TÉCNICAS DIGITALES 2

Algunos finales resueltos

24 DE DICIEMBRE DE 2014

lianjosho

Índice de contenido

Nota de introducción: en este compilado resolví algunos de los finales que encontré dando vueltas en el grupo del 5R1; digo algunos porque muchos se repiten, y otros no los pude resolver, por ejemplo el de hacer un registro de desplazamiento, o el de guardar la hora en formato hh.mm.ss, como no pude ni simularlos siquiera no los puse directamente. Lo que sí pude simular son los ejercicios de C y ASM, pero solo de los que me parecieron medios complicados o de los que no encontré resolución en las fotocopias. Los demás ejercicios que vienen saliendo son sencillos (sacar promedio, sumar los elementos de un vector, y cosas por el estilo) así que no los puse tampoco, además hay varios ejercicios resueltos en las fotocopias y de distinta forma; yo estudié directamente de ahí y creo que con eso basta para ir bien preparado. Los circuitos «reloj1», «adap2», etc. son sub-circuitos, y los puse al final del apunte porque los uso en varios ejercicios; es para no andar repitiendo dibujos.

Como el Word no permite la actualización automática de las referencias (Figura 1b, Tabla2, etc.) a medida que agrego o elimino gráficos, es posible que algunas referencias estén mal o que haya errores. También es posible que haya errores en los valores de los esquemáticos, pues en muchos casos copié y pegué el circuito y le cambié los valores de las resistencias. **Así que toma con pinzas todo este apunte, pues está hecho un poco a las apuradas.** De todos modos, el concepto es el mismo, así que para sacarte las dudas tenés que hacer el circuito en Proteus y simularlo.

Los profes en el final son piolas y no te la complican. Eso es todo, suerte en el final.

Finales de 2015 (2 finales)

19 de febrero

A.- Se desea medir hasta la séptima armónica de una señal monofásica de red doméstica.

A1.- Frecuencia de muestreo necesaria. Justifique.

A2.- Circuito de muestreo para operar por interrupciones. Si ha de usar S/H establecer el tiempo de muestreo y el de retención. Especificar el tipo de interrupción utilizada.

A3.- Diseño del circuito para entrar al ADC con una tensión alterna **350V** pico respecto al neutro. Establecer tolerancia de los elementos pasivos para ε_T**=2%**.

B.- Con un transductor cuya función de transferencia es **0,4µA/0,1ºK** se desea medir desde **–10** a **50°C**.

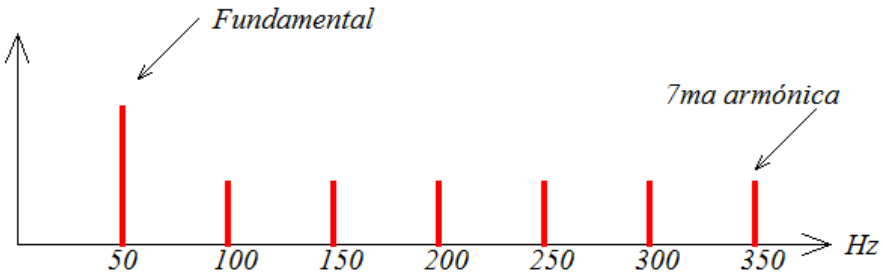
B1.- Con una resolución de **0,1°C,** ¿puede usarse el ADC del arm?, ¿y con una resolución de **0,01ºC**?

B2.- Realizar el circuito de adaptación, establecer tolerancia de elementos pasivos para ε_T**=1%**.

B3.- Las lecturas del ADC se guardan en direcciones consecutivas de memoria. La posición de inicio está en **R2**, y la posición actual está en el **R4**. Se debe escribir la rutina de assembler para calcular el promedio de las últimas 4 lecturas.

Punto A:

Punto A1:



La máxima frecuencia de la señal a muestrear es de **350Hz**, y como se recomienda usar una frecuencia de muestreo de al menos 5 veces la frecuencia máxima, tengo:

$$f_{sample} = f_s = 5 \cdot f_{max} = 5 \cdot 350 = \boxed{1750Hz = f_s}$$

Punto A2:

Los tiempos de muestreo y de retención son:

$$T_{muestreo} = 0,1 \cdot T_{min} = 0,1 \cdot 571,4\mu = 57,1\left[\mu s\right] \qquad T_{retención} = 0,9 \cdot T_{min} = 0,9 \cdot 571,4\mu = 513,9\left[\mu s\right]$$

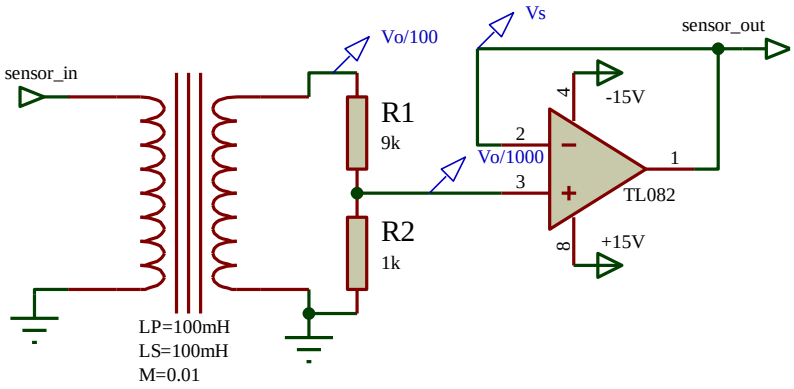
- $T_{min} = \dfrac{1}{f_{max}} = \dfrac{1}{1750} = 571,4\left[\mu s\right]$

A la resistencia R_A de «reloj1» la calculo con **f=1750Hz**, **C=10nF** y R_B**=6,5kΩ**:

$$R_A = \frac{1,44}{f \cdot C} - 2 \cdot R_B = \frac{1,44}{1750 \cdot 10n} - 2 \cdot 6,8k = 68685 \approx 68\left[k\Omega\right]$$

Punto A3:

El circuito del sensor para reducir el voltaje de entrada que uso es:



El divisor resistivo seguido del operacional configurado como seguidor de tensión funciona por lo menos hasta una división de **100** de la señal, alguna vez lo probé y creo que sirve para trabajar con frecuencias de hasta 1,**75kHz** (que es la calculada en el punto A1). No me animé a dividirla por **1000** porque nunca lo probé, así que por eso puse un transformador. **LP** y **LS** son la inductancia del primario y del secundario respectivamente, y **M** es el factor de acoplamiento. Con **M=0,01** el transformador reduce el voltaje 100 veces, **LP=LS=100mH** porque esos valores no producen ni desfasaje ni atenuación del secundario con relación al primario (no sé por qué). La función de transferencia de este sensor es $\mathbf{G_{sensor}=1/1000=0,001V/V}$.

Los parámetros de la Figura 12 (debajo de dicha figura explico cada uno de las nomenclaturas) valen:

$$V_{FE,o} = 700\,[V]$$

$$m_o = m_s = m = 2^n = 2^{10} = 1024$$

$$V_{LSB,o} = \frac{V_{FE,o}}{m_o} = \frac{700}{1024} = 683{,}6\,[mV]$$

$$v_o = V_{p,o} \cdot \sin\omega t + V_{OS,o} = (350 \cdot \sin\omega t)\,[V]$$

$$V_{FE,s} = V_{FE,o} \cdot G_{sensor} = 700 \cdot 0{,}001 = 700\,[mV]$$

$$V_{LSB,s} = V_{LSB,o} \cdot G_{sensor} = 683{,}6m \cdot 0{,}001 = 683{,}6\,[\mu V]$$

$$v_s = V_{p,s} \cdot \sin\omega t = (350 \cdot \sin\omega t)\,[mV]$$

- $V_{p,s} = V_{p,o} \cdot G_{sensor} = 350 \cdot 0{,}001 = 350\,[mV]$

$$V_{LSB} = \frac{V_R}{2^n} = \frac{3{,}3}{2^{10}} = 3{,}22\,[mV]$$

$$V_{FE} = V_R = 3{,}3\,[V]$$

$$v_{adap} = V_{p,adap} \cdot \sin\omega t + V_{OS,adap} = (1{,}65 \cdot \sin\omega t + 1{,}65)\,[V]$$

- $V_{p,adap} = V_{p,s} \cdot G_{adap} = 350m \cdot 4{,}71 = 1{,}65\,[V]$
  - $G_{adap} = \frac{V_{LSB}}{V_{LSB,s}} = \frac{3{,}22m}{683{,}6\,\mu} = \boxed{4{,}71 = G_{adap}}$
- $V_{OS,adap} = 1{,}65\,[V]$

Todos estos parámetros se ven en la Figura 1. Como $\mathbf{1<G_{adap}<10}$ podría usar «adap1» pero como tengo que meter un offset uso «adap2». Las resistencias de este circuito (debo conectar también $\mathbf{R_5}$ a **+15V**) son:

$$R_1 = R_2 = R_3 = 1\,[k\Omega] \qquad V_{ref} = +15\,[V]$$

$$R_4 = R_3 \cdot G_{adap} = 1k \cdot 4{,}71 = 4{,}71\,[k\Omega] \qquad R_5 = \frac{V_{ref} \cdot R_4}{V_{OS,adap}} = \frac{15 \cdot 4{,}71k}{1{,}65} = 42{,}8\,[k\Omega]$$

Para calcular la tolerancia de los componentes pasivos uso la siguiente ecuación del error máximo total.

$$\varepsilon_T = \varepsilon_{ADC} + \varepsilon_{riple} + \varepsilon_G$$

Donde $\boldsymbol{\varepsilon_T}$ es el error máximo total y está referido a la tensión de referencia, es un porcentaje de $\mathbf{V_R}$; $\boldsymbol{\varepsilon_{ADC}}$ es el error máximo del ADC y es igual a ½ de $\mathbf{V_{LSB}}$; $\boldsymbol{\varepsilon_{riple}}$ es el error máximo provocado por el riple de la fuente de referencia, es igual a ¼ de $\mathbf{V_{LSB}}$; y $\boldsymbol{\varepsilon_G}$ es el error máximo provocado por la variación de la ganancia

del adaptador, la cual depende a su vez de los componentes pasivos, es igual a ¼ de $\mathbf{V_{LSB}}$, pero como es lo que tengo que calcular, no le asigno este valor sino que lo despejo de la ecuación de arriba.

Nota: en algunas fotocopias de finales resueltos vi que se hace $\varepsilon_{riple}=\varepsilon_G$ y se la pone en función de ε_T y ε_{ADC} solamente. Este es el criterio que usé en el final y me corrigieron bien.

Por último, calculo las tolerancias de los componentes pasivos, $E_{G,VR}$ y $E_{G,VFE}$, que son iguales en caso de que $V_{FE}=V_R$. Resumiendo, tengo:

$$E_{G,VR}=E_{G,VFE}=\frac{\varepsilon_G}{V_R}=\frac{32{,}2m}{3{,}3}=9{,}75m=\boxed{9750\,[ppm]=E_{G,VR}}$$

$$\bullet\ \varepsilon_G=\frac{\varepsilon_T-\varepsilon_{ADC}}{2}=\frac{0{,}02\cdot V_R-V_{LSB}/2}{2}=\frac{0{,}02\cdot 3{,}3-3{,}22m/2}{2}\approx 32{,}2\,[mV]$$

En caso de que $V_{FE}\neq V_R$, entonces tengo que hacer $E_{G,VFE}=\varepsilon_G/V_{FE}$. Es la única vez que hago éste cálculo pues en todos los casos es lo mismo, solo difiere en si V_{FE} es igual o no a V_R.

La siguiente figura muestra el circuito completo y las distintas señales.

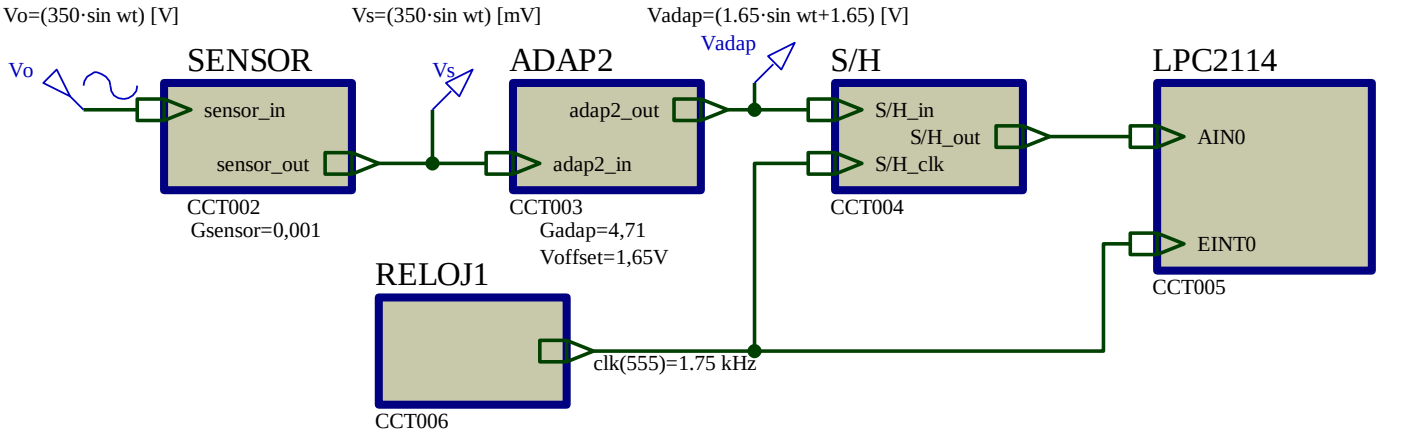


*Figura 1: Circuito completo y señales involucradas, donde v_{adap} va arranca de **0**.*

Punto B:

Punto B1:

$$n_1=\frac{\ln(m_{s1})}{\ln 2}=\frac{\ln 600}{\ln 2}=9{,}23\approx 10\,[bits]$$

$$\bullet\ m_{s1}=\frac{V_{FE,o}}{V_{LSB,o1}}=\frac{V_{o,max}-V_{o,min}}{V_{LSB,o1}}=\frac{325{,}16-263{,}16}{0{,}1}=600$$

$$\bullet\bullet\ V_{o,max}=50^oC+273{,}16=325{,}16\,[^oK]$$

$$\bullet\bullet\ V_{o,min}=-10^oC+273{,}16=263{,}16\,[^oK]$$

$$n_2=\frac{\ln(m_{s2})}{\ln 2}=\frac{\ln 6000}{\ln 2}=12{,}55\approx 13\,[bits]$$

$$\bullet\ m_{s2}=\frac{V_{FE,o}}{V_{LSB,o2}}=\frac{325{,}16-263{,}16}{0{,}01}=6000$$

En el primer caso puedo usar el ADC del ARM, pues éste tiene una resolución de hasta **10 bits**; en cambio en el segundo caso no es posible ya que $\mathbf{n_2 > 10}$.

Punto B2:

Este punto se resuelve igual al punto B del final del 7 de febrero de 2013, solo hay que tener en cuenta que el rango de temperaturas en **ºC** es distinto.

5 de febrero

Para un sistema basado en ARM7:

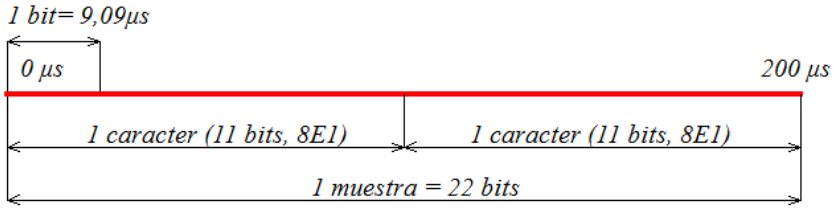
A.- Se tiene que muestrear con el ADC del ARM una señal de **1kHz**, cada dato se debe enviar (usando dos caracteres) por una interfase RS232-C usando paridad par. ¿Cuál es la velocidad mínima, bajo norma, que permite realizar la comunicación?

B.- Escriba una rutina en assembler que lea 20 valores de 32 bits desde un vector que debe ser direccionado usando a **R4** como puntero. Calcule la suma de todos los valores leídos y su promedio (implementar la división en una función).

C.- A un ADC externo de 12 bits de $\mathbf{V_R=3V}$ se debe conectar un puente de termistores con una salida de **0,016mV/0,01ºC**. Se pide diseñar la etapa amplificadora y establecer las tolerancias de los componentes para un error de **0,1%**.

Punto A:

Como la señal a muestrear es de **1kHz**, la frecuencia de muestreo debe ser mínimo 5 veces mayor: **5kHz**. El dato de una muestra debe enviarse antes de realizar la próxima, o sea antes de que pasen **1/5kHz=200µs**. Como se deben enviar dos caracteres, uso 2 cadenas de bits 8E1 (asumo que tiene 1 bit de stop), es decir, 22 bits, **T** vale entonces **T=200µ/22=9,09µs**, lo que significa que la velocidad en bits por segundo será **bi-trate=1/T=1/9,09µ≈110011 bps**. Y como un carácter tiene 8 bits, la velocidad de transmisión en baudios será **Baud Rate=bitrate/8=110011/8=13751 baudios**. El valor próximo normalizado por encima es **19200**.



Velocidad de transmisión normalizada, en baudios

75	110	150
300	600	1200
2400	4800	9600
19200		

Punto B:

Función «main.c»

```
#include <LPC21xx.h>
int valores20(void);

int main (void)
      {valores20();
      while (1);
      return 0;}
```

Función «valores20.s», carga en R1 la suma de los elementos del vector

```
valores20:   mov r4,#vector
             add r4,r4,#0x124
             mov r1,#0
             mov r2,#0x14 /*(20 en hexa)*/
carga:       ldr r0,[r4],#4
             add r1,r1,r0
             subs r2,r2,#1
             bne carga
b promedio
vector:      .word 20,100,14,52,45,56,55,55,78,96,51,11,12,14,15,16,17,18,19,20
bx lr
```

Función «promedio.s», divide por 20 el valor guardado en R1 y lo guarda en R4.

```
promedio:    mov r0,#0x0
             add r0,r0,#0xCCC00000
             add r0,r0,#0x000CC000
```

```
            add r0,r0,#0x00000CC0
            add r0,r0,#0x0000000D

            umull  r0,r2,r1,r0
            mov r4,r2,lsr #4
bx lr
```

El operador MOV no permite mover de forma inmediata el valor 0xCCCCCCCD, por lo que debo hacerlo por partes en 5 instrucciones.

El programa en general tiene la siguiente lógica:

```
      mov r0,#A
      umull  r0,r2,r1,r0
      mov r4,r2,lsr #B
```

Suponga que se quiere hacer **840** dividido en **43**. El dividendo se guarda en **R1**, así que **R1=0x348**. La constante **A** sale de hacer $\mathbf{A=(2^{32+B})/N}$, donde **N** es el divisor, en este caso **43**. Para hallar **A**, lo que hago es ir dándole valores enteros a **B**, comenzando por **1** hasta encontrar el valor máximo tal que $\mathbf{A<2^{32}}$. En este caso **B=5**, ya que $\mathbf{A=(2^{32+5})/43=319'625'432\approx 0{,}3.10^{12}}$. Si **B=6**, entonces $\mathbf{A=(2^{32+6})/43=6'392'509'464\approx 6{,}3.10^{12}}$, y este valor es mayor que $\mathbf{2^{32}=4'294'967'296\approx 4{,}2.10^{12}}$. Así que **A=319625432=0x130D18D8**, y **B=5**.

Cuando finaliza la tercer instrucción, el resultado de **840/43** se guarda en **R4**. No me preguntes cuál es la lógica, saqué la rutina de internet, la probé y anda. Probala antes si tenés dudas, o preguntale al profe.

Punto C:

$$m = m_s = m_o = 2^n = 4096$$

$$V_{LSB} = \frac{V_R}{2^n} = \frac{3}{4096} = 732{,}4\,[\mu V]$$

$$V_{FE} = V_R = 3\,[V]$$

$$V_{LSB,o} = 0{,}01\,[^oC]$$

$$V_{FE,o} = V_{LSB,o} \cdot m_o = 0{,}01 \cdot 4096 = 40{,}96\,[^oC]$$

$$G_{sensor} = \frac{0{,}016\,[mV]}{0{,}01\,[^oC]} = 1{,}6\left[\frac{mV}{^oC}\right]$$

$$V_{FE,s} = V_{FE,o} \cdot G_{sensor} = 40{,}96 \cdot 1{,}6\,m = 65{,}5\,[mV]$$

$$V_{LSB,s} = V_{LSB,o} \cdot G_{sensor} = 0{,}01 \cdot 1{,}6\,m = 16\,[\mu V]$$

$$G_{adap} = \frac{V_{LSB}}{V_{LSB,s}} = \frac{732{,}4\,\mu}{16\,\mu} = 45{,}78$$

Como $\mathbf{10<G_{adap}<100}$ uso «adap2». $\mathbf{R_4}$ vale:

$$R_4 = 1000 \cdot G_{adap} = 1k \cdot 45{,}78 = 45{,}78\,[k\Omega]$$

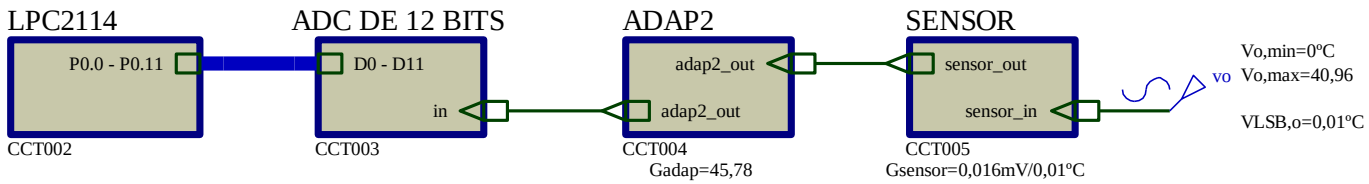


Figura 2: Diagrama de bloques del circuito completo.

Finales de 2014 (3 finales)

20 de noviembre

A.- (Igual al final del 31 de octubre de 2013.) Se tiene un sistema basado en ARM-7 que tiene tres lazos de comunicación por RS 232-C. Se debe implementar el canal N°3 de manera discreta. Las condiciones de trabajo son: Frame: 8 bits de datos, paridad par, un bit stop. **Bitrate: 900 bps**. Usted debe implementar el canal de recepción, que implica:

A1.- Diseño del reloj para el receptor de datos. Esto incluye:

• Especificación de la frecuencia de trabajo y la relación marca-espacio.

• Especificación básica del circuito de reloj (p. ej. basado en 555 o similar; usando dos inversores y un cristal; etc.) NO ES NECESARIO el circuito real.

A2.- Diseño del circuito de sincronización de los datos de recepción, completo. Debe incluir la adaptación de nivel RS 232 a lógica usada.

A3.- Diseño del circuito de pasaje serie-paralelo, incluyendo el detector de paridad.

A4.- Diagrama de bloques de la operación, con el uso de interrupciones.

B.- (Igual al final del 28 de febrero de 2013.) Se desea medir la cantidad de agua en un tanque y para ello se lo pesa. El peso del tanque vacío es de **1780kg** y puede contener hasta **8m^3** de agua. Se pesa el conjunto con una galga extensiométrica y la salida de esta es **0,4µV/kg**. Se pide:

B1.- Para una resolución de **1 litro**, ¿se puede usar el ADC del ARM?, ¿ y para 10 litros? Justificar.

B2.- Se requiere un error de **1%** y resolución de **10 litros**. Diseñe el circuito de adaptación para una referencia de **3V**. Especifique las tolerancias de los componentes pasivos y de la referencia.

B3.- Diseñe lo necesario para que el sistema ARM usado opere por interrupción tomando una muestra cada **10 segundos**. Describa como ha de configurarse el controlador de interrupciones.

B4.- Escriba la rutina en Assembler del ARM que convierta el peso del tanque en LITROS de agua. ATENCIÓN: litros y no litros x 10.

B5.- Dibuje el circuito de control de un display de diodos de 7 segmentos que indique el contenido de agua del tanque en litros, a partir del ARM y hardware externo. Este display NO estará multiplexado.

Punto A:

Punto A1:

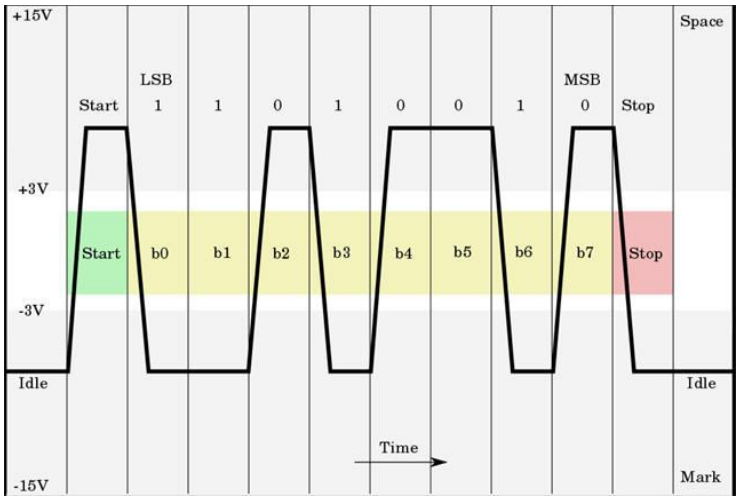
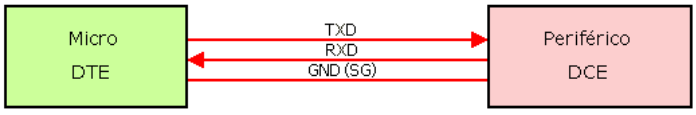
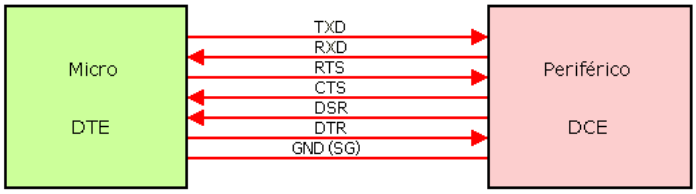


Figura 3: Trama RS232 para el caracter ASCII «K».

