**ANALYSIS OF THE WATER QUALITY MONITORING**

**SYSTEM**

|  |  |  |
| --- | --- | --- |
| **INDEX** | | |
| **Chapter No.** | **TOPIC** | **Page No.** |
| **1**  1.1  1.2  1.3  1.4  1.5 | **INTRODUCTION**  **General**  Existing System  Disadvantage Of Existing System  Proposed System  Advantage Of Proposed System |  |
| **2**  2.1  2.2  2.3  2.3.1  2.3.2  2.3.3  2.3.4  2.3.5  **2.4**  2.4.1  2.4.2  2.4.3  2.4.4  2.4.5  2.4.6  2.4.7  2.4.8  2.4.9  2.4.10  **2.5**  2.5.1  2.5.2  2.5.3  **2.6**  2.6.1  **2.7**  **2.8**  **2.9**  **2.10** | **PROJECT DESCRIPTION**  General Introduction To Embedded System  Block Diagram  Modules  Power Supply  Transformer  Rectifier  Bridge Rectifier  Voltage Regulators  **MICROCONTROLLER**  Arduino Uno  Technical Specifications  Programming  Warnings  Differences With Other Boards  Power  Memory  Circuit Pin Diagram  Communication  Automatic (software) reset    **LIQUID CRYSTAL DISPLAY(LCD)**  Introduction  Pin Diagram  List Of Commands  **IOT**  ESP 8266 WIFI MODULE  **GSM**  **RESET**  **PH SENSOR**  **TURBIDITY SENSOR** |  |
| **3**  3.1  3.2  3.3  3.4  3.5  3.5.1  3.5.2  3.5.3  3.5.4  3.6  3.7  3.8  3.9  3.10  3.11 | **SOFTWARE SPECIFICATIONS**  Arduino IDE  Writing Sketch  File  Edit  Sketch  Tools  Sketch Book  Uploading  Library  Serial Monitor  Preferences  Boards |  |
| **4**  4.1 | **IMPLEMENTATION**  Schematic Design |  |
| **5**  5.1 | **SIMULATION AND DESIGN**  Output Screen Shots |  |
| **6** | **CONCLUSION** |  |
| **7** | **REFERENCES** |  |

**CHAPTER 1**

**INTRODUCTION**

**1.1GENERAL**

Now a days many people are suffering from dangerous diseases which are caused due to impure water. In our project we are doing analysis for water quality monitoring system, it gives data about the quality of water, on a webpage. The quality of water is determined using various sensors like PH sensor and turbidity sensor, connected to the Arduino family microcontroller. The Arduino software is written in embedded C and GSM module is connected to the Arduino. The data will be transferred constantly from the remote sensor organize through microcontroller and Wi-Fi. Wi-Fi module is used to send data to the webpage via internet which is connected to the microcontroller. The total data regarding the purity of water is displayed in the webpage and is analyzed in the form of graph, pie chart and values are given in the table. We transfer this information to cloud and clients can get to this information through web page application, client from anyplace can screen the data whenever.

**1.2 EXISTING SYSTEM**

The current Water Quality observing framework include human towards looking at the water Quality, Testing. At present many measures are taken and innovative development has been introduced in water quality checking. These are finished by utilizing automated fish, laser bar and advanced camera. Likewise enquiry about the fish has been done by utilizing remote sensors.

**1.3 DISADVANTAGES OF EXISTING SYSTEM**

**1.4 PROPOSED SYSTEM**

In proposed system, the water quality will be checked by the Ph and turbidity sensors which are connected to the Node MCU microcontroller. The data is collected and is sent to the cloud for storage. The values are stored in a tabular format. We can get alert message notification to the registered mobile number through sms if the sensor reaches above the limit i.e. Ph<7 and turbidity>60 using GSM. The values that are stored in a tabular form are analyzed in the form of line graph and pie charts.

**1.5 ADVANTAGES OF PROPOSED SYSTEM**

**CHAPTER 2**

**PROJECT DESCRIPTION**

**2.1 GENERAL INTRODUCTION TO EMBEDDED SYSTEM**

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

An embedded system is not always a separate block very often it is physically built in to the device it is controlling. The software written for embedded systems is often called firmware, and is stored in read only memory or flash convector chips rather than a disk drive. It often runs with limited computer hardware resources: small or no keyboard, screen and little memory.

To perform any application in the embedded system we require microprocessor and microcontroller. In the microprocessor an external memory is connected which increases the size of the microprocessor and multiple operations are being performed by the microprocessor but whereas in the microprocessor the memory is inbuilt and also we can use this controller only for the specific applications where the speed is increased so most probably microcontrollers are used in the different applications in the embedded systems rather than microprocessor.

An embedded system can be defined as a computing device that does a specific focused job. Appliances such as the air-conditioner, VCD player, DVD player, printer, fax machine, mobile phone etc. are examples of embedded systems. Each of these appliances will have a processor and special hardware to meet the specific requirement of the application along with the embedded software that is executed by the processor for meeting that specific requirement. The embedded software is also called “firm ware”. The desktop/laptop computer is a general purpose computer. You can use it for a variety of applications such as playing games, *word* processing, accounting, software development and so on. In contrast, the software in the embedded systems is always fixed listed below:

· Embedded systems do a very specific task; they cannot be programmed to do different things. . Embedded systems have very limited resources, particularly the memory. Generally, they do not have secondary storage devices such as the CDROM or the floppy disk. Embedded systems have to work against some deadlines. A specific job has to be completed within a specific time. In some embedded systems, called real-time systems, the deadlines are stringent. Missing a deadline may cause a catastrophe-loss of life or damage to property. Embedded systems are constrained for power. As many embedded systems operate through a battery, the power consumption has to be very low.

· Some embedded systems have to operate in extreme environmental conditions such as very high temperatures and humidity.

**APPLICATION AREAS**

Nearly 99 per cent of the processors manufactured end up in embedded systems. The embedded system market is one of the highest growth areas as these systems are used in very market segment- consumer electronics, office automation, industrial automation, biomedical engineering, wireless communication, data communication, telecommunications, transportation, military and so on.

**CONSUMER APPLIANCES**:

At home we use a number of embedded systems which include digital camera, digital diary, DVD player, electronic toys, microwave oven, remote controls for TV and air-conditioner, VCO player, video game consoles, video recorders etc. Today’s high-tech car has about 20 embedded systems for transmission control, engine spark control, air-conditioning, navigation etc. Even wristwatches are now becoming embedded systems. The palmtops are powerful embedded systems using which we can carry out many general-purpose tasks such as playing games and word processing.

**OFFICE AUTOMATION:**

The office automation products using embedded systems are copying machine, fax machine, key telephone, modem, printer, scanner etc.

**INDUSTRIAL AUTOMATION**:

Today a lot of industries use embedded systems for process control. These include pharmaceutical, cement, sugar, oil exploration, nuclear energy, electricity generation and transmission. The embedded systems for industrial use are designed to carry out specific tasks such as monitoring the temperature, pressure, humidity, voltage, current etc., and then take appropriate action based on the monitored levels to control other devices or to send information to a centralized monitoring station. In hazardous industrial environment, where human presence has to be avoided, robots are used, which are programmed to do specific jobs. The robots are now becoming very powerful and carry out many interesting and complicated tasks such as hardware assembly.

**MEDICAL ELECTRONICS**:

Almost every medical equipment in the hospital is an embedded system. These equipment include diagnostic aids such as ECG, EEG, blood pressure measuring devices, X-ray scanners; equipment used in blood analysis, radiation, colonoscopy, endoscopy etc. Developments in medical electronics have paved way for more accurate diagnosis of diseases.

**COMPUTER NETWORKING**:

Computer networking products such as bridges, routers, Integrated Services Digital Networks (ISDN), Asynchronous Transfer Mode (ATM), X.25 and frame relay switches are embedded systems which implement the necessary data communication protocols. For example, a router interconnects two networks. The two networks may be running different protocol stacks. The router’s function is to obtain the data packets from incoming pores, analyze the packets and send them towards the destination after doing necessary protocol conversion. Most networking equipment’s, other than the end systems (desktop computers) we use to access the networks, are embedded systems.

**TELECOMMUNICATIONS**:

In the field of telecommunications, the embedded systems can be categorized as subscriber terminals and network equipment. The subscriber terminals such as key telephones, ISDN phones, terminal adapters, web cameras are embedded systems. The network equipment includes multiplexers, multiple access systems, Packet Assemblers Dissemblers (PADs), sate11ite modems etc. IP phone, IP gateway, IP gatekeeper etc. are the latest embedded systems that provide very low-cost voice communication over the Internet.

**WIRELESS TECHNOLOGIES**:

Advances in mobile communications are paving way for many interesting applications using embedded systems. The mobile phone is one of the marvels of the last decade of the 20’h century. It is a very powerful embedded system that provides voice communication while we are on the move. The Personal Digital Assistants and the palmtops can now be used to access multimedia service over the Internet. Mobile communication infrastructure such as base station controllers, mobile switching centers are also powerful embedded systems.

**INSEMINATION:**

Testing and measurement are the fundamental requirements in all scientific and engineering activities. The measuring equipment we use in laboratories to measure parameters such as weight, temperature, pressure, humidity, voltage, current etc. are all embedded systems. Test equipment such as oscilloscope, spectrum analyzer, logic analyzer, protocol analyzer, radio communication test set etc. are embedded systems built around powerful processors. Thank to miniaturization, the test and measuring equipment are now becoming portable facilitating easy testing and measurement in the field by field-personnel.

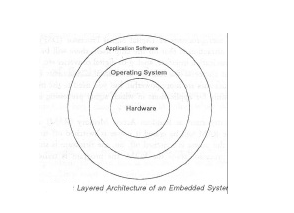
**SECURITY:**

Security of persons and information has always been a major issue. We need to protect our homes and offices; and also the information we transmit and store. Developing embedded systems for security applications is one of the most lucrative businesses nowadays. Security devices at homes, offices, airports etc. for authentication and verification are embedded systems. Encryption devices are nearly 99 per cent of the processors that are manufactured end up in~ embedded systems. Embedded systems find applications in every industrial segment- consumer electronics, transportation, avionics, biomedical engineering, manufacturing, process control and industrial automation, data communication, telecommunication, defense, security etc. Used to encrypt the data/voice being transmitted on communication links such as telephone lines. Biometric systems using fingerprint and face recognition are now being extensively used for user authentication in banking applications as well as for access control in high security buildings.

