

Digital contact-less multi channel capacitance level sensor

The original system of level sensor is proposed. The capacitance level sensing is lied in the base of its functionality. Like in other such sensors its usage principles are built upon next ideas: the dielectric constant of water salt, acid, alkali solutions distinguish from dielectric constant of air and steam.

To implement device prototype was used capacitor shown on figure1. The non-conducting vessel of cylindrical form is covered by two identical plates symmetrically. The plates' sizes are determined by liquid level and capacitor's dimension. Thus, we got a contact-less mean of level measurement. If the height of plates is much greater than vessel diameter, we can neglect by edge effects. The value of this capacitor can be expressed:

$$C = C_0 + \square \cdot h \quad (1)$$

where

C_0 – value of capacitor, when liquid is absent,

\square – coefficient, determined by liquid dielectric constant, vessel geometry and plates allocation,

h – current liquid level in vessel.

As can be seen from formula (1) when liquid level is rising the value of capacitor is progressive. This dependence is underlying of level measurement method, depicted below.

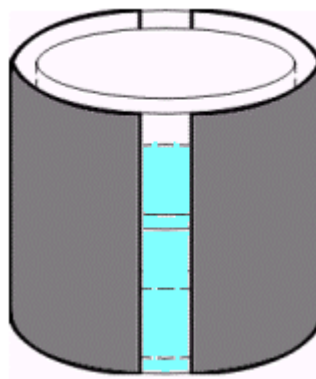


Figure 1. The form of capacitor used in the device prototype.

To analyze electrical circuit where under consideration object is situated was used next equivalent scheme:

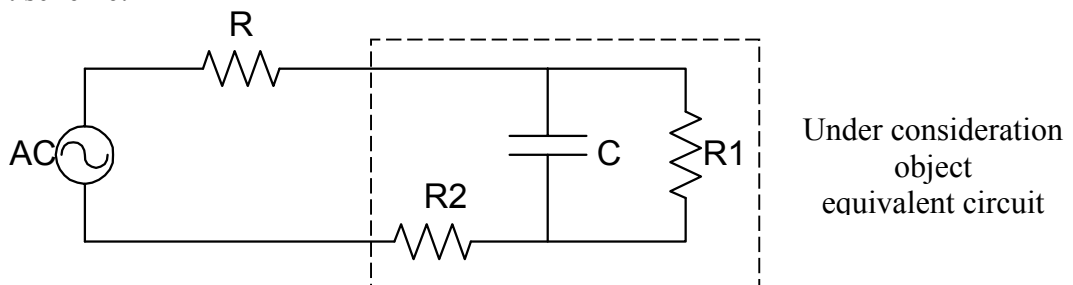


Figure 2. Electrical circuit of level sensor node.

This circuit is powered from harmonic voltage source. Applying sine voltage to capacitor's plates, separates by liquid dielectric, the last always has a little energy losses, caused by viscous rubbing of dipole molecules, and also by dielectric imperfection (a little conductive presence). In this circuit resistor R_1 is parallel connected to capacitor C , that imitate energy losses in real dielectric. R_2 expressed the active resistance of the conductors. If $R_1 \gg 1/(\omega C)$ and $R_2 \ll 1/(\omega C)$, then energy losses on R_1 and R_2 can be neglected.

Knowing operating voltage and voltage drop on capacitor, we can calculate capacitor's value and actual liquid level in vessel.

The considerations below describe a structural scheme of device.

