A Mini Project Report on

# IOT BASED UNDERGROUND CABLE FAULT DETECTION

submitted in partial fulfillment for award of the degree of Bachelor of Technology in

Electronics and Communication Engineering

By

**Sayana Moni Gayathri 17321A0472**

Under the esteemed Guidance of

Internal Guide

**B Eleena**

Assistant Professor, ECE Department



# Bhoj Reddy Engineering College for Women Department of Electronics and Communications Engineering

(Sponsored by Sangam Laxmibai Vidyapeet, approved by AICTE & affiliated to JNTUH)

## Vinaynagar, Santoshnagar X roads, Saidabad, Hyderabad – 500 059

Ph: +91-40-2459 2400 Fax: +91-40-2453 7281, www.brecw.ac.in, principal@brecw.ac.in

# 2020- 2021

Bhoj Reddy Engineering College for Women



(Sponsored by Sangam Laxmibai Vidyapeet, approved by AICTE & affiliated to JNTUH)

Vinaynagar, I S Sadan X Roads, Saidabad, Hyderabad 500 059, Telangana, India

Telephone: 040-2453 7282, Website: www.brecw.ac.in, Email: principal@brecw.ac.in

# CERTIFICATE

This is to certify that the Mini Project Report entitled **“IOT BASED**

**UNDERGROUND CABLE FAULT DETECTION”** is a bonafide work carried out by

## Sayana Moni Gayathri 17321A0472

## 

in partial fulfillment for award of the degree of Bachelor of Technology in Electronics and Communication Engineering from Bhoj Reddy Engineering College for Women, Hyderabad, affiliated to Jawaharlal Nehru Technological University Hyderabad (JNTUH).

**B Eleena Dr. N Shribala**  **Assistant Professor Associate Professor**

**Internal Guide HOD-ECE**

## External Examiner

Sangam Laxmibai Vidyapeet is an educational society for promotion of education among girls and women. It is established in 1952 and registered under the Telangana Societies Registration Act.

## Acknowledgement

The satisfaction that accompanies the successful completion of the task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crown all the efforts with success.

I would like to express my sincere gratitude to **Ms. B.Eleena**, Assistant professor and Seminar guide for the eminent guidance and supervision at every stage.

I am thankful to **Dr.N Shribala**, Head of the Department for her valuable guidance, and encouragement.

I also thank **Dr.J Madhavan**, Principal, BRECW for providing the wonderful education environment in our college.

I am equally thankful to **Ms. Uzma Nazneen**, Project Co-ordinator for her continuous support and all the staff of Electronics and Communication Department of BRECW for their timely help and suggestions in the project.

Sayana Moni Gayathri (17321A0472) smgayathri99@gmail.com

Table of Contents Page No:

Chapter 1. Introduction 1

1.1 Embedded Systems 1

Chapter 2. Description of Hardware Components 2

2.1 ARDUINO 2

2.1.1 Pin description for ARDUINO 6

2.1.3.1 Power and Ground 6

2.1.3.2 Clock 6

2.1.3.3 Reset 7

2.1.3.4 SPI Programming 7

2.1.3.5 I/O Ports 7

2.1.4 Timers/Counters 8

2.1.4.1 Communication 9

2.1.5 Memory 10

2.2 Power Supply 10

2.2.1 Introduction 10

2.2.2 Transformer 11

2.2.3 Rectifier 11

2.2.4 Regulator 15

2.3 RS 232 17

2.3.1 Introduction 17

2.3.2 RS-232 Level Converter 18

2.4 Serial Communication 18

2.5 LCD 20

2.5.1 Commands used in LCD 22

2.6 Relay SPDT 23

2.7 ULN2003 26

2.7.1 Darlington Pair 27

2.8 ESP 8266 29

2.8.1 ESP8266 Features 29

Chapter 3 Circuit diagram 33

Chapter 4. Source Code 34

Chapter 5. Result and Conclusion

5.1 Advantages 41

5.2 Disadvantages 41

5.3 Applications 41

5.4 Conclusion 41

5.5 Final view of kit 42

References 43

### List of Figures Page No

Fig 1.1 Block Diagram 1

Fig 2.1.1 Arduino Uno 2

Fig 2.1.2 Arduino 3

Fig 2.1.3 Internal Block Diagram 5

Fig 2.2.1 Power Supply 10

Fig 2.2.2 Transformer 11

Fig 2.2.3.1 Half Wave Rectifier 12

Fig 2.2.3.2 Full Wave Rectifier 12

Fig 2.2.3.3 Bridge rectifier 13

Fig 2.2.3.4 Filter Capacitor waveform 14

Fig 2.2.4.1 Regulator 15

Fig 2.2.4.2 Circuit Diagram of Power Supply 16

Fig 2.3.1.1 RS232 17

Fig 2.3.1.2 RS232 Logic Waveforms 17

Fig 2.3.2 RS232 Level Converter 18

Fig 2.4.1 Serial Communication 19

Fig 2.5.1 16\*2 LCD module 20

Fig 2.6.1 Relay 23

Fig 2.6.2 Working of Relay 25

Fig 2.6.3 SPDT Relay 26

Fig 2.7.1 ULN2003 26

Fig 2.7.2 Block and pin diagram of ULN2003 27

Fig 2.8.1 ESP8266 29

Fig 2.8.2 ESP01 Module 34

### List of Tables

Page No

Table 2.1 Features of Arduino Uno 3

Table 2 Specifications of IC7805 10

Table 3 Operations of LCD 21

Table 4 Commands used in LCD 22

### Abstract

The Objective of this project is to determine the distance of underground cable fault from base station in kilometers. The underground cable system is common practice followed in many urban areas. While a fault occur for some reason, at that time the repairing process related to that particular cable is difficult due to not knowing the exact location of the fault. The proposed system is to find the exact location of the fault.

The project uses the standard concept of Ohms law i.e., when a low DC voltage is applied at the feeder end through a series resistor (Cable lines), then current would vary depending upon the location of fault in the cable. In case there is a short circuit (Line to Ground), the voltage across series resistors changes accordingly, which is then fed to an ADC to develop precise digital data which the programmed microcontroller of 8051 family would display in kilometers.

The project is assembled with a set of resistors representing cable length in KM’s and fault creation is made by a set of switches at every known KM to cross check the accuracy of the same. The fault occurring at a particular distance and the respective phase is displayed on a LCD interfaced to the microcontroller.

Further this project can be enhanced by using capacitor in an ac circuit to measure the impedance which can even locate the open circuited cable, unlike the short circuited fault only using resistors in DC circuit as followed in the above proposed project.

Chapter 1 Introduction

1.1 Embedded Systems

Embedded systems are designed to do some specific task, rather than be a general-purpose computer for multiple tasks. Some also have real time performance constraints that must be met, for reason such as safety and usability; others may have low or no performance requirements, allowing the system hardware to be simplified to reduce costs.

Wireless communication has become an important feature for commercial products and a popular research topic within the last ten years. There are now more mobile phone subscriptions than wired-line subscriptions. Lately, one area of commercial interest has been low-cost, low-power, and short-distance wireless communication used for \personal wireless networks." Technology advancements are providing smaller and more cost effective devices for integrating computational processing, wireless communication, and a host of other functionalities. These embedded communications devices will be integrated into applications ranging from homeland security to industry automation and monitoring. They will also enable custom tailored engineering solutions, creating a revolutionary way of disseminating and processing information. With new technologies and devices come new business activities, and the need for employees in these technological areas. Engineers who have knowledge of embedded systems and wireless communications will be in high demand. Unfortunately, there are few adorable environments available for development and classroom use, so students often do not learn about these technologies during hands-on lab exercises. The communication mediums were twisted pair, optical fiber, infrared, and generally wireless radio.

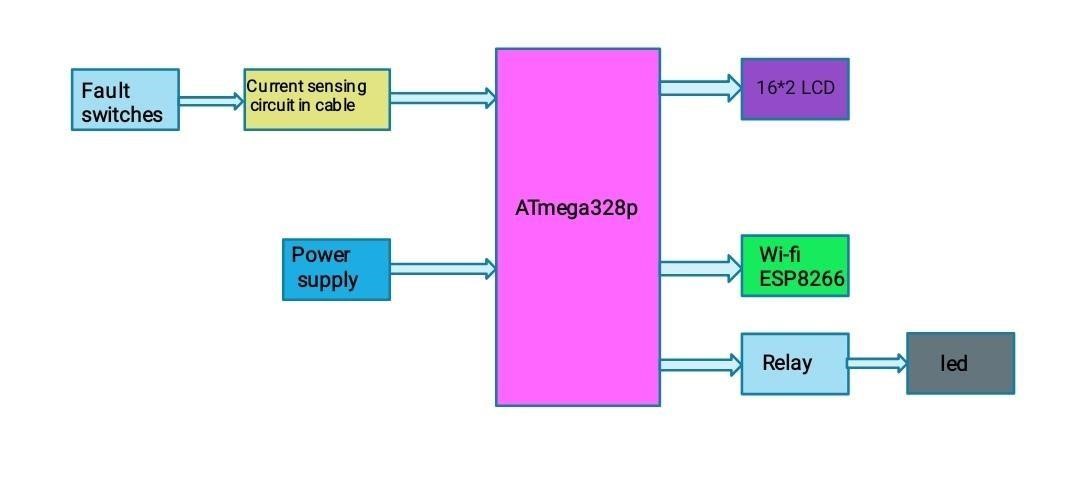


Fig 1.1 Block Diagram

#### Chapter 2 Description of Hardware Components

2.1 Arduino

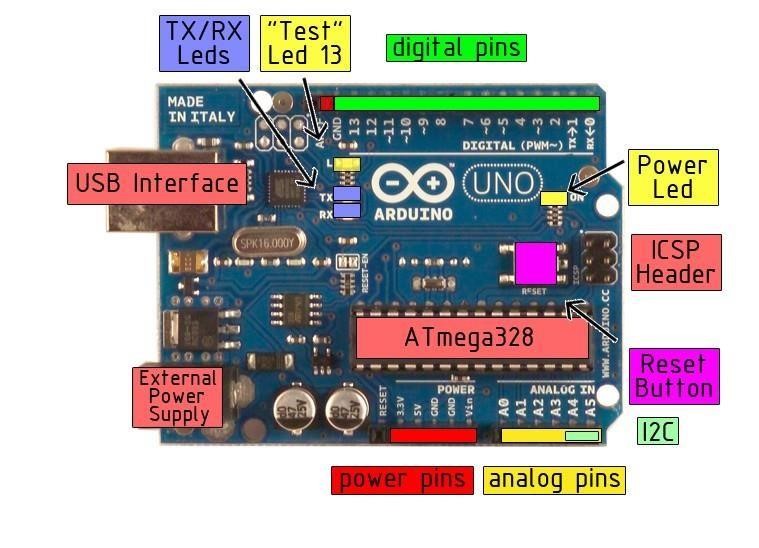


Fig 2.1.1 Arduino Uno

Arduino/Genuino Uno is a microcontroller board based on the ATmega328P

[(datasheet).](http://www.atmel.com/Images/doc8161.pdf) It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.. You can tinker with your UNO without worring too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.

