I/O Interface

Pertemuan IX

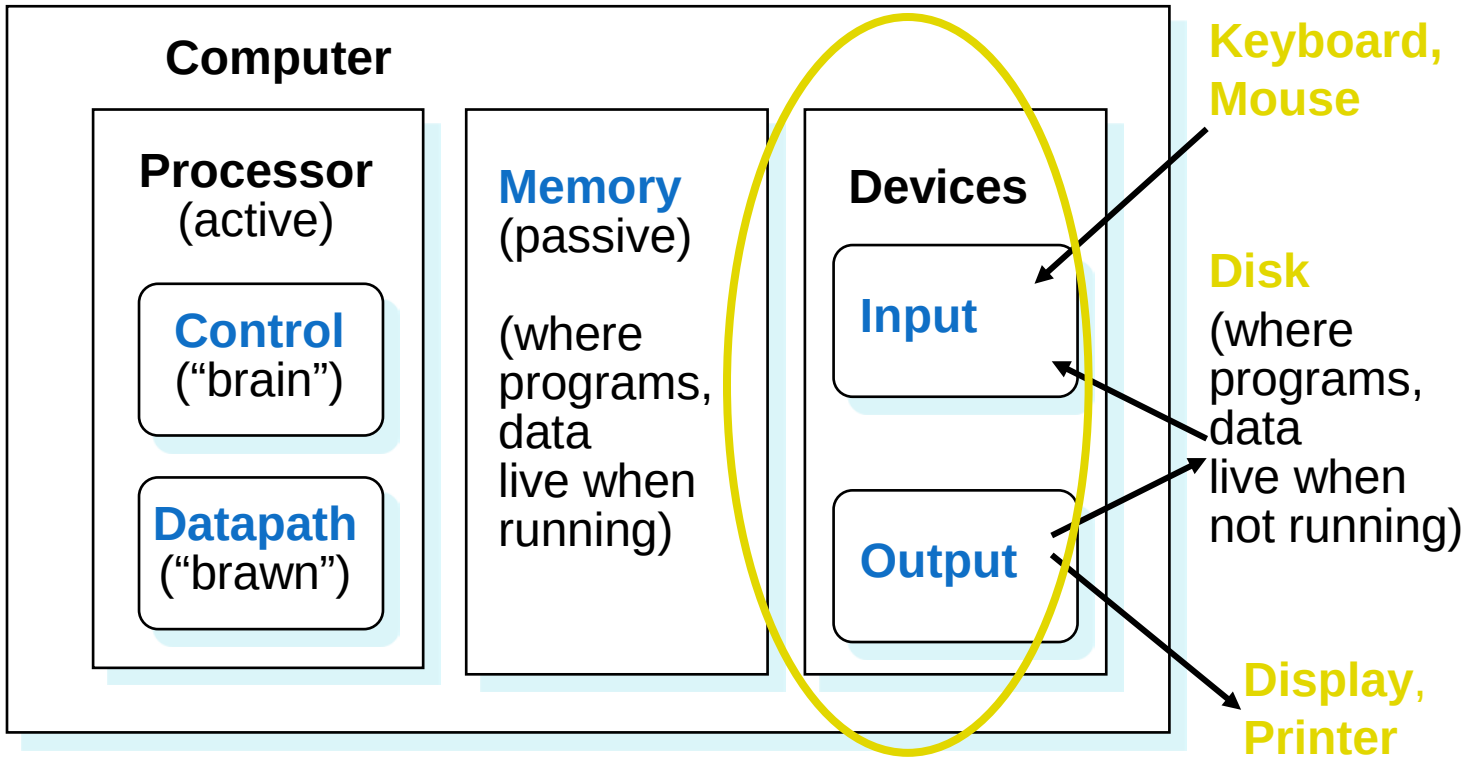
Pengantar

- Modul I/O: merupakan interface bagi bus sistem dan mengontrol satu atau lebih perangkat periferal
- Alasan tidak dihubungkannya periferal langsung dengan bus sistem:
  - Terdapat beberapa ragam piranti periferal yang memiliki bermacam-macam metode operasi
  - Laju transfer periferal jauh lebih lambat dibandingkan dengan laju transfer memori atau CPU
  - Periferal seringkali menggunakan format data atau panjang word yang berlainan dibandingkan dengan komputer yang disambungkan

Pengantar

- Fungsi modul I/o:
  - Sebagai interface ke CPU melalui bus sistem
  - Sebagai interface ke sebuah perangkat periferal atau lebih dengan menggunakan link data tertentu
- Perangkat eksternal mempunyai sifat:
  - Human Readable: display dan printer
  - Machine Readable: disk system, sensor, aktuator
  - Communication: LAN card

