CMP 1203

LECTURE 6

Virtual Memory

- Use hard disk as an extension of RAM thus increasing available address space
- Place on hard drive which holds extra space ➔page file
- Imaginary memory location whose addressing is handled by the OS.
- Usually implemented using paging: main memory is divided into fixed size chunks and programs divided into chunks/blocks of the same size
- Can also be implemented by segmentation or a combination
- Blocks of program brought into memory when they are needed
- No need to store contiguous chucks of program in contiguous chunks of memory
- Out of order storage ➔ need to translate program addresses generated by CPU (virtual memory addresses) to main memory (physical) addresses

Definitions

- ***Virtual address***: logical or program address used by a process
- ***Physical address***: real address in physical memory
- ***Mapping***: translate virtual addresses to physical ones
- ***Page frames***: equal size blocks into which main memory is divided
- ***Pages***: blocks into which virtual memory is divided. Same size as page frame. Stored on disk until needed.
- ***Paging***: Copying virtual page from disk to page frame in main memory
- ***Fragmentation***: unusable memory
- ***Page fault***: An event which occurs when a requested page is not in main memory and must be copied from disk into memory

Paging

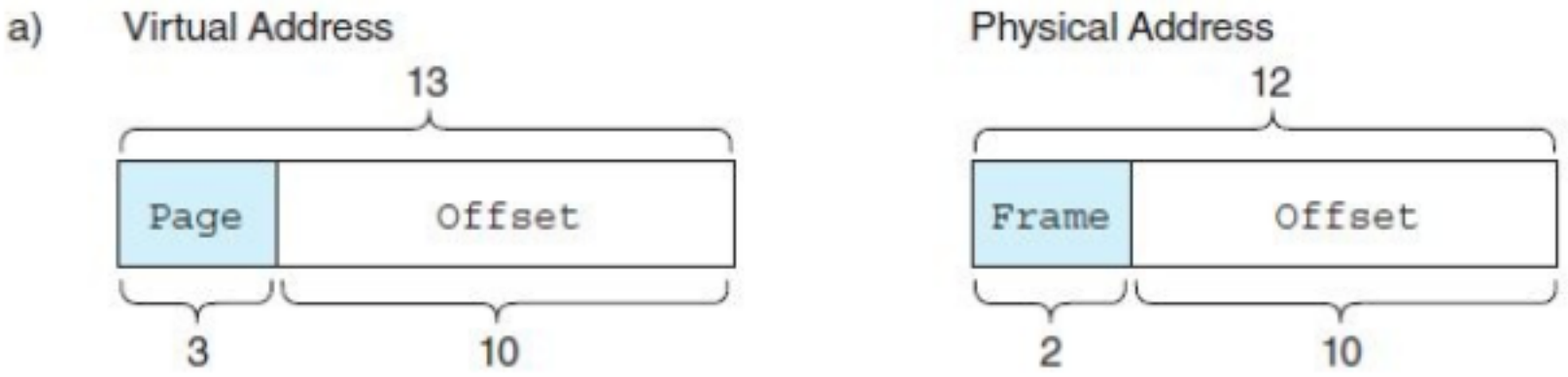
- Allocate physical memory to processes in fixed chunks and keep track of where the pages are using a page table
- Each entry of page table has 2 fields: a valid bit and frame number
- Valid bit indicates whether page is in main memory or not
- Can have additional bits e.g. modify bits, usage bits etc.
- Virtual address translated to physical address by dividing virtual address into 2 fields:
- i) page field indentifies page
- Ii) offset field identifies where in the page the data is located

- **Homework: How do you select page size?**

Example

- Virtual address space of 8K bytes, physical memory size of 4K bytes. Byte-addressable, page size of 1K bytes. Assuming no cache.

