

HOME AUTOMATION SYSTEM USING ARDUINO

Dissertation submitted in partial fulfillment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

By

Archit Goel - 121084

Adviti Sharma - 121099

Swapnil - 121101

UNDER THE GUIDANCE OF

Prof. Dr. Sunil Bhooshan



JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

May, 2016

Table of Contents

DECLARATION BY THE SCHOLAR	i
SUPERVISOR'S CERTIFICATE	ii
PREFACE & ACKNOWLEDGEMENT	iii
ABSTRACT	iv
LIST OF ACRONYMS & ABBREVIATIONS	v
LIST OF FIGURES	vi
1 INTRODUCTION	1
1.1 History	1
1.2 Applications	2
1.2.1 Areas of Home Automation	3
1.2.1.1 Lighting Control: Leaving the Dark Ages and Step- ping Into the Light	3
1.2.1.2 HVAC Regulation: No Longer Burned by Your Heat- ing Bill	3
1.2.1.3 Lawn Irrigation System: The Grass is Always Greener	4
1.2.1.4 Security Systems: Knock, Knock.	4
1.2.2 Technologies	5
1.2.2.1 Bluetooth Smart	5
1.2.2.2 Wi-Fi	5
2 TECHNICAL DETAILS	7
2.1 Arduino	7
2.1.1 Features-:	8
2.1.2 Pin descriptions	10
2.1.2.1 VCC	10
2.1.2.2 GND	10
2.1.2.3 Port B (PB7:0) XTAL1/XTAL2/TOSC1/TOSC2	10
2.1.2.4 Port C (PC5:0)	11
2.1.2.5 PC6/RESET	12
2.1.2.6 Port D (PD7:0)	12
2.1.3 Power	12
2.1.4 Memory	13
2.1.5 Input and Output	13
2.1.6 Communication	14
2.1.7 USB Overcurrent Protection	15
2.1.8 Arduino v/s Microcontroller	15
2.2 Temperature Sensor LM35	16

2.2.1	Applications	17
2.2.2	Specifications:	18
2.2.3	Interfacing LM35 with Arduino	19
2.3	Bluetooth module HC-05	19
2.3.1	The key factory parameters	20
2.3.1.1	Master role	20
2.3.1.2	Pairing	21
2.3.1.3	Multi-device communication	21
2.3.1.4	Default communication baud rate	21
2.3.1.5	Consumption	21
2.3.2	Specifications:	21
2.3.3	Interfacing HC-05 with arduino	22
2.4	IR Sensor	22
2.4.1	IR LED	23
2.4.2	Photodiode	23
2.4.3	Working Principle	24
2.4.3.1	Planck's Radiation Law	25
2.4.3.2	Stephan Boltzmann Law	25
2.4.3.3	Wein's Displacement Law	25
2.4.4	Interfacing with Arduino	26
2.5	Light Sensor	26
2.5.1	Construction of a Photocell	26
2.5.2	Working Principle	27
2.5.3	Types of Light Dependent Resistors	27
2.5.3.1	Intrinsic photo resistors (Un doped semiconductor)	28
2.5.3.2	Extrinsic photo resistors	28
2.5.4	Interfacing with Arduino	28
2.6	Relay	28
2.6.1	Working Principle	33
2.6.2	Interfacing with Arduino	34
3	IMPLEMENTATION	35
3.1	Software Requirements	35
3.1.1	Arduino v1.6.0	35
3.1.1.1	Inbuilt Functions:	35
3.1.1.2	User Defined Functions:	39
3.1.2	Android IDE	39
3.2	Code	39
3.3	Hardware Requirements	48
4	RESULT	49
4.1	Theoretical Results	49
4.2	Pictorial Results	50
4.2.1	Plots	50
4.2.2	Snapshots of the model	51
5	CHALLENGES AND LIMITATIONS	54

6 FUTURE SCOPE

56

Bibliography

57

DECLARATION BY THE SCHOLARS

We hereby declare that the work reported in the B-Tech. thesis entitled "**Home Automation System using Arduino**" submitted at **Jaypee University of Information Technology, Wagnaghat, India** is an authentic record of our work carried out under the supervision of **Prof. Dr. Sunil Bhooshan**. We have not submitted this work elsewhere for any other degree or diploma.

Archit Goel



Adviti Sharma



Swapnil



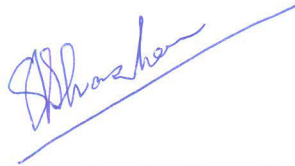
Department of Electronics and Communication Engineering

Jaypee University of Information Technology, Wagnaghat, India

May 20, 2016

SUPERVISOR'S CERTIFICATE

This is to certify that the work reported in the B-Tech. thesis entitled "**Home Automation System using Arduino**", submitted by **Archit Goel, Adviti Sharma** and **Swapnil** at **Jaypee University of Information Technology, Wagnaghat, India**, is a bonafide record of their original work carried out under my supervision. This work has not been submitted elsewhere for any other degree or diploma.



Prof. Dr. Sunil Bhooshan

Head of Department
Electronics and Communication Engineering
Jaypee University of Information Technology, Wagnaghat, India

May 20, 2016

PREFACE & ACKNOWLEDGEMENT

It is a great opportunity for us to have the Bachelor of Technology in Jaypee University of Information Technology, Waknaghat. In the completion of this degree, we are submitting a thesis on “Home Automation System Using Arduino”.

We got to learn many new things about the technology and its practical implementation. This Project proved to be a milestone in our knowledge of present environment. Every say and every moment was an experience in itself, an experience which theoretical study can't provide.

We are very grateful and highly acknowledge the continuous encouragement, invaluable supervision, timely suggestions and inspired guidance offered by our project supervisor **Prof. Dr. Sunil Bhooshan**, Professor and Head, Department of Electronics and Communication Engineering, Jaypee University of Information Technology, Waknaghat in bringing this project to a successful completion.

We would also like to thank Mr. Mohan Sharma and Mr. Pandey, for their consistent support during their project.

The zeal to accomplish the task of formulating the project could not have been realized without the support and cooperation of faculty members of ECE Department. We are sincerely thankful to **Prof. Dr. T. S. Lamba**, Dean (Academic and Research) for his consistent support and encouragement throughout the project work.

ABSTRACT

Automated “homes of the future” have been staple exhibits for World’s Fairs and popular backgrounds in science fiction. However, problems with complexity, competition between vendors, multiple incompatible standards, and the resulting expense have limited the penetration of home automation to homes of the wealthy, or ambitious hobbyists. Possibly the first home “computer” was an experimental home automation system in 1966.

We have used Arduino which is a general purpose, low-cost, programmable, portable, easy to implement, low power consuming and high performance hardware operating on 5V and an output current of a few milli amperes.

We have mainly focused on the areas of Energy management and Lighting. We would like to give a glimpse of what the basic idea of home automation is. Using this we can easily maintain comfort while saving energy: automate lighting, temperatures based on schedules and events. In order to achieve this, a Bluetooth module is interfaced to the Arduino board at the receiver end while on the transmitter end, a GUI application on the cell phone sends ON/OFF commands to the receiver where loads are connected. By touching the specified location on the GUI, the loads can be turned ON/OFF remotely through this technology. The loads will be operated by Arduino board through relays.

LIST OF ACRONYMS & ABBREVIATIONS

ADC	Analog to Digital Converter
ASSR	Auditory Steady State Response
AVR	Advanced Virtual Reduced Instruction Set Computer
EEPROM	Electrically Erasable Programmable Read Only Memory
GFSK	Gaussian Frequency Shift Keying
ICSP	In Circuit Serial Programming
IDE	Integrated Development Environment
IR	Infrared
LED	Light Emitting Diode
MIPS	Microprocessor without Interlocked Pipeline Stages
PDIP	Plastic Dual In-Line Packages
PWM	Pulse Width Modulation
RC	Remote Controlled
RISC	Reduced Instruction Set Computer
SPI	Serial Peripheral Interface
SRAM	Static Random Access Memory
TTL	Transistor Transistor Logic
TQFP	Thin Quad Flat Package
USART	Universal Synchronous/Asynchronous Receiver/Transmitter
USB	Universal Serial Bus

LIST OF FIGURES

2.1	Atmega 328 to Arduino Pin Mapping	8
2.2	Atmega 328 Pinout	11
2.3	Temperature Sensor-LM35	18
2.4	Interfacing LM35 with Arduino	19
2.5	Bluetooth Module HC-05	19
2.6	Bluetooth Module Pin out	22
2.7	Interfacing HC-05 with arduino	23
2.8	IR Sensor Circuit	24
2.9	Interfacing IR sensor with Arduino	26
2.10	Construction of an LDR	27
2.11	Interfacing LDR using Arduino	28
2.12	Electromechanical Relay	30
2.13	Solid State Relay	31
2.14	What a relay looks like	33
2.15	Interfacing Relay using Arduino	34
4.1	IR Sensor Voltage as a function of the distance	50
4.2	LDR voltage value as a function of output voltage	50
4.3	Bluetooth command for room 2 lights	51
4.4	Room 1 lights controlled by Bluetooth	51
4.5	Bluetooth command for room 2 lights	52
4.6	Room 2 lights controlled by Bluetooth	52
4.7	LDR reading on the Serial Monitor	53
4.8	LDR controlled room lights	53

Chapter 1

INTRODUCTION

1.1 History

Home automation has been a feature of science fiction writing for many years, but has only become practical since the early 20th Century following the widespread introduction of electricity into the home, and the rapid advancement of information technology. Early remote control devices began to emerge in the late 1800s. For example, Nikola Tesla patented an idea for the remote control of vessels and vehicles in 1898. The emergence of electrical home appliances began between 1915 and 1920; the decline in domestic servants meant that households needed cheap, mechanical replacements. Domestic electricity supply, however, was still in its infancy—meaning this luxury was afforded only the more affluent households.

Ideas similar to modern home automation systems originated during the World’s Fairs of the 1930s. Fairs in Chicago (1934) and New York (1939 and 1964–65) depicted electrified and automated homes. In 1966 Jim Sutherland, an engineer working for Westinghouse Electric, developed a home automation system called “ECHO IV”; this was a private project and never commercialized. The first “wired homes” were built by American hobbyists during the 1960s, but were limited by the technology of the times. The term “smart house” was first coined by the American Association of Housebuilders in 1984. With the invention of the microcontroller, the cost of electronic control fell rapidly. Remote and intelligent control technolo-

gies were adopted by the building services industry and appliance manufacturers.

