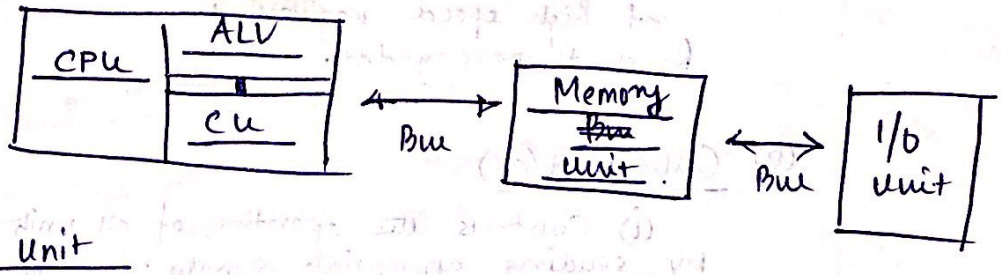
06/01/2020

Microprocessor & Assembly Language Programming

Prof. Jamuna Kanta Sing

Slide -1

Basic Functional Unit of a ~~Cpu~~ Computer System



I/O Unit

Allow the computer system to interact with the outside world by moving data into & out of the system.

- Input Unit : Accept coded information as data
- Output Unit : Sends processed result to the outside world.

Slide -2

Memory Unit :- Stores programmed data

Main Memory	Secondary Memory
(i) Fast (Lowers Memory Access speeds).	(i) Slower
(ii) Expensive	(ii) Cheaper
(iii) Low Capacity	(iii) Large Capacity
(iv) Connects directly to the CPU	(iv) Not connected directly to the CPU
eg. RAM	eg. HDD, Floppy Disk, CD-ROM, etc.

* Program must reside during execution.

Principle of Locality

~~Step-3~~ Slide-3

Central Processing Unit (CPU)

(a) Arithmetic & Logical Unit (ALU)

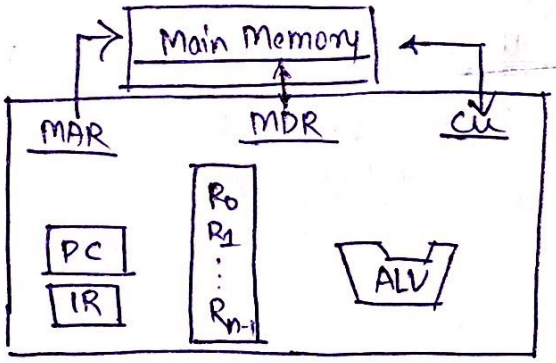
(i) Dedicated unit to perform arithmetic & logical operations

(ii) Processor holds a number of high speed registers to hold temporary data

(b) Control Unit (CU)

(i) Controls the operation of all units by sending appropriate signals to them

Slide-4



Interconnection b/w memory & CPU

8085 → 8-bit registers.
ADDition, Subtraction, Increment, Decrement

PC → holds the address of the inst. to be executed next

Slide - 5

MAR, MDR, PC, IR

↓

Holds the instruction that is executed.

Slide-6

How does a Microprocessor work?

(1) The Instructions are stored ~~to~~ sequentially in memory.
(2) The Microprocessor fetches one instruction at a time, decodes and executes.
(3) The sequence of fetch, decode and execute is continued ~~till~~ ~~the~~ until the microprocessor comes across an instruction to stop.
(4) During the entire process, microprocessor uses the system bus to fetch instruction & data from the memory.
(5) It uses registers within CPU to hold data temporarily and performs arithmetic & logic ~~in~~ using ALU.

09/01/2020 ~~Hardware~~ & Microprocessor

Prof. Jamuna Kanta Sing

Slide - 1

Classification of Microprocessor.
Based on word length.

Word:- It is defined as the number of bits the microprocessor recognizes and process at one time.
It ranges from 4-bit to 64-bit.

4-bit → Intel 4004
8-bit → Intel 8008, 8080, 8085, Motorola 6800 Zilog Z80
32-bit → Intel 8086, 8088
64-bit → Intel Pentium, Pentium II, Pentium-4

Slide - 2 Microprocessor

→ A programmable to device to control ~~processor~~ process / devices.
→ A processing unit of computer
→ A CPU or microprocessor based system

A microprocessor operates on bit 0 & 1.
4 bit string ⟶ Nibble
8 bit string ⟶ Byte
One or more byte ⟶ Word.

Slide 3

Pin Diagram of 8085

$(AD_0 - AD_7) + (A_8 - A_{15}) \longrightarrow$ 16-bit address bus.

lower byte (LHS)
higher byte (RHS)

Sometimes used address/data.

RST 7.5
RST 6.5
RST 5.5
TRAP
INTR
} Interrupt

→ complement

$\bar{x}$ ⟶ active-low signal.

x ⟶ active-high signal.

HLDA → DMA Controller

IO/$\bar{M}$ → I/O operation

$\overline{RD}$, $\overline{WR}$

↓ Read ↓ Write

ALE → Address Latch Enable.

Slide-4

FA Block diagram/architecture of 8085

Slide-5

8085 Bus

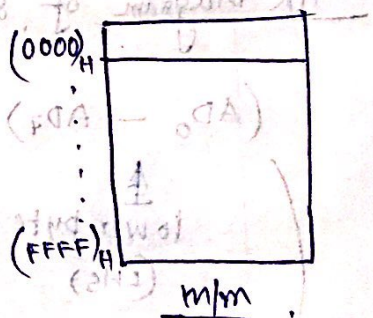
Address Bus.

→ The address bus is a group of 16 lines generally identified as A0 to A15.

