# PERTEMUAN 11

# SET INSTRUKSI(1)

Set instruksi (instruction set) adalah sekumpulan lengkap instruksi yang dapat di mengerti oleh sebuah CPU, set instruksi sering juga disebut sebagai bahasa mesin (machine code), karna aslinya juga berbentuk biner kemudian dimengerti sebagai bahasa assembly, untuk konsumsi manusia (programmer), biasanya digunakan representasi yang lebih mudah dimengerti oleh manusia.

Sebuah instruksi terdiri dari sebuah opcode, biasanya bersama dengan beberapa informasi tambahan seperti darimana asal operand-operand dan kemana hasil-hasil akan ditempatkan. Subyek umum untuk menspesifikasikan di mana operand-operand berada (yaitu, alamat-alamatnya) disebut pengalamatan

Pada beberapa mesin, semua instruksi memiliki panjang yang sama, pada mesin-mesin yang lain mungkin terdapat banyak panjang berbeda. Instruksi-instruksi mungkin lebih pendek dari, memiliki panjang yang sama seperti, atau lebih panjang dari panjang word. Membuat semua instruksi memiliki panjang yang sama lebih muda dilakukan dan membuat pengkodean lebih mudah tetapi sering memboroskan ruang, karena semua instruksi dengan demikian harus sama panjang seperti instruksi yang paling panjang.

Di dalam sebuah instruksi terdapat beberapa elemen-elemen instruksi:

1. Operation code (op code)
2. Source operand reference
3. Result operand reference
4. Xext instruction preference

**Format instruksi (biner):**
Missal instruksi dengan 2 alamat operand : ADD A,B A dan B adalah suatu alamat register.

Beberapa simbolik instruksi:

ADD : Add (jumlahkan)

SUB : Subtract (Kurangkan)

MPY/MUL : Multiply (Kalikan)

DIV : Divide (Bagi)

LOAD : Load data dari register/memory

STOR : Simpan data ke register/memory

MOVE : pindahkan data dari satu tempat ke tempat lain

SHR : shift kanan data

SHL : shift kiri data .dan lain-lain

**Cakupan jenis instruksi:**
Data processing : Aritmetik (ADD, SUB, dsb); Logic (AND, OR, NOT, SHR, dsb); konversidata

Data storage (memory) : Transfer data (STOR, LOAD, MOVE, dsb)

Data movement : Input dan Output ke modul I/O

Program flow control : JUMP, HALT, dsb.

**Bentuk instruksi:**
– Format instruksi 3 alamat

Mempunyai bentuk umum seperti : [OPCODE][AH],[AO1],[AO2]. Terdiri dari satu alamt hasil, dan dua alamat operand, misal SUB Y,A,B Yang mempunyai arti dalam bentuk algoritmik : Y := A – B dan arti dalam bentuk penjelasan : kurangkan isi reg a dengan isi reg B, kemudian simpan hasilnya di reg Y. bentuk bentuk pada format ini tidak umum digunakan di dalam computer, tetapi tidak dimungkinkan ada pengunaanya, dalam peongoprasianya banyak register sekaligus dan program lebih pendek.

Contoh:
A, B, C, D, E, T, Y adalah register
Program: Y = (A – B) / ( C + D × E)
SUB Y, A, B Y := A – B
MPY T, D, E T := D × E
ADD T, T, C T := T + C
DIV Y, Y, T Y:= Y / T
Memerlukan 4 operasi

– Format instruksi 2 alamat

Mempunyai bentuk umum : [OPCODE][AH],[AO]. Terdiri dari satu alamat hasil merangkap operand, satu alamat operand, missal : SUB Y,B yang mempunyai arti dalam algoritmik : Y:= Y – B dan arti dalam bentuk penjelasan : kurangkan isi reg Y dengan isi reg B, kemudian simpan hasillnya di reg Y. bentuk bentuk format ini masih digunakan di computer sekarang,

untuk mengoprasikan lebih sedikit register, tapi panjang program tidak bertambah terlalu banyak.

Contoh :
A, B, C, D, E, T, Y adalah register
Program: Y = (A – B) / ( C + D × E)

| | |
|---|---|
| MOVE Y, A | Y := A |
| SUB Y, B | Y := Y – B |
| MOVE T, D | T := D |
| MPY T, E | T := T × E |
| ADD T, C | T := T + C |
| DIV Y, T | Y:= Y / T |

Memerlukan 6 operasi

– Format instruksi 1 alamat

Mempunyai bentuk umum : [OPCODE][AO]. Terdiri dari satu alamat operand, hasil disimpan di accumulator, missal : SUB B yang mempunyai arti dalam algoritmik : AC:= AC – B dan arti dalam bentuk penjelasan : kurangkan isi Acc dengan isi reg B, kemudian simpan hasillnya di reg Acc. bentuk bentuk format ini masih digunakan di computer jaman dahulu, untuk mengoprasikan di perlukan satu register, tapi panjang program semakin bertambah.

