REMOTE CONTROL CAR STARTER (RCCS)

MOHAMMAD AKMAL BIN AKASHAH

UNIVERSITY COLLEGE OF ENGINEERING & TECHNOLOGY MALAYSIA

REMOTE CONTROL CAR STARTER (RCCS)

MOHAMMAD AKMAL BIN AKASHAH

A thesis submitted in partial fulfillment of the
requirement for the awarded of the Degree of Bachelor of Electrical &
Electronic Engineering

Faculty of Electrical & Electronic Engineering
University College of Engineering and Technology Malaysia

MAY 2006

KOLEJ UNIVERSITI KEJURUTERAAN DAN TEKNOLOGI MALAYSIA

BORANG PENGESAHAN STATUS TESIS

JUDUL **REMOTE CONTROL CAR STARTER (RCCS)**

SESI PENGAJIAN: **2005/2006**

Saya **MOHAMMAD AKMAL BIN AKASHAH (830119-02-5153)**
(HURUF BESAR)

mengaku membenarkan tesis (PSM/Sarjana/Doktor Falsafah)* ini disimpan di Perpustakaan Kolej Universiti Kejuruteraan dan Teknologi Malaysia dengan syarat-syarat kegunaan seperti berikut:

1. Tesis adalah hakmilik Kolej Universiti Kejuruteraan dan Teknologi Malaysia.
2. Perpustakaan Kolej Universiti Kejuruteraan dan Teknologi Malaysia dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan tesis ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. **Sila tandakan (✓)

☐	SULIT	(Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)
☐	TERHAD	(Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi/badan di mana penyelidikan dijalankan)
☑	TIDAK TERHAD	

Disahkan oleh:

_____
(TANDATANGAN PENULIS)
Alamat Tetap:
NO 225 KM. 6 JLN SG KOROK,
MUHAMMAD
05400,ALOR SETAR,
KEDAH DARUL AMAN.

Tarikh: 2 MAY 2006.

Disahkan oleh:

_____
(TANDATANGAN PENYELIA)
Penyelia:
ENCIK BADARUDDIN B

Tarikh: 2 MAY 2006

CATATAN:
* Potong yang tidak berkenaan.
** Jika tesis ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali sebab dan tempoh tesis ini perlu dikelaskan sebagai SULIT atau TERHAD.
◆ Tesis dimaksudkan sebagai tesis bagi Ijazah Doktor Falsafah dan Sarjana secara penyelidikan, atau disertai bagi pengajian secara kerja kursus dan penyelidikan, atau Laporan Projek Sarjans Muda (PSM).

"I hereby declare that I have read this thesis and in
my opinion this thesis is sufficient in terms of scope and
quality for the award of the Degree of
Bachelor of Electrical Engineering (Electronics)"

Signature	:	.....
Supervisor	:	Mr. Badaruddin bin Muhammad
Date	:	2 MAY 2006

I declare that this thesis entitled "***Remote Control Car Starter (RCCS)*** "
is the result of my own research except as cited
in the references. The thesis has not been accepted for any
degree and is not concurrently submitted in candidature of any other degree.

Signature : ……

Author : MOHAMMAD AKMAL BIN AKASHAH

Date : 2 MAY 2006

To my beloved father and mother
who always give me moral support to finish this thesis.

Also, to those who gave guidance and inspiration for me throughout the journey of this project. Thank you for the supports and advices that have been given.

AKNOWLEDGEMENT

First and foremost I want to thank God for blessing me with health and time so that I can finish my thesis completely.

Beside that, I also want to thank my supervisor, Encik Badaruddin bin Muhammad for all his guideline, ideas and supervised me so that I can successfully finish my project and my thesis.

Not forgotten my fellow friend whom supported me and helping me with my project. All the staff involved in giving the opportunity in finishing my project.

Lastly, I want to thank my parent who never gives up supporting me and believing in me in whatever I do. May God bless you all.

ABSTRAK

Projek ini adalah untuk mencipta penghidup kereta kawalan jauh. Kegunaan projek ini adalah untuk menghidupkan kereta dalam jarak jauh. Projek ini mengandungi dua bahagian yang penting iaitu bahagian pemancar dan penerima.Apabila isyarat dikesan pada pemancar, pemancar akan menghantar data ke pemproses mikro dan serentak akan menghidupkan enjin (melalui keluaran relay) Pemproses mikro yang digunakan dalam projek ini adalah jenis PIC16F84A.Dalam teknologi yang serba pantas pada masa ini, rekaan baru dicipta untuk membuat hidup lebih selesa. Sebagai contoh, cuaca yang dingin pada waktu pagi dan panas pada waktu tengahari, pemandu akan menghadapi masalah dalam menunggu kereta mereka untuk enjin dipanaskan serta untuk sejukkan bahagian dalaman. Antara sebab seperti untuk penyempurnaan dan megelak daripada enjin sejuk membawa kepada terciptanya sebuah projek yang dinamakan penghidup kawalan jauh ini.

ABSTRACT

This project is to develop a remote control car starter. The use of this project will be able to start the car remote in a distance range. This project consist two important parts which is transmitter and receiver. When a signal has been detected at the transmitter, the transmitter will send the data to microcontroller and thus bypass the connection in the car (output from relay). The microcontroller that has been used in this project is PIC16F84.Today's fast paced technological developments strive to make our lives comfortable. For example, with the low temperatures in morning and the hot temperatures in the afternoon, drivers suffer while waiting for their car to heat up or cool down. Other reasons such as proper operation and protection from freezing of the engine brought forth the development of the remote engine starter.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

RCCS	-	Remote Control Car Starter
VDC	-	Direct Current Voltage
PCB	-	Planted Circuit Board
PIC	-	Programmable Integrated Circuit
LED	-	Light Emitter Diode
V	-	Volt
A	-	Ampere
I	-	Current
LF		Low Frequency
HF	-	High Frequency
MHZ	-	Mega Hertz
W	-	Watt
P	-	Power

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Background

In this chapter, it explains briefly on the background and system overview of this project. Also in this chapter it explains the objectives of the project, the scope of study and the thesis outline.

1.2 System Overview

The purpose of this project is to design and implement a remote car starter. This remote control engine starter will include three main components: A HF transmitter, a HF receiver and a PIC microcontroller. The system will start by generating a bit pattern from the transmitter module that will be serially inputted. The transmitter will then modulate the bit pattern so that the signal can be transmitted through the analog channel, namely air.

Estimating a goal to be around 10 meter, estimated that the signal would operate at a frequency of about 315MHZ. On the other side, a HF receiver will be able to pick up signals around the same frequency as the transmitter and it will

receive the modulated signal. Once the receiver obtains the signal, it will modulate the analog signal back into bit patterns where will then be sent to the receiver's module end the PT2272 will distinguish the correct signal from the incorrect signals such as noise and send signal to the PIC microcontroller to perform specific functions to start the car.

This project mainly is practical and useful. This project is related to engineering as well as applicable to something that can be related on a daily basis. This project combines both of main part in hardware and software.

This project will operate at least approximation of 10 meter of range. It is considered 10 meter because that a substantial distance is used so that it will be practical and possible for to implement. The design of this project is to perform as many tasks as possible so the device would be as close as possible to a real remote car starter. Such tasks included starting the car and stop after a timed of period. This will allow the car owner to manually stop the program.

The performance of this project will depend a great deal on the transmitter and the receiver and whether or not the signal can be sent from one to the other. The microcontroller chips will have no variables based on the program given, which means it can be considered a constant. The only variable is the transmission of the signal, which can be affected by noise in air as well as competing with other frequencies in laboratory area. This variable is needed to be minimized as much as possible.

1.3 Objectives

The objective of this project is to produce a system using remote control as main transmitter besides using PIC16F84 as the microcontroller. This project can be used as one of the main part of starting the car using a remote device.

The signal should be successfully transmitted from transmitter to the receiver and processed at the PIC controller board. The Relay board should take particular action to the car such as start and turn off the car engine.

1.4 Scope of Study

The scope of study covers the following areas:

a) Remote system - the remote transmitting and receiving data that is used in this project is RF transmitter and receiver module. It will transmit 3 different channels of data. Each channel will react as different operation.

b) Microcontroller - the microcontroller used is PIC16F84. The program will be uploaded to the PIC using software called MPlab programmer which enable the program to be burn into the chip. Program format used is the assembly language and it will be in hex file.

c) Power system - basic switching suggested for this project are 12V relays.

d) Voltage regulator - the voltage regulator is used to step down the supply voltages from 12V to 5V because PIC microcontroller will operates in the range of 2V to 6V only.

e) Car Wiring- the wiring should be done correctly without any error. This is to prevent short circuit and losses at the power cable.

1.5 Thesis Outline

Chapter 1 explains the background of the project. In this chapter it explains briefly on the background and overview of this project. Also in this chapter it explains the objectives of the project, the scope of study and the thesis outline.

Chapter 2 will briefly explain on how the existing project works and operates with the theories. The existing project main function is similar to this project which is to start the engine of the car using the remote device but some different functions from are added on the Remote Control Car Starter (RCCS).

Chapter 3 will explain on the methodologies used for the project. It also focuses on the hardware and the software implementation of the project. The detail of car wiring connection for the project also explained in this chapter. This chapter will also focus on the user manual and the user safety precaution.

Chapter 4 will focus on the results and discussion on the RCCS project done. The results will be explained in tables and discussion will be discussed due to project implementation

Chapter 5 will focus on the conclusion for the RCCS project. It also will discuss on the suggestion that are needed for future development of this project.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In this chapter it will briefly explain on how the existing project works and operates with the theories. The existing project main function is similar which is to start the engine of the car using the remote device but some different functions from the Remote Control Car Starter (RCCS).

2.2 Internet

In today's market this device already has been market with different functionality. From the internet it is found out that a similar project which is named as keyless entry car starter and auto car starter have been in market. For the keyless entry car starter, it do not need key to start and drive the car and it only need a remote control device to start and drive the car. The remote seem to be much more complex with the LCD screen assembled on the remote device. This is the main reason of the higher and expensive price of the device. The LCD will show the current status of the car whether the engine is started or not. For the auto car starter device it comes with the alarm system. It will only function directly to the button pressed on the remote. Even though the engine car is already been started it will

repeat the starting process if the owner of the car press the start switch for the second time. This is the disadvantage of the system which it do not have safety for the system to disable the starting process of the engine to avoid crank on the engine. The price of this device still is higher than expected price of the system due the system includes the alarm system. The price of this system only can be reduced by excluding the alarm system from the system.

2.3 Accessories Shop

From the accessories shop, it is found out that the similar project is already been market. The theory of the system function is that is similar device found out in the internet which it only will start due to the switch on the remote is pressed. If the engine starting switch is pressed for the second time it will still start the car although the engine is already running. This is not safe for the car owner which it will produce crank on the engine of the car. This will also make the starter to cause breakdown. The lifespan of the engine would probably be shorter due to inadequate system installed on the car. This system should be improved by adding a feedback to the system which will disable the engine starting process which will only start the car once thus avoiding the crank on the engine.

