Sistem Operasi

Divais Input/Output

2016

UNIVERSITAS Telkom

Microsoft

has combined the strengths of its three most powerful operating systems to create its next generation operating system.

Microsoft Windows CE

Microsoft Windows Me Millennium Edition

Windows NT Workstation

Microsoft Windows CEMeNT

As hard as a rock and as dumb as a brick

Kata Pengantar

- Salah satu fungsi OS adalah mengendalikan divais
  - Merupakan sebagian besar code (80-90% pada Linux)
- Diinginkan semua divais digunakan
  - nyaman
  - misal: stdin/stdout, pipe, re-direct
- Diinginkan optimasi akses ke divais
  - efisien
  - setiap divais punya keperluan yang berbeda

Outline

- Pengantar ← selesai
- Prinsip Hardware ←
  - Divais I/O
  - Pengendali divais
  - I/O dipetakan ke memori
- Prinsip Software
- Lapis-lapis Software

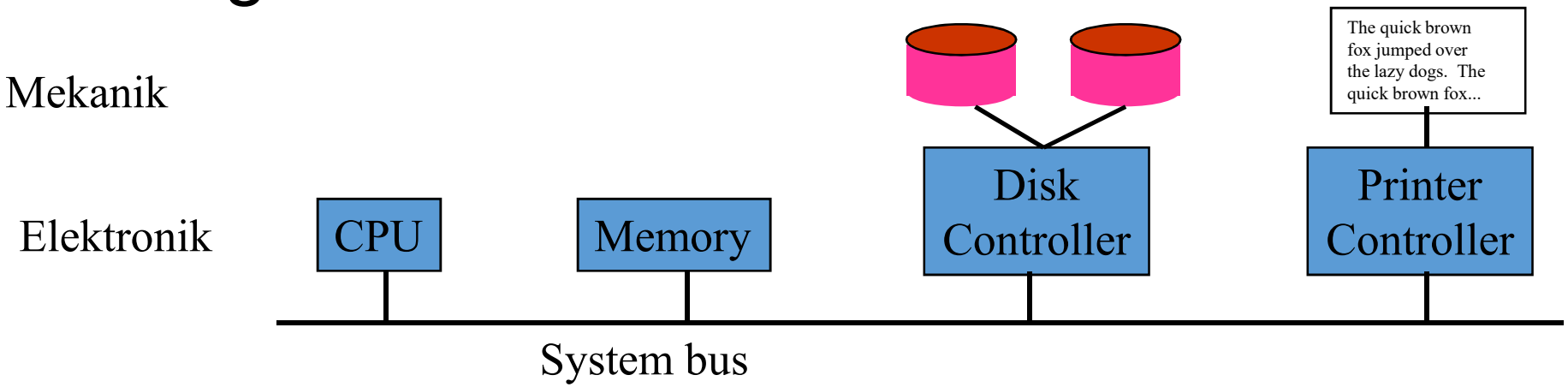
Divais I/O

- Divais blok termasuk disk drive
  - Divais dialamati per blok
  - Setiap blok independen
  - Command termasuk `read, write, seek`
- Divais karakter termasuk keyboard, mouse, serial port, USB
  - Mendukung karakter stream
  - Akses tidak beralamat,serial
  - Command termasuk `get, put`

Divais I/O

Device	Data rate
Keyboard	10 bytes/sec
Mouse	100 bytes/sec
56K modem	7 KB/sec
Telephone channel	8 KB/sec
Dual ISDN lines	16 KB/sec
Laser printer	100 KB/sec
Scanner	400 KB/sec
Classic Ethernet	1.25 MB/sec
USB (Universal Serial Bus)	1.5 MB/sec
Digital camcorder	4 MB/sec
IDE disk	5 MB/sec
40x CD-ROM	6 MB/sec
Fast Ethernet	12.5 MB/sec
ISA bus	16.7 MB/sec
EIDE (ATA-2) disk	16.7 MB/sec
FireWire (IEEE 1394)	50 MB/sec
XGA Monitor	60 MB/sec
SONET OC-12 network	78 MB/sec
SCSI Ultra 2 disk	80 MB/sec
Gigabit Ethernet	125 MB/sec
Ultrium tape	320 MB/sec
PCI bus	528 MB/sec
Sun Gigaplane XB backplane	20 GB/sec

