Struktur dan Fungsi CPU
Pertemuan 9

Oleh :

Riyanto Sigit, S.T, M.Kom

Nur Rosyid Mubtada'i S.Kom

Setiawardhana , S.T

Hero Yudo Martono, S.T

Politeknik Elektronika Negeri Surabaya - ITS

2005

Tujuan

- Mengerti struktur dan Fungsi CPU yaitu dapat melakukan Fech Instruksi, Interpreter instuksi, Fech data, exekusi, dan menyimpan kembali. Serta struktur dari register macam-macam register dan fungsinya
- Mengerti aliran data pada siklus pengambilan, siklus tak langsung, siklus interupt, Mengerti pipelining, dan mengerti teknik-teknik menangani percabangan pada pipelining

Materi

- Bagian ini membahas aspek – aspek struktur dan fungsi CPU untuk dasar pembahasan bab berikutnya, yaitu RISC.
- Fokus bab struktur dan fungsi CPU adalah organisasi prosesor dan register, siklus instruksi dan strategi dalam metode pipelining

4.3. Siklus Instruksi

- Ada beberapa sub-siklus
- Apa saja ?

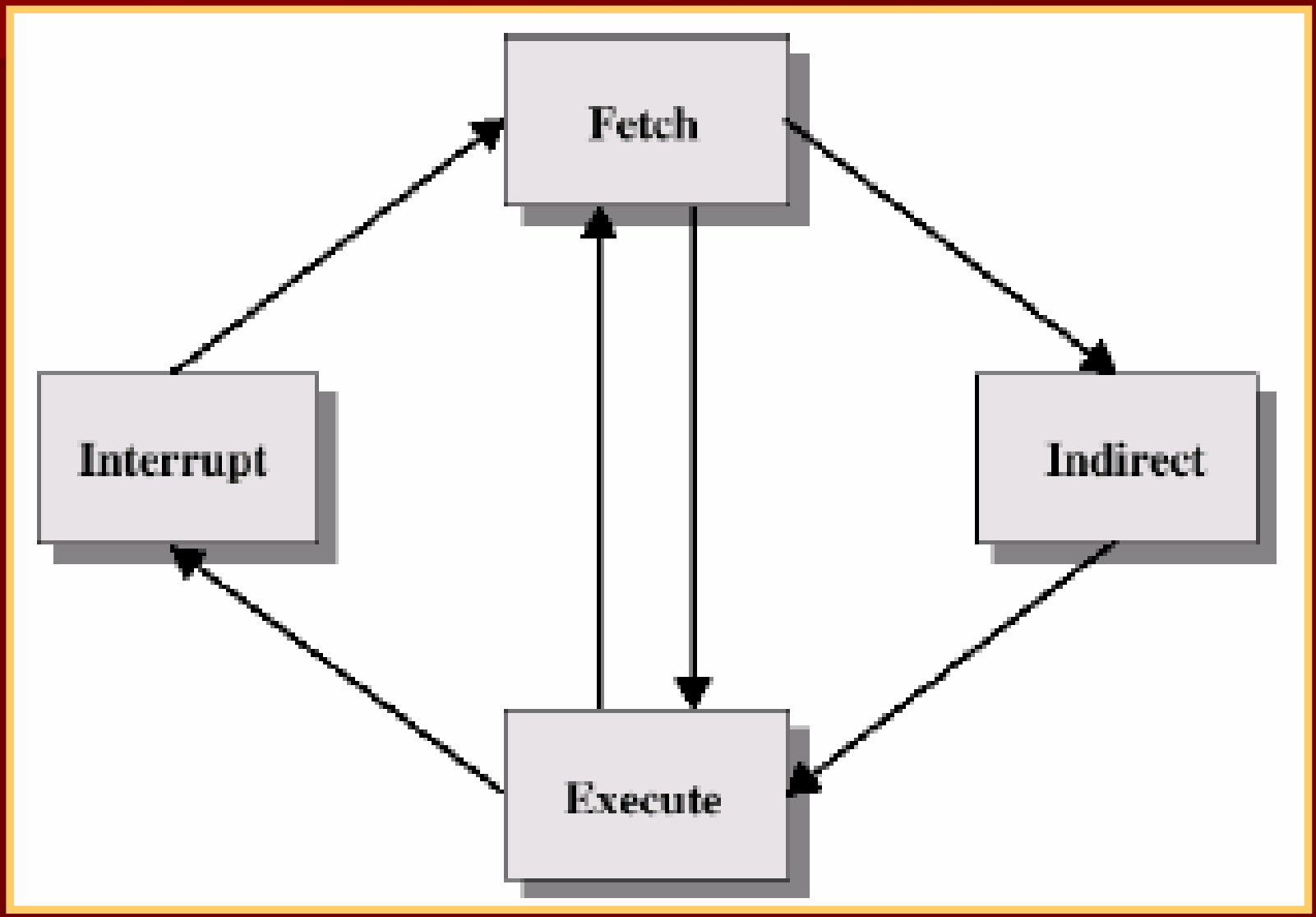
Sub-siklus instruksi

- Fetch
  - Adalah siklus pengambilan data ke memori atau register
- Execute
  - Menginterpretasikan opcode dan melakukan operasi yang diindikasikan
- Interrupt
  - Apabila interrupt diaktifkan dan interrupt telah terjadi, simpan status proses saat itu dan layani interupsi

Siklus Tidak Langsung , Apa itu ?

