HFN:

CSE-321

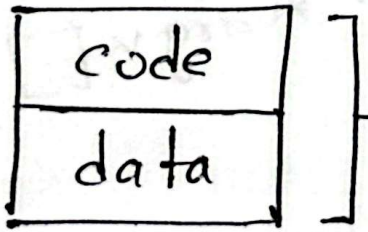
Process:

- program in execution
- CPU
- batch system → job
- unit of work of the OS.
- certain space occupied
  - → certain address at the memory

Program
- → Passive

Process
- → Active

1 ↳ Program → Multiple process
- → 1 code



→ executable files

Lecture-2

CSE-321:

ML (1/0) → binary

Program —sun interpreter→ executable file → .exe

executables

Process:

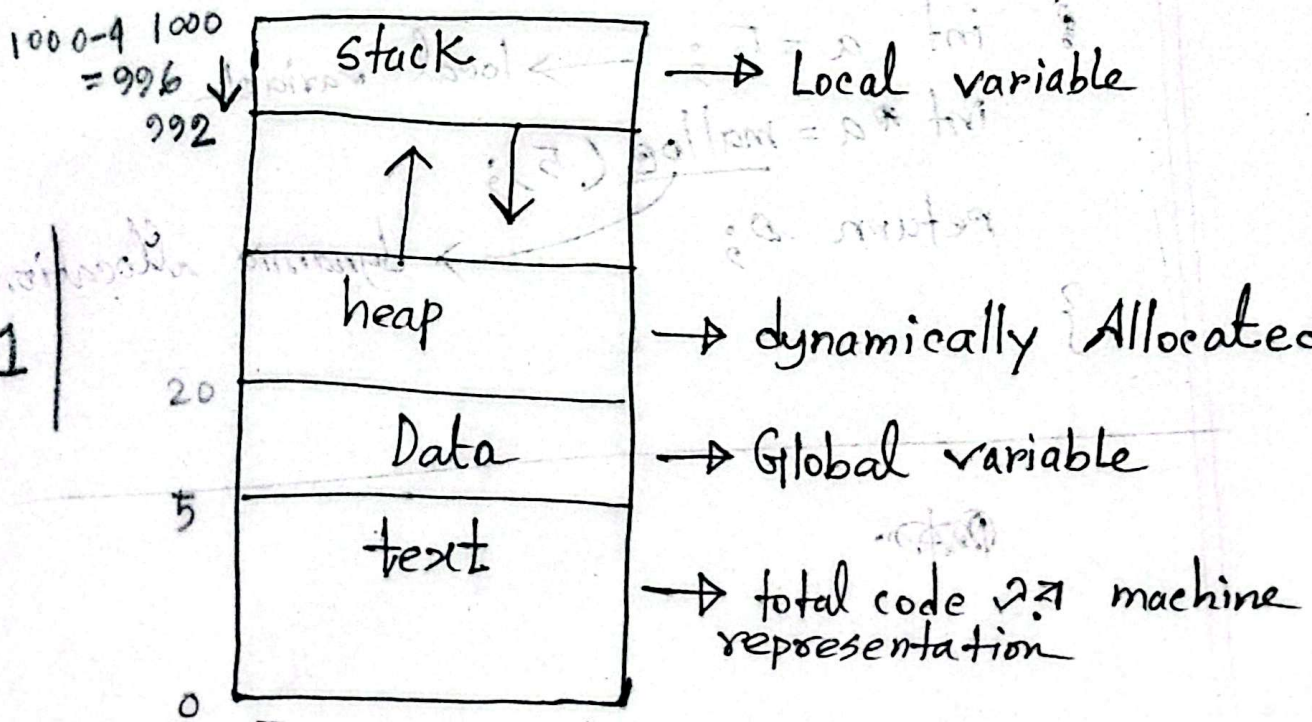
code – 1 টি

process – 2 টি/multiple

A process is a program that is in execution.

প্রতিটি process memory'র certain space occupied.

memory'র certain address থাকবে।



Memory representation of a process

Process:

