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

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

17 de marzo de 2016 EXAMEN PARCIAL (100 minutos)

No se permiten ni libros, ni apuntes, ni calculadoras programables. Sólo se permite el manual del microcontrolador

Todas las respuestas deben estar justificadas.

CUESTIÓN (2 puntos):

Escriba las instrucciones en C necesarias para realizar las siguientes operaciones, sin alterar el comportamiento del resto del sistema:

a) Configurar el pin PB12 como entrada digital
b) Desactivar en el NVIC las interrupciones por EXTI4 (canal 10)

SOLUCIÓN:

a) Hay que escribir en el GPIOB->MODER, en los bits correspondientes al pin 12, un 0 en el bit más significativo y un 0 en el menos significativo. Utilizando máscaras para no afectar al resto de los pines:

```
GPIOB->MODER &= ~(3 << (12*2));
```

b) Hay que escribir en el ICER un 1 en la posición del canal 10. Al ser el canal 10, está en el ICER[0]. Como los 0s no afectan, no hace falta utilizar máscara:

```
NVIC->ICER[0] = (1<<10);
```

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

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

17 de marzo de 2016 EXAMEN PARCIAL (100 minutos)

No se permiten ni libros, ni apuntes, ni calculadoras programables. Sólo se permite el manual del microcontrolador

Todas las respuestas deben estar justificadas.

PROBLEMA 1 (5 puntos):

Basándose en un STM32L152RB con pclk de 32MHz, se ha de diseñar un reloj con control de la frecuencia cardíaca, que cumpla con los siguientes requisitos:

- El reloj debe mostrar (salvo alarmas) siempre la hora y los minutos en el LCD.
- El microcontrolador tiene conectado a una entrada un sensor de pulso, que cada vez que detecta un latido, emite un pulso positivo (pasa de 0 a 1 durante un tiempo y luego baja a 0), de una duración inferior al milisegundo.
- Si el pulso detectado es mayor de 140 latidos por minuto, se debe generar una alarma en forma de una señal periódica de 1KHz que atacará a un altavoz. En cuanto el pulso baja de 140, la alarma debe parar.
    - Esa alarma debe mostrar en el LCD el mensaje "PARE"
- Si el pulso detectado es inferior a 40 latidos por minuto, se debe generar una alarma, esta vez de 500Hz, que atacará al mismo altavoz. En cuanto el pulso sube de 40 latidos, la alarma debe parar.
    - Esa alarma debe mostrar en el LCD el mensaje "MEDICO"

Con estas especificaciones (de forma razonada e intentando reducir el número de recursos utilizados) proporcione:

a) Diagrama de Bloques del sistema. (20%)
b) Indique si utilizará interrupciones, y en caso afirmativo, cuáles y para qué. (20%)
c) Indique qué periféricos utiliza y cómo los configuraría. (30%)
d) Diagrama de flujo. (30%)

SOLUCIÓN:

1) Atendiendo al enunciado, se necesita conectar el microcontrolador al LCD, al sensor y al altavoz. Además habrá que tener un temporizador que lleve la hora del sistema. La entrada del sensor, al tener que medir tiempos entre pulsos (entre flancos de subida, por ejemplo), deberá ir a una entrada TIC de un temporizador. La salida hacia el altavoz, al tener que generar tonos de frecuencias distintas, deberá ir a una salida TOC de un temporizador.
Por tanto es necesario utilizar 3 servicios de temporización. Se podrían utilizar tres temporizadores, o hacer un análisis a ver si se podría utilizar el mismo temporizador para todo. Para ello, hay que mirar qué rangos de tiempos hay que medir.

