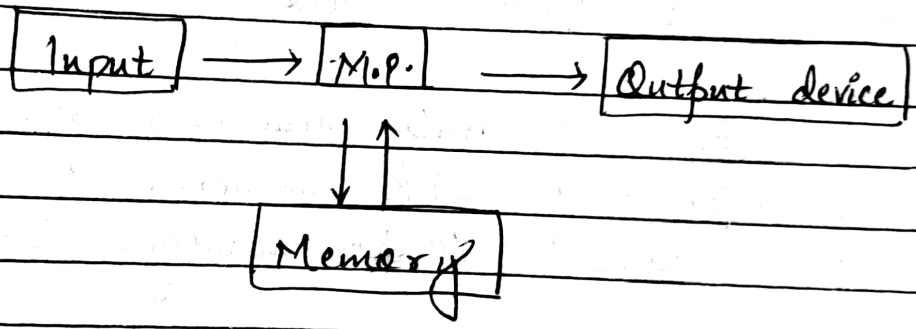
DATE: / /20
PAGE NO.

Micro-Processor

It's a programmable multi-purpose clocked electronic device that reads binary I/Ps from a memory components and then executes I/s via control units.

fetch → execute I/Os



① I/s set → Set of I/s that M.P. can understand.

② Band width → No. of bits proceed in a single I/Os.

③ Clock speed/ rate → No. of operations per second processor can performed (GHz, MHz) It determines the performance of the procesor.

④ Word length → It depends on width of internal bus, register, ALU.

(5) Data types → Data types format like BCD, Binary, ASCII, signed & unsigned.

Evolution of M.P.

(1) 4-bit micro-processor Intel - 1970s

- 4004 → 2300 transistors, intel company, 640 bytes memory, 750 kHz frequency, 6000 Operations per second of speed
- 4040 → advanced version of 4004 with new updations.

(2) 8-bit M.P. 1972

- 8008 → 16kB memory, 200kHz frequency
- 8080 → 64 kB memory, 2 MHz frequency
- 8085 → 64 kB memory, 5MHz frequency

(3) 16-bit M.P. 1978

- 8086 → 1MB memory, 5MHz frequency
- 8088 → 1MB memory, 5MHz frequency
- 80286 → 16MB memory, 8MHz frequency

DATE : / /20
PAGE NO.

(4) 32- bit Micro-processor

- 80386 → 4 GB memory, 16 MHz fregnency
- 80486 → 4 GB memory, 25 MHz fregnency
- Pentium Pro
  4GB memory, 60 MHz fregnency
- Pentium Pro
  64 GB memory, 150 MHz fregnency

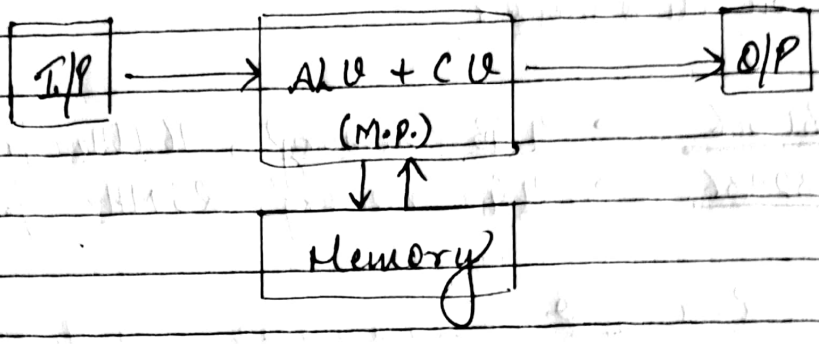
(5) 64- bit Micro-processor

- Pentium II → 1997
  64 GB memory, 233 MHz fregnency
- Pentium III → 1999
  64 GB memory, 650 MHz fregnency
- Pentium IV → 2000
  64 GB memory, 1.4 GHz fregnency
- (Intel) Dude Code → 2005
  In 2005, Intel company released first Dude Code Processor.

Working of M.P. (features, architectures)

INSTRUCTION CYCLE
Fetch → Execute → Decode

- To fetch the data or take data from any device (memory).
- Execution processing
- Decode in human lang. readable via us.



- I/P → which sends the I/S or info. like keyboard, mouse.
- O/P → display the result after processing the data like monitor or screen.

SYSTEM BUS

A communication path bus in M.P.
3 types → Data, Address, Control Bus

→

Data bus	Control Bus	Address Bus
• It's used to carry binary data b/w CPU & memory and I/P & O/P device.	It controls all the management of the system.	It is used to carry the address from memory and I/P device.
• It's unidirectional.	It only controls.	It's bidirectional
• Based on width data bus, we can determine the word length of CPU & even determine the performance of CPU.	Control signals indicate the types of an operations, timing signal is used for synchronisation of memory & operations with CPU clock.	It depends on the width of address bus, determine the capacity of the main memory.

DATE: / /20
PAGE NO.

22-1-25

→ Advantages of M.P.

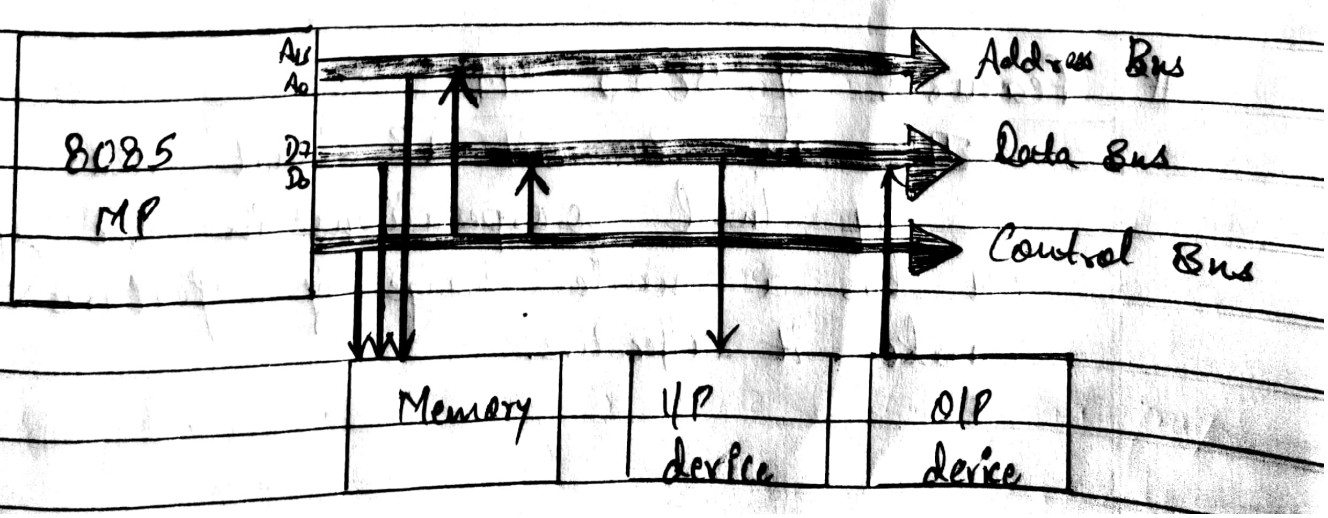
- High processing speed.
- Compact size.
- Easy maintainance
- Flexible / Versatile / Reliable
- Complex solves
- Impressions

→ Disadvantages of M.P.

- large broad size
- NO support of flops
- Performance depends on size of data
- Overheating due to overuse

* Bus Organisation System

8085 → 8-bit of data
→ 8 Data lines ($D_0 - D_7$)
→ 16 Address lines ($A_0 - A_{15}$)



DATE : / /20
PAGE NO.

→ Address Bus (0-15) = 16 bases.

Min code	0	0	0	0	0000H
	0000	0000	0000	0000	
Max code	F	F	F	F	FFFFH
	1111	1111	1111	1111	

H → Hexadecimal

→ Data Bus (0-7) = 8 bases

Min code	0	0	00H
	0000	0000	
Max code	1111	1111	FFH
	F	F	

Operations

→ MEMR → Read operation
MEMW → Write operation
IOR → I-O read
IOW → I-O write

Features of 8085 M.P.

It's an 8- bit M.P. that was introduced by Intel Corporations in the year 197 It has a wide range of features that make it effective.

DATE : / /20
PAGE NO.

① 8-bit Data Bus
8085 has an 8-bit data bus that can transfer data b/w M.P. & other devices.

② 16-bit Address Bus
8085 has 16-bit address bus that can access upto 64 kB of memory.
↳ memory locations

③ 3MHz Clock Speed
8085 has a clock speed of 3MHz which means it can execute upto 3 millions i/s per second.
{64 K → $2^6 \times 2^{10} = 2^{16}$ memory locations (approx)}

④ 8-bit ALU
8085 has 8-bit ALU that can perform arithematic & logical operation on 8-bit datas.

⑤ 8-bit registers
8085 has 6 8-bit registers (A, B, C, D, H, L) that can be used for storing & manupulating data.

⑥ Interrupt
8085 supports 5 interrupts which can be used to interrupt the M.P.'s normal operations. It has also 8-Software interrupts also.

