FIT9137
Introduction to Computer Architecture and Networks

Week 4: Computer I/O Operations and Managing Processes

Safi Uddin

www.shutterstock.com • 548025055

Assignment 1

Assignment 1 released!

See Assessments section on Moodle

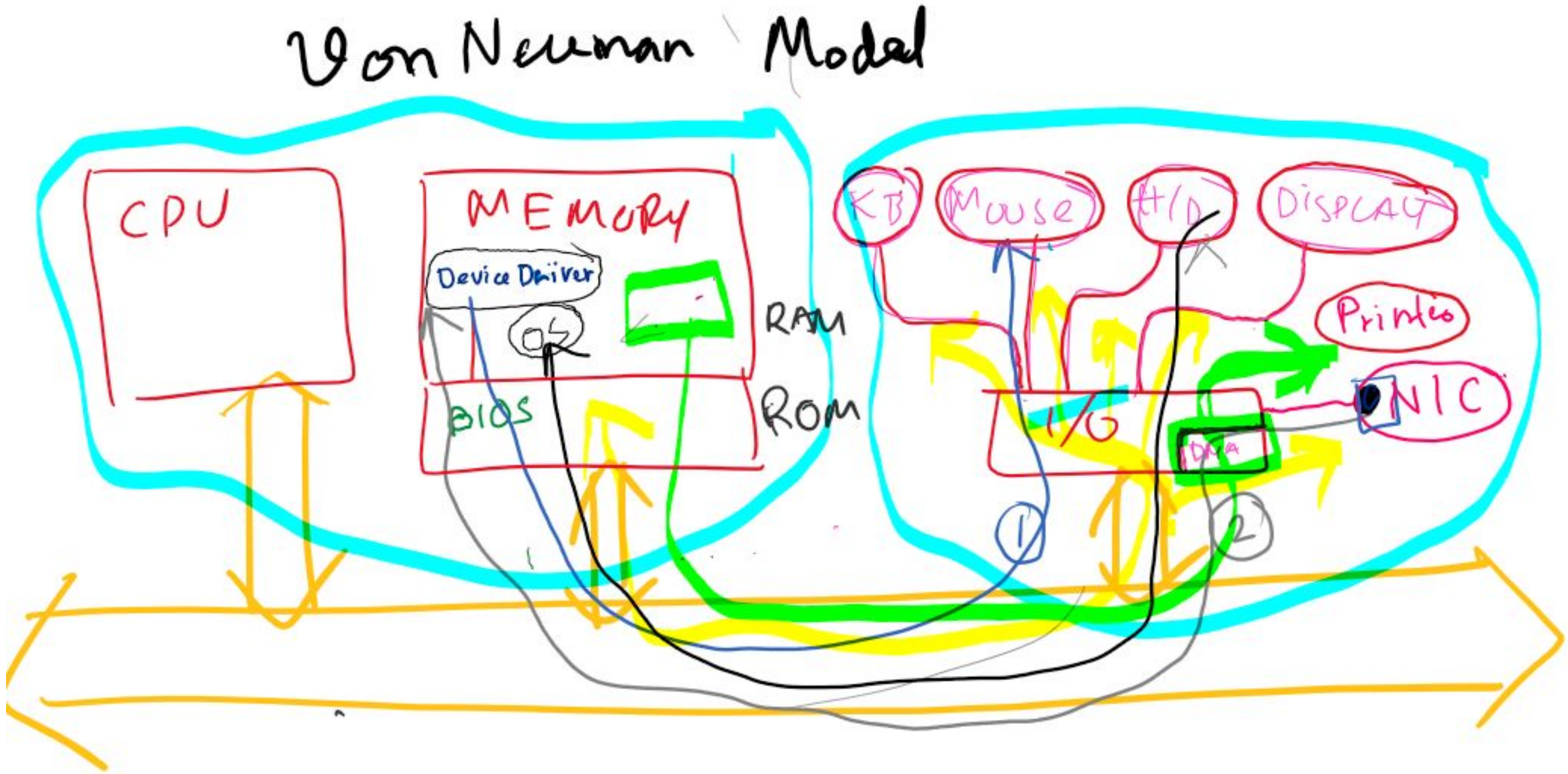
Due in Week 5

Activity A:

Different types of Input/Output (I/O) Mechanisms

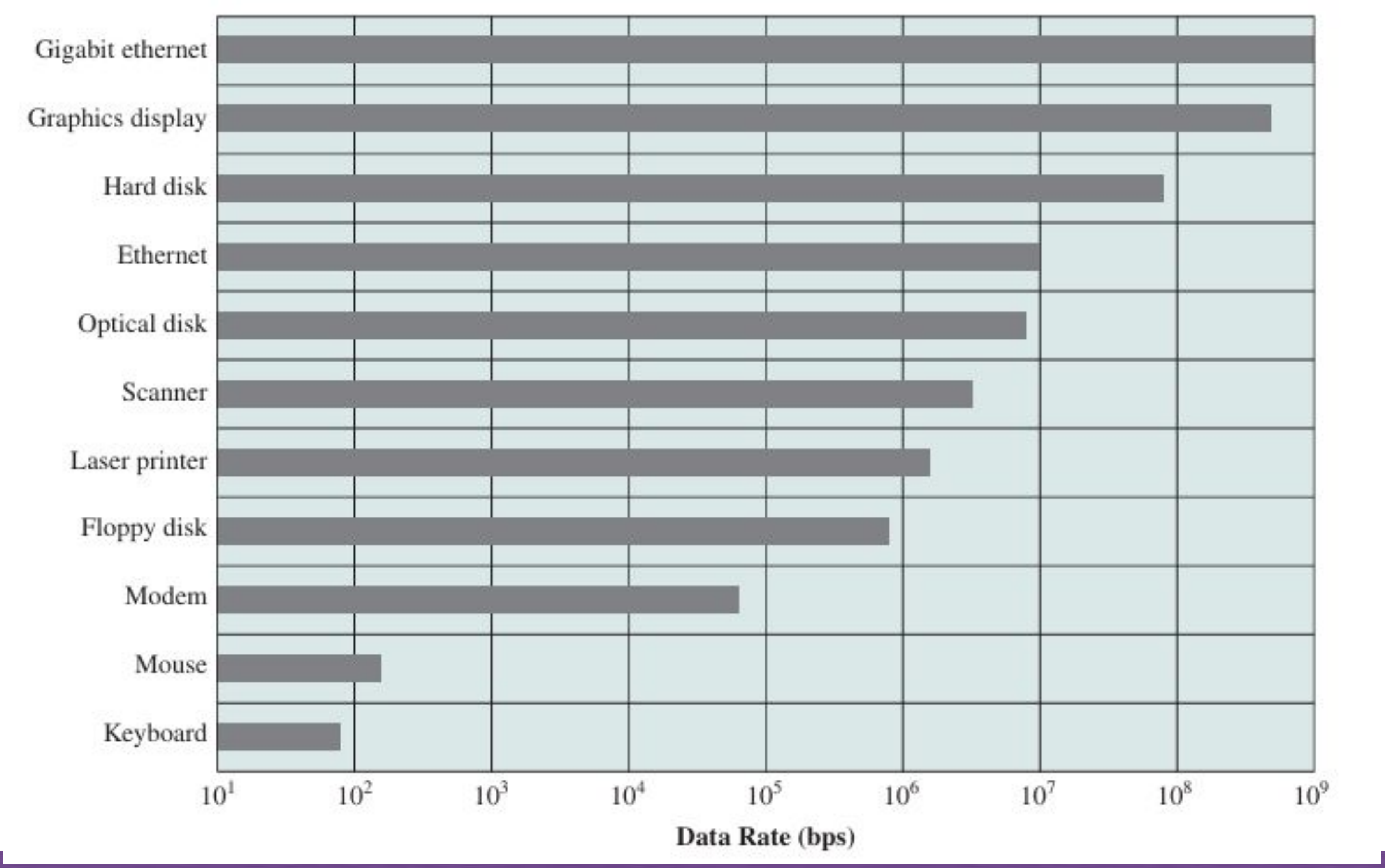
I/O in Computer Systems

- I/O vs Processing
- Management of I/O is important



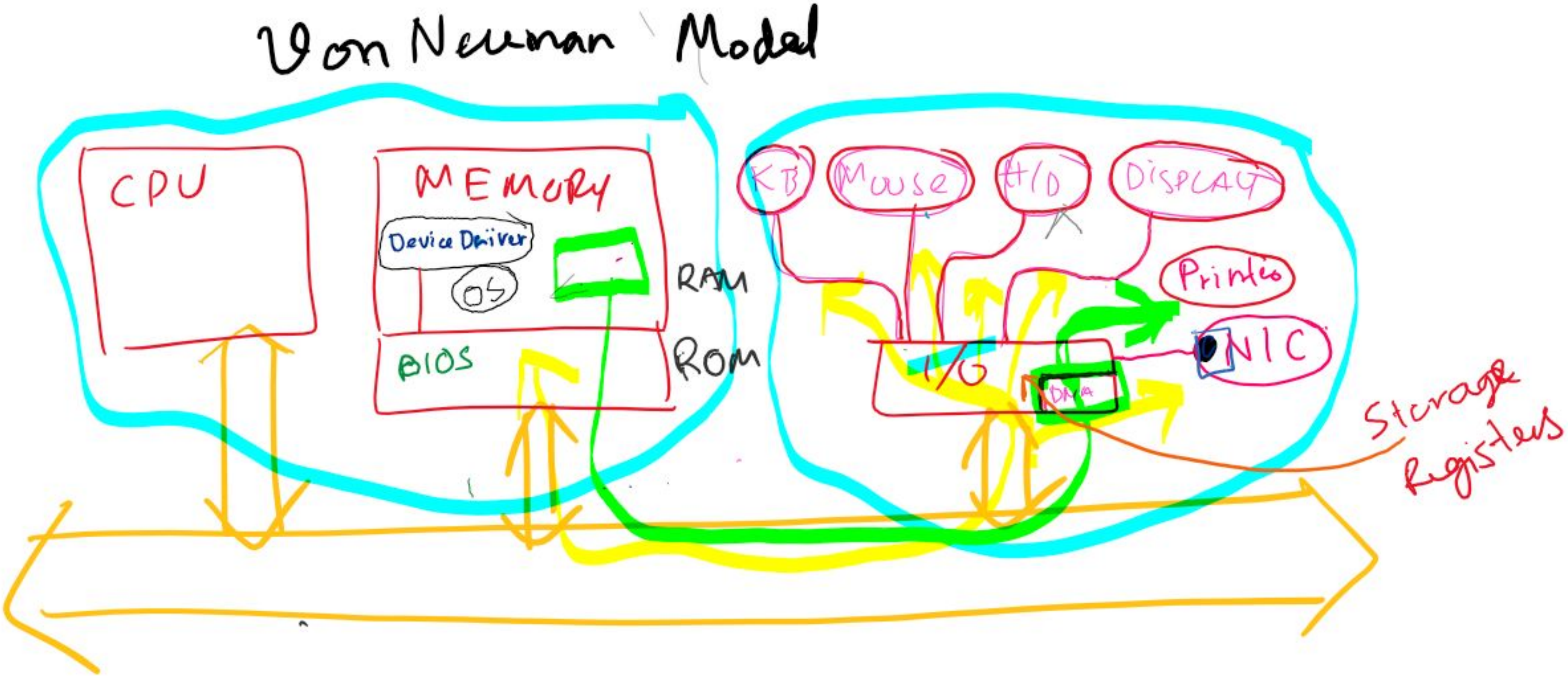
I/O in Computer Systems

- Varying devices in I/O
- requires different mode of operation.