1. Virtual address format: 2^{13} bytes; 2^{3} pages
2. Physical address format? 2^{12} bytes; 2^{2} pageframes

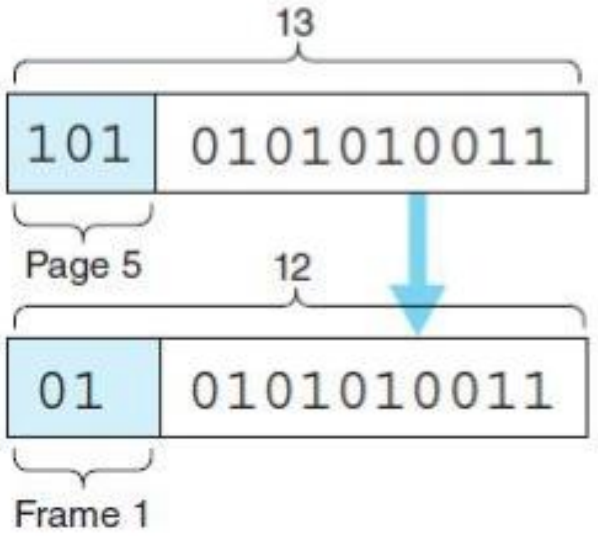


Example

- Assume page table below. How would we translate address 0x1553 = 1010101010011_2
- Page 101 = 5, offset 0101010011

b) Page Table

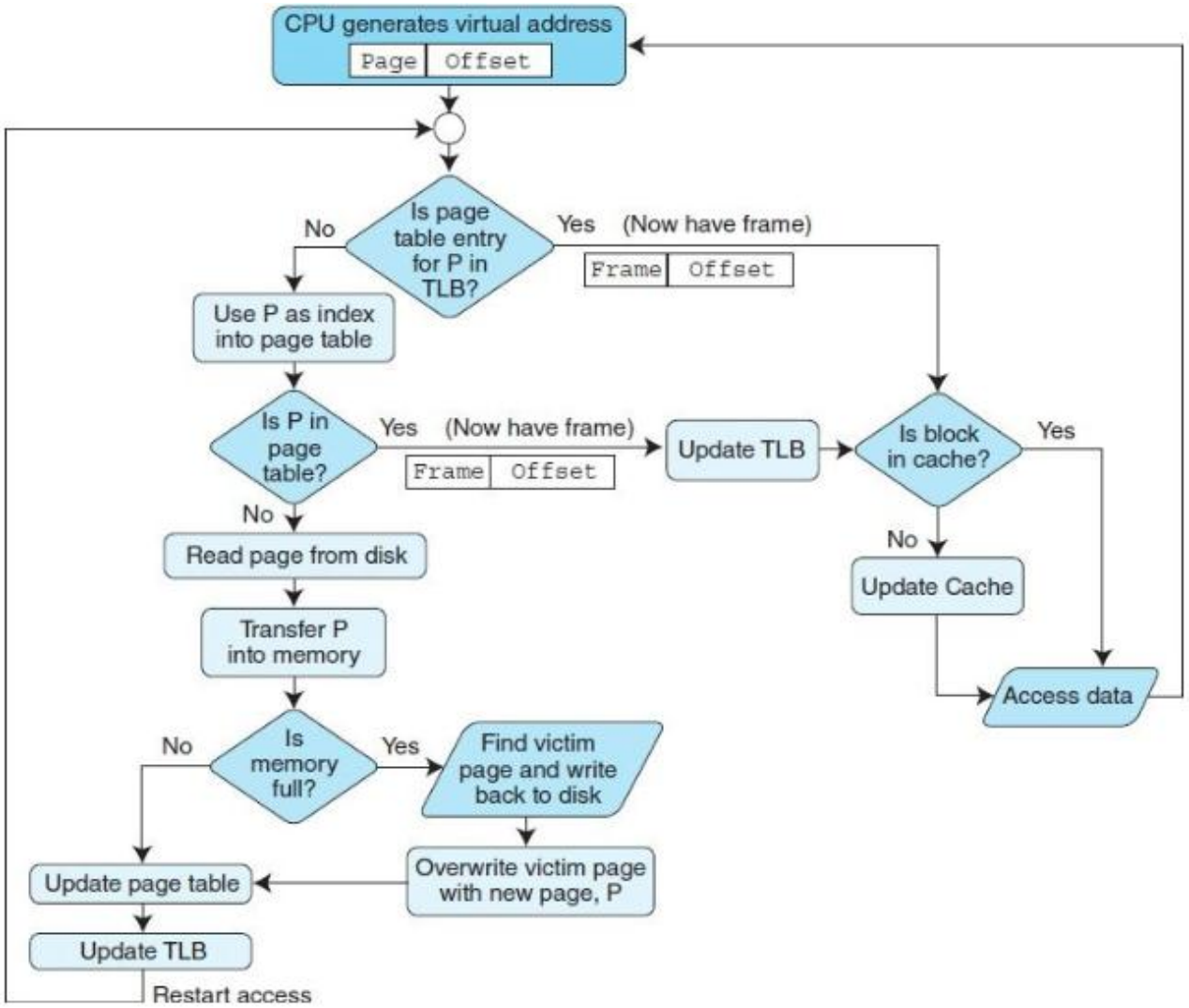
Page	Frame	Valid Bit
0	-	0
1	3	1
2	0	1
3	-	0
4	-	0
5	1	1
6	2	1
7	-	0