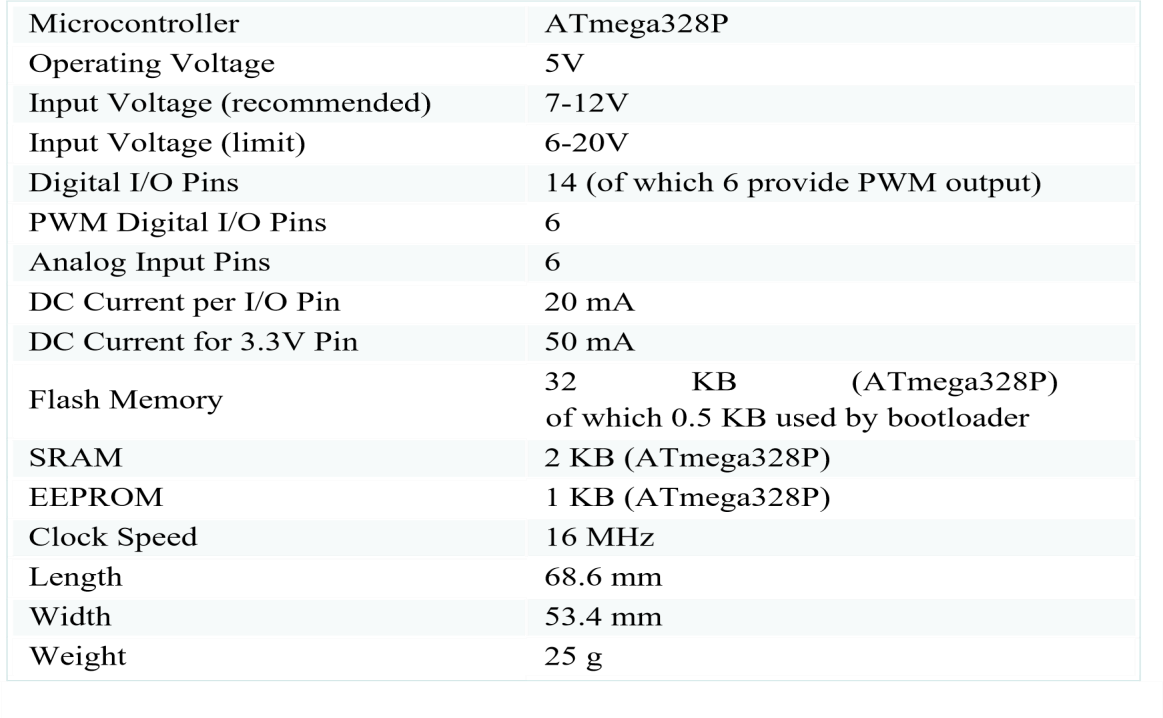
"Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past or outdated boards see the Arduino index of boards.



#### Fig 2.1.2 Arduino

You can find [here y](https://www.arduino.cc/en/Main/warranty)our board warranty informations.

You can find in the [Getting Started section a](https://www.arduino.cc/en/Guide/HomePage)ll the information you need to configure your board, use the Arduino So ft ware (IDE), and start tinker with coding and electronics.



#### Table 2.1 Features

Arduino / Genuino Uno is open-source hardware! You can build your own board using the follwing files:

Programming

The Arduino/Genuino Uno can be programmed with the ([Arduino Software (](https://www.arduino.cc/en/Main/Software)IDE)). Select "Arduino/Genuino Uno from the Tools > Board menu (according to the microcontroller on your board). For details, see the [reference a](https://www.arduino.cc/en/Reference/HomePage)nd [tutorials.](https://www.arduino.cc/en/Tutorial/HomePage)

The ATmega328 on the Arduino/Genuino Uno comes preprogrammed with a [bootloader t](https://www.arduino.cc/en/Hacking/Bootloader?from=Tutorial.Bootloader)hat allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol [(reference](http://www.atmel.com/Images/doc2525.pdf)[, C header files).](http://www.atmel.com/dyn/resources/prod_documents/avr061.zip)

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header using [Arduino ISP o](https://www.arduino.cc/en/Main/ArduinoISP)r similar; see [these instructions f](https://www.arduino.cc/en/Hacking/Programmer)or details.

The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available in the Arduino repository. The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

* On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then rese ing the 8U2.
* On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

You can then use [Atmel's FLIP software (W](http://www.atmel.com/products/microcontrollers/default.aspx)indows) or the [DFU programmer (](http://dfu-programmer.github.io/)Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See [this user-contributed tutorial f](http://forum.arduino.cc/index.php/topic,111.0.html)or more information.

Differences with other boards

The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

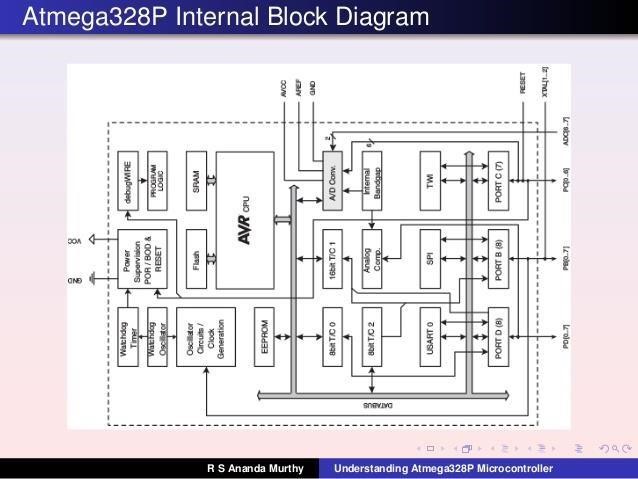
Power

The Arduino/Genuino Uno board can be powered via the USB connection or with an external power supply. The power source is selected automatically.

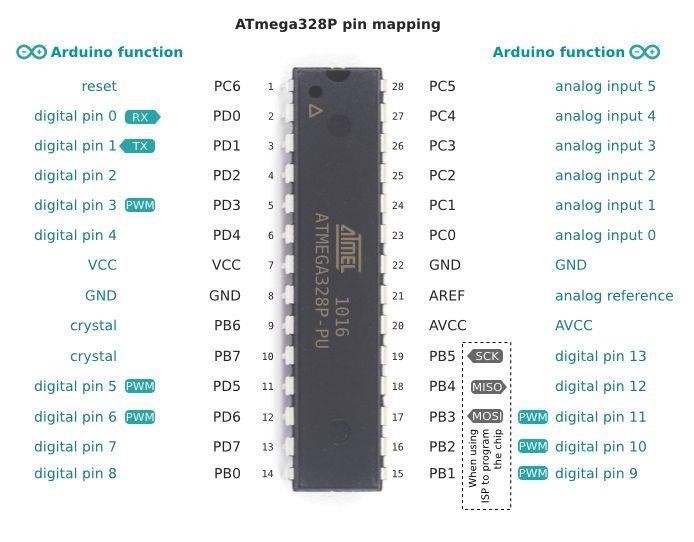
External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector.

The board can operate on an external supply from 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may become unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

Fig 2.1.3 Internal block diagram:

• 

2.1.1 Pin Diagram of ATMEGA328P:



##### 2.1.1.1 Power and Ground

The power supply voltage (5 volts) must be connected to the VCC input on pin 7. The ground connections are on pins 8 and 22.

##### 2.1.1.2 Clock

Some sort of clock signal must be provided in order for the microcontroller to operate. On the ATmega328P the clock can come from one of three di\_erent sources. The selection of the clock source is done by program ming fuse bits in the chip. A TTLcompatible clock signal can be generated externally by other logic and connected to the XTAL1 input (pin 9.) This probably the easiest way to generate the clock for the EE 459 projects. The lab has a supply of DIP oscillators in some of the more common frequencies. These output a TTL level square wave that can be fed directly into the microcontroller and to other chips. Alternatively, the processor can generate a clock if a crystal is connected to the XTAL1 and XTAL2 inputs. This method uses a plain crystal, not the DIP crystal oscillators as described above. The third method uses an internal oscillator that runs at approximately 8MHz. This is probably the least accurate way to generate a clock. Do not use this method if your project requires a clock running close to a specified frequency. The advantage of using the internal clock is that you do not need to provide any external signal and other functions are now available on pin 9. For example it can now be used as Port B bit 6 (PB6) thus giving the microcontroller 22 I/O pins. In applications where the UART0 serial communications interface is being used, the choice of clock frequency determines the baud rates that can be used for transmitting and receiving serial data. The accuracy of the frequency of the baud rate depends on the clock frequency used for the microcontroller. If a high degree of accuracy is required, an external oscillator of the correct frequency will be needed..