CPU and I/O Interactions

- Who are involved?
  - CPU (OS), I/O Device, I/O Device Controller,
- I/O controller has registers for storage (small).

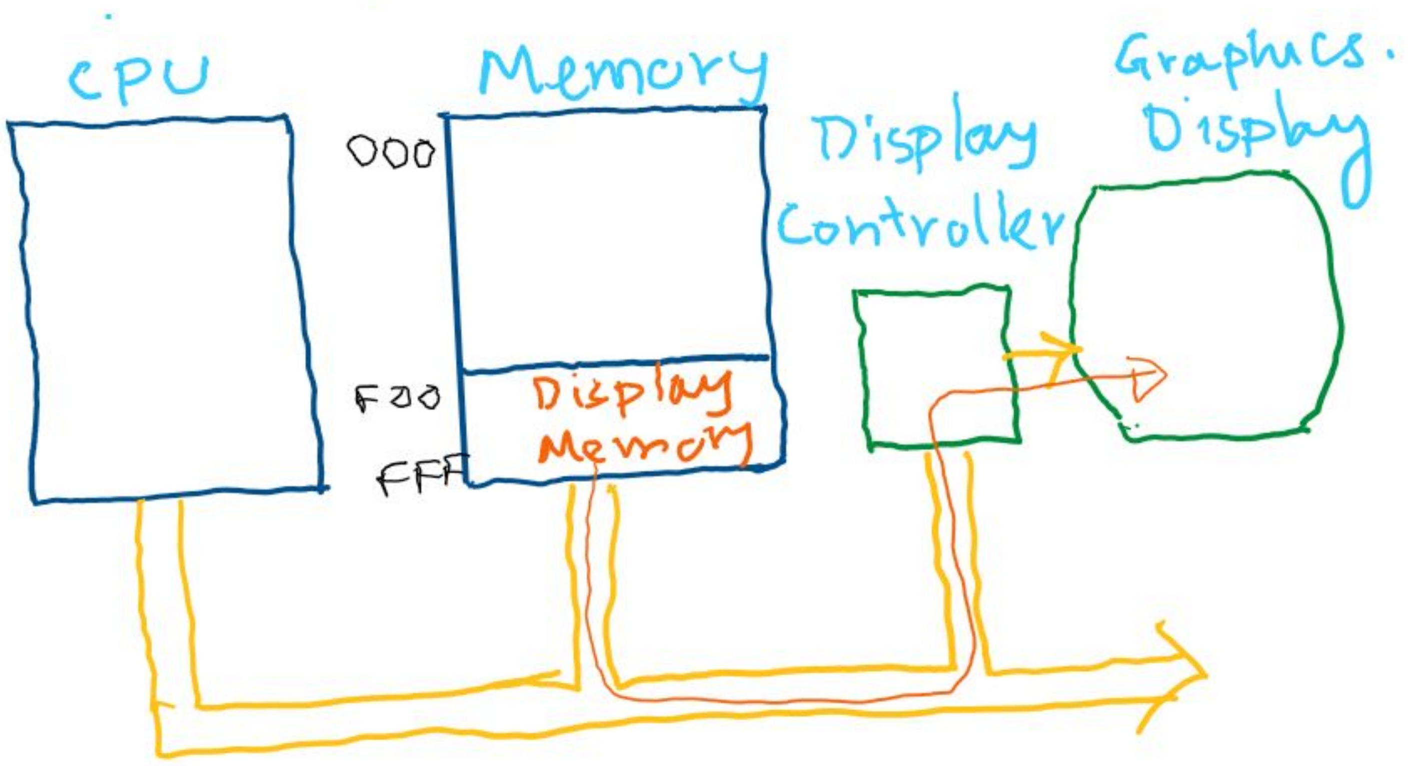


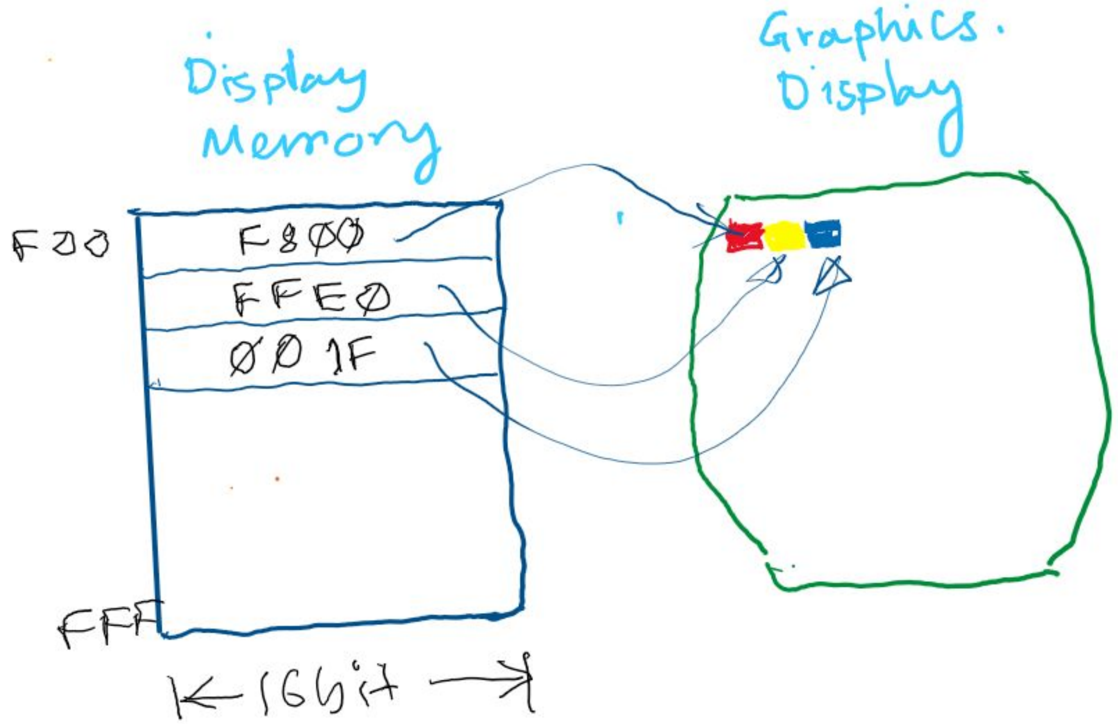
CPU and I/O Interactions:

Memory-mapped I/O

CPU and I/O Interactions:
Memory-mapped I/O

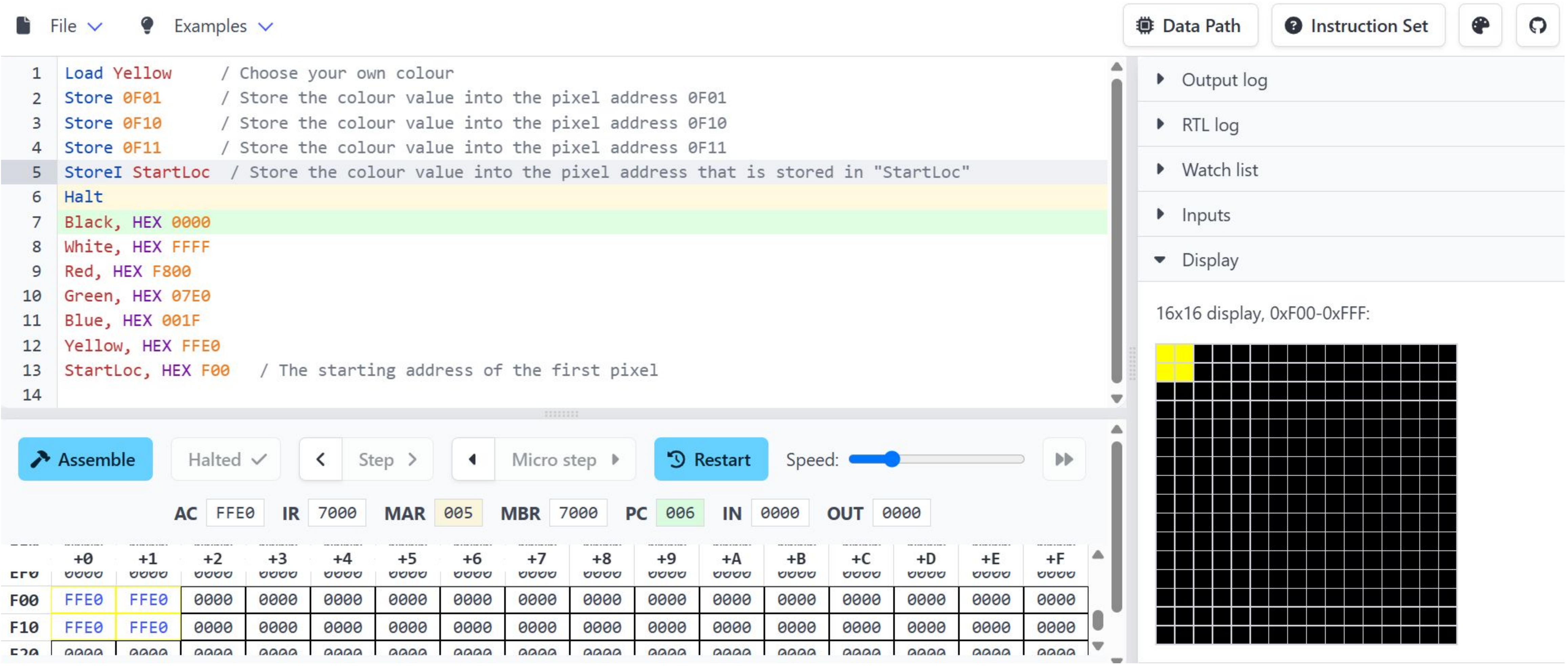
- Device-controller registers are mapped into the address space of the CPU.
- MARIE address space F00 - FFF -> Graphics Display
- Use standard MARIE instructions: “Store F00”





