MODUL IV

LOGIC, SHIFT, and ROTATE INSTRUCTIONS

Untuk mengubah bit-bit secara individual dalam komputer maka menggunakan operasi logika. Nila biner dari 0 dan 1 diperlakukan sebagai salah (0) dan benar (1). Tabel kebenaran dari operasi logika AND, OR, XOR (exclusive OR) dan NOT diberikan sbb:

a	b	a AND b	a OR b	a XOR b
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

a	NOT a
0	1
1	0

Instruksi **AND, OR** dan **XOR** melakukan operasi sesuai dengan namanya. Formatnya:

```
AND   tujuan,sumber
OR    tujuan,sumber
XOR   tujuan,sumber
```

Instruksi **NOT** melakukan operasi komplemen-satu pada *tujuan*. Formatnya adalah:

```
NOT   tujuan
```

Instruksi **TEST** melakukan operasi AND pada ***tujuan*** dengan ***sumber*** tetapi tidak mengubah isi ***tujuan*****.** Tujuan instruksi TEST adalah untuk men-set status flag. Formatnya adalah:

```
TEST tujuan,sumber
```

Perintah Test digunakan untuk mengetahui nilai pada suatu bit, dengan

syntax :

Perintah test akan meng**AND** kedua nilai operand, tetapi hasil yang didapatkan tidak akan berpengaruh terhadap nilai kedua operand tersebut. Setelah perintah Test dilaksanakan yang akan

terpengaruh adalah Flags, sehingga perintah ini sering diikuti dengan perintah yang berhubungan dengan kondisi flags. Adapun flags yang terpengaruh adalah CF,OF,PF,ZF,SF dan AF.

```
TEST AX,0Fh
JNZ Proses ; Lompat jika Zerro flag 0
```

Pada perintah diatas komputer akan menuju ke label Proses bila ada satu bit atau lebih dari AX yang sama dengan 0Fh. Bila diikuti dengan perintah **JC Proses**, maka komputer akan menuju ke label proses bila keempat byte rendah pada AL semuanya 1(?F).

Instruksi **SHL** (shift left) melakukan penggeseran bit-bit ke kiri pada operand ***tujuan*** (*destination operand*). Format untuk pergeseran tunggal adalah:

```
SHL  tujuan,1
```

Berikut ialah ilustrasi penggunaan instruksi SHR dalam melakukan pergeseran Bit

Instruksi : **MOV AX,3Fh**

```
MOV CL,3
SHL AX,CL ; Geser 3 bit ke kiri
```

Akan menghasilkan nilai F8h pada register AX. Operasi detilnya dapat dilihat di bawah ini.

3Fh : 0011 1111

SHL 1 : 0111 1110 (=7Eh)

SHL 2 : 1111 1100 (=FCh)

SHL 3 : 1111 1000 (=F8h)

Instruksi **SHR** (shift right) melakukan penggeseran bit-bit ke kanan pada operand ***tujuan*** (*destination operand*). Format untuk pergeseran tunggal adalah:

```
SHR  tujuan,1
```

Berikut ialah ilustrasi penggunaan instruksi SHR dalam melakukan pergeseran Bit

Instruksi :

```
MOV AX,3Fh
MOV CL,3
SHR AX,CL ; Geser 3 bit ke kanan
```

Akan menghasilkan nilai 07h pada register AX. Operasi detilnya dapat dilihat

di bawah ini.

3Fh : 0011 1111

SHR 1 : 0001 1111 (=1Fh)

SHR 2 : 0000 1111 (=0Fh)

SHR 3 : 0000 0111 (=07h)

Instruksi **ROL** (*rotate left*) menggeser bit-bit ke kiri. MSB digeser ke bit paling kanan. Flag CF juga menerima hasil pergeseran dari bit MSB. Sintaksnya adalah:

```
ROL  tujuan,1
```

Instruksi **ROR** (*rotate right*) bekerja seperti ROL, kecuali bahwa bit-bitnya dirotasi ke kanan. Bit paling kanan digeser ke MSB. Dan juga ke CF (perhatikan gambar). Sintaksnya adalah:

```
ROR  tujuan,1
```

Contoh 1 :

Kalikan AX dengan 10 (1010)

```
SHL AX,1 ; AX kali 2
MOV BX,AX
SHL AX,2 ; AX kali 8
ADD AX,BX
```

Kalikan AX dengan 18 (10010)

```
SHL AX,1 ; AX kali 2
MOV BX,AX
SHL AX,3 ; AX kali 16
ADD AX,BX ; 18 kali AX
```

Kalikan AX dengan 5 (101)

```
MOV BX,AX
SHL AX,1 ;  AX kali 2
SHL AX,1 ; AX kali 4
ADD AX,BX ; 5 kali AX
```

