Pertemuan ke 6

Set Instruksi



Computer Organization

Dosen : Eko Budi Setiawan

Tujuan

- Memahami representasi set instruksi, dan jenis-jenis format instruksi
- Mengetahui jenis-jenis type operand yang digunakan
- Macam-macam mode pengalamatan
- Format Instruksi
- Format Instruksi pada Pentium
- Memahami Implementasi Set Instruksi pada Pentium II

Karakteristik Mesin Instruksi

- Instruksi mesin (machine instruction) yang dieksekusi membentuk suatu operasi dan berbagai macam fungsi CPU
- Kumpulan fungsi yang dapat dieksekusi CPU yang disebut set instruksi (instruction set) CPU
- Mempelajari karakteristik instruksi mesin, yang meliputi
  - Elemen-elemen instruksi mesin
  - Representasi instruksinya
  - Jenis – jenis instruksi
  - Penggunaan alamat
  - Rancangan set instruksi

Elemen Instruksi Mesin

- Untuk dapat dieksekusi CPU, suatu instruksi harus berisi elemen informasi yang diperlukan CPU secara lengkap dan jelas, apa saja elemenya ?

1. Operation Code (Op Code)

   Menspesifikasi operasi yang akan dilakukan. Kode operasi berbentuk kode binder

2. Source Operand Reference

   Operasi dapat berasal dari lebih dari satu sumber. Operand adalah input operasi

3. Result Operand Reference

   Merupakan hasil atau keluaran operasi

4. Next Instruction Reference

   Elemen ini menginformasikan CPU, posisi instruksi berikutnya yang harus diambil dan dieksekusi

Operand dari operasi

- Melihat dari sumbernya, operand suatu operasi dapat berada disalah satu dari daerah berikut ini :
  - Memori utama atau memori virtual
  - Register CPU
  - Perangkat I/O

Representasi Instruksi

- Instruksi Komputer direpresentasikan oleh sekumpulan bit. Instruksi dibagi menjadi beberapa field
- Field – field ini diisi oleh elemen-elemen instruksi yang membawa informasi bagi operasi CPU
- Layout instruksi dikenal dengan format instruksi
- Format Instruksi

Op Code	Alamat

  - Kode Operasi (Op Code) direpresentasikan dengan singkatan-singkatan yang disebut dengan mnemonic
  - Mnemonic mengindikasikan suatu operasi bagi CPU
  - Contoh mnemonic
    - ADD = penambahan
    - SUB = substract (pengurangan)
    - LOAD = muatkan data ke memori

Representasi Instruksi

- Contoh representasi operand secara simbolik
  - ADD X, Y artinya : tambahkan nilai yang berada pada lokasi Y ke isi register X, dan simpan hasilnya diregister X
- Programmer dapat menuliskan program bahasa mesin dalam bentuk simbolik
- Setiap Op Code simbolik memiliki representasi biner yang tetap dan programmer dapat menetapkan lokasi masing-masing operand

Jenis – jenis Instruksi

- Dapat ditarik kesimpulan bahwa instruksi
  X = X + Y
  X dan Y berkorespondensi dengan lokasi 210 dan 211
  Pernyataan dalam bahasa tinkat tinggi tersebut menginstruksikan komputer untuk melakukan langkah berikut ini :
  - Muatkan sebuah register dengan isi lokasi memori 210
  - Tambahkan isi lokasi memori 211 ke register
  - Simpan isi register ke lokasi memori 210
- Terlihat hubungan antara ekspresi bahasa tingkat tinggi dengan bahasa mesin
- Dalam bahasa tingkat tinggi, operasi dinyatakan dalam bentuk aljabar singkat menggunakan variabel
- Dalam bahasa mesin hal tersebut diekpresikan dalam operasi perpindahan antar register

Jenis – jenis Instruksi

- Dapat ditarik kesimpulan bahwa instruksi – instruksi mesin harus mampu mengolah data sebagai implementasi keinginan-keinginan kita
- Terdapat kumpulan unik set instruksi, yang dapat digolongkan dalam jenis – jenisnya, yaitu
  - Pengolahan data (data processing)

