

Nama : Adi Nugroho
Nim : L200150120
Kelas : B

Instuction Set

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 mudah 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. Next 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 alamat 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 penggunaanya, dalam peongoprasianya banyak register sekaligus dan program lebih pendek.

Contoh:

A, B, C, D, E, T, Y adalah register
Program: $Y = (A - B) / (C + D \times E)$
SUB Y, A, B $Y := A - B$
MPY T, D, E $T := D \times E$
ADD T, T, C $T := T + C$
DIV Y, Y, T $Y := Y / T$

- 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 hasilnnya 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: $Y = (A - B) / (C + D \times E)$
MOVE Y, A $Y := A$
SUB Y, B $Y := Y - B$
MOVE T, D $T := D$
MPY T, E $T := T \times E$
ADD T, C $T := T + C$
DIV Y, T $Y := Y / T$

- 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 hasilnnya 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 : $Y = (A - B) / (C + D \times E)$

LOAD D	$AC := D$	LOAD A	$AC := A$
MPY E	$AC := AC \times E$	SUB B	$AC := AC - B$
ADD C	$AC := AC + C$	DIV Y	$AC := AC / Y$
STOR Y	$Y := AC$	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 misal : 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 :

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	$S[top] := D \times E$
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 omputer 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 kompleks 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).

OPCODE OPERAND REFERENCE OPERAND REFERENCE JENIS-JENIS OPERAND

- * Addresses (akan dibahas pada addressing modes)
- * Numbers : – Integer or fixed point – Floating point – Decimal (BCD)
- * Characters : – ASCII – EBCDIC
- * Logical Data : Bila data berbentuk binary: 0 dan 1

JENIS INSTRUKSI

- * Data processing: Arithmetic dan Logic Instructions
- * Data storage: Memory instructions
- * Data Movement: I/O instructions
- * Control: Test and branch instructions

TRANSFER DATA

- * Menetapkan lokasi operand sumber dan operand tujuan.
- * Lokasi-lokasi tersebut dapat berupa memori, register atau bagian paling atas daripada stack.
- * Menetapkan panjang data yang dipindahkan.
- * Menetapkan mode pengalamatan.
- * Tindakan CPU untuk melakukan transfer data adalah :
 - a. Memindahkan data dari satu lokasi ke lokasi lain.
 - b. Apabila memori dilibatkan :
 1. Menetapkan alamat memori.
 2. Menjalankan transformasi alamat memori virtual ke alamat memori aktual.
 3. Mengawali pembacaan / penulisan memori
- * MOVE : memindahkan word atau blok dari sumber ke tujuan
- * STORE : memindahkan word dari prosesor ke memori.
- * LOAD : memindahkan word dari memori ke prosesor.
- * EXCHANGE : menukar isi sumber ke tujuan.
- * CLEAR / RESET : memindahkan word 0 ke tujuan.
- * SET : memindahkan word 1 ke tujuan.

- * PUSH : memindahkan word dari sumber ke bagian paling atas stack.
- * POP : memindahkan word dari bagian paling atas sumber

ARITHMETIC

Tindakan CPU untuk melakukan operasi arithmetic :

1. Transfer data sebelum atau sesudah.
2. Melakukan fungsi dalam ALU.
3. Menetapkan kode-kode kondisi dan flag.

Operasi set instruksi untuk arithmetic :

1. ADD : penjumlahan
2. SUBTRACT : pengurangan
3. MULTIPLY : perkalian
4. DIVIDE : pembagian
5. ABSOLUTE
6. NEGATIVE
7. DECREMENT
8. INCREMENT

Nomor 5 sampai 8 merupakan instruksi operand tunggal. LOGICAL

* Tindakan CPU sama dengan arithmetic

* Operasi set instruksi untuk operasi logical :

1. AND, OR, NOT, EXOR
2. COMPARE : melakukan perbandingan logika.
3. TEST : menguji kondisi tertentu.
4. SHIFT : operand menggeser ke kiri atau kanan menyebabkan konstanta pada ujung bit.
5. ROTATE : operand menggeser ke kiri atau ke kanan dengan ujung yang terjalin.

CONVERSI

Tindakan CPU sama dengan arithmetic dan logical.

* Instruksi yang mengubah format instruksi yang beroperasi terhadap format data.

* Misalnya pengubahan bilangan desimal menjadi bilangan biner.