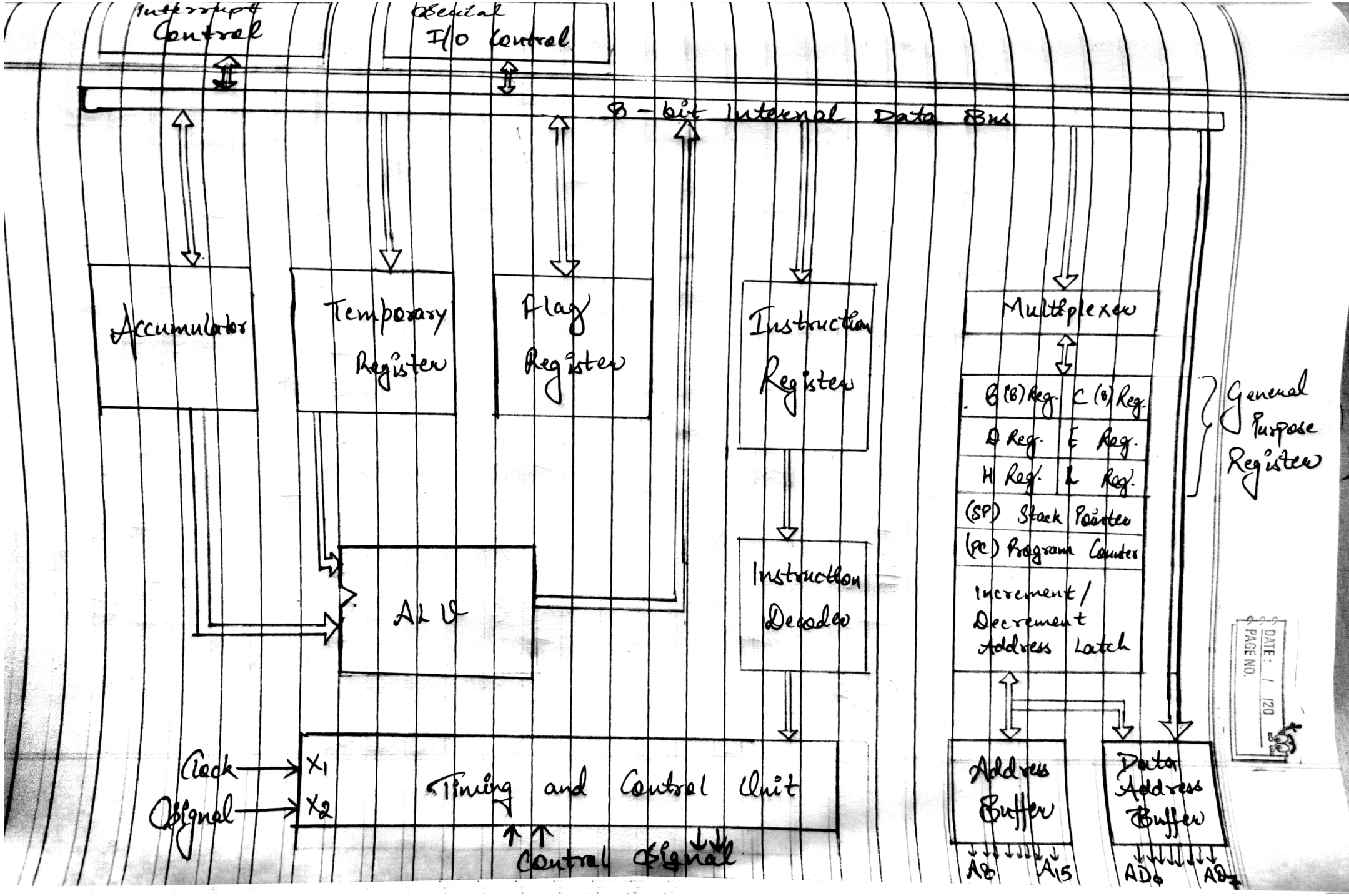
DATE: / /20
PAGE NO.

⑦ Serial I/O
8085 has 2 serial I/O codes, i.e., SID and SOD, that can be used to communicate with other device.

⑧ I/S set
It provides I/S with only 5 addressing modes, which includes arithematic, logical & controlled I/S.

⑨ Single +5V Power Supply
8085 M.P. can operate on single +5V power supply which makes it easy to use in various applications.

⑩ Low Power consumption
8085 has low power consumption which makes it ideal for battery power application.



27-1-25

Description

It's a 8-bit M.P. designed by Intel Company using n-MOS technology. It has following configurations -

- 8-bit Data Bus
  Data is of size 8-bit only.
- 16-bit address Bus
  which can addresses upto 64 kB memory
- 6 8-bit Registers
  arranged in pairs; B C, D E, H L.
- Stack Pointer
  It has 6-bit S.P.
- Power Supply
  +5V supply to operate 2.5 MHz single phased clock.

→ Uses
used in washing machines, micro waves, ovens, mobile phones etc.

Functional Units

(1) Accumulator
It's a part of arithematic & logic section. It's 8-bit registers used to perform ALU operations.

DATE : / /20
PAGE NO.

The result of arithematic & logical unit operation are stored in accumulator. It's identify by A register.

② ALU

It performs arithematic & logic operation like "add", "sub" and OR, XOR etc. The result of these operations stored in the accumulator. It provides status of result to flag register.

③ General Purpose Register

There are 6 general purpose registers in 8085 M.P; BC, DE, HL. Each register can hold 8-bit data. These registers can use in pairs to hold 16-bit data and their pairing combination like BC, DE and HL.

④ Program Counter - PC

It's a 16-bit register used to store a memory address location of next I/Ps to be executed. M.P. increment the program whenever I/s being execute.

⑤ Stack Pointer - SP

It's also a 16-bit registers work like stacks, which is always Input - Output by during push & pop operation take place. The Stack is a set of memory location in

the read-write memory defined by a programmer.

(6) Temporary Register
It's a 8-bit registers which hold the temporary data of ALU operations. They are also used for internal operation of M.P.

(7) Flag Register
It's a 8-bit registers have 5 1-bit flip-flops which holds either 0 or 1 depending upon results stored in an accumulator.

1) Sign Flag
2) Parity Flag
3) Zero Flag
4) Auxilary Flag
5) Carry Flag

28-1-25

1) Sign flag
After the execution of ALU operation, if 8th bit of result is 1, then sign flag is said to be 1, otherwise 0.
(MSB = 1, sign = 1, otherwise 0.)

2) Zero Flag
After the execution of ALU operation, if content of accumulator or result is 0, then zero flag is said to be 1.
(MSB = 0, ZF = 1)

DATE: / /20
PAGE NO.

3) Auxiliary Flag

In arithmatic operation, when carry is generated by bit D_3 and passed to D_4, then AC F is said to 1 otherwise 0.

eg.

D_7 D_6 D_5 D_4	D_3 D_2 D_1 D_0
0 0 1 0	1 0 1 1
0 1 0 0	1 0 0 0
0 1 1 1	0 0 1 1

4) Parity Flag

After the execution of ALU operation, if the result of accumulator, even no.'s is 1 then Pf is said to be 1, otherwise 0.

eg. if 1001 1000 then, Pf = 0
if 1110 0100 then, Pf = 1

5) Carry flag

After the execution of ALU, if carry is generated then carry flag is said to be 1, otherwise 0.

eg.

$$\begin{array}{r} (1)\ 1\ 0\ 0\ 1\ 0\ 0\ 1\ 0 \\ 1\ 1\ 0\ 1\ 1\ 0\ 0\ 1 \\ \hline 1\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1 \end{array}$$

↳ C.f. = 1

DATE: / /20
PAGE NO.

(8) I/S Register & Decoder

It's an 8-bit register when 'n' I/Ss is fetched from memory. It is stored in I/Ss registers. It holds the operation code of current I/S to be executed.

And, the instruction decoder decodes the I/S & identify the operation to be performed which is present in I/S register.

(9) Timing and Control Units

It provides timing & control registers and signal to M.P. to perform operations following the timing & control signal. It generates the control signal which are necessary for communication b/w M.P. and peripheral devices which control external & internal circuits.

Control signal → Ready, $\overline{RD}$, $\overline{WR}$, ALE

ALE = 1 → address $AD_0 - AD_7$
ALE = 0 → data $AD_0 - AD_7$

Status signal → S_0, S_1, IO/$\overline{M}$, ~~SMA~~
IO/$\overline{M}$ = 0, IO - M.P.
IO/$\overline{M}$ = 1, $\overline{M}$ - M.P.

DMA → HOLD, HLDA
Reset signal → Reset In, Reset Out

DATE : / /20
PAGE NO.

(10) Interrupt Controller

As the name suggests it control the interrupt during process when M.P is executing a main programs & whenever an interrupt occur the M.P shift the control from main program to process the incoming request. After the request is completed, the control goes back to the main program, there are five interrupt signals in 8085 M.P.

INTR, (Reset) RST 7.5, RST 6.6, RST 5.5, TRAP.

