COAL MINE MONITORING SYSTEM USING Lo-Ra

A Project work submitted in partial fulfillment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS & COMMUNICATION ENGINEERING

by

M. VENKATA AKHIL

Regd.No:16211A04D3

N. PRADEEP

Regd.No:16211A04D6

M. RAMA SAI KAUSHIK

Regd.No:16211A04E7

Under the esteemed guidance of
Dr.R. Balaji
Associate Professor

Department of Electronics and Communication Engineering
B.V.Raju Institute of Technology, Autonomous
Vishnupur, Narsapur, Medak. (Dt)
(Affiliated to JNTU, Hyderabad)
2020

B.V.Raju Institute of Technology
Vishnupur, Narsapur, Medak. (Dt) Pin:502313
(Affiliated to JNTU, Hyderabad)
Ph: 08458-222000, 222001 Fax: 08458-222002

Department of Electronics & Communication Engineering

CERTIFICATE

This is to certify that the Project work entitled on the Intelligent Coal Mine Monitoring System based on the Lora-cloud is being submitted by Mr. M. Venkata Akhil (16211A04D3), Mr. N. Pradeep (16211A04D6), Mr. M. Rama Sai Kaushik (16211A04E7) in partial fulfillment of the requirement for the award of the degree of **B.Tech. in Electronics & Communication Engineering**, by Jawaharlal Nehru Technological University Hyderabad is a record of bonafide work carried out by them under my guidance and supervision from 2019 to 2020.

The results presented in this project have been verified and are found to be satisfactory.

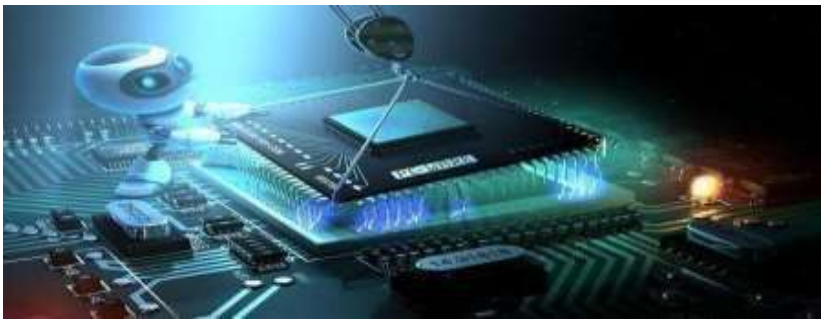
Dr. R. Balaji
Associate Professor

Dr. I. A. Pasha
M.E, Ph.D., Post Doc., MISTE, MIEEE.
Professor & HOD, Dept. of ECE

EXTERNAL EXAMINER

Dept. of Electronics and Communications Engineering

Centre for Embedded Systems Design



CERTIFICATE

This is to certify that Mr. M. Venkata Akhil, N. Pradeep, M. Rama Sai Kaushik bearing Roll Nos. 16211A04D3, 16211A04D6, 16211A04E7 has completed his/her training on Embedded systems and Implemented a Project titled **"Intelligent Coal Mine Monitoring System based on the Lora-cloud"** in Centre for Embedded Systems Laboratory, B.V. Raju Institute of Technology from **2019** to **2020**.

Coordinator

Head of the Department

ACKNOWLEDGMENTS

We take the opportunity to express our indebted gratitude to the persons who contributed to our work, for being our inspiration and guide, which led to the successful completion of the project.

We are grateful for our College Management and our beloved Principal
Dr. K. Laxmi Prasad for providing us the necessary infrastructure and facilities that ensured the smooth and satisfactory execution of the project.

We would like to express our profound gratitude to our Head of the department
Dr. I. A. Pasha, Professor & HOD, Dept. of ECE, for his encouragement inspiration, and close monitoring and guidance he gave us during the execution of the project.

We express our sincere thanks to our coordinator **Mr. R. Anirudh Reddy, Assistant Professor** and our guide **Dr. R. Balaji, Associate Professor**, Dept. of ECE, for his valuable suggestion and motivation in the successful completion of the project.

We also wish to express our thanks to all the faculty members and laboratory staff who were helpful directly and indirectly for the completion of the project.

By
M. VENKATA AKHIL (16211A04D3)
N. PRADEEP (16211A04D6)
M. RAMA SAI KAUSHIK (16211A04E7)

ABSTRACT

Today safety of miners is a major challenge. Miner's health and life are vulnerable to several critical issues, which includes not only the working environment but also the aftereffect of it. Mining activities that are carried out deep underground at high temperatures and humidity where release harmful and toxic gases take place during the process exposes the associated workers into the danger of survival. This puts a lot of pressure on the mining industry. To increase productivity and reduce the cost of mining along with consideration of the safety of workers, an innovative approach is required. Miner's health is in danger mainly because of the toxic gases which are very often released and the humidity level in underground mines. These gases cannot be detected easily by human senses. This thesis investigates the presence of toxic gases in critical regions and their effects on miners. A real-time monitoring system using a wireless sensor network, which includes multiple sensors, is developed. This system monitor surrounding environmental parameters such as temperature, humidity, and multiple toxic gases thereby providing an early warning at the monitoring station, which will be helpful to all miners present inside the mine to save their life before any casualty occurs. The system uses Lo-Ra technology to establish a wireless sensor network. It is a wireless networking standard that is suitable for operation in a harsh environment.

Key Words: Coal mines, Working environment, High temperatures, Humidity, Harmful and toxic gases, Safety of workers, Real-time monitoring system, wireless sensor network, Early warnings, Monitoring station, Lo-Ra technology.

PREFACE

As a part of the B.Tech curriculum and to gain practical knowledge in the field of Electronics and Communication we are required to make a project report on "COAL MINE MONITORING SYSTEM USING Lo-Ra". The project is prepared with a view to including all the details regarding the project that we carried out, the basic objective behind doing this project report is to get knowledge of different software tools and hardware components used.

In this project, we have included various concepts, technology, and implementation regarding the coal mine monitoring system using Lo-Ra. Subject to the limitation of time effort and resources every possible attempt has been made to study the problem deeply.

Doing this project report help us to enhance our knowledge regarding the work through this report we come to know about the importance of teamwork and the role of devotion towards the work.

The organization of the thesis is explained here with:

Chapter – 1:

This chapter gives an introduction about basically our project “coal monitoring system using Lo-Ra” is. What motivated us and the objectives of the project are discussed in this chapter.

Chapter – 2:

It provides the literature survey that has been done to back up the methodology used to come up with a better solution for further enhancement.

Chapter – 3:

This chapter deals with the analysis and design of the project. An overview of embedded systems is discussed. All the hardware components involved and software tools used to develop the project are discussed in detail.

Chapter – 4:

Implementation of the proposed system design and its functioning are discussed in this chapter. The flow chart at the end of this chapter helps in a better understanding of the steps involved in the implementation of the design.

Chapter – 5:

Hardware kit of the project and the software results are provided in this chapter.

Chapter – 6:

Conclusion and future scope of the project are discussed here.

References we considered to develop our project are provided then after.

DECLARATION

We hereby declared that the project work entitled " Intelligent Coal Mine Monitoring System based on the Lora-cloud" submitted to the JNTU Hyderabad and Department of electronics and communication engineering, BVRIT Narsapur, is a record of an original work done by M. Venkata Akhil, N. Pradeep, M. Rama Sai Kaushik bearing Roll Nos. 16211A04D3, 16211A04D6, 16211A04E7 during the period 2019 to 2020 under the guidance of Dr. R. Balaji, Associate Professor, Department of ECE, and this project work is submitted in the partial fulfillment of the requirements for the award of the degree "Bachelor of Technology" in "Electronics & Communication Engineering".

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree.

1)M. VENKATA AKHIL
16211A04D3

2) N. PRADEEP
16211A04D6

3) M. RAMA SAI KAUSHIK
16211A04E7

CONTENTS

LIST OF FIGURES

1. INTRODUCTION

1.1 Introduction

Underground mines are usually extensive labyrinths, of which the tunnels are generally long and narrow with a few kilometers in length and a few meters in width. Thousands of mining personnel are needed to work under extreme conditions according to the construction requirements, and hundreds of miners die from mining accidents every year. It is now widely approved that underground mining operations are of high risk. In this view, a monitoring and control system needs to be deployed as one important infrastructure to ensure the mining safety and coordinate various tasks. However, underground coal mines mainly consist of random passages and branch tunnels, and this disorganized structure makes it very difficult to deploy any networking skeleton. In such a case, the utilization of a wireless sensor network (WSN) and other sensing devices may have special advantages for realizing the automation of underground monitoring and control due to the rapid and flexible deployment. Also, the multiloop transmitting method can well adapt to the tunnel structure and thus provide enough scalability for the construction of a mining system, and it is very suitable for the comprehensive monitoring and controlling of coal mines, which can effectively compensate the deficiencies of the exiting underground cable monitoring system. Traditionally, coal mine safety monitoring and automation systems were typically designed to meet the requirements of a single monitoring application. The coal mine application has already gone beyond the interconnection of a few large back-end systems, and more and more underground physical devices make the state of objects and their surroundings seamlessly accessible to software systems. Most works are based on monolithic system architectures, which are brittle and difficult to adapt. A necessary step towards coal mine monitoring and control automation are to provide timely and fine-grained comprehensive alarming information and corresponding disposal process. It is necessary so that it allows the users to identify the levels for coal mine safety alarming, and possibly to adjust monitoring and control rules to ensure the coal mine safety. Furthermore, the user can also control the physical devices remotely via the Web. Currently, available coal mine safety monitoring and control systems that focus on the real-time information collection are useful, but cannot meet the user needs fully with a very high usage obstacle and often requires a complex operation definition and configuration for monitoring and control automation applications, and cannot meet the demand for ad-hoc services by the end-users. Recently, in the area of comprehensive application integration, some works have introduced the use of “mashup” concepts also known as user-generated comprehensive applications.

However, they mainly focus on mashing up information services and do not address the requirements that come with physical devices integration. The middleware for coalmine monitoring and control automation needs to rapidly coordinate interaction between the business processes and distributed, multisource sensory devices Also, the middleware for coal mine monitoring and control automation should change dynamically in are all-time way confronting with continuously and constantly changing for the underground coal mine physical world. With the help of visualization technology, the graphical user interface of different underground physical sensor devices could be created, which allows the sensors to combine with other resources easily.

1.2 Motivation

In the last decades, without any control, accidents in industries have been increasing. As human safety and property losses are important to maintain in the industrial environment there is a need to monitor the environment in the industry.

There are hazardous factors which cause risk accident like fire, gas leakage, radiation, and high-temperature source due to the combined effects of all hazardous elements, and so accidents occur in coal mine industries.

the existing monitoring system in coal mines uses cable network. Due to the explosion in a hazardous environment in the coal mine, cables get damaged completely. It could not provide the information which is helpful to rescue search and detection of events. So, the main scenario of this project is to develop a reliable communication system.

1.3 Objective

Mining environment often has hidden dangers within such as toxic gases, high temperatures, and humidity which may present severe health exposures to the people working within mining. These gases need to be detected, temperature, and humidity magnitudes there are to be informed at the right time for the safety of miners. Wired network monitoring systems have assisted the mine safety, but it is not ideal for all types of mining environments as the cables laid may be destroyed due to explosions in a hazardous environment. A real-time monitoring system based on wireless

technology can better assist in monitoring and control over the mining environment. Lo-Ra technology offers its most of the advantages ideal for building such a real-time monitoring system. Thus, the primary objective of this project is decided to design an efficient real-time monitoring system so that various leaked mine gases could be identified, temperature and humidity parameters are monitored continuously and preventive measures could be devised accordingly on time.

The research investigations to be carried out with the following objectives:

1. Detection of different toxic gases concentration, temperature, and humidity inside the mine, within the mining environment.
2. Communication establishment between sensors and Lo-Ra.
3. Establishment of Wireless Sensor Network using Lo-Ra
4. Design of a real-time monitoring system.

2. LITERATURE SURVEY

[1]. **Design of IoT Based Coal Mine Safety System Using Nodemcu**
By

Boddapati Venkata Sai Phani Gopal, Pakirabad Akash, P.S.G.Aruna Sri

In this paper, a coal mine safety system is implemented using a Thinger Io platform as a medium to transmit the data. The system is implemented to monitor and control various parameters in the coal mines such as light detection, leakage of gas, temperature and humidity conditions, Fire detection in the coal mine. These all sensors are together considered as one unit and are placed in the coal mines. All the esteems of the sensors are continuously uploaded to the thinger for analysis. Here the gas is continuously monitored if any uncertainties in the level of gas arise, then buzzer is used to alert the workers. In this system LDR sensor is utilized to detect the presence of light. Automatically light gets one and can be controlled using the LED button. In case of any fire occurs in the coal mine, then an alert notification is sent to the mail of the authorized person. Temperature and humidity values are also continuously monitored and displayed on the serial monitor and also in the thinger platform. The developed system is mainly implemented to improve the working condition inside the coal mines and also to ensure workers' safety. Keywords –Cloud Server, Safety system, Sensors, Thinger.i

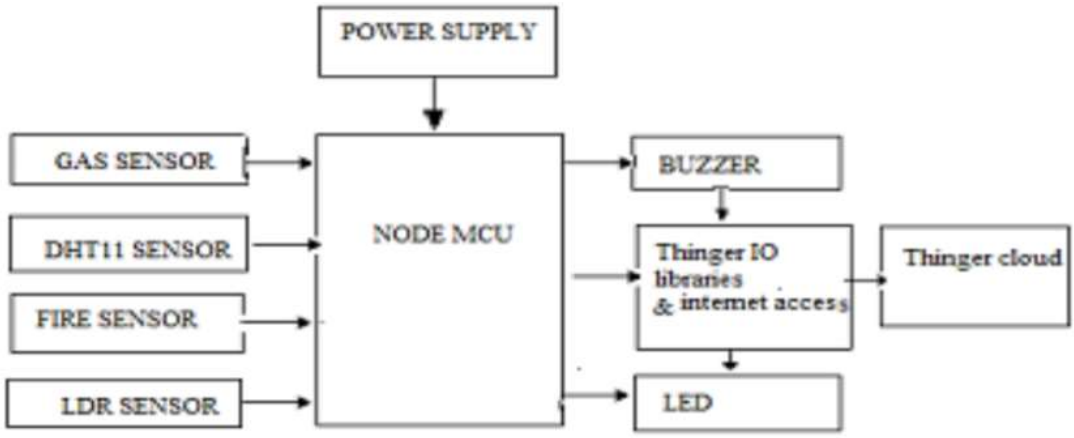


Figure 2.1 Block diagram of Design of IoT Based Coal Mine Safety System Using Nodemcu

[2] **A SMART SECURITY SYSTEM WITH MONITORING IN MINES**

By

Keerthana.E, Kiruthika.M, Kalyani Suruthi.S, Lakshmi Priya.R, Dharini.G

Department of ECE, IFET College of Engineering and Technology, Villupuram, India

This paper is about a security framework to screen the workers and offer security to them. At the entryway, there is no security on permitting all specialists inside without checking whether the workers wearing a wellbeing helmet, overcoat, and shoes. In the proposed framework the smart helmet contains a gas sensor, temperature sensor, heart rate sensor with an Audio playback recorder (APR module). At the entrance of the mines, the representatives can be observed by raspberry pi, CSI (camera opening interface) with the assistance of a camera. The picture of a worker can be caught and checked whether the worker wearing a protective cap or not. The reference database was at that point put away in raspberry pi. If the representative wearing a head protector, a picture can be caught and the caught picture is contrasted and the database will be put away in the raspberry pi, which is possible by

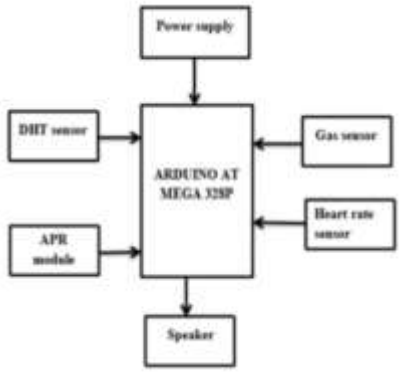


Figure 2.2 Block diagram of A Smart Security system with monitoring in mines

[3]. **Wireless Communication Based on Coal Mining Safety System with Arduino**

By

M. Rajadurai B. E, S. Aishwarya M. E Student, Department of ECE, Arjun College of Technology, Coimbatore, Assitant Professor Department of ECE, Arjun College of Technology, Coimbatore

The most important aspect of the mining workers Act is providing reliable communication for miner's accidents. Reliable communication has always been a challenge in underground mines

due to changing topologies and the environment. It's mainly detected the human heartbeat by using the pulse sensor. Also, disasters disable wired communication occurs due to the fire cause. These may damage the communication infrastructure which rescues efforts and endanger lives. Therefore, durable wireless solutions using advanced communication and sensor network technologies have been investigated to applied reliable communications in underground mines. In this paper, it designs a monitoring system for coal mine safety based on, Zig-bee to zig-bee modules with sensors. In this system, it consists of the main module for a zig-bee transceiver-based device that would cooperate to transport disaster information. This system provides an efficient, faster, and reliable communication even when a disaster occurred. The designed coal mine safety monitoring system based on wireless sensor network, the zig-bee transceiver will improve the level of monitoring production safety and reduce accidents in the coal mine. ATMEGA328 processor is used to Zig-bee transceiver information is passing to speed up the processing of data.

