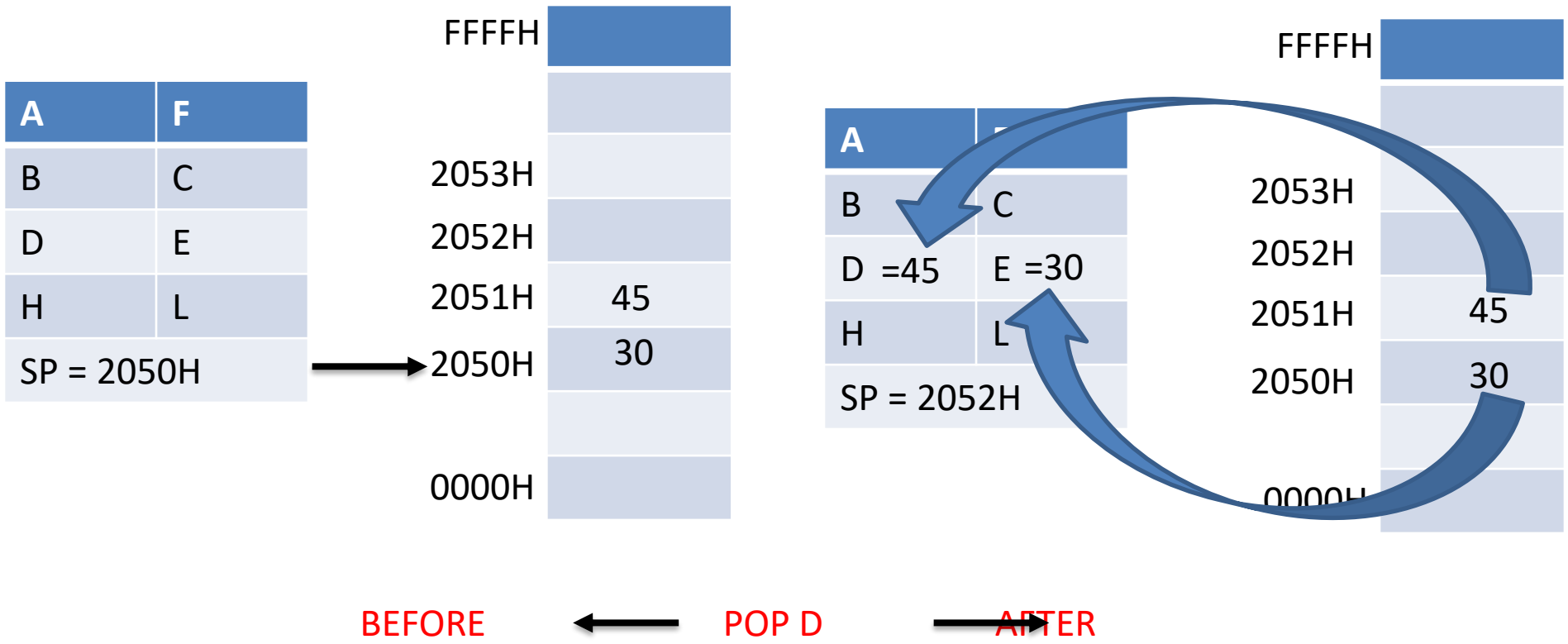
Data Transfer Instructions

Opcode	Operand	Description
POP	Reg. pair	Pop stack to register pair

- The contents of top of stack are copied into register pair.
- The contents of location pointed out by SP are copied to the low-order register (C, E, L, Flags).
- SP is incremented and the contents of location are copied to the high-order register (B, D, H, A).
- **Example:** POP H

Data Transfer Operation

Example : POP D



Data Transfer Operation

Data Transfer Instructions

Opcode	Operand	Description
OUT	8-bit port address	Copy data from accumulator to a port with 8-bit address

- The contents of accumulator are copied into the I/O port.
- **Example:** OUT 78 H

Data Transfer Operation

Data Transfer Instructions

Opcode	Operand	Description
IN	8-bit port address	Copy data to accumulator from a port with 8-bit address

- The contents of I/O port are copied into accumulator.
- **Example:** IN 8C H

Video Link

https://www.youtube.com/watch?v=ekkoIeonyzg

Daily Quiz

- DAD is for Double ADD. DAD B instruction in 8085 is used for ?

  a**) Content of BC is added HL Result is in HL**.

  b) Content of BC is added to Accumulator

  c) Content of BC is added to DE

- DAA instruction is used for ____.

  a) Double Add Accumulator.

  b)Decimal Adjust Accumulator.

  c)Decrement Accumulator

- To save accumulator value and the flag register on to the stack, which of the following instructions is used?

  a) **PUSH PSW**
  b) PUSH A
  c) PUSH SP
  d) POP PSW

Daily Quiz

- What is the content of A at the end of this program?

```
MVI A, 06H
RLC
MOV B, A
RLC
RLC
ADD B
```

a) 3Ch b)18h c)0C h d)00 h

- In 8085, HLT opcode means:

a) Remain idle for 10 seconds.\

b) Remain idle for 0.1 seconds

c) **End of Program**

Recap

- Data transfer instructions of 8085 microprocessor is discussed.

Arithmetic operations

Name of the Topic	Objective of the topic	Mapping with CO
Arithmetic operations	To understand the arithmetic instructions of 8085 microprocessor.	CO2

Prerequisite

- Basic concept of Digital Logic.
- Knowledge of Boolean Algebra.
- Register set of 8085 microprocessor.

Arithmetic operations

Arithmetic Instructions

Addition

- Any 8-bit number, or the contents of register, or the contents of memory location can be added to the contents of accumulator.
- The result (sum) is stored in the accumulator.
- No two other 8-bit registers can be added directly.

Note: The contents of register B cannot be added directly to the contents of register C.

Arithmetic operations

Arithmetic Instructions

Subtraction

- Any 8-bit number, or the contents of register, or the contents of memory location can be subtracted from the contents of accumulator.
- The result is stored in the accumulator.
- Subtraction is performed in 2's complement form.
- If the result is negative, it is stored in 2's complement form.
- No two other 8-bit registers can be subtracted directly.

Arithmetic operations

Arithmetic Instructions

Increment / Decrement

- The 8-bit contents of a register or a memory location can be incremented or decremented by 1.
- The 16-bit contents of a register pair can be incremented or decremented by 1.
- Increment or decrement can be performed on any register or a memory location.

Arithmetic operations

Arithmetic Instructions

Opcode	Operand	Description
ADD	R M	Add register or memory to accumulator

- The contents of register or memory are added to the contents of accumulator. The result is stored in accumulator.
- If the operand is memory location, its address is specified by H-L pair.
- All flags are modified to reflect the result of the addition.
- **Example:** ADD B (A+B $\longrightarrow$ A)

  ADD M (A+ M(HL) $\longrightarrow$ A)

Arithmetic operations

Arithmetic Instructions

Opcode	Operand	Description
ADC	R M	Add register or memory to accumulator with carry

- The contents of register or memory and Carry Flag (CY) are added to the contents of accumulator. The result is stored in accumulator.
- If the operand is memory location, its address is specified by H-L pair.
- All flags are modified to reflect the result of the addition.

Example: ADC B (A+B + CY $\longrightarrow$ A)

ADC M (A+M(HL) + CY $\longrightarrow$ A)

Arithmetic operations

Arithmetic Instructions

Opcode	Operand	Description
ADI	8-bit data	Add immediate to accumulator

- The 8-bit data is added to the contents of accumulator.
- The result is stored in accumulator.
- All flags are modified to reflect the result of the addition.
- **Example:** ADI 45 H (A + 45 → A)

Arithmetic operations

Arithmetic Instructions

Opcode	Operand	Description
ACI	8-bit data	Add immediate to accumulator with carry

- The 8-bit data and the Carry Flag (CY) are added to the contents of accumulator.
- The result is stored in accumulator.
- All flags are modified to reflect the result of the addition.
- **Example:** ACI 45 H (A + 45 + CY $\longrightarrow$ A)

Arithmetic operations

Arithmetic Instructions

Opcode	Operand	Description
DAD	Reg. pair	Add register pair to H-L pair

- The 16-bit contents of the register pair are added to the contents of H-L pair.
- The result is stored in H-L pair.
- If the result is larger than 16 bits, then CY is set.
- No other flags are changed.
- **Example:** DAD B (HL + BC ⟶ HL)

Arithmetic operations

Arithmetic Instructions

PROBLEM

DAD D

Let D=30H, E=20H
H= 1AH, L = 42 H

SOLUTION

H= 4AH & L = 62H

Arithmetic operations

Arithmetic Instructions

Opcode	Operand	Description
SUB	R M	Subtract register or memory from accumulator

- The contents of the register or memory location are subtracted from the contents of the accumulator. The result is stored in accumulator.
- If the operand is memory location, its address is specified by H-L pair.
- All flags are modified to reflect the result of subtraction.

Example: SUB B (A – B ⟶ A)

SUB M (A – M (HL) ⟶ A)

Arithmetic operations

Arithmetic Instructions

Opcode	Operand	Description
SBB	R M	Subtract register or memory from accumulator with borrow

- The contents of the register or memory location and Borrow Flag (i.e. CY) are subtracted from the contents of the accumulator. The result is stored in accumulator.
- If the operand is memory location, its address is specified by H-L pair.
- All flags are modified to reflect the result of subtraction.

