



ASIGNATURA	SISTEMAS ELECTRÓNICOS DIGITALES AVANZADOS	FECHA	MAYO 2017
APELLIDOS, NOMBRE	SOLUCIÓN		GRUPO

PRUEBA DE EVALUACIÓN FINAL

CUESTIÓN 1

El diagrama de bloques de la figura 1 muestra un sistema de control a distancia por infrarrojos de un servomotor basado en el LPC1768 (Fcpu=100Mhz). El sistema consta de un módulo transmisor y un módulo receptor. El módulo transmisor (TX) dispone de un potenciómetro que permite controlar la posición del servomotor conectado al módulo receptor (RX) en un rango de 0 a 180º variando su recorrido.

El sistema de codificación de infrarrojos se muestra en el Anexo I. Tenga en cuenta que la portadora es una señal periódica cuadrada de **38kHz** (zona pintada en negro de la señal). La posición del potenciómetro conectado a **AD0.1** se codifica con **7 bits** y se transmite periódicamente cada **50ms** comenzando por el LSB.

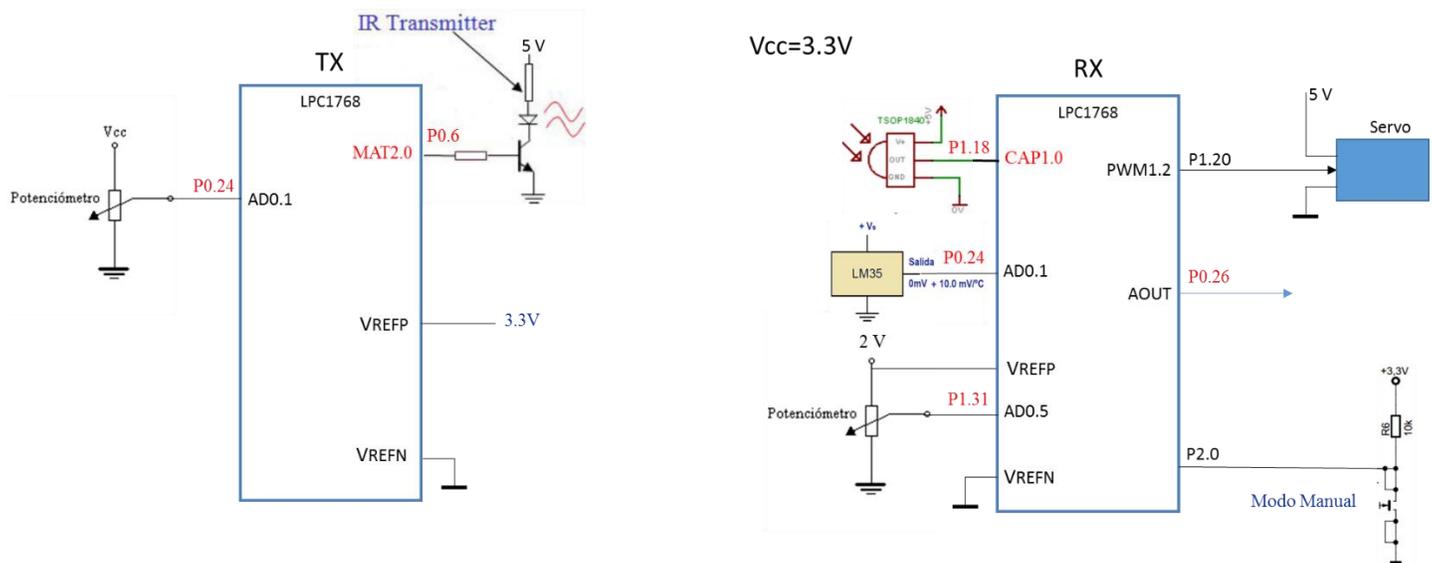


Figura 1. Diagrama de bloques del sistema

El módulo receptor (RX) se encarga de decodificar la señal infrarroja y posicionar el servo en función del código de **7 bits** recibido. El potenciómetro conectado a **AD0.5** (Modo manual) permite mover el servo en su recorrido (0-180º) **mientras se mantiene cerrado el pulsador** conectado a P2.0, a la vez que se varía la frecuencia obtenida a la salida del DAC entre **100 y 1 kHz**.

Un sensor de temperatura (LM35) adosado al servo nos permite monitorizar su temperatura cada minuto.



CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
 LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
 CALL OR WHATSAPP:689 45 44 70

## Módulo Transmisor (TX)

- b) Explique **sin escribir el código** la configuración del **Timer 2** y el modo de funcionamiento para generar **exclusivamente** con dicho Timer la modulación y la transmisión del código según el **Anexo 1**.

**NOTA:** Utilice el Match 0 para generar la portadora de 38kHz y el Match 1 para medir los múltiplos de  $T=600\mu s$ . La función de interrupción incompleta está en el apartado **d)**

Timer 2 en modo salida por comparación (MAT2.0)

Habilitar salida MAT2.0 (P0.6) en PINSELx

Prescaler=24  $\rightarrow$  Ftick= 1Mhz (así, Ttick= 1us para simplificar los cálculos)

**Hab. interrupción MATCH 0 y MATCH 1 !!!  $\rightarrow$  para generar la portadora y medir tiempos (T, 2T y 4T)**

Habilitamos el modo Toggle cuando sea necesario sacar la portadora (38kHz  $\rightarrow$   $T \approx 26\mu s$ ) durante:

- **4T** para el bit START
- **2T** para un "1"
- **T** para un "0"

**Modo RUNNING!!!!  $\rightarrow$  No se resetea el Timer!!!**

En la rutina de interrupción del Timer 2, con Match 0 (en modo toggle) generamos la portadora, y con el Match 1 medimos los tiempos (4T, T, y 2T)

- **T2MR0+=13-1 durante el tiempo que haya que generar la portadora ( $T/2=13\mu s$ )**
- **T2MR1+=2400-1 ( para el bit START)**
- **T2MR1+=600-1 ó 1200-1 ( para el bit "0" o el bit "1")**
- **Siempre después de cada bit transmitido (Portadora en ON) MAT2.0=0 durante 600us.** Pin MAT2.0 =0 (manualmente en EMR para garantizar que la salida quede a nivel bajo)

Cada vez que **interrumpe el ADC** ( $F_s=20\text{Hz}$ ) se lee el dato del canal 1 (proporcional a la posición del potenciómetro) y se arranca el Timer 2 para transmitir el código de 7 bits.

