VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF INFORMATION TECHNOLOGY
FACULTY OF COMPUTER ENGINEERING

MID TERM CAPSTONE PROJECT
TEMPRATURE MEASUREMENT

(MÁY ĐO NHIỆT ĐỘ)

MENTOR: PhD. TRI NHUT DO

Nguyễn Thành Tài 21522568
Nguyễn Đình Sơn 21522554

CLASS: CE103.N22.MTCL

HO CHI MINH CITY, APRIL 2023

Table of contents

LIST OF FIGURES 3

Chapter 1. INTRODUCTION 5

Chapter 2. AN OVERVIEW 6

2.1. Research directions 6

2.2. Problem 8

Chapter 3. THEORY BASIS 10

3.1 8051 microprocessor overview and 8051 family 10

3.2. Devices 11

3.2.1. AT89C52 microcontroller 11

3.2.2. Temperature Sensor LM35 13

3.2.3. LCD 16×2 (LM016L) 15

3.2.4. Analogue-to-Digital Converter (ADC0804) 18

3.2.5. Pot 20

3.2.6. Button 20

3.2.7. DC Voltmeter 21

Chapter 4. SYSTEM IMPLEMENTATION PROCESS 23

4.1. Design 23

4.1.1 Principle diagram 23

4.1.2 Algorithm flowchart 23

4.2 Implementation 24

4.2.1 Draw circuit on Proteus 24

4.2.2 Programming on KeilC 29

Chapter 5. EXPERIMENT AND EVALUATION 36

5.1 Predict the outcome 36

5.2 Experiment 36

5.3 Evaluation 36

Chapter 6. SUMMARY, RESTRICTIONS AND DEVELOPMENT ORIENTATIONS 37

6.1. Summary 37

6.2. Restriction 37

6.3. Development Orientations 37

REFERENCES 38

LIST OF FIGURES

Chapter 1

1.1 Temperature measurement device

Chapter 2

2.1 Non-contact temperature measuring device helps to identify high body temperature.

2.2 An Automatic Temperature Measuring Device for Refrigerators.

Chapter 3

3.1 Introduction to 8051 Microcontroller

3.2 8051 family

3.3 AT89C52 and datasheet

3.4 AT89C52 features

3.5 Port 3 AT89C52

3.6 Temperature Sensor LM35

3.7 LM35 pins

3.8 LM35 features

3.9 LM016L

3.10 LM016L datasheet

3.11 Communication between LCD and microcontroller

3.12 ADC0804

3.13 ADC0804 datasheet

3.14 Pot and its basic connection

3.15 Button

3.16 Basic Button LED Circuit

3.17 DC Volmeter and its circuit diagram

<u>Chapter 4</u>

4.1 Principle diagram

4.2 Algorithm flowchart

4.3 Proteus design suite

4.4 Circuit

4.5 LM35

4.6 ADC0804 and pot

4.7 Formula

4.8 A89C52

4.9 LCD 16x2

4.10 Buttons

4.11 Keil

4.12 LCD Initialization

Chapter 1. INTRODUCTION

Currently, the environment has a great impact on human life, especially temperature. Therefore, temperature measurement was born, helping people to know the temperature of their surroundings. This is a multi-function machine, not only used to measure the ambient temperature, it is also widely used in measuring the surface temperature of objects such as bath water, room temperature, operating machinery temperature, even the temperature of fast food is accurate within seconds, ensuring safety and hygiene. In addition, the temperature meter is a basic premise for humans to create devices that regulate the ambient temperature such as air conditioners, heaters, etc. to help serve life and all activities more convenient.

Temperature measurement is an electronic device that uses a sensor capable of sensing infrared radiation from emitted sources. Especially the handheld measuring devices is very convenient and useful for electricians, mechanics, construction inspectors, machine maintainers, etc.

On the market, there are many types of temperature gauges with different parameters such as: infrared thermometer, laser temperature gun, remote temperature measuring device, ... of many famous manufacturers in the world.

1.1 Temperature measurement device

Based on their popularity and wide use, we decided to design a simple, low-cost temperature measuring device that can measure the temperature range from 0 to 150 degrees Celsius and convert the measured temperature between the three basic temperature scales, Celsius, Fahrenheit, and Kelvin. To be able to realize this device, we use AT89C52 microcontroller, LM35 temperature sensor, ADC0804 converter, LCD LM016L and 3 buttons according to 3 temperature scales.

Chapter 2. AN OVERVIEW

2.1. Research directions

2.1.1. Research directions in the world

Currently, there are many research directions on temperature measuring devices around the world, including some of the following:

Developing smart temperature measuring devices: Researchers are focusing on developing smart temperature measuring devices, allowing for the collection and transmission of temperature data to other devices to monitor and control temperature in automated systems.

Integrating IoT technology into temperature measuring devices: Researchers are developing temperature measuring devices that integrate with Internet of Things (IoT) technology, allowing for the collection of temperature data from multiple sources and automatic adjustment of temperature according to specific requirements.

Using new materials to measure temperature: Researchers are studying new materials for temperature measurement, such as special conductive materials or materials that change color according to temperature.

Research on temperature and health: Researchers are developing special temperature measuring devices to measure human body temperature, to assist in the diagnosis and treatment of temperature-related illnesses.

Research on temperature in industrial applications: Researchers are developing special temperature measuring devices to measure temperature in industrial applications such as production, processing, storage, and transportation of goods.

<u>For example:</u>

The non-contact temperature measuring device helps to identify high body temperature, which has been widely applied and used during the COVID-19 pandemic.

2.1 Non-contact temperature measuring device helps to identify high body temperature.

2.1.2. Research directions in Viet Nam

Currently, in Vietnam as well as around the world, there are many research directions on temperature measuring devices. However, there are some temperature measuring products that have been researched and developed by Vietnamese themselves. Some outstanding products are:

"3-in-1 Temperature Measuring Device for Food" - This product is developed by the Vietnamese Medical Equipment Import-Export and Production Company. The device can not only measure the temperature of food, but also measure the temperature in the oven and in boiling water.

"Automatic Temperature Measuring Device for Refrigerators" - This product is developed by Ocam Technology Joint Stock Company. The device can automatically measure the temperature inside the refrigerator, send data to a mobile application, and alert when the temperature is too high or too low.

"Temperature Measuring Device with Sound Feedback" - This product is developed by public health students at the Ho Chi Minh City University of Medicine and Pharmacy. The device uses a temperature sensor and has the function of emitting sound when the temperature is too high or too low.

"Wireless Temperature and Humidity Measuring Device" - This product is developed by Quang Minh Technology Company Limited. The device can measure temperature and humidity in the air, send data to a mobile application, and can be used in applications such as monitoring air quality in classrooms or offices.

"Temperature Measuring Device in Agricultural Production" - This product is developed by Agriculture Technology and Solution Company Limited. The device has the ability to measure temperature and humidity in the soil, helping farmers adjust watering and caring methods for crops accurately.

For example: An Automatic Temperature Measuring Device for Refrigerators.

2.2 An Automatic Temperature Measuring Device for Refrigerators.

2.2. Problem

There are some important problems to consider when researching temperature measuring devices worldwide and in Vietnam in particular, including:

Measurement quality: This issue relates to the accuracy and reliability of temperature measuring devices. To ensure measurement quality, knowledge of how the measuring device operates, the location of the device, and how to use and maintain it are necessary.

Product diversity and applications: Temperature measuring devices are used in many different fields, from medicine to industry and agriculture. Therefore, research should focus on developing diverse temperature measuring products and widely applicable uses.

Technology development: Temperature measuring technology is developing rapidly, with the emergence of smart, automatic, and Internet-connected devices. Research should focus on developing new temperature measuring devices and improving existing products to meet the increasingly high demand of the market.

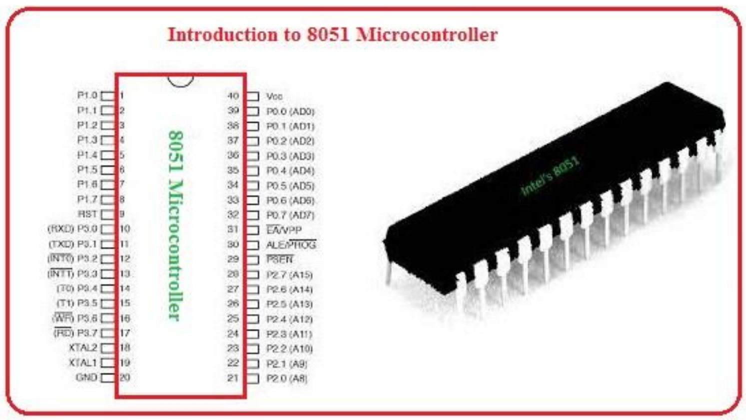
Cost issues: Temperature measuring devices have different prices depending on their features and accuracy. While cheaper products may be used for general purposes, higher-end products with greater accuracy and diversity of features may be more suitable for specialized applications or scientific research.

