SMART HOME SYSTEM USING GSM AND IR TECHNOLOGIES

BY

PUNG KAH HENG

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONS)

Faculty of Information and Communication Technology
(Perak Campus)

MAY 2016

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude and appreciation to my supervisor, Dr. Suliman Mohamed Ahmed Gaber who has given me this bright opportunity to engage in this GSM and IR technologies related development project and have assisted and guided me throughout the completion in this project. It is my first time gaining such knowledge in this field and it has been an eye opening and wonderful experience. For that, a million thanks to you.

I would also like to thank my teammate, Goh Yoong Xue, for his participation and support in completing this project, and for lending me a helping hand to solve the problems I faced in this project. Finally, I must say thanks to my special someone, Tee Kar Man, and my parents for their unconditional love, support and continuous encouragement throughout the course.

ABSTRACT

This project is a smart home system design project implemented with Infrared (IR) and Global System for Mobile (GSM) technologies as an alternative to the mainstream smart home system utilizing Internet of Things (IOT). One of the biggest downsides of the common smart home system is the high price tag. Therefore, this project aims to solve this problem by using the low cost IR technology to cut down the price. GSM is also utilized for communications between user and the system to improve the coverage and stability.

Several related articles and smart home systems are studied and reviewed to find out the pros and cons of each, thus determining the viability of the proposed system. Waterfall method is then used in designing the system. Clear cut procedures are followed from finding requirements, determining the tools and materials, implementation and verification. Installation of various devices such as GSM module and Infrared transmitter sensor on Arduino board will be covered, along with schema and coding required for the system.

In short, the output of the project will be a low cost, high coverage and stability smart home system that can control appliances that have built-in IR sensors. It is also hoped that the project will encourage the invention of more appliances with built-in IR sensors to achieve a truly low cost and practical IR smart home system.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

GSM	Global System for Mobile
IEEE	Institute for Electrical and Electronics Engineers
IoT	Internet of Things
IR	Infra-Red
LED	Light Emitting Diode
PCS	Powerline Carrier Systems
SMS	Short Message Service
USB	Universal Serial Bus
RF	Radio Frequency
LED	Light-Emitting Diode
PWM	Pulse Width Modulation
SIM	Subscriber Identity Module
IDE	Integrated Development Environment
IRP	IR Protocol

CHAPTER 1: INTRODUCTION

1.1 Motivation and Problem Statement

In a world where technologies are moving at an incredible pace, tremendous amount of new inventions arise from time to time. One of the booming field and also often dubbed as “The next big thing” is Internet of Things (IoT). Smart home systems make use of this technology to connect appliances in a home to the internet, allowing owners to control them from anywhere in the world with internet connection. However, this kind of system is still not widely used and is still uncommon due to the high cost of implementation (Cleverism, 2014). There is also the issue of unable to access the smart home system if there is no internet connection coverage or if the internet connection is unstable especially in developing countries.

These issues motivate the invention of a new smart home system that uses the cheaper Infrared (IR) technology instead of IoT, and use Global System for Mobile (GSM) instead of internet connection for those appliances not equipped with IoT facility. IR is a common technology that can be used to transmit signal to control devices remotely, commonly found in remote controls such as TV remote control. GSM can be found in almost all mobile phones, allowing the phones to make and receives calls and messages. The goal is to invent a smart home system that is more affordable, and not restricted by the coverage of internet connection.

1.2 Background Information

Smart home system is a technology that allows home appliances and devices to be controlled and monitored remotely or automatically. Remote control can be a smartphone with Bluetooth or internet, or a computer on the other side of the world connected to internet. Most smart home system use internet to communicate between the user and the appliances. This means that the appliances need to have internet capability to work with the smart home system. The appliances are typically connected through Wi-Fi to a hub which is then accessed with a software application. The common appliances used in smart home systems include thermostats, security systems, blinds and lighting (Consumerreports.org, 2014).

Smart home systems have been around for a long time, but still unable to become a common product. This is mainly due to the high cost, since every appliance need to be specifically bought and comes with internet capability. In this project, a different approach is introduced to cut down the cost significantly: using IR technology instead of internet for controlling the appliances. IR is invisible radiant energy, with longer wavelengths than those of visible light. IR has a lot of applications in our life, and one of it is communications (SearchMobileComputing, 2005). IR data transmission is used for controlling devices such as TV, radio, fan and more. The data transmission is done by switching on and off repeatedly in a certain manner, to encode the data, which will be received by IR sensor. Since IR technology in data transmission is so cheap nowadays, it could be a good cheap alternative to internet in smart home system.

In this project, Arduino Uno is used as a microcontroller, coupled with modules such as GSM and IR transmitter to act as a simple smart home system. The tools and materials used in this project are described in chapter 3, along with the design and schema.

1.3 Project Objectives

The project aims to invent an affordable smart home system that can be alternative to IoT smart home systems, with the following objectives in mind:

- To build a smart home system with cost lower than usual price of IoT smart home system

The overall price of the smart home system should be ideally much lower than the usual price of IoT smart home system. The estimated cost to build the system should be under RM150, excluding the costs for appliances that can work with the system.

- To build a smart home system with built-in GSM module that is not restricted by the coverage and stability of internet

A GSM module should be connected to the smart home system, allowing the system to receive commands in the form of messages through SMS. The smart home system will not rely on the use of internet, and will have good service coverage and stability through the use of GSM module.

- To build a smart home system that can send IR signal and receive SMS

User should be able to send message containing command to the smart home system through SMS. The smart home system should also be able to receive the message, retrieve the containing command and transmit corresponding IR signal through connected IR transmitter to control appliances with built-in IR sensors.

- To build a smart home system that can read and imitate IR signals

In the case when the IR code library implemented in the system is not sufficient, the smart home system should have an IR sensor that can receive and imitate an IR signal sent by an IR remote control. This means that user can let the smart home system learn the IR code by pressing buttons on the IR remote control that comes with an appliance. The smart home system can then imitate and emit the same IR signal when required.

However, this project will not cover smart home automation of appliances that do not have built-in IR receivers.

1.4 Proposed Approach

This project will focus on the use of IR technology in place of IoT technology to develop a smart home system that is more affordable, and thus becoming more common. The smart home system will also use GSM instead of internet connection to communicate with user, allowing user to control appliances at home by using android application designed especially for this purpose as a complementary part for this project. The android application will send and read Short Message Service (SMS) to communicate with the smart home system. This project will cover the building of the central unit for the smart home system. However, the mobile application to communicate with the smart home system is also not covered in this part. IoT technology will also not be covered in this project since the implementation cost of smart home system using IoT technology is too high, which contradicts with the main purpose of this project – build an affordable smart home system. Figure 1.1 shows the flowchart of the overall system.

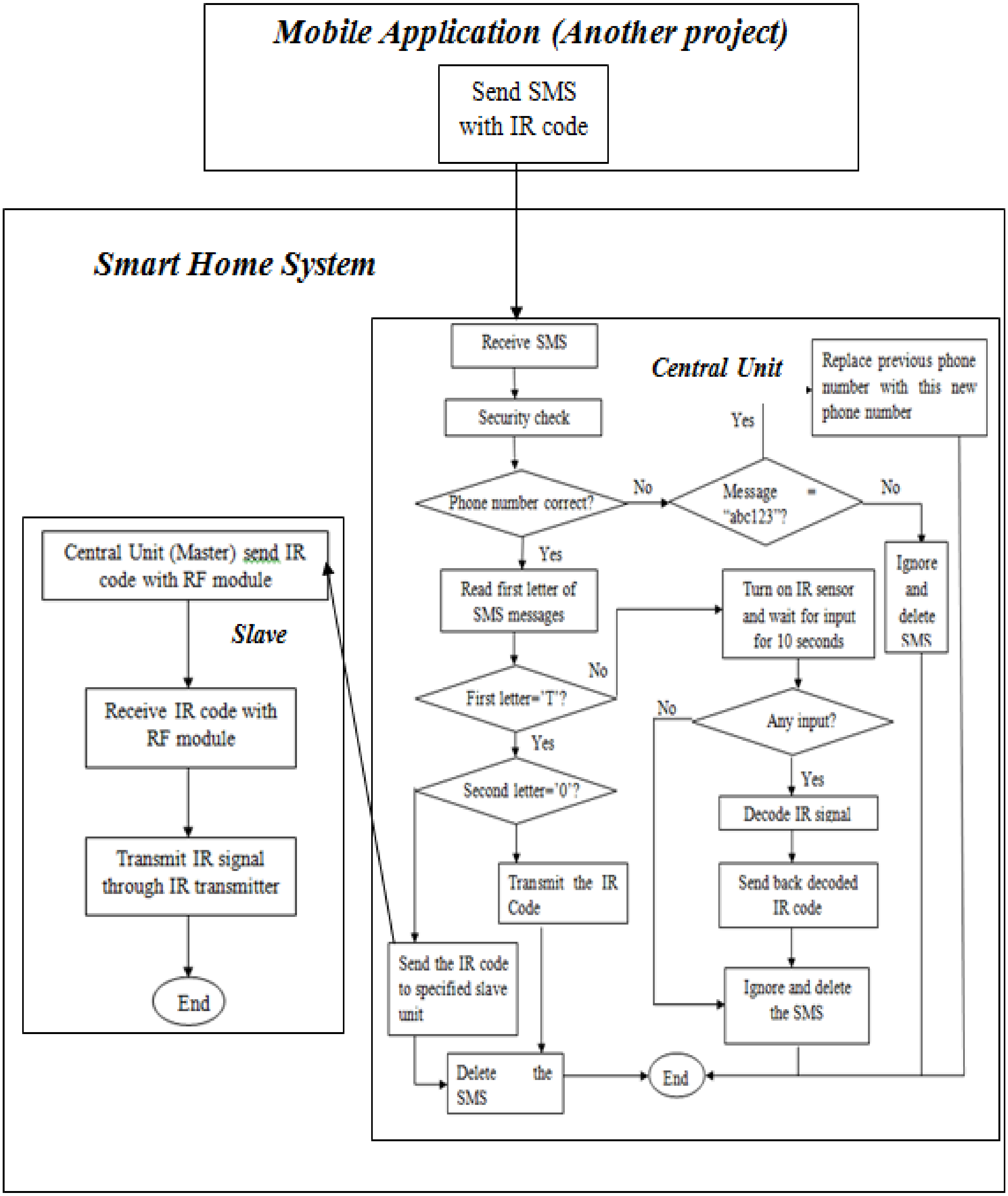


Figure 1.1 Flowchart of Overall System

1.5 Achievements Highlight and Report Organization

A smart home system consists of a master unit and a slave unit has been successfully created in this project. The system made use of IR technology to control appliances and devices with existing IR sensors. It also allow user to easily integrate the system with these devices by using the decode function.

The next chapter of this report will be the literature review of similar works that have been done before, along with comparisons with the system built from this project. Chapter 3 will be the system design, which describes how to assemble and build the system. Chapter 4 will describe the methodology, tools and more used in this project. Chapter 5 describe in depth about IR decoding and transmitting of this project. Chapter 6 gives further details on how a RF module is used for communication between the master and slave units.

1.6 Impact, significance and contribution

One of the biggest obstacles of commercializing smart home systems that use IoT is the high cost of implementation and maintenance. Aside from the purchase of smart home systems, consumers also have to buy smart appliances, which are usually costly than the regular appliances. Thus, very few people can afford to invest in smart home system although it has been in the market for quite a while now. This project can solve the problem by introducing a smart home system that use IR technology that is a quite cheaper, and there is no need to purchase smart appliances. This smart home system can become very common as almost everyone can afford it. In other words, it will be easier to commercialize as a new trend in the smart home systems. This project also solves the problem of unstable internet connection and poor internet coverage in developing countries by using GSM instead of internet to communicate with the smart home system.

This project can be further improved and used for other purposes in the future. By convincing manufactures to implement IR module which is a way cheaper than IoT technology in their appliances, this IR smart home system can control even more appliances: doors, locks, refrigerators, water boilers, cookers and more. Manufacturers can also implement IR module which can transmit signal back to the central unit of the smart home system, allowing two-way communication for purposes such as checking and monitoring status. The methods and ideas used in this project can also be used in factories, where instead of using wire to connect all the machines, IR can be used instead. In short, this project has tremendous amount of possible uses and modifications in the future, limited only by imagination.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

Smart home systems are systems that enable home automation, which may include centralized control of lighting, heating, ventilation, air conditioning, appliances, security locks and more. Smart home systems can provide a better life quality and convenience, allowing users to control and monitor home appliances remotely.

The concept of home automation has been around for quite a while as it was dated back at 1966, when possibly the first experimental home automation system was built (Spicer, 2000). Due to the problems of complexity, multiple incompatible standards and mainly high cost of implementation, smart home systems were not very common. However, the popularity of smart home systems has been increasing greatly in recent years, thanks to the recently popularized concept of IoT. The invention of smartphone and tablet connectivity also contributed to the resurrection of home automation. Various brands of smart home systems are now in the market, including Iris, Loxone and more.

However, most smart home systems in the market utilize the concept of IoT, which means every device and appliance connected to the smart home systems need to have internet capabilities. These smart appliances and devices are high in cost, thus their marketability is still limited at least in Malaysia. One of the examples is Whirlpool Smart Appliances.

2.2 Whirlpool

Whirlpool (Whirlpool, 2016) mainly focuses on the smart appliances in the kitchen and laundry. Their products include smart washer, smart dryer, smart fridge and more. Customers can download a mobile app made for Whirlpool product and control their Whirlpool smart appliances.

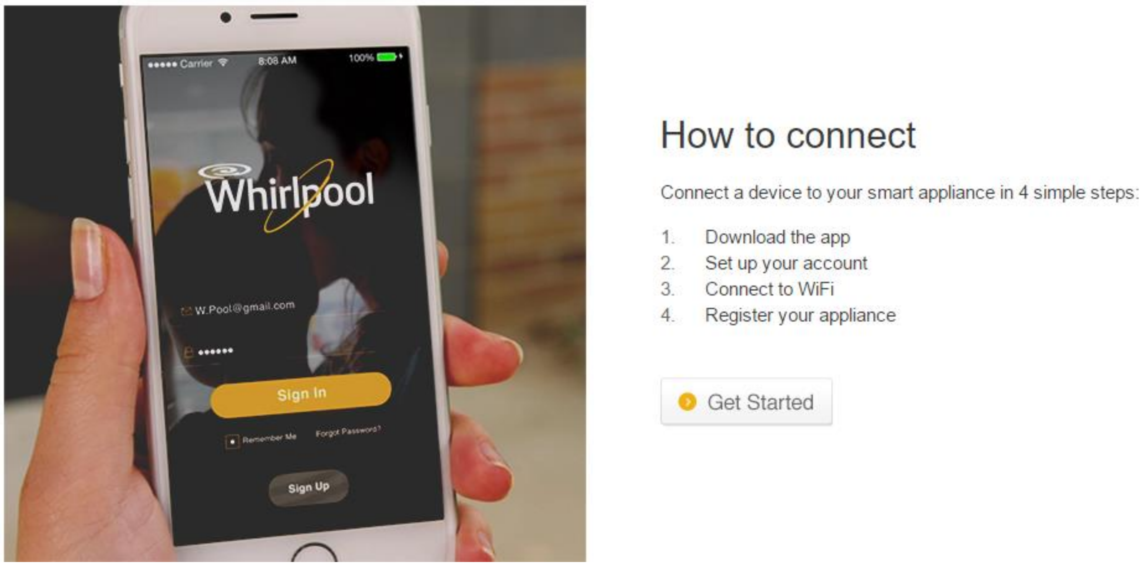


Figure 2.1 Whirpool Mobile App

The Whirlpool smart appliances are connected to the consumer’s home wireless network, which then allow the Whirlpool mobile app to communicate with these appliances online. The way these Whirlpool smart appliances work may not be as sophisticated as most would imagine, as one could say Whirlpool simply stick a computer to their appliances. While this approach allows the appliances to have tons of features, unfortunately it also results in expensive smart appliances both in terms of price and maintenance cost.

For example, by adding a computer to a normal oven that cost around RM 200 in the market, the cost skyrocketed to around RM 3000 as the price for the computer itself is higher than the oven. When you’re paying five times the price for an oven, you would also expect it to last longer than the average life expectancy of 14 years. However, the truth is that the added complexity of an oven after added with a computer actually means a shorter life expectancy (Fitzsimmons, 2016). A broken touch screen, malfunctioned network card, or a faulty chipset are all added possible troubles that might cut short the appliance's life expectancy, or at the very least results in high repairing bills.

Aside from the cost, another downside of Whirlpool smart appliances is how the users communicate with the appliances. As mentioned earlier, the smart appliances are connected to the user's wireless network to access the internet. This means that if the user did not install any wireless internet network in their home, or if the wireless network is experiencing downtime, user cannot remotely control or monitor these appliances.

Another smart home system, Iris Smart Home Management System, takes a slightly different approach.

2.3 Iris Smart Home Management System

Iris (Iris, 2016), like Whirlpool, allows users to control and monitor their smart appliances through a mobile app. However, Iris is different than Whirlpool in the sense that it uses a central unit, Iris Smart Hub to serve as the heart of Iris network. Iris Smart Hub connects all the smart appliances or products such as thermostats, lights, sensors, doors and more together wirelessly. The hub itself is connected to the user's home internet router by cable to access the internet.