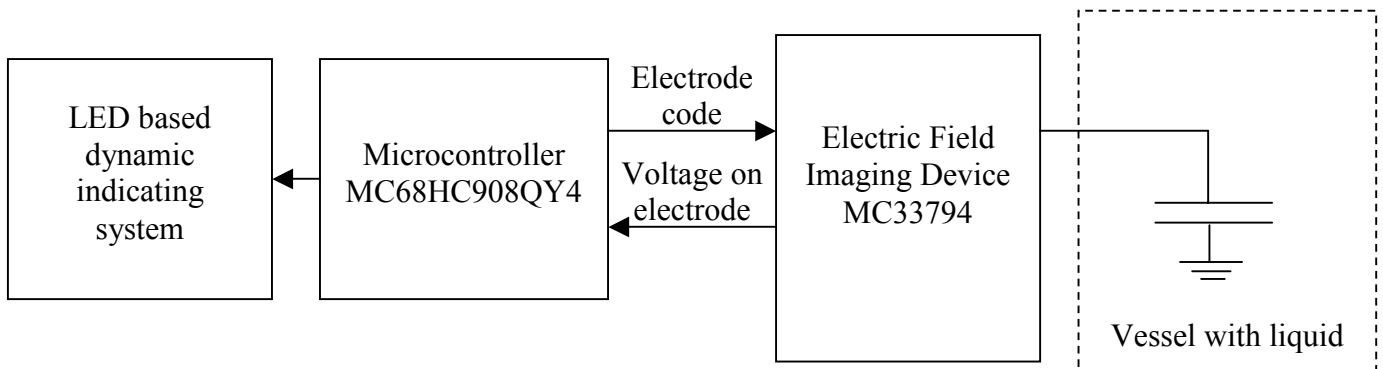


Figure 3. Device structural scheme

The device consists of four main parts: vessel, Electric Field Imaging Device, Microcontroller and LED based dynamic indicating system.

To realize dependence – capacitor's voltage from liquid level, the sinusoidal current passed through it. When signal on capacitor is processed by MC33794, the proportional value of direct analog voltage is moved out for microcontroller analysis. This voltage changes in range $U_{min}...U_{max}$. Each voltage value corresponds to its liquid level in tube (vessel). The dependence $U(h)$ is not linear, where maximum level corresponds to minimum voltage and minimum level to maximum voltage.

The analog output signal from E-Field device is quantized by microcontroller's ADC, and on this base carried out the linear liquid level. The result is shown on LED indicator system, which consists of two decimal digits.

Before device usage it should be calibrated (obtaining parameters C_0 , \square , h_{max}). For this purpose two calibration buttons are provided.

The right calibration order:

1. *Assigning maximum level.*

Press both buttons and hold them. You'll see the counting on display. When number reaches appropriate value – release buttons.

2. *Fixing minimum liquid level.*

Press first button, when vessel contains minimum liquid. You should see a zero value on display.

3. *Fixing maximum level.*

Press second button when vessel contains maximum liquid. You'll see on indicators value h_{max} fixed in item 1.

After calibration passed, the obtained parameters (C_0 , \square , h_{max}) is stored in Flash memory.

Conclusion

The presented system can be used in aggressive industrial automated systems, in home automation (e.g. laundry washer, water supply system), in powerful pumps for detecting liquid presence in tubes (automated turning off in case of liquid absence). Multi channel feature can be useful, where some similar analyzing points are required.

