Lecture 12 - More Virtual Memory

In last lecture we discussed:

- 20 bit VPN → PPN mapping → page tables
  - base ptr (e.g. TLB)
  - it maps virtual addresses (VA) to physical addresses
  - flexible mapping from virtual to physical resources

- page table entry (PTE) has the following fields:
  - 20 bits
  - user/supervisor
  - PPN
  - R/W (X)
  - U/S
  - DA

Example: we have 2 executables running:

/home/joe/hw1
/home/fred/hw1

```
FFFEFFFF
  kernel
  stack
  BPPE: 2
  ./lib
  1.3MB

FFE00000
  ./lib
  .text
```

...
→ all addresses are virtual
  32 bits

\[ \begin{array}{c}
\{ & \text{UV (user virtual)} \\
\end{array} \]

\[ \text{physical mem.} \]

\[ \begin{array}{c}
P \rightarrow \text{UV:} \\
+0000 0000
\end{array} \]

the user only sees this
the kernel sees both.

→ for 3 processes:

\[ \begin{array}{c}
\text{CR3} \\
\end{array} \]

\[ \text{PP} \]

\[ \begin{array}{c}
\text{shared} \\
\end{array} \]

\[ \begin{array}{c}
\text{kernel page table}
\end{array} \]

→ language-agnostic method of allocating memory

- provide a view of contiguous memory for processes

→ efficient use of memory

1) it eliminates external fragmentation
interval vs. external fragmentation:

allocated

the allocator allocate various size
blocks

external - mem that you can't allocate to
other proc.

used

internal - mem that you overallocated

2) lazy allocation: allocate mem when you use it

- how the static mapping is used (dynamic behaviour)

page fault

not present

not present (r, w, x, u/s)

need to be able to re-start an instruction
Swapping mechanism:

This phys bloc allocated

If we have a page fault here, we need to swap, by retrieving a page from disk.

data

code

Summary:

Monday: page tables

Today: Virtual Memory

= map VA -> { PA, permissions }

dynamic behavior - page fault handling
What does VM give us?

flexible mapping
language dynamic allocation
efficient use of memory.