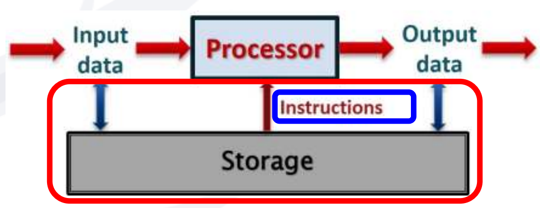
ENS1161 Computer Fundamentals
Module 6
Storage

Moving forward..

- Last module:
  - Programming languages
  - Programming 'tools' that convert programs to binary code
- Focus of this module: Storage
  - Types of storage
  - How data and instructions are moved from storage to the processor and back



Module Objectives

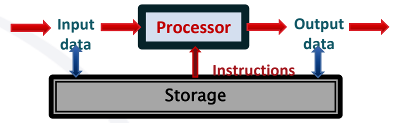
On completion of this module, students should be able to:

- Explain the differences between primary and secondary storage.
- Describe the different types of primary and secondary storage technologies covered and their principles of operation.
- Explain the how the characteristics and specifications of different storage technologies impacts their usage in computer systems.
- Evaluate different types of storage based on their specifications and determine the most suitable for a given application.

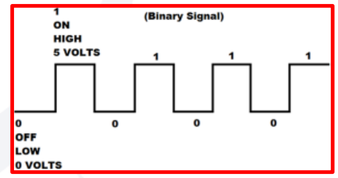
Introduction

- **Module Scope**
  - The role of storage in a computer system
  - Types of storage
    - Classification based on usage and technology
  - How different storage devices function
  - Pros and cons of different storage media

Data & Instructions *(recap)*

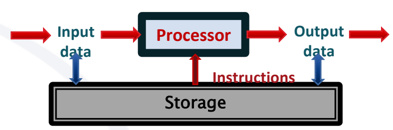


- Computer systems are digital systems
  - All **data** and **instructions** are in the form of binary signals
    - signals that have only two possible values
      - **1** or **0**
      - HIGH or LOW
      - ON or OFF

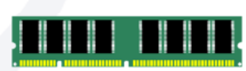


- These digital signals may be used to represent:
  - one bit of a binary number
  - one bit of a binary code
    - ASCII,BCD, instruction code, …
  - a control signal state, etc.

Storage *(recap)*



- Components used to 'hold' the binary data
  - The hardware used to store the software
- **Primary storage / memory**
  - Memory directly accessible by the processor
    - E.g. RAM, ROM, cache memory
    - Fast access, but normally *volatile*
      - Data disappears when power goes off



- **Secondary storage**
  - Devices that can store data more permanently (even when power off)
  - E.g. hard disk, flash drive, CD/DVD, etc.

Storage

Primary Storage

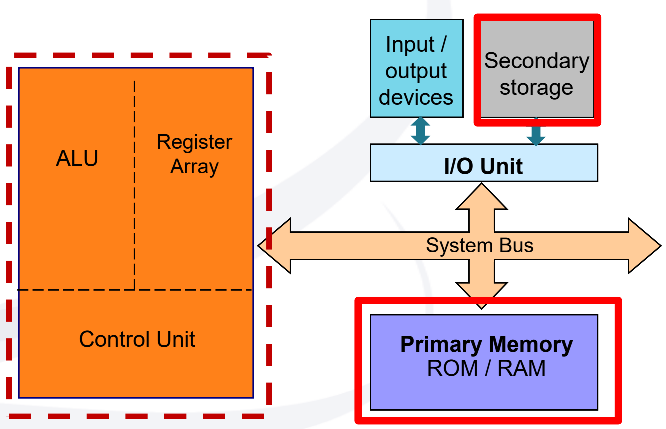
- Sometimes called **main memory** (or just *memory*)
- Storage that is directly accessible by the CPU
  - Connected directly via system bus
- The 'working storage' for the CPU
  - Currently running programs and data stored here
- Fast access, but limited capacity
- Most of it (RAM) loses data when powered off (*volatile*)

Secondary Storage

- Storage accessed indirectly
  - Connected via I/O interface
- Mainly for longer term storage of data
- Can also be used for temporary storage of memory data (*swap space*)
- Large capacity, but slower access
- Data preserved even when powered off (*non-volatile*)
- Could be physically remote from CPU
- Cheaper (lower cost per byte of data)

Basic Components of a Computer *(recap – Module 2)*

- Every computer contains the same basic components:
  - Arithmetic logic unit (ALU)
  - Register array
  - Control unit
  - Memory
  - Input/Output (I/O) unit
  - System Bus



'Processor'

Storage in a computer

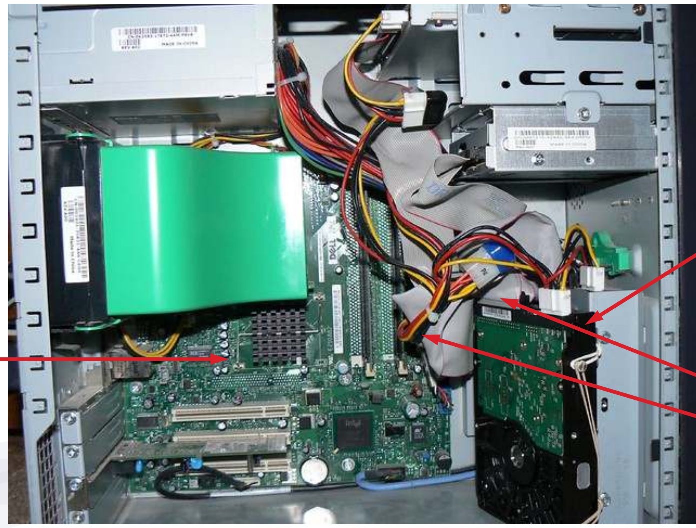
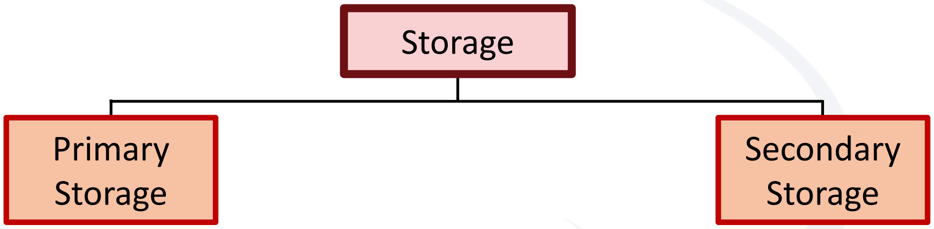


