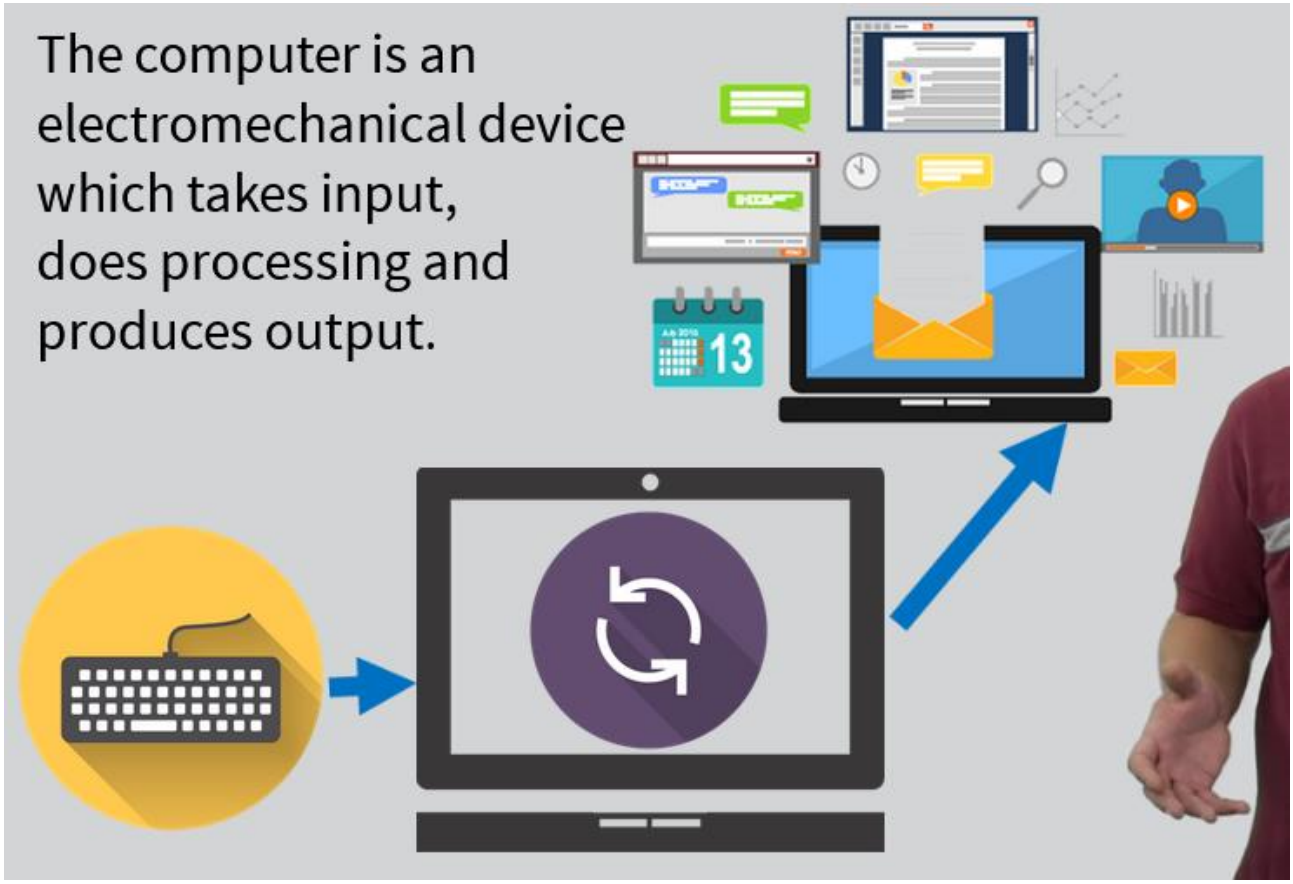
Edx - NYUx FCS.OS.1Computer Hardware and Operating Systems

Todo book: Any version of this book is a good match: https://www.amazon.com/Modern-Operating-Systems-Andrew-Tanenbaum/dp/013359162X