**Después del reset:**

Stop Timer 2, Toggle habilitado, T2MR0=12 para generar la portadora, T2MR1=2400 para generar el bit START

- c) Complete las líneas de la función de interrupción del ADC para guardar en una variable global el código de 7 bits correspondiente a la posición del potenciómetro.



CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70

- d) Complete y comente el código de la función de interrupción del **Timer 2** que permite generar la señal modulada que excita el transistor NPN (ver Anexo I)

```

void Timer2_IRQHandler(void)
{
static char contador;

if(LPC_TIMER2->IR&(1<<0)){
    LPC_TIMER2->MR0+=13-1; // Ftick/Fport/2 -1
    LPC_TIMER2->IR|=0x01; // Borrar flag Match 0
}

else {
    LPC_TIMER2->IR|=0x02; // borrar flag
    // Siempre portadora en OFF(salida a nivel bajo durante 1T)
    if((portadora_ON)==0)&&(contador< 7 ) ) {

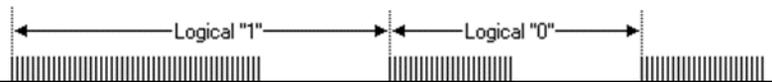
        LPC_TIMER2->EMR&=~(0x03<<4); // Deshabilitamos modo Toggle;
        LPC_TIMER2->EMR&=~(1<<0); // MAT2.0=0 (garantizamos un nivel bajo)
        LPC_TIMER2->MR1+=600-1 ; // Match para 1T
        portadora_ON=1;
    }

    else if((portadora_ON)==1)&&(contador< 7 )){

        LPC_TIMER2->EMR =(3<<4); // Habilitamos modo Toggle;
        if (tx_codigo&0x01) LPC_TIMER2->MR1+=1200-1 ;
        else LPC_TIMER2->MR1+=600-1 ;
        tx_codigo>>=1;
        portadora_ON=0;
        contador++;
    }

    else {
        LPC_TIMER2->TCR=0x02; // Stop Timer
        LPC_TIMER2->EMR&=~(1<<0); // MAT2.0=0 (garantizamos un nivel bajo)
        contador=0;
        LPC_TIMER2->MR1+=2400-1 ; // Preparo MR1 para START
    }
}
}

```



CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70

Cartagena99

## Módulo Receptor (RX)

- e) Complete la función de inicialización del ADC si la frecuencia de muestreo del canal 1 es de aproximadamente 10kHz. Calcule la frecuencia real de muestreo obtenida y proponga una solución simple para aproximar mejor a la frecuencia de muestreo deseada.

```
void init_ADC(void)
{
LPC_SC->PCONP|= (1<<12);           // Power ON
LPC_PINCON->PINSEL1|= ( 1<<16 );   // P0.24 es AD0.1
LPC_PINCON->PINSEL3|= ( 3<<30 );   // P1.31 es AD0.5
LPC_PINCON->PINMODE1|= (2<<16);    // Deshabilita pullup/pulldown
LPC_PINCON->PINMODE3|= (2<<30);    // Deshabilita pullup/pulldown
LPC_ADC->ADCR= (0b00100010<<0 )|   // Canales 1 y 5
              (18<<8)|             // Fclk=25/19 Mhz
              (1<<21)|             // PDN=1
              (1<<16);             // Modo BURST ;!
LPC_ADC->ADINTEN=0;                 // Sin interrupción
}
```

$F_s = 25e6 / 2 * 65 * (CLKDIV + 1) = 10 \text{ kHz} \rightarrow CLKDIV = 18 \rightarrow F_s \text{ real} = 10,121 \text{ kHz}$

Modificamos el prescaler del ADC ( $\div 1$ ,  $\div 2$ ,  $\div 4$ ,  $\div 8$ ) y vemos que valor de  $F_{pclk}$  hace que  $CLKDIV$  se aproxime mejor a un entero.

Si  $F_{pclk} = 100 \text{ Mhz} \rightarrow CLKDIV + 1 = 76,92 \rightarrow CLKDIV = 76 \rightarrow F_s \text{ real} = 9,99 \text{ Khz}$

- f) A la vista de la función de interrupción del **Timer 3** que saca las muestras hacia el DAC para generar la señal senoidal, complete la función que genera previamente las muestras de la función seno, y calcule la frecuencia máxima de interrupción.

NOTA: Considere declarado el array *muestras[ ]*, y que  $V_{ref} = 2V$ .

```
void genera_muestras_seno(void)
{
char i;
for(i=0; i<64 ; i++) muestras[i] = (511 + 511 * sin(2*pi*i/64)) << 6;
}
```

$F_{max} \text{ interrupción Timer3} = 64 * 1 \text{ kHz} = 64 \text{ kHz}$



CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP: 689 45 44 70

- g) Explique en detalle, sin escribir código, qué recurso utilizaría y como lo configuraría para reducir la carga de CPU durante la generación de la señal senoidal.

Evitaríamos la ejecución de la interrupción del Timer3 empleando el DMA en modo *Linked* con el propio canal (ej. Canal 0), para que al finalizar la transferencia se inicie de nuevo sin que sea necesaria la interrupción del DMA.

```
LPC_GPDMA0->DMACCSrcAddr = (uint32_t) &muestras[0];           // Fuente Memoria
LPC_GPDMA0->DMACCDestAddr = (uint32_t) &(LPC_DAC->DACR);      //Destino Periférico (ojo 16 bits)
```

Transfer Size=64;

Tamaño transferencia= **16 bits**; las muestras del array ya están desplazadas al bit 6

Incrementa Fuente;

No incremento Destino

Sin interrupción (ojo, modo Linked)

Para modificar la Fout (ej. en la rutina del SysTick):

```
LPC_DAC->DACNTVAL = (F_pclk/64/Fout) -1; // Frecuencia transf. DMA
```

- h) Explique en detalle, sin escribir código, qué recurso utilizaría y como lo configuraría para decodificar la señal obtenida a la salida del detector de IR.

Entrada de captura CAP1.0, configurada en flanco de bajada.

Prescaler timer =24 (para simplificar los cálculos) →  $F_{tick} = F_{pclk}/(PR+1) = 1\text{MHz}$  ( $T = 1\mu\text{s}$ )

Al entrar la primera vez la interrupción se captura el valor y se guarda, se cambia la captura a flanco de subida.