(11) Serial I-O control

It controls the serial data communicate by using these 2 I/S - SID and SOD.

SID → Serial I/P data lines

It takes one bit data from serial port of 8085 and store the bit at the 8th position, i.e., MSB of accumulator, and RIM - Real Internal Mask I/S is used to transfer the bit for SID.

SOD → Serial O/P data

It is a serial O/P data line. It takes 1 bit to accumulator to serial port of 8085 & takes the bit from 8th position MSB of accumulator and SIM - Set Interrupt Mask I/S is used to transfer the bit for SOD.

DATE: / /20
PAGE NO.

(12) Address Bus and Data Bus

Data bus carry the data to be stored. It's bidirectional. whereas, address bus carry the location to where it should be stored and the address bus is unidirectional, i.e., address flows only in 1 direction from M.P. to peripheral device.

It's used to transfer data and address to I-O devices.

(13) Address Buffer

The content stored in Stack Pointer (SP) & Program Counter (PC) is loaded into the address buffer and this communicate with the CPU.

Pin-Description Configuration of 8085

1 & 2 } Crystal Oscillator → generate freq 6 MHz but 3 MHz is needed

3 Active high → works on 1

4 Output given one by one through RIM I/O

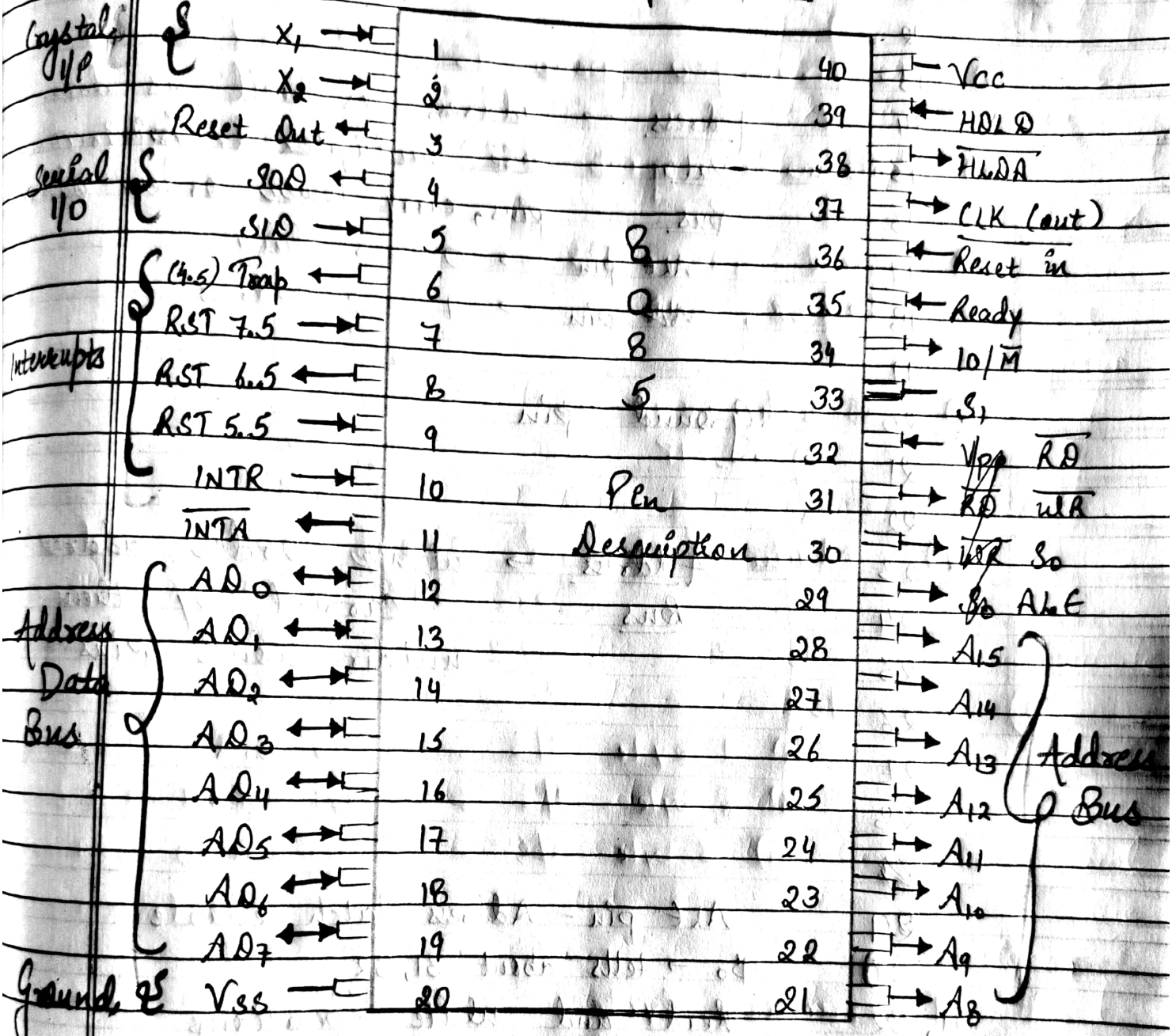
5 Input taken by MSB through SIM I/O

6, 7, 8, 9 } Vectored Interrupts → have specified location in memory

10 Interrupt Request is known, given by Interrupt Pin

DATE : / /20
PAGE NO.

Pin Descriptions of 8085



Interrupts → Non vectored I. → no memory → INTR, INTA
→ Vectored Interrupt → no memory size → RST, TRAP

S_0	S_1	Operations
0	0	HOLD (stop)
0	1	READ
1	0	WRITE
1	1	OP code fetch

DATE: / /20
PAGE NO.

11 Interrupt Acknowledgement, (permission)
12
13
14 Address → address as well as data,
15 Data are multiplexed, bidirectional
16 Bus pins, carry lower order addres
17 ALE pin → 0 → data
18 ALE pin → 1 → address
19
20 Ground pin
21
22
23 Address → higher order address
24 Bus carried by them,
25 unidirectional pins
26
27
28
29 ALE pin - Address latch enable
30 ϕ_0 → tells about 31, 32
31, 32 Read and write operations
33 ϕ_1 → tells about 31, 32
34 0 → Memory & 1 → I/O device
35 tells that M.P. is ready to take I/p
36 make PC = 0, works in active low → on
37 tells a completion of single clock
38 Acknowledge pin send by MP to I/O devices
39 HOLD the address & data bus on request
40 Supply → +5 V

5-2-25

DATE : / /20
PAGE NO.

INSTRUCTION

→ OP code + operand
→ depends on 2 parts → size and functionality

An I/S is a command given to comp. to perform specified operation on the given data. eg. →

MOV C,A

this instruction will move the content of register A to the register C.

OP code format

Each I/S contains 2 parts operation code, i.e., OP code & operand.

- The first part of I/S specified the task to be performed by the comp. is called OP code.
- The second part of I/S is a data to be operated on is called operand. The operand given I/S may be various types such as 8-bit data or 16-bit data.

→ The I/S code be divided in 2 ways where M.P. works → ① Functionality I/S
② Size Type I/S

DATE : / /20
PAGE NO.

① FUNCTIONALITY I/S

(i) Data transfer group I/S → MOV B, 25H

(ii) Arithematic group I/S → ADD, SUB

(iii) Logical group I/S
→ X-OR, AND

(iv) Branch group I/S
→ JMP

(v) Machine Control Group I/S

② SIZE I/S

(i) 1-byte I/S :- MOV B, A → 8-bits

(ii) 2-byte I/S :- MVI A, 05H → 16-bits

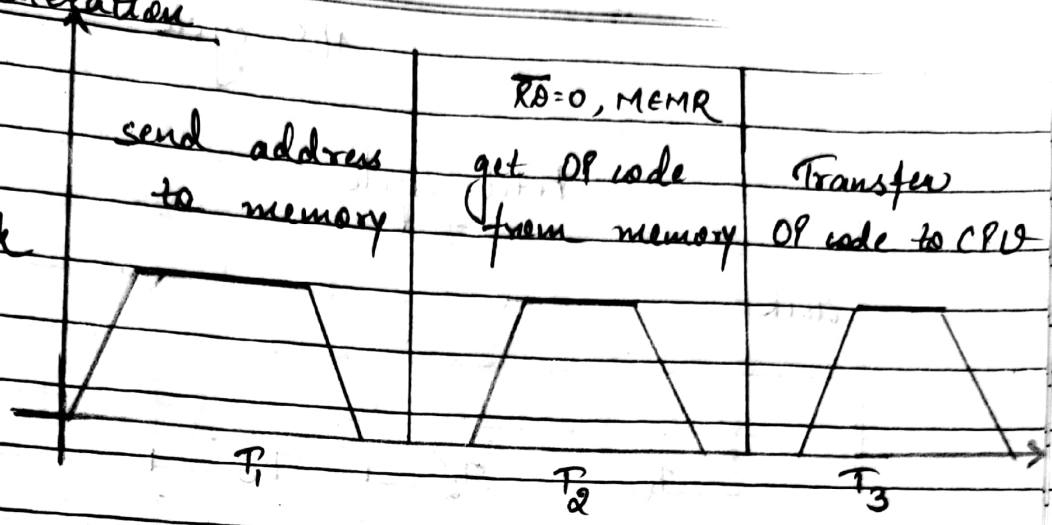
(iii) 3-byte I/S :- JMP 2050 → 24-bits
8 16-address

