/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Lab 3A\_1 - Memory clock with Basic Timer 1 \*/

/\* \*/

/\* \*/

/\* \*/

/\* MSP430 Teaching ROM \*/

/\* Produced by: MSP430@UBI Group - www.msp430.ubi.pt \*/

/\* \*/

/\* Exercise: Using the MSP-EXP430FG4618 Development Tool and the \*/

/\* MSP430FG4618 device implement a Memory clock with \*/

/\* Basic Timer 1 \*/

/\* \*/

/\* Instructions: \*/

/\* LCD show time in HOUR : MINUTES : SECONDS format \*/

/\* \*/

/\* LED1 turn on while executing basic\_timer\_ISR, it can be used to \*/

/\* measure ISR executing time. \*/

/\* \*/

/\* LED2 toogle each second. \*/

/\* \*/

/\* Complete the code below: \*/

/\* - Configure Watchdog timer \*/

/\* - Configure FLL+ \*/

/\* - Configure Basic Timer1 \*/

/\* - Configure P2 \*/

/\* \*/

/\* \*/

/\* \*/

/\* Copyright by Texas Instruments \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <msp430xG46x.h>

#include "LCD\_defs.h"

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Global data

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

unsigned char hour, min, sec;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Write Hour in LCD

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void LCD\_hour()

{

unsigned char hour1,hour2;

hour1 = hour/10; // find hour most significant digit

hour2 = hour%10; // find hour less significant digit

switch (hour1) // switch to write hour first digit

{

case 1: P7\_A1;

break;

case 2: P7\_A2;

break;

case 3: P7\_A3;

break;

case 4: P7\_A4;

break;

case 5: P7\_A5;

break;

case 6: P7\_A6;

break;

case 7: P7\_A7;

break;

case 8: P7\_A8;

break;

case 9: P7\_A9;

break;

case 0: P7\_A0;

break;

}

switch (hour2) // switch to write hour second digit

{

case 1: P6\_A1;

break;

case 2: P6\_A2;

break;

case 3: P6\_A3;

break;

case 4: P6\_A4;

break;

case 5: P6\_A5;

break;

case 6: P6\_A6;

break;

case 7: P6\_A7;

break;

case 8: P6\_A8;

break;

case 9: P6\_A9;

break;

case 0: P6\_A0;

break;

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Write minutes in LCD

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void LCD\_min()

{

unsigned char min1,min2;

min1 = min/10; // find minutes less significant digit

min2 = min%10; // find minutes less significant digit

switch (min1) // switch to write minutes first digit

{

case 1: P5\_A1;

break;

case 2: P5\_A2;

break;

case 3: P5\_A3;

break;

case 4: P5\_A4;

break;

case 5: P5\_A5;

break;

case 6: P5\_A6;

break;

case 7: P5\_A7;

break;

case 8: P5\_A8;

break;

case 9: P5\_A9;

break;

case 0: P5\_A0;

break;

}

switch (min2) // switch to write minutes second digit

{

case 1: P4\_A1;

break;

case 2: P4\_A2;

break;

case 3: P4\_A3;

break;

case 4: P4\_A4;

break;

case 5: P4\_A5;

break;

case 6: P4\_A6;

break;

case 7: P4\_A7;

break;

case 8: P4\_A8;

break;

case 9: P4\_A9;

break;

case 0: P4\_A0;

break;

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Write seconds in LCD

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void LCD\_sec()

{

unsigned char sec1,sec2;

sec1 = sec / 10;

sec2 = sec % 10;

switch (sec1) // switch to write seconds first digit

{

case 1:P3\_A1;

break;

case 2:P3\_A2;

break;

case 3:P3\_A3;

break;

case 4:P3\_A4;

break;

case 5:P3\_A5;

break;

case 6:P3\_A6;

break;

case 7:P3\_A7;

break;

case 8:P3\_A8;

break;

case 9:P3\_A9;

break;

case 0:P3\_A0;

break;

}

switch (sec2) // switch to write seconds second digit

{

case 1: P2\_A1;

break;

case 2: P2\_A2;

break;

case 3: P2\_A3;

break;

case 4: P2\_A4;

break;

case 5: P2\_A5;

break;

case 6: P2\_A6;

break;

case 7: P2\_A7;

break;

case 8: P2\_A8;

break;

case 9: P2\_A9;

break;

case 0: P2\_A0;

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//LCD clean. All segments are turnned off

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void LCD\_all\_off(void)

{

LCDM2 = 0x00;

LCDM3 = 0x00;

LCDM4 = 0x00;

LCDM5 = 0x00;

LCDM6 = 0x00;

LCDM7 = 0x00;

LCDM8 = 0x00;

LCDM9 = 0x00;

LCDM10 = 0x00;

LCDM11 = 0x00;

LCDM12 = 0x00;

LCDM13 = 0x00;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// BAsic Timer Interrupt Service Routine. Run with 1 sec period

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#pragma vector=BASICTIMER\_VECTOR

\_\_interrupt void basic\_timer\_ISR(void)

{

--------------- // LED1 turn on

--------------- // LED2 toogle

sec++; // increment seconds

LCD\_sec(); // refresh seconds field in LCD

if (sec == 60) // one minute

{

sec = 0; // reset seconds counter

min++; // increment minutes

LCD\_min(); // refresh minutes field in LCD

if (min == 60) // one hour

{

min = 0; // reset minutes counter

hour++; // increment hours

LCD\_min(); // refresh hours field in LCD

if (hour == 24) // one day

{

hour = 0; // reset hours counter

}

}

}

if (sec & 0x01) // toogle clock dots

{

P3\_DOT\_ON;

P5\_DOT\_ON;

}

else

{

P3\_DOT\_OFF;

P5\_DOT\_OFF;

}

P2OUT &=~0x02; // LED1 turn off

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Main routine

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void main(void)

{

WDTCTL = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_; // Stop WDT

FLL\_CTL0 |= XCAP18PF; // Set load cap for 32k xtal

// LCD COM0-COM1-COM2-COM3 configuration

P5DIR |= 0x1E; // Ports P5.2, P5.3 and P5.4 as outputs

P5SEL |= 0x1E; // Ports P5.2, P5.3 and P5.4 as special function (COM1, COM2 and COM3)

// LCD\_A S0-S21 configuration

LCDAPCTL0 = LCDS24 | LCDS20 | LCDS16 | LCDS12 | LCDS8 | LCDS4;

// LCDA configuration

LCDACTL = LCDFREQ\_192 | LCD4MUX | LCDSON | LCDON; // (ACLK = 32768)/192, 4-mux LCD, LCD\_A on, Segments on

LCDAVCTL0 = LCDCPEN; // Charge pump enable

LCDAVCTL1 = VLCD\_3\_44; // VLCD = 3,44 V

// Clean LCD

LCD\_all\_off();

// Set memory clock variables

sec = 0;

min = 0;

hour = 0;

// Write time in LCD

LCD\_sec();

LCD\_min();

LCD\_hour();

// LED1 e LED2 configuration

P2DIR |= \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_; // P2.2 and P2.1 as digital output

P2OUT |= \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_; // LED1 on and LED2 off

// Basic Timer 1 Configuration

BTCTL = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_; // 1 sec interrupt - (ACLK/256)/128

IE2 |= \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_; // Enable Basic Timer 1 interrupt

\_BIS\_SR(LPM3\_bits + GIE); // Enter LPM3 with all interrupts enabled

}