DESIGN OF A DSP-BASED SERVO SPEED CONTROLLER

by

Jiabing Lu, B. Eng

A thesis submitted to

Dublin City University

for the degree of

Master of Engineering

School of Electronic Engineering

DUBLIN CITY UNIVERSITY

July 1992

Abstract

The brushless servo drive is arguably the most important emerging drive category for robotics, machine tools and other applications. This places increasingly high demands on the servo motor and controller.

In this thesis, a digital speed and thermal protection controller is developed for a high performance brushless DC servo system. The speed controller is designed to produce a high accuracy and a fast dynamical response. The thermal protection controller prevents the motor against overheating while providing a high utilization of the drive.

Two PID design methods are studied for the speed controller, an "analog design approach" and a "grapho-analytical pole-placement procedure". The former provided an easy design and the later resulted in a more satisfactory control performance.

The thermal protection controller uses a generic lumped capacitance-resistance thermal model to predict the motor temperature. A current limit regulator is developed to maintain the motor temperature below this insulation limit, and to maximize the motor output once the limit is reached.

A simulation scheme for this servo system is developed to investigate the control characteristics of the system before experimental testing.

The digital speed controller has been implemented using the TMS320C30, a high performance digital signal processor. The control software, written in the TMS320C30 assembly language, is developed.

Experimental results are presented, which demonstrate the performance improvement of the designed control system.

DECLARATION

I hereby declare that all the work presented in this thesis is my own, except where references have been made. I also declare that no part of this thesis has been submitted for a degree at any other institution.

SIGNATURE: Jiabing Lu

DATE: 10th. Sep. 1992

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to my supervisor, **Dr. A. Murray**, who provided me the opportunity to undertake this research project for a higher degree. Throughout the course of this work, he has been a constant source of advice, support and encouragement, which is and will always be much appreciated.

I also wish to thank **Dr. M.J. Barrett, J. Se ton** and **P. Moran** for their help in proof-reading the thesis and helping to turn it into a readable document.

Thanks to all technical staffs of the school of Electronic Engineering, especially **C. Maguire, S. Neville, D. Condell** and **J. Whelan** for providing the necessary equipment and help to complete this research.

Thanks to my fellow postgrads in the power electronics laboratory for their kind help and support.

Finally a special thanks to **Dr. J. Yan** for his valuable assistance and encouragement during this research.

CONTENTS

CHAPTER 1

INTRODUCTION

Brushless dc motors with permanent magnetic material, such as Samarium-Cobalt, have been in ever-increasing use for more than ten years. They are currently considered among the best options not only in high performance control systems such as machine tool feed drives, robotics, indexing equipment, punch/press machines, radar/antenna drives, and tracking systems, but also in more mundane applications such as consumer and commercial air conditioning, etc [1-1]. At present, modern methods , technology and materials have enabled the industry to develop large (up to 200 hp) brushless dc motors to operate as general purpose machines to do the routine tasks of variable speed control in the industry applications such as pumps, conveyors, extruders, presses etc. The permanent magnet brushless DC motor has the following major advantages:

(1) high starting torque;

(2) high torque at low speed;

(3) excellent speed regulation (greater than 20,000:1) from no load to full load;

(4) low inertia;

(5) easy maintenance;

(6) high power/volume ratios since the windings are only on the stator, therefore the I^2R thermal losses can be more easily dissipated in the air - this allows a brushless motor of the same frame size as a brush motor to have a higher specific power output;

(7) a high degree of accuracy of the adjustable speed or position;

(8) easy to control;

(9) capable of operating in clean and hazardous environments in industries such as food processing, chemical and aeronautics industries or operating fully immersed in fluid or vacuum;

(10) better overall efficiency because any friction losses between the brush and the commutator are eliminated;

(11) a constant power factor, better than 90% even at zero speed, and typically 95% when driven by a pulse width modulated motor drive.

The prices of brushless dc motors are becoming competitive than ever due to the increasing market for these motors, the continual improvement in the manufacturing process of the motors, and the advancements in the solid-state power electronic devices and circuits. These brushless motors are replacing, at a rapid pace, the existing hydraulic as well as the conventional electric drive systems in a number of applications that require the above advantages.

At present most brushless dc drives are controlled using analogue controllers. However, with the rapid development of microcomputers, especially the DSP processor, digital feedback control is being now applied. The control results achieved through digital control are often better than analogue systems and they are much less expensive to implement, change or replace. Digital controllers can exchange complex blocks of information at quite high speeds and can store different programs, some of which are scheduled only if certain conditions are met. They are not affected by component aging and temperature drift, and they enhance greatly the performance characteristics of the system. At present, digital computers are much smaller, lighter and more powerful. Their cost and their power requirements are reasonably low and becoming lower every year. They are ideal control device for the servo application.

The aim of this thesis is to develop a digital speed controller for a permanent magnet brushless DC servo motor that will replace the analogue speed loop of the system and develop a thermal protection for the brushless dc motor. A digital signal processor, TMS320C30, is used for this purpose.

This thesis presents two digital control methods used in the design of the speed controller. An "analogue design of discrete controller" method [1-7] is successfully achieved due to the high speed of the TMS320C30. This method combines the advantages of fast sampling intervals of the TMS320C30 and well-developed analogue design techniques. This makes the controller design become simple and convenient. An

alternate design of the speed controller is a direct-digital proportional-integral-derivative implementation based on a pole-placement technique [1-8]. Using this scheme one can obtain a desired performance by adjusting the control parameters on the accuracy, speed of response, and stability margin of the system. This controller has a better dynamic performance due to the fact that a more accurate direct digital design method than the first designed controller is used.

A simulation method is also described in the thesis. The method using a block-oriented technique exactly simulates the designed system on a personal computer. This can examine the system and adjust the parameters of the controller to the desired level before a practical test is carried out on the system. Implementations of the simulation are presented.

The thermal protection of PM brushless motors is a key problem in motor servo drives in industrial applications. This thesis presents a real-time thermal protection control scheme for a brushless DC servo motor. A thermal model of the motor is established, and the thermal controller uses this to predict the temperature of the motor windings. Once the predicted temperature reaches the winding insulation limit, the thermal controller maintains the motor operation within a maximum allowable speed-torque region. This keeps the winding temperature below the insulation limit, while maximizing the motor power output. Simulation results for this scheme are presented.

An implementation of the TMS320C30 based control system is detailed and a real-time control software program is developed to realize this DSP based servo system. The experimental results are presented and this shows the successful design of the digital speed controller for the brushless DC drive system using the TMS320C30 DSP device.

Thesis Structure

The thesis is divided into eight chapters. Chapter 1, the introduction, is an overview. Chapter 2 is a general description of the permanent magnet brushless DC servo system. It describes the component parts of the brushless servo system. A typical analogue

current-controlled brushless DC motor servo system is studied. Chapter 3 details the mathematical model of the brushless DC servo system based on the AEG BHT 2214 brushless servo drive. Following this, two digital speed controllers are designed. One method utilizes the "analogue design of discrete controller" technique and the other is a digital PID controller using the pole-placement scheme to complete digital servo control. Chapter 4 develops an application-oriented simulation technique based on the fourth-order Runge-Kutta method. Chapter 5 presents a thermal protection scheme for the brushless servo system. This scheme ensures a maximum utilization of the motor output power. Chapter 6 introduces the TMS320C30 PC System board and its application for the speed controller of the brushless DC servo. Chapter 7 shows the experimental and simulation results. Chapter 8 summarizes the overall research and gives recommendations for further work.

CHAPTER 2

PERMANENT MAGNET BRUSHLESS DRIVE SYSTEMS

2.1 Introduction

DC motors have many desirable features when applied to servo systems whilst still having some disadvantages. There has been a desire to replace the DC motor with a machine of similar performance characteristics, but without the brushes and commutators. Brushes require replacement, commutator surfaces wear and have to be turned, arcing cannot be permitted in certain hazardous locations, and the system imposes severe speed limitations on the motor. With the development of electronic switching devices, these mechanical switching components in the conventional dc motor can be replaced by power electronics components. The system is built using a motor with a multipolar permanent-magnet rotor and stator windings, electronic switching circuits, and a position sensing system. This is called a brushless DC motor because it behaves like a DC motor. From the following discussions, we can see that it is a dc motor in name only, since it is more similar to a permanent magnet synchronous motor.

In a conventional dc motor, the torque-speed characteristics are linear, except at high torque levels, where armature reaction effects become significant. The term "brushless dc motor" is used to identify the combination of an ac machine, a solid-state inverter, and rotor position sensors that results in a drive system having a linear torque-speed characteristic, as in a conventional dc machine. The 'ac' motor has polyphase windings on the stator and permanent magnets on the rotor. The motor operation is made self-synchronous by the addition of a rotor position sensor which controls the firing signals for the solid-state inverter. In response to these firing signals, the inverter directs

current through the stator phase windings in the controlled sequence to give a constant torque. It is much like a standard permanent magnet synchronous machine and operates as a self controlled synchronous motor. A distinction is that the synchronous motor requires sinusoidal current excitation, whereas the brushless DC motor is energized with square-wave or quasi-square-wave currents. The rotor position sensors for the brushless DC motor usually consist of a number of simple position detectors such as Hall-effect devices that sense the rotor magnetic field and so determine the phase switching points. The synchronous motor requires more precise position information to allow accurate synthesis of the sinusoidal current waveforms.

The torque contribution of a particular stator phase of the motor is a function of phase current and rotor position. If a constant direct current is supplied to one stator phase and the rotor is allowed to rotate, the developed torque due to the interaction between the winding current and the magnet flux will vary periodically with shaft position. This characteristic is known as the torque function or static torque/angle characteristic of the motor. In the brushless dc motor, the torque function is trapezoidal, whereas in the permanent magnet synchronous motor, the torque function is sinusoidal.

2.2 The brushless DC drive

The brushless DC drive system consists of the following components:

2.2.1 The motor

The motor consists of a rotor on which permanent magnets are mounted in pole pairs to supply the field flux. The stator contains the stator windings in which the current is fed in the correct sequence so as to produce a constant torque.

2.2.2 The sensing system

In order for the coils to be switched in the correct sequence and at the correct time, the angular location of the rotor field magnets must be known. This requires a position

sensing system which can consist of Hall sensors, an encoder, or a resolver.

2.2.3 The electronic commutator

The electronic commutator performs like the commutator in a brush DC motor. It uses information from the sensors and the control input to switch the inverter, this adjusts the DC power to drive the brushless DC motor.

2.3 Mode of operation

The brushless motor can have two, three or more stator phase windings. The three phase motor is the most common and has following modes of operation.

- Three-phase half-wave BLDC operation
- Three-phase full-wave BLDC operation

2.3.1 The three-phase half-wave brushless DC motor drive

For economic reasons, the three-phase half-wave brushless DC motor drive is often selected because the cost of the electronic package is lower than that of the three-phase full-wave brushless motor.

Figure 2-1 shows a basic three-phase half-wave brushless DC motor system The three stator phases are wye-connected with the neutral point joined to the positive terminal of the DC supply. Transistors TR1, TR2, and TR3 deliver unidirectional phase currents in response to base drive signals which are under the control of the rotor position sensor. This simple half-wave circuit is unusual in that there are no free-wheeling or feedback diodes to provide an alternative path for the inductive winding current when a transistor is turned off. Figure 2-2 shows a switching process of transistors and the current flow in one phase each time.

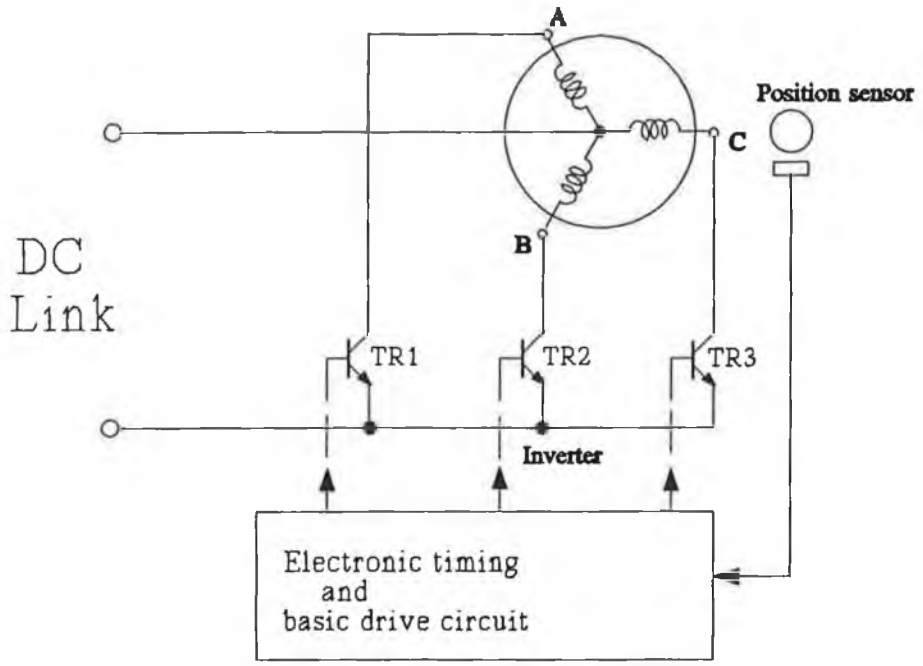


Figure 2-1 A three-phase, half-wave brushless dc motor drive

2.3.2 The three-phase full-wave brushless DC motor operation

A typical three-phase full-wave brushless DC motor system is shown in figure 2-3 in which a direct voltage is applied to a three-phase, wye-connected stator, and two of the three phases are active at all times. The respective sequences of the motor current is shown in figure 2-4 (g). Figure 2-5 shows the resulting flux vectors from the current, and illustrates how the particular switching sequence causes a clockwise field rotation. The switching is done electronically and the switching timing is done using a position sensor.

For line to line DC currents, the motor has the idealized trapezoidal torque functions of figures 2-4 (a), (b), and (c), each showing a 60-degree flat-topped region. Motor operation will be in the constant-torque region if the phases are supplied with a quasi-square-wave current as shown in figures 2-4 (d), (e), and (f). The current is obtained by switching the transistors, in figure 2-3, at 60-degree intervals to give a 120-degree

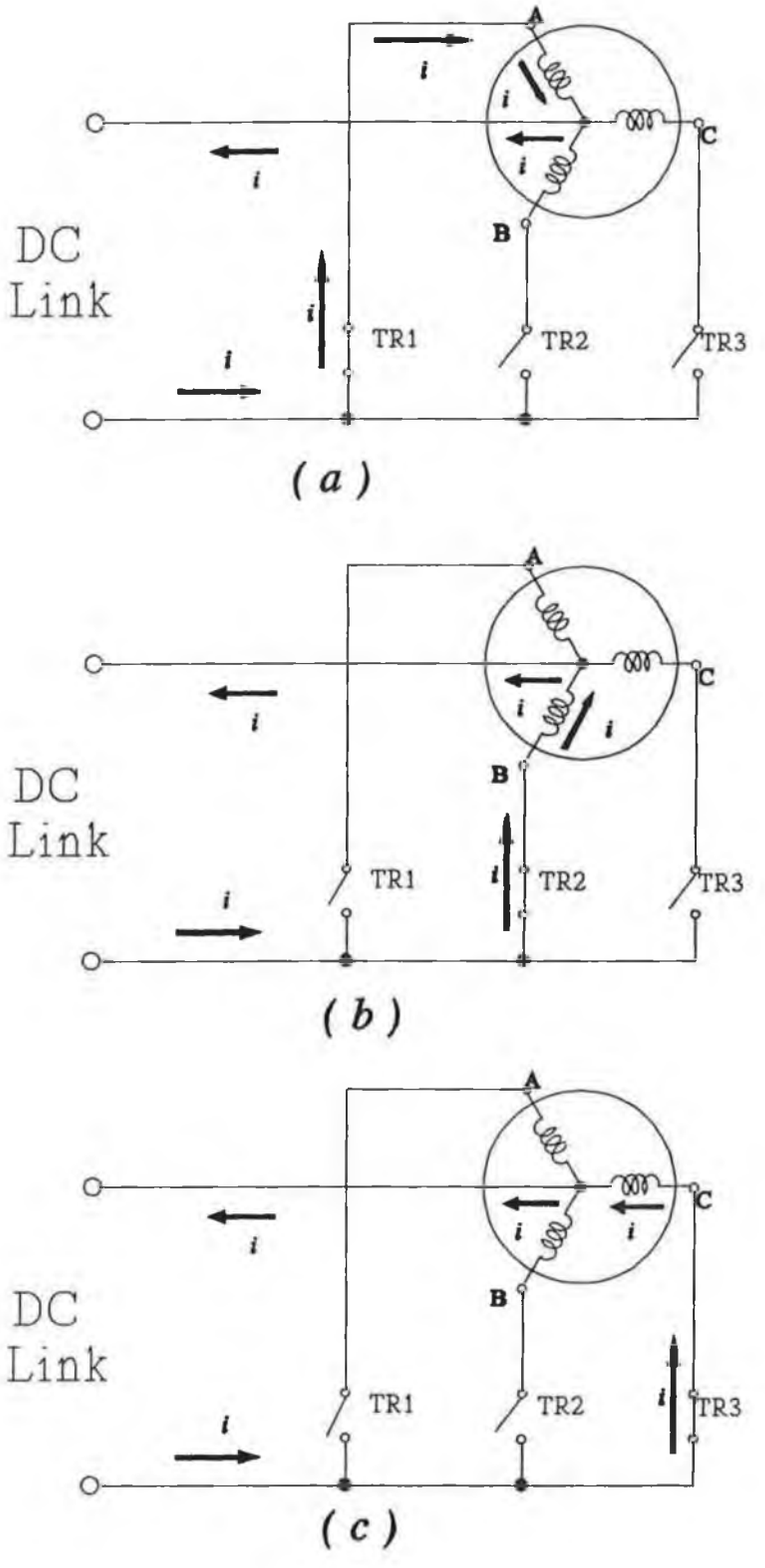


Figure 2-2 Three switch positions of the three-phase half-wave brushless DC motor

conduction angle. Each transistor switching occurs in response to the rotor position

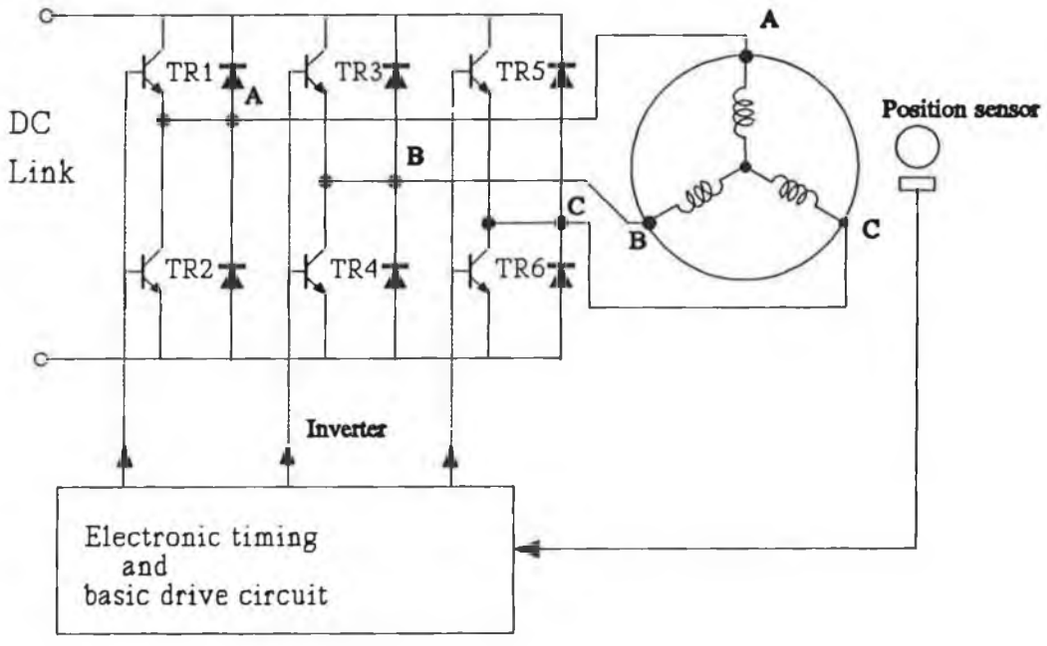


Figure 2-3 A three-phase, full-wave brushless DC motor

sensor. Figure 2-4 also shows that each motor phase conducts for a 120-degree period in each cycle, giving twice the winding utilization of the three-phase half-wave system. A steady nonpulsating torque of magnitude $T=K_TI$ is developed, and the torque reversal is achieved by phase-shifting the transistor base drive signals by 180 degrees. The idealized quasi-square-wave currents of figure 2-4 imply instantaneous switching from one phase combination to the next. In a practical voltage-fed system, the inductive load will delay the build-up of current and will also prolong conduction after the theoretical turn-off instant. In an actual motor, the torque function will also depart somewhat from the ideal trapezoidal waveshape. Despite these practical imperfections, the commercial brushless dc motor can achieve a very low torque ripple and is eminently suitable for use in a high-performance servo drive.

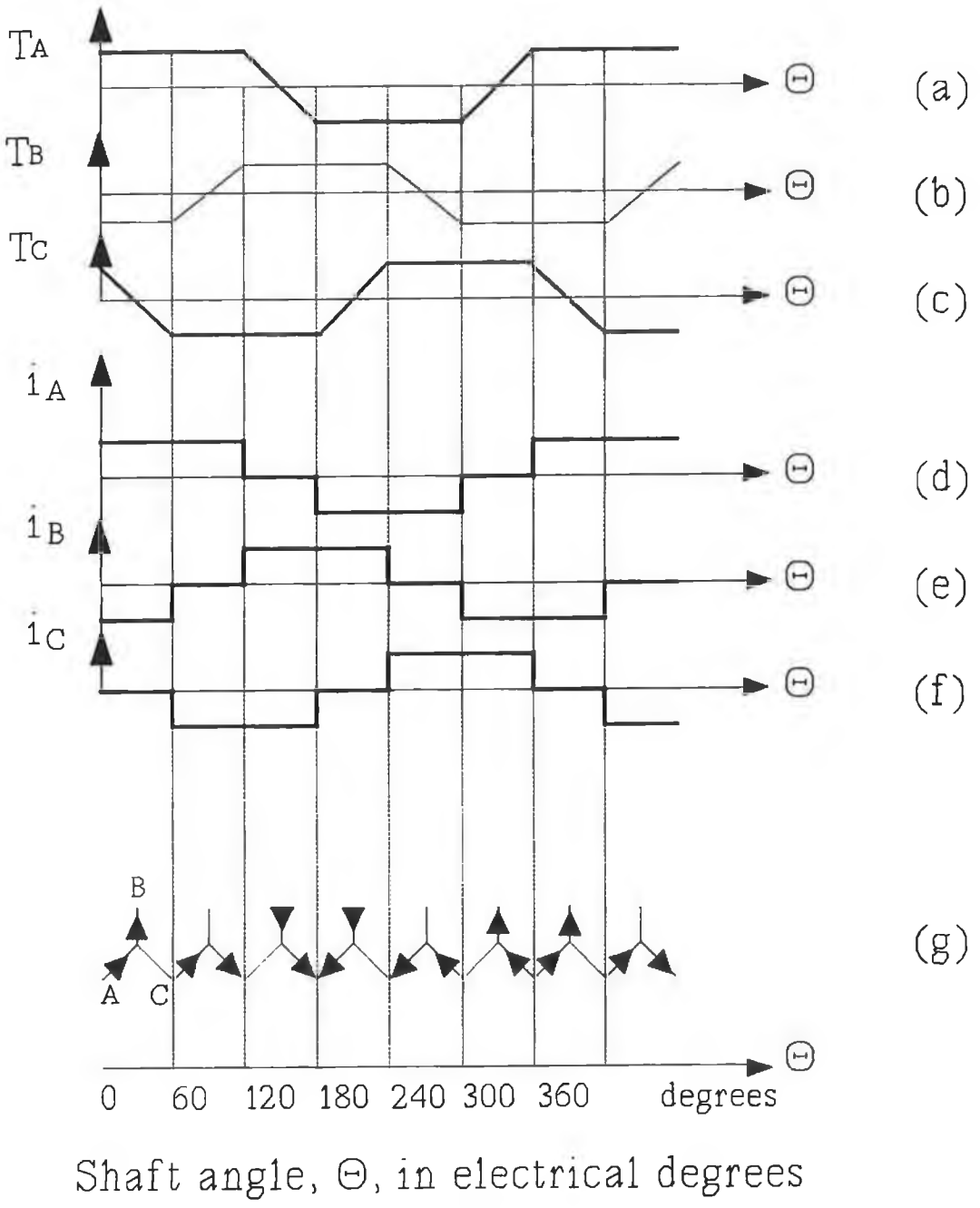


Figure 2-4 Idealized waveforms for the three-phase, full-wave brushless dc motor: (a), (b), (c) static torque/angle characteristics; (d), (e), (f) phase current for positive torque; (g) commutation sequence for clockwise rotation.

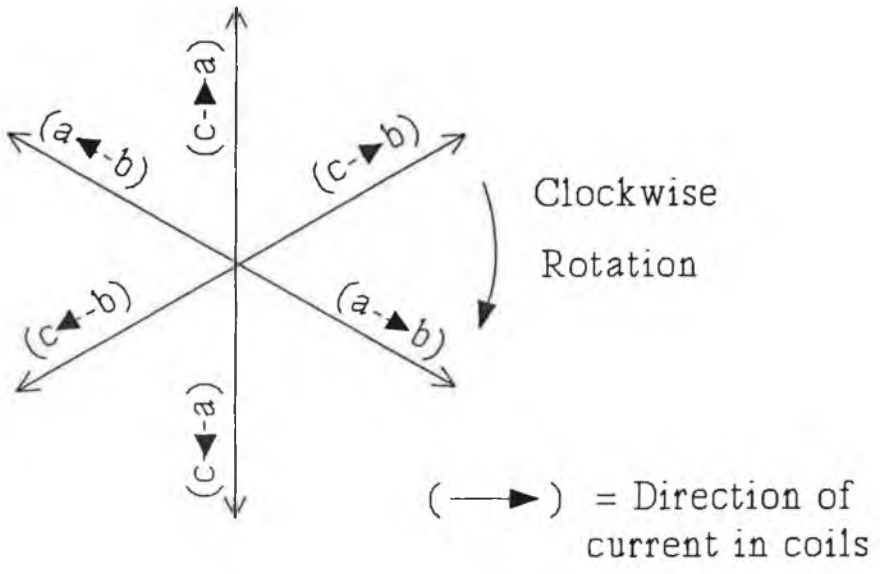


Figure 2-5 Resultant flux vectors from current in Fig.2-3 three-phase, full-wave, BLDC motor

2.4 Brushless motor characteristics

The brushless DC motor is an inverter driven motor with electronic commutators and a permanent magnet rotor. The basic equations and terminal characteristics are very similar to those of a DC motor. Ignoring second-order effects, the trapezoidal brushless DC motor has the following general voltage equation:

$$v = Ri + L\, di/dt + K_E\omega \qquad (2.4.1)$$

where v is the motor terminal voltage.

i is the motor current.

R is the resistance of a phase winding.

L is the inductance of a phase winding.

ω is the angular velocity of the rotor.

K_E is the back emf constant.

As stated already, the trapezoidal motor develops a torque of $K_T i$, where K_T is the torque constant. The dynamic equation is:

$$T = K_T i = J\, d\omega/dt + D\omega + T_F + T_L \qquad (2.4.2)$$

where J is the total system inertia
D is the viscous damping coefficient
T_F is the frictional torque
T_L is the load torque.

Equation (2.4.1) and (2.4.2) are the usual DC machine equations, and within the international system of units (SI), the values of K_E and K_T are numerically equal.