**FINANCE:**

Financial dealing through cash and cheques are now slowly paving way for transactions using smart cards and ATM (Automatic Teller Machine, also expanded as Any Time Money) machines. Smart card, of the size of a credit card, has a small micro-controller and memory; and it interacts with the smart card reader! ATM machine and acts as an electronic wallet. Smart card technology has the capability of ushering in a cashless society. Well, the list goes on. It is no exaggeration to say that eyes wherever you go, you can see, or at least feel, the work of an embedded system.

**OVERVIEW OF EMBEDDED SYSTEM ARCHITECTURE**

Every embedded system consists of custom-built hardware built around a Central Processing Unit (CPU). This hardware also contains memory chips onto which the software is loaded. The software residing on the memory chip is also called the ‘firmware’. The embedded system architecture can be represented as a layered architecture as shown in Fig. The operating system runs above the hardware, and the application software runs above the operating system. The same architecture is applicable to any computer including a desktop computer. However, there are significant differences. It is not compulsory to have an operating system in every embedded system. For small appliances such as remote control units, air conditioners, toys etc., there is no need *for* an operating system and you can write only the software specific to that application. For applications involving complex processing, it is advisable to have an operating system. In such a case, you need to integrate the application software with the operating system and then transfer the entire software on to the memory chip. Once the software is transferred to the memory chip, the software will continue to run *for* a long time you don’t need to reload new software.

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

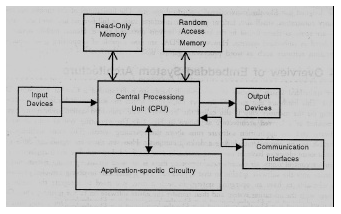
· Central Processing Unit (CPU)

· Memory (Read-only Memory and Random Access Memory)

· Input Devices

· Output devices

· Communication interfaces

· Application-specific circuitry

**CENTRAL PROCESSING UNIT (CPU):**

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

**MEMORY:**

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

**INPUT DEVICES**:

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

**OUTPUT DEVICES**:

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

**2.2 BLOCK DIAGRAM**

Micro

Controller

PH sensor

Turbidity sensor

Power

Supply

LCD

IOT

Reset

GSM

**2.3MODULES**

**2.3.1. POWER SUPPLY**

The power supply section is the section which provide +5V for the components to work. IC LM7805 is used for providing a constant power of +5V.

The ac voltage, typically 220V, is connected to a transformer, which steps down that ac voltage down to the level of the desired dc output. A diode rectifier then provides a full-wave rectified voltage that is initially filtered by a simple capacitor filter to produce a dc voltage. This resulting dc voltage usually has some ripple or ac voltage variation.

A regulator circuit removes the ripples and also retains the same dc value even if the input dc voltage varies, or the load connected to the output dc voltage changes. This voltage regulation is usually obtained using one of the popular voltage regulator IC units.

****

Figure. Block Diagram of Power Supply

**2.3.2 TRANSFORMER**

Transformers convert AC electricity from one voltage to another with little loss of power. Transformers work only with AC and this is one of the reasons why mains electricity is AC.

Step-up transformers increase voltage, step-down transformers reduce voltage. Most power supplies use a step-down transformer to reduce the dangerously high mains voltage (230V in India) to a safer low voltage.

The input coil is called the primary and the output coil is called the secondary. There is no electrical connection between the two coils; instead they are linked by an alternating magnetic field created in the soft-iron core of the transformer. Transformers waste very little power so the power out is (almost) equal to the power in. Note that as voltage is stepped down current is stepped up. The transformer will step down the power supply voltage (0-230V) to (0- 6V) level. Then the secondary of the potential transformer will be connected to the bridge rectifier, which is constructed with the help of PN junction diodes. The advantages of using bridge rectifier are it will give peak voltage output as DC.

**2.3.3 RECTIFIER**

There are several ways of connecting diodes to make a rectifier to convert AC to DC. The bridge rectifier is the most important and it produces full-wave varying DC. A full-wave rectifier can also be made from just two diodes if a centre-tap transformer is used, but this method is rarely used now that diodes are cheaper. A single diode can be used as a rectifier but it only uses the positive (+) parts of the AC wave to produce half-wave varying DC

**2.3.4 BRIDGE RECTIFIER**

When four diodes are connected as shown in figure, the circuit is called as bridge rectifier. The input to the circuit is applied to the diagonally opposite corners of the network, and the output is taken from the remaining two corners. Let us assume that the transformer is working properly and there is a positive potential, at point A and a negative potential at point B. the positive potential at point A will forward bias D3 and reverse bias D4.



Figure. Bridge Rectifier

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

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

i. The main advantage of this bridge circuit is that it does not require a special centre tapped transformer, thereby reducing its size and cost.

ii. The single secondary winding is connected to one side of the diode bridge network and the load to the other side as shown below.

iii. The result is still a pulsating direct current but with double the frequency.

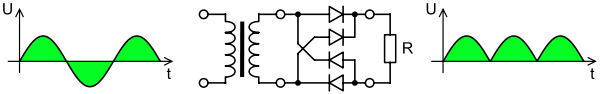


Figure: Output Waveform of DC

**SMOOTHING**

Smoothing is performed by a large value electrolytic capacitor connected across the DC supply to act as a reservoir, supplying current to the output when the varying DC voltage from the rectifier is falling. The capacitor charges quickly near the peak of the varying DC, and then discharges as it supplies current to the output.

**2.3.5 VOLTAGE REGULATORS**

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

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

The series 78 regulators provide fixed positive regulated voltages from 5 to 24 volts. Similarly, the series 79 regulators provide fixed negative regulated voltages from 5 to 24 volts. Voltage regulator ICs are available with fixed (typically 5, 12 and 15V) or variable output voltages. They are also rated by the maximum current they can pass. Negative voltage regulators are available, mainly for use in dual supplies. Most regulators include some automatic protection from excessive current ('overload protection') and overheating ('thermal protection').

Many of the fixed voltage regulator ICs has 3 leads and look like power transistors, such as the 7805 +5V 1Amp regulator. They include a hole for attaching a heat sink if necessary.

****

Figure. Regulator

****

Figure. Circuit Diagram of Power Supply

**2.4. MICROCONTROLLER**

A Microcontroller (or MCU) is a [computer](file:///D:\\wiki\\Computer)-on-a-[chip](file:///D:\\wiki\\Integrated_circuit) used to control [electronic](file:///D:\\wiki\\Electronics) [devices](file:///D:\\wiki\\Devices). It is a type of [microprocessor](file:///D:\\wiki\\Microprocessor) emphasizing self-sufficiency and cost-effectiveness, in contrast to a general-purpose microprocessor (the kind used in a [PC](file:///D:\\wiki\\Personal_computer)). A typical microcontroller contains all the [memory](file:///D:\\wiki\\Memory) and [interfaces](file:///D:\\wiki\\Interface_(computer_science)) needed for a simple application, whereas a general purpose microprocessor requires additional chips to provide these functions.

A microcontroller is a single [integrated circuit](file:///D:\\wiki\\Integrated_circuit) with the following key features:

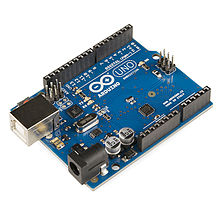
1. [central processing unit](file:///D:\\wiki\\Central_processing_unit) - ranging from small and simple 8-[bit](file:///D:\\wiki\\Bit) processors to sophisticated 32- or 64-bit processors
2. [input/output](file:///D:\\wiki\\Input\\output) [interfaces](file:///D:\\wiki\\Network_interface) such as [serial ports](file:///D:\\wiki\\Serial_port)
3. [RAM](file:///D:\\wiki\\RAM) for data storage
4. [ROM](file:///D:\\wiki\\Read-only_Memory), [EEPROM](file:///D:\\wiki\\EEPROM) or [Flash memory](file:///D:\\wiki\\Flash_memory) for [program](file:///D:\\wiki\\Computer_program) storage
5. [clock generator](file:///D:\\wiki\\Clock_generator) - often an oscillator for a quartz timing crystal, resonator or [RC](file:///D:\\wiki\\RC_circuit) circuit

Microcontrollers are inside many kinds of [electronic equipment](file:///D:\\wiki\\Electronic_equipment) (see [embedded system](file:///D:\\wiki\\Embedded_system)). They are the vast majority of all processor chips sold. Over 50% are "simple" controllers, and another 20% are more specialized [digital signal processors (DSPs)](file:///D:\\wiki\\Digital_signal_processor) (ref?). A typical home in a [developed country](file:///D:\\wiki\\Developed_country) is likely to have only one or two general-purpose microprocessors but somewhere between one and two dozen microcontrollers. A typical mid-range vehicle has as many as 50 or more microcontrollers. They can also be found in almost any [electrical](file:///D:\\wiki\\Electrical) [device](file:///D:\\wiki\\Tool): [washing machines](file:///D:\\wiki\\Washing_machine), [microwave ovens](file:///D:\\wiki\\Microwave_oven), [telephones](file:///D:\\wiki\\Telephone) etc.

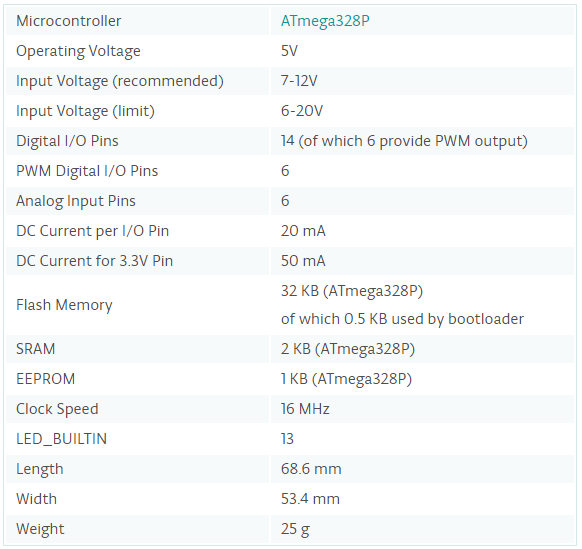
**2.4.1 Arduino UNO**

Arduino/genuino uno is a microcontroller board based on the atmega328p (datasheet). It has 14 digital input/output pins (of which 6 can be used as pwm outputs), 6 analog inputs, a 16 mhz quartz crystal, a usb connection, a power jack, an icsp header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a usb cable or power it with a ac-to-dc adapter or battery to get started.. You can tinker with your uno without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.

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



**2.4.2 TECHNICAL SPECIFICATIONS**



**2.4.3 Programming**

The arduino/genuino uno can be programmed with the (arduino software (ide)). Select "arduino/genuino uno from the tools > board menu (according to the microcontroller on your board). For details, see the reference and tutorials.

The atmega328 on the arduino/genuino uno comes preprogrammed with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original stk500 protocol (reference, c header files).