Effective Access Time

- Virtual memory using paging introduces time penalty
- For each memory access, there are 2 memory accesses: 1 for referencing page table, 2nd for the actual data
- Costs an additional memory access because page table is stored in main memory
- Speed up page table look up by using a page table cache called translation look-aside buffer (TLB)
- Store most recent page lookup values here

Cache + TLBs + Paging



Advantages and Disadvantages of Paging and Virtual Memory

- Time penalty
- Extra resource consumption (need to store page tables)
- Need special hardware and OS support

BUT

- No physical memory limitations : ease programming
- Can run multiple programs concurrently : each uses less actual physical memory
- So increased CPU utilization and throughput

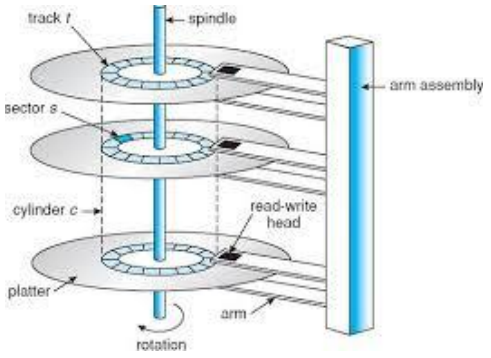
- **Homework: read other implementations of virtual memory: segmentation and combined paging + segmentation.**

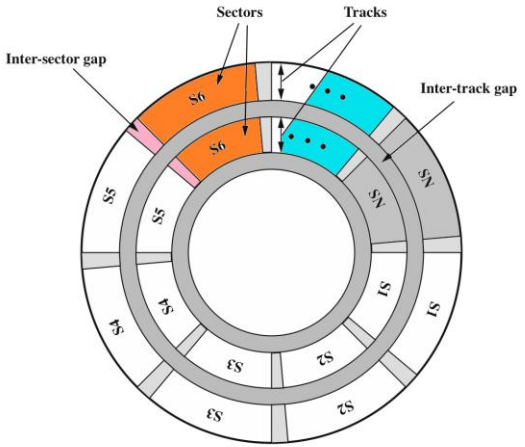
Memory: Performance measures

- ***Density***: number of bits that can be represented on a fixed chip size. Higher density more memory but higher power and higher heat generation
- ***Separation of read and write performance***: time to fetch/store data from/to memory
- ***Latency and cycle time***
- Memory controller interfaces between processor and memory
- Translates read/write requests into signals understood by memory
- ***Latency*** is time between start and completion of an operation e.g. a write
- Controller may need additional time between operations e.g. to reset hardware so cant just use latency alone
- Instead measure time required for successive operations
- ***Read cycle time and write cycle time***: time between start of one memory access and time when another access can be started. Includes latency

Magnetic Disks

- Consists of concentric circles called tracks, divided into sectors.
- Each sector has a unique address
- Each track has same number of sectors and each sector holds same number of bytes.
- One or more metal or glass disks (platters) coated with magnetic material
- Platters are stacked on a spindle which is rotated by motor
- Read/write heads are magnetized with current to write data to disk and magnetized spots can induce current in head to allow data to be read