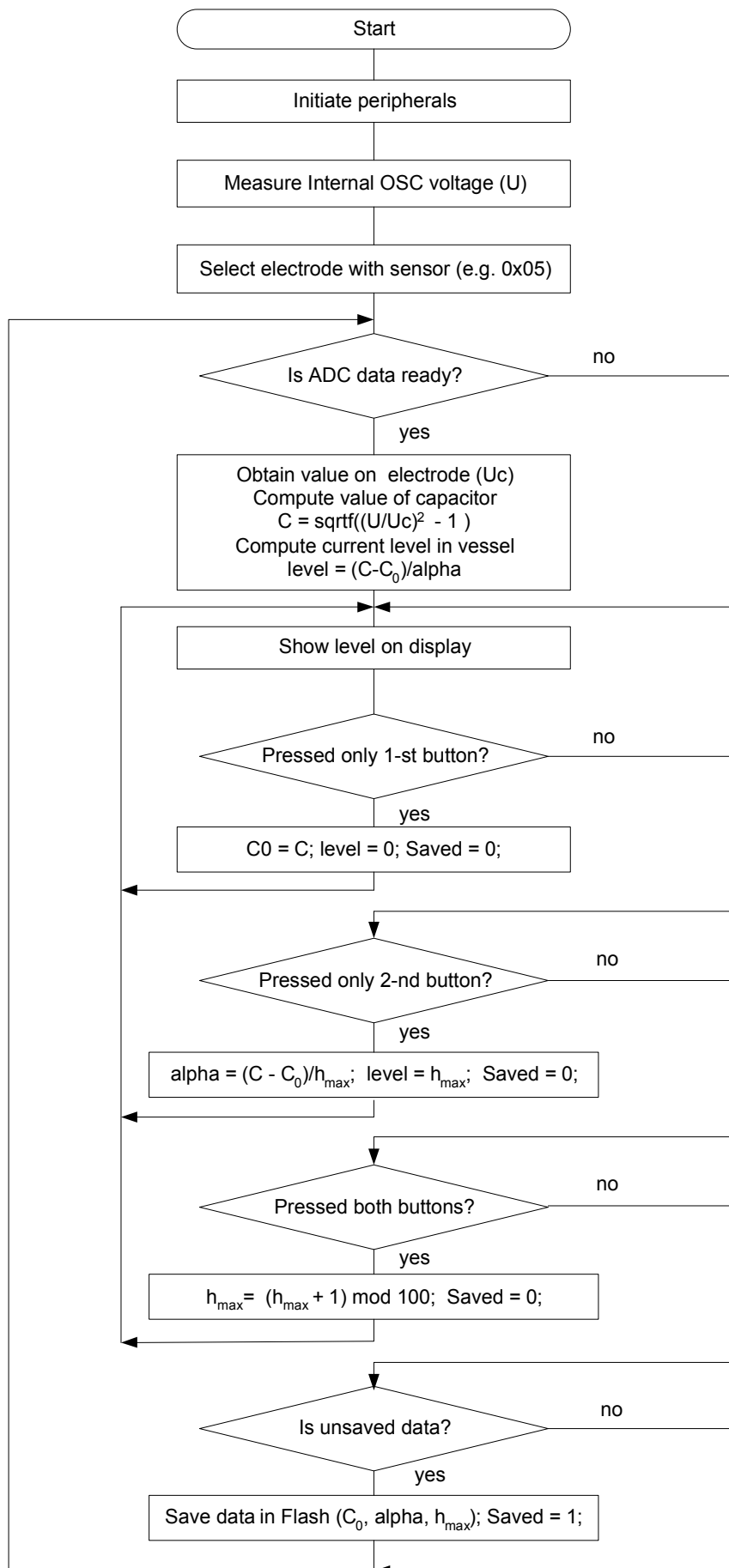


Figure 4. Device firmware flowchart

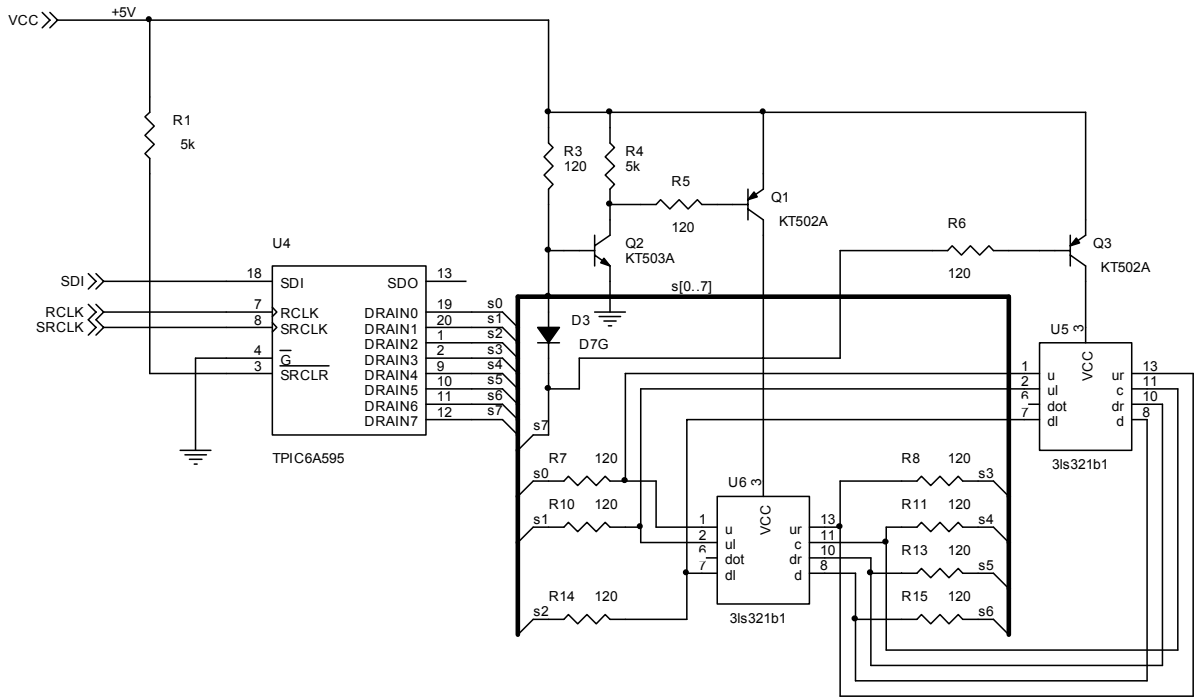


Figure 6. Device Electrical Scheme (Page 2 – Display System)