The combination of the inverter, rotor position sensor and the brushless motor constitutes an electronic commutator in which the inverter DC link voltage and current correspond to the armature voltage and current of the motor. Consequently, the brushless DC motor can employ standard DC drive techniques for speed and torque control. Thus the average DC link voltage for the inverter can be controlled by a series transistor acting as PWM regulator. In this manner, the voltage supplied to the electronic commutator is varied and motor speed is controlled. This approach is clearly analogous to speed control by armature voltage regulation of a DC motor. It is more usual to operate a DC motor with an inner current loop that gives direct torque control, and in the brushless DC system, a series transistor regulator in the DC link can operate in a current-controlled PWM mode. However, this external transistor is not really necessary because the main inverter transistors can regulate the amplitude of the motor current by PWM control as well as commutating the current from phase to phase at the appropriate shaft positions. Current sensing is required in the motor leads, and the current feedback signal is used in a conventional PWM current loop. These controllers are now available in integrated circuit form for brushless DC motor applications which will be discussed later.

2.5 Position sensing system

The rotor position sensors are an integral part of the brushless DC motor system. They are used to indicate the intermediate position of the rotor so that appropriate switching signals can be generated for the inverter. For small motors, the rotor position sensors are usually mounted on the inside surface of the stator, whilst for larger motors, they are usually a separate unit fixed onto the non-drive end of the shaft. There are several types of rotor position sensors; hall effect sensors, electro optical sensors, resolvers, and digital encoders are the most commonly used device. For brushless DC motors, hall effect sensors are normally selected to detect the magnitude and direction of magnetic fields. Motors require three of these sensors symmetrically mounted on the stator. The output signals from the sensors are processed to provide the position signals required for the base device circuits to switch the transistors in the inverter.

2.6 Brushless motor servo control

In this section, a typical current-controlled brushless dc motor servo drive system is described, as shown in figure 2-6.

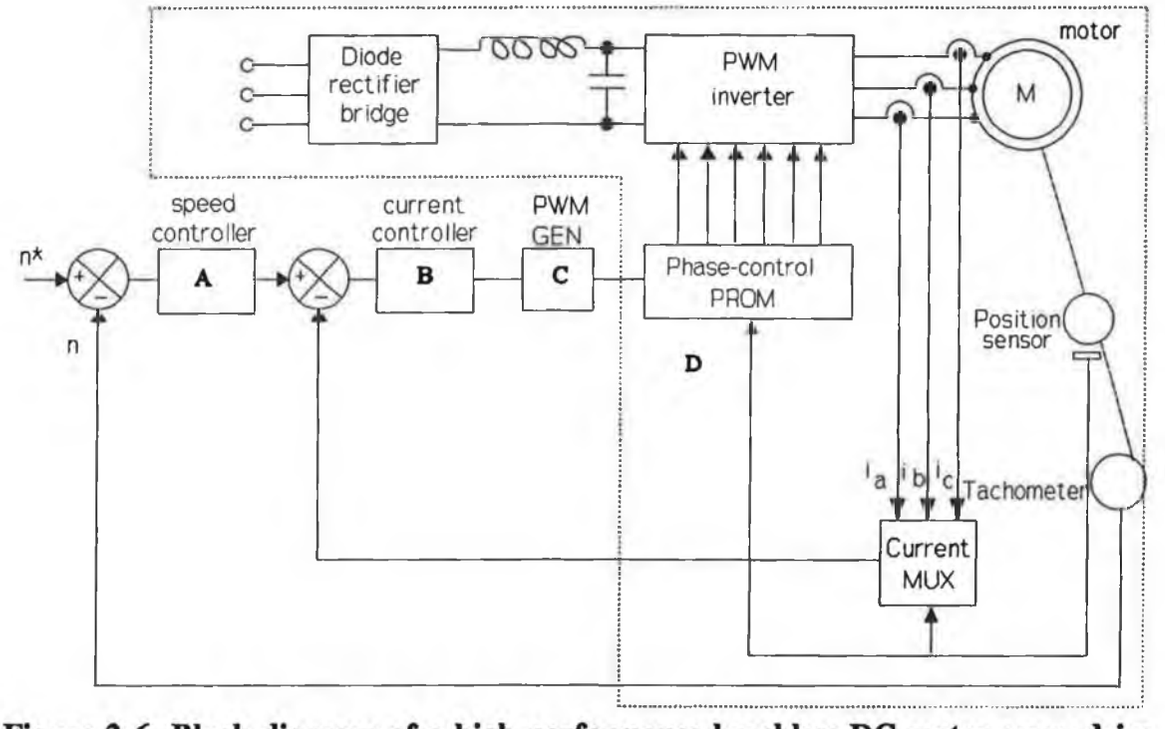


Figure 2-6 Block diagram of a high-performance brushless DC motor servo drive

Precise control of the motor speed is achieved by a classical double-loop control scheme with an outer high-gain speed loop and an inner current loop. For the sake of discussion, the major portions of the system are labelled A through D and described as follows.

1. Speed loop (A)

The speed loop adjusts the motor torque to ensure that the motor speed follows the input command. The magnitude and polarity of the set speed command represent the desired motor speed and direction of rotation, respectively. This signal is compared with the tachometer signal to produce a speed error that is fed to the speed loop A. Normally, the tachometer used in a brushless drive is itself brushless. The speed loop is compensated in various ways, such as with lead and lag networks or PID techniques, to ensure stable operation and to allow a dynamic match of the drive system to the load. As in the classical DC servo drive, the compensated speed error is the demanded current and hence the demanded torque in the motor. The negative value of the speed error signifies a negative torque demand and, in the brushless DC motor, this torque reversal is achieved by energizing each motor phase in the negative-torque region of the static torque/angle characteristic. Therefore the servo drive can be controlled in all four quadrants.

B. Current controller (B)

The current loop gives a signal to the PWM generator to ensure that the motor current follows the motor torque command. In figure 2-8, the current from the speed loop is compared with the actual current by multiplexing the current feedback signals from the three phases into one loop. The error signal is fed into the current loop which is also compensated in the same manner as the speed loop, with lead and lag networks, to ensure a fast and accurate response during load variations.

C. PWM generator (C)

The PWM generator is used to generate base drive signals for the inverter transistors. This loop is implemented using a usual current-controlled PWM technique, in which the amplified current error is fed to a comparator circuit with a fixed-frequency triangular wave of several kilohertz. The comparator functions as a PWM generator and delivers PWM waveforms whose duty cycle varies with the current error.

D. Brushless motor drive system (D)

Block D includes a brushless dc motor, power supply (uncontrolled diode bridge rectifier and filter), inverter, position sensing system and tachometer. The PWM signals and the position sensor signals are fed to a programmable read only memory (PROM), as shown in figure 2-8. This phase control PROM stores a table containing the correct state (on or off) of each transistor for each position of the shaft and for positive and negative torque. Therefore, positive and negative torque can be developed with a magnitude determined by the demanded current.

The current controlled brushless DC motor servo drive system gives full four-quadrant operation of the brushless DC motor. When the speed is reduced suddenly, the negative speed error results in a large braking torque and rapid deceleration of the motor, and the energy is regenerated through the power inverter to the DC link. This regenerated energy must be dissipated in a dynamic braking resistor across the DC link.

This brushless DC motor servo drive can be used for high-performance industrial servo systems due to its static stiffness and low speed torque smoothness. The dynamic characteristics of the drive can equal or excel those of conventional DC brush servo drives with PWM transistor amplifiers. Compared with sinusoidal brushless drives, the cost advantage of the motor can be emphasized because a low-resolution sensor will suffice to detect the phase switching points. In addition, for a given shaft torque, the peak current demand is less in a trapezoidal system than a sinusoidal system, and therefore a lower current capacity is required by the inverter. Although the brushless DC motor can have a much greater torque ripple than the sinusoidally based system or the induction motor, the careful magnetic circuit design of the trapezoidal machine and

the use of rare earth materials result in satisfactory torque smoothness. The effects of any residual ripple can be suppressed by the closed-loop action of the velocity and current feedback loop, to give excellent low-speed performance. Torque smoothness and static stiffness are perfectly satisfactory for servo applications in machine tools and robotics. Because of its simplicity, low cost, and good performance characteristics, the trapezoidal brushless dc motor is a major contender in the field of high-performance servo drives.

CHAPTER 3

DESIGN OF THE DIGITAL SPEED CONTROLLER

3.1 Introduction

The previous chapter described the permanent magnet brushless DC motor with a typical current control servo system. In early applications of the brushless servo motor, the electronic control systems were always built using analogue components. With the development of the microprocessor, especially the digital signal processor(DSP), the controller can now be designed more precisely and more flexibly using digital techniques.

This thesis presents a digital speed controller for a permanent magnet brushless DC servo drive. The design is based on an existing analogue system and it is implemented by a digital signal processor, the TMS320C30. Digital controllers have many advantages over analogue controllers.

- They are not affected by component ageing and temperature drift and they provide stable performance.

- When the design is done in the z-domain, the behaviour of digital controllers can be more precisely controlled.

- Digital controllers are programmable, thus making them more easy to upgrade.

- They can be timeshared to implement different functions in the system, like notch filters and system control,thus reducing system cost.

The controller is designed using two schemes, one directly in the z-plane, and the other in the continuous domain which is converted into a digital form. The speed of the TMS320C30 processor is very fast, so a short sample time can be selected such as 100 μs or less. Due to the small interval, the delay of the ZOH (zero-order hold) can almost be neglected and so the behaviour of the system is similar to that of an analogue system. Obviously, if the sampling rate is fast enough, the sample period approaches zero, and the speed response of the digital control system approaches that of the continuous system. Therefore we use a design scheme, "analogue design of discrete controller", which is appropriate for the TMS320C30 due to its fast speed. The design of a speed controller using well known analogue control technology is described first. The controller is then transformed into a digital form using the Tustin's transformation [3-2]. An alternative speed controller using a direct digital design is also described. This controller uses a feedback PID implementation in which the parameters are adjusted using a graph-analytical method of pole-placement. This controller has a faster dynamic response and better performance than conventional PID controllers. Both designs are based on an AEG BHT brushless servo drive system. The modelling of the analogue drive system is detailed in appendix A. The model simplification is described in section 3.3 and this is used for both designs.

3.2 The servo system model

The servo drive selected for this project is one of the AEG BHT brushless motor series. It is a current-controlled brushless DC servo system, and as already stated it generally consists of a double closed loop control, ie, an inner current loop and an outer speed loop as shown in Fig 2-8. The whole analogue servo system model is shown in figure 3-1. The modelling procedure is described in detail in appendix A.

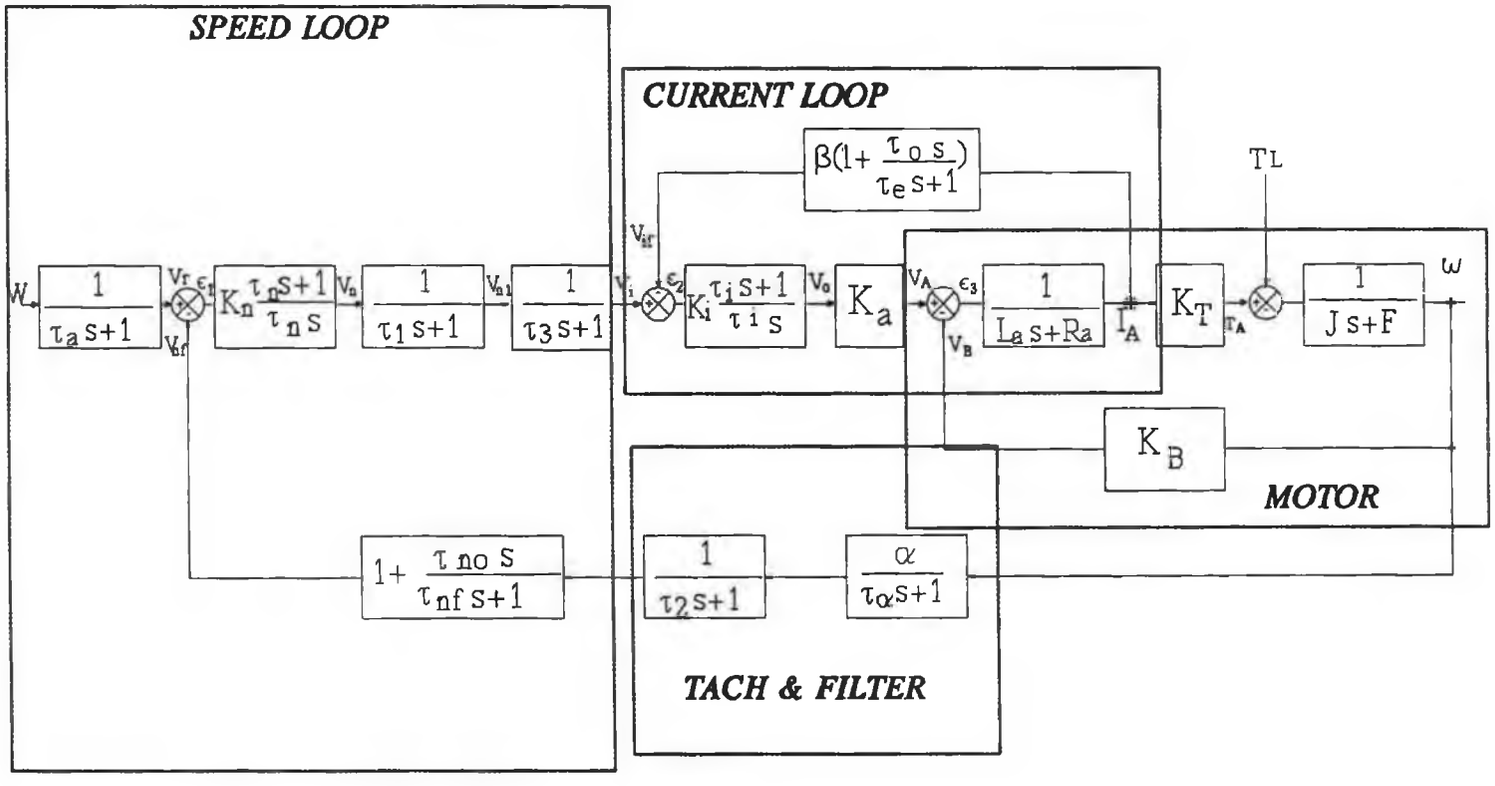


Fig. 3. 1 Block diagram of the speed servo system

3.3 Simplifying the mathematical model

For the purpose of the design, the complex system model can be simplified into an equivalent lower order transfer function.

3.3.1 Current loop simplification

First, we simplify the current loop in figure 3-1. The transfer function I_A/V_A is found from the following equations :

$$\frac{\omega}{I_A} = \frac{K_T}{Js+F} \qquad (3.3.1)$$

$$\frac{I_A}{e_3} = \frac{1}{L_a s+R_a} \qquad (3.3.2)$$

$$e_3 = V_A - V_B \qquad (3.3.3)$$

Because the back EMF of the motor is given by $V_B = \omega K_B$, equation (3.3.1)~(3.3.3) can be combined into a single equation.

$$I_A = \frac{1}{L_a s+R_a}(V_A - \omega K_B) = \frac{1}{L_a s+R_a}\left(V_A - \frac{K_T \cdot I_A}{Js+F} \cdot K_B\right) \qquad (3.3.4)$$

$$\therefore \quad \left[1+\frac{K_B K_T}{(L_a s+R_a)(Js+F)}\right] \cdot I_A = \frac{1}{L_a s+R_a} V_A \qquad (3.3.5)$$

The transfer function I_A/V_A is thus:

$$\frac{I_A}{V_A} = \frac{\frac{1}{L_a s+R_a}}{\frac{(L_a s+R_a)(Js+F)+K_B K_T}{(L_a s+R_a)(Js+F)}} = \frac{Js+F}{(Js+F)(L_a s+R_a)+K_B K_T} \qquad (3.3.6)$$

Using the motor parameters, $\tau_e = L_a/R_a$; $\tau_{em} = JR_A/K_B K_T$; $\tau_m = J/F$, equation (3.3.6) can be expressed as:

$$\frac{I_A}{V_A} = \frac{\tau_{em}(\tau_m s+1)}{\tau_m R_a\left[\tau_{em}\tau_e s^2+\tau_{em}s+\frac{\tau_e\tau_{em}}{\tau_m}s+\frac{\tau_{em}}{\tau_m}+1\right]} \qquad (3.3.7)$$

In equation (3.3.7), the τ_{em} and τ_e are very small time constant. For the servo system, $1/\tau_{em}$ and $1/\tau_e$ are much greater than the system crossover frequency ω_c, hence the term $\tau_{em}\tau_e s^2$ can be ignored. The term $(\tau_e\tau_{em})/\tau_m$ is much smaller than τ_{em} and the term τ_{em}/τ_m « 1, so they can also be ignored, and equation (3.3.7) can be simplified as:

$$\therefore \quad \frac{I_A}{V_A} \doteq \frac{\tau_{em}(\tau_m s+1)}{\tau_m R_a(\tau_{em}s+1)} \qquad (3.3.8)$$

This represents the motor transfer function in the current loop as shown in figure 3-2.

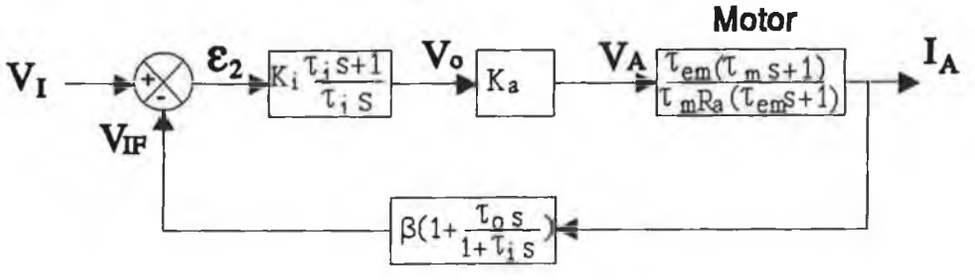


Fig. 3-2 The block diagram of the current loop

Next we can simplify the current loop in figure 3-2 into an equivalent transfer function.

$$\frac{I_A}{V_I} = \frac{\frac{K_i \cdot K_a \cdot \tau_{em}}{\tau_m R_a}(\tau_i s+1)(\tau_m s+1)}{\tau_i s(\tau_{em}s+1)+\frac{K_i \cdot K_a \cdot \beta \cdot \tau_{em}}{\tau_m R_a}[1+(\tau_0+\tau_i)s](\tau_m s+1)} \qquad (3.3.9)$$

Substituting $K=(K_i K_a \beta \tau_{em})/(\tau_m R_a)$ into equation (3.3.9) gives:

$$\frac{I_A}{V_I} = \frac{\frac{K}{\beta}(\tau_i s+1)(\tau_m s+1)}{(\tau_i\tau_m+K\tau_0\tau_{em}+K\tau_m\tau_i)s^2+(\tau_i+K\tau_0+K\tau_i+K\tau_m)s+K} \quad (3.3.10)$$

Because $\tau_i\tau_{em} << K(\tau_0\tau_m+\tau_i\tau_m)$ and $\tau_i << K(\tau_0+\tau_i+\tau_m)$, equation (3.3.10) can be approximated as:

$$\frac{I_A}{V_I} = \frac{\frac{K}{\beta}(\tau_i s+1)(\tau_m s+1)}{K(\tau_0\tau_m+\tau_i\tau_m)s^2+K(\tau_0+\tau_i+\tau_m)s+K}$$

$$= \frac{\frac{1}{\beta}(\tau_i s+1)(\tau_m s+1)}{[(\tau_0+\tau_i)s+1](\tau_m s+1)} \quad (3.3.11)$$

$$= \frac{\tau_i s+1}{\beta[(\tau_0+\tau_i)s+1]}$$

This simplified transfer function represents the current loop and the motor windings.

3.3.2 Filters and tachogenerator simplification

At this point, we substitute the parameters of BHT servo system in Appendix A into the system block diagram, as shown in figure 3-4. The transfer function of the filter in block 4 cancels the numerator of the current loop transfer function.

Block 12 and block 13 in the feedback loop can be combined and simplified as follows:

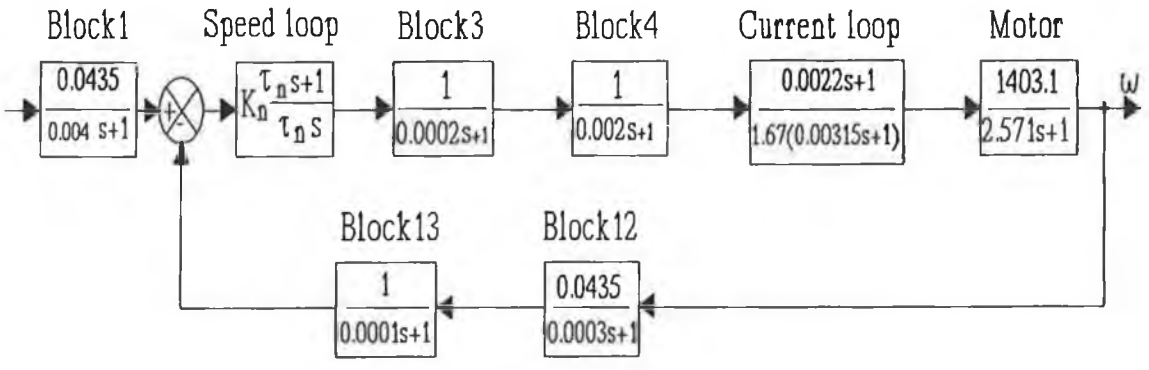


Fig. 3-4 The transfer function block diagram of the servo system

$$\frac{1}{0.001s+1}\times\frac{0.0435}{0.003s+1}$$

$$=\frac{0.0435}{3\times10^{-6}s^2+0.004s+1} \qquad (3.3.12)$$

$$\doteq\frac{0.0435}{0.004s+1}$$

where the coefficient of the term s^2, much smaller than $1/\omega_c$ (crossover frequency of the system), is neglected.

This feedback combination may be replaced by two transfer functions in the forward path as shown in figure 3-5.

The leftside transfer function, 0.0435/(0.004s+1) can be cancelled by block 1, and hence the whole servo system model can be simplified into figure 3-6.

Finally, in figure 3-6, the intermediate three blocks can be simplified to a single block

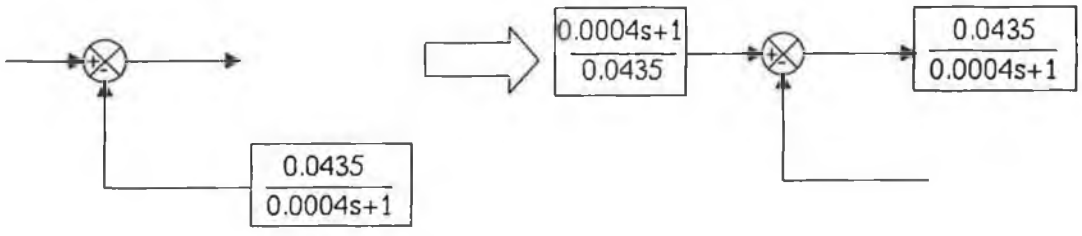


Fig. 3-5 The transferring feedback block to forward path

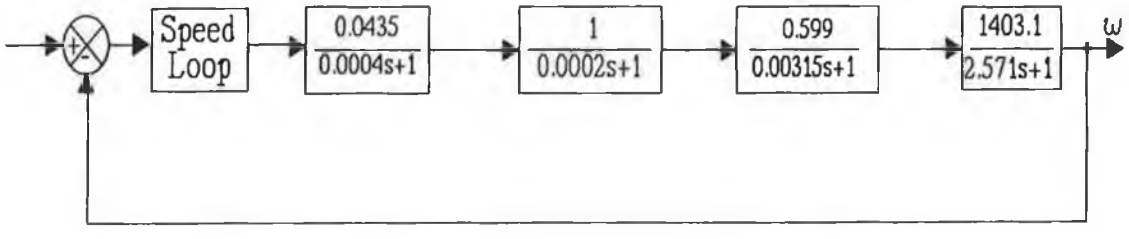


Fig. 3-6 The simplified block diagram of Fig. 3-4

as shown in figure 3-7 and described below:

$$\frac{0.0435}{0.0004S+1}\cdot\frac{1}{0.0002S+1}\cdot\frac{0.599}{0.00315S+1} \approx \frac{0.0435\times0.599}{(0.0004+0.0002+0.00315)S+1}=\frac{0.026}{0.00375S+1} \qquad (3.3.13)$$

where the second and third order terms are ignored because these time constants are much smaller than $1/\omega_c$.

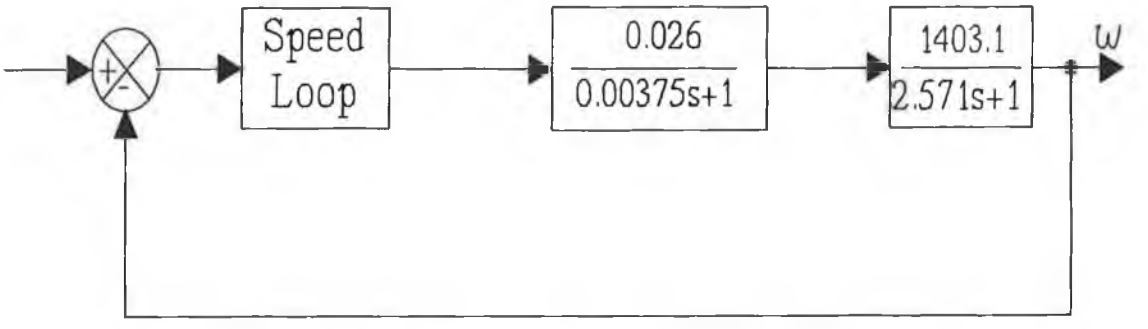


Fig. 3-7. The simplified diagram of Fig. 3-6

3.3.3 The open-loop transfer function of the servo system

The open-loop transfer function of the servo system plant of figure 3-7 is of the following form:

$$G(s) = \frac{K_d}{(\Sigma\tau s+1)(\tau_m s+1)} \qquad (3.3.14)$$

where $k_d = 0.026 \times 140.3 = 36.48$; $\Sigma\tau = 0.00375$; $\tau_m = 2.571$. Because $\tau_m >> \Sigma\tau$ and it can be simplified as:

$$G(s) = \frac{K_d}{\tau_m s(\Sigma\tau s+1)} \qquad (3.3.15)$$

This simplified system model is used for the design of the digital controllers.

3. Analogue design of the discrete controller

The actual controller can be designed in two different ways, either directly in the z-plane or indirectly in the s- plane using well-developed analogue design techniques and

then converting it into a digital form. The AEG brushless servo system has a conventional analogue PI controller which can be expressed as:

$$G_c(s) = 10\frac{0.015s+1}{0.015s}$$

This is described in detail in appendix A. The PI scheme has been developed over several decades, and there are many possible design methods. The following sections will describe two of them. One is the minimum overshoot method and the other is an optimal model method.

3.4.1 Minimum peak overshoot method [3-8]

This method selects a control law which makes the system dynamic response have a minimum peak overshoot. The design procedure is based on a frequency response technique in which the desired open-loop log magnitude characteristics is shown in figure 3-8.

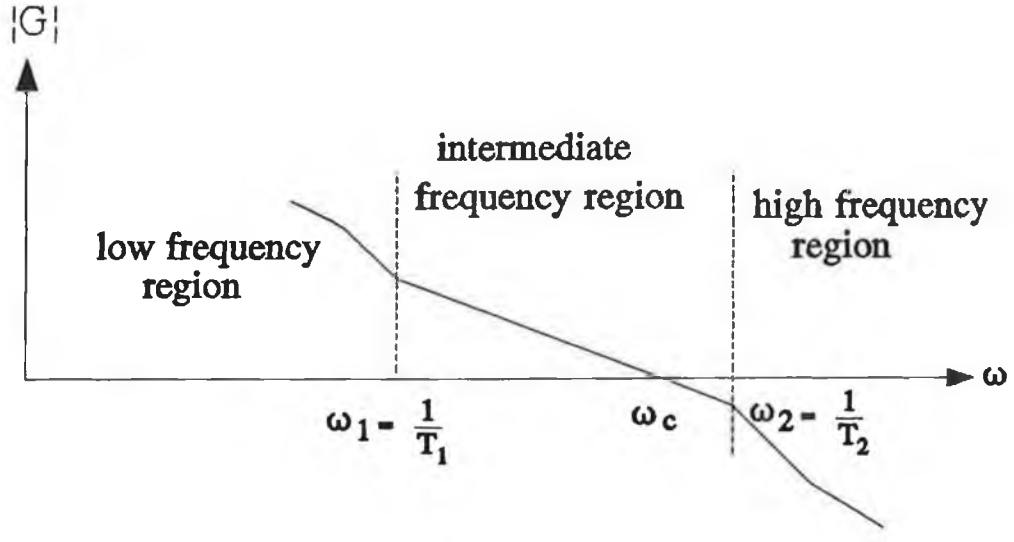


Fig. 3-8 The expected open-loop log frequency-response characteristics

For figure 3-8, we should meet the following requirements.

