unit-1

8086 Architecture

Microprocessor : microprocessor is a multipurpose, programmable device that accepts digital data as input processes it according to instructions, and stores in its memory, provides result as output.

Microcontroller : Microcontroller is a small computer on a single integrated circuit containing a processor core, memory and programmable input/output peripherals.

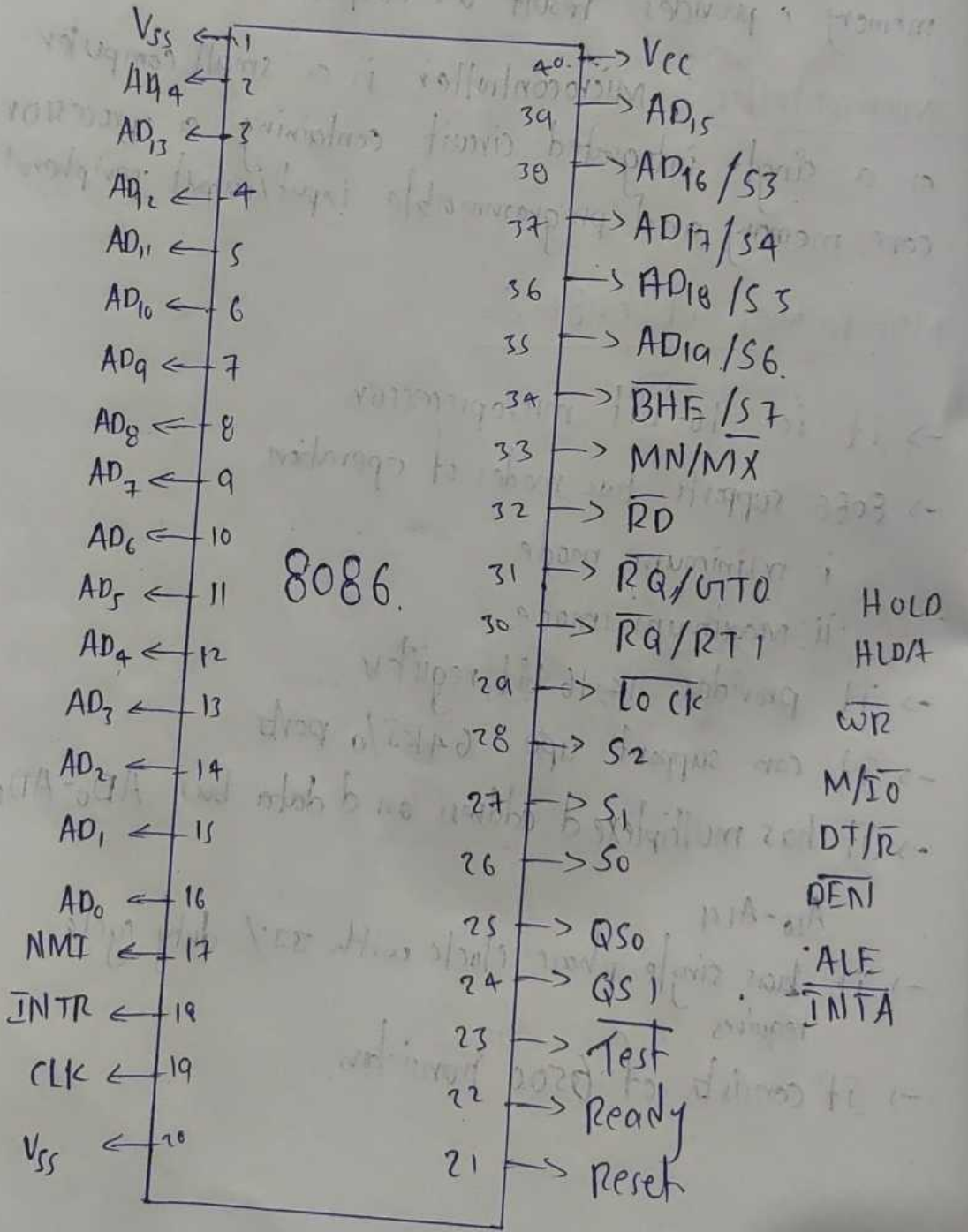
Main features of 8086 :

-> It is 16-bit microprocessor.
-> 8086 supports two modes of operation.
   i minimum mode.
   ii maximum mode.
-> It provides 14, 16 bit registers
-> It can support upto 64k I/O ports
-> It has multiplexed address and data bus AD_0-AD_{15} & A_{16}-A_{19}.
-> It has single phase clock with 33% duty cycle.
-> It consists of 6500 transistors.

Pin diagram and Description of 8086

-> An 8086 microprocessor is also a 40 pin IC. but have few separate pins for minimum and maximum mode.

-> The below figure represents pin diagram of 8086.



Description of 8086

power supply and clock signal

V_{CC} -> 5V DC supply at pin 40, ~~GND[Vss]~~ =>
GND [V_{SS}] -s ground at pin 1 and 20.
clock signal [CLK] -s clock signal at pin 19.
clock signal frequency 5MHz-10MHz

Address / Data bus [AD_0 - AD_{15}]

-> 16 address liner and data lines are multiplexed. to minimize the number of pins in IC.

AD_0 - AD_{14} -> pins 2-16.
AD_{15} at pin 39.

Address / status Bus :

AD_{16}/S_3 , AD_{17}/S_4 , AD_{18}/S_5 , AD_{19}/S_6.

pin 38 , 37 , 36 , 35.

S_3, S_4 flags are used to select segment registers

S_3	S_4	
0	0	-> ES
0	1	-> SS
1	0	-> CS
1	1	-> DS

$\overline{BHE}$ / S_7 :

-> $\overline{BHE}$ / S_7 is at pin 34.
-> $\overline{BHE}$ - Bus High enable.
-> $\overline{BHE}$ - Active low signal.

BHE	S_1	
0	0	→ All 16 bits will be accessed
0	1	→ upper bytes
1	0	→ lower bytes
1	1	→ idle state

MN/$\overline{MX}$:

→ Minimum mode / Maximum mode - pin 33.

→ when it is high (1), it works in minimum mode

→ " low (0), " " " Maximum "

Read → $\overline{RD}$

→ Read → $\overline{RD}$ is at pin-32.

→ Active low signal - used to read the signal.

Ready:

→ Ready is at pin 22.

→ when this signal is High (1) that indicates that the data is ready for transfer.

Reset:

→ Real Reset is at pin 21

→ It is used to restart the execution.

Maximum mode	minimum mode
[Pin : 24 to 31	
Pin 24 : QS_1	$\overline{INTA}$
pin 25 : QS_0	ALE
pin 26 : $\overline{S_0}$	$\overline{DEN}$
pin 27 : $\overline{S_1}$	$\overline{DT}/R$
pin 28 : $\overline{S_2}$	$M/\overline{IO}$
pin 29 : $\overline{Lock}$	$\overline{WR}$
pin 30 : $\overline{RQ}/GT1$	HLDA
pin 31 : $\overline{RQ}/GT0$.	HOLD

Minimum mode

$\overline{INTA}$:

-> INTA stands for Interupt acknowledge.

-> The INTA is a responce to INTR input signal.

$\overline{ALE}$:

-> ALE - Address latch enable.

-> whenever the address is present in ~~present~~ in the multiplexed address & data bus then this pin is enable

$\overline{DEN}$:

-> $\overline{DEN}$ - Data enable.

-> This is an active low pin.

DT/R :

-> This pin is used to show whether the data getting transmitted (or) received.

-> High (1) - data transmitting

-> Low (0) - " receiving.

M/$\overline{IO}$:

when M/$\overline{IO}$ = 1, memory operation

M/$\overline{IO}$ = 0, I/O operation.

$\overline{WR}$:- [write].

-> used to write data into memory.

HOLD

-> pin number 31.

-> It holds the value upto external device enables it

HLDA

-> HLDA - Hold acknowledgement. pin - 31

Maximum mode

RQ/GT1 and RQ/GT0.

-> RQ/GT1 - Request & grant signals used by other processors.

-> RQ/GT0 has high priority than RQ/GT1

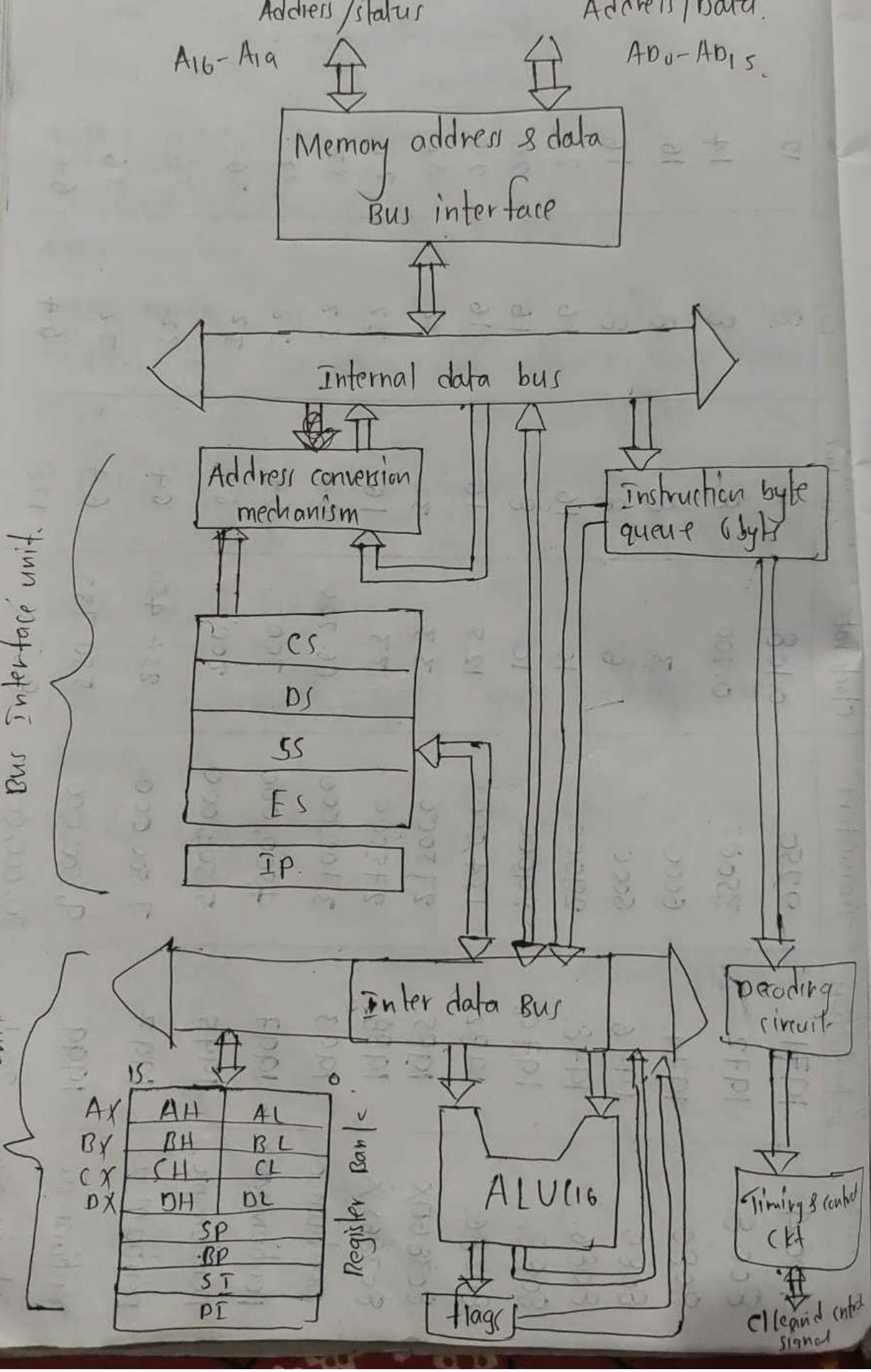
Lock :

-> It indicates other processors not to ask the cpu to release bus.

8086 microprocessor family

processor.	year of Introduction.	Transistors.	clock rate	External data bus.	Internal data bus	address Bus
4004	1971	2,250	0.108.	4	8	12.
8008.	1972.	3,500	0.200	8	8	14
8080.	1974.	6000	3	8	8	16.
8085	1976.	6,000	6	8	8	16.
8086.	1978.	29,000	10	16	16	26.
8088.	1979.	29,000	10	8	16	20.
80286.	1982.	134,000	12.5	16	16	25.
80386DX	1985.	275000	33	32	32	32.
80386SX	1988.	275,000	33	16	32	24.
pentium c	1993	3,100,000	66-200	64	32	32.
pentium MMX	1997	4,500,000	300	64	32	32
pentium pro.	1995	5,500,000	200	64	32	36.
pentium II	1997	7,500,000	233-450	64	32	36
pentium-III	1999	9,500,000	550-733	64	32	36
Itanium	2001	30,000,000	800----	128	64	64

Architecture of 8086



→ The function Block diagram of 8086 is divided into two functional units.

① Bus Interface unit ② Execution unit

①. Bus Interface unit

Bus interface unit is a gate interface between peripheral devices and processor.

→ The Bus interface unit contains.

ⓐ Instruction queue:-

→ In 8086 microprocessor, Instruction queue is a 6 byte register. used to store permanent data from the I/O devices.

→ This queue operates on the principle FIFO.

ⓑ Segment register

→ In 8086 processor, there are four segment registers. They are

ES - Extra segment
CS - code "
DS - data "
SS - stack "

→ The max memory of 8086 processor is 1MB.

→ Each segment have 64KB, so four segments will store 256 KB of memory.

ⓒ Instruction pointer:-

→ Instruction pointer will give the next address instruction to be executed.

2. Execution unit :-

The execution unit contains:

a. control unit.
b. Instruction decoders

c. ALU
d. general purpose register
e. Flat register

general purpose register

i. AX register (Accumlator):

→ AX register can give 16 bit data only.

2. BX register

→ BX register is the base register. It is used to save base data.

3. CX register.

→ CX register is code register.

4. DX register

→ DX register is data register, It is used to store data.

5. Stack pointer

→ stack pointer keeps the toap of the stack.
→ It operates on the principle LIFO.

6. Base pointer

→ Base pointer is used to store the base address of the memory.

7. Source Index:

→ Source index is used to ~~sto~~ hold the index valu of a source operand.