Type of computer

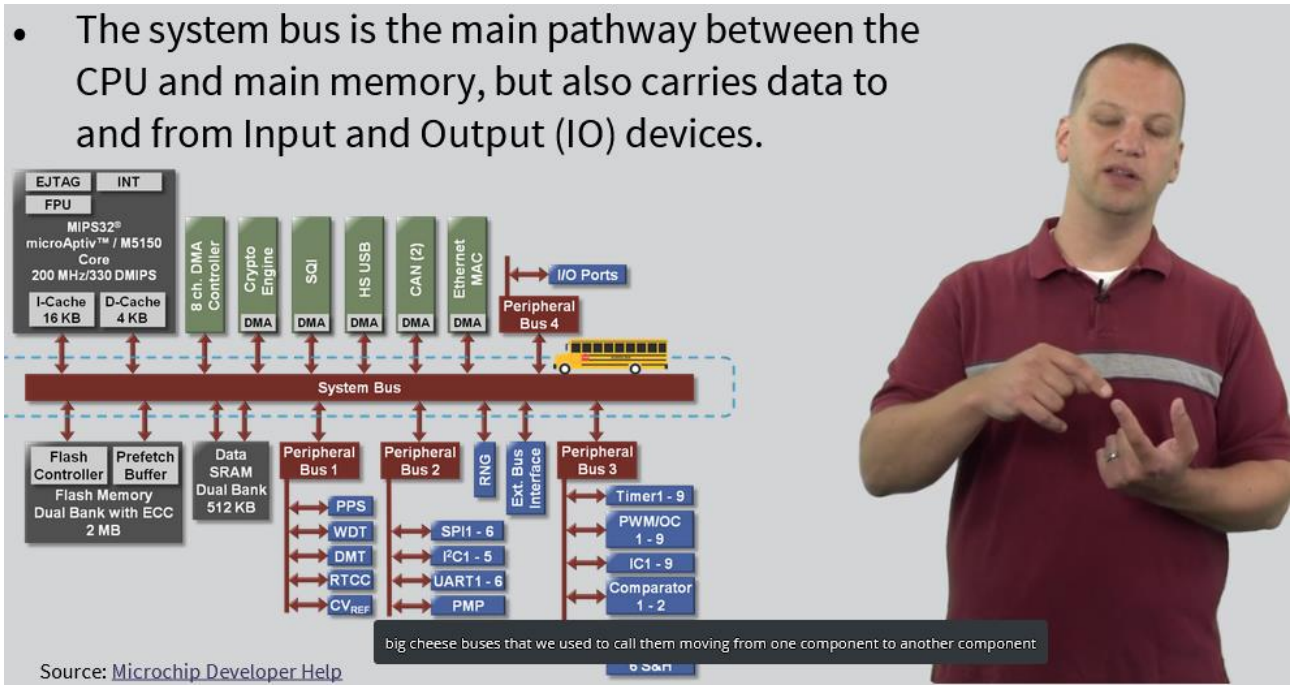
- personal (laptop tablet mobile
- server
- mainfarme
- cloude (farm of servers)

Inside a computer

- tertiary storage device (cd, dvd)
- power supply (form 220/240 to 5/12 volts)
- mainboard/ motherboard (passin infomation)
- video card (gpu graphic processing unit)
- cpu central processing unit (brain)
- RAM primary storage (when comp is down its cleard)
- secondary storage (hard drive, ssd) long tearm storage
- network interface for communications
- peripheal interface (usb, firewire, scsi)

