FAN SPEED CONTROLLED USING SMARTPHONE

A final project report
presented to
the Faculty of Engineering

By

Yasin Noer Huda P
002201200006

in partial fulfillment
of the requirements of the degree
Bachelor of Science in Electrical Engineering

President University
March 2016
DECLARATION OF ORIGINALITY

I declare that this final project report, entitled “Fan Speed Controlled Using Smartphone” is my own original piece of work and, to the best of my knowledge and belief, has not been submitted, either in whole or in part, to another university to obtain a degree. All sources that are quoted or referred to are truly declared.

Cikarang, Indonesia, 18 March 2016

Yasin Noer Huda P.
APPROVAL PAGE

FAN SPEED CONTROLLED USING SMARTPHONE

By
Yasin Noer Huda P
002201200006

Approved by,

Arthur Silitonga, M.Sc
Final Project Supervisor

Antonius Suhartomo, Ph.D.
Head of Study Program of Electrical Engineering

Dr.-Ing Erwin Sitompul
Dean of Faculty of Engineering
ACKNOWLEDGEMENT

In the name of Allah the Most Gracious, the Most Merciful. Alhamdulillah because His love and blessings I could finish this thesis for my bachelor degree in Electrical Engineering.

I would like to thank my final project supervisor Arthur Silitonga, M.Sc who gave me supports, during my thesis work and some lessons. Indeed, I will always remember later the lesson about life that you already shared during thesis discussion to be a good man.

Thank you so much to my parents, Bapak Nurhadi and Ibu Iswati who have given me support, love and sincerity. Furthermore, they let me to learn in taking a hard decision in complex situation in my life. My brothers and sisters Rani Aurnur Rahima, Assyafa Rizki, Madina, and Alfath, thank you for always being my reason to be a best brother. Every hope that they send me is a motivation to do a better thing in this life.

I thank you also for my President University especially for all Engineering batch 2012 for our craziness, bond and love in this short-university life, And Fikri Aswan B.Sc and Novaldy Khalifatama B.Sc for their guidance and help during college.

The last, thank you so much for all friend from my childhood who entertained my bored times in completing this thesis.
APPROVAL FOR SCIENTIFIC PUBLICATION

I hereby, for the purpose of development of science and technology, certify and approve to give President University a non-exclusive royalty-free right upon my final project report with the title:

FAN SPEED CONTROLLED USING SMARTPHONE

Along with the related software or hardware prototype (if needed). With this non-exclusive royalty-free right, President University is entitled to conserve, to convert, to manage in a database, to maintain, and to publish my final project report. These are to be done with the obligation from President University to mention my name as the copyright owner of my final project report.

Cikarang, 18 March 2016

Yasin Noer Huda P.
002201200006
ABSTRACT

Nowadays, the technology seems regularly used to substitute human’s job. Electronic devices are one of the advanced technologies that can substitute the job. The devices are varies such as security device, automatic toll gate card and getting information using a smartphone are the advancement in the electronic technology that make human’s life easier. However, not all of the electronic devices are automatically works, for example a fan. From this point of view, the idea to develop and implement this final project comes up. The main objective is to build a controller which will turn to increase and decrease speed of fan using a smartphone over Bluetooth to ease the fan usage. An automatic controlled fan will help some people who used to increase or decrease speed of fan manually to be easier by using a smartphone. The prototype can control two speeds of fan which are speed one, speed two, and off state.

Keywords: Bluetooth, Smartphone, Fan, Automatic-Controlled-Fan
TABLE OF CONTENT

DECLARATION OF ORIGINALITY ................................................................. ii
APPROVAL PAGE ............................................................................................... v
ACKNOWLEDGEMENT ...................................................................................... vi
APPROVAL FOR SCIENTIFIC PUBLICATION ................................................. vii
ABSTRACT .......................................................................................................... viii
TABLE OF CONTENT ....................................................................................... ix
CHAPTER 1 ......................................................................................................... 1
INTRODUCTION ................................................................................................. 1
  1.1. Final Project Background ...................................................................... 1
  1.2. Problem Statement .............................................................................. 1
  1.3. Objective .............................................................................................. 2
  1.4. Scopes and Limitations ...................................................................... 2
  1.5. Final Project Methodology ................................................................. 2
  1.6. Final Project Outline .......................................................................... 3
CHAPTER 2 ......................................................................................................... 4
DESIGN SPECIFICATION ............................................................................... 4
  2.1. Arduino UNO ....................................................................................... 4
  2.1.1. The advantage of Arduino UNO ...................................................... 4
  2.2.2. Arduino Uno Pin Configuration ...................................................... 5
  2.3. Relay Module ...................................................................................... 6
  2.4. Bluetooth Module HC - 05 ................................................................. 7
  2.5. Fan .................................................................................................... 9
  2.6. AC/DC Adaptor ................................................................................. 10
CHAPTER 3 ....................................................................................................... 12
DESIGN DEVELOPMENT AND IMPLEMENTATION .................................. 12
  3.1. Requirement ....................................................................................... 12
LIST OF TABLE

Table 2.1 Table Hardware Specification ......................................................... 8
Table 5.1 Final Project Achievements ............................................................. 29
LIST OF FIGURES