(8). Destination Index:

→ It is used to hold the index value of destination operand for string instructions.

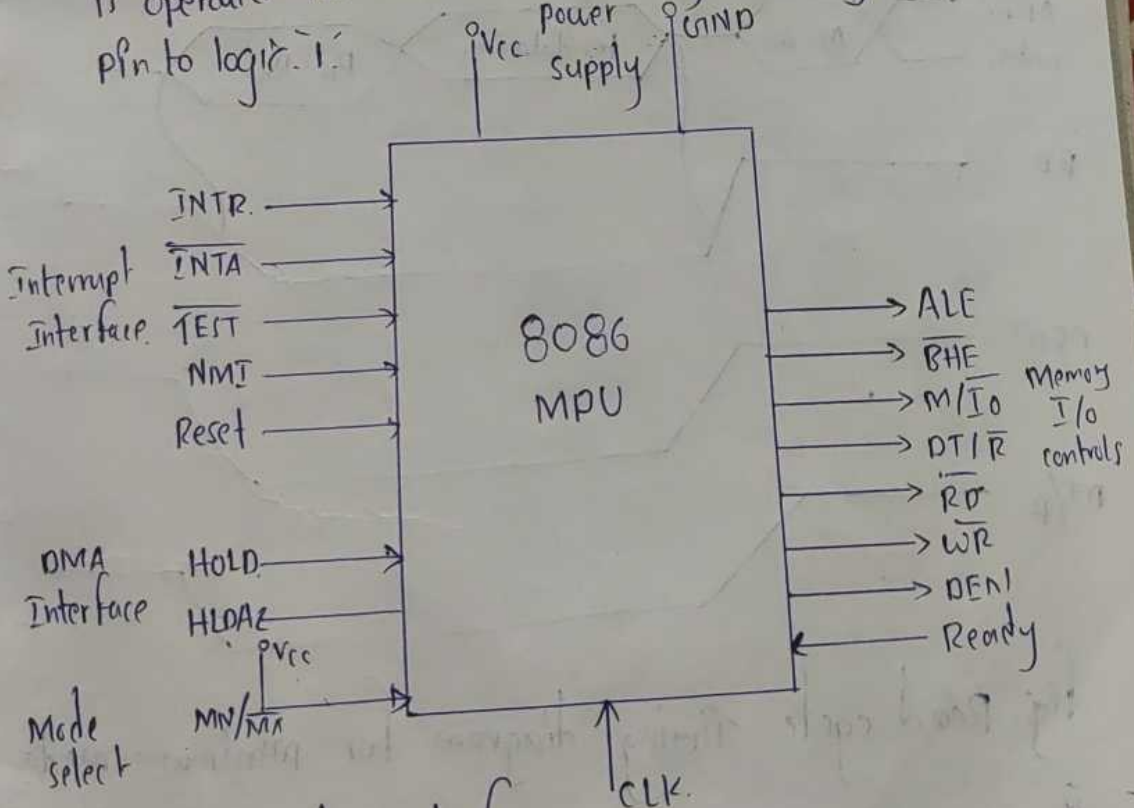
flag register

The three control flags are
1. ~~Trace~~ Trap flag
2. Interrupt flag
3. direction flag

minimum mode and maximum mode configuration

minimum mode of 8086

→ In minimum mode of 8086 system, the 8086 MP is operated in minimum mode by strapping it's MN/$\overline{MX}$ pin to logic 1.



Minimum mode Interface

→ Address latch enable (ALE):

-> ALE is a control signal.
-> ALE is a pulse to logic 1.

-> IO/M line: Memory transfer is selected.
(compliment for 8086.)

Timing diagram.

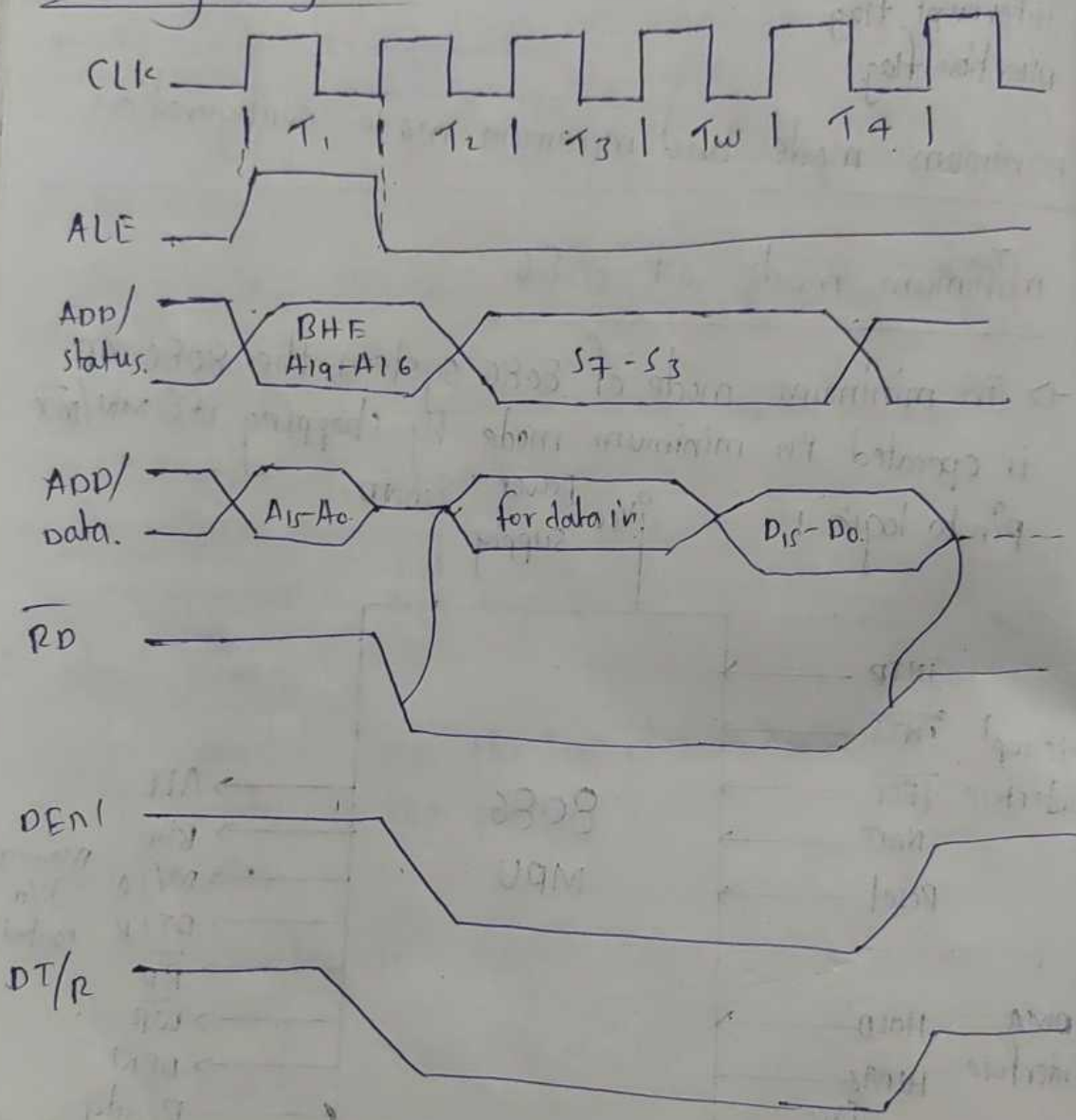


fig. Read cycle Timing diagram for minimum mode

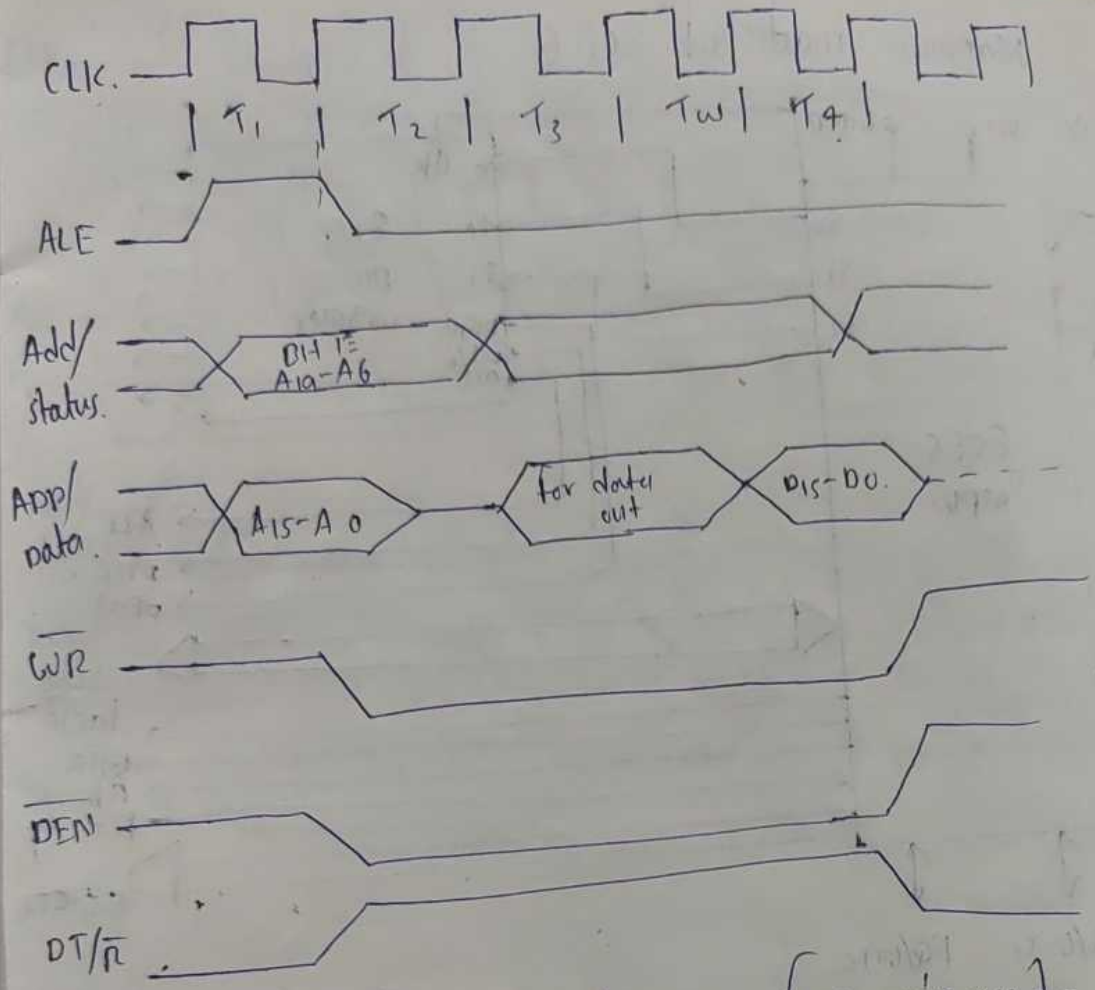
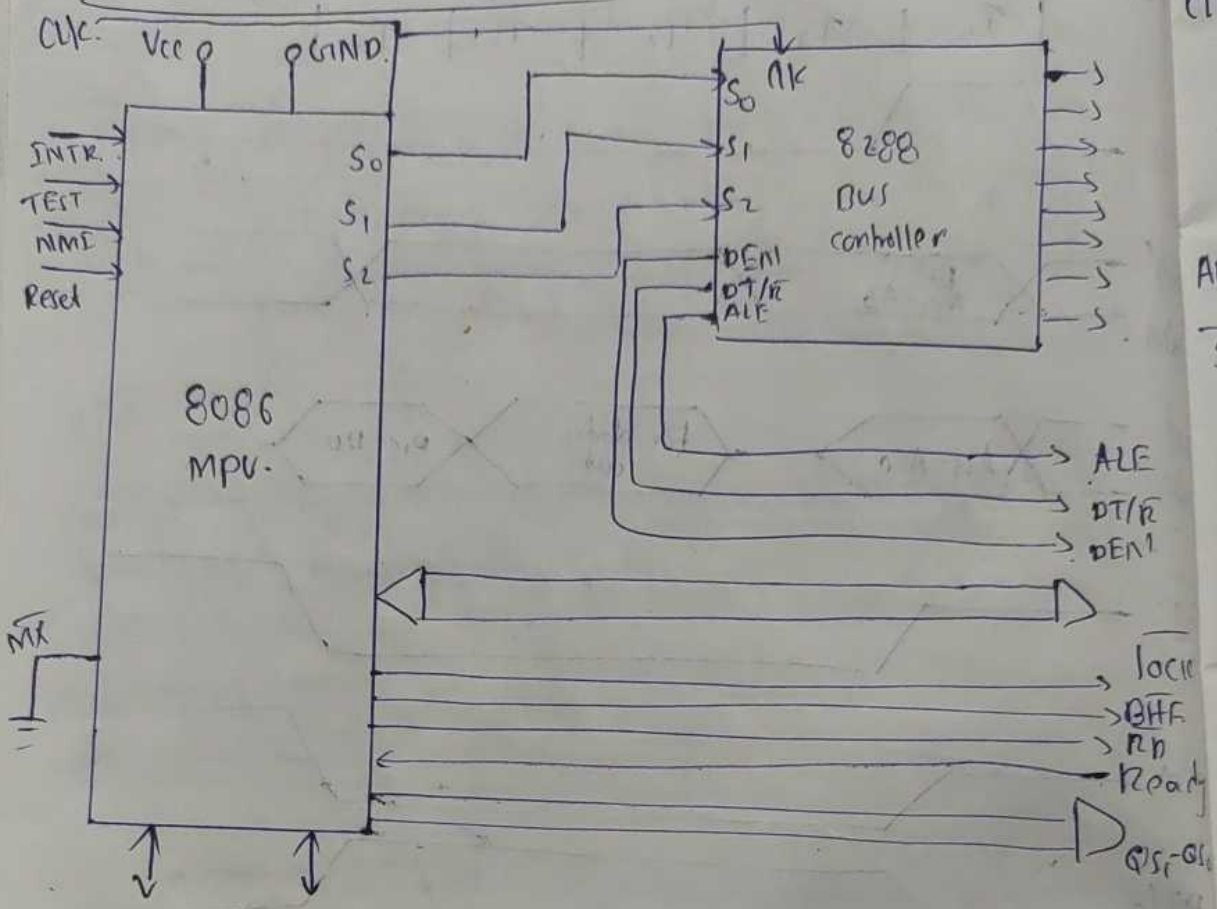
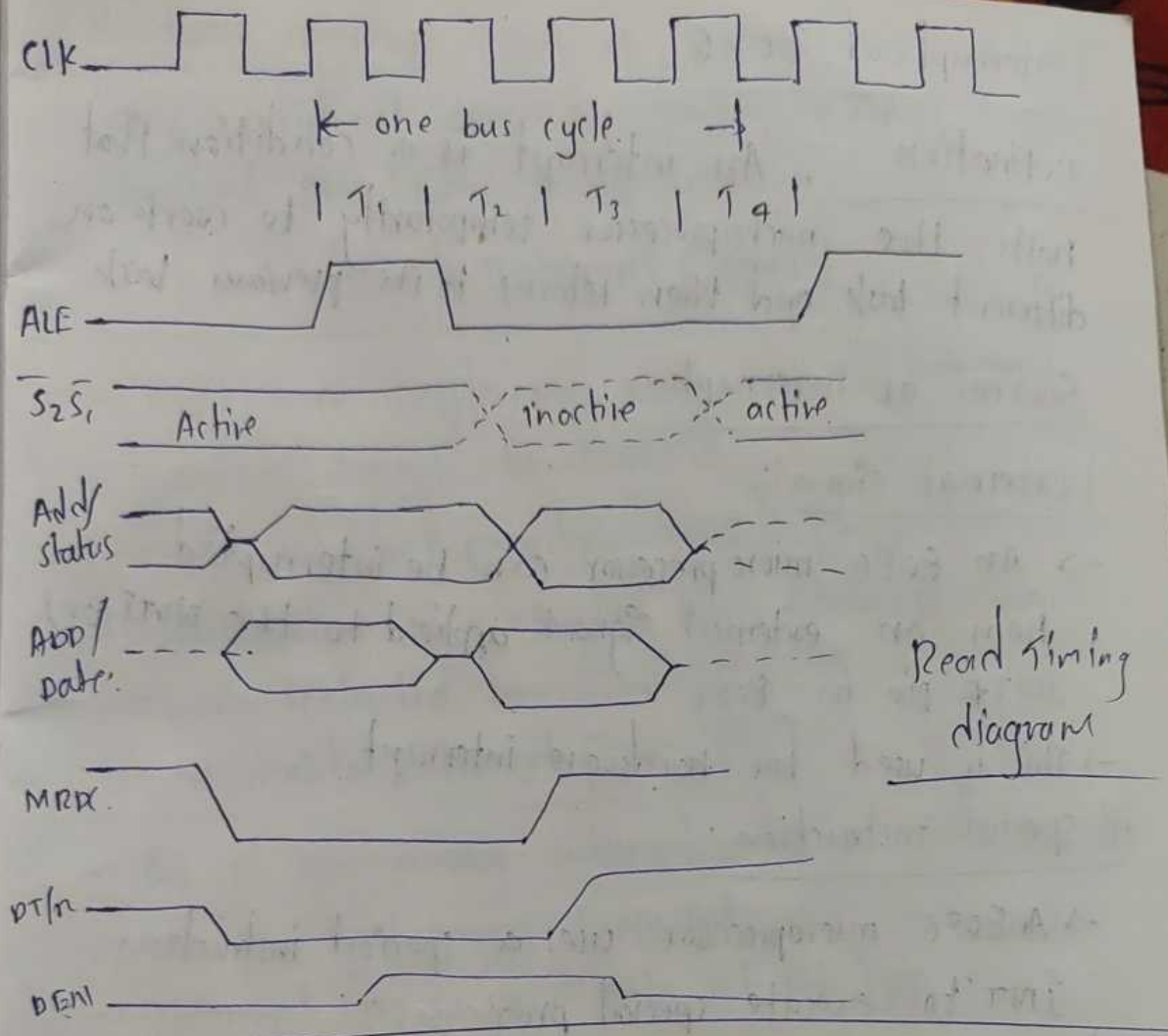


