

William Stallings

Computer Organization

and Architecture

Chapter 3

Sistem Bus

(sistem dan struktur interkoneksi komputer)

Konsep Program

- ⌘ Sistem Hardware-nya tidak dapat diubah-ubah
- ⌘ Fungsi kerja hardware dapat melakukan tugas berbeda-beda, memberikan sinyal kontrol yang benar
- ⌘ Daripada melakukan pengawatan baru, lebih baik menyediakan sinyal kontrol yang baru

Apa itu program?

- ⌘ Serangkaian langkah sequensial
- ⌘ Untuk setiap langkah, sebuah operasi aritmetik atau logik
- ⌘ Untuk setiap operasi, diperlukan sinyal kontrol yang berbeda

Fungsi Unit Kontrol

- ⌘ Untuk setiap operasi disediakan kode-kode yang unik
 - ☐ contoh: ADD, MOVE
- ⌘ Bagian hardware akan menerima kode dan mengeluarkan sinyal kontrol

Komponen-komponen (1)

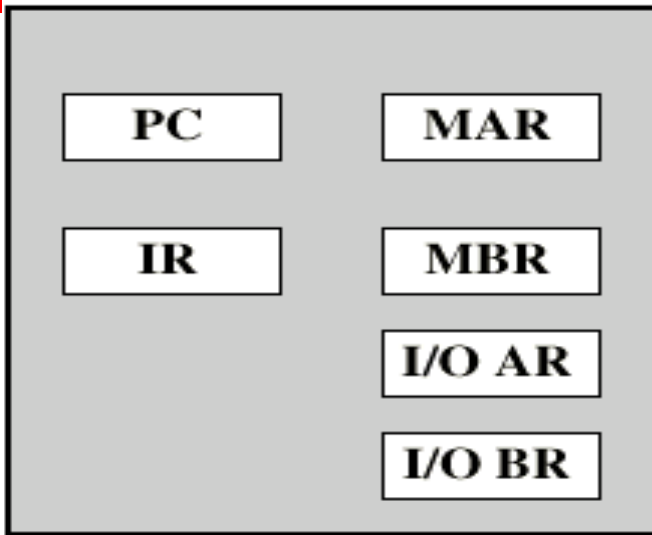
- ⌘ Unit kontrol dan unit aritmetik dan logik merupakan bagian dari CPU
- ⌘ Data dan instruksi memerlukan media untuk masuk ke sistem dan menghasilkan output
 - ☑ Input/output
- ⌘ Tempat penyimpanan sementara kode dan output sangat diperlukan
 - ☑ Main memory

Komponen-komponen (2)

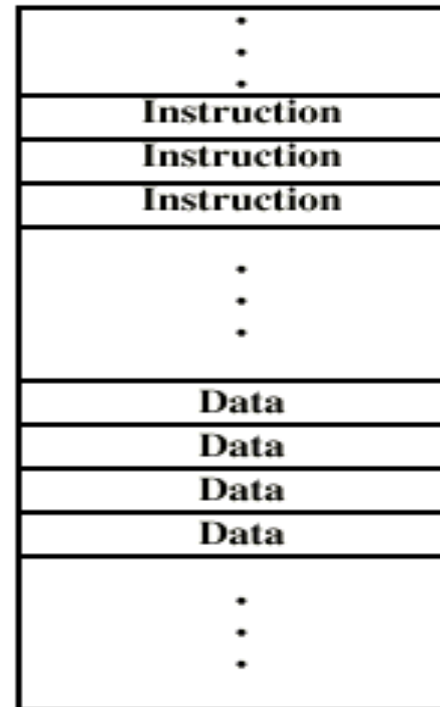
- ⌘ **Prosesor:** mengontrol operasi komputer dan melakukan fungsi pengolahan data. Jika hanya ada satu prosesor, maka disebut CPU
- ⌘ **Memori utama:** menyimpan data dan program
- ⌘ **Modul I/O:** memindahkan data antara komputer dengan lingkungan eksternalnya. Ex. Perangkat memori sekunder, terminal
- ⌘ **Interkoneksi sistem:** Beberapa struktur dan mekanisme yang melakukan komunikasi antara prosesor, memori utama, dan modul I/O

Komponenten Computer: Top Level View

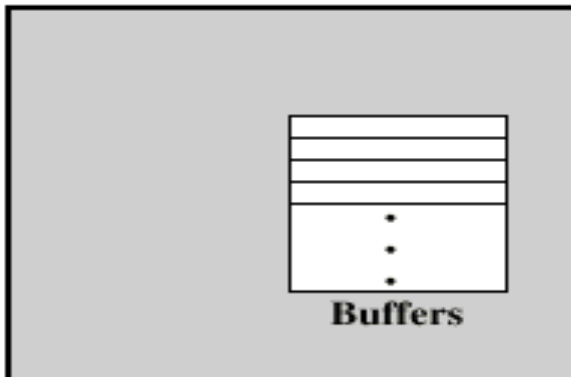
CPU



Memory



I/O Module



- PC** = Program counter
- IR** = Instruction register
- MAR** = Memory address register
- MBR** = Memory buffer register
- I/O AR** = I/O address register
- I/O BR** = I/O buffer register

Komponen-komponen (3)