Figure 2.1 Arduino uno ................................................................. 4
Figure 2.2 Arduino uno pin configuration .............................. 6
Figure 2.3 Relay schematic ....................................................... 7
Figure 2.4 Relay module ............................................................ 7
Figure 2.5 Bluetooth module HC – 05 schematic ...................... 8
Figure 2.6 Miyako KAD927B ................................................. 10
Figure 2.7 AC/DC adaptor ......................................................... 10
Figure 3.1 V-Model of the Design .......................................... 13
Figure 3.2 Block diagram of device ........................................ 14
Figure 3.3 Flow chart of device ............................................... 16
Figure 3.4 Arduino uno circuit ............................................... 17
Figure 3.5 Bluetooth module HC - 05 circuit ......................... 18
Figure 3.6 Solenoid circuit ....................................................... 19
Figure 3.7 RemoteXY .............................................................. 20
Figure 3.8 Arduino IDE program ........................................... 20
Figure 3.9 RemoteXY interface ............................................. 21
Figure 3.10 Code from RemoteXY ........................................ 21
Figure 3.11 Arduino software idle interface ......................... 22
Figure 3.12 Block diagram of program ................................. 23
Figure 4.1 Final project device .............................................. 25
Figure 4.2 Circuit of final project .......................................... 26
Figure 4.3 Control interface on smartphone ......................... 26
Figure 4.4 Fan with speed one ............................................ 27
Figure 4.5 Fan with speed two ............................................ 27
CHAPTER 1
INTRODUCTION

1.1. Final Project Background

Nowadays the weather temperature becomes hotter than before. It causes some people consider to purchase electronic equipment that can reduce the hot temperature in the room. For example, air conditioner is installed in the room to make a people live more comfortable with the temperature. Fortunately, air conditioner is already provided by a remote control so people can easily control the temperature from the long distance. On the other hand, some people do not have capability to purchase an air conditioner because the price is expensive. Besides, the consumption of electricity is high. Hence, some people prefer to choose the fan but the problem is that most of the fan that sell in Indonesia comes out in manual on accelerating or decreasing speed fan.

In contrast, Indonesian people use the smartphone almost regularly. Based on tekno.kompas.com, it is said that Indonesian people mostly used their smartphone more than 3 hours a day. It indicates that there is opportunity to develop a product to control a fan using a smartphone. But the problem is there are so many operation system that each of system has different configuration. Stat counter global states reported that the largest of operating system that used by Indonesian people with the total market share is 59.91 %. By looking this number, it concludes this project will be focus on android operation system.

1.2. Problem Statement

Reflecting from the final project background, the purpose of this project is then to answer this following question:

- How to develop a fan speed controlled using a smartphone.
1.3. Objective

- The main objective of this project is to develop a fan speed controlled using a smartphone.
- Help the sick people to be able to turn on the fan without getting up

1.4. Scopes and Limitations

Due to limited resources, there are only some scopes and limitation that would be covered in this project.

- The programming language that will used and developed by using software Arduino IDE.
- The range of Bluetooth module is 10 meters.
- The lifetime of Arduino Uno is 1 years.
- This project does not count the RPM of the fan (Rotation per Minute).
- This project can automatically turn on to speed one, speed two or off state.

1.5. Final Project Methodology

There is a methodology to compile this final project. This process are done in a consecutive order; project initiation, planning, implementation, and evaluation.

- Initiation and planning
  This is basically the preliminary step is to define the topic, background, objective, and scope of the final. In order to get much information and related to idea, the research is one of the action.

- Working principle study
  In this step, the basic concept is analyze to be applied in final project. It includes the study of arduino, relay module, and source code, and the bluetooth module.

- Execution
  This is where the software and hardware developed. Moreover, all of the gathered information from previous section is implemented here.

- Evaluation
  This is the last stage in this final project that include circuit testing and
1.6. Final Project Outline
The final project report consists of specifically into the following chapters:

Chapter I: Introduction
This chapter consist of Final Project background, Problem statement, Objective, Scope and Limitation, and Final Project outline.

Chapter II: Design Specification
This chapter describes about the components used and discusses their functions.

Chapter III: Design development and implementation
This chapter contains the specification of all the supporting principles and theories. Besides that, it covers the model and the detailed description of the utilized method. Complete software and hardware configuration are also included in this chapter.

Chapter IV: Result and Discussions
This chapter consists of the analysis of the hardware and software of the developed project including the result and discussions.

Chapter V: Conclusions and Recommendations
This chapter consists of conclusions obtained throughout this project and recommendations for future projects.
CHAPTER 2
DESIGN SPECIFICATION

2.1. Arduino UNO

A microcontroller is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals. Microcontrollers are used in automatically controlled products and devices, such as automobile engine control systems, implantable medical devices, remote controls, office machines, etc. In this project, the microcontroller that will use is Arduino UNO shown in Figure 2.1.

![Arduino UNO](image)

Figure 2.1 Arduino uno [1,2]

The Arduino Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started.