→ The address bus is unidirectional : bits flow in one-direction from the MPU to peripheral devices

→ The MPU uses the address bus to perform the first function : identifying a peripheral/memory location.

Main Memory



8085 MPU Bus Structure (Diagram).

Slide-6

Data Bus

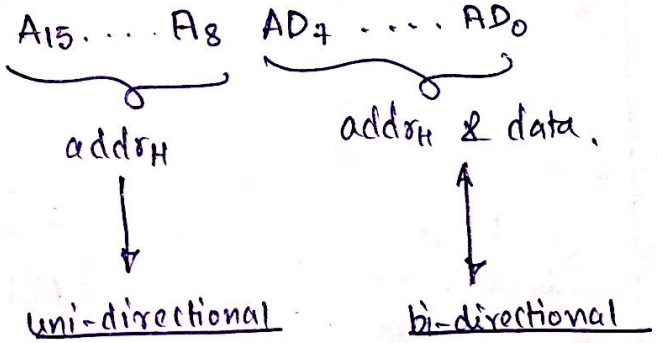
→ The data bus is a group of 8 lines used for data flow.

→ These lines are bi-directional - data flow in both directions b/w the MPU & memory & peripheral devices.

→ The MPU uses the data bus to perform the second th, transferring the binary information

→ The 8 data lines enable the MPU to manipulate 8-bit data ranging from 00 to FF. ($2^8 = 256$ no).

→ The largest no. that can appear in 1111 1111.



To prevent overwrite of ——, ALE is used (during reading/writing operation)

16/01/2020

Slide-1

Microprocessor

Prof. Yamuna Kanta Sing

Control Bus

→ The control bus carries synchronization signals and provide timing signals.

→ The MPU generates specific control signals for every operation it perform. These signals are used to identify a device type with which the MPU wants to communicate.

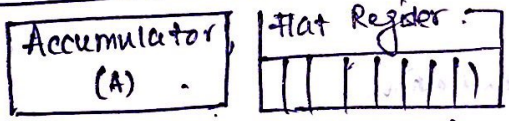
Registers of 8085

→ The 8085 has 6 general-purpose registers to store 8-bit data during program execution.

→ These registers are identified as B, C, D, E, H & L.

→ They can be combined as register pairs BC, DE & HL to perform some 16-bit operations.

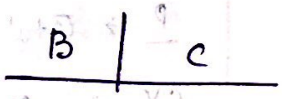
Slide-2



B (8) C (8)
D (8) E (8)
H (8) L (8)

Stack Pointer (16)

Program Counter (16)



Data Bus

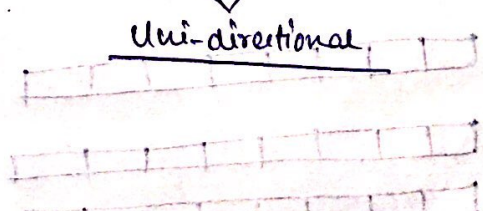
8-lines

Bi-directional

Address Bus

16 lines

Uni-directional



Slide 3.

Flags

→ The ALU includes five flip-flops that are set/reset according to the result of an operation.

→ The microprocessor uses the flags for testing the data conditions.

→ They are zero (Z), carry (CY), sign (S), parity (P), and auxiliary carry (AC) flags. The most commonly used flags are sign, zero & carry.

Bit pos^n.

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

S = 1 → Negative

Z = 1 → Content is zero

AC → auxiliary carry. (if 4th bit pos^n generate carry).
generate carry while add/subtraction

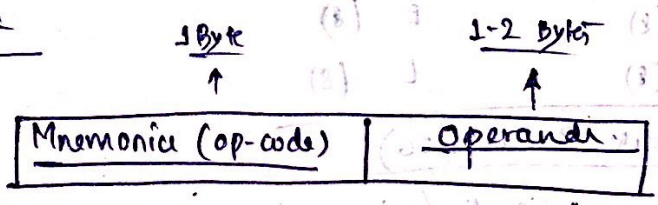
P → Total no. of ones

CY → Carry from add/subtraction

Slide 4

Instruction set

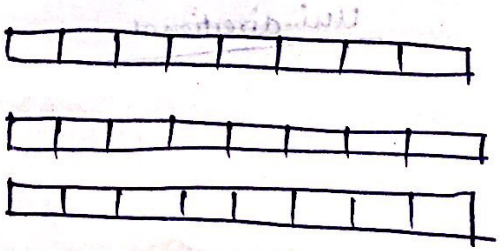
Format :-



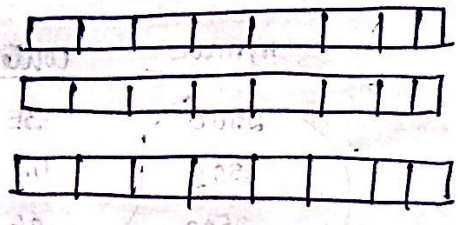
Types of Instruction based on: Add, subtract-

Size: 1 Byte, 2Byte, 3Byte

Instruction based on size



3 Byte



Slide - 4

Address	Content	Inst. Type
0000_H	...	
$000i_H$	op code	1 byte Inst.
	...	
$000j_H$	op-code	2 byte instruction
$000j+1_H$	operand	
	...	
$000K_H$	op-code	3 byte instruction
$000K+1_H$	operand$_L$	
$000K+2_H$	operand$_H$	

Slide - 5

Programming in 8085.

Add 2 number 05 & 03.

```
MVI   A, 05H    ; A=05
MVI   B, 03H    ; B=03
ADD   B         ; A = A+B
STA   2050H     ; M[2050] = A
RST 5
```

Comment.