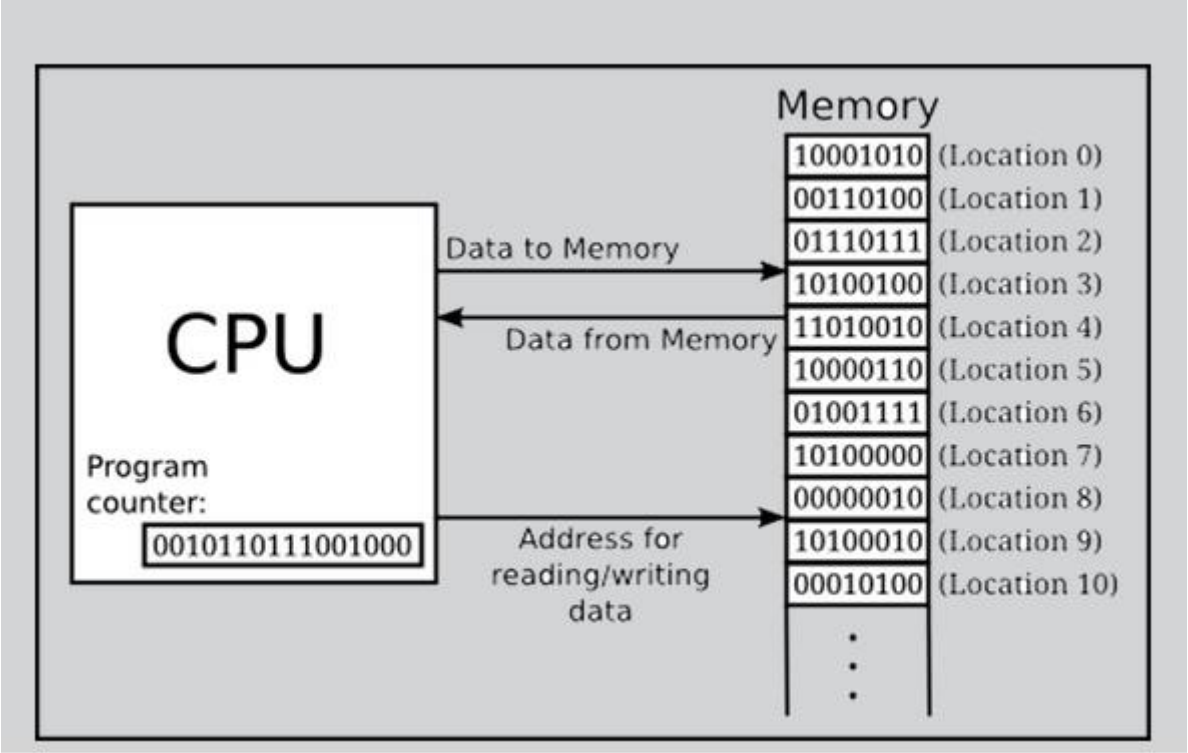
Communications between the devices

Internal communication in a machine is done via a "bus" – physical pathway for communication between two or more devices (components inside comp)