Input/Output: Gerbang Ke Dunia Luar

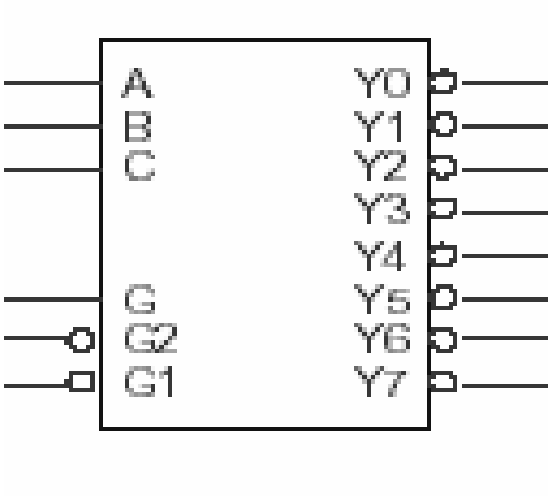


I/O Port Address Decoding

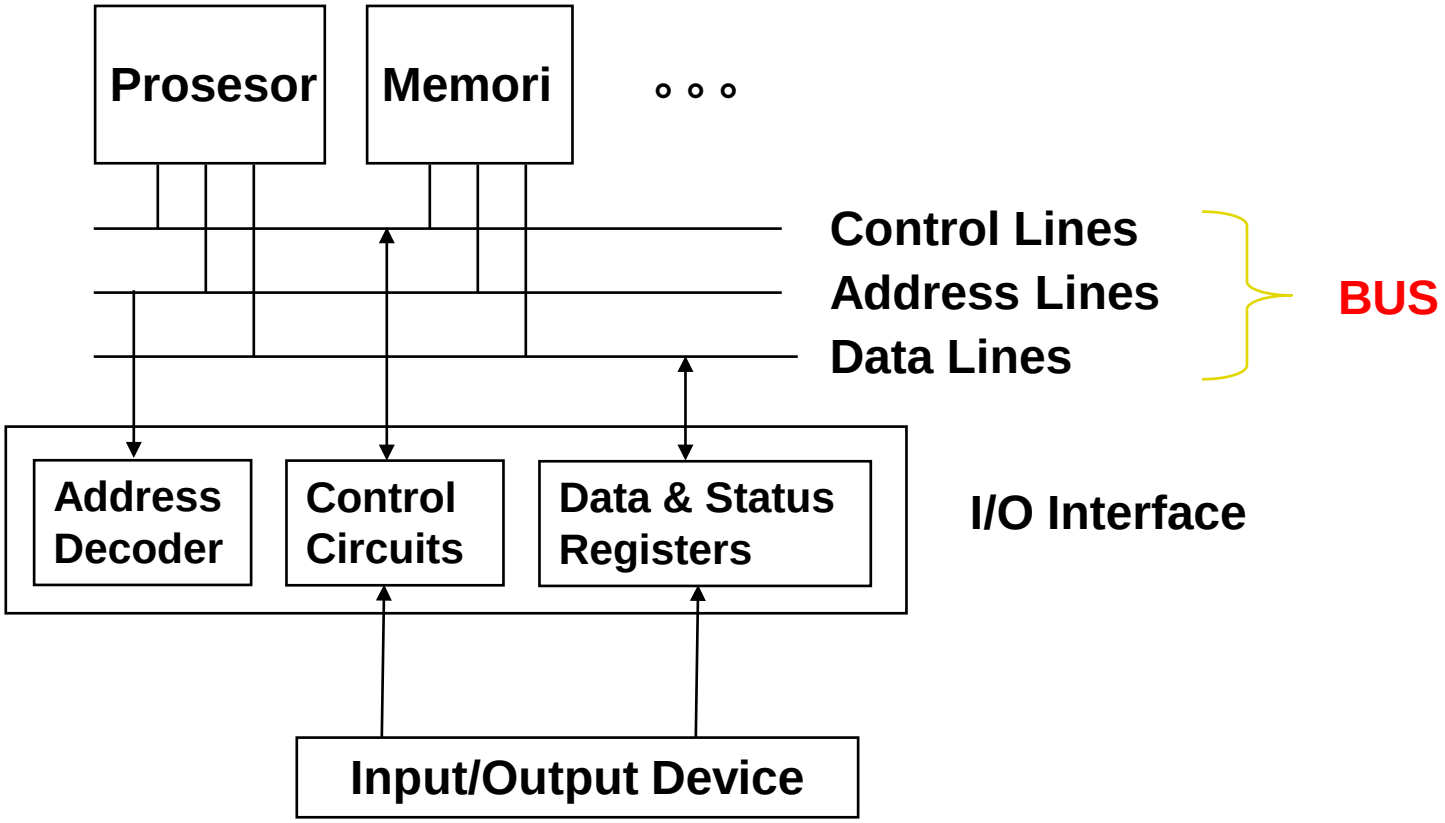
- I/O port address decoding = memory address decoding
- Perbedaan utama antara memori decoding dan isolated I/O decoding adalah banyaknya alamat pada pin yang terhubung ke decoder
- Perbedaan lainnya adalah penggunaan IOR dan IOW untuk mengaktifkan device I/O pada operasi BACA dan TULIS
- Pin mikroprosesor sebelumnya IO/M = ‘1’, dan pin RD dan RW digunakan untuk mengaktifkan device I/O
- Pada mikroproses terbaru M/IO = 0, dan pin W/R digunakan untuk mengaktifkan device I/O

