WIRELESS TEMPERATURE MONITORING AND CONTROL USING BLUETOOTH

A Project report submitted in partial fulfillment
of the requirements for the degree of B. Tech in Electrical Engineering

by

ANKIT GHOSH (11701615006)

PRINCE KUMAR (11701615030)

SUBHADEEP RAY (11701616005)

SUBHAJIT MONDAL (11701615056)

Under the supervision of

Mr. Budhaditya Biswas
Assistant Professor
Department of Electrical Engineering

श्रमम् बिना न किमपि साध्यम्

Department of Electrical Engineering
RCC INSTITUTE OF INFORMATION TECHNOLOGY
CANAL SOUTH ROAD, BELIAGHATA, KOLKATA – 700015, WEST BENGAL
Maulana Abul Kalam Azad University of Technology (MAKAUT)


Save

Department of Electrical Engineering
RCC INSTITUTE OF INFORMATION TECHNOLOGY
CANAL SOUTH ROAD, BELIAGHATA, KOLKATA – 700015, WEST BENGAL
PHONE: 033-2323-2463-154, FAX: 033-2323-4668
Email: hodeercciit@gmail.com, Website: http://www.rcciit.org/academic/ee.aspx

CERTIFICATE

To whom it may concern

This is to certify that the project work entitled **WIRELESS TEMPERATURE MONITORING AND CONTROL USING BLUETOOTH** is the bonafide work carried out by **ANKIT GHOSH (11701615006), PRINCE KUMAR (11701615030), SUBHADEEP RAY (11701616005), SUBHAJIT MONDAL (11701615056)**, the students of B.Tech in the Department of Electrical Engineering, RCC Institute of Information Technology (RCCIIT), Canal South Road, Beliaghata, Kolkata-700015, affiliated to Maulana Abul Kalam Azad University of Technology (MAKAUT), West Bengal, India, during the academic year 2018-19, in partial fulfillment of the requirements for the degree of Bachelor of Technology in Electrical Engineering and that this project has not submitted previously for the award of any other degree, diploma and fellowship.

(Budhaditya Biswas)
Assistant Professor
Department of Electrical Engineering
RCC Institute of Information Technology

Countersigned by

(Dr. Debasish Mondal)
HOD, Electrical Engineering Dept
RCC Institute of Information Technology

(External Examiner)

ACKNOWLEDGEMENT

It is our great fortune that we have got opportunity to carry out this project work under the supervision of **Mr. Budhaditya Biswas** in the Department of Electrical Engineering, RCC Institute of Information Technology (RCCIIT), Canal South Road, Beliaghata, Kolkata-700015, affiliated to Maulana Abul Kalam Azad University of Technology (**MAKAUT**), West Bengal, India. We express our sincere thanks and deepest sense of gratitude to our guide for his constant support, unparalleled guidance and limitless encouragement.

We would also like to convey our gratitude to all the faculty members and staffs of the Department of Electrical Engineering, RCCIIT for their whole hearted cooperation to make this work turn into reality.

We are very thankful to our Department and to the authority of RCCIIT for providing all kinds of infrastructural facility towards the research work.

Thanks to the fellow members of our group for working as a team.

ANKIT GHOSH (11701615006)

PRINCE KUMAR (11701615030)

SUBHADEEP RAY (11701616005)

SUBHAJIT MONDAL (11701615056)

To

The Head of the Department
Department of Electrical Engineering
RCC Institute of Information Technology
Canal South Rd. Beliagahata, Kolkata-700015

Respected Sir,

In accordance with the requirements of the degree of Bachelor of Technology in the Department of Electrical Engineering, RCC Institute of Information Technology, We present the following thesis entitled "**WIRELESS TEMPERATURE MONITORING AND CONTROL USING BLUETOOTH**". This work was performed under the valuable guidance of Mr. Budhadtiya Biswas, Assistant Professor in the Dept. of Electrical Engineering.

We declare that the thesis submitted is our own, expected as acknowledge in the test and reference and has not been previously submitted for a degree in any other Institution.

Yours Sincerely,

ANKIT GHOSH (11701615006)

PRINCE KUMAR (11701615030)

SUBHADEEP RAY (11701616005)

SUBHAJIT MONDAL (11701615056)

Contents

List of Figures

List of Tables

ABBREVIATIONS AND ACRONYMS

BT – Bluetooth
IC **-** Integrated Circuit
PCB – Printed Circuit Board
µC – Micro Controller
BJT **-** Bi-polar Junction Transistor
SPDT **-** Single Pole Double Throw
NO **-** Normally Open
NC **-** Normally Closed
COM – Common
LCD – Liquid Crystal Display
LED **-** Light Emitting Diode
POT – Potentiometer
SMPS – Switch Mode Power Supply
IR – Infrared
ISM – Industrial, scientific and medical
TDR **–** Time Delay Relay

ABSTRACT

In industrial and domestic applications temperature monitoring is important at each and every moment. Many faces a lot of problems due to lack of proper temperature monitoring system. In this project we use Temperature Sensor (DS18B20) which senses the temperature of a particular area; a piezo electric buzzer gets activated whenever a particular threshold temperature is sensed. Also a heater and a cooler are activated based on the temperature limit. The threshold value of temperature of different areas can be set accordingly so that we are alarmed based on that value. In such areas we can use this model so that it will help us in sensing the danger level of that area and help us in preventing any loss of life.

In this model, all the data of temperature are monitored each and every time in LCD and retrieved in mobile app through Bluetooth module. User can connect the smart phone wirelessly with the system and monitor the process variable. When the app is running on the smart phone, data sent by HC-05 Bluetooth module is received and displayed on the app.

CHAPTER 1

(Introduction)

1.1 INTRODUCTION

A temperature controller is a device that is used to control temperature. It does this by first measuring the temperature (**process variable**), it then compares it to the desired value (**set value**). The difference between these values is known as the error (Deviation). Temperature controllers use this error to decide how much heating or cooling is required to bring the process temperature back to the desired value. Once this calculation is complete the controller will produce an output signal that effects the change required. This output signal is known as the (**manipulated value)** and is normally connected to a heater, control valve, fan or some other "final control element" which actually injects or removes heat from the process.

Temperature measurement and control are vital in many industrial processes. Accurate control of the temperature is essential in nearly all chemical processes. In some applications, an accuracy of around $5\text{-}10^0$ may be acceptable. There are also some industrial applications which require better than 1^0 accuracy.
Temperature sensors come in many different forms and a number of techniques have evolved for the measurement of temperature. There are new forms of sensors which require no contact with the medium whose temperature is to be sensed. The majority of sensors still require to touch the solid, liquid, or the gas whose temperature is to be measured. Four technologies are currently in use: thermocouples (TCs), thermistors, resistance temperature detectors (RTDs), and IC sensors.

1.2 IC Temperature sensor

An IC Temperature Sensor is a two terminal integrated circuit temperature transducer that produces an output current proportional to absolute temperature. The sensor package is small with a low thermal mass and a fast response time. The most common temperature range is -55 to 150°C (-58 to 302°F). The solid state sensor output can be analog or digital.

Voltage Output IC Sensors

- Typically 10mV per degree C with nominal output correlated to 0K, 25°C.
- Some sensors have an offset at 0°C so that they can be used and read below 0°C without having to use a negative power supply.
- Non-linearity typically less than 1°C across their temperature range.

Current Output IC Sensors

- Nominal Output: 298 µA at 25°C
- 1 µA output per °C

Digital Output IC Sensors

- Have built in A-D Converters
- The number of digits in the A-D converter provides the resolution
- 10 Bit plus sign provides temperature resolved in increments of 0.25°C
- 12 Bit plus sign provides temperature resolved in increments of 0.0625°C

Where are IC Sensors used?

- On circuit boards to monitor and control temperature.
- In computers to control CPU temperature.
- In telecommunications applications (cell phones & PDA™).

- In some industrial immersion applications.

Strengths and Weaknesses of IC Sensors

Strengths:

- Analog or Digital outputs available
- Low cost
- Direct voltage, current or digital output needing no additional circuitry
- Linear output, no curve fitting
- Direct reading of temperature (1.000 = 100C and 298ÂµA = 298K or 25°C) on some analog devices
- Various communication interfaces

Weaknesses:

- Narrow temperature range: -55 to 150°C Max
- Wider interchangeability than most RTDs and thermistors
- Wide variation in accuracy between different models
- Small package sizes can be a barrier to low cost applications in some immersion designs

1.3 Comparison of IC temperature sensors with other types of temperature seonsors

The analog IC solid state sensors provide an output as a voltage or current that is proportional with temperature without additional circuitry. The digital IC sensors provide an output that has been processed thru an integral A-D converter and is ready for input into digital control and monitoring systems. The IC sensors do not require linearization or other circuitry. The cost of IC sensors are also very competitive with, in some cases less costly than, RTD and thermistor sensors

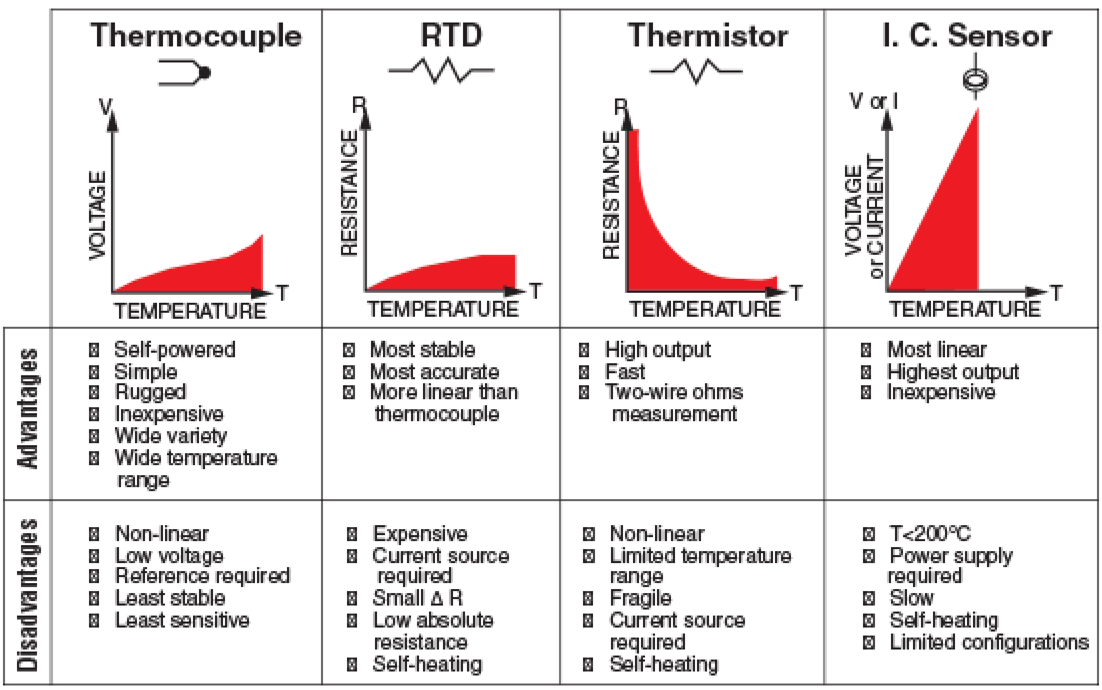


Figure 1: comparison of different temperature sensors

1.4 Wireless Communication - Bluetooth

Bluetooth wireless technology is a short range communications technology intended to replace the cables connecting portable unit and maintaining high levels of security. Bluetooth technology is based on **Ad-hoc technology** also known as **Ad-hoc Pico nets**, which is a local area network with a very limited coverage.

History of Bluetooth

WLAN technology enables device connectivity to infrastructure based services through a wireless carrier provider. The need for personal devices to communicate wirelessly with one another without an established infrastructure has led to the emergence of **Personal Area Networks (PANs)**.

- Ericsson's Bluetooth project in 1994 defines the standard for PANs to enable communication between mobile phones using low power and low cost radio interfaces.
- In May 1988, Companies such as IBM, Intel, Nokia and Toshiba joined Ericsson to form the Bluetooth Special Interest Group (SIG) whose aim was to develop a defacto standard for PANs.
- IEEE has approved a Bluetooth based standard named IEEE 802.15.1 for Wireless Personal Area Networks (WPANs). IEEE standard covers MAC and Physical layer applications.

Bluetooth specification details the entire protocol stack. Bluetooth employs Radio Frequency (RF) for communication. It makes use of **frequency modulation** to generate radio waves in the **ISM** band.

The usage of Bluetooth has widely increased for its special features.

- Bluetooth offers a uniform structure for a wide range of devices to connect and communicate with each other.
- Bluetooth technology has achieved global acceptance such that any Bluetooth enabled device, almost everywhere in the world, can be connected with Bluetooth enabled devices.
- Low power consumption of Bluetooth technology and an offered range of up to ten meters has paved the way for several usage models.
- Bluetooth offers interactive conference by establishing an adhoc network of laptops.
- Bluetooth usage model includes cordless computer, intercom, cordless phone and mobile phones.

Piconets and Scatternets

Bluetooth enabled electronic devices connect and communicate wirelessly through shortrange devices known as Piconets. Bluetooth devices exist in small ad-hoc configurations with the ability to act either as master or slave the specification allows a mechanism for master and slave to switch their roles. Point to point configuration with one master and one slave is the simplest configuration.

When more than two Bluetooth devices communicate with one another, this is called a **PICONET**. A Piconet can contain up to seven slaves clustered around a single master. The device that initializes establishment of the Piconet becomes the **master**.

The master is responsible for transmission control by dividing the network into a series of time slots amongst the network members, as a part of **time division multiplexing** scheme which is shown below.

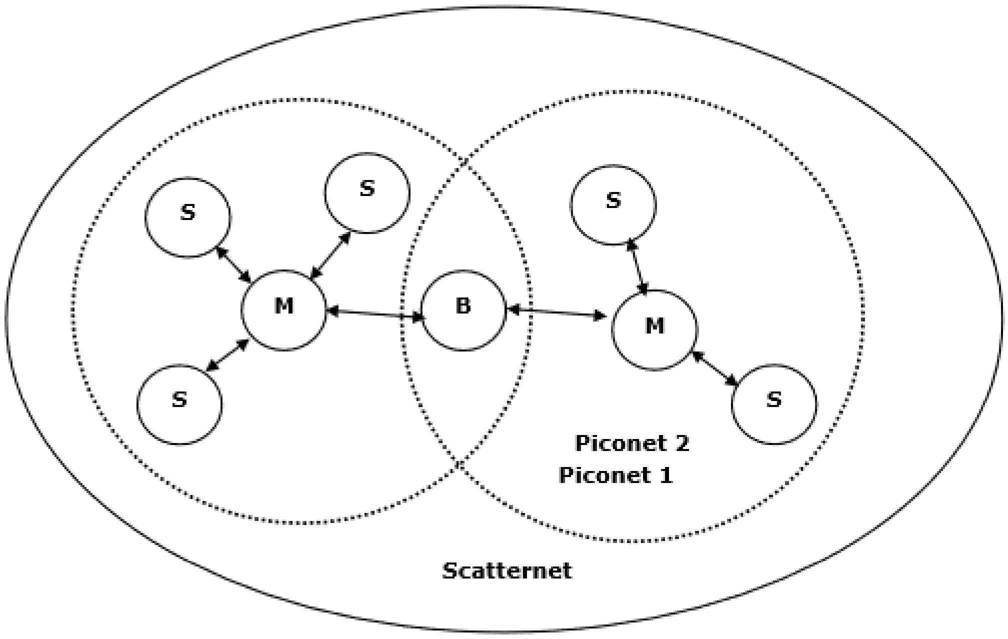


Figure 2: Piconets and Scatternets

The features of Piconets are as follows −

- Within a Piconet, the timing of various devices and the frequency hopping sequence of individual devices is determined by the clock and unique **48-bit address** of master.
- Each device can communicate simultaneously with up to seven other devices within a single Piconet.
- Each device can communicate with several piconets simultaneously.
- Piconets are established dynamically and automatically as Bluetooth enabled devices enter and leave piconets.
- There is no direct connection between the slaves and all the connections are essentially master-to-slave or slave-to-master.
- Slaves are allowed to transmit once these have been polled by the master.
- Transmission starts in the slave-to-master time slot immediately following a polling packet from the master.
- A device can be a member of two or more piconets, jumping from one piconet to another by adjusting the transmission regime-timing and frequency hopping sequence dictated by the master device of the second piconet.

- It can be a slave in one piconet and master in another. It however cannot be a master in more than once piconet.
- Devices resident in adjacent piconets provide a bridge to support inner-piconet connections, allowing assemblies of linked piconets to form a physically extensible communication infrastructure known as **Scatternet**.

Spectrum

Bluetooth technology operates in the unlicensed industrial, scientific and medical (ISM) band at 2.4 to 2.485 GHZ, using a spread spectrum hopping, full-duplex signal at a nominal rate of 1600 hops/sec. the 2.4 GHZ ISM band is available and unlicensed in most countries.

Range

Bluetooth operating range depends on the device Class 3 radios have a range of up to 1 meter or 3 feet Class 2 radios are most commonly found in mobile devices have a range of 10 meters or 30 feet Class 1 radios are used primarily in industrial use cases have a range of 100 meters or 300 feet.

Data rate

Bluetooth supports 1Mbps data rate for version 1.2 and 3Mbps data rate for Version 2.0 combined with Error Data Rate.

1.5 Overview and benefits of the project

Remote Control Technology's line of dependable, durable wireless remote switching systems can and will make money for you and your business. Wireless remote control benefits include:

No legal issues
Obtaining access to or traversing properties with hard lines is extremely difficult.

No copper wire to steal
As the price of copper increases, so does the possibility that your wire will be stolen. Using a wireless remote system means no wire for thieves to steal.

Extended range
Unlike much of the equipment on the market, Remote Control Technology's wireless remote equipment has long-range communication capabilities — up to 5 miles.

Eliminate the need for wire and conduit
Wire and conduit are expensive and high maintenance. Typical wear-and-tear, digging, rodent damage, theft, etc., are all examples of problems that can damage wire. RCT's wireless remote systems put an end to these drawbacks of wired technology.

Higher profits
Wireless remote switching systems eliminate the costly, labor-intensive process of trenching and laying wire. As a result, the contractor can enjoy an increased profitability of 200 percent or more in this facet of the job.

No FCC licensing required
RCT equipment does not require FCC licensing, whereas much of the other equipment on the market does. This is a significant benefit, as the FCC licensing process alone may take up to 8 weeks.

Less maintenance and servicing
In many states a contractor is obligated by law to maintain pumping systems for up to a year after its installation. RCT switching systems eliminate a majority of these maintenance and servicing issues by automating the job. Fewer service calls mean higher profits.

Reliability and compatibility
All of the components that a contractor puts into a project must interface with one another and have the utmost reliability. RCT wireless remote equipment has proven to be highly compatible with standard equipment used in most industries, as well as offering unparalleled reliability in use with programmable logic controllers (PLCs), various switches and relays, etc.

1.6 Organisation of thesis

The thesis is organised into five chapters including the chapter of introduction. Each chapter is different from the other and is described along with the necessary theory required to comprehend it.

Chapter 2 deals with the literature reviews. From this chapter we can see before our project who else works on this topic and how our project is different and advance from those projects.

Chapter 3 deals with the theory required to do the project. The basic of operation of DS18B20 temperature sensor and how to interface with AT89c51 microcontroller are described there.

Chapter 4 deals with the hardware modelling of the projects. The main features, photographs, step by step operation of the prototype, component listing and the hardware interfacing of the required components are described here.

Chapter 5 describes the operation of the prototype circuit. A flow chart is presented on the actions which describes the principle of DS18B20 temperature sensor detection. Once the temperature is measured by the sensor the controller display it over a 16X2 LCD screen and send it to a remote device through bluetooth.

Chapter 6 concludes the work performed so far. The possible limitations in proceeding research towards this work are discussed. The future work that can be done in improving the current scenario is mentioned. The future potential along the lines of this work is also discussed.

Chapter 7 References are listed in this chapter

Appendix A, B & C Hardware description, software coding and datasheets are listed here.

CHAPTER 2

(Literature Review)

Rashmi Singh and S.P Singh, ***"Development of a Low Cost Wireless Temperature Monitoring System for Industrial & Research Application",*** School of Energy and Environmental Studies, Devi Ahilya University, Khandwa Road, Indore 452001, India , 07 Feb 2015

The proposed paper focus on developing a low cost and easy handling temperature monitoring system, based on radio frequency for wireless data transmission. The wireless system module architecture consists of a power supply, a sensor and a main node system mainly based on wireless radio frequency (RF) technology

Pradeep Kasale, Shekhar Kedar, Mrinal Kishore, Prof. Kanchan Maske, ***"Android based Greenhouse Monitoring and Controlling System",*** Electronics and Telecommunication & Pune University, India, March 2015, pg.68 – 71

In agricultural country like India, greenhouses form an important aspect of agricultural and horticulture sectors. In greenhouses, plants are grown under favourable climatic conditions for its production and growth. Thus monitoring and control of greenhouse environment is necessary for production and management of greenhouses. This project is designed to monitor and control the indoor humidity and weather conditions affecting the plants using embedded system and Android mobile phone. The android phone is connected to a central server which then connects to microcontroller and humidity sensor via serial communication**.**

Purnima, S.R.N. Reddy, ***"Design of Remote Monitoring and Control System with Automatic Irrigation System using GSM-Bluetooth"***, International Journal of Computer Applications (0975 – 888), Volume 47– No.12, June 2012

In past few years, automatic irrigation system has seen a rapid growth in terms of technology. At present cost-saving technology, labor-saving are the addressing key issues in irrigation. This paper gives a review of these systems based on existing technologies and also proposes an economical and generic automatic irrigation system based on wireless sensors with GSM-Bluetooth for irrigation system controller and remote monitoring system

Mohammed Juned, Srija Unnikrishnan***, "Bluetooth Based Remote Monitoring & Control System",*** Journal of Basic and Applied Engineering Research, Volume 1, Number 8; October, 2014 pp. 108-111

The Bluetooth is an open wireless technology which revolutionized the connectivity by providing freedom from wired connections. In this paper we mainly focus on Implementing a Bluetooth Based Remote Monitoring & Control System which is designed using the Microcontroller (ATmega16) as an embedded target and Bluetooth device which is connected to the controller along with different sensors to measure different real time parameters such as temperature, pressure & humidity, it also controls the temperature of a process**.**

R Vijayarani, S. Praveen Kumar, ***"Environmental Monitoring and Controlling Various Parameters in a Closed Loop",*** International Journal of Engineering Research and General Science Volume 2, Issue 3, April-May 2014

This paper focuses on environmental monitoring and controlling various parameters in a closed loop.

K. Jaganmohan Reddy, B.Santhosh Kumar, ***"Patient Temperature Monitoring System Using Bluetooth Communication" ,*** International Journal of Engineering Trends and Technology (IJETT) – Volume 16 Number 4 – Oct 2014

This paper demonstrates portable wireless biomedical temperature monitoring system where the temperature of the body of the patient can be measured and transferred through wireless communication.

CHAPTER 3

(Theory)

3.1 DS18B20 Programmable Resolution 1-Wire Digital Thermometer

The DS18B20 digital thermometer provides 9-bit to 12-bit Celsius temperature measurements and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18B20 communicates over a 1-Wire bus that by definition requires only one data line (and ground) for communication with a central microprocessor. It has an operating temperature range of -55°C to +125°C and is accurate to ±0.5°C over the range of -10°C to +85°C. In addition, the DS18B20 can derive power directly from the data line ("parasite power"), eliminating the need for an external power supply.

Each DS18B20 has a unique 64-bit serial code, which allows multiple DS18B20s to function on the same 1-Wire bus. Thus, it is simple to use one microprocessor to control many DS18B20s distributed over a large area. Applications that can benefit from this feature include HVAC environmental controls, temperature monitoring systems inside buildings, equipment, or machinery, and process monitoring and control systems.

3.2 FEATURES OF DS18B20 TEMPERATURE SENSOR

- Unique 1-Wire® Interface Requires Only One Port Pin for Communication
- Each Device has a Unique 64-Bit Serial Code Stored in an On-Board ROM
- Multidrop Capability Simplifies Distributed Temperature-Sensing Applications
- Requires No External Components
- Can Be Powered from Data Line; Power Supply Range is 3.0V to 5.5V
- Measures Temperatures from -55°C to +125°C (-67°F to +257°F)
- ±0.5°C Accuracy from -10°C to +85°C
- Thermometer Resolution is User Selectable from 9 to 12 Bits
- Converts Temperature to 12-Bit Digital Word in 750ms (Max)
- User-Definable Nonvolatile (NV) Alarm Settings
- Alarm Search Command Identifies and Addresses Devices Whose Temperature is Outside Programmed Limits (Temperature Alarm Condition)
- Available in 8-Pin SO (150 mils), 8-Pin µSOP, and 3-Pin TO-92 Packages
- Software Compatible with the DS1822
- Applications Include Thermostatic Controls, Industrial Systems, Consumer Products, Thermometers, or Any Thermally Sensitive System

3.3 PIN CONFIGURATIONS

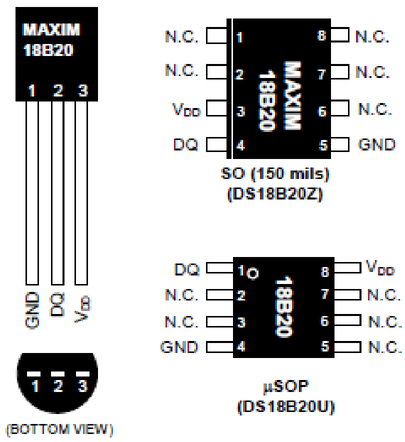


Figure 3: Pin description of DS18B20

3.4 OVERVIEW OF DS18B20

Figure 4 shows a block diagram of the DS18B20, and pin descriptions are given in the *Pin Description* table. The 64-bit ROM stores the device's unique serial code. The scratchpad memory contains the 2-byte temperature register that stores the digital output from the temperature sensor. In addition, the scratchpad provides access to the 1-byte upper and lower alarm trigger registers (TH and TL) and the 1-byte configuration register. The configuration register allows the user to set the resolution of the temperature-to-digital conversion to 9, 10, 11, or 12 bits. The TH, TL, and configuration registers are nonvolatile (EEPROM), so they will retain data when the device is powered down.

The DS18B20 uses Maxim's exclusive 1-Wire bus protocol that implements bus communication using one control signal. The control line requires a weak pullup resistor since all devices are linked to the bus via a 3-state or open-drain port (the DQ pin in the case of the DS18B20). In this bus system, the microprocessor (the master device) identifies and addresses devices on the bus using each device's unique 64-bit code. Because each device has a unique code, the number of devices that can be addressed on one bus is virtually unlimited. The 1-Wire bus protocol, including detailed explanations of the commands and "time slots," is covered in the *1-Wire Bus System* section.

Another feature of the DS18B20 is the ability to operate without an external power supply. Power is instead supplied through the 1-Wire pullup resistor via the DQ pin when the bus is high. The high bus signal also charges an internal capacitor (CPP), which then supplies power to the device when the bus is low. This method of deriving power from the 1-Wire bus is referred to as "parasite power." As an alternative, the DS18B20 may also be powered by an external supply on VDD.

