INPUT-OUTPUT ORGANIZATION

- Peripheral Devices
- Input-Output Interface
- Asynchronous Data Transfer
- Modes of Transfer
- Direct Memory Access

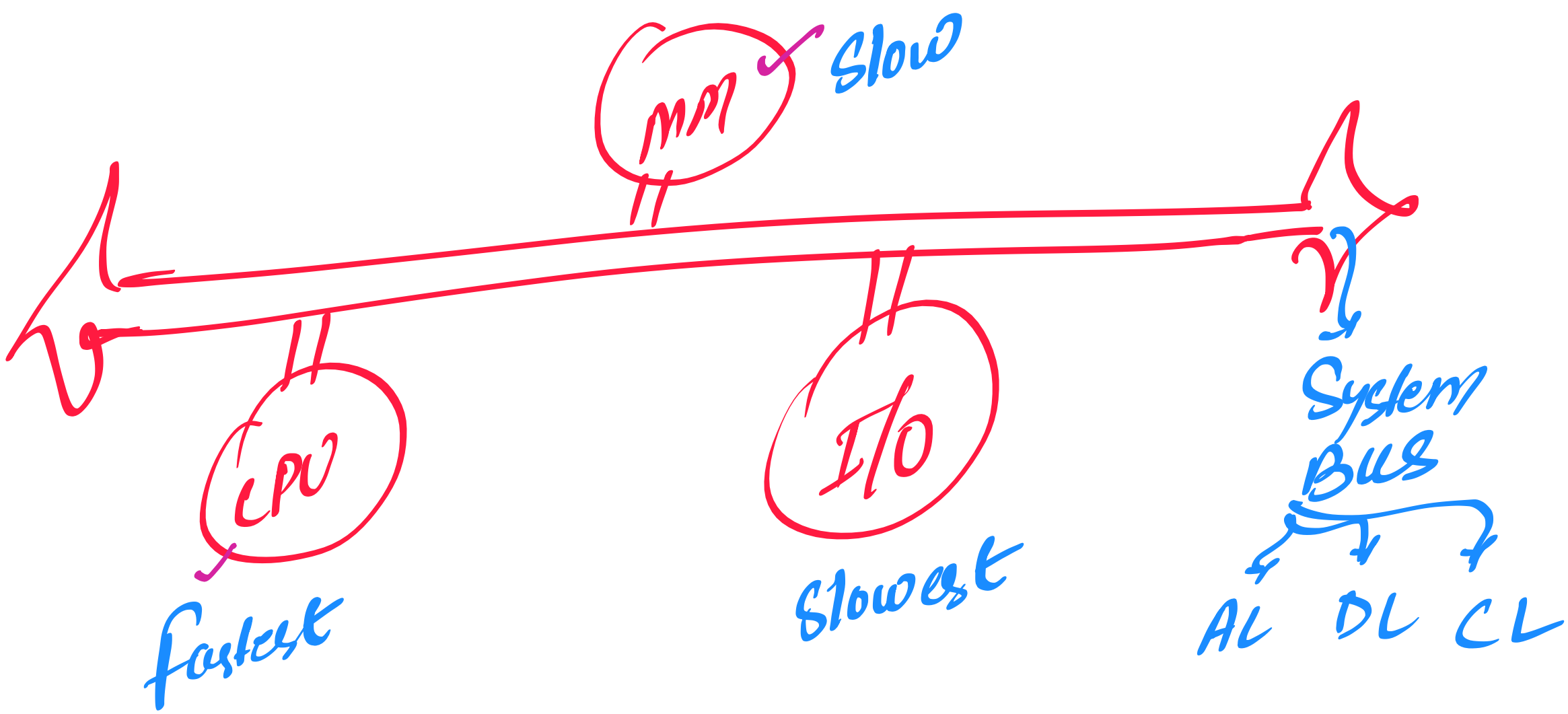
EST

Total → 7 Questions ⇒ 7 marks each

Attempt 6 Question

1. Ques- MST
2. Pipeline, Hazards
3. Addressing modes, Instr" format
4. Hardwired CU
5. Cache memory : Address format, Block Replac.
6. Virtual Mem: Page fault, Page Replace Algo Algo

7. IO → Programmed I/O, DMA → Numerical


MM
Slow
CPU
Fastest
I/O
Slowest
System
Bus
AL
DL
CL

PERIPHERAL DEVICES

Input Devices

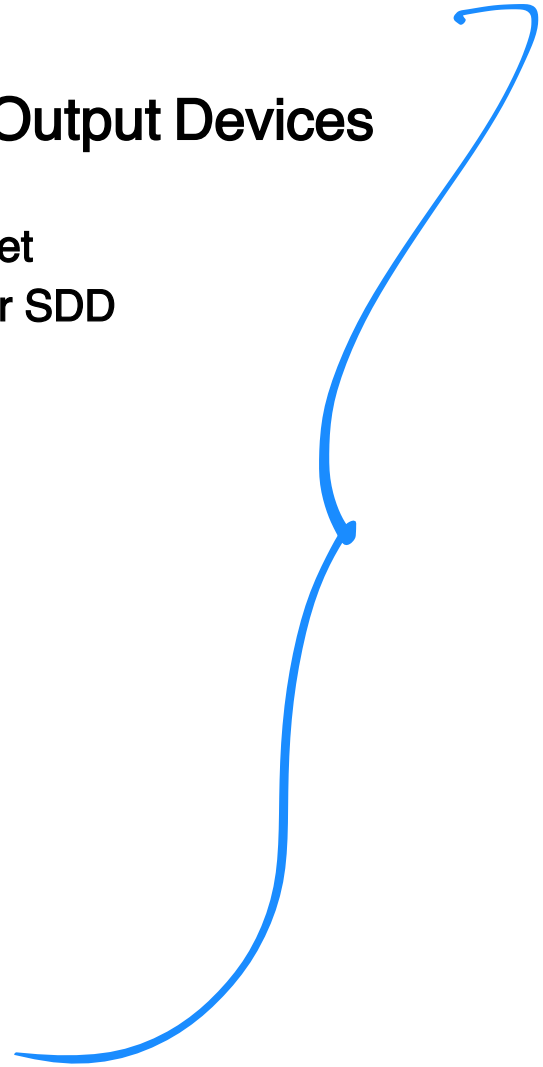
- Keyboard
- Optical input devices
  - Card Reader
  - Paper Tape Reader
  - Bar code reader
  - Digitizer
  - Optical Mark Reader
- Magnetic Input Devices
  - Magnetic Stripe Reader
- Screen Input Devices
  - Touch Screen
  - Light Pen
  - Mouse
- Analog Input Devices

