JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY

ANANTAPURAMU

VEHICLE ACCIDENT DETECTION USING GPS AND GSM MODULE

A Project Report Submitted to the

Jawaharlal Nehru Technological University, Anantapuramu.

In Partial Fulfillment for the Award of the Degree of

BACHELOR OF TECHNOLOGY
IN
ELECTRONICS AND COMMUNICATION ENGINEERING

By

M. DHANUSH BABU (20JN1A0487)
P. MOHAMMAD ALI KHAN (20JN1A04B7)
M. SAI VENKATA KIRAN (20JN1A0488)
P.ADARSH (20JN1A04C5)

33

Under the Esteemed Guidance of

Mr. P. V. NARASIMHA SWAMY, M.E.,
Associate Professor.

DEPARTMENT OF
ELECTRONICS AND COMMUNICATION ENGINEERING

SREE VENKATESWARA COLLEGE OF ENGINEERING

NAAC 'A' Grade Accredited Institution, An ISO 9001:: 2015 Certified Institution
(Approved by AICTE, New Delhi and Affiliated to JNTU, Anantapuramu)
NORTHRAJUPALEM(VI), KODAVALURU(M), S.P.S.R NELLORE (DT) – 524 316

2020-2024

SREE VENKATESWARA COLLEGE OF ENGINEERING

NAAC 'A' Grade Accredited Institution, An ISO 9001:: 2015 Certified Institution
(Approved by AICTE, New Delhi and Affiliated to JNTU, Anantapuramu)
NORTHRAJUPALEM(VI), KODAVALURU(M), S.P.S.R NELLORE (DT) – 524 316

Department of
ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

Certified that the project entitled "VEHICLE ACCIDENT DETECTION USING GPS AND GSM MODULE" submitted by M. DHANUSH BABU (HT No. 20JN1A0487), P. MOHAMMAD ALI KHAN (HT No. 20JN1A04B7)M. SAI VENKATA KIRAN (HT No. 20JN1A0488), M. DHANUSH BABU (HT No. 20JN1A0487), P. ADARSH (HT No. 20JN1A04C5) in partial fulfilment for the award of degree of **BACHELOR OF TECHNOLOGY** in **ELECTRONICS AND COMMUNICATION ENGINEERING** by **Jawaharlal Nehru Technological University, Anantapuramu** during the academic year **2023-2024**. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said degree.

Signature of the Guide	Signature of Head of the Dept.
Mr. P.V. NARASIMHA SWAMY	**Dr. P. Giri Prasad**

Internal Examiner External Examiner

ACKNOWLEDGEMENT

With immense respect, We express my sincere gratitude to **Dr. P. Guna Sekhar, Chairman**, who took keen interest and encouraged in every effort throughout this course. We owe our gratitude to **Dr. V. Anil Kumar, Principal**, **SREE VENKATESWARA COLLEGE OF ENGINEERING, NELLORE** for giving us the opportunity to fulfil our aspirations and become Engineers.

We would like to extend ardent thanks to **Dr. P. Giri Prasad, Professor & Head of the Department**, **Electronics and Communication Engineering**, for endowing a practical environment in the institute.

We take this opportunity to express my sincere deep sense of gratitude to my guide **Mr. P.V. Narasimha Swamy,** M.E., **Department of Electronics and Communication Engineering**, for his significant suggestions and help in every aspect to accomplish the project.

We are thankful to the teaching and non-teaching staff members of my department for their cooperation in the completion of this project.

Finally, we thank one and all who directly or indirectly supported us in the completion of this project successfully.

M. Dhanush Babu (HT NO. 20JN1A0487)
P. Mohammad Alikhan (HT NO. 20JN1A04B7)
M. Sai Venkata Kiran (HT NO. 20JN1A0488)
P. Adrarsh (HT NO. 20JN1A04C5)

Dept. of Electronics and Communication Engineering
Sree Venkateswara College of Engineering,
NH5 Bypass Road, North Rajupalem, Nellore.

DECLARATION

We **M. DHANUSH BABU (HT No. 20JN1A0487), P. MOHAMMAD ALI KHAN (HT No. 20JN1A04B7), M. SAI VENKATA KIRAN (HT No. 20JN1A0488), P. ADARSH (HT No. 20JN1A04C5)** do hear by declare that the project entitled **"VEHICLE ACCIDENT DETECTION USING GPS AND GSM MODULE"** was carried out by our batch under the guidance of **Mr. P.V. Narasimha Swamy,** M. E, **ECE.** This Project work is submitted to Jawaharlal Nehru Technological University, Anantapuram in fulfillment of requirement for the award of **BACHELOR OF TECHNOLOGY** in **ELECTRONICS AND COMMUNICATION ENGINEERING** during the academic year 2023-2024.

Project Associates,

M. Dhanush Babu (HT NO. 20JN1A0487)
P. Mohammad Alikhan (HT NO. 20JN1A04B7)
M. Sai Venkata Kiran (HT NO. 20JN1A0488)
P. Adarsh (HT NO. 20JN1A04C5)

LIST OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

S.No.	Shortcut	Full Form
1.	PIN	PERSON INDENTIFICATION NUMBER
2.	RPS	REGULATED POWER SUPPLY
3.	IDE	INTEGRATED DEVELOPMENT ENVIRONMENT
4.	POR	POWER -ON -RESET
5.	LCD	LIQUID CRYSTAL DISPLAY
6.	Wi-Fi	WIRELESS FIDELITY
7.	ESP	ESPRESSIF MODULES
8.	HC	HOST CONTROLLER

ABSTRACT

Nowadays certain actions are taken to improve the level of cleanliness in the country. People are getting more active in doing all the things possible to clean their surroundings. When the garbage reaches the maximum level, a notification will be sent to the municipality office, and then the employees can take further action to empty the bin. This system will help to clean the city in a better way. Garbage bins remain uncollected for long periods putting the lives of marketers at risk if there is a Cholera outbreak, especially during the rainy season. To avoid such a situation, this project proposes the design and implementation of a GPS and IOT Based Garbage and Waste Collection Bin Overflow Management System using GPS and IOT technology to provide real-time information on the status of the garbage bins, i.e. when they are full so that appropriate action can be carried out. The proposed system has an IR sensor. Once a human comes to the nearby bin, it automatically detects and opens the bin door using a servo motor. At the top of the bin is an ultrasonic sensor, it measures the level of the bin and automatically sends live location using GPS to municipal servers using an IOT modem. All components are associated with the Arduino microcontroller. Arduino microcontroller used to process input and produce output by using ARDUINO IDE with Embedded C programming and operated through Regulated power supply which gives 5v of DC voltage to all hardware modules.

CHAPTER 1

INTRODUCTION

The major death rates in the world are due to the road accidents. India faces the highest death rate in the world. Reasons for the accident are speed driving, lacking sufficient sleep, drink and drive. Automatic accident detection helps to recognize the location of the accident and to find the location of the accident. For an ambulance vehicle, every second is important. If there is a delay in the arrival of ambulance, there will be a loss of life. Delay is caused mainly because of the traffic signals. Therefore, time factor is an important task. Radio Frequency module is used to control the traffic signals automatically. Therefore, the ambulance vehicle will reach the hospital in exact time to save the human. In addition, the main goals for the automatic accident detection techniques are to detect the accident and to send the message automatically to the emergency contacts along with the location. Emergency contacts include family members, friends, hospitals, police station etc. The incidents of accidental deaths have shown increasing trend during the year 2000-2015 with an increase of 50 percent in the year 2010 as compared to the year 2000. According to Planning Commission of India, the total annual economic loss is 2.5% of India’s GDP due to rising number of road fatalities. Another important reason can be improper medical help. Survey shows that each minute that an injured crash victim does not receive emergency medical care can cause into fatality. Most victims lose their lives due to such reasons. Therefore, this idea of saving lives by curing the problem comes into existence. Real-time position of the vehicles are informed by the system using the pre-install smart sensing accelerometer equipment. This data is recorded and all the information can be observed by remote location to provide the required services to the victims. Tracking of the vehicle can be done in all-weather condition. GPS and GSM technologies are used in this system to provide all the data to the remote server which are then processed and the extracted information is used to provide the services to the individual at the time of emergency. The main contributions of this paper are: (a) Vehicle registration and preparation, (b) Passengers ‘registration, (c) Monitoring accidents through a web interface located in the PSO headquarter. An embedded system is a combination of software and hardware to perform a dedicated task. Some of the main devices used in embedded products are Microprocessors and Microcontrollers.

Microprocessors are commonly referred to as general purpose processors as they simply accept the inputs, process it and give the output. In contrast, a microcontroller not only accepts the data as inputs but also manipulates it, interfaces the data with various devices, controls the data and thus finally gives the result.

The "**Arduino based Vehicle Tracking**" using Arduino microcontroller is an exclusive project which is used to designing a completely automated security access system for vehicles.

CHAPTER 2

LITERATURE SURVEY

Due to higher accident rates vehicle tracking is very important now days. This can be done easily using the GPS technology. Various other applications can also be used to do so [1]. These applications are also used in fleet management, anti-theft vehicle systems and accident recovery [2]. Vehicle Tracking: The vehicle tracking technology uses the GPS systems via many applications. These applications are very helpful as the track the vehicles and their partner web applications also monitor the vehicles continuously [1]. There are various ways to track a vehicle. Larger organizations use web services to tract large number of vehicles whereas small scale industries can use various mobile apps. To find exact location, distance and estimating time to reach particular destination an android app is developed [3]. Theoretical it is easy to say we can get the exact location of a vehicle, but practically sometimes it is next to impossible. Even though we have advanced technology it is very difficult to actually obtain the geographical coordinates correct all the time. Use Kalman filter can be done, to get an exact longitude and latitude positioLocation Identifier and immediate recovery of accident: As we already know there are numerous ways to track the location of an vehicle which has already met with an accident. We also know that there are ways where we can notify the authorized person about the mishap that has happened. When accidents happen, it becomes very difficult to send help to the victims as no notification the accident has the reached the hospitals, police or the family members of the victim. Thus resulting in a huge lose of life. To avoid such situations, we can send an automated SMS to the predefined numbers in the system. Bluetooth Technology is used as a medium to activate the GPS by the sensors. It is an intermediate between the sensors and the GPS. But now not only Bluetooth technology can be used but also MESA technology can be used to activate GPS and send the location coordinates to the predefined numbers [10].

CHAPTER 3

EXISTING SYSTEM

There are many solutions proposed for the concerned problem and each one have some advantage over others. Among the other GSM and GPS solutions, some proposed the solution of finding the accident condition using only accelerometer sensor which may be a problem as it may lead to false alarm for some of the cases. Our system uses more than one sensor to increase the accuracy of the system and also, we have provision to avoid the intimation in case of false alarm. The existing system also uses the external GPS and GSM modules hence increasing the cost of their project. Our system cut the unnecessary cost by using the already existing infrastructure like GPS; GSM built in the user’s mobile phone.

1.Accident Detection Mechanism: Implement sensors or algorithms to detect sudden changes in vehicle acceleration, deceleration, or orientation indicative of a collision or accident.

2.GPS Module Integration: Integrate GPS technology to accurately determine the vehicle's location coordinates at the time of the accident.

3.GSM Module Integration: Incorporate GSM communication capabilities to transmit the accident details, including location coordinates, to predefined emergency contacts or services.

4.Automatic Emergency Alert: Automatically trigger an emergency alert upon detection of an accident, ensuring immediate notification to emergency responders.

5.Real-Time Monitoring Interface: Develop a user interface accessible to emergency services or authorized personnel for real-time monitoring of accident alerts and location tracking.

6.Geofencing and Mapping: Implement geofencing capabilities to define specific geographical areas, enabling efficient dispatch of rescue teams. Integrate mapping services for visualization of accident locations.

7.Power Management: Optimize power consumption to ensure continuous operation of the system without draining the vehicle's battery excessively.

8.Robust Communication Protocols: Implement reliable communication protocols to ensure seamless data transmission even in remote or low connectivity areas.

9.User-Friendly Design: Design the system with simplicity and ease of use in mind, allowing for straightforward installation and operation by vehicle owners or fleet managers.

CHAPTER 4

PROPOSED SYSTEMS

This system can monitor the vehicle, prevent, as well as get auto location when accident occur and prevents accidents when any vehicle came near to this vehicle through ir sensor. In this Project it is proposed to design an embedded system which is used for tracking and positioning of any vehicle by using Global Positioning System (GPS) and Global system for mobile communication (GSM). In this project ARDUINO microcontroller is used for interfacing to various hardware peripherals. The current design is an embedded application, which will continuously monitor a moving Vehicle and report the status of the Vehicle on demand. For doing so an ARDUINO microcontroller is interfaced serially to a GSM Modem and GPS Receiver.

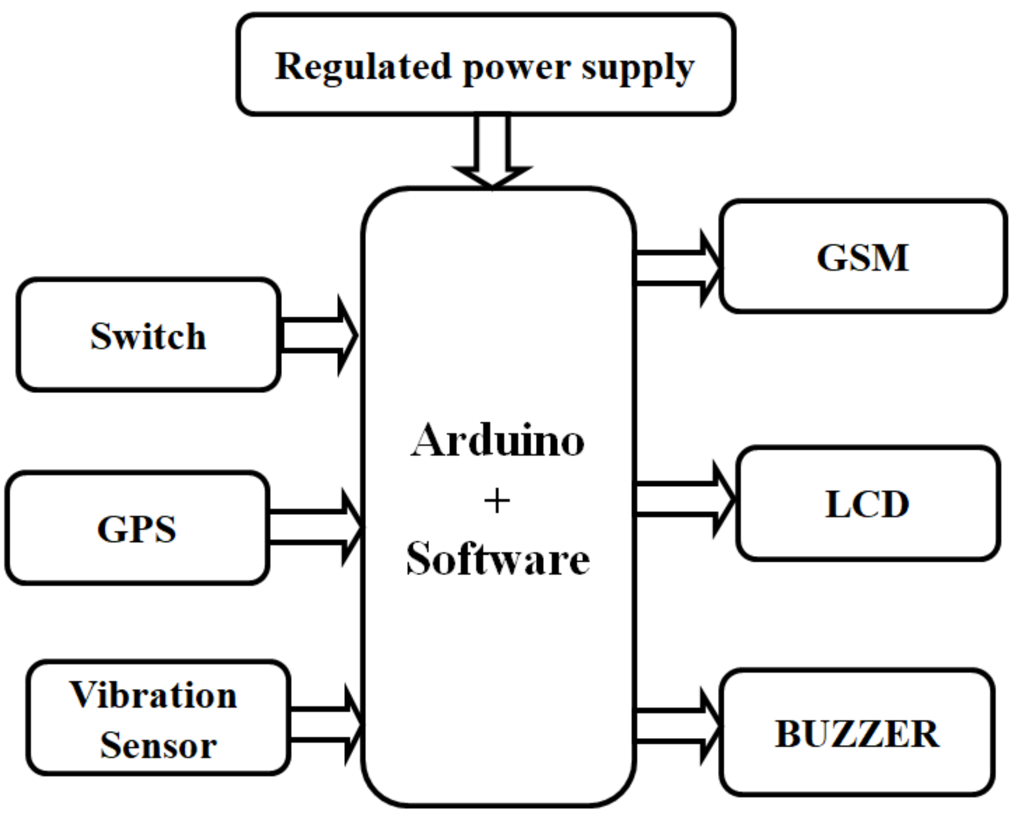


Fig 4.1: Block Diagram

A GSM modem is used to send the position (Latitude and Longitude) of the vehicle from a remote place. The GPS modem will continuously give the data i.e. the latitude and longitude indicating the position of the vehicle. The GPS modem gives many parameters as the output, but only the NMEA data coming out is read and displayed on to the LCD. The same data is sent to the mobile at the other end from where the position of the vehicle is demanded. An EEPROM is used to store the data received by GPS receiver. The hardware interfaces to microcontroller are LCD display, GSM modem and GPS Receiver. In order to interface GSM modem and GPS Receiver to the controller, a MUX is used. The design uses RS-232 protocol for serial communication between the modems and the microcontroller. A serial driver IC is used for converting TTL voltage levels to RS-232 voltage levels. Different types or sensors such as infrared sensors and fire detector are used for detecting different types of problem encountered in the vehicle such as theft, accident, fire warning etc. In any of these cases messages will be automatically send to the intended receiver. When a request by user is sent to the number at the modem, the system automatically sends a return reply to that particular mobile indicating the position of the vehicle in terms of latitude and longitude. A Program has been developed which is used to locate the exact position of the vehicle and also to navigated track of the moving vehicle on Google Map.

Whenever a vehicle is met with an accident then owner can send an SMS to the vehicle to know the location or position of the vehicle. The SMS sent would pass through the GSM service provider and then reach the vehicle, which is travelling, because the vehicle has a GSM device with a SIM card. This GSM modem will receive the SMS and send to the microcontroller in the vehicle. The microcontroller will receive this SMS and compare the password and the command. If the information matches the already programmed one, then it will perform the request required by the owner. It will then send the required location; latitude, longitude and time to the registered number of the owner and the results will be display on the screen of the owner’s mobile phone.