Example: SBB B (A – B – CY ⟶ A)

SBB M (A – M (HL) – CY ⟶ A)

Arithmetic operations

Arithmetic Instructions

Opcode	Operand	Description
SUI	8-bit data	Subtract immediate from accumulator

- The 8-bit data is subtracted from the contents of the accumulator.
- The result is stored in accumulator.
- All flags are modified to reflect the result of subtraction.
- **Example:** SUI 54 H (A – 54 ⟶ A)

Arithmetic operations

Arithmetic Instructions

Opcode	Operand	Description
SBI	8-bit data	Subtract immediate from accumulator with borrow

- The 8-bit data and the Borrow Flag (i.e. CY) is subtracted from the contents of the accumulator.
- The result is stored in accumulator.
- All flags are modified to reflect the result of subtraction.
- **Example:** SBI 45 H (A – 45 – CY $\longrightarrow$ A)

Arithmetic operations

Arithmetic Instructions

Opcode	Operand	Description
INR	R M	Increment register or memory by 1

- The contents of register or memory location are incremented by 1. The result is stored in the same place.
- If the operand is a memory location, its address is specified by the contents of H-L pair.

Example: INR B (B + 1 ⟶ B)

INR M (M + 1 ⟶ M)

Arithmetic operations

Arithmetic Instructions

Opcode	Operand	Description
INX	R	Increment register pair by 1

- The contents of register pair are incremented by 1.
- The result is stored in the same place.
- **Example:** INX H (HL+1 ⟶ HL)

Arithmetic operations

Arithmetic Instructions

Opcode	Operand	Description
DCR	R M	Decrement register or memory by 1

- The contents of register or memory location are decremented by 1. The result is stored in the same place.
- If the operand is a memory location, its address is specified by the contents of H-L pair.

Example: DCR B (B – 1 ⟶ B)

DCR M (M – 1 ⟶ M)

Arithmetic operations

Arithmetic Instructions

Opcode	Operand	Description
DCX	R	Decrement register pair by 1

- The contents of register pair are decremented by 1.
- The result is stored in the same place.
- **Example:** DCX H (HL – 1 ⟶ HL)

Arithmetic operations

Arithmetic Instructions

Opcode	Operand	Description
DAA		Decimal Adjust Accumulator (BCD format)

- It converts hexadecimal to BCD no.
- This instruction is used after addition instruction.

Operation:

- If lower nibble is >9 or AC = 1 then add 06.
- If higher nibble is >9 or CY = 1 then add 06.

Arithmetic operations

Arithmetic Instructions

DAA Instruction Example

1) A=34H & Data = 48H

2) A=37H & Data = 15H

3) A=85H & Data = 68H

Video Link

https://www.youtube.com/watch?v=ekkoIeonyzg

Daily Quiz

- What is the content of PC Register at the end of the following program?

  LXI H 8A79h
  MOV A L
  ADD H
  DAA
  MOV H A
  PCHL

  a) 2304 b) **6979** c) 7100 d) 8255

- What does microprocessor speed depends on?

  a) Clock
  b) Data bus width
  c) Address bus width
  d) All of these

Daily Quiz

- What is the status of z flag, cy flag, sign flag at the end of this program?

```
MVI A, 02H
MVI B, 03H
ADD B
XRA A
```

a) **1,0,0** b)0,1,0 c)1,0,0 d)1,0,1

- What is the content of Register A at the end?

```
XRA A
MVI B, 4DH
SUI 4FH
ANA B
 HLT
```

a) 00h b) **01h** c) 4Dh

Weekly Assignment

1. Specify the contents of the A,B,C,D,E,H,L ,M registers as each if the following instructions is being executed.

```
MVI C,FFH
LXI H,2030H
LXI D,7050H
MOV M,C
XCHG
LDAX D
HLT
```

2. Specify the number of times the following loops are executed

```
LXI B,1000H
LOOP: DCX B
NOP
JNZ LOOP
```

Weekly Assignment

3. Specify the number of times the following loops are executed

```
MVI A,17H
LOOP: RAL
ORA A
JNC LOOP
```

4. Explain the meaning of the following program

```
LXI H,0000H
DAD SP
MOV A,H
OUT PORT1
MOV A,L
OUT PORT2
```

Recap

- Arithmetic instructions of 8085 microprocessor with example is discussed.

Logic Operations

Name of the Topic	Objective of the topic	Mapping with CO
Logic Operations	To understand logic instructions of 8085 microprocessor.	CO2

Prerequisite

- Basic concept of Digital Logic
- Knowledge of Boolean Algebra
- Register set of 8085 microprocessor.

Logic Operations

Logical Instructions

- These instructions perform logical operations on data stored in registers, memory and status flags.
- The logical operations are:
  - AND
  - OR
  - XOR
  - Rotate
  - Compare
  - Complement

Logic Operations

AND, OR, XOR

- Any 8-bit data, or the contents of register, or memory location can logically have
  - AND operation
  - OR operation
  - XOR operation

  with the contents of accumulator.
- The result is stored in accumulator.

Logic Operations

Rotate

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

Logic Operations

Compare

- Any 8-bit data, or the contents of register, or memory location can be compares for:
  - Equality
  - Greater Than
  - Less Than

  with the contents of accumulator.
- The result is reflected in status flags.

Logic Operations

Complement

- The contents of accumulator can be complemented.
- Each 0 is replaced by 1 and each 1 is replaced by 0.

Logic Operations

Logical Instructions

Opcode	Operand	Description
ANA	R M	Logical AND register or memory with accumulator

- The contents of the accumulator are logically ANDed with the contents of register or memory.
- The result is placed in the accumulator.
- If the operand is a memory location, its address is specified by the contents of H-L pair.
- S, Z, P are modified to reflect the result of the operation.
- **Example:** ANA B or ANA M.

Logic Operations

Logical Instructions

Opcode	Operand	Description
ANI	8-bit data	Logical AND immediate with accumulator

- The contents of the accumulator are logically ANDed with the 8-bit data.
- The result is placed in the accumulator.
- S, Z, P are modified to reflect the result.
- CY is reset, AC is set.
- **Example:** ANI 86H.

Logic Operations

Logical Instructions

Opcode	Operand	Description
ORA	R M	Logical OR register or memory with accumulator

- The contents of the accumulator are logically ORed with the contents of the register or memory. The result is placed in the accumulator.
- If the operand is a memory location, its address is specified by the contents of H-L pair.
- S, Z, P are modified to reflect the result. CY and AC are reset.
- **Example:** ORA E or ORA M.

Logic Operations

Logical Instructions

Opcode	Operand	Description
ORI	8-bit data	Logical OR immediate with accumulator

- The contents of the accumulator are logically ORed with the 8-bit data.
- The result is placed in the accumulator.
- S, Z, P are modified to reflect the result.
- CY and AC are reset.
- **Example:** ORI 20H.

Logic Operations

Logical Instructions

Opcode	Operand	Description
XRA	R M	Logical XOR register or memory with accumulator

- The contents of the accumulator are XORed with the contents of the register or memory.
- The result is placed in the accumulator.
- If the operand is a memory location, its address is specified by the contents of H-L pair.
- S, Z, P are modified to reflect the result of the operation.
- CY and AC are reset.
- **Example:** XRA D or XRA M.

Logic Operations

Logical Instructions

Opcode	Operand	Description
XRI	8-bit data	XOR immediate with accumulator

- The contents of the accumulator are XORed with the 8-bit data.
- The result is placed in the accumulator.
- S, Z, P are modified to reflect the result.
- CY and AC are reset.
- **Example:** XRI 23H.

Logic Operations

Instruction	Sign (S)	Zero (Z)	Auxillary Carry (AC)	Parity (P)	Carry (CY)
ANA R	√	√	1	√	0
ANI 8 bit data	√	√	1	√	0
ORA R	√	√	0	√	0
ORI 8 bit data	√	√	0	√	0
XRA R	√	√	0	√	0
XRI 8 bit data	√	√	0	√	0

√ - Will change according to result

Logic Operations

Logical Instructions

Opcode	Operand	Description
CMP	R M	Compare register or memory with accumulator

- The contents of the operand (register or memory) are compared with the contents of the accumulator.
- Both contents are preserved .
- The result of the comparison is shown by setting the flags of the PSW as follows:

Logic Operations

Logical Instructions

- if (A) < (reg/mem): carry flag is set
- if (A) = (reg/mem): zero flag is set
- if (A) > (reg/mem): carry and zero flags are reset.

