



PROGRAM STUDI

**S1 SISTEM KOMPUTER**

**UNIVERSITAS DIPONEGORO**

# Bus Sistem

Oky Dwi Nurhayati, ST, MT  
email: [okydnd@undip.ac.id](mailto:okydnd@undip.ac.id)

# KONSEP PROGRAM

- ◉ Pemrograman (hardware) merupakan proses penghu-bungan berbagai komponen logik pada konfigurasi yang diinginkan untuk membentuk operasi aritmatik dan logik pada data tertentu
- ◉ *Hardwired program* tidak flexibel
- ◉ *General purpose hardware* dapat mengerjakan berbagai macam tugas tergantung sinyal kendali yang diberikan
- ◉ Daripada melakukan *re-wiring*, Lebih baik menambah-kan sinyal-sinyal kendali yang baru

# PROGRAM ?

- ⊙ Adalah suatu deretan langkah-langkah
- ⊙ Pada setiap langkah, dikerjakan suatu operasi arithmetic atau logical
- ⊙ Pada setiap operasi, diperlukan sejumlah sinyal kendali tertentu

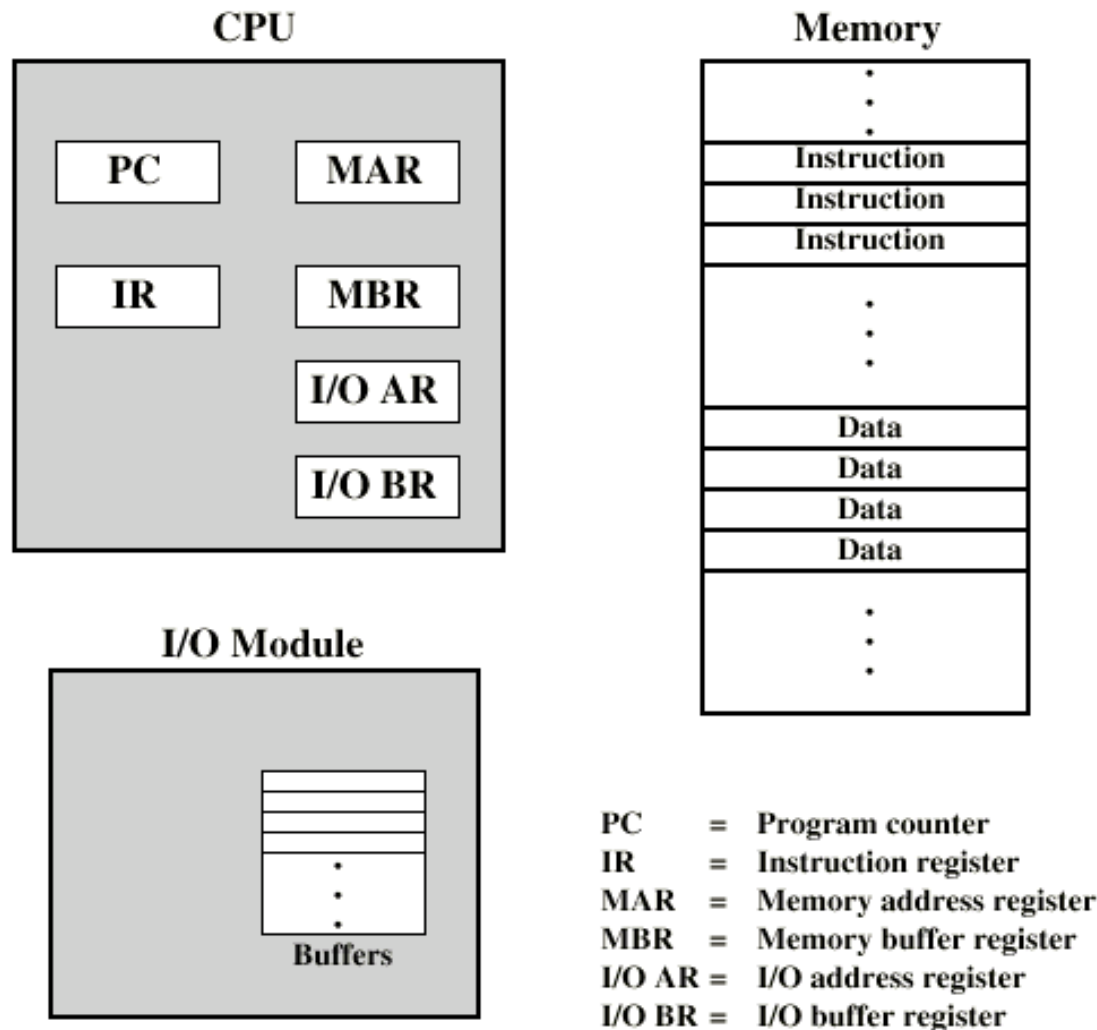
# FUNGSI CONTROL UNIT

- ⦿ Untuk setiap operasi disediakan kode yang unik
  - Contoh: ADD, MOVE
- ⦿ Bagian hardware tertentu menerima kode tersebut kemudian menghasilkan sinyal-sinyal kendali
- ⦿ Jadilah komputer!

# KOMPONEN YANG DIPERLUKAN

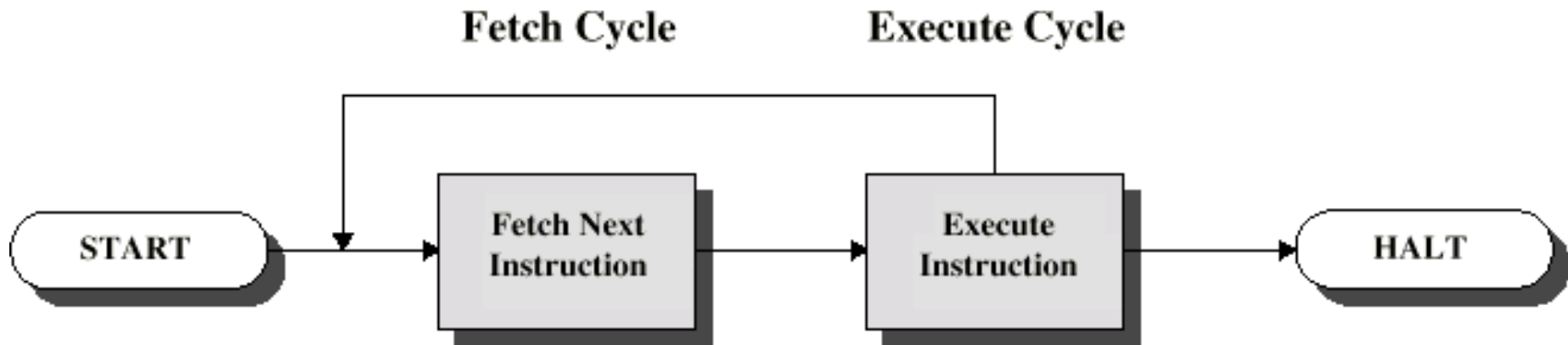
- ◉ Control Unit (CU) dan Arithmetic and Logic Unit (ALU) membentuk Central Processing Unit (CPU)
- ◉ Data dan instruksi harus diberikan ke sistem dan dikeluarkan dari sistem
  - Input/output
- ◉ Diperlukan tempat untuk menyimpan sementara kode instruksi dan hasil operasi.
  - Main memory

