CHAPTER 1

INTRODUCTION OF PROJECT

1.1 INTRODUCTION TO PROJECT

Building Automation Systems is a set of technologies that results in operation of machines and systems without significant human intervention and achieves superior performance to manual operation .In the scope of Industrialization, automation is a step beyond mechanization. whereas mechanization provided human operators with machinery to assist them with the muscular requirements of work, automation greatly decrease the need for human sensory and mental requirements as well .

1.2 INTRODUCTION TO EMBEDDED SYSTEMS

Each day, our lives become more dependent on 'embedded systems', digital information technology that is embedded in our environment. More than 98% of processors applied today are in embedded systems, and are no longer visible to the customer as 'computers' in the ordinary sense. An Embedded System is a special-purpose system in which the computer is completely encapsulated by or dedicated to the device or system it controls. Unlike a general-purpose computer, such as a personal computer, an embedded system performs one or a few pre-defined tasks, usually with very specific requirements. Since the system is dedicated to specific tasks, design engineers can optimize it, reducing the size and cost of the product. Embedded systems are often mass- produced, benefiting from economies of scale. The increasing use of PC hardware is one of the most important developments in high-end embedded systems in recent years. Hardware costs of high-end systems have dropped dramatically as a result of this trend, making feasible some projects which previously would not have been done because of the high cost of non-PC-based embedded hardware. But software choices for the embedded PC platform are not nearly as attractive as the hardware. Typically, an embedded system is housed on a single microprocessor board with the programs stored in ROM. Virtually all appliances that have a digital interface -- watches, microwaves, VCRs, cars -- utilize embedded systems. Some embedded systems include an operating system, but many are so specialized that the entire logic can be implemented as a single program. Physically, Embedded Systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic lights, factory controllers, or the systems controlling nuclear power plants. The applications software on such processors is sometimes referred to as firmware. The simplest devices consist of a single microprocessor (often called a "chip”), which may itself be packaged with other chips in a hybrid system or Application Specific Integrated Circuit

(ASIC). Its input comes from a detector or sensor and its output goes to a switch or activator which (for example) may start or stop the operation of a machine or, by operating a valve, may control the flow of fuel to an engine. As the embedded system is the combination of both software and hardware.

1.3 DEFINITION OF AN EMBEDDED SYSTEM

Embedded system is defined as, for a particular/specific application implementing the software code to interact directly with that particular hardware what we built. Software is used for providing features and flexibility.

Hardware = {Processors, ASICs, Memory...} is used for Performance (& sometimes security).There are many definitions of embedded system but all of these can be combined into a single concept. An embedded system is a special purpose computer system that is used for particular task.

1.4 EXAMPLES OF AN EMBEDDED SYSTEMS

Embedded systems are found in wide range of application areas. Originally they were used only for expensive industrial control applications, but as technology brought down the cost of dedicated processors, they began to appear in moderately expensive applications such as automobiles, communication and office equipments and television Today's embedded systems are so inexpensive that they are used in almost every electronic product in our life. Embedded systems are often designed for mass production.

1. Automatic Teller Machines
2. Cellular telephone and telephone switches
3. Computer network equipment
4. Computer printers
5. Disk drives
6. Home automation products
7. Handheld calculators
8. Household appliances
9. Medical equipment
10. Measurement equipment
11. Multifunction wrist watches
12. Multifunction printers

1.5 HISTORY OF AN EMBEDDED SYSTEMS

The first recognizably modern embedded systems was the Apollo Guidance Computer, developed by "Charles Stark Draper" at the MIT Instrumentation Laboratory. At the project inception, the Apollo guidance computer was considered the riskiest item in the Apollo project as it employed the then newly developed monolithic integrated circuits to reduce the size and weight. An early mass produced embedded system was the Autonetics D-17 guidance computer for the Minuteman missile, released in 1961.It was built from transistor logic and had a hard disk for main memory .When the Minuteman II went into production in 1966, the D-17 was replaced with a new computer that was the first high volume use of integrated circuits. This program alone reduced prices on quad and gate ICs from $1000/ each to $3/ each permitting their use in commercial products.

Since these early applications in the 1960s, embedded systems have come down in price and there has been a dramatic rise in processing power and functionality. The first microprocessor for example, the Intel 4004, was designed for calculators and other small systems but still required many external memory and support chips. In 1978 National Engineering Manufactures Association released a "standard" for programmable microcontrollers, including almost any computer based controllers, such as single board computers, numerical and event based controllers. Embedded Systems are designed to some specific task, rather than be a general-purpose computer for multitasks. Some also have real-time performances constraints that must be met, for reasons such as safety and usability; others may have low or no performance requirement, allowing the system hardware to be simplified to reduce cost.

1.6 FEATURES OF AN EMBEDDED SYSTEM

The versatility of the embedded computer system lends itself to utility in all kinds of enterprises, from the simplification of deliverable products to a reduction in costs in their development and manufacture. Complex systems with rich functionality employ special Operating systems that take into account major characteristics of embedded systems. Embedded operating systems have minimized footprint and may follow real-time operating system specifics.The special computers system is usually less powerful than general-purpose systems, although some expectations do exist where embedded systems are very powerful and complicated. Usually a low power consumption CPU with a limited amount of memory is used in embedded systems. Many embedded systems use very small operating systems; most of these provide very limited operating system capabilities.

Since the embedded system is dedicated to specific tasks, design engineers can optimize it, reducing the size and cost of the product, or increasing the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale. Some embedded systems have to operate in extreme environment conditions such as very high temperature & humidity. For high volume systems such as portable music players or mobile phones, minimizing cost is usually the primary design consideration. Engineers typically select hardware that is just “good enough” to implement the necessary functions. For low volume or prototype embedded systems, general purpose computers may be adapted by limiting the programs or by replacing the operating system with a real-time operating system.

1.7 CHARACTERSTICS OF AN EMBEDDED SYSTEM

Embedded computing systems generally exhibit rich functionality; complex functionality is usually the reason for introducing CPUs into the design. However, they also exhibit many non-functional requirements that make the task especially challenging:

- Real-time deadlines that will cause system failure if not met
- Multi-rate operation
- In many cases, low power consumption
- Low manufacturing cost, which often means limited code size.

Workstation programmers often concentrate on functionality. They may consider the performance characteristics of a few computational kernels of their software, but rarely analyze the total application. They almost never consider power consumption and manufacturing cost. The need to juggle all these requirements makes embedded system programming very challenging and is the reason why embedded system designers need to understand computer architecture.

1.8 OVERVIEW OF AN EMBEDDED SYSTEM ARCHITECHTURE

Every embedded system consists of custom-built hardware built around a Central Processing Unit (CPU). This hardware also contains memory chips onto which the software is loaded. The software residing on the memory chip is also called the ‘firmware’. The embedded system architecture can be represented as a layered architecture as shown in Fig.

The same architecture is applicable to any computer including a desktop computer. However, there are significant differences. It is not compulsory to have an operating system in every embedded system. For small appliances such as remote-control units, air conditioners, toys etc., there is no need foran operating system and you can write only the software specific to that application.

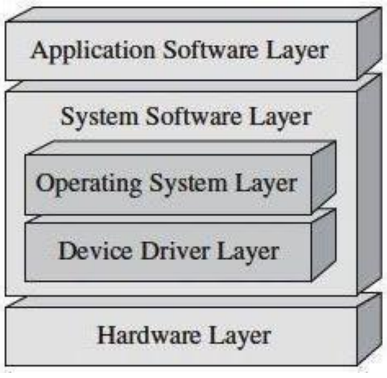


Fig 1.1 Layered Architechture of an Embedded System

For applications involving complex processing, it is advisable to have an operating system. In such a case, you need to integrate the application software with the operating system and then transfer the entire software on to the memory chip. Once the software is transferred to the memory chip, the software will continue to run *for* a long time you don't need to reload new software.

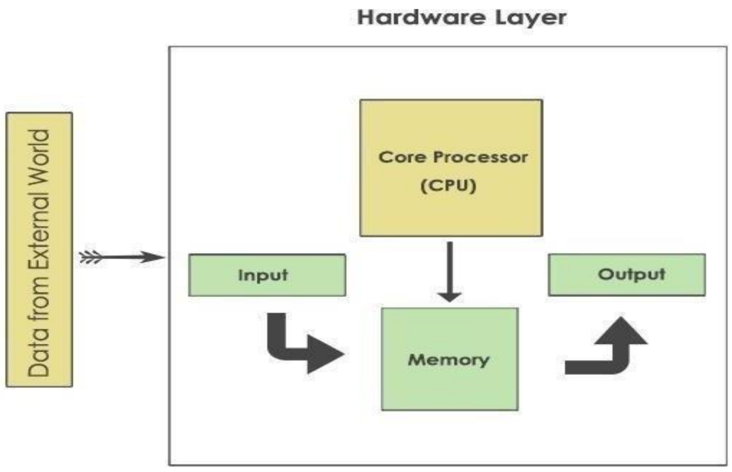


Fig 1.2 Block diagram of Embedded systems

Now, let us see the details of the various building blocks of the hardware of an embedded system. As shown in Fig. the building blocks are;

- Central Processing Unit (CPU)
- Memory (Read-only Memory and Random Access Memory)
- Input Devices
- Output devices

- Communication interface
- Application-specific circuitry

Central Processing Unit (CPU)

The Central Processing Unit (processor, in short) can be any of the following:

microcontroller, microprocessor or Digital Signal Processor (DSP). A micro-controller is a low cost processor. Its main attraction is that on the chip itself, there will be many other components such as memory, serial communication interface, analog-to digital converter etc. So, for small applications, a micro-controller is the best choice as the number of external components required will be very less. On the other hand, microprocessors are more powerful, but you need to use many external components with them. D5P is used mainly for applications in which signal processing is involved such as audio and video processing.

Memory

The memory is categorized as Random Access 11emory (RAM) and Read Only Memory (ROM). The contents of the RAM will be erased if power is switched off to the chip, whereas ROM retains the contents even if the power is switched off. So, the firmware is stored in the ROM. When power is switched on, the processor reads the ROM; the program is program is executed Input devices

Unlike the desktops, the input devices to an embedded system have very limited capability. There will be no keyboard or a mouse, and hence interacting with the embedded system is no easy task. Many embedded systems will have a small keypad-you press one key to give a specific command. A keypad may be used to input only the digits. Many embedded systems used in process control do not have any input device for user interaction; they take inputs *from* sensors or transducers 1'fnd produce electrical signals that are in turn fed to other systems.

Output devices

The output devices of the embedded systems also have very limited capability. Some embedded systems will have a *few* Light Emitting Diodes (LEDs) *to* indicate the health status of the system modules, or *for* visual indication of alarms. A small Liquid Crystal Display (LCD) may also be used to display *some* important parameters.

Communication interfaces

The embedded systems may need to, interact with other embedded systems at they may have to transmit data to a desktop. To facilitate this, the embedded systems are provided with one or a *few* communication interfaces such as RS232, RS422, RS485, Universal Serial Bus (USB), IEEE 1394, Ethernet etc.

Application-specific circuitry

Sensors, transducers, special processing and control circuitry may be required fat an embedded system, depending on its application. This circuitry interacts with the processor to carry out the necessary work. The entire hardware has to be given power supply either through the 230 volts main supply or through a battery. The hardware has to design in such a way that the power consumption is minimized.

1.9 DEBUGGING OF AN EMBEDDED SYSTEMS

Embedded debugging may be performed at different levels, depending on the facilities available. From simplest to most sophisticate they can be roughly grouped into the following areas:

- Interactive resident debugging using the simple shell provided by the embedded operating system(e.g. Forth and Basic)
- External debugging using logging or serial port output to trace operating using either a monitor in flash or using a debug server like the Remedy Debugger which even works for heterogeneous multi core systems.
- An in-circuit debugger (ICD), a hardware device that connects to the microprocessor via a JTAG or NEXUS interface. This allows the operation of the microprocessor to be controlled externally, but is typically restricted to specific debugging capabilities in the processor.
- An in- circuit emulator (ICE) replaces the microprocessor with a simulated equivalent, providing full control over all aspects of the microprocessor.

CHAPTER 2

Literature Review

IOT Technologies and Protocols

1-Domicile – An IOT based smart home automation system.