2.4 System Theories

The main theory of the system is to start the engine by using the remote system. It is to solve the problem of the car owner comfort ability. Imagine if the car engine is cold and uncomfortable to be driven in the early morning, this will be the solution which it will heat up the engine without needing the owner to be in the car. It is simpler to say that the system operates in the range of distance which will not need the owner to get into the car. The theories should also follow the step of the

system functions. First it will need the remote and receiver to operate the system which the transmitter will transmit data and the receiver will receive data and the data received should operate the system as expected results.

From the receiver it will be connected to the chip of the controller which will control the functions needed on the car. This will be connected to the car existing wiring. If the connection are correctly connected to the car existing wiring, it will function as same as the car switch is being turned-on during the ignition of the engine process is done with the key enters the key hole. But with this system, it do not need the car key to enter the main switch of the car. It only needs to receive a signal from the remote device to tell the car to start by its own.

CHAPTER 3

HARDWARE AND SOFTWARE IMPLEMENTATION

3.1 Introduction

This chapter will explain on the methodologies used for the project. It also focuses on the hardware and the software implementation of the project. The detail of car wiring connection for the project also explained in this chapter. This chapter will also focus on the user manual and the user safety precaution.

3.2 Block Diagram

From the block diagram in figure 3.2.1 shows how the remote engine starter system flows. All connection is made according to the above block diagram. From the car engine the wiring are connected to the air conditioner and the starter motor and many other parts such as the alternator, the distributor and many other lamp wiring. It needs to be focus on the starter motor and the air conditioner. The starter motor and the air conditioner are then connected to the Fuse Box for the components safety and finally connected to the Computer Box of the car before each switches of the operation. It will bypass the air conditioner at its air conditioner switch directly but for the starter it has to be bypass at the main key switch.

The signal from the RF transmitter module will be transmitted to the RF receiver module. When the receiver received the signal, the signal will be processed at the PIC controller board. The relay board after the controller board functions as the safety board which will cut-off current flow through the connection to the controller board. The controller board will separate each signals into channels. Each channel has different function for the car. For the time being, it will separate into two channels which is to start the car and to on and off the air conditioner. The system will bypass directly the air conditioner switch which will be either in the normally closed when the system needs to turn on the air conditioner and normally open when it want to switch off the air conditioner.

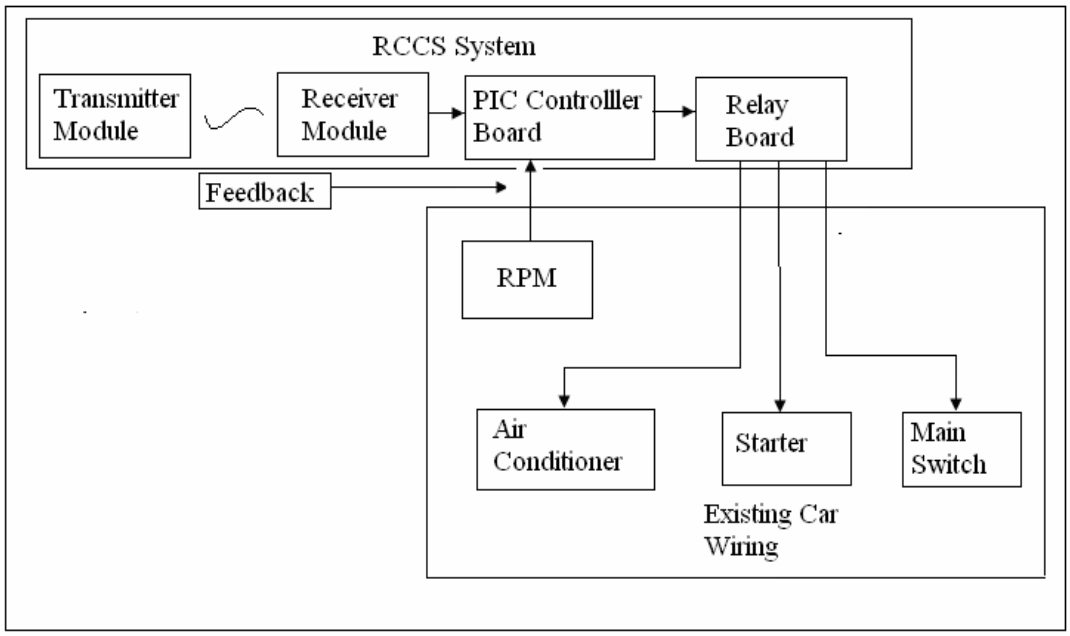


Figure 3.2.1 The Operation Block Diagram with Feedback

3.3 PIC Configuration

Figure 3.3(a) show the pin configuration for PIC16F84A. It consist of two ports which are Port A and Port B .Port A are located at pin no1, 2, 3, 17 and 18. Port B is located at pin number 6, 7, 8, 9, 10, 11, 12 and 13. The ports are bidirectional where it can be initialized as input port or output port. This will depend on the programming. Commonly for assembly language it will equate the ports at the storage register.

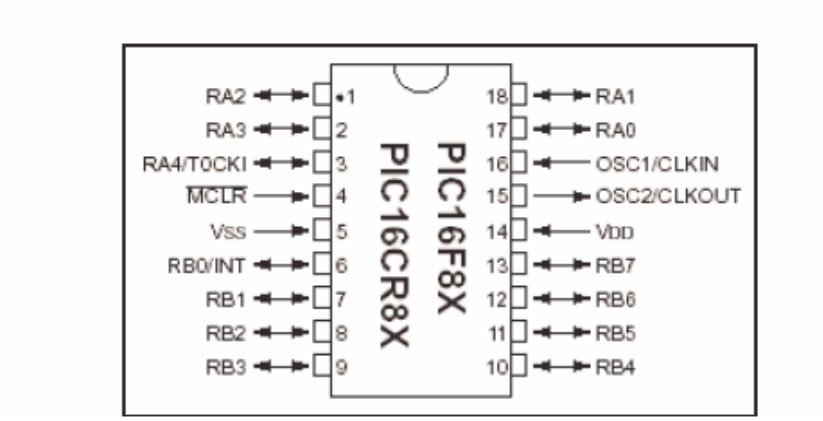


Figure 3.3(a) Pin Configuration

3.3.1 The PIC16F84A

PIC is the name for the microchip microcontroller (MCU) family, consisting of a microprocessor, input/output ports, timers and other internal, integrated hardware. The main advantages of using the PIC are low external part count, a wide range of chip sizes (now from 5-pin up) available, nice choice of compilers (assembly, C, BASIC, etc.) good wealth of example/tutorial source code and easy programming. Once bought, the PIC's program memory is empty, and needs to be programmed with code (usually HEX files) to be usable in a circuit. For the purpose,

a wide range of simple programmer hardware docs and software is downloadable from the net.

In figure 3.1.2 it show the PIC16F84 microcontroller chip pin configuration. This microcontroller is used in the remote control car starter to read data from the receiver module. The outputs from the receiver module can be connected to any of the RB0-RB7 ports. If any of the inputs is a high (5V supplied from receiver) the output from the PIC will be on port A which is RA0-RA3. It will output a high of +5V, which will then be connected to relay board. The relay will toggle the switch of each function.

3.3.2 PIC16F84A Architecture

A microcontroller is an inexpensive single chip computer. Single chip computer means that the entire computer system lies within the confines of the integrated circuit chip. The microcontroller is capable of storing and running a program. The microcontroller ability to store and run unique programs makes it extremely versatile. The microcontroller ability to perform math and logic function allows it to mimic sophisticated logic and electronic circuits. It contain a CPU,RAM(random access memory),ROM(read only memory),I/O(input/output lines)serial and parallel port, timers and sometimes other built in peripheral such as A/D(analog to digital) and D/A(digital / analog) converter. The microcontroller used in this project is microcontroller chips called PIC. The PIC16F84A belongs to the mid-range family of the PIC microcontroller devices. The program memory contains 1K words, which translates to 1024 instructions, since each 14-bit program memory word is the same width as each device instruction. The data memory (RAM) contains 68 bytes. Data EEPROM is 64 bytes .There is also 13 I/O pins that are user-configured on a pin-to-pin basis.

The PIC chips have two separate 'data' busses, one for instructions and one for everything else. Instructions are essentially in ROM and dedicate the microcontroller to doing one task. Then there is one series that is of special interest to the hobbyist, the 16F84, chips which have electrically reprogrammable EEPROM memory for instructions. A block diagram of PIC16F84 is shown in figure3.3.2 (a):

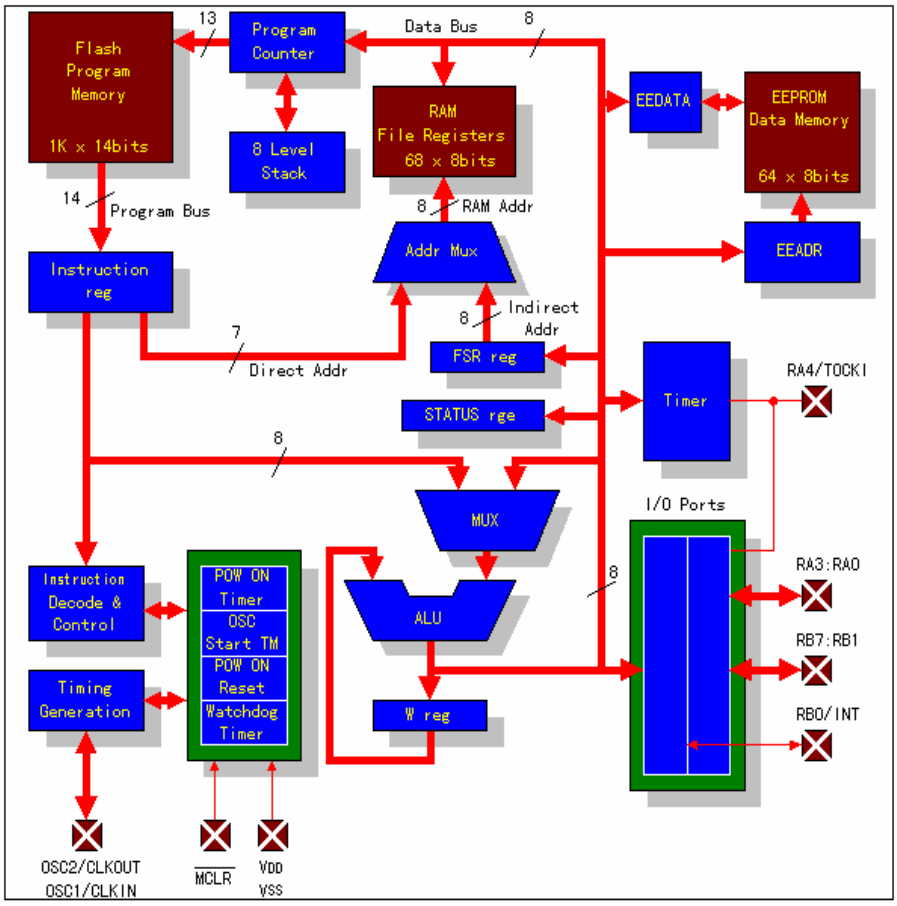


Figure 3.3.2(a) PIC Architecture

3.3.3 Flash Program Memory

Flash memory is used to store the program. One word is 14 bit long and 1024 word (1k word) can be stored. Even if the power is switched off the contents of the flash will not be lost. Flash memory can be written using the writer, but the number of times it be rewritten is limited to 1000 times. Figure 3.1.2(a) show the detail of flash memory register

