Virtual Memory

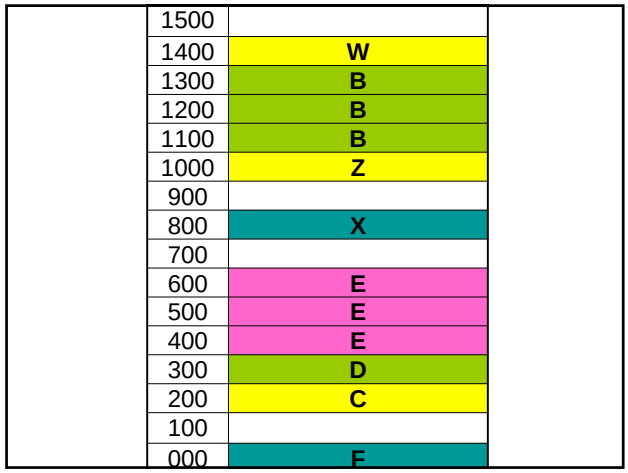
COMP755 Advanced
Operating Systems

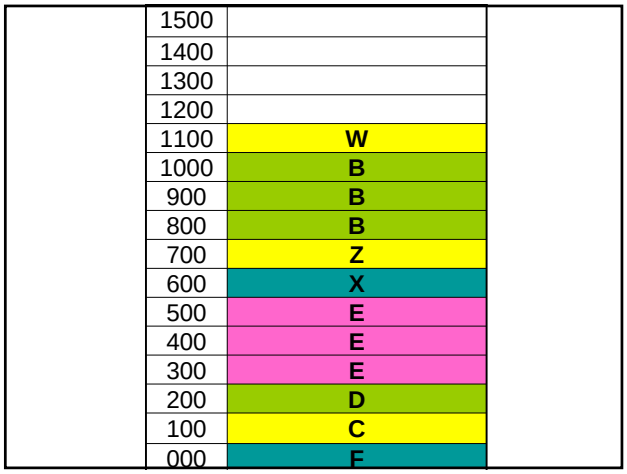
Fragmentation

- Even the best dynamic memory allocation scheme causes external fragmentation.
- As the amount of RAM in a system increases, the difficulty in allocating memory decreases.
- As the average size of programs increases the difficulty in allocating memory increases.

Moving Programs with Dynamic Memory Allocation

- If memory becomes fragmented, it can be possible to move programs to reduce fragmentation.
- Programs must be suspended to move them.
- It takes CPU time to move a program in memory.





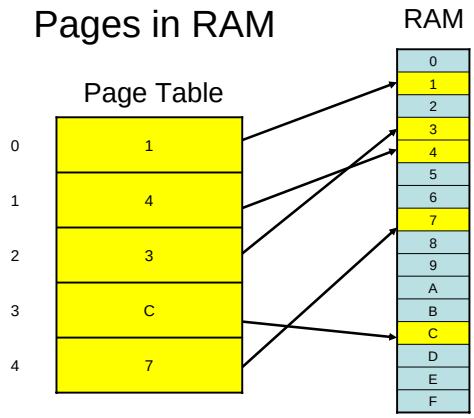
Dynamic Memory Challenges

- Some memory areas may be allocated for a long time, possibly fragmenting memory.
- Certain programs are hard to move.
  - Real time programs that may need to run immediately.
  - Shared memory requires suspended all programs using it.

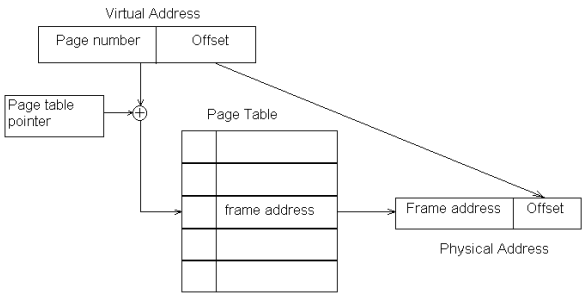
Paged Memory

- RAM is divided into fixed sized pages or frames. Pages are usually between ½K and 8K (always a power of 2).
- Programs are divided into fixed sized pages and stored in the RAM page frames
- Program pages can be stored anywhere in any order.
- A page table is used to map program addresses to physical addresses.

Pages in RAM



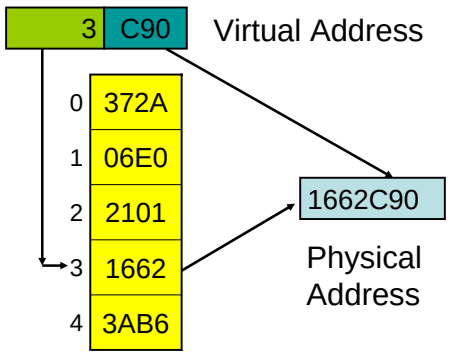
Program Address Translation



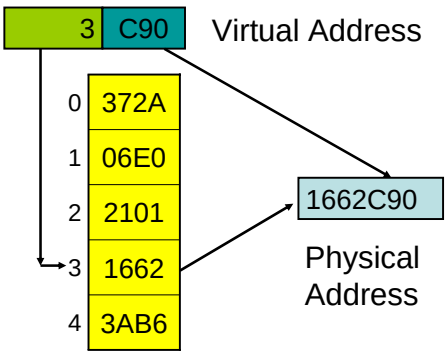
Sizes

- The length of a virtual address is $\log_2$(size of addressable memory)
- The length of the offset portion of the address is the $\log_2$(size of a page)
- The page number is all bits left of the offset.
- The number of page table entries is the programs size / page size.
- For 32 bit addresses with 4K pages (12 bits) the page number is 20 bits. A 1 MB program will have 256 page table entries.

Example



How big is this program?