* Operasi set instruksi untuk konversi :

1. TRANSLATE : menterjemahkan nilai-nilai dalam suatu bagian memori berdasarkan tabel korespondensi.
2. CONVERT : mengkonversi isi suatu word dari suatu bentuk ke bentuk lainnya.

INPUT / OUTPUT

* Tindakan CPU untuk melakukan INPUT / OUTPUT :

1. Apabila memory mapped I/O maka menentukan alamat memory mapped.
2. Mengawasi perintah ke modul I/O

* Operasi set instruksi Input / Output :

1. INPUT : memindahkan data dari perangkat I/O tertentu ke tujuan
2. OUTPUT : memindahkan data dari sumber tertentu ke perangkat I/O
3. START I/O : memindahkan instruksi ke prosesor I/O untuk mengawasi operasi I/O
4. TEST I/O : memindahkan informasi dari sistem I/O ke tujuan TRANSFER CONTROL

* Tindakan CPU untuk transfer control : Mengupdate program counter untuk subrutin, call / return.

* Operasi set instruksi untuk transfer control :

1. JUMP (cabang) : pemindahan tidak bersyarat dan memuat PC dengan alamat tertentu.
2. JUMP BERSYARAT : menguji persyaratan tertentu dan memuat PC dengan alamat tertentu atau tidak melakukan apa tergantung dari persyaratan.
3. JUMP SUBROUTIN : melompat ke alamat tertentu.
4. RETURN : mengganti isi PC dan register lainnya yang berasal dari lokasi tertentu.
5. EXECUTE : mengambil operand dari lokasi tertentu dan mengeksekusi sebagai instruksi
6. SKIP : menambah PC sehingga melompati instruksi berikutnya.

7. SKIP BERSYARAT : melompat atau tidak melakukan apa-apa berdasarkan pada persyaratan
8. HALT : menghentikan eksekusi program.
9. WAIT (HOLD) : melanjutkan eksekusi pada saat persyaratan dipenuhi
10. NO OPERATION : tidak ada operasi yang dilakukan.

* Hanya dapat dieksekusi ketika prosesor berada dalam keadaan khusus tertentu atau sedang mengeksekusi suatu program yang berada dalam area khusus, biasanya digunakan dalam sistem operasi. * Contoh : membaca atau mengubah register kontrol.

* Salah satu cara tradisional untuk menggambarkan arsitektur prosesor adalah dengan melihat jumlah alamat yang terkandung dalam setiap instruksinya.

- * Jumlah alamat maksimum yang mungkin diperlukan dalam sebuah instruksi :
1. Empat Alamat (dua operand, satu hasil, satu untuk alamat instruksi berikutnya)
 2. Tiga Alamat (dua operand, satu hasil)
 3. Dua Alamat (satu operand merangkap hasil, satunya lagi operand)
 4. Satu Alamat (menggunakan accumulator untuk menyimpan operand dan hasilnya)

Macam-macam instruksi menurut jumlah operasi yang dispesifikasikan

1. O – Address Instruction
2. 1 – Address Instruction.
3. N – Address Instruction
4. M + N – Address Instruction

Macam-macam instruksi menurut sifat akses terhadap memori atau register

1. Memori To Register Instruction
2. Memori To Memori Instruction
3. Register To Register Instruction

ADDRESSING MODES

Jenis-jenis addressing modes (Teknik Pengalamatan) yang paling umum:

- * Immediate
- * Direct
- * Indirect
- * Register
- * Register Indirect
- * Displacement
- * Stack

Instruction Set Processor INTEL

- 4004
- 8008 / Datapoint 2200
- 8080 (111 Instructions), 8085 (113 Instructions)

1. DATA TRANSFER INSTRUKSI

MOV Rd, Rs ; berfungsi mengcopy nilai dari Rs ke Rd

MOV Rd, M ; berfungsi mengcopy nilai dari M ke Rd

MOV M, Rs ; berfungsi mengcopy nilai dari M ke Rs

MVI Rd, d8 ; berfungsi memindahkan nilai register d8 ke register d8 MVI

M, d8 ; berfungsi memindahkan nilai register d8 ke register M

LDA addr16 ; berfungsi menyalin data memori pada alamat yang spesifik addr16 LDAX

rp ; berfungsi mengcopy data pada register pair (rp)