By the end of the 1990s, “domotics” was used to describe any system in which informatics and telematics were combined to support activities in the home. The phrase is a neologism formed from domus (Latin, meaning house) and informatics.

Despite interest in home automation, by the end of the 1990s there was not a widespread uptake, with such systems still considered the domain of hobbyists or the rich. The lack of a single, simplified, protocol and high cost of entry has put off consumers.

Constructed in 1998, the INTEGER Millennium House is a demonstration house built partially to showcase a variety of intelligent home automation technologies, including a building management system that could optimize the performance of the heating system, an automatic garden irrigation system that could sense soil humidity conditions and water accordingly, an intelligent security system, lighting that could be set to one of four predefined moods, and microchip-embedded programmable door keys. The house also featured advanced communication technologies such as a telephone service distributed via a local building exchange, digital satellite and terrestrial television available in every room, WebTV, and a closed-circuit television (CCTV) system.[2]

1.2 Applications

Even if we don’t consider our self well informed about the exploding innovations of the Internet of Things and machine-to-machine communication, and understand are not the most tech-savvy consumer, it’s impossible that we’ve missed the abundance of home automation products filling the shelves and ads of every home improvement store. Suddenly an ordinary errand for light bulbs will leave you wondering if your lamp could send you a message alerting you that the light bulb needs to be replaced?

Any day-to-day, repeatable process is automatable with smart home applications. The greater the control and flexibility of these processes, the more energy and cost savings the resident experiences, which are factors anyone who pays utilities strives to moderate. The smart home revolution is likely to be more of an evolution, with the incorporation of one or two home systems at a time, gradually automating our households through smart mobile devices. However, with these elements of efficiency comes the question of ease of use. Will it bring you enjoyment or exasperation?

The most common applications of home automation are lighting control, HVAC, outdoor lawn irrigation, kitchen appliances, and security systems.

1.2.1 Areas of Home Automation

1.2.1.1 Lighting Control: Leaving the Dark Ages and Stepping Into the Light

Smart lighting allows you to control wall switches, blinds, and lamps, but how intuitive is a lighting control system? It turns out, quite; its capabilities are extensive. You're able to schedule the times lights should turn on and off, decide which specific rooms should be illuminated at certain times, select the level of light which should be emitted, and choose how particular lights react through motion sensitivity.

1.2.1.2 HVAC Regulation: No Longer Burned by Your Heating Bill

As fuel costs rise and the availability and sustainability of our resources becomes a greater concern, heating/cooling our homes efficiently is less a budgetary bonus and more of a necessity. Over the past year, smart thermostats and automated home heating systems have become more readily available and easily incorporate into any home. Heating and cooling our homes consumes an average of 50% of energy costs yearly, making daily HVAC regulation progressively rewarding. Maintaining

a substantial lead among the nearly non-existent competition, the Nest Learning Thermostat, learns your heating and cooling preferences over time, eliminating the need for programming and is accessible from your smart phone app. With automated HVAC you are able to reduce the heat when a room is unoccupied, and increase or decrease it at specific times based on your schedule and occupancy.

1.2.1.3 Lawn Irrigation System: The Grass is Always Greener

A lush and healthy lawn is a source of pride for most homeowners, but the weather doesn't always cooperate and provide the adequate elements for a flourishing landscape. For decades we've relied on sprinkler systems to keep our yards at peak presentation, but at what cost? The average American home spends approximately 30% of their daily water usage on lawn and garden maintenance. Nearly half of that amount is wasted due to inefficiency. If you apply that statistic to the national average, up to 4.5 billion gallons of water is wasted per day through ineffective watering methods. If we reflect upon the monetary impact of this, it results in Americans spending over a thousand dollars a year in water, with a portion of that being waste. The global effects are even greater when you consider the growing concern over climate change and the dramatic decrease in agricultural natural resources. However, sprinkler control systems, like Skydrop, are providing water regulation through real-time communication with local weather data. If a rainstorm develops and deposits two inches of rainwater on your lawn, the automated sprinkler detects the saturation and disables its scheduled watering. Conversely, the system will be alerted to dry conditions and supply the necessary amount of nourishment, without over-watering.

1.2.1.4 Security Systems: Knock, Knock...

Who's there? The Internet of Things. While efficiency and conservation are certainly IoT benefits, its potential to have improved control over home security is a primary focus. Smart locks, like Kwikset's Kevo, a Bluetooth enabled electronic deadbolt, and various connected home security systems, such as iSmartAlarm, offer a variety of features including door and window sensors, motion detectors, video

cameras and recording mechanisms. All of which are connected to a mobile device and accessible via the cloud, thus enabling you to access real-time information on the security status of your home. Naturally, there is a great deal of scrutiny regarding the level of trust in controlling your home's security system via a mobile device, but it begs earnest exploration when weighing the potential benefits and peace of mind it provides homeowners.

1.2.2 Technologies

1.2.2.1 Bluetooth Smart

Bluetooth Smart, or Bluetooth Low Energy (BLE), is a energy efficient version of Bluetooth wireless technology often seen in smartphones, and ideal for use with a headset. Though its range is limited, its energy usage is quite low. Its energy efficiency, combined with its compatibility with existing smartphones and other devices, makes it easy for developers and OEMs to create solutions that can immediately be added to existing systems.

Although it uses different channels, BLE operates in the already crowded 2.400 GHz-2.4835 GHz ISM band and the required data compression diminishes audio quality. If BLE is primarily used within the home or in close proximity, its low energy usage and extensive battery life make it a viable option.

1.2.2.2 Wi-Fi

Undoubtedly, Wi-Fi is a fixture in nearly every home, so it goes without saying that a wide range of automated devices are already compatible with this standard and thus its advantages are numerous. Nearly everyone has access to Wi-Fi and many, if not all, consumers considering home automation systems have it installed. Wi-Fi was designed to handle large amounts of data traffic, so bandwidth to control your home devices is not an issue. From a security perspective, Wi-Fi Protected Access (WPA and WPA2) encryption has been shown to provide reasonable security for home users when activated and implemented.

However, Wi-Fi's limited bandwidth is an issue. Its connection speeds can be lowered or even disrupted by having other Wi-Fi devices in the same general area. Thus, you can expect slower response times. The good news is that most home automation endpoints are not bandwidth hogs, so response time may not matter. Battery consumption is also problematic, as Wi-Fi consumes much more power than other technologies, and when combined with its limited bandwidth, it may not be an optimal solution.

Chapter 2

TECHNICAL DETAILS

2.1 Arduino

Arduino is an open-source computer hardware and software company, project and user community that designs and manufactures microcontroller-based kits for building digital devices and interactive objects that can sense and control objects in the physical world. It can be used to develop creative objects and that's why it mainly intended for artists, designers, hobbyists and anyone who is creative.

It takes input from variety of switches and sensors, and controlling of variety of motors, lights and other physical outputs. Projects based on Arduino can be stand-alone; they can communicate with software running on the computer. The Arduino board can be manually assembled or purchased preassembled; the open source IDE can be downloaded for free. The Arduino programming language is nothing but the implementation of wiring. It is similar to the physical computing platform, which is based on the programming needed in multimedia environment.

Uno is an Italian count equivalent to one and it named to mark the release of Arduino 1.0.

The Arduino Uno is a microcontroller board which uses ATmega328. It consists of 14 digital i/o pins. In those 14 pins, 6 pins are used for PWM outputs, another 6

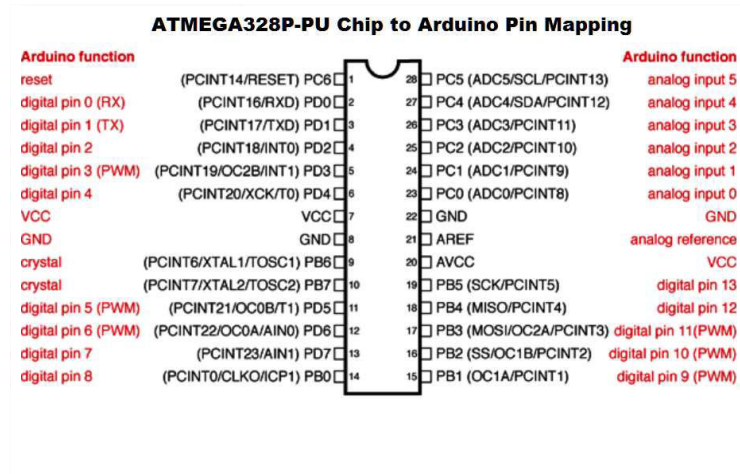


Figure 2.1: Atmega 328 to Arduino Pin Mapping

pins are analog inputs. It has 16 MHz ceramic resonator, a USB connection that is use to connect the Arduino to system, a power jack for power supply. It also has an ICSP header, and a reset button which is used to reset the codes stored in the Arduino microcontroller. The board has everything which is needed to support a microcontroller to function. Arduino can be powered up either by connecting it to the computer through USB cable or by using AC-to-DC adapter or battery which is connected to power jack.[1]

2.1.1 Features-:

- High Performance, Low Power AVR 8-Bit Microcontroller.
- Advanced RISC Architecture
 - 131 Powerful Instructions – Most Single Clock Cycle Execution
 - 32 x 8 General Purpose Working Registers
 - Fully Static Operation
 - Up to 20 MIPS Throughput at 20 MHz
 - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments

- 4/8/16/32K Bytes of In-System Self-Programmable Flash program memory
- 256/512/1K Bytes EEPROM
- 512/1K/2K Bytes Internal SRAM
- Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
- Data retention: 20 years at 85°C/100 years at 25°C
- Optional Boot Code Section with Independent Lock Bits
- In-System Programming by On-chip Boot Program
- True Read-While-Write Operation
- Programming Lock for Software Security

●Peripheral Features

- Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
- One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode

- Real Time Counter with Separate Oscillator
- Six PWM Channels
- 8-channel 10-bit ADC in TQFP and QFN/MLF package
 - Temperature Measurement
- 6-channel 10-bit ADC in PDIP Package
 - Temperature Measurement
- Programmable Serial USART
- Master/Slave SPI Serial Interface
- Byte-oriented 2-wire Serial Interface (Philips I²C compatible)
- Programmable Watchdog Timer with Separate On-chip Oscillator
- On-chip Analog Comparator
- Interrupt and Wake-up on Pin Change

●Special Microcontroller Features

- Power-on Reset and Programmable Brown-out Detection
- Internal Calibrated Oscillator
- External and Internal Interrupt Sources

- Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby

- I/O and Packages

- 23 Programmable I/O Lines
- 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF

- Operating Voltage:

- 1.8 - 5.5V

- Temperature Range:

- -40 °C to 85 °C

- Speed Grade:

- 0 - 4 MHz @1.8 - 5.5V; 0 - 10 MHz @2.7 - 5.5.V; 0 - 20 MHz @4.5 - 5.5V

- Power Consumption at 1 MHz, 1.8V, 25°C

- Active Mode: 0.2 mA
- Power-down Mode: 0.1 μ A
- Power-save Mode: 0.75 μ A (Including 32 kHz RTC)

2.1.2 Pin descriptions

2.1.2.1 VCC

Digital voltage supply of 5V.