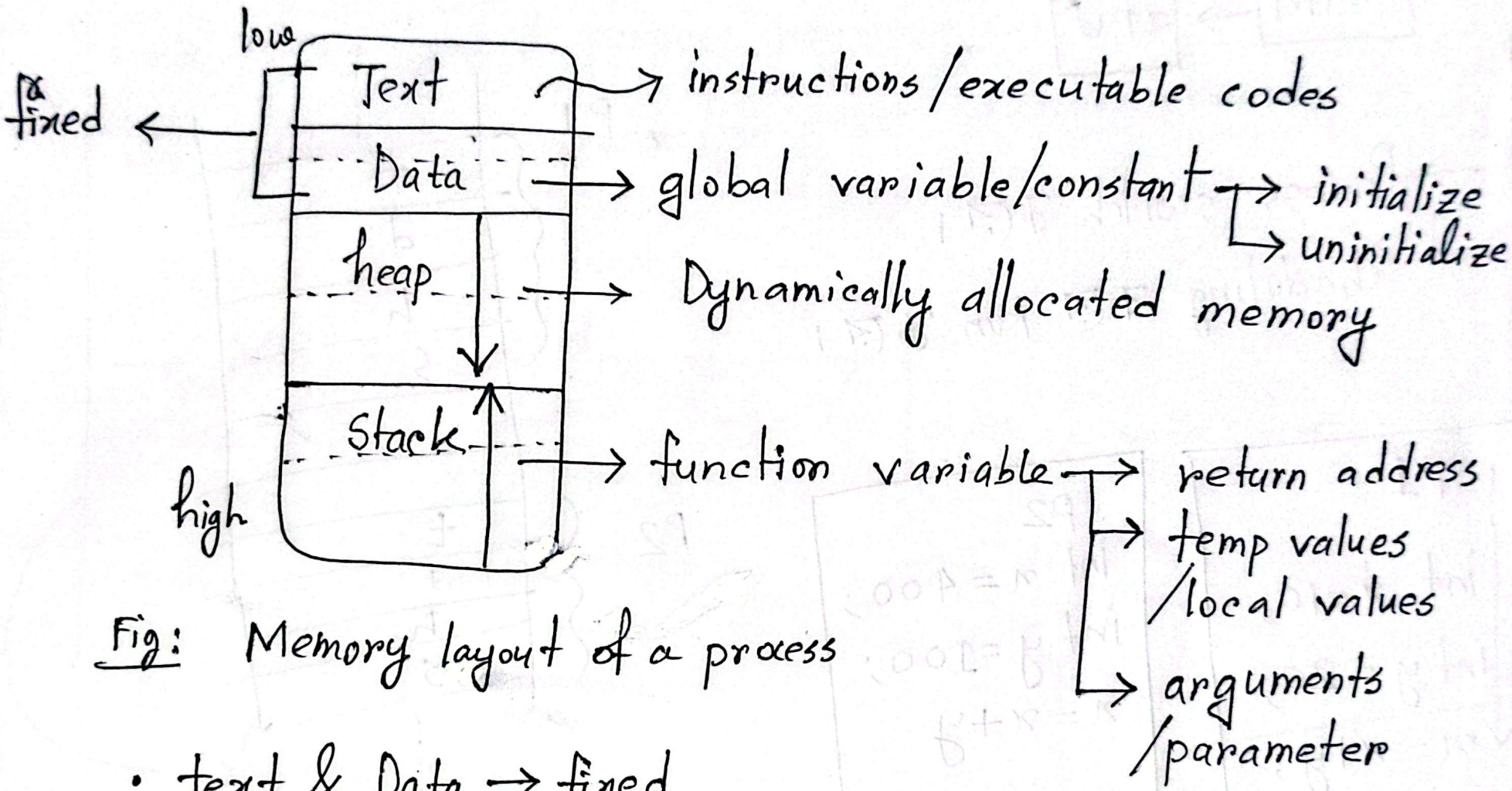
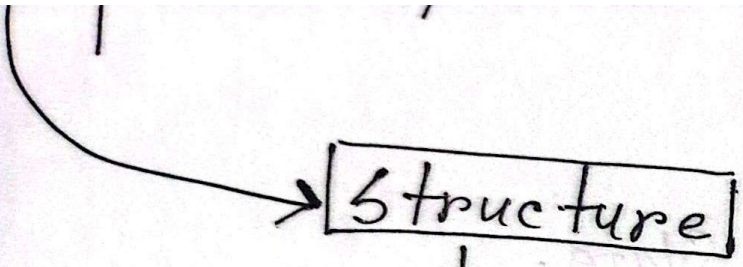


Fig: Memory layout of a process

- text & Data → fixed
- heap & Stack → Shrink & Expand
- heap & Stack → can't overlap



↳ PCB (Process control Block)

- OS এর process কে store করতে যে Data structure use করে সেটা PCB

Process is represented in OS by Process Control Block (PCB) or Task Control Block.

1) Process state: running, waiting etc
2) Prog counter - location of instruction to execute next?
3) CPU register - contents of all process-centric registers.
4) CPU scheduling information - priorities, scheduling queue.
5) Memory-management info - memory allocation.
6) Accounting info - CPU used, process কতক্ষণ চলছে (age)
7) I/O status info - Input/output কিছু? কতটা?

States of Process

1) New
2) Running
3) Waiting
4) Ready
5) Terminated

Delay : Ready

Process State :