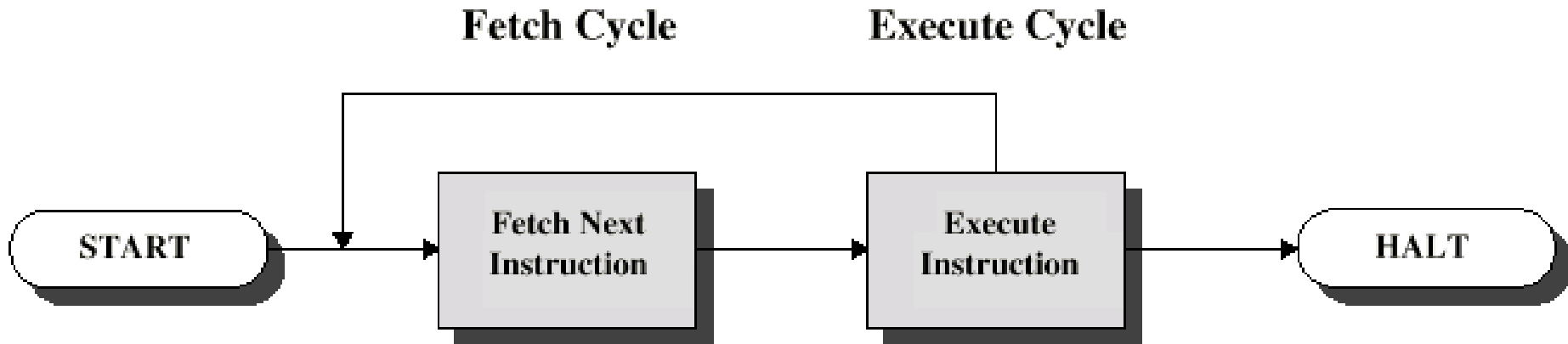
- ⌘ PC : Berisi alamat instruksi yang akan diambil
- ⌘ IR : Berisi instruksi terakhir yang digunakan
- ⌘ MAR : Menandakan alamat dalam memori untuk keperluan write/read berikutnya
- ⌘ MBR : Berisi data yang akan dituliskan ke dalam memori atau menerima data yang dibaca dari memori
- ⌘ I/O AR : menandakan perangkat I/O tertentu
- ⌘ I/O BR : Digunakan untuk pertukaran data antara modul I/O dengan memori

Siklus Instruksi

⌘ Terdiri dari dua langkah :

☑ Fetch (mengambil)

☑ Execute (eksekusi)



Siklus Fetch

- ⌘ Program Counter (PC) menetapkan address instruksi berikutnya untuk fetch
- ⌘ prosesor mengambil instruksi dari lokasi memori ditunjuk oleh PC
- ⌘ Penambahan PC atau pengurangan PC
- ⌘ Instruksi diisikan ke Instruction Register (IR)
- ⌘ Processor menerjemahkan instruksi and menjalankan tindakan yang diinginkan

Siklus Eksekusi

⌘ Processor-memory

- ☑ transfer data antara CPU dan memory utama

⌘ Processor I/O

- ☑ transfer data antara CPU and bagian I/O

⌘ Pengolahan data

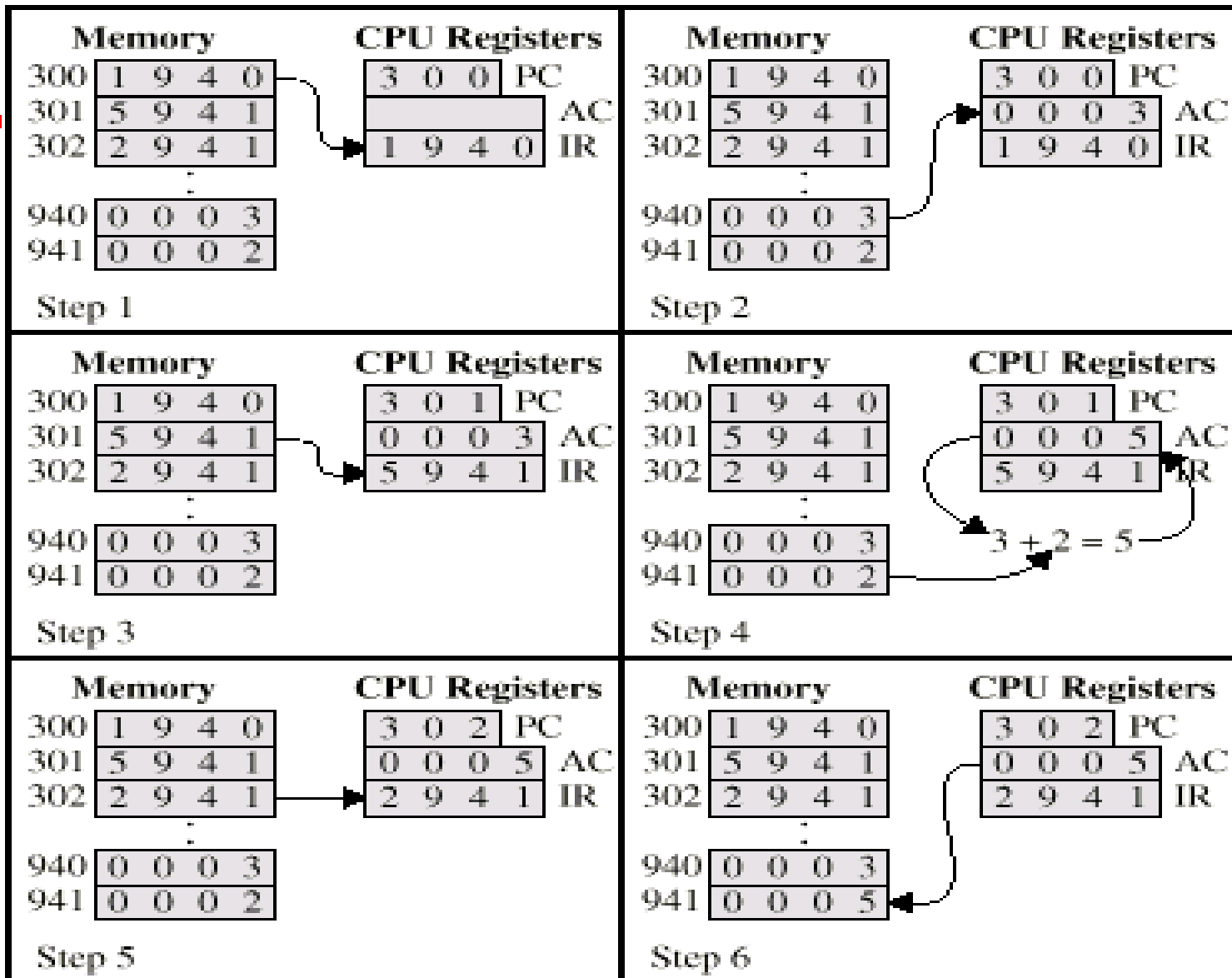
- ☑ Beberapa operasi aritmetik atau logik dari data