Store Accumulator.

end.

Restart.

(4)

m/mloc	content	
2500	3E	→ MVI A, 05_H
2501	05	
2502	06	→ MVI B, 03_H
2503	03	
2504	80	→ ADD B
2505	32	→ STA 2050 (H L)
2506	50	
2507	20	
2508	EF	→ RST 5

Addressing Modes.

It refers to the way in which operands are specified in an instruction. In other words, it refers to the way by which processor accesses the operand.

1. Immediate Addressing
2. Direct Addressing
3. Register Addressing
4. Register Indirect Addressing
5. Implied Addressing.

opcode	operand

Immediate Addressing

→ Data is specified in the instruction itself. The data will be part of program instruction.

Examples.

MVI Reg $Data_8$, Reg = {A, B, C, D, E, H, L}.

MVI B ; 3EH ; B = 3EH

LXI RegPair $Data_{16}$, RegPair = {B, D, H, SP} (B: BC, D: DE, H: HL)

LXI SP, 2700H ; SP = 2700H, SPL = 00H, SP_H = 27H.

MVI → Move Immediate

LXI → Load Register Pair Immediate

Direct Addressing

→ In direct addressing, the address of the data is specified in the instruction. The data will be in memory. In this addressing mode, the program instructions and data stored in different memory.

Example.

LDA Address16 ; → Load in Accumulator.

LDA 1050H ; A = M[1050H].

STA Address16 ; → Store In Accumulator.

LHLD 3000H ; L = M[3000H]; H = M[3001H].

→ Load HL direct

Register Addressing

In register addressing mode, the instruction specifies the name of the register in which the data is available.

Example. MOV RegD, RegS ;

RegD, RegS = {A, B, C, D, E, H, L}

MOV A, B ; A=B
ADD C ; A=A+C

Register Indirect Addressing

The instruction specifies the name of the register in which the address of the data is available. Here the data will be in memory & the address will be in the register pair.

Example MOV A, M ; A = M[HL] ;
LDAX B ; A = M[BC] ;

LDAX → Load Accumulator from the Pair.

Implied Addressing

The instruction itself specifies the data to be operated. Example CMA : Complement the content of Accumulator
RAL → Rotate Acc left by 1 bit.

Instructions

→ Data Transfer Instruction.

It includes the instructions that move (copies) data b/w registers or between memory locations & register. In all data transfer operations the content of source register is not altered. Hence the data transfer is copying operation.

Example.

① MOV A, B ; A=B
② MVI C, 45H ; C=45
③ LXI H, 2005H ; H=20, L=05
④ ~~MOV A ; M=A ;~~ MOV A, M ; A=M[HL]
⑤ LDA 2050H ; A = M[2050H]
⑥ STA 2010H ; M[2010H] = A.

Arithmetic Instruction

Includes the instruction, which performs the addn & subtraction, increment / decrement operations. The flag conditions are altered after ~~instruction~~ execution of an instruction in this group.

Eg.
```
ADD B ;  A = A+B
ADC B ;  A = A+B+CY ;
ADD M ;  A = A+M[HL] ;
ADC M ;  A = A+M[HL]+CY .
ADI 05H ;  A = A+05H
SUB B ;  A = A-B
SUB M ;  A = A-M[HL]
SUI 05H ;  A = A-05H
```
Increment ← (INR) B ; B = B+1
INX B ; BC = BC+1
Decrement ← (DCR) B ; B = B-1
DCX B ; BC = BC-1

Logical Instructions

The instructions which performs the logical operations like AND, OR, EXCLUSIVE OR, complement, compare & rotate instructions are grouped under this heading. The flag condn are altered after execution of an instruction in this group.

Eg. (ORA) A → OR
(ANI) B, 01H → AND
masking of digits

Branching Instructions

The instructions that are used to transfer the program control from one memory location to another memory location are grouped under this heading.

Eg.
1. CALL 2050H (Subroutine call)
2. JMP 4100H (Unconditional jump)
3. JNZ 2040H (Conditional jump).

Machine Control Instruction

It includes the instructions related to interrupts & the instruction used to stop the programe execution.

Ex. ① NOP
② HLT
③ RST 1.

Stack Instruction

→ Used for stack operation

Eg. PUSH B; M[SP--] ; M[SP--] = C;
POP B; C = M[++SP]; B = M[++SP];

IO Instruction

Used for I/O instruction.

① IN 45H
② OUT 55H.

03/02/2020

Microprocessor & Assembly Language Programming

Prof. Tamuna Panda Sir

→ Program must reside in Main Memory

Instruction Execution

Each instruction is executed in 2 cycles :-
(i) Fetch Cycle (ii) Execution Cycle

(i) Fetch Cycle

(a). Instruction addressed by the content of Program counter (PC) is loaded in instruction register (IR) from memory. MAR = [PC], MDR = M[MAR], IR = [MDR].

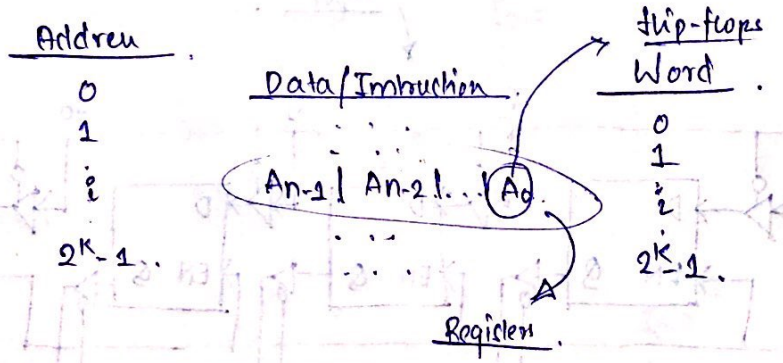
(b) PC is incremented. PC = PC + 1.