2.1.1. The advantage of Arduino UNO

- Arduino does not need any voltage regulator and capacitor because the entire component has already inside the arduino.
- The price of arduino is affordable.
• There are so many people that work in the similar environment so it will be many information about arduino.
• There a lot of libraries in this device so it could be easier to interface the hardware.
• The Arduino IDE does not need any payment in implementing the code

2.2.2. Arduino Uno Pin Configuration
There so many pin configuration in Arduino uno, these pin configuration can be divided into many parts. There are digital, analog, communication, PWM (Pulse Width Modulator), Vcc, and Ground.

a) Power Pin :
• 3.3V and 5V
There are 2 output voltage port that can be out from the Arduino Uno by connecting to one of port, it could produce 5 V or 3.3 V
• Ground
Supply Negative Voltage.
• Vin
There are input voltage port that can become the source of the Arduino Uno using cable, the range of the voltage is between 7V until 12V.

b) Analog In Pin
Analog In pin on Arduino Uno is from A0 until A5 which has DC current 40mA per port. Each of which also provide 10 bits of resolution.

c) Digital Pin
Digital pin on Arduino Uno is from 0 until 13. Pin digital only gives the output of 0 or 1.
d) Communication Pin

Communication pin is displayed as TX and RX. The communication pin is in digital port, 0 for RX and 1 for TX. RX used to receive and TX is to transmit TTL serial data, the connection between RX and TX is very crucial or the circuit would not working properly.

e) PWM Pin

The PWN pin is 3, 5, 6, 9, 10, and 11 in digital pin. PWM provide 8-bit output with the analog Write().

f) LED

There is a built-in LED connected to digital pin 13. When the value of this pin HIGH then the LED will be turn on and when the value of the pin is LOW, the LED will be turn off.

![Figure 2.2 Arduino uno pin configuration](image)

Figure 2.2 Arduino uno pin configuration [1,2]

2.3. Relay Module

A relay is an electrical switch that uses an electromagnet to move the switch from the off to on position automatically. There are a lot of relay types such as Single Pole Single Throw (SPST), as can be seen in Figure 2.3 Single Pole
Double Throw (SPDT) has three contact that labelled Normally Open (NO), Normally Close (NC), Common (CO). Relay module uses a schematic of Single Pole Double Throw. The normally close usually will be connected to the common when there is no power is through the coil. The normally open will be open when there is no power through the coil. The Relay module has maximum load which 250VAC with 10 A or 10VDC with 10A, See Figure 2.4

![Relay pin out](image)

**Figure 2.3 Relay schematic [3]**

![Relay module](image)

**Figure 2.4 Relay module [3]**

### 2.4. Bluetooth Module HC - 05

Bluetooth module HC – 05 is a device that easy to use Bluetooth SPP (Serial Port Protocol) Module, it is designed for wireless serial connection setup. Serial port Bluetooth is equipped with Bluetooth V2.0 + EDR (Enhanced Data Rate) 3Mbps modulation with supported frequencies 2.4 GHz radio transceiver and baseband.

The features on Bluetooth module HC – 05:

- Typical -80dBm sensitivity
- Up to +4dBm RF transmit power
- Low Power Operation with 1.8 V to 5 V
- With integrated antenna
- UART interface with programmable baud rate
- With edge connector

![Bluetooth module HC–05 schematic](image)

**Figure 2.5 Bluetooth module HC–05 schematic [4]**

**Table 2.1 Table Hardware Specification [4]**

<table>
<thead>
<tr>
<th>PIN Name</th>
<th>PIN #</th>
<th>PAD Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GND</td>
<td>13,21,22</td>
<td>VSS</td>
<td>Ground Pot</td>
</tr>
<tr>
<td>3.3 VCC</td>
<td>12</td>
<td>3.3V</td>
<td>Integrated 3.3 (+) supply with on-chip linear regulator output within 3.15 – 3.3 V</td>
</tr>
<tr>
<td>AIO0</td>
<td>9</td>
<td>Bi-directional</td>
<td>Programmable input or output line</td>
</tr>
<tr>
<td>AIO1</td>
<td>10</td>
<td>Bi-directional</td>
<td>Programmable input or output line</td>
</tr>
<tr>
<td>PIO0</td>
<td>23</td>
<td>Bi-directional</td>
<td>Programmable input or output line, control output for LNA</td>
</tr>
<tr>
<td>PIO1</td>
<td>24</td>
<td>Bi-directional</td>
<td>Programmable input or output line, control output for PA</td>
</tr>
<tr>
<td>PIO2</td>
<td>25</td>
<td>Bi-directional</td>
<td>Programmable input or output line</td>
</tr>
<tr>
<td>PIO3</td>
<td>26</td>
<td>Bi-directional</td>
<td>Programmable input or output line</td>
</tr>
<tr>
<td>PIO4</td>
<td>27</td>
<td>Bi-directional</td>
<td>Programmable input or output line</td>
</tr>
<tr>
<td>PIO5</td>
<td>28</td>
<td>Bi-directional</td>
<td>Programmable input or output line</td>
</tr>
<tr>
<td>PIO6</td>
<td>29</td>
<td>Bi-directional</td>
<td>Programmable input or output line</td>
</tr>
<tr>
<td>PIO7</td>
<td>30</td>
<td>Bi-directional</td>
<td>Programmable input or output line</td>
</tr>
<tr>
<td>PIO8</td>
<td>31</td>
<td>Bi-directional</td>
<td>Programmable input or output line</td>
</tr>
</tbody>
</table>
### 2.5. Fan

Fan is a device that create a current of air by movement of a surface using a rotating motor. Miyako KAD927B is the fan that used in this final project. The power source that needed for the fan is 220 V\(_{\text{AC}}\) with utility frequencies 50 Hz. The fan has 2 speeds which speed 1 requires 30 V\(_{\text{AC}}\) and speed 2 requires 35 V\(_{\text{AC}}\). See in Figure 2.6.