- Eksekusi sebuah instruksi melibatkan sebuah operand atau lebih di dalam memori, yang masing-masing operand memerlukan akses memori
- Pengambilan alamat-alamat tak langsung dapat dianggap sebagai sebuah subsiklus instruksi atau lebih

Siklus instruksi

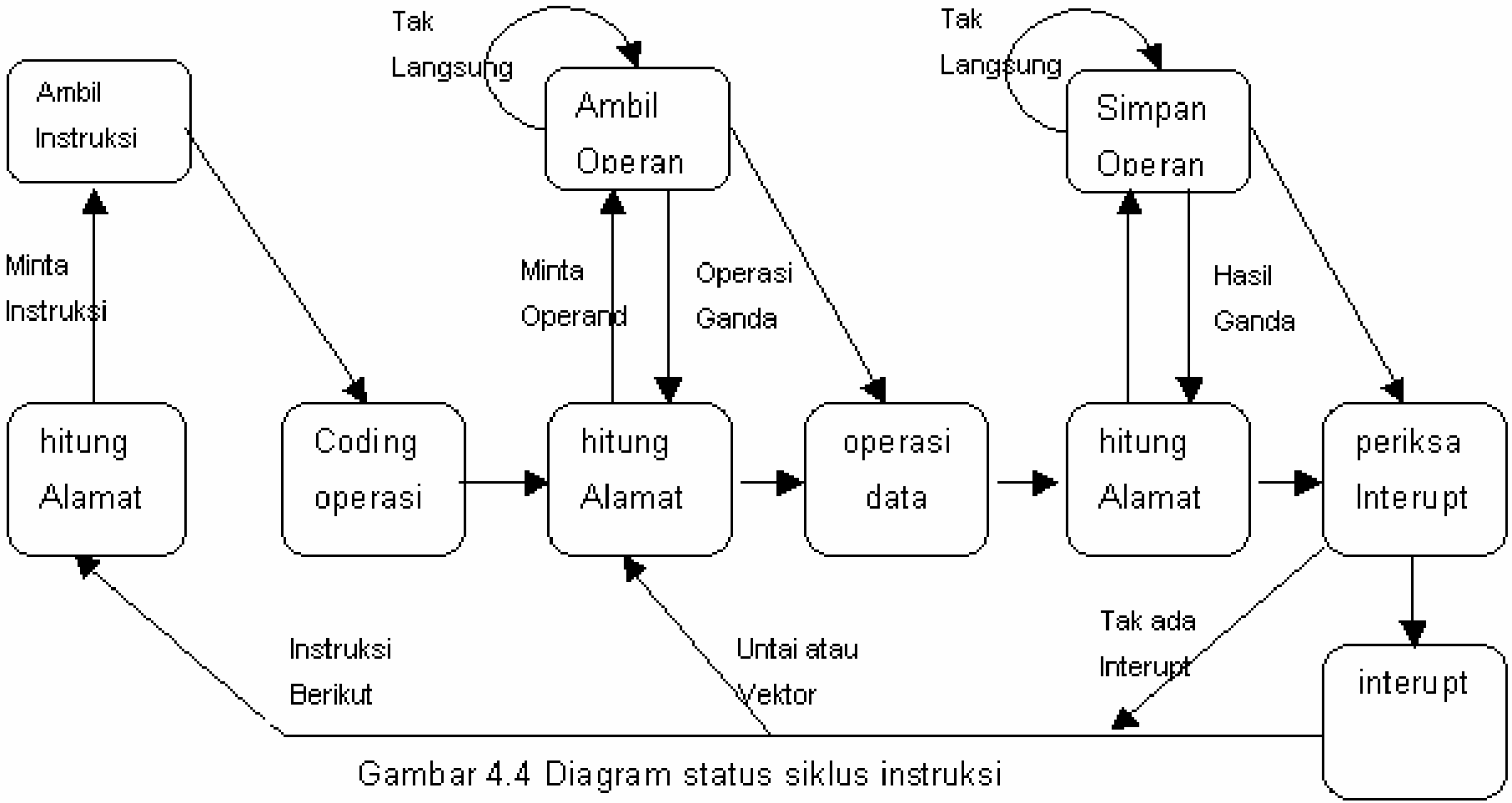


- Apakah ada cara pandang yang lain ?
  - ADA
- Bagaimana ?

Sifat siklus instruksi

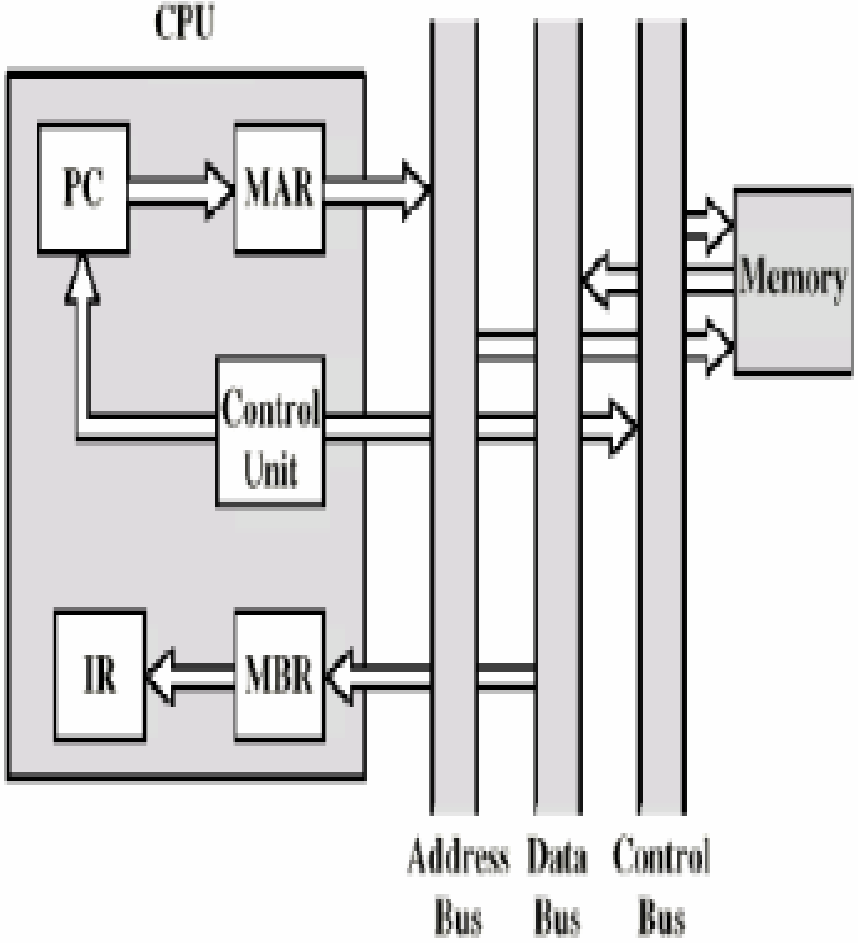
- Sekali instruksi telah diambil, maka operand specifier-nya harus diidentifikasikan.
- Kemudian seluruh operand input yang berada di dalam memori akan diambil, dan proses ini mungkin memerlukan pengalamatan tak langsung.
- Operand berbasis register tidak perlu diambil.
- Apabila opcode telah dieksekusi, proses yang sama akan diperlukan untuk menyimpan hasilnya di dalam memori