You can also bypass the bootloader and program the microcontroller through the icsp (in-circuit serial programming) header using arduino isp or similar; see these instructions for details.

The atmega16u2 (or 8u2 in the rev1 and rev2 boards) firmware source code is available in the arduino repository. The atmega16u2/8u2 is loaded with a dfu bootloader, which can be activated by:

On rev1 boards: connecting the solder jumper on the back of the board (near the map of italy) and then reseting the 8u2.

On rev2 or later boards: there is a resistor that pulling the 8u2/16u2 hwb line to ground, making it easier to put into dfu mode.

You can then use atmel's flip software (windows) or the dfu programmer (mac os x and linux) to load a new firmware. Or you can use the isp header with an external programmer (overwriting the dfu bootloader). See this user-contributed tutorial for more information.

**2.4.4 Warnings**

The arduino/genuino uno has a resettable polyfuse that protects your computer's usb ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 ma is applied to the usb port, the fuse will automatically break the connection until the short or overload is removed.

**2.4.5 Differences with other boards**

The uno differs from all preceding boards in that it does not use the ftdi usb-to-serial driver chip. Instead, it features the atmega16u2 (atmega8u2 up to version r2) programmed as a usb-to-serial converter.

**2.4.6 Power**

The arduino/genuino uno board can be powered via the usb connection or with an external power supply. The power source is selected automatically.

External (non-usb) power can come either from an ac-to-dc adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the gnd and vin pin headers of the power connector.

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

The power pins are as follows:

Vin. The input voltage to the arduino/genuino board when it's using an external power source (as opposed to 5 volts from the usb connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.

5v.this pin outputs a regulated 5v from the regulator on the board. The board can be supplied with power either from the dc power jack (7 - 12v), the usb connector (5v), or the vin pin of the board (7-12v). Supplying voltage via the 5v or 3.3v pins bypasses the regulator, and can damage your board. We don't advise it.

3v3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 ma.

Gnd. Ground pins.

Ioref. This pin on the arduino/genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the ioref pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5v or 3.3v.

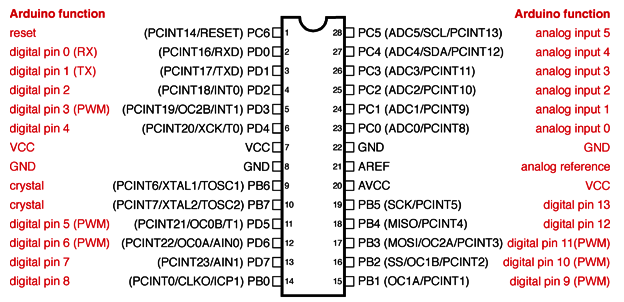
**2.4.7 Memory**

The atmega328 has 32 kb (with 0.5 kb occupied by the bootloader). It also has 2 kb of sram and 1 kb of eeprom (which can be read and written with the eeprom library).

Input and output

See the mapping between arduino pins and atmega328p ports. The mapping for the atmega8, 168, and 328 is identical.

**2.4.8 CIRCUIT PIN DIAGRAM**



Each of the 14 digital pins on the uno can be used as an input or output, using pinmode(), digitalwrite (), and digitalread () functions. They operate at 5 volts. Each pin can provide or receive 20 ma as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40ma is the value that must not be exceeded on any i/o pin to avoid permanent damage to the microcontroller.

In addition, some pins have specialized functions:

Serial: 0 (rx) and 1 (tx). Used to receive (rx) and transmit (tx) ttl serial data. These pins are connected to the corresponding pins of the atmega8u2 usb-to-ttl serial chip.

External interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachinterrupt() function for details.

Pwm: 3, 5, 6, 9, 10, and 11. Provide 8-bit pwm output with the analog write() function.

Spi: 10 (ss), 11 (mosi), 12 (miso), 13 (sck). These pins support spi communication using the spi library.

Led: 13. There is a built-in led driven by digital pin 13. When the pin is high value, the led is on, when the pin is low, it's off.

Twi: a4 or sda pin and a5 or scl pin. Support twi communication using the wire library.

The uno has 6 analog inputs, labeled a0 through a5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the aref pin and the analogreference() function.

There are a couple of other pins on the board:

Aref.Reference voltage for the analog inputs. Used with analogreference().

Reset. Bring this line low to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

**2.4.9 Communication**

Arduino/genuino uno has a number of facilities for communicating with a computer, another arduino/genuino board, or other microcontrollers. The atmega328 provides uartttl (5v) serial communication, which is available on digital pins 0 (rx) and 1 (tx). An atmega16u2 on the board channels this serial communication over usb and appears as a virtual com port to software on the computer. The 16u2 firmware uses the standard usb com drivers, and no external driver is needed. However, on windows, a .inf file is required. The arduino software (ide) includes a serial monitor which allows simple textual data to be sent to and from the board. The rx and txleds on the board will flash when data is being transmitted via the usb-to-serial chip and usb connection to the computer (but not for serial communication on pins 0 and 1).

A softwareserial library allows serial communication on any of the uno's digital pins.

The atmega328 also supports i2c (twi) and spi communication. The arduino software (ide) includes a wire library to simplify use of the i2c bus; see the documentation for details. For spi communication, use the spi library.

**2.4.10 Automatic (software) reset**

Rather than requiring a physical press of the reset button before an upload, the arduino/genuino uno board is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (dtr) of the atmega8u2/16u2 is connected to the reset line of the atmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The arduino software (ide) uses this capability to allow you to upload code by simply pressing the upload button in the interface toolbar. This means that the bootloader can have a shorter timeout, as the lowering of dtr can be well-coordinated with the start of the upload.

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

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

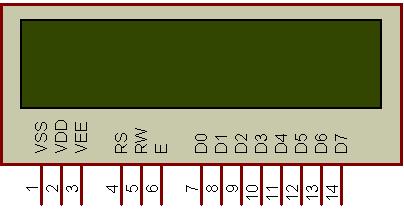
**2.5 LIQUID CRYSTAL DISPLAY**

**2.5.1 INTRODUCTION**

LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over [seven segments](http://www.engineersgarage.com/content/seven-segment-display) and other multi segment [LED](http://www.engineersgarage.com/content/led)s. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special & even [custom characters](http://www.engineersgarage.com/microcontroller/8051projects/create-custom-characters-LCD-AT89C51) (unlike in seven segments), [animations](http://www.engineersgarage.com/microcontroller/8051projects/display-custom-animations-LCD-AT89C51) and so on.

A **16x2 LCD** means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data.

The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD.



**Fig: 16x2 LCD**

**INTRODUCTION**

The most commonly used Character based LCDs are based on Hitachi's HD44780 controller or other which are compatible with HD44580.

**2.5.2 PIN DESCRIPTION**

Most LCDs with 1 controller has 14 Pins and LCDs with 2 controller has 16 Pins (two pins are extra in both for back-light LED connections). Pin description is shown in the table below.

**PIN CONFIGURATION TABLE FOR A 16X2 LCD CHARACTER DISPLAY:-**

|  |  |  |
| --- | --- | --- |
| **Pin Number** | **Symbol** | **Function** |
| **1** | Vss | Ground Terminal |
| **2** | Vcc | Positive Supply |
| **3** | Vdd | Contrast adjustment |
| **4** | RS | Register Select; 0→Instruction Register, 1→Data Register |
| **5** | R/W | Read/write Signal; 1→Read, 0→ Write |
| **6** | E | Enable; Falling edge |
| **7** | DB0 | Bi-directional data bus, data transfer is performed once, thru DB0 to DB7, in the case of interface data length is 8-bits; and twice, through DB4 to DB7 in the case of interface data length is 4-bits. Upper four bits first then lower four bits. |
| **8** | DB1 |
| **9** | DB2 |
| **10** | DB3 |
| **11** | DB4 |
| **12** | DB5 |
| **13** | DB6 |
| **14** | DB7 |
| **15** | LED-(K) | Back light LED cathode terminal |
| **16** | LED+(A) | Back Light LED anode terminal |

Fig: Pin Configuration of LCD

**DATA/SIGNALS/EXECUTION OF LCD**

LCD accepts two types of signals, one is data, and another is control. These signals are recognized by the LCD module from status of the RS pin. Now data can be read also from the LCD display, by pulling the R/W pin high. As soon as the E pin is pulsed, LCD display reads data at the falling edge of the pulse and executes it, same for the case of transmission.

          LCD display takes a time of 39-43µS to place a character or execute a command. Except for clearing display and to seek cursor to home position it takes 1.53ms to 1.64ms. Any attempt to send any data before this interval may lead to failure to read data or execution of the current data in some devices. Some devices compensate the speed by storing the incoming data to some temporary registers.

**INSTRUCTION REGISTER (IR) AND DATA REGISTER (DR)**

There are two 8-bit registers in HD44780 controller Instruction and Data register. Instruction register corresponds to the register where you send commands to LCD e.g LCD shift command, LCD clear, LCD address etc. and Data register is used for storing data which is to be displayed on LCD. when send the enable signal of the LCD is asserted, the data on the pins is latched in to the data register and data is then moved automatically to the DDRAM and hence is displayed on the LCD. Data Register is not only used for sending data to DDRAM but also for CGRAM, the address where you want to send the data, is decided by the instruction you send to LCD. We will discuss more on LCD instruction set further in this tutorial.

**2.5.3 COMMANDS AND INSTRUCTION SET**

Only the instruction register (IR) and the data register (DR) of the LCD can be controlled by the MCU. Before starting the internal operation of the LCD, control information is temporarily stored into these registers to allow interfacing with various MCUs, which operate at different speeds, or various peripheral control devices. The internal operation of the LCD is determined by signals sent from the MCU. These signals, which include register selection signal (RS), read/write signal (R/W), and the data bus (DB0 to DB7), make up the LCD instructions (Table 3). There are four categories of instructions that:

* Designate LCD functions, such as display format, data length, etc.
* Set internal RAM addresses
* Perform data transfer with internal RAM
* Perform miscellaneous functions

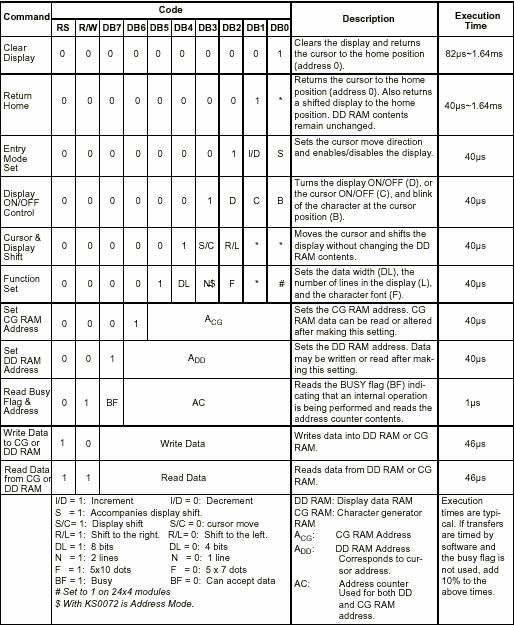


Fig: Code for LCD

Although looking at the table you can make your own commands and test them. Below is a brief list of useful commands which are used frequently while working on the LCD.