Para una comunicación full duplex desde la UART de un microprocesador o microcontrolador deben conectarse un mínimo número de señales, concretamente TXD y RXD así como la masa (GND, SG o Signal Ground), aunque una interfaz típica RS232 requiere al meme 7 señales. Pero aquí usaremos la conexión mínima (3 conexiones).



Requerimientos de señal para mínima conexion full duplex

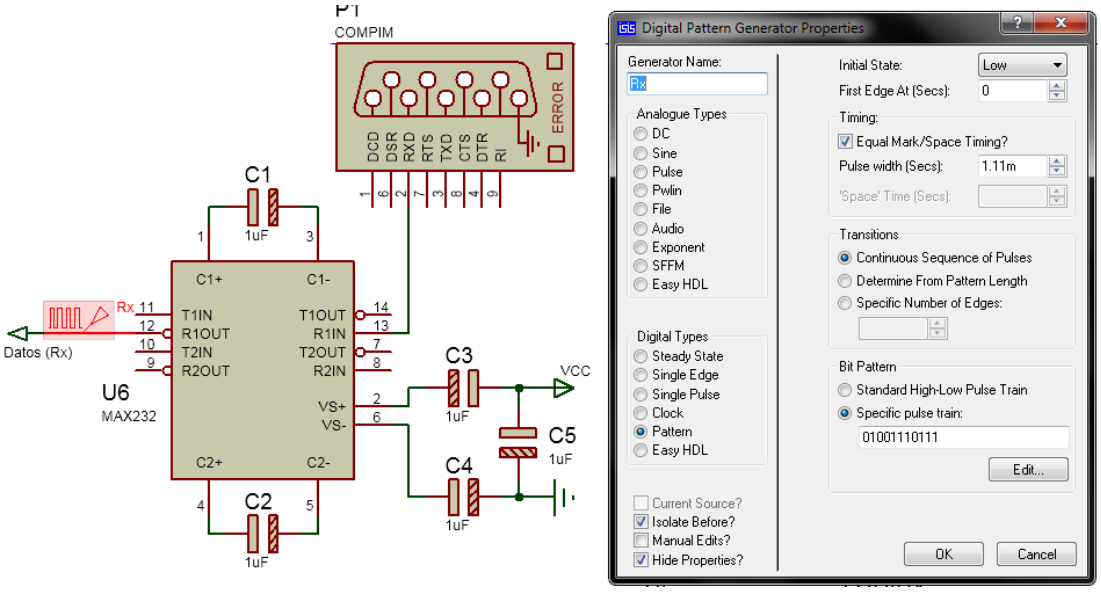


Requerimientos de señal típica para comunicacion full duplex

Las líneas adicionales se utilizan para la puesta de acuerdo entre el DTE (por ejemplo un PC) y el DCE (por ejemplo un ratón). El terminal para transmitir datos (TXD) es utilizado para transferir datos del DTE al DCE, por lo que debe ser conectado a la línea receptora serie del periférico. De manera idéntica la línea receptora de datos (RXD) debe ser conectada a la línea transmisora del periférico.

Punto A2:

El bloque de recepción comienza con el esquemático de la entrada: una ficha DB9, que en el proteus se llama «COMPIM». Del pin 2, RXD, sale la señal hacia el MAX232, que es un dispositivo que adapta las señales RS232 a niveles lógicos TTL/CMOS. Del pin 12 del MAX232 sale la señal adaptada (y además invertida) y esa es la señal «Datos(Rx)». Esta señal la simulamos con el proteus con un generador externo, en el cual además activamos la opción «Isolate Before?» que hace que toda señal que venga de detrás no se tenga en cuenta. En este caso el patrón generado es 01001110111, donde los bits subrayados son los de datos.

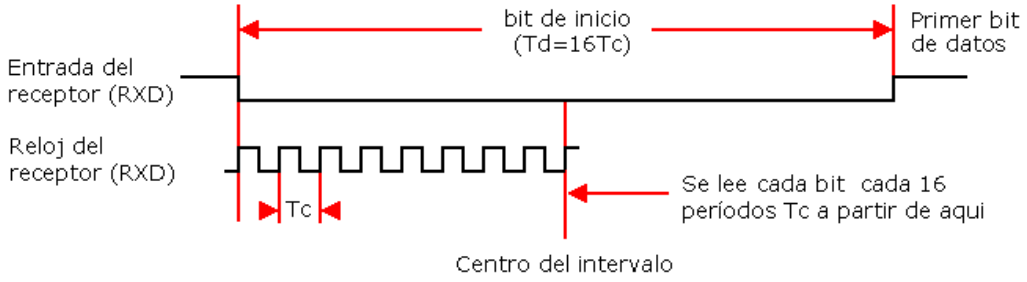


El bit de start está en bajo, al contrario de lo que se ve en la Figura 3, porque el integrado MAX232, además de adaptar los niveles de RS232 a TTL, invierte las señales. Por este motivo también los bits de paridad y de stop están así. **Nota:** el bit de paridad es un 1 porque lo hice al azar, pero el ejercicio requiere que este bit sea 0, pues hay paridad par. Me di cuenta recién cuando estaba armando el .doc.

Para realizar la sincronización del dato recibido debemos comprobar el bit en la mitad del intervalo del tiempo que dura para evitar la lectura de falsas transiciones producto del ruido en la línea; por esto se llama **sincronización asíncrona**. Para sincronizar uso un reloj externo de período $\mathbf{T_c}$ que cumple la relación:

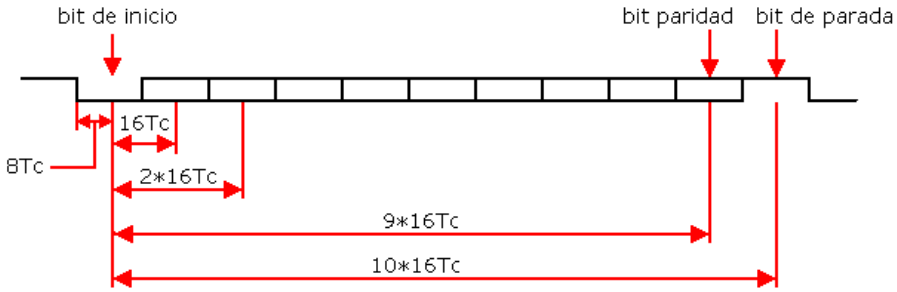
$$T_c = k \cdot T$$

Donde **T** es el tiempo de duración de 1 bit, es decir **T=1/f=1/900=1,11ms**; y **k** es una constante, generalmente de valor 16. Para lograr la sincronización entre el transmisor y el receptor tanto $\mathbf{T_c}$ como **k** deben ser el mismo para ambos, ello permitirá que el bit de datos se compruebe en el momento preciso sin necesidad de conectar una línea adicional de reloj para lograr el sincronismo. En la figura siguiente vemos cómo después de detectado el bit de inicio y transcurridas 8 transiciones de reloj, ha transcurrido un tiempo igual a la mitad del bit de información que establece el inicio de la recepción de un nuevo caracter. A partir de ese tiempo se leerán los datos cada 16 pulsos de reloj.



Deteccion del bit de inicio.

Esto permite comprobar la información en la mitad del intervalo de cada bit de información.



Comprobación de la informacion en la mitad del intervalo de cada bit

Todo esto se logra con el siguiente bloque, en donde hice que **k=4** para que se pueda ver mejor el gráfico. El bloque «LOGICA» no es más que un contador que pone en alto la señal «reset(11)» a los 11 pulsos de la señal «f*4/4»; ídem para «reset(9)», que es usada en otro bloque más adelante.

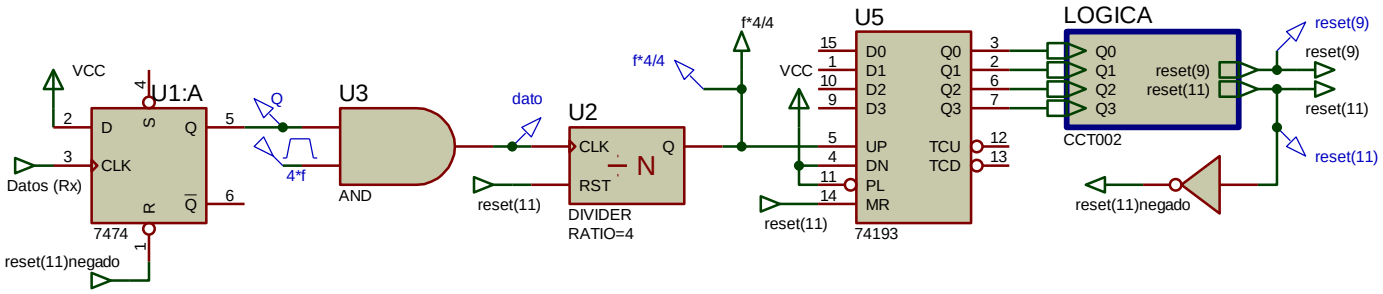


Figura 4: Bloque de sincronismo.

Punto A3:

El bloque siguiente es el que lee los datos. A través de un shift register, convierto los datos series en paralelo. El clock de este dispositivo (74164) está conectado a la señal «f*4/4», con esto me aseguro que el dato sea leído a la mitad de transcurrido el tiempo de 1 bit. A la salida del shift register va un latch (74273) que retiene los datos y los envía al ARM (D0-D7). La frecuencia de esta señal, «clk(555)», la puedo sacar de un 555, y como el ejercicio no pide el circuito completo, lo que hago es simularla con un generador de pulsos. La frecuencia a la que trabaja es igual a 11 veces la duración de 1 bit, es decir que manda datos al ARM cuando se completa una trama; este valor es $\mathbf{11 \cdot f_{1\ bit} = 11 \cdot 1/T = 11 \cdot 1/1{,}11m \approx 81{,}81\ [Hz]}$. La duración de este pulso no debe ser mucha, ya que los datos cambian a una frecuencia **f=900 [Hz]**; yo le puse que el ciclo de trabajo es igual a un **2%**.

Nota: la señal «reset(11)negado» no llega al master reset (pin 9) del shift register, no sé si será porque tiene corta duración o qué, la cuestión es que no me complico y le pongo un generador externo.

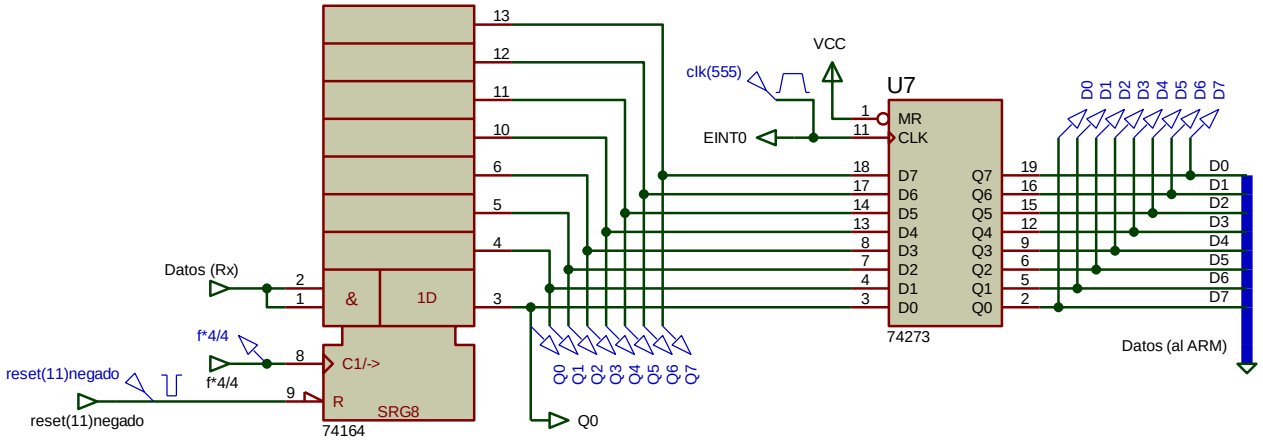


Figura 5: bloque de conversión y envío de datos

Por último, para verificar la paridad, uso este último bloque, que utiliza un 74180. Este circuito permite comparar el bit 10 de la trama de datos (que es el bit de paridad, el cual está almacenado en la salida Q0 del shift register) con los bits generados por el 74180, para determinar si la paridad es la correcta o si echamos moco. Esto se debe hacer en el tiempo en que todavía Q0 pasa del último bit de información al bit de paridad, lo cual sucede en el noveno pulso de la señal de la señal «f*4/4», por eso es necesaria la señal «reset(9)» salida del bloque «LOGICA». En realidad lo que hice es ver qué señal o combinación de señales podía usar para habilitar el clock del flip-flop durante ese tiempo, y vi que todo ocurría cuando la señal «f*4/4» tira el noveno pulso (podría haber usado otras siempre y cuando esa combinación de estados se de en ese intervalo de tiempo nada más), así que aproveché el contador de 11 pulsos y mediante lógica combinacional obtuve «reset(9)».

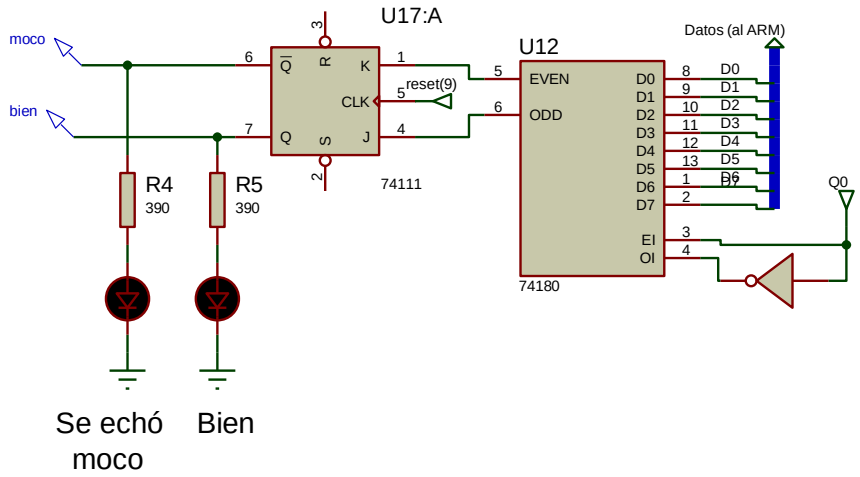


Figura 6: Comprobación de la paridad.

Punto A4:

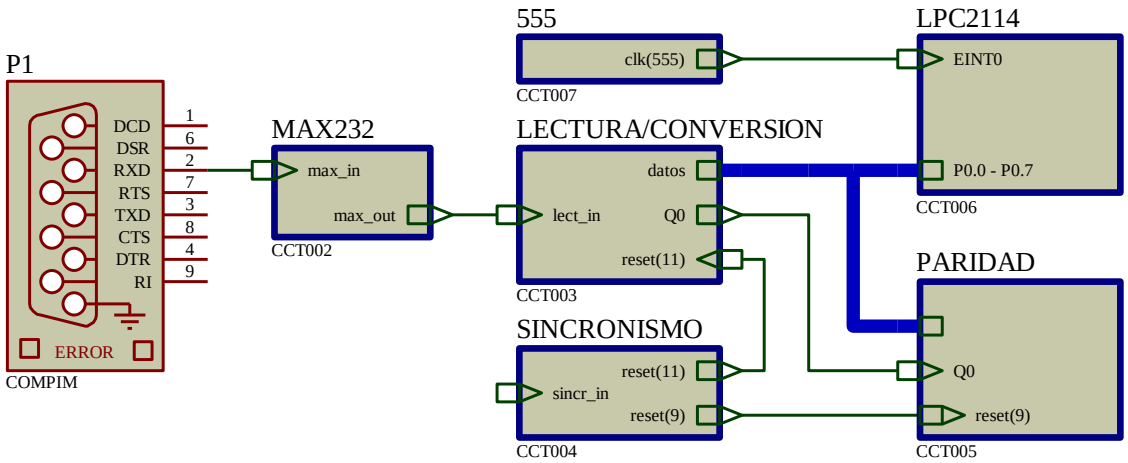
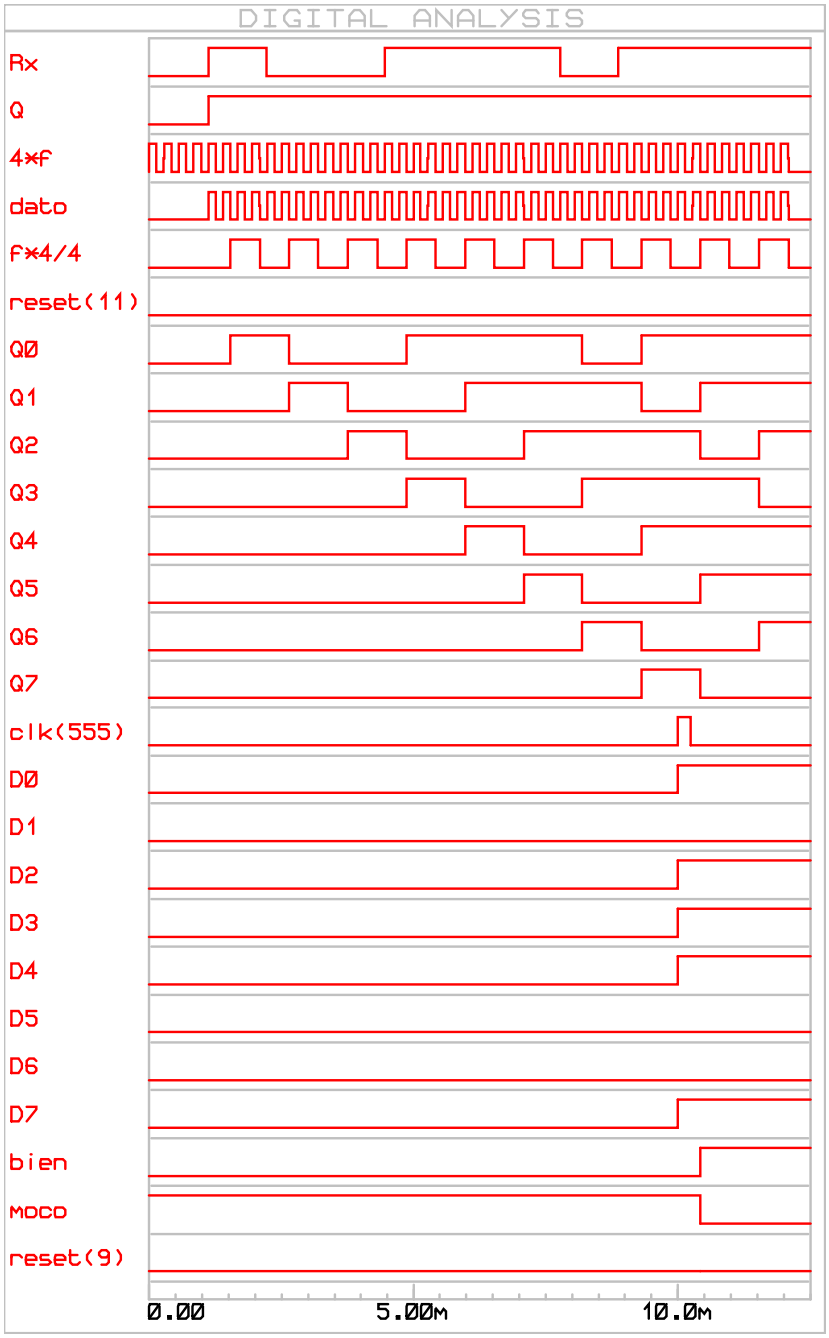


Figura 7: Diagrama de bloques completo del circuito de recepción de datos.

En el siguiente gráfico se ven las señales en el tiempo.



Punto B:

Punto B1:

1 [m^3] de agua pesa 1000 kilogramos masa [kg] , y 1 [litro]=1 [dm^3]. Con estos datos tengo:

$$V_{o,min} = 0\left[kg\right]$$

$$V_{o(tanque)} = 1780\left[kg\right]$$

$$V_{o,(agua)} == 8\left[m^3\right]\cdot\frac{1000\left[kg\right]}{1\left[m^3\right]} = 8000\left[kg\right]$$

$$V_{o,max} = V_{o(tanque)} + V_{o,(agua)} = 1780 + 8000 = 9780\left[kg\right]$$

$$V_{LSB,o(1\,litro)} = 1\left[litro\right]\cdot\frac{1\left[dm^3\right]}{1\left[litro\right]}\cdot\frac{0{,}001\left[m^3\right]}{1\left[dm^3\right]}\cdot\frac{1000\left[kg\right]}{1\left[m^3\right]} = 1\left[kg\right]$$

$$V_{LSB,o(10\,litros)} = 10\left[litros\right]\cdot\frac{1\left[dm^3\right]}{1\left[litro\right]}\cdot\frac{0{,}001\left[m^3\right]}{1\left[dm^3\right]}\cdot\frac{1000\left[kg\right]}{1\left[m^3\right]} = 10\left[kg\right]$$

Resolución de 1 [litro]

$$n = \frac{ln(m_s)}{ln(2)} = \frac{ln(8000)}{ln(2)} = 12,96 \approx \boxed{13\,[bits] = n}$$

- $m_s = \frac{V_{FE,o}}{V_{LSB,o(1\,litro)}} = \frac{8000}{1} = 8000$
  - $V_{FE,o} = V_{o,max} - V_{o,(tanque)} = 9780 - 1780 = 8000\,[kg]$

El fondo de escala $\mathbf{V_{FE,o}}$ tiene ese valor porque la entrada no tiene su mínimo en **0kg** sino cuando no hay agua; ahí la señal primaria será $\mathbf{v_o=1780kg}$, que es el peso del tanque vacío.

Resolución de 10 [litros]

$$n = \frac{ln(m_s)}{ln(2)} = \frac{ln(800)}{ln(2)} = 9,64 \approx \boxed{10\,[bits] = n}$$

- $m_s = \frac{V_{FE,o}}{V_{LSB,o(10\,litros)}} = \frac{8000}{10} = 800$

El número máximo de bits en el ADC del LPC2114 es 10. Por tanto, no es posible el primer caso.

