What is a Signal ?

What is a Signal handler ?

Difference between Signals and interrupts

Examples of Signals in linux

- Signals are sent to a process to notify it of an event that has occured.

- Signals can either be sent by one user process to another or by kernel to the user process. ctrl + c → shell

- A signal can be received either synchronously or asynchronously.

- Synchronous : signals are delivered to the same process that performed the instruction that caused the signal. Eg. divide by zero, illegal memory access. Divide by zero, →

- Asynchronous : signals are generated by an external event to the running process . Eg. ctrl + c from keyboard to terminate a process - shell →

Signal handlers

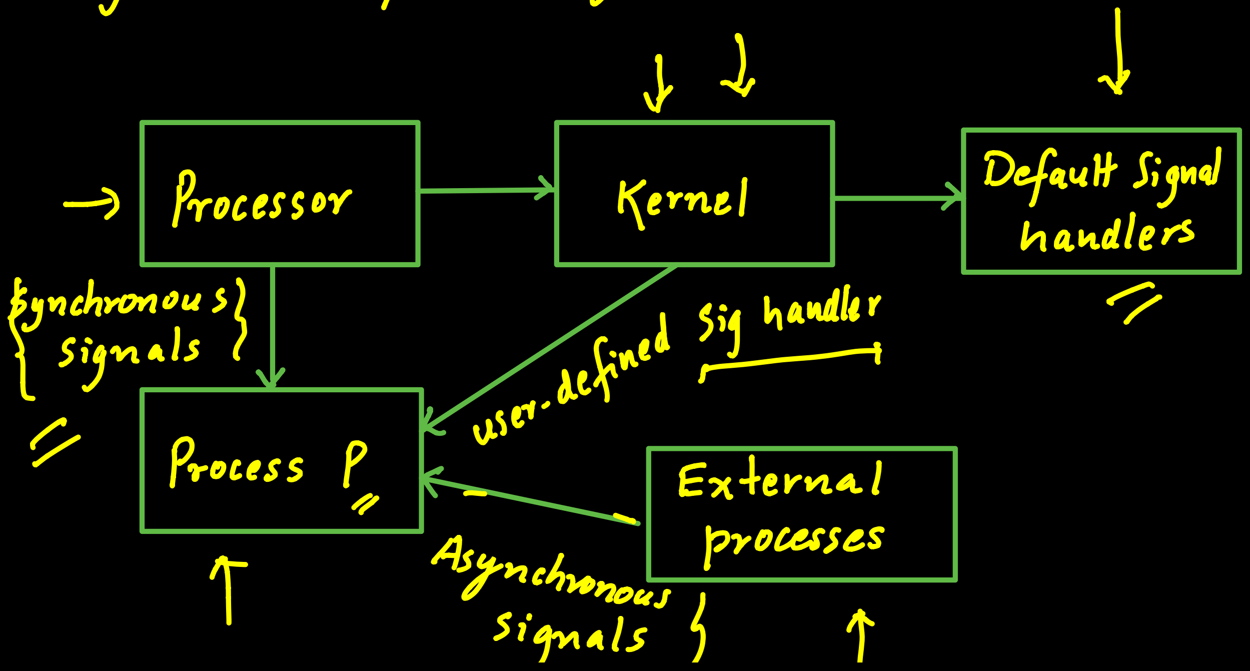
- Signal handlers are the functions that handles the signal received by a process

- A signal may be handled by one of the two possible handlers :—

1. Defaut Signal handler ✓ → SIGINT( ){
   {— — .
   }
2. User-defined Signal handler ✓

- Each signal has a default signal handler that kernel runs when handling the signal.

- This default action can be overridden by user-defined signal handlers.



Interrupts vs Signals

- Interrupts can be viewed as a mean of communication between the CPU and OS Kernel,
- Signals : Kernel → process
- Interrupts may be initiated by :

  CPU – divide by Zero, page fault etc.