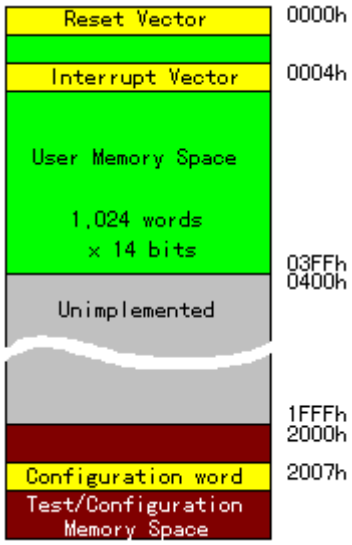


Figure 3.1.2(a) Flash Memory

3.3.4 PIC Address Register

The PIC microcontroller used a Harvard architecture, which means that the memory is divided into program memory and data memory. The advantage to this architecture is that both memories can be accessed during the same clock instruction; this makes it faster than the standard Von Neumann architecture, which uses single memory for program and data. User program memory space extends from 0x0000h to 0x033FFh, accessing a memory space above 03FFh will cause a wrap around to

the beginning of the memory space. Figure 3.3.4 (a) show the register map. This memory is partitioned into two space called banks. The interrupt vector is shown at FSR 04h. Upon an interrupt, the return address is saved and the program execution continues at the return address previously saved

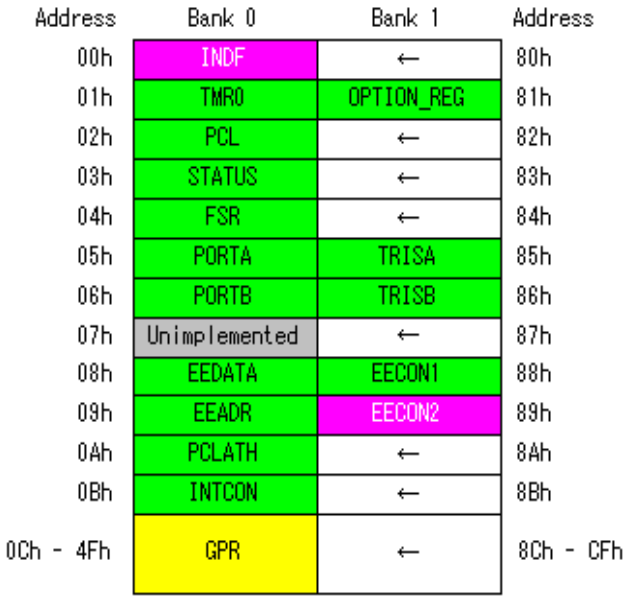


Figure 3.3.4 (a) PIC Register Map

3.3.5 PIC Controller Circuit Diagram

Figure 3.3.5 (a) shows the PIC Controller Circuit Diagram. It uses the Orcad PSpice to draw the circuit. This circuit acts as the main brain of the RCCS system. This is very important to make sure that this part operates appropriately. The LED on this circuit indicates the signal is transmitted to the PIC. It consists of five LEDs but in real simulation it only uses three LED. Each LED represents each channel for the system. This circuit only uses 3 channels for receive and transmit.

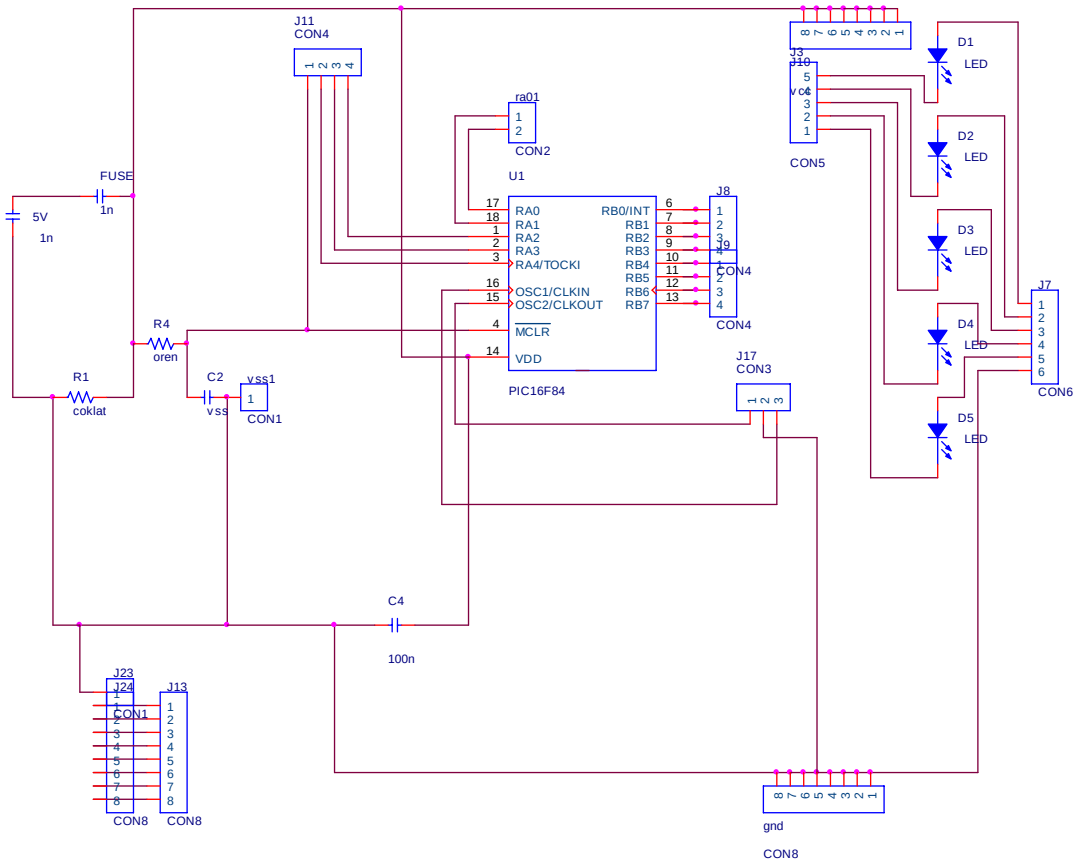


Figure 3.3.5 (a) PIC Controller Circuit Diagram

3.4 Plotting the PCB board

In this section, the plotter is used to design the circuit on the PCB. The most usual problem that is discovered if using the plotter is that the bit was very sensitive. The 0.2mm plotter bit is used to plot the circuit design on the PCB. After plotting it will put off the entire unwanted layer on the PCB with the 1.0mm plotter bit. After the entire part was put off it will drill the unplanted part. The drilling procedures will need the 0.8mm drill bit. It will drill all the unplanted part.

3.5 Relay Board Circuit Diagram

Figure 3.5.1 show the circuit of the basic switching method. In the basic switching method of the relay, the relay need transistor as the main switch to energized it. The transistor reacts as to ground the connection of the relay while maintaining the current flow through the transistor. The transistors used are the NPN type.

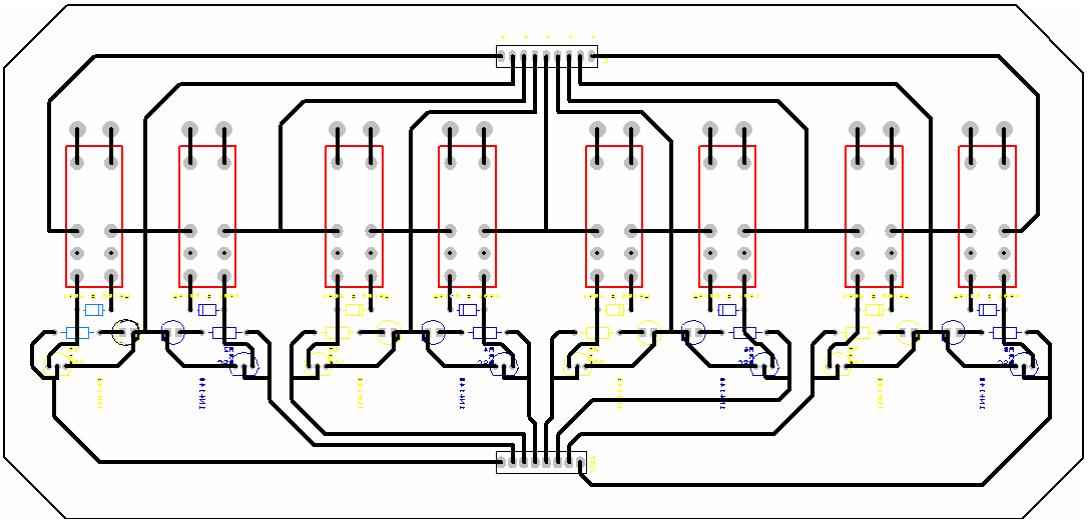


Figure 3.5.1 Relay Circuit Diagram

3.6 Voltage Regulator

In this project, it uses the LM78L05 Voltage Regulator to buck the voltage from 12VDC to 5VDC. The Voltage Regulator was used due to some device needs operation in 5VDC. The voltage regulator LM78L05 version is used in this project The reason why it used the voltage regulator was that it only need one main voltage supply input from the car. As the car battery only operated as 12VDC. Due to 12VDC, it needs to buck the voltage supplied to 5VDC. Refer to figure 5 in appendices for the real photo taken on the Voltage Regulator.

3.6.1 General Description of LM78L05

The LM78LXX series of three terminal positive regulators is available with several fixed output voltages making them useful in a wide range of applications. When used as a zener diode/resistor combination replacement, the LM78LXX usually results in an effective output impedance improvement of two orders of magnitude, and lower quiescent current. These regulators can provide local on card regulation, eliminating the distribution problems associated with single point regulation.

The voltages available allow the LM78LXX to be used in logic systems, instrumentation, HiFi, and other solid state electronic equipment. The LM78LXX is available in the plastic TO-92 (Z) package, the plastic SO-8 (M) package and a chip sized package (8-Bump micro SMD) using National's micro SMD package technology.