(c) Fetching of operand data initiated.

(ii) Execution Cycle

(a) Instruction in IR is executed & specified operation is performed by CPU.

Memory Location & Address

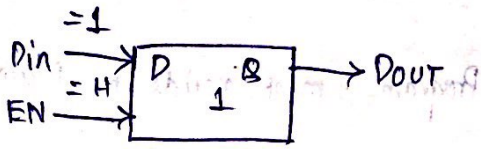


Memory

① Read/Write (R/W) memory
→ Made of Register.
→ Register made of flipflops (F/F).

② Read Only Memory
→ Made of Register
→ Register made of diode.

D-flipflop :



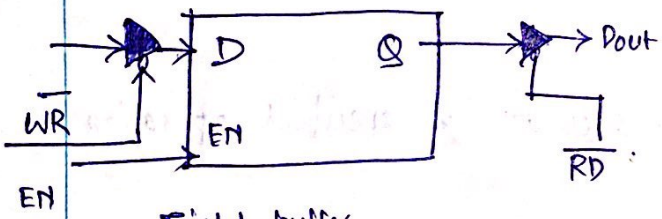
F/F or Latch .

→ To prevent unintentional change of input (i/p) & control of read operation, the F/F may be modified using a tri-state buffer.

To prevent .

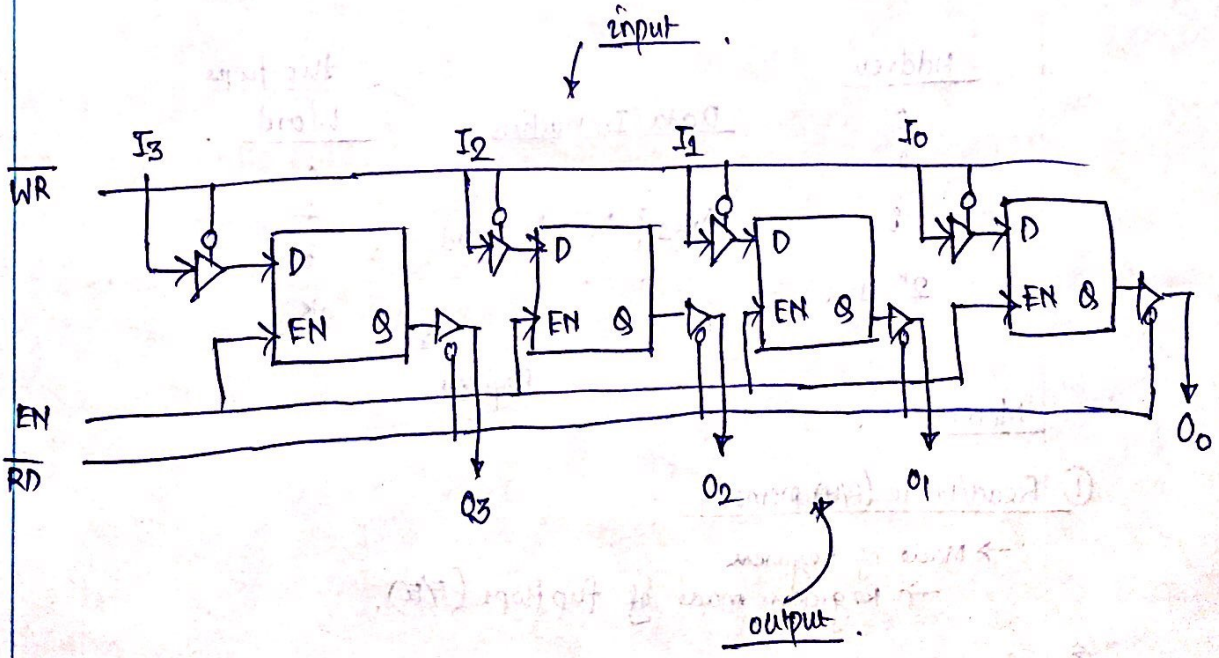
Din & Dout can work simultaneously

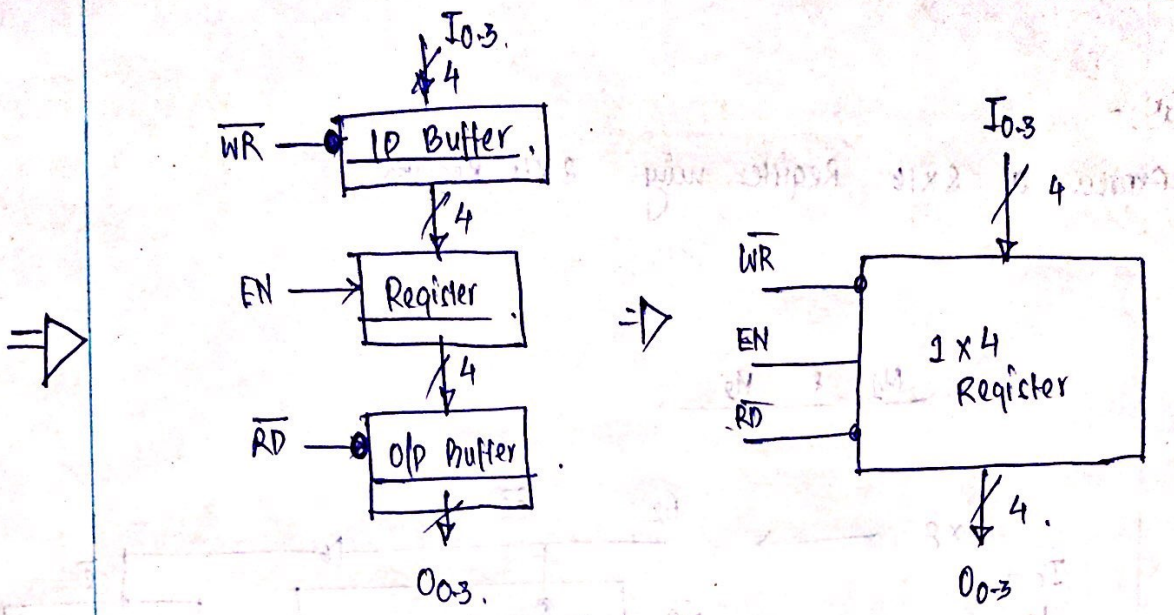
No mechanism to prevent such operations.