fig write cycle timing diagram for min mode.

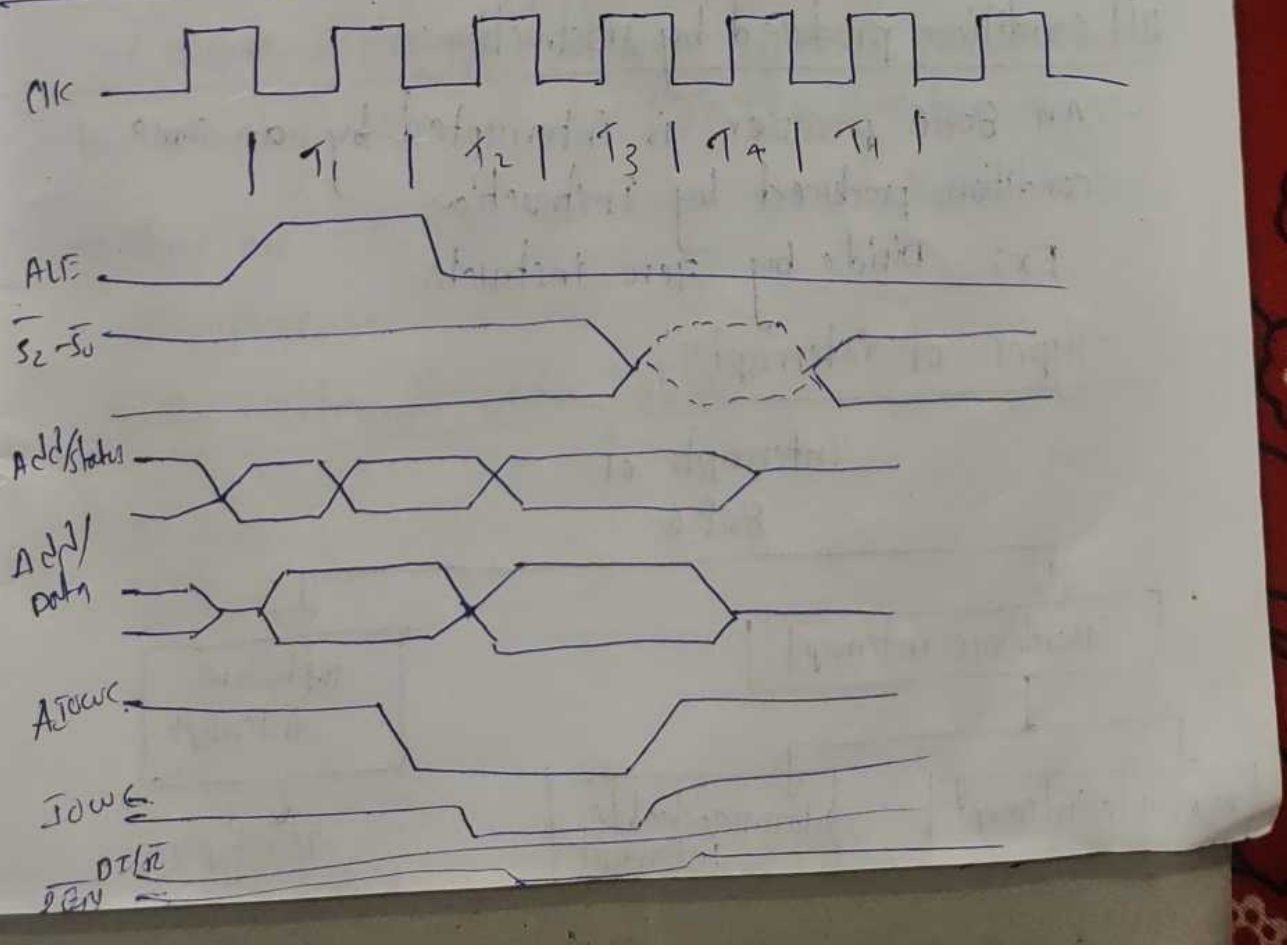
Maximum mode of 8086



Description in back pages



Read Timing diagram



Interrupts of 8086

Defination : An interrupt is a condition that halts the microprocessor temporarily to work on different task and then returns to it's previous task.

Sources of interrupts

i External signal :-

-> An 8086 microprocessor can be interrupted from an external signal applied to the NMI (or) INTR pin in 8086.

-> This is used for hardware interrupt.

ii) Special Instruction :-

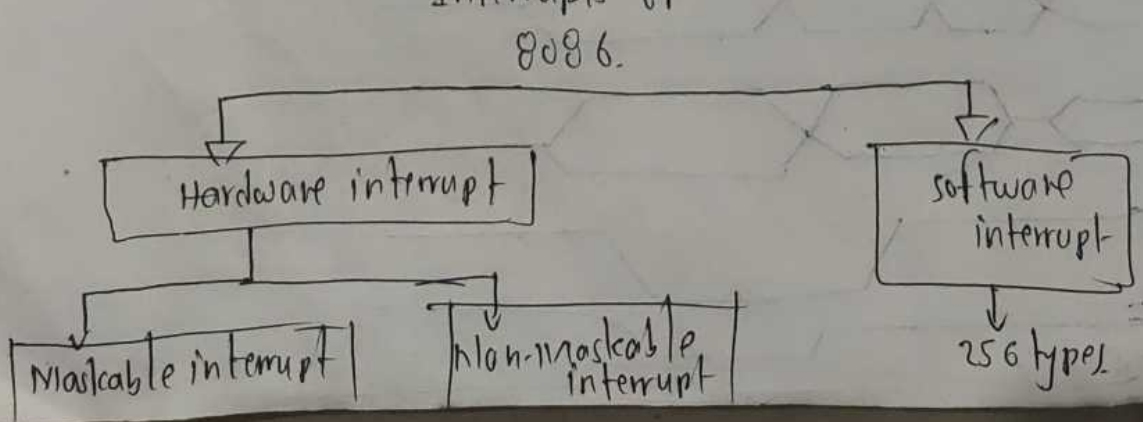
-> An 8086 microprocessor uses a special instruction 'INT' to execute special program

iii) condition produced by Instruction :-

-> An 8086 processor is interrupted by an some condition produced by Instruction.

Ex: Divide by zero instruction.

Types of interrupt :-



Hardware interrupt: Hardware interrupt is caused by any peripheral device by sending a signal through specified pin to the microprocessor.

i Non-maskable Interrupt [NMI]

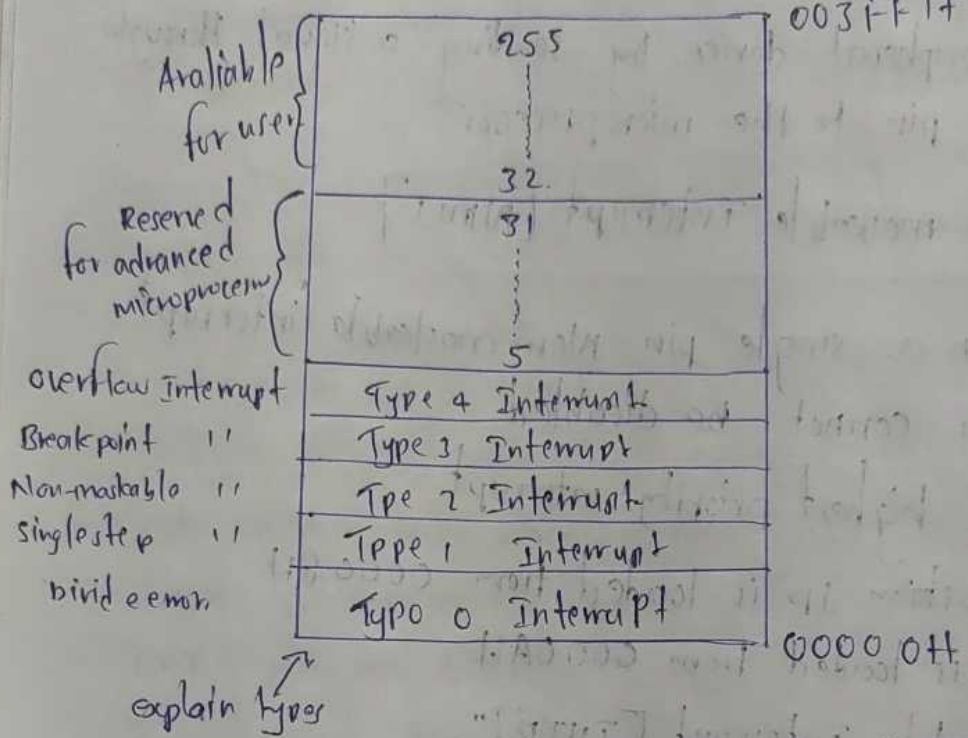
-> It is a single pin non-maskable interrupt which cannot be disabled.
-> It is highest priority interrupt.
-> ~~Instruction~~ Ip is loaded from 00008H.
-> Cs is loaded from 0000AH.

ii Maskable Interrupt [INTR]:-

-> This is low priority interrupt and is maskable as well
-> we can level trigger these interrupts.
-> Interrupt flag is used to set this type.

Software Interrupt: software interrupt can be generated by inserting the instruction "INT" within the program.
-> There are 256 software interrupts available in 8086 microprocessor.

8086 interrupt vector table:



priority Interrupt

Interupt	priority
Diride error.	Highest
NMI	↓
INTR	↓
single step Interrupt	lowest

Unit 2 8086 programming

✡ program development steps

1. Defining the problem:
   - first we need to define a problem, then we get output (or) result, by using flowchart, algorith etc
2. Requirement analysis:
   - program split into small parts.
3. Designing algorithms:
   - It is a step by step procedure to solve a problem.
4. Algorithm Testing:
   - we need to check algorithm to check correctness.
5. coding:
   - So, here we can use any programming language like C, C++, C++, Java, python, etc.
6. Testing:
   - compiler used
   - List of errors generated
   - errors are checked and rectified by one, by one
   - correct the errors.
   - Repeat testing.
7. Documentation:
   : To understand the logic behind the program.
8. Implementation:
   - program installed on the end user machine.
9. Maintainance, enhancement, Evolution etc

Assembly language program development tools

-> There are several tools to assembly language program development. st
-> They are discussed in brief here.

1. Editor:

-> It is a program which allows us to create a file containing the assembly language statement for our program.
-> Examples are pc write, word stars.
-> Creates source file to be processed by assembler.