➔ **In this project**, we built an ambient temperature measuring device with low cost and high accuracy (but there are still negligible numbers). The instruments that we use are inexpensive devices and we try to make the measurement quality of the equipment highly accurate. Thereby creating a product that measures temperature with low cost but high efficiency.

Chapter 3. THEORY BASIS

3.1 8051 microprocessor overview and 8051 family

The 8051 microcontroller is one of the most popular 8-bit microcontrollers widely used in embedded applications. It was introduced by Intel in 1980 and has become the industry standard for embedded applications. The 8051 microcontroller features support for large memory, power-saving modes, multitasking, multithreading, and other functions.



3.1 Introduction to 8051 Microcontroller

The 8051 family includes:

8051: This is the original version of the 8051 microcontroller. It has 128 bytes of internal memory, 4KB of ROM, and 128 bytes of RAM.

8031: This is an improved version of the 8051 microcontroller. It has an additional internal timer and an external timer to support memory expansion.

8052: This is an extended version of the 8051 microcontroller. It has more ROM, more RAM, and an external timer.

The 8051 microcontroller and its family are still widely used in modern embedded applications, and there are many types of microcontrollers based on it, such as the **89C51**, **89C52** microcontroller.

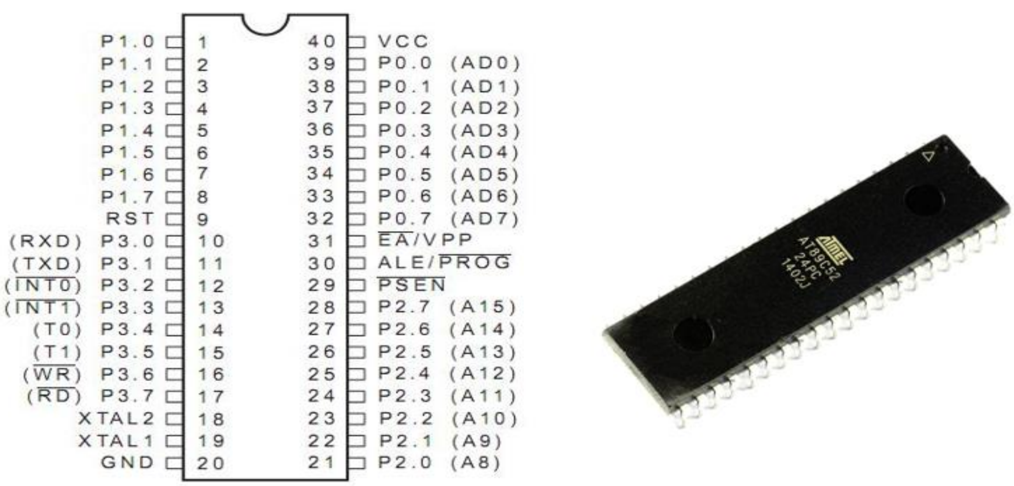
Feature	8051	8052	8031
ROM (on-chip program space in bytes)	4K	8K	0K
RAM (bytes)	128	256	128
Timers	2	3	2
I/O pins	32	32	32
Serial port	1	1	1
Interrupt sources	6	8	6

3.2 8051 family

3.2. Devices

3.2.1. AT89C52 microcontroller

In this project, we use a microcontroller in the 8051 family to build the device, **AT89C52**. **AT89C52** is a type of microcontroller based on the 8051 architectures from Atmel Corporation. It is an upgraded version of AT89C51 and has improved features such as larger memory, faster processing speed.



AT89C52 and datasheet

3.3 AT89C52 and datasheet

AT89C52 has a larger memory than AT89C51, with 8KB of reprogrammable Flash memory and 256 bytes of RAM. It operates at a frequency of 1-24 MHz. Because of that, **AT89C52** is widely used in embedded applications, such as motor control, lighting control, measurement devices, remote control systems, and other embedded systems.

Features

- Compatible with MCS-51™ Products
- 8K Bytes of In-System Reprogrammable Flash Memory
- Endurance: 1,000 Write/Erase Cycles
- Fully Static Operation: 0 Hz to 24 MHz
- Three-level Program Memory Lock
- 256 x 8-bit Internal RAM
- 32 Programmable I/O Lines
- Three 16-bit Timer/Counters
- Eight Interrupt Sources
- Programmable Serial Channel
- Low-power Idle and Power-down Modes

3.4 AT89C52 features

AT89C52 is an 8-bit microcontroller with 40 pins. Each pin has its own specific function, which can be categorized as follows:

1. VCC: This is the serial power supply pin.

2. GND: This is the ground pin.

3. XTAL1 and XTAL 2: These are the input and output pins for the external oscillator circuit.

4. RST: This pin is used to reset the microcontroller.

5. PSEN: This is the program to store enabled output. It is used to enable program memory.

6. ALE: This is the address latch enables output. It is used to latch the address from the external devices to the microcontroller.

7. EA: This is the external access enable and programming voltage input. It is used to enable external program memory.

8. **PORT P0 (P0.0 to P0.7)**: PORT 0 has 2 functions:

- I/O function: pins in PORT 0 are used as input and output.

- Data bus and address bus function (AD0 to AD7): 8 pins of PORT 0 are also responsible for getting data from ROM or external RAM (if there is an external connection), and these 8 pins are also used for external addressing.

9. PORT P1 (P1.0 to P1.7): PORT 1 is only used as input and output.

10. PORT 2 (P2.0 to P2.7): PORT 2 has 2 functions:

- I/O function: pins in PORT 2 are used as input and output.

- High address bus function (A8 to A15): When connecting to external memory with large memory, 2 bytes are required to address the memory, the low byte is taken by the pins of P0, and the high byte is taken by the pins of P2.

11. PORT 3 (P3.0 to P3.7): PORT 3 has 2 functions.

- I/O function: pins in PORT 3 are used as input and output.

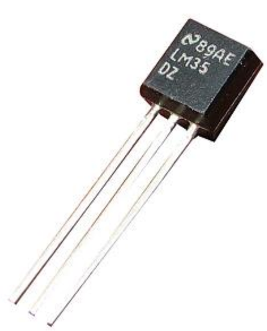
- For each pin of PORT 3 there is a second dedicated function as shown in the following table:

Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{\text{INT0}}$ (external interrupt 0)
P3.3	$\overline{\text{INT1}}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	$\overline{\text{WR}}$ (external data memory write strobe)
P3.7	$\overline{\text{RD}}$ (external data memory read strobe)

3.5 Port 3 AT89C52

3.2.2. Temperature Sensor LM35

To measure the temperature, we use **LM35 temperature sensor**. **LM35 temperature sensor** is a type of temperature sensor used to measure temperature in electronic applications. It is designed to measure temperature with high accuracy and high resolution, with excellent stability and temperature stability.



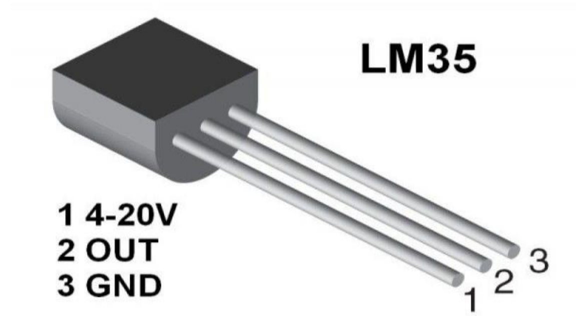
3.6 Temperature Sensor LM35

LM35 temperature sensor has 3 pins:

Pin 1 (Vs): This pin is used to provide power supply to the sensor. It should be connected to a voltage source in the range of 4V to 20V.

Pin 2 (VOUT): This pin is the output pin of the sensor. It provides an analog voltage output proportional to the temperature being sensed. The output voltage increases by 10mV per degree Celsius rise in temperature.

Pin 3 (GND): This pin is connected to the ground of the circuit to complete the power supply loop.