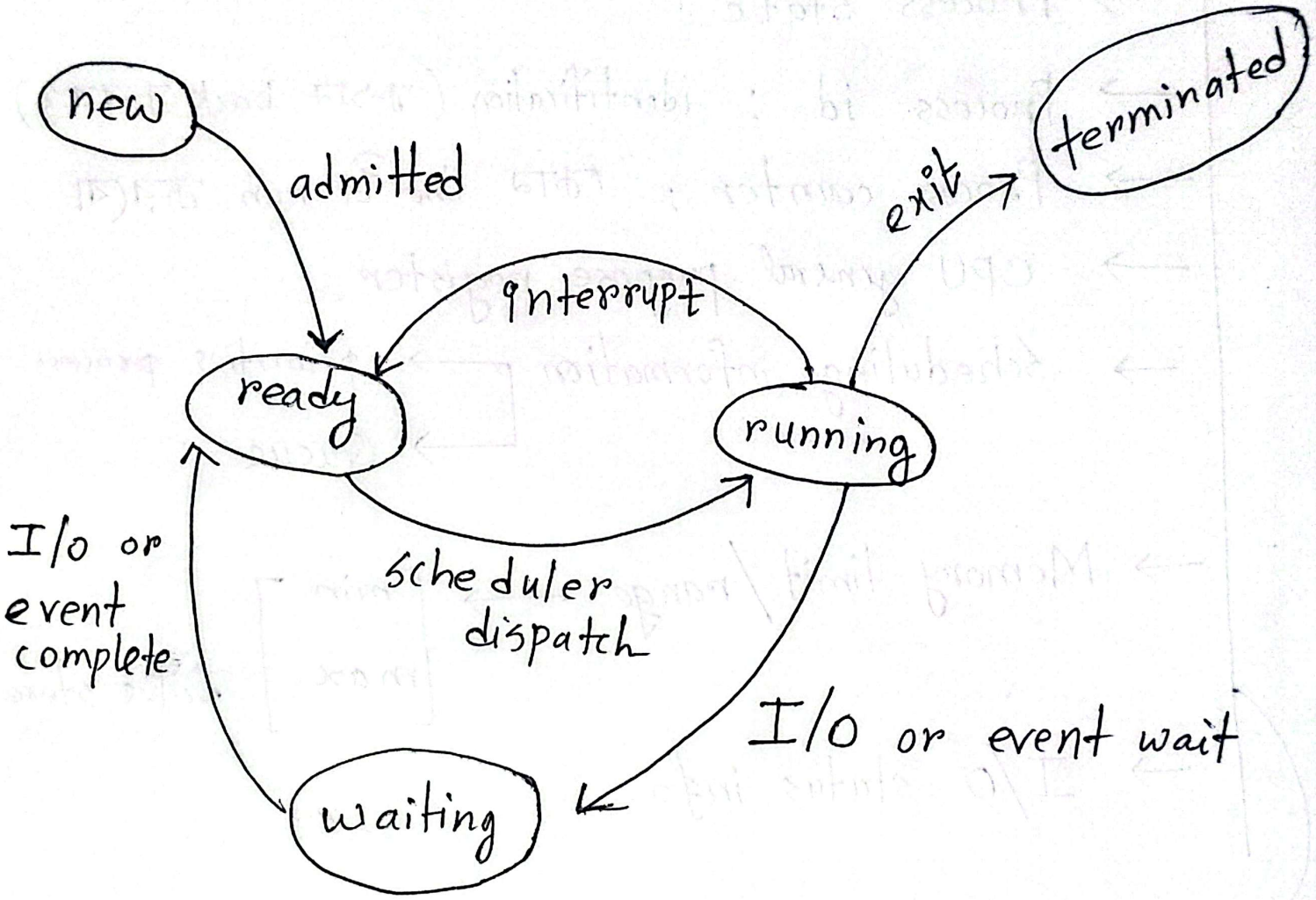
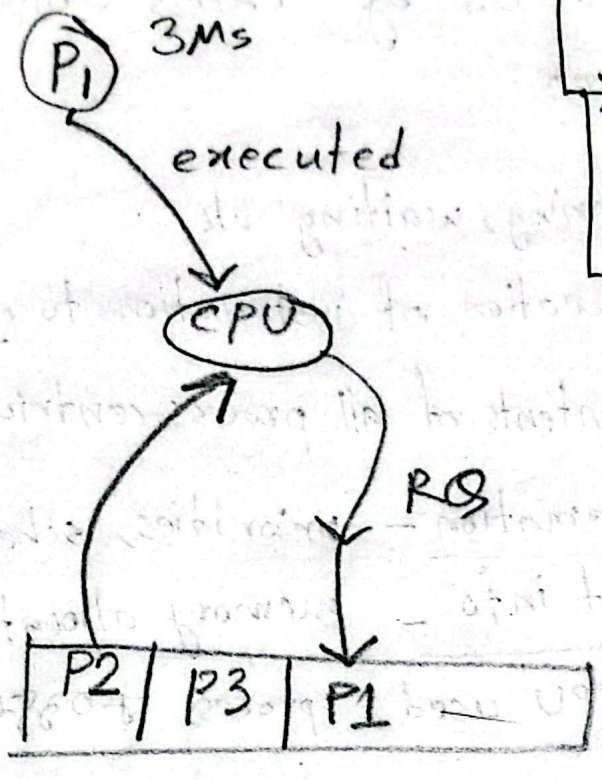


Fig: Diagram of Process State.

