UNIVERSITI MALAYSIA PAHANG

BORANG PENGESAHAN STATUS TESIS♦

JUDUL: **DC MOTOR CONTROLLER USING LINEAR QUDRATIC REGULATOR (LQR) IMPLEMENTION ON PIC**

SESI PENGAJIAN: **2008/2009**

Saya **NUR IZNIE AFRAH BINTI MOHD ISA (860909-33-5794)**
(HURUF BESAR)

mengaku membenarkan tesis (Sarjana Muda/~~Sarjana~~ /~~Doktor Falsafah~~)* ini disimpan di Perpustakaan dengan syarat-syarat kegunaan seperti berikut:

1. Tesis adalah hakmilik Universiti Malaysia Pahang (UMP).
2. Perpustakaan 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)

_____ (TANDATANGAN PENYELIA)

Alamat Tetap:

NO. 3757, KG AIR MELINTAS BESAR,
13300, TASEK GELUGOR, S.P.U,
PULAU PINANG.

HASZURAIDAH BINTI ISHAK
(Nama Penyelia)

Tarikh: **17 NOVEMBER 2008**

Tarikh: : **17 NOVEMBER 2008**

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

"I hereby acknowledge that the scope and quality of this thesis is qualified for the award of the Bachelor Degree of Electrical Engineering (Electronics)"

Signature : _____

Name : HASZURAIDAH BINTI ISHAK

Date : 17 NOVEMBER 2008

DC MOTOR CONTROLLER USING LINEAR QUADRATIC REGULATOR (LQR) ALGORITHM IMPLEMENTION ON PIC

NUR IZNIE AFRAH BINTI MOHD ISA

This thesis is submitted as partial fulfillment of the requirements for the award of the Bachelor of Electrical Engineering (Hons.) (Electronics)

Faculty of Electrical & Electronics Engineering
Universiti Malaysia Pahang

NOVEMBER, 2008

Signature : _____

Author : NUR IZNIE AFRAH BINTI MOHD ISA

Date : 17 NOVEMBER 2008

Dedicated to my beloved family, and the lecturers and friends of Universiti Malaysia Pahang.

ACKNOWLEDGEMENT

First and foremost, the highest thank to God as finally I can accomplish my Final Year Project. Special thanks to my supervisor, Mrs Haszuraidah binti Ishak who help me in many things and give me the encouragement and guideline while doing this project. To my friend, Wan Robaah binti W Ahmad, I really appreciate your help and support in helping me finish this project.

I also would like to give this appreciation to all the panels for their advice. Not forgetting, to all the lecturers and staff of Faculty of Electrical and Electronic Engineering and to all my friends who were helping me direct or indirect, your help, support and cooperation will always be remembered.

ABSTRACT

Linear Quadratic Regulator (LQR) algorithm is one of the controller methods to control a system. In this project, the LQR was implemented on the PIC microcontroller to control the dc motor. The main objective of this controller is to minimize the deviation of the speed of dc motor. Dc motor speed is controlled by its driving voltage. The higher the voltage, the higher the motor speed. The speed of the motor is specifying that will be the input voltage of the motor and the output will be compare with the input. As the result, the output must be the same as or approximately the same as the input voltage. In this project, the LQR algorithm was implemented on the PIC microcontroller so the result can be shown. Before the implementation on the PIC, the dc motor state-space has to be derived. Then, from the state-space, we can design the LQR controller by using the MATLAB software. The stable system is got by tuning the Q and R value that can be seen by the simulation.

ABSTRAK

Algorithma Linear Quadratic Regulator (LQR) merupakan salah satu daripada kaedah pengawal untuk mengawal sesuatu sistem. Melalui projek ini, LQR dilaksanakan ke atas pengawal mikro PIC untuk mengawal dc motor. Objektif utama pengawal LQR adalah bagi meminimumkan kadar pesongan kelajuan motor. Kelajuan dc motor dikawal oleh pemanduan voltannya. Semakin tinggi voltan, semakin meningkat kelajuannya. Kelajuan dc motor yang dispesifikasikan akan menjadi voltan masuk kepada motor dan kemudiannya akan dibandingkan dengan voltan yang dikeluarkan. Hasilnya, keluaran mestilah sama ataupun hampir sama dengan kemasukan. Dalam projek ini, algorithma LQR dilaksanakan ke atas pengawal miko PIC supaya hasilnya dapat dilihat. Sebelum pelaksanaan ke atas pengawal mikro PIC, *state-space* bagi dc motor haruslah didapatkan. Kemudian, daripada *state-space* tersebut barulah pengawal LQR dapat dibentuk dengan menggunakan perisian MATLAB. Sistem yang stabil akan dihasilkan dengan membetulkan nilai Q dan R dan dapat dilihat melalui simulasi.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATION

DC	-	Direct Current
LQR	-	Linear Quadratic Regulator
PIC	-	Programmable Integrated Circuit
PWM	-	Pulse Width Modulation
SFR	-	Special Function Registers
RAM	-	Random Access Memory
USB	-	Universal Serial Bus
PC	-	Personal Computer

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Background

DC motor can be controlled either by software or directly by hardware. Software controlling needs computers which are bulky and common man cannot afford for it, so hardware controls are in use. Even in hardware if it is programmable device then it is preferred because it can be modeled according to the requirements of the user.

This project needs to develop the DC motor controller which is brushless servomotor by using Linear Quadratic Regulator (LQR) that is implemented on Programmable Integrated Circuit (PIC). To make the motor operate, first we must make the program on PIC. On the program, the input has to be set up. Then the input voltage is converted from analog signal into digital signal by ADC in the PIC. The PWM (Pulse Width Modulation) function of PIC is used for the electric current control to drive a motor.

The function of Linear Quadratic Regulator (LQR) is to minimize the deviation of the speed of the motor. The speed of the motor is specifying that will be the input voltage of the motor and the output will be compare with the input. The output must be the same as or approximately the same as the input voltage.

The advantages of used LQR are it is easy to design and increases the accuracy of the state variables by estimating the state. The nice feature of the LQR control as compared to pole placement is that instead of having to specify where n eigenvalues should be placed a set of performance weighting are specified that could have more intuitive appeal. The result is a control that is guaranteed to be stable.

1.2 Problem Statement

The speed of the DC motor may change due to disturbance present surrounding it. This will make the desired speed sometimes change and will be not maintain. By using LQR control algorithm, the deviation of the speed can be minimized.

1.3 Objectives

The objectives of this project are;

i. To implement Linear Quadratic Regulator (LQR) controller in Programmable Integrated Circuit (PIC) which can minimized the error of the speed of dc servomotor. The PIC that has been used in this project was PIC16F84A.

ii. To develop dc motor controller by using LQR algorithm.

1.4 Scope

This project actually concentrates on derivation of the mathematical model of dc servomotor and gets the value of K in LQR algorithm. K is the gain of the close loop system. To get the value of K, the state-space of the servomotor must be define first. So, LQR algorithm was used by means to minimize the deviation of the dc motor speed. The LQR is used to tune the value of Q and R. The value of Q and R is tuned the get the stable system.

CHAPTER 2

LITERATURE REVIEW

2.1 DC Motor

A DC motor is devised to convert electrical power into mechanical power. In DC motor, electrical energy is converted into mechanical energy through the interaction of two magnetic fields. One field is produced by permanent magnet assembly (on the start) and the other field is produced by an electrical current flowing in the motor winding (on the rotor). These two fields result in a torque that tends to rotate the rotor. As the rotor turns, the current in the windings is commutated to produce a continuous torque output.[1]

DC motor speed is controlled by controlling its driving voltage. The higher the voltage, the higher the motor speed. In many applications, a simple voltage regulation would cause lots of power loss in the control circuit, so a PWM method is used in many DC motor-controlling applications. In basic PWM method, the operating power to the

motors is turned on and off to modulate the current to the motor. The ration of on time to off time is what determines the speed of the motor.

A PWM circuit can be implemented by using discrete components. However, this approach cannot provide the desired flexibility and controllability is expensive. A better implementation method for PWM circuitry is to use the PWM functions available in many microcontrollers today. Most of the PIC 18 devices have PWM functions.[1]

Controlling the speed of the motor is an important area to be considered. The speed of motor is directly proportional to the DC voltage applied across its terminals. Hence, if we control the voltage applied across its terminal we actually control its speed.

A PWM (Pulse Width Modulation) wave can be used to control the speed of the motor. Here the average voltage given or the average current flowing through the motor will change depending on the ON and OFF time of the pulses controlling the speed of the motor. The duty cycle of the wave controls its speed. [2]

For this project, Clifton Precision Servo Motor Model JDH-2250-HF-2C-E is used. The specifications of this DC motor are;

- Torque Constant: 15.76 oz-in. / A
- Back EMF: 11.65 VDC / KRPM
- Peak Torque: 125 oz-in.
- Cont. Torque: 16.5 oz-in.
- Encoder: 250 counts / rev.
- Channels A, B in quadrature, 5 VDC input (no index)
- Body Dimensions: 2.25" dia. x 4.35" L (includes encoder)
- Shaft Dimensions: 8 mm x 1.0" L w/flat

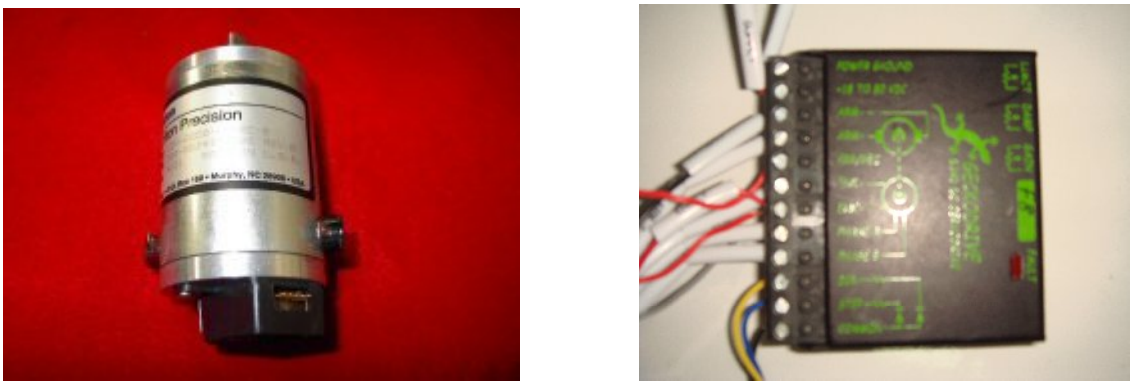


Figure 2.1 Clifton Precision Servo Motor Model JDH-2250-HF-2C-E with its encoder

2.2 Linear Quadratic Regulator (LQR) Algorithm

In layman's terms, Linear Quadratic Regulator (LQR) means the settings of a (regulating) controller governing either a machine or process (like an airplane or chemical reactor) are found by using a mathematical algorithm that minimizes a cost function with weighting factors supplied by a human (engineer). The "cost" (function) is often defined as a sum of the deviations of key measurements from their desired values. In effect this algorithm therefore finds those controller settings that minimize the undesired deviations, like deviations from desired altitude or process temperature. Often the magnitude of the control action itself is included in this sum as to keep the energy expended by the control action itself limited.[5]

In the particular case of a quadratic performance index combining the square of the error and square of the actuation, the solution to the optimal control problem is a feedback control where the measurements used for the feedback are all of the state variables. In this feedback control, each of the state variables is multiplied by a gain and the results are summed to get a single actuation value. The result of the LQR formulation is the set of gains, based on the relative weighting of the error and actuation in the performance index. [6]

Identification techniques based on minimization of equation error give good results, but it is essential that their sensitivity to high frequency noise and sample rate be compensated. This fact has generally been overlooked in previous literature on identification of flexible structures. Alternative approaches based on minimization of output error perform poorly, probably due to the existence of local minima in the performance index. [5]

Suppose that the space model is

$$\dot{x} = Ax + Bu, \tag{2.1}$$

with x(t) € R n , u(t) € R m. The initial condition is x (0).[6]

Suppose that we have sensors to measure the entire state and that we use a controller (regulator)

$$u = -K x \tag{2.2}$$

that seeks to drive the state to zero. You could use pole placement via Ackerman's formula. Here, we use the LQR methodology to specify the gain K. For this, let

$$J = \int_0^\infty (x^t Q x + u^T R u) d t \tag{2.3}$$

where Q = Q ≥ 0 and R = R > 0.The term "linear-quadratic" refers to the linear system dynamics and the quadratic cost function and we seek to find the gain vector K to minimize this "cost function."[7]

2.3 Programmable Integrated Controller (PIC) Microcontroller

A PIC is a **P**rogrammable **I**ntegrated **C**ircuit microcontroller, a 'computer-on-a-chip'. They have a processor and memory to run a program responding to inputs and controlling outputs, so they can easily achieve complex functions which would require several conventional ICs.[4]

The input voltage to PIC is converted by A/D converter. Changed voltage is used for the PWM function of the CCP to control the motor drive. At the circuit this time, a small motor is used as the generator to detect the number of rotations of the motor. The input voltage (the control voltage) to PIC is changed by the fluctuation of the number of rotations of the motor. The PWM (Pulse Width Modulation) function of PIC is used for the electric current control to drive a motor.[8]

Many methods evolved to control the revolution of a motor. DC motors can be controlled either by software or directly by hardware. Software controlling needs computers which are bulky and common man cannot afford for it, so hardware controls are in use. Even in hardware if it is programmable device then it is preferred because it can be modeled according to the requirements of the user.[2]

Advantages of using PIC over other controlling devices for controlling the DC motor are given below:

- **Speed:** The execution of an instruction in PIC IC is very fast (in micro seconds) and can be changed by changing the oscillator frequency. One instruction generally takes 0.2 microseconds.
- **Compact:** The PIC IC will make the hardware circuitry compact.
- **RISC processor:** The instruction set consists only 35 instructions.
- **EPROM program memory:** Program can be modified and rewritten very easily.
- **Inbuilt hardware support:** Since PIC IC has inbuilt programmable timers, ports an interrupts, no extra hardware is needed.
- **Powerful output pin control:** Output pins can be driven to high state, using a single instruction. The output pin can drive a load up to 25mA.
- **Inbuilt I/O ports expansions:** This reduces the extra IC's which are needed for port expansion and port can be expanded very easily.
- **Integration of operational features**: Power on reset and brown/out protection ensures that the chip operates only when the supply voltage is within specification. A watchdog timer resets PIC if the chip ever malfunctions and deviates from its normal operation.

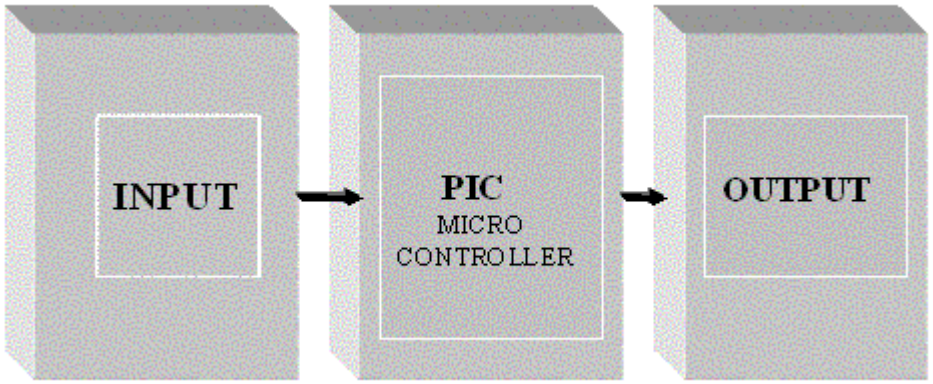


Figure 2.2 DC Motor Controlling System

The block diagram of the circuit is shown above. This circuit controls the speed and direction of the motor. The PWM (Pulse Width modulation) output from the four port pins is given to the H-Bridge circuit which drives the motor. On changing the duty cycle (ON time), we can change the speed. By interchanging output ports, it will effectively change direction of the motor. [9]

The PIC microcontroller is the brain of the circuit controlling all actions to be done. Inputs are given to control the speed and direction of the motor. The PIC output controls the DC motor.

CHAPTER 3

METHODOLOGY

This chapter explains on getting the state-space model of the dc servomotor, LQR design, hardware configuration and the implementation of LQR controller on PIC.

