REFERENCES

[1] Ferran Reverter , Xiujun Li , Gerard C.M. Meijer, "Liquid-level measurement system based on a remote grounded capacitive sensor", 2007

[2] Hüseyin Canbolat, "A Novel Level Measurement Technique Using Three Capacitive Sensors for Liquids", 2009

[3] Guirong Lu, Mitio Seto and Katsunori Shida , "A new proposal of multi-functional level meter", 2003

[4] Edin Terzic , C.R.Nagarajah, Muhammad Alamgir, "Capacitive sensor-based fluid level measurement in a dynamic environment using neural network", 2010

[5] Ferry N. Toth, Gerard C. M. Meijer and Matthijs van der Lee, "A Planar Capacitive Precision Gauge", 1997

[6] Pei Li, Ning Tigang, H Wang Lei, Yang Bin, Li Yantao, Jian Shuisheng, "A smart optical liquid level sensor based on Hg cladding optical waveguide", 2006

[7] Hossein Golnabi, "Design and operation of a fiber optic sensor for liquid level detection", 2002

[8] F. P´erez-Oc´on, M. Rubi˜no, J.M. Abril, P. Casanova, J.A. Mart´ınez, "Fiber-optic liquid-level continuous gauge", 2005

[9] Pekka Raatikainen a,U, Ivan Kassamakov b, Roumen Kakanakov b, Mauri Luukkala, "Fiber-optic liquid-level sensor", 1997

[10] Kyung-Rak Sohn, Joon-Hwan Shim, "Liquid-level monitoring sensor systems using fiber Bragg grating embedded in cantilever", 2009

[11] C. Vázquez, A.B. Gonzalo, S. Vargas, J. Montalvo, "Multi-sensor system using plastic optical fibers for intrinsically safe level measurements", 2004

[12] Tatsuo Nakagawa, Akihiko Hyodo, Kenji Kogo, Hideaki Kurata, Kenichi Osada and Shigeru Oho, "Contactless Liquid-Level Measurement With Frequency-Modulated Millimeter Wave through Opaque Container", 2013

[13] Jenny Terzic, C.R. Nagarajah, Muhammad Alamgir , "Fluid level measurement in dynamic environments using a single ultrasonic sensor and Support Vector Machine (SVM)", 2010

[14] Ti-Ho Wang a, Ming-Chih Lu a, Chen-Chien Hsu b,*, Cheng-Chuan Chen a, Jia-Dong Tan, "Liquid-level measurement using a single digital camera", 2009

[15] Method of Detecting water Level in steam Boilers, Available: http://www.spiraxsarco.com/Resources/Pages/Steam-Engineering-Tutorials/the-boiler-house/methods-of-detecting-water-level-in-steam-boilers.aspx

[16] A Dozen Ways to Measure Fluid Level and How They Work, Available:http://www.sensorsmag.com/sensors/leak-level/a-dozen-ways-measure-fluid-level-and-how-they-work-1067

[17] A dozen ways to measure fluid level, Available:http://new.abb.com/products/measurement-products/level/a-dozen-ways-to-measure-fluid-level

[18] Traceability in Liquid Flow Measurement Working group 1.53, Available: https://www.ptb.de/cms/en/ptb/fachabteilungen/abt1/fb-15/ag-153.html

[19] Encyclopedia of Chemical Engineering Equipment, Available:http://encyclopedia.che.engin.umich.edu/Pages/ProcessParameters/LevelMeasurement/LevelMeasurement.html

[20] Liquid Level Switches Information, Available**:**http://www.globalspec.com/learnmore/sensors_transducers_detectors/level_sensing/liquid_level_switche**s**

[21] Drum Gauges and Drum Sticks, Available: http://www.bestcontainers.com/gauges.html

Appendix A:

Data

Liquid: Water

Tank: Cement cubical tank

No	Dipstick Reading (cm)	Sensometer Reading (cm)	Capacitive base level measurement sensor (cm)				
			i	ii	iii	iv	v
Temperature			**26°C**				
1	0	0	-0.02	0.00	-0.06	0.04	0.04
2	10	10	10.13	10.08	10.01	10.18	10.15
3	20	20	19.74	20.00	19.87	19.79	20.14
4	30	30	29.72	29.93	30.13	29.63	29.72
5	40	40	40.32	40.12	40.16	40.09	40.06
6	50	50	50.12	49.69	50.19	50.15	50.30
7	60	60	60.28	60.32	60.28	60.20	60.09
8	70	70	70.14	69.97	70.19	69.97	70.23
9	80	80	80.19	80.00	79.86	79.96	80.19
Temperature			**30.5°C**				
1	0	0	-0.03	0.10	-0.03	-0.03	-0.01
2	10	10	10.22	10.22	10.24	10.26	10.14
3	20	20	19.86	19.78	19.94	19.49	19.94
4	30	30	30.22	30.22	30.27	30.19	30.16
5	40	40	40.28	40.25	40.18	40.21	40.31
6	50	50	50.11	49.68	50.18	50.07	50.29
7	60	60	60.20	60.27	60.43	59.96	60.20
8	70	70	69.92	70.14	70.18	69.96	70.18
9	80	80	79.58	79.62	80.13	80.04	79.99
Temperature			**34.5°C**				
1	0	0	0.00	-0.02	-0.02	0.03	0.00
2	10	10	10.11	10.13	10.11	10.16	10.09
3	20	20	20.12	20.01	19.75	19.77	20.12
4	30	30	30.03	30.15	30.18	30.15	30.09
5	40	40	40.24	40.27	40.27	40.17	40.44
6	50	50	50.10	50.46	50.17	50.21	50.28
7	60	60	60.31	60.27	60.23	60.27	60.19
8	70	70	69.96	70.26	70.17	70.21	70.21
9	80	80	80.17	79.99	80.36	80.13	80.03