1. In the medium frequency region, the -20db/decade goes through zero db and should have a certain width to ensure the stability of the system.
2. The corner frequency must be big enough to give a fast system respqnse.
3. The gain in the low frequency region must be high enough to ensure static accuracy.
4. The attenuation in high frequency region must be high enough to reject noise.

It is not easy to satisfy all the above requirements owing to the contradictions among them, but if the servo system can be transformed into a typical *Type 2* system, this issue can be solved. For the *Type 2* system, the open-loop transfer function of the system is expressed as:

$$G(s) = \frac{K(\tau_1 s + 1)}{s^2(\tau_2 s + 1)} \qquad (3.4.1)$$

Its closed-loop block diagram is shown in figure 3-9.

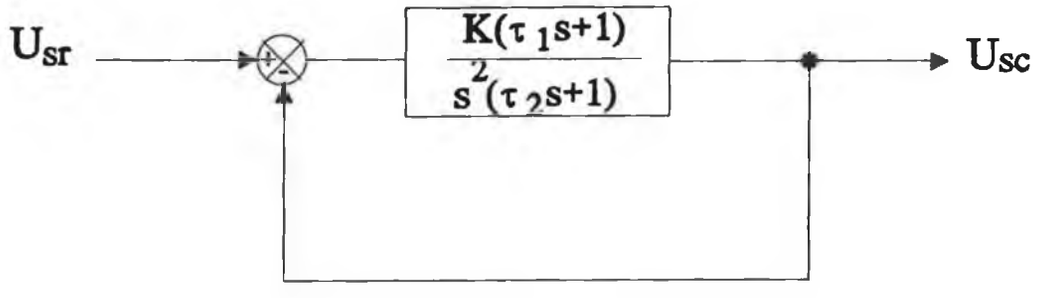


Fig. 3-9 The closed-loop block diagram of typical *Type 2* system

For the AEG BHT servo system, the plant can be compensated to fit into a *Type 2* system using the controller. The plant has a $(\Sigma\tau s+1^{-1}$ (inertia factor and a s^{-1}

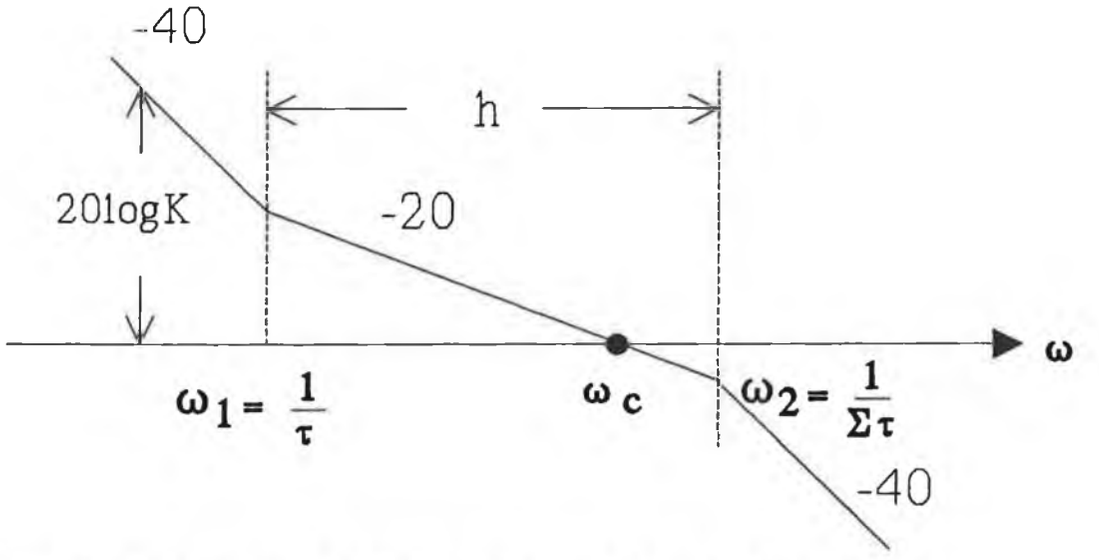


Fig. 3-10 The open-loop log magnitude curve of typical *Type 2* system

(integral factor, its transfer function uses the simplified model of the brushless servo system, as the given equation

$$G(s) = \frac{K_d}{\tau_m s(\Sigma\tau s+1)} \qquad (3.4.2)$$

The PI-controller must be selected to compensate the plant into a *Type 2* system as shown in the equation below:

$$Go(s) = G_{c1}(s)\,G(s) = K_n\frac{\tau s+1}{\tau s}\cdot\frac{K_d}{\tau_m s(\Sigma\tau s+1)} = \frac{K(\tau s+1)}{s^2(\Sigma\tau s+1)} \qquad (3.4.3)$$

where $K=(K_nK_d$

Figure 3-10 shows the log magnitude curve of the servo system according to figure 3-8. The three frequency response characteristic parameters in the *Type 2* system are $_2=1/\Sigma\tau$ and respectively. The system can be determined if these three parameters are selected. Since the $\Sigma\tau$ is the intrinsic parameter of the plant, only and $_c$ need to be adjusted. The width of the intermediary frequency region is *h*, and if we determine **h**, we can find the parameter τ from:

$$h = \frac{\omega 2}{\omega 1} = \frac{\tau}{\Sigma \tau} \qquad (3.4.4)$$

From figgure 3-10, we have

$$20\log K = 40\log\omega_1 + 20\log\frac{\omega_c}{\omega_1} = 20\log\omega_1\omega_c \qquad (3.4.5)$$

So,

$$K = \omega_1\omega_c \qquad (3.4.6)$$

Using the rule that the peak overshoot M_p of the system response is a minimum as shown in figure 3-11, we can determine the and $_c$ and find the gain K. The closed-

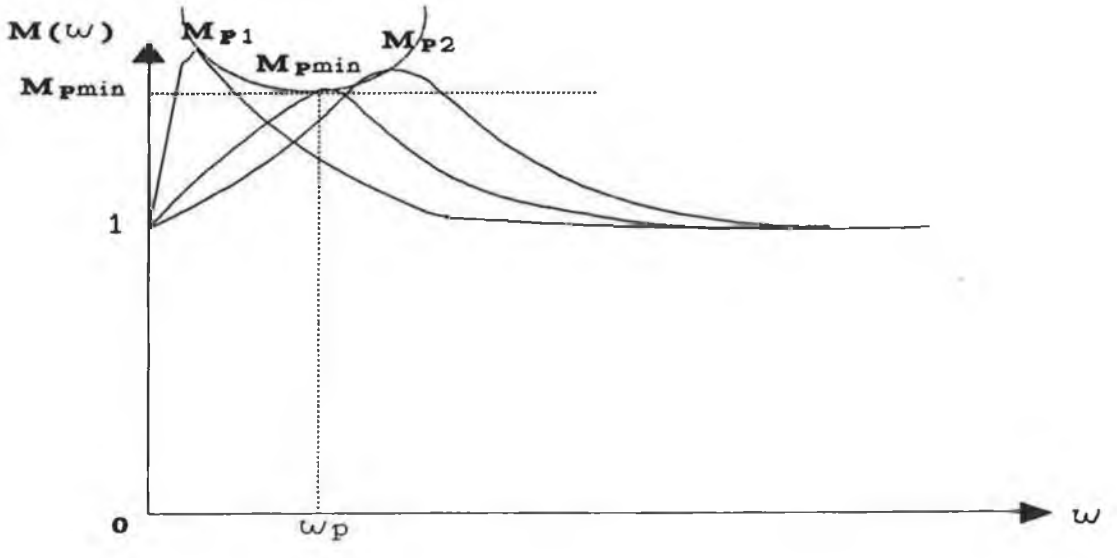


Figure 3-11

loop transfer function of the *type 2* system can be expressed as:

$$G_B(s) = \frac{G(s)}{1+G(s)} = \frac{K(h\Sigma\tau s+1)}{\Sigma\tau s^3+s^2+Kh\Sigma\tau s+K} \qquad (3.4.7)$$

The frequency response form of equation (3.4.7 is:

$$G_B(j\omega)=\frac{K(1+jh\Sigma\tau\omega)}{(K-\omega^2)+j(Kh\Sigma\tau-\Sigma\tau\omega^2)\omega} \qquad (3.4.8)$$

where the overshoot can be expressed by the amplitude of $G_B(j\omega$

$$M(\omega,K) = |G_B(j\omega)|$$

$$= \frac{K\sqrt{1+h^2\Sigma\tau^2\omega^2}}{\sqrt{\Sigma\tau^2\omega^6+(1-2Kh\Sigma\tau^2)\omega^4+(K^2h^2\Sigma\tau^2-2K)\omega^2+K^2}} \qquad (3.4.9)$$

When $\frac{\partial M}{\partial}=0$, we can obtain M_P as a function of K, and letting $\frac{\partial M_P}{\partial K}=0$ we can

obtain the minimum peak overshoot M_{Pmin} of the closed-loop system as shown in figure 3-11. At the point of the M_{pmin}, there are optimal ratios of ω_2 / ω_c and / which are expressed as functions of h [3-8]:

$$\frac{\omega_2}{\omega_c} = \frac{2h}{h+1} \qquad (3.4.11)$$

$$\frac{\omega_c}{\omega_1} = \frac{h+1}{2} \qquad (3.4.12)$$

The M_{pmin} with respect to h can be expressed as:

$$M_{pmin} = \frac{h+1}{h-1} \qquad (3.4.13)$$

By selecting different values of h, the corresponding M_p , and / values can be derived and are given in table 3.4.1 [3-8].

Table 3.4.1 M_{pmin} and optimal frequency ratio for different values of h

h	3	4	5	6	7	8	9	10
M_{pmin}	2	1.67	1.5	1.4	1.33	1.29	1.25	1.22
ω_2/ω_c	1.5	1.6	1.67	1.7	1.75	1.78	1.80	1.82
ω_c/ω_1	2	2.5	3.0	3.5	4.0	4.5	5.0	5.5

From the experience[3-8], we know that when M_{pmin} is between 1.2 to 2.0, the dynamic performance of the system is reasonable. So the value of h can be selected between 3 to 10 in table 3.4.1. From equation (3.4.4 and (3.4.12 we can obtain:

$$K = \frac{h+1}{2h^2\tau_2^2} \qquad (3.4.14)$$

From table 3.4.1, we select $M_{pmin}=1.5$ and using equation (3.4.13 h=5 can be found. From the open-loop transfer function of the servo system, we have $\Sigma\tau$ =0.00375. Using equation (3.4.4 we can obtain

$$\tau = h\cdot\Sigma\tau = 5\times 0.00375 = 0.0187 \qquad (3.4.15)$$

From equation (3.4.14 K can be obtained as:

$$K = \frac{h+1}{2h^2\Sigma\tau^2} = \frac{5+1}{2\times 5^2(0.00375)^2} = 8533.3 \qquad (3.4.16)$$

Then we can find K_n of the speed loop from equation (3.4.3

$$K_n = \frac{8533.3\times 0.0187}{14.24} = 11.2 \qquad (3.4.17)$$

The parameters of the analogue PI controller has been derived, with K_n=**11.2** and τ=**0.0187**, and it can be expressed as:

$$G_{c1}(s) = 11.2\frac{0.0187s+1}{0.0187s}$$

3.4.2 "Optimiing model" method [3-9]

The "optimizing model" method is to regulate the uncompensated system into an optimal third order system model. This system has the fastest settling time and minimum steady state error. For our permanent magnetic brushless servo system, the system model is regulated into an optimal third order model using a PI controller, as shown in figure 3-12.

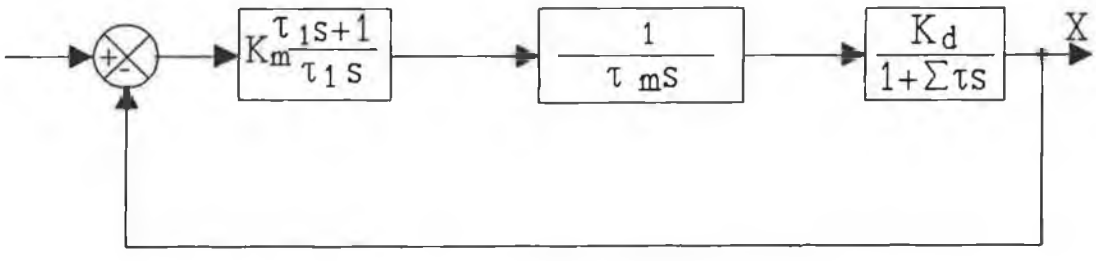


Fig. 3-12 Typical third order systems

The open-loop transfer function of the system is expressed as:

$$G_o(s) = K_m\left(1+\frac{1}{\tau_1 s}\right)\cdot\frac{1}{\tau_m s}\cdot\frac{K_d}{1+\Sigma\tau s} \qquad (3.4.22)$$

The closed-loop transfer function is:

$$G(s) = \frac{G_o(s)}{1+G_o(s)} = \frac{(1+\tau_1 s)}{1+T_1 s+T_2 s^2+T_3 s^3} \qquad (3.4.23)$$

where,

The dynamic response is determined by the coefficients T_1, T_2 and T_3 and when $T_2=T_1^2/2$ and $T_3=T_1^3/8$, the system has the optimal form [3-9]. From equation (3.4.24 and the above conditions, we can obtain:

$$T_1 = \tau_1$$

$$T_2 = \frac{\tau_1 \tau_m}{K_m K_d} \qquad (3.4.24)$$

$$T_3 = \frac{\tau_1 \tau_m \Sigma\tau}{K_m K_d}$$

$$\tau_1 = 4\Sigma\tau; \quad K_m = \frac{T_m}{2\Sigma\tau K_d} \qquad (3.4.26)$$

Substituting equation (3.4.26 and (3.4.24 to equation (3.4.23 we have the optimal third order transfer function:

$$G(S) = \frac{1+4\Sigma\tau s}{1+4\Sigma\tau s+8\Sigma\tau^2 s^2+8\Sigma\tau^3 s^3} \qquad (3.4.27)$$

This third order system has optimal dynamic characteristics with a step input signal, where the transient response of its output is expressed as:

$$f(t) = 1+e^{-\frac{t}{2\Sigma\tau}} - \frac{2}{\sqrt{3}} e^{-\frac{t}{4\Sigma\tau}} \sin\left(\frac{\sqrt{3}}{4\Sigma\tau} t\right) \qquad (3.4.28)$$

The response curve is illustrated in figure 3-13,where the time to first zero error $t_0=7.6\Sigma\tau$; the maximum overshot $\Delta X_{max}=8.1\%$ and the settling time $t_s=16.5\Sigma\tau$. Using the optimal third order system, we can obtain the PI speed controller for the BHT servo system. We know the simplified open-loop transfer function of the servo system plant is equation (3.3.15 The parameters are $\tau_m=2.57$, $\Sigma\tau=0.00375$, and $K_d=36.6$. Substituting these parameters into equation (3.4.26 we obtain:

$$K_m = \frac{\tau_m}{2\Sigma\tau K_d} = \frac{2.57}{2\times 0.00375\times 36.6} = 9.36 \qquad (3.4.29)$$

$$\tau_1 = 4\Sigma\tau = 4\times 0.00375 = 0.015$$

So, the PI controller can be expressed as:

$$G_{c2}(s) = 9.36\frac{0.015s+1}{0.015s}$$

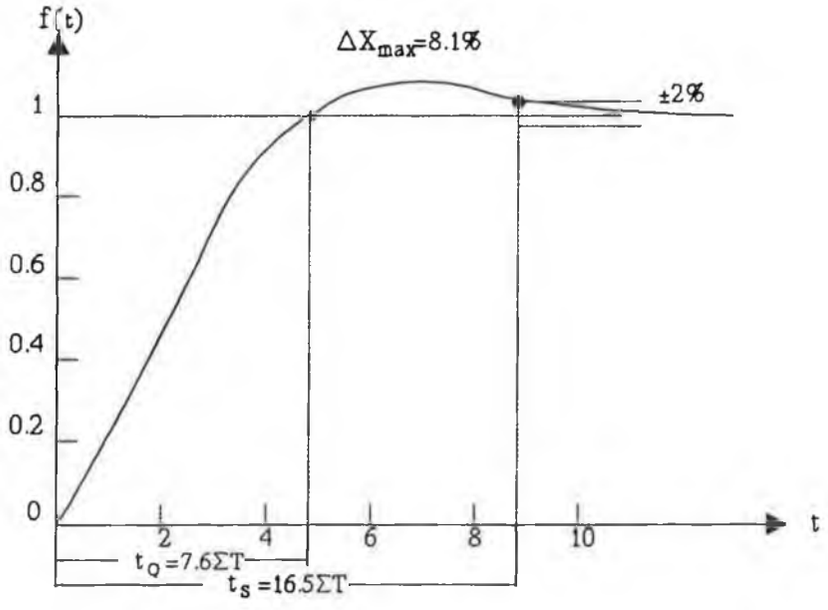


Fig. 3-13 The response curve of optimal third order system

Comparing both designed PI speed controllers, we can see that they are very similar. They are also nearly same as the speed loop of the BHT system in appendix 1 which proves that the analogue controller of the AEG system is well designed.

3.4.3 Discretiation of the analogue controller

The simplest method of obtaining the digital speed controller is directly to discretize the analogue controller to a digital form. However, due to the digital controller having a zero order hold, the accuracy and stability of the system can be effected. Fortunately, since we use the very fast DSP-TMS320C30, this can be greatly reduced. The design procedure is as follows:

(1 Design the controller G_c(s in the continuous-time domain.

(2 Determine the delay of the series connection of the leading sampler and the

ZOH (zero order hold which is approximated by a continuous transfer function G_{ap}:

$$G_{ap}(s) = T^{-1}\frac{2T}{Ts+2} = \frac{2}{Ts+2} \qquad (3.4.30)$$

Where T^{-1} represents the transfer function of the sampler and $2T(Ts+2^{-1}$ is Pade's first approximation of the ZOH.

(3 Find the digital controller D(s in the continuous-time domain.

$$D(s) = \frac{G_c(s)}{G_{ap}(s)} \qquad (3.4.31)$$

(4 Transform the adopted controller D(s transfer function to the z-plane by using Tustin's transformation [3-2].

$$[D(s)]_{Tu.} \rightarrow D(z) \qquad (3.4.32)$$

(5 A combination of the continuous-time controller design and Tustin's transformation may yield a physically unrealizable system if there are poles at z=1. It is usual to replace the excessive poles (z+1 by their d.c. gain as $(z+1)_{z=1}=2$.

(6 It may be necessary to adjust the open-loop transfer function gain in order to meet the design requirements precisely.

Step 1: We have already obtained the analogue speed controller for the AEG BHT servo system in the previous sections.

$$G_{c2}(s) = 9.36\frac{(0.015s+1)}{0.015s}$$

Step 2: We find the transfer function of zero-order hold, $G_{ap}(s$ using equation (3.4.30 Since the TMS320C30 executes 1 instruction in 60 ns, if we select the sample period T=100 μs (refer to section 3.5.3.1 the TMS320C30 DSP can executes 1666 instructions, this is adequate for real time control of the brushless drive controller. The $G_{ap}(s$ is:

$$G_{ap}(s) = \frac{2}{Ts+2} = \frac{2}{0.0001\ s+2} = \frac{20000}{s+20000} \qquad (3.4.33)$$

Step 3: $D_c(s$ is

$$D_c(s) = \frac{G_c(s)}{G_{ap}(s)} = \frac{4.68\times10^{-4}(s+20000)(s+66.67)}{s} \qquad (3.4.34)$$

Step 4: Using Tustin's transformation, we can obtain D(z for the discrete controller.

$$D_c(z) = [D_c(s)]_{Tu} = \frac{18.78(1-0.99335z^{-1})}{(1+z^{-1})(1-z^{-1})} \qquad (3.4.35)$$

Step 5: There is a pole z= -1 in D(z We need to substitute the pole by the gain $(1+z^{-1})_{z=1} = 2$ as this pole can create a very oscillatory respond of the system.

$$D_c'(z) = \frac{9.39(1-0.99335z^{-1})}{1-z^{-1}}$$

$$\frac{U(z)}{E(z)} = \frac{9.39(1-0.99335z^{-1})}{1-z^{-1}} \qquad (3.4.36)$$

Step 6: Inverse *z*-transformation, a computer implementation form of the control law is given as:

$$\frac{u(t)}{e(t)} = \mathcal{Z}^{-1}\{\frac{9.39(1-0.99335z^{-1})}{1-z^{-1}}\} \qquad (3.4,37)$$

$$u(n) = u(n-1) + 9.39[e(n)-0.99335e(n-1)]$$

Equation (3.4.37 can be used as the algorithm of the digital controller for the simulation and the DSP application.

3. Design of a digital controller using the ole-laceent technique

The discrete design of a digital controller is a more straightforward design procedure than that of discretizing an analogue controller. In this section, we describe a direct discrete speed controller design for the permanent magnetic brushless servo system using a forward and feedback PID digital regulator. The synthesis of the controller parameters is carried out by a pole-placement procedure. The suggested grapho-analytical method of pole-placement, applicable to third-order digital control systems, enables us to readily obtain information about effects of system constants and to adjust parameters based on accuracy, speed of response, and stability margin. The design procedure is as follows:

(1 Set the design requirements ;

(2 Design the motor speed control system;

(3 Select parameters of the controller using grapho-analytical method of pole-placement.

3..1 Design reuirements

Design requirements are:

(1 zero steady-state error to the unit step input;

(2 relative damping coefficient ζ=0.456, ie, maximum overshoot M_m=20%;

(3 time to peak about 20 ms.

3..2 Design the speed digital controller

Figure 3-14 shows the block diagram of a digital control system. The D/A converter converts the digital output of the microcomputer, u(n into an analog signal, u(t The output of the motor, c(t is measured by a sensor and converted into digital signal, c(n by the A/D. The reference signal R(n is a command to digital controller.

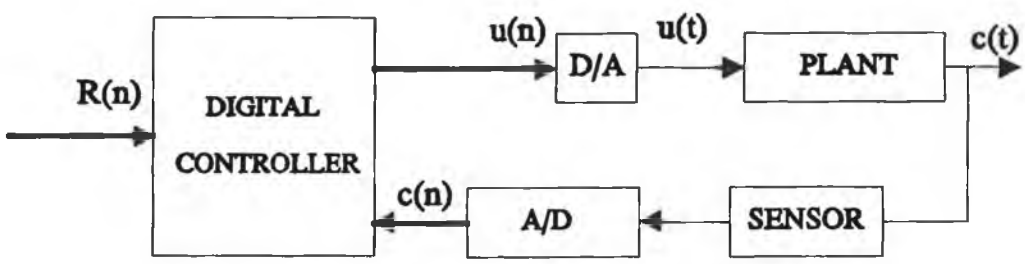


Fig. 3-14 Digital control system with cascade digital controller

The digital controller uses a forward and feedback PID compensation as shown in figure 3-15.

The integral term, as the forward compensation, ensures zero steady-state error. The proportional and derivative terms are the feedback compensators. The proportional component scales the input-output relationship while the derivative term controls the speed of the system response to any (including the initial change of the set point. The controller algorithm of the PID forward and backward compensation can be written in a velocity form [3-11]:

$$U(z) = -K_P C(z) + K_I \frac{R(z) - C(z)}{1 - z^{-1}} - K_D (1 - z^{-1}) C(z) \qquad (3.5.1)$$

where C(Z and R(Z are output and input of the speed loop, the coefficients K_P, K_I and

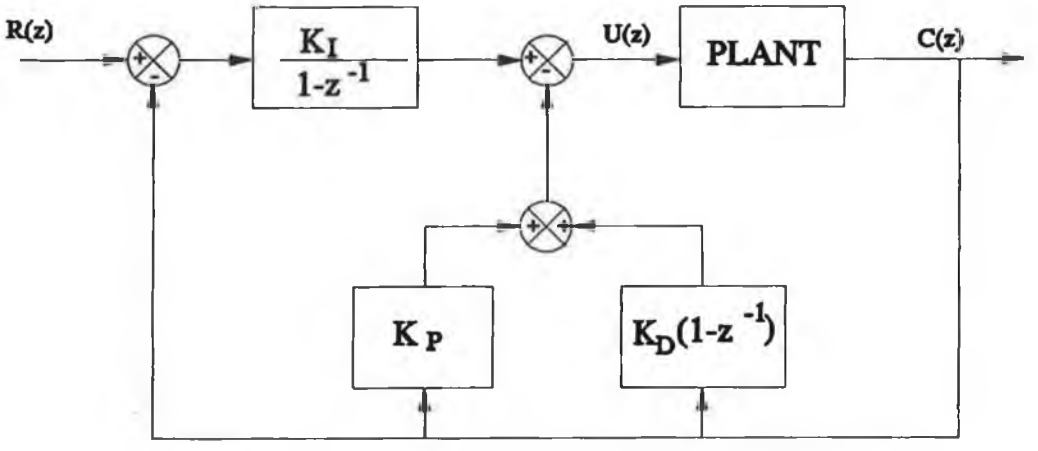


Fig. 3.15 The block diagram of speed loop

K_D are the proportional, integral and derivative gains. Notice that in equation (3.5.1 only the integral control term involves the input R(z Hence, the integral term cannot be excluded from the digital controller. Normally, in a PID control, the controller variables (K_P, K_I, K_D must be determined experimentally, and a number of trial and error attempts need to be done. However, in this thesis, we introduce a grapho-analytical method of pole-placement to adjust the parameter of the controller to get the desired accuracy, speed response and stability margin of the system.

3..3 Selection of controller parameters

3.5.3.1 The selection of sampling interval

In digital control system, analog signals are sampled every sampling period T. If the sample frequency is sufficiently high compared with the highest-frequency component involved in the continuous-time signal, the amplitude characteristics of the continuous-time signal may be preserved in the envelope of the sampled signal. In order to reconstruct the digital signal from a sampled signal, there is a certain minimum frequency that the sampling operation must satisfy

$$\omega_s > 2\omega_1 \qquad (3.5.2)$$

Where ω_s is the sampling frequency, which is defined as $2\pi/T$.

ω_1 is the highest-frequency component present in the continuous-time signal x(t Normally, the sample time should be less than the smallest time constant in the plant model, by factor 30 to 100 minimum [3-10]. For the BHT servo system the smallest time constant is 3.75 ms, the sample interval T should be selected between 0.0375ms~0.2ms. For our application, we selected T=0.1 ms.

3.5.3.2 The characteristic equation of the closed-loop system

We design the digital speed controller using the continuous parts of the brushless motor servo system as the plant. The transfer function of the continuous part can be obtained from equation (3.3.15 and it is expressed as:

$$G_p(s) = \frac{K_d}{T_m s(\Sigma Ts+1)} \qquad (3.5.3)$$

The transfer function of zero-order hold can be expressed as:

$$G_{ZOH}(s) = \frac{1-e^{-Ts}}{s} \qquad (3.5.4)$$

We transfer the plant to digital form by using the z-transformation.

$$G_p(z) = \zeta[G_{ZOH}(s)\,G_p(s)] = \frac{K(z+1)}{(z-P_1)(z-P_2)} \qquad (3.5.7)$$

where K=0.00186 P_1=1 and $P_2=e^{-\Sigma T\cdot T}$=0.947.

