Donaldo Sebastian Garcia Jimenez 19683

Laboratorio #5 SO

- Funcionamiento y sintaxis de uso de structs.
    a. Una estructura es una palabra clave que crea un tipo de datos definido por el usuario en C/C++. Una estructura crea un tipo de datos que se puede usar para agrupar elementos de tipos posiblemente diferentes en un solo tipo.

```
struct address
{
   char name[50];
   char street[100];
   char city[50];
   char state[20];
   int pin;
};
```

    b.
- Propósito y directivas del preprocesador.
    a. El preprocesador C, a menudo conocido como cpp, es un procesador de macros que el compilador C utiliza automáticamente para transformar el programa antes de la compilación.
- Diferencia entre * y & en el manejo de referencias a memoria (punteros).
    a. La diferencia es que por ejemplo se tiene
        i. int i: se declara un número entero
        ii. int* p: se declara un puntero a un entero
        iii. int& r = i: se declara una referencia a un entero y se inicializa refiriéndose a i.
- Propósito y modo de uso de APT y dpkg.
    a. Dpkg es una herramienta de sistema de bajo nivel para extraer, analizar, descomprimir e instalar o eliminar archivos .deb. Sin embargo, es mejor usar el Centro de software o Gdebi para instalar o eliminar archivos .deb que se obtuvieron de otras fuentes porque estos dos programas tienen resolución de dependencia. Sin embargo, la anatomía de una actualización usando dpkg y apt-get. Si, por ejemplo, llamamos a apt-get para instalar cheese, se analiza el árbol de dependencias y apt-get obtiene los archivos necesarios, que luego los pasa a dpkg para extraerlos, analizarlos e instalarlos en las ubicaciones correctas y configurarlos de acuerdo con los guiones dentro de ellos.
- ¿Cuál es el propósito de los archivos sched.h modificados?
    a. Definir estructuras, constantes y prototipos de funciones para implementarlas en distintas políticas de calendarización.
- ¿Cuál es el propósito de la definición incluida y las definiciones existentes en el archivo?
    a. Estas macros definen constantes para identificar las políticas de calendarización implementadas por el sistema.

Donaldo Sebastian Garcia Jimenez 19683

- ¿Qué es una task en Linux?
    a. Un proceso.
- ¿Cuál es el propósito de task_struct y cuál es su análogo en Windows?
    a. Su análogo en Windows sería KPROCESS y EPROCESS. Es la implementación de un PCB en linux.
- ¿Qué información contiene sched_param?
    a. La subrutina sched_setparam establece los parámetros de programación del proceso especificado por el parámetro pid en los valores especificados por la estructura sched_param a la que apunta el parámetro param. El valor del miembro sched_priority en la estructura sched_param es cualquier número entero dentro del rango de prioridad inclusivo para la política de programación actual. Los valores numéricos más altos para la prioridad representan prioridades más altas.
- ¿Para qué sirve la función rt_policy y para qué sirve la llamada unlikely en ella?
    a. La función auxiliar rt_policy se utiliza para decidir si una determinada política de programación pertenece a la clase de tiempo real (sched_rr y sched_fifo) o no. task_has_rt_policy determina esta propiedad para una tarea dada.
- Describa la precedencia de prioridades para las políticas EDF, RT y CFS, de acuerdo con los cambios realizados hasta ahora
    a. En orden de prioridad descendente, EDF, luego RT y luego CFS.
- Explique el contenido de la estructura casio_task.
    a. Esta estructura define una tarea calendarizable por la SCHED_CASIO_POLICY existente en el sistema. Se representa con un nodo en el árbol red-black para calendarizar la ejecución de casio_tasks.
- Explique el propósito y contenido de la estructura casio_rq.
    a. Esta función, al invocar la función de lista de tareas find_casio, obtiene el puntero a la tarea de estructura casio almacenada en la lista vinculada de la estructura casio_rq que apunta a la tarea p. Luego, actualiza la fecha límite absoluta e inserta casio_task en el árbol rojo-negro.
- ¿Qué es y para qué sirve el tipo atomic_t? Describa brevemente los conceptos de operaciones RMW (read-modify-write) y mappeo de dispositivos en memoria (MMIO).
    a. r
- ¿Qué indica el campo .next de esta estructura?
    a. Indica la siguiente clase de calendarización en la jerarquía de prioridades.
- Tomando en cuenta las funciones para manejo de lista y red-black tree de casio_tasks, explique el ciclo de vida de una casio_task desde el momento en el que se le asigna esta clase de calendarización mediante

sched_setscheduler. El objetivo es que indique el orden y los escenarios en los que se ejecutan estas funciones, así como las estructuras de datos por las que pasa. ¿Por qué se guardan las casio_tasks en un red-black tree y en una lista encadenada?

a. El red black tree organiza únicamente aquellas que están listas para ejecución.

- ¿Cuándo preemptea una casio_task a la task actualmente en ejecución?
    a. Cuando la tarea en ejecución no es una casio_task.