Universidad Zaragoza

Proyecto Hardware
Trabajo de la asignatura

Implementación de un juego sudoku

Guillermo Robles González - NIP: 604409

Supervisado por:
Javier Resano Ezcaray (coordinador)
María Villarroya Gaudó
Enrique Torres Moreno
Jesús Alastruey Benedé
Darío Suárez Gracia

14 de diciembre de 2015

Índice

1. Resumen

2. Introducción

Apoyandonos sobre el trabajo desarrollado en la practica anterior, el desarrollo del mismo consistió en la creación de una librería grafica particular (sudoku_graphics) encargado de actuar de Proxy entre el método Main y la librería Lcd, permitiendonos generalizar el método Main, de tal forma que en la situación en la cual se desee llevar este proyecto a otro hardware solo se tendría que cambiar el modulo Proxy (sudoku_graphics), y no será necesario modificar la función Main. Esta propiedad de generalización ha sido uno de los objetivos primarios de este proyecto, intentando siempre que sea posible el mover cualquier función generalizable a su lugar adecuado (Como ejemplo practico de esto, en el marcaje de errores se ha elegido el marcar en negativo las casillas error, Para ello se creó una función que inicialmente estaba en el Main, pero primero fué movida a sudoku_graphics, y finalmente fué completamente generalizada y movida a Lcd, de esta forma si en algún momento se ha de desarrollar otro proyecto con este hardware, la librería ya esta preparada para soportar más operaciones)

3. Objetivos

Los objetivos se organizan en 2 conjuntos, los objetivos propuestos por el profesorado, y un conjunto de objetivos autopropuestos por el alumno.

Objetivos del profesorado:

- Evitado de rebotes en los botones
- Implementación de un juego de sudoku
- Conocimiento de un sistema hardware particular

Objetivos del alumno:

- Generalización de funciones y funcionalidades
- Organizado del código adecuada, uso de archivos .h
- Uso de un sistema de documentado en código estandar (Doxygen)
- Uso de un sistema de control de versiones (Git)

4. Partes del proyecto

4.1. Trabajo previo

El trabajo a realizar requiere un conjunto de funcionalidades previas, que pese a no estar explicitamente incluidas en el producto final, nos permiten desarrollar adecuadamente las secciones siguientes. Estas funcionalidades son:

- Conocimiento del funcionamiento de los timers, y uso de los mismos tanto para medidas de tiempo como para programar tareas.

- Implementado de una sencilla pila de debug, que nos permite almacenar estampillas temporales e informacion deseada, y nos simplifica por ejemplo medir cual es el tiempo que duran los rebotes.
- Manejo de excepciones, tanto para informar al programador como para eventualmente resolveralas y recuperarnos de las mismas.

4.1.1. Timers

El control de sucesos temporales es vital, tanto para ejecutar funciones periódicamente (por ejemplo, capturas de datos) o con un cierto retardo (por ejemplo, esperas).

En el proyecto, el uso de los timers en los rebotes es usado tanto para, y basado en el proyecto dado (Un sencillo programa que parpadea alternativamente 2 leds) se pudo generalizar una simple librería de tiempos que cumplía las caracteristicas necesitadas. El timer usado es el 2, lo cual deja también el 3 inutilizable

La unidad de medida inicial elegida fueron los µsegundos, que posteriormente fué cambiado a milisegundos, dado que en la medida de tiempos de juego (que pueden llegar a las horas) ocurrían problemas de overflow.

4.1.2. Pila de Debug

Una pila (lo llamamos pila pero la estructura implementada sería más cercana a una cola circular) de Debug es una zona especial de la memoria, en la cual introducimos información que podemos consultar, que en nuestra situación es usada para reconocer y ordenar sucesos temporales, como excepciones o interrupciones, que guarda una estampilla temporal, un suceso e información extra.

En nuestro caso la pila se ha escogido de tamaño 20, situada en la zona de las pilas, por debajo de la pila de usuario, de tal forma que podamos localizarla con facilidad.

Esta pila utiliza el Timer programado previamente para la generación de las estampillas temporales, por lo que para su correcto funcionamiento el módulo Timer2 ha de ser inicializado por el usuario de esta pila.

4.1.3. Excepciones

Las excepciones son sucesos inesperados, generalmente asociados a situaciones de error, que nos permiten reconocer situaciones inesperadas, y advertir al programador, al usuario o incluso intentar recuperarnos de una excepcion para continuar el programa o finalizarlo suavemente.