Example: CMP C or CMP D or CMP M

Logic Operations

Logical Instructions

Opcode	Operand	Description
CPI	8-bit data	Compare immediate with accumulator

- The 8-bit data is compared with the contents of accumulator.
- The values being compared remain unchanged.
- The result of the comparison is shown by setting the flags of the PSW as follows:

Logic Operations

Logical Instructions

- if (A) < data: carry flag is set
- if (A) = data: zero flag is set
- if (A) > data: carry and zero flags are reset

Example: CPI A6H

Logic Operations

Logical Instructions

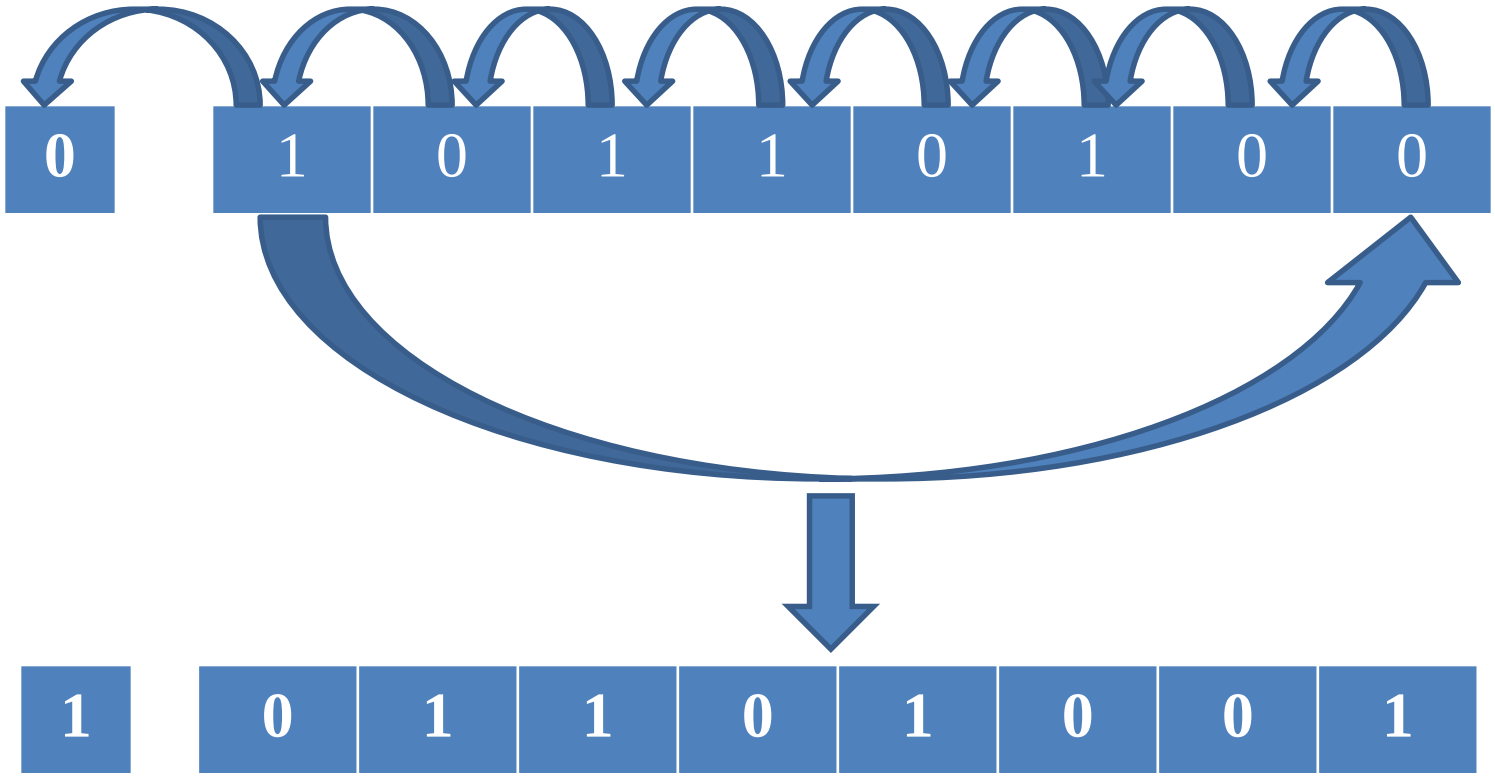
Opcode	Operand	Description
RLC	None	Rotate accumulator left

- Each binary bit of the accumulator is rotated left by one position.
- Bit D7 is placed in the position of D0 as well as in the Carry flag.
- CY is modified according to bit D7.
- S, Z, P, AC are not affected.

Example: RLC

Logic Operations

Logical Instructions



Logic Operations

Logical Instructions

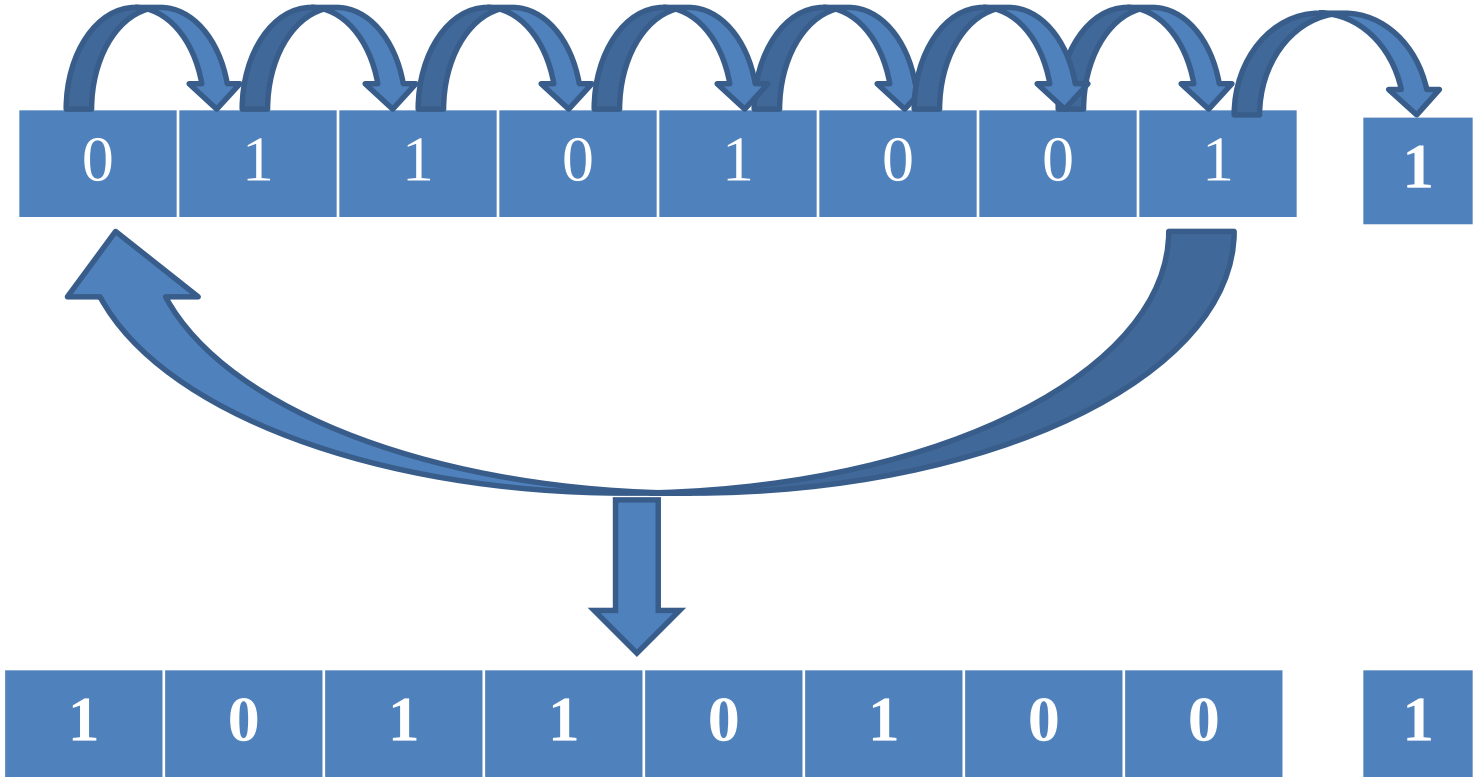
Opcode	Operand	Description
RRC	None	Rotate accumulator right

- Each binary bit of the accumulator is rotated right by one position.
- Bit D0 is placed in the position of D7 as well as in the Carry flag.
- CY is modified according to bit D0.
- S, Z, P, AC are not affected.