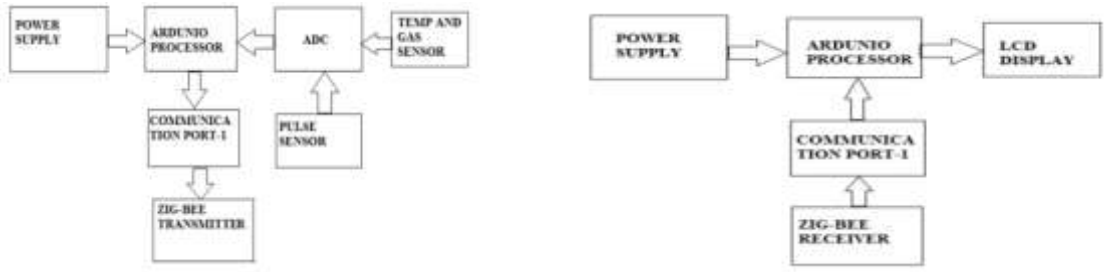


Figure 2.3 Block diagram of Wireless Communication Based on Coal Mining Safety System

[4]. **Using Zigbee Integrated Alerting and Coal Mine Safety Monitoring System**

By

Shilpa Lande PG student, E&TC Department, G. H. Raisoni College of Engineering, Chas, Ahmednagar, India

The coal mine safety system replaces traditional coal mine safety which is a wired system. This monitoring system for coal mines based on ZIGBEE wireless sensor network. Because of this, there is an important development in coal mine safety production which is safe. Apart from this, it is unsuitable to lay the cables in life which is costly and consumes more time. To solve this ambiguity there is a need for the design and development of coal mine safety using WSN. Because

of this, there is an improvement in production safety and reduces accidents. The purpose of this project is to propose a solution suitable to mine wireless communication, safety monitoring, and give proof to further study.

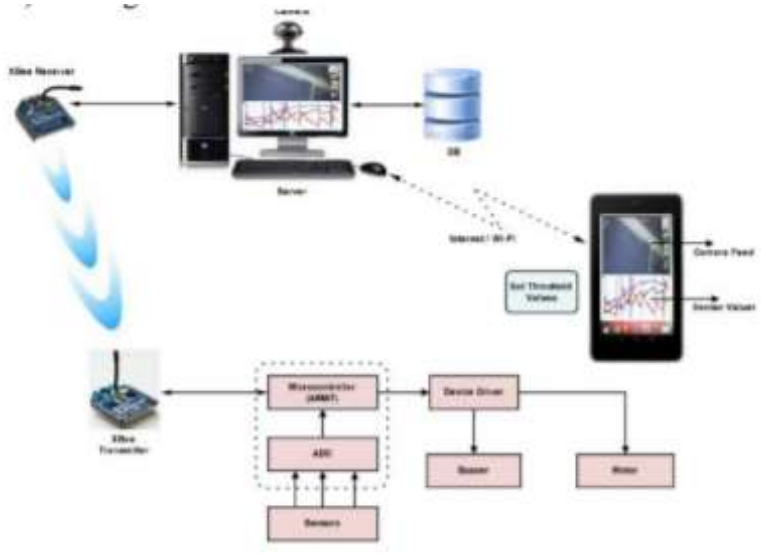


Figure 2.4 Block diagram of Using Zigbee Integrated Alerting and Coal Mine Safety Monitoring System

3. ANALYSIS & DESIGN

3.1 Overview of Embedded System

An embedded system is a special-purpose computer system designed to perform one or a few dedicated functions, often with real-time computing constraints. It is usually embedded as part of a complete device including hardware and mechanical parts. In contrast, a general-purpose computer, such as a personal computer, can do many different tasks depending on programming. Embedded systems have become very important today as they control many of the common devices we use.

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.

In general, an "embedded system" is not an exactly defined term, as many systems have some element of programmability. For example, Handheld computers share some elements with embedded systems — such as the operating systems and microprocessors which power them — but are not truly embedded systems, because they allow different applications to be loaded and peripherals to be connected.

Embedded systems provide several functions

Monitor the environment; embedded systems read data from input sensors. This data is then processed and the results displayed in some format to a user or users

Control the environment; embedded systems generate and transmit commands for actuators.

Transform the information; embedded systems transform the data collected in some meaningful way, such as data compression/decompression

Although interaction with the external world via sensors and actuators is an important aspect of embedded systems, these systems also provide functionality specific to their applications. Embedded systems typically execute applications such as control laws, finite state machines, and signal processing algorithms. These systems must also detect and react to faults in both the internal computing environment as well as the surrounding electromechanical systems.

Block diagram of Embedded System

An embedded system usually contains an embedded processor. Many appliances that have a digital interface -- microwaves, VCRs, cars -- utilize embedded systems. Some embedded systems include an operating system. Others are very specialized resulting in the entire logic being implemented as a single program. These systems are embedded into some device for some specific purpose other than to provide general-purpose computing. A typical embedded system is shown in Fig 1.1

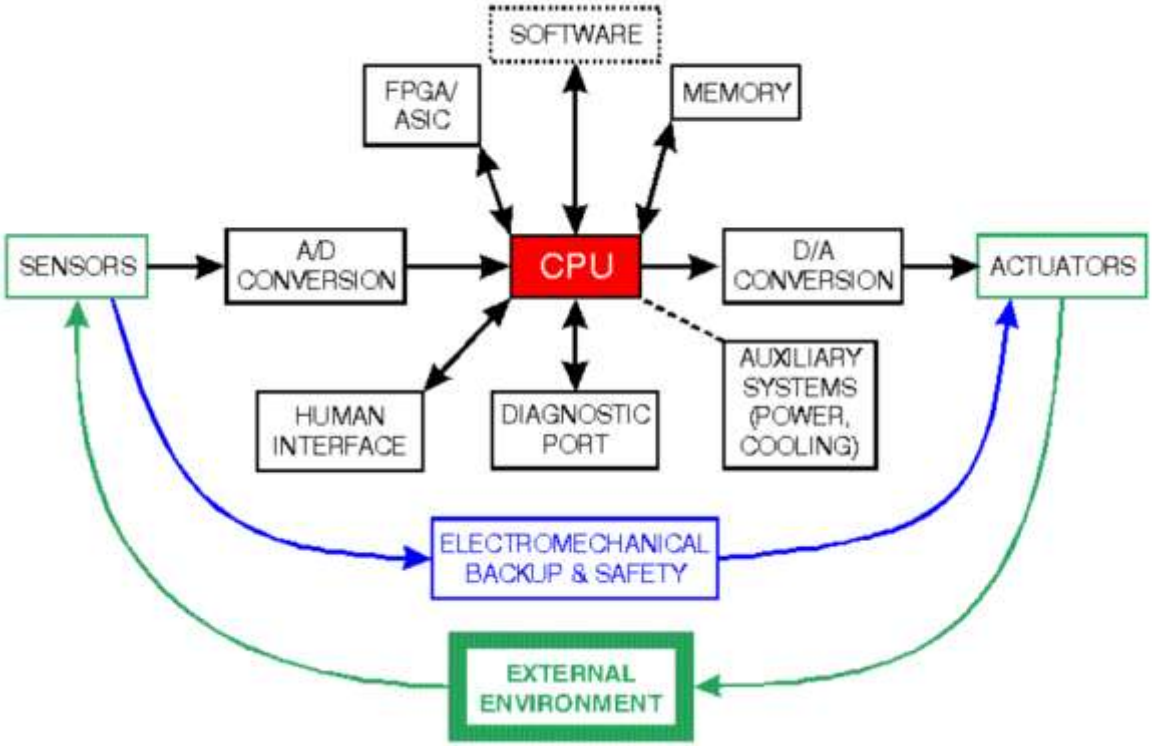


Figure 3.1 Block diagram of a typical Embedded System

Characteristics of Embedded System

Embedded systems are characterized by a unique set of characteristics. Each of these characteristics imposed a specific set of design constraints on embedded systems designers. The challenge of designing embedded systems is to conform to the specific set of constraints for the application.

Application-specific systems

Embedded systems are not general-purpose computers. Embedded system designs are optimized for a specific application. Many of the job characteristics are known before the hardware is designed. This allows the designer to focus on the specific design constraints of a well-defined application. As such, there is limited user reprogram ability. Some embedded systems, however, require the flexibility of reprogramming ability. Programmable DSPs are common for such applications.

Reactive systems

As mentioned earlier, a typical embedded systems model responds to the environment via sensors and control the environment using actuators. This requires embedded systems to run at the speed of the environment. This characteristic of an embedded system is called “reactive”. Reactive computation means that the system (primarily the software component) executes in response to external events. External events can be either periodic or aperiodic. Periodic events make it easier to schedule processing to guarantee performance. A periodic event is harder to schedule. The maximum event arrival rate must be estimated to accommodate worst-case situations. Most embedded systems have a significant reactive component. One of the biggest challenges for embedded system designers in performing accurate worst-case design analysis on systems with statistical performance characteristics (e.g., cache memory on a DSP or other embedded processor). Real-time system operation means that the correctness of a computation depends, in part, on the time at which it is delivered. Systems with this requirement must often design to worst-case performance. But accurately predicting the worst case may be difficult for complicated architectures. This often leads to overly pessimistic estimates erring on the side of caution. Many embedded systems have a significant requirement for real-time operation to meet external I/O and control stability requirements. Many real-time systems are also reactive systems.

Distributed Systems

A common characteristic of an embedded system is one that consists of communicating processes executing on several CPUs or ASICs which are connected by communication links. The reason for this is the economy. Economical 4 8-bit microcontrollers may be cheaper than 32-bit processors. Even after adding the cost of the communication links, this approach may be preferable. In this approach, multiple processors are usually required to handle multiple time-critical tasks. Devices under control of embedded systems may also be physically distributed.

Heterogeneous Architectures

Embedded systems often are composed of heterogeneous architectures (Fig 1.2). They may

contain different processors in the same system solution. They may also be mixed-signal systems. The combination of I/O interfaces, local and remote memories, and sensors and actuators makes embedded system design truly unique. Embedded systems also have tight design constraints, and heterogeneity provides better design flexibility.

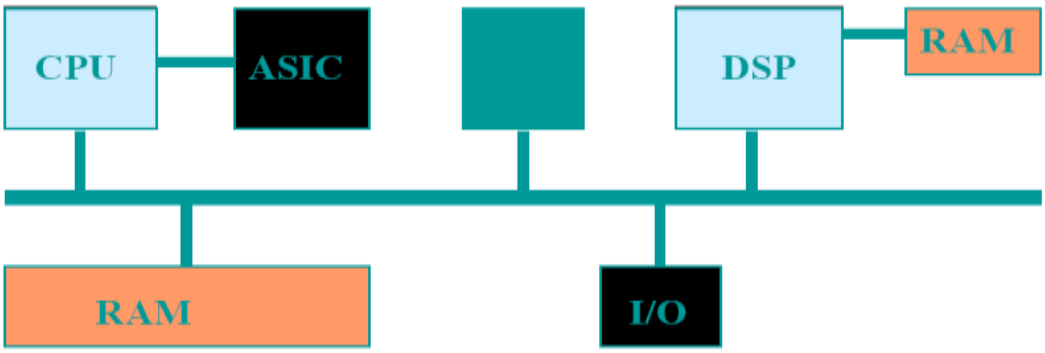


Figure 3.2 Embedded Systems having Heterogeneous Architectures

Harsh environment

Many embedded systems do not operate in a controlled environment. Excessive heat is often a problem, especially in applications involving combustion (e.g., many transportation applications). Additional problems can be caused by embedded computing by a need for protection from vibration, shock, lightning, power supply fluctuations, water, corrosion, fire, and general physical abuse.

System safety and Reliability

As embedded system complexity and computing power continue to grow, they are starting to control more and more of the safety aspects of the overall system. These safety measures may be in the form of software as well as hardware control. Mechanical safety backups are normally activated when the computer system loses control to safely shut down system operation. Software safety and reliability is a bigger issue. The software doesn't normally "break" in the sense of hardware. However, the software may be so complex that a set of unexpected circumstances can cause software failures leading to unsafe situations. Discussion of this topic is outside the scope of this book, but the challenges for embedded designers include designing reliable software and building cheap, available systems using unreliable components. The main challenge for embedded system designers is to obtain low-cost reliability with minimal redundancy.

Control of Physical Systems

One of the main reasons for embedding a computer is to interact with the environment. This is

often done by monitoring and controlling external machinery. Embedded computers transform the analog signals from sensors into digital form for processing. Outputs must be transformed back to analog signal levels. When controlling physical equipment, large current loads may need to be switched to operate motors and other actuators. To meet these needs, embedded systems may need large computer circuit boards with many non-digital components. Embedded system designers must carefully balance system tradeoffs among analog components, power, mechanical, network, and digital hardware with corresponding software.

Small and low weight

Many embedded computers are physically located within some larger system. The form factor for the embedded system may be dictated by aesthetics. For example, the form factor for a missile may have to fit inside the nose of the missile. One of the challenges for embedded systems designers is to develop non-rectangular geometries for certain solutions. Weight can also be a critical constraint. Embedded automobile control systems, for example, must be lightweight for fuel economy. Portable CD players must be lightweight for portability purposes.

Cost sensitivity

Cost is an issue in most systems, but the sensitivity to cost changes can vary dramatically in embedded systems. This is mainly due to the effect of computer costs have on profitability and is more a function of the proportion of cost changes compared to the total system cost.

Power Management

Embedded systems have strict constraints on power. Given the portability requirements of many embedded systems, the need to conserve power is important to maintain battery life as long as possible. Minimization of heat production is another obvious concern for embedded systems

3.2 Hardware Components

3.2.1 DHT11 sensor:

The DHT11 is a basic, ultra-low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air and spits out a digital signal on the data pin (no analog input pins needed). It is fairly simple to use but requires careful timing to grab data. The only real downside of this sensor is you can only get new data from it once every 2 seconds.

Figure 3.3 DHT11 sensor

Features:

Low cost
3 to 5V power and I/O
2.5mA max current use during conversion (while requesting data)
Good for 20-80% humidity readings with 5% accuracy
Good for 0-50°C temperature readings ±2°C accuracy
No more than 1 Hz sampling rate (once every second)
Body size 15.5mm x 12mm x 5.5mm
4 pins with 0.1" spacing

3.2.2 Smoke sensor

Catalytic bead sensors are used primarily to detect combustible gases. They have been in use for more than 50 years. Initially, these sensors were used for monitoring gas in coal mines, where they replaced canaries that had been used for a long period.

The sensor itself is quite simple in design and is easy to manufacture. In its simplest form, as used in the original design, it was comprised of a single platinum wire. Catalytic bead sensors were produced all over the world by a large number of different manufacturers, but the performance and reliability of these sensors varied widely among these various manufacturers.

Figure 3.4 Smoke sensor

Principle of Operation

Combustible gas mixtures will not burn until they reach ignition temperature. However, in the presence of certain chemical media, the gas will start to burn or ignite at lower temperatures. This phenomenon is known as catalytic combustion. Most metal oxides and their compounds have these catalytic properties. For instance, volcanic rock, which is comprised of various metal oxides,

is often placed in gas-burning fireplaces. This is not only decorative, but it also helps the combustion process and results in cleaner and more efficient burning in the fireplace. Platinum, palladium, and thoria compounds are also excellent catalysts for combustion. This explains why the automobile exhaust system is treated with platinum compounds and is called a catalytic converter. This kind of gas sensor is made based on the catalytic principle and therefore is called the catalytic gas sensor. A gas molecule oxidizes on the catalyzed surface of the sensor at a much lower temperature than its normal ignition temperature. All electrically conductive materials change their conductivity as temperature changes. This is called the coefficient of temperature resistance (Ct). It is expressed as the percentage of change per degree change in temperature. Platinum has a large Ct in comparison to other metals. Also, its Ct is linear between 500°C to 1000°C, which is the temperature range at which the sensor needs to operate. Because the signal from the sensor is linear, this means that the concentration of gas readings is in direct proportion to the electrical signal. This improves accuracy and simplifies the electronic circuitry. Also, platinum possesses excellent mechanical properties. It is physically strong and can be transformed into a fine wire which can be processed into small sensor beads.

Furthermore, platinum has excellent chemical properties. It is corrosion resistant and can be operated at elevated temperatures for a long period without changing its physical properties. It is capable of producing a constant reliable signal over an extended period

3.2.3 Heartbeat sensor

In this experiment you will not record the pulse wave over an artery but from the tip of the finger; this is called a peripheral pulse. It is recorded using a photoelectric pulse transducer, which measures changes in blood volume (plethysmography). A light source in the transducer transilluminates the fingertip, and a photoconductor detects changes in light intensity within the finger caused by pulsatile variations in blood volume.