The closed-loop transfer function of the system is:

$$\frac{C(z)}{R(z)} = \frac{K_I z^2 G_p(z)}{(z-1)z+\{K_p z(z-1)+K_I z^2+K_D(z-1)^2\}G_p(z)} \qquad (3.5.8)$$

The characteristic equation of closed loop system is given by:

$$z(z-1)+[K_P z(z-1)+K_I z^2+K_D(z-1)^2]G_p(z) = 0 \qquad (3.5.9)$$

Substituting $G_p(z$ from equation (3.5.7 into (3.5.9 and then simplifying (3.5.9 into the polynomial form, one can obtain, after simple manipulations

$$z^4 + a_3 z^3 + a_2 z^2 + a_1 z + a_0 = 0 \qquad (3.5.10)$$

where

$a_3=K(K_P+K_I+K_D$

$a_2=p_1p_2+(p_1+p_2$ D

$a_1=KK_D-K(K_P+2K_D$

$a_0=KK_D$

Notice that: when $K_D=0$, the coefficient a_0 is zero and the system characteristic equation becomes

$$z^3+A_2 z^2+A_1 z+A_0= 0 \qquad (3.5.11)$$

where $A_2=K(K_P+K_I$ $_1+p_2$ I

$A_1=p_1p_2+(p_1+p_2$

$A_0=-p_1p_2-KK_P=-0.974-0.0186K_P$

The characteristic equation (3.5.11 is a third-order equation. The proper root-set of this equation can be achieved by employing the charts in figure 3-17 or figure 3-18. The construction of the charts and their application in solving the pole placement problem for the third-order digital control system are explained in detail in the following section.

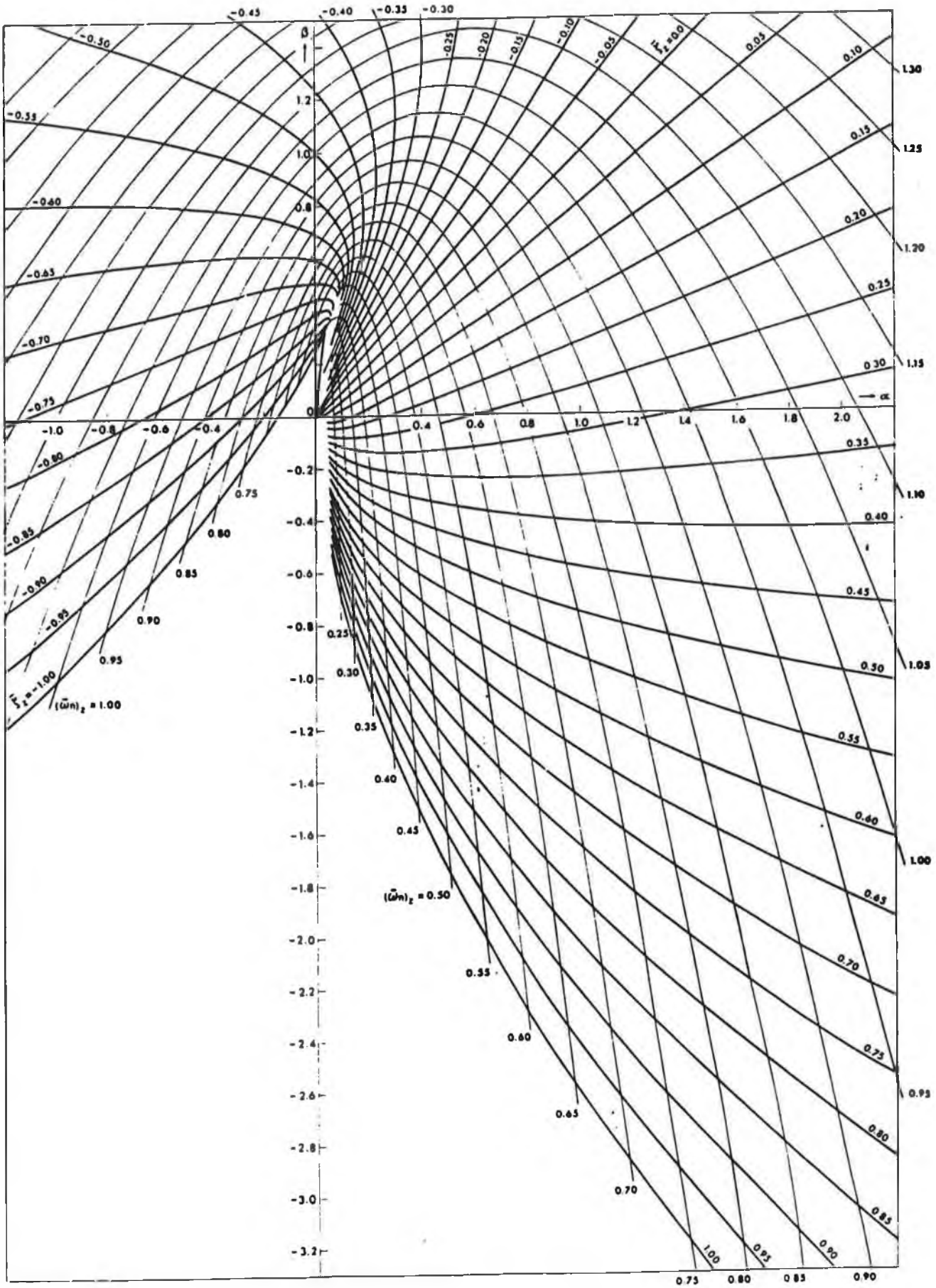


Figure 3-17 Root loci of complex-poles in the parameter plane

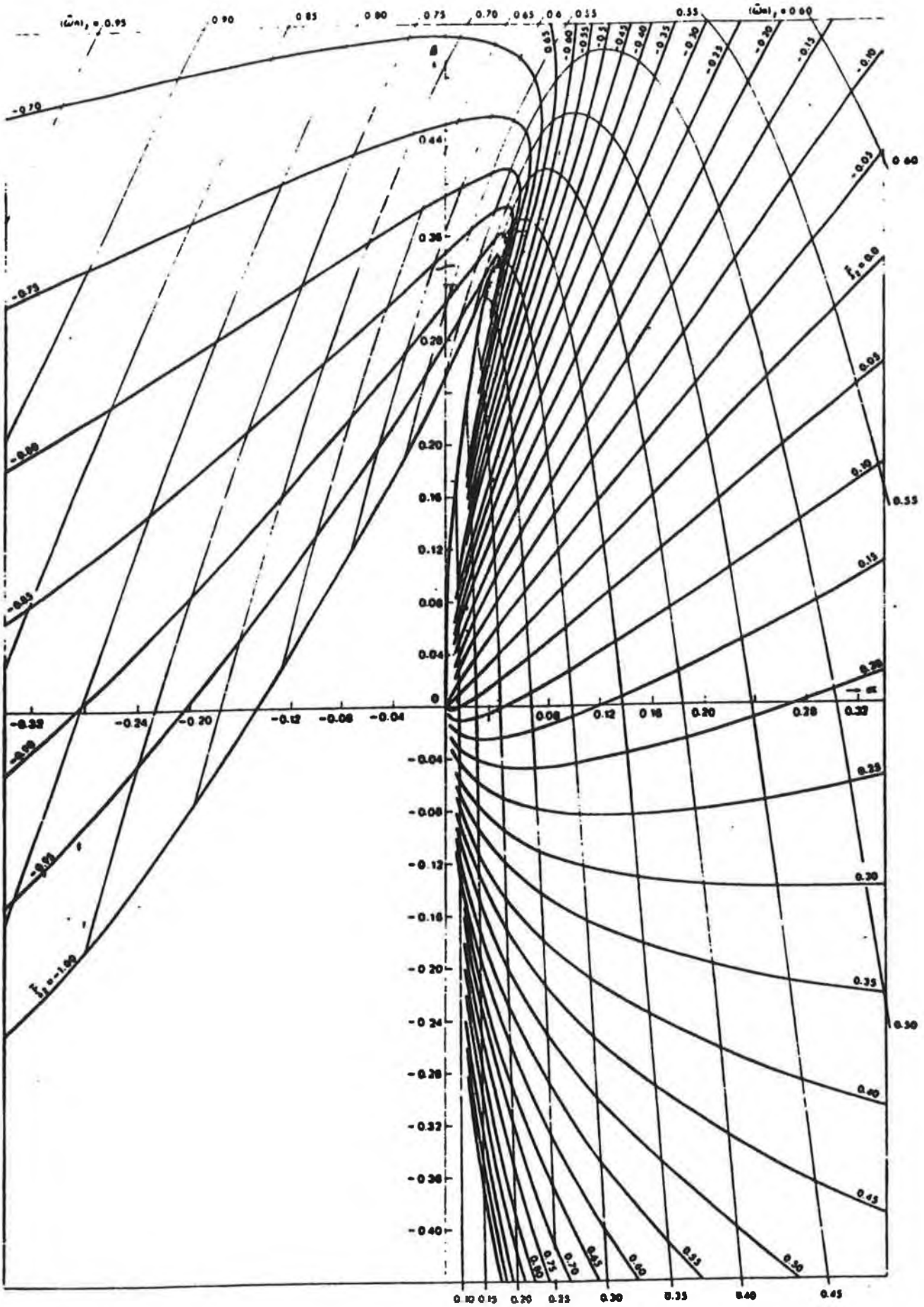


Figure 3-18 Enlarged region of figure 3-17 around the origin

3.5.3.3 Grapho-analytical method of pole-placement[3-6]

We introduce the new complex variable $\overline{Z}$ by putting $Z=A_2\ \overline{Z}$ into equation (3.5.11 to obtain:

$$\overline{Z}^3+\overline{Z}^2+\beta\overline{Z}+\alpha = 0 \qquad (3.5.12)$$

where only two variables coefficients appear:

$$\alpha = \frac{A_0}{A_2^3}, \qquad \beta = \frac{A_1}{A_2^2} \qquad (3.5.13)$$

In order to plot the root loci of complex-poles of the parameter plant, we set $\overline{Z}=(\overline{\omega}_n)_z\, e^{j\overline{\Psi}}=(\overline{\omega}_n)_z\overline{\xi}_z+j(\overline{\omega}_n)_z\sqrt{1-\overline{\xi}_z^2}$, with $\overline{\Psi}=\arccos\overline{\xi}_z$ and $-1\le\ \overline{\xi}_z\ \le 1$, and put it into equation (3.5.12 then separate real and imaginary parts of this equation, and equate both parts to zero. We can obtain two simultaneous equations in two unknown variables α and β:

$$(\overline{\omega}_n)_z^3 T_3(\overline{\xi}_z) + (\overline{\omega}_n)_z^2 T_2(\overline{\xi}_z) + \beta(\overline{\omega}_n)_z T_1(\overline{\xi}_z) + \alpha T_0(\overline{\xi}_z) = 0 \quad (3.5.14)$$

$$(\overline{\omega}_n)_z^3 U_3(\overline{\xi}_z) + (\overline{\omega}_n)_z^2 U_2(\overline{\xi}_z) + \beta(\overline{\omega}_n)_z U_1(\overline{\xi}_z) + \alpha U_0(\overline{\xi}_z) = 0 \quad (3.5.15)$$

where $T_K(\overline{\xi}_z)$ *and* $U_K(\overline{\xi}_z)$ are the Chebyshev functions of the first and second kind, respectively. [3-6]

Substitute the Chebyshev function

$$T_k(\overline{\xi}_z) = \overline{\xi}_z U_k(\overline{\xi}_z) - U_{k-1}(\overline{\xi}_z) \qquad (3.5.16)$$

into equation (3.5.14 and we obtain

$$(\overline{\omega}_n)_z^3 U_2(\overline{\xi}_z) + (\overline{\omega}_n)_z^2 U_1(\overline{\xi}_z) + \beta(\overline{\omega}_n)_z U_0(\overline{\xi}_z) + \alpha U_{-1}(\overline{\xi}_z) = 0 \quad (3.5.17)$$

Equations (3.5.15 and (3.5.17 can be solved easily for variable coefficients α and β to obtain:

$$\alpha = (\overline{\omega}_n)_z^2[(\overline{\omega}_n)_z U_2^*(\overline{\xi}_z) + U_1^*(\overline{\xi}_z)]$$
$$\beta = -(\overline{\omega}_n)_z[(\overline{\omega}_n)_z U_3^*(\overline{\xi}_z) + U_2^*(\overline{\xi}_z)] \qquad (3.5.18)$$

where $|\overline{\xi}_z|<1$ and $U_1^*(\overline{\xi}_z)$, $U_2^*(\overline{\xi}_z)$, and $U_3^*(\overline{\xi}_z)$ are the Chebyshev polynomials of the second kind[3-18] defined by

$$U_k^*(\overline{\xi}_z) = \frac{\sin(k\cdot\arccos\overline{\xi}_z)}{\sqrt{1-\overline{\xi}_z^2}}$$

Using equation (3.5.18 two families of curves with respect to different values of $\overline{\xi}_z$ = constant and $(\overline{\omega}_n)_z$ = constant can be plotted as shown in figure 3-17.

Figure 3-18 shows the enlarged region of figure 3-17 around the origin. The curves in figure 3-17 and figure 3-18 are in the (α,β that represent the loci of complex poles of equation (3.5.12 as the parameters α and β vary. Using figure 3-16 and figure 3-17, we can easily use a pole placement technique for the third-order polynomials of the feedback control system.

To relate the z-plane and the $\overline{z}$ -plane, we substitute

$z = (\omega_n)_z\xi_z + j(\omega_n)_z\sqrt{1-\xi_z^2}$ and according to the relation $z=A_2\overline{z}$, we can obtain :

$$(\omega_n)_z = |A_2|(\overline{\omega}_n)_z \qquad (3.5.19)$$

$$\xi_z = (sgn(A_2))\overline{\xi}_z \qquad (3.5.20)$$

where equation (3.5.19 and (3.5.20 are the parameters defining the complex-pole

locations in the planes of the complex variables z and $\bar{z}$.

Since $z=e^{sT}$, the parameters (ω_n and ζ_z, which determine the corresponding complex-pole locations in the z-plane, can be obtained from the relative undamped natural frequency ω_n and the damping coefficient ζ :

$$(\omega_n)_z = e^{-\omega_n \xi T} \quad and \quad \xi_z = \cos(\omega_n T\sqrt{1-\xi^2}) \qquad (3.5.22)$$

where, the Nyquist frequency band is:

$$0 \le \omega_n \le \frac{\pi}{T\sqrt{1-\xi^2}} \qquad (3.5.23)$$

Therefore, in the proposed procedure for the synthesizing controller parameters, it is necessary first to specify the relative undamped natural frequency ω_n and the damping coefficient ζ from the desired system characteristics with respect to the stability margin and the speed of the continuous-time response. Then using equations (3.5.22 (3.5.19 and (3.5.20 the parameters $\bar{\ }$ *and* $\bar{\xi}$ can be determined. Then the loci diagram of pole-placement can be employed to find the intersection point $M(\alpha_m,\beta_m$ of the curve $\bar{\xi}_z$ and $(\bar{\ }_n)_z$ hence the respective parameters α_m and β_m can be obtained.

According to the Viète rule, the third real root of equation (3.5.12 can be calculated from :

$$\bar{\sigma}_z = -1-2\bar{\xi}_z(\bar{\omega}_n)_z \qquad (3.5.24)$$

The stability condition requires that all roots of equation (3.5.11 are inside the unit circle; hence, we select A_2, the absolute value of A_2 must satisfy the constraints ($\omega_n < 1$ and $\sigma_z < 1$,

or

$$|A_2| < \frac{1}{|\bar{\sigma}_z|} \quad and \quad |A_2| < \frac{1}{(\bar{\omega}_n)_z} \qquad (3.5.25)$$

The other two coefficients A_1 and A_0 in equation (3.5.11 can be obtained by using equation (3.5.13

$$A_1 = \beta_m A_2^2 \quad and \quad A_0 = \alpha_m A_2^3 \qquad (3.5.26)$$

Putting the numerical values of A_2, A_1, A_0 into equation (3.5.11 the parameter K_P, and K_I can be obtained. K_D can be adjusted later on.

3.5.3.4 Parameter calculation

From the design requirements, we know the damping coefficient ζ=0.456 and the time to peak t_m is 20 msec. Using equation (3.5.27

$$t_m = \frac{\pi}{\omega_n\sqrt{1-\zeta^2}} = 0.02 \qquad (3.5.27)$$

we can obtain the undamped natural frequency:

$$\omega_n = 176.2 \ rad/sec \qquad (3.5.28)$$

The Nyquist frequency band is specified by equation (3.5.23 and the respective value is 0< 176.2 <35300. From equation (3.5.22 the relationship between the $_n$ and ζ from the continuous-time response, and the parameters (and ζ_z determine corresponding complex-pole locations in the z-plane which are readily obtained as

$$(\omega_n)_z = e^{-\omega_n\zeta T} = e^{-176.5\times0.456\times0.0001} = 0.992 \quad \text{and}$$

$$\zeta_z = \cos(\omega_n T\sqrt{1-\zeta^2}) = \cos(176.2\times0.0001\sqrt{1-0.456^2}) = 0.9998 \ .$$

Meeting the stability condition $\omega_n < 1$ and $\sigma_z < 1$, A_2=-0.2945 can be selected. From equation (3.5.19 and (3.5.20 we can obtain $\bar{\xi}_z$=-0.9998 *and* ($\bar{\ }_z$)=0.34 .

Once $\bar{\zeta}_z$ is determined, along the curve $\bar{\zeta}_z$ = -0.9998=const of figure 3-16 or 3-17 the point M(0.04,0.34 corresponding to the desired (can be placed. The coefficients A_1 and A_0 in equation (3.5.11 are obtained by using equation (3.5.13

$$A_0 = \alpha_m \cdot A_2^3 = 0.04 \times (-2.954)^3 = -1.02$$

and

$$A_1 = \beta_m \cdot A_2^2 = 0.34 \times (-2.945)^2 = 2.9488$$

Substituting the numerical values of A_2, A_1 and A_0 into (3.5.11 and solving equation (3.5.11 for the controller parameters, we finally obtain

$$K_P = \frac{-A_0 - p_1 p_2}{K} = \frac{1.02 - 0.974}{0.00186} = 24.7$$

$$K_I = \frac{A_1 - (p_1 + p_2 + p_1 p_2)}{K} = \frac{2.9488 - (1 + 0.947 + 1 \times 0.947)}{0.00186} = 0.4$$

The parameter K_D has a small effect on the dynamic performance and it can be adjusted in the practical controller later. The PID control algorithm can be rewritten by three equations for DSP implementation.

$$e(kT0 = r(kT - c(kT$$
$$u_1(kT = K_I e(kT + u_1[(k-1$$
$$u(kT = -K_P c(kT - K_D\{c(kT - c[(k-1 + u_1(kT$$

where e(kT and u_1(kT are the sample-data of the error signal and the output of the cascade integral compensator, respectively.

3. Conclusion

This chapter has described two designs of a digital speed controller for a brushless servo drive. First, we described the 'analogue method of digital controller design' using the well-known analogue technologies in which the analogue controller with the traditional PI implementation was converted into digital form which can be implemented by the TMS320C30. This method makes the digital design become simple since we can directly utilize the existing analogue controller. A more accurate design of a direct-digital PID controller was also described. The method simplified the digital system into a third-order digital feedback control system in which the controller parameters were adjusted by using a graph-analytical method of pole-placement technique. Using this method, the steady-state error of the system was eliminated and the controller parameters were adjusted to give the desired accuracy, response speed, and stability margin of the system, respectively. This controller has a faster response and greater accuracy than the first traditional PI control but is more complex. The experimental results in chapter 7 show its advantages.

CHAPTER 4

SIMULATION OF BRUSHLESS SERVO SYSTEM

4.1 Introduction

In order to study the dynamic performance of the brushless servo system before a practical test, a computer simulation scheme is used to model the prototype system. This chapter introduces a simulation scheme for the brushless drive servo system, which uses a block-oriented transfer function program written in the C langauge. Using this scheme, we can examine the dynamical behaviour based on the system block diagram.

The brushless servo system can be described by a set of transform function blocks as shown in figure 4-1 which is divided into two parts, the digital and the analogue part. The digital part can be simulated easily, but the continuous part needs first to be converted to a numerical form.

The continuous part can be represented by several blocks which may be one of the following types:

(1) *Integral*: $\frac{K}{s}$

(2) *Proportion and integral*: $\frac{K_1 s + K_2}{s}$

(3) *Inertia*: $\frac{K}{Ts + 1}$

(4) *First order lead or lag*: $K\frac{T_1 s + 1}{T_2 s + 1}$

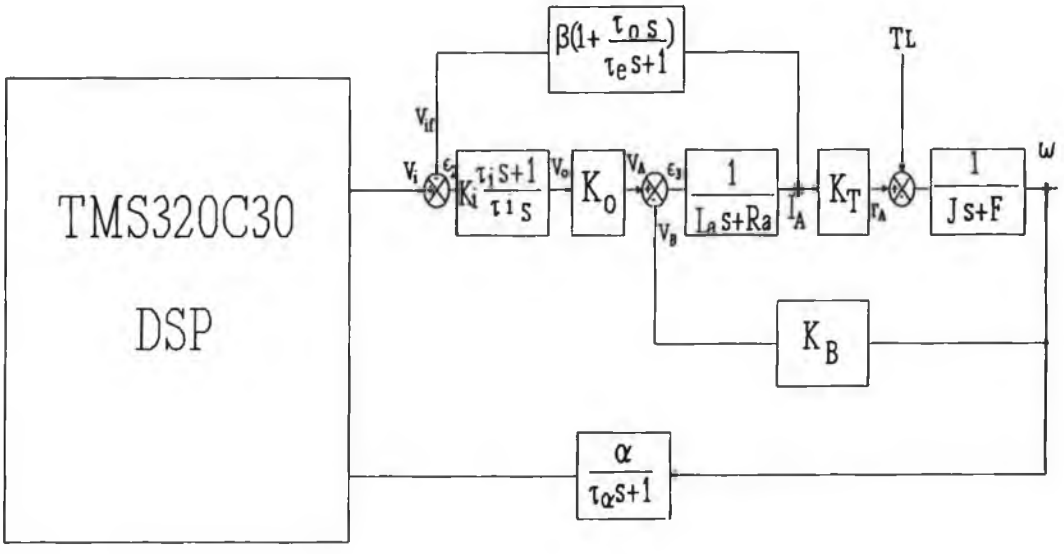


Figure 4-1 The block diagram of brushless servo system

These types can be expressed by a typical transform function block loop as follow:

$$\frac{Y_i}{U_i} = \frac{C_i + D_i s}{A_i + B_i s} \qquad (4.1.1)$$

where U_i is the input variable,

Y_i is the output variable,

A_i, B_i, C_i, D_i are the coefficients of the transform function.

By changing the values of A_i, B_i, C_i, or D_i, any of the above functions can be expressed by the typical function in equation (4.1.1 Therefore, all the continuous parts of brushless servo system can be simulated using a block-oriented program which solves each block of the system separately without simplifying the whole system. The numerical solution of a typical block is described below in the section 4.2. Using the numerical solution of first order differential equations , a real time simulation can be realized. Section 4.3 introduces a simulation for the brushless servo system and section 4.4 shows the simulation results.

.2 The solution of tical block

Block (4.1.1 can be expressed as:

$$\frac{Y(s)}{U(s)} = \frac{1}{A + Bs}(C + Ds) = \frac{Z(s)}{U(s)} \times \frac{Y(s)}{Z(s)} \qquad (4.2.1)$$

where Z(s is an internal variable.

Equation (4.2.1 can be expressed separately by follows:

$$\frac{Z(s)}{U(s)} = \frac{1}{A + Bs} \qquad (4.2.2)$$

$$\frac{Y(s)}{Z(s)} = C + Ds \qquad (4.2.3)$$

Equation (4.2.2 and (4.2.3 can be expressed in the time domain as:

$$AZ(t) + B\frac{dZ(t)}{dt} = U(t) \qquad (4.2.4)$$

$$CZ(t) + D\frac{dZ(t)}{dt} = Y(t) \qquad (4.2.5)$$

In order to obtain the time-domain solution, we convert equation (4.2.4 into:

$$\frac{dZ(t)}{dt} = -\frac{A}{B}Z(t) + \frac{U(t)}{B} \qquad (4.2.6)$$

It should be noted that the denominator B can not be zero, this means the typical block (4.1.1 must always include an integral term. Substituting equation (4.2.6 into (4.2.5 we have:

$$Y(t) = (C - D\frac{A}{B})\ Z(t) + \frac{D}{B}\ U(t) \qquad (4.2.7)$$

Z(t can be obtained by solving the differential equation (4.2.6 The first order differential numerical solution is described in the following section.

4.2.1 Numerical integration method for the differential euation

There are a number of numerical integration techniques which can be used to solve differential equations, such as the fourth-order Runge-Kutta method with fixed interval, Simpson's rule integration, trapezoidal integration, the Adams-second order method, rectangular integration, etc. When specifying an integration method, particular attention must be paid to obtaining sufficient accuracy without using excessive computing time. For the simulation of the brushless servo system the fourth-order Runge-Kutta method is good enough. The reasons are as follows [4-1] pp-87:

(1 Small truncation error of the order of h^5

(2 Time-domain solution available

Although a simulation program using the fourth Runge-kutta method takes more time to run than other methods, it is still feasible on a PC IBM AT 386. For example, a two-second simulation program of the servo system takes about 10 minutes, executing on an AT 386. For the purpose of examining the servo system dynamic response, this is fast enough. Therefore, the fourth-order Runge-Kutta method was chosen.

Detailed information about the principle, accuracy, and stability of the fourth-order Runge-Kutta method can be obtained in reference [4-1], [4-2]. The following is the formula for the fourth-order Rugne-Kutta method.

$$Z(i+1) = Z(i) + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4) + h^5 \qquad (4.2.9)$$

where h is the step length, h^5 is error, and

$$K_1 = MZ_i + NU_i \qquad (4.2.10)$$

$$K_2 = M(Z_i + \frac{1}{2}hk_1) + NU_i \qquad (4.2.11)$$

$$K_3 = M(Z_i + \frac{1}{2}hK_2) + NU_i \qquad (4.2.12)$$

$$K_4 = M(Z_i + hK_3) + NU_i \qquad (4.2.13)$$

.3 siulation rogra

In general, for a complex single input and single output system, the mathmetical model needs to be simplified into a high order differential equation:

$$\frac{d^n}{dt^n}y(t) + a_1\frac{d^{n-1}}{dt^{n-1}} + \cdots\cdots + a_{n-1}\frac{d}{dt}y(t) + a_n y(t) = c_1\frac{d^{n-1}}{dt^{n-1}}u(t) + c_2\frac{d^{n-2}}{dt^{n-2}}u(t) + \cdots\cdots + c_{n-1}\frac{d}{dt}u(t) + c_n u(t) \qquad (4.3.1)$$

where y(t is the output of system

u(t is the input of system

This simplification often ignores some functions in the system, which effects the accuracy of the simulation. Here we introduce a block-oriented transfer-function program which employs the typical loop and proportional gain to complete the system block diagram. Therefore, we do not need to simplify the total transfer function of the system.

4.3.1 Simulation of the continuous part of the brushless servo system

The brushless servo system, shown in figure 4-1, has several blocks each of which can be represented by the typical block or a gain. For example, the current loop is a

proportional and integral loop which is expressed by $K_i\frac{T_i s+1}{T_i s}$ and the

parameters of the typical block for this is $A_2=0$, $B_2=T_i$, $C_2=K_i$, $D_2=K_iT_i$. The proportion loop of the inverter can be expressed as the gain K_0.

Obviously, each block of the system is a transfer function which can be expressed by changing the parameters of the typical block. The program is written according to the system block diagram as shown in figure 3-5. The output of the previous loop is the input of the next loop. At the summing junction, the difference of outputs of two loops becomes the input to the next loop. The program for the continuous control part of the brushless dc servo system is presented in Appendix 1. The Runge-Kutta step h of the continuous-domain parts must be selected short enough to minimize the delay. Normally it should be more than a hundred times smaller than the sampling interval of the digital controller.

4.3.2 The simulation of the digital controller

The digital controller designs have been described in the previous chapter. For design I, we utilize equation (3.3.37 where the sampling period $T=100\ \mu$sec, $K_P=9.39$, and $K_I=0.99335$. The simulation program is presented in Appendix 2 in which an input command of 5 (volt represents a motor speed of 5000 rpm.

In design II, we use equation (3.4.1 which can be rewritten as three equations for the simulation.

$$\begin{aligned} e(kT) &= r(kT) - c(kT) \\ u_1(kT) &= K_I e(kT) + u_1[(K-1)] \\ u(kT) &= -K_P c(kT) - K_D\{B(kT) - b[(k-1)T]\} + u_1(kT) \end{aligned} \qquad (4.3.2)$$

where e(kT and u_1(kT represent the sample-data of the error signal and the output of the cascade integral compensator, respectively. The program for design II, presented in Appendix 3, has controller parameters which are selected as: sampling period T =

0.5 msec, K_P =18, K_I =0.09, K_D =55 and an input of 5 volt represents a motor speed of 5000 rpm.

