DIEGO CRESPO 19541

h. Aunque ya ha visto bastante código en C y probablemente tenga experiencia previa con el lenguaje, es importante conocer algunas de sus características para que el código que se provea en este laboratorio no sea copiado ciegamente, sino entendido en el proceso. Por ello, investigue y resuma:

- Funcionamiento y sintaxis de uso de `structs`.

  Struct es una palabra que esta reservada dentro del lenguaje C , esta tiene la funcion de indicar que lo que se declara es de tipo estructura de datos, la sintaxis es la siguiente:

  ```
  struct Diego{
        int age;
        char[20] name;
  ```

- Propósito y directivas del preprocesador.

  Estas tiene como fin en mente realizar operaciones en distintas directivas de preprocesador para facilitar y optimizar compilación.

- Diferencia entre `*` y `&` en el manejo de referencias a memoria (punteros).

  El * es la direccion donde esta una variable mientras que & es para ingresar a esta direccion.

- Propósito y modo de uso de APT y `dpkg`.

  Ambos son basados en dabian y permiten instalar paquetes , la diferencia se encuentra en apt debido a que este utiliza repositorios mas actualizados

  sudo
  apt-get install paquete, dpkg -i <archivo.deb>.

j.

- ¿Cuál es el propósito de los archivos `sched.h` modificados?

  Brinda los parametros de implementacion de politicas de calendarizacion.

- ¿Cuál es el propósito de la definición incluida y las definiciones existentes en el archivo?

  Crear los parametros de la calendarizacion y tambien se definen en el sistema al momento de iniciar.

k.

- ¿Qué es una task en Linux?

  Unidad de ejecucion basica.

- ¿Cuál es el propósito de `task_struct` y cuál es su análogo en Windows?

  Almacenar la informacion que esta relacionada con su calendarización, KPROCESS es el proceso similar en WINDOWS.

l.

- ¿Qué información contiene `sched_param`?
  Priorización de calendarización.

m.

- ¿Para qué sirve la función `rt_policy` y para qué sirve la llamada `unlikely` en ella?
  Sirve para determinar si la política de calendarización es de tiempo real . Unlikely verifica los cambios en la calendarización.
- ¿Qué tipo de tareas calendariza la política EDF, en vista del método modificado?

  Politicas de CASIO , SCHED_CASIO_POLICY.

n.

- Describa la procedencia de prioridades para las políticas EDF, RT y CFS, de acuerdo con los cambios realizados hasta ahora.

  1. EDF
  2. RT
  3. CFS

p.

- Explique el contenido de la estructura `casio_task`.

Este posee la informacion de una tarea calendarizada, proviene de un nodo de un arbol redblack , con etiquetas las cuales son deadlines.

q.

- Explique el propósito y contenido de la estructura `casio_rq`.

Posee el stack de tareas en ejecucion y la cola de casiotask

w.

- ¿Qué indica el campo `.next` de esta estructura?

  Este muestra la clase que le sigue en la calendarización dependiendo la prioridad.

x.

- ¿Por qué se guardan las `casio_tasks` en un red-black tree y en una lista encadenada?

  Esto debido a que el redblack tree organiza las tareas que si pueden ser ejecutadas.
- ¿Cuándo preemptea una `casio_task` a la task actualmente en ejecución?