SCHEMATIC DIAGRAM

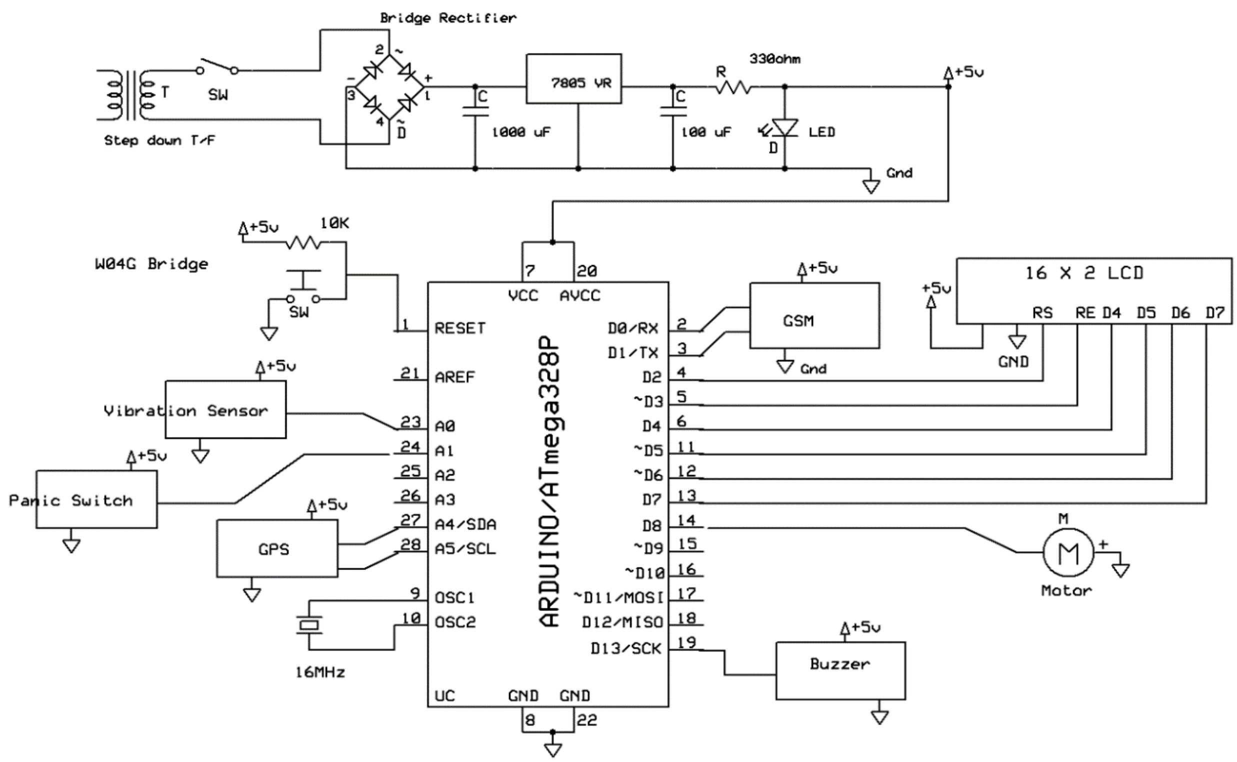


Fig 4.2: Schematic Diagram

The schematic diagram for a vehicle accident detection system using GPS and GSM modules involves several components connected to a microcontroller unit (MCU) like Arduino. Here's a breakdown of the key elements.

Circuit Connections of this **Vehicle Accident Alert System Project** is simple. Here Tx pin of **GPS module** is directly connected to digital pin number 10 of Arduino. By using Software Serial Library here, we have allowed serial communication on pin 10 and 11, and made them Rx and Tx respectively and left the Rx pin of GPS Module open. By default, Pin 0 and 1 of Arduino are used for serial communication but by using the Software Serial library, we can allow serial communication on other digital pins of the Arduino. 5V supply is used to power the GPS Module.

GSM module's Tx and Rx pins of are directly connected to pin D2 and D3 of Arduino. For GSM interfacing, here we have also used software serial library. GSM module is also powered by 12v supply. An **optional LCD's** data pins D4, D5, D6, and D7 are connected to pin number 6, 7, 8, and 9 of Arduino. Command pin RS and EN of LCD are connected with pin number 4 and 5 of Arduino and RW pin is directly connected with ground. A Potentiometer is also used for setting contrast or brightness of LCD.

CHAPTER 5

EMBEDDED SYSTEMS

5.1 Embedded Systems:

An embedded system is a computer system designed to perform one or a few dedicated functions often with real-time computing constraints. It is embedded as part of a complete device often including hardware and mechanical parts. By contrast, a general-purpose computer, such as a personal computer (PC), is designed to be flexible and to meet a wide range of end-user needs. Embedded systems control many devices in common use today.

Embedded systems are controlled by one or more main processing cores that are typically either microcontrollers or digital signal processors (DSP). The key characteristic, however, is being dedicated to handle a particular task, which may require very powerful processors. For example, air traffic control systems may usefully be viewed as embedded, even though they involve mainframe computers and dedicated regional and national networks between airports and radar sites. (Each radar probably includes one or more embedded systems of its own.)

Since the embedded system is dedicated to specific tasks, design engineers can optimize it to reduce the size and cost of the product and increase the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

Physically embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic lights, factory controllers, or the systems controlling nuclear power plants. Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large chassis or enclosure.

In general, "embedded system" is not a strictly definable term, as most systems have some element of extensibility or programmability. For example, handheld computers share some elements with embedded systems such as the operating systems and microprocessors which power them, but they allow different applications to be loaded and peripherals to be connected. Moreover, even systems which don't expose programmability as a primary feature generally need to support software updates. On a continuum from "general purpose" to "embedded", large application systems will have subcomponents at most points even if the system as a whole is "designed to perform one or a few dedicated functions", and is thus appropriate to call "embedded". A modern example of embedded system is shown in fig: 2.1.

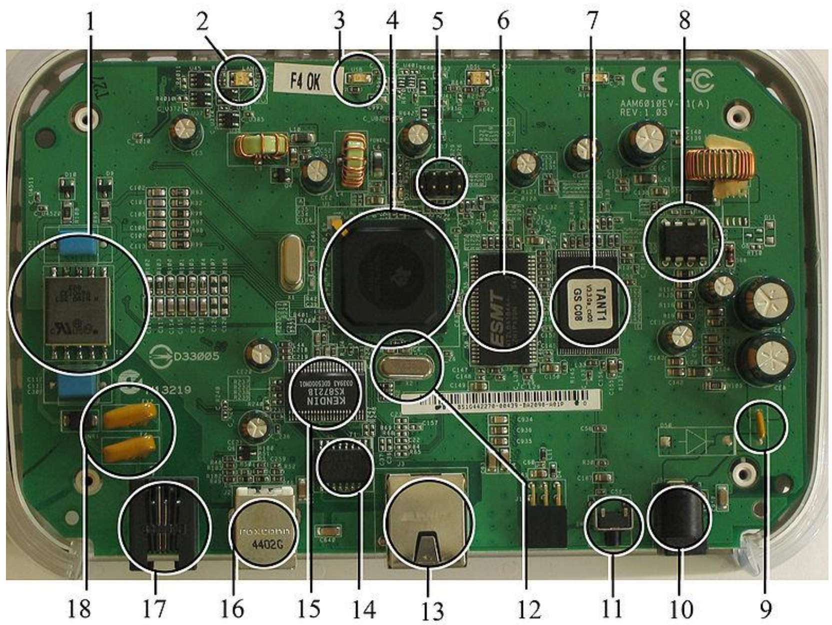


Fig 5.1:A modern example of embedded system

Labeled parts include microprocessor (4), RAM (6), flash memory (7). Embedded systems programming is not like normal PC programming. In many ways, programming for an embedded system is like programming PC 15 years ago. The hardware for the system is usually chosen to make the device as cheap as possible. Spending an extra dollar a unit in order to make things easier to program can cost millions. Hiring a programmer for an extra month is cheap in comparison. This means the programmer must make do with slow processors and low memory, while at the same time battling a need for efficiency not seen in most PC applications. Below is a list of issues specific to the embedded field.

5.1.1 Tools:

Embedded development makes up a small fraction of total programming. There's also a large number of embedded architectures, unlike the PC world where 1 instruction set rules, and the Unix world where there's only 3 or 4 major ones. This means that the tools are more expensive. It also means that they're lowering featured, and less developed. On a major embedded project, at some point you will almost always find a compiler bug of some sort.

Debugging tools are another issue. Since you can't always run general programs on your embedded processor, you can't always run a debugger on it. This makes fixing your program difficult. Special hardware such as JTAG ports can overcome this issue in part. However, if you stop on a breakpoint when your system is controlling real world hardware (such as a motor), permanent equipment damage can occur. As a result, people doing embedded programming quickly become masters at using serial IO channels and error message style debugging.

5.1.2 Resources:

To

Memory is also an issue. For the same cost savings reasons, embedded systems usually have the least memory they can get away with. That means their algorithms must be memory efficient (unlike in PC programs, you will frequently sacrifice processor time for memory, rather than the reverse). It also means you can't afford to leak memory. Embedded applications generally use deterministic memory techniques and avoid the default "new" and "malloc" functions, so that leaks can be found and eliminated more easily. Other resources programmers expect may not even exist. For example, most embedded processors do not have hardware FPUs (Floating-Point Processing Unit). These resources either need to be emulated in software, or avoided altogether.

5.1.3 Real Time Issues:

Embedded systems frequently control hardware, and must be able to respond to them in real time. Failure to do so could cause inaccuracy in measurements, or even damage hardware such as motors. This is made even more difficult by the lack of resources available. Almost all embedded systems need to be able to prioritize some tasks over others, and to be able to put off/skip low priority tasks such as UI in favor of high priority tasks like hardware control.

5.2 Need For Embedded Systems:

The uses of embedded systems are virtually limitless, because every day new products are introduced to the market that utilizes embedded computers in novel ways. In recent years, hardware such as microprocessors, microcontrollers, and FPGA chips have become much cheaper. So when implementing a new form of control, it's wiser to just buy the generic chip and write your own custom software for it. Producing a custom-made chip to handle a particular task or set of tasks costs far more time and money. Many embedded computers even come with extensive libraries, so that "writing your own software" becomes a very trivial task indeed. From an implementation viewpoint, there is a major difference between a computer and an embedded system. Embedded systems are often required to provide Real-Time response. The main elements that make embedded systems unique are its reliability and ease in debugging.

5.2.1 Debugging:

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

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

- A complete emulator provides a simulation of all aspects of the hardware, allowing all of it to be controlled and modified and allowing debugging on a normal PC.
- Unless restricted to external debugging, the programmer can typically load and run software through the tools, view the code running in the processor, and start or stop its operation. The view of the code may be as assembly code or source-code.

Because an embedded system is often composed of a wide variety of elements, the debugging strategy may vary. For instance, debugging a software(and microprocessor) centric embedded system is different from debugging an embedded system where most of the processing is performed by peripherals (DSP, FPGA, co-processor). An increasing number of embedded systems today use more than one single processor core. A common problem with multi-core development is the proper synchronization of software execution. In such a case, the embedded system design may wish to check the data traffic on the busses between the processor cores, which requires very low-level debugging, at signal/bus level, with a logic analyzer, for instance.

5.2.2 Reliability:

Embedded systems often reside in machines that are expected to run continuously for years without errors and in some cases recover by them if an error occurs. Therefore the software is usually developed and tested more carefully than that for personal computers, and unreliable mechanical moving parts such as disk drives, switches or buttons are avoided.

Specific reliability issues may include:

- The system cannot safely be shut down for repair, or it is too inaccessible to repair. Examples include space systems, undersea cables, navigational beacons, bore-hole systems, and automobiles.
- The system must be kept running for safety reasons. "Limp modes" are less tolerable. Often backups are selected by an operator. Examples include aircraft navigation, reactor control systems, safety-critical chemical factory controls, train signals, engines on single-engine aircraft.
- The system will lose large amounts of money when shut down: Telephone switches, factory controls, bridge and elevator controls, funds transfer and market making, automated sales and service.

A variety of techniques are used, sometimes in combination, to recover from errors—both software bugs such as memory leaks, and also soft errors in the hardware:

- Watchdog Immunity Aware Programming

5.2.3 Real-time embedded systems:

Embedded systems which are used to perform a specific task or operation in a specific time period those systems are called as real-time embedded systems. There are two types of real-time embedded systems.

- Hard Real-time embedded systems:

These embedded systems follow an absolute dead line time period i.e.., if the tasking is not done in a particular time period then there is a cause of damage to the entire equipment.

Eg: consider a system in which we have to open a valve within 30 milliseconds. If this valve is not opened in 30 ms this may cause damage to the entire equipment. So in such cases we use embedded systems for doing automatic operations.

- Soft Real Time embedded systems:

These embedded systems follow a relative dead line time period i.e.., if the task is not done in a particular time that will not cause damage to the equipment.

Eg: Consider a TV remote control system , if the remote control takes a few milliseconds delay it will not cause damage either to the TV or to the remote control. These systems which will not cause damage when they are not operated at considerable time period those systems comes under soft real-time embedded systems.

CHAPTER 6

HARDWARE DESCRIPTION

6.1 Micro controller:

Fig 6.1: Microcontrollers

6.1.1 Introduction to Microcontrollers:

Circumstances that we find ourselves in today in the field of microcontrollers had their beginnings in the development of technology of integrated circuits. This development has made it possible to store hundreds of thousands of transistors into one chip.

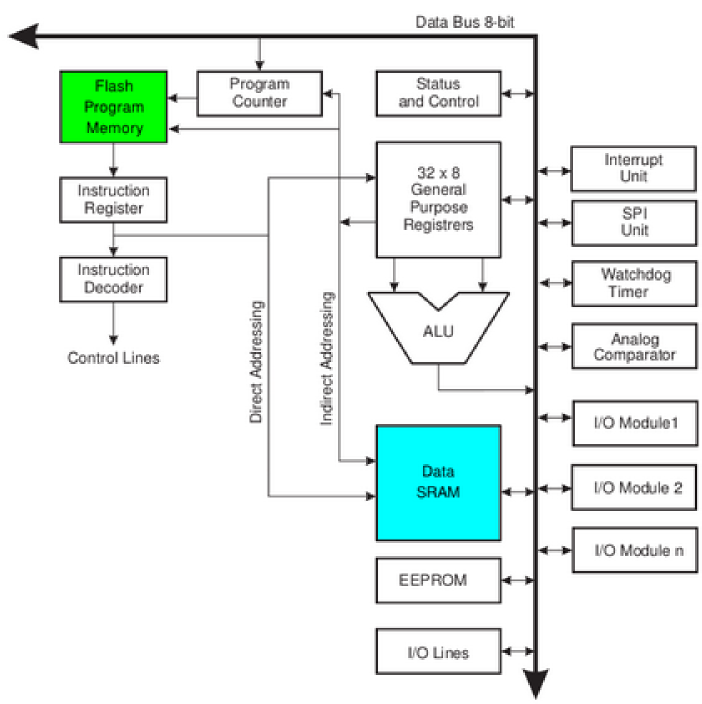


Fig 6.2: Architecture

That was a prerequisite for production of microprocessors, and the first computers were made by adding external peripherals such as memory, input-output lines, timers and other. Further increasing of the volume of the package resulted in creation of integrated circuits. These integrated circuits contained both processor and peripherals. That is how the first chip containing a microcomputer, or what would later be known as a microcontroller came about.

Microprocessors and microcontrollers are widely used in embedded systems products. Microcontroller is a programmable device. A microcontroller has a CPU in addition to a fixed amount of RAM, ROM, I/O ports and a timer embedded all on a single chip. The fixed amount of on-chip ROM, RAM and number of I/O ports in microcontrollers makes them ideal for many applications in which cost and space are critical.

6.2 AVR-ARDUINO MICROCONTROLLER:

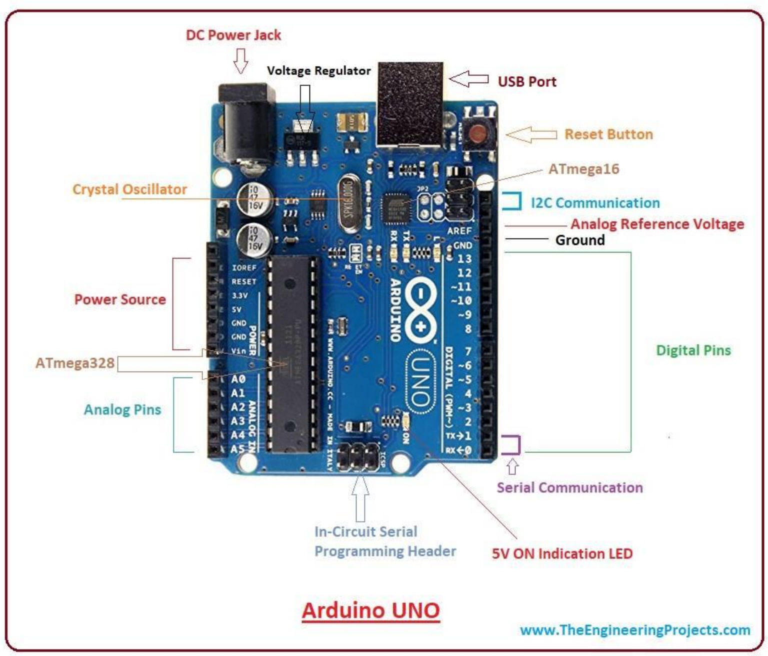


Figure 6.3: ARDUINO Development Board

The AVR is a modified Harvard architecture 8-bit RISC single chip microcontroller which was developed by Atmel in 1996. The AVR was one of the first microcontroller families to use on-chip flash memory for program storage, as opposed to One-Time Programmable ROM, EPROM, or EEPROM used by other microcontrollers at the time.