Para reducir el tiempo de ejecución trabajamos con Cuentas del Timer (no medimos tiempos!!!!)

Una vez detectado el bit Start (tras dos interrupciones y ver comprobado la duración 4T) queda configurada **siempre la entrada en flanco de subida, así resulta más sencillo detectar los bits:**

2T-> 1200 cuentas → para el "0"

3T-> 1800 cuentas → para el "1"



**Una vez detectado el bit START, y dentro de la función de interrupción:**

```
N= LPC_TIM1->CRO - TEMP;           // cuentas entre flanco actual y anterior
if ( (N>1190) && (N<1210) ) codigo>>=1; // se ha recibido un "0", sólo desplazamos ; margen de tolerancia ± 10us
else if ( (N>1790) && (N<1810) ) { // se ha recibido un "1"
    codigo>>=1;
    codigo+=1;
}
```

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70

**NOTA: Una mejora es añadir un watchdog con un Match del timer, para que si no se han recibido todos los bits antes de T<sub>máx</sub> se reinicie todo el proceso. Esto evita que se quede un código incompleto en la recepción.**

- i) Escriba el código de la función de interrupción del SYSTICK (periodo 50ms) que se encarga de leer el estado del pulsador y de modificar la posición del servo en función del potenciómetro, a la vez que modifica el valor de la frecuencia de salida del DAC, así como de almacenar en una variable global la temperatura. Añada las instrucciones necesarias para actualizar cada minuto en un LCD la temperatura y enviarla por el puerto serie (ver Anexo 2).

NOTA: Considere que la señal PWM del servo ya está configurada y para un periodo de **15ms**, y que el tiempo a nivel alto varíe entre **0.8-2.4ms**, para un movimiento de su posición entre 0º y 180º. Considere que el temporizador asociado al PWM incrementa la cuenta cada microsegundo.

```
void SysTick_IRQHandler(void)
{
    cont++;
    if ((LPC_GPIO2->FIOPIN&0x01)==0) {

        if(codigo!=(LPC_ADC->ADDR5>>8)&0xFF){ // si cambia el valor

            LPC_PWM1->MR2=(uint32_t)(0.8e3 + 1.6e3*codigo/255); // Ftick=1Mhz ;!
            LPC_PWM1->LER|=(1<<2)|(1<<0);
            Fout= 100 + 900*codigo/255; // Fout salida del DAC
            LPC_TIMER3->MR0=Fpclk/Fout/64 -1; // 64 muestras/ciclo
            LPC_TIMER3->TC=0;
            LPC_TIMER3->TCR=0x01; // Start Timer
        }

    }

    else LPC_TIMER3->TCR=0x02; // Stop Timer

    codigo=(LPC_ADC->ADDR5>>8)&0xFF; // guardamos la posición del potenciómetro

    //cada minuto leo la temperatura y la envio por la UART0 y al LCD
    if(cont%(20*60)==0)
    {
        temperatura=((LPC_ADC->ADDR1>>4)&0xFFF)*200/4095; // grados=mV/10
        sprintf(cadena,"temperatura= %2.1f grados",temperatura);
        tx_cadena_uart0(cadena);
        GUI_Text(10,10, cadena, WHITE, BLACK);
    }
}
}
```



Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70

**CUESTIÓN 2.**

Se desea controlar un sistema empotrado basado en el LPC1768 mediante una conexión WIFI, para lo cual se conecta con la UART3 el módulo ESP8266.

- Complete el diagrama de conexión del LPC1768 con el módulo WIFI.
- Complete la función de configuración del puerto serie y **realice los cálculos** para configurar la velocidad a 115200 baudios, 8 bits por dato y sin paridad.
- Calcule la velocidad real obtenida y el tiempo que tarda en transmitirse el comando enviado al ejecutarse la sentencia, `tx_cadena_UART3("Ejecutando...\n\r")`. Ver **Anexo 2**
- Qué parámetro cambiaría (que no sea DivAddval ni MulVal) para mejorar la exactitud de la velocidad real.

```
void uart3_init(void)
{
    LPC_PINCON->PINSEL9|=3<<26; // RXD3 (P4.29);
    LPC_PINCON->PINSEL9|=3<<24; // TXD3 (P4.28);
    LPC_UART3->LCR=0x83;
    LPC_UART3->DLL= 14;
    LPC_UART3->DLM= 0;
    LPC_UART3->LCR= 0x83;
}
```

Consideramos FR=1

$$DL_{16} = 25e6 / (16 * 115200) * FR = 13,56 \rightarrow 14$$

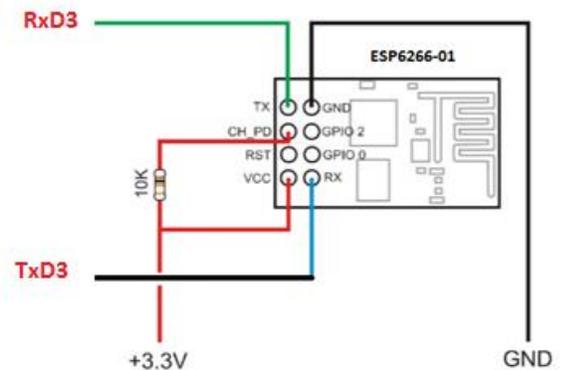
$$Vt = 25e6 / (16 * 14) = 111.607 \text{ baudios}$$

$$T_{mensaje} = (1/Vt) * N^{\circ} \text{ caracteres} * 10 \text{ bits/carácter} = 8.96e-6 * 15 * 10 = 1,34ms$$

Modificando PCLK de la UART a 50 Mhz (configurar el prescaler ÷2) resulta:

$$DL_{16} = 50e6 / (16 * 115200) * FR = 27,12 \rightarrow 27$$

$$Vt = 50e6 / (16 * 27) = 115.740,7 \text{ baudios}$$



Cartagena99

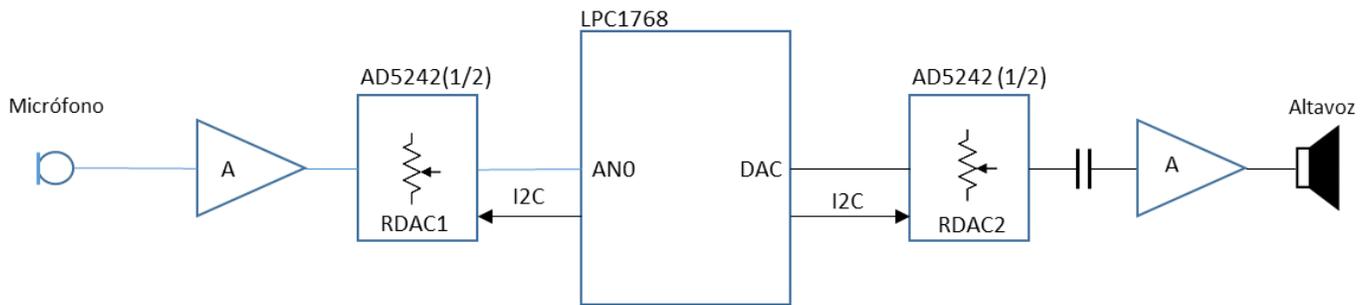
CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