Hardware / OS

- The hardware has to implement paging. Every memory access has to be translated.
- The operating system creates the page tables and allocates the pages to programs.

Paging Advantages

- No external fragmentation
- A small amount of internal fragmentation
- A program can be stored in any available page.
- Programs can occupy all of the available memory.
- Easy to share memory.

Paging Disadvantages

- More complex addressing. The addressing hardware is more complex and slower.
- Two RAM accesses are required for each memory request; one to the page table and one to access the data.

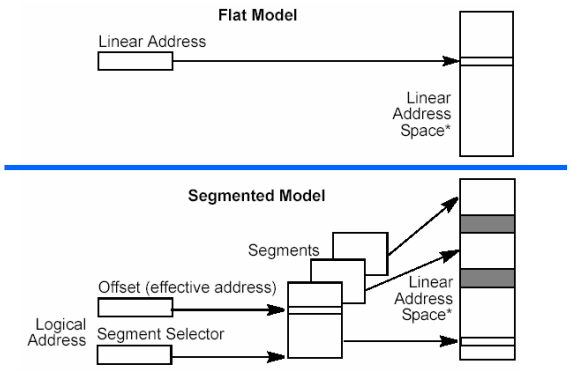
Translation Lookaside Buffer

- The **T**ranslation **L**ookaside **B**uffer (TLB) is a special cache for the page table. The TLB is on the CPU chip for fast access.
- Recently accessed page table entries are kept in the TLB.
- Most page table accesses are resolved in the TLB.

Segmentation

- Memory can be separated into segments based on the program.
  - instruction segments for each function
  - data segments for global data and heap
  - stack segments
- Segmentation provides greater function isolation and makes linking easier.
- Segments can have virtual memory pages

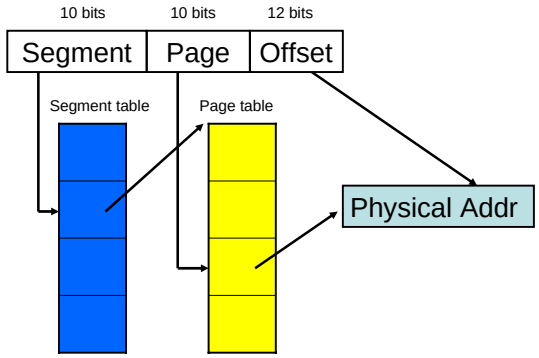
Memory Models



Combining Paging and Segmentation

- An addressing system can use both paging and segmentation.
- Large segments can be composed of pages.
- Allows simple memory allocation and logical program division.

Intel Segment Addresses



Sharing Memory

- Often it is advantageous to share memory.
- Multiple users running the same program.
- Programs communicating with shared data
- The segment tables of different users can point to the same shared memory location.
- Depending upon the program address used, a user might access private segments or shared segments.

Big Programs

- Even when using every byte of RAM, it is not always possible to load all the programs users would like.
- Frequently large parts of programs are never executed. There are many features of Microsoft Word® you have never used.
- More programs could fit in memory if only the used portions were loaded into RAM.

Locality of Reference

- **Temporal locality** - a referenced location is likely to be referenced again.
- **Spatial locality** - nearby locations are likely to be referenced soon.

Virtual Memory

- Unused pages of a program do not need to be in RAM to execute the program.
- A “resident” bit is added to the page table.
- Pages not in RAM have the resident bit cleared.
- Unused pages are stored on disk.
- When a program references a page with the resident bit clear, the hardware creates a page fault interrupt.

Virtual Memory OS

- When a page fault interrupt occurs, the OS reads the desired page from disk into an available page of RAM.
- The user's page table is updated to point to the newly loaded page.
- The program is placed back on the ready list to be executed.

Virtual Memory Advantages

- Allows you to fit many large programs into a relatively small RAM.
- Only part of a program needs to be loaded into memory.
- Eliminates the need to "fit" programs into memory holes.

Virtual Memory Disadvantages

- Requires complex hardware support.
- Makes address translation much more complicated.
- Can reduce performance.
- Makes program execution time less predictable.

Virtual Memory Performance

- Computers can retrieve a word from RAM in about 60 ns (6×10^{-8} sec)
- A good disk drive can read a block of data in about 6 ms (6×10^{-3} sec)
- If the needed page is not in RAM and has to be read from the disk, it takes about 100,000 times longer to get the data.
- Page fault interrupts have to be kept to a minimum.

Performance Factors

- The hardware handles the normal address translation.
- When a page fault occurs, the OS determines where the page will be loaded and what page will be overwritten.
- The OS must minimize the number of page faults.

Size of the Page

- Set by the hardware
- Often between 512 -8K bytes
- Some system have very large pages, such as 64K or 1MB
- Bigger the better
- Should correspond to a size easily written to disk.

Fetch Policy

- When should a page be loaded into RAM?
- Almost all systems copy a page to RAM when referenced.
- Prepaging can be effective for initial program load.

Placement Policy

- Where in RAM to put the program?
- All memory is equal. Put the page anywhere.

Cleaning policy

- When is a changed page written back to disk?
- When a program changes a variable, the RAM copy is different from the disk copy.
- Written to disk when page is needed.
- Write to disk when system is idle.
- If the program terminates, none of the pages need to be written back to disk.

Replacement Policy

- What page should be overwritten?
- Optimal - requires looking forward in time. Useful for evaluating other algorithms.
- LRU - works well.
- FIFO - denies the concept of locality
- Clock - used by Mac OS.
- Page Buffering - System keeps a list of available and released pages.

Resident Set Management

- Fixed or variable number of pages per process.
- Pages replaced globally or within a process.
- Working set

Multiprogramming Level

- How many programs can run simultaneously?
- Swap out idle programs