Authors :- Syed Ishmam Ahmad, Md. Lutfur Rahman

Publisher :- ICREST in 2019

We are living in the fourth industrial revolution. cost-effective smart home automation system. II. RELATED WORKS Our life is becoming more comfortable and smarter with the help of rapid upgrade of technology. Internet of things (IoT) is many researches are going on in this home automation playing a massive role in this. One of the major sides of IoT is a system. J. Saha with his coauthors presented a system on smart home. As we are in the era of never-ending growth of the Advanced IOT Based Combined Remote Health internet and its application, smart home system or home Monitoring, Home Automation and Alarm System where automation system is highly increasing to provide comfort in the authors presented a system for both remote health and life and improving the quality of life. In this project, we present home automation with alarm system. K. GB, D. Kumar, an IoT based low-cost smart home automation system. This K. Pai, Mannikandan J proposed a system called design of a system is based on a web portal which controlled by an ESP32 Wi-Fi module. Also, a custom-made private home web server is phoneme-based voice-controlled home automation system developed for maintaining the current states of home where authors researched on different type of voice signals appliances. and by using those voice signals they proposed a home automation system. A. Shinde with coauthor present a Keywords—Home Automation, IoT, ESP32 Module, Relay project on smart home automation system using IP, Module, Cloud Server

LIMITATIONS

1-Costly

2- less accuracy

3- complex circuit

2- Home automation gaining popularity

Authors :- Md. Lutfur Rahman, Mahedi Hasan

Publisher :- International Journal of Smart Home in 2017

Home automation is a topic which gaining popularity day by day, because of large advantages. One can achieve home automation by simply connecting home appliance electrical devices to the internet or cloud storage. the reason for this surge demand of network enabled home automation is reaching the zenith in recent days for its simplicity and comparable affordability. Platforms based on cloud computing help to connect to the things surroundings everyone so that one can find it easy to access anything and everything at any time and place in a user friendly manner using custom defined portals. Hence, cloud act as a front end to access IOT. Here we are assuming a system which can control devices through wireless based network or cloud based approach. In project we use IOT based home automation system which goal is to develop a home automation system that gives the user complete control over all remotely controllable aspects of his or her home. The automation system will have ability to be controlled from a central host PC, the internet, and also remotely accessed via a packet PC with a windows mobile based application. Keywords: Automation, IOT, Relay, Arduino.

LIMITATIONS

1-Costly

2- less accuracy

3-Internet of Things (IoT) based Home Automation System (HAS) Implementation of Real-Time Experiment

Authors :- Md. Lutfur Rahman, Mahedi Hasan

Publisher :- IJERT in 2021

Internet of things (IOT) is a next generation of internet. The IOT providing an easy way of life with comforts to human being by managing and interacting remotely control of home appliances. The home automation system is a new technology for control remotely by IOT technology infrastructure such as sensors, communication devices, microcontroller, NodeMCU without the interaction of human beings. The internet of things helps people live and work smarter, as well as gain complete control over their lives. In addition to offering smart devices to automate homes, IoT is essential to business. IoT provides businesses with a real-time look into how their systems really work, delivering insights into everything from the performance of machines to supply chain and logistics operations. IoT enables companies to automate processes and reduce labor costs. It also cuts down on waste and improves service delivery, making it less expensive to manufacture and deliver goods, as well as offering transparency into customer transactions.

LIMITATIONS

1-Costly

2- less accuracy

3-big in size

4- More power consumption

CHAPTER 3

METHODOLOGY

3.1 EXISTING SYSTEM

Building automation systems are the way of the future. With SMART technology, modern advancements in automation, and the flexible mobility of smartphones, managing a building's HVAC and other systems is easier than it's ever been before.

In the past, managing or owning a building was a very different role than it is today. In today's modern, technologically dependent environment, managing a building means maintaining, supervising, and dictating a building's heating, lighting, security, HVAC systems, and other tasks that can change on a day-to-day basis.

This is a fairly recent evolution though, as it wasn't that long ago when building managers simply didn't have access to the kind of information that automation systems can now provide. However, while automation is a relatively new field of technology in the public eye, the ideas that have given it life have been around for a very long time.

3.2 LIMITATIONS OF EXISTING SYSTEM

Compatability

Currently there is no international standard of compatibility for the tagging and monitoring equipment. I believe this disadvantage is the most easy to overcome the manufacturing companies of these equipment just need to agree to a standard such as Wi-Fi , USB etc. There is nothing new or innovative needed.

Reliability

A smart home will be extremely brilliant on your internet connection if your connection drops you will be left with a lot of smart products that won't work. Additionally wireless signals can possibly the interrupted by each electronics in your home and cause some of your smart products to functions slowly or not at all

3.3 PROPOSED SYSTEM

The proposed model of home automation system contains server, sensors, relay modules, light appliances and microcontrollers. The hind-end server will be setup to controlling, monitoring of the sensor devices. The proposed home automation system will be remotely control by wireless technological communication devices like smart phones, tabs and other wireless devices remotely through Internet. In this proposed building automation system can be control, managed remotely of automatic lights on and off, automatic entry of any person in specified building or hotel detected by sensors, etc. are automatically control and managed by building automation system. The proposed without interacting of human being the building automation system monitor as well as control , lights on & off system by IoT related communication devices. The NodeMCU is a brain of this system and executing various processes for home appliances system. NodeMCU connect, communicate with various sensors are gathered real-time information for building automation system.

The NodeMCU (Node Micro Controller Unit) is a open source contains software and hardware that built-up very less expensive system designed on chip known as ESP8266. The development board equips the ESP-12E module containing ESP-32 chip having TensilicaXtensa® 32-bit LX106 RISC microprocessor which operates at 80 to 160 MHz adjustable clock frequency and supports RTOS. Data and information received to the Internet and ultimately the results viewed by end user for visibility. Also we proposed relay module, 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. Also we proposed IR sensors here, IR sensor is an electronic device, that emits the light in order to sense some object of the surroundings. An IR sensor can measure the heat of an object as well as detects the motion. Usually, in the infrared spectrum, all the objects radiate some form of thermal radiation. These types of radiations are invisible to our eyes, but infrared sensor can detect these radiations. Proposed System Model The building automation system especially managing the house hold appliances remotely for convenient to human being. This system contains for notifying occur any violation for providing security and violating dangerous things will not be happened in the home like. And also there is a alert SMS to user mobile or e-mail can be sent to the concerned user for alert for safety in home .

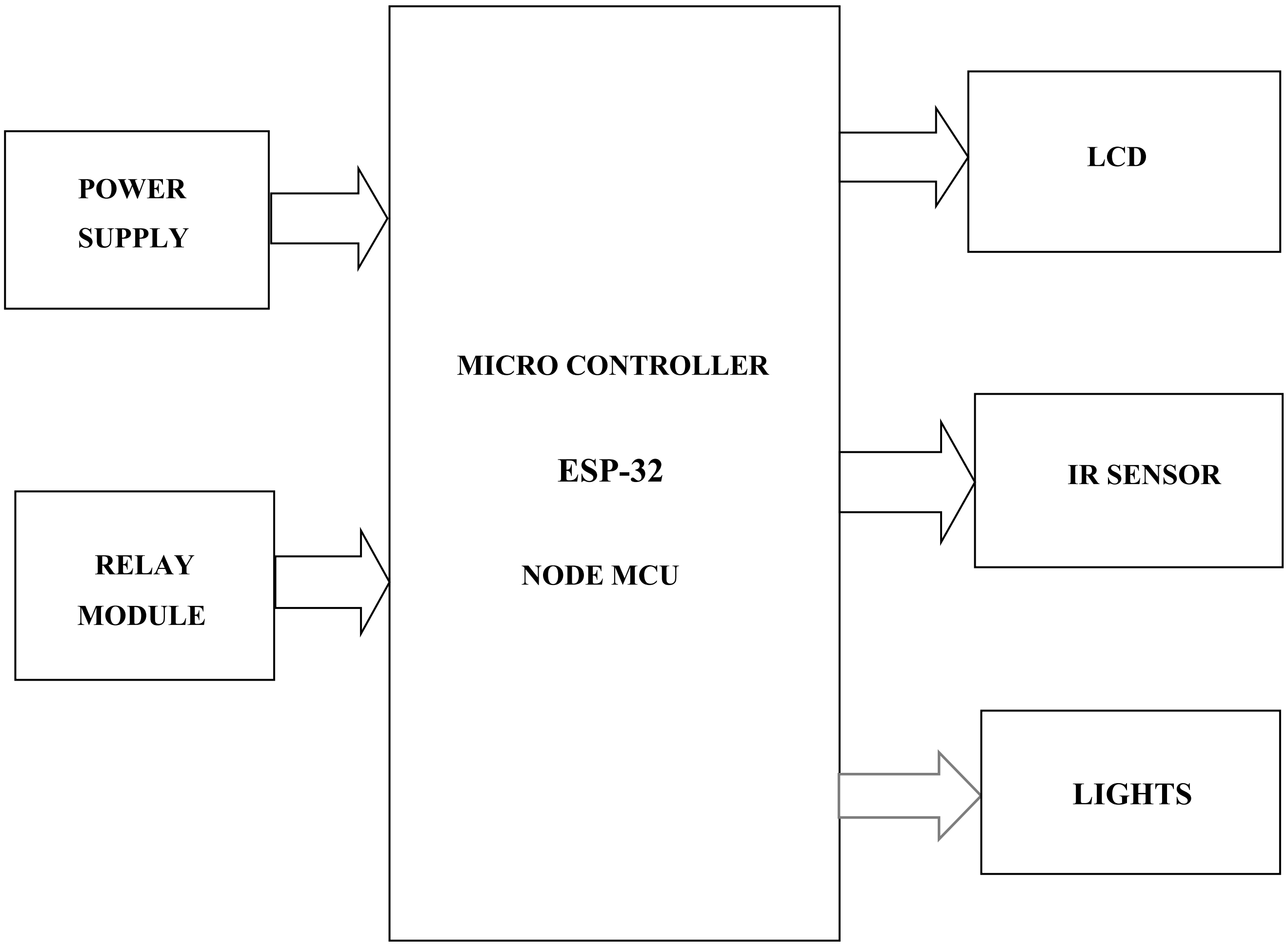
3.4 MOTIVATION

As we are engineering students from electronics communications engineering, we just want to introduce the technology IOT (Internet Of Things) in our day to day life. We have been seeing that the world has been changing a lot from old mechanism to a modern world . Every where we see is going to be in digital. So lets make thought why not we use this IOT module as a digital creator. So to improve and to make our work easy we are introducing this project.