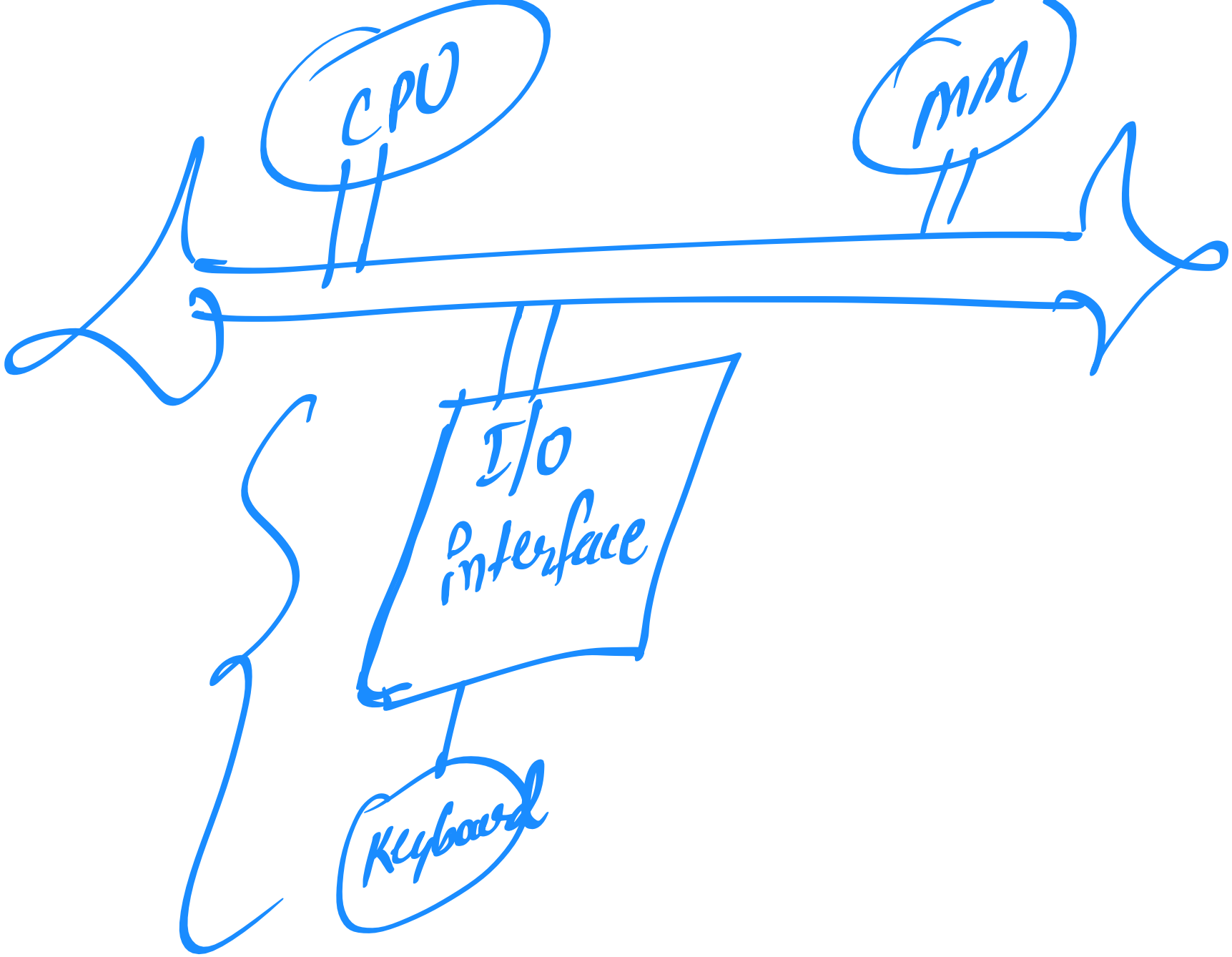
Output Devices

- Card Puncher, Paper Tape Puncher
- CRT
- Printer (Impact, Ink Jet, Laser, Dot Matrix)
- Plotter
- Analog
- Voice

Input/Output Devices

- Headset
- HDD or SDD

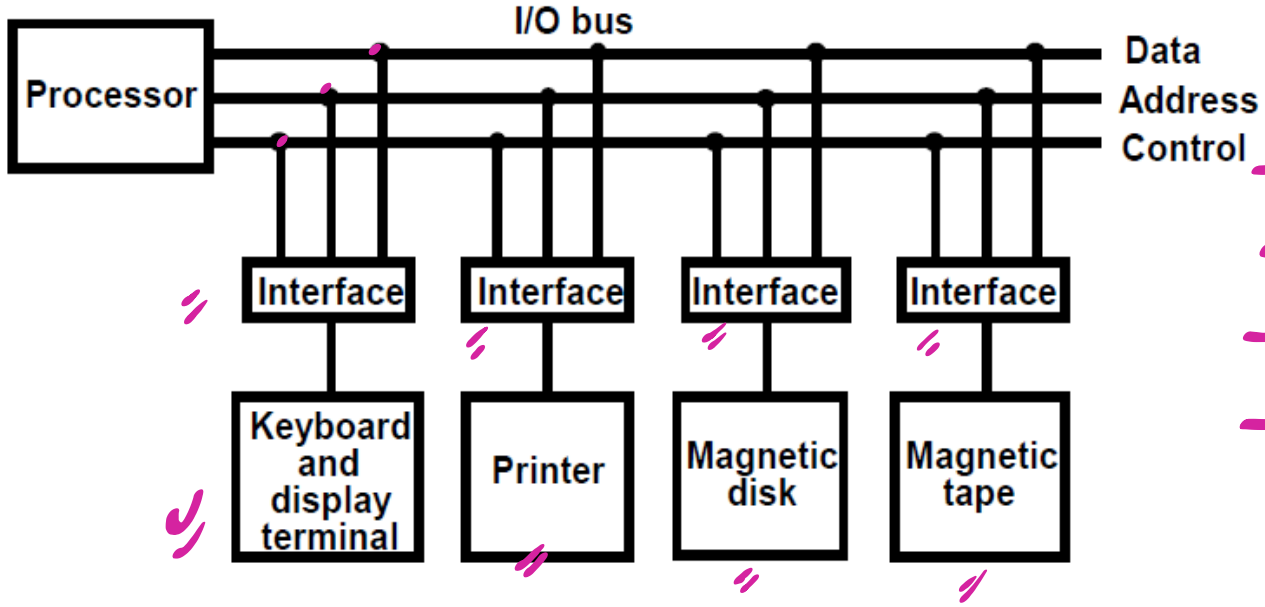



CPU
MM
I/O
Interface
Keyboard

INPUT/OUTPUT INTERFACE

- Provides a method for transferring information between internal storage (such as memory and CPU registers) and external I/O devices
- Resolves the *differences* between the computer and peripheral devices
  - Peripherals - Electromechanical Devices
  - CPU or Memory - Electronic Device
  - Data Transfer Rate
    - » Peripherals - Usually slower
    - » CPU or Memory - Usually faster than peripherals
      - Some kinds of Synchronization mechanism may be needed
  - Unit of Information
    - » Peripherals – Byte, Block, …
    - » CPU or Memory – Word
  - Data representations may differ