Tristate-buffer .

→ Controlling Read & Write signals to prevent error .





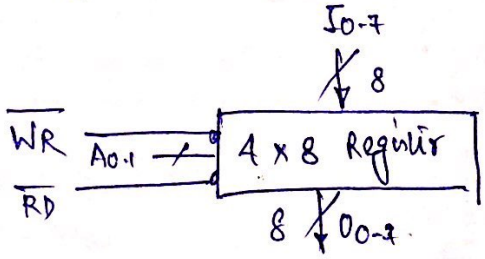
Memory Organization

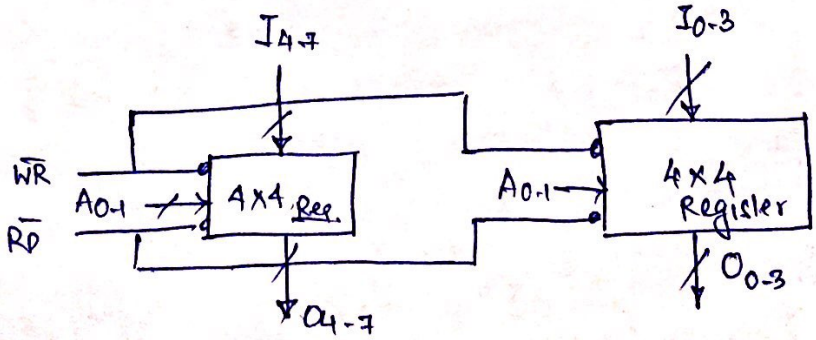
In a memory (m/m) chip all the registers are arranged in a sequence & are identified by a binary number called the m/m address.

Communication with memory

1) Select the memory chip.
2) ~~Identified~~ Identify the required register.
3) ~~Performed~~ the Read & Write operation.

Perform



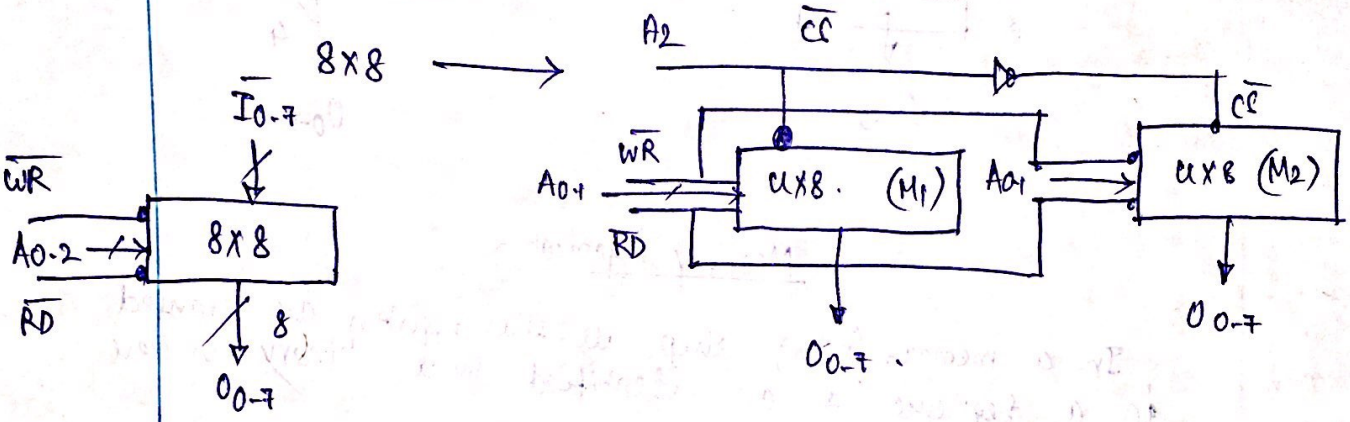


Made up with 2 chips with two 4x4 registers

Task:-

Express a 8×16 Register using 8×4 Register.

M_1 & M_2.



A_2 acts as a chip select.

Made up with 2 nos chips

Task:-

Express a 16×16 Register using 4×16 Register.

06/02/2020

~~Com~~ Microprocessor & Assembly Language Proceing

Prof. Jamuna Kanta Sing

Given a memory chip:-

addr :- 16 bits

memory locations :- 2^{16}.

0000_H ⇓ $FFFF_H$.

① How is memory mapped & address generated?

Soln. 1) Select the no. of address lines required to identify one register of the memory module. Assign the address lines starting from A_{0-i}.

