



# Ceng 111 – Fall 2021

## Week 4b

### Digital Computation

**Credit:** Some slides are from the “Invitation to Computer Science” book by G. M. Schneider, J. L. Gersting and some from the “Digital Design” book by M. M. Mano and M. D. Ciletti.

# Von Neumann Architecture

## ■ Pros:

- Simplifies hardware (both circuit-design and layout)
- Easier to generate re-locatable code, which makes multi-tasking easier to implement.

## ■ Cons:

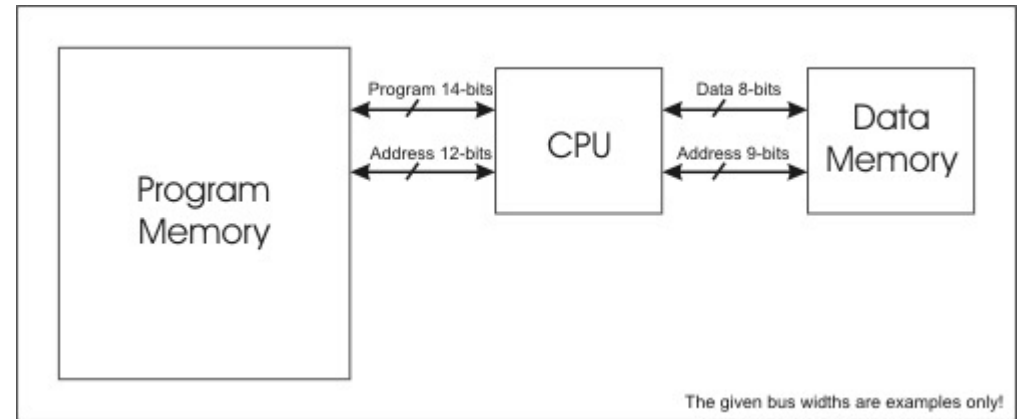
- Instructions must be multiples of the data bus-width - can be inefficient.
- Variable number of cycles required for instructions. For example, an instruction that requires data from memory must wait at least another cycle before it can complete, whereas some instructions execute much faster. This can be a problem for time-critical applications.

<http://www.mhennessy.f9.co.uk/pic/architecture.htm>



Previously on CENG111!

# Harvard Architecture



<http://www.mhennessy.f9.co.uk/pic/architecture.htm>

## ■ Pros:

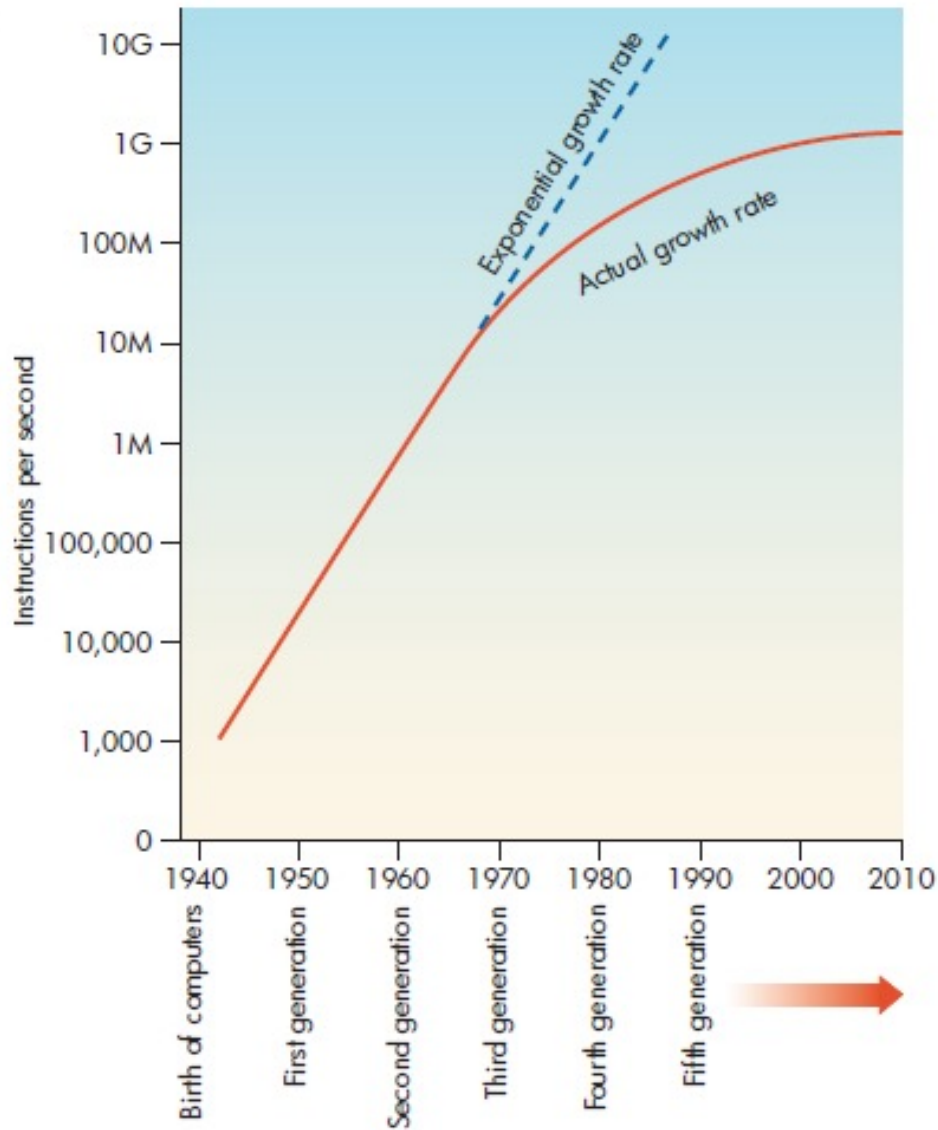
- Data and address busses can be of different widths. This means the program memory word can be wide enough to incorporate an instruction and a literal (fixed data) in a single instruction.
- A built-in two-stage pipeline overlaps fetch and execution of instructions, meaning most instructions execute in a single clock cycle.

## ■ Cons:

- Slightly more confusing.
- Hardware is more complicated.