Instruction Cycle

fetch → decode → execute

1-2-25

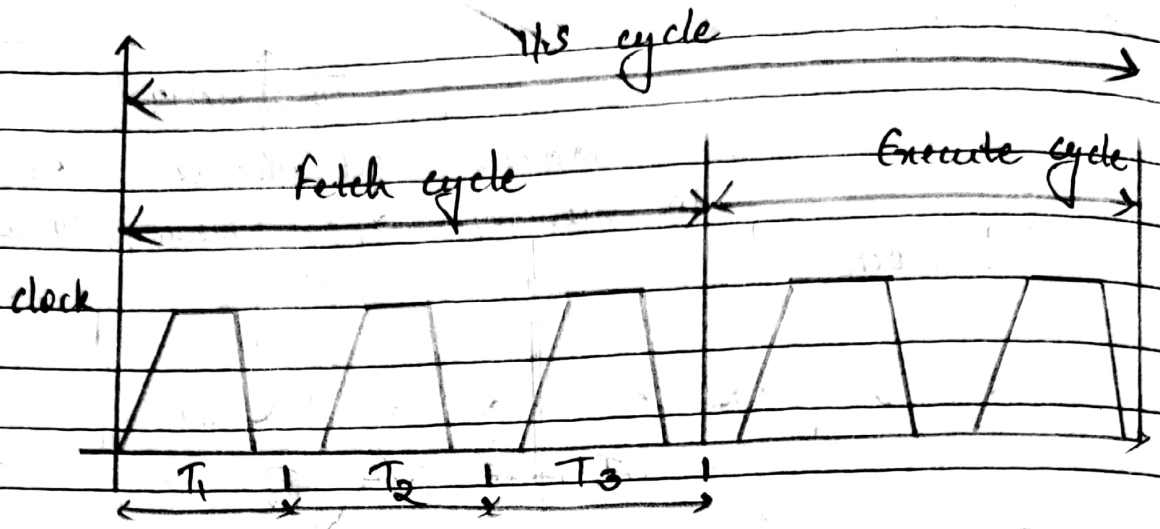
Fetch Operation



The PC keeps the ~~track~~ address of next I/S to be executed. In the beginning of fetch operation, the content of PC which is the address of the memory location where OP code of I/S is stored is sent to the memory. The memory places the OP code of the I/S on the data bus so as to transfer it to CPU. This is called fetch operation. To fetch an OP code of I/S, 3 clock cycles are required.

Execute Operation

After the OP code of an I/S is fetched from the memory, the execution begins. OP code fetched goes to data register, then it goes to I/S decoder which decodes the I/S. After decoding the I/S the task specified in the I/S is carried out. This is called execute operation.



Machine Cycle

In I/S cycle have machine cycle and in ~~each~~ machine cycle has T states. The necessary steps which are required to perform fetch operation or read or write operation constitutes a machine cycle. In machine cycle one basic operation such as OP code fetch, I/O read, I/O write, memory read or memory write operation. I/S cycle consists of several machine cycles. An operation performed in 1 clock period is called 'T' state.

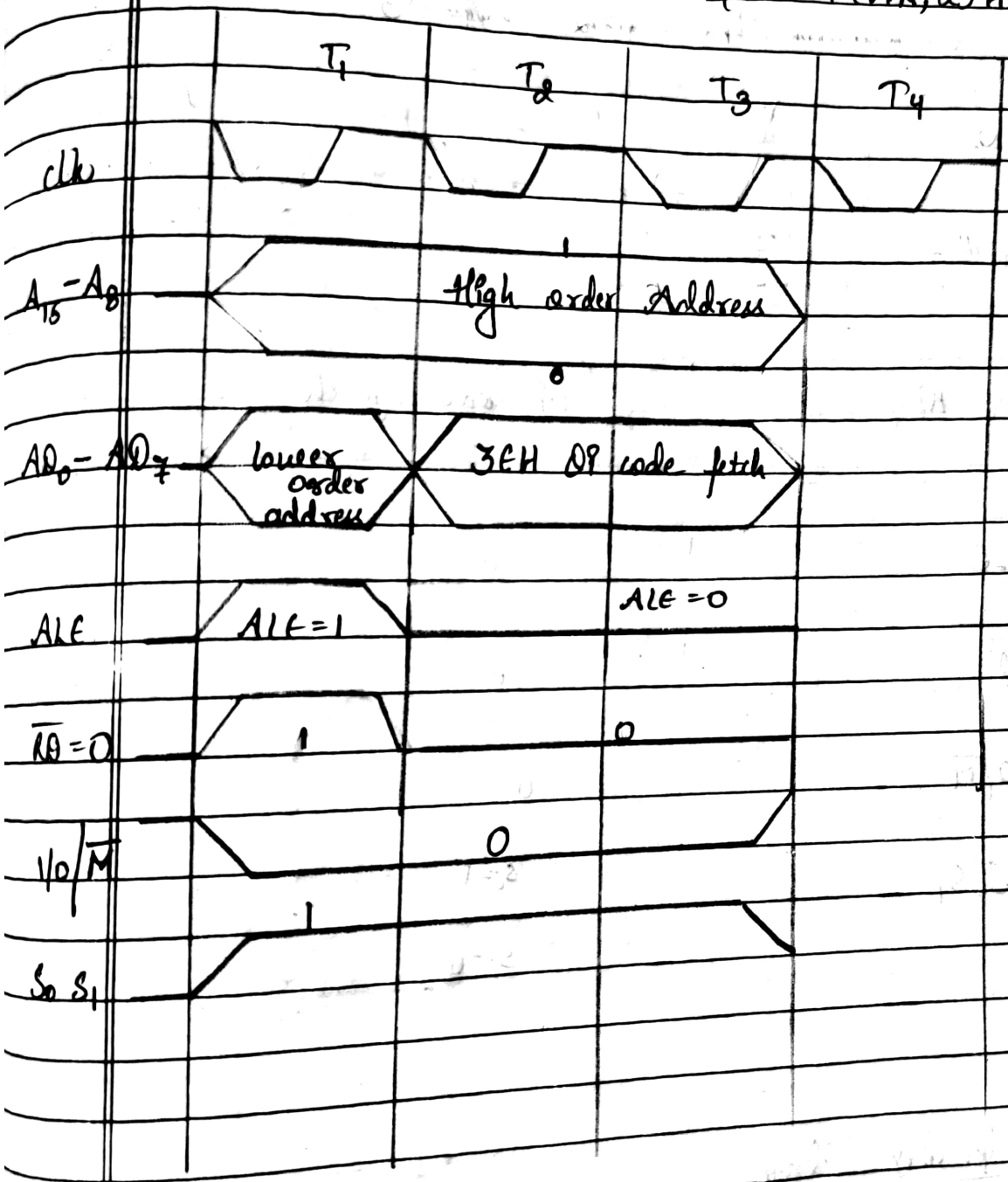
Timing Diagram

It is a graphical representation of I/S cycle is known as timing diagram. It performs various operations.