Figure 2.2 Iris Home Automation Smart kit

By using the central hub to connect and control the smart appliances, the costs of smart appliances are drastically lowered. Instead of using the "add computer to appliance" approach by Whirlpool, only simple controller and wireless module are added to the smart devices as all the processing is done at the central hub.

While Iris Smart Home system uses the central hub approach that result in a cheaper price, only appliances or devices manufactured or approved by Iris can work with the smart home system. Iris smart home system also has the same vulnerability as Whirlpool – require internet service to be installed at home first. Without internet service, the Iris Smart Hub cannot access to the internet through the router.

2.4 Panasonic Smart Home System

Panasonic smart home (Panasonic.com, 2016) is another home automation system similar to Iris smart home system. It also uses a smart hub as central unit, which can control some devices and appliances made by Panasonic.

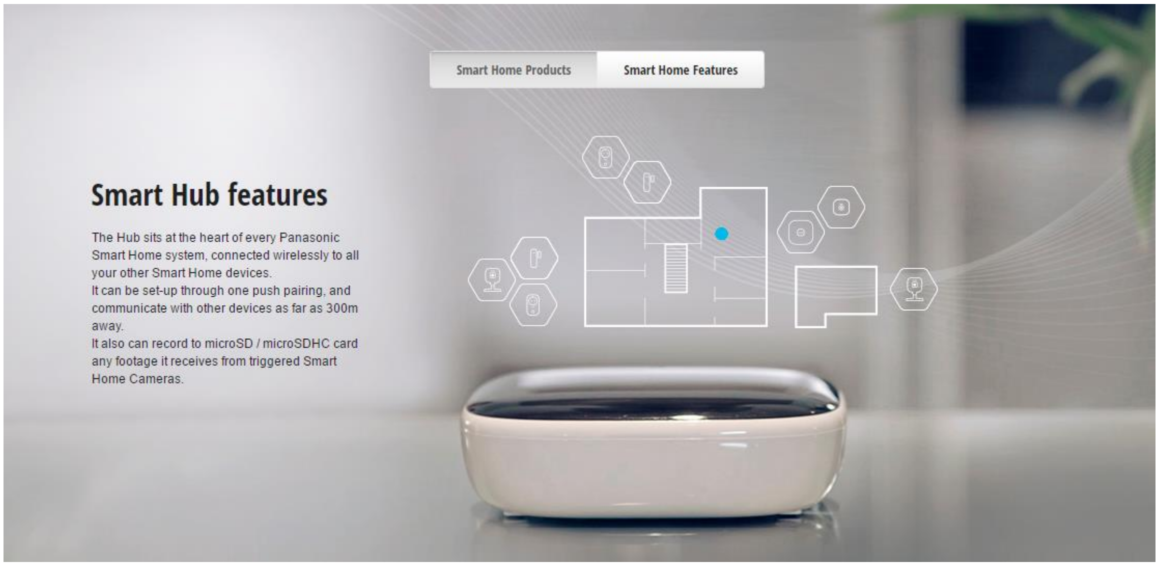


Figure 2.3 Panasonic Smart Hub

The smart hub communicates with user through a mobile app, allowing user to monitor and control the smart devices at home remotely. Similar to Iris, Panasonic smart home system also uses internet as a communication media between the smart hub and mobile app.

While the cost of Panasonic and Iris smart home system are still not exactly value for money, the idea of using a central hub for processing in order to cut down the cost works and has reduced the prices significantly. This concept is used in an article proposing a smart home system utilizing a flexible, low cost smart central hub, titled “Ubiquitous Smart Home System Using Android Application”.

2.5 Ubiquitous Smart Home System Using Android Application

The system presented by Kumar (Kumar, 2014) used an Arduino Ethernet Server to act as a central unit, which is lower in cost and more flexiblecomparing to purchasing a manufactured smart home system in the market such as Iris. The Arduino Ethernet is used to eliminate the use of a personal computer to keep the cost of the overall system as minimum as possible. This system can control devices such as light, switches, power plugs, temperature sensors and sirens. The overview of the proposed system is shown in Figure 2.4.

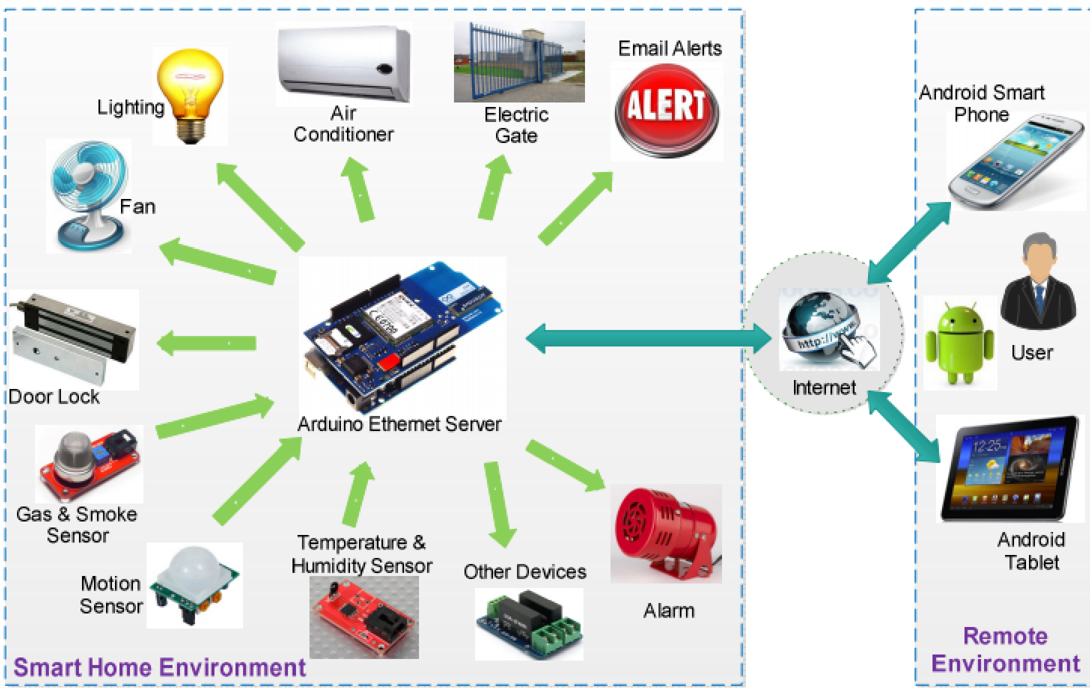


Figure 2.4 The System Architecture of Ubiquitous Smart Home

A smart home app is built and installed on an Android based smart phone. The app allows user to connect to the central unit through internet connection. User can monitor and control devices connected to the smart home system through the smartphone app. The general operation of the central unit is shown in Figure 2.5.

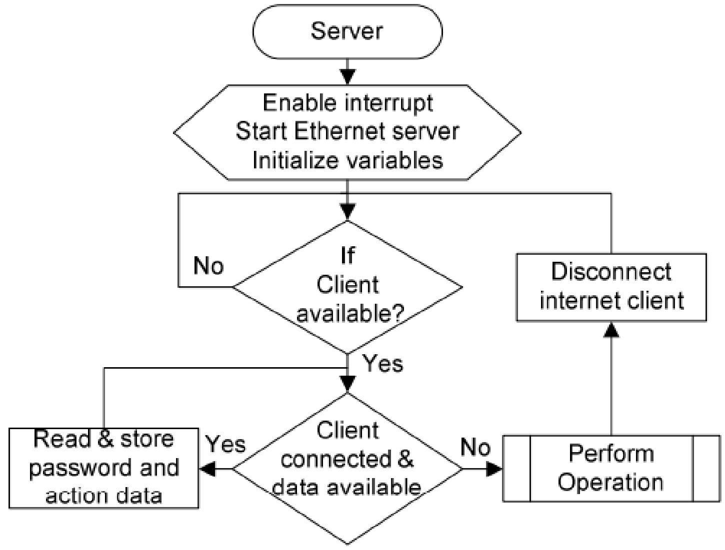


Figure 2.5 Central unit operations

This system can be programmed to match the wireless protocol of smart appliances, which means users can connect to almost any smart appliances that has wireless connectivity. This system also provide security capability as packets sent from the smart home app are encrypted and contain secret password to be identified by the central unit. Figure 2.5 shows how this is achieved.

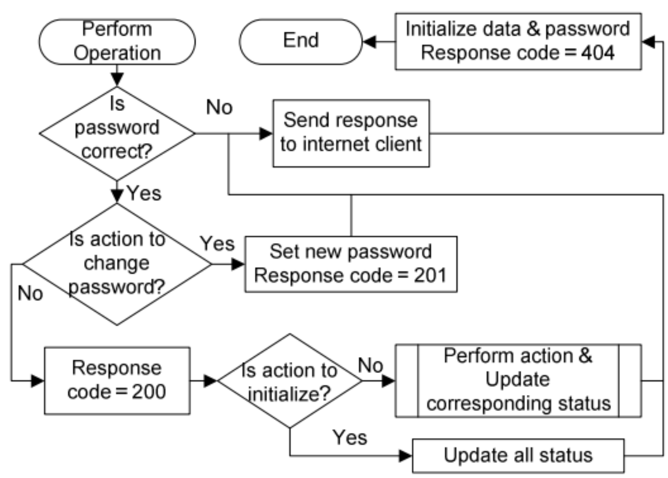


Figure 2.6 The central unit's perform operation sub-routine

While the idea of using Arduino board as an alternative of manufactured central unit to cut down the cost is great, its functionality still depends on the availability and stability of internet connection, similar to Whirlpool and Iris.

2.6 Home Security System Using GSM Modem

To solve the problem of internet instability, Potnis (Potnis, 2015) proposed an alternative approach by using GSM modem instead of internet connection between the users and the system. Although the system proposed in this article is not a smart home system like Whirlpool or Iris, the idea of using GSM connection instead of internet is interesting as it can solve the problem of internet unavailability or instability in places such as developing countries or rural area. Potnis's home security system connects a GSM modem with an ATmega16 micro-controller installed with IR transceiver. It uses the IR transceiver for intruder detection and when intruders are detected, GSM module is used to send call/SMS a number to inform the user.

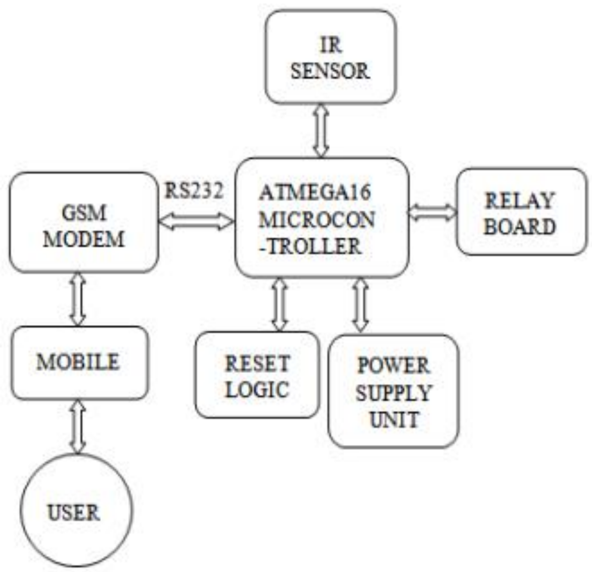


Figure 2.7 Block Diagram of entire system

As seen from the block diagram in Figure 2.7, GSM modem is used for two-way communication between the system and the user. The only problem of using this approach in a smart home system is that even if the internet connection of the previous smart home central units such as Iris Smart Hub is replaced with GSM modem, expensive smart appliances with internet capability are still required. An interesting article by I. Andrew (Andrew, 2014) about GSM infrared remote control seems to have the solution for this problem.

2.7 GSMRC – GSM Infrared Remote Control

Andrew (Andrew, 2014) implemented a system that controls infrared device remotely by using SMS instead of using wireless protocols like Iris. An Arduino board is connected to a Bluetooth module and IR receiver and transmitter. A mobile application is then developed and installed on an Android smart phone allowing it to be connected to the Arduino board through Bluetooth and assign IR code to buttons which can be customized.

Before using the system, user needs to add and customize buttons on the app, storing IR code for each button. To use the system, the user need to send a SMS containing a command, for example “airOn” to the android smartphone connected to the Arduino board. A mobile application that can read SMS is installed on the phone, and when the phone received the SMS with the command “airOn”, the app translates the command to IR code and sends it to the arduino board. The arduino board then transmits the IR signal through IR transmitter.

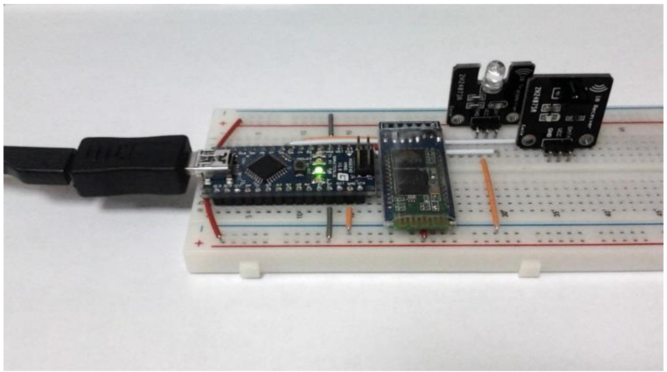


Figure 2.8 GSM Infrared Remote Control using Arduino

Andrew’s system used the cheap IR technology to control devices with built-in IR sensors, which includes a number of appliances, such as televisions, radios, curtains, fans, lighting and more. Infrared technology is cheap to implement in appliances, thus could be an alternative for the communications between the smart home central unit such as Iris Smart Hub with the appliances.

2.8 Comparison

Table 2.1 highlights and compares the pros and cons of all the reviewed systems.

Systems	**Pros**	**Cons**
Whirpool Smart Appliances	- Many features	- Very high cost - Require internet service at home
Iris Smart Home Management	- Use central unit to lower cost significantly	- Still high cost - Require internet service at home
Panasonic Smart Home System	- Use central unit to lower cost significantly - No monthly fees	- Still high cost - Require internet service at home
Ubiquitous Smart Home System	- Low cost central unit	- High cost to buy smart appliances with internet capability - Require internet service at home
Home security System GSM	- Good coverage and stability for connection between central unit and user(no need internet)	- Not a smart home system, can't control appliances -If used in smart home system, still need to buy expensive smart appliances
GSMRC	- Good coverage and stability - Can control appliances with built-in IR sensor (no need to buy expensive smart appliances)	- Is just a remote control - Can only control limited number of appliances.

Table 2.1 A brief comparison of reviewed systems

After the critical comparison of the reviewed systems, it appears that a smart home system with the following features will be a great alternative: has a central unit, controlled by users through GSM, and communicate with appliances through IR.

2.9 Chapter Conclusion

In this chapter, various systems and technologies related to smart home systems were studied and reviewed. Based on the findings, it is safe to conclude that home automation is still an under development field. The high implementation cost of smart home systems which utilize IoT hinders the marketability of smart home systems, where only limited number of consumers can afford to install the system. The complexity of smart home systems based on IoT also contribute to the fact that smart home systems are still uncommon. The utilization of internet connection in the smart home systems also greatly reduces the consumer amount in developing countries, where internet coverage and stability is still a serious issue.

All these findings prompt an invention of a smart home system that is cheap, simple, and independent from the use of internet connection, which is the goal of this project. By combining the GSM and IR to build a smart home system, this can be achieved.

CHAPTER 3: SYSTEM DESIGN

3.1 Overview

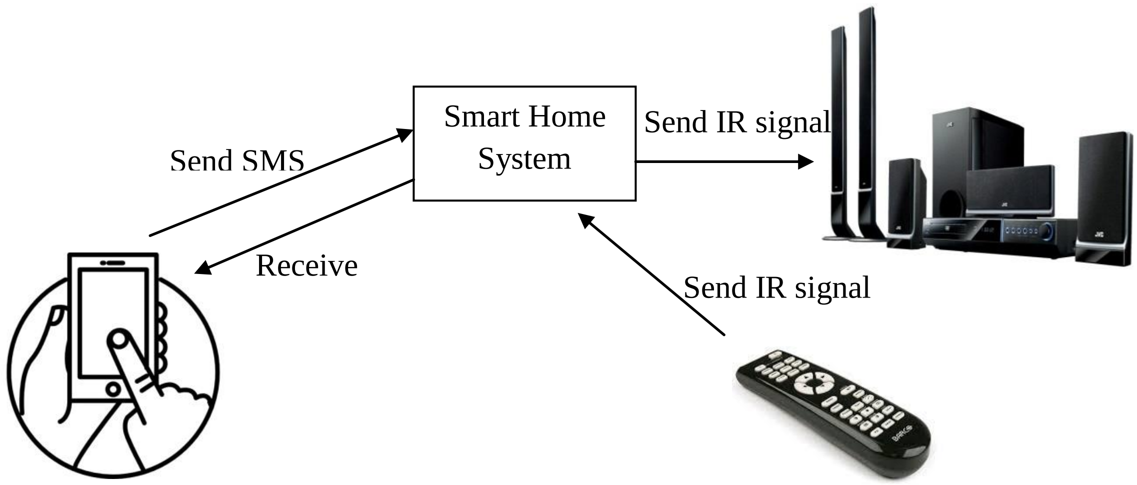


Figure 3.1 System Overview

The main function of the system is to receive SMS containing IR code, and then transmit them through its IR transmitter. However, it is impossible for user to know all the IR codes for every appliance. In another related project, a mobile app with IR library will be developed to help user communicate with the central unit. Instead of remembering all the IR codes and send them through SMS by themselves, the user only need to click buttons such as ‘on’ or ‘off’ in the mobile app. The mobile app will then retrieve the IR code from the IR library, and send it by SMS to the central unit.