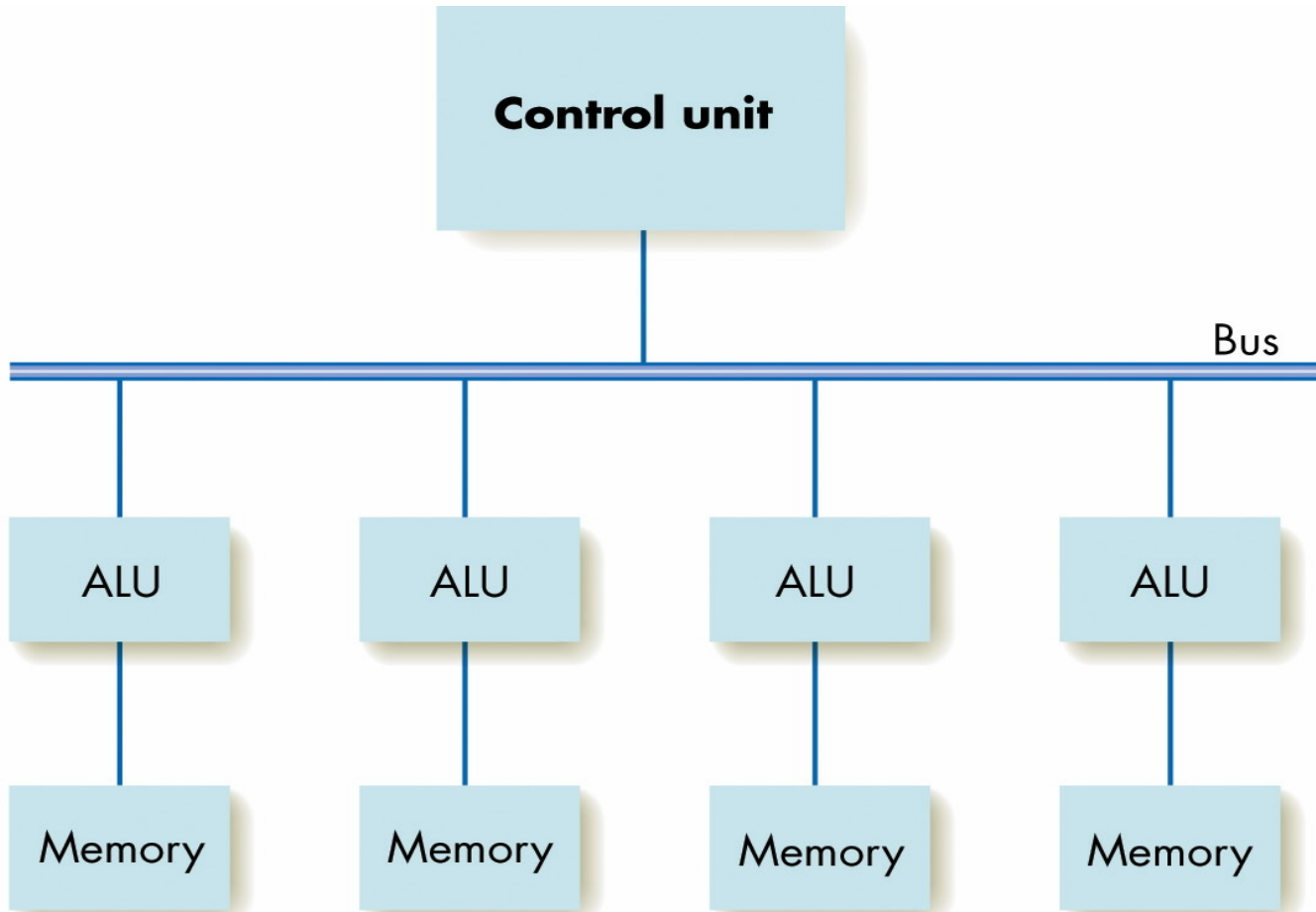
# The Future





**Replicated  
ALU units**

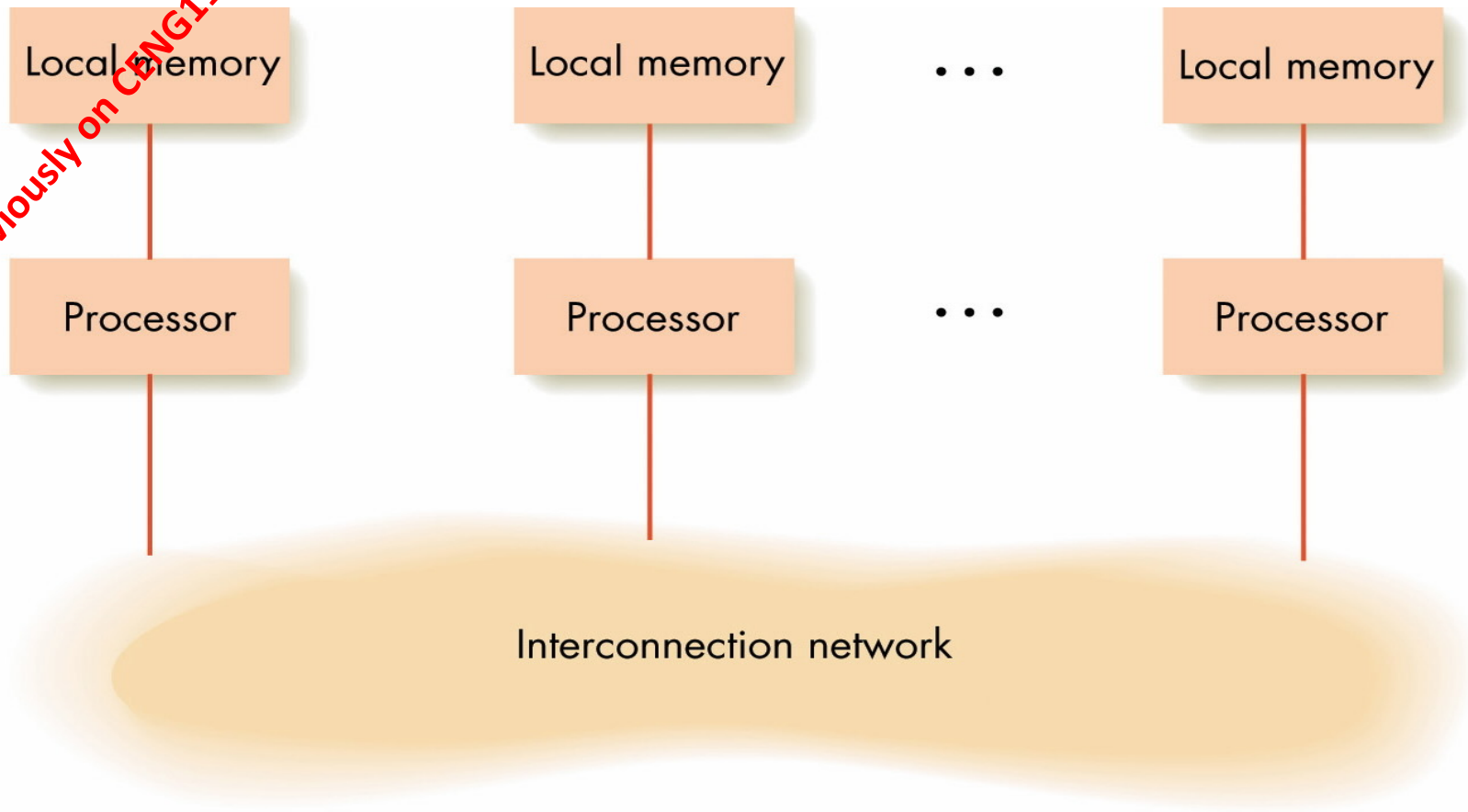
**Local  
memory**



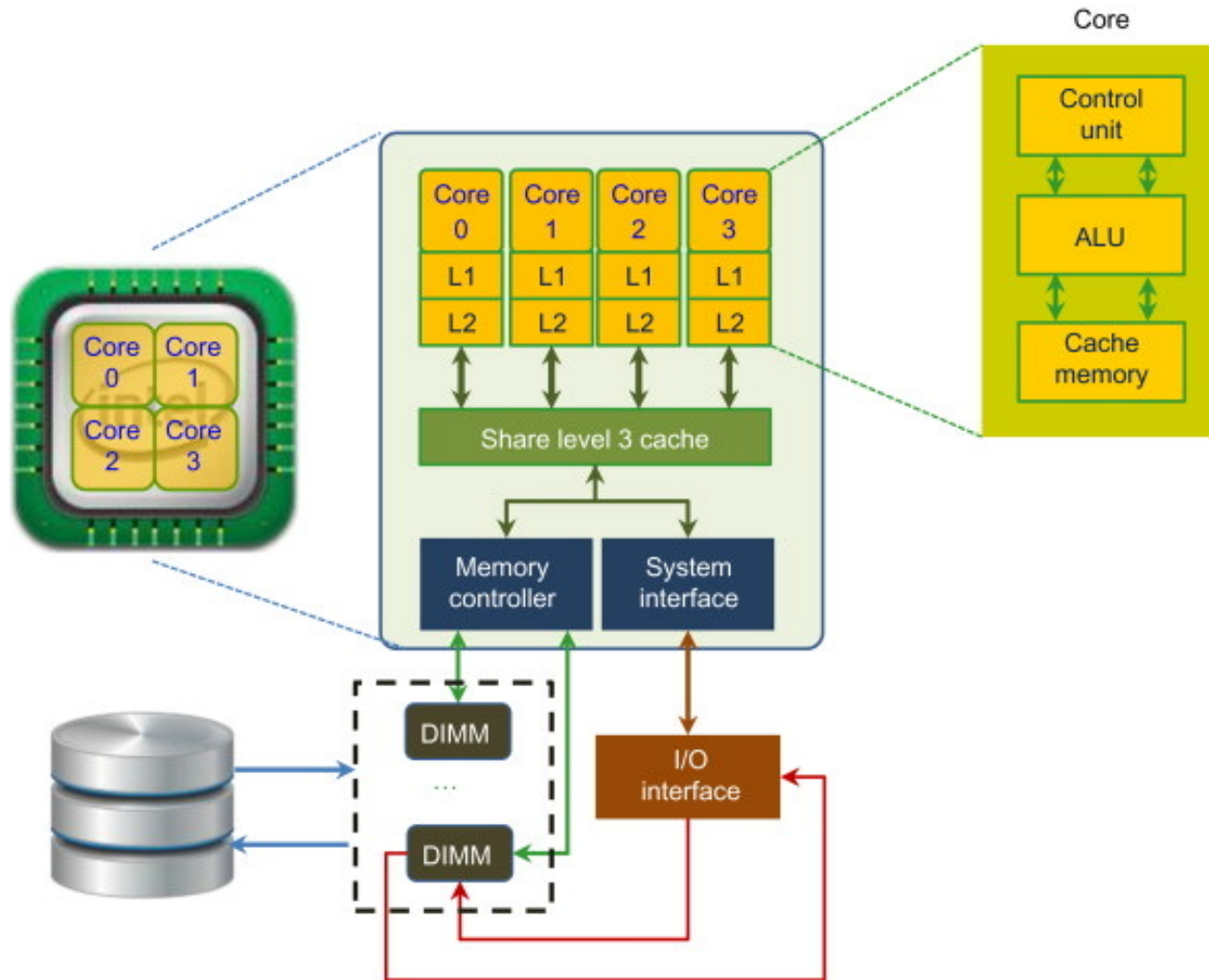