⌘ Kontrol

- ☑ Mengubah rangkaian operasi
- ☑ Contoh: jump

⌘ Kombinasi poin-poin diatas

Example of Program Execution



Interrupts/interupsi

- ⌘ Mekanisme kerja oleh modul lain (ex. I/O), interupsi pada rangkaian proses

- ⌘ Program

 - ☒ ex. Pembagian oleh bilangan nol

- ⌘ Timer

 - ☒ Dihasilkan oleh timer prosesor internal

 - ☒ Digunakan untuk mencegah multi-tasking

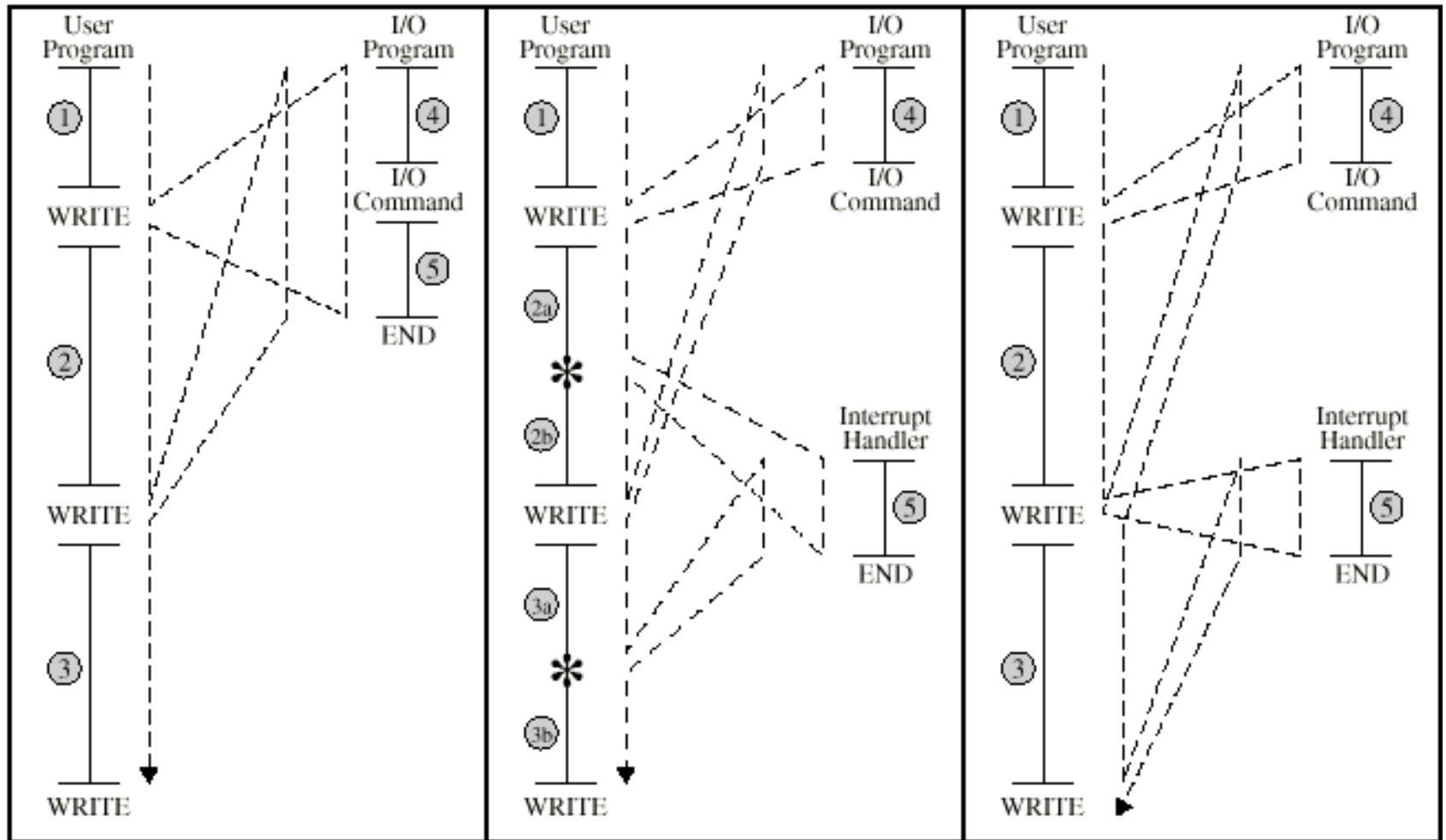
- ⌘ I/O

 - ☒ from I/O controller

- ⌘ Kesalahan Hardware

 - ☒ ex. memory parity error

Program Flow Control



(a) No interrupts

(b) Interrupts; short I/O wait

(c) Interrupts; long I/O wait

Siklus Interupsi

- ⌘ Ditambahkan ke siklus instruksi
- ⌘ Memeriksa prosesor untuk interupsi
 - ☑ Ditandai dengan sinyal interupsi
- ⌘ Jika tidak ada interupsi, ambil instruksi berikutnya
- ⌘ Jika interupsi dibatalkan :
 - ☑ Menunda eksekusi program yang sedang berjalan
 - ☑ Menyimpan konteks
 - ☑ Set PC ke alamat awal penanganan interupsi
 - ☑ Interupsi proses
 - ☑ Menyimpan konteks dan meneruskan program yang terinterupsi