Plethysmography is the most widely used measurement technique for assessing peripheral blood flow. The technique is based on the differential light-absorbing characteristics of tissue and blood. Specifically, living tissue is relatively transparent to light in the infrared range, whereas blood is relatively opaque to light within this frequency range (Fig. 1).

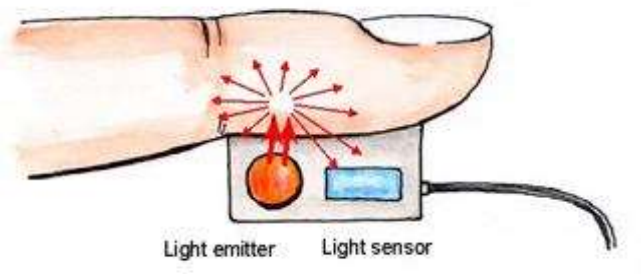


Figure 3.5 Optical finger plethysmograph

Procedures

With the subject seated, attach the transducer snugly to the palmar surface of the middle fingertip.

Record the pulse for 10 seconds with the subject's arm resting on the lab table.

Now have the subject raise the transducer above his/her head (arm extended) for 30 seconds and record the pulse during the last 10 seconds.

Then have his/her lower the transducer (arm hanging at his/her side) for 30 seconds and record during the last 10 seconds.

Plethysmography measurement of systolic blood pressure:

Blood pressure results from two forces. One is created by the heart as it pumps blood into the arteries and through the circulatory system. The other is the force of the arteries as they resist the blood flow.

The systolic pressure is always stated first and the diastolic pressure second. For example, 118/76. The higher (systolic) number represents the pressure while the heart contracts to pump blood to the body. The lower (diastolic) number represents the pressure when the heart relaxes between beats. Blood pressure below 120 over 80 mmHg (millimeters of mercury) is considered best for adults.

The most accurate method for arterial pressure measurement is a direct intra-arterial measurement with a catheter. However, this technique is neither practical.

The indirect Korotkoff method of measurement is commonly used. With this technique, the pressure required to collapse the artery in the upper arm or leg is determined by the use of a sphygmomanometer (an occluding cuff, stethoscope, and manometer). The cuff is inflated to a level above arterial pressure (as indicated by the obliteration of the pulse). As the cuff is gradually deflated, the pressure is noted at which sounds produced by the arterial pulse waves (Korotkoff sounds) appear and disappear again as flow through the artery resumes. Indirect measurement Korotkoff method is sufficiently accurate for many diagnostic and therapeutic studies and is continue to be used because it is practical, simple, low in cost, and non-invasion.

It this experiment a photoelectric pulse transducer, which measures changes in blood volume (plethysmography) is used. A light source in the transducer transilluminates the fingertip and a photoconductor detects changes in light intensity within the finger caused by pulsatile variations in blood volume. Plethysmography is the most widely used measurement technique for assessing peripheral blood flow. The technique is based on the differential light-absorbing characteristics of tissue and blood. Specifically, living tissue is relatively transparent to light in the infrared range, whereas blood is relatively opaque to light within this frequency range (Fig. 1).

Interfacing with Opto Devices

An Optocoupler (or optoisolator) is an electronic component with an LED and photo-sensitive device, such as a photodiode or phototransistor encased in the same package. The Opto-coupler which we look at in a previous tutorial interconnects two separate electrical circuits using a light-sensitive optical interface. This means that we can effectively interface two circuits of different voltage or power ratings together without one electrically affecting the other.

Optical Switches (or opto-switches) are another type of optical (photo) switching devices which can be used for input interfacing. The advantage here is that the optical switch can be used for input interfacing harmful voltage levels onto the input pins of microcontrollers, PICs and other such digital circuits or for detecting objects using light as the two components are electrically separate but optically coupled providing a high degree of isolation (typically 2-5kV).

Optical switches come in a variety of different types and designs for use in a whole range of interfacing applications. The most common use for Opto-switches is in the detection of moving or stationary objects. The phototransistor and photodarlington configurations provide most of the features required for photo-switches and are therefore the most commonly used.

As well as using slotted or reflective photoswitches for the input interfacing of circuits, we can also use other types of semiconductor light detectors such as photo resistive light detectors, PN junction photodiodes, and even solar cells. All these photosensitive devices use ambient light such as sunlight or normal room light to activate the device allowing them to be easily interfaced with any type of electronic circuit.

Normal signal and power diodes have their PN junction sealed within a plastic body for both safeties and to stop photons of light from hitting it. When a diode is reverse biased it blocks the flow of current, acting as a high resistance open switch. However, if we were to shine a light onto this PN junction the photons of light open up the junction allowing current to flow depending upon the intensity of the light on the junction.

Photodiodes exploit this by having a small transparent window that allows light to strike their PN junction making the photodiode extremely photosensitive. Depending upon the type and amount

of semiconductor doping, some photodiodes respond to visible light, and some to infrared (IR) light. When there is no incident light, the reverse current, is almost negligible and is called the "dark current". An increase in the amount of light intensity produces an increase in the reverse current.

Then we can see that a photodiode allows reverse current to flow in one direction only which is the opposite of a standard rectifying diode. This reverse current only flows when the photodiode

receives a specific amount of light acting as very high impedances under dark conditions and as low impedance devices under bright light conditions and as such of the photodiode can be used in many applications as a high-speed light detector.

Interfacing Photodiodes

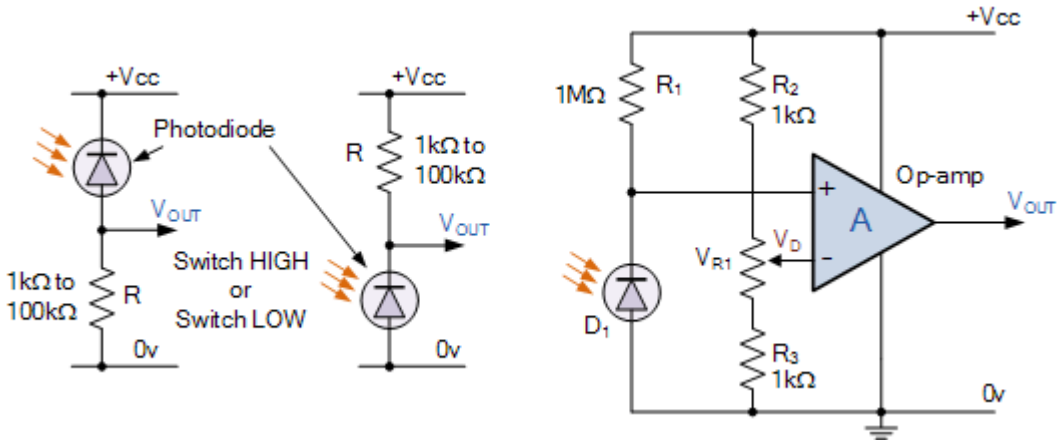


Figure 3.6 Interfacing of photodiodes and circulatory

In the two basic circuits on the left, the photodiode is simply reverse-biased through the resistor with the output voltage signal taken from across the series resistor. This resistor can be of a fixed value, usually between the 10kΩ to 100kΩ range, or as a variable 100kΩ potentiometer as shown. This resistor can be connected between the photodiode and 0v ground, or between the photodiode and the positive Vcc supply.

While photodiodes such as the BPX48 give a very fast response to changes in light level, they can be less sensitive compared to other photo-devices such as the Cadmium Sulphided LDR cell so some form of amplification in the form of a transistor or op-amp may be required. Then we have seen that the photodiode can be used as a variable-resistive device controlled by the amount of light falling on its junction. Photodiodes can be a switch from "ON" to "OFF" and back very fast sometimes in nano-seconds or with frequencies above 1MHz and so are commonly used in optical encoders and fiber optic communications.

As well as PN junction photo devices, such as the photodiode or the phototransistor, other types of semiconductor light detectors operate without a PN junction and change their resistive characteristics with changes or variations in light intensity. These devices are called Light Dependent Resistors, or LDR's.

The LDR, also known as a cadmium-sulfide (CdS) photocell, is a passive device with a resistance that varies with visible light intensity. When no light is present their internal resistance is very high in the order of mega-ohms (MΩ's). However, when illuminated their resistance falls to below 1kΩ in strong sunlight. Then light-dependent resistors operate similarly to potentiometers but with light intensity controlling their resistive value.

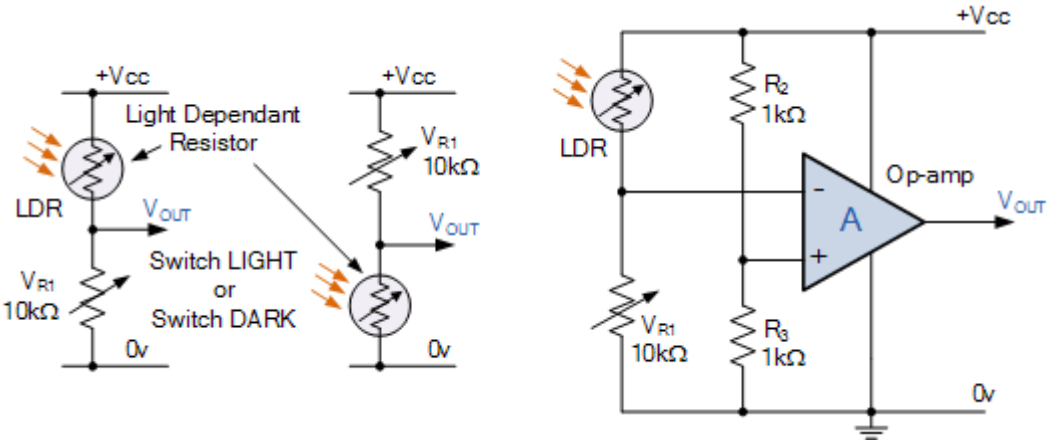


Figure 3.7 Interfacing of LDR photoresistors and circulatory

Light-dependent resistors change their resistive value in proportion to the light intensity. Then LDRs can be used with a series resistor, R to form a voltage divider network across the supply. In the dark, the resistance of the LDR is much greater than that of the resistor so by connecting the LDR from supply to resistor or resistor to ground, it can be used as a light detector or as a dark detector as shown.

As LDRs such as the NORP12, produce a variable voltage output relative to their resistive value, they can be used for analog input interfacing circuits. But LDRs can also be connected as part of a Wheatstone Bridge arrangement as the input of an op-amp voltage comparator or a Schmitt trigger circuit to produce a digital signal for interfacing to digital and microcontroller input circuits.

Simple threshold detectors for either light level, temperature or strain can be used to produce TTL compatible outputs suitable for interfacing directly to a logic circuit or digital input port. Light and temperature level threshold detectors based on an op-amp comparator generate a logic "1" or a logic "0" input whenever the measured level exceeds or falls below the threshold setting.

3.2.4 Methane Gas sensor

Ideal sensor for use to detect the presence of a dangerous LPG leak in your car or a service station, storage tank environment. This unit can be easily incorporated into an alarm unit, to sound an alarm or give a visual indication of the LPG concentration. The sensor has excellent sensitivity combined with the quick response time. The sensor can also sense iso-butane, propane, LNG, and cigarette smoke.

Figure 3.8 Methane gas sensor module

Applications

- The gas leak detection system
- Fire/Safety detection system
- Gas leak alarm
- Gas detector

Features

- High Sensitivity
- Detection Range: 100 - 10,000 ppm iso-butane propane
- Fast Response Time: <10s
- Heater Voltage: 5.0V
- Dimensions: 18mm Diameter, 17mm High excluding pins, Pins - 6mm High

3.2.5 Lo-Ra module

LORA is a Long-Range communication module, which can be used for a Long-range up to 2 KM based on serial communication. It is applicable in many ways like Point to point communication or Node-based communication Lo-Ra uses license-free sub-gigahertz radio frequency bands like 433 MHz, 868 MHz (Europe), 915 MHz (Australia and North America) and 923 MHz (Asia). Lo-Ra enables long-range transmissions (more than 10 km in rural areas) with low power consumption. The technology covers the physical layer, while other technologies and protocols such as Lo-Ra WAN (Long Range Wide Area Network) cover the upper layers.

In January 2018, new Lo-Ra chipsets were announced, with reduced power consumption, increased transmission power, and reduced size compared to the older generation.
Lo-Ra devices have geolocation capabilities used for trilateration positions of devices via timestamps from gateways. Lo-Ra and Lo-Ra WAN permit long-range connectivity for the Internet of Things (IoT) devices in different types of industries.

Figure 3.9 LORA module

Specification

- Operating frequency: 866 MHz
- SPI Data Interface
- Sensitivity: -136dBm
- Output Power: +20dBm
- Data Rate: <300 kbps
- Built-in bit synchronizer for clock recovery
- Working Temperature: -40°C ~+80°C
- Standby current: ≤ 1uA
- Supply voltage: 1.8~3.6V
- Baud rate: 1200-115200 symbols/set his Lora Module has Center Frequency of 868 MHz with 20 DB
- It has a coverage range up to 2 KM
- It is Serial communication device
- It has features like Point to Point, Repeaters by data hopping

3.2.6 Arduino

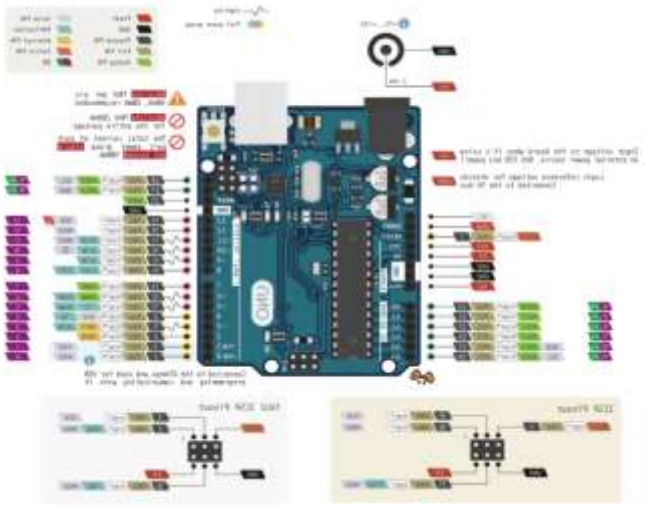


Figure 3.10 Arduino UNO

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, 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 an AC-to-DC adapter or battery to get started.

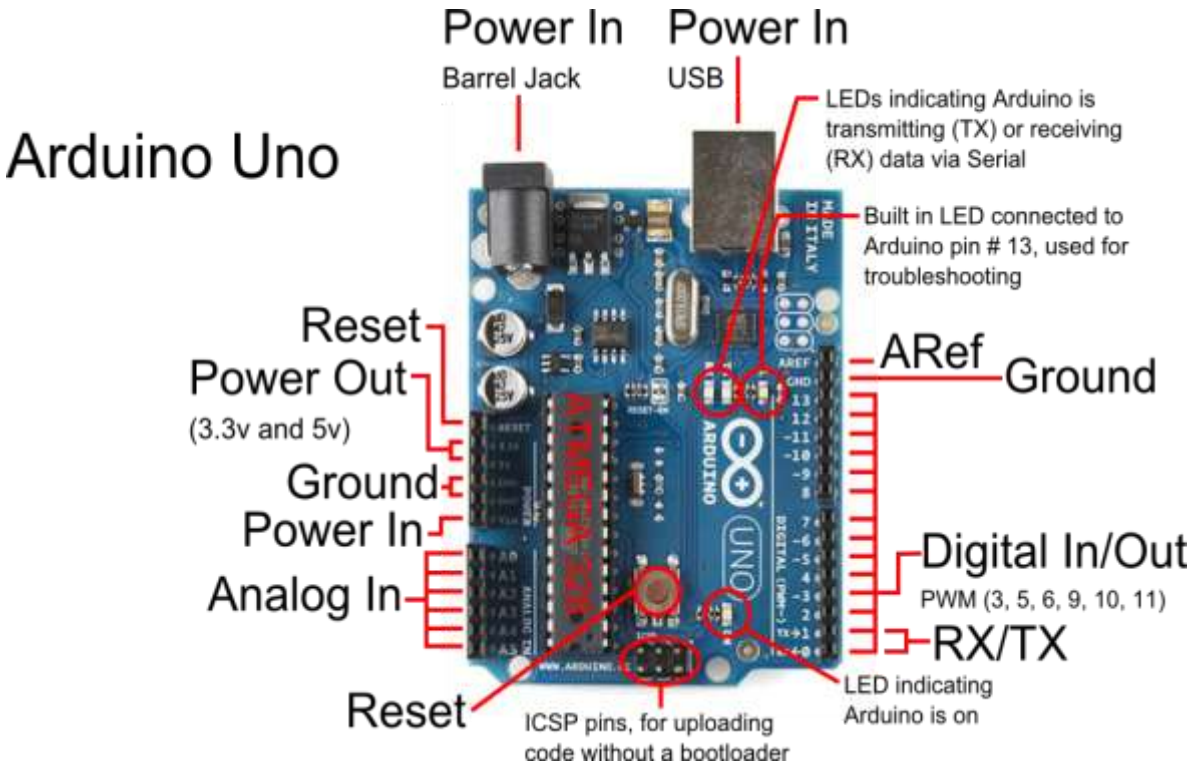


Figure 3.11 Arduino peripherals description

These boards below use the same micro-controller, just in a different package. The Lilypad is designed for use with conductive thread instead of wire and the Arduino Mini is simply a smaller package without the USB, Barrel Jack, and Power Outs.