## A SIMD Parallel Processing System



Previously on CEng111!



## Model of MIMD Parallel Processing



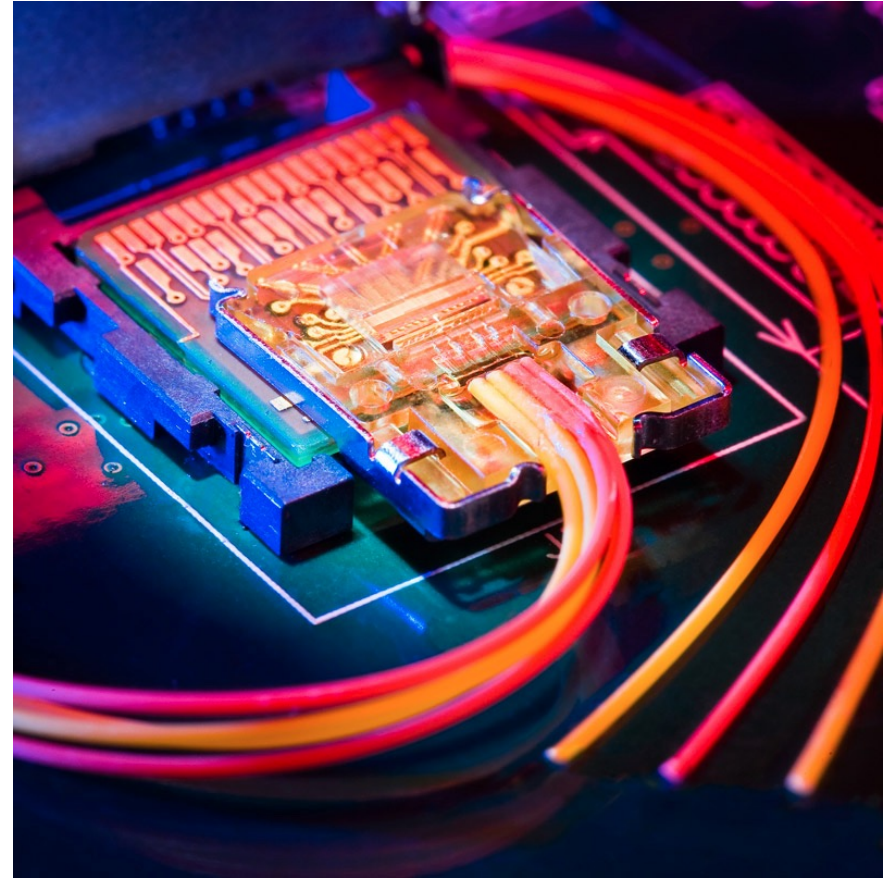
<https://www.sciencedirect.com/topics/computer-science/core-processor>



Previously on CENG111!