Diagram status siklus instruksi



Gambar 4.4 Diagram status siklus instruksi

Aliran data siklus pengambilan

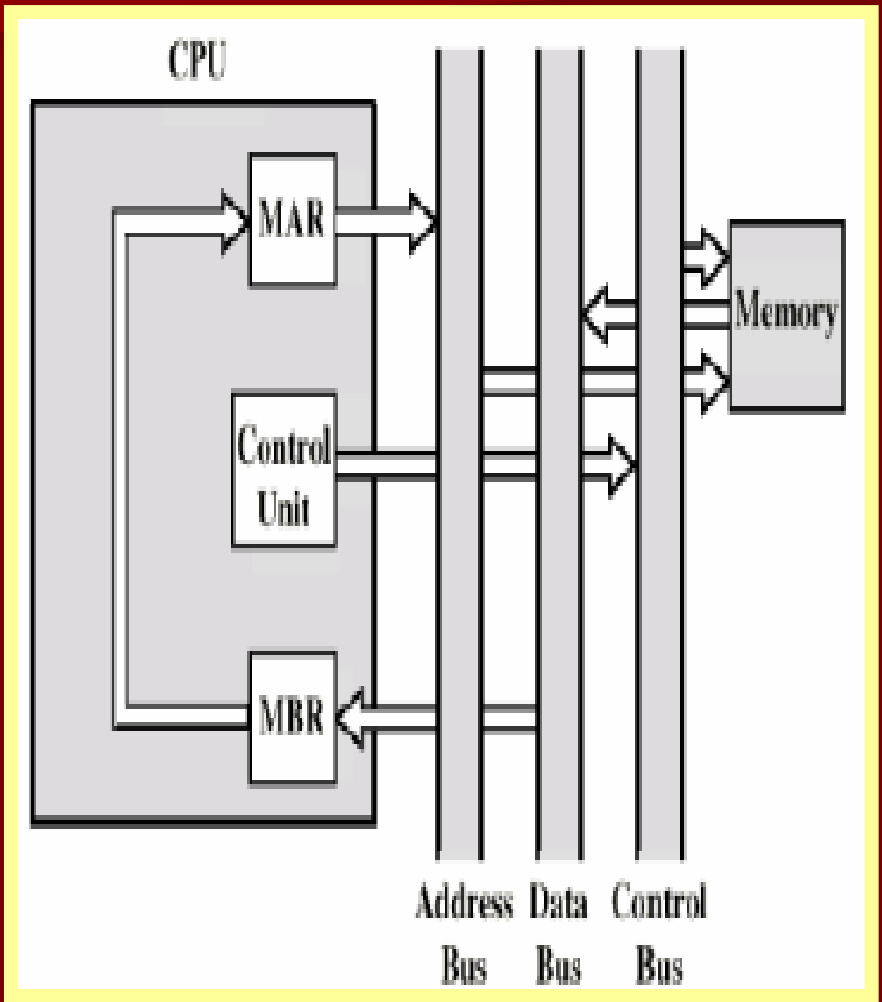


- Urutan kejadian selama siklus instruksi tergantung pada rancangan CPU.
- Asumsi: sebuah CPU yang menggunakan register memori alamat (MAR), register memori buffer (MBR), pencacah program (PC), dan register instruksi (IR).

Prosesnya :

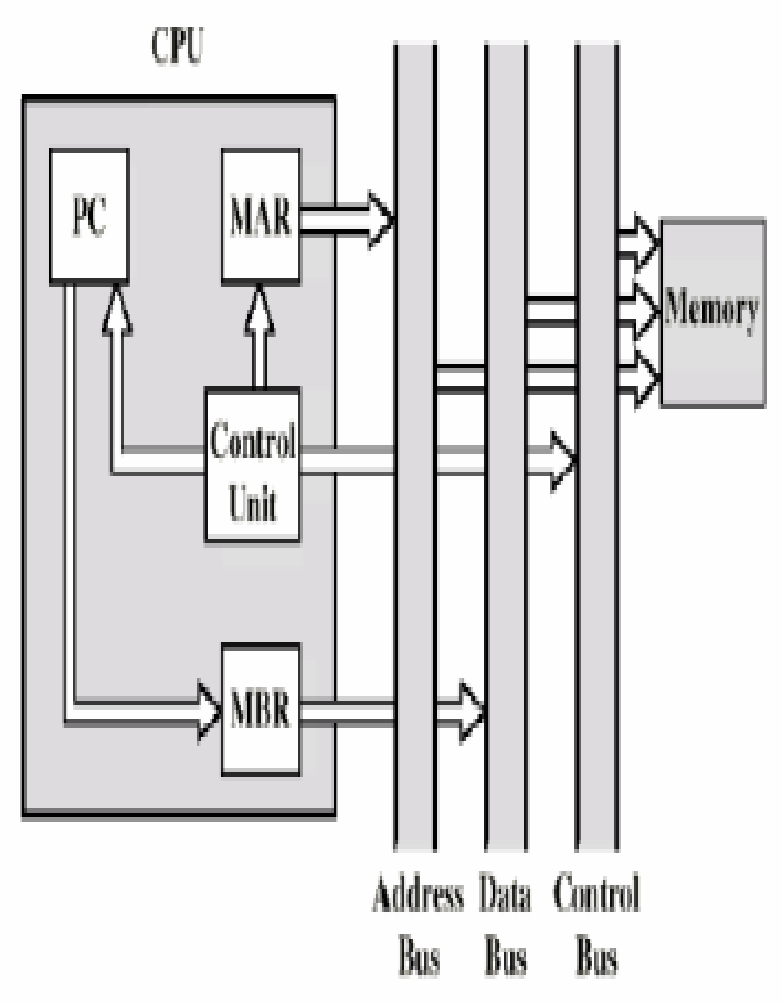
- Pada saat siklus pengambilan (fetch cycle), instruksi dibaca dari memori.
- PC berisi alamat instruksi berikutnya yang akan diambil.
- Alamat ini dipindahkan ke MAR dan ditaruh di bus alamat.
- Unit kontrol meminta pembacaan memori dan hasinya disimpan di bus data dan disalin ke MBR dan kemudian dipindahkan ke IR.
- PC naik nilainya 1, sebaai persiapan untuk pengambilan selanjutnya.
- siklus selesai, unit kontrol memeriksa isi IR untuk menentukan apakah IR berisi operand specifier yang menggunakan pengalamatan tak langsung

