Assignment

Request Byte ( 0 - )

One Malloc call to make a sudo ram - > 4096 (block of buffer)

After that, bytes are allocated and freed in that block.

+ <- debugging purposes

32 Bytes -> + \* 32 when I print it

Default -> -

A way of keeping records of what I gave out.

\*\*No zero page part needed to be coded

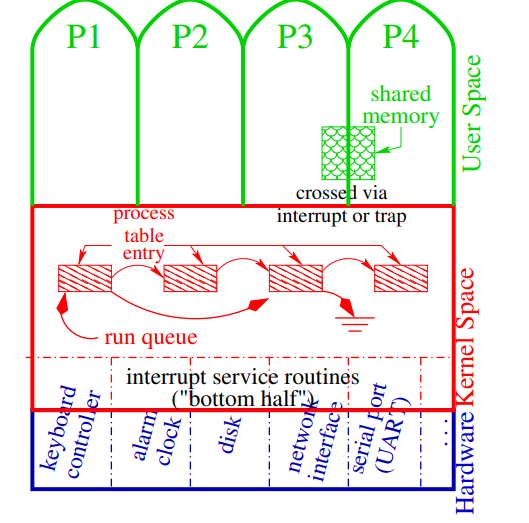
Slide 15 Modern (UNIX-type) systems: 3 types of device drivers

BLOCK -

NETWORK -

CHARACTER – 1BYTE at a time

Slide 16 Device Driver Functions



Top half – communicate with user kernel

Bottom half – interrupt service routines(slide 7)

Slide 19 Critical Sections in Device Driver

There needs to be a feature to prevent deadlocks( when something is allocating and the other thing is freeing at the same time)

**These are functions in the kernel**

int state = spltty() -> as soon as it is called everything is stopped?

int splx(int prevState) returns the interrupt mask to the previous state

int tsleep(void \*id, int flags, char \*mesg, int timeo)

id -> void\* -> Why void\*? To make sure it is unique in the kernel.(easy way to make an unique number of the thread id)

mesg-> debugging state, timeo -> timeout but After this time then wake up.

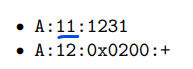
int wakeup(void \*id) -> A thread cannot wake up itself so another thread has to wake up the one that’s asleep.

* SO how does another thread know the id of sleeping threads?

Slide 21 Device Driver Skeleton – intr

It is not a function. It’s a interrupt handler.

\*\*\*\*\*\*\*Assignment tips



That is an number of the memory space if you call 11 and 11 again they are the same ones

4096 -> 4096 can’t be assigned .

Overhead should be at least 8

Read write …etc make pc get into kernel space

Spltty() ---🡪 splx(s) : Only one thread at a time in between

---slide 22

Stack ->(In scopes of functions, local variables) 0xFFFFFFFFFFFFF-> 1

Data -> (data segment -> const, strings, static, heap( malloc, free, pointers ))

Text -> code 0x0000000000000000-> 0

---slide23

Process -could have several threads

1Process Table Entry -> 1thread

1Process Table Entry -> 1thread

1Process Table Entry -> 1thread

Process Table? -🡪 study again

TTY -> what hardware?

STAT -> S -> sleeping R -> Running

---slide 25

Mutex : one at a time

semaphores: allow N-at-a-time

* One process -> one processor