<table>
<thead>
<tr>
<th>Name</th>
<th>Number</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIO9</td>
<td>32</td>
<td>Bi-directional</td>
<td>Programmable input or output line</td>
</tr>
<tr>
<td>PIO10</td>
<td>33</td>
<td>Bi-directional</td>
<td>Programmable input or output line</td>
</tr>
<tr>
<td>PIO11</td>
<td>34</td>
<td>Bi-directional</td>
<td>Programmable input or output line</td>
</tr>
<tr>
<td>RESETB</td>
<td>11</td>
<td>CMOS input</td>
<td>Reset if low, input debounced so must be low</td>
</tr>
<tr>
<td>UART_RTS</td>
<td>4</td>
<td>CMOS output</td>
<td>UART request to send, active low</td>
</tr>
<tr>
<td>UART_CTS</td>
<td>3</td>
<td>CMOS input</td>
<td>UART clear to send, active low</td>
</tr>
<tr>
<td>UART_RX</td>
<td>2</td>
<td>CMOS input</td>
<td>UART data input</td>
</tr>
<tr>
<td>UART_TX</td>
<td>1</td>
<td>CMOS output</td>
<td>UART data output</td>
</tr>
<tr>
<td>SPI_MOSI</td>
<td>17</td>
<td>CMOS input</td>
<td>Serial peripheral interface data input</td>
</tr>
<tr>
<td>SPI_CSB</td>
<td>16</td>
<td>CMOS input</td>
<td>Chip select for serial peripheral interface, active low</td>
</tr>
<tr>
<td>SPI_CLK</td>
<td>19</td>
<td>CMOS input</td>
<td>Serial peripheral interface clock</td>
</tr>
<tr>
<td>SPI_MISO</td>
<td>18</td>
<td>CMOS input</td>
<td>Serial peripheral interface data output</td>
</tr>
<tr>
<td>USB_-</td>
<td>15</td>
<td>Bi-directional</td>
<td></td>
</tr>
<tr>
<td>USB_+</td>
<td>20</td>
<td>Bi-directional</td>
<td></td>
</tr>
<tr>
<td>NC</td>
<td>14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCM_CLK</td>
<td>5</td>
<td>Bi-directional</td>
<td>Synchronous PCM data clock</td>
</tr>
<tr>
<td>PCM_OUT</td>
<td>6</td>
<td>CMOS output</td>
<td>Synchronous PCM data output</td>
</tr>
<tr>
<td>PCM_IN</td>
<td>7</td>
<td>CMOS input</td>
<td>Synchronous PCM data input</td>
</tr>
<tr>
<td>PCM_SYNC</td>
<td>8</td>
<td>Bi-directional</td>
<td>Synchronous PCM data strobe</td>
</tr>
</tbody>
</table>
2.6. AC/DC Adaptor

Many common electronic device like laptop, smartphone, personal computer and external hard drives use AC/DC adapter in charging or powering. The AC/DC power from electrical outlet (220V) is changed into the type of power (12V) by using AC/DC adapter that contains step down transformer, voltage regulator, capacitor, silica diode and resistor. There are several advantage of using adapter such as heat and electronic interference are kept away from the device that is being powered. So this adaptor will be the external power supply for the arduino uno.

The picture of the AC/DC Adaptor is shown in Figure 2.7.
CHAPTER 3

DESIGN DEVELOPMENT AND IMPLEMENTATION

3.1. Requirement

The objective of this final project is to be able to control the speed of a fan by using smartphone. Sometimes the people tend to increase or decrease the speed of the fan manually. On the other side, the wall fan makes the people hard to change the fan’s speed since it is hanged on the wall. The needs of accessing the technology is very high, so the author made this based on daily needs. This project uses two programs as the basic concept which are Arduino IDE and RemoteXY which application that can control speed fan using Bluetooth in smartphone. Arduino is microcontroller that can be used in controlling the devices such as prototype of the project which is fan. The concept that is used for this project is sending the data by using smartphone with RemoteXY. After that, the data is received by Bluetooth module HC 05. The data that has been received by Bluetooth module HC – 05 will be processed by Arduino uno. Thus, the data is sent as pulse to relay module to make it different between open or close, and the last the speed of fan can be controlled by using smartphone.

To make it easier to understand, the author begins this chapter according to the flows of the V- Model sketched in Figure 3.1
3.2. Specification

The specification of this project is divided into two classes: software design and hardware design. Smartphone sending data to Bluetooth module HC-05 and delivered to Arduino Uno, then from Arduino Uno, the data that given by Bluetooth module HC-05 is sent through to module relay so it select speed of the fan. The author made block diagram of this circuit project is shown in Figure 3.2.
Figure 3.2 Block diagram of device
The component that showed by block diagram are used in this project. Smartphone plays role as the input of the system, whereas the Bluetooth module HC – 05, and relay module as output of the system. The Arduino uno performs as the microcontroller to control relay module and the data from Bluetooth module HC 05.