Liquid: Water

Tank: PVC Tank

No	Dipstick Reading (cm)	SensorMeter Reading (cm)	Capacitive base level measurement sensor (cm)				
			i	ii	iii	iv	v
Temperature			**30.5°C**				
1	0	0	-0.02	-0.04	-0.02	0.04	0.04
2	10	10	10.12	10.12	10.15	10.22	10.27
3	20	20	19.73	19.91	19.91	19.83	20.10
4	30	30	29.67	29.97	29.88	29.67	29.76
5	40	40	40.36	40.17	40.20	40.13	39.94
6	50	50	49.98	50.13	50.23	50.31	50.34
7	60	60	60.25	60.37	60.17	60.25	60.01
8	70	70	70.19	70.02	70.10	70.02	70.10
9	80	80	80.37	80.28	80.51	80.18	80.23

Liquid: Diesel

Tank: Horizontal Cylindrical Tank

No	Dipstick Reading (cm)	Capacitive base level measurement sensor (cm)				
		i	ii	iii	iv	v
Temperature		**25.5°C**				
1	0	-0.05	-0.53	-0.05	-0.05	0.68
2	20	20.41	20.41	20.66	20.16	20.66
3	40	40.54	40.02	40.02	39.76	40.28
4	60	60.53	60.53	59.99	59.73	60.26
5	80	80.89	80.34	79.79	80.62	80.06
6	100	100.21	99.92	99.92	100.21	99.64
7	120	120.99	120.70	120.41	120.11	120.41
Temperature		**31°C**				
1	0	-0.19	-0.44	0.29	-0.19	0.53
2	20	20.52	20.27	20.77	19.76	20.52
3	40	40.65	40.13	39.87	39.87	40.13
4	60	60.65	60.38	60.12	59.58	59.85
5	80	80.19	80.19	79.92	79.64	80.47
6	100	100.06	100.35	99.50	100.06	100.35
7	120	120.85	120.26	119.67	120.55	120.26
Temperature		**35°C**				
1	0	0.49	-0.73	-0.24	0.00	0.49
2	20	20.97	19.97	20.47	20.72	20.47
3	40	39.83	40.60	40.08	40.08	40.08
4	60	60.60	60.33	59.80	60.07	60.07
5	80	80.15	80.15	79.59	80.15	80.15
6	100	99.73	100.30	100.01	99.73	99.16
7	120	120.80	120.21	119.63	119.92	120.21

Liquid: Petrol

Tank: Horizontal Cylindrical Tank

No	Dipstick Reading (cm)	Capacitive base level measurement sensor (cm)				
		i	ii	iii	iv	v
	Temperature	**24.5°C**				
1	0	-0.13	-0.34	0.08	-0.13	0.51
2	20	20.04	20.26	20.26	20.70	20.92
3	40	39.86	40.09	40.32	40.54	39.86
4	60	60.19	60.43	60.66	59.72	60.19
5	80	81.05	80.56	80.07	80.32	79.83
6	100	99.92	99.42	100.17	99.42	99.67
7	120	120.21	119.69	120.47	119.95	120.21
	Temperature	**30.5°C**				
1	0	0.21	0.00	0.00	-0.21	0.00
2	20	19.74	20.18	19.96	20.83	20.18
3	40	40.91	40.23	40.23	40.91	40.00
4	60	60.82	60.34	60.58	60.11	60.82
5	80	80.23	80.48	80.97	80.23	80.72
6	100	99.84	100.34	100.09	100.09	100.85
7	120	119.87	120.39	120.65	120.91	120.91
	Temperature	**35°C**				
1	0	-0.08	-0.08	0.13	-0.08	0.13
2	20	20.52	20.30	20.08	20.96	20.30
3	40	39.90	40.13	40.36	40.36	39.90
4	60	60.47	60.24	59.76	60.24	59.53
5	80	80.12	80.60	80.36	80.36	81.09
6	100	99.71	99.96	100.47	100.21	100.21
7	120	120.52	120.25	119.99	120.52	121.04

Liquid: Petrol

Tank: Horizontal Cylindrical Tank

No	Dipstick Reading (cm)	OCIO level Measurement (cm)	Capacitive Sensor (cm)				
			i	ii	iii	iv	v
1	40	40	39.99	40.22	40.44	40.67	39.54
2	80	80	81.18	80.69	80.20	80.44	79.96
3	120	120	119.29	119.82	120.60	119.55	120.34

Liquid: Diesel

Tank: Horizontal Cylindrical Tank

No	Dipstick Reading (cm)	OCIO level Measurement (cm)	Capacitive Sensor (cm)				
			i	ii	iii	iv	v
1	40	40	40.39	39.87	39.87	40.39	40.13
2	80	80	80.75	79.64	79.92	79.64	80.75
3	120	120	120.85	120.26	119.67	120.55	120.26