3.1 Program Flow Chart

The flow chart in Figure 4 shows the flow of this project. The mathematical modeling is doing to find the mathematical model for the DC motor where we will get the state-space model. This followed by getting the LQR controller from the state-space model. The MATLAB software is used to get the result. This mathematical modeling had to be done so the result that was get can be compare with the result from the experiment. The hardware design is needed to implement the DC motor with the PIC. The software that used to make the programming of the PIC is MicroCode Studio. Then, the integration between the software and hardware will run the motor.

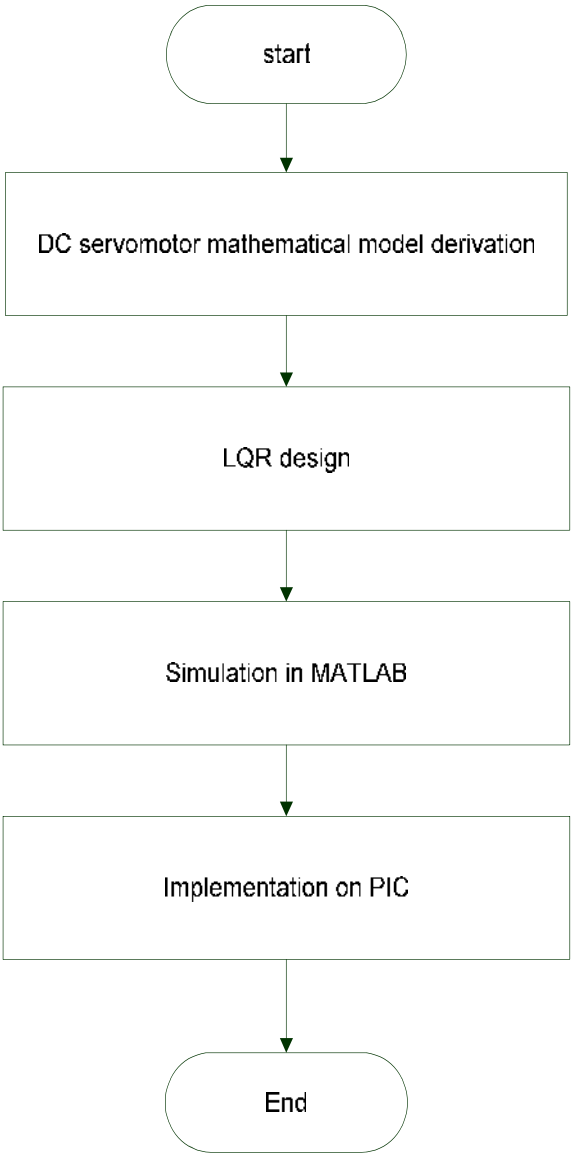


Figure 3.1 Flow chart of the project

3.2 Mathematical Model of DC Motor

The equivalent electrical circuit of a DC motor is illustrated in Figure 3.2. It can be represented by a voltage source (V_a) across the coil of the armature. The electrical equivalent of the armature coil can be described by an inductance (L) in series with a resistance (R) in series with an induced voltage (V_c) which opposes the voltage source. The induced voltage is generated by the rotation of the electrical coil through the fixed flux lines of the permanent magnets. This voltage is often referred to as the back emf (electromotive force).

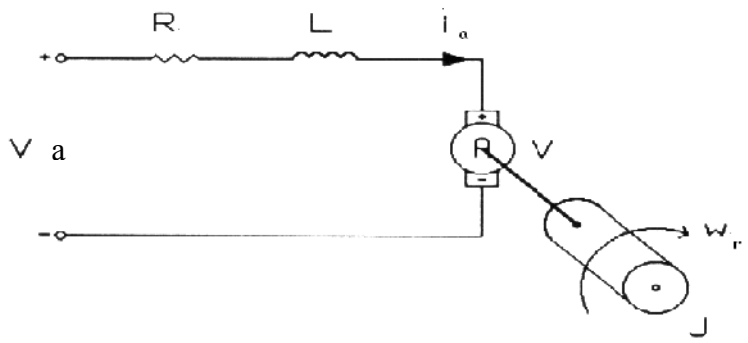


Figure 3.2 Electrical Circuit Representation Of A Dc Motor.

The physical parameters have been set up as followed;

Parameters	Values
Electric Resistance, R	2.7 Ω
Electric Inductance, L	0.004 H
Electromotive Force Constant, K	0.105 Nm A^{-1}
Moment of Inertia of the Rotor, J	0.0001 Kg m2
Damping Ratio of the Inertia of the Rotor, B	0.0000093 Nms rad^{-1}

3.2.1 Electrical characteristic

From the Figure 3.2, the following equations based on Newton's Law and Kirchoff's Law has been had.

$$\frac{di_a}{dt} = \frac{R}{L} i_a - \frac{K}{L} \omega_r + \frac{1}{L} V_a \tag{3.1}$$

$$\frac{d\omega_r}{dt} = \frac{K}{J} i_a - \frac{B}{J} \omega_r \tag{3.2}$$

3.2.2 Mechanical characteristic

The motor torque, T is related to the armature current, i_a by a constant factor, K and can be written as;

$$T = Ki_a \tag{3.3}$$

3.3.3 State-space Representation

In the state-space form, the equations above can be expressed by choosing the rotating speed and electrical current as the states variables and the voltage as an input. The output is chosen to be the rotating speed.

$$\begin{bmatrix} \frac{di_a}{dt} \\ \frac{d\omega_r}{dt} \end{bmatrix} = \begin{bmatrix} -\frac{R}{L} & -\frac{K}{L} \\ \frac{K}{J} & -\frac{B}{J} \end{bmatrix} \begin{bmatrix} i_a \\ \omega_r \end{bmatrix} + \begin{bmatrix} \frac{1}{L} \\ 0 \end{bmatrix} V_a \tag{3.4}$$

$$= \mathrm{A}x + \mathrm{B}u$$

$$y = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} i_a \\ \omega_r \end{bmatrix} + [0] V_a \tag{3.5}$$

$$= \mathrm{C}x + \mathrm{D}u$$

By substituting the parameters on the state-space model, the value of A, B, C and D can be getting.

$$\begin{bmatrix} \frac{di_a}{dt} \\ \frac{d\omega_r}{dt} \end{bmatrix} = \begin{bmatrix} -\frac{2.7}{0.004} & -\frac{0.105}{0.004} \\ \frac{0.105}{0.0001} & -\frac{0.0000093}{0.0001} \end{bmatrix} \begin{bmatrix} i_a \\ \omega_r \end{bmatrix} + \begin{bmatrix} \frac{1}{0.004} \\ 0 \end{bmatrix} V_a$$

$$y = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} i_a \\ \omega_r \end{bmatrix} + [0] V_a$$

$$A = \begin{bmatrix} -675 & -26.25 \\ 1050 & -0.093 \end{bmatrix} \qquad B = \begin{bmatrix} 250 \\ 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 1 \end{bmatrix} \qquad D = \begin{bmatrix} 0 \end{bmatrix}$$

3.3 LQR Design

To meet the desired specification, the controller is designed using LQR methodology. Let the cost function be defined as

$$J = \int_0^\infty (x^t Q x + u^T R u)\, d t \quad (3.6)$$

Where Q is weighting factors of states (positive semidefinite matrix) and R is weighting factors of control variables (positive definite matrix). To design the LQR controller, the first step is to select the weighting matrices Q and R. The value R weight inputs more than the states while the value of Q weight the state more than the inputs. Then the feedback K can be computed and the closed loop system responses can be found by simulation.

The LQR controller is given by

$$u = - Kx \quad (3.7)$$

where K is the constant feedback gain obtained from the solution of the discrete algebraic Riccati equation. The gain matrix K which solve the LQR problem is

$$K = R^{-1} B^T P^* \quad (3.8)$$

Where P^* is unique, positive semidefinite solution to the Riccati equation;

$$A^T P + PA - PBR^{-1} B^T P + Q = 0 \quad (3.9)$$

Using the equation (3.9), the value of P is;

[0.0051 0.0036; 0.0036 0.0035]

Then, the value of K can be compute by using equation (3.8) which is

[1.2814 0.9002]

3.4 Coding in MATLAB

To create the open-loop response, the following command is entered in MATLAB;

```
r = 2.7;
L = 0.004;
J = 0.0001;
k = 0.105;
b = 0.0000093;
A = [-r/L -k/L; k/J -b/J];
B = [1/L; 0];
C = [0 1];
D = [0];
step (A, B, C, D)
```

The state feedback is designed by using nominal values of parameters and the matrices Q and R is choose in such a way that both the state response of displacements x_1, x_2 unit initial conditions in both and to a delayed unit step exhibit nearly critical damping.

The K is found to minimize the J involves solving the Riccati equation. The MATLAB ‘lqr’ command is used to directly solve the gain vector K given A, B, Q and R.

The coding for getting the simulation of the controller was as followed;

```
r = 2.7;
L = 0.004;
J = 0.0001;
k = 0.105;
b = 0.0000093;
A = [-r/L -k/L; k/J -b/J];
B = [1/L; 0];
C = [0 1];
D = [0];
Q = diag([1 1]);
R = 1;
[K,S,e] = lqr(A,B,Q,R);
Ac = A - B*K;
step(Ac,B,C,D,1,t)
```

In order to improve the design, the various Q matrices and R values were tried. For Q matrices, [1 0; 0 10] and [1 0; 100] were used and for R, 0.1 and 0.01 were applied.

3.5 Hardware Design

3.5.1 PIC Microcontroller

It is used to control the rotation of the motor. This senses the input and process it using the program burned in it and gives the required PWM output on the required port pins. To control the speed we need to control the duty cycle ('ON'TIME / PERIOD).

The output from the port A is given to the H-bridge. It is the arrangement of four transistors as shown below. Here four power transistors 2N3055 are used (C1, C2, C3, C4). At any time either C1, C3 or C2, C4 are made 'ON', hence current flows from the source (V_{dd}) to ground (Vss) through the motor. The direction of the motor is dependent on the polarity of the current. Hence by changing the 'ON' transistor pairs we can control the direction of the motor. To 'ON' the transistor we need to give high to its base. On controlling output coming out of port we can achieve the control over the motor rotation direction.

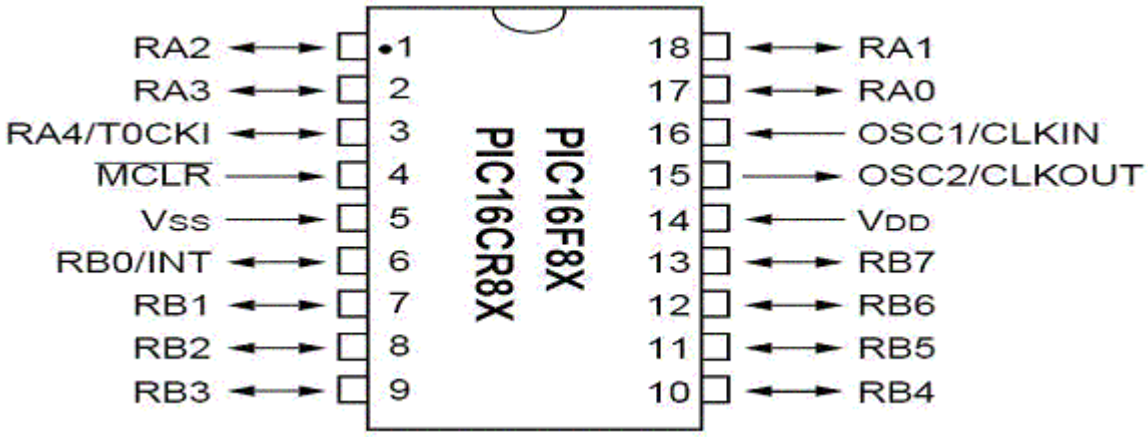


Figure 3.3 PIC16F84A IC Pin Diagram

3.5.2 Power Supply Circuit

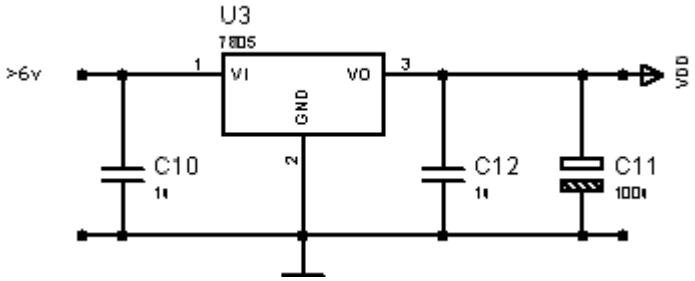


Figure 3.4 Power Supply Circuit

This circuit is very important as it was supply the power to the all component. Three terminal regulator is used to get the operating voltage for PIC. The left pin is for input voltage, the middle pin second is connected to ground while the right pin is 5V output voltage which connected to power the PIC.

3.5.3 Complete Circuit

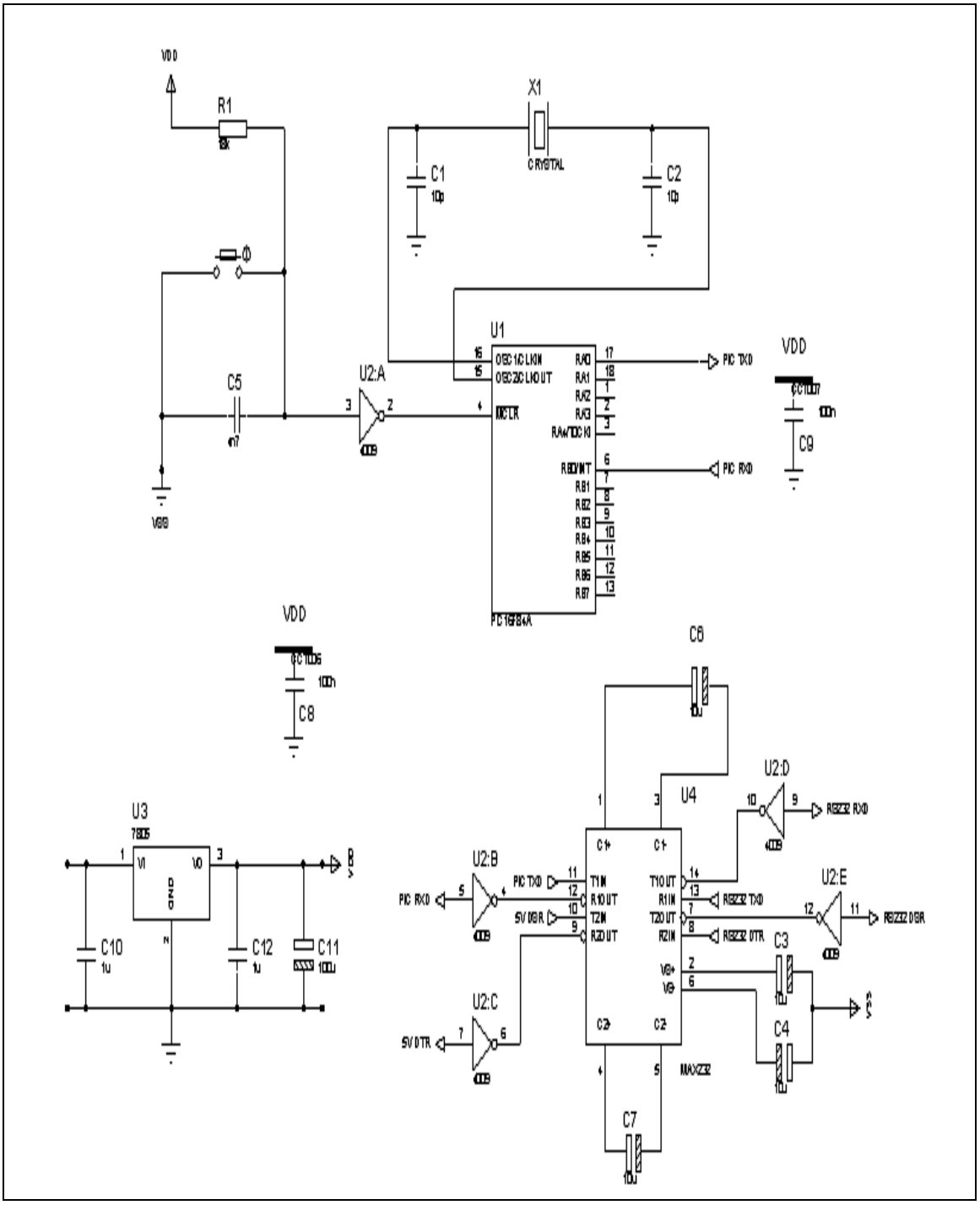


Figure 3.5 Complete Circuit Diagram

The circuit is built around the PIC IC 16F84A. The circuit is built as shown in Figure 3.5. The power supply of 5V and ground is given to appropriate pin of the PIC IC and to the H Bridge circuit. A 4M Hz crystal oscillator is connected as shown in the figure. Other type of oscillators can also be used. The crystal oscillator is more stable compare to other types. Oscillator acts as a clock source for the PIC IC operation.

