Paula Domínguez Martínez

Práctica 4 EC

Apartado 1: Desarrollo del módulo de gestión de la hora y fecha

hora_Init (hora, fecha) {

Guardar la hora y la fecha en memoria
Habilitar el System Timer con periodo 1 segundo y dirección hora_Incr
}

Dirección hora	La dirección de memoria de la hora para guardarla	R2
Dirección fecha	La dirección de memoria de la fecha para guardarla	R2
Hora	La hora que se pasa por parámetro	R0
Fecha	La fecha que se pasa por parámetro	R1

hora_Incr () {

```
Añado un segundo a los segundos
Si (segundos == 60) {
        Añado un minuto a los minutos y pongo segundos a 0
        Si (minutos == 60) {
                Añado una hora a las horas y pongo minutos y segundos a 0
                Si (horas == 24) {
                        Añado un día a los días y pongo horas, minutos y segundos a 0
                        Si (días == 30) {
                                Añado un mes a los meses y pongo días a 1
                                Si (meses == 12) {
                                        Añado un año a años y pongo meses y días a 1
                                }
                        }
                }
        }
}
Convierto cada digito a ascii y muestro poniendo ":" entre horas, minutos y segundos y
"/" entre días, meses y años
```

}

Dirección hora	Dirección de memoria de la hora donde se guarda	R0, R4
Hora	La hora actual	R1
Variables de segundos, minutos, horas, días, meses, años	Los valores actuales con los que se opera	R2, R5, R0
Valores auxiliares	Valores con los que se hacen operaciones como BIC	R2
Dirección fecha	Dirección de memoria de la fecha donde se guarda	R3, R4
Fecha	La fecha actual	R4
Línea	Línea donde se escribe, paso de parámetro a Set_Cursor	R0
Posición	Posición donde se escribe, paso de parámetro a Set_Cursor	R1
Unidad de segundos, minutos, horas…	Segundo dígito de cada valor, segundos, minutos…	R5
Caracteres ":" y "/"	Separadores entre segundos, minutos… (dirección y valor)	R0
Variables auxiliares para la descomposición del año	Decenas, centenas del año	R6, R7

hora_Off () {

Deshabilita el System Timer
}

T_hora {

Pido fecha y hora
hora_Init ()
Pérdida de tiempo de un segundo
hora_Incr ()
Pérdida de tiempo de un segundo
hora_Incr ()
Pérdida de tiempo de un segundo
hora_Incr ()
}

Dirección hora	Dirección de memoria de la hora para guardarla	R4
Dirección mensajes	Dirección de memoria de los mensajes que piden	R0
Valor de hora, minutos, segundos, días, meses, años	Valores pedidos que entran por read_integer	R0
Dirección fecha	Dirección de memoria de la fecha para guardarla	R4
Hora	Valor de la hora que se pasa como parámetro	R0
Fecha	Valor de la fecha que se pasa como parámetro	R1
Segundo	Valor de un segundo en microsegundos para el delay	R0

Apartado 2: Atención a las interrupciones

ST_Intr (periodo, dirección) {

Guardar la dirección de la función en memoria
Guardar el periodo T en memoria
Poner ST_RTI como RTI del comparador 3
Quitar las posibles peticiones de interrupciones
Habilitar las interrupciones para el comparador 3
Habilitar el comparador 3 con tiempo de tiempo actual + T

}

Dirección tiempo	Dirección de memoria del tiempo que indica el periodo	R0
Tiempo	Periodo de la interrupción	R0 (parámetro), R4
Dirección función	Dirección de memoria de la función que se ejecuta	R0
Función	Función que se ejecuta periódicamente	R1
Dirección de las direcciones lógicas del ST y del CI	Dirección de memoria de donde se guardan las direcciones lógicas	R0
Direcciones lógicas del ST y del CI	Dirección donde se encuentran el ST y CI	R3, R0
Dirección de la rutina de atención	Dirección de memoria de ST_RTI para ponerla como rutina de atención	R1

ST_RTI {

Salvado de contexto
Eliminar la interrupción
Salto a tarea de periodo T
Habilitar el comparador 3 con tiempo de tiempo actual + T
Restaurar contexto

}

Dirección lógica del ST	Dirección donde se encuentra el ST	R0
Valores auxiliares	Valor que desactiva la petición de interrupción y valor que habilita el comparador de nuevo	R1
Tiempo	Valor del periodo de la interrupción	R2
Dirección de vuelta de la función	Dirección que permite volver después de ejecutar la función periódica	LR
Función	Dirección de la función que se va a ejecutar de forma periódica	R0, PC

ST_Off () {

    Eliminar ST_RTI como RTI del comparador 3
    Deshabilitar las interrupciones del comparador 3
}