2.1.2.2 GND

Ground corresponds to the reference point and acts as the heat sink.

2.1.2.3 Port B (PB7:0) XTAL1/XTAL2/TOSC1/TOSC2

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics

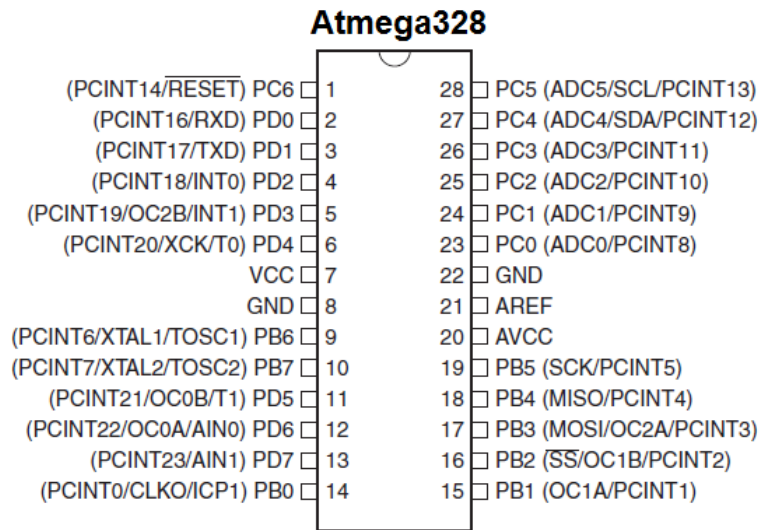


Figure 2.2: Atmega 328 Pinout

with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit. Depending on the clock selection fuse settings, PB7 can be used as output from the inverting Oscillator amplifier.

If the Internal Calibrated RC Oscillator is used as chip clock source, PB7...6 is used as TOSC2...1 input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

2.1.2.4 Port C (PC5:0)

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC5...0 output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C

pins are tri-stated when a reset condition becomes active, even if the clock is not running.

2.1.2.5 PC6/RESET

If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C.

If the RSTDISBL Fuse is unprogrammed, PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a Reset.

2.1.2.6 Port D (PD7:0)

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

2.1.3 Power

We can power up Arduino Uno board by two ways firstly we can use USB cable; secondly, power supply can be external one.

External power supply that is non-USB can either be an AC-to-DC adapter or a battery. The adapter has a centre-positive plug of 2.1mm that can be connected into the on board power jack. The leads of the battery should be inserted in the pins of power connector one of which is Gnd and another one is the Vin pin.

The board needs 6 to 20 volts provided externally to operate. If the voltage supplied to the board is less than 7 volts the operations of the board may be unstable as 5V pin may supply voltage that will be less than 5 volt. If the supplied voltage is greater than 12 volt the voltage regulator may overheat and it may cause damage to the board. So the voltage should be in the range of 7 to 12 volts.

The power pins of the Arduino Uno board are as follows:

- Vin- It is used to provide external voltage to the board (via power jack).
- 5V- This pin gives a regulated 5V as its output from the regulator present on the Arduino board. If we are providing power to the board through the DC power jack the input voltage to the board will be in range of 7V to 12V, if supply is through the USB connector the input voltage will be 5V, else if we are using the VIN pin of the board then the range will be 7V to 12V. Voltage supplied via the 5V or 3.3V pins bypasses the on board regulator, and can cause damage to board.
- 3V3- This pin outputs regulated 3.3V from the on board regulator.
- GND- It is the ground pin
- IORF- It provides the reference voltage with which the ATmega328 operates.

2.1.4 Memory

The ATmega328 has memory of 32 KB. 0.5 KB of this memory is used for the bootloader. It has SRAM (static random access memory) of 2 KB and also have 1 KB of EEPROM which is used to read and write with the EEPROM library.

2.1.5 Input and Output

It has the 14 digital pins that can be used as an input or output pins. Each needs 5V to operate. Each pin has an internal pull-up resistor of 20-50 kOhms which is disconnected by default and able to provide or receive 40 mA.

Some pins have specialised functions:

- Serial (0 and 1 pin) –It is used to receive (pin 0) and transmit (pin 1) TTL serial data. These pins are connected to USB-to-TTL Serial chip pins of the ATmega8U2 .

- External Interrupts (2 and 3 pin)- These pins can be programmed to generate an interrupt on a that either be a low value, a rising or falling edge, or there is a change in value

- PWM: (3, 5, 6, 9, 10, and 11 pin) - these pins used to give 8-bit PWM output.

- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK) pin - These pins uses SPI library for SPI communication.

- LED (13 pin) - At digital pin 13 a built-in LED is connected. The LED glows when the pin is high, else it's off.

- There are 6 analog input pins; labelled as A0, A1, A2, A3, A4 and A5, each of which has resolution of 10 bits that is total 1024 different values can be provided. They measure 5V by default from ground, but the upper end of their range can be changed by using the pin labelled as AREF.

- TWI: (A4 and A5 pin) - A4 is also known as SDA pin and A5 is also known as SCL pin. This two pin uses wire library to supports TWI communication.

There are some other pins on the board:

- AREF- It is used to provide a reference voltage for the analog inputs. It is Used with `analogReference()`.

- Reset- It is used to reset the microcontroller. Basically used to add a reset button to shields which is used to block the one present on the board.

2.1.6 Communication

The Arduino Uno board has the ability to communicate with the computer, other Arduino, or the other microcontrollers. The ATmega328 has the ability to communicate serially with the help of digital pin 0 which is used to receive and digital pin 1 is used to transmit. The '16U2 microcontroller needs no external driver; it only uses the standard USB COM drivers. The software used to program Arduino module needs a serial monitor, through which simple textual data are sent to and

from the Arduino board. When data is transmitted through USB-to serial chip and USB connected to the computer the on board LEDs of RX and TX will glow. Any of the digital pins of the Uno board can be enabled for serial communication by using Software Serial library.

2.1.7 USB Overcurrent Protection

The Arduino Uno has a polyfuse which is resettable by nature and helps to protect the USB ports of the computer that is used from being short or overcurrent. Generally most of computers have their own subsystems that provide internal protection; the fuse purposely needed for an extra layer of protection. The fuse will break automatically whenever the current applied to the USB port is more than 500mA until the short or overload is rectified.

2.1.8 Arduino v/s Microcontroller

There are many other microcontroller platforms that can be used for physical computing. Some of them are Phidgets, Netmedia's BX-24, MIT's Handy board, Parallax Basic Stamp and there are many more which offer similar functionalities. All of these need the knowledge of microcontroller programming to write the code so that the microcontroller is able to perform specific function but these all are easy-to-use package. Though Arduino also simplifies the process of working and to code the microcontroller but has some additional advantages over other platforms especially it is easy to understand for teachers, students and even the interested amateurs can easily use the board.

- Inexpensive- Arduino Uno boards does not cost much compared to other microcontroller platforms. It is possible to assemble it manually.
- Cross platform- Arduino software has the compatibility with different operating systems such as Windows, Macintosh OSX, and Linux whereas most of microcontroller systems are only compatible to the windows.
- Simple clear programming environment-programming is quite simple and easy to use for beginners; even it is flexible enough that work for advanced user.

- Open source and extensible software-the software used to program Arduino is an open source tool. The language is simple, easy to understand and can be easily expanded through C++ libraries; it is based on C programming which is one of the great advantage of Arduino over AVR. We can add AVR-C code directly into your Arduino programs.

- Open source and extensible hardware - The circuit designers can make their own version of the module by extending or improving its functionalities, as the published plans for the modules are under a Creative Commons license.

2.2 Temperature Sensor LM35

The LM35 series are precision integrated-circuit temperature devices with an output voltage linearly-proportional to the Centigrade temperature. The LM35 device has an advantage over linear temperature sensors calibrated in Kelvin, as the user is not required to subtract a large constant voltage from the output to obtain convenient Centigrade scaling. The LM35 device does not require any external calibration or trimming to provide typical accuracies of $\pm\frac{1}{4}^{\circ}\text{C}$ at room temperature and $\pm\frac{3}{4}^{\circ}\text{C}$ over a full -55°C to 150°C temperature range. Lower cost is assured by trimming and calibration at the wafer level. The low-output impedance, linear output, and precise inherent calibration of the LM35 device makes interfacing to readout or control circuitry especially easy. The device is used with single power supplies, or with plus and minus supplies. As the LM35 device draws only $60\mu\text{A}$ from the supply, it has very low self-heating of less than 0.1°C in still air. The LM35 device is rated to operate over a -55°C to 150°C temperature range, while the LM35C device is rated for a -40°C to 110°C range (-10°C with improved accuracy). The LM35-series devices are available packaged in hermetic TO transistor packages, while the LM35C, LM35CA, and LM35D devices are available in the plastic TO-92 transistor package. The LM35D device is available in an 8-lead surface-mount small-outline package and a plastic TO-220 package.

2.2.1 Applications

The LM35 can be applied easily in the same way as other integrated-circuit temperature sensors. It can be glued or cemented to a surface and its temperature will be within about 0.01 °C of the surface temperature.

This presumes that the ambient air temperature is almost the same as the surface temperature; if the air temperature were much higher or lower than the surface temperature, the actual temperature of the LM35 die would be at an intermediate temperature between the surface temperature and the air temperature. This is especially true for the TO-92 plastic package, where the copper leads are the principal thermal path to carry heat into the device, so its temperature might be closer to the air temperature than to the surface temperature.

To minimize this problem, be sure that the wiring to the LM35, as it leaves the device, is held at the same temperature as the surface of interest. The easiest way to do this is to cover up these wires with a bead of epoxy which will insure that the leads and wires are all at the same temperature as the surface, and that the LM35 die's temperature will not be affected by the air temperature.