DS18B20 Block Diagram

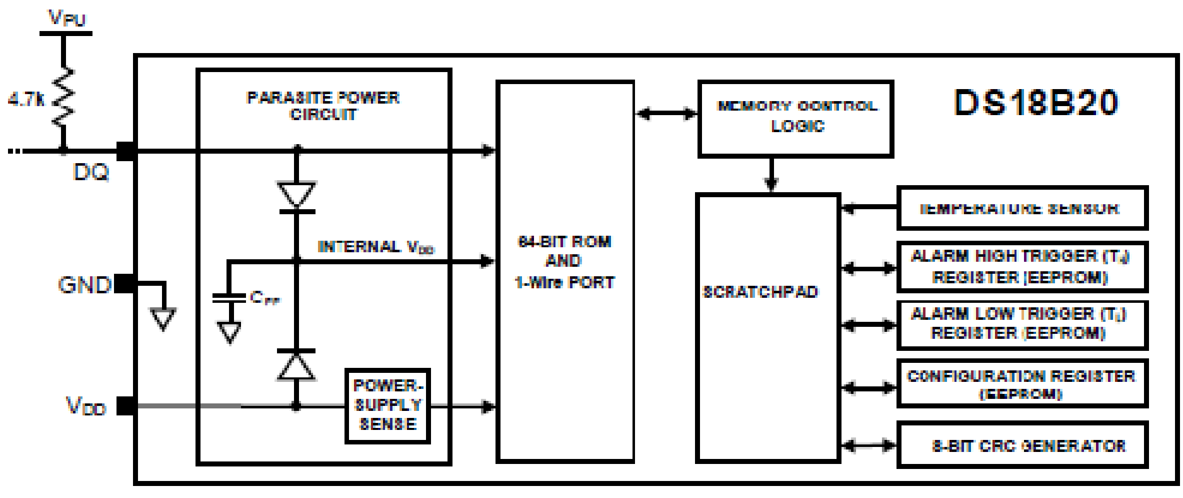


Figure 4: DS18B20 Block Diagram

3.5 OPERATION—MEASURING TEMPERATURE

The core functionality of the DS18B20 is its direct-to-digital temperature sensor. The resolution of the temperature sensor is user-configurable to 9, 10, 11, or 12 bits, corresponding to increments of 0.5°C, 0.25°C, 0.125°C, and 0.0625°C, respectively. The default resolution at power-up is 12-bit. The DS18B20 powers up in a low-power idle state. To initiate a temperature measurement and A-to-D conversion, the master must issue a Convert T [44h] command. Following the conversion, the resulting thermal data is stored in the 2-byte temperature register in the scratchpad memory and the DS18B20 returns to its idle state. If the DS18B20 is powered by an external supply, the master can issue "read time slots" after the Convert T command and the DS18B20 will respond by transmitting 0 while the temperature conversion is in progress and 1 when the conversion is done. If the DS18B20 is powered with parasite power, this notification technique cannot be used since the bus must be pulled high by a strong pullup during the entire temperature conversion. The bus requirements for parasite power are explained in detail in the *Powering the DS18B20* section.

The DS18B20 output temperature data is calibrated in degrees Celsius; for Fahrenheit applications, a lookup table or conversion routine must be used. The temperature data is stored as a 16-bit sign-extended two's complement number in the temperature register (see Figure 2). The sign bits (S) indicate if the temperature is positive or negative: for positive numbers S = 0 and for negative numbers S = 1. If the DS18B20 is configured for 12-bit resolution, all bits in the temperature register will contain valid data. For 11-bit resolution, bit 0 is undefined. For 10-bit resolution, bits 1 and 0 are undefined, and for 9-bit resolution bits 2, 1, and 0 are undefined. Table 1 gives examples of digital output data and the corresponding temperature reading for 12-bit resolution conversions.

	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
LS BYTE	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
	BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
MS BYTE	S	S	S	S	S	2^6	2^5	2^4

S = SIGN

Figure 5: Temperature Register Format

Table 1: Temperature/Data Relationship

TEMPERATURE (°C)	DIGITAL OUTPUT (BINARY)	DIGITAL OUTPUT (HEX)
+125	0000 0111 1101 0000	07D0h
+85*	0000 0101 0101 0000	0550h
+25.0625	0000 0001 1001 0001	0191h
+10.125	0000 0000 1010 0010	00A2h
+0.5	0000 0000 0000 1000	0008h
0	0000 0000 0000 0000	0000h
-0.5	1111 1111 1111 1000	FFF8h
-10.125	1111 1111 0101 1110	FF5Eh
-25.0625	1111 1110 0110 1111	FE6Fh
-55	1111 1100 1001 0000	FC90h

*The power-on reset value of the temperature register is +85°C.

3.6 ROM COMMANDS

After the bus master has detected a presence pulse, it can issue a ROM command. These commands operate on the unique 64-bit ROM codes of each slave device and allow the master to single out a specific device if many are present on the 1-Wire bus. These commands also allow the master to determine how many and what types of devices are present on the bus or if any device has experienced an alarm condition. There are five ROM commands, and each command is 8 bits long. The master device must issue an appropriate ROM command before issuing a DS18B20 function command.

SEARCH ROM [F0h]

When a system is initially powered up, the master must identify the ROM codes of all slave devices on the bus, which allows the master to determine the number of slaves and their device types. The master learns the ROM codes through a process of elimination that requires the master to perform a Search ROM cycle (i.e., Search ROM command followed by data exchange) as many times as necessary to identify all of the slave devices. If there is only one slave on the bus, the simpler Read ROM command (see below) can be used in place of the Search ROM process. For a detailed explanation of the Search ROM procedure, refer to the *iButton® Book of Standards* at www.maxim-ic.com/ibuttonbook. After every Search ROM cycle, the bus master must return to Step 1 (Initialization) in the transaction sequence.

READ ROM [33h]

This command can only be used when there is one slave on the bus. It allows the bus master to read the slave's 64-bit ROM code without using the Search ROM procedure. If this command is used when there is more than one slave present on the bus, a data collision will occur when all the slaves attempt to respond at the same time.

MATCH ROM [55h]

The match ROM command followed by a 64-bit ROM code sequence allows the bus master to address a specific slave device on a multidrop or single-drop bus. Only the slave that exactly matches the 64-bit ROM code sequence will respond to the function command issued by the master; all other slaves on the bus will wait for a reset pulse.

SKIP ROM [CCh]

The master can use this command to address all devices on the bus simultaneously without sending out any ROM code information. For example, the master can make all DS18B20s on the bus perform simultaneous temperature conversions by issuing a Skip ROM command followed by a Convert T [44h] command.

Note that the Read Scratchpad [BEh] command can follow the Skip ROM command only if there is a single slave device on the bus. In this case, time is saved by allowing the master to read from the slave without sending the device's 64-bit ROM code. A Skip ROM command followed by a Read Scratchpad command will cause a data collision on the bus if there is more than one slave since multiple devices will attempt to transmit data simultaneously.

ALARM SEARCH [ECh]

The operation of this command is identical to the operation of the Search ROM command except that only slaves with a set alarm flag will respond. This command allows the master device to determine if any DS18B20s experienced an alarm condition during the most recent temperature conversion. After every Alarm Search cycle (i.e., Alarm Search command followed by data exchange), the bus master must return to Step 1 (Initialization) in the transaction sequence. See the *Operation—Alarm Signaling* section for an explanation of alarm flag operation.

3.7 DS18B20 FUNCTION COMMANDS

After the bus master has used a ROM command to address the DS18B20 with which it wishes to communicate, the master can issue one of the DS18B20 function commands. These commands allow the master to write to and read from the DS18B20's scratchpad memory, initiate temperature conversions and determine the power supply mode. The DS18B20 function commands, which are described below, are summarized in Table 3 and illustrated by the flowchart in Figure 12.

CONVERT T [44h]

This command initiates a single temperature conversion. Following the conversion, the resulting thermal data is stored in the 2-byte temperature register in the scratchpad memory and the DS18B20 returns to its low-power idle state. If the device is being used in parasite power mode, within 10µs (max) after this command is issued the master must enable a strong pullup on the 1-Wire bus for the duration of the conversion (tCONV) as described in the *Powering the DS18B20* section. If the DS18B20 is powered by an external supply, the master can issue read time slots after the Convert T command and the DS18B20 will respond by transmitting a 0 while the temperature conversion is in progress and a 1 when the conversion is done. In parasite power mode this notification technique cannot be used since the bus is pulled high by the strong pullup during the conversion.

WRITE SCRATCHPAD [4Eh]

This command allows the master to write 3 bytes of data to the DS18B20's scratchpad. The first data byte is written into the TH register (byte 2 of the scratchpad), the second byte is written into the TL register (byte 3), and the third byte is written into the configuration register (byte 4). Data must be transmitted least significant bit first. All three bytes MUST be written before the master issues a reset, or the data may be corrupted.

READ SCRATCHPAD [BEh]

This command allows the master to read the contents of the scratchpad. The data transfer starts with the least significant bit of byte 0 and continues through the scratchpad until the 9th byte (byte 8 – CRC) is read. The master may issue a reset to terminate reading at any time if only part of the scratchpad data is needed.

COPY SCRATCHPAD [48h]

This command copies the contents of the scratchpad TH, TL and configuration registers (bytes 2, 3 and 4) to EEPROM. If the device is being used in parasite power mode, within 10µs (max) after this command is issued the master must enable a strong pullup on the 1-Wire bus for at least 10ms as described in the *Powering the DS18B20* section.

RECALL E2 [B8h]

This command recalls the alarm trigger values (TH and TL) and configuration data from EEPROM and places the data in bytes 2, 3, and 4, respectively, in the scratchpad memory. The master device can issue read time slots following the Recall E2 command and the DS18B20 will indicate the status of the recall by transmitting 0 while the recall is in progress and 1 when the recall is done. The recall operation happens automatically at power-up, so valid data is available in the scratchpad as soon as power is applied to the device.

READ POWER SUPPLY [B4h]

The master device issues this command followed by a read time slot to determine if any DS18B20s on the bus are using parasite power. During the read time slot, parasite powered DS18B20s will pull the bus low, and externally powered DS18B20s will let the bus remain high. See the *Powering the DS18B20* section for usage information for this command.

Table 2: DS18B20 Function Command Set

COMMAND	DESCRIPTION	PROTOCOL	1-Wire BUS ACTIVITYAFTER COMMAND IS ISSUED	NOTES
TEMPERATURE CONVERSION COMMANDS				
Convert T	Initiates temperature conversion.	44h	DS18B20 transmits conversion status to master (not applicable for parasite-powered DS18B20s).	1
MEMORY COMMANDS				
Read Scratchpad	Reads the entire scratchpad including the CRC byte.	BEh	DS18B20 transmits up to 9 data bytes to master.	2
Write Scratchpad	Writes data into scratchpad bytes 2, 3, and 4 (T_H, T_L, and configuration registers).	4Eh	Master transmits 3 data bytes to DS18B20.	3
Copy Scratchpad	Copies T_H, T_L, and configuration register data from the scratchpad to EEPROM.	48h	None	1
Recall E^2	Recalls T_H, T_L, and configuration register data from EEPROM to the scratchpad.	B8h	DS18B20 transmits recall status to master.	
Read Power Supply	Signals DS18B20 power supply mode to the master.	B4h	DS18B20 transmits supply status to master.	

Note 1: For parasite-powered DS18B20s, the master must enable a strong pullup on the 1-Wire bus during temperature conversions and copies from the scratchpad to EEPROM. No other bus activity may take place during this time.
Note 2: The master can interrupt the transmission of data at any time by issuing a reset.
Note 3: All three bytes must be written before a reset is issued.

3.8 INTERFACING THE TEMPERATURE SENSOR WITH THE MICROCONTROLLER

The one wire temperature sensor DS18B20 can be interfaced with most of the microcontroller. In this project it is connected with AT89c51 microcontroller. A 4.7 KΩ pullup resistor is required for the proper connection. The beauty of this temperature sensor is that it occupies only one digital IO pin of the microcontroller. It can be possible to connect infinite number of sensor in the same pin because each sensor has a unique 64 bit address. In figure 6 it is shown that a temperature sensor is connected to 8051 microcontroller.

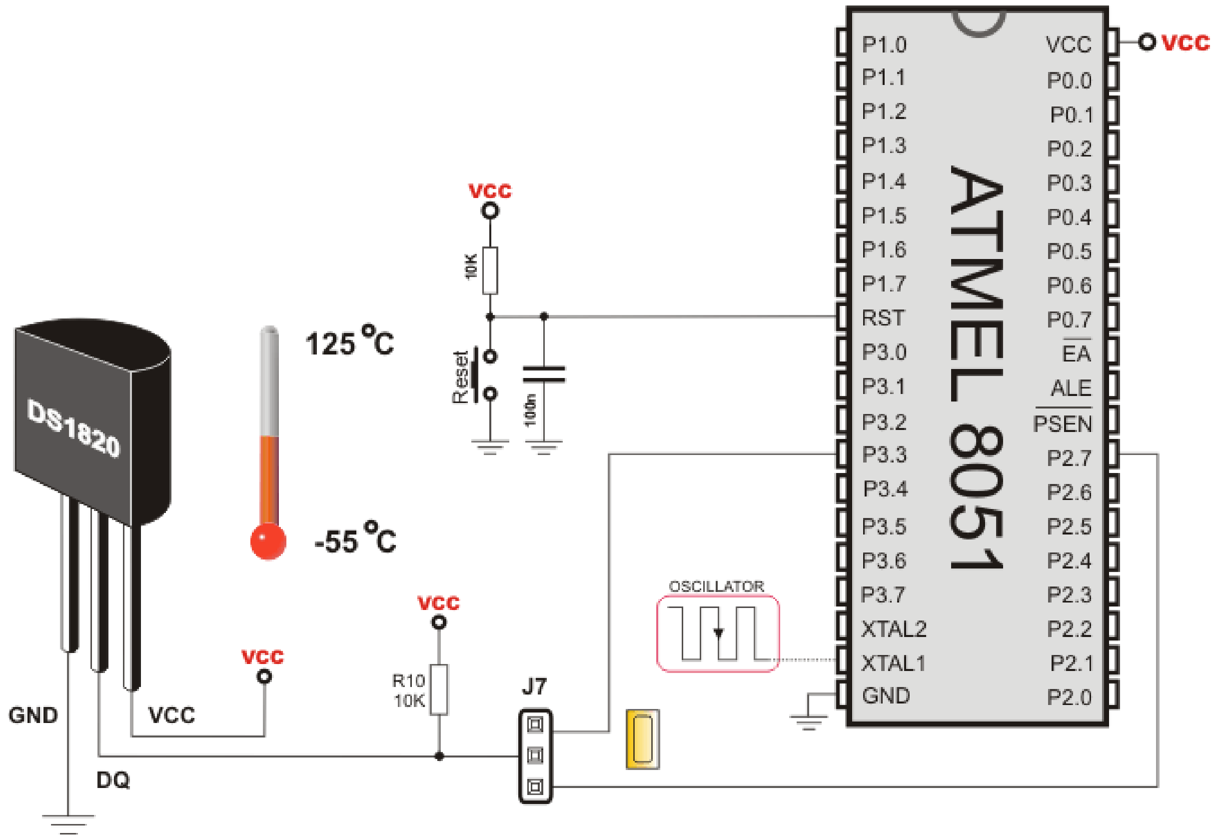


Figure 6: Interfacing DS18B20 with 8051

3.9 Basic of Serial Communication

The asynchronous serial protocol has a number of built-in rules - mechanisms that help ensure robust and error-free data transfers. These mechanisms, which we get for eschewing the external clock signal, are:

- Data bits,
- Synchronization bits,
- Parity bits,
- and Baud rate.

Through the variety of these signaling mechanisms, you'll find that there's no one way to send data serially. The protocol is highly configurable. The critical part is making sure that **both devices on a serial bus are configured to use the exact same protocols**.

3.9.1 Baud Rate

The baud rate specifies **how fast** data is sent over a serial line. It's usually expressed in units of bits-per-second (bps). If you invert the baud rate, you can find out just how long it takes to transmit a single bit. This value determines how long the transmitter holds a serial line high/low or at what period the receiving device samples its line.

Baud rates can be just about any value within reason. The only requirement is that both devices operate at the same rate. One of the more common baud rates, especially for simple stuff where speed isn't critical, is **9600 bps**. Other "standard" baud are 1200, 2400, 4800, 19200, 38400, 57600, and 115200.

The higher a baud rate goes, the faster data is sent/received, but there are limits to how fast data can be transferred. You usually won't see speeds exceeding 115200 - that's fast for most microcontrollers. Get too high, and you'll begin to see errors on the receiving end, as clocks and sampling periods just can't keep up.

3.9.2 Framing the data

Each block (usually a byte) of data transmitted is actually sent in a *packet* or *frame* of bits. Frames are created by appending synchronization and parity bits to our data.

Figure 7: A serial frame.

Some symbols in the frame have configurable bit sizes. Let's get into the details of each of these frame pieces.

3.9.3 Data chunk

The real meat of every serial packet is the data it carries. We ambiguously call this block of data a *chunk*, because its size isn't specifically stated. The amount of data in each packet can be set to anything from 5 to 9 bits. Certainly, the standard data size is your basic 8-bit byte, but other sizes have their uses. A 7-bit data chunk can be more efficient than 8, especially if you're just transferring 7-bit ASCII characters.

After agreeing on a character-length, both serial devices also have to agree on the **endianness** of their data. Is data sent most-significant bit (msb) to least, or vice-versa? If it's not otherwise stated, you can usually assume that data is transferred **least-significant bit (lsb) first**.

3.9.4 Synchronization bits

The synchronization bits are two or three special bits transferred with each chunk of data. They are the **start bit** and the **stop bit(s)**. True to their name, these bits mark the beginning and end of a packet. There's always only one start bit, but the number of stop bits is configurable to either one or two (though it's commonly left at one).

The start bit is always indicated by an idle data line going from 1 to 0, while the stop bit(s) will transition back to the idle state by holding the line at 1.

3.9.5 Parity bits

Parity is a form of very simple, low-level error checking. It comes in two flavors: odd or even. To produce the parity bit, all 5-9 bits of the data byte are added up, and the evenness of the sum decides whether the bit is set or not. For example, assuming parity is set to even and was being added to a data byte like `0b01011101`, which has an odd number of `1`'s (5), the parity bit would be set to `1`. Conversely, if the parity mode was set to odd, the parity bit would be `0`.

Parity is *optional*, and not very widely used. It can be helpful for transmitting across noisy mediums, but it'll also slow down your data transfer a bit and requires both sender and receiver to implement error-handling (usually, received data that fails must be re-sent).

3.9.6 9600 baud rate communication using AT89c51

9600 8N1 - 9600 baud, 8 data bits, no parity, and 1 stop bit - is one of the more commonly used serial protocols. So, what would a packet or two of 9600 8N1 data look like? Let's have an example

A device transmitting the ASCII characters 'O' and 'K' would have to create two packets of data. The ASCII value of *O* (that's uppercase) is 79, which breaks down into an 8-bit binary value of `01001111`, while *K*'s binary value is `01001011`. All that's left is appending sync bits.

It isn't specifically stated, but it's assumed that data is transferred least-significant bit first. Notice how each of the two bytes is sent as it reads from right-to-left.

Figure 8: 9600 baud rate bit pattern

Since we're transferring at 9600 bps, the time spent holding each of those bits high or low is 1/(9600 bps) or 104 µs per bit.

For every byte of data transmitted, there are actually 10 bits being sent: a start bit, 8 data bits, and a stop bit. So, at 9600 bps, we're actually sending 9600 bits per second or 960 (9600/10) bytes per second.

Now that you know how to construct serial packets, we can move on to the hardware section. There we'll see how those 1's and 0's and the baud rate are implemented at a signal level.

3.10 Serial Communication using HC05

Bluetooth is a **standardized protocol** for sending and receiving data via a 2.4GHz wireless link. It's a secure protocol, and it's perfect for short-range, low-power, low-cost, wireless transmissions between electronic devices.

Bluetooth®

These days it feels like *everything* is wireless, and Bluetooth is a big part of that wireless revolution. You'll find Bluetooth embedded into a great variety of consumer products, like headsets, video game controllers, or (of course) livestock trackers.

In our world of embedded electronics, Bluetooth serves as an excellent protocol for wirelessly transmitting relatively small amounts of data over a short range (<100m). It's perfectly suited as a wireless replacement for serial communication interfaces. Or you can use it to create a DIY HID Computer Keyboard. Or, with the right module, it can be used to build a homebrew, wireless MP3-playing speaker.

This tutorial aims to provide a quick overview of the Bluetooth protocol. We'll examine the specifications and profiles that form its foundation, and we'll go over how Bluetooth compares to other wireless protocols.

3.10.1 How Bluetooth Works

The Bluetooth protocol operates at 2.4GHz in the same unlicensed ISM frequency band where RF protocols like ZigBee and WiFi also exist. There is a standardized set of rules and specifications that differentiates it from other protocols. If you have a few hours to kill and want to learn every nook and cranny of Bluetooth, check out the published specifications, otherwise here's a quick overview of what makes Bluetooth special.

3.10.2 Masters, Slaves, and Piconets

Bluetooth networks (commonly referred to as **piconets**) use a master/slave model to control when and where devices can send data. In this model, a single master device can be connected to up to seven different slave devices. Any slave device in the piconet can only be connected to a single master.

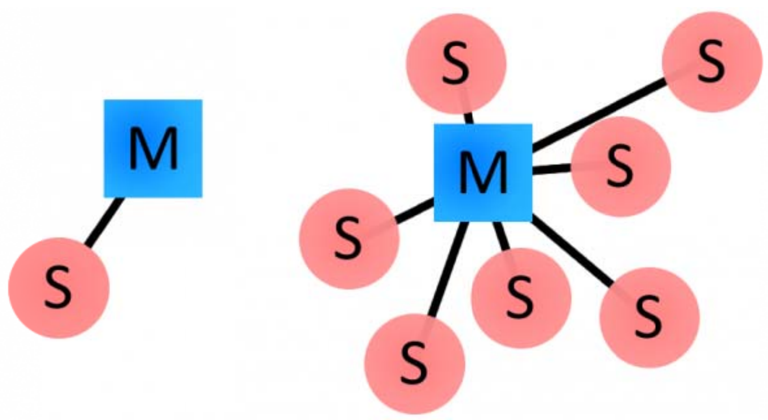


Figure 9: Examples of Bluetooth master/slave piconet topologies.

The master coordinates communication throughout the piconet. It can send data to any of its slaves and request data from them as well. Slaves are only allowed to transmit to and receive from their master. They can't talk to other slaves in the piconet.

3.10.3 Wireless Comparison

Bluetooth is far from the only wireless protocol out there. You might be reading this tutorial over a WiFi network. Or maybe you've even played with ZigBees or XBees. So what makes Bluetooth different from the rest of the wireless data transmission protocols out there?

Let's compare and contrast. We'll include BLE as a separate entity from *Classic* Bluetooth.

Table 3: wireless protocol comparison

Name	Bluetooth Classic	Bluetooth 4.0 Low Energy (BLE)	ZigBee	WiFi
IEEE Standard	802.15.1	802.15.1	802.15.4	802.11 (a, b, g, n)
Frequency (GHz)	2.4	2.4	0.868, 0.915, 2.4	2.4 and 5
Maximum raw bit rate (Mbps)	1-3	1	0.250	11 (b), 54 (g), 600 (n)
Typical data throughput (Mbps)	0.7-2.1	0.27	0.2	7 (b), 25 (g), 150 (n)
Maximum (Outdoor) Range (Meters)	10 (class 2), 100 (class 1)	50	10-100	100-250
Relative Power Consumption	Medium	Very low	Very low	High
Example Battery Life	Days	Months to years	Months to years	Hours
Network Size	7	Undefined	64,000+	255

Bluetooth isn't the best choice for every wireless job out there, but it does excel at short-range **cable-replacement**-type applications. It also boasts a typically more convenient connection process than its competitors (ZigBee specifically).

ZigBee is often a good choice for monitoring networks – like home automation projects. These networks might have dozens of wireless nodes, which are only sparsely active and never have to send a lot of data.

BLE combines the convenience of classic Bluetooth, and adds significantly lower power consumption. In this way it can compete with Zigbee for battery life. BLE can't compete with ZigBee in terms of network size, but for single device-to-device connectivity it's very comparable.

3.10.4 Asynchronous Serial communication using HC 05

Over the years, dozens of serial protocols have been crafted to meet particular needs of embedded systems. USB (universal *serial* bus), and Ethernet, are a couple of the more well-known computing serial interfaces. Other very common serial interfaces include SPI, I^2C, and the serial standard we're here to talk about today. Each of these serial interfaces can be sorted into one of two groups: synchronous or asynchronous.

A synchronous serial interface always pairs its data line(s) with a clock signal, so all devices on a synchronous serial bus share a common clock. This makes for a more straightforward, often faster serial transfer, but it also requires at least one extra wire between communicating devices. Examples of synchronous interfaces include SPI, and I^2C.

Asynchronous means that data is transferred **without support from an external clock signal**. This transmission method is perfect for minimizing the required wires and I/O pins, but it does mean we need to put some extra effort into reliably transferring and receiving data. The serial protocol we'll be discussing in this tutorial is the most common form of asynchronous transfers. It is so common, in fact, that when most folks say "serial" they're talking about this protocol (something you'll probably notice throughout this tutorial).

The clock-less serial protocol we'll be discussing in this tutorial is widely used in embedded electronics. If you're looking to add a GPS module, Bluetooth, XBee's, serial LCDs, or many other external devices to your project, you'll probably need to whip out some serial-fu.