3.7 LM35 pins

LM35 operates based on the electronic thermal effect, in which temperature changes will create a corresponding voltage. And these are some features of LM35 temperature sensor:

Features

- Calibrated directly in ˚ Celsius (Centigrade)
- Linear + 10.0 mV/˚C scale factor
- 0.5˚C accuracy guaranteeable (at +25˚C)
- Rated for full −55˚ to +150˚C range
- Suitable for remote applications
- Low cost due to wafer-level trimming
- Operates from 4 to 30 volts
- Less than 60 µA current drain
- Low self-heating, 0.08˚C in still air
- Nonlinearity only ±¼˚C typical
- Low impedance output, 0.1 Ω for 1 mA load

3.8 LM35 features

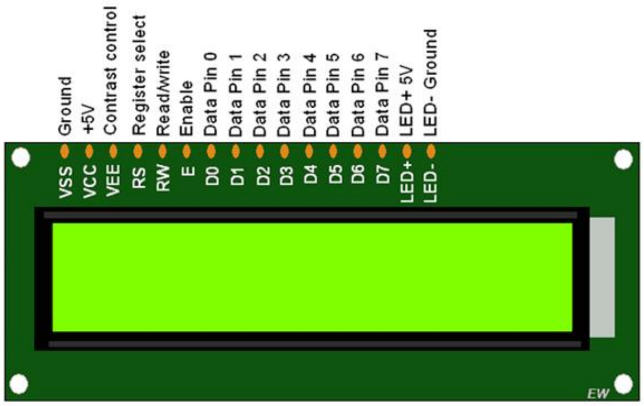
LM35 is a very common type of sensor and widely used in electronic applications, including temperature measurement applications in temperature control systems, temperature-related applications in industry, and medical applications.

3.2.3. LCD 16×2 (LM016L)

To display the temperature that we have measured and interact with users, we use the **LCD LM016L**. **LCD LM016L** is a 16-character, 2-line digital clock display (LCD) widely used in electronic applications. It is designed to display characters and numbers on the screen with high clarity and contrast. **LCD LM016L** is a popular product and is used in many electronic applications, including measurement, control, and display of information.

3.9 LM016L

The LM016L display has a size of 80mm x 36mm and is connected through an 8-bit or 4-bit parallel interface to microcontrollers such as **8051**, AVR, PIC, and ARM. It can display alphanumeric characters, numbers, and special characters, with some additional features such as displaying blinking cursor and scroll bar.



LCD LM016L datasheet

3.10 LM016L datasheet

The **LCD LM016L** is a 16x2 character LCD display screen with 16 connection pins, including:

VSS (Pin 1): This pin is connected to the GND level and is used to supply negative power to the screen.

VDD (Pin 2): This pin is connected to a 5V DC power source and is used to supply positive power to the screen.

VEE (Pin 3): This pin is connected to the screen brightness control signal. The voltage at this pin will affect the brightness of the screen. When VEE is connected to GND, the screen will display with maximum brightness.

RS (Pin 4): This pin is used to specify the content of data sent to the screen. When RS is LOGIC HIGH, data sent to the screen will be decoded as characters. When RS is LOGIC LOW, data sent to the screen will be decoded as control commands.

RW (Pin 5): This pin is used to specify the type of data transfer between the microcontroller and the screen. When RW is LOGIC HIGH, data is read from the screen (Read Mode). When RW is LOGIC LOW, data is written to the screen (Write Mode).

E (Pin 6):

Enable pins (Enable). After the signals are placed on the DB0-DB7 bus, commands are accepted if there is an enabled pulse of the E pin.

- In write mode: The data on the bus will be transferred (accepted) by the LCD into its internal register when a pulse (high-to-low transition) of the signal pin E is detected.

- In read mode: Data will be output by LCD to DB0-DB7 when detecting low to-high transition at pin E and kept by LCD on bus until E pin goes low.

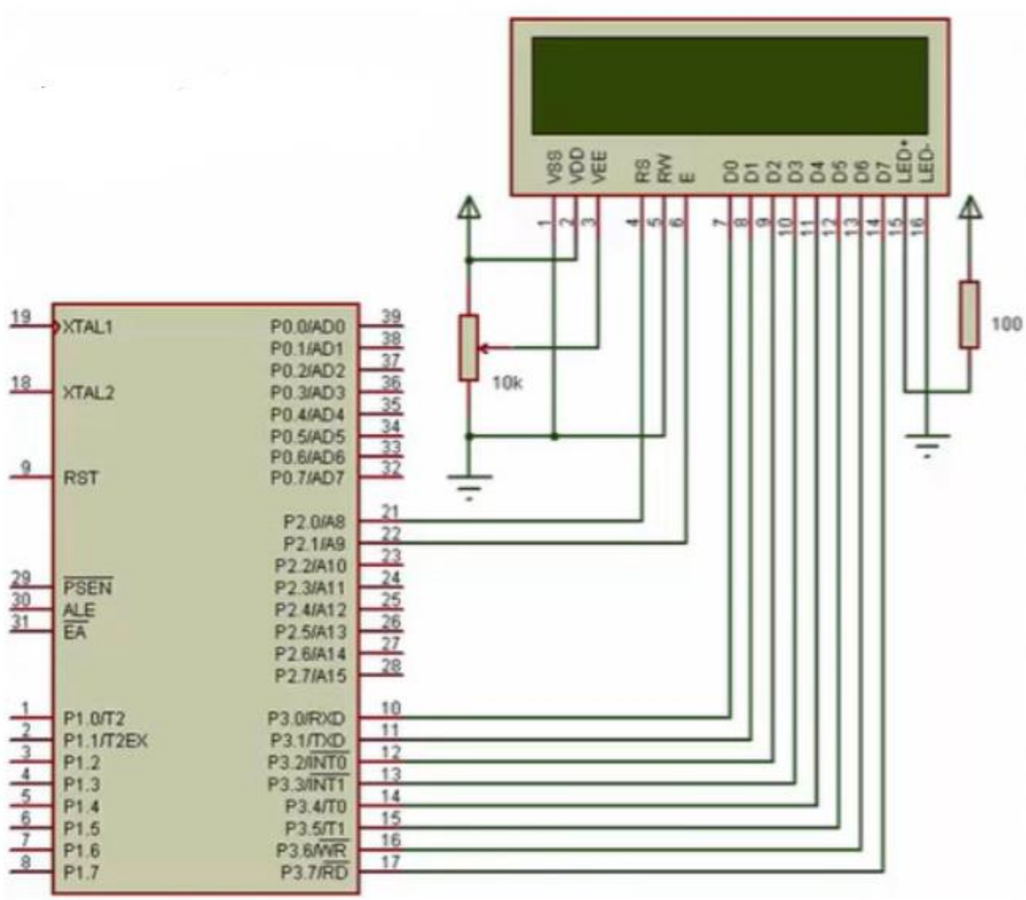
D0-D7 (Pins 7-14): These are the data pins corresponding to the bits of the 8-bit data sent to the screen.

LED+ (Pin 15): This pin is used to connect to the LCD backlight.

LED- (Pin 16): This pin is used to connect to the GND level for the LCD backlight.

Interface with LCD:

<u>Communication with microcontroller:</u>



3.11 Communication between LCD and microcontroller

Some LCD LM016L popular commands:

Code	Function	Time executes
0x01	Erase all content on LCD.	1.52ms
0x02	Move the cursor to the top of the screen.	1.52ms
0x06	Automatically move the cursor to the next position every 1-character LCD output.	37us
0x0C	Turn the display on and off the cursor.	37us
0x0E	Turn the display and the cursor on.	37us
0x80	Move the cursor to the beginning of line 1.	37us
0xC0	Move the cursor to the beginning of line 1.	37us
0x38	8-bit communication, 2-line display, font size 5x7.	37us

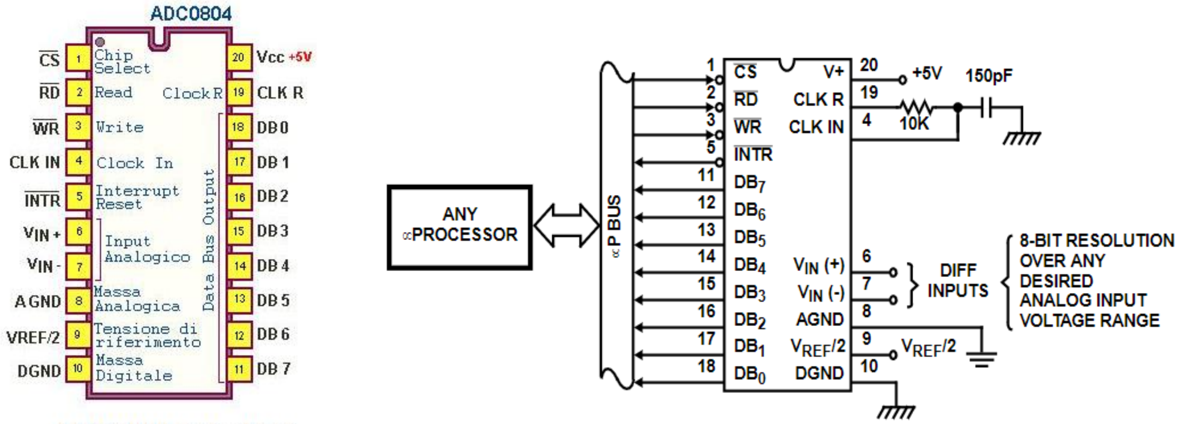
3.2.4. Analogue-to-Digital Converter (ADC0804)

Because the output of the **LM35** is an analog signal, and the input of the **8051** is a digital signal, they are not compatible with each other. Therefore, we need to use an analog to digital converter, namely the **ADC0804. ADC0804** is an 8-bit analog-to-digital converter (ADC) widely used in electronic applications. It can convert analog signals into digital signals with high accuracy and 8-bit resolution.

