

Miniproject based on Arduino

A Report on Miniproject Submitted for the requirement of

University of Mumbai

The practical work done during Semester-V in

**Miniproject I
(Electronics and Telecommunication Engineering)**

by

Shubham Tawade (15ET1105)

Riddesh Sonawane (15ET1043)

Vishal Singh (15ET1138)

Under the Guidance of

Mrs. Jyoti Vengurlekar



Department of Electronics and Telecommunication Engineering
Ramrao Adik Institute of Technology,
Sector 7, Nerul , Navi Mumbai
(Affiliated to University of Mumbai)
October 2017



Ramrao Adik Education Society's
Ramrao Adik Institute of Technology
(Affiliated to the University of Mumbai)
Dr. D. Y. Patil Vidyanagar, Sector 7, Nerul, Navi Mumbai 400706.

Certificate

This is to certify that, the Miniproject I titled

“Bluetooth Controlled Switch ”

is a bonafide work done by

Shubham Tawade (15ET1105)

Riddesh Sonawane (15ET1043)

Vishal Singh (15ET1138)

and is submitted in the partial fulfillment of the requirement for the
degree of

Third Year of Engineering
(Electronics and Telecommunication Engineering)
to the
University of Mumbai.



Project Guide
Mrs. Jyoti Vengurlekar

Project Coordinator
Mrs. Amruta Chintawar

Head of Department
Dr. M. D. Patil

Principal
Dr. Ramesh Vasappanavara

Certificate of Approval by Examiners

This is to certify that the submission entitled for the project

“Bluetooth Controlled Switch ”

is a bonafide work done by **Shubham Tawade, Riddesh Sonawane and Vishal Singh** under the guidance of **Mrs. Jyoti Vengurlekar** . This project work has been approved for semester V in **Miniproject-I**, University of Mumbai.

Examiners:

Internal Examiner:

Examiner Name

External Examiner:

Examiner Name

Acknowledgments

We are very glad to thank to our project guide **Mrs. Jyoti Vengurlekar** and our **H.O.D., Dr.M.D.Patil** for their encouragement and tremendous guidance. Here it is very special thank feelings for our colleagues for their support and help. We have been fortunate to have received many useful suggestions from our colleagues which have greatly improved the clarity of my report. At the end special thanks to our **Principal Dr. Ramesh Vasappanavara**. We would like to appreciate suggestions and criticisms about the report from the readers.

Abstract

In today's life electrical energy has a special role. All of our appliances like washing machine, geyser, smartphones and TVs work on electrical energy. To produce electrical energy it requires a huge cost so it is our duty to use this energy efficiently. So we tried to create a circuit which helps us to do so.

Beside energy saving this circuit also protects your appliance from getting damaged. It will increase the lifetime of your device and also avoid any burning off. The microcontroller used to create the automated turn-off circuit was an Arduino NANO. This project utilizes two different components, a Bluetooth module (HC-05) and a Servo Motor. The Bluetooth and Servo motor controls the triggering of your household circuits. Bluetooth is used to send the signal which is the amount of delay to Arduino and the servo is used to switch off the button. The circuit is very simple and light in weight and hardly occupies any space. User can carry it anywhere he/her wants.

To make it very simple for use we created an Android-based application which is used to send data to hardware i.e. Arduino.

List of Figures

1.1	Arduino Nano	1
1.2	Pin Diagram of Arduino Nano	2
1.3	Arduino Software	3
2.1	Arduino Nano	5
2.2	Pin Diagram of Arduino Nano	6
2.3	Arduino Software	7
2.4	Servo Motor (Micro Servo SG90)	8
2.5	Pulse Width Modulation technique used in servo motor	9
2.6	Bluetooth Module(HC-05)	10
2.7	Schematic of bluetooth	10
2.8	Design block of app	11
2.9	Coding block of app	11
3.1	Schematic of auto turn off circuit	13
3.2	Block Diagram of auto turn off circuit	14
4.1	PCB Layout	15
4.2	15

Contents

Abstract	ii
List of Figures	iii
1 Arduino Nano	1
1.1 Hardware	1
1.1.1 PinOutof Arduino Nano	2
1.1.2 Features	3
1.2 Software	3
2 Components	5
2.1 Arduino	5
2.1.1 Hardware	5
2.1.2 Software	7
2.1.3 How is the servo controlled?	9
2.2 Bluetooth Module (HC-05)	9
2.3 MIT App Inventor	10
2.4 List of components	12
3 Working	13
3.1 Circuit diagram	13
4 Result	15
5 Conclusion	16
Bibliography	17

Chapter 1

Arduino Nano

1.1 Hardware

Most Arduino boards consist of an Atmel 8-bit AVR microcontroller (ATmega8, ATmega168, ATmega328, ATmega1280, ATmega2560) with varying amounts of flash memory, pins, and features. The 32-bit Arduino Due, based on the Atmel SAM3X8E was introduced in 2012. The boards use single or double-row pins or female headers that facilitate connections for programming and incorporation into other circuits.

