Electronic Throttle Control

by

Ivan Ng Heng Guan

Dissertation submitted in partial fulfillment of
the requirements for the
Bachelor of Engineering (Hons)
(Electrical & Electronics)

JUNE 2009

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

Electronic Throttle Control

by

Ivan Ng Heng Guan

A project dissertation submitted to the
Electrical & Electronics Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfillment of the requirement for the
BACHELOR OF ENGINEERING (Hons)
(ELECTRICAL & ELECTRONICS ENGINEERING)

Approved by,

(Assoc. Prof. Dr. Mohd Noh Bin Karsiti)
Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

JUNE 2009

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

IVAN NG HENG GUAN

ABSTRACT

Electronic Throttle Control (ETC) is becoming an important part of automotive industry today. Also known as the Drive-By-Wire technology, applied to our daily life, it can result in an efficient throttle body controlling improving fuel efficiency, handling and ride comfort in an automobile. The objective of the project is to understand where the future of electronic control in the automotive industry is going due to advances in technology. The main focus of the project is focused on controlling the position of the throttle opening also known as the "butterfly". For this purpose, a remote controlled (RC) servomotor was used for positioning the "butterfly" opening. The RC servomotor will be controlled by a PIC microcontroller with the fastest response it can achieve. For error detection purpose, a feedback signal will be sent back to the microcontroller via a throttle positioning sensor (TPS) to be analyzed. With existing feedback data, the microcontroller would be able to correct the error significantly.

ACKNOWLEDGEMENTS

Foremost, I would like to express my sincere gratitude to my project supervisor Assoc. Prof. Dr. Mohd Noh Karsiti for the continuous support of my final year project, for his patience, motivation, enthusiasm and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

I also thank my fellow coursemates and technician in Universiti Teknologi PETRONAS who helped me whenever they can. Without them, I would not been inspired doing my work.

Last but not the least, I would like to thank my family and friends for giving me the moral support, supporting me spiritually throughout my life.

TABLE OF CONTENTS

LIST OF FIGURE

LIST OF TABLE

LIST OF ABBREVIATIONS

ETC	Electronic Throttle Control
ECM	Electronic Control Module
ECU	Engine Control Unit
RC	Remote Control
TPS	Throttle Position Sensor
WOT	Wide-Open Throttle
UTP	Universiti Teknologi Petronas
EE	Electrical & Electronics Engineering
PIC	Programmable Interface Computer
ADC	Analogue-to-Digital Converter
SPDT	Single-Pole, Double-Throw
LED	Light Emitting Diode
PWM	Pulse-Width-Modulation
PPS	Pedal Position Sensor

CHAPTER 1
INTRODUCTION

1.1 Background of Study

Electronic Throttle Control (ETC) has been around for the past 25 years. Also known as the "Drive-By-Wire" system, it was first introduced by the British on their military aircraft in the 60s called the "Fly-By-Wire" [1]. Nowadays, most high-end cars use this system to enhance driver's experience which is to reduce fuel consumption by controlling the amount of air-to-fuel ratio into the engine. Since there is also ruling in emission gases nowadays, electronic throttle control is a way to ensure that harmful exhaust emissions are kept to absolute minimum and drivability is maintained, regardless of any circumstances [2].

Throttle control has been used widely not only in the automotive industry, but also in industry like aviation, machinery and hobbyist product. In the automotive industry, it is traditionally accomplished by a physical connection between the throttle pedal and the throttle body which controls the engine airflow. This project is important to understand where the future of electronic control is going due to advances in technology and the need for all parts of the automobile to communicate with each other. Electronic control in automotive allows engine, transmission, brakes, just to name a few, to work together to achieve a much faster response to the driver's needs. With this communication system, the electronic components could work closely with the mechanical components to achieve a bright result.

In a drive-by-wire car, a microcontroller determines the correct throttle plate position. If the driver needs sudden acceleration and steps on the accelerator pedal, the sensor on the pedal transmits the driver's pressure on the pedal to the microcontroller, which calculates and relays the correct throttle position to the motor, which moves the butterfly opening accordingly. Electronic control of the butterfly opening offers improved fuel economy and emissions by maintaining optimal throttle conditions at all times, which is something human driver cannot do.

Since this project deals with throttle bodies, it is important to know about throttle bodies and how they work. The functions of the vacuum connectors on throttle bodies are also studied.

The idea of this project is to control a throttle electronically, therefore it is important to know about engines and motors and how they actually works [3]. RC servo was chosen due to its close-loop control system where feedback from the output can be used for error detection. Compared to a stepper motor, it is very suitable for position control where the control movements are much smoother [4]. Programmed to a programming interface chip (PIC) with more than 1000 resolution control, the servomotor should control the "butterfly" opening of the throttle precisely.

1.2 Problem Statement

1.2.1 Problem Identification

Since most of us demand for a car which is fuel efficient and meets gas emission standards, car manufacturer had been finding ways to built them. There are many attempts to solve this issue, for example, building smaller cars, hybrid cars or even lighter cars. Through my experience, a car could also waste more energy depending on the driver's driving lifestyle. Since it is difficult to change a driver's habit in driving, Electronic Throttle Control (ETC) also known as Drive-by-Wire technology was introduced. The concept of this is to control the "butterfly" opening on a throttle to maximize the percentage of combustion in the engine. Since most car throttle communicates with the Engine Control Unit (ECU), the angle of the "butterfly" opening could be important information for the ECU for better efficiency and response.

1.2.2 Significant of The Project

In fuel injection systems, when the air-to-fuel ratio is poor, the amount of fuel going into the engine will exceed the amount of air required for a complete combustion [5]. When cases like this happen, incomplete combustion will occur causing wastage of fuel supplied. This would not only effect the poor emission of the vehicle, but also effect the poor fuel consumption and performance of the vehicle. Due to this weakness in mechanical throttles, Electronic Throttle Control (ETC) is introduced to eliminate such cases. With the flow controlled into the engine monitored precisely, the consumption and performance of a vehicle could be monitored closely.

1.3 Objective and Scope of Study

1.3.1 Relevance of The Project

The main objective of this project is to build a functional Electronic Throttle Control system and study its control element in the process. This will be accomplished by building a prototype model of an ETC monitored with an aid of a throttle response simulator built.

1.3.2 Feasibility of The Project Within Scope and Time Frame

The project work involves simulation and control. It has to be completed within two final semesters, 28 weeks. The first semester covers mostly literature reviews of written materials on the topic and a simulation. While in the second semester, a prototype model is constructed. The allocated time frame is reasonable.

CHAPTER 2
LITERATURE REVIEW & THEORY

2.1 Theory

Getting started with a project, it is important to conduct research to understand about the project more. This project started by researching on the weaknesses faced by mechanically driven throttles and what has been done to solve the issues. Knowing the objective of this project, the throttle body of an engine was studied closely. With condition which would affect a throttle's response, a simulator was modeled to study its response towards its physical condition. This was important for us to build a throttle which is responsive. Research was done on the throttle body on how it controls the air intake into an engine at desired condition.

Since Electronic Throttle Control (ETC) has been around, research was also done on it especially on why a DC servomotor is chosen over other motors. Due to this, focus was moved to understanding how a motor works and how to control them. This was important to determine the type of motor which is best used for this project. For example, its torque rating or rotating speed which was important to obtain a quick and stable response. Finalizing this, a control board will be designed to control the motor which controls the opening of the throttle.

2.2 Electronic Throttle Body

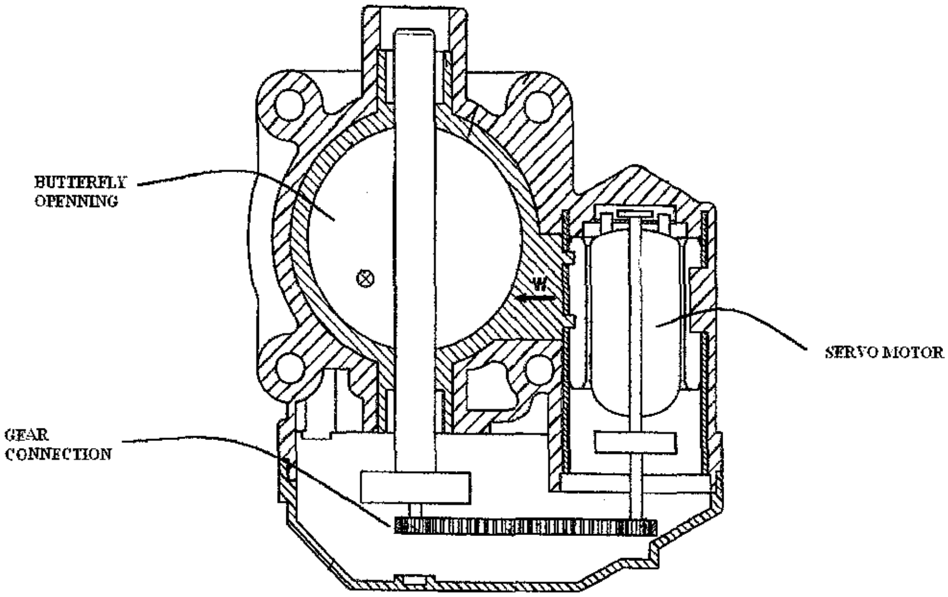


Figure 1

Figure 1: Electronic Throttle Body

A schematic diagram of an electronic throttle body is as shown in Figure 1. Basically, the butterfly opening is similar to the mechanically driven throttle body. Difference is the butterfly opening is driven by a servomotor connecting by a set of gears while in mechanically driven throttles, the butterfly opening is controlled by a cable connected directly to the butterfly opening [5]. The electronic throttle body is connected to an ETC module which controls the butterfly opening angle depending on how the driver controls the throttle pedal. Figure 2 shows an electronic throttle system diagram adapted from Robert Bosch [7].

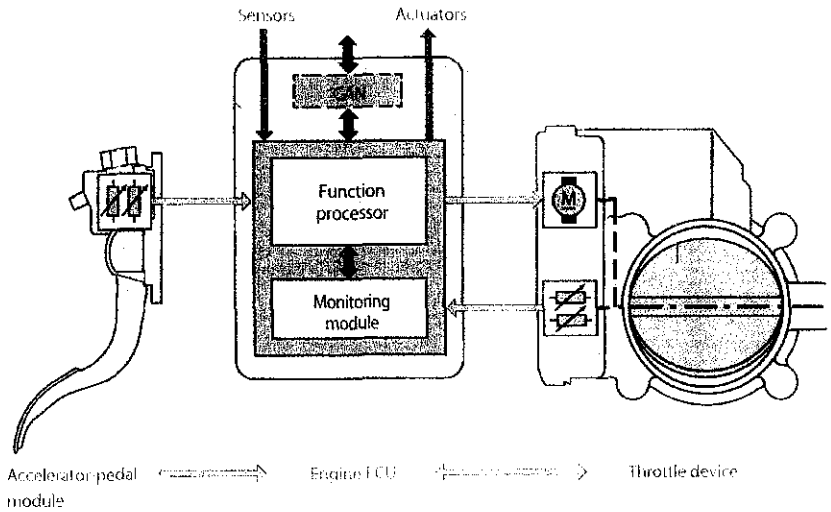


Figure 2: Electronic Throttle System

When a driver presses the throttle pedal, the throttle position sensor will send a signal to the ETC module. The ETC module will control the opening angle of the butterfly opening depending on the amount of pressure exerted on the throttle pedal. The Throttle Position Sensor (TPS) will provide a feedback to the ETC module. This is a feedback loop for error correction on the position of the butterfly opening.