1. OP code fetch
2. Memory Read
3. Memory Write
4. I/O Read
5. ...

- Op Code fetch

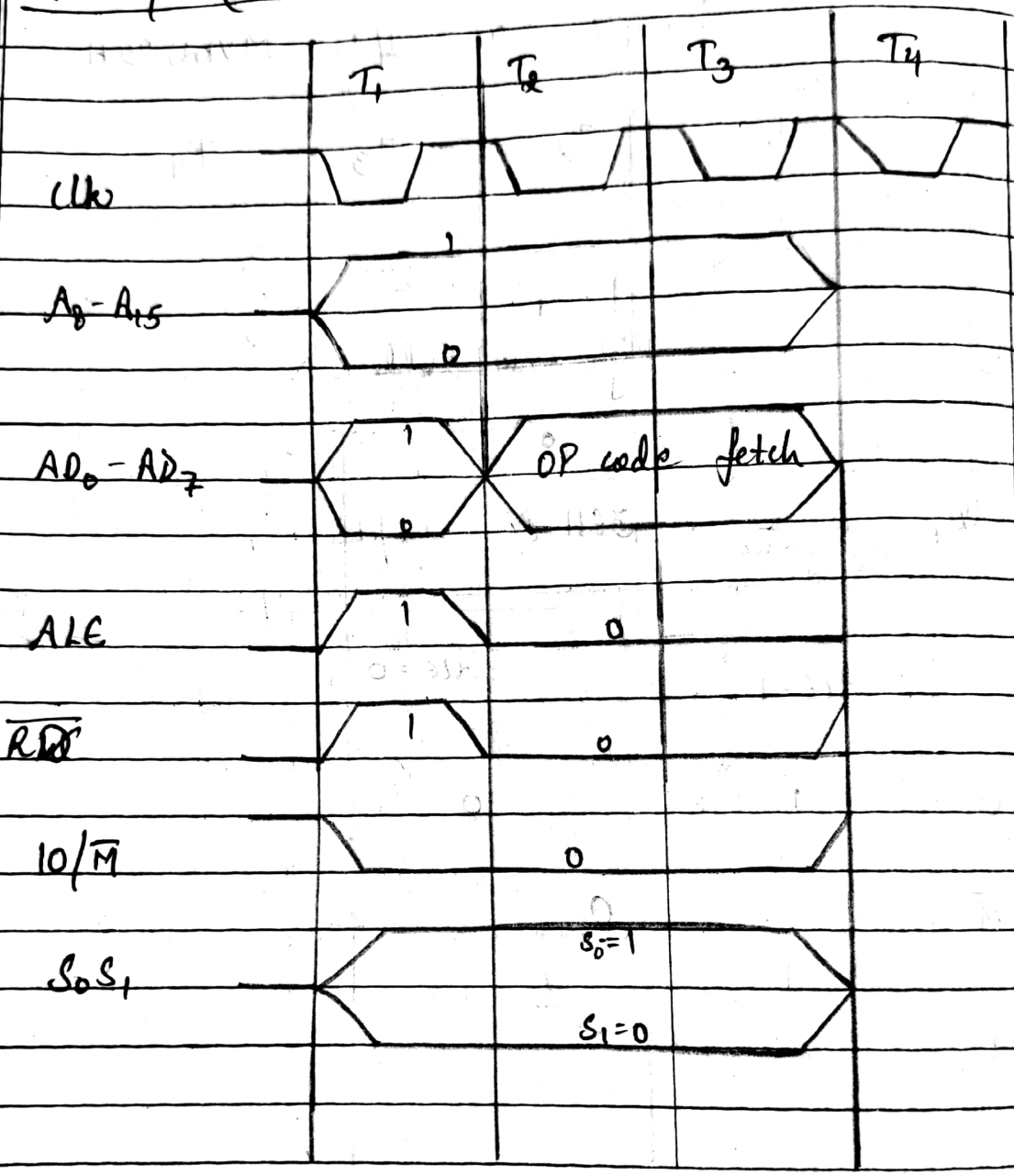
Consider an I/S MVI A, 25H



DATE : / /20
PAGE NO.

10-2-25

- **Memory Read**



- **Memory Write**

$\overline{RD} \longrightarrow \overline{WR}$

$IO/\overline{M} \longrightarrow 0$

$S_0\ S_1 \longrightarrow S_0 = 0,\ S_1 = 1$

/

DATE : / /20
PAGE NO.

Addressing Modes

It's known as the every I/S required a data on which it operates a particular function. The data are specified in different modes in the I/S. Various ways of specifying data are called addressing modes.

8085 MP has 5 addressing modes -

- Immediate A.M.
- Register A.M.
- Direct A.M.
- Indirect A.M.
- Implied A.M.

① Immediate Addressing modes

eg. MVI Reg, Data (Data move in register)

When the data is directly specified in the I/S is called immediate addressing modes.

→ MVI B, 72H

→ LXI H, 5000H

load pair immediate, HL pair registers

② Register Addressing Modes

[Register to Register]

When the data is stored in register & if the register is specified in the I/S, then it's called register addressing modes.

eg. → MOV Reg, MOV Rd, Rs

(destination) (source)

→ MOV D, C

Content of C is moved to D

DATE: / /20
PAGE NO.

③ Direct Addressing Modes
[Content of address in accumulator]
When memory address specified within I/S, then it's called direct addressing modes.
eg. → LDA 5000 H
5000H memory content is stored into an accumulator.

④ In-direct Addressing Modes
(H-L pair register has address, content to accumulator) load the data to the accumulator from the memory (memory pointer), the data to be pointed by the H-L pair register, is indirect addressing H.
eg. → MOV A, M
The data moved to the accumulator from memory which is pointed by memory pointer.

⑤ Implied Addressing Modes
(direct operations) or instructions)
When the I/S itself specifies the data to be operated, then it's called implied addressing modes. for eg. → CMA
means, compliment the content of accumulator

→ RRC
rotate accumulator content without carry

DATE : / /20
PAGE NO.

11-2-25

* Instruction sets

There are 5 types of I/S set.

(1) Data transfer I/S

The group of I/S copies data from a location is called source, to another location called a destination, without modifying the content of a source. The term data transfer is used for copying data.

Various data transfer I/S are-

(i) MOVE → MOV Rd, Rs

- 1 byte I/S (Opcode + operand)
- move data from 1 register to another
- eg. MOV B, C

(ii) → MOV Rd, M

- content of memory location into Rd. (Opcode + operand)
- eg. MOV B, M

(iii) MVI

- move immediate 8-bit data
- direct data is given
- MVI Rd, Data and MVI M, Data
- eg. → MVI B, 92H and MVI M, 3AH

(iv) LXI (load register pair immediate)

- 16-bit data
- eg. → LXI Rp, 16-bit data
- LXI B, [illegible]

(v) LDA, Address (M) (load accumulator direct)
- eg. → LDA, 2400H
- Direct addressing modes

(vi) STA, Address (store accumulator direct)
- eg. → STA, 2000H

(vii) LHLD, Address (load H-L register direct)
- eg → LHLD, 2500H

(viii) SHLD, Address (store H-L register direct)
- eg → SHLD, 2050H

(ix) LDAX, Register (load accumulator indirectly by pair registers)
- eg → LDAX, B
  ↳ LDAX, B-C

(x) STAX Register (Store accumulator indirect)
- eg → STAX B
  (content of pair register B-C to accumulator)

2-25

② Arithematic Group I/Os

8085 µP can perform various arithematic I/s like add", sub", increment, decrement etc. In add" & sub" operations, accumulator is one of the operand. However, operations like increment & decrement can performed on any register.

(i) ADD, R (add" of R with accumulator)

• eg. ADD, B
↳ (content of B added to accumulator)

let,

Ac → 47H
B → 51H

0100 0111
0101 0001
1001 1000
↳ 98H accumulator

(ii) ADD, M (HL pair registers with accumulator)

• eg. ADD, M

(iii) ADC, R (add R to Acc. with carry)

• eg. ADC, B [set Carry flag]

(iv) ADC, M (add content of M to Acc. with carry)

• eg. ADC, M [Carry flag = 1]

(v) ADI, 8 bit Data (add immediate, data with Acc.)

• eg. ADI, 59H

(vi) ACI, Data (add immediate data with carry)

• eg. ACI, 57H [Carry flag = 1, if

(vii) DAD

(viii) SUB, R (sub R from accumulator)

• eg. SUB, C
↳ (sub content of C from Acc.)

DATE : / /20
PAGE NO.

C = 40H
A = 37H

40 1's C → 0100 0000
→ 1011 1111
+1
2's C 1100 0000 → C0

C0 + 40H, 1100 0000
0100 0000
1 0000 0000 → [illegible]

(ix) SUB, M (sub M from Acc.)
(x) SBB, R (Sub. R from accumulator with borrow)
eg. SBB, B

(xi) SBB, M (sub. M with borrow from Acc.)
(xii) SUI, 8-bit Data (sub immediate Data from Acc.)
• eg. SUI, 58H

(xiii) SBI, Data (Sub I Data with borrow from Acc.)
• eg SBI, 59H

(xiv) INR, R (increment R's content by 1)
• eg. INR, B

(xv) INR, M (H-L pair increment)
(xvi) DCR, R (decrement R's content by 1)
• eg DCR, B

(xvii) DCR, M (H-L pair decrement)
(xviii) INX, H (Increment pair H-L registers)
(xix) DCX, R (decrement content of R pair by 1)

18-2-25

(b) Logical Set I/S (Hex-code)

The 8085 M.P. can perform logical operations like AND, OR, XOR, Complement etc. All the logical operations are performed with the content of accumulator, result of operation is placed in accumulator.

(i) ANA, R (Logical AND Register with accumulator)
eg. ANA, B

A ⇒ 54, B = 82, result A → 00H

(ii) ANA, M (Logical AND memory (H-L) with accumulator)

(iii) ANI, Data (AND immediate with accumulator)
eg. ANI, 97H

(iv) ORA, R (Logical OR Register with accumulator)
eg. ORA, B

(v) ORA, M (Logical OR memory (H-L) with accumulator)
eg.

(vi) ORI, Data (OR immediate with accumulator)
eg. ORI, 42H

(vii) XRA, R (X-OR register with accumulator)
eg. XRA, B

(viii) XRA, M (X-OR memory with accumulator)

(ix) XRI, Data (X-OR immediate data with accumulator)
eg: XRI, 42H

(x) CMA (complement the content of accumulator)

(xi) CMC (complement carry flag)

(xii) STC (set carry flag) [set 1]

(xiii) CMP, R (Compare register with accumulator)

→ if A < R, then carry flag is set to be 1 and 0 flag is reset, i.e., 0.
→ if A = R, then carry flag is reset, i.e., 0 and 0 flag is set to be 1.
→ if A > R, then carry & 0 flag are reset, i.e., 0.

eg. CMP, B
if A = 57, B = 62 then A < B
so, carry flag = 1, 0 flag = 0.