The port B's four pins are given to the H-Bridge as shown in the figure. These pins are the control lines (PWM output lines) given to H-Bridge to rotate in the desired speed.

3.6 Software Development

3.6.1 Inside PIC16F84A

3.6.1.1 Flash Program Memory

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

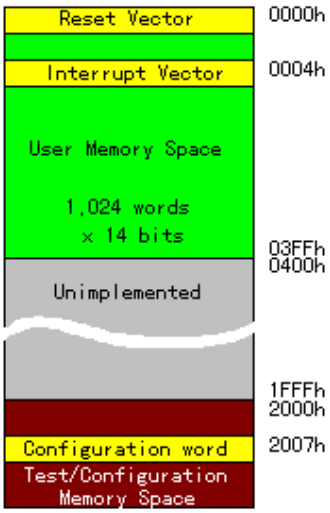


Figure 3.6 Program Flash Memory

The usage of some program memory addresses is already decided.

i. Reset Vector (0000h)

When a reset is executed, either by turning power on, by the WDT (Watchdog Timer) or any other factor, the program will start from this address.

ii. Peripheral Interrupt Vector (0004h)

When there is a time-out interruption from the timer (TMR0) or an outside interrupt, the program will start from this address.

iii. Configuration word (2007h)

The basic operation of the PIC is specified at this memory location. The enable bits of the Power-up timer, and the Watch-dog timer as well as the oscillator selection bits are set here. This area is behind the usual program area and can not be accessed by the program. These parameters must be specified using the burner when burning the program into flash memory.

3.6.1.2 SFR

16 different SFR (Special Function Registers) can be specified by the bank switching technique. The figure below shows the RAM File Registers. The memory capacity is only 160 bytes. The contents of the registers with the left pointing arrow are the same on both banks. The other registers of the SFR are accessible through bank switching, and the gray colored registers are not used.

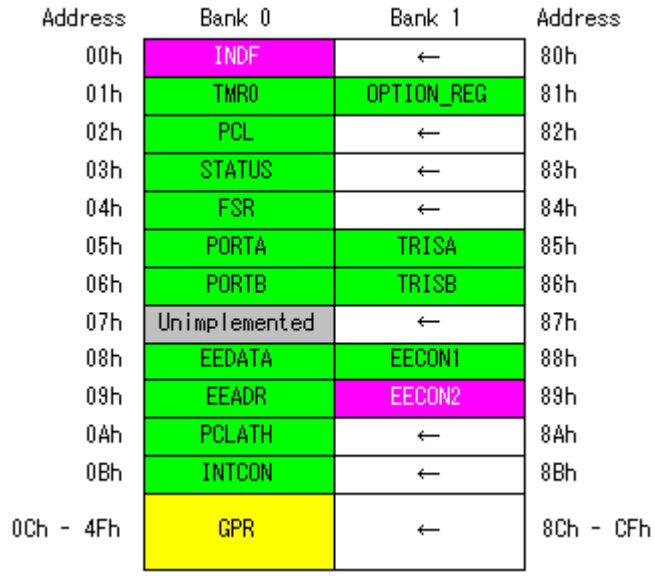


Figure 3.7 SFR Registers

Each SFR has the following function;

INDF : Data memory contents by indirect addressing

TMR0 : Timer counter

PCL : Low order 8 bits of program counter

STATUS : Flag of calculation result

FSR : Indirect data memory address pointer

PORTA : PORTA DATA I/O

PORTB : PORTB DATA I/O

EEDATA : Data for EEPROM

EEADR : Address for EEPROM

PCLATH : Write buffer for upper 5 bits of the program counter

INTCON : Interruption control

OPTIN_REG : Mode set

TRISA : Mode set for PORTA

TRISB : Mode set for PORTB

EECON1 : Control Register for EEPROM

EECON2 : Write protection Register for EEPROM

3.6.2 Programming on PIC16F84A

To interface the hardware with the electronic equipment, a software program that runs on a PC to develop applications for Microchip microcontrollers is required. Melabs EPIC™ Programmer software is used but we need to change the type of document by compiling it with the MicroCode Studio.

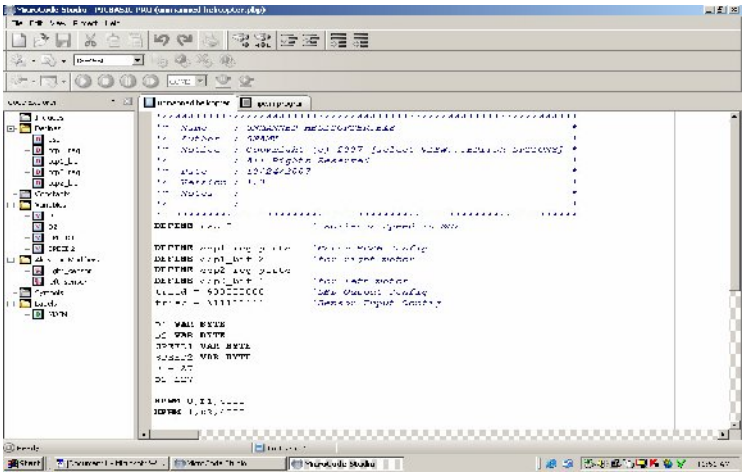


Figure 3.8 Programming of the system

Figure 3.8 shows the compilation of the programming of a system. The document must be save first before the compilation is done. After the compiling the documentation type will change to the .HEX file.

Select for PIC16F84A on the Melabs EPIC™ Programmer. The Melabs EPIC™ Programmer connects to a PC compatible parallel printer port. The Melabs Serial Programmer connects to a PC compatible serial port. The Melabs USB Programmer and Melabs U2 Programmer connect to a PC USB port or powered USB hub. Each programmer may be controlled by the Melabs Programmer software.

After the selection of the PIC type, setup the Melabs configuration for PLL applications as show in the Figure 3.9. The configuration must be filled correctly.

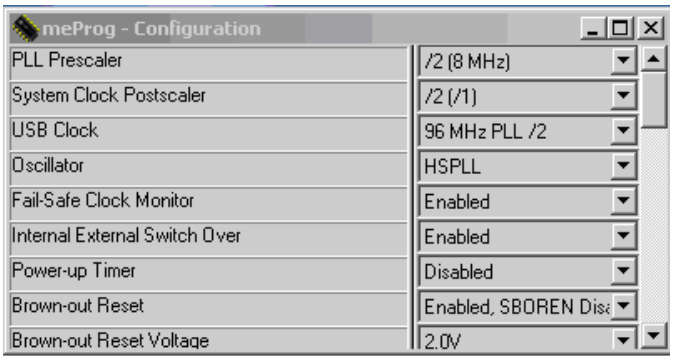


Figure 3.9 Setup for PLL

After the deleting the previous program in the PIC, select the programming that had been saving in .HEX type documentation. Verify the program and the PIC is ready to be used.

The example of the program from PIC by using MicroCode Studio programmer is shown as below. This program is to make the dc motor moving forward.

```
motor VAR portb.0

    trisb = 0
    trisa = 1


forward:   PULSOUT 0,170
           PULSOUT 1,130
           PAUSE 20
           GOTO forward
```

CHAPTER 4

RESULT AND DISCUSSION

This chapter will explain about how to get the results that have been made through the simulation in MATLAB software.

4.1 Simulation Result from MATLAB

The simulations below were get from the simulation in MATLAB. From these simulations, the value of Rise Time, Settling Time and Steady-State can be found. The Rise Time is the time required for the response to rise from 10% to 90% , 5% to 95% OR 0% to 100% of its required final value. For underdamped second-order system, the 10% to 90% rise time are normally used. The settling time is the time required for the response to reach and stay within a range about the final value of size specified by absolute percentage of the final value (usually 2% or 5%).

4.1.1 Open-loop simulation result

The step response shows that the motor achieve its maximum speed at 9.5 rad/s and it takes 0.091 seconds to reach its steady-state speech.

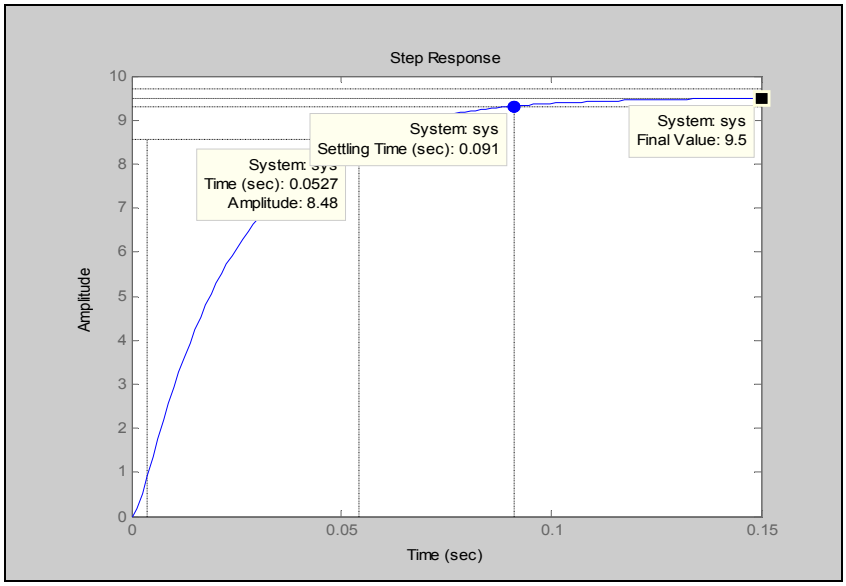


Figure 4.1 Open-Loop Step Response

4.1.1.1 Steady-state Error

Steady-state error is the difference between the input and the output for a prescribed test input as t → ∞. Unstable systems represent loss of control in the steady-state and are not acceptable for use at all. The expressions we derive to calculate the steady-state error can be applied erroneously to an unstable system.

The values of A, B and C are as get from the state-space in Chapter 3. There are;

$$A = \begin{bmatrix} -675 & -26.25 \\ 1050 & -0.093 \end{bmatrix} \qquad B = \begin{bmatrix} -250 \\ 0 \end{bmatrix} \qquad C = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

For unit step input, the steady-state error given by;

$$E(\infty) = 1 + CA^{-1}B \tag{4.1}$$

Therefore, the steady-state error is;

$$\begin{aligned} E(\infty) &= 1 + CA^{-1}B \\ &= 1 + \begin{bmatrix} 26.25 & 0.093 \end{bmatrix} \begin{bmatrix} 250 \\ 0 \end{bmatrix} \\ &= 1 + (6562.5 + 0) \\ &= 6563.5 \end{aligned}$$

For a ramp unit, the steady-state error is given by;

$$E(\infty) = \left[\lim \left(1 + CA^{-1}B \right) + \left(1 + C\left(A^{-1} \right)^2 B \right) \right] \tag{4.2}$$

Solving for $1 + C(A^{-1})^2B$;

$$\begin{aligned} 1 + C\left(A^{-1} \right)^2 B &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 428062.5 & -708847.65 \\ 17721.19 & -27562.49 \end{bmatrix} \begin{bmatrix} 250 \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} 17721.19 & -27562.49 \end{bmatrix} \begin{bmatrix} 250 \\ 0 \end{bmatrix} \\ &= 4430297.5 \end{aligned}$$

Then, the steady error is;

$$E(\infty) = \left[\lim t \to \infty \ \left(1 + CA^{-1}B\right)t\_ + \left(1 + C\left(A^{-1}\right)^2 B\right) \right]$$
$$= \left[\lim\left(6563.5\right)t + \left(4430297.5\right)\right]$$
$$= \infty$$

4.1.2 Closed-loop simulation result (applying LQR algorithm)

The step response shows that the motor achieve its maximum speed at 0.995 rad/s and it takes 0.0107 seconds to reach its steady-state speech.

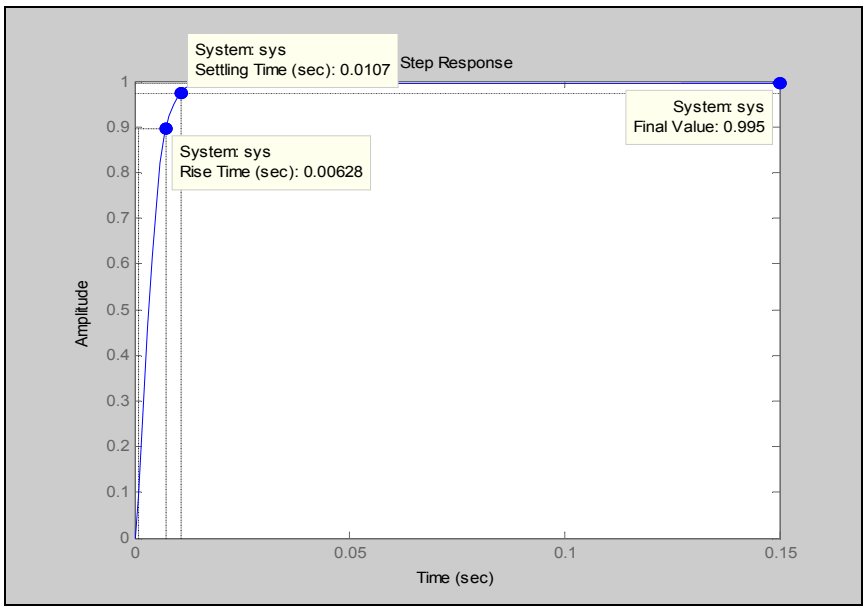


Figure 4.2 Closed-Loop Step Response

From these two simulations, it is shows that the closed loop system that was using the LQR controller is much better and makes the system more stable than the open-loop system.

4.1.3 Tuning the Q Value

These simulations is getting by tune the Q matrices where the value of R = 1 is maintained.

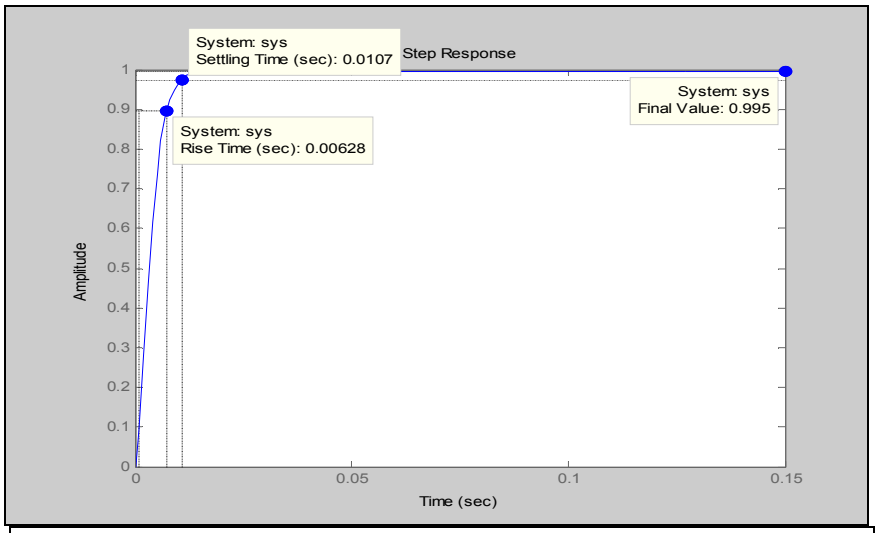


Figure 4.3 When Q = [1 0; 0 1] and R = 1

The step response shows that the motor achieve its maximum speed at 0.995 rad/s and it takes 0.0107 seconds to reach its steady-state speech.