Multiple Interupsi

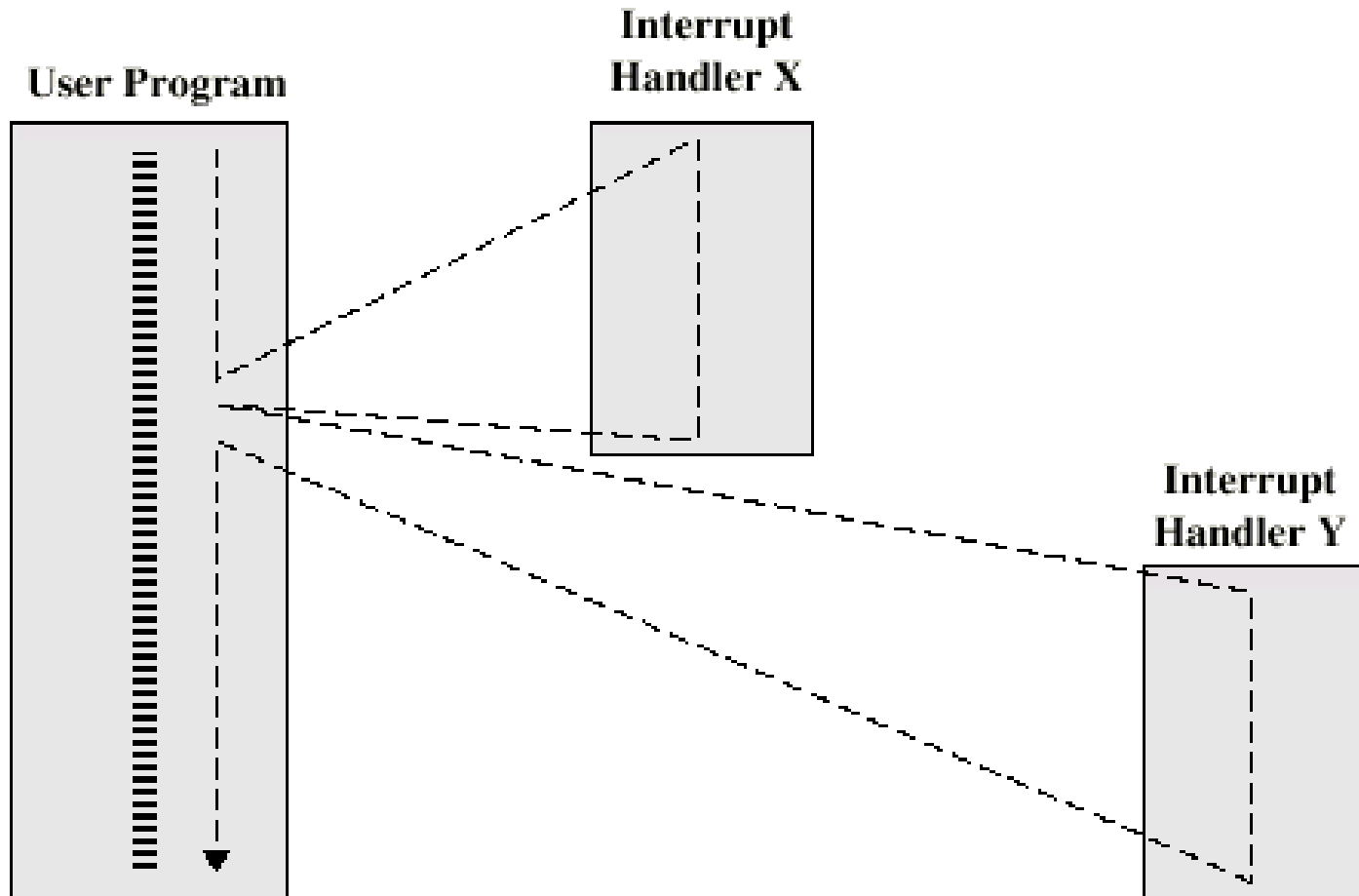
⌘ Interupsi yang dihentikan

- ☑ Prosesor akan mengabaikan interupsi ketika sedang memproses satu interupsi
- ☑ Interupsi akan ditunda dan di cek lagi kemudian jika interupsi pertama selesai diproses
- ☑ Interupsi akan digagalkan jika memang tidak diinginkan

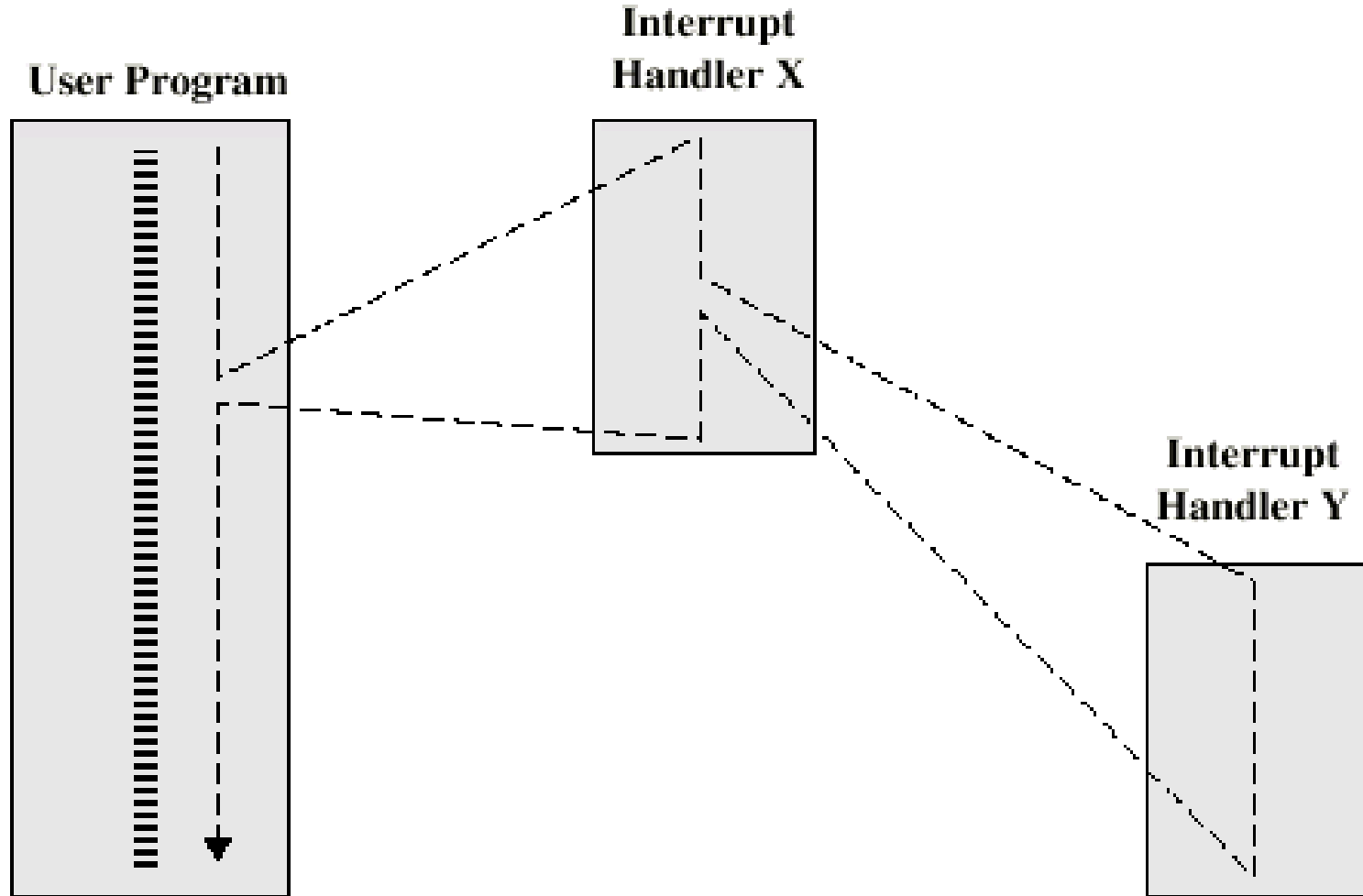
⌘ Diprioritaskan jika :