  Devices – keyboard, disks

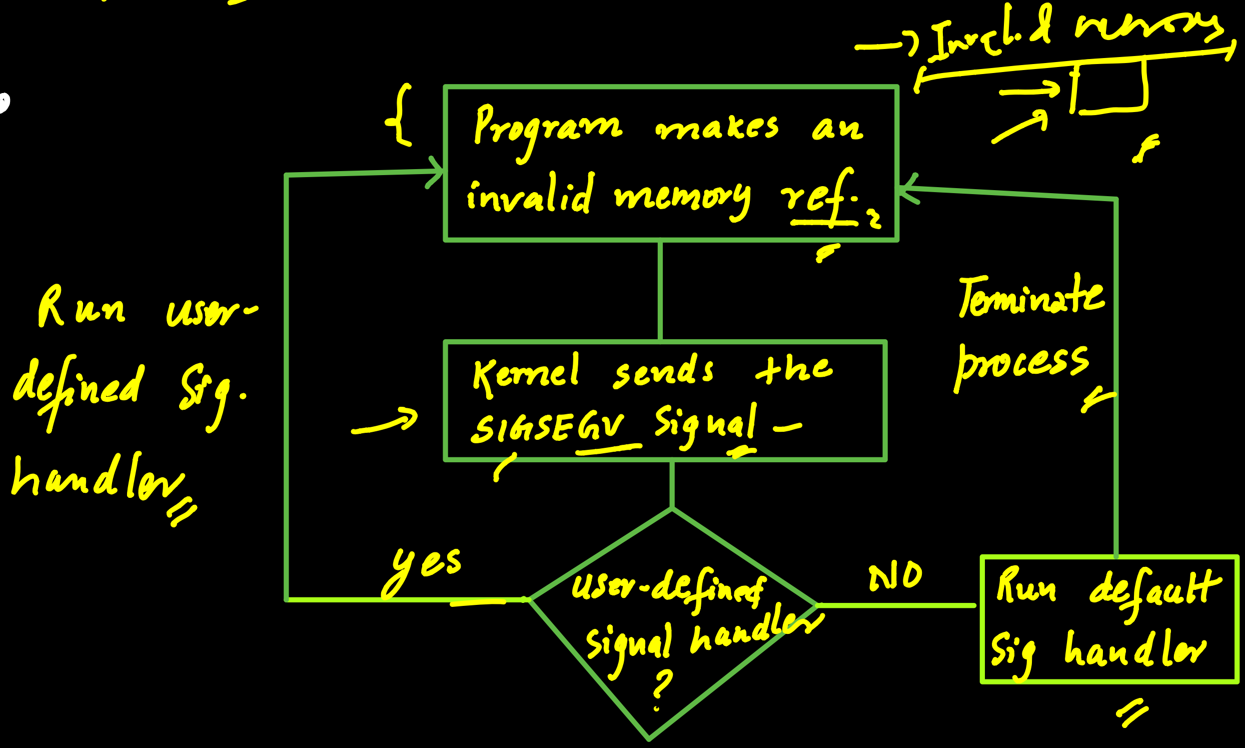
  CPU instructions – traps (INT),
- Signals are initiated by the OS kernel (SIGFPE, SIGSEGV etc.) or by a process.

- Interrupts → CPU → Interrupt handler
- Signals → Kernel → Signal handler

Examples

SIGSEGV → Segmentation fault

- This signal is sent by the kernel to a process when it makes invalid memory reference.

- 
→ Invalid memory
Program makes an invalid memory ref.
Kernel sends the SIGSEGV signal
User-defined signal handler ?
yes
NO
Run default Sig handler
Terminate process
Run user-defined Sig. handler

SIGCHLD

→ CPU → Kernel fork() wait() ← P

- This signal is sent to the parent process when its child terminates.

→ CPU ←

- If the parent is waiting on child using wait() system call, the exit status of Child is caught by the parent.
- Zombie vs orphan process video

SIGFPE → Floating point exception

Divide by Zero

Kerenel

SIGINT → Terminate the process

Ctrl + C on Keyboard

SIGSTP → Suspend the execution

Ctrl + Z on Keyboard

SIGKILL → Terminate immediately

cannot be Caught or ignored