ADC0804 is a popular product and widely used in electronic applications, including measurement, control, and processing of analog signals.

3.12 ADC0804

ADC0804 can convert analog signals with an input range from 0 to 5V and has 8 input channels to convert signals (suitable for **8051**). It is controlled by a clock signal and uses several digital input/output lines to retrieve conversion data. Additionally, ADC0804 can also be controlled by microcontrollers such as **8051**, PIC, and AVR.



ADC0804 datasheet **Typical Application Schematic**

3.13 ADC0804 datasheet

The **ADC0804** has 20 connection pins, including:

CS (Pin 1): This pin is used if more than 1 ADC module is used. By default, is grounded.

RD (Pin 2): This pin must be grounded to read analog values.

WR (Pin 3): This pin must be high to initiate data conversion.

CLK IN (Pin 4): An external clock can be connected here; another RC can be used to access the internal clock.

INTR (Pin 5): Go high for interrupt request.

Vin+ (Pin 6): Differential analog input +. Connect to the ADC input.

Vin- (Pin 7): Differential analog input -. Connect to the ground.

AGND (Pin 8): The analog ground pin is the ground of the circuit.

VREF/2 (Pin 9): Reference voltage for ADC conversion.

DGND (Pin 10): Digital ground pin to ground of the circuit.

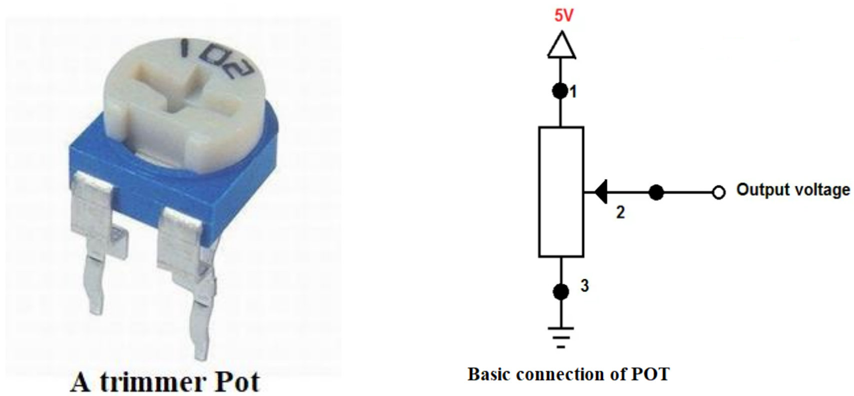
DB7 to DB0 (Pin 11 to Pin 18): 7 output data bit pins from which the output is obtained.

CLK R (Pin 19): RC timing resistor input pin for internal gen clock.

VCC (Pin 20): Power the ADC module, using +5V.

3.2.5. Pot

"**Pot**" in electronics stands for "**potentiometer**", which is an electronic component used to adjust and control the level of voltage or current in an electrical circuit. It is designed in the form of a rotating shaft or a sliding bar that can be moved and is connected to resistors with different values to create an adjustable output voltage or current. **Pots** are divided into many different types, including trimmer potentiometers, slide potentiometers, servo potentiometers….



3.14 Pot and its basic connection

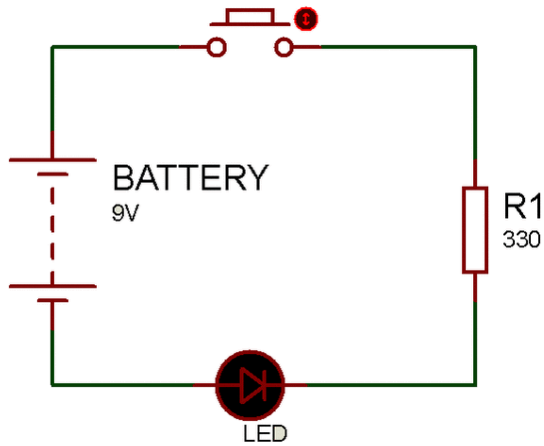
Potentiometers are widely used in electronic applications, including sound, measurement, control, and other electronic applications. They allow users to adjust the level of voltage or current as desired to meet the requirements of different applications.

3.2.6. Button

Because our device has some features. We use **buttons** to make it possible for users to interact and select some features of our device. A **button** is a simple electronic component used to generate an electrical signal in a circuit. It is commonly used to allow users to interact with electronic devices by providing input signals.

3.15 Button

When the **button** is pressed, it generates an electrical signal that can be used to activate or deactivate an electronic device or perform a certain function. It is widely used in electronic applications, including control, measurement, and communication.



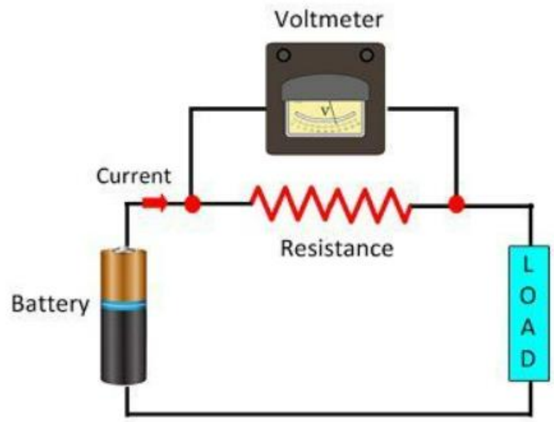
3.16 Basic Button LED Circuit

There are many different types of **buttons**, including single push buttons, double push buttons, round push buttons, and toggle switches. Each type of **button** has a different design and functionality to suit the requirements of different electronic applications.

3.2.7. DC Voltmeter

DC voltmeter is a device used to measure direct current (DC) voltage in an electrical circuit. It can be used to measure different voltage levels at various points in the circuit. To measure voltage using **DC voltmeter**, we need to connect the positive terminal of the **DC voltmeter** to the point in the circuit with higher voltage, and the negative terminal of the **DC voltmeter** to the point in the circuit with lower voltage, adjust the sensitivity level of the **DC voltmeter** to ensure the most accurate measurement and then read the voltage value on the **DC Voltmeter.**

DC Voltmeter 0-15V



DC Voltmeter circuit diagram

3.17 DC Volmeter and its circuit diagram

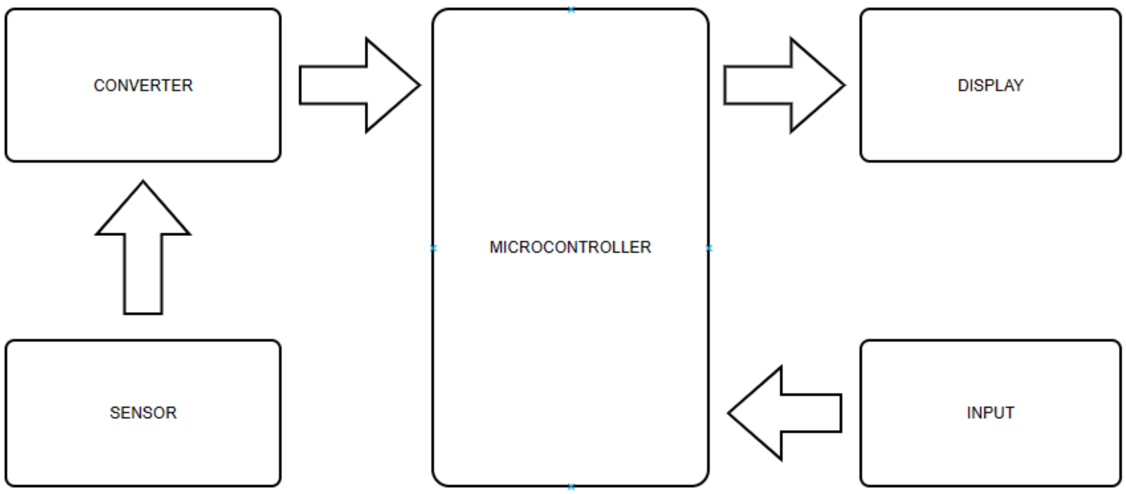
There are various types of **DC voltmeters**, including analog and digital. Analog DC voltmeters use a needle and index on a circular display to indicate the voltage value, while digital DC voltmeters use numeric displays or LCD screens to indicate the voltage value.