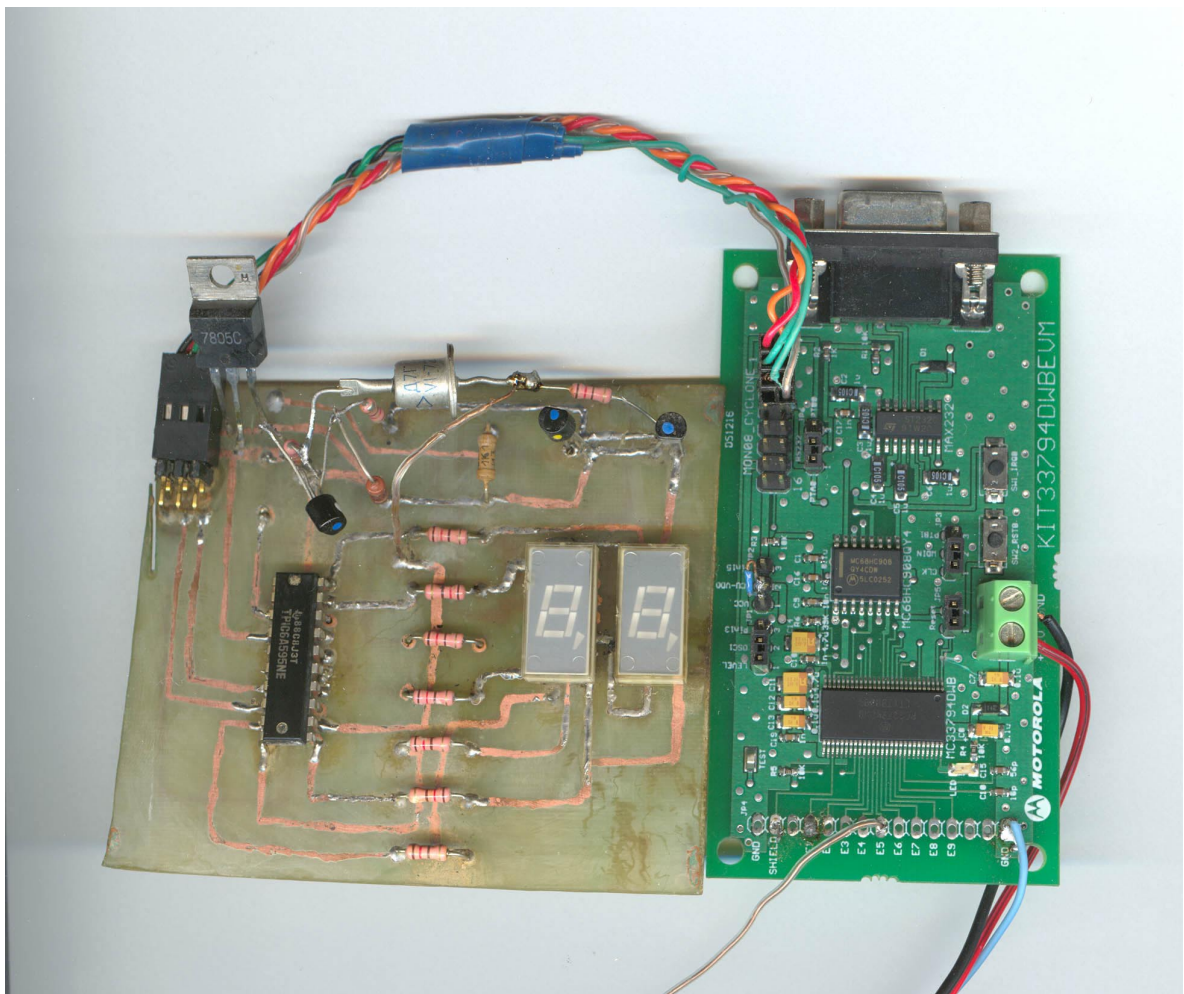


Figure 7. Device photograph

MCU firmware listing

```
#include <hidef.h>
#include <startup.h>
#include <MC68HC908QY4.h>
#include <math.h>

// flash program definitions
#define FLASH_CONFIG_START_ADDRESS          0xEE00
#define FLASH_CONTROL_LOCATION             0x88
#define CPU_BUS_CLOCK                       13
#define ERASE_ROUTINE_ADDRESS              0x2806
#define PROGR_ROUTINE_ADDRESS              0x2809

// indicators' definitions (PIC6A595NE - shift register)
#define SRCLK PTB_PT0
#define RCLK  PTA_PTA0
#define SDI   PTB_PT0

// data location for Flash programming
static struct
{
    byte CTRLBYT;
    byte CPUSPD;
    word LADDR;
    char hmax;
    float C0;
    float alpha;
}
    flash_control @FLASH_CONTROL_LOCATION;

// Page in Flash used for storing calibration device data
unsigned char hmax @0xEE00;
float C0 @0xEE01, alpha @0xEE05;

// global variables
unsigned char hmax_RAM=25,x=0;
float C,C0_RAM=1,alpha_RAM=2;

__interrupt void _ADC_Interrupt(void){}
__interrupt void _KBD_Interrupt(void){}

__interrupt void _TOF_Interrupt(void){}
__interrupt void _TCH0_Interrupt(void){}

__interrupt void _TCH1_Interrupt(void){}
__interrupt void _IRQ_Interrupt(void){}

void SaveDataInFlash()
{
    DisableInterrupts;

    flash_control.CTRLBYT = 0;
    flash_control.CPUSPD = CPU_BUS_CLOCK;
    flash_control.LADDR = FLASH_CONFIG_START_ADDRESS+8;
    flash_control.hmax = hmax_RAM;
    flash_control.C0 = C0_RAM;
    flash_control.alpha = alpha_RAM;
    //Erase Page 0xEE00..0xEE39
    asm
    {
        ldhx #FLASH_CONFIG_START_ADDRESS
        jsr ERASE_ROUTINE_ADDRESS
    }
    //Save Data 0xEE00..0xEE08
    asm
    {
        ldhx #FLASH_CONFIG_START_ADDRESS
        jsr PROGR_ROUTINE_ADDRESS
    }
}
```

```

    }
    EnableInterrupts;
}

void send_byte(char byte){
    char i;
    RCLK=0;
    for (i=0; i<8; i++){
        SRCLK=0;
        if (byte & (1<<i)) SDI=1; else SDI=0;
        SRCLK=1;
    }
    RCLK=1;
}

void show_level(char level){ // shows every time only one digit (low or high)
    char i;
    char number[16]={0xF6,0x14,0xBA,0x9E,0x5C,0xCE,0xEE,0x94,0xFE,0xDE,
                    0xFC,0x6E,0xE2,0x3E,0xEA,0xE8};
    char radix=10;
    if (x) i=number[level/radix]|0; else i=number[level%radix]|1;