I/O BUS AND INTERFACE MODULES

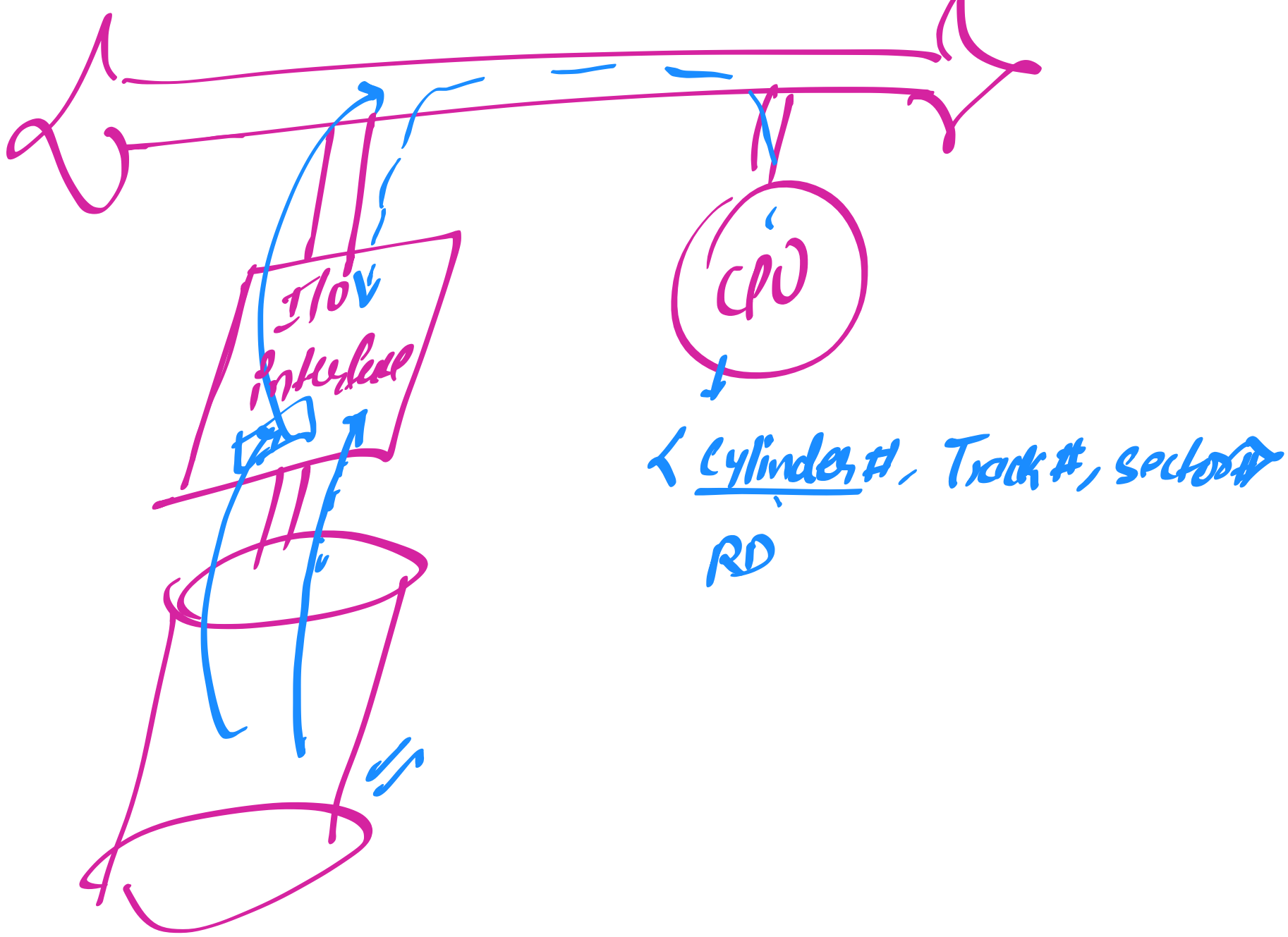


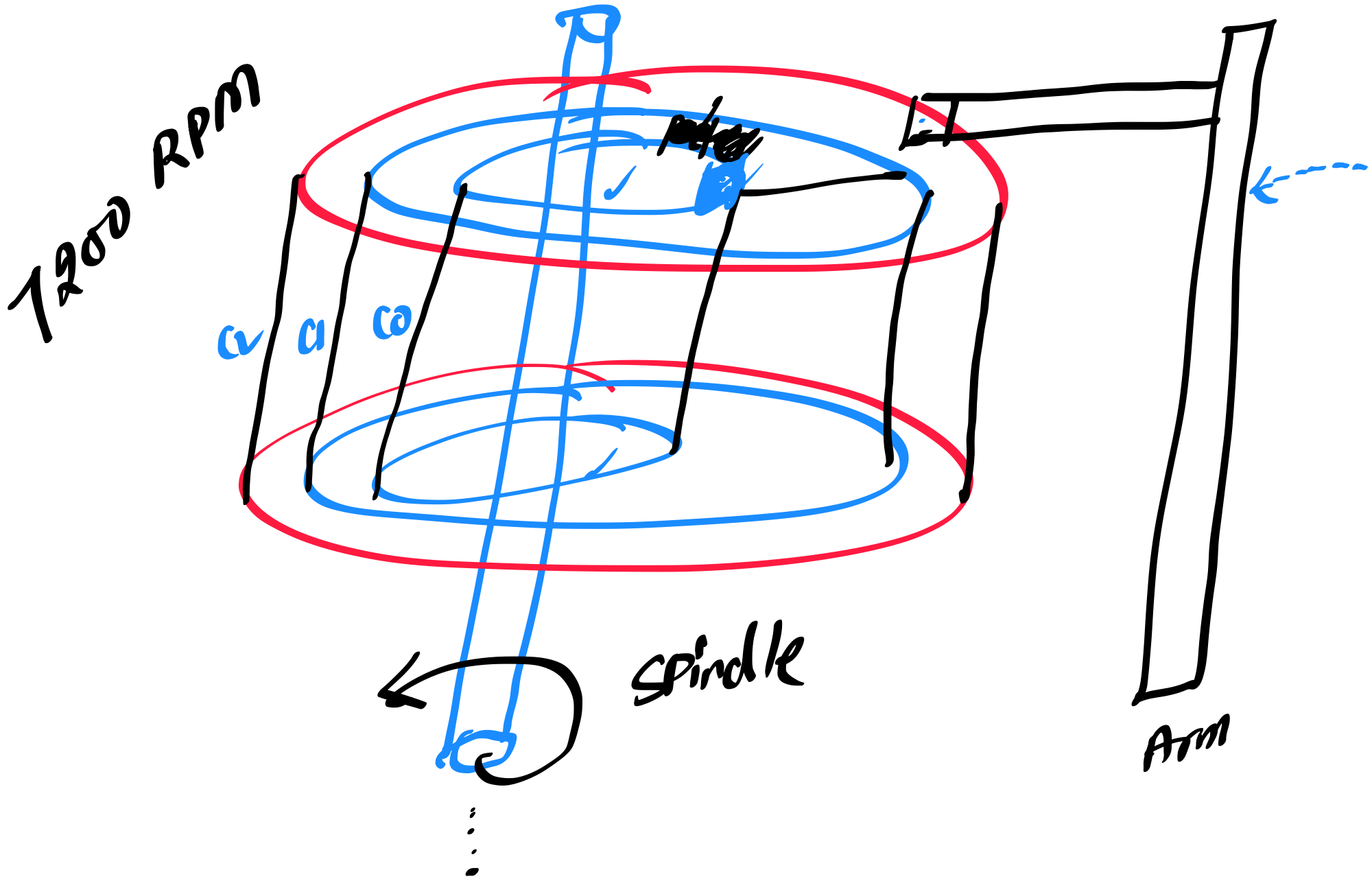
- Each peripheral has an interface module associated with it
- Interface
  - Decodes the device address (device code)
  - Decodes the commands (operation)
  - Provides signals for the peripheral controller
  - Synchronizes the data flow and supervises the transfer rate between the peripheral and CPU or Memory

Typical I/O instruction

Op. code	Device address	Function code

(Command)