CHAPTER 4

BLOCK DIAGRAM AND SCHEMATIC OF THE PROJECT

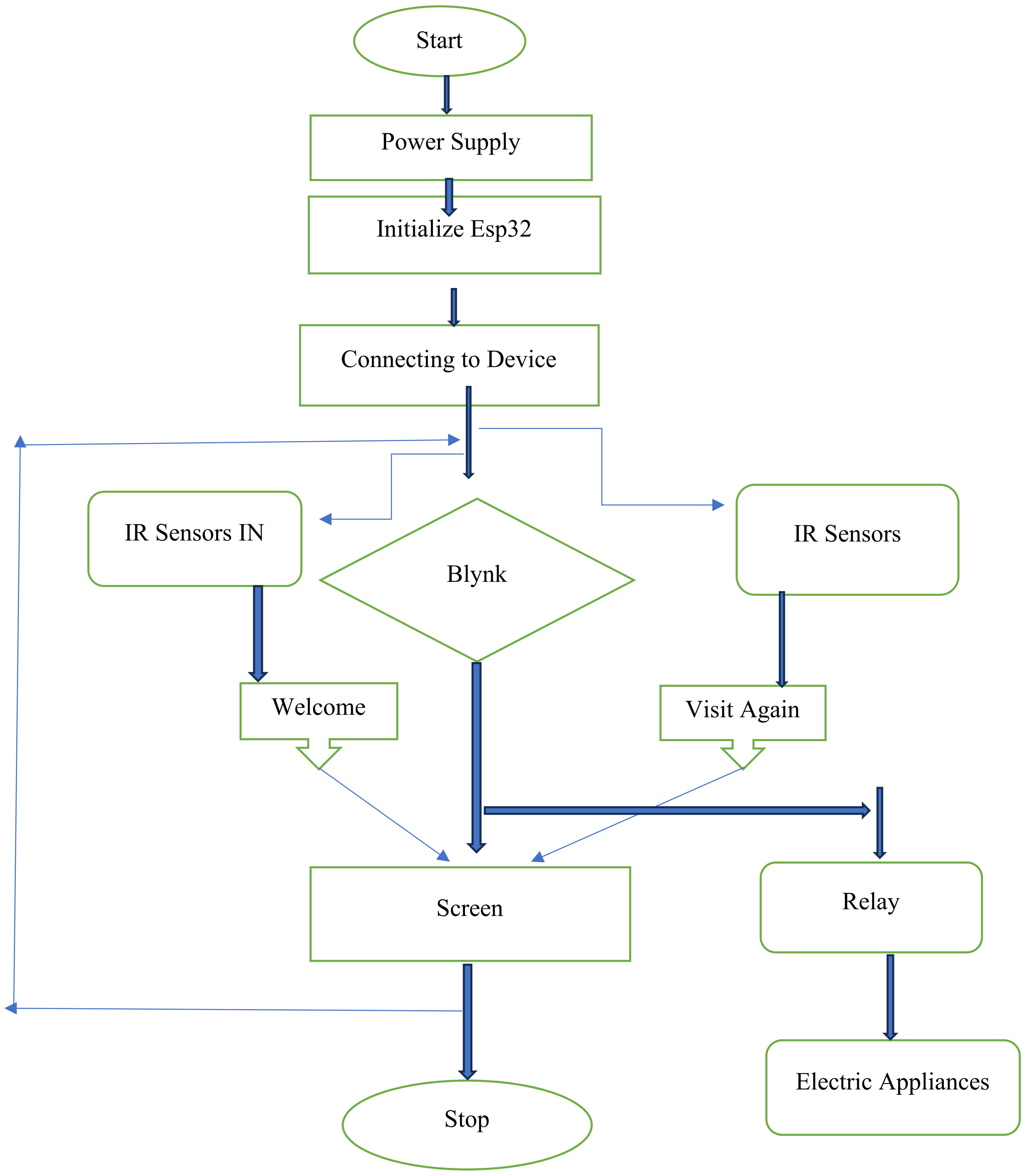
4.1 BLOCK DIAGRAM



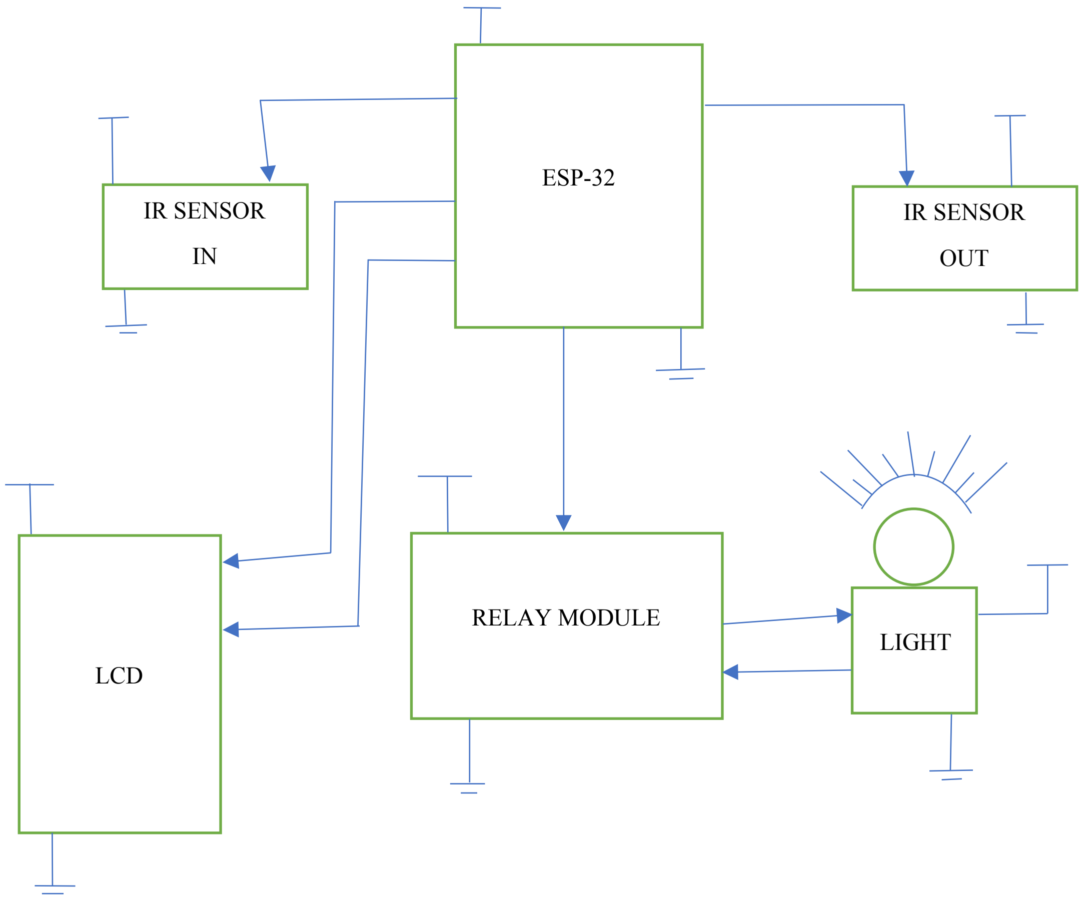
4.2 BLOCK DIAGRAM EXPLANATION

This project can be easily understandable by seeing this block diagram pattern . Here in block diagram it contains modules like power supply , relay module , LCD display , IR sensor , mainly the micro controller (ESP32) and lights . Here the project is executed by understanding that when we give the power supply to the microcontroller i.e ESP-32 here IR sensor senses the number of count that were passed through it there are two IR sensors one is at entry gate and other is at exit . When someone passes through the entry gate IR Sensor it starts counting and says to each count i.e WELCOME and same at exit gate IR sensor it senses that how many were passed through the sensor and counts and it says that VISIT AGAIN . That's the role of IR sensors and coming to point of relay module it works on Mobile Application named Blynk IOT (@playstore) the power supply is also supplied to the relay along with connected light and when we connect our WiFi Hotspot through mobile using micro controller which has wifi enabling system , here we can control our lights to turn it ON/OFF.

4.3 FLOW CHART



4.4 SCHEMATIC OF THE PROJECT



4.5 SCHEMATIC EXPLANATION

Initially we have to connect the VDD and VSS to all the electronic equipments that are present in the board. IR sensors (in) is connected to pin 18 and IR sensors (out) is connected to pin no .19 of Esp32 . lcd screen is connected to pin no (SDA- 21 and SCL -22) .

Relay is connected to pin 15 of Esp32 and one wire of electric appliances is connected to relay and other with ac supply

Blynk is connected to esp32 using wifi configuration .

All set to use Buliding automation .

CHAPTER 5

NODE MCU

5.1 INTRODUCTION & FEATURES

The Internet of Things (IoT) has been a trending field in the world of technology. It has changed the way we work. Physical objects and the digital world are connected now more thanever. Keeping this in mind, Espressif Systems (A Shanghai-based Semiconductor Company) hasreleased an adorable,bite-sizedWiFi enabled micro-controller– ESP-32,atan unbelievableprice! For less than $3, it can monitor and control things from anywhere in the world – perfect forjustaboutanyIoTproject.

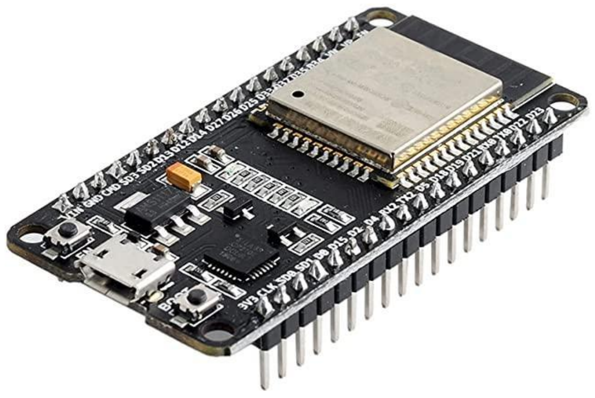


Fig:5.1 NodeMCUESP-32 Micro-Controller

The development board equips the ESP-12E module containing ESP-32 chip having TensilicaXtensa® 32-bit LX106 RISC microprocessor which operates at 80 to 160 MHz adjustable clock frequency and supports RTOS.

ESP-32 DEVKIT V1ESP-32Specifications&Features

- Single or Dual-Core 32-bit LX6 Microprocessor with clock frequency up to 240 MHz.
- OperatingVoltage: 3.3V
- InputVoltage:7-12V.
- 34 Programmable GPIO’s.
- Supports 802.11 b/g/n Wi-Fi connectivity with speeds up to 150 MBPS.
- Serial Connectivity include 4 x SPI, 2 x I2C, 2 x I2S, 3 x UART.
- 520 KB of SRAM, 448 KB of ROM, Flash up-to 16 MB and 16 KB of RTC SRAM.
- ClockSpeed: 240MHz.

- 1 Host controller for SD/SDIO/MMC and 1 Slave controller for SDIO/SPI.
- PCBAntenna.
- Built-in CP21XX USB to UART Serial Bridge.
- Wi-Fi 802.11 b/g/n/e/i.
- Bluetooth v4.2 BR/EDR and BLE.
- 2 x 8-bit DAC [D26][D25].
- 9 x touch sensors. Note: Touch Sensor 1 is [D0].
- 36 GPIO pins
- Internal temperature Sensor.

5.2 ARCHITECTURE

ESP-32ECHIP

- TENSILICAXTENSA32 BITLX106
- 240MHzClockFrequency
- 520 KBinternalRAM
- 448 KB of SROM.
- 16 MBExternalFlash
- 802.11/b/g/nWi-Fitransmitter.

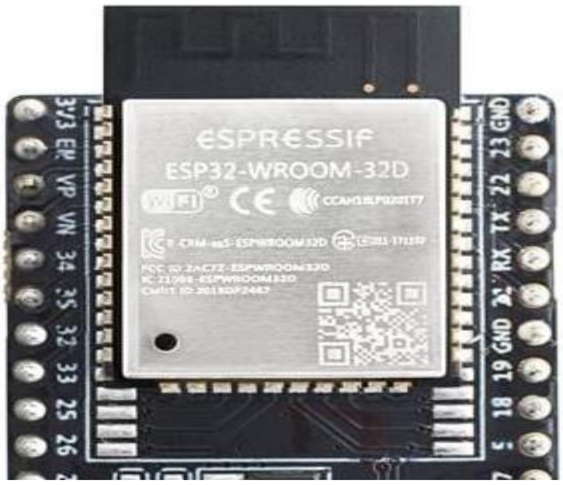

ESPRESSIF
ESP32-WROOM-32D
3V3 EN VP VN 34 35 32 33 25 26
GND 23 22 TX RX 21 GND 19 18 5

Fig 5.2 ESP-32EChip

5.3 PINDIAGRAM

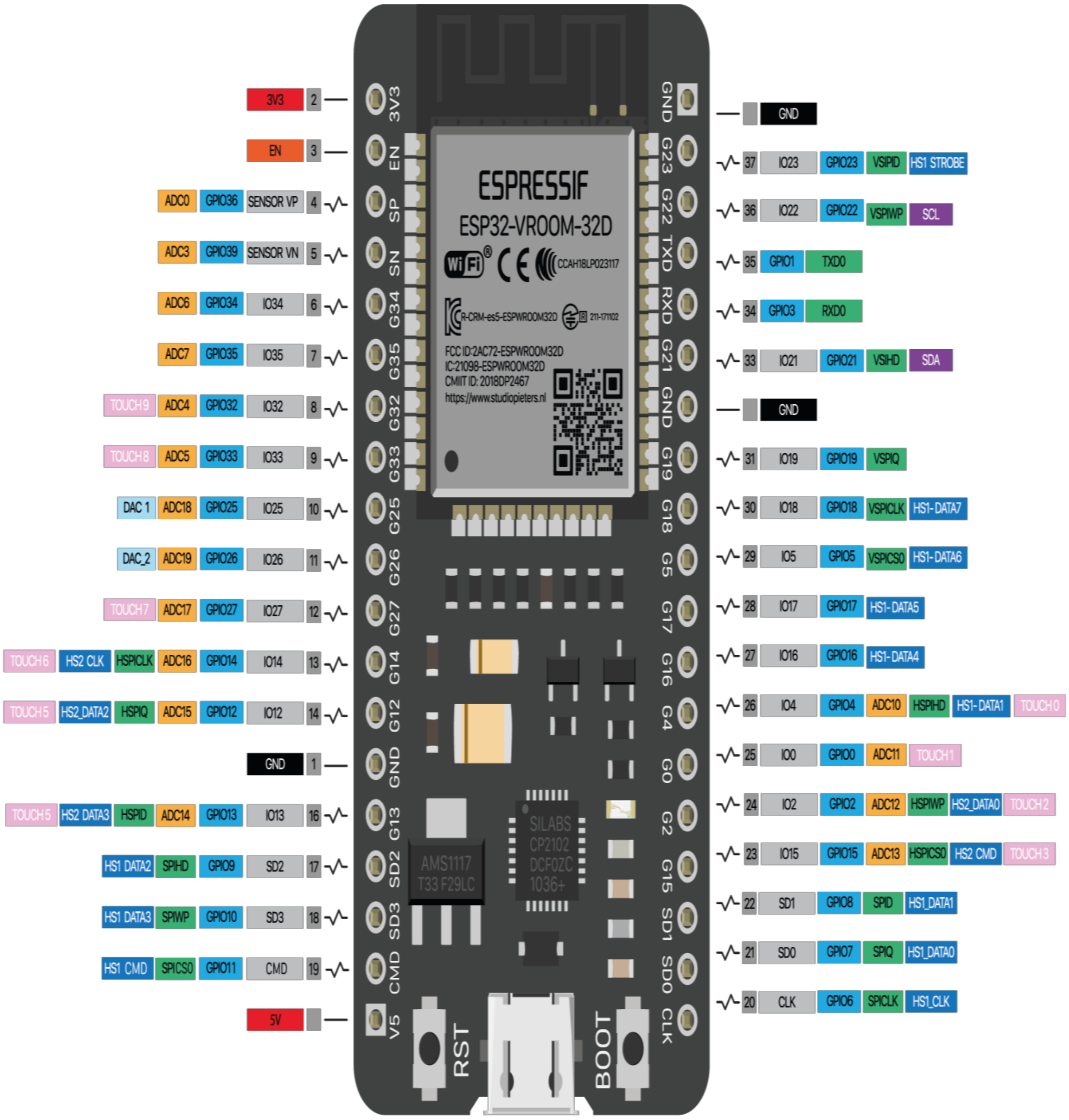


Fig.5.3 PinDiagram ofESP-32

5.4 PIN DESCRIPTION

Input/output pins:

There are totally 39 digital Pins on the ESP32 out of which 34 can be used as GPIO and the remaining are input only pins. The device supports 18-channels for 12-bit ADC and 2-channel for 8-bit DAC. It also has 16 channels for PWM signal generation and 10 GPIO pins supports capacitive touch features. The ESP32 has multiplexing feature, this enables the programmer to configure any GPIO pin for PWM or other serial communication through program. The ESP32 supports 3 SPI Interface, 3 UART interface, 2 I2C interface, 2 I2S interface and also supports CAN protocol.