CHAPTER 4

(Hardware Modeling)

4.1 Main features of the prototype

The features of the developed prototype are:

- LCD display (showing the real time temperature in degree celcious)
- 2 independent load can be controlled (ON/OFF control)
- 5 Volt operation (both control board and ON & OFF relay)
- User can set the temperature limit remotely.
- Remote data sending capabilities using Bluetooth
- Can be control from any Bluetooth host device (mobile, laptop etc)
- Local and remote temperature display capabilities.
- Sound an alarm if the temperature crosses the upper/lower limit.
- Cost effective (Rs 800/- approx)

4.2 Overview of the Project

The working process of the project is described in figure 10. At first the user need to connect the Bluetooth module (HC 05) with the host device (may be a mobile phone or a laptop). After connecting the module the controller will ask for setting the temperature limits. The user can skip the limit set if want. The controller will ask for enabling the alarm. The user also skip this step is want. Then the temperature in degree celcious will be display in the lcd screen (local display) as well as the remote host device (mobile screen) using Bluetooth technology. If the temperature crosses the limits then the required device (heater / cooler) will be switched ON and control the temperature automatically)

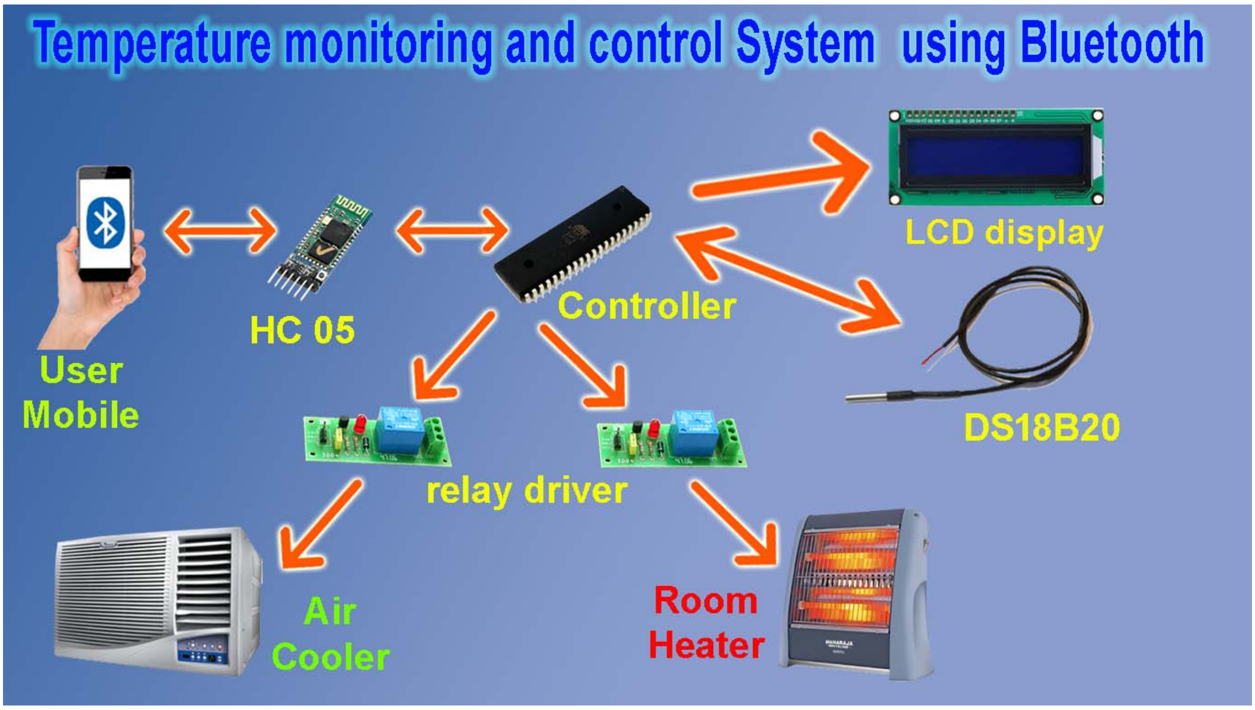
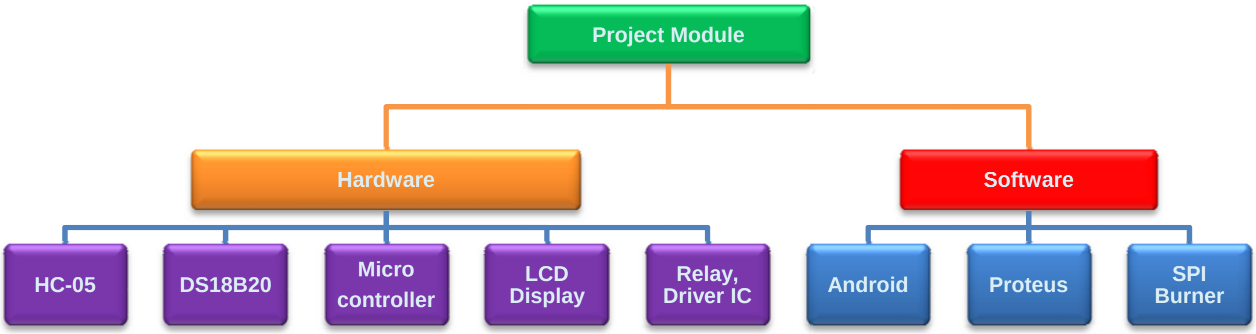


Figure 10: Working of the prototype

4.3 Project layout



4.4 Photographs of the prototype

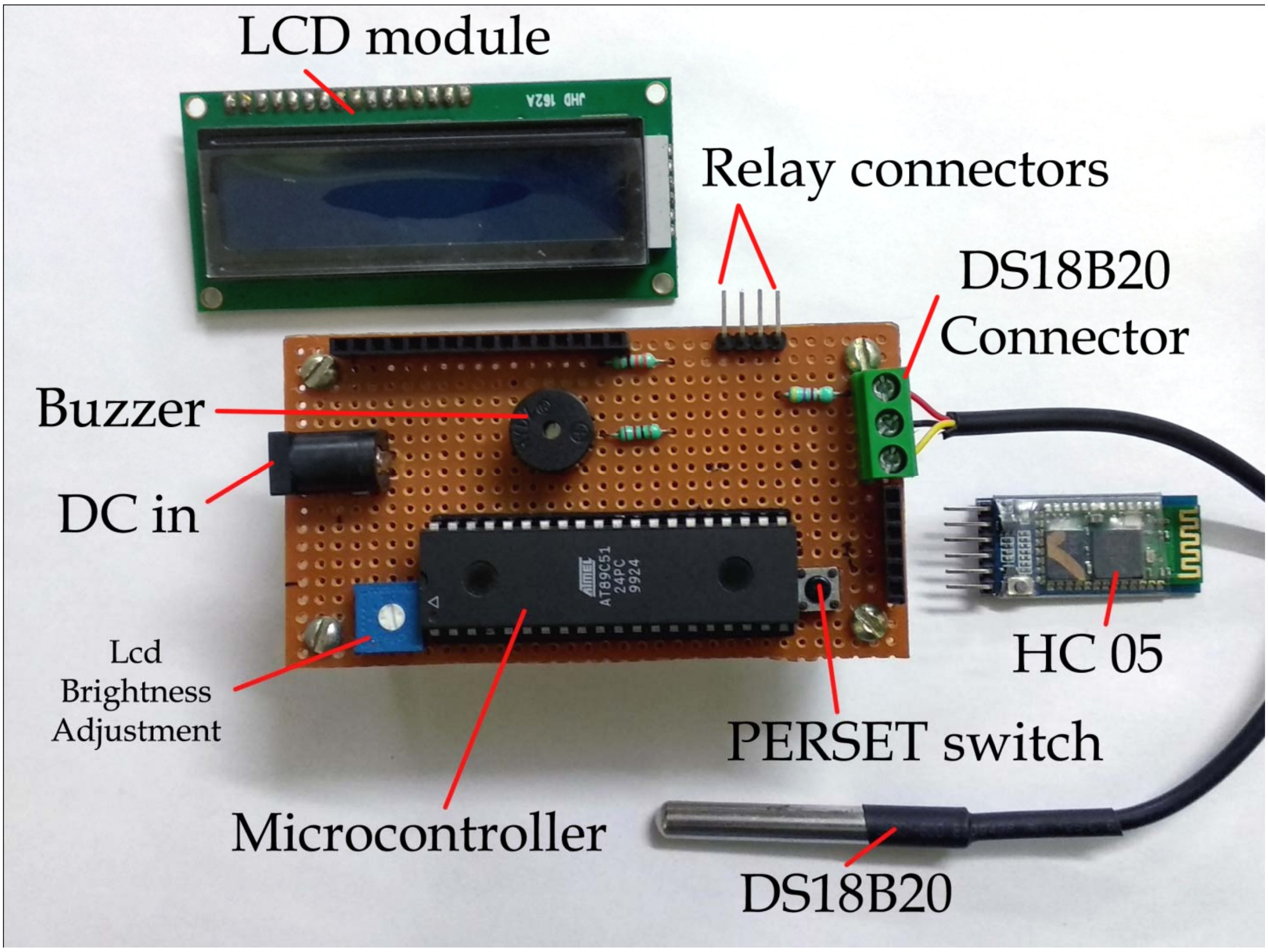
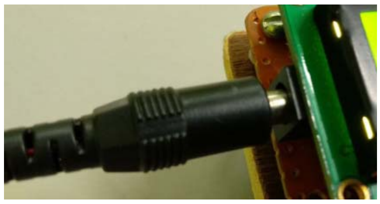


Figure 11: Main Controller and Relay board

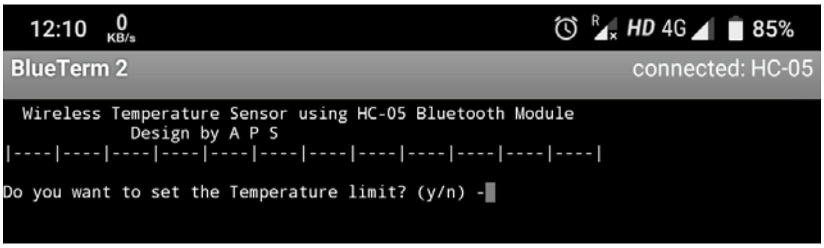
4.5 Step by step operation of the prototype

1. Connect the dc power jack to the prototype board (5V DC).

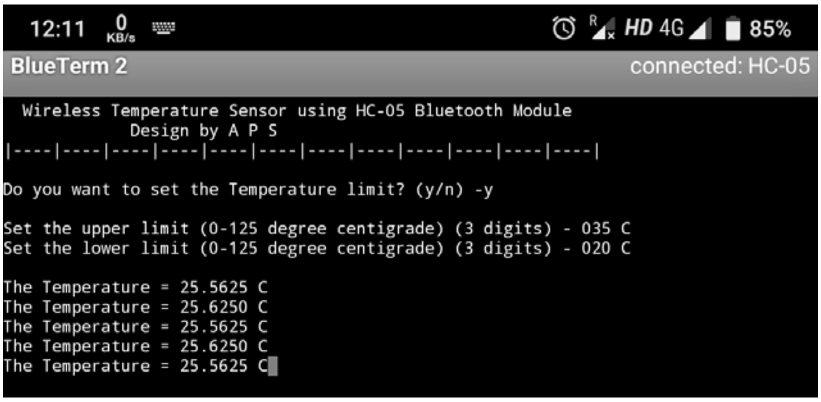


2. Open the app "Blue Term 2" in the mobile and connect the HC05 (choose from the physical address).

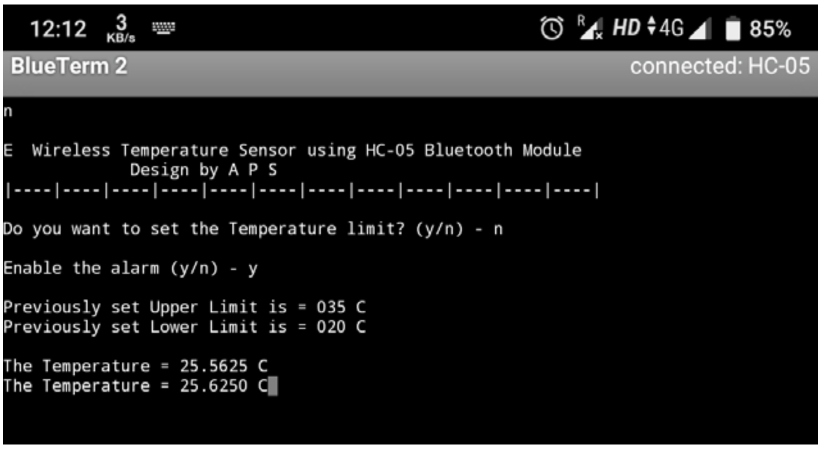
3. After connecting the HC 05 see the initial massage.



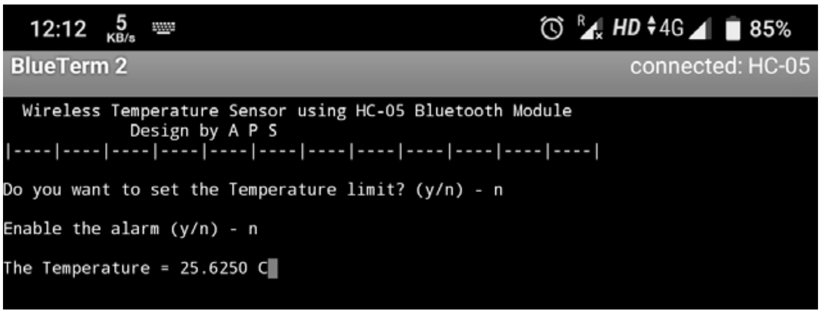
4. If you want to set the temperature limit then set the upper and lower limit in degree celcious



5. If you do not want to set the temperature limit then it will display the previously set temperature limits and ask for enabling the alarm



6. If you do not want to set the alarm then only the temperature will display on the screen.



4.6 Components required

Table 4: Component listing

Sl. No.	Component	Qtn
1.	HC-05	1
2.	MAX 232	1
3.	1 µF, 16V Capacitor	4
4.	AT89c51 µC	1
5.	11.0592 MHz Crystal	1
6.	33 pf Capacitor	2
7.	0.1µF Capacitor	2
8.	10µF Capacitor	2
9.	ULN 2003 A IC	1
10.	Static Relay (5 volt)	2
11.	16x2 LCD Module	1
12.	3 mm LED (Red/Green)	2
13.	General blank PCB	1
14.	Jumper wire	9
15.	330 Ω Resistance	2
16.	Piezo Buzzer	1
17.	16 pin IC base	1
18.	40 pin IC base	1
19.	Single strand wire	3m
20.	Wire nipper	1
21.	Wire striper	1
22.	Soldering Iron	1
23.	Soldering material	1
24.	De-soldering pump	1
25.	Female pin header	1
26.	Male pin header	1
27.	10K POT	1

4.7 Cost estimation of the prototype

Table 5: Cost estimation of the project

Sl. No.	Component	Qtn	Price
1.	HC-05	1	350
2.	MAX 232	1	20
3.	1 µF, 16V Capacitor	4	8
4.	AT89c51 µC	1	45
5.	11.0592 MHz Crystal	1	10
6.	33 pf Capacitor	2	2
7.	0.1µF Capacitor	2	2
8.	10µF Capacitor	2	4
9.	ULN 2003 A IC	1	20
10.	Static Relay (5 volt)	2	40
11.	16x2 LCD Module	1	130

12.	3 mm LED (Red/Green)	2	3
13.	General blank PCB	1	30
14.	Jumper wire	9	30
15.	330 Ω Resistance	2	2
16.	Piezo Buzzer	1	10
17.	16 pin IC base	1	5
18.	40 pin IC base	1	5
19.	Single strand wire	3m	30
20.	Female pin header	1	20
21.	10K POT	1	10
Total			**780/-**

4.8 Hardware connection

4.8.1 16 × 2 LCD Module interfacing

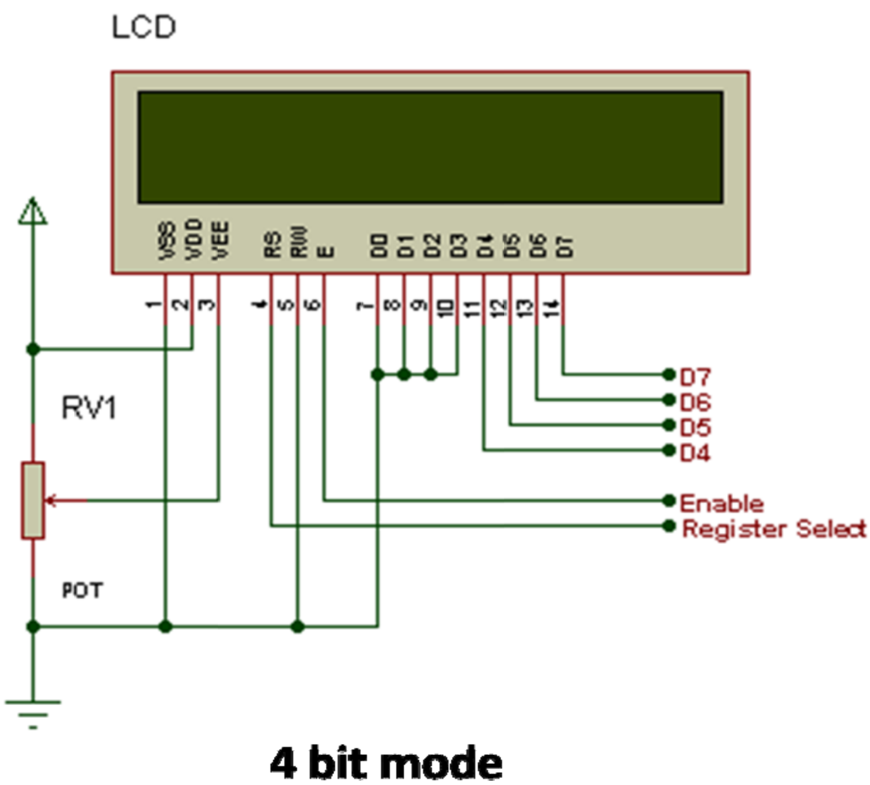


Figure 12: 4 bit mode LCD Interfacing

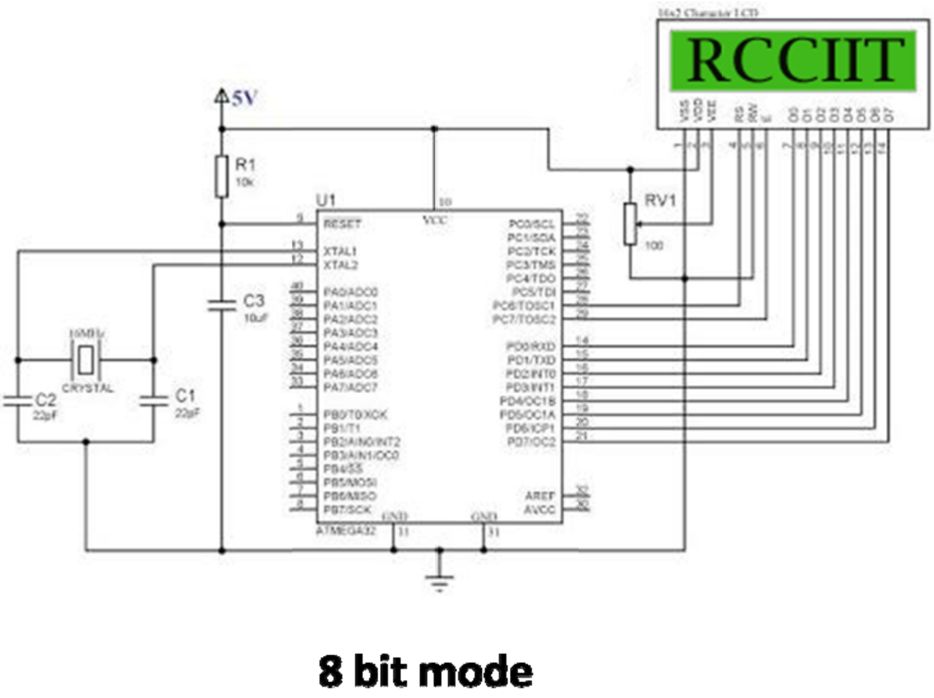


Figure 13: 8 bit mode LCD Interfacing

4.8.2 LCD AT Commands

Table 6: LCD Attention (AT) Command

Code (Hex)	Command to LCD Instruction Register
1	Clear display screen
2	Return home
4	Decrement cursor (shift cursor to left)
6	Increment cursor (shift cursor to right)
5	Shift display right
7	Shift display left
8	Display off, cursor off
A	Display off, cursor on
C	Display on, cursor off
E	Display on, cursor blinking
F	Display on, cursor blinking
10	Shift cursor position to left
14	Shift cursor position to right
18	Shift the entire display to the left
1C	Shift the entire display to the right
80	Force cursor to beginning to 1st line
C0	Force cursor to beginning to 2nd line
38	2 lines and 5x7 matrix

4.8.3 LCD Reset subroutine in AT89c51

```
lcd_reset:                    ;LCD reset sequence
        mov lcd_port, #0FFH
        mov delay,#20         ;20mS delay
        acall delayms
        mov lcd_port, #83H    ;Data = 30H, EN = 1, First Init
        mov lcd_port, #03H    ;Data = 30H, EN = 0
        mov delay,#10         ;Delay 10mS
        acall delayms
        mov lcd_port, #83H    ;Second Init, Data = 30H, EN = 1
        mov lcd_port, #03H    ;Data = 30H, EN = 0
        mov delay,#1          ;Delay 5mS
        acall delayms
        mov lcd_port, #83H    ;Third Init
        mov lcd_port, #03H
        mov delay,#1          ;Delay 5mS
        acall delayms
        mov lcd_port, #82H    ;Select Data width (20H for 4bit)
        mov lcd_port, #02H    ;Data = 20H, EN = 0
        mov delay,#1          ;Delay 5mS
        acall delayms
ret
```

4.8.4 LCD Initialization subroutine in AT89c51 (4 bit mode)

```
 lcd_init:
        mov a,#28H            ;4-bit, 2 line, 5x7 dots
        acall CMND
        mov a,#0CH            ;LCD_DATAlay ON cursor OFF
        acall CMND
        mov a,#06H            ;Set entry mode (Auto increment)
        acall CMND
        mov a,#80H            ;Bring cursor to line 1
        acall CMND
 ret
```

4.8.5 LCD Command subroutine in AT89c51 (4 bit mode)

```
 CMND:                        ;LCD command Routine
        mov temp,a            ;Save a copy of command to temp
        swap a                ;Swap to use higher nibble
        anl a,#0FH            ;Mask the first four bits
        add a,#80H            ;Enable = 1, RS = 0, RW = 0
        mov lcd_port,a        ;Move it to lcd port
        mov delay,#3          ;5mS delay
```

```
        acall delayms
        CLR EN

        mov a,temp           ;Reload the command from temp
        anl a,#0FH           ;Mask first four bits
        add a,#80H           ;Enable = 1
        mov lcd_port,a       ;Move to port
        mov delay,#3         ;5mS delay
        acall delayms
        CLR EN
ret
```

4.8.6 LCD Display subroutine in AT89c51 (4 bit mode)

```
DISP:                        ;LCD data Routine
        mov temp,a           ;Keep copy of data in temp
        swap a               ;We need higher nibble
        anl a,#0FH           ;Mask first four bits
        add a,#0A0H          ;Enable = 1, RS = 1, RW = 0
        mov lcd_port,a       ;Move to lcd port
        mov delay,#3         ;5mS delay
        acall delayms
        clr en               ;Enable = 0

        mov a,temp           ;Reload the data from temp
        anl a,#0FH           ;we need lower nibble now
        add a,#0A0H          ;Enable = 1, RS = 1, RW = 0
        mov lcd_port,a       ;Move to lcd port
        mov delay,#3         ;5mS delay
        acall delayms
        clr en               ;Enable = 0
ret
```

4.8.7 Relay Driver interfacing with microcontroller

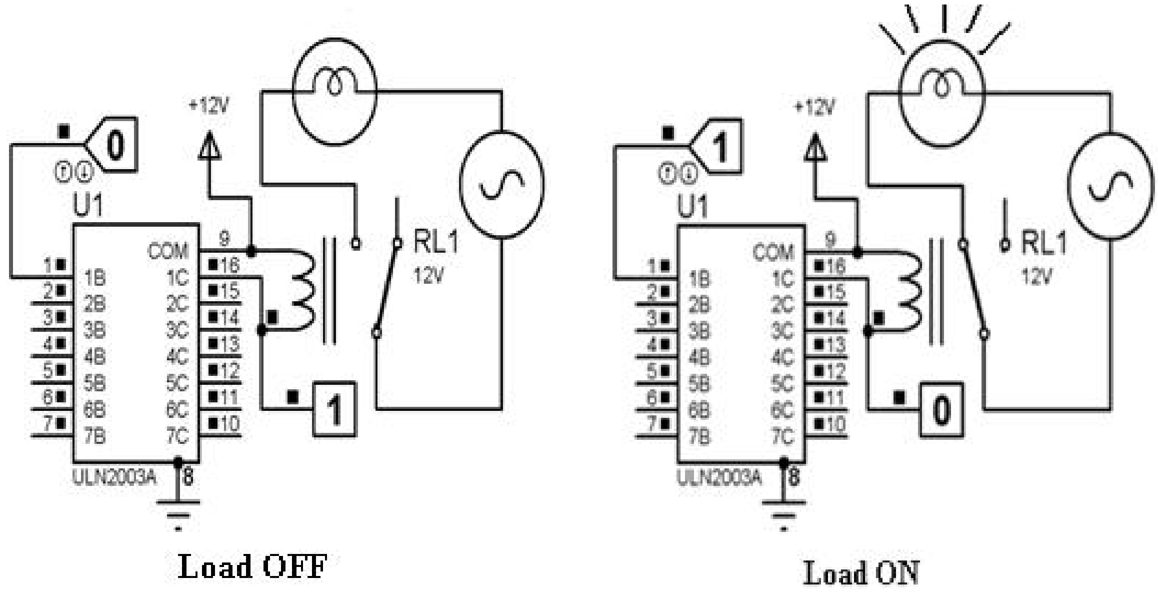


Figure 14: ULN2003A interfacing with microcontroller

The ULN2003A is a active high relay driver. 7 relays are controlled by this relay driver. Pin 1-7 are for controlling the relay which are connected to pin 10-16. For a ‘0’ from microcontroller the corresponding relay is turned off and a ‘1’ from microcontroller is turned on the relay.

4.8.8 DS18B20 interfacing with microcontroller

The one wire temperature sensor DS18B20 can be interfaced with most of the microcontroller. In this project it is connected with AT89c51 microcontroller. A 4.7 KΩ pullup resistor is required for the proper connection. The beauty of this temperature sensor is that it occupies only one digital IO pin of the microcontroller. It can be possible to connect infinite number of sensor in the same pin because each sensor has a unique 64 bit address. In figure 6 it is shown that a temperature sensor is connected to 8051 microcontroller.

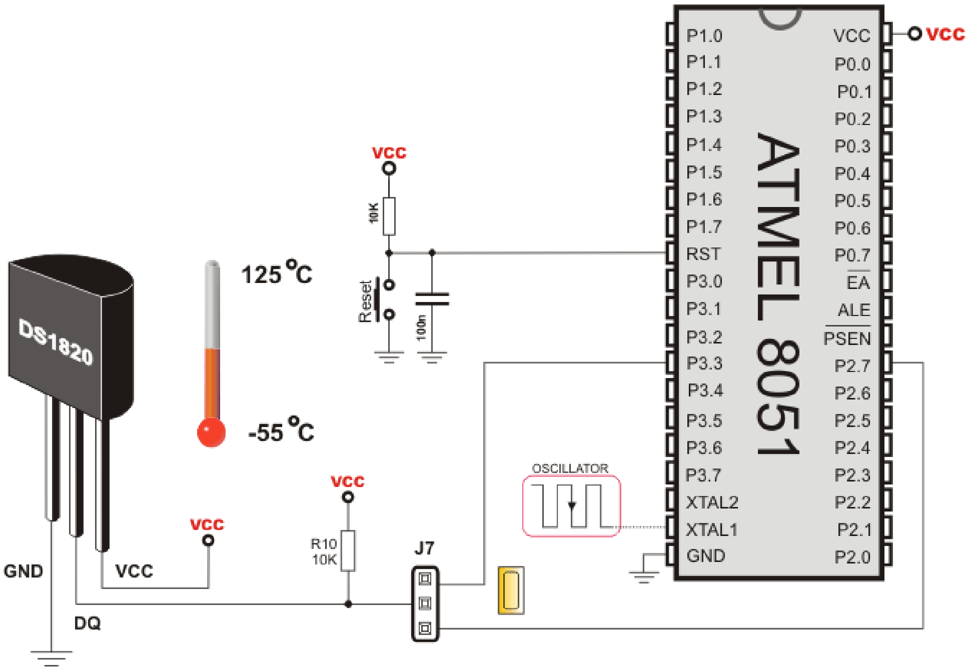


Figure 15: DS18B20 interfacing with microcontroller

4.8.9 Subroutines for interfacing ds18b20 with microcontroller

RESET of DS18B20

```
CONFIGURE18B20:
    CLR DQ
    ACALL DELAY480US
    SETB DQ                     ;DQ=P1.0=1 (make it as an input port)
    ACALL DELAY480US
RET
```

READ BYTE of DS18B20

```
READ18B20:
     MOV R0, #08H        ; Set Counter for 8 bits
NEXTBITREAD:
     CLR DQ          ; Start clock cycle
     NOP
     SETB DQ
     ACALL DELAY10US
     MOV C, DQ     ; Move incoming bit to DQ
     RRC A           ; Rotate A Right through Carry Bit (C goes to Highest Bit of A)
     ACALL DELAY55US
     SETB DQ
     DJNZ R0, NEXTBITREAD
RET
```

WRITE BYTE of DS18B20

```
WRITE18B20:
        MOV R0, #08H          ; Set Counter for 8 bits
NEXTBITWRITE:
        RRC A         ; Rotate A Right into Carry Bit (Lowest Bit in A goes to C)
        JC WRITE1
        CLR DQ
        ACALL DELAY10US
        ACALL DELAY10US
        ACALL DELAY10US
        MOV DQ, C   ; Move outgoing bit to DQ
        ACALL DELAY70US
        SETB DQ
        DJNZ R0, NEXTBITWRITE
        SJMP LAST
WRITE1:    CLR DQ
        nop
        SETB DQ                   ; Rising Edge of Clock makes one clock cycle
        ACALL DELAY24US
        MOV DQ, C
        ACALL DELAY10US
        ACALL DELAY10US
        ACALL DELAY10US
        ACALL DELAY10US
        ACALL DELAY10US
        ACALL DELAY10US
        SETB DQ
        DJNZ R0, NEXTBITWRITE
LAST:        RET
```

CHAPTER 5

(Logic & Operation)

5.1 INTRODUCTION

After assembling the system, what remains is to observe its operation and efficiency of the system. The total system is divided in several sub systems, like

- DS18B20 interfacing
- HC 05 (Bluetooth Transceiver) interfacing
- Relay with driver
- LCD section

The operation of the whole circuit is depending on every sections performance.

