Name: Eva Islam

ID: 2020-1-60-273

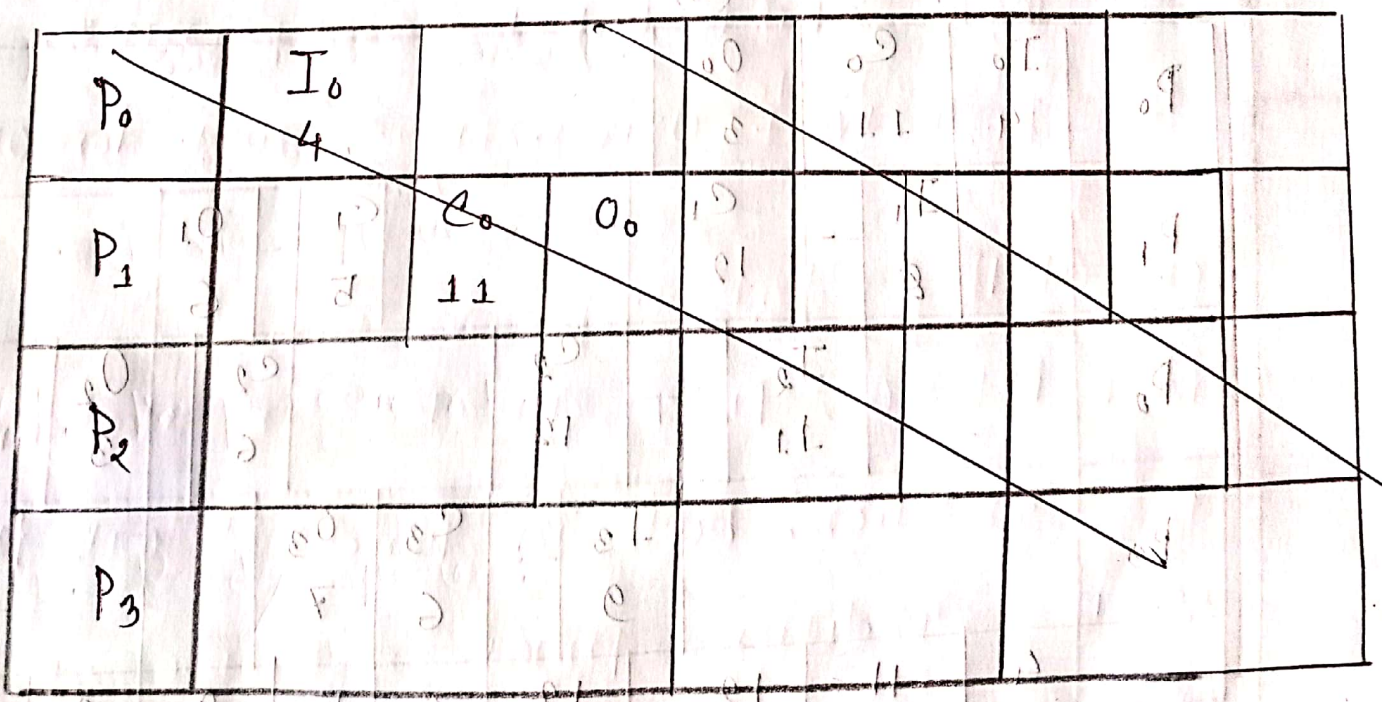
Sec: 02 Subject: CSE 325

Answer to the question No: 01

(a)

	P_0	P_1	P_2	P_3
Input	I_0 4	I_1 8	I_2 11	I_3 9
CPU	C_0 11	C_1 17	C_2 18	C_3 6
output	O_0 3	O_1 6	O_2 3	O_3 7

(b)



(a)

Process 0			Process 1			Process 2			Process 3		
I_0 4			I_1 8			I_2 11			I_3 9		
	C_0 11			C_1 17			C_2 18			C_3 6	
		O_0 3			O_1 6			O_2 3			O_3 7