Chapter 4. SYSTEM IMPLEMENTATION PROCESS

4.1. Design

4.1.1 Principle diagram

Below is the most basic overview diagram, which helps readers get an overview of our project and understand its basic working principle.



4.1 Principle diagram

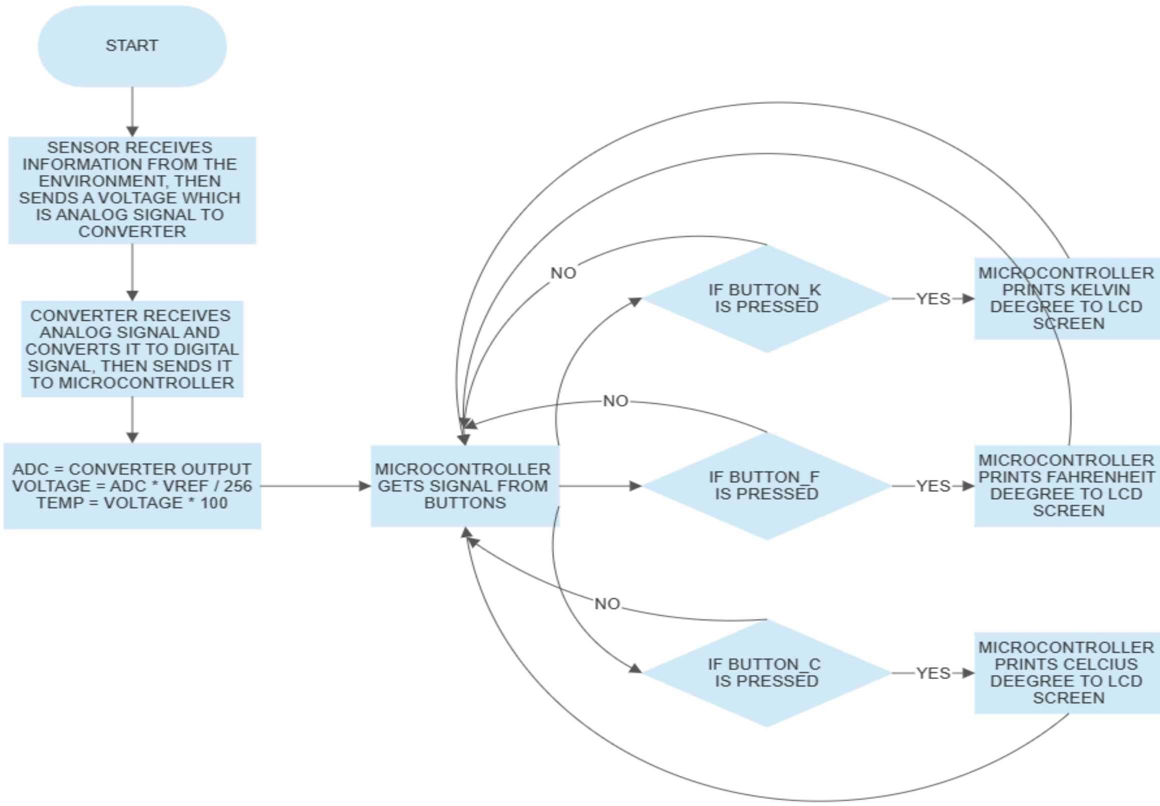
First, the temperature sensor will get information from the environment, its output will be a voltage which is an analog signal. Since the microcontroller input only receives digital information, we use a converter to convert the analog signal to the digital signal.

After the conversion is done, the result will be transferred to the microcontroller. After a few steps of conversion based on the relationship between current and temperature, the microcontroller will output the result on the display. This process will take place continuously with a fixed frequency. That means, the sensor will continuously receive temperature information from the environment and send it to the converter continuously, and the temperature displayed on the screen will also continuously change according to the temperature of the environment.

Not only measuring the ambient temperature, but we also provide an additional function that helps to convert between 3 popular temperature scales today widely used, which are Celsius, Kelvin and Fahrenheit, so there are some input devices to support this feature.

4.1.2 Algorithm flowchart

Based on the basic flowchart designed in section 4.1.1, we have designed an algorithmic flowchart to carry out the processes of receiving and transferring information as well as converting signals. This algorithmic flowchart will be converted into a programming language in the next sections.



4.2 Algorithm flowchart

(The formulas will be expained in the next sections)

4.2 Implementation

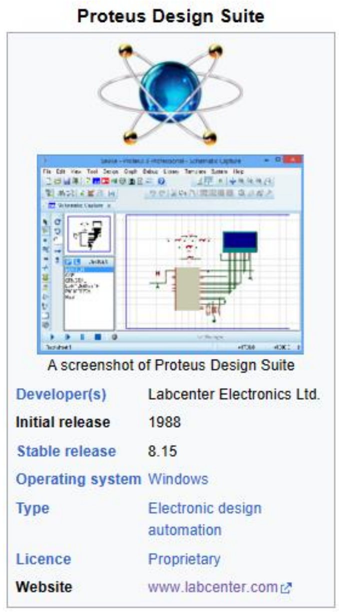
After successfully designing the basic principle diagram and algorithm flowchart for the device, we will begin to implement it on hardware, the simulation software used is Proteus 8.8

4.2.1 Draw circuit on Proteus

4.2.1.1 What is Proteus?

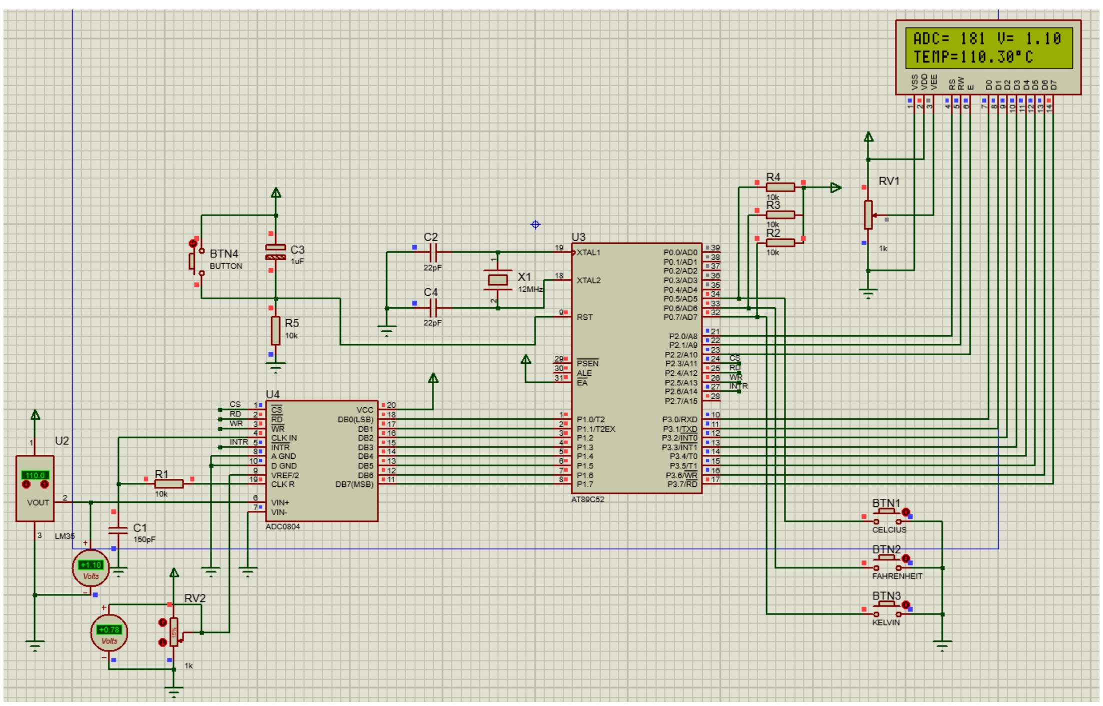
The Proteus Design Suite is a proprietary software tool suite used primarily for electronic design automation. The software is used mainly by electronic design engineers and technicians to create schematics and electronic prints for manufacturing printed circuit boards.