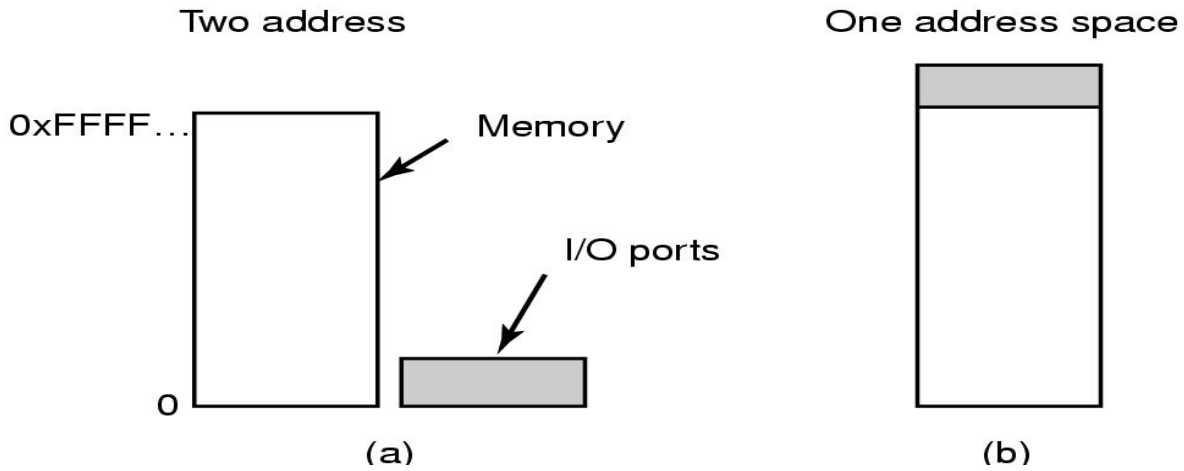
Pengendali Divais



- OS berhubungan dengan elektronik
  - Kendali divais
    - Standar!
    - Sebuah contoh tentang disk

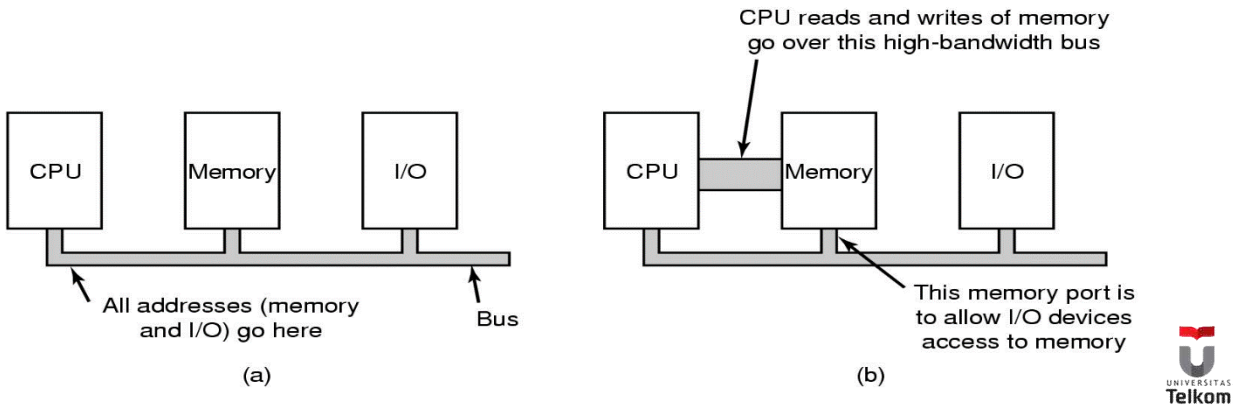
I/O dipetakan ke memori

- Penyimpanan di kendali divais
  - Register: status, command, data …
  - Buffer: lebih banyak data.
- Pendekatan alamat