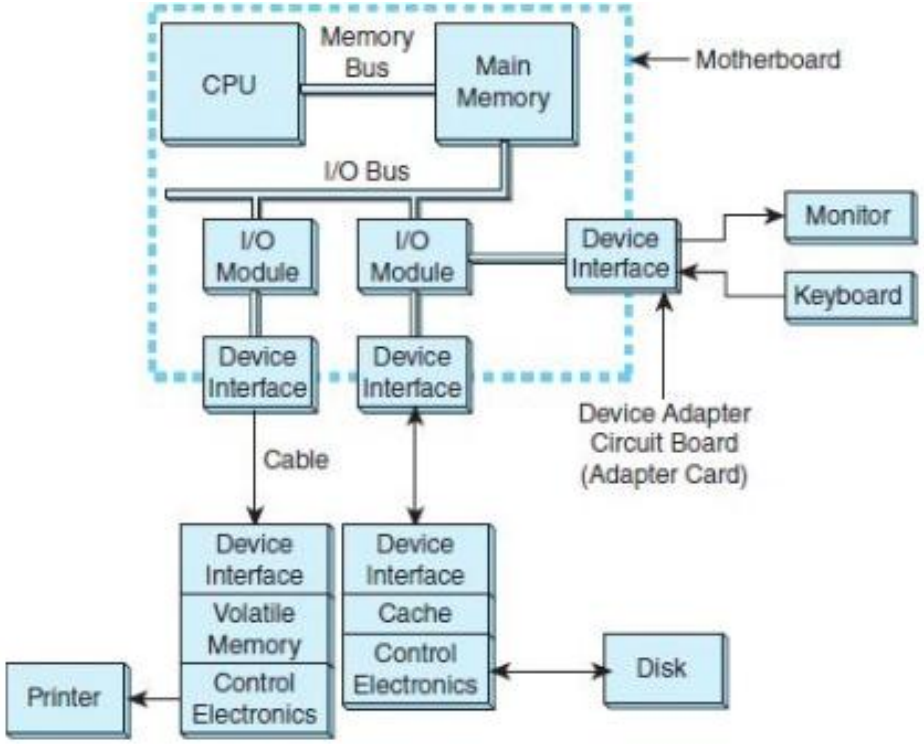


Optical Disks

- CD-ROM,-R,-RW, DVD, Blu-ray
- Plastic discs coated with a layer of aluminium
- Data is recorded in small pits made by a laser or stamping machine
- Reflects light from green laser to photo-detector which converts light to electric signals.
- Data is read from the varying intensity of light reflected
- Signals sent to decoder electronics

Input/Output

- I/O: sub system of components which moves coded data between external devices and host system (CPU + main memory)



/O Configuration

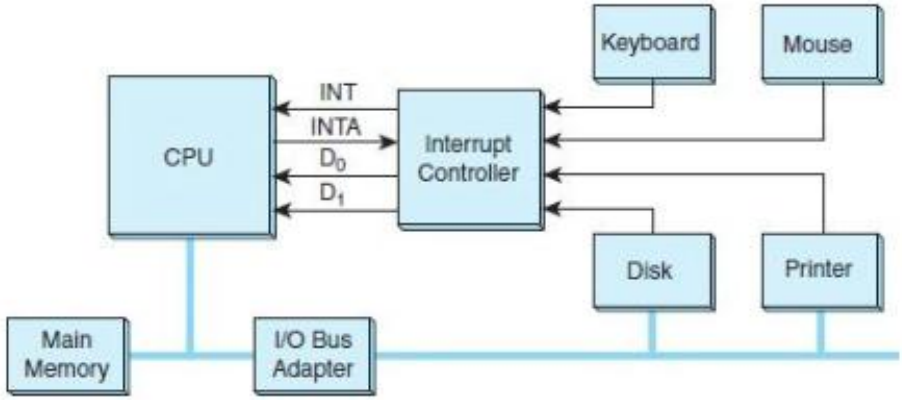
- I/O modules move data between memory and device interfaces
- Interfaces ensure devices are ready for data / host can receive data from the peripherals
- Protocols define the form and meaning of signals between sender and receiver
- Handshaking: receiver must acknowledge commands and data sent to it or indicate readiness to receive data

I/O Control Methods

- **How do I/O modules communicate with the CPU?**
- 5 methods
- ***Programmed I/O***
- Simple
- Also called polled I/O or port I/O
- CPU continuously monitors control registers associated with each I/O port
- When data arrives in port, a bit in control register is set.
- When CPU polls port, notices it has data to send
- CPU gets data and processes it and resets control bit
- Resumes polling
- **Advantages**: can control behavior of each device by modifying code e.g. number of devices, polling intervals, priority
- **Disadvantages**: CPU wastes time polling, how to decide how frequently to poll each device?
- Suitable for special purpose systems e.g. ATMs, environmental monitoring