Contoh 2 :

Menghitung jumlah bit 1 pada BX Dengan menggunakan instruksi ROL, hasil Simpan di AX

```
XOR AX,AX ; perhitungan bit di AX
MOV CX,16 ; pencacah 16 bit
 ULANG :
    ROL BX,1 ; CF = rotasi Bit
    JNC NEXT ; 0 Bit
    INC AX ; 1 Bit, increment total
NEXT:
    LOOP ULANG ; terus ulang sampai 16x
```

Tugas Pendahuluan

1. a) Jelaskan Instruksi **RCR** dan **RCL**

   b) jelaskan dan gambarkan melalui ilustrasi contoh perbedaan antara instruksi **TEST** dan **AND, NOT dan NEG**

2. apabila diketahui nilai register AL = EAH, CL =2 dan carry flag dianggap telah di *clear*, jabarkan instruksi dibawah ini melalui gambar diagram pergeseran bit hingga menghasilkan nilai yang dimaksud.

   A) SHL AL,1

   B) SHL AL,CL

   C) SHR AL,1

   D) SAR AL,1

   E) ROL AL,1

   F) ROL AL,CL

   G) ROR AL,1

   H) RCL AL,1

   I) RCR AL,1

   J) RCR AL,CL

3. Buatlah Program dibawah ini dengan memanfaatkan instruksi logika dan pergeseran bit.

   ```
   masukkan sebuah karakter  : B

   Kode ASCII B dalam Heksadesimal adalah 42
   ```

Latihan

1. Algoritma untuk Input Biner

   Clear BX /* BX akan menyimpan nilai biner */

   Input sebuah karakter /* '0' atau '1' */

   WHILE karakter <> Carriage return DO

   Ubah karakter ke nilai biner

   Geser ke kiri BX

   Sisip nilai ke dalam LSB dari BX

   Input sebuah karakter

   END_WHILE

   Bahasa Assembly :

```
ORG 100H
     XOR  BX,BX      ; clear BX
     MOV  AH,1       ; input character function
     INT  21h
WHILE_:
     CMP  AL,0Dh     ; CR ?
     JE   END_WHILE      ; ya, kerjakan
     AND  AL,0Fh     ; bukan, ubah ke nilai biner
     SHL  BX,1       ; buat ruang untuk nilai yang baru
     OR   BL,AL          ; simpan nilai kedalam BX
     INT  21h        ; baca sebuah karakter
     JMP  WHILE_     ; loop kembali
END_WHILE:
```

2. Algoritma untuk input hexadecimal

   Clear BX /* BX akan menyimpan nilai masukan */

   Input karakter hexadecima

   WHILE karakter <> Carriage return DO

   Ubah karakter ke nilai biner

   Geser ke kiri BX empat kali

   Sisip nilai ke dalam BX 4 bit bagian bawah (lower bit)

   Input sebuah karakter

   END_WHILE

   Bahasa Assembly :

```
ORG 100H
XOR  BX,BX      ; clear BX
MOV  CL,4       ; pencacah untuk pergeseran 4 kali
MOV  AH,1
INT  21h
WHILE_:
     CMP  AL,0Dh     ; CR ?
     JE   END_WHILE      ; ya, kerjakan
                         ; Ubah karakter ke nilai biner
     CMP  AL,39h     ; sebuah digit ?
     JG   HURUF      ; bukan, sebuah huruf
                     ; masukan adalah sebuah digit
     AND  AL,0Fh     ; ubah digit ke nilai biner
     JMP  GESER      ;
HURUF:
     SUB  AL,37h     ; ubah huruf menjadi nilai biner
GESER:
     SHL  BX,CL      ; buat ruang untuk nilai yang baru
                     ; sisip nilai ke dalam BX
     OR   BL,AL      ; Sisip nilai ke dalam BX 4 bit bagian
                       bawah
     INT  21h        ; input sebuah karakter
     JMP  WHILE_     ; loop hingga CR
END_WHILE:
```

Tugas Praktikum

1. Buatlah program untuk menginputkan bilangan heksadesimal (0 s.d. 9 dan A s.d. F), kemudian menampilkannya dalam bentuk biner 8 bit.

   Contoh Output

   ```
   Input heksadesimal  : 9
   Output biner        : 00001001
   ```

2. Buatlah program untuk menginputkan bilangan biner 8 bit, kemudian menampilkan bilangan tersebut dalan bentuk heksadesimal.

   Contoh Output

   ```
   Input biner         : 10100001
   Output heksadesimal : A1
   ```

3. Buatlah program untuk menginputkan beberapa nilai, kemudian menjumlahkan nilai tersebut dan menampilkan hasilnya dalam bentuk heksadesimal.

Contoh Output

```
Input bilangan      : 1320567
Jumlah              : 18H
```