Punto B2:

Tomo el valor de resolución como de **10 litros**, es decir $\mathbf{V_{LSB,o}=10kg}$. Con esto, los parámetros son:

$V_{FE,o} = 8000\,[kg]$

$V_{LSB,o} = 10\,[kg]$

$m_o = \frac{V_{FE,o}}{V_{LSB,o}} = \frac{8000}{10} = 800$

$V_{FE,s} = V_{FE,o} \cdot G_{sensor} = 8000 \cdot 0,4\mu = 3,2\,[mV]$

$V_{LSB,s} = V_{LSB,o} \cdot G_{sensor} = 10 \cdot 0,4\mu = 4\,[\mu V]$

$m_s = m_o = 800$

$G_{adap} = \frac{V_{LSB}}{V_{LSB,s}} = \frac{2,93m}{4\mu} \approx 736$

- $V_{LSB} = \frac{V_R}{m} = \frac{3}{1024} = 2,93\,[mV]$
  - $m = 2^n = 2^{10} = 1024$

$V_R = 3\,[V]$

$V_{FE} = V_{LSB} \cdot m_s = 2,93m \cdot 800 = 2,34\,[V]$

$V_{LSB} = 2,93\,[mV]$

$m = 1024$

Como **100<G<1000** uso un amplificador de instrumentación, «adap3». La señal no adaptada, $\mathbf{v_s}$, va desde el voltaje correspondiente al tanque vacío hasta el voltaje correspondiente al tanque lleno con agua, es decir:

$v_s = V_s \cdot sin(\omega t) + V_{OS} = (1,6 \cdot sin\,\omega t + 2,31)\,[mV]$

- $V_s = \frac{V_{s,max} + V_{s,(tanque)}}{2} = \frac{3,91m + 712\mu}{2} = 1,6\,[mV]$
  - $V_{s,(tanque)} = V_{o,(tanque)} \cdot G_{sensor} = 1780 \cdot 0,4\mu = 712\,[\mu V]$
  - $V_{s,max} = V_{o,max} \cdot G_{sensor} = 9780 \cdot 0,4\mu = 3,91\,[mV]$
- $V_{OS} = V_s + V_{s,min} = 1,6m + 712\mu \approx 2,31\,[mV]$

El valor de $\mathbf{R_g}$, tomando $\mathbf{R_1=100k\Omega}$, será:

$$G = 1 + \frac{2 \cdot R_1}{R_g} \quad \rightarrow \therefore R_g = \frac{2 \cdot R_1}{G - 1} = \frac{2 \cdot 100k}{736 - 1} = 272{,}1\left[\Omega\right]$$

La tolerancia de los componentes pasivos es:

$$E_{G,VR} = E_{G,VFE} = \frac{\varepsilon_G}{V_R} = \frac{14{,}3m}{3} = 4{,}76m = \boxed{4760\left[ppm\right] = E_{G,VR}}$$

$$\bullet\ \varepsilon_G = \frac{\varepsilon_T - \varepsilon_{ADC}}{2} = \frac{0{,}01 \cdot V_R - V_{LSB}/2}{2} = \frac{0{,}01 \cdot 3 - 2{,}93m/2}{2} \approx 14{,}3\left[mV\right]$$

Punto B3:

El valor de $\mathbf{R_A}$ del circuito «reloj1», tomando $\mathbf{f_{osc}=0{,}1Hz}$ (pues la interrupción debe ocurrir cada **10seg**), $\mathbf{C=10\mu F}$ y $\mathbf{R_B=680k\Omega}$, es:

$$R_A = \frac{1{,}44}{f_{osc} \cdot C} - 2R_B = \frac{1{,}44}{0{,}1 \cdot 10\mu} - 2 \cdot 680k = 80\left[k\Omega\right]$$

Configuración del VIC:

```
VICIntSelect = (0≪14)    //Canal 14 (EINT0) configurado como una interrupción IRQ.
VICIntEnable = (1≪14)    //Interrupción EINT0 activada.
VICVectCntl0 = (14≪0)    //Asignación de prioridad 0 al canal 14 (bits 4-0).
VICVectCntl0 |= (1≪5)    //Asignación a la categoría de vectorizado al canal 14 (bit 5).
EXTMODE = (1≪0)          //Interrupción activada por flanco (bit 0 para EINT0).
EXTPOLAR= (1≪0)          //Interrupción activada por flanco ascendente (bit 0 para EINT0).
```

Punto B4:

El peso mínimo que advierte la balanza es **1780kg**, con esto $\mathbf{V_{s,min}=712\mu V}$, que al pasar por la adaptación queda $\mathbf{v_{adap,min}=0{,}52V}$, y como necesito que arranque de cero debo conectar $\mathbf{R_9}$ en «adap3» a **+15V**. $\mathbf{R_9}$ valdrá:

$$R_9 = \frac{R_{10} \cdot V_{ref}}{V_{offset}} = \frac{1k \cdot 15}{0{,}52} \approx 28{,}8\left[k\Omega\right]$$

Esto me permite poner $\mathbf{v_{ADC,min}=0V}$. Este valor provoca que el ADC guarde el binario 0000.0000.0000.

El peso mínimo que puede mostrar la balanza después de **1780kg**, es **1790kg**, pues la resolución es de **10kg** (equivalentes a 10 litros), lo cual provoca una $\mathbf{Vs=716\mu V}$, que al pasar por la adaptación es $\mathbf{V_{ADC}=2{,}93mV}$. El ADC guardará entonces el binario 0000.0000.0001.

Así hasta completar los 800 escalones o pasos. Resumiendo, tengo:

Volumen [litros]	Peso [kg]	Binario ADC	Decimal	Hexa
0	1780	0000.0000.0000	0	0x0
10	1790	0000.0000.0001	1	0x1
20	1800	0000.0000.0010	2	0x2
…	…	…	…	…
8000	9780	0011.0010.0000	800	0x320

Suponiendo que el número quede guardado en R0, la rutina en assembler debe convertir ese valor en litros. Es decir, multiplicarlo por 10.

```
peso_en_litros:    r1=A;
                   MUL r0,r0,r1       ;salida*10
                   bx      lr
```

Punto B5:

Uso el circuito «display1». Antes de eso, debo realizar un programa que convierta el número binario a su equivalente a BCD de 4 dígitos. Por ejemplo, el programa debe convertir el número 0011.0010.0000b, que es el valor máximo, en 1000.0000.0000.0000b, que colocará un 8 en el primer dígito, y ceros en los tres restantes.

El circuito completo queda:

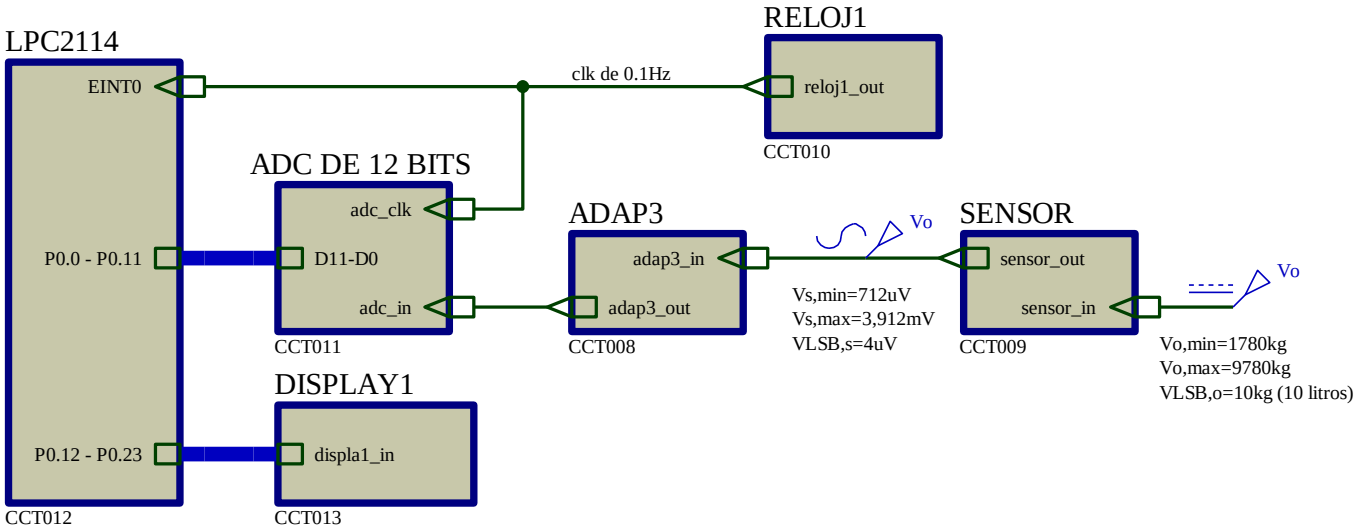
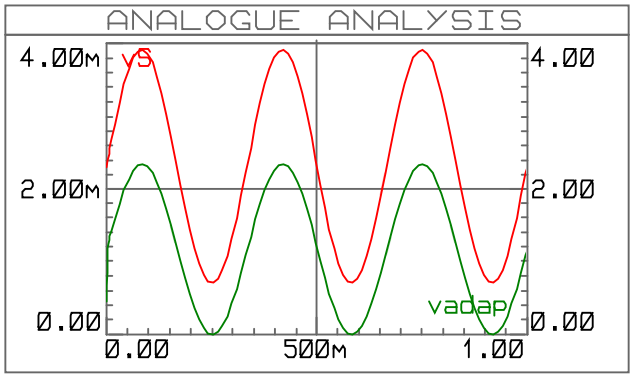


Figura 8: Circuito completo del final del 23 de octubre de 2013.



9 de octubre

Para un sistema basado en ARM-7:

A.- Diseñe la interface para operar un convertidor AD externo de 12 bits. Esta interface debe:

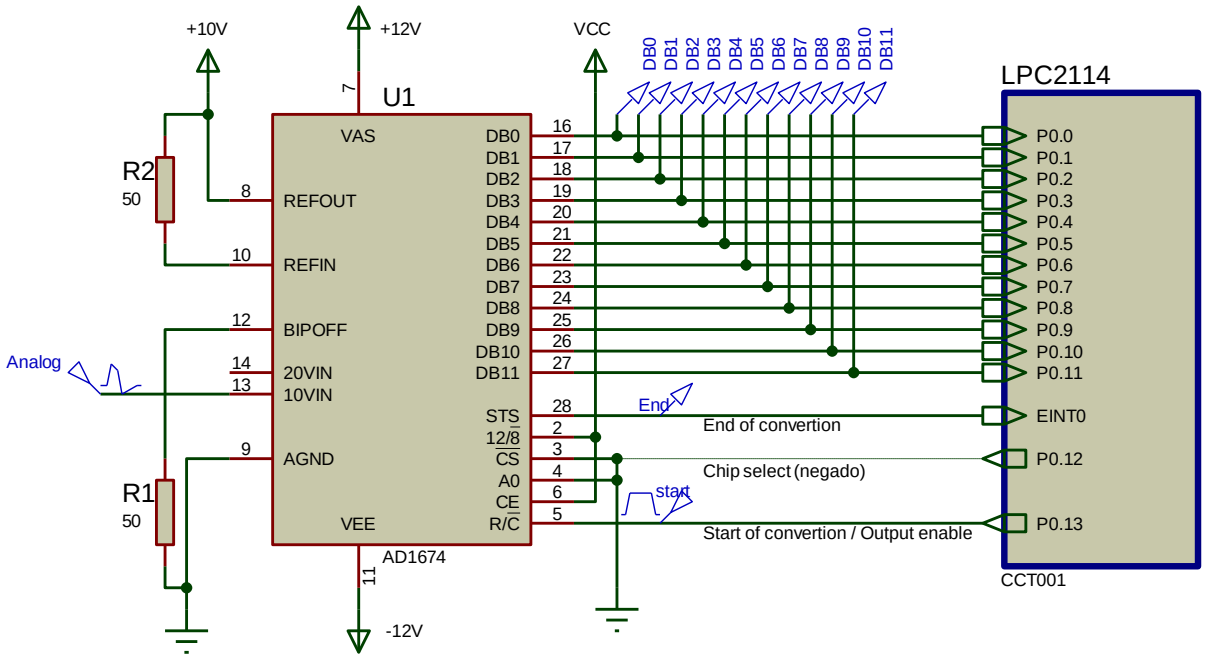
- Tener la señal de «start of convertion». Dicha señal opera por bajo.
- Contemplar un modo de lectura de los datos por parte del micro, esto implica:

  * Una línea de «Output enable» que opere por bajo.
  * Una línea de CS que opere por bajo.
  * Las líneas de datos necesarias.
  * Una línea de «End of convertion» usada para pedir interrupción.

Esta interface se implementará usando los dispositivos del ARM.

B.- A este convertidor que usa una referencia de **3V** se debe conectar un puente de termistores con una salida de **0,016 mV/0,01 ºC**. Se pide diseñar la etapa amplificadora y establecer las tolerancias de los componentes para un error de 0,1%.

Punto A:



El convertidor DA que uso es el AD1674.

- El pin 4, $\overline{CS}$, se activa por bajo y debe ir conectado a una salida digital del ARM (P0.12 en este caso). La línea en puntos significa que no está conectada, para la simulación conecté este pin del ADC a tierra.
- Las líneas de datos del ADC son DB0-DB12 y van a las entradas digitales del ARM P0.0-P0.11.
- La línea «End of convertion» se pone en alto cuando se está realizando la conversión y en bajo cuando la misma terminó.
- Las líneas «Start of convertion» y «Output enable» son las mismas. El pin 5 del ADC, $R/\overline{C}$, cuando está en alto comienza la conversión, mientras tanto tiene las salidas digitales en alta impedancia. Cuando el pin 5 se pone en bajo se activan las salidas digitales.

En la Figura 9a se ve cómo varían las salidas digitales DB0-DB11 en función de la señal de entrada analógica, siendo DB0 el LSB. En la Figura 9b se ve un zoom donde se aprecia que cuando la señal «start» se pone en bajo, las salidas digitales se ponen en alta impedancia y la señal «end» se pone en alto indicando que se está realizando el proceso de conversión; proceso que finaliza cuando la señaal «end» se pone en bajo nuevamente. En realidad la señal «end» no se pone en alto al mismo tiempo que «start» se pone en alto, por eso el zoom1. En la Figura 9c, un zoom más alejado, se ve cómo las salidas digitales se activan cuando la señal «start» se pone en alto nuevamente.

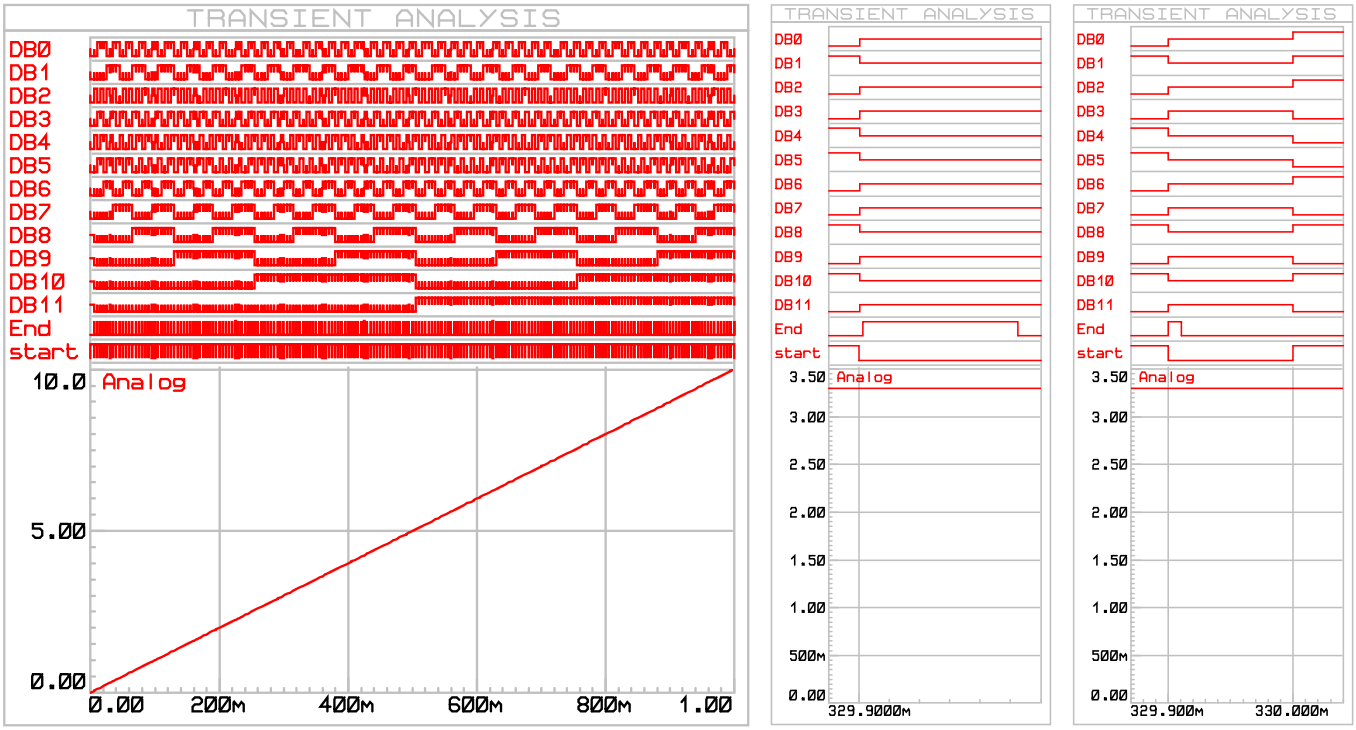


Figura 9: a) Figura completa, b) zoom1, c) zoom2.

Punto B:

$$G_{adap} = \frac{V_{LSB}}{V_{LSB,s}} = \frac{732{,}42\,\mu}{16\,\mu} \approx \boxed{45{,}78 = G_{sensor}}$$

- $V_{LSB} = \frac{V_R}{2^n} = \frac{3}{2^{12}} = 732{,}42\,[\mu V]$
- $V_{LSB,s} = V_{LSB,o} \cdot G_{sensor} = 0{,}01[^oC] \cdot \frac{0{,}016\,[mV]}{0{,}01[^oC]} = 0{,}016\,m = 16\,[\mu V]$

Nota: como me dan la función de transferencia (ganancia) del sensor y no la resolución deseada de la señal primaria, $\mathbf{V_{LSB,o}}$, lo que hago es igualar esta última al denominador de la ganancia del sensor: **0,01ºC**.

Como $\mathbf{10<G_{sensor}<100}$, uso el circuito «adap2», cuyo valor de $\mathbf{R_4}$ es:

$$R_4 = 1000 \cdot G_{adap} = 1k \cdot 45{,}78 = \boxed{45{,}78\,[k\Omega] = R_4}$$

La tolerancias de los componentes pasivos y el error de ganancia son:

$$E_{G,R} = \frac{\varepsilon_G}{V_R} = \frac{1{,}3m}{3} \approx 433\mu = \boxed{433\,[ppm] = E_{G,R}}$$

- $\varepsilon_G = \frac{\varepsilon_T - \varepsilon_{ADC}}{2} = \frac{0{,}001 \cdot V_R - V_{LSB}/2}{2} = \frac{0{,}001 \cdot 3 - 805{,}7\,\mu/2}{2} \approx 1{,}3\,[mV]$

Nota: como el ejercicio no me especifica ningún fondo de escala de señal primaria, $\mathbf{V_{FE,o}}$, uso la máxima que me permite la resolución de 12 bits; por eso $\mathbf{V_{FE}=V_R}$.

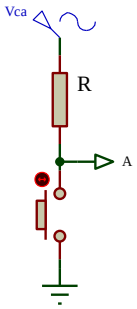
24 de julio

A.- Ud. dispone de dos pines E/S de un procesador ARM. Con ellos debe operar un contactor con bobina de **220Vca** y **300mA** de consumo. El mismo DEBE ser accionado por el flanco descendente de un pulso de **1µs** de ancho (mínimo). El diseño debe asumir que usted memoriza ese flanco en un componente externo para accionar el contactor y utiliza el otro pin de E/S del ARM para desactivarlo. Se debe:

A1.- Dibujar el circuito de mando, con todos los componentes necesarios.

A2.- Escribir una rutina en assembler del ARM para generar el flanco descendente, con la duración especificada. (ES NECESARIO ESTE PUNTO PARA ABROBAR.)

B.- Diseñe la interface necesaria para el manejo de la señal de entrada de la figura, usando el punto (A) como acceso hacia el procesador. Si **R=47kΩ**,dimensione los componentes para que la constante de tiempo sea menor que **5ms**.



C.- Se dispone de un procesador basado en arquitectura ARM, con un ADC externo de **12 bits** y una V_R**=3,3V**. Se tiene también un transductor cuya función de transferencia es **80µV/ºC** (**0V=0ºC**). Se pide:

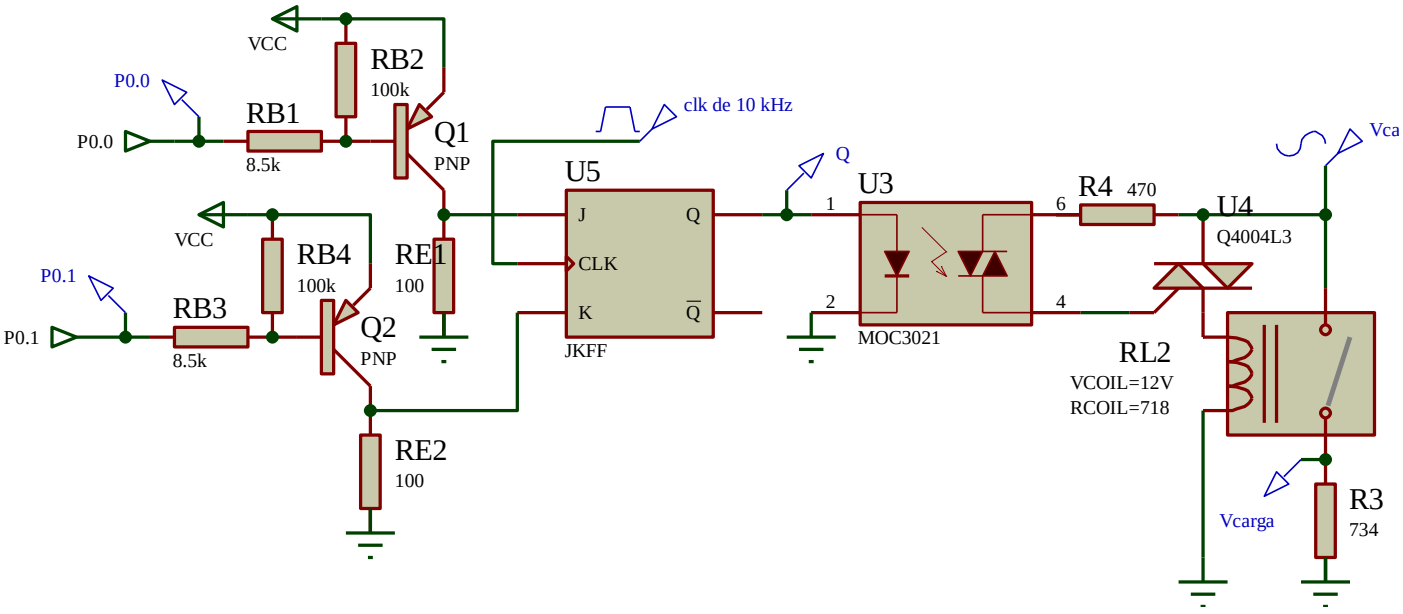
C1.- La ganancia necesaria para poder medir una temperatura de hasta **100ºC** con una resolución de **0,05ºC** por cuenta.

C2.- Especificar la tolerancia de los resistores que establecen la ganancia del amplificador para un error de **0,1%**.

D.- Ejercicio RS232.

Punto A:

Punto A1:



El circuito tiene tres etapas: la etapa del transistor y FF, la etapa del opto-triac y la etapa del triac y el relé.

Etapa del triac y el relé: la carga es R_L, y como está alimentada por **220Vca** y por ella pasan **300mA**, su valor es $R_L=V_{ca}/I_L=220V/0,3A=733,33\Omega$. En serie con R_L va conectado el contactor del relé, el cual es normal abierto. La elección del triac debe ser acorde a las necesidades del circuito, en este caso conecto la bobina del contactor en serie con el triac, el cual debe bancarse, en el peor de los casos, $V_{pico}-V_{COIL}=311-12=299V$ (los **311V** corresponden al voltaje pico de una onda de **220Vca**, y **12V** es el voltaje necesario en la bobina para

que el contactor se cierre); por eso elijo un triac que tenga un $\mathbf{V_{DRM}>299mV}$, en este caso el **Q4004L3**, que tiene un $\mathbf{V_{DRM}=400V}$.

El voltaje $\mathbf{V_{DRM}}$ significa *«Repetitive peak blocking voltage»* y es algo así como el voltaje pico repetitivo en estado de bloqueo, es muy importante que este valor supere los **299V**. Si uso otro triac, por ejemplo el **Q2004L3** que tiene un $\mathbf{V_{DRM}=200V}$, se va a modificar el ángulo de conducción o la forma de onda, pues el triac se banca **200V** y se tendría que bancar **299V**. En este caso se debe hacer que los **99V** restantes caigan en una resistencia **R** conectada entre el triac y la bobina del relé, la cual puedo calcular con la siguiente ecuación: $\mathbf{R=V_R/I_R=V_R \cdot R_{COIL}/V_{COIL}=99 \cdot 718/12 \approx 5{,}9k\Omega}$, donde $\mathbf{R_{COIL}}$ es la resistencia de la bobina.

Etapa del opto-triac: la resistencia $\mathbf{R_4}$ limita la corriente que pasa por el foto-triac para evitar que se dañe y puede estar del orden de los **360** a **470Ω**. No hace falta mayores explicaciones de esta etapa, cuando en el diodo de entrada hay un voltaje de **1,15V** se activa el triac de salida, el cual está opto-acoplado.

Etapa del transistor y FF: se trata de un circuito amplificador con transistor PNP en configuración emisor común. Los pines GPIO del ARM arrojan **0V** para un «0 lógico» y **3,3V** y **–4mA** para un «1 lógico»; es por eso que uso un PNP, para que con **3,3V** la corriente circule hacia el puerto GPIO.

Para que el transistor esté encendido necesito que la tensión en la base sea menor a $\mathbf{V_{CC}-V_{BEQ}=5-0{,}75=4{,}25V}$. Asumo que $\mathbf{R_{b1}<<R_{b2}}$ (ver Figura 13), entonces la base tiente el mismo potencial que la señal «P0.0», es decir, entre **0** y **3,3V**. Cuando **P0.0=0V** el transistor $\mathbf{Q_3}$ se enciende porque su diodo base-emisor queda polarizado en directo; cuando **P0.0=3,3V** el voltaje del diodo, $\mathbf{V_{BEQ}}$, no supera los **0,75V** y el transistor se apaga.