# New Trend: Optical Computing

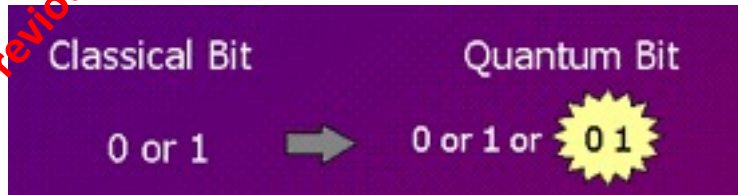
- Currently available for only data transfer.
- Work in progress towards “photonic logic” for designing circuits which use photons:
  - There are a lot of challenges and disadvantages



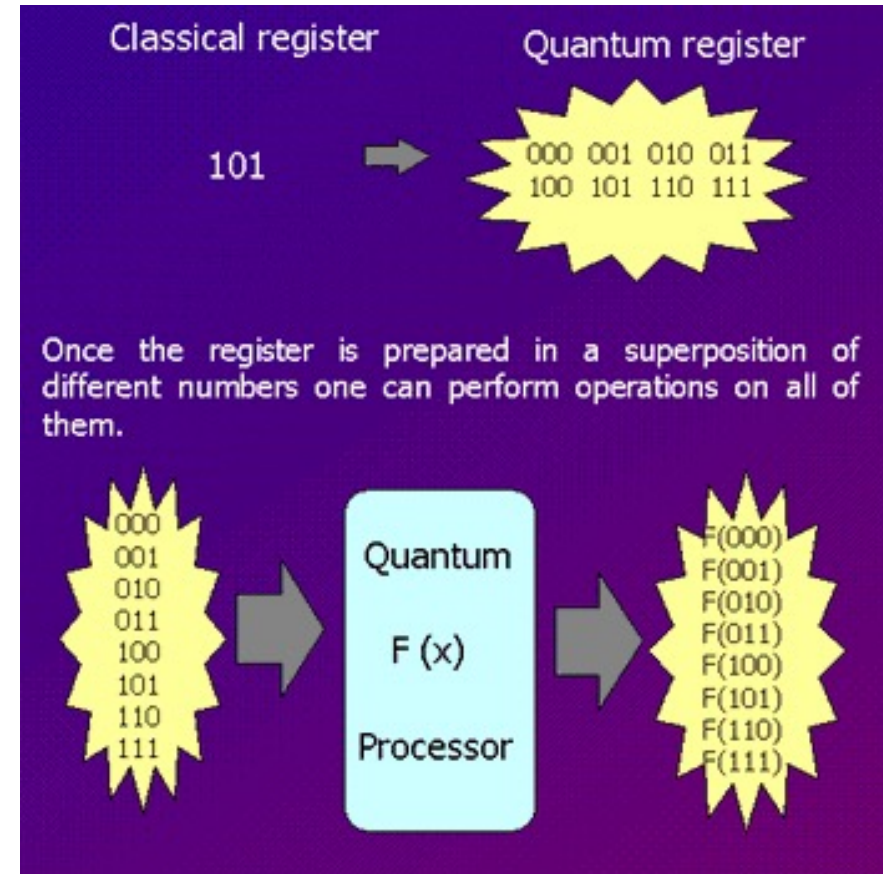
[http://en.wikipedia.org/wiki/Optical\\_computing](http://en.wikipedia.org/wiki/Optical_computing)



# New Trend: Quantum Computing



- There are several problems:
  - Decoherence: The more the qubits, the more their effect on the environment.

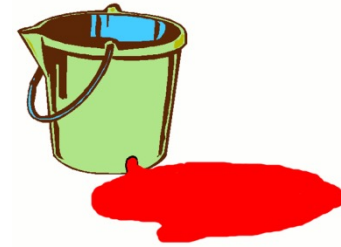


[http://media.defenseindustrydaily.com/images/PUB\\_CQC\\_Cambridge\\_Quantum\\_Computing\\_Explained\\_lg.png](http://media.defenseindustrydaily.com/images/PUB_CQC_Cambridge_Quantum_Computing_Explained_lg.png)

**Read Especially This:** [http://www.cs.virginia.edu/~robins/The\\_Limits\\_of\\_Quantum\\_Computers.pdf](http://www.cs.virginia.edu/~robins/The_Limits_of_Quantum_Computers.pdf)



# Memory Types



## ■ Dynamic Memory

- The voltages stored in capacitors die away with time.
- Solution?
  - Refresh the memory frequently; i.e., re-write the contents of the memory.

## ■ Static Memory

- A coupled transistor system stores the information.
- One of the couples *triggers* the other.

## ■ DRAM is cheaper and more widely used.

[http://wiki.xtronics.com/index.php/How\\_Memory\\_Works](http://wiki.xtronics.com/index.php/How_Memory_Works)



# Memory Types

- Volatile Memory:
  - The stored values are lost when power is off.
  - DRAM, SRAM
  
- Non-volatile:
  - Read-only Memory (ROM), flash memory

[http://wiki.xtronics.com/index.php/How\\_Memory\\_Works](http://wiki.xtronics.com/index.php/How_Memory_Works)



# Today

- More on memory
- Peripherals
- Interrupts
- Booting a computer



# Administrative Notes

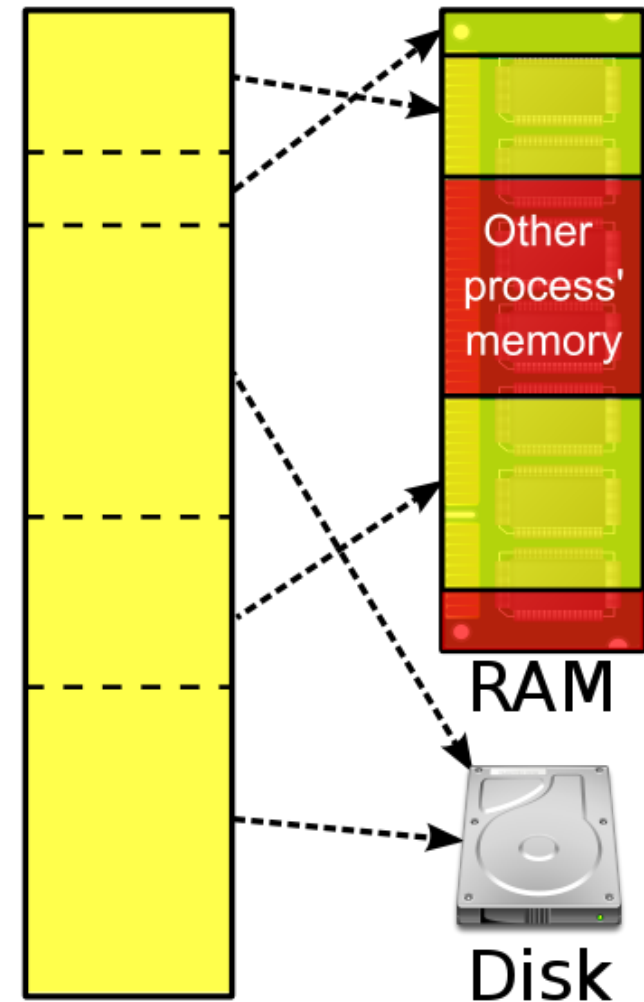
- THE1 announced!
  - Exam this Friday
- Tentative midterm date:
  - 22 December, Wednesday, 18:00

# Virtual Memory

- Memory management technique for “abstraction” over several RAM and disk storage devices.
- A program sees only “one RAM”; it does not care about the details.
- Not suitable for real-time systems.
- “Page”s are used in Virtual Memory.

