HOJA DE EJERCICIOS A ENTREGAR AL FINALIZAR LA SESIÓN

PRACTICA E/S POR POLLING EN LINUX

Apellidos	Nombre	Fecha (dd/mm/aaaa)	Hora	Grupo (G1, G2, G3)
Pérez Álvarez	José Manuel	18/5/2020	19:40	G1

Los siguientes apartados se realizan sobre la plantilla de la práctica (fichero de C), de forma que **deberá aportar al final el código completo** que contemple el contenido de todos los apartados.

Pasos a seguir:

1. Incorpore en el programa principal (*main*):
   a) La **llamada a la función** (llamada al sistema) **de desbloqueo de puestos de E/S** con su parámetro correspondiente (desbloquear el acceso a bajo nivel (2) mediante la llamada al sistema **iopl()**, ref: http://man7.org/linux/man-pages/man2/iopl.2.html).
   b) Posteriormente, compruebe que funciona correctamente realizando una lectura simple de la hora del RTC. A la hora de mostrar un valor proporcionado por el RTC utilizando la función printf**, el tipo de dato que debemos indicar es hexadecimal.** Utilice el identificador **"%02X"**. Este indicador en una función printf en C indica con la "X" que el valor a imprimir está en hexadecimal (sirve también para BCD), y que como mínimo tiene 0 dígitos y como máximo 2.
   c) Defina los números de puerto de E/S que vaya a utilizar en su programa, **como constantes** antes de main (por ejemplo: #define PUERTO_DD 0x68).

2. **Implemente una función "lee"** que reciba por parámetros un valor *unsigned char*, que representará el registro del RTC que queremos leer, y retorne otro *unsigned char*, que representará el valor leído. Realice un programa de ejemplo en el *main* para probar que funcione correctamente para cualquier registro dado. Por ejemplo, **puede adaptar el código del apartado 1 de lectura** de la hora para parametrizarlo, modularizarlo y meterlo así en una función reutilizable para leer el resto de los valores del RTC.

3. Una espera activa es un proceso que **de forma continua verifica si se cumple una condición.** Implemente dos funciones que permitan esperar un ciclo completo de actualización de la siguiente forma:
   a) Función "espera_hasta_actualizacion": realiza una espera activa **mientras que el bit 7 del registro de estado esté a cero**. *Se sugiere hacer uso de máscaras.*
   b) Función "espera_hasta_despues_actualizacion": realiza una espera activa **mientras que el bit 7 del registro de estado esté a uno.** *Se sugiere hacer uso de máscaras.*
   c) **Una vez finalizadas las llamadas** a estas dos funciones **consecutivamente** en *main,* **se podrá acceder a los datos del RTC** de forma segura.
   ***El RTC en la llamada a la función del paso a) estaba **en el proceso o camino de llegar hasta la actualización**. Al finalizar la llamada de la función del paso b) **el RTC ya se ha actualizado** y tiene por delante todo un ciclo hasta la siguiente actualización (típicamente 1 segundo), lo cual proporciona tiempo de sobra para leerlo con seguridad.*

4. Utilizando las funciones anteriores complete el programa *main* para **mostrar por pantalla la hora, minutos, segundos, día de la semana, mes y año** (proporcionados

todos por el RTC) con en el siguiente formato: "Dia_semana, DD/MM/AAAA – HH:MM:SS". Para mostrar el día de la semana ("Domingo", "Lunes", etc) **ayúdese del vector de cadenas "dia_semana"** que aparece **en la plantilla** del programa. **El RTC devuelve la posición** en dicho vector.
Realice un **bucle continuo para ir mostrando estos datos a medida que van siendo actualizados** (equivalente a crear un cronómetro).
Nota:* utilice **"\r" en el printf (no "\n") para que tras cada muestra por pantalla se actualice en la misma línea. **Añada después** del printf la instrucción **"fflush(stdout);"**. Investigue por qué es necesaria dicha función y por qué tiene el parámetro "stdout".*

5. Complete el código anterior con **una nueva función llamada "escribe"**. Esta función escribe un dato en un puerto (dato) del RTC. Utilice esta función para **modificar la hora del RTC**. Para ello debe **pedir al usuario** (*) **que introduzca la hora del sistema por teclado** (sólo la hora, no hacen falta los minutos y/o los segundos). Recuerde que los datos que le proporcionamos al RTC han de encontrarse en BCD empaquetado (**). Tras ello, **muestre la hora tal y como se hacía en el apartado 4** para comprobar que se ha modificado correctamente.
*(*) Para pedir al usuario un dato, repase/investigue el funcionamiento de la función **"scanf"**.*
*(**) Tendrá que hacer **una conversión del número decimal a BCD**, separando los dígitos (que máximo serán dos) y adecuándolos al formato.*