The application on smartphone that will use in this project is RemoteXY to transmit data. RemoteXY is the software that will process data from smartphones. Through RemoteXY will be delivering and receiving data to Arduino. Whenever the user wanted to change the speed of fan, the RemoteXY will transmit data through Bluetooth module HC – 05 then delivered to arduino uno which will be sent digital data to relay module to take action which speed fan that will use according to Figure 3.3.
Figure 3.3 Flow chart of device
3.3. Hardware Design

This device using an adaptor AC/DC for source power through Arduino uno. This adaptor has the output 7V. Arduino UNO has a minimum power source which is 5V. It is best if the power source is 7V – 12V for the optimal result. Since the Bluetooth module HC – 05 and relay module only need 5V, then the power source is coming from Arduino UNO.

3.3.1 Arduino UNO Board

Arduino UNO is the main component of this project. All hardware can be control directly using Arduino, the amount of software and hardware development that required to do in order to get system running can be simplify by the author. Moreover, Arduino has a lot of libraries to made easier programming of a microcontroller. The author using an adaptor AC/DC to provide a power source to Arduino. The Arduino Uno circuit schematic will be shown in Figure 3.4.
3.3.2 Bluetooth Module HC - 05
Bluetooth module HC – 05 is used to received data from smartphone. There is (VCC, RX, TX, GND) to control received and transmit data to the user. The connection to Arduino UNO are as follows:

- RX to pin 2
- TX to pin 3
- VCC to pin power 5V
- GND to pin power GND

Bluetooth module are only have 4 pins. The interfacing of Bluetooth module HC – 05 to Arduino uno can be seen in Figure 3.5

![Figure 3.5 Bluetooth module HC - 05 circuit](image-url)
3.3.3 Relay module

Relay module is used by arduino to control switch fan. There is (VCC, IN1, IN2, GND) to control NO, COMM, and NC. The connection to arduino uno are as follows:

- IN1 to pin 7
- IN2 to pin 8
- VCC to pin power 5V
- GND to pin power GND

The interconnection are attached on Figure 3.6

![Figure 3.6 Solenoid circuit](image)

3.4. Software design

RemoteXY is the software that will process data from smartphones. Through RemoteXY will be delivering and receiving data to Arduino. After data from RemoteXY sent to Arduino, the Arduino will process the data and deliver it to the relay module depend to the RemoteXY as shown in Figure 3.7. The author uses Arduino software IDE to configure the arduino program and RemoteXY code as shown in Figure 3.8.
1. Program Implementation
The first step is made graphical management interface in RemoteXY. Graphical management interface is used to control fan speed by using Android Smartphone. In this case interfaces need a “select” control in elements sub menu to deliver the information in separating command related speed one and speed two. It can be implemented also to another fan that have more than two speed switch. The step in graphical management interface is drag the select control to the interface, then download the library code by clicking get source as shown in Figure 3.9.
The code from the RemoteXY can be shown in Figure 3.10 Furthermore, the second step is uploading the code that got from RemoteXY to arduino using arduino software idle with usb port cable from laptop. The code in arduino can be shown in Figure 3.11.

```cpp

// RemoteXY select connection mode and Include library 
#define REMOTEXY_USE_SOFTWARESERIAL 
#include <SoftwareSerial.h>

#define REMOTEXY_SERIAL_RX 2
#define REMOTEXY_SERIAL_TX 3
#define REMOTEXY_SERIAL_SPEED 9600

// RemoteXY configure */
unsigned char RemoteXY_CONF[] = 
{ 1,0,9,0,9,5,3,13,23,19 };

// this structure defines all the variables of your control interface */
struct
{
    // input variable */
    unsigned char select_1; /* 0 if select position A, -1 if position B, +1 if position C, ... */

    // other variable */
    unsigned char connect_flag; /* -1 if wire connected, else +0 */

```

Figure 3.10 Code from RemoteXY
After inputting the code in the arduino software idle interface, click the upload button that coded by number 1 to start the uploading process. In other hand, download an application in smartphones from playstore with name RemoteXY. After the code upload is done, the data in arduino will be sent to the smartphone through Bluetooth module HC – 05. After that, the interface that has been designed in RemoteXY will be shown in smartphones. The explanation of this as shown in Figure 3.12.
/* RemoteXY select connection mode and include library */
#define REMOTEXY_MODE__SOFTWARESERIAL
#include <SoftwareSerial.h>
#include <RemoteXY.h>

/* RemoteXY connection settings */
#define REMOTEXY_SERIAL_RX 2
#define REMOTEXY_SERIAL_TX 3
#define REMOTEXY_SERIAL_SPEED 9600

/* RemoteXY configure */
unsigned char RemoteXY_CONF[] =
{ 1,0,50,0,4,0,3,131,15,20
 ,70,26,2,129,0,21,11,9,6,9
 ,79,102,102,0,129,0,39,11,23,6
 ,9,83,112,101,101,100,32,49,0,129
 ,0,65,11,23,6,9,83,112,101,101
 ,100,32,50,0 }
;

/* this structure defines all the variables of your
 control interface */
struct {