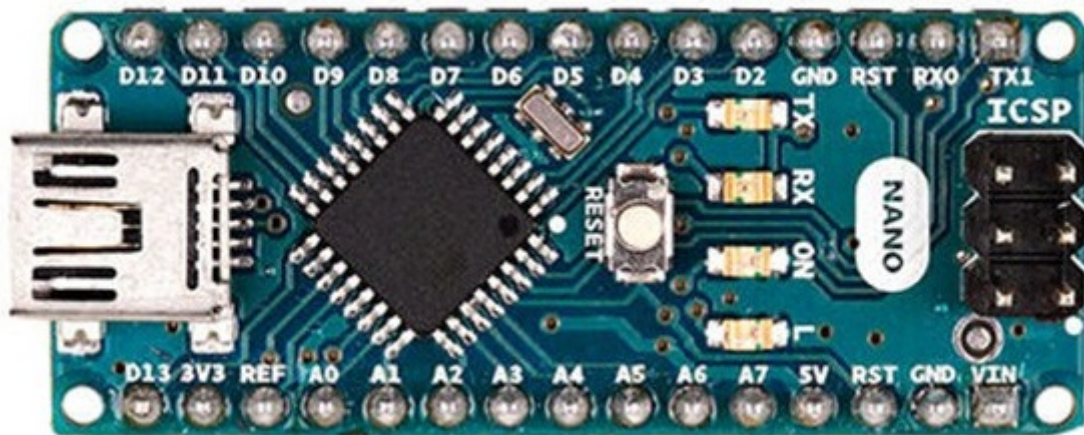


Figure 1.1: Arduino Nano

These may connect with add-on modules termed shields. Multiple, and possibly stacked shields may be individually addressable via an IC serial bus. Most boards include a 5 V linear regulator and a 16 MHz crystal oscillator or ceramic resonator. Some designs, such as the LilyPad, run at 8 MHz and dispense with the onboard voltage regulator due to specific form-factor restrictions. Arduino microcontrollers are pre-programmed with a boot loader that simplifies uploading of programs to the on-chip flash memory. The default bootloader of the Aduino UNO is the optiboot bootloader. Boards are loaded with program code via a serial connection to another computer. Some serial Arduino boards contain a level shifter circuit to convert between RS-232 logic levels and transistor-transistor logic (TTL) level signals. Current Arduino boards are programmed via Universal Serial Bus (USB), implemented using USB-to-serial adapter chips such as the

FTDI FT232. Some boards, such as later-model Uno boards, substitute the FTDI chip with a separate AVR chip containing USB-to-serial firmware, which is reprogrammable via its own ICSP header. Other variants, such as the Arduino Mini and the unofficial Boarduino, use a detachable USB-to-serial adapter board or cable, Bluetooth or other methods, when used with traditional microcontroller tools instead of the Arduino IDE, standard AVR in-system programming (ISP) programming is used.