4 11 3 8 17 6 11 18 3 9 6 7

= 103

(b)

P_0	I_0 4	C_0 11	O_0 3							
P_1		I_1 8	--	C_1 12			C_1 5	--	O_1 6	
P_2			I_2 11	--	C_2 12			C_2 6	--	O_2 3
P_3				I_3 9	--	C_3 6	O_3 7			
	4	11	12	12	6	5	8	3	= 61	

Parent

```
#include<unistd.h>
#include<stdio.h>
int main(){
pid_t pid;
pid_t pid1;
pid_t pid2;
pid = fork();
if (pid==0)
{ pid1 = fork();
printf("East west university");
}
else if (pid1<0){
pid2 = fork();
printf("CSE Department");}
else{
printf("EWU reopens on February 27, 2022");}}
```

Child 1

```
pid = fork();
if (pid == 0){
pid1 = fork();
printf("East west
        university");}
else if (pid1<0){
pid2 = fork();
printf("CSE Department");}
else{
printf("EWU reopens on
February 27, 2022");}}
```

Child 2

```
pid1 = fork();
printf("East west university");
}
else if (pid1<0){
pid2 = fork();
printf("CSE ...");
else{
printf("EWU ...");}}
```

Answer to the question NO: 02

For Parent:

When Pid ≠0 that's mean it is the parent. Then no other condition will check it will just print =("EWU reopens on February 27, 2022"). Just for once.

For Child 1:

When Pid = 0 that's mean it is the child. Then it will check the 'if' condition. In the 'if' condition child 1 will creat a new child Pid 1 and print = ("East West University"). Just for once.

For Child 2:

When Pid = 0, Pid 1 created in the first 'if' condition. But in here it will print the 'if' condition print =("East west University"). It will not go into the else if condition so Pid 2 will no creat.

Out put :

parent : EWU reopens on February 07, 2022.

Child 1 : East West University

Child 2 : East West University.

Answer to the question NO: 03

When interrupt signaled processor executes a routine called an interrupt handler to deal with the interrupt. An Interrupt Controller can handle 8 device together. There are two types of interrupt :

(1) Maskable (2) Non-maskable

In maskable interrupt it checks IE flag on priority. If IE = 1, the processor acknowledge the interrupt.

In non-maskable it does not check IE flag on priority.

There are 4 maskable and 3 non maskable interrupts. So there will be 1 interrupt controllen and 7 devices.