6.3 Crystal Oscillator:

XTAL1 and XTAL2 are input and output, respectively, of an inverting amplifier which can be configured for use as an On-chip Oscillator, Either a quartz

Crystal or a ceramic resonator may be used. The CKOPT Fuse selects between two different Oscillator amplifier modes. When CKOPT is programmed, the Oscillator output will oscillate a full rail-to-rail swing on the output. This mode is suitable when operating in a very noisy environment or when the output from XTAL2 drives a second clock buffer. This mode has a wide frequency range. When CKOPT is unprogrammed, the Oscillator has a smaller output swing. This reduces power consumption considerably.

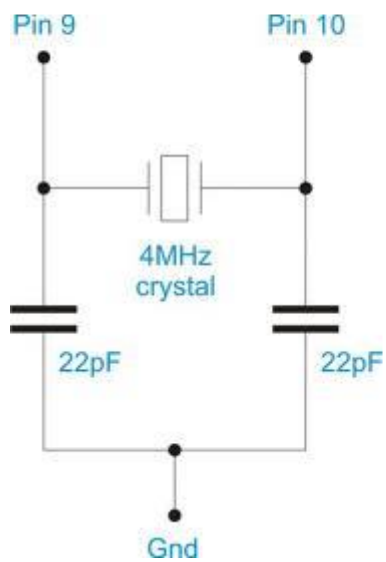


Figure 6.4: crystal oscillator

This mode has a limited frequency range and it cannot be used to drive other clock buffers. For resonators, the maximum frequency is 8 MHz with CKOPT unprogrammed and 16 MHz with CKOPT programmed. C1 and C2 should always be equal for both crystals and resonators. The optimal value of the capacitors depends on the crystal or resonator in use, the amount of stray capacitance, and the electromagnetic noise of the environment. For ceramic resonators, the capacitor values given by the manufacturer should be used. The Oscillator can operate in three different modes, each optimized for a specific frequency range. The operating mode is selected by the fuses CKSEL3..1

6.4 Architecture:

Memory: It has **8 Kb** of Flash program memory (10,000 Write/Erase cycles durability), **512 Bytes** of EEPROM (100,000 Write/Erase Cycles). **1Kbyte** Internal SRAM

I/O Ports: 23 I/ line can be obtained from three ports; namely Port B, Port C and Port D.

Interrupts: Two External Interrupt source, located at port D. 19 different interrupt vectors supporting 19 events generated by internal peripherals.

Timer/Counter: Three Internal Timers are available, two 8 bit, one 16 bit, offering various operating modes and supporting internal or external clocking.

SPI (Serial Peripheral interface): ATmega8 holds three communication devices integrated. One of them is Serial Peripheral Interface. Four pins are assigned to Atmega8 to implement this scheme of communication.

USART: One of the most powerful communication solutions is USART and ATmega8 supports both synchronous and asynchronous data transfer schemes. It has three pins assigned for that. In many projects, this module is extensively used for PC-Micro controller communication.

TWI (Two Wire Interface): Another communication device that is present in ATmega8 is Two Wire Interface. It allows designers to set up a commutation between two devices using just two wires along with a common ground connection, As the TWI output is made by means of open collector outputs, thus external pull up resistors are required to make the circuit.

Analog Comparator: A comparator module is integrated in the IC that provides comparison facility between two voltages connected to the two inputs of the Analog comparator via External pins attached to the micro controller.

Analog to Digital Converter: Inbuilt analog to digital converter can convert an analog input signal into digital data of **10bit** resolution. For most of the low end application, this much resolution is enough.

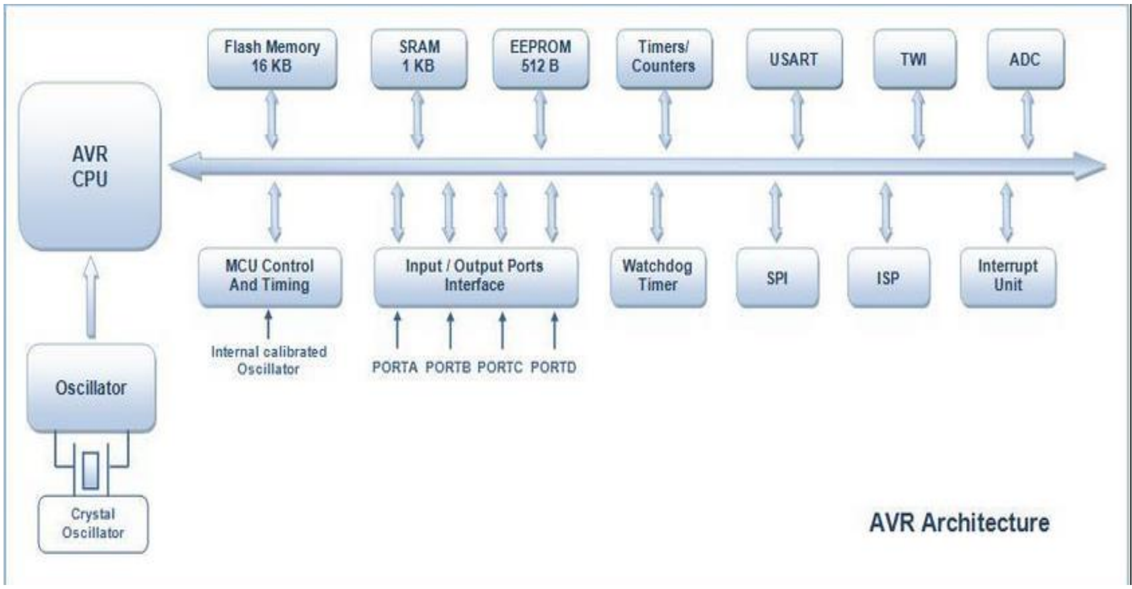


FIG 6.5: AVR Architecture

Microcontroller: Microcontroller can be termed as a single on chip computer which includes number of peripherals like RAM, EEPROM, Timers etc., required to perform some predefined task.

The computer on one hand is designed to perform all the general purpose tasks on a single machine like you can use a computer to run a software to perform calculations or you can use a computer to store some multimedia file or to access internet through the browser, whereas the microcontrollers are meant to perform only the specific tasks, for e.g., switching the AC off automatically when room temperature drops to a certain defined limit and again turning it ON when temperature rises above the defined limit.

There are number of popular families of microcontrollers which are used in different applications as per their capability and feasibility to perform the desired task, most common of these are 8051, **AVR** and PIC microcontrollers. In this article we will introduce you with **AVR** family of microcontrollers.

AVR was developed in the year 1996 by Atmel Corporation. The architecture of **AVR** was developed by Alf-Egil Bogen and Vegard Wollan. AVR derives its name from its developers and stands for **Alf-Egil Bogen Vegard Wollan RISC microcontroller**, also known as **A**dvanced **V**irtual **R**ISC. The AT90S8515 was the first microcontroller which was based on **AVR architecture** however the first microcontroller to hit the commercial market was AT90S1200 in the year 1997.

AVR microcontrollers are available in three categories:

1. **TinyAVR** – Less memory, small size, suitable only for simpler applications
2. **MegaAVR** – These are the most popular ones having good amount of memory (upto 256 KB), higher number of inbuilt peripherals and suitable for moderate to complex applications.
3. **XmegaAVR** – Used commercially for complex applications, which require large program memory and high speed.

Device architecture

Flash, EEPROM, and SRAM are all integrated onto a single chip, removing the need for external memory in most applications. Some devices have a parallel external bus option to allow adding additional data memory or memory-mapped devices. Almost all devices (except the smallest TinyAVR chips) have serial interfaces, which can be used to connect larger serial EEPROMs or flash chips.

Program memory

Program instructions are stored in non-volatile flash memory. Although the MCUs are 8-bit, each instruction takes one or two 16-bit words.

The size of the program memory is usually indicated in the naming of the device itself (e.g., the ATmega64x line has 64 kB of flash while the ATmega32x line has 32 kB).

There is no provision for off-chip program memory; all code executed by the AVR core must reside in the on-chip flash. However, this limitation does not apply to the AT94 FPSLIC AVR/FPGA chips.

Internal data memory

The data address space consists of the register file, I/O registers, and SRAM.

Internal registers

The AVRs have 32 single-byte registers and are classified as 8-bit RISC devices.

In most variants of the AVR architecture, the working registers are mapped in as the first 32 memory addresses (0000_{16}–$001F_{16}$) followed by the 64 I/O registers (0020_{16}–$005F_{16}$).

Actual SRAM starts after these register sections (address 0060_{16}). (Note that the I/O register space may be larger on some more extensive devices, in which case the memory mapped I/O registers will occupy a portion of the SRAM address space.)

Even though there are separate addressing schemes and optimized opcodes for register file and I/O register access, all can still be addressed and manipulated as if they were in SRAM.

In the XMEGA variant, the working register file is not mapped into the data address space; as such, it is not possible to treat any of the XMEGA's working registers as though they were SRAM. Instead, the I/O registers are mapped into the data address space starting at the very beginning of the address space. Additionally, the amount of data address space dedicated to I/O registers has grown substantially to 4096 bytes (0000_{16}–$0FFF_{16}$). As with previous generations, however, the fast I/O manipulation instructions can only reach the first 64 I/O register locations (the first 32 locations for bitwise instructions). Following the I/O registers, the XMEGA series sets aside a 4096 byte range of the data address space which can be used optionally for mapping the internal EEPROM to the data address space (1000_{16}–$1FFF_{16}$). The actual SRAM is located after these ranges, starting at 2000_{16}.

EEPROM

Almost all AVR microcontrollers have internal EEPROM for semi-permanent data storage. Like flash memory, EEPROM can maintain its contents when electrical power is removed.

In most variants of the AVR architecture, this internal EEPROM memory is not mapped into the MCU's addressable memory space. It can only be accessed the same way an external peripheral device is, using special pointer registers and read/write instructions which makes EEPROM access much slower than other internal RAM.

However, some devices in the SecureAVR (AT90SC) family use a special EEPROM mapping to the data or program memory depending on the configuration. The XMEGA family also allows the EEPROM to be mapped into the data address space.

Since the number of writes to EEPROM is not unlimited — Atmel specifies 100,000 write cycles in their datasheets — a well designed EEPROM write routine should compare the contents of an EEPROM address with desired contents and only perform an actual write if the contents need to be changed.

Note that erase and write can be performed separately in many cases, byte-by-byte, which may also help prolong life when bits only need to be set to all 1s (erase) or selectively cleared to 0s (write).

Program execution

Atmel's AVRs have a two stage, single level pipeline design. This means the next machine instruction is fetched as the current one is executing. Most instructions take just one or two clock cycles, making AVRs relatively fast among eight-bit microcontrollers.

The AVR processors were designed with the efficient execution of compiled C code in mind and have several built-in pointers for the task.

MCU speed

The AVR line can normally support clock speeds from 0 to 20 MHz, with some devices reaching 32 MHz. Lower powered operation usually requires a reduced clock speed. All recent (Tiny, Mega, and Xmega, but not 90S) AVRs feature an on-chip oscillator, removing the need for external clocks or resonator circuitry. Some AVRs also have a system clock prescaler that can divide down the system clock by up to 1024. This prescaler can be reconfigured by software during run-time, allowing the clock speed to be optimized.

Since all operations (excluding literals) on registers R0 - R31 are single cycle, the AVR can achieve up to 1 MIPS per MHz, i.e. an 8 MHz processor can achieve up to 8 MIPS. Loads and stores to/from memory take two cycles, branching takes two cycles. Branches in the latest "3-byte PC" parts such as ATmega2560 are one cycle slower than on previous devices

Features:

• High-performance, Low-power Atmel®AVR® 8-bit Microcontroller

• Advanced RISC Architecture

– 130 Powerful Instructions – Most Single-clock Cycle Execution

– 32 × 8 General Purpose Working Registers

– Fully Static Operation

– Up to 16MIPS Throughput at 16MHz

– On-chip 2-cycle Multiplier

• High Endurance Non-volatile Memory segments

– 8Kbytes of In-System Self-programmable Flash program memory

– 512Bytes EEPROM

– 1Kbyte Internal SRAM

– Write/Erase Cycles: 10,000 Flash/100,000 EEPROM

– Data retention: 20 years at 85°C/100 years at 25°C(1)

– Optional Boot Code Section with Independent Lock Bits

In-System Programming by On-chip Boot Program

True Read-While-Write Operation

– Programming Lock for Software Security

• Peripheral Features

– Two 8-bit Timer/Counters with Separate Prescaler, one Compare Mode

– One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode

– Real Time Counter with Separate Oscillator

– Three PWM Channels

– 8-channel ADC in TQFP and QFN/MLF package

Eight Channels 10-bit Accuracy

– 6-channel ADC in PDIP package

Six Channels 10-bit Accuracy

– Byte-oriented Two-wire Serial Interface

– Programmable Serial USART

– Master/Slave SPI Serial Interface

– Programmable Watchdog Timer with Separate On-chip Oscillator

– On-chip Analog Comparator

• Special Microcontroller Features

– Power-on Reset and Programmable Brown-out Detection

– Internal Calibrated RC Oscillator

– External and Internal Interrupt Sources

– Five Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, and Standby

• I/O and Packages

– 23 Programmable I/O Lines

– 28-lead PDIP, 32-lead TQFP, and 32-pad QFN/MLF

• Operating Voltages

– 2.7V - 5.5V (ATmega8L)

– 4.5V - 5.5V (ATmega8)

• Speed Grades

– 0 - 8MHz (ATmega8L)

– 0 - 16MHz (ATmega8)

• Power Consumption at 4Mhz, 3V, 25oC

– Active: 3.6mA

– Idle Mode: 1.0mA

– Power-down Mode: 0.5µA

Brown-out Detector:

If the Brown-out Detector is not needed in the application, this module should be turned off. If the Brown-out Detector is enabled by the BODEN Fuse, it will be enabled in all sleep modes, and hence, always consume power. In the deeper sleep modes, this will contribute significantly to the total current consumption. Refer to “Brown-out Detection” on page 38 for details on how to configure the Brown-out Detector.

Internal Voltage Reference the Internal Voltage Reference will be enabled when needed by the Brown-out Detector, the Analog Comparator or the ADC. If these modules are disabled as described in the sections above, the internal voltage reference will be disabled and it will not be consuming power. When turned on again, the user must allow the reference to start up before the output is used. If the reference is kept on in sleep mode, the output can be used immediately. Refer to “Internal Voltage Reference” on page 40 for details on the start-up time. Watchdog Timer If the Watchdog Timer is not needed in the application, this module should be turned off.

If the Watchdog Timer is enabled, it will be enabled in all sleep modes, and hence, always consume power. In the deeper sleep modes, this will contribute significantly to the total current consumption. Refer to “Watchdog Timer” on page 41 for details on how to configure the Watchdog Timer. Port Pins When entering a sleep mode, all port pins should be configured to use minimum power.

The most important thing is then to ensure that no pins drive resistive loads. In sleep modes where the both the I/O clock (clkI/O) and the ADC clock (clkADC) are stopped, the input buffers of the device will be disabled. This ensures that no power is consumed by the input logic when not needed. In some cases, the input logic is needed for detecting wake-up conditions, and it will then be enabled. Refer to the section “Digital Input Enable and Sleep Modes” on page 53 for details on which pins are enabled. If the input buffer is enabled and the input signal is left floating or have an analog signal level close to VCC/2, the input buffer will use excessive power.

Power-on Reset:

A Power-on Reset (POR) pulse is generated by an On-chip detection circuit. The detection level is defined in Table 15. The POR is activated whenever VCC is below the detection level. The POR circuit can be used to trigger the Start-up Reset, as well as to detect a failure in supply voltage.

A Power-on Reset (POR) circuit ensures that the device is reset from Power-on. Reaching the Power-on Reset threshold voltage invokes the delay counter, which determines how long the device is kept in RESET after VCC rise. The RESET signal is activated again, without any delay, when VCC decreases below the detection level.

External Reset:

An External Reset is generated by a low level on the RESET pin. Reset pulses longer than the minimum pulse width (see Table 15) will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset. When the applied signal reaches the Reset Threshold Voltage – VRST on its positive edge, the delay counter starts the MCU after the time-out period tTOUT has expired.