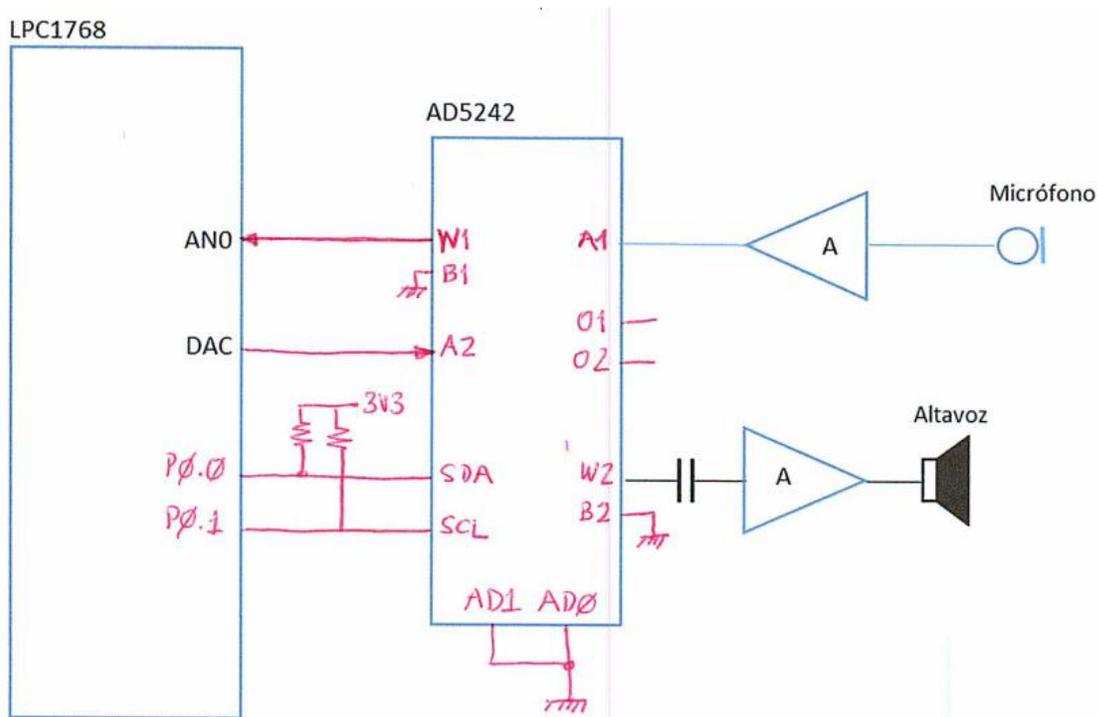
ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70

**CUESTIÓN 3.**

El diagrama de bloques del circuito de la figura representa un sistema de grabación/reproducción de audio en el que se utilizan dos potenciómetros digitales para controlar la ganancia del micrófono y el volumen del altavoz.



Dibuja detalladamente el diagrama de conexión del AD5242 con el LPC1768 teniendo en cuenta la librería de funciones del bus I2C (SDA a P0.0 y SCL a P0.1) y escriba la función que permite variar el volumen de salida entre el 0 y 100%



```
void I2CSendByte(unsigned char byte);
void I2CSendAddr(unsigned char addr, unsigned char rw);
unsigned char I2CGetByte(unsigned char ACK);
void I2CSendStop(void);
```

*i2c\_volumen(char porcentaje)*



**CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70**

---

**ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70**



## CUESTIÓN 5

Se desea realizar el control del tren de aterrizaje de un avión. El sistema tiene los siguientes elementos:

- Un sensor situado en el amortiguador del tren de aterrizaje delantero indica si el avión está en el suelo o está en el aire (si hay peso o no hay peso en el amortiguador). La variable que contiene el estado de este sensor se llama **NOSE** y puede tomar los valores '**Ground**' o '**Air**'.
- Una palanca situada en el control de mandos permite al piloto subir o bajar el tren de aterrizaje. La variable donde se almacena el estado se llama '**LEVER**' y puede tomar los valores '**Up**' o '**Down**'.
- El panel de mandos tiene tres luces relacionadas con el tren de aterrizaje, una por cada uno de los tres conjuntos de ruedas. Estas luces pueden estar en verde, rojo, apagadas o parpadeando entre rojo y verde. Las variables en las que habrá que escribir para cambiar el estado de los indicadores son **LED\_1**, **LED\_2** y **LED\_3** y pueden tomar los estados '**Green**', '**Red**', '**Black**' y '**Flash**'. La luz de cada tren estará en verde cuando el tren de aterrizaje esté extendido (abajo), en rojo cuando esté en tránsito (bajando o subiendo), apagada cuando esté replegado (arriba) y parpadeando cuando se haya producido un fallo.
- Hay dos finales de carrera en cada uno de los trenes de aterrizaje, uno para indicar que está completamente replegado (**FC\_UP\_1**, **FC\_UP\_2**, **FC\_UP\_3**) y otro que indica que está completamente extendido (**FC\_DOWN\_1**, **FC\_DOWN\_2**, **FC\_DOWN\_3**). Cuando están presionados dan el valor '**On**' y cuando no están presionados '**Off**'.
- Una señal de control pone en marcha el sistema hidráulico (**PUMP=On**) o lo desactiva (**PUMP=Off**) y otra indica el sentido del movimiento que hace que el tren de aterrizaje suba y o baje (**DIR** que puede tomar los valores '**Up**' o '**Down**').



