Etc ☑ Exámenes

Exámenes

(ETC) P15 - Test de sesión (curso 20/21)

Volver a la Lista de Exámenes

Parte 1 de 1 - 12.0 Puntos

Preguntas 1 de 6

2.0 Puntos

[56] El manejador de excepciones MiMoS v.1 visto en las prácticas, incorpora la gestión del periférico RELOJ. Dicho periférico emite una interrupción (Int2) cada segundo. Se ha añadido al manejador una función de sistema '*get_time*' de número $v0 = 91, que devuelve el valor de los segundos almacenado en una variable del sistema del mismo nombre.

```
get_time:  lw $v0, segundos
           b retexc
```

La interfaz del reloj de PCSpim es la siguiente:

Dirección Base=0xFFFF0010, con un único registro de órdenes y estado. Interrumpe por la línea Int2*

Registro de órdenes y estado (Lectura/escritura)

- **R (bit 1, lectura/escritura). Indicador de dispositivo preparado: R = 1 a cada segundo.**
  - **Cancelación (R = 0): se escribirá en bit R un 0.**
- **E: (bit 0, lectura/escritura). Habilitación de la interrupción (mientras E = 1, el valor R = 1 activa la línea de interrupción del dispositivo)**

Complete el código de la rutina de servicio de la interrupción 2, de forma que se cancele la interrupción y se actualice debidamente la variable '*segundos*'.

```
int2:     la $t0, 0x ✔ffff0010        # Dir. base
          li $t1, ✔0x01
          sb $t1,   ✔0 ($t0)
          ✔lw $t1, segundos
          ✔addi $t1, $t1, 1
          sw  $t1, ✔segundos
          b retexc
```

Respuesta correcta: ffff0010|FFFF0010|0xffff0010|0xFFFF0010, 1|0x1|0x01|0x001|0x001, 0|0x0, lw, addi|ADDI|addiu|ADDIU, segundos

Preguntas 2 de 6

2.0 Puntos

[52] El manejador de excepciones MiMoS v.3 incorpora un sencillo esquema de multi-tarea que conmuta entre el proceso de usuario y un proceso ocioso. Para ello se utiliza una variable de estado declarada como:

```
# Estado del proceso principal
    LISTO = 0
    ESPERANDO_TECLADO = 2
    ESPERANDO_CONSOLA = 3
    ESPERANDO_RELOJ = 4

estado:    .word 0
```

y la sección del código dedicada al retorno del manejador es como sigue:

```
retexc:   SI (estado == LISTO)
                  $k0 = dirección retorno proceso usuario
             si no
                  $k0 = proceso_ocioso
            fin si
            restaurar registros
            rfe
            jr $k0
```

La interfaz de teclado del simulador PCSPIM contiene dos registros de 4 bytes ubicados a partir de la direción base DB = 0xFFFF0000 y tiene la posibilidad de interrumpir por la linea Int0*. La descripción de los registros es la siguiente:

Registro de Estado/Órdenes (lectura/escritura, dirección = DB)

- **R (bit 0, sólo lectura). Indicador de dispositivo preparado. R=1 cada vez que se pulsa una tecla.**
  - **Cancelación: Para hacer que R=0, es necesario leer el registro de datos.**
- **E (bit 1, lectura/escritura). Habilita la petición de interrupciones del dispositivo. Mientras E=1, se activará Int0* cada vez que R=1.**

Registro de Datos (Sólo lectura, dirección = DB + 4)

- **COD (bits 0..7). Código ASCII de la última tecla pulsada. Leer de este registro provoca la puesta a cero del bit R.**

Se desea completar este manejador para poder leer caracteres desde el teclado (*int0*). Para ello se ha desenmascarado la *int0* en la UCP, pero no en el teclado. El código de la rutina de servicio de la interrupción es como sigue:

```
int0:    lw $t0, estado
         li $t1, ESPERANDO_TECLADO
         bne $t0, $t1, retexc
         la $t0, 0xffff0000
         lb $a0, 4($t0)        # Lee el carácter del teclado
         sb $0, 0($t0)         # Deshabilita la int0
         li $t1, LISTO         # estado = LISTO
         sw $t1, estado
         b retexc
```