2 Assembler:

-> It is program which is used to transfer the assembly language instruction to corresponding binary codes
-> It works ~~on~~ in passes, they are first and second passes.
-> It generates two files namely objective file [OBJ] and assembler list file. [IST].

3. Linker:

-> It is program which is used to join several objective files into large objective files.
-> while writing large program it is good to divide them into modules, so that each module can be written, tested, debugged ~~iname~~ independently and then use linker to combine all modules to form actual program.

4. Locator :

-> It is a program used to assign the specific address, where the segments of object code are to be loaded into memory.

5. Debugger

-> It is a program which allows you to load your object code program into system memory

-> It allows us to set Break points in at any points in the program.

-> Debuggers [Borland Turbo debugger.
microsoft code view debugger.

6. Emulators

-> one way to run your program.

-> It is mixture of software and hardware.

-> It is used to test and debug software and hardware.

Addressing modes of 8086 microprocessor.

Def: Addressing mode is defined as the method of specifying data to be operated by an instruction.

Types

1. Immediate mode:

-> The data is specified in instruction itself.

-> Data is a part of instruction.

Ex: MOV BL, 14H.
BL ← 14H.

2. Register mode:-

-> In this mode the data is specified using register

Ex: Mov AX, CX

-> The content of CX is moves into AX.

AX ← CX

3 Direct memory addressing mode:-

-> In this mode, the 16 bit effective address of the data is specified in the instruction.

Ex Mov CL, [4143H]

CL ← [4143H]

physical address => DS * 10H + 4143

=> 5000 X 10H + 4143

=> [54143H] -> CL.

4 Register indirect addressing mode:-

-> In this addressing the effective address is in SI, DI (or) BX.

Ex physical address = segment address + effective address

Mov AX, [DI]

Mov AL, [BX]

Mov AX, [SI].

(5). Based indexed mode:-

-> In this effective address is sum of base register and index register.

Ex Mov AL, [BP+SI]

Mov AX, [BX+DI].

Indexed mode :-

-> In this effective address is sum of inde
register and displacement.

Ex Mov AX, [SI+2000]

Mov AL, [DI+3000].

Based addressing mode :- In this addressing m
the effective address is sum of base register a
displacement. Ex: Mov AL, [BP+0100]

Input/output addressing mode :-

→ The addressing mode is related to input-output operations

Ex: IN A, 45
OUT A, 50.

String addressing mode :-

→ This addressing mode is related to string instructions.
→ The values of SI and DI are auto incremented and decremented.

Ex: MOVS B
MOVSW.

Based Indexed ad displacement mod.

→ In this the effective address is sum of index regist base register and displacement.

Ex: Mov AL, [SI+BP+2000].

✡ Assembler directives

Def: Assembler directives are the special instructions used to indicate the assembler. how a program is assembled and executed in a proper way.

→ Assembler directives are also called as pseudo-operations.

* Assume :- Assume directive assigns a logical segment to physical segment at any given time.
-> 8086 MP has 4 physical segments and no. of logical segments

Ex: Assume DS: Data.
Assume CS: CODE

* DB, DW, DD, DQ and DT:

-> These directives are used to define different types of variables

i. DB -> (Define byte).
-> It is used to declare byte variable.

Ex: Number DB: 10H, 20H, 30H.
-> Declare 3 bytes named 'Number'

ii. DW -> (Define word)
-> It defines word data types and initializes storage for word size.

iii DD : (Define double word)
-> This directive is used to declare a variable of type ~~word~~ double word.

iv DQ : (Define quadword)
-> This directive directs the assembler to receive 4 words (or) 8 bytes of memory for specified value

v DT [Define Ten bytes]
-> This directive directs the assembler to receive 5 words (or) 10 bytes memory for specified value

* EQU :- [Equate]

-> It is used to redefine a data name or variable with another name or variable.

Ex:- one EQU x+1;

name EQU <string>

Ex: Name EQU < Enter the name>;

* ORG [originat]:

-> This directive changes the starting offset address of the data.

Ex' ORG 1000H -> set location to 1000 H.

* END:

-> End of the program.

* proc and ENDP:

-> It defines the procedure used in the program.

Ex: name proc type

FACT PROC FAR.

-> This directive identifies the start of a program named FACT. and procedure FAR type.

-> ENDP indicates the end of the program.

EX FACT PROC FAR

:

ENDP.

* MACRO and ENDM

-> MACRO directive defines the start of the MACROS in the program.

-> ENDM indi defines the end of the MACRO

EX INIT MACRO

⋮

ENDM.

* SEGMENT and ENDS

-> SEGMENT directive defines the start of the segment

-> ENDS -> directive defines the end of the segment.

CODE SEGMENT

⋮

CODE ENDS

✡ Instructions of 8086.

Instruction: Instruction is a command given to the microprocessor to perform a specific task on specified data.

-> each instruction has two parts.

Instruction

opcode	operand

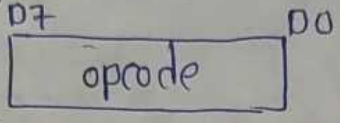
- op-code -> The task to be performed.
- operand -> The data to be operated on.

-> Instruction are two types
i Instruction formats . ii Instruction sets

i Instruction formats.

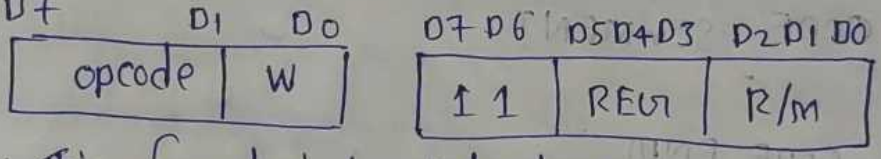
-> There are six instructions formats for 8086.
they are:-

i one byte Instruction :-



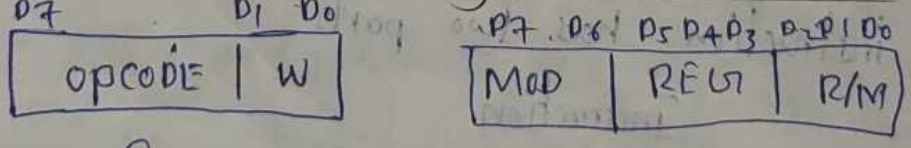
-> This format is only one bit long.
-> It may have implied data.

ii Register to Register (or) Two byte Instruction



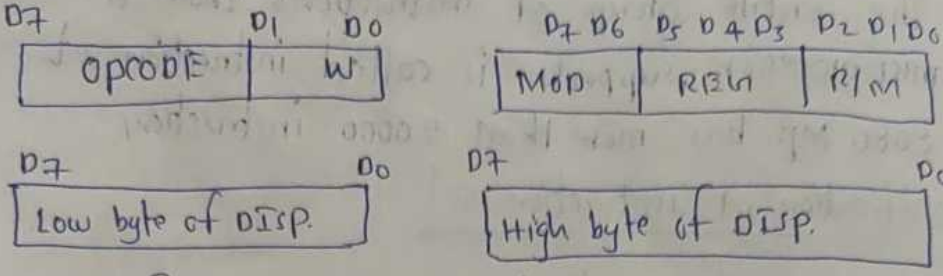
-> This format is two bytes long.
-> The first byte represents opcode and width of the operand 'w'. [w=1, 16 bit] [w=0, 8 bit]
-> The second byte represents register operands and R/M feilds.

iii Register to/ ~~form~~ from memory with no displacement :



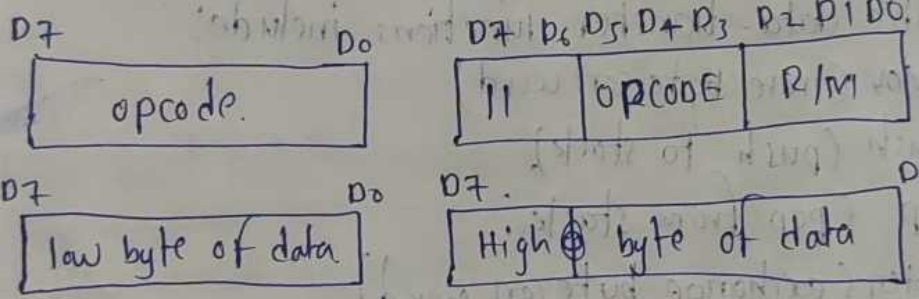
-> This format is also two bytes long, and similar to register to register format, except the MOD feild.
-> MOD represents mode of addressing.

IV Register to/from memory with displacement



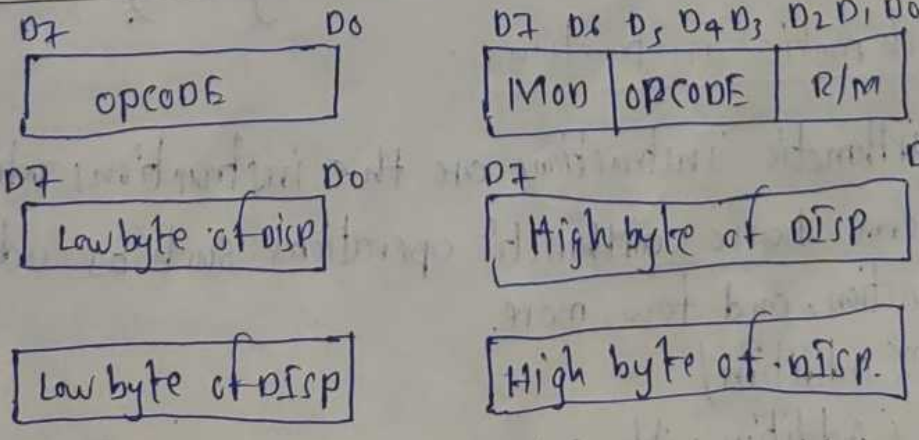
-> This format contains one (or) two additional bytes of DISP along with 2-byte format of register to/from (or) memory without displacement.

V Immediate operand to register:-



-> In this format, the first byte and 3-bits from the second byte are used to represent opcode.

VI Immediate operant to memory with displacement



-> This format 5 (or) 6 bytes long.
-> The first 2 bytes represent, opcode, MOD and R/M feilds.

ii Instruction set of 8086:-

-> The entire group of instructions that a microprocessor supports is called instruction set.

-> 8086 MP has more than 20000 instructions.

classification of Instruction set

1. Data Transfer Instructions:

-> Data Transfer instructions, which are used to movement of the data from one location to another location such as memory to microprocessor, microprocessor to memory vice-versa.

-> The data transfer functions include:

- MOV (move byte (or) word)
- push (push to stack)
- pop (pop from stack)
- XCHG (exchange byte (or) word)
- IN (Input to the port (or) copies a byte (or) word)
- OUT (output to the)
- PUSHA (push all registers into stack)
- popA (get words from stack to all registers)

2 Airthmetic Instructions:-

-> Airthmetic Instructions are the instructions which perform basic airthmetic operations such as addition, substraction, and few more.

- ADD (Addition)
- ADC (Addition with carry)
- SUB (substraction)

- SBB (substraction with Borrow).
- MUL (multiplication)
- IMUL (Integer multiplication)
- DIV (Division.
- IDIV (Integer Division)
- CMP (compare
- INC (Increment)
- DEC (Decrement)

3. Logical Instructions :

-> logical instructions are the instructions which perform basic logical operations such as AND, OR etc.

- NOT
- AND
- OR
- XOR.

4 Shift and Rotate instructions

-> shifting means to move bits rigght and left inside the operand.

- SHL (shift left)
- SHR (shift right)
- SAL (shift Airthmetic left)
- SAR (" " right)
- ROL (Rotate left)
- ROR (" right)
- RCL (Rotate carry left)
- RCR (" " right).

⑤. Branch and loop instruction

-> These instructions are used to transfer / Branch the instructions during an execution.

-> unconditional Branch

- CALL
- RET
- Jump
- LOOP

-> conditional Branch

- JC
- JNC
- JE/JZ
- JNE/JNZ
- JO
- JNO.
- LOOPE/LOOPZ
- LOOPNE/LOOPNZ

Processor control Instructions

-> processor control Instructions are instructions which control the processor's action by setting(1) (or) resetting(0). the values of flag register.

- sets carry flag to 1
- resets " " to 0
- sets directional flag to 1
- resets " " " to 0.
- sets the interrupt flag to @ 1.
- resets " " " to 0

Unit-3

Semiconductor memories interfacing

Def: A device used for storing digital information that is fabricated by using integrated circuit technology is called semiconductor memory.

-> Types of semiconductor memories:-

1. Random access memory.
   - i DRAM
   - ii SRAM.
2. Read only memory.
   - i PROM
   - ii MROM.
   - iii EPROM.
   - iv EEPROM

1. Random access memory: Random access memory is a form of semiconductor memory technology. that is used for reading and writing data in any order.

-> Data is stored and Read many times.

-> Random access memory is used huge in computer applications.

i Dynamic RAM':

-> Dynamic RAM is a type of RAM that uses capacitors to store each bit of data.

-> DRAM is slower and less expensive than SRAM.

-> These capacitors does not hold their charge indefinitely.

-> DRAM is used in personal computers and workstations.

unit - 4

Microcontroller

Microcontroller : Microcontroller is a compact integrated circuit designed to govern a specific operation in a embedded system.

-> The general microcontroller consist of processor, and the memory, serial ports, peripherals. etc.