##### 2.1.1.3 Reset

The reset input (RESET, pin 1) must be in the high state for the processor to operate normally. This pin has an internal pull-up and does not have to be externally pulled-up to VCC in order for the processor to operate normally.

##### 2.1.1.4 SPI Programming

The Flash memory on the ATmega328P is programed using connections to the reset input and three other pins: PB3, PB4 and PB5. These three I/O pins can be used for other purposes as long as the design allows the programming hardware to have sole access to these pins during the programming process. Make sure that none of these pins is used as in input from some source that will continue to drive a signal at the 328P while the reset line is in the low state.

##### 2.1.1.5 I/O Ports

See the mapping between Arduino pins and ATmega328P ports. The mapping for the Atmega8, 168, and 328 is identical Each of the 14 digital pins on the Uno can be used as an input or output, using [pinMode(),](https://www.arduino.cc/en/Reference/PinMode) [digitalWrite(),](https://www.arduino.cc/en/Reference/DigitalWrite) and [digitalRead() f](https://www.arduino.cc/en/Reference/DigitalRead)unctions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 2050k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

Port B (PB)

Port B on the ATmega328P has seven usable pins (PB0 through PB5 and PB7). A eighth bit, PB6, shares a pin with the XTAL1 input. If the chip is configured for an external clock, this pin is not available for I/O. Three of the pins (PB3, PB4 and PB5) are use for the SPI interface for programming the Flash memory. These pins should not be used as inputs connected to sources that will continue to drive signals at the 328P while in the reset state. Port C (PC)

Port C on the 328P has six pins (PC0 through PC5). A seventh bit, PC6, shares a pin with the RESET input. By changing the configuration fuse settings this bit can be use for I/O. Most of the pins in PC are shared with the analog-to-digital converter so if the ADC function is used one or more pins will not be available for general purpose I/O.

In addition, PC4 and PC5 are use for the I2C interface and will not be available for general I/O if I2C is used

Port D (PD)

Port D on the 328P has eight pins (PD0 through PD7). Two of the pins, PD0 and PD1, are shared with the serial communications interface and can not be used as I/O if the USART0 functions are used.

##### 2.1.4 Timer/Counters

The ATmega328P contains three timers:

Timer/Counter0 - an 8-bit counter.

Timer/Counter1 - a 16-bit counter.

Timer/Counter2 - an 8-bit counter similar to Timer/Counter0 but with asynchronous clocking capability. The internal timers can be used to count events and generate an interrupt when a specified number of events has occurred. A common use of a timer is to implement a delay function by counting the number of internal clock cycles that occur. The example on the class web site in program at328-2.c and discussed below uses the 16-bit timer but the the procedure is similar for the 8-bit timers. To implements a delay \_rst set the timer for \Clear Timer on Compare Match" (CTC) mode using Output Compare Register A (OCR1A). The mode is set using four bits: WGM12 and WGM13 in TCCR1B, and WGM10 and WGM11 in TCCR1A. In most situations enabling the CTC interrupt is also required. This is done by setting the OCIE1A bit in the TIMSK1 register. In this mode the counter counts up to the value in OCR1A, generates an interrupt, clears the count and starts counting up again. Use the rate of the internal clock to calculate what count value the counter will need to count to. If the maximum value exceeds the range of the timer's 16-bit register (greater then 65,535), determine what value to use in the prescaler to divide the internal clock by (8, 64, 256 or 1024) before it reaches the timer. The maximum count value, after any prescaling, is loaded into the Output Capture Register (OCR1A). The prescaler is controlled by bits CS10, CS11 and CS12 in TCCR1B. The action of setting the prescaler bits to something other than all zeros starts the timer counting. To turn the timer o\_, set the prescaler bits to all zero. When the counter reaches the maximum count value, it generates an interrupt, resets the count value

to zero and continues to count. The user program should service the interrupt and take whatever action is necessary. Keep in mind that the counter does not stop and wait for the interrupt to be serviced. It continues to count regardless of when or if the user program services the interrupt.

In addition ,some pins have specialized functiones:

* Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
* External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.

* PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function. • SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
* LED: 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
* TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analog Reference() function. There are a couple of other pins on the board:

* AREF. Reference voltage for the analog inputs. Used with analog Reference().
* Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

###### 2.1.4.1 Communication

Arduino/Genuino Uno has a number of facilities for communicating with a computer, another Arduino/Genuino board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, [on Windows, a .inf file is required.](https://www.arduino.cc/en/Guide/Windows#toc4) The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and

1). [A Software Serial library a](https://www.arduino.cc/en/Reference/SoftwareSerial)llows serial communication on any of the Uno's digital pins.

The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino Software (IDE) includes a Wire library to simplify use of the I2C bus; see the [documentation f](https://www.arduino.cc/en/Reference/Wire)or details. For SPI communication, use the [SPI library.](https://www.arduino.cc/en/Reference/SPI)

Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino/Genuino Uno board is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino Software (IDE) uses this capability to allow you to upload code by simply pressing the upload button in the interface toolbar. This means that the

bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Uno board contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESETEN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; for details.

2.1.4.1.1 Revisions

Revision 3 of the board has the following new features:

* 1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible with both the board that uses the AVR, which operates with 5V and with the Arduino Due that operates with 3.3V. The second one is a not connected pin, that is reserved for future purposes.
* Stronger RESET circuit.
* Atmega 16U2 replace the 8U2.

##### 2.1.5 Memory

The ATmega328 has 32 KB (with 0.5 KB occupied by the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written withthe [EEPROM library)](https://www.arduino.cc/en/Reference/EEPROM)

2.2 Power Supply

All digital circuits require regulated power supply. In this article we are going to learn how to get a regulated positive supply from the mains supply.

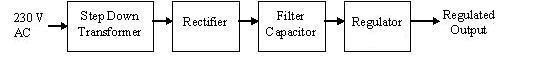


Fig 2.2.1 Power Supply

Figure 2.2.1 shows the basic block diagram of a fixed regulated power supply. Let us go through each block.

2.2.2 Transformer

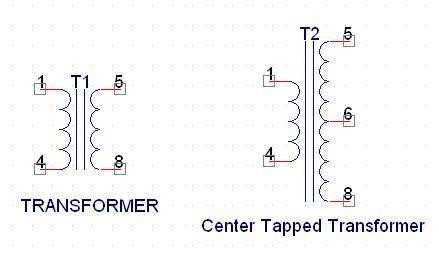


Fig 2.2.2.1 Transformer

A transformer consists of two coils also called as “Windings” namely Primary & Secondary.

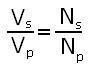
They are linked together through inductively coupled electrical conductors also called as Core. A changing current in the primary causes a change in the Magnetic Field in the core & this in turn induces an alternating voltage in the secondary coil. If load is applied to the secondary then an alternating current will flow through the load. If we consider an ideal condition then all the energy from the primary circuit will be transferred to the secondary circuit through the magnetic field.



So



The secondary voltage of the transformer depends on the number of turns in the Primary as well as in the secondary.



2.2.3 Rectifier

A rectifier is a device that converts an AC signal into DC signal. For rectification purpose we use a diode, a diode is a device that allows current to pass only in one direction i.e. when the anode of the diode is positive with respect to the cathode also called as forward biased condition & blocks current in the reversed biased condition.

Rectifier can be classified as follows:

1. Half Wave rectifier.

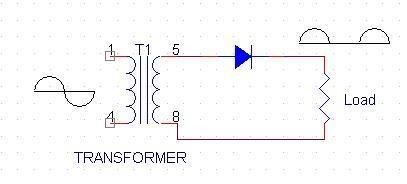


Fig 2.2.3.1 Half Wave rectifier

This is the simplest type of rectifier as you can see in the diagram a half wave rectifier consists of only one diode. When an AC signal is applied to it during the positive half cycle the diode is forward biased & current flows through it. But during the negative half cycle diode is reverse biased & no current flows through it. Since only one half of the input reaches the output, it is very inefficient to be used in power supplies.

1. Full wave rectifier.

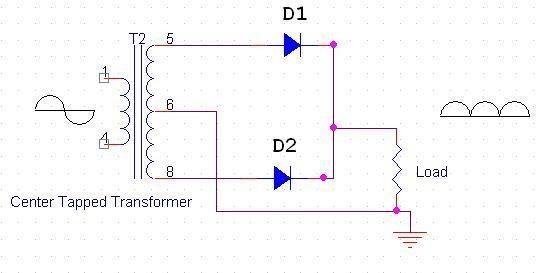


Fig 2.2.3.2 Full Wave rectifier

Half wave rectifier is quite simple but it is very inefficient, for greater efficiency we would like to use both the half cycles of the AC signal. This can be achieved by using a center tapped transformer i.e. we would have to double the size of secondary winding & provide connection to the center. So during the positive half cycle diode D1 conducts & D2 is in reverse biased condition. During the negative half cycle diode D2 conducts & D1 is reverse biased. Thus we get both the half cycles across the load.

One of the disadvantages of Full Wave Rectifier design is the necessity of using a center tapped transformer, thus increasing the size & cost of the circuit. This can be avoided by using the Full Wave Bridge Rectifier.

1. Bridge Rectifier.

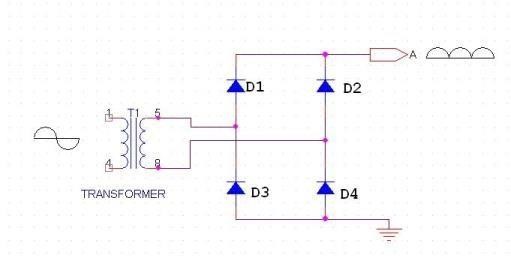
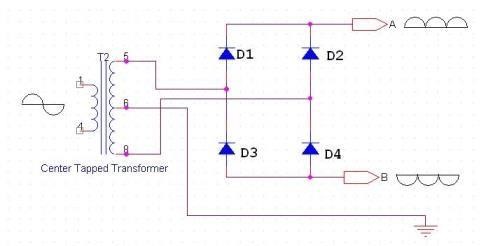


Fig 2.2.3.3 Bridge rectifier

As the name suggests it converts the full wave i.e. both the positive & the negative half cycle into DC thus it is much more efficient than Half Wave Rectifier & that too without using a center tapped transformer thus much more cost effective than Full Wave

Rectifier.