- ☑ Interupsi dengan prioritas rendah akan diinterupsi oleh priorotas yang lebih tinggi
- ☑ Saat telah menyelesaikan interupsi priorotas tinggi, akan kembali ke interupsi sebelumnya

Multiple Interrupts - Sequential



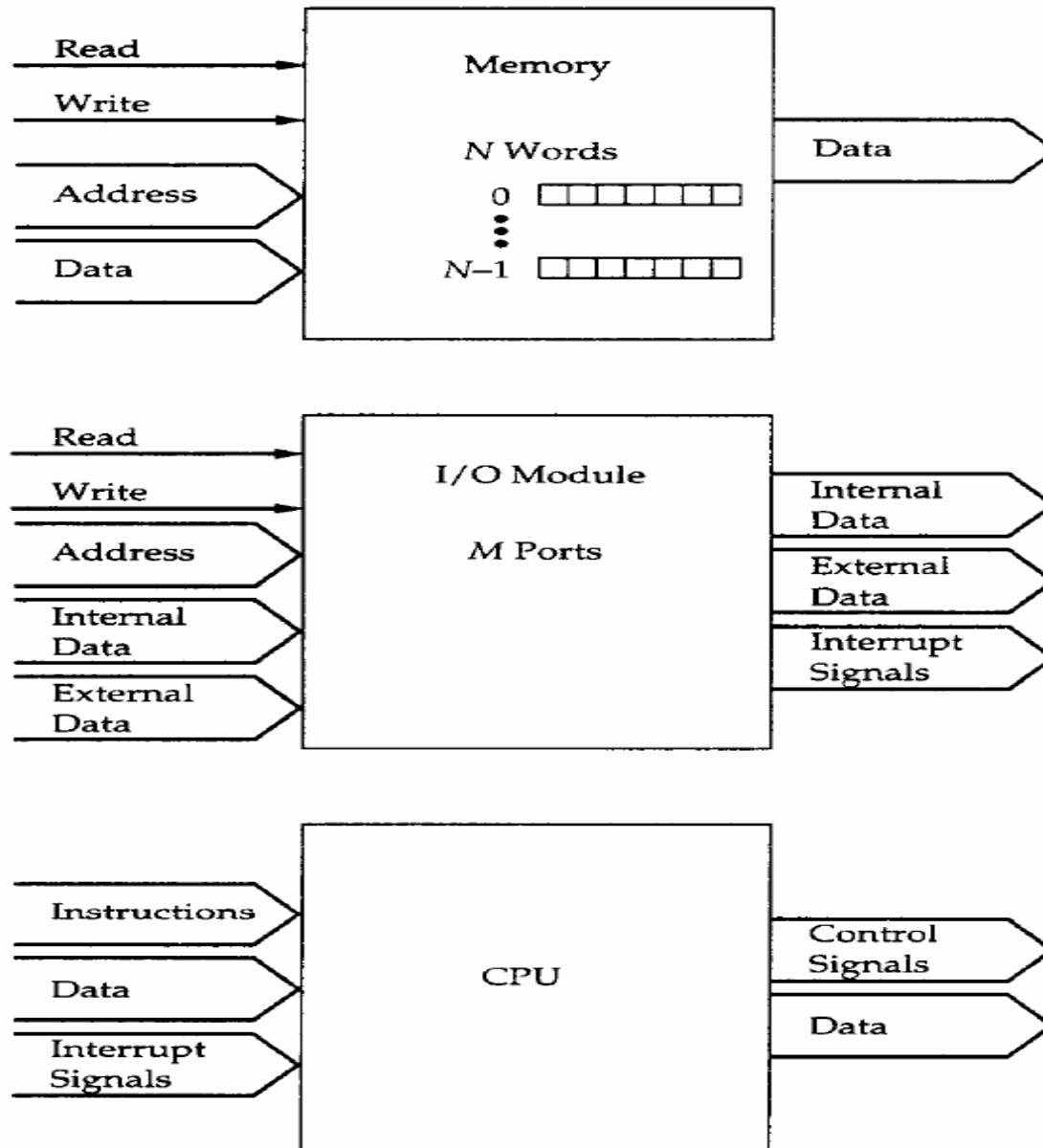
Multiple Interrupts - Nested



Hubungan

- ⌘ Semua unit harus saling berhubungan
- ⌘ Setiap unit akan membutuhkan tipe koneksi yang berbeda
 - ☑ Memory
 - ☑ Input/Output
 - ☑ CPU

Struktur interkoneksi



Hubungan Memory

- ⌘ Menerima dan mengirim data
- ⌘ Menerima alamat-alamat (lokasi)
- ⌘ Menerima sinyal kontrol
 - ⊞ Read/baca
 - ⊞ Write/tulis
 - ⊞ Timing/waktu