I/O
interface
CPU
< Cylinder #, Track #, Sector #
RD


1200 RPM
Spindle
Arm

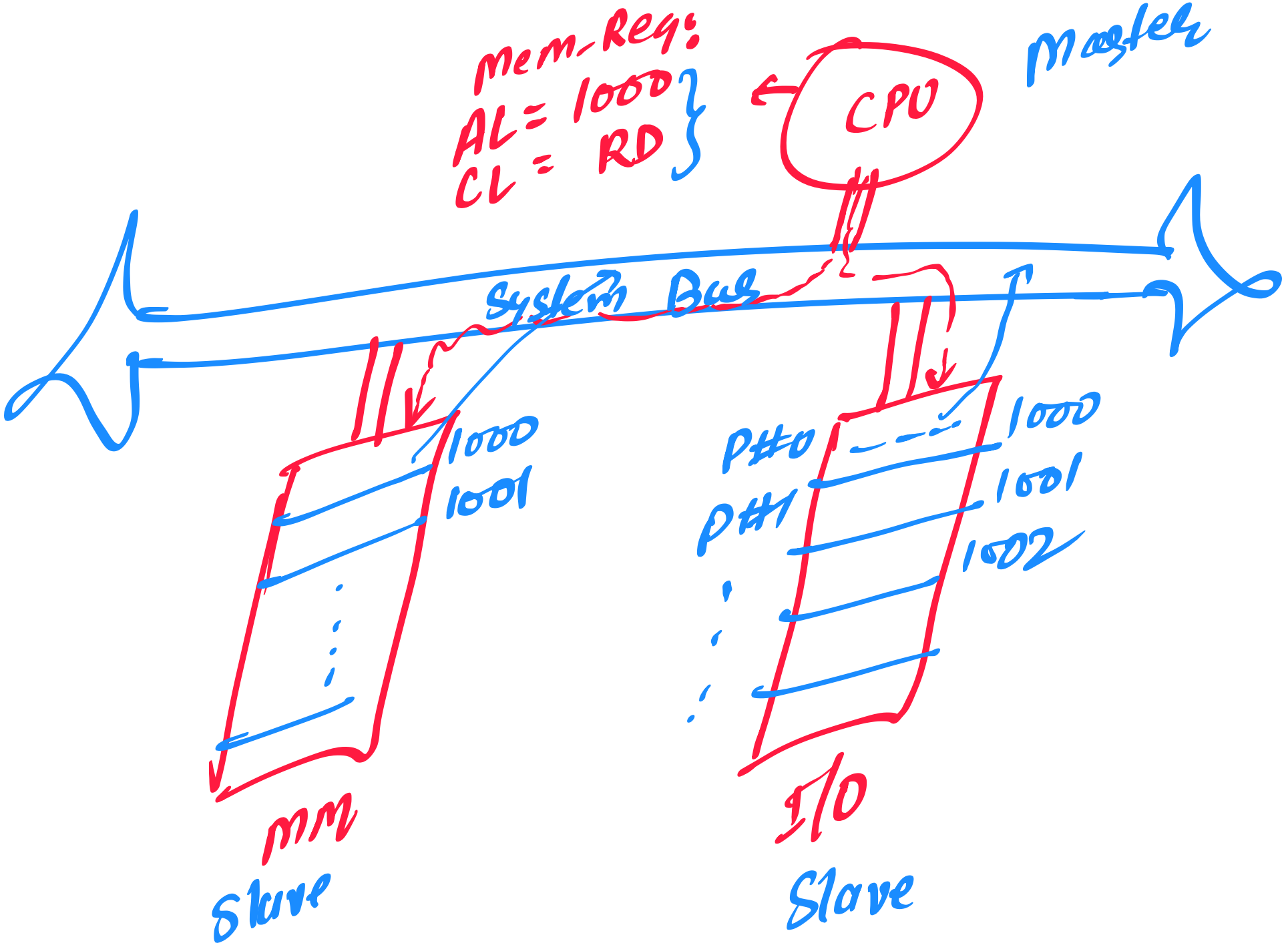
I/O BUS AND MEMORY BUS

Functions of Buses

- *MEMORY BUS* is for information transfers between CPU and the MM
- *I/O BUS* is for information transfers between CPU and I/O devices through their I/O interface

Physical Organizations

- Many computers use a common single bus system for both memory and I/O interface units
  - Use one common bus but separate control lines for each function
  - Use one common bus with common control lines for both functions
- Some computer systems use two separate buses,: **one to communicate with memory and the other with I/O interfaces.**


Mem_Req:
AL = 1000
CL = RD
CPU
Master
System Bus
1000
1001
P#0
P#1
1000
1001
1002
MM
Slave
I/O
Slave

0

I/O BUS AND MEMORY BUS

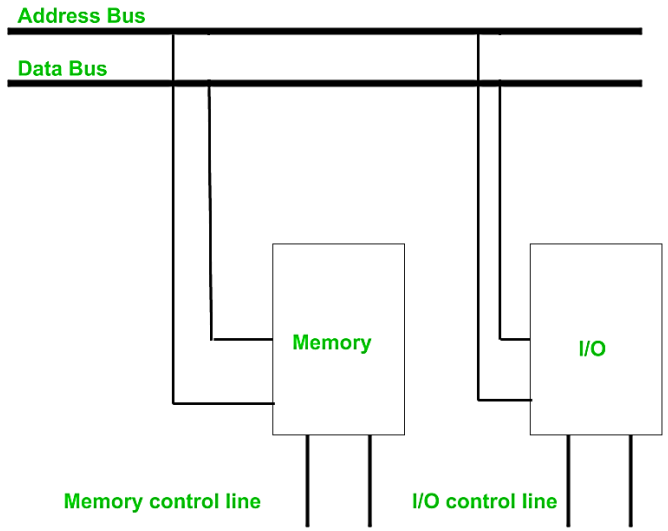
I/O Bus

- Communication between CPU and all interface units is via a common I/O Bus
- An interface connected to a peripheral device may have a number of *data registers*, a *control register*, and a *status register*
- A command is passed to the peripheral by sending to the appropriate interface register
- Function code and sense lines are not needed (Transfer of data, control, and status information is always via the common I/O Bus)

ISOLATED vs MEMORY MAPPED I/O

Isolated I/O

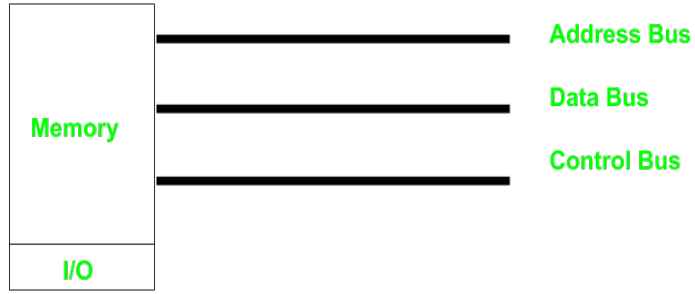
- Separate I/O read/write control lines in addition to memory read/write control lines
- Separate (isolated) memory and I/O address spaces
- Distinct input and output instructions



Isolated I/O

Memory-mapped I/O

- A single set of read/write control lines
  (no distinction between memory and I/O transfer)
- Memory and I/O addresses share the common address space
  -> reduces memory address range available
- No specific input or output instruction
  -> The same memory reference instructions can be used for I/O transfers
- Considerable flexibility in handling I/O operations