Image: Nick Ares, 2007

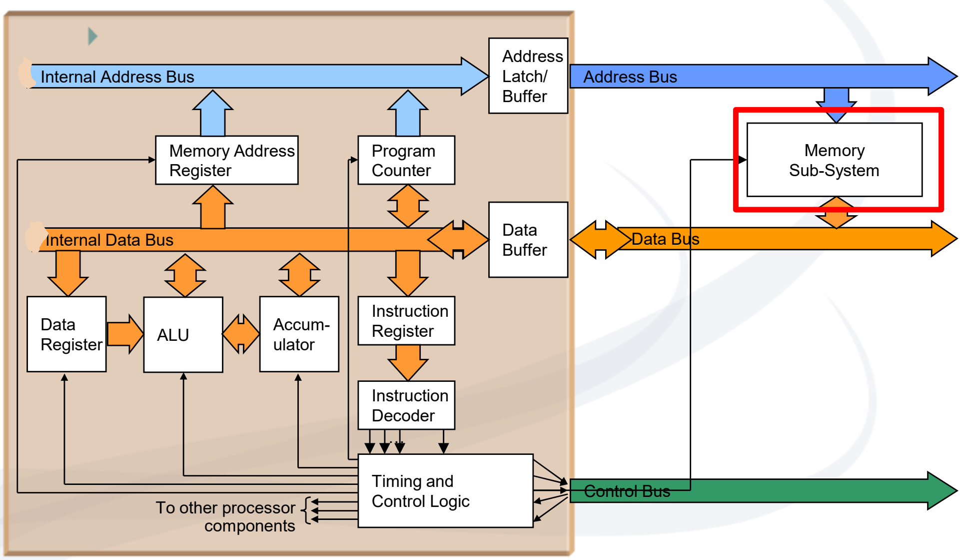
Storage speed vs capacity / frequency of use

- Trade-off in storage: speed vs capacity
- Higher speed storage, lower capacity
  - Mainly due to cost per byte
  - Storage with lower access times tend to be more expensive
- Speed is also normally inversely proportional to distance
  - Further away storage is, the slower it is
  - Due to greater distance, more complex interface
- Data to be used (more frequently) is transferred to faster ('closer') storage
  - E.g. files stored in secondary storage are transferred to RAM for CPU to access
  - This principle is also used in other mechanisms *(to be covered)*



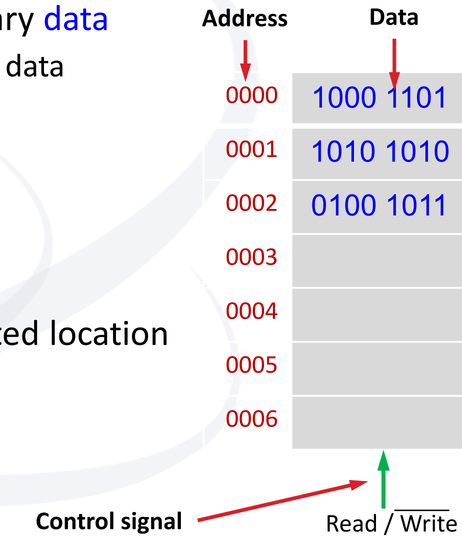
- Sometimes called **main memory** (or just *memory*)
- Storage that is directly accessible by the CPU
  - Connected directly via system bus
- The 'working storage' for the CPU
  - Currently running programs and data stored here
- Fast access, but limited capacity
- Most of it (RAM) loses data when powered off (*volatile*)

Simplified Diagram of a Processor *(recap)*



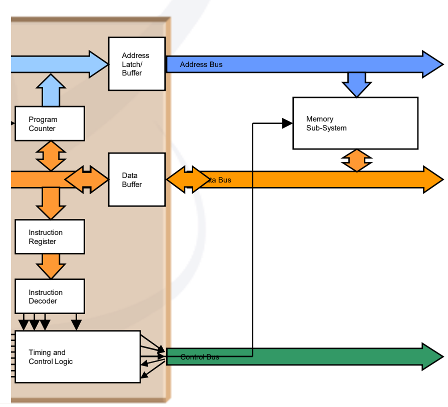
Memory *(recap – Module 2)*

- Multiple storage locations, each containing binary data
  - Generally, each location contains 8 bits (1 byte) of data
  - Could contain multiple bytes (e.g. 16 bits, 32 bits)
- Each location has an address
  - Used to specify location to use
  - Has to be set first so 1 location is selected
- Data can be read from or written into the selected location
  - Depends on the control signal
  - For read, data in memory location → data bus
  - For write, data on data bus → memory location



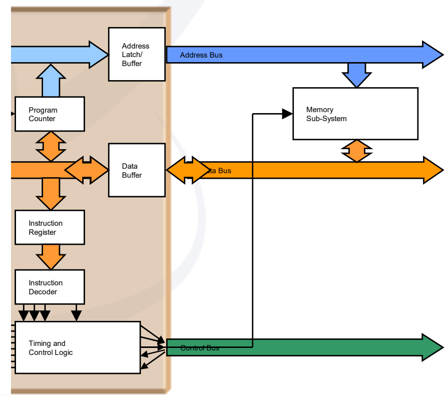
Address bus *(recap – Module 2)*

- Used to select locations within the processors addressable space for reading or writing data
- The wider (more bits) the address bus, the greater the addressable space
  - ***2^N* addresses from an *N*-bit address bus**
- The address bus is unidirectional
  - From the processor to memory or device
  - No information is read from it



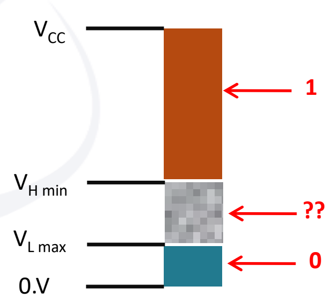
Data bus *(recap – Module 2)*

- Used to transfer data to and from memory (or other device)
- All information in the system is transferred through this bus
  - data, program instructions, operand addresses, etc.
- Width of the data bus depends on processor
  - Normally the number of bits processor can process at one time
  - The wider the bus, more data can be transferred at one go
- The data bus is bidirectional
  - READ operation: data from *memory location specified on address bus* → processor
  - WRITE operation: data from processor → *memory location specified on address bus*