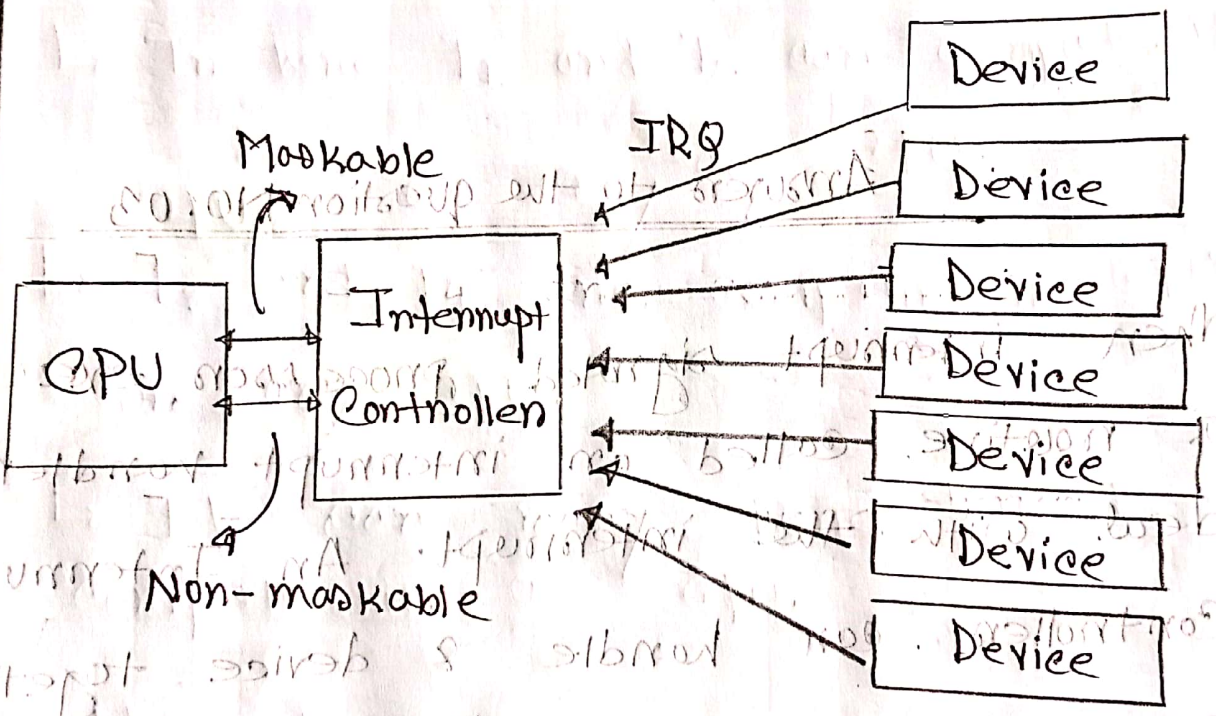


Fig: Interrupt Vector.

Answer to the question NO: 04

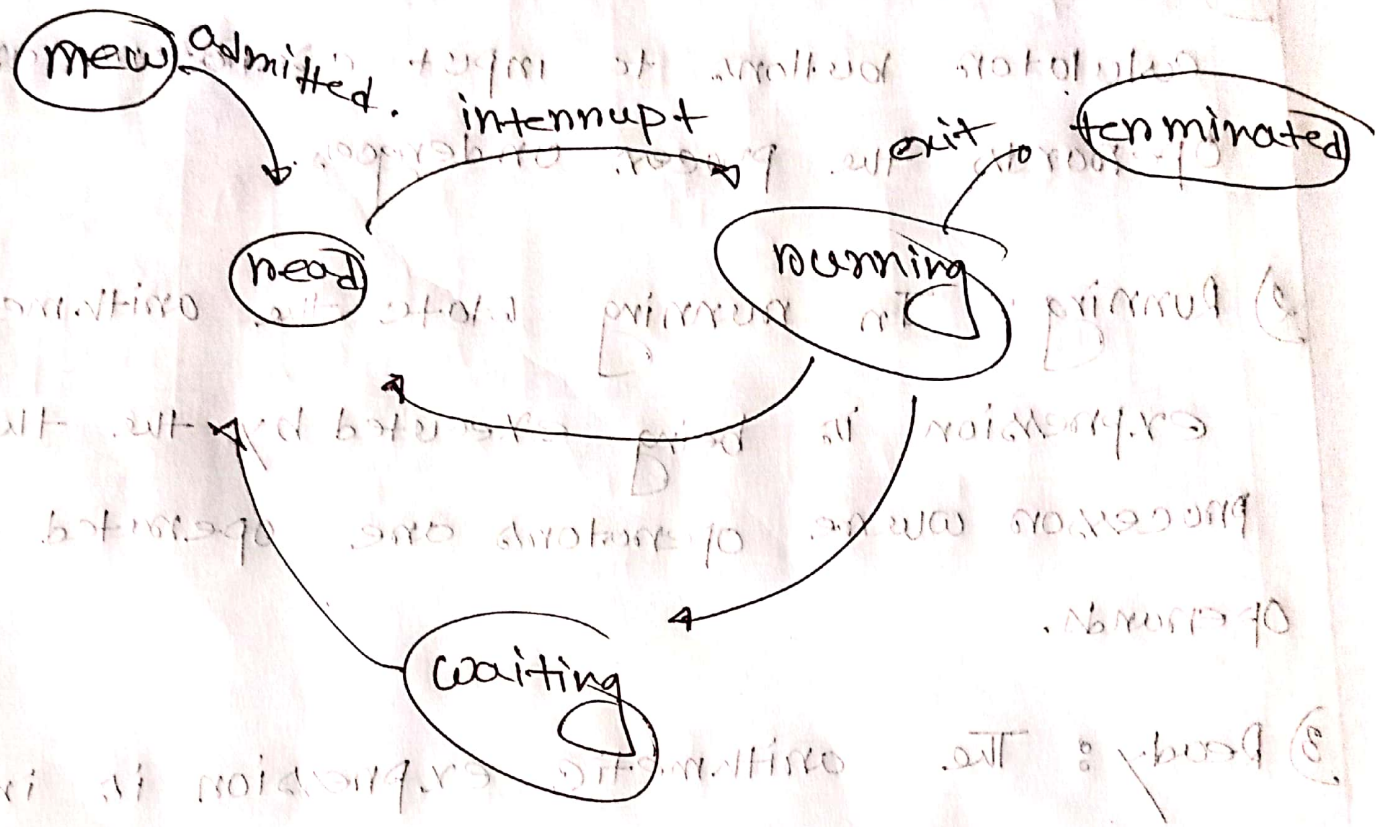
1) New: Whenever a user click on specific calculator buttons to input operators and operands the process undergoes.