El procesador ya incluye un sencillo sistema de gestión de Excepciones, que simplemente atasca o reinicia la placa. Para sobreescribir este sistema simplemente se han de cambiar los punteros `pISR_UNDEF`, `pISR_SWI`, `pISR_PABORT`, `pISR_DABORT` ha una función, que será llamada en la situación de que salte alguna de las excepciones.

Puntero	Excepción que lo llama	Descripción
`pISR_RESET`	Reset	Reset por hardware de la placa
`pISR_UNDEF`	Undefined Instruction	Opcode no reconocido
`pISR_SWI`	Software Interrupt(SWI)	Lanzado mediante la instrucción swi
`pISR_PABORT`	Prefetch Abort	Error al realizar el fetch de la instrucción
`pISR_DABORT`	Data Abort	Error al leer los argumentos de una instrucción

Para reconocer la excepcion en la cual nos encontramos se puede mirar el modo de procesador, dado que el lanzado de ciertas excepciones fuerza el procesador a modos particulares, por lo que mirando el modo actual se puede detectar la excepción lanzada, tanto para intentar recuperar (por ejemplo, en el caso de SWI) como para poder reconocer y guardar la excepción ocurrida.

Una situación en la cual no se puede reconocer la excepción por el modo es entre `pISR_PABORT` y `pISR_DABORT`, dado que ambas pasan al modo `Abort`; o entre `pISR_RESET` y `pISR_SWI` dado que ambas pasan al modo `Supervisor`; en estos casos la única posibilidad es utilizar varias funciones gestoras, de tal forma que por la propia función en la que estemos nos indique que excepción estamos manejando.

Dado el alcance de este proyecto, no se ha decidido complicar la librería de gestión de excepciones con estas consideraciones, pero en una situación en la cual este módulo quiera generalizarse, habría que tener en cuenta tanto la gestión correcta de algunas excepciones (por ejemplo `pISR_RESET`) como la recuperación de aquellas que lo sean (como `pISR_SWI`).

4.2. Rebotes en los botones

El problema de los rebotes ha sido uno de los primeros problemas a los que tenemos que enfrentarnos en un proyecto de estas caracteristicas, y uno de los más complejos.

Este problema aparece cuando nos enfrentamos a hardware (botones) reales. Mientras un botón teórico da una señal similar a la siguiente: Un botón real sigue un patrón más cercano a este:

Estos rebotes son un problema grave, dado que mientras el usuario cree que ha pulsado el botón una única vez (asumiendo como una vez una pulsación y un soltado) el sistema ha recibido múltiples pulsaciones, y por tanto reacciona a todos ellos (en nuestro caso, el sistema incrementa su contador interno múltiples veces).

Existen múltiples soluciones a este problema, tanto por hardware como por software.

Dentro de los solucionadores por hardware aparecen 2 grandes grupos, los basados en SR latches y los basados en circuitos RC, ambos suficientemente efectivos para la mayoría de situaciones.

Desafortunadamente, el hardware escogido no tiene estos sistemas, lo cual nos fuerza a utilizar solucionados por software. Al igual que para el solucionado por hardware, para el solucionado por software

4.3. Interaccion

4.4. Graficos

5. Metodología

6. Resultados y características

7. Conclusiones

La finalización del trabajo fué satisfactoria, cumpliendo la mayoría de los objetivos propuestos. Algunos

7.1. Margen de mejora

- El módulo Boton no es suficientemente general, dado que esta integrado tanto el control del 8 segmentos como algunas variables de control (next, update...). Esto se podría arreglar haciendo que en el constructor el botón reciba 2 punteros a función, que serán las que llamará cuando sea pulsado alguno de los botones; además de un conjunto de parámetros (Si el botón tiene autorepetición, tiempo de repetición...). No se ha realizado dado que se considera que complicaría el trabajo innecesariamente.
- El modulo Lcd ofrecido por el profesorado era bastante pobre, por ello se ha extendido con algunas funciones que nos permiten generalizarlo más.
- Por falta de tiempo, no ha sido posible añadir música ni efectos de sonido.
- Las animaciones de los creditos son bastante crudas, y tienen posibilidad de mejorar.

8. Bibliografía

- Manuales de consulta ofrecidos por el profesorado
- Proyectos de la placa ofrecidos por el profesorado
- `http://www.eng.utah.edu/~cs5780/debouncing.pdf`
- `http://infocenter.arm.com/help/index.jsp` (Especialmente el manual de referencia de ARM7)
- `http://www.sudoku-solutions.com/`