Hubungan Input/Output (1)

⌘ Output

- ☑ Menerima data dari komputer
- ☑ Mengirim data ke peripheral

⌘ Input

- ☑ Menerima data dari peripheral
- ☑ Mengirim data ke komputer

Peripheral adalah perangkat yang memberikan unit pengolahan tertentu dengan kemampuan dapat berkomunikasi dengan dunia luar

Hubungan Input/Output (2)

- ⌘ Menerima sinyal kontrol dari komputer
- ⌘ Mengirim sinyal kontrol dari peripherals
 - ☒ ex. putaran disk
- ⌘ Menerima alamat dari komputer
 - ☒ ex. Nomor port untuk mengidentifikasi peripheral
- ⌘ Mengirim sinyal interupsi (kontrol)

Hubungan CPU

- ⌘ Membaca instruksi dan data
- ⌘ Menulis data (sesudah proses)
- ⌘ Mengirim sinyal kontrol ke unit lain
- ⌘ Menerima (dan melakukan) interupsi

Bus-bus

- ⌘ Sejumlah bilangan dari interkoneksi sistem yang mungkin
- ⌘ Struktur BUS yang Single dan multiple kebanyakan bersatu
- ⌘ e.g. Control/Address/Data bus (PC)
- ⌘ e.g. Unibus (DEC-PDP)

Apa itu Bus?

- ⌘ Jalur komunikasi untuk menghubungkan dua atau lebih perangkat
- ⌘ Biasanya digunakan pada broadcast
- ⌘ Kadang-kadang berkelompok
 - ☒ Jumlah channel pada satu bus
 - ☒ ex. 32 bit data bus adalah 32 single bit channels

Bus Data

⌘ Carries data (data)

☑ Ingat, pada level ini tidak ada perbedaan antara "data" dengan "instruksi"

⌘ Lebar bit adalah kunci penentu tampilan

☑ 8, 16, 32, 64 bit

Address bus (alamat bus)

- ⌘ Mengidentifikasi sumber dan tujuan data
 - ☑ Ex. CPU perlu membaca instruksi (data) dari sebuah lokasi di memori
- ⌘ Lebar bus disesuaikan dengan kapasitas maks memori sistem
 - ☑ ex. 8080 mempunyai 16 bit address bus memberikan 64k ruang address

Control Bus (bus kontrol)