   /* input variable */
   unsigned char select_1; /* =0 if select position A, =1
 if position B, =2 if position C, ... */

   /* other variable */
   unsigned char connect_flag; /* =1 if wire connected,
 else =0 */
} RemoteXY;

The code from RemoteXY is easily to be understood. Principally, the design is constructed to make the interface has similar template with the smartphones. This interface, have three button select that include, off, speed 1, and speed 2. The
command from the interface in smartphones deliver information to arduino for processing the decision. The other software that is used is Arduino Development Environment (IDE) and it is libraries code. The development environment using Java, the code libraries are using C and C++. This the main code of arduino that is used in this project.

```c
#include <SoftwareSerial.h>
#include "remotexy.h"
int PIN_SELECT_1 = 7; // SET PIN Relay Module Input 1
int PIN_SELECT_2 = 8; // SET PIN Relay Module Input 2

void setup()
{
    RemoteXY_Init ();
    RemoteXY_Init ();
    pinMode (PIN_SELECT_1, OUTPUT);    // SET POWER
    pinMode (PIN_SELECT_2, OUTPUT);   // SET POWER
    // TODO you setup code
}

void loop()
{
    RemoteXY_Handler ();

    if (RemoteXY.select_1==1) digitalWrite(PIN_SELECT_1, LOW); // Speed 1
    else digitalWrite(PIN_SELECT_1, HIGH);

    if (RemoteXY.select_1==2) digitalWrite(PIN_SELECT_2, LOW); // Speed 2
    else digitalWrite(PIN_SELECT_2, HIGH);

    // TODO you loop code
    // use the RemoteXY structure for data transfer
}
```

The explanation of the code is, if the command is taken from the interface of smartphones, the arduino will do the action. The arduino sends the command to relay module to switch the fan speeds mechanism. If there is no any command, the speed will stay from the last command of smartphone’s interface.
CHAPTER 4
RESULT AND DISCUSSIONS

4.1. Result
This final project is designed to control speed fan using Bluetooth over the smartphone. The Bluetooth module that used in this device work well. The Bluetooth module work even the smartphone is outside the room over 10 meters range. If the smartphone that paired with Bluetooth module is out of cover range then it will disconnected. The device of this final project can be seen in Figure 4.1

![Figure 4.1 Final project device](image)

Figure 4.1 Final project device

The arduino, Bluetooth module, and the relay module is putted on top of acrylic. The Bluetooth module is putted on top of arduino as seen in Figure 4.2. The fan speed is using two relay module which is relay one for speed one and relay two for speed two. All the relay module works fine when it connected to arduino while arduino received and transmit data over Bluetooth module. When a smartphone paired with Bluetooth module, the speed fan is start from off. The control interface on smartphone can be shown in Figure 4.3.
Figure 4.2 Circuit of final project

Figure 4.3 Control interface on smartphone
The control consists of three selects, the select A is indicate that fan is off and the select B is indicate the fan using speed one and the select C is indicate that fan using speed two. When smartphone slide the select from A to B then the fan will be turn on with speed one and the speed two is off, then when A to C then fan will be turn in with speed two and the speed one is off. It shown in the Figure 4.4 and 4.5 to demonstrate the device work.

**Figure 4.4 Fan with speed one**  
**Figure 4.5 Fan with speed two**

### 4.2. Discussion

This project successfully implemented the Arduino and Bluetooth module and works properly although the device is not that perfect. The Bluetooth module only can be paired with one smartphone. The time that needed for the pairing is take a time, sometimes the command from smartphone get a slow action from Arduino to relay module, sometimes relay module turn off even after the command on smartphone is to speed 1. To overcome this problem, the author suggest that the range for pairing is 1 meters and do not slide over and over again from smartphone so the arduino will not get confused received data from smartphone that will caused relay module have an error.

The strengths of the proposed system in this project are:

- The program that implemented on Arduino for controlling speed fan using
The smartphone works perfectly

- The range of Bluetooth module which 10 meter is good so the fan will be turn on even we are on the outside room

The weaknesses of the proposed system in this project are:

- The process of pairing Bluetooth module with smartphone take a time
- The adaptor that used in this device is 12V with 2A which is sometimes arduino will get hotter.
- The relay module sometimes take a time to understand command from Arduino.
CHAPTER 5
CONCLUSION AND RECOMMENDATION

5.1. Conclusion
In this final project, control speed fan using smartphone has been built. This device achieves the objectives of this project by having the following abilities as shown in Table 5.1.

<table>
<thead>
<tr>
<th>Objective</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build control speed fan using smartphones</td>
<td>The arduino uno and Bluetooth module are successfully implemented to achieve the objective of this final project.</td>
</tr>
<tr>
<td>Implement program to smartphones</td>
<td>The program RemoteXY has been successfully reading the data from arduino so it can control the fan.</td>
</tr>
</tbody>
</table>

In conclusion, the control speed fan using smartphone is working well, the device works well even it has been tested for 8 hours using while the author sleeping at night

5.2. Recommendation
Some developments can be made to improve the performance in future:

- Using Intel Galileo for made a pairing Bluetooth module over smartphone will be faster.
- The casing should be designed properly so the cable
REFERENCES