2.3 Case Study on Vacuum Outlet from Throttle Bodies

A study was conducted on the vacuum outlet of throttle bodies to ensure any modification done on a throttle body would not affect its output. Normally, the vacuum outlet from a throttle body are connected to the power brakes, air-condition, wipers and etc. For the Daihatsu Detomaso throttle body which was bought, the vacuum outlet are connected to the power brakes, Electronic Control Unit (ECU) and air condition. The vacuum outlets are as the figure 2.3.

Figure 3: Daihatsu Detomaso Vacuum outlets

2.4 Open-Loop Control versus Close-Loop Control

Servomotors were chosen as a driver in Electronic Throttle Controls (ETC) because it can be controlled in a closed loop function. A simple closed-loop and open-loop control system is as shown as below:

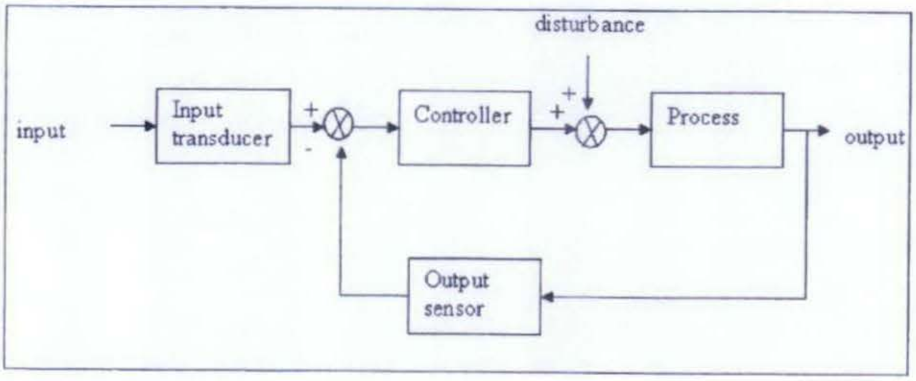


Figure 4: Closed-Loop System

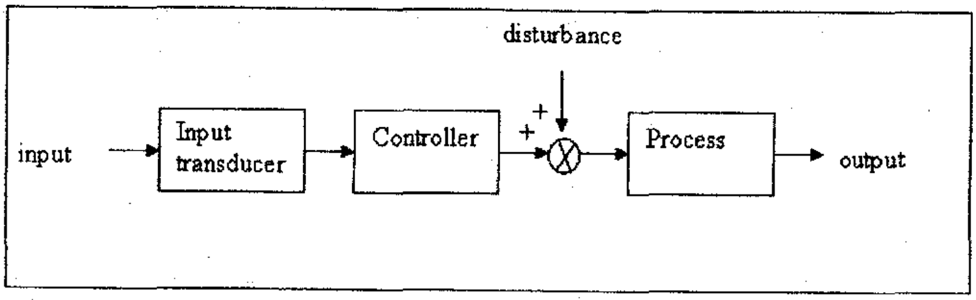


Figure 5: Open-Loop System

Comparing the closed-loop and open-loop system above, the main problem with open loop systems is its inability to detect error [4]. Looking at the close-loop control system, the output signal is feed back into the input to eliminate any error that may occur. Therefore, it is more stable to use a close-loop control system than a open-loop control system because its feedback signal will corrects any error that may occur.

2.5 Throttle Position Sensor (TPS)

A throttle position sensor (TPS) is a sensor is used to monitor the position of the throttle in an engine. The sensor is located on the butterfly spindle to directly monitor the position of the throttle valve "butterfly".

The sensor usually consist of a potentiometer [11], as shown in figure 6, where it provides a variable resistance dependent upon the position of the valve ("butterfly position). It converts the throttle valve angle into an electric signal. As the throttle opens, the signal voltage increases. A graph of resistance over time of a throttle position sensor is as shown in figure 7.

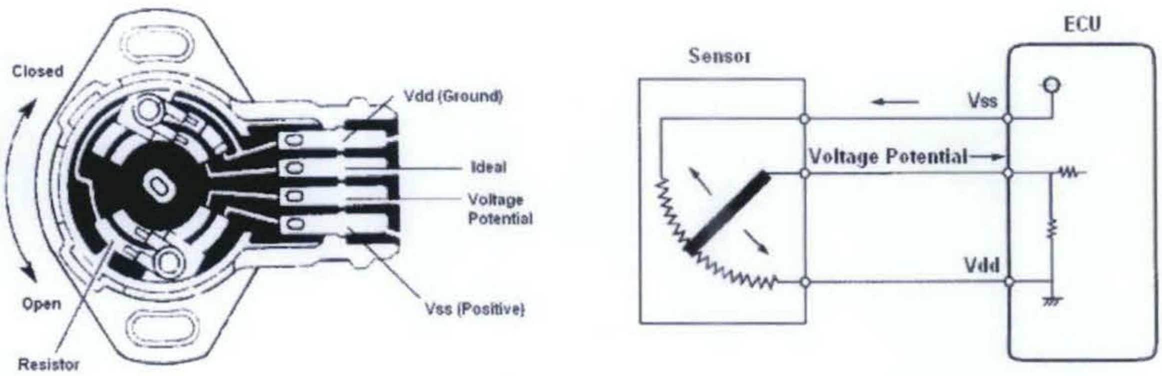


Figure 6 : Throttle Position Sensor Circuit

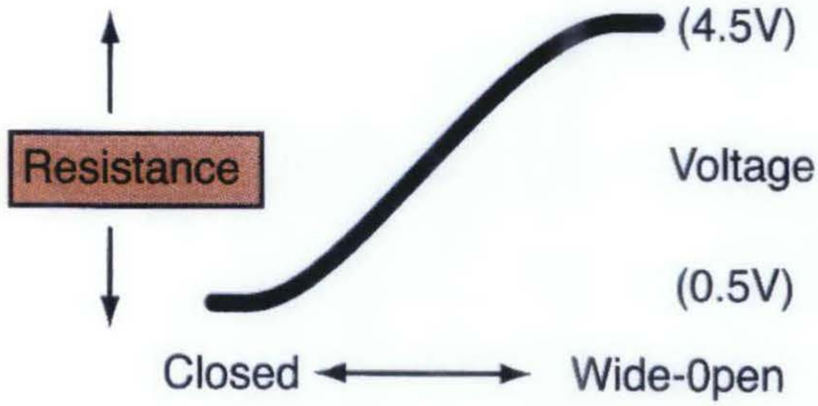


Figure 7 : Graph of Resistance over time

A typical TPS has 1000 ohm with the throttle in the idle position, and 4000ohm at wide open throttle (WOT). The voltage signal from a typical TPS is 0.5 volts to 1 volt at idle and 4.5 volts at WOT, and this signal informs the computer regarding the exact throttle position.

The sensor signal is useful for the engine control unit (ECU) as an input to its control system. The ignition timing and fuel injection timing are altered depending upon the position of the throttle, and also depending on the rate of change of that position.

For example, in fuel injection engines, in order to avoid stalling, extra fuel may be injected if the throttle is opened rapidly.

2.6 RC Servo Motor

Remote Controlled (RC) servo motor are designed to receive a position command and move its motor shaft to the commanded position [8]. They are made from DC motor with potentiometer and servo controller chip built to them to determine the angle of its motor shaft and receive commanded position and drive the commanded position in a feedback loop. A typical internal construction of the Sanwa servo with its component list is as shown in **APPENDIX III.**

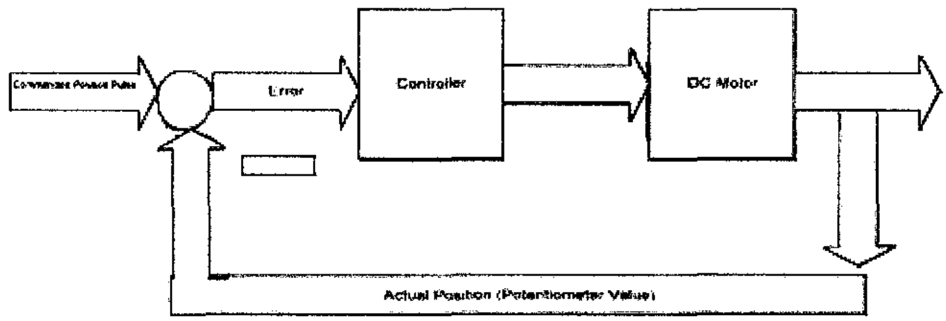


Figure 8: RC Servo Block Diagram

An RC servo has limited motion, where it can only rotate between -90 to +90 degree. To protect it from over rotating, it has a mechanical stop to ensure the motion is restricted. The figure below shows a servo consist of a DC motor with 2 gears connected to a potential meter for feedback and some control circuitry.

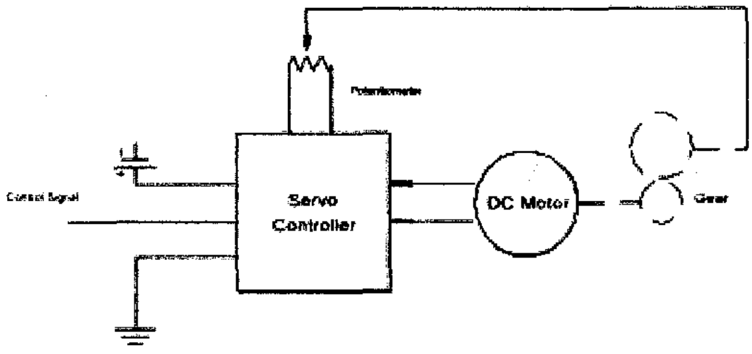


Figure 9: RC servo functional block diagram

There are 3 inputs for a RC servo, one for power supply (5V), another for ground, and the third for giving control signals. Usually, a 1.5ms pulse width would indicate 0-degree position (neutral) of the motor shaft. A pulse width of 1ms indicates -90 degrees and a pulse width of 2ms would indicate +90-degree [8]. A sample of motion is as shown in figure 9.