- **3 UART interface:** The ESP32 supports 3 UART interface for TTL communication. This would require 3 sets of Rx and Tx pins. All the 6 pins are software configurable and hence any GPIO pin can be programmed to be used for UART.
- **External Interrupt:** Again since the ESP32 supports multiplexing any GPIO pin can be programmed to be used as an interrupt pin.
- **GPIO23 (MOSI), GPIO19(MISO), GPIO18(CLK) and GPIO5 (CS):** These pins are used for SPI communication. ESP32 supports two SPI, this is the first set.
- **GPIO13 (MOSI), GPIO12(MISO), GPIO14(CLK) and GPIO15 (CS):** These pins are used for SPI communication. ESP32 supports two SPI, this is the second set.
- **GPIO21(SDA), GPIO22(SCL):** Used for IIC communication using Wire library.
- **Reset Pin:** The reset pin for ESP32 is the Enable (EN) pin. Making this pin LOW, resets the microcontroller.

ESP32 Pin-out Configuration

Pin Category	**Pin Name**	**Details**
Power	Micro-USB, 3.3V, 5V, GND	**Micro-USB:** ESP32 can be powered through USB port **5V:** Regulated 5V can be supplied to this pin which is we be again regulated to 3.3V by on board regulator, to power the board. **3.3V:** Regulated 3.3V can be supplied to this pin to power the board. **GND:** Ground pins.
Enable	En	The pin and the button resets the microcontroller.
Analog Pins	ADC1_0 to ADC1_5 and ADC2_0 to ADC2_9	Used to measure analog voltage in the range of 0-3.3V. 12-bit 18 Channel ADC
DAC pins	DAC1 and DAC2	Used for Digital to analog Conversion
Input/Output Pins	GPIO0 to GPIO39	Totally 39 GPIO pins, can be used as input or output pins. 0V (low) and 3.3V (high). But pins 34 to 39 can be used as input only
Capacitive Touch pins	T0 to T9	These 10 pins can be used as touch pins normally used for capacitive pads
RTC GPIO pins	RTCIO0 to RTCIO17	These 18 GPIO pins can be used to wake up the ESP32 from deep sleep mode.

Serial	Rx, Tx	Used to receive and transmit TTL serial data.
External Interrupts	All GPIO	Any GPIO can be used to trigger an interrupt.
PWM	All GPIO	16 independent channel is available for PWM any GPIO can be made to work as PWM through the software
VSPI	GPIO23 (MOSI), GPIO19(MISO), GPIO18(CLK) and GPIO5 (CS)	Used for SPI-1 communication.
HSPI	GPIO13 (MOSI), GPIO12(MISO), GPIO14(CLK) and GPIO15 (CS)	Used for SPI-2 communication.
IIC	GPIO21(SDA), GPIO22(SCL)	Used for I2C communication.
AREF	AREF	To provide a reference voltage for input voltage.

Table: 5.1 ESP-32 Pin Configuration

ESP32 Technical Specifications

Microprocessor	TensilicaXtensa LX6
Maximum Operating Frequency	240MHz
Operating Voltage	3.3V
Analog Input Pins	12-bit, 18 Channel
DAC Pins	8-bit, 2 Channel
Digital I/O Pins	39 (of which 34 is normal GPIO pin)
DC Current on I/O Pins	40 mA
DC Current on 3.3V Pin	50 mA
SRAM	520 KB
Communication	SPI(4), I2C(2), I2S(2), CAN, UART(3)
Wi-Fi	802.11 b/g/n
Bluetooth	V4.2 – Supports BLE and Classic Bluetooth

Table 5.2 ESP-32 technical Specification

CHAPTER 6

FUNCTIONAL MODULES

6.1 RELAY MODULE

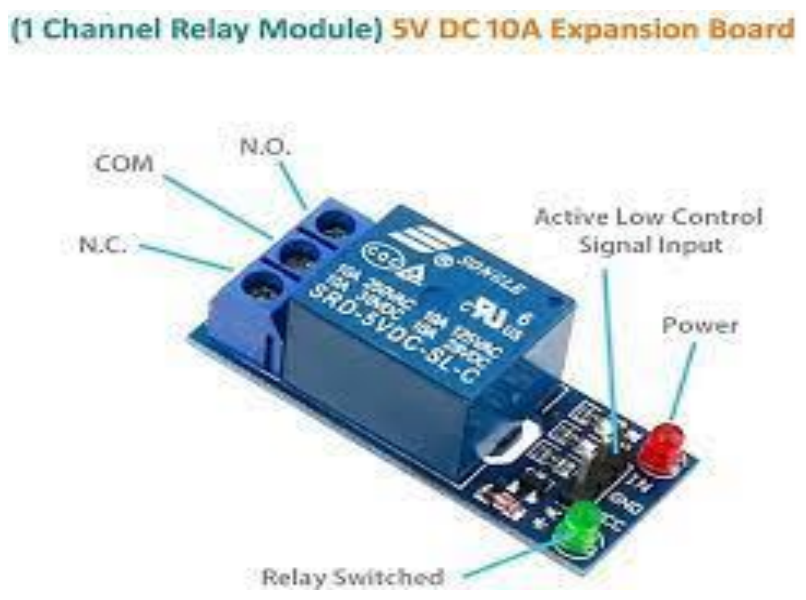

(1 Channel Relay Module) 5V DC 10A Expansion Board
COM
N.O.
N.C.
Active Low Control Signal Input
Power
Relay Switched

Fig 6.1 Relay Module

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 labelled 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. COM/POLE is connected to this when the relay coil is MAGNETIZED and vice versa.

In a 5V 4-channel relay interface board, each channel needs a 15-20mA driver current. It can be used to control various appliances and equipment with large current. It is equipped with high current relays that work under AC250V 10A or DC30V 10A. It has a standard interface that can be controlled directly by microcontroller. When the signal port is at low level, the signal light will light up and the optocoupler 817c (it transforms electrical signals by light and can isolate input and output electrical signals) will conduct, then the transistor will conduct, the relay coil will be electrified, and the normally open contact of the relay will be closed. When the signal port is at high level, the normally closed contact of the relay will be closed. So you can connect and disconnect the load by controlling the level of the control signal port.

6.2 IR SENSOR

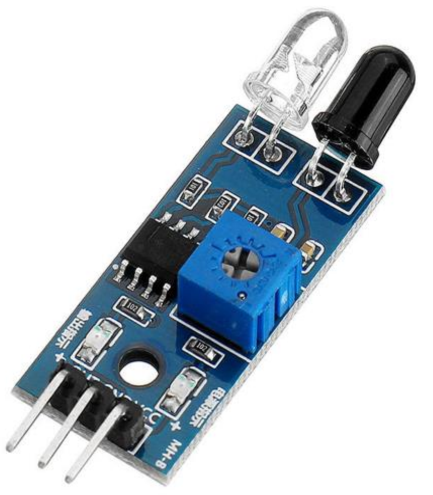


Fig 6.2 IR Sensor

IR sensor is an electronic device, that emits the light in order to sense some object of the surroundings. An IR sensor can measure the heat of an object as well as detects the motion. Usually, in the infrared spectrum, all the objects radiate some form of thermal radiation. These types of radiations are invisible to our eyes, but infrared sensor can detect these radiations.The emitter is simply an IR LED (Light Emitting Diode) and the detector is simply an IR photodiode . Photodiode is sensitive to IR light of the same wavelength which is emitted by the IR LED. When IR light falls on the photodiode, the resistances and the output voltages will change in proportion to the magnitude of the IR light received.

There are five basic elements used in a typical infrared detection system: an infrared source, a transmission medium, optical component, infrared detectors or receivers and signal processing. Infrared lasers and Infrared LED’s of specific wavelength used as infrared sources.

There are different types of infrared transmitters depending on their wavelengths, output power and response time. An IR sensor consists of an IR LED and an IR Photodiode, together they are called as PhotoCoupler or OptoCoupler.

IR Transmitter or IR LED

Fig 6.3 IR TRANSMITTER

Infrared Transmitter is a light emitting diode (LED) which emits infrared radiations called as IR LED’s. Even though an IR LED looks like a normal LED, the radiation emitted by it is invisible to the human eye.

IR Receiver or Photodiode

Fig 6.4 IR Receiver

Infrared receivers or infrared sensors detect the radiation from an IR transmitter. IR receivers come in the form of photodiodes and phototransistors. Infrared Photodiodes are different from normal photo diodes as they detect only infrared radiation. Below image shows the picture of an IR receiver or a photodiode

6.3 LCD DISPLAY

LCD stands for Liquid Crystal Display. LCD is finding wide spread use replacing LEDs (seven segment LEDs or other multi segment LEDs) because Of the following reasons: The declining prices of LCDs

Fig 6.5 LCD

The ability to display numbers, characters and graphics. This is in contrast to LEDs, which are limited to numbers and a few characters.
Incorporation of a refreshing controller into the LCD, thereby relieving the CPU of the task of refreshing the LCD. In contrast, the LED must be refreshed by the CPU to keep displaying the data. Ease of programming for characters and graphics.
These components are “specialized” for being used with the microcontrollers, which means that they cannot be activated by standard IC circuits. They are used for writing different messages on a miniature LCD.

A model described here is for its low price and great possibilities most frequently used in practice. It is based on the HD44780 microcontroller (Hitachi) and can display messages in two

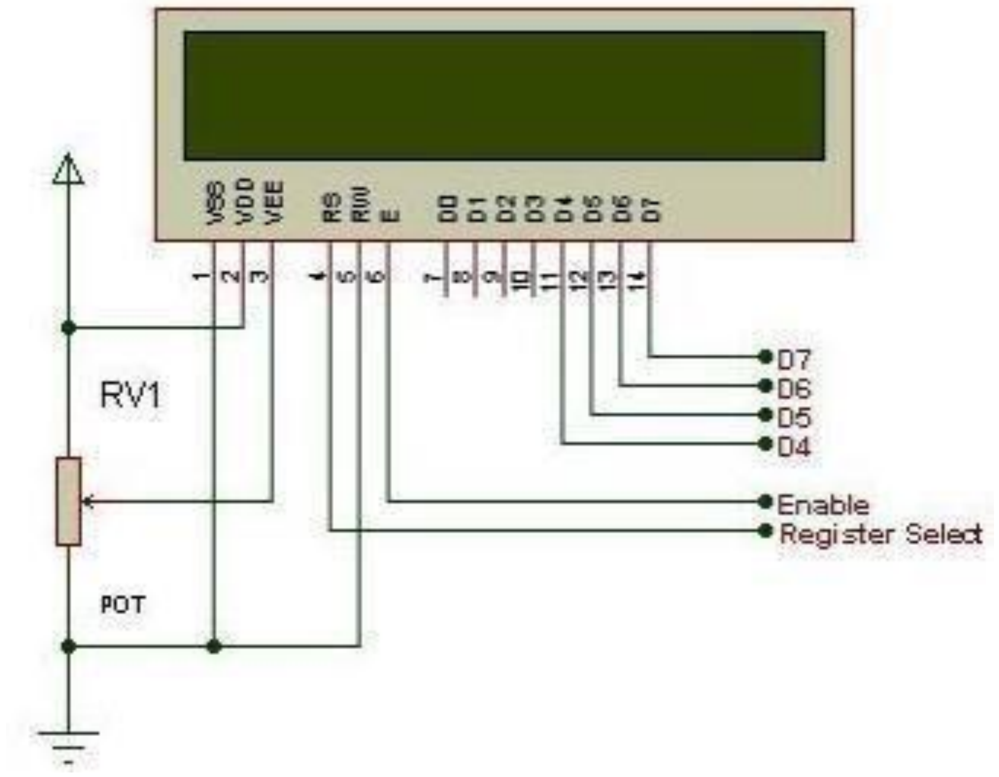


Fig 6.6 Pin Connections of LCD

lines with 16 characters each. It displays all the alphabets, Greek letters, punctuation marks, mathematical symbols etc. In addition, it is possible to display symbols that user makes up on its own. Automatic message on display (shift left and right), appearance of the pointer, backlight etc. are considered as useful characteristics.