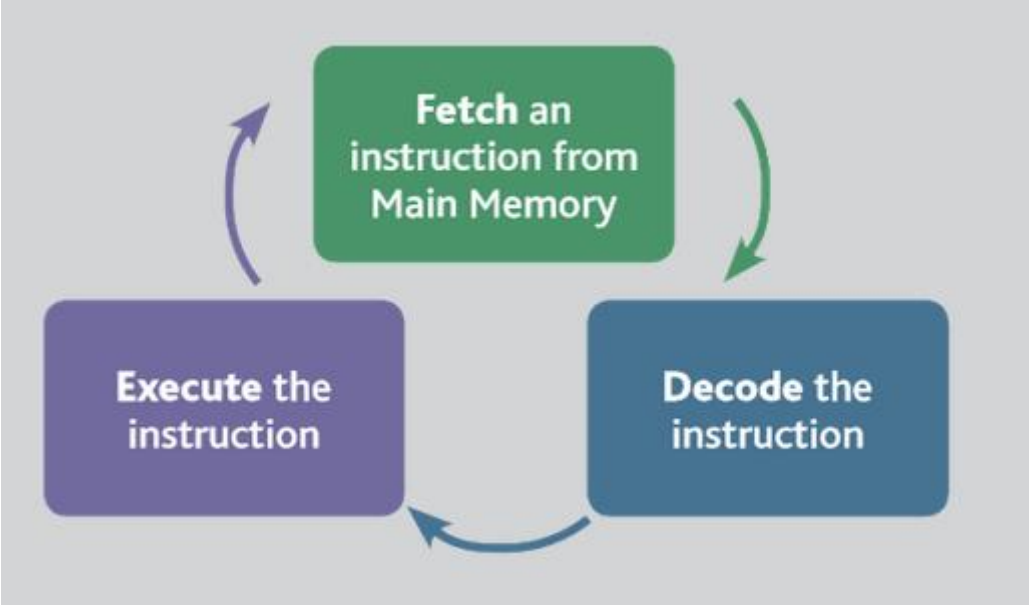


CPU

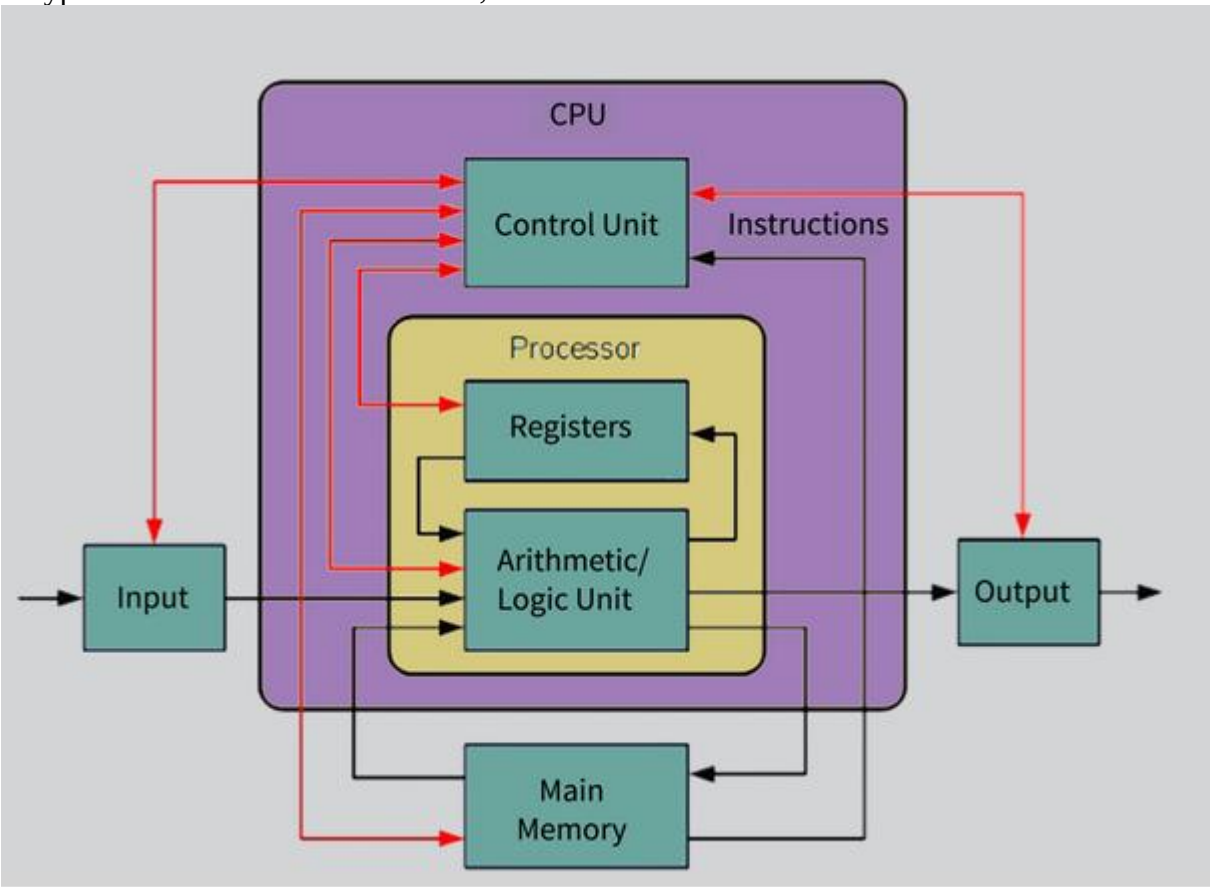
VHDL programing languae to create CPU (they get turned into a phaysical representation of CPU)
Only location where code is actually executed in the system
Its is a single pice of silicion in the form of a chip
CPU only runs machine language code



Operates on a fethc decode execute cycle



Each type of CPU has its own set of inst, which understands



Each cpu has small amunt of memory call registers which it uses to perform opertaion and store results (there is no bus), but can have a "cache" memory to perform more quickly

Machine Language code
CPU has very basic instructions (move, add, subtract, compare, jump) no if or similar operations
Instruction set is list of instrction that the CPU can perform
hight level code can be compiled or intrepeted into instruction set

Fetch execute (fetch decode execute) cycle
CPU throught bus go form proram (one instruction and then put it in his instruction register) then decodes the instruction (params and other things then moves addidtiona dana that might be necessary with that instruction)
last step execites code (instrction) 1 on second 100 bilio instructions can be executed