I/O Port Address Decoding

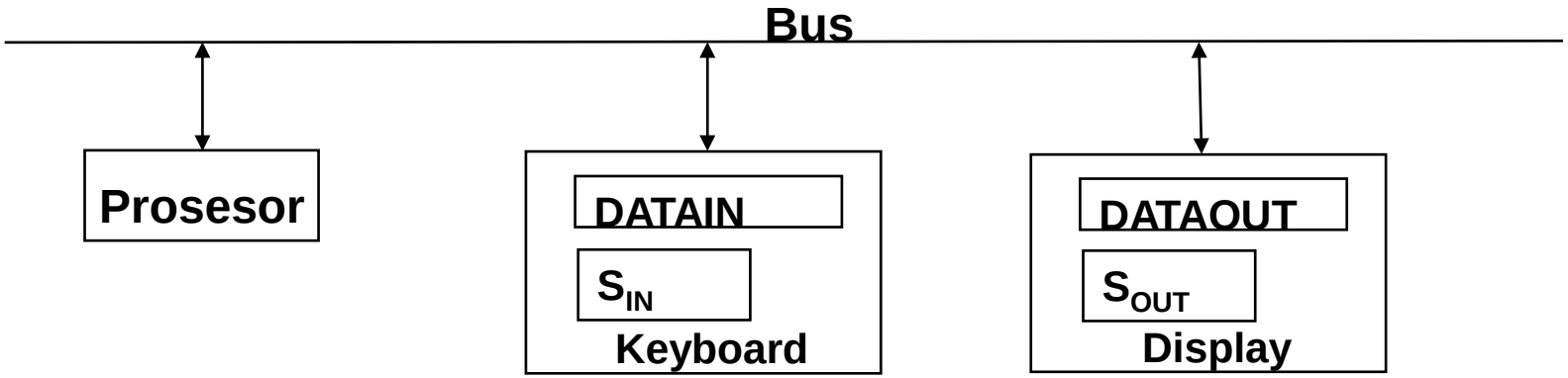
- **Decoding 8-Bit I/O Address**
  - Instruksi I/O yang ditetapkan pada 8-bit I/O untuk menggunakan alamat port yang tampak pada A_{15}-A_0 sebagai 0000H - 00FFH.
  - Ilustrasi pada Gambar IC 74ALS138 merupakan decoding 8-bit I/O.



Organisasi I/O



Organisasi I/O



- I/O Device biasanya memiliki 2 register:
  - 1 register menyatakan kesiapan untuk menerima/mengirim data (I/O ready), sering disebut Status/Control Register → S_{IN}, S_{OUT}
  - 1 register berisi data, sering disebut Data Register → DATAIN, DATAOUT
- Prosesor membaca isi Status Register terus-menerus, menunggu I/O device men-set Bit Ready di Status Register (0 → 1)
- Prosesor kemudian menulis atau membaca data ke/dari Data Register
  - tulis/baca ini akan me-reset Bit Ready (1 → 0) di Status Register

Contoh Program Input/Output

- **Input: Read from keyboard**

```
                Move        #LOC,R0          ; Initialize memory
READ:           In          INSTATUS,R1      ; Read status
                TestBit     #3,R1            ; Keyboard (IN) ready?
                Branch=0    READ             ; Wait for key-in
                In          DATAIN,R1        ; Read character
                Move        R1,(R0)          ; Store in memory
```

- **Output: Write to display**

```
ECHO:           In          OUTSTATUS,R1; Read status
                TestBit     #3,R1            ; Display (OUT) ready?
                Branch=0    ECHO             ; Wait for it
                Move        (R0),R1          ; Get the character
                Out         R1,DATAOUT       ; Write it
                Compare     #CR,(R0)+        ; Is it CR?
                                             ; Meanwhile, stores it
                Branch≠0    READ             ; No, get more
                Call        Process          ; Do something
```

