TIPE DAN FORMAT INSTRUKSI (Bagian 1)

SRI SUPATMI,S.KOM

1. Desain Set Instruksi

- Pekerjaan yang paling signifikan/penting dan kompleks dalam mendesain komputer adalah membuat set instruksi.
- Komputer era sebelumnya tidak melakukan perencanaan set instruksi.
- Kelemahan desain set instruksi mereka secara drastis mempengaruhi ruang memori utama oleh panjang program (bahasa mesin). Karena itu, desain yang baik, perencanaan awal set instruksi memungkinkan compiler membuat kode objek yang kompak (tersusun baik dan padat) tersimpan pada ruang memori.

- Seorang arsitektur komputer harus mempertimbangkan aspek-aspek berikut sebelum menyelesaikan set instruksi:

1. **Kenyamanan Pemrograman:** Jumlah instruksi; pemrogram lebih suka mempunyai sebanyak mungkin instruksi supaya operasi yang tepat dapat dikerjakan oleh rangkaian instruksi. Tetapi mempunyai terlalu banyak instruksi dalam set instruksi menghasilkan desain unit kontrol yang kompleks. Pendekokean instruksi memerlukan sirkuit dan waktu yang besar.
2. **Pengalamatan yang fleksibel:** pemrogram senang jika memungkinkan semua mode pengalamatan operand ada di dalam arsitektur. Hal ini memberikan fleksibilas yang banyak kepada pemrogram. Walaupun desain unit kontrol menjadi kompleks.
3. **Jumlah General Purpose Register (GPR):** Jika CPU mempunyai register yang banyak, pemrogram memperoleh pemrosesan dan transfer data yang cepat. Tetapi biaya perangkat keras CPU meningkat dengan banyaknya GPR.

>> Lanjutan

4. **Target Segmen Pasar:** sasaran bidang aplikasi untuk komputer memerlukan operasi-operasi khusus untuk pemrosesan data yang efisien. Komputer saintifk harus mempunyai aritmetika *floating-point* yang tingkat presisinya baik/tidak terlalu jelek. Sedangkan komputer bisnis harus mendukung aritmetika desimal, komputer iburan harus mempunyai operasi-operasi multimedia.
5. **Kinerja Sistem:** Jika sebuah program mempunyai instruksi sedikit, kinerja sistem meningkat karena waktu yang digunakan oleh CPU dalam pengambilan instruksi berkurang. Untuk program yang pendek, instruksi yang digunakan harus instruksi kompleks. Jadi instruksi tunggal harus dapat melakukan beberapa mikrooperasi. Pemrogram menyadari hal ini mengurangi ukuran program. Tapi di sisi lain menambah kompleksitas unit kontrol dan waktu eksekusi instruksi. Konsep modern arsitektur RISC tidak mendukung instruksi-instruksi kompleks, walaupun semua komputer lama yang berbasis CISC menggunakan instruksi-instruksi kompleks.

2. CPU berbasis Akumulator

- Pada mulanya komputer adalah berbasis akumulator. Hal ini merupakan CPU yang sederhana, dimana akumulator berisi satu operand pada instruksi, demikian juga hasilnya disimpan pada akumulator. Isi akumulator disertakan di dalam operasi-operasi aritmetika seperti penjumlahan, pengurangan dan sebagainya. Hal ini dikenal sebagai **mesin satu-alamat.**
- PDP-8 merupakan minicomputer pertama yang mempunyai jenis CPU seperti ini dan digunakan untuk kendali proses dan aplikasi-aplikasi laboratorium.
- Komputer Mark I juga merupakan komputer khas yang berbasis akumulator.
- Organisasi CPU secara total telh digantikan dengan diperkenalkan CPU berbasis register yang baru.