**LIST OF COMMANDS**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Instruction** | **Hex** | **Decimal** |
| 1 | Function Set: 8-bit, 1 Line, 5x7 Dots | 0x30 | 48 |
| 2 | Function Set: 8-bit, 2 Line, 5x7 Dots | 0x38 | 56 |
| 3 | Function Set: 4-bit, 1 Line, 5x7 Dots | 0x20 | 32 |
| 4 | Function Set: 4-bit, 2 Line, 5x7 Dots | 0x28 | 40 |
| 5 | Entry Mode | 0x06 | 6 |
| 6 | Display off Cursor off (clearing display without clearing DDRAM content) | 0x08 | 8 |
| 7 | Display on Cursor on | 0x0E | 14 |
| 8 | Display on Cursor off | 0x0C | 12 |
| 9 | Display on Cursor blinking | 0x0F | 15 |
| 10 | Shift entire display left | 0x18 | 24 |
| 12 | Shift entire display right | 0x1C | 30 |
| 13 | Move cursor left by one character | 0x10 | 16 |
| 14 | Move cursor right by one character | 0x14 | 20 |
| 15 | Clear Display (also clear DDRAM content) | 0x01 | 1 |
| 16 | Set DDRAM address or cursor position on display | 0x80+add\* | 128+add\* |
| 17 | Set CGRAM address or set pointer to CGRAM location | 0x40+add\*\* | 64+add\*\* |

**Table: Frequently used commands and instructions for LCD**

\* DDRAM address given in LCD basics section see Figure 2,3,4  
\*\* CGRAM address from 0x00 to 0x3F, 0x00 to 0x07 for char1 and so on.

**LIQUID CRYSTAL DISPLAYS INTERFACING WITH CONTROLLER**

The LCD standard requires 3 control lines and 8 I/O lines for the data bus.

• **8 data pins D7:D0**

Bi-directional data/command pins.  
Alphanumeric characters are sent in ASCII format.

• **RS:  Register Select**

RS = 0 -> Command Register is selected  
RS = 1 -> Data Register is selected

• **R/W: Read or Write**

0 -> Write, 1 -> Read

• **E: Enable (Latch data)**

Used to latch the data present on the data pins.  
A high-to-low edge is needed to latch the data.

**2.6 INTERNET OF THINGS (IOT)**

Internet of things (IoT), is another advance technology in IT sector, provides internetworking for numerous of devices such as sensors, actuators, PLCs and other electronic embedded smart devices and controls, and various software’s’ and provides systems network configuration and connectivity, which enables communication between these numerous devices for information exchanging.

In 1995, “thing to thing” was coined by BILL GATES. In 1999, IoT (Internet of Things) was come up by EPC global. IOT interconnects human to thing, thing to thing and human to human. The goal of IoT is bring out a huge network by combining different types connected devices. IoT targets three aspects Communication, automation, cost saving in a system. IOT empowers people to carry out routine activities using internet and thus saves time and cost making them more productive. IOT enables the objects to be sensed and/or controlled remotely across existing network model. IOT in environmental monitoring helps to know about the air and water quality, temperature and conditions of the soil, and also monitor the intrusion of animals in to the field. IOT can also play a significant role in precision farming to enhance the productivity of the farm.

Recent advancements, such as the vision of the Internet of Things (IoT), the cloud computing model, and cyber-physical systems, provide support for the transmission and management of huge amounts of data regarding the trends observed in environmental parameters. In this context, the current work presents three different IoT-based wireless sensors

for environmental and ambient monitoring: one employing User Datagram Protocol (UDP)-based Wi-Fi communication, one communicating through Wi-Fi and Hypertext Transfer Protocol (HTTP), and a third one using Bluetooth Smart. All of the presented systems provide the possibility of recording data at remote locations and of visualizing them from every device with an Internet connection, enabling the monitoring of geographically large areas. The development details of these systems are described, along with the major differences and similarities between them. The feasibility of the three developed systems for implementing monitoring applications, taking into account their energy autonomy, ease of use, solution complexity, and Internet connectivity facility, was analysed, and revealed that they make good candidates for IoT-based solutions.

Nowadays, IoT is one of the most advanced, efficient, and cost less technological solution which encompasses various hardware and software resources; and allows remotely connected sensing devices to sense with more capabilities, provides efficiency and can be monitored and controlled through deployed of existing systems or infrastructures, resulting the physical World integration with computer controllers (or systems).

As IoT provides interconnectivity among various real-time sensing sensors and PLC and other intelligent devices, therefore this technology will be an entity indicated for the more advance cyber-systems encircling the significant developments, “such as smart grid, smart vehicle systems, smart medical systems, smart cities, and others smart systems.” In early future, IoT has striven to provide advance or smart connectivity for variety of electronic and intelligent equipment’s or devices, IT-based systems and the more advanced services through deploying of various traditional and real-time protocols, networks domains, and system software/hardware applications, which will be an work followed by machine-to-machine technological concept.

Through interconnection of various devices and managing ofThe **internet of things** (**IoT**) is the network of physical devices, vehicles, buildings and other items [embedded](https://en.wikipedia.org/wiki/Embedded_system" \o "Embedded system) with [electronics](https://en.wikipedia.org/wiki/Electronics" \o "Electronics), software, sensors, actuators, and [network connectivity](https://en.wikipedia.org/wiki/Internet_access" \o "Internet access) that enable these objects to collect and exchange data. In 2013 the Global Standards Initiative on Internet of Things (IoT-GSI) defined the IoT as "the infrastructure of the information society. The IoT allows objects to be sensed and controlled remotely across existing network infrastructure, creating opportunities for more direct integration of the physical world into computer-based systems, and resulting in improved efficiency, accuracy and economic benefit.

When IoT is augmented with sensors and actuators, the technology becomes an instance of the more general class of [cyber-physical systems](https://en.wikipedia.org/wiki/Cyber-physical_system" \o "Cyber-physical system), which also encompasses technologies such as [smart grids](https://en.wikipedia.org/wiki/Smart_grid" \o "Smart grid), [smart homes](https://en.wikipedia.org/wiki/Smart_home" \o "Smart home), [intelligent transportation](https://en.wikipedia.org/wiki/Intelligent_transportation" \o "Intelligent transportation) and [smart cities](https://en.wikipedia.org/wiki/Smart_city" \o "Smart city). Each thing is uniquely identifiable through its embedded computing system but is able to interoperate within the existing [Internet](https://en.wikipedia.org/wiki/Internet" \o "Internet) infrastructure. Experts estimate that the IoT will consist of almost 50 billion objects by 2020.

IoT is a system defines an environment that encompasses numerous of objects; sensors that connected with these objects are accessible over the Internet through employing of various

Networks connections, such wired or wireless. IoT can be able to carry information from various embedded sensors attached with the physical World, human and any inanimate object, and then transmit them to a system for further analyses. In early future, IoT will be able to connect almost components or parts of industrial infrastructures, smart medical tele monitoring systems, and smart transportation systems; and will provide the information sharing facilities in order to make systems and peoples always updated.

**2.6.1 ESP8266 WIFI IOT MODULE**

**DESCRIPTION:**

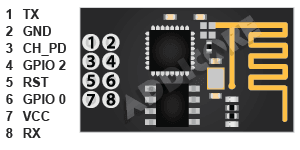
* These modules include 1MB (8Mbit) of flash memory, twice the size of the older blue colored ESP-01 module
* The ESP8266 Serial/UART to WiFi module is a great way to connect your Arduino or other microcontroller projects to a WiFi network
* Create your next internet of things (IOT) project with affordable network connectivity by implementing this module into your design
* The module has the ability to run independent of a host controller
* The eight pin header includes two GPIO pins that allow for direct connection of the module to sensors, peripherals, or host controller
  + Check out our [ESP8266 breadboard adapter](https://www.addicore.com/product-p/ad-bb-adtr.htm" \t "_blank) to use your ESP8266 module with a breadboard
* The ESP8266 has 3.6V tolerant I/Os so you will need a [logic level converter](https://www.addicore.com/product-p/227.htm" \t "_blank) to connect it with higher voltage devices such as Arduino
* The ESP8266 requires 3.3V power so you may need a [3.3V voltage regulator](https://www.addicore.com/product-p/165.htm" \t "_blank) to provide the correct voltage, depending on your setup

**PRODUCT CONTENTS:**

* 1 — ESP8266 ESP-01 WiFi Tranceiver Module with baud rate set at 115200 bps
* 1 (per order) — Addicore ESP8266 info card (includes pinout diagram)

**SPECIFICATIONS:**

* 802.11 b/g/n
* Serial/UART baud rate: 115200 bps
* Integrated TCP/IP protocol stack
* Input power: 3.3V (see "Recommended Accessories" below for 3.3V power options)
* I/O voltage tolerance: 3.6V Max (see "Recommended Accessories" below for level converters to connect to higher voltage devices (i.e. Arduino) )
* Regular operation current draw: ~70mA
* Peak operating current draw: ~300mA
* Power down leakage current: <10µA
* +19.5dBm output in 802.11b mode
* Flash Memory Size: 1MB (8Mbit)
* WiFi security modes: WPA, WPA2
* Module's dimensions: 24.75mm x 14.5mm (0.974" x 0.571")
* Through this module we can connect directly to the website and mobile (app) that is tcp/ip,and laptop
* Here ther are some ports, which are used to connect with different devices like,if i want to connect websitet through port80 which can available this information in datasheet and for mobile port 23 and for loptop putty(it is a software like xctu for communicating (data transmission and receiving purpose)
* For operating this module we have to set an ip address for this module 192.168.4.1



**2.7 GSM(GLOBAL SYSTEM FOR MOBILE)**

Global System for Mobile Communications (GSM) modems are specialized types of modems that operate over subscription based wireless networks, similar to a mobile phone. A GSM modem accepts a Subscriber Identity Module (SIM) card, and basically acts like a mobile phone for a computer. Such a modem can even be a dedicated mobile phone that the computer uses for GSM network capabilities.

Traditional modems are attached to computers to allow dial-up connections to other computer systems. A GSM modem operates in a similar fashion, except that it sends and receives data through radio waves rather than a telephone line. This type of modem may be an external device connected via a Universal Serial Bus (USB) cable or a serial cable. More commonly, however, it is a small device that plugs directly into the USB port or card slot on a computer or laptop.