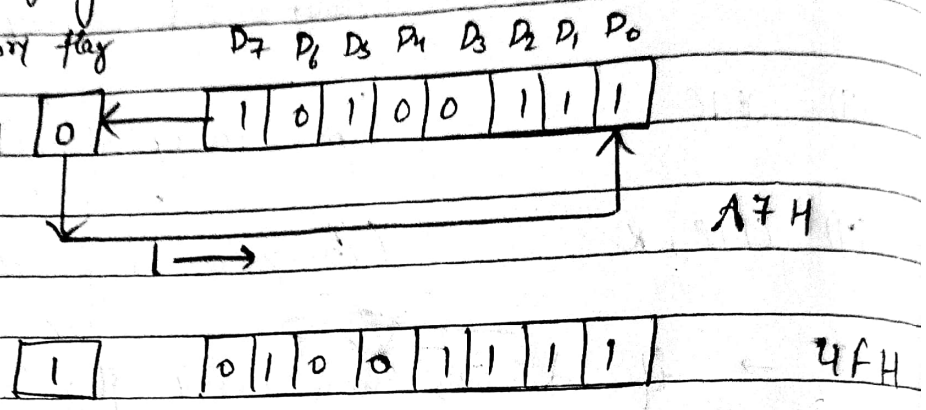
(xiv) CMP, M (compare memory with accumulator)

→ if A < R, carry flag = 1, zero flag = 0
→ if A = R, carry flag = 0, zero flag = 1
→ if A > R, carry flag = 0 = zero flag

(xv) CPI, Data (compare immediate data with accumulator)
→ if A < Data, carry flag = 1, zero flag = 0
→ if A = Data, carry flag = 0, zero flag = 1
→ if A > Data, carry flag = 0, zero flag = 0

(xvi) RLC (rotate accumulator left with carry flag)

The bit D_7 in accumulator is moved in the bit position D_0 as well as the carry flag.



(xvii) RAR (rotate accumulator right with carry flag)

The bit D_0 is moved in the position of D_7 as well as in the carry flag.

(xviii) RAC (rotate accumulator right with through carry)

(4) Branch Group I/Os

This group of I/s changed the normal sequence of the program, the branch group I/s are of 2 types—

- Conditional Branch Group I/s
- Unconditional Branch Group I/s

① Conditional → The branch I/s change the sequence of program when certain conditions are specified.

② Unconditional → The unconditional branch I/s change the sequence of program to a specified location unconditionally.

(i) JMP → Jump Unconditionally I/s

OP Code	Operand	Bytes
JMP	16-bit address	3 bytes

Eg.

Write an I/s at location JMP 2025 H to transfer the program sequence to memory location 2050 H.

Memory Address	Hex code	
2025	C3	JMP 2050
2026	50	Data OP code operand
2027	20	I/S

(ii) JUMP → Jump conditionally I/s

After the execution of a conditionally JUMP I/s, the program sequence is transfer to the I/s specified by the address.

The following are conditional JUMP I/s.-

DATE: / /20
PAGE NO.

	OP code	Operand	Flag Status	Description
(1)	JC	16-bit address	CY = 1	Jump if there's carry, the program jump to the I/S specified by addres
(2)	JNC	16-bit address	CY = 0	Jump if no carry, the program jump the I/S specified by address. If there is no carry, carry flag = 0.
(3)	JP	16-bit address	S = 0	Jump if result is plus, the program jump to the I/S specified by addres if result is +ve.

Stack

LIFO → last in first out principle followed

eg. of I/Ss → • PUSH B • POP B • LXI SP, 2050

→ PUSH B ⇒
Add the content of B-C register pair into the stack.

→ POP B ⇒ Remove the (top layer) content of stack and add to B-C pair register

→ LXI SP, 2050 ⇒ load Address 2050 into stack Pointer

PUSH Eg. → Consider the content of register B are 75 and of C are 3B. The stack is initialised with I/Ss LXI SP, 2699. This I/S will load memory address 2699 in the SP stack pointer. Now the I/S PUSH B will copy the content of register B in a memory location 2698 and content of register C in a memory location 2697. After this PUSH B I/S, the SP register is decremented by 2. Now, it contains the memory address 2696.

Def. ⇒ A stack in 8085 can be described as reserved area in the R/W memory where we can store temporary info. It is a shared resource as it can be shared by M.P & program.

1. Stack operated in LIFO principle.
2. The function of SP is to hold the starting address of the stack. An address can be decided by the programmer. SP points the top of stack.
3. Content can be stored using PUSH I/S & restored by POP I/S.

POP This is 1 byte I/S copy the content of top to memory location of the stack into specified register pair.
The content of memory location is indicate by the SP register are copied into low order register & then SP incremented by 1.

The content of next memory location are copied into the high order register and SP register. It again incremented by 1.

Eg: LXI SP, 2097
LXI H, 42F2
POP H
Delay Counter

DATE: / /20
PAGE NO.

Machine Control I/Os

(1) **IN :-**

I/P data to accumulator from a port with 8-bit address.

IN, 8-bit port address
Opcode operand

The content of I/P port whose address is specified (8-bit address) is read and copied into the accumulator. I/P and O/P devices connected 8085 M.P. have 8-bit port address.

(2) **OUT :-**

O/P data from accumulator to a port with 8-bit address.

OUT, 8-bit port address
Opcode operand

The content of accumulator are copied into the O/P port whose 8-bit address is specified in the I/O.

(3) **PUSH :-** PUSH register pair into stack
(4) **HLT :-** Halt (stop after execution)
(5) **POP :-** Pop ~~reg~~ stack's content to register
(6) **XTHL :-** Exchange the content of H-L with SP (Stack Pointer)
(7) **EI :-** Enable Interrupt
(8) **DI :-** Disable Interrupt

DATE : / /20
PAGE NO.

Subroutine

When the same function required more than 1 in a main program, it's frequently written as subroutine, i.e, it's a subprogram that can be used any number of times by the main program.

Subroutine are powerful programming constructed that allow a program to break down a complex task into smaller manageable pieces.

A subroutine is a block of code that can be called from any pair in the program and then return call back to the calling code when it's done.

The 8085 µP provide a set of I/S that can be used to implement subroutine.

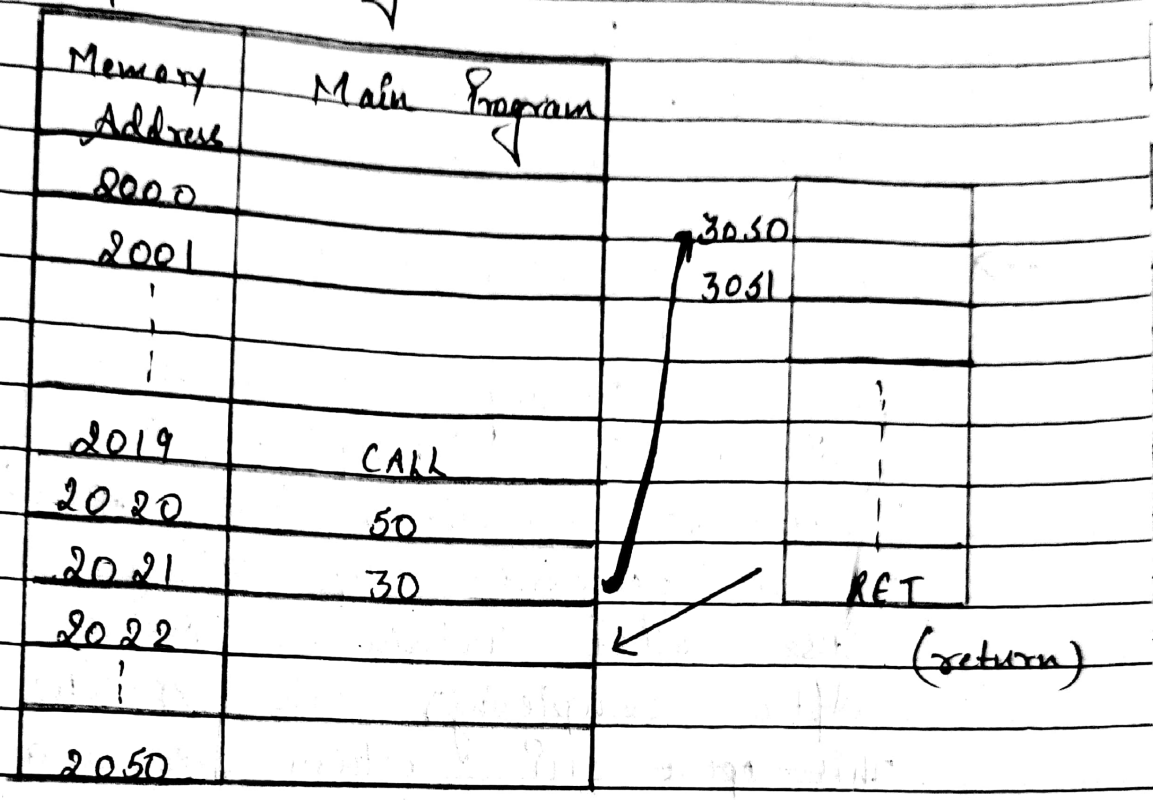
→ Call Instruction

This I/s is used to call subroutine. The current Program Counter is push into the stack & then PC is set to the address of first I/s in the subroutine.