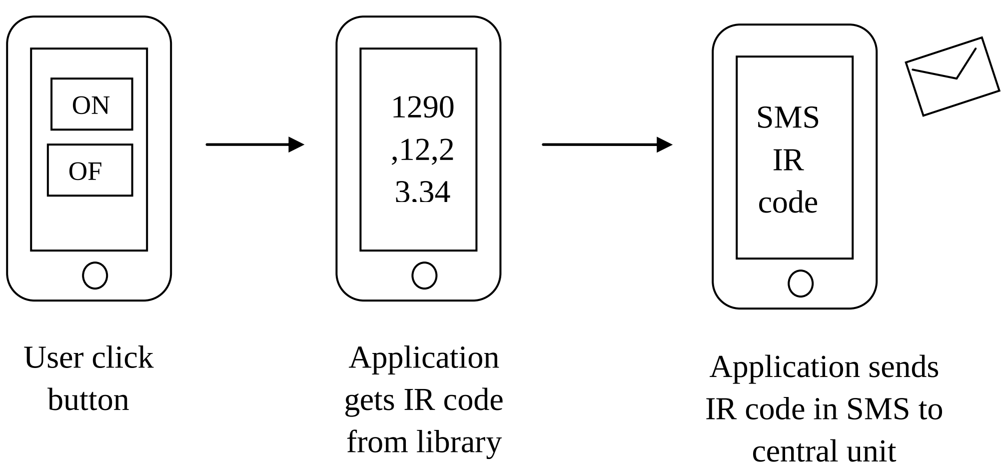


Figure 3.2 Control central unit with mobile application

However, it’s impossible to have a complete IR library since new products are introduced from time to time. It’s also inefficient to fit too much IR codes in the mobile application as it might fills up the phone storage, and most of them will never be used. Hence, this smart home system also comes with IR decoder function. The user can ask the system to decode the IR signal of their appliance, which will then be remembered by the mobile app.

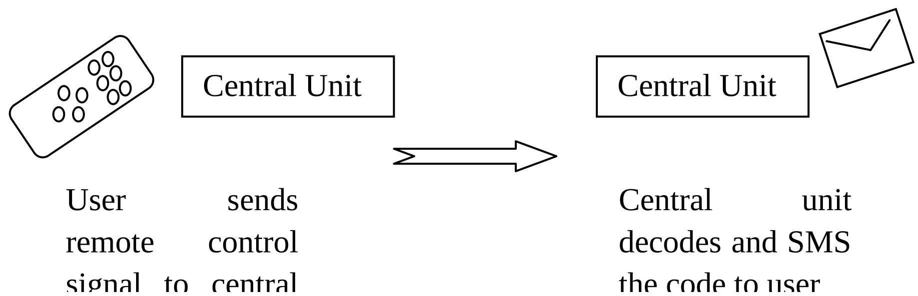


Figure 3.3 Decode IR remote control signals

Another issue is that IR signals cannot transmit through walls or when the line of sight is blocked. Hence in a typical home with multiple rooms, it will not be enough for just a central unit to transmit IR signals. To solve this problem, master-slave architecture will be used in building the smart home system as shown in Figure 3.4.

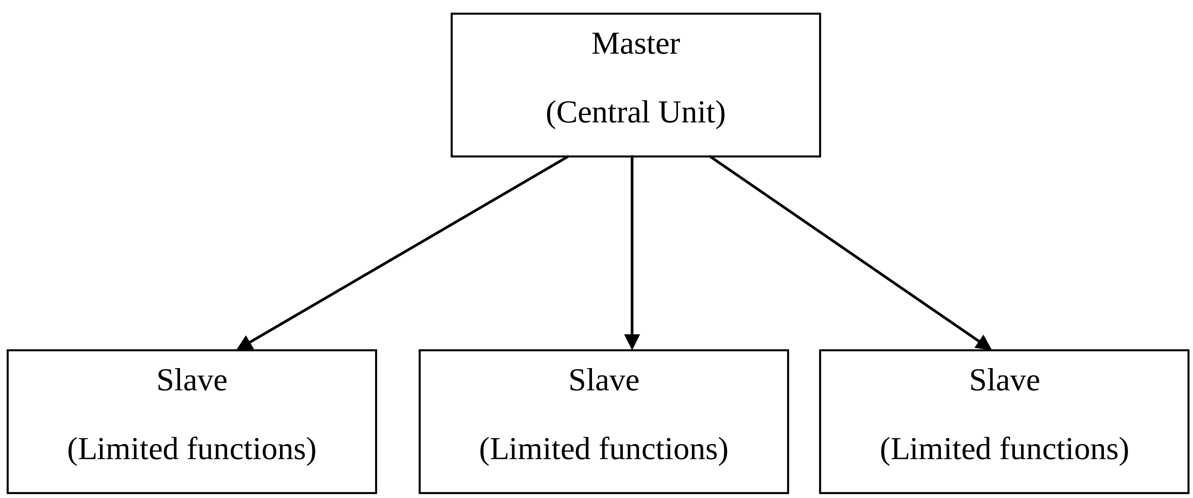


Figure 3.4 Master-slaves Architecture

The proposed smart home system in this report consists of a central unit, and several slave units. The central unit will have all the functions: decoding IR signal, transmitting IR signal, sending and receiving SMS, broadcasting through RF, while slave units have limited functions: receivingIR code through RF module and transmitting IR signals to reduce costs.

Each slave units has RF wireless module, and will be set to always listen for messages passing through central unit's RF wireless module. As a consequence, when the central unit needs to transmit IR signal, the IR code will also be sent to all the slave units to transmit the same IR signal.

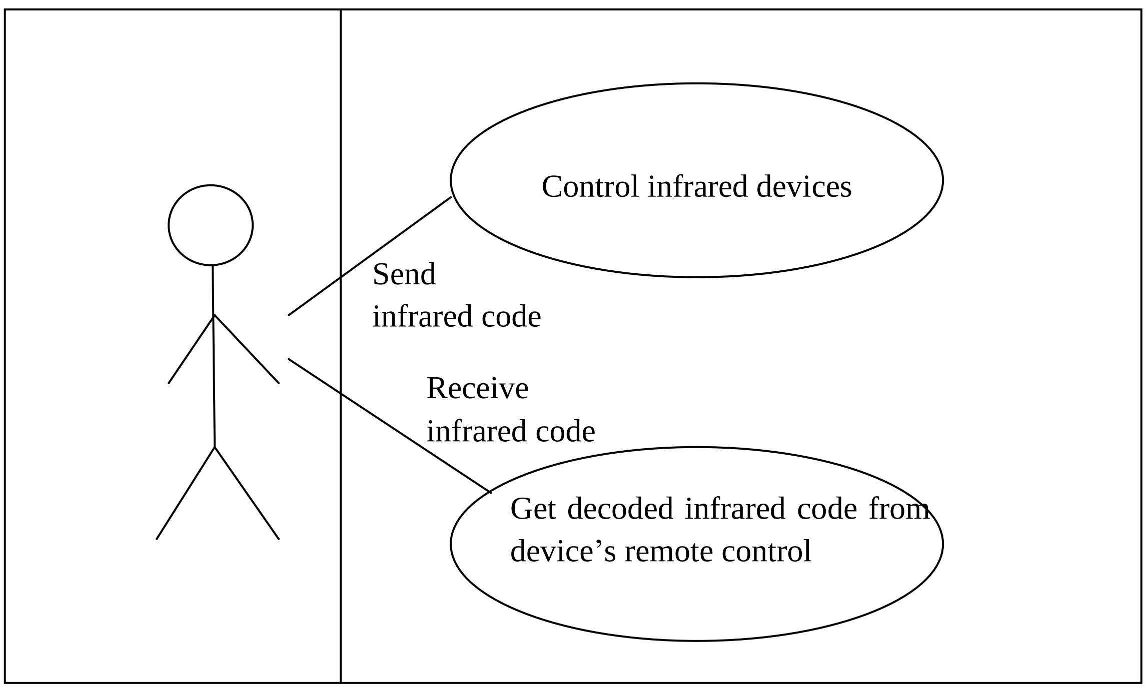


Figure 3.5 Use Case Diagram

As seen in the use case diagram in Figure 3.5, the two main functions of the system are control infrared devices and decode device's remote control infrared signal. For each of those, an activity diagram is drawn respectively as in Figure 3.6 and Figure 3.7.

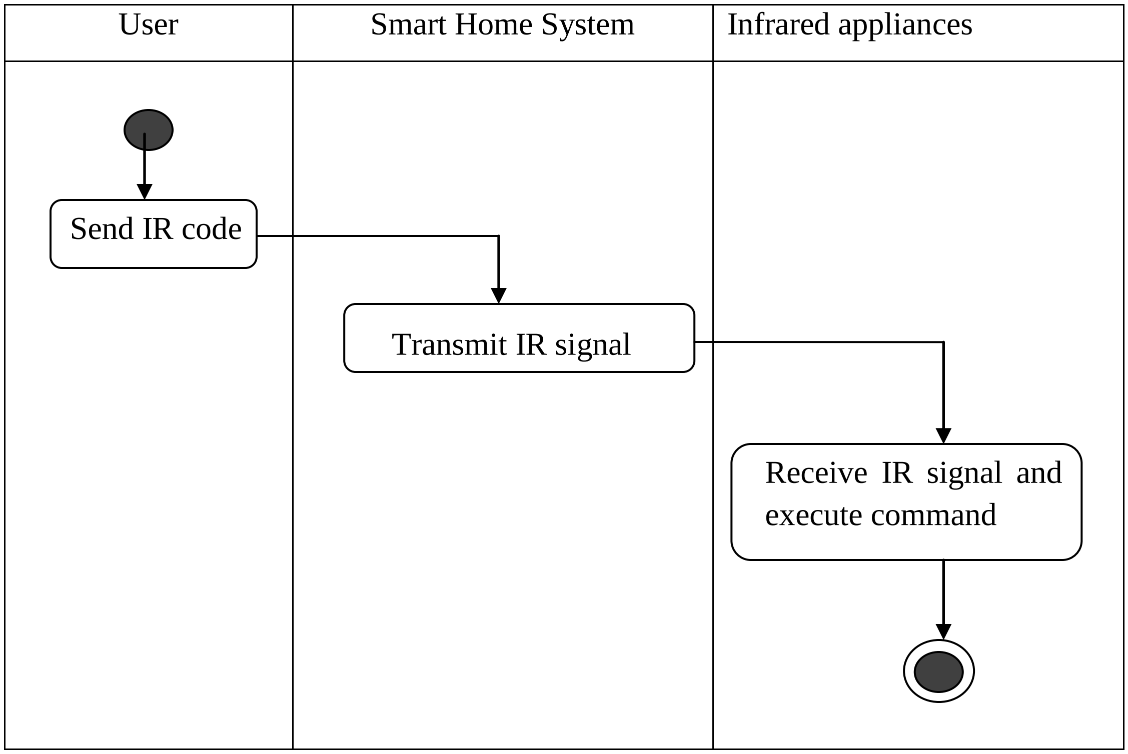


Figure 3.6 Transmit IR Activity Diagram

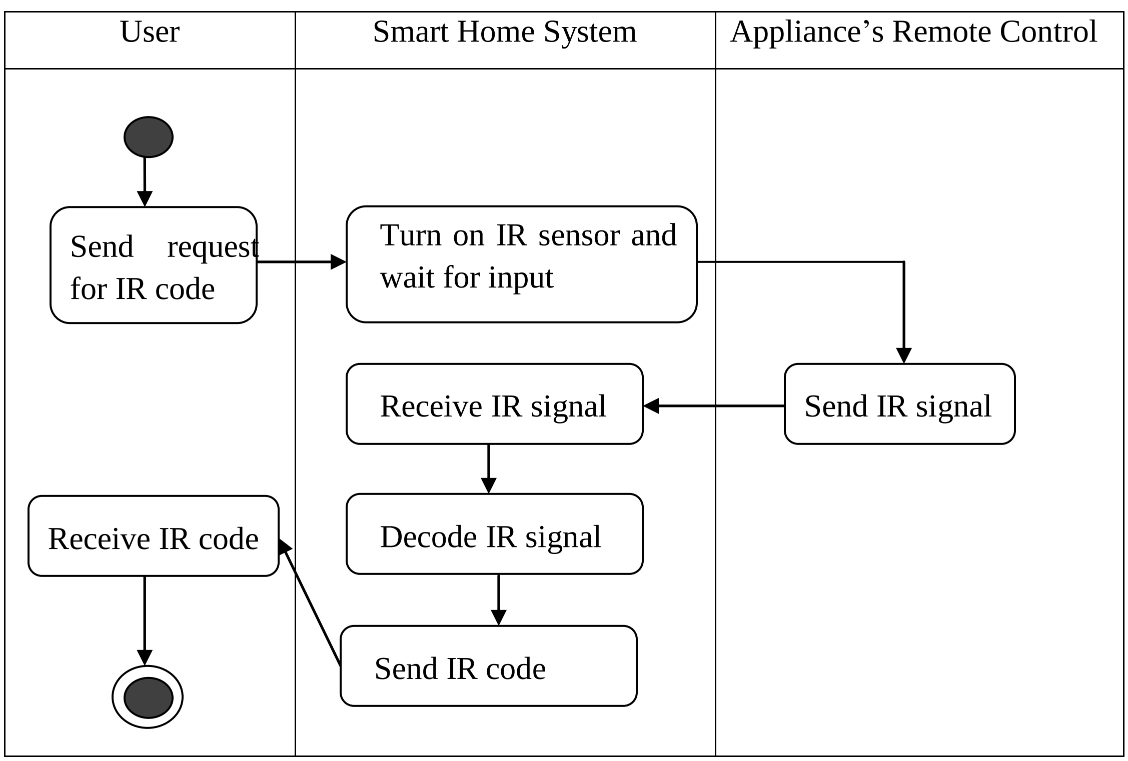


Figure 3.7 Decode IR Activity Diagram

3.2 Hardware

Arduino Uno is used as microcontroller in this project. It comes with 14 digital I/O pins (pin 0 to pin13) which are used to read or transmit data; and 6 analog pins (A0 to A5) which can be used to read analog sensors or act as general purpose digital I/O pins. It also has a 5V and a 3.3V power pin, which can be used to supply power to modules connected to the arduino. In the case that the power pins are not enough, digital I/O pin can be used to supply the power instead. Arduino Uno also comes with a VIN pin, of which voltage can be supplied through this pin to power the arduino. This is useful when batteries are used as power source for the arduino. Other options for supplying power to the arduino include USB connection and AC-to-DC adapter connecting to the power jack.

Other hardware that will be connected to the arduino in this project includes GSM module to send and receive SMS, IR transmitter to transmit IR signals, IR receiver sensor to read and decode IR signals, and RF wireless module to communicate with other unit through radio frequency. All of these modules have their own pins that need to be connected to the arduino. For each master and slave units, the modules connecting to the arduino will be different.

3.2.1 Master Unit

For the master unit (central unit), four different parts will be connected to the arduino: GSM module, IR transmitter, IR receiver, and wireless module. A schematic to show how the hardware will be connected is shown in Figure 3.8:

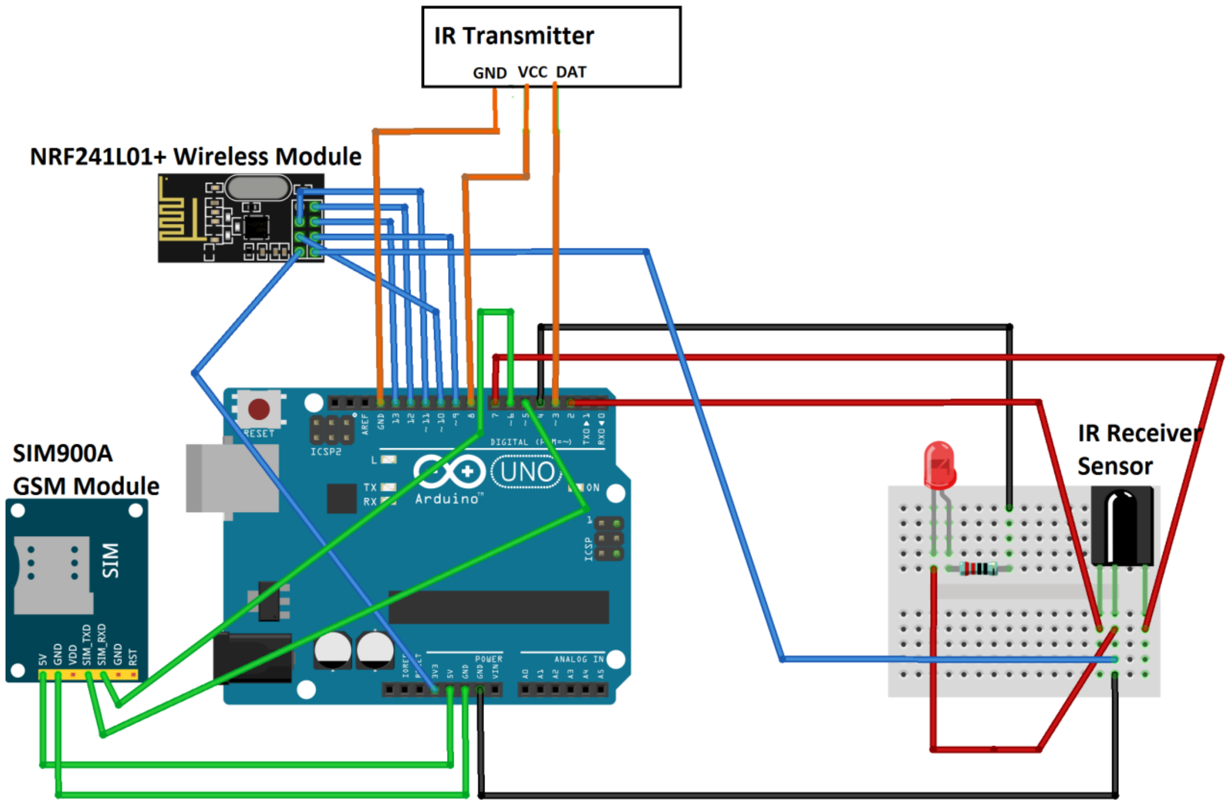


Figure 3.8 Central Unit Circuit Diagram

Due to the lack of GND pins on the arduino, the breadboard will mainly be used as a common ground for all wires connecting to GND. Some digital pins will also be used as power source since there is only a 5V and 3.3V power pin on the arduino. If mobility is required, batteries can be used to supply power through the VIN pin. The LED light function as indicator of the system and is optional.