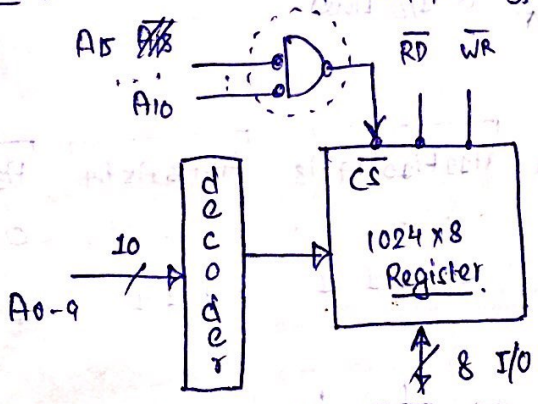
2) Use the remaining address lines $A_{(i+1)-15}$ to generate a unique chip select (CS) signal.

3) Starting address is $A_{15} A_{14} A_{13} A_{12} \ldots A_{i+2} 000 \ldots 0$

Ending address is $A_K \ldots \ldots A_{i+1} 11 \ldots 1$.

Example. Memory address of 1K (1024×8) memory.

↳ Config. is diff. for diff chips.



→ → Complemented (Active Low)

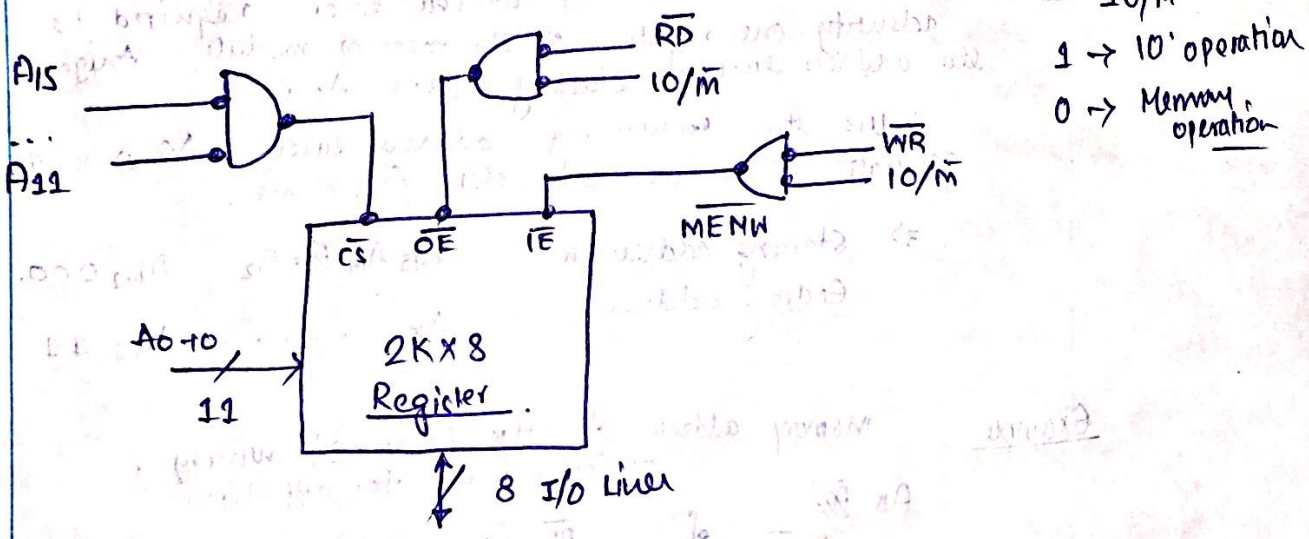
	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
															0	0
															1	1
0000H =	0															
03FFH	0															

• By changing the combination of A_{10}-A_{15}, different memory address range can be formed.

Memory Interfacing

1) Connect the required address lines of the address bus to the address lines of the memory chip.
2) Decode the remaining address lines of the address bus to generate a unique chip select signal
3) Generate control signals $\overline{MEMR}$ and/or $\overline{MEMW}$ by combining $\overline{RD}$ & $\overline{WR}$ signals with IO/$\overline{M}$ signal, respectively. Use them to enable appropriate ~~input~~ input and/or output buffer.