It was developed in Yorkshire, England by Labcenter Electronics Ltd and is available in English, French, Spanish and Chinese languages.



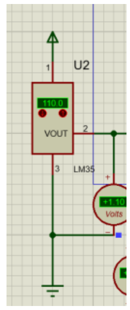
4.3 Proteus design suite

4.2.1.2 Draw circuit



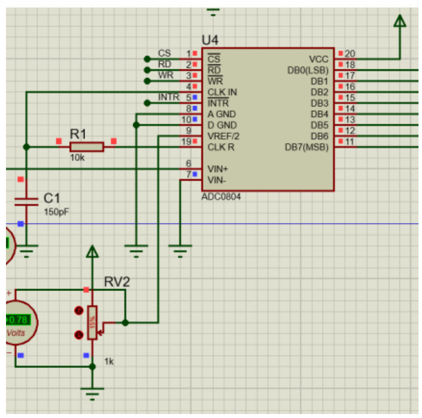
4.4 Circuit

4.2.1.3 Uses of each component



4.5 LM35

First, the sensor receives the temperature from the environment, creating a voltage to the VIN+ pin of the ADC0804. For every 1 degree Celsius increase/decrease on the sensor, the voltage will also increase/decrease 0.01V respectively. Which means Temp = Voltage *100.



4.6 ADC0804 and pot

At the ADC, the voltage will be converted to an 8-bit binary digital signal and fed to Port 1 of the microcontroller. So how did that transition happen? We have the following formula:

$$\frac{Resolution\ of\ the\ ADC}{System\ Voltage} = \frac{ADC\ Reading}{Analog\ Voltage\ Measured}$$

4.7 Formula

=> ADC Reading = Analog Voltage Measured * Resolution of the ADC / System Voltage

According to the above formula, we have the following quantities:

ADC Reading: ADC value we need to calculate.

Resolution of the ADC: ADC0804 can handle 8 bits of data, so the resolution is 2^8 = 256

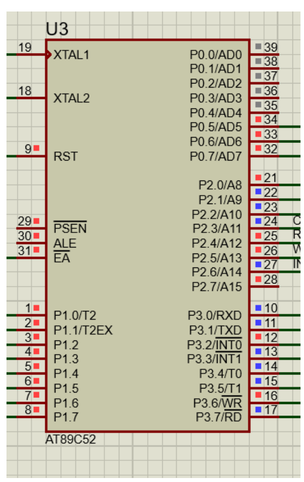
System Voltage: Reference voltage Vref, in simulation image (figure 4.4) the reference voltage is 0.78*2 = 1.56V

Analog Voltage Measured: Input voltage VIN+, in simulation image is 1.1V

From that we have ADC = 1.1 * 256 / 1.56 = 180.51

It can be seen that the calculated ADC value is approximate to the measured ADC value displayed on the LCD, which is 181, there is a small difference between the estimated ADC value and the measured ADC because the ADC0804 device only works with 8 bit numbers, results will be rounded. And this small difference is the cause of the measurement error in the conversion from voltage to temperature to display on the LCD screen and this error is inevitable.

Another important aspect of the ADC0804 causing measurement errors is the step size. The step size is closely related to the reference voltage Vref (System Voltage) and resolution of the ADC0804, calculated by the formula System Voltage/Resolution. The step size is proportional to the ADC value, if the input voltage is increased/decreased to a value equal to the step size, the ADC value will increase/decrease by 1 bit. For example, when Vref is at the default value and does not add a pot, it carries a value of 5V (the Vref/2 pin is applied 2.5V), now the calculated step size is 5/256 = 0.01953125V, which means for every 0.01953125V increase/decrease in the analog input signal, the ADC output value will increase/decrease by 1 bit. So the smaller this step size is, the higher the accuracy will be, but the Vref value must still be greater than or equal to the input voltage value. That's why in the simulation, we use a pot to adjust Vref/2 to 0.78V which means Vref is 1.56V because the maximum input voltage is 1.5V corresponding to 150 degrees Celsius.



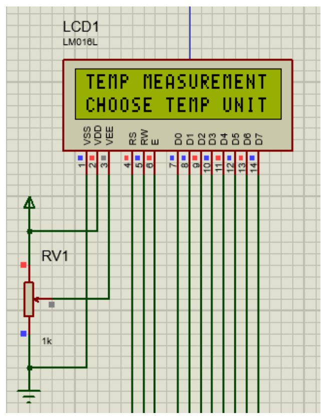
4.8 A89C52

Why is the step size value related to the temperature error? After converting the analog signals to digital signals and sending them to the microcontroller, we have to convert that digital signal back into a voltage value to print to the LCD screen. Take a look back at the formula in figure 4.7.

=> Analog Voltage Measured = ADC Reading * System Voltage / Resolution of the ADC

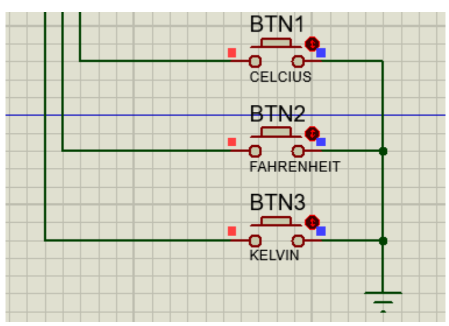
Analog Voltage Measured is the voltage that we need and **The System Voltage / Resolution of the ADC** value is the step size, they are related to each other.

So, the conversion process can be summarized as follows: First, the voltage from the sensor will be converted into an 8-bit binary value according to the formula shown in figure 4.7 at ADC0804. Next, this 8-bit value will be transferred to Port 1 of the microcontroller. Here, we use the formula 4.7 to convert the 8-bit binary value back to voltage current, and then the voltage current will be converted to temperature.



4.9 LCD 16x2

When there are 3 values are ADC value, voltage and temperature, we will print them to the LCD screen. The signal is transmitted from the microcontroller to the LCD screen at Port 3. Specific details of the process of printing data to LCD will be presented in the 4.2.2 section.



4.10 Buttons

During the display, the temperature scale can be customized using the 3 buttons. Corresponding to the selected scale type, the temperature value will be changed according to that scale and printed on the LCD screen. Specific details of the principle of operation of the buttons will be presented in the 4.2.2 section.

4.2.2 Programming on KeilC

4.2.2.1 What is keilC?

Keil is a German software subsidiary of Arm Holdings. It was founded in 1982 by Günter and Reinhard Keil, initially as a German GbR. In April 1985 the company was converted to Keil Elektronik GmbH to market add-on products for the development tools provided by many of the silicon vendors. Keil implemented the first C compiler designed from the ground-up specifically for the 8051 Microcontroller.

Keil provides a broad range of development tools like ANSI C compiler, macro assemblers, debuggers and simulators, linkers, IDE, library managers, real-time operating systems (currently RTX5) and evaluation boards for over 8,500 devices.

In October 2005, Keil (Keil Elektronik GmbH in Munich, Germany, and Keil Software, Inc. in Plano, Texas) were acquired by Arm.

Since the merger with Arm, the company is still active in providing products and services.

Keil

KEIL™ Tools by ARM

Industry	Embedded Software
Founded	1986
Headquarters	Munich; Plano
Products	compiler, assemblers, debuggers, linkers, IDE, real-time operating systems, evaluation boards
Website	www.keil.com

4.11 Keil

4.2.2.2 Programming

LCD Functions

```
sbit LCD_RS = P2^0;
sbit LCD_E = P2^2;
sbit LCD_RW = P2^1;
#define LCD_DATA P3
```

First, we connect the LCD pins to the microcontroller's pins: Register select (RS) pin connected to P2.0, Enable (E) pin connected to P2.2, Read/write (RW) pin connected to P2.1 and P3 used to exchange information with LCD.

```
void LCD_cmd(unsigned char cmd){        //thuc hien lenh
    LCD_RW = 0;         //cho phep ghi
    LCD_RS = 0;         //ghi lenh
    LCD_DATA = cmd;     //thuc hien lenh cmd
    //tao xung cho E
    LCD_E = 1;
    LCD_E = 0;
    //
    delay(2);
}
```

The most important part of LCD is command execution, LCD cannot work if we do not give instructions for LCD to execute.

So, to execute a command, how to configure LCD?

First, we set the RW pin of the LCD to 0 to allow the LCD to operate in write mode.

