MODUL I

PENGENALAN ASSEMBLY

Apakah bahasa assembly?

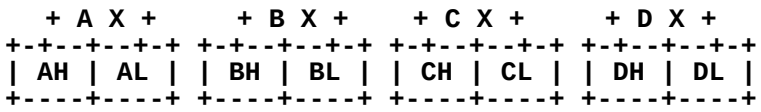
Bahasa assembly adalah bahasa pemrograman dengan korespondensi satu-satu antara perintah-perintah/pernyataannya dan bahasa mesin komputer. Bahasa assembly tidak satu jenis sebagaimana CPU komputer pun bermacam-macam. Setiap bahasa assembly secara langsung dipengaruhi oleh set instruksi mesin komputer dan arsitektur perangkat keras.

Register

Register merupakan lingkungan kerja khusus dalam CPU, dirancang untuk diakses pada kecepatan tinggi. Dalam melakukan pekerjaannya, CPU selalu menggunakan register sebagai perantaranya, jadi register dapat diibaratkan sebagai kaki tangannya mikroprosesor. Jenis – jenis register :

a. General purpose (Register data)

Yang termasuk kedalam register adalah AX, BX, CX, dan DX yang masing – masing terdiri atas 16 bit. Register – register ini dapat dipisah menjadi 2 bagian dimana masing – masing bagian terdiri atas 8 bit, yaitu AH, AL, BH, BL, CH, CL, DH, DL.



Secara umum register-register dalam kelompok ini dapat digunakan untuk berbagai keperluan, walaupun demikian ada pula penggunaan khusus dari masingmasing register ini yaitu :

Register **AX**, secara khusus digunakan pada operasi aritmatika terutama dalam operasi pembagian dan pengurangan.

Register **BX**, biasanya digunakan untuk menunjukkan suatu alamat offset dari suatu segmen.

Register **CX**, digunakan secara khusus pada operasi looping dimana register ini menentukan berapa banyaknya looping yang akan terjadi.

Register **DX**, digunakan untuk menampung sisa hasil pembagian 16 bit.

b. Segmen register

Register yang termasuk kedalam register ini adalah CS, DS, ES, SS

c. Index dan pointer register

Register yang termasuk kedalam register ini adalah IP, SP, SI, DI, BP

d. Flag register

Register ini merupakan register 16 bit untuk menunjukkan kondisi dari suatu keadaan dari hasil suatu operasi.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X	x	x	x	O	D	I	T	S	Z	X	A	x	P	x	C

O : Overflow
D : Direction
I : Interupt
T : Trap
S : Sign
Z : Zero
A : Auxiliary
P : Parity
C : Carry
x : tidak terdefinisi

Sintaks Bahasa Assembly

Statement

Statement terdiri dari paling banyak empay buah field yaitu :

NAME OPERATION OPERAND COMMENT

Contoh :

```
Mulai:  Mov cx, 5
        ORG 100h
        Inc AX          ; AX= AX+1
```

Data

Data yang di gunakan dalam bahasa assembly :

1. Data Biner, penulisan data biner harus diakhiri dengan huruf B.
   Contoh : 01101011B

2. Data Heksadesimal, penulisan data heksadesimal harus diakhiri dengan huruf H, bila data tersebut diawali dengan karakter A, B, C, D, E, F maka harus disisipkan angka 0 didepan data tersebut.
   Contoh : 43AFH 0B45H
3. Data Desimal, penulisan data decimal dapat diakhiri dengan huruf D atau tidak diberi tanda sama sekali.
   Contoh : 68D 68
4. Data Karakter, penulisan data karakter (string) harus diapit oleh tanda petik ganda atau tanda petik tunggal.
   Contoh : "TEKNIK" atau 'TEKNIK'

Variabel

Setiap variabel mempunyai tipe data dan oleh program ditetapkan alamat memorinya.