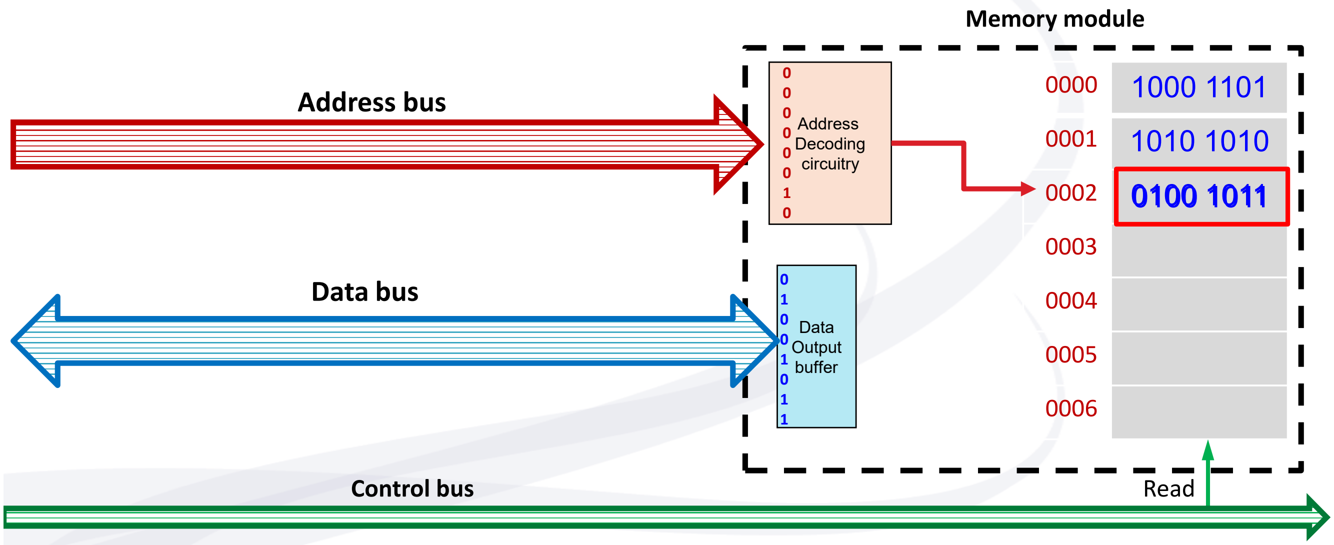


Logic levels

- Voltages are used to represent 0 and 1
  - 0 V nominally used to represent a ‘0’
  - V_{cc} is used to represent ‘1’
    - V_{cc} = the device power high voltage
      - Typically 5V, but many mobile devices use 3.3V or lower
- In reality, a range of voltages can be accepted as a ‘1’ or ‘0’
  - Range depends on the particular technology used
  - $V_{H\ min}$ = lowest voltage that will be read as a ‘1’
  - $V_{L\ max}$ = highest voltage that will be read as a ‘0’
  - In between is an ‘indeterminate’ range
    - Devices may read them as ‘1’ or ‘0’ (unpredictable)
    - Should not attempt to read when in this range



Memory read sequence



Memory access time

- Memory read takes a number of steps from the time the address and read signal are sent till the data is available
  - Allow address signals (voltages) to ‘settle’ (stop changing)
  - Decode the address
  - Select the correct memory location
  - Transfer data to data buffer that is connected to the data bus
  - Enable data onto the bus
  - Allow data signals (voltages) to ‘settle’ (stop changing)
- Each step takes a finite amount of time
- This is called the ***access time*** for the memory
  - Normally in the order of 10 – 150 ns (nanoseconds)

- Storage
  - Primary Storage
    - ROM
    - RAM
  - Secondary Storage

ROM

- ***R****ead* ***O****nly* ***M****emory*
- Non-volatile memory
- Can’t be written to in normal operation
- Normally used for startup code and low-level functions

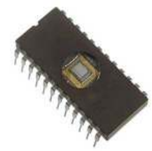
RAM

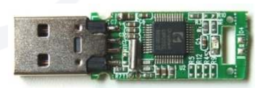
- ***R****andom* ***A****ccess* ***M****emory*
- Can be read from and written to
- Should actually be called *Read Write Memory*!
- Volatile memory – data disappears when no power
- The main working memory

- Storage
  - Primary Storage
    - ROM
      - Mask ROM
      - PROM
      - EPROM
      - EEPROM
        - Flash memory
    - RAM
  - Secondary Storage

Types of Read Only Memory (ROM)

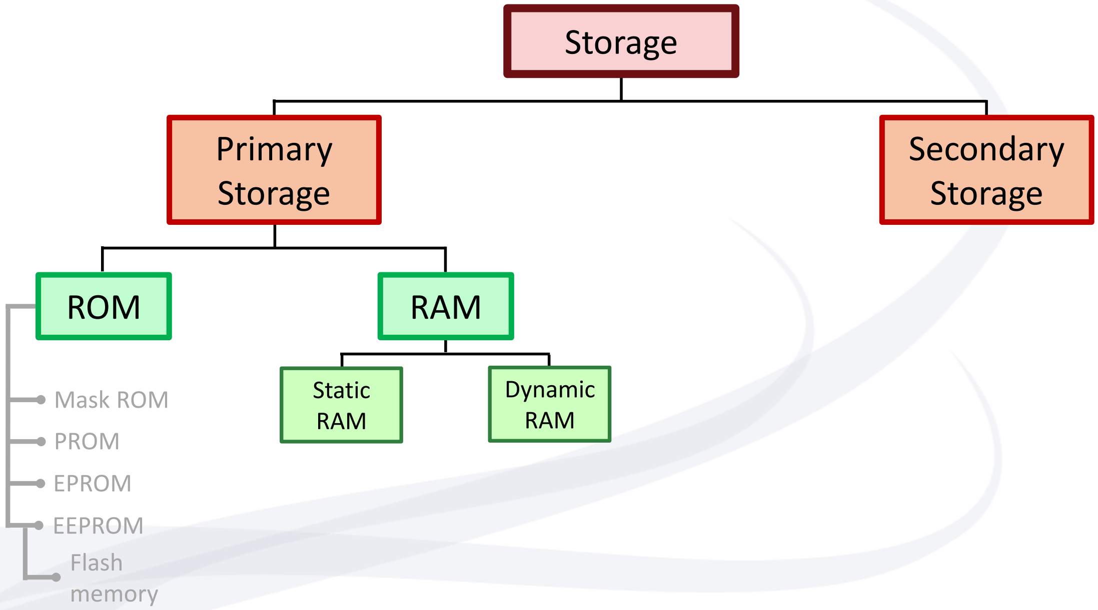
- Mask ROMs – or just ROM
  - Data stored is built into chip at manufacture using a 'mask'
- PROMs – (One-time) Programmable ROM
  - Data is written in by 'fusing' (burning) connections
- EPROM – Erasable Programmable ROM
  - Data can be cleared by exposing chip to UV light
  - Can then be reprogrammed
- EEPROM – Electrical Erasable PROM
  - Can be re-programmed *in-circuit*
  - Disadvantage – low density, higher cost
- Flash Memory – Higher density EEPROM
  - Designed for large block erase and writes