Para el diseño hago lo siguiente: el valor de $\mathbf{I_C}$ es

$$I_C = \frac{V_{CC}}{R_E} = \frac{5}{100} = 50\,[mA]$$

Para que $\mathbf{Q_3}$ opere en la zona de saturación la corriente en la base debe ser mayor a $\mathbf{I_C/\beta}$.

$$I_b > \frac{I_C}{\beta} = \frac{50m}{120} = 416\,[\mu A]$$

Tomo $\mathbf{I_b=500\mu A}$. Aplico LKV a la red de polarización y despejo $\mathbf{R_{b1}}$.

$$V_b = -I_b\left(R_{b1} \,||\, R_{b2}\right) - V_{BEQ} + V_{CC} \qquad \rightarrow \therefore R_{b1} \,||\, R_{b2} = \frac{V_{CC} - V_{BEQ} - V_b}{I_b}$$

Como asumí que $\mathbf{R_{b1}<<R_{b2}}$, me queda

$$R_{b1} \approx \frac{V_{CC} - V_{BEQ}}{I_b} = \frac{5 - 0{,}75}{500\mu} = 8{,}5\,[k\Omega]$$

Para la señal **P0.1** hago lo mismo. De los colectores de $\mathbf{Q_3}$ y $\mathbf{Q_4}$ salen las salidas al flip-flpo JK, para que cuando haya un «0 lógico» en **P0.0**, entre un «1 lógico» en J y se active la salida Q del FF. Dicha salida solo cambiará su estado cuando en **P0.1** haya un «0 lógico». La entrada del clk al FF debe tener una frecuencia lo suficientemente alta como para que su período dure menos que el ancho de pulso de las señales **P0.0 y P0.1**, para darle tiempo al FF a cambiar su estado. En el ejemplo usé una frecuencia de **10kHz**, esto es porque los pulsos en caída de las señales **P0.0 y P0.1** son muy cortos.

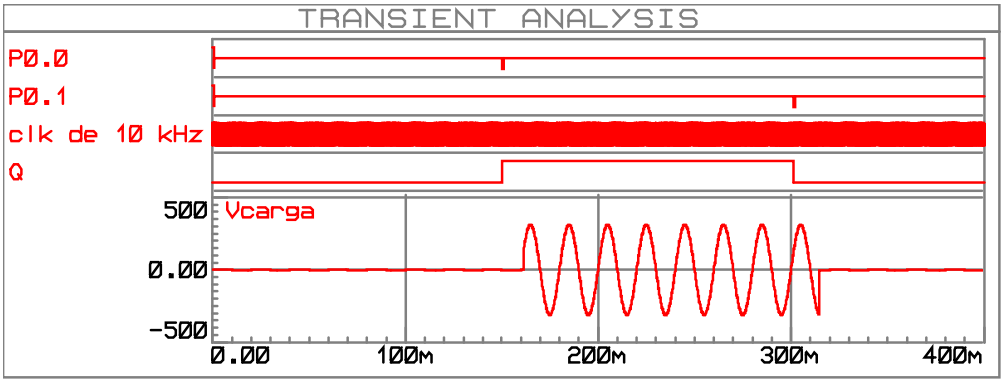


Figura 10: Gráfica en el tiempo de las señales involucradas.

Punto A2:

Código del main.c

```
#include <LPC21xx.h>
int pulso(void);
int pulso1(void);

int main (void)
      {int i=0;
      IODIR0=3;
      IOSET0=3;

      for (i=0;i<20000;i++);    /*Este for seguido de la instrucción nula (;) es para hacer
                                tiempo antes del pulso en P0.0. Por el mismo motivo está el
                                for de más abajo*/
      pulso();
      for (i=0;i<20000;i++);
      pulso1();

      while (1);
      return 0;}
```

Código de pulso.s

```
pulso:      mov r0,#0xE0000000
            mov r1,#0x00020000
            mov r2,#0x00008000
            mov r3,#0x0000000C
            add r0,r0,r1
            add r0,r0,r2
            add r0,r0,r3       //r0=E002800C (IOCLR0)
            /*La dirección de IOCLR0 es 0E002800C (figura en la hoja de datos del LPC2114),
            pero la instrucción mov no permite mover este valor en ningún registro por direc
            cionamiento inmediato, así que tengo que cargar dicha dirección por partes en r0.
            Debe haber una manera mejor y meme tediosa de hacer esto, pero es la primera que
            se me ocurrió y funciona.*/

            mov r4,#0x1
            str r4,[r0]        //IOCLR0=1
            /*Con esto cargo un 1 en la dirección 0xE002800C, que es lo mismo que hacer
            IOCLR0=0x1 en el main.c*/

//ancho de pulso
            mov r6,#0xAA
aux:        cmp r6,#0x0
            beq sig
            sub r6,r6,#0x1
            bne aux
//fin ancho de pulso
            /*El ancho de pulso me da como mínimo, por lo menos lo que simulé en el Proteus,
            unos 4us aproximadamente, que es el tiempo en que el micro se demora en procesar
            (sin la rutina ancho de pulso) las 3 instrucciones de más abajo. En la Figura 11
            se ven los distintos anchos de pulso que obtengo dándole valores a r6.*/

sig:  mov r5,#0x8
            sub r0,r0,r5       //r0=E0028004 (IOSET0)
            str r4,[r0]        //IOSET0=1
            /*Para hacer que P0.0 vuelva a un «1 lógico» pongo un 1 en la dirección E0028004,
            que es donde está ubicado el registro IOSET0, por eso le resto 0x8 a r0.*/
bx lr
```

Código de pulso1.s

```
pulso1:     add r0,r0,r5        //r0=E002800C (IOCLR0)
            mov r4,#0x2
            str r4,[r0]         //IOCLR0=2
            /*Con el pulso en P0.1 hago lo mismo que antes, solo que esta vez por tratarse del
pin P0.1 del ARM, debo poner en alto el bit 1 de IOCLR0, es decir hacer IOCLR0=0x2.*/

//ancho de pulso
            mov r6,#0xAA
aux:        cmp r6,#0x0
            beq sig
            sub r6,r6,#0x1
            bne aux
//fin ancho de pulso

sig:        sub r0,r0,r5        //r0=IOSET0
            str r4,[r0]         //IOSET0=2
bx lr
```

En la Figura 11 se ve cómo la señal **P0.0** varía el ancho de pulso negativo para distintos valores. Esto lo hago dándole distintos valores al registro **R6** en la función en assembler.

Duda: la figura muestra que el mínimo tiempo de duración (≈**3µs**) lo obtengo cuando no tengo la rutina que provoca la espera, es decir, de por sí el programa ya se demora más de **1µs** en pasar **P0.0** de alto a bajo y de bajo a alto. Así que no sé cómo hacer para obtener un ancho igual a **1µs**.

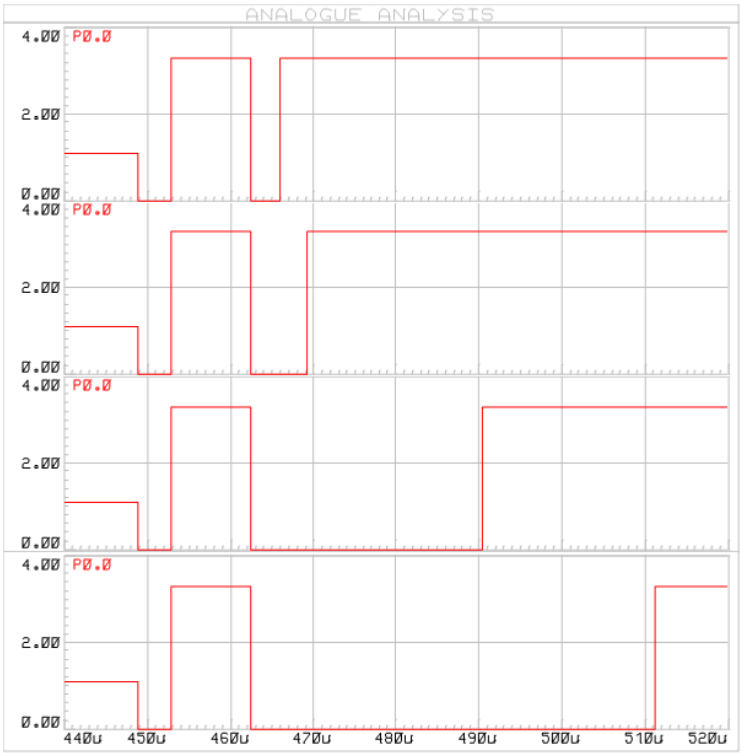
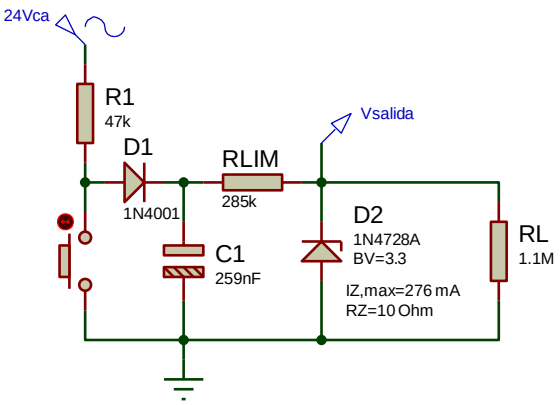


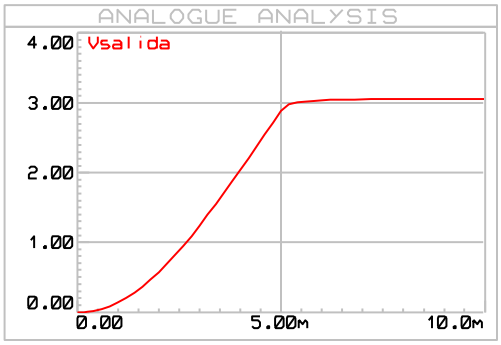
Figura 11: Duración del ancho de pulso de P0.0 para diferentes valores cargados en r6, a) sin rutina, b) con r6=0x0, c) con r6=0x5 y d) con r6=0xA.

Punto B:

Nota: El ejercicio así hecho no sé si está bien, pero es lo que encontré. Lo saqué de internet y los valores dan más o menos.

El circuito de adaptación es el siguiente:





Para el cálculo del condensador uso la fórmula:

$$C = \frac{F_R \cdot I_{lim} \cdot t}{\Delta V_{pp}} = \frac{0{,}0502 \cdot 103\mu \cdot 20m}{0{,}4} \approx \boxed{259\,[nF] = C}$$

- $I_{lim} = I_Z + I_L = 103\,[\mu A]$
  - $I_Z = 100\,[\mu A]$ $(valor\ estimado)$
  - $I_L = \frac{V'_Z}{R_L} = \frac{3{,}3}{1{,}1M} = 3\,[\mu A]$
    - $V'_Z = V_Z + I_Z \cdot R_Z = 3{,}3 + 100\mu \cdot 10 \approx 3{,}3\,[V]$

Donde **F_R** es el factor de regulación del zener, **t** es el tiempo que corresponde a la frecuencia de **50Hz** de salida debido al rectificador de media onda (1N4001), y **ΔV_{pp}** es el rizado de la tensión de salida, que en realidad en el ejercicio es **0,1** pero yo le doy más rizado para que el capacitor resultante provoque una constante de tiempo de **5ms** aproximadamente. Todos estos valores los saqué del ejercicio de internet. **I_Z** es el valor estimado, uso **100µA**, un valor mucho menor que **$I_{Z,max}$=276mA**; **R_L** es más o meme el valor aproximado de la resistencia de entrada de un puerto de uso general del ARM. (En realidad los datos que figuran en el manual de ARM son **V_{IH}=3,3V** e **I_{IH}=3µA**, y de ahí saco **R_L**.)

La resistencia limitadora, **R_{lim}**, la calculo con:

$$R_{lim} = \frac{V_{in} - V'_Z - (R_Z + 47k) I_Z}{I_Z\left(1 + \frac{R_Z}{R_L}\right) + \frac{V'_Z}{R_L}} = \frac{33{,}24 - 3{,}3 - (10 + 47k)103\mu}{103\mu\left(1 + \frac{10}{1{,}1M}\right) + \frac{3{,}3}{1{,}1M}} \approx \boxed{285\,[k\Omega] = R_{lim}}$$

- $V_{in} = 24 \cdot \sqrt{2} - V_D = 24 \cdot \sqrt{2} - 0{,}7 = 33{,}24\,[V]$

Donde **R_Z** es la resistencia interna del zener, sacado de la hoja de datos; y **V_{in}** es la tensión de entrada máxima después del diodo, hay que usar el valor pico de los **24Vca**, que es un valor eficaz. El gráfico está a la derecha del circuito, y alcanza el valor de **3,1V** en menos de **5ms** aproximadamente.

Punto C:

Punto C1:

La Figura 12 ayuda a entender los parámetros involucrados en la transformación de la señal primaria, **v_o**, en la señal eléctrica no adaptada, **v_s**, para luego convertirse en la señal adaptada, **v_{adap}**, la cual entra al ADC.

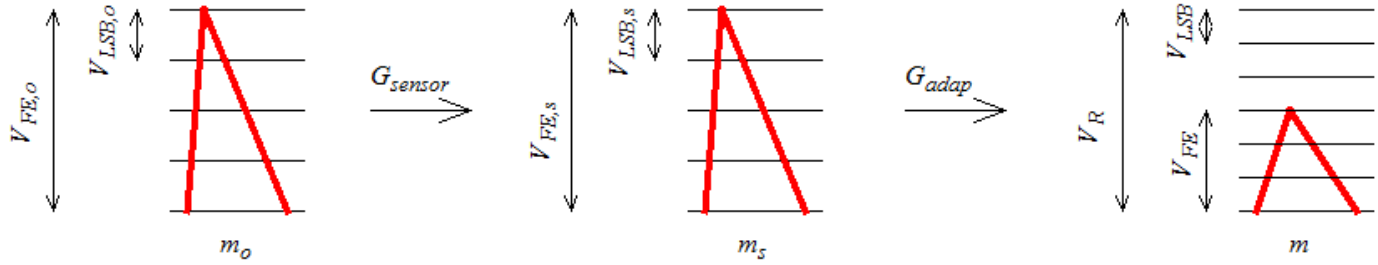


*Figura 12: Señal de entrada primaria, **v_o**, señal eléctrica no adaptada, **v_s**, y señal adaptada, **v_{adap}**, con todos los parámetros involucrados.*

$V_{FE,o}$ es el fondo de escala de la señal[1] primaria, por eso el sufijo «o», es la diferencia entre los valores máximo y mínimo que dicha señal podrá alcanzar. Por lo general se mide en ºC, ºK, kg, mA, litros, etc. En este caso **$V_{FE,o}$=100ºC**, acá asumo que el valor máximo es 100 y el mínimo 0, pues no se dice otra cosa.

$V_{LSB,o}$ es el escalón o bit meme significativo (**L**ess **S**ignificant **B**it) de la señal primaria. Es la resolución de la señal **v_o**. Aquí **$V_{LSB,s}$=0,05ºC**, pues se da como dato.

m_o es la cantidad de escalones que tiene la señal **v_o**, y es el cociente entre las dos señales anteriores. En este caso **$m_o=V_{FE,o}/V_{LSB,o}$=100/0,05=2000**.

G_{sensor} es la función de transferencia del sensor que transforma la señal primaria, v_o, en la señal eléctrica sin adaptar, **v_s**. El enunciado dice que **G_{sensor}=80µV/ºC**.

$V_{FE,s}$ es el fondo de escala de la señal sin adaptar, **v_s**, y se mide en **V** o **mV**. Su valor es igual al producto entre la ganancia del sensor y el fondo de escala de la señal primaria. En este caso **$V_{FE,s}=G_{sensor}\cdot V_{FE,o}$= 80µ·100=8mV**.

$V_{LSB,s}$ es la resolución de dicha señal. De la misma forma que **$V_{FE,s}$**, su valor es el producto entre la ganancia del sensor y la resolución de la señal primaria. **$V_{LSB,s}=G_{sensor}\cdot V_{LSB,o}$= 80µ·0,05=4µV**.

m_s es la cantidad de escalones de **v_s**, y es el cociente entre los dos valores anteriores. Su valor es siempre igual a **m_o**. Así que **$m_s=m_o$=2000**.

G_{adap} es la ganancia de nuestro adaptador (amplificador) que lleva los niveles de vs a los niveles de la referencia del convertidor AD (sea externo o del ARM). Es igual al cociente entre **V_{FE}** y **$V_{FE,s}$**, o entre **V_{LSB}** y **$V_{LSB,s}$**. Como no calculo **V_{FE}** ni **V_{LSB}**, no puedo hallar **G_{adap}** por ahora.

V_R es la tensión de referencia del ADC, puede ser de **3,3V** si es el del ARM, o de otro valor, por lo general mayor, si es externo. En este caso el ADC es externo pero **V_R** coincide con el del ARM, **V_R=3,3V**.

Duda: el rango de entrada del ADC del LPC2114 es de **0** a **3V**, lo dice en su hoja de datos. Pero en el Proteus puedo poner hasta **3,3V** a la entrada de cualquiera de los cuatros ADC y funciona perfectamente. Voy a tomar este valor como **V_R** del ARM de aquí en adelante excepto que el enunciado diga explícitamente que **V_R=3V**.

V_{LSB} es la resolución o el tamaño del escalón de la señal adaptada, **v_{adap}**. Su valor es igual al cociente entre **V_R** y **m**. Como no tengo **m** no puedo saber tampoco **V_{LSB}**, el cálculo queda para más adelante.

V_{FE} es el fondo de escala de **v_{adap}**, no es igual a **V_R** pues representa el voltaje máximo que alcanza vadap cuando se usaron todos los escalones **m**, que dicho sea de paso este último **no** es igual a **m_s**. En el caso de que se usen todos los escalones **m**, entonces **$V_{FE}=V_R$**. Su valor es igual al producto entre **V_{LSB}** y **m_s**, como no tengo el primero, tampoco puedo calcular nada.

m es el número de escalones en el cual se va a dividir la entrada al ADC. Depende directamente del número de bits del ADC que estemos usando, **n**. También depende indirectamente de **m_s**, pues éste último me va a determinar cuántos bits son necesarios usar para la resolución deseada. El valor de **m** es igual a 2 elevado a la **n** potencia. En este caso **n=12** (dato), entonces **$m=2^n=2^{12}$=4096**.

Con este valor calculo lo que me falta.

$$G_{adap}=\frac{V_{LSB}}{V_{LSB,s}}=\frac{805,67\mu}{4\mu}=\frac{V_{FE}}{V_{FE,s}}=\frac{1,61}{8m}\approx \boxed{201,41=G_{adap}}$$

$$\bullet\ V_{FE}=V_{LSB}\cdot m_s=805,67\mu\cdot 2000=1,61\left[V\right]$$

$$\bullet\bullet\ V_{LSB}=\frac{V_R}{m}=\frac{3,3}{4096}=805,67\left[\mu V\right]$$

[1] En realidad no es una señal, porque los kilos o litros, por ejemplo, no constituyen una propiamente dicha. Pero a fines prácticos vamos a considerar sus variaciones en el tiempo como una señal eléctrica.

Punto C2:

$$E_{G,R} = \frac{\varepsilon_G}{V_R} = \frac{1{,}45m}{3{,}3} = 438{,}9\mu = \boxed{439\,[ppm] = E_{G,R}}$$

- $\varepsilon_G = \frac{\varepsilon_T - \varepsilon_{ADC}}{2} = \frac{0{,}001 \cdot V_R - V_{LSB}/2}{2} = \frac{0{,}001 \cdot 3{,}3 - 805{,}67\mu/2}{2} \approx 1{,}45\,[mV]$

$$E_{G,FE} = \frac{\varepsilon_G}{V_{FE}} = \frac{1{,}45m}{1{,}61} \approx 900\mu = \boxed{900\,[ppm] = E_{G,R}}$$

El circuito completo es:

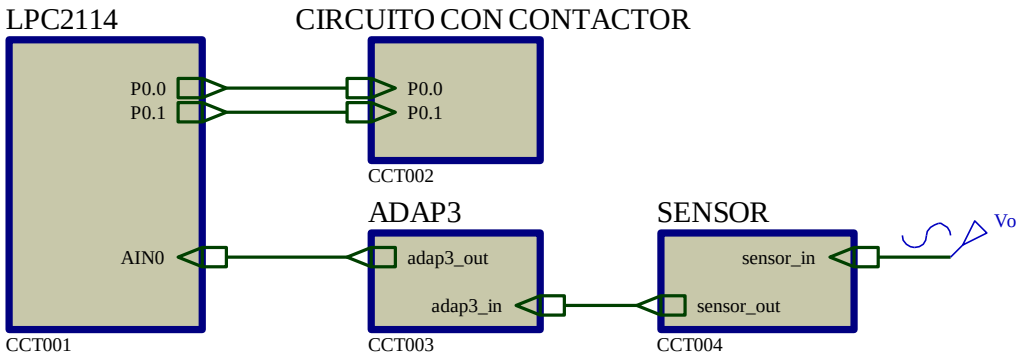


Figura 13: circuito completo.

Finales de 2013 (3 finales)

23 de mayo

Usando un procesador basado en arquitectura ARM, con un ADC interno de **10 bits** y tensión de referencia fija de **3,3V**, resuelva los siguientes problemas:

A.- Utilizando un transductor cuya función de transferencia es **20µV/Kg**, calcular:

A1.- Ganancia necesaria para medir una carga de hasta **800kg** con una resolución exacta de **1kg** por cuenta.

A2.- Máxima resolución posible y ganancia necesaria para una carga ahora de **300kg**.

A3.- Calcular el error de cuantificación para ambos casos.

B.- Especificar nombres de registro y valores a asignar para configurar el controlador de interrupciones de la siguiente forma:

B1.- Asignar el timer0 al vector 0 (IRQ).

B2.- Asignar el ADC a la FIQ.

Nota: solo para el caso de la interrupción vectorizada incluir además los registros necesarios para especificar la función de atención.

C.- Una variable de tipo word denominada «puntero» posee la dirección de memoria de un vector de 16 elementos de tipo byte sin signo. Escribir el código en assembler necesario para calcular el promedio de este vector.

D.- Dibuje el circuito de control de un display de diodos de siete segmentos de 4 dígitos y su interface con el microcontrolador, para la misma no se permite usar más de 4 líneas GPIO del microcontrolador en total.

Punto A:

Punto A1:

Los parámetros de la Figura 12 son:

$V_{FE,o} = 800\,[kg] \qquad (dato)$

$V_{LSB,o} = 1\,[kg] \qquad (dato)$

$$m_o = \frac{V_{FE,o}}{V_{LSB,o}} = \frac{800}{1} = 800$$

$$V_{FE,s} = G_{sensor} \cdot V_{FE,o} = 20\mu \cdot 800 = 16\,[mV]$$

$$V_{LSB,s} = G_{sensor} \cdot V_{LSB,o} = 20\ \mu \cdot 1 = 20\,[\mu V]$$

$$m_s = m_o = 800$$

$$G_{adap} = \frac{V_{LSB}}{V_{LSB,s}} = \frac{3,22m}{20\mu} = 161,13 \approx 161$$

- $V_{LSB} = \frac{V_R}{m} = \frac{V_R}{2^n} = \frac{3,3}{2^{10}} = 3,22\,[mV]$
  - $m = 2^n = 2^{10} = 1024$

$V_R = 3,3\,[V]$

$V_{FE} = V_{LSB} \cdot m_s = 3,22m \cdot 800 = 2,58\,[V]$

Punto A2:

Este punto lo hice teniendo mis propias interpretaciones. El ejercicio pide calcular la ganancia del amplificador de adaptación, **G_{adap}**, y la resolución máxima, que el mínimo valor que puede tomar **$V_{LSB,o}$**. Respecto a este último parámetro, depende de **$V_{FE,o}$** y de **m_o**. Como **$V_{FE,o}$=300kg**, tengo que **$V_{LSB,o}=V_{FE,o}/m_o$=300/800= 0,375kg**. Pero este valor puede ser más chico si tomo **m_o=m**, y como el enunciado no dice nada, tengo:

$$V_{LSB,o} = \frac{V_{FE,o}}{m_o} = \frac{300\,[kg]}{1024} \approx 0{,}293\,[kg]$$

Con esto la ganancia queda:

$$G_{adap} = \frac{V_{LSB}}{V_{LSB,s}} = \frac{3{,}22m}{5{,}86\,\mu} = 549{,}49 \approx \boxed{550 = G_{adap}}$$

- $V_{LSB,s} = G_{sensor} \cdot V_{LSB,o} = 20\,\mu \cdot 0{,}293 = 5{,}86\,[\mu V]$

Punto A3:

Duda: no entiendo este punto, el error de cuantificación es **ε_{ADC}**, y este depende de **V_{LSB}**, el cual depende de **V_R** y de **m**, es decir, depende del ADC del ARM. Y como éste siempre sigue siendo el mismo, el error de cuantificación también lo es.