Full Bridge Wave Rectifier consists of four diodes namely D1, D2, D3 and D4. During the positive half cycle diodes D1 & D4 conduct whereas in the negative half cycle diodes D2 & D3 conduct thus the diodes keep switching the transformer connections so we get positive half cycles in the output.



If we use a center tapped transformer for a bridge rectifier we can get both positive & negative half cycles which can thus be used for generating fixed positive & fixed negative voltages.

Filter Capacitor

Even though half wave & full wave rectifier give DC output, none of them provides a constant output voltage. For this we require to smoothen the waveform received from the rectifier. This can be done by using a capacitor at the output of the rectifier this capacitor is also called as “Filter Capacitor” or “Smoothing Capacitor” or “Reservor Capacitor”. Even after using this capacitor a small amount of ripple will remain.

We place the Filter Capacitor at the output of the rectifier the capacitor will charge to the peak voltage during each half cycle then will discharge its stored energy slowly through the load while the rectified voltage drops to zero, thus trying to keep the voltage as constant as possible.

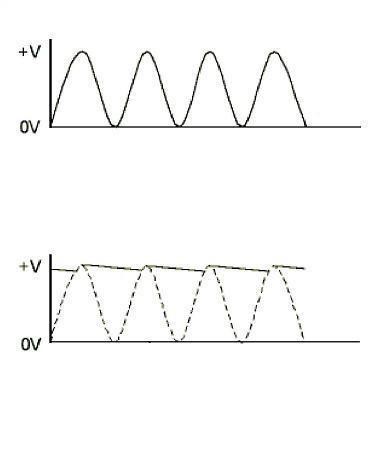
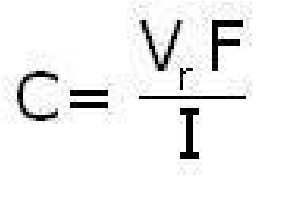


Fig 2.2.3.4 Filter Capacitor waveforms

If we go on increasing the value of the filter capacitor then the Ripple will decrease. But then the costing will increase. The value of the Filter capacitor depends on the current consumed by the circuit, the frequency of the waveform & the accepted ripple.



Where,

Vr= accepted ripple voltage.( should not be more than 10% of the voltage) I= current consumed by the circuit in Amperes.

F= frequency of the waveform. A half wave rectifier has only one peak in one cycle so F=25hz

Whereas a full wave rectifier has Two peaks in one cycle so F=100hz.

2.2.4 Regulator

A Voltage regulator is a device which converts varying input voltage into a constant regulated output voltage. Voltage regulator can be of two types

1. Linear Voltage Regulator

Also called as Resistive Voltage regulator because they dissipate the excessive voltage resistively as heat.

1. Switching Regulators.

They regulate the output voltage by switching the Current ON/OFF very rapidly. Since their output is either ON or OFF it dissipates very low power thus achieving higher efficiency as compared to linear voltage regulators. But they are more complex & generate high noise due to their switching action. For low level of output power switching regulators tend to be costly but for higher output wattage they are much cheaper than linear regulators.

The most commonly available Linear Positive Voltage Regulators are the 78XX series where the XX indicates the output voltage. And 79XX series is for Negative Voltage Regulators.

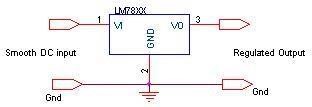


Fig 2.2.4.1 Regulator

After filtering the rectifier output the signal is given to a voltage regulator. The maximum input voltage that can be applied at the input is 35V.Normally there is a 2-3 Volts drop across the regulator so the input voltage should be at least 2-3 Volts higher than the output voltage. If the input voltage gets below the Vmin of the regulator due to the ripple voltage or due to any other reason the voltage regulator will not be able to produce the correct regulated voltage.

3 Circuit diagram:

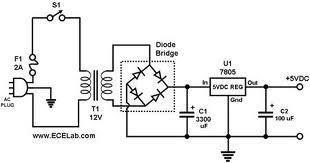


Fig 2.2.4.2 Circuit Diagram of power supply

IC 7805:

7805 is an integrated three-terminal positive fixed linear voltage regulator. It supports an input voltage of 10 volts to 35 volts and output voltage of 5 volts. It has a current rating of 1 amp although lower current models are available. Its output voltage is fixed at 5.0V. The 7805 also has a built-in current limiter as a safety feature. 7805 is manufactured by many companies, including National Semiconductors and Fairchild Semiconductors.

The 7805 will automatically reduce output current if it gets too hot.The last two digits represent the voltage; for instance, the 7812 is a 12-volt regulator. The 78xx series of regulators is designed to work in complement with the 79xx series of negative voltage regulators in systems that provide both positive and negative regulated voltages, since the 78xx series can't regulate negative voltages in such a system.

The 7805 & 78 is one of the most common and well-known of the 78xx series regulators, as it's small component count and medium-power regulated 5V make it useful for powering TTL devices.

|  |  |
| --- | --- |
| SPECIFICATIONS | IC 7805 |
| Vout | 5V |
| Vein - Vout Difference | 5V - 20V |
| Operation Ambient Temp | 0 - 125°C |
| Output Imax | 1A |

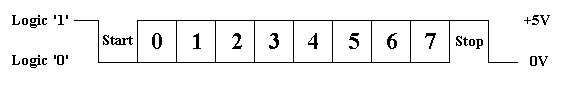
Table 2.2. Specifications of IC7805

### 2.3 RS232

#### 2.3.1 Introduction



Fig 2.3.1.1 RS-232 WAVEFORM



TTL/CMOS Serial Logic Waveform

The diagram above shows the expected waveform from the UART when using the common 8N1 format. 8N1 signifies 8 Data bits, No Parity and 1 Stop Bit. The RS-232 line, when idle is in the Mark State (Logic 1). A transmission starts with a start bit which is (Logic 0). Then each bit is sent down the line, one at a time. The LSB (Least Significant Bit) is sent first. A Stop Bit (Logic 1) is then appended to the signal to make up the transmission.

The data sent using this method, is said to be framed. That is the data is framed between a Start and Stop Bit. RS-232 Voltage levels

* +3 to +25 volts to signify a "Space" (Logic 0)
* -3 to -25 volts for a "Mark" (logic 1).
* Any voltage in between these regions (i.e. between +3 and -3 Volts) is undefined. The data byte is always transmitted least-significant-bit first.

The bits are transmitted at specific time intervals determined by the baud rate of the serial signal. This is the signal present on the RS-232 Port of your computer, shown below.

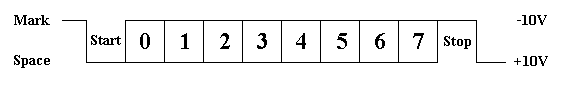


Fig 2.3.1.2 RS-232 Logic Waveform

2.3.2 RS-232 Level Converter

Standard serial interfacing of microcontroller (TTL) with PC or any  [RS232C](http://www.arcelect.com/rs232.htm) Standard device , requires TTL to RS232 Level converter . A [MAX232 i](http://www.bsc.nodak.edu/electron/rs232.htm)s used for this purpose. It provides 2-channel RS232C port and requires external 10uF capacitors. The driver requires a single supply of +5V. MAX-232 includes a Charge Pump, which generates +10V and -10V from a single 5v supply

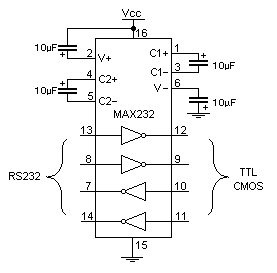
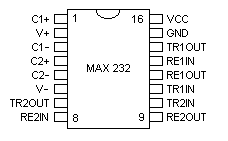


Fig 2.3.2 RS 232 Level Converter

2.4 Serial communication

When a processor communicates with the outside world, it provides data in byte sized chunks. Computers transfer data in two ways: parallel and serial. In parallel data transfers, often more lines are used to transfer data to a device and 8 bit data path is expensive. The serial communication transfer uses only a single data line instead of the 8 bit data line of parallel communication which makes the data transfer not only cheaper but also makes it possible for two computers located in two different cities to communicate over telephone.

Serial data communication uses two methods, asynchronous and synchronous. The synchronous method transfers data at a time while the asynchronous transfers a single byte at a time. There are some special IC chips made by many manufacturers for data communications. These chips are commonly referred to as UART (universal asynchronous receiver-transmitter) and USART (universal synchronous asynchronous receiver transmitter). The AT89C51 chip has a built in UART.

In asynchronous method, each character is placed between start and stop bits. This is called framing. In data framing of asynchronous communications, the data, such as ASCII characters, are packed in between a start and stop bit. We have a total of 10 bits for a character: 8 bits for the ASCII code and 1 bit each for the start and stop bits. The rate of serial data transfer communication is stated in bps or it can be called as baud rate.

To allow the compatibility among data communication equipment made by various manufacturers, and interfacing standard called RS232 was set by the Electronics industries Association in 1960. Today RS232 is the most widely used I/O interfacing standard. This standard is used in PCs and numerous types of equipment. However, since the standard was set long before the advent of the TTL logic family, its input and output voltage levels are not TTL compatible. In RS232, a 1 bit is represented by -3 to -25V, while a 0 bit is represented +3 to +25 V, making -3 to +3 undefined. For this reason, to connect any RS232 to a microcontroller system we must use voltage converters such as MAX232 to connect the TTL logic levels to RS232 voltage levels and vice versa. MAX232 ICs are commonly referred to as line drivers.