Firmware

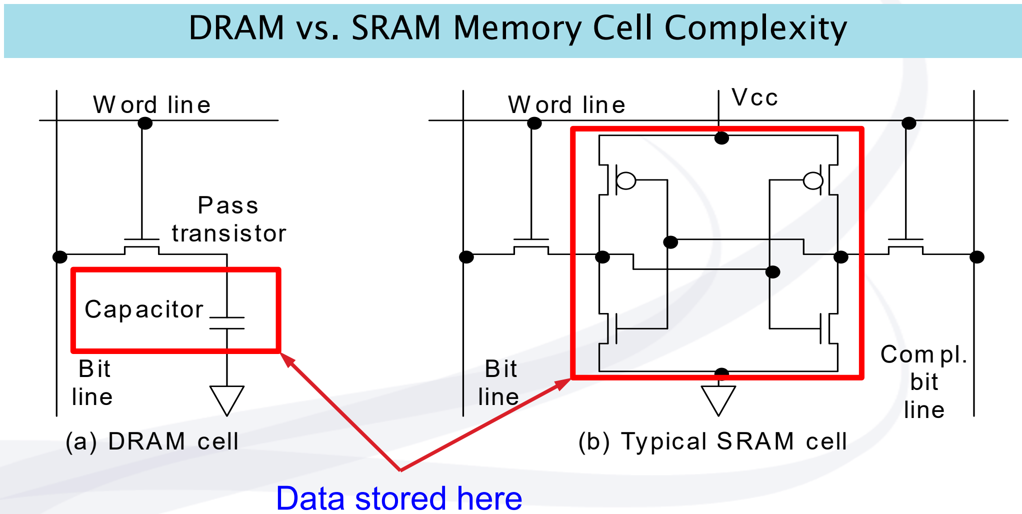
- Software embedded in hardware (ROM)
- The most common application of ROMs
  - E.g. ROM-BIOS (*Basic Input Output System*)
    - Data and program code needed on power-up of computer systems
    - Instructions to initialise the system and invoke an operating system from auxiliary memory
      - *bootstrapping* or *'booting'*
      - Covered in more detail in *Modules 8 and 9*
    - Also low-level functions to handle I/O based on hardware in the system
  - EEPROM or Flash Memory used to allow the firmware to be upgraded if necessary



Types of RAM

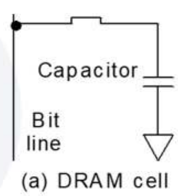
- 2 main types of RAM used in computer systems:
  - **Static RAM (SRAM)**
    - Stores data in logic circuits (flip-flops)
    - Can stay in a given state indefinitely as long as there is power
    - Fast and simple implementation, but space and power inefficient
  - **Dynamic RAM (DRAM)**
    - Data stored as charge on small MOS capacitors
    - Requires periodic recharging - due to leakage of capacitors
    - More complex - needs refresh circuitry
      - Data read, and rewritten in
    - But can be much higher density and power requirements are much lower

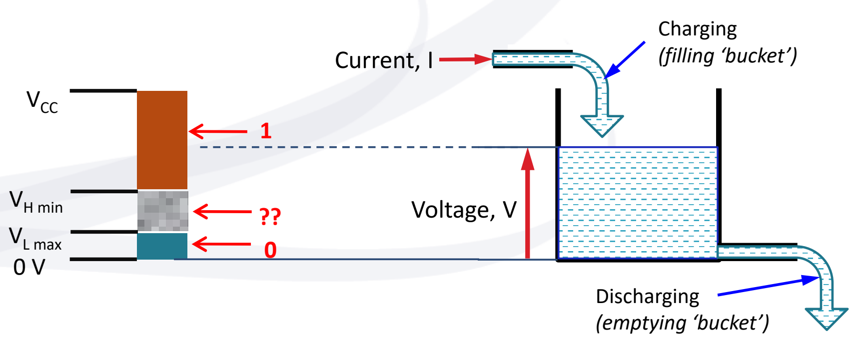
DRAM vs SRAM



DRAM – principles of operation

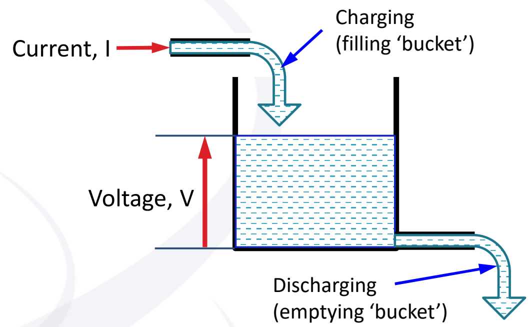
- Essentially just a capacitor
  - Acts as a 'bucket' that stores charge
  - Electric current = flow of charge in or out
  - Voltage depends on 'level' of charge in the bucket
  - Whether cell contains '0' or '1' depends on the voltage

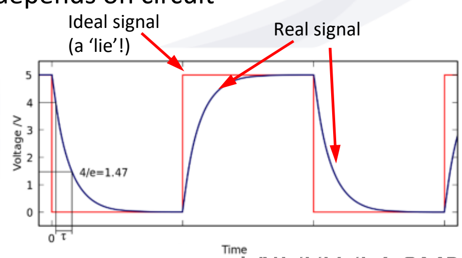


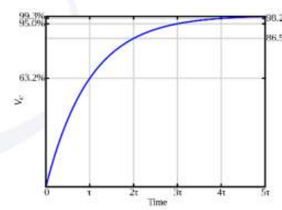


DRAM – Limitations

- **Time to change data**
  - ‘Bucket’ needs time fill up / empty
  - Physically impossible to change voltage instantly
  - Voltage change follows exponential curve
  - This physical limitation applies to all signals
    - Data or addresses
    - Though speed of change depends on circuit
      - ‘Size of bucket’



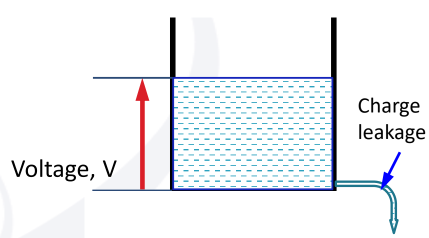




Capacitor Charging

DRAM – Limitations

- **Charge leakage**
  - Capacitors are not perfect
  - Charge can leak out
  - Voltage will therefore drop
  - Data may ‘fade away’
  - Solution: *Refresh circuitry*
    - Special circuit that periodically checks each cell and ‘refreshes’ the data
    - Charges it up if a ‘1’ or discharges fully if a ‘0’
      - ‘Refill’ or ‘empty bucket’ accordingly

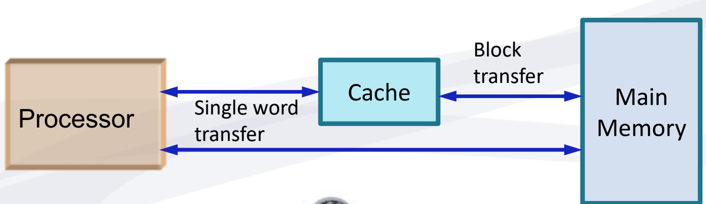


Processor speed vs RAM speed

- A single processor instruction may require multiple memory access
  - 1 instruction read (minimum)
  - 1 or more data read or write
    - Depending on type and complexity of instruction
    - *Refer Module 2*
- Processor clock speeds have increased tremendously
  - E.g. Intel i7 processor standard clock speed = 3.7 GHz
    - i.e. 1 period of clock = 0.27 ns (*nanosecond*)
- RAM access speeds much slower
  - DRAM typically 60 ns
  - SRAM typically 10 ns
- While DRAM is cheap and compact, speed can be a bottleneck

