Agricultural automation using GSM

A Term Paper Report

\Submitted in partial fulfillment of the requirements for

the award of the degree of

Bachelor of Technology

in

ELECTRONICS AND COMMUNICATION ENGINEERING

by

A.SAI VIGNESH (150040022)

P.SAI AVINASH (150040686)

N.MAHESH (150040582)

under the supervision of

SRIPATH ROY



**KONERU LAKSHMAIAH EDUCATIONAL FOUNDATION**

Green Fields,Vaddeswaram,Guntur(Dist)- 522502

Andhra Pradesh, India.

April , 2018

Agricultural automation using GSM



**KONERU LAKSHMAIAH EDUCATIONAL FOUNDATION**

**Green Fields,Vaddeswaram**

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION**

**ENGINEERING**

**Declaration**

The Term Paper Report entitled “Agricultural automation using GSM module“ is a record of bonafide A.Saivignesh(150040022), P.SaiAvinash(150040686), N.Mahesh(150040582)submitted in partial fulfillment for the award of B.Tech in electronics and communication engineering to K L Deemed to be a University. The results embodied in this report have not been copied from any other department/University/Institute.

1.A.Sai vignesh(150040022)

2.P.Sai Avinash(150040686)

3.N.Mahesh (150040582)



**KONERU LAKSHMAIAH EDUCATIONAL FOUNDATION**

Green Fields,Vaddeswaram

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

**Certificate**

This is to certify that the Term Paper Report entitled “ Agricultural automation using GSM module” is being submitted by A.Sai vignesh (150040022), P.Sai Avinash(150040686)

N.Mahesh(150040582) submitted in partial fulfillment for the award of B.Tech in electronics and communication engineering to K L Deemed to be a University is a record of bonafide work carried out under our guidance and supervision.

The results embodied in this report have not been copied from any other department/University/Institute.

**Signature of the Supervisor Signature of the Research Head**

Name : sripath roy Name : sripath roy

Designation : Asst.Prof Designation : Asst.Prof

Affiliation : KLEF,Vijayawada. Affiliation : KLEF,Vijayawada.

**ABSTRACT**

This project work investigates the potential of `GSM based Agricultural control', which is the aim of the Agricultural Automation Systems in near future. The analysis and implementation of the agricultural automation technology using Global System for Mobile Communication (GSM) modem to control agricultural appliances such as light ,agricultural motor system, and security system via Short Message Service (SMS) text messages is presented in this paper. The proposed research work is focused on functionality of the GSM protocol, which allows the user to control the target system away from residential using the frequency bandwidths. The concept of serial communication and AT-commands has been applied towards development of the smart GSM-based agricultural automation system.Farmers will be able to receive feedback status of any agricultural appliances under control whether switched on or off remotely from their mobile phones. ATmega328 microcontroller with the integration of GSM provides the smart automated agricultural system with the desired baud rate of 9600 bps. The proposed prototype of GSM based agricultural automation system was implemented and tested with maximum of four loads and shows the accuracy of ≥98%.

We had also developed an app using MIT app inventor which will give more user interface and made controlling easy.

**Table of concepts**

1. Introduction
2. Hardware & Software used
3. Theoretical Analysis & architectures
4. Block diagram
5. Procedure
6. Flow diagram
7. Code
8. Discussion of Results
9. Summary, Conclusion and Future Scope

(If any Original Contribution then it is to be highlighted)

1. Reference/bibliography
2. Appendices (if any)

# **INTRODUCTION**

In recent years, there has been a growing interest among consumers in the automation concept .Agricultural automation contain multiple, connected devices such as Agricultural motor control, security systems, lighting, access control systems and surveillance. Intelligent Agricultural automation system is incorporated into smart agricultural lands to provide comfort, convenience, and security to farmers . Agricultural automation system represents and reports the status of the connected devices in an intuitive, user-friendly interface allowing the user to interact and control various devices with the touch of a few buttons. Some of the major communication technologies used by today’s Agricultural automation system include [1-3]Bluetooth, WiMAX and Wireless LAN (Wi-Fi), Zigbee, and Global System for Mobile Communication (GSM).

All GSM is one of the most widely used cellular technologies in the world [4,5]. With the increase in the number of GSM subscribers, research and development [6-8] is heavily supported in further investigating the GSM implementation. In 2009, Das, Sanaullah, et. al. [9] developed a cell phone based remote management and control system for home appliances. However, a few limitations for the system include not being able to control multiple appliances concurrently and the home automations system could not verify the status of the appliances. ElKamchouchi and ElShafee [2] presented the design and prototype implementation of basic home automation system based on SMS technology using AT89C55 Atmel microcontroller. The microcontroller acts as the bridge between the GSM network and sensors of the home automation system. Further researches have been conducted to analyze the performance of other home automation control system [10,12]. Internet and wireless communications have also been utilized in parallel with GSM for home automations [13]. Among the cellular technologies, GSM network is preferred for the communication between the home appliances and the user due to its wide spread coverage [1] which makes the whole system online for almost all the time. Another advantage of using the GSM network in home automation is its high security infrastructure, which provides maximum reliability whereby other people cannot monitor the information sent or received. Using this technology we had implemented it in agricultural machines with SMS based control using the GSM architecture without accessing the local network

**2.Hardware & Software used**

**2.1 Hardware used :**

1.SIM900A GSM MODULE

2.ARDUINO UNO

3.CHANNEL RELAY(10A 250V)

4.DS3231 RTC MODULE

**SIM900A GSM MODULE:**

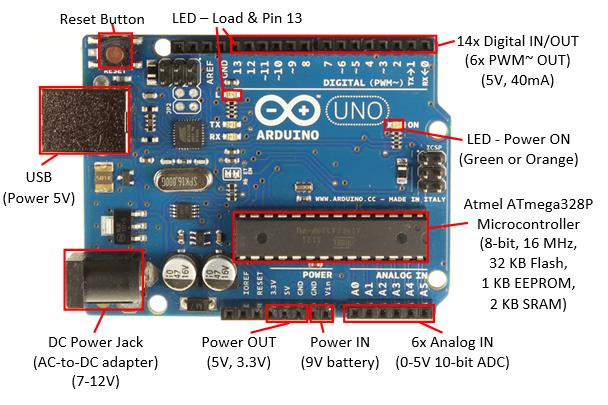


[14] SIM800A Quad Band GSM/GPRS Module with RS232 Interface is a complete Quad-band GSM/GPRS solution in a LGA(Land grid array) type which can be embedded in the customer applications. SIM800A support Quad-band 850/900/1800/1900 MHz, it can transmit Voice, SMS and data information with low power consumption. With tiny size of 100 x 53 x 15 mm, it can fit into slim and compact demands of customer design. Featuring and Embedded AT, it allows total cost savings and fast time-to-market for customer applications. Here are some GSM module in[Robu.in](https://robu.in/product-category/wireless/gsmgprs/).The SIM800A modem has a SIM800A GSM chip and RS232 interface while enables easy connection with the computer or laptop using the USB to Serial connector or to the micro-controller using the RS232 to TTL converter. Once you connect the SIM800A modem using the USB to RS232 connector, you need to find the correct COM port from the Device Manger of the USB to Serial Adapter. Then you can open Putty or any other terminal software and open an connection to that COM port at 9600 baud rate, which is the default baud rate of this modem. Once a serial connection is open through the computer or your micro-controller you can start sending the AT commands. When you send AT command for example: “AT\r” you should receive back a reply from the SIM800A modem saying “OK” or other response depending on the command send.