El funcionamiento del sistema debería ser el siguiente:

- Cuando el avión está en tierra lo normal es que la palanca del tren de aterrizaje esté en posición **Down** y los tres indicadores correspondientes en verde.
- Si estando en tierra (**NOSE='Ground'**) se pusiera por error la palanca en modo **Up**, el tren de aterrizaje no debería moverse.
- Cuando avión está en vuelo y se activa la palanca indicando la subida o bajada del tren de aterrizaje, el sistema deberá esperar **5 segundos** antes de responder para que, si fuera una activación involuntaria de la palanca que el piloto, tuviera tiempo de corregir la posición.
- Pasados los **5 segundos**, se pondrán los tres indicadores luminosos en rojo y se darán las órdenes de movimiento al sistema hidráulico.

bajar el tren de aterrizaje será el mismo pero finalizando con todas las luces en '**Verde**'.

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70

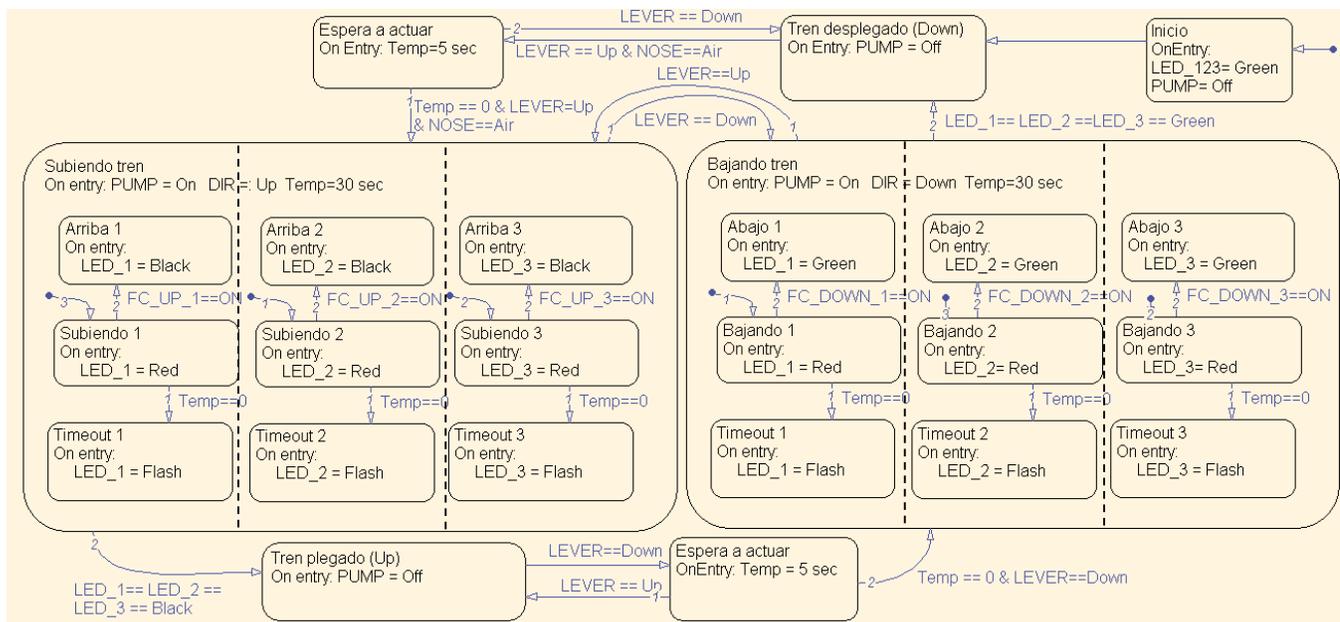
Cartagena99

- Si el tren de aterrizaje está subiendo y el piloto indica que vuelva a bajarse, deberá el sistema deberá reaccionar **inmediatamente** cambiando de sentido.
- Cuando se notifique un error en la extensión o replegado del tren de aterrizaje, el piloto deberá poder intentar subir o bajar de nuevo el tren de aterrizaje actuando sobre la palanca.

Se pide:

- Indique las señales de entrada y salida del sistema de control.
- Sabiendo que el diagrama adjunto es el esqueleto de una posible solución de StateChart que modela el comportamiento del sistema. Complete el diagrama con nombres de los estados, eventos y acciones para que modele el comportamiento que se indica en el enunciado.

Entradas al sistema de control	Salidas del sistema de control
<p><b>NOSE</b>  <b>LEVER</b>                      FC_UP_1, FC_UP_2, FC_UP_3                      FC_DOWN_1, FC_DOWN_2, FC_DOWN_3</p>	<p><b>PUMP</b>  <b>DIR</b>                      LED_1, LED_2 y LED_3</p>



**CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE**  
**LLAMA O ENVÍA WHATSAPP: 689 45 44 70**

---

**ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS**  
**CALL OR WHATSAPP:689 45 44 70**

## CUESTION 6

Suponiendo un sistema con un LPC1768 con las tareas que se indican a continuación, identifique los parámetros **D** y **T** de cada una de ellas y **justifique el valor explicando el subsistema interno del microcontrolador que utilizaría** y en qué modo se configuraría. Tenga en cuenta que es un único microcontrolador con recursos limitados.

- Generación de una señal cuadrada con un **ciclo de trabajo de un 50%** de frecuencia variable entre 1kHz y 100kHz.
- Generación de una señal PWM de 10kHz de frecuencia con un ciclo de trabajo que oscila entre un 10% y un 90% **utilizando el Match** de un temporizador.
- Reproducción repetida de un mensaje de audio de 5 segundos de duración con una frecuencia de muestro de 8kHz y **muestras de 10 bits** utilizando DMA.
- Lectura de **4 sensores** analógicos a una frecuencia de muestreo de 10kHz.
- Medida del ciclo de trabajo de una señal que 10kHz con un ciclo de trabajo que oscila entre un 10% y un 90%

## Solución

- La señal más restrictiva en lo que respecta a ejecutabilidad es la de 100kHz con 10us de periodo. Se puede generar la señal con un Match configurado en modo Toggle incrementando el valor del Match con cada interrupción sin parar el temporizador. En este caso  $D = T = 5\mu s$
- Se puede configurar el Match asociado a un pin para que, dependiendo del valor de un flag que se conmuta con cada interrupción:
  - Si el flag es por ejemplo '1', configura la salida asociada para que genere un nivel bajo al llegar en Match y configurar el Match para que interrumpa pasado un TH además de conmutar el flag.
  - Si el flag es '0', configura la salida asociada para que genere un nivel alto al llegar en Match y configurar el Match para que interrumpa pasado un TL además de conmutar el flag.