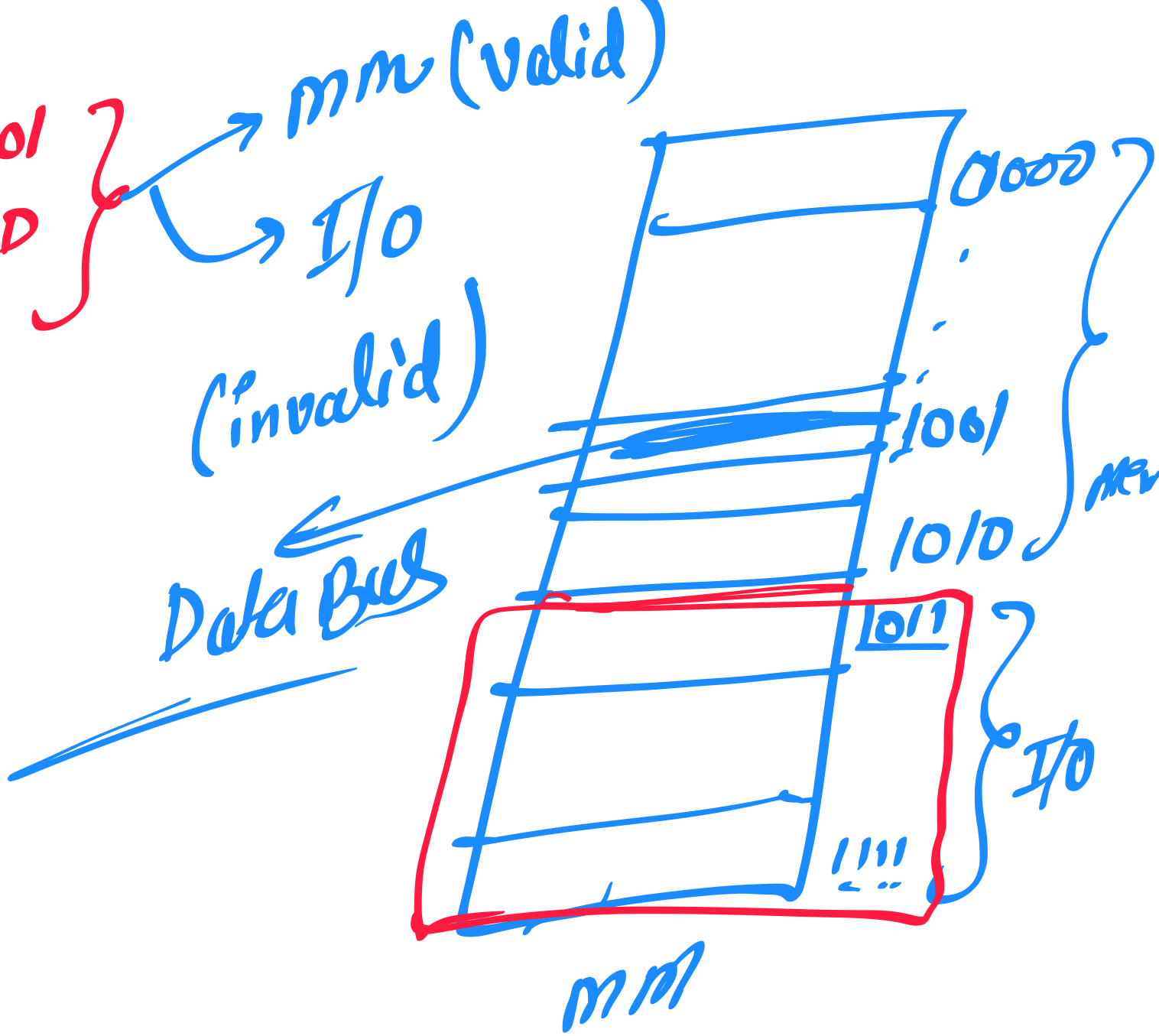


Memory-mapped I/O

CPU → Mem_Req: AL = 1001
CL = RD

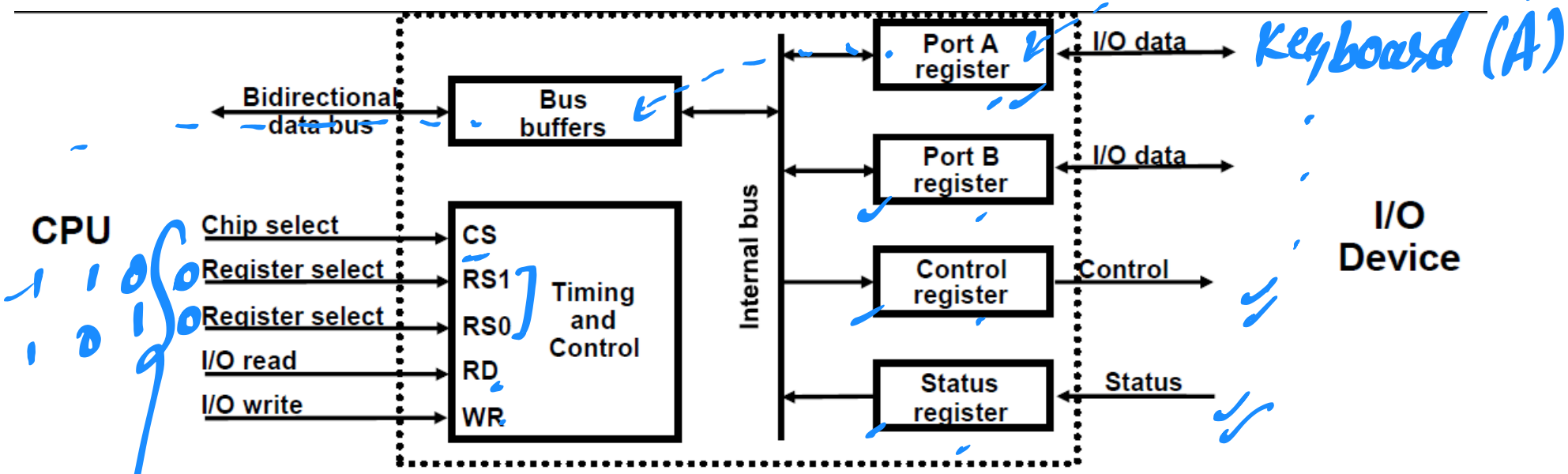
→ MM (valid)

→ I/O (invalid)



Pisado

I/O INTERFACE



CS	RS1	RS0	Register selected
0	x	x	None - data bus in high-impedence
1	0	0	Port A register
1	0	1	Port B register
1	1	0	Control register
1	1	1	Status register

I/O INTERFACE

Programmable Interface

- Information in each port can be assigned a meaning
  depending on the mode of operation of the I/O device
  → Port A = Data; Port B = Command; Port C = Status