Aliran data siklus tak langsung



- N bit paling kanan pada MBR, yang berisi referensi alamat, dipindahkan ke MAR.
- Unit kontrol meminta pembacaan memori, agar mendapatkan alamat operand yang diinginkan ke dalam MBR
- Siklus pengambilan dan siklus tak langsung cukup sederhana dan dapat diramalkan.
- Siklus instruksi (instruction cycle) mengambil banyak bentuk karena bentuk bergantung pada bermacam-macam instruksi mesin yang terdapat di dalam IR.
- Siklus meliputi pemindahan data di antara register-register, pembacaan atau penulisan dari memori atau I/O, dan atau penggunaan ALU

Aliran data siklus interupsi

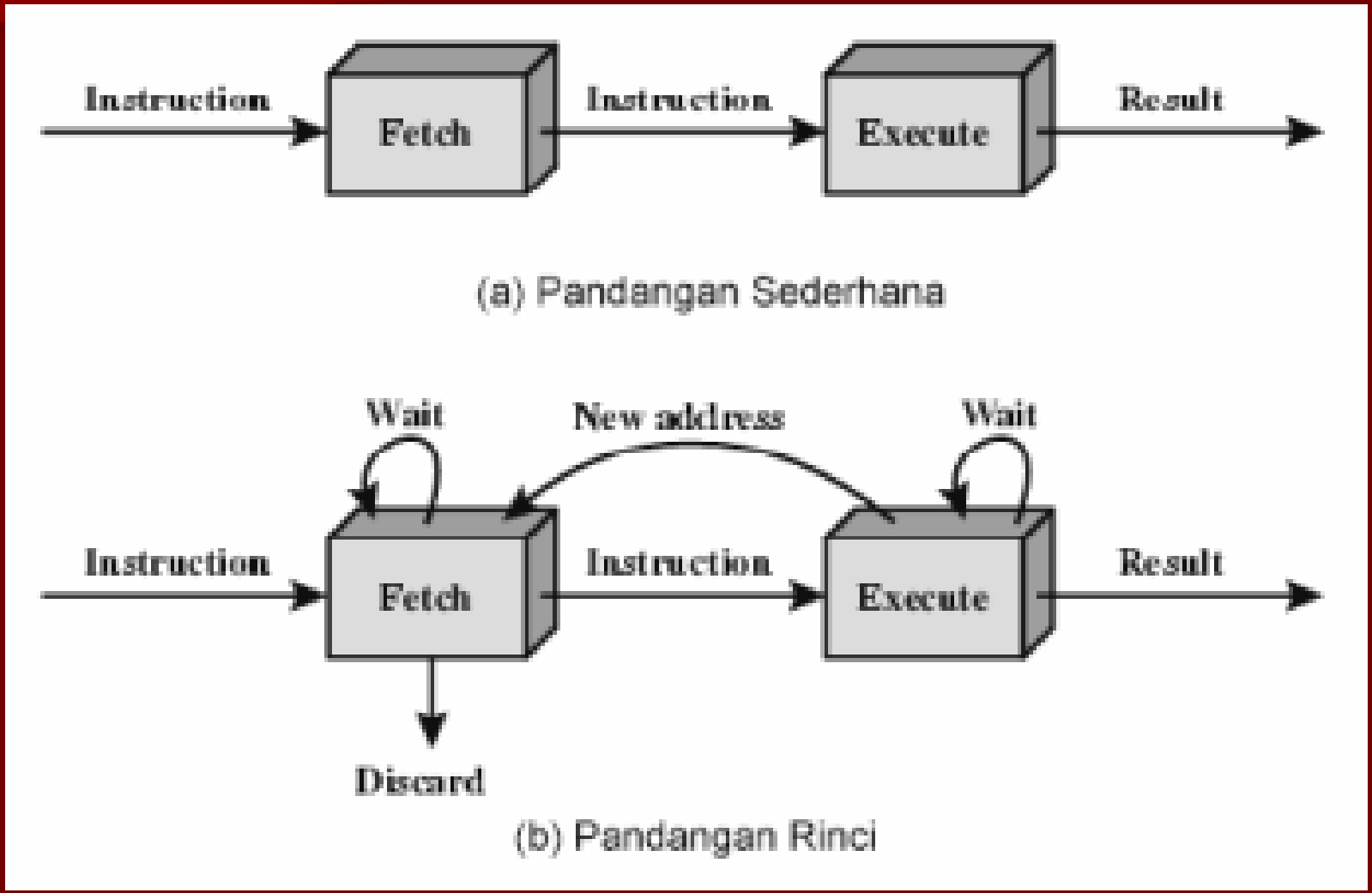


- Isi PC saat itu harus disimpan sehingga CPU dapat melanjutkan aktivitas normal setelah terjadinya interrupt.
- Cara: isi PC dipindahkan ke MBR untuk kemudian dituliskan ke dalam memori.
- Lokasi memori khusus yang dicadangkan untuk keperluan ini dimuatkan ke MAR dari unit kontrol.
- Lokasi ini berupa stack pointer.
- PC dimuatkan dengan alamat rutin interrupt.
- Akibatnya, siklus instruksi berikutnya akan mulai mengambil instruksi yang sesuai

4.4. Strategi Pipelining

- Pipelining, Apa itu ?
  - Input baru akan diterima pada sebuah sisi sebelum input yang diterima sebelumnya keluar sebagai output di sisi lainnya
- Pendekatan :
  - Pipelining instruksi mirip dengan penggunaan rangkaian perakitan pada pabrik.
  - Rangkaian perakitan memanfaatkan kelebihan yang didapat dari fakta bahwa suatu produk diperoleh dengan melalui berbagai tahapan produksi.
  - Dengan menaruh proses produksi di luar rangkaian perakitan, maka produk yang berada diberbagai tahapan dapat bekerja secara bersamaan.