LCD CONNECTIONS

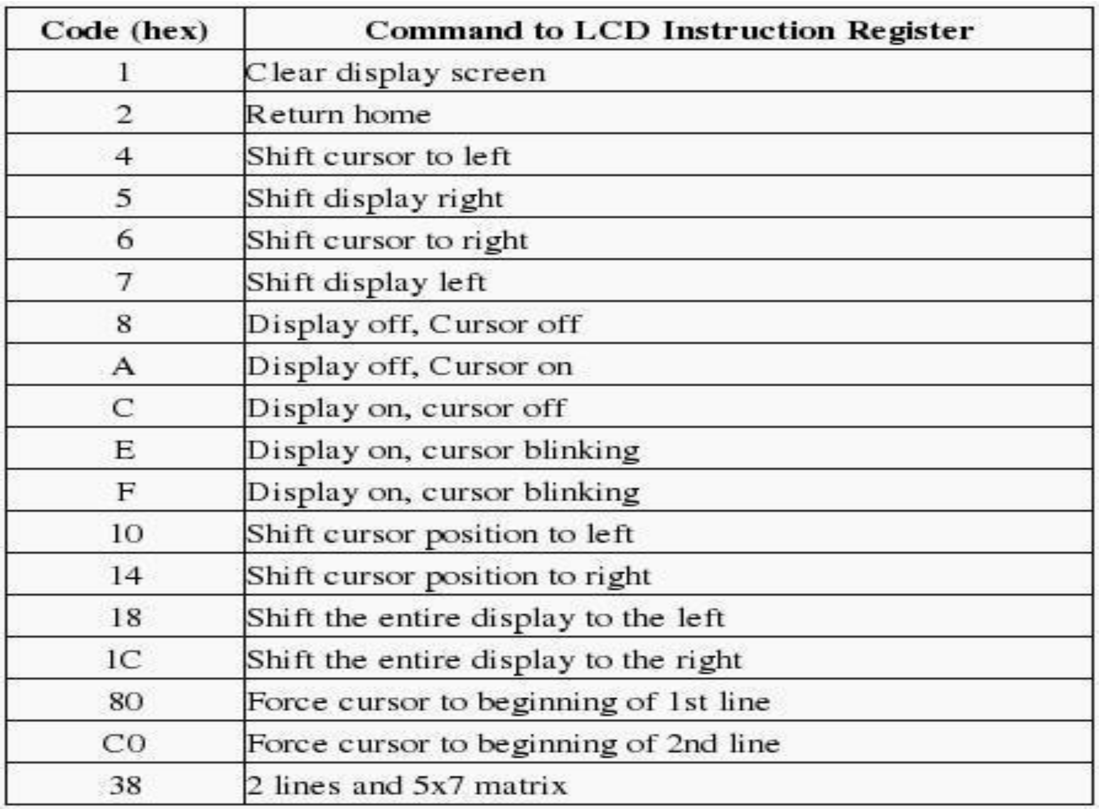
Depending on how many lines are used for connection to the microcontroller, there are 8bit and 4bit LCD modes. The appropriate mode is determined at the beginning of the process in a phase called initialization. In the first case, the data are transferred through outputs D0-D7 as it has been already explained. In case of 4-bit LED mode, for the sake of saving valuable I/O pins of the microcontroller, there are only 4 higher bits (D4-D7) used for communication, while other may be left unconnected.

Consequently, each data is sent to LCD in two steps: four higher bits are sent first (that normally would be sent through lines D4-D7), four lower bits are sent afterwards. With the help of initialization, LCD will correctly connect and interpret each data received. Besides, with regards to the fact that data are rarely read from LCD (data mainly are transferred from microcontroller to LCD) one more I/O pin may be saved by simple connecting R/W pin to the Ground. Such saving has its price. EvenEven though message displaying will be normally performed, it will not be possible to read from busy flag since it is not possible to read from display.

6.4 PIN CONFIGURATION OF LIQUID CRYSTAL DISPLAY

PinNo	Pin Name	Description
1	Vss (Ground)	Ground pin connected to system ground
2	Vdd (+5 Volt)	Powers the LCD with +5V (4.7V – 5.3V)
3	VE (Contrast V)	Decides the contrast level of display. Grounded to get maximum contrast.
4	Register Select	Connected to Microcontroller to shift between command/data register
5	Read/Write	Used to read or write data. Normally grounded to write data to LCD
6	Enable	Connected to Microcontroller Pin and toggled between 1 and 0 for data acknowledgement
7	Data Pin 0	Data pins 0 to 7 forms a 8-bit data line. They can be connected to Microcontroller to send 8-bit data. These LCD's can also operate on 4-bit mode in such case Data pin 4,5,6 and 7 will be left free..
15	LED Positive	Backlight LED pin positive terminal
16	LEDNegative	Backlight LED pin negative terminal

Table 6.1 Pin description table of 16*2 LCD display



Code (hex)	Command to LCD Instruction Register
1	Clear display screen
2	Return home
4	Shift cursor to left
5	Shift display right
6	Shift cursor to 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 of 1st line
C0	Force cursor to beginning of 2nd line
38	2 lines and 5x7 matrix

Table 6.2 Basic commands of LCD

6.5 IOT MODULE

Fig 6.7 Node MCU

An IoT module is a small electronic device embedded in objects, machines, and things connected to wireless networks that sends and receives data. Sometimes referred to as a "wireless module", "rf module" or "IoT chip," the IoT module contains the same technology and data circuits found in mobile phones but without features like a display or keypad. The INTERNET OF THINGS **(IoT)** is a set of technologies that uses sensors and actuators to inform us about the status of everyday items such as vehicles, tools and even living beings. It allows us to interact with them, enabling connectivity with platforms in the cloud that receive and process information for posterior analysis. This analyzed data is then used to make decisions.

5.6 POWER SUPPLY

The DC power supply is practically converted to each and every stage in an electronic system. Thus a common requirement for all these phases will be the DC power supply. All low power system can be run with a battery. But, for a long time operating devices, batteries could prove to be costly and complicated. The best method used is in the form of an unregulated power supply – a combination of a transformer, rectifier and a filter**.** All devices will have a certain power supply limit and the electronic circuits inside these devices must be able to supply a constant DC voltage within this limit. This DC supply is regulated and limited in terms of voltage and current. But the supply provided from mains may be fluctuating and could easily break down the electronic equipment, if not properly limited. This work of converting an unregulated alternating current (AC) or voltage to a limited Direct current (DC) or voltage to make the output constant regardless of the fluctuations in input, is done by a regulated power supply circuit.

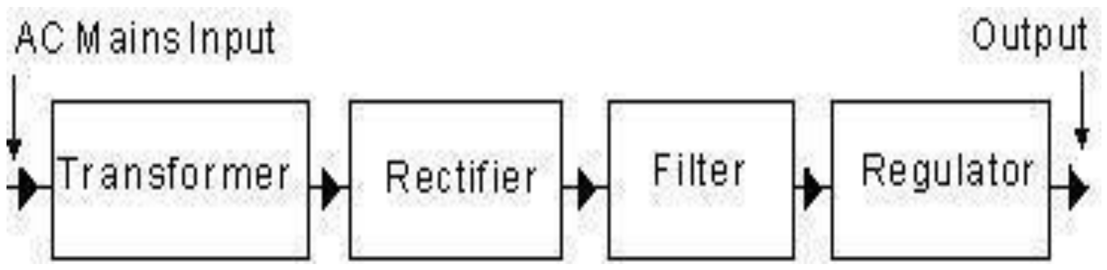


Fig 6.8 Block diagram of Regulated power supply

All the active and passive electronic devices will have a certain DC operating point (Q-point or Quiescent point), and this point must be achieved by the source of DC power. A step down transformer is used to reduce the voltage level to the devices needs. In India, a 1 Ø supply is available at 230 volts. The output of the transformer is a pulsating sinusoidal AC voltage, which is converted to pulsating DC with the help of a rectifier. This output is given to a filter circuit which reduces the AC ripples, and passes the DC components. But here are certain disadvantages in using an unregulated power supply. Regulated power supply is an electronic circuit that is designed to provide a constant dc voltage of predetermined value across load terminals irrespective of ac mains fluctuations or load variations. A regulated power supply essentially consists of an ordinary power supply and a voltage regulating device, as illustrated in the figure. The output from an ordinary power supply is fed to the voltage regulating device that provides the final output. The output voltage remains constant irrespective of variations in the ac input voltage or variations in output (or load) current.

CHAPTER 7

SOFTWARE DISCRIPTION AND SOURCE CODE

7.1 INTRODUCTION TO ARDUINO IDE

Arduino is a prototype platform (open-source) based on an easy-to-use hardware and software. It consists of a circuit board, which can be programmed (referred to as a microcontroller) and a ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board.

7.2 KEY FEATURES OF ARDUINO IDE

Arduino boards are able to read analog or digital input signals from different sensors and turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions.

You can control your board functions by sending a set of instructions to the microcontroller on the board via Arduino IDE (referred to as uploading software).

Unlike most previous programmable circuit boards, Arduino does not need an extra piece of hardware (called a programmer) in order to load a new code onto the board. You can simply use a USB cable.

Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program.

Finally, Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package.

After learning about the main parts of the Arduino UNO board, we are ready to learn how to set up the Arduino IDE. Once we learn this, we will be ready to upload our program onthe Arduino board.

7.3 ARDUINO DATA TYPES

Data types in C refers to an extensive system used for declaring variables or functions of different types. The type of a variable determines how much space it occupies in the storage and how the bit pattern stored is interpreted.

The following table provides all the data types that you will use During Arduino programming.

Void

The void keyword is used only in function declarations. It indicates that the function is expected to return no information to the function from which it was called.

Example:

```
Void Loop ( )
{
// rest of the code
}
```

Boolean

A Boolean holds one of two values, true or false. Each Boolean variable occupies one byte of memory.

Example:

```
Boolean state= false ; // declaration of variable with type boolean and initialize it with false.
Boolean state = true ; // declaration of variable with type boolean and initialize it with false.
```

Char

A data type that takes up one byte of memory that stores a character value. Character literals are written in single quotes like this: 'A' and for multiple characters, strings use double quotes: "ABC".

However, characters are stored as numbers. You can see the specific encoding in the ASCII chart. This means that it is possible to do arithmetic operations on characters, in whichthe ASCII value of the character is used. For example, 'A' + 1 has the value 66, since theASCII value of the capital letter A is 65.

Example:

```
Char chr_a = ‘a’ ;//declaration of variable with type char and initialize it with character a.
Char chr_c = 97 ;//declaration of variable with type char and initialize it with character 97
```

Unsigned char

Unsigned char is an unsigned data type that occupies one byte of memory. The unsigned char data type encodes numbers from 0 to 255.

Example:

Unsigned Char chr_y = 121 ; // declaration of variable with type Unsigned char and initialize it with character y

Byte

A byte stores an 8-bit unsigned number, from 0 to 255.

Example:

byte m = 25 ;//declaration of variable with type byte and initialize it with 25

int

Integers are the primary data-type for number storage. int stores a 16-bit (2-byte) value. This yields a range of -32,768 to 32,767 (minimum value of -2^15 and a maximum value of (2^15) - 1).

The int size varies from board to board. On the Arduino Due, for example, an int stores a 32-bit (4-byte) value. This yields a range of -2,147,483,648 to 2,147,483,647 (minimum value of -2^31 and a maximum value of (2^31) - 1).

Example:

int counter = 32 ;// declaration of variable with type int and initialize it with 32.

Unsigned int

Unsigned int's (unsigned integers) are the same as int in the way that they store a 2 byte value. Instead of storing negative numbers, however, they only store positive values, yielding a useful range of 0 to 65,535 (2^16) - 1). The Due stores a 4 byte (32-bit) value, ranging from 0 to 4,294,967,295 (2^32 - 1).

Example:

Unsigned int counter= 60 ; // declaration of variable with type unsigned int and initialize it with 60.

Word

On the Uno and other ATMEGA based boards, a word stores a 16-bit unsigned number. On the Due and Zero, it stores a 32-bit unsigned number.

Example

word w = 1000 ;//declaration of variable with type word and initialize it with 1000.

Long Long variables are extended size variables for number storage, and store 32 bits (4 bytes), from 2,147,483,648 to 2,147,483,647.