Example: RRC

Logic Operations

Logical Instructions



Logic Operations

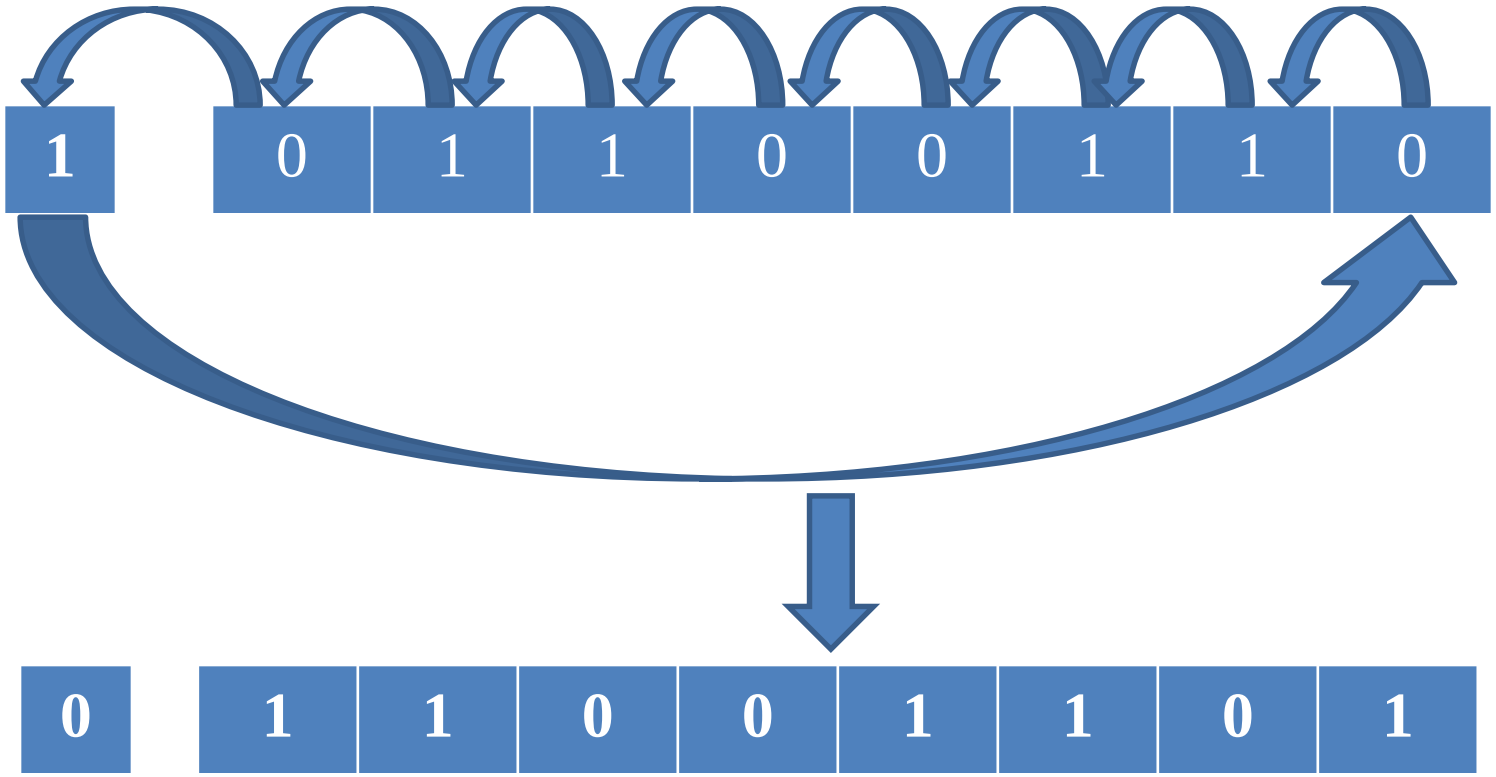
Logical Instructions

Opcode	Operand	Description
RAL	None	Rotate accumulator left through carry

- Each binary bit of the accumulator is rotated left by one position through the Carry flag.
- Bit D7 is placed in the Carry flag, and the Carry flag is placed in the least significant position D0.
- CY is modified according to bit D7.
- S, Z, P, AC are not affected.
- **Example:** RAL.

Logic Operations

Logical Instructions



Logic Operations

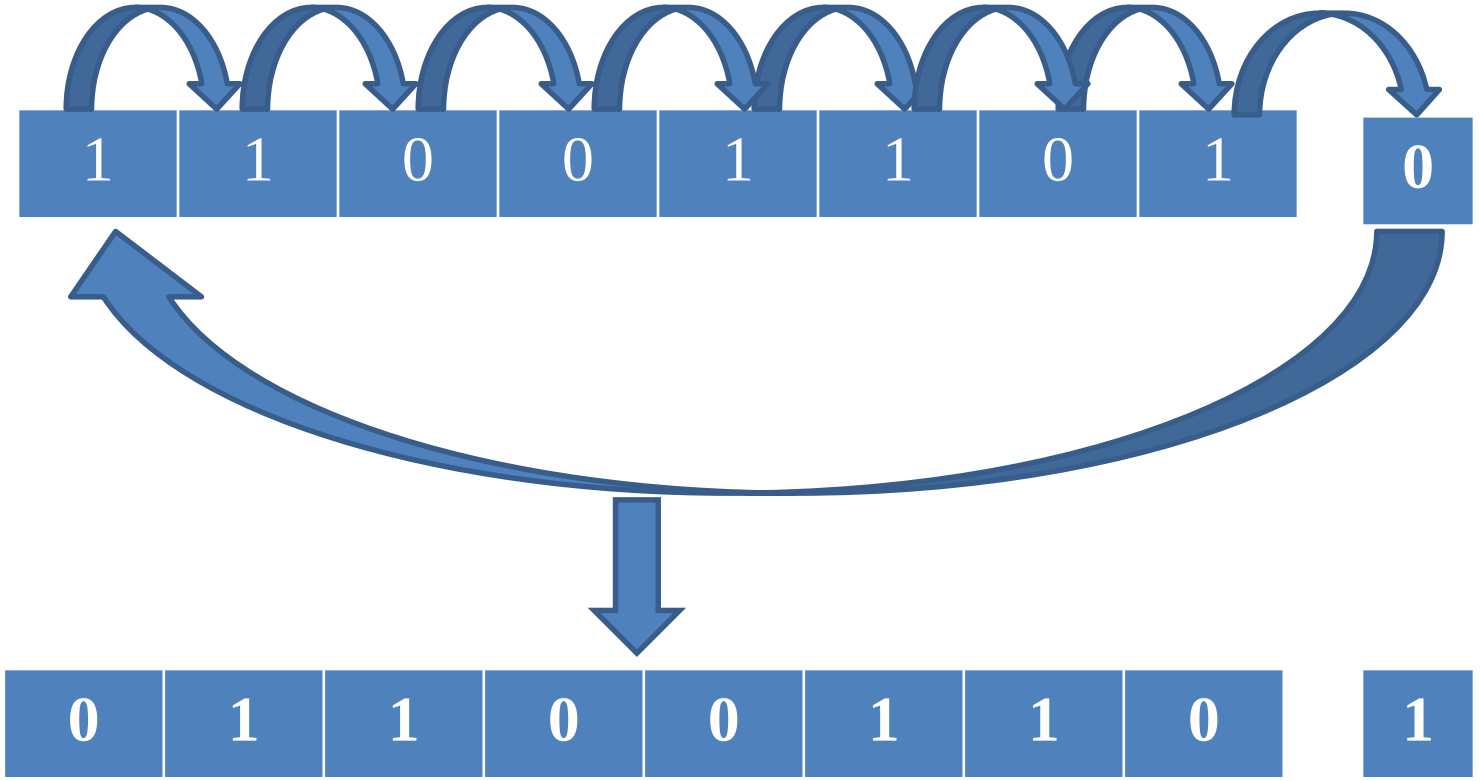
Logical Instructions

Opcode	Operand	Description
RAR	None	Rotate accumulator right through carry

- Each binary bit of the accumulator is rotated right by one position through the Carry flag.
- Bit D0 is placed in the Carry flag, and the Carry flag is placed in the most significant position D7.
- CY is modified according to bit D0.
- S, Z, P, AC are not affected.
- **Example:** RAR.

Logic Operations

Logical Instructions



Logic Operations

Logical Instructions

Opcode	Operand	Description
CMA	None	Complement accumulator

- The contents of the accumulator are complemented.
- No flags are affected.
- **Example:** CMA

  A= ABH

Logic Operations

Logical Instructions

Opcode	Operand	Description
CMC	None	Complement carry

- The Carry flag is complemented.
- No other flags are affected.

Example: CMC

Logic Operations

Logical Instructions

Opcode	Operand	Description
STC	None	Set carry

- The Carry flag is set to 1.
- No other flags are affected.
- **Example:** STC.

Video Link

https://www.youtube.com/watch?v=ekkoIeonyzg

Daily Quiz

- The instruction that represents the ‘rotate source, count’ is

  a) RCL

  b) RCR

  c) ROR

  d) **All of the mentioned**

- What is the function of STC instruction?

  a) Store to C Register, the value of Accumulator

  b) Set Carry to 1

  c) Clear the Stack pointer.

- After "XRA A" instruction is executed, what will be the status of Zero Flag?

  a)1 b) 0 c) No change

Daily Quiz

- What is the content of A-Register at the end of this program?

  XRA A

  MVI B, F0H

  SUB B

  a) 01H b) 0FH c)F0H **d)10H**

- What is the content of A at the end of this program?

  STC

  MVI A, 35H

  ACI 26H

  a) 2Dh **b) 5Ch** c) 23h d) 5B h

Recap

- Logical instructions of 8085 microprocessor with example is discussed.

Branch operation

Name of the Topic	Objective of the topic	Mapping with CO
Branch operation	To understand the branch & control instructions of 8085 microprocessor.	CO2

Prerequisite

- Basic concept of Digital Logic
- Knowledge of Boolean Algebra
- Understanding of 8085 microprocessor architecture.

Branch operation

Branching Instructions

- The branching instruction alter the normal sequential flow.
- These instructions alter either unconditionally or conditionally.