- **Keuntungan CPU berbasis akumulator adalah:**

1. Isi akumulator diperuntukkan bagi satu operand. Karena itu tidak memerlukan medan alamat operand (untuk satu operand) dalam instruksi. Hal ini menghasilakn instruksi yang pendek dan ruang memori yang sedikit. Karena tidak adanya medan alamat, maka jenis CPU ini mendukung instruksi-instruksi non-alamat dan satu-alamat. Instruksi satu-alamat mempunyai satu operand di dalam memori utama dan yang lainnya di dalam akumulator.
2. Siklus instruksi menggunakan waktu yang singkat sebab menghemat waktu dalam pengambilan instruksi karena tidak ada siklus pengambilan operand.

- **Kekurangan CPU berbasis akumulator adalah:**

1. Ukuran programnya menjadi panjang karena banyak menggunakan instruksi dalam ekspresi-ekspresi kompleks. Karena itu ukuran memori bertambah.
2. Waktu eksekusi program bertambah karena bertambahnya jumlah instruksi dalam program.

>> Contoh1:

- Tuliskan sebuah program bahasa rakitan dalam arsitektur CPU berbasis akumulator untuk menyelesaikan statement X=(A+B)-(C+D)
- Solusi:

LOAD A : Salin A ke dalam akumulator

ADD B : Jumlahkan B dengan isi akumulator dan hasilnya disimpan di akumulator (akumulator berisi A+B)

STORE T : simpan hasil AB dalam T, suatu lokasi memori sementara

LOAD C : salin C ke dalam akumulator

ADD D : Jumlahkan D ke dalam akumulator dan simpan hasilnya dalam akumulator (akumulator berisi C+D)

SUB T : Perkurangkan isi akumulator dari T dan simpan hasilnya dalam akumulator

STORE X : Simpan isi akumulator di dalam lokasi memori X.

3. CPU Berbasis Register

- Pada CPU jenis ini, banyak register yang digunakan sebagai akumulator. Dengan kata lain, ada lebih dari satu akumulator.
- CPU seperti ini mempunyai organisasi register umum GPR (*general purpose register*). Penggunaan register-register tersebut menghasilkan program yang pendek dengan instruksi yang sedikit. IBM System/360 dan PDP-11 merupakan contoh khas.

>> Contoh2:

- Tuliskan sebuah program bahasa rakitan dalam arsitektur CPU berbasis register untuk menyelesaikan statement

  X=(A+B)-(C+D)

- Solusi:

  LOAD R1,A : salin A ke dalam register

  ADD R1,B : Jumlahkan B dengan isi R1 dan hasilnya disimpan di R1

  LOAD R2,C : salin C ke dalam R2

  ADD R2,D : Jumlahkan D ke dalam R2 dan simpan hasilnya di R2

  SUB R1,R2 : perkurangkan isi R2 dan R1 dan simpan hasilnya dalam R1

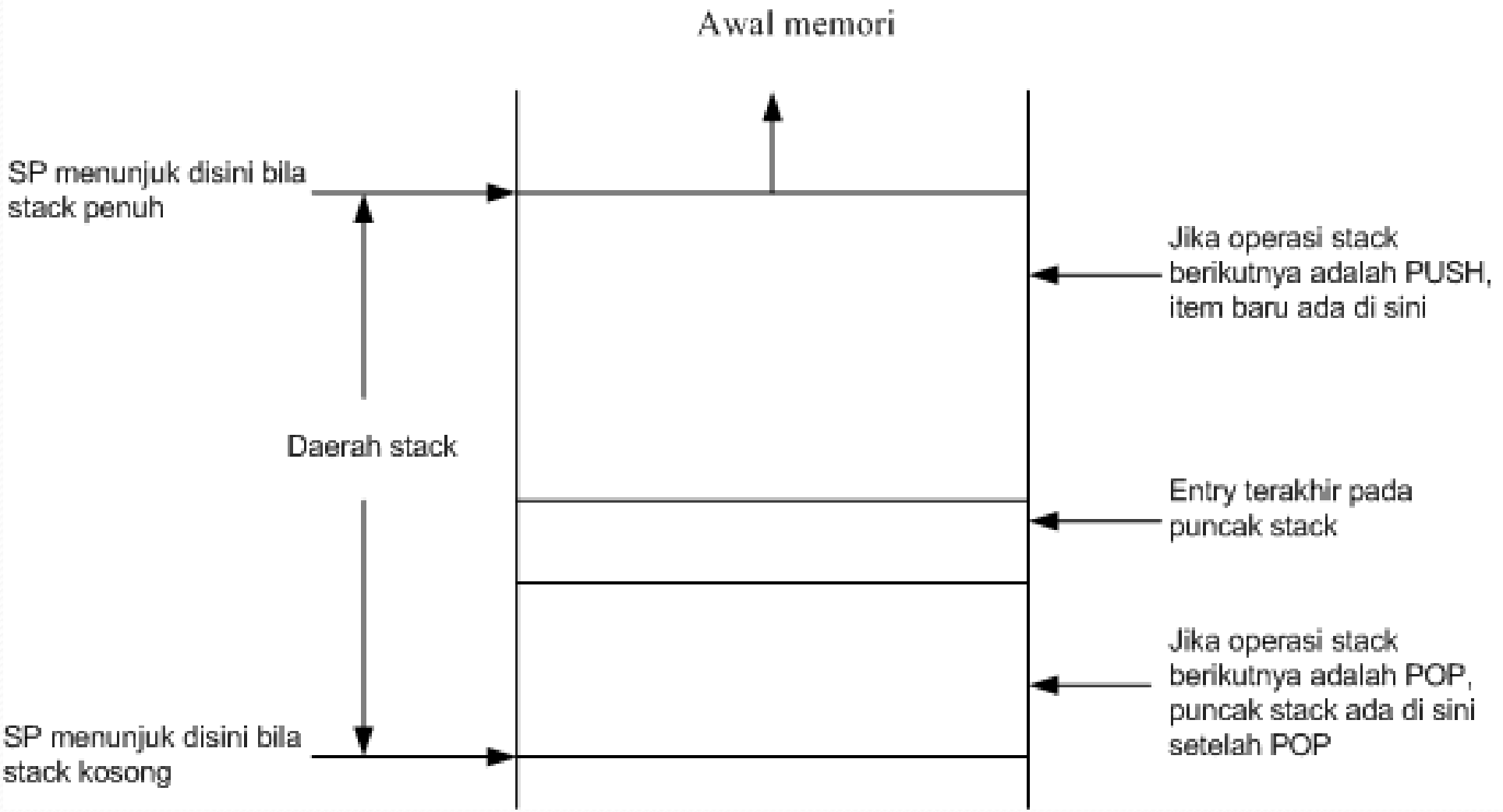
  STORE,X : simpan hasil di dalam lokasi memori X

- Dibandingkan dengan contohı terlihat bahwa CPU berbasis register (arsitektur GPR) menghasilkan ukuran program yang lebih pendek daripada CPU berbasis akumulator.
- Dan juga program pada CPU yang berbasis akumulator memerlukan lokasi memori untuk menyimpan asil sementara(parsial).
- Karena itu diperlukan akses memori tambahan selama eksekusi program.
- Jadi, penambahan jumlah register akan menambah efisiensi CPU.