6.5 Pin diagram:

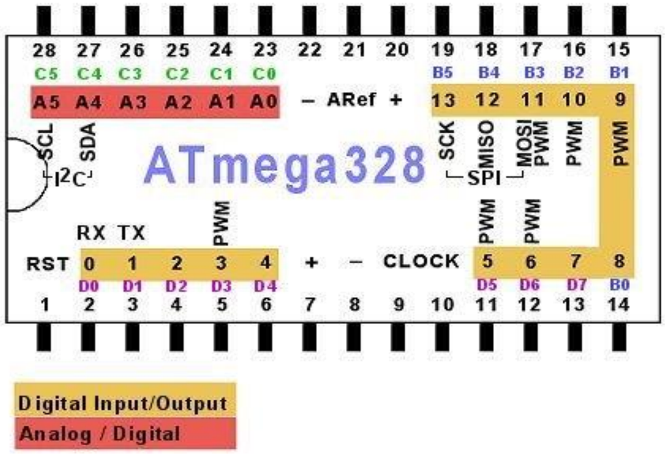


Fig 6.6: PIN DIAGRAM OF ATMEGA328

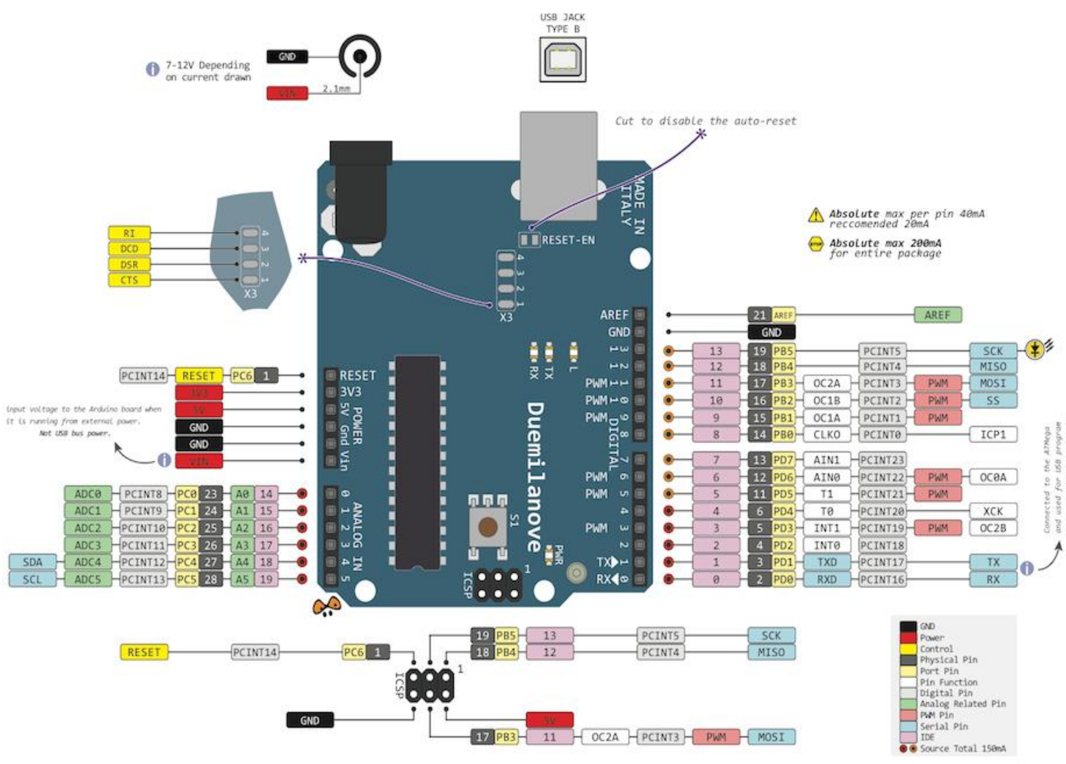


Fig 6.7: PIN DIAGRAM OF ATMEGA328

VCC

Digital supply voltage magnitude of the voltage range between 4.5 to 5.5 V for the ATmega8 and 2.7 to 5.5 V for ATmega8L

GND

Ground Zero reference digital voltage supply.

PORTB (PB7.. PB0)

PORTB is a port I / O two-way (bidirectional) 8-bit with internal pull-up resistor can be selected. This port output buffers have symmetrical characteristics when used as a source or sink. When used as an input, the pull-pin low externally will emit a current if the pull-up resistor is activated it. PORTB pins will be in the condition of the tri-state when RESET is active, although the clock is not running.

PORTC (PC5.. PC0)

PORTC is a port I / O two-way (bidirectional) 7-bit with internal pull-up resistor can be selected. This port output buffers have symmetrical characteristics when used as a source or sink. When used as an input, the pull-pin low externally will emit a current if the pull-up resistor is activated it. PORTC pins will be in the condition of the tri-state when RESET is active, although the clock is not running.

PC6/RESET

If RSTDISBL Fuse programmed, PC6 then serves as a pin I / O but with different characteristics. PC0 to PC5 If Fuse RSTDISBL not programmed, then serves as input Reset PC6. LOW signal on this pin with a minimum width of 1.5 microseconds will bring the microcontroller into reset condition, although the clock is not running.

PORTD (PD7.. PD0)

PORTD is a port I / O two-way (bidirectional) 8-bit with internal pull-up resistor can be selected. This port output buffers have symmetrical characteristics when used as a source or sink. When used as an input, the pull-pin low externally will emit a current if the pull-up resistor is activated it. PORTD pins will be in the condition of the tri-state when RESET is active, although the clock is not running.

RESET

Reset input pin. LOW signal on this pin with a minimum width of 1.5 microseconds will bring the microcontroller into reset condition, although the clock is not running. Signal with a width of less than 1.5 microseconds does not guarantee a Reset condition.

AVCC

AVCC is the supply voltage pin for the ADC, PC3 .. PC0, and ADC7..ADC6. This pin should be connected to VCC, even if the ADC is not used. If the ADC is used, AVCC should be connected to VCC through a low-pass filter to reduce noise.

Aref

Analog Reference pin for the ADC.

ADC7 .. ADC6

ADC analog input there is only on ATmega8 with TQFP and QFP packages / MLF.

PORTS

Term "port" refers to a group of pins on a microcontroller which can be accessed simultaneously, or on which we can set the desired combination of zeros and ones, or read from them an existing status. Physically, port is a register inside a microcontroller which is connected by wires to the pins of a microcontroller. Ports represent physical connection of Central Processing Unit with an outside world. Microcontroller uses them

The Atmega8 has 23 I/O ports which are organized into 3 groups:

- Port B (PB0 to PB7)
- Port C (PC0 to PC6)
- Port D (PD0 to PD7)

We will use mainly 3 registers known as **DDRX, PORTX** & **PINX**. We have total four PORTs on my ATmega16. They are **PORTA, PORTB, PORTC** and **PORTD**. They are multifunctional pins. Each of the pins in each port (total 32) can be treated as input or output pin.

Applications

AVR microcontroller perfectly fits many uses, from automotive industries and controlling home appliances to industrial instruments, remote sensors, electrical door locks and safety devices. It is also ideal for smart cards as well as for battery supplied devices because of its low consumption.

EEPROM memory makes it easier to apply microcontrollers to devices where permanent storage of various parameters is needed (codes for transmitters, motor speed, receiver frequencies, etc.). Low cost, low consumption, easy handling and flexibility make ATmega8 applicable even in areas where microcontrollers had not previously been considered (example: timer functions, interface replacement in larger systems, coprocessor applications, etc.).

In System Programmability of this chip (along with using only two pins in data transfer) makes possible the flexibility of a product, after assembling and testing have been completed. This capability can be used to create assembly-line production, to store calibration data available only after final testing, or it can be used to improve programs on finished products.

6.6 Regulated Power Supply:

A regulated power supply converts unregulated AC (Alternating Current) to a constant DC (Direct Current). A regulated power supply is used to ensure that the output remains constant even if the input changes. A regulated DC power supply is also known as a linear power supply; it is an embedded circuit and consists of various blocks. The regulated power supply will accept an AC input and give a constant DC output. The figure below shows the block diagram of a typical regulated DC power supply.

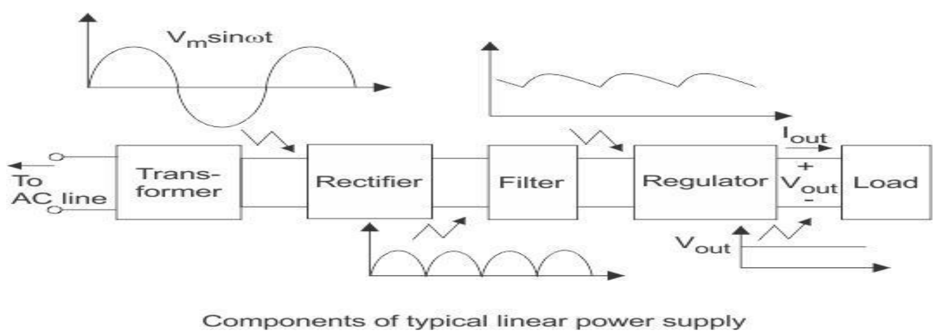


Figure 6.8: Block diagram of power supply

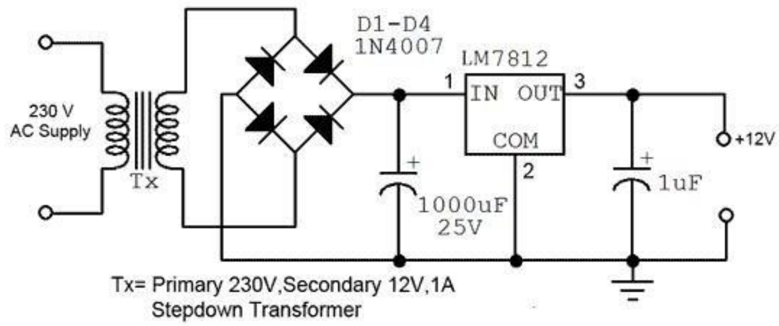


Fig 6.9: Power supply circuit diagram

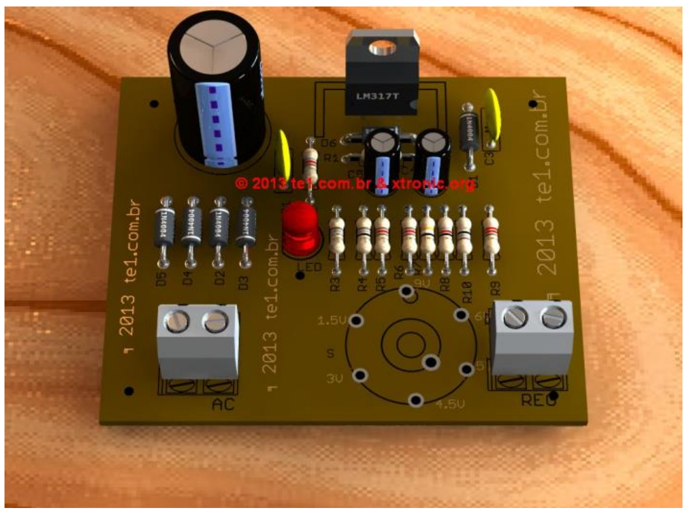


Fig 6.10: Regulated Power Supply

The basic circuit diagram of a regulated power supply (DC O/P) with led connected as load is shown in fig: 3.3.3.

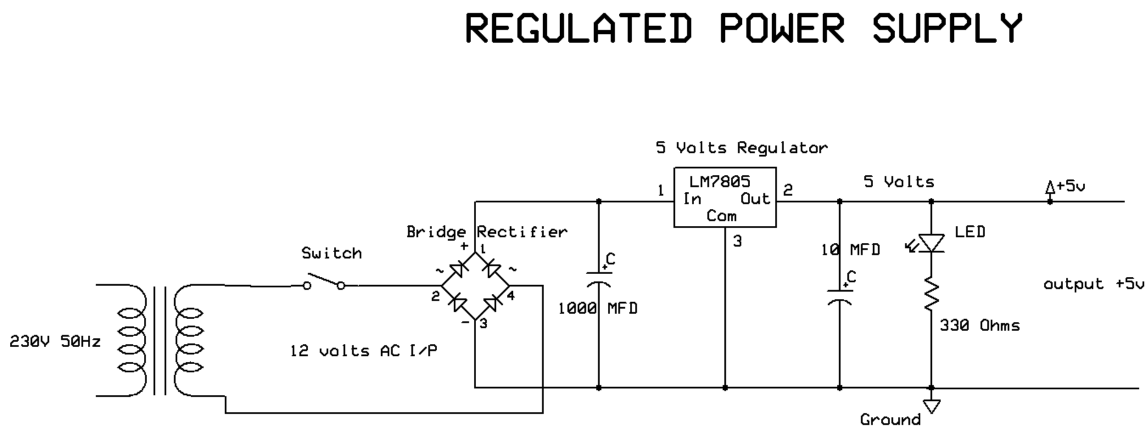


Fig 6.11: Circuit diagram of Regulated Power Supply with Led connection

The components mainly used in above figure are

- 230V AC MAINS
- TRANSFORMER
- BRIDGE RECTIFIER(DIODES)
- CAPACITOR
- VOLTAGE REGULATOR(IC 7805)
- RESISTOR
- LED(LIGHT EMITTING DIODE)

The detailed explanation of each and every component mentioned above is as follows:

Transformation: The process of transforming energy from one device to another is called transformation. For transforming energy we use transformers.

Step down transformer:

Incase of step down transformer, Primary winding induces more flux than the secondary winding, and secondary winding is having less number of turns because of that it accepts less number of flux, and releases less amount of voltage.

Bridge full wave rectifier:

The Bridge rectifier circuit is shown in fig:3.8, which converts an ac voltage to dc voltage using both half cycles of the input ac voltage. The Bridge rectifier circuit is shown in the figure. The circuit has four diodes connected to form a bridge. The ac input voltage is applied to the diagonally opposite ends of the bridge. The load resistance is connected between the other two ends of the bridge.

For the positive half cycle of the input ac voltage, diodes D1 and D3 conduct, whereas diodes D2 and D4 remain in the OFF state. The conducting diodes will be in series with the load resistance R_L and hence the load current flows through $R_{L.}$

For the negative half cycle of the input ac voltage, diodes D2 and D4 conduct whereas, D1 and D3 remain OFF. The conducting diodes D2 and D4 will be in series with the load resistance R_L and hence the current flows through R_L in the same direction as in the previous half cycle. Thus a bi-directional wave is converted into a unidirectional wave.

Input Output

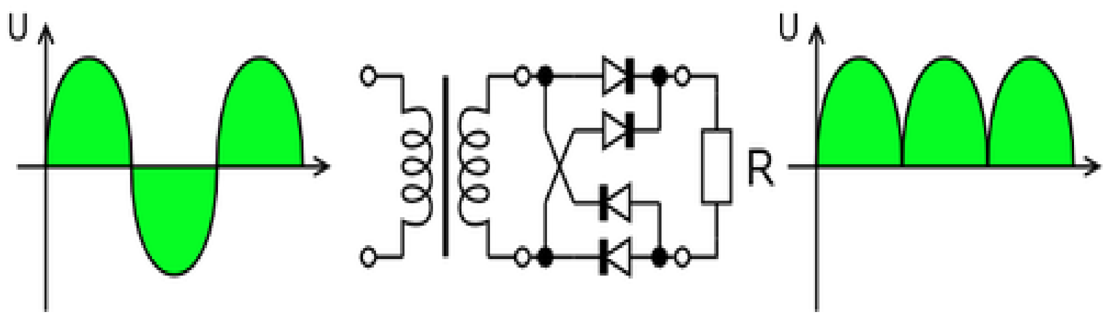


Fig 6.12: Bridge rectifier: a full-wave rectifier using 4 diodes

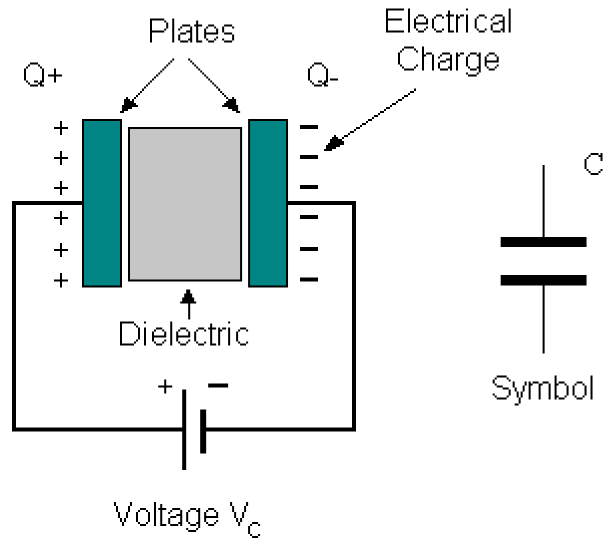
DB107:

Now -a -days Bridge rectifier is available in IC with a number of DB107. In our project we are using an IC in place of bridge rectifier. The picture of DB 107 is shown in fig: 3.3.8.

Fig 6.13: DB107

Capacitor Filtration:

The process of converting a pulsating direct current to a pure direct current using filters is called as filtration._Electronic filters are electronic circuits, which perform signal-processing functions, specifically to remove unwanted frequency components from the signal, to enhance wanted ones.The **Capacitor** or sometimes referred to as a Condenser is a passive device, and one which stores energy in the form of an electrostatic field which produces a potential (static voltage) across its plates. In its basic form a capacitor consists of two parallel conductive plates that are not connected but are electrically separated either by air or by an insulating material called the Dielectric. When a voltage is applied to these plates, a current flows charging up the plates with electrons giving one plate a positive charge and the other plate an equal and opposite negative charge this flow of electrons to the plates is known as the Charging Current and continues to flow until the voltage across the plates (and hence the capacitor) is equal to the applied voltage Vcc. At this point the capacitor is said to be fully charged and this is illustrated below. The construction of capacitor and an electrolytic capacitor are shown in figures 3.3.9 and 3.3.10 respectively.



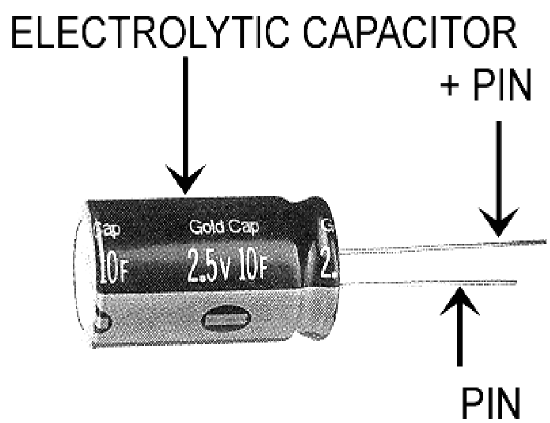


Fig 6.14: Construction Of a Capacitor **Fig 6.15: Electrolytic**

Voltage Regulation:

The process of converting a varying voltage to a constant regulated voltage is called as regulation. For the process of regulation we use voltage regulators.

Voltage Regulator:

A voltage regulator (also called a 'regulator') with only three terminals appears to be a simple device, but it is in fact a very complex integrated circuit. It converts a varying input voltage into a constant 'regulated' output voltage. Voltage Regulators are available in a variety of outputs like 5V, 6V, 9V, 12V and 15V. The LM78XX series of voltage regulators are designed for positive input. For applications requiring negative input, the LM79XX series is used. Using a pair of 'voltage-divider' resistors can increase the output voltage of a regulator circuit.