Virtual Memory  
(Per Process)

Physical  
Memory

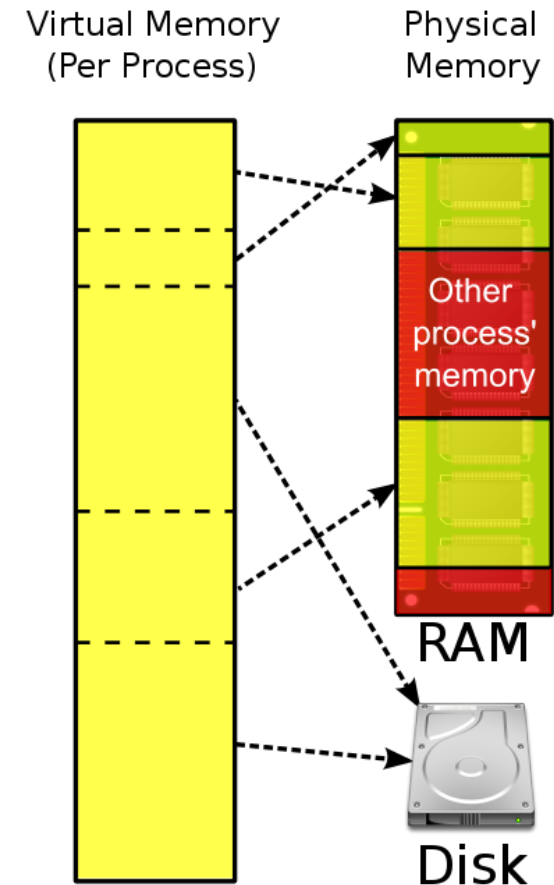




# Virtual Memory

## - Paging -

- A **page**: at least 4KB of consecutive virtual memory addresses.
- An application program is stored in a set of pages.
- A **page table** maps the **logical/virtual addresses** of the pages with the **physical addresses**.

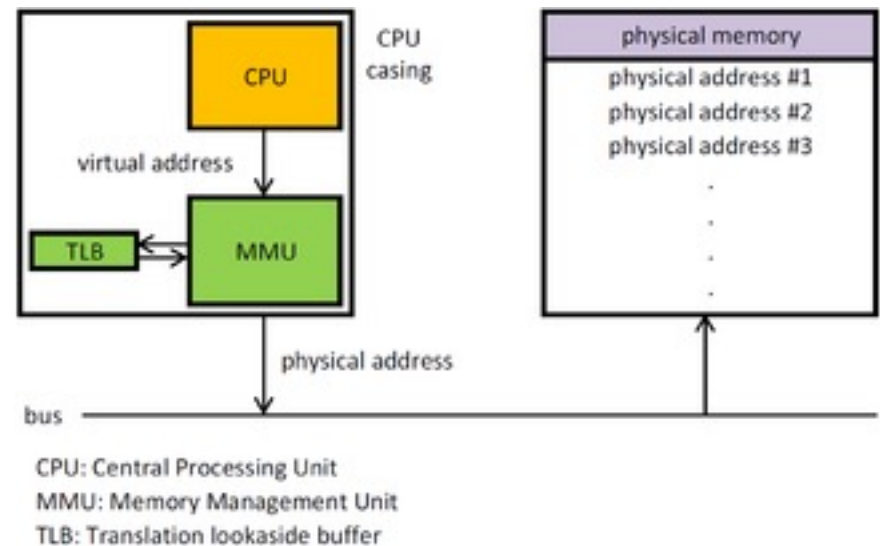


[http://en.wikipedia.org/wiki/Virtual\\_memory](http://en.wikipedia.org/wiki/Virtual_memory)



# Memory Management Unit (MMU)

- Stores a lookup table in memory or a small device called Translation Lookaside Buffer (TLB)
- Can keep track of whether a page is on RAM, in use, last-used, updated, ..
- Non-contiguous physical addresses can be mapped to contiguous virtual addresses

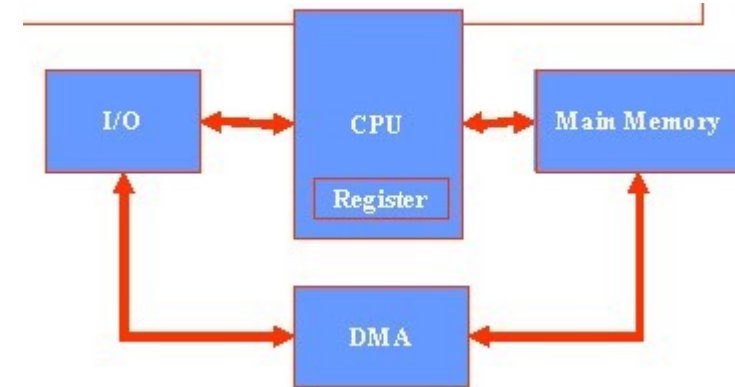
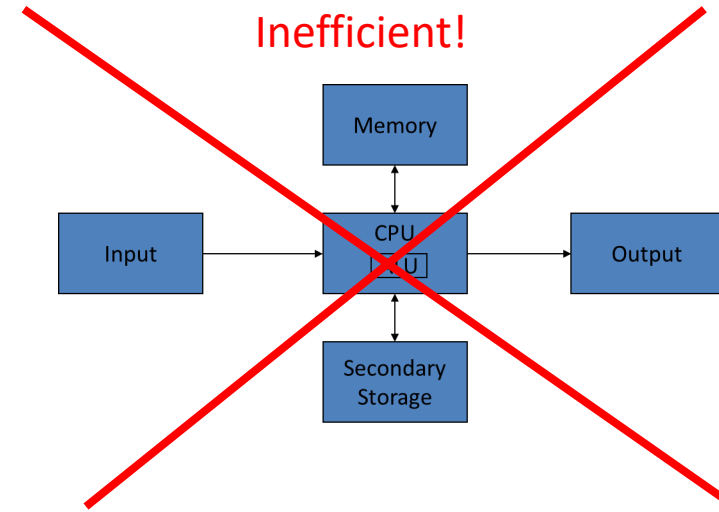


[https://en.wikipedia.org/wiki/Memory\\_management\\_unit](https://en.wikipedia.org/wiki/Memory_management_unit)



# Direct Memory Access (DMA)

- All memory accesses go over the CPU.
  - CPU gets the virtual address,
  - It looks up the physical address,
  - It writes the data to the physical device (memory, in most cases)
- This slows down the CPU for simple and consecutive memory accesses.
- “Modern” computers have a channel for DMA.
- The CPU just initiates the DMA command, and a separate unit (DMA controller) handles the copy of data between a device and a memory.





# PERIPHERALS



# I/O

## ■ Memory-mapped I/O

- Treats I/O devices as if they are memory
- “Load” and “Store” instructions are used for accessing an I/O device
- Single physical address space for memory and I/O devices
- Devices: Graphics card, hard drive

## ■ Port-mapped I/O (or isolated I/O)

- Special instructions for performing I/O
- Separate bus/pin for I/O devices
- Devices: Keyboard, mouse, ...