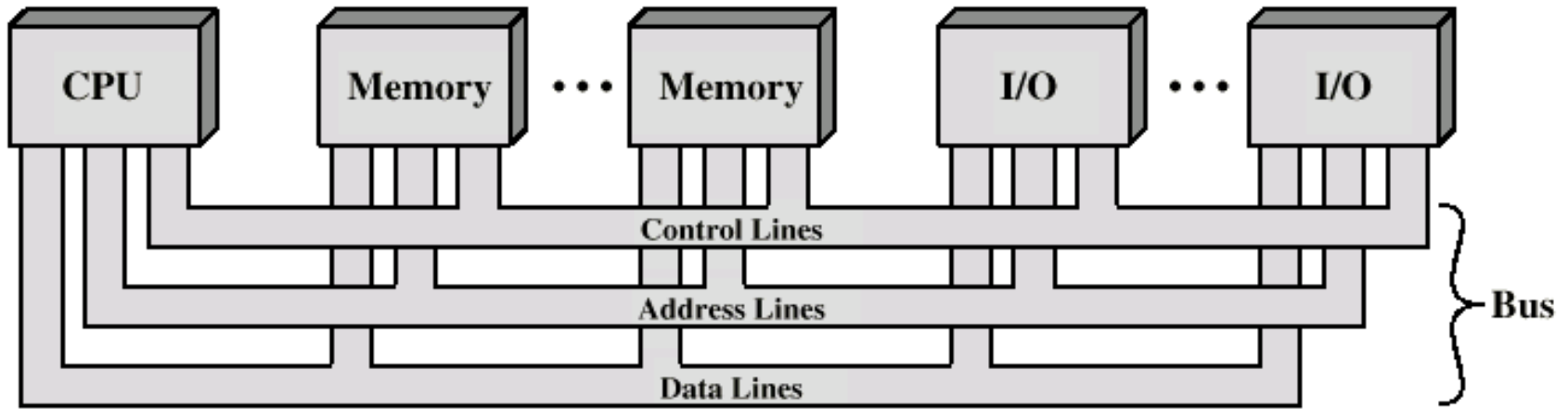
⌘ Informasi kontrol and timing

- ☑ Memori sinyal read/write

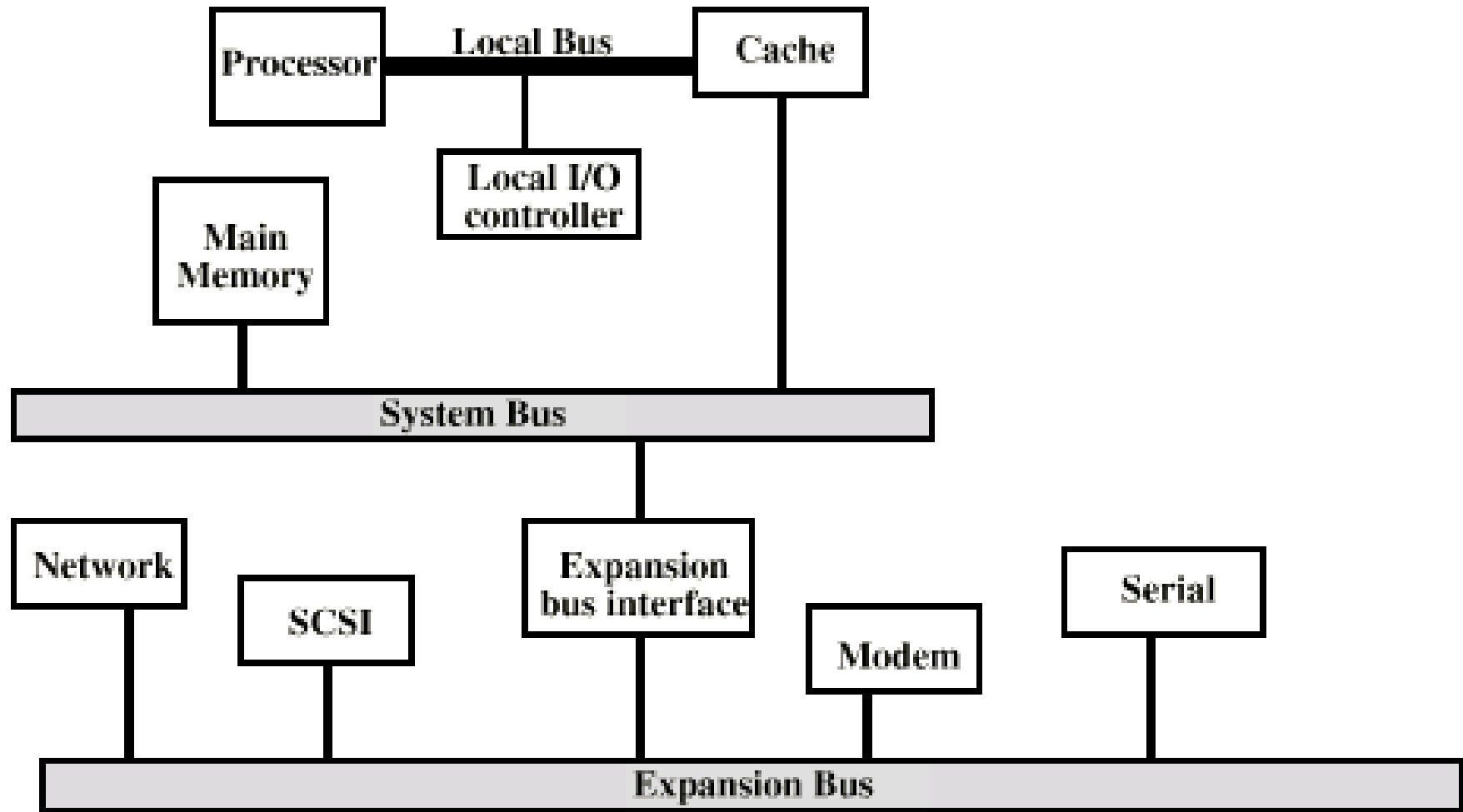
- ☑ Permintaan interupsi

- ☑ Sinyal clock

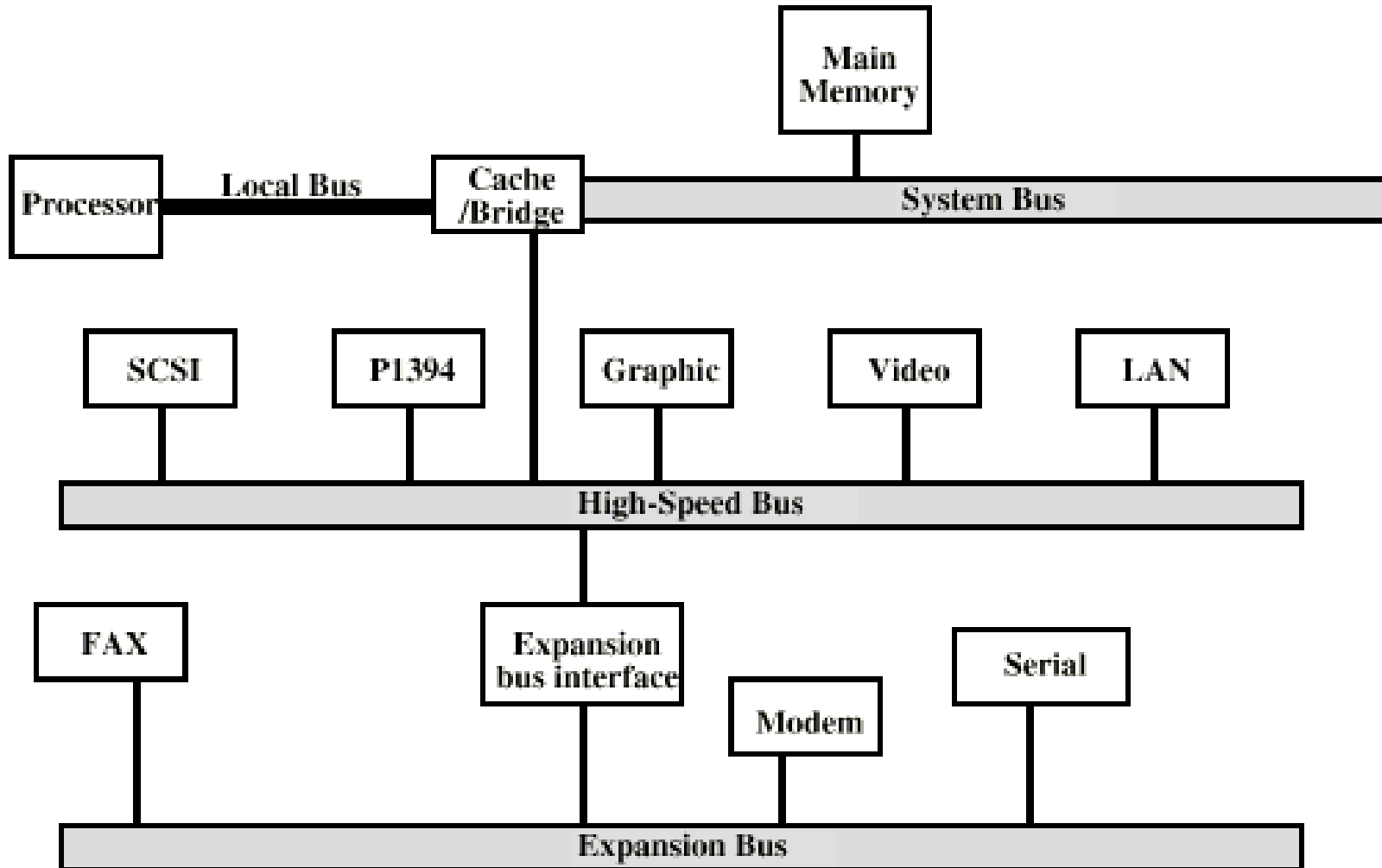
Skema Interkoneksi Bus



Traditional (ISA) (with cache)



