UNIVERSIDAD CARLOS III DE MADRID (Dpto. de Tecnología Electrónica)

Sist. Dig. Basados en Microprocesador **(Gr. Ing. Telemática)**

23 de mayo de 2012 EXAMEN FINAL (3,5 horas)

No se permiten ni libros, ni apuntes, ni calculadoras programables. Sólo se permite el manual del microcontrolador Caja ejercicio se contestará en hojas independientes. Se pueden utilizar tantas hojas por ejercicio como considere oportuno (salvo el ejercicio 1 que se contestará aquí). Las respuestas han de entregarse escritas en bolígrafo o pluma. Todas las respuestas deben estar jutificadas

APELLIDOS	NOMBRE	NIA

EJERCICIO 1 (2 puntos, 30 minutos):

Diseñe un microprocesador de arquitectura Von Neumann, con una memoria de 8 bits x 64K, indicando:

a) Tamaño del bus de datos:____bits y del bus de direcciones:_____ bits
b) Tamaño del PC:_____bits y del MAR:_____bits
c) Si necesitamos gestionar 16 instrucciones, ¿Cuál es el tamaño del opcode? _____ bits
d) Si contásemos con 4 registros de propósito general que podrían ser utilizados indistintamente como fuente y como destino, y partiendo de la información del apartado c) ¿Cuál sería el tamaño mínimo del IR? _____bits
e) Si el Registro de Instrucción tuviera una capacidad múltiplo de 8 bits, y nos pidiesen que implementásemos la siguiente instrucción:

MOV Reg, <dirección>
Reg ← (dirección)

a. ¿Qué espacio mínimo ocuparía esta instrucción en memoria? _____ bits / _____ palabras
b. ¿Podría ejecutarse en este micro? ☐Sí ☐No.
c. Si fuera posible, indique la secuencia de pasos en sentencia RTL al mayor nivel de detalle.

f) Describa la secuencia de pasos en RTL con el mayor nivel de detalle para ejecutar la siguiente instrucción:

ADD Reg, <dirección_8bits>
Reg ← Reg + <dirección_8bits>

g) ¿Qué limitación clara tendría la instrucción anterior? (responda en 1 línea)

_____

EJERCICIO 2 (4 puntos, 90 minutos):

Basándose en el RTC, en dos GPIO y en el display, diseñe un reloj que muestre: HH:MM, con las siguientes funcionalidades:

1. Dos modos de funcionamiento:
   - *Normal:* El reloj mostrará la hora en el formato indicado
   - *Programación:* El reloj permitirá que se modifique horas y minutos.
2. El cambio de modo "programación" a "modo normal" y viceversa, se producirá pulsando los dos botones a la vez.
3. Uno de los botones servirá para moverse de forma circular entre horas y minutos cuando el reloj se encuentre en modo programación. Mientras permanezca en este modo, la hora no cambiará de forma automática.
4. El otro botón servirá para incrementar el parámetro seleccionado (horas o minutos) en modo circular (horas de 0 a 23 y minutos de 0 a 59).

SE PIDE:

a) Desarrolle la función de configuración y arranque del RTC: "`void config_RTC(int hora, int minuto)`" (hora: 0 - 23 , minuto: 0-59).

Nota: *LSE = 32.768 Hz*

b) Escriba la función "`void hora_to_str(unsigned char *cadena)`", que pase la hora del registro de tiempo del RTC a un string en formato "HH:MM".

c) Diagrama de flujo de la función "`int get_botones(void)`" sabiendo que:
   - Devuelve 0 si no se ha pulsado ningún botón.
   - Devuelve 1, 2 o 3 si se ha pulsado y soltado el botón 1, el botón 2, o ambos botones a la par.

Nota: *Tenga en cuenta que pulsar 2 botones "exactamente a la vez" es casi imposible.*

d) Escriba la función "`int get_botones(void)`" con la siguiente condición de diseño:
   - Utilice los pines PA11 y PA12, sabiendo que si el botón está pulsado entrega 0 voltios y si no lo está entrega 3 voltios.

EJERCICIO 3 (4 puntos, 90 minutos):

En las próximas hojas aparece el código de un programa a cargar en un sistema basado en un STM32L152RB, **al cual le puede faltar alguna función y/o alguna definición y/o contener algún error de programación**. El programa corresponde a la implementación de un velocímetro de bicicleta, capaz de mostrar por la pantalla la información sobre la distancia recorrida, la velocidad instantánea y la temperatura. De dicho sistema se conocen las siguientes características:

- Tiene dispositivos conectados cuya señal de entrada al STM32L152RB, es un valor cualquiera entre 0 y 3,3V, con una variación de velocidad menor de 1Hz.
- Puede tener algún otro dispositivo más conectado.