Difference b/w microcontroller and Microprocessor

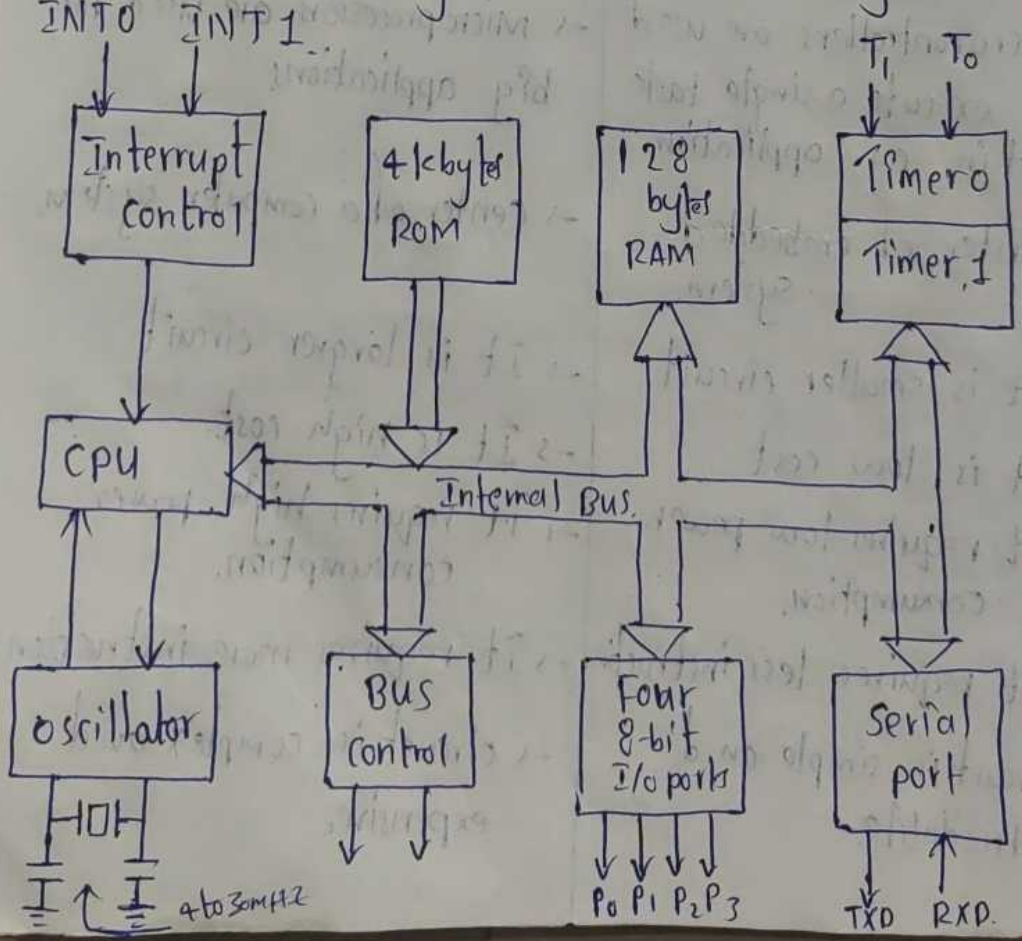
Micro controller	Microprocessor.
-> Microcontrollers are used to execute a single task within an application.	-> Microprocessors are used for big applications.
-> center of embedded system.	-> center of a computer system.
-> It is smaller circuit.	-> It is larger circuit.
-> It is low cost.	-> It is high cost.
-> It requires low power consumption.	-> it requires high power consumption.
-> It requires less instructions.	-> it requires more instructions.
-> circuit is simple and affordable.	-> circuit is complex and expensive.

-> It have more register.	-> It have less register
-> Memory and I/o Components are internal to it.	-> Memory and I/o Components are external to it.
-> RAM, ROM and other peripheral are present.	-> RAM / ROM and other peripherals are absent.
-> speed is depends upon architecture.	-> High speed.

Architecture of 8051

-> 8051 is an 8-bit microcontroller, ~~in the~~ designed by intel in 1981.

-> It has 4k bytes of ROM and 128 bytes of RAM.



* control processing unit :-

-> Airthmetic logic unit [ALU].
-> Registers - A, B, Temporary registers.
-> 1-bit program counter.
-> Data painter.

-> we know that CPU is the brain of the processing device.
-> It moniters and controls the all operations performed by the microcontroller.

* Interrupt :-

-> 8051 provides 5 interrupt sources.

- Time o overflow interrupt.
- Time 1 overflow interrupt.
- External hardware interrupt INTO.
- External hardware interupt INIT1.
- Serial communication interupt RI/TI.

Memory :-

-> Microcontroller requires a program which is a collection of instructions.
-> These programs require a memory to save and read by microcontroller.
-> The memory which is used to save program is called ROM (or) code memory.
-> 8051 has 4kbyts ROM and 128 bytes RAM.

Internal Bus

-> Bus is defined as collection of wires, which work as communication channel.
-> These bus can carry 8 bit, 16 bits simultaneously.
-> Here two types of buses.
- Address Bus.
- Data Bus.

Address Bus

-> 8051 has 16 bit address bus for transfer data.
-> It has four addressing modes.
1. Immediate addressing mode.
2. Register " "
3. Indirect register " "
4. Direct a " "

Data Bus

-> 8051 has 8 bit of data bus,
-> used to carry data.

oscillator :

-> we know that microcontroller is a device.
-> It require a clock pulses.
-> For this 8051 has on-chip oscillator which works as a clock soure for cpu.
-> The output pulses are stable.

Input/output port :

-> 8051 has 32 bidirectional I/O pins.
-> To connect it to other machines or peripherals we require I/O interfacing ports.
-> 8051 has 4 I/O ports.
port 0 - I/p port and multiplexed AD_0-AD_7.
port 1 - only for I/p.
port 2 - I/p port and High order address.
port 3 - I/O ports and multifunctional.

Serial port

-> The serial port of 8051 is duplex.
-> It can transmit and receive data.
-> used for serial communication.
- TXD
- RXD

Applications

-> Consumer applications
-> Home appliances.
-> communication systems.
-> In offices
-> Automobiles.
-> medical equipment.
-> Robotics.
-> Aeronautical and space.

Features

-> Full duplex.
-> Five interrupts.
-> 32 I/O ports.
-> special function register.
-> 16-bit program counter.
-> 8-bit, stack pointer.

Special function Registers of 8051

-> 8051 is an 8-bit microcontoller and designed by intel in 1981.

-> In 8051 microcontoller SFRs are to used to perform special functions.

-> SFR's are used as a control table for accessing and monitering the various operations of microcontoller.

-> SFRs are located between 80H to FFH in RAM.

-> out of 128 memory locations, there are only 21 locations assigned to SFRs.

-> List of SFRs

1. Acc - Accumlator.
2. B - Register.
3. PSW - program status word.
4. Stack pointer.
5. DPTR - Data pointer.
6. I/O ports - $P0, P1, P2, P3$.
7. IE - Interupt enable register.
8. IP - " priority "
9. TMOD - Timer mode control.
10. TCON - Timer control.
11. SCON - serial control.

1. Accumlator :

-> It is a 8-bit special purpose register, used to access cpu.

-> Address : 0E0H, in RAM.

2. B-Register :

-> It is 8-bit general purpose register.

-> Address 0F0 H.

3. program status word (PSW):

-> It is a flag register.

-> Address : 0D0H to 0D7H.

0D7	D6	D5	D4	D3	D2	D1	0D0
CY	AC	F0	RS1	RS0	OV	-	P.

CY - carry flag.
AC - Auxillary carry flag.
F0 - flag 0 for user.
RS1 & RS0 - Register bank select signals.
OV - over flow flag.
P - priority flag.

④. Stack pointer

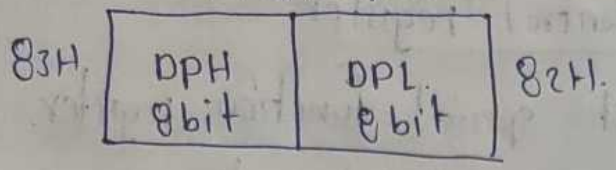
-> It is a 16-bit register, used to store address of latest instruction in stack.

-> Address B1H.

⑤. DATA pointer [DPTR]:

-> It is a 16-bit register, used to hold a 16-bit address / data.

16 bit DPTR.



⑥. I/O ports

-> 4 parallel 8-bit Input/output ports.

P0 - port 0 : Address 80H.
P1 - port 1 : Address 90H.
P2 - port 2 : Address 0A0H.
P3 - port 3 : Address 0B0H.

⑦. Interrupt Related registers

(i) IE - Interrupt enable register.

-> It is a 8-bit register, which is responsible for enabling / disabling the interrupts.

-> Address: 0A8H.

(ii) IP - Interrupt priority register.

-> It is used to change the priority levels of the interrupts.

-> Address. : 0B8H.

⑧. Timer / counter registers

i TMOD - Timer mode control register

-> It is a 8-bit special function register., which is responsible for Timer 0 and Timer 1 operations.

-> Address: 89H

ii TCON - Timer control register.

-> It is an 8-bit special function register.
-> used to specify external interrupts
-> Address : 88H.

Serial port registers

i SCON :- serial control register.
-> It is an 8-bit SFR used to select programmable mode.

ii PCON :- power control register.
-> It is a 8-bit SFR used to control ~~pow~~ power levels.

Addressing Modes of 8051.

Def :- It is defined as the way of specifying the data in an instruction.

Types

1. Immediate addressing mode

-> In this mode the data is provided in the instruction itself.
-> Data is a constant value.

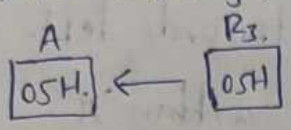
Ex MOV A, #05H ; Mov data 05H to the accumulator.

A
[05] ← 05H.

2. Register addressing Mode

-> In this mode the source and destination data should be given in the register. $[R_0 - R_7]$.

Ex: MOV A, R_3;



3. Direct addressing Mode :-

-> In this mode, the source (or) destination address is specified by using 8 bit in the instruction itself.

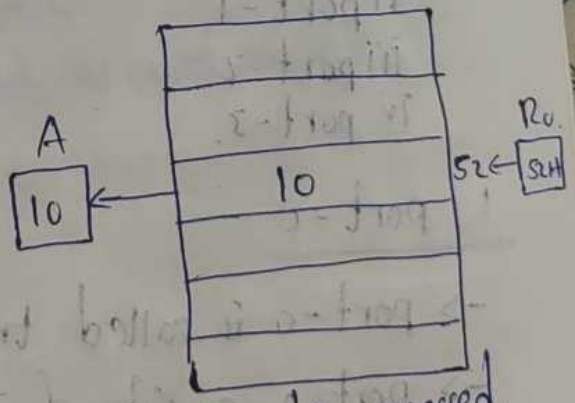
Ex: MOV R_0, 45H.

4. Register Indirect addressing Mode

-> In this mode the source and destination address are given in the register.

Ex: MOV, A, @R_0.

-> @ symbol is used.



5. Indexed addressing Mode

-> In this mode only program memory can be accessed.

-> The destination operand is Accumulator.

Ex. MOVC A, @A+DPTR.

⑥: Implied Addressing mode

-> In this mode the operands are specified Implicitly in the instruction itself.
-> single operand is used.

Ex RLA : Rotate A two towards left.
SwapA ; swap the nipples within A.

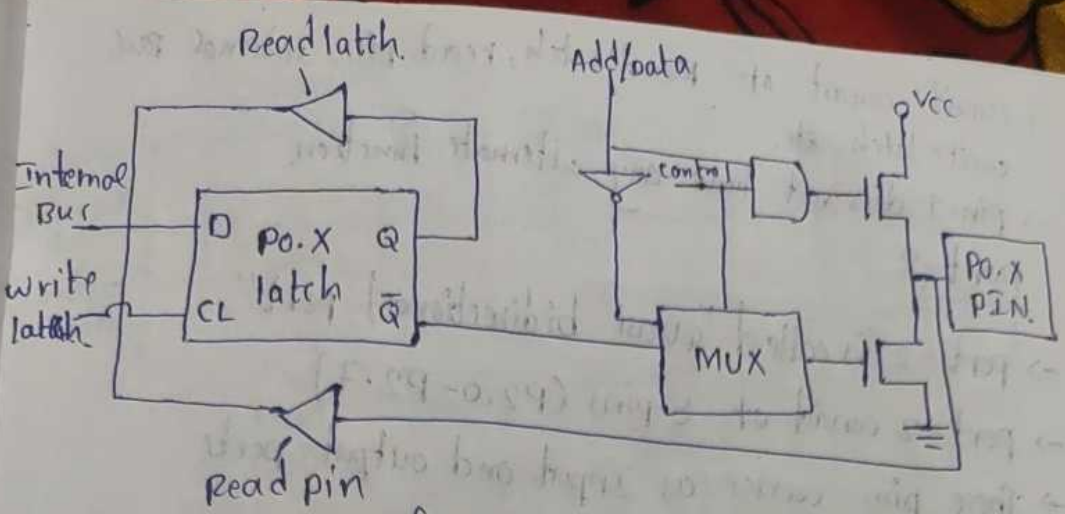
Instruction set of 8051 microcontroller

I/o- pins ports and circuits

-> Each port of 8051 has bidirectional capability.
-> Basically 8051 microcontroller have 4 ports.
-> They are:
i port-0
ii port-1
iii port-2
Iv port-3.

i port-0

-> port-0 is called true bidirectional port.
-> port-0 consist of total 8-pins from (pin No. 32-40)
-> These pins are work as Input (or) output ports.
-> Dual role: AD0-AD7 - used for both address and data.
-> port-0 circuit diagram.



-> circuit consist of Read pin, Read latch, control, MUX, two MOSFETs. Internal Bus, write latch etc.

-> when control is 0 - Both output MOSFETs are off.

-> when control is 1 - upper MOSFET is OFF and lower MOSFET is ON.

-> port 0 is used for external memory access.

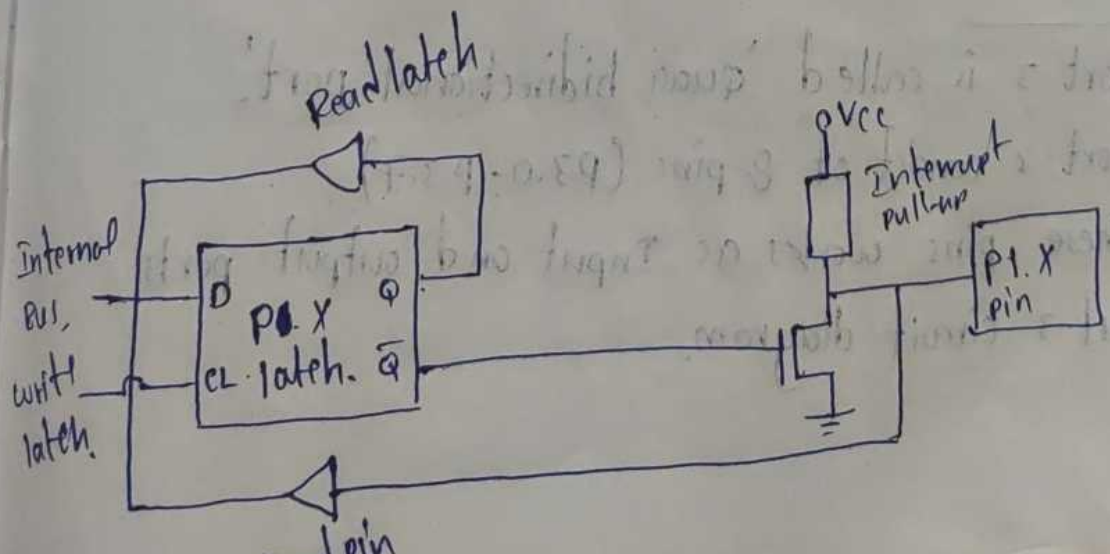
ii port-1

-> port-1 is called 'Quasi bidirectional port'.