5.2Flow Diagram

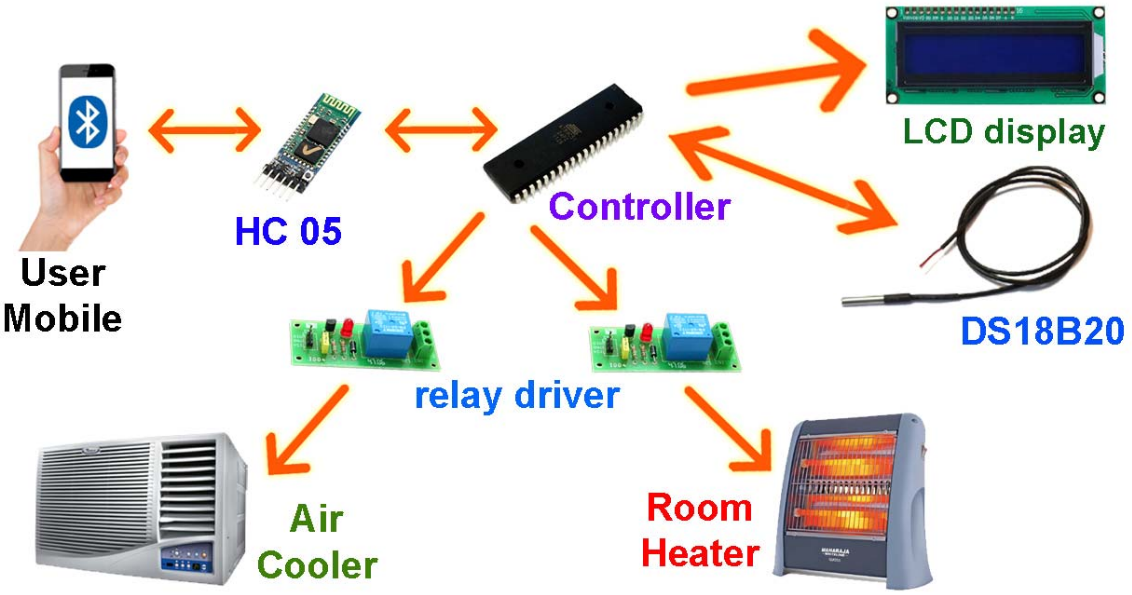


Figure 16: flow diagram of the prototype

5.3 Principle & Operations

DS18B20 is a digital thermometer and it communicates with the microcontroller on 'one wire' protocol. Only one digital IO pin is required to connect the DS18B20 with the microcontroller. Literally infinite number of sensors may be connected in the same IO pin because each sensor has a unique 64 bit address on it. After sensing the temperature it converts the temperature in digital form with 16 bit digital data. The resolution of the temperature sensor is user-configurable to 9, 10, 11, or 12 bits, corresponding to increments of 0.5°C, 0.25°C, 0.125°C, and 0.0625°C, respectively. The default resolution at power-up is 12-bit. In this project user can set the upper limit and lower limit of the temperature from a remote place using Bluetooth communication protocol. These limits are written in the memory register inside DS18B20. If the temperature crosses the limits a buzzer will sound and the corresponding load (heater/cooler) will ON. The temperature will display on the local device, i.e. a 16X2 alphanumeric LCD screen as well as remotely (user mobile screen) simultaneously.

5.4 Advantages of the project

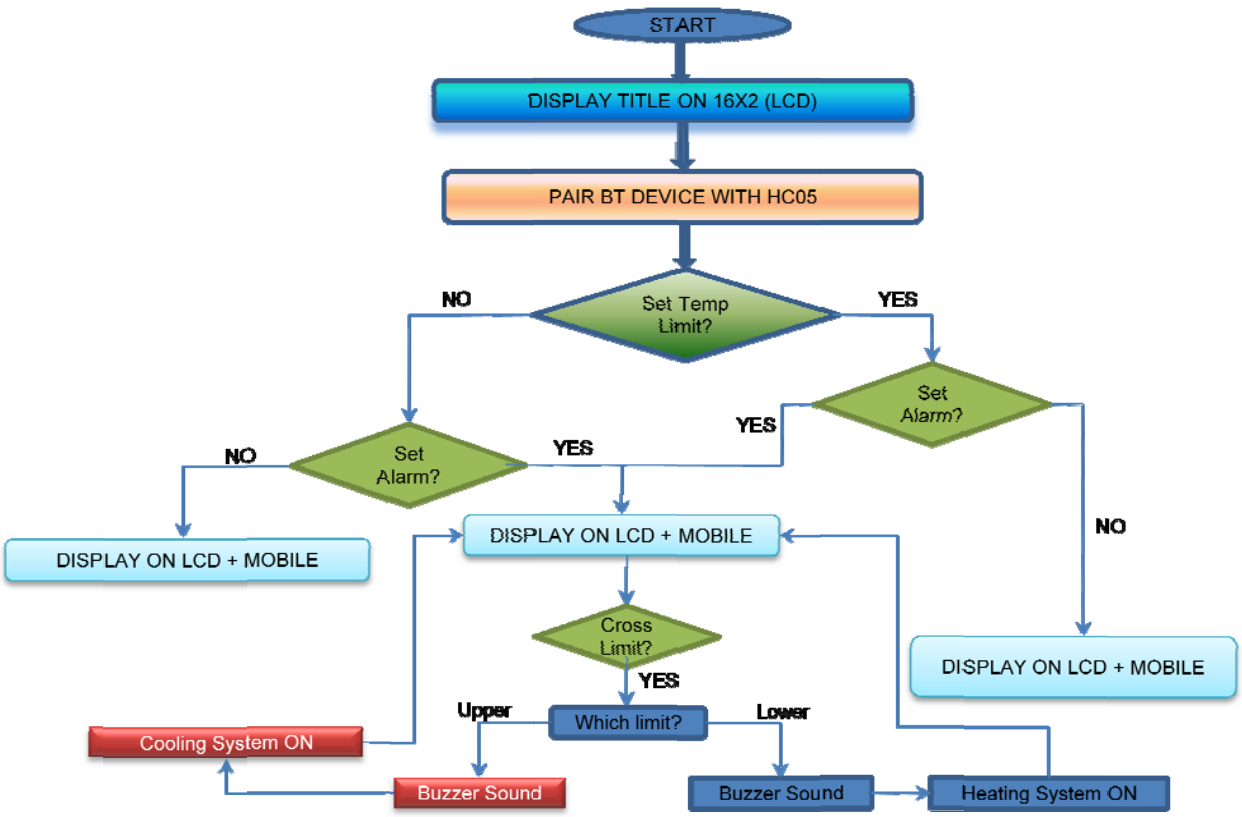
- Very accurate temperature measurement (resolution 0.0625 ^{0}C)
- Cost effective
- Relay driver included
- 250V, 7A load may be connected at the output
- Remote operation
- LCD display for temperature
- Alarm signal for attention of the observer

5.5 Disadvantages

- Limited range (20 – 30 meter) as it uses Bluetooth technology.
- Large electrical load cannot be handled by the circuit (250V, 7A max)**
- Once set the limits, user cannot change the limits without resetting the microcontroller.

** it may be overcome if we replace the static relay by electrical contactor.

5.6 Flow chart of the program



5.7 Photographs of the prototype

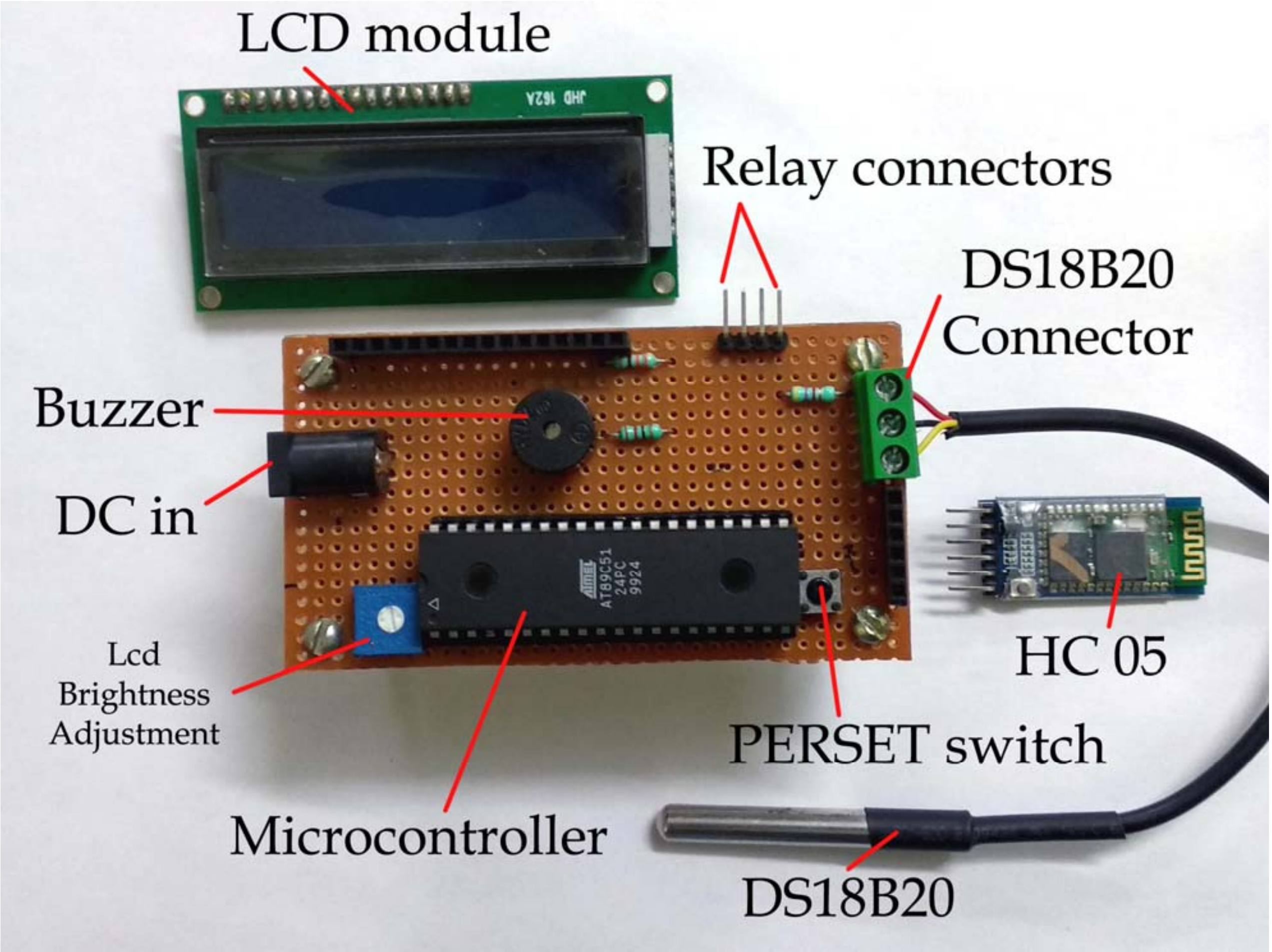


Figure 17: complete photograph of the prototype

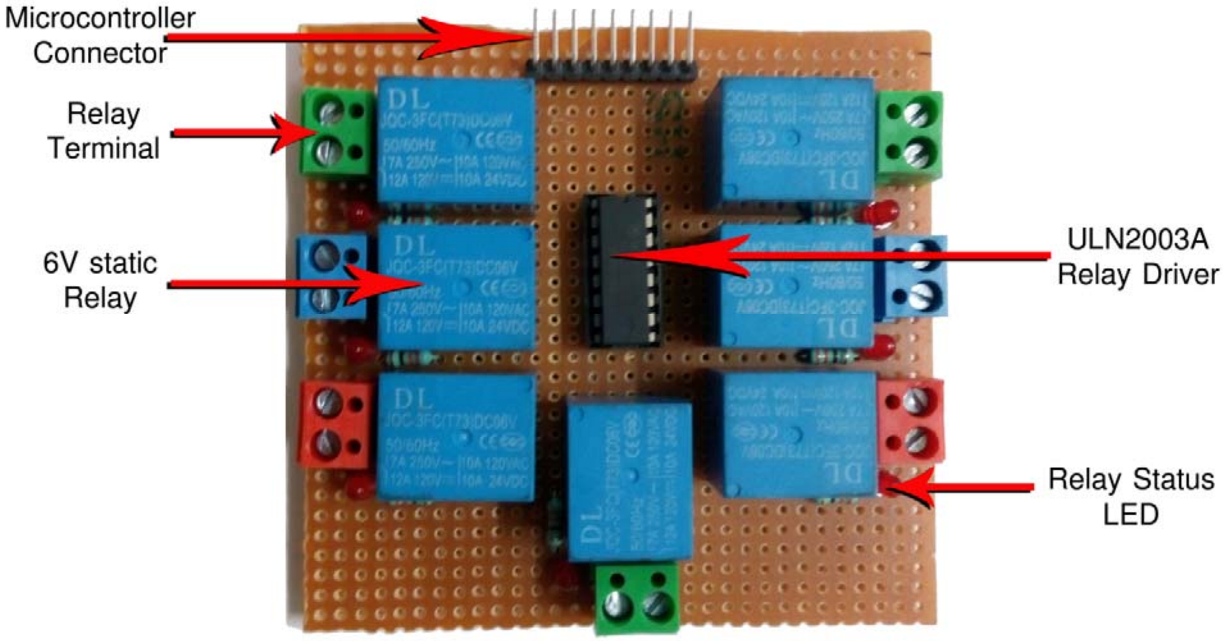


Figure 18: Relay Driver Board

5.8 Circuit diagram

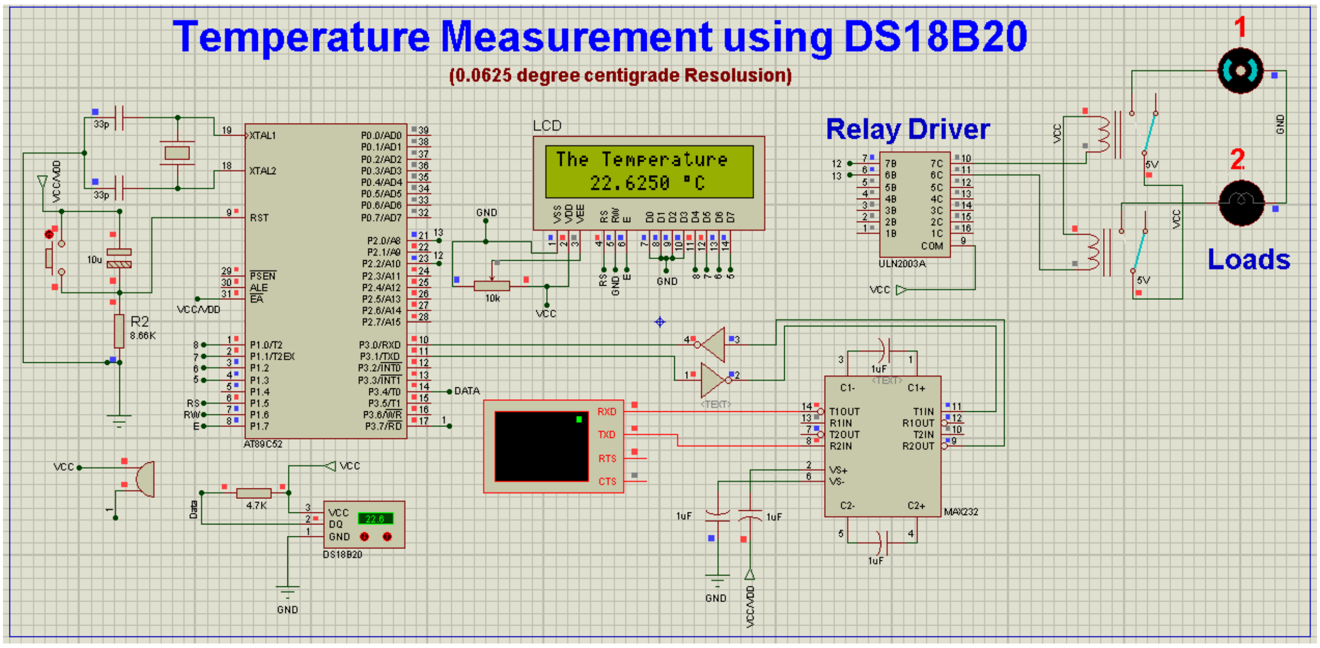


Figure 19: Hardware Circuit Diagram

Chapter 6

(Conclusion & Future Scope)

6.1 CONCLUSION

Here we have developed a wireless temperature monitoring and control system using Bluetooth.

In this project we use Temperature Sensor (DS18B20) which senses the temperature of a particular area; a piezo electric buzzer gets activated whenever a particular threshold temperature is sensed. Also a heater and a cooler is activated based on the temperature limit. The threshold value of temperature of different areas can be set accordingly so that we are alarmed based on that value. In such areas we can use this model so that it will help us in sensing the danger level of that area and help us in preventing any loss of life.

In this model, all the data of temperature are monitored each and every time in LCD and retrieved in mobile app through Bluetooth module. User can connect the smart phone wirelessly with the system and monitor the process variable. When the app is running on the smart phone, data sent by HC-05 Bluetooth module is received and displayed on the app.

6.2 RESULTS

The experimental model was made following the circuit diagram and the desired results were obtained .Every time the temperature surpassed the upper threshold limit the cooler was activated. Similarly the heater was activated whenever the lower limit was crossed.

6.3 FUTURE WORK

Temperature measurement and control are vital in many industrial processes. Accurate control of the temperature is essential in nearly all chemical processes.

Hence the future scope of this technology is wide spread and quite essential in both domestic and industrial applications.

Chapter 7

(References)

1. Rashmi Singh and S.P Singh***, Development of a Low Cost Wireless Temperature Monitoring System for Industrial & Research Application ,*** School of Energy and Environmental Studies, Devi Ahilya University, Khandwa Road, Indore 452001, India , 07 Feb 2015

2. Pradeep Kasale, Shekhar Kedar, Mrinal Kishore, Prof. Kanchan Maske, ***Android based Greenhouse Monitoring and Controlling System,*** Electronics and Telecommunication & Pune University, India, March 2015, pg.68 – 71

3. Purnima, S.R.N. Reddy, PhD. ***Design of Remote Monitoring and Control System with Automatic Irrigation System using GSM-Bluetooth***,International Journal of Computer Applications (0975 – 888), Volume 47– No.12, June 2012

4. Mohammed Juned, Srija Unnikrishnan***, Bluetooth Based Remote Monitoring & Control System,*** Journal of Basic and Applied Engineering Research, Volume 1, Number 8; October, 2014 pp. 108-111

5. R Vijayarani, S. Praveen Kumar, Scholars, SRM University, Kattankulathur, Chennai, ***Environmental Monitoring and Controlling Various Parameters in a Closed Loop,*** International Journal of Engineering Research and General Science Volume 2, Issue 3, April-May 2014.

6. K. Jaganmohan Reddy, B.Santhosh Kumar, ***Patient Temperature Monitoring System Using Bluetooth Communication ,*** International Journal of Engineering Trends and Technology (IJETT) – Volume 16 Number 4 – Oct 2014.

Appendix A

(Hardware description)

Transformer less AC to DC power supply circuit using dropping capacitor

Production of low voltage DC power supply from AC power is the most important problem faced by many electronics developers and hobbyists. The straight forward technique is the use of a step down transformer to reduce the 230 V or 110V AC to a preferred level of low voltage AC. But *SMPS* power supply comes with the most appropriate method to create a low cost power supply by avoiding the use of bulky transformer. This circuit is so simple and it uses a voltage dropping capacitor in series with the phase line. Transformer less power supply is also called as capacitor power supply. It can generate 5V, 6V, 12V 150mA from 230V or 110V AC by using appropriate zener diodes.

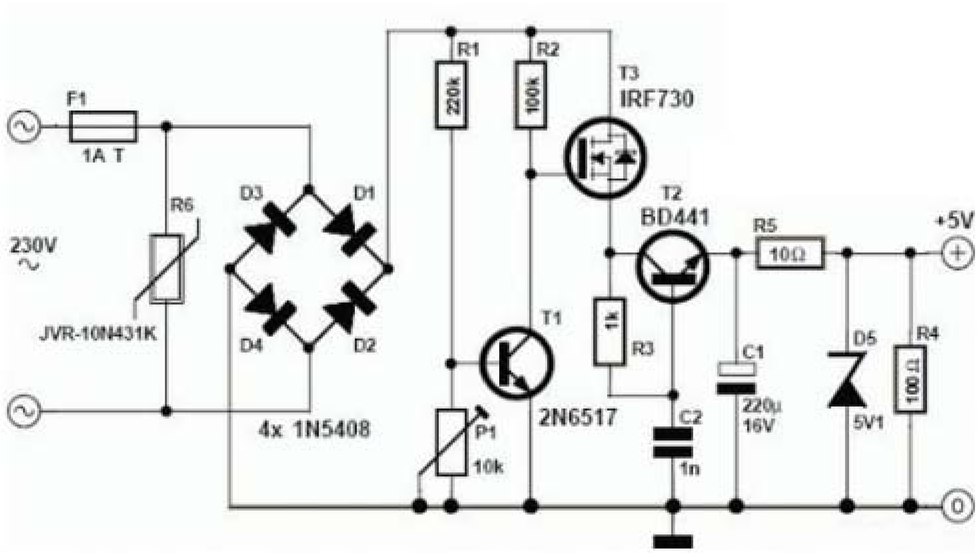


Figure 21: Transformer less SMPS 5 volt power supply

Working of Transformer less capacitor power supply

- This transformer less power supply circuit is also named as capacitor power supply since it uses a special type of AC capacitor in series with the main power line.
- A common capacitor will not do the work because the mains spikes will generate holes in the dielectric and the capacitor will be cracked by passing of current from the mains through the capacitor.
- X rated capacitor suitable for the use in AC mains is vital for reducing AC voltage.
- A X rated dropping capacitor is intended for 250V, 400V, 600V AC. Higher voltage versions are also obtainable. The dropping capacitor is non polarized so that it can be connected any way in the circuit.
- The 470kΩ resistor is a bleeder resistor that removes the stored current from the capacitor when the circuit is unplugged. It avoids the possibility of electric shock.
- Reduced AC voltage is rectified by bridge rectifier circuit. We have already discussed about bridge rectifiers. Then the ripples are removed by the 1000µF capacitor.

- This circuit provides 24 volts at 160 mA current at the output. This 24 volt DC can be regulated to necessary output voltage using an appropriate 1 watt or above zener diode.

- Here we are using 6.2V zener. You can use any type of zener diode in order to get the required output voltage.

AT 89c51 Microcontroller

AT89C51 is an 8-bit microcontroller and belongs to Atmel's 8051 family. **ATMEL 89C51** has 4KB of Flash programmable and erasable read only memory (PEROM) and 128 bytes of RAM. It can be erased and program to a maximum of 1000 times.

In 40 pin AT89C51, there are four ports designated as P_1, P_2, P_3 and P_0. All these ports are 8-bit bi-directional ports, *i.e.*, they can be used as both input and output ports. Except P_0 which needs external pull-ups, rest of the ports have internal pull-ups. When 1s are written to these port pins, they are pulled high by the internal pull-ups and can be used as inputs. These ports are also bit addressable and so their bits can also be accessed individually.

Port P_0 and P_2 are also used to provide low byte and high byte addresses, respectively, when connected to an external memory. Port 3 has multiplexed pins for special functions like serial communication, hardware interrupts, timer inputs and read/write operation from external memory. AT89C51 has an inbuilt UART for serial communication. It can be programmed to operate at different baud rates. Including two timers & hardware interrupts, it has a total of six interrupts.