It is widely used mobile communication system in the world. GSM is an open and digital cellular technology used for transmitting mobile voice and data services operates at the 850MHz, 900MHz, 1800MHz and 1900MHz frequency bands.

**Features**

* Improved spectrum efficiency
* International roaming
* Compatibility with integrated services digital network (ISDN)
* Support for new services.
* SIM phonebook management
* Fixed dialing number (FDN)
* Real time clock with alarm management
* High-quality speech
* Uses encryption to make phone calls more secure
* Short message service (SMS)

A GSM modem is a wireless modem that works with a GSM wireless network. A wireless modem behaves like a dial-up modem. The main difference between them is that a dial-up modem sends and receives data through a fixed telephone line while a wireless modem sends and receives data through radio waves.

A GSM modem can be an external device or a PC Card / PCMCIA Card. Typically, an external GSM modem is connected to a computer through a serial cable or a USB cable. A GSM modem in the form of a PC Card / PCMCIA Card is designed for use with a laptop computer. It should be inserted into one of the PC Card / PCMCIA Card slots of a laptop computer. Like a GSM mobile phone, a GSM modem requires a SIM card from a wireless carrier in order to operate.

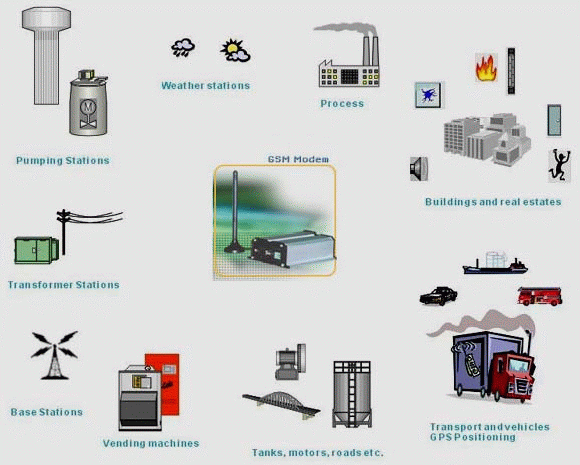
As mentioned in earlier sections of this SMS tutorial, computers use AT commands to control modems. Both GSM modems and dial-up modems support a common set of standard AT commands. You can use a GSM modem just like a dial-up modem.

In addition to the standard AT commands, GSM modems support an extended set of AT commands. These extended AT commands are defined in the GSM standards. With the extended AT commands, you can do things like:

* Reading, writing and deleting SMS messages.
* Sending SMS messages.
* Monitoring the signal strength.
* Monitoring the charging status and charge level of the battery.
* Reading, writing and searching phone book entries.

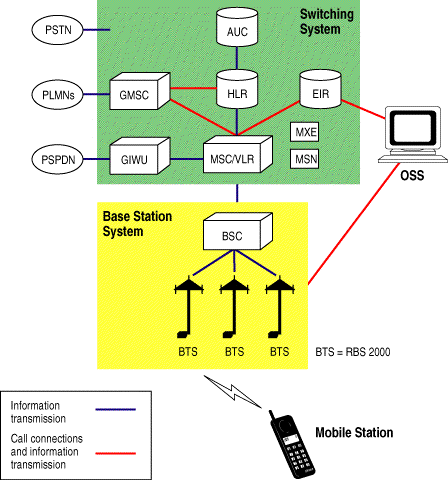
The number of SMS messages that can be processed by a GSM modem per minute is very low -- only about six to ten SMS messages per minute.

**GSM MODEM APPLICATION**



**THE GSM NETWORK**

GSM provides recommendations, not requirements. The GSM specifications define the functions and interface requirements in detail but do not address the hardware. The reason for this is to limit the designers as little as possible but still to make it possible for the operators to buy equipment from different suppliers. The GSM network is divided into three major systems: the switching system (SS), the base station system (BSS), and the operation and support system (OSS).



GSM Network Elements

**The Switching System**

The switching system (SS) is responsible for performing call processing and subscriber-related functions. The switching system includes the following functional units.

* **Home location register (HLR)**—The HLR is a database used for storage and management of subscriptions. The HLR is considered the most important database, as it stores permanent data about subscribers, including a subscriber's service profile, location information, and activity status. When an individual buys a subscription from one of the PCS operators, he or she is registered in the HLR of that operator.
* **mobile services switching center (MSC)**—The MSC performs the telephony switching functions of the system. It controls calls to and from other telephone and data systems. It also performs such functions as toll ticketing, network interfacing, common channel signaling, and others.
* **visitor location register (VLR)**—The VLR is a database that contains temporary information about subscribers that is needed by the MSC in order to service visiting subscribers. The VLR is always integrated with the MSC. When a mobile station roams into a new MSC area, the VLR connected to that MSC will request data about the mobile station from the HLR. Later, if the mobile station makes a call, the VLR will have the information needed for call setup without having to interrogate the HLR each time.
* **authentication center (AUC)**—A unit called the AUC provides authentication and encryption parameters that verify the user's identity and ensure the confidentiality of each call. The AUC protects network operators from different types of fraud found in today's cellular world.
* **equipment identity register (EIR)**—The EIR is a database that contains information about the identity of mobile equipment that prevents calls from stolen, unauthorized, or defective mobile stations. The AUC and EIR are implemented as stand-alone nodes or as a combined AUC/EIR node.

**The Base Station System (BSS)**

All radio-related functions are performed in the BSS, which consists of base station controllers (BSCs) and the base transceiver stations (BTSs).

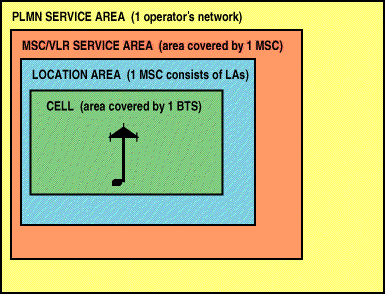
* **BSC**—The BSC provides all the control functions and physical links between the MSC and BTS. It is a high-capacity switch that provides functions such as handover, cell configuration data, and control of radio frequency (RF) power levels in base transceiver stations. A number of BSCs are served by an MSC.
* **BTS**—The BTS handles the radio interface to the mobile station. The BTS is the radio equipment (transceivers and antennas) needed to service each cell in the network. A group of BTSs are controlled by a BSC.

**The Operation and Support System**

The operations and maintenance center (OMC) is connected to all equipment in the switching system and to the BSC. The implementation of OMC is called the operation and support system (OSS). The OSS is the functional entity from which the network operator monitors and controls the system. The purpose of OSS is to offer the customer cost-effective support for centralized, regional, and local operational and maintenance activities that are required for a GSM network. An important function of OSS is to provide a network overview and support the maintenance activities of different operation and maintenance organizations.

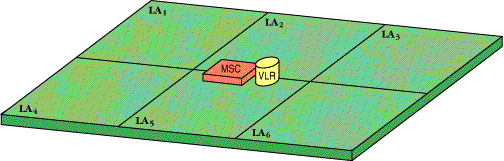
**GSM NETWORK AREAS**

The GSM network is made up of geographic areas. As shown in 4.2, these areas include cells, location areas (LAs), MSC/VLR service areas, and public land mobile network (PLMN) areas.



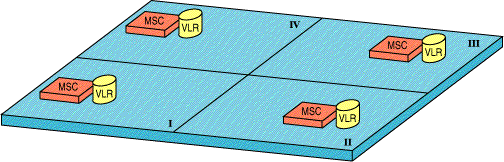
**Network Areas**

The cell is the area given radio coverage by one base transceiver station. The GSM network identifies each cell via the cell global identity (CGI) number assigned to each cell. The location area is a group of cells. It is the area in which the subscriber is paged. Each LA is served by one or more base station controllers, yet only by a single MSC (fig 4.3). Each LA is assigned a location area identity (LAI) number.



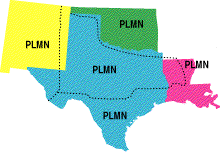
**Location Areas**

An MSC/VLR service area represents the part of the GSM network that is covered by one MSC and which is reachable, as it is registered in the VLR of the MSC (fig 4.4).



MSC /VLR Service Areas

The PLMN service area is an area served by one network operator(fig 4.5).



**SPECIFICATIONS AND CHARACTERISTICS FOR GSM**

* **frequency band**—The frequency range specified for GSM is 1,850 to 1,990 MHz (mobile station to base station).
* **duplex distance**—The duplex distance is 80 MHz. Duplex distance is the distance between the uplink and downlink frequencies. A channel has two frequencies, 80 MHz apart.
* **channel separation**—The separation between adjacent carrier frequencies. In GSM, this is 200 kHz.
* **modulation**—Modulation is the process of sending a signal by changing the characteristics of a carrier frequency. This is done in GSM via Gaussian minimum shift keying (GMSK).
* **transmission rate**—GSM is a digital system with an over-the-air bit rate of 270 kbps.
* **access method**—GSM utilizes the time division multiple access (TDMA) concept. TDMA is a technique in which several different calls may share the same carrier. Each call is assigned a particular time slot.
* **speech coder**—GSM uses linear predictive coding (LPC). The purpose of LPC is to reduce the bit rate. The LPC provides parameters for a filter that mimics the vocal tract. The signal passes through this filter, leaving behind a residual signal. Speech is encoded at 13 kbps.

**GSM SUBSCRIBER SERVICES**

There are two basic types of services offered through GSM: telephony (also referred to as teleservices) and data (also referred to as bearer services). Telephony services are mainly voice services that provide subscribers with the complete capability (including necessary terminal equipment) to communicate with other subscribers. Data services provide the capacity necessary to transmit appropriate data signals between two access points creating an interface to the network. In addition to normal telephony and emergency calling, the following subscriber services are supported by GSM:

* **dual-tone multifrequency (DTMF)**—DTMF is a tone signaling scheme often used for various control purposes via the telephone network, such as remote control of an answering machine. GSM supports full-originating DTMF.
* **facsimile group III**—GSM supports CCITT Group 3 facsimile. As standard fax machines are designed to be connected to a telephone using analog signals, a special fax converter connected to the exchange is used in the GSM system. This enables a GSM–connected fax to communicate with any analog fax in the network.
* **short message services**—A convenient facility of the GSM network is the short message service. A message consisting of a maximum of 160 alphanumeric characters can be sent to or from a mobile station. This service can be viewed as an advanced form of alphanumeric paging with a number of advantages. If the subscriber's mobile unit is powered off or has left the coverage area, the message is stored and offered back to the subscriber when the mobile is powered on or has reentered the coverage area of the network. This function ensures that the message will be received.
* **cell broadcast**—A variation of the short message service is the cell broadcast facility. A message of a maximum of 93 characters can be broadcast to all mobile subscribers in a certain geographic area. Typical applications include traffic congestion warnings and reports on accidents.
* **voice mail**—This service is actually an answering machine within the network, which is controlled by the subscriber. Calls can be forwarded to the subscriber's voice-mail box and the subscriber checks for messages via a personal security code.
* **fax mail**—With this service, the subscriber can receive fax messages at any fax machine. The messages are stored in a service center from which they can be retrieved by the subscriber via a personal security code to the desired fax number.