I/O dipetakan ke memori

- Kelebihan pendekatan B
  - Tidak perlu instruksi I/O khusus
  - Tidak perlu proteksi khusus
  - Instruksi lebih sederhana: misal TEST PORT_4
- Kekurangan pendekatan B
  - Caching
  - Bus Ganda



Outline

- Pengantar ← selesai
- Prinsip Hardware ← selesai
  - Divais I/O
  - Pengendali divais
  - I/O dipetakan ke memori
- Prinsip Software ←
- Lapis-lapis Software

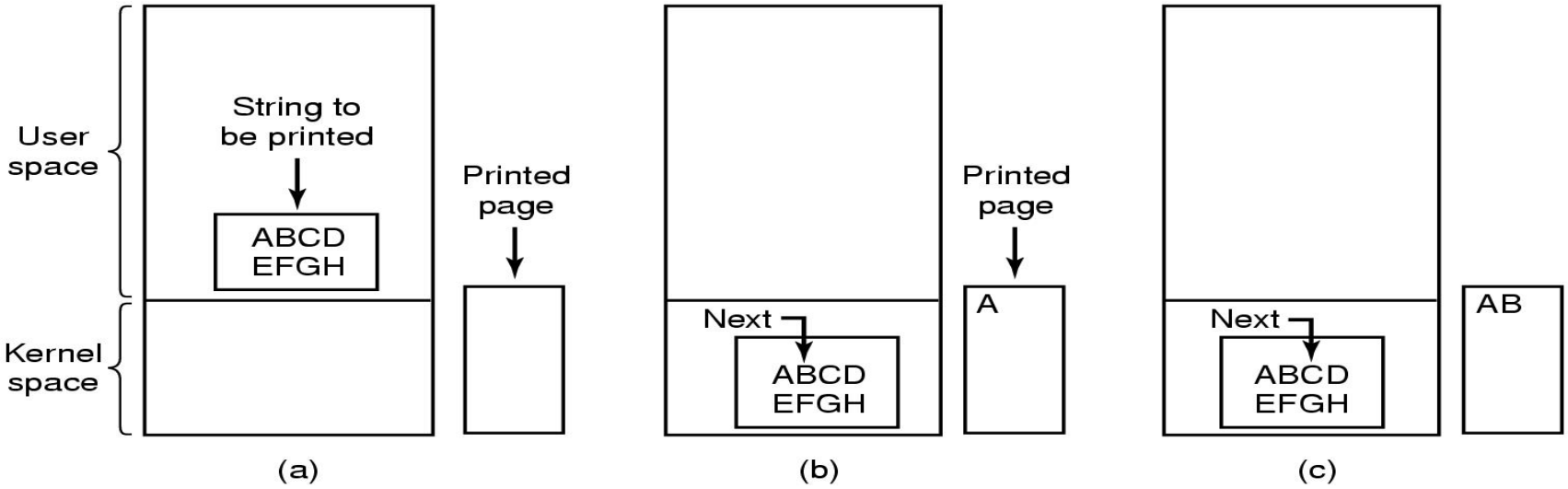
Polling

- Register
  - Command
  - Data-In & Data-Out
  - Status
    - `command-ready`
    - `busy`
    - `Error`
- Siklus busy-wait untuk menunggu I/O dari divais

Misal menulis output

- Host berulang kali membaca bit *busy* di register *status* sampai bit itu menjadi clear
- Host men-set bit *write* di register *command* dan menulis sebuah byte ke register *data-out*
- Host men-set bit *cmd-ready* di regsiter *cmd*
- Ketika pengendali melihat bit *cmd-ready* kondisi set, dia akan men-set bit *busy*
- Pengendali membaca register *cmd* dan melihat *write* cmd. Dia membaca reg *data-out* untuk mendapatkan data dan melakukan I/O ke divais.
- Pengendali men-clear bit *cmd-ready*, men-clear bit *error* dan bit *busy* di register *status*.