It depends on what you want to do with it. There are two different purposes outlined above for the voltage divider, we will go over both.

If you wish to use the voltage divider as a sensor reading device first you need to know the maximum voltage allowed by the analog inputs you are using to read the signal. On an Arduino this is 5V. So, already we know the maximum value we need for Vout. The Vin is simply the amount of voltage already present on the circuit before it reaches the first resistor. You should be able to find the maximum voltage your sensor outputs by looking on the Datasheet, this is the maximum amount of voltage your sensor will let through given the voltage in of your circuit. Now we have exactly one variable left, the value of the second resistor. Solve for R2 and you will have all the components of your voltage divider figured out! We solve for R1's highest value because a smaller resistor will simply give us a smaller signal which will be readable by our analog inputs.

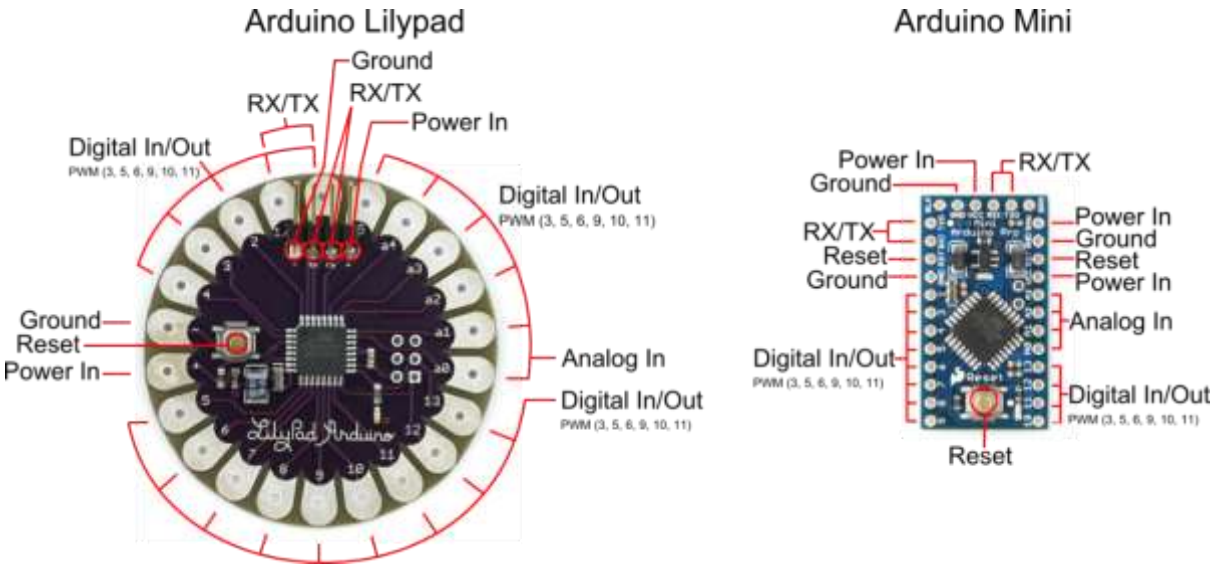


Figure 3.12 Arduino lilypad And Arduino mini peripherals description

Powering an analog Reference is the same as reading a sensor except you have to calculate for the Voltage Out value you want to use as the analog Reference.

All of the electrical signals that the Arduino works with are either Analog or Digital. It is extremely important to understand the difference between these two types of signals and how to manipulate the information these signals represent.

PIN DIAGRAM:

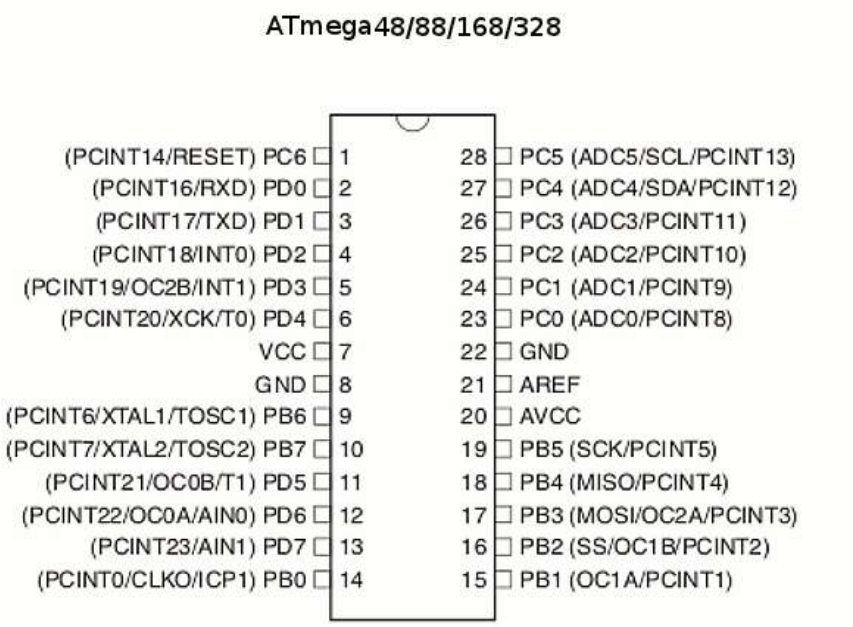


Figure 3.13 Pin description of AT mega

Digital

An electronic signal transmitted as binary code that can be either the presence or absence of current, high and low voltages or short pulses at a particular frequency.

Humans perceive the world in analog, but robots, computers, and circuits use Digital. A digital signal is a signal that has only two states. These states can vary depending on the signal, but simply defined the states are ON or OFF, never in between.

In the world of Arduino, Digital signals are used for everything except Analog Input. Depending on the voltage of the Arduino the ON or HIGH of the Digital signal will be equal to the system voltage, while the OFF or LOW signal will always equal 0V. This is a fancy way of saying that on a 5V Arduino the HIGH signals will be a little under 5V and on a 3.3V Arduino the HIGH signals will be a little under 3.3V.

To receive or send Digital signals the Arduino uses Digital pins # 0 - # 13. You may also set up your Analog In pins to act as Digital pins. To set up Analog In pins as Digital pins use the command:

pinMode (pinNumber, value);

where pinNumber is an Analog pin (A0 – A5) and value is either INPUT or OUTPUT. To setup, Digital pins use the same command but reference a Digital pin for pinNumber instead of an Analog pin. Digital pins default as input, so really you only need to set them to OUTPUT in pinMode. To read these pins use the command:

digitalRead(pinNumber);

where pinNumber is the Digital pin to which the Digital component is connected. The digitalRead command will return either a HIGH or a LOW signal. To send a Digital signal to a pin using the command:

digitalWrite(pinNumber, value);

where pinNumber is the number of the pin sending the signal and value is either HIGH or LOW.

The Arduino also can output a Digital signal that acts as an Analog signal, this signal is called Pulse Width Modulation (PWM). Digital Pins # 3, # 5, # 6, # 9, # 10 and #11 have PWM capabilities. To output a PWM signal uses the command:

analogWrite(pinNumber, value);

where pinNumber is a Digital Pin with PWM capabilities and value is a number between 0 (0%) and 255 (100%). For more information on PWM see the PWM worksheets or S.I.K. circuit 12.

THINGS TO REMEMBER ABOUT DIGITAL:

Digital Input/Output uses the Digital pins, but Analog In pins can be used as Digital

To receive a Digital signal use: digitalRead(pinNumber);

To send a Digital signal use: digitalWrite(pinNumber, value);

Digital Input and Output are always either HIGH or LOW

All of the electrical signals that the Arduino works with are either Analog or Digital. It is extremely important to understand the difference between these two types of signals and how to manipulate the information these signals represent.

Analog

Humans perceive the world in analog. Everything we see and hear is a continuous transmission of information to our senses. The temperatures we perceive are never 100% hot or 100% cold, they are constantly changing between our ranges of acceptable temperatures. (And if they are out of our range of acceptable temperatures then what are we doing there?) This continuous stream is what defines analog data. Digital information, the complementary concept to Analog, estimates analog data using only ones and zeros.

In the world of Arduino, an Analog signal is simply a signal that can be HIGH (on), LOW (off), or anything in between these two states. This means an Analog signal has a voltage value that can be anything between 0V and 5V (unless you mess with the Analog Reference pin). Analog allows you to send output or receive input about devices that run at percentages as well as on and off. The Arduino does this by sampling the voltage signal sent to these pins and comparing it to a voltage reference signal (5V). Depending on the voltage of the Analog signal when compared to the Analog Reference signal the Arduino then assigns a numerical value to the signal somewhere

between 0 (0%) and 1023 (100%). The digital system of the Arduino can then use this number in calculations and sketches.

To receive Analog Input the Arduino uses Analog pins # 0 - # 5. These pins are designed for use with components that output Analog information and can be used for Analog Input. There is no setup necessary, and to read them use the command:

analogRead(pinNumber);

where pinNumber is the Analog In pin to which the Analog component is connected. The analogRead command will return a number including or between 0 and 1023.

The Arduino also can output a digital signal that acts as an Analog signal, this signal is called Pulse Width Modulation (PWM). Digital Pins # 3, # 5, # 6, # 9, # 10 and #11 have PWM capabilities. To output a PWM signal uses the command:

analogWrite(pinNumber, value);

where pinNumber is a Digital Pin with PWM capabilities and value is a number between 0 (0%) and 255 (100%). On the Arduino UNO, PWM pins are signified by a ~ sign. For more information on PWM see the PWM worksheets or S.I.K. circuit 12.

THINGS TO REMEMBER ABOUT ANALOG:

Analog Input uses the Analog In pins, Analog Output uses the PWM pins

To receive an Analog signal use: analogRead(pinNumber);

To send a PWM signal use: analogWrite(pinNumber, value);

Analog Input values range from 0 to 1023 (1024 values because it uses 10 bits, 2^{10})

PWM Output values range from 0 to 255 (256 values because it uses 8 bits, 2^8)

All of the electrical signals that the Arduino works with are either input or output. It is extremely important to understand the difference between these two types of signals and how to manipulate the information these signals represent.

OUTPUT SIGNALS

Output to the Arduino pins is always Digital, however, there are two different types of Digital Output; regular Digital Output and Pulse Width Modulation Output (PWM). The output is only possible with Digital pins # 0 - # 13. The Digital pins are preset as Output pins, so unless the pin was used as an Input in the same sketch, there is no reason to use the pinMode command to set the pin as an Output. Should a situation arise where it is necessary to reset a Digital pin to Output from Input use the command:

pinMode(pinNumber, OUTPUT);

where pinNumber is the Digital PIN set as Output. To send a Digital Output signal use the command:

digitalWrite(pinNumber, value);

where pinNumber is the Digital pin that is outputting the signal and value is the signal. When outputting a Digital signal value can be either HIGH (On) or LOW (Off).

analogWrite(pinNumber, value);

where pinNumber is a Digital Pin with PWM capabilities and value is a number between 0 (0%) and 255 (100%). For more information on PWM see the PWM worksheets or S.I.K. circuit 12.

Output can be sent to many different devices, but it is up to the user to figure out which kind of Output signal is needed, hook up the hardware and then type the correct code to properly use these signals.

THINGS TO REMEMBER ABOUT OUTPUT:

The output is always Digital

There are two kinds of Output: regular Digital or PWM (Pulse Width Modulation)

To send an Output signal use analogWrite(pinNumber, value); (for analog) or digitalWrite(pinNumber, value); (for digital)

Output pin mode is set using the pinMode command: pinMode(pinNumber, OUTPUT);

Regular Digital Output is always either HIGH or LOW

PWM Output varies from 0 to 255

All of the electrical signals that the Arduino works with are either input or output. It is extremely important to understand the difference between these two types of signals and how to manipulate the information these signals represent.

INPUT SIGNALS

Analog Input enters your Arduino through the Analog In pins # 0 - # 5. These signals originate from analog sensors and interface devices. These analog sensors and devices use voltage levels to communicate their information instead of a simple yes (HIGH) or no (LOW). For this reason, you cannot use a digital pen as an input pin for these devices. Analog Input pins are used only for receiving Analog signals. It is only possible to read the Analog Input pins so there is no command necessary in the setup( ) function to prepare these pins for input. To read the Analog Input pins use the command:

analogRead(pinNumber);

where pinNumber is the Analog Input PIN. This function will return an Analog Input reading

between 0 and 1023. A reading of zero corresponds to 0 Volts and a reading of 1023 corresponds to 5 Volts. These voltage values are emitted by analog sensors and interfaces. If you have an Analog Input that could exceed Vcc + .5V you may change the voltage that 1023 corresponds to by using the Aref pin. This pin sets the maximum voltage parameter your Analog Input pins can read. The Aref pin's preset value is 5V.

Digital Input can enter your Arduino through any of the Digital Pins # 0 - # 13. Digital Input signals are either HIGH (On, 5V) or LOW (Off, 0V). Because the Digital pins can be used either as input or output you will need to prepare the Arduino to use these pins as inputs in your setup( )function. To do this type the command:

pinMode(pinNumber, INPUT);

inside the curly brackets of the setup( ) function where pinNumber is the Digital PIN you wish to declare as an input. You can change the pinMode in the loop( )function if you need to switch a pin back and forth between input and output, but it is usually set in the setup( )function and left untouched in the loop( )function. To read the Digital pins set as inputs use the command:

digitalRead(pinNumber);

where pinNumber is the Digital Input PIN.

Input can come from many different devices, but each device's signal will be either Analog or Digital, it is up to the user to figure out which kind of input is needed, hook up the hardware and then type the correct code to properly use these signals.

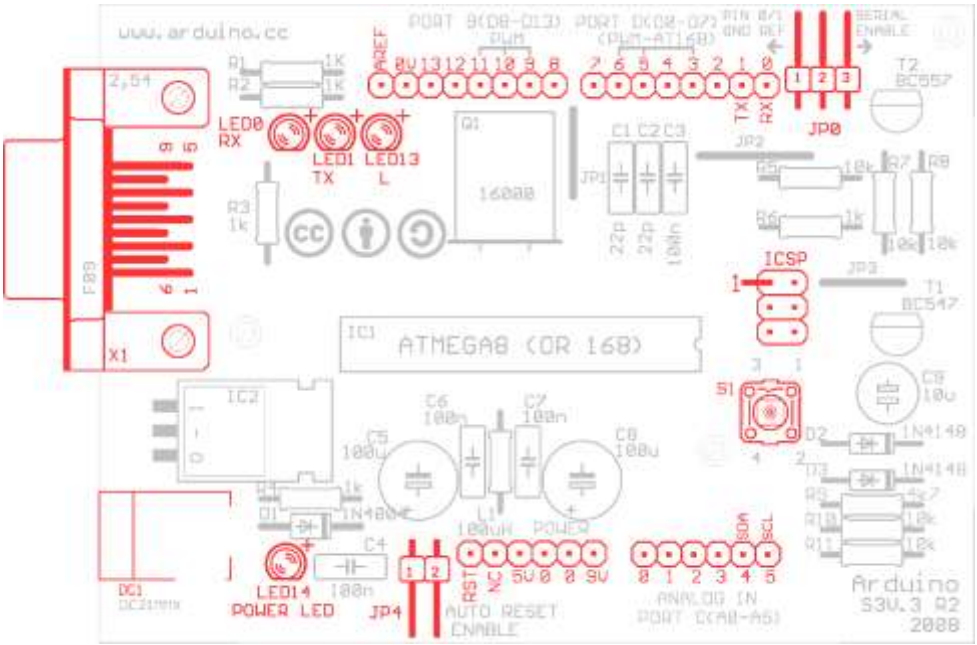


Figure 3.14 Pcb design of Arduino UNO

X1:

DE-9 serial connector

Used to connect a computer (or other devices) using the RS-232 standard. Needs a serial cable, with at least 4 pins connected: 2, 3, 4, and 5. Works only when JP0 is set to 2-3 positions.

DC1:

2.1 mm. power jack

Used to connect the external power source. Centre positive. Voltage Regulator Works with regulated +7 to +20 volts DC (9v. to 12v. is recommended). It is possible to alternatively connect external power using 9v. pin or 5v. pin. (see POWER PINOUT)

ICSP:

2x3 pin header Used to program Atmega with the bootloader. The number 1 on both sides of the board indicates a cable pin1 position. Used to upload sketches on Atmega ICs without bootloader (available only in Arduino IDE versions 0011 and 0012).

JP0

3 pins jumper When in position 2-3, this jumper enables serial connection (through X1 connector) to/from computer/devices. Use this as the default position. When in position 1-2, it disables serial communication and enables external pull-down resistors on pin0 (RX) and pin1 (TX). Use this only to prevent noise on RX (that seems incoming data to Atmega), which sometimes makes sketch not starting. When removing this jumper, serial communication is disabled, and pin0 and pin1 work as a normal (floating) digital pin. Useful when more digital pins are needed, but only when serial communication is not necessary. An external pull-down/pull-up resistor is required.