→ occurs when CPU switches from one process to another

Context Switch:-



→ (very costly) operation

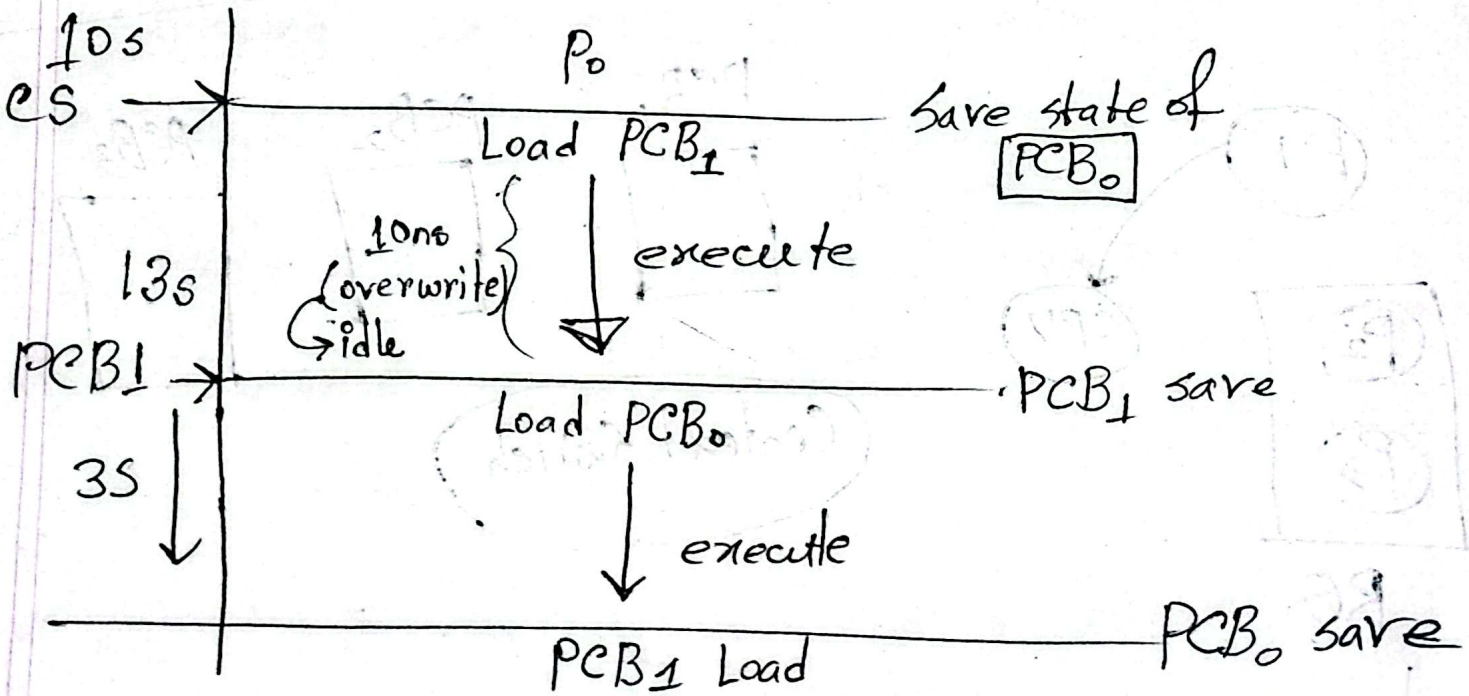
→ Memory access 2 बार ! costly

→ memory redrive 2 बार !

→ overwrite 2 बार !

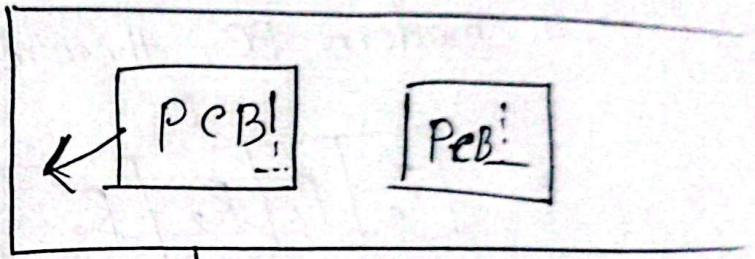
→ लेना काम 2 बार न (10 nano sec)

Context switch

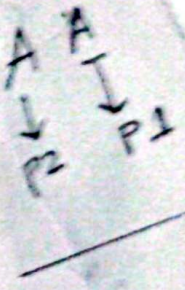


Process Switching :

- Context Switching
  - Pure Overhead
    - Pure overhead ↑ CPU idly বসে থাকে। কাজ কমবে।
    - Pure overhead যত কমবে, CPU তত ভালো কাজ করবে।



