Design and Development of 8051 Based Resistance Meter for Interfacing with Sensors

Neilay Khasnabish
Alumnus, University of Kalyani, Nadia, West Bengal

Abstract
The meter designed here uses a unique method to use 555-timer along with 8051 microcontroller to measure resistance with very high linearity and accuracy. This meter implements a method to reduce the possibility of polarization while measuring resistance changes of liquid.

Keywords: 8051, 555-timer, resistance meter

1. Introduction

The objectives of this research project are as follows:

1. To design a circuit the output of which varies with the change of the resistance under test.
2. The circuit should reduce the possibility of polarization while measuring resistance changes of liquid sensors.
3. The output should be linear.
4. The range of the meter should be wide.
5. To interface an 8051 microcontroller to the circuit and display the output to the LCD panel.
6. It should be able to do both absolute and relative measurements.

While designing the system, a 555-timer and a 8051-microcontroller have been used. The 555-timer is configured as astable- multivibrator. 8051 is used to measure the pulse duration, and display the result on LCD after calibration. 8051 was originally designed by Intel in 1980s using NMOS technology; it is a Harvard architecture, which has separate storage for instructions and data. The prominent manufacturers of 8051 are Philips, Intel, Atmel, etc.

The most significant difference between this measuring instrument and others is its special design for the usage with sensors. The program can be changed to calibrate this meter for studying accurately other parameters influencing the change of resistance of liquid or resistive sensors.

In this research project, all programs are written in C using Keil uVision4.

2. System Description

2.1. Hardware Description

The hardware circuit consists of two main components: 555-timer and 8051-microcontroller as shown in Fig. 1. The 555-timer is used to generate pulses and 8051 is used to measure the pulse duration which is again proportional to the resistance under measurement.

The astable-multivibrator produces square wave with high and low pulses. The time duration of high pulse duration $T_H = 0.693\times(R_1+R_2)\times C$ and the time duration of low pulse $T_L=0.693\times R_2\times C$. So, we can prove that time duration of low pulse is directly proportional to the resistance R_2 which is the sensor. And, also according to the principle of 555-timer based astable-multivibrator, the current through R_2 changes its direction periodically due to charging and discharging of the capacitor C; this avoids polarization while measuring resistance changes of liquid. The microcontroller unit is used to measure this low time duration ($T_L=0.693\times R_2\times C$) and hence the resistance is calculated. Here R_1=10 kilo-ohm and C=100 microfarad. Varying the capacitance of C, we can adjust the range of the meter. So the range of the meter depends on the value of C and bit length of microcontroller. Here C is 100 microfarad and 8051 is 8-bit; the range of our measurement is 100 Ohm to 57 Kilo-Ohm. For R_2=100 Ohm, T_L=7 millisecond. Thus, R_2=10 Ohm cannot be measured because T_L will be 0.7 millisecond, and the system can ideally measure 1 millisecond at least. The variable which holds the value of

resistance is declared as *unsigned integer*. So it is a 16 bit variable and its range is 0-65535. Hence, using higher bits, we can increase range.

The output of the 555-timer is fed to 7404 Hex Inverting Gates chip to invert. So T_L is now converted to high pulse. The 8051 microcontroller measures the high pulse duration from 7404, using its timer and interrupt, which is a direct measure of T_L of 555-timer; this high pulse duration is the linear measure of the resistance under measurement (R_2). The maximum propagation delay of 7404 is about 20 nanoseconds.

The output from 7404 is fed to the 8051 microcontroller's INT0 (P3.2) and T0 (P3.4) pins. The program downloaded into the 8051 precisely calculates the high pulse duration from the hex inverter 7404's output, calculates the resistance value, and then displays it on the LCD.

For this application, an NXP Semiconductor's P89V51RD2 8-bit 64-kB flash microcontroller has been used. The flash program memory supports both parallel programming and in serial ISP (In System Programming). It has both TTL and CMOS compatible interface. Clock frequency used here is 11.0592 MHz.

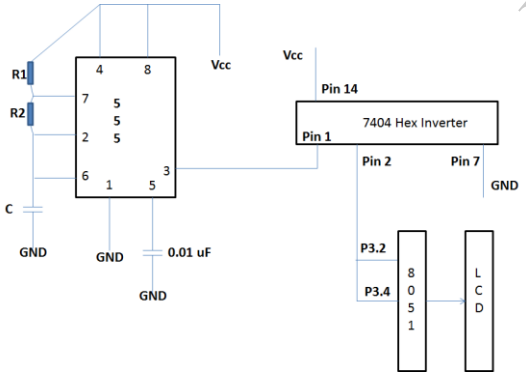

Fig. 1: CIRCUIT BLOCK DIAGRAM

The prototype-design is made using breadboard as well as 8051 development board as shown in Fig. 2. The ICs on the breadboard are powered from the regulated voltage sources mounted on the 8051 development board.

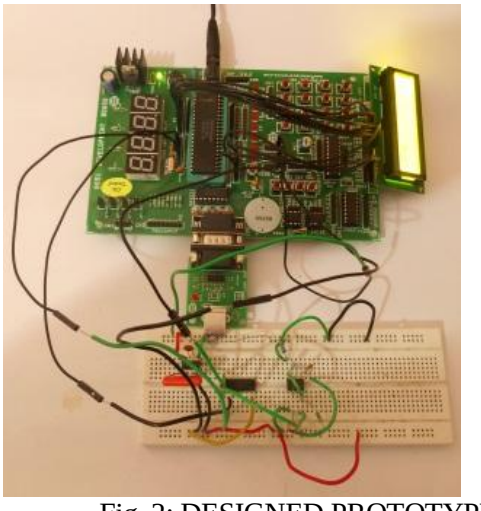

Fig. 2: DESIGNED PROTOTYPE

2.2. Software Description

KEIL MICROVISION4: It provides IDE for programming 8051 family microcontrollers. It supports 8051 programming in C. It also includes a simulator. The HEX file, which Keil Microvision4 generates is needed to be downloaded into 8051, using a programmer.