Example:

Long velocity= 102346 ;//declaration of variable with type Long and initialize it with 102346

Unsigned long

Unsigned long variables are extended size variables for number storage and store 32 bits (4 bytes). Unlike standard longs, unsigned longs will not store negative numbers, making their range from 0 to 4,294,967,295 (2^32 - 1).

Example:

Unsigned Long velocity = 101006 ;// declaration of variable with type Unsigned Long and initialize it with 101006.

Short

A short is a 16-bit data-type. On all Arduinos (ATMega and ARM based), a short stores a 16-bit (2-byte) value. This yields a range of -32,768 to 32,767 (minimum value of -2^15 and a maximum value of (2^15) - 1).

Example:

short val= 13 ;//declaration of variable with type short and initialize it with 13

Float

Data type for floating-point number is a number that has a decimal point. Floating-point numbers are often used to approximate the analog and continuous values because they have greater resolution than integers.

Floating-point numbers can be as large as 3.4028235E+38 and as low as 3.4028235E+38. They are stored as 32 bits (4 bytes) of information.

Example:

float num = 1.352;//declaration of variable with type float and initialize it with

1.352.

Double

On the Uno and other ATMEGA based boards, Double precision floating-point number occupies four bytes. That is, the double implementation is exactly the same as the float, with no gain in precision. On the Arduino Due, doubles have 8-byte (64 bit) precision

Example:

double num = 45.352 ;// declaration of variable with type double and initialize it with 45.352.

7.4 STEPS TO UPLOAD THE PROGRAM IN ARDUINO BOARD

In this section, we will learn in easy steps, how to set up the Arduino IDE on our computer and prepare the board to receive the program via USB cable.

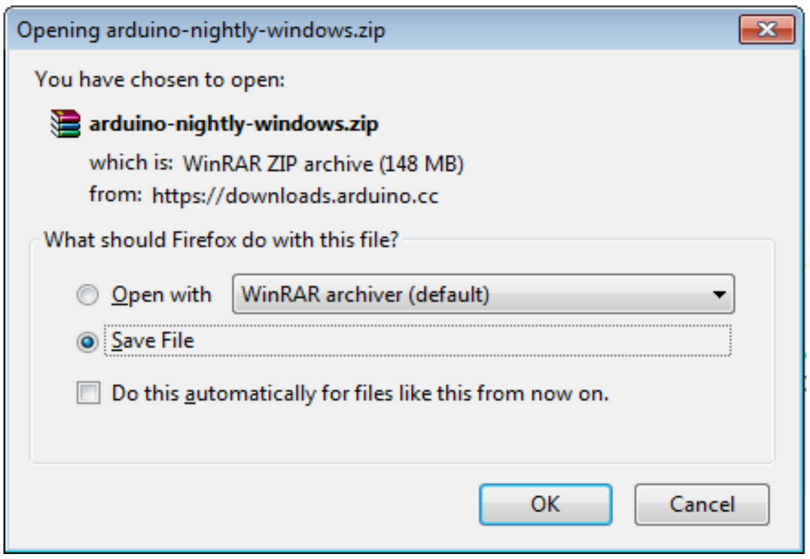
Step 1: First you must have your Arduino board (you can choose your favorite board) and a USB cable.

In case you use Arduino UNO, Arduino Duemilanove, Nano, Arduino Mega2560, or Diecimila, you will need a standard USB cable (A plug to B plug), the kind youwould connect to a USB printer as shown in the following image.

Fig 7.1 USB Cable

Step 2: Download Arduino IDE Software.

You can get different versions of Arduino IDE from the Download page on the Arduino Official website. You must select your software, which is compatible with your operating system (Windows, IOS, or Linux). After your file download is complete, unzip the file.



Opening arduino-nightly-windows.zip

You have chosen to open:

arduino-nightly-windows.zip

which is: WinRAR ZIP archive (148 MB)

from: https://downloads.arduino.cc

What should Firefox do with this file?

( ) Open with WinRAR archiver (default)

[✓ Save File]

[ ] Do this automatically for files like this from now on.

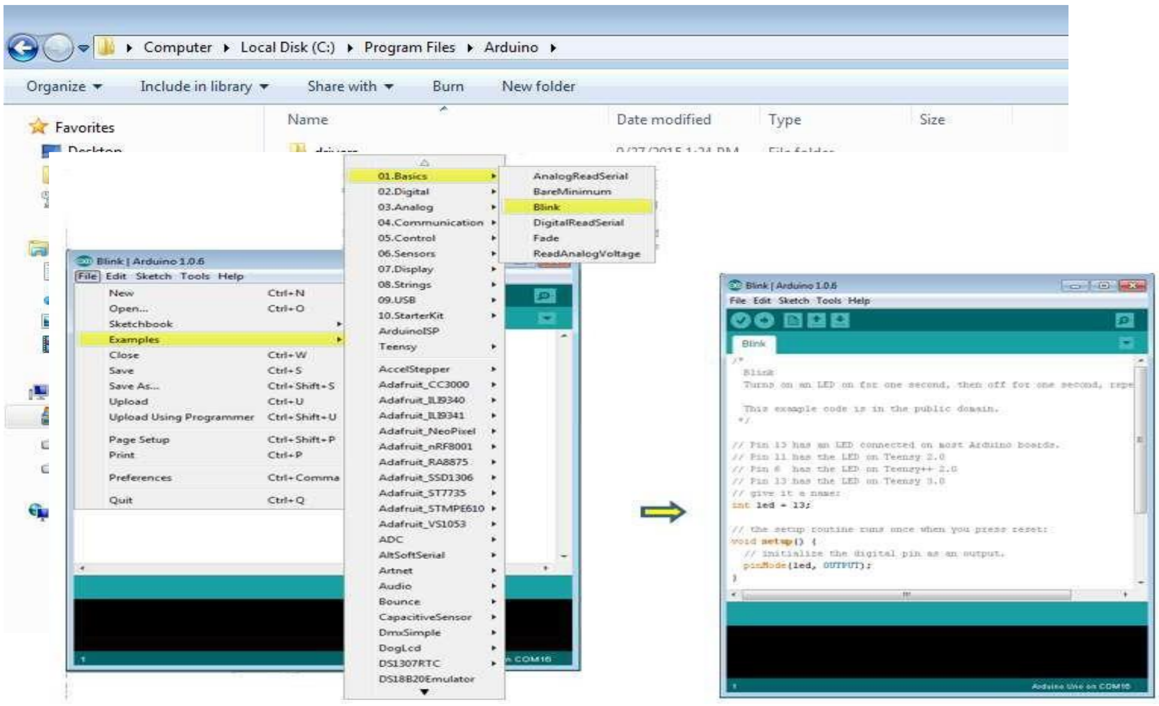
Fig 7.2 Opening Arduino

Step 3: Power up your board.

The Arduino Uno, Mega, Duemilanove and Arduino Nano automatically draw power from either, the USB connection to the computer or an external power supply. If you are using an Arduino Diecimila, you have to make sure that the board is configured to draw power from the USB connection. The power source is selected with a jumper, a small piece of plastic that fits onto two of the three pins between the USB and power jacks. Check that it is on the two pins closest to the USB port. Connect the Arduino board to your computer using the USB cable. The green power LED (labeled PWR) should glow.

Step 4: Launch Arduino IDE.

After your Arduino IDE software is downloaded, you need to unzip the folder. Inside the folder, you can find the application icon with an infinity label (application.exe). .



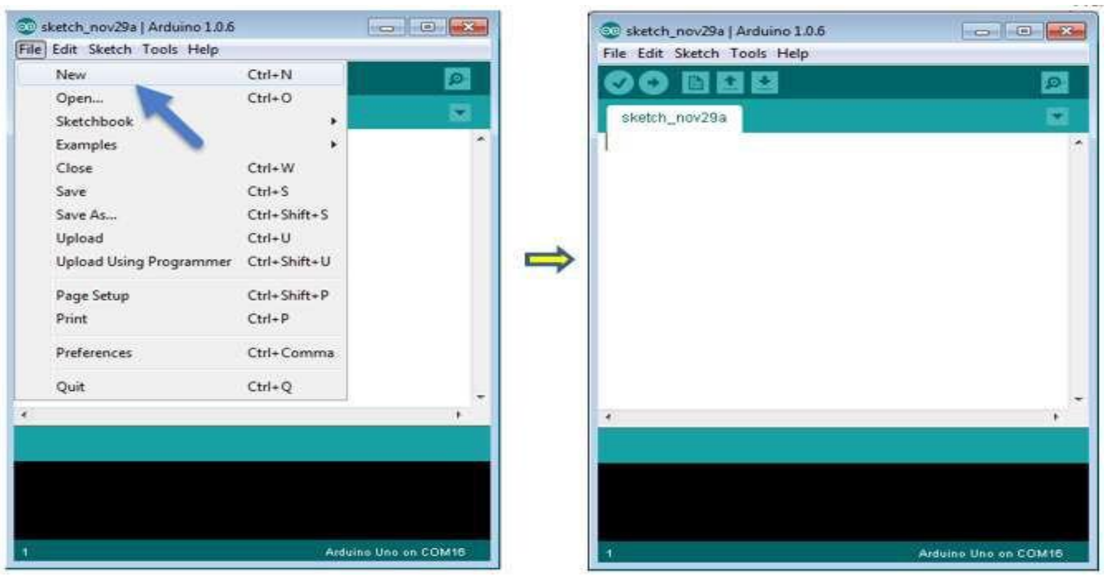
```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repe

  This example code is in the public domain.
 */

// Pin 13 has an LED connected on most Arduino boards.
// Pin 11 has the LED on Teensy 2.0
// Pin 6  has the LED on Teensy++ 2.0
// Pin 13 has the LED on Teensy 3.0
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}
```

Fig 7.3 Launching Arduino

Step 5: Open your first project.Once the software starts, you have two options: Create a new project.

Open an existing project example.

To create a new project, select File --> New.To open



To open an existing project example, select File -> Example -> Basics -> Blink.

Fig 7.4 CREATING A PROJECT

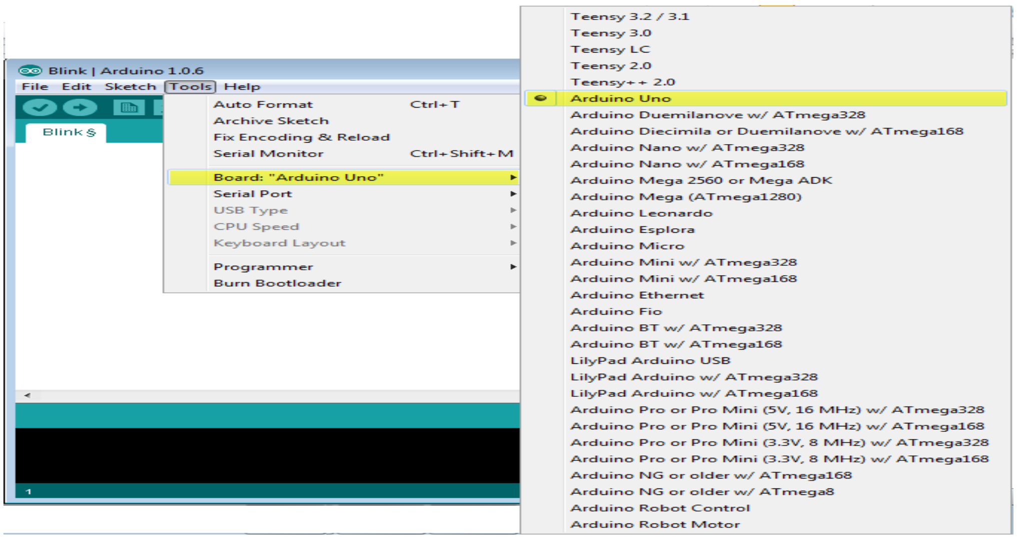

Blink | Arduino 1.0.6
File Edit Sketch Tools Help
Blink §
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Board: "Arduino Uno"
Serial Port
USB Type
CPU Speed
Keyboard Layout
Programmer
Burn Bootloader
Teensy 3.2 / 3.1
Teensy 3.0
Teensy LC
Teensy 2.0
Teensy++ 2.0
Arduino Uno
Arduino Duemilanove w/ ATmega328
Arduino Diecimila or Duemilanove w/ ATmega168
Arduino Nano w/ ATmega328
Arduino Nano w/ ATmega168
Arduino Mega 2560 or Mega ADK
Arduino Mega (ATmega1280)
Arduino Leonardo
Arduino Esplora
Arduino Micro
Arduino Mini w/ ATmega328
Arduino Mini w/ ATmega168
Arduino Ethernet
Arduino Fio
Arduino BT w/ ATmega328
Arduino BT w/ ATmega168
LilyPad Arduino USB
LilyPad Arduino w/ ATmega328
LilyPad Arduino w/ ATmega168
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168
Arduino NG or older w/ ATmega168
Arduino NG or older w/ ATmega8
Arduino Robot Control
Arduino Robot Motor
1

Fig 7.5 SELECTING THE ARDUINO BOARD

Here, we are selecting just one of the examples with the name Blink. It turns the LED on and off with some time delay. You can select any other example from the list.