Apartado 3: Programa de prueba

```
Prueba3 {
        Pido fecha y hora
        hora_Init ()
        do {
                letra = abecedario [valor]
                Muestro letra
                Pérdida de tiempo de medio segundo
                valor++
        } while (valor != 26)
        hora_Off ()
}
```

Dirección mensajes	Dirección de memoria de los mensajes que piden	R0
Dirección hora	Dirección de memoria de la hora para guardarla	R4
Hora	Valor de la hora actual, paso de parámetro a hora_Init	R0
Dirección fecha	Dirección de memoria de la fecha para guardarla	R4
Fecha	Valor de la fecha, paso de parámetro a hora_Init	R1
Valor de segundos, minutos, horas…	Valores que retorna read_integer al pedir	R0
Dirección abecedario	Dirección del array que guarda el abecedario	R4
Letra	Carácter de la letra del abecedario	R5, R0 (parámetro)
Posición letra	Posición de la letra en el abecedario (desde 0 por array)	R6
Delay	Valor de delay en espera, medio segundo	R7

Preguntas sobre la práctica

1. **Indica el formato en que has almacenado la fecha y la hora y los motivos por los que has escogido ese formato.**

He escogido un formato en el que la fecha es un .word y la hora es otro .word, en mi caso, se almacena en la variable hora, con el formato xxhhmmss, de tal forma que cada campo es un byte y se coge con un strb, he utilizado este formato ya que es mucho más simple comparar estos valores como números que en un string, y así los que más veces van a cambiar están en la primera posición.

En la fecha he utilizado un formato parecido, aaaammdd, ya que el campo al que más veces se va a acceder son los días y está en la primera posición. Para el año solamente se necesitan dos bytes para almacenarlo, por lo que en el .word entraría, para acceder al año, en lugar de strb se utilizaría un strh con un inmediato #3, pero funciona de la misma forma. Esta manera de almacenarlo la he escogido por ser más cómoda al usar números ya que para transformarlos a string se pueden dividir y también al depurar es más fácil saber que hay en cada momento, en lugar de tener que convertirlo a número desde ascii.

Por estas razones creo que mi elección es la más útil a pesar de tener la desventaja de tener que dividir al convertirlo a caracteres para mostrarlo en el LCD.

2. **Indica qué está sucediendo en la Raspberry mientras se ejecuta el programa de prueba T_Intr1.**

En la Raspberry se está ejecutando el programa principal que llama a la función que programa el timer, que activa el comparador 3 del System Timer, habilita las interrupciones de este controlador, pone la rutina de atención a la interrupción programada como rutina de atención para el comparador 3 y pone un tiempo en el comparador para que se produzca la interrupción.

Se pasa de nuevo al programa principal que muestra una letra, mientras, se va actualizando el contador del System Timer, en el momento en el que se llega al valor indicado se produce una interrupción, cuando se atiende la interrupción se ejecuta la rutina de atención a la interrupción realizada que quita la interrupción producida, produce un salto a la rutina que se pasa por parámetro en la rutina de inicialización anterior y pone un tiempo en el comparador de nuevo.

Cuando se vuelve al programa principal, realiza de nuevo el proceso anterior hasta que le llegue una G, que es la última letra que quiere mostrar donde se desactiva el comparador 3, se deshabilitan las interrupciones de este comparador y se quita la rutina de atención realizada para las interrupciones de este comparador.

Durante todo este proceso, el contador del timer se va actualizando, es decir, según el programa se ejecuta, se va contando hasta que se llega al tiempo indicado que es cuando se pide la interrupción, con esto se pueden realizar varias tareas, si una se quiere cada un cierto tiempo.

3. **Indica que cambios deberían hacerse en las rutinas desarrolladas del System Timer si se desea ejecutar dos tareas, una con un periodo de T segundos (T como parámetro) y la otra con un periodo de 10T segundos.**

Para que se puedan ejecutar dos rutinas se podría introducir una variable del driver que contara cuantas veces se ha ejecutado la tarea de periodo T, de tal forma que, cuando llevara 9 ejecuciones, en la siguiente se ejecutaran las dos.

Los pseudocódigos de estas funciones con este cambio serían los siguientes.

```
ST_Intr (periodo, direccion1, direccion2) {
        Guardar las direcciones de las funciones de las tareas en memoria
        Guardar el periodo T en memoria
        Crear una variable I = 0 en memoria
        Poner ST_RTI como RTI del comparador 3
        Quitar las posibles peticiones de interrupciones
        Habilitar las interrupciones para el comparador 3
        Habilitar el comparador 3 con tiempo de tiempo actual + T
}
```

```
ST_RTI {
        Salvado de contexto
        Eliminar la interrupción
        Si (I != 9) {
                I++
        }
        Sino {
                I = 0
                Salto a tarea de periodo 10 T
        }
        Salto a tarea de periodo T
        Habilitar el comparador 3 con tiempo de tiempo actual + T
        Restaurar contexto
}
```

La función ST_Off no necesitaría cambios.