CPU and I/O Interactions: Memory-mapped I/O

- MARIE address space F00 - FFF -> Graphics Display
- Sequence of MARIE instructions to generate graphics display.



```
Load Yellow      / Choose your own colour
Store 0F01       / Store the colour value into the pixel address 0F01
Store 0F10       / Store the colour value into the pixel address 0F10
Store 0F11       / Store the colour value into the pixel address 0F11
StoreI StartLoc  / Store the colour value into the pixel address that is stored in "StartLoc"
Halt
Black, HEX 0000
White, HEX FFFF
Red, HEX F800
Green, HEX 07E0
Blue, HEX 001F
Yellow, HEX FFE0
StartLoc, HEX F00    / The starting address of the first pixel
```

AC FFE0 IR 7000 MAR 005 MBR 7000 PC 006 IN 0000 OUT 0000

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
EF0	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
F00	FFE0	FFE0	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
F10	FFE0	FFE0	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
F20	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

16x16 display, 0xF00-0xFFF:

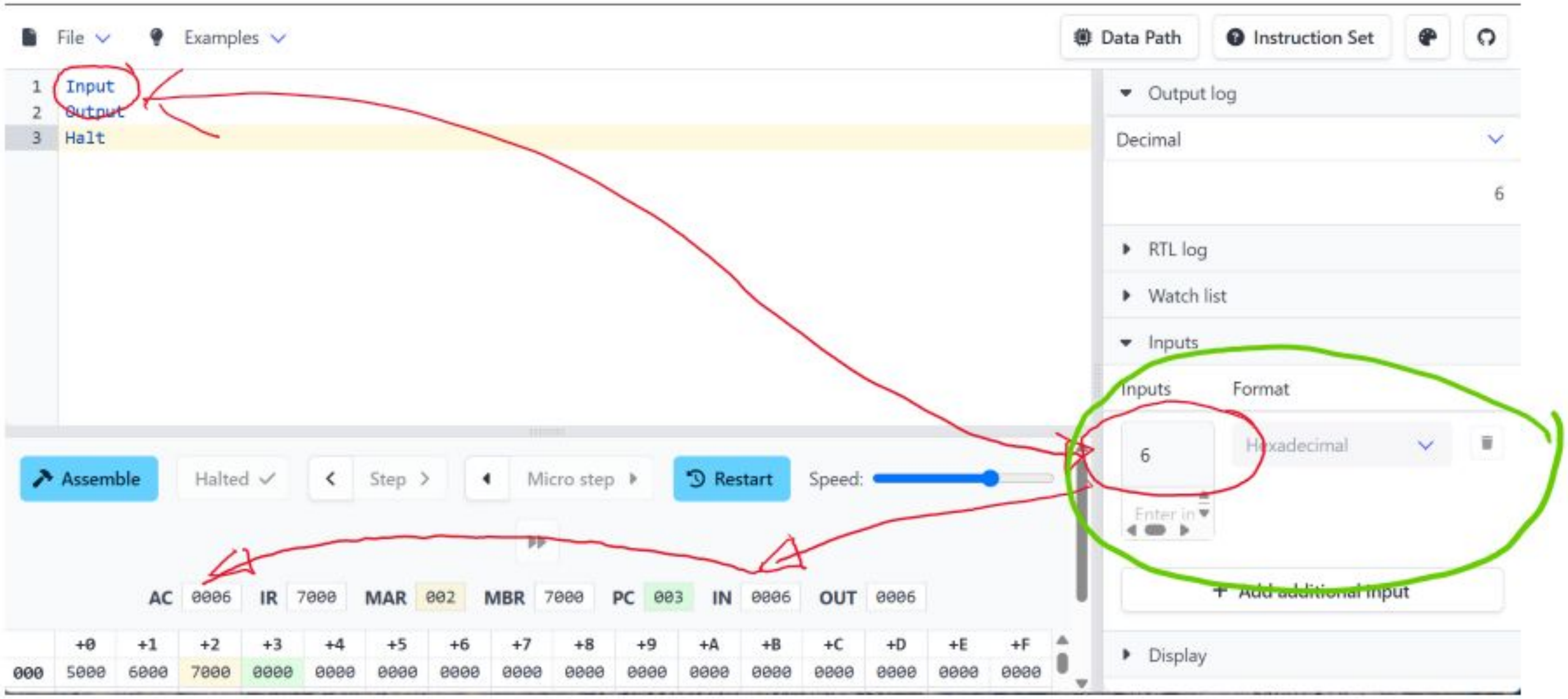
CPU and I/O Interactions:
Memory-mapped I/O

CPU and I/O Interactions:

Port-mapped I/O

CPU and I/O Interactions:
Port-mapped I/O

- Use of special I/O instructions that triggers the transfer to an I/O port address.
- MARIE “Input” command: accepts input (keyboard) and stores it in register IN, then transfers the data to AC.
-



```
Input
Output
Halt
```

Output log: Decimal

6

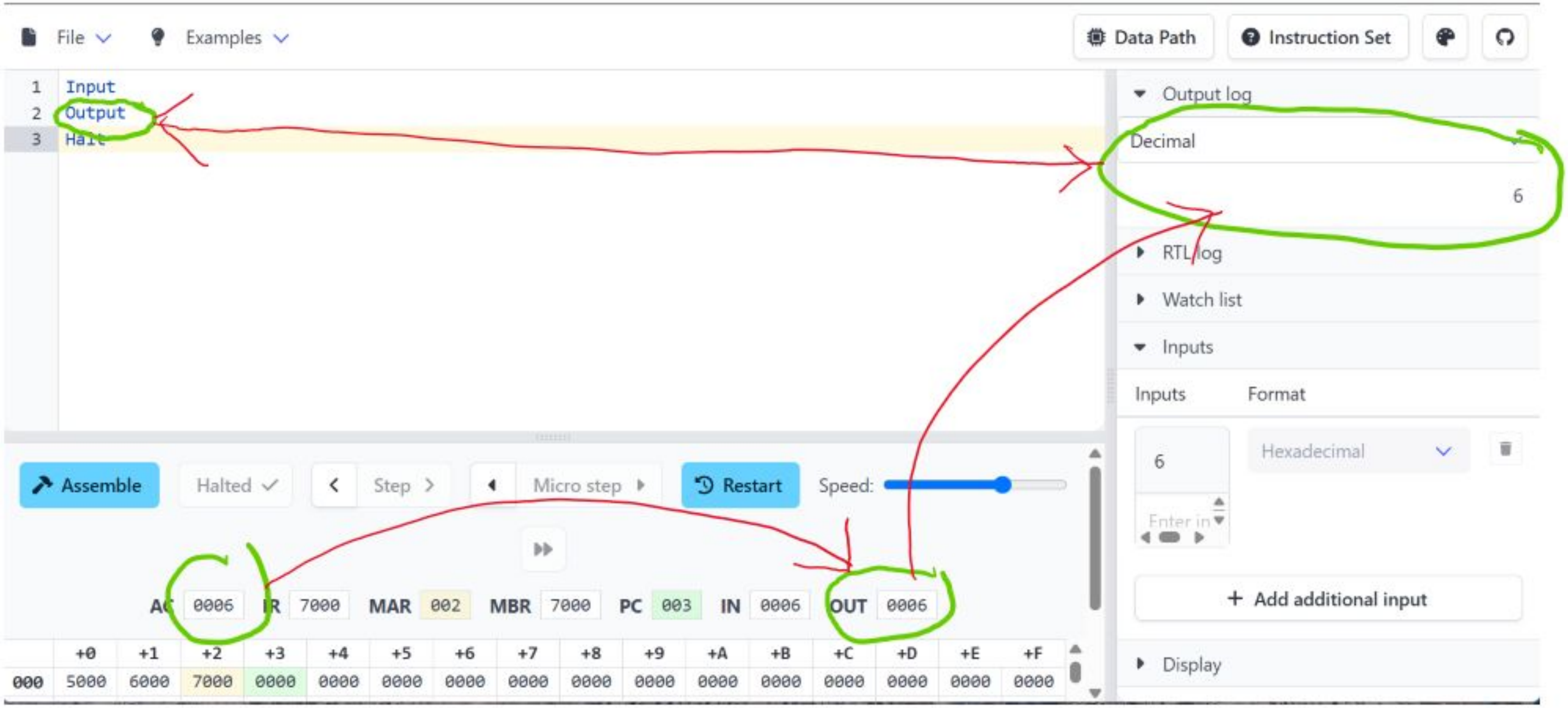
Inputs: 6 | Format: Hexadecimal

AC 0006 IR 7000 MAR 002 MBR 7000 PC 003 IN 0006 OUT 0006

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
000	5000	6000	7000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

CPU and I/O Interactions:
Port-mapped I/O

- MARIE "Output" command: copies data from AC and stores it in the output register OUT, which is linked to Output log i.e. text display.



```
Input
Output
Halt
```

Output log: Decimal

6

Inputs: 6 Format: Hexadecimal

AC 0006 IR 7000 MAR 002 MBR 7000 PC 003 IN 0006 OUT 0006

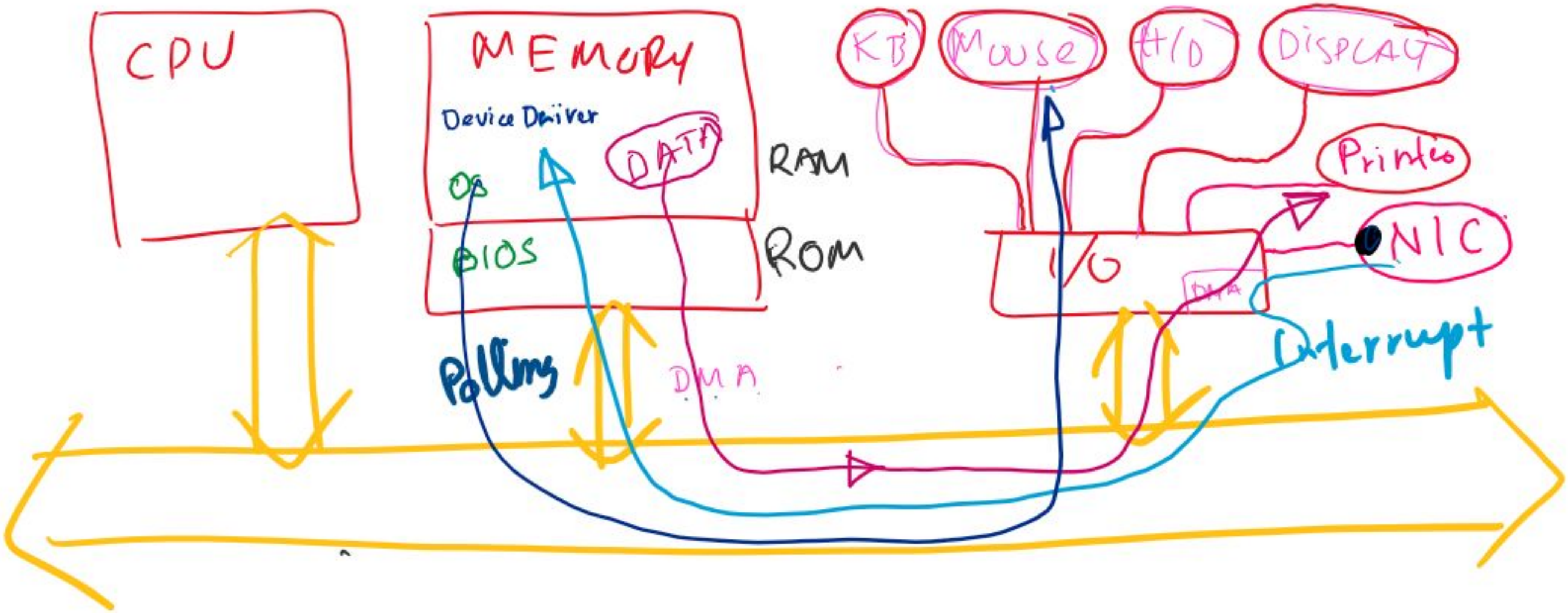
	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
000	5000	6000	7000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

Determine when to perform I/O?