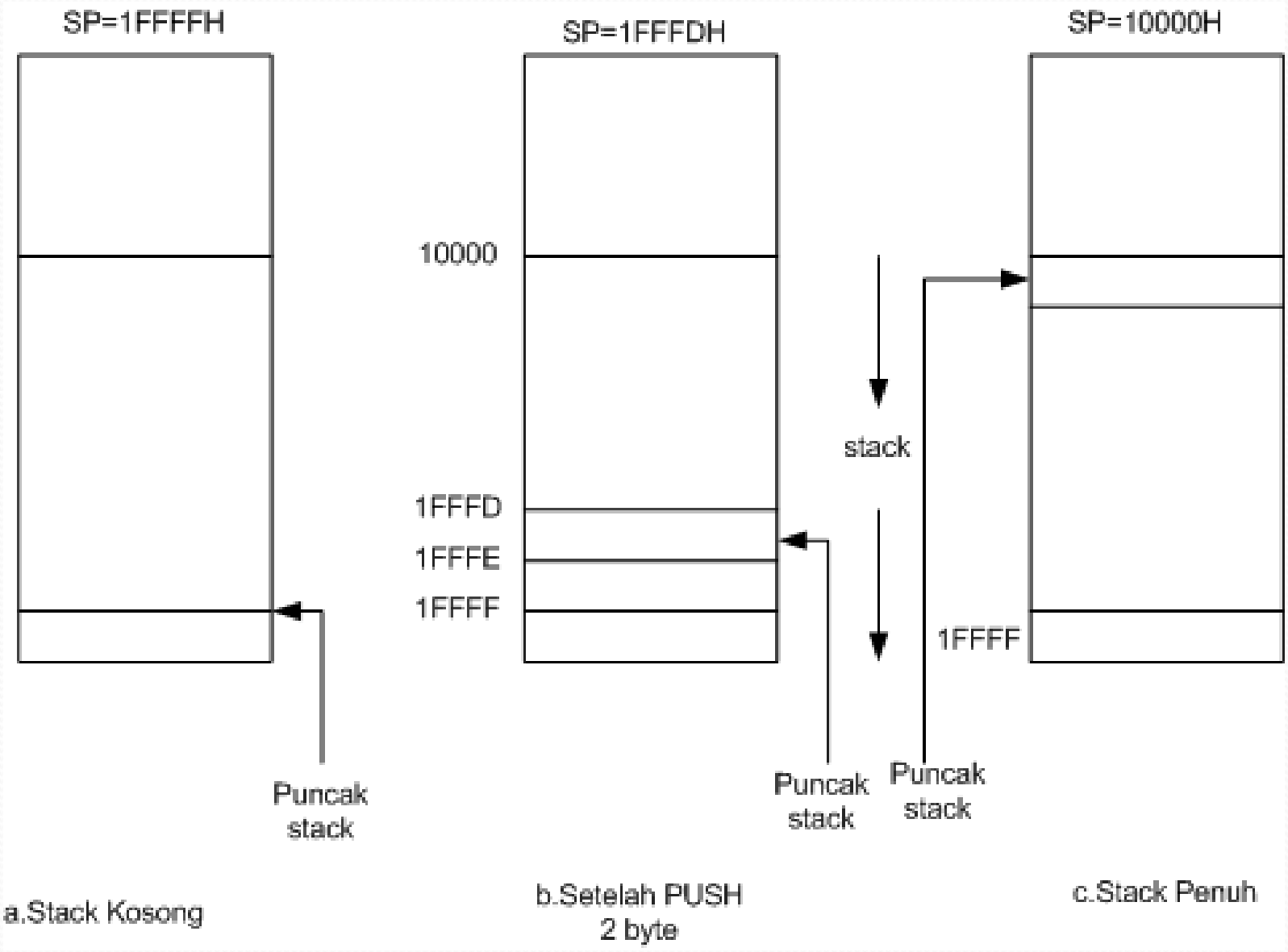
3.CPU Berbasis Stack

- Stack merupakan daftar yang didorong ke bawah dengan mekanisme akses LIFO (*Last In Firs Out*).
- Stack yang menyimpan operand-operand.
- Penggunaan stack dapat berada di dalam CPU atau merupakan bagian dari memori.
- Suatu register(atau lokasi memori) digunakan untuk menunjuk ke alamat lokasi kosong pada puncak stack. Register ini dikenal dengan *Stack Pointer (SP).*
- Bila tidak ada yang tersimpan dalam stack, berarti stack kosong dan SP menunjuk ke bagian stack terbawah.
- Bila suatu item disimpan di dalam stack, maka dinamakan oerasi PUSH; dan isi SP diturunkan (*decrement*)
- Bila stack penuh, SP menunjuk ke bagian stack teratas.
- Bila suatu item diambil dari satck(operasi POP),maka SP dinaikkan (*uncrement).*