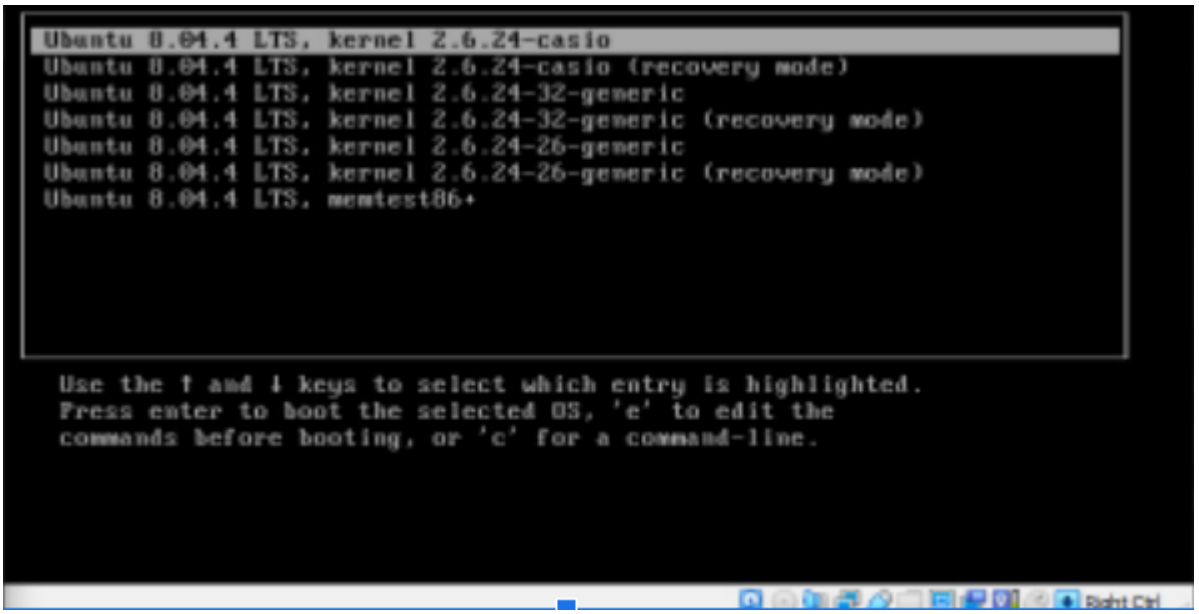
  Esto sucede cuando la deadline absoluta tiene un valor mayor a la deadline absoluta de una tarea pero del lado izquierdo en el arbol redblack.
- Ejecute nuevamente el archivo `casio_system` tal como se hizo al inicio del laboratorio, pero guardando los resultados en un archivo diferente. Adjunte ambos archivos de resultados de `casio_system` a su entrega, comentando sobre sus diferencias.

  Logramos ver que en lso archivos luego de adicionar esta calendarizacion se crean mas tareas con prioridad de 6.
- Ubique el archivo de log de eventos registrados por la calendarización implementada. Adjunte este archivo con su entrega.
- Agregue comentarios explicativos a los archivos `casio_task.c` y `casio_system.c` que permitan entender el propósito y funcionamiento de este código. Asegúrese de aclarar el uso de instrucciones y estructuras que no conozca (como, por ejemplo, los timers y la estructura `itimerval`). ¿Qué información contiene el archivo `system` que se especifica como argumento en la ejecución de `casio_system`?

  El archivo tiene la funicon de crear tareas de prueba, los procesadores a usar y los deadlines permiten simular la calendarizacion.
- Investigue el concepto de aislamiento temporal en relación a procesos. Explique cómo el calendarizador `SCHED_DEADLINE`, introducido en la versión 3.14 del kernel de Linux, añade al algoritmo EDF para lograr aislamiento temporal.

  El aislamiento temporal se puede definir como una capacidad en un SO de realizar tareas de manera independiente , donde los errores que suceden en otras tareas concurrentes no sean

propagados.


Ubuntu 8.04.4 LTS, kernel 2.6.24-casio
Ubuntu 8.04.4 LTS, kernel 2.6.24-casio (recovery mode)
Ubuntu 8.04.4 LTS, kernel 2.6.24-32-generic
Ubuntu 8.04.4 LTS, kernel 2.6.24-32-generic (recovery mode)
Ubuntu 8.04.4 LTS, kernel 2.6.24-26-generic
Ubuntu 8.04.4 LTS, kernel 2.6.24-26-generic (recovery mode)
Ubuntu 8.04.4 LTS, memtest86+
Use the ↑ and ↓ keys to select which entry is highlighted.
Press enter to boot the selected OS, 'e' to edit the
commands before booting, or 'c' for a command-line.