LXI rp, d16 ; berfungsi mengisi register pair (rp) dari nilai data d16 (alamat 16 bit) LHLD

addr16 ; berfungsi menyalin data memori pada alamat yang spesifik addr16 STA addr16

; berfungsi menyimpan nilai data langsung dalam memori addr16 STAX rp ; berfungsi menyimpan nilai data pada alamat register pair (rp)

SHLD addr16 ; berfungsi menyimpan data register H & L langsung dalam memori alamat addr16

SPHL ; berfungsi memindahkan isi dari H & L ke pointer stack XCHG ;

berfungsi menukar register H & L dengan register D & E XHTL ;

berfungsi menukar stack tertinggi dengan register H & L PUSH rp ;

push 2 byte data ke stack pada register pair (rp)

PUSH PSW ; push 2 byte data ke stack pada processor status word (8-bit) POP

rp ; Pop Two Bytes of Data off the Stack

2. ARITHMETIC INSTRUKSI

ADD reg ; instruksi penambahan pada register reg

ADD M ; instruksi penambahan pada register M

ADI d8 ; instruksi penambahan data secara immediate pada register d8 ADC reg ;

instruksi penambahan menggunakan carry flag pada register reg ADC M ; instruksi

penambahan menggunakan carry flag pada register M ACI d8 ; instruksi

penambahan data d8 secara immediate menggunakan carry DAA ; instruksi untuk mengatur bentuk desimal

DAD rp ; penambahan register pair ganda ke H & L register pair (rp) SUB

reg ; instruksi pengurangan pada register reg

SUB M ; instruksi pengurangan pada register M

SUI d8 ; instruksi pengurangan data pada d8 secara immediate

SBB reg ; instruksi pengurangan menggunakan carry flag pada register reg SBB

M ; instruksi pengurangan menggunakan carry flag pada register M

SBI d8 ; instruksi pengurangan secara immediate menggunakan carry flag pada register d8

INR reg ; instruksi kenaikan data reg setiap 1 byte INR M

; instruksi kenaikan data M setiap 1 byte INX rp ;

instruksi kenaikan 1 data register pair (rp) DCR reg ;

instruksi penurunan data reg setiap 1 byte

3. LOGIKA INSTRUKSI

ANA reg ; menggunakan logika AND dengan logika accumulator pada data reg ANA M ; menggunakan logika AND dengan logika accumulator pada data M ANI d8 ; menggunakan logika AND dengan logika accumulator immediate d8 ORA reg ; menggunakan logika OR dengan logika accumulator OR pada reg ORA M ; menggunakan logika OR dengan logika accumulator OR pada M ORI d8 ; menggunakan logika OR dengan logika accumulator OR immediate register d8 XRA reg ; menggunakan logika eksklusif OR dengan logika accumulator eksklusif OR reg XRA M ; menggunakan logika eksklusif OR dengan logika accumulator eksklusif OR reg M XRI d8 ; menggunakan logika eksklusif OR dengan data immediate pada register d8 CMP reg ; membandingkan data pada reg CMP M ; membandingkan data pada register M CPI d8 ; membandingkan data secara immediate pada d8 CMA ; pelengkap accumulator data pada prosesor 8085 CMC ; pelengkap carry flag pada prosesor 8085 STC ; pengatur/set/setting carry flag RLC ; pengatur rotasi/putaran accumulator pada bagian kiri RAL ; Rotate Left Through Carry RRC ; pengatur rotasi/putaran accumulator pada bagian kanan RAR ; Rotate Right Through Carry

4. BRANCHING INSTRUKSI

JMP addr16 ; berfungsi untuk membuat program beralih/loncat ke addr16 J addr16 ; CALL addr16 ; berfungsi untuk memanggil data pada addr16 C addr16 ; RET ; berfungsi untuk kembali pada instruksi awal R ; RST n ; berfungsi sebagai instruksi restart secara khusus PCHL ; berfungsi untuk memindahkan H & L pada program counter

5. MACHINE CONTROL INSTRUKSI

SIM ; membuat settingan mask interrupt pada mesin prosesor 8085 RIM ; membaca mask interrupt pada mesin prosesor 8085 DI ; menonaktifkan system interrupt pada mesin prosesor 8085 EI ; mengaktifkan system interrupt pada mesin prosesor 8085 HLT ; memberhentikan mesin NOP ; tidak ada operasi apapun pada kontrol mesin