I/O

Disclaimer: some slides are adopted from book authors’ slides with permission
Concepts to Learn

• I/O subsystems
• Blocking, non-blocking, asynchronous I/O
• Memory-mapped I/O
• Programmed I/O vs. DMA
• Disk
Input/output (I/O) Subsystems
I/O Subsystems: the Goal

• Provide easy to use standardized interfaces
  – This code works for many different devices
    int fd = open("/dev/somedev", O_RDWR);
    char buf[80];
    for (int i = 0; i < 10; i++) {
      sprintf(buf, "i: %d\n", i);
      write(fd, buf, 80);
    }
    close(fd);
  – Hide the details of each device to users
Standard Device Types

• Block devices
  – E.g., disk, cd-rom, USB stick
  – High speed, block (sector) level accesses

• Character devices
  – E.g., keyboard, mouse, joystick
  – Low speed, character level accesses

• Network devices
  – E.g., ethernet, wifi, bluetooth
  – Socket interface
Types of I/O Operations

• Blocking I/O
  – Wait (i.e., the calling process is put to sleep) until the data is ready

• Non-blocking I/O
  – Immediately return to the caller no matter what.
  – I/O may not be completed

• Asynchronous I/O
  – Notify later when the I/O is completed (via callback or interrupts)
How Does CPU Talk to Devices?

- CPU talks to device controllers
  - Via I/O instructions or **memory mapped I/O**
# Memory Mapped I/O

<table>
<thead>
<tr>
<th>Base Address</th>
<th>Limit Address</th>
<th>Size</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0000_0000</td>
<td>0x0001_0000</td>
<td>64 KB</td>
<td>iROM</td>
</tr>
<tr>
<td>0x0200_0000</td>
<td>0x0201_0000</td>
<td>64 KB</td>
<td>iROM (mirror of 0x0 to 0x10000)</td>
</tr>
<tr>
<td>0x0202_0000</td>
<td>0x0206_0000</td>
<td>256 KB</td>
<td>iRAM</td>
</tr>
<tr>
<td>0x0300_0000</td>
<td>0x0302_0000</td>
<td>128 KB</td>
<td>Data memory or general purpose of Samsung Reconfigurable Processor SRP.</td>
</tr>
<tr>
<td>0x0302_0000</td>
<td>0x0303_0000</td>
<td>64 KB</td>
<td>L-cache or general purpose of SRP.</td>
</tr>
<tr>
<td>0x0303_0000</td>
<td>0x0303_9000</td>
<td>36 KB</td>
<td>Configuration memory (write only) of SRP</td>
</tr>
<tr>
<td>0x0381_0000</td>
<td>0x0383_0000</td>
<td>–</td>
<td>AudioSS’s SFR region</td>
</tr>
<tr>
<td>0x0400_0000</td>
<td>0x0500_0000</td>
<td>16 MB</td>
<td>Bank0 of Static Read Only Memory Controller (SMC) (16-bit only)</td>
</tr>
<tr>
<td>0x0500_0000</td>
<td>0x0600_0000</td>
<td>16 MB</td>
<td>Bank1 of SMC</td>
</tr>
<tr>
<td>0x0600_0000</td>
<td>0x0700_0000</td>
<td>16 MB</td>
<td>Bank2 of SMC</td>
</tr>
<tr>
<td>0x0700_0000</td>
<td>0x0800_0000</td>
<td>16 MB</td>
<td>Bank3 of SMC</td>
</tr>
<tr>
<td>0x0800_0000</td>
<td>0x0C00_0000</td>
<td>64 MB</td>
<td>Reserved</td>
</tr>
<tr>
<td>0x0C00_0000</td>
<td>0x0CD0_0000</td>
<td>–</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

- **USB, SD/MMC, Timer, ...**
- **DRAM**
- **SFR region of Nand Flash Controller (NFCON)**
- **SFR region**
- **Memory of Dynamic Memory Controller (DMC)-0**
- **Memory of DMC-1**