PIN Diagram:

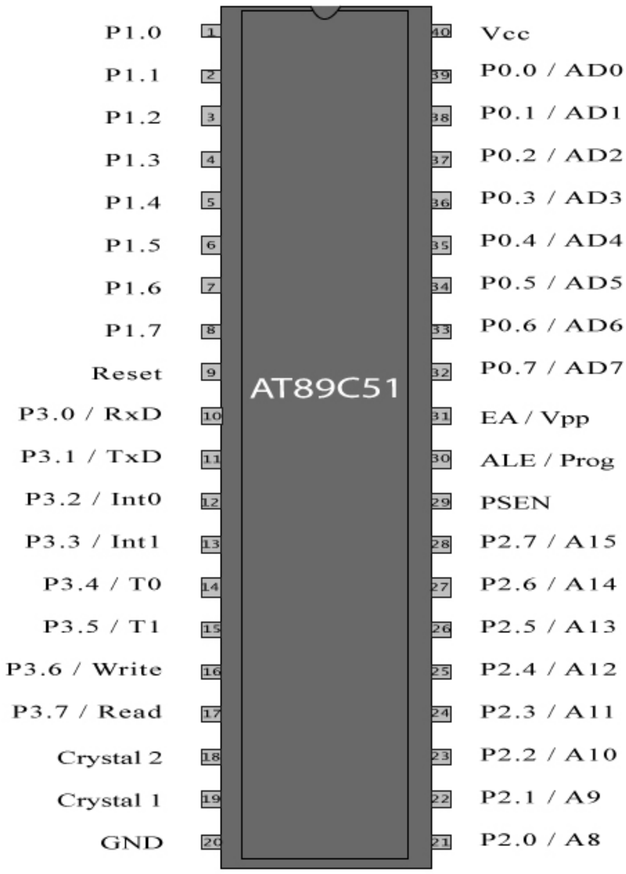


Figure 22: 89c51 Microcontroller Pin Diagram

PIN Description:

<table>
<tr><th>Pin No</th><th colspan="3">Function</th><th>Name</th></tr>
<tr><td>1</td><td colspan="3" rowspan="8">8 bit input/output port (P_1) pins</td><td>$P_1.0$</td></tr>
<tr><td>2</td><td>$P_1.1$</td></tr>
<tr><td>3</td><td>$P_1.2$</td></tr>
<tr><td>4</td><td>$P_1.3$</td></tr>
<tr><td>5</td><td>$P_1.4$</td></tr>
<tr><td>6</td><td>$P_1.5$</td></tr>
<tr><td>7</td><td>$P_1.6$</td></tr>
<tr><td>8</td><td>$P_1.7$</td></tr>
<tr><td>9</td><td colspan="3">Reset pin; Active high</td><td>Reset</td></tr>
<tr><td>10</td><td>Input (receiver) for serial communication</td><td>RxD</td><td rowspan="8">8 bit input/output port (P_3) pins</td><td>$P_3.0$</td></tr>
<tr><td>11</td><td>Output (transmitter) for serial communication</td><td>TxD</td><td>$P_3.1$</td></tr>
<tr><td>12</td><td>External interrupt 1</td><td>Int0</td><td>$P_3.2$</td></tr>
<tr><td>13</td><td>External interrupt 2</td><td>Int1</td><td>$P_3.3$</td></tr>
<tr><td>14</td><td>Timer1 external input</td><td>T_0</td><td>$P_3.4$</td></tr>
<tr><td>15</td><td>Timer2 external input</td><td>T_1</td><td>$P_3.5$</td></tr>
<tr><td>16</td><td>Write to external data memory</td><td>Write</td><td>$P_3.6$</td></tr>
<tr><td>17</td><td>Read from external data memory</td><td>Read</td><td>$P_3.7$</td></tr>
<tr><td>18</td><td colspan="3" rowspan="2">Quartz crystal oscillator (up to 24 MHz)</td><td>Crystal 2</td></tr>
<tr><td>19</td><td>Crystal 1</td></tr>
<tr><td>20</td><td colspan="3">Ground (0V)</td><td>Ground</td></tr>
<tr><td>21</td><td colspan="3" rowspan="8">8 bit input/output port (P_2) pins
/
High-order address bits when interfacing with external memory</td><td>$P_2.0/ A_8$</td></tr>
<tr><td>22</td><td>$P_2.1/ A_9$</td></tr>
<tr><td>23</td><td>$P_2.2/ A_{10}$</td></tr>
<tr><td>24</td><td>$P_2.3/ A_{11}$</td></tr>
<tr><td>25</td><td>$P_2.4/ A_{12}$</td></tr>
<tr><td>26</td><td>$P_2.5/ A_{13}$</td></tr>
<tr><td>27</td><td>$P_2.6/ A_{14}$</td></tr>
<tr><td>28</td><td>$P_2.7/ A_{15}$</td></tr>
<tr><td>29</td><td colspan="3">Program store enable; Read from external program memory</td><td>PSEN</td></tr>
<tr><td>30</td><td colspan="3">Address Latch Enable</td><td>ALE</td></tr>
<tr><td></td><td colspan="3">Program pulse input during Flash programming</td><td>Prog</td></tr>
<tr><td>31</td><td colspan="3">External Access Enable; Vcc for internal program executions</td><td>EA</td></tr>
<tr><td></td><td colspan="3">Programming enable voltage; 12V (during Flash programming)</td><td>Vpp</td></tr>
<tr><td>32</td><td colspan="3" rowspan="8">8 bit input/output port (P_0) pins

Low-order address bits when interfacing with external memory</td><td>$P_0.7/ AD_7$</td></tr>
<tr><td>33</td><td>$P_0.6/ AD_6$</td></tr>
<tr><td>34</td><td>$P_0.5/ AD_5$</td></tr>
<tr><td>35</td><td>$P_0.4/ AD_4$</td></tr>
<tr><td>36</td><td>$P_0.3/ AD_3$</td></tr>
<tr><td>37</td><td>$P_0.2/ AD_2$</td></tr>
<tr><td>38</td><td>$P_0.1/ AD_1$</td></tr>
<tr><td>39</td><td>$P_0.0/ AD_0$</td></tr>
<tr><td>40</td><td colspan="3">Supply voltage; 5V (up to 6.6V)</td><td>Vcc</td></tr>
</table>

Table 7: Pin Description of 89c51 microcontroller

16x2 LCD Module:

- 16 character 2 lines display
- 4 bit and 8 bit data transfer mode
- display alpha numeric display
- backlight compatible
- contrast adjustment
- backlight intensity adjustment
- 5 volt operation
- compatible to almost every microcontroller

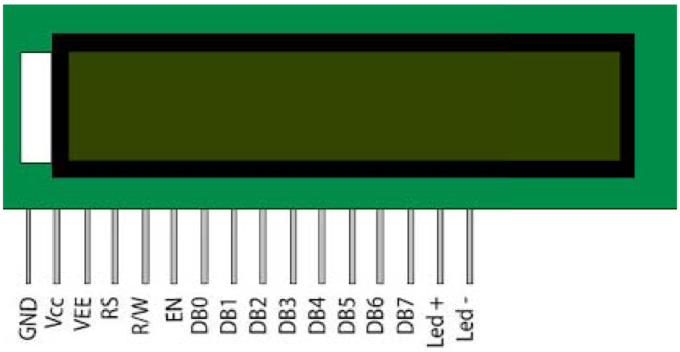


Figure 23: 16X2 LCD Module

LCD Pin outs

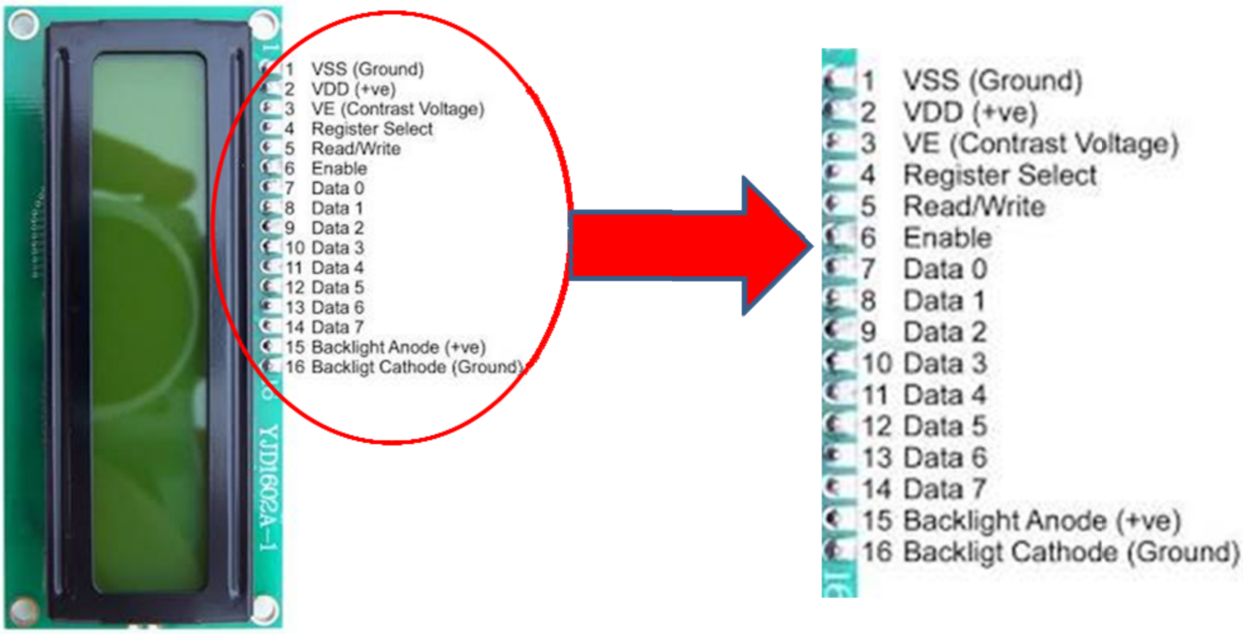


Figure 24: LCD Pin Diagram

Relay Driver

- The ULN2003 is a monolithic high voltage and high current Darlington transistor arrays.
- It consists of seven NPN Darlington pairs that features high-voltage outputs with common-cathode clamp diode for switching inductive loads.
- The collector-current rating of a single Darlington pair is 500mA.
- The ULN functions as an inverter.
- If the logic at input 1B is high then the output at its corresponding pin 1C will be low.

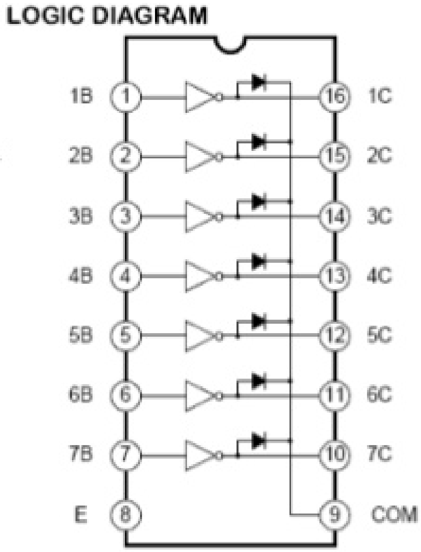


Figure 25: ULN2003A Internal Block Diagram

Resistor

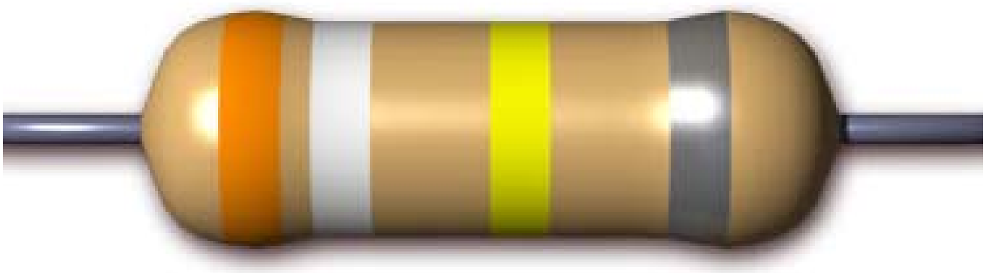


Figure 26: Resistor

Resistance is the opposition of a material to the current. It is measured in Ohms Ω. All conductors represent a certain amount of resistance, since no conductor is 100% efficient. To control the electron flow (current) in a predictable manner, we use resistors. Electronic circuits use calibrated lumped resistance to control the flow of current. Broadly speaking, resistor can be divided into two groups viz. fixed & adjustable (variable) resistors. In fixed resistors, the value is fixed & cannot be varied. In variable resistors, the resistance value can be varied by an adjuster knob. It can be divided into (a) Carbon composition (b) Wire wound (c) Special type. The most common type of resistors used in our projects is carbon type. The resistance value is normally indicated by color bands. Each resistance has four colors, one of the band on either side will be gold or silver, this is called fourth band and indicates the tolerance, others three band will give the value of resistance (see table). For example if a resistor has the following marking on it say red, violet, gold. Comparing these colored rings with the color code, its value is 27000 ohms or 27 kilo ohms and its tolerance is ±5%. Resistor comes in various sizes (Power rating).The bigger the size, the more power rating of 1/4 watts. The four color rings on its body tells us the value of resistor value.

Color Code of the resistor

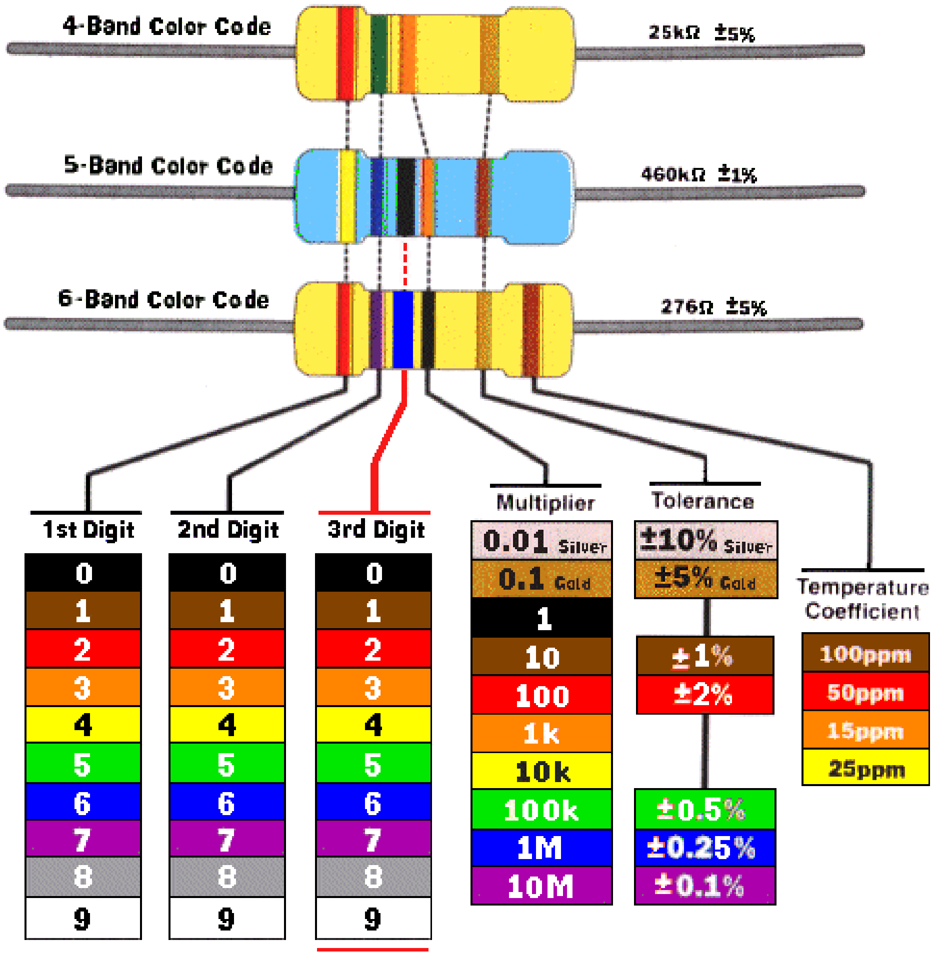


Figure 27: Color Code for resistance

RELAY

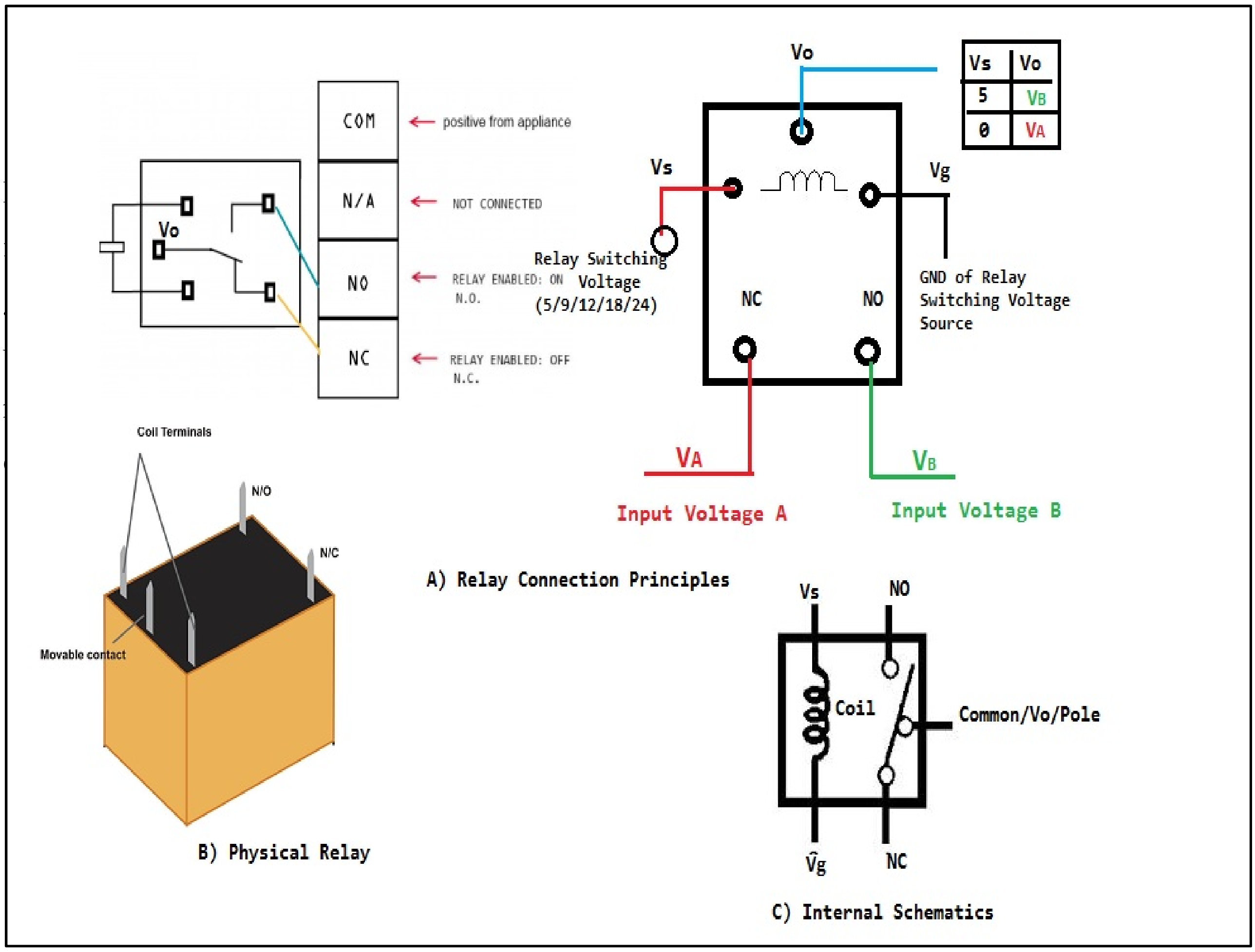


Figure 28: 6 volt Cube Relay

A relay is an electrically operated switch. Current flowing through the coil of the relay creates a magnetic field which attracts a lever and changes the switch contacts. The coil current can be on or off so relays have two switch positions and they are double throw (changeover) switches.

The relay's switch connections are usually labeled COM (POLE), NC and NO:

COM/POLE= Common, NC and NO always connect to this, it is the moving part of the switch.

NC = Normally Closed, COM/POLE is connected to this when the relay coil is not magnetized.

NO = Normally Open, COM/POLE is connected to this when the relay coil is MAGNETIZED and vice versa.

Capacitors

It is an electronic component whose function is to accumulate charges and then release it.

To understand the concept of capacitance, consider a pair of metal plates which all are placed near to each other without touching. If a battery is connected to these plates the positive pole to one and the negative pole to the other, electrons from the

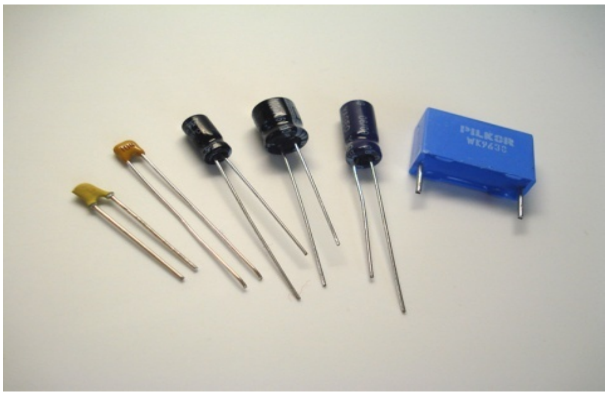


Figure 29: Types of capacitors

battery will be attracted from the plate connected to the positive terminal of the battery. If the battery is then disconnected, one plate will be left with an excess of electrons, the other with a shortage, and a potential or voltage difference will exists between them. These plates will be acting as capacitors. Capacitors are of two types: - (1) **fixed type** like ceramic, polyester, electrolytic capacitors - these names refer to the material they are made of aluminum foil. (2) **Variable type** like gang condenser in radio or trimmer. In fixed type capacitors, it has two leads and its value is written over its body and variable type has three leads. Unit of measurement of a capacitor is farad denoted by the symbol F. It is a very big unit of capacitance. Small unit capacitor are pico-farad denoted by pf (1pf=1/1000,000,000,000 f) Above all, in case of electrolytic capacitors, it's two terminal are marked as (-) and (+).

Crystal Oscillator

A crystal oscillator is an electronic oscillator circuit that uses the mechanical resonance of a vibrating crystal of piezoelectric material to create an electrical signal with a precise frequency. This frequency is commonly used to keep track of time, as in quartz wristwatches, to provide a stable clock signal for digital integrated circuits, and to stabilize frequencies for radio transmitters and receivers. The most common type of piezoelectric resonator used is the quartz crystal, so oscillator circuits incorporating them became known as crystal oscillators, but other piezoelectric materials including polycrystalline ceramics are used in similar circuits.

Figure 30: Crystal Oscillator

Quartz crystals are manufactured for frequencies from a few tens of kilohertz to hundreds of megahertz. More than two billion crystals are manufactured annually. Most are used for consumer devices such as wristwatches, clocks, radios, computers, and cell phones. Quartz crystals are also found inside test and measurement equipment, such as counters, signal generators, and oscilloscopes.

Piezo buzzer

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric. Typical uses of buzzers and beepers include alarm devices, timers and confirmation of user input such as a mouse click or keystroke. A piezoelectric element may be driven by an oscillating electronic circuit or other audio signal source, driven with a piezoelectric audio amplifier. Sounds commonly used to indicate that a button has been pressed are a click, a ring or a beep.

Figure 31: Piezo Buzzer

Blank PCB

A **printed circuit board** (**PCB**) mechanically supports and electrically connects electronic components using conductive tracks, pads and other features etched from copper sheets laminated onto a non-conductive substrate. PCBs can be *single sided* (one copper layer), *double sided* (two copper layers) or *multi-layer* (outer and inner layers). Multi-layer PCBs allow for much higher component density. Conductors on different layers are connected with plated-through holes called vias. Advanced PCBs may contain components - capacitors, resistors or active devices - embedded in the substrate.

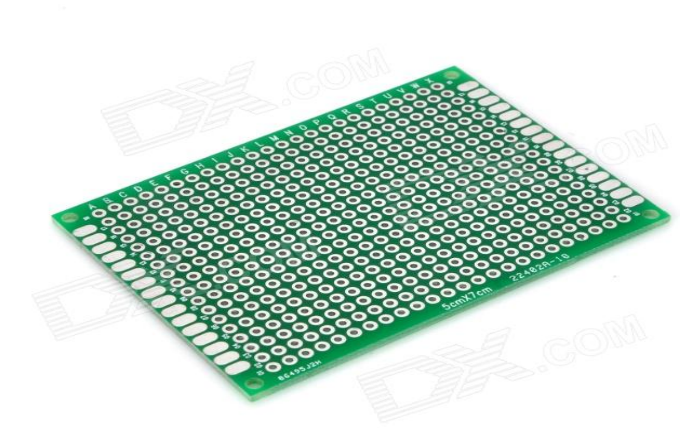


Figure 32: Blank glass epoxy PCB Board

FR-4 glass epoxy is the primary insulating substrate upon which the vast majority of rigid PCBs are produced. A thin layer of copper foil is laminated to one or both sides of an FR-4 panel. Circuitry interconnections are etched into copper layers to produce printed circuit boards. Complex circuits are produced in multiple layers.

Printed circuit boards are used in all but the simplest electronic products. Alternatives to PCBs include wire wrap and point-to-point construction. PCBs require the additional design effort to lay out the circuit, but manufacturing and assembly can be automated. Manufacturing circuits with PCBs is cheaper and faster than with other wiring methods as components are mounted and wired with one single part. Furthermore, operator wiring errors are eliminated.

Appendix B

(Software coding)