## **2.1.1 Specifications and Features :**

1. Quad band 850/900/1800/1900MHz.
2. GPRS class 2/10.
3. Input Voltage : 9V-12V DC.
4. Dimensions : 100 x 53 x 15 (L x W x H)mm.
5. Weight : 40 gm.
6. Low power.
7. Operating temperature : 40 – 85C.
8. Control via AT commands (3GPP TS 27.007, 27.005 and SIMCOM enhanced AT commandset).
9. High Quality Product (Not hobby grade).
10. 5V interface for direct communication with MCU kit.
11. Configurable baud rate.
12. Built in SIM Card holder.
13. Built in Network Status LED.
14. Inbuilt Powerful TCP/IP protocol stack for internet data transfer over GPRS

**2.1.2 ARDUINO UNO:**



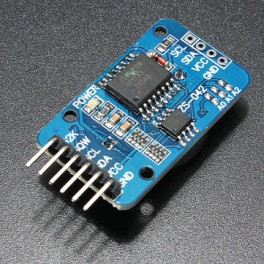
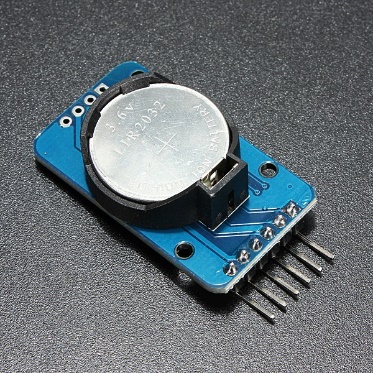
|  |
| --- |
| **Features:**   1. 13 digital pins and 6 analog pins 2. 32 KB of flash memory 3. ICSP connector for bypassing the USB port and interfacing the Arduino directly 4. on-board LED attached to digital pin 13 5. button to reset the program on the chip. 6. can be powered directly off of a USB port without any external power. 7. easy USB interface 8. Very convenient power management and built-in voltage regulation |

**2.1.3 CHANNEL RELAY(10A 250V):**



* [17]Equiped with high-current relay, AC250V 10A ; DC30V 10A
* 5V 4-Channel Relay interface board, and each one needs 50-60mA Driver Current
* Be able to control various appliances, and other equipments with large current
* Indication LED's for Relay output status

**2.1.4 DS3231 RTC MODULE :**

**Description:**  
DS3231 is a low cost, extremely accurate I2C real time clock (RTC), with a crystal oscillator with integrated temperature compensation (TCXO) and crystal.  
The device incorporates a battery inlet, disconnect the main power supply and maintains accurate timing. Built-in oscillator improve the device's long-term accuracy and reduces the number of components in the production line. The DS3231 is available in both commercial and industrial temperature ranges, using a 300mil to 16 pin .DS3231 AT24C32 IIC RTC box maintains seconds, minutes, hours, days, date, month, and year information. Less than 31 days of the month, the end date will be adjusted automatically, including corrections for the leap year. The clock operates either within 24 hours or AM / PM band / indication of the 12-hour format. Provides two configurable alarm clock and a calendar can be set to a square wave output. Address and data are serialized by a bidirectional I2C bus.A compensated precision voltage reference and comparator circuit monitors the status of the VCC to detect power outages, provide a reset output, and, if necessary, automatically switch to backup power. In addition, / RST pin is monitored as generating uP manually reset. DS3231 AT24C32 IIC

**Functionality:**  
  
Save time and higher accuracy, DS3231 also has some other features that extend the host system with additional features and a range of options. The device incorporates a very precise digital temperature sensor via the I2C interface \* to access it (at the same time). This accuracy of the temperature sensor is ± 3 ° C on chip power control circuit can automatically detect and manage the main power and backup (ie, low voltage battery) to switch between power supply. If the main power failure, the device can continue to provide accurate timing and temperature, the performance is not affected. When the main statements can restore the power supply or voltage value within the allowed range, the RESET on-chip function can be used to restart the system's microprocessor. DS3231 AT24C32 IIC

**Specification: DS3231 AT24C32 IIC**

* Size: 38mm (length) x 22mm (Width) x 14mm (height)
* Weight: 8g
* Operating voltage: 3.3 - 5.5V
* Clock chip: high-precision clock DS3231
* Accuracy Clock: 0-40 ° C range, the accuracy 2ppm, the error was about 1 minute
* Calendar alarm clock with two
* Programmable square-wave output
* Real time clock generator seconds, minutes, hours, day, date, year and year valid 2100 leap year compensation
* Chip temperature sensor with an accuracy of ± 3 ° C
* Memory chips: AT24C32 (32K storage capacity)
* IIC bus interface, the maximum transmission speed of 400KHz (working voltage of 5V)
* Can be cascaded with other IIC device, 24C32 addresses can be shorted A0 / A1 / A2
* modify default address is 0x57

**Wiring instructions (with Arduino uno r3 for example):**

* SCL → A5
* SDA → A4
* VDC → 5V
* GND → GND

**2.2 SOFTEARES USED**

1.Arduino IDE [19]:

The Arduino [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) is a [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) application (for [Windows](https://en.wikipedia.org/wiki/Windows), [macOS](https://en.wikipedia.org/wiki/MacOS), [Linux](https://en.wikipedia.org/wiki/Linux)) that is written in the programming language [Java](https://en.wikipedia.org/wiki/Java_(programming_language)). It originated from the IDE for the languages [*Processing*](https://en.wikipedia.org/wiki/Processing_(programming_language))and [*Wiring*](https://en.wikipedia.org/wiki/Wiring_(development_platform)). It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, [brace matching](https://en.wikipedia.org/wiki/Brace_matching), and [syntax highlighting](https://en.wikipedia.org/wiki/Syntax_highlighting), and provides simple *one-click* mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus. The source code for the IDE is released under the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License), version 2.