# KOMPONEN KOMPUTER: TOP LEVEL VIEW



# SIKLUS INSTRUKSI

- Two steps:
  - Fetch
  - Execute



# FETCH CYCLE

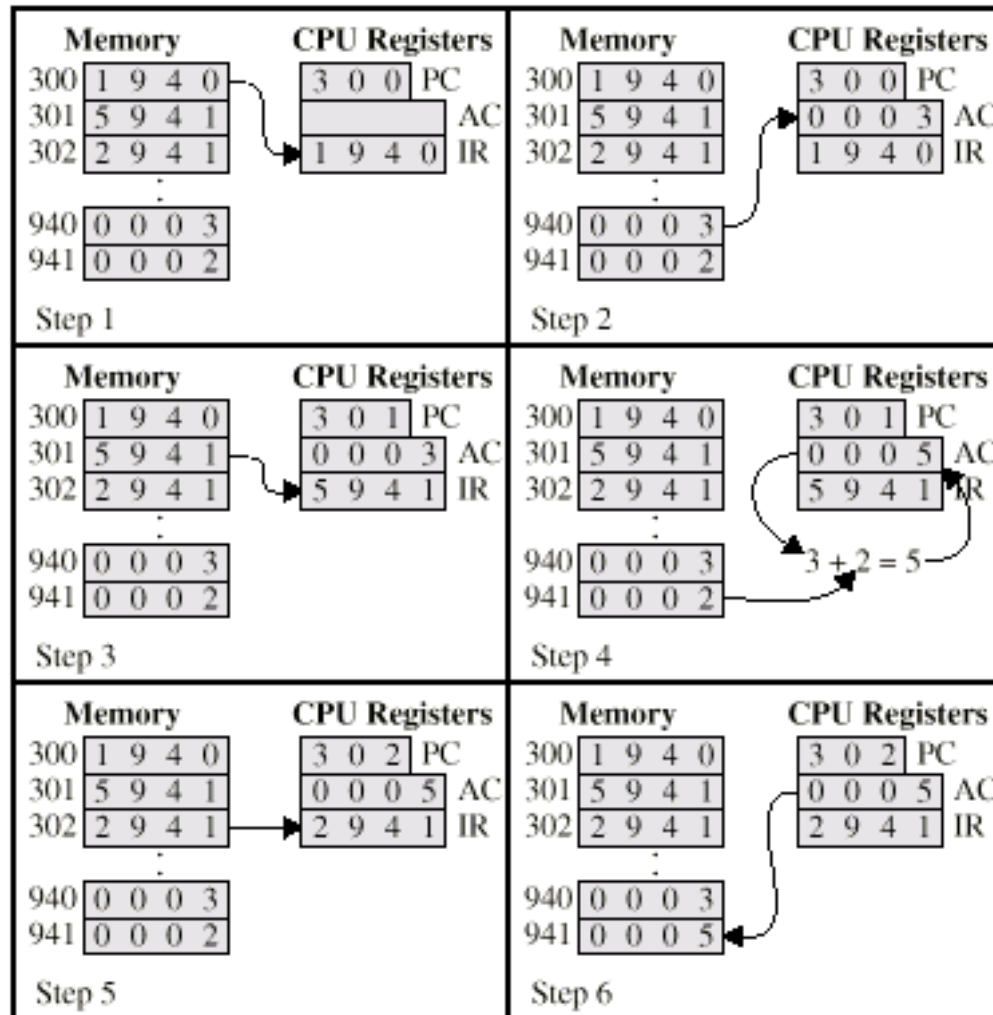
- ◉ Program Counter (PC) berisi address instruksi berikutnya yang akan diambil
- ◉ Processor mengambil instruksi dari memory pada lokasi yang ditunjuk oleh PC
- ◉ Naikkan PC
  - Kecuali ada perintah tertentu
- ◉ Instruksi dimasukkan ke Instruction Register (IR)
- ◉ Processor meng-interpret dan melakukan tindakan yang diperlukan



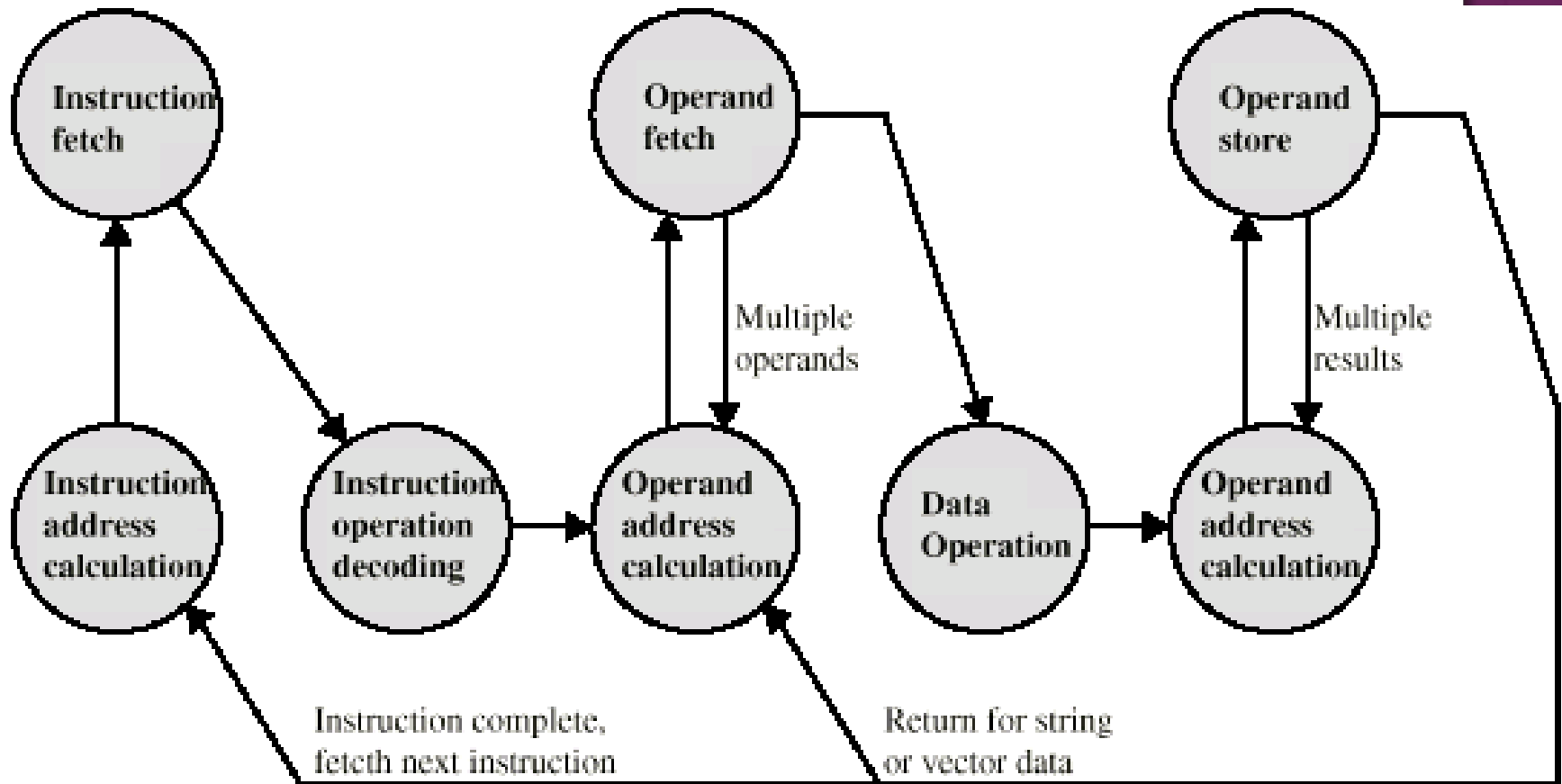
# EXECUTE CYCLE

- ◉ Processor-memory
  - Transfer data antara CPU dengan main memory
- ◉ Processor I/O
  - Transfer data antara CPU dengan I/O module
- ◉ Data processing
  - Operasi arithmetic dan logical pada data tertentu
- ◉ Control
  - Mengubah urutan operasi
  - Contoh: jump
- ◉ Kombinasi diatas