1) GSM Module

Port	Arduino Pin	Function
VCC	5V	Provide power
TXD	~5	Transmit data to arduino
RXD	~6	Receive data from arduino
GND	GND	To ground the circuit

Table 3.1 GSM Module port table

The GSM module will be powered by the 5V pin of arduino. Normally, arduino will transmit data to GSM module by serial connection through its TXD pin to GSM module's RXD port, and vice versa. However in this case, pin 10 and 11 are used instead for the convenience of debugging. If TXD and RXD pins of arduino are occupied, codes cannot be uploaded to arduino since serial connection pass through these pins too. Therefore, two PWM output pins of arduino are chosen to emulate RXD and TXD for the connection with GSM module.

2) IR Transmitter

Port	Arduino pin	Function
VCC	8	Provide power
GND	GND	To ground the circuit
DAT	~3	Receive data from arduino

Table 3.2 IR Transmitter port table

Since there is limited power pin on arduino, digital pin 8 is used instead. Digital pin 8 will be set to HIGH when needed to provide the transmitter approximately 5V of power. The pin 9 of arduino has PWM output, enabling arduino to send interrupt to IR transmitter at HIGH state, which in turn allows arduino to send IR signal (Turn on/off according to IR code).

3) IR Receiver Sensor

Port	Arduino pin	Function
VCC	7	Provide power
GND	GND	To ground the circuit
DAT	2	Send data to arduino

Table 3.3 IR Receiver Sensor port table

Pin 8 of arduino will be used to power the IR sensor, whereas pin 9 will be connected to DAT of the sensor. When the IR sensor received IR signal, it will pass the raw data through its DAT port, which will be received by pin 9 of arduino.

4) RF Wireless Module

Port	Arduino Pin	Function
VCC	3.3V	Provide power
CE	~9	Chip enable activates RX or TX mode
CSN	~10	SPI chip select
SCK	13	SPI clock
MOSI	~11	SPI slave data input
MISO	12	SPI slave data output
GND	GND	To ground the circuit

Table 3.4 RF Wireless Module port table

This module will take up most of the pins on arduino. This module is used to share IR code that will be sent. This means that when there is an IR code that needs to be sent, the IR code will also be broadcasted through this module to other listening arduinos.

3.2.2 Slave Unit

There will only be two modules/parts connected to the arduino acting as slave: RF wireless module and IR transmitter. No breadboard will be required as there will be enough pins to connect all parts. The schematic is shown in Figure 3.9:

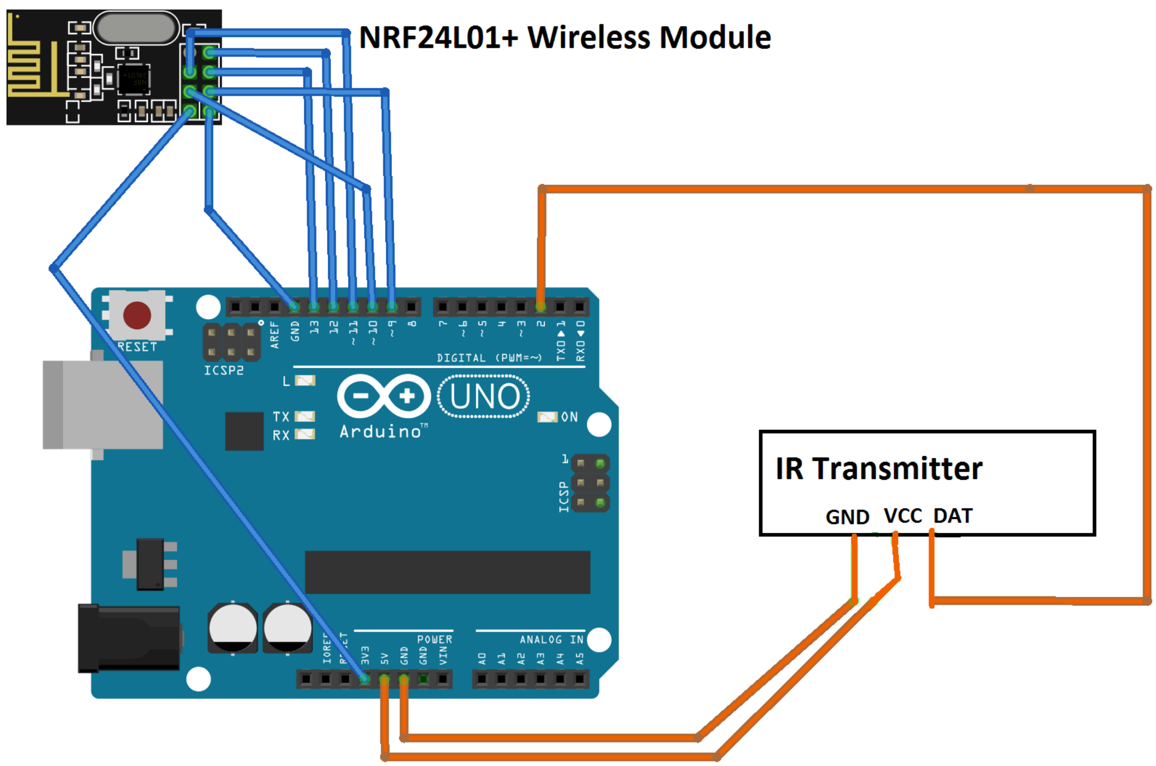


Figure 3.9 Slave Unit Circuit Diagram

As seen in the schematic, the slave unit is just a simplify version of the central unit. The connections are slightly different due to the less complicated circuit.

1) RF Wireless Module

Port	Arduino Pin	Function
VCC	3.3V	Provide power
CE	~9	Chip enable activates RX or TX mode
CSN	~10	SPI chip select
SCK	13	SPI clock
MOSI	~11	SPI slave data input
MISO	12	SPI slave data output
GND	GND	To ground the circuit

Table 3.5 Slave Unit RF Wireless Module port table

The connections between RF wireless module and arduino of the slave unit will be exactly the same as the central unit. This means the code for RF module in central unit can be reused without much change, which in turn simplifies the implementation of slave unit.

2) IR Transmitter

Port	Arduino pin	Function
VCC	5V	Provide power
GND	GND	To ground the circuit
DAT	~3	Receive data from arduino

Table 3.6 Slave Unit IR Transmitter port table

The connections between IR transmitter and arduino remain about the same, except the power source. The VCC port of IR transmitter is connected to the 5V pin of the arduino for a more stable voltage and current. This connection is more preferable but impossible to implement in the central unit since the GSM module needs the stable power more. Hence, the slave unit theoretically has slightly better coverage in terms of transmitting IR signals.

3.3 Software

The overall system process flow is shown in Figure 3.10:

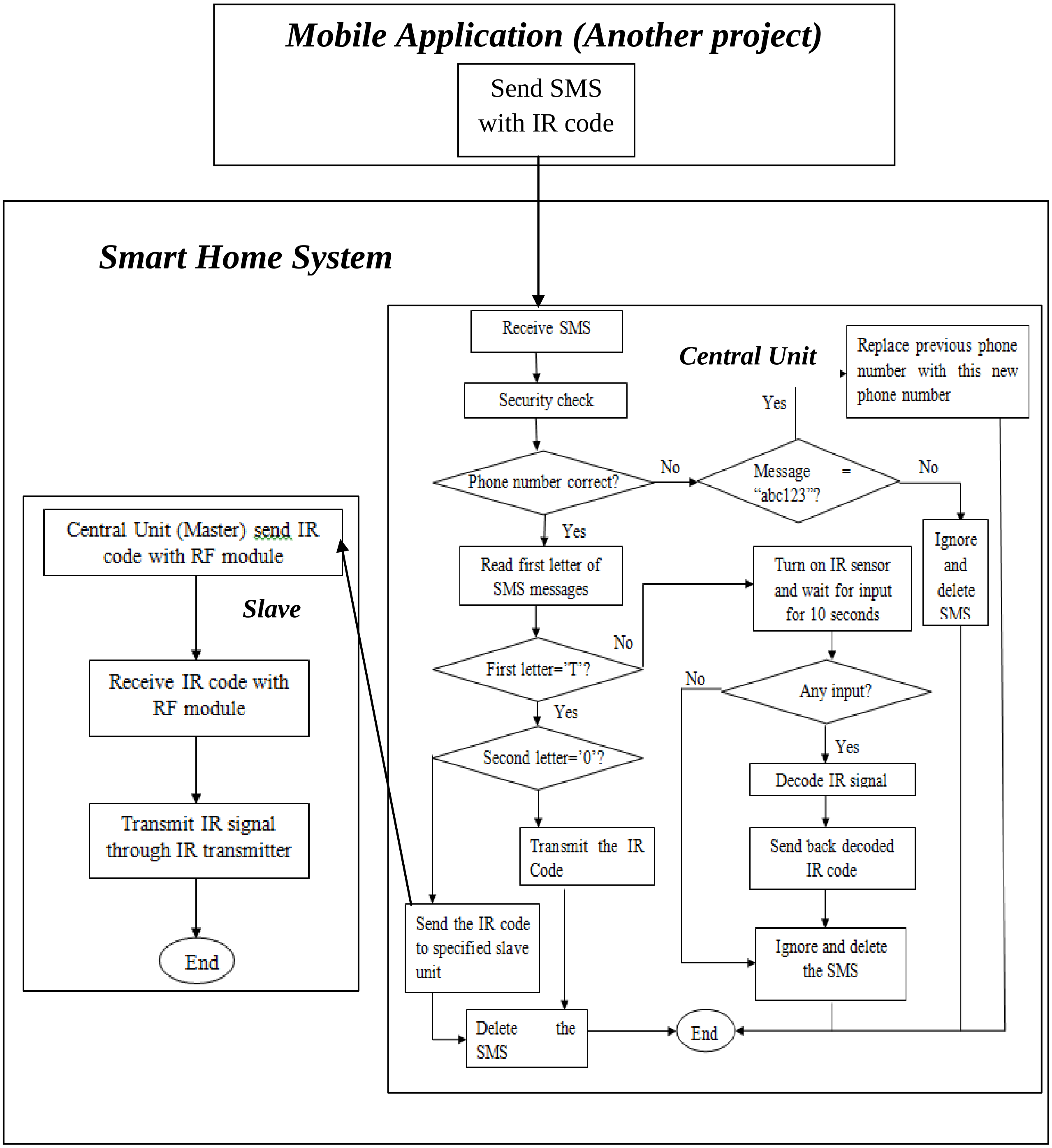


Figure 3.10 Flow Chart of Overall System

3.3.1 Master Unit

A system flow chart for the central unit acting as master is shown in Figure 3.11 to illustrate how the system works from the software perspective.

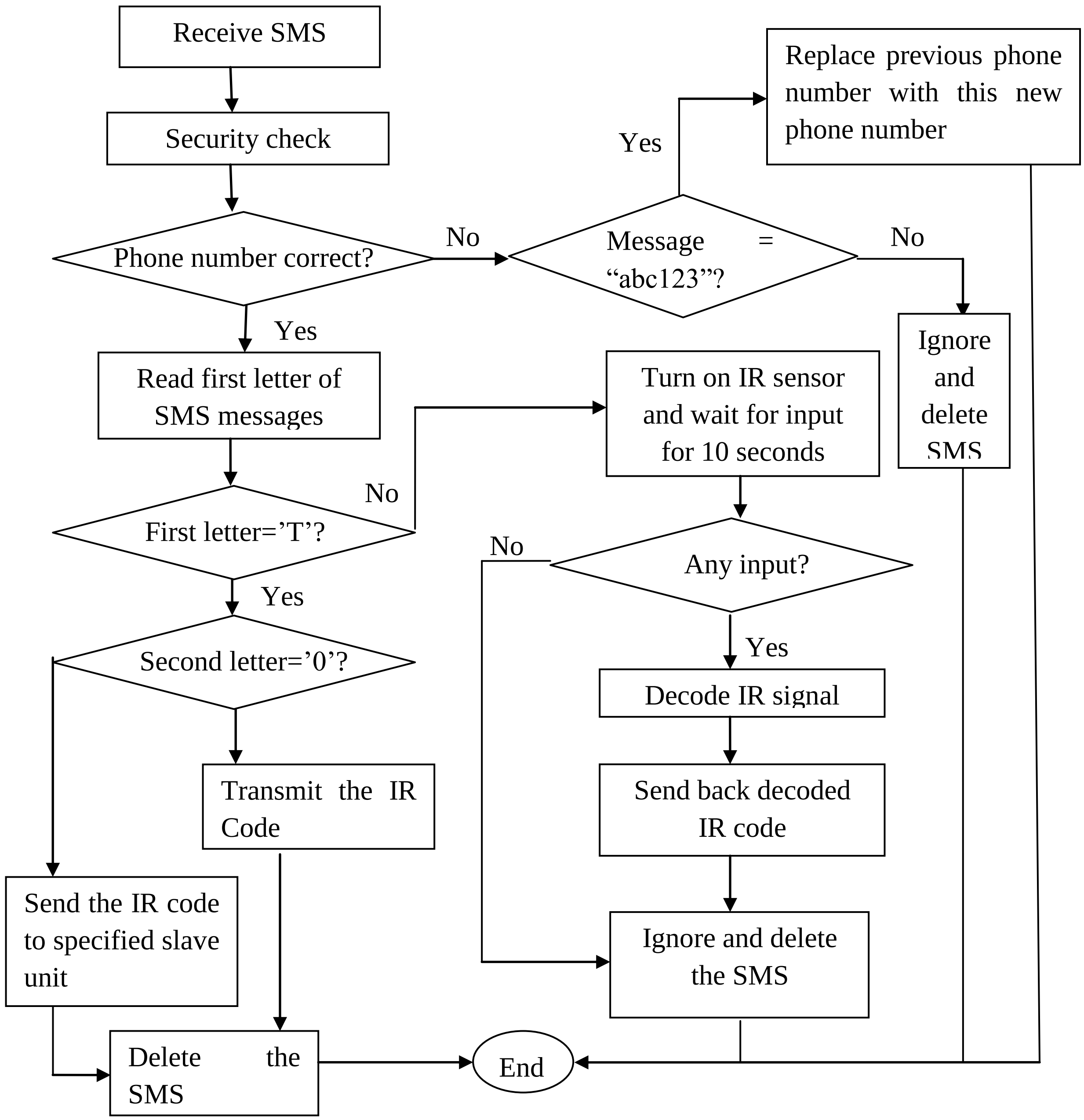


Figure 3.11 Central Unit Flow Chart

The central unit will always be on standby to receive any incoming SMS. When it receives a SMS, it will check which phone number the SMS comes from. If the phone number matches the preset phone number in the central unit, it means the SMS comes from legit source. This helps secure the system from potential attempts to control the smart home system from unknown sources. If the phone number doesn't match, the system will check if the message contains the secret key "abc123". If it does, the existing phone number kept in the system will be replaced with the new phone number the message was sent from. After that, the SMS will be deleted and the system will wait for the next SMS. If the phone number is verified, the system will then check if the SMS message starts with the letter D or T.

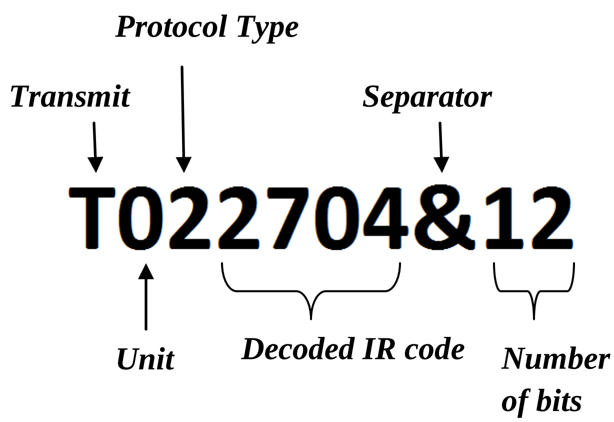


Figure 3.12 Formatted IR code

If the message starts with 'T', it means the user wants to perform transmit IR signals. The format of the transmit IR code format is shown in Figure 3.12. The system will then check the second letter. The second letter tells the system which unit should transmit the IR code: master unit, slave unit 1, slave unit 2, or slave unit 3 etc. The system will read the rest of the message which contains Protocol Type, decoded IR code, separator and number of bits. If the unit is 0, the IR code will be transmitted through the master unit. If the unit is anything other than 0, the system will send the IR code to specified slave unit through its RF wireless module. The slave unit will pick up the IR code and transmit them through their own respective IR transmitter too. This way, the IR code can reach the whole house if slave units are placed in other rooms where the central unit cannot reach. When all these are done, the system will delete the SMS and go back to waiting for the next SMS. The SMS need to be deleted since they are stored in the SIM card of the GSM module, which has limited storage.