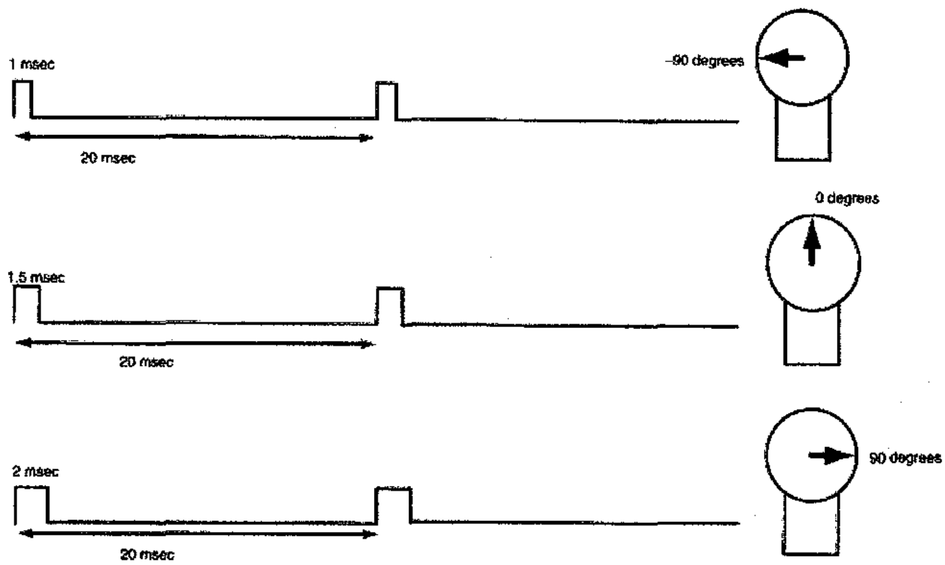


Figure 10: RC Servo control signal

CHAPTER 3
METHODOLOGY

3.1 Research Methodology

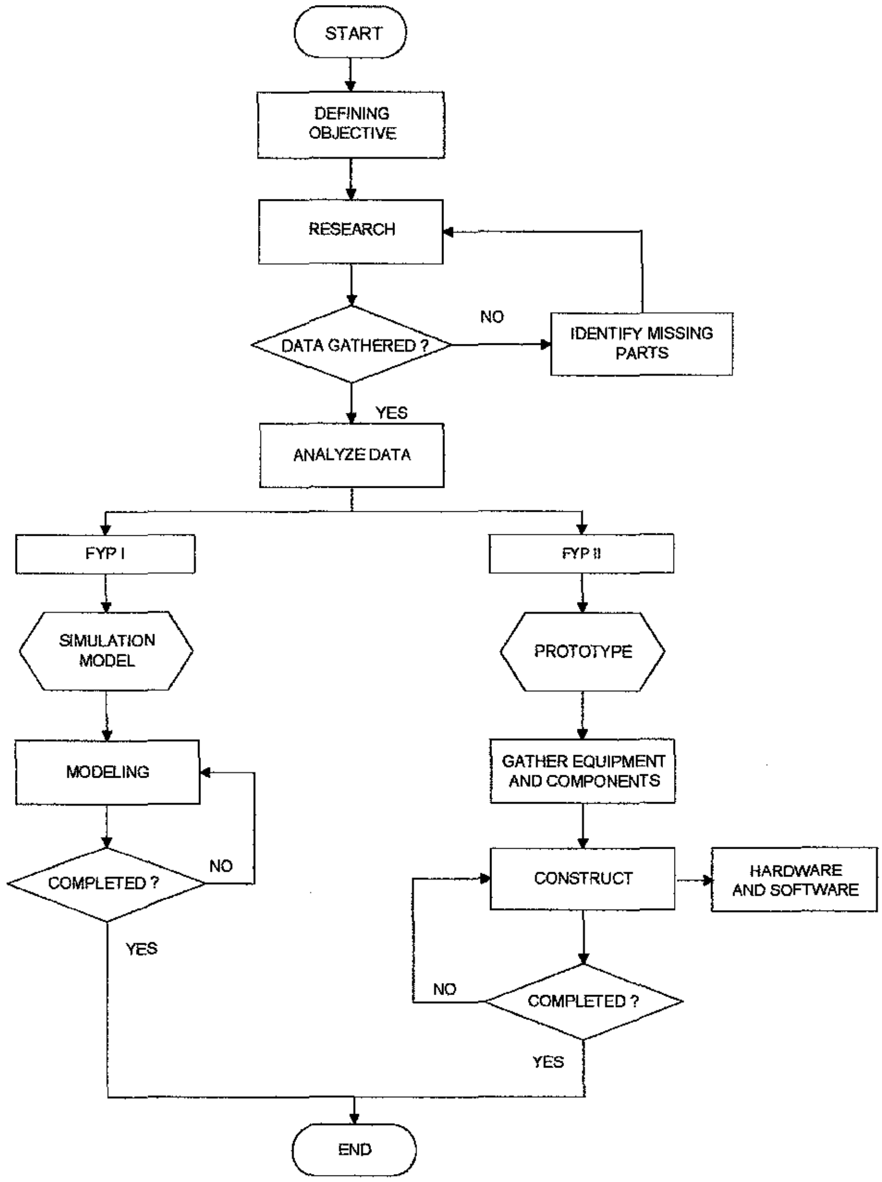


Figure 11: Process Flow Diagram

During the first 14 weeks, final year project 1 course, research was conducted on what Electronic Throttle Control (ETC) was all about. The main purpose of this is to identify the problem statement of this project and solutions to overcome it. Knowing the problem faced, research was done on DC motors especially which type of DC motor which will be suitable for this project. Since throttle bodies contribute vacuum effects in automotive technology, the effects were also eyed on. By the end of the semester, a simulator was modeled for this project to simulate the response of the throttle under certain physical conditions. There are 3 main mathematical model for the simulation, ETC module, throttle and throttle position sensor. The model is as shown below,

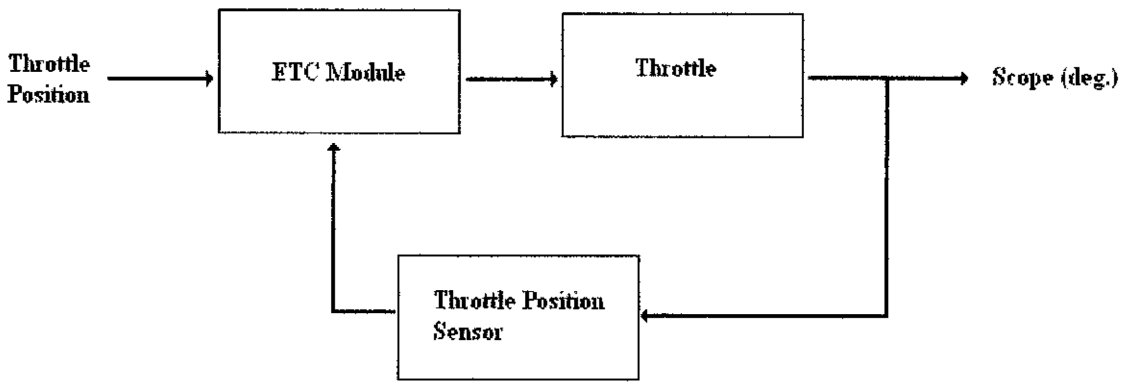


Figure 12: Simple model

During the second semester (final year project 2) a prototype of an Electronic Throttle Control (ETC) was built. A pair of gear sets was first modeled linking the servomotor and the throttle opening shaft. A controller was later built using a 16F84A microprocessor controlled by a single-pole, double-throw (SPDT) switch. Since the project requires a potentiometer as a control element to control the RC Servo, a 16F684 microcontroller was used replacing the 16F84 microcontroller. C Programming was the programming language used to program the microcontroller. For error detection purpose, the throttle position sensor (TPS) is used to provide a feedback to the microcontroller.

3.2 MATLAB Simulink

Due to its simulation capabilities, MATLAB was chosen as the simulation software to be used in this project. MATLAB Simulink was used to model a simulator to simulate the response of a throttle under certain conditions. For example for the current mechanical throttle which was bought, the conditions are stiffness of the spring, k, inertia movement of air-flow, J, and damping effect, c. Considering all this conditions, the response of the throttle will be studied. A model of the throttle can be seen in **APPENDIX IV**.

With the optimal performance produced, the physical values of the throttle will be applied when the prototype is built. This is to ensure a working throttle with great efficiency can be built.

3.3 Gear Set

A gear set was used to link between the RC Servomotor and the throttle plate shaft. A picture of the gear set is as shown in figure below.

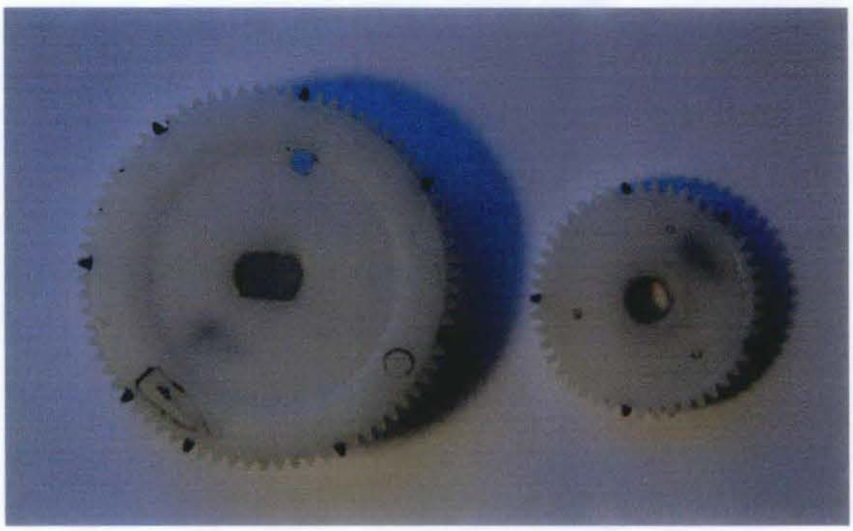


Figure 13: Gear Set

Looking at figure 13, the gear on the left of the picture was built to be connected to the throttle body shaft while the gear on the right is built to be connected to the RC servomotor.

3.4 PIC Microcontrollers

Programmable Interface Computer (PIC) microcontroller is used in this project due their flash based reprogram ability, and reasonable costs. Due its simple architecture structure, it is easier to program. Since there are a lot of types of microcontroller in the market, selection has to be made based on the specifications. For this project, we would require a microcontroller which has an analogue-to-digital converter (ADC) module. This was essential in order for us to control the "butterfly" opening via a potentiometer.

At the beginning, I started using a PIC 16F84A microcontroller to control to control the servo. After some attempts, I decided to switch to a PIC16F684 microcontroller. This was due to some functions which were needed for the servo controller. The reasons for choosing this particular microcontroller are numerous. The 16F684 has a maximum of eleven I/O and we only need two. We need one analog input to read the position of the potentiometer and one digital output to supply the control signal to the servo. The 16F684 has an analog-to-digital converter (ADC) built into it. It also has an internal oscillator capable of providing several oscillator frequencies from 31 kHz to 8 MHz which is very helpful. The 12F683 also has a built-in Pulse-Width-modulation (PWM) module which makes the programming a lot simpler.

The program was written in C and compiled using CCS, PCW compiler. The program is grouped into four major sections: initialization of registers and I/O, reading and averaging the value of the potentiometer, loading the value into the PWM registers, and setting the internal oscillator frequency. A sample of the programming code is as shown in **APPENDIX V**.

The program starts with a clock speed of 4 MHz and then reads timer 2 to determine if the clock speed should be switched to 250 kHz. We have to make sure that the oscillator is not switched to 250 kHz before the output pulse goes low. If it does go low, the width of the pulse would be stretched out and the servo would not function properly.

3.5 Project Activities

During the first semester, research was done on a Remote Control (RC) servomotor and throttles on how they actually work. A Daihatsu Detomaso 1.6cc fuel injection car throttle body was purchased and recondition for the project. Research was also done on why a servomotor was chosen for this project and on this, other alternatives were taken into account. By the end of week 13, a simulation model was built in MATLAB.

