**Escuela de Ingeniería Informática**

Grado en Ingeniería Informática



**Periféricos e Interfaces**

Práctica de laboratorio

**Módulo 1 – Práctica 1**

MIPS-32: Excepciones e interrupciones

Nombre: Zabai Armas Herrera

Grupo: 1.18.44

Curso: 2015/2016

Periodo de realización: 3 semanas

**Índice**

1. [Objetivos](#Objetivos)

2. [Procedimiento](#Procedimiento)

3. [Realización práctica](#Realización)

4. [Conclusión](#Conclusión)

­­­­­­**Objetivos**

* Familiarizarse con el simulador PCSpim para la ejecución y depuración de programas en lenguaje ensamblador del procesador MIPS.
* Aprender aspectos básicos del funcionamiento de los dispositivos hardware simulados en el PCSpim.
* Diseñar programas con operaciones de entrada/salida con los periféricos emulados del en el PCSpim utilizando el método de sincronización de consulta de estado.
* Entender la gestión de interrupciones en la arquitectura MIPS-32 a través del estudio del manejador de excepciones e interrupciones del PCSpim (“exceptions.s”).
* Programar una aplicación con operaciones entrada/salida con los periféricos simulados en el PCSpim, combinando la consulta de estado y el sistema de interrupciones.

**Procedimiento**

Para la elaboración de la práctica se ha utilizado las siguientes herramientas:

* Simulador: PCSpim
* Editor de texto: Sublime Text 3
* Diseño gráfico: Adobe Illustrator CC

Los pasos para construir el programa han sido las tareas del apartado de “Realización práctica básica” en la ficha de la práctica.

1. Primero la elaboración el método de “PrintCharacter”, el cual tuve que hacerlo primero con llamadas al sistema para repasar el lenguaje ensamblador y finalmente con consultas de estado.
2. Realizar la rutina “Sentence” para el bucle de impresión de la frase utilizando el método que imprime carácter a carácter.
3. Analizar el manejador de excepciones con la producción de un diagrama de flujo del mismo.
4. Programar las interrupciones por teclado y timer. Implementando tanto su habilitación como sus rutinas de servicio e interrupción en el programa principal y en el manejador.
5. Implementar la funcionalidad de comprobar en la rutina de interrupción del teclado la tecla pulsada para poder identificar la combinación Ctrl+R y realizar otro procedimiento para fijar la hora local.
6. Por último, crear métodos opcionales para mejorar la práctica como pausar el programa, cambiar el delay, configurar el registro compare…

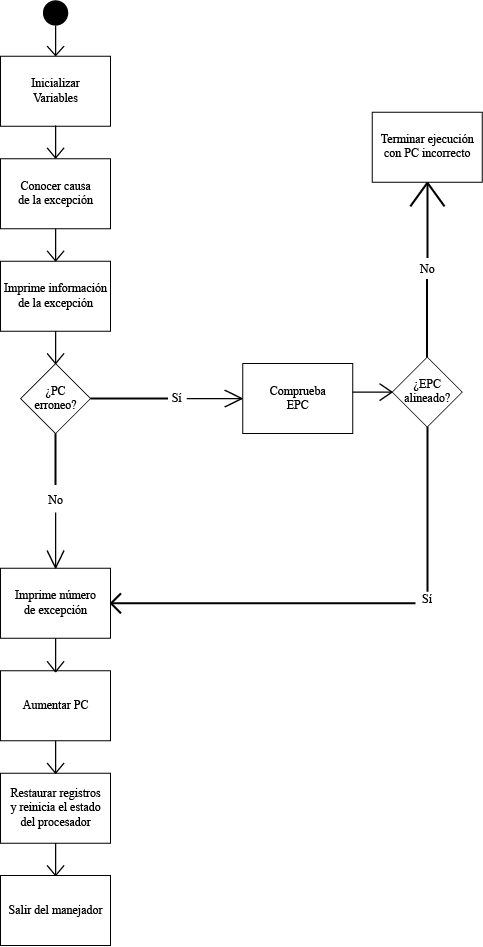
**Realización practica**

El programa está dividido en varias secciones bien diferenciadas: programa principal, utilidades y las subrutinas que están relacionadas con la gestión de interrupciones.