PROGRAM CODE:

```
;   Temperature measurement using DS18B20 and AT89c51
;   DQ Pin is connected to P3.4 (input/output)
;   all the delay routine is based on 11.0592 MHz Crystal
;-----------------------------------------------------------------------------
---
; LCD module is connected as
;                D4 - P1.0, D5 - P1.1, D6 - P1.2, D7 - P1.3
; Controls are connected as
;                EN - P1.7; RW - P1.6, RS - P1.5
;-----------------------------------------------------------------------------
---
;                D E C L E R A T I O N
;-----------------------------------------------------------------------------
---
    lcd_port equ P1        ;LCD connected to Port1
    en equ P1.7            ;Enable connected to P1.7
    DQ equ P3.4
    delay equ R7
    var1 equ R6
    temp1 equ 30H                   ;temporary storage of the temperature
    temp2 equ 31H                   ;temporary storage of the temperature
    temp3 equ 32H                   ;fractional part of temperature
    temp4 equ 33H                   ;0th digit of the temperature (integer)
    temp5 equ 34H                   ;10th digit of the temperature (integer)
    temp6 equ 35H                   ;100th digit of the temperature (integer)
    temp8 equ 36H                   ;temporary storage of the temperature
    temp9 equ 37H                   ;temporary storage of the temperature
    TEMP10 EQU 38H
    TEMPHIGH EQU 39H
    TEMPLOW EQU 3AH
    TEMP11 EQU 3BH
    TEMP12 EQU 3CH
    temp13 equ 3DH
    temp14 equ 3EH
    temp equ R5
    temp7 equ R4
;-----------------------------------------------------------------------------
---
ORG 00H
MOV temp8, #00H
MOV temp9, #00H
SETB P3.7
CLR P2.0               ; low limit relay connected
CLR P2.2               ; high limit relay connected
MOV TEMP14, #00H       ; ALARM FLAG INITIALLY 0
;-----------------------------------------------------------------------------
---
;                      Initial Message Display
;-----------------------------------------------------------------------------
---
ACALL LCD_RESET                 ;4 bit LCD RESET (software reset)
acall lcd_init         ;4 bit mode LCD initialization

MOV A,#01H                              ;Clear screen
ACALL LCD_CMD
MOV A, #81H
ACALL LCD_CMD
MOV DPTR, #MYDATA1                      ;'DS18B20 Based' Display
H_1:        CLR A
    MOVC A, @A+DPTR
    JZ b_1
    ACALL LCD_DATA
    INC DPTR
    SJMP H_1
b_1:
MOV A,#0C2H                             ;Jump to second line, position 5
ACALL LCD_CMD
    MOV DPTR, #MYDATA2          ;'Thermometer' Display
H_2:        CLR A
    MOVC A, @A+DPTR
    JZ b_2
    ACALL LCD_DATA
    INC DPTR
    SJMP H_2
b_2:        MOV DELAY, #250
    ACALL DELAYmS
    MOV DELAY, #250
    ACALL DELAYmS
    MOV DELAY, #250
    ACALL DELAYmS

MOV A,#01H                              ;Clear screen
ACALL LCD_CMD
;MOV A,#82H                             ;Cursor line one , position 2
;ACALL LCD_CMD
;   MOV DPTR, #MYDATA5          ;'Developed by' Display
;H_5:       CLR A
;   MOVC A, @A+DPTR
;   JZ b_5
;   ACALL LCD_DATA
;   INC DPTR
;   SJMP H_5
;b_5:
MOV A,#081H                     ;Jump to second line, position 1
ACALL LCD_CMD
    MOV DPTR, #MYDATA6          ;'BUDHADITYA' Display
H_6:        CLR A
    MOVC A, @A+DPTR
    JZ b_6
    ACALL LCD_DATA
    INC DPTR
    SJMP H_6
b_6:        MOV DELAY, #250
    ACALL DELAYmS
    MOV DELAY, #250
    ACALL DELAYmS
    MOV DELAY, #250
    ACALL DELAYmS

MOV A,#01H                              ;Clear screen
ACALL LCD_CMD
MOV A, #80H
ACALL LCD_CMD

    MOV DPTR, #MYDATA7          ;'THE TEMPERATURE' Display
H_7:        CLR A
    MOVC A, @A+DPTR
    JZ b_7
    ACALL LCD_DATA
    INC DPTR
    SJMP H_7
B_7:        MOV DELAY, #250
    ACALL DELAYmS
    MOV DELAY, #250
    ACALL DELAYmS
;-----------------------------------------------------------------------------
---
;                Serial Communication
;-----------------------------------------------------------------------------
---
    mov TMOD,#20h                   ;Timer 2 (8 bit auto reload mode)
      mov TH1,#0FDH                 ;9600 Baud rate in 11.0592 MHz Crystal
      mov SCON,#50h
      setb TR1
MOV DPTR, #MYDATA8                      ; Wireless Temperature Sensor using HC-05 Bluetooth Module
H_8:        CLR A
    MOVC A, @A+DPTR
    JZ B_8
    ACALL TRANS
    INC DPTR
    SJMP H_8
```

```
B_8:
MOV A, #0DH                          ; for 'ENTER'
ACALL TRANS
MOV DPTR, #MYDATA9
H_9:        CLR A
   MOVC A, @A+DPTR
   JZ B_9
   ACALL TRANS
   INC DPTR
   SJMP H_9
B_9:
MOV A, #0DH                          ; for 'ENTER'
ACALL TRANS
MOV DPTR, #MYDATA11                  ; Do you want to
set the Temperature limit? (y/n) - display
H_11:       CLR A
   MOVC A, @A+DPTR
   JZ B_11
   ACALL TRANS
   INC DPTR
   SJMP H_11
B_11:
   ACALL RECEIVE
   CJNE A, #6EH, YES2         ;if 'n(6E)' go to YES2
;  MOV TEMP14, #01H
   ACALL TRANS
   MOV A, #0DH
   ACALL TRANS
   SJMP YES3
YES2: LJMP YES
YES3:
MOV A, #0DH                          ; for 'ENTER'
ACALL TRANS
MOV DPTR, #MYDATA16                  ; Enable the alarm
(y/n) display
H_16:       CLR A
   MOVC A, @A+DPTR
   JZ B_16
   ACALL TRANS
   INC DPTR
   SJMP H_16
B_16:       ACALL RECEIVE
   CJNE A, #6EH, YES1         ;if 'n(6E)' go to YES1
   MOV TEMP14, #01H;SET THE ALARM FLAG
   ACALL TRANS
   MOV A, #0DH
   ACALL TRANS
   LJMP START

YES1:       CJNE A, #79H, B_16 ;if 'y(79)' go to B_16
   ACALL TRANS

   ACALL CONFIGURE18B20
   MOV A, #0CCH               ; 'Skip ROM' Command
   ACALL WRITE18B20
   MOV A, #0BEH               ;'Read Scratchpad' command
   ACALL WRITE18B20

   ACALL READ18B20
   MOV TEMP1, A
   ACALL READ18B20
   MOV TEMP2, A
   ACALL READ18B20
   MOV TEMP11, A              ;TEMP11 CONTAINS THE
UPPER LIMIT
   ACALL READ18B20
   MOV TEMP12, A              ;TEMP12 CONTAINS THE
LOWER LIMIT
   ACALL CONFIGURE18B20
MOV A, #0DH                          ; for 'ENTER'
ACALL TRANS
MOV A, #0DH                          ; for 'ENTER'
ACALL TRANS
MOV DPTR, #MYDATA17                  ; Previously set
Upper Limit is =
H_17:       CLR A
   MOVC A, @A+DPTR
   JZ B_17
   ACALL TRANS
   INC DPTR
   SJMP H_17
B_17:       MOV A, TEMP11
   MOV B, #100
   DIV AB
   ADD A, #30H
   ACALL TRANS
   MOV A, B
   MOV B, #10
   DIV AB
   ADD A, #30H
   ACALL TRANS
   MOV A, B
   ADD A, #30H
   ACALL TRANS
   ACALL DEGREEC
MOV A, #0DH                          ; for 'ENTER'
ACALL TRANS
MOV DPTR, #MYDATA18                  ; Previously set
Lower Limit is =
H_18:       CLR A
   MOVC A, @A+DPTR
   JZ B_18
   ACALL TRANS
   INC DPTR
   SJMP H_18
B_18:       MOV A, TEMP12
   MOV B, #100
   DIV AB
   ADD A, #30H
   ACALL TRANS
   MOV A, B
   MOV B, #10
   DIV AB
   ADD A, #30H
   ACALL TRANS
   MOV A, B
   ADD A, #30H
   ACALL TRANS
   ACALL DEGREEC
MOV A, #0DH                          ; for 'ENTER'
ACALL TRANS
   LJMP START

YES:        CJNE A, #79H, YES4;'79H = y'
   ACALL TRANS
   SJMP OK2
YES4: LJMP B_11

OK2:MOV A, #0DH                      ; for 'ENTER'
ACALL TRANS
MOV A, #0DH                          ; for 'ENTER'
ACALL TRANS
MOV DPTR, #MYDATA12                  ; set the upper
temperature limit display
H_12:       CLR A
   MOVC A, @A+DPTR
   JZ B_12
   ACALL TRANS
   INC DPTR
   SJMP H_12
B_12:       ACALL RECEIVE
   MOV TEMPHIGH, A
   SUBB A, #32H
   JNC B_12
   CLR C                      ; CLEAR THE CARRY
FLAG
   MOV A, TEMPHIGH
   ACALL TRANS
   SUBB A, #30H
   MOV B, #64H
   MUL AB
   MOV TEMPHIGH, A
   ACALL RECEIVE
   ACALL TRANS
   SUBB A, #30H
   MOV B, #0AH
   MUL AB
   ADD A, TEMPHIGH
   MOV TEMPHIGH, A
   ACALL RECEIVE
   ACALL TRANS
   SUBB A, #30H
```

```
    ADD A, TEMPHIGH
    MOV TEMPHIGH, A
    acall degreec
    MOV A, TEMPHIGH
    SUBB A, #7EH                ; 7E=126 exceed upper limit
    JC B_14
MOV A, #0DH                     ; for 'ENTER'
ACALL TRANS
MOV A, #0DH                     ; for 'ENTER'
ACALL TRANS
MOV DPTR, #MYDATA14             ; upper limit
exceeds dispaly
H_14:       CLR A
    MOVC A, @A+DPTR
    JZ OK2
    ACALL TRANS
    INC DPTR
    SJMP H_14
B_14:       CLR C
MOV A, #0DH                     ; for 'ENTER'
ACALL TRANS
MOV DPTR, #MYDATA13             ; set the lower
temperature limit display
H_13:       CLR A
    MOVC A, @A+DPTR
    JZ B_13
    ACALL TRANS
    INC DPTR
    SJMP H_13
B_13:       ACALL RECEIVE
    MOV TEMPLOW, A
    SUBB A, #32H
    JNC B_13
    CLR C                       ; CLEAR THE CARRY
FLAG
    MOV A, TEMPLOW
    ACALL TRANS
    SUBB A, #30H
    MOV B, #64H
    MUL AB
    MOV TEMPLOW, A
    ACALL RECEIVE
    ACALL TRANS
    SUBB A, #30H
    MOV B, #0AH
    MUL AB
    ADD A, TEMPLOW
    MOV TEMPLOW, A
    ACALL RECEIVE
    ACALL TRANS
    SUBB A, #30H
    ADD A, TEMPLOW
    MOV TEMPLOW, A
    acall degreec
;MOV A, #0DH                    ; for 'ENTER'
;ACALL TRANS
    MOV A, TEMPLOW
    SUBB A, TEMPHIGH
    JC B_15
MOV A, #0DH                     ; for 'ENTER'
ACALL TRANS
MOV A, #0DH                     ; for 'ENTER'
ACALL TRANS
MOV DPTR, #MYDATA15             ; Lower limit
should be less than the Upper limit dispaly
H_15:       CLR A
    MOVC A, @A+DPTR
    JZ B_14
    ACALL TRANS
    INC DPTR
    SJMP H_15
B_15:       CLR C
MOV A, #0DH                     ; for 'ENTER'
ACALL TRANS
;---------------------------------------------------------------------------
---
OK3:        ACALL CONFIGURE18B20
    MOV A, #0CCH                ; 'Skip ROM' Command
    ACALL WRITE18B20
    MOV A, #4EH
    ACALL WRITE18B20
    MOV A, TEMPHIGH
    ACALL WRITE18B20
    MOV A, TEMPLOW
    ACALL WRITE18B20
    MOV A, #7FH                 ; control word for 12 bit
resolution
    ACALL WRITE18B20
    ACALL CONFIGURE18B20
    MOV A, #0CCH
    ACALL WRITE18B20
    MOV A, #48H                 ;48H=COPY
SCRATCHPAD TO EPROM
    ACALL WRITE18B20

START:      ACALL CONFIGURE18B20    ;Reset DS18B20
    MOV A, #0CCH                ;'Skip ROM' command
    ACALL WRITE18B20
    MOV A, #44H                 ;'Convert Temperature'
command
    ACALL WRITE18B20
    SETB DQ
    ACALL DELAY800MS

    ACALL CONFIGURE18B20
    MOV A, #0CCH                ; 'Skip ROM' Command
    ACALL WRITE18B20
    MOV A, #0BEH                ;'Read Scratchpad' command
    ACALL WRITE18B20

    ACALL READ18B20
    MOV TEMP1, A
    ACALL READ18B20
    MOV TEMP2, A
    ACALL READ18B20
    ADD A, #01H                 ;upper range including the
limitinig temp
    MOV TEMP11, A
    ACALL READ18B20
    SUBB A, #01H                ;lower range including the
limiting temp
    MOV TEMP12, A

    MOV A, TEMP14               ;TEMP14 CONTAIN
ALLARM FLAG
    CJNE A, #00H, EXIT2         ;CHECKING THE ALARM
FLAG 01=SET,00=NO ALARM

    MOV A, TEMP1
    ANL A, #0F0H
    SWAP A
    MOV TEMP10, A
    MOV A, TEMP2
    ANL A, #0FH
    SWAP A
    ADD A, TEMP10
    MOV TEMP10, A

    SUBB A, TEMP11
    JC EXIT
    CLR P3.7
    setb p2.2
    SJMP EXIT2
EXIT:       MOV A, TEMP10
    SUBB A, TEMP12
    JNC EXIT1
    CLR P3.7
    setb p2.0
    SJMP EXIT2
EXIT1:      SETB P3.7
    clr p2.0
    clr p2.2
EXIT2:
;****************************************************
****************************************************
;temperature conversion done, now convert the 2 byte
temperature into equivalent decimal
;****************************************************
****************************************************
    MOV A, TEMP1
    MOV TEMP3, A
```

```
   ANL A, #0Fh                  ;TAKING THE FRACTIONAL PART ONLY
   MOV temp1, A                 ;Temp1 contain the fractional part of the temperature

   MOV A, temp3
   ANL A, #0F0H
   SWAP A
   MOV temp3, A                 ;Temp3 contain the LOWER nibble of the temperature
   MOV A, temp2
   ANL A, #0FH
   SWAP A
   ADD A, TEMP3                 ;A contain the total temperature byte (8 bit)
   MOV temp3, A                 ;Temp3 contain the total temperature byte (8 bit)

   MOV B, #10
   DIV AB
   MOV temp4, B                 ;temp4 contain the 0th digit
   MOV B, #10
   DIV AB
   MOV temp5, B                 ;temp5 contain the 10th digit
   MOV temp6, A                 ;temp6 contain the 100th digit

MOV A, #0C2H
ACALL LCD_CMD
   mov a, temp6
   JZ mp2
   acall disp_num
   SJMP MP4
mp2:        MOV A, #' '
   ACALL LCD_DATA
   mov a, temp5
   JZ mp3
MP4:        MOV A, TEMP5
   acall disp_num
   SJMP MP5
mp3:        MOV A, #' '
   ACALL LCD_DATA
MP5:        mov a, temp4
   acall disp_num
;****************************************************************************************************
;This section display the fractional part
;****************************************************************************************************
   mov a, #'.'          ;for displaying 'degree C'
   acall LCD_DATA
   MOV A, TEMP1
   CJNE A, #00, L1
   ACALL ZERO
   LJMP L16
L1:         CJNE A, #01, L2
   ACALL ONE
   LJMP L16
L2:         CJNE A, #02, L3
   ACALL TWO
   LJMP L16
L3:         CJNE A, #03, L4
   ACALL THREE
   LJMP L16
L4:         CJNE A, #04, L5
   ACALL FOUR
   LJMP L16
L5:         CJNE A, #05, L6
   ACALL FIVE
   LJMP L16
L6:         CJNE A, #06, L7
   ACALL SIX
   LJMP L16
L7:         CJNE A, #07, L8
   ACALL SEVEN
   LJMP L16
L8:         CJNE A, #08, L9
   ACALL EIGHT
   LJMP L16
L9:         CJNE A, #09, L10
   ACALL NINE
   LJMP L16
L10:        CJNE A, #10, L11
   ACALL TEN
   LJMP L16
L11:        CJNE A, #11, L12
   ACALL ELEVEN
   LJMP L16
L12:        CJNE A, #12, L13
   ACALL TWELVE
   LJMP L16
L13:        CJNE A, #13, L14
   ACALL THIRTEEN
   LJMP L16
L14:        CJNE A, #14, L15
   ACALL FORTEEN
L15:        CJNE A, #15, L16
   ACALL FIFTEEN

L16:        mov a, #' '
   acall LCD_DATA
   mov a, #0DFH                 ;for 'degree' display
   acall LCD_DATA
   mov a, #'C'
   acall LCD_DATA
   ACALL DELAY800MS
;****************************************************************************************************
;   Serial Display of the temperature
;****************************************************************************************************
   MOV A, temp1
   SUBB A, temp8
   CJNE A, #00H, OK1
   LJMP START
OK1:        MOV temp8, temp1
   MOV DPTR, #MYDATA10
H_10:       CLR A
   MOVC A, @A+DPTR
   JZ B_10
   ACALL TRANS
   INC DPTR
   SJMP H_10
b_10:
   MOV A, TEMP6
   JZ SKIP
   ADD A, #30H
   ACALL TRANS
SKIP:       MOV A, TEMP5
   ADD A, #30H
   ACALL TRANS
   MOV A, TEMP4
   ADD A, #30H
   ACALL TRANS
   MOV A, #'.'
   ACALL TRANS

   MOV A, TEMP1
   CJNE A, #00, L101
   ACALL ZERO1
   LJMP L166
L101:       CJNE A, #01, L22
   ACALL ONE1
   LJMP L166
L22:        CJNE A, #02, L33
   ACALL TWO1
   LJMP L166
L33:        CJNE A, #03, L44
   ACALL THREE1
   LJMP L166
L44:        CJNE A, #04, L55
   ACALL FOUR1
   LJMP L166
L55:        CJNE A, #05, L66
   ACALL FIVE1
   LJMP L166
L66:        CJNE A, #06, L77
   ACALL SIX1
   LJMP L166
L77:        CJNE A, #07, L88
   ACALL SEVEN1
   LJMP L166
```

```
L88:        CJNE A, #08, L99
   ACALL EIGHT1
   LJMP L166
L99:        CJNE A, #09, L100
   ACALL NINE1
   LJMP L166
L100:       CJNE A, #10, L111
   ACALL TEN1
   LJMP L166
L111:       CJNE A, #11, L122
   ACALL ELEVEN1
   LJMP L166
L122:       CJNE A, #12, L133
   ACALL TWELVE1
   LJMP L166
L133:       CJNE A, #13, L144
   ACALL THIRTEEN1
   LJMP L166
L144:       CJNE A, #14, L155
   ACALL FORTEEN1
L155:       CJNE A, #15, L166
   ACALL FIFTEEN1

L166:       mov a, #' '
   acall TRANS
   mov a, #0F8H             ;for 'degree' display
   acall TRANS
   mov a, #'C'
   acall TRANS
   ACALL DELAY800MS
LJMP START
;***************************************************
***************************************************
zero1:      mov a, #'0'      ;for displaying '0000'
   acall TRANS
   mov a, #'0'
   acall TRANS
   mov a, #'0'
   acall TRANS
   mov a, #'0'
   acall TRANS
RET
ONE1:       mov a,#'0'       ;for displaying '0625'
   acall TRANS
   mov a,#'6'
   acall TRANS
   mov a,#'2'
   acall TRANS
   mov a,#'5'
   acall TRANS
RET
TWO1:       mov a,#'1'       ;for displaying '1250'
   acall TRANS
   mov a,#'2'
   acall TRANS
   mov a,#'5'
   acall TRANS
   mov a,#'0'
   acall TRANS
RET
THREE1:     mov a,#'1'       ;for displaying '1875'
   acall TRANS
   mov a,#'8'
   acall TRANS
   mov a,#'7'
   acall TRANS
   mov a,#'5'
   acall TRANS
RET
FOUR1:      mov a,#'2'       ;for displaying '2500'
   acall TRANS
   mov a,#'5'
   acall TRANS
   mov a,#'0'
   acall TRANS
   mov a,#'0'
   acall TRANS
RET
FIVE1:      mov a,#'3'       ;for displaying '3125'
   acall TRANS
   mov a,#'1'
   acall TRANS
   mov a,#'2'
   acall TRANS
   mov a,#'5'
   acall TRANS
RET
SIX1:       mov a,#'3'       ;for displaying '3750'
   acall TRANS
   mov a,#'7'
   acall TRANS
   mov a,#'5'
   acall TRANS
   mov a,#'0'
   acall TRANS
RET
SEVEN1:     mov a,#'4'       ;for displaying '4375'
   acall TRANS
   mov a,#'3'
   acall TRANS
   mov a,#'7'
   acall TRANS
   mov a,#'5'
   acall TRANS
RET
EIGHT1:     mov a,#'5'       ;for displaying '5000'
   acall TRANS
   mov a,#'0'
   acall TRANS
   mov a,#'0'
   acall TRANS
   mov a,#'0'
   acall TRANS
RET
NINE1:      mov a,#'5'       ;for displaying '5625'
   acall TRANS
   mov a,#'6'
   acall TRANS
   mov a,#'2'
   acall TRANS
   mov a,#'5'
   acall TRANS
RET
TEN1:       mov a,#'6'       ;for displaying '6250'
   acall TRANS
   mov a,#'2'
   acall TRANS
   mov a,#'5'
   acall TRANS
   mov a,#'0'
   acall TRANS
RET
ELEVEN1:    mov a,#'6'       ;for displaying '6875'
   acall TRANS
   mov a,#'8'
   acall TRANS
   mov a,#'7'
   acall TRANS
   mov a,#'5'
   acall TRANS
RET
TWELVE1:    mov a,#'7'       ;for displaying '7500'
   acall TRANS
   mov a,#'5'
   acall TRANS
   mov a,#'0'
   acall TRANS
   mov a,#'0'
   acall TRANS
RET
THIRTEEN1:mov a,#'8'         ;for displaying '8125'
   acall TRANS
   mov a,#'1'
   acall TRANS
   mov a,#'2'
   acall TRANS
   mov a,#'5'
   acall TRANS
RET
FORTEEN1:mov a,#'8'          ;for displaying '8750'
   acall TRANS
   mov a,#'7'
```

```
    acall TRANS
    mov a,#'5'
    acall TRANS
    mov a,#'0'
    acall TRANS
RET
FIFTEEN1:mov a,#'9'         ;for displaying '9375'
    acall TRANS
    mov a,#'3'
    acall TRANS
    mov a,#'7'
    acall TRANS
    mov a,#'5'
    acall TRANS
RET
;**************************************************
**************************************************
zero:       mov a, #'0'     ;for displaying '0000'
    acall LCD_DATA
    mov a, #'0'
    acall LCD_DATA
    mov a, #'0'
    acall LCD_DATA
    mov a, #'0'
    acall LCD_DATA
RET
ONE:        mov a,#'0'      ;for displaying '0625'
    acall LCD_DATA
    mov a,#'6'
    acall LCD_DATA
    mov a,#'2'
    acall LCD_DATA
    mov a,#'5'
    acall LCD_DATA
RET
TWO:        mov a,#'1'      ;for displaying '1250'
    acall LCD_DATA
    mov a,#'2'
    acall LCD_DATA
    mov a,#'5'
    acall LCD_DATA
    mov a,#'0'
    acall LCD_DATA
RET
THREE:      mov a,#'1'      ;for displaying '1875'
    acall LCD_DATA
    mov a,#'8'
    acall LCD_DATA
    mov a,#'7'
    acall LCD_DATA
    mov a,#'5'
    acall LCD_DATA
RET
FOUR:       mov a,#'2'      ;for displaying '2500'
    acall LCD_DATA
    mov a,#'5'
    acall LCD_DATA
    mov a,#'0'
    acall LCD_DATA
    mov a,#'0'
    acall LCD_DATA
RET
FIVE:       mov a,#'3'      ;for displaying '3125'
    acall LCD_DATA
    mov a,#'1'
    acall LCD_DATA
    mov a,#'2'
    acall LCD_DATA
    mov a,#'5'
    acall LCD_DATA
RET
SIX:        mov a,#'3'      ;for displaying '3750'
    acall LCD_DATA
    mov a,#'7'
    acall LCD_DATA
    mov a,#'5'
    acall LCD_DATA
    mov a,#'0'
    acall LCD_DATA
RET
SEVEN:      mov a,#'4'      ;for displaying '4375'
    acall LCD_DATA
    mov a,#'3'
    acall LCD_DATA
    mov a,#'7'
    acall LCD_DATA
    mov a,#'5'
    acall LCD_DATA
RET
EIGHT:      mov a,#'5'      ;for displaying '5000'
    acall LCD_DATA
    mov a,#'0'
    acall LCD_DATA
    mov a,#'0'
    acall LCD_DATA
    mov a,#'0'
    acall LCD_DATA
RET
NINE:       mov a,#'5'      ;for displaying '5625'
    acall LCD_DATA
    mov a,#'6'
    acall LCD_DATA
    mov a,#'2'
    acall LCD_DATA
    mov a,#'5'
    acall LCD_DATA
RET
TEN:        mov a,#'6'      ;for displaying '6250'
    acall LCD_DATA
    mov a,#'2'
    acall LCD_DATA
    mov a,#'5'
    acall LCD_DATA
    mov a,#'0'
    acall LCD_DATA
RET
ELEVEN:     mov a,#'6'      ;for displaying '6875'
    acall LCD_DATA
    mov a,#'8'
    acall LCD_DATA
    mov a,#'7'
    acall LCD_DATA
    mov a,#'5'
    acall LCD_DATA
RET
TWELVE:     mov a,#'7'      ;for displaying '7500'
    acall LCD_DATA
    mov a,#'5'
    acall LCD_DATA
    mov a,#'0'
    acall LCD_DATA
    mov a,#'0'
    acall LCD_DATA
RET
THIRTEEN:mov a,#'8'         ;for displaying '8125'
    acall LCD_DATA
    mov a,#'1'
    acall LCD_DATA
    mov a,#'2'
    acall LCD_DATA
    mov a,#'5'
    acall LCD_DATA
RET
FORTEEN:mov a,#'8'          ;for displaying '8750'
    acall LCD_DATA
    mov a,#'7'
    acall LCD_DATA
    mov a,#'5'
    acall LCD_DATA
    mov a,#'0'
    acall LCD_DATA
RET
FIFTEEN:mov a,#'9'          ;for displaying '9375'
    acall LCD_DATA
    mov a,#'3'
    acall LCD_DATA
    mov a,#'7'
    acall LCD_DATA
    mov a,#'5'
    acall LCD_DATA
RET
```

```
;*****************************************************
*****************************************************
;REST of DS18B20
;*****************************************************
*****************************************************
CONFIGURE18B20:
   CLR DQ
   ACALL DELAY480US
   SETB DQ                          ;DQ=P1.0=1 (make it as an
input port)
   ACALL DELAY480US
RET
;*****************************************************
*****************************************************
;READ BYTE of DS18B20
;*****************************************************
*****************************************************
READ18B20:
   MOV R0, #08H         ; Set Counter for 8 bits
NEXTBITREAD:
   CLR DQ               ; Start clock cycle
   NOP
   SETB DQ
   ACALL DELAY10US
   MOV C, DQ            ; Move incoming bit to DQ
   RRC A                ; Rotate A Right through Carry Bit (C
goes to Highest Bit of A)
   ACALL DELAY55US
   SETB DQ
   DJNZ R0, NEXTBITREAD
RET
;*****************************************************
*****************************************************
;WRITE BYTE of DS18B20
;*****************************************************
*****************************************************
WRITE18B20:
   MOV R0, #08H         ; Set Counter for 8 bits
NEXTBITWRITE:
   RRC A                ; Rotate A Right into Carry Bit (Lowest
Bit in A goes to C)
   JC WRITE1
   CLR DQ
   ACALL DELAY10US
   ACALL DELAY10US
   ACALL DELAY10US
   MOV DQ, C            ; Move outgoing bit to DQ
   ACALL DELAY70US
   SETB DQ
   DJNZ R0, NEXTBITWRITE
   SJMP LAST
WRITE1:     CLR DQ
   nop
   SETB DQ              ; Rising Edge of Clock makes one clock
cycle
   ACALL DELAY24US
   MOV DQ, C
   ACALL DELAY10US
   ACALL DELAY10US
   ACALL DELAY10US
   ACALL DELAY10US
   ACALL DELAY10US
   ACALL DELAY10US
   SETB DQ
   DJNZ R0, NEXTBITWRITE
LAST:       RET
;---------------------------------------------------------------------------
------
TRANS:
   MOV SBUF, A
   JNB TI, $
   CLR TI
RET

RECEIVE:
   JNB RI,$
   clr RI
   mov A,SBUF
RET
;---------------------------------------------------------------------------
-----
```

```
;   Different DELAY subroutine
;---------------------------------------------------------------------------
---
DELAY480US:                                     ;Delay for 470us
   MOV TMOD, #21H               ;FE2A      PROVIDE      A
DELAY OF 470us (11.0592 MHz)
   MOV TL0, #4FH
   MOV TH0, #0FEH
   SETB TR0
   JNB TF0, $
   CLR TR0
   CLR TF0
RET
;---------------------------------------------------------------------------
---
DELAY70US:                                      ;Delay for 55us
   MOV TMOD, #21H               ;FFC9 PROVIDE A DELAY
OF 55us (11.0592 MHz)
   MOV TL0, #0CEH
   MOV TH0, #0FFH
   SETB TR0
   JNB TF0, $
   CLR TR0
   CLR TF0
RET
;---------------------------------------------------------------------------
---
DELAY6US:                       ;Delay for 6us (DOUBT)
   NOP
RET
;---------------------------------------------------------------------------
---
DELAY60US:                                      ;Delay for 48us
   MOV TMOD, #21H               ;FFD0 PROVIDE A DELAY
OF 48us (11.0592 MHz)
   MOV TL0, #0D4H
   MOV TH0, #0FFH
   SETB TR0
   JNB TF0, $
   CLR TR0
   CLR TF0
RET
;---------------------------------------------------------------------------
---
DELAY10US:                                      ;Delay for 10us
   NOP
   NOP
   NOP
   NOP
   NOP
RET
;---------------------------------------------------------------------------
---
DELAY7US:                       ;Delay for 7US
   NOP
   NOP
RET
;---------------------------------------------------------------------------
---
DELAY55US:                                      ;Delay for 42us
   MOV TMOD, #21H               ;FFD3 PROVIDE A DELAY
OF 42us (11.0592 MHz)
   MOV TL0, #0DAH
   MOV TH0, #0FFH
   SETB TR0
   JNB TF0, $
   CLR TR0
   CLR TF0
RET
;---------------------------------------------------------------------------
---
DELAY24US:                                      ;Delay for 8us
   MOV TMOD, #21H               ;FFF8 PROVIDE A DELAY
OF 8 us (11.0592 MHz)
   MOV TL0, #0F9H
   MOV TH0, #0FFH
   SETB TR0
   JNB TF0, $
   CLR TR0
   CLR TF0
RET
```

```
;------------------------------------------------------------------------------
---
DELAY410US:                                  ;Delay for 400us
    MOV TMOD, #21H              ;FE70 PROVIDE A DELAY
OF 400us (11.0592 MHz)
    MOV TL0, #90H
    MOV TH0, #0FEH
    SETB TR0
    JNB TF0, $
    CLR TR0
    CLR TF0
RET
;------------------------------------------------------------------------------
---
DELAY10MS:                                   ;Delay for 10Ms
    MOV R7, #01H

    MOV TMOD, #21H              ;0BDC   PROVIDE   A
DELAY OF 750Ms (11.0592 MHz)
LOOP2:      MOV TL0, #00H
    MOV TH0, #0DCH
    SETB TR0
    JNB TF0, $
    CLR TR0
    CLR TF0
    DJNZ R1, LOOP2
RET
;------------------------------------------------------------------------------
---
disp_num:
    mov DPTR,#hex_table
    movc a,@a+dptr
    acall LCD_DATA
ret
;------------------------------------------------------------------------------
---
lcd_reset:                      ;LCD reset sequence
    mov lcd_port, #0FFH
    mov delay,#20       ;20mS delay
    acall delayms
    mov lcd_port, #83H      ;Data = 30H, EN = 1, First Init
    mov lcd_port, #03H      ;Data = 30H, EN = 0
    mov delay,#10       ;Delay 10mS
    acall delayms
    mov lcd_port, #83H      ;Second Init, Data = 30H, EN = 1
    mov lcd_port, #03H      ;Data = 30H, EN = 0
    mov delay,#1        ;Delay 5mS
    acall delayms
    mov lcd_port, #83H      ;Third Init
    mov lcd_port, #03H
    mov delay,#1        ;Delay 5mS
    acall delayms
    mov lcd_port, #82H      ;Select Data width (20H for 4bit)
    mov lcd_port, #02H      ;Data = 20H, EN = 0
    mov delay,#1        ;Delay 5mS
    acall delayms
ret
;------------------------------------------------------------------------------
---
lcd_init:
    acall lcd_reset     ;Call LCD Reset sequence
    mov a,#28H          ;4-bit, 2 line, 5x7 dots
    acall lcd_cmd       ;Call LCD command
    mov a,#0CH          ;LCD_DATAlay ON cursor OFF
    acall lcd_cmd       ;Call LCD command
    mov a,#06H          ;Set entry mode (Auto increment)
    acall lcd_cmd       ;Call LCD command
    mov a,#80H          ;Bring cursor to line 1
    acall lcd_cmd       ;Call LCD command
ret
;------------------------------------------------------------------------------
---
lcd_cmd:                ;LCD command Routine
    mov temp,a          ;Save a copy of command to temp
    swap a              ;Swap to use higher nibble
    anl a,#0FH          ;Mask the first four bits
    add a,#80H          ;Enable = 1, RS = 0, RW = 0
    mov lcd_port,a      ;Move it to lcd port
    mov delay,#5        ;5mS delay
    acall delayms
    CLR EN
    mov a,temp          ;Reload the command from temp
    anl a,#0FH          ;Mask first four bits
    add a,#80H          ;Enable = 1
    mov lcd_port,a      ;Move to port
    mov delay,#5        ;5mS delay
    acall delayms
    CLR EN
ret
;------------------------------------------------------------------------------
---
LCD_DATA:               ;LCD data Routine
    mov temp,a          ;Keep copy of data in temp
    swap a              ;We need higher nibble
    anl a,#0FH          ;Mask first four bits
    add a,#0A0H         ;Enable = 1, RS = 1, RW = 0
    mov lcd_port,a      ;Move to lcd port
    mov delay,#5        ;5mS delay
    acall delayms
    clr en              ;Enable = 0

    mov a,temp          ;Reload the data from temp
    anl a,#0FH          ;we need lower nibble now
    add a,#0A0H         ;Enable = 1, RS = 1, RW = 0
    mov lcd_port,a      ;Move to lcd port
    mov delay,#5        ;5mS delay
    acall delayms
    clr en              ;Enable = 0
ret
;------------------------------------------------------------------------------
---
delayms:
    mov var1,#255                   ;changed from 230
d:
    nop
    nop
    djnz var1,d
    djnz delay,delayms
ret
;------------------------------------------------------------------------------
---
DEGREEC:
    mov a, #' '
    acall TRANS
    mov a, #0F8H                    ;for 'degree' display
    acall TRANS
    mov a, #'C'
    acall TRANS
;   ACALL DELAY800MS
RET
;------------------------------------------------------------------------------
----
DELAY800MS:                                  ;Delay for 750Ms
    MOV R7, #0BH

    MOV TMOD, #21H              ;0BDC   PROVIDE   A
DELAY OF 750Ms (11.0592 MHz)
LOOP1:      MOV TL0, #8CH
    MOV TH0, #0AH
    SETB TR0
    JNB TF0, $
    CLR TR0
    CLR TF0
    DJNZ R7, LOOP1
RET

;------------------------------------------------------------------------------
----
hex_table:    DB '0','1','2','3','4','5','6','7','8','9'
MYDATA1:  DB 'DS18B20 Based', 0
MYDATA2:  DB 'Thermometer', 0
MYDATA5:  DB 'Developed by', 0
MYDATA6:  DB 'by  A P S S', 0
MYDATA7:  DB 'The Temperature', 0
MYDATA8:  DB '   Wireless Temperature Sensor using HC-05
Bluetooth Module', 0
MYDATA9:  DB '         Design by A P S S',0DH, '|----|----|----|-
---|----|----|----|----|----|----|----|----|',0DH, 0
MYDATA10:           DB 0DH,'The Temperature = ', 0
MYDATA11:           DB 'Do you want to set the Temperature
limit? (y/n) - ',0
```

```
MYDATA12:     DB 'Set the upper limit (0-125 degree centigrade) (3 digits) - ',0
MYDATA13:     DB 'Set the lower limit (0-125 degree centigrade) (3 digits) - ',0
MYDATA14:     DB 'Upper limit exceeds',0
MYDATA15:     DB 'Lower limit should be less than the Upper limit',0DH, 0
MYDATA16:     DB 'Enable the alarm (y/n) - ', 0
MYDATA17:     DB 'Previously set Upper Limit is = ', 0
MYDATA18:     DB 'Previously set Lower Limit is = ', 0
;----------------------------------------------------------------------------------
END
```

