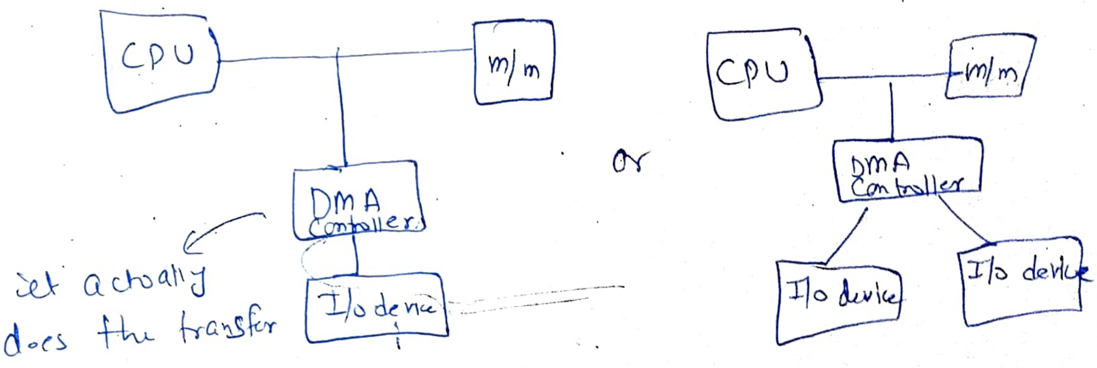
# DMA (Direct Memory Access)

Both Programmed I/o, Interrupt driven I/o has some drawbacks

- ~~Programmed I/o needs~~
- The I/o transfer rate is limited by the speed with which Up can test & service a device
- The up is tied up in managing I/o transfer, a no. of instr^n must be executed for each I/o transfer.

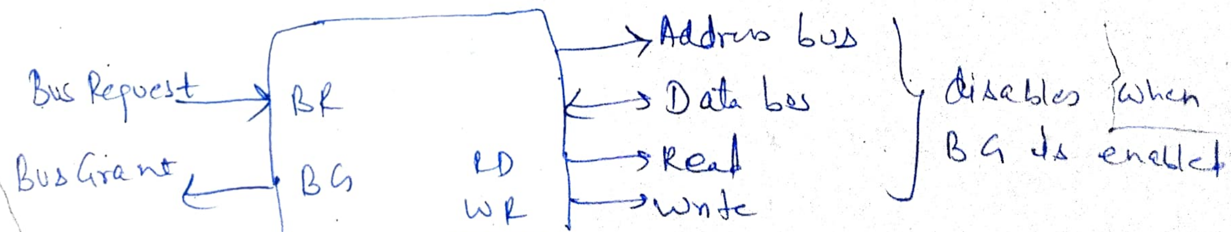
In programmed I/o → CPU is idle.
In interrupt driven I/o → Interrupt has to be handled
Interrupt can generate say after each 4 bytes transfer

DMA :- It is a data transfer technique used to transfer large amount (block of data) of data b/w I/o device & memory. eg b/w HDD ⟺ m/m, External port ⟺ m/m without severly impacting CPU performance much.