→ কিছু Process কাজ করে সরায় দিয়ে আবার New Process কাজ করবে।

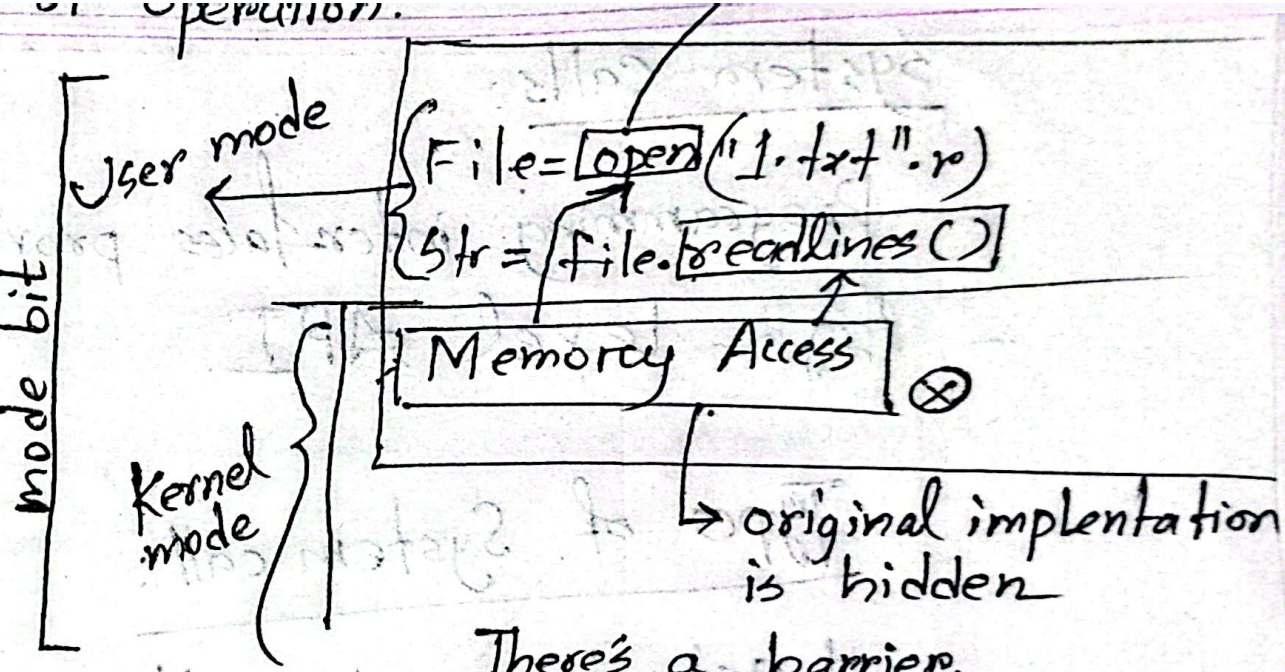


Two modes:

1) User mode
2) Kernel mode

User: bit (1)

Kernel mode: bit (0)



mode bit define is it in user or kernel?

→ System call করলেই kernel এ যাবে। (mode bit 1) এরপর আবার Back করবে (mode change) ↓ change হবে চলে যাবে user mode এ.

directly can't hardware

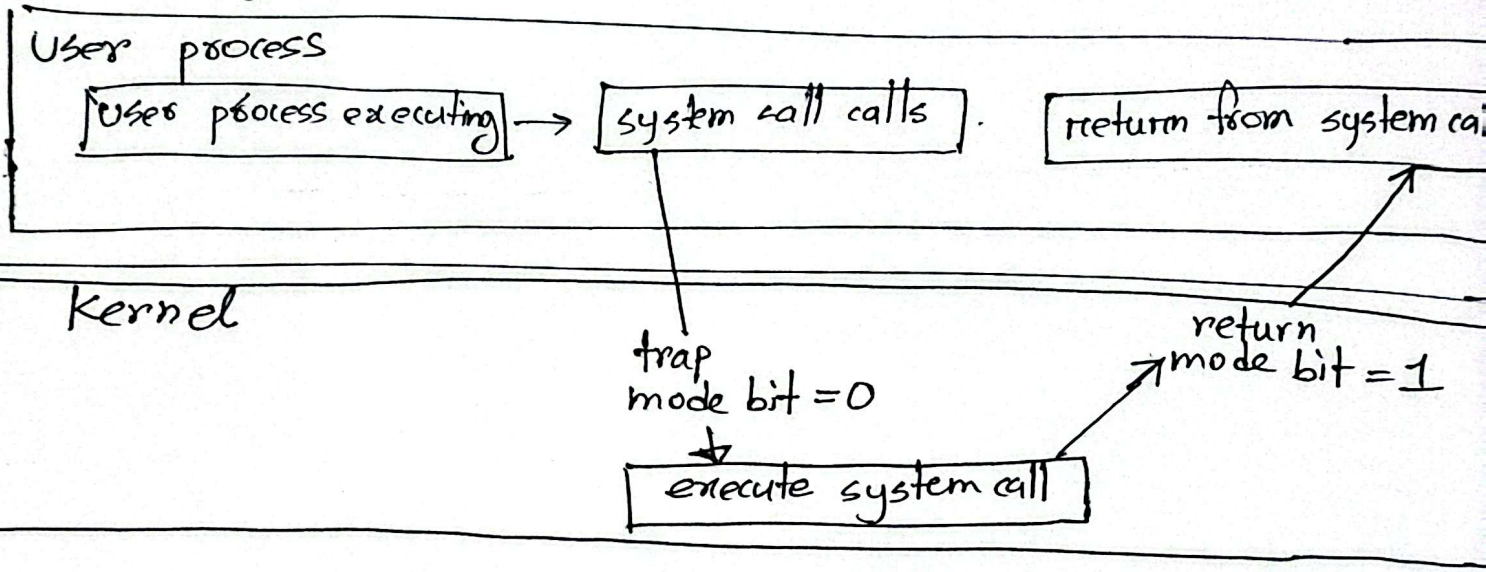
so that user directly can't
access memory / hardware

↳ so that one program can't effect another program.