# CONTOH EKSEKUSI PROGRAM



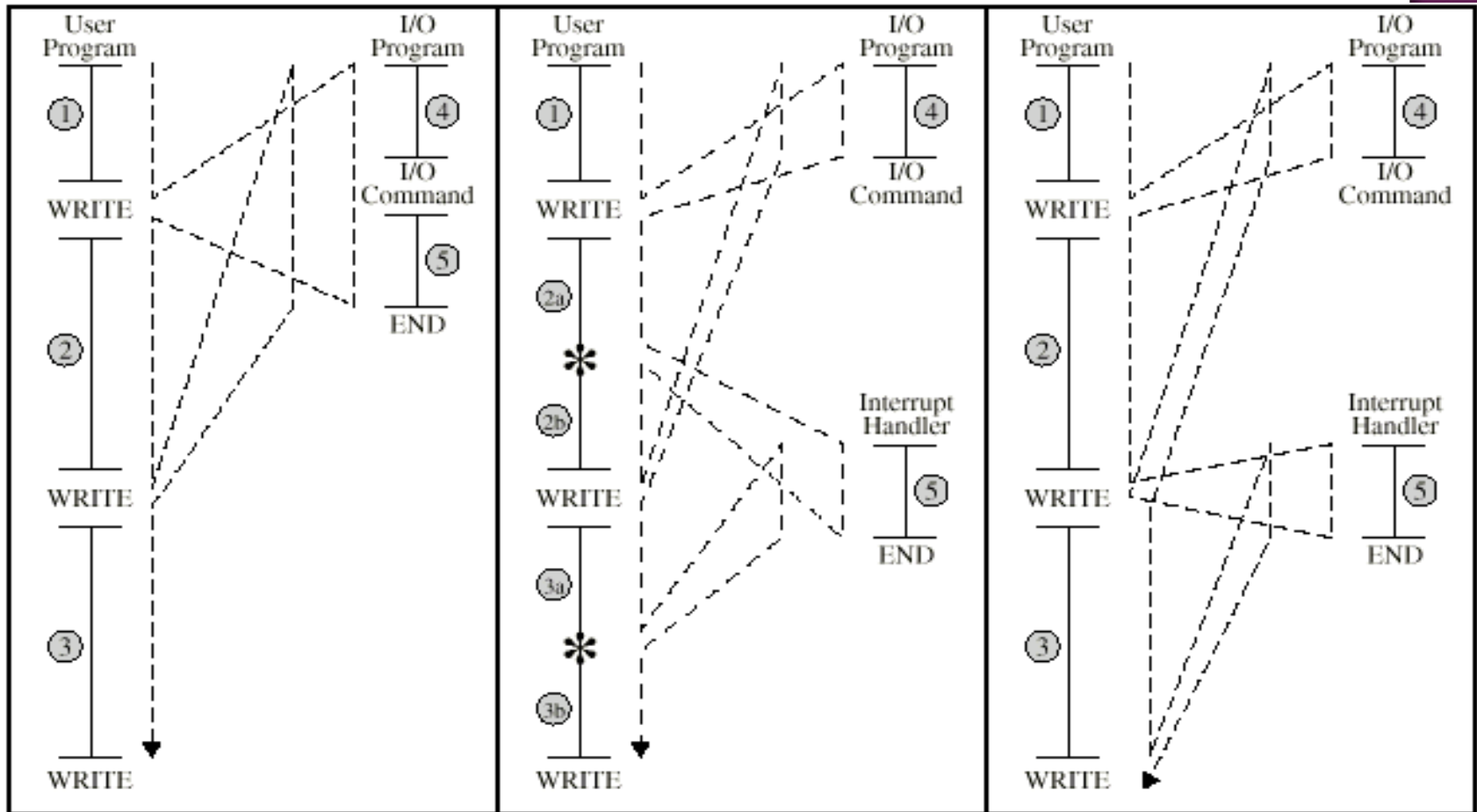
# DIAGRAM KEADAAN SIKLUS INSTRUKSI



# INTERRUPT

- ⊙ Suatu mekanisme yang disediakan bagi modul-modul lain (mis. I/O) untuk dapat meng-interrupt operasi normal CPU
- ⊙ Program
  - Misal: overflow, division by zero
- ⊙ Timer
  - Dihasilkan oleh internal processor timer
  - Digunakan dalam pre-emptive multi-tasking
- ⊙ I/O
  - dari I/O controller
- ⊙ Hardware failure
  - Misal: memory parity error

# PROGRAM FLOW CONTROL



(a) No interrupts

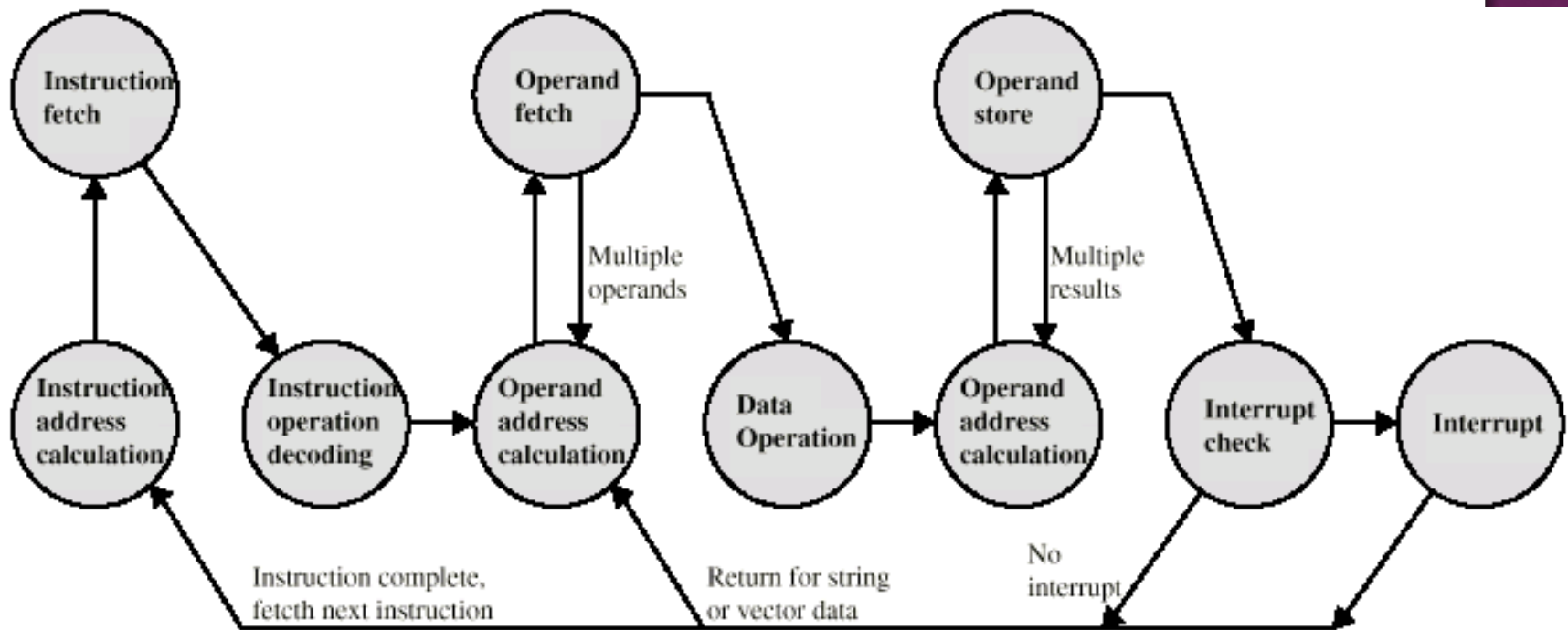
(b) Interrupts; short I/O wait

(c) Interrupts; long I/O wait

# SIKLUS INTERUPSI

- ◉ Ditambahkan ke instruction cycle
- ◉ Processor memeriksa adanya interrupt
  - Diberitahukan lewat interrupt signal
- ◉ Jika tidak ada interrupt, fetch next instruction
- ◉ Jika ada interrupt:
  - Tunda eksekusi dari program saat itu
  - Simpan *context*
  - Set PC ke awal address dari routine interrupt handler
  - Proses interrupt
  - Kembalikan *context* dan lanjutkan program yang terhenti.