**2.8 RESET**

**2.9 PH SENSOR**

**MEASUREMENT OVERVIEW**

The Greenspan pH sensor utilizes a robust gel filled industrial pH electrode for field monitoring in a variety of environments.

The pH electrode consists of a pH sensitive glass membrane sealed to a glass insulating tube containing a solution of fixed pH in contact with a silver-silver chloride half cell. The potential developed across the membrane is compared to a stable reference potential eg. a silver-silver chloride half cell in contact with a gel electrolyte containing chloride. Completion of the circuit is by means of a porous constriction (the salt bridge) which allows the reference electrolyte to slowly flow into the sample. The gel electrode is a sealed reference.

pH provides an indication of whether a solution is acidic or basic and is defined as :

pH = - log (H+)

and covers a scale from 0 (acid) to 14 (alkaline) where H+ is the hydrogen concentration in solution, at normal room temperatures

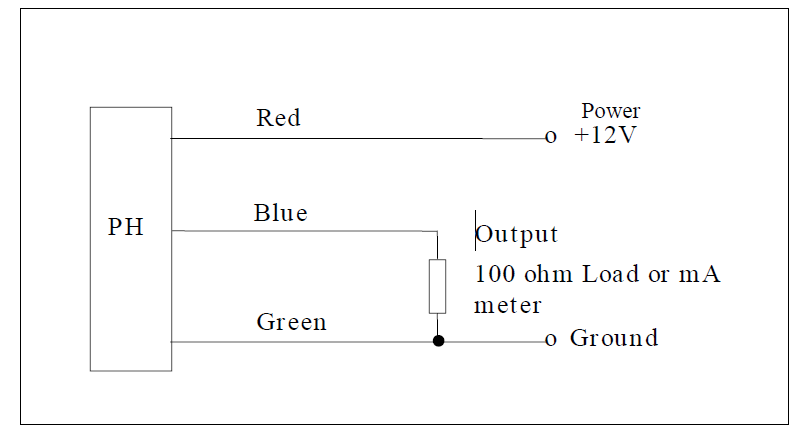
H2O = H+ +OH-

The concentration of each type of ion is approximately10-7gm molecule/litre and hence the pH of pure water is:

pH = - log 10-7 = 7

**CONNECTION DIAGRAM**

The following diagram describes the wiring arrangement for the PH100



**SPECIFICATION**

**Specification Model PH100**

Standard Range available 0-14pH

Temperature Range 0 - 50°C

Resolution 0.001 pH

Accuracy ± 0.2 pH

Temperature Compensation 0 - 50°C

Supply Voltage 11-13VDC

Reverse polarity protected

Surge protected to 2kV

Output 4 - 20mA, 3 wire

Dimensions length 226 mm, 32 mm OD

Storage Temperature - 20 to +60°C

**2.10 TURBIDITY SENSOR**

**Turbidity sensor**

Introduction The turbidity sensor detects water quality by measuring the levels of turbidity. It uses light to detect suspended particles in water by measuring the light transmittance and scattering rate, which changes with the amount of total suspended solids (TSS) in water. As the TTS increases, the liquid turbidity level increases. Turbidity sensors are used to measure water quality in rivers and streams, wastewater and effluent measurements, control instrumentation for settling ponds, sediment transport research and laboratory measurements. This sensor provides analog and digital signal output modes. The threshold is adjustable when in digital signal mode. You can select the mode according to your MCU.



**Specification**

* Operating Voltage: 5V DC
* Operating Current: 40mA (MAX)
* Response Time : <500ms
* Insulation Resistance: 100M (Min)
* Output Method:
* Analog output: 0-4.5V
* Digital Output: High/Low level signal (you can adjust the threshold value by adjusting the potentiometer)
* Operating Temperature: 5Ԩ~90Ԩ
* Storage Temperature: -10Ԩ~90Ԩ
* Weight: 30g
* Adapter Dimensions: 38mm\*28mm\*10mm/1.5inches \*1.1inches\*0.4inches

**Interface Description:**

1. "D/A" Output Signal Switch 1. "A": Analog Signal Output, the output value will decrease when in liquids with a high turbidity

2. "D": Digital Signal Output, high and low levels, which can be adjusted by the threshold potentiometer 2. Threshold Potentiometer: you can change the trigger condition by adjusting the threshold potentiometer in digital signal mode.

**CHAPTER 3**

**SOFTWARE SPECIFICATION**

**ARDUINO INTEGRATED DEVELOPMENT ENVIRONMENT (IDE)**

Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

**3.1 Writing Sketches**

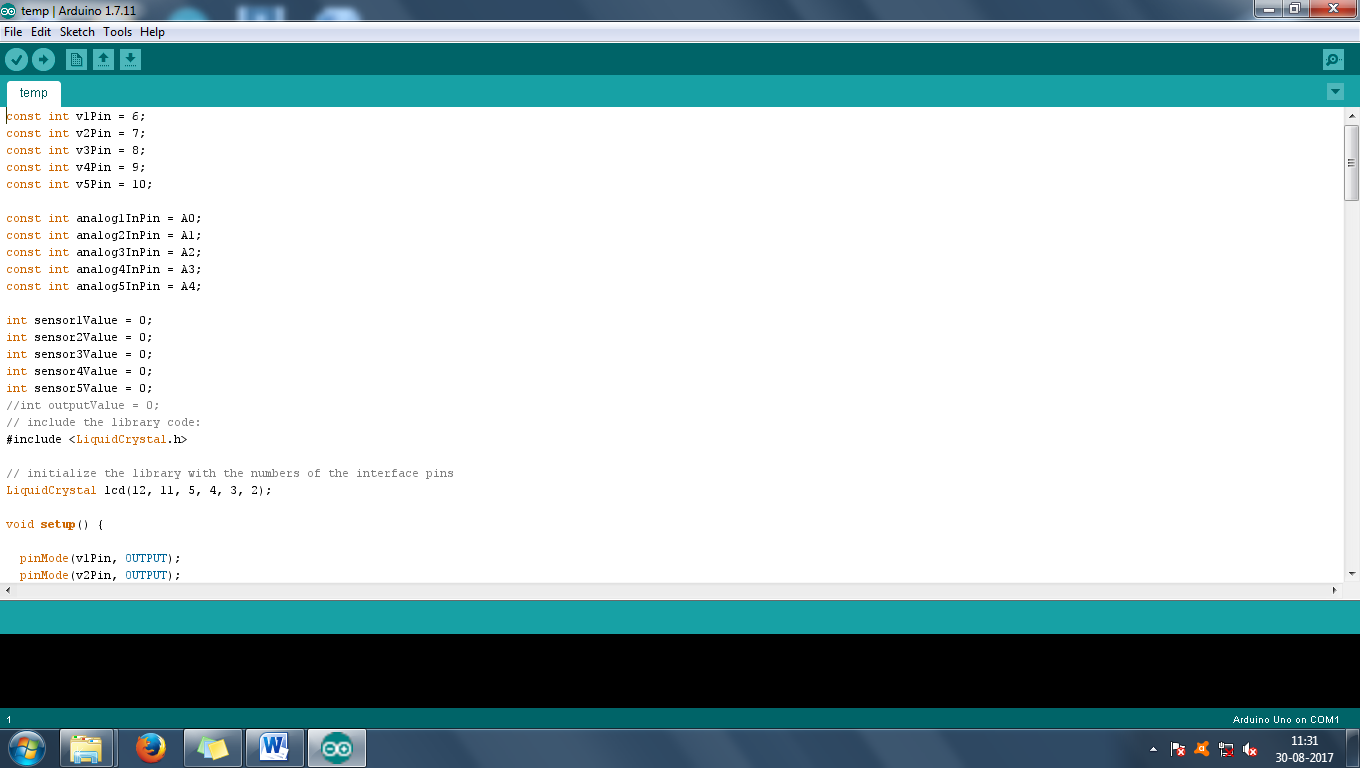
Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension ino. the editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom right hand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

NB: Versions of the Arduino Software (IDE) prior to 1.0 saved sketches with the extension .pde. It is possible to open these files with version 1.0, you will be prompted to save the sketch with the .ino extension on save.

|  |  |
| --- | --- |
|  | *Verify*  Checks your code for errors compiling it. |
| https://www.arduino.cc/en/uploads/Guide/export.png | *Upload*  Compiles your code and uploads it to the configured board. See [uploading](https://www.arduino.cc/en/guide/environment" \l "uploading) below for details.  Note: If you are using an external programmer with your board, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer" |
| https://www.arduino.cc/en/uploads/Guide/new.png | *New*  Creates a new sketch. |
| https://www.arduino.cc/en/uploads/Guide/open.png | *Open*  Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.  Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the File | Sketchbookmenu instead. |
| https://www.arduino.cc/en/uploads/Guide/save.png | *Save*  Saves your sketch. |
| https://www.arduino.cc/en/uploads/Guide/serial_monitor.png | *Serial Monitor*  Opens the [serial monitor](https://www.arduino.cc/en/guide/environment" \l "serialmonitor). |

Additional commands are found within the five menus: File, Edit, Sketch, Tools, Help. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