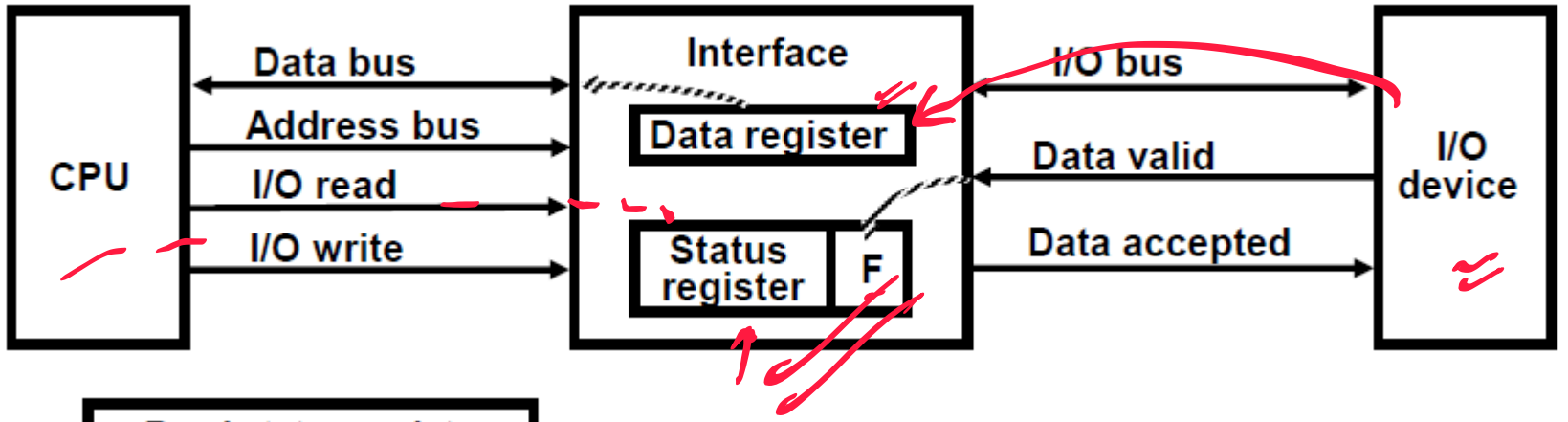
- **CPU initializes(loads) each port by transferring a byte to the Control Register**
  → Allows CPU can define the mode of operation of each port
  → ***Programmable Port***: By changing the bits in the control register, it possible to change the interface characteristics

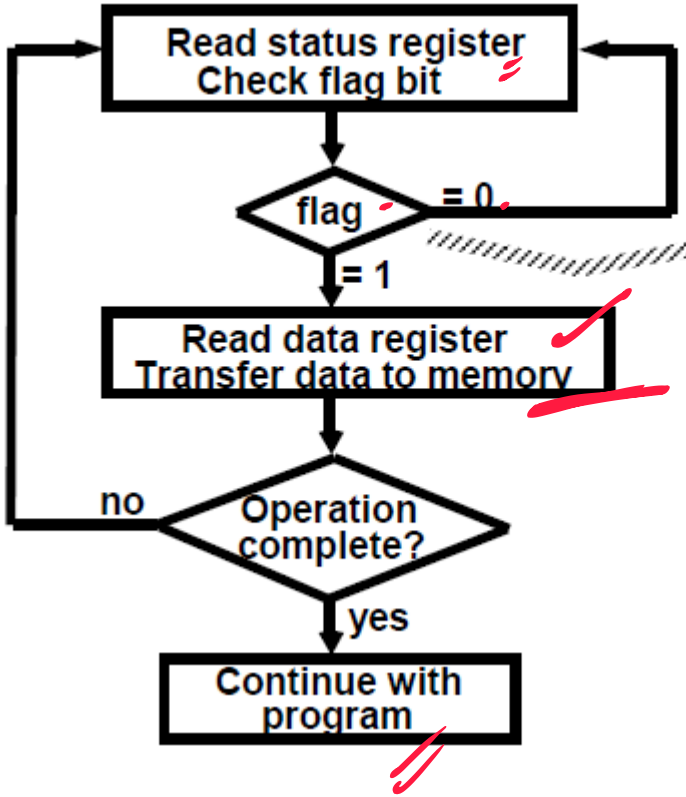
MODES OF TRANSFER - PROGRAM-CONTROLLED I/O -

3 different Data Transfer Modes between the central computer(CPU or Memory) and peripherals

1. Program-Controlled I/O
2. Interrupt-Initiated I/O
3. Direct Memory Access (DMA)

MODES OF TRANSFER – 1. Program-Controlled I/O





Polling or Status Checking

- Continuous CPU involvement
- CPU slowed down to I/O speed
- Simple
- Least hardware

MODES OF TRANSFER – 2. INTERRUPT INITIATED I/O

- Polling takes valuable CPU time
- Open communication only when some data has to be passed -> ***Interrupt***.
- I/O interface, instead of the CPU, monitors the I/O device
- When the interface determines that the I/O device is ready for data transfer, it generates an ***Interrupt Request*** to the CPU.
- Upon detecting an interrupt, CPU stops momentarily the task it is doing, branches to the service routine to process the data transfer, and then returns to the task it was performing

MODES OF TRANSFER – 3. DMA (Direct Memory Access

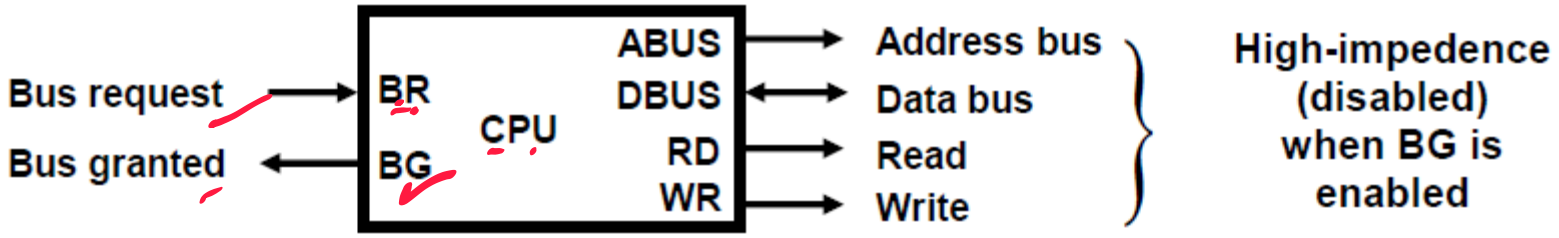
DMA (Direct Memory Access)

- Large blocks of data transferred at a high speed to or from high-speed devices, magnetic drums, disks, tapes, etc.
- DMA controller