# Input/Output and Mass Storage

- Communication with outside world and external data storage
  - *Input unit*
    - *“Receiving” section of computer*
    - *Obtains data from input devices*
      - *Usually a keyboard, mouse, disk or scanner*
    - *Places data at disposal of other units*
  - *Output unit*
    - *“Shipping” section of computer*
    - *Puts processed info on various output devices*
      - *Screens, paper printouts, speakers*
    - *Makes info available outside the computer*



# Input/Output and Mass Storage

- Human interfaces: monitor, keyboard, mouse
- Archival storage: not dependent on constant power
- External devices vary tremendously from each other



# Input/Output and Mass Storage (continued)

## ■ Volatile storage

- Information disappears when the power is turned off
- Example: RAM

## ■ Nonvolatile storage

- Information does not disappear when the power is turned off
- Example: mass storage devices such as disks and tapes

# Input/Output and Mass Storage (continued)

- Mass (*secondary*) storage devices
  - Direct access storage device
    - Hard drive, CD-ROM, DVD, etc.
    - Uses its own addressing scheme to access data
  - Sequential access storage device
    - Tape drive, etc.
    - Stores data sequentially
    - Used for backup storage **these days**





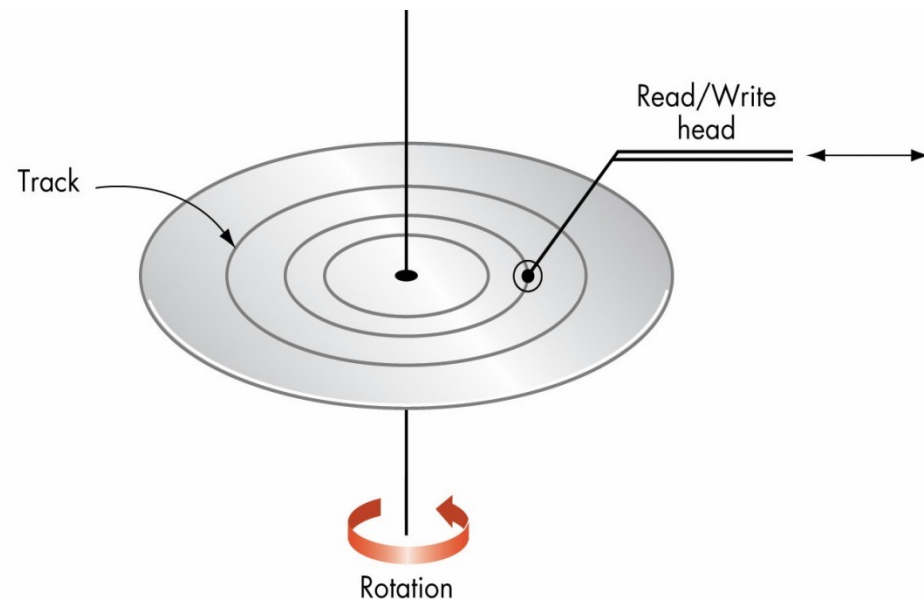
# Input/Output and Mass Storage (continued)

- *Mass or Secondary storage unit*
  - Long-term, high-capacity “warehouse”
  - Stores programs or data not currently being used by other units on *secondary storage devices* (like discs)
  - Takes longer to access than primary memory
  - Data location



# Input/Output and Mass Storage (continued)

- Direct access storage devices
- Data stored on a spinning disk
- Disk divided into concentric rings (tracks)
- Read/write head moves from one ring to another while disk spins
- Access time depends on:
  - Time to move head to correct sector
  - Time for sector to spin to data location
- We will come back to rotational storage media!





## The IBM 350

- 1956—the first hard disk drive
- 4.4 MB of storage
- Weighed over a ton
- About 1,000 systems built

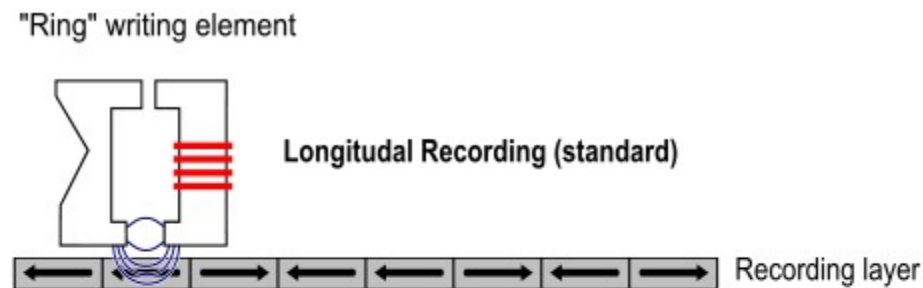
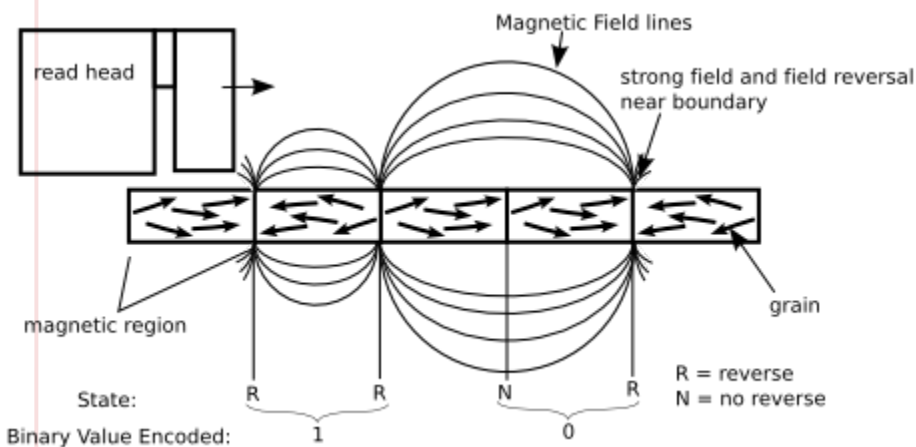
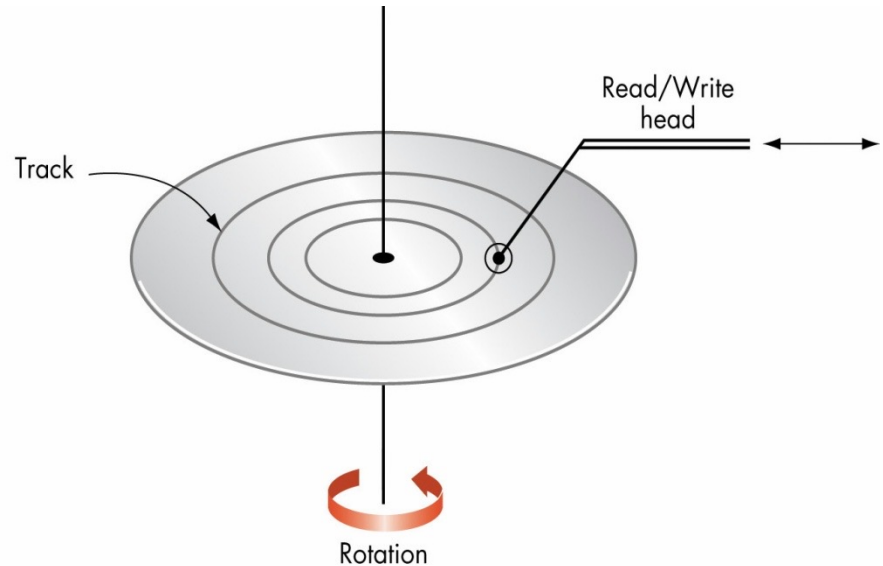
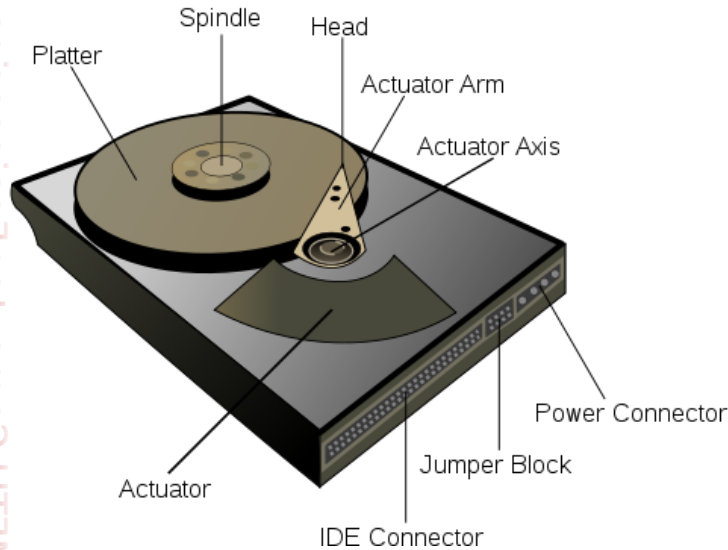