Pipeline instruksi dua tahap



Pipeline ? Pengolahan instruksi

1. Pengambilan instruksi
2. Pengeksekusian instruksi.

- Terdapat waktu yang dibutuhkan selama proses eksekusi sebuah instruksi pada saat memori sedang tidak diakses.
- Waktu ini dapat digunakan untuk mengambil instruksi berikutnya secara paralel (bersamaan) dengan eksekusi instruksi saat itu

Tahapan Pipeline

- Tahapannya Independen
- Mengapa ?
  - Tiap tahapan bekerja sendiri
  - Kedua bekerja dalam waktu yang bersamaan
- Ada berapa ?

- Ada 2 tahap
- Tahapan pertama mengambil instruksi dan mem-buffer-kannya.
- Ketika tahapan kedua bebas, tahapan pertama mengirimkan instruksi yang di-buffer-kan tersebut.
- Pada saat tahapan kedua sedang mengeksekusi instruksi, tahapan pertama memanfaatkan siklus memori yang tidak dipakai untuk mengambil dan membufferkan instruksi berikutnya.
  - Proses ini disebut instruction prefetch atau fetch overlap

Efek Pipeline

- Mempercepat eksekusi instruksi.
- Apabila tahapan pengambilan dan eksekusi instruksi memerlukan waktu yang sama, maka siklus instruksi akan berkurang menjadi separuhnya

- Penggandaan kecepatan eksekusi tidak akan terjadi apabila beberapa hal terjadi
- Apa saja alasannya ?
- Bagaimana mengatasinya ?

Alasan ?

- Umumnya waktu eksekusi akan lebih lama dibandingkan dengan pengambilan instruksi. Kenapa ?
  - Eksekusi akan meliputi pembacaan dan penyimpanan operand serta kinerja sejumlah operasi sehingga tahapan pengambilan mungkin perlu menunggu beberapa saat sebelum mengosongkan buffer-nya
- Instruksi pencabangan bersyarat akan membuat alamat instruksi berikutnya yang akan diambil tidak diketahui.
  - Tahapan pengambilan harus menunggu sampai menerima alamat instruksi berikutnya dari tahapan eksekusi. Dengan demikian tahapan eksekusi harus menunggu pada saat fetch

Solusi

- Kerugian waktu yang diakibatkan tahapan kedua dapat dikurangi dengan cara :

Menebak = Prediksi

Aturan Prediksi ?

- Aturannya sederhana
  - Instruksi pencabangan bersyarat dikirimkan dari tahapan pengambilan ke tahapan eksekusi, tahapan pengambilan mengambil instruksi berikutnya di dalam memori setelah terjadinya instruksi pencabangan itu.
  - Apabila pencabangan tidak dilakukan, maka tidak akan terdapat watu yang hilang.
  - Apabila pencabangan dilakukan, instruksi yang diambil harus dibuang dan instruksi yang baru harus diambil

- Faktor-faktor di atas mengurangi efektivitas pipeline dua tahap, namun terjadi juga beberapa percepatan.
- Untuk memperoleh percepatan lebih lanjut, pipeline harus memiliki lebih banyak tahapan