Example:- Interfacing 2K Read/Write Memory

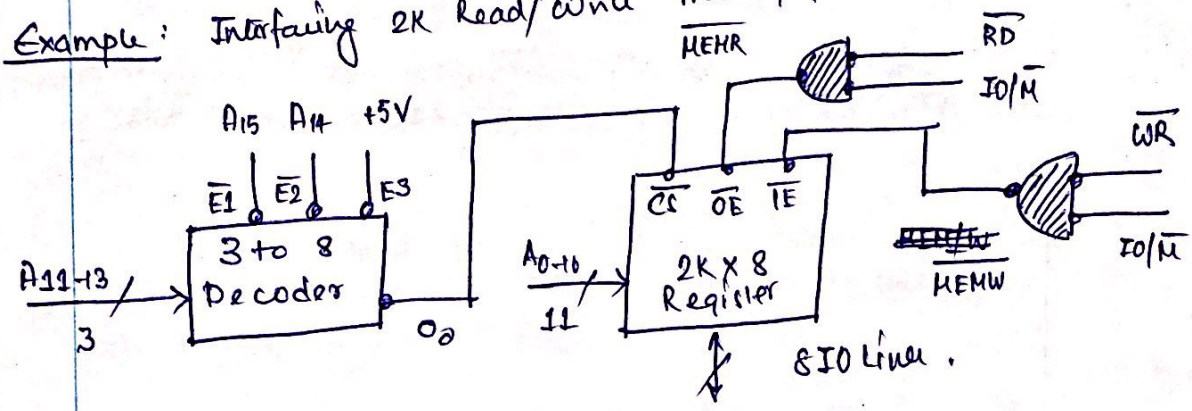


IO/$\overline{M}$
1 → IO operation
0 → Memory operation

		$\overline{A_{15}}$ $\overline{A_{14}}$ $\overline{A_{13}}$ $\overline{A_{12}}$	$\overline{A_{11}}$ $\overline{A_{10}}$ $\overline{A_9}$ $\overline{A_8}$	$\overline{A_7}$ $\overline{A_6}$ $\overline{A_5}$ $\overline{A_4}$	$\overline{A_3}$ $\overline{A_2}$ $\overline{A_1}$ $\overline{A_0}$
0000H	=	0 0 0 0	. . . .	. . . .	0 0 0 0
07FFH	=	0 . . .	0 1 1 1	1 1 1 1	1 1 1 1

- Limitation due to ~~fan~~ fan-in of the NAND gate to decode address.

Example: Interfacing 2K Read/Write memory.



A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Initial of NAND gate use a decoder to decode address.
0000H = 0	0	0	.	.	.	.	.	.	.	.	.	.	.	.	.	0
07FFH = 0	.	.	.	0	1	.	.	.	.	.	.	.	.	.	.	1

A13 A12 A11
0 0 1 => 01

A15 A14
X X ← 4 combination

Because by fixing one, we can have a primary memory while other will be foldback memories.

Address Range.

0800H
0FFFH

Address Decoding.

→ Complete/Absolute Decoding :- All the available address lines are used for address decoding.

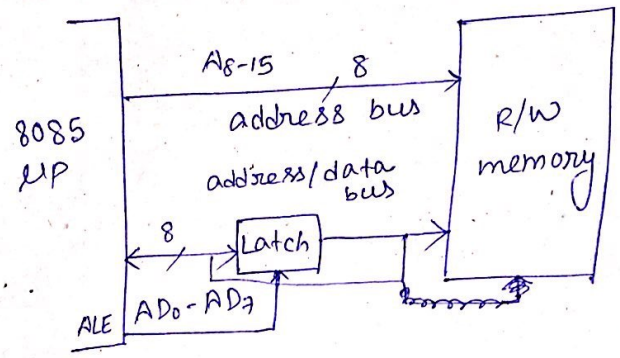
→ Partial Decoding :- Not all the available address lines are used for address decoding: others are kept as don't care condition.

Foldback/Mirror Memory :- For each combination of don't care address lines, we have seperate address range. Keeping one combination as primary address range, remaining address ranges are called foldback (or mirror memory of the primary address range.

⊛ Subroutine Call
Nested Call

Functions performed by the Microprocessor →

1) Microprocessor initiated
2) Internal Operations
3) External initiated (peripheral Operations)



1) Microprocessor initiated:

a) Memory read
b) memory write
c) I/O read
d) I/O write

☞ To perform above operations microprocessor does following steps-

1) Identify the peripheral /memory
2) Provide synchronization signals
3) Transfer data

MPU performs Read Operation when:
1) Fetches Op code of an instruction from m/m
2) Fetches the operand address portion of an instruction from m/m.

3) Fetches data from m/m or i/o device to execute an instruction.

MPU performs Write Operation when:

1) Transfers data from internal MPU register to m/m.

2) Transfers data from internal MPU register to an i/o device.

(*) Timing Diagram is a graphical representation. It represents the execution time taken by each instruction in a graphical format. The execution time is represented in T states.

Instruction Cycle → The time required to execute an instruction is called instruction cycle. It consist (fetch cycle + execution cycle)

Machine Cycle → The time required to access the memory or input/output devices is called machine cycle.

T state (clock cycle) → The machine cycle and instruction cycle takes multiple clock cycles (periods). A portion of an operation carried out in one system clock period is called as T-state.

Machine cycles of 8085 →

The 8085 microprocessor has 5 basic machine cycles. They are →

Opcode fetch cycle (4 T state)

Memory read cycle (3T)

Memory write cycle (3T)

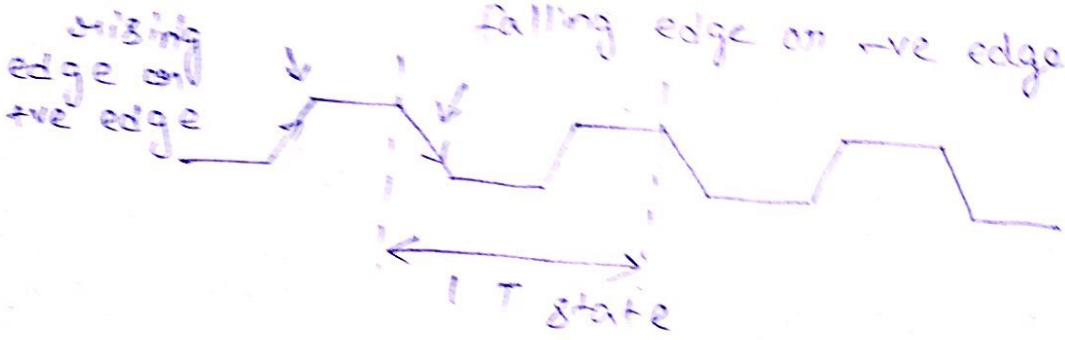
I/O read cycle (3T)

I/O write cycle (3T)

Note

Time period; $T = \frac{1}{f}$; where f = Internal clock frequency

Falling edge or -ve edge.