Samsung Exynos 4412 (ARM) Processor **Physical Address Map**
Memory Mapped I/O

• Parts of physical memory space are mapped to hardware controllers
  – Mapped to control registers and buffers
• Reading/writing from/to the memory mapped regions in device specific ways
  – Device drivers’ job
#define CTRL_BASE_ADDR 0xCE000000

int *io_base = (int *)ioremap_nocache(CRTL_BASE_ADDR, 4096);

// initialize the device (by writing some values to h/w regs)
*io_base = 0x1;
*(io_base + 1) = 0x2;
*(io_base + 2) = 0x3;
...
// wait until the device is ready (bit31 = 0)
while (*io_base & 0x80000000);

// send data to the device
for (i = 0; i < sizeof(buffer); i++) {
    *(io_base + 0x10) = buffer[i];
    while (*io_base & 0x80000000);
}
Data Transfer Methods

• Programmed I/O
  – Via CPU’s load/store instructions
  – Simple h/w, but high CPU load

• Direct Memory Access
  – Controllers directly read/write from/to DRAM
  – Interrupts the CPU on the completion of I/O ops.
  – Complex h/w, but low CPU overhead
Direct Memory Access

1. device driver is told to transfer disk data to buffer at address X
2. device driver tells disk controller to transfer C bytes from disk to buffer at address X
3. disk controller initiates DMA transfer
4. disk controller sends each byte to DMA controller
5. DMA controller transfers bytes to buffer X, increasing memory address and decreasing C until C = 0
6. when C = 0, DMA interrupts CPU to signal transfer completion
Interrupt Driven I/O Cycle

1. CPU
   - device driver initiates I/O
   - CPU executing checks for interrupts between instructions
   - CPU receiving interrupt, transfers control to interrupt handler
   - interrupt handler processes data, returns from interrupt
   - CPU resumes processing of interrupted task

2. I/O controller
   - initiates I/O

3. input ready, output complete, or error generates interrupt signal

4. CPU
   - resuming after handling the interrupt

5. CPU resumes processing of interrupted task

6. CPU resumes processing of interrupted task

7. CPU resumes processing of interrupted task
Disk

• Magnetic disks (HDD)
  – Still used as the main storage device on many computers
  – Mechanical device (moving parts)
  – Cheap but slow

• Solid-state disks (SSD)
  – All smartphones and tables, many notebooks
  – No moving parts, use NAND flash chips
  – Still a bit expensive but faster
The First Commercial Disk Drive

1956
IBM RAMDAC computer included the IBM Model 350 disk storage system

5M (7 bit) characters
50 x 24” platters
Access time = < 1 second
Magnetic Disk

- track $t$
- sector $s$
- cylinder $c$
- platter
- spindle
- arm assembly
- rotation
- read-write head
- arm
Hard Disk Drive (HDD)

- **Storage size**
  - ~ 3TB
- **Performance**
  - B/W: ~1Gb/s
  - Seek time: 3-12ms
Disk Scheduling

- Goal: minimize seek time
- FCFS, SSTF (shortest seek time first), SCAN

```
queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53
```

SCAN scheduling
NAND Flash Memory Chip

Figure source: Micron, "TN-29-19: NAND Flash 101", 2010
Solid-State Disk (SSD)

- Same I/f as HDD
  - SATA.
- Flash Translation Layer (FTL)
  - S/W running on the controller
  - Provides disk abstraction
- Storage size
  - ~1TB
- No seek time
- Bandwidth
  - SATA (6Gbps) is the bottleneck
  - Some use PCIe I/F
Summary

• I/O subsystems
  – Standardized interfaces to access various i/o devices

• I/O device types
  – Block, characters, network devices

• I/O mechanisms
  – Memory-mapped I/O, I/O instructions
  – Programmed I/O vs. DMA

• Disk
  – HDD vs. SDD