1.1.1 PinOutof Arduino Nano

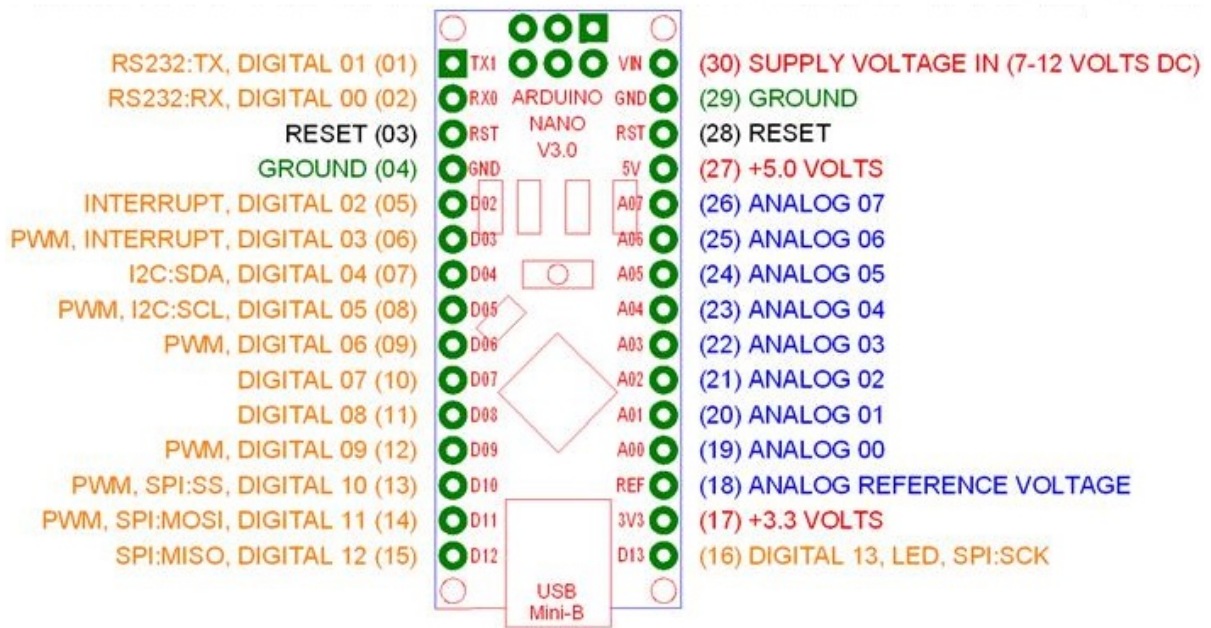


Figure 1.2: Pin Diagram of Arduino Nano

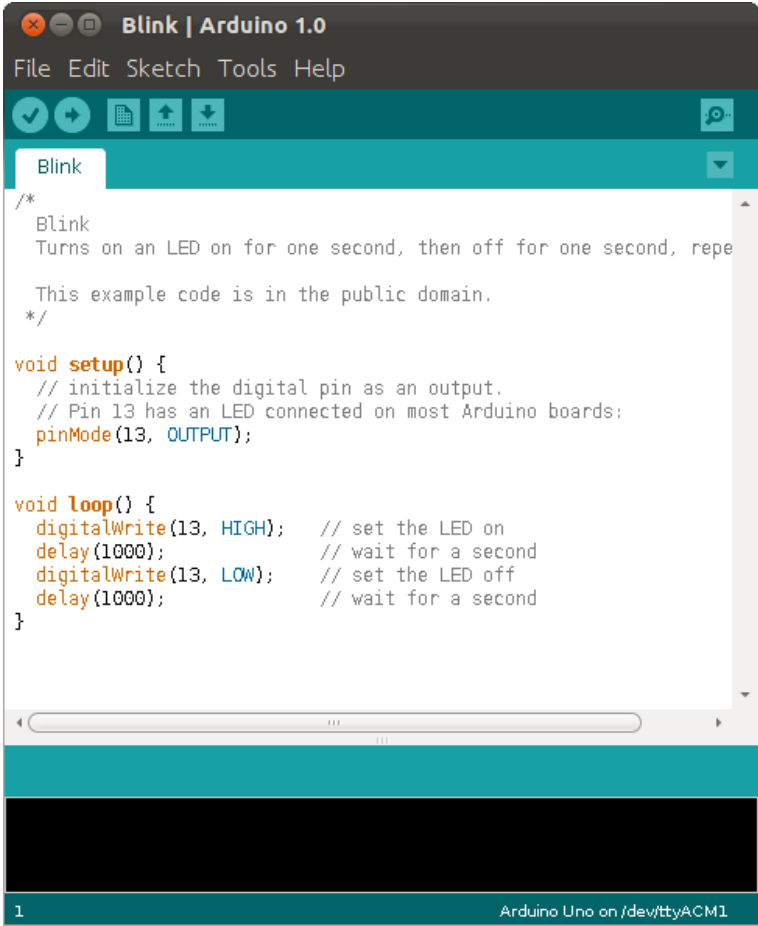
Table 1.1: Specifications of arduino

SR.NO.	Parameters	Values
1	Microcontroller	Atmel ATmega328
2	Operating Voltage (logic level)	5 V
3	Input Voltage (recommended)	7 to 12 V
4	Input Voltage (limits)	6-20 V
5	Digital I/O Pins	14 (of which 6 provide PWM output)
6	Analog Input Pins	8
7	DC Current per I/O Pin	40 mA
8	Flash Memory	32 KB (of which 2KB used by bootloader)
9	SRAM	2 KB
10	EEPROM	1 KB
11	Clock Speed	16 MHz
12	Dimensions	0.70 x 1.70

1.1.2 Features

- i Automatic reset during program download
- ii Power OK blue LED
- iii Green (TX), red (RX) and orange (L) LED
- iv Auto sensing/switching power input
- v Small mini-B USB for programming and serial monitor
- vi ICSP header for direct program download
- vii Standard 0.1 spacing DIP
- viii Manual reset switch

1.2 Software



The image shows a screenshot of the Arduino IDE software interface. The window title is "Blink | Arduino 1.0". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for saving, undo, redo, and uploading. The main text area contains the following code:

```
/*
 * Blink
 * Turns on an LED on for one second, then off for one second, repeatedly.
 * This example code is in the public domain.
 */

void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH); // set the LED on
  delay(1000);           // wait for a second
  digitalWrite(13, LOW); // set the LED off
  delay(1000);           // wait for a second
}
```

At the bottom of the window, the status bar indicates "1" on the left and "Arduino Uno on /dev/ttyACM1" on the right.

Figure 1.3: Arduino Software

A program for Arduino may be written in any programming language for a compiler that produces binary machine code for the target processor. Atmel provides a development environment for their microcontrollers, AVR Studio and the newer Atmel Studio.

The Arduino project provides the Arduino integrated development environment (IDE), which is a cross-platform application written in the programming language Java. It originated from the IDE for the languages Processing and Wiring. It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and 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.

A program written with the IDE for Arduino is called a sketch. Sketches are saved on the development computer as text files with the file extension `.ino`. Arduino Software (IDE) pre-1.0 saved sketches with the extension `.pde`.

The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring 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 program with the 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.

Chapter 2

Components

2.1 Arduino

2.1.1 Hardware

Most Arduino boards consist of an Atmel 8-bit AVR microcontroller (ATmega8, ATmega168, ATmega328, ATmega1280, ATmega2560) with varying amounts of flash memory, pins, and features. The 32-bit Arduino Due, based on the Atmel SAM3X8E was introduced in 2012. The boards use single or double-row pins or female headers that facilitate connections for programming and incorporation into other circuits.