Beginning of the second semester, a pair of gear set was bought for the motor and moving shaft. Since the shape of the hole on the gears does not fit the motor and shaft, they were molded to fit in it. Later, a controller was then built to control the RC servomotor's direction. Initially, a SPDT switch type controller was used to control the RC servo. Since the project requires a precision in control of the RC servo, the best solution is to use a potentiometer to control the RC servo. The throttle position sensor (TPS) which came along together with the throttle was then used for error detection.

3.6 Tools

At the beginning of this project, a Daihatsu Detomaso 1.6cc fuel injection throttle body was purchased. A RC servomotor was also given to me by my supervisor for study purpose. Pictures of some of the equipments available are shown in **Appendix II.**

Since a servomotor was feasible for the project, it was used to drive the throttle shaft. I gear set, as shown previously in figure 13 was used to link them together. To build a controller for the motor, the collection of electronic components, for e.g. PIC microcontrollers, breadboards and resistors was found in the university. Stamping of a microcontroller is done in the microprocessor lab in the university.

For other components which are not available in the university, it was purchased elsewhere.

CHAPTER 4
RESULTS & DISCUSSION

4.1 Results

4.1.1 MATLAB Simulink

Since RC servos were chosen as the driver to drive the throttle plate in this project, a simulated model was created in MATLAB Simulink to determine its response. A model of the RC servo can be seen in Figure 14 below. The model is connected to a 5V step-signal where its response versus its effect was studied. This is to ensure that the close-loop system is stable and can be connected to the throttle.

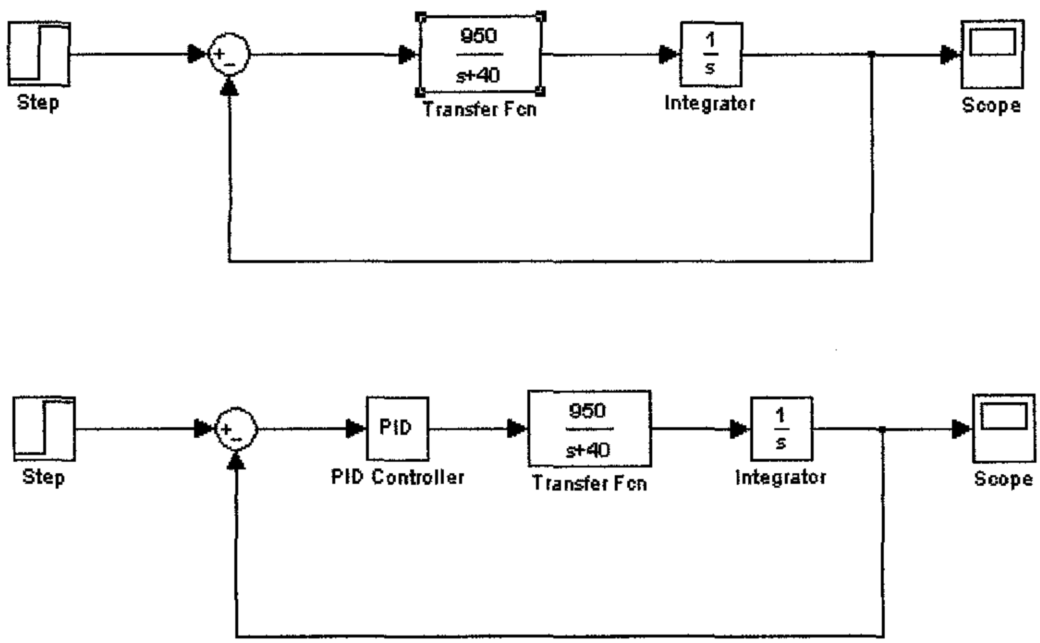


Figure 14: RC Servo Models

As seen in Figure 14, a PID controller was connected to in second circuit. This is to eliminate the overshoot which was not desirable when a 5V step signal is supplied to the system. Figure 15 and 16 shows the difference when a PID controller was implemented on the circuit.

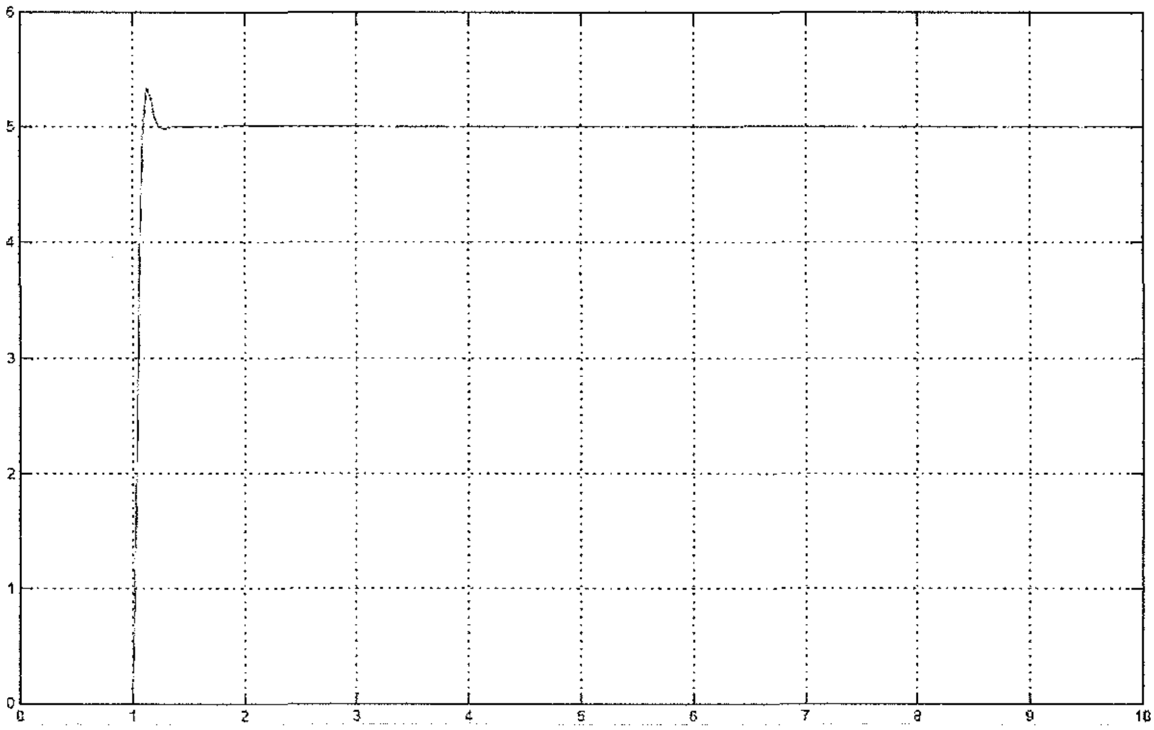


Figure 15: RC Servo Response without PID

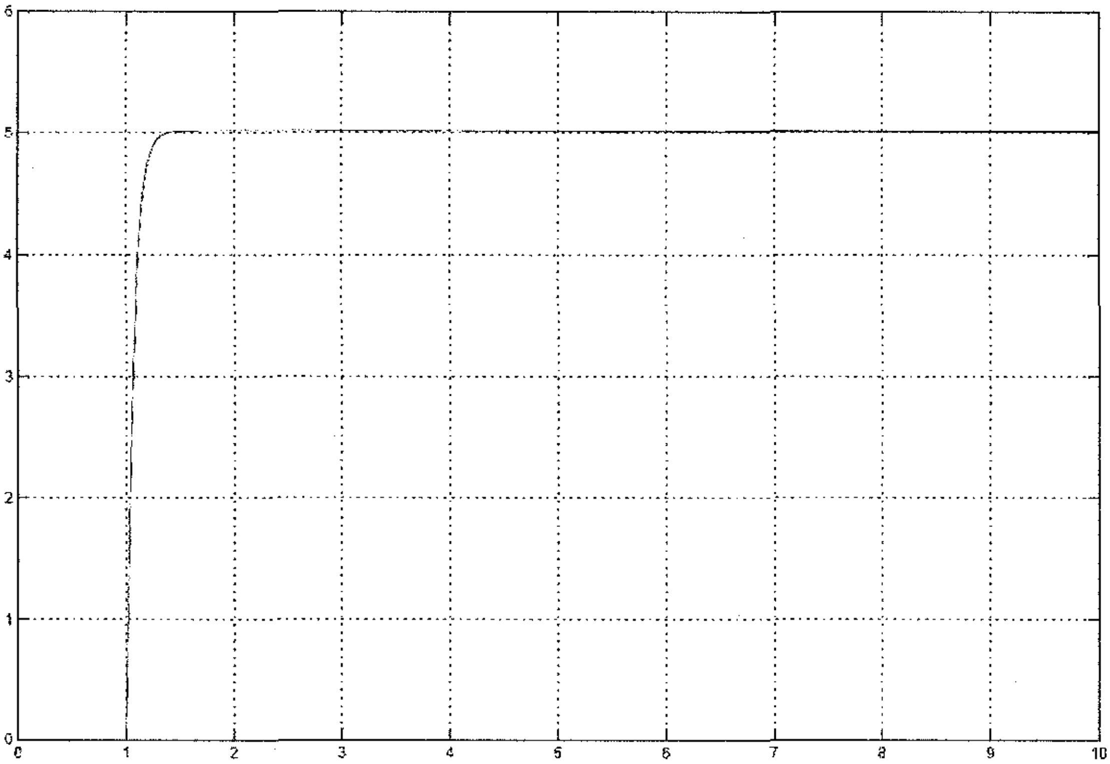


Figure 16: RC Servo Response With PID

Next, the RC Servo model was linked to a throttle model as shown in Figure 14. The subsystem model of the throttle is as show in **APPENDIX IV.**

