4. 8086 Microprocessor

* Write short note on Addressing Mode in 8086
→ The different ways in which a source operand is denoted in an instruction is known as addressing modes.
There are 5 different addressing modes in 8086 programming -

1. Immediate addressing mode
The addressing mode in which data operand is a part of instruction itself is knownas immediate addressing mode.
Example:
1. MOV CX, 4929 H, 2. ADD AX, 2387 H
3. MOV AL, FFH

2. Register addressing mode
It means that the registers is the source of an operand for an instruction
Example:
MOV CX, AX ; copies the contents of the 16 bit AX register into ~~the 16 bit~~ CX register
ADD BX, AX

3. Direct addressing mode:
The addressing mode in which the effective address of the memory location is written directly in the instruction.
MOV AX, [1592H], MOV AL, [0300H]

4. Register indirect addressing mode

This addressing mode allows data to be addressed at any memory location through an offset address held in any of the following registers: BP, BX, DI & SI

Example

MOV AX, [BX] ; Suppose the register BX contains 4895H, then the contents of 4895H are moved to AX

ADD CX, [BX]

5. Based addressing mode

In this addressing mode, the offset address of the operand is given by the sum of contents of the BX/BP registers and 8 bit / 16 bit displacement

Example:

MOV DX, [BX+04]

ADD CL, [BX+08]

6. Indexed addressing mode

In this addressing mode, the operands offset address is found by adding the contents of SI or DI register and 8 bit / 16 bit displacements

Example

MOV BX, [SI+16]

ADD AL, [DI+16]

7. Based index addressing mode

In this addressing mode, the offset address of the operand is computed by summing the base register to the contents of an Index register.

Page No. Date

Example: AD CX, [AX+SI],
MOV Ax, [AX+DI]

8. Based indexed with displacement mode
In this addressing mode, the operand offset is computed by adding the base register contents. An Index registers contents and 8 or 16 bit displacement
Example
MOV AX, [BX+DI+08]
ADD CX, [BX+SI+16]

8086 Instructions:

1. Data Transfer Instructions:
These instruction are used to transfer the data from the source operand to the destination operand.
Following are the list of instructions under this group:

Instruction to transfer a word

1) MOV - Used to copy the byte or word from the provided source to the provided destinatio

2) PPUSH - Used to put a word at the top of the stack

3) Pop - Used to get a word from the top of the stack to the provided location

4) PUSHA - Used to put all the registers into the stack.

5) XCHG - Used to exchange the data from two location.

6) XLAT - Used to translate a byte in AL using a table in the memory.

Instructions for input and output port trans

1) IN - Used to read a byte or word from the provided port to the accumulator

2) OUT - Used to send out a byte or word from the accumulator to the provided port.

Instructions to transfer the address

1) LEA - Used to load the address of operand into the provided register

2) LDS - Used to load DS register and other provide register from the memory.

3) LES - Used to load ES register and other provided register from the memory.

Instruction to transfer flag register:

1) LAHF - Used to load AH with the low byte of the flag register.

2) SAHF - Used to store AH register to low byte of the flag register.

3) PUSHF - Used to copy the flag register at the top of the stack.

4) POPF - Used to copy a word at the top of the stack to other flag register

Page No. Date

Arithmetic Instruction

These instructions are used to perform arithmetic operations like addition, subtraction multiplication, division etc.

* Instructions to perform addition

- ADD - Used to add the provided byte to byte word to word
- ADC - Used to add with cary
- INC - Used to increment the provided byte/ word by 1.
- AAA - Used to adjust ASCII after addition
- DAA - Used to adjust the decimal addit^{n}

* Instructions to perform subtraction

- SUB - Used to subtract the byte/word from word
- SBB - Used to perfom subtraction with borrow
- DEC - Used to decrement the provided byte/word
- NPG - Used to negate each bit of the provided byte/word and add 1/2's complement
- CMP - Used to compare 2 provided byte/word
- AAS - Used to adjust ASCII codes after subtract
- DAS - Used to decimal after subtraction

* Instruction to perform multiplication

- MUL - Used to multiply unsigned byte by byte
- IMUL - Used to multiply signed byte by byte
- AAM - Used to adjust ASCII codes after multiplication

Instructions to perform division

- DIV - Used to divide the unsigned word by byte or unsigned double word by word.
- IDIV - Used to divide the signed word by byte or signed double word by word.
- AAD - Used to adjust ASCII codes after division
- CSW - Used to fill the upper byte of the word with the copies of sign bit of the lower byte
- CWD - Used to fill the upper word of the double word with the sign bit of the lower word.

*** Program Execution Transfer Instructions
Branched Loop Instructions

These instruction are used to transfer / branch the instructions during an execution
It includes the following instructions

* Instructions to transfer the instruction during an execution without any condition

1) CALL - Used to call a procedure and save their return address to the stack.

2) RET - Used to return from the procedure to the main program.

3) JMP - Used to jump if below to the provided address to proceed to the next instruction

* Instructions to transfer the instruction during an execution with some condition

4) JA/JNBE - Used to jump if above/not below/equal instruction satisfies.

5) JAE/JNB - Used to jump if above/ not below instruction satisfies.

6) JBE/JNB - Used to jump if below/equal/not above instruction statisfies

7) JC - Used to jump if carry flag CF = 1

8) JE/JZ - Used to jump if equal/zero flag ZF = 1

9) JNC - Used to jump if no carry flag CF = 0

10) JNE/JNZ - Used to jump if not equal/ zero flag ZF = 0

** Programming model
=> General purpose register in 8086