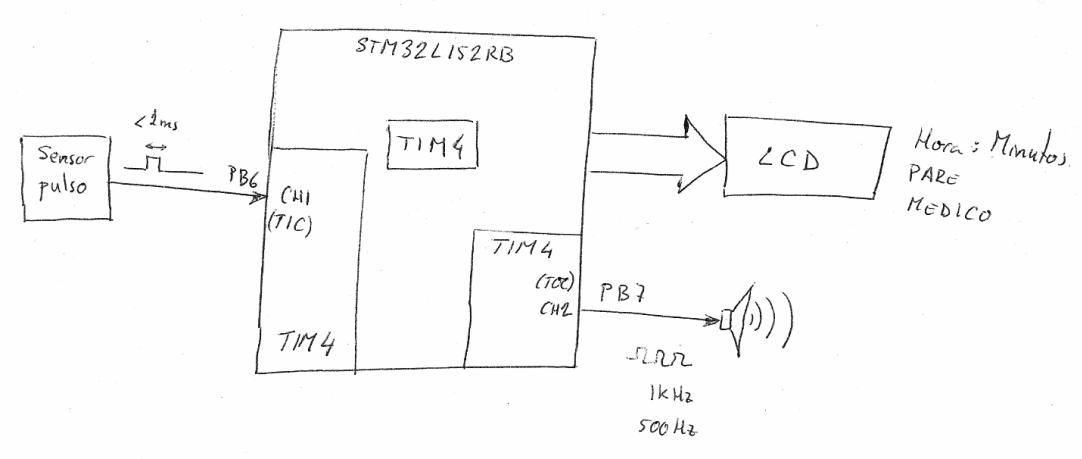
- *Para la hora del sistema, se tendrían que medir segundos, aunque también se podrían medir tiempos menores y acumular dichos tiempos.*
- *Para el TIC, teniendo en cuenta que la alarma saltaría cuando bajase de 40 latidos por minuto (1,5 segundos entre pulsos), o subiese de 140 latidos por minuto (0,43 segundos entre latidos), se puede plantear medir centésimas de segundo.*

- *Para el TOC, teniendo en cuenta que los tonos son de 1KHz (0,5ms entre flancos) y 500Hz (1ms entre flancos), los intervalos de tiempo a gestionar deberían ser como máximo de 0,1ms.*
    - *Si se tomase que los saltos del CNT fuesen de 0,1ms, entonces, utilizando el mismo temporizador para los otros servicios:*
        - *El TIC debería contar por encima de 15000 (1,5 segundos)*
        - *El reloj debería contar cada 10000*

Por lo tanto todo está en el mismo rango de medida, así que se puede utilizar un mismo temporizador con las siguientes características:

- *Que mida intervalos de 0,05ms:*
- *TOC:*
    - *Con salida hardware que cambie en modo toogle*
    - *Con interrupción que salte en la comparación para actualizar el valor del CCMR*
        - *Para la señal de 1KHz: CCRx += 10*
        - *Para la señal de 500Hz: CCRx += 20*
    - *Que se active y desactive la funcionalidad de TOC (o la salida externa) cada vez que tenga que sonar una alarma.*
- *TIC:*
    - *Con entrada sensible a uno de los flancos (por ejemplo, el de subida)*
    - *Activa por interrupción.*
    - *Que reste el valor capturado respecto al anterior, para obtener la frecuencia cardíaca*
        - *Para pulsaciones inferiores a 40 latidos por minuto, el número de cuentas del CNT entre flancos debe ser mayor de 30.000*
        - *Para pulsaciones superiores a 140 latidos por minuto, el número de cuentas del CNT entre flancos debe ser inferior a 8.600*
- *Reloj:*
    - *Un segundo ocurrirá cada 20.000 saltos del CNT.*

Entonces, usando el mismo TIM, el diagrama de bloques quedaría de la siguiente manera:



b) Con la explicación anterior, queda claro que sí que se van a utilizar IRQs, y esas interrupciones serán del TIM4 y con las siguientes funcionalidades:

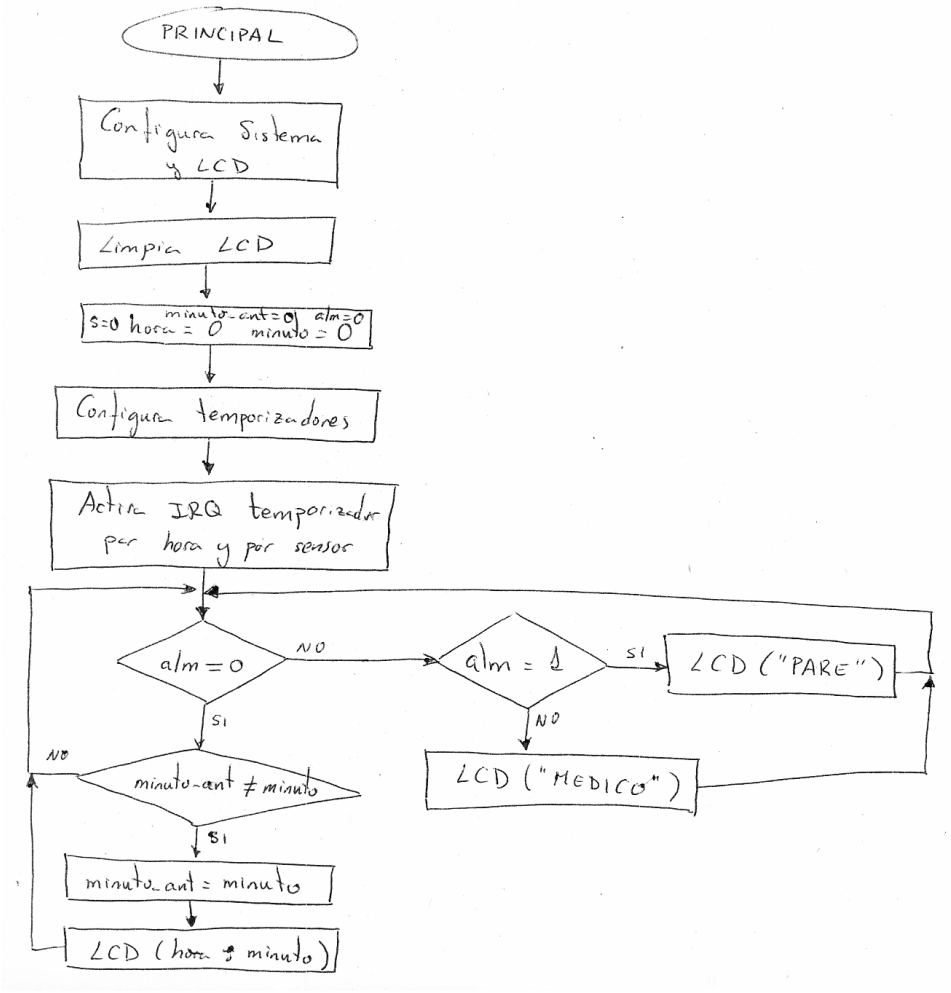
- *Canal 1 por TIC:*
  - *Saltará con cada flanco y actualizará el valor del ritmo cardíaco, obteniendo la diferencia con la captura anterior y sacando la frecuencia.*
- *Canal 2 por TOC:*
  - *Para actualizar el valor del CCR2 atendiendo a la frecuencia de señal a sacar.*
- *Canal 3 por TOC:*
  - *Para actualizar el número de segundos de la hora y, si es necesario, el valor de la hora.*

c) También se ha dado respuesta a los periféricos a utilizar, que serían el TIM4 y el NVIC. Los detalles de la configuración son:

- *Pre-escalado para que se cuenten intervalos de 0,05ms:*
  - *PSC = 1600 (con pclk = 32MHz)*
- *Habilitado todo el rango:*
  - *ARR = 0xFFFF*
- *Sin auto-reload*
  - *CR1[ARPE] = 0*
- *Inicialización del CNT*
  - *CNT = 0*
- *Canal 1 por TIC y flanco de subida habilitando la entrada:*
  - *GPIOB->MODER[PB6] = 10 (por AF)*
  - *GPIOB->AFR[0][AFRL6] = 2 (por AF2 (TIM3/TIM4))*