JP4

2 pins jumper When in position 1-2, this jumper enables the auto-reset feature, useful when uploading a sketch to Arduino, resetting Atmega automatically. It makes unnecessary to press the reset button (S1) when uploading sketches. Be sure that computer COM Port speed is set to 19200bps otherwise auto-reset will not work properly. If removed, disables the auto-reset feature. Very useful to prevent undesired Atmega reset when using sketches that need serial communication. Auto-reset works with DTR pulse on serial pin4. Sometimes Arduino senses a DTR pulse when connecting X1 (serial connector) and some software sends a DTR pulse when it starts or when it closes, which makes Atmega reset when not desired.

S1

Tactile button This button resets Atmega, to restart uploaded sketch or to prepare Arduino to receive a sketch through serial connector (when auto-reset is not active).

LEDs

Indicative LEDs POWER led Turns on when Arduino is powered through DC1, +9v. pin or +5v. pin. RX led Blinks when receiving data from the computer/device through a serial connection. TX led Blinks when sending data to the computer/device through the serial connection. L led This led is connected to digital pin13 with a current limiter resistor (that doesn't affect pin13). Useful to test sketches. It is normal to blink when boot loading too.

POWER PINOUT

6 pin header

RST pin

Makes Atmega reset when connected to GND. Useful for Shield Boards, or to connect external reset.

NC pin

This pin is not connected to Arduino S3v3. Arduino Diecimila has a 3.3 volts pin in the same position.

+9v. pin

When Arduino DC1 is powered (with battery or DC adaptor), this pin is used as Vout, with the same voltage supplied on DC1 (see DC1), minus 0,7 volts. The total supplied current depends on external power source capacity When Arduino DC1 is not powered, +9v. the pin can be used as Vin, connecting it to an external regulated power source (+7 to +20 volts) and connecting 0v. pinto external power source GND. In this case,

+5v. pin

can be used as Vout, supplying +5 volts. +5v. pin When Arduino DC1 is powered (with battery or DC adaptor), +5v. pin supplies +5 volts as a Vout pin. The total supplied current depends on Voltage Regulator (7805 supplies up to 1A). This applies only to +5v. pin: Atmega in/out pins only supplies max. 40mA on each pin. When Arduino DC1 is not powered, this pin can be used as Vin, connecting it to a regulated +5v. and connecting 0v. pinto power source GND. In this case, +9v. the pin is inactive. 0v. pin (GND) Two 0v. pins between +5v. and +9v. / One

0v. pin

beside the AREF pin. When Arduino DC1 is powered, 0v. pin supplies 0 volts reference (GND) for +5v. pin and +9v. pin. When DC1 is not powered, and Arduino is powered through +5v. pin or +9v. pin, 0v. pin must be used as GND reference, connecting it to the external power source GND.

GND pin

sees 0v. pin (GND).

AREF pin

The AREF can be set to AVcc (default), internal 2.56 volts (Atmega8), internal 1.1 volts (Atmega168), or external AREF. In the case of AVcc or internal AREF, the AREF pin can be used to attach an external capacitor to decouple the signal, for better noise performance. In the case of external AREF, the AREF pin is used to attach the external reference voltage. Remember that it is necessary to change defuses (wiring file), and re-upload sketch, before connecting external voltage to AREF

3.2.7 Raspberry Pi

The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing and to learn how to program in languages like Scratch and Python. It's capable of doing everything you'd expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games.

What's more, the Raspberry Pi can interact with the outside world and has been used in a wide array of digital maker projects, from music machines and parent detectors to weather stations and tweeting birdhouses with infra-red cameras. We want to see the Raspberry Pi being used by kids all over the world to learn to program and understand how computers work.

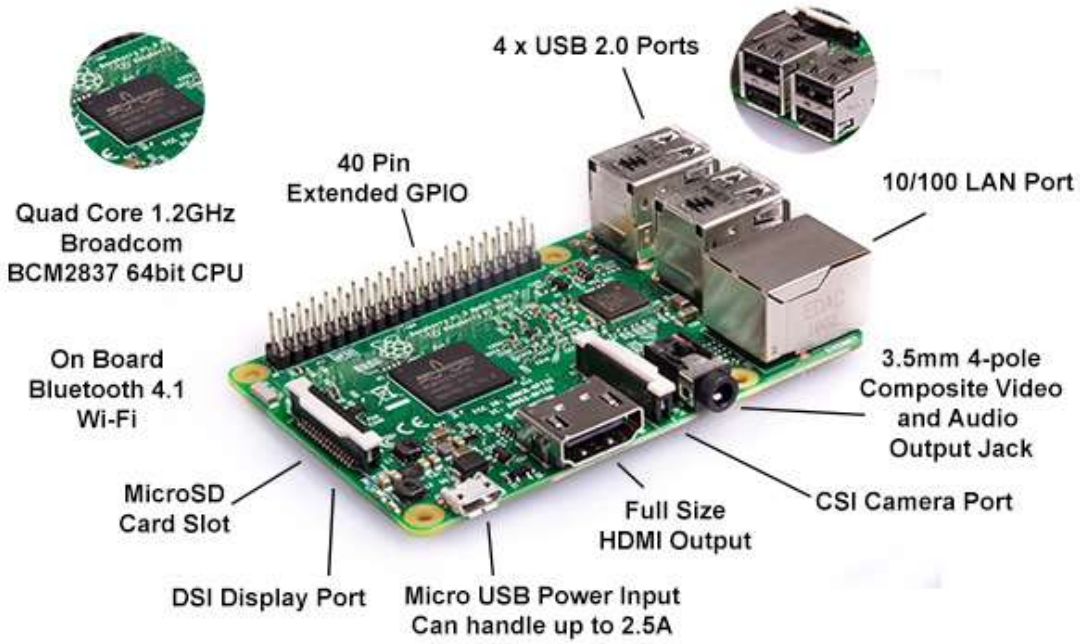


Figure 3.15 Raspberry PI

The Raspberry Pi was created with the goal of education in mind. This ultra-tiny computer was designed to be small and cheap so that schools could easily afford them to teach students about computers in the classroom. This is great for two reasons, the first is that it provides extremely cheap access to a computer, and second it is a great tool for learning more about computers (student or not)!

So how cheap are we talking exactly? Well, there are two versions of the Raspberry Pi, the model A is $25 and the model B is $35. This price point makes it pretty easily available to students, hobbyists, and even yours truly!

Let's talk about what the Raspberry Pi has on it. In model B, you get an HDMI out, RCA video out, 2 USB ports, an SD card slot, a headphone jack, and an Ethernet port. The board itself has half a gigabyte of RAM and an onboard ARM processor.

The model A has all of the same features of the model B minus one of the USB plugs, the Ethernet port, and half of the RAM.

No matter how you look at it though, it gives you quite a bit of equipment to work with for not being much bigger than a credit card!

Here's how it works: An SD card inserted into the slot on the board acts as the hard drive for the Raspberry Pi. It is powered by USB and the video output can be hooked up to a traditional RCA TV set, a more modern monitor, or even a TV using the HDMI port. This gives you all of the basic abilities of a normal computer. It also has an extremely low power consumption of about 3 watts.

To put this power consumption in perspective, you could run over 30 Raspberry Pis in place of a standard light bulb!

If you're feeling a bit overwhelmed with all of the talk of the features included in the Raspberry Pi, check out my previous episodes that go in more detail of what HDMI and RCA ports are exactly.

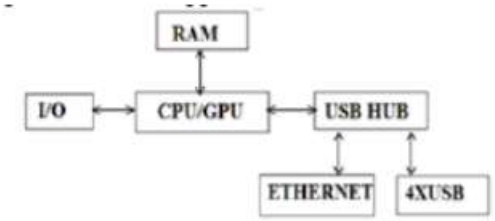


Figure 3.16 The hardware of raspberry pi

So far, we've covered what the Raspberry Pi is and what features it comes with. But what can you do with such a little computer?

The answer is a *ton*! Some of my favorite projects are creating wireless home speakers, center for my TV, and even a personal webserver! These projects are driven by the free operating system you can download for the Raspberry Pi called Raspbian. This operating system is a lightweight version of Linux that is optimized for this low powered device.

Each of these projects requires some work to set up, but they're a perfect way to get your feet wet with computer programming and operation.

The last thing I want to mention about the Raspberry Pi is the community. This is probably the greatest feature of the Raspberry Pi, but probably one of the least talked about. The community that uses the Raspberry Pi is the most helpful one that I have seen across the tech spectrum. One of the perks of the educational aspect of this device is that every project someone does with it is extremely well documented with step-by-step instructions and often includes pictures. The forums for the community span many counties and every level of expertise.

RASPBERRY PI POWER SUPPLY

Model B+ Power Supply: To make the B+ more reliable and actually reduce the current draw, the power supply is completely redesigned.

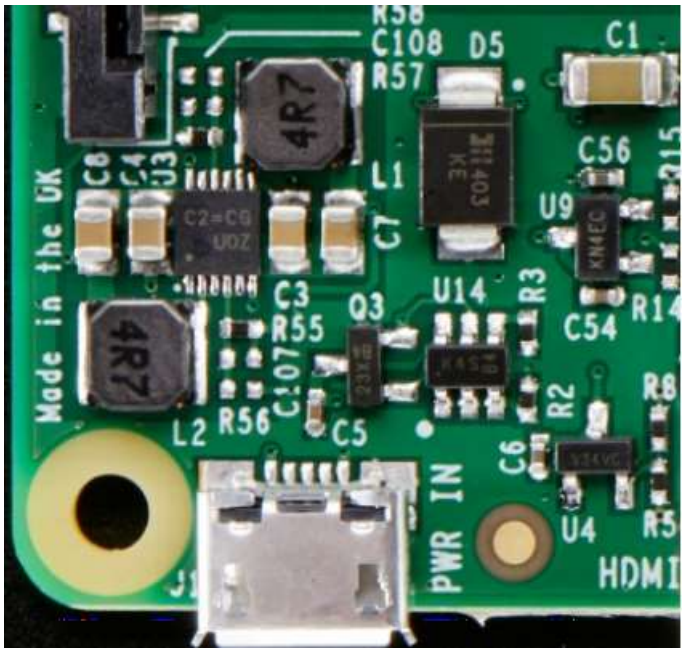


Figure 3.17 Power unit of raspberry pi

There's still the micro USB jack on the left, and the 1A fuse has been upgraded to a 2A fuse. There's also a DMG2305UX (http://adafru.it/dGU) P-Channel MOSFET. This acts as a polarity protection switch but is much lower 'drop-out' than a diode. It has only 52mW resistance so @ 2A its about 0.1V voltage drop. Most diodes would be at least 0.5V. Watch this great video about this technique here: To the right is a protection TVS diode (D5 part #SMBJ5) which protects from over-voltages. So not a lot has changed here (other than putting in a protection FET) There is a PNP-matched-pair action going on around the polarity FET, but its 3 AM and I are not 100% sure what it's for so I'll wait till I get some rest before doing any analysis. Let's look at the 3.3V & 1.8V supplies:

Instead of heat-spewing LDO (low dropout) regulators, we now have a dual buck converter. These are high-efficiency converters that can take 5V down to 3.3V or 1.8V without as much heat loss. They're more expensive than LDO's but not so! The input to the dual buck is 5V (VIN1 and VIN2) - there's no part number marked here for some reason but it has 12 pins, is a DFN-shaped part (I deal with DFN's all day so I can spot them), and has the marking code C2=CGU0G. with some searching around for a 12-DFN dual buck with 1.8V and 3.3V fixed outputs...

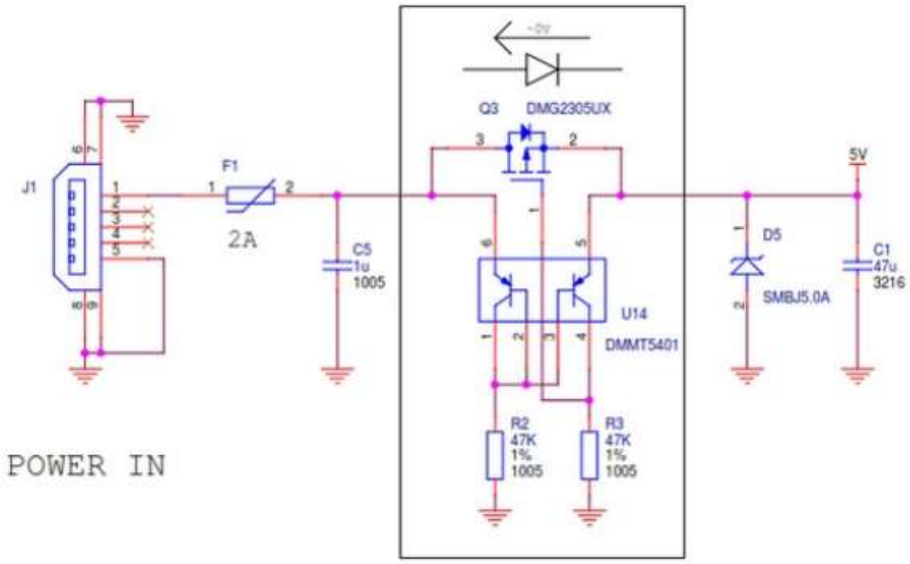


Figure 3.18 Schematic diagram of power input of raspberry pi

Features of Raspberry pi:

- CPU: Quad-core 64-bit ARM Cortex A53 clocked at 1.2 GHz
- GPU: 400MHz Video Core IV multimedia

- Memory: 1GB LPDDR2-900 SDRAM (i.e. 900MHz)
- USB ports: 4
- Video outputs: HDMI, composite video (PAL and NTSC) via 3.5 mm jack
- Network: 10/100Mbps Ethernet and 802.11n Wireless LAN
- Peripherals: 17 GPIO plus specific functions, and HAT ID bus
- Bluetooth: 4.1
- Power source: 5 V via Micro USB or GPIO header
- Size: 85.60mm × 56.5mm
- Weight: 45g (1.6 oz)

3.3 Power Supply Unit

Block Diagram

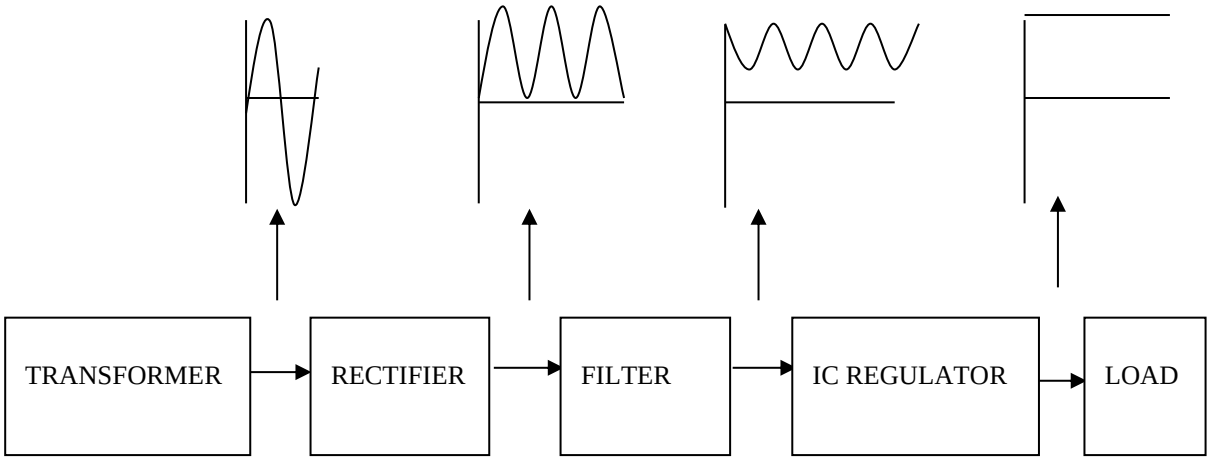


Figure 3.19 Block diagram of Power Supply

3.3.1 Transformer

The potential transformer will step down the power supply voltage (0-230V) to (0-6V) level. Then the secondary of the potential transformer will be connected to the precision rectifier, which is constructed with the help of op-amp. The advantages of using a precision rectifier are it will give peak voltage output as DC; the rest of the circuits will give only RMS output.

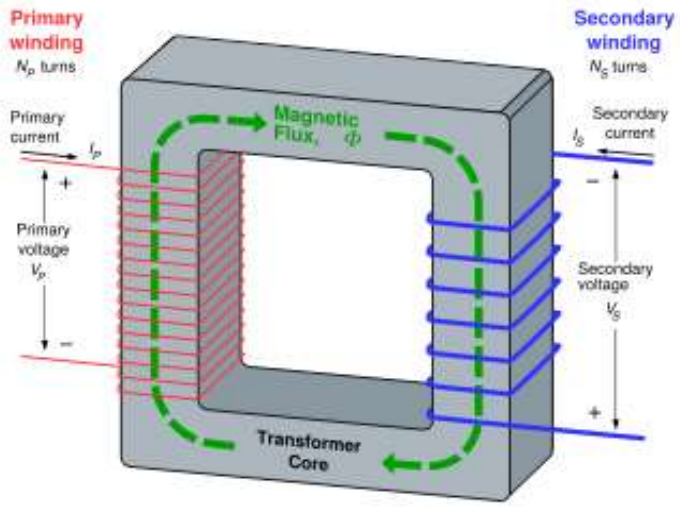


Figure 3.20 Transformer

3.3.2 Bridge Rectifier

When four diodes are connected as shown in the figure, the circuit is called a bridge rectifier. The input to the circuit is applied to the diagonally opposite corners of the network, and the output is taken from the remaining two corners.

