UNIVERSIDAD CARLOS III DE MADRID (Dpto. de Tecnología Electrónica)
Sist. Dig. Basados en Microprocesador **(Gr. Ing. Telemática)**
29 de mayo de 2013 EXAMEN FINAL (3 horas)

EJERCICIO 1 (2.0 puntos):

En una arquitectura Harvard, con una memoria de programa de 32Kx16 y una memoria de datos de 128KB siendo los datos de 8 bits, y considerando que la CPU tiene 8 registros internos para datos, y que se implementa siguiendo una filosofía Load & Store:

1) (70%) Diseñe una codificación de los distintos tipos de instrucciones microprocesador, minimizando en lo posible el tamaño de la instrucción ajustando a números enteros de palabras. Los tipos de instrucciones que debe tener el microprocesador, son:

- 4 instrucciones de transferencia de datos entre memoria y registros internos, con direccionamiento absoluto.
- 7 instrucciones aritmético/lógicas de operar entre registros, indicando en la instrucción tanto los registros operandos, como el registro que almacena el resultado.
- 8 instrucciones aritmético/lógicas con un operando dado con direccionamiento inmediato, y donde el otro operando y el registro de resultado son el mismo.
- 3 instrucciones de control con direccionamiento inherente
- 8 saltos condicionales con direccionamiento relativo a contador de programa, siendo el desplazamiento relativo de más/menos 1KB

2) (30%) Indique una respuesta justificada a cada una de las siguientes preguntas:

a) Número mínimo de palabras que usa una instrucción
b) Número máximo de palabras que usa una instrucción
c) Tamaño del Registro de Instrucción
d) Tamaño del Contador de Programa
e) Tamaño de los Registros internos

EJERCICIO 2 (4.0 puntos):

Dado el código fuente del Anexo I (que puede tener errores o carecer de algunas funciones) y sabiendo que el dispositivo es un piano electrónico simplificado (fosc=32 MHz), responda a las siguientes preguntas:

1) Indique los periféricos utilizados y describa su configuración.
2) Dibuje el diagrama de bloques del sistema.
3) Indique el error de cuantificación del DAC
4) Dibuje de forma aproximada la onda de salida para la nota DO. Acote la amplitud, el periodo, el valor máximo, el valor mínimo, el valor medio y el tiempo entre dos muestras consecutivas cualquiera.
5) ¿Qué pasaría si pulsase y soltase una tecla?, ¿Qué pasaría si pulsase dos teclas a la vez y después soltase sólo una?
6) Dibuje el diagrama de flujo de la aplicación

EJERCICIO 3 (4.0 puntos):

Se necesita diseñar una estación de control que se comunica con un dispositivo externo a través de un puerto serie asíncrono con características 33600,8,N,1. La comunicación se plantea como un sistema maestro esclavo, donde la estación de control es el maestro, y el dispositivo es el esclavo. Por tanto, toda comunicación con el dispositivo se iniciará por parte del maestro, mediante el envío de una trama de bytes de, al menos, 3 bytes, donde el 3º indica la longitud en número de bytes del resto del comando. Antes de enviarlo, el comando se encontrará previamente formado en una variable llamada buffer_salida. Los envíos se realizarán por espera activa.
En cuanto se ha terminado de enviar un comando, se esperará a qué el dispositivo envíe una respuesta, la cual debería empezar a llegar antes de que pasen 300ms. La respuesta está compuesta de un número de caracteres superior o igual a 1, donde el primer carácter indica el número de bytes que contendrá la respuesta. Se debe cumplir que, una vez iniciada la respuesta, el tiempo máximo entre un carácter y otro sea de 50ms.
El sistema adicionalmente dispondrá de 3 LEDs, uno rojo, uno verde y uno amarillo. El LED amarillo se encenderá mientras se esté enviando el comando. El LED verde se encenderá cuando se esté recibiendo la respuesta. El LED rojo se encenderá cuando haya habido un error de timeout, hasta que el usuario haya pulsado un botón. Tanto los LEDs como el botón son activos a nivel alto.
Además, cuando haya ocurrido un error de timeout, se generará una señal de 1KHz durante 1 segundo, para que, sacada por un pin, active un altavoz y haga sonar un pitido de 1KHz.