It is not possible to obtain a voltage lower than the stated rating. You cannot use a 12V regulator to make a 5V power supply. Voltage regulators are very robust. These can withstand over-current draw due to short circuits and also over-heating. In both cases, the regulator will cut off before any damage occurs. The only way to destroy a regulator is to apply reverse voltage to its input. Reverse polarity destroys the regulator almost instantly. Fig: 3.3.11 shows voltage regulator.

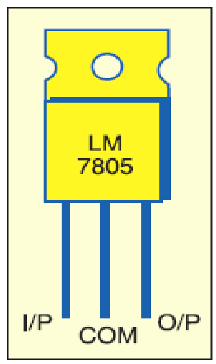


Fig 6.16: Voltage Regulator

Resistor:

A resistor is a two-terminal electronic component that produces a voltage across its terminals that is proportional to the electric current passing through it in accordance with Ohm's law:

$$V = IR$$

Resistors are elements of electrical networks and electronic circuits and are ubiquitous in most electronic equipment. Practical resistors can be made of various compounds and films, as well as resistance wire (wire made of a high-resistivity alloy, such as nickel/chrome).

The primary characteristics of a resistor are the resistance, the tolerance, maximum working voltage and the power rating. Other characteristics include temperature coefficient, noise, and inductance. Less well-known is critical resistance, the value below which power dissipation limits the maximum permitted current flow, and above which the limit is applied voltage. Critical resistance is determined by the design, materials and dimensions of the resistor.

dissipated by a resistor (or the equivalent resistance of a resistor network) is calculated using the following:

$$P = I^2 R = IV = \frac{V^2}{R}$$

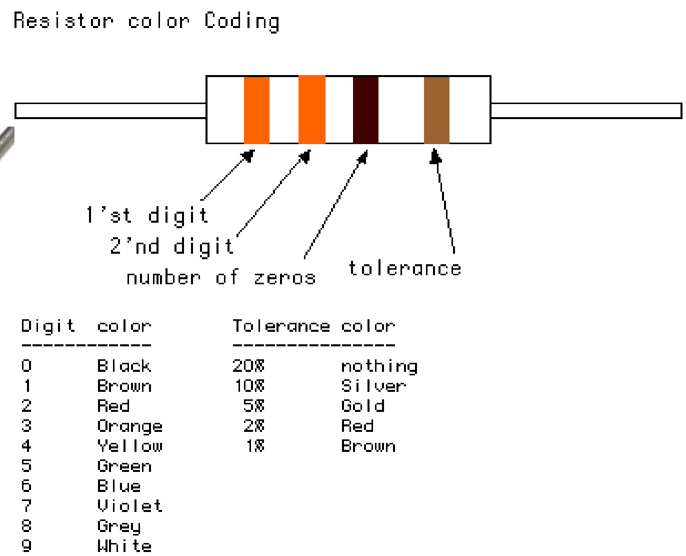


Fig 6.17: Resistor **Fig 6.18: Color Bands In Resistor**

LED:

A light-emitting diode (LED) is a semiconductor light source. LED's are used as indicator lamps in many devices, and are increasingly used for lighting. Introduced as a practical electronic component in 1962, early LED's emitted low-intensity red light, but modern versions are available across the visible, ultraviolet and infrared wavelengths, with very high brightness. The internal structure and parts of a led are shown in figures 3.4.1 and 3.4.2 respectively.

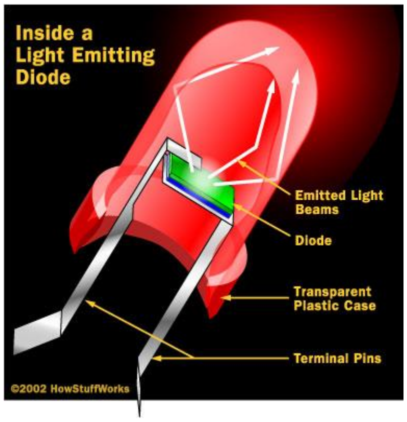


Fig 6.19: Inside a LED

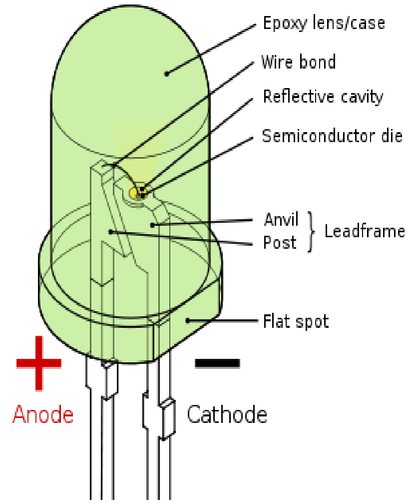


Fig 6.20: Parts of a LED

6.7 Buzzer

Basically, the sound source of a piezoelectric sound component is a piezoelectric diaphragm. A piezoelectric diaphragm consists of a piezoelectric ceramic plate which has electrodes on both sides and a metal plate (brass or stainless steel, etc.). A piezoelectric ceramic plate is attached to a metal plate with adhesives. Applying D.C. voltage between electrodes of a piezoelectric diaphragm causes mechanical distortion due to the piezoelectric effect. For a misshaped piezoelectric element, the distortion of the piezoelectric element expands in a radial direction. And the piezoelectric diaphragm bends toward the direction.

The metal plate bonded to the piezoelectric element does not expand. Conversely, when the piezoelectric element shrinks, the piezoelectric diaphragm bends in the direction Thus, when AC voltage is applied across electrodes, the bending is repeated, producing sound waves in the air. To interface a buzzer the standard transistor interfacing circuit is used. Note that if a different power supply is used for the buzzer, the 0V rails of each power supply must be connected to provide a common reference.

If a battery is used as the power supply, it is worth remembering that piezo sounders draw much less current than buzzers. Buzzers also just have one ‘tone’, whereas a piezo sounder is able to create sounds of many different tones. To switch on buzzer -high 1 To switch off buzzer -low 1

Fig 6.21: Picture of buzzer

6.8 LCD DISPLAY

LCD Background:

One of the most common devices attached to a micro controller is an LCD display. Some of the most common LCD's connected to the many microcontrollers are 16x2 and 20x2 displays. This means 16 characters per line by 2 lines and 20 characters per line by 2 lines, respectively.

Basic 16x 2 Characters LCD

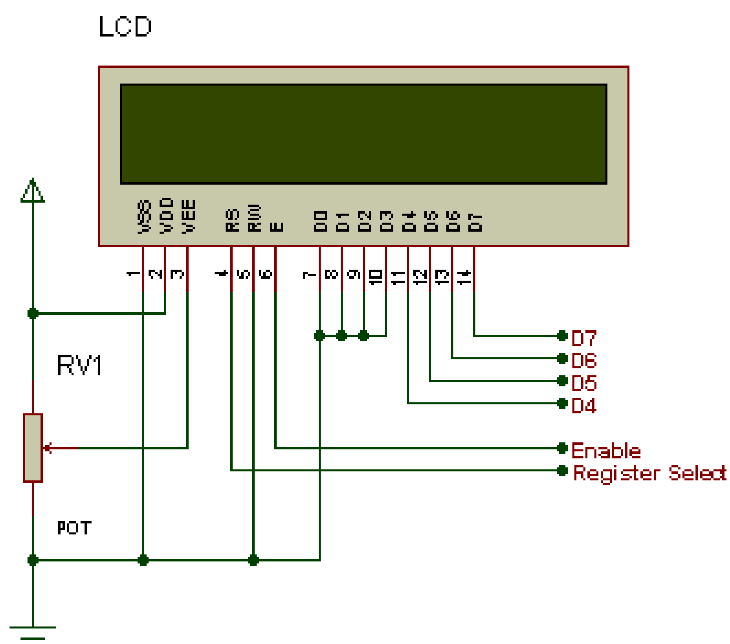


Fig 6.22: LCD Pin diagram

Pin description:

Pin No.	Name	Description
Pin no. 1	**VSS**	Power supply (GND)
Pin no. 2	**VCC**	Power supply (+5V)
Pin no. 3	**VEE**	Contrast adjust
Pin no. 4	**RS**	0 = Instruction input 1 = Data input
Pin no. 5	**R/W**	0 = Write to LCD module 1 = Read from LCD module
Pin no. 6	**EN**	Enable signal
Pin no. 7	**D0**	Data bus line 0 (LSB)
Pin no. 8	**D1**	Data bus line 1
Pin no. 9	**D2**	Data bus line 2
Pin no. 10	**D3**	Data bus line 3
Pin no. 11	**D4**	Data bus line 4
Pin no. 12	**D5**	Data bus line 5
Pin no. 13	**D6**	Data bus line 6
Pin no. 14	**D7**	Data bus line 7 (MSB)

Table 1: Character LCD pins with Microcontroller

The LCD requires 3 control lines as well as either 4 or 8 I/O lines for the data bus. The user may select whether the LCD is to operate with a 4-bit data bus or an 8-bit data bus. If a 4-bit data bus is used the LCD will require a total of 7 data lines (3 control lines plus the 4 lines for the data bus). If an 8-bit data bus is used the LCD will require a total of 11 data lines (3 control lines plus the 8 lines for the data bus).

The three control lines are referred to as **EN**, **RS**, and **RW**.

The **EN** line is called "Enable." This control line is used to tell the LCD that we are sending it data. To send data to the LCD, our program should make sure this line is low (0) and then set the other two control lines and/or put data on the data bus. When the other lines are completely ready, bring **EN** high (1) and wait for the minimum amount of time required by the LCD datasheet (this varies from LCD to LCD), and end by bringing it low (0) again.

The **RS** line is the "Register Select" line. When RS is low (0), the data is to be treated as a command or special instruction (such as clear screen, position cursor, etc.). When RS is high (1), the data being sent is text data which should be displayed on the screen. For example, to display the letter "T" on the screen we would set RS high.

The **RW** line is the "Read/Write" control line. When RW is low (0), the information on the data bus is being written to the LCD. When RW is high (1), the program is effectively querying (or reading) the LCD. Only one instruction ("Get LCD status") is a read command. All others are write commands--so RW will almost always be low.

Finally, the data bus consists of 4 or 8 lines (depending on the mode of operation selected by the user). In the case of an 8-bit data bus, the lines are referred to as DB0, DB1, DB2, DB3, DB4, DB5, DB6, and DB7.

Schematic:

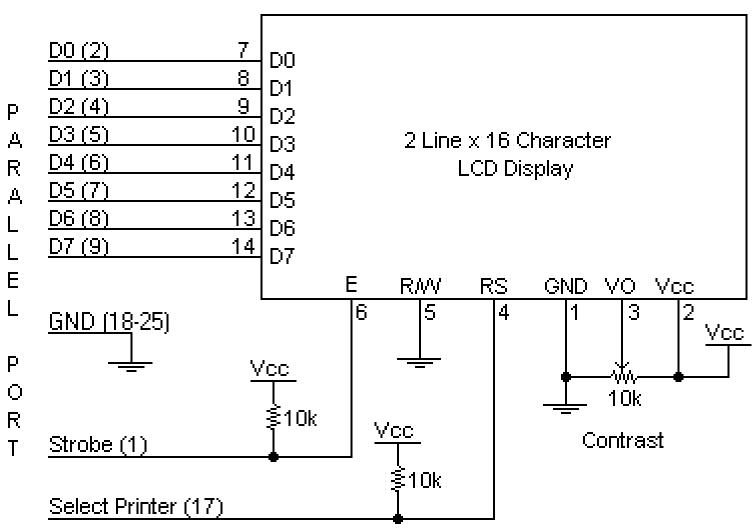


Fig 6.23: Schematic diagram

Circuit Description:

Above is the quite simple schematic. The LCD panel's Enable and Register Select is connected to the Control Port. The Control Port is an open collector / open drain output. While most Parallel Ports have internal pull-up resistors, there is a few which don't. Therefore by incorporating the two 10K external pull up resistors, the circuit is more portable for a wider range of computers, some of which may have no internal pull up resistors.

We make no effort to place the Data bus into reverse direction. Therefore we hard wire the R/W line of the LCD panel, into write mode. This will cause no bus conflicts on the data lines. As a result we cannot read back the LCD's internal Busy Flag which tells us if the LCD has accepted and finished processing the last instruction. This problem is overcome by inserting known delays into our program.

The 10k Potentiometer controls the contrast of the LCD panel. Nothing fancy here. As with all the examples, I've left the power supply out. We can use a bench power supply set to 5v or use an onboard +5 regulator. Remember a few de-coupling capacitors, especially if we have trouble with the circuit working properly.

SETB RW

Handling the EN control line:

As we mentioned above, the EN line is used to tell the LCD that we are ready for it to execute an instruction that we've prepared on the data bus and on the other control lines. Note that the EN line must be raised/ lowered before/after each instruction sent to the LCD regardless of whether that instruction is read or write text or instruction. In short, we must always manipulate EN when communicating with the LCD. EN is the LCD's way of knowing that we are talking to it. If we don't raise/lower EN, the LCD doesn't know we're talking to it on the other lines.

Thus, before we interact in any way with the LCD we will always bring the **EN** line low with the following instruction:

CLR EN

And once we've finished setting up our instruction with the other control lines and data bus lines, we'll always bring this line high:

SETB EN

The line must be left high for the amount of time required by the LCD as specified in its datasheet. This is normally on the order of about 250 nanoseconds, but checks the datasheet. In the case of a typical microcontroller running at 12 MHz, an instruction requires 1.08 microseconds to execute so the EN line can be brought low the very next instruction. However, faster microcontrollers (such as the DS89C420 which executes an instruction in 90 nanoseconds given an 11.0592 MHz crystal) will require a number of NOPs to create a delay while EN is held high. The number of NOPs that must be inserted depends on the microcontroller we are using and the crystal we have selected.

The instruction is executed by the LCD at the moment the EN line is brought low with a final CLR EN instruction.

Checking the busy status of the LCD:

As previously mentioned, it takes a certain amount of time for each instruction to be executed by the LCD. The delay varies depending on the frequency of the crystal attached to the oscillator input of the LCD as well as the instruction which is being executed.

While it is possible to write code that waits for a specific amount of time to allow the LCD to execute instructions, this method of "waiting" is not very flexible. If the crystal frequency is changed, the software will need to be modified. A more robust method of programming is to use the "Get LCD Status" command to determine whether the LCD is still busy executing the last instruction received.

The "Get LCD Status" command will return to us two tidbits of information; the information that is useful to us right now is found in DB7. In summary, when we issue the "Get LCD Status" command the LCD will immediately raise DB7 if it's still busy executing a command or lower DB7 to indicate that the LCD is no longer occupied. Thus our program can query the LCD until DB7 goes low, indicating the LCD is no longer busy. At that point we are free to continue and send the next command.

6.9 D.C. Motor:

A dc motor uses electrical energy to produce mechanical energy, very typically through the interaction of magnetic fields and current-carrying conductors. The reverse process, producing electrical energy from mechanical energy, is accomplished by an alternator, generator or dynamo. Many types of electric motors can be run as generators, and vice versa. The input of a DC motor is current/voltage and its output is torque (speed).

Fig 6.24: DC Motor

The DC motor has two basic parts: the rotating part that is called the armature and the stationary part that includes coils of wire called the field coils. The stationary part is also called the stator. Figure shows a picture of a typical DC motor, Figure shows a picture of a DC armature, and Fig shows a picture of a typical stator. From the picture you can see the armature is made of coils of wire wrapped around the core, and the core has an extended shaft that rotates on bearings. You should also notice that the ends of each coil of wire on the armature are terminated at one end of the armature. The termination points are called the commutator, and this is where the brushes make electrical contact to bring electrical current from the stationary part to the rotating part of the machine.

Operation:

The DC motor you will find in modem industrial applications operates very similarly to the simple DC motor described earlier in this chapter. Figure 12-9 shows an electrical diagram of a simple DC motor. Notice that the DC voltage is applied directly to the field winding and the brushes. The armature and the field are both shown as a coil of wire. In later diagrams, a field resistor will be added in series with the field to control the motor speed.

When voltage is applied to the motor, current begins to flow through the field coil from the negative terminal to the positive terminal. This sets up a strong magnetic field in the field winding. Current also begins to flow through the brushes into a commutator segment and then through an armature coil. The current continues to flow through the coil back to the brush that is attached to other end of the coil and returns to the DC power source. The current flowing in the armature coil sets up a strong magnetic field in the armature.