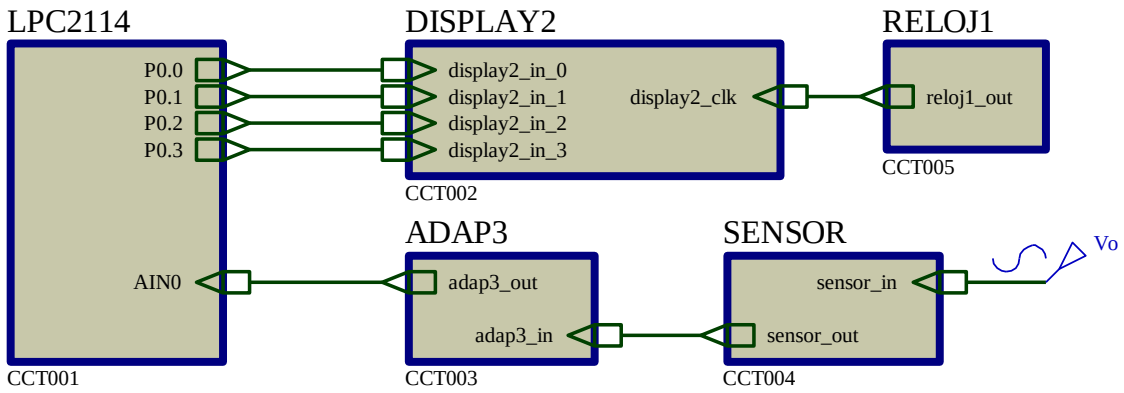
$$\varepsilon_{ADC} = \frac{V_{LSB}}{2} = \frac{3{,}22m}{2} = 1{,}61\,[mV]$$

Punto B:

```
VICIntEnable = (1≪4) | (1≪18);
VICIntSelect = (0≪4) | (1≪18);
VICVectCntl =  (1≪5) | (4≪0);
VICVectAddr =   rutina_irq;
```

Punto D:

Este punto usa el sub-circuito «display2». El circuito completo es:



El clock del circuito «display2» está conectado a un reloj que debe estar sincronizado con la conmutación de las salidas **P0.3-P0.0**, para que la multiplexación se realice correctamente.

5 de septiembre

(Igual al del 25 de julio de 2013). Usando un procesador basado en arquitectura ARM, por favor elabore este problema: una planta de tratamiento térmico implica un proceso de carga de un horno, calentamiento controlado del material cargado, un tiempo a temperatura constante y luego enfriamiento también controlado.

A.- Diseñe el hardware necesario para que se detecte la condición de carga completa. Esto implica:

A1.- Un sensor de proximidad con salida de **220Vca** que detecta que el carro con la carga está en posición. Diseñar la interface hacia la entrada del ARM.

A2.- Otro sensor óptico con salida de luz UV que detecta que la puerta está cerrada (con interface opto-acoplada). Se pide el circuito foto-receptor y su interface hacia el ARM.

B.- Para calentar el horno debe leerse una señal en un potenciómetro cuyos extremos están conectados a **12 Vcc** y a masa. Se lo debe leer en 8 bits. Diseñe la etapa de adaptación de señal necesaria para usar el A/D del ARM. Esta señal es la pendiente de la recta de calentamiento.

C.- La señal leída en el punto B debe multiplicarse por una constante. Escriba en assembler del ARM una rutina para hacer esa multiplicación. El valor de la salida del potenciómetro (POT) como la constante (CON) son dos variables que están en memoria, una en formato BYTE y la otra en formato WORD. La rutina debe resolver la diferencia de tamaño entre ambos valores. El resultado de la multiplicación es mucho menor que 32 bits. (**RES≪2^{32}**).

D.- La temperatura del horno, que varía entre **10** y **950°C**, se mide con una termocupla que da un valor de **3µV/°C** (ignore la compensación de punta fría). Diseñe la interface para que esta lectura se pueda manejar por el A/D del ARM y especifique las tolerancias de componentes para error < **2%**. Considere la resolución normal del A/D y que el fondo de escala corresponde a **1024°C**.

E.- El mando de potencia es un contactor cuya bobina opera en **220 Vca** y consume **0,2A**. Diseñe la interface adecuada a partir de un pin de I/O del ARM. La corriente máxima de salida es de **8mA**.

Punto A:

Punto A1:

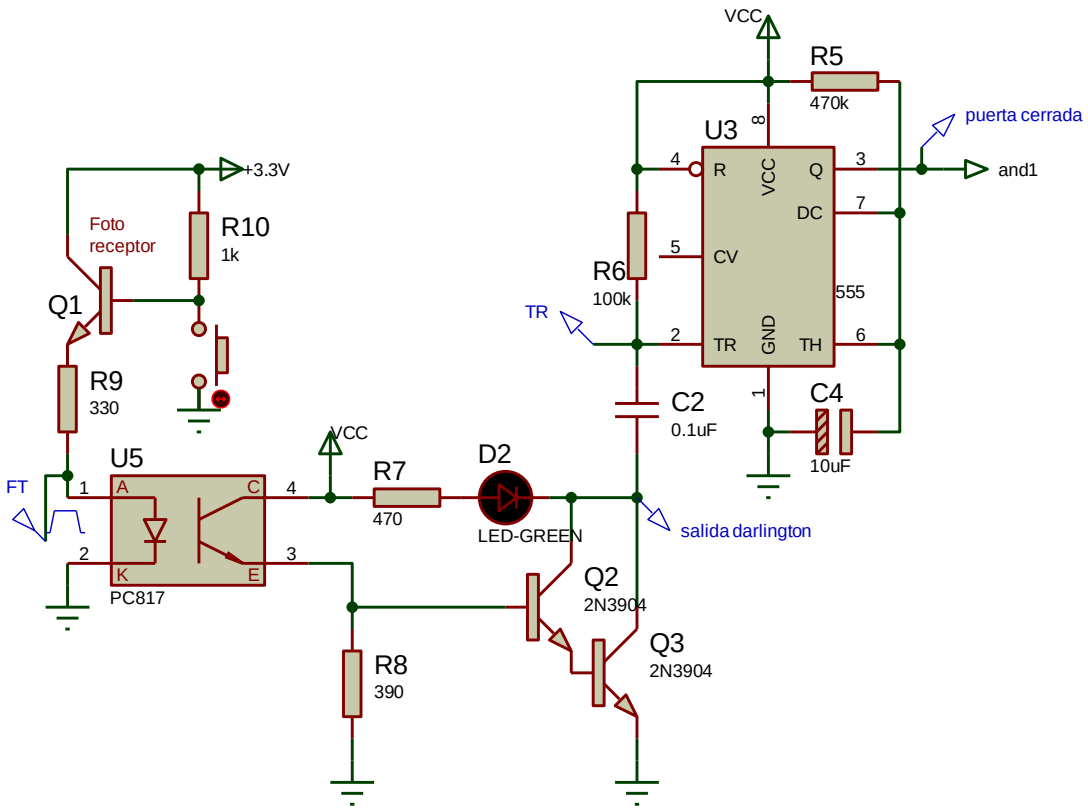


Figura 14: circuito que genera un pulso de 5s de duración cuando la puerta se cierra.

Esta etapa está conformada por una señal que le avisa al ARM que la condición de carga se cumple. Esta señal va a una entrada digital (GPIO) o a una interrupción externa (EINT) del ARM, o a alguna entrada para que el micro pueda saber que se cumplió la condición. La compuerta AND recibe dos señales, la que indica que puerta está cerrada y la que indica que el carro está en posición.

La primera de estas señales, «puerta cerrada», que se ve en la siguiente figura, consta de un 555 configurado como monoestable, que tira un pulso positivo de un poco más de 5 segundos de duración (la cual se puede variar según la fórmula $\mathbf{T=1{,}1\cdot R_5\cdot C_4}$). Este pulso (salida del pin 3) se produce cuando en la entrada **TR** (pin 2) hay un voltaje negativo menor a $\mathbf{V_{CC}/3}$ (que en este caso es **+5V**).

Este voltaje negativo lo obtengo de la configuración Darlington (Q2 y Q3), la cual está para amplificar la señal proveniente del opto. Lo que hay detrás es una etapa amplificadora con el foto transistor, el cual conduce cuando la puerta está cerrada. Esta condición la simulo con la señal pulsante **FT**.

En la Figura 15 se ven las formas de onda de las distintas señales. Cuando **FT** se pone en alto, «salida del darlington» se pone en bajo; esto hace que $\mathbf{C_2}$ se descargue en un instante de tiempo y **TR** cuando cae por debajo de $\mathbf{V_{CC}/3}$ provoca que el 555 tire un pulso positivo de **5s** aproximadamente.

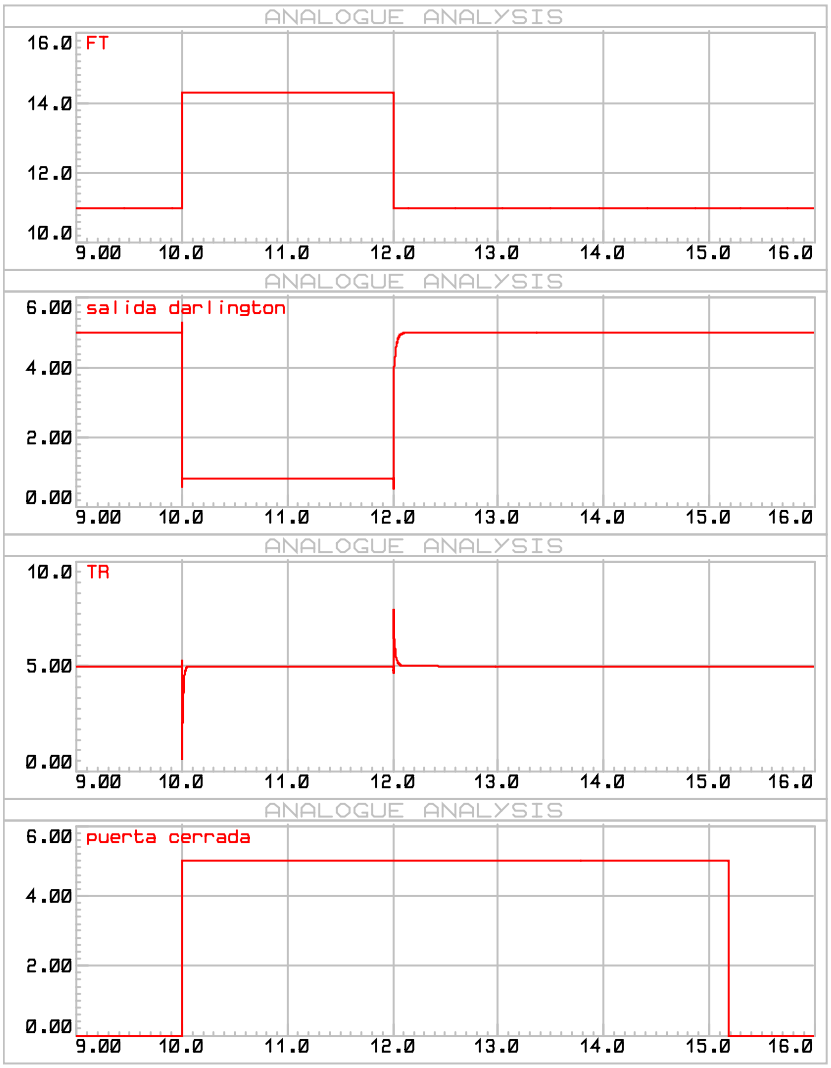


Figura 15: forma de onda de las señales del circuito de «puerta cerrada».

Punto A2:

Aquí directamente considero que la fuente de alterna de entrada, VCA, se activa cuando el sensor de proximidad se activa. Esta etapa no necesita mucha explicación, rectifica la señal alterna y logra poner una continua a la entrada de U2, lo que provoca que «and2» se ponga en alto.

Esta señal, «and2», y «and1», van a la compuerta AND de la Figura 17 y la salida de este va a una interrupción «avisándole» al ARM que las dos condiciones se cumplieron.

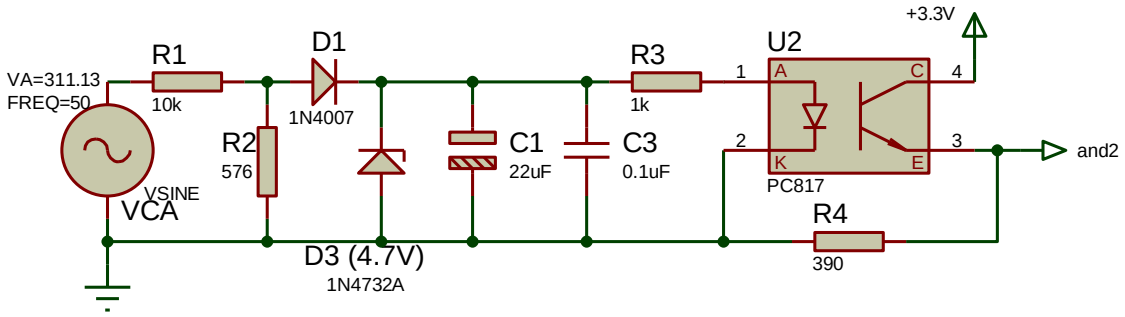


Figura 16: circuito que avisa que el carro está en posición.

Punto B:

Este es un caso típico de adaptación sólo que en este caso arrancamos de la señal no adaptada, **v_s**, que es la que sale del pote conectado a los **12V** de continua.

$$V_{FE,s} = 12\,[V]$$

$$V_{LSB,s} = \frac{V_{FE,s}}{m_s} = \frac{12}{256} \approx 46,88\,[mV]$$

- $m_s = m = 2^n = 2^8 = 256$

$$G_{adap} = \frac{V_{LSB}}{V_{LSB,s}} = \frac{12,89m}{46,88m} \approx 0,27$$

- $V_{LSB} = \frac{V_R}{m} = \frac{3,3}{256} = 12,89\,[mV]$

$$V_R = 3,3\,[V]$$

$$V_{FE} = V_{LSB} \cdot m = 12,89m \cdot 256 = 3,3\,[V]$$

$$V_{LSB} = 12,89\,[mV]$$

$$m = 256$$

Como **G_{adap}<1** lo que hago es usar el circuito «adap2», pues la señal debe invertirse dos veces para quedar positiva. La resistencia **R_4** vale:

$$R_4 = 1000 \cdot G_{adap} = 1k \cdot 0,27 = 270\,[\Omega] = R_4$$

Punto D:

$$V_{FE,o} = 1024\,[^oC]$$

$$V_{LSB,o} = \frac{V_{FE,o}}{m_o} = \frac{1024}{1024} = 1\,[^oC]$$

- $m_o = m = 2^n = 2^{10} = 1024$

$$m_o = 1024$$

$$V_{FE,s} = V_{FE,o} \cdot G_{sensor} = 1024 \cdot 3\mu = 3,07\,[mV]$$

$$V_{LSB,s} = V_{LSB,o} \cdot G_{sensor} = 1 \cdot 3\mu = [\mu V]$$

$$m_s = m_o = 1024$$

$$G_{adap} = \frac{V_{LSB}}{V_{LSB,s}} = \frac{3,22m}{3\mu} \approx 1073,3$$

- $V_{LSB} = \frac{V_R}{m} = \frac{3,3}{1024} = 3,22\,[mV]$

$V_R = 3,3\,[V]$

$V_{FE} = V_{LSB} \cdot m = 3,22m \cdot 1024 = 3,3\,[V]$

$V_{LSB} = 3,22\,[mV]$

$m = 1024$

Como **G>1000**, uso «adap4», entonces $\mathbf{R_G}$ vale

$$R_G = \frac{50k}{G_{adap} - 1} = \frac{50k}{1073,3 - 1} \approx 46,6\,[\Omega]$$

Punto E:

Este punto se calcula igual que el punto A del final del 24 de julio de 2014. El circuito completo es:

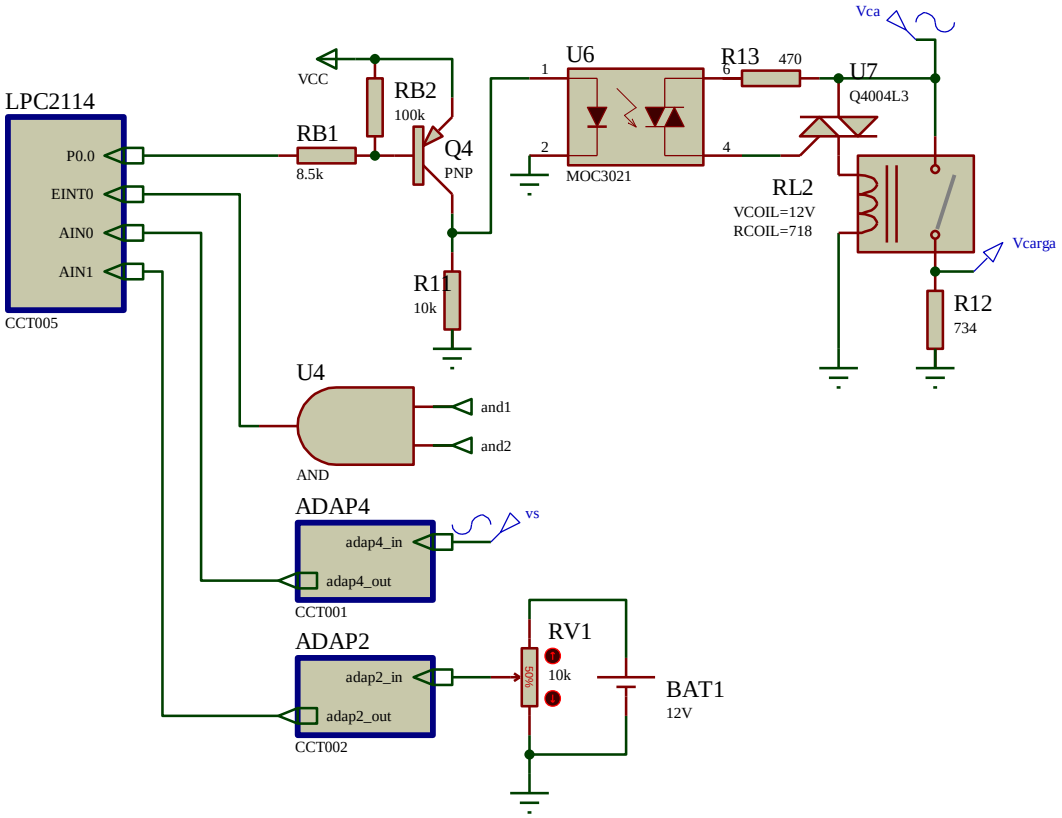


Figura 17: circuito completo.

7 de febrero

(Igual al del 15-dic-2011 y al del 4-jul-2013.) Usando un procesador basado en arquitectura ARM, elabore este problema. Un kilo de gramos se ha instrumentado con sensores de temperatura AD592C. Se usan 16 de estos sensores, con un multiplexor adecuado. La función de transferencia del AD592C es **1µA/ºK**. Se requiere una resolución de **0,1ºC** en el intervalo de **10** a **60ºC**. Ud. debe:

A.- Diseñar el hardware necesario para que se muestree esa señal a una tasa de **0,5Hz** operando por interrupción. Esto significa que se mide UN SENSOR cada medio segundo. Debe incluir el reloj, el multiplexor, el controlador de interrupción del ARM y debe detallar la configuración del mismo.

B.- Diseñar el hardware para acoplar el transductor al convertidor elegido. Recuerde que se mide en **ºC** y el transductor opera en **ºK**.

C.- Establezca las tolerancias de los componentes pasivos de un circuito para un error de **0,1%**.

D.- Cada vez que se complete el multiplexado de las señales (cada 8 segundos) se debe calcular el promedio de ellas y además la diferencia de cada una de las medidas con ese promedio. Escriba el diagrama de flujo para este. Escriba en assembler del ARM la rutina para calcular el promedio.

E.- Si la diferencia de cualquiera de las mediciones con el promedio supera los **2ºC**, se debe activar un ventilador mediante un contactor extremo cuya bobina opera en **220V** de corriente alterna y consume **0,2A**. Diseñe la interfase necesaria. (Este punto se calcula igual que el punto A del final del 24 de julio de 2014, solo hay que cambiar la resistencia $\mathbf{R_L}$ pues circula otra corriente.)

Punto A:

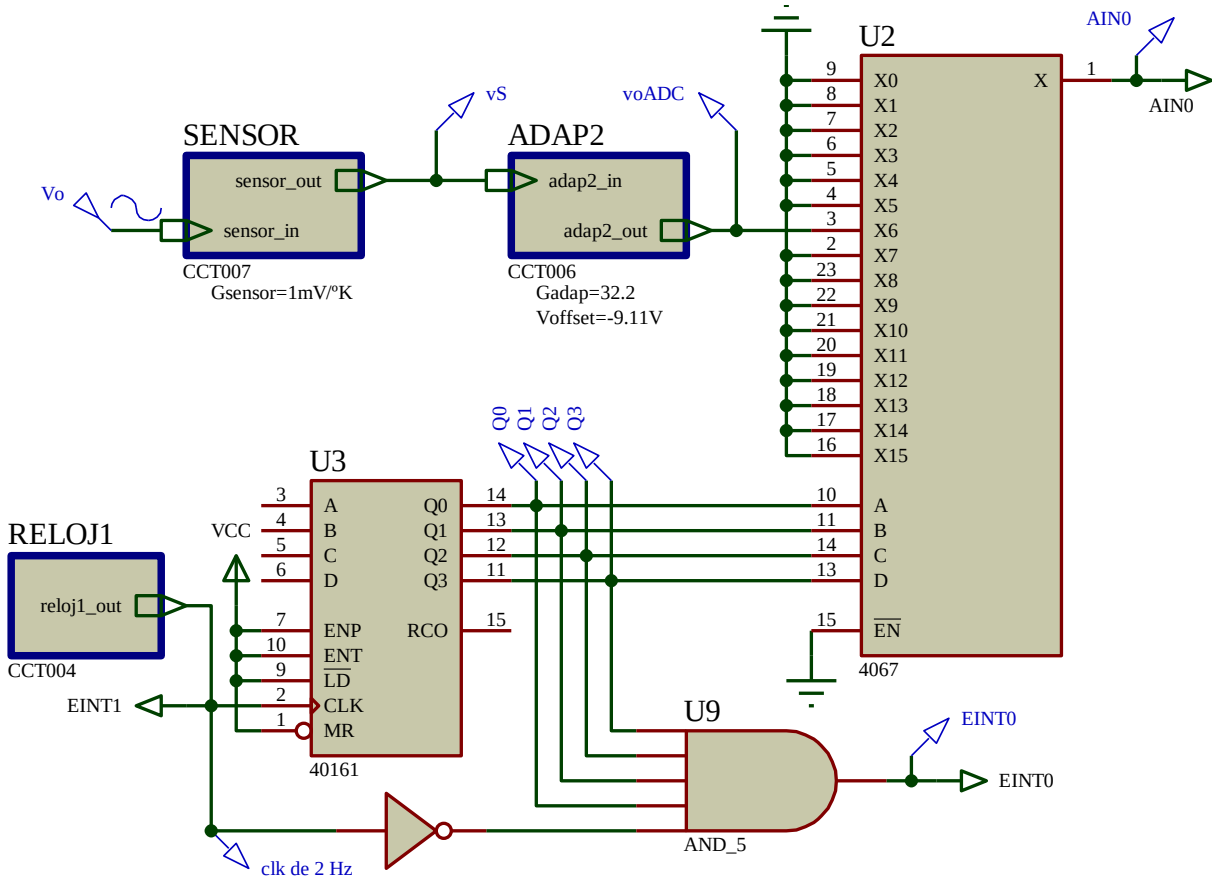


Figura 18: multiplexado de las señales, de las 16 entradas, usé 1 sola, la X6, para la simulación, el resto de las entradas están a masa..

De la hoja de datos del **555** hallo los valores de las resistencias y del capacitor para tener una frecuencia de salida de **2Hz**. Esta señal se introduce en el clock del **40161** para producir un conteo en las entradas A, B, C y D del multuplexor **4067**, el cual, dependiendo de estas entradas, deja pasar a la señal «AIN0», las señales provenientes de los sensores 1 al 16. En la Figura 18 tomo como ejemplo la entradas **X6**.

La Figura 19 muestra la señal $\mathbf{f_{out}=2Hz}$, las salidas $\mathbf{Q_0}$-$\mathbf{Q_3}$ del **74193**, y la salida **al ADC**. Esta salida es igual a **X6** (pin 3 del 4067) cuando $\mathbf{Q_3}$-$\mathbf{Q_0}$**=0110**.

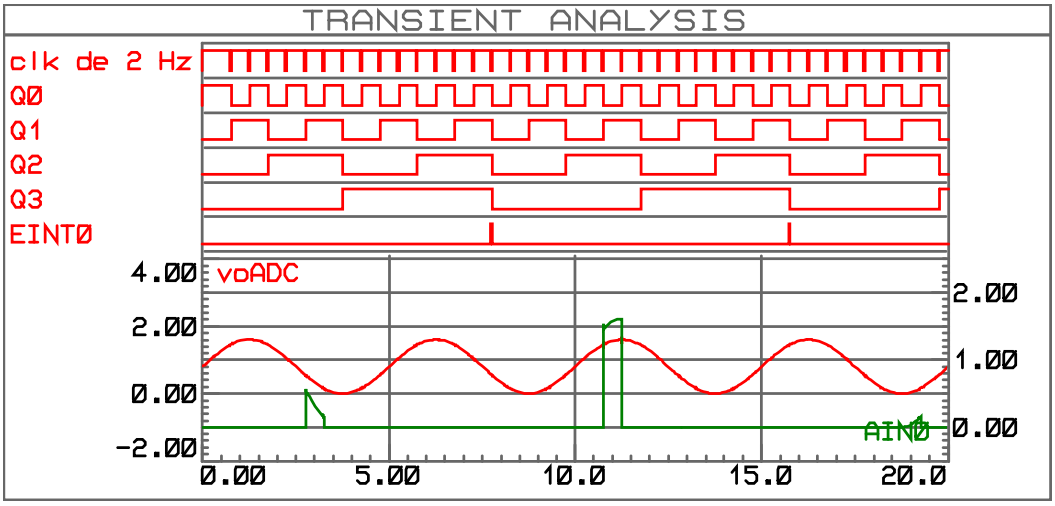


Figura 19: señales involucradas (clk, Q3-Q0, EINT0, $v_{o,ADC}$ y AIN0).

