0 a 9999 Multiplexado de Displays 7 segmentos

#ifdef \_\_USE\_CMSIS

#include "LPC17xx.h"

#endif

void muestraValor ();

void descompone();

void configuraTimer0();

void configuraIO();

int tabla[10]={ 0b11111100, //0

0b01100000, //1

0b11011010, //2

0b11110010, //3

0b01100110, //4

0b10110110, //5

0b10111110, //6

0b11100000, //7

0b11111110, //8

0b11100110};//9

int display=0; //Registra que decimal se esta mostrando

int cuenta =0; //Lleva la cuenta propiamente dicha

int unidad=2;

int decena=0;

int centena=1;

int uMil=5;

int segundos = 20000000;

int multiplex = 700501;

int main(void) {

configuraTimer0(); //Timer0 se usa para llevar la cuenta y para el multiplexado.

configuraIO();

while(1) {

}

return 0 ;

}

void descompone (){

unidad = cuenta % 10;

decena = (cuenta /10) %10;

centena =(cuenta /100) %10;

uMil = (cuenta /1000) %10;

}

void configuraTimer0(){

LPC\_SC->PCONP |= 1 << 1; // Activar Timer 0

LPC\_TIM0->MR0 = segundos; // Valor de MR0. Usado para incrementar la cuenta

LPC\_TIM0->MR1 = multiplex;//Valor de MR1. Usado para el multiplexado del display

LPC\_TIM0->MCR = 0b1011; // Bit0: Interrup para MR0=TC

// Bit1: Reset TC si MR0=TC

// Bit3: Interrup para MR1=TC

NVIC\_EnableIRQ(TIMER0\_IRQn); // Activar Interrupcion por Timer0

LPC\_TIM0->TCR |= 1 << 0; // Timer0 en modo Contador - pag 494

}

void configuraIO(){

//P0 se usa para el multiplexado de displays

LPC\_GPIO0->FIODIR |= (1 << 23); // P0[21] COMO SALIDA

LPC\_GPIO0->FIODIR |= (1 << 24); // P0[22] COMO SALIDA

LPC\_GPIO0->FIODIR |= (1 << 25); // P0[27] COMO SALIDA

LPC\_GPIO0->FIODIR |= (1 << 26); // P0[28] COMO SALIDA

LPC\_GPIO2->FIODIR |= (0x000000FF<<0);

//p2[0]-p2[7] como salida. Se usa para los LEDs del display

}

/\*Rutina de interrupción del TIMER0\*/

void TIMER0\_IRQHandler(void){

//Preguntamos si la interrupcion fue por TC=MR0

If (LPC\_TIM0->TC == segundos){

cuenta ++;

if (cuenta==10000){

cuenta=0;

}

//descompone();

LPC\_TIM0->IR |= 0b1; //Bajamos la bandera de interrupcion por MR0

}

else { //if(LPC\_TIM0->TC==multiplex) //Si se llega hasta aca, es por que la interrupcion se produjo por TC=MR1

LPC\_TIM0->IR |= 0b10; //Bajamos la bandera de interrupcion por MR1

display++;

if (display ==4){

display=0;

} // Fin del if

muestraValor();

} Fin del else

} // Fin de la función (Handler del TIMER0)

void muestraValor (){

LPC\_GPIO0->FIOCLR |= (1<<23); //Apago display conectado en p0[21]

LPC\_GPIO0->FIOCLR |= (1<<24); //Apago display conectado en p0[22]

LPC\_GPIO0->FIOCLR |= (1<<25); //Apago display conectado en p0[27]

LPC\_GPIO0->FIOCLR |= (1<<26); //Apago display conectado en p0[28]

switch (display){ //////

case 0:

LPC\_GPIO0->FIOSET |= (1<<23); //Prendo display conectado en po[21]

LPC\_GPIO2->FIOSET = tabla [(unidad)<<0];//Muestro la unidad

break;

case 1:

LPC\_GPIO0->FIOSET |= (1<<24);//Prendo display conectado en po[22]

LPC\_GPIO2->FIOSET = tabla [(decena)<<0];//Muestro la decena

break;

case 2:

LPC\_GPIO0->FIOSET |= (1<<25); //Prendo display conectado en po[27]

LPC\_GPIO2->FIOSET = tabla [(centena)<<0]; //Muestro la centena

break;

case 3:

LPC\_GPIO0->FIOSET |= (1<<26); //Prendo display conectado en po[28]

LPC\_GPIO2->FIOSET = tabla [(uMil)<<0]; //Muestro la unidad de mil

break;

default:

break;

} //Fin del switch

}//Fin de la función muestraValor

TMM

#include "LPC17xx.h"

int main (void)

{LPC\_SC->PCONP |= 1 << 1; // Activar Timer 0

LPC\_TIM0->MR0 = 2000000; // Valor de MR0 que será comparado con TC

LPC\_TIM0->MCR = 3; // Bit0: Interrup para MR0=TC; Bit1: Reset TC

NVIC\_EnableIRQ(TIMER0\_IRQn); // Activar Interrupcion por Timer0

LPC\_TIM0->TCR |= 1 << 0; // Timer0 en modo Contador - pag 494

LPC\_GPIO0->FIODIR |= (1 << 22); // P0[22] COMO SALIDA

int cuenta = 0;

int tabla[10] ={};

while (1) {}

}

/\* funcion para atender la interrupcion \*/

void TIMER0\_IRQHandler (void)

{ cuenta++;

if (cuenta ==10){

cuenta=0;

}

LPC\_GPIO2->FIOCLR = (256<<0);

LPC\_GPIO2->FIOSET = (tabla<<0);

if((LPC\_TIM0->IR & 0x01) == 0x01) // si hay interrupcion por MR0

{ LPC\_TIM0->IR |= 1 << 0; // resetear MR0 interrupt - pag 493

If (LPC\_GPIO0->FIOPIN &(1<<22))

LPC\_GPIO0->FIOCLR = (1 << 22); // P0[22] = 0

else

LPC\_GPIO0->FIOSET = (1 << 22); // P0[22] = 1

} }

SERIE

auto baudaje y DMA no los usamos

para receptor, el pull down debe estar DESACTIVADO <---------

el clock del uart es 16 veces mayor que la de transmicion (no hace falta configurar, lo pongo para que lo tenga en cuenta al momento de leer la formula. esto es a los fines de que cuente a la "mitad" de un dato, y no en un flanco)

------------------------------------------------

Pasos:

\*encender periferico UART3(pconp) (pag 63)

\*configurar los clock (clock del sistema que alimenta al periferic)PCLK\_UART3 pag 57

\*dividir frecuencia (pag 57)

\*definir como va a ser la trama de transmicion (registro LSR, pag 307)

\*configurar velocidad de transmision "baud rate". en el ejemplo no se usa la parte fraccionaria.UARTn divisor latch pag 301

\*fijar la velocidad para que nadie lo pueda modificar

\*habilitar interrupcion (UARTn interrupt enable. p303)

\*seleccionar modo de pines con PINSEL

\*configurar systick para que en sus interrupciones, envia el dato

\*ver el uartn reciver buffer

ca3046: transistores integrados

TIMER

clk ppal!= clock rc default

clk main debe ser seteado por software. ante reset, se activa el RC, NO el clk main

Pll permite frecuencia más alta que la de clock. es un VCO. Hay 3 fuentes de clk: RC, oscilador externo y real time clk. Diagrama en bloques del clk en p 29. p36: los valores de n y m se proveen por software Atencion p38. Fcco NO es el valor de salida para el sistema

p43, ejemplos de calculos de clock.