Determine when to perform I/O?

(i) Programmed I/O (or Polling I/O)
(iii) Interrupt-Driven I/O
(i) Direct Memory Access (DMA)

Determine when to perform I/O?



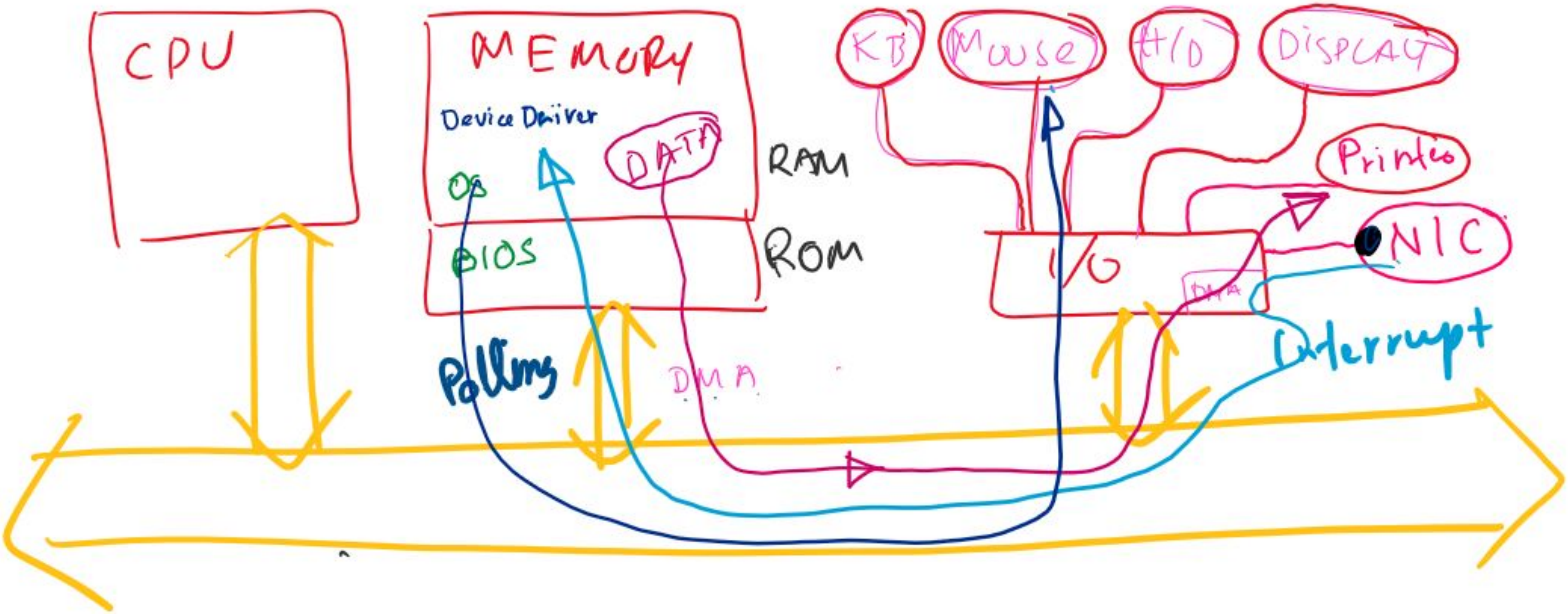
Determine when to perform I/O?
Programmed I/O (or Polling I/O)

Determine when to perform I/O?

Programmed I/O (or Polling I/O)

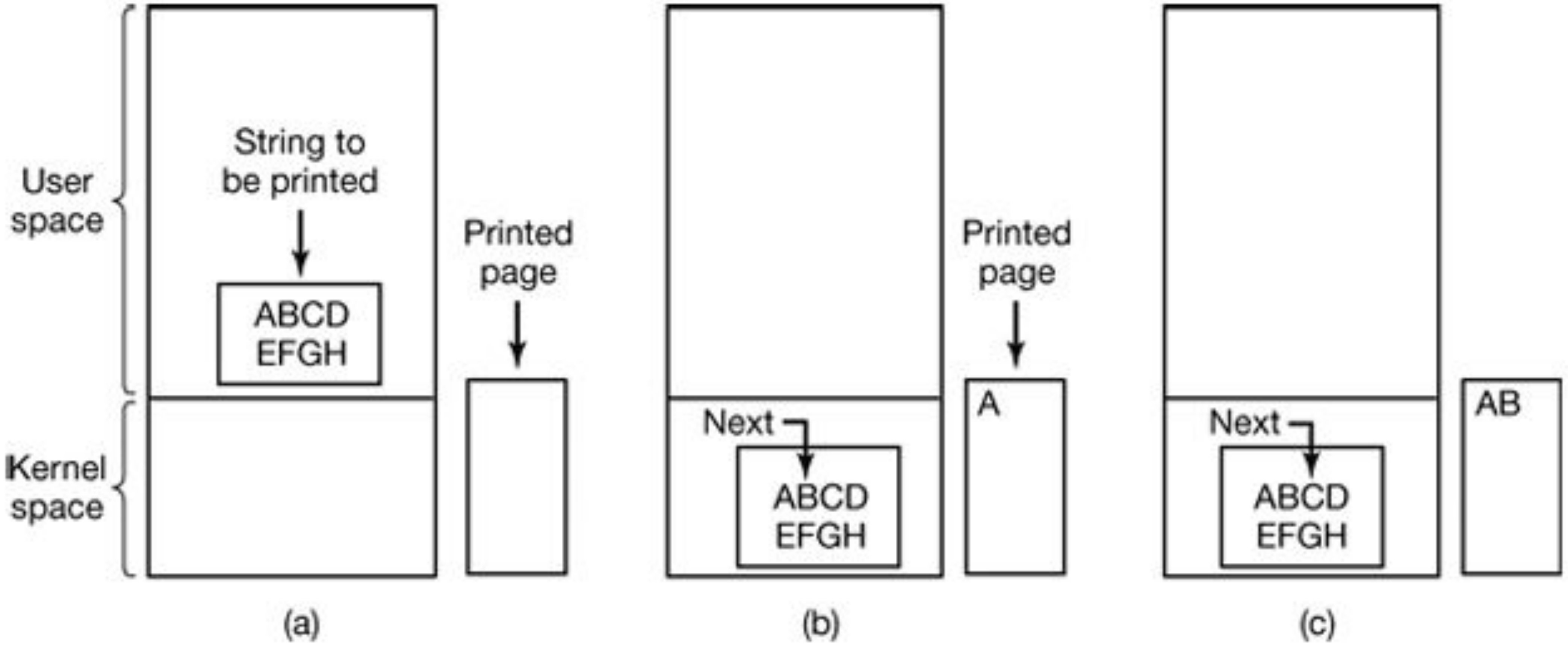
Determine when to perform I/O? Programmed I/O (or Polling I/O)

- Let the CPU do all the work. Simple.
- CPU is full time busy until all the I/O is done.



Determine when to perform I/O? Programmed I/O (or Polling I/O)

- An example: transfer string “ABCDEFGH” to a printer.



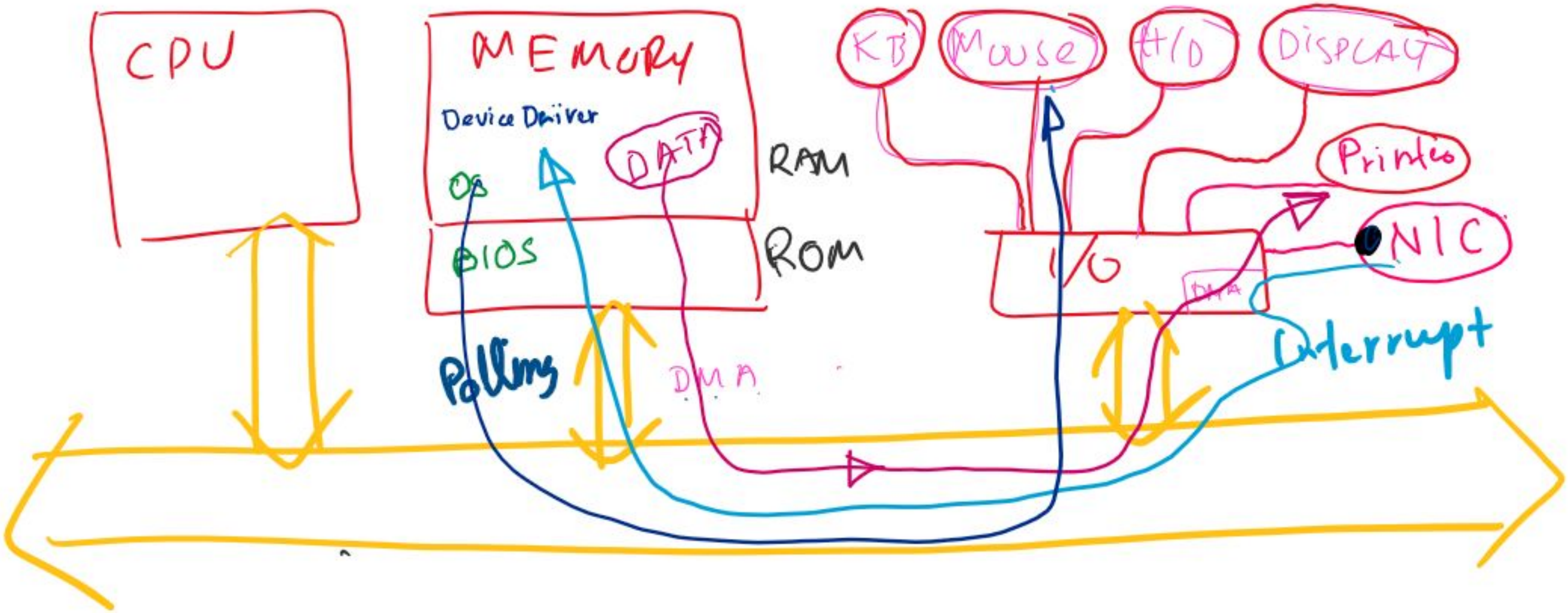
Determine when to perform I/O?
Interrupt Driven I/O

Determine when to perform I/O?