    Meliputi operasi – operasi aritmatika dan logika
  - Perpindahan data (data movement)

    Berisi instruksi perpindahan data antar register maupun modul I/O
  - Penyimpanan data (data storage)

    Instruksi penyimpanan sangat penting dalam operasi komputasi, karena data tersebut akan digunakan untuk operasi berikutnya
  - Kontrol aliran program (program flow control)

    Berisi instruksi pengontrolan operasi dari percabangan, instruksi ini berguna untuk pengontrolan status dan mengoperasikan percabangan ke set instruksi lain

Jumlah Alamat

- Jumlah register atau alamat yang digunakan dalam operasi CPU tergantung format operasi masing-masing CPU
- Ada format operasi yang menggunakan 3, 2, 1 dan 0 register
- Umumnya yang digunakan adalah 2 register dalam suatu operasi. Desain CPU saat ini telah menggunakan 3 alamat dalam suatu operasi terutama dalam suatu operasi, terutama dalam MIPS (million instruction per second)
- Alamat per instruksi yang lebih sedikit akan membuat instruksi lebih sederhana dan pendek, tetapi lebih sulit mengimplementasikan fungsi-fungsi yang kita inginkan
- Karena instruksi CPU sederhana maka rancangan CPU juga lebih sederhana

Jumlah Alamat

- Jumlah bit dan referensi per instruksi lebih sedikit sehingga fetch dan eksekusi lebih cepat
- Jumlah instruksi per program biasanya jauh lebih banyak
- Pada jumlah alamat per instruksi banyak, jumlah bit dan referensi instruksi lebih banyak sehingga waktu eksekusi lebih lama
- Diperlukan register CPU yang banyak, namun operasi antar register lebih cepat
- Lebih mudah mengimplementasikan fungsi – fungsi yang kita inginkan
- Jumlah instruksi per program jauh lebih sedikit
- Untuk lebih jelas perhatikan contoh instruksi – instruksi dengan jumlah register berbeda untuk menyelesaikan persoalan yang sama

e.g Set Instruksi

- Contoh penggunaan set instruksi dengan alamat 1, 2 da 3 untuk menyelesaikan operasi hitungan

$$Y = (A-B) / (C+D*E)$$

Instruksi 3 alamat :

Instruksi		Komentar
SUB	Y, A, B	Y =A – B
MPY	T, D, E	T= D x E
ADD	T, T, C	T=T + C
DIV	Y, Y, T	Y= Y ÷ T

Instruksi 2 alamat :

Instruksi		Komentar
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

e.g Set Instruksi

Instruksi 1 Alamat

Instruksi		Komentar
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

Instruksi	Keterangan	isi stack
PUSH	B	B
PUSH	A	B,A
SUB	A-B	(A-B)
PUSH	E	(A-B),E
PUSH	D	(A-B),E,D
MUL	D*E	(A-B),(D*E)
PUSH	C	(A-B),(D*E),C
ADD	C+(D*E)	(A-B),(C+D*E)
DIV	(A-B)/(C+(D*E))	(A-B)/(C+(D*E))

e.g Set Instruksi

- Spesifikasi instruksi 3 alamat
  - Simbolik a = b + c
  - Format alamat : hasil , operand 1, operand 2
  - Digunakan dalam arsitektur MIPS
  - Memerlukan word panjang dalam suatu instruksi

- Spesifikasi instruksi 2 alamat :
  - Simbolik a = a + b
  - Satu alamat diisi operand terlebih dahulu kemudian digunakan untuk menyimpan haslnya
  - Tidak memerlukan instruksi yang panjang
  - Jumlah instruksi perprogram akan lebih banyak daripada 3 alamat
  - Diperlukan penyimpanan sementara untuk menyimpan hasil

e.g Set Instruksi

- Spesifikasi instruksi 1 alamat
  - Memerlukan alamat implisit untuk operasi
  - Menggunakan register akumulator (AC) dan digunakan pada mesin lama
- Spesifikasi instruksi 0 alamat :
  - Seluruh alamat yg digunakan implisit
  - Digunakan pada organisasi memori, terutama operasi stack