17/02/2020

Machine Cycles of 8085

Microprocessor & Assembly Language Programming

Prof Yamuna Kanta Sing

→ Each instruction of the 8085 processor consists of one ~~to~~ to five machine cycles i.e. when the 8085 processor executes an instruction, it will execute some of the machine cycles in a specific order.

→ The processor takes a definite time to execute the machine cycle. The time taken by the processor to execute a machine cycle is expressed in T-states.

→ One T-state is equal to the time period of the internal clock signal of the processor.

→ The T-state starts at the falling edge of a clock.

The 8085 microprocessor has 5 (five basic) machine cycles:-

① Opcode Fetch machine Cycle

→ Each instruction of the processor has one byte opcode.

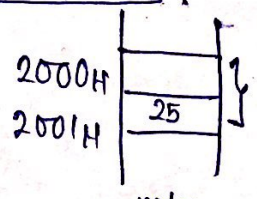
→ The opcodes are stored in memory. So, the processor executes the opcode fetch machine cycle to fetch the opcode from memory.

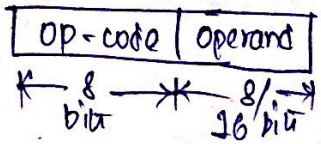
→ Hence, every instruction starts with opcode fetch machine cycle.

→ The time taken by the processor to execute the opcode fetch cycle is 4T.

→ In this time, the first 3 T-states are used for fetching the opcode from memory & the remaining T-states are used for internal operations by the processor.

MVI A, 25H





optional.

m/m → data bus → MAR → IR.

Eg. Timing Diagram of Opcode fetch cycle.

② Memory Read Machine Cycle of 8085

→ The memory read machine cycle is executed by the processor to read a data byte from memory

→ The processor takes 3T state to execute this cycle.

→ The instructions which have more than one byte word size will use this machine cycle after the opcode fetch machine cycle.

Eg. Timing diagram of memory read cycle.

③ Memory Write Cycle of 8085

→ The memory write machine cycle is executed by the processor to write a data byte in a memory location.

→ The processor takes 3T state to execute the machine cycle.

→ Timing Diagram for MVI B, 43H.

Complete execution cycle diagram.

→ Timing Diagram for INR M

① O/P fetch
② m/m read
③ m/m write

→ Timing Diagram for STA 526AH

Microprocessor

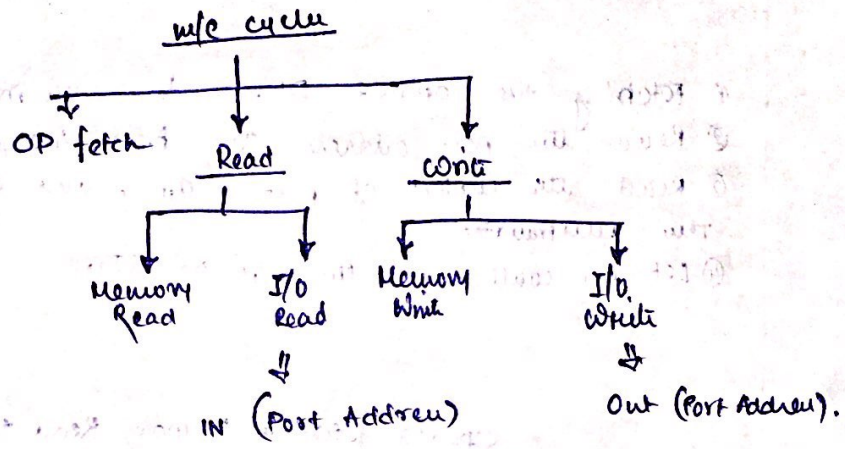
Prof. Tamuna Kanta Sing

I/O Read Cycle of 8085

→ The I/O Read cycle is executed by the processor to read data byte from I/O port or from ~~their~~ the peripheral which is I/O mapped in the system.

→ The processor takes 3T state to execute this machine machine cycle

→ The IN instruction uses this machine cycle during its execution.



Please see :- Timing Diagram of I/O Read Cycle.

I/O Write Cycle of 8085

→ The I/O write machine cycle is executed by the processor to write a data byte in the I/O port/peripheral, which I/O mapped in system.

→ The processor takes 3T state to execute this machine cycle.

→ The OUT instruction uses the machine cycle during execution.

Please see : Timing Diagram of I/O Write Cycle

Address	Mnemonics	Opcode
4125	IN CO_H	DB_H
4126		CO_H

Opcode fetch

4T + 3T + 3T.

I/O Read

Memory Read.

① Fetching the opcode DB_H, from the memory 4125_H.
② Read the port address CO_H from 4126_H
③ Read the content of port CO_H, and send it to the accumulator
④ Let the content of the port be $5E_H$

Opcode fetch + Memory Read + I/O Read.

Time Delay

→ Sometimes there is a need to perform tasks after some specific delay. The amount of delay can be realized by writing a delay program.

```
        MVI  C   XX_H      ——→  4
loop:   DCR  C              ——→  4
        JNZ  Loop           ——→  10.
```

Time Delay for T loop: $\{(\text{No. of states}) * T\_Time * nn\} - 3 * T\_Time$.

$= \{(4+10) * P * nn\} - 3P$

$= (14P * nn) - 3P$.

Total $= T + 7P$

$= (14P * nn) - 3P + 7P$

$= (14P * nn) + 4P$.

$= 5s$ for example

Example

let CPU clock speed : 3MHz.

T. Time (P) = $\frac{1}{3} \times 10^{-6}$ seconds.