With adequate heat sinking the regulator can deliver 100 Ma output current. Current limiting is included to limit the peak output current to a safe value. Safe area protection for the output transistors is provided to limit internal power dissipation. If internal power dissipation becomes too high for the heat sinking provided, the thermal shutdown circuit takes over preventing the IC from overheating.

3.6.2 Features of LM78L05:

i. LM78L05 in micro SMD package Output voltage tolerances of ±5% over the temperature range
ii. Output current of 100 Ma
iii. Internal thermal overload protection
iv. Output transistor safe area protection
v. Internal short circuit current limit
vi. Available in plastic TO-92 and plastic SO-8 low profile packages

vii. No external components

viii. Output voltages of 5.0V, 6.2V, 8.2V, 9.0V, 12V, 15V

Table 3.6.3 Voltage Regulator Power Dissipation

Power Dissipation	**Internally Limited**
Input Voltage	35V
Storage Temperature	−65°C to +150°C

Table 3.6.4 Operating Junction Temperature

SO-8	0°C to 125°C
micro SMD	−40°C to 85°C

Table 3.6.5 Soldering Information

Infrared or Convection	(20 sec.) 235°C
Wave Soldering (10 sec.)	260°C (lead time)
ESD Susceptibility (Note 2) 1Kv	

3.6.3 LM78L05 Electrical Characteristics

Limits in standard typeface are for TJ = 25°C, Bold typeface applies over 0°C to 125°C for SO-8 package and −40°C to 85°C for micro SMD package. Limits are guaranteed by production testing or correlation techniques using standard Statistical Quality Control (SQC) methods.

Unless otherwise specified: IO= 40 mA, CI = 0.33 μF, CO = 0.1 μF.

3.7 Air Conditioner Wiring

The figure 3.7.1 show how the connection of the air conditioner is done. Firstly, this is how and where the wiring of the air conditioner is done. It should be tested using the multi meter. The purpose of testing with the multi meter is to identify which wires need to be bypassed at the switch.

There are two main devices that will be turned on during the switch-on air conditioner operation. So the wiring needs to be connected in pairs to two single relays in the RCCS. It needs two different relays because it will turn on two different devices which the pair of wiring should not be connected together. The two devices are the air conditioner fan and the compressor of the air conditioner. Both relays will energize during the operation of turning on the air conditioner. At the same time it will turn on the air conditioner. It will allow current to flow through and the air conditioner will be turned on. Figure 7.3.1 shows how the complete connection of the air conditioner. With this connection one channel is complete and ready to operate.

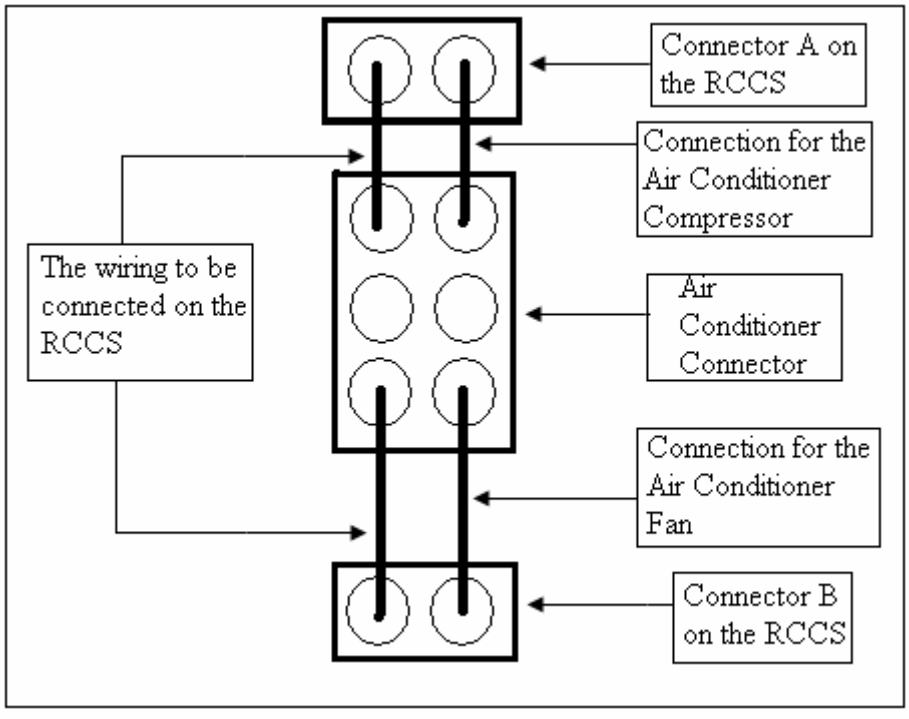


Figure 3.7.1 Air Conditioner Connection

3.8 The Feedback Wiring

Figure 3.8.1 show how the connection of the car feedback wiring needs to be done. It is important to test by using multi-meter on the functionality of the feedback. This feedback wiring must be connected to the RPM meter. The testing must be done to check whether voltage supplied varied during the running of the engine and during the engine is stop. From the RCCS it needs zero voltage supplied during the engine is off and it needs 12 volt supply from the feedback during the car engine is started. This variable voltage is obtained from the existing car wiring. The 12 volt supply will be then converted to 5 volt through the voltage regulator. This step is done to make sure that only 5 volt output will be supplied to the PIC input pin. The program of the system will be executed due to the feedback responses.

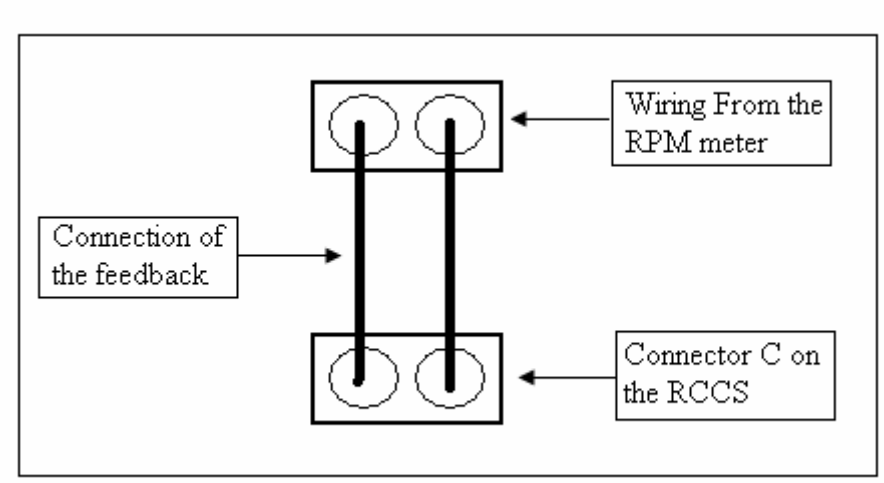


Figure 3.8.1 Feedback Connection

3.9 Main Switch Wiring

This part is the most important part of the RCCS system wiring. Refer to figure 3.9.1 for the connections of the main switch wiring. It needs to be done correctly and carefully due to prevent from error and losses. It needs to be tested independently on each of the function of the existing car switch. The testing should be considered on during the main switch turns on, during the accessories switch turns on and during the starter switch turns on. All this is tested using the multi-meter. User has to make sure that it is done correctly with the accepted wire size. The wire size needs to be at least 3 Ampere maximum current flow that can be flow through it. This is to prevent from high losses during the operation of the RCCS system.

If small wire size is used and the current flow of the wiring exceeds maximum current of the wire provided, high losses will occur and respectively make the wire to burn up. So this have to be considered as the safety precaution step needed to be followed by the user of this system. From the figure, it shows how the real connection of the RCCS system. There is 12 volt supply wire that is connected in the existing car wiring. This 12 volt supply needs to be shared for the main car switch, the accessories switch and the car starter switch. Every pair of the switching is independently connected to different relays. This is due to every switch needed to be switch on independently. Every pair of connection is connected to the normally open at the relays.

As the relay energized, it will turn on to normally closed and switch on the connection connected to it. For this part the system needs three different relays to operate the system. The first relay needed for turning on the main switch. The second relay needed to turn on the accessories switch and the third relay needed to turn on the car started switch. The car starter relay will be disabled if car feedback is obtained from the RPM wiring. For this part, it requires two different channels for system operation. One channel is needed to turn on the main switch and the accessories switch and the second channel needed to turn on the starter switch.

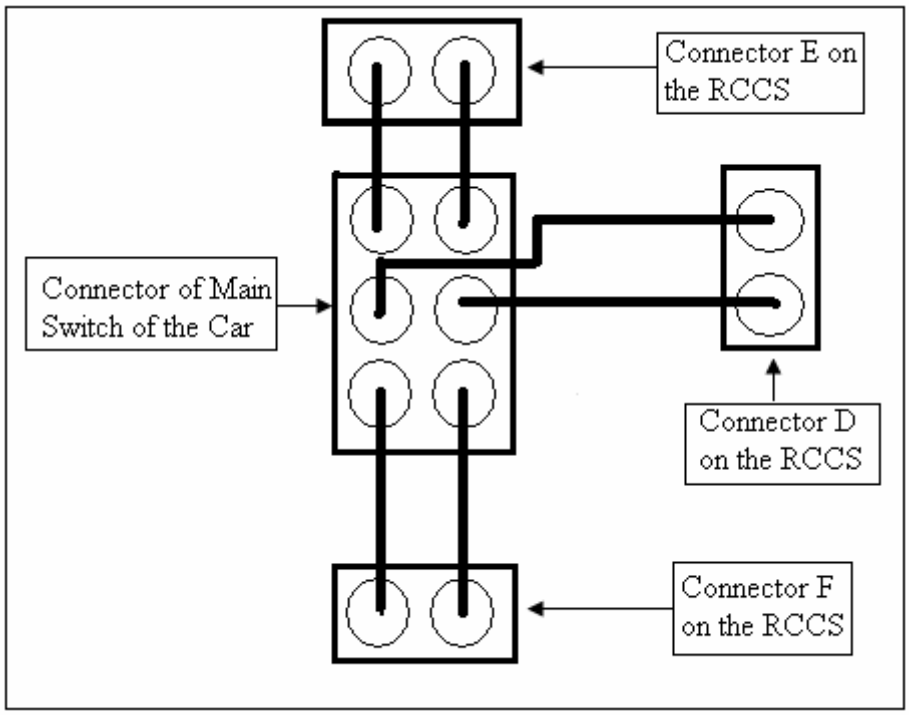


Figure 3.9.1 Main Switch Connection

3.10 Users Guide

This device is a programmed device which will allow users to communicate with car using the remote controller. It is divided into four main switches. Only three switches are usable which are the Button A switch, Button B switch and Button C switch. This is known as different channels used where every switches have it own operation. (Refer to appendix A in appendices)

For Channel A, which is the Button A switch, it will operate as to switch on the main power of the car switch. It will toggle between each time it is push on. The car main power will turn on when the Channel A button is pressed and the main power of the car will turn of when the Channel Button A is pressed for the second time. At this period, the car engine is not started.