The TO-46 metal package can also be soldered to a metal surface or pipe without damage. Of course, in that case the V-terminal of the circuit will be grounded to that metal. Alternatively, the LM35 can be mounted inside a sealed-end metal tube, and can then be dipped into a bath or screwed into a threaded hole in a tank. As with any IC, the LM35 and accompanying wiring and circuits must be kept insulated and dry, to avoid leakage and corrosion. This is especially true if the circuit may operate at cold temperatures where condensation can occur. Printed-circuit coatings and varnishes such as Humiseal and epoxy paints or dips are often used to insure that moisture cannot corrode the LM35 or its connections.

These devices are sometimes soldered to a small light-weight heat fin, to decrease the thermal time constant and speed up the response in slowly-moving air. On the

other hand, a small thermal mass may be added to the sensor, to give the steadiest reading despite small deviations in the air temperature.

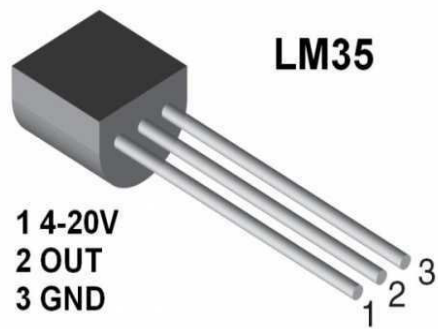


Figure 2.3: Temperature Sensor-LM35

2.2.2 Specifications:

The features include:

- Calibrated directly in Celsius
- Linear +10-mV/ °C Scale Factor
- 0.5 °C Ensured Accuracy (at 25 °C)
- Rated for Full -55 °C to 150 °C range
- Suitable for remote applications
- Low cost due to water level trimming
- Operates from 4V to 30V
- Less than 60 μ A current drain
- Low self heating, 0.08 °C in still air
- Non linearity only 0.25 °C typical
- Low Impedance output, 0.1 Ω for 1mA load

cally. When the module is at the order-response work mode, user can send the AT command to the module to set the control parameters and sent control order.

It can be preset to be master module or slave module, but by default the factory setting is SLAVE.

HC-05 is a class-2 bluetooth module with Serial Port Profile, which can configure as either Master or slave. a Drop-in replacement for wired serial connections, transparent usage. You can use it simply for a serial port replacement to establish connection between MCU, PC to your embedded project.

The module operates on 3.3 V DC power & the acceptable TX/RX signal level is 3.3v & not 5v. The KEY pin (pin 34) on a HC05 module plays an important role in entering AT mode of the module.

It has 6 pins-:

- V_{cc}
- GND
- TXD
- RXD
- Key
- State

2.3.1 The key factory parameters

2.3.1.1 Master role

It means that the module has no function to remember the last paired slave device. It can be made paired to any slave device. In other words, just set AT+CMODE=1 when out of factory. If you want HC-05 to remember the last paired slave device address like HC-06, you can set AT+CMODE=0 after paired with the other device.

2.3.1.2 Pairing

The master device can not only make pair with the specified Bluetooth address, like cell-phone, computer adapter, slave device, but also can search and make pair with the slave device automatically.

Typical method: On some specific conditions, master device and slave device can make pair with each other automatically. (This is the default method.)

2.3.1.3 Multi-device communication

There is only point to point communication for modules, but the adapter can communicate with multi-modules.

2.3.1.4 Default communication baud rate

9600 symbols per second and 4800-1.3M rates are settable.

2.3.1.5 Consumption

During the pairing, the current is fluctuant in the range of 30-40mA. The mean current is about 25mA. After paring, no matter processing communication or not, the current is 8mA. There is no sleep mode. This parameter is same for all the Bluetooth modules.

2.3.2 Specifications:

The specifications are:

- Bluetooth protocol: Bluetooth Specification v2.0 EDR
- Frequency: 2.4GHz ISM band
- Modulation: GFSK(Gaussian Frequency Shift Keying)
- Emission power: $\leq 4\text{dBm}$, Class 2
- Sensitivity: $\leq -84\text{dBm}$ at 0.1% BER
- Speed: Asynchronous: 2.1Mbps(Max) / 160 kbps, Synchronous: 1Mbps/1Mbps
- Security: Authentication and encryption
- Profiles: Bluetooth serial port

- Power supply: 3.3VDC 50mA
- Working temperature: -20 75Centigrade
- Dimension: 26.9mm x 13mm x 2.2 mm
- Typical -80dBm sensitivity
- Up to +4dBm RF transmit power
- Low Power 1.8V Operation ,1.8 to 3.6V I/O
- PIO control
- UART interface with programmable baud rate
- With integrated antenna
- Bluetooth V2.0+EDR (Enhanced Data Rate) 3Mbps Modulation
- Complete 2.4GHz radio transceiver and baseband.



Figure 2.6: Bluetooth Module Pin out

2.3.3 Interfacing HC-05 with arduino

The interfacing circuit is a shown in figure 2.7

2.4 IR Sensor

An IR sensor is basically a device which consists of a pair of an IR LED and a photodiode which are collectively called a photo-coupler or an opto-coupler. The

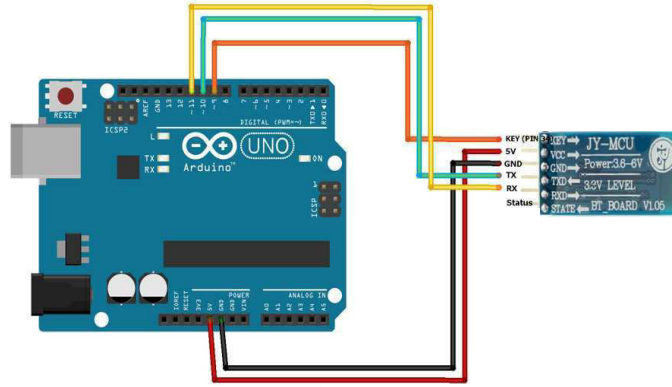


Figure 2.7: Interfacing HC-05 with arduino

IR LED emits IR radiation, reception and/or intensity of reception of which by the photodiode dictates the output of the sensor.

2.4.1 IR LED

Infrared radiation is the portion of electromagnetic spectrum having wavelengths longer than visible light wavelengths, but smaller than microwaves, i.e., the region IR sensor roughly from $0.75\mu m$ to $1000\mu m$ is the infrared region. Infrared waves are invisible to human eyes. The wavelength region of $0.75\mu m$ to $3\mu m$ is called near infrared, the region from $3\mu m$ to $6\mu m$ is called mid infrared and the region higher than $6\mu m$ is called far infrared.

2.4.2 Photodiode

A photodiode is a type of diode which detects light. We can think of it as having a very high resistance when no light is falling on it. As we increase the intensity of light incident on it, the current through it gradually increases too. So, by increasing the incident light on a photodiode, we convert it into a normal low value resistor, which conducts current. We should note here that a photodiode looks exactly like an LED, sometimes, with a dark blue or black film on the outer casing.

2.4.3 Working Principle

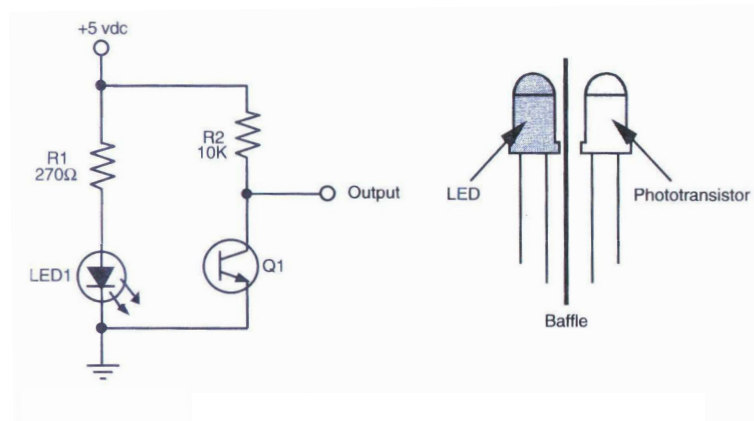


Figure 2.8: IR Sensor Circuit

If the IR LED emissions become incident on the photodiode, the photodiode's resistance comes down to a finite value. The drop across the 10K series resistor is what we use as the input, which is compared with the threshold. The point to be noted here is that more the incident radiation on the photodiode, less will be the drop across it, and hence more will be the drop across the series resistor. If the voltage developed across the resistor is greater than the threshold set by us, the output of the IC will be high, else it will be low. Hence, if our reflected radiation is never strong enough to be greater than the threshold and we have a constant low as output, we can reduce the threshold voltage by turning the “minus shaped” slit in the variable resistance towards its terminal where we connected Gnd. In case our threshold is very low and the output is always high in spite of no radiation or if it is just too sensitive, then you can increase the threshold by turning the slit the other way. When the emissions are absorbed by a black surface, the resistance of the photodiode becomes very high due to no incidence of IR emissions on it, and the output remains low.

There are different types of IR sensors working in various regions of the IR spectrum but the physics behind “IR sensors” is governed by three laws:[3]

2.4.3.1 Planck's Radiation Law

Every object at a temperature T not equal to 0 K emits radiation. Infrared radiant energy is determined by the temperature and surface condition of an object. Human eyes cannot detect differences in infrared energy because they are primarily sensitive to visible light energy from 400 to 700 nm . Our eyes are not sensitive to the infrared energy.

2.4.3.2 Stephan Boltzmann Law

The total energy emitted at all wavelengths by a black body is related to the absolute temperature as

$$W_b = \sigma T^4$$

where,

W_b : Total energy emitted

σ : constant = $5.67 * 10^{-8}\text{ m}^{-2}\text{K}^{-4}$

T : Temperature of the object

2.4.3.3 Wein's Displacement Law

Wein's Law tells that objects of different temperature emit spectra that peak at different wavelengths. It provides the wavelength for maximum spectral radiant emittance for a given temperature.

The relationship between the true temperature of the black body and its peak spectral exitance or dominant wavelength is described by this law

$$\lambda_{max} = k/T = 2898/T \text{ OR } \lambda_{max}T = 2898$$

The world is not full of black bodies; rather it comprises of selectively radiat-

ing bodies like rocks, water, etc. and the relationship between the two is given by emissivity (E).

Emissivity depends on object color, surface roughness, moisture content, degree of compaction, field of view, viewing angle and wavelength.