. iulation results

The dynamical response of the simulation of DESIGN I is shown in figure 4-2, the settling time is about 0.06 sec and the overshoot of the response is about 10% . This closely matches the design requirements which are: the settling time 0.06 sec and overshoot 8%. When we change the parameters, we can see the sensitivity of the system dynamical responses as shown in figure 4-3. Reducing the sample time to 70 μs increases the overshoot of the dynamic response of the servo system as shown in series 1. By changing the K_I to 0.83, K_P to 7.5, the response curve is as shown in series 3 in which the overshoot is reduced and the rise time is increased.

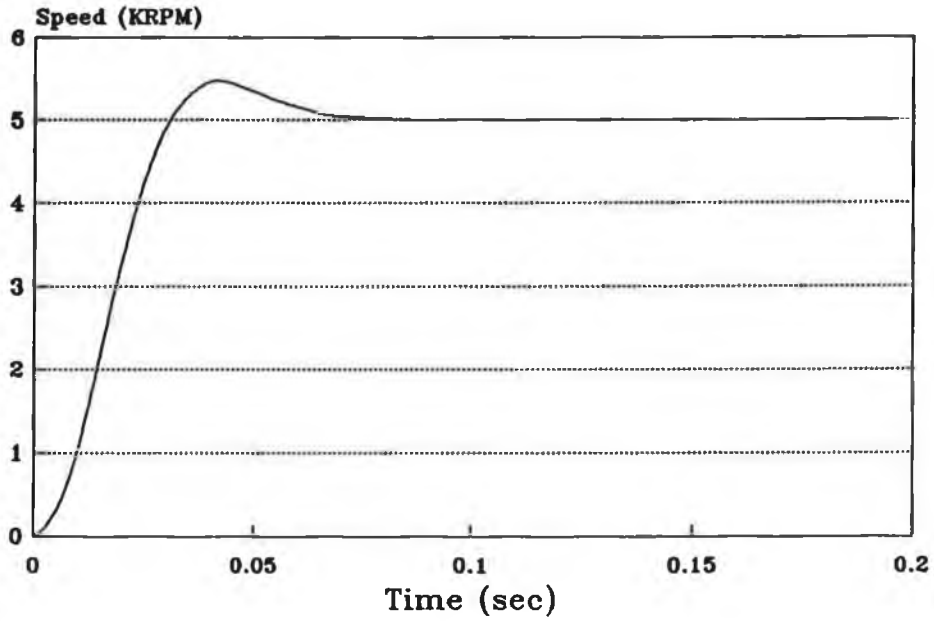


Figure 4-2 The simulated step response of Design I

For design II, the step response of the simulation is shown in figure 4-4, the rise time is about 0.01 sec, the settling time is about 0.026 sec, and the time to peak is about 0.015 sec overshoot is 20%. This result is close to the design requirements which are:

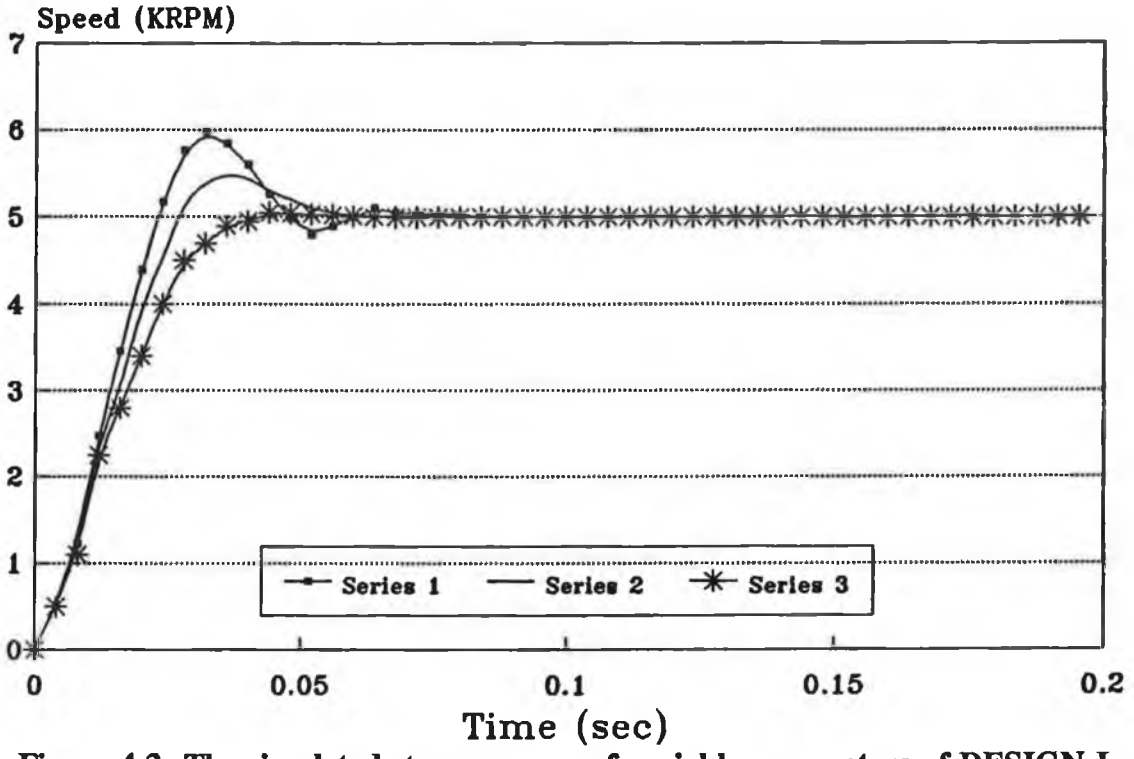


Figure 4-3 The simulated step response of variable parameters of DESIGN I

the overshoot 20% and the settling time 0.02 sec. Figure 4-5 shows different step responses of the speed of the system, in which, we obtained series 1 with K_I= 0.05, series 2 with K_I=0.085, series 3 with K_I =0.12, and series 4 with K_I=0.15. We see when K_I is increased, the dynamical response is faster but the overshoot is increased and the system becomes unstable.

4.5 Conclusion

The variable speed brushless servo system is simulated using a block-oriented simulation scheme. The block diagram is implemented by employing a typical block to represent each component of the brushless motor drive. Using this scheme, the designs of the speed controller for the brushless DC motor drive were examined and the dynamic behaviours of the system was demonstrated accurately. The simulation results proved that the controller designs are very close to the design requirements.

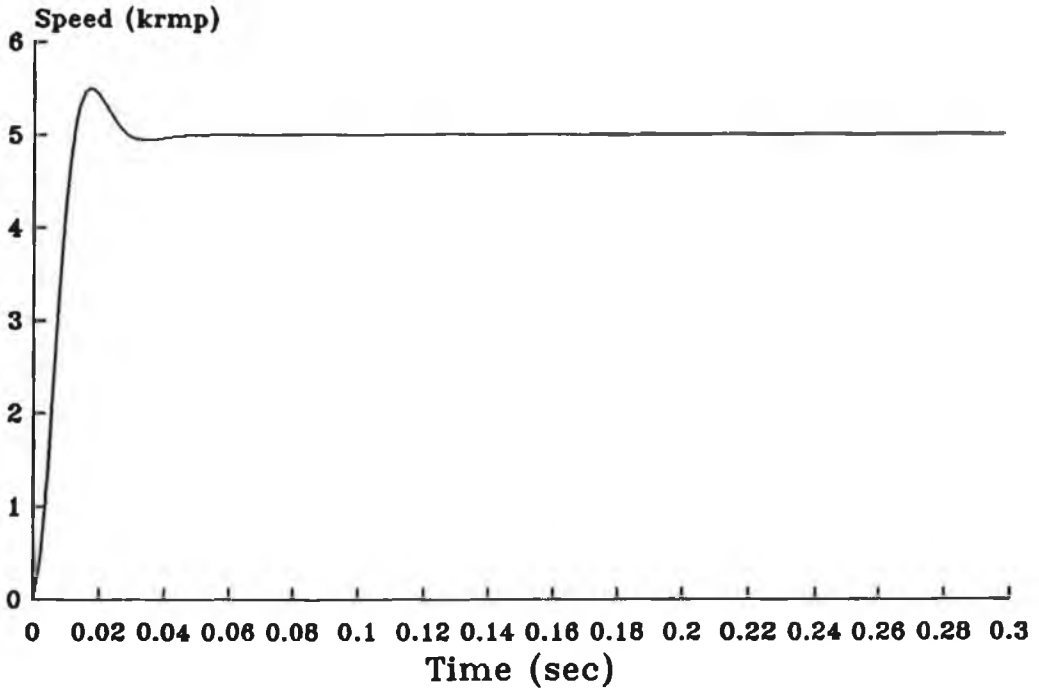


Figure 4-4 The simulated step response of DESIGN II

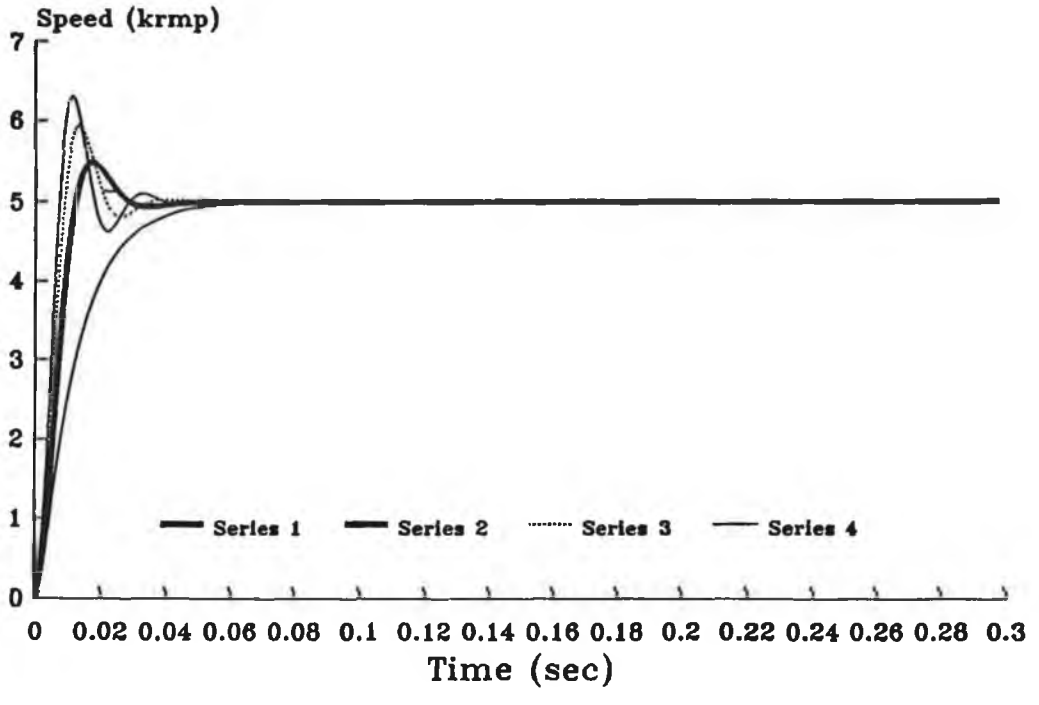


Figure 4-5 The simulated step responses of variable Design II parameters

Chapter 5

BRUSHLESS MOTOR THERMAL PROTECTION

5.1 Introduction

Permanent magnetic brushless motors have a high performance which places a high demand on motor protection devices. It is recommended to run these motors close to their thermal limits in order to achieve high specific outputs. Therefore it is necessary to provided a high degree of protection to avoid excessive downtime and unnecessary shutdown.

Motors may fail due to mechanical stresses or environmental factors, however the majority of failures are still a result of supply and loading faults which cause excessive insulation temperatures. Many of these occur even when protective devices, such as I^2t thermal protection or thermal overload relays, are fitted [5-11]. This is because the device may not have the same thermal characteristics as motors which are being protected.

This chapter introduces a thermal protection control scheme for a PM brushless servo motor. This scheme uses a motor thermal model to predict the motor winding temperature and then sets the current limit to maintain the temperature below the motor insulation limit. The thermal model can be established using a single component (the motor itself), two components (the motor stator winding and stator iron) or four components (the stator winding, stator iron, rotor and remaining masses). The power losses, calculated from the motor load current, the speed and a constant PWM loss, are used to estimate the motor temperature. The motor protection control ensures that the

motor stator winding temperature always remains below the insulation class limit. This provides a maximum utilization of the motor output power during transient loads.

The brushless motor power losses are described in section 5.2, they can be subdivided into copper losses, core losses, windage losses, and mechanical losses. Then section 5.3 introduces a single-component thermal simulation which is established using an electrical R-C analogue and is used to predict the temperature of the motor. Section 5.4 describes a two-component simulation. Section 5.6 describes the thermal protection control procedure. Finally, section 5.7 introduces a simulation of this procedure for an AEG BHT 2214M brushless motor. Using this scheme, the motor thermal capacity can be utilized more efficiently and can drive loads exceeding its continuous rating for short periods of time.

5.2 Power loss

The heat generated within the motor is a result of electrical and internal power losses which include:

1 Copper loss :

This is the Joule heat, I^2R_a in the armature resistance which is expressed as:

$$P_c = I^2 \times R_a \qquad (5.2.1)$$

where I is the equivalent DC armature current.

R_a is the equivalent DC resistance.

2 Windage loss:

The work required for the rotor to move through the air. This is converted to heat inside the rotor and in the air.

3 Mechanical loss:

This is caused by friction.

4 Core loss (or iron loss):

This has two components, one is the hysteresis loss, the other is the eddy-current loss.

5.2.1 Power flow of a brushless motor

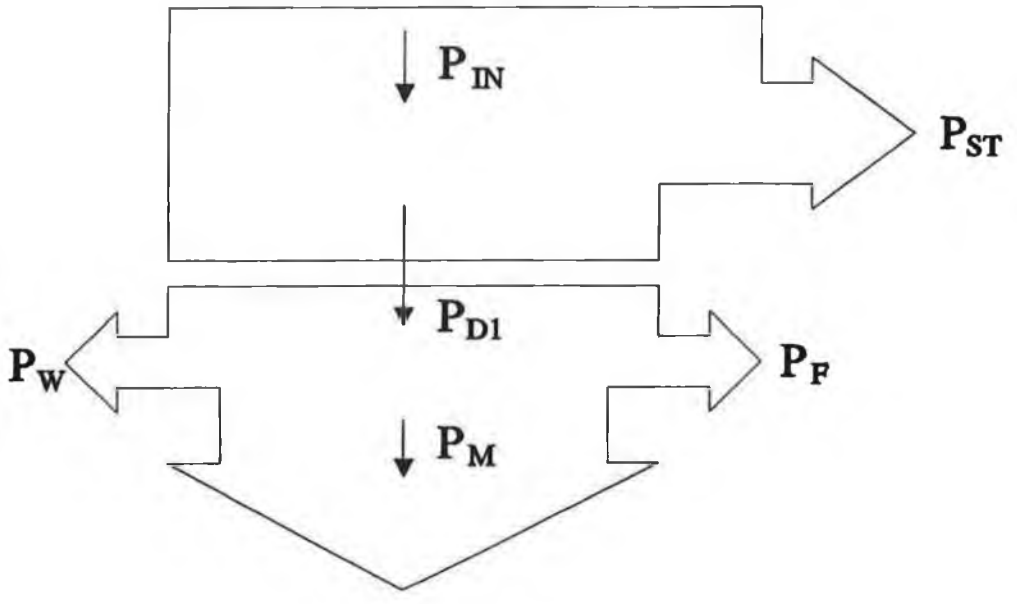


Figure. 5-1 Flow of power of brushless motor

A power flow diagram, illustrated in figure 5-1, shows the power loss process in the motor. The electrical input power P_{IN} enters the motor terminals. Copper losses, P_c, occur in the motor windings and the loss, P_f, in the stator iron. The total loss, P_{ST}, leads to a temperature rise in the stator. The remaining power P_{D1} is transmitted via the electromagnetic field and the air gap, and is uniformly divided throughout the rotor. The frictional loss P_{Fr} and windage loss P_W, which are summed as the rotor losses, P_{RT}, must be subtracted from P_{D1}. The remaining power is the effective power P_M which is given out as the mechanical power of the shaft. Where, $P_{ST} = P_f + P_c$; and $P_{RT} = P_{Fr} + P_W$.

For permanent magnet brushless motor protection, the most important element of the motor is the stator winding which is set into the stator iron. While the rotor is made of permanent magnetic material in which little heat is created. Therefore, the rotor losses

P_{RT} normally may be ignored.

5.2.2 Thermal power loss calculation

The power loss must be calculated in order to predict and adjust the motor temperature. The windage and mechanical losses can be considered to be proportional to the square of the speed. In the core loss, due to the rotating rotor flux, the hysteresis loss is proportional to the speed, whilst the eddy-current loss is almost proportional to the square of the speed. The speed is proportional to the back e.m.f, E. In the equivalent circuit diagram of the brushless DC motor, in figure 5-2, a resistance R_h is placed in the circuit in which the power loss is E^2/R_h. Hence, an approximate value of R_h is determined by including components of the iron, windage and mechanical losses.

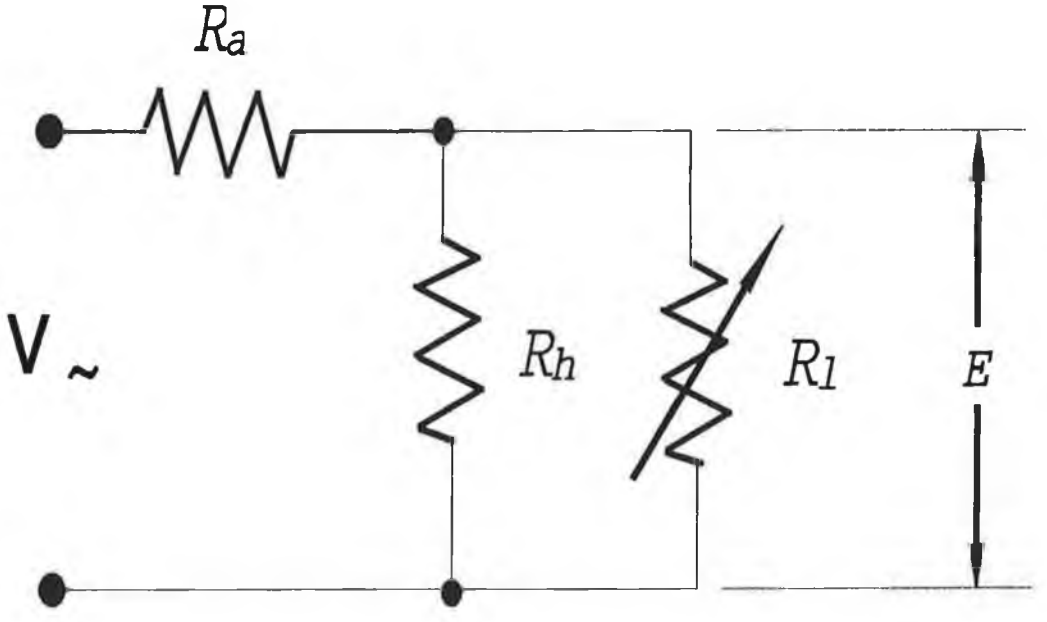


Figure 5-2 Motor equivalent circuit

In figure 5-2, R_a is the armature resistance. R_l is the equivalent load resistance.

The brushless motor losses can be segregated into three components which affect the motor windings:

1) Constant: PWM core loss.

2) Proportional to ω^2 : E^2/R_h including core loss, friction loss and windage loss.

3) Proportional to I^2R_a : copper losses.

The losses (1) and (2), denoted P_f, cause heat in the motor stator iron. The rotor loss P_{RT} is small and can be included in P_f. The loss (3) is P_c which creates heat in stator windings. P_f can be expressed as:

$$P_f = K_0 + \frac{E^2}{R_h} \qquad (5.2.2)$$

and P_c is:

$$P_c = I^2 \times R_a \qquad (5.2.3)$$

The total thermal power loss of a brushless motor can be calculated from:

$$P_{loss} = P_f + P_c = K_0 + \frac{E^2}{R_h} + I_a^2 \times R_a \qquad (5.2.4)$$

where P_{loss} is the total motor power loss.
K_0 is a constant representing the PWM loss.
E is the back e.m.f.
R_h is the equivalent resistance of loss (2).
I_a is the equivalent DC current.
R_a is the equivalent DC armature resistance.

5.2.3 Maximum allowable power loss

In order to set the limit for the highest allowable motor temperature, we need to calculate the maximum allowable power loss. The speed-torque limits of a permanent magnet brushless motor are shown in figure 5-3.

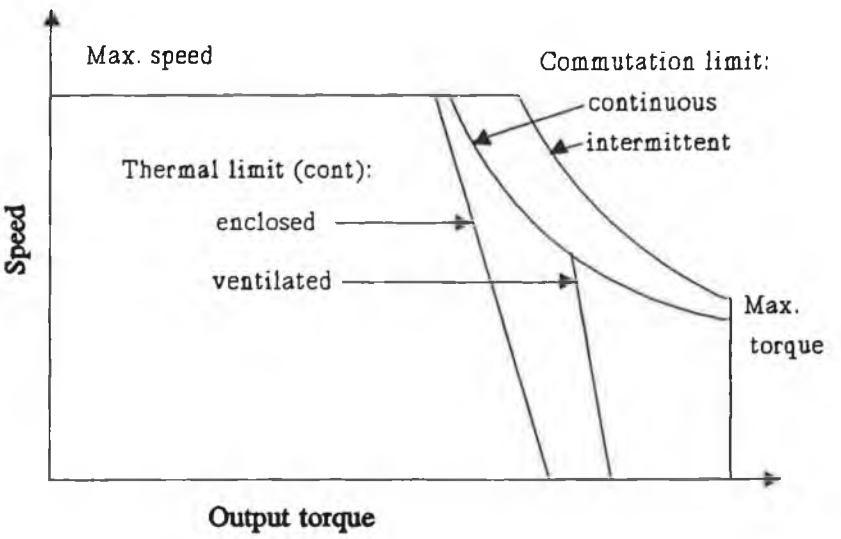


Figure 5-3 Permanent-magnet brushless motor: operating limits

Among the limits, the maximum continuous torque depends on the insulation class limit, and at which point the allowable power loss is the maximum. In the BHT 2214M performance curve, shown in figure 5-4, the motor stall torque is the maximum continuous torque and at which point the continuous thermal power loss is also the maximum. So, the maximum thermal power can be obtained from equation (5.2.5) by calculating the power loss for the stall torque using equation (5.2.4) in which E is zero (zero-speed).

$$P_M = I_M^2 \times R_a + P_{PWM} \qquad (5.2.5)$$

where P_M is the maximum thermal power loss.

I_M is the current of maximum continuous stall load.

R_a is the equivalent DC armature resistance of motor.

P_{PWM} is power loss of high frequency PWM which can be obtained by method of appendix 2.

P_M is the maximum allowable thermal power which raises the motor temperature to the insulation limit. Good thermal protection control requires that the motor temperature

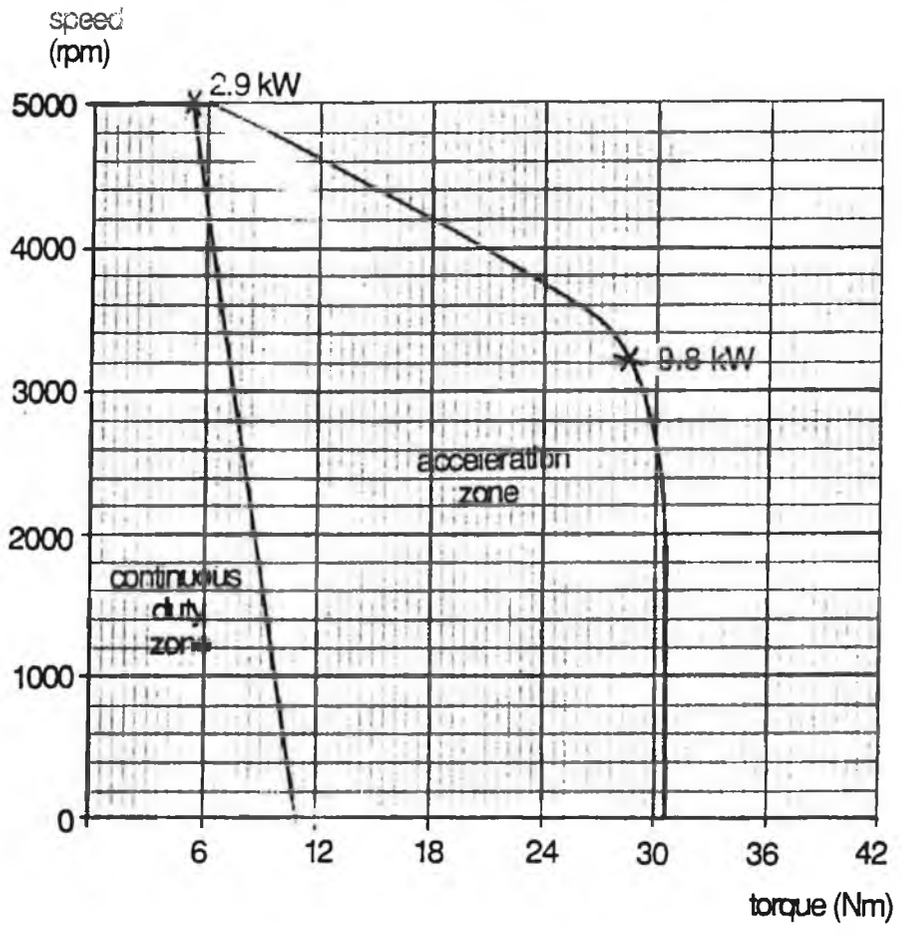


Figure 5-4 BHT 2214M Brushless Motor Speed-Torque Characteristic

always stays below the insulation class limit, and motor outputs the maximum allowable power. This demands that the motor maximum power loss is calculated as close as possible to the actual motor power loss.

For a practical application, the maximum thermal power can be obtained by using equation (5.2.5). For example, we use the above method to compute the maximum power loss of the BHT 2214M brushless motor which has a maximum stall current of 21.1 A. The loss P_M is:

$$P_M = P_{pwm} + I_a^2 \times R_a = 2 + 133.56 = 135.56\ W \qquad (5.2.6)$$

where the PWM power loss is 2W (see appendix 2 for the measurement of the BHT 2214M brushless motor PWM power loss) and the equivalent DC armature resistance is 0.3Ω.

5.2.4 Calculating the equivalent resistance R_h

In order to calculate the power loss, we need to know the equivalent resistance R_h shown

in figure 5-2. This can be calculated from the motor continuous duty speed-torque curve. A maximum power loss point can also be found on this curve at the motor maximum speed. The power loss at this point is approximately equal to that at the maximum stall torque. Hence, R_h can be obtained:

$$R_h = \frac{E^2}{P_M - I_a^2 R_a - P_{PMW}} \qquad (5.2.7)$$

where P_M is the maximum power loss.

I_a is the current at the point of the maximum output power.

E is the back e.m.f at the point of the maximum output power, which can be derived from speed/K_b (back E.M.F constant).

R_a is the equivalent DC armature resistance of motor.

For the AEG BHT 2214M brushless motor, at the point of the maximum power output, the speed is 5000 rpm and the torque is 5.2 Nm. The current, therefore, is torque/K_T = 5.2/0.536=9.7 A. The back e.m.f, E=speed*K_b=5000*0.056=280 V. According to equation (5.2.7), R_h is:

$$R_h = \frac{E^2}{P_{th} - P_{pwm} - I^2 R_a} = \frac{280^2}{135.56-2-9.7^2\times 0.3} \approx 748\Omega$$

5.2.5 Drawing the maximum continuous Speed-Torque characteristic curve

In order to continuously obtain the maximum allowable output power during variable speeds and transient loads, we need to draw the maximum continuous speed-torque characteristic curve as the output power limit. P_M, the maximum continuous thermal loss, the power loss in equation (5.2.4), has the form of the parametric equation for an ellipse. Using this equation, the continuous speed-torque curve is drawn as an elliptic shaped curve to set motor maximum allowable current limits (Normally the continuous speed-torque characteristic curve is drawn by an oblique line linking both the maximum power output point and the stall point). This curve can be drawn using the following steps:

1. Calculate the maximum allowable power loss.
2. Draw the curve according to the power loss equation (5.2.4), in which the power loss P_{loss} equals P_M.