→ Return Instruction

This I/s is used to return from a subroutine. Topmost bonus stack is

popped into the Program Counter which store the original program & continuous execution of calling code.



Interrupts

Interrupt is a subroutine call. An interrupt request is an asynchronous event that may occur at any time during a program execution. The CPU allows normal program execution to be interrupted in order to carry out a specific task. The CPU can be interrupted in following ways-

1. By an external signal generated by a peripheral.
2. By an internal signal generated by a special I/S in the program.

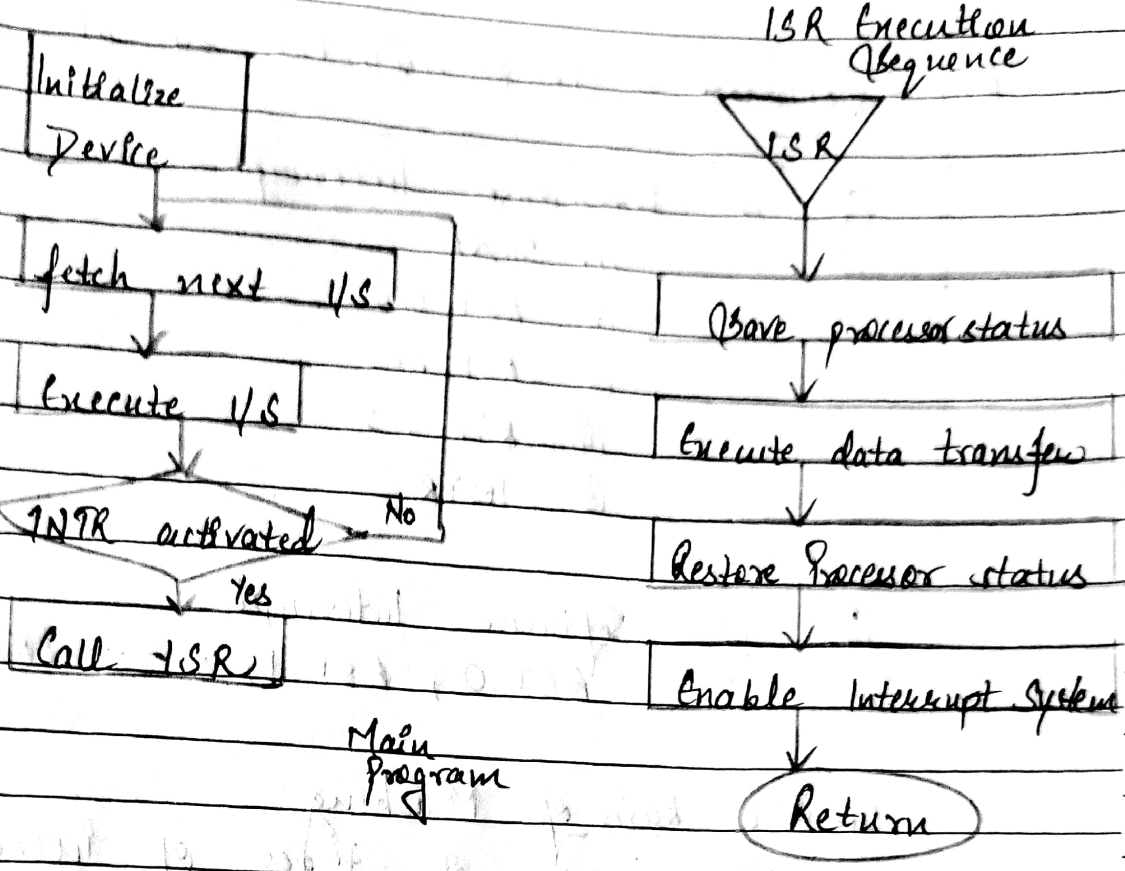
The external interrupt are used to implement interrupt driven data transfer schemes. The interrupt generated by a special IVs are called software interrupts. They are used to implement system services.

→ Sequence of Interrupt Operation

While executing the main program, the µP can be interrupted to perform some specific operations. These specific operations can be performed by calling a subroutine also called Interrupt Service Routine (ISR). After completing this specific task, ~~This opera~~ µP return to main program. The sequence of interrupt operation is explained as shown figure (flow of chart).

(2 diagrams)

→ Interrupt Classification



(3) Branch Addresses
- → Vectored (have address)
- → Non-vectored (no fixed location address)

(4) Various type of interrupt are used in a µP based system.

(1) On the basis of Initialization of types of Interrupt

a) Hardware → In this, the subroutine call is initiated by external devices or interrupt request is coming from external devices.

DATE: / /20
PAGE NO.

b) Software → It's initiated by μP itself.

Eg. of Hardware Interrupt
① TRAP
② RST 5.5
③ RST 6.5
④ RST 7.5
⑤ INTR

Eg. of Software Interrupt
RST 0, RST 1, ---- RST 7

(2) On the basis of Masking
There are 2 types of Interrupts

a) Maskable → Those interrupts which can be ignored & μP can enabled & disabled the interrupt
Eg. → RST 7.5, RST 6.5, RST 5.5, INTR

b) Non-maskable → Those interrupt which can not be ignored by the μP,
Eg. → TRAP

(3) On the basis of Branch Address
There are 2 types of interrupts

a) Vectored → In this, the branch address is supplied by the external devices. The interrupt which have fixed

memory location for transfer of control for normal execution.

Eg. →

TRAP	0024
RST 5.5	002C
RST 6.5	0034
RST 7.5	003C

Non-Vectored → In this branch has no fixed memory location for transfer of control.

Eg. → INTR

Assembly language Program

→ Machine language

A computer can understand information comprised of only zeroes and ones. It uses binary digits for its operation. A program written in the form of zeroes and ones in called a machine language programs.

The Data & instructions are stored in the memory of a computer in the form of zeroes and ones. In machine learning, there is a specific binary code for each instruction.

8085 µP has a word length of 8 bits. It can process 8 bit of data simultaneously

Program

Simple Data transfer

① place 07 in register C → MVI C, 07

② Place 07 in reg. A & then move it to reg. C
→ MVI A, 07
→ MOV C, A
→ HLT

③ Memory location 2050 contain 07. Place this content directly to accumulator & then move this data to register B.
→ LDA , 2050
→ MOV B, A
→ HLT

④ The content of memory location 2050 is 09H. move this content to Register B.
→ LXI H, 2050
→ MOV B, M
→ HLT

5-3-25

⑤ transfer the content of memory location 2050 to the register B & content of 2051 to register C. the content of memory location 2050 are 10H & of 2051 are 11H.
→ LXI H, 2050
→ MOV B, M
→ INX H
→ MOV C, M
→ HLT

DATE: / /20
PAGE NO.

⑥ Place 08 in accumulator, increment it by 1 and store the result in memory location 2050.

→ MVI A, 08
→ INR A
→ STA 2050
→ HLT

⑦ Addition of 2 8-bit numbers, sum is of 8-bits
Add 48H & 56H.

➡ let's, the first no. 48 is in memory location 2401 & the second no. 56 is in memory location 2402, the result should be stored in the memory location 2403.

→ LXI H, 2401
→ MOV A, M
→ INX H
~~→ MOV A, M~~ → ADD M
→ STA 2403
→ HLT

⑧ Subtract 33H from 48H. [8-bit subtraction]

➡ let's assume, the first no. 48H is in memory location 2401 & 33H in 2402. The result is stored in memory location 2403.

→ LXI H, 2401
→ MOV A, M
→ INX H
→ SUB M
→ STA 2403
→ HLT

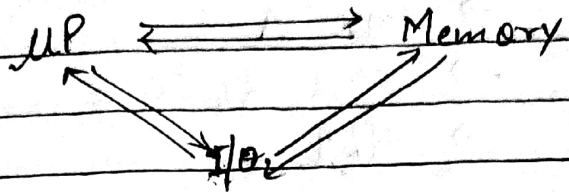
or

→ INX H
→ MOV M, A
→ HLT

DATE: / /20
PAGE NO.

12-3-25

* Memory and I/O device interface in Serial and Parallel communication interface



DATA TRANSFER SCHEMES

There are 2 methods used to transmit data b/w digital device -

① Parallel Data Transmission
② Serial Data Transmission

In µP data transfer can take place b/w the µP & I/O devices, memory & I/O devices and µP & memory. These I/O devices are manufactured by using different manufacturing techniques like electrical, electronics, electromachanical, machanical, optical etc.