2.4.4 Interfacing with Arduino

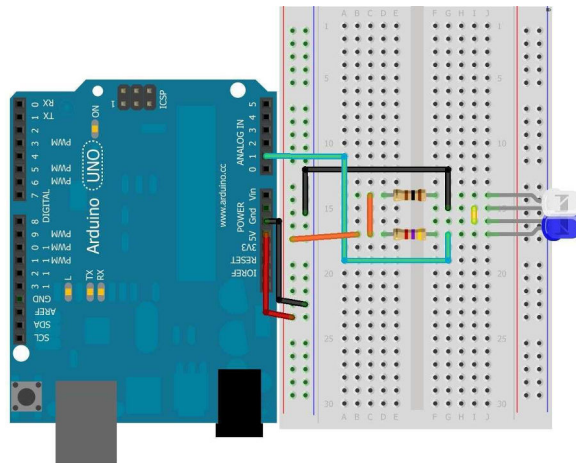


Figure 2.9: Interfacing IR sensor with Arduino

2.5 Light Sensor

A Light Dependent Resistor (LDR) or a photo resistor is used as a light sensor. It is a device whose resistivity is a function of the incident electromagnetic radiation. Hence, they are light sensitive devices.

2.5.1 Construction of a Photocell

The structure of a light dependent resistor consists of a light sensitive material which is deposited on an insulating substrate such as ceramic in zigzag pattern to obtain the desired resistance and power rating. This zigzag area separates the metal deposited areas into two regions. Then the ohmic contacts are made on the either sides of the area. The resistances of these contacts should be as less as possible to make sure that the resistance mainly changes due to the effect of light

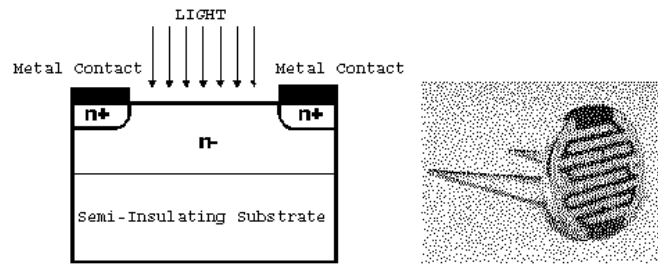


Figure 2.10: Construction of an LDR

only. Materials normally used are cadmium sulphide, cadmium selenide, indium antimonide and cadmium sulphonide. The use of lead and cadmium is avoided as they are harmful to the environment.

2.5.2 Working Principle

A light dependent resistor works on the principle of photo conductivity. Photo conductivity is an optical phenomenon in which the materials conductivity is increased when light is absorbed by the material. When light falls i.e. when the photons fall on the device, the electrons in the valence band of the semiconductor material are excited to the conduction band. These photons in the incident light should have energy greater than the band gap of the semiconductor material to make the electrons jump from the valence band to the conduction band. Hence when light having enough energy strikes on the device, more and more electrons are excited to the conduction band which results in large number of charge carriers. The result of this process is more and more current starts flowing through the device when the circuit is closed and hence it is said that the resistance of the device has been decreased. This is the most common working principle of LDR.[6]

2.5.3 Types of Light Dependent Resistors

Based on the materials used they are classified as:

2.5.3.1 Intrinsic photo resistors (Un doped semiconductor)

These are made of pure semiconductor materials such as silicon or germanium. Electrons get excited from valance band to conduction band when photons of enough energy fall on it and number charge carriers is increased.

2.5.3.2 Extrinsic photo resistors

These are semiconductor materials doped with impurities which are called as dopants. Theses dopants create new energy bands above the valence band which are filled with electrons. Hence this reduces the band gap and less energy is required in exciting them. Extrinsic photo resistors are generally used for long wavelengths.

2.5.4 Interfacing with Arduino

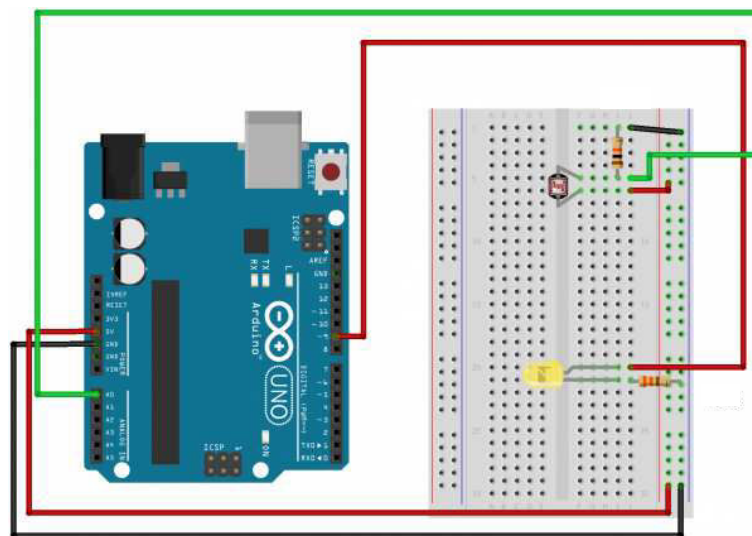


Figure 2.11: Interfacing LDR using Arduino

2.6 Relay

Thus far we have seen a selection of input devices that can be used to detect or sense a variety of physical variables and signals and are therefore called Sensors. But there are also a variety of electrical and electronic devices which are classed as

Output devices used to control or operate some external physical process. These output devices are commonly called Actuators.

Actuators convert an electrical signal into a corresponding physical quantity such as movement, force, sound etc. An actuator is also classed as a transducer because it changes one type of physical quantity into another and is usually activated or operated by a low voltage command signal. Actuators can be classed as either binary or continuous devices based upon the number of stable states their output has.

For example, a **relay** is a binary actuator as it has two stable states, either energised and latched or de-energised and unlatched, while a motor is a continuous actuator because it can rotate through a full 360o motion. The most common types of actuators or output devices are Electrical Relays, Lights, Motors and Loudspeakers.

Solenoids can be used to electrically open latches, doors, open or close valves, and in a variety of robotic and mechatronic applications, etc. However, if the solenoid plunger is used to operate one or more sets of electrical contacts, we have a device called a relay that is so useful it can be used in an infinite number of different ways.

Electrical Relays can also be divided into mechanical action relays called Electromechanical Relays and those which use semiconductor transistors, thyristors, triacs, etc, as their switching device called Solid State Relays or SSRs.

(i)**Electromechanical Relay**

The term Relay generally refers to a device that provides an electrical connection between two or more points in response to the application of a control signal. The most common and widely used type of electrical relay is the electromechanical relay or EMR. The most fundamental control of any equipment is the ability to turn it “ON” and “OFF”. The easiest way to do this is using switches to interrupt the electrical supply. Although switches can be used to control something, they have their disadvantages. The biggest one is that they have to be manually

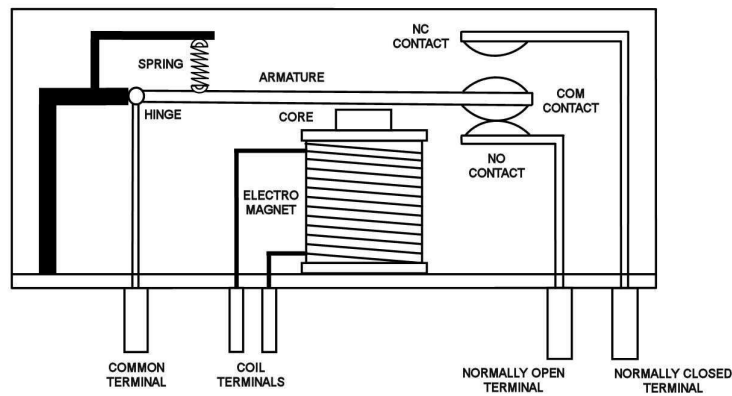


Figure 2.12: Electromechanical Relay

(physically) turned “ON” or “OFF”. Also, they are relatively large, slow and only switch small electrical currents.

Electrical Relays however, are basically electrically operated switches that come in many shapes, sizes and power ratings suitable for all types of applications. Relays can also have single or multiple contacts within a single package with the larger power relays used for mains voltage or high current switching applications being called “Contactors”.

In this tutorial about electrical relays we are just concerned with the fundamental operating principles of “light duty” electromechanical relays we can use in motor control or robotic circuits. Such relays are used in general electrical and electronic control or switching circuits either mounted directly onto PCB boards or connected free standing and in which the load currents are normally fractions of an ampere up to 20+ amperes. The relay circuit are common in Electronics applications.

As their name implies, electromechanical relays are electro-magnetic devices that convert a magnetic flux generated by the application of a low voltage electrical control signal either AC or DC across the relay terminals, into a pulling mechanical force which operates the electrical contacts within the relay. The most common form of electromechanical relay consist of an energizing coil called the “primary

circuit” wound around a permeable iron core.

This iron core has both a fixed portion called the yoke, and a moveable spring loaded part called the armature, that completes the magnetic field circuit by closing the air gap between the fixed electrical coil and the moveable armature. The armature is hinged or pivoted allowing it to freely move within the generated magnetic field closing the electrical contacts that are attached to it. Connected between the yoke and armature is normally a spring (or springs) for the return stroke to “reset” the contacts back to their initial rest position when the relay coil is in the “de-energized” condition, i.e. turned “OFF”.

(iii) Solid State Relay

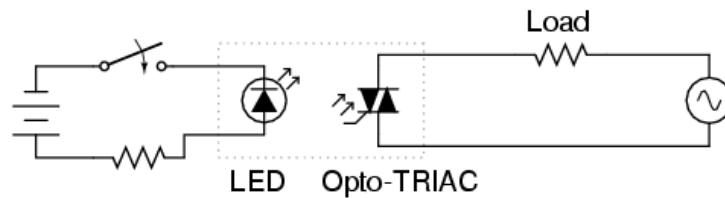


Figure 2.13: Solid State Relay

While the electromechanical relay (EMR) are inexpensive, easy to use and allow the switching of a load circuit controlled by a low power, electrically isolated input signal, one of the main disadvantages of an electromechanical relay is that it is a “mechanical device”, that is it has moving parts so their switching speed (response time) due to physically movement of the metal contacts using a magnetic field is slow.

Over a period of time these moving parts will wear out and fail, or that the contact resistance through the constant arcing and erosion may make the relay unusable and shortens its life. Also, they are electrically noisy with the contacts suffering

from contact bounce which may affect any electronic circuits to which they are connected.

To overcome these disadvantages of the electrical relay, another type of relay called a Solid State Relay or (SSR) for short was developed which is a solid state contactless, pure electronic relay.

The solid state relay being a purely electronic device has no moving parts within its design as the mechanical contacts have been replaced by power transistors, thyristors or triac's. The electrical separation between the input control signal and the output load voltage is accomplished with the aid of an opto-coupler type Light Sensor.

The Solid State Relay provides a high degree of reliability, long life and reduced electromagnetic interference (EMI), (no arcing contacts or magnetic fields), together with a much faster almost instant response time, as compared to the conventional electromechanical relay.