Polling



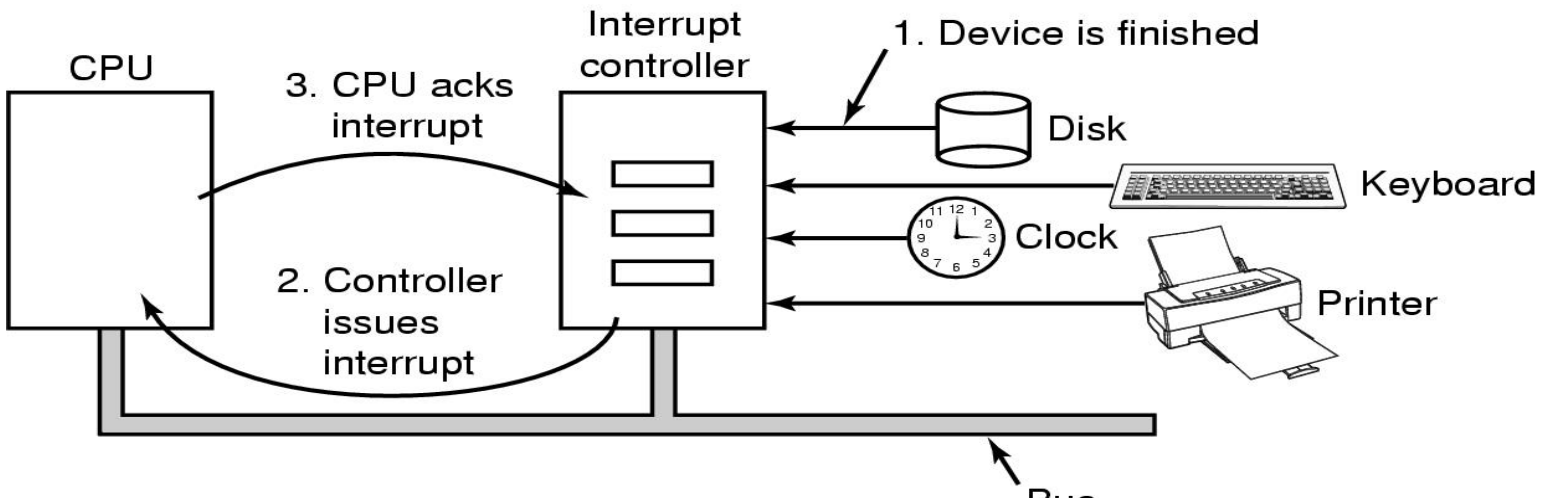
Tahapan mencetak sebuah string

Polling

```
copy_from_user(buffer, p, count);                  /* p is the kernel bufer */
for (i = 0; i < count; i++) {                      /* loop on every character */
        while (*printer_status_reg != READY) ;     /* loop until ready */
        *printer_data_register = p[i];             /* output one character */
}
return_to_user( );
```

Mencetak sebuah string ke printer menggunakan polling

Interrupt



Bagaimana interrupt terjadi. Hubungan antara divais dan pengendali interrupt sesungguhnya menggunakan jalur interrupt di bus daripada saluran khusus

Interrupt-Driven I/O

```
copy_from_user(buffer, p, count);
enable_interrupts( );
while (*printer_status_reg != READY) ;
*printer_data_register = p[0];
scheduler( );
```

(a)

