```
/*
 * lcdadcctc.c
 *
 * 16x2 LCD library 4 Data 3 Cmd same port
 * one analog input
 * Atmega 128 L at 16Mhz
 *
 * Created: 22-10-2012 23:12:54
 *  Author: sergio
 */

#ifndef F_CPU
#define F_CPU 16000000
#endif
#define XTAL 16000000
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <stdio.h>
#include <stdlib.h>
//PINOS
/***GENERAL START***/
#define TRUE 1
#define FALSE 0
#define GI 7
/***GENERAL STOP***/

void PORTS_init(void);

/***LCD START***/
//HARDWARE 4 DATA PIN 3 CMD
#define LCD_DDR    DDRA
#define LCD_PORT   PORTA
#define LCD_PIN    PINA
//ASIGN PORT PINS TO LCD
#define RS 0
#define RW 1
#define EN 2
#define NC 3
#define DB0 4
#define DB1 5
#define DB2 6
#define DB3 7
//CMD RS
#define INST 0
#define DATA 1
//PROTOTYPES
void lcd_pulse(void);
void lcd_write(char c, unsigned short D_I);
char lcd_read(unsigned short D_I);
void lcd_BF(void);
void lcd_string(char *s);
void lcd_init(void);
/***LCD STOP***/

/***ADC START***/
#define ADC_DDR    DDRF
#define ADC_PORT   PORTF
#define ADC_PIN    PINF
#define ADPS 0
#define REFS 6
#define MUX 0
#define ADC0 0

unsigned int ADC_value=0;
char ADC_string[20];
unsigned short ADC_flag;

ISR(ADC_vect)
{
        ADC_value=ADC;
        ADC_flag=0;
}

void adc_init(void);
/***ADC STOP***/

/***CTC START***/
#define CTC0_DDR   DDRC
#define CTC0_PORT  PORTC
#define CTC0_PIN   PINC
#define CS0 0
#define COM0 4
#define OC0 4
#define FLIP 1

unsigned int CTC0_count=0;
unsigned short CTC0_flag;

ISR(TIMER0_COMP_vect)
{
        if(CTC0_count>10){
                CTC0_flag=0;
                CTC0_count=0;
                CTC0_PORT^=(1<<FLIP);
                CTC0_flag=1;
        }
        CTC0_count++;
}

void ctc0_init(void);
/***CTC STOP***/

/***MAINMAINMAINMAINMAIN***/
```

```
int main(void)
{
	PORTS_init();
	lcd_init();
	adc_init();
	ctc0_init();
	unsigned int i=0;

	while(TRUE){
		//TODO:: Please write your application code
		if(ADC_flag==0){
			//ADC_value=(ADC_value/1023)*5;
			sprintf(ADC_string,"ADC: %u   ",ADC_value);
			ADC_flag=1;
		}
		//if(CTC0_flag==0){//looses precision
			//CTC0_PORT^=(1<<FLIP);
			//CTC0_flag=1;
		//}
		lcd_write(0x80,INST);//position
		lcd_BF();
		lcd_string(ADC_string);
		switch(i){
			case 0:
				lcd_write(0xC0,INST);//position
				lcd_BF();
				lcd_string("UM    ");
				lcd_BF();
				i=1;
				break;
			case 1:
				lcd_write(0xC0,INST);//position
				lcd_BF();
				lcd_string("DOIS  ");
				lcd_BF();
				i=2;
				break;
			case 2:
				lcd_write(0xC0,INST);//position
				lcd_BF();
				lcd_string("TRES  ");
				lcd_BF();
				i=3;
				break;
			case 3:
				lcd_write(0xC0,INST);//position
				lcd_BF();
				lcd_string("QUATRO");
				lcd_BF();
				i=0;
				break;
			default:
					break;
			}




			_delay_ms(250);//quatro por segundo
	}
	return 0;
}
/*****************SOURCES*********************/

void PORTS_init(void){

	DDRB=(1<<OC0);
	PORTB=(0<<OC0);

	LCD_DDR=(1<<DB0)|(1<<DB1)|(1<<DB2)|(1<<DB3)|(1<<RS)|(1<<RW)|(1<<EN);
	LCD_PORT=0x00;

	ADC_DDR=(0<<ADC0);
	ADC_PORT=(0<<ADC0);