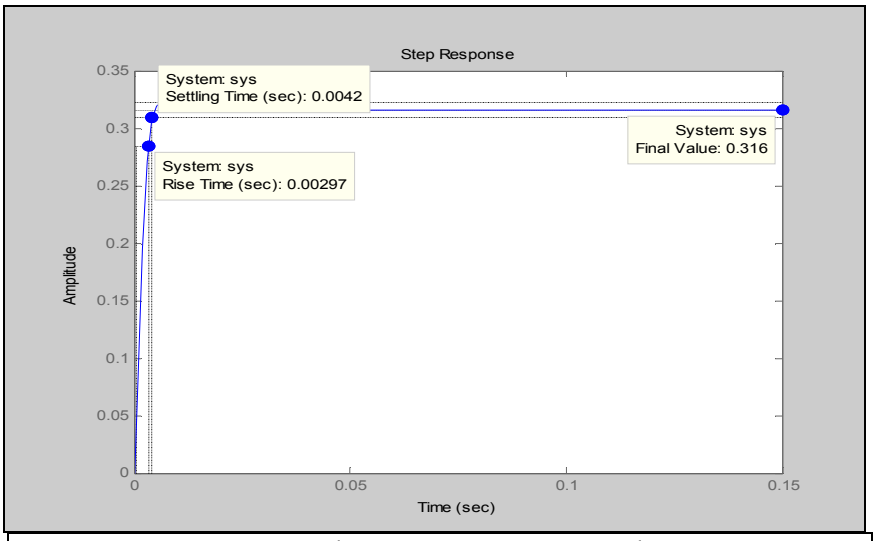


Figure 4.4 When Q = [1 0 ; 0 10] and R = 1

The step response shows that the motor achieve its maximum speed at 0.316 rad/s and it takes 0.0042 seconds to reach its steady-state speech.

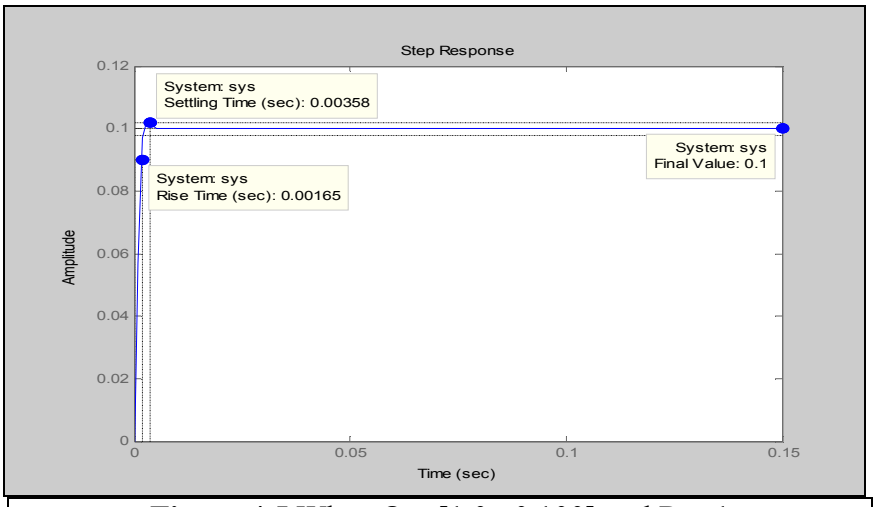


Figure 4.5 When Q = [1 0 ; 0 100] and R = 1

The step response shows that the motor achieve its maximum speed at 0.1 rad/s and it takes 0.00358 seconds to reach its steady-state speech.

The table below shows data analysis for the result of values of K that were get from the 'lqr' command while the value of Time Rise, Settling Time and Steady-State is getting from the result of simulation.

Table 4.1 The value of gain, K, Time Rise, Settling Time and Steady-State when the Q was tuned.

Q	R	K (Gain)	Time Rise	Settling Time	Steady State
[1 0 ; 0 1]	1	K = [1.2814 0.9002]	0.00628	0.0107	0.995
[1 0 ; 0 10]	1	K = [3.1294 3.0585]	0.00297	0.0042	0.316
[1 0 ; 0 100]	1	K = [6.8606 9.8947]	0.00165	0.00358	0.1

From the table, we can see that the Settling Time and Rise Time become faster as increasing of the value of Q. This show the system is become more stable and the deviation of the speed of the dc motor is minimized.

4.1.3 Tuning the R Value

These simulations is getting by tune the R value where the value of the Q matrices equal to [1 0; 0 1] is maintained.

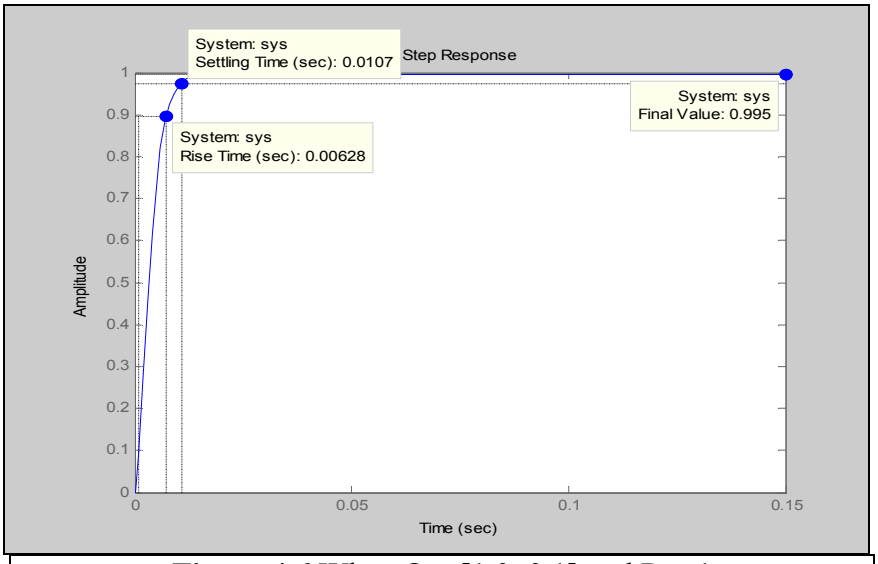


Figure 4.6 When Q = [1 0; 0 1] and R = 1

The step response shows that the motor achieve its maximum speed at 0.995 rad/s and it takes 0.0107 seconds to reach its steady-state speech.

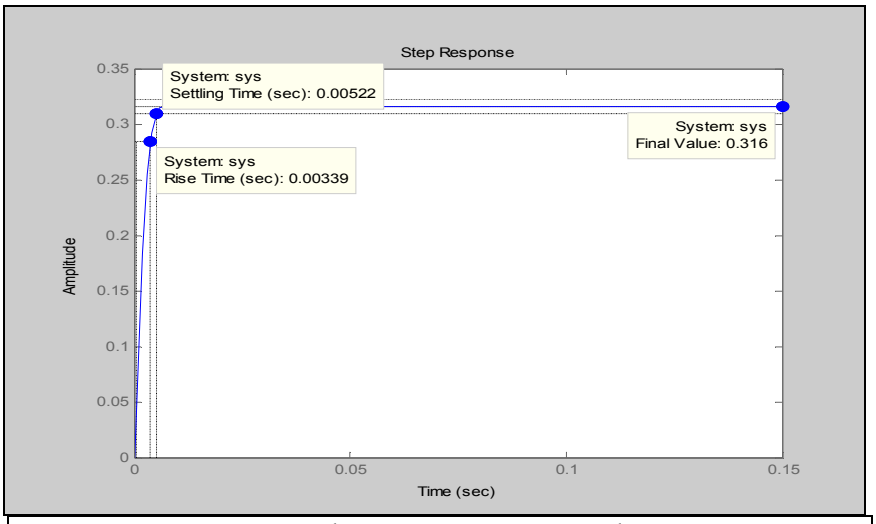


Figure 4.7 When Q = [1 0 ; 0 1] and R = 0.1

The step response shows that the motor achieve its maximum speed at 0.316 rad/s and it takes 0.00522 seconds to reach its steady-state speech.

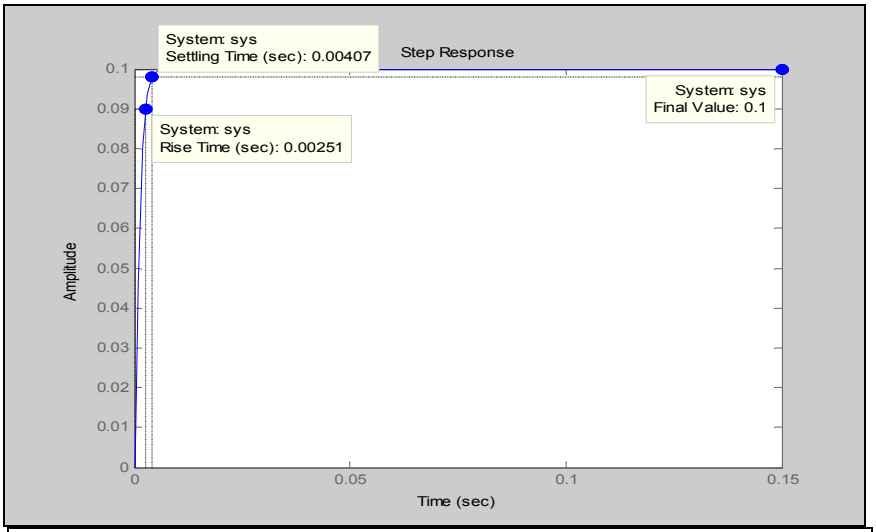


Figure 4.8 When Q = [1 0 ; 0 1] and R = 0.01

The step response shows that the motor achieve its maximum speed at 0.1 rad/s and it takes 0.00407 seconds to reach its steady-state speech.

Table 4.2 The value of gain, K, Time Rise, Settling Time and Steady-State when R was tuned.

Q	R	K (Gain)	Time Rise(s)	Settling Time(s)	Steady State
[1 0 ; 0 1]	1	K = [1.2814 0.9002]	0.00628	0.0107	0.995
[1 0 ; 0 1]	0.1	K = [3.8560 3.0584]	0.00339	0.00522	0.316
[1 0 ; 0 1]	0.01	K = [11.0986 9.8943]	0.00251	0.00407	0.1

From the table, we can see that the Settling Time and Rise Time become faster as decreasing the value of R. This show the system is become more stable and the deviation of the speed of the dc motor is minimized.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

This chapter will explain about the overall project that has been made and future recommendation to improve the study and make this project more valuable and effective.

5.1 Conclusion

The purpose of using Linear Quadratic Regulator (LQR) algorithm is to minimize the deviation or the error in a system. In DC motor control, the LQR is applied to make sure the speed of the motor is always be maintained.

By the simulation that have been made by using MATLAB software, this algorithm is approved can reduce the error of the dc motor speed. But, for the implementation on the PIC microcontroller, it did not success since there is the problem in making the programming for the LQR algorithm.

5.2 Recommendation

Some recommendation may apply in this system to make it more reliable and useful especially in control system field. The recommendations discussed here are focused more on how to improve the system to be much better than what have been done in this study.

Below are several recommendations which are suitable to apply in the existing system.

i. Sliding mode control has been applied to many engineering fields due to excellent robustness. So, in order to posses the optimal performance and robustness against model uncertainty and external disturbance of the dc motor, the LQR algorithm can be combined with this controller.

ii. Use another type of PIC microcontroller which has more features. For example, PIC 16F873 which has CCP (Capture/Compare/PWM) feature that can be used to control the motor drive.

iii. Built LED displaying circuit to monitor the drive situation of the motor.

5.3 Costing & Commercialization

5.3.1 Costing

This part explains about the costing of this project. The total project cost for all components is estimated to be RM 3089.58. The highest cost is reflected in the price of Clifton Precision Servo Motor Model JDH-2250-HF-2C-E. Even though the price of these components is expensive, it is still a necessary item. The component chosen based on the performance of the component, means that the chosen component rating is above designed value. The table 4.3 shows the cost that takes for the overall project.

Table 4.3 Approximation Cost for the Project

No	Components	Specifications	Price / unit	Quantity	Estimation Cost
1	PIC	16F84A	RM 25.00	1	RM 25.00
2	IC base	18 pin	RM 0.20	1	RM 0.20
3	IC Base	16 pin	RM 0.18	1	RM 0.18
4	Capacitor	10pF	RM 0.08	2	RM 0.16
5	Capacitor	4.7nF	RM 0.08	1	RM 0.08
6	Capacitor	100nF	RM 0.10	2	RM 0.20
7	Capacitor	10uF	RM 0.12	4	RM 0.48
8	Capacitor	1uF	RM 0.12	2	RM 0.24
9	Capacitor	100uF	RM 0.20	1	RM 0.20
10	Resistor	16KΩ	RM 0.04	1	RM 0.04
11	Header		RM 0.80	10	RM 8.00
12	Heat Sink		RM0.70	1	RM0.70
13	Regulator 7805		RM 2.00	1	RM 2.00
14	Reset switch		RM 0.60	1	RM 0.60
15	Crystal	4MHz	RM 1.90	1	RM 1.90
16	Wire Wrap		RM 40.00	1	RM 40.00
17	Wrap Tool		-	1	-
18	RS232		RM 0.60	1	RM 0.60
19	MAX232		RM 4.00	1	RM 4.00
20	Strip Board		RM 5.00	1	RM 5.00
21	DC servomotor	JDH-2250-HF-2C-E	RM 3000	1	RM 3000
				TOTAL	**RM 3089.58**

5.3.2 Commercialized

This project has the ability to be commercialized as it function is to stabilize the speed of the dc motor. So, the effectiveness of the system that used the dc servomotor can be increase.

Besides, this project can be the reference for the further research. The improvement may be able to be done to make the better system.

REFERENCES

[1] Huang, H. W. (2004). *PIC Microcontroller: An Introduction to Software And Hardware Interfacing*. Thomson Delmar Learning,

[2] 9 March 2008 source URL http://www.electrofriends.com/

[3] 13 February 2000 source URL http://en.wikipedia.org/wiki/Linear-quadratic_regulator

[4] Auslaunder, D. M.(2002) *Feedback Control Using Computer*. University of Michigan, Ann Arbor, Michigan.

[5] Rovner, K. M. *(*1987), *Experiment In Adaptive Control Of a Very Flexible One Link Manipulator*, Ph.D., Stanford University.

[6] EE 4343/5329 - *Control System Design Project* available at http://arri.uta.edu/acs

[7] Passino, K. M. and Quijano N. (2002). *Linear Quadratic Regulator and Observer Design for a Flexible Joint Dept. Electrical Engineering*, The Ohio State University.

[8] 23 February 2008 source URL
http://www.interq.or.jp/

[9] Nise, N. S. (2004). *Control System Engineering*. 4th Edition. River Street, Hoboken: John Wiley & Sons, Inc.

[10] Ibrabim D. (2006). *PIC Basic Project*. Burlington: Elsevier's Science & Technology.