Rancangan Set Instruksi

- Aspek paling menarik dalam arsitektur komputer adalah perancangan set instruksi, karena rancangan ini berpengaruh banyak pada aspek lainnya
- Set instruksi menentukan banyak fungsi yang harus dilakukan CPU
- Set instruksi merupakan alat bagi para pemrogram untuk mengontrol kerja CPU
- Pertimbangan : kebutuhan pemrogram menjadi bahan pertimbangan dalam merancang set instruksi

Tipe Operasi

- Dalam perancangan arsitektur komputer, jumlah kode operasi akan sangat berbeda untuk masing-masing komputer, tetapi terdapat kemiripan dalam jenis operasinya
- Jenis Operasi Komputer
  - Transfer data – Konversi
  - Aritmatika – Input / output
  - Logika – Kontrol sistem dan transfer kontrol

Operasi set instruksi

Jenis	Nama Operasi	Keterangan
Pemindahan Data	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 dengan 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 stack teratas ke tujuan.

Operasi set instruksi

Logika	AND	Melakukan operasi logika tertentu terhadap bit.
	OR	
	NOT	
	Exclusive-OR	
	Test	Menguji kondisi tertentu.
	Compare	Membandingkan dua operand (secara logika maupun aritmetika).
	Set Variabel Kontrol	Instruksi untuk menyetel kontrol bagi keperluan proteksi, interrupt, kontrol timer.
	Shift	Melakukan penggeseran bit – bit operand.
	Rotate	Melakukan pemutaran bit – bit operand.

Operasi set instruksi

Pemindahan Kontrol	Jump (cabang)	Perpindahan tidak bersyarat; memuatkan PC dengan alamat tertentu.
	Jump bersyarat	Menguji persyaratan tertentu; melakukan aktivitas tergantung persyaratannya.
	Jump ke subrutin	Menempatkan informasi data kontrol program saat itu di lokasi yang ditentukan; melompat ke alamat tertentu.
	Return	Mengganti ini PC dan register lainnya yang berasal dari lokasi tertentu.

Operasi set instruksi

Pemindahan Kontrol	Execute	Mengambil operand dari lokasi tertentu dan mengeksekusinya sebagai instruksi.
	Skip	Menambah PC sehingga melompati instruksi berikutnya.
	Skip bersyarat	Melompat berdasarkan syarat tertentu.
	Halt	Menghentikan eksekusi program.
	Wait (hold)	Menghentikan eksekusi program dan menguji persyaratan.
	No operation	Tidak ada operasi yang dilakukan, namun eksekusi program dilanjutkan.

Operasi set instruksi

Input/Output	Input (read)	Memindahkan data dari port I/O ke tujuan.
	Output (write)	Memindahkan data dari prosesor ke port atau modul I/O.
	Start I/O	Memindahkan instruksi ke prosesor I/O untuk memulai proses I/O.
	Test I/O	Memindahkan informasi status dari sistem I/O ke tujuan.

Format & Panjang Instruksi

- Format instruksi menentukan susunan dan tata letak bit suatu instruksi
- Format instruksi harus mencakup opcode serta implisit dan eksplisit operand
- Biasanya set instruksi memiliki lebih dari satu format instruksi
- Inti dari format instruksi adalah menentukan panjang instruksi dan alokasi bit dalam instruksi tersebut
- Penentuan panjang instruksi mempengaruhi dan dipengaruhi oleh
  - Ukuran memori
  - Organisasi memori
  - Struktur Bus
  - Kompleksitas CPU
  - Kecepatan CPU