>> Konsep Stack



>> Operasi Stack



- Keterangan konsep stack & operasi stack:

- Item yang terakhr didorong ke dalam stack, akan keluar pertama jika ada operasi POPberikutnya.
- Pada CPU berbasis stack, semua operasi oleh CPU dikerjakan pada isi stack.
- Demikian halnya,hasil suatu operasi juga disimpan pada stack.
- Pada eksekusi suatu instruksi aritmetika seperti ADD, operand-operand teratas yang di-*pop*
- Komputer Burroughs B5000 dan HP 3000 merupakan contoh dari komputer berbasis stack.

Contoh3:

- Tuliskan sebuah program bahasa rakitan dalam arsitektur CPU berbasis stack untuk menyelesaikan statement **X=(A+B)-(C+D)**

Statement	Isi Stack setelah eksekusi instruksi	Lokasi stack yang diduduki
PUSH A	A	1
PUSH B	A,B	2
ADD	A+B	1
PUSH C	(A+B),C	2
PUSH D	(A+B),C,D	3
ADD	(A+B),(C+D)	2
SUB	(A+B),(C+D)	1
POP X	Kosong	0

- Dari isi stack, dapat dilihat bahwa stack berubah bil beberapa operasi PUSH mendapat tempat.
- Bila suatu instruksi dieksekusi, operand dipindahkan dari stack dan hasilnya menempati posisi pada puncak stack.
- Contoh3 menunjukkan bahwa ukuran program untuk komputer berbasis stack lebih besar dibandingkan dengan CPU berbasis register.

- **Keuntungan CPU berbasis Stack adalah:**

1. Pemrograman mudah/efisiensi compiler tinggi
2. Sangat cocok untuk bahasa-bahasa blok-terstruktur (block-structered language)
3. Instruksi tidak mempunyai fieldalamat; instruksi pendek.

- **Kelemahan CPU berbasis Stack:**

1. Diperlukan sirkuit hardware tambahan untuk implementasi stack
2. Ukuran program meningkat.

4. Panjang Istruksi

- **Intruksi yang terlalu panjang mempunyai kekurangan:**
  1. Instruksi menempati ruang memori yang lebih besar, yang meningkatkan kebutuhan memori sistem
  2. lebar luas data besar atau pengambilan instruksi lebih memakan waktu. Kondisi pertama menambah biaya hardware sedangkan yang kedua menambah waktu siklus instruksi.
- **Instruksi yang terlalu pendek mempunyai kekurangan:**

1. Terlalu banyak instruksi di dalam program. Karena itu banyak waktu yang terbuang untuk fase pengambilan
2. Ukuran program bertambah. Karena itu kebutuhan memori bertambah.

5. Format Instruksi

- Umumnya format instruksi terdiri dari kode operasi dan operand.
- Suatu instruksi memberikan palng banyak empat informasi pada CPU :

1. Operasi yang akan dikerjakan oleh instruksi
2. Operand (data) yang harus dioperasikan
3. Lokasi (memori atau register) di mana hasil operasi harus disimpan
4. Lokasi memori dimana instruksi berikutnya harus diambil

ADD opcode	Alamat Operand 1	Alamat Operand2	Alamat hasil	Alamat instruksi berikutnya

Instruksi Empat-alamat

contoh4

- Medan panjang instruksi dan medan alamat operand masing-masing adalah 36 bit dan 14 bit. Jika instruksi dua-operand yang digunakan sebanyak 240, berapa banyak instruksi satu-operand yang memungkinkan?
- **Solusi:**

  Panjang instruksi = 36 bit

  sebuah instruksi dua-operand membutuhkan 28 bit untuk alamat operand (2x14 bit).karena itu

  Ukuran opcode=36-28 = 8bit

  Total instruksi yang mungkin = 2 pangkat 8 = 256

  Jumlah instruksi satu-operand= 256-240 = 16

Selesai