APPENDIX A

PIC16F84A Datasheet

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:

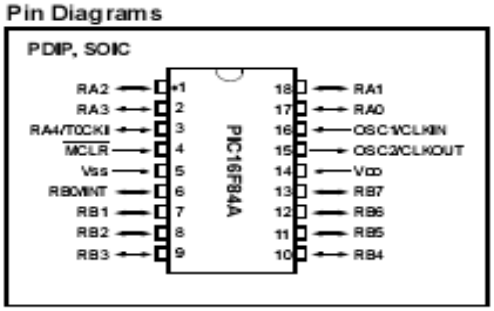
- 13 I/O pins with individual direction control
- High current sink/source for direct LED drive
  - 25 mA sink max. per pin
  - 25 mA source max. per pin
- TMR0: 8-bit timer/counter with 8-bit programmable prescaler

Special Microcontroller Features:

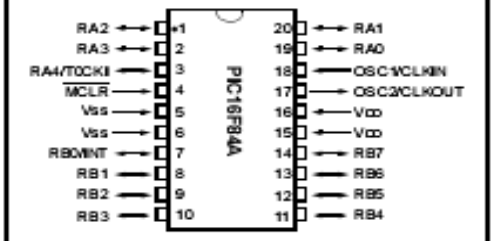
- 10,000 erase/write cycles *Enhanced* FLASH Program memory typical
- 10,000,000 typical erase/write cycles EEPROM Data memory typical
- EEPROM Data Retention > 40 years
- In-Circuit Serial Programming™ (ICSP™) - via two pins
- Power-on Reset (POR), Power-up Timer (PWRT), Oscillator Start-up Timer (OST)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

Pin Diagrams

PDIP, SOIC



SSOP



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

PIC16F84A

TABLE 1-1: PIC16F84A PINOUT DESCRIPTION

Pin Name	PDIP No.	SOIC No.	SSOP No.	I/O/P Type	Buffer Type	Description
OSC1/CLKIN	16	16	18	I	ST/CMOS(3)	Oscillator crystal input/external clock source input.
OSC2/CLKOUT	15	15	19	O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode. In RC mode, OSC2 pin outputs CLKOUT, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
$\overline{MCLR}$	4	4	4	I/P	ST	Master Clear (Reset) input/programming voltage input. This pin is an active low RESET to the device.
						PORTA is a bi-directional I/O port.
RA0	17	17	19	I/O	TTL	
RA1	18	18	20	I/O	TTL	
RA2	1	1	1	I/O	TTL	
RA3	2	2	2	I/O	TTL	
RA4/T0CKI	3	3	3	I/O	ST	Can also be selected to be the clock input to the TMR0 timer/counter. Output is open drain type.
						PORTB is a bi-directional I/O port. PORTB can be software programmed for internal weak pull-up on all inputs.
RB0/INT	6	6	7	I/O	TTL/ST(1)	RB0/INT can also be selected as an external interrupt pin.
RB1	7	7	8	I/O	TTL	
RB2	8	8	9	I/O	TTL	
RB3	9	9	10	I/O	TTL	
RB4	10	10	11	I/O	TTL	Interrupt-on-change pin.
RB5	11	11	12	I/O	TTL	Interrupt-on-change pin.
RB6	12	12	13	I/O	TTL/ST(2)	Interrupt-on-change pin. Serial programming clock.
RB7	13	13	14	I/O	TTL/ST(2)	Interrupt-on-change pin. Serial programming data.
Vss	5	5	5,6	P	—	Ground reference for logic and I/O pins.
VDD	14	14	15,16	P	—	Positive supply for logic and I/O pins.

Legend: I= input O = Output I/O = Input/Output P = Power
— = Not used TTL = TTL input ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured in RC oscillator mode and a CMOS input otherwise.

2.0 MEMORY ORGANIZATION

There are two memory blocks in the PIC16F84A. These are the program memory and the data memory. Each block has its own bus, so that access to each block can occur during the same oscillator cycle.

The data memory can further be broken down into the general purpose RAM and the Special Function Registers (SFRs). The operation of the SFRs that control the "core" are described here. The SFRs used to control the peripheral modules are described in the section discussing each individual peripheral module.

The data memory area also contains the data EEPROM memory. This memory is not directly mapped into the data memory, but is indirectly mapped. That is, an indirect address pointer specifies the address of the data EEPROM memory to read/write. The 64 bytes of data EEPROM memory have the address range 0h-3Fh. More details on the EEPROM memory can be found in Section 3.0.

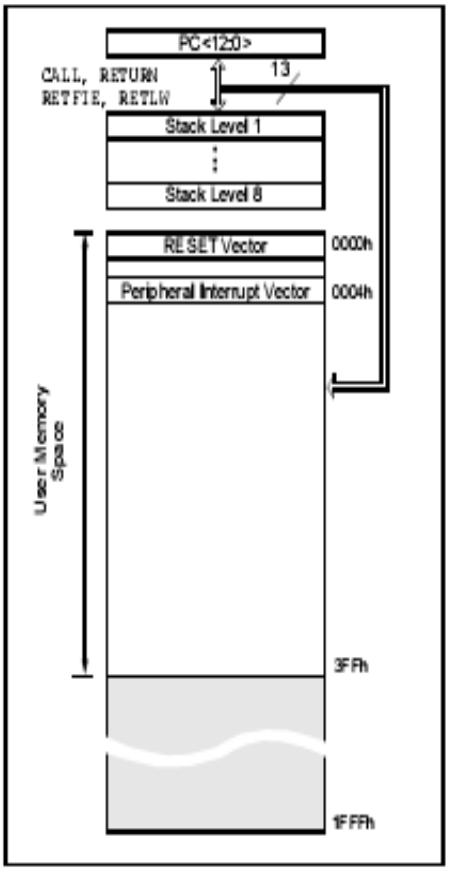
Additional information on device memory may be found in the PICmicro™ Mid-Range Reference Manual, (DS33023).

2.1 Program Memory Organization

The PIC16FXX has a 13-bit program counter capable of addressing an 8K x 14 program memory space. For the PIC16F84A, the first 1K x 14 (0000h-03FFh) are physically implemented (Figure 2-1). Accessing a location above the physically implemented address will cause a wraparound. For example, for locations 20h, 420h, 820h, C20h, 1020h, 1420h, 1820h, and 1C20h, the instruction will be the same.

The RESET vector is at 0000h and the interrupt vector is at 0004h.

FIGURE 2-1: PROGRAM MEMORY MAP AND STACK - PIC16F84A



2.2 Data Memory Organization

The data memory is partitioned into two areas. The first is the Special Function Registers (SFR) area, while the second is the General Purpose Registers (GPR) area. The SFRs control the operation of the device.

Portions of data memory are banked. This is for both the SFR area and the GPR area. The GPR area is banked to allow greater than 116 bytes of general purpose RAM. The banked areas of the SFR are for the registers that control the peripheral functions. Banking requires the use of control bits for bank selection. These control bits are located in the STATUS Register. Figure 2-2 shows the data memory map organization.

Instructions MOVWF and MOVF can move values from the W register to any location in the register file ("F"), and vice-versa.

The entire data memory can be accessed either directly using the absolute address of each register file or indirectly through the File Select Register (FSR) (Section 2.5). Indirect addressing uses the present value of the RP0 bit for access into the banked areas of data memory.

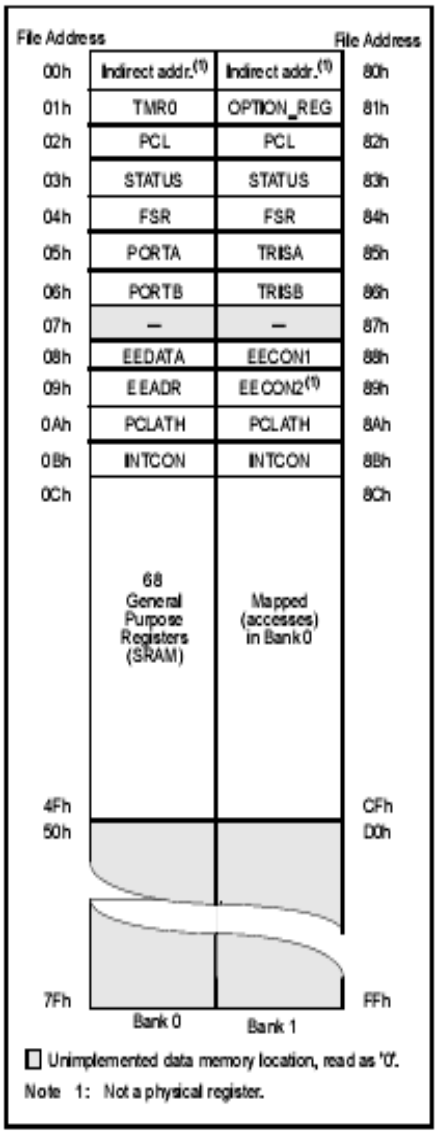
Data memory is partitioned into two banks which contain the general purpose registers and the special function registers. Bank 0 is selected by clearing the RP0 bit (STATUS<5>). Setting the RP0 bit selects Bank 1. Each Bank extends up to 7Fh (128 bytes). The first twelve locations of each Bank are reserved for the Special Function Registers. The remainder are General Purpose Registers, implemented as static RAM.

2.2.1 GENERAL PURPOSE REGISTER FILE

Each General Purpose Register (GPR) is 8-bits wide and is accessed either directly or indirectly through the FSR (Section 2.5).

The GPR addresses in Bank 1 are mapped to addresses in Bank 0. As an example, addressing location 0Ch or 8Ch will access the same GPR.

FIGURE 2-2: REGISTER FILE MAP - PIC16F84A



2.3 Special Function Registers

The Special Function Registers (Figure 2-2 and Table 2-1) are used by the CPU and Peripheral functions to control the device operation. These registers are static RAM.

The special function registers can be classified into two sets, core and peripheral. Those associated with the core functions are described in this section. Those related to the operation of the peripheral features are described in the section for that specific feature.

TABLE 2-1: SPECIAL FUNCTION REGISTER FILE SUMMARY

Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on RESET	Details on page
Bank 0											
00h	INDF	Uses contents of FSR to address Data Memory (not a physical register)								---- ----	11
01h	TMR0	8-bit Real-Time Clock/Counter								xxxx xxxx	20
02h	PCL	Low Order 8 bits of the Program Counter (PC)								0000 0000	11
03h	STATUS(2)	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	8
04h	FSR	Indirect Data Memory Address Pointer 0								xxxx xxxx	11
05h	PORTA(4)	—	—	—	RA4/T0CKI	RA3	RA2	RA1	RA0	---x xxxx	16
06h	PORTB(5)	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0/INT	xxxx xxxx	18
07h	—	Unimplemented location, read as '0'								—	—
08h	EEDATA	EEPROM Data Register								xxxx xxxx	13,14
09h	EEADR	EEPROM Address Register								xxxx xxxx	13,14
0Ah	PCLATH	—	—	—	Write Buffer for upper 5 bits of the PC(1)					---0 0000	11
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	10
Bank 1											
80h	INDF	Uses Contents of FSR to address Data Memory (not a physical register)								---- ----	11
81h	OPTION_REG	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	9
82h	PCL	Low order 8 bits of Program Counter (PC)								0000 0000	11
83h	STATUS (2)	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	8
84h	FSR	Indirect data memory address pointer 0								xxxx xxxx	11
85h	TRISA	—	—	—	PORTA Data Direction Register					---1 1111	16
86h	TRISB	PORTB Data Direction Register								1111 1111	18
87h	—	Unimplemented location, read as '0'								—	—
88h	EECON1	—	—	—	EEIF	WRERR	WREN	WR	RD	---0 x000	13
89h	EECON2	EEPROM Control Register 2 (not a physical register)								---- ----	14
0Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of the PC(1)					---0 0000	11
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	10

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0', q = value depends on condition

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a slave register for PC<12:8>. The contents of PCLATH can be transferred to the upper byte of the program counter, but the contents of PC<12:8> are never transferred to PCLATH.

2: The $\overline{TO}$ and $\overline{PD}$ status bits in the STATUS register are not affected by a $\overline{MCLR}$ Reset.

3: Other (non power-up) RESETS include: external RESET through $\overline{MCLR}$ and the Watchdog Timer Reset.

4: On any device RESET, these pins are configured as inputs.

5: This is the value that will be in the port output latch.

3.0 DATA EEPROM MEMORY

The EEPROM data memory is readable and writable during normal operation (full VDD range). This memory is not directly mapped in the register file space. Instead it is indirectly addressed through the Special Function Registers. There are four SFRs used to read and write this memory. These registers are:

- EECON1
- EECON2 (not a physically implemented register)
- EEDATA
- EEADR

EEDATA holds the 8-bit data for read/write, and EEADR holds the address of the EEPROM location being accessed. PIC16F84A devices have 64 bytes of data EEPROM with an address range from 0h to 3Fh.

The EEPROM data memory allows byte read and write. A byte write automatically erases the location and writes the new data (erase before write). The EEPROM data memory is rated for high erase/write cycles. The write time is controlled by an on-chip timer. The write-time will vary with voltage and temperature as well as from chip to chip. Please refer to AC specifications for exact limits.

When the device is code protected, the CPU may continue to read and write the data EEPROM memory. The device programmer can no longer access this memory.

Additional information on the Data EEPROM is available in the PICmicro™ Mid-Range Reference Manual (DS33023).

REGISTER 3-1: EECON1 REGISTER (ADDRESS 88h)

U-0	U-0	U-0	R/W-0	R/W-x	R/W-0	R/S-0	R/S-0
—	—	—	EEIF	WRERR	WREN	WR	RD
bit 7							bit 0

bit 7-5 Unimplemented: Read as '0'

bit 4 EEIF: EEPROM Write Operation Interrupt Flag bit
1 = The write operation completed (must be cleared in software)
0 = The write operation is not complete or has not been started

bit 3 WRERR: EEPROM Error Flag bit
1 = A write operation is prematurely terminated
(any $\overline{\text{MCLR}}$ Reset or any WDT Reset during normal operation)
0 = The write operation completed

bit 2 WREN: EEPROM Write Enable bit
1 = Allows write cycles
0 = Inhibits write to the EEPROM

bit 1 WR: Write Control bit
1 = Initiates a write cycle. The bit is cleared by hardware once write is complete. The WR bit can only be set (not cleared) in software.
0 = Write cycle to the EEPROM is complete

bit 0 RD: Read Control bit
1 = Initiates an EEPROM read RD is cleared in hardware. The RD bit can only be set (not cleared) in software.
0 = Does not initiate an EEPROM read

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

4.0 I/O PORTS

Some pins for these I/O ports are multiplexed with an alternate function for the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

Additional information on I/O ports may be found in the PICmicro™ Mid-Range Reference Manual (DS33023).

4.1 PORTA and TRISA Registers

PORTA is a 5-bit wide, bi-directional port. The corresponding data direction register is TRISA. Setting a TRISA bit (= 1) will make the corresponding PORTA pin an input (i.e., put the corresponding output driver in a Hi-Impedance mode). Clearing a TRISA bit (= 0) will make the corresponding PORTA pin an output (i.e., put the contents of the output latch on the selected pin).

Note:	On a Power-on Reset, these pins are configured as inputs and read as '0'.

Reading the PORTA register reads the status of the pins, whereas writing to it will write to the port latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read. This value is modified and then written to the port data latch.

Pin RA4 is multiplexed with the Timer0 module clock input to become the RA4/T0CKI pin. The RA4/T0CKI pin is a Schmitt Trigger input and an open drain output. All other RA port pins have TTL input levels and full CMOS output drivers.

EXAMPLE 4-1: INITIALIZING PORTA

```
BCF     STATUS, RP0 ;
CLRF    PORTA       ; Initialize PORTA by
                    ; clearing output
                    ; data latches
BSF     STATUS, RP0 ; Select Bank 1
MOVLW   0x0F        ; Value used to
                    ; initialize data
                    ; direction
MOVWF   TRISA       ; Set RA<3:0> as inputs
                    ; RA4 as output
                    ; TRISA<7:5> are always
                    ; read as '0'.
```

FIGURE 4-1: BLOCK DIAGRAM OF PINS RA3:RA0

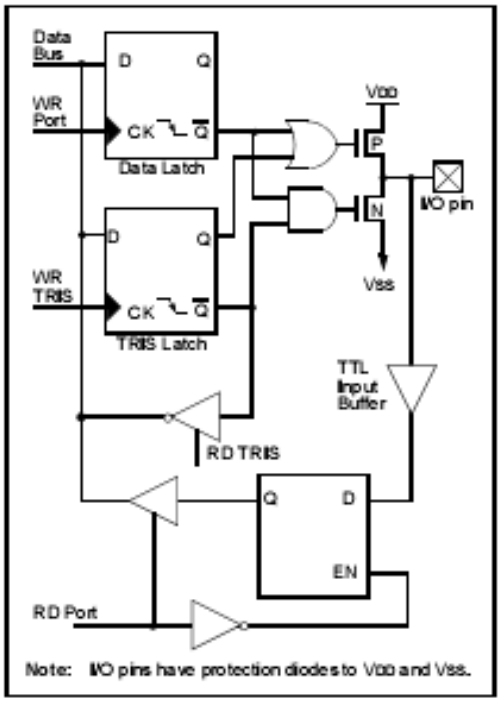


FIGURE 4-2: BLOCK DIAGRAM OF PIN RA4

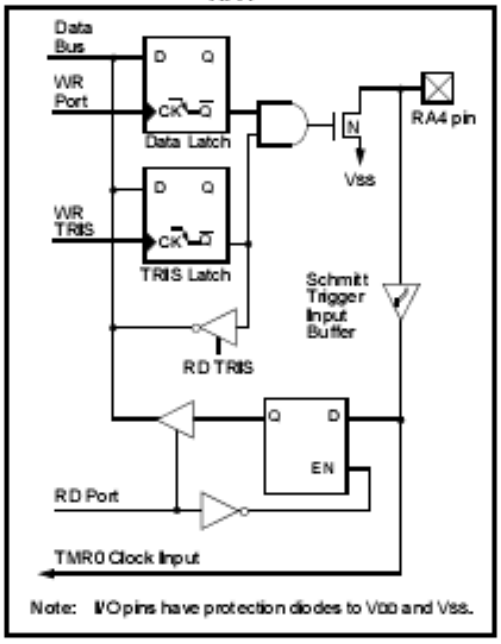


TABLE 4-1: PORTA FUNCTIONS

Name	Bit0	Buffer Type	Function
RA0	bit0	TTL	Input/output
RA1	bit1	TTL	Input/output
RA2	bit2	TTL	Input/output
RA3	bit3	TTL	Input/output
RA4/T0CKI	bit4	ST	Input/output or external clock input for TMR0. Output is open drain type.