Next, the RS pin of the LCD must be set to 0 to enable command writing.

Next assign 8 LCD pins with command data.

Finally, we create a high to low transition pulse at pin E. When this pulse is detected, the instruction data will be transferred to the LCD instruction register, the instruction will be executed.

```
void LCD_char_cp(char ch){   // in ra 1 ki tu tai vi tri hien tai
    LCD_RW = 0;         //cho phep ghi
    LCD_RS = 1;         //ghi du lieu
    LCD_DATA = ch;      //ghi ki tu ch
    //tao xung cho E
    LCD_E = 1;
    LCD_E = 0;
    //
    delay(2);
}
```

As well as command execution, data display is also an integral part of LCD, to display data on LCD screen, we need to perform the following steps:

First, we set the RW pin of the LCD to 0 to allow the LCD to operate in write mode.

Next, the RS pin of the LCD must be set to 1 to enable data writing.

Next assign 8 LCD pins with data that we want to write.

Finally, we create a high to low transition pulse at pin E, When this pulse is detected, the data will be transferred to the LCD instruction register, and then it will be printed to the screen.

However, the above steps are just to display a character, to display a string of characters to the LCD screen, we need to add a function that includes many functions to print a character:

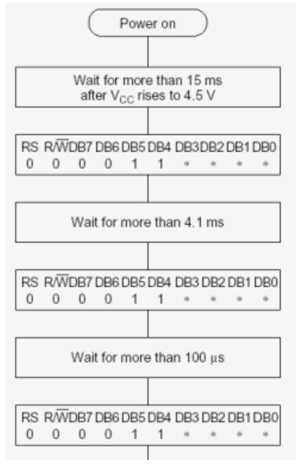
```
void LCD_string_cp(char *str){   //in string
    unsigned char i =0;
    while (str[i]!=0){
        LCD_char_cp(str[i]);
        i++;
    }
}
```

With a string entered, we will print each character of the string until there are no more characters to print.

Finally, once we have the basic functions of LCD, we will initialize and configure an LCD.

```
void LCD_init(){    //khoi tao LCD
    //khoi dong lcd
    LCD_cmd(0x30);
    delay(5);
    LCD_cmd(0x30);
    delay(1);
    LCD_cmd(0x30);
    //
    LCD_cmd(0x38);          //so dong la 2, 5x8, che do giao tiep 8bit
    LCD_cmd(0x01);          //clear screen
    LCD_cmd(0x0C);          //bat hien thi, tat con tro
    LCD_cmd(0x06);          //che do tu dong dich con tro
}
```

When the LCD has just been started, we must execute 3 0x30 instructions to initialize the LCD.



4.12 LCD Initialization

When the initialization process is complete, you can add a few basic commands depending on your preferences such as choosing communication mode, turning on/off cursor display, etc.

Once all the above steps are completed, our LCD is ready to go.

(For the use of LCD pins and the basic LCD instruction set, read Chapter 3).

ADC0804 Functions

```
// ADC
sbit ADC_CS = P2^3;
sbit ADC_RD = P2^4;
sbit ADC_WR = P2^5;
sbit ADC_INTR = P2^6;
#define ADC_DATA P1
```

First, we connect the ADC0804 pins to the microcontroller's pins: Chip select (CS) pin connected to P2.3, Read (RD) pin connected to P2.4, Write (WR) pin connected to P2.5, Interrupt (INTR) pin connected to P2.6 and ADC data connected to P1 is used to get 8-bit digital signal from ADC0804.

```
unsigned char ADC_read(){
    unsigned char kq;
    ADC_CS = 0;  // chon chip
    // tao xung bat dau chuyen doi
    ADC_WR = 0;
    ADC_WR = 1;
    //
    while (ADC_INTR);    // doi den khi chuyen doi xong
    // doc gia tri sau khi chuyen doi
    ADC_RD = 0;
    kq = ADC_DATA;
    ADC_RD = 1;
    //
    return kq;
}
```

First set the CS value to 0

Then we create a pulse at the WR pin to start the conversion. When the ADC is in the conversion process, the INTR will be 1 and it will go back to 0 when the conversion is over.

After successful conversion, we will read the conversion result. To read, first we switch the RD pin to 0, the read result will be returned to serve other functions in the program, after finishing reading, the RD pin will change to the value 1.

(For the use of ADC0804 pins, read Chapter 3).

Main

```
void main(){
    unsigned char adc_value;
    float t, v;
    LCD_init();
    LCD_cmd(0x80); //dua con tro ve dau hang 1
    LCD_string_cp("TEMP MEASUREMENT");

    LCD_cmd(0xC0); //dua con tro ve dau hang 2
    LCD_string_cp("CHOOSE TEMP UNIT");

    while(1){
        if(BUTTON_C==0){
            LCD_cmd(0x01); //clear screen
            LCD_string_cp("TEMP IN CELCIUS");
            delay(100);
            while(BUTTON_C==1 && BUTTON_F==1 && BUTTON_K ==1){
                adc_value = ADC_read();  //doc output cua ADC
                LCD_cmd(0x80);//dua con tro ve dau hang 1
                LCD_string_cp("ADC= "); printa(adc_value);  //in adc output
                LCD_string_cp(" "); LCD_string_cp("V= ");
                v = (float)adc_value * 1.56/256;
                printv(v); //in voltage
                if (BUTTON_C==0 || BUTTON_F==0 || BUTTON_K ==0) break;  //giam delay nut bam
                LCD_cmd(0xC0);//dua con tro ve dau hang 2
                LCD_string_cp("TEMP=");
                adc_value = ADC_read();
                t = (float)adc_value * 1.56/256 * 100;
                printtemp(t);
                LCD_cmd(0xCB);  //dua con tro den o thu 11 hang 2
                LCD_char_cp(0xDF);//in ki hieu nhiet do
                LCD_char_cp('C');
            }
        }
```

```
        if(BUTTON_F==0){
            LCD_cmd(0x01);  //clear screen
            LCD_string_cp("TEMP IN FAHRENHEIT");
            delay(100);
            while(BUTTON_C==1 && BUTTON_F==1 && BUTTON_K ==1){
                adc_value = ADC_read();  //doc output cua ADC
                LCD_cmd(0x80);//dua con tro ve dau hang 1
                LCD_string_cp("ADC= "); printa(adc_value);  //in adc output
                LCD_string_cp(" "); LCD_string_cp("V= ");
                v = (float)adc_value * 1.56/256;
                printv(v); //in voltage
                if (BUTTON_C==0 || BUTTON_F==0 || BUTTON_K ==0) break;    //giam delay nut bam
                LCD_cmd(0xC0);  //dua con tro ve dau hang 2
                LCD_string_cp("TEMP=");
                adc_value = ADC_read();
                t = (float)adc_value * 1.56/256 * 100 * 1.8 + 32;    //do F = do C*1.8+32
                printtemp(t);
                LCD_cmd(0xCB);  //dua con tro den o thu 11 hang 2
                LCD_char_cp(0xDF); //in ki hieu nhiet do
                LCD_char_cp('F');
            }
        }

        if(BUTTON_K==0){
            LCD_cmd(0x01); //clear screen
            LCD_string_cp("TEMP IN KELVIN");
            delay(100);
            while(BUTTON_C==1 && BUTTON_F==1 && BUTTON_K ==1){
                adc_value = ADC_read();  //doc output cua ADC
                LCD_cmd(0x80);//dua con tro ve dau hang 1
                LCD_string_cp("ADC= "); printa(adc_value);  //in adc output
                LCD_string_cp(" "); LCD_string_cp("V= ");
                v = (float)adc_value * 1.56/256;
                printv(v); //in voltage
                if (BUTTON_C==0 || BUTTON_F==0 || BUTTON_K ==0) break;   //giam delay nut bam
                LCD_cmd(0xC0);//dua con tro ve dau hang 2
                LCD_string_cp("TEMP=");
                adc_value = ADC_read();
                t = (float)adc_value * 1.56/256 * 100 + 273.15;  //do K = do C + 273.15
                printtemp(t);
                LCD_cmd(0xCB);  //dua con tro den o thu 11 hang 2
                LCD_char_cp(0xDF);//in ki hieu nhiet do
                LCD_char_cp('K');
            }
        }
    }

}
```

First, start the LCD and display the scale selection menu, use the LCD's position initialization commands to display the required character strings.

At this point, the user is allowed to select the temperature scale, using 1 of 3 buttons.

If the BUTTON_C button is pressed, we will first clear the LCD screen to erase the existing text on the LCD screen to avoid the display text being overwritten.