High Performance Bus



Tipe Bus

⌘ Single

- ☑ Memisahkan data dan jalur alamat

⌘ Multiplexed

- ☑ Jalur dipakai bersama
- ☑ Jalur kontrol, alamat yg benar atau data yg benar
- ☑ Keuntungan – jalur lebih sedikit
- ☑ Kekurangan
 - ☒ Kontrol lebih rumit
 - ☒ Kemampuannya terbatas

Bus Arbitration (pemisahan bus)

- ⌘ Jika terdapat lebih dari satu kontrol modul bus
- ⌘ Ex. CPU dan DMA controller
- ⌘ Hanya satu modul yang boleh mengontrol bus dalam satu waktu
- ⌘ Dua metoda pemisahan :
 - Dipusatkan : central bus controller menjadi media semua peralatan
 - Didistribusikan : semua peralatan bisa akses kontrol ke bus

Pemisahan yang didistribusikan

- ⌘ Masing-masing modul boleh mengklaim bus
- ⌘ Control logic pada semua modul

Bus Timing

⌘ Synchronous

- ☒ Event ditentukan oleh sinyal clock
- ☒ Control Bus termasuk didalamnya clock line
- ☒ Semua peralatan bisa membaca clock line
- ☒ sync pada leading edge
- ☒ Satu siklus untuk satu event
- ☒ Ex. PCI bus

⌘ Asynchronous

- ☒ Event yang terjadi mengikuti dan tergantung dari event sebelumnya
- ☒ Lebih fleksible tapi lebih rumit
- ☒ Ex. Futurebus+

Synchronous Timing Diagram

Clock



Start



Read



Address
Lines



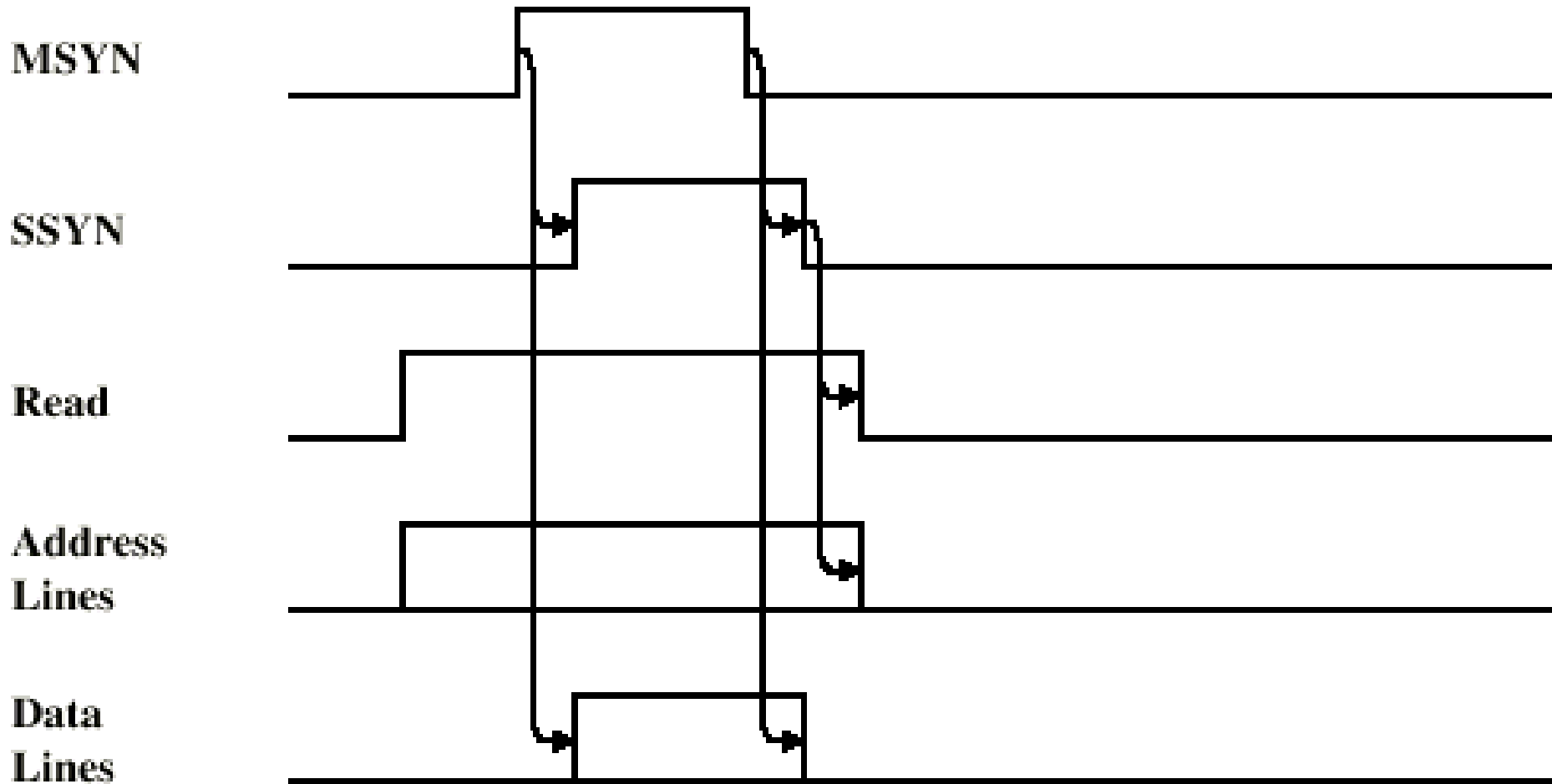
Data
Lines



Acknowledge



Asynchronous Timing Diagram



Foreground Reading

⌘ Stallings, chapter 3 (all of it)

⌘ www.pcguides.com/ref/mbsys/buses/

⌘ In fact, read the whole site!

⌘ www.pcguides.com/

⌘ thread