Image from [www.ichigo.se/noterat/wp-content](http://www.ichigo.se/noterat/wp-content);  
information from <http://en.wikipedia.org/wiki/RAMAC>

[http://dwave.files.wordpress.com/2007/03/20070301\\_demo\\_slides.pdf](http://dwave.files.wordpress.com/2007/03/20070301_demo_slides.pdf)



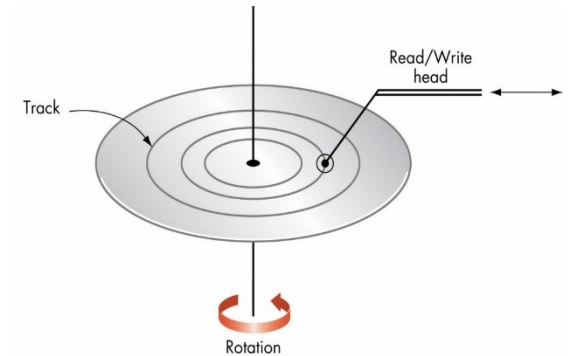
# Rotational Media as External Storage





# Rotational Drive Performance Characteristics

- For bigger disks, multiple layers & condensed layers are used.
- Access Time = Seek Time + Rotational Latency
  - Seek Time
    - “the time it takes the head to travel to the track of the disk where the data will be read or written”
    - around 9ms
  - Rotational Latency
    - “delay waiting for the rotation of the disk to bring the required disk sector under the read-write head.”
    - Depends on Rotations Per Minute (RPM)
- Data Transfer Rate
  - Covers the internal rate & the external rate
  - SATA: 3.0 Gbit/s



Note that alternative definitions of access time include data transfer rate as well. However, we will stick to this definition.

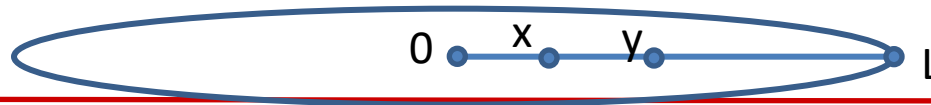
[http://en.wikipedia.org/wiki/Disk\\_drive\\_performance\\_characteristics](http://en.wikipedia.org/wiki/Disk_drive_performance_characteristics)



# One Last Bit of Information Regarding Rotational Media

- Average Seek Time
  - $x$ : track that we want to move the arm to.
  - $y$ : current position of the arm.
  - How can we model it in terms of the number of tracks?
- Average Seek Time = average distance  $|x - y|$  multiplied by the time to move the arm by one track.
- The average distance =  $L/3$ . How?

$$\text{Average Distance} = \frac{1}{L^2} \int_0^L \int_0^L |x - y| \, dx dy = L / 3$$





# Transfer of Information

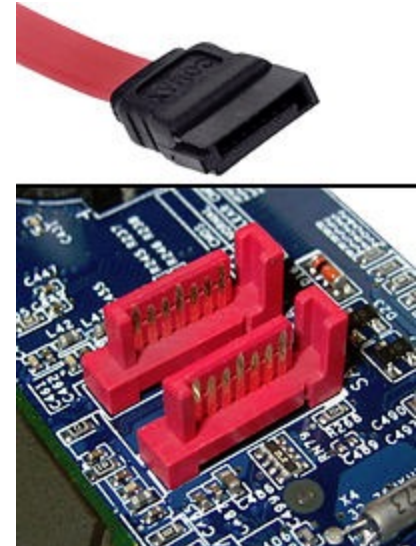
## The Concept of Ports

- What is a port, literally?
- What is a computer port, then?
  - SATA, e-SATA, USB, Firewire, Serial, Parallel (e.g., PS/2) ports.



# Transfer of Information In/On the Computer

- SATA – Serial Advanced Technology Attachment:
  - Bus interface between motherboard and storage devices, using high-speed cable.
  - Several versions (protocol & encoding is different):
    - SATA1: 1.5Gbit/s
    - SATA2: 3.0Gbit/s
    - SATA3: 6.0Gbit/s
- Alternatives:
  - IDE
  - PATA – Parallel Advanced Technology Attachment







# Transfer of Information In/On the Computer

- Firewire – IEEE 1394 Interface
- High-Definition Audio-Video Network Alliance standard interface for audio/video communication.
- Several versions exist.
  - Around 800Mbit/s.
- Alternative:
  - Universal Serial Bus (i.e., USB)







# Transfer of Information In/On the Computer

- Universal Serial Bus – USB
- Alternative to Firewire
  - Minor differences exist
- In addition to communication, USB bus can provide power to the device.
- Several versions exist: 1.0, 2.0, 3.0 and 4.0.
  - 2.0 is the current widely used standard.
  - 3.0 is available ~~but still not widely used.~~
  - 4.0 is becoming popular with the USB-C connector.





# Transfer of Information Between Computers

- Ethernet
  - For inter-net as well as intra-net.
- Different versions:
  - 100Mbit/s
  - 1Gbit/s
- Ones that are not widely supported yet:
  - 10Gbit/s
  - 100Gbit/s