En este caso el tiempo mínimo en interrupciones se produce en los extremos del 10% y 90% con  $D=T=10\mu s$

- Si el mensaje es repetido, se puede configurar uno o varios canales de DMA en modo linked con transferencias de 16bits entre memoria y el DAC configurando el temporizador del DAC como corresponda. En este caso no influiría en la ejecutabilidad por no tener interrupción asociada.
- Si es necesario procesar los datos con cada muestreo, una opción sería configurar el ADC en modo ráfaga configurando el tiempo de conversión en  $100\mu s/4$  habilitando la interrupción de uno de los canales para leer las 4 muestras. En este caso hay una interrupción cada 100us por lo que  $T=100\mu s$  pero los datos hay que leerlos antes de que se produzca una nueva conversión de uno de los canales por lo que  $D=100/4 = 25\mu s$ .
- El ciclo de trabajo se puede leer con una entrada de captura. Se produciría una interrupción en el peor de los casos cada 10us:  $T = D = 10\mu s$ .



CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70

## CUESTIÓN 7

Suponiendo que un sistema basado en el LPC1768 tiene tres tareas con los parámetros que se indican en la tabla y el programa principal tiene una región crítica de 1ms

Tarea	Prioridad	Subprioridad	C	T	D
Tarea 1	0	1	2	10	5
Tarea 2	0	2	5	30	30
Tarea 3	0	3	10	30	30

**Nota:** Unidades en ms

Se pide:

- Analice si el sistema es ejecutable.
  - Si el sistema es ejecutable:
    - Calcule la duración máxima de una región crítica del programa principal que haría que el sistema dejara de ser ejecutable. Demuestre que es ejecutable.
- Si el sistema no es ejecutable:
- Modifique las prioridades y/o las subprioridades de manera que el sistema sea ejecutable. Demuestre que es ejecutable.

### Solución

a)

Tarea 1:

$$R_1 = C_1 + B_1 + I_1 = C_1 + \max(C_2, C_3, RC) + 0 = 1 + 10 = 11 > D_1 \rightarrow \text{No ejecutable}$$

b)

Una opción sería cambiar la prioridad como se indica en la tabla

Tarea	Prioridad	Subprioridad
Tarea 1	0	0
Tarea 2	1	0
Tarea 3	2	0

Tarea 1:

$$R_1 = C_1 + B_1 + I_1 = C_1 + RC + 0 = 1 + 1 = 2 \leq D_1 = 5\mu s$$

Tarea 2:

$$R_2 = C_2 + B_2 + I_2 = C_2 + RC + \left\lceil \frac{R_2}{T_1} \right\rceil \cdot C_1 = 5 + 1 + \left\lceil \frac{R_2}{10} \right\rceil \cdot 2$$

$$w^0 = 5 + 1 + 2 = 8$$

$$w^1 = 6 + \left\lceil \frac{w^0}{10} \right\rceil \cdot 2 = 6 + \left\lceil \frac{8}{10} \right\rceil \cdot 2 = 8 \rightarrow R_2 = 8\mu s \leq D_2 \leq 30\mu s$$

Tarea 3:

$$R_3 = C_3 + B_3 + I_3 = C_3 + RC + \left\lceil \frac{R_3}{T} \right\rceil \cdot C_1 + \left\lceil \frac{R_3}{T} \right\rceil \cdot C_2 = 10 + 1 + \left\lceil \frac{R_3}{10} \right\rceil \cdot 2 + \left\lceil \frac{R_3}{30} \right\rceil \cdot 5$$

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70

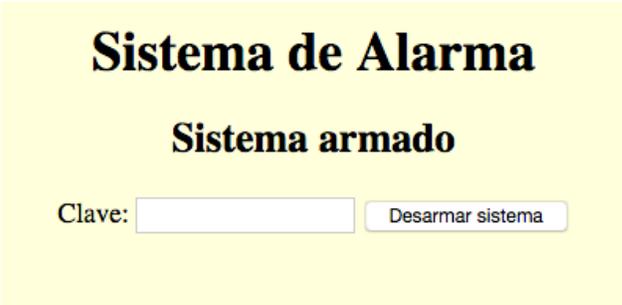
Cartagena99

## CUESTIÓN 8.

En una aplicación de monitorización remota de un sistema de alarma, se desea visualizar remotamente el estado del sistema y posibilitar su activación o desactivación al acceder a la página **alarma.cgi**. El estado del sistema se almacena en la variable global **uint8\_t estado** que podrá tomar los valores 0 (sistema desarmado), 1 (sistema armado) y 2 (alarma activa). Además de indicar textualmente el estado del sistema, el fondo de la pantalla dependerá del estado: verde (sistema desarmado), amarillo (sistema armado), y rojo (alarma activada). Por seguridad, para poder armar o desarmar el sistema, deberá introducirse una clave que deberá coincidir con la cadena de texto almacenada en el array **char password[10]**. Si este código es correcto se cambiará el estado accediendo a la variable **estado**. En la tabla siguiente se muestra el código HTML y la salida visualizada en el navegador dependiendo del estado.

Indique el contenido del fichero **alarma.cgi** al que accedería el navegador y el contenido de las funciones **cgi\_func(...)**, **cgi\_process\_data(...)**, y/o **cgi\_process\_var(...)** necesarias. Suponga definidas las siguientes variables:

```
const char* color[]={"lightgreen","lightyellow","red"};
const char* info[]={"Sistema desarmado","Sistema armado","Alarma activa"};
const char* boton[]={"Armar sistema","Desarmar sistema"," Desarmar sistema"};
```