Appendix B:

```
#include      <pic.h>
#include      <stdio.h>
#include    <math.h>

#include      "delay.h"
#include      "lcd.h"
#include      "test.h"

/*********************************************************************
MAIN PROGRAM BEGINS HERE
*********************************************************************/
void main()
{
int_port();
init_a2d();
Init_Lcd();
init_portb_change();
Prj_name_Display();
short_beep();
DelayS(1);
lcdclr();

get_parameters();
init_calibration();
DelayS(3);
```



```
                FLAG.data_ok=0;         // read temperature
                while(!FLAG.data_ok){
                ADIF=0;
                ADIE=1;
                ADGO=1;
                DelayMs(30);
                }
                ADIE=0;
                sys_temp=actual_temp;
                dis_temperature();

f_count=0;
FLAG.freq_ok=0;
init_usart();

while(1)
{
        if(FLAG.freq_ok){    // 5second average
        fa_total=fa_total+fa_temp;
        f_count++;
                if(f_count==25){
                buzzer();
                T0IE=0;
                fa=fa_total/(float)5000;    //in KHz
```

```
                //fa=56.78;
                //fa_total=fa*5000;

                dis_freq((unsigned int)(fa_total/50));
                send_frequency();
                h=(((eq_F/fa)-(eq_A)-(c0))/eq_B);

                        if(h>0){
                        b2_bcd(h*(float)10000);
                        dis_height();
                        }
                        else{
                        dis_error_height();
                        }
                DelayS(3);
                FLAG.data_ok=0;
                while(!FLAG.data_ok){
                ADIF=0;
                ADIE=1;
                ADGO=1;
                DelayMs(30);
                }
                ADIE=0;
                dis_temperature();
                        if((actual_temp>(sys_temp+50))||(actual_temp<(sys_temp-50))){
                        dis_temp_error();
                        DelayS(2);
                        }
                        fa_total=0;
                        f_count=0;
                        init_tmr0();
                        init_timer1();
                        T0IE=1;
                        }
        FLAG.freq_ok=0;
        }
}
}

//************************* Interrupt Programme**************

void interrupt isr()
{
        if(T0IF){
        TMR0=7;
        toif_count++;
                if(toif_count==125){
                fa_temp=((float)((TMR1H*256)+TMR1L)*8);
                TMR1L=0;
                TMR1H=0;
                toif_count=0;
                FLAG.freq_ok=1;
                }
        T0IF=0;
```

```
        }

        if(RBIF){
        key_scan();
        RBIF=0;
        }

        if(ADIF){
        ad_value=0;
        ad_value=ad_value+ADRESH;
        ad_value=ad_value<<8;
        ad_value=ad_value+ADRESL;   //10 bit A/D result

        ++a2d_count;
        tot_ad_value+=ad_value;

                if(a2d_count==5)
                {
                a2d_count=0;
                ad_value=tot_ad_value/5;
                tot_ad_value=0;

                temperature=(double)ad_value*0.50968921;
                actual_temp=(unsigned int)temperature;
                FLAG.data_ok=1;
                }
        ADIF=0;
        }
}

//**********************************Parameter Request******************
void get_parameters()
{
KEY.press=0;
key_value=0;
menu_l=1;

        while(menu_l==1){
        lcdclr();
        Line_1();
        printf("Do You Know the ");
        Line_2();
        printf("Permittivity..?" );
                if(!KEY.press){
                DelayS(3);
                }
        Line_1();
        printf("If YES-> Press #");
        Line_2();
        printf("If NO -> Press *");
                if(!KEY.press){
                DelayS(3);
                }
                else{
```

```
                    if(key_value=='#'){
                    menu_l=3;
                    }
                    /*else if(key_value=='*'){
                    menu_l=2;
                    }*/
                KEY.press=0;
                }
        }

while(menu_l==3){
lcdclr();
Line_1();
printf("1)Sensor A Para.");
Line_2();
printf("  Da1-> 00.00 mm");
lcd_gotoxy(2,8);
        Write_Lcd_Data(eeprom_read(0)+0x30);
        value[0]=eeprom_read(0);
        Write_Lcd_Data(eeprom_read(1)+0x30);
        value[1]=eeprom_read(1);
        Write_Lcd_Data('.');
        Write_Lcd_Data(eeprom_read(2)+0x30);
        value[2]=eeprom_read(2);
        Write_Lcd_Data(eeprom_read(3)+0x30);
        value[3]=eeprom_read(3);

get_dia_value();
da1=value[0]*10+value[1]+value[2]*0.1+value[3]*0.01;
        for(i=0; i<4; i++){
        eeprom_write(i,value[i]);
        }
DelayMs(250);

Line_2();
printf("  Da2-> 00.00 mm");
lcd_gotoxy(2,8);
        Write_Lcd_Data(eeprom_read(4)+0x30);
        value[0]=eeprom_read(4);
        Write_Lcd_Data(eeprom_read(5)+0x30);
        value[1]=eeprom_read(5);
        Write_Lcd_Data('.');
        Write_Lcd_Data(eeprom_read(6)+0x30);
        value[2]=eeprom_read(6);
        Write_Lcd_Data(eeprom_read(7)+0x30);
        value[3]=eeprom_read(7);

get_dia_value();
da2=value[0]*10+value[1]+value[2]*0.1+value[3]*0.01;
lcd_gotoxy(2,7);
        for(i=4; i<8; i++){
        eeprom_write(i,value[i-4]);
        }
DelayMs(250);
```

```
Line_2();
printf("  Da3-> 00.00 mm");
lcd_gotoxy(2,8);
       Write_Lcd_Data(eeprom_read(8)+0x30);
       value[0]=eeprom_read(8);
       Write_Lcd_Data(eeprom_read(9)+0x30);
       value[1]=eeprom_read(9);
       Write_Lcd_Data('.');
       Write_Lcd_Data(eeprom_read(10)+0x30);
       value[2]=eeprom_read(10);
       Write_Lcd_Data(eeprom_read(11)+0x30);
       value[3]=eeprom_read(11);

get_dia_value();
da3=value[0]*10+value[1]+value[2]*0.1+value[3]*0.01;
       for(i=8; i<12; i++){
       eeprom_write(i,value[i-8]);
       }
DelayMs(250);

Line_2();
printf("  Ha-> 000.00 cm");
lcd_gotoxy(2,7);
       Write_Lcd_Data(eeprom_read(12)+0x30);
       value[0]=eeprom_read(12);
       Write_Lcd_Data(eeprom_read(13)+0x30);