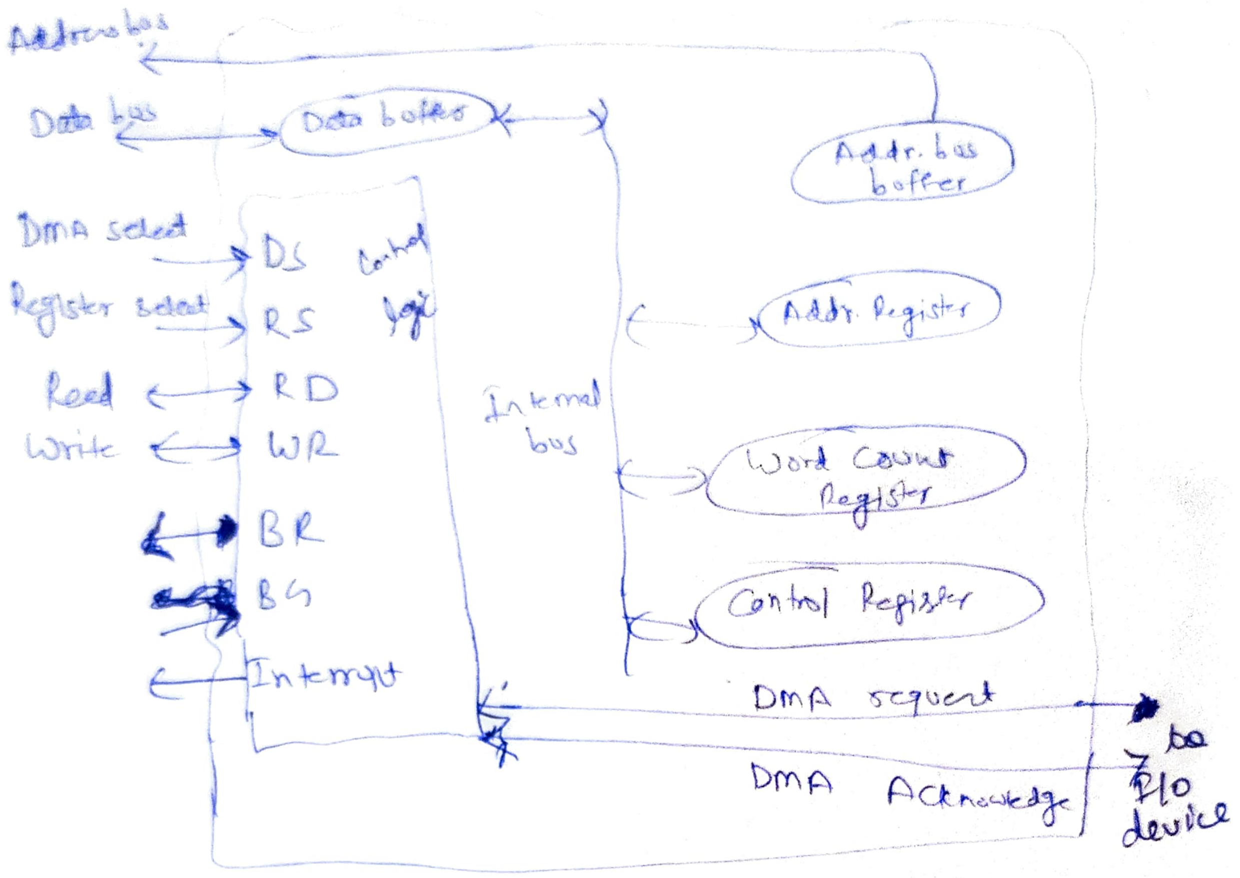


2 things needs to be understood here :- 1) CPU wrt data
2) DMA Controller figure.

CPU Bus signals for DMA transfer



DMA controller :- A circuit to handle $\overset{large}{\wedge}$ data trans. b/w ~~D~~ I/O device & m/m (A small processor)



Address Register :- Stores the starting address of m/m from where the data $\overset{or\ (block)}{}$ has to be read or written

Word Count Register :- how many words to be transferred

Control register :- Stores READ or WRITE

1) When I/O device sends some DMA request (for telling it has to store some data into m/m, say).

2) The DMA Controller sends BR request to CPU. ~~With~~ to tell CPU to leave the buses for DMA. CPU will enable DS, RS

DMA select, register select

3) CPU makes BG=1, telling DMA that its (CPU) buses are disabled. & DMA can use them for its transfr. as now CPU has no control over its buses. ✓

(4) Now DMA Controller makes DMA Ack line=1 telling I/o device that it's work will be done sooner.

(5) ~~Now the address in Address Reg. is transferred~~

(5) Now for I/o → m/m transfer, I/o device provides some data & put in data Register in DMA controller.

(6) CPU will initialise DMA controller registers via its data bus & It puts the starting address of m/m where data has to be stored.

→ put # of words to write (inside CX register)

→ put Read/write in control register (acc. to what has to be done by I/o device)

→ (Also tells address of HDD (I/o) block, which has to be written, or I/o device involved)

(6) So after CPU initialises DMA controller, & DMA contr. gets ~~data~~ block of data from I/o device. DMA controller is now ready for data transfer to m/m

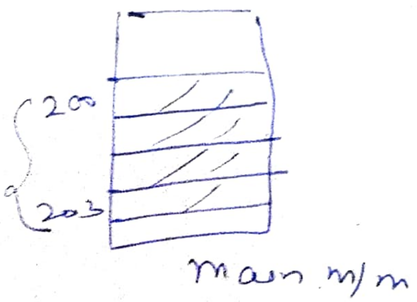
Remember:- Now CPU has no control over its bus towards m/m. DMA will use these buses

While Data transfer b/w I/o & m/m occurs, via buses, CPU will do other non-m/m tasks

Addr. Register = 200

WC " = 4 bytes

Control register = Write to m/m



(7) After each byte is written to m/m, WC is decremented. & add register value is incremented. When WC = 0, then DMA controller stops. Now DMA sends a $\boxed{\text{interrupt}}$ to CPU. telling its data transfer to m/m is over.