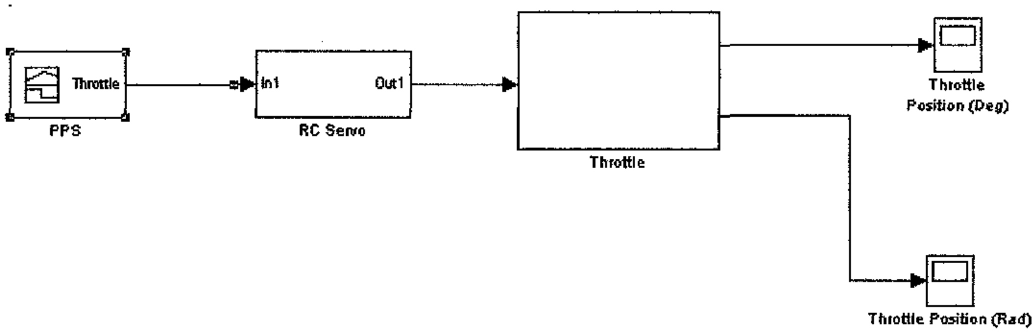


Figure 17:ETC model with RC Servo

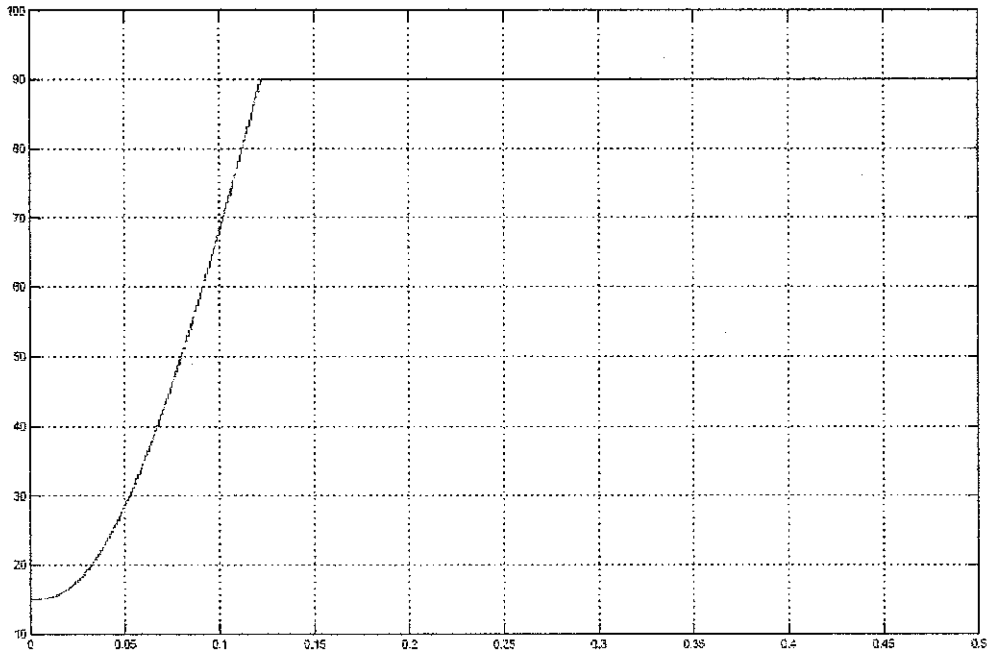


Figure 18: Ideal Response of ETC

Basically, an ideal response of the electronic throttle body should response as Figure 18. The response manages to stabilize at 0.125s when a throttle signal of 5V is supplied.

Setting the throttle's parameter as the table below, we obtain a throttle response as figure 19.

PARAMETERS	**VALUE**
Inertia (J)	0.35
Damping (c)	11
Stiffness (k)	1.4

Table 1: Physical condition of a throttle body

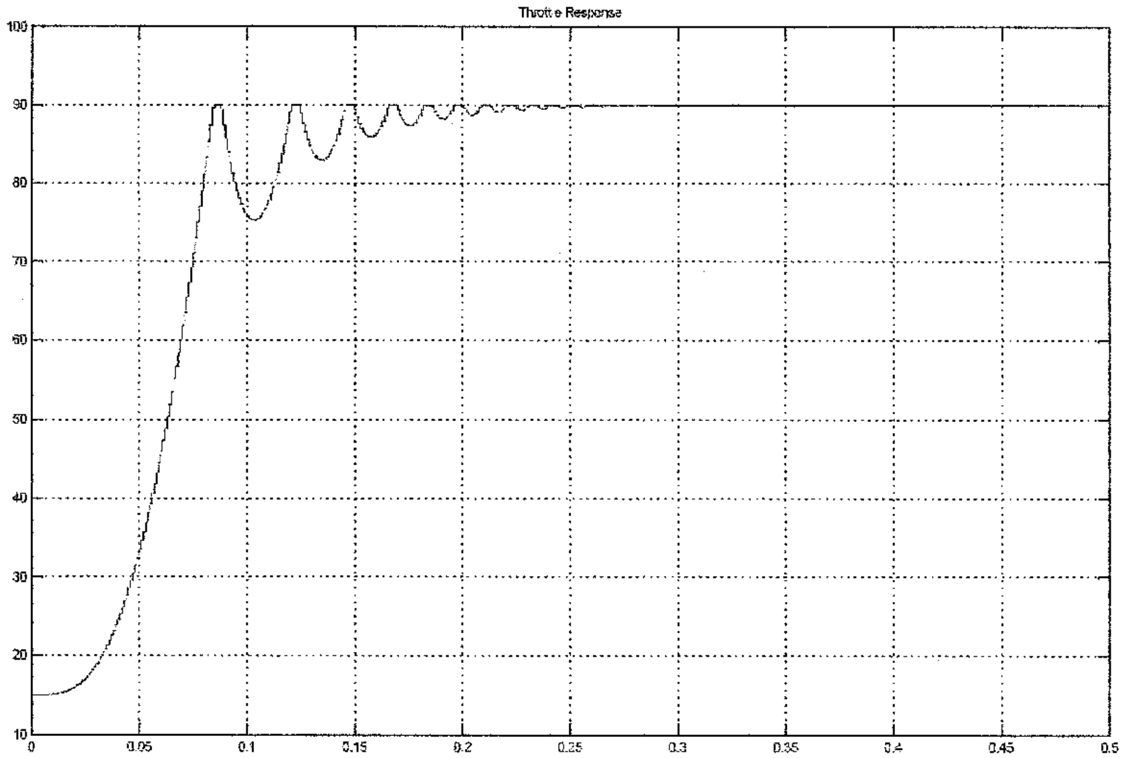


Figure 19: Response of ETC with parameter set

4.1.2 Gear ratio

For this project, we will be using two spur gears. The larger gear with 74-teeth will be linked to the throttle opening shaft while the smaller gear with 36-teeth will be connected to the RC Servomotor. Therefore,

Gear Ratio = 46 : 74

= 1 : 1.61

As we know, the desired maximum opening for a throttle is only 75 degrees; therefore, the range to drive the throttle gear is:

$$\frac{75}{360} = \frac{x}{74}$$

$$\therefore x = 15.5\ teeth$$

The servomotor should be programmed to operate in the range of:

$$\frac{15.5}{46} = \frac{x}{360}$$

$$\therefore x = 121.3°$$

Therefore, the RC servomotor should operate in the range of $0° \ll x \ll 121.3°$

4.1.3 Controller Circuit

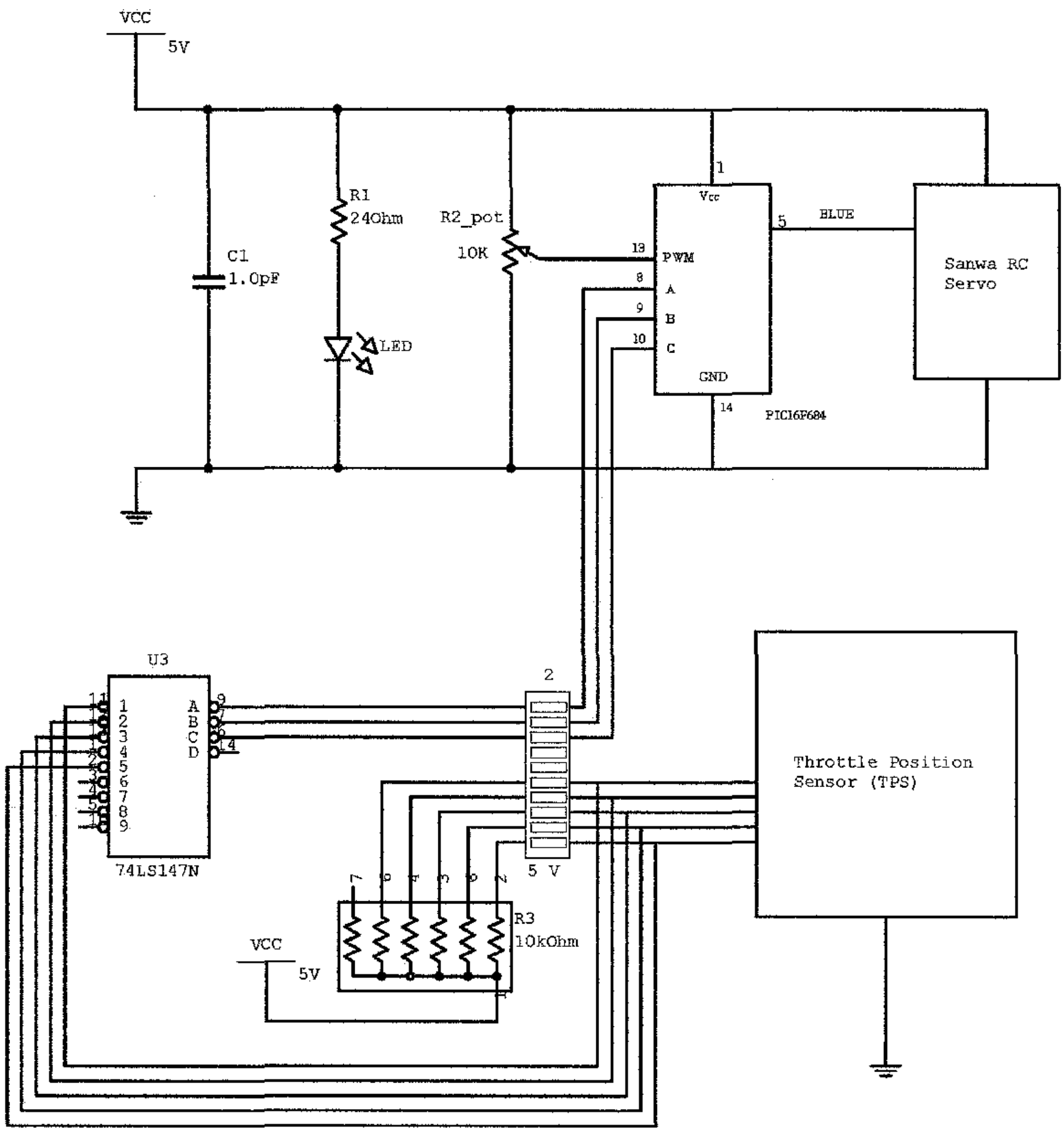


Figure 20 : Controller Circuit Schematic

Since we could create a microcontroller, we could also determine the pin we would like to connect the devices to. Since the maximum voltage we can supply to a RC Servo is 6V, a supply voltage of 5 V is supplied to the microcontroller and the RC servo. A light emitting diode (LED) is connected to the controller to ensure there's enough power across the circuit.

4.1.4 Programming the PIC controller

A PIC 16F684 microcontroller was used to control the RC Servo position using a potentiometer. The flowchart for the program is as shown in figure 21, and the C programming code is in **APPENDIX V**.