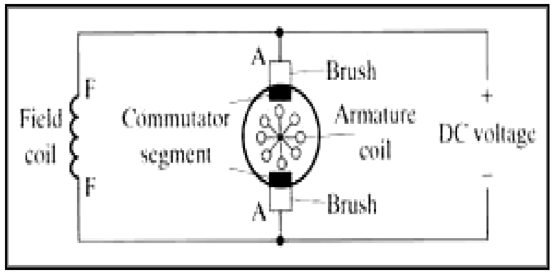


Fig 6.25: Simple electrical diagram of DC motor

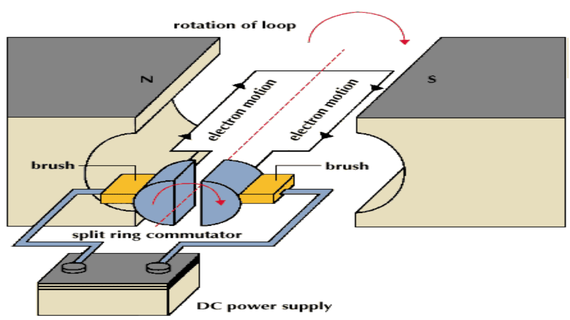


Fig: 6.26: Operation of a DC Motor

The magnetic field in the armature and field coil causes the armature to begin to rotate. This occurs by the unlike magnetic poles attracting each other and the like magnetic poles repelling each other. As the armature begins to rotate, the commutator segments will also begin to move under the brushes. As an individual commutator segment moves under the brush connected to positive voltage, it will become positive, and when it moves under a brush connected to negative voltage it will become negative. In this way, the commutator segments continually change polarity from positive to negative. Since the commutator segments are connected to the ends of the wires that make up the field winding in the armature, it causes the magnetic field in the armature to change polarity continually from North Pole to South Pole. The commutator segments and brushes are aligned in such a way that the switch in polarity of the armature coincides with the location of the armature's magnetic field and the field winding's magnetic field. The switching action is timed so that the armature will not lock up magnetically with the field. Instead the magnetic fields tend to build on each other and provide additional torque to keep the motor shaft rotating.

When the voltage is de-energized to the motor, the magnetic fields in the armature and the field winding will quickly diminish and the armature shaft's speed will begin to drop to zero. If voltage is applied to the motor again, the magnetic fields will strengthen and the armature will begin to rotate again.

6.10: GPS MODULE

Global Positioning System

6.10.1. Introduction:

The Global Positioning System (GPS) is a burgeoning technology, which provides unequalled accuracy and flexibility of positioning for navigation, surveying and GIS data capture. The GPS NAVSTAR (Navigation Satellite timing and Ranging Global Positioning System) is a satellite-based navigation, timing and positioning system. The GPS provides continuous three-dimensional positioning 24 hrs a day throughout the world. The technology seems to be beneficiary to the GPS user community in terms of obtaining accurate data up to about100 meters for navigation, meter-level for mapping, and down to millimeter level for geodetic positioning. The GPS technology has tremendous amount of applications in GIS data collection, surveying, and mapping.

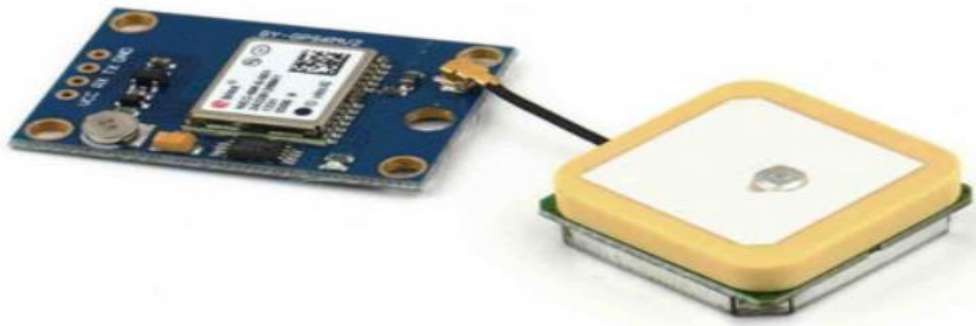


Fig 6.27: GPS Module

The Global Positioning System (GPS) is a U.S. space-based radio navigation system that provides reliable positioning, navigation, and timing services to civilian users on a continuous worldwide basis -- freely available to all. For anyone with a GPS receiver, the system will provide location and time. GPS provides accurate location and time information for an unlimited number of people in all weather, day and night, anywhere in the world.

The Global Positioning System (GPS) is a satellite-based navigation system made up of a network of 24 satellites placed into orbit by the U.S. Department of Defense. GPS was originally intended for military applications, but in the 1980s, the government made the system available for civilian use. GPS works in any weather conditions, anywhere in the world, 24 hours a day. There are no subscription fees or setup charges to use GPS.

The GPS is made up of three parts: satellites orbiting the Earth; control and monitoring stations on Earth; and the GPS receivers owned by users. GPS satellites broadcast signals from space that are picked up and identified by GPS receivers. Each GPS receiver then provides three-dimensional location (latitude, longitude, and altitude) plus the time.

Individuals may purchase GPS handsets that are readily available through commercial retailers. Equipped with these GPS receivers, users can accurately locate where they are and easily navigate to where they want to go, whether walking, driving, flying, or boating. GPS has become a mainstay of transportation systems worldwide, providing navigation for aviation, ground, and maritime operations. Disaster relief and emergency services depend upon GPS for location and timing capabilities in their life-saving missions. Everyday activities such as banking, mobile phone operations, and even the control of power grids, are facilitated by the accurate timing provided by GPS. Farmers, surveyors, geologists and countless others perform their work more efficiently, safely, economically, and accurately using the free and open GPS signals.

Geo positioning -- Basic Concepts:

By positioning we understand the determination of stationary or moving objects. These can be determined as follows:

1. In relation to a well-defined coordinate system, usually by three coordinate values and
2. In relation to other point, taking one point as the origin of a local coordinate system.

The first mode of positioning is known as point positioning, the second as relative positioning. If the object to be positioned is stationary, we term it as static positioning. When the object is moving, we call it kinematics positioning. Usually, the static positioning is used in surveying and the kinematics position in navigation.

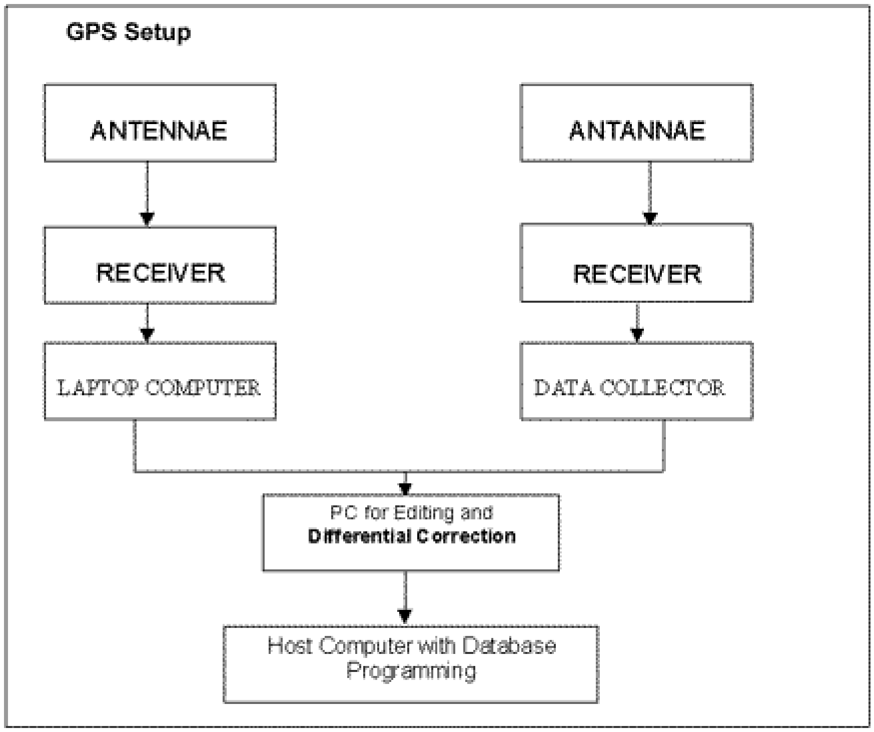


Fig 6.28: GPS Setup

This is a complete GPS module that is based on the NEO-6M. This unit uses the latest technology to give the best possible positioning information and includes a larger built-in 25 x 25mm active GPS antenna with a UART TTL socket. A battery is also included so that you can obtain a GPS lock faster. This is an updated GPS module that can be used with ardupilot mega v2. This GPS module gives the best possible position information, allowing for better performance with your Ardupilot or other Multirotor control platform.

The NEO-6M GPS engine on this board is a quite good one, with the high precision binary output. It has also high sensitivity for indoor applications. NEO-6M GPS Module has a battery for power backup and EEPROM for storing configuration settings. The antenna is connected to the module through a ufl cable which allows for flexibility in mounting the GPS such that the antenna will always see the sky for best performance. This makes it powerful to use with cars and other mobile applications.

The GPS module has serial TTL output, it has four pins: TX, RX, VCC, and GND. You can download the u-center software for configuring the GPS and changing the settings and much more.

Features NEO-6M GPS Module:-

- 5Hz position update rate
- Operating temperature range: -40 TO 85°CUART TTL socket
- EEPROM to save configuration settings
- Rechargeable battery for Backup
- The cold start time of 38 s and Hot start time of 1 s
- Supply voltage: 3.3 V
- Configurable from 4800 Baud to 115200 Baud rates. (default 9600)
- SuperSense ® Indoor GPS: -162 dBm tracking sensitivity
- Support SBAS (WAAS, EGNOS, MSAS, GAGAN)
- Separated 18X18mm GPS antenna

6.11 GSM

Global System for Mobile Communication (GSM)

Definition:

GSM, which stands for Global System for Mobile communications, reigns (important) as the world’s most widely used cell phone technology. Cell phones use a cell phone service carrier’s GSM network by searching for cell phone towers in the nearby area.

Global system for mobile communication (GSM) is a globally accepted standard for digital cellular communication.

GSM is the name of a standardization group established in 1982 to create a common European mobile telephone standard that would formulate specifications for a pan-European mobile cellular radio system operating at 900 MHz. It is estimated that many countries outside of Europe will join the GSM partnership.

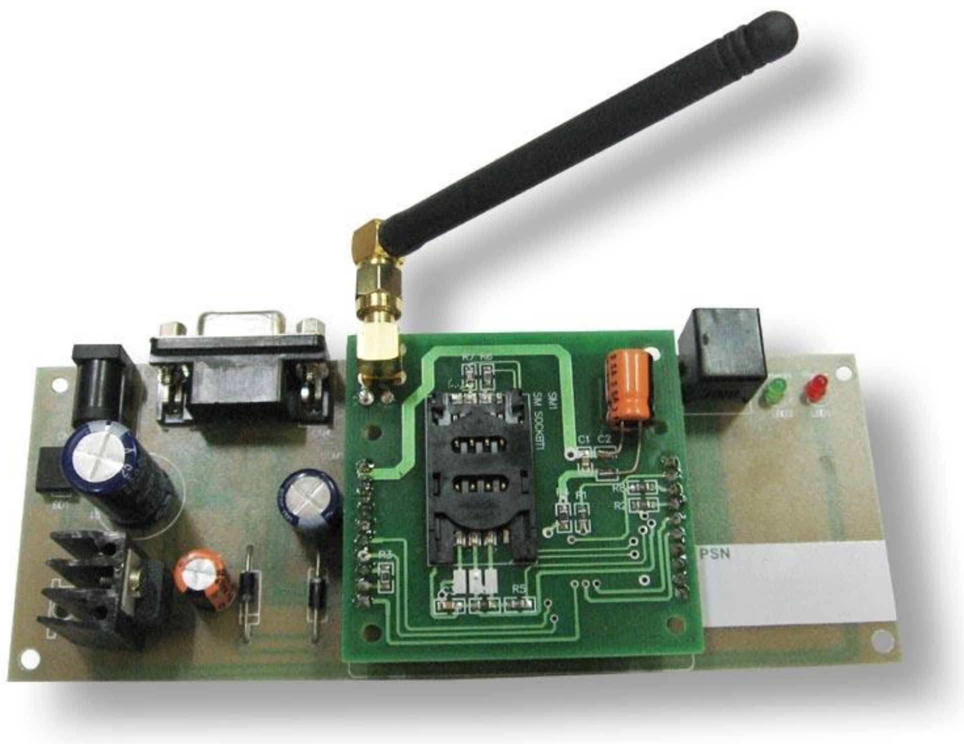


Fig 6.29: GSM Module

MODEM SPECIFICATIONS:

The SIM300 is a complete Tri-band GSM solution in a compact plug-in module.

Featuring an industry-standard interface, the SIM300 delivers GSM/GPRS900/1800/1900Mhz performance for voice, SMS, data and Fax in a small form factor and with low power consumption.

The leading features of SIM300 make it deal fir virtually unlimited application, such as WLL applications (Fixed Cellular Terminal), M2M application, handheld devices and much more.

1. Tri-band GSM/GPRS module with a size of 40x33x2.85
2. Customized MMI and keypad/LCD support
3. An embedded powerful TCP/IP protocol stack
4. Based upon mature and field proven platform, backed up by our support service, from definition to design and production.

General Features:

- Tri-band GSM/GPRS900/1800/1900Mhz
- GPRS multi-slot class 10
- GPRS mobile station class –B
- Complaint to GSM phase 2/2+
    i. -class 4(2W @900MHz)
    ii. -class 1(1W @/18001900MHz)
- Dimensions: 40x33x2.85 mm
- Weight: 8gm
- 7. Control via AT commands
- (GSM 07.07, 07.05 and SIMCOM enhanced AT commands)
- SIM application tool kit
- supply voltage range 3.5…….4.5 v
- Low power consumption
- Normal operation temperature: -20 ’C to +55 ‘C
- Restricted operation temperature : -20 ’C to -25 ‘C and +55 ’C to +70 ‘C

- storage temperature: -40 ‘C to +80 ‘C

Specifications for Fax:

Group 3 and class 1

Specifications for Data:

- GPRS class 10: max 85.6 kbps (downlink)
- PBCCH support
- coding schemes Cs 1,2,3,4
- CSD upto 14.4 kbps
- USSD
- Non transperant mode
- PPP-stack
-

Specifications for SMS via GSM/GPRS:

- Point to point MO and MT
- SMS cell broadcast
- Text and PDU mode

Compatibility:

At cellular command interface

Specifications for voice:

1. Tricodec

-Half rate (HR)

-Full rate (FR)

-Enhanced full rate (EFR)

2. Hands free operation

(Echo cancellation)

Drivers:

Microsoft windows mobile RIL driver

MUX driver

Interfaces:

- Interface to external SIM 3v 1.8v
- 60 pins board-to-board connector
- Two analog audio interfaces
- Keypad interfaces
- LCD interface
- RTC backup
- AT commands via serial interface
- Dual-Serial interfaces
- Antenna connector and antenna pad

Approvals:

- FTA
- Local type approval
- CE

Need of GSM:

The GSM study group aimed to provide the followings through the GSM:

- Improved spectrum efficiency.
- International roaming.
- Low-cost mobile sets and base stations (BS)

- High-quality speech
- Compatibility with Integrated Services Digital Network (ISDN) and other telephone company services.
- Support for new services.

Vibration TILT sensor

6.12 Mercury switch

A mercury switch is an electrical switch that opens and closes a circuit when a small amount of the liquid metal mercury connects metal electrodes to close the circuit. There are several different basic designs (tilt, displacement, radial, etc.) but they all share the common design strength of non-eroding switch contacts.

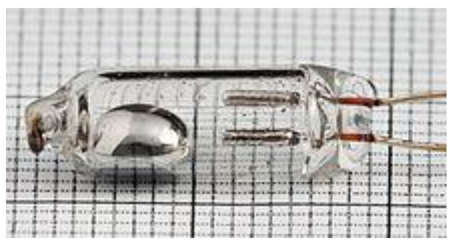


Fig 6.30: Mercury Switch

The most common is the mercury tilt switch. It is in one state (open or closed) when tilted one direction with respect to horizontal, and the other state when tilted the other direction. This is what older style thermostats used to turn a heater or air conditioner on or off.

The mercury displacement switch uses a 'plunger' that dips into a pool of mercury, raising the level in the container to contact at least one electrode. This design is used in relays in industrial applications that need to switch high current loads frequently. These relays use electromagnetic coils to pull steel sleeves inside hermetically sealed containers.

Fig 6.31: Plunger

Description:

Mercury switches have one or more sets of electrical contacts in a sealed glass envelope that contains a small quantity of mercury. The envelope may also contain air, an inert gas, or a vacuum. Gravity constantly pulls the drop of mercury to the lowest point in the envelope. When the switch is tilted in the appropriate direction, the mercury touches a set of contacts, thus completing an electrical circuit. Tilting the switch in the opposite direction moves the mercury away from that set of contacts, breaking that circuit.[1] The switch may contain multiple sets of contacts, closing different sets at different angles, allowing, for example, single-pole, double-throw (*SPDT*) operation.

Advantages

Mercury switches offer several advantages over other switch types:

- The contacts are enclosed, so oxidation of the contact points is unlikely.
- In hazardous locations, interrupting the circuit does not emit a spark that could ignite flammable gases.
- Contacts stay clean, and even if an internal arc occurs, the contact surfaces renew on every operation, so they don't wear out.
- Even a small drop of mercury has low resistance, so switches can carry useful amounts of current in a small size.[2]

- Sensitivity of the drop to gravity provides a unique sensing function, and lends itself to simple, low-force mechanisms for manual or automatic operation.
- The switches are quiet, as no contacts abruptly snap together.
- The mass of the moving mercury drop provides an over center effect to avoid chattering as the switch tilts.
- The envelope can include contacts for two or more circuits.

6.13 PUSH BUTTON/ CONTROL SWITCH:

A push-button (also spelled pushbutton) (press-button in the UK) or simply button is a simple switch mechanism for controlling some aspect of a machine or a process. Buttons are typically made out of hard material, usually plastic or metal. The surface is usually flat or shaped to accommodate the human finger or hand, so as to be easily depressed or pushed. Buttons are most often biased switches, though even many un-biased buttons (due to their physical nature) require a spring to return to their un-pushed state. Different people use different terms for the "pushing" of the button, such as press, depress, mash, and punch.

Uses:

The "push-button" has been utilized in calculators, push-button telephones, kitchen appliances, and various other mechanical and electronic devices, home and commercial.