Appendix A: RemoteXY Library

A.1 Coding

/* RemoteXY.h
A RemoteXY Library - Remote device control
version 1.1.0

===========================================================
For use RemoteXY library visit website http://remotexy.com
This website will help you use the library for configuring
a remote control from a smartphone or tablet.
===========================================================
*/

#ifndef _REMOTEXY_H_
define _REMOTEXY_H_

#if defined(REMOTEXY_MODE__SOFTWARESERIAL)
#include "RemoteXY_Serial.h"
define RemoteXY_Init() RemoteXY_SerialSoft_Object = new RemoteXY_SerialSoft(REMOTEXY_SERIAL_RX, REMOTEXY_SERIAL_TX, REMOTEXY_SERIAL_SPEED, RemoteXY_CONF, &RemoteXY)
define RemoteXY_Handler() RemoteXY_SerialSoft_Object->Handler()
RemoteXY_SerialSoft *RemoteXY_SerialSoft_Object;
#endif defined(REMOTEXY_MODE__SOFTWARESERIAL)

#if defined(REMOTEXY_MODE__SERIAL)
#include "RemoteXY_Serial.h"
define RemoteXY_Init() RemoteXY_SerialHard_Object = new RemoteXY_SerialHard(&REMOTEXY_SERIAL, REMOTEXY_SERIAL_SPEED, RemoteXY_CONF,
#endif defined(REMOTEXY_MODE__SERIAL)
&RemoteXY)
#define RemoteXY_Handler() RemoteXY_SerialHard_Object->Handler()
RemoteXY_SerialHard *RemoteXY_SerialHard_Object;

#else
#error No define RemoteXY mode: REMOTEXY_MODE__XXX
#endif
#endif //_REMOTEXY_H_

/* RemoteXY_Serial.h
   A RemoteXY Library - Remote device control
   version 1.1.0

====================================================================================================
For use RemoteXY library visit website http://remotexy.com
This website will help you use the library for configuring
a remote control from a smartphone or tablet.
====================================================================================================
*/

ifndef _REMOTEXY_SERIAL_H_
#define _REMOTEXY_SERIAL_H_

#include "RemoteXY_Base.h"
#include <inttypes.h>
#include <SoftwareSerial.h>
#include <HardwareSerial.h>

class RemoteXY_Serial : public RemoteXY_Base {
    private:
        uint16_t count; // unsigned short (means max is 16 bit)
uint16_t index; // unsigned short (means max is 16 bit)
uint8_t command; // unsigned char (means max is 8 bit)
uint8_t crc;    // unsigned char (means max is 8 bit)
uint32_t timeout; // unsigned int (means max is 32 bit)
uint8_t timeouttry; // unsigned short (means max is 8 bit)

public:
RemoteXY_Serial (void * _conf, void * _var);
void Handler();

private:
virtual uint16_t rxy_available() = 0;
virtual uint8_t rxy_read() = 0;
virtual void rxy_write(uint8_t b) = 0;

void WriteByte (uint8_t c, uint8_t * crc);
void Send (uint8_t *p, uint16_t len); // *p is pointer
void Receive (uint8_t *p, uint16_t len);

};

#ifndef REMOTEXY_MODE__SOFTWARESERIAL

class RemoteXY_SerialSoft : public RemoteXY_Serial {
private:
   SoftwareSerial * serial;

public:
   RemoteXY_SerialSoft (uint8_t _rx, uint8_t _tx, long _serialspeed, void * _conf, void * _var): RemoteXY_Serial (_conf, _var) {serial = new SoftwareSerial (_rx, _tx);serial->begin (_serialspeed);};

public:
   uint16_t rxy_available() {return serial->available();};
   uint8_t rxy_read() {return serial->read();};
   void rxy_write(uint8_t b){serial->write (b);};

};
#endif
class RemoteXY_SerialHard : public RemoteXY_Serial {
private:
    HardwareSerial *serial;

public:
    RemoteXY_SerialHard (HardwareSerial *serial, long _serialspeed, void * _conf, void *
    _var): RemoteXY_Serial (_conf, _var) {serial = _serial;serial->begin (_serialspeed);};

private:
    uint16_t rxy_available(){return serial->available ();};
    uint8_t rxy_read(){return serial->read ();};
    void rxy_write(uint8_t b){serial->write (b);};
};
// #define REMOTEXY_MODE__DEBUGLOGS

#include "RemoteXY_Serial.h"
#include <Arduino.h>
#include <inttypes.h>

#ifndef REMOTEXY_MODE__DEBUGLOGS
#include <HardwareSerial.h>
#endif

#define REMOTEXY_TIMOUT 300  
#define REMOTEXY_TIMOUT_TRY 10  
#define REMOTEXY_TIMOUT_DISCONNECTED 3000