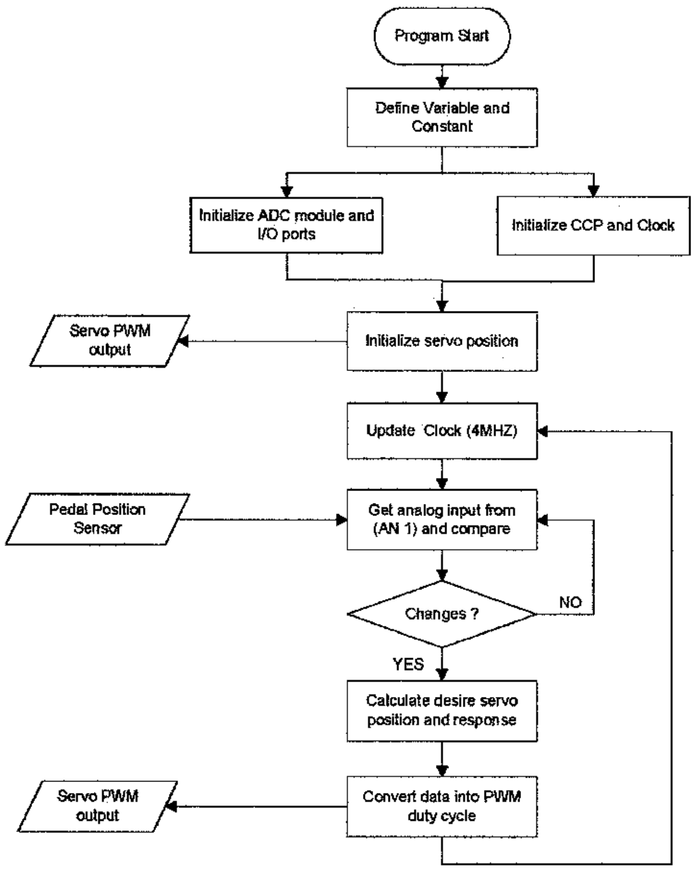


Figure 21: Controller Flow Chart

As can be observed from figure 21, the controller reads a pedal position sensor (PPS) via a potentiometer for positioning. In another way, we can conclude that the PPS controls the PWM wave supplied to the RC servo.

4.2 Throttle Position Sensor (TPS)

For a throttle control it is important to create a feedback to the controller for error detection and "butterfly" positioning correction. When the throttle was purchased, it came along with a throttle switch. Throttle switch are a type of TPS which uses binary bits to encode a certain angle. To determine the binary coding corresponding to the throttle angle opening, an experiment was conducted and the results was as following:

Angle	Coding Direct from TPS	Coding after encoding
15° – 30°	00001	100
30° – 45°	00010	000
45° – 60°	00100	111
60° – 75°	01000	011
75° – 90°	10000	101

Table 2: Coding of the TPS corresponding to the angle

Using a Motorola 74LS147 8-to-3 line encoder, the coding was reduced from a 5-bit output to a 3-bit output as shown in table 2. By this way, feedback input to the controller was reduced reducing the amount of data needed to be executed in a microprocessor. By this way, the microcontroller can perform faster and the throttle response could be improved.

4.3 Discussion

The MATLAB Simulink model was simulated to determine the response time of the throttle considering the physical factor (inertia, stiffness and damping) of the throttle body. Throughout this project, a spring was not use to turn the throttle opening back to its origin. Instate, a RC servo motor was used to control the "butterfly" opening freely to open and close without any disturbance. Using this concept, the motor does not need to counter against the back force of the throttle spring when it opens.

The only disturbance the RC servo motor will encounter is the vacuum effect when the surrounding air flows into the engine via the throttle. Even with the existence of this problem, the RC servo motor is capable of handling the load (3.5 kg/cm at 6.0v) exerted by the air flow due to its nature.

Referring to the gear set in figure 21, if a motor is driving the big gear via the small gear, it will require a smaller torque compared to when it drives a small gear via a big gear.

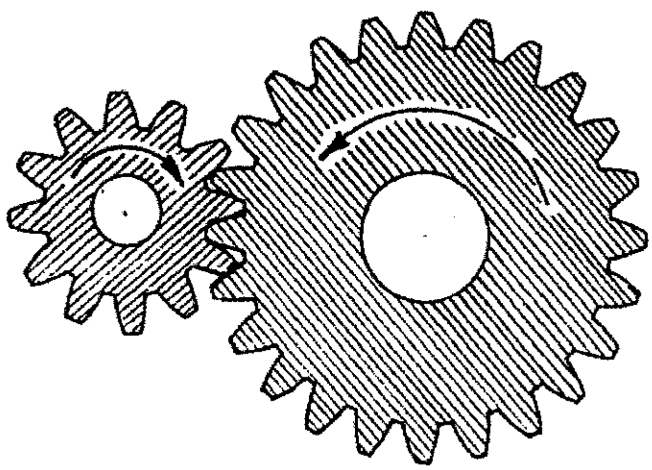


Figure 22: Gear set

In terms of response, it is better if the RC motor drives the throttle shaft (small gear) via the bigger gear. The only disadvantage is the RC servo motor will require a higher torque to counter the disturbance and the response will be slower. In this project, the RC servomotor was linked to the throttle shaft (bigger gear) via a small gear.

The response time of the motor is very important in this project. In order to achieve a fast response, it is important to supply a higher duty cycle frequency to the motor. For a RC servomotor, the maximum duty cycle we can supply to the RC servomotor is 60 hertz (Hz) [10]. Beyond 60 Hz, the RC servo will experience humping movements when rotating.

Besides controlling the frequency supplied to the RC Servo motor, it is also important to reduce the amount of programming lines (if ,else) for fast response. Normally, if there's a lot of programming lines in a program, the microprocessor will take a longer time to execute it. Therefore, it is important to keep the program as simple as possible and reduce the amount of programming lines.

CHAPTER 5
CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions and Recommendations

One of the limitations of the current RC servomotor is the operating speed which affect the response time. When the throttle pedal is pushed instantly from 0 to 140 degree, there was a lag of about 0.5 seconds. To solve this problem, the current RC servomotor can be replaced by high speed RC Servomotor. Doing this, the lag of the throttle can be reduced by more than half.

The prototype uses a throttle position sensor (TPS) to detect its “butterfly” position error. Since there are 5 steps from the current throttle position sensor (TPS), the TPS could only detect an error of more if it exceeds the 10 degree detection range. To further improve the throttle position detection, the TPS can be replaced with a potentiometer type of TPS. With the potentiometer type of TPS, the resolution of the error detection of the throttle can be increased.

In an electronic throttle control (ETC), the pedal position sensor (PPS) plays an important role especially ergonomically for the driver. Normally, a driver would exert a certain pressure depending on its demand on the pedal to control a vehicle via the throttle. For this project, focus was not put on designing a PPS which is ergonomically. Therefore, for further studies on the project, a more desirable PPS can be improved for the project.

Overall, a prototype of an electronic throttle control (ETC) was successfully built using a Remote Controlled (RC) servomotor. For further research, a circuit board could be built to read the data from the controller. The circuit board can then be interfaced via a computer and the data from the controller could then be simulated. The results from the simulation can then be studied and planed for further improvement.

REFERENCES

[1] Morris, James W. and Gavin, Jenney, *Fly-by-Wire,* A Historical and Design Perspective, Warrendale, Pa, 1998.

[2] NZ Government, 29 July 08, <http://www.beehive.govt.nz/release/exhaust+emissions+rule+improve environment>

[3] Brain, Marshall, 29 July 08 <http://auto.howstuffworks.com/engine.htm>.

[4] Robbins & Myers/Electro-Craft, *DC Motors Speed Controls Servo Systems*, 4th Ed, Minnesota, 1978.

[5] Don Knowles, *Automotive Computer Systems*, Delmar Publishers, Albany, 1996.

[6] Norman S. Nise, *Control System Engineering*, 4 Ed., Wiley & Sons Publication, Massachusetts, 2004.

[7] Robert Bosch, Robert Bosch GmbH Staff, Gasoline-Engine Management, 3rd Ed, Bentley Publishers, 2006.

[8] Kachroo, Pushkin and Mellodge, Patricia, *Mobile Robotic Car Design*, McGraw Hill, New York, 2005.

[9] Sanwa RC Servo Specification, 29th Sept 2008 <http://www.acehobby.co.nz/ossb2/root/OSSBEC1/showitem.asp?PID=54650>

[10] John Iovine, *PIC Microcontroller Project Book*, McGraw Hill, New York, 2004.

[11] Mark Schnubel, *Today's Technician*, Advance Engine Performance, Waterbury, CT, 2006.

[9] Lennon, Tony, 29th Sept 2008 <http://www.mathworks.com/matlabcentral/>

[10] Dabney, James B. and Harman, Thomas L., *Mastering Simulink*, McGraw Hill, New York, 2004.

[11] Pico Technology, 1st Sept 2008 <http://www.picoauto.com/applications/electronic-throttle-control.html>

[12] Motorola 74LS147 8-to-3 line encoder datasheet <www.datasheetcatalog.com>

APPENDIX I : GANTT CHART

FINAL YEAR PROJECT I

NO.	Detail/Work	1	2	3	4	5	6	7	8	9	10		11	12	13	14
1	**Selection of project topic**															
	Submission of proposal															
	Approval of proposal and supervisor															
2	**Preliminary research/literature review**															
	Background study on ETC															
	Research on throttles & motors															
3	**Submission of Preliminary**				15/8											
4	**Seminar**					TBA										
5	**Preliminary research/literature review**															
	Mechanism on ETCs															
	Controlling motors															
	Simulation softwares															
6	**Submission of Progress Report**									17/9						
7	**Interim Reoprt Presentation**															
	Draft Copy of report															
	Analyze all data gathered															
8	**Submission of Interim Report**													15/10		
9	**Final Oral Presentation**														TBA	

Final Year Project II

NO.	Detail/Work	15	16	17	18	19	20	21		22	23	24	25	26	27	28
1	**Project Continues**															
	Gather equipments for prototype															
	Research on PLCs															
2	**Submission of Progress Report**				TBA											
3	**Further Studies of the Project**															
	Response Improvement															
	Timing Improvement															
4	**Submission of Progress Report**									TBA						
5	**Seminar**															
6	**Poster Exhibition**											TBA				
7	**Submission of Dissertation (soft bound)**															
8	**Oral Presentation**															
9	**Submission of Dissertation**															

APPENDIX II : PICTURES

Picture 1:Remote Control Servomotor

Picture 2:Honda C70 Throttle Body

Picture 3: Daihatsu Throttle Body

APPENDIX III: SANWA SX-01 SERVO

Specification [9]

Speed : 0.21 / 60degrees at 4.8V

: 0.19 / 60degrees at 6.0v

Torque : 3.2 kg/cm at 4.8v

3.5 kg/cm at 6.0v

Dimension : 39.0 x 20.0 x 36.0 mm

Weight : 45 gm

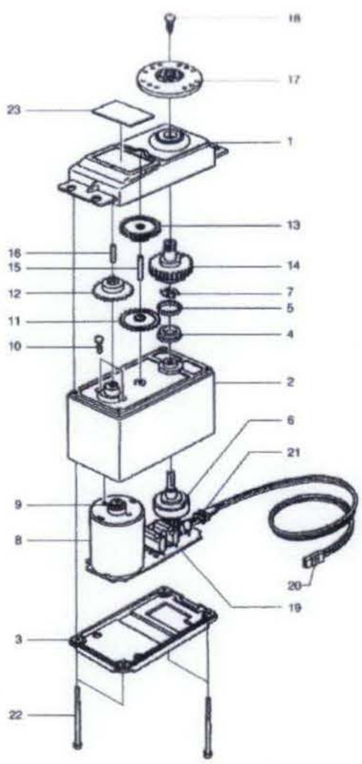
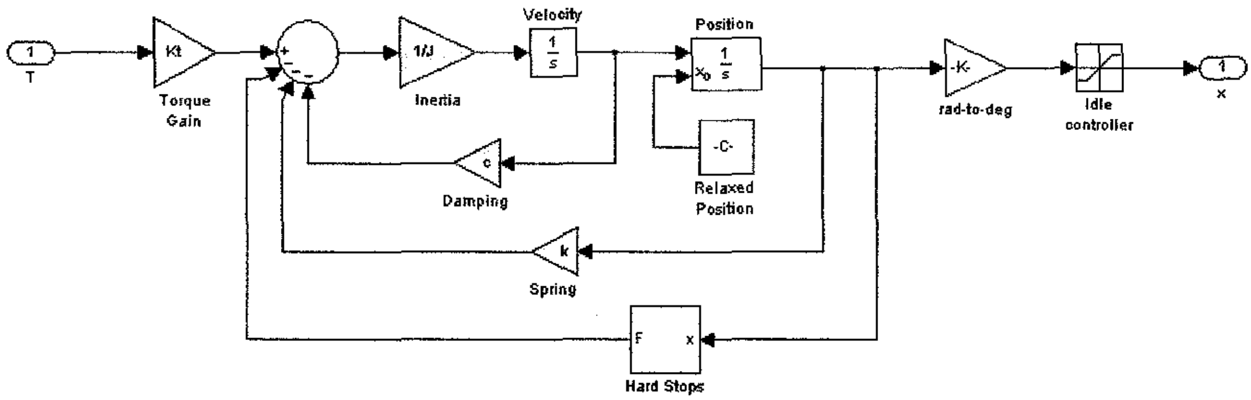