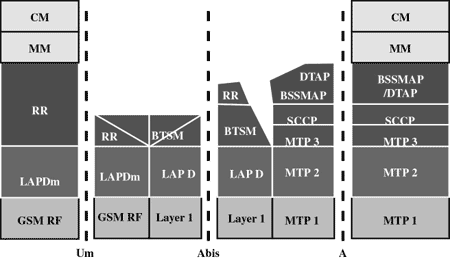
The Arduino IDE supports the languages [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B) using special rules of code structuring. The Arduino IDE supplies a [software library](https://en.wikipedia.org/wiki/Software_library) from the [Wiring](https://en.wikipedia.org/wiki/Wiring_(development_platform)) project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable [cyclic executive](https://en.wikipedia.org/wiki/Cyclic_executive) program with the [GNU toolchain](https://en.wikipedia.org/wiki/GNU_toolchain), also included with the IDE distribution. The Arduino IDE employs the program *avrdude* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

**3.Theoretical Analysis & architectures**

**GSM:**

[21] GSM architecture is a layered model that is designed to allow communications between two different systems. The lower layers assure the services of the upper-layer protocols. Each layer passes suitable notifications to ensure the transmitted data has been formatted, transmitted, and received accurately.

The GMS protocol stacks diagram is shown below:



## **MS Protocols**

Based on the interface, the GSM signaling protocol is assembled into three general layers:

* **Layer 1** : The physical layer. It uses the channel structures over the air interface.
* **Layer 2** : The data-link layer. Across the Um interface, the data-link layer is a modified version of the Link access protocol for the D channel (LAP-D) protocol used in ISDN, called Link access protocol on the Dm channel (LAP-Dm). Across the A interface, the Message Transfer Part (MTP), Layer 2 of SS7 is used.
* **Layer 3** : GSM signalling protocol’s third layer is divided into three sublayers:
  + Radio Resource Management (RR),
  + Mobility Management (MM), and
  + Connection Management (CM).

## **MS to BTS Protocols**

The RR layer is the lower layer that manages a link, both radio and fixed, between the MS and the MSC. For this formation, the main components involved are the MS, BSS, and MSC. The responsibility of the RR layer is to manage the RR-session, the time when a mobile is in a dedicated mode, and the radio channels including the allocation of dedicated channels.

The MM layer is stacked above the RR layer. It handles the functions that arise from the mobility of the subscriber, as well as the authentication and security aspects. Location management is concerned with the procedures that enable the system to know the current location of a powered-on MS so that incoming call routing can be completed.

The CM layer is the topmost layer of the GSM protocol stack. This layer is responsible for Call Control, Supplementary Service Management, and Short Message Service Management. Each of these services are treated as individual layer within the CM layer. Other functions of the CC sublayer include call establishment, selection of the type of service (including alternating between services during a call), and call release.

## **BSC Protocols**

The BSC uses a different set of protocols after receiving the data from the BTS. The Abis interface is used between the BTS and BSC. At this level, the radio resources at the lower portion of Layer 3 are changed from the RR to the Base Transceiver Station Management (BTSM). The BTS management layer is a relay function at the BTS to the BSC.

The RR protocols are responsible for the allocation and reallocation of traffic channels between the MS and the BTS. These services include controlling the initial access to the system, paging for MT calls, the handover of calls between cell sites, power control, and call termination. The BSC still has some radio resource management in place for the frequency coordination, frequency allocation, and the management of the overall network layer for the Layer 2 interfaces.

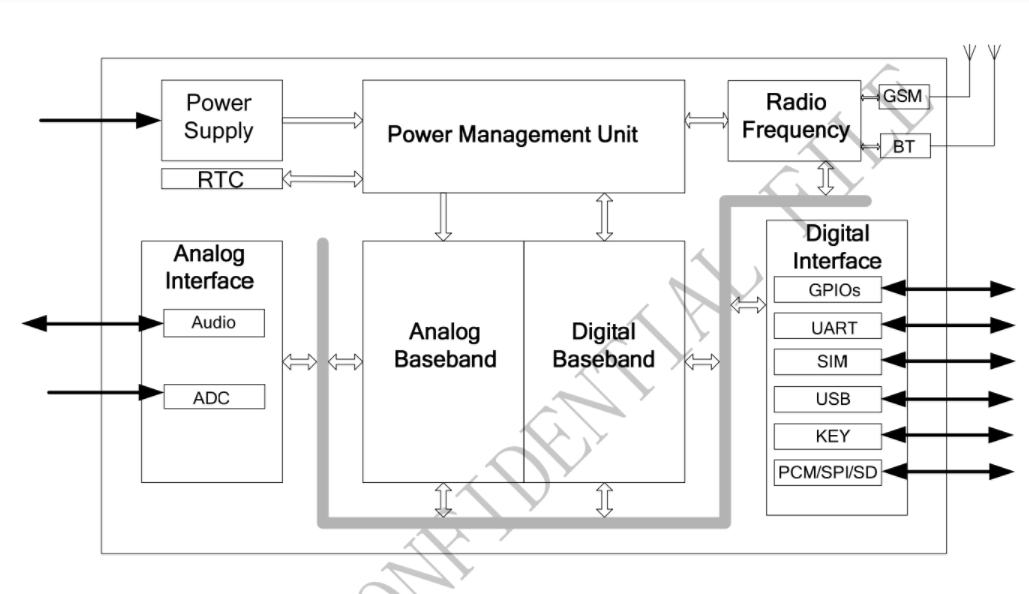
To transit from the BSC to the MSC, the BSS mobile application part or the direct application part is used, and SS7 protocols is applied by the relay, so that the MTP 1-3 can be used as the prime architecture.

## **MSC Protocols**

At the MSC, starting from the BSC, the information is mapped across the A interface to the MTP Layers 1 through 3. Here, Base Station System Management Application Part (BSS MAP) is said to be the equivalent set of radio resources. The relay process is finished by the layers that are stacked on top of Layer 3 protocols, they are BSS MAP/DTAP, MM, and CM. This completes the relay process. To find and connect to the users across the network, MSCs interact using the control-signalling network. Location registers are included in the MSC databases to assist in the role of determining how and whether connections are to be made to roaming users.

Each GSM MS user is given a HLR that in turn comprises of the user’s location and subscribed services. VLR is a separate register that is used to track the location of a user. When the users move out of the HLR covered area, the VLR is notified by the MS to find the location of the user. The VLR in turn, with the help of the control network, signals the HLR of the MS’s new location. With the help of location information contained in the user’s HLR, the MT calls can be routed to the user.