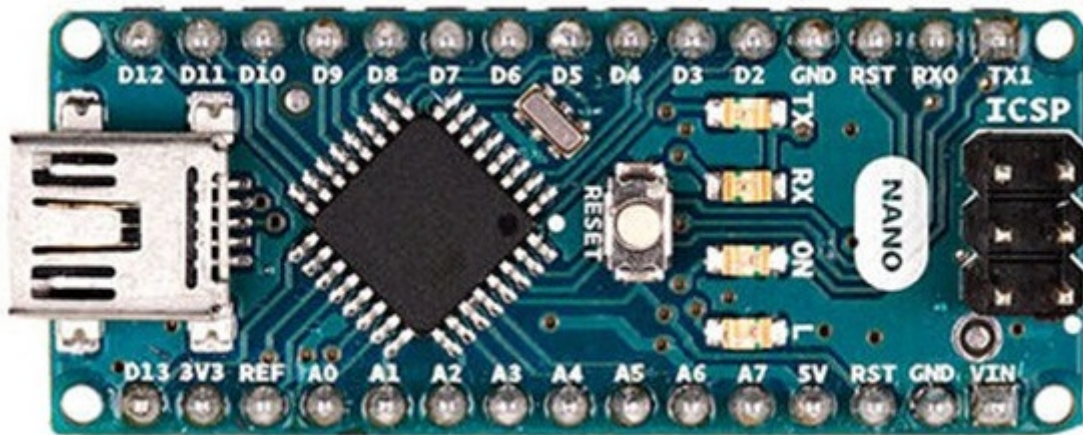


Figure 2.1: Arduino Nano

These may connect with add-on modules termed shields. Multiple, and possibly stacked shields may be individually addressable via an IC serial bus. Most boards include a 5 V linear regulator and a 16 MHz crystal oscillator or ceramic resonator. Some designs, such as the LilyPad, run at 8 MHz and dispense with the onboard voltage regulator due to specific form-factor restrictions. Arduino microcontrollers are pre-programmed with a boot loader that simplifies uploading of programs to the on-chip flash memory. The default bootloader of the Arduino UNO is the optiboot bootloader. Boards are loaded with program code via a serial connection to another computer. Some serial Arduino boards contain a level shifter circuit to convert between RS-232 logic levels and transis-

tor?transistor logic (TTL) level signals. Current Arduino boards are programmed via Universal Serial Bus (USB), implemented using USB-to-serial adapter chips such as the FTDI FT232. Some boards, such as later-model Uno boards, substitute the FTDI chip with a separate AVR chip containing USB-to-serial firmware, which is reprogrammable via its own ICSP header. Other variants, such as the Arduino Mini and the unofficial Boarduino, use a detachable USB-to-serial adapter board or cable, Bluetooth or other methods, when used with traditional microcontroller tools instead of the Arduino IDE, standard AVR in-system programming (ISP) programming is used.

PinOutof Arduino Nano

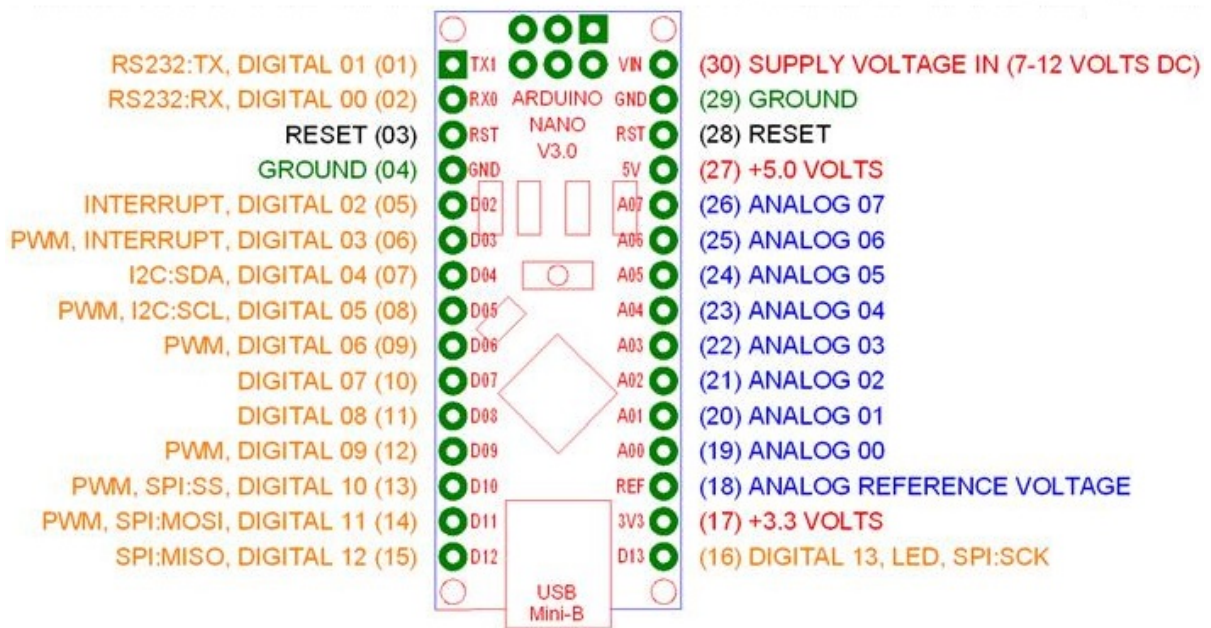


Figure 2.2: Pin Diagram of Arduino Nano

Features

- i Automatic reset during program download
- ii Power OK blue LED
- iii Green (TX), red (RX) and orange (L) LED
- iv Auto sensing/switching power input
- v Small mini-B USB for programming and serial monitor
- vi ICSP header for direct program download
- vii Standard 0.1" spacing DIP
- viii Manual reset switch

Table 2.1: Specifications of arduino

SR.NO.	Parameters	Values
1	Microcontroller	Atmel ATmega328
2	Operating Voltage (logic level)	5 V
3	Input Voltage (recommended)	7 to 12 V
4	Input Voltage (limits)	6-20 V
5	Digital I/O Pins	14 (of which 6 provide PWM output)
6	Analog Input Pins	8
7	DC Current per I/O Pin	40 mA
8	Flash Memory	32 KB (of which 2KB used by bootloader)
9	SRAM	2 KB
10	EEPROM	1 KB
11	Clock Speed	16 MHz
12	Dimensions	0.70? x 1.70?