Complete el código de la función de sistema 'Read_char' que habilite la Int0 en el teclado, deje el proceso de usuario suspendido hasta que se pulse una tecla, de forma que el proceso de usuario reciba en $a0 el código ASCII correspondiente a la tecla pulsada.

```
Read_char:    la $t0, 0x ✔ffff0000     # Dir base teclado
              li $t1, ✔0x02
              sb $t1, ✔0 ($t0)
              li $t1, ✔ESPERANDO_TECLADO
              sw $t1, estado
              b retexc
```

Respuesta correcta: ffff0000|FFFF0000|0xffff0000|0xFFFF0000, 2|0x2|0x02|0x002|0x0002, 0|0x0|0x00, 2|ESPERANDO_TECLADO|esperando_teclado|0x2|0x02

Preguntas 3 de 6

2.0 Puntos

[51] El manejador de excepciones MiMoS v.3 incorpora un sencillo esquema de multi-tarea que conmuta entre el proceso de usuario y un proceso ocioso. Para ello se utiliza una variable de estado declarada como:

```
# Estado del proceso principal
     LISTO = 0
    ESPERANDO_TECLADO = 2
    ESPERANDO_CONSOLA = 3
    ESPERANDO_RELOJ = 4

estado:    .word 0
```

y la sección del código dedicada al retorno del manejador es como sigue:

```
retexc:   SI (estado == LISTO)
                  $k0 = dirección retorno proceso usuario
            si no
                  $k0 = proceso_ocioso
          fin si
          restaurar registros
          rfe
          jr $k0
```

La interfaz de consola del PCSPIM está en la dirección base 0xFFFF0008 y tiene los siguientes registros

Registro de órdenes y estado (Lectura/escritura. Dirección = Base)

- **R (bit 0, sólo lectura). Indicador de dispositivo preparado: R = 1 cuando la consola está disponible.**
  - **Cancelación (R = 0) : cuando se escribe en el registro de datos.**
- **E: (bit 1, lectura/escritura). Habilitación de la interrupción (mientras E = 1, el valor R = 1 activa la línea de interrupción del dispositivo)**

Registro de datos (Sólo escritura. Dirección = Base + 4)

- **COD (bits 7...0). Código ASCII de del carácter que se ha de escribir en la consola. Escribir en este registro provoca que R = 0.**

Se desea completar este manejador para poder escribir caracteres en la consola (*int1*). Para ello se ha desenmascarado la *int1* en la UCP, pero no en la consola. El código de la función de sistema '*Print_char*', a la que se le pasa en $a0 el caracter a escribir en la consola, es como sigue:

```
Print_char:    la $t0, 0xffff0008     # Dir base consola
               li $t1, 2
               sb $t1, 0($t0)         # Habilita int1 en la consola
               li $t1, ESPERANDO_CONSOLA
               sw $t1, estado
               b retexc
```

Complete el código de la rutina de servicio de la interrupción de la consola (int1) de forma que si el proceso de usuario está esperando a la consola, entonces se imprima el carácter (almacenado en $a0) y se devuelva el estado del proceso a LISTO.

```
int1: lw $t0, estado
      li $t1, ✔ESPERANDO_CONSOLA
      bne $t0, $t1, retexc
      la $t0, 0xffff0008
      sb $a0, ✔4 ($t0)
      sb $0, 0($t0)
      li  $t1, ✔listo
      sw $t1, estado
      b retexc
```

Respuesta correcta: 3|0x3|0x03|ESPERANDO_CONSOLA|esperando_consola, 4|0x4|0x04, 0|0x0|0x00|LISTO|listo