SIM800 functional diagram



**AT commands.**

AT commands are instructions used to control a modem. AT is the abbreviation of ATtention. Every command line starts with "AT" or "at". That's why modem commands are called AT commands. Many of the commands that are used to control wired dial-up modems, such as ATD (Dial), ATA (Answer), ATH (Hook control) and ATO (Return to online data state), are also supported by GSM/GPRS modems and mobile phones. Besides this common AT command set, GSM/GPRS modems and mobile phones support an AT command set that is specific to the GSM technology, which includes SMS-related commands like AT+CMGS (Send SMS message), AT+CMSS (Send SMS message from storage), AT+CMGL (List SMS messages) and AT+CMGR (Read SMS messages)

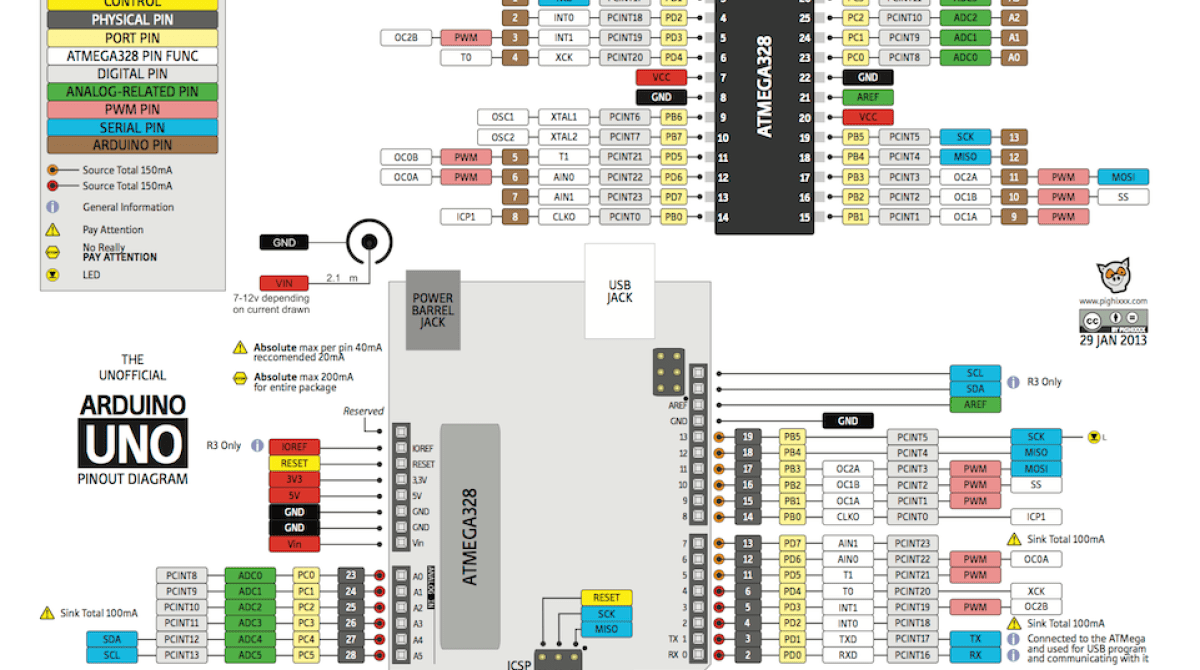
We are using AT commands for interfacing with Arduino. AT commands used for massaging are as fallows.:

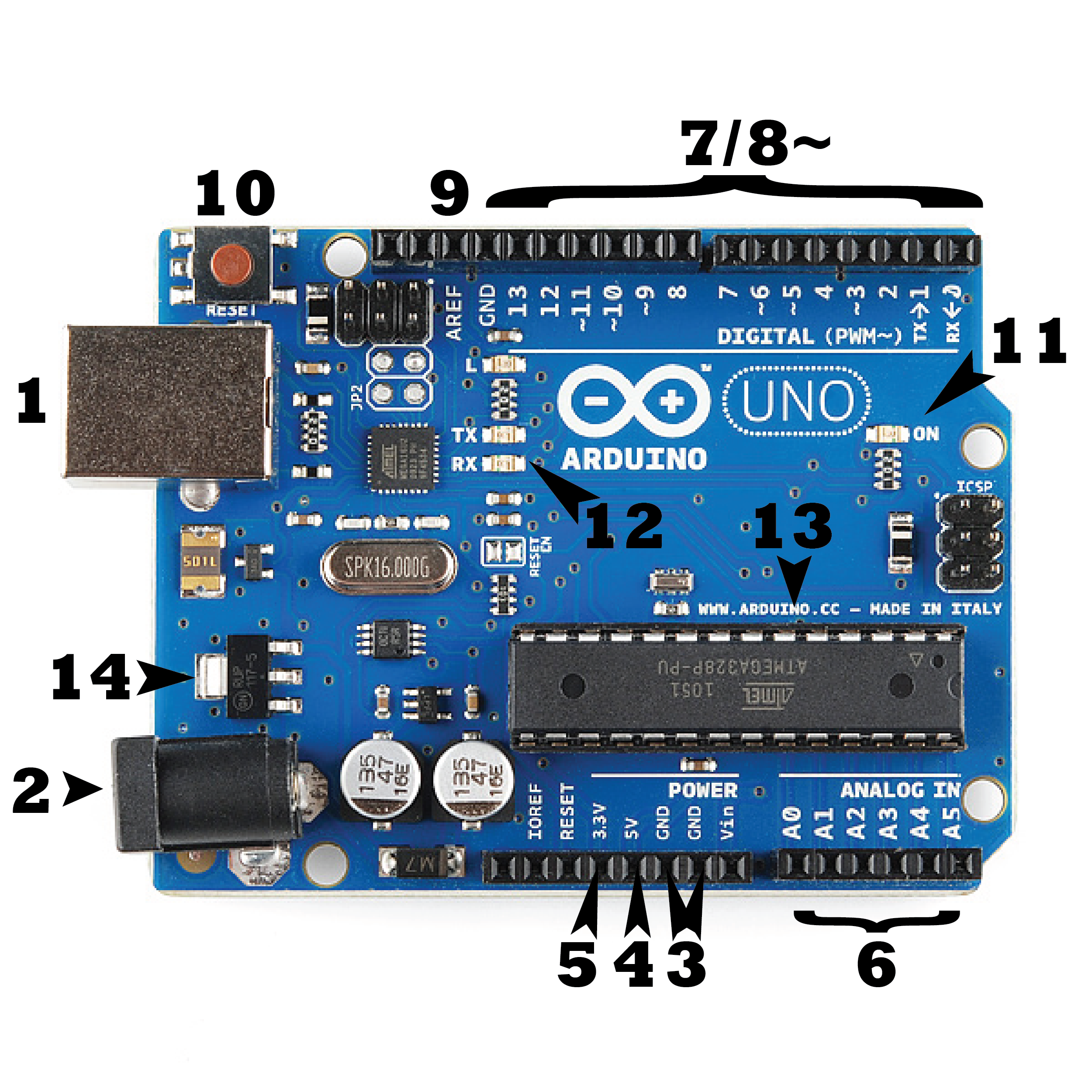
**Message**

To use SMS service following AT commands are used.

|  |  |  |
| --- | --- | --- |
| Command | Description | Response |
| AT+CMGF=<index>  index- 0: PDU       1: Text | Select message format | OK |
| AT+CMGS=”9881xxxxxx” | Send message | > ”Type message here” press ‘ctrl+z’ to end msg or ‘ESC’ to exit without sending  OK |
| AT+CMGR=<index> | Read message at that index | +CMGR: “Message Header”  Message Body  OK |
| AT+CMGD=<index> | Delete message at that index | OK (if present at that index) |
| AT+CMGDA=”DEL ALL” | Delete all SMS | OK |

**Arduino:**





### **[22] Power (USB / Barrel Jack)**

Every Arduino board needs a way to be connected to a power source. The Arduino UNO can be powered from a USB cable coming from your computer or a wall power supply ([like this](https://www.sparkfun.com/products/8269)) that is terminated in a barrel jack. In the picture above the USB connection is labeled **(1)** and the barrel jack is labeled **(2)**.