-> port-1 consist of total 8-pins.

-> These pins are work as Input (or) output ~~pin~~ port.

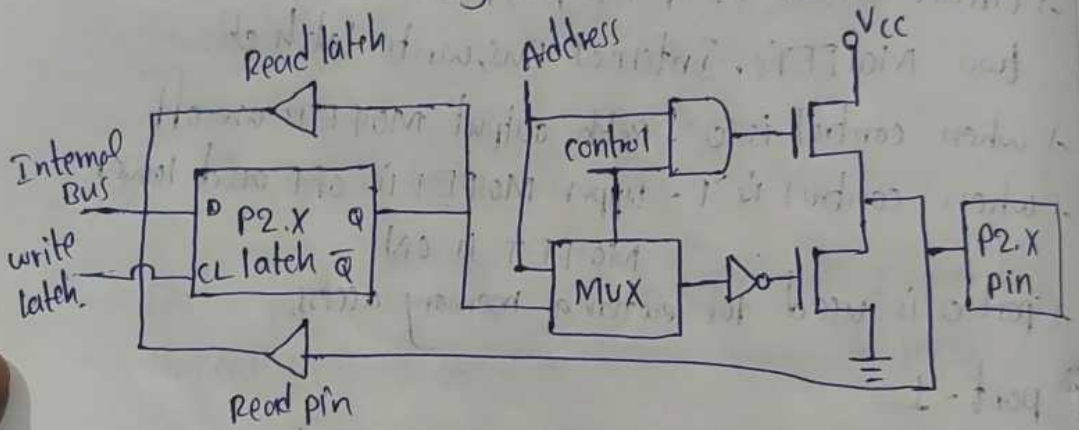
-> port-1 circuit diagram.



-> Circuit consist of Read latch, read pin, Internal Bus, write latch etc.
-> Pin-1 does not have any alternate function.

Port-2

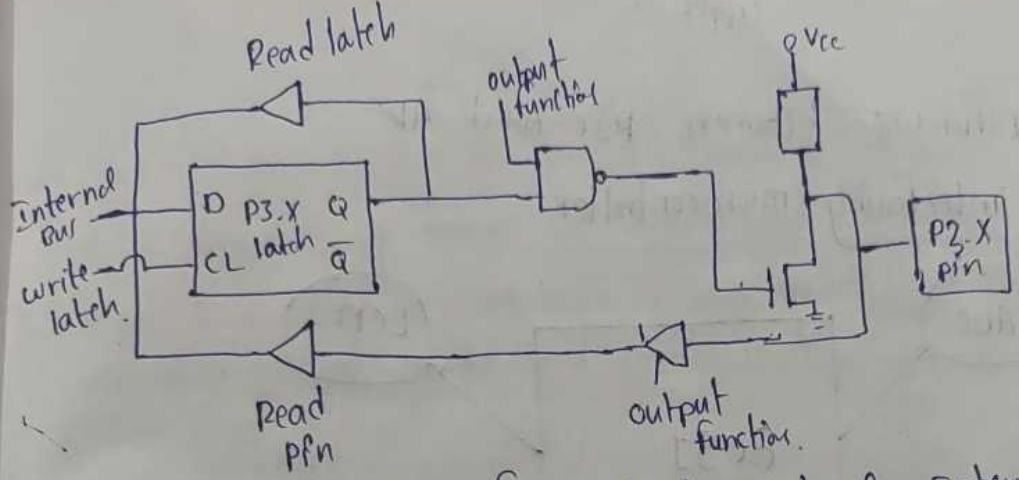
-> port-2 is called "quasi bidirectional port"
-> port-2 consist of 8-pins (P2.0 - P2.7)
-> These pins works as Input and output ports.
-> port-2 circuit diagram



-> circuit consist Read latch, Read pin, Internal Bus, write latch, MUX. etc.
-> port-2 is used for higher external address.
-> The I/O operation is similar to port-1.

Port-3

-> port-3 is called 'quasi bidirectional port'.
-> port-3 consist of 8-pins (P3.0 - P3.7)
-> These pins works as Input and output ports.
-> port-3 circuit diagram.

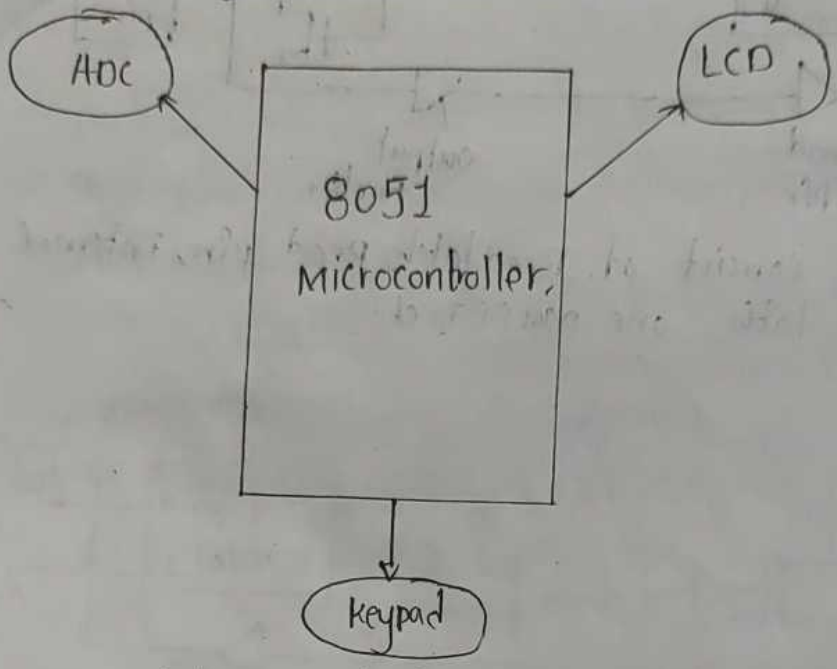


→ The circuit consist of Read latch, Read pin, Internal Bus, write latch, one MOSFET. etc.

Unit-5

Difference between PIC and AR

Interfacing microcontroller



-> Every electrical and electronics projects designed to develop electronic gadgets, that are frequently used in our day-to-day life.

-> We utilites the microcontrollers and interfacing devices

-> There are different types of applications, that are designed using microcontroller-based projects.

-> In maximum number of applications, the microcontroller is connected with some external devices called as interfacing devices, to perform specific tasks.

-> For example consider a security system with a user changeable, in which interfacing device, keypad is interfaced with microcontroller to enter the password

Programming 8051 Timer

-> In intel 8051, there are two 16-bit timer registers.
-> These registers are known as Timer0 and Timer1.
-> These timer registers are used in two modes, these modes are called Timer mode and counter mode.

Timer mode

-> In timer mode, internal machine cycles are counted.
-> So, the register is incremented in each cycle.
-> So, when clock frequency is 12MHz, the timer register increments in each millisecond.
-> This mode ignores the timer I/p pins.

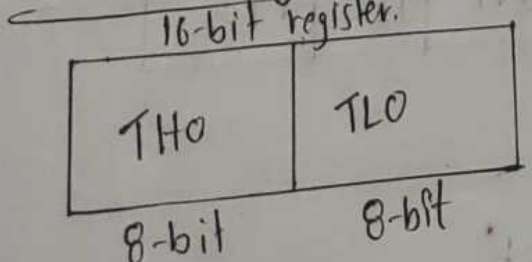
Counter Mode

-> In counter mode, the external events are counted.
-> So, the register is incremented for each 1 to 0 transition of the external I/p pin.
-> These transitions are treated as events.
-> In this mode two machine cycles are needed.

-> 8051 has two 16-bit timer registers

i Timer '0'
ii Timer '1'

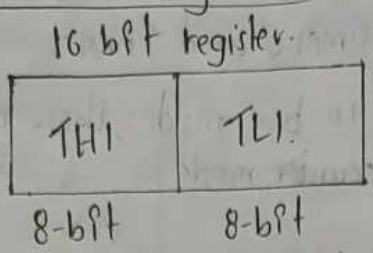
i. Timer '0' register



-> The 16-bit register of Timer '0' has low and high byte.

TL0 - Low byte register
TH0 - High byte register

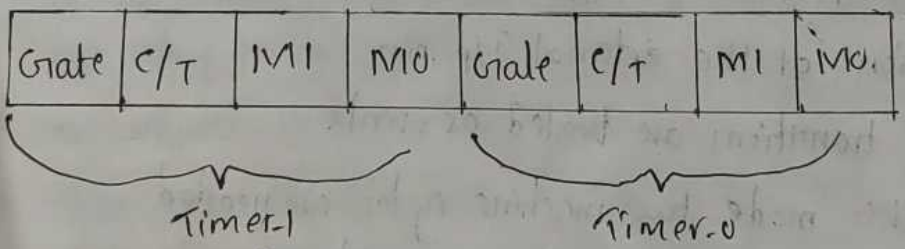
ii) Timer 1 Register :



-> The 16-bit Timer 1 has low and high byte.
TL0 - Low byte register.
TH0 - High byte register.

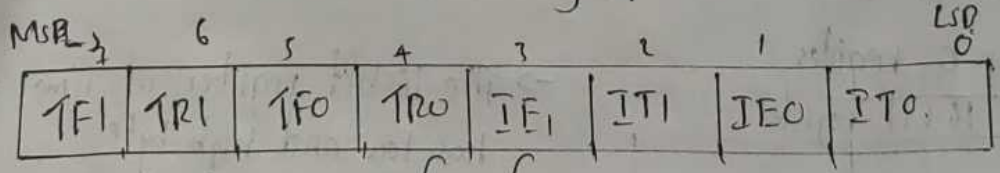
TMOD Register :

-> TMOD is the 8-bit special function register, which is responsible for the following operations:
- select Timer 0
- select Timer 1
- select modes of the Timer



TCON Register

-> TCON is the 8-bit special function register, is used to control following operations



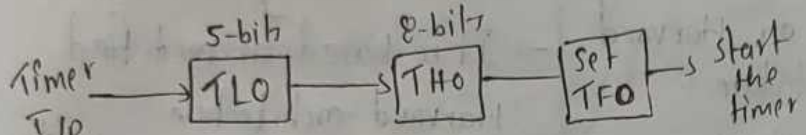
-> TF1 - Timer 1 overflow flag
-> TR1 - Timer 1 Run control bit.
-> TF0 - Timer 0 over flow flag
-> TR0 - Timer 0 Run control bit.

-> IE1 - Interupt 1 type edge flag
-> IT1 - Interupt 1 type contol bit.
-> IE0 - Interupt 0 edge flag
-> IT0 - Interupt 0 type contol bit.

Modes of operation

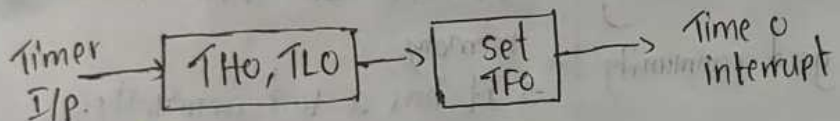
i Mode 0:

-> It is a 13-bit timer/counter



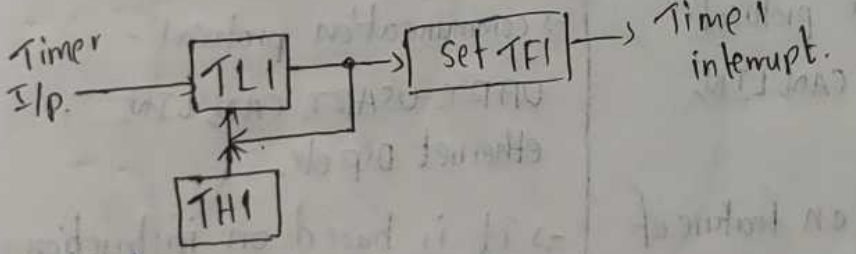
ii) Mode 1:

-> It is a 16-bit timer/counter.



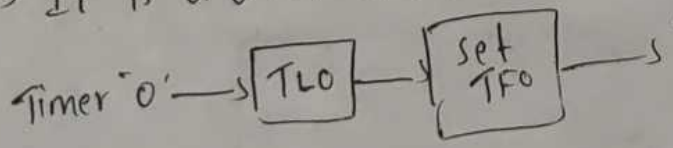
iii Mode 2:

-> It is a 8-bit auto reload timer/counter.



Iv Mode-3:

-> It is a 8-bit timer/counter.



Comparision b/w PIC and ARM

PIC	ARM.
-> PIC stands for peripheral interface controller.	-> ARM stands for Advanced RISC machine.
-> It's manfacturer is microship.	-> It's manfacturer is apple, Nvidia, samsung. and TI etc.
-> It is based on Harvard memory architecture.	-> It is based on modified Harvard architecture.
-> It is avaliable, in 8, 16, 32-bits.	-> It is avaliable in, 32 and 64 bits.
-> It uses SRAM, flash memory	-> It used SDRAM, flash, EEPROM memory.
-> It has very good community support.	-> It has a vast community support.
-> Average cost.	-> low cost.
-> communication protocol: UART, USART, CAN, LIN, ethernet etc.	-> communication protocal - UART, USART, CAN, LIN, ethernet DSP etc.
-> It is based on feature of RISC.	-> It is bared on instruction set of RISC.

Stepper Motor

-> A stepper motor is one of the most commonly used motor for precise angular movement.