La señal «clk de 2 Hz» va a la entrada EINT1 del ARM, el cual correrá una rutina para leer el valor presente en «AIN0» y guardarlo en un vector. Mediante la compuerta AND produzco la señal «EINT0» que corre una rutina que saca el promedio de las lecturas guardadas, hace la diferencia de cada una de las lecturas con ese promedio, y si alguna diferencia es mayor a **2ºC** activa un ventilador a través de P0.0 del ARM. La señal «EINT0» se pone en alto cada vez que se completa el ciclo de las 16 lecturas (8 seg), esto sucede cuando Q3-Q0=1111; la quinta entrada de la AND es para asegurarme que la interrupción se produzca un tiempo después de que se realizó la última lectura, dándole tiempo así al ARM de realizar y guardar la conversión.

Punto B (adaptación de señal):

Como es medio largo este punto, lo divido en partes:

- Señal de entrada al sensor, **v_o**: la temperatura mínima a la entrada del sensor es **10ºC=283,15ºK**; y la máxima, **60ºC=333,15ºK**. Por lo que **$V_{FE,o}$=333,15–283,15=50ºK**. La cantidad de pasos la puedo sacar haciendo **(60ºC–10ºC)/0,1ºC=500**. Y la resolución será **$V_{LSB,o}=V_{FE,o}/m_o$=50ºK/500=0,1ºK**. Por lo que vo podría ponerla como:

$$v_o = \left(25 \cdot \sin \omega t + 308{,}15\right)\left[^o K\right]$$

- Ganancia del sensor, **G_{sensor}**: la diferencia entre las temperatura indicada y real se llama error de calibración. Es de fábrica, y contribuye a un error total. La Figura 20a muestra de manera exagerada cómo un error de calibración variaría respecto del ideal en función de la temperatura.

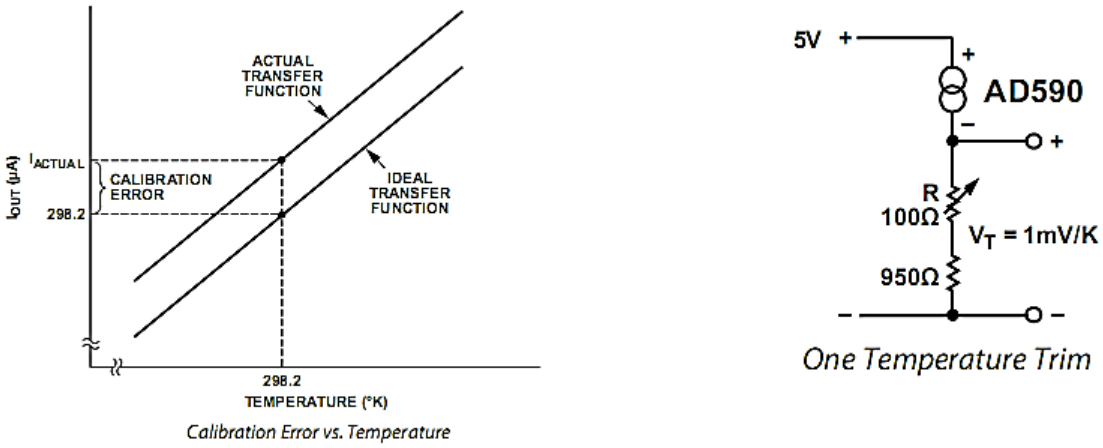


Figura 20: a) Error de calibración en el AD590C y b) circuito equivalente de trasferencia.

Este error es el principal contribuyente al error máximo total en todo el rango de temperatura en que funciona el AD590. Sin embargo, debido a que es un error de factor de escala, es fácil de recortar. La Figura 20b muestra la forma más elemental de lograr esto.

Para recortar este circuito, la temperatura de la AD590 se mide por un sensor de temperatura de referencia y **R** se regula de manera que **V_T=1mV/ºK** a la temperatura deseada. Y este es justamente el valor de la ganancia del sensor, **G_{sensor}=1mV/ºK**.

- Señal de entrada al adaptador, $\mathbf{v_s}$: esta señal es proporcional a $\mathbf{v_o}$, pues la obtengo multiplicando la señal $\mathbf{v_o}$ por la ganancia del sensor, $\mathbf{G_{sensor}}$. Por ende, el número de escalones, $\mathbf{m_s}$, es igual a $\mathbf{m_o}$.

$$V_{LSB,s} = G_{sensor} \cdot V_{LSB,o} = 1\left[\frac{mV}{^oK}\right] \cdot 0{,}1\,[^oK] = 0{,}1\,[mV]$$

$$V_{FE,s} = G_{sensor} \cdot V_{FE,o} = 1\left[\frac{mV}{^oK}\right] \cdot 50\,[^oK] = 50\,[mV]$$

$$m_s = m_o = 500$$

La señal $\mathbf{v_s}$ sinusoidal que uso como entrada del circuito es:

$$v_s = V_{p,s} \cdot \sin(10 \cdot t) + V_{OS,s} = (25 \cdot \sin \omega t + 308{,}15)\,[mV]$$

$$\bullet\ V_{p,s} = V_{p,o} \cdot G_{sensor} = 25 \cdot 1m = 25mV$$

$$\bullet\ V_{OS,s} = V_{OS,o} \cdot G_{sensor} = 308{,}15 \cdot 1m = 308{,}15mV$$

- Señal adaptada, $\mathbf{v_{o,ADC}}$: los parámetros de esta señal dependen de otros factores, que son la cantidad de escalones, **m**, y el voltaje de referencia, $\mathbf{V_R}$. El valor de **m** depende del número de bits, **n**, del convertidor que use. Como el ejercicio no dice nada, tengo la libertad de elegir un ADC externo o el del ARM. Elijo el del ARM, esto supone que **n=10** y $\mathbf{V_R}$**=3V**. Con esto, tengo:

$$m = 2^n = 2^{10} = 1024$$

$$V_{LSB} = \frac{V_R}{m} = \frac{3{,}3}{1024} = 3{,}22\,[mV]$$

$$V_{FE} = V_{LSB} \cdot m_s = 3{,}22m \cdot 500 = 1{,}61\,[V]$$

- Ganancia del adaptador, $\mathbf{G_{adap}}$: la ganancia del circuito adaptador se calcula como el cociente entre los valores de resolución ($\mathbf{V_{LSB}}$ y $\mathbf{V_{LSB,s}}$) o los valores de fondo de escala ($\mathbf{V_{FE}}$ y $\mathbf{V_{FE,s}}$).

$$G = \frac{V_{LSB}}{V_{LSB,s}}\left(= \frac{V_{FE}}{V_{FE,s}}\right) = \frac{3{,}22}{0{,}1m} = 32{,}2$$

Diseño del circuito adaptador:

Como **G>10** uso «adap2», con esto las resistencias $\mathbf{R_4}$ y $\mathbf{R_5}$ son:

$$R_4 = \frac{G_{adap} \cdot R_3}{10} = \frac{32{,}2 \cdot 10k}{10} = 32{,}2\,[k\Omega] \qquad R_5 = \frac{R_4 \cdot V_{ref}}{v_{adap,min}} = \frac{32{,}2k \cdot 15}{9{,}92 - 0{,}805} \approx 52{,}9\,[k\Omega]$$

Donde $\mathbf{v_{adap,min}}$ es el voltaje mínimo que alcanza $\mathbf{v_{adap}}$, esto es, cuando $\mathbf{v_o}$**=+10ºC=283,15ºK**. Como hay que restar el voltaje de offset para que $\mathbf{v_{o,ADC,min}}$**=0V**, el voltaje de **15V** que se suma a la entrada a través de $\mathbf{R_5}$ es negativo.

La Figura 21a muestra la señal primaria $\mathbf{v_o}$ (medida en ºK, eje izquierdo) que pasa por el sensor con una ganancia de $\mathbf{G_{sensor}}$**=1mV/ºK**, produciendo la señal $\mathbf{v_s}$ (medida en mV, eje derecho); en realidad hay una sola sinusoide porque están superpuestas. Lo mismo sucede con la Figura 21b, con las señales $\mathbf{v_s}$ y $\mathbf{v_{o,ADC}}$ (con $\mathbf{G_{adap}}$ de por medio).

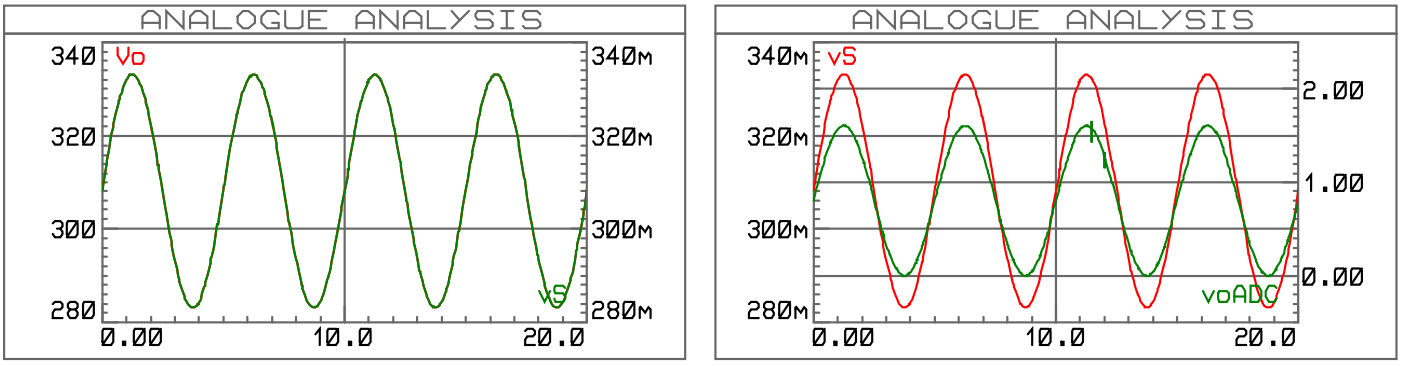


Figura 21: Gráfica en el tiempo de las señales a) v_o, en rojo y v_s, en verde; y b) v_s, en verde y $v_{o,ADC}$, en rojo.

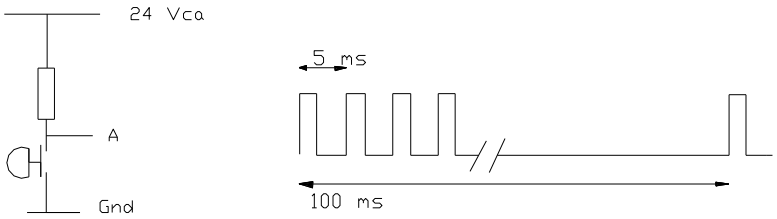
Finales del 2011 (2 finales)

22 de diciembre

Diseñe una balanza para pesar ganado con un ε_T**=0,1%**. Debe usar 4 celdas de carga de **1500kg** de capacidad c/u conectadas a cada una de las esquinas de una jaula cuadrada. Se necesita una $V_{LSB,o}$**=1kg** en cada celda. La sensibilidad de las celdas es G_{sensor}**=5μV/kg**. El valor máximo a pesar es el de 4 animales juntos, estimado en **4000kg**, a los que hay que sumar la jaula que los contiene, unos **600kg**. Se pide:

A.- Asuma que V_R**=10V**. Diseñe el circuito de adaptación de señal para usar esa referencia, estableciendo como condición que CADA ESCALÓN de4x la salida del convertidor DEBE ser equivalente a **1kg**.

B.- Diseñe un mecanismo de ajuste para compensar el peso de la jaula. La indicación de cero se da por medio de una tecla (cero). Ud. puede usar el hardware para compensar el peso de la jaula INCLUYÉNDOLO EN EL DISEÑO DEL PUNTO 1. O por software, generar un offset para hacer lo propio. La señal de la tecla (cero) se obtiene de un circuito como el de la figura de la izquierda. Diseñe la interfase adecuada desde A hasta el micro y el diagrama del programa.



C.- Con una tecla igual a la de la figura de la izquierda (pesar) se inicia un ciclo de pesada. Diseñe el multiplexor para las señales de las celdas de carga, con el reloj adecuado para ello. Cada **100ms** se debe hacer un ciclo de lecturas de las 4 celdas, que deben leerse a intervalos de **5ms** entre sí (figura de la derecha). Debe hacerse un total de 8 ciclos de lectura de las 4 celdas para cada pesada. Diseñe también el circuito que hace la temporización y el diagrama del programa que realiza la lectura.

D.- Asuma que los valores leídos de cada celda se guardan en 4 bloques de memoria en forma consecutiva. Escriba en assembler la rutina que tome los valores leídos de cada una de las 4 celdas, y los sume para encontrar el peso de los animales.

Punto A:

Parámetros del circuito de adaptación:

$$V_{FE,o} = \frac{Peso\ total}{Número\ de\ celdas} = \frac{Peso\ de\ los\ animales + Peso\ de\ la\ \ jaula}{Número\ de\ celdas} = \frac{4000+600}{4} = 1150\,[kg]$$

$$V_{LSB,o} = 1\,[kg]\,(dato)$$

$$m_o = \frac{V_{FE,o}}{V_{LSB,o}} = \frac{1150}{1} = 1150 = m_s$$

$$G_{sensor} = 5\,[\mu V / kg]\,(dato)$$

$$V_{FE,s} = G_{sensor} \cdot V_{FE,o} = 5\mu \cdot 1150 = 5{,}75\,[mV]$$

$$V_{LSB,s} = G_{sensor} \cdot V_{LSB,o} = 5\mu \cdot 1 = 5\,[\mu V]$$

$$V_{LSB} = \frac{V_R}{m} = \frac{V_R}{2^n} = \frac{10}{2^{12}} = 2{,}44\,[mV]$$

- $n = \dfrac{ln(m_s)}{ln(2)} = \dfrac{ln(1150)}{ln(2)} = 10{,}16 \qquad \rightarrow uso\ n = 12$

$$V_{FE} = V_{LSB} \cdot m_s = 2{,}44m \cdot 1150 \approx 2{,}81\,[V]$$

$$G_{adap} = \frac{V_{LSB}}{V_{LSB,s}} = \frac{2{,}44m}{5\mu} = 488$$

Como **100<G<1000** uso «adap3». Con esto, $\mathbf{R_g}$ es:

$$R_g = \frac{2 \cdot R_1}{G_{adap} - 1} = \frac{2 \cdot 100k}{488 - 1} \approx 411\,[\Omega]$$

Las señales de simulación $\mathbf{v_o}$, $\mathbf{v_s}$ y $\mathbf{v_{adap}}$ son:

$$v_o = V_{p,o} \cdot \sin\omega t + V_{OS,o} = (500 \cdot \sin\omega t + 650)\,[kg]$$

- $V_{p,o} = \frac{V_{o,max} - V_{o,min}}{2} = \frac{1150 - 150}{2} = 500\,[kg]$
- $V_{OS,o} = V_{o,min} + V_{p,o} = 150 + 500 = 650\,[kg]$

$$v_s = V_{p,s} \cdot \sin\omega t + V_{OS,s} = (2{,}5 \cdot \sin\omega t + 3{,}25)\,[mV]$$

- $V_{p,s} = V_{p,o} \cdot G_{sensor} = 500 \cdot 5\mu = 2{,}5\,[mV]$
- $V_{OS,s} = V_{OS,o} \cdot G_{sensor} = 650 \cdot 5\mu = 3{,}25\,[mV]$

$$v_{adap} = V_{p,adap} \cdot \sin\omega t + V_{OS,adap} = (1{,}22 \cdot \sin\omega t + 1{,}59)\,[V]$$

- $V_{p,adap} = V_{p,s} \cdot G_{adap} = 2{,}5m \cdot 488 = 1{,}22\,[V]$
- $V_{OS,adap} = V_{OS,s} \cdot G_{adap} = 3{,}25m \cdot 488 = 1{,}59\,[V]$

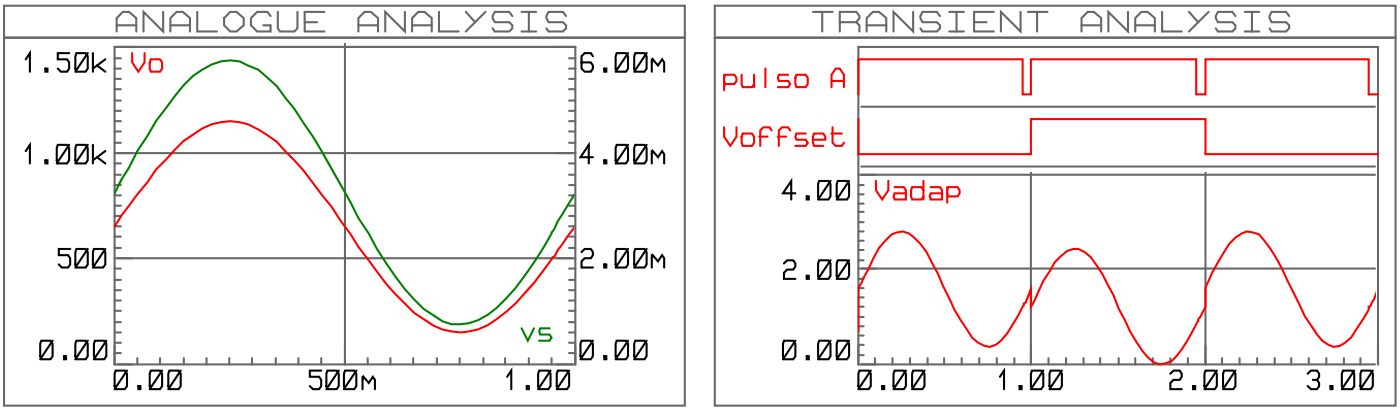


Figura 22: a) Formas de onda de $\mathbf{v_o}$ (eje izquierdo) y $\mathbf{v_s}$ (eje derecho); b) forma de onda de «pulso A», «$\mathbf{V_{offset}}$» y $\mathbf{v_{adap}}$.

Punto B (compensación en el circuito de adaptación):

Para restar el voltaje equivalente al peso de la jaula conecto $\mathbf{R_9}$ en «adap3» como en la Figura 23. Para lograr que $\mathbf{v_{adap}}$ arranque de **0V**, debo restarle $\mathbf{V_{adap,min}}$. Con esto saco $\mathbf{R_9}$:

$$R_9 = \frac{V_{CC} \cdot R_8}{V_{adap,min}} = \frac{V_{CC} \cdot R_8}{V_{OS,adap} - V_{p,adap}} = \frac{5 \cdot 10k}{1{,}59 - 1{,}22} \approx 135\,[k\Omega]$$

Con cada pulso negativo de A, $\mathbf{V_{offset}}$ cambia entre **0V** y **5V**; y $\mathbf{v_{adap}}$; entre $\mathbf{v_{adap}}$ y $\mathbf{v_{adap}}$**–0,37V**.

Punto C (8 trenes de pulsos de 4 mediciones cada uno):

La Figura 24 muestra el pulso B (que viene de un circuito parecido al del punto B), cuando éste se vuelve a poner en 1 (flanco ascendente) se pone en «1» el pin **Q** del primer flip-flop, Q (JFKK), y habilita el reloj 1, y se produce así la señal **clk1'**, que hace contar al primer 40161 en forma ascendente, cuenta que arranca de **0000**. Cuando las salidas **Q3-Q0** de este 40161 llega a **1000**, lo resetean a él y al primer flip-flop mediante las señales R1n y R1 respectivamente. Esto quiere decir que la señal **clk1'** cuenta 7 pulsos, el «octavo pulso» sale en la figura con muy corta duración porque apenas **Q3** se pone en «1» se resetea el primer flip-flop y deshabilita el **clk1**.

La **señal clk1'** actúa como reloj del segundo flip-flop que a su vez habilita el **clk2** y produce la señal **clk2'**. Esta señal hace que el segundo 40161 cuente de forma ascendente de **0000** hasta **0100**, cuando llega a este último valor se resetean el segundo flip-flop y el segundo 40161 del mismo modo que en el caso anterior.

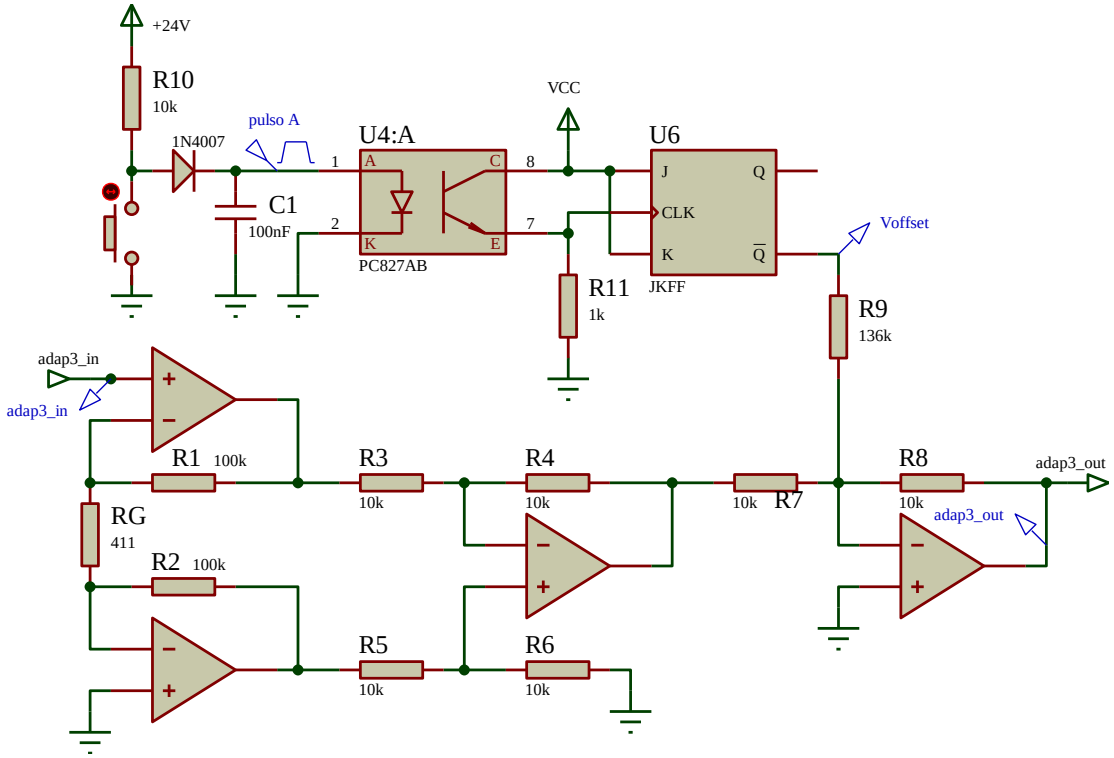


Figura 23: circuito adap3 y conexión del circuito para restar peso de la jaula.