The USB connection is also how you will load code onto your Arduino board. More on how to program with Arduino can be found in our [Installing and Programming Arduino](https://learn.sparkfun.com/tutorials/installing-arduino-ide) tutorial.

**NOTE:** Do NOT use a power supply greater than 20 Volts as you will overpower (and thereby destroy) your Arduino. The recommended voltage for most Arduino models is between 6 and 12 Volts.

### **Pins (5V, 3.3V, GND, Analog, Digital, PWM, AREF)**

The pins on your Arduino are the places where you connect wires to construct a circuit (probably in conjuction with a [breadboard](https://learn.sparkfun.com/tutorials/how-to-use-a-breadboard/) and some [wire](https://learn.sparkfun.com/tutorials/working-with-wire). They usually have black plastic ‘headers’ that allow you to just plug a wire right into the board. The Arduino has several different kinds of pins, each of which is labeled on the board and used for different functions.

* **GND (3)**: Short for ‘Ground’. There are several GND pins on the Arduino, any of which can be used to ground your circuit.
* **5V (4) & 3.3V (5)**: As you might guess, the 5V pin supplies 5 volts of power, and the 3.3V pin supplies 3.3 volts of power. Most of the simple components used with the Arduino run happily off of 5 or 3.3 volts.
* **Analog (6)**: The area of pins under the ‘Analog In’ label (A0 through A5 on the UNO) are Analog In pins. These pins can read the signal from an analog sensor (like a [temperature sensor](https://www.sparkfun.com/products/10988)) and convert it into a digital value that we can read.
* **Digital (7)**: Across from the analog pins are the digital pins (0 through 13 on the UNO). These pins can be used for both digital input (like telling if a button is pushed) and digital output (like powering an LED).
* **PWM (8)**: You may have noticed the tilde (~) next to some of the digital pins (3, 5, 6, 9, 10, and 11 on the UNO). These pins act as normal digital pins, but can also be used for something called Pulse-Width Modulation (PWM). We have [a tutorial on PWM](https://learn.sparkfun.com/tutorials/pulse-width-modulation), but for now, think of these pins as being able to simulate analog output (like fading an LED in and out).
* **AREF (9)**: Stands for Analog Reference. Most of the time you can leave this pin alone. It is sometimes used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

### **Reset Button**

Just like the original Nintendo, the Arduino has a reset button **(10)**. Pushing it will temporarily connect the reset pin to ground and restart any code that is loaded on the Arduino. This can be very useful if your code doesn’t repeat, but you want to test it multiple times. Unlike the original Nintendo however, blowing on the Arduino doesn’t usually fix any problems.

### **Power LED Indicator**

Just beneath and to the right of the word “UNO” on your circuit board, there’s a tiny LED next to the word ‘ON’ **(11)**. This LED should light up whenever you plug your Arduino into a power source. If this light doesn’t turn on, there’s a good chance something is wrong. Time to re-check your circuit!

### **TX RX LEDs**

TX is short for transmit, RX is short for receive. These markings appear quite a bit in electronics to indicate the pins responsible for [serial communication](https://learn.sparkfun.com/tutorials/serial-communication). In our case, there are two places on the Arduino UNO where TX and RX appear – once by digital pins 0 and 1, and a second time next to the TX and RX indicator LEDs **(12)**. These LEDs will give us some nice visual indications whenever our Arduino is receiving or transmitting data (like when we’re loading a new program onto the board).

### **Main IC**

The black thing with all the metal legs is an IC, or Integrated Circuit **(13)**. Think of it as the brains of our Arduino. The main IC on the Arduino is slightly different from board type to board type, but is usually from the ATmega line of IC’s from the ATMEL company. This can be important, as you may need to know the IC type (along with your board type) before loading up a new program from the Arduino software. This information can usually be found in writing on the top side of the IC. If you want to know more about the difference between various IC’s, reading the datasheets is often a good idea.