  - *CCMR1[CC1S] = 01*
  - *CCMR1[IC1F] = 0000*
  - *CCMR1[IC1PSC] = 00*
  - *CCER[CC1NP] = 0*
  - *CCER[CC1P] = 0*
  - *CCER[CC1E] = 1*
- *Canal 2 por TOC, con toogle y salida hardware:*
  - *GPIOB->MODER[PB7] = 10 (por AF)*
  - *GPIOB->AFR[0][AFRL7] = 2 (por AF2 (TIM3/TIM4))*
  - *CCMR1[CC2S] = 00*
  - *CCMR1[OC2CE] = 0*
  - *CCMR1[OC2M] = 011*
  - *CCMR1[OC2FE] = 0*
  - *CCER[CC2NP] = 0*
  - *CCER[CC2P] = 0*
  - *CCER[CC2E] = 0 (se deja inicialmente deshabilitada, se activa con la alarma)*
  - *CCR2 = 0 (se actualizará en el momento de saltar la alarma)*
- *Canal 3 por TOC, sin salida hardware*
  - *CCMR2[CC3S] = 00*
  - *CCMR2[OC3CE] = 0*
  - *CCMR2[OC3M] = 000*
  - *CCMR2[OC3FE] = 0*
  - *CCER[CC3NP] = 0*
  - *CCER[CC3P] = 0*
  - *CCER[CC3E] = 0*
  - *CCR3 = 20000*
- *Habilitada las interrupciones de CH1 y CH3 (la del 2 se activará cuando se active la alarma)*
  - *DIER[CC1IE] = 1*
  - *DIER[CC2IE] = 0*
  - *DIER[CC3IE] =1*
- *Habilitado el contador*
  - *CR1[CEN] = 1*
- *Forzar actualización de los registros:*
  - *EGR[UG] = 1*

d) Los diagramas de flujo serán:



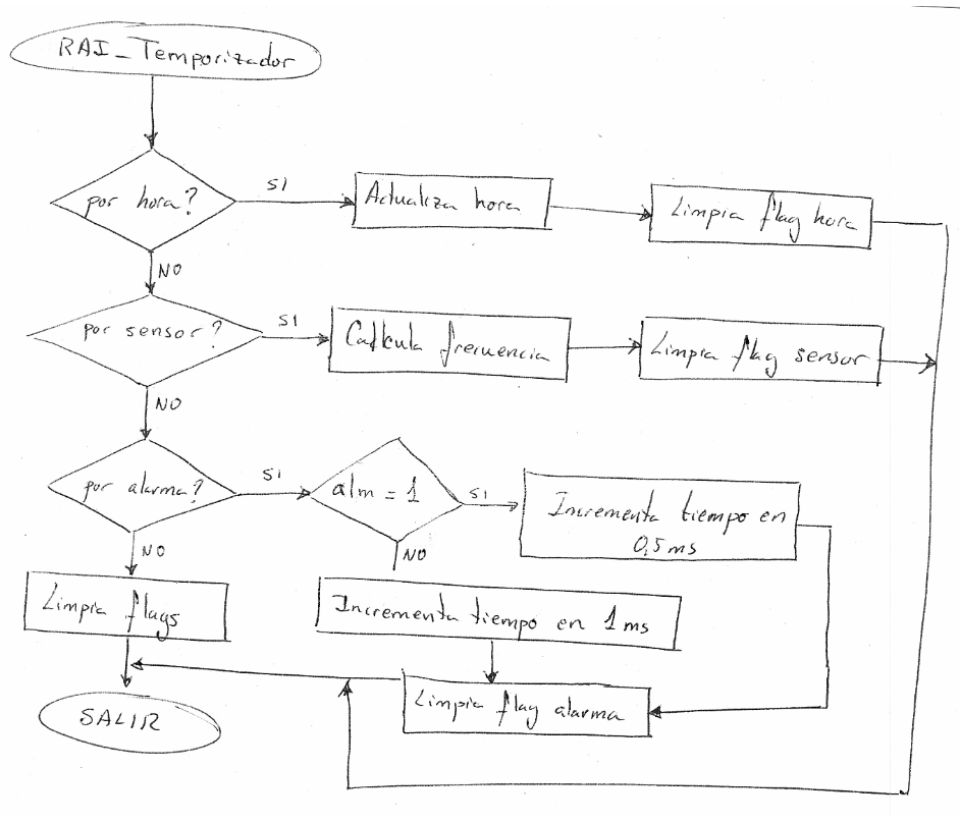
UNIVERSIDAD CARLOS III DE MADRID (Dpto. de Tecnología Electrónica)