       value[1]=eeprom_read(13);
       Write_Lcd_Data(eeprom_read(14)+0x30);
       value[2]=eeprom_read(14);
       Write_Lcd_Data('.');
       Write_Lcd_Data(eeprom_read(15)+0x30);
       value[3]=eeprom_read(15);
       Write_Lcd_Data(eeprom_read(16)+0x30);
       value[4]=eeprom_read(16);

get_height_value();
ha=(value[0]+value[1]*0.1+value[2]*0.01+value[3]*0.001+value[4]*0.0001);// convert
to meter
       for(i=12; i<17; i++){
       eeprom_write(i,value[i-12]);
       }
DelayMs(250);

Line_2();
Write_Lcd_Data(0x20);
Write_Lcd_Data(0x20);
Write_Lcd_Data(0xE3);
printf("ai-> 00.00      ");
lcd_gotoxy(2,8);
       Write_Lcd_Data(eeprom_read(17)+0x30);
       value[0]=eeprom_read(17);
       Write_Lcd_Data(eeprom_read(18)+0x30);
       value[1]=eeprom_read(18);
```

```
        Write_Lcd_Data('.');
        Write_Lcd_Data(eeprom_read(19)+0x30);
        value[2]=eeprom_read(19);
        Write_Lcd_Data(eeprom_read(20)+0x30);
        value[3]=eeprom_read(20);

get_dia_value();
eai=value[0]*10+value[1]+value[2]*0.1+value[3]*0.01;
        for(i=17; i<21; i++){
        eeprom_write(i,value[i-17]);
        }
DelayMs(250);

Line_2();
Write_Lcd_Data(0x20);
Write_Lcd_Data(0x20);
Write_Lcd_Data(0xE3);
printf("a -> 00.00      ");
lcd_gotoxy(2,8);
        Write_Lcd_Data(eeprom_read(21)+0x30);
        value[0]=eeprom_read(21);
        Write_Lcd_Data(eeprom_read(22)+0x30);
        value[1]=eeprom_read(22);
        Write_Lcd_Data('.');
        Write_Lcd_Data(eeprom_read(23)+0x30);
        value[2]=eeprom_read(23);
        Write_Lcd_Data(eeprom_read(24)+0x30);
        value[3]=eeprom_read(24);

get_dia_value();
ea=value[0]*10+value[1]+value[2]*0.1+value[3]*0.01;
        for(i=21; i<25; i++){
        eeprom_write(i,value[i-21]);
        }
DelayMs(250);

Line_2();
Write_Lcd_Data(0x20);
Write_Lcd_Data(0x20);
Write_Lcd_Data(0xE3);
printf("l -> 00.00      ");
lcd_gotoxy(2,8);
        Write_Lcd_Data(eeprom_read(25)+0x30);
        value[0]=eeprom_read(25);
        Write_Lcd_Data(eeprom_read(26)+0x30);
        value[1]=eeprom_read(26);
        Write_Lcd_Data('.');
        Write_Lcd_Data(eeprom_read(27)+0x30);
        value[2]=eeprom_read(27);
        Write_Lcd_Data(eeprom_read(28)+0x30);
        value[3]=eeprom_read(28);

get_dia_value();
el=value[0]*10+value[1]+value[2]*0.1+value[3]*0.01;
```

```
        for(i=25; i<29; i++){
        eeprom_write(i,value[i-25]);
        }
DelayMs(250);

Line_2();
printf("Ra-> 0       Ohm");
get_resistor_value();
ra=r_temp/1000;
DelayMs(250);

Curser_Normal();
dis_thankyou();
menu_1=0;
}
}

void get_dia_value()
{
lcd_gotoxy(2,8);
Curser_Blink();
        /*for(i=0; i<4; i++){
        value[i]=0;          // clear the variable array
        }      */
key_value=0;
dig_count=0;

        while(dig_count<4 && key_value!='#'){
            if(KEY.press && key_value=='*'){
                dig_count=0;
                lcd_gotoxy(2,8);
                        for(i=0; i<4; i++){
                        value[i]=0;          // clear the variable
                        }
                printf("00.00");
                lcd_gotoxy(2,8);
                KEY.press=0;
                }

                if(KEY.press && key_value>0x2F && key_value<0x3A){
                value[dig_count]=key_value-0x30;
                Write_Lcd_Data(key_value);
                        if(dig_count==1){
                        Cursor_Right();
                        }
                KEY.press=0;
                dig_count++;
                }
        }
KEY.press=0;
}

void get_height_value()
{
```

```
lcd_gotoxy(2,7);
Curser_Blink();
	/*for(i=0; i<5; i++){
	value[i]=0;		// clear the variable array
	}*/
key_value=0;
dig_count=0;

	while(dig_count<5 && key_value!='#'){

		if(KEY.press && key_value=='*'){
		dig_count=0;
		lcd_gotoxy(2,7);
			for(i=0; i<5; i++){
			value[i]=0;		// clear the variable
			}
		printf("000.00");
		lcd_gotoxy(2,7);
		KEY.press=0;
		}

		if(KEY.press && key_value>0x2F && key_value<0x3A){
		value[dig_count]=key_value-0x30;
		Write_Lcd_Data(key_value);
			if(dig_count==2){
			Cursor_Right();
			}
		KEY.press=0;
		dig_count++;
		}
	}
KEY.press=0;
}

void get_resistor_value()
{
lcd_gotoxy(2,5);
Curser_Blink();
	for(i=0; i<7; i++){
	value[i]=0;		// clear the variable array
	}
key_value=0;
dig_count=0;
r_temp=0;

	while(dig_count<7 && key_value!='#'){
		if(KEY.press && key_value=='*'){
		dig_count=0;
		lcd_gotoxy(2,5);
			for(i=0; i<7; i++){
			value[i]=0;		// clear the variable
			}
		printf("0      ");
		lcd_gotoxy(2,5);
```

```
                KEY.press=0;
                }

                if(KEY.press && key_value>0x2F && key_value<0x3A){
                r_temp=(r_temp*10)+(key_value-0x30);
                Write_Lcd_Data(key_value);
                KEY.press=0;
                dig_count++;
                }
        }
KEY.press=0;
}
//**************************Inital Calibration**********************

void init_calibration()
{
lcdclr();
Line_1();
printf("Int. Calibration");
Line_2();
printf("  H -> 000.00 cm");
get_height_value();
h=(value[0]+value[1]*0.1+value[2]*0.01+value[3]*0.001+value[4]*0.0001);// convert
to meter
//h=value[0]*100+value[1]*10+value[2]+value[3]*0.1+value[4]*0.01;
DelayS(3);
Line_1();
printf("Pls. Set Sensor  ");
Line_2();
printf("& Press '#' Key  ");
menu_l=1;
KEY.press=0;
key_value=0;
        while(menu_l==1){           // check '#' Key
                if(KEY.press){
                        if(key_value=='#'){
                        menu_l=0;
                        }
                KEY.press=0;
                }
        }
Line_1();
printf("Calculating C0..");
Line_2();
printf("Please wait 5sec");

FLAG.freq_ok=0;
fa_total=0;
f_count=0;
init_tmr0();
init_timer1();

//TMR1H=0x07;
//TMR1L=0xB8; //testing
```

```
        while(f_count<25){
                if(FLAG.freq_ok){
                fa_total=fa_total+fa_temp;
                f_count++;
                FLAG.freq_ok=0;
                }
        }

dis_freq((unsigned int)(fa_total/50));
fa=fa_total/(float)5000;    // 5sec average in KHz
DelayS(2);