Penentuan Panjang Instruksi

- Pertimbangan (INSTRUKSI)
  - Instruksi yang kompleks mempengaruhi perancangan perangkat keras prosesorm karena fungsi-fungsi yang disajikan CPU harus diimplementasikan dalam perangkat keras
  - Semakin kompleks perangkat keras, tentunya akan meningkatkan faktor biaya walau belum tentu meningkatkan kinerja komputer secara keseluruhan
  - Penentuan panjang instruksi menjadi sangat essensi untuk mencapai kinerja komputer yang maksimal
- Pertimbangan (PROGRAMMER)
  - Menginginkan opcode, operand, dan mode pengalamatan yang lebih banyak serta range alamat yang lebih besar karena semua itu akan mempermudah pemrogram
  - Opcode, operand dan mode pengalamatan yang lebih banyak akan membutuhkan ruang yang lebih besar
  - Instruksi 32 bit akan menempati ruang 2x lebih banyak daripada 16 bit, namun kegunaannya belum tentu 2x lebih banyak

Penentuan Panjang Instruksi

- Pertimbangan Lain
  - Panjang instruksi harus sama dengan panjang perpindahan memori (pada sistem bus, panjang bus data) dan panjang instruksi seharunya merupakan kelipatan panjang instruksi lainnya
  - Hal ini harus dipertimbangkan untuk mendapatkan optimalisasi proses eksekusi instruksi nantinyam baik kecepatan perpindahan maupun alikasi memorinya
  - Kecepatan perpindahan data tidak dapat diatasi dengan menambah kecepatan prosesor
  - Kecepatan prosesor hanya berhubungan dengan eksekusi internalnya, sedangkan kecepatan perpindahan tergantung Bus, memori dan data itu sendiri
  - Cara meningkatkan kecepatan perpindahan data adalah dengan menggunakan cache memori dan menggunakan instruksi – instruksi yang lebih pendek

Penentuan Panjang Instruksi

- Panjang instruksi harus merupakan kelipatan panjang karakter, yang umumnya 8 bit, dan kelipatan panjang bilangan fixed point

- Diabaikan ?
  - Terjadi pemborosan bit pada setiap word ketika sejumlah karakter disimpan didalamnya

- Keputusan salah yang pernah diambil IBM
  - Mengeluarkan arsitektur prosesor 36 bit, terjadi banyak peborosan karena ukuran karakter 8 bit
  - Arsitektur tersebut diganti dengan arsitektur 32 bit

Alokasi Bit

- Inti dalam alokasi bit adalah berada pada untung – rugi antara jumlah opcode dengan kemampuan pengalamatannya
- Opcode yang banyak akan menyebabkan bit yang lebih banyak pada field opcode, yang secara otomatis akan mengurangi jumlah bit untuk pengalamatan
- Faktor yang merupakan hal-hal yang penting dalam menentukan penggunaan bit-bit pengalamatan :
  - Jumlah mode pengalamatan
  - Jumlah Operand
  - Register vs Memori
  - Jumlah set register
  - Granularitas Alamat

Alokasi Bit

- Jumlah mode pengalamatan
  - Mode pengalamatan dapat dilakukan secara implisit atau eksplisit, yang kesemuanya memerlukan jumlah bit yang berbeda
- Jumlah Operand
  - Jumlah operand sangat mempengaruhi kemampuan instruksi
  - Jumlah operand yang sedikit biasanya akan menjadikan instruksi yang panjang dalam suatu fungsi
- Register vs memori
  - Penggunaan register maupun memori membutuhkan jumlah bit yang berbeda
  - Pada pengalamatan implisit dengan register akan dibutuhkan bit lebih kecil daripada mode pengalamatan langsung ke memori

Alokasi Bit

- Jumlah set register
  - Jumlah set register juga mempengaruhi penggunaan bit-bit instruksi
  - General Purpose Register yang umumnya dimiliki hampir seluruh komputer dapat digunakan untuk register alamat maupun register instruksi
- Jangkauan alamat
  - Untuk alamat – alamat yang mereferensi memori secara eksplisit jangkauan ditentukan oleh jumlah bit yang digunakan untuk pengalamatan
  - Pertimbangan menggunakan mode displacement patut dipertimbangan untuk memiliki jangkauan pengalamatan yang besar
- Granularitas alamat
  - Pengalamatan yang mereferensi memori dapat digunakan pengalamatan yang mereferensi word atau byte