RemoteXY_Serial::RemoteXY_Serial (void * _conf, void * _var):RemoteXY_Base (_conf, _var) {
    index = 0;
    crc = 0;
    timeout_try = REMOTEXY_TIMOUT_TRY;
#ifndef REMOTEXY_MODE__DEBUGLOGS
    Serial.begin (9600);
#endif
}

void RemoteXY_Serial::Handler () {
    uint8_t c;
    uint32_t tim;
    while (rxy_available () > 0) {
        c = rxy_read ();
#ifndef REMOTEXY_MODE__DEBUGLOGS
        if (index==0) Serial.write("\r\n< ");
        Serial.print(c, HEX);
        Serial.write(" ");
#endif
    }
} // CRemoteXY

void RemoteXY_Serial::Handler () {
    uint8_t c;
    uint32_t tim;
    while (rxy_available () > 0) {
        c = rxy_read ();
#ifndef REMOTEXY_MODE__DEBUGLOGS
        if (index==0) Serial.write("\r\n< ");
        Serial.print(c, HEX);
        Serial.write(" ");
#endif
    }
} // CRemoteXY
if (index==0) crc=c;
else crc+=c;
switch (index) {
    case 0:
        count=c;
        break;
    case 1:
        count+=c<<8;
        break;
    case 2:
        command=c;
        break;
    default:
        if (index-3<var_length)
            *(var_buffer+index-3)=c;
} index++;
timeout=millis();
timeout_try = 0;
if ((index >= 4) && (index == count) && (count >= 4)) {
    if (crc == 0) {
        switch (command) {
            case 0x00:
                Send (conf, conf_length);
                break;
            case 0x40:
                Send (var, output_length+input_length);
                break;
            case 0x80:
                Receive (var, index-4);
                Send (0, 0);
                *connect_flag=1;
                break;
            case 0xC0:
Send (var+output_length, input_length);
*connect_flag=1;
break;
}
}
index = 0;
}
}
if (millis()-timeout>REMOTEXY_TIMEOUT) {
index = 0;
if (timeout_try<REMOTEXY_TIMEOUT_TRY) {
timeout_try++;
#endif
remotexy_DEBUGLOGS
Serial.print("0");
#endif
rxy_write (0);
timeout=millis();
}
else *connect_flag=0;
}
} //Handler

void RemoteXY_Serial::WriteByte (uint8_t c, uint8_t * crc) {
#ifdef REMOTEXY_MODE__DEBUGLOGS
Serial.print(c, HEX);
Serial.write(" ");
#endif
rxy_write (c);
*crc-=c;
} //WriteByte

void RemoteXY_Serial::Send (uint8_t *p, uint16_t len) {
uint8_t c;
uint8_t crc = 0;
uint16_t i = len+4;
#ifdef REMOTEXY_MODE__DEBUGLOGS
  Serial.write("\r\n> ");
#endif

WriteByte (i & 0xff, &crc);
WriteByte ((i & 0xff00)>>8, &crc);
WriteByte (command, &crc);
while (len--) {
  WriteByte (*p++, &crc);
}
#endif REMOTEXY_MODE__DEBUGLOGS
Serial.print(crc, HEX);
#endif

rxy_write (crc);
}  //Send

void RemoteXY_Serial::Receive (uint8_t *p, uint16_t len) {
  uint8_t *pi = var_buffer;
  while (len--) *p++=*pi++;
}  //Receive

/* RemoteXY_Base.h
   A RemoteXY Library - Remote device control
   version 1.1.0

   ****************************************************************************
   For use RemoteXY library visit website http://remotexy.com
   This website will help you use the library for configuring
   a remote control from a smartphone or tablet.
   ****************************************************************************

   */
#endif _REMOTEXY_BASE_H_
#define _REMOTEXY_BASE_H_
#include <inttypes.h>
class RemoteXY_Base {
    protected:
        uint8_t output_length;
        uint8_t input_length;
        uint16_t conf_length;
        uint8_t *conf;
        uint8_t var_length;
        uint8_t *var;
        uint8_t *connect_flag;
        uint8_t *var_buffer;

    public:
        RemoteXY_Base (void * _conf, void * _var);
    };
#endif
/* RemoteXY_Serial.cpp
 A RemoteXY Library - Remote device control
 version 1.1.0

 For use RemoteXY library visit website http://remotexy.com
 This website will help you use the library for configuring
 a remote control from a smartphone or tablet.

 */

#include "RemoteXY_Base.h"
#include <stdlib.h>
#include <inttypes.h>
#define REMOTEXY_OUTPUT_LENGTH_INDEX 0
#define REMOTEXY_INPUT_LENGTH_INDEX 1
#define REMOTEXY_CONF_LENGTH_LO_INDEX 2
#define REMOTEXY_CONF_LENGTH_HI_INDEX 3
#define REMOTEXY_CONF_INDEX 4

RemoteXY_Base::RemoteXY_Base (void * _conf, void * _var) {
    output_length = *((uint8_t*)_conf+REMOTEXY_OUTPUT_LENGTH_INDEX);
    input_length = *((uint8_t*)_conf+REMOTEXY_INPUT_LENGTH_INDEX);
    conf_length = *((uint8_t*)_conf+REMOTEXY_CONF_LENGTH_HI_INDEX);
    conf_length = (conf_length<<8) +
    *((uint8_t*)_conf+REMOTEXY_CONF_LENGTH_LO_INDEX);
    conf = (uint8_t*)_conf+REMOTEXY_CONF_INDEX;
    var_length = output_length+input_length+1;
    var = (uint8_t*)_var;
    var_buffer = (uint8_t*)malloc (var_length);
    connect_flag = var+var_length-1;

    uint8_t* p = var;
    for (uint8_t i=0;i<var_length;i++) *p++=0;
} //RemoteXY_Base