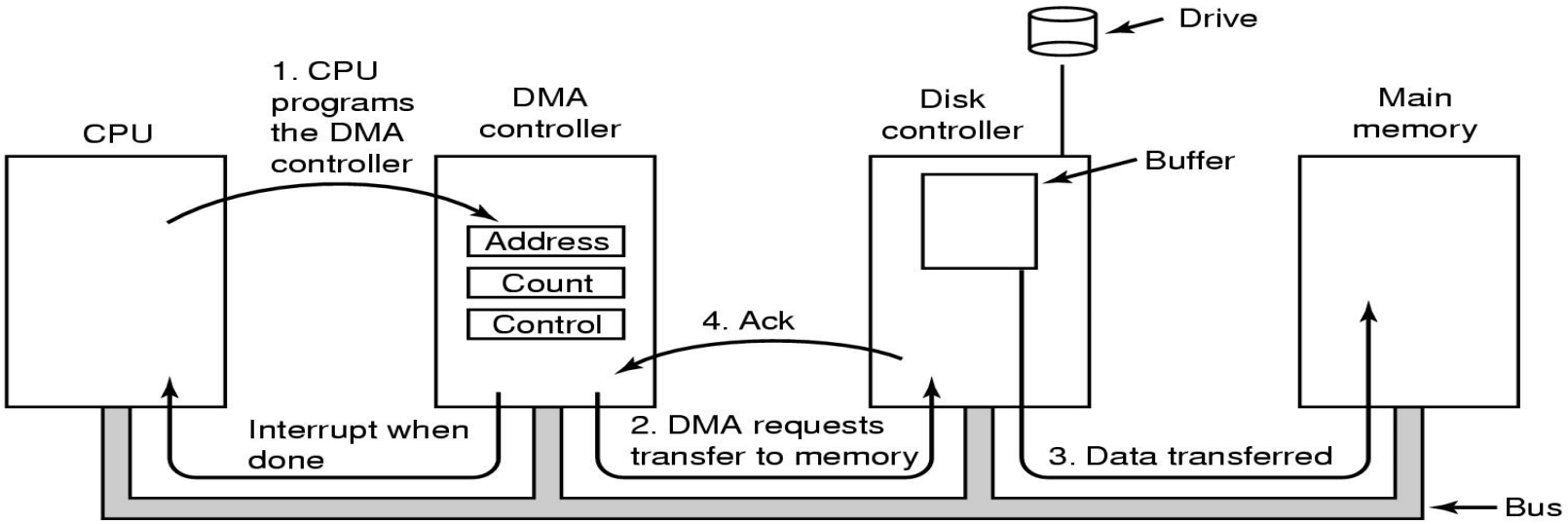
```
if (count == 0) {
    unblock_user( );
} else {
    *printer_data_register = p[i];
    count = count – 1;
     i = i + 1;
}
acknowledge_interrupt( );
return_from_interrupt( );
```

(b)

- Mencetak sebuah string ke printer menggunakan interrupt-driven I/O
  - (a) Code dieksekusi ketika call sistem print dibuat
  - (b) Prosedur layanan interrupt
- Masalah?
  - Terlampau banyak interrupt!

Direct Memory Access (DMA)

Operasi sebuah transfer DMA



I/O Menggunakan DMA

```
copy_from_user(buffer, p, count);
set_up_DMA_controller( );
scheduler( );
```

(a)

```
acknowledge_interrupt( );
unblock_user( );
return_from_interrupt( );
```

(b)

- Mencetak sebuah string ke printer menggunakan DMA
  - code dieksekusi ketika call sistem print dibuat
  - Prosedur layanan interrupt

Outline

- Pengantar ← selesai
- Prinsip Hardware ← selesai
  - Divais I/O
  - Pengendali divais
  - I/O dipetakan ke memori
- Prinsip Software ← selesai
- Lapis-lapis Software ←
  - Interrupt Handler
  - Device Driver
  - Software I/O independen divais
  - Software I/O ruang pengguna

Interrupt Handler

1. Simpan register yang belum disimpan oleh hardware interrupt
2. Set up context untuk prosedur layanan interrupt
3. Set up stack untuk prosedur layanan interrupt
4. Ack pengendali interrupt, reenable interrupt
5. Copy register dari tempatnya disimpan
6. Jalankan prosedur layanan
7. Set up context MMU untuk proses yang dijalankan berikutnya
8. Load register proses baru
9. Mulai menjalankan proses baru

Pentium Event-Vector Table

vector number	description
0	divide error
1	debug exception
2	null interrupt
3	breakpoint
4	INTO-detected overflow
5	bound range exception
6	invalid opcode
7	device not available
8	double fault
9	coprocessor segment overrun (reserved)
10	invalid task state segment
11	segment not present
12	stack fault
13	general protection
14	page fault
15	(Intel reserved, do not use)
16	floating-point error
17	alignment check
18	machine check
19Ð31	(Intel reserved, do not use)
32Ð255	maskable interrupts