2) Running: In running state the arithmetic expression is being executed by the the processor where operators are operated on operands.

3) Ready: The arithmetic expression is in ready state to be executed by cpu.

4) Waiting: The process is waiting for some event to occur.

5) Terminated: The process has finished execution.

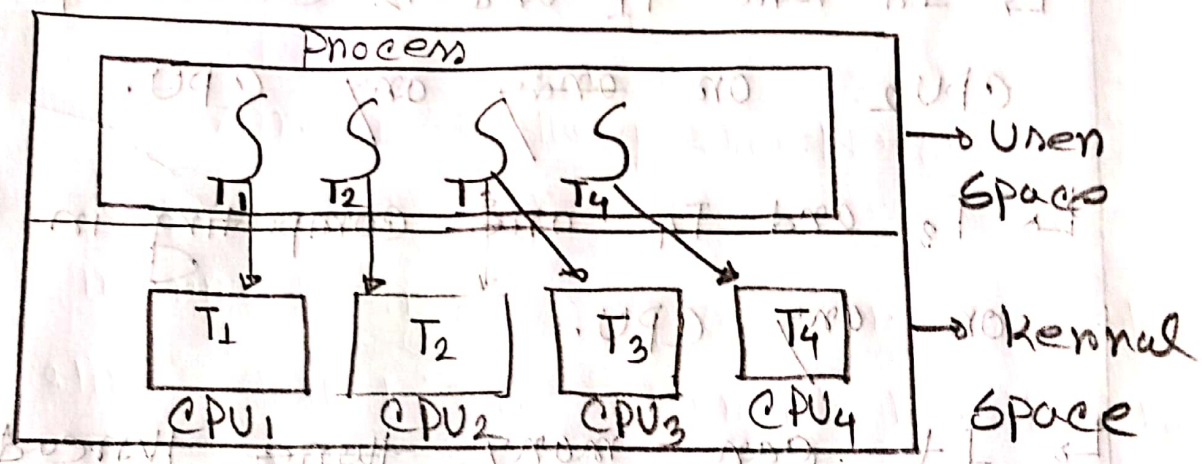

new
admitted
interrupt
exit
terminated
ready
running
waiting

Answer to the question NO: 05

Given that,

There are four thread T_1, T_2, T_3, T_4 and one running with four different processors namely CPU_1, CPU_2, CPU_3 and CPU_4.