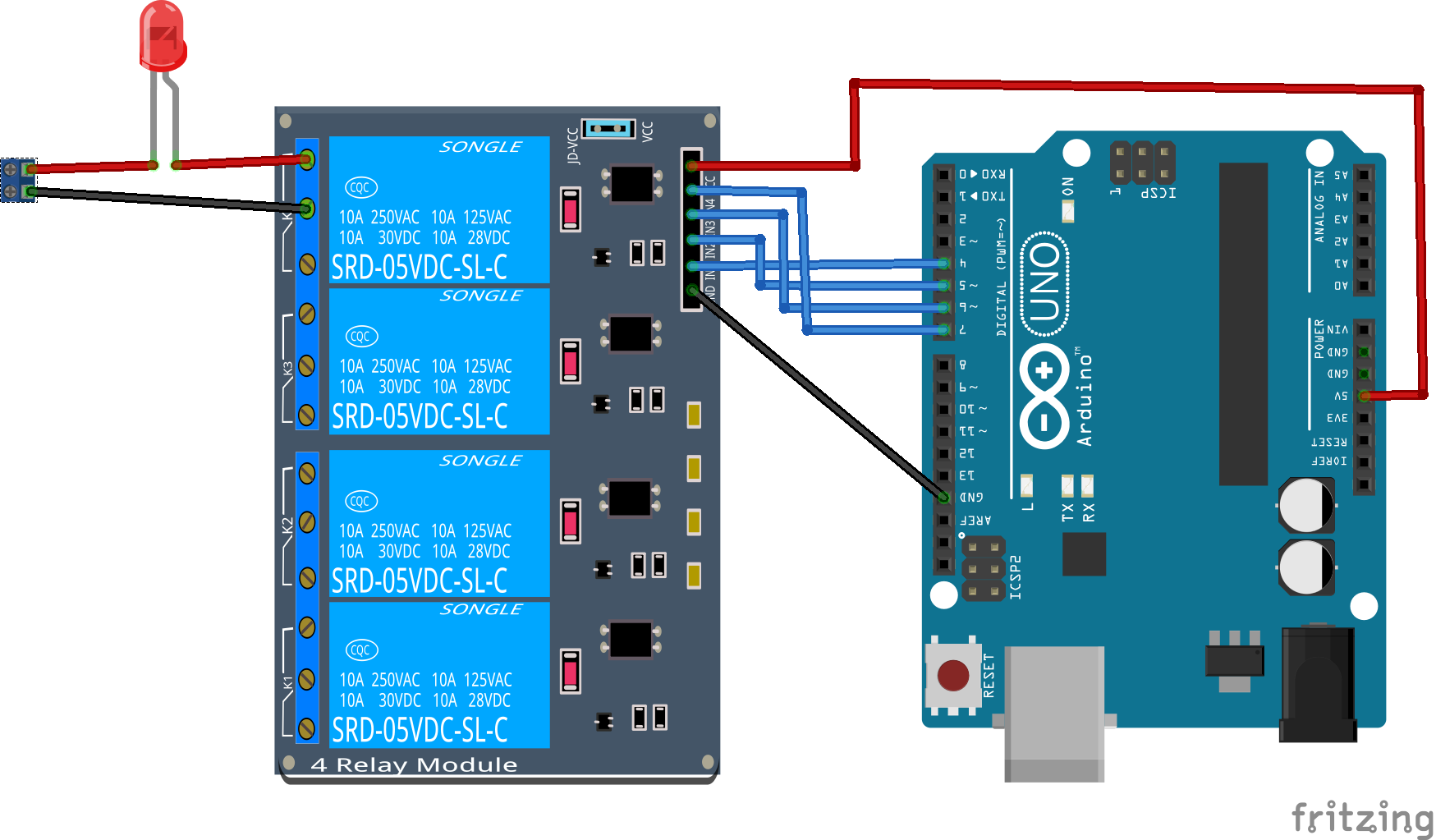
### **Voltage Regulator**

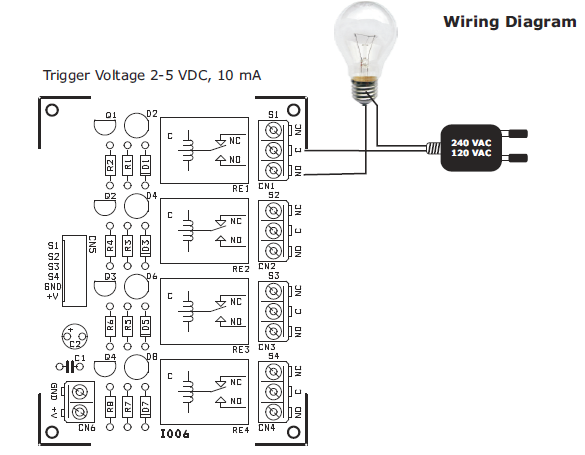
The voltage regulator **(14)** is not actually something you can (or should) interact with on the Arduino. But it is potentially useful to know that it is there and what it’s for. The voltage regulator does exactly what it says – it controls the amount of voltage that is let into the Arduino board. Think of it as a kind of gatekeeper; it will turn away an extra voltage that might harm the circuit. Of course, it has its limits, so don’t hook up your Arduino to anything greater than 20 volts.

**Relay:**

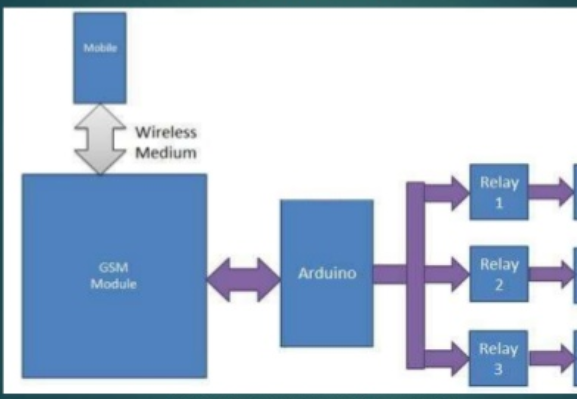
A **relay** is an [electrically](https://en.wikipedia.org/wiki/Electric) operated [switch](https://en.wikipedia.org/wiki/Switch). Many relays use an [electromagnet](https://en.wikipedia.org/wiki/Electromagnet) to mechanically operate a switch, but other operating principles are also used, such as [solid-state relays](https://en.wikipedia.org/wiki/Solid-state_relay). Relays are used where it is necessary to control a circuit by a separate low-power signal, or where several circuits must be controlled by one signal. The first relays were used in long distance [telegraph](https://en.wikipedia.org/wiki/Electrical_telegraph)circuits as amplifiers: they repeated the signal coming in from one circuit and re-transmitted it on another circuit. Relays were used extensively in telephone exchanges and early computers to perform logical operations.

Connections:





**4.Block diagram**



In this project we had connected Arduino and gsm module(sim 900a) and relay for controlling peripherals. Here we had used at commands using serial communication in Arduino for interfacing with GSM module.

**5.Procedure**

Step1:Developing the below code in Arduino ide.

Step2:Dumping the code in Arduino uno.

Step3:connect the Arduino, relay, GSM module according to instructions.

i.e (Rx of GSM to Tx of Arduino)

(Tx of GSM to Rx of Arduino)

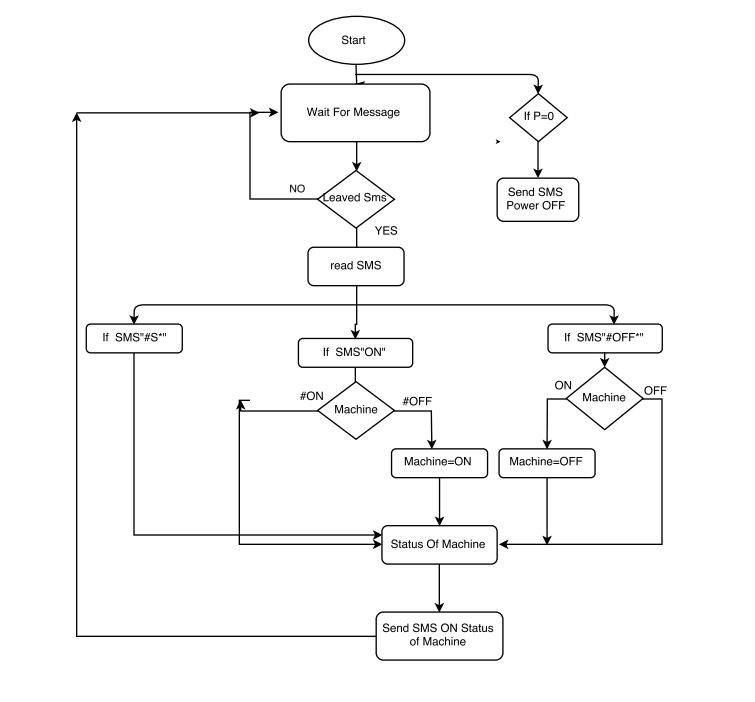
(3,4,5,6 to relay 1,2,3,4)

Step4:Insert SIM in GSM module.