# DIAGRAM KEADAAN SIKLUS INSTRUKSI DENGAN INTERRUPT



# MULTIPLE INTERRUPTS

## ⦿ Disable interrupts

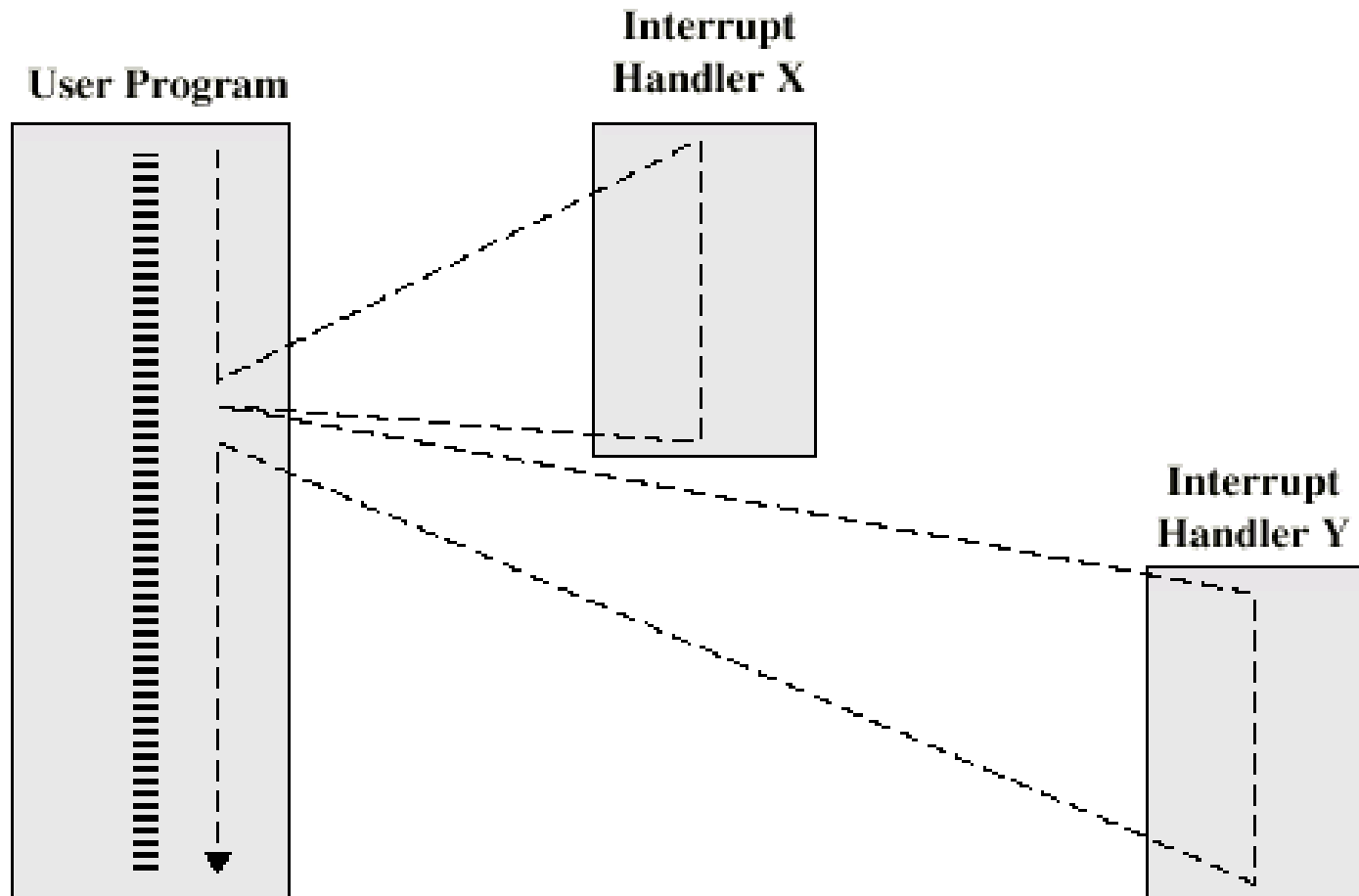
- Processor akan mengabaikan interrupt berikutnya
- Interrupts tetap akan diperiksa setelah interrupt yang pertama selesai dilayani
- Interrupts ditangani dalam urutan sesuai datangnya

## ⦿ Define priorities

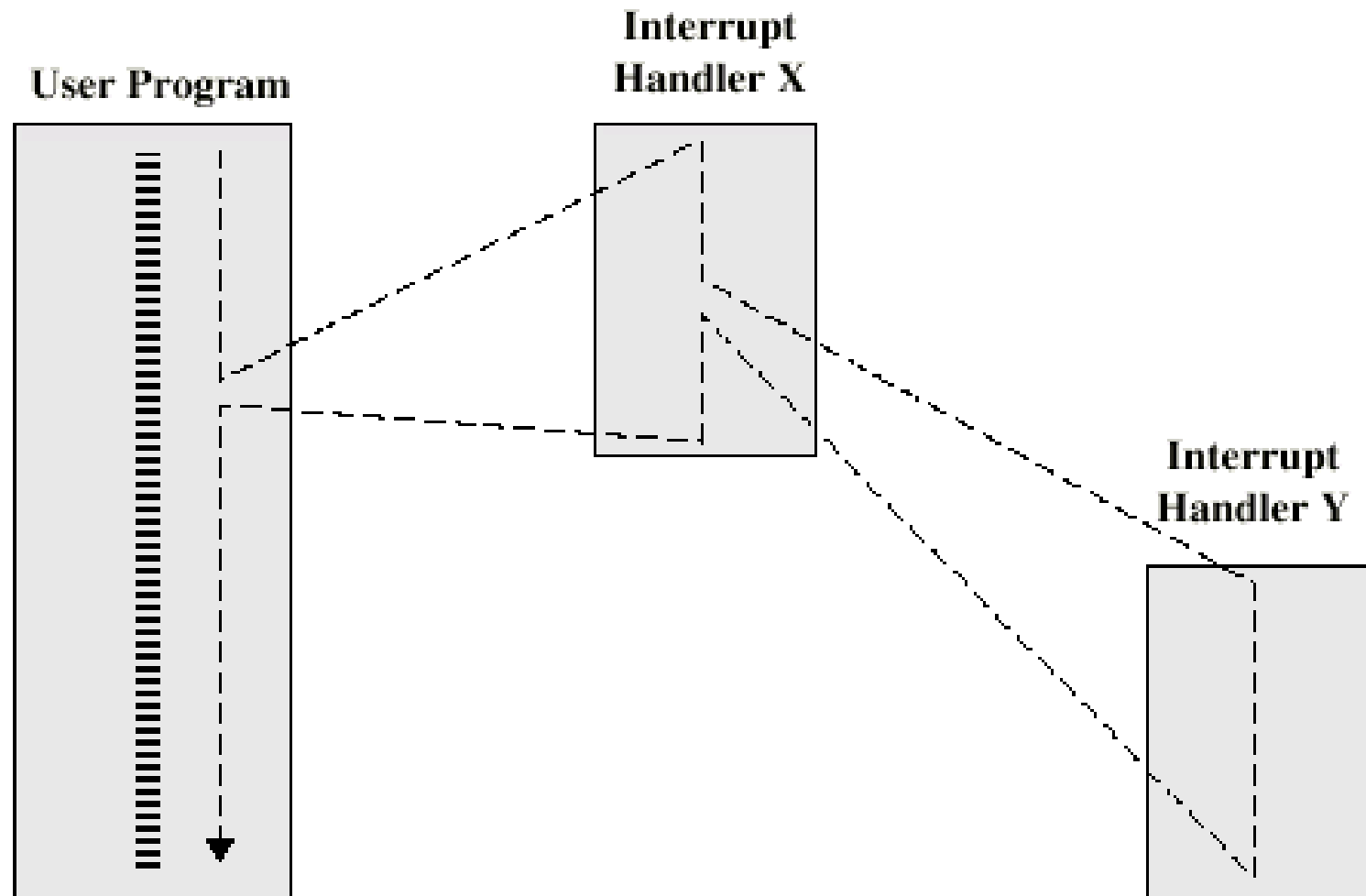
- Low priority interrupts dapat di interrupt oleh higher priority interrupts
- Setelah higher priority interrupt selesai dilayani, akan kembali ke interrupt sebelumnya.