On the other hand, if the message in the SMS starts with ‘D’, it means the user wants to decode an IR remote signal. The system will turn on its IR sensor, and look for an IR signal for the next 10 seconds. If no IR signal is detected after 10 seconds, the system will turn off IR sensor, delete the SMS and return to waiting for the next SMS. If an IR signal is detected, the system will decode the IR signal, and then send the decoded IR code through SMS back to the user.

3.3.2 Slave Unit

The system flow chart for the slave unit is different, as illustrated in Figure 3.13.

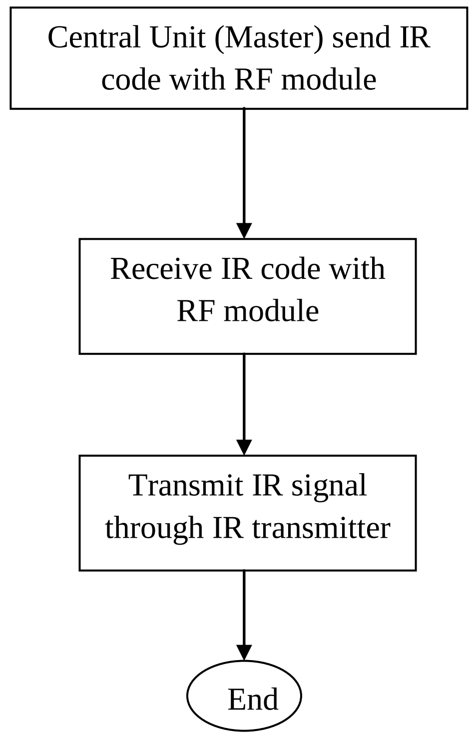


Figure 3.13 Flow Chart of Slave Unit

The RF module of the slave unit will always be listening on the RF module of the central unit. If a message is passed through the RF module of the central unit, the slave unit will pick up the message, which is IR code. The IR code will then be transmitted through its IR transmitter.

3.4 Implementation

The first step to implement the system is to assemble the hardware as described previously, resulting in one master unit and one slave unit.

After the hardware is done, install the arduino software, Arduino IDE on a computer. In this example, the software is installed in the path: C:\Program Files (x86)\Arduino as shown in Figure 3.14.

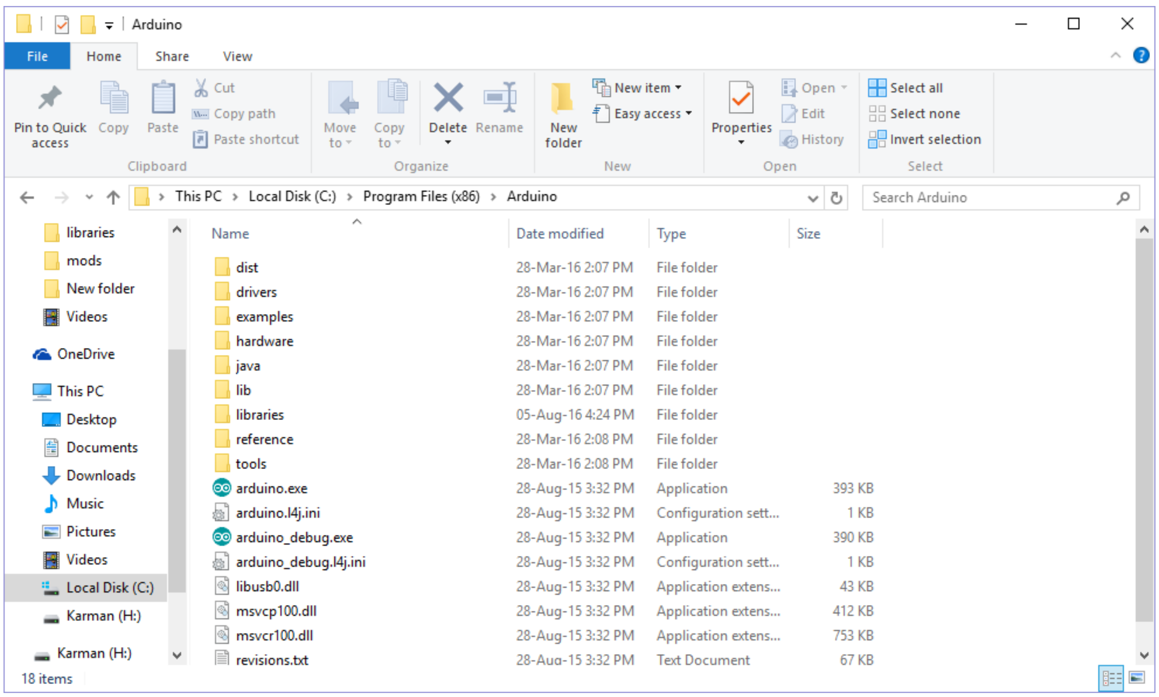


Figure 3.14 Arduino IDE Folder

Next, open the libraries folder and put in the following libraries: IRLib, RF24, SoftwareSerial and SPI. After the libraries are placed, open the file Smart_Home_Master.ino with Arduino IDE and connect the master unit to the computer through USB Cable. The Arduino IDE should be able to detect the unit and will be assigned a port number as shown in Figure 3.15.

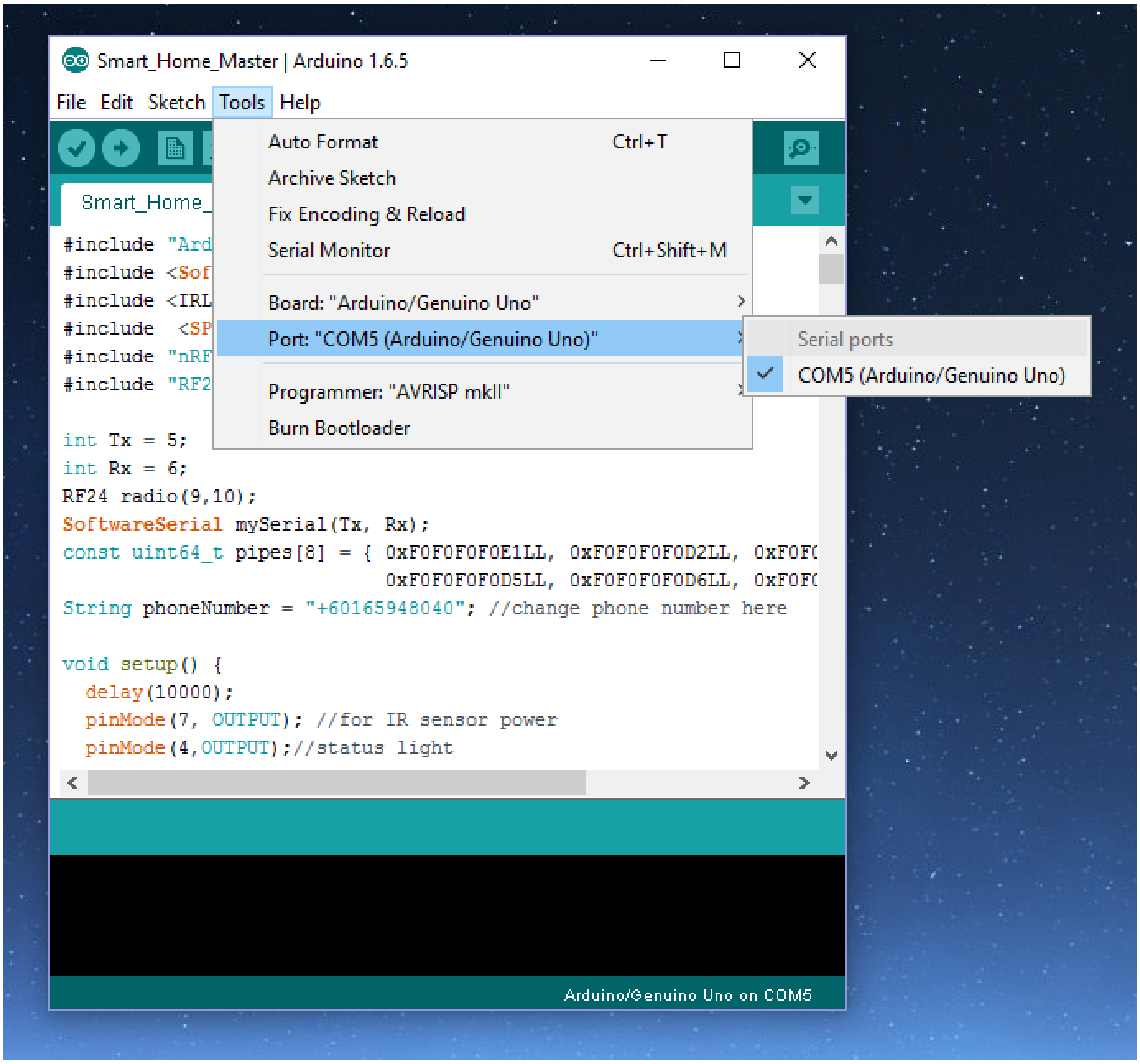


Figure 3.15 Arduino IDE Port

After choosing the serial port, click the upload button to upload the code to the master unit. After the uploading is done, the master unit can be unplugged from the computer and powered by other source such as a power bank.

For the slave unit, repeat the same steps by using Smart_Home_Slave.ino instead. Figure 3.16 and 3.17 shows the functioning master unit and slave unit.

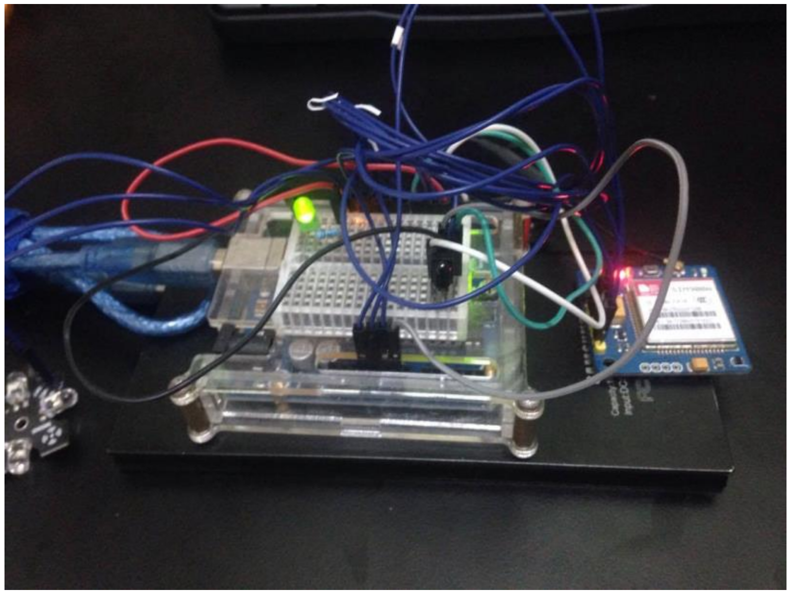


Figure 3.16 Master Unit

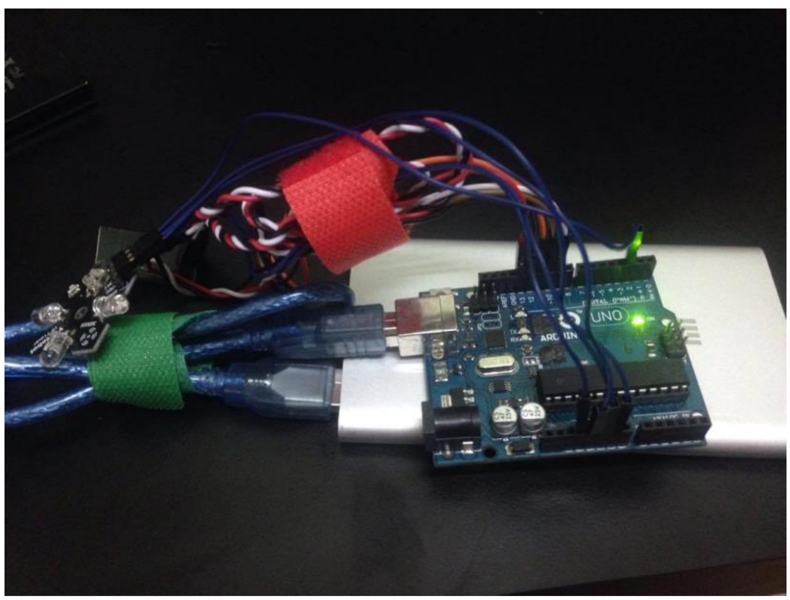


Figure 3.17 Slave Unit

CHAPTER 4 DESIGN SPECIFICATION

4.1 Methodologies and General Work Procedures

A combination of waterfall model and prototyping model is chosen as the methodology for the realization of this project. Firstly, the requirements to conduct the project will be researched and determined. The functions required for this system are determined at this stage. After all the requirements are researched and determined, an iteration circle of design, implementation and verification will be conducted to build prototypes for each different function.

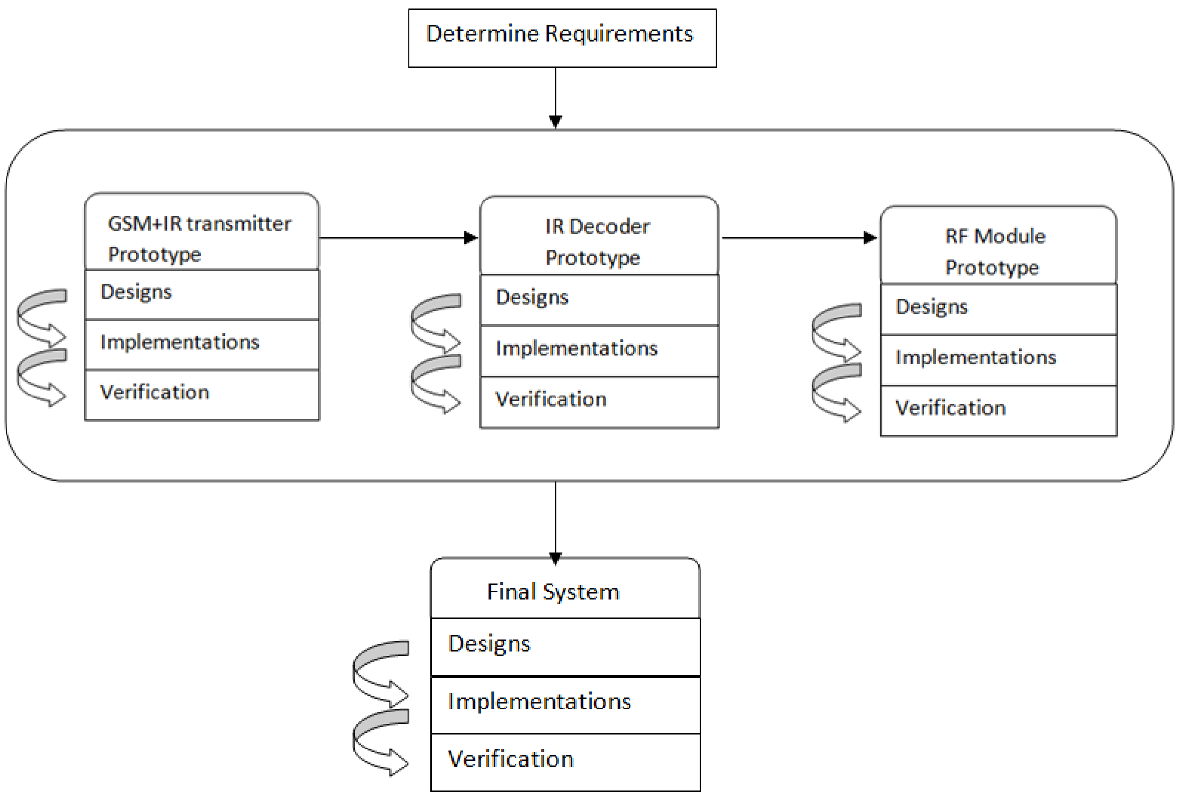


Figure 4.1 Methodology flow

Firstly, a prototype that can receive SMS, and able to transmit IR command contained in the received SMS will be designed, implemented and then verified. Next, a prototype that can read and decode IR signal from any IR remote control, then send a SMS containing the decoded IR signal will be built. The third prototype will be an arduino that can send messages to other arduinos through RF. Lastly, a final prototype that can combine the functions of all the previous three prototypes will be designed, implemented and verified.

4.2 Tools to use

1) Arduino Uno Rev 3

Figure 4.2 Arduino UNO Rev 3

Arduino Uno Rev 3 is the chosen microcontroller for this project. It is based on the Atmega328P, comes with 14 digital input/output pins (of which 6 can be used as PWM outputs), 6analog inputs, a 16MHz quartz crystal, and a USB connection. This board has enough ports to connect all the other modules and parts needed in this project.

