

Exceptions

Note Title

12/5/2007

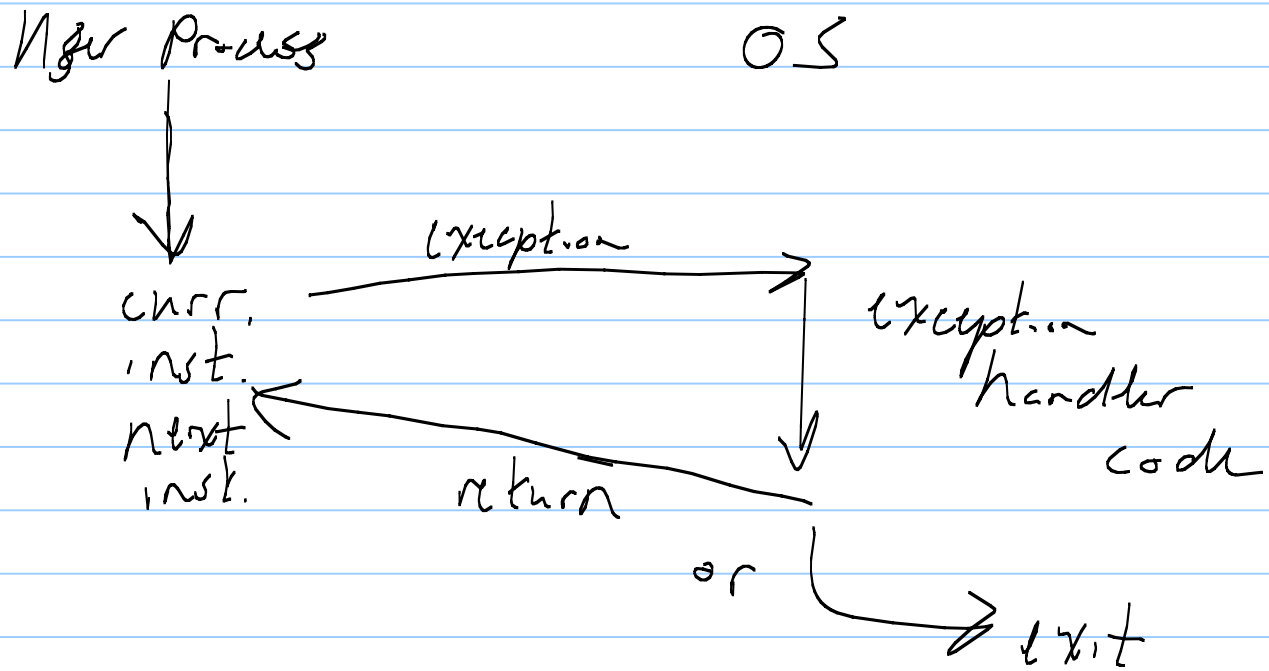
Normal flow control

- 1 next instruction immediately follows
- 2 " " result of jump branch call

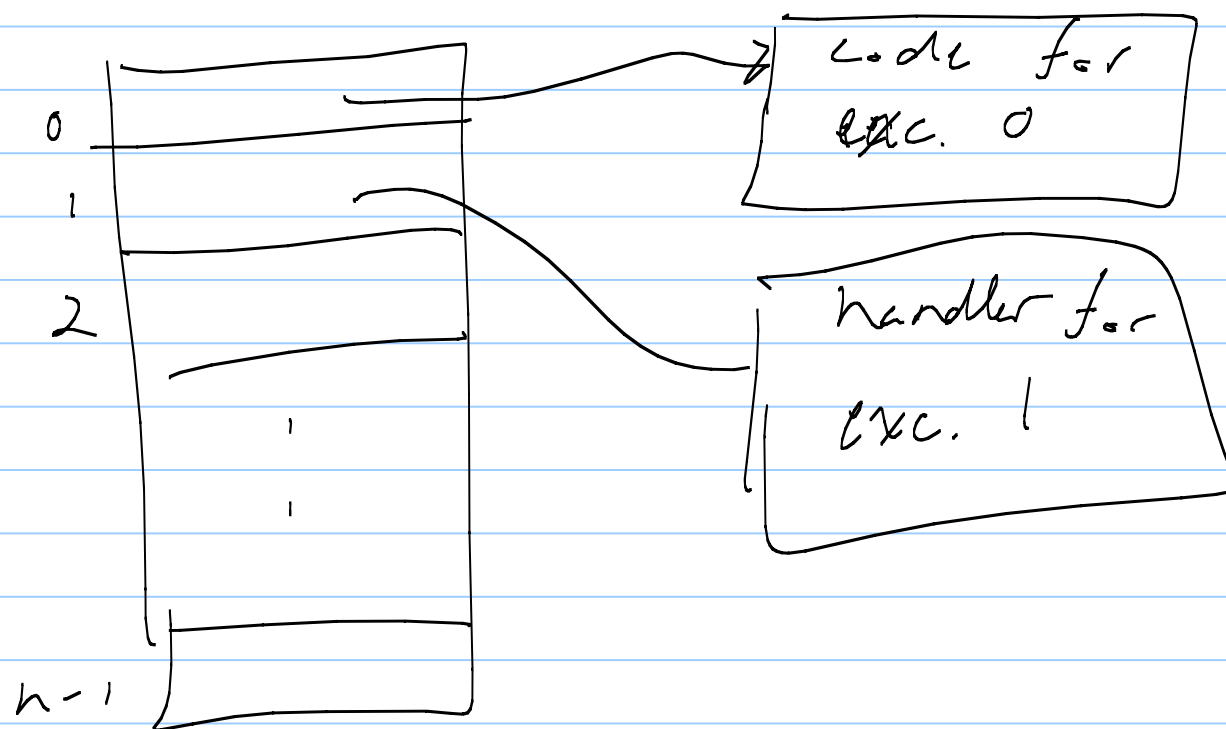
Exceptional flow control

- 1 external event (e.g. interrupt)
- 2 internal event (e.g. page fault, div by zero)
- 3 explicit request (e.g. ask for controlled resource)

name	timing	return
1 interrupt	async	next inst.
2 fault	sync	curr inst. (maybe)
2 abort	sync	never
3 trap	sync	next inst.



explicit table of pointers to exception handlers



Interrupts

Action : complete curr inst.
branch to vector'd int. service routine
return to next inst. (ISR)

Hints when writing ISRs

keep it short

queue important work for later

Faults

action branch to except. handler
handler decides whether to
abort or
return to curr inst.