# MULTIPLE INTERRUPTS - SEQUENTIAL



# MULTIPLE INTERRUPTS - NESTED



# SAMBUNGAN

- ◉ Semua unit harus tersambung
- ◉ Unit yang berbeda memiliki sambungan yang berbeda
  - Memory
  - Input/Output
  - CPU

# SAMBUNGAN MEMORI

- ⦿ Menerima dan mengirim data
- ⦿ Menerima addresses
- ⦿ Menerima sinyal kendali
  - Read
  - Write
  - Timing

# SAMBUNGAN INPUT/OUTPUT

- ⦿ Serupa dengan sambungan memori
- ⦿ Output
  - Menerima data dari computer
  - Mengirimkan data ke peripheral
- ⦿ Input
  - Menerima data dari peripheral
  - Mengirimkan data ke computer

# SAMBUNGAN INPUT/OUTPUT

- ⊙ Menerima sinyal kendali dari computer
- ⊙ Mengirimkan sinyal kendali ke peripherals
  - Contoh: spin disk
- ⊙ Menerima address dari computer
  - Contoh: nomor port
- ⊙ Mengirimkan sinyal interrupt

# CPU CONNECTION

- ⦿ Membaca instruksi dan data
- ⦿ Menuliskan data (setelah diproses)
- ⦿ Mengirimkan sinyal kendali ke unit-unit lain
- ⦿ Menerima (& menanggapi) interrupt

# BUS

- ◉ Ada beberapa kemungkinan interkoneksi sistem
- ◉ Yang biasa dipakai: Single Bus dan multiple BUS
- ◉ PC: Control/Address/Data bus
- ◉ DEC-PDP: Unibus



# WHAT IS A BUS?

- ⦿ Jalur komunikasi yang menghubungkan beberapa device
- ⦿ Biasanya menggunakan cara broadcast
- ⦿ Seringkali dikelompokkan
  - Satu bus berisi sejumlah kanal (jalur)
  - Contoh bus data 32-bit berisi 32 jalur
- ⦿ Jalur sumber tegangan biasanya tidak diperlihatkan

# DATA BUS

- ⦿ Membawa data
  - Tidak dibedakan antara “data” dan “instruksi”
- ⦿ Lebar jalur menentukan performance
  - 8, 16, 32, 64 bit

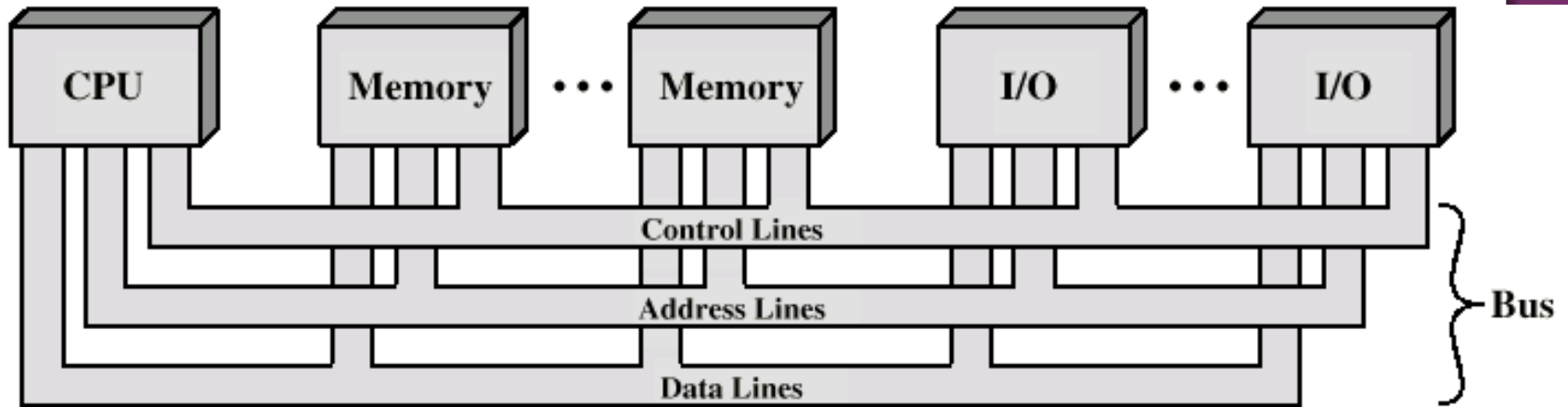
# ADDRESS BUS

- ⦿ Menentukan asal atau tujuan dari data
- ⦿ Misalkan CPU perlu membaca instruksi (data) dari memori pada lokasi tertentu
- ⦿ Lebar jalur menentukan kapasitas memori maksimum dari sistem
  - Contoh 8080 memiliki 16 bit address bus maka ruang memori maksimum adalah 64k

# CONTROL BUS

- ◉ Informasi kendali dan timing
  - Sinyal read/write memory (MRD/MWR)
  - Interrupt request (IRQ)
  - Clock signals (CK)

