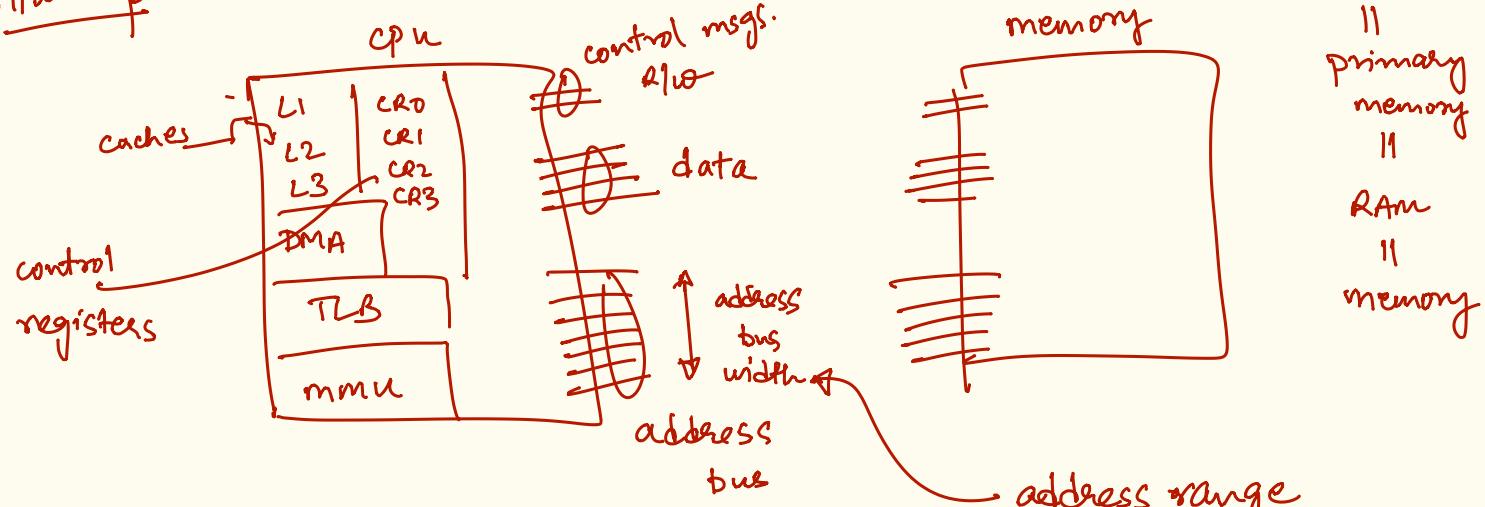
CS744 Lecture 11

(*) the address space abstraction ~ virtual memory

(i) H/w picture



address range from $0 \text{ —— } 2^n - 1$ where 'n' is width in # bits

eg: 32-bit CPU

~ addressable range

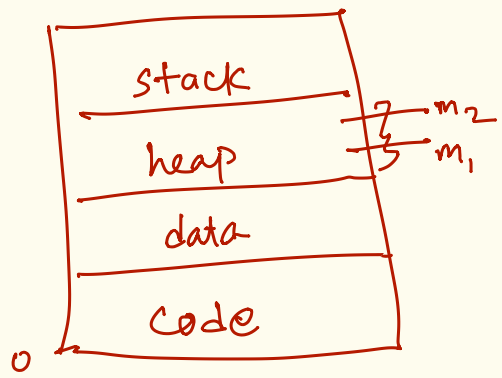
2^{32} ~ 4GB

(*) addressable range vs. physical memory

max. addressable loc^{n}s. vs. available memory.

1 address ⇒ 1 byte. (byte-addressable)

(ii) how does a process view its memory?



process view

malloc

(iii) how to manage memory across processes / (process view for this)

requirements:

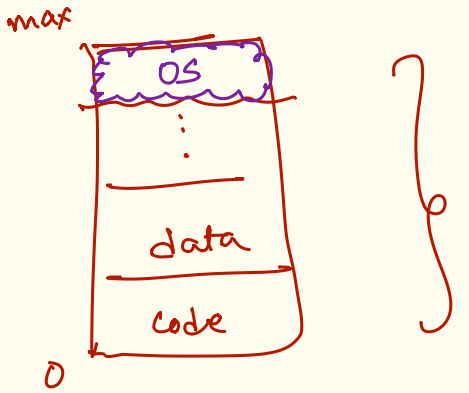
(i) isolation.

(ii) sharing possibilities / efficiency.

(iii) zero-starting linear address range w/ full addressability.