Dual Mode operation :

- Distinguish between [OS code (system) and user defined code (application / web-browser / docs / spreadsheet)]
- Mode bit → Kernel mode (0) :- bit = 0
  → User mode (1) :- bit = 1
- Provides protection of the OS from errant users
- Program 2 types of lines of codes (sensitive code) that can (change behavior of system). That lines of codes are (privileged instructions) → executed only in Kernel mode

figure:



System calls:

- Programming interfaces provided by OS
- High level API

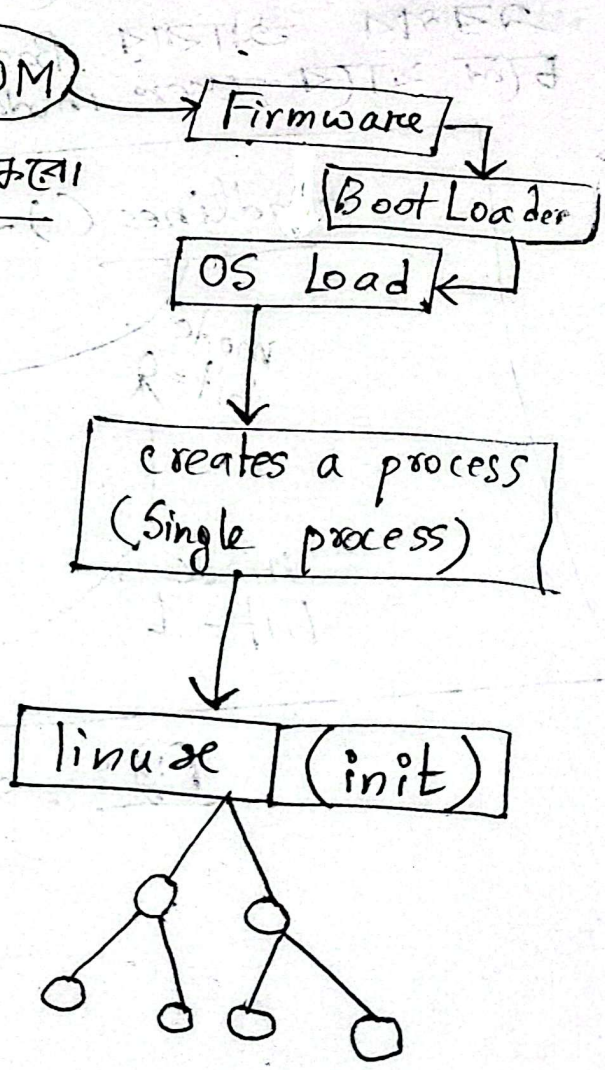
Types of System call:

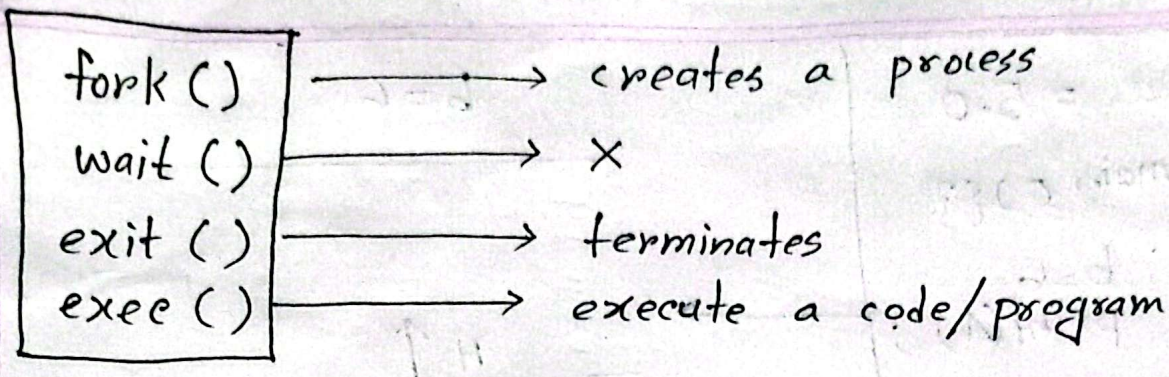
getpid → process id

⊞ Operations on Process:

- pc start করলে → ROM → Firmware → Boot Loader → OS Load
- init → Multiple process create করে।
  - creates more process

- Firmware: check everything. if my mouse, keyboard all are wooking or not.





process creation:

- parent process
- child process

} (tree) like structure

process identifier (pid)

Sharing options:

i) parent & child share all resources

ii) child share subset of parent's resources

iii) Parent & child share no resources

execution option:

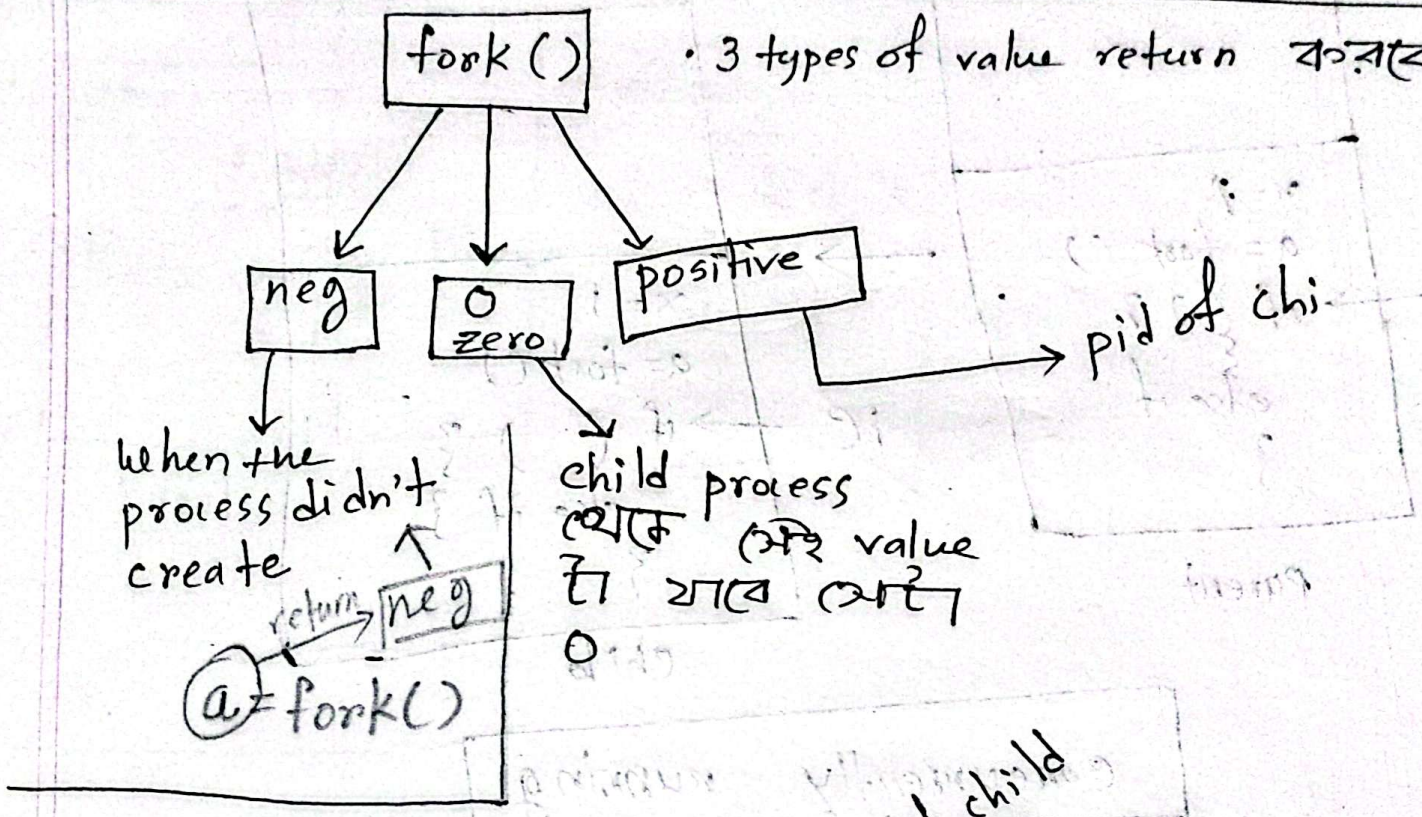
i) Parent & children execute concurrently

ii) Parents wait until children terminates

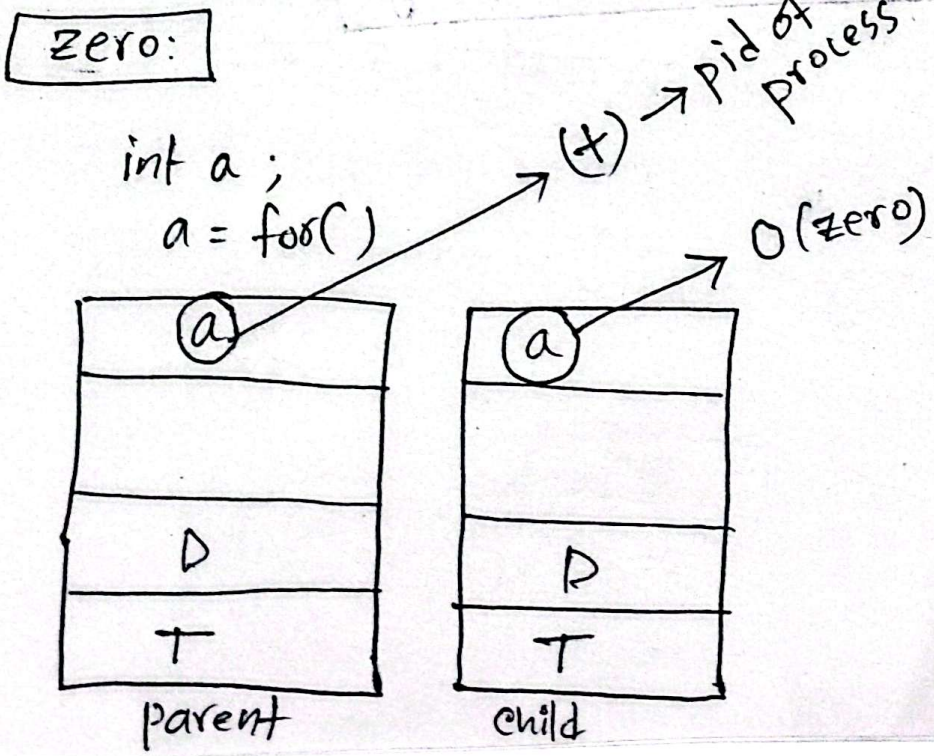
* ১ টা process থেকে আরেকটা process create ক
execute হয়, scheduler দিয়ে