# SKEMA INTERKONEKSI BUS



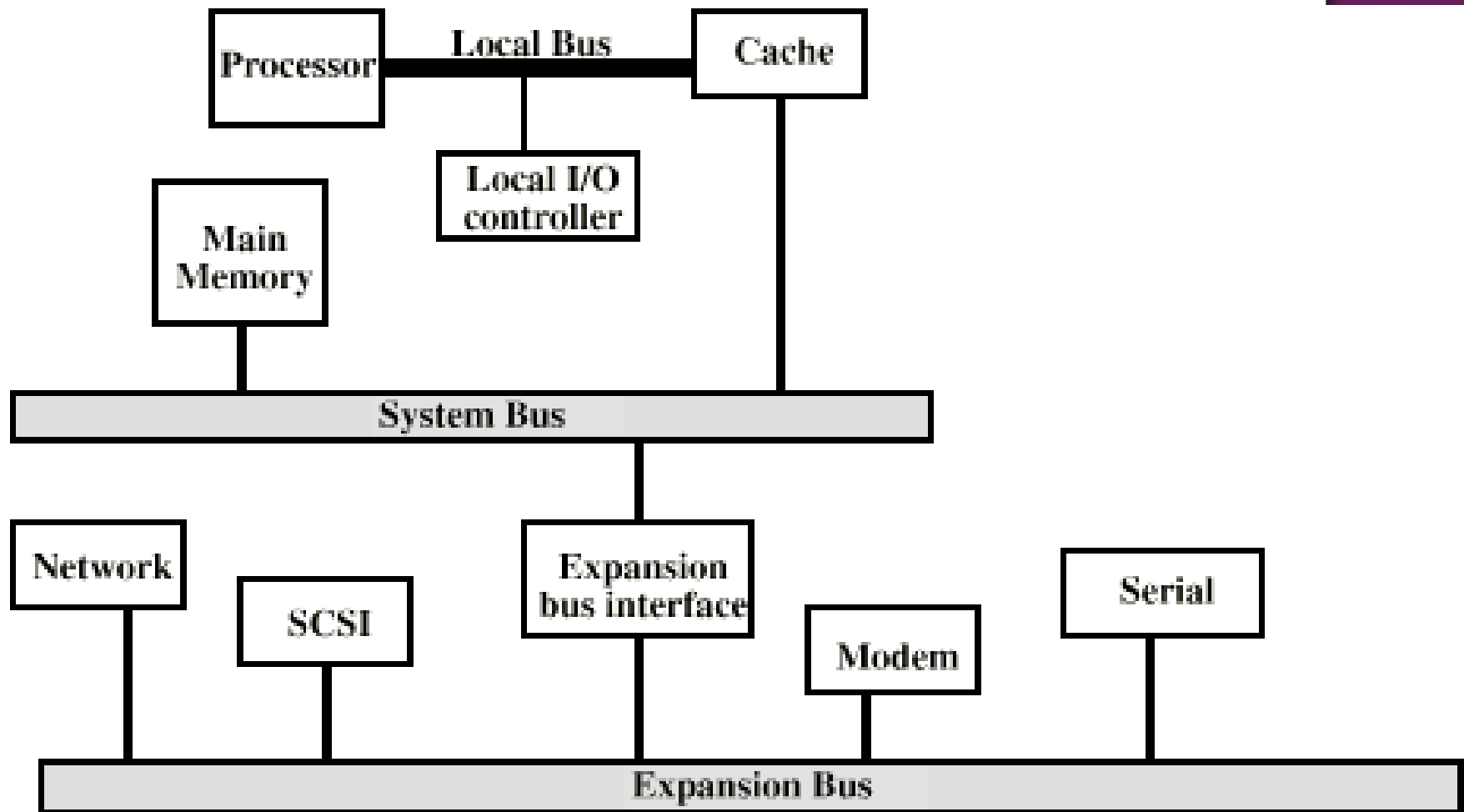
# BENTUK FISIK

- ◉ Bagaimana bentuk fisik bus?
  - Jalur-jalur parallel PCB
  - Ribbon cables
  - Strip connectors pada mother boards
    - contoh PCI
  - Kumpulan kabel

# PROBLEM PADA SINGLE BUS

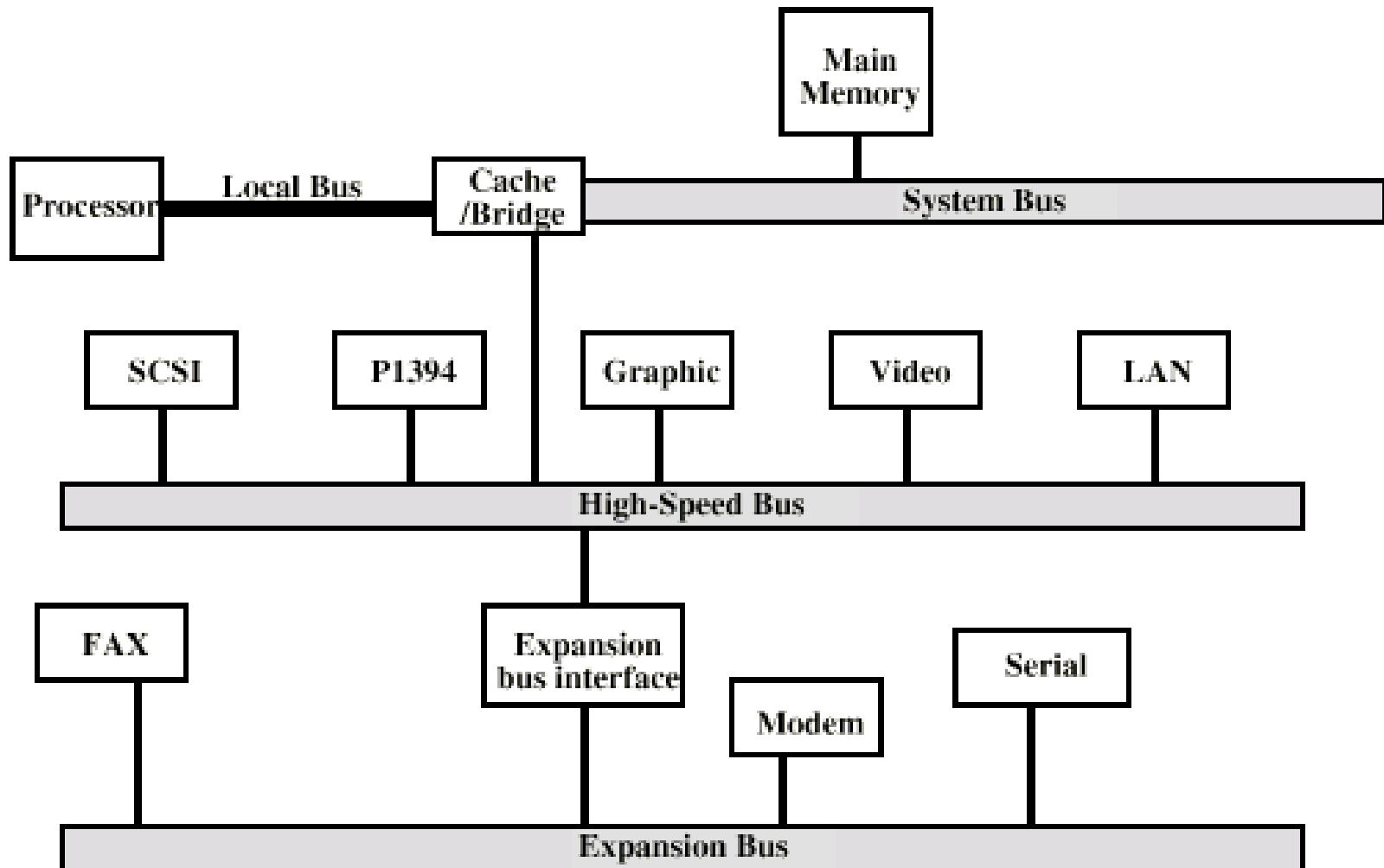
- ⊙ Banyak devices pada bus tunggal menyebabkan:
  - Propagation delays
    - Jalur data yg panjang berarti memerlukan koordinasi pemakaian shg berpengaruh pada performance
    - If aggregate data transfer approaches bus capacity
- ⊙ Kebanyakan sistem menggunakan multiple bus

# BUS TRADITIONAL (ISA) (MENGGUNAKAN CACHE)





# HIGH PERFORMANCE BUS



# JENIS BUS

## ◉ Dedicated

- Jalur data & address terpisah

## ◉ Multiplexed

- Jalur bersama
- Address dan data pada saat yg beda
- Keuntungan - jalur sedikit
- Kerugian
  - Kendali lebih kompleks
  - Mempengaruhi performance

# ARBITRASI BUS

- ⦿ Beberapa modul mengendalikan bus
- ⦿ contoh CPU dan DMA controller
- ⦿ Setiap saat hanya satu modul yg mengendalikan
- ⦿ Arbitrasi bisa secara centralised atau distributed

# ARBITRASI CENTRALISED

- ⦿ Ada satu hardware device yg mengendalikan akses bus
  - Bus Controller
  - Arbitrer
- ⦿ Bisa berupa bagian dari CPU atau terpisah