<pre>&lt;!DOCTYPE html&gt; &lt;html&gt; &lt;head&gt; &lt;meta http-equiv='refresh' content='15'&gt; &lt;/head&gt; &lt;body style='background-color:lightgreen;'&gt; &lt;center&gt; &lt;H1&gt; Sistema de Alarma&lt;/H1&gt; &lt;H2&gt; Sistema desarmado&lt;/H2&gt; &lt;/center&gt; &lt;form action='/alarma.cgi' method='get'&gt; &lt;center&gt; Clave: &lt;input type='text' name='clave1' value=''&gt; &lt;input type='submit' value='Armar sistema'&gt; &lt;/center&gt; &lt;/form&gt; &lt;/body&gt; &lt;/html&gt;</pre>	
<pre>&lt;!DOCTYPE html&gt; &lt;html&gt; &lt;head&gt; &lt;meta http-equiv='refresh' content='15'&gt; &lt;/head&gt; &lt;body style='background-color:lightyellow;'&gt; &lt;center&gt; &lt;H1&gt; Sistema de Alarma&lt;/H1&gt; &lt;H2&gt; Sistema armado&lt;/H2&gt; &lt;/center&gt; &lt;form action='/alarma.cgi' method='get'&gt; &lt;center&gt; Clave: &lt;input type='text' name='clave2' value=''&gt; &lt;input type='submit' value='Desarma sistema'&gt; &lt;/center&gt; &lt;/form&gt; &lt;/body&gt; &lt;/html&gt;</pre>	
<pre>&lt;!DOCTYPE html&gt; &lt;html&gt; &lt;head&gt; &lt;meta http-equiv='refresh' content='15'&gt;</pre>	

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70

</html>

```

t <!DOCTYPE html>
t <html>
t <head>
t <meta http-equiv='refresh' content='15'>
t </head>
c a 1 <body style='background-color:%s;'>
t <center>
t <H1> Sistema de Alarma</H1>
c a 2 <H2>%s</H2>
t </center>
t <form action='/alarma.cgi' method='get'>
c a 3 <center> Clave: <input type='text' name='clave%d' value=''>
c a 4 <input type='submit' value='%s'>
t </center>
t </form>
t</body>
t</html>

```

```

U16 cgi_func (U8 *env, U8 *buf, U16 buflen, U32 *pcgi) {
    U32 len = 0;

    switch (env[0]) {
        case 'a' :
            switch (env[2]) {
                case '1':
                    len = sprintf((char *)buf,(const char *)&env[4],color[estado]);
                    break;
                case '2':
                    len = sprintf((char *)buf,(const char *)&env[4],info[estado]);
                    break;
                case '3':
                    len = sprintf((char *)buf,(const char *)&env[4],estado + 1);
                    break;
                case '4':
                    len = sprintf((char *)buf,(const char *)&env[4],boton[estado]);
                    break;
                default:
                    break;
            }
            break;
        default:
            break;
    }
    return ((U16)len);
}

```

```

void cgi_process_var (U8 *qs) {
    U8 *var;
    char passwd[10];

    var = (U8 *)alloc_mem (40);
    do {
        qs = http_get_var (qs, var, 40);
        if (var[0] != 0) {
            if (str_scomp (var, "clave1=") == __TRUE) {
                if (str_scomp (&var[7], password) == __TRUE)
                    estado = 1;
            }
        }
    } while (1);
}

```



Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70

## CUESTIÓN 9.

Responda a las siguientes preguntas:

a) La característica más importante de un DSP es que tiene instrucciones MAC. ¿Qué son? ¿Para qué se utilizan?

b) ¿En qué se diferencia un DSP de un microcontrolador?

c) ¿En qué se diferencia un DSP de un SoC?

## CUESTIÓN 10.

¿Qué ventajas e inconvenientes tiene utilizar un Sistema Operativo en Tiempo Real en un microcontrolador en un proyecto similar a la práctica de laboratorio?

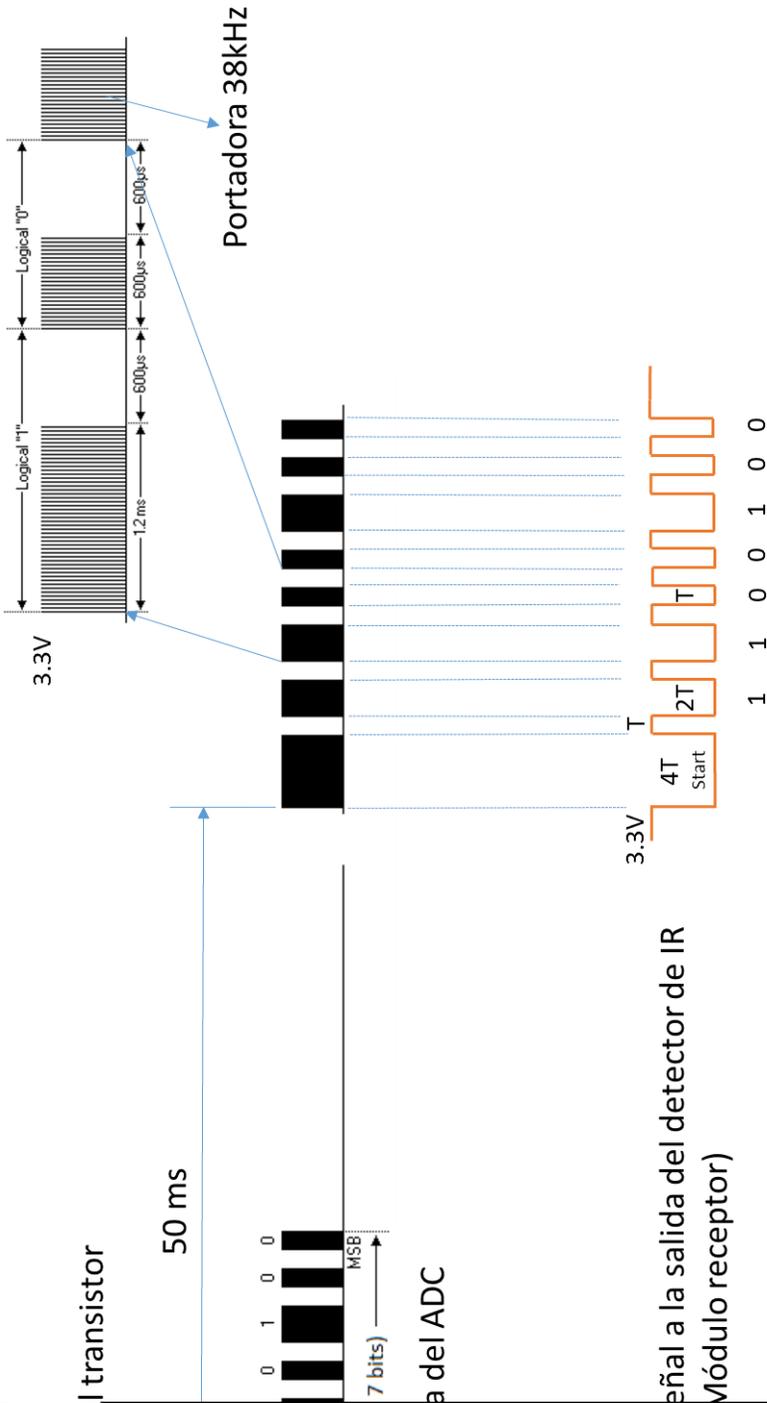
The logo for Cartagena99 features the text 'Cartagena99' in a stylized, blue, serif font. The '99' is significantly larger and more prominent than the rest of the text. The logo is set against a light blue background with a white, curved shape behind the text, and a yellow and orange gradient bar at the bottom.