Interrupt Driven I/O

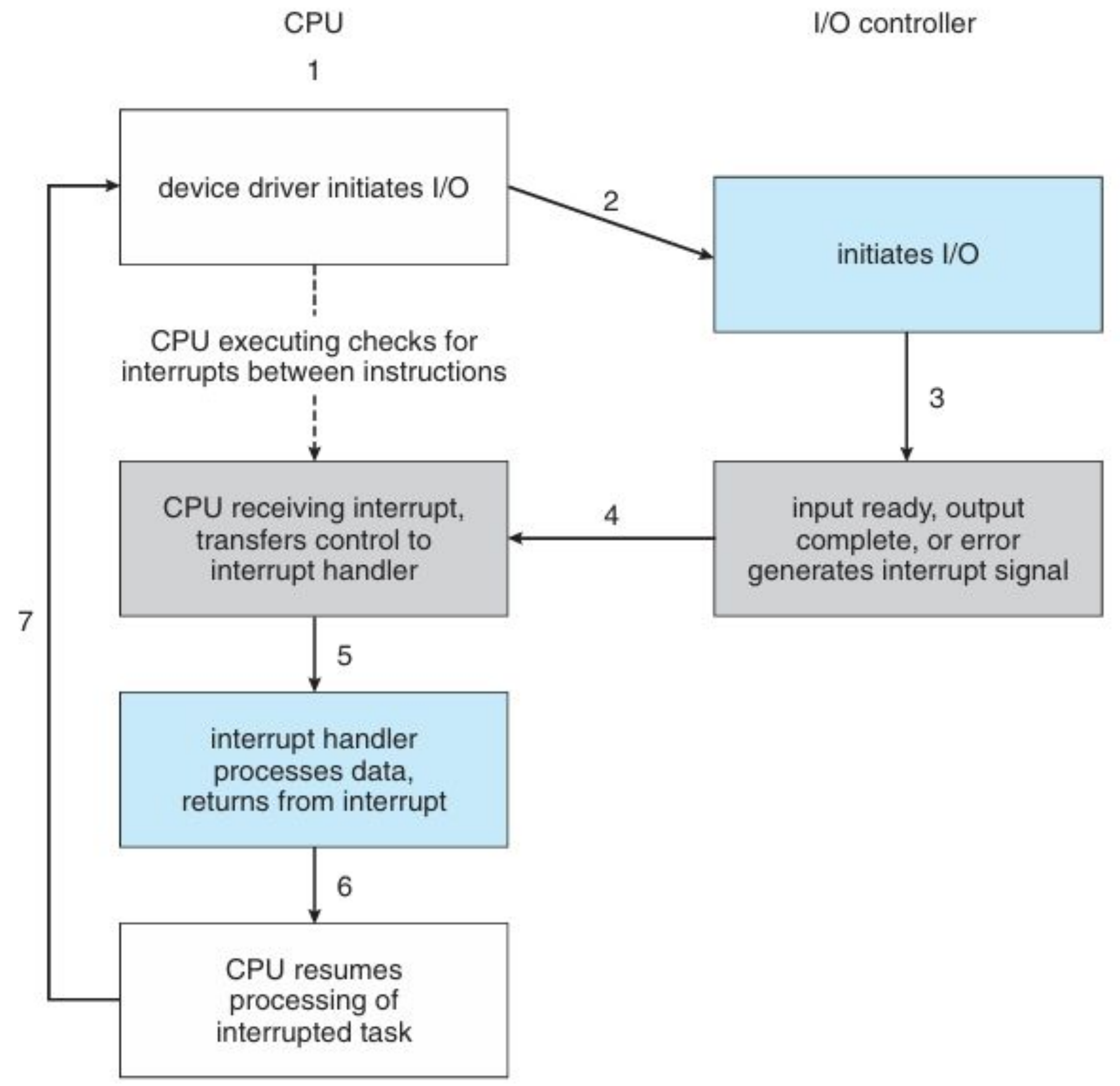
Determine when to perform I/O?
Interrupt Driven I/O

- Allow the CPU do something else the I/O is getting ready.
- Invokes the OS (scheduler) to allow other processes to use CPU.



Determine when to perform I/O?
Interrupt Driven I/O

- The OS (scheduler) switches the CPU between the print process and other processes.


CPU
I/O controller
1
device driver initiates I/O
2
initiates I/O
CPU executing checks for interrupts between instructions
3
CPU receiving interrupt, transfers control to interrupt handler
4
input ready, output complete, or error generates interrupt signal
5
interrupt handler processes data, returns from interrupt
6
CPU resumes processing of interrupted task
7

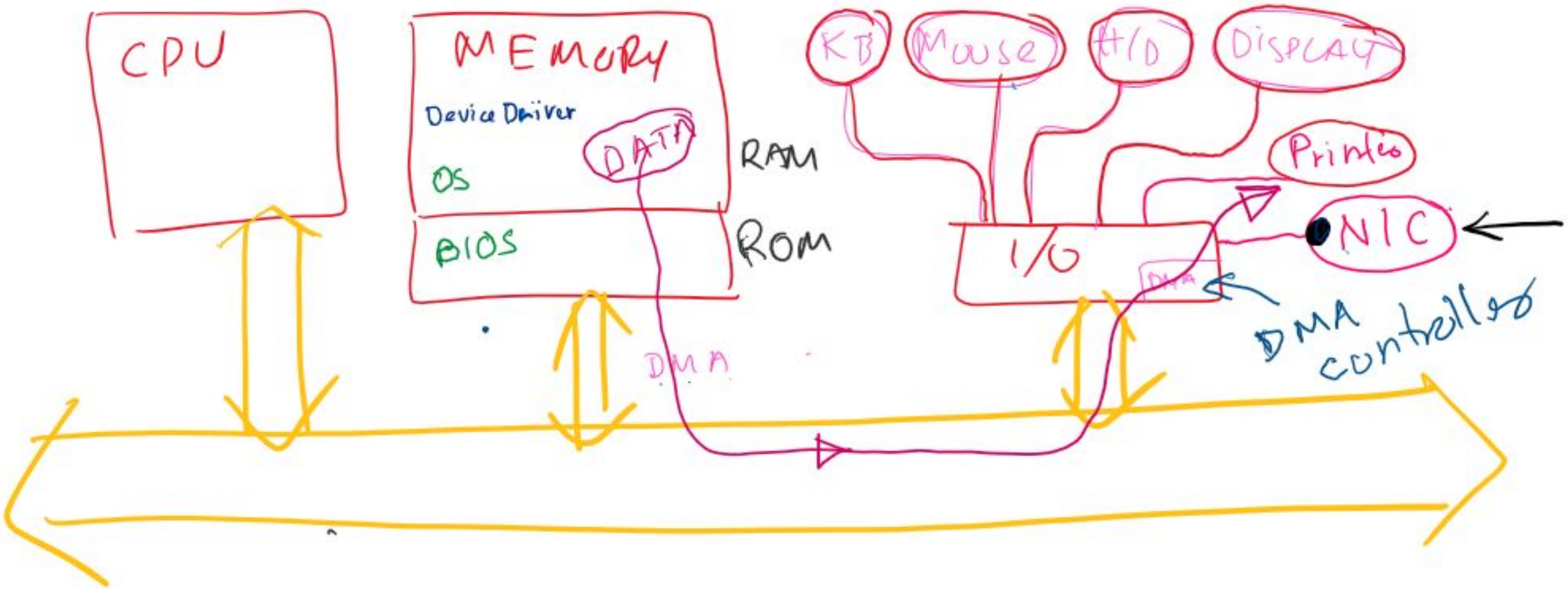
Determine when to perform I/O?
I/O Using Direct Memory Access (DMA)

Determine when to perform I/O?

I/O Using Direct Memory Access (DMA)

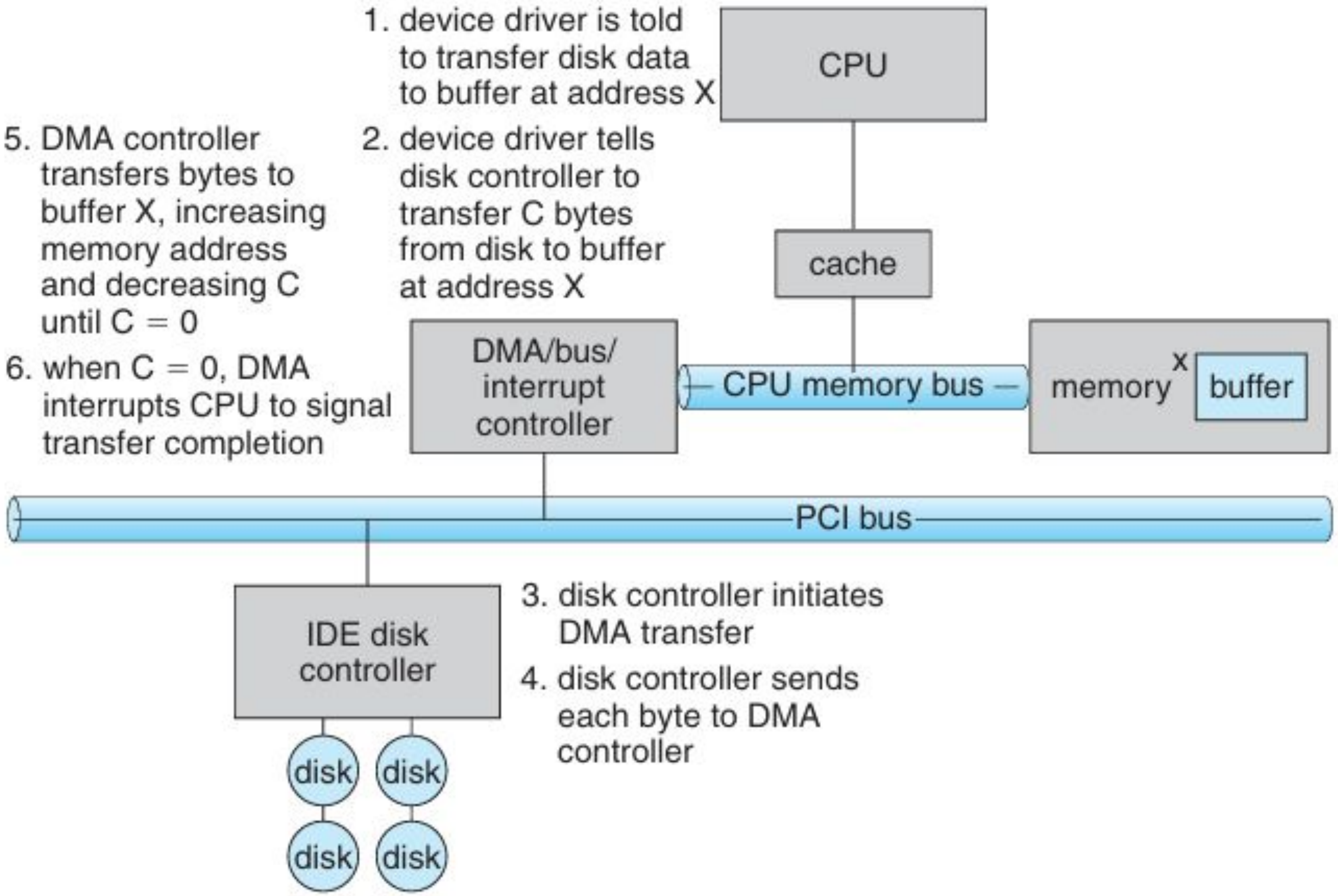
Determine when to perform I/O?
I/O Using Direct Memory Access (DMA)

- A programmed I/O with the DMA controller doing all the work (not CPU).
- Reduces the number of interrupts.



