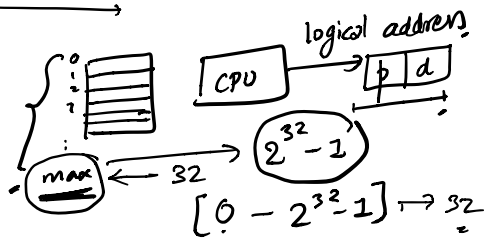
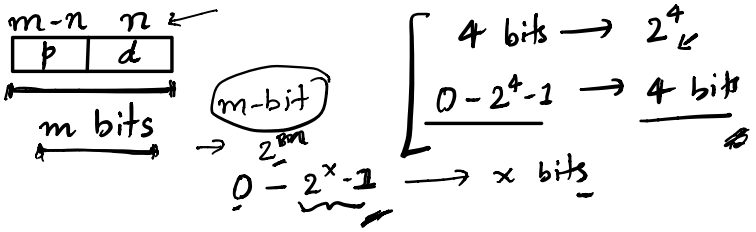
CALCULATING PAGE NUMBER AND OFFSET FROM LOGICAL ADDRESS

- Given page size which is a multiple of 2 - 2^n bytes
- Virtual address space - m-bit or 2^m bytes
- 32-bit virtual/logical address space size ?



- Binary number basics
  - Using N bits, 2^N different values can be represented including 0
  - Conversely, you need log2(N) bits to represent N values from 0 to N-1

- Logical address space -



- Total number of pages = logical address space size / page size
  = 2^m / 2^n = 2^(m-n) pages

(m-n) bits $\quad 0 - (2^{(m-n)}-1)$

- How many bits you require to represent 0 - 2^(m-n)-1 page numbers? 😛

- Page offset can vary from 0 - 2^n - 1. How many bits to represent the offset?

$0 - (2^n - 1) \rightarrow n$ bits

- How many bits are required for storing the frame number? 🤔
- Depends on the size of RAM/physical address space and frame size
- Frame size is same as the page size
- Calculate the maximum number of frames into which physical memory can be divided(similar to calculation of number of pages)
- If maximum number of frames is 2^x , then x bits are required to store the frame number

$0 - (2^x - 1) \rightarrow x$ bits

- The size of the page table depends on the number of bits required to store frame numbers

$$\text{no of frames} = \frac{\text{physical addr space size}}{\text{frame size}}$$

32 bit → 2^{32}

PRACTICE PROBLEM

Size of virtual address = 32-bit = m = 2^{32} bytes
Size of physical address = 30-bit = 2^{30} bytes
Page size = 4 KB = $2^2 \times 2^{10}$ bytes = 2^{12} bytes

$2^{\wedge n}$

Calculate the number of bits required to store page number, page offset and frame number.

$m = 32$
$n = 12$

$p = 32 - 12 = 20$ bits
$d = 12$ bits

no of frames = $\frac{2^{30}}{2^{12}} = 2^{18}$

$\left\{ \begin{matrix} 0 \\ \vdots \\ 2^{18} - 1 \end{matrix} \right.$ → 18 bits