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70

### Anexo 1 (Codificación – Decodificación IR)



transistor

50 ms

0 1 0 0  
7 bits) MSB

del ADC

señal a la salida del detector de IR  
Módulo receptor)

**CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70**

...

**ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70**



## Anexo 2. Funciones de control del Puerto Serie (UART0) y del GLCD

```
void UART0_IRQHandler(void) {
    switch(LPC_UART0->IIR&0x0E) {
        case 0x04: /* RBR, Receiver Buffer Ready */
            *ptr_rx=LPC_UART0->RBR; /* lee el dato recibido y lo almacena */
            if(*ptr_rx++ ==13){ /* Caracter return --> Cadena completa */
                *ptr_rx=0; /* Añadimos el caracter null para tratar los datos recibidos como una cadena*/
                rx_completa = 1; /* rx completa */
                ptr_rx=buffer; /* puntero al inicio del buffer para nueva recepción */
            }
            break;
        case 0x02: /* THRE, Transmit Holding Register empty */
            if(*ptr_tx!=0)
                LPC_UART0->THR = *ptr_tx++; /* carga un nuevo dato para ser transmitido */
            else
                tx_completa=1;
            break;
    }
}
```

```
void tx_cadena_UART0(char *cadena)
{
    ptr_tx=cadena;
    tx_completa=0;
    LPC_UART0->THR = *ptr_tx; // IMPORTANTE: Introducir un carácter al comienzo para iniciar TX o
} // activar flag interrupción por registro transmisor vacío
```

Table 285: UARTn Fractional Divider Register (U0FDR - address 0x4000 C028, U2FDR - 0x4009 8028, U3FDR - 0x4009 C028) bit description

Bit	Function	Value	Description	Reset value
3:0	DIVADDVAL	0	Baud-rate generation pre-scaler divisor value. If this field is 0, fractional baud-rate generator will not impact the UARTn baudrate.	0
7:4	MULVAL	1	Baud-rate pre-scaler multiplier value. This field must be greater or equal 1 for UARTn to operate properly, regardless of whether the fractional baud-rate generator is used or not.	1
31:8	-	NA	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0

$$UARTn_{baudrate} = \frac{PCLK}{16 \times (256 \times UnDLM + UnDLL) \times \left(1 + \frac{DivAddVal}{MulVal}\right)}$$

### Funciones del GLCD

```
/* Private function prototypes -----*/
```

**CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70**

---

**ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70**

**Cartagena99**

### Anexo 3. Características del AD5242

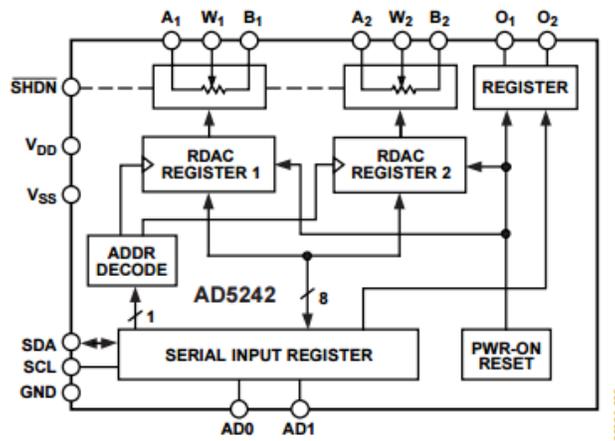


Figure 2. AD5242 Functional Block Diagram

Table 2.

S	0	1	0	1	1	AD1	AD0	R/W	A	A/B	RS	SD	O <sub>1</sub>	O <sub>2</sub>	X	X	X	A	D7	D6	D5	D4	D3	D2	D1	D0	A	P
Slave Address Byte									Instruction Byte									Data Byte										

where:

S = start condition

P = stop condition

A = acknowledge

X = don't care

AD1, AD0 = Package pin programmable address bits. Must be matched with the logic states at Pin AD1 and Pin AD0.

R/W = Read enable at high and output to SDA. Write enable at low.

A/B = RDAC subaddress select; 0 for RDAC1 and 1 for RDAC2.

RS = Midscale reset, active high.

SD = Shutdown in active high. Same as SHDN except inverse logic.

O<sub>1</sub>, O<sub>2</sub> = Output logic pin latched values

D7, D6, D5, D4, D3, D2, D1, D0 = data bits.

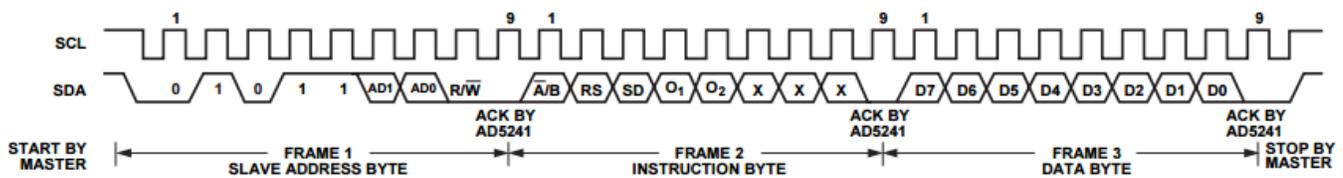
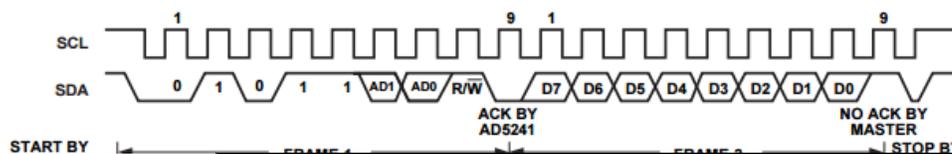


Figure 4. Writing to the RDAC Serial Register



CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE  
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

---

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS  
CALL OR WHATSAPP:689 45 44 70