//da3=25.5;
//da2=3.5;
//da1=0.4;
//el=80;
//ea=1;
//eai=3;
//h=0;
//ha=1;
//ra=100;
//fa=79.05;

eq_a=(log(da3/da2)/ea);
eq_b=(log(da2/da1)/eai);
eq_c=(log(da3/da2)/el);

eq_A=(55.654977/(eq_a+eq_b));
eq_B=(55.654977/(eq_c+eq_b));
eq_F=(1000000/(ra*log(4)));

c0=(((eq_F)/fa)-(eq_A*ha));//
eq_B=(eq_B-eq_A);
c0=c0-(eq_B)*h;

b2_bcd((int)(c0*100));
Line_1();
printf("Int. Claibration");
Line_2();
printf("done. [C0=     ]");
lcd_gotoxy(2,10);
Write_Lcd_Data(dig4+0x30);
Write_Lcd_Data(dig3+0x30);
Write_Lcd_Data('.');
Write_Lcd_Data(dig2+0x30);
Write_Lcd_Data(dig1+0x30);

f_count=0;
init_tmr0();
init_timer1();
fa_total=0;
Curser_Normal();
}
```

```
//************************General sub functions***************
void int_port()
{
PORTA=0;
PORTC=0;
TRISB=0xF1;
TRISA=0x01;
TRISC=0x81;
//ADCON1=7;
OPTION=0x04;
}

void init_tmr0()
{
OPTION=0x04;  //1:32 pre scaler
T0IF=0;
T0IE=1;
TMR0=6;
toif_count=0;
PEIE=1;
GIE=1;
}

void init_timer1()
{
T1CON=0x33; // P/S=8, Ex clock
TMR1L=0;
TMR1H=0;
}

void init_portb_change()
{
RBIF=0;
RBIE=1;
PEIE=1;
GIE=1;
}

void b2_bcd(unsigned int value)
{
       dig5=value/(int)10000;
       value=value%(int)10000;
       dig4=value/(int)1000;
       value=value%(int)1000;
       dig3=value/100;
       value=value%100;
       dig2=value/10;
       dig1=value%10;
}

void beep(char count)
{
```

```
        while(count!=0){
        RC4=1;
        DelayBigMs(500);
        RC4=0;
        DelayS(1);
        --count;}
}

void short_beep()
{
RC4=1;
DelayS(1);
RC4=0;
}

void key_beep()
{
RC4=1;
DelayBigMs(250);
RC4=0;
}

void buzzer()
{
RC4=1;
DelayMs(250);
RC4=0;
}

void send_frequency()
{
rs232_out('F');
rs232_out(dig5+0x30);
rs232_out(dig4+0x30);
rs232_out(dig3+0x30);
rs232_out('.');
rs232_out(dig2+0x30);
rs232_out(dig1+0x30);
rs232_out('!');
}

void init_usart()
{
SPBRG=129;
TXSTA=0x24;
RCSTA=0x90;
RCIE=1;
PEIE=1;
GIE=1;
}

void rs232_out(unsigned char tx_data)
{
while(TXIF==0){}
```

```
TXREG=tx_data;
}

void init_a2d()
{
ADCON0=0x41;
ADCON1=0x8E;          //Right justify, RA0 input
ADIF=0;
ADIE=0;
}

//*****************************LCD sub functions*******************

void putch(char c)
{
Write_Lcd_Data(c);
RB7=1;
}

void Prj_name_Display()
{
lcdclr();
Line_1();
printf("----WELCOME-----");
Line_2();
printf(" Modern Sensor  ");
DelayS(2);
lcdclr();
Line_1();
printf("  Liquid Level  ");
Line_2();
printf("   Measurement  ");
DelayS(2);
}

void dis_thankyou()
{
lcdclr();
Line_1();
printf("Setting Complete");
Line_2();
printf("Please wait.....");
DelayS(3);
}

void dis_error_height()
{
lcdclr();
Line_1();
printf("Frequency Error!");
Line_2();
printf("Pls.check sensor");
}
```

```
void dis_temp_error()
{
lcdclr();
Line_1();
printf("Liq.Temp. Error!");
Line_2();
printf("Pls.check sensor");
}

void dis_height()
{
Line_2();
printf("Height: ");
lcd_gotoxy(2,7);
Write_Lcd_Data(dig5+0x30);
Write_Lcd_Data(dig4+0x30);
Write_Lcd_Data(dig3+0x30);
Write_Lcd_Data('.');
Write_Lcd_Data(dig2+0x30);
Write_Lcd_Data(dig1+0x30);
Write_Lcd_Data(' ');
Write_Lcd_Data('c');
Write_Lcd_Data('m');
}

void dis_freq(float freq)
{
Line_1();
b2_bcd((unsigned int)freq);
printf("Freq. :  ");
Write_Lcd_Data(dig5+0x30);
Write_Lcd_Data(dig4+0x30);
Write_Lcd_Data(dig3+0x30);
Write_Lcd_Data('.');
Write_Lcd_Data(dig2+0x30);
Write_Lcd_Data(dig1+0x30);
Write_Lcd_Data('K');
Write_Lcd_Data('H');
Write_Lcd_Data('z');
}

void dis_temperature()
{
b2_bcd(actual_temp);
Line_1();
printf("Liq. Temp :    C");
lcd_gotoxy(1,11);
Write_Lcd_Data(dig3+0x30);
Write_Lcd_Data(dig2+0x30);
Write_Lcd_Data('.');
Write_Lcd_Data(dig1+0x30);
}

//*********************************************************************
```

```
void key_scan()
{
KEY.press=0;
key_value=0;

              if(!RB7)             // Column 1 check
              {
              TRISB=0x7F;          //Row 1 input
              PORTB=0;
              DelayBigMs(10);
                     if(!KEY.press && PORTB==0x77)
                     {
                     KEY.press=1;
                     key_value='1';
                     }
                     if(!KEY.press && PORTB==0x7B)
                     {
                     KEY.press=1;
                     key_value='2';
                     }
                     if(!KEY.press && PORTB==0x7D)
                     {
                     KEY.press=1;
                     key_value='3';
                     }
              }

              if(!RB6)             // Column 2 check
              {
              TRISB=0xBF;          //Row 2 input
              PORTB=0;
              DelayBigMs(10);
                     if(!KEY.press && PORTB==0xB7)
                     {
                     KEY.press=1;
                     key_value='4';
                     }
                     if(!KEY.press && PORTB==0xBB)
                     {
                     KEY.press=1;
                     key_value='5';
                     }

                     if(!KEY.press && PORTB==0xBD)
                     {
                     KEY.press=1;
                     key_value='6';
                     }

              }

              if(!RB5)             // Column 3 check
              {
              TRISB=0xDF;          //Row 3 input
```

```
		PORTB=0;
		DelayBigMs(10);
			if(!KEY.press && PORTB==0xD7)
			{
			KEY.press=1;
			key_value='7';
			}
			if(!KEY.press && PORTB==0xDB)
			{
			KEY.press=1;
			key_value='8';
			}
			if(!KEY.press && PORTB==0xDD)
			{
			KEY.press=1;
			key_value='9';
			}
		}

		if(!RB4)		// Column 4 check
		{
		TRISB=0xEF;		//Row 1 input
		PORTB=0;
		DelayBigMs(10);
			if(!KEY.press && PORTB==0xE7)
			{
			KEY.press=1;
			key_value='*';
			}
			if(!KEY.press && PORTB==0xEB)
			{
			KEY.press=1;
			key_value='0';
			}
			if(!KEY.press && PORTB==0xED)
			{
			KEY.press=1;
			key_value='#';
			}
		}

		if(KEY.press)
		{
		key_beep();
		}

	TRISB=0xF0;
	PORTB=0;
		while(!RB7 || !RB6 || !RB5 || !RB4)		// Wait for KEY
release
		{}
	DelayBigMs(50);
		while(!RB7 || !RB6 || !RB5 || !RB4)		// Wait for KEY
release
```

```
            {}
DelayBigMs(50);
}
```

