# Chapter 6

CSE-214

# Flow Control Instructions

- In assembly “Jump” and “Loop” instructions transfer control to another part of program
- This transfer can be conditional or unconditional
- In case of conditional transfer the transfer depends on the state of the FLAGS Register

# Unconditional Jump

- Syntax:
  - JMP        destination_label
- This instruction is similar to the goto instruction found in C

# Unconditional Jump

```
Lable1:
   x=3;
   y=4;
   goto Label1;
```

C

```
Label1:
   MOV x,3
   MOV y,4
   JMP Label1
```

Assembly

# Conditional Jumps

- Syntax:
  - Jxxx destination_label
- Equivalent to decision making instructions of high level languages
- Example:
  - JNZ Label1
- Range:
  - Destination label must precede the jump instruction by no more than 126 bytes, or follow it by no more than 127 bytes

# Signed and Unsigned Jumps

- Sign jumps consider MSB as the sign bit whereas unsigned jumps do not
- Condition are often provided by CMP instruction:
- Syntax:
  - CMP destination, source
- CMP is same as SUB except the fact that the result is not stored.
- Example:
  - CMP AX, BX
    JG BELOW
- Different types:
  - JG/JNLE and JA/JNBE
  - JLE/JNG and JBE/JNA
  - Others

# IF-THEN

```
IF AX<0
  THEN
        replace AX by -AX
END_IF
```

Pseudocode

```
    CMP AX,0
    JNL END_IF
    NEG AX
END_IF:
```

Assembly

# IF-THEN-ELSE

```
IF AL<=BL
  THEN
        display AL
ELSE
        display BL
END_IF
```

Pseudocode

```
   CMP AL,BL
   JNLE ELSE
   MOV DL, AL
   JMP DISPLAY
ELSE:
   MOV DL, BL
DISPLAY:
   MOV AH, 2
   INT 21H
```

Assembly

# ELSE-IF

```
IF AL<=BL
   THEN
          display AL
ELSE IF AL  <=  BH
   THEN
          display BH
END_IF
```

Pseudocode

```
   CMP AL,BL
   JNLE ELSE_IF
   MOV DL, AL
   JMP DISPLAY
ELSE_IF:
   CMP AL,BH
   JNLE END_IF
   MOV DL,BH
DISPLAY:
   MOV AH, 2
   INT 21H
END_IF:
```

Assembly

# AND Condition

```
IF AL<BL && AL<BH
  THEN
      DISPLAY AL
END_IF
```

Pseudocode

```
   CMP AL,BL
   JNL END_IF
   CMP AL,BH
   JNL END_IF
   MOV AH,2
   MOV DL,AL
   INT 21H
 END_IF:
```

Assembly

# OR Condition

```
IF AL>BL || AL>BH
  THEN
      ADD AL,5
END_IF
```

Pseudocode

```
    CMP AL,BL
    JG TASK
    CMP AL,BH
    JG TASK
    JMP END_IF
TASK:
    ADD AL,5
END_IF:
```

Assembly

# CASE

```
CASE AX
   <0: PUT -1 IN BX
   =0: PUT 0 IN BX
   >0: PUT 1 IN BX
END_CASE
```

Pseudocode

```
   CMP AX,0
   JL NEGATIVE
   JE ZERO
   JG POSITIVE
NEGATIVE:
   MOV BX,-1
   JMP END_CASE
ZERO:
   MOV BX,0
   JMP END_CASE
POSITIVE:
   MOV BX,1
END_CASE:
```

Assembly

# Loop Instruction

- For loop structures the ‘LOOP’ instruction is used. Syntax:

  ```
  LOOP     destination_label
  ```

- Decrements value of CX
- Checks if value of CX is zero
- If not then jumps to destination_label
- Otherwise does nothing

# For Loop

```
FOR 80 times DO
   display ‘*’
END_FOR:
```

Pseudocode

```
   MOV CX, 80
   MOV AH,2
   MOV DL, ‘*’
TOP:
   INT 21H
   LOOP TOP
```

Assembly

# While Loop

```
initialize count to 0
read a character
WHILE character !='$'
   count=count+1
   read a character
END_WHILE
```

Pseudocode

```
    MOV DX,0
    MOV AH,1
    INT 21H
WHILE:
    CMP AL, '$'
    JE END_WHILE
    INC DX
    INT 21H
    JMP WHILE
END_WHILE:
```

Assembly

# Repeat Loop

```
REPEAT
   read a character
UNTIL character is blank
```

Pseudocode

```
   MOV AH,1
REPEAT:
   INT 21H
   CMP AL, ‘ ’
   JNE REPEAT
```

Assembly

# THANK YOU