Informe Practica de Laboratorio 4

Cesar Rojas, #31406902

Julio 2025

¿Qué diferencias hay entre gestionar E/S con sondeo y hacerlo con interrupciones?

El sondeo es una forma de gestionar E/S que involucra revisar el estado actual de la E/S constantemente, monitoreando los dispositivos hasta que se detecte una entrada o salida.

Las interrupciones funcionan de manera distinta. En lugar de que el CPU monitoree a los dispositivos periféricos, estos lo alertan mediante una *interrupción* cuando ocurra una entrada o salida.

¿Qué ventajas tiene el uso de interrupciones en términos de uso del procesador?

Las interrupciones permiten al CPU realizar otras tareas en lugar de revisar la E/S constantemente. Ya que el dispositivo 'avisa' al CPU cuando esta listo, este no tiene que usar sus recursos revisándolo. Las interrupciones también pueden reducir la latencia de respuesta y casi siempre resultan en mayores prestaciones que el sondeo.

¿Qué registros especiales se utilizan en MIPS32 para gestionar interrupciones?

Usando la herramienta *Keyboard And Display MMIO Simulator* (Simulador de Teclado y Pantalla de E/S Mapeada en Memoria), se utilizan los siguientes registros:

- Registro de Control (0xffff0000): Normalmente es 0, pero cuando se detecta una entrada por teclado se escribe 1 en este. Una vez dicha entrada se lea, se vuelve a escribir 0 hasta que se detecte otra entrada.
- Registro de Datos (0xffff0004): Contiene la ultima leida por teclado.
- Registro de Control de Salida (0xffff0008): Es 1 si la salida por pantalla esta lista para escribir.
- Registro de Transmision de Datos: (0xffff000c): Cualquier dato guardado en este registro es escrito por pantalla si el Registro de Control de Salida esta listo.

¿Por qué es necesario guardar el contexto (registros) al entrar en una rutina de servicio?

Esto es necesario para poder devolver dichos registros a su estado original. Es decir, volver a escribir en estos los valores que tenían antes de que se llamara la

rutina. De esta forma, cualquier código que ocurra después de la llamada de la rutina podra hacer uso de los valores dentro de estos registros, en lugar de que estos se pierdan cada vez que ocurra una llamada.

Momentos en que pueden generarse excepciones en un sistema MIPS32.

Enumera al menos 4 situaciones en las que se pueda generar una excepción (por ejemplo: desbordamiento aritmético, fallo de dirección, etc.).

1. Acceso de memoria erróneo: Dirección no existente, o en RAM no disponible.
2. División entre cero.
3. Error de Alineación: Usar *lw* o *sw* en una dirección no divisible por 4.
4. Interrupciones de E/S.

Explica qué etapas del pipeline pueden provocar una excepción y por qué.

- IF: Esta etapa accede a la memoria para obtener instrucciones, por lo que puede levantar excepciones de fallo de dirección o de fallo de alineación.
- ID: Esta etapa se encarga de decodificar las instrucciones, por lo que se encarga de llamar excepciones si las instrucciones tienen formato incorrecto.
- EX: Esta etapa se encarga de realizar operaciones aritméticas, por lo que las excepciones aritméticas ocurren aquí (división entre cero, sobrecargas, etc...)
- MEM: Esta etapa también accede a la memoria, pero en particular a los contenidos dentro de las direcciones. También pueden ocurrir excepciones de alineamiento y fallo de acceso de memoria.
- WB: No puede hacer excepciones. Solamente escribe en registros.

Estrategias de tratamiento de excepciones e interrupciones

Explica las diferencias entre interrupciones y excepciones (¿son síncronas o asíncronas?).

Las excepciones son *síncronas*. Es decir, se ejecutan por el CPU en un cierto tiempo, mientras que las interrupciones son *asincronas* y se llaman por eventos

externos.

Las excepciones, al ser síncronas, son predecibles y se ocurren en ciertos intervalos dictados por el CPU. Las interrupciones provienen desde afuera del CPU, por lo que son impredecibles y pueden ocurrir en cualquier momento, haciendolas *asincronas*.

0.1 Describe brevemente dos estrategias para tratar excepciones en un sistema MIPS32 ¿Cómo se redirige la ejecución hacia la rutina de servicio? ¿Cuál es la función del registro EPC (Exception Program Counter)?

Cuando ocurre una excepción, el sistema guarda la dirección de la instrucción que causo el problema en el registro EPC, y luego le pasa el control a sistema operativo para saltar a alguna dirección especifica donde se realizan las acciones necesarias para manejar la excepcion.