For the BHT 2214M motor, as an example, they are given below:

1. The flowchart of the drawing characteristic curve program is shown in figure 5-5.

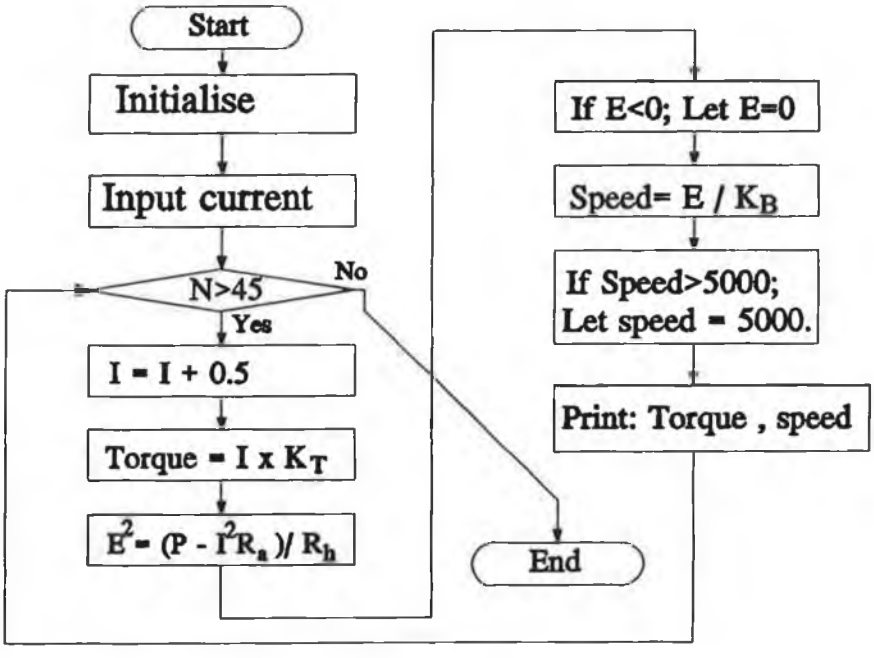


Figure 5-5 Flowchart of the extended characteristic curve

2. The maximum power loss obtained in section 5.2.3 is 135.56W.
3. The program for drawing this expanded curve is given in Appendix 1.
4. The characteristic curve graph of BHT 2214M is shown in figure 5-6.

In figure 5-6, curve 1 is the previous BHT 2214M motor continuous operation curve and curve 2 is the expanded characteristic curve. Obviously, curve 2 expands the motor continuous duty zone because of the more accurate P_M calculation.

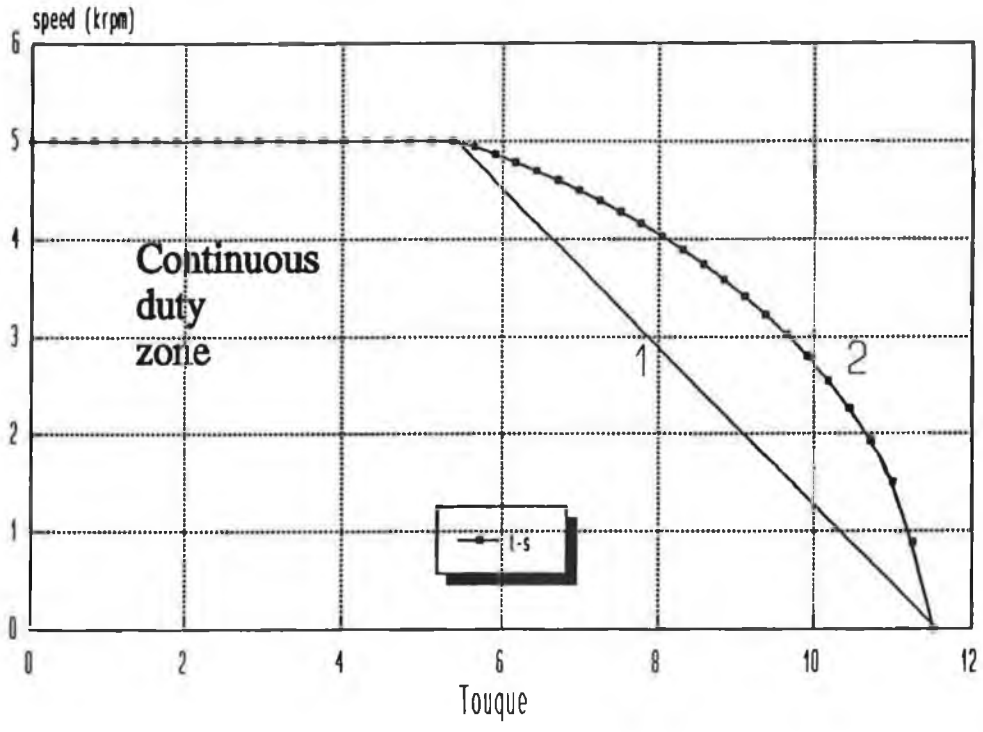


Figure 5-6 BHT 2214M continuous characteristic curve

5.3 Single-component thermal model

The thermal model of a brushless DC motor can be simulated by an electric circuit. In general, the flow of thermal energy takes place from a region of high temperature to a region of lower temperature, which is similar to the electric circuit, in which current flows from a point of high potential to a point of lower potential. Heat transfer is therefore analogous to current flow. This is shown in figure. 5-7.

5.3.1 Thermal resistance

Thermal resistance, denoted by R_θ, is a measure of the temperature difference per watt of heat flow (°C/W), just as electrical resistance is a measure of the voltage difference per ampere of current flow in volts/ampere or ohms. From figure 5-7 (b), by analogy with Ohm's law, the steady-state temperature difference, when a constant thermal power

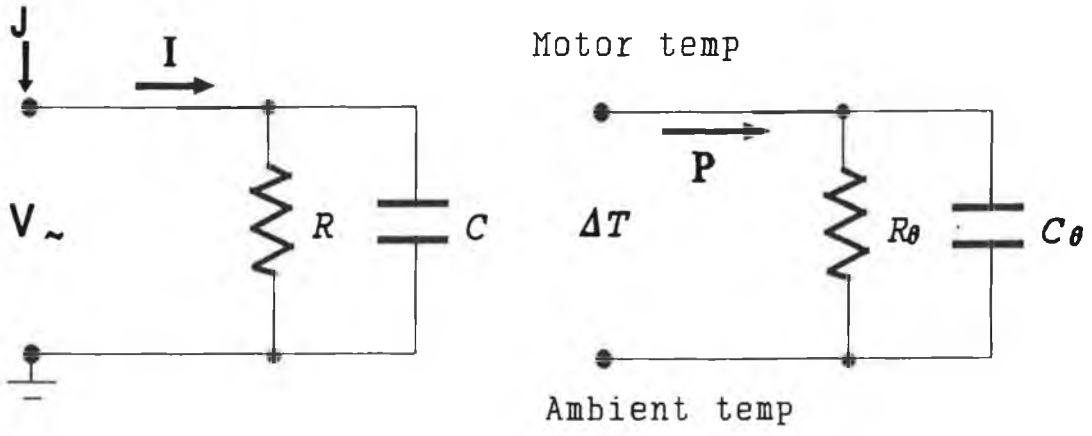


Figure 5-7 Electric circuit analogy of the thermal circuit

of P watts flows through a thermal resistance of R_θ, is given by:

$$T_1 - T_2 = P \times R_\theta \qquad (5.3.1)$$

Also

$$R_\theta = \frac{T_M - T_a}{P} \qquad (5.3.2)$$

Where, T_M is the maximum allowable motor winding temperature (°C).

T_a is the ambient temperature (°C).

P is the constant power loss (W).

R_θ is the thermal resistance (°C/W).

The thermal resistance can be calculated using the insulation temperature and the power loss at point of maximum continuous current. For example, the maximum allowable power loss for the BHT 2214M brushless motor is obtained from equation (5.2.6) is 135.56W. Assuming the maximum ambient temperature is 40°C, and the insulation temperature is 180°C, R_θ can be obtained from equation (5.3.2):

$$R_\theta = \frac{180 - 40}{135.56} \doteq 1.03\ ^\circ C/W$$

where we assume the thermal resistance R_θ is a constant.

5.3.2 Thermal capacitance

For transient conditions, the motor thermal storage capacity should be taken into consideration in order to calculate the motor winding temperature. Thermal capacity is the heat energy stored per degree of the temperature rise. When the motor temperature rises rapidly on overcurrents, the thermal capacity can be utilized to give a significant short term overcurrent capability and thereby avoid an unduly conservative design.

Thermal capacity is analogous to capacitance in the electrical circuit as shown in the figure 5-7. The thermal capacitance C_θ can be determined by the thermal time constant T_{th}.

$$R_\theta \times C_\theta = T_{th}\ ; \quad Also \quad C_\theta = \frac{T_{th}}{R_\theta} \qquad (5.3.3)$$

5.3.3 Thermal equation

The motor transient thermal behaviour can be modeled approximately by an R-C network, shown in figure 5-7 (b). Assume a constant heating power P, is suddenly applied to the motor at zero time. For this step input in heating power, there is a corresponding step input of current in the electrical analog of figure 5-7 (a). This current must charge up the network capacitance before a voltage can appear at input terminal J. Consequently, there is a delay in the build up of voltage at the input terminals. Similarly, there is a time lag in the decay of winding temperature when the input P is removed. For an electrical RC network in figure 5-7 (a), the voltage equation is:

$$V = I\,R - R\,C\,\frac{dU}{dt}$$

A corresponding thermal equation can be obtained as:

$$T_m - T_a = PR_\theta - R_\theta\,C_\theta\,\frac{d\Delta T}{dt} \qquad (5.3.4)$$

where T_m is the motor temperature.

T_a is the ambient temperature.

P is the thermal power.

ΔT is the temperature difference.

t is the time variable.

C_θ is the thermal capacitance.

R_θ is the thermal resistance.

For real time motor temperature prediction, equation (5.3.4) must be transformed to a digital form. The Laplace transform of equation (5.3.4) is:

$$T(s) = P(s)\,R_\theta - T_{th}\,T(s)\,s \qquad (5.3.5)$$

where T(S) is the difference of T_m - T_a.

T_{th} is the thermal time constant which is equal to $R_\theta C_\theta$.

Also,

$$\frac{T(s)}{P(s)} = \frac{R_\theta}{T_{th}\,s + 1} \qquad (5.3.6)$$

The z transform of equation (5.3.6) can be expressed as:

$$\frac{T(k)}{P(k)} = \frac{b\,z^{-1}}{1 - a\,z^{-1}} \qquad (5.3.7)$$

$$\begin{cases} a = e^{\frac{-k}{T_{th}}} \\ b = R_\theta\,(1 - e^{\frac{-k}{T_{th}}}) \end{cases}$$

From equation (5.3.7), we can obtain:

$$T(k) = e^{\frac{-k}{T_{th}}} T(k-1) + R_\theta \left(1 - e^{\frac{-k}{T_{th}}}\right) P(k-1) \qquad (5.3.8)$$

where k is the sample time. This can be used for the TMS320C30 implementation.

5.4 Two-component thermal model

5.4.1 Establishing thermal model

For a more accurate thermal model, two main elements, copper and iron, can be considered as shown in figure 5-8. In this model, the point **W** represents the motor winding, its temperature is denoted by T_w, and the point **I** represents the iron of the motor stator, its temperature is denoted by T_i. The capacitor C_1 represents the thermal capacity of the windings. The capacitor C_2 represents the thermal capacity of all the iron and other masses of the motor. The transfer of heat to the air gap between stator and rotor is represented by a resistance R_1. A resistance R_0 represents the thermal transfer across the insulation to the iron. The transfer of heat to the surroundings is represented by a resistance R_2. The copper losses P_c, which are calculated from the current consumption of stator windings (see section 5.2.1), are fed into the capacitor C_1. The iron losses, P_f (see section 5.2.1), are fed into the capacitor C_2, they are a major part of the total heating within the motor.

5.4.2 Winding temperature calculation

Let us use the electrical circuit nodal method again to predict the motor winding temperature. Applying Kirchhoff's current law at the points **W** and **I**, the thermal equations are:

$$P_f = \frac{T_i}{R_2} + C_2 \times \frac{dP_f}{dt} - P_0 \qquad (5.4.2.3)$$

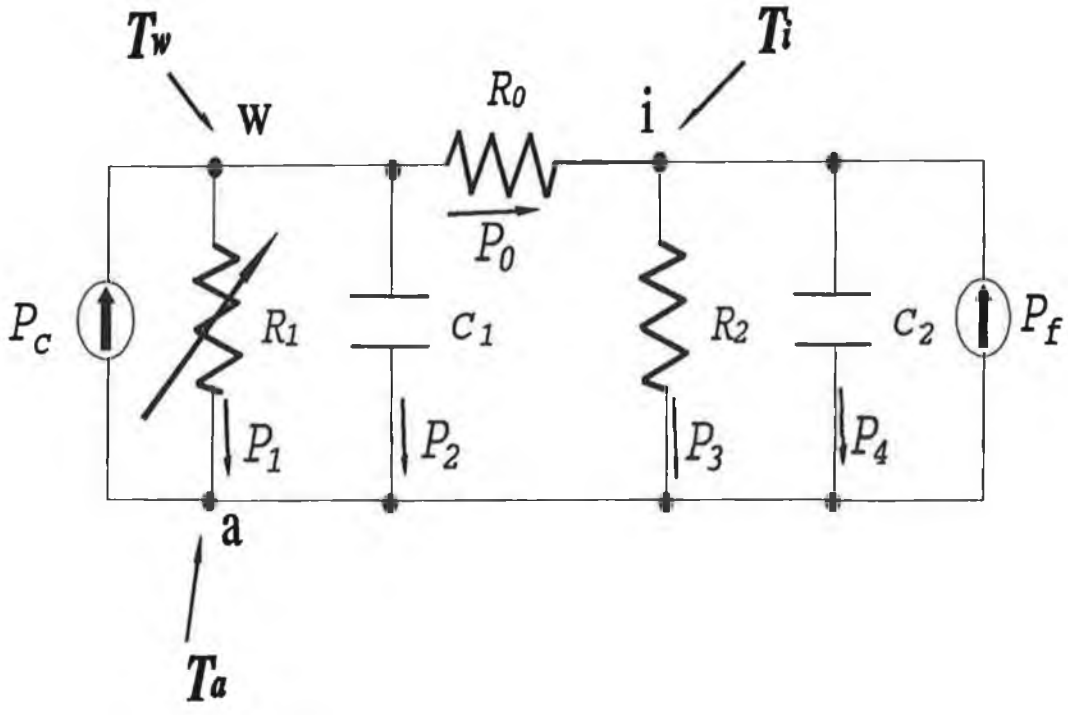


Figure 5-8 Thermal equivalent circuit

$$P_c = \frac{T_w}{R_1} + C_1 \times \frac{dP_c}{dt} + P_0 \qquad (5.4.2.1)$$

The difference in temperature between the winding and the iron is:

$$T_w - T_i = R_0 \times P_0 \qquad (5.4.2.4)$$

Using the Laplace transform, a set of simultaneous equations can be obtained:

$$\begin{cases} R_1 P_c(s) = T_w(s) + C_1 R_1 s T_w(s) + R_1 P_0(s) \\ R_2 P_f(s) = T_i(s) + C_2 R_2 s T_i(s) - R_0 P_0(s) \\ T_w(s) - T_i(s) = R_0 P_0(s) \end{cases} \qquad (5.4.2.5)$$

The solution to the above is:

$$T_w = \frac{R_1 P_c(s) + \frac{R_1 R_2}{R_0} P_f(s)}{1 + R_1 C_1 s + \frac{R_1}{R_0} + \frac{\frac{R_1 R_2}{R_0^2}}{1 - \frac{R_2}{R_0} + R_2 C_2 s}} \qquad (5.4.2.6)$$

The motor winding temperature can be predicted using this equation.

This two component thermal equation can be used to produce a heating/cooling characteristic that can give a very close approximation to the actual heating/cooling characteristic of the motor. Therefore a better thermal protection scheme can be obtained using this model than using the single-component model.

5.5 Brushless motor thermal protection control procedure

Using brushless motor thermal protection control, the motor winding temperature is always kept below the insulation class limit, while maximizing the motor output power during the variation of speed and loads. The control procedure, shown in figure 5-10, is described by following two steps:

First, predict motor temperature.

1. Calculate motor power losses.
2. Predict the motor winding temperature using the thermal model.

Second, Compare the temperature calculated from the thermal model with the insulation class temperature. If the motor temperature is over the insulation limit, the controller begins to limit the motor current to keep the winding temperature at the insulation limit. The procedure is:

1. Calculate the motor maximum power loss using the method in section 5.2.3.
2. Maintain the motor output power along the maximum continuous characteristic curve. Calculate the motor current at the measured speed in the maximum thermal power loss curve. For the single-component thermal model, this current is:

$$I_a = \sqrt{\frac{P_M - \frac{E^2}{R_h} + P_{PWM}}{R_a}}$$

3. Output this current as an input to the current loop. This keeps the motor temperature below the allowable limit.

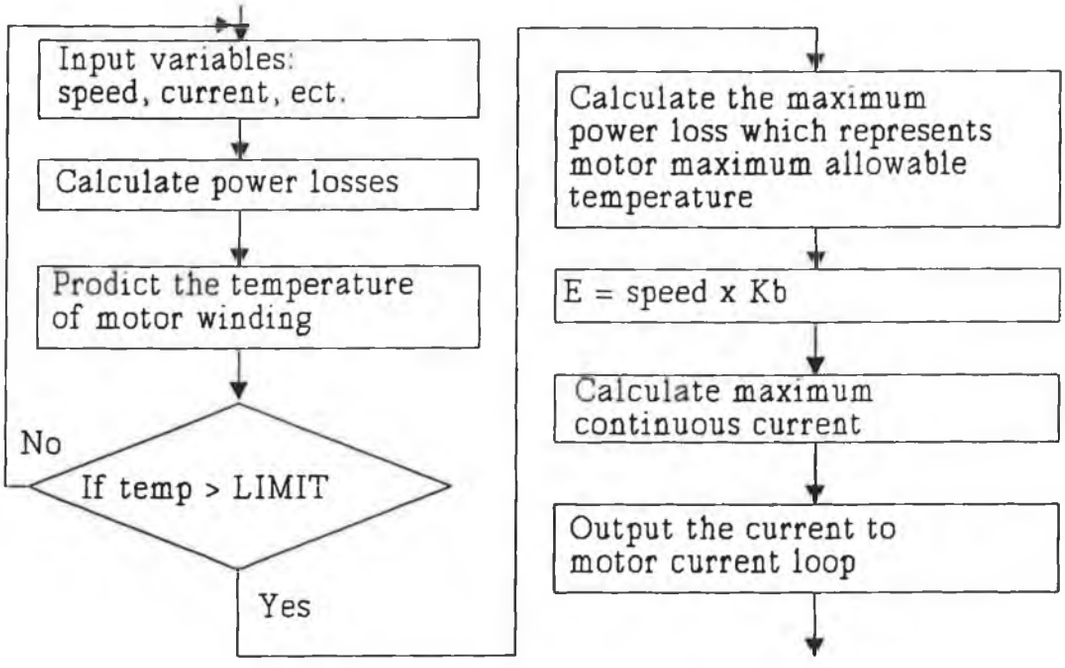


Figure 5-10 Flowchart of the motor thermal protection

5.6 Thermal protection simulation

This section introduces a simulation of the thermal protection control for the AEG BHT 2214M brushless motor. The motor power loss can be obtain from equation (5.2.4).

$$P_{lose} = K_0 + \frac{E^2}{R_h} + I_a^2 \times R_a \qquad (5.7.1)$$

Where P_{loss} is the total motor power loss. K_0 is a constant PWM loss of 2W (see appendix 2). The motor back E.M.F is $E = \omega \bullet K_B$. $R_h = 748\ \Omega$ (see section 5.2.4). I_a is the motor equivalent armature current. R_a is an equivalent DC armature resistance, $R_a = 0.3\ \Omega$.

The single-component thermal model is used to predict the motor winding temperature. It is expressed as:

$$T(k) = e^{\frac{-k}{T_{th}}} T(k-1) + R_\theta \left(1 - e^{\frac{-k}{T_{th}}}\right) P(k-1) \qquad (5.7.2)$$

$$= e^{-\frac{1}{45}} T(k-1) + 748\left(1-e^{-\frac{1}{45}}\right) P(k-1)$$

where T(k) is the predicted temperature and P(k) is the power loss P_{loss}.

The motor insulation limit is 140°C. Once the predicted temperature obtained by equation (5.7.2) compares equally or greater than the insulation limit, the protection control begins working. The procedure is as follows:

The maximum allowable power loss is 135.56W from which the maximum continuous speed-torque curve can be obtained (see section 5.2.5). The motor output power is limited along this curve. First, sample the motor speed from the tachogenerator, then calculate the maximum allowable current using equation (5.2.4):

$$I_a = \sqrt{\frac{P_M - \frac{E^2}{R_h} - P_{PWM}}{R_a}}$$

$$= \sqrt{\frac{135.56 - \frac{\omega^2}{748} - 2}{0.3}}$$

where P_{loss} is the maximum allowable power loss, and ω is the motor speed.

This current is fed to the current loop in the motor controller to limit the motor temperature. The flowchart of the simulation program of the thermal protection control is shown in figure 5-11. The simulation program is presented in appendix 3.

The thermal protection simulation results are shown in figure 5-16. Figure 5-12 shows the motor speed variation. Figure 5-13 shows the motor load variation. Figure 5-14 shows the power loss obtained from figures 5-12 and 5-13. Figure 5-15 shows the

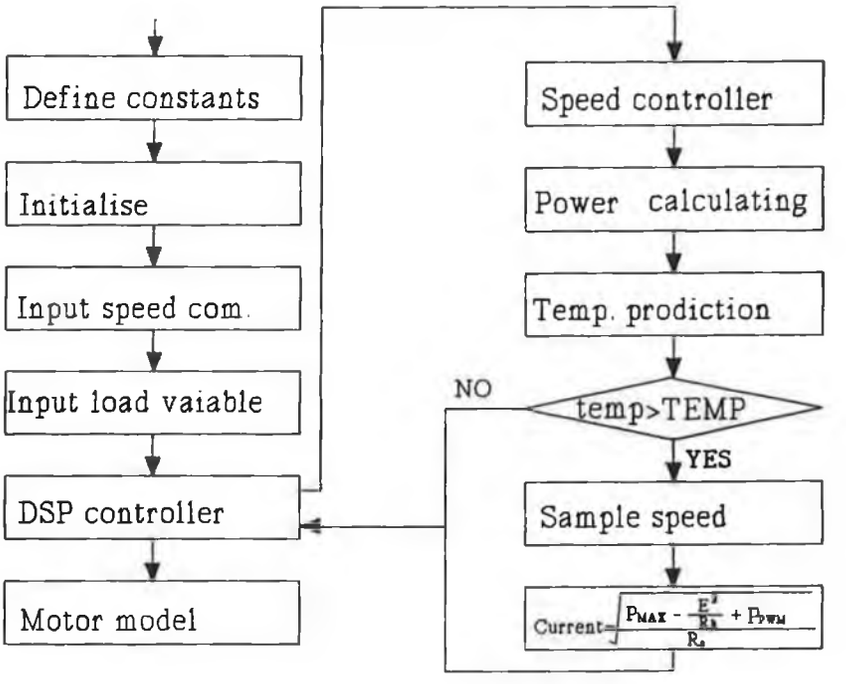


Figure 5-11 Flowchart of the thermal protection simulation of the BHT 2214M brushless motor

motor winding temperature without motor thermal protection control. From these figures, it can be seen that motor temperature follows the change of the power loss which is produced from the speed and load variation. Without thermal protection, the motor temperature will go beyond the insulation limit as shown in figure 5-15. Using this temperature prediction and current limiting controller, the motor temperature is always below the insulation temperature limit as shown in figure 5-16, and the motor is protected in a safe state and utilized to the maximum allowable output power.

5.7 Conclusion

Thermal protection of PM brushless motors is a key problem in motor servo drives used in robots and machine tools. In this chapter, a real-time thermal protection control scheme was presented for a PM brushless servo motor. Motor thermal models were established, and the thermal controller used one of them to predict the motor winding temperature. Once the predicted temperature reaches the winding insulation limit, the thermal controller maintains the motor operation within a maximum allowable speed-

torque region. This keeps the winding temperature below the insulation limit, while maximizing the motor power output. The simulation of the thermal protection control was described and the results showed this scheme could complete a high precision thermal protection control for a brushless DC motor.