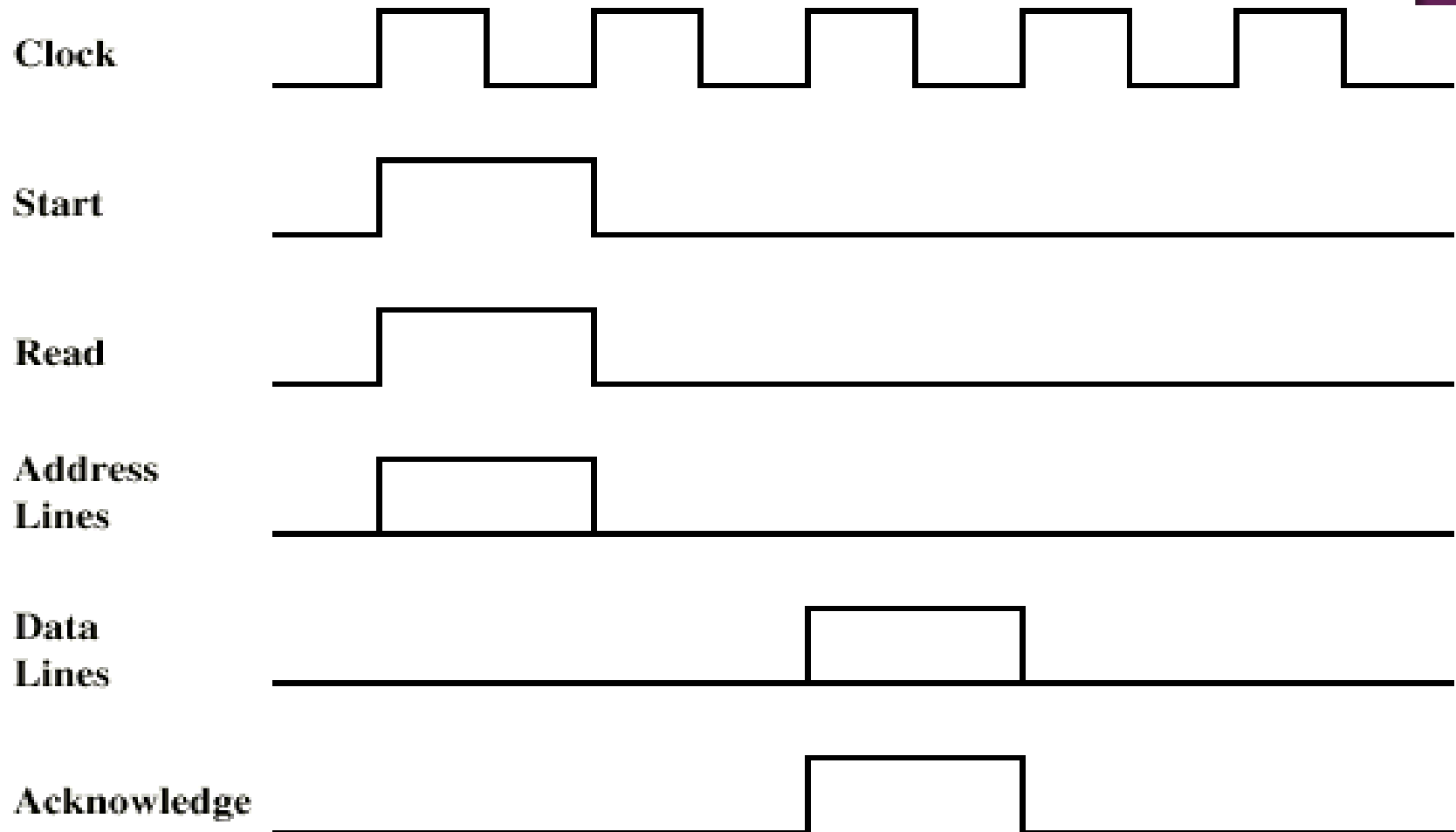
# ARBITRASI DISTRIBUTED

- ⦿ Setiap module dapat meng-klaim bus
- ⦿ Setiap modules memiliki Control logic

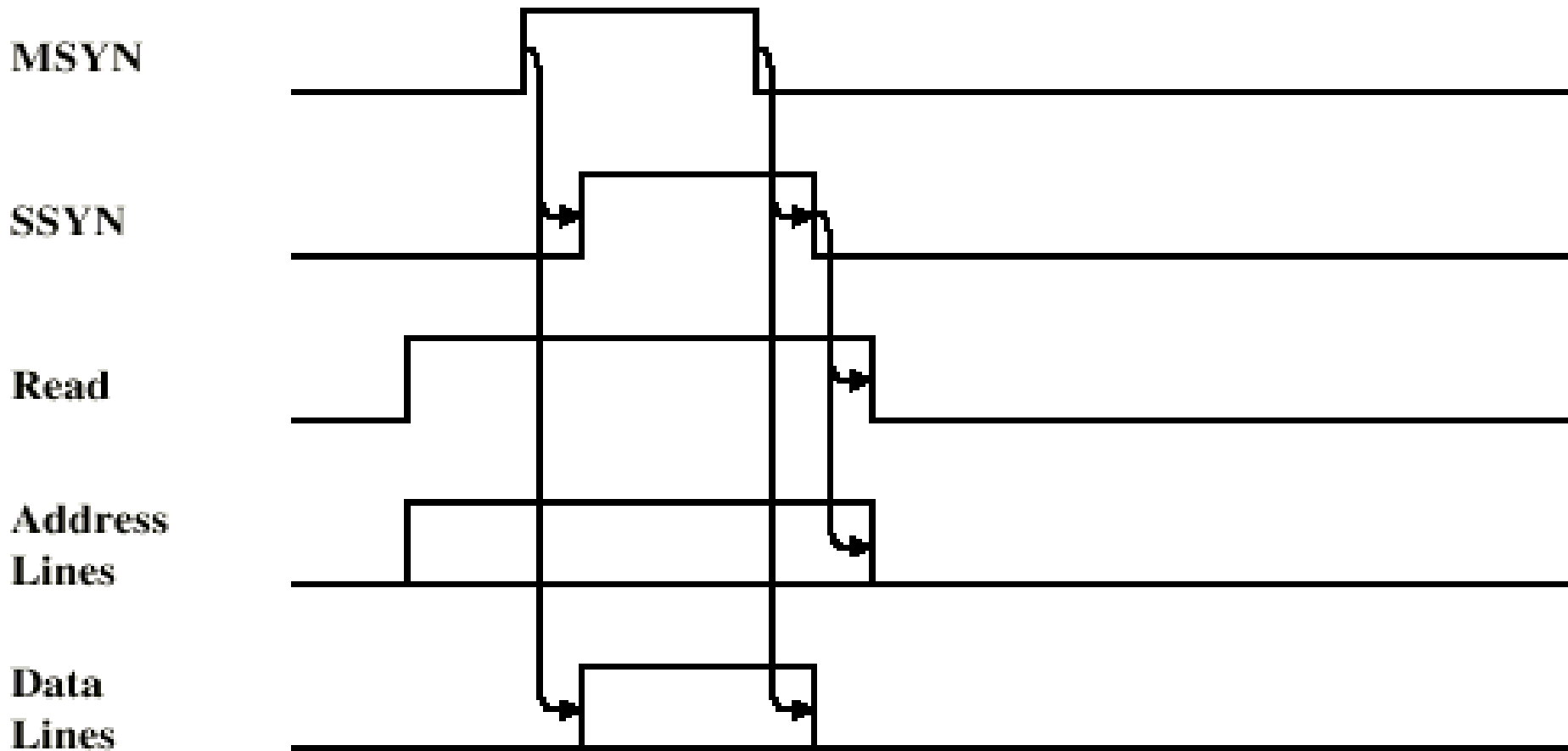
# TIMING

- ⊙ Koordinasi event pada bus
- ⊙ Synchronous
  - Event ditentukan oleh sinyal clock
  - Control Bus termasuk jalur clock
  - Siklus bus ( bus cycle) transmisi 1 ke 0
  - Semua devices dpt membaca jalur clock
  - Biasanya sinkronisasi terjadi pada tepi naik (leading edge)
  - Suatu event biasanya dimulai pada awal siklus

# SYNCHRONOUS TIMING DIAGRAM



# ASYNCHRONOUS TIMING DIAGRAM





# BUS PCI

- ⦿ Peripheral Component Interconnection
- ⦿ Dikeluarkan oleh Intel sebagai public domain
- ⦿ 32 atau 64 bit
- ⦿ 50 Jalur

# JALUR PADA BUS PCI (YANG HARUS)

- ◉ Jalur System
  - clock and reset
- ◉ Address & Data
  - 32 jalur multiplex address/data
  - Jalur validasi
- ◉ Interface Control
- ◉ Arbitrasi
  - Not shared
  - Direct connection to PCI bus arbiter
- ◉ Error lines