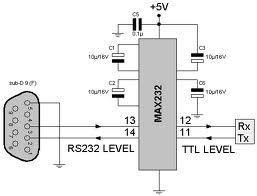
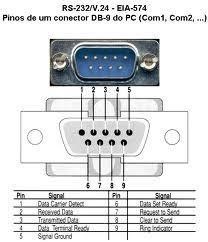


Fig 2.4.1 Serial communication

The RS232 cables are generally referred to as DB-9 connector. In labeling, DB-9P refers to the plug connector (male) and DB-9S is for the socket connector (female). The simplest connection between a PC and microcontroller requires a minimum of three pin, TXD, RXD, and ground. Many of the pins of the RS232 connector are used for handshaking signals. They are bypassed since they are not supported by the UART chip.



IBM PC/ compatible computers based on x86(8086, 80286, 386, 486 and Pentium) microprocessors normally have two COM ports. Both COM ports have RS232 type connectors. Many PCs use one each of the DB-25 and DB-9 RS232 connectors. The COM ports are designated as COM1 and COM2. We can connect the serial port to the COM 2 port of a PC for serial communication experiments. We use a DB9 connector in our arrangement.

2.5 LCD Module

To display interactive messages we are using LCD Module. We examine an intelligent LCD display of two lines,16 characters per line that is interfaced to the controllers. The protocol (handshaking) for the display is as shown. Whereas D0 to D7th bit is the Data lines, RS, RW and EN pins are the control pins and remaining pins are +5V, -5V and GND to provide supply. Where RS is the Register Select, RW is the Read Write and EN is the Enable pin.

The display contains two internal byte-wide registers, one for commands (RS=0) and the second for characters to be displayed (RS=1). It also contains a userprogrammed RAM area (the character RAM) that can be programmed to generate any desired character that can be formed using a dot matrix. To distinguish between these two data areas, the hex command byte 80 will be used to signify that the display RAM address 00h will be chosen.Port1 is used to furnish the command or data type, and ports 3.2 to3.4 furnish register select and read/write levels.

The display takes varying amounts of time to accomplish the functions as listed. LCD bit 7 is monitored for logic high (busy) to ensure the display is overwritten.

Liquid Crystal Display also called as LCD is very helpful in providing user interface as well as for debugging purpose. The most common type of LCD controller is HITACHI 44780 which provides a simple interface between the controller & an LCD. These LCD's are very simple to interface with the controller as well as are cost effective.



Fig 2.5.1 2x16 Line Alphanumeric LCD Display

The most commonly used *ALPHANUMERIC*displays are *1x16* (Single Line & 16 characters), *2x16* (Double Line & 16 character per line) & *4x20*(four lines & Twenty characters per line).

The LCD requires 3 control lines (RS, R/W & EN) & 8 (or 4) data lines. The number on data lines depends on the mode of operation. If operated in 8-bit mode then 8 data lines + 3 control lines i.e. total 11 lines are required. And if operated in 4-bit mode then 4 data lines + 3 control lines i.e. 7 lines are required. How do we decide which mode to use? It’s simple if you have sufficient data lines you can go for 8 bit mode & if there is a time constrain i.e. display should be faster then we have to use 8-bit mode because basically 4-bit mode takes twice as more time as compared to 8-bit mode.

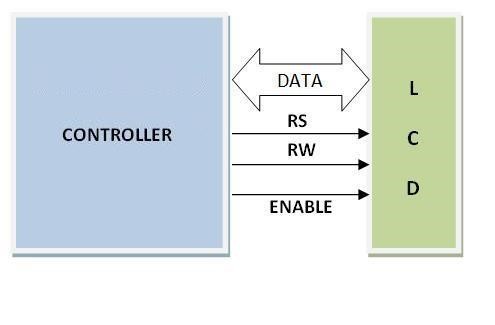
|  |  |  |
| --- | --- | --- |
| Pin | Symbol | Function |
| 1 | Vss | Ground |
| 2 | Vdd | Supply Voltage |
| 3 | Vo | Contrast Setting |
| 4 | RS | Register Select |
| 5 | R/W | Read/Write Select |
| 6 | En | Chip Enable Signal |
| 7-14 | DB0-DB7 | Data Lines |
| 15 | A/Vee | Gnd for the backlight |
| 16 | K | Vcc for backlight |

Table 2.3 Operations of LCD

When RS is low (0), the data is to be treated as a command. When RS is high (1), the data being sent is considered as text data which should be displayed on the screen.

When R/W is low (0), the information on the data bus is being written to the LCD. When RW is high (1), the program is effectively reading from the LCD. Most of the times there is no need to read from the LCD so this line can directly be connected to Gnd thus saving one controller line.

The ENABLE pin is used to latch the data present on the data pins. A HIGH - LOW signal is required to latch the data. The LCD interprets and executes our command at the instant the EN line is brought low. If you never bring EN low, your instruction will never be executed.



##### 2.5.1 COMMANDS USED IN LCD

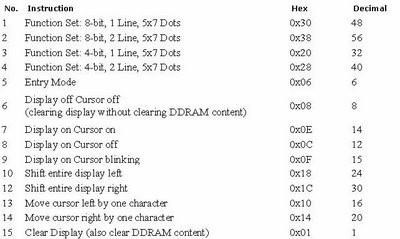


Table 2.4 Commands used in LCD

2.6 Relay Spdt



Fig 2.6.1 Relay

Overview of Relays

A relay is an electrically operated switch used to isolate one electrical circuit from another. In its simplest form, a relay consists of a coil used as an electromagnet to open and close switch contacts. Since the two circuits are isolated from one another, a lower voltage circuit can be used to trip a relay, which will control a separate circuit that requires a higher voltage or amperage. Relays can be found in early telephone exchange equipment, in industrial control circuits, in car audio systems, in automobiles, on water pumps, in high-power audio amplifiers and as protection devices.

Relay Switch Contacts

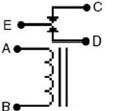
The switch contacts on a relay can be "normally open" (NO) or "normally closed" (NC)that is, when the coil is at rest and not energized (no current flowing through it), the switch contacts are given the designation of being NO or NC. In an open circuit, no current flows, such as a wall light switch in your home in a position that the light is off. In a closed circuit, metal switch contacts touch each other to complete a circuit, and current flows, similar to turning a light switch to the "on" position. In the accompanying schematic diagram, points A and B connect to the coil. Points C and D connect to the



switch. When you apply a voltage across the coil at points A and B, you create an electromagnetic field, which attracts a lever in the switch, causing it to make or break contact in the circuit at points C and D (depending if the design is NO or NC). The switch contacts remain in this state until you remove the voltage to the coil. Relays come in different switch configurations. The switches may have more than one "pole," or switch contact. The diagram shows a "single pole single throw" configuration, referred to as SPST. This is similar to a wall light switch in your home. With a single "throw" of the switch, you close the circuit.

The Single Pole Double Throw Relay

A single pole double throw (SPDT) relay configuration switches one common pole to two other poles, flipping between them. As shown in the schematic diagram, the common point E completes a circuit with C when the relay coil is at rest, that is, no voltage is applied to it.



This circuit is "closed." A gap between the contacts of point E and D creates an "open" circuit. When you apply power to the coil, a metal level is pulled down, closing the circuit between points E and D and opening the circuit between E and C. A single pole double throw relay can be used to alternate which circuit a voltage or signal will be sent

SPDT Relay:

(**S**ingle **P**ole **D**ouble **T**hrow **Relay**) an electromagnetic switch, consist of a coil (terminals 85 & 86), 1 common terminal (30), 1 normally closed terminal (87a), and one normally open terminal (87) (Figure 1).

When the coil of an SPDT relay (Figure 1) is at rest (not energized), the common terminal (30) and the normally closed terminal (87a) have continuity. When the coil is energized, the common terminal (30) and the normally open terminal (87) have continuity.

The diagram below center (Figure 2) shows an SPDT relay at rest, with the coil not energized. The diagram below right (Figure 3) shows the relay with the coil energized. As you can see, the coil is an electromagnet that causes the arm that is always connected to the common (30) to pivot when energized whereby contact is broken from the normally closed terminal (87a) and made with the normally open terminal (87).

When energizing the coil of a relay, polarity of the coil does not matter unless there is a diode across the coil. If a diode is not present, you may attach positive voltage to either terminal of the coil and negative voltage to the other, otherwise you must connect positive to the side of the coil that the cathode side (side with stripe) of the diode is connected and negative to side of the coil that the anode side of the diode is connected.

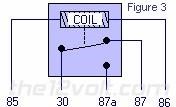
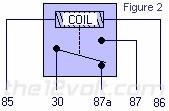
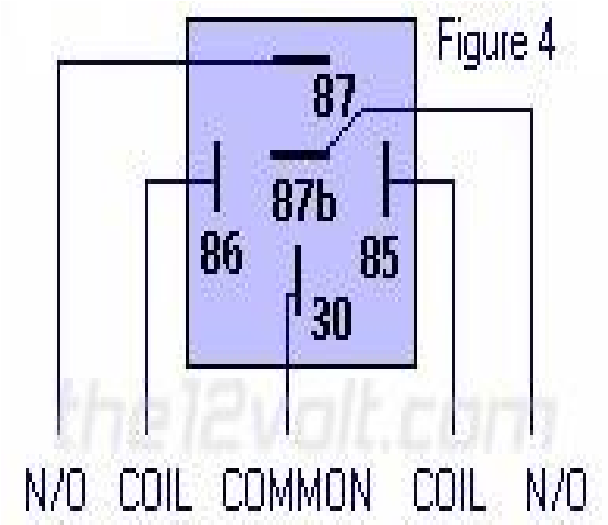


Fig 2.6.2 Working of relay

Why do I want to use a relay and do I really need to?