2.1.2 Software

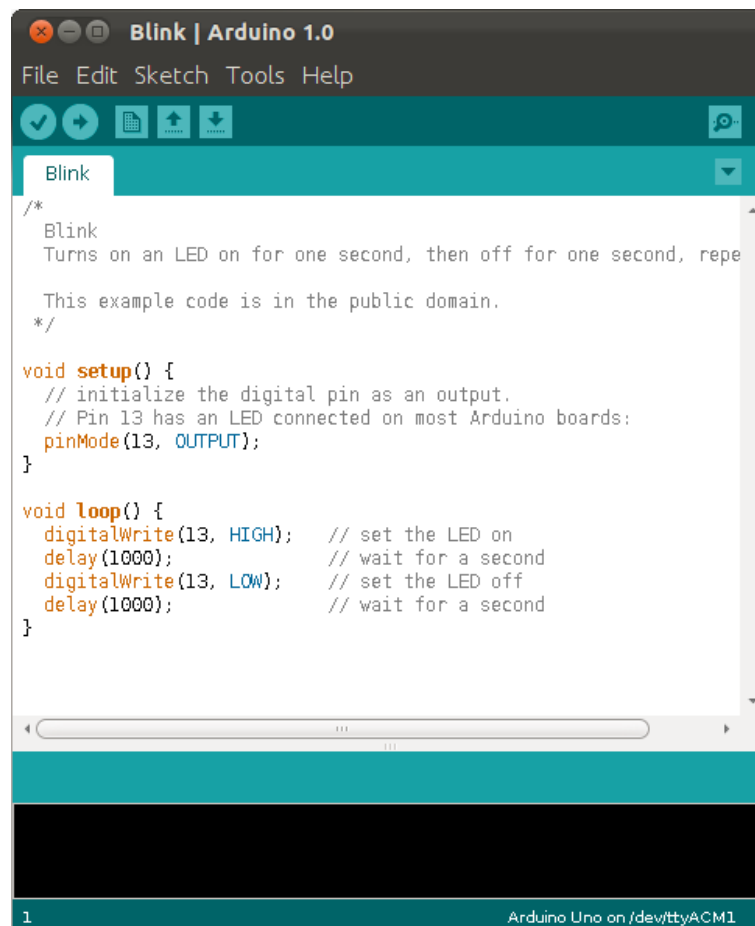


Figure 2.3: Arduino Software

A program for Arduino may be written in any programming language for a compiler that produces binary machine code for the target processor. Atmel provides a development environment for their microcontrollers, AVR Studio and the newer Atmel Studio.

The Arduino project provides the Arduino integrated development environment (IDE), which is a cross-platform application written in the programming language Java. It originated from the IDE for the languages Processing and Wiring. It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and 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.

A program written with the IDE for Arduino is called a sketch. Sketches are saved on the development computer as text files with the file extension `.ino`. Arduino Software (IDE) pre-1.0 saved sketches with the extension `.pde`.

The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring 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 program with the 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.



Figure 2.4: Servo Motor (Micro Servo SG90)

rotation of the motor is required for just a certain angle not continuously for long period of time. For these applications, some special types of motor are required with some special arrangement which makes the motor to rotate a certain angle for a given electrical input (signal). For this purpose servo motor comes into picture. This is normally a simple DC motor which is controlled for specific angular rotation with the help of additional servomechanism (a typical closed loop feedback control system). Now day's servo system has huge industrial applications.

2.1.3 How is the servo controlled?

Servos are controlled by sending an electrical pulse of variable width, or pulse width modulation (PWM), through the control wire. There is a minimum pulse, a maximum pulse, and a repetition rate. A servo motor can usually only turn 90 in either direction for a total of 180 movement. The motor's neutral position is defined as the position where the servo has the same amount of potential rotation in the both the clockwise or counter-clockwise direction. The PWM sent to the motor determines position of the shaft, and

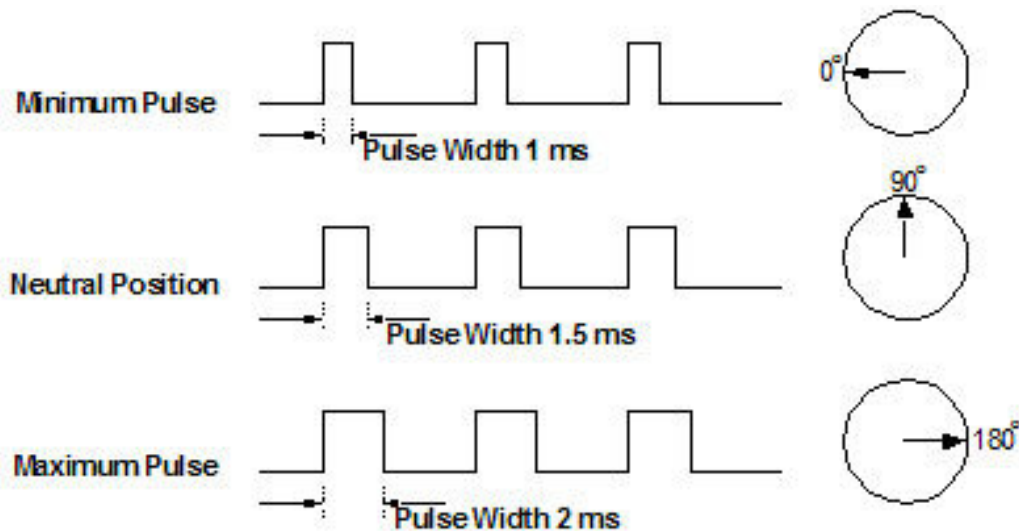


Figure 2.5: Pulse Width Modulation technique used in servo motor

based on the duration of the pulse sent via the control wire; the rotor will turn to the desired position. The servo motor expects to see a pulse every 20 milliseconds (ms) and the length of the pulse will determine how far the motor turns. For example, a 1.5ms pulse will make the motor turn to the 90 position. Shorter than 1.5ms moves it in the counter clockwise direction toward the 0 position, and any longer than 1.5ms will turn the servo in a clockwise direction toward the 180 position.