Luego, el programa puede terminar sus ejecucion o continuarla, en cuyo caso se usa la dirección guardada en el registro EPC para retomar la ejecución.

Para tratar las excepciones se necesita las razones por la que ocurrieron y la instruccion que la causo.

Un método consiste en usar un registro de causa para indicar la razón de la excepción (este es el método utilizado por MIPS), mientras que otro es utilizando *Vectorización*.

Esto consiste en tener varias direcciones dedicadas para distintos tipos de excepciones, para así poder manejarlas de manera mas eficiente.

Procesamiento de interrupciones

Describe paso a paso qué ocurre cuando se produce una interrupción de reloj:

Desde que el evento ocurre hasta que la rutina de servicio termina.

Una interrupción de reloj produce interrupciones cada cierto tiempo. Cuando una de estas ocurre, el CPU la detecta y guarda la dirección de retorno para volver a ella una vez se finaliza la interrupción.

El CPU guarda la causa de la interrupción, salta a la dirección especificada donde se trata la interrupción, y luego vuelve a la dirección desde donde se lanzo la interrupción.

¿Qué registros se guardan y restauran?

Todos los registros de uso general, el *$sp*, y los registros *hi* y *lo* si es necesario. Esto es para no cambiar ningun registro que se encuentre en uso por el usuario. EPC por su parte se guarda y restaura por el hardware, junto con el registro de estado.

¿Qué hace el hardware y qué hace el software (sistema operativo o rutina)?

El hardware se encarga de manejar los registros EPC, Status, y de causa. También se encarga de detectar las interrupciones de saltar a la dirección necesaria para manejar la interrupción, mientras que el software trata las interrupciones y guarda y restaura todos los otros registros necesarios.

¿Por qué es importante guardar el contexto (registros generales, EPC, Status) al entrar en la rutina?

Guardar los registros generales es necesario para evitar corromper los programas del usuario, mientras que el registro EPC y Status se guardan para mantener la integridad del sistema. Si no se guarda el EPC, el CPU podría perder su posición en el sistema y causar un crash, mientras que el registro Status contiene el estado actual del CPU, y permite resumir la ejecución de manera correcta.

Interrupciones de reloj y control de ejecución

Explica cómo una interrupción de reloj puede usarse para:

Evitar que un programa quede en un bucle infinito.

Una interrupción de reloj puede activarse cuando el bucle de un programa se ejecute por demasiado tiempo sin ningún cambio. Si el bucle se ejecuta sin finalización durante un intervalo de tiempo muy grande, se puede levantar una interrupción para interrumpir un posible bucle infinito.

Finalizar programas que superan un tiempo máximo de ejecución.

Las interrupciones de reloj ocurren después de cierto tiempo. Si el programa supera esta cantidad de tiempo de ejecución, la interrupción de reloj se levantaría y el software entonces puede detener la ejecución del programa prematuramente.

¿Qué debe hacer el software si el programa finaliza antes de que ocurra la interrupción de reloj?

El software deberá detener la interrupción de reloj pendiente, limpiar cualquier recurso de sistema utilizado y prevenir que la interrupción se active *durante* dicha limpieza.

Análisis y Discusión de los Resultados

El ejercicio de buffer circular utiliza un buffer circular pequeño de tan solo 3 elementos. Dentro de este se guardan los últimos 3 caracteres escritos usando

la herramienta *Keyboard and Display MMIO Simulator* de Mars para simular interrupciones de teclado.

Se utiliza el registro 0xffff0000 y 0xffff0004 para leer las 'interrupciones' y el carácter escrito por el usuario, y este se guarda en el buffer circular. Una vez un periodo de tiempo ocurra (en este caso, 20 segundos) se imprime por pantalla el contenido del buffer y este se vacía.

El resultado es los últimos tres caracteres escritos por el usuario mostrados por pantalla. Su orden no es necesariamente correcto ya que el buffer circular vuelve a la primera posición una vez se alcance la ultima.

El ejercicio de semáforo por su parte es mas simple. Solo se realiza una lectura (de la letra 's' minúscula en especifico) para iniciar un ciclo de interrupciones de reloj que imprimen por pantalla el estado actual del semáforo.

El resultado es que se imprime por pantalla el estado del semáforo cada cierta cantidad de segundos, hasta que este se vuelva verde otra vez. Entonces se espera de nuevo una interrupción de teclado especifica para iniciar el ciclo nuevamente.