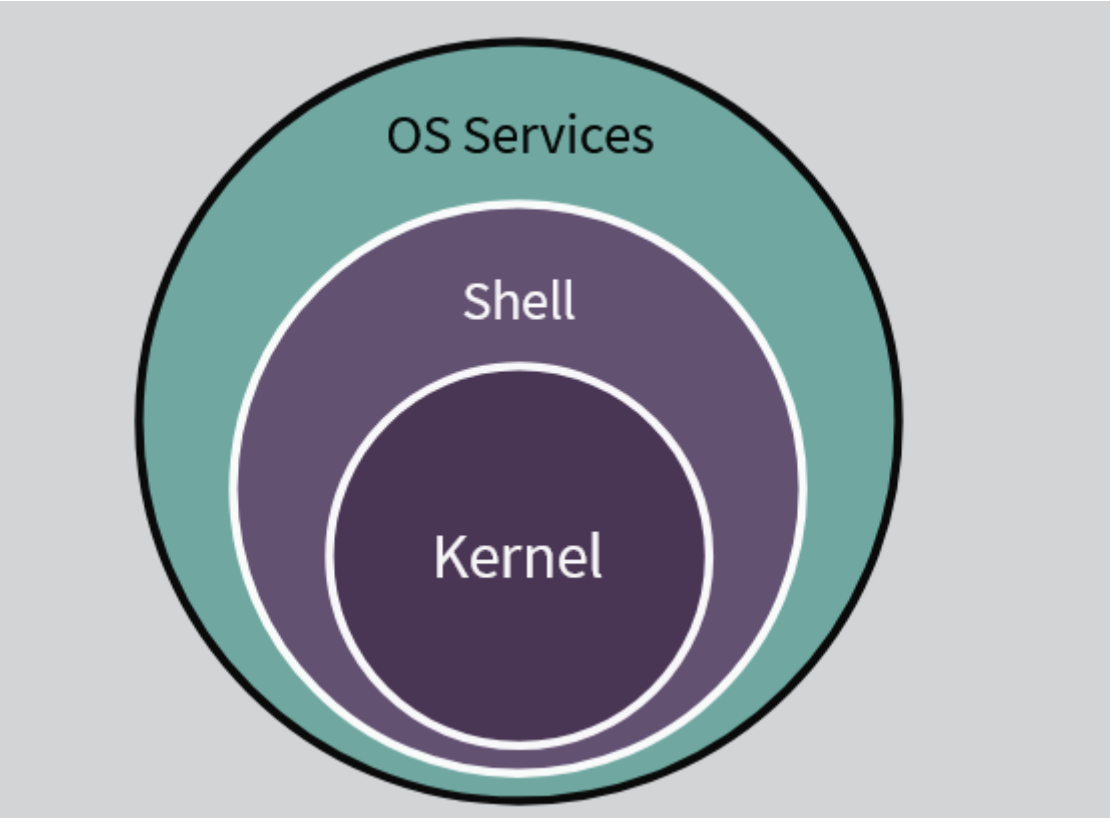
Main memory
The instructiona and all the dana where come from. Memory is not in CPU because it is expencive. CPU register are fast but not many, Main memory is slow to get but we can have a lot.
As we grow in industry hardware grows then software grows so then hardver must grow.
Register (576 B) -> Cache L1 (64 KB) -> Cache L2 (20 MB) -> DDR/ Main Memory (384 GB)
Faster and smaller as we gow down but Biger and smoller as we gown down.
RAM (random access memory because any place In it can be accessed in the same ampunt of time)
Each box (bit) have address (64 and 32 bit number for store adress of bit). If you turn down computer ram memory is cleared
Program (instructron are loaded into RAM)
Secondary Storage two classes HDD and SSD
HDD (hard disk drive) magenetic disk wich rotate togeather, constan read write head wich move to different possiton on disk size in terabaytes (very big, but slover) -> for server is ok
SSD (number of chips (like usb) dana is stored in chips, smaller than HDD butn faster, access wveryting in same amount in time, more storage more cost) -> for portalbe devices (laptop...)

OS

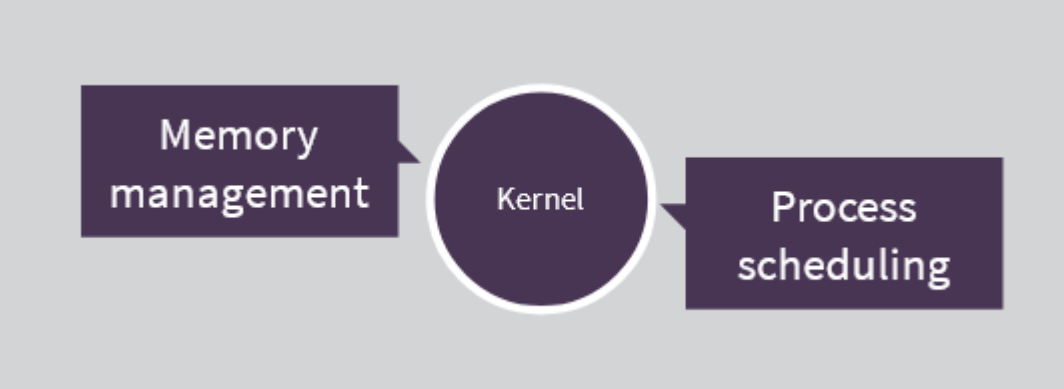
Program that controls execution of app and acts as an interface between app and computer hardware
Baiscly it is pice of software that manages the system. Manage app access to hardver. In past computers had one CPU (that runs on the same processor as the users program code)
OS doest not inclued applications

