Lecture 8

Midterm Review

I/O and DMA

Disks, block devices, RAID

Midterm Review

Q1: Context Switching

main

1. call getline

2. call putline

putline

call putchar

putchar

... yield

getline

call getchar

if c = \n
getchar

... call yield

loop
Q2:

$$A = \frac{n_1 + n_2 + n_3}{3}$$

```
count = 0
sum = 0, average.
condition c
m1 (=m2 =m3) (x)
    count ++ ; sum + = x
    if count <= 3
        wait c:
    else
        broadcast c
return sum/3
count = 0
average = sum/3
sum = 0
return avg
```
bool m1_busy
cond c1
m1:
  if m1_busy
    wait c1
  m1_busy = T
  \{ code from above \}
  m1_busy = F
signal c1.

I/O and DMA:

\[ \text{CPU} \rightarrow \text{Memory} \rightarrow \text{Frame Buffer} \rightarrow \text{Frame Buffer} \rightarrow \text{RAM} \]

\[ \text{F}_{xxx} \rightarrow \text{Frame Buffer} \]

\[ \text{EPFF} \rightarrow \text{RAM} \]

\[ 0000 \rightarrow \text{RAM} \]

\[ \text{DDR} \rightarrow \text{PCI} \]

\[ \rightarrow 1000 \text{ cycles} \]

\[ \rightarrow 300 \text{ cycles} \]

\[ \rightarrow 1 \text{ cycle} \]

\[ \rightarrow 3 \text{GHz} \]

DMA: direct memory access
CPU

→ prepare DMA desc.
   cmd, (transmit a packet, read acmd)
   ptr
   len

→ put desc in circular buffer

→ update "head" register
   read status
   done

DEVICE
→ read desc.
   read desc
   [read data]
   do something
   [write data]
   set status
   → interrupt.

Io device sets the status in the DMA register.

CPU

← go

[ device

← descriptor

← data

← interrupt
DMA: Allows I/O devices to access system memory directly.

Timeline of I/O from process point of view:

User: `read()`

Kernel: `wait()` -> `signal()` - called by interrupt handler

Hardware: `interrupt`

- Each I/O device can have multiple buffers. The buffers are in memory.
- DMA reduces latency for reading/writing.
- Device drivers reside in kernel & they use physical memory addresses.