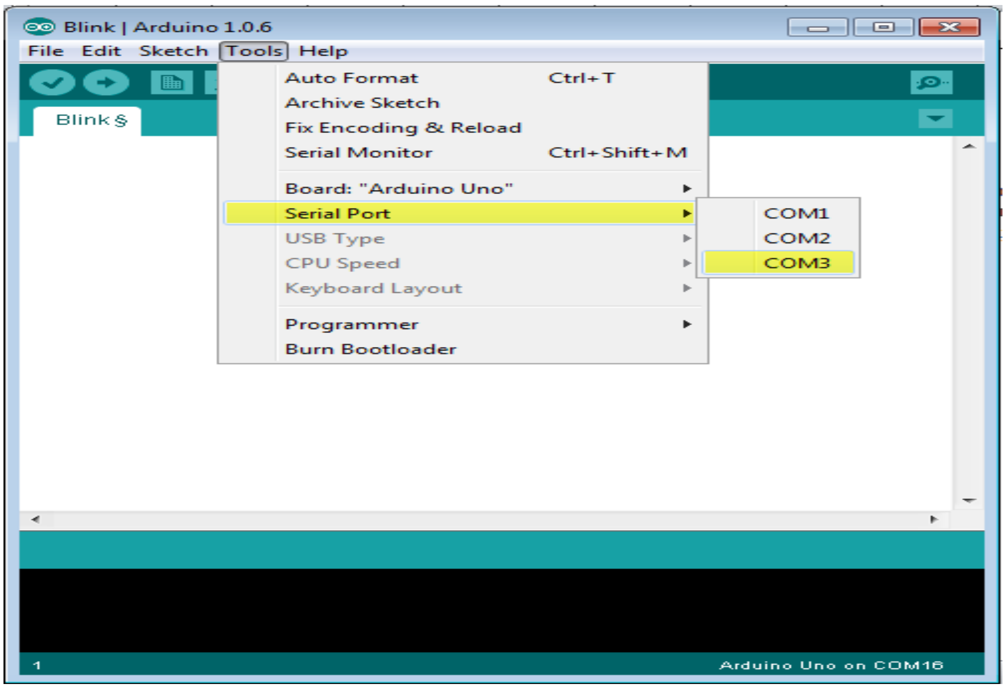


Fig 7.6 SELECTING THE PORT

Step 6: Select your Arduino board.

To avoid any error while uploading your program to the board, you must select the correct Arduino board name, which matches with the board connected to your computer.

Go to Tools -> Board and select your board

Here, we have selected Arduino Uno board according to our tutorial, but you must select the name matching the board that you are using

Step 7: Select your serial port.

Select the serial device of the Arduino board. Go to Tools ->Serial Port menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu, the entry that disappears should be of the Arduino board. Reconnect the board and select that serial port.

Step 8: Upload the program to your board.

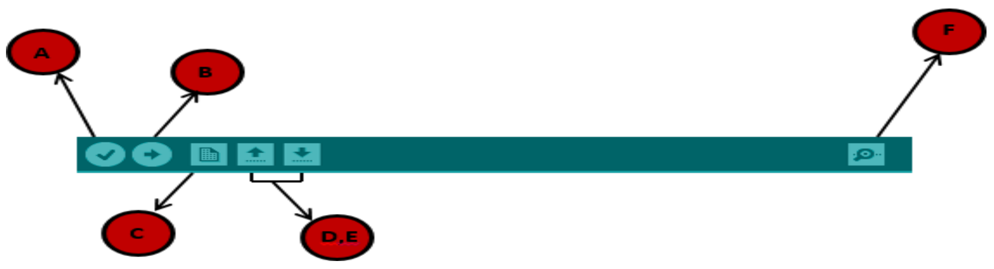


Fig 7.7 UPLOADING THE PROGRAM

A-Used to check if there is any compilation error.

B- Used to upload a program to the Arduino board.

C- Shortcut used to create a new sketch.

D- Used to directly open one of the example sketch.

E- Used to save your sketch.

F- Serial monitor used to receive serial data from the board and send the serial data to the board.

Now, simply click the "Upload" button in the environment. Wait a few seconds; you will see the RX and TX LEDs on the board, flashing. If the upload is successful, the message "Done uploading" will appear in the status bar.

Note: If you have an Arduino Mini, NG, or other board, you need to press the reset button physically on the board, immediately before clicking the upload button on the Arduino Software.

7.5 PROGRAMMING STRUCTURE

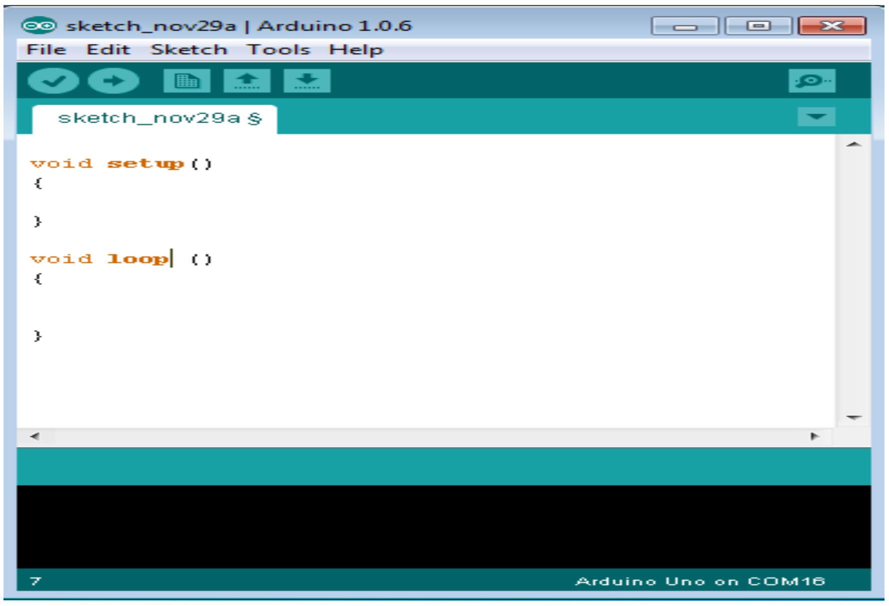
In this chapter, we will study in depth, the Arduino program structure and we will learn more new terminologies used in the Arduino world. The Arduino software is open-source. The source code for the Java environment is released under the GPL and the C/C++ microcontroller libraries are under the LGPL.

Sketch: The first new terminology is the Arduino program called "sketch".

Structure

Arduino programs can be divided in three main parts: Structure, Values (variables and constants), and Functions. In this tutorial, we will learn about the Arduino software program, step by step, and how we can write the program without any syntax or compilation error.

Let us start with the Structure. Software structure consist of two main functions:



```
void setup()
{

}

void loop ()
{


}
```

Fig 7.8 ARDUINO PROGRAMMING STRUCTURE

Setup( ) function

Loop( ) function

PURPOSE:

The setup() function is called when a sketch starts. Use it to initialize the variables, pin modes, start using libraries, etc. The setup function will only run once, after each power up or reset of the Arduino board.

INPUT OUTPUT

RETURN

```
Void Loop ( )
{
}
```

PURPOSE:

After creating a setup() function, which initializes and sets the initial values, the loop() function does precisely what its name suggests, and loops secutively, allowing your program to change and respond. Use it to actively control the Arduino board.

INPUT

OUTPUT

RETURN

7.6 Source code :

```
#ifdef ESP32
  #include <Arduino.h>
  #include <WiFi.h>
  #include <AsyncTCP.h>
  #include <ESPAsyncWebServer.h>
  #include <AsyncElegantOTA.h>
#else
  #include <Arduino.h>
  #include <ESP8266WiFi.h>
  #include <ESPAsyncTCP.h>
  #include <ESPAsyncWebServer.h>
  #include <AsyncElegantOTA.h>
#endif

AsyncWebServer server(80);

#define BLYNK_TEMPLATE_ID "TMPL3z6ZYJYUn"
#define BLYNK_TEMPLATE_NAME "BUILDING AUTOMATION"
#define BLYNK_AUTH_TOKEN "ItyBMkJeHFbFr6q3KZ60eE3NnJK4o-jz"

#define BLYNK_PRINT Serial
#ifdef ESP32
  #include <WiFi.h>
  #include <BlynkSimpleEsp32.h>
#else
  #include <ESP8266WiFi.h>
  #include <BlynkSimpleEsp8266.h>
#endif

char auth[] = "ItyBMkJeHFbFr6q3KZ60eE3NnJK4o-jz";
char ssid[] = "project17";                              // Your WiFi credentials.
char pass[] = "project17";

#include<LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);

#define VPIN_BUTTON V2

#define light 15

String state1="OFF";

#define entry 18
#define exit 19
int entry_sense,exit_sense;
int count=0;

WidgetLCD lcd1(V0);
WidgetLED indicator(V1);

void setup()
{
  Serial.begin(9600);

  pinMode(entry,INPUT);
  pinMode(exit,INPUT);

  pinMode(light,OUTPUT);
  digitalWrite(light,HIGH);

  lcd.init();
  lcd.backlight();

  lcd.clear();
  lcd.print("CONNECTING TO...");
  lcd.setCursor(0,1);
  lcd.print(ssid);
  delay(1000);

  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, pass);
  Serial.print("Connecting to WiFi ..");
  while(WiFi.status() != WL_CONNECTED)
  {
    Serial.print('.');
    delay(500);
  }
  Blynk.config(auth);
  Serial.println("READY");

  lcd.clear();
  lcd.print(" WIFI CONNECTED ");
  lcd.setCursor(0,1);
  lcd.print(WiFi.localIP());
  delay(5000);
  lcd.clear();

  server.on("/", HTTP_GET, [](AsyncWebServerRequest *request) {
    request->send(200, "text/plain", "Hi! I am ESP32.");
  });

  AsyncElegantOTA.begin(&server);    // Start ElegantOTA
  server.begin();
  Serial.println("HTTP server started");
}
BLYNK_CONNECTED()
{
  Blynk.syncVirtual(VPIN_BUTTON);
}
BLYNK_WRITE(VPIN_BUTTON)
{
  int toggleState = param.asInt();
  Serial.println(toggleState);
  if(toggleState==1)
  {
    state1="ON";
    digitalWrite(light,LOW);
  }
  else
  {
    state1="OFF";
    digitalWrite(light,HIGH);
  }
}

void loop()
{
  entry_sense=digitalRead(entry);
  exit_sense=digitalRead(exit);

  if(entry_sense==0)
  {
    count=count+1;
    lcd.clear();lcd1.clear();
    lcd.print(" *  WELCOME  * ");lcd1.print(0,0," *  WELCOME  * ");

digitalWrite(2,HIGH);indicator.on();delay(300);digitalWrite(2,LOW);indicator.off();delay(300);
    delay(1000);
  }
  else if(exit_sense==0 && count >=1)
  {
    count=count-1;
    lcd.clear();lcd1.clear();
    lcd.print("* VISIT  AGAIN ");lcd1.print(0,0," VISIT  AGAIN *");

digitalWrite(2,HIGH);indicator.on();delay(300);digitalWrite(2,LOW);indicator.off();delay(300);
    delay(1000);
  }

  lcd.clear();lcd1.clear();
  lcd.print("COUNT : ");lcd1.print(0,0,"COUNT : ");lcd.print(count);lcd1.print(8,0,count);
  lcd.setCursor(0,1);
  lcd.print("LIGHT : ");lcd1.print(0,1,"LIGHT : ");lcd.print(state1);lcd1.print(8,1,state1);


 digitalWrite(2,HIGH);indicator.on();delay(300);digitalWrite(2,LOW);indicator.off();delay(300);
  Blynk.run();
  }
```