Layers of interactions
User (use keyboras and applications) not with OS
Application interac whtih OS that communicate with haedware. OS need to know about hardware. (EX: Intel architecture is recognized by windovs OS). OS need to tailoer to harfware wirh means that app does not need to be tailored to hardware

Kernal is the core component of the OS.



Responsible for managing sys resource and assist application wit performing work. Kernel is the operting system in the basic



OS as Resource Manager
Manage memory (space) and cpu (time)
Os is program also so he need to mange resurces for sebe
we can have multiple insstace of program -> process

A process is:
- A program in a running state
- Loaded into main memory
- Scheduled

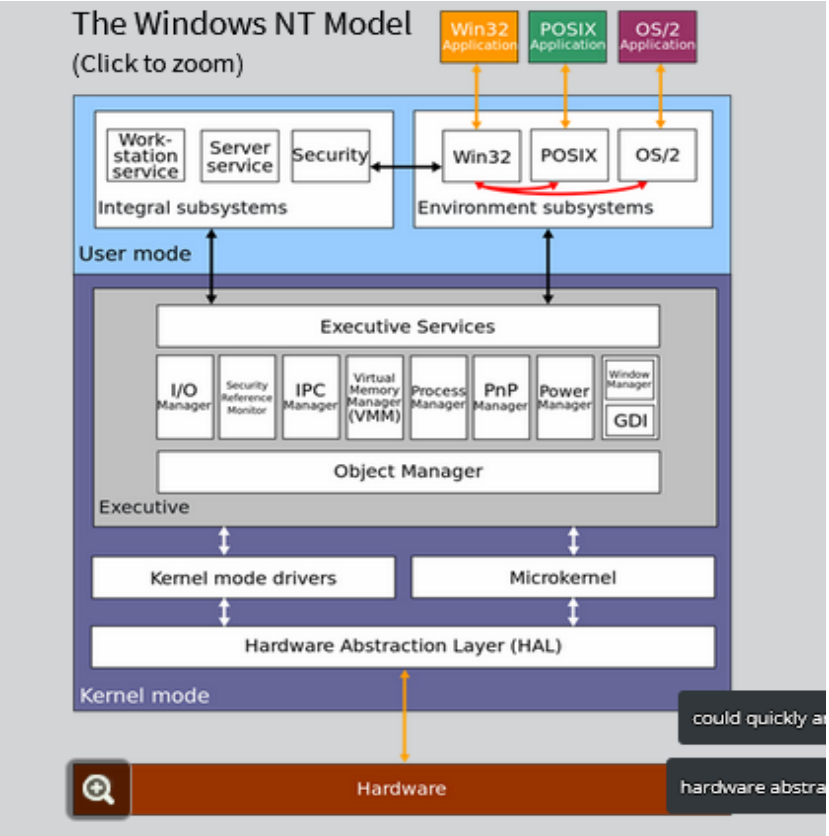
A process has:
- Access to files
- Access to networking connections
- Code
  - The code is what's run

procedures – call by os
interrupt – hardware notification

5, 6, 7 -> kernel of OS
6 -> SSD, hardvare

8.13 OS but not kernel

OS Levels	
Level	Name
13	Shell
12	User Processes
11	Directories
10	Devices
9	File Systems
8	Communications
7	Virtual Memory
6	Secondary Storage
5	Primitive Processes
4	Interrupts
3	Procedures
2	Processor Instruction Set
1	Electronic Circuits