Legend: TTL = TTL input, ST = Schmitt Trigger input

TABLE 4-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
05h	PORTA	–	–	–	RA4/T0CKI	RA3	RA2	RA1	RA0	---x xxxx	---u uuuu
85h	TRISA	–	–	–	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	---1 1111	---1 1111

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are unimplemented, read as '0'.

4.2 PORTB and TRISB Registers

PORTB is an 8-bit wide, bi-directional port. The corresponding data direction register is TRISB. Setting a TRISB bit (= 1) will make the corresponding PORTB pin an input (i.e., put the corresponding output driver in a Hi-Impedance mode). Clearing a TRISB bit (= 0) will make the corresponding PORTB pin an output (i.e., put the contents of the output latch on the selected pin).

EXAMPLE 4-2: INITIALIZING PORTB

```
BCF     STATUS, RP0 ;
CLRF    PORTB       ; Initialize PORTB by
                    ; clearing output
                    ; data latches
BSF     STATUS, RP0 ; Select Bank 1
MOVLW   0xCF        ; Value used to
                    ; initialize data
                    ; direction
MOVWF   TRISB       ; Set RB<3:0> as inputs
                    ; RB<5:4> as outputs
                    ; RB<7:6> as inputs
```

Each of the PORTB pins has a weak internal pull-up. A single control bit can turn on all the pull-ups. This is performed by clearing bit $\overline{\text{RBPU}}$ (OPTION<7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on a Power-on Reset.

Four of PORTB's pins, RB7:RB4, have an interrupt-on-change feature. Only pins configured as inputs can cause this interrupt to occur (i.e., any RB7:RB4 pin configured as an output is excluded from the interrupt-on-change comparison). The input pins (of RB7:RB4) are compared with the old value latched on the last read of PORTB. The "mismatch" outputs of RB7:RB4 are OR'ed together to generate the RB Port Change Interrupt with flag bit RBIF (INTCON<0>).

This interrupt can wake the device from SLEEP. The user, in the Interrupt Service Routine, can clear the interrupt in the following manner:

a) Any read or write of PORTB. This will end the mismatch condition.

b) Clear flag bit RBIF.

A mismatch condition will continue to set flag bit RBIF. Reading PORTB will end the mismatch condition and allow flag bit RBIF to be cleared.

The interrupt-on-change feature is recommended for wake-up on key depression operation and operations where PORTB is only used for the interrupt-on-change feature. Polling of PORTB is not recommended while using the interrupt-on-change feature.

FIGURE 4-3: BLOCK DIAGRAM OF PINS RB7:RB4

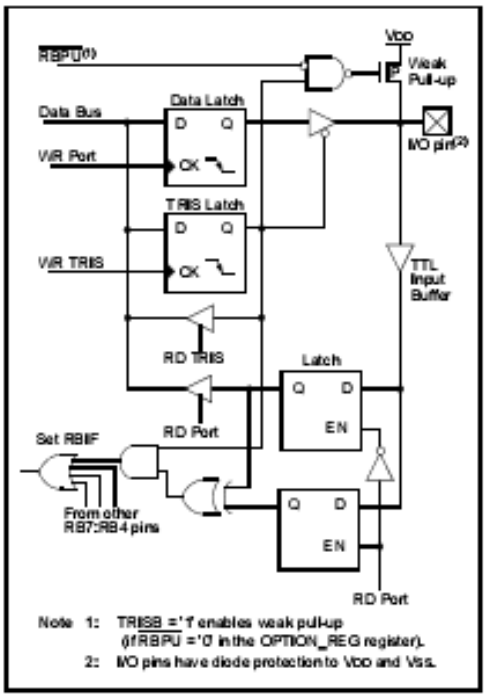


FIGURE 4-4: BLOCK DIAGRAM OF PINS RB3:RB0

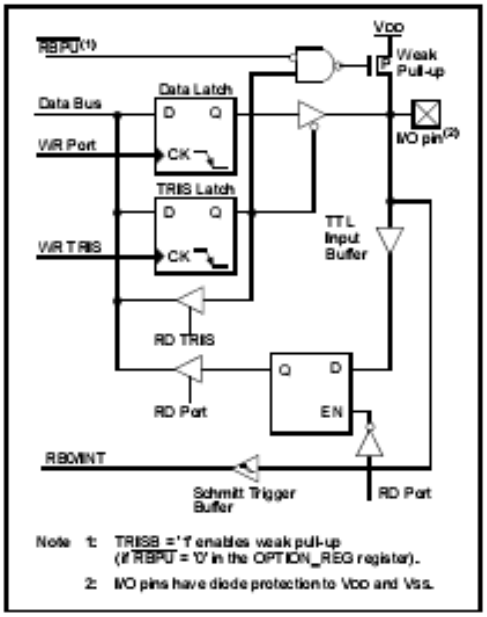


TABLE 4-3: PORTB FUNCTIONS

Name	Bit	Buffer Type	I/O Consistency Function
RB0/INT	bit0	TTL/ST(1)	Input/output pin or external interrupt input. Internal software programmable weak pull-up.
RB1	bit1	TTL	Input/output pin. Internal software programmable weak pull-up.
RB2	bit2	TTL	Input/output pin. Internal software programmable weak pull-up.
RB3	bit3	TTL	Input/output pin. Internal software programmable weak pull-up.
RB4	bit4	TTL	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up.
RB5	bit5	TTL	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up.
RB6	bit6	TTL/ST(2)	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up. Serial programming clock.
RB7	bit7	TTL/ST(2)	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up. Serial programming data.

Legend: TTL = TTL input, ST = Schmitt Trigger.
Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.

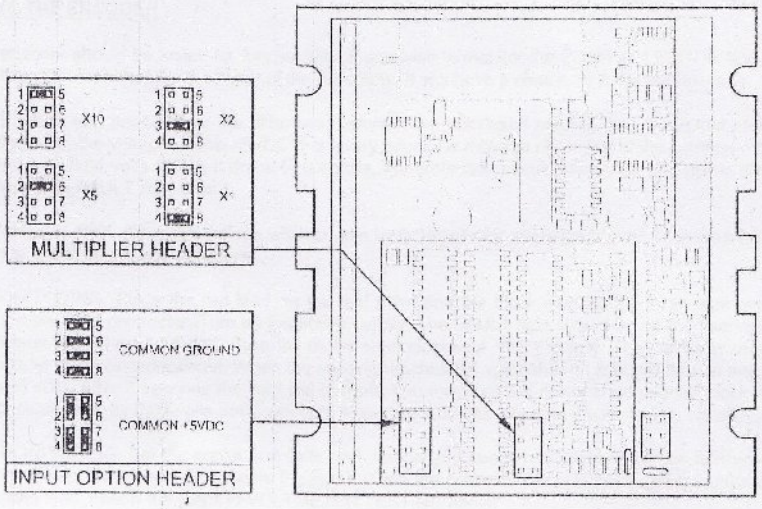
TABLE 4-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other RESETS
06h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0/INT	xxxx xxxx	uuuu uuuu
86h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	1111 1111
81h	OPTION_REG	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
0Bh,8Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u

Legend: x = unknown, u = unchanged. Shaded cells are not used by PORTB.

APPENDIX B

Clifton Precision Servo Motor Model JDH-2250-HF-2C-E Datasheet



G340 TERMINAL WIRING:

IMPORTANT: When first testing the G320, connect ERR/RES (term. 5) to ENC+ (term. 7).
Please follow the next steps in the sequence they are given.

STEP 1: ENCODER HOOK-UP

The encoder must be at minimum a 25 line-count digital quadrature encoder and must operate on a single +5VDC power supply. If the encoder supply current is more than 50 mA, use an external +5VDC supply. It may have an INDEX output, which will not be used. If it has differential outputs, use only the "+" phase outputs. **IMPORTANT: Connect a 470-ohm resistor from TERM. 6 to TERM. 7 if an external power supply is used for the encoder.**

(TERM. 6) ENC- Connect the encoder power supply ground to this terminal.
(TERM. 7) ENC+ Connect the encoder +5VDC to this terminal
(TERM. 8) PHASE A Connect the encoder phase "A" to this terminal
(TERM. 9) PHASE B Connect the encoder phase "B" to this terminal

To determine the optimal encoder line count, please follow the instructions below.

1.) Determine motors no load RPM
2.) Calculate rated RPM as 80% of no load RPM
3.) Divide (#2) by 60 to get revolutions per second
4.) Determine the CNC program's maximum step pulse frequency (in Hz)
5.) Divide (#4) by (#3), which will give you the maximum counts per revolution
6.) Divide (#5) by 4, which will give you the max line count
7.) Pick the first standard line count below (#6)

An example of using that formula with a 45kHz step pulse frequency and a maximum motor RPM of 3000:

(45kHz / 40) / 4 = 281.25

STEP 2: POWER SUPPLY HOOK-UP

CAUTION! Never put a switch on the DC side of the power supply! This will damage, if not destroy, your drive!

Keep the power supply leads short and use the largest wire gauge that will fit in the terminals. If the lead length is more than 18" use a 1000 uF capacitor across the G340 power supply terminals. Make sure your power supply can provide the peak current the motor may draw. The power supply voltage must be between 18 VDC and 80 VDC. The actual voltage should not be more than 5 volts higher than the motor's rated voltage.

(TERM. 1) POWER GROUND Connect the power supply ground to this terminal.
(TERM. 2) +18 TO 80 VDC Connect the power supply "+" to this terminal

STEP 3: TESTING THE ENCODER

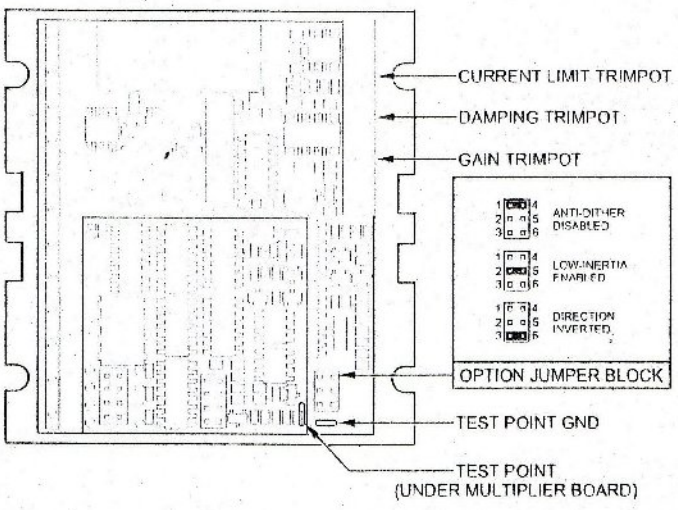
At this point the encoder should be tested for functionality. If you wish to monitor the POSITION ERROR test point with a voltmeter or oscilloscope, then remove the cover of the drive now. If you have a choice, pick the oscilloscope.

The POSITION ERROR test point shows the difference between the command position and the actual motor position. When both are the same, the voltage will be +5VDC. For every count the motor is clockwise of the command position, the voltage will decrease by 0.04 volts. When it drops to 0.4 volts, the protection circuit takes over and resets the drive for 3 seconds. While reset, the FAULT light is on.

For counter-clockwise position errors the voltage will increase by 0.04 volts for every count until it reaches 9.6 volts when again the protection circuit takes over as before.

VOLTMETER MONITORING: Place the red lead on the test point and the black lead on the large blue capacitor lead (GND) furthest from the main connector. Turn on the power supply. The FAULT light should be on for 3 seconds and then turn off. The voltmeter should read +5VDC. Turn the motor shaft clockwise VERY slowly. The voltmeter reading should decrease 0.04 volts for every encoder count. When the reading reaches 0.4 volts, the red light will turn on and the voltage will jump back to +5VDC. After 3 seconds the light will turn off. You may turn the motor shaft counter-clockwise as well. The voltage will increase then by 0.04 volts per count until it reaches 9.6 volts and trips the protection circuit.

OSCILLOSCPE MONITORING: Set the scope to 2 volts / cm vertical and about 1 millisecond per cm horizontal. Zero the trace to the bottom line on the screen. DC couple the input. Place the probe on the test point and the ground clip to the blue capacitor ground lead. Follow the steps in VOLTMETER TESTING above.



STEP 4: CONTROL INPUT HOOK-UP

The control input group is the standard step motor drive STEP, DIRECTION and +5VDC lines. The STEP and DIRECTION signal drivers must be TTL compatible and have edge transition times of 100 ns or faster. The +5VDC is the opto-isolator common anode line and must be returned to the pulse source +5VDC supply.

(TERM. 10) DIR Connect the DIRECTION line to this terminal.
(TERM. 11) STEP Connect the STEP line to this terminal.
(TERM. 12) +5VDC Connect this terminal to the controller +5VDC power supply

STEP 5: TESTING THE CONTROL INPUTS

You may wish to test the functionality of these inputs. If you usec an oscilloscope in the previous section, leave it connected to the test point. If you used a voltmeter, then remove it from the drive.

Set the STEP pulse generator to about 40 pulses per second and set the DIRECTION output to clockwise (logical "1"). Turn on the power supply. After the power-on reset period of 5 seconds the FAULT light will turn off.

If you are using an oscilloscope, then the test point voltage will begin to increase until 3 seconds later it trips the protection circuit at 9.6 volts. The FAULT light will turn on for 5 seconds and voltage will snap back to +5VDC. After the FAULT turns off, the sequence will repeat again.

If you are not using an oscilloscope, just see if the FAULT light turns on and off every three seconds.

STEP 6: MOTOR HOOK-UP

Make sure the power is off and the STEP pulse source is set to zero pulses per second. Check to see if the trimpot settings are set according to the instructions on page 2. You may wish to secure the motor so it can't jump off the bench.

(TERM. 3) ARM- Connect the BLACK motor lead to this terminal.
(TERM. 4) ARM+ Connect the RED motor lead to this terminal.

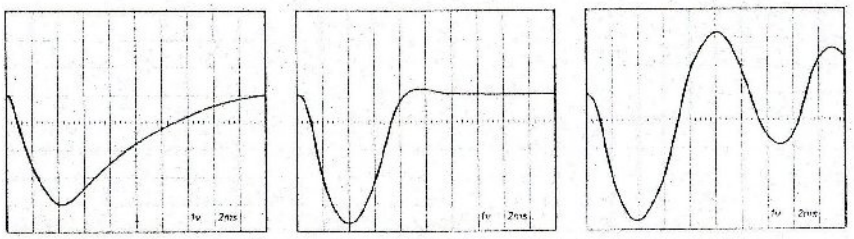
STEP 7: TUNING THE SERVO

Turn on the power supply. The FAULT light should turn off after 3 seconds. If everything is correct you should hear the motor "singing". This is normal. The motor is dithering or bouncing between adjacent encoder counts. The integral term in a PID loop has infinite DC gain over time and will amplify even the smallest position error. Because encoder feedback can only occur on count edges, the loop is "blind" until it encounters an encoder count edge. It then reverses the motor direction until another edge is found, then the process repeats.