For Channel B, which is the Button B switch, it will operate as to turn on the ignition for the starter. For Channel B it is important to make sure that the feedback

wiring for the car is correctly connected as stated in sub-chapter 7.4. The feedback is important because it will disable the starter to repeat starting process if the car already started. This is for the safety of starter motor and battery saving mode. With correct feedback connection, the system of RCCS will react intelligently as not produce engine crank (unwanted noise which occurred between the rotor of the engine starter motor).

For channel C, it will turn on the air conditioner. It will toggle on and off each time the button C is pressed.

3.11 User Safety Precaution

Users must follow the safety precaution to prevent from error or breakdown of the RCCS system. The safety precaution should also be followed by users to prevent from breakdown of the existing car wiring.

The first safety precaution is to make sure that the size of wire used to connect the wiring of RCCS to the existing car wiring is correct size. This is due to power losses occur during the transmission of the current source. If excessive power losses occurred the losses will produce heat which will end up the wiring to burn down. So make sure that the wire used is capable to maintain current flow up to 3 Ampere. The maximum current flow is 1.2A for the car motor starter. For the main switch and air conditioner, the current flow only 0.5 ampere.

Make sure that the feedback of the RCCS system is connected correctly to the Car Existing Wiring. This is to make sure that the program flows are correctly runs as programmed in the system of RCCS.

Always make sure that the grounding of the car wiring exists. This can be done by checking with the multi-meter. To do so, user need to test the connection of main ground supply (negative terminal of car battery) with the car body. If there are

no connections, connect a 5mm wire cable from the negative terminal of car battery to car body. This step is to make sure that system breakdown will not happen to the system.

Always check the negative and positive terminal of the wire cable is connected correctly. Also check the fuse functionality. Turn off the RCCS system after inserting the car key. This is for the purpose to maintain the existing air conditioner switch function.

The RCCS system only will operate in the 20 meters range of distance. So it will not operate if the user pressed button in further than the range stated.

3.12 Software Implementation

Refer to appendix P for main program for the RCCS system. Basically, there is a program involved in this project and the program language used is assembly language.

The program is the main brain of this project. This program is written to enable the system to recognize a correct signal and output to the car system. Ports are also configured to enable to recognize feedback from the car system and interrupts are also enabled. For example, my program would be able to cut off power to the car system if the stop button is pressed without the key inserted into the car. In addition, my device would be able to time the program appropriately based on whether if the car's engine has been started successfully or not.

Timers are used to time the process of the microcontroller and counters are used to keep track of the number of attempts at starting the engine. For example, the device would only attempt to start the engine twice before it fails.

3.12.1 Starter Program

The starter program is shown as in appendix P in the appendices. It is program according to the instructions in the datasheet (refer appendix Q). The program was successfully done with no error. The program is done using the assembly language.

3.12.2 Program Using Mplab

The starter main program is done using the MPLAB IDE version 7.11 software which it is able to build the program. Firstly, the program needs to be written either on the notepad or directly on the new file of the MPLAB IDE. The program needs to be saved in the .asm file format. After the program is successfully been written down, the build all action need to be taken. The reason of building a program is to seek for error. Error could be in syntax error. After building the program using the MPLAB, the program needs to be assembled if no errors occurred. After assembling the program, the file of .asm file format would turn/change to .hex file format. The .hex file format would then be imported to the MELAB programmer software.

3.12.3 Melab Programmer

The MELAB Programmer is the software needed to burn the program onto the PIC. After opening the MELAB Programmer make new project (refer to figure 7 in the appendices). Then the saved .hex files format need to be opened. After opening the file, set the PIC version used on the MELAB programmer. This is done to recognize which PIC are used to be burned. Lastly, click on the program, this action would burn the PIC and verify it.

3.12.4 Burning Program Process

The burning process is simple, only click on the program tool bar (refer appendix J). Then click program, then the LED light on the burner (refer appendix K) will turn on for a few second until the program is successfully burn into the PIC. The burning process can be repeatedly burn without erasing the existing program. This is because PIC uses flash memory.

3.13 Flowchart

It is important to draw a flow chart for the process of a program. A flow chart is so important because it will be a guide for the programmer to easily write the right flow of the program. If a program is made without the guide of a flow chart, it will be a non-systematic program or error might occur.

For the RCCS program flow chart, it has to follow the right order or step of the program to make the system work as planned. (Refer figure 3.13 (a)).

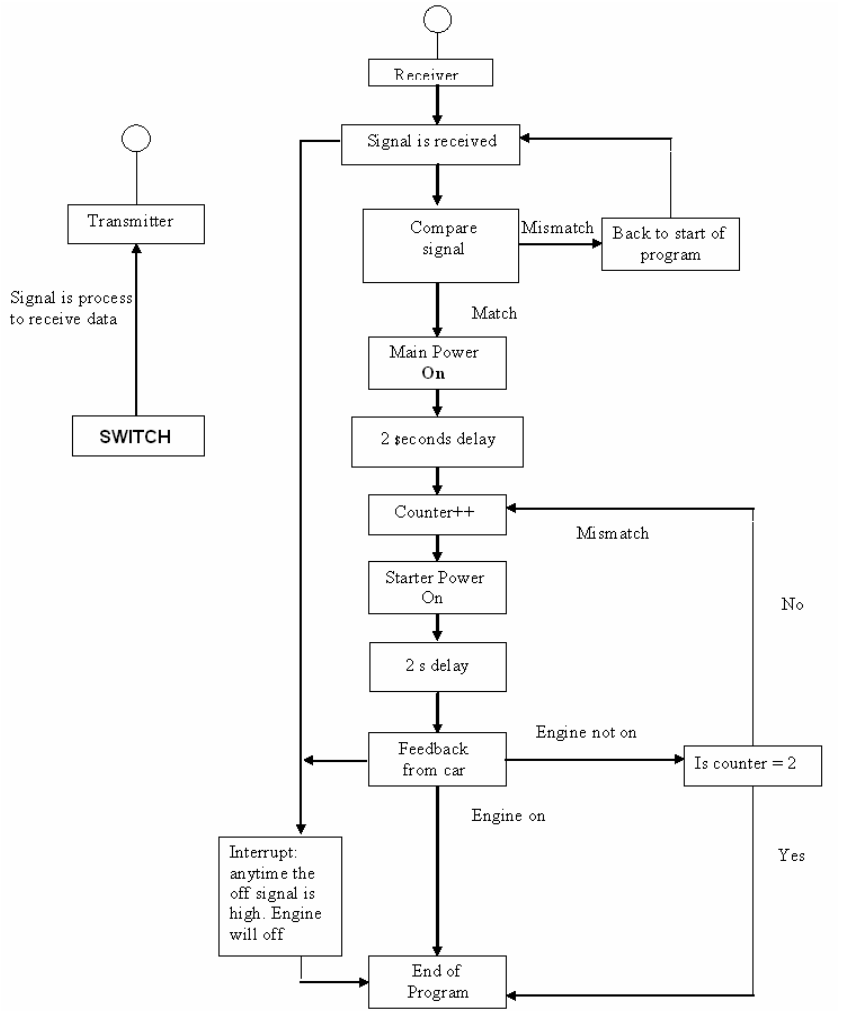


FIGURE 3.13 (a) Flowchart

The program that is written down will be burn in the PIC in hex file format. In order to burn the hex file into the microcontroller, the PIC16F84 programming board was used for this purpose. This programming device, programmer with accessory software installed on PC is in charge of writing the physical contents of hex file into the internal memory of a microcontroller. The programming board has a socket for inserting the PIC chips and connecting it to the computer for programming using a DB25 cable. The on-board flash memory can endure a minimum of 1000 erase/write cycles, so previous program does not have to be erased before the new program burn into the PIC. The programmer used is Melab programmer.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Background

This chapter will focus on the results and discussion of the RCCS project. The results will be explain in tables and discussion will be discussed due to project implementation

4.2 Results

Figure 4.3 show the table of result obtained from the RCCS system. The result is tested on the relay output. It is important to know the initial condition of the relay output. After the switch A is turned on, the relay A and relay B will energize and relays output will be turn to normally open. As the normally open for relay A and relay B is connected to normally open, it will turn on the accessories switch and Main Switch at the car connection.

If button B is pressed, it will energize relay C and the relay will toggle to normally open. As the relay toggle to normally open, it will turn on the ignition or the starter. It will start for two seconds and wait for the feedback from the car to tell the PIC weather the engine is started or not. If the engine is started, it will toggle the relay D to normally close. It will hold this position while the engine is running this is for the safety of the engine to avoid from crank. The relay output of relay C and

relay D is connected serially. If the engine is not started at the first trial, it will repeat for the second time trial to start the engine. If the car also not started for the second trial the program will end.

When button C is pressed, it will energize the relay E and it will toggle to normally close. As it toggle to normally close, it will turn on the air conditioner.

Table 4.3 RCCS Results

INPUT	**RELAY**	**RELAY Output**	**Result**
Button A	RELAY A	N/O	ON ACC
Button A	RELAY B	N/O	ON M/S
Button B	RELAY C	N/O	ON STARTER
Car Feedback	RELAY D	N/C	OFF STARTER
Button C	RELAY E	N/C	ON AIR COND.

4.3 Discussion

During research on the existing car wiring, the difficulty occurs on the assembling process of the project into car. This is due to complex car wiring and controller box on existing car wiring. Most modern vehicles wiring are very sensitive. Thus, the wiring needs to be done properly on the sensitive part such as the Computer Box of the vehicles. The wiring must not touch the existing wiring connected to the computer box. The wiring that is connected must be done out of the computer box to prevent from breakdown of the car existing wiring. The problem may be solved by assembling the RCCS to the car by exploring the functionality of each connection considered on the switches. The main switches of the car functionality must be check by the multi meter. The positive and negative terminals of each terminal that need to be connected have to be check before connecting the wires. This has been stated in chapter 3 on how the wiring should be done.

The size of wires that need to be connected to the car wiring also needs to be considered. This is due to power losses that occur during the current flowing process. A 3 ampere maximum current rating wires are needed in this project. The maximum current that will flow through the connection of the starter switch is 1.2 ampere. It is possible for not to use the wires that the current rating below than 3 ampere. It will cause large power losses and produce excessive heat to the wire connection. If this continuous, it is possible that the wire will burn out and cause breakdown to the system. Both RCCS and existing car wiring will be affected due to the power losses.

CHAPTER 5

CONCLUSION AND SUGGESTION

5.1 Background

This chapter will focus on the conclusion of the RCCS project. It also discuss on the suggestion that are needed for future development of this project.