Also the input control power requirements of the solid state relay are generally low enough to make them compatible with most IC logic families without the need for additional buffers, drivers or amplifiers. However, being a semiconductor device they must be mounted onto suitable heatsinks to prevent the output switching semiconductor device from over heating.

A relay is an electrically operated switch. Many relays use an electromagnet to mechanically operate a switch, but other operating principles are also used, such as solid-state relays. Relays are used where it is necessary to control a circuit by a low-power signal (with complete electrical isolation between control and controlled circuits), or where several circuits must be controlled by one signal. The first relays were used in long distance 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.

2.6.1 Working Principle

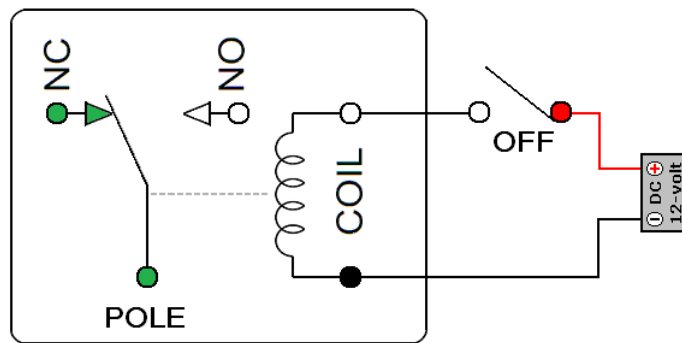


Figure 2.14: What a relay looks like

In our relay, we have two sets of electrically conductive contacts.

(i) **Normally open (NO) contacts** : They connect the circuit when the relay is activated; the circuit is disconnected when the relay is inactive.

(ii) **Normally closed (NC) contacts** : They disconnect the circuit when the relay is activated; the circuit is connected when the relay is inactive. It is also called a "Form B" contact or "break" contact.

(iii) **Common (COM)** : It is the contact where high voltage input is applied.

The relays contacts are electrically conductive pieces of metal which touch together completing a circuit and allow the circuit current to flow, just like a switch. When the contacts are open the resistance between the contacts is very high in the Mega-Ohms, producing an open circuit condition and no circuit current flows.

When the contacts are closed the contact resistance should be zero, a short circuit, but this is not always the case. All relay contacts have a certain amount of contact resistance when they are closed and this is called the On-Resistance, similar to FET's.

2.6.2 Interfacing with Arduino

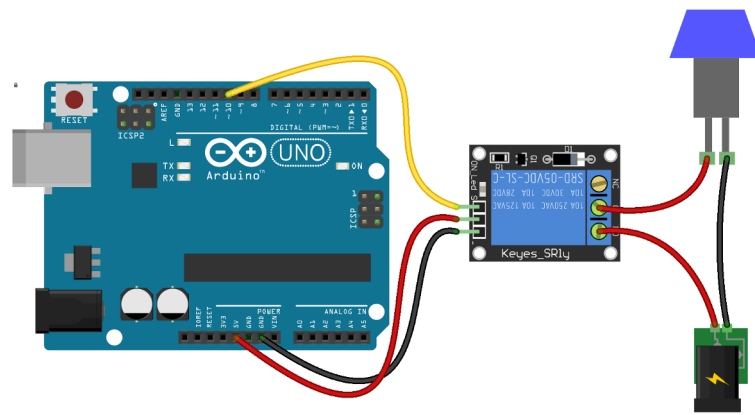


Figure 2.15: Interfacing Relay using Arduino

Chapter 3

IMPLEMENTATION

3.1 Software Requirements

3.1.1 Arduino v1.6.0

Arduino 1.0.5 is open-source environment software. Thus, this property makes it easy to write code and upload it to the i/o board. It can run on Windows, Mac OS X, and Linux. The code is written in C/C++, Java and based on Processing, avr-gcc, and other open source software. Because it is open source software thus it is very user friendly. A new file in the Arduino is called as the sketch. Here, in our project it is basically used to control real time devices by responding to humans in the language of Android. Arduino consists of various keywords and inbuilt functions. Out of all those, the one which are used in our project are discussed below.

3.1.1.1 Inbuilt Functions:

- setup()

This function is called when a sketch starts. It is used to initialize variables, pin modes, start using libraries, etc. After every power up or reset of the Arduino board, this function will run only for one time.

- loop()

After creating a setup() function, the loop() function as its name suggests, loops consecutively, allows the program to change and respond. It is used to actively control the Arduino board.

- KEYWORDS:

As we are only using the digital i/o pins, following given are the keywords:

- pinMode(pin, mode)

Configures the specified pin to behave either as an input or an output.

Parameters:

Pin: the number of the pin whose mode we wish to set

Mode: INPUT, OUTPUT, or INPUT_PULLUP.

Returns: None

- digitalWrite(pin, value)

It is used to make the value of a digital pin to be HIGH or LOW. If the pin is acting as an OUTPUT, its voltage will be set to the corresponding value: 5V (or 3.3V on 3.3V boards) for HIGH, 0V (ground) for LOW. And if the pin is acting as an INPUT, this function will enable (HIGH) or disable (LOW) the internal pull up on the input pin.

Parameters:

Pin: the pin number whose voltage we wish to set

Value: HIGH or LOW.

Returns: None

- digitalRead(pin)

This function reads the value from a specified digital pin, and tells whether it

is HIGH or LOW.

Parameters:

Pin: the number of the digital pin we want to read. It only takes integer value.

Returns: HIGH or LOW

○ analogWrite(pin, value)

It is used to make the value of an analog pin to be between 0 to 255. If the pin is acting as an OUTPUT ,its voltage will be set to the corresponding value in this range.

Parameters:

Pin: the pin number whose voltage we wish to set

Value: 0 to 255

Returns: None

○ analogRead(pin)

This function reads the value from a specified analog pin, and will give the corresponding voltage of the pin that lies somewhere in between the value 0 and 255 to the pull up on the input pin.

Parameters:

Pin: the number of the digital pin we want to read. It only takes integer value.

Returns: 0 to 255

○ delay (ms)

It is used to give an intermediate pause by delaying the program for the amount of time (in milliseconds) specified as parameter.

Parameters:

ms: the number of milliseconds to pause (unsigned long).

Returns: None

◦ map(value, fromLow, fromHigh, toLow, toHigh) [8]

Re-maps a number from one range to another. That is, a value of fromLow would get mapped to toLow, a value of fromHigh to toHigh, values in-between to values in-between, etc. Does not constrain values to within the range, because out-of-range values are sometimes intended and useful. The constrain() function may be used either before or after this function, if limits to the ranges are desired. Note that the "lower bounds" of either range may be larger or smaller than the "upper bounds" so the map() function may be used to reverse a range of numbers. The map() function uses integer math so will not generate fractions, when the math might indicate that it should do so. Fractional remainders are truncated, and are not rounded or averaged.

Parameters

value: the number to map

fromLow: the lower bound of the value's current range

fromHigh: the upper bound of the value's current range

toLow: the lower bound of the value's target range

toHigh: the upper bound of the value's target range

Returns

The mapped value.

◦ constrain(x, a, b)

Constrains a number to be within a range.[9]

Parameters:

x: the number to constrain, all data types

a: the lower end of the range, all data types

b: the upper end of the range, all data types

Returns:

x: if x is between a and b

a: if x is less than a

b: if x is greater than b

3.1.1.2 User Defined Functions:

We shall see different user-defined functions in the next section.

3.1.2 Android IDE

Using Android IDE, we can build an application on the Android platform compatible with phones, that is used for the monitoring our command signal sent to or sent by our Arduino Board wirelessly with the use of Bluetooth. Also, if you have a little knowledge of Android, you can download the already available application from Google playstore named as “ARDUDROID” for simplicity[4].

3.2 Code

```
const int rx=0;
const int tx=1;
const int ir11=2;
const int ir12=3;
const int ir21=4;
const int ir22=5;
const int ir31=6;
const int ir32=7;
const int ir41=8;
const int ir42=9;

const int fan1=10;
const int light1=11;
const int light2=12;
const int stair=13;

const int tempS=A0;
const int lightS=A1;

int tempSenVal=0,temp;
int litSenVal=0,light;

int count=0;
int ic1=0,ic2=0;
int in1=0,in2=0,in3=0,in4=0;
int irv1,irv2;

int side1, side2, irf, in;
```

```

int bfl1=0;int bfa1=0; /*light1 bluetooth flag*/
int bfl2=0;           /*light2 bluetooth flag*/
int bfaF=0;int bfF=0; /*fan's bluetooth flag*/
int bfS=0;           /*stair bluetooth flag*/

int irf1=0,irf2=0,irf3=0,irf4=0;
int tf=0;//==1 iff temp>25
int lf=0;//==1 iff

int state =0;
int flag =0;

//****ANDROID interface variables****

#define START_CMD_CHAR '*'
#define END_CMD_CHAR '#'
#define DIV_CMD_CHAR '|'
#define CMD_DIGITALWRITE 10
#define CMD_ANALOGWRITE 11
#define CMD_TEXT 12
#define CMD_READ_ARDUDROID 13
#define MAX_COMMAND 20
#define MIN_COMMAND 10
#define IN_STRING LENGHT 40
#define MAX_ANALOGWRITE 255
#define PIN_HIGH 3
#define PIN_LOW 2

String inText;

int ard_command = 0;
int pin_num = 0;
int pin_value = 0;

int pin_value_l1 = 0;
int pin_value_f = 0;

char get_char = ' '; //read serial

//****start VOID() loop****

void setup()
{

```

```

pinMode(stair,OUTPUT);
digitalWrite(stair, LOW);

pinMode(fan1,OUTPUT);
digitalWrite(fan1, LOW);

pinMode(light1,OUTPUT);
digitalWrite(light1, LOW);

pinMode(light2,OUTPUT);
digitalWrite(light2, LOW);

pinMode(ir11, INPUT);
pinMode(ir12, INPUT);

pinMode(ir21, INPUT);
pinMode(ir22, INPUT);

pinMode(ir31, INPUT);
pinMode(ir32, INPUT);

pinMode(ir41, INPUT);
pinMode(ir42, INPUT);

Serial.begin(9600);
Serial.flush();
}

void loop()
{
  irSensor(ir11,ir12,irf1,in1);
  irSensor(ir21,ir22,irf2,in2);
  TempSensor1();
  LightSensor2();
  Bluetooth();

  if(lf==1 || bf11 == 1 || irf1==1)
  {
    digitalWrite(light1,HIGH);
  }
  if(lf==1 || bf12 == 1 )
  {
    digitalWrite(light2,HIGH);
  }
  if(lf==0 && bf11==0 && irf1==0)