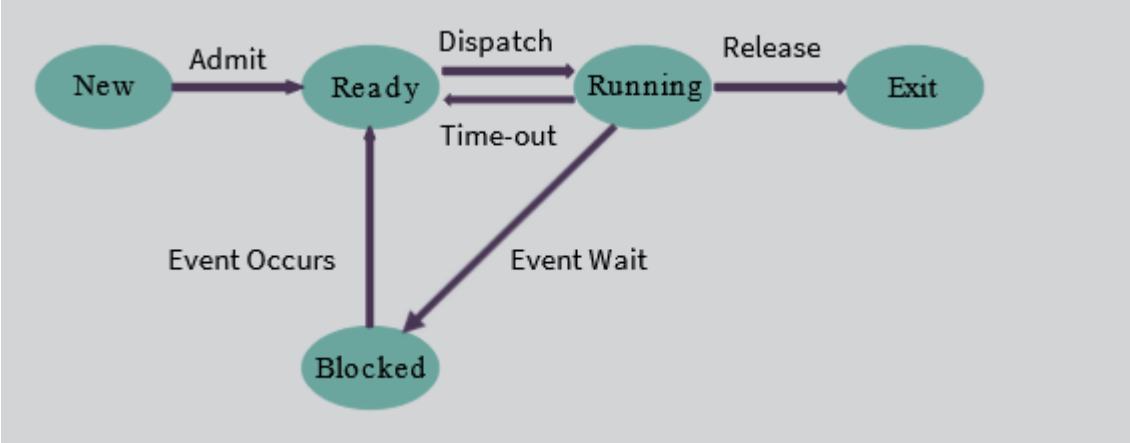


HAL Hardware abstraction Layer
For replacing architecture (intel, amd ... (HAL can be revriten for architecture -> it is for tables))
Layer which can provied the kernel with a set of functions to call whicj program the hardware properly

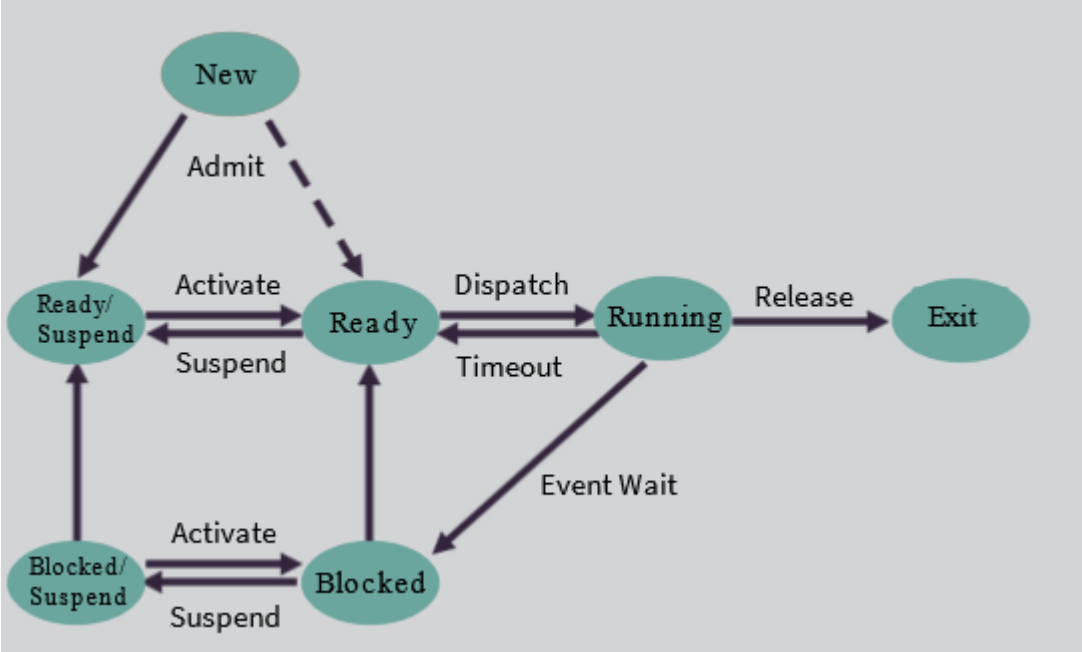
Windows Device Drives
Cams, Keyboard, USB Controllers
Device Drivers are kernel layer software written by companies that design hardware. They provide functions for the kernel to call in orderr to access the hardware
Run inside kernel

Processes

Process is running program in a system state (code, data, context)
Is stored in memory space is created by OD to keep track of state of program and resources assigned to the running program
State of process is a condition that the process will spend a amount of time in.
New - read code of program (IO operation)
Ready – ready to GO, no CPU assgined
Running – CPU assigned can go time out (no more time for you) or blocked (waith event, relase CPU)
Blocked – waith for something to happen so that can g0o to read and be executed again
Exit – program is done, procees still exist, must be done, send info to parent process or os