	CTC0_DDR=(1<<FLIP);
	CTC0_PORT&=~(1<<FLIP);
}

/***LCD START***/
//LCD ENABLE PULSE
void lcd_pulse(void)
{
	LCD_PORT|=(1<<EN);
	LCD_PORT&=~(1<<EN);
}

//LCD WRITE
void lcd_write(char c, unsigned short D_I)
{
	LCD_DDR=(1<<DB0)|(1<<DB1)|(1<<DB2)|(1<<DB3)|(1<<RS)|(1<<RW)|(1<<EN);
	if(D_I==0) LCD_PORT&=~(1<<D_I); else LCD_PORT|=(D_I<<RS);
	LCD_PORT&=~(1<<RW);
	if(c & 0x80) LCD_PORT|=1<<DB3; else LCD_PORT&=~(1<<DB3);
	if(c & 0x40) LCD_PORT|=1<<DB2; else LCD_PORT&=~(1<<DB2);
	if(c & 0x20) LCD_PORT|=1<<DB1; else LCD_PORT&=~(1<<DB1);
	if(c & 0x10) LCD_PORT|=1<<DB0; else LCD_PORT&=~(1<<DB0);
	lcd_pulse();
	if(c & 0x08) LCD_PORT|=1<<DB3; else LCD_PORT&=~(1<<DB3);
	if(c & 0x04) LCD_PORT|=1<<DB2; else LCD_PORT&=~(1<<DB2);
	if(c & 0x02) LCD_PORT|=1<<DB1; else LCD_PORT&=~(1<<DB1);
	if(c & 0x01) LCD_PORT|=1<<DB0; else LCD_PORT&=~(1<<DB0);
	lcd_pulse();
}
```

```
//LCD READ
char lcd_read(unsigned short D_I)
{
	char c=0;
	LCD_DDR=(0<<DB0)|(0<<DB1)|(0<<DB2)|(0<<DB3)|(1<<RS)|(1<<RW)|(1<<EN);
	if(D_I==0) LCD_PORT&=~(1<<D_I); else LCD_PORT|=(D_I<<RS);
	LCD_PORT|=(1<<RW);
	lcd_pulse();
	if(LCD_PIN & (1<<DB3)) c|=1<<7; else c&=~(1<<7);
	if(LCD_PIN & (1<<DB2)) c|=1<<6; else c&=~(1<<6);
	if(LCD_PIN & (1<<DB1)) c|=1<<5; else c&=~(1<<5);
	if(LCD_PIN & (1<<DB0)) c|=1<<4; else c&=~(1<<4);
	lcd_pulse();
	if(LCD_PIN & (1<<DB3)) c|=1<<3; else c&=~(1<<3);
	if(LCD_PIN & (1<<DB2)) c|=1<<2; else c&=~(1<<2);
	if(LCD_PIN & (1<<DB1)) c|=1<<1; else c&=~(1<<1);
	if(LCD_PIN & (1<<DB0)) c|=1<<0; else c&=~(1<<0);
	return c;
}

//LCD Busy Flag check
void lcd_BF(void){
	unsigned int i;
	for(i=0;0x80&lcd_read(INST);i++){
		if(i>40000)// 2.5 ms for 16Mhz max wait
		break;
	}
}

//LCD STRING WRITE
void lcd_string(char *s)
{
	char tmp;
	while(*s){
		tmp=*(s++);
		lcd_write(tmp,DATA);
		lcd_BF();
	}
}

//LCD INIC
void lcd_init(void)
{
	/***INICIALIZACAO LCD**datasheet*/
	_delay_ms(40);

	lcd_write(0x33,INST); //function set
	_delay_us(80);

	lcd_write(0x2B,INST); //function set
	_delay_us(80);
	lcd_write(0x2B,INST); //function set
	_delay_us(80);

	lcd_write(0x0C,INST);// display on/off control
	_delay_us(80);

	lcd_write(0x01,INST);// clear display
	_delay_ms(2.50);

	lcd_write(0x06,INST);// entry mode set (crazy settings)
	_delay_us(80);

	/***INICIALIZATION END***/

	lcd_write(0x1F,INST);// cursor or display shift
	lcd_BF();

	lcd_write(0x03,INST);// return home
	lcd_BF();

}
/***LCD STOP***/

/***ADC START***/
void adc_init(void){
	ADMUX=(0<<REFS)|(0<<ADLAR)|(0<<MUX);
	ADCSRA=(1<<ADEN)|(1<<ADSC)|(1<<ADFR)|(0<<ADIF)|(1<<ADIE)|(4<<ADPS);
	SREG|=(1<<GI);
}
/***ADC STOP***/

/***CTC START***/
void ctc0_init(void){
	//Timer zero settings (type, output and prescaler)
	TCCR0=(0<<FOC0)|(1<<WGM01)|(0<<WGM00)|(1<<COM0)|(7<<CS0);
	//Timer Interrupt Mask
	TIMSK=(0<<OCIE2)|(0<<TOIE2)|(0<<TICIE1)|(0<<OCIE1A)|(0<<OCIE1B)|(0<<TOIE1)|
(1<<OCIE0)|(1<<TOIE0);
	//Timer compare match valores admitidos de 1 até 254.
	OCR0=254;
	//Asynchronous ASSR
	//Interrupt Flags view TIFR
	//SFIOR
	SREG|=(1<<GI);
}
/***CTC STOP***/
```