    send_byte(i);
    x=~x;
}

float get_subroot_expr(unsigned char U,unsigned char Uc){
    float x,y;
    x=(float)U;
    y=(float)Uc;
    return (x*x/(y*y)-1);
}

void main(void) {
    unsigned char Uc,U,level,Saved=1;
    unsigned int pause;

    DisableInterrupts;
    CONFIG1=0x3D;
    CONFIG2=0x00;
    EnableInterrupts;

    // Initiate I/O ports A and B
    DDRA_DDRA0=1;
    DDRA_DDRA3=0; // button on RST pin
    DDRB=0xFF; // port B is output

    PTAPUE=0x0C; // set internal pullups for buttons

    PTB=0xCF; // Choose Internal OSC, Enable Shield and Lamp Control on
for (pause=0;pause<20000; pause++); // wait while MC33794 ready data for ADC

    // Initiate ADC
    ADICLK=0x20; // ADC frequency = Bus clock / 2
    ADSCR=0x03; // AIEN=0 ADCO=0 CH4(PTA5)

    //get Internal OSC Voltage - U
    while (!(ADSCR&0x80));
    U=ADR;

    // Start measurement of liquid level
    PTB=0x5F; // Choose 5th Electrode, Enable Shield and Lamp Control on
    ADSCR=0x23; // AIEN=0 ADCO=1 CH4(PTA5)

    //Initiate Calibration Data after last saving program in Flash
    if (hmax==0xFF){
        hmax_RAM=25;
        CO_RAM=1;
        alpha_RAM=2;
        SaveDataInFlash();
    }
    pause=0; hmax_RAM=hmax;
}

```

```

for(;;) {
    // get data of last conversion ADC and start next conversion
    if ((ADSCR&0x80)){
        Uc=ADR;
        C=sqrtf(get_subroot_expr(U,Uc)); // calculate current capacitance on
electrode
        level=(unsigned char)((C-C0)/alpha);//calculate actual liquid level in tube
    }
    // request buttons
    buttons:
    show_level(level);// show current liquid level on LEDs
    switch(PTA&0x0C){
        case 0x08:
            alpha_RAM=(C-C0)/(float)hmax;
            level=hmax;
            Saved=0;
            goto buttons;break; // pressed MAX
        case 0x04:
            C0_RAM=C;
            level=0;
            Saved=0;
            goto buttons;break; // pressed MIN
        case 0x00:
            if (pause==1700) {hmax_RAM++;pause=0;} else pause++;
            if (hmax_RAM==100) hmax_RAM=1;
            level=hmax_RAM;
            Saved=0;
            goto buttons;break; // pressed MAX+MIN
        default:
            if (!Saved) {SaveDataInFlash();Saved=1;pause=0;}
    }
}; /* loop forever*/

```