$\boxed{\text{Cycle Stealing}}$

Normally CPU has to do transfer b/w CPU to m/m
Now DMA. Controller will get hold of m/m bus & CPU cycle in b/w for some time & use bus for ~~b~~block data transfer b/w m/m ⟷ I/o

⇒ DMA can work in 2 modes ☆

Burst/block mode

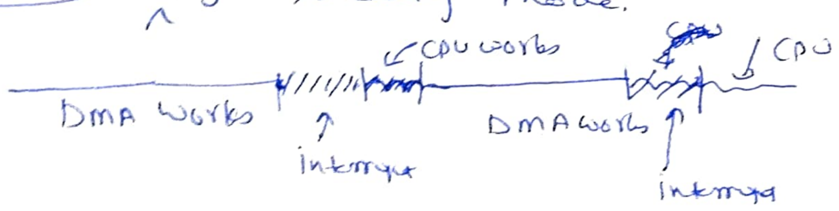
Cycle stealing mode ~~Block transfer mode~~ (CSM)

⇒ transfer all data once b/w I/o ⟷ m/m in continous manner.

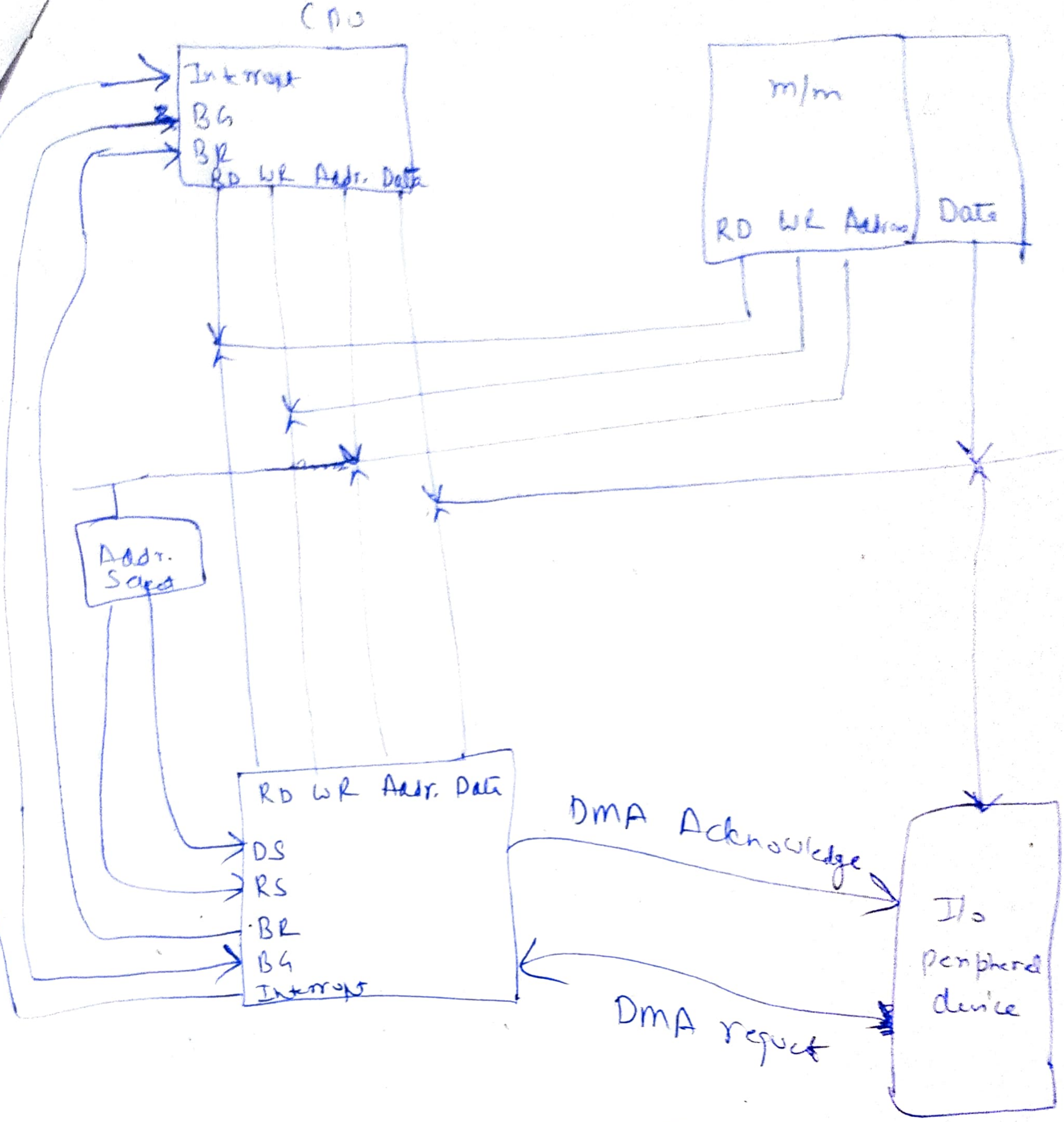
⇒ transfer data in ~~same~~ ~~blocks~~ words.