Let us assume that the transformer is working properly and there is a positive potential, at point A and a negative potential at point B. the positive potential at point A will forward bias D3 and reverse bias D4.

The negative potential at point B will forward bias D1 and reverse D2. At this time D3 and D1 are forward biased and will allow current flow to pass through them; D4 and D2 are reverse biased and will block current flow.

The path for current flow is from point B through D1, up through RL, through D3, through the secondary of the transformer back to point B. this path is indicated by the solid arrows. Waveforms (1) and (2) can be observed across D1 and D3.

One-half cycle later the polarity across the secondary of the transformer reverse, forward biasing D2 and D4 and reverse biasing D1 and D3. Current flow will now be from point A through D4, up through RL, through D2, through the secondary of T1, and back to point A. This path is indicated by the broken arrows. Waveforms (3) and (4) can be observed across D2 and D4. The current flow through RL is always in the same direction.

In flowing through RL this current develops a voltage corresponding to that shown waveform (5). Since current flows through the load (RL) during both half cycles of the applied voltage, this bridge rectifier is a full-wave rectifier.

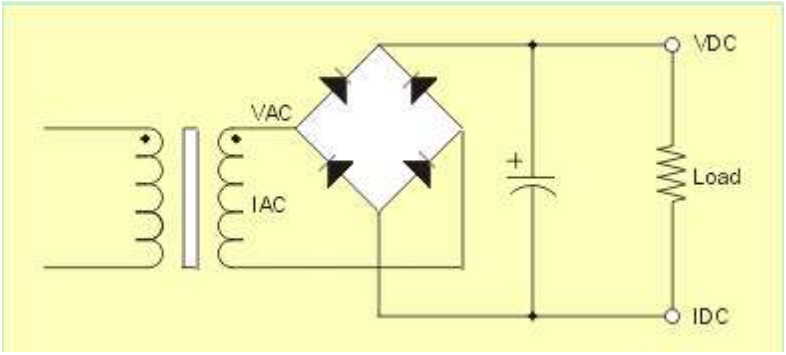


Figure 3.21 Bridge rectifier

One advantage of a bridge rectifier over a conventional full-wave rectifier is that with a given transformer the bridge rectifier produces a voltage output that is nearly twice that of the conventional full-wave circuit.

This may be shown by assigning values to some of the components shown in views A and B. Assume that the same transformer is used in both circuits. The peak voltage developed between points X and y are 1000 volts in both circuits. In the conventional full-wave circuit shown—in view A, the peak voltage from the center tap to either X or Y is 500 volts.

Since only one diode can conduct at any instant, the maximum voltage that can be rectified at any instant is 500 volts.

The maximum voltage that appears across the load resistor is nearly-but never exceeds-500 v0lts, as a result of the small voltage drop across the diode. In the bridge rectifier shown in view B, the maximum voltage that can be rectified is the full secondary voltage, which is 1000 volts. Therefore, the peak output voltage across the load resistor is nearly 1000 volts. With both circuits using the same transformer, the bridge rectifier circuit produces a higher output voltage than the conventional full-wave rectifier circuit.

3.3.3 IC Voltage regulators

Voltage regulators comprise a class of widely used ICs. Regulator IC units contain the circuitry for reference source, comparator amplifier, control device, and overload protection all in a single IC. IC units provide regulation of either a fixed positive voltage, a fixed negative voltage, or an adjustably set voltage. The regulators can be selected for operation with load currents from

hundreds of milliamperes to tens of amperes, corresponding to power ratings from milliwatts to tens of watts.

A fixed three-terminal voltage regulator has an unregulated dc input voltage, Vi, applied to one input terminal, a regulated dc output voltage, Vo, from a second terminal, with the third terminal connected to the ground.

The series 78 regulators provide fixed positive regulated voltages from 5 to 24 volts. Similarly, the series 79 regulators provide fixed negative regulated voltages from 5 to 24 volt

Figure 3.22 Voltage regulator

3.4 Software Used

3.4.1 Arduino IDE

The Arduino integrated development environment (IDE) is a cross-platform application written in Java and derives from the IDE for the Processing programming language and the Wiring projects.

It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and

Automatic indentation, and is capable of compiling and uploading programs to the board with a single click. A program or code written for Arduino is called a "sketch".

Arduino programs are written in C or C++. The Arduino IDE comes with a software library called "Wiring" from the original Wiring project, which makes many common input/output operations much easier. The users need only to define two functions to make an executable cyclic executive program. The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions, and a series of menus. It connects to the Arduino and Genuine hardware to upload programs and communicate with them.

Figure 3.23 Sketch of Arduino IDE

Writing sketches: written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension. no. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port. The toolbar
buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

Creates a new instance of the editor, with the bare minimum structure of a sketch already in place.

- Open

  Allows loading a sketch file browsing through the computer drives and folders.
- Open Recent

  It provides a shortlist of the most recent sketches, ready to be opened.
- Sketchbook

  Shows the current sketches within the sketchbook folder structure; clicking on any name opens the corresponding sketch in a new editor instance.
- Examples

  Any example provided by the Arduino Software (IDE) or library shows up in this menu item. All the examples are structured in a tree that allows easy access by topic or library.
- Close

  Closes the instance of the Arduino Software from which it is clicked.

- Save

  Saves the sketch with the current name. If the file hasn't been named before, a name will be provided in a "Save as." window.

- Save as

  Allows saving the current sketch with a different name.

- Page setup

  It shows the Page Setup window for printing.

EDIT

- Print

  Sends the current sketch to the printer according to the settings defined in Page Setup.

- Preferences

  Opens the Preferences window where some settings of the IDE may be customized, as the language of the IDE interface.

- Quit

  Closes all IDE windows. The same sketches open when Quit was chosen will be automatically reopened the next time you start the IDE.

- Undo/Redo

  Goes back of one or more steps you did while editing; when you go back, you may go forward with Redo.

- Cut

  Removes the selected text from the editor and places it into the clipboard.

- Copy

  Duplicates the selected text in the editor and places it into the clipboard.

- Copy from forum

  Copies the code of your sketch to the clipboard in a form suitable for posting to the forum, complete with syntax coloring.

- Copy as HTML

  Copies the code of your sketch to the clipboard as HTML, suitable for embedding on the web.

- Paste

  Put the contents of the clipboard at the cursor position, in the editor.

- Select All

  Selects and highlights the whole content of the editor.

- Comment/Uncomment

  Puts or removes the // comment marker at the beginning of each selected line.
- Increase/Decrease Indent

  Adds or subtracts a space at the beginning of each selected line, moving the text one space on the right or eliminating a space at the beginning.
- Find

  Opens the Find and Replace window where you can specify the text to search inside the current sketch according to several options.
- Find Next

  Highlights the next occurrence - if any - of the string specified as the search item in the Find window, relative to the cursor position.
- Find Previous

  Highlights the previous occurrence - if any - of the string specified as the search item in the Find window relative to the cursor position.

SKETCH

- Verify/Compile

  Checks your sketch for errors compiling it; it will report memory usage for code and variables in the console area.
- Upload

  Compiles and loads the binary file onto the configured board through the configured Port.
- UploadUsingProgrammer

  This will overwrite the bootloader on the board; you will need to use Tools > Burn Bootloader to restore it and be able to Upload to USB serial port again. However, it allows you to use the full capacity of the Flash memory for your sketch. Please note that this command will NOT burn the fuses. To do so a Tools -> Burn Bootloader command must be executed.
- ExportCompiledBinary

  Saves a .hex file that may be kept as an archive or sent to the board using other tools.
- ShowSketchFolder

  Opens the current sketch folder.
- IncludeLibrary

  Adds a library to your sketch by inserting #include statements at the start of your code. For

For more details, see the libraries below. Additionally, from this menu item, you can access the Library Manager and import new libraries from .zip files.

- AddFile...

  Adds a source file to the sketch (it will be copied from its current location). The new file appears in a new tab in the sketch window. Files can be removed from the sketch using the tab menu accessible clicking on the small triangle icon below the serial monitor one on the right side of the toolbar.

TOOLS

- Auto Format

  This formats your code nicely: i.e. indents it so that opening and closing curly braces line up, and that the statements inside curly braces are indented more.

- Archive Sketch

  Archives a copy of the current sketch in .zip format. The archive is placed in the same directory as the sketch.

- Fix Encoding & Reload

  Fixes possible discrepancies between the editor char map encoding and other operating systems char maps.

- serial monitor

  Opens the serial monitor window and initiates the exchange of data with any connected board on the currently selected Port. This usually resets the board, if the board supports Reset over the serial port opening.

- Board

  Select the board that you're using. See below for descriptions of the various boards.

- Port

  This menu contains all the serial devices (real or virtual) on your machine. It should automatically refresh every time you open the top-level tools menu.

- Programmer

  For selecting a hardware programmer when programming a board or chip and not using the onboard USB-serial connection. Normally you won't need this, but if you're burning a bootloader to a new microcontroller, you will use this.

- Bootloader

  The items in this menu allow you to burn a bootloader onto the microcontroller on an Arduino board. This is not required for the normal use of an Arduino or Genuine board but is useful

  if you purchase a new ATmega microcontroller (which normally comes without a bootloader). Ensure that you've selected the correct board from the Boards menu before burning the bootloader on the target board. This command also sets the right fuses.

HELP

- Here you find easy access to several documents that come with the Arduino Software (IDE). You have access to Getting Started, Reference, this guide to the IDE, and other documents locally, without an internet connection. The documents are a local copy of the online ones and may link back to our online website.
- Reference

  This is the only interactive function of the Help menu: it directly selects the relevant page in the local copy of the Reference for the function or command under the cursor.
- Sketchbook
- The Arduino Software (IDE) uses the concept of a sketchbook: a standard place to store your programs (or sketches). The sketches in your sketchbook can be opened from the File > Sketchbook menu or the Open button on the toolbar. The first time you run the Arduino software, it will automatically create a directory for your sketchbook. You can view or change the location of the sketchbook location from the Preferences dialog.
- Beginning with version 1.0, files are saved with a .ino file extension. Previous versions use the. pdf extension. You may still open. pde named files in version 1.0 and later, the software will automatically rename the extension to. ino.
- Tabs, Multiple Files, and Compilation
- It allows you to manage sketches with more than one file (each of which appears in its tab). These can be normal Arduino code files (no visible extension), C files (.c extension), C++ files (.cpp), or header files (.h).

Uploading

Before uploading your sketch, you need to select the correct items from the Tools > Board and Tools > Port menus. The boards are described below. On the Mac, the serial port is probably something like /dev/tty. usbmodem241 (for an Uno or Mega2560 or Leonardo) or /dev/tty.usbserial-1B1 (for a Duemilanove or earlier USB board), or /dev/tty.USA19QW1b1P1.1 (for a serial board connected with a Keyspan USB-to-Serial

adapter). On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, you look for USB serial device in the ports section of the Windows Device Manager. On Linux, it should

be /dev/ttyACMx , /dev/ttyUSBx or similar. Once you've selected the correct serial port and board, press the upload button in the toolbar or select the Upload item from the Sketch menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino Software (IDE) will display a message when the upload is complete, or show an error.

When you upload a sketch, you're using the Arduino bootloader, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The bootloader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The bootloader will blink the on-board (pin 13) LED when it starts (i.e. when the board resets).

Libraries

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from the Sketch > Import Library menu. This will insert one or more #include statements at the top of the sketch and compile the library with your sketch. Because libraries are uploaded to the board with your sketch, they increase the amount of space it takes up. If a sketch no longer needs a library, simply delete its #includestatements from the top of your code.

There is a list of libraries in the reference. Some libraries are included with the Arduino software. Others can be downloaded from a variety of sources or through the Library Manager. Starting with version 1.0.5 of the IDE, you can import a library from a zip file and use it in an open sketch. See these instructions for installing a third-party library.

To write your library, see this tutorial.

Third-Party Hardware

Support for third-party hardware can be added to the hardware directory of your sketchbook directory. Platforms installed there may include board definitions (which appear in the board menu), core libraries, bootloaders, and programmer definitions. To install, create the hardware directory, then unzip the third-party platform into its sub-directory. (Don't use "Arduino" as the sub-directory name or you'll override the built-in Arduino platform.) To uninstall, simply delete its directory.

For details on creating packages for third-party hardware, see the Arduino IDE 1.5 3rd party Hardware specification.

Serial Monitor

Displays serial data being sent from the Arduino or Genuine board (USB or serial board). To send data to the board, enter text, and click on the "send" button or press enter. Choose the baud rate from the drop-down that matches the rate passed to Serial.begin in your sketch. Note that on Windows, Mac, or Linux, the Arduino or Genuine board will reset (rerun your sketch execution to the beginning) when you connect with the serial monitor.

You can also talk to the board from Processing, Flash, MaxMSP, etc (see the interfacing page for details).

Preferences

Some preferences can be set in the preferences dialog (found under the Arduino menu on the Mac, or File on Windows and Linux). The rest can be found in the preferences file, whose location is shown in the preference dialog.

Language Support

Since version 1.0.1, the Arduino Software (IDE) has been translated into 30+ different languages. By default, the IDE loads in the language selected by your operating system. (Note: on Windows and possibly Linux, this is determined by the locale setting which controls currency and date formats, not by the language the operating system is displayed in.)

If you would like to change the language manually, start the Arduino Software (IDE) and open the Preferences window. Next to the Editor Language, there is a dropdown menu of currently supported languages. Select your preferred language from the menu, and restart the software to use the selected language. If your operating system language is not supported, the Arduino Software (IDE) will default to English.

You can return the software to its default setting of selecting its language based on your operating system by selecting System Default from the Editor Language drop-down. This setting will take effect when you restart the Arduino Software (IDE). Similarly, after changing your operating system's settings, you must restart the Arduino Software (IDE) to update it to the new default language.

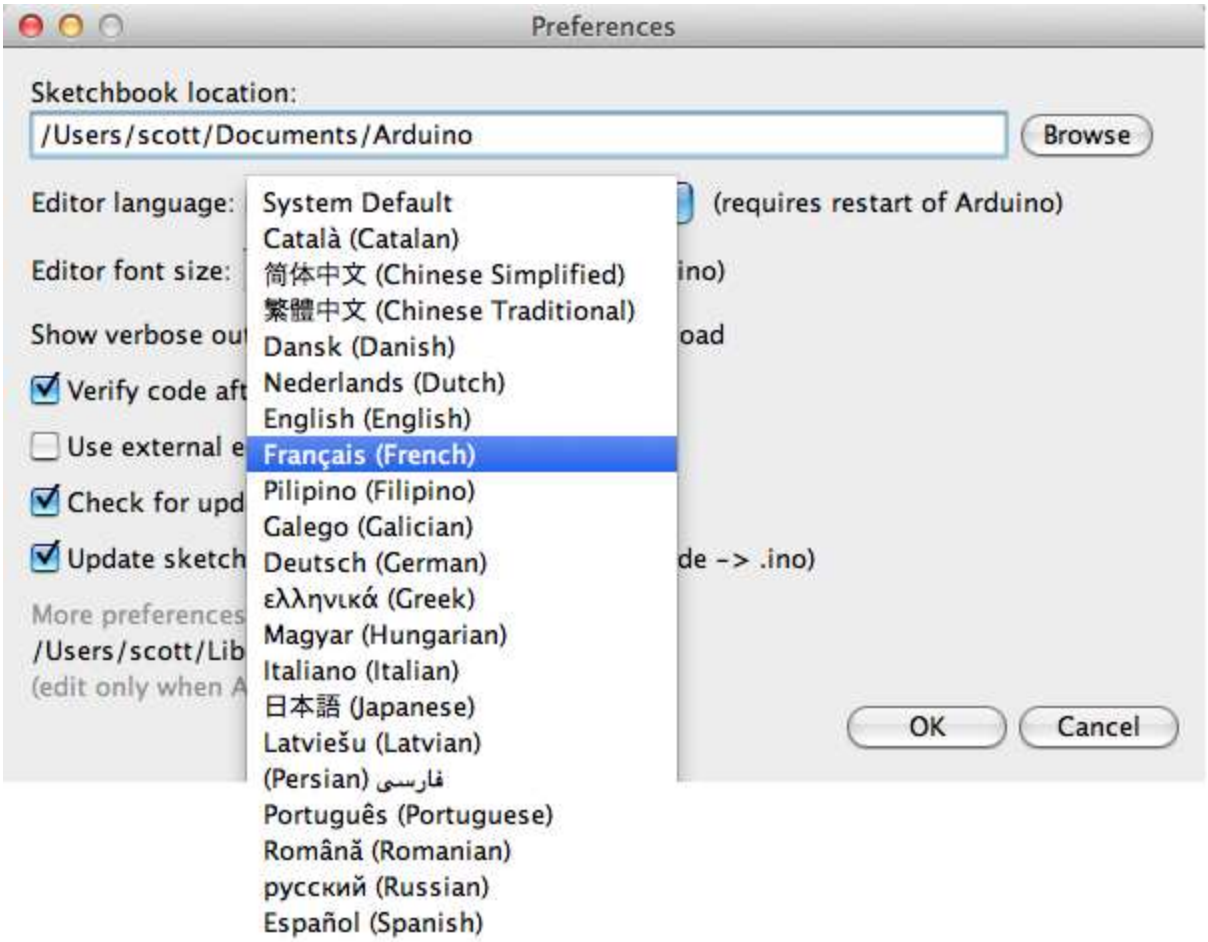


Figure 3.24 Language support in Arduino IDE

3.4.2 Python

History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms. The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or

other languages callable from C). Python is also suitable as an extension language for customizable applications.