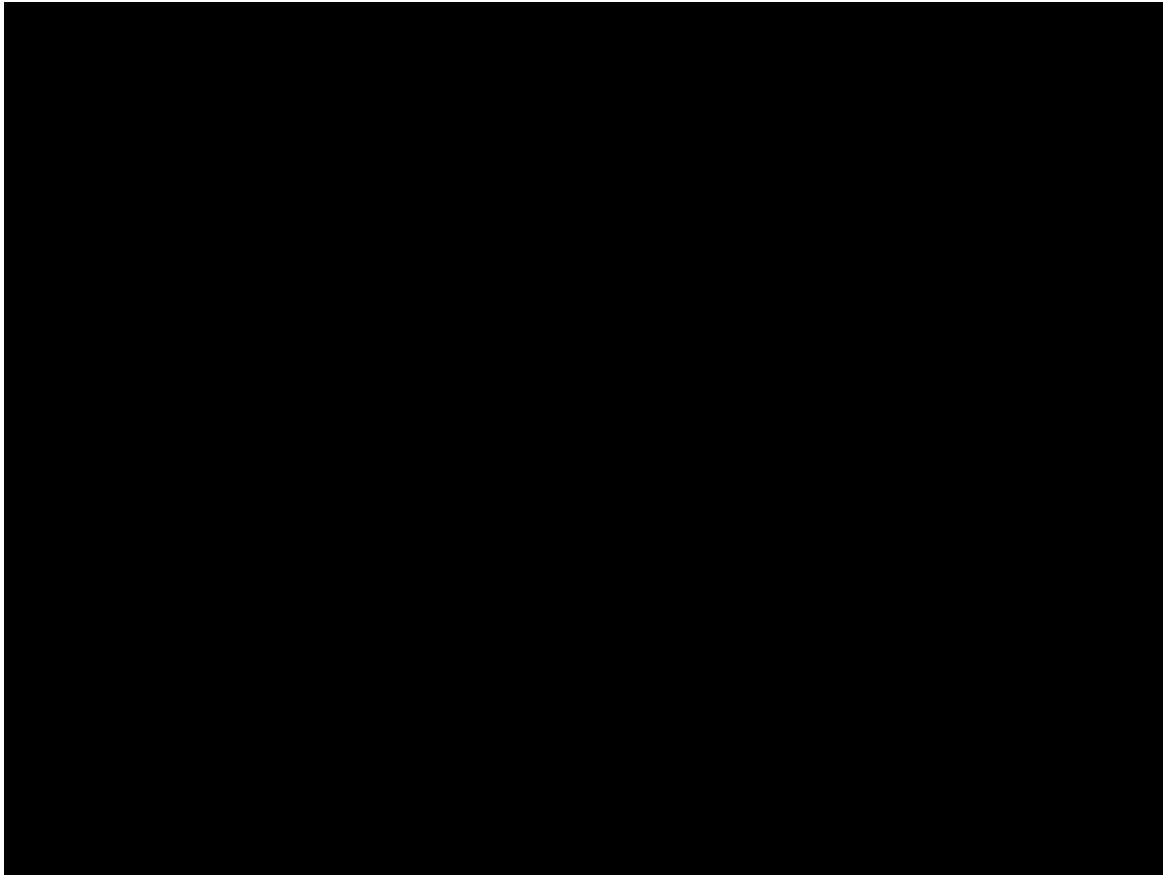




# INTERRUPTS



# “Are we there yet?”





# Interrupts

- When you type a character at the keyboard, the keyboard controller transmits the character to the CPU.
- However, the CPU is busy all the time and the keyboard controller has a limited storage; usually, only for one character.
- How to get the CPU's attention?
- Polling vs interrupt.
  - Polling: a phone without a bell.

[http://www.atarimagazines.com/compute/issue149/60\\_Interrupts\\_made\\_easy.php](http://www.atarimagazines.com/compute/issue149/60_Interrupts_made_easy.php)



# Interrupts

- I/O, time management, power-off signals, traps, ...
- When an interrupt is received,
  - CPU stops the current task/program.
  - It saves the current “context”. **How?**
  - It executes the code necessary for the received interrupt.



# Interrupt Table

- After booting, devices and operating systems register service routines that tell the computer “what to do” on different interrupts.
- The addresses of the routines that will handle the interrupts are stored in the interrupt table.

Interrupt vector	Description
00h	CPU: Executed after an attempt to divide by zero or when the quotient does not fit in the destination
01h	CPU: Executed after every instruction while the trace flag is set
02h	CPU: NMI, used e.g. by POST for memory errors
03h	CPU: The lowest non-reserved interrupt, it is used exclusively for debugging, and the INT 03 handler is always implemented by a debugging program
04h	CPU: Numeric Overflow. Usually caused by the INTO instruction when the overflow flag is set.
05h	Executed when Shift-Print screen is pressed, as well as when the BOUND instruction detects a bound failure.
06h	CPU: Called when the Undefined Opcode (invalid instruction) exception occurs. Usually installed by the operating system.
07h	CPU: Called when an attempt was made to execute a floating-point instruction and no numeric coprocessor was available.
08h	<b>IRQ0: Implemented by the system timing component; called 18.2 times per second (once every 55 ms) by the <a href="#">PIC</a></b>
09h	<b>IRQ1: Called after every key press and release (as well as during the time when a key is being held)</b>
0Bh	<b>IRQ3: Called by <a href="#">serial ports</a> 2 and 4 (COM2/4) when in need of attention</b>
0Ch	<b>IRQ4: Called by serial ports 1 and 3 (COM1/3) when in need of attention</b>
0Dh	<b>IRQ5: Called by hard disk controller (PC/XT) or 2nd <a href="#">parallel port</a> LPT2 (AT) when in need of attention</b>
0Eh	<b>IRQ6: Called by <a href="#">floppy disk controller</a> when in need of attention</b>
0Fh	<b>IRQ7: Called by 1st parallel port LPT1 (printer) when in need of attention</b>





# Timer Interrupt

- A system timer generates interrupts on every “tic”
- Called 18.2 times every second; once every 55ms.
  - Programmable
- Why is it needed?
  - Memory refreshing
  - OS time management
    - OS reads the initial time when booted
    - It increments the initial value with every time interrupt



# Masking Interrupts

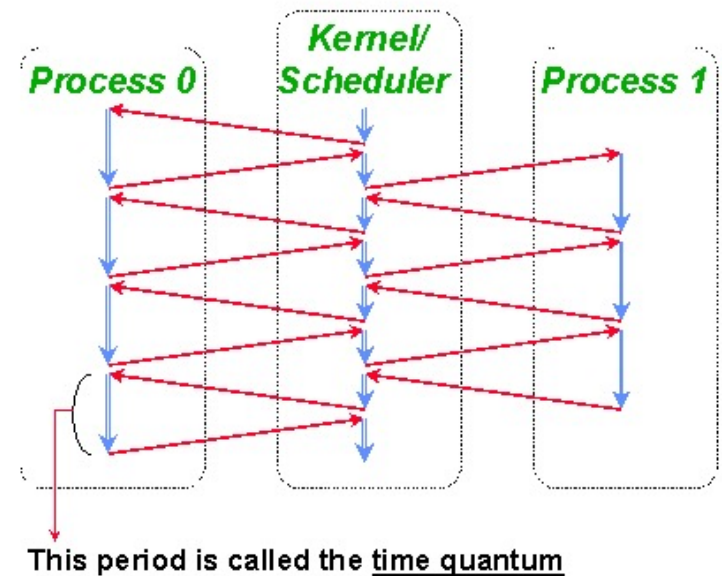
- Disabling interrupts for a limited time.
- When is it needed?
  - Busy, handling an important interrupt already
  - Being interrupted by the interrupt being handled.

[http://en.wikipedia.org/wiki/Interrupt\\_mask\\_register](http://en.wikipedia.org/wiki/Interrupt_mask_register)

# Interrupts & Multi-tasking

- Most of the time, there are more than one programs/processes running.
- Each program runs for a fixed amount of time  $T$ .
  - $T$  depends on the priority of the program/process.
- After  $T$  is finished, the CPU “interrupts”.

## Timesharing/Process Scheduling



<http://6004.csail.mit.edu/Spring98/Lectures/lect21/sld009.htm>