FLASH MAGIC: Flash Magic is used to download the HEX file into the 8051-microcontroller, using Serial Port or USB to Serial Converter. This software is widely used for programming flash based microcontrollers from NXP.

3. Algorithm Developed

This algorithm is written in C. The size of the HEX file is 2.95 kB.

THE ALGORITHM:

- Three key variables R, T0_ISR_Count and flag are declared globally. Initially all variables have 0.
- Main program starts with LCD initialization.
- External Interrupt 0 (P3.2) is configured for Edge Triggered feature.
- TMOD for Timer 0 (P3.4) is configured for GATE enabled and 16 bit Mode 1. The timer counts every millisecond.
- The Main function continuously checks for any value in the variable T0_ISR_Count

through flag variable using a while loop.

- Timer 0 gets started immediately when high pulse arrives at pin T0 (P3.2) and continues till the pulse is high. In this time, 8051 calculates calculate the time duration in millisecond, using timer 0 interrupt.
- As soon as the high pulse triggers from high to low at INT0 (P3.4), the interrupt gets activated, and the respective Interrupt Service Routine (ISR) is called.
- The ISR turns off the timer T0 and raises the flag variable high.
- As the flag variable is high, the program counter enters the while loop. The codes inside the while loop does the followings sequentially:
    - Deactivates all interrupts and timer.
    - Calculates **R=T0_ISR_Count/(Capacitance of capacitor C X 693)**.
    - Displays value of R in LCD.
    - Clears T0_ISR_Count .
    - Reloads Timer with predefined values to count milliseconds.
    - Flag variable is set 0 again.
    - Restarts timer T0.
    - Re-enables all interrupts.
- The same steps are repeated over and over again, and the LCD continuously displays the resistance value.

THE PSEUDOCOE:

```
include header file

unsigned integer T0_ISR_count = 0;
unsigned integer R=0;
integer flag=0;

function TIMER0 (void) interrupt
{
T0_ISR_count++;
Timer_Flag_Bit(TF0)=0;
Timer_High_Byte_Register(TH0)=0xFC;
Timer_Low_Byte_Reg(TL0)=0x67;
Timer_Start_Bit(TR0)=1;
}

function INT0 (void) interrupt
{
TR0=0;
flag=1;
}

function lcd display(value)
{
LCD Display Code
}

void main()
{
Initialise LCD;
Enable External Interrupt 0 with Edge Trigger;
TMOD = 0x09;  //Mode 1 T0 GATED
TH0=0xFC;
TL0=0x67;
TR0=1;
while (1)
{
if(flag == 1)
{
Disable all interrupts;
R=(T0_ISR_count/(Capacitor Value*693));
Call function lcd display (R);
TH0=0xFC;
TL0=0x67;
flag=0;
Start timer 0;
Enable all interrupt;
}
}
}
```

4. Results

Experiments have been conducted with different resistance values. The non-standard values of resistances are made with the help of 20K potentiometer. The values are measured with commercially available standard multi-meter first, and then with the 8051-based Resistance meter. The resistance values of resistors are re-checked with their color code. Then the percentage of error with respect to the resistance value measured with commercially available multi-meter is also studied. In Table. 1, column **M1** contains the Multi-meter measured resistance values and **M2** contains the 8051-based measured values of resistances. The formula used to calculate percentage of error is given below:

% of Error = ((M1-M2)/M1) ×100

The experimental result is given in the Table. 1 below:

Sl. No.	M1 (Ohm)	M2 (Ohm)	Magnitude Value of % of Error
1	101	101	0
2	334	340	1.7
3	1004	1010	0.5
4	2230	2251	0.9
5	4170	4184	0.3
6	5200	5255	1.0
7	5780	5815	0.6
8	5990	5901	1.4
9	6920	6969	0.7
10	7200	7287	1.2
11	10169	10245	0.7
12	11543	11399	1.2
13	13680	13896	1.5
14	17500	17734	1.3
15	22000	22251	1.1
16	47000	47388	0.8
17	57900	58311	0.7

Table. 1: EXPERIMENTAL DATA

The maximum magnitude value of percentage of error is 1.7%, and hence this system can be used to measure resistance changes of sensors.

5. Conclusion

The accuracy of measurement can be further increased with a microcontroller having higher bits and higher clock speed. The ideal capacitance values considered in the equations to calculate (100 microfarad) resistance gets deviated from their real values, influencing the results. The inverter chip 7404 has also its own delay time, which is almost negligible.

Also, the time duration is calculated in milliseconds in this program. To improve the accuracy, higher bit microcontrollers are required where time duration can be calculated in microseconds. So, using a higher bit-microcontroller and higher clock speed and the same algorithm and hardware configuration, results can be obtained with much higher accuracy and range.

6. Acknowledgments

This research is done independently by the author without involving any organisations.

7. References

[1] Mazidi, M., Mazidi, J.,McKinlay, R., 2007, The 8051 Microcontroller and Embedded Systems Using Assembly and C, Pearson Education.

[2] Kiuchi, N., Murakami, N., Segawa, H., Tazawa, I., Tominaga, C., Mar 28, 1989, US4816748 A, US Patent.