Determine when to perform I/O?
I/O Using Direct Memory Access (DMA)

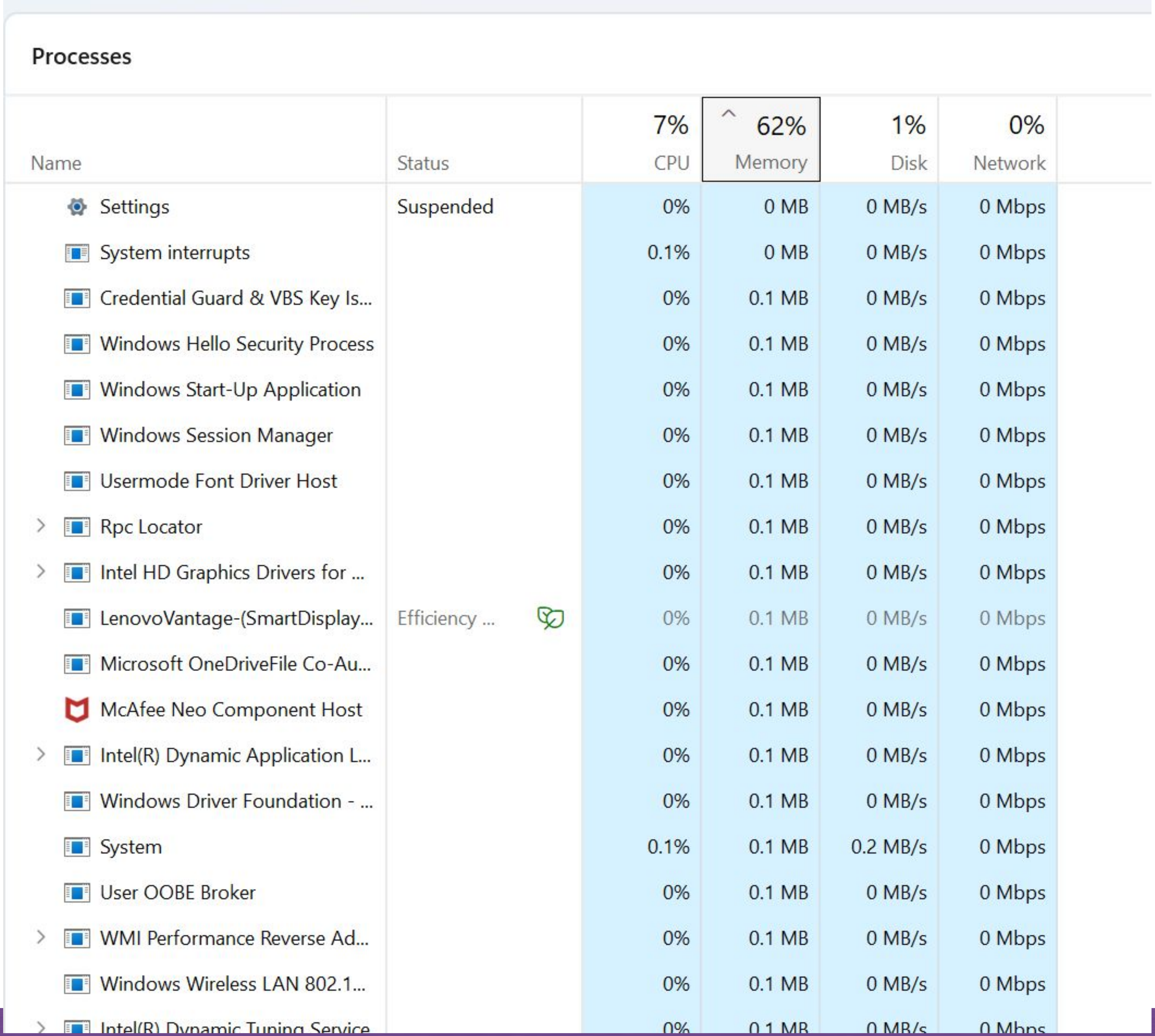
- DMA controller is managing the transfer.
- DMA sends characters to printer without CPU being used.


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

Activity B:

Process Management - Round Robin Scheduling

Process Management
In a multi-processing environment

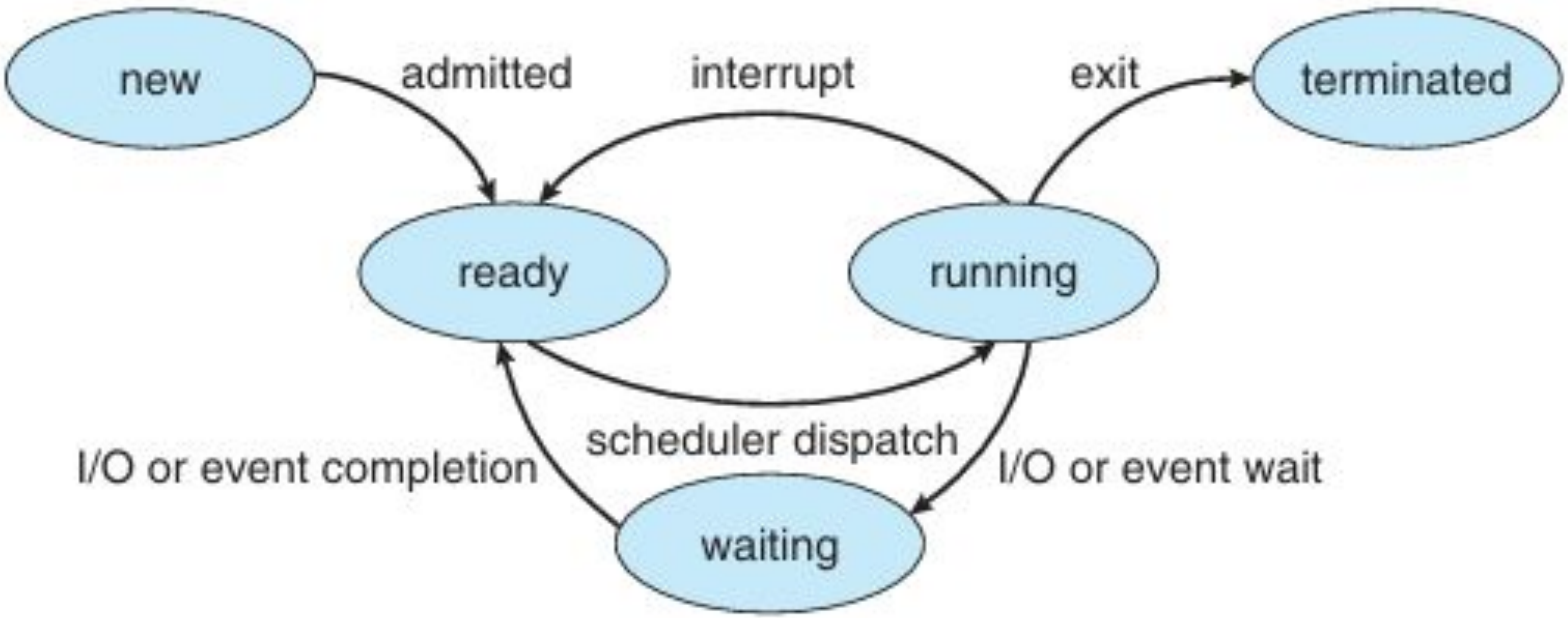


Processes

		7%	^ 62%	1%	0%
Name	Status	CPU	Memory	Disk	Network
Settings	Suspended	0%	0 MB	0 MB/s	0 Mbps
System interrupts		0.1%	0 MB	0 MB/s	0 Mbps
Credential Guard & VBS Key Is...		0%	0.1 MB	0 MB/s	0 Mbps
Windows Hello Security Process		0%	0.1 MB	0 MB/s	0 Mbps
Windows Start-Up Application		0%	0.1 MB	0 MB/s	0 Mbps
Windows Session Manager		0%	0.1 MB	0 MB/s	0 Mbps
Usermode Font Driver Host		0%	0.1 MB	0 MB/s	0 Mbps
> Rpc Locator		0%	0.1 MB	0 MB/s	0 Mbps
> Intel HD Graphics Drivers for ...		0%	0.1 MB	0 MB/s	0 Mbps
LenovoVantage-(SmartDisplay...	Efficiency ...	0%	0.1 MB	0 MB/s	0 Mbps
Microsoft OneDriveFile Co-Au...		0%	0.1 MB	0 MB/s	0 Mbps
McAfee Neo Component Host		0%	0.1 MB	0 MB/s	0 Mbps
> Intel(R) Dynamic Application L...		0%	0.1 MB	0 MB/s	0 Mbps
Windows Driver Foundation - ...		0%	0.1 MB	0 MB/s	0 Mbps
System		0.1%	0.1 MB	0.2 MB/s	0 Mbps
User OOBE Broker		0%	0.1 MB	0 MB/s	0 Mbps
> WMI Performance Reverse Ad...		0%	0.1 MB	0 MB/s	0 Mbps
Windows Wireless LAN 802.1...		0%	0.1 MB	0 MB/s	0 Mbps
> Intel(R) Dynamic Tuning Service		0%	0.1 MB	0 MB/s	0 Mbps