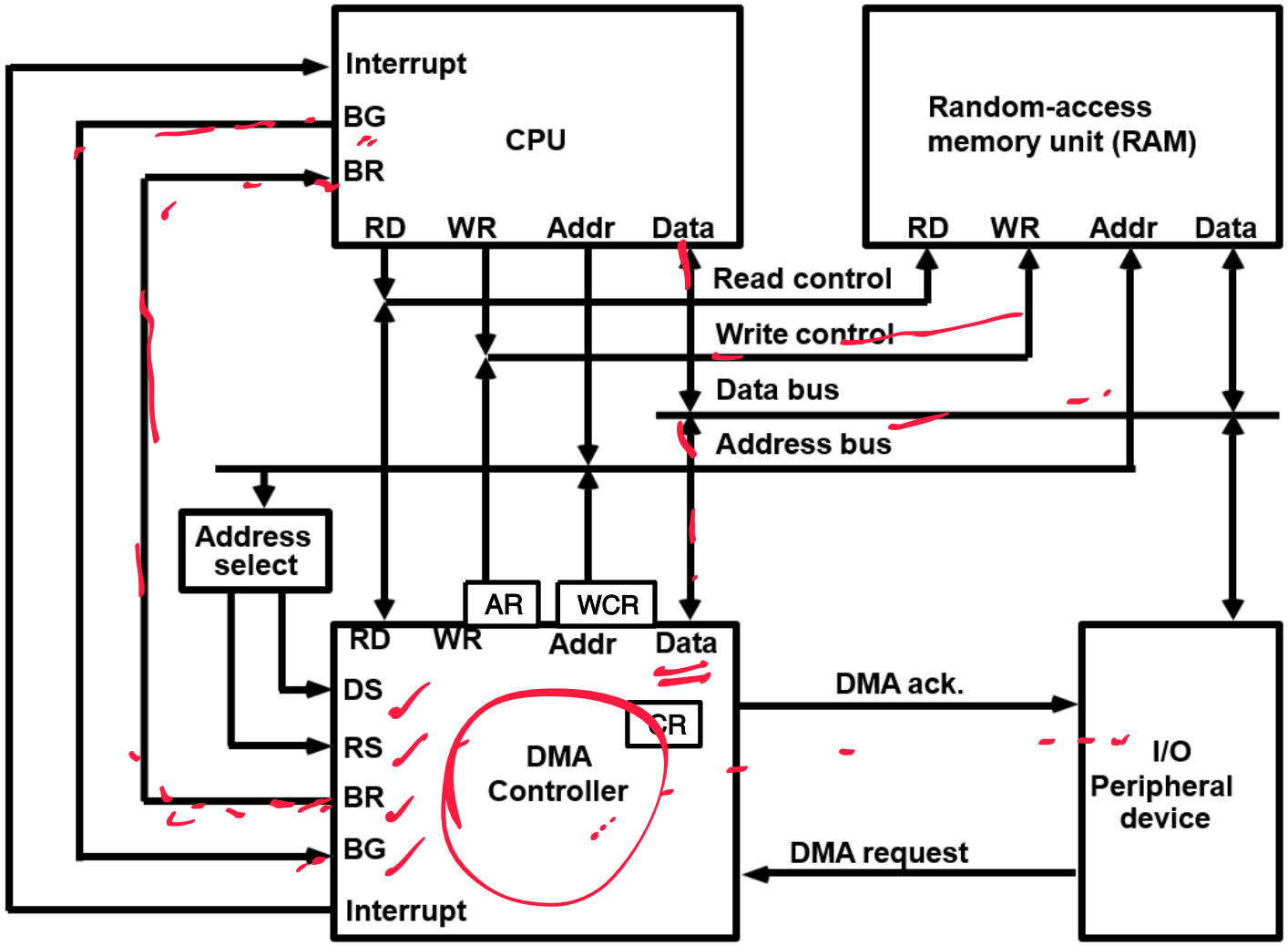
  Interface that provides I/O transfer of data directly to and from the memory and the I/O device
- CPU initializes the DMA controller by sending a memory address and the number of words to be transferred
- Actual transfer of data is done directly between the device and memory through DMA controller -> Freeing the CPU

DIRECT MEMORY ACCESS

- Block of data transfer from high-speed devices, Drum, Disk, Tape
- DMA controller - Interface which allows I/O transfer directly between Memory and Device, freeing CPU for other tasks
- CPU initializes DMA Controller by sending memory address and the block size(number of words)
- CPU bus signals for DMA transfer



Block diagram of DMA Controller



DMA I/O OPERATION

Starting an I/O

- CPU executes instruction to

  Load Memory Address Register Load Word Counter
  Load Function(Read or Write) to be performed
  Issue a GO command

Upon receiving a GO Command DMA performs I/O operation as follows independently from CPU

Input

1 Input Device <- R (Read control signal)
2 Buffer(DMA Controller) <- Input Byte; and assembles the byte into a word until word is full
4 M <- memory address, W(Write control signal)
5 Address Reg <- Address Reg +1; WC(Word Counter) <- WC - 1
6 If WC = 0, then Interrupt to acknowledge done, else go to [1]

Output

1 M <- M Address, R
M Address R <- M Address R + 1, WC <- WC - 1
2 Disassemble the word
3 Buffer <- One byte; Output Device <- W, for all disassembled bytes
4 If WC = 0, then Interrupt to acknowledge done, else go to [1]

DMA I/O OPERATION: Mode of Transfer

1. Burst Mode

2. Cycle Stealing Mode

CYCLE STEALING

While DMA I/O takes place, CPU is also executing instructions

DMA Controller and CPU both access Memory -> Memory Access Conflict

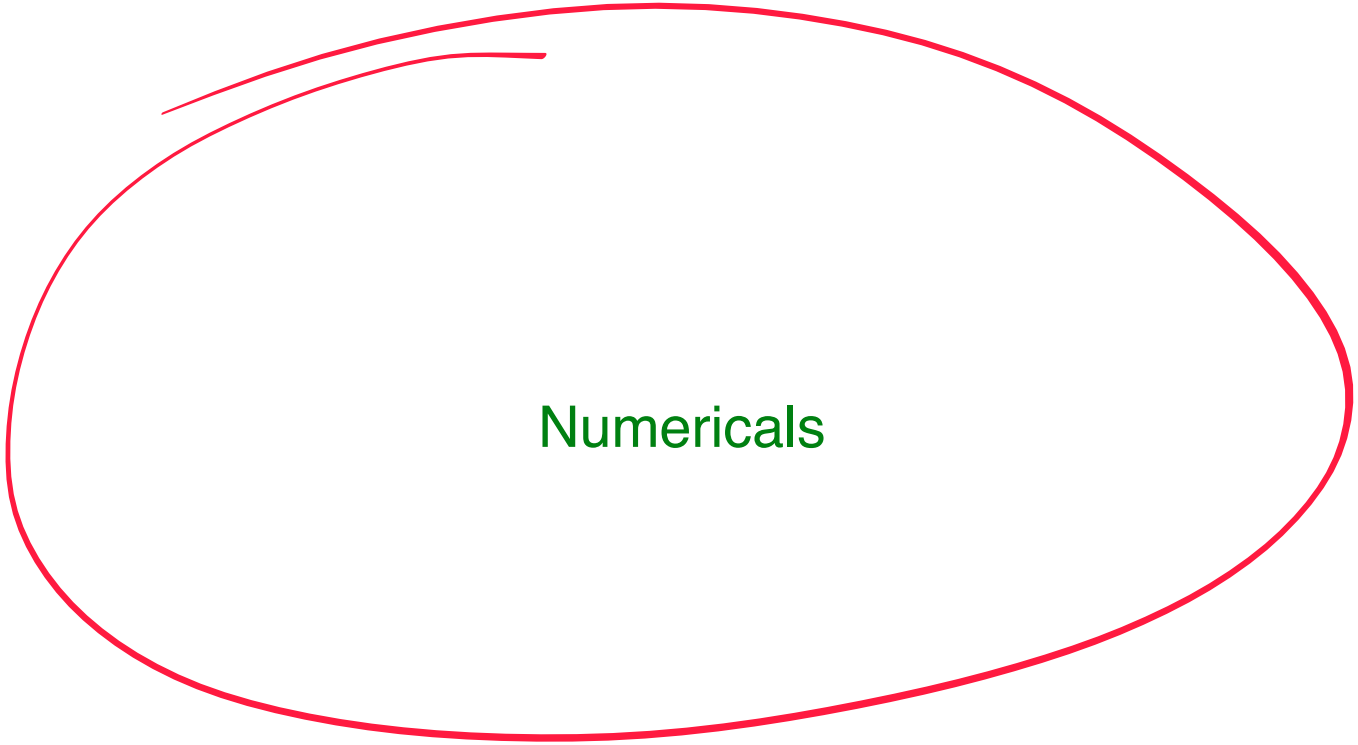
Memory Bus Controller

- Coordinating the activities of all devices requesting memory access
- Priority System

Memory accesses by CPU and DMA Controller are interwoven, with the top priority given to DMA Controller
-> Cycle Stealing

Cycle Steal

- CPU is usually much faster than I/O(DMA), thus CPU uses the most of the memory cycles
- DMA Controller steals the memory cycles from CPU
- For those stolen cycles, CPU remains idle
- For those slow CPU, DMA Controller may steal most of the memory cycles which may cause CPU remain idle long time


Numericals

The size of the data count register of a DMA controller is 16 bits. The processor needs to transfer a file of 29,154 kilobytes from disk to main memory. The memory is byte addressable. The minimum number of times the DMA controller needs to get the control of the system bus from the processor to transfer the file from the disk to main memory is _____.