Se pretende que el alumno analice dicho código y, a partir de ahí conteste **razonadamente** a las siguientes preguntas (*algunas de las justificaciones se pueden realizar indicando las líneas de código donde se encuentra la evidencia*):

1. ¿Qué elementos (periféricos) del STM32L152RB se están utilizando? (10%)
2. Teniendo en cuenta que tiene una configuración de reloj que hace que el pclk de todos los periféricos vaya a 12MHz, indique la configuración de cada uno de los periféricos del microcontrolador cuando se encuentren en funcionamiento. Absténgase de simplemente decir el valor de cada registro de configuración; lo que se pide es la funcionalidad que se obtiene. Es imprescindible detallar la escala temporal (caso de que exista) que utilizan los periféricos. (10%)
3. De un significado breve a las siguientes variables del programa. No tienen por qué ser acrónimos de ningún tipo (20%):
    a. cero
    b. uno
    c. dos
    d. tres
    e. cuatro
    f. cinco
    g. seis
4. El código presenta al menos tres errores. Dos de esos errores está en los nombres de las funciones, mientras que el otro se encuentra en el funcionamiento del mismo. Encuéntrelos, justifíquelos y dé una solución a los mismos. (20%)
5. Analizando la pureza del código, un experto expone una crítica a la línea 35. ¿Podría decir qué crítica es, y cómo la solucionaría? (10%)
6. Realice el Diagrama de flujo de todas y cada una de las funciones utilizadas, así como del programa principal. No haga siempre una transposición directa del código en un diagrama de flujo, sino represente la funcionalidad obtenida, mediante dicho diagrama de flujo. (30%)

No se permiten ni libros, ni apuntes, ni calculadoras programables. Sólo se permite el manual del microcontrolador Caja ejercicio se contestará en hojas independientes. Se pueden utilizar tantas hojas por ejercicio como considere oportuno (salvo el ejercicio 1 que se contestará aquí). Las respuestas han de entregarse escritas en bolígrafo o pluma. Todas las respuestas deben estar jutificadas

ANEXO I

```
1   #include "stm32l1xx.h"
2   #include "Biblioteca_SDM.h"
3   unsigned char radio=23;
4   unsigned uno, dos;
5   unsigned char cero;
6   unsigned tres;
7   unsigned cuatro;
8   unsigned cinco;
9   unsigned seis;
10
11  void RAI1 (void) {
12     EXTI->PR = 0x01;
13     NVIC->ICER[0] |= (1 << 6);
14     if ((EXTI->RTSR & 0x01) == 0) {
15        TIM4->CCR2 = TIM4->CNT + 2000;
16        TIM4->CR1 |= 0x0001;
17        EXTI->RTSR |= 0x01;
18        EXTI->FTSR &= ~(0x01);
19        }
20     else {
21        TIM4->CR1 &= ~(0x0001);
22        EXTI->FTSR |= 0x01;
23        EXTI->RTSR &= ~(0x01);
24        }
25     NVIC->ISER[0] |= (1 << 6);
26     }
27
28  void RAI2 (void) {
29     if (TIM4->SR & 0x0004 == 1) {
30        cero = 1;
31        TIM4->CR1 &= ~(0x0001);
32        TIM4->SR = 0x0004;
33        }
34     else {
35        seis = 2 * 3142 * radio;
36        tres += seis;
37        dos = TIM4->CCR1 - uno;
38        if (dos < 0) dos += 0xFFFFFFFF;
39        cinco = seis / dos;
40        uno = TIM4->CCR1;
41        TIM4->SR = 0x0002;
42        }
43     }
44
45  int main (void) {
46     Init_SDM();
47     Init_LCD();
48     cero = 0;
```

```
49      GPIOA->MODER |= 0x00000300;
50      GPIOA->MODER &= ~(1 << (0*2 +1));
51      GPIOA->MODER &= ~(1 << (0*2));
52      GPIOA->PUPDR &= ~(11 << (0*2));
53      GPIOB->MODER |= 0x00000001 << (2*6 +1);
54      GPIOB->MODER &= ~(0x00000001 << (2*6));
55      GPIOB->AFR[0] &= ~(0x0F << (4*6));
56      GPIOB->AFR[0] |= 0x02 << (4*6);
57      ADC1->CR2 &= ~(0x00000001);
58      ADC1->CR1 = 0x02000000;
59      ADC1->CR2 = 0x00000472;
60      ADC1->SMPR1 = 0;
61      ADC1->SMPR2 = 0;
62      ADC1->SMPR3 = 0;
63      ADC1->SQR1 = 0x00000000;
64      ADC1->SQR5 = 0x00000004;
65      ADC1->CR2 |= 0x00000001;
66      while ((ADC1->SR&0x0040)==0);
67      ADC1->CR2 |= 0x40000000;
68      EXTI->FTSR |= 0x01;
69      EXTI->RTSR &= ~(0x01);
70      SYSCFG->EXTICR[0] = 0;
71      EXTI->IMR |= 0x01;
72      NVIC->ISER[0] |= (1 << 6);
73      TIM4->CR1 = 0x0000;
74      TIM4->CR2 = 0x0000;
75      TIM4->SMCR = 0x0000;
76      TIM4->PSC = 12000;
77      TIM4->CNT = 0;
78      TIM4->ARR = 0xFFFF;
79      TIM4->CCR2 = 0;
80      TIM4->DCR = 0;
81      TIM4->DIER = 0x0006;
82      TIM4->CCMR1 = 0x0001;
83      TIM4->CCMR2 = 0x0000;
84      TIM4->CCER = 0x0003;
85      TIM4->CR1 |= 0x0001;
86      TIM4->EGR |= 0x0001;
87      TIM4->SR = 0;
88      NVIC->ISER[0] |= (1 << 30);
89      uno = 0;
90      while (1) {
91         if (cero!=0) {
92            uno = 0;
93            tres = 0;
94            }
95         while ((ADC1->SR & 0x0002)==0);
96         cuatro = (unsigned char)(ADC1->DR & 0x000000FF);
97         MuestraVelocimetro(cinco, tres, cuatro);
98         }
99      }
```