(PC): program counter: points to the next line to execute

fork() → এর জন্য আলাদা child create হবে।
Parent process এর copy হচ্ছে।



zero:



concurrently running

---

wait() → এটি Parent এর under এ যতগুলো child থাকে
সবগুলো যতক্ষণ execute করবে না ততক্ষণ
wait করতে হবে। (Forcefully stop করে রাখবে)

wait() recap:

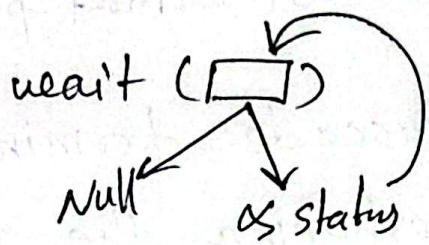
Parent waits until all of its children complete execution.

↳ Parameter:

① wait() → Null

② wait(&a) → address (status)

* Cache : store the temporary data.



Process Termination :

child

① exit() → when process executes its last statement it asks the operating system to delete it with exit()

→ i) Returns [status] → wait()

↳ status data from child to paren

→ ii) Resources → release.

↳ deallocated by OS.

② Abort () : (→ Parent)

↳ Parent might terminate the execulation of child process.

why abort is used ?

→ Child has exceeded resources allocation.
→ task assigned to child no longer needed.
→ Parent is exiting. (os will not allow to continue child without parent)

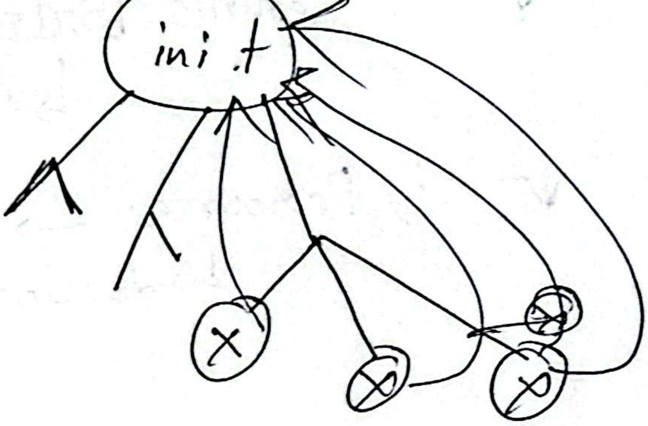
③ Abort () :
↳ i) Cascade termination
↳ ii) Reshaping the head
→ if a process terminates children must be terminated

i) Cascading : All process & subsequent process (child/grandchild) etc. gets terminated

ii) Reshaping the head :

এত গুলো child হয়। Parent মারলে, তারা child সরাসরি intit (root) এ reach করে।

i.e. it gets reshaped.



④ → Parent

wait for child to terminate then, the call returns status info and id of the terminated process.

pid = wait(&status)

⑤ if no parent waiting (no wait())
↳ process is a **zombie**

⑥ if parent is terminated without wait(),
↳ process is an **orphan**
↳ Parent done but didn't wait for child.

Zombie situation remedy: wait() should be in parent.

IPC: (Interprocess Communication)

How two process communicate with each other?
↳ a shared memory or buffer.

1) Independent: [(cannot affected/affect other process)
doesn't share data

2) cooperating: [(can affect/affected by other process)
Share data
↓
2) Process cooperation:

i) Information sharing: share data
ii) Computation speedup: for faster comm, breaks into subtask
iii) Modularity: seperate process or threads
iv) Convenience.

→ IPC is a mechanism to exchange data & info among other processes.

Two fundamentals model of IPC

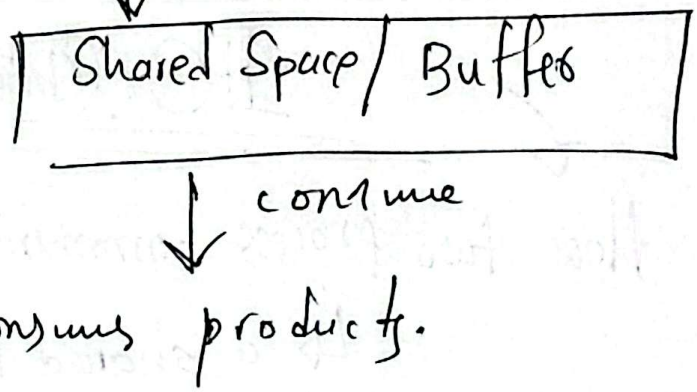
1) Shared Memory

2) Message passing

1) Shared memory:

(Producer - Consumer problem)

Producer: produces products for consumer



Consumer: consumes products.

Message passing system:

→ If P & Q wants to communicate they must send messages to & recieve messages from each other.

→ A communication link must exist betw P & Q.