Device Driver

- Pengendali divais berbeda-beda
- Device driver
  - Code tergantung hardware
  - Oleh pembuat divais
- Menerima permintaan abstrak dari lapis atas
  - misal: “read block `n`”
- Struktur umum
  - Periksa validitas dari permintaan
  - Periksa jika divais sedang dipergunakan
  - Mulai mengeluarkan urutan command
    - Akses register dan buffer divais
  - Blok (dirinya sendiri) sampai interrupt datang (kadangkala tidak datang)
  - Periksa error dan kirimkan data ke software independen divais

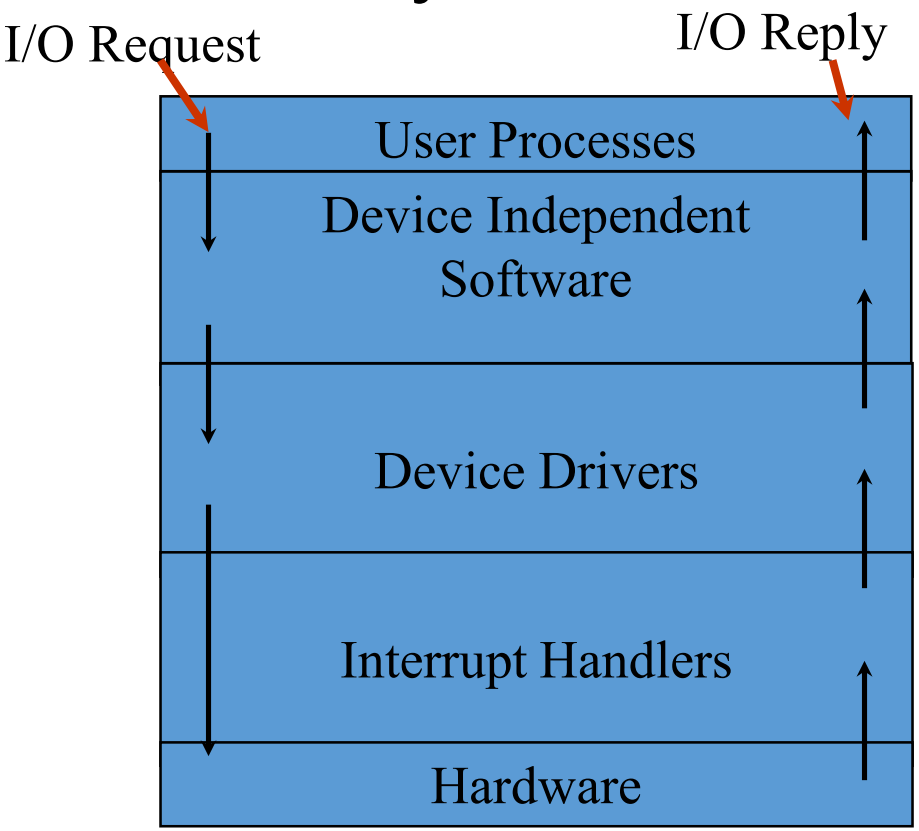
Software I/O Independen Divais

- Banyak code driver independen terhadap divais
- Lakukan fungsi I/O umum untuk semua divais
- Contoh:
  - Uniform Interfacing
  - Buffering → contoh : print spooler di windows
  - Error reporting
  - Allocating and Releasing
  - Device-Independent Block Size

Software I/O Ruang Pengguna

- misal: `count = write(fd, buffer, bytes);`
- Taruh parameter di tempat untuk system call
- Dapat melakukan hal yang lebih: formatting
  - `printf(), scanf()`
- Spooling
  - spool directory, daemon
  - misal: printing, networking

Summary Sistem I/O



Make I/O call; Format I/O;
Spooling
Naming, protection,
blocking, buffering,
allocation

Setup device registers;
check status

Wakeup driver when
I/O completed

Perform I/O operation