2) SIM900A GSM Module

Figure 4.3 SIM900A GSM Module

The GSM module chosen in this project is a modified GSM module that has reduced size and comes at a lower price. It is a complete Dual-band GSM/GPRS module delivering 900/1800MHz performance for voice, data, SMS, and fax with low power consumption. It is used to send and receive SMS in this project.

3) Multi-directions IR transmitter

Figure 4.4 Multi-Directions IR Transmitter

This transmitter comes with 4 IR transmitter LEDs, allowing IR signals to be sent in all directions. It is used in this project to provide a wide coverage when transmitting IR signals to appliances.

4) TSOP38238 IR Receiver Sensor 38 KHz

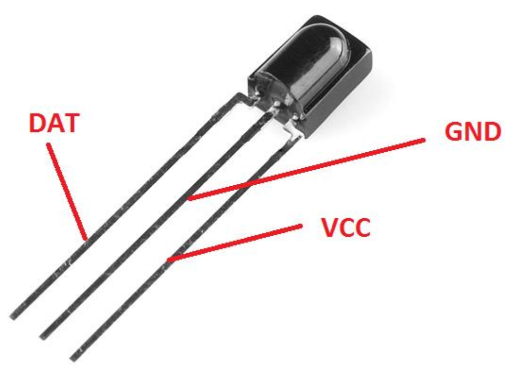


Figure 4.5 IR Receiver Sensor

This is an IR sensor designed to receive IR signals at 38 KHz frequency. It runs at 3V to 5V, so any microcontroller can easily power it. This sensor does not decode the IR signal received, just passes the raw data along.

5) NRF24L01+ RF Wireless Module

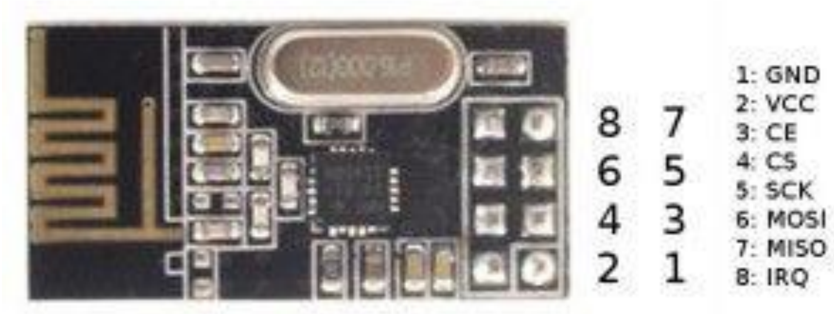


Figure 4.6 RF Wireless Module

This is a highly integrated, ultra-low power 2Mbps RF transceiver. IT operates at 2.4GHz, and can be powered by just 1.9V to 3.6V.This is used in this project to enable multiple arduinos to communicate with each other.

6) AC/DC Adapter 12V/2A

Figure 4.7 AC/DC Adapter

This adapter is used to provide power for arduino in this project. Alternatively, batteries or power bank can be used.

7) Breadboard

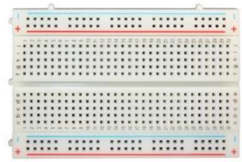


Figure 4.8 Breadboard

Since there is limited GND (ground) port in arduino, breadboard is used in connecting a GND port of arduino with the GND port of all other modules and parts.

8) Wire Jumpers

Figure 4.9 Wire Jumpers

These wires include male-to-female, female-to-female, and male-to-male wires. They are used throughout the project to connect arduino with other modules and parts.

4.3 System Performance Definition

The system performance is defined by mainly three aspects: coverage, response time, and compatibility.

- Coverage is determined by the ability of the system's transmitted IR signal to reach as much appliances and devices as possible in all angles. The more appliances the system can cover, the better it is.
- Response time is determined by how fast the central unit can process requests by user and respond correspondingly. For example, the amount of time it takes to receive a SMS containing IR signal, process the message and transmit the IR signals. A good response time is defined by how short this amount of time is.
- Compatibility is determined by how many IR appliances the system can control. No existing IR library has the IR codes of all IR appliances. Thus, the system is equipped with the ability to decode IR signal from an IR remote control, so that it can emulate the signal when needed. The more types of appliances' remote control the system can decode, the better the compatibility is.

4.4 Verification Plan

The system will be tested by using it control several IR appliances, such as CD player and radio. A simple IR LEDs lamp will also be built by connecting an arduino to multiple LEDs and an IR sensor. The lamp will light up different LED based on the IR signal received.

CHAPTER 5: INFRARED DECODING AND TRANSMITTING

5.1 Infrared Basics

Infrared light can be found everywhere, such as light bulb, sun, or fire. It is mostly invisible and is useful in transferring data. Infrared signal is basically turning on and off infrared light very fast in an organized sequence. A technique called modulation is used to reduce data loss caused by noise or interference of other natural source of infrared light such as sun ray. The IR LED will be turned on and off at a frequency – most common is 38kHz, which means it will be on for roughly 13.15 microseconds (see calculation in figure 5.1), and off again for the same duration, repeatedly.

38kHz = 1/38000 seconds as one cycle of IR LED, therefore

One cycle of IR LED = 26.3 microseconds

Each cycle consist of two phase, ON and OFF, thus

ON phase = 26.3 microseconds / 2

= 13.15 microseconds

OFF phase = 13.15 microseconds

Figure 5.1 Calculation

To send or read data, the duration of turning the IR LED on and off at 38 kHz is measured. For example, modulating IR LED at 38kHz for 2000 microseconds, keeping it off for 50 microseconds, then again modulating it for 2000 microseconds, and finally turn off the LED. The duration of the modulated IR signal is on is called mark, whereas the duration of the modulated IR signal is off is called space as shown in Figure 5.2.

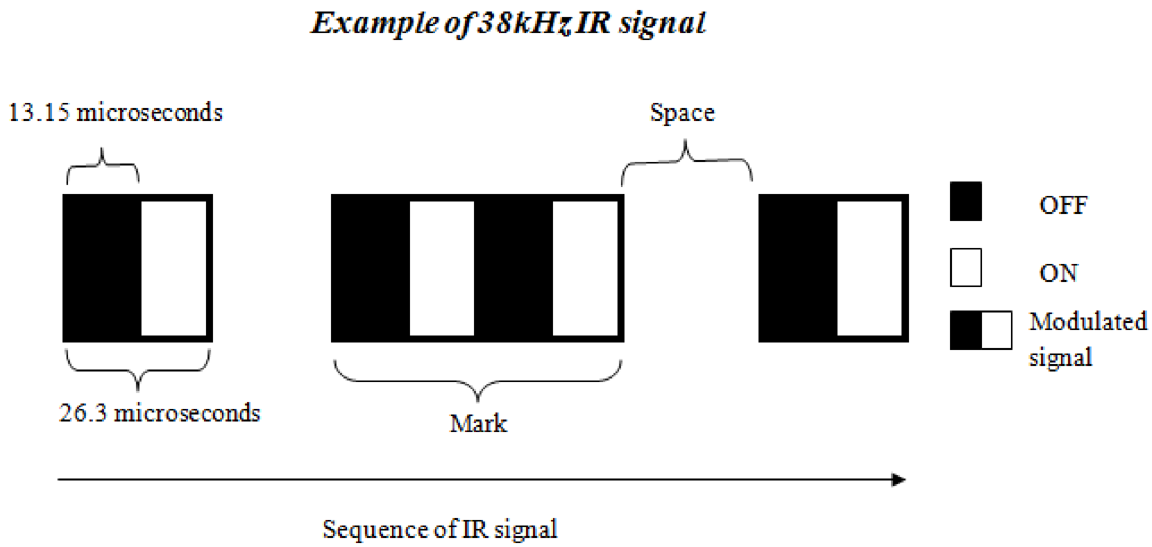


Figure 5.2 Example of 38kHz IR signal

The sequence and duration of mark and space are dependent on which IR protocol is used. Some IR protocol repeat the sequence a few times, whereas some protocol use special identifier in the beginnings – every protocol has their own way to send the sequence of modulated IR signals. Therefore, to be able to transmit and decode IR signals of manufactured remote controllers, it is fundamental to find out what protocol is used and how does the protocol works.

5.2 IR Protocols

In this project, a library called IRLib is used. The library supports IR protocols of most devices out there, allowing easier decoding and transmitting of IR signals. In total, there are 7 protocols supported by the library: NEC, Sony, RC5, RC6, Panasonic, NECx, and JVC. However, Panasonic will not be covered as it used a 38 kHz modulation, which is not supported by the smart home system in this project. Aside from brands like Sony and NEC who develop their own protocol, most other brands such as Phillips use more common protocols such as RC5 and RC6. Each of these protocols has their own way of transmitting sequence of modulated IR signals. In IRLib, when an IR signal is received, the first step is to identify what protocol is used to transmit the signal, and then decode the signal using the protocol into a value.

5.2.1 SONY

IR signal using Sony Protocol can only be 8, 12, 15, or 20 bits in length. Thus, this can be used to determine if Sony Protocol is used. If the IR signal is indeed the same length as specified, a generic function, decodeGeneric will be used to decode the signal with specific parameters for Sony Protocol. Figure 5.3 shows the code for decoding function of Sony.

```
bool IRdecodeSony::decode(void) {
  IRLIB_ATTEMPT_MESSAGE(F("Sony"));
  if(rawlen!=2*8+2 && rawlen!=2*12+2 && rawlen!=2*15+2 && rawlen!=2*20+2)
      return RAW_COUNT_ERROR;
  if(!decodeGeneric(0, 600*4, 600, 600*2, 600, 600,0)) return false;
  decode_type = SONY;
  return true;
}
```

Figure 5.3 Decoding for Sony

The function decodeGeneric use a general routine because most protocols have the same basic structure. This means that some other protocols in the library will also be using this function to decode IR signals. However, the function has 7 parameters, which are different for each protocol. For Sony protocol, the 7 parameters are 0, 600*4, 600, 600*2, 600, 600, and 0, which are basically length of marks, length of spaces etc. Inside the decodeGeneric function, it will check whether the parameters fit the raw data. If it doesn't fit, it means it is not Sony protocol and will exit the function, and try again with another protocol which has the same requirement (8, 12, 15, or 20 bits in length).

To transmit a value using Sony protocol, another generic function sendGeneric is used for the same reason: most protocols have the same basic structure.

```
void IRsendSony::send(unsigned long data, int nbits) {
  for(int i=0; i<3;i++){
     sendGeneric(data,nbits, 600*4, 600, 600*2, 600, 600, 600, 40,
             false,((nbits==8)? 22000:45000));
  }
};
```

Figure 5.4 Transmitting for Sony

However, it uses a variable length mark and a fixed length space rather than a fixed mark and a variable space. Therefore, an "if else" condition is used to check the number of bit to determine the length mark. Aside from that, the protocol requires the command to be sent at least three times, so the sendGeneric function is repeated three times.

5.2.2 NEC

For NEC, the only way to know if an IR signal is encoded by using NEC protocol is to use the generic function decodeGeneric with the specific parameters as shown in Figure 5.5.

```
#define NEC_RPT_SPACE   2250
bool IRdecodeNEC::decode(void) {
  IRLIB_ATTEMPT_MESSAGE(F("NEC"));
  // Check for repeat
  if (rawlen == 4 && MATCH(rawbuf[2], NEC_RPT_SPACE) &&
    MATCH(rawbuf[3],564)) {
    bits = 0;
    value = REPEAT;
    decode_type = NEC;
    return true;
  }
  if(!decodeGeneric(68, 564*16, 564*8, 0, 564, 564*3, 564)) return false;
  decode_type = NEC;
  return true;
}
```

Figure 5.5 Decoding for NEC

One thing to note about NEC is that it uses a special sequence of marks and spaces that mean "I held down the button so you should repeat whatever I sent you last time". This is useful in remote controller so that for example, user can just press the volume up button without releasing it to keep turning up the volume. This special sequence is checked in the first "if" condition, if it is detected, there is no need to call decodeGeneric as the value will be set to 0xFFFFFFFF. When this value is received by the transmit function of NEC in the library, a specific sequence of mark and space will be sent so that the device received it will repeat whatever command it received last time, as shown in Figure 5.6.

```
void IRsendNEC::send(unsigned long data)
{
  if (data==REPEAT) {
    enableIROut(38);
    mark (564* 16); space(564*4); mark(564);space(56*173);
  }
  else {
    sendGeneric(data,32, 564*16, 564*8, 564, 564, 564*3, 564, 38, true);
  }
};
```

Figure 5.6 Transmitting for NEC

If the value is not 0xFFFFFFFF, then the generic function sendGeneric will be used to transmit the value instead.

5.2.3 NECx

NECx is identical to NEC except for the length of the initial header mark. There is also no special sequence to ask the device to repeat command receive previously. The code for this is shown in Figure 5.7.

```
bool IRdecodeNECx::decode(void) {
  IRLIB_ATTEMPT_MESSAGE(F("NECx"));
  if(!decodeGeneric(68,564*8,564*8,0,564,564*3,564))
      return false;
  decode_type = NECX;
  return true;
}
```

Figure 5.7 Decoding for NECx

As seen in Figure 5.7, NECx is also using decodeGeneric function with the same parameters, except for the second parameter which stands for the length of header mark. To transmit a value via NECx protocol, sendGeneric function is used again with specific parameters as shown in Figure 5.8.

```
void IRsendNECx::send(unsigned long data)
{
  sendGeneric(data,32, 564*8, 564*8, 564, 564, 564*3, 564, 38, true, 108000);
};
```

Figure 5.8 Transmitting for NECx

5.2.4 JVC

```
bool IRdecodeJVC::decode(void) {
  IRLIB_ATTEMPT_MESSAGE(F("JVC"));
  if(!decodeGeneric(36,525*16,525*8,0,525,525*3,525))
  {
     IRLIB_ATTEMPT_MESSAGE(F("JVC Repeat"));
     if (rawlen==34)
     {
        if(!decodeGeneric(0,525,0,0,525,525*3,525))
           {return IRLIB_REJECTION_MESSAGE(F("JVC repeat failed generic"));}
        else {
           if (MATCH(rawbuf[4],(525*3)))
           {
              value |= 0x8000;
           }
           else
           {
             if (!MATCH(rawbuf[4],525)) return DATA_SPACE_ERROR(525);
           }
        }
        bits++;
     }
     else return RAW_COUNT_ERROR;
  }
  decode_type =JVC;
  return true;
}
```

Figure 5.9 Decoding for JVC

As shown in Figure 5.9, JVC needs more processing when using the generic function decodeGeneric. This is due to the fact that JVC does not send any header if there is a repeat. If the first time calling decodeGeneric failed, it will check whether it is because of the lack of header and use different parameters when calling decodeGeneric for the second time. If successfully called it the second time, it means that the signal is indeed a repeat code and the most significant bit need to be added before returning the value.

When transmitting with JVC protocol, it is necessary to omit the header when repeat sending.

```
void IRsendJVC::send(unsigned long data, bool First)
{
  sendGeneric(data, 16,525*16*First, 525*8*First, 525, 525,525*3, 525, 38, true);
  space(525*45);
  if(First) sendGeneric(data, 16,0,0, 525, 525,525*3, 525, 38, true);
}
```

Figure 5.10 Sending for JVC

As shown in Figure 5.10, if it is not first time sending, it means that it is a repeat frame and the mark and space header will be set to 0 by multiplying it with 0. This function called sendGeneric twice if it is not a repeat frame because some older JVC device requires at least two frames.