TABLE 5.5 The Components in the Futaba Servo

01	Upper case
02	Middle case
03	Bottom case
04	Metal bearing
05	Metal bearing
06	Potentiometer
07	Potentiometer drive plate
08	Motor
09	Motor pinion
10	Screw
11	1st gear
12	2nd gear
13	3rd gear
14	Final gear
15	Intermediate shaft
16	2nd shaft
17	Servo horn
18	Screw
19	Circuit board
20	Connector and cable
21	Cable bushing
22	Main assembly screw
23	Nameplate

Futaba Servo internals (image copyright from Futaba Corporation)

APPENDIX IV: MATLAB THROTTLE MODEL



APPENDIX V: PROGRAM CODE

```
#include <16F684.h>
#fuses HS, XT, NOWDT, NOPROTECT, NOBROWNOUT, PUT,
#use delay( clock = 4000000 )                           // SET INITIAL CLOCK TO 4MHz

int8 C0 = 0;                                            // FOR THE BUTTON ON PIN C0
int8 C1 = 0;                                            // FOR THE BUTTON ON PIN C1
int8 C2 = 0;                                            // FOR THE BUTTON ON PIN C2

void setup( void )                                      // SETUP THE A TO D, I/O AND CCP
{
        setup_adc_ports( 0 );                           // AN1 ANALOG INPUT
        setup_adc( ADC_CLOCK_DIV_8 );                   // ADC CLOCK
        set_adc_channel( 0 );                           // SELECT ADC CHANNEL 0
        setup_ccp1( ccp_pwm );                          // CONFIG CCP1 AS PWM
        setup_timer_2( T2_DIV_BY_16, 255, 1 );          // SETUP CLOCK AND PERIOD
        output_low(PIN_A1);                             // SET UNUSED I/O TO OFF
        output_low(PIN_A4);
        output_low(PIN_A5);

}
void main()
{
int32 sum;
int16 value, pwm;
int no_acqs, t2_value;

set_tris_C(0x111);                                      // SET PIN C0,C1 AND C2 AS INPUT

        setup();                                        // SETUP A TO D and PWM
        while ( TRUE )
                {

                value = Read_ADC();                     // READ POTENTIOMETER VALUE
                sum = sum + value;                      // SUM THE VALUE
                no_acqs ++;                             // INCREMENT AQUISITIONS
                if ( no_acqs > 127 )
                {
                        pwm = sum / 128;                // AVERAGE THE POT READING
                        set_pwm1_duty( pwm + 256 );     // SET THE PWM % + 1mS OFFSET
                        sum = 0;                        // CLEAR THE SUM
                        no_acqs = 0;                    // CLEAR THE ACQUISITION COUNTER
                }

                t2_value = get_timer2();                // GET THE VALUE OF TIMER #
                if ( t2_value > 140 && t2_value < 205 ) CHECK IF OSC SHOULD BE 250k or 4M
                {
```

```
                setup_oscillator( OSC_250KHZ );              // SET INTERNAL OSCILLATOR 250KHz
        }
        else
        {
                setup_oscillator( OSC_4MHZ );                // SET INTERNAL OSCILLATOR 4MHz
        }
        if (input(PIN_C0==0 && PIN_C1==1 && PIN_C2==1) )
        {
                get_timer2() > 140 && get_timer2() < 153;
        }
        else
        {
                get_timer2();
        }
        if (input(PIN_C0==1 && PIN_C1==1 && PIN_C2==1) )
        {
                get_timer2()> 153 && get_timer2()< 166;
        }
        else
        {
                get_timer2();
        }
        if (input(PIN_C0==0 && PIN_C1==1 && PIN_C2==1) )
        {
                get_timer2()> 166 && get_timer2()< 179;
        }
        else
        {
                get_timer2();
        }
        if (input(PIN_C0==0 && PIN_C1==1 && PIN_C2==1) )
        {
                get_timer2()> 179 && get_timer2()< 192;
        }
        else
        {
                get_timer2();
        }
        if (input(PIN_C0==0 && PIN_C1==1 && PIN_C2==1) )
        {
                get_timer2()> 192 && get_timer2()< 205;
        }
        else
        {
                get_timer2();
        }
    }
}
```

APPENDIX VI: PIC MICROPROCESSOR PINS & SPECIFICATION

MICROCHIP

PIC16F84A

18-pin *Enhanced* FLASH/EEPROM 8-Bit Microcontroller

High Performance RISC CPU Features:

- Only 35 single word instructions to learn
- All instructions single-cycle except for program branches which are two-cycle
- Operating speed: DC - 20 MHz clock input
  DC - 200 ns instruction cycle
- 1024 words of program memory
- 68 bytes of Data RAM
- 64 bytes of Data EEPROM
- 14-bit wide instruction words
- 8-bit wide data bytes
- 15 Special Function Hardware registers
- Eight-level deep hardware stack
- Direct, indirect and relative addressing modes
- Four interrupt sources:
  - External RB0/INT pin
  - TMR0 timer overflow
  - PORTB<7:4> interrupt-on-change
  - Data EEPROM write complete

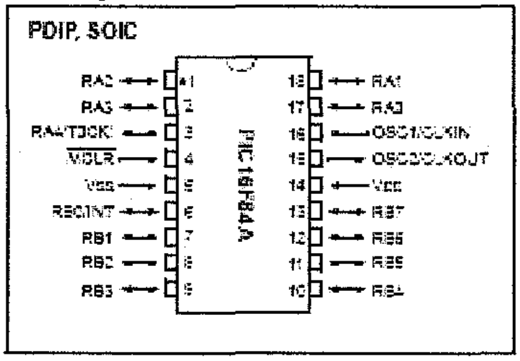
Peripheral Features:

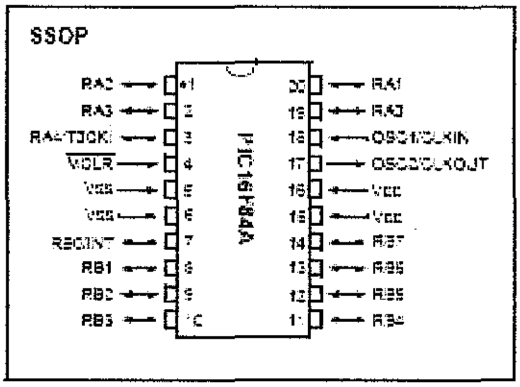
- 13 I/O pins with individual direction control
- High current sink/source for direct LED drive
  - 25 mA sink max. per pin
  - 25 mA source max. per pin
- TMR0: 8-bit timer/counter with 8-bit programmable prescaler

Special Microcontroller Features:

- 10,000 erase/write cycles *Enhanced* FLASH Program memory typical
- 10,000,000 typical erase/write cycles EEPROM Data memory typical
- EEPROM Data Retention > 40 years
- In-Circuit Serial Programming™ (ICSP™) - via two pins
- Power-on Reset (POR), Power-up Timer (PWRT), Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own On-Chip RC Oscillator for reliable operation
- Code protection
- Power saving SLEEP mode
- Selectable oscillator options

Pin Diagrams





CMOS Enhanced FLASH/EEPROM Technology:

- Low power, high speed technology
- Fully static design
- Wide operating voltage range:
  - Commercial: 2.0V to 5.5V
  - Industrial: 2.0V to 5.5V
- Low power consumption:
  - < 2 mA typical @ 5V, 4 MHz
  - 15 µA typical @ 2V, 32 kHz
  - < 0.5 µA typical standby current @ 2V

MICROCHIP

PIC16F684

14-Pin Flash-Based, 8-Bit CMOS Microcontrollers with nanoWatt Technology

High-Performance RISC CPU:

- Only 35 instructions to learn:
  - All single-cycle instructions except branches
- Operating speed:
  - DC – 20 MHz oscillator/clock input
  - DC – 200 ns instruction cycle
- Interrupt capability
- 8-level deep hardware stack
- Direct, Indirect and Relative Addressing modes

Special Microcontroller Features:

- Precision Internal Oscillator:
  - Factory calibrated to ±1%
  - Software selectable frequency range of 8 MHz to 31 kHz
  - Software tunable
  - Two-speed Start-up mode
  - Crystal fail detect for critical applications
  - Clock mode switching during operation for power savings
- Power-saving Sleep mode
- Wide operating voltage range (2.0V-5.5V)
- Industrial and Extended Temperature range
- Power-on Reset (POR)
- Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Brown-out Detect (BOD) with software control option
- Enhanced low-current Watchdog Timer (WDT) with on-chip oscillator (software selectable nominal 268 seconds with full prescaler) with software enable
- Multiplexed Master Clear with pull-up/input pin
- Programmable code protection
- High Endurance Flash/EEPROM cell:
  - 100,000 write Flash endurance
  - 1,000,000 write EEPROM endurance
  - Flash/Data EEPROM retention: > 40 years