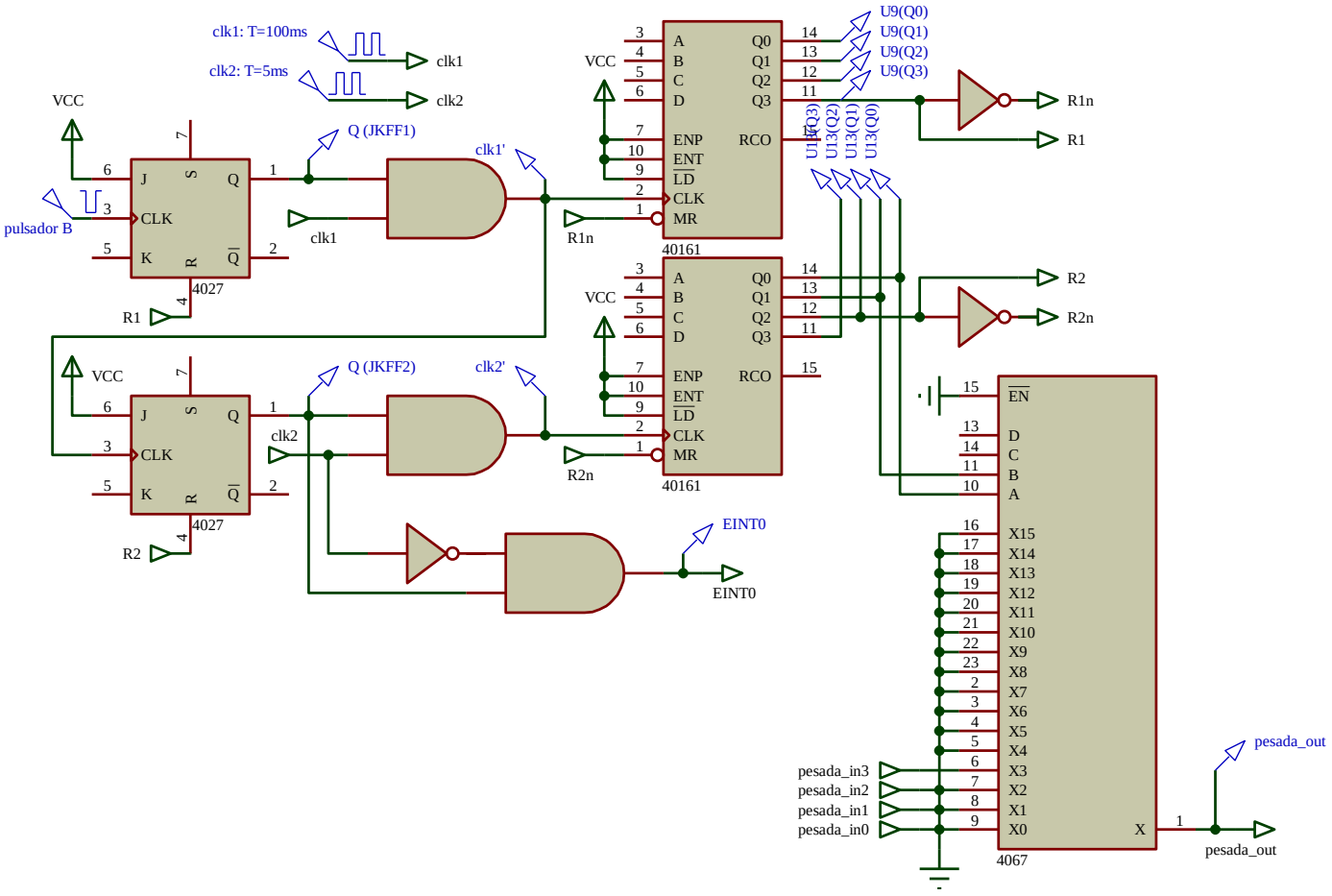


Figura 24: sub-circuito «PESADA».

La señal **clk1'** tiene 7 pulsos más un pulso de muy corta duración, este último pulso basta para hacer de clock del segundo flip-flop y producir un octavo tren de 4 pulsos en **clk2'**. En la Figura 25a se ven los 8 trenes de 4 pulsos de **clk2'**, que son las 8 tandas de 4 mediciones cada una que va a realizar el ARM. Cada uno de estos 8 trenes habilita las entradas **A** y **B** del multiplexor **4067** y se produce así la conexión entre cada una de las señales de los 4 sensores y la salida, **pesada_out**. En la Figura 25b), ampliación del eje del tiempo, se ven mejor los pulsos de **clk2'** y cómo la señal **pesada_out** iguala a la señal del sensor en el tiempo correspondiente.

Por último, en la Figura 25b) se aprecia la señal **EINT0**, que es la que va a la interrupción del ARM para indicarle que hay una señal a la entrada de **AIN0**.

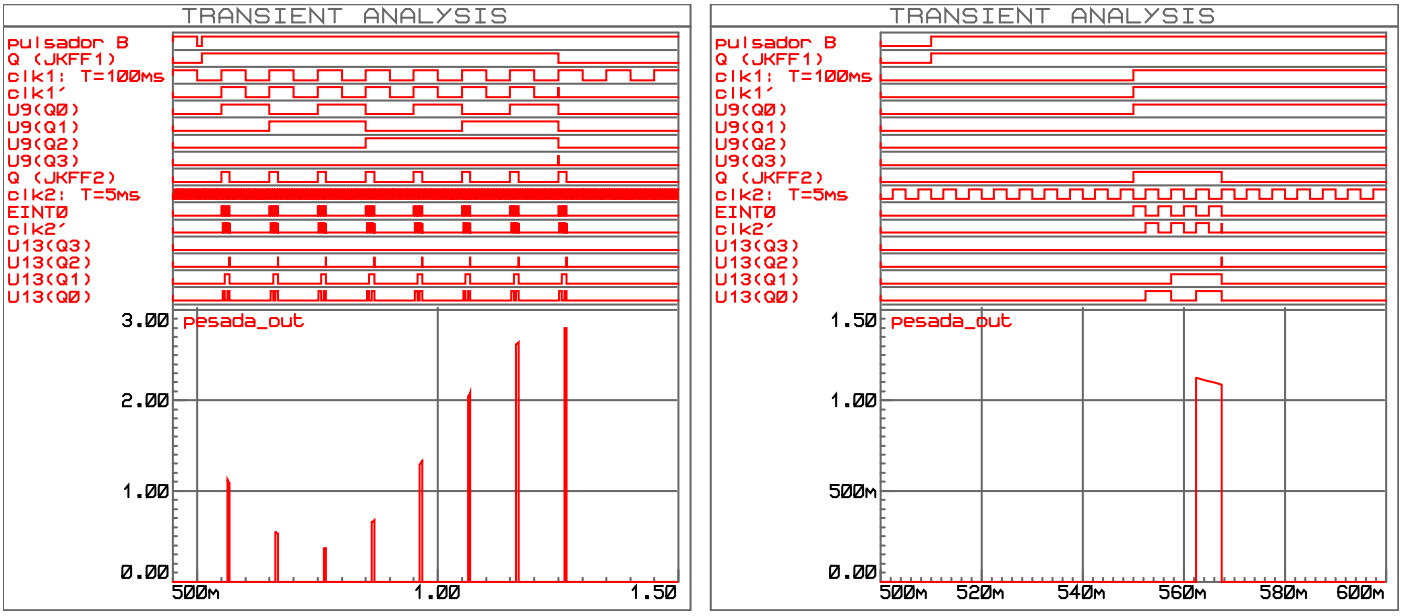


Figura 25: señales involucradas en MUX.

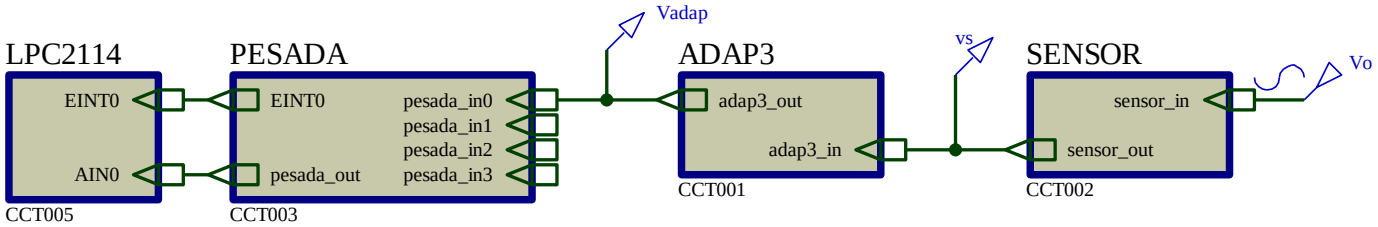


Figura 26: circuito completo.

Punto D:

```
1  reset :      mov R0,#VECT
2               mov R1,#0
3               mov R2,#5
4  carga:       subs r2,r2,#1
5               beq loop
6               ldr r3,[r0],#4
7               add r1,r1,r3
8  b carga
9  loop: b loop
10 VECT: .WORD 1,2,3,4
```

27 de julio

Usando un procesador basado en arquitectura ARM, resuelva los siguientes problemas:

A.- Un transductor genera información con una función de transferencia de **2,5µA/bit**. Diseñe la interfase del mismo hacia un conversor A/D de 12 bits, con referencia de **10V**. La señal es de continua.

A1.- Establezca las tolerancias de los componentes pasivos de su circuito para un error de **0,1%**.

A2.- Diseñe el hardware necesario para que se muestree esa señal a una tasa de **10Hz**, operando por interrupción. Debe incluir el reloj, el controlador de interrupción que use y debe detallar la configuración del mismo.

B.- Un conjunto de displays de LED (cátodo común) de 7 segmentos conectado a un puerto del microcontrolador mostrará el valor leído, el conjunto se opera de forma multiplexada, a una frecuencia de **50 [Hz]**.

B1.- ¿Cuántos dígitos debe usarse?

B2.- Dibuje el circuito asociado al display (un solo dígito). Establezca el conjunto de señales necesarias para que este display opere adecuadamente (reloj, líneas de control de flujo de datos, enables, etc.)

B3.- Escriba el diagrama de flujo del programa que controla la operación de su micro en relación a este display. De algún modo este diagrama debe incluir el clock de comando del sistema. Este clock se establece en **50 [Hz]**.

Punto A

Punto A1:

$$V_R = V_{FE} = 10\,[V]$$

$$m = 2^n = 2^{12} = 4096 = m_o = m_s$$

$$G_{adap} = \frac{V_{LSB}}{V_{LSB,s}} = \frac{2,44m}{100\mu} = 24,4$$

- $V_{LSB} = \frac{V_R}{2^n} = \frac{10}{2^{12}} = 2,44\,[mV]$
- $V_{LSB,s} = V_{LSB,o} \cdot G_{sensor} = 1[bit] \cdot \frac{100\,[\mu V]}{1[bit]} = 100\,[\mu V]$

Para obtener $\mathbf{G_{sensor}}$ uso el circuito de la Figura 20b, y cambio las resistencias hasta obtener $\mathbf{V_T=100\ [\mu V/bit]}$. De ahí que $\mathbf{G_{sensor}=100\ [\mu V/bit]}$. Como **10<Gadap<100**, uso «adap2», con esto R4 es:

$$R_4 = 1k \cdot G_{adap} = 1k \cdot 24,4 = 24,4\,[k\Omega]$$

La tolerancia de los componentes pasivos es:

$$E_{G,VR} = E_{G,VFE} = \frac{\varepsilon_G}{V_R} = \frac{8,17m}{10} = \boxed{817\,[ppm] = E_{G,VR}}$$

- $\varepsilon_G = \varepsilon_T - \left(\varepsilon_{ADC} + \varepsilon_{riple}\right) = 0,001 \cdot V_R - \frac{3 \cdot V_{LSB}}{4} = 0,001 \cdot 10 - \frac{3 \cdot 2,44m}{4} = 8,17\,[mV]$

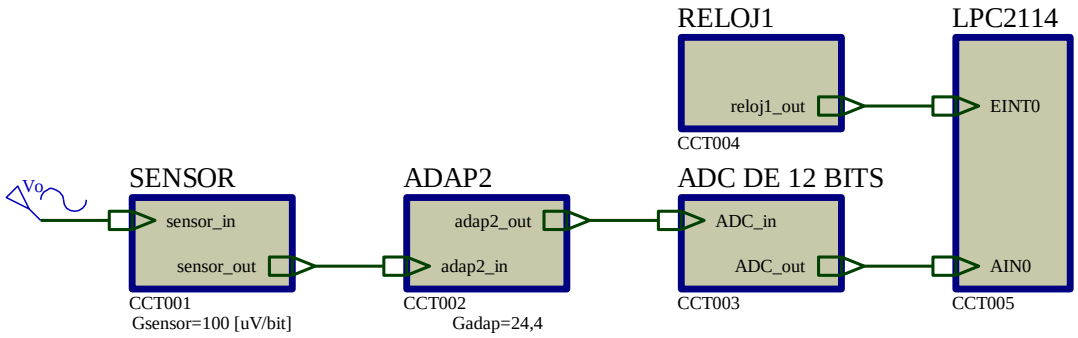
Punto A2:

La configuración del VIC es:

VICIntEnable=(1≪14)	asigna el canal 14 (EINT0) a la categoría FIQ.
VICIntSelect=(1≪14)	habilita el canal 14 en el VIC.
EXTMODE=(0≪0)	la interrupción EINT0 se activa por nivel.
EXTPOLAR=(1≪0)	la interrupción EINT0 se activa por un «1 lógico»

Los componentes del reloj son $\mathbf{R_B=68k\Omega}$, **C=1µF**, y con estos valores $\mathbf{R_A}$ vale:

$$R_A = \frac{1,44}{f \cdot C} - 2 \cdot R_B = \frac{1,44}{10 \cdot 1\mu} - 2 \cdot 68k = 8\,[k\Omega]$$

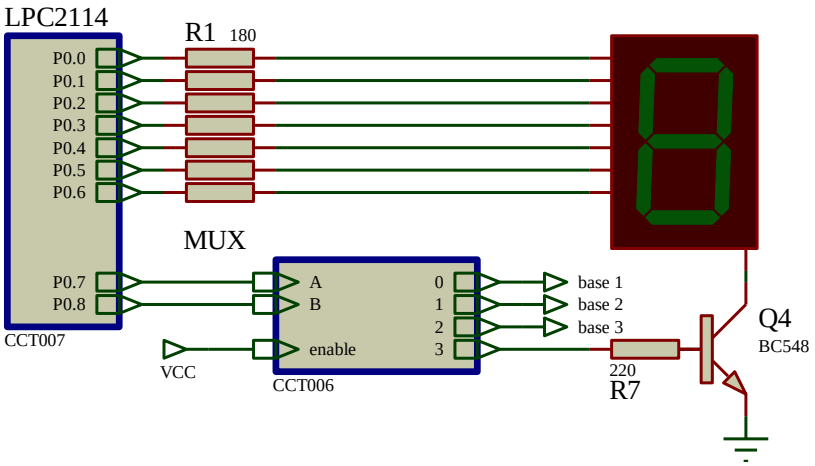


Punto B:

Punto B1:

La cantidad de pasos es m=4096, y para representar este número en dígitos hacen falta 4 displays.

Punto B2:



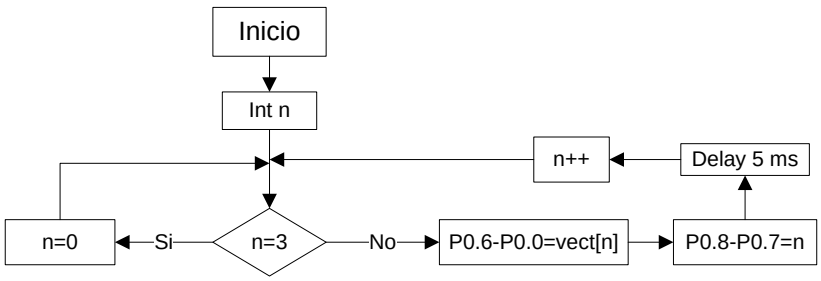
Los valores de **R1** (resistencia de 1 led) y el de **R7** (de 7 leds) son:

$$R_1 = \frac{V_{OH,P0.0} - foward\ voltage}{segment\ on\ current} = \frac{3{,}3 - 1{,}5}{10m} = 180\,[\Omega]$$

$$R_7 = \frac{V_{OH,mux}}{\frac{I_C}{\beta}} = \frac{V_{OH,mux}}{\frac{7 \cdot segment\ on\ current}{\beta}} = \frac{5}{\frac{7 \cdot 10m}{100}} \approx 6{,}8\,[k\Omega]$$

Las salidas **P0.6-P0.0** llevan el dato ya convertido correspondiente al primer dígito, a la vez que las salidas **P0.8-P0.9=00**, lo que hace que MUX active la salida «base 1». Pasado un tiempo, el soft se encarga de poner las salidas **P0.6-P0.0** con el valor del segundo dígito, pero el soft también hace que **P0.6-P0.0=01**, lo cual con lo cual MUX pone en alto «base 2» ahora. Y así hasta llegar al cuarto dígito, luego vuelve al primero.

Punto B3:



Anexo A: Registros de interrupciones del ARM y algunas tablas

Registros de GPIO y de interrupciones del LPC2114

Registros de GPIO (General Prupose Input Output) IO0DIR e IO1DIR

IO0DIR e IO1DIR: Su contenido indica qué pines GPIO se comportarán como entradas y qué pines como salidas. Tiene 32 bits, cada bit corresponde a uno de los 32 pines del puerto en cuestión. Los bits 0-31 de IO0DIR controlan los 32 pines del puerto 0, los bits 16-31 controlan los 16 pines del pueto 1.

1 configura un pin como salida;

0 configura un pin como entrada.

IO0SET e IO1SET: Controla la puesta en alto de los pines de cada puerto; no puede hacer nada para ponerlos en bajo. Por ejemplo, escribir un «1 lógico» en el bit 29 de IO1SET significa poner en alto el pin P1.29. Un «0 lógico» en este registro no tiene efecto en ningún pin de nigún puerto.

IO0CLR e IO1CLR: Controla la puesta en bajo de los pines de cada puerto; no puede hacer nada para ponerlos en alto. Por ejemplo, escribir un «1 lógico» en el bit 3 de IO0SET significa poner en bajo el pin P0.3. Un «0 lógico» en este registro no tiene efecto en ningún pin de nigún puerto.

IO0PIN e IO1PIN: Permite ver el contenido de los puertos (modo entrada) o determinar su salida lógica (modo salida). Por ejemplo, si el bit 9 del registro IO0PIN está en alto, significa que P0.9 está conectado a 3,3V (modo entrada) o que va a generar sí o sí en P0.9 un voltaje de 3,3V (modo salida).

Registros de interrupciones

VICIntSelect: Asigna la interrupción requerida a la categoría FIQ con un «1 lógico»; o a la categoría IRQ con un «0 lógico». Cada uno de los 32 bits de este registro se corresponde con un canal del mapa de canales.

VICIntEnable: Habilita las interrupciones de un determinado canal (el cual ya se ha definido como IRQ o IFQ) para que las «atienda» VIC. Al igual que VICIntSelect, cada uno de los 32 bits de este registro se corresponde con un canal del mapa de canales. Un «0 lógico» no tiene efecto en este registro, no deshabilita las interrupciones de un canal, para eso está VICIntClear.

VICIntClear: Es el complmenteo del registro VICIntEnable, pues con un «1 lógico» deshabilita las interrupciones de un determinado canal; un «0 lógico» no tiene efecto en este registro.

VICVectCntlX: Le asigna a un determinado canal (bits 4−0, donde 0000 es el canal 1 y 1111 el canal 31) la categoría de vectorizado (con un «1 lógico» en el bit 5) o no vectorizado (con un «0 lógico» en el bit 5). La X es un número que indica la prioridad de la interrupción (0 para la más alta prioridad hasta 15 la más baja).

Bit 5: 1 para la categoría de vectorizado.

0 para la categoría de no vectorizado.

Bits 4-0: 0.0000 para el canal 0 del mapa de canales (watchdog interrupt).

1.1111 para el canal 31 del mapa de canales (CAN reserved).

VICVectAddrX: Se guarda la rutina (ISR) que se va a ejecutar cuando se produzca la interrupción de prioridad X (número que puede ir de 0 a 15).

EXTMODE: activa por nivel o por flanco la interrupción externa.

0 → activado por nivel

1 → activado por flaco

EXTPOLAR: activa por nivel bajo o alto, o por flanco descendente o ascendente, depende de EXTMODE.

0 → si EXTMODE=0 → activado por un «0 lógico»

Si EXTMODE=1 → activado por flanco descendente.

1 → si EXTMODE=0 → activado por un «1 lógico»

Si EXTMODE=1 → activado por flanco ascendente.

Algunas tablas sacadas de la hoja de datos del LPC2114

Table 31: Power Control for Peripherals Register for LPC2119/2129/2292 (PCONP - 0xE01FC0C4)

PCONP	Function	Description	Reset Value
0	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0
1	PCTIM0	When 1, TIMER0 is enabled. When 0, TIMER0 is disabled to conserve power.	1
2	PCTIM1	When 1, TIMER1 is enabled. When 0, TIMER1 is disabled to conserve power.	1
3	PCURT0	When 1, UART0 is enabled. When 0, UART0 is disabled to conserve power.	1
4	PCURT1	When 1, UART1 is enabled. When 0, UART1 is disabled to conserve power.	1
5	PCPWM0	When 1, PWM0 is enabled. When 0, PWM0 is disabled to conserve power.	1
6	Reserved	User software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0
7	PCI2C	When 1, the I^2C interface is enabled. When 0, the I^2C interface is disabled to conserve power.	1
8	PCSPI0	When 1, the SPI0 interface is enabled. When 0, the SPI0 is disabled to conserve power.	1
9	PCRTC	When 1, the RTC is enabled. When 0, the RTC is disabled to conserve power.	1
10	PCSPI1	When 1, the SPI1 interface is enabled. When 0, the SPI1 is disabled to conserve power.	1
11	Reserved	User software should write 0 here to reduce power consumption.	1
12	PCAD	When 1, the A/D converter is enabled. When 0, the A/D is disabled to conserve power.	1
13	PCCAN1	When 1, CAN Controller 1 is enabled. When 0, it is disabled to save power. **Note:** the Acceptance Filter is enabled if any of CAN Controllers 1-2 is enabled.	1
14	PCCAN2	When 1, CAN Controller 2 is enabled. When 0, it is disabled to save power.	1
31:15	Reserved	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Tabla 1: PCONP. Este registro habilita la alimentación para los diferentes periféricos.

PINSEL0	Pin Name	Function when 00	Function when 01	Function when 10	Function when 11	Reset Value
1:0	P0.0	GPIO Port 0.0	TxD (UART0)	PWM1	Reserved	00
3:2	P0.1	GPIO Port 0.1	RxD (UART0)	PWM3	EINT0	00
5:4	P0.2	GPIO Port 0.2	SCL (I^2C)	Capture 0.0 (TIMER0)	Reserved	00
7:6	P0.3	GPIO Port 0.3	SDA (I^2C)	Match 0.0 (TIMER0)	EINT1	00
9:8	P0.4	GPIO Port 0.4	SCK (SPI0)	Capture 0.1 (TIMER0)	Reserved	00
11:10	P0.5	GPIO Port 0.5	MISO (SPI0)	Match 0.1 (TIMER0)	Reserved	00
13:12	P0.6	GPIO Port 0.6	MOSI (SPI0)	Capture 0.2 (TIMER0)	Reserved	00
15:14	P0.7	GPIO Port 0.7	SSEL (SPI0)	PWM2	EINT2	00
17:16	P0.8	GPIO Port 0.8	TxD UART1	PWM4	Reserved	00
19:18	P0.9	GPIO Port 0.9	RxD (UART1)	PWM6	EINT3	00
21:20	P0.10	GPIO Port 0.10	RTS (UART1)	Capture 1.0 (TIMER1)	Reserved	00
23:22	P0.11	GPIO Port 0.11	CTS (UART1)	Capture 1.1 (TIMER1)	Reserved	00
25:24	P0.12	GPIO Port 0.12	DSR (UART1)	Match 1.0 (TIMER1)	Reserved	00
27:26	P0.13	GPIO Port 0.13	DTR (UART1)	Match 1.1 (TIMER1)	Reserved	00
29:28	P0.14	GPIO Port 0.14	CD (UART1)	EINT1	Reserved	00
31:30	P0.15	GPIO Port 0.15	RI (UART1)	EINT2	Reserved	00

PINSEL1	Pin Name	Function when 00	Function when 01	Function when 10	Function when 11	Reset Value
1:0	P0.16	GPIO Port 0.16	EINT0	Match 0.2 (TIMER0)	Capture 0.2 (TIMER0)	00
3:2	P0.17	GPIO Port 0.17	Capture 1.2 (TIMER1)	SCK (SPI1)	Match 1.2 (TIMER1)	00
5:4	P0.18	GPIO Port 0.18	Capture 1.3 (TIMER1)	MISO (SPI1)	Match 1.3 (TIMER1)	00
7:6	P0.19	GPIO Port 0.19	Match 1.2 (TIMER1)	MOSI (SPI1)	Match 1.3 (TIMER1)	00
9:8	P0.20	GPIO Port 0.20	Match 1.3 (TIMER1)	SSEL (SPI1)	EINT3	00
11:10	P0.21	GPIO Port 0.21	PWM5	Reserved	Capture 1.3 (TIMER1)	00
13:12	P0.22	GPIO Port 0.22	Reserved	Capture 0.0 (TIMER0)	Match 0.0 (TIMER0)	00
15:14	P0.23	GPIO Port 0.23	RD2 (CAN Controller 2)	Reserved	Reserved	00
17:16	P0.24	GPIO Port 0.24	TD2 (CAN Controller 2)	Reserved	Reserved	00
19:18	P0.25	GPIO Port 0.25	RD1 (CAN Controller 1)	Reserved	Reserved	00
21:20	P0.26	Reserved				00
23:22	P0.27	GPIO Port 0.27	AIN0 (A/D Converter)	Capture 0.1 (TIMER0)	Match 0.1 (TIMER0)	01
25:24	P0.28	GPIO Port 0.28	AIN1 (A/D Converter)	Capture 0.2 (TIMER0)	Match 0.2 (TIMER0)	01
27:26	P0.29	GPIO Port 0.29	AIN2 (A/D Converter)	Capture 0.3 (TIMER0)	Match 0.3 (TIMER0)	01
29:28	P0.30	GPIO Port 0.30	AIN3 (A/D Converter)	EINT3	Capture 0.0 (TIMER0)	01
31:30	P0.31	Reserved				00

Tabla 2: PINSEL0 y PINSEL1, este registro controla cada una de las entradas P0.0-P0.31 del ARM, por defecto vale 0, así que está configurado para que todos estos pines funcionen como GPIO apenas arranca el programa.