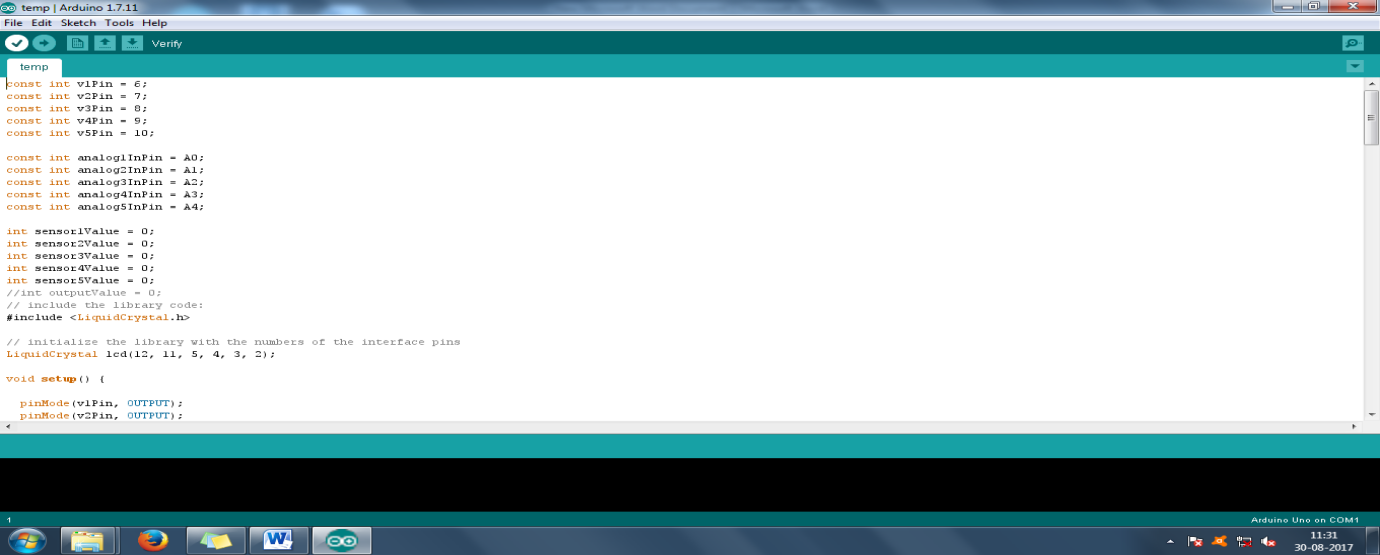
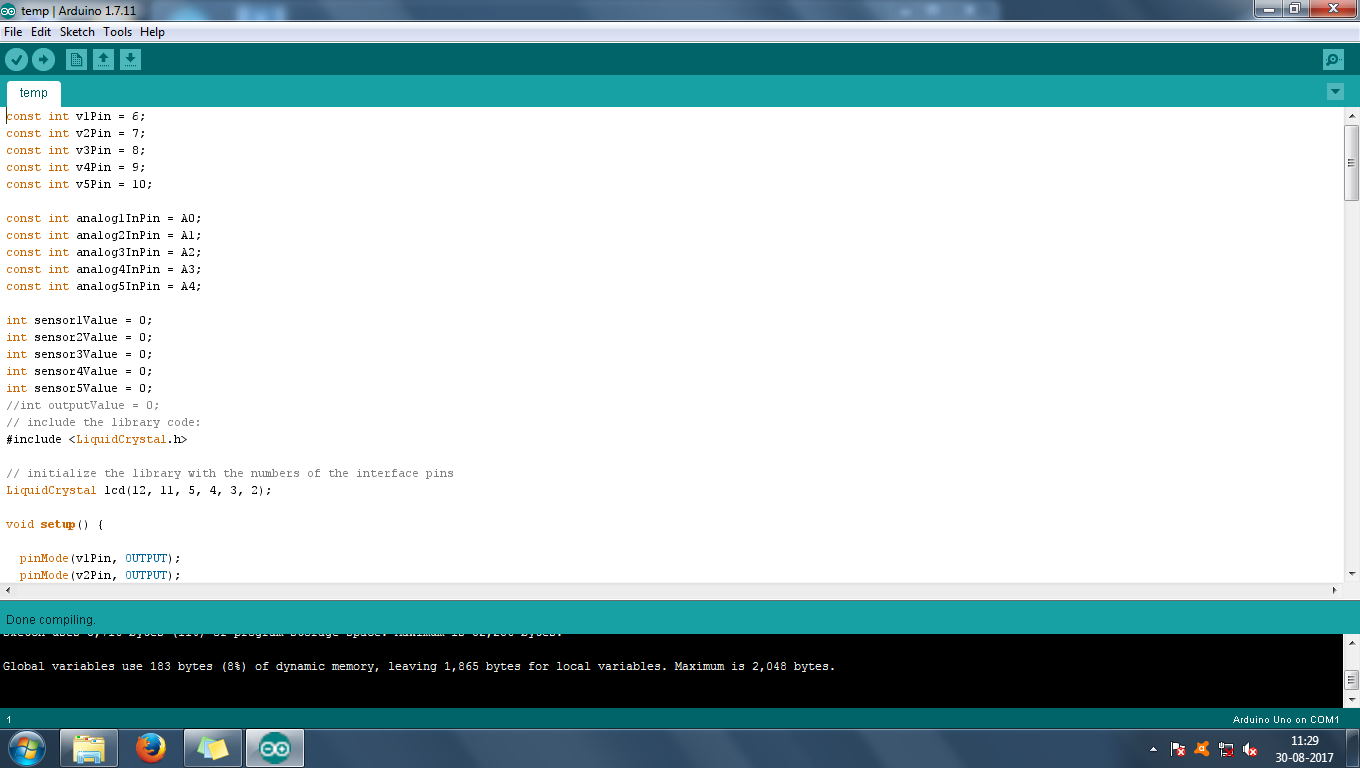
**3.2 File**

* *New*   
  Creates a new instance of the editor, with the bare minimum structure of a sketch already in place.
* *Open*   
  Allows to load a sketch file browsing through the computer drives and folders.
* *Open Recent*   
  Provides a short list of the most recent sketches, ready to be opened.
* 
* *Sketchbook*   
  Shows the current sketches within the sketchbook folder structure; clicking on any name opens the corresponding sketch in a new editor instance.
* *Examples*   
  Any example provided by the Arduino Software (IDE) or library shows up in this menu item. All the examples are structured in a tree that allows easy access by topic or library.
* *Close*   
  Closes the instance of the Arduino Software from which it is clicked.
* *Save*   
  Saves the sketch with the current name. If the file hasn't been named before, a name will be provided in a "Save as.." window.
* *Save as...*   
  Allows to save the current sketch with a different name.
* *Page Setup*   
  It shows the Page Setup window for printing.
* *Print*   
  Sends the current sketch to the printer according to the settings defined in Page Setup.
* *Preferences*   
  Opens the Preferences window where some settings of the IDE may be customized, as the language of the IDE interface.
* *Quit*   
  Closes all IDE windows. The same sketches open when Quit was chosen will be automatically reopened the next time you start the IDE.

**3.3 Edit**

* *Undo/Redo*   
  Goes back of one or more steps you did while editing; when you go back, you may go forward with Redo.
* *Cut*   
  Removes the selected text from the editor and places it into the clipboard.
* *Copy*   
  Duplicates the selected text in the editor and places it into the clipboard.
* *Copy for Forum*   
  Copies the code of your sketch to the clipboard in a form suitable for posting to the forum, complete with syntax coloring.
* *Copy as HTML*   
  Copies the code of your sketch to the clipboard as HTML, suitable for embedding in web pages.
* *Paste*   
  Puts the contents of the clipboard at the cursor position, in the editor.
* *Select All*   
  Selects and highlights the whole content of the editor.
* *Comment/Uncomment*   
  Puts or removes the // comment marker at the beginning of each selected line.
* *Increase/Decrease Indent*   
  Adds or subtracts a space at the beginning of each selected line, moving the text one space on the right or eliminating a space at the beginning.
* *Find*   
  Opens the Find and Replace window where you can specify text to search inside the current sketch according to several options.
* *Find Next*   
  Highlights the next occurrence - if any - of the string specified as the search item in the Find window, relative to the cursor position.
* *Find Previous*   
  Highlights the previous occurrence - if any - of the string specified as the search item in the Find window relative to the cursor position.

**3.4 Sketch**

* *Verify/Compile*   
  Checks your sketch for errors compiling it; it will report memory usage for code and variables in the console area.
* 
* *Upload*   
  Compiles and loads the binary file onto the configured board through the configured Port.
* 
* *Upload Using Programmer*   
  This will overwrite the bootloader on the board; you will need to use Tools > Burn Bootloader to restore it and be able to Upload to USB serial port again. However, it allows you to use the full capacity of the Flash memory for your sketch. Please note that this command will NOT burn the fuses. To do so a *Tools -> Burn Bootloader* command must be executed.
* *Export Compiled Binary*   
  Saves a .hex file that may be kept as archive or sent to the board using other tools.
* *Show Sketch Folder*   
  Opens the current sketch folder.
* *Include Library*   
  Adds a library to your sketch by inserting #include statements at the start of your code. For more details, see [libraries](https://www.arduino.cc/en/guide/environment" \l "libraries) below. Additionally, from this menu item you can access the Library Manager and import new libraries from .zip files.
* *Add File...*   
  Adds a source file to the sketch (it will be copied from its current location). The new file appears in a new tab in the sketch window. Files can be removed from the sketch using the tab menu accessible clicking on the small triangle icon below the serial monitor one on the right side o the toolbar.

**3.5. Tools**

* *Auto Format*   
  This formats your code nicely: i.e. indents it so that opening and closing curly braces line up, and that the statements inside curly braces are indented more.
* *Archive Sketch*   
  Archives a copy of the current sketch in .zip format. The archive is placed in the same directory as the sketch.
* *Fix Encoding & Reload*   
  Fixes possible discrepancies between the editor char map encoding and other operating systems char maps.
* *Serial Monitor*   
  Opens the serial monitor window and initiates the exchange of data with any connected board on the currently selected Port. This usually resets the board, if the board supports Reset over serial port opening.
* *Board*   
  Select the board that you're using. See below for [descriptions of the various boards](https://www.arduino.cc/en/guide/environment" \l "boards).
* *Port*   
  This menu contains all the serial devices (real or virtual) on your machine. It should automatically refresh every time you open the top-level tools menu.
* *Programmer*   
  For selecting a harware programmer when programming a board or chip and not using the onboard USB-serial connection. Normally you won't need this, but if you're [burning a bootloader](https://www.arduino.cc/en/Tutorial/Bootloader) to a new microcontroller, you will use this.
* *Burn Bootloader*   
  The items in this menu allow you to burn a [bootloader](https://www.arduino.cc/en/Hacking/Bootloader) onto the microcontroller on an Arduino board. This is not required for normal use of an Arduino or Genuino board but is useful if you purchase a new ATmega microcontroller (which normally come without a bootloader). Ensure that you've selected the correct board from the Boards menu before burning the bootloader on the target board. This command also set the right fuses.

Help

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

* *Find in Reference*   
  This is the only interactive function of the Help menu: it directly selects the relevant page in the local copy of the Reference for the function or command under the cursor.

**3.6 Sketch book**

The Arduino Software (IDE) uses the concept of a sketchbook: a standard place to store your programs (or sketches). The sketches in your sketchbook can be opened from the File >Sketchbook menu or from the Open button on the toolbar. The first time you run the Arduino software, it will automatically create a directory for your sketchbook. You can view or change the location of the sketchbook location from with the Preferences dialog.