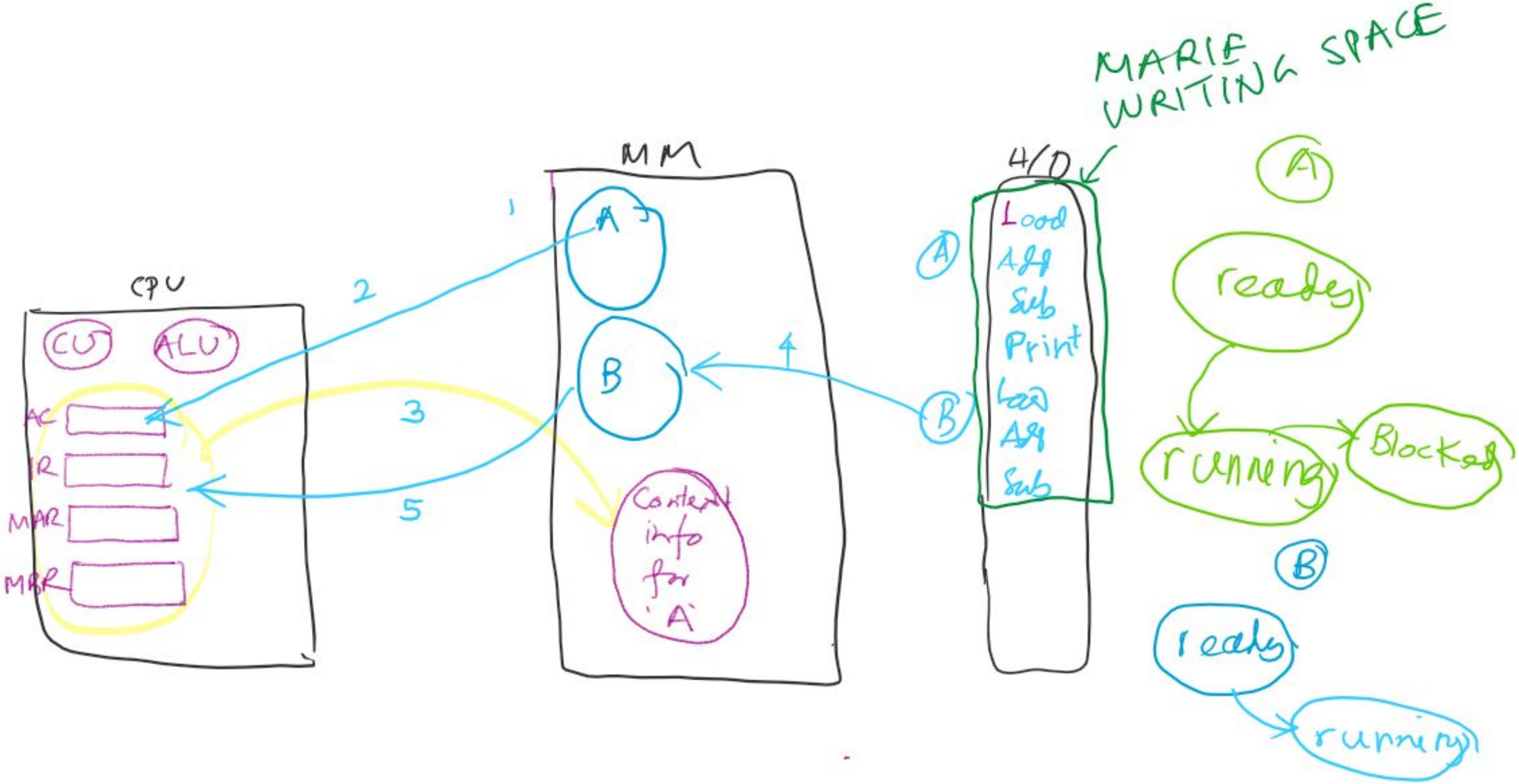
Process Management
In a multi-processing environment

- Each process move through different stages.
- Ready -> Running -> Waiting->Ready



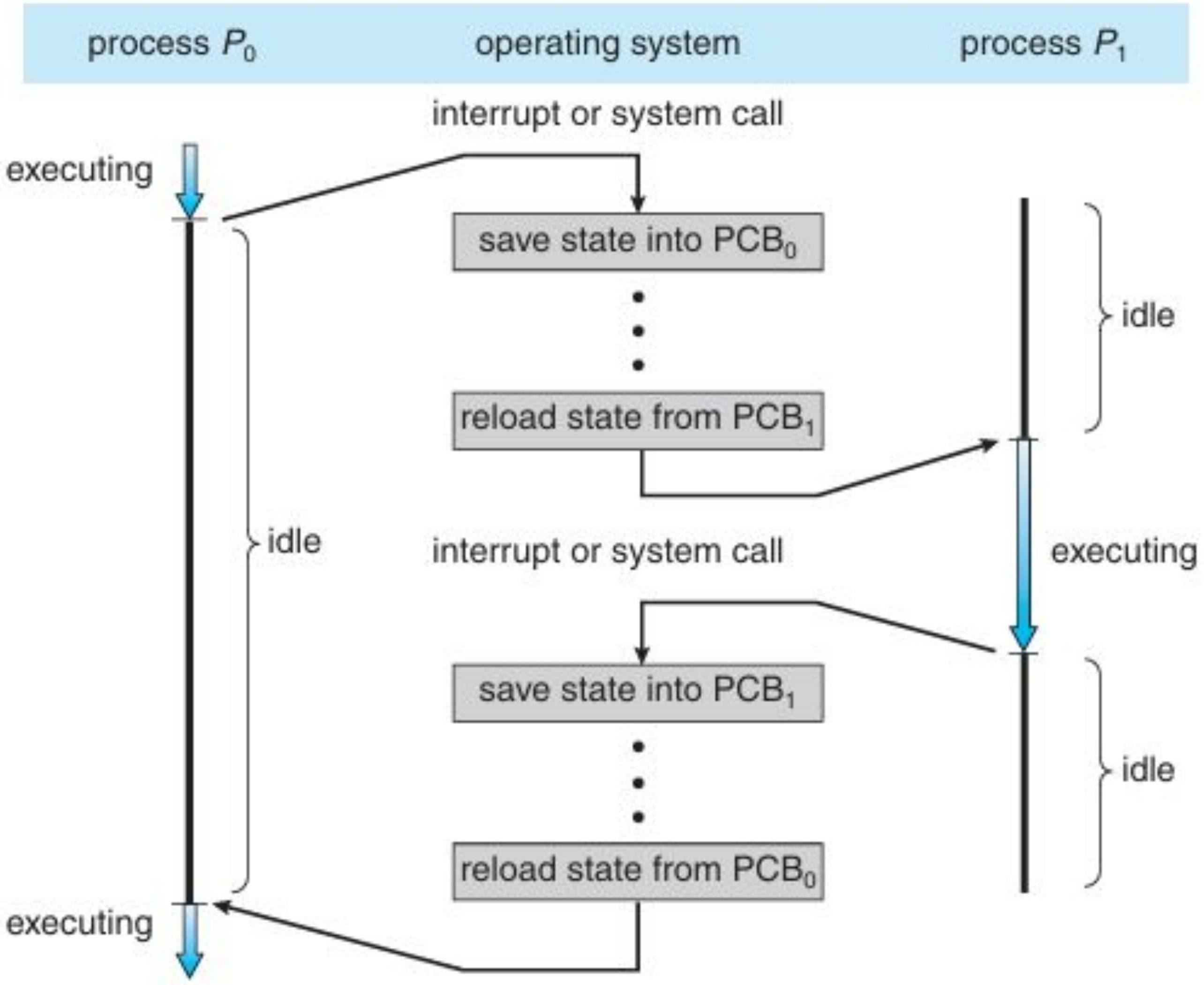
Process Management
In a multi-processing environment

- Process switching.
- Context Switching.



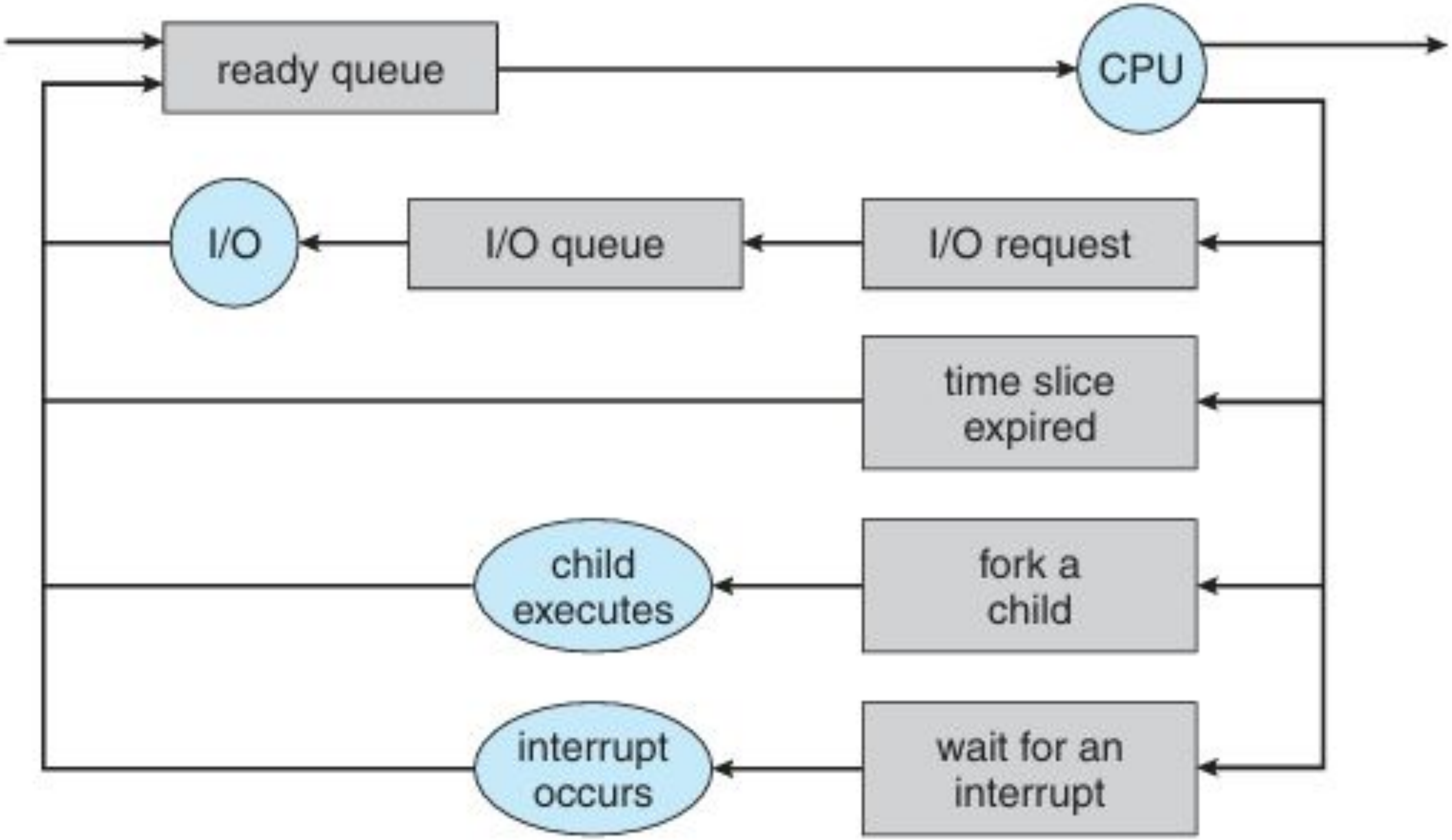
Process Management
In a multi-processing environment

- Use of Process Control Block (PCB).