Cache Memory

- Small amount of faster memory that sits 'between' CPU and main memory
  - Stores copies of frequently used data and instructions
  - To improve performance of primary storage
- If data required is in cache (*hit*), processor will read data from there
- If not in cache (*miss*), will then go to main memory to retrieve the data
- Static RAM used for cache memory
  - Bulkier, more expensive but faster



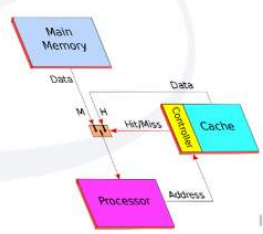
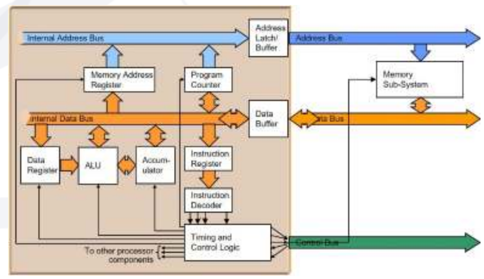


Image: Ferruccio Zulian, Italy

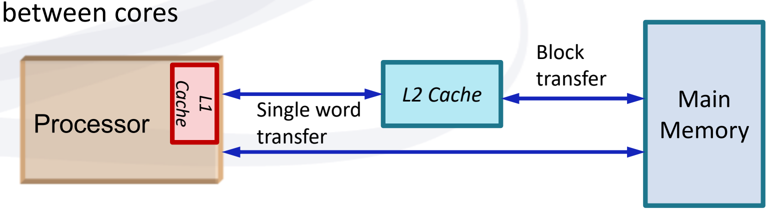
Internal vs External buses *(recap Module 2)*

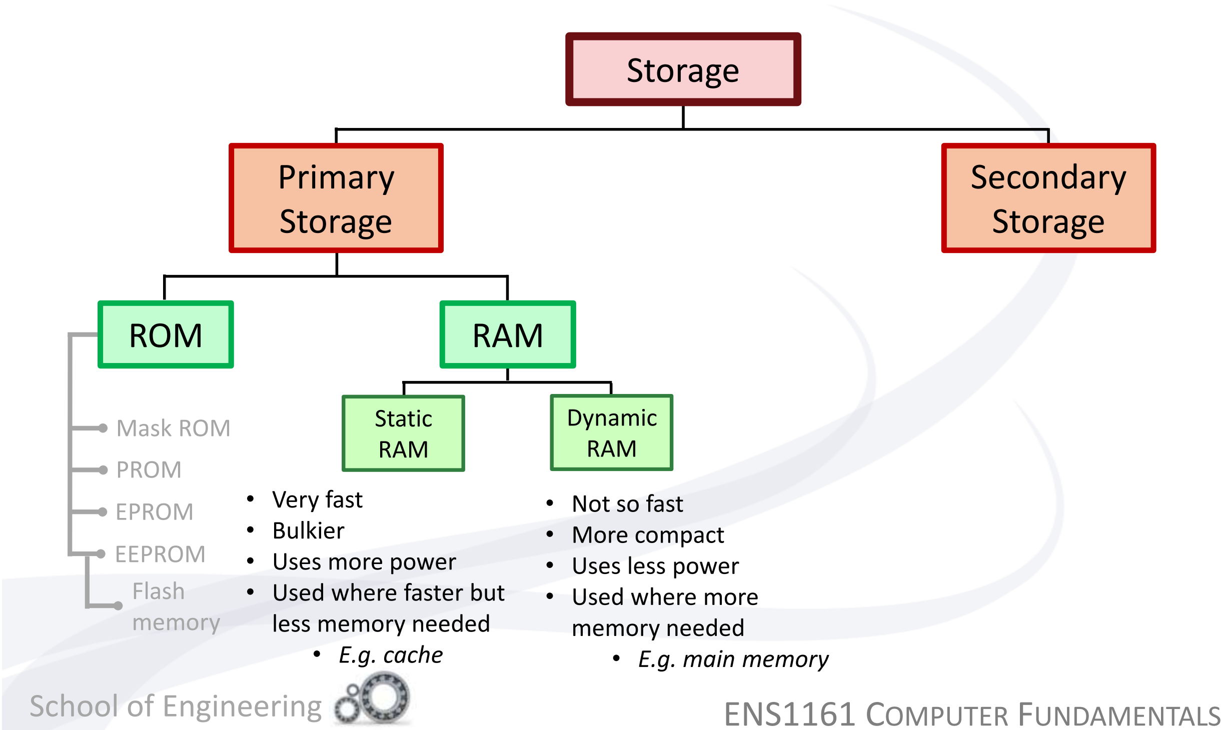
- The *internal* address and data buses connect the processor components within the processor chip
  - Registers, ALU, etc.
- The *external* data bus connects the processor chip to external components
  - Memory, I/O devices
- Internal bus transfers are much faster because:
  - Transfers are within chip
    - Shorter distance, less interference
  - On chip components are faster
    - E.g. registers have a much quicker response time compared to external memory