Con estas especificaciones, se pide diseñar la estación de control utilizando un microcontrolador ST32L152RB, teniendo en cuenta que la estación realizará muchas más operaciones, y que, por lo tanto, no puede quedarse en bucles de espera activa, salvo en el momento de la transmisión. Para ello conteste a las siguientes preguntas (si necesita hacer alguna suposición o decisión adicional sobre el diseño, indíquelo claramente):

a) Diagrama de bloques de la estación (10%)
b) ¿Necesitaría definir rutinas de atención a la interrupción? Si es que sí, indique cuáles, por qué y que función tendría. No realice aquí el diagrama de flujo. (20%)
c) Configuración inicial de los periféricos involucrados, con explicación de dicha configuración. (20%)
d) Diagrama de flujo del programa principal y de todas aquellas rutinas necesarias. (50%)

Anexo I

```
#include "stm32l1xx.h"
#include "Biblioteca_SDM.h"

/*****************************************************************************
Do 1: 65,406 Hz  -> 805 us entre muestras
                 (20 muestras -> 19 intervalos -> 1/(65,406 * 19) = 805 us)
Re 1: 73,416 Hz  -> 717 us
Mi 1: 82,407 Hz  -> 638 us
Fa 1: 87,307 Hz  -> 603 us
Sol 1: 97,999Hz  -> 537 us
La 1: 110 Hz     -> 478 us
Si 1: 123,471 Hz -> 426 us
*****************************************************************************/

#define DO              805   // us entre muestras
#define RE              717
#define MI              638
#define FA              603
#define SOL             537
#define LA              478
#define SI              426

#define ESTADO_DO       0     // estados
#define ESTADO_RE       1
#define ESTADO_MI       2
#define ESTADO_FA       3
#define ESTADO_SOL      4
#define ESTADO_LA       5
#define ESTADO_SI       6
#define SIN_ESTADO      7

#define ON              1
#define OFF             0

// Variables globales
int estado = SIN_ESTADO;
int usCuentaAtras = 5000;
int flagCuentaAtras = OFF;
int index=0;


// Valores de la señal de salida
int senoide[19] ={2048, 2713, 3306, 3762, 4033, 4089, 3924, 3555, 3023, 2385, 1711,
1073, 541, 173, 7, 63, 333, 790, 1383};

// Tiempos entre muestras para cada nota
int sample[7] = {DO, RE, MI, FA, SOL, LA, SI};
```

```
void TIM4_IRQHandler(void) {

   if ((TIM4->SR & 0x0002)!=0) {

            if (estado != SIN_ESTADO){

                  index++;
                  if(index >= 20) index = 0;

                  if(flagCuentaAtras == ON){
                        usCuentaAtras -= sample[estado];


                        if(usCuentaAtras <= 0){
                              estado = SIN_ESTADO;
                              TIM4->CR1 &= ~0x0001;
                        }
                  }

            }

            // reset timer
            TIM4->CNT = 0;
            TIM4->SR = 0x0000;   // Limpia todos los flags
      }
}


//
void config_toc4ch1(int tiempo){

   //
   TIM4->CR1 = 0x0000;
   TIM4->CR2 = 0x0000;
   TIM4->SMCR = 0x0000;
   TIM4->PSC = 31;
   //
   TIM4->CNT = 0;
   TIM4->ARR = 0xFFFF;
   TIM4->CCR1 = tiempo;
   //
   TIM4->DCR = 0;
   TIM4->DIER = 0x0002;
   // Modo de salida
   TIM4->CCMR1 = 0x0000;
   TIM4->CCMR2 = 0x0000;
   TIM4->CCER = 0x0000;
   //
   TIM4->CR1 |= 0x0001;

   TIM4->EGR |= 0x0001;
   TIM4->SR = 0;
   NVIC->ISER[0] |= (1 << 30);

}
```

```
void config_dac(void){
   //
   GPIOA->MODER |= 0x00000C00;
   DAC->CR = 0x00010000;
}

//
void config_gpio(void){
    GPIOB->MODER &= 0xC000;
}

//
int get_tecla(void){

      int tecla = -1;

    if ( (GPIOA->IDR&0x00000001) == (1<<0) ) {
            tecla = 0;
      }else if( (GPIOA->IDR&0x00000002) == (1<<1) ){
            tecla = 1;
      }else if( (GPIOA->IDR&0x00000004) == (1<<2) ){
            tecla = 2;
      }else if( (GPIOA->IDR&0x00000008) == (1<<3) ){
            tecla = 3;
      }else if( (GPIOA->IDR&0x00000010) == (1<<4) ){
            tecla = 4;
      }else if( (GPIOA->IDR&0x00000020) == (1<<5) ){
            tecla = 5;
      }else if( (GPIOA->IDR&0x00000040) == (1<<6) ){
            tecla = 6;
      }

      return(tecla);

}

int main(void){

   int index_old = index;
   int tecla = -1;
   int tecla_old = -1;

   Init_SDM();
   config_toc4ch1(1000);
   config_dac();
   config_gpio();

   while (1) {

        // coge la tecla pulsada
        tecla = get_tecla();

        // si ha cambiado
        if(tecla != tecla_old){

                  tecla_old = tecla;
```

```
                //
                if(tecla >= 0){
                      estado = tecla;
                      config_toc4ch1(sample[estado]);
                      index = 0;
                      flagCuentaAtras = OFF;

                }else{
                      //
                      usCuentaAtras = 5000;
                      flagCuentaAtras = ON;
                }

        }

      //
      if (index_old != index) {
            index_old = index;
            DAC->DHR12R2 = senoide[index];
      }
   }
}
```