---

design 0: 1:1 mapping of process address range with physical memory



+ve: meets (i) & (iii)

-ve: ~~(iii)~~ (ii) is not met. — content extends to memory region

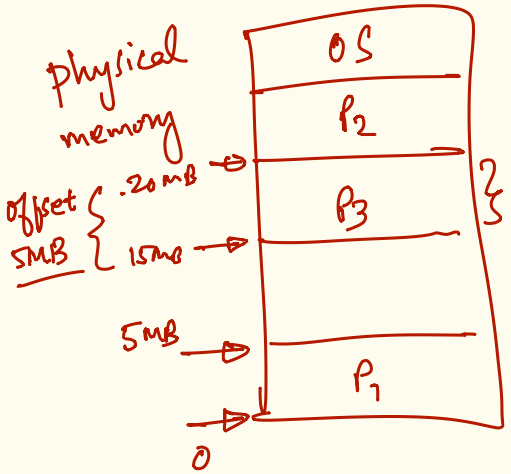
— spatial usage related inefficiency.

---

design 1:

```
int *p;
while (i)
    (*p++) = 0;
```

n:1 processes



+ve: improving on (ii)

-ve: restricts per process range

not zero-starting.

design 2:

(*) decouple the process view from physical memory

└ address space abstraction

└ addressable memory range of each process.

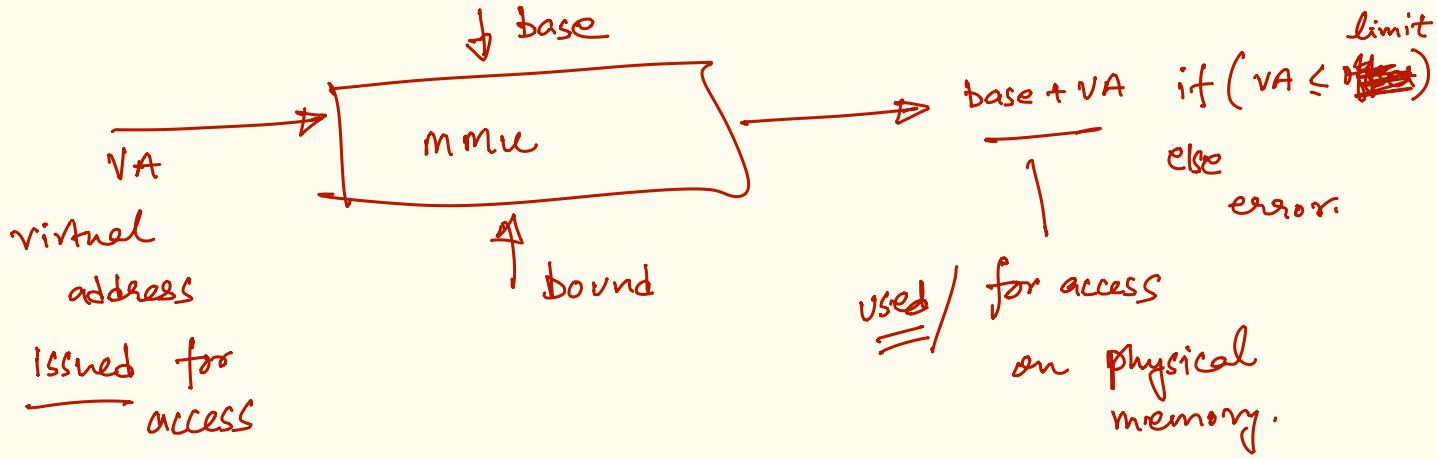
└ 0 ____ max.

(*) base & bound registers ~ CPU

└ HW supported.

base — start physical address for a process

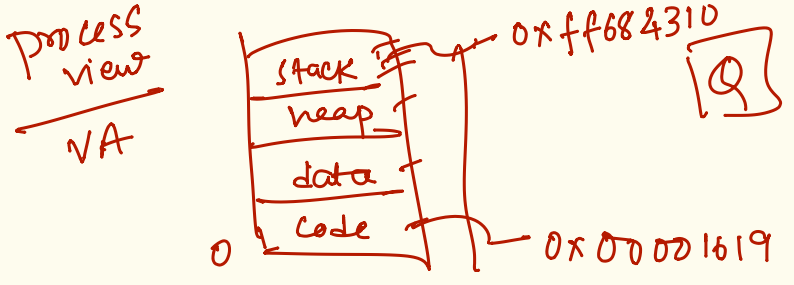
bound — max. offset of the process.



design 3: segmentation.

~ break process view segments for allocation.

~ 4 segments, individually allocated memory.



Q - how to identify which segment an address belongs to?

mov 0x00001619, 23

mov 0xff584310, 27