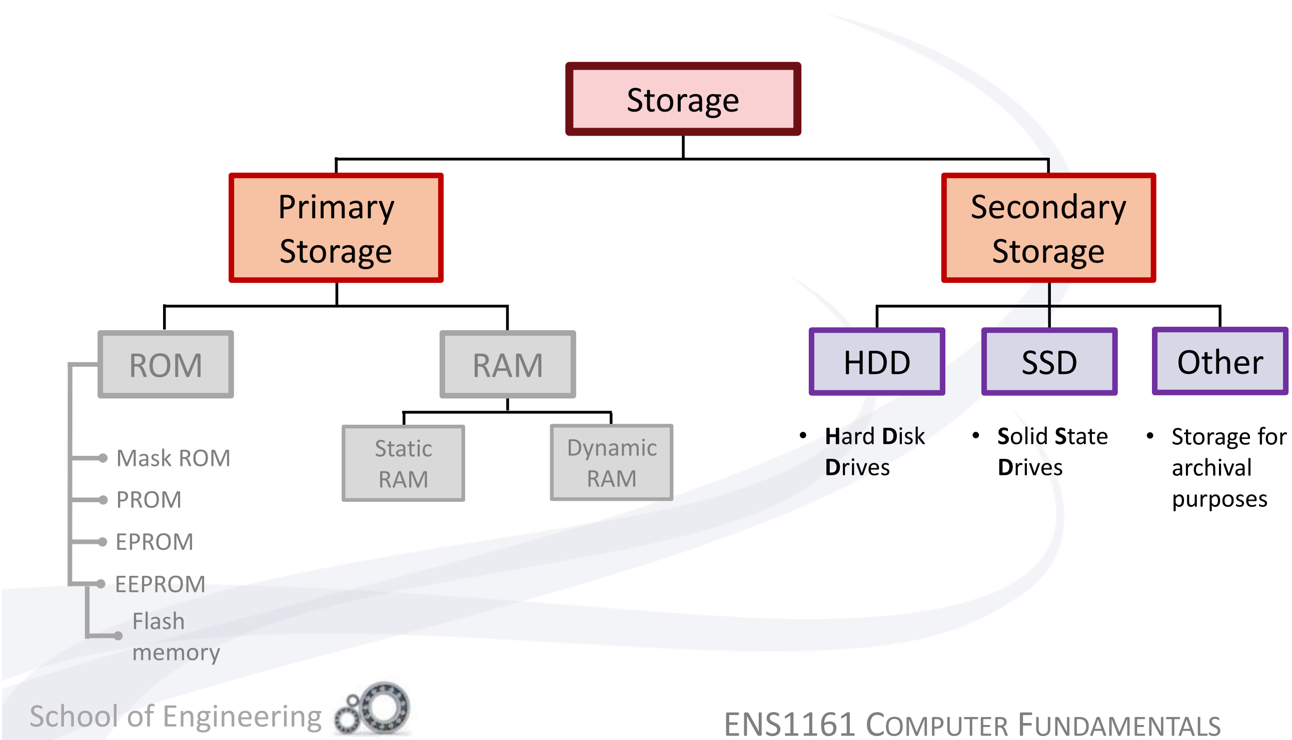
On-chip Cache

- L1 cache is an *additional level* of cache memory
- Very fast (but small) cache memory that is *within* the processor chip
  - Takes advantage of the speed of internal transfers within chip
  - Previously described cache memory called *L2 cache*
- Needed as disparity between processor and memory speeds grew larger
- Evolving terminology (for multi-core processors):
  - L1 cache: On-chip cache for 1 core
  - L2 cache: On-chip cache shared between cores
  - L3 cache: Off-chip cache





Storage
Primary Storage
Secondary Storage
ROM
RAM
Mask ROM
PROM
EPROM
EEPROM
Flash memory
Static RAM
Dynamic RAM
• Very fast
• Bulkier
• Uses more power
• Used where faster but less memory needed
• E.g. cache
• Not so fast
• More compact
• Uses less power
• Used where more memory needed
• E.g. main memory
School of Engineering
ENS1161 COMPUTER FUNDAMENTALS



Storage
Primary Storage
Secondary Storage
ROM
RAM
Mask ROM
PROM
EPROM
EEPROM
Flash memory
Static RAM
Dynamic RAM
HDD
SSD
Other
• Hard Disk Drives
• Solid State Drives
• Storage for archival purposes
School of Engineering
ENS1161 COMPUTER FUNDAMENTALS

Hard Disk Drives (HDD)

- Data is stored on metallic platters ('hard disks')
  - Platters are coated on both sides with magnetic oxide
  - The magnetic 'orientation' of material changed to represent 1s or 0s
- Heads read the 1's and 0's from the platter
  - Or write them to the platter
  - Heads for top and bottom of each platter
- Platters spin at high speed
  - Typically 5,400 rpm or higher
- Actuator arm moves the heads together radially
- Allows any spot to be accessed

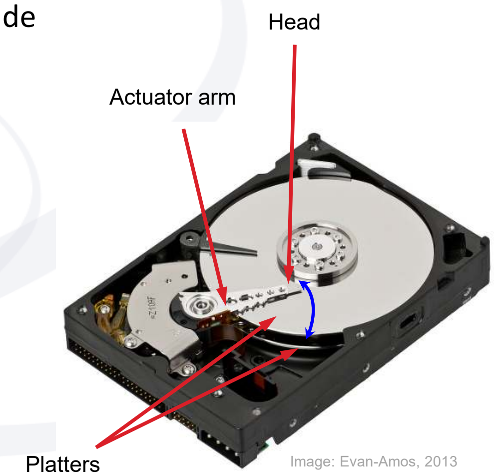


Image: Evan-Amos, 2013

HDD data layout

- Data on hard drives is organised in *tracks*
- Tracks are subdivided into blocks called *sectors*
  - Normally 512 bytes
- Operating systems allocate space in clusters
  - Groups of sectors
- Moving the heads from track to track takes time
  - Slows down read / write time
- So data is normally spread across the same track on the different surfaces of the platters
  - Blocks of data send to different heads
- This combination of tracks is called a *cylinder*

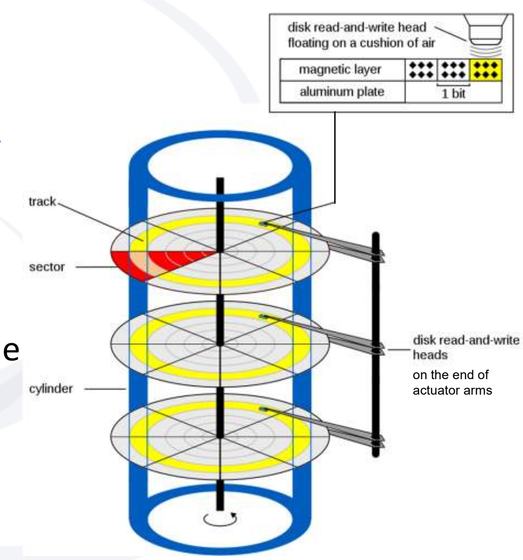


Image: Wimox, 2019

HDD pros and cons

- Access time is relatively long
  - 4 to 20 ms (*milliseconds*), depending on drive
  - Seek time in addition to normal access delays
  - *Seek time* = time to get head positioned over data location
    - Time to *move head to track*
    - + time for *disk to spin required sector under head*
- Possibility of mechanical failure
  - e.g. *head crash* – head hits platter due to mechanical shock
- Power consumption and noise
  - Because platters always rotating
- Cheap
  - Very low cost per byte
- Mature technology
  - Reliability, speed, noise and power consumption has been improved over time

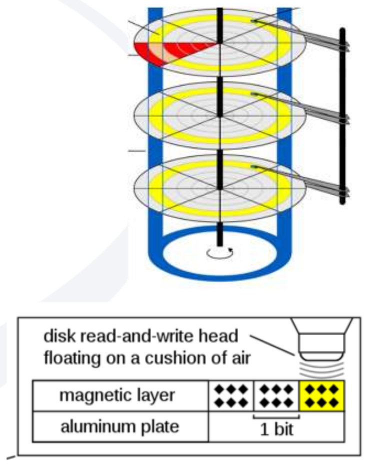
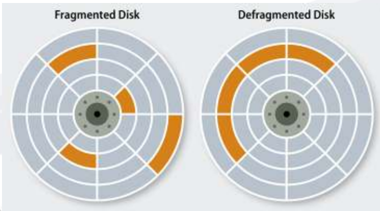