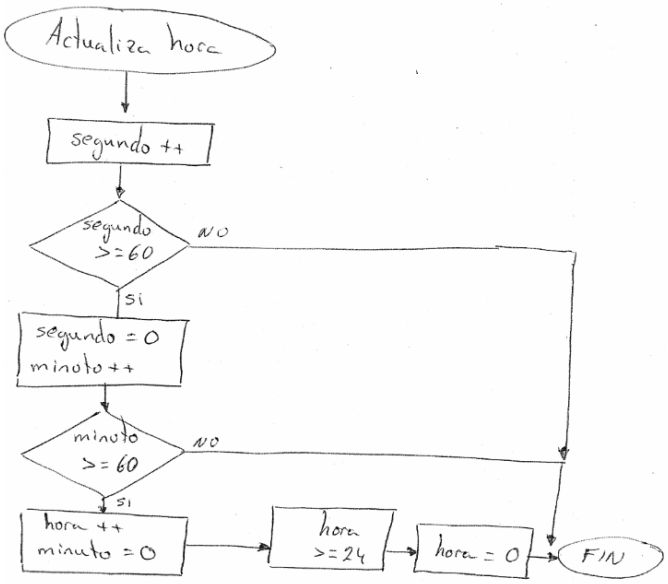
Sist. Dig. Basados en Microprocesador **(Gr. Ing. Telemática)**

17 de marzo de 2016 EXAMEN PARCIAL (100 minutos)

No se permiten ni libros, ni apuntes, ni calculadoras programables. Sólo se permite el manual del microcontrolador

Todas las respuestas deben estar justificadas.





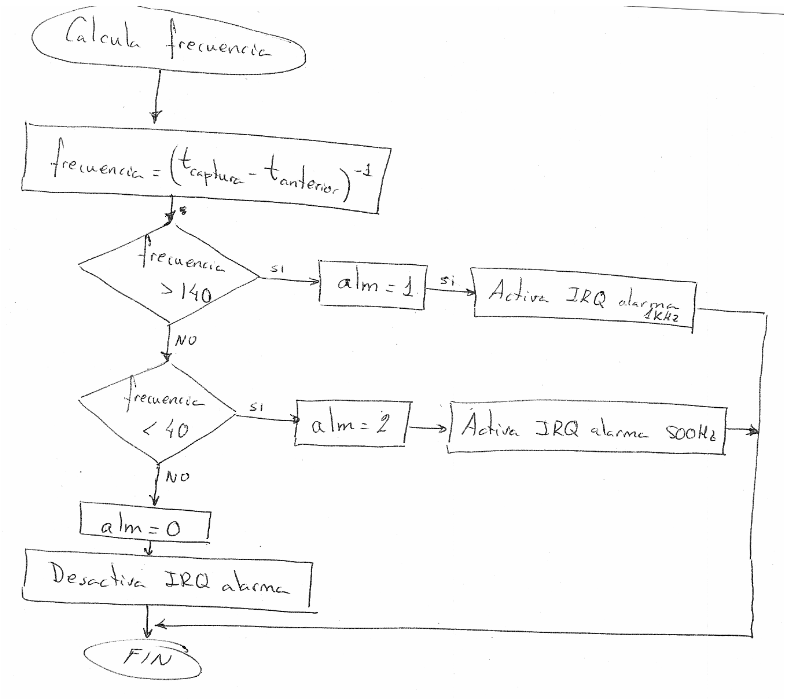
UNIVERSIDAD CARLOS III DE MADRID (Dpto. de Tecnología Electrónica)