The message indicating the selected scale type will be displayed on the screen for about 100ms, this timeout period is to ensure that the user cannot continuously manipulate the button (spam) and avoid some button errors such as double click, bounce keys, …

When the waiting time is over, at this time, the button is sure to be released, as long as the button is not operated by the user, the screen will continuously display the temperature according to the scale that the user has pre-selected. The quantities ADC, voltage, temperature will be calculated according to the formula in the previous section and will be displayed and changed continuously according to the ambient temperature.

Because the execution time of processing statements in a while loop can be a bit long, checking whether the button is pressed or not will take place with a small frequency, so sometimes when the user presses the button, it doesn't work because it's not time to check the button press condition yet. To fix this, we added a condition in the middle of the while loop:

```
if (BUTTON_C==0 || BUTTON_F==0 || BUTTON_K ==0) break;    //giam delay nut bam
```

This will reduce the time from one button condition test to the next button condition test by half, avoiding button error.

You may wonder if the user presses the button, the program will exit the while loop with the above condition or the condition of the while loop itself, but to enter the next while loop and continue temperature displaying, user will have to press one more time. For example, if I am in C scale and want to display temperature in F scale, I press F button, I will just exit the while loop and I have to press F again to display F scale.

But no, the time to execute 2 instructions continuously in the program is very small, when I press the F button and have not even released it, the program has finished executing the break statement or checking the condition of the while loop, and start executing the 3 button test command, so the program will still work normally.

BUTTON_F and BUTTON_K have the same operating mechanism as BUTTON_C, with only a few small changes in the temperature conversion formulas which are:

$F = C * 1.8 + 32$

$K = C + 273.15$

In addition, some functions are needed to convert the calculated values into string or characters for easier LCD printing.

```
void printv(float t){        //in voltage
    sprintf(LCD_buff,"%0.2f", t);
    LCD_string_cp(LCD_buff);
}
void printtemp(float t){     //in nhiet do
    int i;
    sprintf(LCD_buff,"%0.2f", t);
    i = strlen(LCD_buff);
    while (i<6)
    {
        LCD_char_cp(' ');
        i++;
    }
    LCD_string_cp(LCD_buff);
}
void printa(unsigned int t){ //in ADC
    if (t/100!=0){ // Neu t >= 100
        LCD_char_cp(t/100+0x30); // In ra hang tram
        LCD_char_cp(t%100/10+0x30);  // In ra hang chuc
    }
    else if(t/100==0) // Neu t < 100
        if (t%100/10!=0) // Neu t > 9
            LCD_char_cp(t%100/10+0x30); //In ra hang chuc
    LCD_char_cp(t%10+0x30); // In ra hang don vi
}
```

Chapter 5. EXPERIMENT AND EVALUATION

5.1 Predict the outcome

As mentioned in Chapter 4, because this is a small-scale and low-budget project, errors in measurement and calculation are inevitable. We can only reduce the error to a certain extent, not completely eliminate it.

The measurement results may have a small error of 0.1 – 0.5 degrees Celsius.

5.2 Experiment

This is a simulation project, so there will be no experiments in the environment, we only simulate the measurement process on Proteus 8.8 software.

VIDEO: See attachment.

5.3 Evaluation

The device runs very well on the simulator. As we expected, this product has high accuracy, error is small, within the acceptable range. The button works perfectly fine, no button error, and has many functions to be able to interact with the user such as having buttons for each display function and being able to switch between function to alternate 3 scales: degrees Celsius, degrees Fahrenheit and degrees Kelvin. In conclusion, we have built the product that we had in mind and satisfied the requirements set forth, a temperature measuring device with low cost and high accuracy.

Chapter 6. SUMMARY, RESTRICTIONS AND DEVELOPMENT ORIENTATIONS

6.1. Summary

The temperature measuring product that we built has achieved certain successes in many aspects:

- This project helped us to understand more about 8051's family microprocessors and microcontrollers. Also, this project helped us to understand how to build a product by using microcontrollers and understand about how microcontrollers connect with other devices and control those.

- Not only understand about microcontroller, but we also understand and can interface with some other devices like ADC, LCD 16x2, Sensors, …. These devices are very common and important in IoT.

- Our temperature measuring product is low cost, all the materials we use to build this device are cheap, inexpensive things. So, if it is going to be manufactured, the cost wouldn't be high and if marketed, it could be a good choice and inexpensive option for customers.

- Although our product is cheap, it is quite accurate, user-friendly (not too hard to use) and has other function for users such as user can switch the temperature that measured between degrees Celsius, degrees Kelvin and degrees Fahrenheit which is most of temperature measuring products nowadays don't have.

6.2. Restriction

In addition to certain successes, our product still has some restrictions:

- Our product is still a model simulated on software and has not yet been realized in real life. Although we were successful when we simulated it but when it came to manufacture real life, we had to face many other external factors that make our product not as successful as simulation.

- Although our device is highly accurate but still has errors and most importantly our device still cannot measure negative or the high positive temperature. Our product can only measure temperature in the range of 0-150 degrees Celsius, that's make our product can't be used to measure the temperature that is too cold or too high. So, if users want to measure the temperature which is lower than 0 degrees or higher than 150 degrees Celsius, our device can't measure it.

6.3. Development Orientations

Some development orientations we can do after this project:

- Realize the product in real life, not a simulation.

- By upgrading the measured temperature in the product, the measured temperature will no longer be limited to the range of 0-150 but will be able to be higher, measuring even negative temperatures without significantly increasing the cost.

- Can be attached to some other devices and functions to improve the product such as alarm bells, ... If the machine feels the temperature is not safe for humans (too hot or too cold), the machine will emit an alarm and automatically balance the temperature to a safe level if possible.

REFERENCES

Chapter 1

- General information about temperature measurements https://lidinco.com/tim-hieu-chung-ve-may-nhiet

Chapter 2

- Research directions

https://businessinthenews.co.uk/2020/12/04/celsium-wearable-smart-thermometer-system-launches-promising-10x-more-accuracy-than-current-devices/

https://infraredcameras.com/news-center/how-does-an-infrared-thermometer-work

http://hethongcongnghiep.com/thiet-bi-do-nhiet-do-cam-tay-cellacast-pt-keller-viet-nam-499

https://www.htvsci.com/659-6-loai-nhiet-ke-tu-lanh-tu-mat.html

Chapter 3

- AT89C52 https://www.alldatasheet.com/datasheetpdf/pdf/56216/ATMEL/AT89C52.html

- LM35 https://pdf1.alldatasheet.com/datasheet-pdf/view/517588/TI1/LM35.html

- ADC0804 https://dientutuonglai.com/tim-hieu-adc0804.html

- LCD LM016L http://tailieudientu.lrc.tnu.edu.vn/chi-tiet/lcd-hd44780-giao-tiep-va-lap-trinh-dieu-khien-8096.html

- Pot https://hiokivn.com/tin-tuc/bien-tro-la-gi-cong-dung-phan-loai-cau-tao-va-cach-mac-bien-tro-2084.html

- DC Voltmeter https://thietbibenthanh.com/dong-ho-ho-volt-von-ke-la-gi-172-26.html

Chapter 4

- Proteus design suite https://en.wikipedia.org/wiki/Proteus_Design_Suite

- Keil https://en.wikipedia.org/wiki/Keil_(company)

- Analog to Digital Conversion https://learn.sparkfun.com/tutorials/analog-to-digital-conversion/all

- Communicating with buttons https://www.youtube.com/watch?v=v9Gf2lg8eqA&list=PLhFjtzzUovr-YW6vlzkiUJRo88T4deV23&index=17

- ADC0804 https://www.youtube.com/watch?v=DUHs6dCvIzo&list=PLhFjtzzUovr-YW6vlzkiUJRo88T4deV23&index=85

- LCD HD44780 (Part 1) https://www.youtube.com/watch?v=UCXdTd0ks6c&list=PLhFjtzzUovr-YW6vlzkiUJRo88T4deV23&index=44

- LCD HD44780 (Part 2) https://www.youtube.com/watch?v=8dPGVXvDHXQ&list=PLhFjtzzUovr-YW6vlzkiUJRo88T4deV23&index=45

- LCD HD44780 (Part 3) https://www.youtube.com/watch?v=Y9hC6rnBzUU&list=PLhFjtzzUovr-YW6vlzkiUJRo88T4deV23&index=46

- C & 8051 - Thermal sensor LM35 https://www.youtube.com/watch?v=taK6mJREVIY&list=PLhFjtzzUovr-YW6vlzkiUJRo88T4deV23&index=87