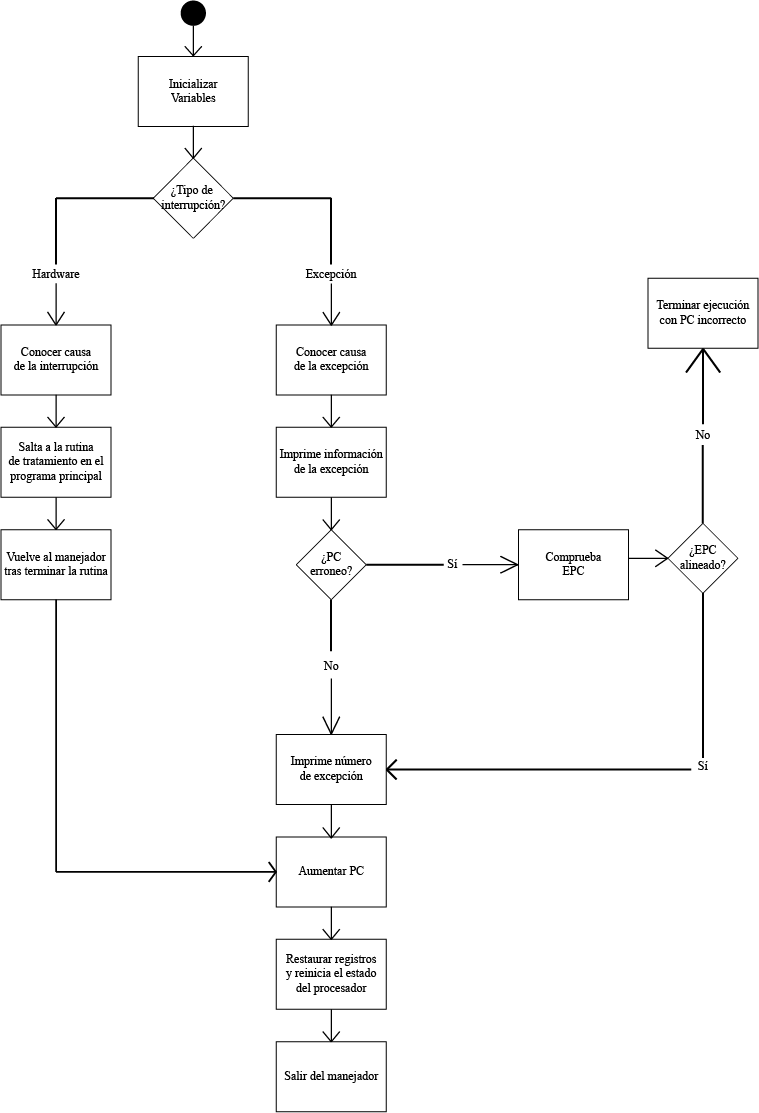
1. El programa principal no tiene mucha complejidad ya que sólo consta de dos llamadas para habilitar las interrupciones del teclado y timer, además de un bucle para imprimir una frase de forma indefinida.
2. En las utilidades tenemos la rutina para imprimir un carácter por consulta de estado y un delay simple.
3. Las subrutinas de las interrupciones es la parte más “compleja” del programa dada su cantidad y longitud de métodos. Por un lado están los habilitadores de excepciones, los cuales configuran los registros del coprocesador 0 (Status, Count, Compare) y de control del teclado. Por otro están las rutinas de tratamiento de las interrupciones que implementan la funcionalidad requerida de la práctica.
   1. La hora local se ha programado de forma que cada entero es una variable, por lo que a la hora de fijar una hora si el número introducido es de dos cifras se hace uso de una división para almacenar cada uno en una variable (siempre y cuando el valor introducido sea válido, es decir, si se introduce 26 horas dará un mensaje de error y continuara la ejecución del programa).
   2. Para las demás rutinas restantes no hace falta explicación ya que son bastante simples.

Con respecto al manejador se han hecho pocas modificaciones. Una sección antes del tratamiento de las excepciones para identificar si la causa fue una interrupción de hardware, que llevara a un método con la dirección de la rutina de tratamiento de dicha interrupción. A continuación se muestran diagramas de flujo del funcionamiento del fichero “exceptions.s”, antes y después de la modificación.

Antes de la modificación:



Después de la modificación:



**Conclusión**

Esta práctica abarca las tareas necesarias para el entendimiento del funcionamiento de las interrupciones en los procesadores. Con esto se puede llegar a programar funciones específicas para cada periférico entrada/salida. Además de familiarizarse con el uso de registros como el Timer del coprocesador.