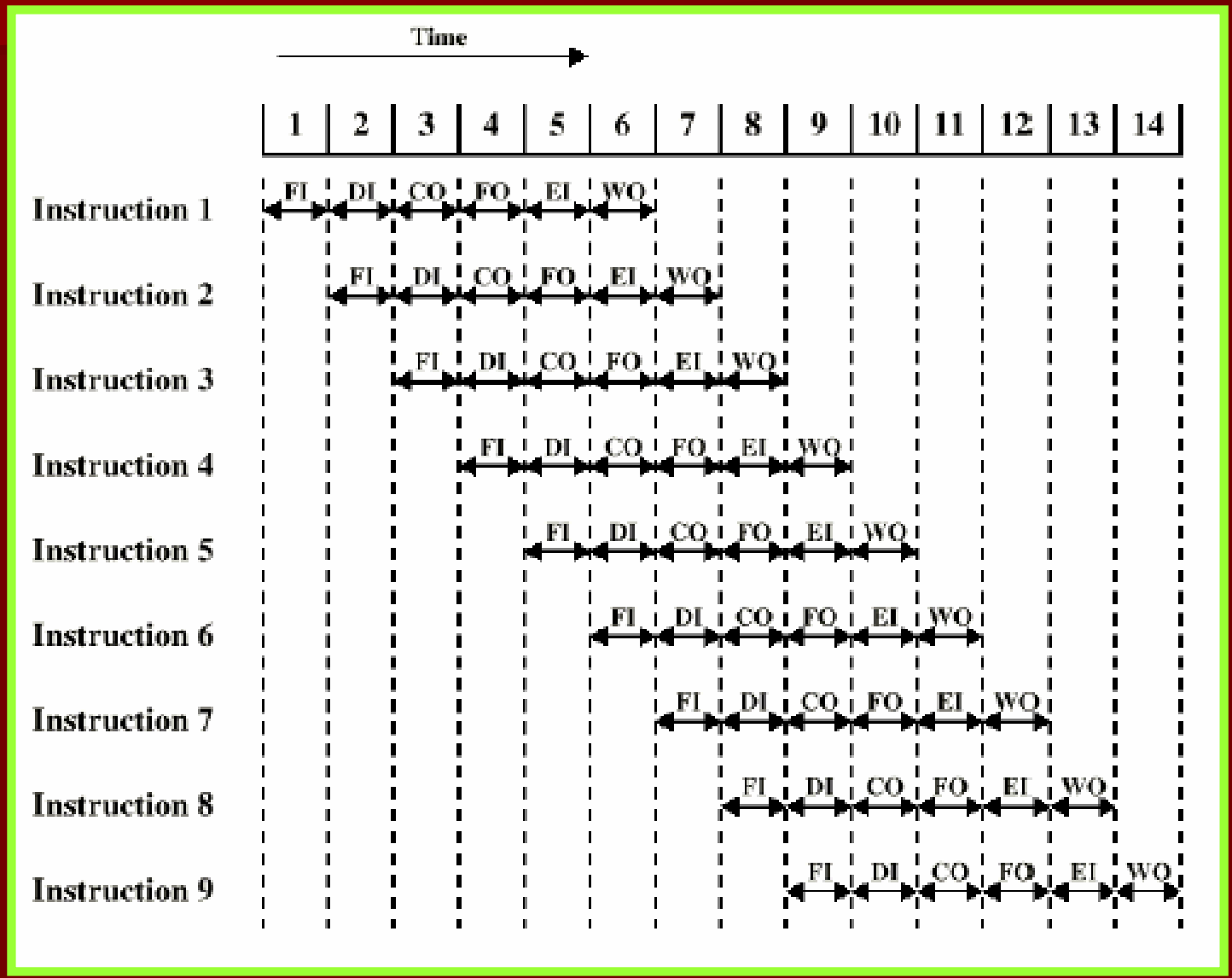
Dekomposisi pengolahan instruksi

- Fetch Instruction (FI)
  - membaca instruksi berikutnya ke dalam buffer
- Decode Instruction (DI)
  - menentukan opcode dan operand specifier
- Calculate Operand (CO)
  - menghitung alamat efektif seluruh operand sumber. Hal ini mungkin melibatkan displacement, register indirect, atau bentuk kalkulasi alamat lainnya
- Fetch Operand (FO)
  - mengambil semua operand dari memori. Operand-operand yang berada di register tidak perlu diambil
- Execute Instruction (EI)
  - melakukan operasi yang diindikasikan dan menyimpan hasilnya
- Write Operand (WO)
  - menyimpan hasilnya di dalam memori

Efek dekomposisi diatas apa ?

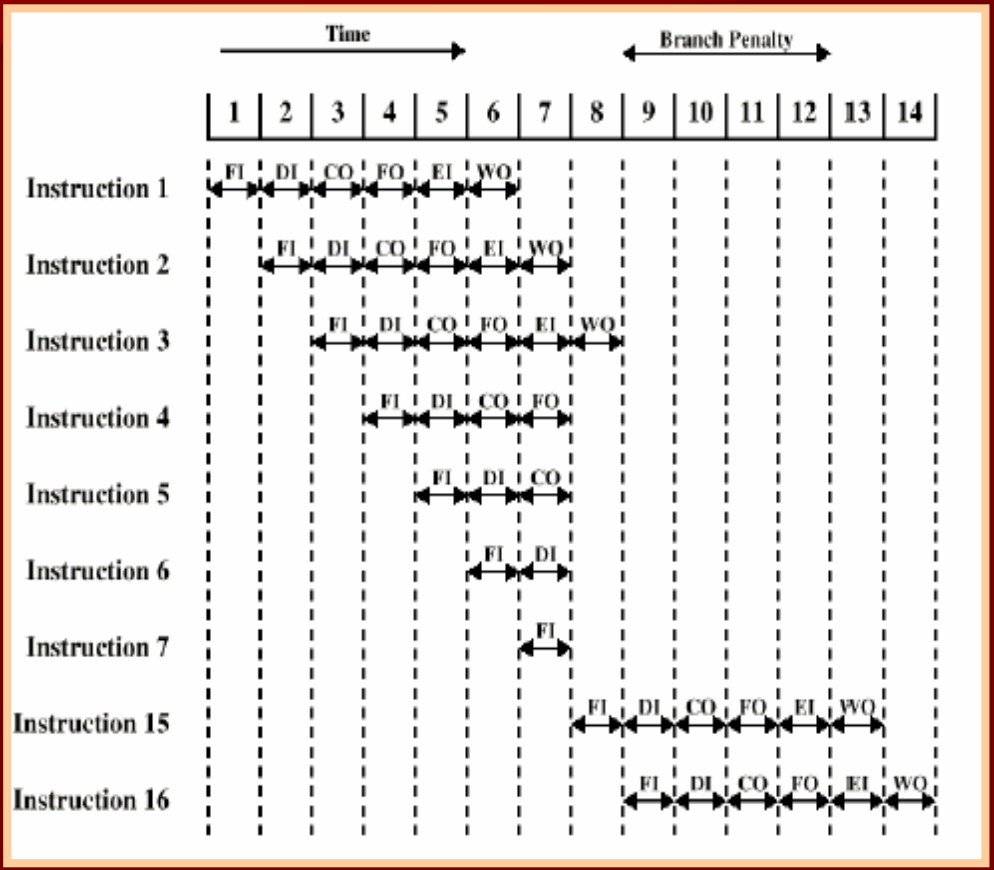
- Bermacam tahapan dapat memiliki durasi yang hampir sama
- Contoh bagaimana ?
  - Pipeline enam tahap dapat mengurangi waktu eksekusi 9 buah instruksi dari 54 satuan waktu menjadi 14 satuan waktu
  - Bagaimana gambar prosesnya ?

