Let,

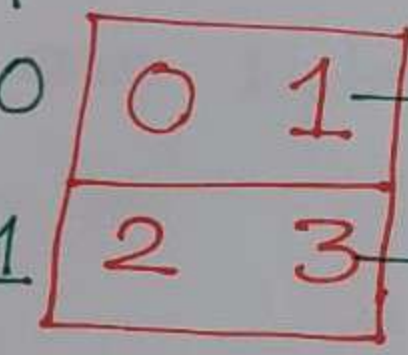
Process Size = 4 Byte

Page Size = 2 Byte

Number of Process Pages Required

For this Process = 4/2

= 2



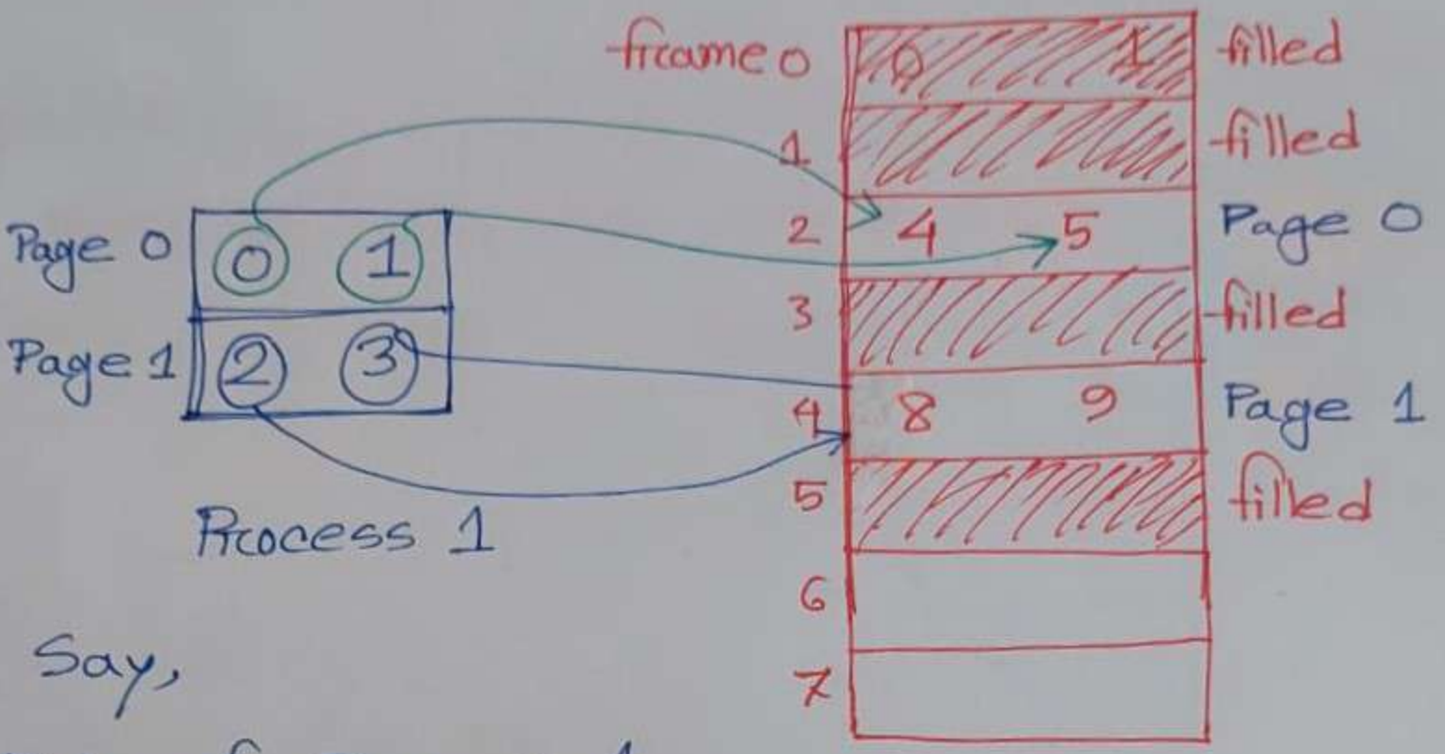
Let, Main Memory Size = 16 Byte

Frame Size = Page Size = 2 Byte

Num of Frames = 16/2 = 8

* We Can Store 16 Bytes of Data in this M.M

* Each Byte of M.M can contain one Byte of Data



Let's Say,

0th Byte of Process 1

will be stored at 4th Memory Byte of M.M.

1th Byte in 5th Byte of M.M

.

.

.

Now CPU does not know about any info related to Paging.

Lets, CPU are executing this P1 Process. CPU need each instruction / each byte for executing the Process.

Let's, CPU wants Byte 3 of Process P1

Now this 3rd Byte of Process 1 may not reside in 3rd Byte of M.M.

This 3rd B. of Process 1 can be in any available Bytes of M.M

Here CPU are generating address Page Number and Byte Number of Process 1

Page 1	Byte 3

But this address is not actual memory address of Byte 3 of Process 1 :

Byte 3 actually stores in 9th Byte of Main Memory.

MMU converts CPU generated logical address into actual Physical Address of M.M

MMU uses Page Table to perform this conversion.

Page Table contains the frame number of M.M where this 3rd Byte of Process 1 actually stores.