Tema 1 Conceptos básicos:

SSOO:

Dos misiones--**gestionar los recursos para que los programas lo vean como una maquina completa**--**Abstraer los detalles del hardware**

**-**Crea y gestiona **procesos**(instancia ejecutable de un programa Necesita código(Texto), datos y metadatos).Solo uno a la vez en ejecución por núcleo. La CPU va dando turnos a procesos sin que otros hayan terminado necesariamente.

-Todo input/output pasa por el SSOO forzosamente, este ofrece una API formalizada hacia los programas para usar sus servicios.

-Kernel: parte mas importante del SSOO. Se ejecuta en modo supervisor con acceso a todo, los procesos user no pueden acceder a ciertas cosas.

**Heap y Stack:**

Stack: estructura dinámica que guarda info de las subrutinas que se ejecuten en un programa (Variables, metedos).

Heap: estructura dinámica que guarda datos en ejecución (memoria dinámica).

Interrupción: Se desaloja e proceso de la CPU y se ejecuta el Kernel, pude conceder otros procesos a la CPU al acabar.

Ciclo de vida de un proceso: estado listo(antes de estar en ejecución),en ejecución,bloqueado(cuando llama a una operación bloqueante(input/output), al acabar la op bloq pasa a listo otra vez),zombie(cuando termina de ejecutarse por completo).

Arranque del SSOO:

1Reset: Al encender el ordenador el pin reset recibe una señal y pone todos los registros a valor inicial, cont de programa incluido. Con esto la CPU ejecuta su código maquina. Registro especial Intel 64bits🡪0xFFFFFFF0, ARM🡪0xFFFF0000

Iniciador de hardware: Tras el código maquina de la CPU el equipo salta a un programa grabado en la ROM que detecta dispositivos conectados y realiza pruebas de funcionamiento(POST Power-On-Self-Test). Al acabar el POST se busca un dispositivo de almacenamiento masivo con un sector de arranque configurado siguiendo el orden de la BIOS. Hasta el momento todo en modo supervisor y memoria real. El iniciador carga en memoria el contenido del MBR (Master Boot Record) de la unidad de arranque y cede el control. Este ocupa los primeros 512 bytes y ahí puede haber un salto a un cargador del sistema (Linux, Windows) o un selector de arranque para elegir que SSOO arrancar. Nada de lo anterior es del SSOO aun.

Carga del sistema operativo: a continuación comienza la carga del SO residente (Kernel y algunos drivers). Segimos mod Supervis y Mem Real. El SO residente crea sus tablas de gestión(BCP, pags mem, …) y cuando acaba habilita las interrupciones y crea el proceso **init** en Unix que es el proceso padre de todos los demás procesos leyendo los scripts de /etc/init.d (en Linux modernos init = systemd)

-El kernel esta cargado en RAM para su rápido acceso desde la CPU. Se le despierta mediante **interrupciuones**

Interrupciones:

-La CPU sabe q hay una interrupción xq le llega al pin INTR o NMI.

-Cuando hay una interrupción: CPU termina de ejecutar la instrucción de cod maquina en curso—salva el PC y registro de Flags en su stack y carga el PC con la rutina de trata de interrupciones(parte del kernel). Se pone la CPU en modo super. Se recuperan los valores del PC y registro de Flags al terminar de tratar la interr y pone la CPU a modo user.

Tick—interrupción periódica que junto a las interrup de HW levantan al kernel sin inervencion del programador

Interrupción de SW—el SW fuerza una interrupción (se ofrecen bibliotecas para abstraer al user de esto)

POSIX—colección de APIs delos sistemas UNIX

Mecanismos para que los procesos se comunquen entre sí: 1Señales,2pipes,3colas de msg,4Sockets,5Mem shared

Todos se manejan desde el Kernel y se ofrecen con bibliotecas de alto nivel

-8086 no tenia mod supervisor (introducido en el 80286).

**-El espacio de direcciones se divide en low half y high half**

Conjunto de parámetros que se almacenan en el stack al invocar una función en un micro Intel🡪frame

Frame pointer es para facilitar el manejo de la pila

Raspbian es una versión Linux para arquitectura ARM.