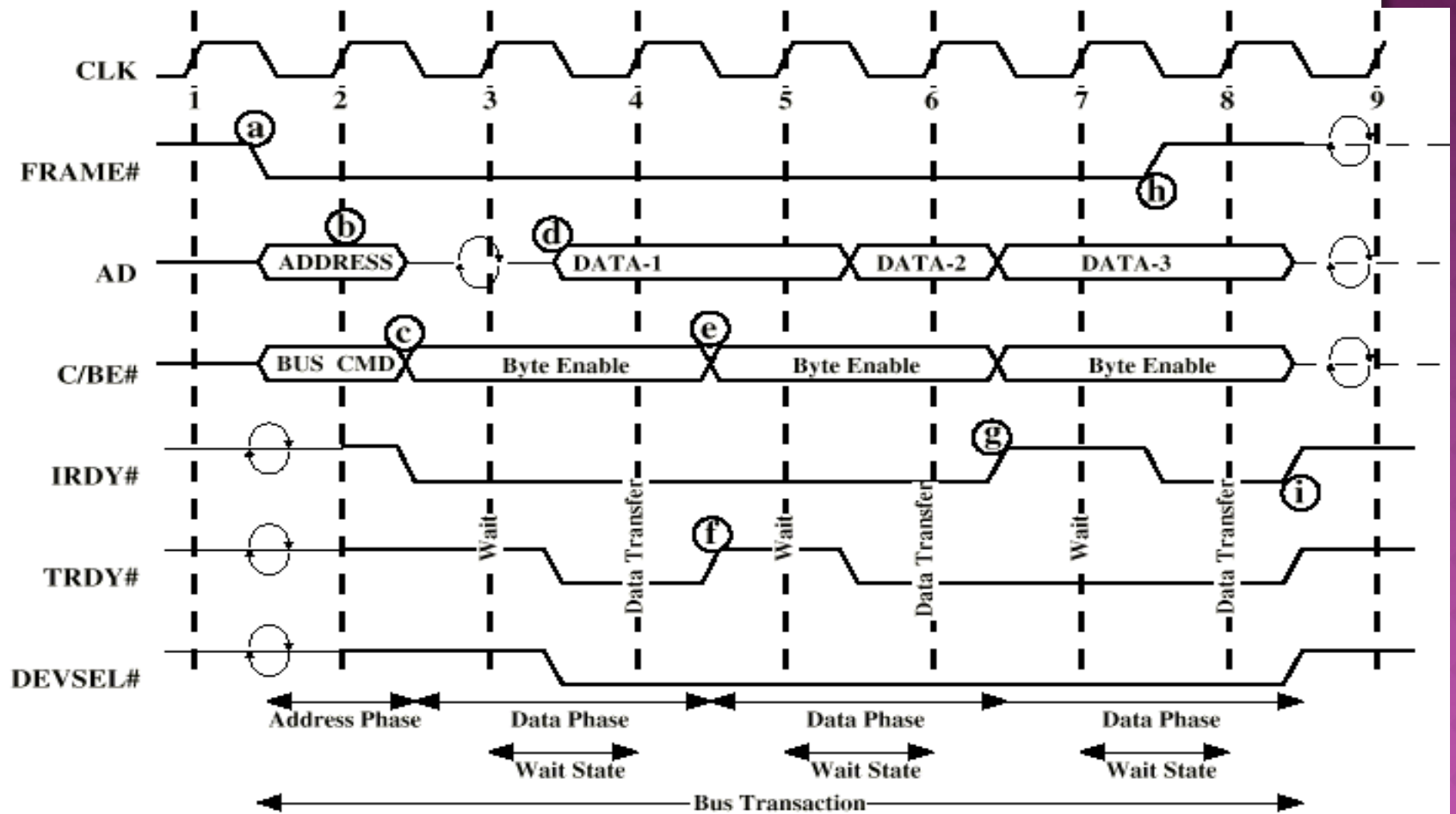
# JALUR BUS PCI (OPTIONAL)

- ⦿ Interrupt lines
  - Not shared
- ⦿ Cache support
- ⦿ 64-bit Bus Extension
  - Additional 32 lines
  - Time multiplexed
  - 2 lines to enable devices to agree to use 64-bit transfer
- ⦿ JTAG/Boundary Scan
  - For testing procedures

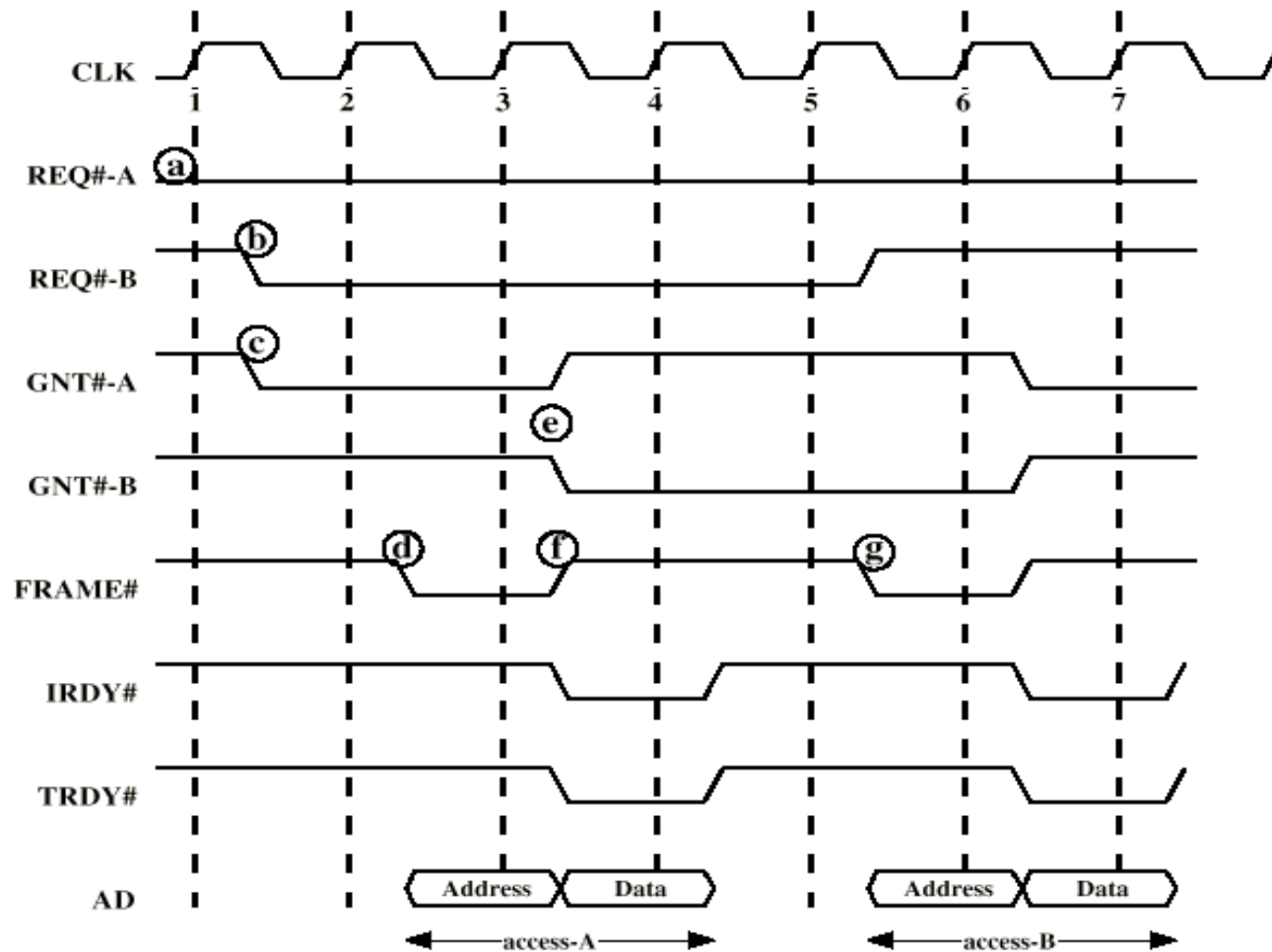
# COMMAND PADA PCI

- ⦿ Transaksi antara initiator (master) dg target
- ⦿ Master pegang kendali bus
- ⦿ Master menentukan jenis transaksi
  - Misal I/O read/write
- ⦿ Fase Address
- ⦿ Fase Data

# PCI READ TIMING DIAGRAM



# PCI BUS ARBITRATION



# INTERNET RESOURCE

- ◉ [www.pcguide.com/ref/mbsys/buses/](http://www.pcguide.com/ref/mbsys/buses/)
- ◉ [www.pcguide.com/](http://www.pcguide.com/)