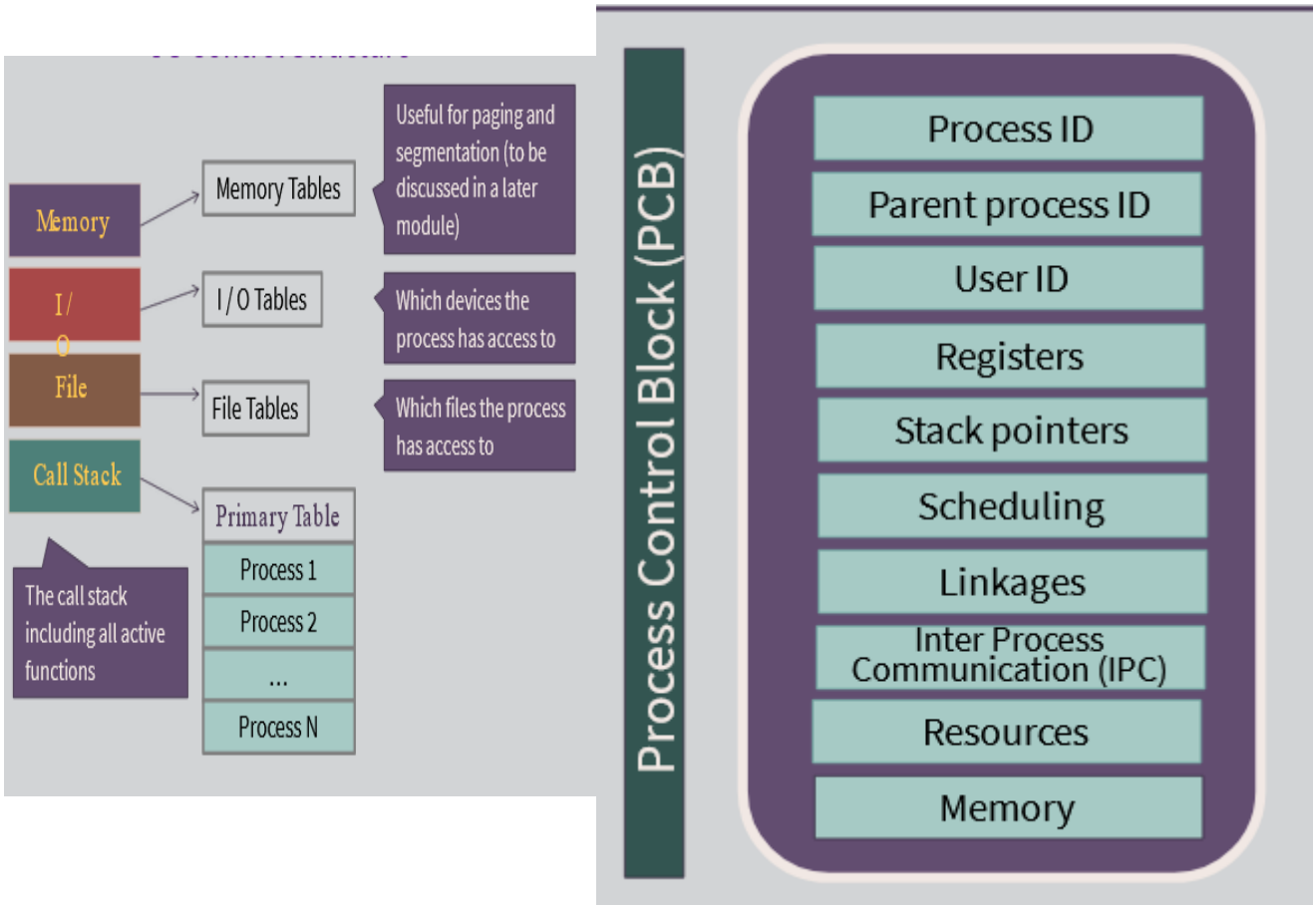


Suspension Processes that need to be run at current time so we want remove it from main memory, but wen we start it again we want to start where we where last time
-> save state on secondary memory so that main memory can be relased.
-> Controlled by medium- term scheduling algorithm
The process will not be aware of the suspension
Used for debuggin, long term delay, freeing main memory



From Ready/Suspend -> to Read need log time (load from secondary memory to main memory)

Process Image – The PCB (Process Control Block)
All that OS needs to run and control the process



User Mode vs Kernal Mode
stao : https://nyuepoly.articulate-online.com/p/7714612652/story_html5.html?Cust=77146&DocumentID=19e6718b-3985-45b4-a494-8c8743cc5fb3&Popped=True&v=1&InitialPage=story.html