Ketidaksesuaian Kecepatan Prosesor dan I/O

Ilustrasi

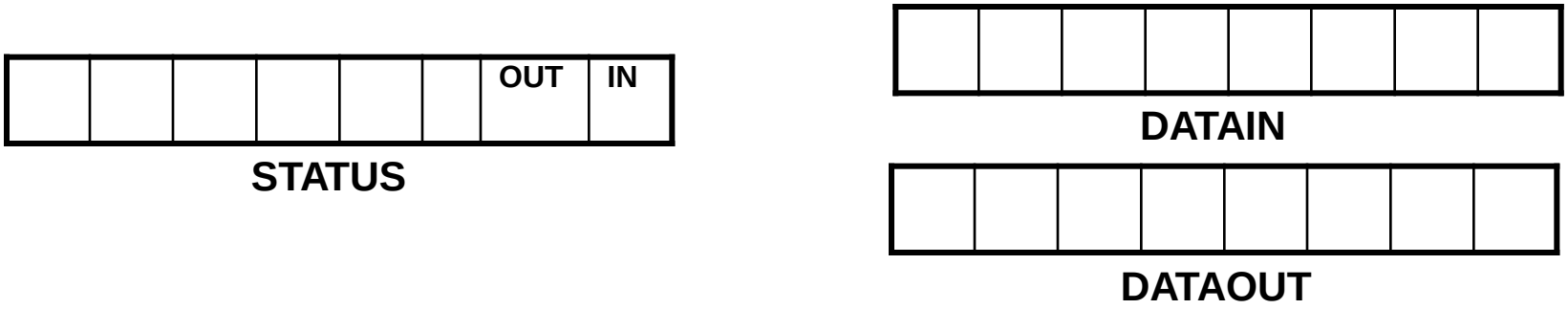
- Mikroprosesor 500 MHz dapat mengeksekusi 500 juta instruksi per detik atau menyimpannya.
  - Piranti I/O hanya memiliki kecepatan 0,01 KB/s sampai ke 30.000 KB/s.
- Piranti Input mungkin tidak dapat untuk mengirim data secepat prosesor . Juga, mungkin akan menunggu bagi manusia untuk bertindak.
- Piranti Output juga mungkin tidak dapat menerima data secepat prosesor
- Jadi apa yang harus dilakukan??

Polling

- Menentukan status device
  - `command-ready`
  - `busy`
  - `Error`

- Siklus busy wait ke wait untuk I/O dari device

Program-Controlled I/O: Polling



- Input: Read from keyboard

```
                Move        R1,#line        ; Initialize memory
WAITK:          TestBit     STATUS,#0       ; Keyboard (IN) ready?
                Branch=0    WAITK           ; Wait for key-in
                Move        R0,DATAIN       ; Read character
```

- Output: Write to display

```
WAITD:          TestBit     STATUS,#1       ; Display (OUT) ready?
                Branch=0    WAITD           ; Wait for it
                Move        DATAOUT,R0      ; Write character
                Move        (R1)+,R0        ; Store & advance
                Compare     R0,#$0D         ; Is it CR?
                Branch≠0    WAITK           ; No, get more
                Call        Process         ; Do something
```

- Processor waiting for I/O called "Polling"

Biaya Polling

Asumsikan untuk prosesor dengan clock 500 MHz dibutuhk an 400 siklus clock untuk operasi polling (rutin panggilan polling, mengakses perangkat, dan kembali). Tentukan % dari waktu prosesor untuk polling :

- Mouse: polled 30 times/sec so as not to miss user movement
- Floppy disk: transfers data in 2-byte units and has a data rate of 50 KB/second.
  No data transfer can be missed.
- Hard disk: transfers data in 16-byte chunks and can transfer at 8 MB/second. Again, no transfer can be missed.

% Processor time to poll mouse

- Times Mouse Polling/sec

  = 30 polls/sec

- Mouse Polling Clocks/sec

  = 30 * 400 = 12000 clocks/sec

- % Processor for polling:

  $12*10^3/500*10^6 = 0.002\%$

  → Polling mouse little impact on processor

% Processor time to poll Floppy

- Times Polling Floppy/sec

  = 50 KB/s /2B = 25K polls/sec

- Floppy Polling Clocks/sec

  = 25K * 400 = 10,000,000 clocks/sec

- % Processor for polling:

  $10*10^6/500*10^6 = 2\%$

  → OK if not too many I/O devices

% Processor time to hard disk

- Times Polling Disk/sec

  = 8 MB/s /16B = 500K polls/sec

- Disk Polling Clocks/sec

  = 500K * 400 = 200,000,000 clocks/sec

- % Processor for polling:

  $200*10^6/500*10^6 = 40\%$

  → Unacceptable