3.4.3 ThingSpeak Cloud

ThingSpeak is an IoT analytics platform service that allows you to aggregate, visualize, and analyze live data streams in the cloud. ThingSpeak provides instant visualizations of data posted by your devices to ThingSpeak. With the ability to execute MATLAB® code in ThingSpeak you can perform online analysis and processing of the data as it comes in. ThingSpeak is often used for prototyping and proof of concept IoT systems that require analytics.

What is IoT?

Internet of Things (IoT) describes an emerging trend where a large number of embedded devices (things) are connected to the Internet. These connected devices communicate with people and other things and often provide sensor data to cloud storage and cloud computing resources where the data is processed and analyzed to gain important insights. Cheap cloud computing power and increased device connectivity are enabling this trend.

IoT solutions are built for many vertical applications such as environmental monitoring and control, health monitoring, vehicle fleet monitoring, industrial monitoring and control, and home automation.

At a high level, many IoT systems can be described using the diagram below:

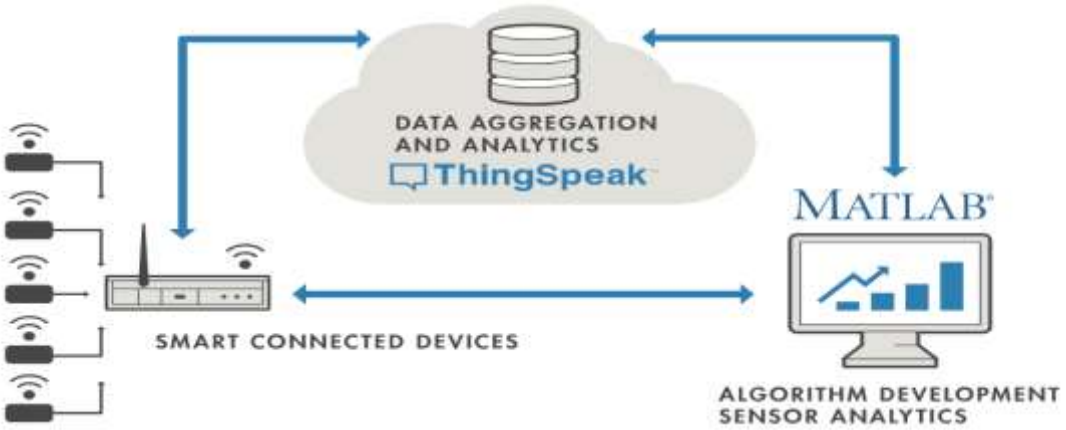


Figure 3.25 Architecture of ThingSpeak cloud

On the left, we have the smart devices (the “things” in IoT) that live at the edge of the network. These devices collect data and include things like wearable devices, wireless temperatures

sensors, heart rate monitors, and hydraulic pressure sensors, and machines on the factory floor.

In the middle, we have the cloud where data from many sources is aggregated and analyzed in real-time, often by an IoT analytics platform designed for this purpose.

The right side of the diagram depicts the algorithm development associated with the IoT application. Here an engineer or data scientist tries to gain insight into the collected data by performing historical analysis on the data. In this case, the data is pulled from the IoT platform into a desktop software environment to enable the engineer or scientist to prototype algorithms that may eventually execute in the cloud or on the smart device itself.

An IoT system includes all these elements. ThingSpeak fits in the cloud part of the diagram and provides a platform to quickly collect and analyze data from internet-connected sensors.

ThingSpeak Key Features

ThingSpeak allows you to aggregate, visualize, and analyze live data streams in the cloud. Some of the key capabilities of ThingSpeak include the ability to:

Easily configure devices to send data to ThingSpeak using popular IoT protocols.

Visualize your sensor data in real-time.

Aggregate data on-demand from third-party sources.

Use the power of MATLAB to make sense of your IoT data.

Run your IoT analytics automatically based on schedules or events.

Prototype and build IoT systems without setting up servers or developing web software.

Automatically act on your data and communicate using third-party.

Create a Channel

Sign In to ThingSpeak™ using your MathWorks® Account, or create a new MathWorks account.

Click Channels > MyChannels.

Figure 3.26 Creating a channel in ThingSpeak

On the Channels page, click New Channel.

Check the boxes next to Fields 1–3. Enter these channel setting values:

Name: Dew Point Measurement

Field 1: Temperature (F)

Field 2: Humidity

Field 3: Dew Point

Figure 3.27 Assigning fields in the channel that is to be created in ThingSpeak

Click Save Channel at the bottom of the settings.

You now see these tabs:

Private View: This tab displays information about your channel that only you can see.

Public View: If you choose to make your channel publicly available, use this tab to display selected fields and channel visualizations.

Channel Settings: This tab shows all the channel options you set at creation. You can edit, clear, or delete the channel from this tab.

Sharing: This tab shows channel sharing options. You can set a channel as private, shared with everyone (public), or shared with specific users.

API Keys: This tab displays your channel API keys. Use the keys to read from and write to your channel.

Data Import/Export: This tab enables you to import and export channel data.

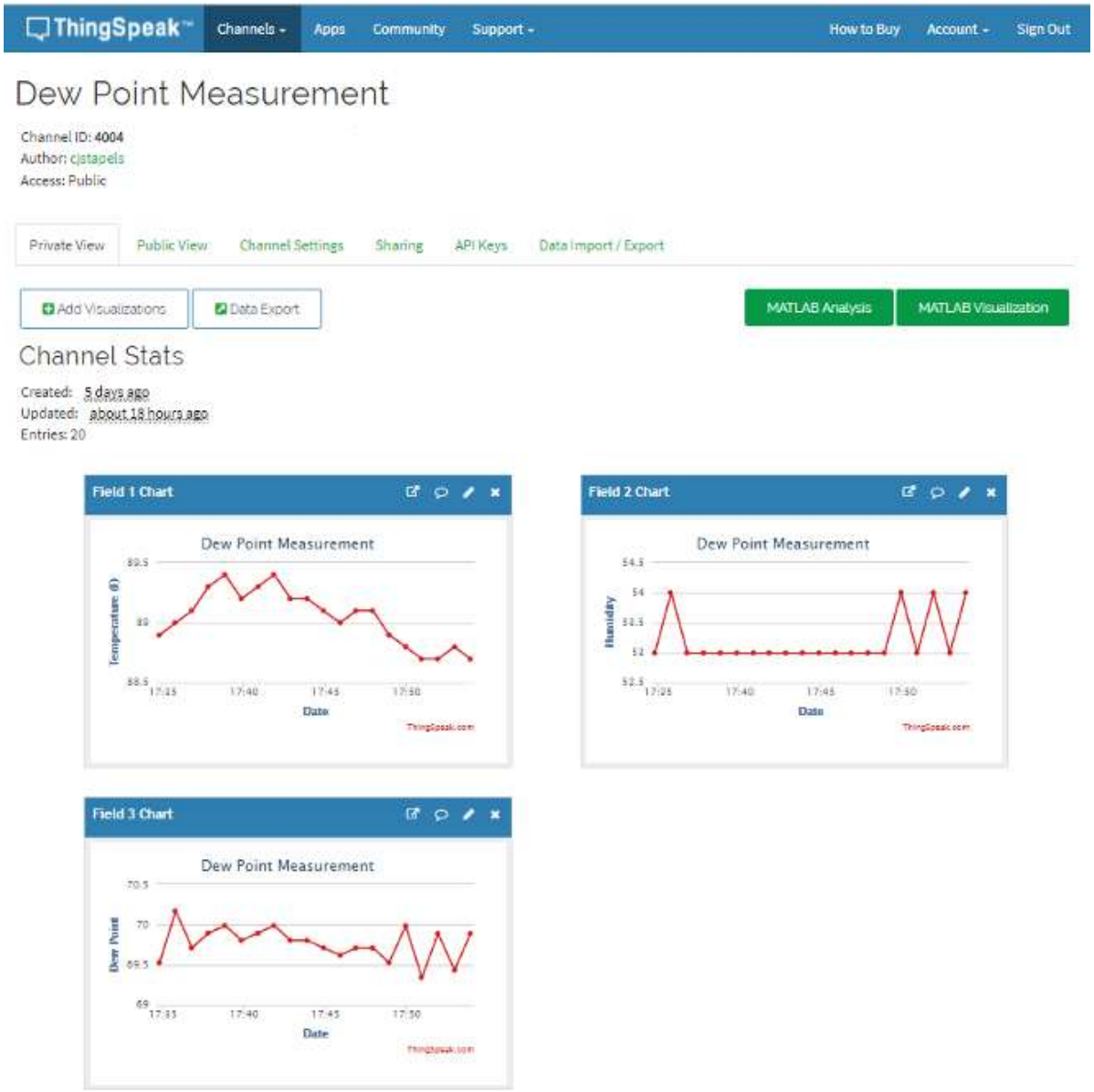


Figure 3.28 Displaying channel stats in ThingSpea

4. IMPLEMENTATION

4.1 Design and Functioning

In this project, we are going to do the coal miners security system using LORA and cloud why we are using LORA because we don't have internet system in the underground it is difficult to transmit the data collected by the sensors by using LORA we can transmit the data up to 10km but in our project, we are using the range of 80m due to cost issues and after receiving the data by RASPBERRY PI which will upload the data to the cloud and if the parameters reach the threshold limit we get SMS.

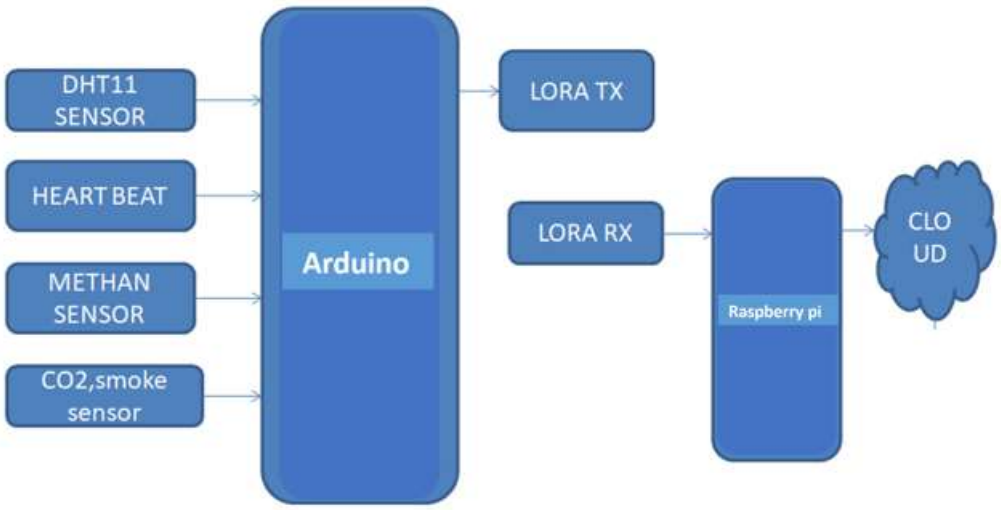


Figure 4.1 Block diagram of Intelligent Coal Mine Monitoring System based on the Lora-cloud

4.2 Working

The data sent by the Lora module at the transmitter is received by the Lora module employed in the receiver section once the data is received 30 available with raspberry Pi the same is adapted into the think speak loud whose API key is specified in the software code does at the monitoring section through the think speak webpage we can understand the environment inside the mind continuously as long as the mining process takes place, therefore, using this real-time wireless monitoring system early warnings can be provided to the miners present inside the mine and save

their lives before any casualties occur. As shown in the diagram above coal mine monitoring system built consists of two sections one is transmitting section and the other is receiving section transmitter section is fixed inside the mind where the mining person is going on receiver section will be on the monitoring unit on the surface transmitter section is built around few sentences which can provide data regarding various parameters like temperature humidity concentrations of harmful and toxic gases which are released during mining activity the methane Gas carbon monoxide coal gas propane smoke and heartbeat sensor is also employed to detect the pulse rate of workers the data collected censored by the senses is sent to the Arduino which then feed this data to LORA transmitter.

4.3 Schematic Diagram

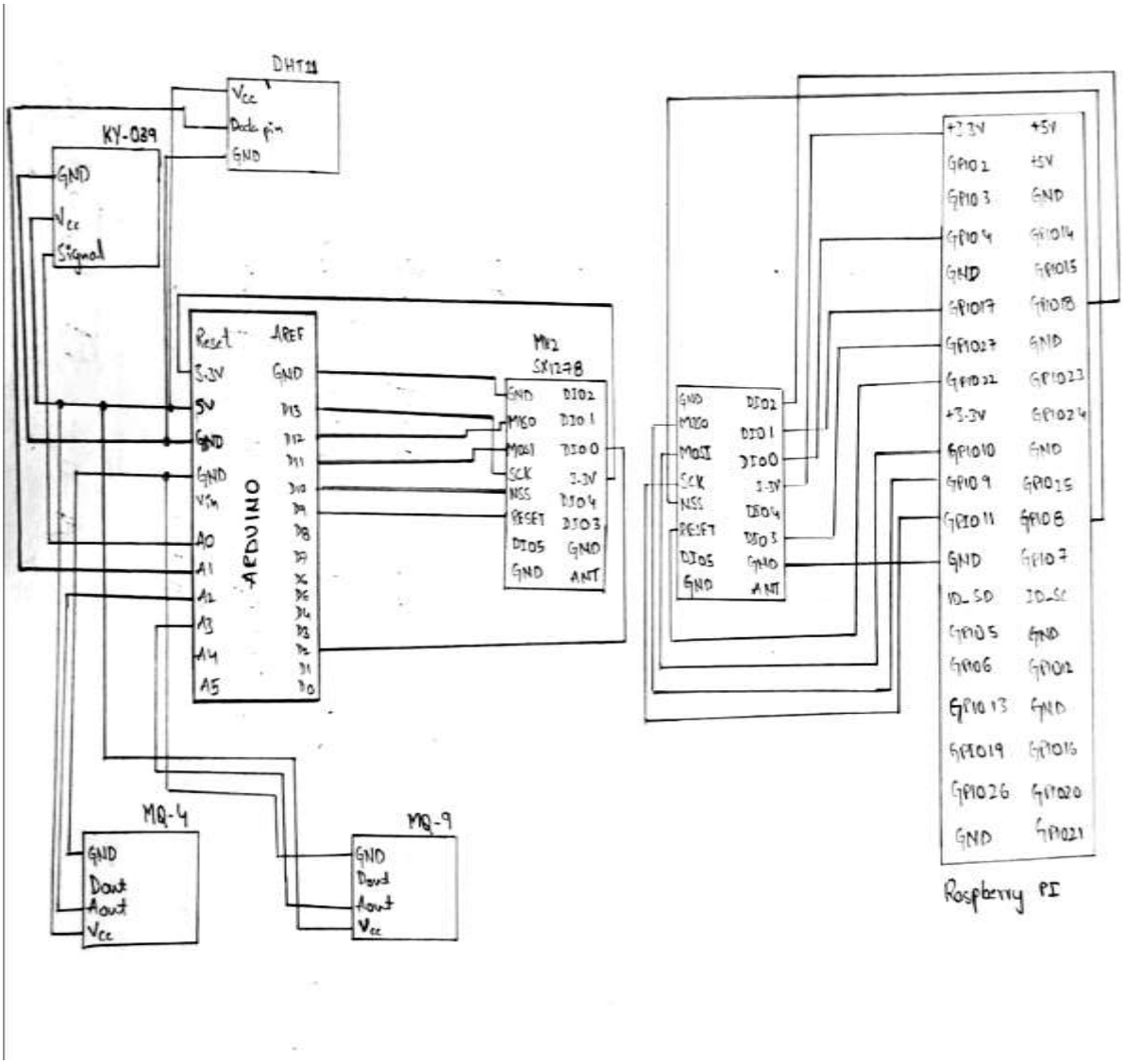
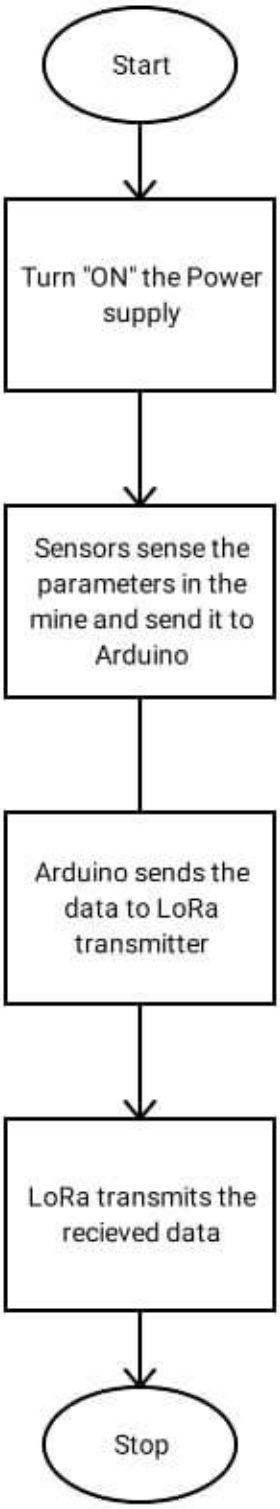