5.2.5 RC5 and RC6

The RC5 and RC6 protocols are developed by Phillips but also used by various other manufacturers. These two protocols use a special toggle bit to let you know whether the signal is sent by holding down the button or whether this is an independent key press. This means that if the same button is press and release every time, the decoded value will have one bit inverted (0 or 1) each time. However if the button is hold down, the remote sends repeated codes and the bit will not change. Thus, special handling is required when decoding RC5 and RC6.

```
#define MIN_RC5_SAMPLES 11
#define MIN_RC6_SAMPLES 1

bool IRdecodeRC5::decode(void) {
  IRLIB_ATTEMPT_MESSAGE(F("RC5"));
  if (rawlen < MIN_RC5_SAMPLES + 2) return RAW_COUNT_ERROR;
  offset = 1; // Skip gap space
  data = 0;
  used = 0;
  // Get start bits
  if (getRClevel(&used, RC5_T1) != MARK) return HEADER_MARK_ERROR(RC5_T1);
  for (nbits = 0; offset < rawlen; nbits++) {
    RCLevel levelA = getRClevel(&used, RC5_T1);
    RCLevel levelB = getRClevel(&used, RC5_T1);
    if (levelA == SPACE && levelB == MARK) {
      // 1 bit
      data = (data << 1) | 1;
    }
    else if (levelA == MARK && levelB == SPACE) {
      // zero bit
      data <<= 1;
    }
    else return DATA_MARK_ERROR(RC5_T1);
  }
  // Success
  bits = 13;
  value = data;
  decode_type = RC5;
  return true;
}
```

Figure 5.11 Decoding for RC5

Figure 5.11 shows the code for decoding RC5. In RC5, Manchester coding is used, which means that each bit will have both a mark and space in the output signal. If the bit is a 1, the first half of the bit time is a mark and the second half is a space, and vice versa if the bit is a 0. Therefore, it is easier if the data is broken into time intervals, which is achieved by using the getRClevel function as shown in Figure 5.12.

```
IRdecodeRC::RCLevel IRdecodeRC::getRClevel(unsigned char *used, const unsigned int t1) {
  if (offset >= rawlen) {
    // After end of recorded buffer, assume SPACE.
    return SPACE;
  }
  unsigned int width = rawbuf[offset];
  IRdecodeRC::RCLevel val;
  if ((offset) % 2) val=MARK; else val=SPACE;

  unsigned char avail;
  if (MATCH(width, t1)) {
    avail = 1;
  }
  else if (MATCH(width, 2*t1)) {
    avail = 2;
  }
  else if (MATCH(width, 3*t1)) {
    avail = 3;
  }
  else {
    if((IgnoreHeader) && (offset==1) && (width<t1))
      avail =1;
    else{
      return ERROR;}
  }
  (*used)++;
  if (*used >= avail) {
    *used = 0;
    (offset)++;
  }
  return val;
}
```

Figure 5.12 The function getRClevel

For RC6, the function getRClevel is also used since Manchester coding is also used. However, in RC6, if the bit is a 1, the first half of the bit time is a space and the second half is a mark, and vice versa if the bit is a 0. This is a direct opposite of RC5. Another difference is that RC6 has something called leader pulse and trailer bits in the header. This makes the decoding more complicated but also allow the protocol to be more versatile. The decoding code is shown in Figure 5.13.

```
bool IRdecodeRC6::decode(void) {
  IRLIB_ATTEMPT_MESSAGE(F("RC6"));
  if (rawlen < MIN_RC6_SAMPLES) return RAW_COUNT_ERROR;
  // Initial mark
  if (!IgnoreHeader) {
    if (!MATCH(rawbuf[1], RC6_HDR_MARK)) return HEADER_MARK_ERROR(RC6_HDR_MARK);
  }
  if (!MATCH(rawbuf[2], RC6_HDR_SPACE)) return HEADER_SPACE_ERROR(RC6_HDR_SPACE);
  offset=3;//Skip gap and header
  data = 0;
  used = 0;
  // Get start bit (1)
  if (getRClevel(&used, RC6_T1) != MARK) return DATA_MARK_ERROR(RC6_T1);
  if (getRClevel(&used, RC6_T1) != SPACE) return DATA_SPACE_ERROR(RC6_T1);
  for (nbits = 0; offset < rawlen; nbits++) {
    RCLevel levelA, levelB; // Next two levels
    levelA = getRClevel(&used, RC6_T1);
    if (nbits == 3) {
      // T bit is double wide; make sure second half matches
      if (levelA != getRClevel(&used, RC6_T1)) return TRAILER_BIT_ERROR(RC6_T1);
    }
    levelB = getRClevel(&used, RC6_T1);
    if (nbits == 3) {
      // T bit is double wide; make sure second half matches
      if (levelB != getRClevel(&used, RC6_T1)) return TRAILER_BIT_ERROR(RC6_T1);
    }
    if (levelA == MARK && levelB == SPACE) { // reversed compared to RC5
      // 1 bit
      data = (data << 1) | 1;
    }
    else if (levelA == SPACE && levelB == MARK) {
      // zero bit
      data <<= 1;
    }
    else {
      return DATA_MARK_ERROR(RC6_T1);
    }
  }
  // Success
  bits = nbits;
  value = data;
  decode_type = RC6;
  return true;
}
```

Figure 5.13 Decoding of RC6

For the transmitting of RC5 and RC6, it is roughly the same as shown in Figure 5.14 and Figure 5.15.

```
#define RC5_T1       889
#define RC5_RPT_LENGTH  46000
void IRsendRC5::send(unsigned long data)
{
  enableIROut(36);
  data = data << (32 - 13);
  Extent=0;
  mark(RC5_T1); // First start bit
  for (unsigned char i = 0; i < 13; i++) {
    if (data & TOPBIT) {
      space(RC5_T1); mark(RC5_T1);// 1 is space, then mark
    }
    else {
      mark(RC5_T1);  space(RC5_T1);// 0 is mark, then space
    }
    data <<= 1;
  }
  space(114000-Extent); // Turn off at end
}
```

Figure 5.14 Transmitting of RC5

```
#define RC6_HDR_MARK    2666
#define RC6_HDR_SPACE   889
#define RC6_T1      444
void IRsendRC6::send(unsigned long data, unsigned char nbits)
{
  enableIROut(36);
  data = data << (32 - nbits);
  Extent=0;
  mark(RC6_HDR_MARK); space(RC6_HDR_SPACE);
  mark(RC6_T1);  space(RC6_T1);// start bit "1"
  int t;
  for (int i = 0; i < nbits; i++) {
    if (i == 3) {
      t = 2 * RC6_T1;       // double-wide trailer bit
    }
    else {
      t = RC6_T1;
    }
    if (data & TOPBIT) {
      mark(t); space(t);//"1" is a Mark/space
    }
    else {
      space(t); mark(t);//"0" is a space/Mark
    }
    data <<= 1;
  }
  space(107000-Extent); // Turn off at end
}
```

Figure 5.15 Transmitting of RC6

A space/mark pair indicates 1 and a mark/space indicates 0 for RC5. This is reversed for RC6.

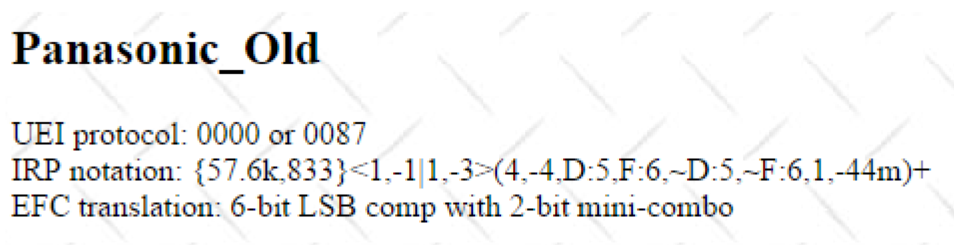
5.3 Unsupported Protocols

5.3.1 Introduction

As mentioned earlier, the library IRLib used in developing the smart home system in this project supports majority of the IR devices out there. However, new appliances and devices are introduced from time to time and it’s inevitable to have some new protocols that IRLib do not support. Therefore, research is done in this project to study, analyze and add new protocol into IRLib. To add a new protocol into IRLib, the first thing to do is to understand how the IR Protocol works. While one can collect the raw data and analyze them to find out the pattern and sequence to understand the protocol, the more practical way would be to lookup the specification of that IR Protocol.

5.3.2 Add new protocols

Most of the times the developer of that protocol will have documents describing the protocol, allowing easier understanding and can write the code straight away. A good source of information about IR protocols can be found on hifi-remote website (Hifi-remote.com, n.d.). An example of this is the description of the old Panasonic protocol as shown in Figure 5.16.



Panasonic_Old

UEI protocol: 0000 or 0087
IRP notation: {57.6k,833}<1,-1|1,-3>(4,-4,D:5,F:6,~D:5,~F:6,1,-44m)+
EFC translation: 6-bit LSB comp with 2-bit mini-combo

Figure 5.16 Description of Panasonic_Old protocol

The easiest way to understand an IR protocol is by reading its IR Protocol notation (IRP notation). For the old Panasonic protocol, the part {57.6k, 833} means that the frequency is 57.6 kHz and the base length for the pulse is 833. The next part <1,-1|1,3> describes what “0” and “1” is. The first part means that “0” is 833*1 marks, and 833*1 spaces. The second part means that “1” is 833 marks followed by 833*3=2499 spaces. The parentheses parts (4,-

4,D:5,F:6,~D:5,~F:6,1,-44m)+ means what data gets sent. For this part, it means that the data starts with header consisting of 4*833 marks and 4*833 spaces. It is then followed by 5 device bits (D), 6 function bits (F), 5 complemented device bits (~D) and 6 complemented function bits (~F). These are the bits that are changed into values when decoding. It then concludes with a single “1” bit followed by 44 blank spaces. The “+” at the end means that it is only necessary to transmit this sequence once and it can repeat as many times as required. In total, the entire protocol is 5+6+5+6=22 bits long since stop bit and blank space is not encoded.

After the information is extracted, the next thing to do is to check if the generic function decodeGeneric can be used in this case. Since this protocol does not need special handling such as half interval like RC5 and RC6 did, decodeGeneric can be used. Figure 5.17 shows the parameters for the function.

```
bool IRdecodeBase::decodeGeneric(unsigned char Raw_Count, unsigned int Head_Mark,
                                 unsigned int Head_Space, unsigned int Mark_One,
                                 unsigned int Mark_Zero, unsigned int Space_One,
                                 unsigned int Space_Zero) {
```

Figure 5.17 Parameters of decodeGeneric

The first parameter Raw_Count means the length of the raw data, which should be 23*2 for the data bits (*2 because for each bit there is mark and space) and 2 for the header (mark and space). In total, Raw_Count should be 23*2+2=48.

For the second and third parameters, the value of the header mark and space is 4*833 and 4*833 respectively as mentioned before.

“0” → 833 Marks

→ 833 Spaces

“1” → 833 Marks

→ 833*3 Spaces

Figure 5.18 Mark and space for one and zero

From the IRP notation <1, -1|1,3> earlier, Figure 5.18 is drawn. The parameter Mark_One should be 0 if the length of the space varies. In this case, it is obvious that Marks for both “0” and “1” remains the same whereas the Spaces for “0” and “1” varies. Therefore, Mark_One should be 0 in this case. From Figure 5.18, the other three parameters, Mark_One, Space_Zero, and Space_One are 833, 833*3, and 833 respectively.

Therefore, when calling the generic function, the parameters should be written as shown in Figure 5.19.

```
bool IRdecodePanasonic_Old::decode(void) {
  IRLIB_ATTEMPT_MESSAGE(F("Panasonic_Old"));
  if(!decodeGeneric(48,833*4,833*4,0,833,833*3,833))
      return false;
  decode_type = PANASONIC_OLD;
  return true;
}
```

Figure 5.19 Decoding for Panasonic_Old IR Protocol

To transmit using the added protocol, just reuse the sendGeneric function with the parameters found from the IRP notation as shown in Figure 5.20.

```
void IRsendPanasonic_Old::send(unsigned long data)
{
  sendGeneric(data,22, 833*4, 833*4, 833, 833, 833*3, 833,57, true);
};
```

Figure 5.20 Transmitting for Panasonic_Old IR Protocol

For the parameters, the data is the value that will be transmitted. The value 22 is the bit length of the protocol without stop bit, which was calculated earlier from the IRP notation. For the Mark_One parameter, it is not necessary to replace the value 833 with 0 to indicate the space varies as this is a transmitting function. The parameter 57 means that the modulated frequency should be 57 kHz. The Boolean true represents whether stop bit is included. If stop bit is included, there is no need to add the parameter Max_Extent to limit the maximum length of the protocol. In this case, since there is a single ‘1’ bit as stop bit, the Boolean should be true.

CHAPTER 6: TRANSCEIVER MODULES

6.1 Introduction

The smart home system in this project used master and slaves architecture, which means there are multiple arduinos connected to each other. To achieve this wirelessly, 2.4GHz radio transceivers nRF24L01 are used.

The nRF24L01 is a low-cost transceiver module that allows communication between arduinos. It allows data to be sent and received in packets of several bytes at a time. This transceiver has a lot of internal complexity but there are a few arduino libraries available to make using it easier. In this project, the library RF24 by maniacbug is used.

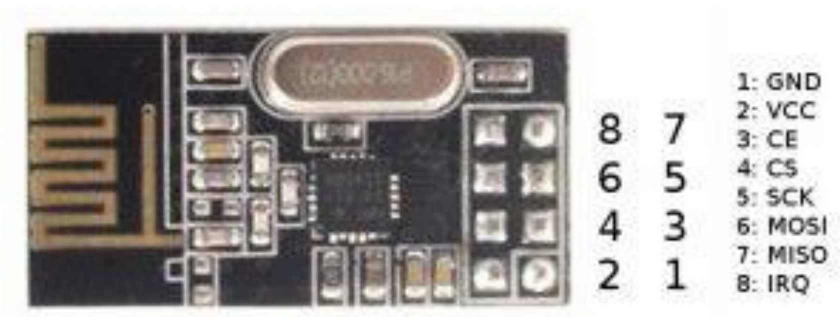


Figure 6.1 nRF24L01 module

The nRF24L01 transceiver should be powered by 3.3V, which is provided from the arduino 3.3V port in this project. The transceiver range is dependent on the surroundings and obstacles, where the range is higher with clear line of sight. Without any additional antenna to power it (which is the case in this project), the transceiver can reached around 100 meters. The range can be further improved in several ways, such as using separate power source other than arduino port, or lowering the data rate to just fast enough.

6.2 Implementation of nRF24L01

6.2.1 Pipes

In master unit, the nRF24L01 transceiver module is called when IR code should be transmitted by a slave module. The master unit is assigned the number ‘0’, whereas slave units are numbered from ‘1’ to ‘9’. If the unit integer in the command received is ‘1’, the master unit will not transmit the IR signal. Instead, it will send the command to slave unit labeled ‘1’. To do this, the first thing to be done is to open a reading pipe and a writing pipe on both the master unit and slave unit ‘1’.

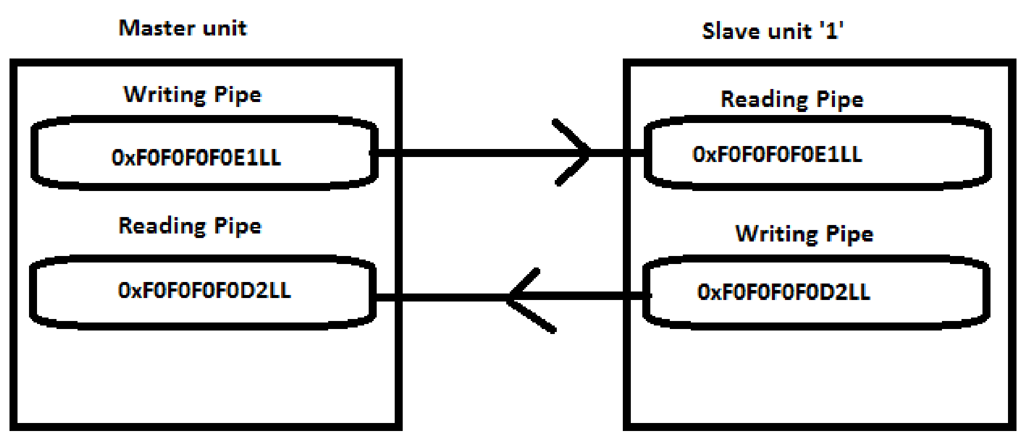


Figure 6.2 Illustration of ‘pipes’

The nRF24L01 uses “pipes” that connect from transmitter to receiver, and each pipe needs an address. The writing pipe of the transmitter must have the same address as the reading pipe of the receiver, which is shown in Figure 6.2. Two pipes are created on each unit to allow both way communications. In the codes, to determine what address to be used for the pipes, an array of address is created as shown in Figure 6.3.

```
const uint64_t pipes[2] = { 0xF0F0F0F0E1LL, 0xF0F0F0F0D2LL, 0xF0F0F0F0D3LL, 0xF0F0F0F0D4LL,
                            0xF0F0F0F0D5LL, 0xF0F0F0F0D6LL, 0xF0F0F0F0D7LL, 0xF0F0F0F0D8LL};
```

Figure 6.3 Array of pipes addresses

As shown in Figure 6.3, there are a total of 8 addresses, which means a maximum of 4 slave units can be controlled. This can be increased as many as desired since in the codes only 0 or 1 reading pipe is listening at all times regardless of the number of slave units (maximum reading pipe at the same time is 6). The code chooses which address from the array to be used based on the unit number and the calculations as shown in figure 6.4.

```
int i =unit*2-2, j=unit*2-1;

radio.openWritingPipe(pipes[i]);
radio.openReadingPipe(1,pipes[j]);
```

Figure 6.4 Getting the address for pipes

As shown in figure 6.4, if the slave unit to be connected is ‘1’, then the 0th and 1st elements of the address array will be used; Whereas if the slave unit to be connected is ‘2’, then the 2nd and 3rd elements of the address array will be used etc.

Once the pipes are built on both ends, the communication can begin.

6.2.2 Three-Way Handshakes

After the pipes are built, a connection is first established using three-way handshakes before actual data is transmitted as shown in figure 6.5.