Preguntas 4 de 6

2.0 Puntos

[59] El manejador de excepciones MiMoS v.3 incorpora un sencillo esquema de multi-tarea que conmuta entre el proceso de usuario y un proceso ocioso. Para ello se utiliza una variable de estado declarada como:

```
# Estado del proceso principal
    LISTO = 0
    ESPERANDO_TECLADO = 2
    ESPERANDO_CONSOLA = 3
    ESPERANDO_RELOJ = 4

estado:    .word 0
```

y la sección del código dedicada al retorno del manejador es como sigue:

```
retexc:   SI (estado == LISTO)
```

```
                $k0 = dirección retorno proceso usuario
          si no
                $k0 = proceso_ocioso
          fin si
          restaurar registros
          rfe
          jr $k0
```

La interfaz de TECLADO del simulador PCSPIM contiene dos registros de 4 bytes ubicados a partir de la dirección base DB = 0xFFFF0000 y tiene la posibilidad de interrumpir por la línea *Int0. La descripción de los registros es la siguiente:**

Registro de Estado/Órdenes (lectura/escritura, dirección = DB)

- **R (bit 0, sólo lectura). Indicador de dispositivo preparado. R=1 cada vez que se pulsa una tecla.**
  - **Cancelación: Para hacer que R=0, es necesario leer el registro de datos.**
- **E (bit 1, lectura/escritura). Habilita la petición de interrupciones del dispositivo. Mientras E=1, se activará Int0* cada vez que R=1.**

Registro de Datos (Sólo lectura, dirección = DB + 4)

- **COD (bits 0..7). Código ASCII de la última tecla pulsada. Leer de este registro provoca la puesta a cero del bit R.**

El código de la función de sistema *'Read_char'* es el siguiente:

```
Read_char:    la $t0, 0xffff0000     # Dir base teclado
              li $t1, 0x02
              sb $t1, 0($t0)
              li $t1, ESPERANDO_TECLADO
               sw $t1, estado
               b retexc
```

Se desea completar este manejador escribiendo la rutina de servicio de la interrupción del teclado (*int0*), que debe hacer lo siguiente

```
int0:     si (estado == ESPERANDO_TECLADO)
               Cancela y deshabilita la int0
               pone en $a0 el carácter del teclado
               estado = LISTO
          fin si
          b retexc
```

Complete el código siguiente de la rutina de servicio:

```
int0:      lw $t0, estado
         li $t1, ✔ESPERANDO_TECLADO
         bne $t0, $t1, ✔retexc
          la $t0, 0x ✔ffff0000
         lb $a0, ✔4 ($t0)
         sb $0, ✔0 ($t0)
         li $t1, ✔listo
         sw $t1, estado
         b retexc
```

Respuesta correcta:
ESPERANDO_TECLADO|esperando_teclado|2|0x2|0x02|0x002|0x0002|0X2|0X02|0X002|0X0002,
retexc|RETEXC, 0xffff0000|0xFFFF0000|ffff0000|FFFF0000,
4|0x4|0x04|0x004|0x004|0X4|0X04|0X004|0X0004, 0|0x0|0x00|0x000|0x000|0X0|0X00|0X000|0X0000,
LISTO|listo|0|0x0|0x00|0x000|0x0000|0X0|0X00|0X000|0X0000
Preguntas 5 de 6

2.0 Puntos

[57] El manejador de excepciones MiMoS v.1 visto en las prácticas, incorpora la gestión del periférico RELOJ. Dicho periférico emite una interrupción (Int2) cada segundo. Se ha añadido al manejador una función de sistema '*get_time*' de número $v0=999, que devuelve el valor de los segundos almacenado en una variable del sistema llamada 'segundos'.

```
get_time:  lw $v0, segundos
             b retexc
```

La interfaz del reloj de PCSpim (modificada respecto a la de la práctica) es la siguiente:

Dirección Base=0xFFFF0510, con un único registro de órdenes y estado. Interrumpe por la línea Int2*

Registro de órdenes y estado (Lectura/escritura)