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

17 de marzo de 2016 EXAMEN PARCIAL (100 minutos)

No se permiten ni libros, ni apuntes, ni calculadoras programables. Sólo se permite el manual del microcontrolador

Todas las respuestas deben estar justificadas.



PROBLEMA 2 (3 puntos):

El código siguiente se ha de ejecutar en la placa STM-Discovery, y su intención es utilizar la conversión analógico/digital de un único canal mediante interrupciones, sacando el valor convertido en el LCD. Dicho código está lleno de errores, por lo que se le solicite que los localice y proponga cómo arreglarlos.

SOLUCIÓN:

Los errores localizados son los siguientes:

- *La variable valor debería ser global, para que así pueda acceder a ella tanto el main como la RAI.*
  - *La línea 10 debería estar entre la 2 y la 4*
- *El pin que usa el ADC debería ser el PA0, mientras que el que está configurado como analógico es el PA3.*
  - *La línea 16 debería ser GPIOA->MODER |= 3 << (0*3);*
- *La interrupción está habilitada (tanto en el periférico como en el NVIC) antes de haber configurado el periférico.*
  - *Las líneas 17 y 18 deberían estar justo antes del while (1), y en ese mismo orden.*
  - *Sería aconsejable hacer una limpieza de los flags antes de activar el NVIC, por lo que la línea 17 debería estar antes de la 29 y la 18 después de la 29.*
- *La configuración del número de canales es incorrecta, porque en la línea 24 se dice que hay dos canales, y el enunciado habla de un único canal*
  - *La línea 24 debería ser ADC1->SQR1 = 0;*
- *Se está intentando escribir en el LCD un número binario en lugar de una cadena de caracteres.*
  - *Antes de la línea 32 debería haber una conversión de binario a ASCII de valor en texto y luego llamar a LCD_Texto(texto);*

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

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

17 de marzo de 2016 EXAMEN PARCIAL (100 minutos)

No se permiten ni libros, ni apuntes, ni calculadoras programables. Sólo se permite el manual del microcontrolador

Todas las respuestas deben estar justificadas.

APLICACIÓN

```
1  #include "stm32l1xx.h"
2  #include "Biblioteca_SDM.h"
3
4  void ADC1_IRQHandler (void) {
5     valor = ADC1->DR;
6     ADC1->SR = 0;
7     }
8
9  int main(void){
10    unsigned short valor;
11    unsigned char texto[6];
12    Init_SDM();
13    Init_LCD();
14    valor = 0;
15
16    GPIOA->MODER |= 3<<(2*3);
17    ADC1->CR1 = 0x00000022;
18    NVIC->ISER[0] = (1 << 18);
19    ADC1->CR2 &= ~(0x00000001);
20    ADC1->CR2 = 0x00000412;
21    ADC1->SMPR1 = 0;
22    ADC1->SMPR2 = 0;
23    ADC1->SMPR3 = 0;
24    ADC1->SQR1 = 1 << 20;
25    ADC1->SQR5 = 0x00000000;
26    ADC1->CR2 |= 0x00000001;
27    while ((ADC1->SR&0x0040)==0);
28    ADC1->CR2 |= 0x40000000;
29    ADC1->SR = 0;
30
31    while (1) {
32       LCD_Texto(valor);
33       }
34    }
35
```