2.2 Bluetooth Module (HC-05)

HC-05 module is an easy to use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup. The HC-05 Bluetooth Module can be used in a Master or Slave configuration, making it a great solution for wireless communication.

This serial port bluetooth module is fully qualified Bluetooth V2.0+EDR (Enhanced Data Rate) 3Mbps Modulation with complete 2.4GHz radio transceiver and baseband. It uses CSR Bluecore 04 External single chip Bluetooth system with CMOS technology and with AFH (Adaptive Frequency Hopping Feature). The Bluetooth module HC-05 is a MASTER/SLAVE module. By default the factory setting is SLAVE. The Role of the module (Master or Slave) can be configured only by AT COMMANDS. The slave modules cannot initiate a connection to another Bluetooth device, but can accept connections. Master module can initiate a connection to other devices. The user can use it simply for a serial port replacement to establish connection between MCU and GPS, PC to your embedded project, etc.

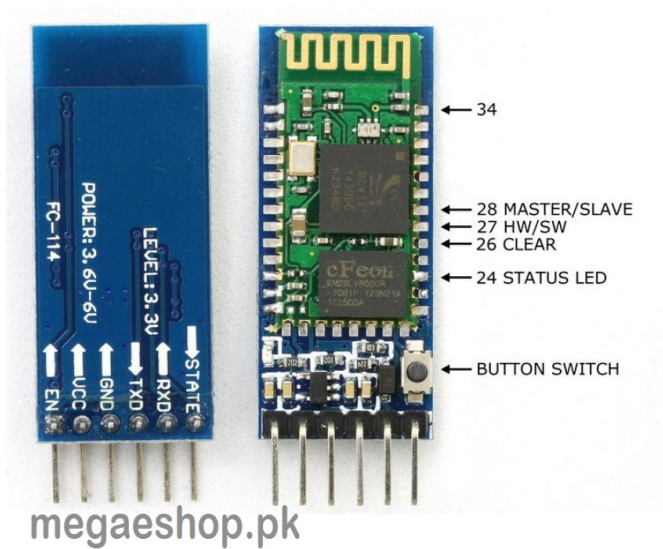


Figure 2.6: Bluetooth Module(HC-05)

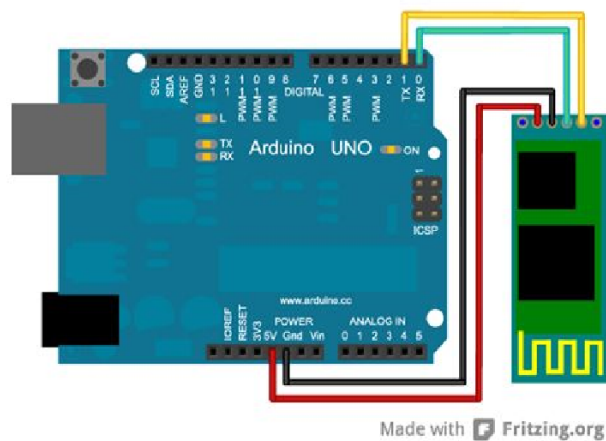


Figure 2.7: Schematic of bluetooth

As we know that Vcc and Gnd of the module goes to Vcc and Gnd of Arduino. The TXD pin goes to RXD pin of Arduino and RXD pin goes to TXD pin of Arduino i.e. (digital pin 0 and 1). The user can use the on board Led. But here, Led is connected to digital pin 12 externally for betterment of the process.

2.3 MIT App Inventor

MIT App Inventor is an intuitive, visual programming environment that allows everyone ? even children ? to build fully functional apps for smartphones and tablets. Those new to MIT App Inventor can have a simple first app up and running in less than 30 minutes. And what's more, our blocks-based tool facilitates the creation of complex,