Block	Flag(s)	VIC Channel #
WDT	Watchdog Interrupt (WDINT)	0
-	Reserved for software interrupts only	1
ARM Core	Embedded ICE, DbgCommRx	2
ARM Core	Embedded ICE, DbgCommTx	3
TIMER0	Match 0 - 3 (MR0, MR1, MR2, MR3) Capture 0 - 3 (CR0, CR1, CR2, CR3)	4
TIMER1	Match 0 - 3 (MR0, MR1, MR2, MR3) Capture 0 - 3 (CR0, CR1, CR2, CR3)	5
UART0	Rx Line Status (RLS) Transmit Holding Register Empty (THRE) Rx Data Available (RDA) Character Time-out Indicator (CTI)	6
UART1	Rx Line Status (RLS) Transmit Holding Register Empty (THRE) Rx Data Available (RDA) Character Time-out Indicator (CTI) Modem Status Interrupt (MSI)	7
PWM0	Match 0 - 6 (MR0, MR1, MR2, MR3, MR4, MR5, MR6)	8
I^2C	SI (state change)	9
SPI0	SPI Interrupt Flag (SPIF) Mode Fault (MODF)	10
SPI1	SPI Interrupt Flag (SPIF) Mode Fault (MODF)	11
PLL	PLL Lock (PLOCK)	12
RTC	Counter Increment (RTCCIF) Alarm (RTCALF)	13
System Control	External Interrupt 0 (EINT0)	14
System Control	External Interrupt 1 (EINT1)	15
System Control	External Interrupt 2 (EINT2)	16
System Control	External Interrupt 3 (EINT3)	17
A/D	A/D Converter	18
CAN	CAN and Acceptance Filter (1 ORed CAN, LUTerr int) CAN1 Tx CAN2 Tx CAN3 Tx (LPC2194/2292/2294 only, otherwise Reserved) CAN4 Tx (LPC2194/2292/2294 only, otherwise Reserved) Reserved CAN1 Rx CAN2 Rx CAN3 Rx (LPC2194/2292/2294 only, otherwise Reserved) CAN4 Rx (LPC2194/2292/2294 only, otherwise Reserved) Reserved	19 20 21 22 23 24-25 26 27 28 29 30-31

Tabla 3: Mapa de canales para el control de los periféricos.

Anexo B: Sub-circuitos

1. Adaptación

La adaptación de la señal de entrada la realizamos con amplificadores operacionales, y la configuración depende del valor de ganancia que calculemos. Tengo que:

- G<10: uso el amplificador no inversor.
- 10<G<100: uso el amplificador con dos etapas en configuración inversora.
- 100<G<1k: uso el amplificador de instrumentación formado por 4 operacionales.
- 1k<G: uso un amplificador de instrumentación integrado en un solo chip.

1.1 - Adaptación con un amplificador no inversor (G<10, adap1)

La ganancia de este circuito es:

$$G = \frac{v_{salida}}{v_{entrada}} = \frac{adapt1\_out}{adap1\_in} = 1 + \frac{R_2}{R_1}$$

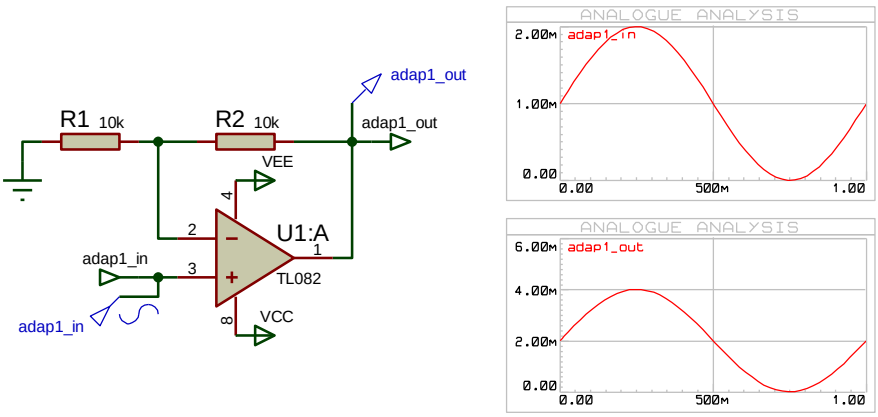


Figura 27: Amplificador de instrumentación.

1.2 - Adaptación con dos etapas inversoras (10<G<100, adapt2)

La ganancia de este circuito es:

$$G = \frac{v_{salida}}{v_{entrada}} = \frac{adapt2\_out}{adap2\_in} = G_1 \cdot G_2 = \left(-\frac{R_4}{R_3}\right) \cdot \left(-\frac{R_2}{R_1}\right) = \frac{R_4 \cdot R_2}{R_3 \cdot R_1}$$

Elijo **R_1=1kΩ** y **R_2=10kΩ** para hacer que **G_1=10**, y elijo **R_3=10kΩ**, con esto **R_4** es:

$$\therefore R_4 = \frac{G_{adap} \cdot R_3 \cdot R_1}{R_2} = \frac{G_{adap} \cdot 10k \cdot 1k}{10k} = 1000 \cdot G_{adap}$$

Si la señal de entrada tiene valores negativos, entonces hará falta conectar **R_5** a **V_R=+15V**. A la resistencia **R_5** la calculo como:

$$R_5 = \frac{R_4 \cdot V_{ref}}{A}$$

Donde **A** es la amplitud que hay que sumar a la señal **v_s** según convenga.

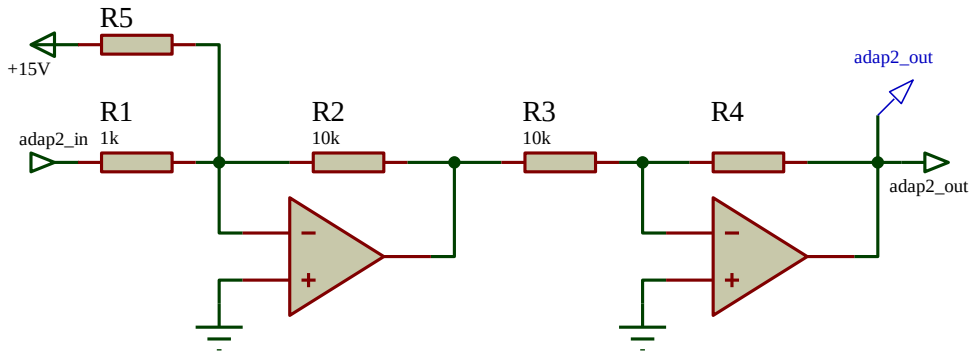


Figura 28: Amplificador de dos etapas inversoras.

1.3 - Adaptación con un amplificador de instrumentación (100<G<1k, adapt3)

La ganancia de este circuito es:

$$G = \frac{v_{salida}}{v_{entrada}} = \frac{adap3\_out}{adap3\_in} = 1 + \frac{2 \cdot R_1}{R_g}$$

La señal «adap3_in» no es sinusoidal, así como tampoco son ninguna de las señales de prueba de los dos casos anteriores, pues son las muestras de la señal de entrada no adaptada. Solamente la hago sinusoidal para simular valores máximo y mínimo y poder ver la salida.

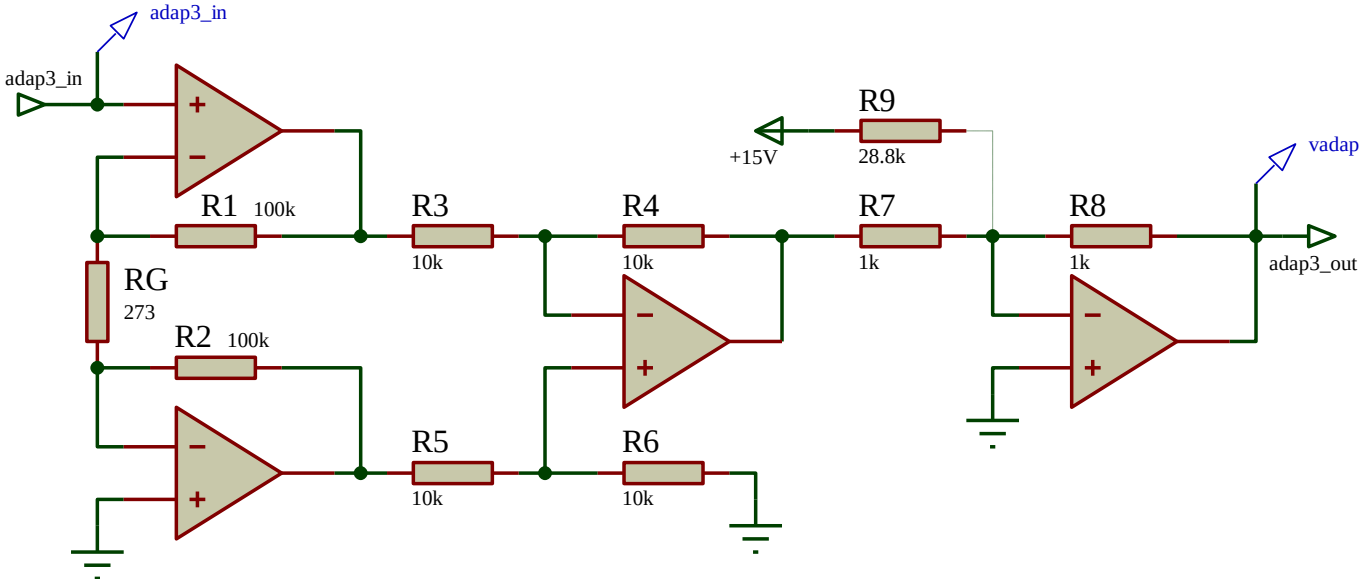


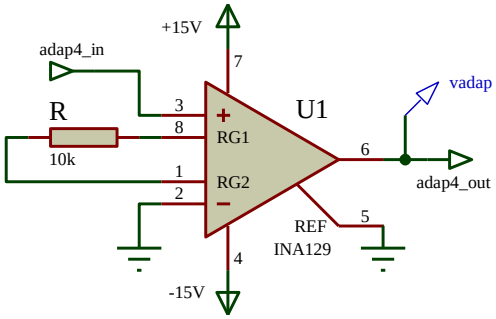
Figura 29: Amplificador de instrumentación.

Si necesito tensión de offset a la salida, conecto la línea de puntos. Elijo **R_{10}=1kΩ**, entonces **R_9** vale:

$$R_9 = \frac{R_{10} \cdot V_{ref}}{V_{offset}}$$

Donde **V_{ref}** es la tensión usada como referencia, en este caso vale **+15V**, pues necesito «bajar» el voltaje de salida; en caso de necesitar «subirla», uso **–15V**. Y **V_{offset}** es la tensión que tengo que sumarle o restarle a la salida según necesite.

1.4 - Adaptación con un amplificador de instrumentación integrado (1k<G, adap4)



La ganancia de este circuito es:

$$G = \frac{v_{salida}}{v_{entrada}} = \frac{adap4\_out}{adap4\_in} = 1 + \frac{50k\Omega}{R}$$

2. Display

2.1 - Cuatro displays no multiplexados (display1)

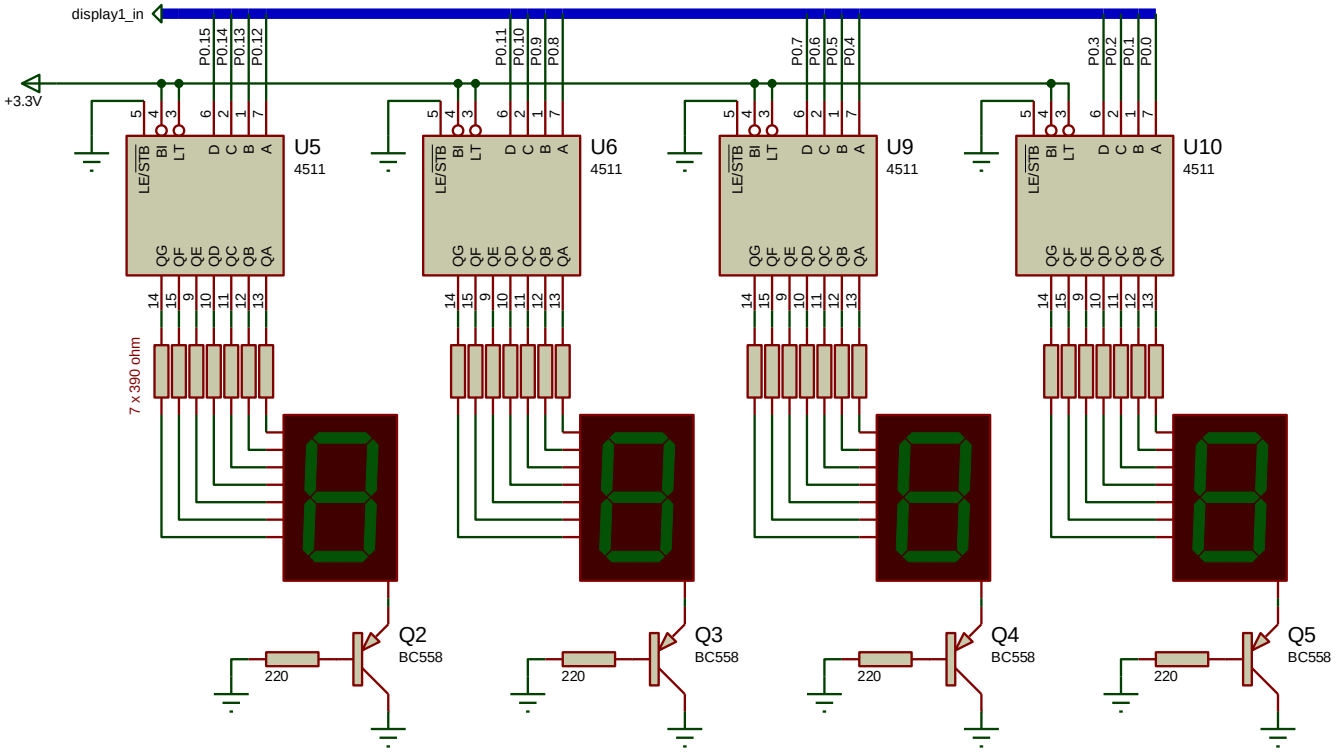


Figura 30: 4 displays no multiplexados.

2.2 - Cuatro displays multiplexados (display2)

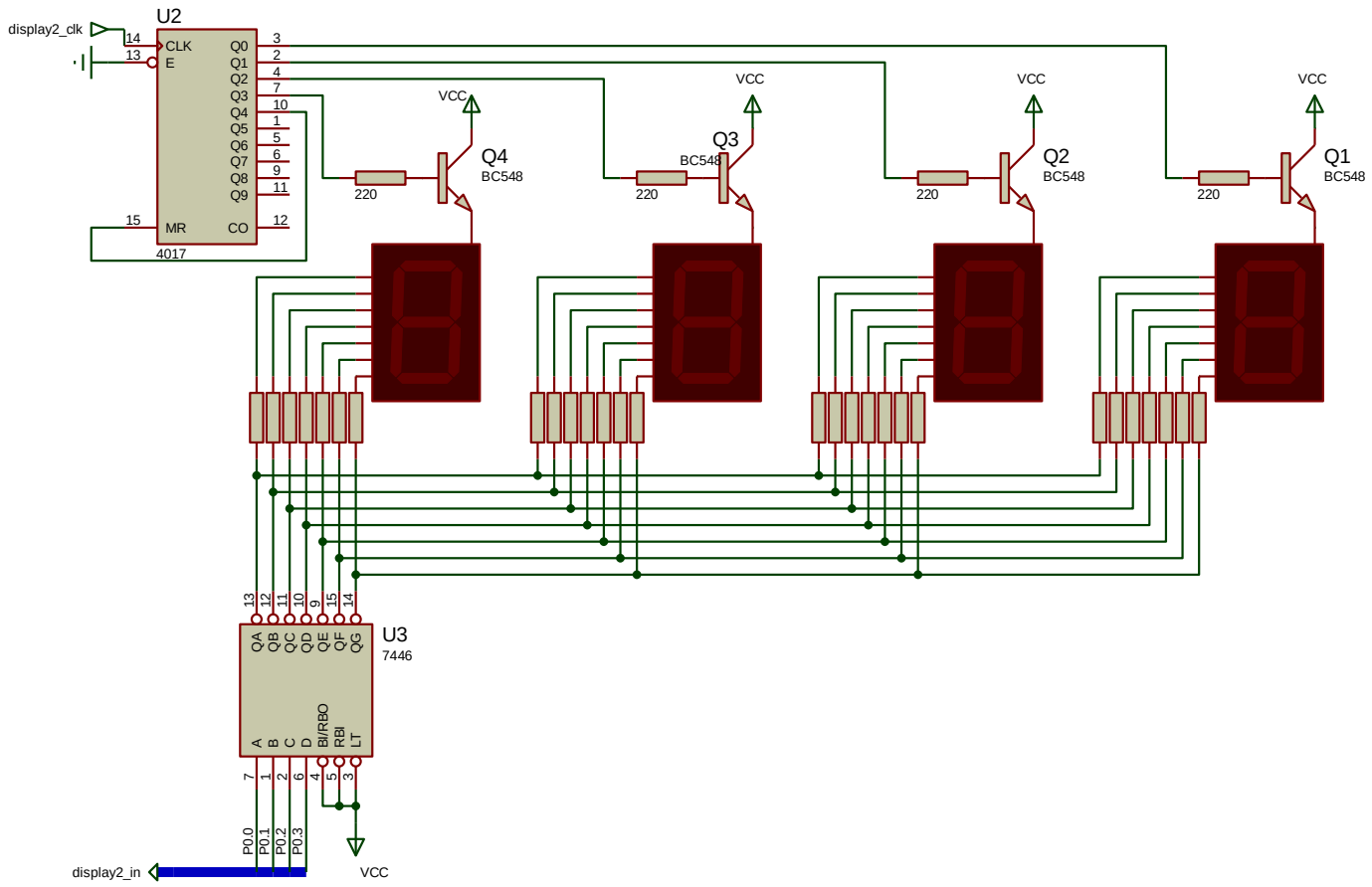
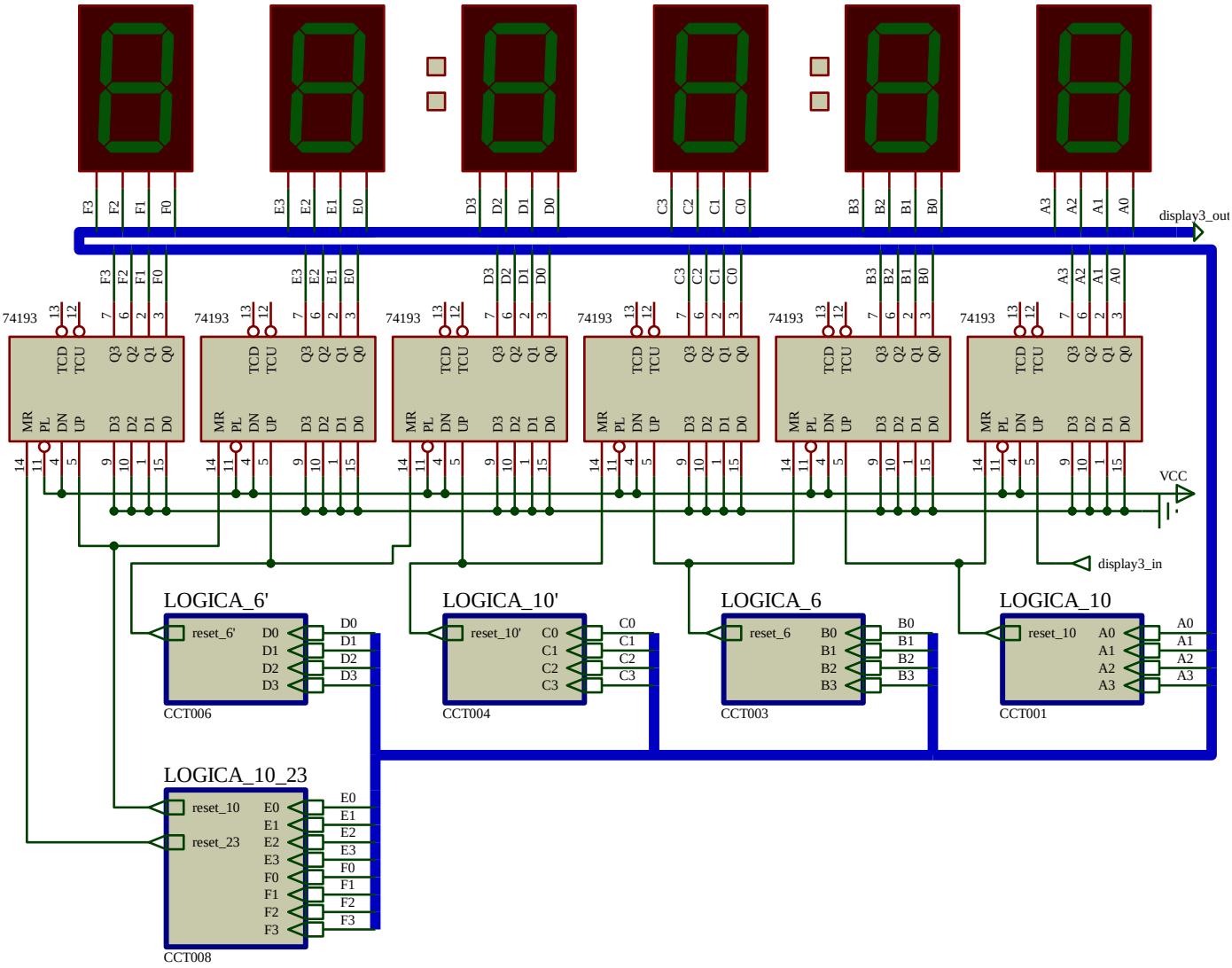


Figura 31: 4 displays multiplexados..

2.3 – 6 displays que dan la hora en formato hh:mm:ss (display3)

El sub-circuito «lógica_10» pone un 1 en «reset_10» cuando **A3-A0=10_{10}=1010_2**. Lo mismo para con «lógica_6» cuando **B3-B0=6_{10}**. Los otros dos son iguales, el apóstrofe lo puse porque Proteus no me permite usar los mismos nombres. Con respecto a «lógica_10_23» tira un 1 en «reset_10» cuando **F3-F0 E3-E0=10_{10}**, y un 1 por «reset_23» cuando **F3-F0 E3-E0=23_{10}**.



3. Reloj1

Uso un 555 en modo astable. La fórmula para obtener la frecuencia de oscilación del 555 como astable es:

$$f_{osc} = \frac{1{,}44}{(R_A + 2R_B)C}$$

En este caso uso una frecuencia de trabajo **f_{osc}=0,1 [Hz]** (pues la interrupción debe ocurrir cada **10 [seg]**) elijo **C=10 [µF]** y **R_B=680 [kΩ]**, me queda:

$$R_A = \frac{1{,}44}{f_{osc} \cdot C} - 2R_B = \frac{1{,}44}{0{,}1 \cdot 10\mu} - 2 \cdot 680k = 80\,[k\Omega]$$

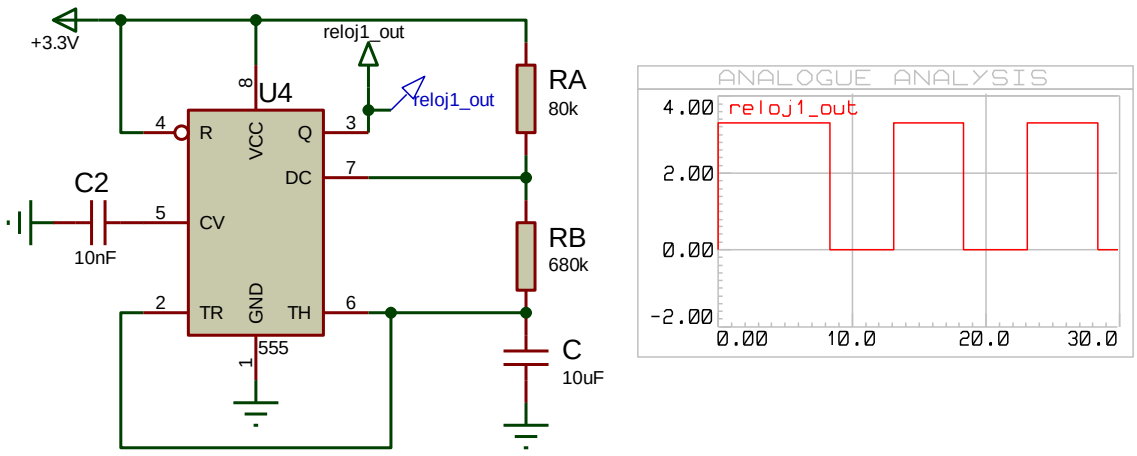


Figura 32: 555 configurado como astable para generar una señal de clock.