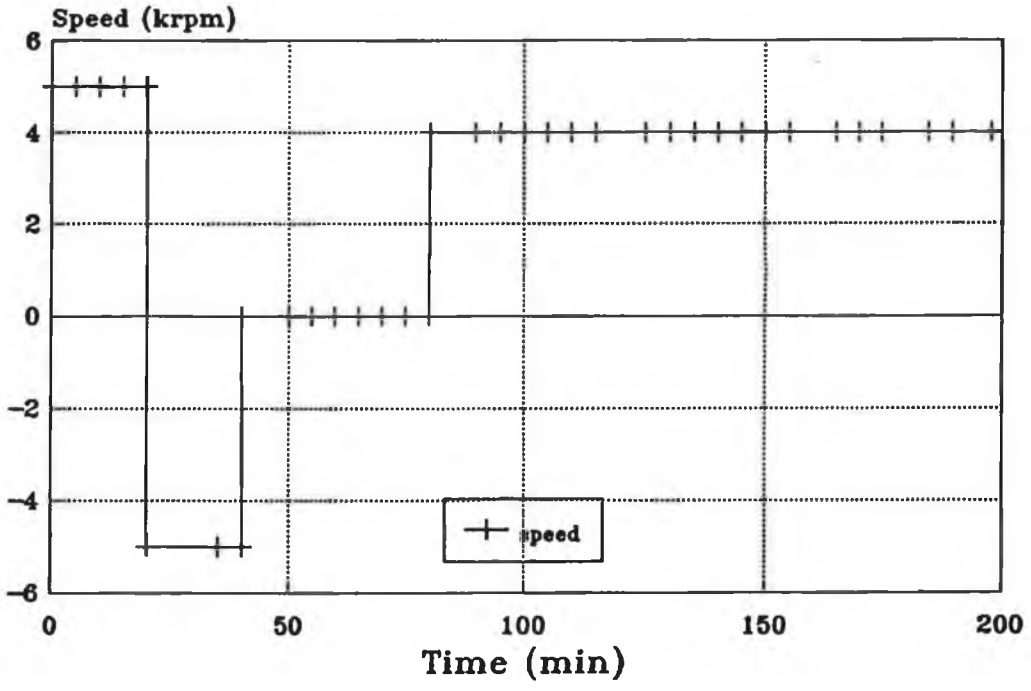


Figure 5-12 Variable Speeds of motor

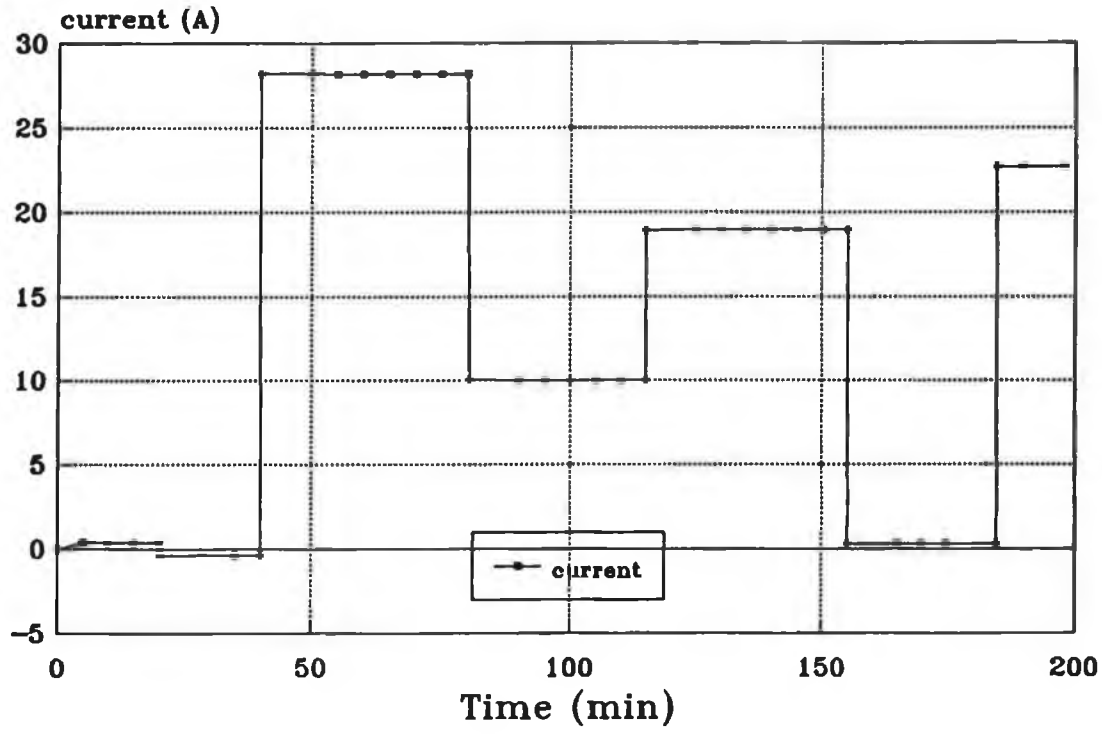


Figure 5-13 Variable loads

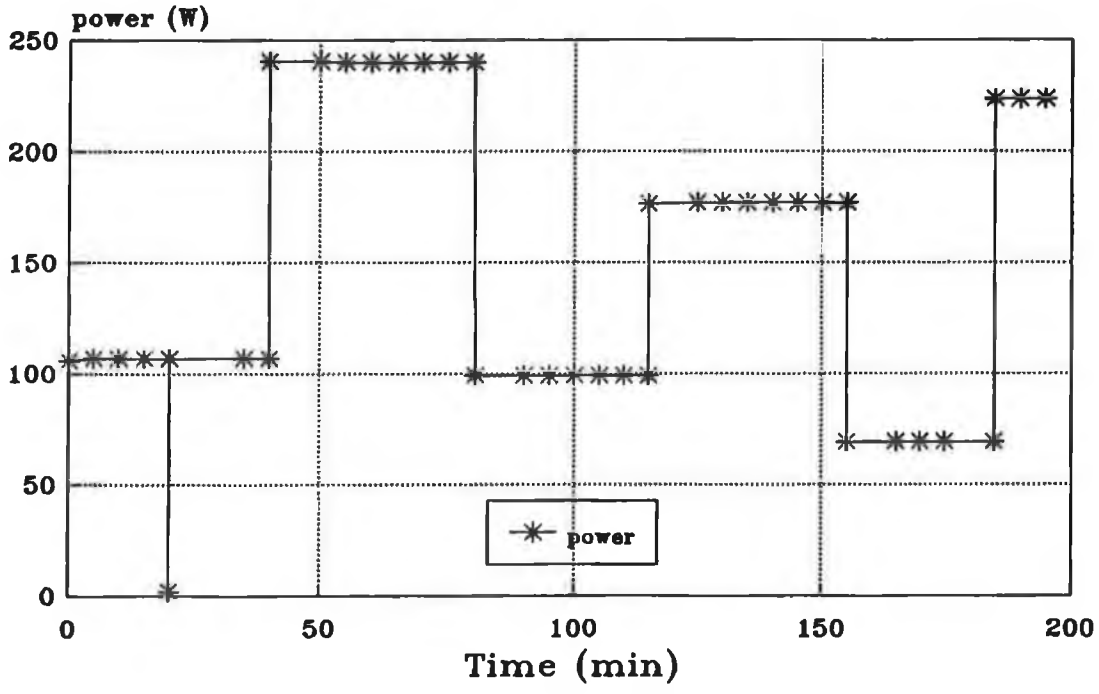


Figure 5-14 Motor thermal power loss

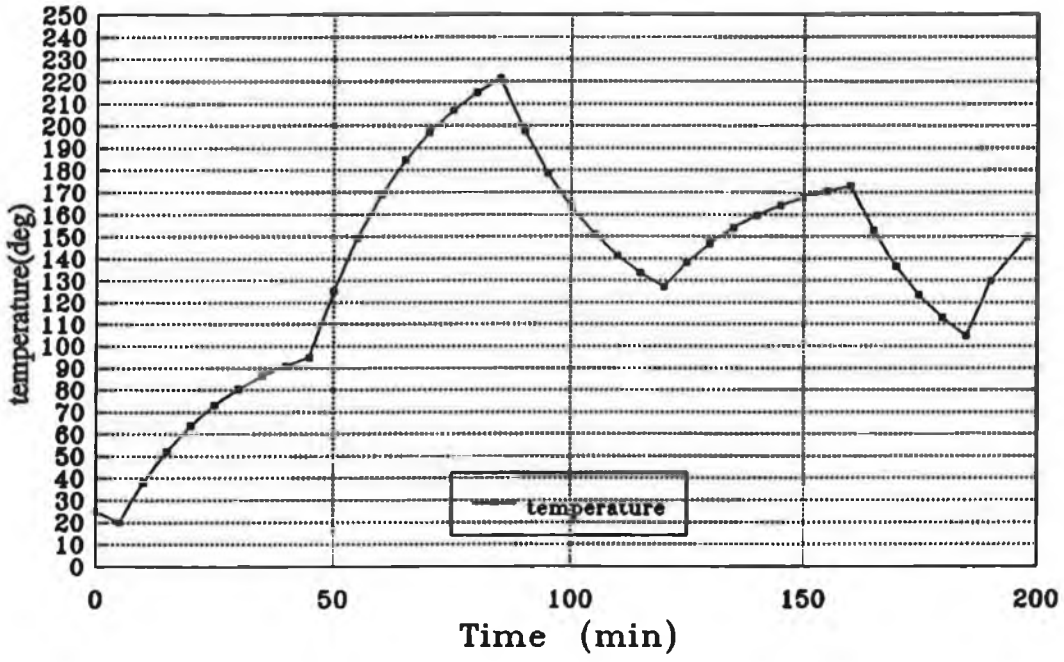


Figure 5-15 Motor temperature without thermal protection

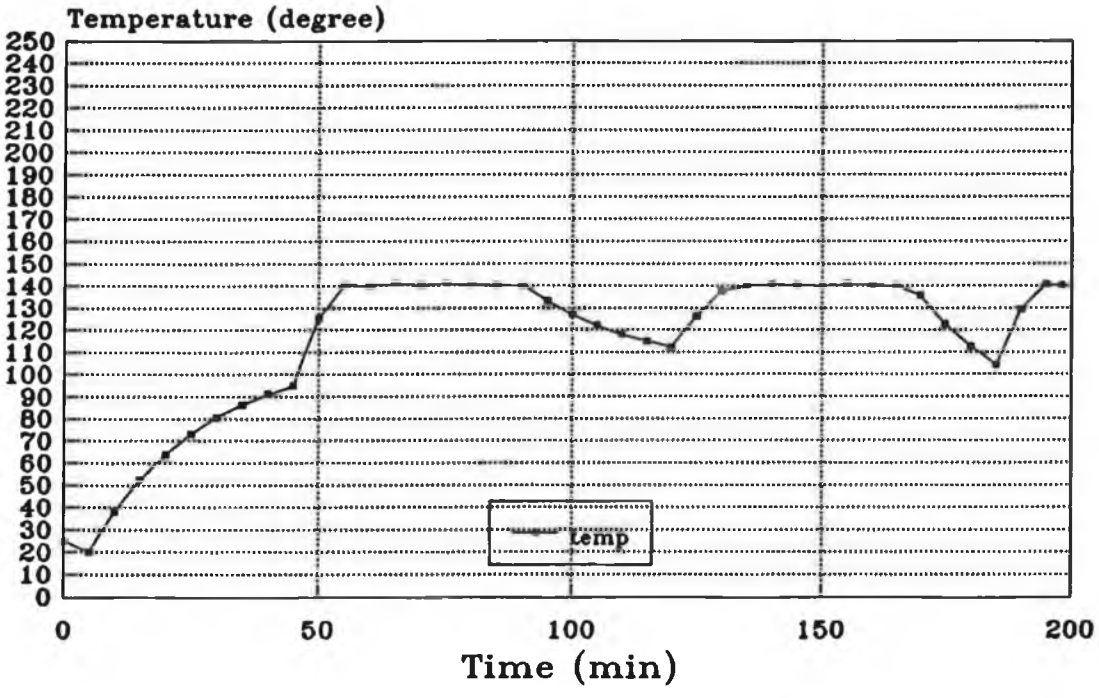


Figure 5-16 Motor temperature with thermal protection

CHAPTER 6

IMPLEMENTING THE TMS320C30 CONTROLLER

6.1. Introduction

The TMS320C30 is a high-performance CMOS 32-bit device in the TMS320 family of single-chip digital signal processors. It has been widely applied to industrial fields such as servo control.

The design of a digital speed controller for the permanent magnet brushless servo system was described previously. This chapter discusses the implementation of the digital controller using the TMS320C30. This includes the description of the DSP architectures and the development of the control software.

Digital controllers for the brushless servo system have advantages over analogue controllers as described in chapter 3. But the high performance permanent magnet brushless servo system requires a very fast response and very high precision. If we use a general purpose microprocessor or a microcontroller, it is difficult to realize a high precision servo control system for the following reasons:

1. Digital controllers monitor signals at discrete time intervals or finite sampling rates. If the signal is not sampled fast enough, some of the information may be lost. The processing of the signal takes a finite amount of time. The processing has to be completed before the arrival of the next sample, or preferably as soon as possible. Too much delay in the output can cause loss of information or

excessive phase delay in the system, leading to instability. These conditions impose certain minimum performance requirements on the processor. Most of the processors currently used to implement controllers are usually not fast enough to process signals in real time, they rely upon lookup tables with precomputed results. [6-4]

2. Digital controllers use discrete steps to represent a signal, which is limited to the wordlength of the processor. Coefficients or gain constants also have to be represented in the limited wordlength. This discretization or loss of resolution is referred to as quantization error. In addition, results of mathematical operations have to fit in a limited wordlength and may lose part of the result due to this limitation. This is referred as truncation errors. Both of these errors cause oscillations or limit cycles and can lead to instability.

3. Another problem that occurs in digital controllers is overflow of registers. Successive mathematical operations can cause registers to overflow. Registers in most processors wrap around, causing the result of a calculation to go from most positive to least negative, in turn causing the output to reverse directions.

Most of these problems described above occur in processors that have 8/16-bit ALUs and registers. This 8/16-bit architecture limits the accuracy of intermediate and final results and generates truncation/quantization errors. Lack of scaling shifters to maintain the required significant bit can cause additional quantization/truncation errors. The general processors also lack the performance to perform fast real time processing, so they rely upon lookup tables, thus limiting precision to low-performance or low-bandwidth systems. If used in higher-performance systems, they can cause excessive loop delays, leading to instability.

The problems discussed above can be eliminated by the TMS320C30. This is because the TMS320C30 not only has an optimized architecture that minimizes numerical problems in signal processing, but also has the performance to meet the bandwidth requirements of high performance systems. The TMS320C30 has a fast execution speed

which minimizes the computation delay time and also allows very fast sampling rates to be implemented for high bandwidth systems. In the TMS320C30, the fast arithmetic operations and high throughput to handle mathematically intensive algorithms in real time are accomplished by using the following concepts: hardware architecture using both the Harvard and the von Neumann types; extensive four-phase pipelining in parallel; a dedicated hardware multiplier; special DSP instruction; and a fast instruction cycle.

The TMS320C30 controller is implemented by a TMS320C30 PC system board which consists of a TMS320C30 processor, expansion memories, a two-channel 16-bit analog interface , a PC interface, a parallel expansion and a serial expansion. It is an integrated 33.3 million floating-point computations per second signal-processing system and provides a high-speed communication between the board and PC.

In this chapter, section 6.2 gives a general description of the TMS320C30 board. Section 6.3 introduces the software design for the speed controller which uses the TMS320C30 assembly language. Section 6.4, introduces a PC control program, which is used to set the control input command and adjust the motor control parameters on PC. In addition, appendix C describes an interface circuit design between the TMS320C30 board and the electronic controller of the servo system.

6.2. The TMS320C30 PC SYSTEM Board Description

6.2.1 Processor

The TMS320C30 DSP is a very fast processor which executes 16.7 million instructions per second for an instruction cycle time of 60 nanoseconds. It has a large memory space with 16 million 32-bit words and floating point arithmetic capabilities (32-bit integer/40-bit floating-point multiplier and ALU). [6-1]

The TMS320C30, the third-generation device in the TMS320C30 family, can perform parallel multiply and ALU operations on integer or floating-point data in a single cycle. The processor also possesses a general-purpose register file, program cache, dedicated

auxiliary register arithmetic units, internal dual-access memories, one DMA channel supporting concurrent I/O, and a short machine-cycle time.

The TMS320C30 can enhance general-purpose application by using the large address space, multiprocessor interface, internally and externally generated wait states, two timers, two serial ports, and multiple interrupt structure.

The TMS320C30 can use high-level language through a register-based architecture, large address space, powerful addressing modes, flexible instruction set, and supports floating-point arithmetic.

6.2.2 Memory Map

The total memory space of the TMS320C30 is 16M 32-bit words. Programs, data, and I/O space are contained within this, allowing tables, coefficients, program code, or data to be stored in either RAM or ROM. RAM block 0 and 1 are each 1K x 32 bit. The ROM block is 4K x 32 bits. Each RAM and ROM block is capable of supporting two accesses in a single cycle. They are separate program buses, data buses.

6.2.2.1 Memory Maps

The TMS320C30 board uses microprocessor mode memory map as shown in figure 6-1 with both the TMS320C30 DSP memory maps and TMS320C30 PC SYSTEM BOARD memory maps.

6.2.2.2 Peripheral Bus Map

The memory-mapped peripheral registers are located starting at address 808000h. The peripheral bus memory map is shown in figure. 6-2. Each peripheral occupies a 16-word region of the memory map.

6.2.2.3 Reset/Interrupt/Trap Vector Map

The address for the reset, interrupt, and trap vectors are 0h through 3Fh, as shown in figure. 6-3. The vectors stored in these locations are the addresses of the start of

0h BFh	INTERRPT LOCATIONS AND REVERED (192) EXTERNAL STRB ACTIVE
C0h 7FFFFFh	EXTERNAL STRB ACTIVE
800000h 801FFFh	EXPASION BUS MSTRB ACTIVE (8K)
802000h 803FFFh	RESERVED (8K)
804000h 805FFFh	EXPASION BUS IOSTRB ACTIVE (8K)
806000h 807FFFh	RESERVED (8K)
808000h 8097FFh	PERIPHERAL BUS MAPPED-MEMORY REGISERS (INTERNAL) (6K)
809800h 809BFFh	RAM BLOCK 0 (1K) (INTRNAL)
809C00h 809FFFh	RAM BLOCK 1 (1K) (INTRNAL)
80A000h 0FFFFfFh	EXTERNAL STRB ACTIVE

Microprocessor mode

BANK	SIZE (WORDS)	WAIT STATES	ADDRESS (hex)
0 Area A	64K	0/1	000000 00FFFF
1 Area A	64K	0/1	010000 01FFFF
2 Area A	64K	0/1	020000 02FFFF
3 Area B	16k or 64K	0/1	030000 03FFFF
Memory Expsion Connector	7963K	User Defined	040000 7FFFFF
DSPLINK	8K	2	800000 901FFF
Reserved	8K		802000 803FFF
Analog I/O & PC Interrupts	8K	2	804000 805CFF
Reserved	8K		806000 807FFF
On-Chip Perigherals	6K	Internal	808000 8097FF
RAM 0	1K	Internal	809800 809BFF
RAM 1	1K	Internal	809C00 809FFF
Memory Expsion Con.	8152K	User Defined	80A000 FFFFFF

TMS320C30 Board Memory Map

Fig. 6-1 Memory Map

respective reset, interrupt, and trap routines.

6.2.2.4 External Memory Map

The board provides two memory areas off-chip which are divided into areas **A** and **B** as shown in figure. 6-1. The external memory area **A** is divided into three areas again. One 64K bank is populated on delivery with zero wait state devices. External memory uses a hardware wait state generator to accommodate the different access times required by the various memory areas. The board is equipped with 64 KWords of one wait state memory in the area **B** at the address 30000h upwards. It is intended that this memory

808000h 80800Fh	DMA CONTROLLER REGISTERS (16)
808010h 80801Fh	RESERVED (16)
808020h 80802Fh	TIMER 0 REGISERS (16)
808030h 80803Fh	TIMER 1 REGISERS (16)
808040h 80804Fh	SERIAL PORT 0 REGISERS (16)
808050h 80805Fh	SERIAL PORT 1 REGISERS (16)
808060h 80806Fh	PRIMARY AND EXPASION PORT REGISERS (16)
808070h 80807Fh	RESERVED (16)

Figure 6-2 Peripheral bus memory map

00h	RESET
01h	INT0
02h	INT1
03h	INT2
04h	INT3
05h	XINT0
06h	RINT0
07h	XINT1
08h	RINT1
09h	TINT0
0Ah	TINT1
0Bh	DINT
0Ch...	RESERVED...
20h...	TRAP0...
3Ch...	TRAP 28 (RESERVED)...
3Fh	TRAP 31 (RESERVED)

Figure 6-3 Reset, Interrupt, and Trap vector locations

be used for the transfer of the data between PC and the TMS320C30.

6.2.3 Analog interface

It is necessary for the TMS320C30 based servo control system to use A/D and D/A conversion. The TMS320C30 PC SYSTEM BOARD contains a complete "analog I/O subsystem". There are two separate channels, each containing its own sample/hold amplifier, A/D, D/A, and analog filters on input and output. The analog I/O signals are accessed via the "end-plate", which is accessible through the back of the PC. The A/D's are Burr-Brown PCM78P devices which offer 16-bit precision with up to 200 kHz sample rates. The sample/hold amplifiers and D/A converters are also Burr-Brown devices which are matched to the capabilities of the A/D. The sample/hold device is the SHC5320, and the D/A device is the PCM56P. The conversion is triggered by timer 1 (the 32-bit on-chip timer) which can be programmed by software to a resolution of 120nsec. The analog I/O subsystem is accessed through three 16-bit secondary I/O bus mapped registers. They are in the region between the address 804000 hex and 804008h as shown in Table 1.

Table 1 Analog I/O Mapping

Address	Function
804000h	Read/Write Channel A's A/D & D/A
804001h	Read/Write Channel B's A/D & D/A
804008h	Software Conversion Trigger

Although each register consumes 32 bits of space, only the top 16 bits of each register are used. The first two registers are used to access the A/D and D/A converters on the two Analog I/O channels. The A/D's and D/A's deal with data in 16-bit 2's complement format. All the registers are accessed with two memory wait states. This results in an overall time of 180 nsec to access each register.

6.3. Software design for TMS320C30 board

6.3.1 Selection of Software tools

The TMS320C30 is well supported by a full set of software development tools which include assembly-language, C language or a mixed combination. In brushless motor control, the real time control program must execute as fast as possible. In our application, the program is written in assembly-language. Although C is easier to use than assembly, it is slower in the control mode. For example, most of DSP algorithms spend the vast majority of execution time on a small section of code (Normally 90 percent of the time is spent on 10 percent of the code.) [6-8].

6.3.2 Program flowchart

The program flowcharts are shown in figure 6-4. The TMS320C30 assembly language uses a number of 'sections', which are relocatable blocks of code or data, to execute a program. In the program, interrupt vectors are located in the section "*.init*" beginning at address 0h. The "*.data*" section contains register addresses, control words and initial variables. The ".text" section executes the program which is divided into two main parts the initial routine and control routine as described below.

6.3.2.1 Initializing routine

This routine consists of a page pointer, stack pointer, external bus operation, timer, and wait states.

(1) Data Page Pointer

In order to use direct addressing, the page pointer is set up first. In the direct addressing mode, the data address is formed by the combination of the 8 least significant bits of the data page pointer (DP) with the 16 least significant bits of the instruction word (expr). This produces 256 pages of space, and gives the programmer enough memory space without needing to change the page pointer. The syntax of the page pointer is LDP which is expressed as :

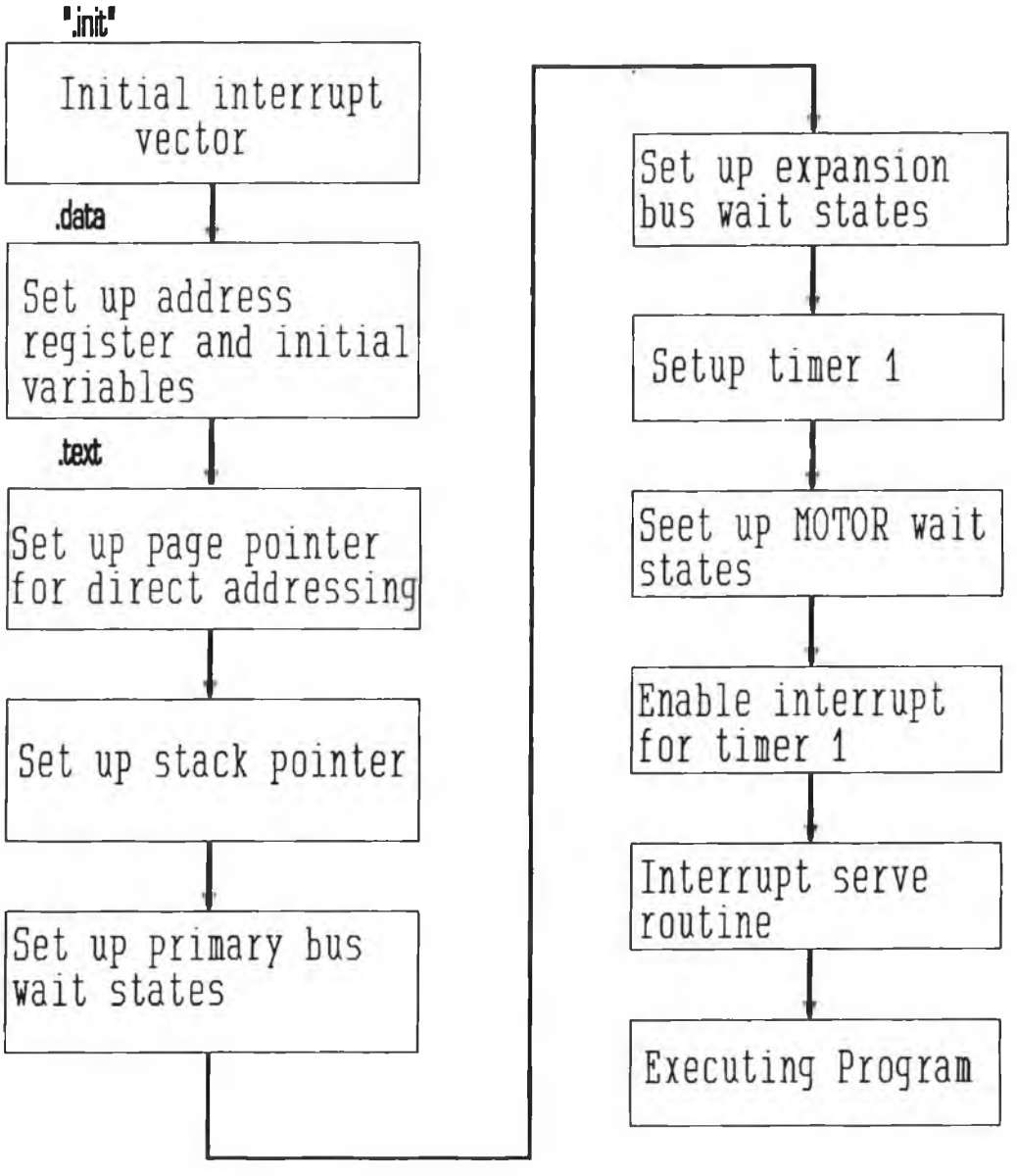


Figure 6-4. Program Flow-chart diagram

[*label*] **LDP** *expression* [*.register*]

where the *expression* is a relocable address, which is usually represented by a symbol name. The 8 MSBs of the address are loaded into the destination register. If a register is not specified, the assembler will use the DP register as a default which is expressed as:

.data

```
PRIMCTL   .word   00808065h
          .text
      LPD   PRIMCTL
```

The *PRIMCTL* as an address variable located at the ".data" section, the LDP instruction loads the DP register with the 8-bit pointer to the page on which *PRIMCTL* is located. Note that the *register* operand was omitted from the *LDP* instruction; *DP* was used as the default.

(2) Stack pointer

The TMS320C30 provides a dedicated system stack pointer (SP) for building stacks in memory. The program counter is pushed on the system stack on interrupt. It is popped from the system stack on return. The syntax of the stack pointer is expressed as:

```
          .data
RSP       .word     STACK
STACK          .word     0
          .text
LDI       @RSP,SP
```

where *SP* is the stack pointer.

(3) External Bus Operation

Two external interfaces are provided on the TMS320C30: the primary bus and the expansion bus. The TMS320C30 processor uses them to access data between memory and external peripheral devices. The primary bus consists of a 32-bit data bus and a 24-bit address bus. The expansion bus consists of a 32-bit data bus and a 13-bit address bus. Both the primary bus and the expansion bus have associated control registers which are memory-mapped in table 2.

Table 2. Memory-mapped External Interface Control Register

Peripheral Address	Register
808060h	Expansion bus control
808064h	Primary bus control

For the TMS320C30 SYSTEM BOARD operation, the external memory area **A** must always be configured via the primary bus control register to respond only to externally-controlled wait states. That allows external hold requests and can have a 64KWord bank size. Bits 8-12 of the primary bus control register called BNKCMP determine the size of the bank as shown in table 3.