Beginning with version 1.0, files are saved with a .ino file extension. Previous versions use the .pde extension. You may still open .pde named files in version 1.0 and later, the software will automatically rename the extension to .ino.

Tabs, Multiple Files, and Compilation

Allows you to manage sketches with more than one file (each of which appears in its own tab). These can be normal Arduino code files (no visible extension), C files (.c extension), C++ files (.cpp), or header files (.h).

**3.7 Uploading**

Before uploading your sketch, you need to select the correct items from the Tools > Board and Tools > Port menus. The [boards](https://www.arduino.cc/en/guide/environment" \l "boards) are described below. On the Mac, the serial port is probably something like /dev/tty.usbmodem241 (for an Uno or Mega2560 or Leonardo) or /dev/tty.usbserial-1B1 (for a Duemilanove or earlier USB board), or /dev/tty.USA19QW1b1P1.1 (for a serial board connected with a Keyspan USB-to-Serial adapter). On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, you look for USB serial device in the ports section of the Windows Device Manager. On Linux, it should be /dev/ttyACMx , /dev/ttyUSBx or similar. Once you've selected the correct serial port and board, press the upload button in the toolbar or select the Upload item from the Sketch menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino Software (IDE) will display a message when the upload is complete, or show an error.

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

**3.8 Libraries**

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

There is a [list of libraries](https://www.arduino.cc/en/Reference/Libraries) in the reference. Some libraries are included with the Arduino software. Others can be downloaded from a variety of sources or through the Library Manager. Starting with version 1.0.5 of the IDE, you do can import a library from a zip file and use it in an open sketch. See these [instructions for installing a third-party library](https://www.arduino.cc/en/Guide/Libraries).

To write your own library, see [this tutorial](https://www.arduino.cc/en/Hacking/LibraryTutorial).

Third-Party Hardware

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

For details on creating packages for third-party hardware, see the [Arduino IDE 1.5 3rd party Hardware specification](https://github.com/arduino/Arduino/wiki/Arduino-IDE-1.5-3rd-party-Hardware-specification" \t "_blank).

**3.9 Serial Monitor**

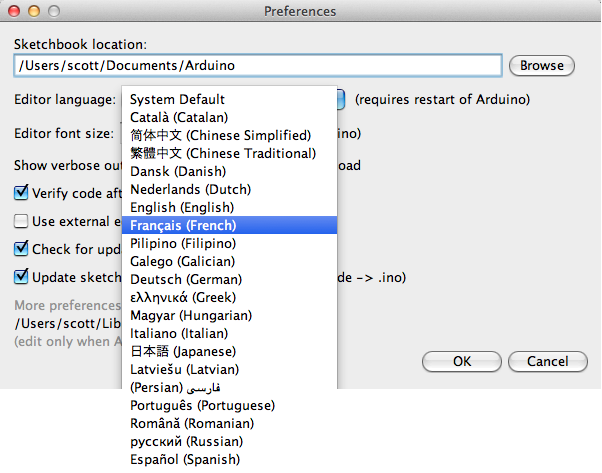
Displays serial data being sent from the Arduino or Genuino board (USB or serial board). To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down that matches the rate passed to Serial.begin in your sketch. Note that on Windows, Mac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

You can also talk to the board from Processing, Flash, MaxMSP, etc (see the [interfacing page](http://www.arduino.cc/playground/Main/Interfacing) for details).

**3.10 Preferences**

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

**Language Support**



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

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

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

**3.11 Boards**

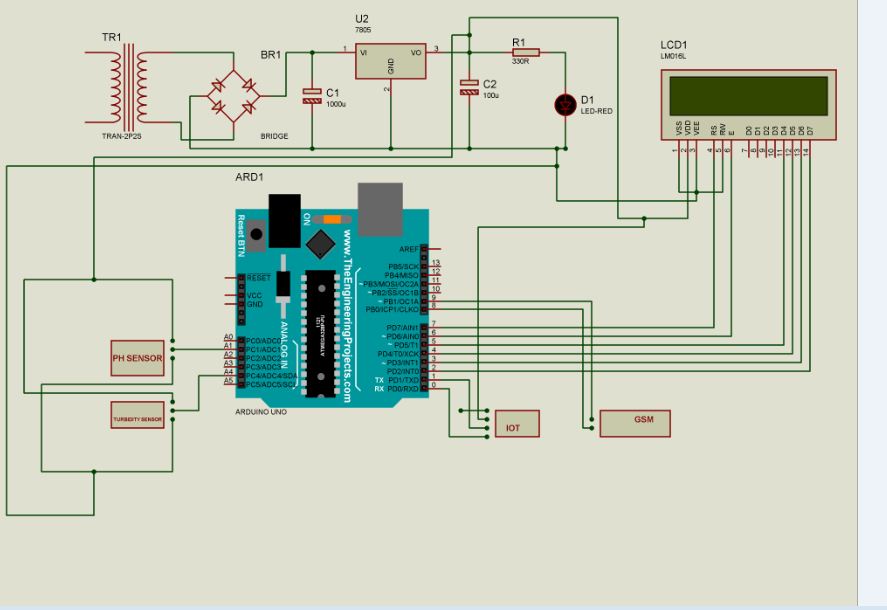
The board selection has two effects: it sets the parameters (e.g. CPU speed and baud rate) used when compiling and uploading sketches; and sets and the file and fuse settings used by the burn bootloader command. Some of the board definitions differ only in the latter, so even if you've been uploading successfully with a particular selection you'll want to check it before burning the bootloader. You can find a comparison table between the various boards [here](https://www.arduino.cc/en/Products/Compare).

Arduino Software (IDE) includes the built in support for the boards in the following list, all based on the AVR Core. The [Boards Manager](https://www.arduino.cc/en/Guide/Cores) included in the standard installation allows to add support for the growing number of new boards based on different cores like Arduino Due, Arduino Zero, Edison, Galileo and so on.

**CHAPTER 4**

**IMPLEMENTATION**

**4.1 SCHEMATIC DIAGRAM:**

****

**4.2 WORKING PRINCIPLE**

**CHAPTER 5**

**SIMULATION AND DESIGN**

**5.1 OUTPUT SCREEN SHOTS**

**CHAPTER 6**

**CONCLUSION**

The Project “An IoT based system for water quality monitoring” has been successfully designed and experimented. We have seen the success of sensors in various fields; the same idea has been applied to this water quality monitoring system. In this paper we have analysis different water quality monitoring systems to do the same. All these techniques are expensive and difficult in terms of analysis and collecting the data. ACKNO

**REFERENCES:**

1.M.F. Saaid, A.Sanuddin, Megat Ali “Automated ph controller system for hydroponic cultivation” Symposium on computer applications and industrial electronics(ISCAIE) IEEE, pp.186-190,2015.

[2] Lakshmanan, L. and Jalasri., (2019), “A Survey: Integration of IoT and Fog Computing”, IEEE International Conference on , ICCSP 2018, IEEE Conference Proceeding International Conference on Green Computing and Internet of Things (ICGCIoT),pp. 0235 - 0239, DOI: 10

[3] Ayushi S.jaiswal, Vaidehi Baporikar “Embedded wireless data acquisition syatem for unnamed vehicle in underwater” IEEE underwater technology, vol 10,pp.1-6,2015.

[4] F Ntambi, C P kruger, B J silva “Design of a water management system” AFRICON IEEE pp.1-5,2015

[5] Jesudoss A., Muthuram B.O. and Lourdson Emmanuel A., “Safe Driving using IoT Sensors”, conducted by International Conference on Engineering and Advancement in Technologies (ICEAT 2018), conduted by Sri Krishna College of Technology, Kovaipudur, Coimbatore

[6] Ji wang, Xiao-liRen, Yu-lishen “A Remote wireless sensor networks for water quality monitoring” International conference on innovative computing and communication, IEEE pp.7-12, 2010

[7] Cheng-Liang Lai, Chien-Lun chiu “Using image processing technology for water quality monitoring system” International conferenceonmachine learning and cybernetics.IEEE pp.1856- 1861,2011.

[8] Lakshmanan, L. and Jesudoss and Ulagamuthalvi, (2018), “Cluster Based Routing Scheme for Heterogeneous Nodes in WSN–A Genetic Approach”,ICICI 2018: International Conference on Intelligent Data CommunicationTechnologies and Internet of Things (ICICI) 2018,LNDECT, volume 26,pp 1013-1022

[9] Aravindan S.Rao, Stephan Martial, Jayavardhan Gubbi, “Design of low cost autonomous water quality monitoring system” International conference on advances in computing communications and informatics(ICACCI) IEEE,pp.14-19.2013.

[10] H Doukkali, L Nuaymi, “Analysis of MAC protocols of underwater sensor networking” Vehicular Technology conference UTC 2005 IEEE 61st,vol.2,pp.1307-1311,2005

[11] IF Akyildiz, D pompili, T Melodia,”Underwater acoustic sensor networks:research challenges”, Ad hoc Networks,vol.3,no.3,pp.257- 259,2005.

[12] Jesudoss A., R. Vybhavi and B. Anusha, “Design of Smart Helmet for Accident Avoidance”, IEEE International Conference on Communciations and Signal Processing, organized by Adhiparasakthi Engineering College, Chennai, April 4 to 6, 2019.

[13] I.Vasilesu, K.Kotay, D.Rus, P.Corke “Data collection storage and retrieval with an underwater sensor network”, IEEE Sensys, pp.154-165, 2005.

[14] Niel Andre Cleote, Reza Malekian and Lakshmi nair “Design of smart sensors for real time water quality monitoring” IEEE Access vol-4,no.9, pp.1-16,2014.

[15] P-Jiang, H.Xia, Z.He, Z-Wang “Design of a water environment monitoring system based on wireless sensor networks”, Sensors,Vol.9,no.8,pp.6411-6434,2009.

[16] R.Fisher, L.Ledwaba, G.P Hancke, C.P Kruger, “Open hardware :A Role to play in wireless sensor Networks” sensrs 2015, Vol.15, no.3,pp 6818-6844,2015

[17] T-P Lambrro, C.C Anastastiou, C.G Panayiotou, “A nephelometric turbidity systems for monitoring residential drinking water quality” in sensor Applications Experimentation Vol 29,2010.

[18] Vinod Raut, Sushama shelke, “Wireless acquisition system for water quality monitoring”, Advaances in singleprocessing (CASP) conference on pp.371-374,2016.

[19] Yi Hong Wang, Ju Chen, Shi Feng Tao, “Waelet Fusion Application in water quality warning based on bio-detection technology”,Applied mechanics ad materials, Vol 295-298,pp 924,2013..1109/ICGCIoT.2018.8753010.