Test-h

```
void putch(char c);
void MainDisplay(void);
void int_port();
void b2_bcd(unsigned int value);
void init_usart();
void init_a2d();
void init_tmr0();
void dis_value();
void tx_data();
void Prj_name_Display();
void dis_functions();
void init_timer1();
void beep(char count);
void show_temperature();
void short_beep();
void rs232_out(unsigned char tx_data);

bank1 unsigned int ad_value,tot_ad_value,actual_temp;
bank1 double temperature;
bank1 unsigned char key_value,menu_1,dig_count,a2d_count,dig1,dig2,dig3;
bank1 unsigned char value[10];

bank1 float ra,rb,temp;
bank1 unsigned int pulse_a,pulse_b,pulse_c,fa,fb,fc;

bank1 struct keys
{
unsigned press:1;
}KEY;

bank1 struct flags
{
unsigned data_ok :1;
unsigned freq_ok:1;
}FLAG;

bank2 float da1,da2,da3,ha,eai,ea,el,h;
bank2 float eq_a,eq_b,eq_c,eq_A,eq_B,eq_F;

//bank3 float db1,db2,db3,hb,ebi,c0;
//bank3 float dc1,dc2,dc3,hc,eci;
//bank3 float eq_a;
```

Lcd-c

```
//*********************************************************************/
```

```
#include <pic.h>
#include "lcd.h" // function prototypes, defines..
#include "delay.h"

char lcdtemp;

void CLK_LCD()
{
lcden=1;
DelayUs(250);
lcden=0;
}

void Init_Lcd( void ) // initialize LCD display
{
        TRISA = 0x01;
        lcdport = 0x00;

        DelayMs(20); // ~15mS delay upon powerup
        lcdport = 0x06; // output setup data to LCD
        CLK_LCD();
        DelayMs(6);
        lcdport = 0x06; // output setup data to LCD
        CLK_LCD();
        DelayMs(5);
        lcdport = 0x06; // output setup data to LCD
        CLK_LCD();
        DelayMs(5);
        lcdport = 0x04; // output setup data to LCD
        CLK_LCD();
        DelayMs(5);
        lcdport = 0x04; // output setup data to LCD
        CLK_LCD();
        lcdport = 0x20; // output setup data to LCD
        CLK_LCD();
        DelayMs(5);
        lcdport = 0x00; // output setup data to LCD
        CLK_LCD();
        lcdport = 0x20; // output setup data to LCD
        CLK_LCD();
        DelayMs(5);
        lcdport = 0x00; // output setup data to LCD
        CLK_LCD();
        lcdport = 0x02; // output setup data to LCD
        CLK_LCD();
        DelayMs(5);
        lcdport = 0x00; // output setup data to LCD
        CLK_LCD();
        lcdport = 0x0C; // output setup data to LCD
        CLK_LCD();
        DelayMs(5);
        lcdport = 0x00; // output setup data to LCD
```

```
	CLK_LCD();
	lcdport = 0x28; // output setup data to LCD
	CLK_LCD();
	DelayMs(5);
}

void Write_Lcd_Data(char data)
{
	lcdport = 0x40 + 0x80;

	lcdtemp = lcdport & 0x40;
	lcdtemp |=  (data >> 4)  ;
	lcdport=lcdtemp<<1;

		if((lcdtemp&0x08)==0x08){
		RA5=1;
		}else{
		RA5=0;
		}

	lcdrs = 1;
	CLK_LCD();
	lcdrs = 0;

	lcdtemp = lcdport & 0x40;
	lcdtemp |= (data & 0x0f);

	lcdport=lcdtemp<<1;

		if((lcdtemp&0x08)==0x08){
		RA5=1;
		}else{
		RA5=0;
		}
	lcdrs = 1;
	CLK_LCD();
	lcdrs = 0;
	DelayBigUs(500);
}

void Write_Lcd_Cmd(char cmd)
{
	lcdtemp = lcdport & 0x40; //green led at portd6
	lcdtemp |= (cmd >> 4) ;
	lcdport=lcdtemp<<1;

		if((lcdtemp&0x08)==0x08){
		RA5=1;
		}else{
		RA5=0;
		}
	lcdrs = 0;
```

```
        CLK_LCD();

        lcdtemp = lcdport & 0x40;
        lcdtemp |= (cmd & 0x0f);
        lcdport=lcdtemp<<1;

                if((lcdtemp&0x08)==0x08){
                RA5=1;
                }else{
                RA5=0;
                }

        lcdrs = 0;
        CLK_LCD();
        DelayBigUs(500);
}

void Line_1(void)
{
Write_Lcd_Cmd(0x80);
}

void Line_2(void)
{
Write_Lcd_Cmd(0xC0);
}

void lcdclr(void)
{
Write_Lcd_Cmd(0x01);
}

void Cursor_Right(void)
{
        Write_Lcd_Cmd(0x14);
}

void Cursor_Left(void)
{
        Write_Lcd_Cmd(0x10);
}

void Display_Shift(void)
{
        Write_Lcd_Cmd(0x1C);
}

void Curser_Blink(void)
{
        Write_Lcd_Cmd(0x0D);
}
```

```
void Curser_Normal(void)
{
        Write_Lcd_Cmd(0x0C);
}

void Font_2(void)
{
        Write_Lcd_Cmd(0x12);
}

//#pragma interrupt_level 1
void lcd_gotoxy (char row, char col)
{
int i;
        if (row == 1)
        Line_1();
        if (row == 2)
        Line_2();
        for ( i = 0; i < col; i++)
        Cursor_Right();
}


void Delay200us(void)// approximate 200us delay
{
        char delay;
        for (delay=66; delay>0;delay--);
}
```