Step5:Connect all power pins.

Step6:Send to test the model.

**6.Flow Diagram:**



**7.CODE**

#include <SoftwareSerial.h>

#include <Wire.h>

#include "RTClib.h"

RTC\_DS1307 rtc;

#define Fan 3

#define Light 4

#define TV 5

#define AC 6

int temp=0,i=0;

int led=13;

String fback;

char str[15];

void setup()

{

Serial.begin(9600);

pinMode(led, OUTPUT);

pinMode(Fan, OUTPUT);

pinMode(Light, OUTPUT);

pinMode(TV, OUTPUT);

pinMode(AC, OUTPUT);

digitalWrite(Fan,HIGH);

digitalWrite(Light,HIGH);

digitalWrite(TV,HIGH);

digitalWrite(AC,HIGH);

rtc.adjust(DateTime(F(\_\_DATE\_\_), F(\_\_TIME\_\_)));

Serial.println("AT+CNMI=2,2,0,0,0");

delay(500);

Serial.println("AT+CMGF=1");

delay(1000);

}

void loop()

{

if(temp==1)

{

check();

temp=0;

i=0;

delay(1000);

}

DateTime now = rtc.now();

if((now.hour()==20)&&(now.minute()==42)&&(now.second()==20)){

digitalWrite(Light, HIGH);

digitalWrite(Fan, HIGH);

digitalWrite(TV, HIGH);

digitalWrite(AC, HIGH);

fback="Rtc turned on";

SendMessage();

delay(200);

}

else if((now.hour()==20)&&(now.minute()==43)&&(now.second()==20)){

digitalWrite(Light, LOW);

digitalWrite(Fan, LOW);

digitalWrite(TV, LOW);

digitalWrite(AC, LOW);

fback="rtc turned off";

SendMessage();

delay(200);

}

}

void SendMessage()

{

Serial.println("AT+CMGF=1"); //Sets the GSM Module in Text Mode

delay(1000); // Delay of 1000 milli seconds or 1 second

Serial.println("AT+CMGS=\"+919440082570\"\r"); // Replace x with mobile number

delay(1000);

Serial.println(fback);// The SMS text you want to send

delay(100);

Serial.println((char)26);// ASCII code of CTRL+Z

delay(1000);

}

void serialEvent()

{

while(Serial.available())

{

if(Serial.find("#A."))

{

digitalWrite(led, HIGH);

delay(1000);

digitalWrite(led, LOW);

while (Serial.available())

{

char inChar=Serial.read();

str[i++]=inChar;

if(inChar=='\*')

{

temp=1;

return;

}

}

}

}

}

void check()

{

if(!(strncmp(str,"tv off",6)))

{

digitalWrite(TV, HIGH);

fback="tv is turned off";

SendMessage();

delay(200);

}

else if(!(strncmp(str,"tv on",5)))

{

digitalWrite(TV, LOW);

fback="tv is turned on";

SendMessage();

delay(200);

}

else if(!(strncmp(str,"ac off",6)))

{

digitalWrite(AC, HIGH);

fback="ac is turned off";

SendMessage();

delay(200);

}

else if(!(strncmp(str,"ac on",5)))

{

digitalWrite(AC, LOW);

fback="ac is turned on";

SendMessage();

delay(200);

}

else if(!(strncmp(str,"fan off",7)))

{

digitalWrite(Fan, HIGH);

fback="fan is turned off";

SendMessage();

delay(200);

}

else if(!(strncmp(str,"fan on",6)))

{

digitalWrite(Fan, LOW);

fback="fan is turned on";

SendMessage();

delay(200);

}

else if(!(strncmp(str,"light off",9)))

{

digitalWrite(Light, HIGH);

fback="light is turned off";

SendMessage();

delay(200);

}

else if(!(strncmp(str,"light on",8)))

{

digitalWrite(Light, LOW);

fback="light is turned on";

SendMessage();

delay(200);

}

else if(!(strncmp(str,"all off",7)))

{

digitalWrite(Light, HIGH);

digitalWrite(Fan, HIGH);

digitalWrite(TV, HIGH);

digitalWrite(AC, HIGH);

fback="all switches off";

SendMessage();

delay(200);

}

else if(!(strncmp(str,"all on",6)))

{

digitalWrite(Light, LOW);

digitalWrite(Fan, LOW);

digitalWrite(TV, LOW);

digitalWrite(AC, LOW);

fback="all switches on";

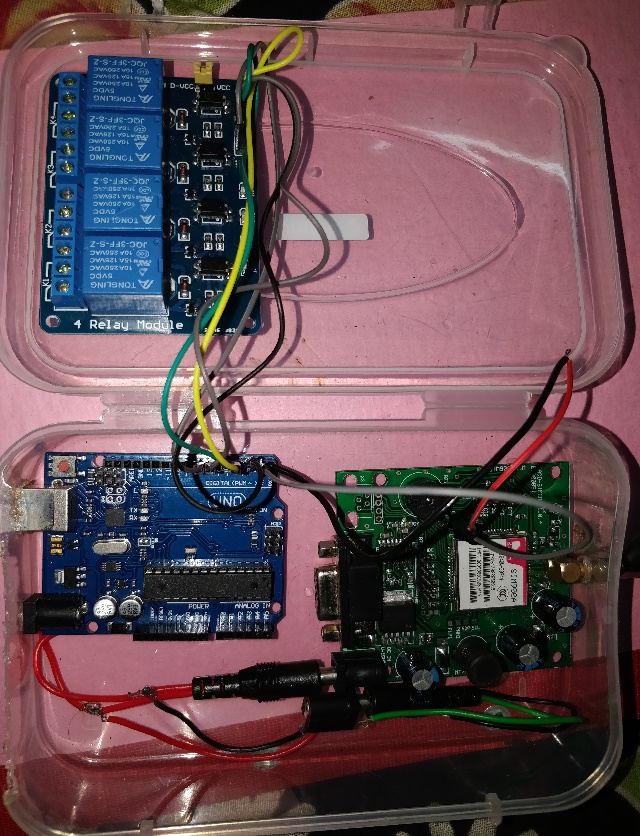
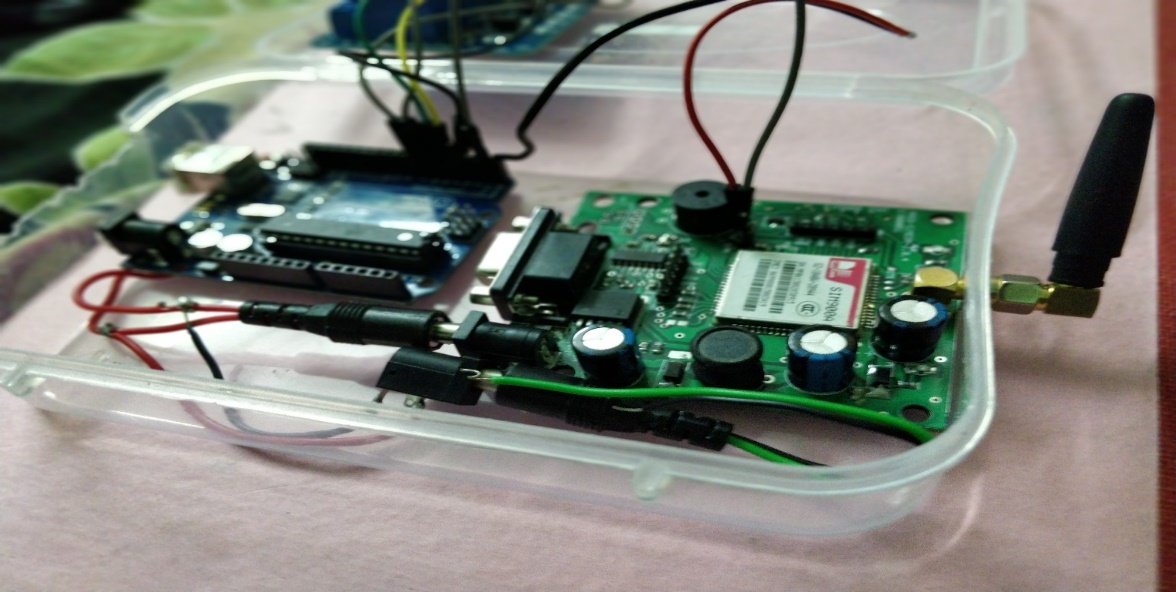
SendMessage();