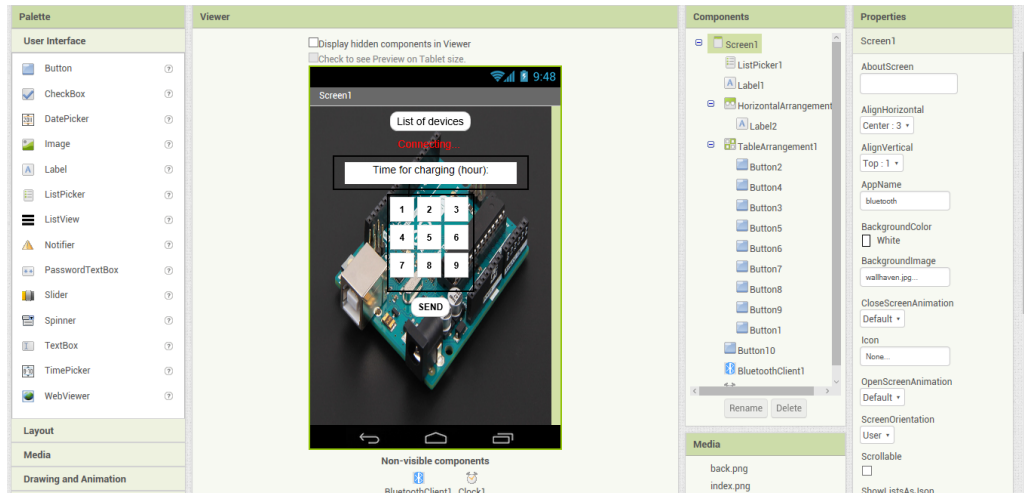


Figure 2.8: Design block of app

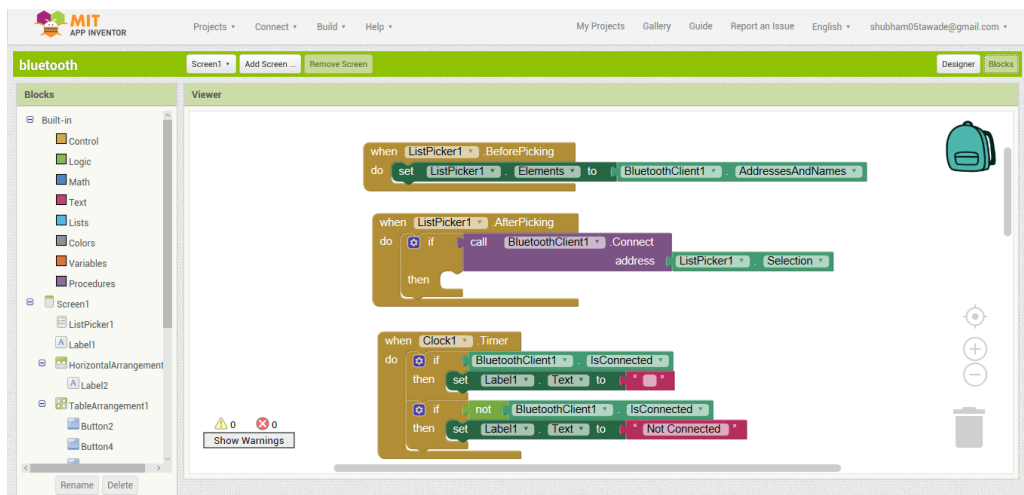


Figure 2.9: Coding block of app

high-impact apps in significantly less time than traditional programming environments. The MIT App Inventor project seeks to democratize software development by empowering all people, especially young people, to move from technology consumption to technology creation.

A small team of CSAIL staff and students, led by Professor Hal Abelson, forms the nucleus of an international movement of inventors. In addition to leading educational outreach around MIT App Inventor and conducting research on its impacts, this core team maintains the free online app development environment that serves more than 6 million registered users. . Blocks-based coding programs inspire intellectual and creative empowerment. MIT App Inventor goes beyond this to provide real empowerment for kids to make a difference – a way to achieve social impact of immeasurable value to their communities. In fact, App Inventors in school and outside of traditional educational settings have come together and done just that:

2.4 List of components

It consists of various components named below:

- Arduino Nano
- Servo Motor
- Bluetooth Module(HC-05)
- Button (1)
- DC Motor
- Jumpers
- PCB

Chapter 3

Working

3.1 Circuit diagram

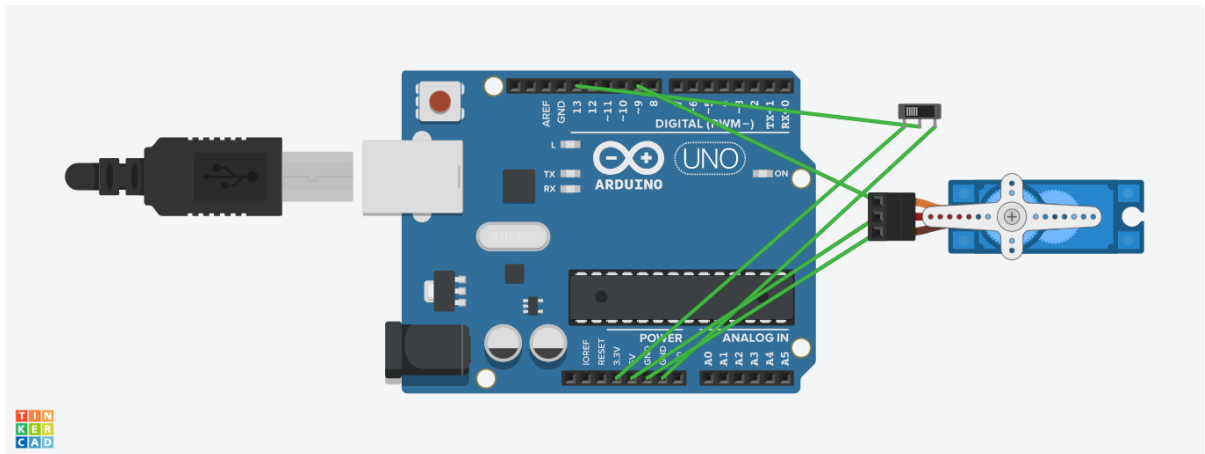


Figure 3.1: Schematic of auto turn off circuit

1. The circuit looks very simple. Arduino is connected to two components which is Servo Motor and Bluetooth Module.
2. Servo requires +5V Supply, one input from PWM pin(D10) and GND connection
3. Bluetooth has total 6 pins out of which we will use only four pins. It requires +3.3V and GND connection. It has two pins RX and TX which receive signal and transmit signal respectively.
4. To transmit the signal that is to provide delay we have used an android app which passes signal to arduino.
5. Arduino calculates the delay which is received from app. The delay operation is performed for respective time.
6. After the delay get finished servo performs rotating operation for certain angle by PWM pulses and the switch gets triggered.
7. This is how automatic triggered circuit performs.