After each word ~~block~~ is transferred by DMA by stealing ~~no~~ CPU cycle, a interrupt is sent to CPU by DMA. Then interrupt is handled. Again, new word ~~block~~ is transferred by DMA

Note:- In cycle stealing mode.



CSM:- A way of data transfer b/w m/m & I/o such that DMA steals some CPU cycle & do its task & then CPU works & then DMA works. in interleaved fashion.



DMA transfer in Computer

* ~~CPU~~ DMA steals m/m cycles from CPU

* When DMA & CPU both have data to transfer (from to/from m/m, then DMA gets the priority. for transfer of data. (Why)? → due to data loss.

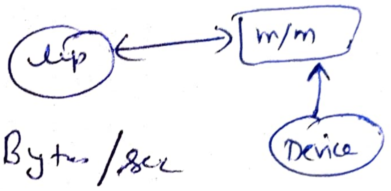
Because → $\text{DMA} \leftarrow \text{I/P}$ 1) the I/P may overrun the DMA buffer & data will be lost

2) Similarly, the O/P to DMA may also lost if DMA is not given priority for transfer. Some garbage data may be given to O/P lines.

$\text{DMA} \rightarrow \text{O/P}$

Q) A processor is fetching instructions at the rate ~~LMP~~ 1 MIPS. A DMA module is used to transfer characters to RAM (main m/m) from a device transmitting at 9600 bps. ~~How much~~ (assume 1 byte transferred in 1 cycle)
Find
a) time to transfer 1 character by DMA module
b) " " fetch 1 instruction by CPU
c) How much time will the processor be slowed down due to DMA activity?

Ans) a) DMA



~~1200~~ 9600 bps = $\frac{9600}{8}$ Bytes/sec

= 1200 Bytes/sec
or
Character [1 char = 1 Byte]

1200 character → in 1 sec.

1 " → $\frac{1}{1200}$ sec = 833 μs Ans

b) CPU

$10^6$ instructions → 1 sec

1 instructions → $\frac{1}{10^6}$ sec

= 1 μsec = $10^{-6}$ sec

c) ~~Slowdown = $\frac{1200}{10^6}$ = 1.2 ms~~

d) Slowdown % = $\frac{1200}{10^6} \times 100$ = $1.2 \times 10^{-3} \times 10^2$

= $1.2 \times 10^{-1}$

= 0.12%

(d) If 4 bytes are transferred to m/m in every 1 cycle by DMA controller (using cycle stealing), then find DMA cycles involved in transferring ~~9600 bps~~ 9600 bps.

9600 bps = 1200 bytes/sec

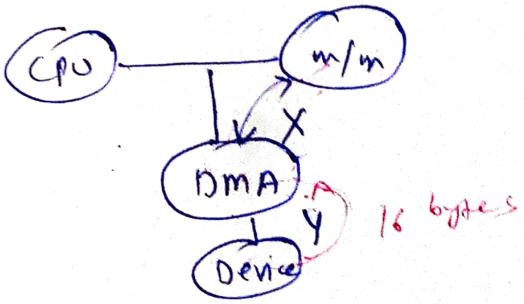
# of DMA cycles = $\frac{\cancel{1200 \text{ bytes/sec}}}{\cancel{4}}$ = $\frac{1200}{4}$

= 300 DMA cycles.

Q) Consider a device 1 MBps is operating in a cycle stealing mode of DMA whenever 16 Byte word (wordsize = 16 Bytes) is available, it is transferred into m/m in 4 us. What is the % of time up is blocked due to DMA?

Ans) % of time up is blocked due to DMA = $\frac{\text{time for which data is transferred from DMA to m/m } (X)}{\text{time for which 1 word is transferred by device to DMA controller } (Y)}$

In general
X = CPU idle time or m/m cycle time
Y = Data transfer time by I/o device



X = ~~m/m cycle~~ 4 usec

Y = ?

1 word = 16 bytes

Transfer rate of device = $10^6$ Bytes/sec

⇒ time required to transfer 1 word (16 bytes) = 16 u sec

⇒ $10^6$ bytes → 1 sec
1 bytes → $\frac{1}{10^6}$
16 " → 16 u sec

Required % = $\frac{\cancel{4} \text{ u sec}}{{}_4\cancel{16} \text{ u sec}} \times 100$ = 25%