```

```

{
    digitalWrite(light1,LOW);
}
if(lf==0 && bf12==0 )
{
    digitalWrite(light2,LOW);
}

if(tf==1 || bfF==1)
    digitalWrite(fan1,HIGH);

else if(tf==0 && bfF==0)
    digitalWrite(fan1,LOW);

    if(bfal1==1 || bfaF==1 )
{
    if(bfal1==1)analogWrite(light1, pin_value_l1);
    if(bfaF==1)analogWrite(fan1, pin_value_f);

}
    delay(200);
}

//*****

void irSensor(int side1, int side2, int irf, int in)
{

    irv1 = digitalRead(side1);
    irv2 = digitalRead(side2);

if(irv1==1 || irv2==1)
{
    if(irv1==1){ic1++;delay(10);count++;}
    if(irv2==1){ic2++;delay(10);count++; }

    delay(500);//best 400

if(count%2==1){
    if (ic1>ic2)
    {
        in++;
        delay(40);//best 40
    }
}
}

```

```

else if(ic1<ic2)
{
    in--;
    delay(40);
}

else
    delay(40);
}
}
if(in>0)
{
    switch (side1)
    {
        case 2:
            irf1=1;
            in1=in;
            break;

        case 4:
            irf2=1;
            in2=in;
            break;

    }

}

else if(in<=0)
{

    ic1=0;ic2=0;
    switch (side1)
    {
        case 2:
            irf1=0;
            in1=0;
            break;

        case 4:
            irf2=0;
            in2=0;
            break;

    }
}
}

```

```

Serial.print("count1=");
Serial.println(ic1);

Serial.print("count2=");
Serial.println(ic2);

  Serial.print("count=");
  Serial.println(count);

Serial.print("in=");
Serial.println(in);

}

//*****

void TempSensor1()
{

  tempSenVal = analogRead(tempS);
  temp=(tempSenVal/2);

  Serial.println(temp);

  if(temp>25)
  {
    tf=1;      Serial.println("T >25");
  }
  else if(temp<=25)
  {
    tf=0;      Serial.println("T <25");
  }

}

//*****

void LightSensor2()
{
  litSenVal = analogRead(lightS);

  litSenVal = map(litSenVal, 0, 1023, 0, 255);
  litSenVal = constrain(litSenVal, 0, 255);
}

```

```

    Serial.println(litSenVal);

    if(litSenVal <60)
        {lf=1;          Serial.println("L<100");}
    else
        { lf=0;          Serial.println("L>100");}

}

//*****

void Bluetooth()
{

    Serial.flush();

    // wait for incoming data
    if (Serial.available() < 1) return;
    // if serial empty, return to loop().

    // parse incoming command start flag
    get_char = Serial.read();
    if (get_char != START_CMD_CHAR) return;

    // parse incoming command type
    ard_command = Serial.parseInt();

    // parse incoming pin# and value
    pin_num = Serial.parseInt();
    pin_value = Serial.parseInt();

    if(pin_num == 10)
        pin_value_f = pin_value;

    if(pin_num == 11)
        pin_value_l1 = pin_value;

    // 1) GET digitalWrite DATA FROM ARDUDROID
    if (ard_command == CMD_DIGITALWRITE)
    {
        if (pin_value == PIN_LOW)
        {
            digitalWrite_low(pin_num);}
        else if (pin_value == PIN_HIGH)

```

```

    {
        digital_write_high(pin_num);}
    else
        return;
        Serial.println("D" );
        Serial.println(bf11) ;
        Serial.println(bf12) ;
        Serial.println(bfF);
        Serial.println(bfS) ;
        Serial.println("A" );
        Serial.println(bfal1) ;
        Serial.println(bfaF) ;

}

// 2) GET analogWrite DATA FROM ARDUDROID
if (ard_command == CMD_ANALOGWRITE && pin_value>0)
{
    analog_write( pin_num, pin_value );
    Serial.println("D" );
    Serial.println(bf11) ;
    Serial.println(bf12) ;
    Serial.println(bfF);
    Serial.println(bfS) ;
    Serial.println("A" );
    Serial.println(bfal1) ;
    Serial.println(bfaF) ;
}

// 3) SEND DATA TO ARDUDROID
if (ard_command == CMD_READ_ARDUDROID)
{
    Serial.print(" Analog 0 = ");
    Serial.println(analogRead(A0));
    return;
}

} // end of void()

void analog_write(int pin_num, int pin_value)
{
    switch (pin_num)
    {

```



```

    case 11:

        bfa1=1;bfl1=0;
        break;

    case 10:

        bfaF=1;bF=0;
        break;

    }
}

void digital_write_high(int pin_num)
{

    switch (pin_num)
    {

        case 13:
            bfS=1;
            break;

        case 12:
            bfl2=1;
            break;

        case 11:
            bfl1=1;bfa1=0;
            break;

        case 10:
            bF=1;bfaF=0;
            break;

    }
}

void digital_write_low(int pin_num)
{
    switch (pin_num)
    {
        case 13:
            bfS=0;
            break;

```

```
case 12:
    bf12=0;
    break;

case 11:
    bf11=0; bfa11=0;
    break;

case 10:
    bfF=0; bfaF=0;
    break;
}
}
```

3.3 Hardware Requirements

- Arduino Uno Board
- Temperature Sensor-LM35
- Light Dependent Resistor
- Bluetooth Module-HC05
- IR Sensor
- Relay
- 220V 50Hz source
- 12V male adapter
- Connecting Wires
- Android Device
- Breadboard
- A model of a home for demonstration of the project

Chapter 4

RESULT

4.1 Theoretical Results

We have got the following results:

- Bluetooth module has been interfaced to the Arduino board using a self made Android GUI interface.
- A GUI application on the cell phone sends and receives our commands from the Bluetooth module.
- IR sensor counts the number of people entering a room and switch the appliances accordingly.
- Light sensor automates lighting according to a threshold value of light falling on it.
- Temperature sensor controls fan/AC switching by keeping a track of the current temperature.
- Using relays we are able to control the real life appliances ON/OFF .
- Implementation of our idea using a small model demonstrates the above points.

4.2 Pictorial Results

4.2.1 Plots

The following is the plot for IR Sensor receiver voltage dependent v/s distance for figure 2.8 for the V_{out} i.e IR Sensor Value

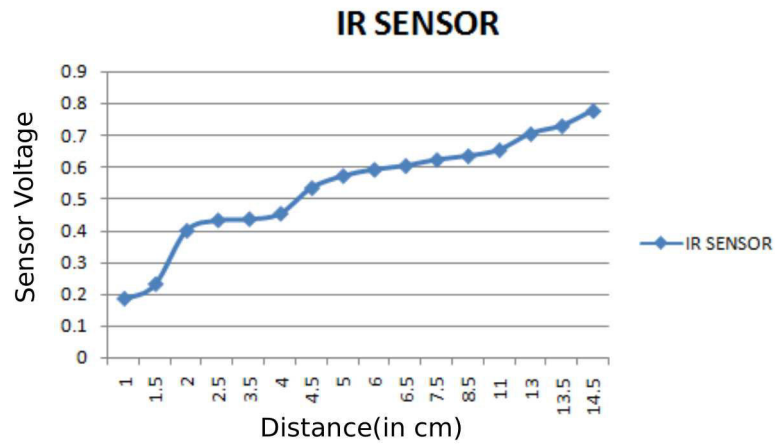


Figure 4.1: IR Sensor Voltage as a function of the distance

The following is the plot for Light intensity measured by Arduino versus the LDR voltage for figure 2.11 for the V_{out} i.e LDR Sensor Value