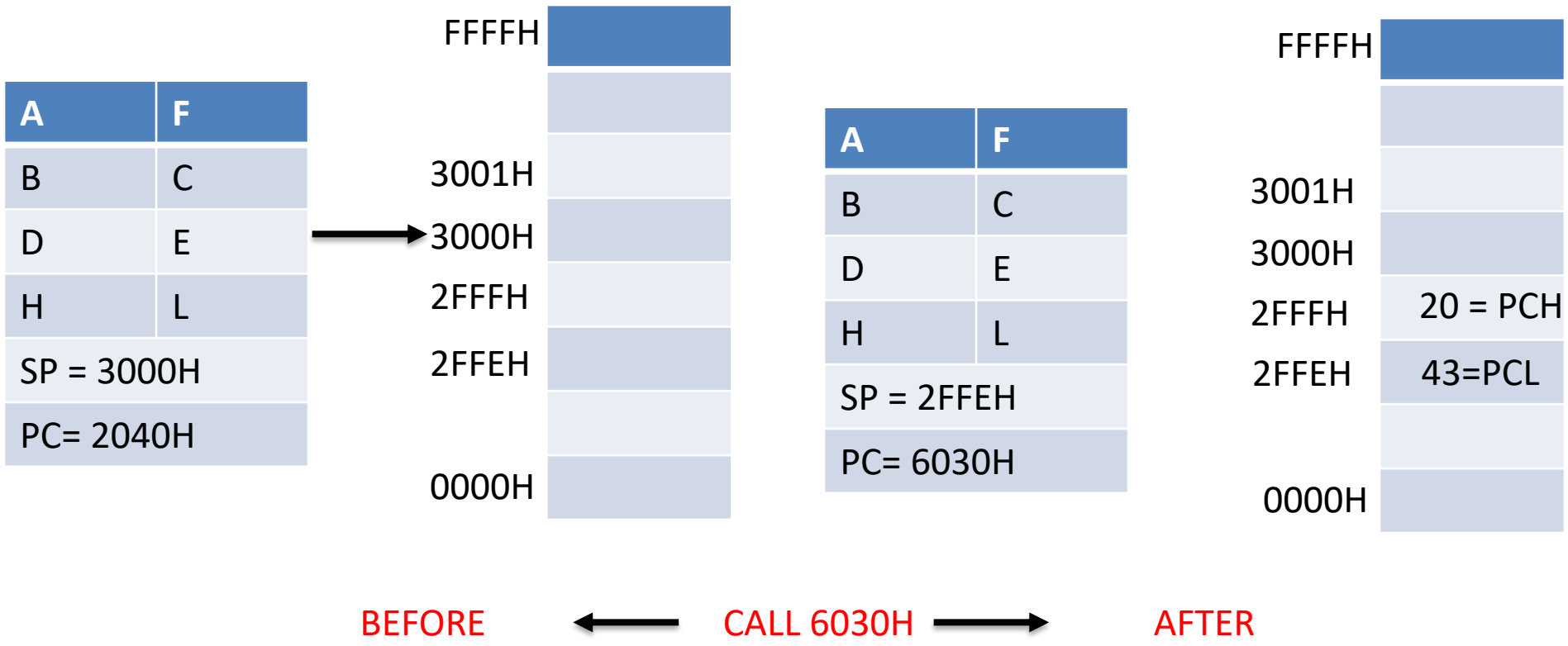
Branch operation

Opcode	Operand	Description
CALL	16-bit address	Call unconditionally

- The program sequence is transferred to the memory location specified by the 16-bit address given in the operand.
- Before the transfer, the address of the next instruction after CALL (the contents of the program counter) is pushed onto the stack.
- **Example:** CALL 2034 H.

Branch operation

Example : CALL 6030H



Branch operation

Opcode	Operand	Description
Cx	16-bit address	Call conditionally

- The program sequence is transferred to the memory location specified by the 16-bit address given in the operand based on the specified flag of the PSW.
- Before the transfer, the address of the next instruction after the call (the contents of the program counter) is pushed onto the stack.
- **Example:** CZ 2034 H.

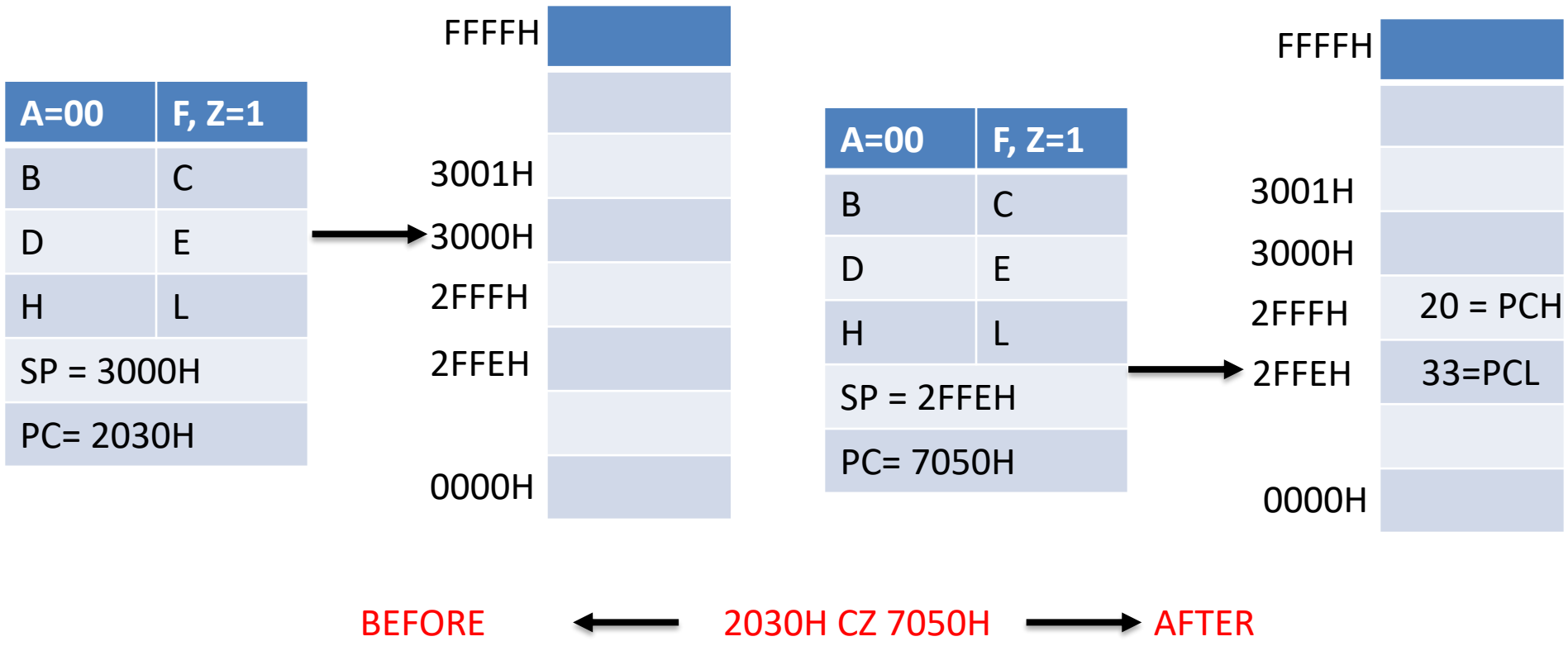
Branch operation

Call Conditionally

Opcode	Description	Status Flags
CC	Call on Carry	CY = 1
CNC	Call on Not Carry	CY = 0
CP	Call on Positive	S = 0
CM	Call on Minus	S = 1
CZ	Call on Zero	Z = 1
CNZ	Call on Not Zero	Z = 0
CPE	Call on Parity Even	P = 1
CPO	Call on Parity Odd	P = 0

Branch operation

Example : 2030H CZ 7050H



Branch operation

Opcode	Operand	Description
RET	None	Return unconditionally

- The program sequence is transferred from the subroutine to the calling program.
- The two bytes from the top of the stack are copied into the program counter, and program execution begins at the new address.
- **Example:** RET.

Branch operation

Opcode	Operand	Description
Rx	None	Call conditionally

- The program sequence is transferred from the subroutine to the calling program based on the specified flag of the PSW.
- The two bytes from the top of the stack are copied into the program counter, and program execution begins at the new address.
- **Example:** RZ.

Branch operation

Return Conditionally

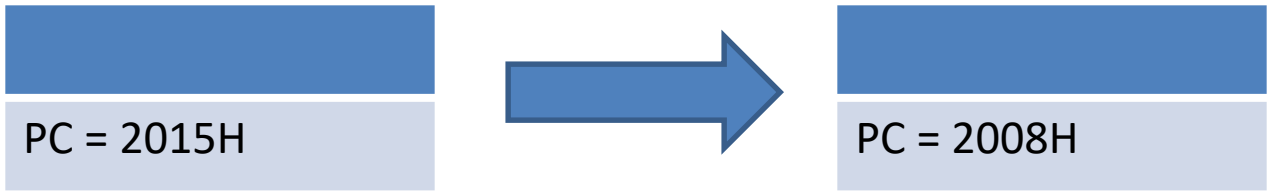
Opcode	Description	Status Flags
RC	Return if Carry	CY = 1
RNC	Return if Not Carry	CY = 0
RP	Return if Positive	S = 0
RM	Return if Minus	S = 1
RZ	Return if Zero	Z = 1
RNZ	Return if Not Zero	Z = 0
RPE	Return if Parity Even	P = 1
RPO	Return if Parity Odd	P = 0

Branch operation

Branching Instructions

Opcode	Operand	Description
JMP	16-bit address	Jump unconditionally

- The program sequence is transferred to the memory location specified by the 16-bit address given in the operand.
- **Example:** 2015H JMP 2008 H.



