TIPE DAN FORMAT INSTRUKSI (Bagian 2)

SRI SUPATMI,S.KOM

6. Lokasi Operand

- Ada beberapa pilihan dalam menempatkan operand (lokasi operand) yaitu pada: memori utama, register CPU, I/O port dan pada instruksi itu sendiri.
- Membiarkan operand dalam register CPU lebih efektif daripada mengambilnya dari memori utama karena waktu akses register CPU lebih singkat. Hasil ini mengurangi waktu siklus instruksi.
- Penempatan operand dalam instruksi digunakan untuk instruksi-instruksi khusus saja.
- Isi suatu port dapat digunakan sebagai operand seperti halnya isi suatu lokasi memori.
- Ada beberapa instruksi yang tidak mempunyai operand.
- Contoh yang khas adalan instruksi HALT dan NOOP.
- Ada beberapa instruksi yang berfungsi untuk menguji status komponen-komponen hardware seperti register, flip-flop, lokasi memori dsb.
- Pada kasus ini, tidak terdapat operand,demikian halnya ada beberapa instruksi yang hanya mencari sejumlah sinyal eksternal.

7.Lokasi hasil pemrosesan

- Ada beberapa pilihan dalam menyimpan hasil pemrosesan instruksi yaitu: memori utama, register CPU, poer keluaran dsb.
- Beberapa instruksi seperti HALT dan NOOP secara eksplisit tidak mempunyai hasil untuk disimpan sedangkan beberapa instruksi hanya men-set atau me-reset flip-flop atau register.

8. Urutan data Little-endian dan Big-endian

- Ada dua metode konvensi yang diikuti untuk penempatan informasi dalam memori dan pengalamatan yaitu:
  - Penempatan dengan Big-endian
  - Penempatan dengan Little-endian
- Pada metode *big-endian, MSB (most significant byte)* disimpan pada alamat bawah dan *LSB (least significant byte)* disimpan pada alamat atas.
- Pada metode *litlle-endian,LSB(least significant byte)* disimpan pada alamat bawah dan *MSB (Most significant byte)* disimpan pada alamat atas.

- Anggap kita mempunyai sebuah informasi 32-bit 12345678(hexa) yang akan disimpan didalam lokasi memori 1000 ke atas.
- Karena terdapat 4-byte, maka informasi menempati alamat 1000 sampai 1003.
- Solusi:
  - Alamat → 1000 1001 1002 1003

Big-endian	**12**	**34**	**56**	**78**
Little-endian	**78**	**56**	**34**	**12**

- Jika kita mempunyai data 64-bit (8-byte), maka 4-byte sisanya dilanjutkan lagi dari alamat 1004 ke atas dengan menggunakan cara yang sama.
- Metode *big-endian* lebih natural pada sebagian besar orang karena lebih mudah untuk membaca posisi hex.

9. Tipe Instruksi

- Instruksi-instruksi diklasifikasikan dalam tipe yang berbeda berdasarkan faktor-faktor berikut:

1. **opcode:** kode operasi yang harus dikerjakan oleh instruksi.
2. **Data:** tipe data:biner,desimal dan sebagainya
3. **Lokasi Operand:** memori, register dan sebagianya
4. **Pengalamatan Operand:** metode penentuan lokasi operand (alamat)
5. **Panjang Instruksi:** satu byte, dua byte dan sebagainya
6. **Jumlah Medan alamat:** nol alamat, satu alamat, dua alamat dan sebagainya.

- Instruksi-instruksi dapat diklasifikasikan ke dalam delapan jenis:

1. Instruksi transfer data:instruksi ini menyalin data dari satu register/lokasi memori ke yang lainnya.
2. Instruksi aritmetika: instruksi ini melakukan operasi-operasi aritmetika
3. Instruksi Logika: instruksi ini melakukan operasi-operasi logika boolean
4. Instruksi transfer kontrol: instruksi ini melakukan modifikasi/mengubah urutan eksekusi program.
5. Instruksi I/O: instruksi ini melakukan transfer antara peripheral eksternal dan inti sistem (CPU/memori)
6. Instruksi Manipulasi String: Instruksi ini melakukan manipulasi string byte, word, double word dan sebagainya.
7. Instruksi translate: instruksi ini melakukan konversi data dari satu format ke format lain.
8. Instruksi kontrol prosesor: instruksi ini melakukan kontrol operasi prosesor.