In industrial and commercial applications, push buttons can be linked together by a mechanical linkage so that the act of pushing one button causes the other button to be released. In this way, a stop button can "force" a start button to be released. This method of linkage is used in simple manual operations in which the machine or process have no electrical circuits for control.

Pushbuttons are often color-coded to associate them with their function so that the operator will not push the wrong button in error. Commonly used colors are red for stopping the machine or process and green for starting the machine or process.

Red pushbuttons can also have large heads (called mushroom heads) for easy operation and to facilitate the stopping of a machine. These pushbuttons are called emergency stop buttons and are mandated by the electrical code in many jurisdictions for increased safety. This large mushroom shape can also be found in buttons for use with operators who need to wear gloves for their work and could not actuate a regular flush-mounted push button. As an aid for operators and users in industrial or commercial applications, a pilot light is commonly added to draw the attention of the user and to provide feedback if the button is pushed. Typically this light is included into the center of the pushbutton and a lens replaces the pushbutton hard center disk. The source of the energy to illuminate the light is not directly tied to the contacts on the back of the pushbutton but to the action the pushbutton controls. In this way a start button when pushed will cause the process or machine operation to be started and a secondary contact designed into the operation or process will close to turn on the pilot light and signify the action of pushing the button caused the resultant process or action to start.

In popular culture, the phrase "the button" (sometimes capitalized) refers to a (usually fictional) button that a military or government leader could press to launch nuclear weapons.

Fig 6.32: Push Button

A Load control switch is a remotely controlled relay that is placed on home appliances which consume large amounts of electricity, such as air conditioner units and electric water heaters.

Most load control switches consist of a communication module and the relay switch and can be used as part of a demand response energy efficiency system such as a smart grid. Such a switch operates similarly to a pager, receiving signals from the power company or electrical frequency shift to turn off or reduce power to the appliance during times of peak electrical demand. Usually, the device has a timer that will automatically reset the switch back on after a preset time. Some operation intolerant appliances, such as dryers, use switches that can reduce or shut off power to their heating coils yet still tumble until signaled to resume full power

CHAPTER 7

SOFTWARE DESCRIPTION

The Arduino Software (IDE) makes it easy to write code and upload it to the board offline. We recommend it for users with poor or no internet connection. This software can be used with any Arduino board. here are currently two versions of the Arduino IDE, one is the IDE 2.0.0.

7.1 Arduino IDE – Compiler

here are currently two versions of the Arduino IDE, one is the IDE 1.x.x and the other is IDE 2.x. The IDE 2.x is new major release that is faster and even more powerful to the IDE 1.x.x. In addition to a more modern editor and a more responsive interface it includes advanced features to help users with their coding and debugging.

The following steps can guide you with using the offline IDE (you can choose either IDE 1.x.x or IDE 2.x):

1. Download and install the Arduino Software IDE:

- **Arduino IDE 1.x.x** (Windows, Mac OS, Linux, Portable IDE for Windows and Linux, ChromeOS).
- **Arduino IDE 2.x**

2. Connect your Arduino board to your device.

3. Open the Arduino Software (IDE).

The Arduino Integrated Development Environment - or Arduino Software (IDE) - connects to the Arduino boards to upload programs and communicate with them. Programs written using Arduino Software (IDE) are called **sketches**. These sketches are written in the text editor and are saved with the file extension .ino.

Using the offline IDE 1.x.x

The editor contains the four main areas:

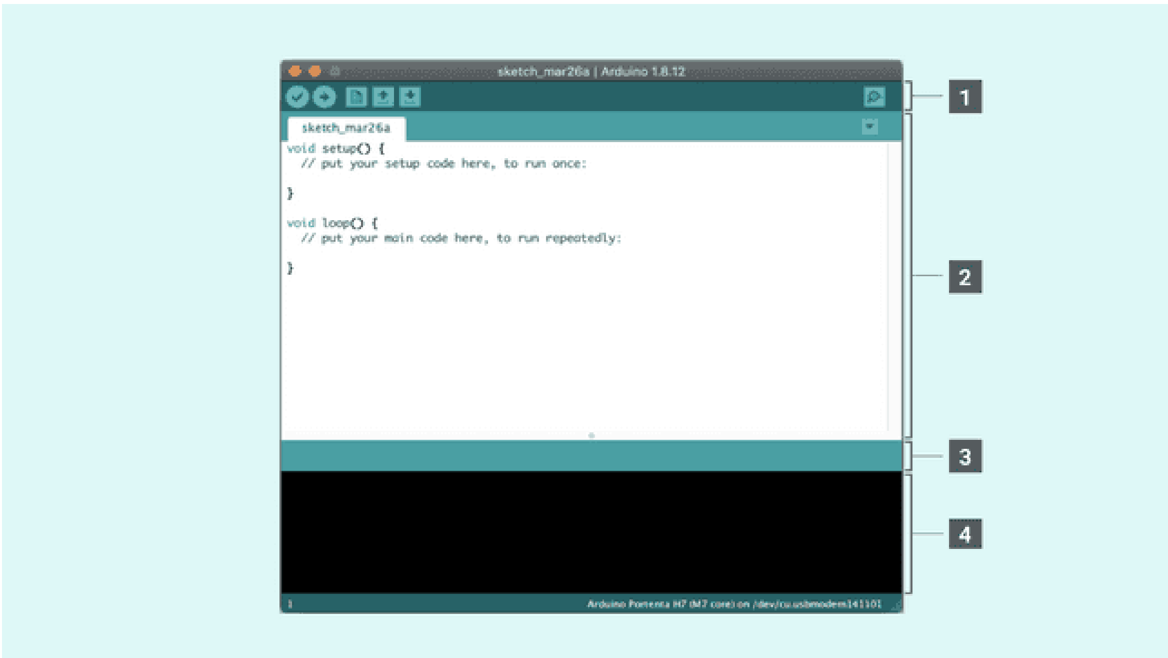
1. A **Toolbar with buttons** for common functions and a series of menus. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

2. The **message area**, gives feedback while saving and exporting and also displays errors.

3. The **text editor** for writing your code.

4. The **text console** displays text output by the Arduino Software (IDE), including complete error messages and other information.

The bottom right-hand corner of the window displays the configured board and serial port.

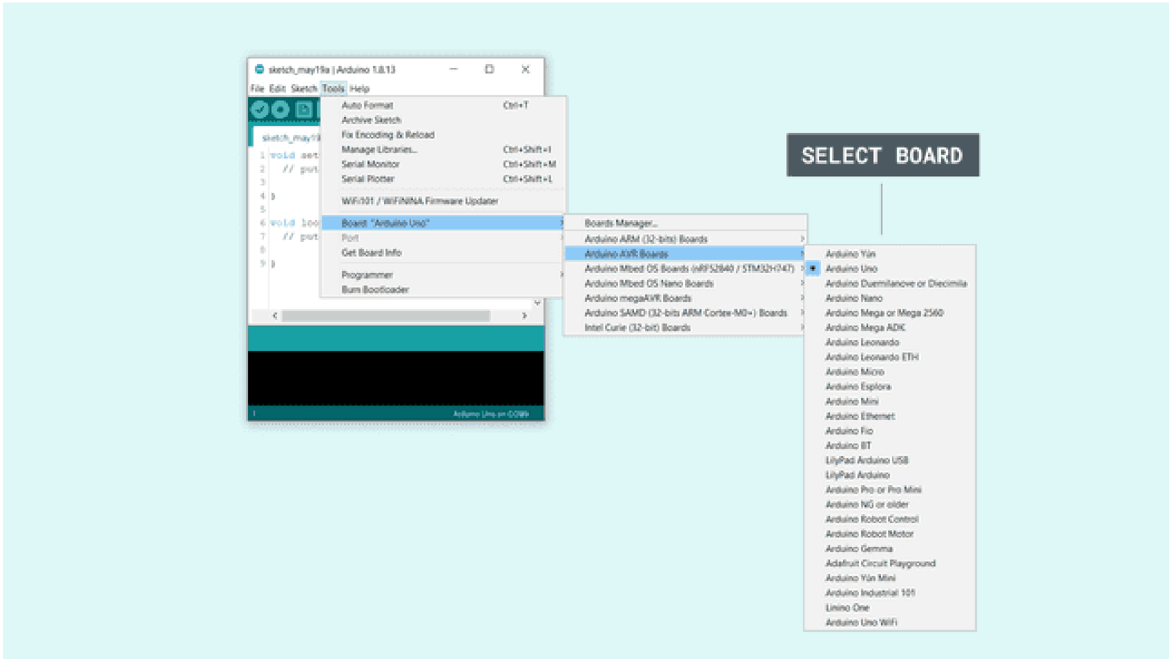


The Arduino Software IDE

Now that you are all set up, **let's try to make your board blink!**

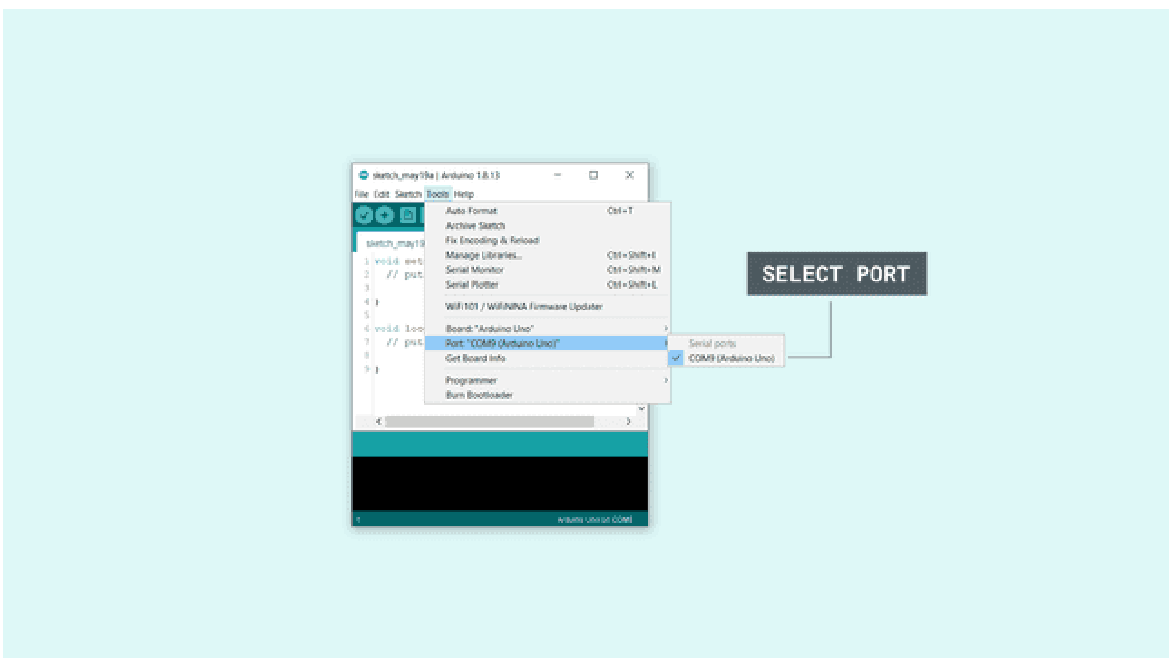
5. **Connect your Arduino** or Genuino board to your computer.

6. Now, you need to **select the right core & board**. This is done by navigating to **Tools** > **Board** > **Arduino AVR Boards** > **Board**. Make sure you select the board that you are using. If you cannot find your board, you can add it from **Tools** > **Board** > **Boards Manager**.



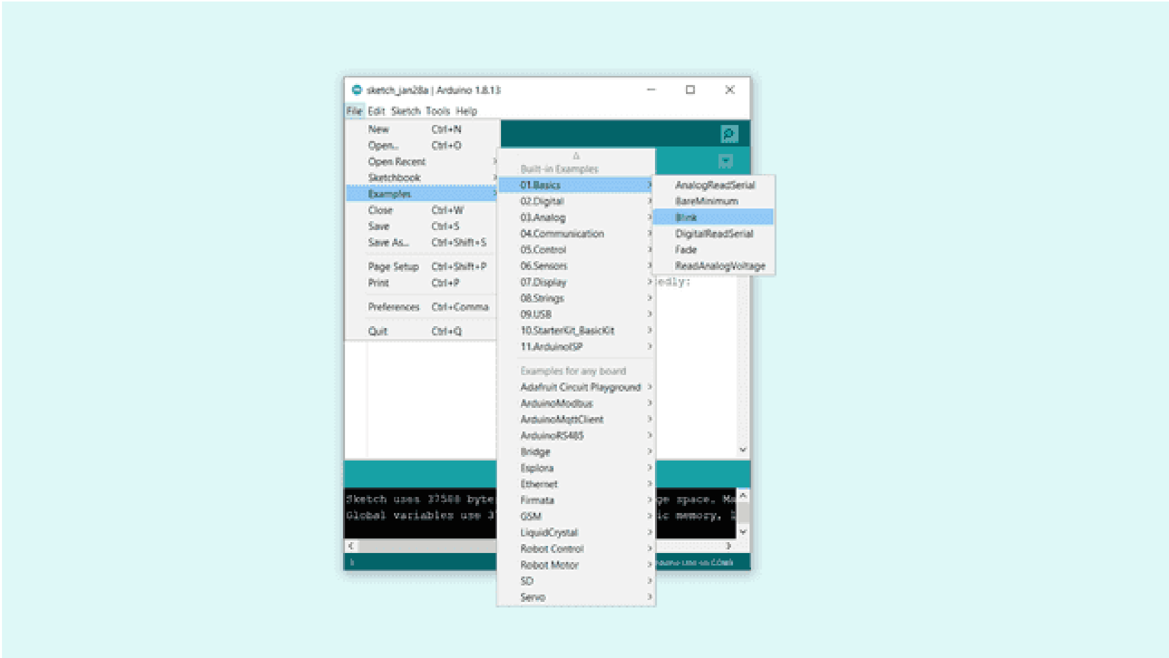
Selecting a board

7. Now, let's make sure that your board is found by the computer, by **selecting the port**. This is simply done by navigating to **Tools** > **Port**, where you select your board from the list.



Selecting the port

8. Let’s **try an example**: navigate to **File** > **Examples** > **01.Basics** > **Blink.**



Opening an example

9. To **upload it to your board**, simply click on the arrow in the top left corner. This process takes a few seconds, and it is important to not disconnect the board during this process. If the upload is successful, the message "Done uploading" will appear in the bottom output area.

10. Once the upload is complete, you should then see on your board the yellow LED with an L next to it start blinking. You can **adjust the speed of blinking** by changing the delay number in the parenthesis to 100, and upload the Blink sketch again. Now the LED should blink much faster.

The editor contains the four main areas:

1. A **toolbar with buttons** for common functions and a series of menus. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, choose your board and port and open the serial monitor.

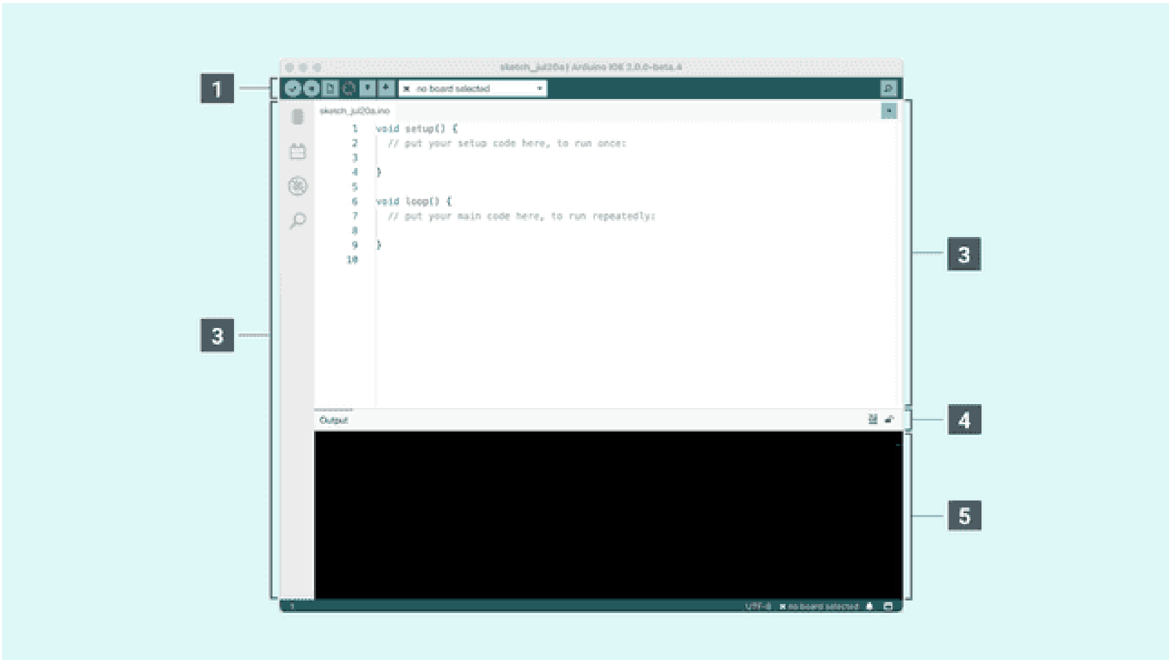
2. The **Sidebar** for regularly used tools. It gives you quick access to board managers, libraries, debugging your board as well as a search and replacement tool.

3. The **text editor** for writing your code.

4. Console controls gives control over the output on the console.

5. The **text console** displays text output by the Arduino Software (IDE), including complete error messages and other information.

The bottom right-hand corner of the window displays the configured board and serial port.