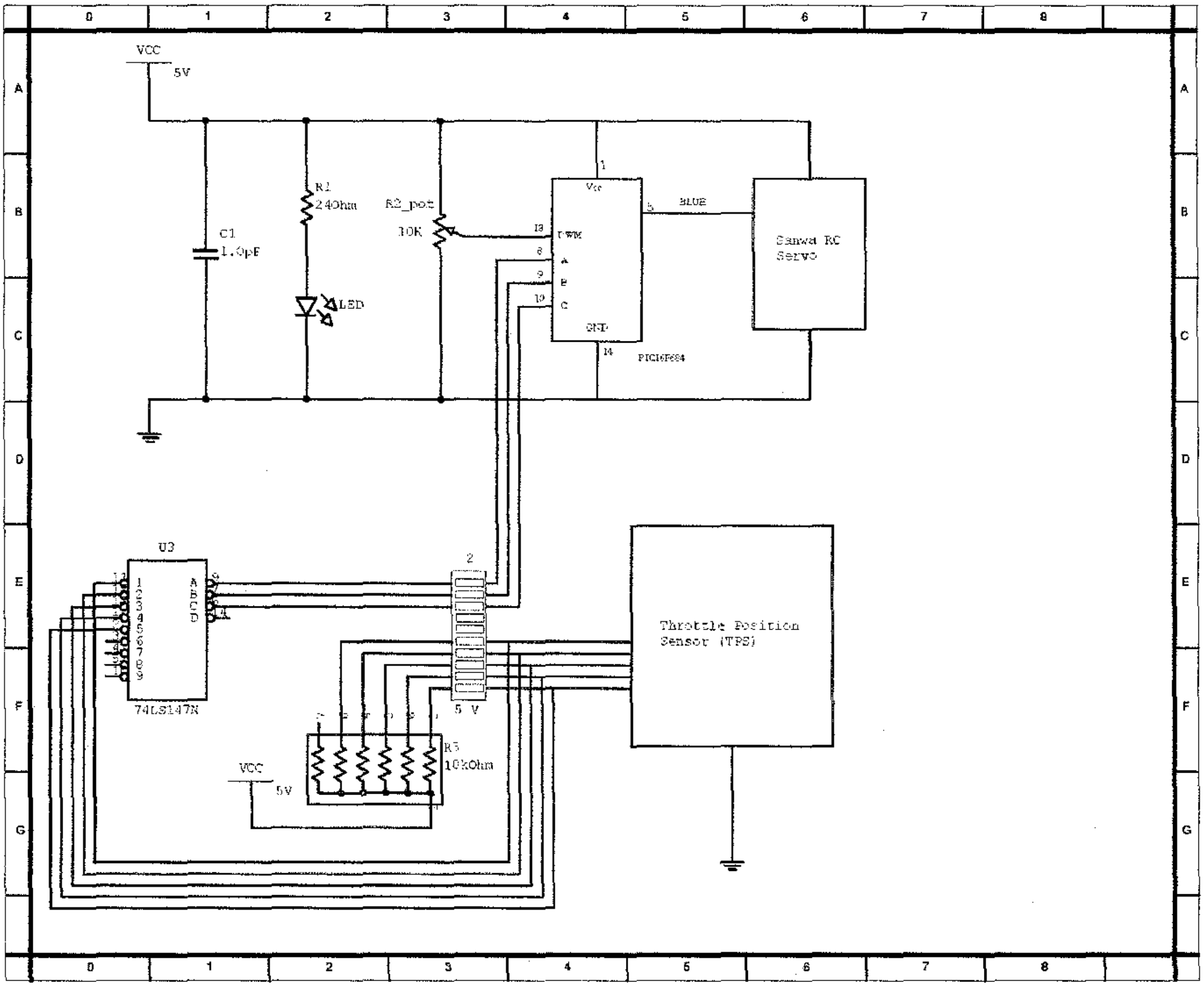
Low-Power Features:

- Standby Current:
  - 1 nA @ 2.0V, typical
- Operating Current:
  - 8.5 µA @ 32 kHz, 2.0V, typical
  - 100 µA @ 1 MHz, 2.0V, typical
- Watchdog Timer Current:
  - 1 µA @ 2.0V, typical

Peripheral Features:

- 12 I/O pins with individual direction control:
  - High current source/sink for direct LED drive
  - Interrupt-on-pin change
  - Individually programmable weak pull-ups
  - Ultra Low-power Wake-up (ULPWU)
- Analog comparator module with:
  - Two analog comparators
  - Programmable on-chip voltage reference (CVREF) module (% of VDD)
  - Comparator inputs and outputs externally accessible
- A/D Converter:
  - 10-bit resolution and 8 channels
- Timer0: 8-bit timer/counter with 8-bit programmable prescaler
- Enhanced Timer1:
  - 16-bit timer/counter with prescaler
  - External Gate Input mode
  - Option to use OSC1 and OSC2 in LP mode as Timer1 oscillator if INTOSC mode selected
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Enhanced Capture, Compare, PWM module:
  - 16-bit Capture, max resolution 12.5 ns
  - Compare, max resolution 200 ns
  - 10-bit PWM with 1, 2 or 4 output channels, programmable "dead time", max frequency 20 kHz
- In-Circuit Serial Programming™ (ICSP™) via two pins

APPENDIX VII: SCHEMATIC DIAGRAM AND COMPONENT USED



Controller Schematic Diagram

Controller Parts List

- ❖ Four 1.5V alkaline batteries
- ❖ Green LED
- ❖ 10K potentiometer
- ❖ 16F684 Microcontroller
- ❖ 0.01 μF Capacitor
- ❖ Miscellaneous: battery plug, eight pin DIP socket, female servo plug, breadboard, wires, socket connectors

APPENDIX VIII: DATA SHEET FOR 74LS147 8-TO-3 LINE ENCODER

MOTOROLA

10-LINE-TO-4-LINE AND 8-LINE-TO-3-LINE PRIORITY ENCODERS

SN54/74LS147
SN54/74LS148
SN54/74LS748

10-LINE-TO-4-LINE AND 8-LINE-TO-3-LINE PRIORITY ENCODERS

LOW POWER SCHOTTKY

The SN54/74LS147 and the SN54/74LS148 are Priority Encoders. They provide priority decoding of the inputs to ensure that only the highest order data line is encoded. Both devices have data inputs and outputs which are active at the low logic level.

The LS147 encodes nine data lines to four-line (8-4-2-1) BCD. The implied decimal zero condition does not require an input condition because zero is encoded when all nine data lines are at a high logic level.

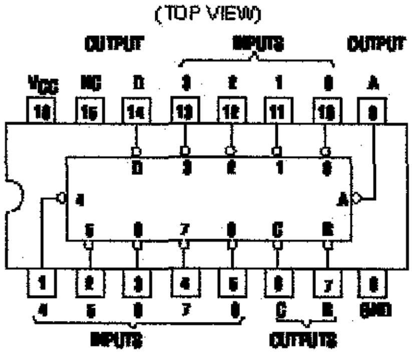
The LS148 encodes eight data lines to three-line (4-2-1) binary (octal). By providing cascading circuitry (Enable Input EI and Enable Output EO) octal expansion is allowed without needing external circuitry.

The SN54/74LS748 is a proprietary Motorola part incorporating a built-in deglitcher network which minimizes glitches on the GS output. The glitch occurs on the negative going transition of the EI input when data inputs 0–7 are at logical ones.

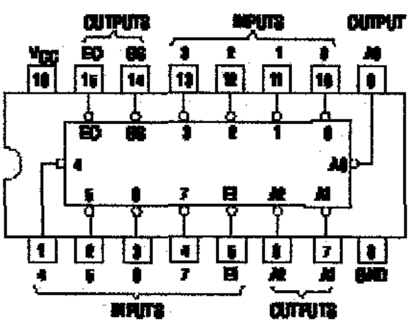
The only dc parameter differences between the LS148 and the LS748 are that (1) Pin 10 (input0) has a fan-in of 2 on the LS748 versus a fan-in of 1 on the LS148; (2) Pins 1, 2, 3, 4, 11, 12 and 13 (inputs 1, 2, 3, 4, 5, 6, 7) have a fan-in of 3 on the LS748 versus a fan-in of 2 on the LS148.

The only ac difference is that t_{PHL} from EI to EO is changed from 40 to 45 ns.

SN54/74LS147
(TOP VIEW)

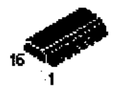


SN54/74LS148
SN54/74LS748
(TOP VIEW)



J SUFFIX
CERAMIC
CASE 620-09

N SUFFIX
PLASTIC
CASE 648-08



D SUFFIX
SOIC
CASE 751B-03

ORDERING INFORMATION

SN54LSXXXJ	Ceramic
SN74LSXXXN	Plastic
SN74LSXXXD	SOIC

SN54/74LS147
FUNCTION TABLE

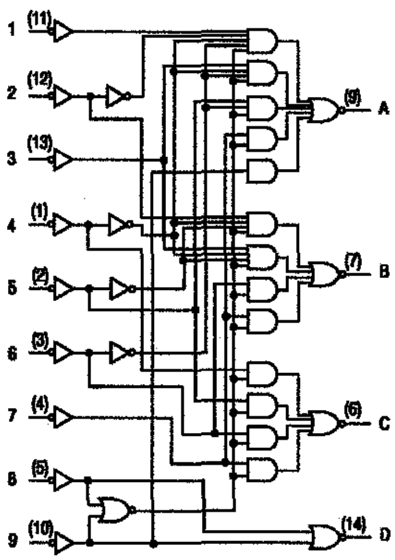
INPUTS									OUTPUTS			
1	2	3	4	5	6	7	8	9	D	C	B	A
H	H	H	H	H	H	H	H	H	H	H	H	H
X	X	X	X	X	X	X	X	L	L	H	H	L
X	X	X	X	X	X	X	L	H	L	H	H	H
X	X	X	X	X	X	L	H	H	H	L	L	L
X	X	X	X	X	L	H	H	H	H	L	L	H
X	X	X	X	L	H	H	H	H	H	L	H	L
X	X	X	L	H	H	H	H	H	H	L	H	H
X	X	L	H	H	H	H	H	H	H	H	L	L
X	L	H	H	H	H	H	H	H	H	H	L	H
L	H	H	H	H	H	H	H	H	H	H	H	L

H = HIGH Logic Level, L = LOW Logic Level, X = Irrelevant

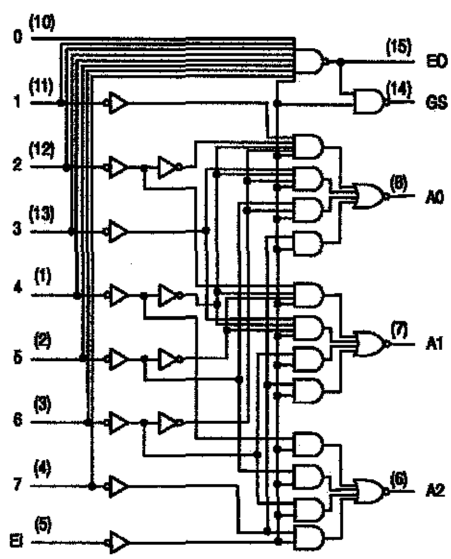
SN54/74LS148
SN54/74LS748
FUNCTION TABLE

INPUTS									OUTPUTS				
EI	0	1	2	3	4	5	6	7	A2	A1	A0	GS	EO
H	X	X	X	X	X	X	X	X	H	H	H	H	H
L	H	H	H	H	H	H	H	H	H	H	H	H	L
L	X	X	X	X	X	X	X	L	L	L	L	L	H
L	X	X	X	X	X	X	L	H	L	L	H	L	H
L	X	X	X	X	X	L	H	H	L	H	L	L	H
L	X	X	X	X	L	H	H	H	L	H	H	L	H
L	X	X	X	L	H	H	H	H	H	L	L	L	H
L	X	X	L	H	H	H	H	H	H	L	H	L	H
L	X	L	H	H	H	H	H	H	H	H	L	L	H
L	L	H	H	H	H	H	H	H	H	H	H	L	H

FUNCTIONAL BLOCK DIAGRAMS



SN54/74LS147



SN54/74LS148

APPENDIX IX: PICTURE OF THE PROTOTYPE MODEL