If the motor jumps slightly and the FAULT light immediately turns back on, then either the motor is wired backwards or the trimpots are misadjusted. Check the trimpot settings. If they seem right then switch the motor leads and try again. If it still doesn't work and you followed all the previous steps, call me at the number at the end of this document.

Now turn on your STEP pulse source and ramp the speed up to see if the motor turns. It should turn clockwise with a logical "1" on the DIRECTION input.

The optimum way to tune the servo is to induce an impulse load on the motor while watching an oscilloscope to see how the motor behaves in response, then adjusting the PID co-efficient for optimal behavior.



OVERDAMPED *CRITICALLY DAMPED* *UNDERDAMPED*

In all cases the motor must return to the command position, what matters is how it does it. The manner in which the motor returns to its command position is called damping. At one extreme, called over damped response, the motor returns to position after a long, drawn out delay. At the other extreme, called under damped response, the motor returns to its position too rapidly, overshoots, returns and undershoots and so on until it finally settles at its command position. This is also called ringing; when extreme, the over/undershoot builds in amplitude until the motor enters violent oscillation. Between the two extremes is the optimal response called critical damping. Here the motor rapidly returns to its position with little or no overshoot in the minimal amount of time.

GAIN AND DAMPING

GAIN and DAMPING settings generally track each other. If you increase GAIN (greater stiffness), then increased DAMPING is needed as well to restore critical damping. Be careful, increasing GAIN without increasing DAMPING may cause the motor to break out into violent oscillation.

The higher GAIN is set, the noisier the motor will be when stopped. This is because higher gain causes more vigorous dithering between encoder counts at rest. There is a trade-off between high gain (high stiffness) one hand and excessive dithering (noise and motor heating) on the other. Use judgment here.

To see how your servo is compensated it is first necessary to induce a disturbance. The easiest way is to switch the DIRECTION input while commanding a constant speed via the STEP input. The abrupt direction change puts just the momentary load needed on the motor while you watch how it responds.

If you are using an oscilloscope, use channel 1 on the test point and channel 2 on the DIRECTION input. Set the trigger to "normal", trigger source to channel 2 and trigger edge to "+". You should see a single sweep for every clockwise change in direction.

Slowly increase STEP speed until you get a picture similar to one of the three above, and then do the following:

1) OVERDAMPED: Decrease DAMPING or increase GAIN
2) CRITICALLY DAMPED: Do nothing; you're there
3) UNDERDAMPED: Decrease GAIN or increase DAMPING

(POSITION ERROR TEST POINT NOTE)

Don't confuse the POSITION ERROR with the motor or machine position. The signal is actually the differential position error between the command speed and the motor speed. As noted above, sending clockwise STEP pulses moves the POSITION ERROR voltage more positive while turning the motor clockwise moves the POSITION ERROR voltage more negative.

When the motor encoder counts match the number of STEP pulses being sent one for one, the POSITION ERROR voltage stays at +5VDC. If the motor gets ahead of the STEP pulses such as during very rapid deceleration, the voltage will decrease by 0.04 volts for every encoder count the motor is ahead of the STEP pulses sent. The PID algorithm will force the motor to match the STEP input over time and restore the POSITION ERROR voltage back to +5 VDC.

CURRENT LIMIT

The current LIMIT trimpot sets maximum current the motor is permitted to have. It is adjustable from 0 amps to 20 amps. Normally the LIMIT trimpot is set to maximum (20 amps) unless you want to limit motor torque to a lower value.

Motor speed and position is unaffected by the current LIMIT setting unless the torque demand due to load exceeds this setting, then the motor position will fall behind the command position because of insufficient torque.

FAULT INDICATOR

The FAULT indicator is on while the drive is in power-on reset, the DISABLE input is held "low" or if the protection circuit is tripped due to a fault condition. All power MOSFETs are turned off and all internal counters are reset. The FAULT condition lasts for 3 seconds, and then self-resets to try again. If the protection circuit tripped it and the cause is not cleared, then it will immediately re-enter the FAULT state again and repeat the cycle.

There are two conditions that will trip the protection circuit. One condition is if a short-circuit occurs and current exceeds 20 amps. The other condition is if the POSITION ERROR exceeds +/- 120 counts causing a break of the servo-lock. This condition can have several causes:

1) The loop settings are severely under-damped and the motor breaks out into oscillation.
2) Excessive motor load due to acceleration or workload.
3) The speed command in excess of what the motor can deliver.
4) The current LIMIT is set too low.
5) The power supply current is insufficient for the demand.
6) The power supply voltage is below 18 VDC.
7) The motor is wired backwards, is broken or disconnected.
8) Encoder failure.

REVERSING DEFAULT MOTOR DIRECTION

The G340 will turn the motor in the CW direction when the DIRECTION input is "high" (logical "1", or +5VDC). If instead CCW is preferred, then:

1) Reverse the motor "+" and "-" leads (term. 3 with term. 4)
2) Reverse the encoder "channel A" and "channel B" leads (term. 8 with term. 9)

USING THE G340 WITH VERY SMALL MOTORS

Very small motors have low inductance relative to their operating current. Consequently ripple current due to pulse-width modulation can quickly overheat and destroy these motors. If the G340 will be used with these motors, then an external low pass filter must be used to attenuate ripple current to tolerable levels.

A suggested filter consists of two 150 micro Henry inductors in series with each motor lead and a 2 microfarad, low inductance film capacitor across the motor leads. The inductors must be rated for the anticipated peak motor current.

This filter is also need if ironless-armature or "pancake" motors are used. These motors have very low inductance as well and will overheat if driven directly by the G340.

(TERM. 5) ERR / RES

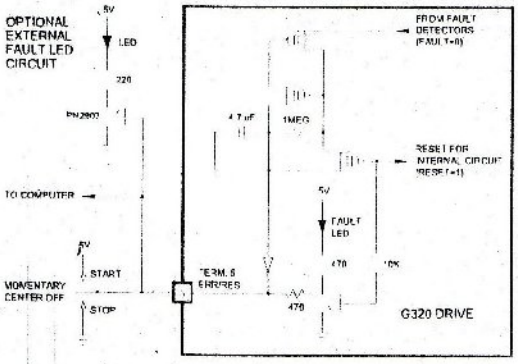
This terminal functions as an ERROR output and as a RESET input. Because this terminal functions as both an input and an output, some detailed description is necessary.

When first testing the G340, ERR/RES (term. 5) was connected to ENC+ (term. 7). It can be left that way if it is not necessary to read the state of the ERROR output. Otherwise, the following details are important.

The ERROR output is latched in the "ERROR" state (term. 5 = "0") by the power-on reset circuitry in the G340. It will stay in this state indefinitely until it is cleared by applying +5V to this terminal for at least 1 second.

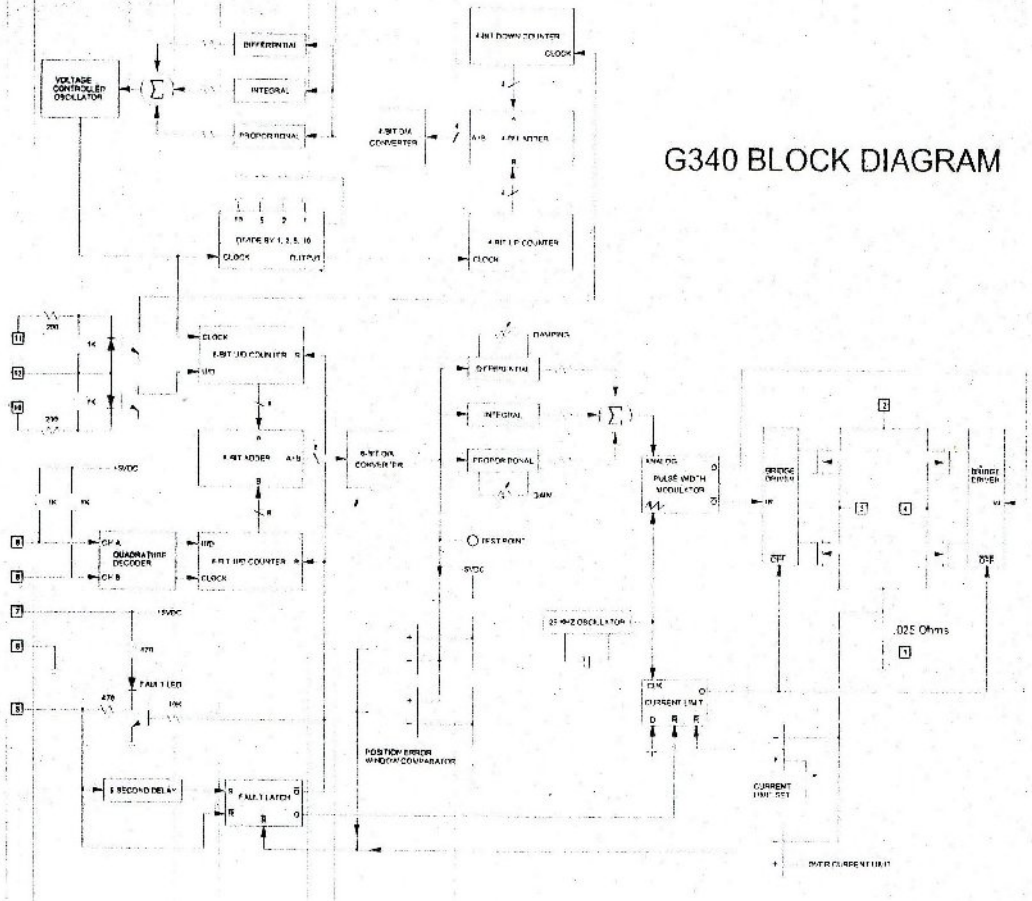
The voltage on this terminal is +5VDC when the G340 is functioning normally. The voltage on this terminal goes to 0VDC whenever the FAULT indicator is lit. This output can be used to signal your controller that an error has occurred.

Normally when the G340 is first powered up, it will be necessary to push the momentary switch to START for 5 seconds. This will clear the power-on reset condition and extinguish the FAULT LED. The motor will then be enabled and the drive will begin to operate. If at anytime after that a condition occurs that causes the G340 to "fault out", such as not being able to complete a step command, the ERR/RES terminal will go to "0", signaling the computer an error has occurred. This will require the operator to correct the problem that caused the fault and then push the switch to "START" for 5 seconds to re-enable the G340.



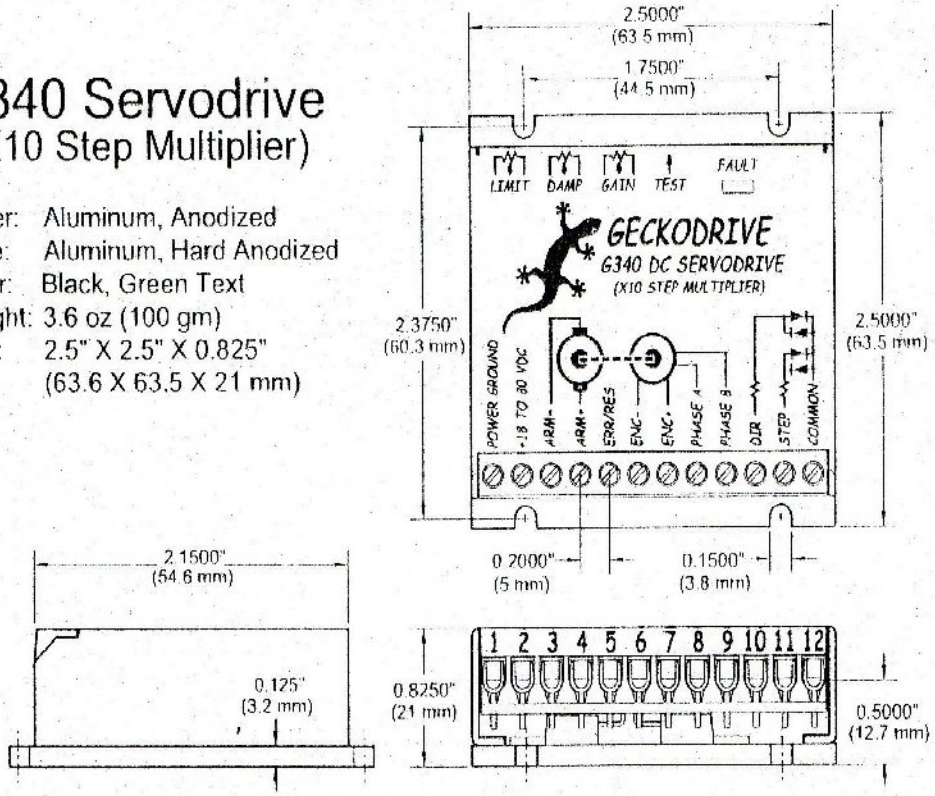
At anytime the operator can push the switch to the "STOP" position to immediately halt the G340 drive. Anytime the G340 is in the "FAULT" state (FAULT LED lit), all switching action stops and the motor freewheels and is unpowered. This will light the "FAULT" light.

G340 BLOCK DIAGRAM



G340 Servodrive
(X10 Step Multiplier)

Cover: Aluminum, Anodized
Plate: Aluminum, Hard Anodized
Color: Black, Green Text
Weight: 3.6 oz (100 gm)
Size: 2.5" X 2.5" X 0.825"
(63.6 X 63.5 X 21 mm)



G340 SPECIFICATIONS:

Power Supply	18 to 80 VDC
Motor Current	0 to 20 Amps
Lock Range	+/- 128 count following error
Feedback	Quadrature TTL Encoder
Feedback Resolution	X4 Encoder Line Count
Switching Frequency	25 kHz
Current Limit	0 to 20 Amp, Trimpot Adjustable
Analog PID	Damping and Gain Trimpots
Step Pulse Frequency	0 to 250 kHz
Step Pulse "0" Time	0.5 Microseconds Min.
Step Pulse "1" Time	3.5 Microseconds Min.
Multiplier Settings	X1, X2, X4, X5 and X10
Size	2.5" X 2.5" X 0.825"
Weight	3.6 oz weight
Encoder Supply	+5VDC, 50mA max

Geckodrive Inc.
14662 Franklin Ave
Suite E
Tustin, CA 92780

Phone: 1-714-832-8874
Fax: 1-714-832-8082
Web Site: www.geckodrive.com

A Full Service Motion Control Distributor and Systems Integrator.

JDH-2250-HF-2C-E
DM-683
CLIFTON PRECISION
REF. ONLY..

Supply Voltage, Vcc ..... -.05 to 7V
Output Voltage, Vo ..... -.05 to Vcc
Output Current per Channel..... -1.0 ma to 5 ma

PIN OUT

[1] GND
[2]
[3] Channel A
[4] + Vcc
[5] Channel B

• Pin One

APPENDIC C

LM7805 Datasheet

FAIRCHILD SEMICONDUCTOR®

www.fairchildsemi.com

MC78XX/LM78XX/MC78XXA

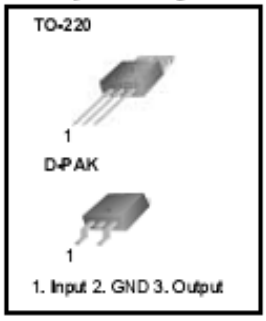
3-Terminal 1A Positive Voltage Regulator

Features

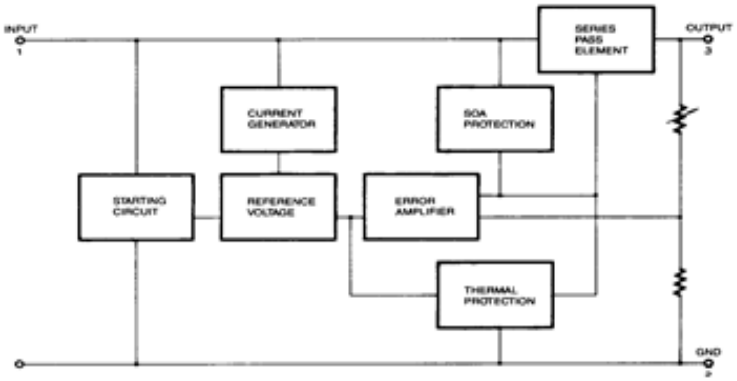
- Output Current up to 1A
- Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V
- Thermal Overload Protection
- Short Circuit Protection
- Output Transistor Safe Operating Area Protection

Description

The MC78XX/LM78XX/MC78XXA series of three terminal positive regulators are available in the TO-220/D-PAK package and with several fixed output voltages, making them useful in a wide range of applications. Each type employs internal current limiting, thermal shut down and safe operating area protection, making it essentially indestructible. If adequate heat sinking is provided, they can deliver over 1A output current. Although designed primarily as fixed voltage regulators, these devices can be used with external components to obtain adjustable voltages and currents.