Abort

action terminates process

Trap

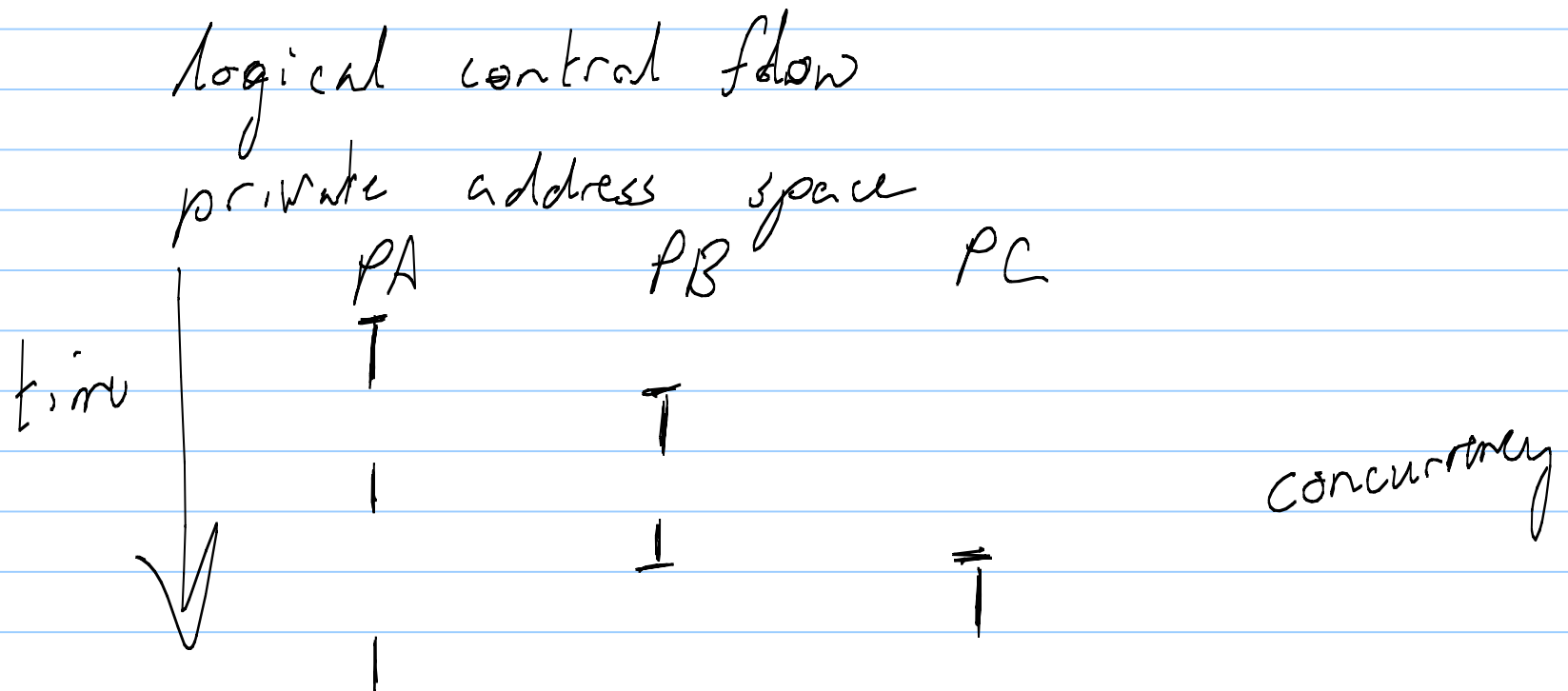
Action

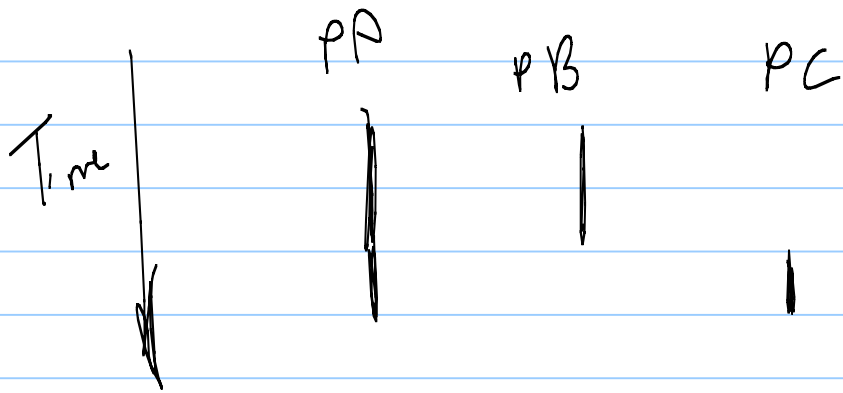
branch

OS provides requested service
M.T. to next inst.

Processes

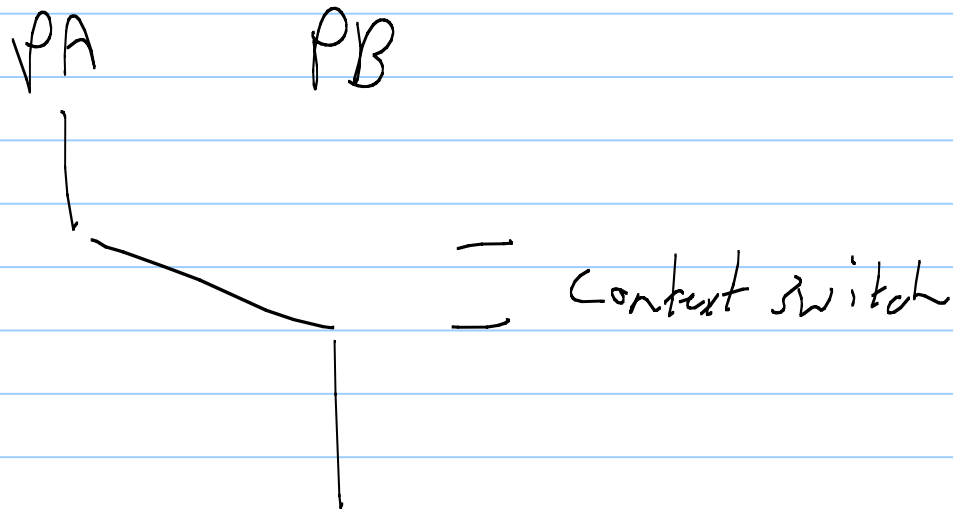
process is an instance of a running program





parallelism

Context switch

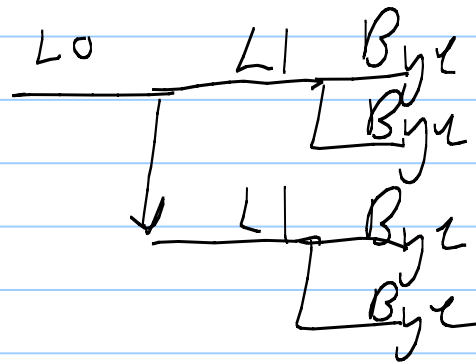


fork

```
int fork(void)
```

```
if (fork() == 0) {  
    printf("child");  
}  
else {  
    printf("parent");  
}
```

```
void foo() {  
    printf("Lo");  
    fork();  
    printf("L1");  
    fork();  
    printf("Bye");  
}
```



void exit (int status)

parent reaps status via wait ()

exec ()