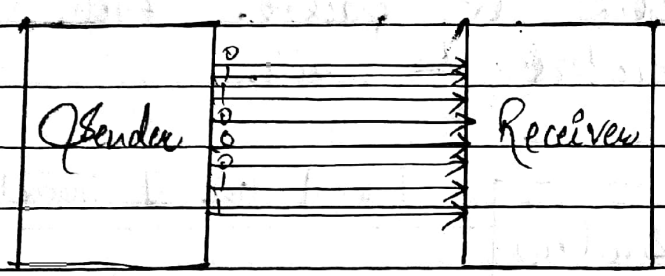
Due to using different manufacturing techniques, these I/O devices has different characteristics & special interfacing ckts are required to interface I/O with µP.

The data transfer is preferred only when small amount of data are transferred by these schemes. The schemes are following - ① Parallel data transmission
② Serial data transmission

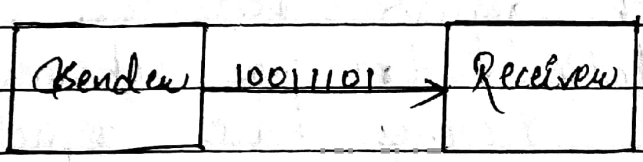
DATE: / /20
PAGE NO.

① Parallel data transmission

In this, various databits are simultaneously transmitted using multiple communication links b/w sender & receiver. The data is transmitted using parallel data transmission over the multiple channels at the same time. This means, data can be send much faster than using serial data transmission.



② Serial data transmission



In this, the data bits are transmitted serially over a common communication link, one after the other. ① It is normally used for long distance data transfer.

② It's also used in case where the amount of data being send relatively small.

③ It ensures that data integrity is maintained as it transmitted the data bit.

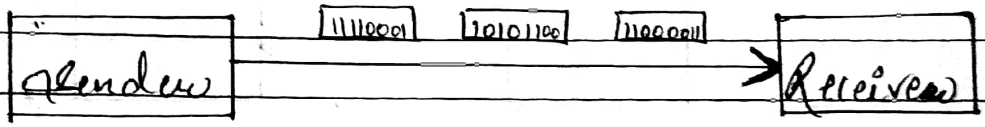
In this way, data bits are received asynchronous to one another.

④ Using a single wire, reduces cost but slows down the speed of transmission.

2 types → ① Synchronous data transmission
② Asynchronous data transmission

② a) Synchronous data transfer transmission

In this, a lot of data is sent in a block or packet. Each block has many characters & bits.



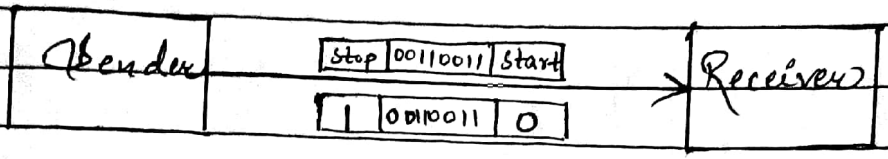
the word synchronisation means occurence at regular intervals. In this type of data transfer schemes, the transmitter & receiver are synchronised with same clock. It's simplest scheme from all data transfer schemes

b) Asynchronous data transfer transmission

In this, only 1 character is sent at a time whether that a character is no. or alphabet. It uses start & stop bits for transfering data. The word asynchronous means occurence at the irregular intervals.

DATE: / /20
PAGE NO.

It's low speed data transfer technique. Data transfer b/w the µP & peripheral is generally asynchronous bz the speed of I/O devices doesn't match the speed of µPs.



Start bit = 0
Stop bit = 1

17-3-25

Hardware Description of Registers

A, B-C, D-E, H-L, PC, SP, Temp. Register, Flag R., I/O R.

Memory Management

→ Primary
→ Secondary

Memory address and mapped

8085 µP → 16 Address lines
→ 2^{16} = 64K B
↳ bytes → 8 bit

65,536 registers.

Min. address location → 0000H
Max. address location → FFFFH

→ Memory mapped I/O & I/O mapped schemes

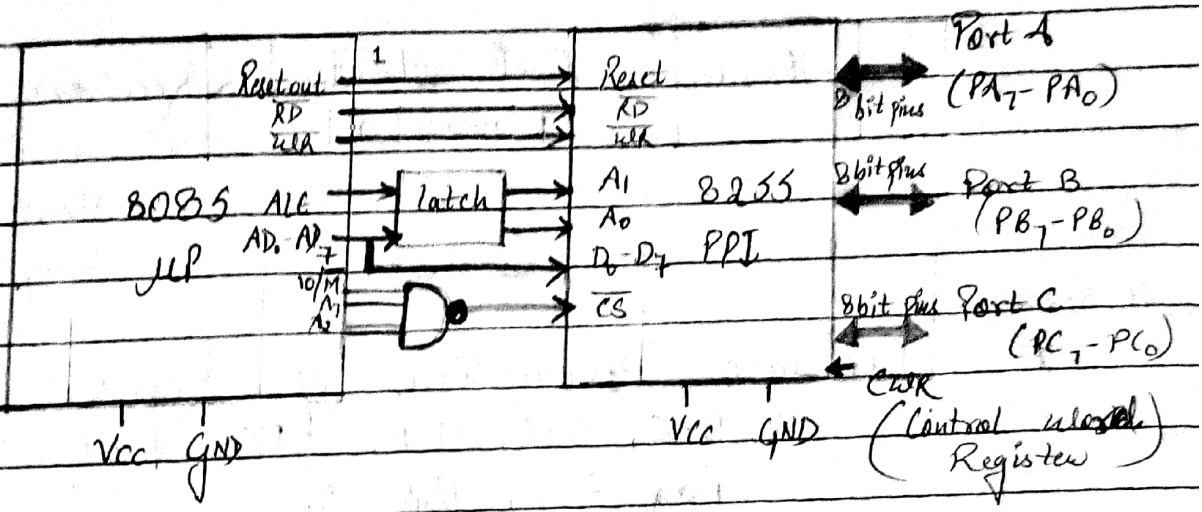
→ Memory interfacing ← RAM
ROM - EPROM

→ 8255 port
(Programmable peripheral interface)

Memory chip = 256 bytes

$= 2^8$ → 8 address lines

8085 and 8255 interfacing



A_1	A_0	Operation
0	0	Select port A
0	1	B
1	0	C
1	1	CWR

CWR → Control word register is a 8 bit Register which control the 8255 internal operations & also control port direction.

ALE = 1 → address lines

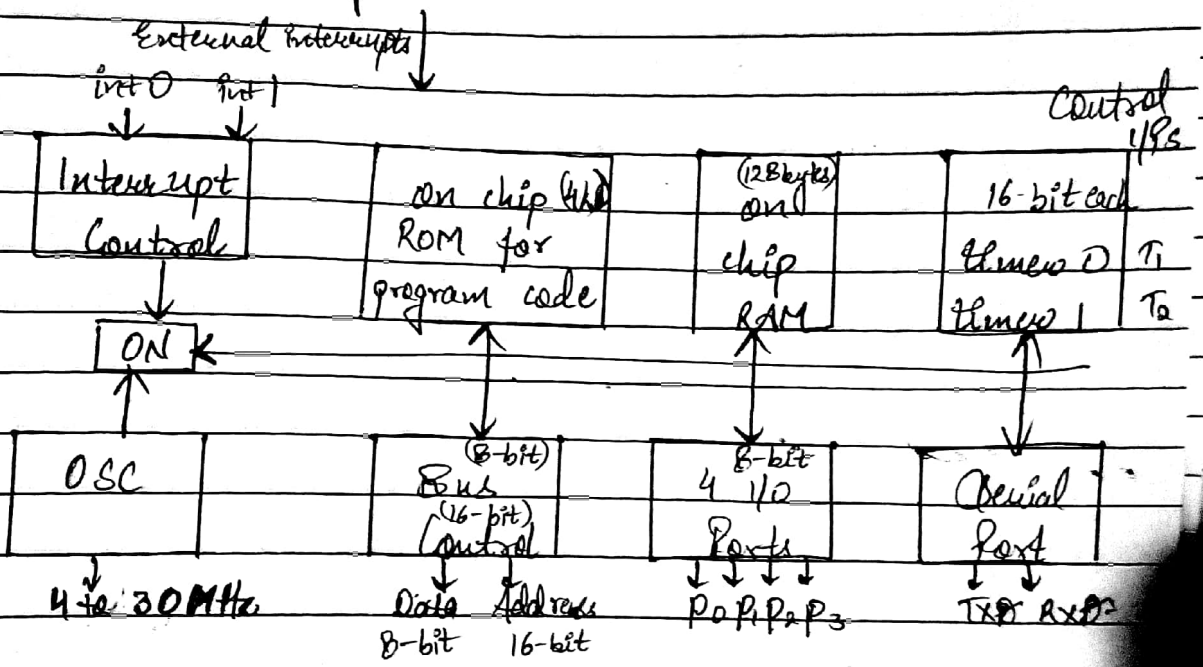
ALE = 0 → data lines

27-3-25

DATE: / /20
PAGE NO.

* Microcontroller 8051

→ Difference b/w M.P. & M.C.
→ RISC and CISC
→ Architecture of 8051



→ 32 general purpose registers (8-bit each)
→ 6 interrupts (3 internal, 2 external, Reset)
→ 16 bit PC & DPTR 16-bit
→ 8 bit SP