Image: Wimox, 2019

Disk Fragmentation

- As files are added and deleted from disk, and files on disk grow and shrink as they are updated, the files become fragmented
  - Occupy non-contiguous clusters
- Increases the read time as heads will need to move around from track to track
- May require defragmentation routines to be run to improve performance
  - Move data around on disk so that they lie on contiguous blocks

Image: Enterprise Storage Forum, 2019



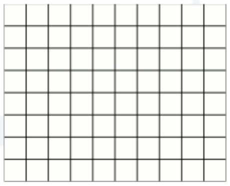
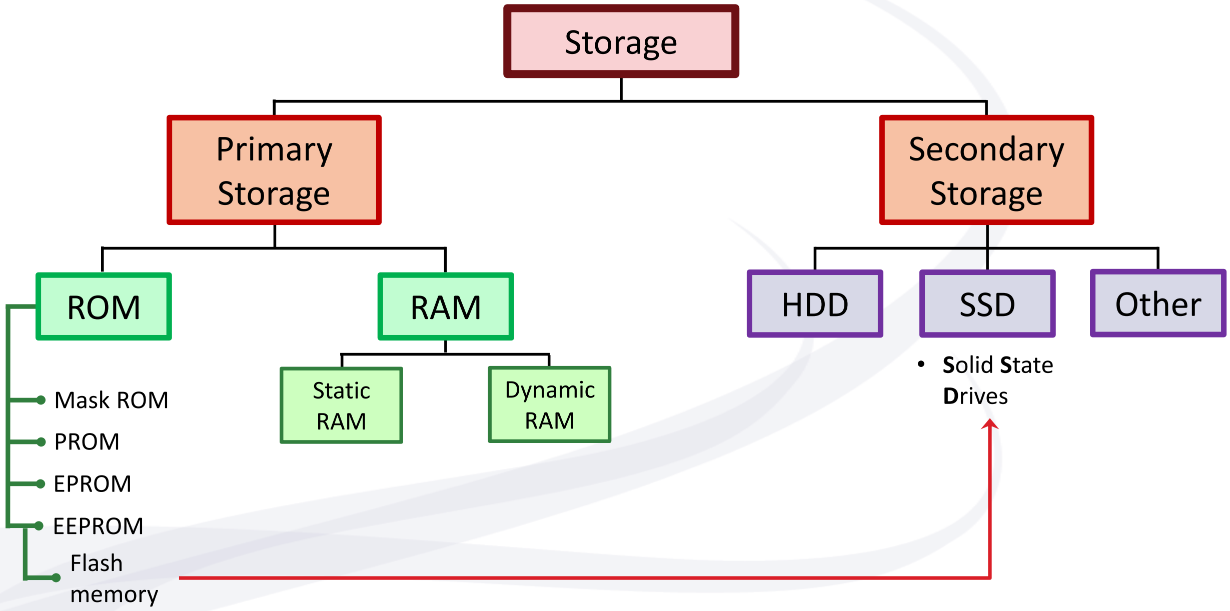


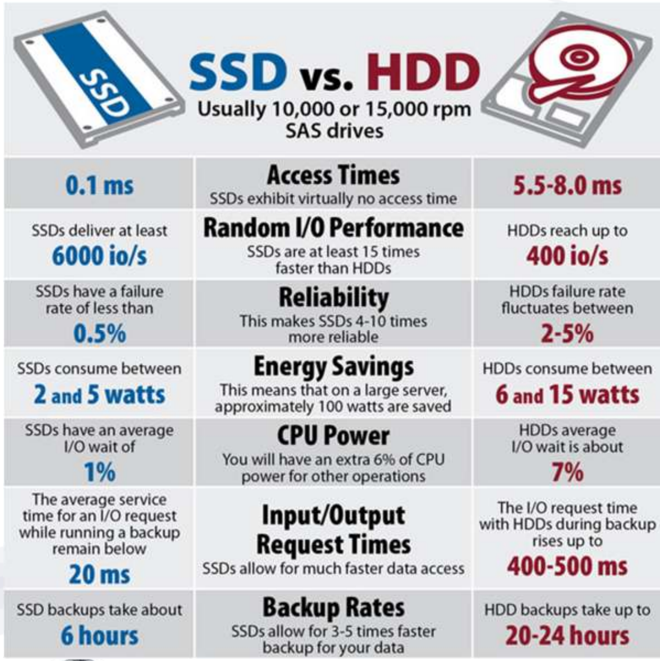
Image: XZise, 2008



Solid State Drives

- Mainly based on Flash ROM technology
  - Works in similar fashion to RAM
  - But transfer blocks of data rather than single bytes
- Main advantages:
  - Fast access (no seek time)
    - Typical access times: 35 – 100 µs (microseconds)
    - About 100x faster than HDD
  - Quiet - no spinning disks
  - Reliable - no moving mechanical parts
  - Fast startup - no delays waiting for disks to get up to speed
- Main disadvantage: High cost per byte
  - Though price keeps dropping