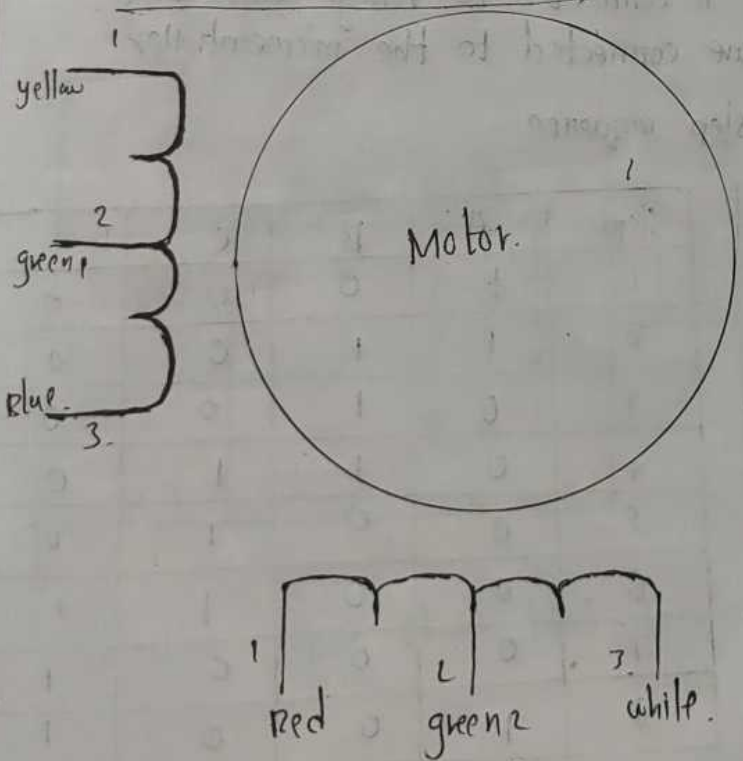
-> The advantage of stepper motor is that angular position of the motor can be controlled without any feedback mechanism.

-> The stepper motors are widely used in industrial and commerical applications.

-> They are also commonly used in robots, washing machine etc.

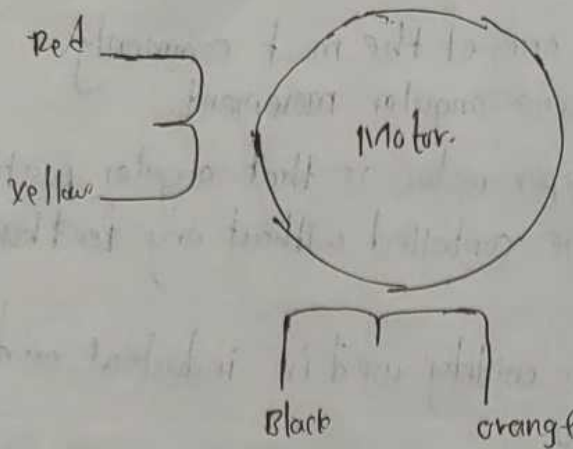
-> stepper motor is of two types
  i. unipolar stepper motor.
  ii Bipolar stepper motor.

i unipolar 6-lead stepper motor.



Here : 1, 3 : endpoints.
2 : common point.
——— phase 1
——— phase 2.

ii. Bipolar stepper motor:



-> stepper motor can be unipolar and Bipolar.
-> And here we are using unipolar stepper motor.
-> unipolar stepper motor consist of six wires out of which four are connected to coil of the motor and two are common wires.
-> Each common wire is connected to voltage source and remaining wires are connected to the microcontroller.

other options for step sequence

steps	A	B	C	D
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

wave sequence

steps	A	B	C	D
1	1	0	0	0
2	1	1	0	0
3	0	1	0	0
4	0	1	1	0
5	0	0	1	0
6	0	0	1	1
7	0	0	0	1
8	1	0	0	1

Half drive sequence

Motor speed:

It is measured in steps / sec.

LCD

-> LCD
disp
-> LCD
-> A
-> 16x
an
-> LCD
-> Co
st

LCD and keyboard interfacing :-

-> LCD - Stands for Liquid crystal display which can display the characters per line.

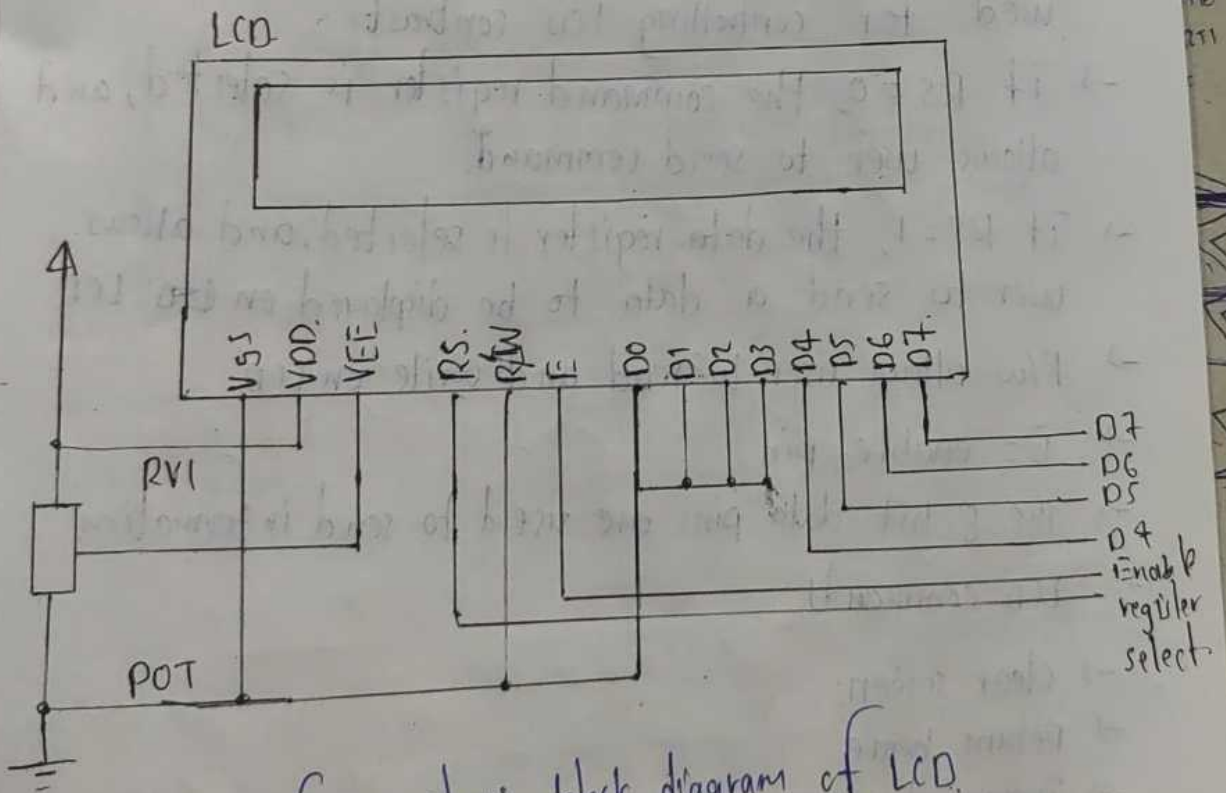
-> LCD used in wide range of applications.

-> A popular LCD is 16X2 LCD.

-> 16X2 LCD means it can display 16 characters per line and there are 2 such lines.

-> LCD has two registers command and data.

-> command register stores commands and data register stores displayed data.



-> Above figure shows block diagram of LCD.

Pin No:	symbol	External connection.
1.	Vss.	power supply
2.	V_{DD}.	power supply
3.	V_o	power supply.
4.	RS	MPU
5.	R/W	MPU
6.	E	MPU
7-10.	DB0 - DB3	MPU.
11-14.	DB4 - DB 7.	MP.

-> Table shows pin discription of LCD.

-> Vss and V_{DD} provide +5v and ground V_o is used for controlling LCD contrast.

-> If RS = 0, the command register is selected, and allows user to send command.

-> If RS = 1, the data register is selected, and allows user to send a data to be displayed on LCD.

-> R/w allows user to read and write on LCD.

-> E = enable pin.

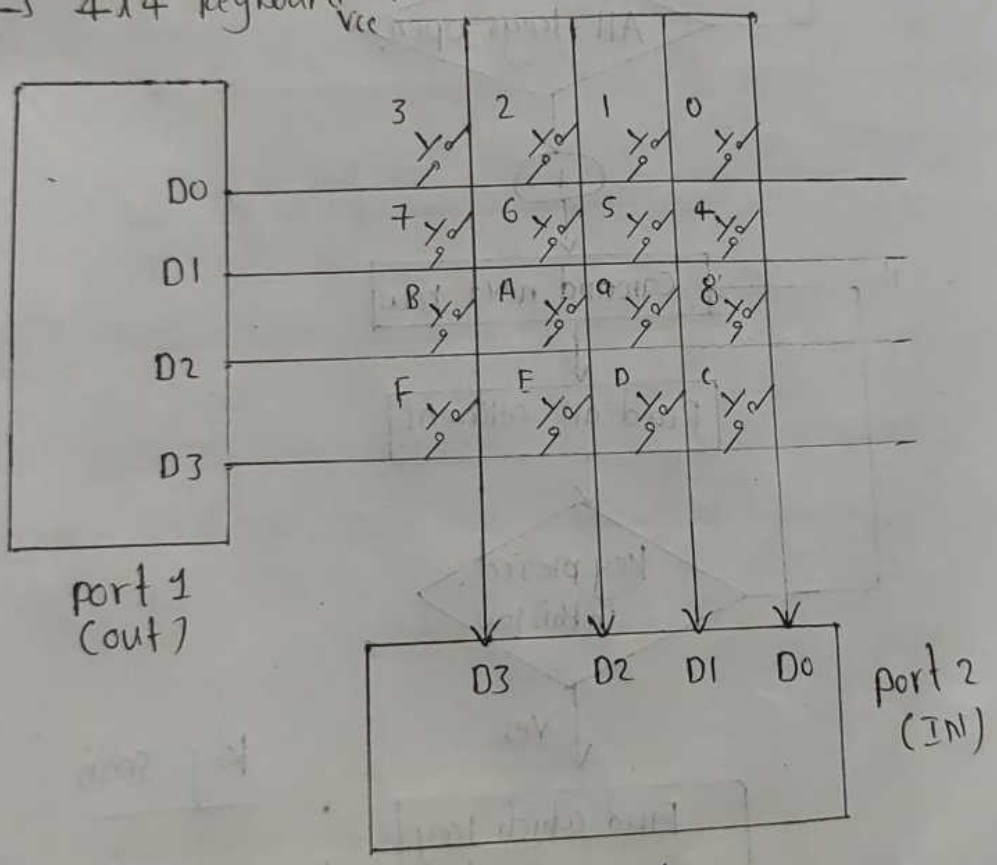
-> The 8-bit data pins are used to send information.

-> LCD commands

-> clear screen.
-> Return home.
-> Increment cursor.
-> Decrement cursor.
-> shift display right
-> shift " left
and soon.

Keyboard Interfacing

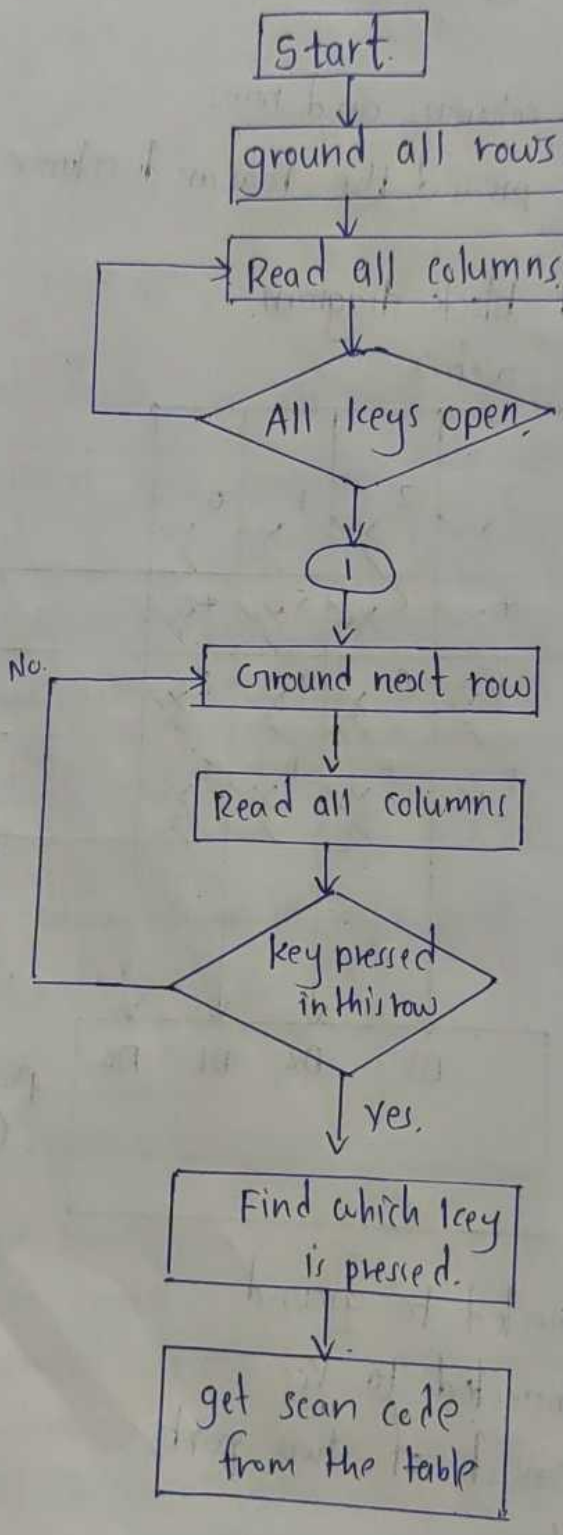
-> Keyboard is most widely used input device.
-> Keyboards are organized in a matrix of rows and columns.
-> CPU access both columns and rows.
-> When switch is pressed, the row and column make contact.
-> Matrix keyboard block diagram.
-> 4x4 keyboard matrix



-> All rows are connected to ground.
-> All columns are connected to Vcc.
-> Block diagram consist of two ports.
  i port1- output.
  ii port2- Input.

-> The rows are connected to an output port (1) and colums are connected to an Input port (2).

-> Flow chart for keyboard interfacing



Key scan

1110 - key presed in colum
1101 - " " " "
1011 - " " " 2
0111 - " " " 3

ADC Interfacing

-> ADC - Analog to digital converter interfacing plays a very essential part in many embedded projects.