Anytime you want to switch a device which draws more current than is provided by an output of a switch or component you'll need to use a relay. The coil of an **SPDT** or an **SPST** relay that we most commonly use draws very little current (less than 200 milliamps) and the amount of current that you can pass through a relay's common, normally closed, and normally open contacts will handle up to 30 or 40 amps. This allows you to switch devices such as headlights, parking lights, horns, etc., with low amperage outputs such as those found on keyless entry and alarm systems, and other components. In some cases you may need to switch multiple things at the same time using one output. A single output connected to multiple relays will allow you to open continuity and/or close continuity simultaneously on multiple wires.

There are far too many applications to list that require the use of a relay, but we do show many of the most popular applications in the pages that follow and many more in our Relay Diagrams - Quick Reference application. If you are still unclear about what a relay does or if you should use one after you browse through the rest of this section, please post a question in the12volt's install bay. (We recommend Tyco (formerly Bosch) or Potter & Brumfield relays for all of the SPDT and SPST relay applications shown on this site.)

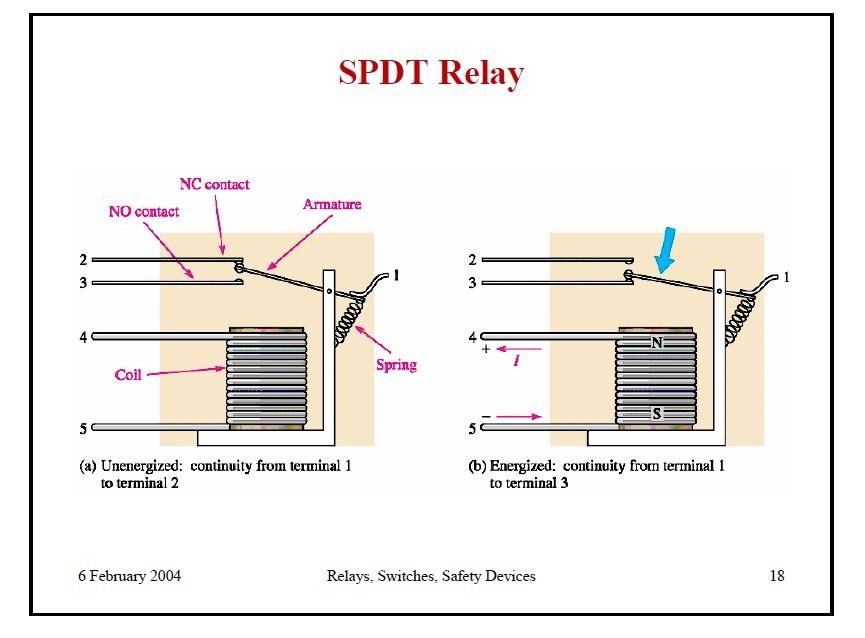


Fig 2.6.3 SPDT relay

2.7 ULN2003



Fig 2.7.1 ULN2003

ULN is mainly suited for interfacing between low-level circuits and multiple peripheral power loads,.The series ULN20XX high voltage, high current darlington arrays feature continuous load current ratings. The driving circuitry in- turn decodes the coding and conveys the necessary data to the stepper motor, this module aids in the movement of the arm through steppers

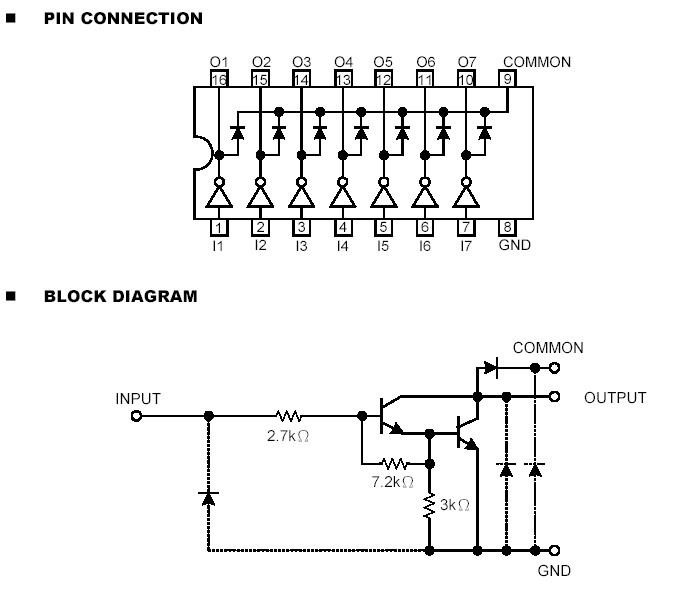


Fig 2.7.2 Block and Pin Diagram

* The driver makes use of the ULN2003 driver IC, which contains an array of 7 power Darlington arrays, each capable of driving 500mA of current. At an approximate duty cycle, depending on ambient temperature and number of drivers turned on, simultaneously typical power loads totaling over 230w can be controlled.
* The device has base resistors, allowing direct connection to any common logic family. All the emitters are tied together and brought out to a separate terminal. Output protection diodes are included; hence the device can drive inductive loads with minimum extra components. Typical loads include relays, solenoids, stepper motors, magnetic print hammers, multiplexed LED, incandescent displays and heaters.

##### 2.7.1 Darlington Pair

* A Darlington pair is two transistors that act as a single transistor but with a much higher current gain.
* What is current gain?
* Transistors have a characteristic called current gain. This is referred to as its hFE. The amount of current that can pass through the load when connected to a transistor that is turned on equals the input current x the gain of the transistor (hFE) The current gain varies for different transistor and can be looked up in the data sheet for the device. Typically it may be 100. This would mean that the current available to drive the load would be 100 times larger than the input to the transistor.

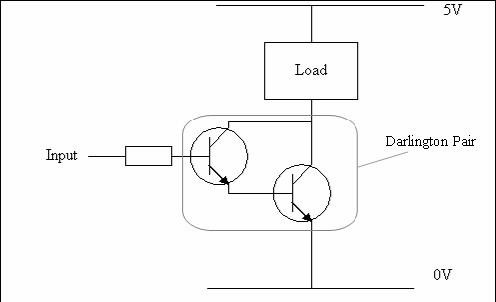


Fig 2.7.1 Darlington Pair

Why use a Darlington Pair?

In some application the amount of input current available to switch on a transistor is very low. This may mean that a single transistor may not be able to pass sufficient current required by the load.

As stated earlier this equals the input current x the gain of the transistor (hFE). If it is not be possible to increase the input current then we need to increase the gain of the transistor. This can be achieved by using a Darlington Pair.

A Darlington Pair acts as one transistor but with a current gain that equals:

Total current gain (hFE total) = current gain of transistor 1 (hFE t1) x current gain of transistor 2 (hFE t2)

So for example if you had two transistors with a current gain (hFE) = 100:

(hFE total) = 100 x 100

(hFE total) = 10,000

You can see that this gives a vastly increased current gain when compared to a single transistor. Therefore this will allow a very low input current to switch a much bigger load current.

Base Activation Voltage

Normally to turn on a transistor the base input voltage of the transistor will need to be greater that 0.7V. As two transistors are used in a Darlington Pair this value is doubled. Therefore the base voltage will need to be greater than 0.7V x 2 = 1.4V.

It is also worth noting that the voltage drop across collector and emitter pins of the Darlington Pair when the turn on will be around 0.9V Therefore if the supply voltage is 5V (as above) the voltage across the load will be will be around 4.1V (5V – 0.9V)

2.8 ESP 8266:

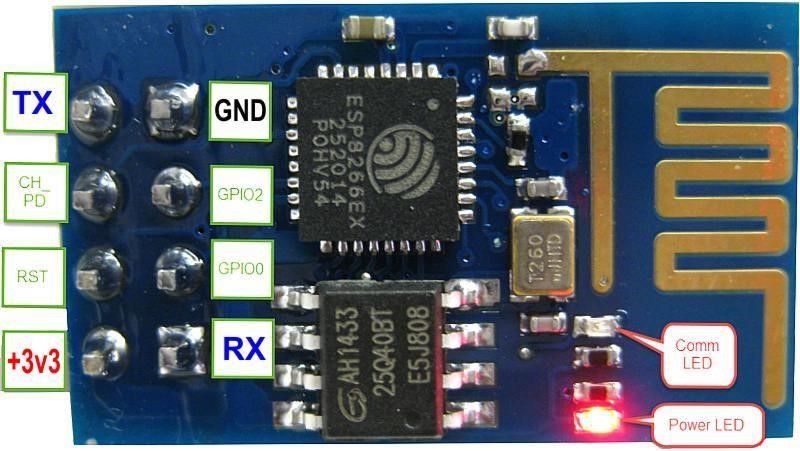


Fig 2.8.1 ESP 8266

2.8.1 ESP8266 Features

* 802.11 b/g/n protocol
* Wi-Fi Direct (P2P), soft-AP
* Integrated TCP/IP protocol stack
* Integrated TR switch, balun, LNA, power amplifier and matching network
* Integrated PLL, regulators, and power management units
* +19.5dBm output power in 802.11b mode
* Integrated temperature sensor
* Supports antenna diversity
* Power down leakage current of < 10uA
* Integrated low power 32-bit CPU could be used as application processor
* SDIO 2.0, SPI, UART
* STBC, 1×1 MIMO, 2×1 MIMO
* A-MPDU & A-MSDU aggregation & 0.4µs guard interval
* Wake up and transmit packets in < 2ms
* Standby power consumption of < 1.0mW (DTIM3)

The ESP8266 is a low-cost Wi-Fi chip with full TCP/IP stack and MCU(Microcontroller Unit) capability produced by Shanghai-based Chinese manufacturer,

The chip first came to the attention of western makersin August 2014 with the ESP-01 module, made by a third-party manufacturer, AI-Thinker. This small module allows microcontrollers to connect to a Wi-Fi network and make simple TCP/IP connections using Hayes-style commands. However, at the time there was almost no Englishlanguage documentation on the chip and the commands it accepted. The very low price and the fact that there were very few external components on the module which suggests that it could eventually be very inexpensive in volume, attracted many hackers to explore the module, chip, and the software on it, as well as to translate the Chinese documentation.