Internal Block Digram



Absolute Maximum Ratings

Parameter	Symbol	Value	Unit
Input Voltage (for V_O = 5V to 18V) (for V_O = 24V)	V_I V_I	35 40	V V
Thermal Resistance Junction-Cases (TO-220)	$R_{\theta JC}$	5	°C/W
Thermal Resistance Junction-Air (TO-220)	$R_{\theta JA}$	65	°C/W
Operating Temperature Range	T_{OPR}	0 ~ +125	°C
Storage Temperature Range	TSTG	-65 ~ +150	°C

Electrical Characteristics (MC7805/LM7805)

(Refer to test circuit ,0°C < T_J < 125°C, I_O = 500mA, V_I = 10V, C_I= 0.33µF, C_O= 0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions		MC7805/LM7805			Unit
				Min.	Typ.	Max.	
Output Voltage	V_O	T_J =+25 °C		4.8	5.0	5.2	V
		5.0mA ≤ I_O ≤ 1.0A, P_O ≤ 15W V_I = 7V to 20V		4.75	5.0	5.25	
Line Regulation (Note1)	Regline	T_J=+25 °C	V_O = 7V to 25V	-	4.0	100	mV
			V_I = 8V to 12V	-	1.6	50	
Load Regulation (Note1)	Regload	T_J=+25 °C	I_O = 5.0mA to1.5A	-	9	100	mV
			I_O =250mA to 750mA	-	4	50	
Quiescent Current	I_Q	T_J =+25 °C		-	5.0	8.0	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1.0A		-	0.03	0.5	mA
		V_I= 7V to 25V		-	0.3	1.3	
Output Voltage Drift	$\Delta V_O/\Delta T$	I_O=5mA		-	-0.8	-	mV/°C
Output Noise Voltage	V_N	f = 10Hz to 100KHz, T_A=+25 °C		-	42	-	µV/Vo
Ripple Rejection	RR	f = 120Hz V_O = 8V to 18V		62	73	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25 °C		-	2	-	V
Output Resistance	r_O	f = 1KHz		-	15	-	mΩ
Short Circuit Current	I_{SC}	V_I= 35V, T_A =+25 °C		-	230	-	mA
Peak Current	I_{PK}	T_J =+25 °C		-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Changes in V_o due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Typical Perfomance Characteristics

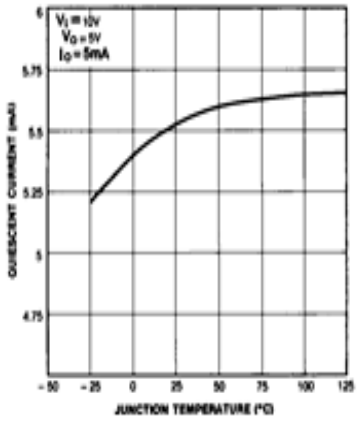


Figure 1. Quiescent Current

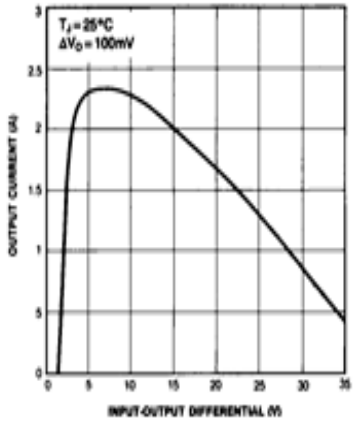


Figure 2. Peak Output Current

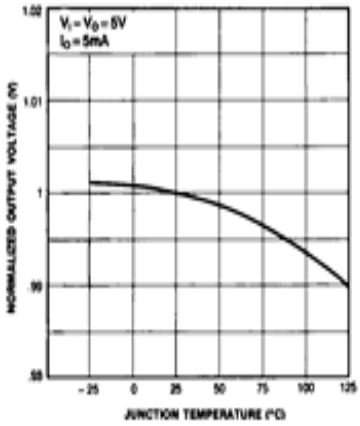


Figure 3. Output Voltage

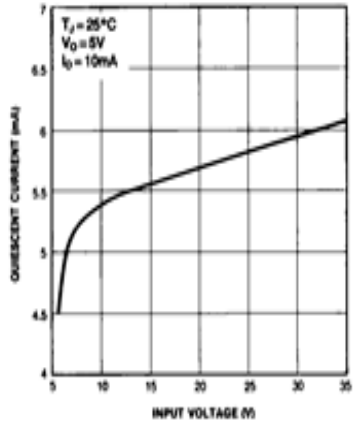


Figure 4. Quiescent Current

Typical Applications

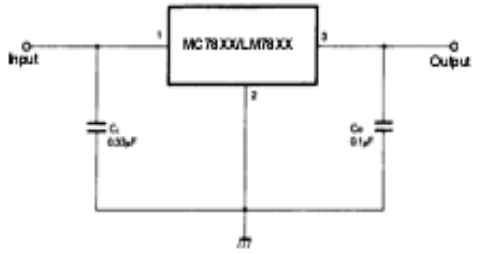


Figure 5. DC Parameters

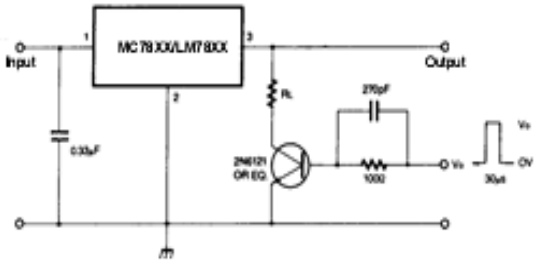


Figure 6. Load Regulation

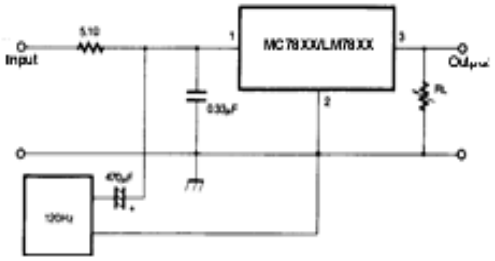


Figure 7. Ripple Rejection

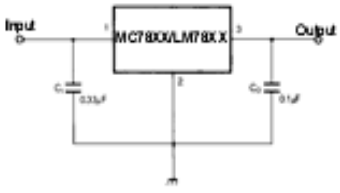
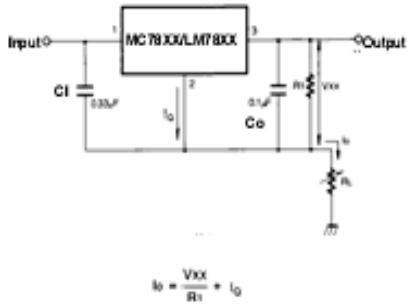


Figure 8. Fixed Output Regulator



$$I_O = \frac{V_{XX}}{R_1} + I_Q$$

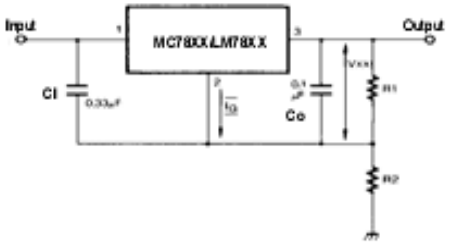
Figure 9. Constant Current Regulator

Notes:

(1) To specify an output voltage, substitute voltage value for "XX." A common ground is required between the input and the Output voltage. The input voltage must remain typically 2.0V above the output voltage even during the low point on the input ripple voltage.

(2) C_I is required if regulator is located an appreciable distance from power Supply filter.

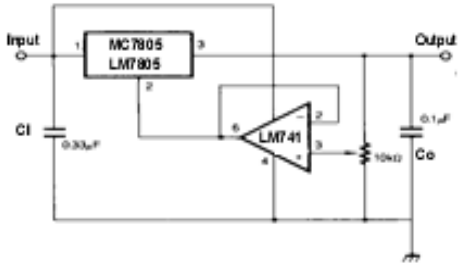
(3) C_O improves stability and transient response.



$$I_{R1} \geq 5I_Q$$

$$V_O = V_{XX}(1+R_2/R_1)+I_QR_2$$

Figure 10. Circuit for Increasing Output Voltage



$$I_{R1} \geq 5\ I_Q$$

$$V_O = V_{XX}(1+R_2/R_1)+I_QR_2$$

Figure 11. Adjustable Output Regulator (7 to 30V)

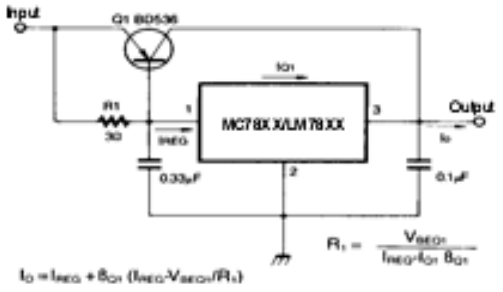


Figure 12. High Current Voltage Regulator

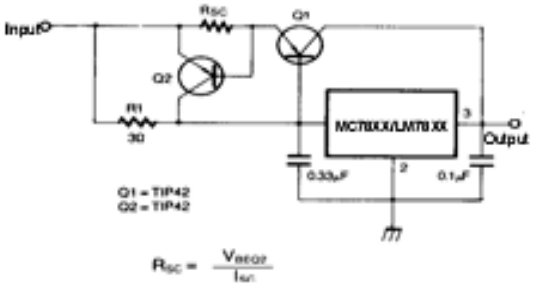


Figure 13. High Output Current with Short Circuit Protection

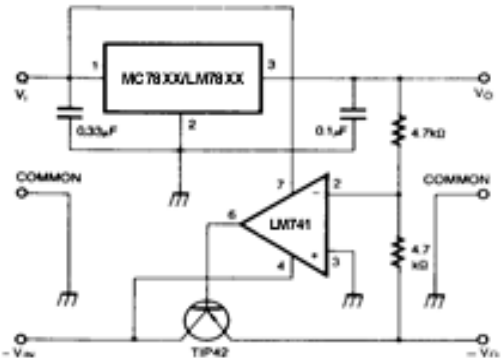


Figure 14. Tracking Voltage Regulator

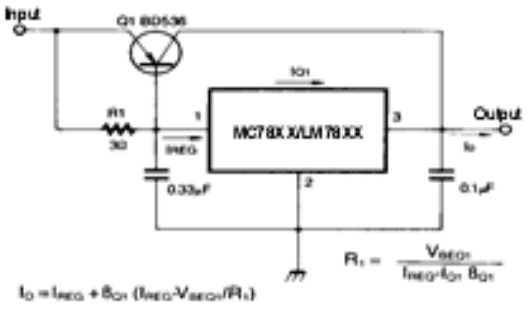


Figure 12. High Current Voltage Regulator

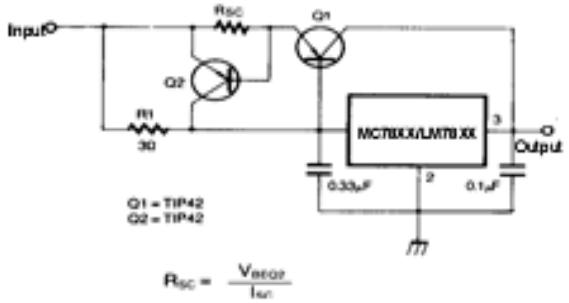


Figure 13. High Output Current with Short Circuit Protection

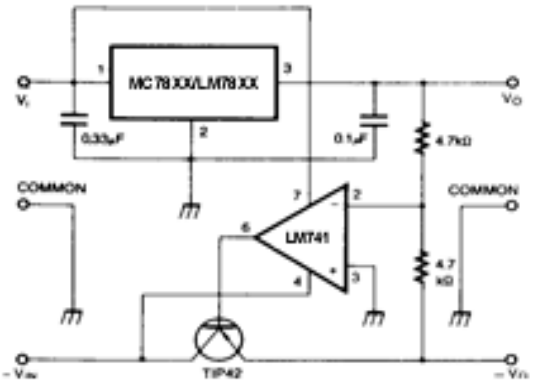
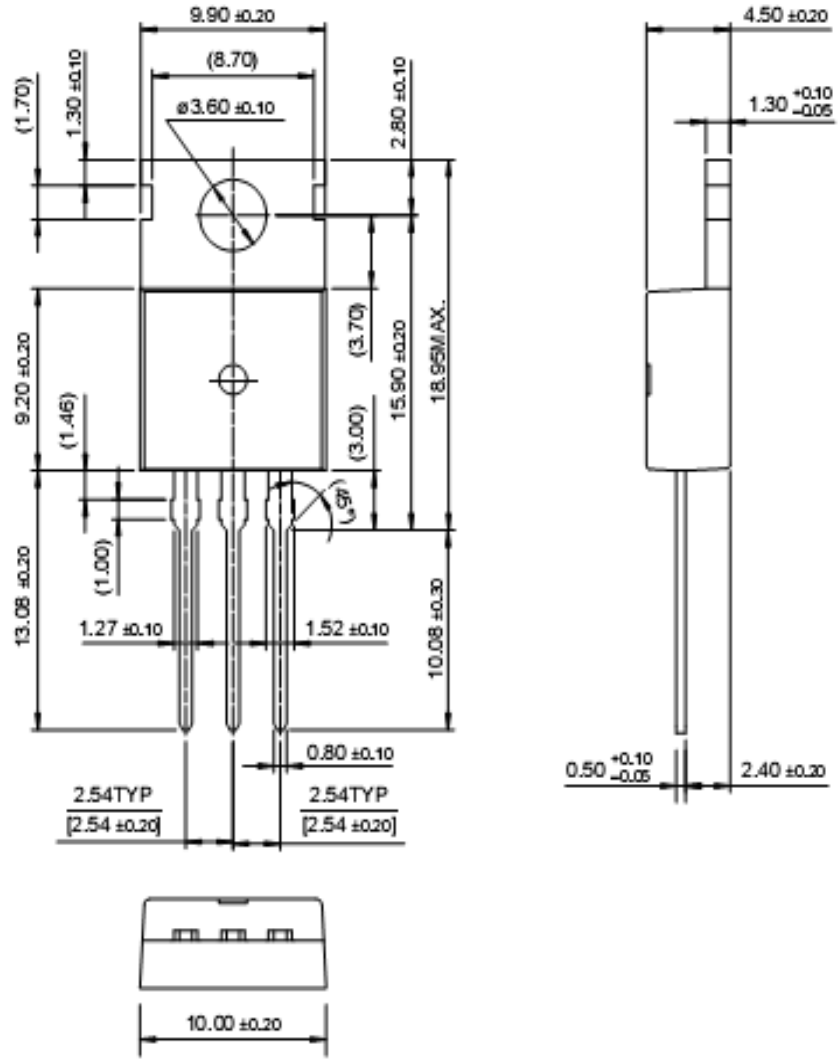


Figure 14. Tracking Voltage Regulator

Mechanical Dimensions

Package

TO-220



APPENDIX D

ASM File

```
'*  Name    : UNTITLED.BAS                                          *
'*  Author  : [select VIEW...EDITOR OPTIONS]                        *
'*  Notice  : Copyright (c) 2008 [select VIEW...EDITOR OPTIONS]     *
'*          : All Rights Reserved                                   *
'*  Date    : 10/13/2008                                            *
'*  Version : 1.0                                                   *
'*  Notes   : dc motor                                              *
'*          :                                                       *
'******************************************************************
motor VAR portb.0

    trisb = 0
    trisa = 1


forward:   PULSOUT 0,170
           PULSOUT 1,130
           PAUSE 20
           GOTO forward

backward:  PULSOUT 0,130
           PULSOUT 1,170
           PAUSE 20
           GOTO backward

END
```

APPENDIX E
Circuit Diagram for the Whole System