Pseudo-op	Arti
DB	Penetapan byte
DW	Penetapan word
DD	Penetapan doubleword (dua word berurutan)
DQ	Penetapan quadword (empat word berurutan)
DT	Penetapan tenword (sepuluh word berurutan)

Contoh :

```
Alpha db 4 ;  1 Byte nilai Awal 4 dengan Variable bernama Alpha
Bait db ?   ; 1 word tidak diketahui isinya Dengan variable bernama Bait
Wrd dw –2 ; 1 word nilai awal 0FFFE H
Array_data db 10h, 20h, 30h ; variable "Array_data" berukuran 3 byte dengan 3 elemen data
B DB 4,4,4,2,?          ; 1*5 byte, nilai awal=4,4,4,2,? Dengan variable bernama B
C DB 4 DUP(5)           ; 1*4 byte, nilai awal='5' Dengan variable bernama C
D DB 'TEKKOM'           ; 6 byte berisi 6 karakter Dengan variable bernama D
E DW ?                  ; 1 word tidak diketahui isinya Dengan variable bernama E
F DW ?,?,?              ; 3 word tidak diketahui isinya Dengan variable bernama F
```

```
G DW 10 DUP(?)          ;10 word tidak diketahui isinya Dengan variable bernama G
H DD ?                  ; 1 DoubleWord tanpa nilai awal Dengan variable bernama H
I DF ?,?                ; 2 FarWord tanpa nilai awal Dengan variable bernama I
J DQ 0A12h              ; 1 QuadWord, nilai awal='0A12' Dengan variable bernama J
K DT 25*80              ; 1 TenBytes, nilai awal='2000' Dengan variable bernama K
L EQU 666               ; Konstanta, L=666 Dengan variable bernama L
M DB '123'              ; String '123' Dengan variable bernama M
N DB '1','2','3'        ; String '123' Dengan variable bernama N
```

Instruksi Dasar

ORG 100H

Pada program COM perintah ini akan selalu digunakan. Perintah ini digunakan untuk memberitahukan assembler supaya program pada saat dijalankan (diload ke memory) ditaruh mulai pada offset ke 100h(256) byte. Dapat dikatakan juga bahwa kita menyediakan 100h byte kosong pada saat program dijalankan. 100h byte kosong ini nantinya akan ditempati oleh PSP(Program Segment Prefix) dari program tersebut. PSP ini digunakan oleh DOS untuk mengontrol jalannya program tersebut.

JMP

Perintah JMP(JUMP) ini digunakan untuk melompat menuju tempat yang ditunjukkan oleh perintah JUMP. Adapun syntaxnya adalah:

JUMP Tujuan .

Dimana tujuannya dapat berupa label seperti yang digunakan pada bagan diatas. Mengenai perintah JUMP ini akan kita bahas lebih lanjut nantinya. Perintah JUMP yang digunakan pada bagan diatas dimaksudkan agar melewati tempat data program, karena jika tidak ada perintah JUMP ini maka data program akan ikut dieksekusi sehingga kemungkinan besar akan menyebabkan program anda menjadi Hang.

MOV

Instruksi ini digunakan untuk mentransfer data antara register, antara register dengan lokasi memori atau mentransfer secara langsung suatu bilangan ke register atau lokasi memori.

Sintaks : **MOV tujuan, sumber**

XCHG

Instruksi XCHG (*exchange*) digunakan untuk mempertukarkan isi dua register, atau antara register dan lokasi memori.

Sintaks : **XCHG tujuan, sumber**

ADD, SUB, INC, dan DEC

Instruksi ADD dan SUB digunakan untuk menjumlahkan dan mengurangkan isi dua buah register dan suatu lokasi memori, atau menjumlahkan dan mengurangkan suatu bilangan ke/dari register atau lokasi memori. Sintaksnya sbb:

ADD tujuan, sumber

SUB tujuan, sumber

Instruksi INC (*increment*) digunakan untuk menambahkan 1 ke register atau lokasi memori, sedangkan DEC (*decrement*) digunakan untuk mengurangi 1 isi register atau lokasi memori. Sintaksnya :

INC tujuan

DEC tujuan

NEG

Instruksi NEG digunakan untuk membalik isi target yang dituju (dapat berupa register atau lokasi memori). NEG sebenarnya melakukan fungsi komplemen-2. Sintaksnya sbb:

NEG tujuan

INT 21H

Bila dihasilkan interupsi 21h apa yang akan dikerjakan oleh komputer ?. Jawabnya, ada banyak sekali kemungkinan. Pada saat terjadi interupsi 21h maka pertama-tama yang dilakukan komputer adalah melihat isi atau nilai apa yang terdapat pada register AH. Misalkan bila nilai AH adalah 2 maka komputer akan mencetak sebuah karakter, berdasarkan kode ASCII yang terdapat pada register DL. Bila nilai pada register AH bukanlah 2, pada saat dilakukan interupsi