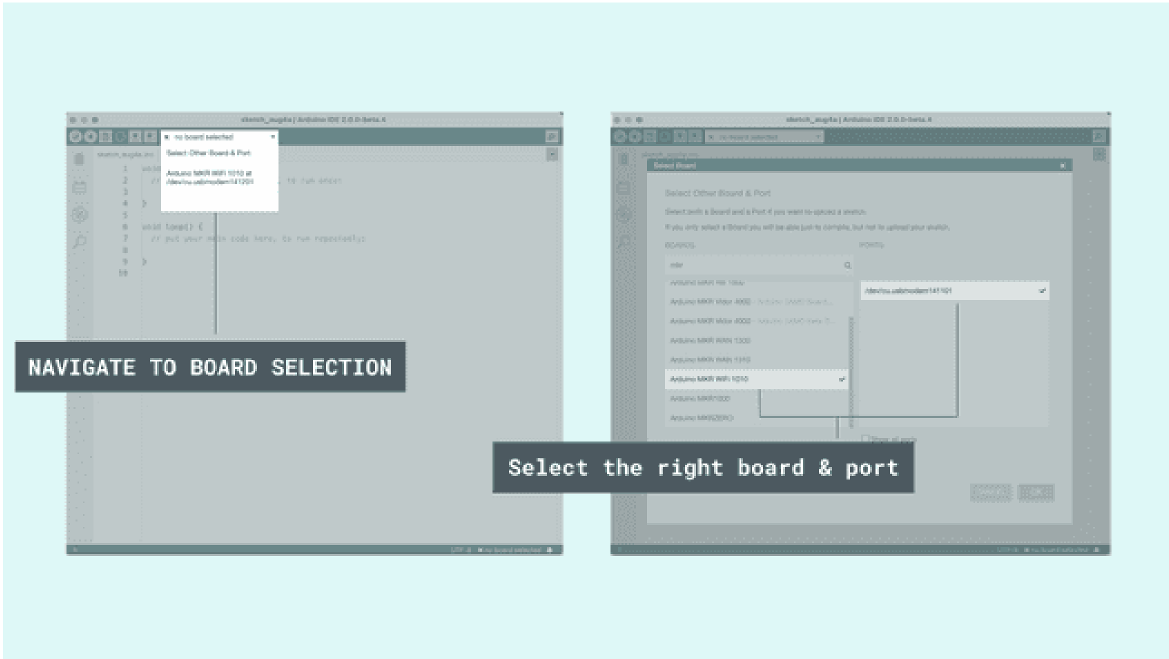


The Arduino Software IDE

Now that you are all set up, **let’s try to make your board blink!**

1. Connect your Arduino or Genuino board to your computer.

2. Now, you need to **select the right board & port**. This is done from the toolbar. Make sure you select the board that you are using. If you cannot find your board, you can add it from the board manager in the sidebar.



Selecting a board & port

3. To **upload it to your board**, simply click on the arrow in the top left corner. This process takes a few seconds, and it is important to not disconnect the board during this process. If the upload is successful, the message "Done uploading" will appear in the bottom output area.

CHAPTER 9

RESULTS

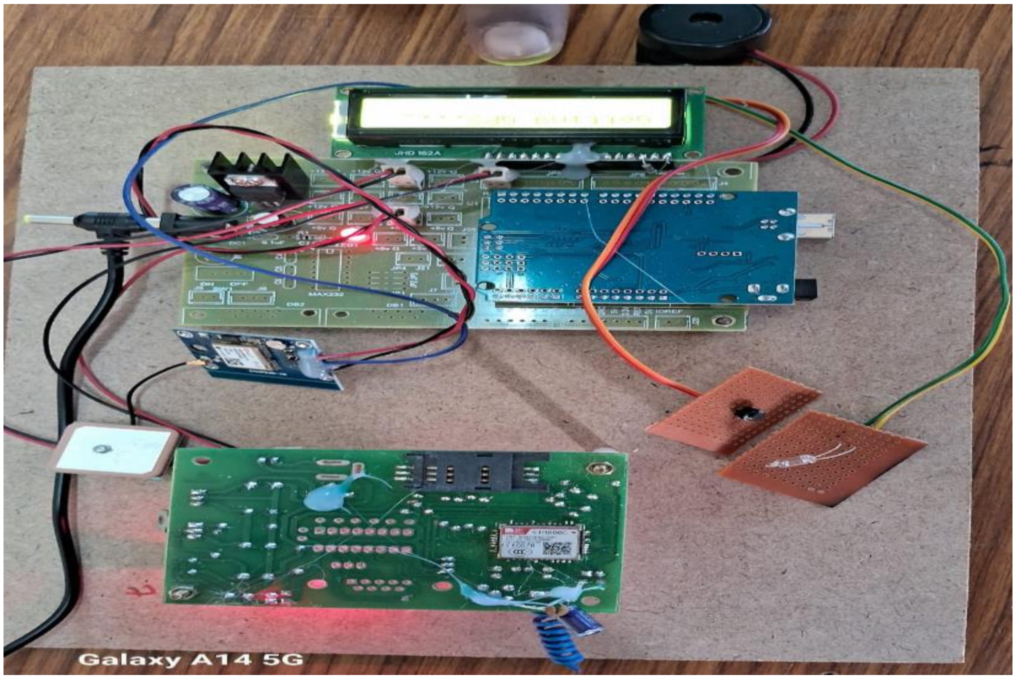


Fig 9.1:Implemened Circuit

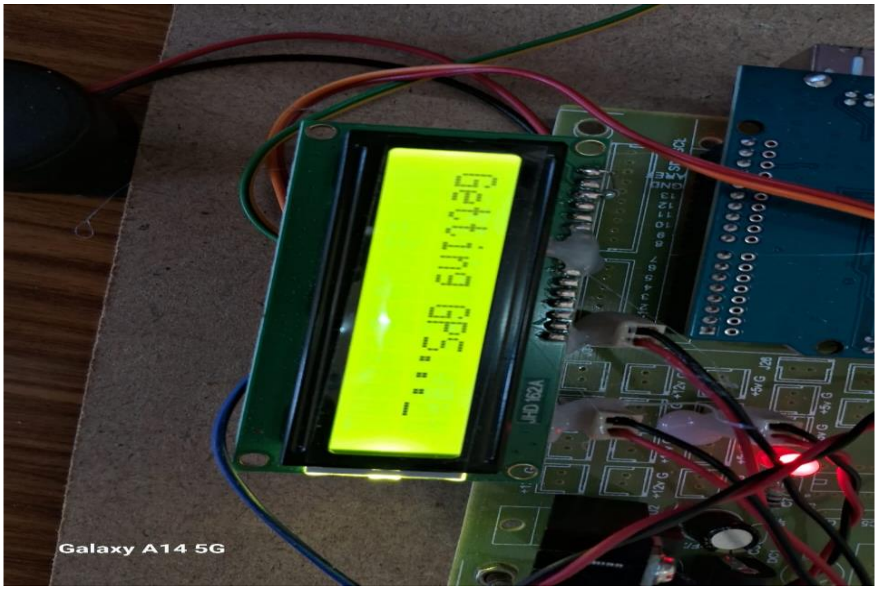


Fig 9.2: Generating GPS

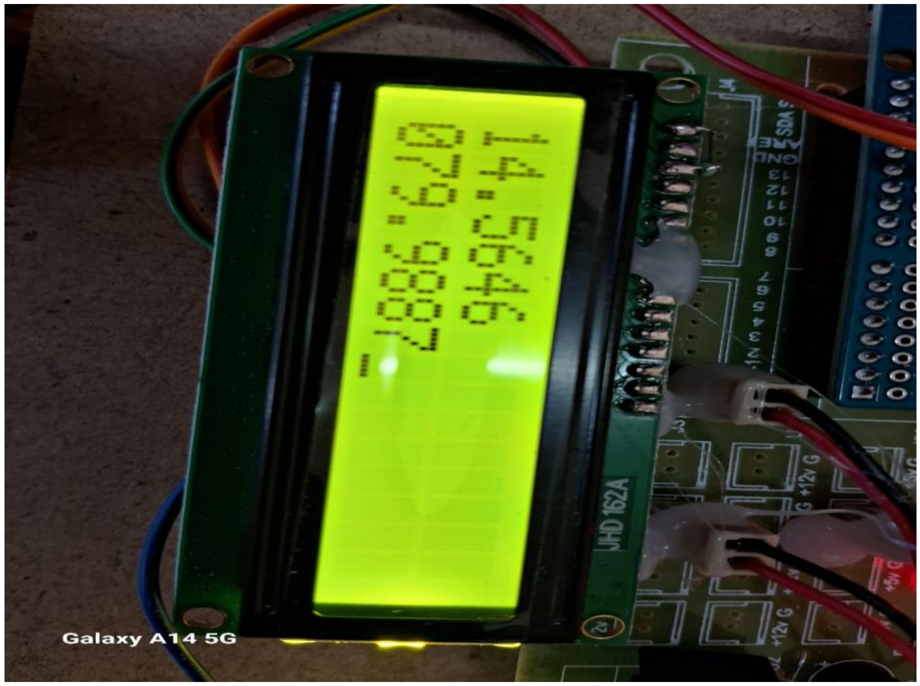


Fig 9.3: Sending SMS to Mobile Phone

The above figures show the hardware equipment of the project. The kit is turned ON by giving the regulated power supply of 12v which is then converted to 5v dc current. The LED is the indication for 5v current. If there is 5v current, then automatically the LED glows. The generated 5v dc current passes to every hardware component in the circuit.

The GPS location is uploaded in the server by using the GPS tracker and IoT module.

The smart vehicle we will completely different from other vehicles. Inthese vehicles we use so many sensors to track the vehicles to protectaccident from break failure, Automatic start on and automatic start offeverything we implement in this vehicle. Now a day’s battery cell wecan implement as a fuel As for Government decided So no sound, nopollution is there in this vehicles otherwise we implement solar cell asfor fuel cell. It can working by remote, but user should be know theworking principle of vehicle. Without knowing we cannot working thevehicles.

CHAPTER 10

CONCLUSION

The vehicle tracking system works mainly by receiving messages from a mobile phone. There is a message command by which we can track the vehicle. And this command is to send an SMS; “TRACK VEHICLE” to the registered SIM card number in the GSM modem. This command initiates the GPS modem and receives the latitude and longitude position and this information will then be sent as SMS to the mobile device. Whenever theft occurs or on demand request of the vehicles location, the device sends a message to the vehicle owner’s mobile.

In conclusion, the implementation of a vehicle accident detection system utilizing GPS and GSM modules offers significant promise in enhancing road safety and emergency response. Through the integration of these technologies, we have developed a solution capable of accurately detecting accidents in real-time and promptly notifying emergency services and relevant stakeholders.

Our evaluation of the system's performance demonstrated its reliability in detecting various types of accidents, including collisions, rollovers, and sudden impacts, with high accuracy and minimal false positives. Furthermore, the utilization of GPS technology enabled precise location tracking, facilitating swift emergency assistance and potentially reducing response times.

The integration of GSM modules facilitated seamless communication between the vehicle and emergency services, ensuring timely transmission of critical information such as the accident location, vehicle identification, and severity assessment. This real-time data exchange has the potential to improve emergency response coordination and save lives by expediting the arrival of medical assistance and rescue teams.

However, while our system shows promise, there are areas for further refinement and enhancement. Future research could focus on optimizing the system's algorithms to improve accuracy and reduce false alarms, as well as exploring additional sensors or technologies to enhance accident detection capabilities in diverse driving conditions.

Moreover, the scalability and compatibility of the system with existing infrastructure and vehicle models should be addressed to ensure widespread adoption and integration into automotive safety standards. Additionally, considerations regarding data privacy, security, and regulatory compliance must be thoroughly addressed to safeguard user information and ensure legal compliance.

In conclusion, the development of a vehicle accident detection system leveraging GPS and GSM modules represents a significant advancement in road safety technology. By harnessing the power of real-time data and communication capabilities, this system has the potential to mitigate the impact of road accidents and save lives. Continued research and development in this field are essential to realizing the full potential of such innovative solutions in promoting safer and more secure transportation systems.

REFERENCES

[1] K.S. Alli, C. Ijeh-Ogboi and S.L. Gbadamosi “Design and Construction of a remotely controlled vehicle anti-theft system via GSM network” International Journal of Education and Research vol. 3 No. 5 May 2015.

[2] S. Boopathi, K. Govindaraju, M. Sangeetha, M. Jagadeeshraja, M. Dhanasu “Real Time Based Smart Vehicle Monitoring and Alert Using GSM” International Journal of Advanced Research in Computer and Communication Engineering Vol. 3, Issue 11, November 2014, ISSN (Online): 2278-1021, ISSN (Print): 2319- 5940.

[3] Kunal Maurya , Mandeep Singh, Neelu Jain, “Real Time Vehicle Tracking System using GSM and GPS Technology- An Anti-theft Tracking System”, International Journal of Electronics and Computer Science Engineering, ISSN 2277-1956/V1N3-1103- 1107.

[4] Lu Mai, Min Zaw O “Design and Construction of Microcontroller Based Wireless Remote Controlled Industrial Electrical Appliances Using ZigBee Technology”, International Journal of Scientific Research Engineering & Technology (IJSRET), ISSN 2278 – 0882, Volume 3 Issue 1, pp. 79 – 84, April 2014.

[5] Daniel. W. Lewis, “Fundamental of embedded software”, prentice hall of India, 2004.

[6] Alan G. Smith, “Introduction to Arduino: A piece of cake!” ISBN: 1463698348, ISBN-13: 978-1463698348, http://www.introtoarduino.com

[7] R.S Gaonkar, ―Microprocessor Architecture Programming and Application‖, Wiley Eastern Limited, New Delhi

[8] Chen H., Chiang, Y. Chang, F.H. Wang, Toward RealTime Precise Point Positioning: Differential GPS Based on IGS Ultra Rapid Product, SICE Annual Conference, The Grand Hitel, Taipei, Taiwan August 18-21, (2010)

[9] Vikram Kulkarni & Viswaprakash Babu, ―embedded smart car security system on face detection‘, special issue of IJCCT, ISSN (Online): 2231-0371, ISSN (Print):0975-7449,volume-3, issue-1

[10] El-Medany,W.;Al-Omary,A.;Al-Hakim,R.;AlIrhayim,S.;Nusaif,M.,"A Cost Effective Real-Time Tracking System Prototype Using Integrated GPS/GPRS Module," Wireless and Mobile Communications (ICWMC), 2010 6th International Conference on,vol.,no.,pp.521,525,20-25 Sept.2010

[11] Fleischer, P.B.; Nelson, A.Y.; Sowah, R.A.; Bremang, A., "Design and development of GPS/GSM based vehicle tracking and alert system for commercial inter-city buses," Adaptive Science & Technology (ICAST), 2012 IEEE 4th International Conference on , vol., no., pp.1,6, 25-27 Oct. 2012

[12] Le-Tien, T.; Vu Phung-The, "Routing and Tracking System for Mobile Vehicles in Large Area," Electronic Design, Test and Application, 2010. DELTA '10. Fifth IEEE International Symposium on , vol., no., pp.297,300, 13-15 Jan. 2010

[13] Abed khan M.E.(Student), , Ravi Mishra, ―GPS – GSM Based Tracking System‖ SSCET, CSVTU, Bhilai, India International Journal of Engineering Trends and Technology- vol.3,no.,pp,161-164, 2012

[14] El-Medany, W.M.; Alomary, A.; Al-Hakim, R.; Al-Irhayim, S.; Nousif, M., "Implementation of GPRS- Based Positioning System Using PIC Microcontroller," Computational Intelligence, Communication Systems and Networks (CICSyN), 2010 Second International Conference on , vol., no., pp.365,368, 28-30 July 2010

[15] Iman M. Almomani, Nour Y. Alkhalil, Enas M. Ahmad, Rania M. Jodeh ―Ubiquitous GPS Vehicle Tracking and Management System‖, IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT) 2011

NAME: PATHAN MOHAMMAD ALI KHAN

ROLL NO: 20JN1A04B7

PH NO: 7013131897

E-MAIL: pathanmohammadalikhan1508@gmail.com

NAME: M. SAI VENKATA KIRAN

ROLL NO: 20JN1A0488

PH NO: 6302873699

E-MAIL: saivenkatakiranmanchu@gmail.com

NAME: M. DHANUSH BABU

ROLL NO: 20JN1A0487

PH NO: 9346647470

E-MAIL: mamudurudhanush6@gmail.com

NAME: P. ADARSH

ROLL NO: 20JN1A04C5

PH NO: 8919735457

E-MAIL: adarshraina8919@gmail.com

BRAIN O VISION

AICTE

APSCHE

CERTIFICATE

OF INTERNSHIP COMPLETION

TO **Date: 28-04-2024**

MAMUDURU DHANUSH BABU
SREE VENKATESWARA COLLEGE OF ENGINEERING

This is to certify that **MAMUDURU DHANUSH BABU,** 20JN1A0487 has successfully completed his/her program with **BrainOvision Solutions Pvt. Ltd.** She/He has worked on **EMBEDDED SYSTEMS & IOT** and was actively & diligently involved in the projects and tasks assigned to her/him. During the span, we found her/him punctual and hardworking person. Her/His feedback and evolution proved that she/he is a quick learner.

Congratulations and Best Wishes.

ROLE : **INTERN**
INTERN ID : **BV24NA349**
START DATE : **08-01-2024**
END DATE : **27-04-2024**

Yours Faithfully

Ganesh Nag Doddi
Founder & CEO
Brainovision Solutions India Pvt Ltd

Dr. Buddha Chandrashekar
Chief Coordinating Officer – AICTE
All India Council for Technical Education