Software para sistemas empotrados y dispositivos móviles

Febrero 2019

Grado Ing. Software

1. Diseñar e implementar un programa para controlar el nivel de agua de un tanque. Existen una entrada de agua y una salida que son independientes y cada una está controlada por una válvula, la cual está preparada para dejar salir/ entrar un litro de agua por segundo, en el caso de la entrada y cada dos segundos en el caso de la salida.

El sistema será simulado por tres tareas (hebras), de las cuales las dos primeras están encargadas del control de la válvula en entrada y salida respectivamente, y la tercera tarea (hebra) será la encargada de gestionar y monitorizar el nivel del agua del tanque.

La tarea de control de entrada de agua simulará, mediante una tarea periódica con periodo de 1 sg, el llenado del agua en el tanque, y enviará una **señal** , con una probabilidad del 60% a la tarea de gestión del tanque.

La tarea de control de salida de agua simulará, mediante una tarea periódica con periodo de 2 sg, la salida de agua del tanque, y enviará una **señal,** con una probabilidad del 40%, a la tarea de gestión del tanque.

La tarea de gestión y monitorización del nivel de agua del tanque es una tarea que se encuentra, de forma permanente, a la espera de recibir notificaciones (señales) de las tareas de control de E/S de agua, y actualiza el nivel de agua según las señales recibidas. Además, también mostrará el valor del nivel del agua en el tanque.

Diseñar la solución con los mecanismos de temporizadores y hebras de POSIX. La gestión de las tareas periódicas de control de tornos se realizará mediante **temporizadores** (create_timer, etc).

Para gestionar la probabilidad en la generación de señales de entrada y salida, se puede generar un número aleatorio entre 0 y 100 (*rand()%101*) y comprobar si el número aleatorio supera un determinado umbral. Es necesario inicializar el generador de números aleatorios al principio del programa mediante *srand(time(NULL)).*

2. Se desea implementar el control de la luminosidad para 2 habitaciones de una casa (habitación 1 y habitación 2). **Cada** habitación lleva asociada una **tarea de control,** para vigilar la intensidad de la luz. Cada tarea realiza de forma periódica (según el periodo y plazo indicado en la tabla) el siguiente procedimiento:

- Calcula el valor de la luminosidad de la habitación. Para ello, los valores de luminosidad se pueden generar aleatoriamente, inicializando el generador de números aleatorios al principio del programa mediante *srand(time(NULL)),* y generando los números aleatorios dentro del rango [MIN..MAX] como *MIN+(rand()%(MAX-MIN+1))*. Nota: esta generación de números aleatorios debe protegerse mediante exclusión mutua.

- Si la luminosidad es menos del 80% de la luminosidad máxima MAX, mostrará un mensaje de "Activación de la luz artificial"

- Si la luminosidad de la habitación es mayor de un 80 % y menor de un 90% de la luminosidad máxima MAX, mostrará un mensaje de "luz óptima".

- En el caso de que se superara el 90% de la temperatura máxima MAX, mostrará un mensaje de aviso "Activación cortinas".

En la habitación 1 existe material que no puede tener poca luz y su control más crítico que la habitación 2, por lo que la tarea de control del primera habitación es prioritaria sobre el segunda. Además, existe una tarea de monitorización, con mínima prioridad *MonitorLuz,* que informa periódicamente de la luminosidad de las habitaciones con un periodo de 4 s.

Los requisitos temporales de las tareas son:

Tareas	**Periodo**	**Plazo**
Habitación 1	2000 ms	1500 ms
habitación 2	3000 ms	2500 ms
Monitor	4000 ms	4000 ms

Nota 1.- Asignar prioridades en función de los requisitos temporales y recordar que se debe limitar la inversión de prioridades.

Nota 2.- El comportamiento periódico de las hebras deberá diseñarse con *clock_gettime* y *clock_nanosleep*.

SOLO ALUMNOS QUE OPTAN POR LA EVALUACION CONTOINUA Y NO HICIERON EL CONTROL

Problema (1 punto)

Consideremos el siguiente conjunto de tareas:

Tarea	**Periodo**	**Plazo**	**Tiempo de ejecución en el caso peor**	**Recursos utilizados**
A	90	70	2	R1,R3
B	100	100	9	R1,R2,R5
C	30	30	4	R4,R5
D	1000	600	11	R1,R3
E	150	95	12	R1,R2,R3,R4

Los tiempos de ejecución en el caso peor de los recursos indicados en la tabla son los siguientes:

R1	0.3
R2	0.2
R3	0.2
R4	0.1
R5	0.1

1. Calcular los factores de bloqueo suponiendo que el planificador usa **techo de prioridad** y calcular el tiempo de respuesta de las tareas del sistema**. La asignación de las prioridades debe ser óptima. (0,75)**
2. Supongamos un planificador basado en tick con un Tick de 1 y un C_{tick} de 0,2 y un tiempo de tratamiento de las colas C_{cola}=0,1. Incluiremos el tiempo de cambio de contexto de 0,1. Calcular el tiempo de respuesta de la tarea **C teniendo en cuenta todas estas sobrecargas. (0,25)**