Before After

Branch operation

Branching Instructions

Opcode	Operand	Description
Jx	16-bit address	Jump conditionally

- The program sequence is transferred to the memory location specified by the 16-bit address given in the operand based on the specified flag of the PSW.
- **Example:** JZ 2034 H.

Branch operation

Jump Conditionally

Opcode	Description	Status Flags
JC	Jump if Carry	CY = 1
JNC	Jump if No Carry	CY = 0
JP	Jump if Positive	S = 0
JM	Jump if Minus	S = 1
JZ	Jump if Zero	Z = 1
JNZ	Jump if No Zero	Z = 0
JPE	Jump if Parity Even	P = 1
JPO	Jump if Parity Odd	P = 0

Branch operation

Opcode	Operand	Description
RST	0 – 7	Restart (Software Interrupts)

- The RST instruction jumps the control to one of eight memory locations depending upon the number.
- These are used as software instructions in a program to transfer program execution to one of the eight locations.
- **Example:** RST 3.

Branch operation

Restart Address Table

Instructions	Restart Address
RST 0	0000 H
RST 1	0008 H
RST 2	0010 H
RST 3	0018 H
RST 4	0020 H
RST 5	0028 H
RST 6	0030 H
RST 7	0038 H

Control Operation

Control Instructions

- The control instructions control the operation of microprocessor.

Opcode	Operand	Description
NOP	None	No operation

- No operation is performed.
- The instruction is fetched and decoded but no operation is executed.
- **Example:** NOP

Control Operation

Opcode	Operand	Description
HLT	None	Halt

- The CPU finishes executing the current instruction and halts any further execution.
- An interrupt or reset is necessary to exit from the halt state.
- **Example:** HLT

Control Operation

Opcode	Operand	Description
DI	None	Disable interrupt

- The interrupt enable flip-flop is reset and all the interrupts except the TRAP are disabled.
- No flags are affected.
- **Example:** DI

Control Operation

Opcode	Operand	Description
EI	None	Enable interrupt

- The interrupt enable flip-flop is set and all interrupts are enabled.
- No flags are affected.
- This instruction is necessary to re-enable the interrupts (except TRAP).
- **Example:** EI

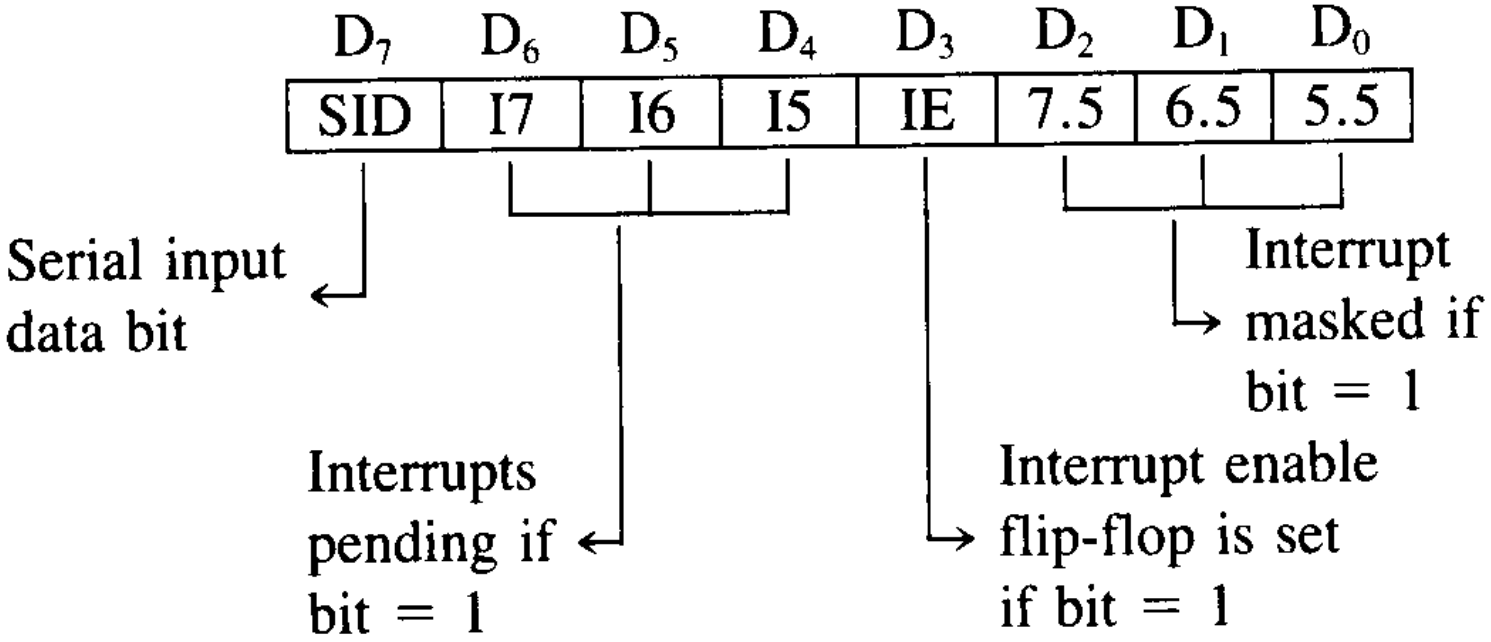
Control Operation

Opcode	Operand	Description
RIM	None	Read Interrupt Mask

- This is a multipurpose instruction used to read the status of interrupts 7.5, 6.5, 5.5 and read serial data input bit.
- The instruction loads eight bits in the accumulator with the following interpretations.
- **Example:** RIM

Control Operation

RIM Instruction



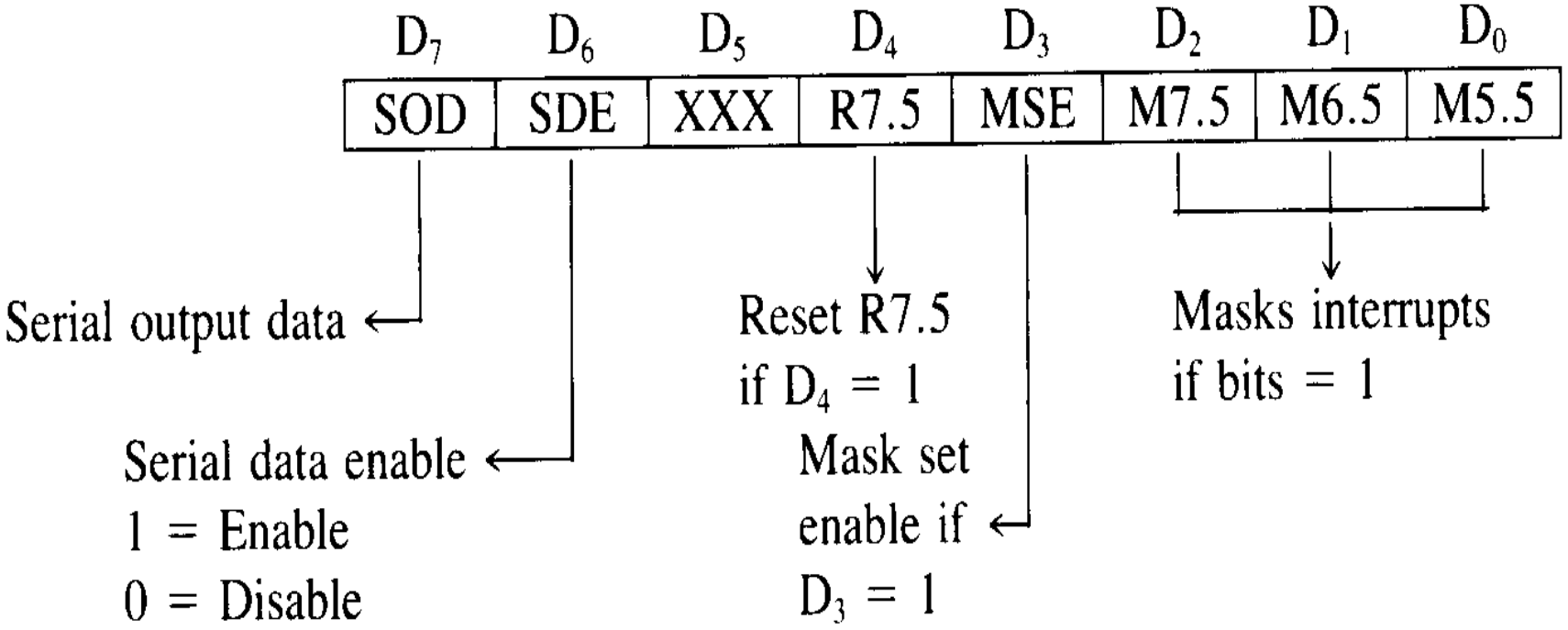
Control Operation

Opcode	Operand	Description
SIM	None	Set Interrupt Mask

- This is a multipurpose instruction and used to implement the 8085 interrupts 7.5, 6.5, 5.5, and serial data output.
- The instruction interprets the accumulator contents as follows.
- **Example:** SIM

