Tutorial – 3 (Assembly Language Programming)

Systems Programming Lab (CS-244)

Dr. Samit Bhattacharya

Associate Professor

Department of Computer Science & Engineering

Indian Institute of Technology Guwahati, Assam.

Instruction Set of 8086

- An instruction is a binary pattern designed inside a microprocessor to perform a specific function.
- The entire group of instructions that a microprocessor supports is called **Instruction Set**.

Opcode and Operand in Microprocessor 8086

- An opcode is a short off “operation code”
- An opcode is a single instruction that can be executed by the CPU. In machine language, it is a binary or hexadecimal value such as B7 loaded into the instruction register.
- In assembly language, mnemonic form of an opcode is a command such as MOV, ADD or JMP.
- **Example:**
  MOV AX, 1000H -> MOV is the opcode and AX (register) is an operand.

An addressing mode means the method by which an operand can be specified in a register or a memory location.

Addressing Modes of 8086:

(1) Register Addressing
(2) Immediate Addressing
(3) Direct Addressing
(4) Register indirect addressing mode
(5) Based addressing mode
(6) Indexed addressing mode
(7) Based-index addressing mode
(8) Based indexed with displacement mode

We will discuss only Register, Immediate and Direct Addressing modes.

Register Addressing Mode

- In this mode, operands are specified using registers. Registers may be used as source operands, destination operands or both.
- This addressing mode is normally preferred because the instructions are compact and fastest executing of all instruction.
- Example:

  MOV BX, DX ; copies the contents of DX into BX

  ADD AL, BH ; adds the contents of BH to contents of AL
- Source and destination registers must have the same size

Immediate Addressing Mode

- In this mode, the operand is specified in the instruction itself.
- Instructions are longer but the operands are easily identified. Example:
- MOV CL, 12H
- This instruction moves 12 immediately into CL register. CL ← 12H

Direct Addressing Mode

- In this mode, address of the operand is directly specified in the instruction.
- Example: MOV AX, [4321H]

  This instruction moves data from location 4321H into register AX.

Classification of Instruction Set

- Data Transfer Instructions
- Arithmetic Instructions
- Bit Manipulation Instructions
- Program Execution Transfer Instructions
- String Instructions
- Processor Control Instructions
- Iteration Control Instructions
- Interrupt Instructions

Data Transfer Instructions

- Used to transfer data from source to destination.
- The operand can be a constant, memory location, register or I/O port address.

1) MOV Des, Src:

- Src operand can be register, memory location or immediate operand.
- Des can be register or memory operand.
- Both Src and Des cannot be memory location at the same time.
- E.g.: MOV CX, 037A H MOV AL, BL MOV BX, [0301 H]

2) PUSH Operand:

- It pushes the operand into top of stack.
- E.g.: PUSH BX

3) POP Des:

- It pops the operand from top of stack to Des.
- E.g.: POP AX

Arithmetic Instructions

1) ADD Des, Src:

- It adds a byte to byte or a word to word.
- It affects flag register.
- E.g.: ADD AL, 74H ADD DX, AX

2) SUB Des, Src:

- It subtracts a byte from byte or a word from word.
- It affects flag register. For subtraction, CF acts as borrow flag.
- E.g.: SUB AL, 74H SUB DX, AX

3) MUL:

- Multiplicand is in AX. We have to mention only multiplier.
- E.g.: MUL BX

4) INC Src:

- It increments the byte or word by one. The operand can be a register or memory location.
- It affects flag register.
- E.g.: INC AX

5) DEC Src:

- It decrements the byte or word by one.
- E.g.: DEC AX

6) CMP Des, Src:

- It compares two specified bytes or words. The Src and Des can be a constant, register or memory location.
- The value of source and destination does not change, but the flags are modified to indicate the result.

7) DIV Src:

- Dividend is in AX. We have to mention only divisor.
- E.g.: DIV BX

Bit Manipulation Instructions

1) NOT Src:

- Inverts each bit of the operand.
- E.g.: NOT AL (Initially AL = 0110 1100. After execution, AL = 1001 0011)

2) AND Des, Src:

- Logically ANDs each bit of the source byte or word with the corresponding bit in the destination and stores result in the destination.
- E.g.: AND AL, BL

3) OR Des, Src:

- Logically ORs each bit of the source byte or word with the corresponding bit in the destination and stores result in the destination.
- E.g.: OR AL, BL

3) XOR Des, Src:

- Logically XORs each bit of the source byte or word with the corresponding bit in the destination and stores result in the destination.

4) SHL op1, op2:

- Shifts op1 Left. The number of shifts is set by op2.
- E.g.: MOV AL, 11100000b SHL AL, 1 ; AL = 11000000b, CF=1.

5) SHR op1, op2:

- Shifts op1 Right. The number of shifts is set by op2.
- E.g.: MOV AL, 00000111b SHR AL, 1 ; AL = 00000011b, CF=1.

6) SAR op1, op2:

- Shifts op1 right. The number of shifts is set by op2. It copies the old MSB into the new MSB.
- E.g.: MOV AL, 11100000b SAR AL, 1 ; AL = 11110000b.

7) ROL – Used to rotate bits of byte/word towards the left, i.e. MSB to LSB and to Carry Flag [CF].

8) ROR – Used to rotate bits of byte/word towards the right, i.e. LSB to MSB and to Carry Flag [CF].

9) RCR – Used to rotate bits of byte/word towards the right, i.e. LSB to CF and CF to MSB.

10) RCL – Used to rotate bits of byte/word towards the left, i.e. MSB to CF and CF to LSB.

Program Execution Transfer Instructions

1) JMP Address/Label: Unconditional Jump. Transfers control to another part of the program. E.g.: JMP label1

2) JNE/JNZ – Used to jump if not equal/zero flag ZF = 0

3) JE/JZ – Used to jump if equal/zero flag ZF = 1

4) JG/JNLE – Used to jump if greater/not less than/equal instruction satisfies.

5) JGE/JNL – Used to jump if greater than/equal/not less than instruction satisfies.

6) JL/JNGE – Used to jump if less than/not greater than/equal instruction satisfies.

7) JLE/JNG – Used to jump if less than/equal/if not greater than instruction satisfies.

8) JC – Used to jump if CF=1. **9)** JNC – Used to jump if CF not equal to one.

Processor Control Instructions

1) STC – Used to set carry flag CF to 1

2) CLC – Used to clear/reset carry flag CF to 0

Iteration Control Instructions

1) LOOP – Used to loop a group of instructions until the condition satisfies, i.e., CX = 0

Thank you for your presence !

QUESTIONS …

samit@iitg.ac.in

http://www.iitg.ac.in/samit/