21h maka yang dikerjakaan oleh komputer akan lain lagi. Dengan demikian kita bisa mencetak sebuah karakter yang diinginkan dengan meletakkan angka 2 pada register AH dan meletakkan kode ASCII dari karakter yang ingin dicetak pada register DL sebelum menghasilkan interupsi INT 21h

Contoh :

```
ORG 100h
Proses :
MOV AH,02h ; Nilai servis ntuk mencetak karakter
MOV DL,'A' ; DL = Karakter ASCII yang akan dicetak
INT 21h ; Cetak karakter !!
RET
END.
```

Insruksi Input dan Output

Single-key input

Input: AH = 1

Output: AL = Kode ASCII jika tombol karakter ditekan

Display a character or execute a control function

Input: AH = 2
DL = Kode ASCII dari karakter tampilan atau
karakter kontrol

Output: AL = Kode ASCII dari karakter tampilan atau
karakter kontrol
= 0 jika tombol non-karakter yang ditekan

MENCETAK BEBERAPA KARAKTER

Untuk mencetak beberapa karakter, bisa anda gunakan proses looping. Sebagai contoh dari penggunaan looping ini bisa dilihat pada program di bawah ini :

```
ORG 100h
Proses :
MOV AH,02h ; Nilai servis
MOV DL,'A' ; DL=karakter 'A' atau DL=41h
MOV CX,10h ; Banyaknya pengulangan yang akan
Ulang :
INT 21h ; Cetak karakter !!
```

```
INC DL ; Tambah DL dengan 1
LOOP Ulang ; Lompat ke Ulang
RET
END
```

Perintah **INC DL** akan menambah register DL dengan 1, seperti intruksi , DL:=DL+1 dalam Pascal.

Pengalamatan Tidak Langsung

Pengalamatan register tak langsung, memungkinkan data dialamatkan pada lokasi memori melalui alamat offset yang ditunjukkan oleh register berikut ini : BP, BX, DI, SI.

Symbol [ ] menjelaskan pengalamatan tidak langsung dalam bahasa rakitan, sebagai contoh jika register BX berisi 1000H maka instruksi MOV AX,[BX] akan menyebabkan isi segmen data yang berukuran word pada alamat yang ditunjukkan oleh nilai di register BX yakni alamat 1000H akan disalinkan ke register AX.

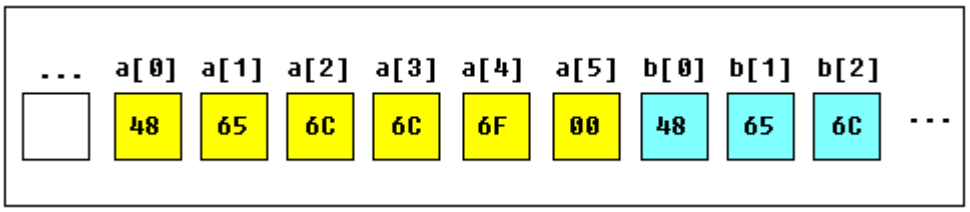
Contoh pengalamatan register Tak langsung :

Instruksi	Ukuran	Operasi
MOV CX,[BX]	16-Bit	Menyalin isi word lokasi memori segmen data yang dialamatkan oleh BX ke register CX
MOV [BP],DL	8-Bit	Menyalin DL ke lokasi memori segmen stack yang dialamatkan BP
MOV [DI],BH	8-bit	Menyalin BH ke lokasi segmen data yang dialamatkan oleh DI

Array (Larik)

Array dapat dilihat sebagai rentetan nilai atau karakter dalam suatu variable index. Array dapat memuat seluruh elemen yang termasuk dalam kode ASCII dengan range nilai (0..255). berikut contoh penerapannya dalam bahasa assembler.

```
a DB 48h, 65h, 6Ch, 6Ch, 6Fh, 00h
b DB 'Hello', 0
```



Array dapat diakses dengan menggunakan tanda kurung persegi seperti di bawah ini :

MOV AL, a[3]

Array juga dapat diakses dengan menggunakan register indeks, sperti : **BX, SI, DI, BP,** berikut contohnya :

MOV SI, 3

MOV AL, a[SI]

