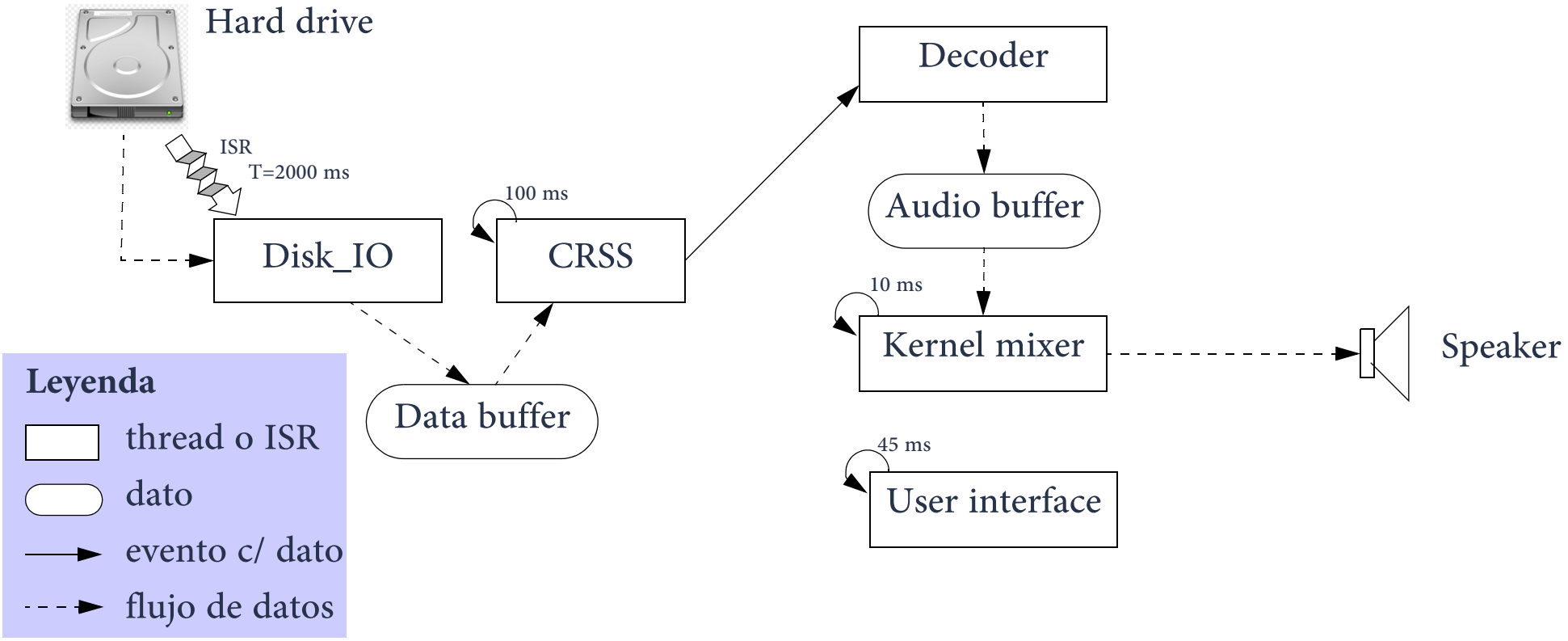
Desarrollo de software para sistemas empotrados

Examen Febrero 2023

Introducción

El objetivo de este ejercicio es realizar el análisis y diseño arquitectónico de una parte de un reproductor de audio digital, que se corresponde con el siguiente diagrama de bloques:



Descripción general

El sistema realizará las siguientes acciones en software:

- *Disk IO*: Actúa como rutina de servicio de interrupción para la interrupción procedente del disco duro. Pertenece al *driver* del disco duro. Lee bloques de datos de audio del disco duro y los deposita en el *data buffer*
- *CRSS*: Actúa como servidor de audio. Lee en el *data buffer* datos de un frame de audio y se los envía mediante un mensaje al decodificador de audio (*decoder*)
- *Decoder*: Se despierta al recibir un frame de audio procedente del servidor *CRSS*. Decodifica el frame y produce audio reproducible que deposita en el *audio buffer*
- *Kernel mixer*: recoge datos de sonido del *audio buffer* y los envía al altavoz
- *User interface*: realiza funciones de interfaz con el usuario

Objetos de datos

El *data buffer* y el *audio buffer*:

- Utilizan un diseño de cola circular que no requiere exclusión mutua
- Cada uno de ellos tiene una función para depositar un dato y otra para recoger un dato
- Al no tener exclusión mutua no influyen en el comportamiento temporal

Software funcional

La funcionalidad del software de las actividades principales está ya desarrollada en forma de 5 funciones:

- `disk_io()`
- `crss()`
- `decoder()`
- `kernel_mixer()`
- `user_interface()`

Los argumentos de estas funciones así como los tipos de datos que se almacenan en los buffers no requieren ser descritos en el diseño, por no ser relevantes para el análisis temporal

Requisitos funcionales

1. El disco duro genera una interrupción periódica, con un periodo de 2000 ms. Esta interrupción es atendida por la función `disk_io()` incluida en el driver del dispositivo *hard drive*. La función, al acabar, escribe datos en el *data buffer*
2. El sistema debe invocar a `crss()` periódicamente, con un periodo de 100 ms. Al acabar envía un evento con datos al decodificador
3. La función `decoder()` se ejecuta al recibir el evento procedente del servidor *CRSS*. Durante su ejecución envía los datos de sonido al *audio buffer*
4. El sistema debe invocar a `kernel_mixer()` periódicamente, con un periodo de 10 ms. El *kernel mixer* recoge datos del *audio buffer* y los envía al altavoz
5. El sistema debe invocar a la función `user_interface()` periódicamente, con un periodo de 45 ms

Requisitos no funcionales

6. Las actividades *disk IO* y *user interface* tienen plazos iguales a los periodos
7. La actividad *kernel mixer* tiene un plazo de 20 ms
8. Existe un plazo de principio a fin de 200ms desde el inicio del periodo de la función `crss()` hasta que el decodificador de audio finaliza
9. Los requisitos temporales deben validarse con un análisis de planificabilidad.
10. Los modelos *temporal* y *arquitectónico* solo tendrán en cuenta los elementos software, incluido el driver del dispositivo *hard drive*. En cambio, el modelo de requisitos tendrá en cuenta también los componentes físicos, tales como el disco duro y el altavoz
11. En los modelos *temporal* y *arquitectónico* no es preciso modelar los buffers de datos, pues no influyen en el comportamiento temporal

Requisitos no funcionales (cont.)

12. Los tiempos de ejecución de peor y mejor caso (C y C^b) medidos para las funciones software ya disponibles son los siguientes. También se indican las prioridades inicialmente asignadas por el diseñador:

Función	C (ms)	C^b(ms)	Prioridad
`disk_io()`	6.0	5.0	32
`crss()`	2.0	1.0	15
`decoder()`	30.0	25.0	9
`kernel_mixer()`	2.0	1.1	24
`user_interface()`	5.25	3.2	8

Otros requisitos no funcionales

Los desarrollos y el software básico estarán basados en una plataforma que dispone de un procesador con un sistema operativo gobernado por eventos, con prioridades fijas y las siguientes características

Propiedad	Valores (ms)
Rango de prioridades	1...32
Rango de prioridades de interrupción	32...32
Tiempo de cambio de contexto (ms)	0.002...0.003
Overhead de las interrupciones	0.002...0.002
Tipo de Temporizador	Alarm Clock
Overhead del temporizador (ms)	0.001...0.002

Ejercicios

1. Dentro del proceso de análisis de requisitos, generar uno o varios diagramas UCM para los requisitos del sistema
2. Modelar con AADL una arquitectura con 4 flujos de eventos:
   - rutina de interrupción del disco duro
   - servidor CRSS y decodificador
   - kernel mixer
   - interfaz de usuario

Ejercicios

3. Modelar con MAST la arquitectura del sistema para realizar un análisis de planificabilidad inicial y responder a estas preguntas:
   - ¿Es planificable el sistema con las prioridades asignadas por el diseñador?
   - En caso negativo, ¿Qué prioridades harían que el sistema fuese planificable?

Entregar 4 ficheros

Un informe en pdf con:

- diagrama(s) UCM
- diagrama AADL de nivel de sistema, con el máximo nivel de detalle
- una captura de pantalla de los resultados de MAST
- las respuestas a las preguntas planteadas

Workspace de UCMNav comprimido

Workspace de OSATE comprimido

Ficheros del modelo MAST en un archivo comprimido