IMPLEMENTACIÓN FINAL:

```
//Inclusiones de librerías
#include <stdio.h>
#include <sys/io.h>

// Definición de las constantes utilizadas (Nº puertos)
#define Puerto_SS 0x00
#define Puerto_MM 0x02
#define Puerto_HH 0x04
#define Puerto_SEM 0x06
#define Puerto_MES 0x08
#define Puerto_DD 0x07
#define Puerto_YY1 0x09
#define Puerto_YY2 0x32

//Prototipo de la función de lectura de un valor del RTC  (paso 2)
// TODO
unsigned char lee (unsigned char);

//Prototipo de la función de escritura de un valor del RTC  (paso 5)
// TODO
void escribe (unsigned char, unsigned char);

//Prototipo de la función de traducción de unsigned char a BCD empaquetado
//(paso5)
// TODO
unsigned char convertidorBCD (unsigned int);

//Declaración de variable global (facilita la representación del día de la
//semana)
char* dia_semana[8] =
       {"Domingo","Domingo","Lunes","Martes","Miercoles","Jueves","Viernes",
       "Sabado"};

//Función principal
```

```
int main()
{
	unsigned char semana, dia, mes, anyo1, anyo2, segundos, minutos, horas,
	horaModificada, horaDelSistema;
	unsigned int horaPedida;

	//1. Desprotección los puertos de E/S  (paso 1)
	// TO DO

	iopl(3);

	//2. Modificación de la hora: se pide la hora y se actualiza el RTC (paso
	5)
	// TO DO

	outb(Puerto_YY2, 0x70);
	anyo2 = inb(0x71);

	outb(Puerto_YY1, 0x70);
	anyo1 = inb(0x71);

	outb(Puerto_DD, 0x70);
	dia = inb(0x71);


	outb(Puerto_MES, 0x70);
	mes = inb(0x71);

	outb(Puerto_SEM, 0x70);
	semana = inb(0x71);

	outb(Puerto_HH, 0x70);
	horas = inb(0x71);

	outb(Puerto_MM, 0x70);
	minutos = inb(0x71);

	outb(Puerto_SS, 0x70);
	segundos = inb(0x71);


	//3. Bucle continuo{ (paso 4)
	// TO DO

	while(1){


	//3.1. Espera activa mientras el bit 7 del registro 0x0A vale 0 (paso 3)
	// TO DO
	outb(0x0A, 0x70);
	while(inb(0x71) & 0x80==0);


	//3.2. Espera activa mientras el bit 7 del registro 0x0A vale 1 (paso 3)
	// TO DO
	while(inb(0x71) & 0x80!=0);


	//3.3. Lectura de cada dato del RTC y almacenamiento en variables (paso
	4)
	// TO DO
```

```
        semana = lee(Puerto_SEM);
        dia = lee(Puerto_DD);
        mes = lee(Puerto_MES);
        anyo1 = lee(Puerto_YY1);
        anyo2 = lee(Puerto_YY2);
        segundos = lee(Puerto_SS);
        minutos = lee(Puerto_MM);



        //3.4. Se muestra por pantalla los valores leídos (paso 4) -- TO MODIFY
        printf("\r%s,  %02X/%02X/%02X %02X:%02X:%02X", dia_semana[semana], dia,
        mes, anyo1, anyo2, horas, minutos, segundos);
        fflush(stdout);
        //Falta el formato de muestreo
        scanf("Introduzca que hora es: %d",&horaPedida);
        horaModificada=convertidorBCD(horaPedida);
        escribe(horaModificada,Puerto_DD);

        outb(Puerto_DD, 0x70);
        horaDelSistema = inb(0x71);
        printf("\rHora del Sistema: %02X", horaDelSistema);
        }
        //}Fin Bucle continuo (paso 4)
        // TODO

        //Fin del programa
        return 0;
}




//Implementación de la función de lectura de un valor del RTC (paso 2)
// TODO
unsigned char lee (unsigned char param){
        outb(param, 0x70);
        return inb(0x71);
}

//Implementación de la función de escritura de un valor del RTC  (paso 5)
// TODO
void escribe (unsigned char hora, unsigned char param){
        outb(param, 0x70);
        outb(hora, 0x71);
}

//Implementación de la función de traducción de unsigned char a BCD empaquetado
//(paso 5)
// TODO
unsigned char convertidorBCD (unsigned int num){
        return (((num & 0xF0) >> 4) * 10) + (num & 0x0F);
}
```