-> ADC used 0804 ADC, before going through the interfacing procedure we must neatly understand how ADC 0804 works.

-> ADC 0804

-> ADC0804 is ~~the~~ s an 8-bit sucessive approximation analogue to digital converter. from National semiconductor.

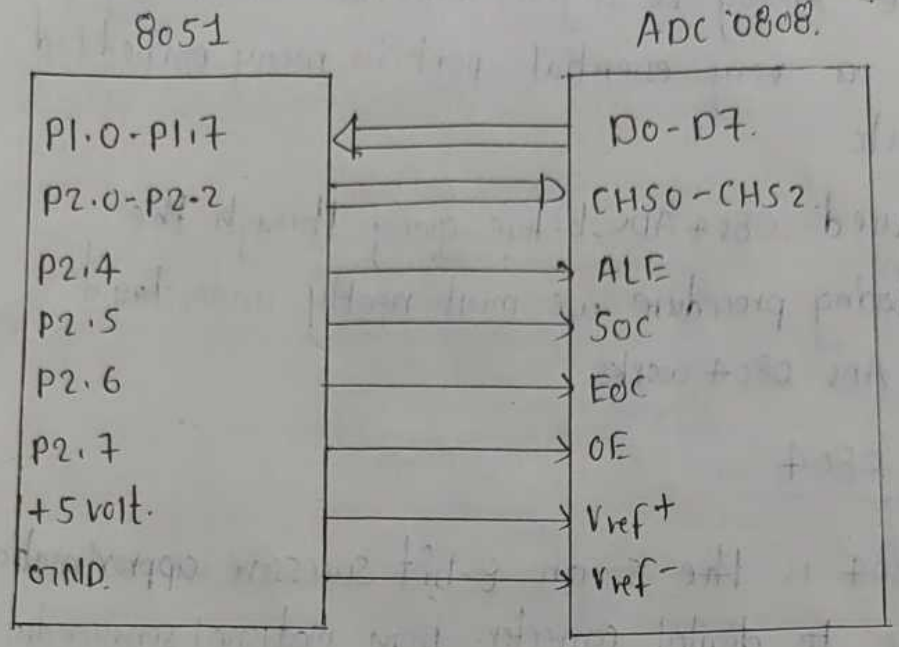
-> Features of ADC0804.

- Differential analogue voltage inputs.
- Input voltage range is 0-5v.
- No zero adjustment.
- Built in clock generator.

-> The pin out diagram of ADC0804 is shown in below fig.

-> ADC0804 is consist of 20 pins.

ADC Interfacing with 8051



-> Steps to convert analogue input

-> Make CS=0 and send a low to high pulse to WR to start conversion.

-> Now keep checking INTR pin.

- INTR will be 1 if conversion is not finished.
- INTR will be 0 if " " finished.

-> Make CS=0 and send a high to low pulse to RD to read data from ADC.

DAC Interfacing

-> DAC - Digital to analog converter interfacing.

-> Digital to analog converter (DAC) is a device that is widely used for converting digital pulses into analog pulses.

-> There are two methods for converting DAC.

-> They are binary weighted method and R/2R ladder method.

-> But we will use DAC0808 digital to analog converter.

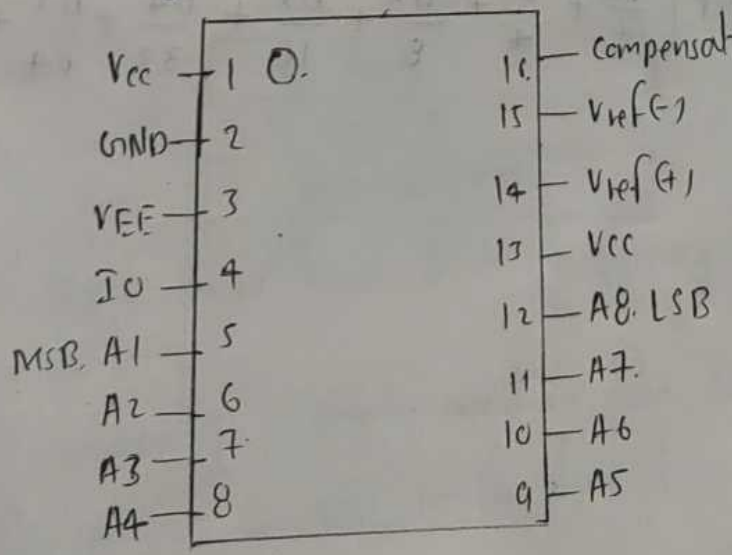
-> DAC 0808.

-> DAC0808 is an 8-bit sucessive approximation digital to analogue converter from National semiconductors.

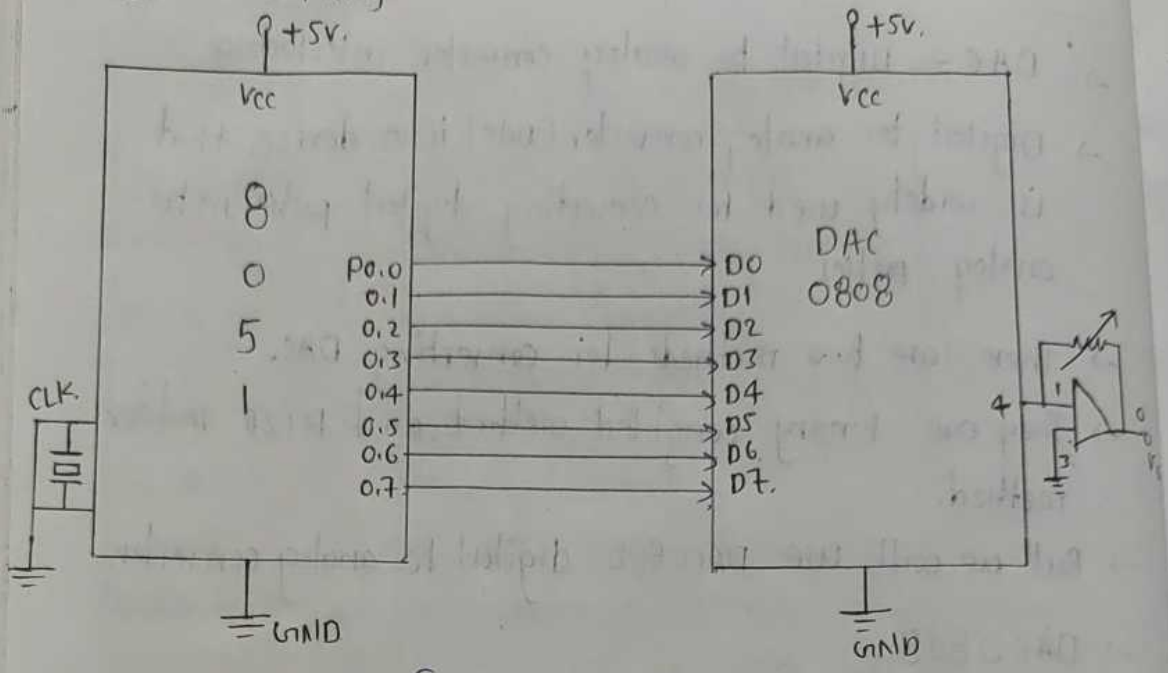
-> Features of DAC0808.

- 8-bit parallel digital data input.
- Input voltage range ±4.5V to ±18V.
- Low power consumption.
- More ↓ Dissipation.

-> pin out diagram of DAC0808 is shown in below figure.



→ DAC Interfacing



-> Block diagram of DAC interfacing DAC shown in above figure.

+ Input supply for Both 8051 and DAC 0808 is +5v.

-> 8051 consist of clock signal and pins from P0.0 - 0.7.

-> DAC0808 consist of pins from D0 - D7.

-> opam is used fro for converting current signal into voltage signal.

$$V_o = V_{ref}\left[\frac{D0}{2} + \frac{D1}{4} + \frac{D2}{8} + \frac{D3}{16} + \frac{D4}{32} + \frac{D5}{64} + \frac{D6}{128} + \frac{D7}{256}\right]$$

Sensor interfacing

-> LM35 temperature sensors:

-> LM35 sensors are precision integrated circuit-temperature sensors, whose output voltage is linearly proportional to celsium temperature.

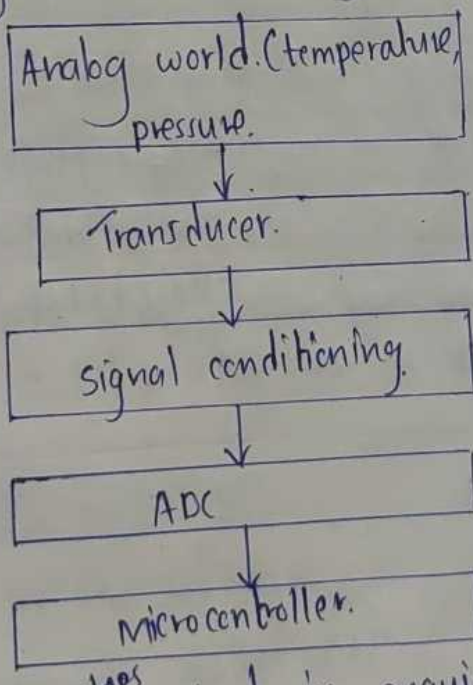
-> LM35 does not require external caliberation, since it is internally caliberated.

-> LM34 sensors are precision integrated circuit-temperature sensors, whose output voltage is linearly proportional to Fahrenheit temperature.

-> It is also internally caliberated.

-> Signal conditioning and LM35 to the 8051

Getting data from analog world.



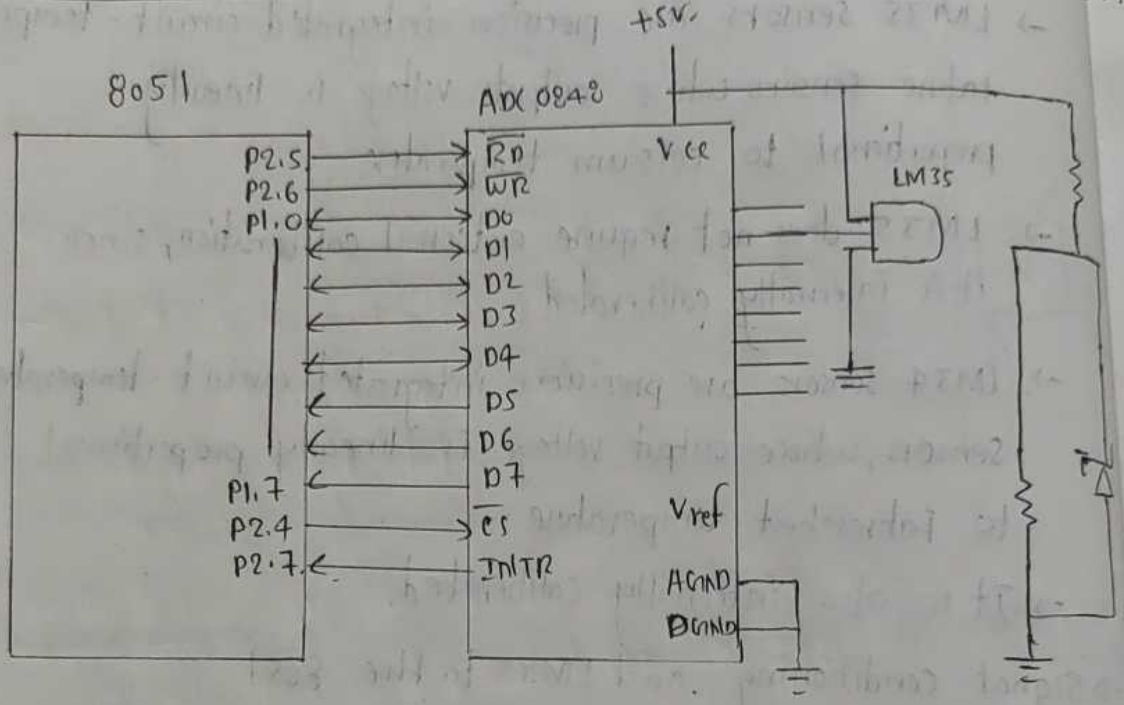
-> The above figure shows steps involved in acquiring data from analog world.

-> signal conditioning is widely used in data acquisition.

-> Transducer produce an o/p in the form of voltage, charge capacitance, resistance.

→. The conversion is commonly called signal conditioning.

→ Block diagram



Serial-port

→. The seria
can tra

→ pins us

→ Registers
i sco

i SCON

→ SCON
Sele

MSB 7

SM0	

SM0	SM
0	
0	
1	
1	

SM

RI

T

Serial-port programming

→ The serial port of 8051 is full duplex, i.e it can transmit and receive data simultaneously

→ pins used for serial communication.
TXD and RXD in port 3.

→ Registers used in serial communication.
i SCON ii SBUF iii PCON

i SCON Register : SCON - serial control register.

→ SCON register is an 8-bit register used to select the programmable mode of 8051.

MSB 7 6 5 4 3 2 1 0 LSB

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0	SM1	programmable mode
0	0	Mode0 : Shift register.
0	1	Mode1 : 8-bit UART
1	0	Mode2 : 9-bit UART
1	1	Mode3 : 9-bit UART.

SM2 : → Serial mode control bit.
→ It is set by the software.

REN : → Receiver enable control bit.
→ To enable/disable serial data reception.

TB8 : → Transmit Bit-8.
→ It is set by hardware to determine 9th bit transmitted in 9-bit UART.

RB8 : → It is set by hardware to indicate the 9th bit received.
→ Received Bit-8.

TI : Transmit Interrupt flag.

RI : Receive Interrupt flag.

operating modes of serial port

→ i. Mode 0 : In this mode serial data enters and exits through TXD/RXD.
→ It uses 8-bit data.
→ Baud rate $\frac{1}{12}$.

ii Mode 1 : In this mode 10 bits are transmitted through TxD and received through RxD.
→ Baud rate is variable.

iii. Mode 2 : In this mode 11 bits are transmitted through TxD and received through RxD.
→ Baud rate is $\frac{1}{32}$ (or) $\frac{1}{64}$.

iv Mode 3 : It is similar to mode 2.
→ 11 bits are transmitted and received.
→ Baud rate is variable.