**Ejercicio de Timer**

#include "LPC17xx.h"

#ifdef \_\_USE\_CMSIS

#endif

int cuenta = 0;

int tabla[10] = {0X3F,

0X06,

0X5B,

0X4F,

0X66,

0X6D,

0X7D,

0X07,

0X7F,

0X67

};/\*{0b00000001,

0b00000010,

0b00000100,

0b00001000,

0b00010000,

0b00100000,

0b01000000,

0b10000000,

0b11111111,

0b00000000};

\*/

/\*{0b00111111,

0b00000110,

0b01011011,

0b01001111,

0b01100110,

0b01101101,

0b01111100,

0b00000111,

0b01111111,

0b01100111}

\* {0X3F,

0X06,

0X5B,

0X4F,

0X66,

0X6D,

0X7D,

0X07,

0X7F,

0X67

};\*/

/\*{0b00000000,

0b11111100,

0b01100000,

0b11011010,

0b11110010,

0b01100110,

0b10110110,

0b10111110,

0b11100000,

0b11111110};\*/

/\*{0x0000000F,

0X0000000F,

0x0000000F,

0x0000000F,

0x0000000F,

0b11110000,

0b11110000,

0b11110000,

0b11110000,

0b11110000,

0b11110000};\*/

int main (void)

{

LPC\_SC->PCONP |= 1 << 1; // Activar Timer 0

LPC\_TIM0->MR0 = 20000000; // Valor de MR0 que será comparado con TC 2000000

LPC\_TIM0->MCR = 3; // Bit0: Interrup para MR0=TC; Bit1: Reset TC

NVIC\_EnableIRQ(TIMER0\_IRQn); // Activar Interrupcion por Timer0

LPC\_TIM0->TCR |= 1 << 0; // Timer0 en modo Contador - pag 494

LPC\_GPIO0->FIODIR |= (1 << 21); // P0[21] COMO SALIDA

LPC\_GPIO2->FIODIR |= (0x000000FF<<0); //puerto 2 como salida

while (1) {}

return 0; } //Fin del Main

/\* funcion para atender la interrupcion \*/

void TIMER0\_IRQHandler (void)

{ cuenta++;

if (cuenta ==10){

cuenta=0;

} /Fin del if

LPC\_GPIO2->FIOCLR = (0x000000FF<<0);

LPC\_GPIO2->FIOSET = ( tabla[cuenta]<<0);

if((LPC\_TIM0->IR & 0x01) == 0x01) // si hay interrupcion por MR0, if grande

{

LPC\_TIM0->IR |= 1 << 0; // resetear MR0 interrupt - pag 493

if(LPC\_GPIO0->FIOPIN &(1<<21))

{//LPC\_GPIO0->FIOCLR = (1 << 21); // P0[22] = 0

}

else

LPC\_GPIO0->FIOSET = (1 << 21); // P0[22] = 1

} //fin del if grande; } //Fin de la función handler del TIMER0

Ejercicio de pulsadores

/\*Si el pulsador esta presionado, el LED parpadea, si no lo esta, el LED se mantiene...

se activa la resistencia pull up\*/

#ifdef \_USE\_CMSIS

#include "LPC17xx.h"

#endif

void delay(int);

int time;

int main (void){

LPC\_PINCON -> PINSEL0 &= !(3<<12);

LPC\_PINCON -> PINSEL0 &= !(3<<18);

LPC\_GPIO0 -> FIODIR &= !(1<<6);

LPC\_GPIO0 -> FIODIR |= !(1<<9);

LPC\_PINCON -> PINMODE0 &= !(3<<12);

while (1){ if (LPC\_GPIO0 ->FIOPIN &= (1<<6)){

LPC\_GPIO0 -> FIOSET |=(1<<9);

delay(1000000);

LPC\_GPIO0 -> FIOCLR |= (1<<9);

delay(1000000); } // Fin del If

else{ LPC\_GPIO0 -> FIOSET |= (1<<9);

delay(1000000); } // Fin del else

} // Fin del while

return 0; } // Fin del Main

void delay(int time)

{ for (int i =0; i<time; i++){} // Fin del for } // Fin de la funcion delay