5.2 Conclusion

As a conclusion, this will be a project that can be used widely for the car owners. For this project all the objectives and scopes of this project are accomplished. The expected results are obtained. With this system installed on the user's car, the car owners can safe time and it will be an advantage for them to use this project for their car. It will also be comfortable for drivers and passengers. This device also could be affordable by most of the car owner because of the cheaper price than the one in the market. Even though this project needs a lot of effort and patient in completing every stage, it has achieved its goals to be installed in real car instead of simulation on the project board.

5.3 Suggestion

This system can be upgradeable to more better and complex device. For future design it is suggested to be more complex due to more functions can be added on to the system. More functions means that it can implement more channels on the device. In the future design the remote might be able to control the whole systems of the car. It may consist of the head light, the radio and many more that can be turned on using the remote control device.

The system may have to consider more on the safeties of the car. The solution for this problem is to add on an extra channel to operate the central lock of the car. As the car engine has been started the car owner may lock the car by pressing another channel switch to energize the central lock of the car. This can prevent from unwanted event to occur.

For future design, the system may be improved by adding on an indicator on the remote to indicate the current status of the car. The system will be more intelligent with an LCD indicator. It might increase the cost of the project but it will make the system better by having a two ways communication system.

The size of the project mainly for the system installed in the car also can be reduce by combining all the circuits which are the receiver module, the controller circuit and the relay circuit into a single PCB board plotted double layer using the PCB plotter. If this suggestion is done for future development of this project, it will reduce to half the size of this RCCS project.

The range of operation also can be extending to a longer distance range by using a better RF transmitter and RF receiver module. The main type of better module depends on the antenna used in the module. A better antenna provides better range of operation.

REFERENCES

WEBSITE:

1. Microchip, “Data Sheet for pic16F872,” December 2001, http://www.microchip.com/devices/16f87/index.htm.

2. Marshall Brian, “How microcontroller works,” January 1999, http://www.howstuffworks.com/microcontroller.htm.

3. Parts Shop, *PIC Microcontroller Programmer* http://www.ece.uiuc.edu/eshop/availablemodules/PIC/pic.htm.

4. The Crime stopper cool car starter http://www.crimestopper.com.

5. Microchip, “Data Sheet for pic16F84A,” December 2001, http://www.microchip.com/

6. Parts Shop, *PIC Microcontroller Programmer* http://www.ece.uiuc.edu/eshop/availablemodules/PIC/pic.htm

BOOKS:

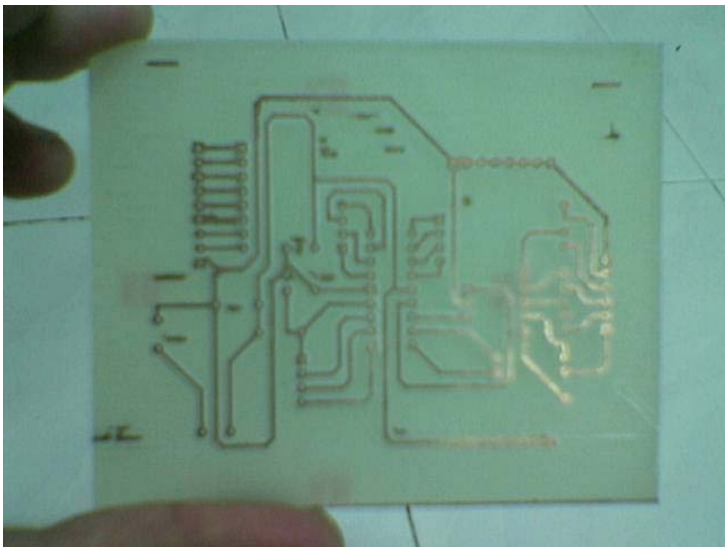
1.PIC Robotics Author: John Iovine

2.PIC Microcontroller Project Book Author: John Iovine

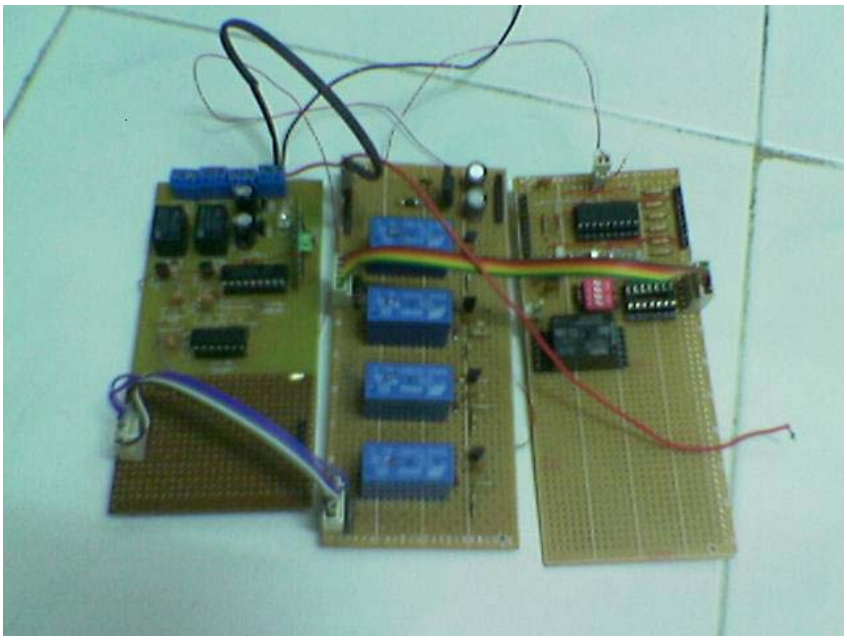
APPENDICES

APPENDIX A: RF Transmitter Module

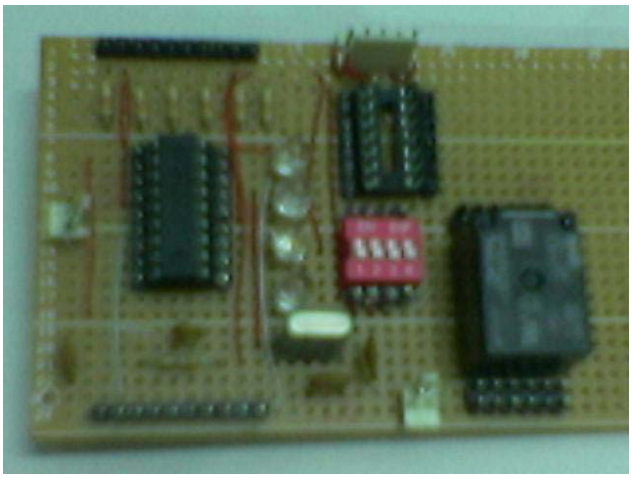
APPENDIX B : PCB Using Plotter



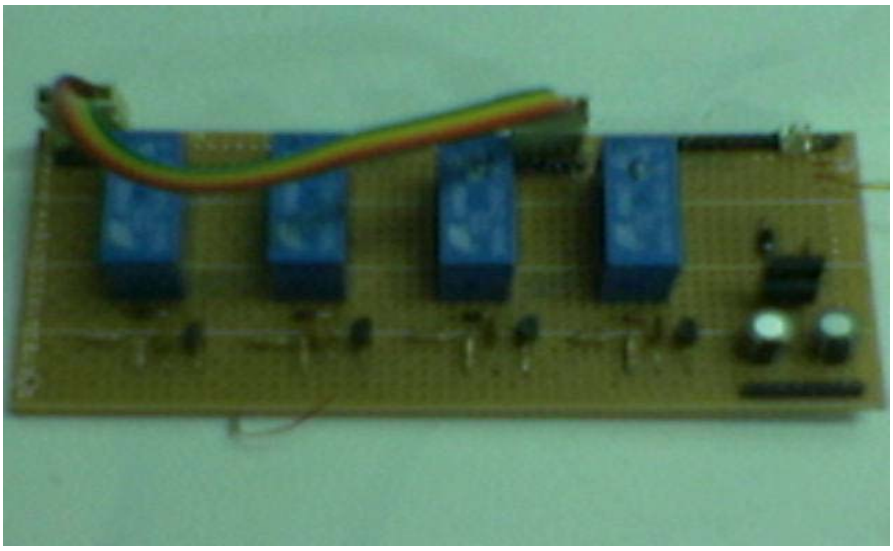
APPENDIX C : Combined Circuit



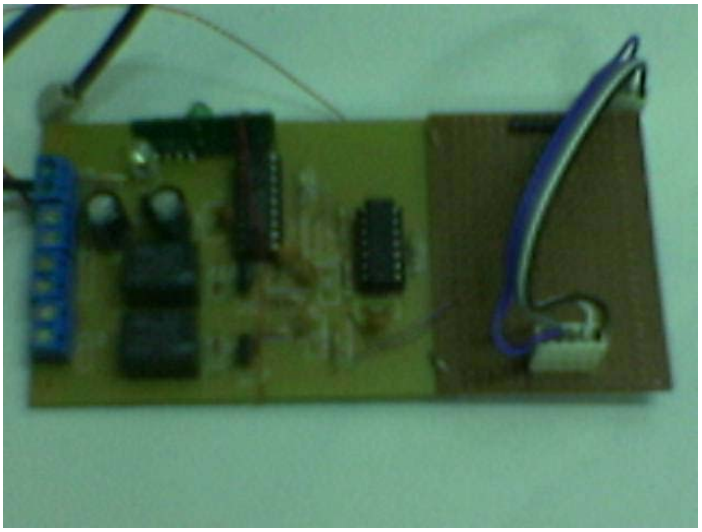
APPENDIX D : PIC Controller Circuit



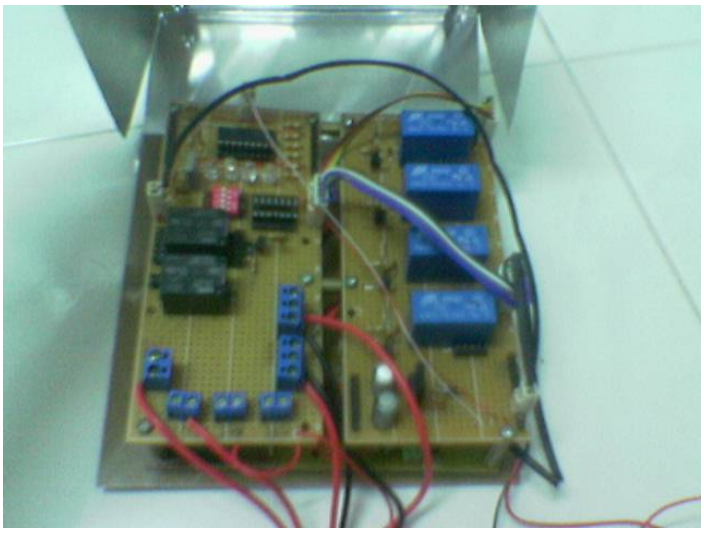
APPENDIX E : Relay Circuit Board



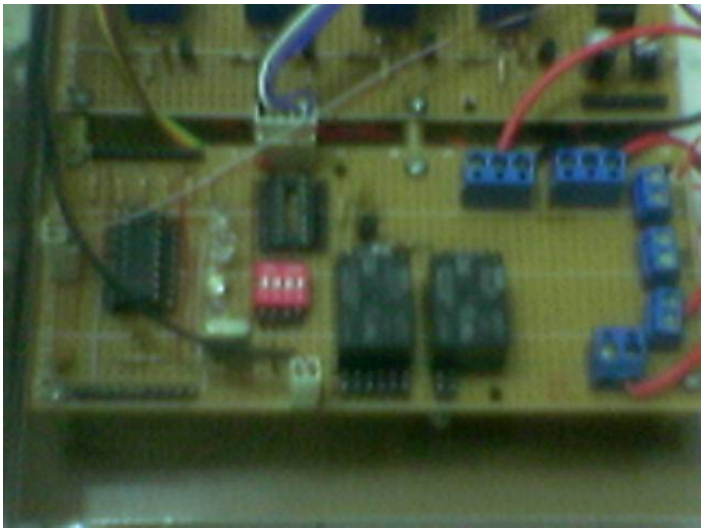
APPENDIX F : RF Receiver Module



APPENDIX G : RCCS Assembled

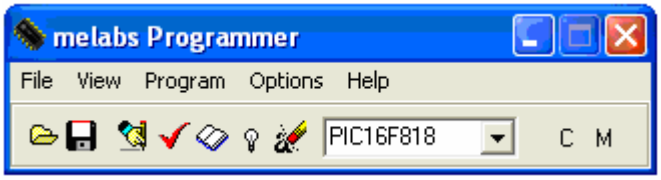


APPENDIX H : PIC Circuit on RCCS

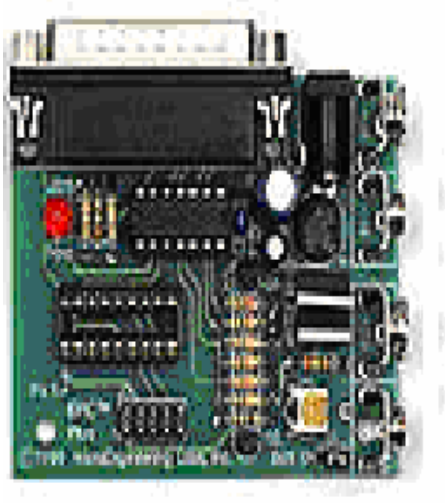


APPENDIX I : Complete RCCS

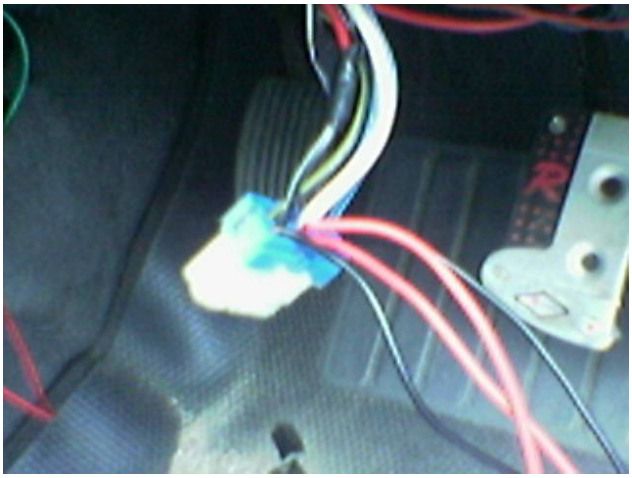
APPENDIX J : Melab Programmer



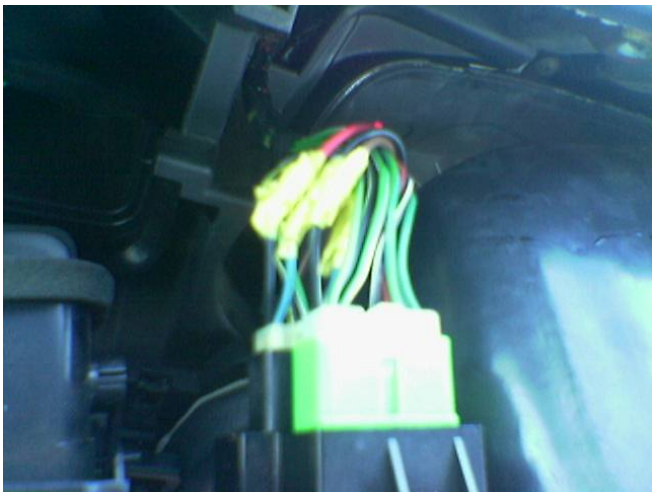
APPENDIX K : Melab PIC Burner



APPENDIX L : Main Wiring



APPENDIX M : Air Conditioner Wiring



APPENDIX N : Feedback Wiring

APPENDIX P : Main Program

```
;CarStarter.ASM  for 16F84A
;    This program provides a car starter
;    control sequence program.   It includes an input switch attached
;    to pin RA1.  The control sequence outputs are RB0 to RB3.
;    Pressing the switch will run the start the car
;    Pressing the switch for the second time will turn the starter off.
;    Speed of the sequence is controlled by the time delay subroutine.
;    The OPTION register is set to /256 to give timing
;    pulses of 1/32 of a second.
;*************************************************************
; EQUATES SECTION:
TMR0     EQU   1     ;  means TMR0 is file 1
STATUS   EQU   3     ; means STATUS is file 3
OPTION_REG EQU  0x81  ;  The option register is at file 0x81
PORTA    EQU   5     ; means PORTA is file 5
PORTB    EQU   6     ; means PORTB is file 6
TRISA    EQU   h'85'  ; means Port A direction register in Bank 1
TRISB    EQU   h'86'  ; means Port B direction register in Bank 1
ZEROBIT  EQU   2     ; means ZEROBIT is bit 2
COUNT    EQU   30H    ;  means COUNT is file 30 a register to count events
COUNT2        EQU   32H      ;  means COUNT is file 32 a register to count2 events
;*************************************************************
     LIST P=16F84A
     ORG     0   ;  START ADDRESS IN MEMORY 0
     goto start

;************************************************
; Poll Subroutine:
; Subroutine to poll RBO/INT , if RBO/INT becomes a '1'
; then program will restart
POLL
```

```
        BTFSC       PORTA, 1            ; Poll for a high at RA1
        GOTO start                      ; If bit is set, then program will restart and all
ports
        RETURN                                  ; will be cleared
;***********************************************
;SUBROUTINE SECTION
delay   clrf   TMR0
        movf   TMR0,w
        sublw  .66
        retlw  0

delay2  clrf     TMR0
        loopb   movf   TMR0,w
        sublw   .99
        btfsc    PORTA,1
        goto     loopb
        return
        retlw    0
;***********************************************
;MAIN PROGRAM
start
        bsf      STATUS,5
        movlw  B'00011111'  ;  Port A is input
        movwf  TRISA
        movlw  B'00000000'  ;  Port B is output
        movwf  TRISB
        movlw  B'00000111'
        movwf  OPTION_REG
        bcf      STATUS,5
        clrf     PORTB
begin
        btfss    PORTA,0            ;bit clear RA0 in real simulation
        btfss    PORTA,1            ;bit clear RA1 in real simulation
```

```
        goto    STARTER
        goto    begin
;*************************************************
; Starter:
; This subroutine will turn on the starter for 2 seconds
STARTER
        BSF    PORTB,1      ; If we receive the correct signal, then output high on RB1/car_starter
        CALL  delay         ; We will let RB1 remain high for a period of time, 3 secs for starter
        BCF    PORTB,1      ; Set RB1 back to low
        CALL  delay2        ; We will let RA1 remain low for a period of time
TRY
        DECFSZ      COUNT       ; Decrement the counter and skip if zero
        DECF        COUNT       ; Will iterate twice before FAIL
        BTFSS       COUNT,2
        GOTO        start
        BTFSS       PORTA,1     ;Bit test to see if tachometer is high, RA2 will tell us if engine started
        BCF         PORTB,1
        GOTO TRY                ; If not high, decrement counter and try again
        RETURN
        END
```

APPENDIX Q : Instruction Set For PIC16F84A

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS						
ADDWF	f, d	Add W and f	1	00 0111 dfff ffff	C,DC,Z	1,2
ANDWF	f, d	AND W with f	1	00 0101 dfff ffff	Z	1,2
CLRF	f	Clear f	1	00 0001 lfff ffff	Z	2
CLRW	-	Clear W	1	00 0001 0xxx xxxx	Z	
COMF	f, d	Complement f	1	00 1001 dfff ffff	Z	1,2
DECF	f, d	Decrement f	1	00 0011 dfff ffff	Z	1,2
DECFSZ	f, d	Decrement f, Skip if 0	1 (2)	00 1011 dfff ffff		1,2,3
INCF	f, d	Increment f	1	00 1010 dfff ffff	Z	1,2
INCFSZ	f, d	Increment f, Skip if 0	1 (2)	00 1111 dfff ffff		1,2,3
IORWF	f, d	Inclusive OR W with f	1	00 0100 dfff ffff	Z	1,2
MOVF	f, d	Move f	1	00 1000 dfff ffff	Z	1,2
MOVWF	f	Move W to f	1	00 0000 lfff ffff		
NOP	-	No Operation	1	00 0000 0xx0 0000		
RLF	f, d	Rotate Left f through Carry	1	00 1101 dfff ffff	C	1,2
RRF	f, d	Rotate Right f through Carry	1	00 1100 dfff ffff	C	1,2
SUBWF	f, d	Subtract W from f	1	00 0010 dfff ffff	C,DC,Z	1,2
SWAPF	f, d	Swap nibbles in f	1	00 1110 dfff ffff		1,2
XORWF	f, d	Exclusive OR W with f	1	00 0110 dfff ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS						
BCF	f, b	Bit Clear f	1	01 00bb bfff ffff		1,2
BSF	f, b	Bit Set f	1	01 01bb bfff ffff		1,2
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01 10bb bfff ffff		3
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01 11bb bfff ffff		3
LITERAL AND CONTROL OPERATIONS						
ADDLW	k	Add literal and W	1	11 111x kkkk kkkk	C,DC,Z	
ANDLW	k	AND literal with W	1	11 1001 kkkk kkkk	Z	
CALL	k	Call subroutine	2	10 0kkk kkkk kkkk		
CLRWDT	-	Clear Watchdog Timer	1	00 0000 0110 0100	$\overline{TO}$,$\overline{PD}$	
GOTO	k	Go to address	2	10 1kkk kkkk kkkk		
IORLW	k	Inclusive OR literal with W	1	11 1000 kkkk kkkk	Z	
MOVLW	k	Move literal to W	1	11 00xx kkkk kkkk		
RETFIE	-	Return from interrupt	2	00 0000 0000 1001		
RETLW	k	Return with literal in W	2	11 01xx kkkk kkkk		
RETURN	-	Return from Subroutine	2	00 0000 0000 1000		
SLEEP	-	Go into standby mode	1	00 0000 0110 0011	$\overline{TO}$,$\overline{PD}$	
SUBLW	k	Subtract W from literal	1	11 110x kkkk kkkk	C,DC,Z	
XORLW	k	Exclusive OR literal with W	1	11 1010 kkkk kkkk	Z	

Note 1: When an I/O register is modified as a function of itself (e.g., `MOVF PORTB, 1`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned to the Timer0 Module.