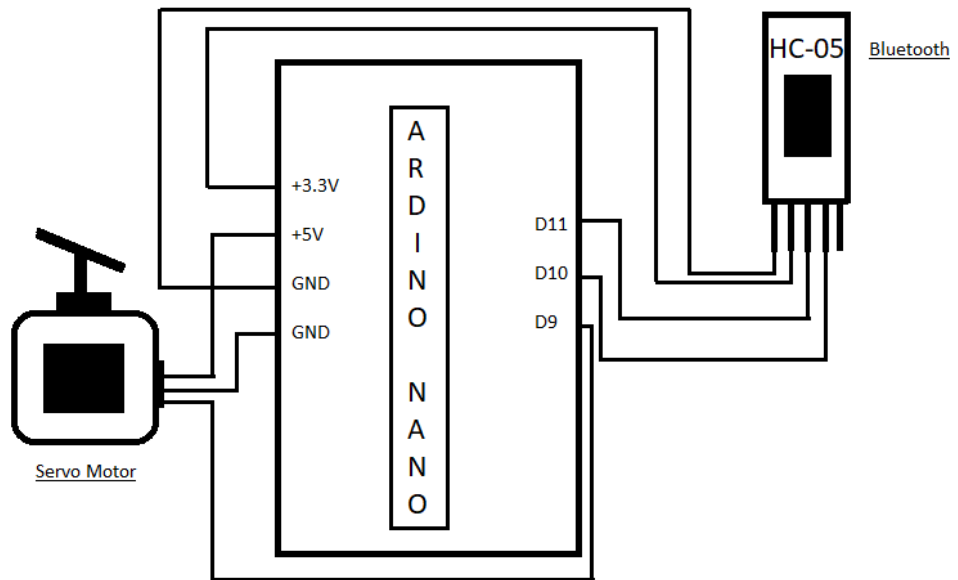


Figure 3.2: Block Diagram of auto turn off circuit

Chapter 4

Result

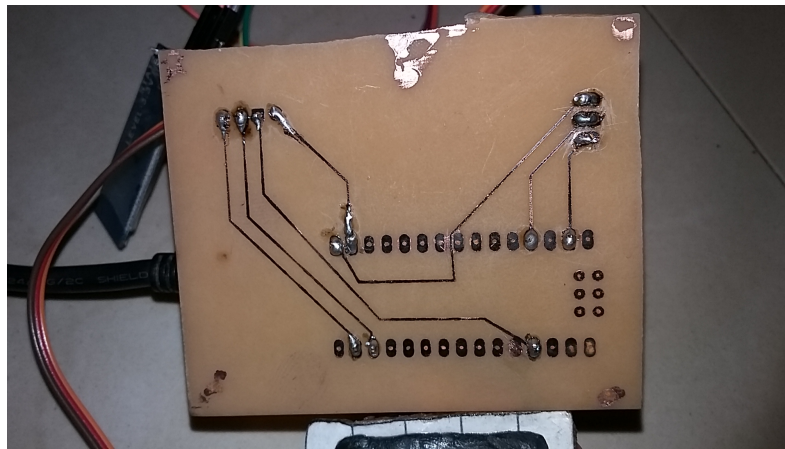


Figure 4.1: PCB Layout

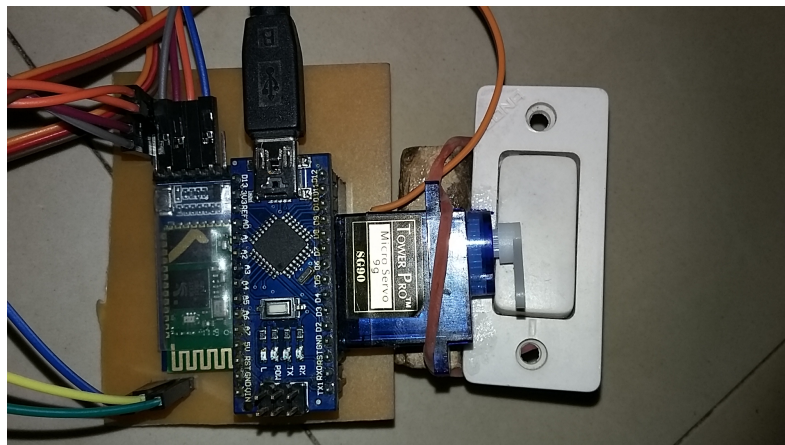


Figure 4.2:

Chapter 5

Conclusion

In terms of quality, this work holds a great quality due to various testing in each step of progress and is able to perform smooth tasks of turning off any appliance. It can be used anywhere due to its small size and light weight. Users can carry this circuit anywhere they want. To operate this circuit, it requires a very less power.

Bibliography

- [1] https://www.makerlab-electronics.com/my_uploads/2016/08/arduino-nano-1.jpg <https://www.buyapi.ca/wp-content/uploads/2015/02/SKU047142.41.jpg>
- [2] <https://www.makerfabs.com/image/cache/makerfabs/HC-05>
- [4] Technical terms from www.learnabout-electronic.org And from Wikipedia
- [5] "Book Name ", Arduino Cookbook, 2nd Edition by Michael Margolis

Bibliography