- **R (bit 0, lectura/escritura). Indicador de dispositivo preparado: R = 1 a cada segundo.**
  - **Cancelación (R = 0): se escribirá en bit R un 0.**
- **E: (bit 4, lectura/escritura). Habilitación de la interrupción (mientras E = 1, el valor R = 1 activa la línea de interrupción del dispositivo)**

Complete el código de la rutina de servicio de la interrupción 2, de forma que se cancele la interrupción y se actualice debidamente la variable '*segundos*'.

```
int2:     la $t0, 0x ✔0xffff0510        # Dir. base
           li $t1, 0x ✔10
           sb $t1,    ✔0 ($t0)
           ✔lw $t1, segundos
           ✔addi $t1, $t1, 1
           sw  $t1, ✔segundos
           b retexc
```

Respuesta correcta: ffff0510|FFFF0510|0xffff0510|0xFFFF0510, 10|010|0010|00000010, 0|0x0, lw, addi|ADDI, segundos
Preguntas 6 de 6

2.0 Puntos

[60] El manejador de excepciones MiMoS v.3 incorpora un sencillo esquema de multi-tarea que conmuta entre el proceso de usuario y un proceso ocioso. Para ello se utiliza una variable de estado declarada como:

```
# Estado del proceso principal
    LISTO = 0
    ESPERANDO_TECLADO = 2
    ESPERANDO_CONSOLA = 3
    ESPERANDO_RELOJ = 4

estado:    .word 0
```

y la sección del código dedicada al retorno del manejador es como sigue:

```
retexc:   SI (estado == LISTO)
                  $k0 = dirección retorno proceso usuario
             si no
                  $k0 = proceso_ocioso
            fin si
            restaurar registros
            rfe
            jr $k0
```

La interfaz de consola del simulador PCSPIM contiene dos registros de 4 bytes ubicados a partir de la direción base DB = 0xFFFF0008 y tiene la posibilidad de interrumpir por la linea Int1*. La descripción de los registros es la siguiente:

Registro de Estado/Órdenes (lectura/escritura, dirección = DB)

- **R (bit 0, sólo lectura). Indicador de dispositivo preparado. R=1 cada vez que se escribe un carácter.**
  - **Cancelación: Para hacer que R=0, es necesario leer el registro de datos.**
- **E (bit 1, lectura/escritura). Habilita la petición de interrupciones del dispositivo. Mientras E=1, se activará Int1* cada vez que R=1.**

Registro de Datos (Sólo lectura, dirección = DB + 4)

- **COD (bits 0..7). Código ASCII del carácter a escribir. Leer de este registro provoca la puesta a cero del bit R.**

Se desea completar este manejador para poder escribir caracteres en la consola(*int1). Para ello se ha desenmascarado la *int1* en la UCP, pero no en la consola. El código de la rutina de servicio de la interrupción es como sigue:**

```
int1:    lw $t0, estado
         li $t1, ESPERANDO_CONSOLA
         bne $t0, $t1, retexc
         la $t0, 0xffff0008
         sb $a0, 4($t0)        # Escribe el carácter en consola
         sb $0, 0($t0)         # Deshabilita la int1
         li $t1, LISTO         # estado = LISTO
         sw $t1, estado
         b retexc
```

Complete el código de la función de sistema 'Print_char' que habilite la Int1* en el teclado, deje el proceso de usuario suspendido hasta que se imprima el carácter almacenado en el registro $a0 en consola.

```
Print_char:   la $t0, 0x ✔ffff0008

              li $t1, ✔0x02

              sb $t1, ✔0 ($t0)

              li $t1, ✔ESPERANDO_CONSOLA

              sw $t1, estado

              b retexc
```

Respuesta correcta: ffff0008|FFFF0008|0xffff0008|0xFFFF0008, 2|0x2|0x02|0x002|0x0002, 0|0x0|0x00, 2|ESPERANDO_CONSOLA|esperando_consola|0x2|0x02