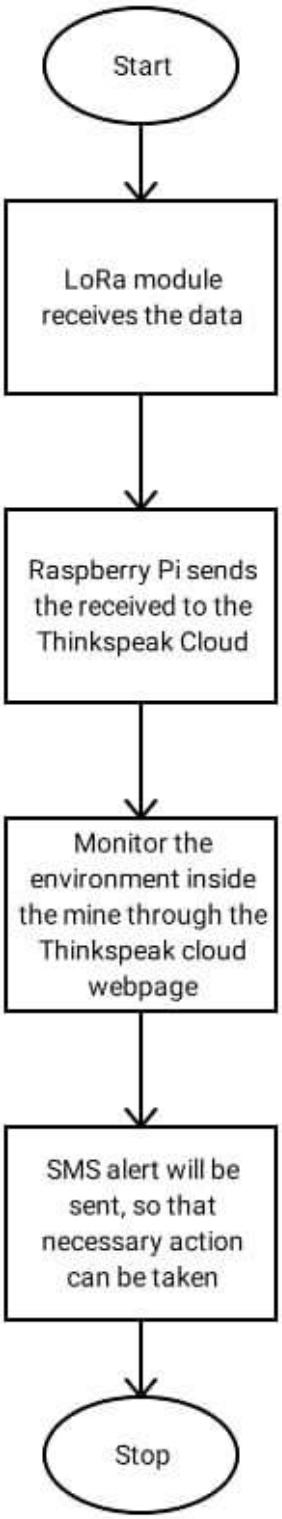
Figure 4.2 Schematic diagram of the proposed system

4.4 Flow Chart

Transmission end



Receiver end



5. RESULTS

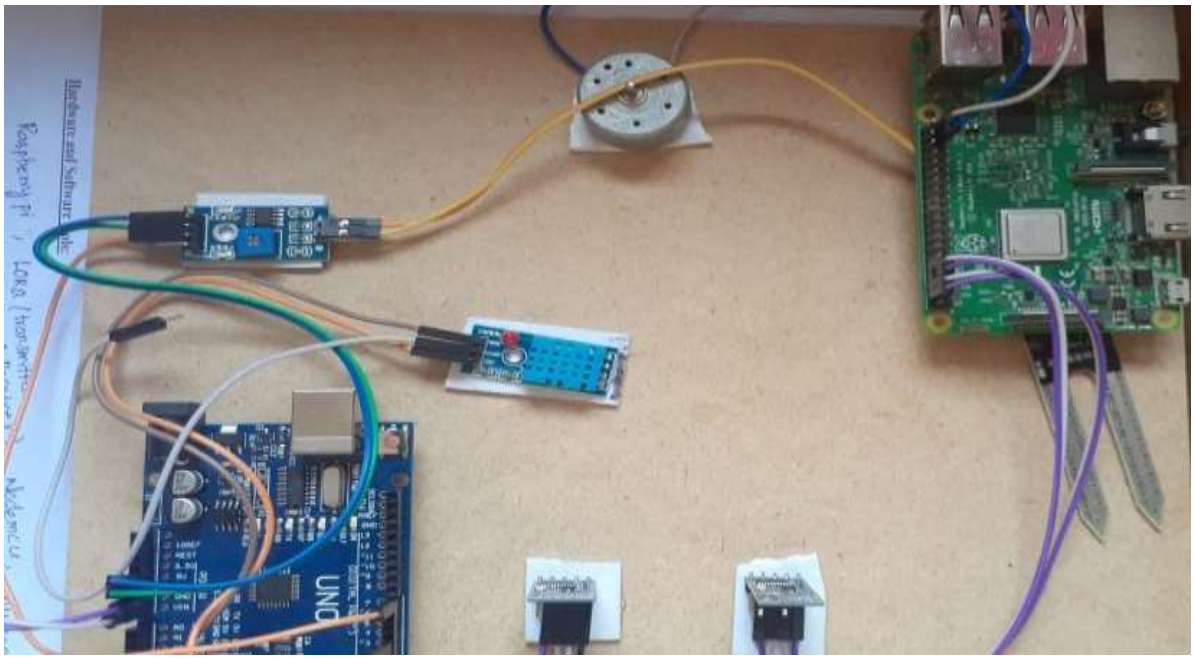


Figure 5.1 Picture illustrating the project work done

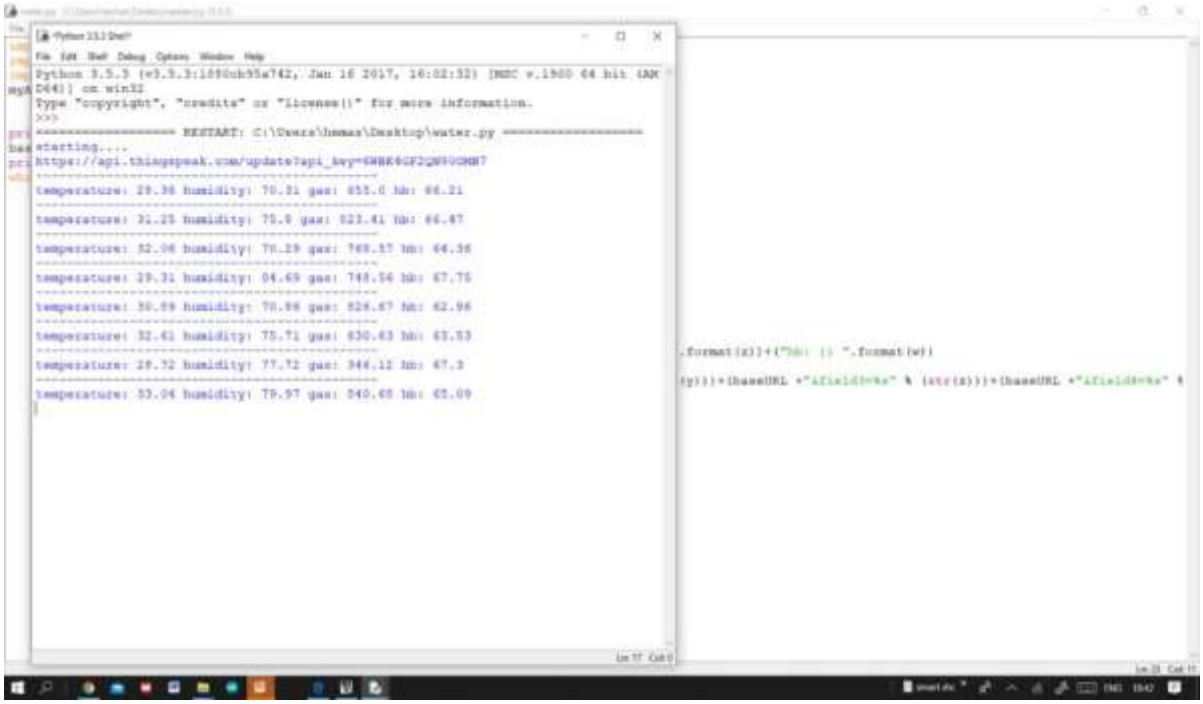


Figure 5.2 Magnitudes of the parameters received by the Lo-Ra

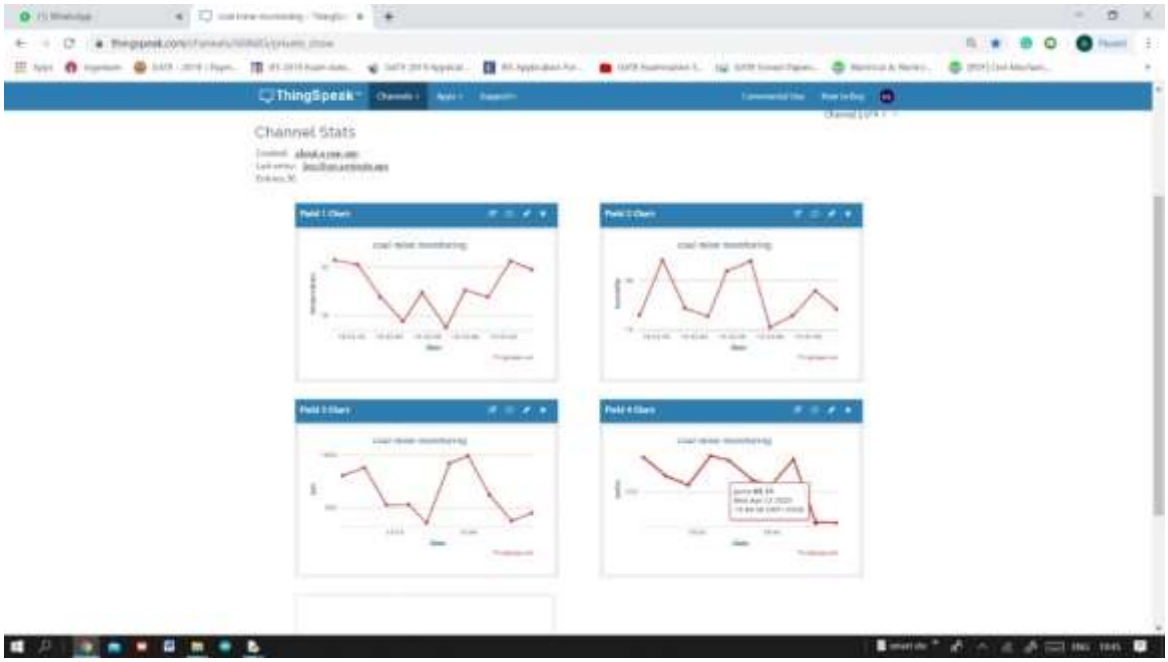


Figure 5.3 Graphical representation of results displayed on Thingspeak

6. CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

This paper builds a middleware for coal mine safety remote monitoring and control visualization. Focus on the design and implementation for underground wireless sensor network deployment, uniform devices access framework, distributed data distribution service, event-driven service execution engine, and RESTful-based open API interface. The main novelty of this study is to develop middleware for coal mine monitoring and control middleware which is easy to use and install for engineers. Since most of the application is Web-based, any personal computer and a web browser can connect the Internet and enter the Web page to use the application, and which can reduce the costs of coal mine safety monitoring and control automation. Therefore, it is expected to be the main contribution to coal mines for better and safer working environments. Several issues remain to be addressed further. First, as the expansion of existing coal mine safety monitoring and control systems, visualization technology can further improve the visibility of underground sensor objects, such as 3D technology, which provides significant support for decision making and real-time control in underground mines. Second, it is essential to optimize the real-time data distribution service and data congest scheduling strategy with different QoS constraints for a large-scale coal mine deployment. These works are currently in progress in our lab.

6.2 Future Scope

Therefore we can now say that LoRa Technology is the DNA of IoT, connecting sensors to the Cloud, and enabling real-time communication of data and analytics that can be utilized to enhance efficiency and productivity. LoRa Technology is the de facto choice for LPWAN connectivity for long-range, low power Internet of Things (IoT) solutions, enabling countless use cases in several key markets including smart cities, buildings, agriculture, metering, logistics and supply chain, and industrial control. LoRa technology solution is ready to run out-of-the-box and here are our few of our examples reg the scope of embedding Lora technology in the real world

1. The remote area monitoring system can be developed using the same concept.
2. An industrial process monitoring system can also be developed.
3. An applet can be designed to monitor the mine through a smartphone in hand.
4. Using additional sensors all possible safety issues could be monitored such as gases, dust, vibrations, fire, etc.

REFERENCES

[1]. Boddapati Venkata Sai Phani Gopal, Pakirabad Akash, P.S.G.Aruna Sri," Design Of IoT Based Coal Mine Safety System Using Nodemcu", International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-8 Issue-6, April 2019.

[2] Keerthana.E, KiruthikaM, Kalyani Suruthi.S, Lakshmi Priya.R, Dharini.G "A SMART SECURITY SYSTEM WITH MONITORING IN MINES", International Journal of Pure and Applied Mathematics Volume 119 No. 14 2018.

[3] M. Rajadurai, S. Aishwarya," Wireless Communication Based on Coal Mining Safety System with Arduino", International Conference on Latest Innovations in Applied Science, Engineering and Technology (ICLIASET 2017), March 2017.

[4] Shilpa Lande," Using Zigbee Integrated Alerting and Coal Mine Safety Monitoring System", International Journal of Science and Research (IJSR) ISSN (Online): 2319-7064 Index Copernicus Value 2013.

[5]. A. Bouguettaya, S. Nepal, W. Sherchan, X. Zhou, J. Wu, S.-P. Chen, D.-X. Liu, L. Li, H. B. Wang, and X.-M. Liu, "End-to-end service support for mashups," IEEE Trans. Serv. Comput., vol. 3, no. 3, pp. 250–263, Jul.–Sep. 2010

[6]. K. Page, "Blood on the coal: The effect of organizational size and differentiation on coal mine accidents," J. Safety Res., vol. 40, no. 2, pp. 85–95, 2009.

[7]. M. Li and Y.-H. Liu, "Underground coal mine monitoring with wireless sensor networks," ACM Trans. Sens. Netw., vol. 5, no. 2, pp. 1–29, 2009.

[8]. G.-Z. Chen, Z.-C. Zhu, G.-B. Zhou, C.-F. Shen, and Y.-J. Sun, "Strategy of deploying sensor nodes in the chain wireless sensor network for an underground mine," J. China Univ. Mining Technol., vol. 18, no. 4, pp. 561–566, 2008.

[9]. X.-G. Niu, X.-H. Huang, Z. Zhao, Y.-H. Zhang, C.-C. Huang, and L.Cui, "The design and evaluation of a wireless sensor network for mine safety monitoring," in Proc. IEEE GLOBECOM, 2007, pp. 1230–1236.

[10]. J. Wood, J. Dykes, A. Slingsby, and K. Clarke, "Interactive visual exploration of a large spatiotemporal dataset: Reflections on a geovisualization mashup," IEEE Trans. Vis. Comput. Graph., vol. 13, no. 6, pp. 1176–1183, Nov.–Dec. 2007.

[11]. . M. Ndoh and G. Y. Delisle, "Underground mines wireless propagation modeling," in Proc. 60th IEEE Veh. Technol. Conf., 2004, vol. 5, pp. 3584–3588

APPENDIX

Program

Arduino code:

```
#include<dht.h>
dht DHT;

#define DHT11_PIN 2
#define gas A0
#define fire A1
void setup() {
  Serial.begin(9600);
  pinMode(gas, INPUT);
  pinMode(pluse, INPUT);
}

void loop() {

 int chk = DHT.read11(DHT11_PIN);
 int humidity = DHT.humidity;
 int temp = DHT.temperature;
  Serial.println(" Humidity " );
  Serial.println(DHT.humidity, 1);
  Serial.println(" Temparature ");
  Serial.println(DHT.temperature, 1);

  int Gas = analogRead(gas);
  Serial.print("Gas: ");
  Serial.println(Gas);
  int Pluse = analogRead(fire);
  Serial.print("Pluse: ");
  Serial.println(Pluse);
  String c1 = "@";
  String c2 = "!";
  String c3 = "#";
  String c4 = "$";
  String sensorvalues = temp + c1 + humidity + c2 + Gas + c3 + pulse + c4;

  Serial.println(sensorvalues);
  delay(2000);
}
```

Python code:

```
import RPi.GPIO as GPIO

import serial

import time
```

```
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(26,GPIO.OUT)
import urllib.request as urllib2
port=serial.Serial("/dev/ttyAMA0", baudrate=9600,timeout=1.0)
myAPI = "Z0NG72JCVLG8EZGV"
baseURL = 'https://api.thingspeak.com/update?api_key=%s' % myAPI # THINGSPEAK
print(baseURL)
while True:
   rcv=port.read(9)
   print(rcv)
   x=(rcv[0:2])
   y=(rcv[2:4])
   z=(rcv[4:5])
   w=(rcv[5:7])
   o=(rcv[7:])
   water=int(o)
   print(water)



   print ("-------------------------------------------")
   data = ("temperature: {} ".format( x)) + ("humidity: {} ".format(y)) + ("heart beat: {}
".format(z))+("smoke: {} ".format(o))
   print(data)
   urllib2.urlopen(((baseURL +"&field1=%s" % (str(x)))+(baseURL +"&field2=%s" %
(str(y)))+(baseURL +"&field3=%s" % (str(z)))+(baseURL +"&field4=%s" % (str(o))))
```