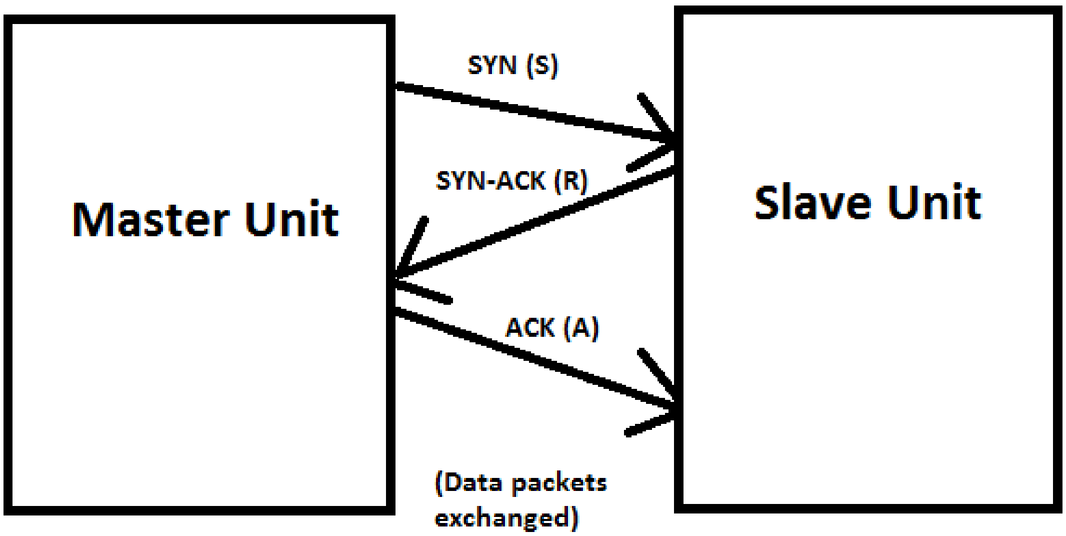


Figure 6.5 Three Way Handshakes between master unit and slave unit

Firstly, the master unit will sent a character ‘S’ to represent SYN packet to the slave unit, and start listening for a reply. If the slave unit received the ‘S’, it will stop listening and sent a character ‘R’ to represent SYN-ACK packet to the master unit. After sending the character, the slave unit will start listening for reply. If the master unit received the character ‘R’, it will stop listening and send a character ‘A’ to represent ACK packet to the slave unit.

It will then start transmitting the data (IR command) to the slave unit. If the slave unit received the character ‘A’, it will decide that a connection has been established and any data received from thereon will be received as the message, until the character ‘*’ is received. If the character ‘*’ is received, it will disconnect the connection and use the message received to transmit IR through its’

IR transmitter. This means that another three-way handshake is required to send another message.

Throughout the three-way handshakes, if no replies are received within 10 seconds, both the master unit and slave units will halt the handshakes. Master unit will retry the handshakes from the start again. If there is no timeout and the connection is established, the master unit will inform the user that the command has been sent successfully.

CHAPTER 7: CONCLUSION

In summary, this project aims to solve the problem of the high cost of current smart home systems which costs are too high to be affordable. It also intends to solve the communication problem between users and the smart home system especially in rural area or developing countries such as Malaysia where internet connection stability is still a problem. To achieve these objectives, a smart home system that is low cost and uses IR and GSM technology is built. The design and implementation of the system were discussed in this project, along with tools used and the methodology adapted in this project.

While designing and implementing this project, several challenges and obstacles are faced. It was hard to find and choose hardware parts to assemble the unit, as there is no precedential project to refer to. Several plans to use certain design or hardware part were discarded in the process due to hardware difficulties such as limited pin number on Arduino. It was also hard to write a complex code with the Arduino IDE due to the limitation, thus Eclipse Arduino IDE is used instead in the middle of implementation. The whole process of designing and implementing this project went through several changes and modifications before the final design is set and implemented.

By inventing a smart home system that uses infrared and GSM technology that is cheap and stable, appliances manufacturers may be convinced to build more smart appliances using infrared which are way cheaper in cost. This could lead to a future where smart home systems are common and affordable to almost anyone.This project can also be further improved in the future by adding more features such as security system.

BIBLIOGRAPHY

Cleverism. (2014). *The Smart Home | Intelligent Home Automation*. Available at: http://www.cleverism.com/smart-home-intelligent-home-automation/ [Accessed 22Feb. 2016].

Consumerreports.org. (2014).Best Home Automation System - Consumer Reports.Available at: http://www.consumerreports.org/cro/magazine/2014/06/run-your-home-from-your-phone/index.htm [Accessed 22Feb. 2016].

SearchMobileComputing. (2005). What is IR wireless (infrared wireless)? - Definition from WhatIs.com. Available at: http://searchmobilecomputing.techtarget.com/definition/IR-wireless [Accessed 19 Aug. 2015].

Spicer, D. (2000). If You Can't Stand the Coding, Stay Out of the Kitchen: Three Chapters in the History of Home Automation.Dr. Dobb's. Available at: http://www.drdobbs.com/architecture-and-design/if-you-cant-stand-the-coding-stay-out-of/184404040 [Accessed 15Aug. 2015].

Whirlpool. (2016). Discover Kitchen, Laundry, and Home Appliances from Whirlpool. Available at: http://www.whirlpool.com/ [Accessed 27Feb. 2016].

Fitzsimmons, C. (2016). Appliances getting smarter but less durable.The Sydney Morning Herald. Available at: http://www.smh.com.au/money/saving/why-the-internet-of-things-will-mean-junking-expensive-appliances-more-often-20160222-gn0q3o.html [Accessed 27Feb. 2016].

Iris. (2015). Iris by Lowe's Simplifies Smart Home Management. Available at: https://www.irisbylowes.com/ [Accessed 13Aug. 2015].

Panasonic.com. (2016).Smart Home - Giving you peace of mind - Panasonic UK & Ireland. Available at: https://www.panasonic.com/uk/consumer/smart-home-learn/smart-home/giving-you-peace-of-mind.html [Accessed 22 Mar. 2016].

Kumar, S. (2014).Ubiquitous Smart Home System Using Android Application.IJCNC, 6(1), pp.33-43.

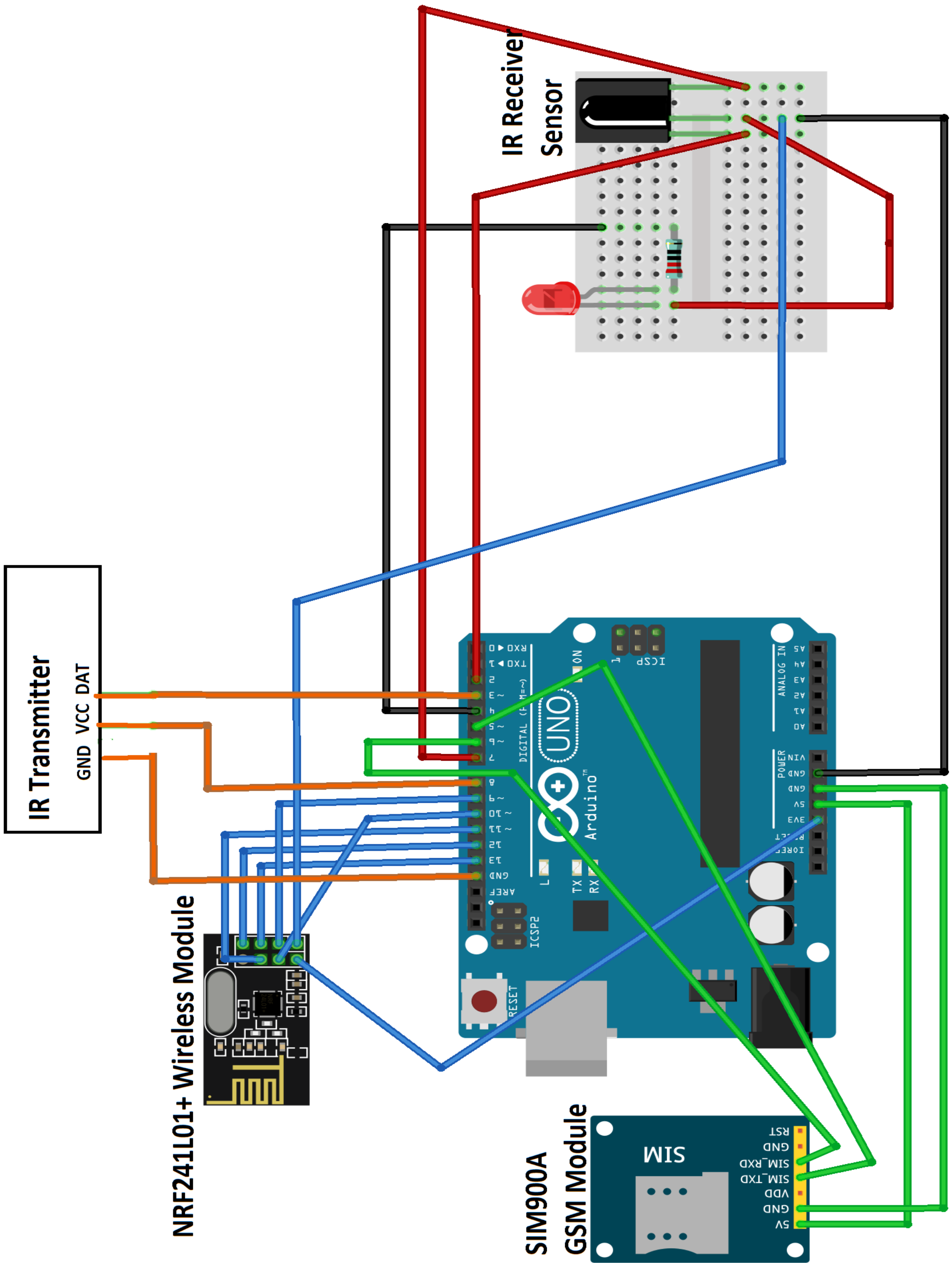
MehekPotnis et al. 'Home Security System Using Gsm Modem' in *Int. Journal of Engineering Research and Applications* ISSN : 2248-9622, Vol. 5, Issue 4, (Part -6) April 2015, pp.143-147

I, Andrew (2014).GSMRC - GSM Infrared Remote Control. Gsmrc.blogspot.com. Available at: http://Gsmrc.blogspot.com [Accessed 13 Aug. 2015].

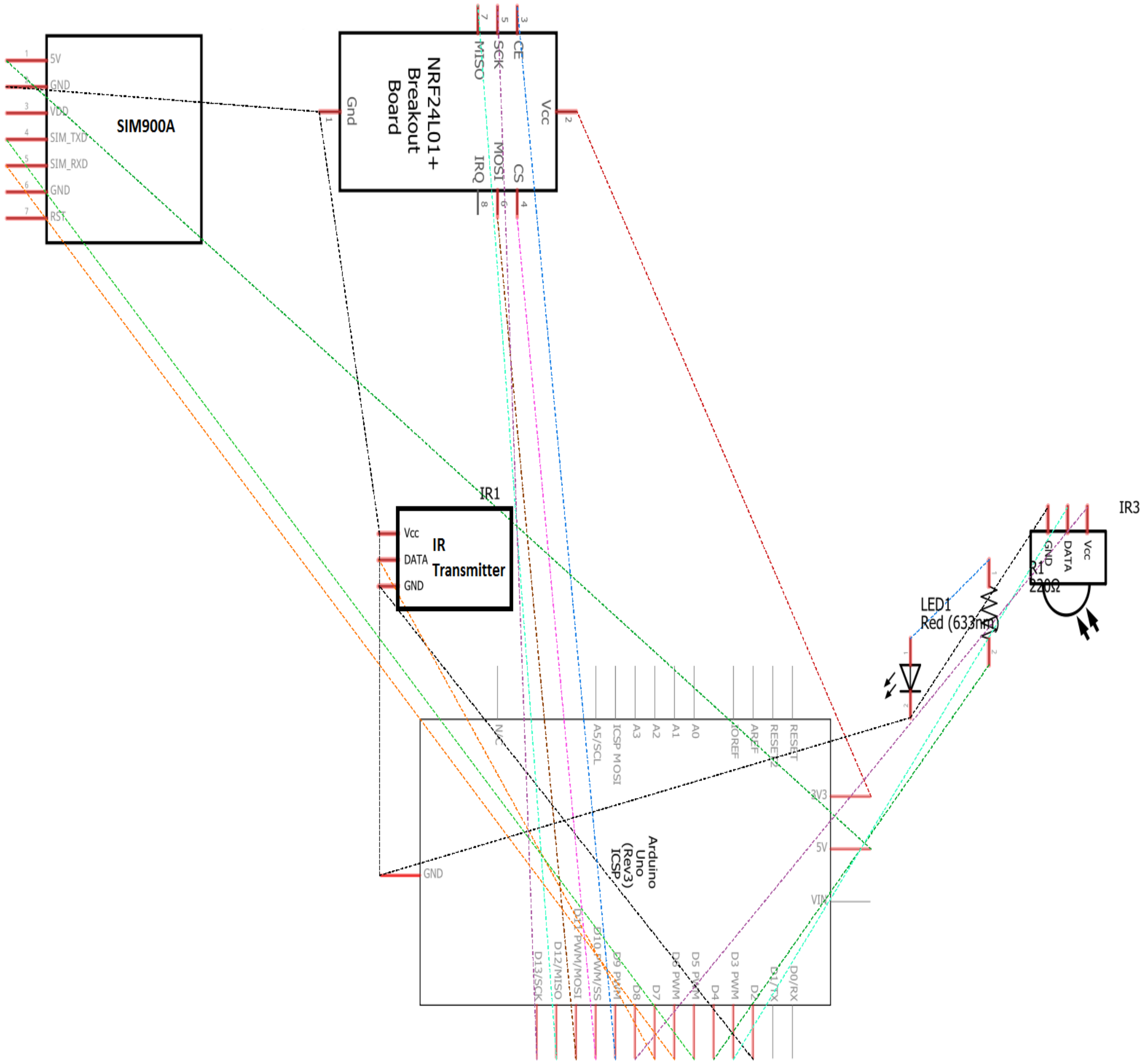
Hifi-remote.com. (n.d.). *Interpreting Decoded IR Signals (v2.45)*. Available at: http://www.hifi-remote.com/johnsfine/DecodeIR.html [Accessed 6 July. 2016].

APPENDIX A

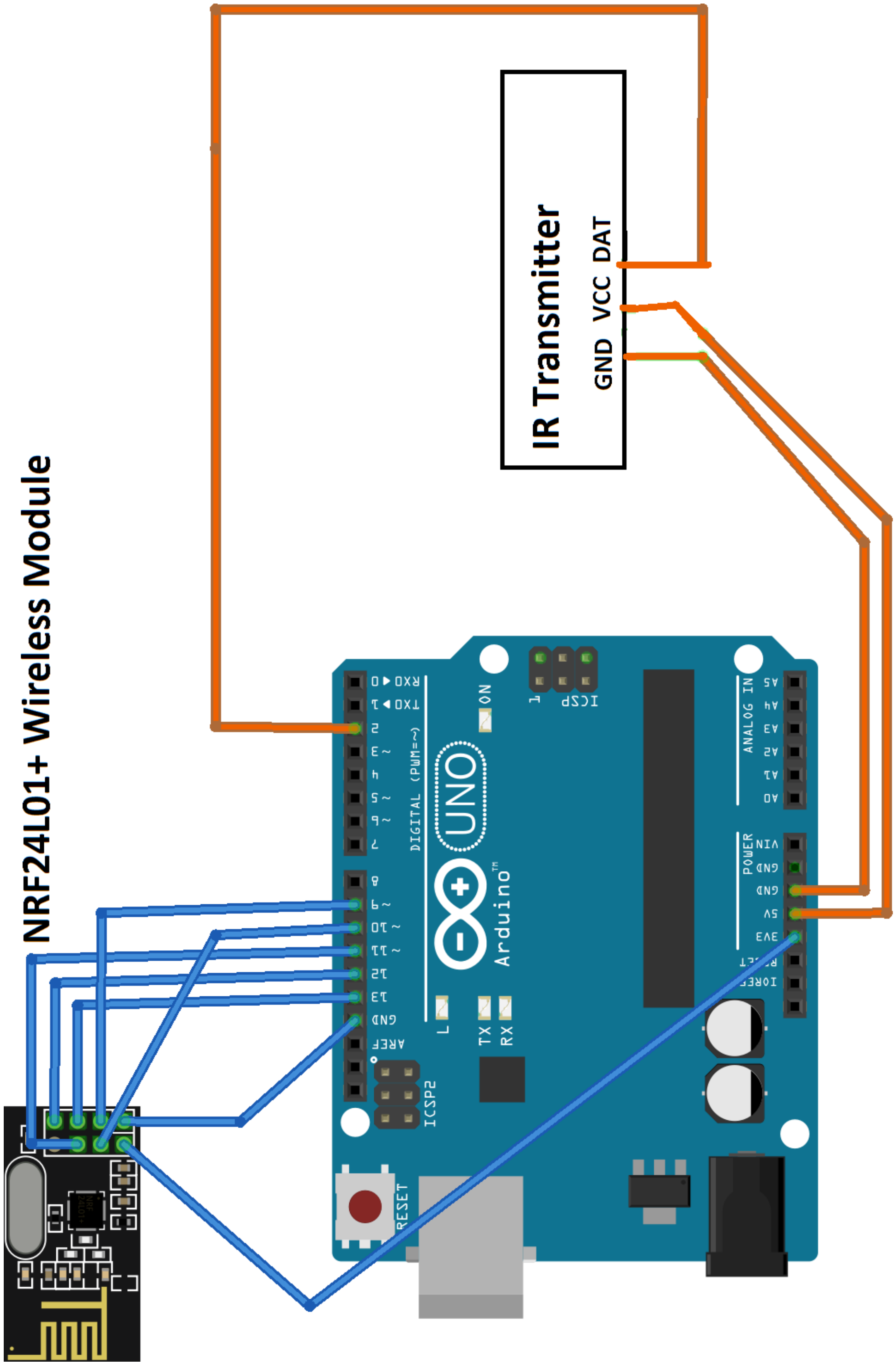
A-1 Master Unit Circuit Diagram



A-2 Master Unit Schematics



A-3 Slave Unit Circuit Diagram



A-4 Slave Unit Schematics