CHAPTER 8

Result And Analysis Report

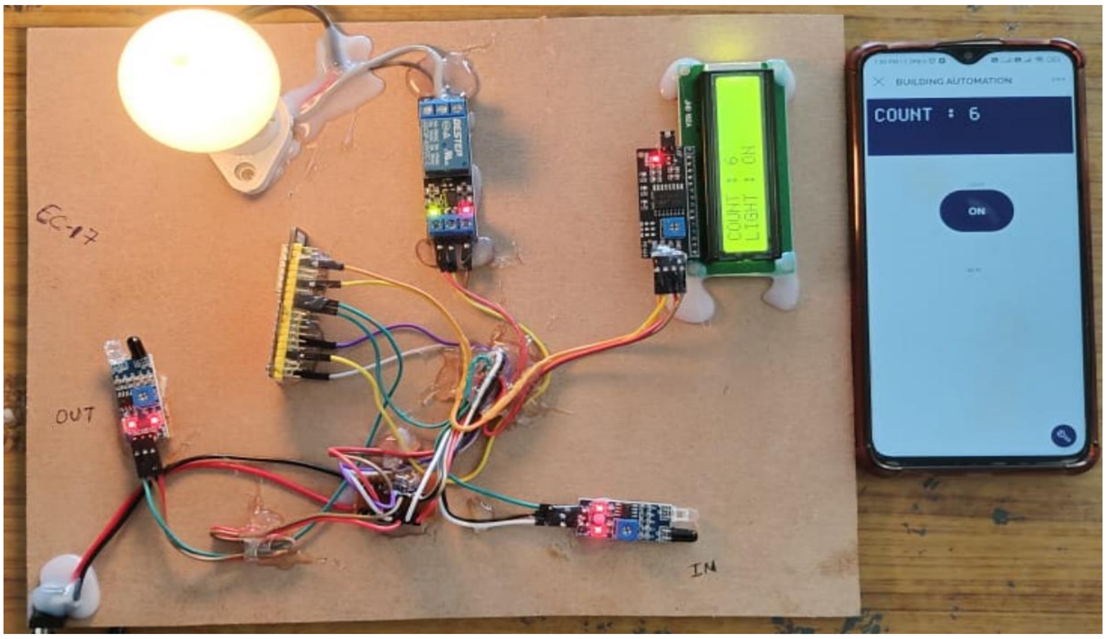


Fig-8.1 : Six person detected and light is ON

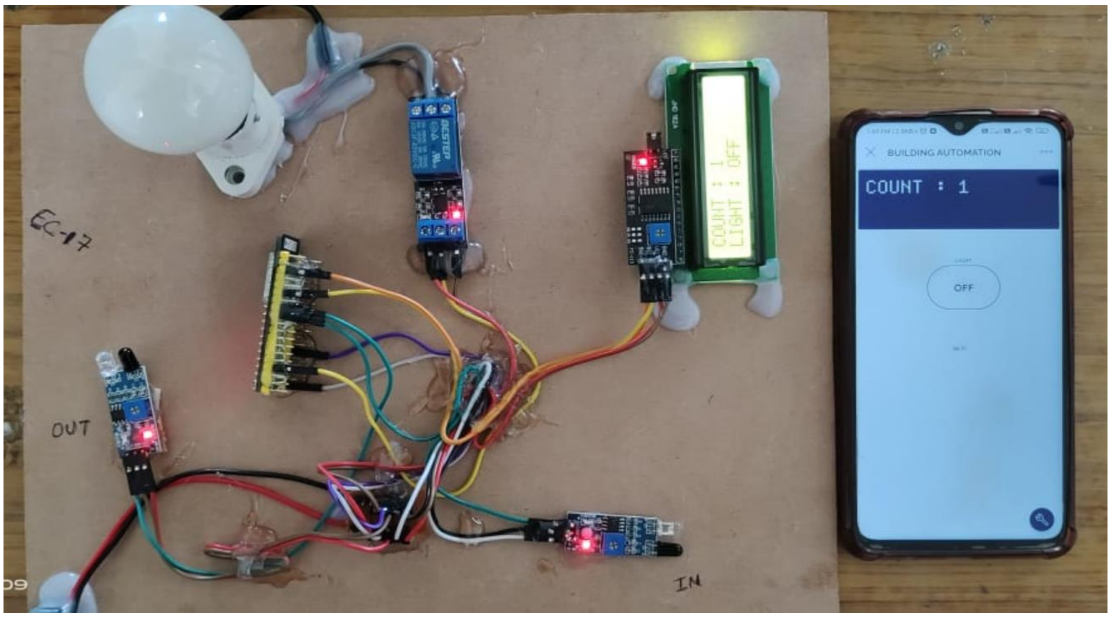


Fig-8.2 : Five is person out & Remaning 1 person and light is OFF

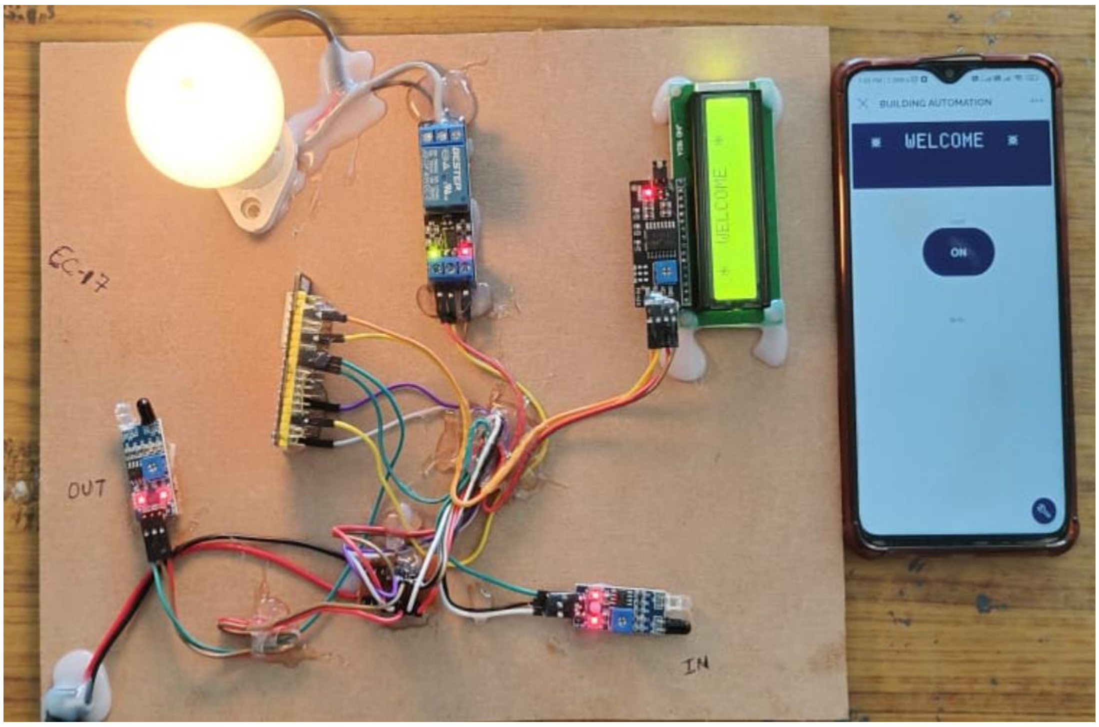


Fig-8.3 : Person entering and screen showing " Welcome" and light is ON

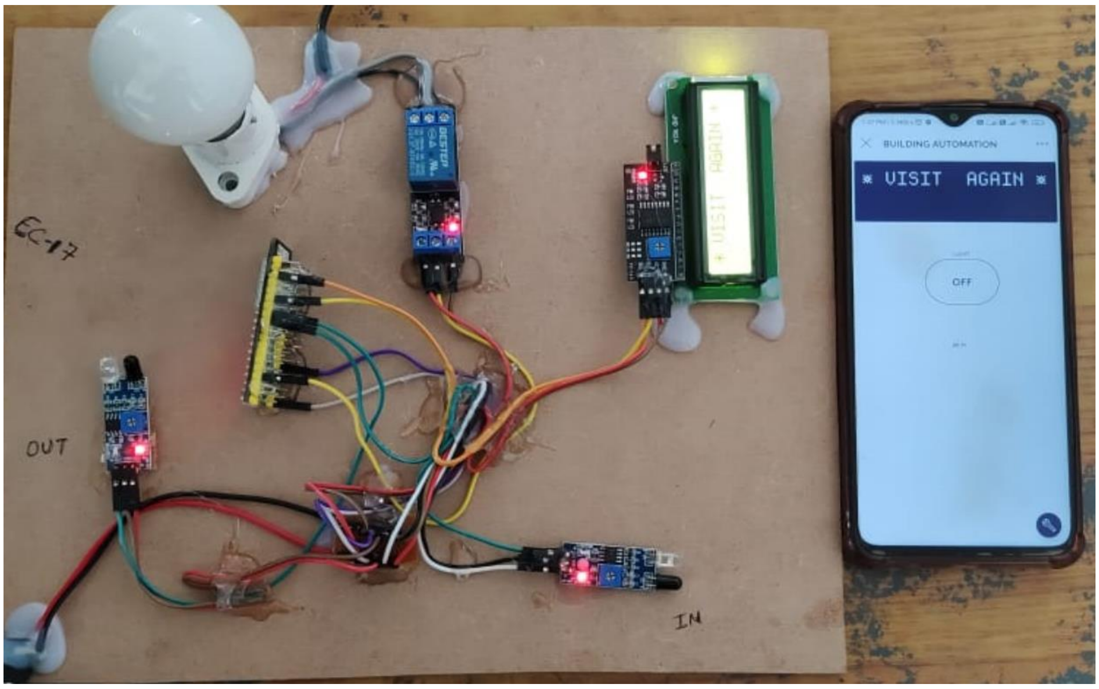


Fig-8.4 : Person exit and screen showing " Visit again " and light is OFF

Advantages

Efficient and time saving the machine to machine interaction provides better efficiency , hence accurate results can be obtained fastly. This results in saving valuable time instead of repeting the same task every day , it enables people to do other creative jobs easily.

Applications

* Using this project ,we can turn on or off appliances remotely, using Phone or tablet.

* The project can be further expanded to a smart home automation System by including some sensors like lights sensors, temperature sensor, Safety sensors etc. and automatically (room temperature), door etc, and Transmit the information to our phone.

* Additionally, we can connect to internet and control the home from Remote location over internet and also montior the safety.

Conclusion

It is evident from this project work that an individual control home automation system can be cheaply made from low-cost locally available components and can be used to control multifarious home appliances ranging from the security lamps, the television to the air conditioning system and even the entire house lighting system. And better still, the components required are so small and few that they can be packaged into a small inconspicuous container. The designed home automation system was tested a number of times and certified to control different home appliances used in the lighting system, air conditioning system, home entertainment system and many more . Hence, this system is scalable and flexible.

Future Scope

The aim is to design a prototype that establishes wireless remote control over a network of home appliances. The application is designed to run on android device providing features like, switch mode control, voice command control and a provision to view the status of the devices on the application itself. Considering its wide range of application, following are the scope of this prototype. The system can be implemented in homes, small offices and malls as well, being in-charge of control of the electrical appliances. For remote access of appliances in internet or intranet. The appliances in the above mentioned environment can be controlled in intra-network or can be accessed via internet. The development of technology friendly environment. The system incorporates the use of technology and making HAS. By the use of day to day gadgets we can utilize them for a different perspective.

BIBLOGRAPHY

[1] Reshma Shinde, Ritika Pardeshi, Archana Vishwakarma and Nayan Barhate, Need for Wireless Fire Detection Systems using IOT , International Research Journal of Engineering and Technology, Volume 04, Issue 01, pp-1078-1081, January 2017.

[2] Alexander Filonenko, Danilo Caceres Hernandez, Ajmal Shahbaz and Kang-Hyun Jo , Unied Smoke and Flame Detection for Intelligent Surveillance System, IEEE, Volume 08, Issue 03, pp-953-957, May 2016.

[3] Xiaofei Ji, Xuan Xie, Xinmeng Zuo and Jiangtao Cao, Design and Implementation of Smoke Early Warning System , IEEE, Volume 03, Issue 17, pp-351-356, July 2017.

[4] Digvijay Singh, Neetika Sharma, Mehak Gupta and Shubham Sharma , Development of System for Early Fire Detection using Arduino UNO, International Journal of Engineering Science and Computing, Volume 07, Issue 05, pp-10857-10860, May 2017.

[5] Robert Sowah, Abdul R. Ofoli, Selase Krakani and Seth Fiawoo, Hard- ware Module Design of a Real time Multi-Sensor Fire Detection and Notica- tion System using Fuzzy Logic , IEEE, Volume 08, Issue 14, pp-653-659, July 2016.

[6] Suneel Mudunuru, V. Narasimha Nayak, G.Madhusudhana Rao and K.Sreenivasa Ravi , Real Time Security Control System for Smoke and Fire Detection Using ZigBee , International Research Journal of Engineering and Technology, Volume 02, Issue 06, pp-2531-2540, October 2011.

[7] Chen, Thou-Ho, et al.The smoke detection for early fire-alarming system based on video processing, in Proceedings of International Conference on Intelligent Information Hiding and Multimedia, 2006.

[8] Kaushik Sen, Jeetsarkar, Sutapa Saha, Anukrishanaroy, Dipsetudey, Sumit Baitalik, Chandrasekhar Nandi, “Automated Fire Detection and Controlling System,” in international advanced research journal in science, engineering and technology, pp. 34-37, 2015.

[9] San-Miguel-Ayanz J, Ravail N. “Active fire detection for fire emergency management: Potential and limitations for the operational use of remote sensing,” Natural Hazards Journal, 2005 July. 35(3), 361–376

[10] R. K. Kodali, S. Soratkal and L. Boppana, “IoT based control of appliances,” 2016 International Conference on Computing, Communication and Automation (ICCCA), Noida, 2016, pp. 1293-1297.