Control Operation

SIM Instruction



Video Link

https://www.youtube.com/watch?v=ekkoIeonyzg

Daily Quiz

- What does the last instruction of each subroutine that transfer the control to the instruction in the calling program with temporary address storage , called as?
  A. jump to subroutine
  B. branch to subroutine
  C. return from subroutine
  D. call subroutine
- What is SIM?
  A. Select interrupt mask
  B. Sorting interrupt mask
  C. Set interrupt mask
  D. None of these

Daily Quiz

- Which instruction is used to set the interrupt by maintaining the serial output bit in set mode of operation?
  - **A. SIM**
  - B. STC
  - C. SBI Data
  - D. SUI Data
- Which instruction indicates the transfer of program sequence to the address specified by 16 bit value if Z flag =0 ?
  - A. CZ Address
  - B. CNZ Address
  - C. CPE Address
  - D. CPO Address

Weekly Assignment

1. How many times NOP instruction will be executed in the following program;

```
MVI A, 10H
MVI B, 10H
BACK: NOP
ADD B
RLC
JNC BACK
HLT
```

Weekly Assignment

2. What is the output at 1236,1237 for the following program?

```
        MVI C,00
        MVI D,80H
        MVI E,80H
        MOV A,D
        MOV B,A
        MOV A,E
        ADD B
        JNC LOOP
        INR C
LOOP:   STA 1236
        MOV A,C
        STA 1237
        HLT
```

Recap

- Logical & control instructions of 8085 microprocessor are discussed.

Assembler Directives

Definition:

- Assembler directives are the directions to the assembler which indicate how an operand or section of the program is to be processed.
- These are also called pseudo operations which are not executable by the microprocessor.
- For example, the assembler must be told at what address to start assembling the program.
- These assembler directives are command placed in the program by the designer that provides information to the processor.

Assembler Directives

Assembler Directives of 8085

1. **DB:** ***Define Byte***

   This directive is used for the purpose of allocating and initializing single or multiple data bytes.

   Example: 2000 DB 30H,50H,40H

2. **DW:** *Define Word*

   It is used for initializing single or multiple data words (16-bit).

   Example: 2000 DW 1030H,5040H

   These two 16-bit data 1020H and 4216H are stored at 4 consecutive locations in the memory

3. **END:** *End of program*

   This directive is used at the time of program termination.

Assembler Directives

Assembler Directives of 8085

4. EQU: *Equate*

- It is used to assign any numerical value or constant to the variable.

  DONE EQU 10H

- Variable name 'DONE' has value 10H

5. MACRO: *Represents beginning*

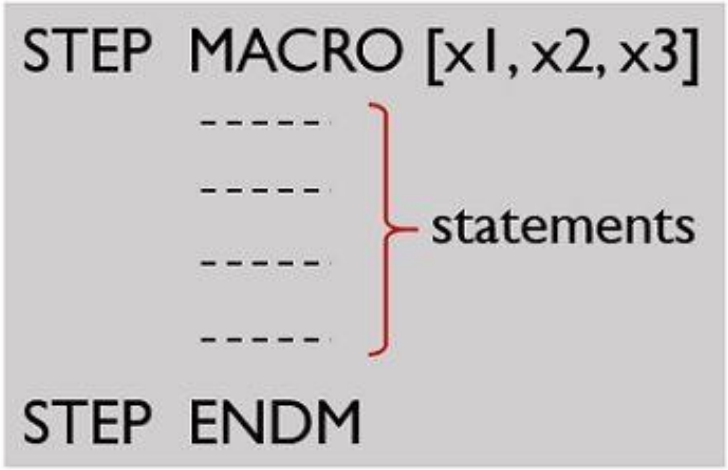
- Shows the beginning of macro along with defining name and parameters.
- A macro instruction is a request to the assembler program to process a predefined sequence of instructions called a macro.

Assembler Directives

Assembler Directives of 8085

6. ENDM: *End of macro*

- ENDM indicates the termination of macro, where macroname (STEP) is specified by the user.



7. ORG: *Origin*

- This directive is used at the time of assigning starting address for a module or segment.

  ORG 1050H

- By this instruction, the assembler gets to know that the statements following this instruction, must be stored in the memory location beginning with address 1050H.

Writing assembly language programs

Basic Programming concepts

- A program is a set of instructions arranged in the specific sequence to do the specific task.
- It tells the microprocessor what it has to do.
- The process of writing the set of instructions which tells the microprocessor what to do is called “Programming“.
- In other words, we can say that Programming is the process of telling the processor exactly how to solve a problem.
- To do this, the programmer must “speak” to the processor in a language which processor can understand is called Microprocessor Programming .

Steps Involved in Programming

1. Specifying the problem :

- The first step in the programming is to find out which task is to be performed. This is called specifying the problem.
- If the programmer does not understand what is to be done, the programming process cannot begin.

2. Designing the problem-solution :

- During this process, the exact step by step process that is to be followed (program logic) is developed and written down.

3. Coding :

- Once the program is specified and designed, it can be implemented. Implementation begins with the process of coding the program.

Steps Involved in Programming

- Coding the program means to tell the processor the exact step by step process in its language.
- Each Microprocessor Programming has a set of instructions.
- Programmer has to choose appropriate instructions from the instruction set to build the program.

4. Debugging:

- Debugging is the process of testing the code to see if it does the given task.
- If program is not working properly, debugging process helps in finding and correcting errors.

Flowchart Programming

- To develop the programming logic, programmer has to write down various actions which are to be performed in proper sequence.
- The flow chart is a graphical tool that allows programmer to represent various actions which are to be performed.
- The graphical representation is very useful for clear understanding of the programming logic.

writing assembly language programs

Symbols

- Oval : It indicates start or stop operation.
- Arrow : It indicates flow with direction.
- Parallelogram : It indicates input/output operation.
- Rectangle : It indicates process operation.
- Diamond : It indicates decision making operation.
- Double sided Rectangle : It indicates execution process (subroutine).
- Circle with alphabet : It indicates continuation.

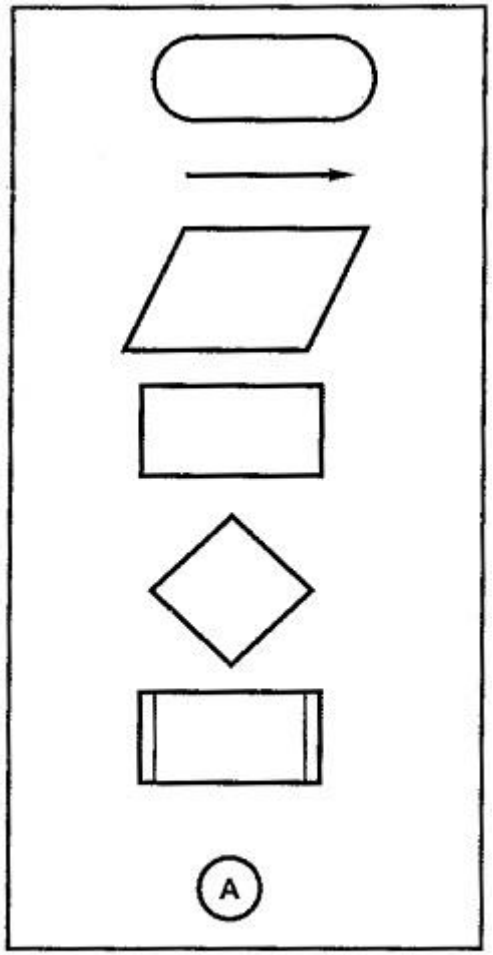


Fig. 3.1 Graphic symbols used in flow chart

writing assembly language programs

Example

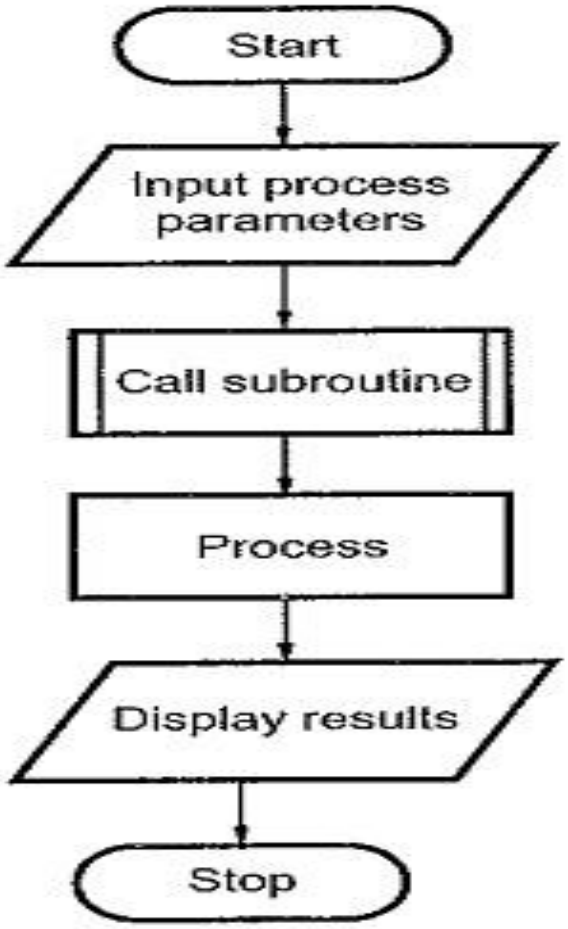


Fig. 3.2 Sample flowchart

Assembly Language Programs

1. Write a program to transfer 07 H in register L.

Memory Address	Machine Code	Mnemonics	Operands
2000 H	2E, 07	MVI	L, 07
2002 H	76	HLT	

Assembly Language Programs

2. Write a program to load register A with 08 H and then move it to register C.

Memory Address	Machine Code	Mnemonics	Operands
2000 H	3E, 08	MVI	A,08
2002 H	4F	MOV	C,A
2003 H	76	HLT	

Assembly Language Programs

3. Write a program to load the contents of memory location 2050 H into accumulator and then move this data into register B.