Diagram pewaktuan operasi



- Faktor menghambat peningkatan kinerja ?
  - Keenam tahapan memiliki durasi yang tidak sama, terjadi waktu tunggu pada beberapa tahapan pipeline
  - Instruksi pencabangan bersyarat, yang dapat mengagalkan beberapa pengambilan instruksi

Contoh "Branch"



Asumsi:

- Instruksi 3 adalah pencabangan bersyarat instruksi 15.
- Sampai saat instruksi dieksekusi, tidak terdapat cara untuk mengetahui instruksi mana yang akan terjadi kemudian.
- Instruksi 4 sampai 14 tidak dilakukan eksekusi sehingga data harus dibersihkan dari jalurnya.
- Eksekusi dilanjutkan saat pencabangan ke instruksi 15 sudah sampai

Rancangan Pipeline IBM S/360

- Ada 2 faktor hambatan , APA ?
- Masih terjadi sampai sekarang

1. Setiap tahapan pipeline terdapat sejumlah overhead yang terjadi pada pemindahan data dari buffer ke buffer dan pada saat melakukan persiapan dan pengiriman fungsi – fungsi. Overhead akan memperpanjang waktu eksekusi instruksi tunggal. Pertambahan waktu ini akan makin terasi apabila instruksi saling tergantung secara logika
2. Jumlah kontrol logika yang diperlukan untuk menangani ketergantungan memori dan register akan meningkat seiring banyaknya tahapan. Hal ini menyebabkan kerumitan dan waktu fungsi pengontrolan

Penanganan Pencabangan

- Untuk apa ?
  - Menjamin terjadinya aliran instruksi yang stabil
  - Kestabilan akan terganggu saat instruksi mengalami pencabangan karena belum bisa ditentukan tujuan pencabangan tersebut
  - Beberapa metode pendekatan masalah digunakan untuk mengatasi hal tersebut ?

Teknik pendekatan

1. Multiple Streams
2. Prefetch branch target
3. Loop buffer
4. Branch prediction
5. Delayed branch

1. Multiple Streams

- Kedua instruksi percabangan diambil dengan dua buah stream.

Kelemahan :

- Adanya persaingan dalam mengakses register dan memori untuk dimasukkan dalam pipeline.
- Bila dalam percabangan terdapat percabangan lagi, tidak mampu ditangani oleh dua stream.
- Walaupun terdapat kelemahan tapi terbukti meningkatkan kinerja pipelining.
  - Teknik ini diterapkan pada IBM 370/168 dan IBM 3033

2. Prefetch branch target

- Apabila pencabangan bersyarat telah diketahui

Prosesnya :

- Dilakukan pengambilan awal (prefetch) terhadap instruksi setelah pencabangan dan target pencabangan.
  - Diterapkan pada IBM 360/91.

Masalah :

- Diperlukan buffer dan register untuk prefetch

3. Loop buffer

- Apabila terdapat pencabangan maka perangkat keras memeriksa apakah target pencabangan telah ada dalam buffer, bila telah ada maka instruksi berikutnya diambil dari buffer.
- Perbedaan dengan prefetch adalah pada loop buffer akan membuffer instruksi ke depan dalam jumlah yang banyak, sehingga bila target tidak berjauhan lokasinya maka secara otomatis telah terbuffer.
- Terkesan teknik ini seperti cache memori, namun terdapat perbedaan karena loop buffer masih mempertahankan urutan instruksi yang diambilnya

4. Branch prediction

- Penganalisaan sejarah instruksi. Kenapa ?
- Instruksi komputer seringkali terjadi berulang.

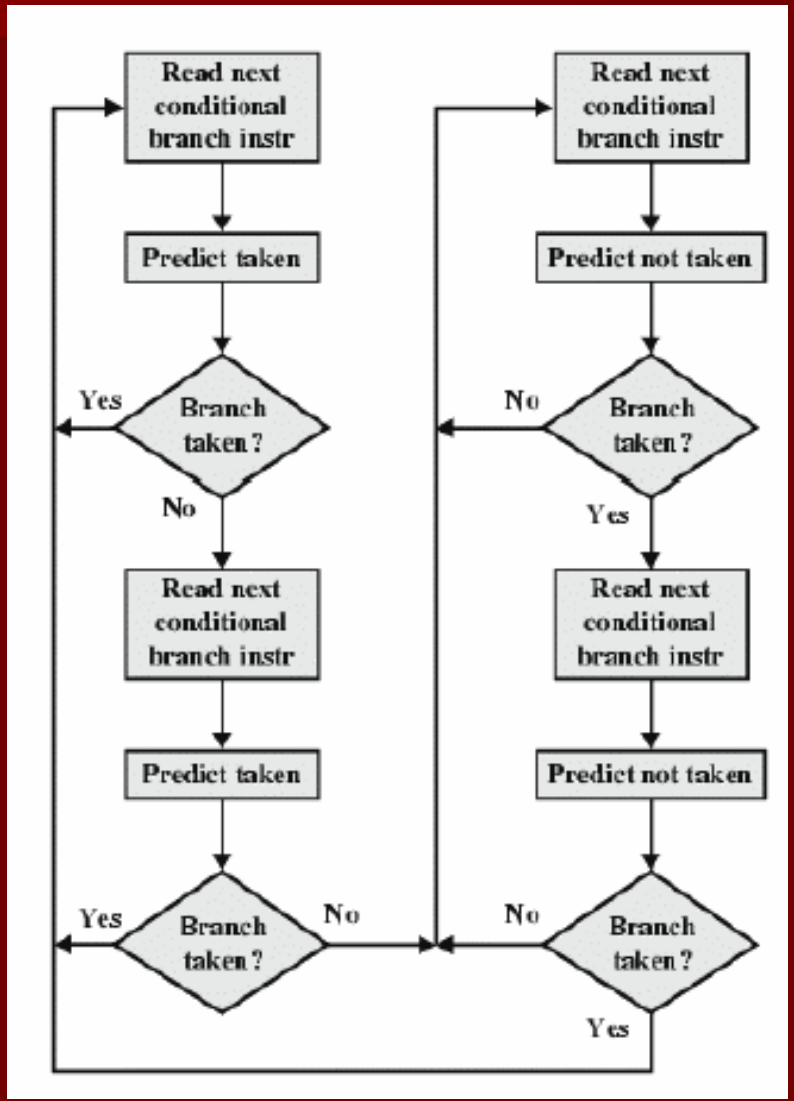
Sehingga :