Contoh :
A, B, C, D, E, Y adalah register
Program: Y = (A – B) / ( C + D × E)

| | |
|---|---|
| LOAD D | AC := D |
| MPY E | AC := AC × E |
| ADD C | AC := AC + C |
| STOR Y | Y := AC |
| LOAD A | AC := A |
| SUB B | AC := AC – B |
| DIV Y | AC := AC / Y |
| STOR Y | Y := AC |

Memerlukan 8 operasi

– Format instruksi 0 alamat

Mempunyai bentuk umum : [OPCODE]. Terdiri dari semua alamat operand implicit, disimpan dalam bentuk stack. Operasi yang biasanya membutuhkan 2 operand, akan mengambil isi stack paling atas dan dibawahnya missal : SUB yang mempunyai arti dalam algoritmik : S[top]:=S[top-1]-S[top] dan arti dalam bentuk penjelasan : kurangkan isi stack no2 dari atas dengan isi stack paling atas, kemudian simpan hasilnya di stack paling atas, untuk mengoprasikan ada beberapa instruksi khusus stack PUSH dan POP.

Contoh :
A, B, C, D, E, Y adalah register

Program: Y = (A – B) / ( C + D × E)

| | |
|---|---|
| PUSH A | S[top] := A |
| PUSH B | S[top] := B |
| SUB | S[top] := A – B |
| PUSH C | S[top] := C |
| PUSH D | S[top] := D |
| PUSH E | S[top] := E |
| MPY | S[top] := D × E |
| ADD | S[top] := C + S[top] |
| DIV | S[top] := (A – B) /S[top] |
| POP Y | Out := S[top] |

Memerlukan 10 operasi

Set instruksi pada CISC:

Berikut ini merupakan karakteristik set instruksi yang digunakan pada beberapa computer yang memiliki arsitektur CISC

**Perbandingan set instruksi**

Beberapa computer CISC (Complex Instruction Set Computer) menggunakan cara implist dalam menentukan mode addressing pada setiap set instruksinya. Penentuan mode addressing dengan cara implicit memiliki arti bahwa pada set instruksi tidak di ada bagian yang menyatakan tipe dari mode addressing yang digunakan, deklarasi dari mode addressing itu berada menyatu dengan opcode. Lain hal nya dengan cara imsplisit, cara eksplisit sengaja menyediakan tempat pada set instruksi untuk mendeklarasikan tipe mode addressing. Pada cara eksplisit deklarasi opcode dan mode addressing berada terpisah.

Data pada tempat deklarasi mode addressing diperoleh dari logaritma basis dua jumlah mode addressing. Jika deklarasi mode addressing dilakukan secara implicit akan menghemat tempat dalam set instruksi paling tidak satu bit untuk IBM 3090 dan 6 bit untuk MC68040. Perubahan satu bit pada set instruksi akan memberikan jangkauan alamat memori lebih luas mengingat range memori dinyatakan oleh bilangan berpangkat dua.

**ELEMEN-ELEMEN DARI INSTRUKSI MESIN (SET INSTRUKSI)**

* Operation Code (opcode) : menentukan operasi yang akan dilaksanakan

* Source Operand Reference : merupakan input bagi operasi yang akan dilaksanakan

* Result Operand Reference : merupakan hasil dari operasi yang dilaksanakan

* Next instruction Reference : memberitahu CPU untuk mengambil (fetch) instruksi berikutnya setelah instruksi yang dijalankan selesai. Source dan result operands dapat berupa salah satu diantara tiga jenis berikut ini:

§ Main or Virtual Memory

- CPU Register
- I/O Device

**DESAIN SET INSTRUKSI**

Desain set instruksi merupakan masalah yang sangat komplek yang melibatkan banyak aspek, diantaranya adalah:

1. Kelengkapan set instruksi
2. Ortogonalitas (sifat independensi instruksi)
3. Kompatibilitas : – Source code compatibility – Object code Compatibility

Selain ketiga aspek tersebut juga melibatkan hal-hal sebagai berikut:

1. Operation Repertoire: Berapa banyak dan operasi apa saja yang disediakan, dan berapa sulit operasinya
2. Data Types: tipe/jenis data yang dapat olah Instruction Format: panjangnya, banyaknya alamat, dsb.
3. Register: Banyaknya register yang dapat digunakan 4.Addressing: Mode pengalamatan untuk operand

**FORMAT INSTRUKSI**

* Suatu instruksi terdiri dari beberapa field yang sesuai dengan elemen dalam instruksi tersebut. Layout dari suatu instruksi sering disebut sebagai Format Instruksi (Instruction Format).