>> Tabel beberapa contoh instruksi untuk setiap jenis instruksi

no	tipe	instruksi	
		Nama	aksi
1	Transfer Data	MOVE	Mentransfer data dari lokasi sumber ke lokasi tujuan
		LOAD	Mentransfer data dari lokasi memori ke register CPU
		STORE	Mentransfer data dari register CPU ke lokasi memori
		PUSH	Mentransfer data dari sumber ke stack (puncak)
		POP	Mentransfer data dari stack (puncak) ke tujuan
		XCHG	Tukar;menukar isi sumber dan tujuan
		CLEAR	Reset tujuan dengan semua bit “0”
		SET	Set tujuan dengan semua bit “1”

>> Tabel beberapa contoh instruksi untuk setiap jenis instruksi lanjutan (1)

no	tipe	instruksi	
		Nama	aksi
2	Aritmetika	ADD	Jumlah;hitung jumlah dari dua-operand
		ADC	Jumlah dengan carry; hitung jumlah dari dua operand dan bit 'carry'
		SUB	Kurang; hitung selisih dua operand
		SUBB	Kurang dengan borrow; hitung selisih dengan 'borrow'
		MUL	Perkalian;hitung hasil kali dari dua operand
		DIV	Pembagian; hitung hasil bagi dan sisa pembagian dari dua bilangan
		NEG	Negate; ganti tanda operand
		INC	Increment; tambahkan 1 pada operand
		DEC	Decrement; kurangkan 1 pada operand
		SHIFT A	Shift arithmatic; geser operand (ke kiri atau kanan) dengan tanda

>> Tabel beberapa contoh instruksi untuk setiap jenis instruksi lanjutan (2)

no	tipe	instruksi	
		Nama	aksi
3	Logika	NOT	komplemenkan (komplemen 1) operand
		OR	Lakukan operasi logika OR pada operand
		AND	Lakukan operasi logika AND pada operand
		XOR	Lakukan operasi logika 'exclusive-OR' pada operand
		SHIFT	Geser operand (ke kiri atau kanan) sis bit kosong dengan 'o'
		ROT	Rotasi;geser operand (ke kiri atau kanan) dengan berputar
		TEST	Uji kondisi yang ditetapkan dan pengaruhi flag yang relevan

>> Tabel beberapa contoh instruksi untuk setiap jenis instruksi lanjutan (3)

no	tipe	instruksi	
		Nama	aksi
4	Transfer Kontrol	JUMP	branch;:masukkan alamat yang ditetapkan ke PC; cabang tak bersyarat (unconditional transfer)
		JUMPIF	Bercabang dengan kondisi;masukkan alamat yang ditetapkan ke PC hanya jika kondisi yang ditetapkan terpenuhi;conditional transfer
		JUMPSUB	CALL;simpan 'program control status' yang sekarang dan masukkan alamat yang ditetapkan ke PC
		RET	*RETURN;unsave(restore)*'program control status' (dari stack) ke PC dan register/flag yang relevan.
		INT	Interupsi;melakukan interupsi software;simpan 'status kontrol program'(ke stack) dan masukkan alamat sesuai dengan kode yang ditetapkan *(vector)* ke PC
		IRET	Interrupt return;ambil kembali 'status program kontrol' dari stack ke PC serta register-register dan flag yang relevan lainnya.
		LOOP	Iterasi;turunkan(decrement) isi register dengan 1 dan uji non-zero;jika tercapai, masukkan alamat yang ditetapkan ke PC.

>> Tabel beberapa contoh instruksi untuk setiap jenis instruksi lanjutan (4)

no	tipe	instruksi	
		Nama	aksi
5	Instruksi input/output	IN	Input;baca data dari port/divais yang ditetapkan ke register yang ditetapkan atau yang terlibat
		OUT	Output; tulis data dari register yang ditetapkan atau yang telibat ke suatu port/divais
		TEST I/O	Baca status dari subsistem I/O dan set kondisi flag
		START I/O	Sinyal prosesor I/O (atau *data channel)* untuk memulai program I/O (perintah untuk program I/O)
		HALT I/O	Sinyal prosesor I/O (atau *data channel)* untuk membatalkan program I/O (perintah untuk program I/O)dalam *progress*

>> Tabel beberapa contoh instruksi untuk setiap jenis instruksi lanjutan (5)

no	tipe	instruksi	
		Nama	aksi
6	Manipulasi String	MOVS	Salin (move) byte atau wors string
		LODS	Salin (load) byte atau word string
		CMPS	Bandingkan byte atau word string
		STOS	Simpan (store) byte atau word string
		SCAS	Scan byte atau word string

>> Tabel beberapa contoh instruksi untuk setiap jenis instruksi lanjutan (6)

no	tipe	instruksi	
		Nama	aksi
7	Translate	XLAT	Translate; ubah kode yang diberikan ke bentuk yang lain dengan table *lookup*
		HLT	*Halt*;hentikan siklus instruksi instruksi (pemrosesan)
		STI(EI)	*Set interrupt (enable interrupt);* men-*se interrupt enable flag* ke '1'
		CLI (DI)	*Clear interrupt (enable interrupt);* me-*reset interrupt enable flag ke* '0'
		WAIT	Penghentian siklus instruksi hingga suatu kondisi terpenuhi (seperti sinyal input menjadi aktif)
		NOOP	No Operation; nothing
		CMC	Komplemenkan *carry flag*
		CLC	Jadikan '0' *carry flag*
		STC	Jadikan '1' *carry flag*