Interrupt- Driven I/O

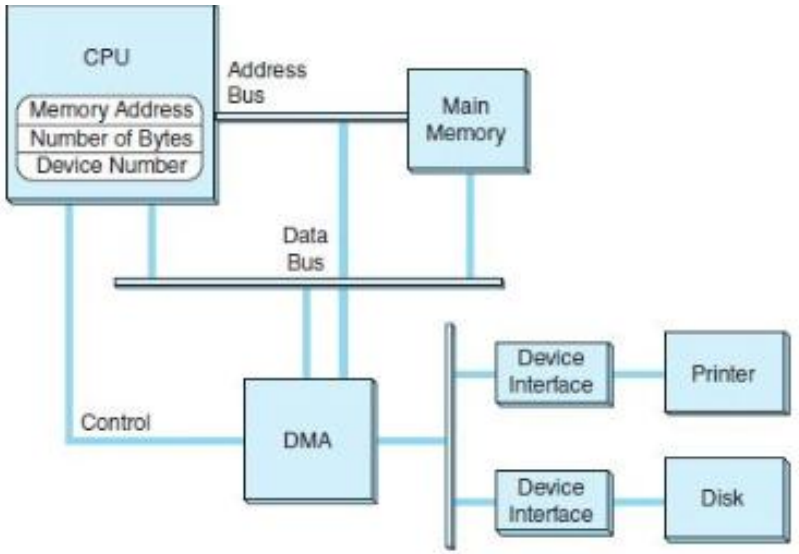
- CPU proceeds with other tasks until devices notify it that they have data to send
- Devices send interrupts to CPU
- Interrupt controller handles all interrupt signals
- All peripherals access controller, and it notifies CPU
- CPU sends ACK when it is ready to process the interrupt
- If two interrupts occur simultaneously, controller decides which takes precedence depending on which which device is more time-critical
- System designers decide device precedence – hardwired into controller



'.2 An I/O Subsystem Using Interrupts

I/O Control

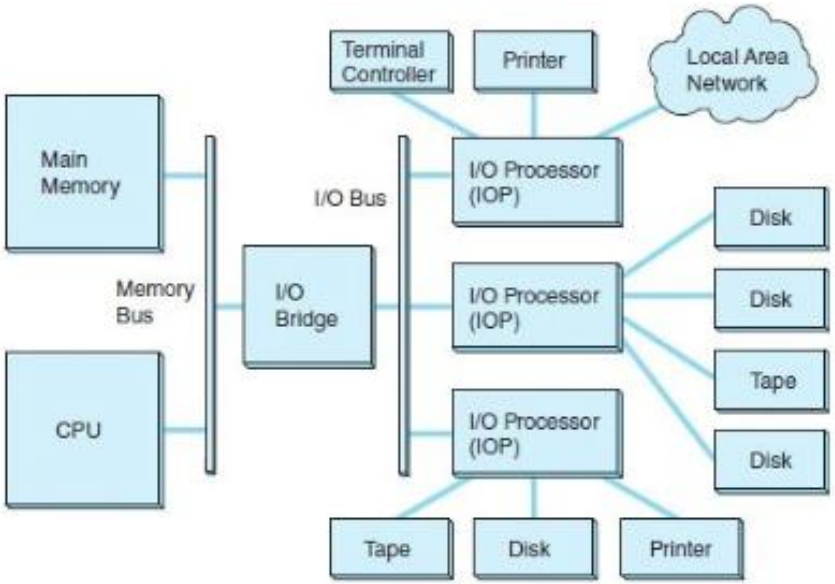
- ***Memory-mapped I/O***
- I/O devices and main memory share same address space
- Each I/O device has reserved block of memory
- Data to/from device involves moving bytes to/from its memory address
- CPU views it as a memory access
- ***Direct Memory access*** involves a DMA controller
- CPU tells controller location of data to be transferred, destination device/memory address, number of bytes etc. and then leaves controller to handle the detailed I/O and proceeds with other tasks
- Both share the bus but I/O has precedence over CPU memory fetches



DMA Configuration

I/O Control

- ***Channel I/O***
- Large computer systems
- I/O channel is an advanced DMA interface
- I/O channels are driven by small CPUs called I/O processors which are optimized for I/O



7.7 A Channel I/O Configuration

Handout

- Read Chapter 6 and 7 of Lobur and Null
- Douglas Comer, Essentials of Computer Architecture, Chapter 10 and 11