Appendix C

(Data sheets)

Features

- **Compatible with MCS-51™ Products**
- **2K Bytes of Reprogrammable Flash Memory**
  - **– Endurance: 1,000 Write/Erase Cycles**
- **2.7V to 6V Operating Range**
- **Fully Static Operation: 0 Hz to 24 MHz**
- **Two-level Program Memory Lock**
- **128 x 8-bit Internal RAM**
- **15 Programmable I/O Lines**
- **Two 16-bit Timer/Counters**
- **Six Interrupt Sources**
- **Programmable Serial UART Channel**
- **Direct LED Drive Outputs**
- **On-chip Analog Comparator**
- **Low-power Idle and Power-down Modes**

8-bit Microcontroller with 2K Bytes Flash

AT89C2051

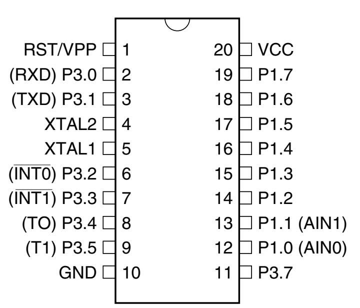
Description

The AT89C2051 is a low-voltage, high-performance CMOS 8-bit microcomputer with 2K bytes of Flash programmable and erasable read only memory (PEROM). The device is manufactured using Atmel's high-density nonvolatile memory technology and is compatible with the industry-standard MCS-51 instruction set. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C2051 is a powerful microcomputer which provides a highly-flexible and cost-effective solution to many embedded control applications.

The AT89C2051 provides the following standard features: 2K bytes of Flash, 128 bytes of RAM, 15 I/O lines, two 16-bit timer/counters, a five vector two-level interrupt architecture, a full duplex serial port, a precision analog comparator, on-chip oscillator and clock circuitry. In addition, the AT89C2051 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port and interrupt system to continue functioning. The power-down mode saves the RAM contents but freezes the oscillator disabling all other chip functions until the next hardware reset.

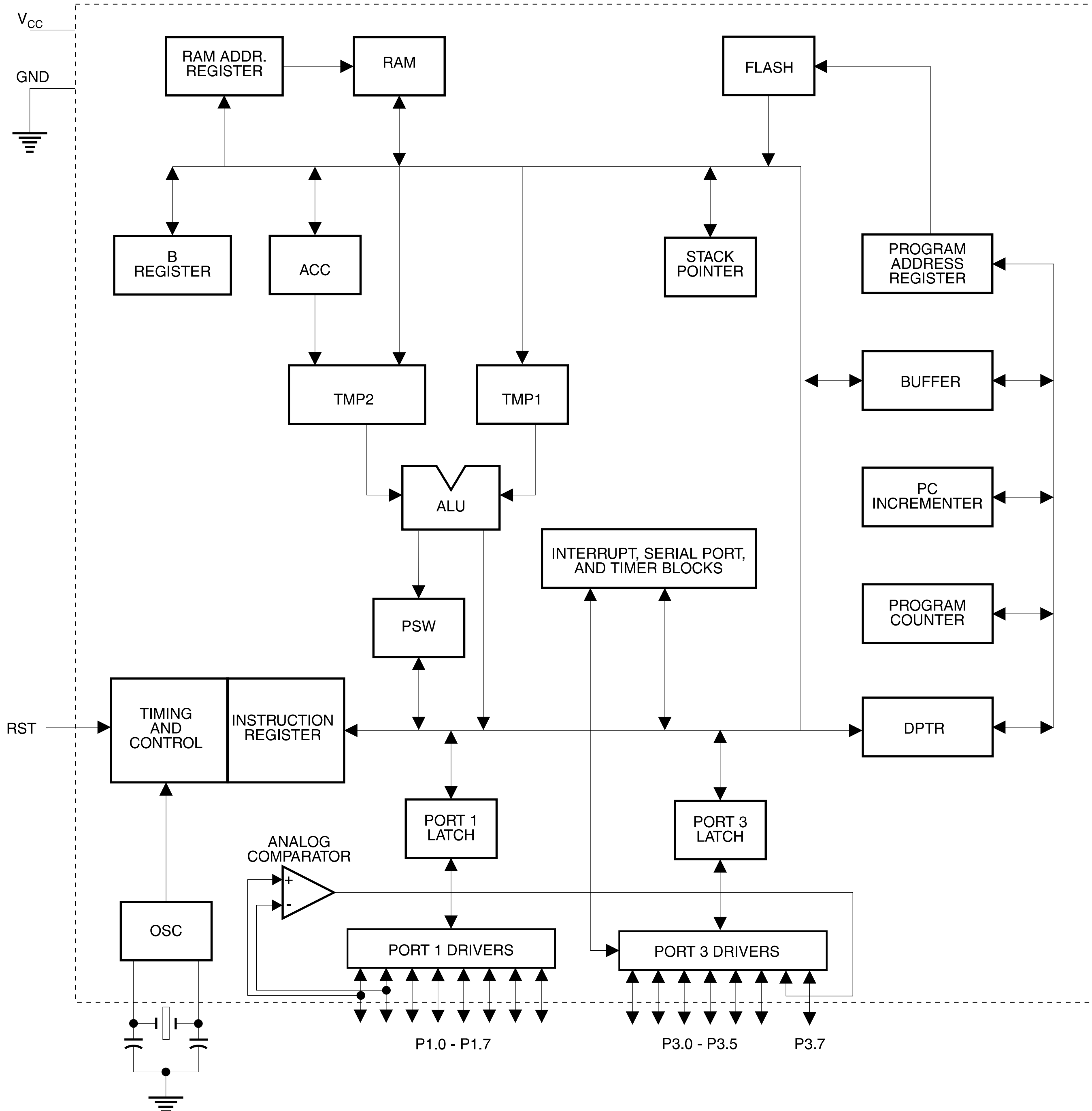
Pin Configuration

PDIP/SOIC



Rev. 0368E–02/00

Block Diagram



Pin Description

VCC
Supply voltage.

GND
Ground.

Port 1
Port 1 is an 8-bit bi-irectional I/O port. Port pins P1.2 to P1.7 provide internal pullups. P1.0 and P1.1 require external pullups. P1.0 and P1.1 also serve as the positive input (AIN0) and the negative input (AIN1), respectively, of the on-chip precision analog comparator. The Port 1 output buffers can sink 20 mA and can drive LED displays directly. When 1s are written to Port 1 pins, they can be used as inputs. When pins P1.2 to P1.7 are used as inputs and are externally pulled low, they will source current (I_{IL}) because of the internal pullups.

Port 1 also receives code data during Flash programming and verification.

Port 3
Port 3 pins P3.0 to P3.5, P3.7 are seven bi-irectional I/O pins with internal pullups. P3.6 is hard-wired as an input to the output of the on-chip comparator and is not accessible as a general purpose I/O pin. The Port 3 output buffers can sink 20 mA. When 1s are written to Port 3 pins they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (I_{IL}) because of the pullups.

Port 3 also serves the functions of various special features of the AT89C2051 as listed below:

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)

Port 3 also receives some control signals for Flash programming and verification.

RST
Reset input. All I/O pins are reset to 1s as soon as RST goes high. Holding the RST pin high for two machine cycles while the oscillator is running resets the device.

Each machine cycle takes 12 oscillator or clock cycles.

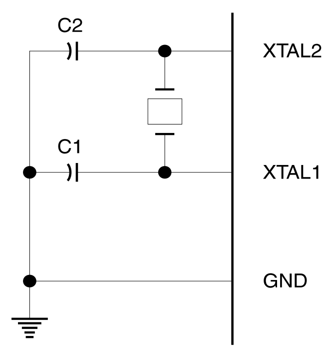
XTAL1
Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

XTAL2
Output from the inverting oscillator amplifier.

Oscillator Characteristics

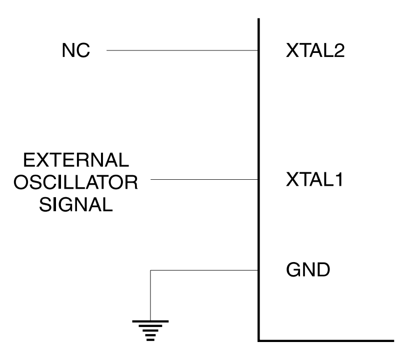
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier which can be configured for use as an on-chip oscillator, as shown in Figure 1. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven as shown in Figure 2. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

Figure 1. Oscillator Connections



Note: C1, C2 = 30 pF ± 10 pF for Crystals
= 40 pF ± 10 pF for Ceramic Resonators

Figure 2. External Clock Drive Configuration



Special Function Registers

A map of the on-chip memory area called the Special Function Register (SFR) space is shown in the table below.

Note that not all of the addresses are occupied, and unoccupied addresses may not be implemented on the chip. Read accesses to these addresses will in general return random data, and write accesses will have an indeterminate effect.

User software should not write 1s to these unlisted locations, since they may be used in future products to invoke new features. In that case, the reset or inactive values of the new bits will always be 0.

Table 1. AT89C2051 SFR Map and Reset Values

0F8H									0FFH
0F0H	B 00000000								0F7H
0E8H									0EFH
0E0H	ACC 00000000								0E7H
0D8H									0DFH
0D0H	PSW 00000000								0D7H
0C8H									0CFH
0C0H									0C7H
0B8H	IP XXX00000								0BFH
0B0H	P3 11111111								0B7H
0A8H	IE 0XX00000								0AFH
0A0H									0A7H
98H	SCON 00000000	SBUF XXXXXXXX							9FH
90H	P1 11111111								97H
88H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000			8FH
80H		SP 00000111	DPL 00000000	DPH 00000000				PCON 0XXX0000	87H

Restrictions on Certain Instructions

The AT89C2051 and is an economical and cost-effective member of Atmel's growing family of microcontrollers. It contains 2K bytes of flash program memory. It is fully compatible with the MCS-51 architecture, and can be programmed using the MCS-51 instruction set. However, there are a few considerations one must keep in mind when utilizing certain instructions to program this device.

All the instructions related to jumping or branching should be restricted such that the destination address falls within the physical program memory space of the device, which is 2K for the AT89C2051. This should be the responsibility of the software programmer. For example, LJMP 7E0H would be a valid instruction for the AT89C2051 (with 2K of memory), whereas LJMP 900H would not.

1. Branching instructions:

LCALL, LJMP, ACALL, AJMP, SJMP, JMP @A+DPTR

These unconditional branching instructions will execute correctly as long as the programmer keeps in mind that the destination branching address must fall within the physical boundaries of the program memory size (locations 00H to 7FFH for the 89C2051). Violating the physical space limits may cause unknown program behavior.

CJNE [...], DJNZ [...], JB, JNB, JC, JNC, JBC, JZ, JNZ With these conditional branching instructions the same rule above applies. Again, violating the memory boundaries may cause erratic execution.

For applications involving interrupts the normal interrupt service routine address locations of the 80C51 family architecture have been preserved.

2. MOVX-related instructions, Data Memory:

The AT89C2051 contains 128 bytes of internal data memory. Thus, in the AT89C2051 the stack depth is limited to 128 bytes, the amount of available RAM. External DATA memory access is not supported in this device, nor is external PROGRAM memory execution. Therefore, no MOVX [...] instructions should be included in the program.

A typical 80C51 assembler will still assemble instructions, even if they are written in violation of the restrictions mentioned above. It is the responsibility of the controller user to know the physical features and limitations of the device being used and adjust the instructions used correspondingly.

Program Memory Lock Bits

On the chip are two lock bits which can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in the table below:

Lock Bit Protection Modes[1]

Program Lock Bits			
	LB1	LB2	Protection Type
1	U	U	No program lock features.
2	P	U	Further programming of the Flash is disabled.
3	P	P	Same as mode 2, also verify is disabled.

Note: 1. The Lock Bits can only be erased with the Chip Erase operation.

Idle Mode

In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special functions registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset.

P1.0 and P1.1 should be set to "0" if no external pullups are used, or set to "1" if external pullups are used.

It should be noted that when idle is terminated by a hardware reset, the device normally resumes program execution, from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hardware inhibits access to internal RAM in this event, but access to the port pins is not inhibited. To eliminate the possibility of an unexpected write to a port pin when Idle is terminated by reset, the instruction following the one that invokes Idle should not be one that writes to a port pin or to external memory.

Power-down Mode

In the power down mode the oscillator is stopped, and the instruction that invokes power down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values until the power down mode is terminated. The only exit from power down is a hardware reset. Reset redefines the SFRs but does not change the on-chip RAM. The reset should not be activated before V_{CC} is restored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize.

P1.0 and P1.1 should be set to "0" if no external pullups are used, or set to "1" if external pullups are used.

Programming The Flash

The AT89C2051 is shipped with the 2K bytes of on-chip PEROM code memory array in the erased state (i.e., contents = FFH) and ready to be programmed. The code memory array is programmed one byte at a time. *Once the array is programmed, to re-program any non-blank byte, the entire memory array needs to be erased electrically.*

Internal Address Counter: The AT89C2051 contains an internal PEROM address counter which is always reset to 000H on the rising edge of RST and is advanced by applying a positive going pulse to pin XTAL1.

Programming Algorithm: To program the AT89C2051, the following sequence is recommended.

1. Power-up sequence:
   Apply power between V_{CC} and GND pins
   Set RST and XTAL1 to GND
2. Set pin RST to "H"
   Set pin P3.2 to "H"
3. Apply the appropriate combination of "H" or "L" logic levels to pins P3.3, P3.4, P3.5, P3.7 to select one of the programming operations shown in the PEROM Programming Modes table.

To Program and Verify the Array:

4. Apply data for Code byte at location 000H to P1.0 to P1.7.
5. Raise RST to 12V to enable programming.
6. Pulse P3.2 once to program a byte in the PEROM array or the lock bits. The byte-write cycle is self-timed and typically takes 1.2 ms.
7. To verify the programmed data, lower RST from 12V to logic "H" level and set pins P3.3 to P3.7 to the appropiate levels. Output data can be read at the port P1 pins.
8. To program a byte at the next address location, pulse XTAL1 pin once to advance the internal address counter. Apply new data to the port P1 pins.
9. Repeat steps 5 through 8, changing data and advancing the address counter for the entire 2K bytes array or until the end of the object file is reached.
10. Power-off sequence:
    set XTAL1 to "L"
    set RST to "L"
    Turn V_{CC} power off

$\overline{\text{Data}}$ Polling: The AT89C2051 features $\overline{\text{Data}}$ Polling to indicate the end of a write cycle. During a write cycle, an attempted read of the last byte written will result in the complement of the written data on P1.7. Once the write cycle has been completed, true data is valid on all outputs, and the next cycle may begin. $\overline{\text{Data}}$ Polling may begin any time after a write cycle has been initiated.

Ready/$\overline{\text{Busy}}$: The Progress of byte programming can also be monitored by the RDY/$\overline{\text{BSY}}$ output signal. Pin P3.1 is pulled low after P3.2 goes High during programming to indicate BUSY. P3.1 is pulled High again when programming is done to indicate READY.

Program Verify: If lock bits LB1 and LB2 have not been programmed code data can be read back via the data lines for verification:

1. Reset the internal address counter to 000H by bringing RST from "L" to "H".
2. Apply the appropriate control signals for Read Code data and read the output data at the port P1 pins.
3. Pulse pin XTAL1 once to advance the internal address counter.
4. Read the next code data byte at the port P1 pins.
5. Repeat steps 3 and 4 until the entire array is read.

The lock bits cannot be verified directly. Verification of the lock bits is achieved by observing that their features are enabled.

Chip Erase: The entire PEROM array (2K bytes) and the two Lock Bits are erased electrically by using the proper combination of control signals and by holding P3.2 low for 10 ms. The code array is written with all "1"s in the Chip Erase operation and must be executed before any non-blank memory byte can be re-programmed.

Reading the Signature Bytes: The signature bytes are read by the same procedure as a normal verification of locations 000H, 001H, and 002H, except that P3.5 and P3.7 must be pulled to a logic low. The values returned are as follows.

(000H) = 1EH indicates manufactured by Atmel
(001H) = 21H indicates 89C2051

Programming Interface

Every code byte in the Flash array can be written and the entire array can be erased by using the appropriate combination of control signals. The write operation cycle is self-timed and once initiated, will automatically time itself to completion.

All major programming vendors offer worldwide support for the Atmel microcontroller series. Please contact your local programming vendor for the appropriate software revision.

DS18B20

Programmable Resolution 1-Wire Digital Thermometer

General Description

The DS18B20 digital thermometer provides 9-bit to 12-bit Celsius temperature measurements and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18B20 communicates over a 1-Wire bus that by definition requires only one data line (and ground) for communication with a central microprocessor. In addition, the DS18B20 can derive power directly from the data line ("parasite power"), eliminating the need for an external power supply.

Each DS18B20 has a unique 64-bit serial code, which allows multiple DS18B20s to function on the same 1-Wire bus. Thus, it is simple to use one microprocessor to control many DS18B20s distributed over a large area. Applications that can benefit from this feature include HVAC environmental controls, temperature monitoring systems inside buildings, equipment, or machinery, and process monitoring and control systems.

Applications

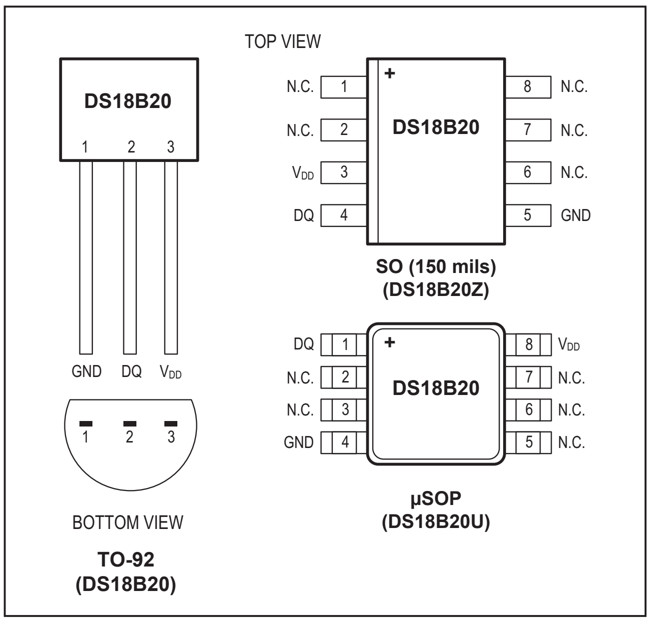
- Thermostatic Controls
- Industrial Systems
- Consumer Products
- Thermometers
- Thermally Sensitive Systems

Ordering Information ***appears at end of data sheet.***

Benefits and Features

- Unique 1-Wire® Interface Requires Only One Port Pin for Communication
- Reduce Component Count with Integrated Temperature Sensor and EEPROM
  - Measures Temperatures from -55°C to +125°C (-67°F to +257°F)
  - ±0.5°C Accuracy from -10°C to +85°C
  - Programmable Resolution from 9 Bits to 12 Bits
  - No External Components Required
- Parasitic Power Mode Requires Only 2 Pins for Operation (DQ and GND)
- Simplifies Distributed Temperature-Sensing Applications with Multidrop Capability
  - Each Device Has a Unique 64-Bit Serial Code Stored in On-Board ROM
- Flexible User-Definable Nonvolatile (NV) Alarm Settings with Alarm Search Command Identifies Devices with Temperatures Outside Programmed Limits
- Available in 8-Pin SO (150 mils), 8-Pin µSOP, and 3-Pin TO-92 Packages

Pin Configurations



Absolute Maximum Ratings

Voltage Range on Any Pin Relative to Ground ....-0.5V to +6.0V
Operating Temperature Range .... -55°C to +125°C
Storage Temperature Range .... -55°C to +125°C
Solder Temperature ....Refer to the IPC/JEDEC J-STD-020 Specification.

These are stress ratings only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

DC Electrical Characteristics

(-55°C to +125°C; V_{DD} = 3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Supply Voltage	V_{DD}	Local power (Note 1)		+3.0		+5.5	V
Pullup Supply Voltage	V_{PU}	Parasite power	(Notes 1, 2)	+3.0		+5.5	V
		Local power		+3.0		V_{DD}	
Thermometer Error	t_{ERR}	-10°C to +85°C	(Note 3)			±0.5	°C
		-30°C to +100°C				±1	
		-55°C to +125°C				±2	
Input Logic-Low	V_{IL}	(Notes 1, 4, 5)		-0.3		+0.8	V
Input Logic-High	V_{IH}	Local power	(Notes 1,6)	+2.2		The lower of 5.5 or V_{DD} + 0.3	V
		Parasite power		+3.0			
Sink Current	I_L	$V_{I/O}$ = 0.4V		4.0			mA
Standby Current	I_{DDS}	(Notes 7, 8)			750	1000	nA
Active Current	I_{DD}	V_{DD} = 5V (Note 9)			1	1.5	mA
DQ Input Current	I_{DQ}	(Note 10)			5		µA
Drift		(Note 11)			±0.2		°C

Note 1: All voltages are referenced to ground.
Note 2: The Pullup Supply Voltage specification assumes that the pullup device is ideal, and therefore the high level of the pullup is equal to V_{PU}. In order to meet the V_{IH} spec of the DS18B20, the actual supply rail for the strong pullup transistor must include margin for the voltage drop across the transistor when it is turned on; thus: $V_{PU\_ACTUAL} = V_{PU\_IDEAL} + V_{TRANSISTOR}$.
Note 3: See typical performance curve in Figure 1. Thermometer Error limits are 3-sigma values.
Note 4: Logic-low voltages are specified at a sink current of 4mA.
Note 5: To guarantee a presence pulse under low voltage parasite power conditions, V_{ILMAX} may have to be reduced to as low as 0.5V.
Note 6: Logic-high voltages are specified at a source current of 1mA.
Note 7: Standby current specified up to +70°C. Standby current typically is 3µA at +125°C.
Note 8: To minimize I_{DDS}, DQ should be within the following ranges: GND ≤ DQ ≤ GND + 0.3V or V_{DD} – 0.3V ≤ DQ ≤ V_{DD}.
Note 9: Active current refers to supply current during active temperature conversions or EEPROM writes.
Note 10: DQ line is high ("high-Z" state).
Note 11: Drift data is based on a 1000-hour stress test at +125°C with V_{DD} = 5.5V.

AC Electrical Characteristics–NV Memory

(-55°C to +125°C; V_{DD} = 3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
NV Write Cycle Time	t_{WR}			2	10	ms
EEPROM Writes	N_{EEWR}	-55°C to +55°C	50k			writes
EEPROM Data Retention	t_{EEDR}	-55°C to +55°C	10			years

AC Electrical Characteristics

(-55°C to +125°C; V_{DD} = 3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Temperature Conversion Time	t_{CONV}	9-bit resolution	(Note 12)			93.75	ms
		10-bit resolution				187.5	
		11-bit resolution				375	
		12-bit resolution				750	
Time to Strong Pullup On	t_{SPON}	Start convert T command issued				10	µs
Time Slot	t_{SLOT}	(Note 12)		60		120	µs
Recovery Time	t_{REC}	(Note 12)		1			µs
Write 0 Low Time	t_{LOW0}	(Note 12)		60		120	µs
Write 1 Low Time	t_{LOW1}	(Note 12)		1		15	µs
Read Data Valid	t_{RDV}	(Note 12)				15	µs
Reset Time High	t_{RSTH}	(Note 12)		480			µs
Reset Time Low	t_{RSTL}	(Notes 12, 13)		480			µs
Presence-Detect High	t_{PDHIGH}	(Note 12)		15		60	µs
Presence-Detect Low	t_{PDLOW}	(Note 12)		60		240	µs
Capacitance	$C_{IN/OUT}$					25	pF

Note 12: See the timing diagrams in Figure 2.
Note 13: Under parasite power, if t_{RSTL} > 960µs, a power-on reset can occur.