Memory Address	Machine Code	Mnemonics	Operands
2000 H	3A, 50, 20	LDA	2050 H
2002 H	47	MOV	B,A
2004 H	76	HLT	

Assembly Language Programs

4. The contents of memory location 2050 H are FF H. Move these contents to Register C.

Programming techniques: looping

Looping

- In this Programming Techniques in Microprocessor 8085, the Program is instructed to execute certain set of instructions repeatedly to execute a particular task number of times.
- For example, to add ten numbers stored in the consecutive memory locations we have to perform addition ten times.

Programming techniques: looping

Looping Example

Memory Address	Mnemonics	Operands
	MVI	C, FF H
LOOP:	DCR	C
	JNZ	LOOP

Programming techniques: looping

Looping Example

Memory Address	Mnemonics	Operands
	MVI	C, FF H
	MVI	D, FF H
LOOP1:	DCR	C
LOOP2:	DCR	D
	JNZ	LOOP2
	JNZ	LOOP1

Programming techniques: Counting

Counting

- This technique allows programmer to count how many times the instruction/set of instructions are executed.

Programming techniques: Counting

Counting Example:

Memory Address	Mnemonics	Operands
	MVI	B, 10 H
LOOP2:	MVI	C, FF H
LOOP1:	DCR	C
	JNZ	LOOP1
	DCR	B
	JNZ	LOOP2

Programming techniques: Indexing

Indexing

- This Programming Techniques in Microprocessor 8085 allows programmer to point or refer the data stored in sequential memory locations one by one.

Assembly Language Programs

- Write a program to shift an 8 bit data four bits right. Assume Data in C Register.

```
MOV A,C
RAR
RAR
RAR
RAR
MOV C,A
HLT
```

Assembly Language Programs

- Write a program to count no of 1's in the contents of D register & store the count in B Register.

```
        MVI B,00H
        MVI C,08H
        MOV A,D
BACK:   RAR
        JNC SKIP
        INR B
SKIP:   DCR C
        JNZ BACK
        HLT
```

Assembly Language Programs

- Write a program to find the 1's compliment of the no stored at memory location 4400H & store the complemented no at memory location 4300H.

```
LDA 4400H
CMA
STA 4300H
HLT
```

Assembly Language Programs

- Write a program to find the 2's compliment of the no stored at memory location 4200H & store the complemented no at memory location 4300H.

```
LDA 4200H
CMA
ADI 01H
STA 4300H
HLT
```

Assembly Language Programs

- Write a program to read data from a temp sensor (PORT 1) & make a relay on (PORT 2) if the temp data < 7F.

```
      IN PORT1
      CPI 7F
      JC L1
      MVI A,00H
      JMP L2
L1:   MVI A,01H
L2:   OUT PORT 2
      HLT
```

Assembly Language Programs

- Write a program for displaying the sum of two no if sum is greater than FFH otherwise display 01H. Numbers are 9BH & A7H.

```
        MVI D, 9BH
        MVI E, A7H
        MOV A,D
        ADD E
        JNC DISPLAY
        MVI A, 00H
DISPLAY: OUT 00H
        HLT
```

Assembly Language Programs

- Write a program to copy a number of byte from one memory location to another.

```
        MVI C, 0AH
        LXI H, 2000H
         LXI H, 3000H
BACK:   MOV A, M
        STAXD
        INX H
        INX D
        DCR C
        JNZ BACK
        HLT
```

Assembly Language Programs

- Write a program for sum of series of 100 bytes.

```
        MVI C, 64H
        MVI B, 00H
        MOV A, B
        LXI H, 2000H
BACK:   ADD M
        JNC NEXT
        INR B
NEXT:   INX H
        DCR C
        JNZ BACK
        STA 3001H
        MOV A,B
        STA 3001H
        HLT
```

Video Link

https://www.youtube.com/watch?v=CPgFscob2T8

Daily Quiz

- In an Intel 8085A, which is the first machine cycle of an instruction?
  - a. An op-code fetch cycle
  - b. A memory read cycle
  - c. A memory write cycle
  - d. An I/O read cycle
- What are level Triggering interrupts?
  - a) INTR&TRAP
  - b)RST6.5&RST5.5
  - c)RST7.5&RST6.5
- What is the RST for the TRAP?
  - a) RST5.5
  - b) RST4.5
  - c) RST4

Daily Quiz

- Which interrupt is not level sensitive in 8085?
  - a) RST6.5 is a raising edge-trigging interrupt.
  - b) RST7.5 is a raising edge-trigging interrupt.
  - c) a & b.
- RIM is used to check whether, _____
  - a) The write operation is done or not
  - b) The interrupt is Masked or not
  - c) a & b
- What is meant by Maskable interrupts?
  - a) An interrupt which can never be turned off.
  - b) An interrupt that can be turned off by the programmer.
  - c) none

Recap

- Problem related to T States & examples of assembly level languages are discussed.

Daily quiz

- Which of the following is used as a primary storage device?
  A. Magnetic drum
  B. PROM
  C. Floppy disk
  D. All of these
- Which of the following memories needs refresh?
  A. SRAM
  B. DRAM
  C. ROM
  D. All of above

Weekly Assignment

1. How many times NOP instruction will be executed in the following program;

```
MVI A, 10H
MVI B, 10H
BACK: NOP
ADD B
RLC
JNC BACK
HLT
```

Weekly Assignment

2. What is the output at 1236,1237 for the following program?

```
        MVI C,00
        MVI D,80H
        MVI E,80H
        MOV A,D
        MOV B,A
        MOV A,E
        ADD B
        JNC LOOP
        INR C
LOOP:   STA 1236
        MOV A,C
        STA 1237
        HLT
```

Topic links

Youtube /other Video Links

https://www.youtube.com/watch?v=iJmcgQRk048

https://www.youtube.com/watch?v=ekkoIeonyzg

https://www.youtube.com/watch?v=79icCUmqyPc

https://www.youtube.com/watch?v=Bn0FhaaH6RA

MCQ s

- Which of the following are the two main components of the CPU?
  A. Control Unit and Registers
  B. Registers and Main Memory
  C. Control unit and ALU
  D. ALU and bus
- The language that the computer can understand and execute is called:
  A. Machine language
  B. Application software
  C. System program
  D. All of the above

MCQ s

- Which of the following is used as a primary storage device?
  A. Magnetic drum
  B. PROM
  C. Floppy disk
  D. All of these
- Which of the following memories needs refresh?
  A. SRAM
  B. DRAM
  C. ROM
  D. All of above

MCQ s

- What is another name of memory stack especially given for the fundamental function performed by it?
  - **A. Last-in-first-out (LIFO)**
  - B. First-in-last-out (FILO)
  - C. First-in-first-out (FIFO)
  - D. Last-in-last-out (LILO)
- Registers, which are partially visible to users and used to hold conditional, are known as
  - A. PC
  - B. Memory address registers
  - **C. General purpose register**
  - D. Flags

Glossary questions

(Machine language, Control unit and ALU, DRAM, PROM)

Choose the correct one from above :

What are the two main components of the CPU?

The language that the computer can understand and execute is called:

Which is used as a primary storage device?

Which memories needs refresh?

Old question papers

1. Write an ALP to multiply two eight bit numbers.
2. Write a program for transferring the contents of Memory location 2050 H to the register B and the contents of 2051 H to Register C. The contents of memory location 2050 H are 10 of 2051 H are 11.
3. Find the total time to execute the above program if the frequency of the processor is 5MHz.

```
1.          MVI  A,17H
2.  LOOP:   RAL
3.          ORA  A
4.          JNC  LOOP
```

Expected Questions for University Exam

1. Explain the sequence of events during the execution of the CALL instruction by 8085 processor with the help of neat timing diagram.
2. Write an assembly language program with comment lines. An 8-bit number is stored in memory location 2A00H. Count number of ones (i.e. 1) in this byte and store this count in memory location 3200H.
3. Explain the following instructions

   CALL, DAD B, XTHL, STAX B, CMP M
4. What is meant by interrupt? And explain different types of interrupt available in 8085.
5. Write a Program to perform the following functions and verify the output steps:
   a) Load the number 5C22H in DE register pair
   b) Move the number 9EH in register C

Recap of unit

This chapter discusses instruction set of 8085 Microprocessor.

Thank You