Apabila anda ingin mendeklarasikan sebuah array berukuran besar, maka syntax DUP dapat digunakan dengan format **VAR number DUP (value(s)),** berikut contoh penggunannya :

VAR : nama variable array

Number : jumlah duplikasi nilai yang akan dideklarasikan

Value(s) :ekspresi atau nilai yang akan diduplikasikan

`c DB 5 DUP(9)` akan sama dengan c DB 9, 9, 9, 9, 9

`d DB 5 DUP(1, 2)` akan sama dengan d DB 1, 2, 1, 2, 1, 2, 1, 2, 1, 2

berikut ialah contoh program untuk mengisi nilai pada 50 lokasi array secara looping, yakni :

```
TEKKOM DW 50 DUP (?)
ORG 100H
MOV AX,0
MOV DX,AX
MOV SI,OFFSET TEKKOM
MOV CX,50
ULANG:
    INC DX
    MOV AX,DX
    MOV TEKKOM[SI],AX
    INC SI
    LOOP ULANG
INT 20H
END
```

Program diatas akan mengisikan 50 elemen array variable TEKKOM dengan nilai dibawah ini

```
2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34,
36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62, 64, 66,
68, 70, 72, 74, 76, 78, 80, 82, 84, 86, 88, 90, 92, 94, 96, 98,
100
```

Latihan

1. Translasi bahasa tingkat tinggi ke bahasa assembly

Bahasa tingkat tinggi	Bahasa assembly
B = A	MOV AX, A MOV BX, AX

A = 5 - A	MOV AX, 5
	SUB AX, A
	MOV A, AX

A = B – 2 x A	MOV AX, B
	SUB AX, A
	SUB AX, A
	MOV A, AX

2.
```
ORG 100H
LEA BX, ARRAY_A
MOV AH, 02H
MOV DL, [BX+1]
INT 21H
EXIT:          RET
ARRAY_A DB 61H, 62H, 63H
```

3.
```
ORG 100H
MULAI:    MOV AH, 02H
          MOV DL, 'A'
          INT 21H
          MOV DL, 41H
          INT 21H
          MOV DL, DATA
          INT 21H
EXIT:          RET
DATA DB 'C'
```

4.
```
ORG 100H
MULAI:    MOV AH, 09H
          LEA DX, KAL1
          INT 21H
          MOV AH, 01H
          INT 21H
          MOV BL, AL
          MOV AH, 09H
          LEA DX, KAL2
          INT 21H
          ADD BL, 20h
```

```
          MOV AH, 02H
          MOV DL, BL
          INT 21H
EXIT:          RET
KAL1 DB 13,10, 'INPUT HURUF BESAR (A s/d Z) :  $'
KAL2 DB 13,10, 'OUTPUT :  $'
```

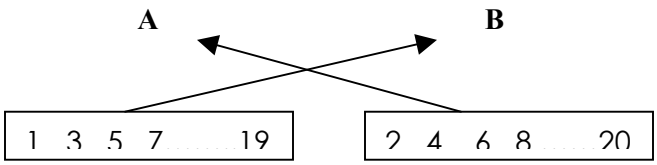
Tugas Pendahuluan

1. Sebutkan dan jelaskan jenis – jenis mode pengalamatan data pada 8086 prosessor dan berikan contoh instruksinya

2. Buat program dengan tampilan sebagai berikut :

```
masukkan angka (0-9) : 6
output
angka 6 telah anda input
```

3. Susun algoritma untuk **Tugas praktikkum** no.2 dan no.3
4. Buatlah program untuk menukar isi variable array A dengan variable array B. mula-mula variable A berisikan 10 buah bilangan ganjil pertama dan Variable B berisikan 10 buah bilangan genap pertama



Tugas Praktikum

1. Translasikan bahasa tingkat tinggi dibawah ini ke dalam bahasa assembly :

   a. A = B + A

   b. A = -A + 1

   c. C = 2 x B – 7

   d. C = (B + A) x 2

2. Buat program untuk menginputkan digit heksadesimal A s/d F, kemudian menampilkannya dalam bentuk desimal.

```
Output
 Input digit heksa (A s/d F)   : B
 Ouput desimal                 : 11
```

3. Buatlah program untuk menginputkan 3 buah nilai ke suatu array kemudian menampilkannya kembali.

```
Input Array          : ABC
A
B
C
```