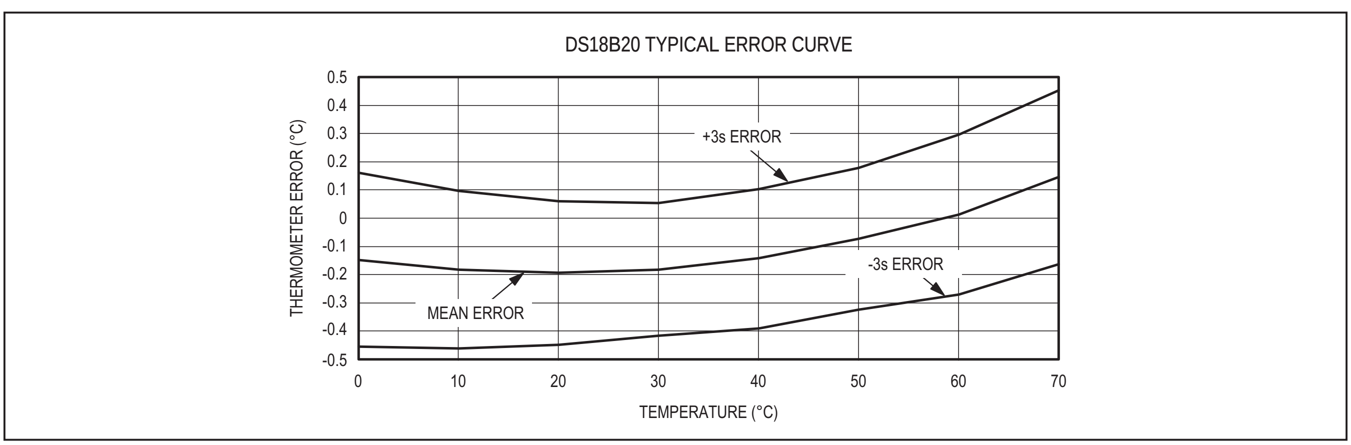


Figure 1. Typical Performance Curve

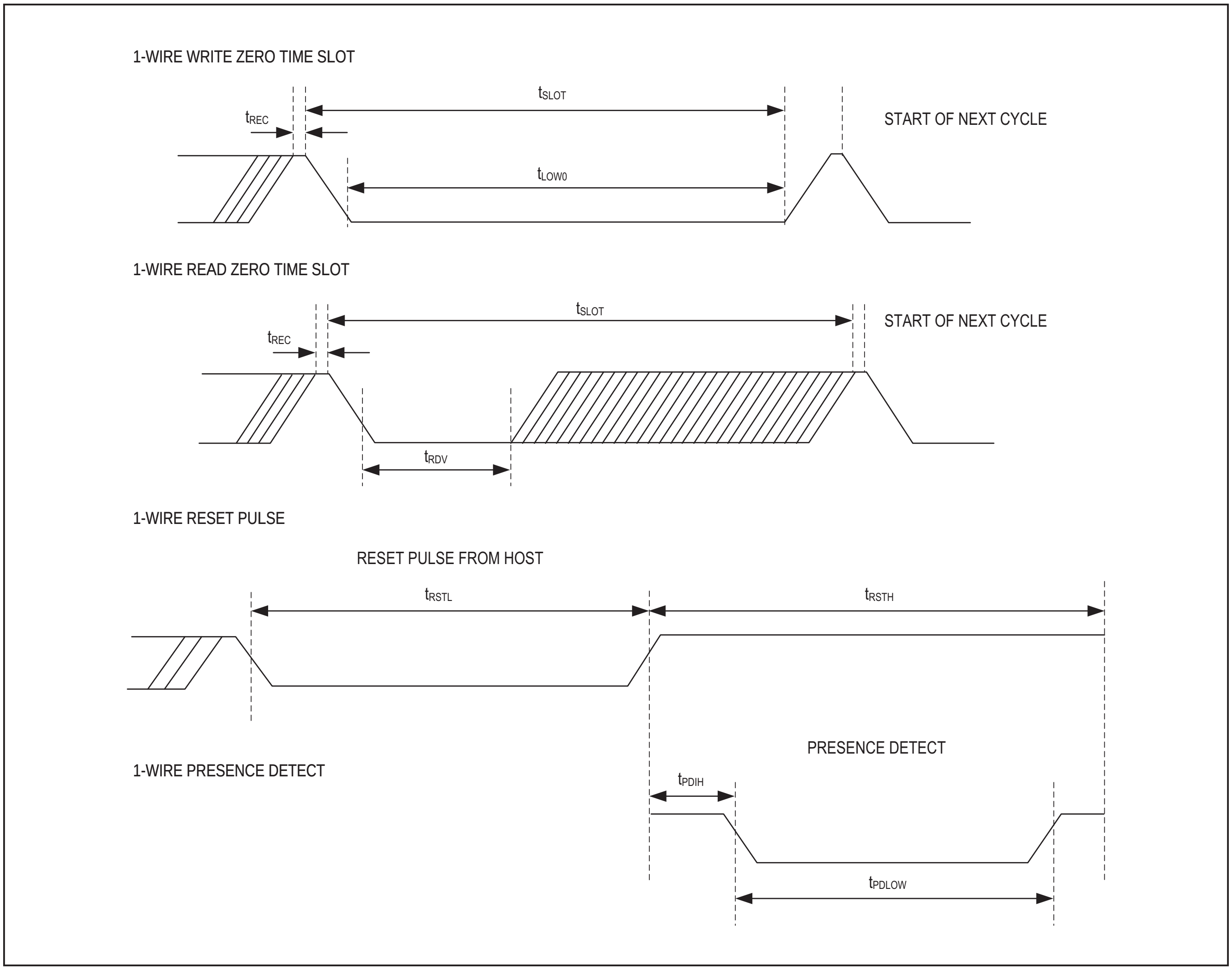


Figure 2. Timing Diagrams

Pin Description

PIN			NAME	FUNCTION
SO	µSOP	TO-92		
1, 2, 6, 7, 8	2, 3, 5, 6, 7	—	N.C.	No Connection
3	8	3	V_{DD}	Optional V_{DD}. V_{DD} must be grounded for operation in parasite power mode.
4	1	2	DQ	Data Input/Output. Open-drain 1-Wire interface pin. Also provides power to the device when used in parasite power mode (see the *Powering the DS18B20* section.)
5	4	1	GND	Ground

Overview

Figure 3 shows a block diagram of the DS18B20, and pin descriptions are given in the *Pin Description* table. The 64-bit ROM stores the device's unique serial code. The scratchpad memory contains the 2-byte temperature register that stores the digital output from the temperature sensor. In addition, the scratchpad provides access to the 1-byte upper and lower alarm trigger registers (T_H and T_L) and the 1-byte configuration register. The configuration register allows the user to set the resolution of the temperature-to-digital conversion to 9, 10, 11, or 12 bits. The T_H, T_L, and configuration registers are nonvolatile (EEPROM), so they will retain data when the device is powered down.

The DS18B20 uses Maxim's exclusive 1-Wire bus protocol that implements bus communication using one control signal. The control line requires a weak pullup resistor since all devices are linked to the bus via a 3-state or open-drain port (the DQ pin in the case of the DS18B20). In this bus system, the microprocessor (the master device) identifies and addresses devices on the bus using each device's unique 64-bit code. Because each device has a unique code, the number of devices that can be addressed on one bus is virtually unlimited. The 1-Wire bus protocol, including detailed explanations of the commands and "time slots," is covered in the *1-Wire Bus System* section.

Another feature of the DS18B20 is the ability to operate without an external power supply. Power is instead supplied through the 1-Wire pullup resistor through the DQ pin when the bus is high. The high bus signal also charges an internal capacitor (C_{PP}), which then supplies power to the device when the bus is low. This method of deriving power from the 1-Wire bus is referred to as "parasite power." As an alternative, the DS18B20 may also be powered by an external supply on V_{DD}.

Operation—Measuring Temperature

The core functionality of the DS18B20 is its direct-to-digital temperature sensor. The resolution of the temperature sensor is user-configurable to 9, 10, 11, or 12 bits, corresponding to increments of 0.5°C, 0.25°C, 0.125°C, and 0.0625°C, respectively. The default resolution at power-up is 12-bit. The DS18B20 powers up in a low-power idle state. To initiate a temperature measurement and A-to-D conversion, the master must issue a Convert T [44h] command. Following the conversion, the resulting thermal data is stored in the 2-byte temperature register in the scratchpad memory and the DS18B20 returns to its idle state. If the DS18B20 is powered by an external supply, the master can issue "read time slots" (see the *1-Wire Bus System* section) after the Convert T command and the DS18B20 will respond by transmitting 0 while the temperature conversion is in progress and 1 when the conversion is done. If the DS18B20 is powered with parasite power, this notification technique cannot be used since the bus must be pulled high by a strong pullup during the entire temperature conversion. The bus requirements for parasite power are explained in detail in the *Powering the DS18B20* section.

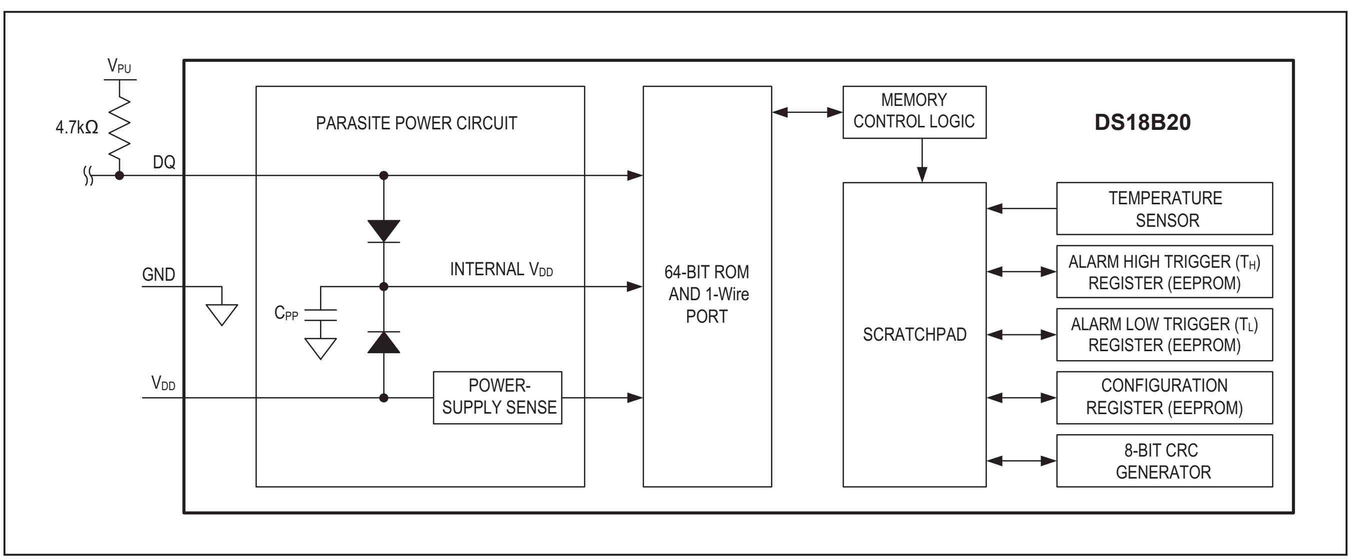


Figure 3. DS18B20 Block Diagram

The DS18B20 output temperature data is calibrated in degrees Celsius; for Fahrenheit applications, a lookup table or conversion routine must be used. The temperature data is stored as a 16-bit sign-extended two's complement number in the temperature register (see Figure 4). The sign bits (S) indicate if the temperature is positive or negative: for positive numbers S = 0 and for negative numbers S = 1. If the DS18B20 is configured for 12-bit resolution, all bits in the temperature register will contain valid data. For 11-bit resolution, bit 0 is undefined. For 10-bit resolution, bits 1 and 0 are undefined, and for 9-bit resolution bits 2, 1, and 0 are undefined. Table 1 gives examples of digital output data and the corresponding temperature reading for 12-bit resolution conversions.

Operation—Alarm Signaling

After the DS18B20 performs a temperature conversion, the temperature value is compared to the user-defined two's complement alarm trigger values stored in the 1-byte T_H and T_L registers (see Figure 5). The sign bit (S) indicates if the value is positive or negative: for positive numbers S = 0 and for negative numbers S = 1. The T_H and T_L registers are nonvolatile (EEPROM) so they will retain data when the device is powered down. T_H and T_L can be accessed through bytes 2 and 3 of the scratchpad as explained in the *Memory* section.

Only bits 11 through 4 of the temperature register are used in the T_H and T_L comparison since T_H and T_L are 8-bit registers. If the measured temperature is lower than

	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
LS BYTE	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
	BIT 15	**BIT 14**	**BIT 13**	**BIT 12**	**BIT 11**	**BIT 10**	**BIT 9**	**BIT 8**
MS BYTE	S	S	S	S	S	2^6	2^5	2^4

S = SIGN

Figure 4. Temperature Register Format

Table 1. Temperature/Data Relationship

TEMPERATURE (°C)	DIGITAL OUTPUT (BINARY)	DIGITAL OUTPUT (HEX)
+125	0000 0111 1101 0000	07D0h
+85*	0000 0101 0101 0000	0550h
+25.0625	0000 0001 1001 0001	0191h
+10.125	0000 0000 1010 0010	00A2h
+0.5	0000 0000 0000 1000	0008h
0	0000 0000 0000 0000	0000h
-0.5	1111 1111 1111 1000	FFF8h
-10.125	1111 1111 0101 1110	FF5Eh
-25.0625	1111 1110 0110 1111	FE6Fh
-55	1111 1100 1001 0000	FC90h

The power-on reset value of the temperature register is +85°C.

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
S	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Figure 5. T_H and T_L Register Format

ULN200x, ULQ200x High-Voltage, High-Current Darlington Transistor Arrays

1 Features

- 500-mA-Rated Collector Current (Single Output)
- High-Voltage Outputs: 50 V
- Output Clamp Diodes
- Inputs Compatible With Various Types of Logic
- Relay-Driver Applications

2 Applications

- Relay Drivers
- Stepper and DC Brushed Motor Drivers
- Lamp Drivers
- Display Drivers (LED and Gas Discharge)
- Line Drivers
- Logic Buffers

3 Description

The ULx200xA devices are high-voltage, high-current Darlington transistor arrays. Each consists of seven NPN Darlington pairs that feature high-voltage outputs with common-cathode clamp diodes for switching inductive loads.

The collector-current rating of a single Darlington pair is 500 mA. The Darlington pairs can be paralleled for higher current capability. Applications include relay drivers, hammer drivers, lamp drivers, display drivers (LED and gas discharge), line drivers, and logic buffers. For 100-V (otherwise interchangeable) versions of the ULx2003A devices, see the SLRS023 data sheet for the SN75468 and SN75469 devices.

The ULN2002A device is designed specifically for use with 14-V to 25-V PMOS devices. Each input of this device has a Zener diode and resistor in series to control the input current to a safe limit. The ULx2003A devices have a 2.7-kΩ series base resistor for each Darlington pair for operation directly with TTL or 5-V CMOS devices.

The ULx2004A devices have a 10.5-kΩ series base resistor to allow operation directly from CMOS devices that use supply voltages of 6 V to 15 V. The required input current of the ULx2004A device is below that of the ULx2003A devices, and the required voltage is less than that required by the ULN2002A device.

Device Information[1]

PART NUMBER	PACKAGE	BODY SIZE (NOM)
ULx200xD	SOIC (16)	9.90 mm × 3.91 mm
ULx200xN	PDIP (16)	19.30 mm × 6.35 mm
ULN200xNS	SOP (16)	10.30 mm × 5.30 mm
ULN200xPW	TSSOP (16)	5.00 mm × 4.40 mm

(1) For all available packages, see the orderable addendum at the end of the data sheet.

Simplified Block Diagram

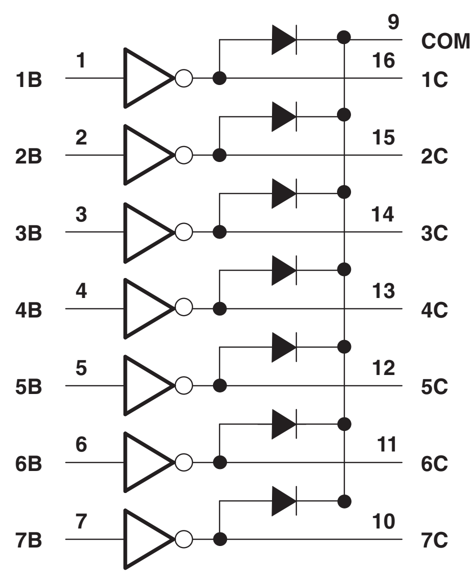


Table of Contents

4 Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from Revision N (June 2015) to Revision O	Page
• Changed Pin Functions table to correct typographical error.	3

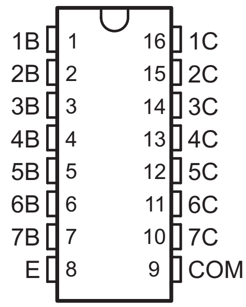
Changes from Revision M (February 2013) to Revision N	Page
• Added *Pin Configuration and Functions* section, *ESD Ratings* table, *Feature Description* section, *Device Functional Modes*, *Application and Implementation* section, *Power Supply Recommendations* section, *Layout* section, *Device and Documentation Support* section, and *Mechanical, Packaging, and Orderable Information* section	1
• Deleted *Ordering Information* table. No specification changes.	1
• Moved *Typical Characteristics* into *Specifications* section.	8

Changes from Revision L (April 2012) to Revision M	Page
• Updated temperature rating for ULN2003AI in the ORDERING INFORMATION table	1

Changes from Revision K (August 2011) to Revision L	Page
• Removed reference to obsolete ULN2001 device	1

5 Pin Configuration and Functions

D, N, NS, and PW Package
16-Pin SOIC, PDIP, SO, and TSSOP
Top View



Pin Functions

PIN		I/O[1]	DESCRIPTION
NAME	NO.		
1B	1		
2B	2		
3B	3		
4B	4	I	Channel 1 through 7 Darlington base input
5B	5		
6B	6		
7B	7		
1C	16		
2C	15		
3C	14		
4C	13	O	Channel 1 through 7 Darlington collector output
5C	12		
6C	11		
7C	10		
COM	9	—	Common cathode node for flyback diodes (required for inductive loads)
E	8	—	Common emitter shared by all channels (typically tied to ground)

(1) I = Input, O = Output

6 Specifications

6.1 Absolute Maximum Ratings

at 25°C free-air temperature (unless otherwise noted)[1]

			MIN	MAX	UNIT
V_{CC}	Collector-emitter voltage			50	V
	Clamp diode reverse voltage[2]			50	V
V_I	Input voltage[2]			30	V
	Peak collector current, See Figure 4 and Figure 5			500	mA
I_{OK}	Output clamp current			500	mA
	Total emitter-terminal current			–2.5	A
T_A	Operating free-air temperature range	ULN200xA	–20	70	°C
		ULN200xAI	–40	105	
		ULQ200xA	–40	85	
		ULQ200xAT	–40	105	
T_J	Operating virtual junction temperature			150	°C
	Lead temperature for 1.6 mm (1/16 inch) from case for 10 seconds			260	°C
T_{stg}	Storage temperature		–65	150	°C

(1) Stresses beyond those listed under *Absolute Maximum Ratings* may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under *Recommended Operating Conditions* is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.
(2) All voltage values are with respect to the emitter/substrate terminal E, unless otherwise noted.

6.2 ESD Ratings

			VALUE	UNIT
$V_{(ESD)}$	Electrostatic discharge	Human body model (HBM), per ANSI/ESDA/JEDEC JS-001[1]	±2000	V
		Charged device model (CDM), per JEDEC specification JESD22-C101[2]	±500	

(1) JEDEC document JEP155 states that 500-V HBM allows safe manufacturing with a standard ESD control process.
(2) JEDEC document JEP157 states that 250-V CDM allows safe manufacturing with a standard ESD control process.

6.3 Recommended Operating Conditions

over operating free-air temperature range (unless otherwise noted)

		MIN	MAX	UNIT
V_{CC}	Collector-emitter voltage (non-V devices)	0	50	V
T_J	Junction temperature	–40	125	°C

6.4 Thermal Information

THERMAL METRIC[1]		ULx200x				UNIT
		D (SOIC)	N (PDIP)	NS (SO)	PW (TSSOP)	
		16 PINS	16 PINS	16 PINS	16 PINS	
$R_{\theta JA}$	Junction-to-ambient thermal resistance	73	67	64	108	°C/W
$R_{\theta JC(top)}$	Junction-to-case (top) thermal resistance	36	54	n/a	33.6	°C/W
$R_{\theta JB}$	Junction-to-board thermal resistance	n/a	n/a	n/a	51.9	°C/W
ψ_{JT}	Junction-to-top characterization parameter	n/a	n/a	n/a	2.1	°C/W
ψ_{JB}	Junction-to-board characterization parameter	n/a	n/a	n/a	51.4	°C/W

(1) For more information about traditional and new thermal metrics, see the *Semiconductor and IC Package Thermal Metrics* application report, SPRA953.

6.5 Electrical Characteristics: ULN2002A

T_A = 25°C

PARAMETER		TEST FIGURE	TEST CONDITIONS		ULN2002A MIN	ULN2002A TYP	ULN2002A MAX	UNIT
$V_{I(on)}$	ON-state input voltage	Figure 14	V_{CE} = 2 V,	I_C = 300 mA			13	V
V_{OH}	High-level output voltage after switching	Figure 18	V_S = 50 V, I_O = 300 mA		V_S – 20			mV
$V_{CE(sat)}$	Collector-emitter saturation voltage	Figure 12	I_I = 250 µA,	I_C = 100 mA		0.9	1.1	V
			I_I = 350 µA,	I_C = 200 mA		1	1.3	
			I_I = 500 µA,	I_C = 350 mA		1.2	1.6	
V_F	Clamp forward voltage	Figure 15	I_F = 350 mA			1.7	2	V
I_{CEX}	Collector cutoff current	Figure 9	V_{CE} = 50 V,	I_I = 0			50	µA
		Figure 10	V_{CE} = 50 V, T_A = 70°C	I_I = 0			100	
				V_I = 6 V			500	
$I_{I(off)}$	OFF-state input current	Figure 10	V_{CE} = 50 V,	I_C = 500 µA	50	65		µA
I_I	Input current	Figure 11	V_I = 17 V			0.82	1.25	mA
I_R	Clamp reverse current	Figure 14	V_R = 50 V	T_A = 70°C			100	µA
			V_R = 50 V				50	
C_i	Input capacitance		V_I = 0,	f = 1 MHz			25	pF

6.6 Electrical Characteristics: ULN2003A and ULN2004A

T_A = 25°C

PARAMETER		TEST FIGURE	TEST CONDITIONS		ULN2003A MIN	ULN2003A TYP	ULN2003A MAX	ULN2004A MIN	ULN2004A TYP	ULN2004A MAX	UNIT
$V_{I(on)}$	ON-state input voltage	Figure 14	V_{CE} = 2 V	I_C = 125 mA						5	V
				I_C = 200 mA			2.4			6	
				I_C = 250 mA			2.7				
				I_C = 275 mA						7	
				I_C = 300 mA			3				
				I_C = 350 mA						8	
V_{OH}	High-level output voltage after switching	Figure 18	V_S = 50 V, I_O = 300 mA		V_S – 20			V_S – 20			mV
$V_{CE(sat)}$	Collector-emitter saturation voltage	Figure 13	I_I = 250 µA,	I_C = 100 mA		0.9	1.1		0.9	1.1	V
			I_I = 350 µA,	I_C = 200 mA		1	1.3		1	1.3	
			I_I = 500 µA,	I_C = 350 mA		1.2	1.6		1.2	1.6	
I_{CEX}	Collector cutoff current	Figure 9	V_{CE} = 50 V,	I_I = 0			50			50	µA
		Figure 10	V_{CE} = 50 V, T_A = 70°C	I_I = 0			100			100	
				V_I = 6 V						500	
V_F	Clamp forward voltage	Figure 16	I_F = 350 mA			1.7	2		1.7	2	V
$I_{I(off)}$	Off-state input current	Figure 11	V_{CE} = 50 V, T_A = 70°C,	I_C = 500 µA	50	65		50	65		µA
I_I	Input current	Figure 12	V_I = 3.85 V			0.93	1.35				mA
			V_I = 5 V						0.35	0.5	
			V_I = 12 V						1	1.45	
I_R	Clamp reverse current	Figure 15	V_R = 50 V				50			50	µA
			V_R = 50 V	T_A = 70°C			100			100	
C_i	Input capacitance		V_I = 0,	f = 1 MHz		15	25		15	25	pF

6.7 Electrical Characteristics: ULN2003AI

T_A = 25°C

PARAMETER		TEST FIGURE	TEST CONDITIONS		ULN2003AI MIN	ULN2003AI TYP	ULN2003AI MAX	UNIT
$V_{I(on)}$	ON-state input voltage	Figure 14	V_{CE} = 2 V	I_C = 200 mA			2.4	V
				I_C = 250 mA			2.7	
				I_C = 300 mA			3	
V_{OH}	High-level output voltage after switching	Figure 18	V_S = 50 V, I_O = 300 mA		V_S – 50			mV
$V_{CE(sat)}$	Collector-emitter saturation voltage	Figure 13	I_I = 250 µA,	I_C = 100 mA		0.9	1.1	V
			I_I = 350 µA,	I_C = 200 mA		1	1.3	
			I_I = 500 µA,	I_C = 350 mA		1.2	1.6	
I_{CEX}	Collector cutoff current	Figure 9	V_{CE} = 50 V,	I_I = 0			50	µA
V_F	Clamp forward voltage	Figure 16	I_F = 350 mA			1.7	2	V
$I_{I(off)}$	OFF-state input current	Figure 11	V_{CE} = 50 V,	I_C = 500 µA	50	65		µA
I_I	Input current	Figure 12	V_I = 3.85 V			0.93	1.35	mA
I_R	Clamp reverse current	Figure 15	V_R = 50 V				50	µA
C_i	Input capacitance		V_I = 0,	f = 1 MHz		15	25	pF

6.8 Electrical Characteristics: ULN2003AI

T_A = –40°C to 105°C

PARAMETER		TEST FIGURE	TEST CONDITIONS		ULN2003AI MIN	ULN2003AI TYP	ULN2003AI MAX	UNIT
$V_{I(on)}$	ON-state input voltage	Figure 14	V_{CE} = 2 V	I_C = 200 mA			2.7	V
				I_C = 250 mA			2.9	
				I_C = 300 mA			3	
V_{OH}	High-level output voltage after switching	Figure 18	V_S = 50 V, I_O = 300 mA		V_S – 50			mV
$V_{CE(sat)}$	Collector-emitter saturation voltage	Figure 13	I_I = 250 µA,	I_C = 100 mA		0.9	1.2	V
			I_I = 350 µA,	I_C = 200 mA		1	1.4	
			I_I = 500 µA,	I_C = 350 mA		1.2	1.7	
I_{CEX}	Collector cutoff current	Figure 9	V_{CE} = 50 V,	I_I = 0			100	µA
V_F	Clamp forward voltage	Figure 16	I_F = 350 mA			1.7	2.2	V
$I_{I(off)}$	OFF-state input current	Figure 11	V_{CE} = 50 V,	I_C = 500 µA	30	65		µA
I_I	Input current	Figure 12	V_I = 3.85 V			0.93	1.35	mA
I_R	Clamp reverse current	Figure 15	V_R = 50 V				100	µA
C_i	Input capacitance		V_I = 0,	f = 1 MHz		15	25	pF