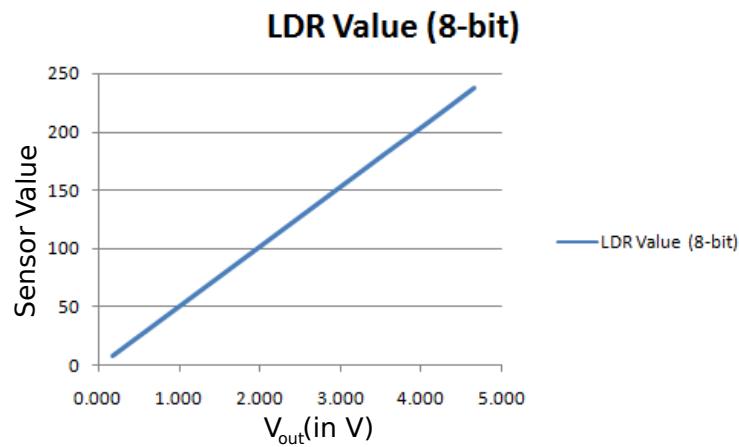


Figure 4.2: LDR voltage value as a function of output voltage

4.2.2 Snapshots of the model

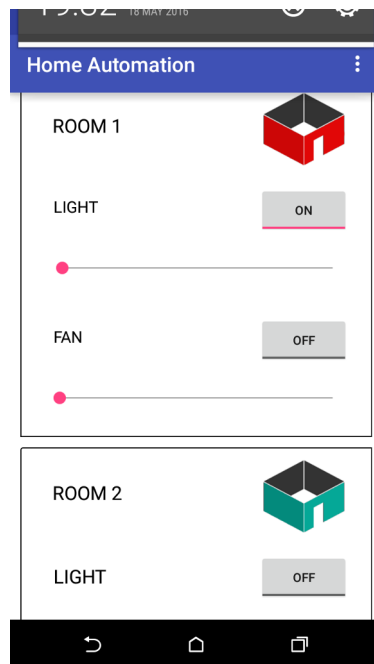


Figure 4.3: Bluetooth command for room 2 lights

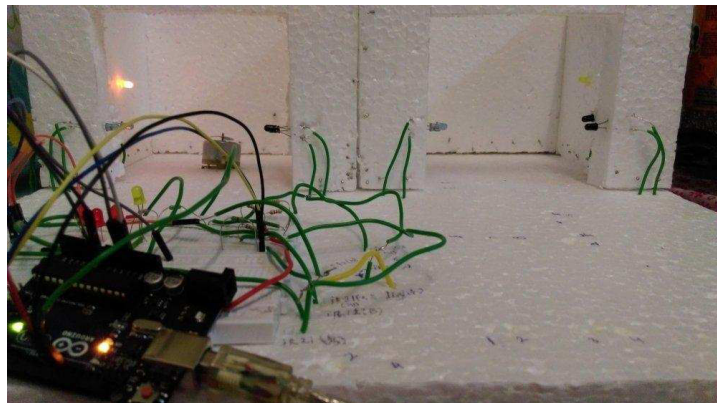


Figure 4.4: Room 1 lights controlled by Bluetooth

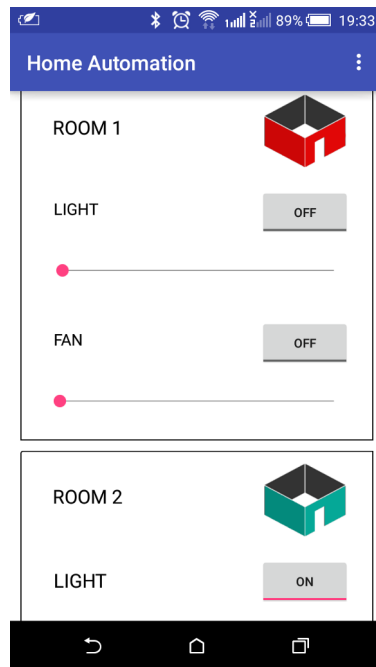


Figure 4.5: Bluetooth command for room 2 lights

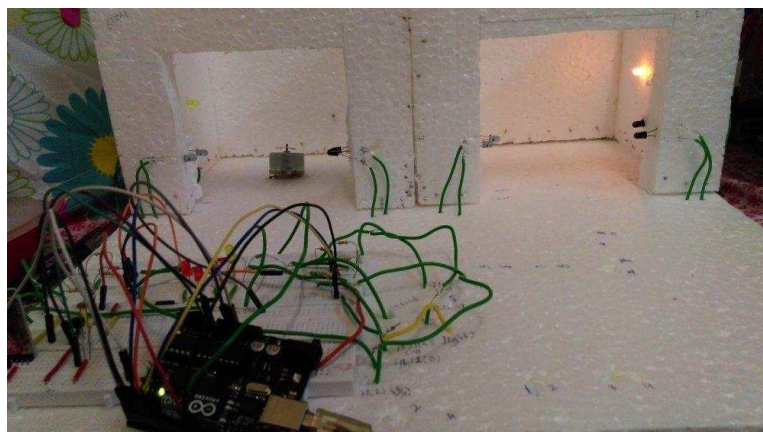


Figure 4.6: Room 2 lights controlled by Bluetooth

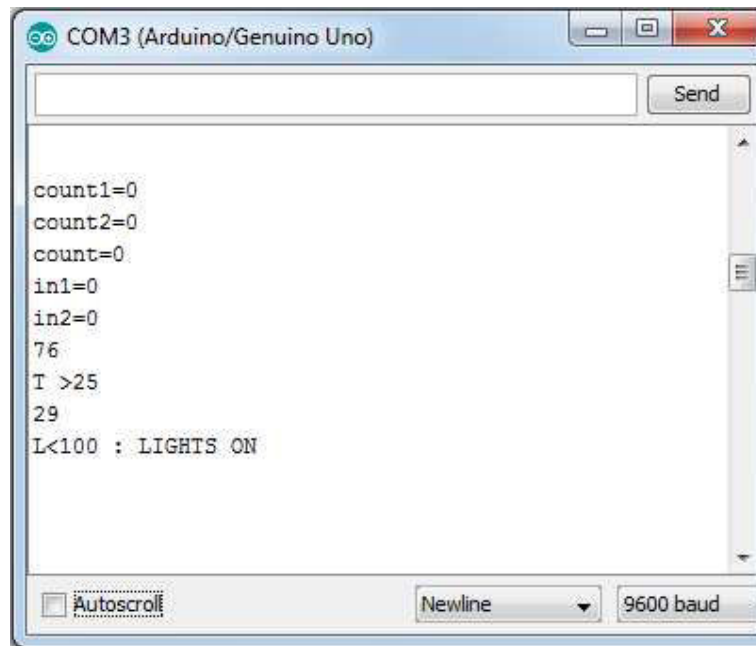


Figure 4.7: LDR reading on the Serial Monitor

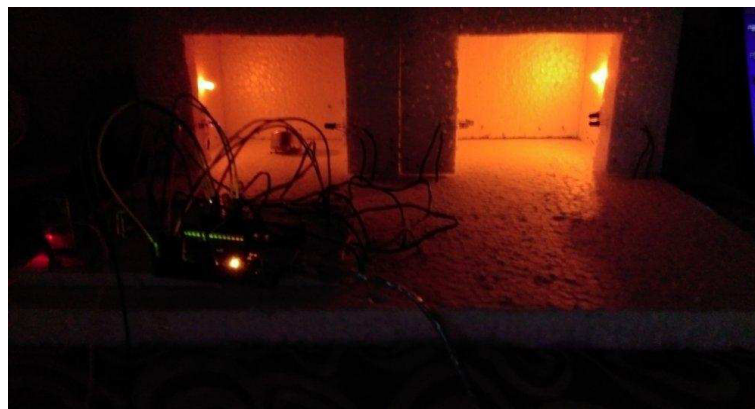


Figure 4.8: LDR controlled room lights

Chapter 5

CHALLENGES AND LIMITATIONS

We have interfaced the above mentioned modules. This project comes up with different advantages and disadvantages as mentioned after.

- ARDUINO

The advantage is that Arduino is low power consuming hardware and provides an open source advantage i.e. it has a great community of users, we get in contact with and there is also a significant number of projects that are published with full source i.e. from wiring diagrams to full codes. It also allows programming and serial communication over USB i.e. both wireless and wired communication. But the limitations are there too :

- Arduino uses its own C language with code written on Java byte code which is being loaded on the controller using boot loader. A lot of times additional cycles are required which consume more energy for the same arithmetic operations.

- The number of pins on an Arduino Board is fixed and in the physical world, its a bit difficult to handle all the appliances of a normal home in day to day life.

- The structure of Arduino is its disadvantage as well. During building a project you have to make its size as small as possible. But with the big structures of Ar-

duino we have to stick with big sized PCB's.

◦You are limited to a small number of MCUs right now: Arduino is only officially supported on the Atmel AVR and Atmel SAM series. There are ports and adaptations to a few other MCUs like the ESP8266 or even the nRF51 series, but they don't seem to be fully supported, so generally you are limited to Atmel MCUs.

◦The Arduino libraries are not very efficient in certain parts and waste RAM and CPU cycles.

- The temperature sensor that we used i.e. LM35 in actual fluctuates a lot beyond the values specified in the datasheets.

- The next big limitation was to use Bluetooth Module HC-05 in terms of its range that was only limited to 10 metres (33 ft.)

Chapter 6

FUTURE SCOPE

The project can be expanded for overcoming the above mentioned limitations:

The bluetooth module can be replaced by a GSM module that provides a range of global area network. The main problem we faced is how to receive notifications in case of an event, and the best practice would be to receive a SMS. So, the GSM module is what you need for that. For a very small network of that of a home, the bluetooth module of higher range can be utilised in the project.

All major appliances can be connected to the home automation system and can be controlled remotely. Depending on your unique requirements, new customized systems can be developed and installed. There is no limit to imagination and every feature one may need or think of can be developed now or in coming future.

Bibliography

- [1] *ATmega48A/PA/88A/PA/168A/PA/328/P:* Atmel,
http://www.atmel.com/images/atmel-8271-8-bit-avr-microcontroller-atmega48a-48pa-88a-88pa-168a-168pa-328-328p_datasheet_summary.pdf
- [2] https://en.wikipedia.org/wiki/Home_automation *Home Automation* (Accessed May 20, 2016)
- [3] <http://www.engineersgarage.com/articles/infrared-sensors> *EngineersGarage ...inspiring creations* (Accessed May 20, 2016)
- [4] <http://www.techbitar.com/ardudroid-simple-bluetooth-control-for-arduino-and-android.html> *TechBitar* (Accessed May 20, 2016)
- [5] Muhammad Ali Mazidi, Janice Gillispie Mazidi and Rolin D.Mckinlay, *The 8051 Microcontroller and Embedded Systems*, 2nd Ed. Prentice Hall, 2000, pp. 427-432, Relays
- [6] http://www.electronics-tutorials.ws/io/io_4.html , *Electronics Tutorials*
- [7] Muhammad Ali Mazidi, Janice Gillispie Mazidi and Rolin D.Mckinlay, *The 8051 Microcontroller and Embedded Systems*, 2nd Ed. Prentice Hall, 2000, pp. 300-308, LCD Interfacing
- [8] <https://www.arduino.cc/en/Reference/Map>
- [9] <https://www.arduino.cc/en/Reference/Constrain>
- [10] <http://www.engineersgarage.com/insight/how-g geared-dc-motor-works?page=6>

- [11] <https://arduino-info.wikispaces.com/BlueTooth-HC05-HC06-Modules-How-To> *BlueTooth-HC05-HC06-Modules-How-To*
- [12] <http://www.instructables.com/id/Arduino-AND-Bluetooth-HC-05-Connecting-easily/> *Arduino AND Bluetooth HC-05 Connecting easily*
- [13] <http://www.rajguruelectronics.com/bluetooth-module.html> *Bluetooth Module*
- [14] <https://alselectro.wordpress.com/2014/10/21/bluetooth-hc05-how-to-pair-two-modules/> *Bluetooth HC05- How to pair two modules*
- [15] <http://www.c-sharpcorner.com/UploadFile/167ad2/how-to-use-hc-05-bluetooth-module-with-arduino/> *How to Use HC-05 Bluetooth Module With Arduino*
- [16] <http://www.ti.com/product/LM35> *LM35*
- [17] <http://www.ti.com/lit/ds/symlink/lm35.pdf> *LM35 Precision Centigrade Temperature Sensors: August, 1999*
- [18] <http://www.electroschematics.com/6393/lm35-datasheet/> *LM35 Datasheet*
- [19] <http://www.instructables.com/id/LM35-Temperature-Sensor/> *LM35 Temperature Sensor: by innovativetom*
- [20] <http://www.facstaff.bucknell.edu/mastascu/elessonshtml/sensors/templm35.html> *Temperature Sensor - The LM35*
- [21] https://en.wikipedia.org/wiki/DC_motor *DC motor*
- [22] <http://www.electrical4u.com/working-or-operating-principle-of-dc-motor/> *Working or Operating Principle of DC Motor*
- [23] http://www.ncert.nic.in/html/learning_basket/electricity/electricity/machine/motor.htm *DC Motor*
- [24] <http://www.electricaleasy.com/2014/01/basic-working-of-dc-motor.html> *How a DC motor works?*