3: If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

Note: Additional information on the mid-range instruction set is available in the PICmicro™ Mid-Range MCU Family Reference Manual (DS33023).

7.1 Instruction Descriptions

ADDLW	Add Literal and W
Syntax:	[*label*] ADDLW k
Operands:	0 ≤ k ≤ 255
Operation:	(W) + k → (W)
Status Affected:	C, DC, Z
Description:	The contents of the W register are added to the eight-bit literal 'k' and the result is placed in the W register.

ADDWF	Add W and f
Syntax:	[*label*] ADDWF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(W) + (f) → (destination)
Status Affected:	C, DC, Z
Description:	Add the contents of the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

ANDLW	AND Literal with W
Syntax:	[*label*] ANDLW k
Operands:	0 ≤ k ≤ 255
Operation:	(W) .AND. (k) → (W)
Status Affected:	Z
Description:	The contents of W register are AND'ed with the eight-bit literal 'k'. The result is placed in the W register.

ANDWF	AND W with f
Syntax:	[*label*] ANDWF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(W) .AND. (f) → (destination)
Status Affected:	Z
Description:	AND the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

BCF	Bit Clear f
Syntax:	[*label*] BCF f,b
Operands:	0 ≤ f ≤ 127 0 ≤ b ≤ 7
Operation:	0 → (f<b>)
Status Affected:	None
Description:	Bit 'b' in register 'f' is cleared.

BSF	Bit Set f
Syntax:	[*label*] BSF f,b
Operands:	0 ≤ f ≤ 127 0 ≤ b ≤ 7
Operation:	1 → (f<b>)
Status Affected:	None
Description:	Bit 'b' in register 'f' is set.

BTFSS	Bit Test f, Skip if Set
Syntax:	[*label*] BTFSS f,b
Operands:	0 ≤ f ≤ 127 0 ≤ b < 7
Operation:	skip if (f<b>) = 1
Status Affected:	None
Description:	If bit 'b' in register 'f' is '0', the next instruction is executed. If bit 'b' is '1', then the next instruction is discarded and a NOP is executed instead, making this a 2TCY instruction.

BTFSC	Bit Test, Skip if Clear
Syntax:	[*label*] BTFSC f,b
Operands:	0 ≤ f ≤ 127 0 ≤ b ≤ 7
Operation:	skip if (f<b>) = 0
Status Affected:	None
Description:	If bit 'b' in register 'f' is '1', the next instruction is executed. If bit 'b' in register 'f' is '0', the next instruction is discarded, and a NOP is executed instead, making this a 2TCY instruction.

CALL	Call Subroutine
Syntax:	[*label*] CALL k
Operands:	0 ≤ k ≤ 2047
Operation:	(PC)+ 1→ TOS, k → PC<10:0>, (PCLATH<4:3>) → PC<12:11>
Status Affected:	None
Description:	Call Subroutine. First, return address (PC+1) is pushed onto the stack. The eleven-bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a two-cycle instruction.

CLRF	Clear f
Syntax:	[*label*] CLRF f
Operands:	0 ≤ f ≤ 127
Operation:	00h → (f) 1 → Z
Status Affected:	Z
Description:	The contents of register 'f' are cleared and the Z bit is set.

CLRW	Clear W
Syntax:	[*label*] CLRW
Operands:	None
Operation:	00h → (W) 1 → Z
Status Affected:	Z
Description:	W register is cleared. Zero bit (Z) is set.

CLRWDT	Clear Watchdog Timer
Syntax:	[*label*] CLRWDT
Operands:	None
Operation:	00h → WDT 0 → WDT prescaler, 1 → $\overline{TO}$ 1 → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$
Description:	CLRWDT instruction resets the Watchdog Timer. It also resets the prescaler of the WDT. Status bits $\overline{TO}$ and $\overline{PD}$ are set.

COMF	Complement f
Syntax:	[*label*] COMF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	($\bar{f}$) → (destination)
Status Affected:	Z
Description:	The contents of register 'f' are complemented. If 'd' is 0, the result is stored in W. If 'd' is 1, the result is stored back in register 'f'.

DECF	Decrement f
Syntax:	[*label*] DECF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f) - 1 → (destination)
Status Affected:	Z
Description:	Decrement register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

DECFSZ	Decrement f, Skip if 0
Syntax:	[*label*] DECFSZ f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f) - 1 → (destination); skip if result = 0
Status Affected:	None
Description:	The contents of register 'f' are decremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'. If the result is 1, the next instruction is executed. If the result is 0, then a NOP is executed instead, making it a 2TCY instruction.

GOTO	Unconditional Branch
Syntax:	[*label*] GOTO k
Operands:	0 ≤ k ≤ 2047
Operation:	k → PC<10:0> PCLATH<4:3> → PC<12:11>
Status Affected:	None
Description:	GOTO is an unconditional branch. The eleven-bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. GOTO is a two-cycle instruction.

INCF	Increment f
Syntax:	[*label*] INCF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f) + 1 → (destination)
Status Affected:	Z
Description:	The contents of register 'f' are incremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.

INCFSZ	Increment f, Skip if 0
Syntax:	[*label*] INCFSZ f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f) + 1 → (destination), skip if result = 0
Status Affected:	None
Description:	The contents of register 'f' are incremented. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'. If the result is 1, the next instruction is executed. If the result is 0, a NOP is executed instead, making it a 2TCY instruction.

IORLW	Inclusive OR Literal with W
Syntax:	[*label*] IORLW k
Operands:	0 ≤ k ≤ 255
Operation:	(W) .OR. k → (W)
Status Affected:	Z
Description:	The contents of the W register are OR'ed with the eight-bit literal 'k'. The result is placed in the W register.

IORWF	Inclusive OR W with f
Syntax:	[*label*] IORWF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(W) .OR. (f) → (destination)
Status Affected:	Z
Description:	Inclusive OR the W register with register 'f'. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.

MOVF	Move f
Syntax:	[*label*] MOVF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f) → (destination)
Status Affected:	Z
Description:	The contents of register f are moved to a destination dependant upon the status of d. If d = 0, destination is W register. If d = 1, the destination is file register f itself. d = 1 is useful to test a file register, since status flag Z is affected.

MOVLW	Move Literal to W
Syntax:	[*label*] MOVLW k
Operands:	0 ≤ k ≤ 255
Operation:	k → (W)
Status Affected:	None
Description:	The eight-bit literal 'k' is loaded into W register. The don't cares will assemble as 0's.

MOVWF	Move W to f
Syntax:	[*label*] MOVWF f
Operands:	0 ≤ f ≤ 127
Operation:	(W) → (f)
Status Affected:	None
Description:	Move data from W register to register 'f'.

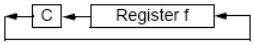
NOP	No Operation
Syntax:	[*label*] NOP
Operands:	None
Operation:	No operation
Status Affected:	None
Description:	No operation.

RETFIE	Return from Interrupt
Syntax:	[*label*] RETFIE
Operands:	None
Operation:	TOS → PC, 1 → GIE
Status Affected:	None

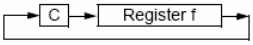
RETLW	Return with Literal in W
Syntax:	[*label*] RETLW k
Operands:	0 ≤ k ≤ 255
Operation:	k → (W); TOS → PC
Status Affected:	None
Description:	The W register is loaded with the eight-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction.

RETURN	Return from Subroutine
Syntax:	[*label*] RETURN
Operands:	None
Operation:	TOS → PC
Status Affected:	None
Description:	Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a two-cycle instruction.

RLF	Rotate Left f through Carry
Syntax:	[*label*] RLF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	See description below
Status Affected:	C
Description:	The contents of register 'f' are rotated one bit to the left through the Carry Flag. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is stored back in register 'f'.



RRF	Rotate Right f through Carry
Syntax:	[*label*] RRF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	See description below
Status Affected:	C
Description:	The contents of register 'f' are rotated one bit to the right through the Carry Flag. If 'd' is 0, the result is placed in the W register. If 'd' is 1, the result is placed back in register 'f'.



SLEEP	
Syntax:	[*label*] SLEEP
Operands:	None
Operation:	00h → WDT, 0 → WDT prescaler, 1 → $\overline{TO}$, 0 → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$
Description:	The power-down status bit, $\overline{PD}$ is cleared. Time-out status bit, $\overline{TO}$ is set. Watchdog Timer and its prescaler are cleared. The processor is put into SLEEP mode with the oscillator stopped.

SUBLW	Subtract W from Literal
Syntax:	[*label*] SUBLW k
Operands:	0 ≤ k ≤ 255
Operation:	k - (W) → (W)
Status Affected:	C, DC, Z
Description:	The W register is subtracted (2's complement method) from the eight-bit literal 'k'. The result is placed in the W register.

SUBWF	Subtract W from f
Syntax:	[*label*] SUBWF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f) - (W) → (destination)
Status Affected:	C, DC, Z
Description:	Subtract (2's complement method) W register from register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.

SWAPF	Swap Nibbles in f
Syntax:	[*label*] SWAPF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f<3:0>) → (destination<7:4>), (f<7:4>) → (destination<3:0>)
Status Affected:	None
Description:	The upper and lower nibbles of register 'f' are exchanged. If 'd' is 0, the result is placed in W register. If 'd' is 1, the result is placed in register 'f'.

XORLW	Exclusive OR Literal with W
Syntax:	[*label*] XORLW k
Operands:	$0 \le k \le 255$
Operation:	(W) .XOR. k → (W)
Status Affected:	Z
Description:	The contents of the W register are XOR'ed with the eight-bit literal 'k'. The result is placed in the W register.

XORWF	Exclusive OR W with f
Syntax:	[*label*] XORWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(W) .XOR. (f) → (destination)
Status Affected:	Z
Description:	Exclusive OR the contents of the W register with register 'f'. If 'd' is 0, the result is stored in the W register. If 'd' is 1, the result is stored back in register 'f'.