The ESP8285 is an ESP8266 with 1 MB of built-in flash, allowing for single-chip devices capable of connecting to Wi-Fi. The successor to these module(s) is ESP32 This is the series of ESP8266-based modules made by Espressif.

##### Active Form Dimension Name Pitch LED Antenna Shielded? Notes pins factor s (mm)

ESP-FCC ID

WRO2AC7Z-

|  |  |  |
| --- | --- | --- |
| No | PCB trace | Yes |

18 0.1" 2×9 DIL 18 × 20

OM-ESPWR

02 OOM02

In the table above (and the two tables which follow), "Active pins" include the GPIO and ADC pins with which you can attach external devices to the ESP8266 MCU. The "Pitch" is the space between pins on the ESP8266 module, which is important to know if you are going to breadboard the device. The "Form factor" also describes the module packaging as "2 x 9 DIL", meaning two rows of 9 pins arranged "Dual In Line", like the pins of DIP ICs. Many ESP-xx modules include a small on-board LED which can be programmed to blink and thereby indicate activity. There are several antenna options for ESP-xx boards including a trace antenna, an on-board ceramic antenna, and an external connector which allows you to attach an external Wi-Fi antenna. Since Wi-Fi communications generates a lot of RFI (Radio Frequency Interference), governmental bodies like the FCC like shielded electronics to minimize interference with other devices. Some of the ESP-xx modules come housed within a metal box with an FCC seal of approval stamped on it. First and second world markets will likely demand FCC approval and shielded Wi-Fi devices.

AI-Thinker modules



Fig 2.8.2 ESP-01 module

These are the first series of modules made with the ESP8266 by the third-party manufacturer *AI-Thinker* and remain the most widely available. They are collectively referred to as "ESP-xx modules". To form a workable development system they require additional components, especially a serial TTL-to-USB adapter (sometimes called a USB-to-UART bridge) and an external 3.3 Volt power supply. Novice ESP-8266 developers are encouraged to consider larger ESP8266 Wi-Fi development boards like the Node MCU which includes the USB-to-UART bridge and a Micro-USB connector coupled with a 3.3 Volt power regulator already built into the board. When project development is complete, you may not need these components and can consider using these cheaper ESP-xx modules as a lower power, smaller footprint option for your production runs This is the series of ESP8266-based modules made by Espressif

**Active Form Dimensions**

**Name Pitch LEDs Antenna Shielded?**

##### pins factor (mm)

|  |  |  |
| --- | --- | --- |
| No | PCB  trace | Yes |

ESP- 18 0.1" 2×9 DIL 18

× 20 WROOM-02

In the table above (and the two tables which follow), "Active pins" include the GPIO and ADC pins with which you can attach external devices to the ESP8266 MCU. The "Pitch" is the space between pins on the ESP8266 module, which is important to know if you are going to breadboard the device. The "Form factor" also describes the module packaging as "2 x 9 DIL", meaning two rows of 9 pins arranged "Dual In Line", like the pins of DIP ICs. Many ESP-xx modules include a small on-board LED which can be programmed to blink and thereby indicate activity. There are several antenna options for ESP-xx boards including a trace antenna, an on-board ceramic antenna, and an external connector which allows you to attach an external Wi-Fi antenna. Since Wi-Fi communications generates a lot of RFI (Radio Frequency Interference), governmental bodies like the FCC like shielded electronics to minimize interference with other devices. Some of the ESP-xx modules come housed within a metal box with an FCC seal of approval stamped on it. First and second world markets will likely demand FCC approval and shielded Wi-Fi devices.

These are the first series of modules made with the ESP8266 by the third-party manufacturer *AI-Thinker* and remain the most widely available. They are collectively referred to as "ESP-xx modules". To form a workable development system they require additional components, especially a serial TTL-to-USB adapter (sometimes called a USB-to-UART bridge) and an external 3.3 Volt power supply. Novice ESP-8266 developers are encouraged to consider larger ESP8266 Wi-Fi development boards like the Node MCU which includes the USB-to-UART bridge and a Micro-USB connector coupled with a 3.3 Volt power regulator already built into the board. When project development is complete, you may not need these components and can consider using these cheaper ESP-xx modules as a lower power, smaller footprint option for your production runs. ESP8266 offers a complete and self-contained Wi-Fi networking solution, allowing it to either host the application or to offload all Wi-Fi networking functions from another application processor.

When ESP8266 hosts the application, and when it is the only application processor in the device, it is able to boot up directly from an external flash. It has integrated cache to improve the performance of the system in such applications, and to minimize the memory requirements.Alternately, serving as a Wi-Fi adapter, wireless internet access can be added to any microcontroller-based design with simple connectivity through UART interface or the CPU AHB bridge interface.

The popularity of many of these "other boards" over the earlier ESP-xx modules is the inclusion of an on-board USB-to-UART bridge (like the Silicon Labs' CP2102 or the WCH CH340G) and a Micro-USB connector coupled with a 3.3 Volt regulator to provide both power to the board and connectivity to the host (software development) computer commonly referred to as the console. With earlier ESP-xx modules, these two items (the USB-to-Serial adaptor and a 3.3 Volt regulator) had to be purchased separately and be wired into the ESP-xx circuit. Modern ESP8266 boards like the Node MCU boards are a lot less painful and offer more GPIO pins to play with. Most of these "other boards" are based on the ESP-12E module, but new modules are being introduced seemingly every few months.

### Chapter 3 Circuit Diagram:



Chapter 4

Source Code:

#include <LiquidCrystal.h>

#define TIMEOUT 2000

int R1 = 8; int R2 = 9; int R3 = 10;

LiquidCrystal lcd(7, 6, 5, 4, 3, 2); unsigned int Voltage; uint8\_t fault=0; void title(void)

{ lcd.clear(); lcd.print(" UNDER GROUND "); lcd.setCursor(0,1);

lcd.print("FAULT DETECTION ");

} char s; void setup() { int l;

pinMode(R1,O UTPUT); digitalWrite(R1,LOW); pinMode(R2,OUTPUT); digitalWrite(R2,LOW); pinMode(R3,OUTPUT); digitalWrite(R3,LOW); lcd.begin(16, 2); lcd.setCursor(0, 0); lcd.print("LCD READY ... "); delay(1000); title(); delay(2000);

//Serial.begin(9600);

Serial.begin(115200);

SendCommand("AT+RST", "Ready");

SendCommand("AT+CWMODE=2","OK");

SendCommand("AT+CWSAP="undercable\_001","12345678",5,3","O

K");

SendCommand("AT+CIPMUX=1","OK");

} int cntr=1000; void loop() {

if(cntr++ >= 1000) { cntr = 0;

Voltage = (analogRead(A0)); lcd.clear(); lcd.print("Voltage: "); lcd.setCursor(8,0); lcd.print(Voltage); lcd.setCursor(0,1); //delay(1000); fault = 0; if(Voltage < 75){ digitalWrite(R1,LOW); digitalWrite(R2,LOW); digitalWrite(R3,LOW); lcd.print(" No Fault ");

// delay(1000);

} else if((Voltage >= 75) && (Voltage <= 85)){

digitalWrite(R1,HIGH); digitalWrite(R2,LOW); digitalWrite(R3,LOW); lcd.print(" R Fault :

1Km ");

// delay(1000); fault |= 1<<0;

}

else if((Voltage >= 95) && (Voltage <= 105)){//93,100 digitalWrite(R1,LOW); digitalWrite(R2,HIGH); digitalWrite(R3,LOW); lcd.print(" Y Fault : 2Km "); /delay(1000); fault |= 1<<1;

} else if((Voltage >= 115) && (Voltage <= 119)){ digitalWrite(R1,LOW); digitalWrite(R2,LOW); digitalWrite(R3,HIGH); lcd.print(" B Fault :

3Km ");

//delay(1000); fault |= 1<<2;

}

else if((Voltage >= 120) && (Voltage <= 125)){ digitalWrite(R1,HIGH); digitalWrite(R2,HIGH); digitalWrite(R3,LOW); lcd.print("RY Fault:1Km,2Km");

// delay(1000); fault |= (1<<0 | 1<<1);

}

else if((Voltage >= 248) && (Voltage <= 255)){ digitalWrite(R1,LOW); digitalWrite(R2,HIGH); digitalWrite(R3,HIGH); lcd.print("YB Fault:2Km,3Km");

// delay(1000); fault |= (1<<1 | 1<<2);

}

else if((Voltage >= 153) && (Voltage <= 160)){ digitalWrite(R1,HIGH);

digitalWrite(R2,LOW); digitalWrite(R3,HIGH);

lcd.print("RB Fault:1Km,3Km");

// delay(1000); fault |= (1<<0 | 1<<2);

} else if(Voltage > 490){ digitalWrite(R1,HIGH); digitalWrite(R2,HIGH); digitalWrite(R3,HIGH); lcd.print("RYBFault:1K,2K,3K");

// delay(1000); fault |= (1<<0 | 1<<1 | 1<<2);

}

}

if(Serial.find((char \*)"+IPD,"))

{

char webpage[1024]; delay(100); int connectionId = Serial.read()-48; /\* We are subtracting 48 from the output because the read() function returns the ASCII decimal value and the first decimal number which is 0 starts at 48\*/ int indx = 0;

strcpy(webpage+indx,"<h1>Underground Cable Fault Checking System</h1><html>"); strcpy(webpage+indx,"<meta http-equiv=\"refresh\"

content=\"10\">"); indx = strlen(webpage); if(fault == 0) { strcpy(webpage+indx,"<body><center><h2>No Cable Fault Detected"); indx = strlen(webpage);

}

else

{

strcpy(webpage+indx,"<body><center><h2>Cable Fault Detected at ");

indx = strlen(webpage);

if(fault & 1<<0) strcpy(webpage+indx,"1Km Distance");

indx = strlen(webpage);

if(fault & 1<<1) { if(fault & 1<<0) strcpy(webpage+indx,", "); indx = strlen(webpage); strcpy(webpage+indx,"2Km Distance"); indx = strlen(webpage);