- Teknik prediksi ini juga diterapkan dalam pengambilan instruksi pada cache memori.
- Diperlukan algoritma khusus untuk melakukan prediksi tersebut.
- Patokan memprediksi target pencabangan
  - Penganalisaan eksekusi – eksekusi yang telah terjadi dan aspek lokalitas.
  - Aspek lokalitas memori adalah kecenderungan penyimpanan instruksi yang berhubungan dalam tempat yang berdekatan

5. Delayed branch

- Eksekusi pada tahapan pipeline yang melibatkan pencabangan akan dilakukan penundaan proses beberapa saat sampai didapatkan hasil pencabangan.
- Namun tahapan pipelining lainnya dapat berjalan seiring penundaan tersebut.
- Teknik penundaan ini menggunakan instruksi NOOP

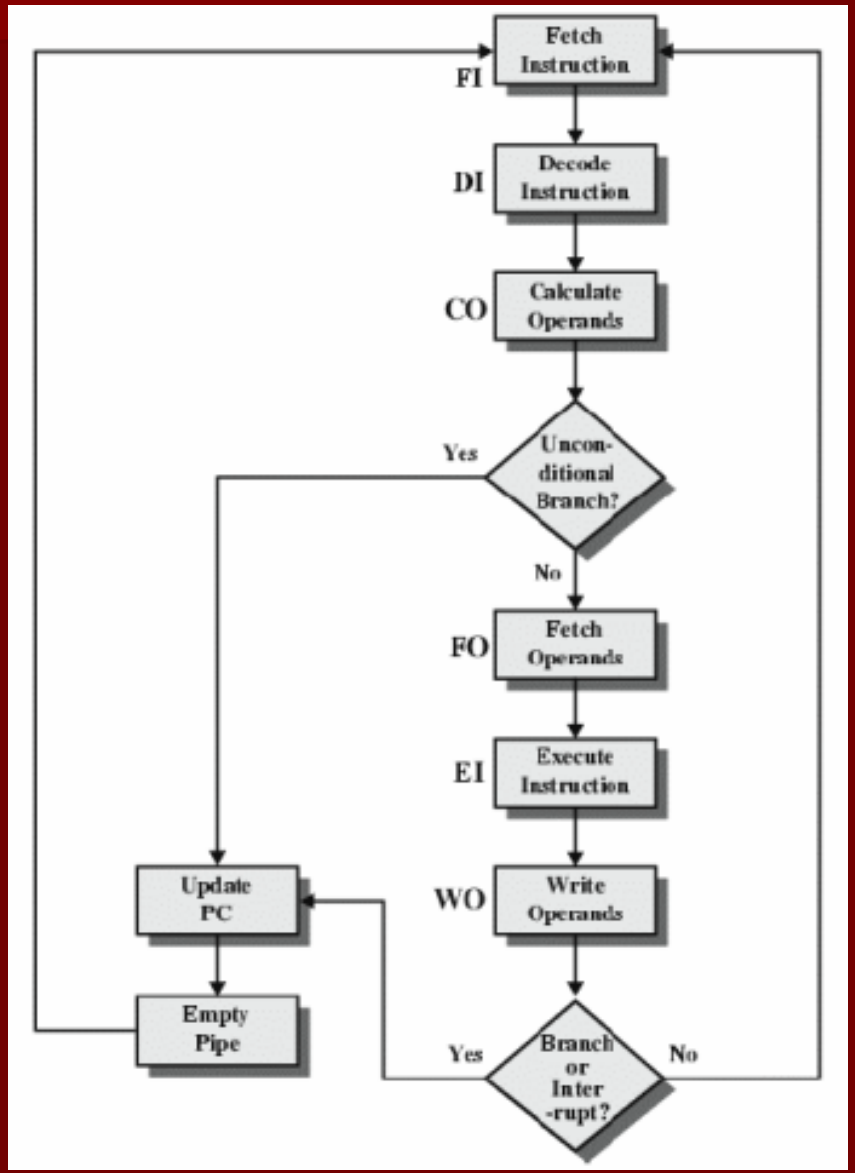
Diagram alir prediksi



Penanganan Interupsi

- Interupsi adalah fasilitas yang disediakan untuk mendukung sistem operasi
- Pengolahan interupsi memungkinkan suatu program aplikasi dapat ditahan, agar kondisi – kondisi insterupsi dapat dilayani, kemudian program dilanjutkan.
- Interupsi adalah suatu masalah dalam pipeline, karena mengganggu aliran instruksi yang telah tersusun.
- Hal yang sering dilakukan dalam perancangan adalah interupsi ditangguhkan beberapa saat sampai program utama mendapatkan titik pemberhentian.
- Hal ini efektif dilakukan sehingga tidak terlalu mengganggu pipelining instruksi.

Diagram alir penanganan



Interupsi ?

- Ada 2 macam :
  1. Interupsi yang dilakukan oleh perangkat keras
  2. Interupsi dari program (Exception)

Kesimpulan

- Syarat agar bisa disebut CPU adalah bisa mengambil instruksi, menterjemahkan, mengambil data, mengolah dan menyimpan kembali
- CPU dibantu memori internal yang disebut register
- Terdapat dua group register yaitu register yang dapat diakses oleh programer (user visible register) dan register yang tidak bisa diakses oleh programer (control status word)
- Siklus instruksi terdiri dari fetch, execute dan interupt
- Dengan cara pipelining kinerja CPU dapat ditingkatkan

Latihan

- Apa fungsi dari general purpose register, data register, address register ?
- Jelaskan kembali alur data pada siklus pengambilan !
- Jelaskan kembali alur data pada siklus tak langsung !
- Jelaskan kembali alur data pada siklus interupt !
- Mengapa Pipelining mempercepat proses ?
- Bagaimana Mengantisipasi kalau terjadi percabangan saat melakukan pipelining?
- Jelaskan tentang NOOP !