Lcd-h

```
// FUNCTION PROTOTYPES
/* Functions defined in file testlcd.c */

void      Init_Lcd( void );
void      Write_Lcd_Data(char data);
void      Write_Lcd_Cmd(char cmd);
void      Line_1(void);
void      Line_2(void);
void      lcdclr(void);
void      Cursor_Right(void);
void      Cursor_Left(void);
void      Display_Shift(void);
void      lcd_gotoxy (char row, char col);
void      Delay200us(void);
void      Lcd_Busy(void);
void      Curser_Blink(void);
void      Curser_Normal(void);

// MACROS ( DEFINED HERE )
#define lcdport PORTA // Port for lcd data
#define lcden RC0
#define lcdrs RC3
```

Delay-c

```
/*

high level delay routines - see delay.h for more info.

Designed by Shane Tolmie of KeyGhost corporation.  Freely distributable.
Questions and comments to shane@keyghost.com
PICuWEB - Program PIC micros with C. Site has FAQ and sample source code.
http://www.workingtex.com/htpic

For Microchip 12C67x, 16C7x, 16F87x and Hi-Tech C

*/

#ifndef __DELAY_C
#define __DELAY_C

#include <pic.h>

unsigned char delayus_variable;

#include "delay.h"

void DelayBigUs(unsigned int cnt)
{
        unsigned char     i;

        i = (unsigned char)(cnt>>8);
        while(i--)
        {
                i--;
                DelayUs(253);
                CLRWDT();
        }
        DelayUs((unsigned char)(cnt & 0xFF));
}

void DelayMs(unsigned char cnt)
{
        unsigned char     i;
        do {
                i = 4;
                do {
                        DelayUs(250);
                        CLRWDT();
                } while(--i);
        } while(--cnt);
}

//this copy is for the interrupt function
void DelayMs_interrupt(unsigned char cnt)
{
```

```
	unsigned char	i;
	do {
		i = 4;
		do {
			DelayUs(250);
		} while(--i);
	} while(--cnt);
}

void DelayBigMs(unsigned int cnt)
{
	unsigned char	i;
	do {
		i = 4;
		do {
			DelayUs(250);
			CLRWDT();
		} while(--i);
	} while(--cnt);
}

void DelayS(unsigned char cnt)
{
	unsigned char i;
	do {
		i = 4;
		do {
			DelayMs(250);
			CLRWDT();
		} while(--i);
	} while(--cnt);
}

#endif
```