Concept:

A hardware device used for **DMA (Direct memory access)** is DMA controller. DMA controller is a control unit, which can transfer blocks of data between I/O devices and main memory with intervention from the processor.

Data count register provides the number of words DMA can transfer in one cycle.

If memory is byte addressable; 1 word = 1 byte

1 kilobytes = 2^{10} bytes

Calculation:

Size of data counter register of DMA controller = 16 bits

It means, 2^{16} words can be transferred in one cycle.

As, memory is byte addressable.

So, 2^{16} bytes in one cycle. (2^{16} bytes = 2^6 kilobytes)

File size = 29154 kilobytes

Minimum number of times the DMA controller needs to get control of the system bus from the processor to transfer the file from the disk to main memory is = $\left\lceil \frac{File\ size}{bytes\ in\ one\ cycle} \right\rceil = \left\lceil \frac{29154\ KB}{2^{16}} \right\rceil = 456$

DMA - Problems

1 Byte	= 8 bits	
1 KB	= 1024 B(ytes)	= 2^{10} B
1 MB	= 1024 KB	= 2^{20} B
1 GB	= 1024 MB	= 2^{30} B

Problem 2:

A hard disk with a transfer rate of 10 Mbytes/second is constantly transferring data to memory using DMA. The processor runs at 600MHz, and takes 300 and 900 clock cycles to initiate and complete DMA transfer respectively. If the size of the transfer is 20 Kbytes, what is the percentage of processor time consumed for the transfer operation?

Solution:

$$Total\ Time = \frac{Total\ Data}{Total\ Transfer\ Rate}$$

Step 1: Find Total Transfer Time

Transfer rate=10 MB per second = 10×2^{20} bytes/sec

Data=20 KB=20×2^{10} bytes

Time=$(20 \times 2^{10})/(10 \times 2^{20}) = 2 \times 2^{-10} = 2 \times 10^{-3} = 2$ ms

$$Processor\ Time = \frac{Cycles\ for\ DMA\ transfer}{Processor\ Speed}$$

$$\%\ Processor\ Time = \frac{Processor\ Time}{Total\ Time} \times 100$$

Step 2 :Find Processor Time

1 MHz = 10^6 cycles/sec

Processor speed= 600 MHz = 600×10^6 Cycles/sec

Cycles required by CPU for DMA Transfer =300+900 =1200 cycles

Time=1200 cycles/(600×10^6) cycles/sec = 2×10^{-6} sec = .002 ms

Step 3 : Find % of Processor Time

% Processor Time = $(.002/2) \times 100 = 0.1\%$

Term	Normal Usage	Usage as Power of 2
K (Kilo)	10^3	$2^{10} = 1,024$
M (Mega)	10^6	$2^{20} = 1,048,576$
G (Giga)	10^9	$2^{30} = 1,073,741,824$
T (Tera)	10^{12}	$2^{40} = 1,099,511,627,776$

Step-(i) CPU initialize DMA ⇒ x

Step-(ii) DMA Transfer ⇒ z

Step-iii CPU completes DMA ⇒ y

$$\frac{x+y}{x+y+z} \%$$

$$T_{Time} = \frac{\text{Data Size}}{\text{Transfarate}} = \frac{20\,KB}{20\,mB} = 1\,ms$$

CPU initialize = 300 clock cycle, $f = \frac{1}{t}, t = \frac{1}{f}$

$$t = \frac{1}{300\,mHz}$$

1 Byte	=	8 bits	
1 KB	=	1024 B(ytes) =	2^{10} B
1 MB	=	1024 KB =	2^{20} B
1 GB	=	1024 MB =	2^{30} B

DMA - Problems

Problem 1:

A hard disk with a transfer rate of 20 Mbytes/second is constantly transferring data to memory using DMA. The processor runs at 300MHz, and takes 300 and 900 clock cycles to initiate and complete DMA transfer respectively. If the size of the transfer is 20 Kbytes, what is the percentage of processor time consumed for the transfer operation?

Solution:

$$Total\ Time = \frac{Total\ Data}{Total\ Transfer\ Rate}$$

Step 1: Find Total Transfer Time

Transfer rate=20 MB per second = 20×2^{20} bytes/sec

Data=20 KB=20×2^{10} bytes

Time=$(20 \times 2^{10})/(20 \times 2^{20}) = 1 \times 2^{-10} = 1 \times 10^{-3}$ =1 ms

$$Processor\ Time = \frac{Cycles\ for\ DMA\ transfer}{Processor\ Speed}$$

$$\%\ Processor\ Time = \frac{Processor\ Time}{Total\ Time} \times 100$$

Step 2 :Find Processor Time

1 MHz = 10^6 cycles/sec

Processor speed= 300 MHz = 300×10^6 Cycles/sec

Cycles required by CPU for DMA Transfer =300+900 =1200 cycles

Time=1200 cycles/(300×10^6) cycles/sec = 4×10^{-6} sec = .004 ms

Step 3 : Find % of Processor Time

% Processor Time = (.004/1) × 100 = 0.4%

Term	Normal Usage	Usage as Power of 2
K (Kilo)	10^3	2^{10} = 1,024
M (Mega)	10^6	2^{20} = 1,048,576
G (Giga)	10^9	2^{30} = 1,073,741,824
T (Tera)	10^{12}	2^{40} = 1,099,511,627,776

$t = \frac{1}{300\ MHz} = \frac{10^{-6}}{300}$ *

$\Rightarrow 300 * \frac{1}{300} \Rightarrow 1\ \mu s = 0.001\ ms$

CPU Complete $\Rightarrow$ 900 cycles

$\Rightarrow 900 * \frac{1}{300\ MHz} \Rightarrow \boxed{3\ \mu s}$

$\Rightarrow 0.003\ ms$

CPU time $\Rightarrow (x) + (y)$

$\Rightarrow 0.003 + 0.001 \Rightarrow 0.004\ ms$

$$T_T = 1\ ms$$

$$CPO\% = \frac{0.004\ ms}{1\ ms} * 100\%$$

$$\Rightarrow \boxed{0.4\%}$$

Consider a disk drive with the following specifications. 16 surfaces, 512 tracks/surface, 512 sectors/track, 1 KB/sector, rotation speed 3000 rpm. The disk is operated in cycle stealing mode whereby whenever one 4-byte word is ready, it is sent to memory; similarly, for writing, the disk interface reads a 4-byte word from memory in each DMA cycle. Memory cycle time is 40 nsec. The maximum percentage of time that the CPU gets blocked during DMA operation is

Solution:

Revolution Per minute=3000 RPM
Or 3000/60=50 RPS
In one track rotation it can read=512 KB
In 50 RPS it can read=512× 50
For 1 byte read=1/(512× 50)=39.06 ns

⇒ For 4 bytes it takes 156 ns

Percentage of time that the CPU gets
blocked during DMA operation=40156× 100
≃25