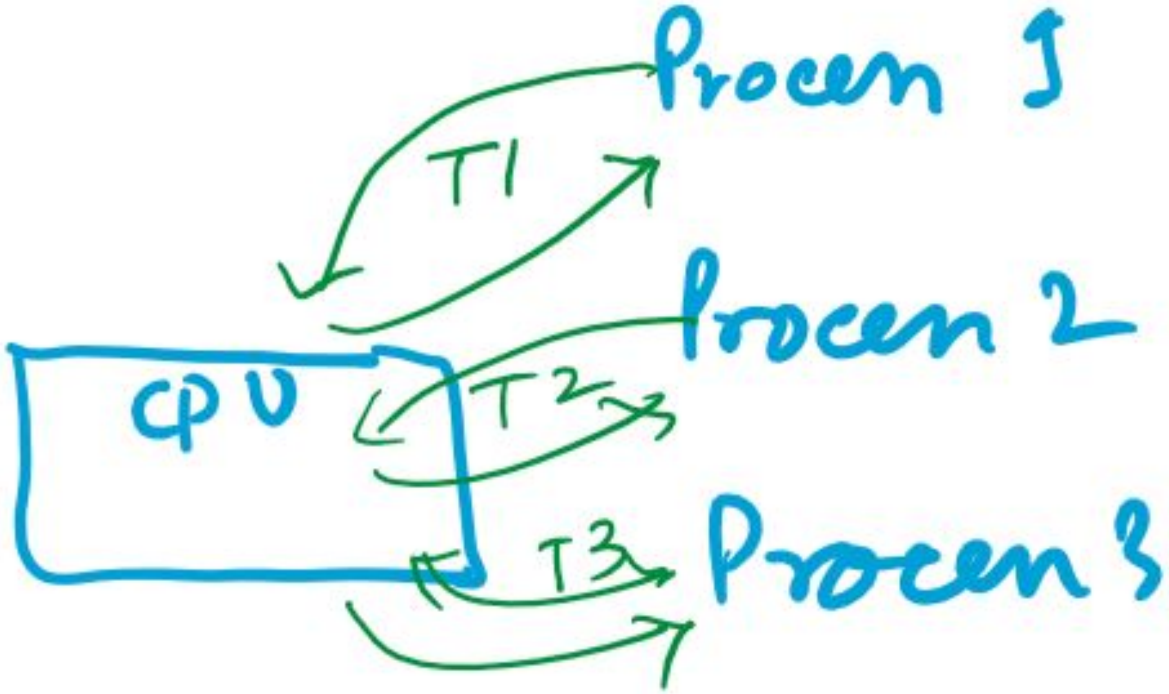
Process Management
In a multi-processing environment

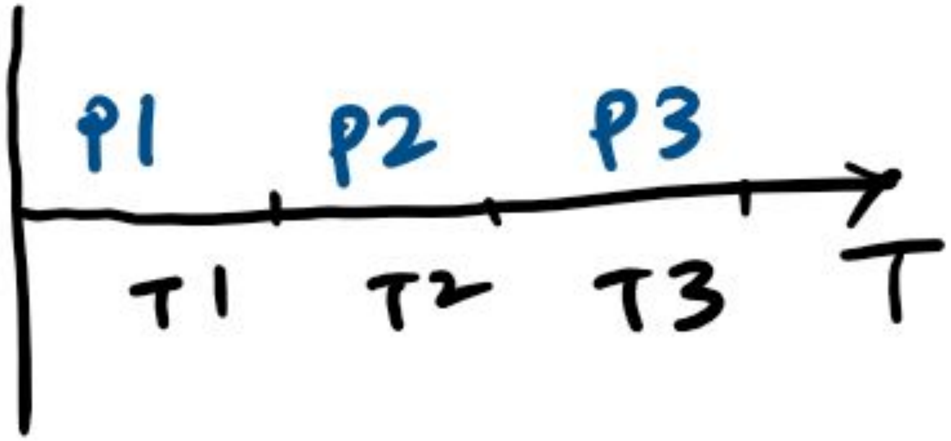
- Queuing to help CPU tp pick a process for the next CPU time.



Process Management
In a multi-processing environment

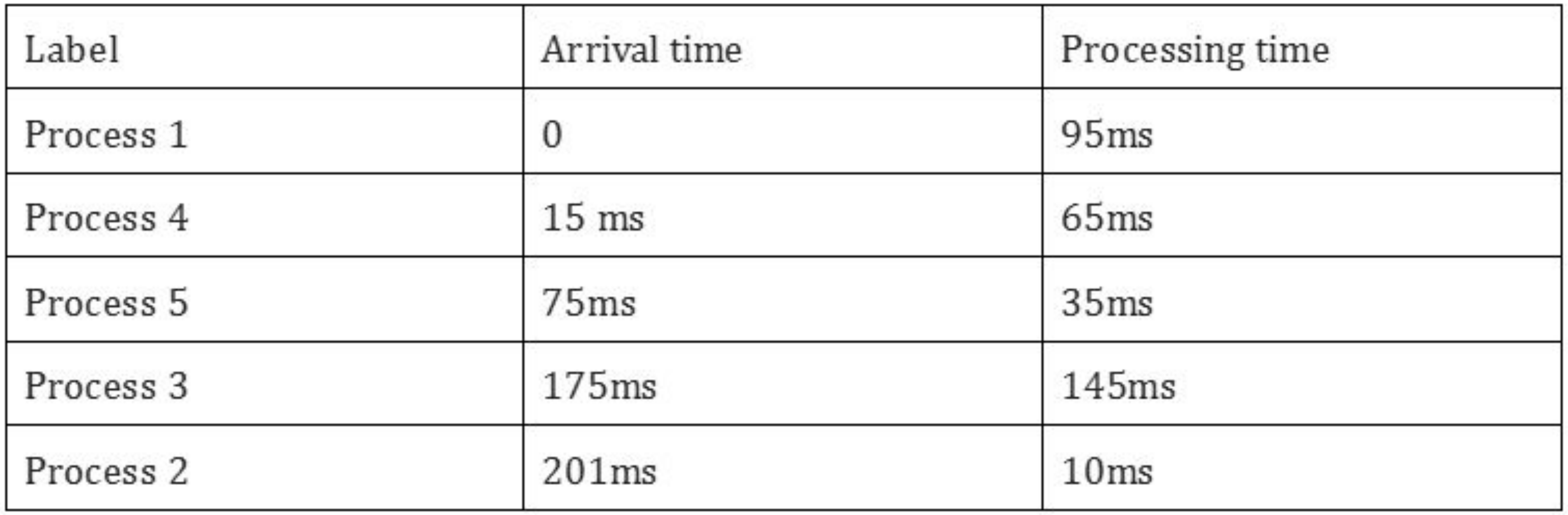
- CPU time slices and allocated processes.





Process Management
Round-Robin Scheduling

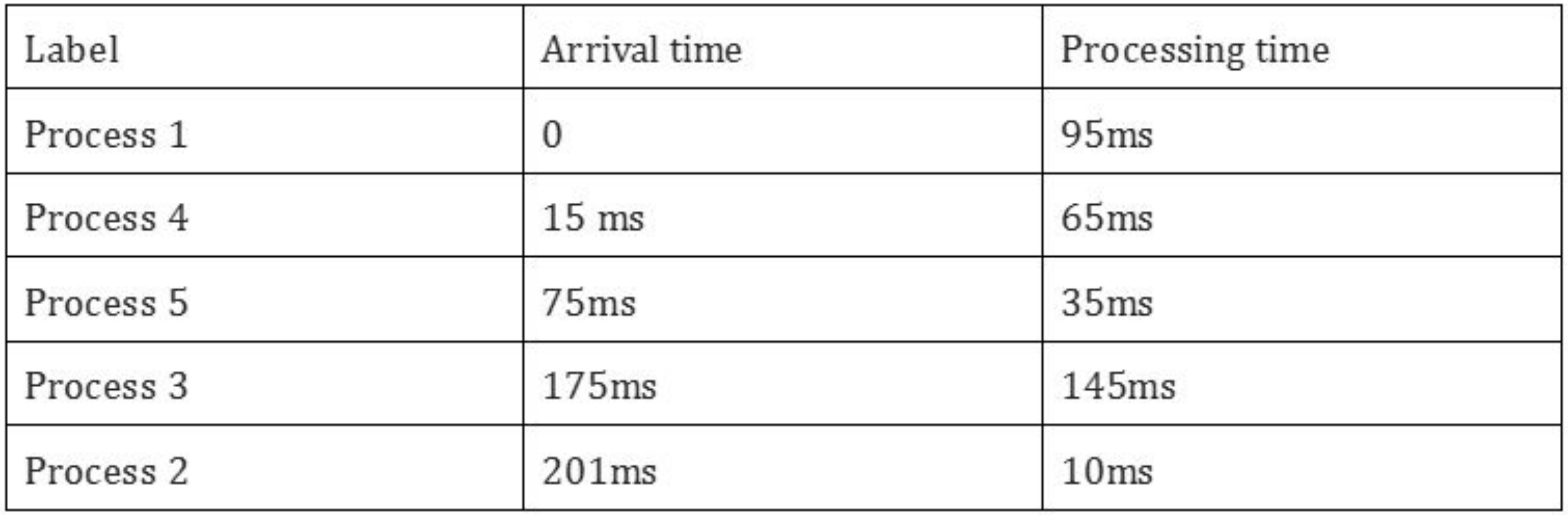
- Round Robin scheduler
- Time quantum is 50 milliseconds.
- Given: (i) start time and (ii) time to finish processes are given
- Unused time is given to the next process immediately.
- Ignore context-switching time.



Label	Arrival time	Processing time
Process 1	0	95ms
Process 4	15 ms	65ms
Process 5	75ms	35ms
Process 3	175ms	145ms
Process 2	201ms	10ms

Process Management
Round-Robin Scheduling

- Answer the following questions:
  - Which process starts first?
  - Which process finishes last? When does it finish?
  - Which process finishes second last?
  - Show all the steps of the Round Robin algorithm using a queue.



Label	Arrival time	Processing time
Process 1	0	95ms
Process 4	15 ms	65ms
Process 5	75ms	35ms
Process 3	175ms	145ms
Process 2	201ms	10ms

Process Management
Round-Robin Scheduling

- Answer the following questions:
  - Which process starts first?
  - Which process finishes last? When does it finish?
  - Which process finishes second last?
  - Show all the steps of the Round Robin algorithm using a queue.

Time Slice	Process having CPU	Remaining Time (end of time slice)	Processes in the queue (added at time)
1 (0-50)			
2 (50-100)			

Follow the Workshop Acitivties Steps & Answer the Questions

End of the Workshop Tasks

Have a nice week ahead.

See You all

Next Week