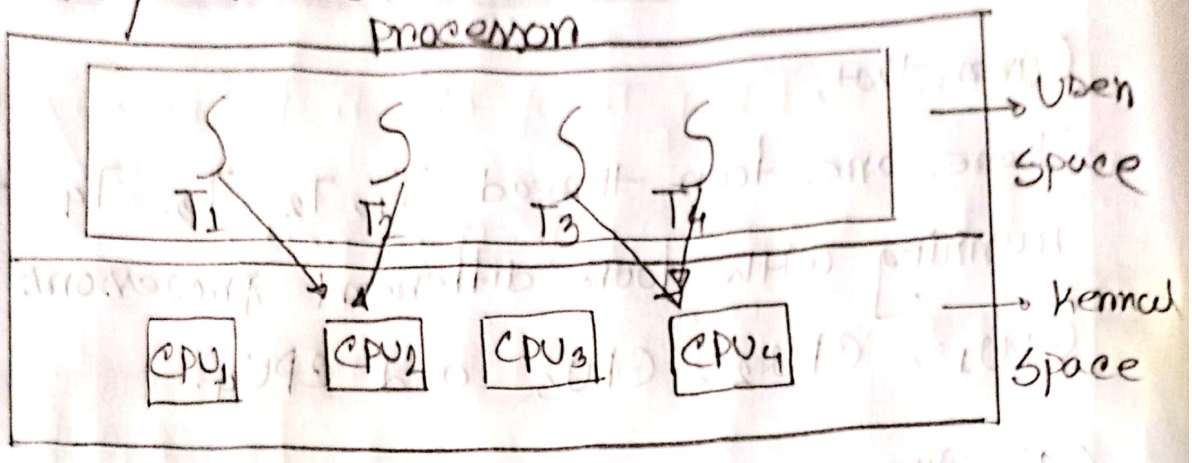
(1) One to one:



↳ Every singel thread with use spenate CPU in the kernal user.

↳ For every thread there will be indivual I/O operations.

② Many to One :

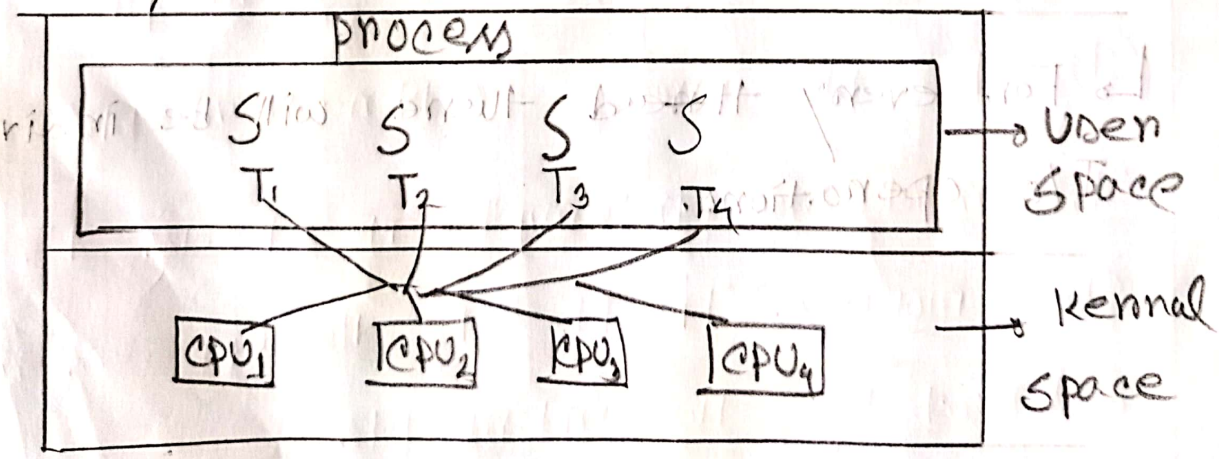


↳ In here T_1 and T_2 can compute in CPU_2 ~~and~~ any CPU.

↳ T_3 and T_4 are computing in CPU_4. or any CPU.

↳ It can many ~~there~~ thread can compute in a CPU.

③ Many to many :



↳ Any thread can compute in any CPU.