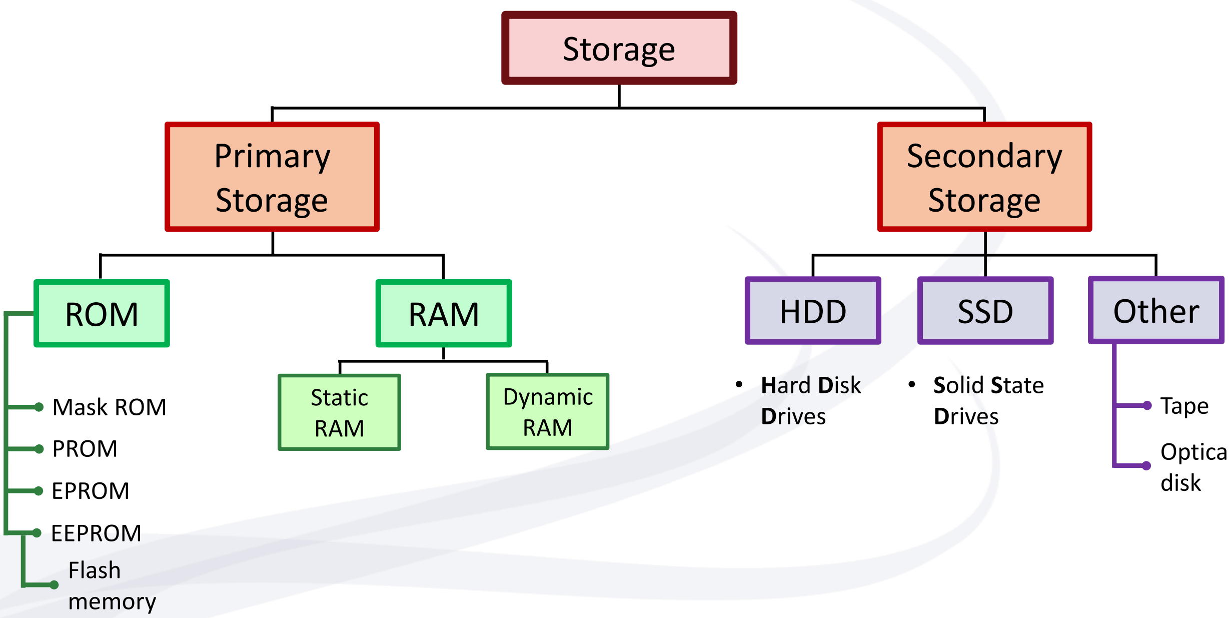
Image: Ordercrazy, 2014



SSD vs. HDD

Usually 10,000 or 15,000 rpm SAS drives

SSD		HDD
0.1 ms	**Access Times** SSDs exhibit virtually no access time	5.5-8.0 ms
SSDs deliver at least 6000 io/s	**Random I/O Performance** SSDs are at least 15 times faster than HDDs	HDDs reach up to 400 io/s
SSDs have a failure rate of less than 0.5%	**Reliability** This makes SSDs 4-10 times more reliable	HDDs failure rate fluctuates between 2-5%
SSDs consume between 2 and 5 watts	**Energy Savings** This means that on a large server, approximately 100 watts are saved	HDDs consume between 6 and 15 watts
SSDs have an average I/O wait of 1%	**CPU Power** You will have an extra 6% of CPU power for other operations	HDDs average I/O wait is about 7%
The average service time for an I/O request while running a backup remain below 20 ms	**Input/Output Request Times** SSDs allow for much faster data access	The I/O request time with HDDs during backup rises up to 400-500 ms
SSD backups take about 6 hours	**Backup Rates** SSDs allow for 3-5 times faster backup for your data	HDD backups take up to 20-24 hours



Tape

- Provides sequential access to data
  - Tape needs to be forwarded / rewound to find data
  - Very large access times
- Earliest computers used tape as secondary storage
- Now used mainly for archival purposes
  - Different formats: DDS-4, Data8, QIC, etc.
- Cheapest media per byte
- Tape can safely store data for decades
  - Provided temperature / humidity conditions maintained
  - *c.f. Hard drives:* mechanisms may seize up if not used for years
  - *c.f. Flash:* can lose data if not powered up for 2 – 3 years

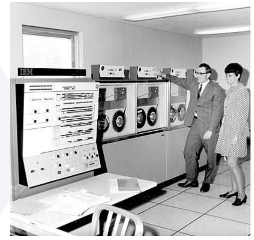


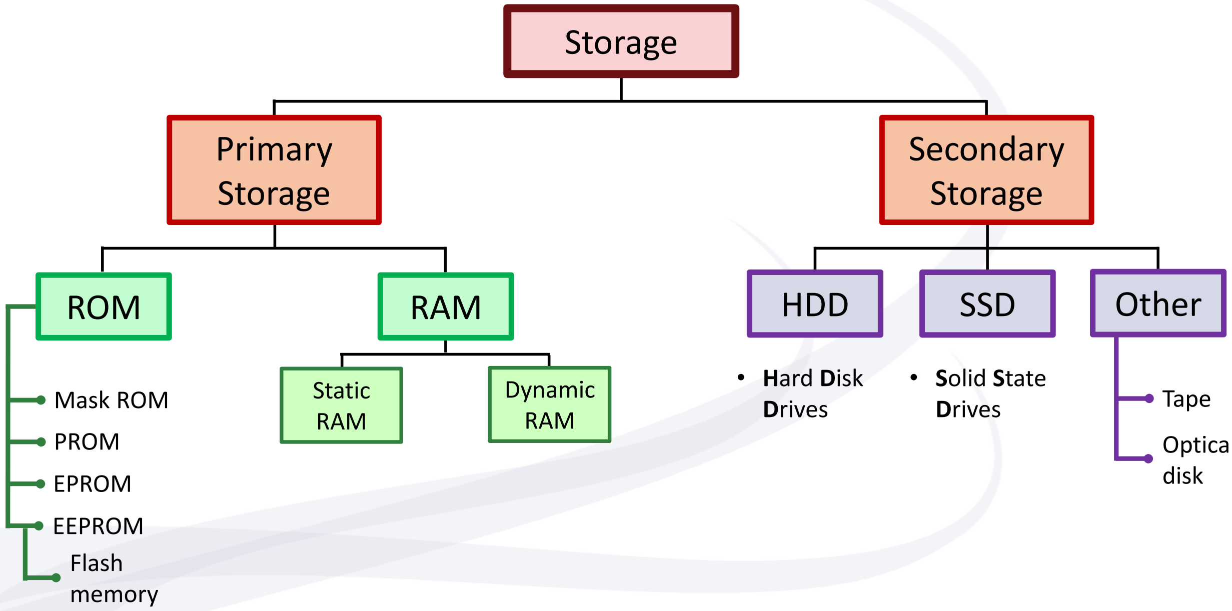
Image: Dave Winer, 2008

Image: Robert Jacek Tomczak, 2005

Optical discs

- Data read by laser heads
- 1s and 0s – 'lands' (reflective) and 'pits' (non-reflective)
- Common formats:
  - CD *(Compact Disc)* : 700 Mb
  - DVD *(Digital Versatile Disc)* : 4.7Gb / 9.4 GB
  - Blu-Ray : 25 GB / 50 GB or more
- Can be read-only, write-once or rewritable
  - Need appropriate disc and drive
- Physical damage to surface (e.g. scratches) can affect data
- Usage has declined with the increased popularity of flash drives

Image: By User:Wanted, User:Ochro



RAID

- Redundant Array of Independent Disks
  - Previously 'Redundant Array of Inexpensive Disks'
- Have different levels that achieve different objectives
  - Speed
    - To overcome latency of HDD
  - Redundancy
    - Ability to recover data if a drive fails
- Video: What is RAID 0, 1, 5, & 10?

Basic RAID Level Summary

- Raid 0 ('striping')
  - Increases speed by distributing data over different drives
  - No redundancy – worse because if 1 drive fails, all data lost
- Raid 1 ('mirroring')
  - Data is duplicated over 2 drives
  - Good redundancy – if one drive fails, data still intact
  - Cost – double number of drives
- Raid 5
  - Has data striped across a number of drives along with *parity* information
  - Data can be rebuilt from parity information if 1 drive fails
- Raid 10
  - Combines Raid 1 and 0

Module Objectives

On completion of this module, students should be able to:

- Explain the differences between primary and secondary storage.
- Describe the different types of primary and secondary storage technologies covered and their principles of operation.
- Explain the how the characteristics and specifications of different storage technologies impacts their usage in computer systems.
- Evaluate different types of storage based on their specifications and determine the most suitable for a given application.