}

if(fault & 1<<2)

{

if((fault & 1<<0) || (fault & 1<<0)) strcpy(webpage+indx,"and "); indx

= strlen(webpage); strcpy(webpage+indx,"3Km Distance"); indx = strlen(webpage);

}

}

strcpy(webpage+indx,"</h2</body></html>"); indx = strlen(webpage); indx = strlen(webpage)

String closeCommand = "AT+CIPCLOSE="; closeCommand+=connectionId; // append connection id closeCommand+="\r\n";

Serial.print("AT+CIPCLOSE=");

Serial.println(connectionId);

}

} } char serial\_rx(void) { while(Serial.available() == 0); return Serial.read();

} char serial1\_rx(void)

{

while(Serial.availabl e() == 0);

return

Serial.read(); } void serial\_tx(char ch)

{

Serial.print(ch);

} void serial\_str(char \*str)

{

Serial.print(str);

} void lcd\_str(char l,char p,char \*str)

{ if(l == 1) lcd.setCursor(p-1,0); else if(l == 2) lcd.setCursor(p-1,1); while(\*str != '\0'){ lcd.print(\*str); str++;

}

}

boolean SendCommand(String cmd, String ack){ Serial.println(cmd); // Send "AT+" command to module if (!echoFind(ack)) // timed out waiting for ack string return true; // ack blank or ack found

} boolean echoFind(String keyword) { byte current\_char = 0; byte keyword\_length =keyword.length(); long deadline = millis() + TIMEOUT; while(millis() < deadline)

{ if (Serial.available()) { char ch = Serial.read(); //Serial.write(ch); if (ch == keyword[current\_char]) if (++current\_char == keyword\_length)

{

//Serial.println(); return true;

}

}

}

}

### Chapter 5

**1.1** Advantages:

1. The main use of this project is CABLE FAULT DETCTING IN UNDER

GROUNDS

1. Low Cost
2. Less Complexity
3. Long Distance Applications

* 1. Disadvantages:

This System Can’t Find The particular point.

* 1. Applications :

1. Industrial Applications.
2. Ground Cable Fault Detection Application.
3. Electrical Cable fault detection Application.

5.4 Conclusion:

The project “UNDERGROUND CABLE FAULT DETECTION ” has been successfully designed and tested. Integrating features of all the hardware components used have developed it. Presence of every module has been reasoned out and placed carefully thus contributing to the best working of the unit. Secondly, using highly advanced IC’s and with the help of growing technology the project has been successfully implemented.

### 5.5 Final view of kit

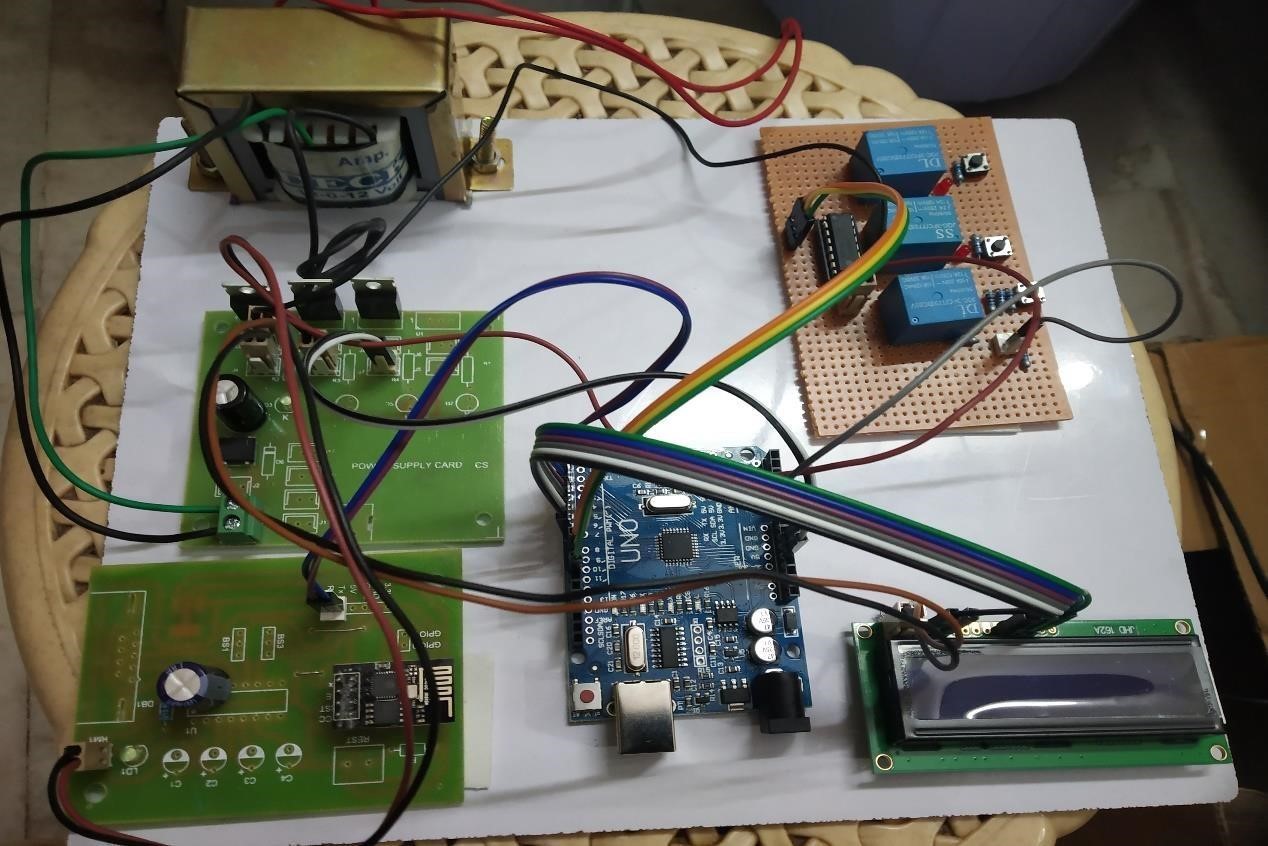


Fig 5.5.1: Overall view

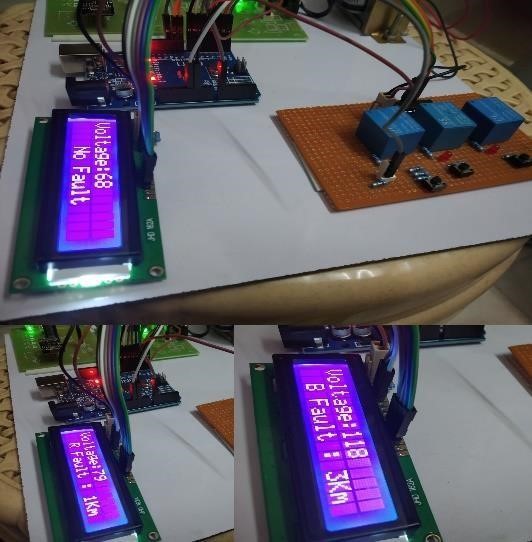
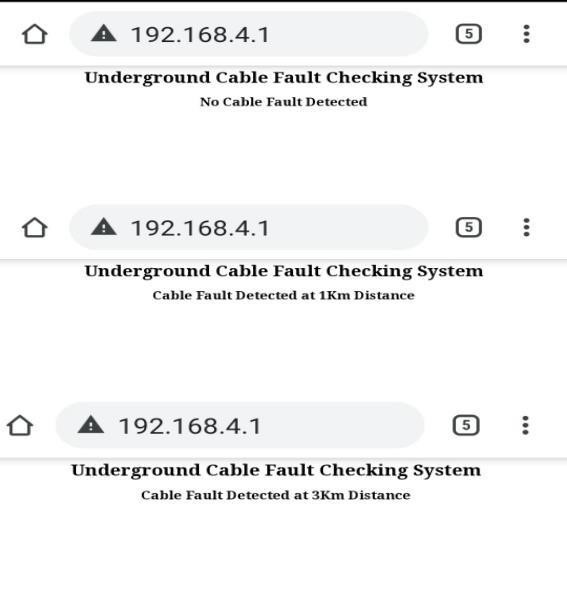
 

Fig 5.5.2: Fault Display on LCD Fig 5.5.3: Fault Display on Web Page

### References

Bibilography

1. WWW.MITEL.DATABOOK.COM
2. WWW.ATMEL.DATABOOK.COM
3. WWW.FRANKLIN.COM
4. WWW.KEIL.COM

*en.wikipedia.org/wiki/****ZigBee***

[***WWW.ZIGBEE.ORG/*** *w*](http://www.zigbee.org/)*ww.nxp.com/documents/user\_manual/UM1 http://www.futurlec.com/GPS.shtml013*

[***HTTP://EN.WIKIPEDIA.ORG/WIKI/GLOBAL\_POSITIONING\_SYSTEM9.PDF***](http://en.wikipedia.org/wiki/Global_Positioning_System9.pdf) <http://electronics.howstuffworks.com/gadgets/travel/gps.htm><http://en.wikipedia.org/wiki/GSM><http://burnsidetelecom.com/whitepapers/gsm.pdf>

<http://www.itu.int/osg/spu/ni/3G/casestudies/GSM-FINAL.pdf>

References

* 1. ARM-systemonchip-architecture by Steve furber.
  2. ARM-user manual UM10114.
  3. ARM System Developers Guide by Andrew N.SLOSS
  4. "Power Electronics” by M D Singh and K B Khanchandan
  5. "Linear Integrated Circuits” by D Roy Choudary & Shail Jain
  6. "Electrical Machines” by S K Bhattacharya
  7. "Electrical Machines II” by B L Thereja
  8. www.8051freeprojectsinfo.com