Delay-h

```
#define PIC_CLK 4000000 //4Mhz

/*

lowlevel delay routines

Designed by Shane Tolmie of KeyGhost corporation.  Freely distributable.
Questions and comments to shane@workingtex.com
PICuWEB - Program PIC micros with C. Site has FAQ and sample source code.
http://www.workingtex.com/htpic/

For Microchip 12C67x, 16C7x, 16F87x and Hi-Tech C

Example C:

#define PIC_CLK 4000000

#include "delay.h"

unsigned int timeout_int, timeout_char;

DelayUs(40);  //do NOT do DelayUs(N) of N<5 @ 4Mhz or else it executes DelayUs(255) !!!!
DelayUs(255); //max

dly250n;      //delay 250ns
dly1u;        //delay 1us

timeout_char=timeout_char_us(1147);
while(timeout_char-- && (RA1==0)); //wait up to 1147us for port RA1 to go high
                                   //  - this is the max timeout

timeout_int=timeout_int_us(491512);
while(timeout_int-- && (RA1==0));  //wait up to 491512us for port RA1 to go high
                                   //  - this is the max timeout

*/

#ifndef __DELAY_H
#define __DELAY_H

extern unsigned char delayus_variable;

#if (PIC_CLK == 4000000)
        #define dly125n please remove; for 32Mhz+ only
        #define dly250n please remove; for 16Mhz+ only
        #define dly500n please remove; for 8Mhz+ only
        #define dly1u asm("nop")
        #define dly2u dly1u;dly1u
#elif (PIC_CLK == 8000000)
        #define dly125n please remove; for 32Mhz+ only
        #define dly250n please remove; for 16Mhz+ only
```

```
        #define dly500n asm("nop")
        #define dly1u dly500n;dly500n
        #define dly2u dly1u;dly1u
#elif ( (PIC_CLK == 16000000) || (PIC_CLK == 16257000) )
        #define dly125n please remove; for 32Mhz+ only
        #define dly250n asm("nop")
        #define dly500n dly250n;dly250n
        #define dly1u dly500n;dly500n
        #define dly2u dly1u;dly1u
#elif (PIC_CLK == 20000000)
        #define dly200n asm("nop")
        #define dly400n dly250n;dly250n
        #define dly2u dly400n;dly400n;dly400n;dly400n;dly400n
#elif (PIC_CLK == 32000000)
        #define dly125n asm("nop")
        #define dly250n dly125n;dly125n
        #define dly500n dly250n;dly250n
        #define dly1u dly500n;dly500n
        #define dly2u dly1u;dly1u
#else
        #error delay.h - please define pic_clk correctly
#endif

//*****
//delay routine

#if PIC_CLK == 4000000
        #define DelayDivisor 4
        #define WaitFor1Us asm("nop")
        #define Jumpback asm("goto $ - 2")
#elif PIC_CLK == 8000000
        #define DelayDivisor 2
        #define WaitFor1Us asm("nop")
        #define Jumpback asm("goto $ - 2")
#elif ( (PIC_CLK == 16000000) || (PIC_CLK==16257000) )
        #define DelayDivisor 1
        #define WaitFor1Us asm("nop")
        #define Jumpback asm("goto $ - 2")
#elif PIC_CLK == 20000000
        #define DelayDivisor 1
        #define WaitFor1Us asm("nop"); asm("nop")
        #define Jumpback asm("goto $ - 3")
#elif PIC_CLK == 32000000
        #define DelayDivisor 1
        #define WaitFor1Us asm("nop"); asm("nop"); asm("nop"); asm("nop"); asm("nop")
        #define Jumpback asm("goto $ - 6")
#else
        #error delay.h - please define pic_clk correctly
#endif

#define DelayUs(x) { \
                        delayus_variable=(unsigned char)(x/DelayDivisor); \
```

```
                        WaitFor1Us; } \
                        asm("decfsz _delayus_variable,f"); \
                        Jumpback;
```

/*

timeouts:

C code for testing with ints:

```
                        unsigned int timeout;
                        timeout=4000;
                        PORT_DIRECTION=OUTPUT;
                        while(1)
                        {
                                PORT=1;
                                timeout=8000;
                                while(timeout-- >= 1);      //60ms @ 8Mhz, opt on, 72ms @ 8Mhz,
opt off
                                PORT=0;
                        }
```

Time taken: optimisations on: 16cyc/number loop, 8us @ 8Mhz
optimisations off: 18cyc/number loop, 9us @ 8Mhz
with extra check ie: && (RB7==1), +3cyc/number loop, +1.5us @ 8Mhz

C code for testing with chars:

similar to above

Time taken: optimisations on: 9cyc/number loop, 4.5us @ 8Mhz
with extra check ie: && (RB7==1), +3cyc/number loop, +1.5us @ 8Mhz

Formula: rough timeout value = (<us desired>/<cycles per loop>) * (PIC_CLK/4.0)

To use: //for max timeout of 1147us @ 8Mhz

```
                        #define LOOP_CYCLES_CHAR    9
        //how many cycles per loop, optimizations on
                        #define timeout_char_us(x)          (unsigned
char)((x/LOOP_CYCLES_CHAR)*(PIC_CLK/4.0))
                        unsigned char timeout;
                        timeout=timeout_char_us(1147);
        //max timeout allowed @ 8Mhz, 573us @ 16Mhz
                        while((timeout-- >= 1) && (<extra condition>));      //wait
```

To use: //for max 491512us, half sec timeout @ 8Mhz

```
                        #define LOOP_CYCLES_INT                 16
                //how many cycles per loop, optimizations on
                        #define timeout_int_us(x)  (unsigned
int)((x+/LOOP_CYCLES_INT)*(PIC_CLK/4.0))
                        unsigned int timeout;
```

```
                        timeout=timeout_int_us(491512);
        //max timeout allowed @ 8Mhz
                        while((timeout-- >= 1) && (<extra condition>));        //wait
*/
#define LOOP_CYCLES_CHAR    9                                                                    //how
many cycles per loop, optimizations on
#define timeout_char_us(x)          (long)(((x)/LOOP_CYCLES_CHAR)*(PIC_CLK/1000000/4))

#define LOOP_CYCLES_INT                 16
        //how many cycles per loop, optimizations on
#define timeout_int_us(x)  (long)(((x)/LOOP_CYCLES_INT)*(PIC_CLK/1000000/4))

//if lo byte is zero, faster initialization by 1 instrucion
#define timeout_int_lobyte_zero_us(x)
        (long)(((x)/LOOP_CYCLES_INT)*(PIC_CLK/4.0)&0xFF00)

//function prototypes
void DelayBigUs(unsigned int cnt);
void DelayMs(unsigned char cnt);
void DelayMs_interrupt(unsigned char cnt);
void DelayBigMs(unsigned int cnt);
void DelayS(unsigned char cnt);

#endif
```