delay(200);

}

}

**8.Discussion of Results**

This system is consists of GSM transceiver and receiver.It has been developed as a hardware module and it is test in a real time environment. User can just install this hardware kit in their agriculture field and take a hassle free rest by remotely monitoring all the data’s through their mobile phone, even they can control the watering system through their phones from any place.****  **10.REFERENCES**

1. S. I. Azid and S. Kumar. (2011). Analysis and Performance of a Low Cost SMS Based Home Security System. *International Journal of Smart Home*, *5(****3****)*, pp.15-24.
2. H. ElKamchouchi and A. ElShafee, “Design and prototype implementation of SMS based home automation system”, in *Proc. 2012 IEEE International Conference on Electronics Design, Systems and Applications (ICEDSA)*, Kuala Lumpur, pp.162-167.
3. M. S. Khandare and A. Mahajan, “Mobile Monitoring System for Smart Home,” in *Proc. 2010 3rd International Conference on Emerging Trends in Engineering and Technology (ICETET)*, Goa, pp.848 – 852.
4. G. Gu and G. Peng, “The survey of GSM wireless communication system,” in *Proc. 2010 International Conference on Computer and Information Application (ICCIA)*, Tianjin, pp.121 – 124.
5. G. Mingming, Shaoliangshan, Huixiaowei, and Sunqingwei, “The System of Wireless Smart House Based on GSM and ZigBee,” in *Proc. 2010* *International Conference on Intelligent Computation Technology and Automation (ICICTA)*, Changsha, pp.1017 – 1020.
6. C. Felix and I. J. Raglend, “Home automation using GSM,” in *Proc. 2011 International Conference on Signal Processing, Communication, Computing and Networking Technologies (ICSCCN)*, Thuckafay, pp.15-19.
7. C. Peijiang and J. Xuehua, “Design and Implementation of Remote Monitoring System Based on GSM,” in *Proc. 2008 Pacific-Asia Workshop on Computational Intelligence and Industrial Application (PACIIA '08)*, Wuhan, pp.678 – 681.
8. B. S. Rao, S. D. V. Prasad, and R.M. Mohan, “A prototype for Home Automation using GSM technology,” in *Proc. 2010 International Conference on Power, Control and Embedded Systems (ICPCES)*, Allahabad, pp.1-4.
9. C. K. Das, M. Sanaullah, H. M. G. Sarower, and M. M. Hassan, (2009). Development of a Cell Phone based Remote Control System: an Effective Switching System for Controlling Home and Office Appliances. *International Journal of Electrical & Computer Sciences*, *9(****10****)*, pp.23-29.
10. K. Gill, S. -H. Yang, F. Yao, and X. Lu. (2009). A zigbeebased home automation system. *IEEE Transactions on Consumer Electronics*, *55(****2****)*, pp.422 – 430.
11. L. Pu, “An Improved Short Message Security Protocol for Home Network,” in *Proc. 2009 International Conference on Future Computer and Communication (FCC '09)*, Wuhan, pp.62–65.
12. T. Perumal, M. N. Sulaiman, Khaironi Yatim Sharif, A. R. Ramli, and C. Y. Leong. (2013). Development of an Embedded Smart Home Management Scheme. *International Journal of Smart Home*, *7(****2****)*, pp.15 – 26.
13. L. Wei, Y. Min, C. LiangLiang, and C. Ping, “The Design of Intelligent Household Control System Based on Internet and GSM,” in *Proc. 2011 Second International Conference on Networking and Distributed Computing (ICNDC)*, Beijing, pp.254–256.
14. <https://robu.in/product/sim800a-quad-band-gsmgprs-module-rs232-interface/>
15. Rozita Teymourzadeh*, CEng, Member IEEE/IET*, Salah Addin Ahmed, Kok Wai Chan, and Mok Vee Hoong “Smart GSM Based Home Automation System “.

*2013 IEEE Conference on Systems, Process & Control (ICSPC2013), 13 - 15 December 2013, Kuala Lumpur, Malaysia*

[16] <https://www.lelong.com.my/arduino-uno-r3-microcontroller-development-board-> usb-cable-fun4u-191101266-2018-04-Sale-P.htm

[17] <https://www.amazon.com/JBtek-Channel-Module-Arduino-Raspberry/dp/B00KTEN3TM/ref=pd_sim_147_3?_encoding=UTF8&pd_rd_i=B00KTEN3TM&pd_rd_r=C8CP7Z355B3HKMAY9B84&pd_rd_w=Gph8h&pd_rd_wg=Hiaz7&psc=1&refRID=C8CP7Z355B3HKMAY9B84>

[18] <http://a2itronic.ma/fr/arduino/1049-ds3231-at24c32-iic-horloge-temps-reel-compatible-arduino-raspberry.html>

[19] https://en.wikipedia.org/wiki/Arduino#IDE

[20] <http://www.rhydolabz.com/wiki/?p=880>

[21] <https://www.tutorialspoint.com/gsm/gsm_billing.htm>

[22] https://learn.sparkfun.com/tutorials/what-is-an-arduino