19UCS129
Assignment No:- 6

PAGE NO.:
DATE: / /

Q.1 Write short note on:

a) source of interrupts

→ The processor receives interrupts from two sources:

- External (hardware generated) interrupts:- External interrupts are received through pins on the processor or through the local APIC.

Any external interrupt that is delivered to the processor by means of the INTR pin or through the local APIC is called mask able hardware interrupt.

- Software-generated interrupts:- The INT n instruction permits interrupts to be generated from within software by supplying an interrupt vector number as an operand.

Interrupts generated in software with the INT n instr^n cannot be masked by the IF flag in the EFLAGS register.

b) Interrupt & exception

→ - Interrupts occur at random times during the execution of a program, in response to signals from hardware.
- Software can also generate interrupts by executing the INT n instr^n.
- Exceptions occur when the processor detects an error condition while executing an instr^n, such as division

PAGE NO.:
DATE: / /

by zero.

- When an interrupt is received or an exception is detected, the currently running procedure or task is suspended while the processor executes an interrupt or exception handler.
- When execution of the handler is complete, the processor resumes execution of the interrupted procedure or task.

Q.2. Write 80386 assembly language program to display return values of system calls related to process. 2

→
```
.section .data
.section .bss
    .lcomm pid, 4
    .lcomm uid, 4
    .lcomm gid, 4

.section .text
.globl _start
_start:
    movl $20, %eax
    int $0x80
    movl %eax, pid
    movl $24, %eax
    int $0x80
    movl %eax, uid
    movl $47, %eax
    int $0x80
    movl %eax, gid
end: movl $1, %eax
    movl $0, %ebx
```

PAGE NO.:
DATE: / /

1 9 U C S 1 2 9

Q.3 Write short note on:

a) linux system call interface

→
- Linux support multiple processes running simultaneously
- At any time, memory images of multiple processes exist simultaneously with the same physical memory.
- Multiple resources (IO devices) are used by a process in mutually exclusive manner with other processes.
- A resource is used by only one process at a time and other processes wait for this process to release the resource before they can use it.
- To protect a process by other processes linux do not permit a process to do direct IO i.e., the OS does not permit OS code to be executed by a process in user mode.
- A process makes a call to a function within an OS to perform such operations. Such functionality within OS is called system call.
- IA32 processors provide instr" for handling operating system calls.
- INTN "N is an immediate constant ranging from 0 to 255.
- In linux, instr" int $0x80 is used for making system call to the operating system.

PAGE NO.:
DATE: / /

19UCS129

b) system call identification.

→ • Linux provide a mechanism for entering into system call by using int $0x80 instruction.
• After executing int $0x80, control of program transfers to a ~~prof~~ predefined location within OS. This code is called as system call handler.
• System call handler handle various system call such as read, write, open etc as per the system call identification process.
• Each system call in ~~two~~ linux is given a unique number for identification.
• In linux register EAX is used to pass the system call number.

c) returning values from system calls.

→ • Return value of system calls is implemented similar to function calls.
• System call return their values in eax register.
• There are two kinds of values
  - A) non negative
  - B) negative
• All non negative values returned by the system call represents successful execution of system call, whereas negative values represents error conditions.
• For write system call, it returns the size of string written to the file

descriptor

b) system call value | system call | description

20	getpid	retrieve process ID
24	getuid	retrieve user ID
47	getgid	retrieves group ID

c) starting a process in linux

→ • In linux, a process is executed by creating a process & loading a program in newly created process.
• & fork system call is used to create a process and a program is loaded using execve system call.
• When a program is loaded, it run from memory location whose address is specified in executable file.
• By default starting address of the program is indicated by label _start.

Q.4. List & explain different system calls related to process management.

→

system call name	system call value	description
fork		Create a child process.
Execve		Execute a program
Exit		Terminate the current process
Nice		change the proces

[19UCS129]

PAGE NO.:
DATE: / /

Kill — priority. Send a sigle to kill a process.

8.5. Write 80386 assembly language program to pass input parameters for write system call. Also display text message "computer science" on terminal screen.

→
```
.section .data
output:
    .ascii "computer science"
output_end:
.equ len, output_end - output
.section .text
.globl _start
_start:
movl $4, %eax
movl $1, %ebx
movl $output, %ecx
movl $len, %edx
int $0x80
movl $1, %eax
movl $0, %ebx
int $0x80.
```