Table 3. BNKCMP and Bank Size

BNKCMP	MSBs DEFINING A BANK	BANK SIZE (32-BIT WORDS)
00000	None	2^{24} = 16M
00001	23	2^{23} = 8M
00010	23-22	2^{22} = 4M
00011	23-21	2^{21} = 2M
00100	23-20	2^{20} = 1M
00101	23-19	2^{19} = 512K
00110	23-18	2^{18} = 256K
00111	23-17	2^{17} = 128K
01000	**23-16**	**2^{16} = 64K**
01001	23-15	2^{15} = 32K
01010	23-14	2^{14} = 16K
01011	23-13	2^{13} = 8K
01100	23-12	2^{12} = 4K
01101	23-11	2^{11} = 2K
01110	23-10	2^{10} = 1K
01111	23-9	2^{9} = 512
10000	23-8	2^{8} = 256
10001 to 11111	Reserved	Undefined

The bold number **01000** determines the bank size of the area **A** which is 64Kwords. After a reset, the primary bus control register sets a control word **800h** corresponding to **01000**. The expansion bus control register is set up by writing **0h**. The program for

the external operation is:

```
                .data
PRIMCTL     .word       00808064h    ; Primary  bus
EXPCTL      .word       00808060h    ; Expansion bus
PRIMWD      .word       00000800h    ; Primary bus control word
EXPWD           .word       00000000h    ; Expansion bus control word

                .text
; Set up primary bus wait states
        LDI @PRIMCTL, AR0
        LDI @PRIMWD, R0
        STI R0, *AR0
; Set up expansion bus wait states
        LDI @EXPCTL, AR0
        LDI @EXPWD, R0
        STI R0, *AR0
```

(4) Timer set up

The TMS320C30 provides two internal timers, timer 0 and timer 1. The TMS320C30 system board uses the timer 1 to signal the external A/D converter to start a conversion with an internal clock. The timer modules are general-purpose 32-bit timer/event counters with two signalling modes and internal or external clocking as shown in figure 6-5. Three memory-mapped registers are used by the timer which are the global control register, period register and counter register. The timer global control register is a 32-bit register that contains the global and port control bits for the timer module in which the bits 3 to 0 are the port control bits and the bits 11 to 6 are the timer global control bits. The global control register determines the timer operating mode, monitors the timer status, and controls the function of the I/O pin of the timer. The 32-bit timer period register specifies the timer's signalling frequency. The counter register is also a 32-bit register which contains the current value of the increment counter. The counter is zeroed whenever its value equals that in the period register. The memory map of

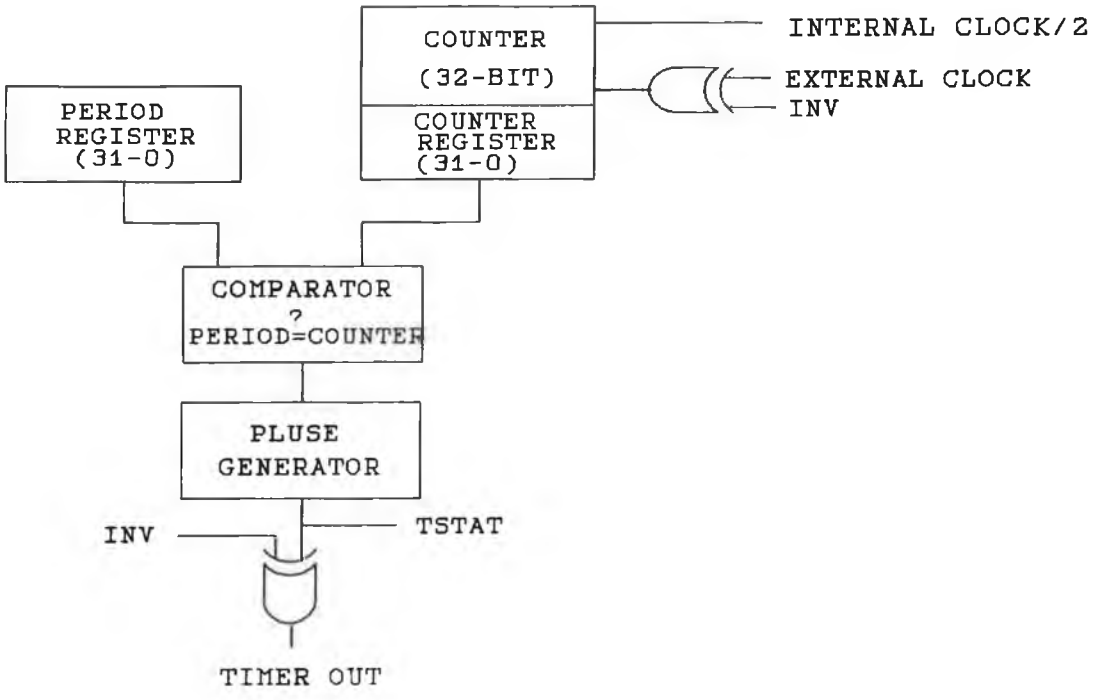


Figure 6-5 Timer Block Diagram

timer 1 is shown in Table 4.

Table 4. Memory-Mapped Timer 1 Locations

Address	Register
808030h	Timer Global Control Register
808034h	Timer Counter Register
808038h	Timer Period Register

The pulse generator, in figure 6-5, generates two basic modes of clock signals, the pulse mode and clock mode. In both modes, an internal clock source has a frequency of f(H1)/2, and an external clock source has a maximum frequency less than f(H1)/2, where the f(H1) is the clock frequency of the TMS320C30. Bit 8 of the global control register, C/P, determines the mode which is used. When C/P=1, the clock mode is chosen and the state flag and external output will have a 50 percent duty cycle. When C/P=0, the pulse mode is chosen and the status flag and external output will be active for one H1 cycle during each timer period. The TMS320C30 board uses pulse mode.

The timer signalling rate is determined by the frequency of the timer input clock and the period register. The equation is expressed as:

f(pulse mode) = f(timer clock)/period register

The internal clock frequency is f(H1)/2, H1= 60nsec. So the sample period is:

Time period = Value of Period register x 120 (nsec)

The timer can receive its input and send its output in several different modes, depending upon the setting of CLKSRC, FUNC, and I/O. Bit 0 of timer global control register is called FUNC which controls the function of TCLK. When FUNC = 0, TCLK is configured as a general-purpose digital I/O port. If FUNC=1, TCLK is configured as a timer pin. Bit 9 of the timer global register is called CLKSRC which specifies the source of the clock. When CLKSRC = 1, an internal clock, with frequency which equals to one-half H1 frequency, is used to increment the counter. When CLKSRC = 0, an external signal from the TCLK pin can be used to increment the counter. Timer 1 is used for triggering A/D conversion, the internal clock is chosen to output a signal to the external bus. So FUNC and CLKSRC are both 1 as shown in figure 6-6.

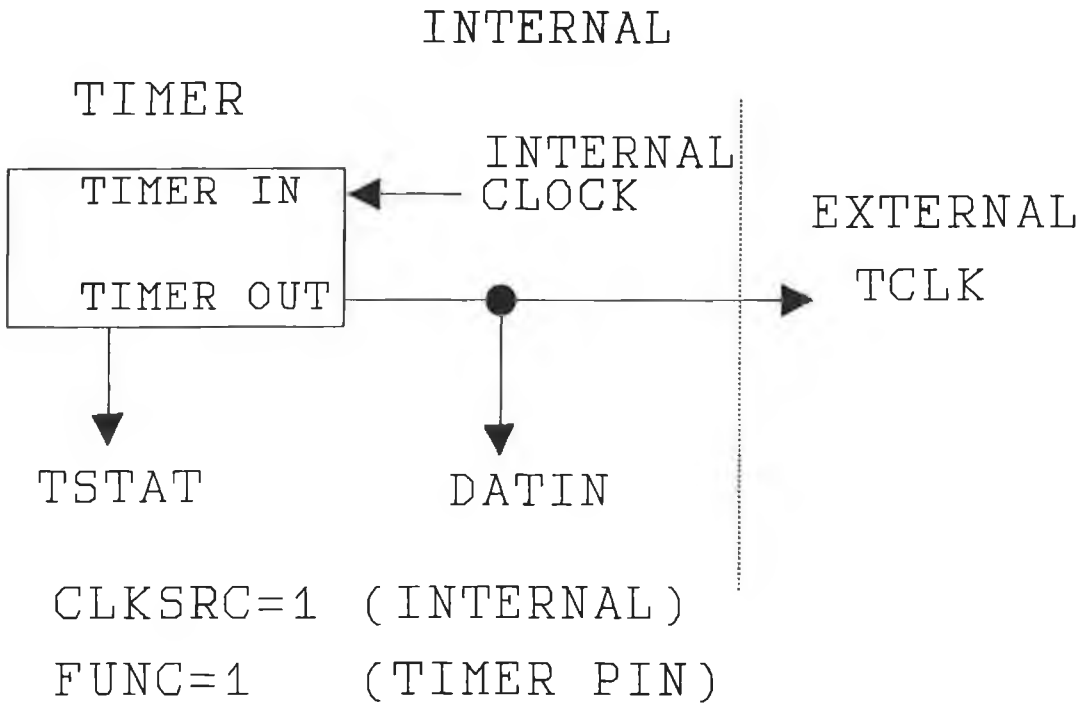


Figure 6-6 Timer Mode

Bit 10 of the global control register is the inverter control bit and is called INV. If INV=1, the output of the pulse generator is routed to TCLK. If INV=0, no inversion is performed on the output or input of the timer. Bits 7 and 8 of the global control register control the timer reset, start and hold as shown in table 5.

Table 5. Result of a Write of Specified Values of GO and HLD

GO	HLD	RESULT
0	0	All timer operations are held. No reset is performed.
0	1	Timer proceeds from state before write.
1	0	All timer operations are held, including zeroing of the counter. The GO bit is not cleared until the timer is taken out of hold.
1	1	Timer reset and started.

The timer global control register is shown in figure 6-7.

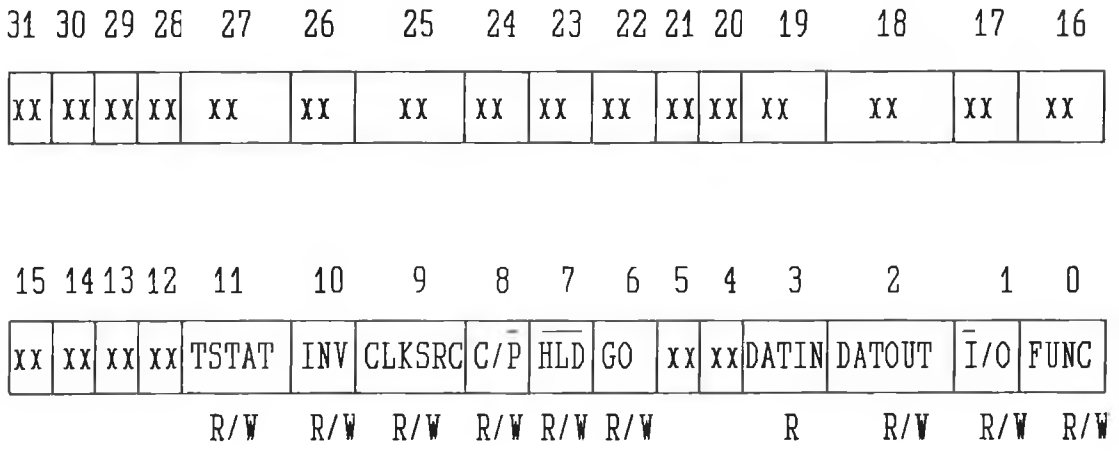


Figure 6-7 Timer Global Register

When resetting the timer, 601h is loaded into the global control register, where the respective bits 0, 9, and 10 are set to 1. While starting the timer, 6C1h is used and the respective bits 0, 6, 7, 9 and 10 are set to 1. The program for setting timer 1 is:

```
                .data
TIMECTL         .word       00808030h    ; Timer 1 control register
PERIOD          .word       00808038h    ; Timer 1 period register
RSTCTRL         .word       00000601h    ; Reset control word
SETCTRL         .word       000006C1h    ; Start timer control word
SAMPLE          .word       2000         ; Sample period
                .text
;Set up timer 1.
        LDI @TIMECTRL, AR0      ; Reset control register
        LDI @RSTCTRL, R0
        STI R0, *AR0
        LDI @PERIOD, R0         ; Set period register
        LDI @SAMPLE, R0
        STI R0, *AR1
        LDI @SETCTRL, R0        ; Start timer 1
        STI R0, *AR0
```

(5) Interrupt control

The TMS320C30 can execute an interrupt control in different ways as shown in figure 6-8.

The TMS320C30 system board uses the CPU external hardware interrupt. The logic function diagram used to implement the interrupt is shown in figure 6-9.

After the timer starts an A/D conversion, the A/D performs the conversion, then outputs an end-of-convert signal which triggers the INTL interrupt request.

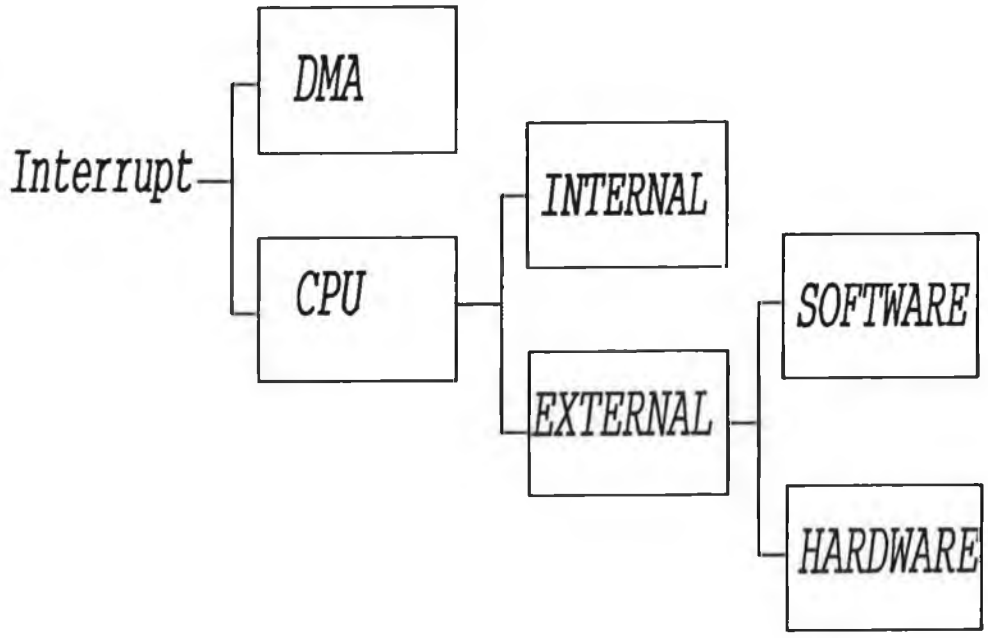


Figure 6-8. Interrupts

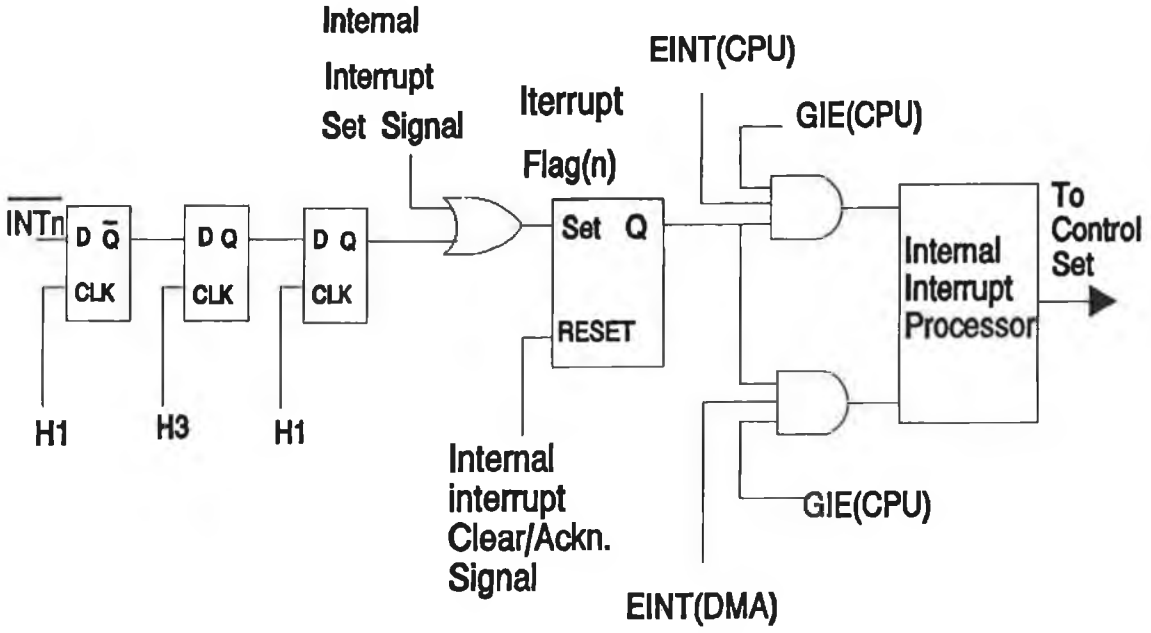


Figure 6-9. Interrupt Logic Functional Diagram

Interrupts are synchronized internally as illustrated by the three flip-flops clocked by **H1**

and H3. Once synchronized, the interrupt input will set the corresponding interrupt flag register (IF) bit if the interrupt is active. When a particular interrupt is processed by the CPU, the corresponding interrupt flag bit is cleared by the internal interrupt acknowledge signal for one cycle and then set to 1 again. When the TMS320C30 is reset, zero is written to the interrupt flag register, thereby clearing the pending interrupt. To enable the interrupt, a '1' must be written to bit 1 of the TMS320C30's IE register (the CPU interrupt enable bit EINT1 is set 1 to enable an interrupt) and a '1' to bit 13 of store register (ST). The CPU global interrupt enables bit GIE, located in the CPU store register (ST), and controls all CPU interrupts. The interrupt processing is shown in figurer 6-10.

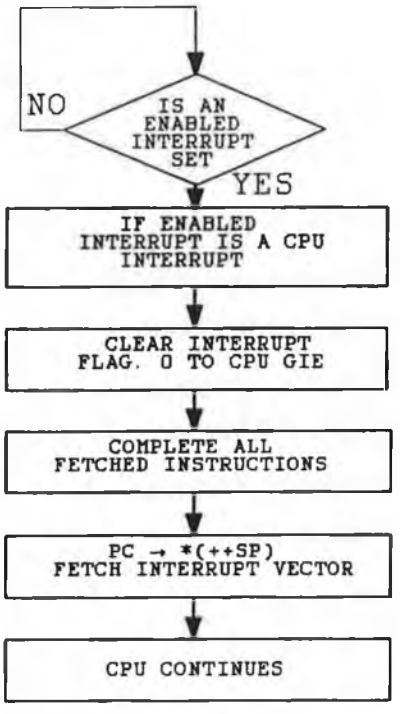


Figure 6-10. Interrupt Processing

6.3.2.2 Control routine

The control routine executes the interrupt service routine and the main control loop as shown in figure 6-11. The interrupt service routine samples signal data from the A/D convert, channel A, and then transforms the data to TMS320C30 floating point format. A flag is used to interface between the interrupt routine and the main control loop which

completes a control operation and outputs the control signal to the D/A converter.

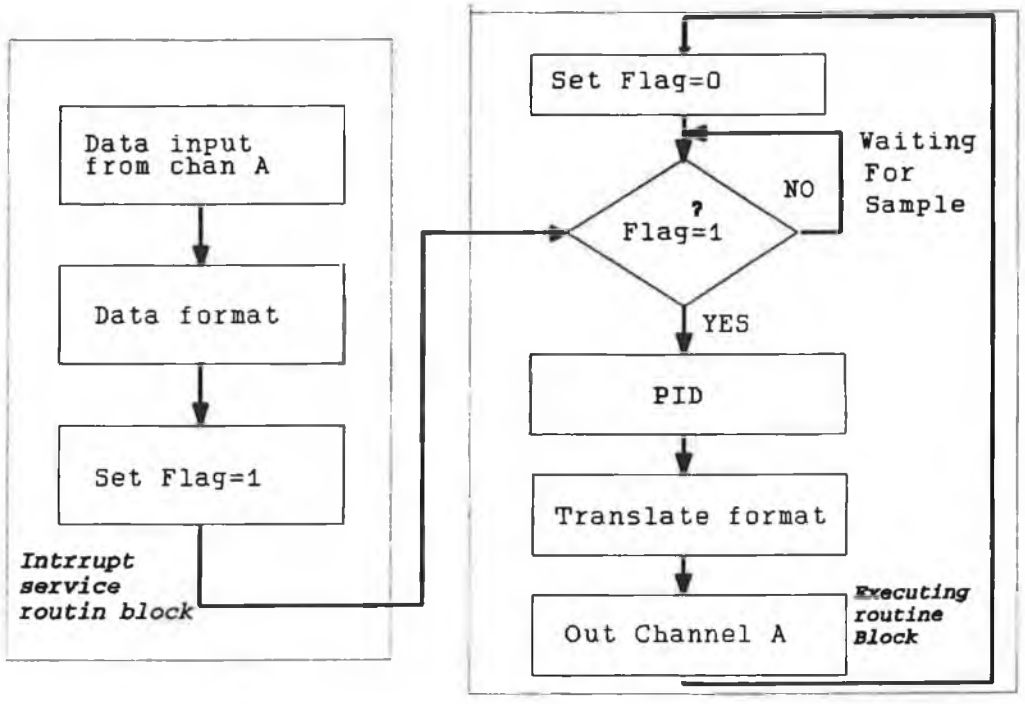


Figure 6-11 Program Flow chart 2

(1) Data format transformation

Data is fetched from the A/D converter using an I/O mapped register located at 804000h. The register is 32 bits long, but only the most significant 16 bits are used in order to handle data in 16-bit 2's complement format (a 16-bit integer, from the TMS320C30's point of view). The TMS320C30 supports two integer formats: a 16-bit short integer format and a 32-bit single-precision integer format. Since data is put on the top 16 bits of the I/O register (the remainder are set to 0), it is a 32-bit integer format, and not a 16-bit integer format. The 40-bit extended-precision register is used as the integer operand, in which only bits 31-0 are used, bits 39-32 are unchanged. Before the extended-precision register fetches data from the I/O register using an INTEGER instruction, it should clear all bits of the extended-precision register each sample time using a floating-point number **1.0**. In the extended-precision floating-point format, a floating-point number is represented by an 8-bit exponent field (*e*) and a 32-bit mantissa field (*man*) with an implied most-significant nonsign bit as shown in figure 6-12. The floating point number $_{x}$ = 0.0 can not be used to clear all bits of the extended-

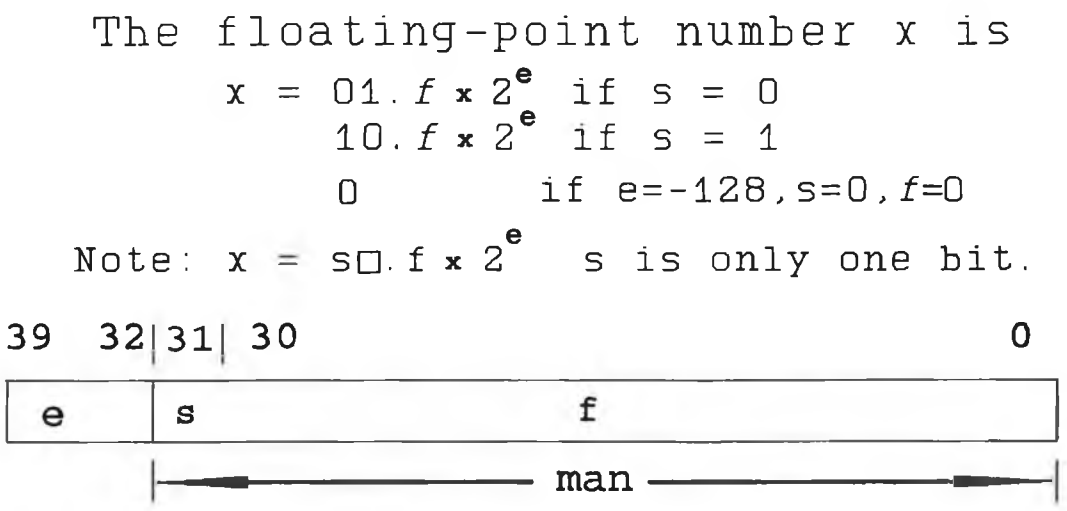


Figure 6-12 Extended precision floating point format

precision register, since x=0.0=1000000000h (the most significant bit is 1). While the floating-point number x=1.0=0000000000h

where $e = 0$

$s = 0$

$f = 0$

This can be used to clear all bits of the extended-precision register.

The analog input to the A/D ranges between -3V and +3V. Voltages between 0 and 3V are converted to TMS320C30 floating-point format numbers between 1.0 to 2.0, and voltages between 0 and -3V are converted to numbers between -1.0 to -2.0. In order to use the data to implement the control algorithms, 1 is subtracted from the positive numbers and 1 is added to the negative numbers. Therefore analog input values between -3V to +3V are converted to floating numbers between -1.0 to +1.0.

(2) Program control transfer

Program control is transferred from the interrupt service routine to the main control loop using a specified flag pointer, FLAG, as shown in figure 6-11. In the interrupt service routine, FLAG is set to 1, while in the main control loop it is set to 0 when waiting for an interrupt.

(3) Motor control routine

In this routine, for design I, the motor control algorithm uses:

$$u(n) = u(n-1) + 9.39[e(n) - 0.99335e(n-1)]$$

which is described in section 3.4.3.

For the design II, the control program can be written as

$$e(kT0 = r(kT) - c(kT)$$

$$u_1(kT) = K_I e(kT) + u_1[(k-1)T]$$

$$u(kT) = -K_P c(kT) - K_D\{c(kT) - c[(k-1)T]\} + u_1(kT)$$

which is detailed in section 3.5.

(4) Data output

Data outputting to the D/A is the opposite to data inputting from the A/D. However, the output data must be $-2.0 < x < 2.0$, otherwise, we have to let $x = -1.99999999$ when $x \leq -2.0$, and $x = 1.99999999$ when $x \geq 2.0$. Note that the floating-point number x can not be - 2.0 or 2.0 which is zero when being converted to the integer format form.

6.4. PC control program

A PC control program is used to control the TMS320C30 board. It downloads the DSP program to the TMS320 board, sets coefficients for the DSP program and sets the input commands. The PC program allows the user to modify parameters of the controller. The flowchart of the program is shown in figure 6-13. The program, which is written in C, is presented in appendix B.

6.5 Conclusion

This chapter has discussed implementation of the digital controller for the brushless servo system using the TMS320C30 processor. This system has numerous advantages over analogue-based designs because its high processing speed allows sophisticated digital control techniques to be used to build a high-precision servo control system. The

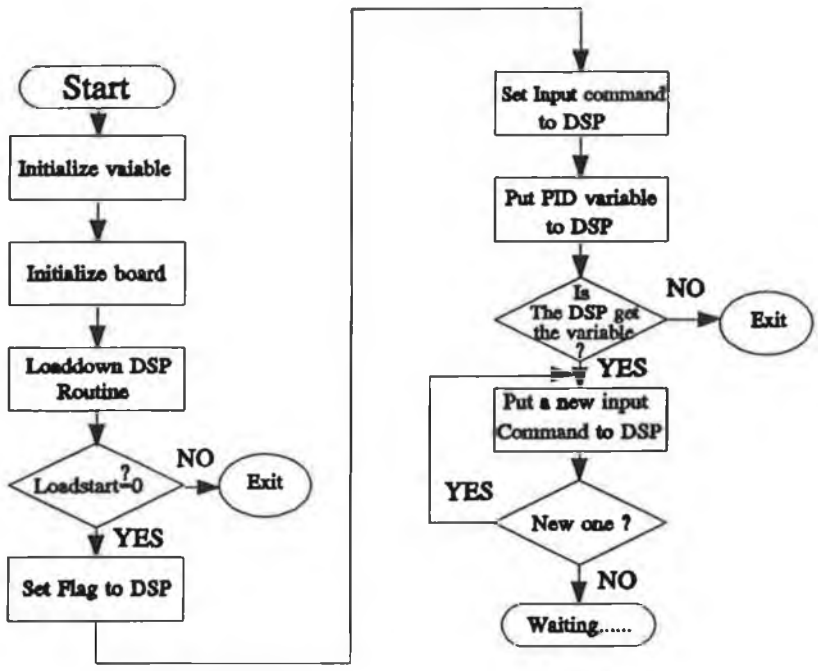


Figure 6-13 The flowchart of the PC control program

digital system is insensitive to component ageing and temperature drift, which minimizes variation in the controller gain coefficients. In addition, the TMS320C30-based system makes the quantization and truncation errors of the digital system negligible due to its high accuracy, high processing speed and improved structure.

Real time control programs for the TMS320C30-based system were developed. With high sampling rates, the fast speed of the TMS320C30 gives an analogue-like performance and minimizes the delay time for the brushless servo drive.

CHAPTER 7

PERFORMANCE MEASUREMENT

7.1 Introduction

In the previous chapters, we described the design of the digital controller for the brushless motor servo system and the implementation on the DSP-TMS320C30. The testing of the designed brushless motor servo system is presented in this chapter. In order to identify the design performance of the system, a series of measurements of the speed responses need to be performed, the results show a high performance of the variable speed control for the TMS320C30 based servo drive system.

7.2 Experimental Equipment and Procedure

The experimental setup consists of a BHT 2214M brushless motor, a BHT controller, a BHT power supply, a TMS320C30 PC system board, an IBM PC and a HP digital oscilloscope as shown in figure 7-1. The servo drive system includes the motor, the controller, the power supply and the TMS320 board. A PC is used to operate this system. The dynamic characteristics of this system are measured by the oscilloscope and the waveforms are sent to PC. The whole procedure is described as follows.

The three phase AC 380 volt power source is fed into a transformer which reduces the voltage to three phase 220V. The BHT power supply rectifies the AC 220V voltage to DC 300 V which is required by the inverter in the BHT controller. An interface circuit is built to converts +/- 10V signal voltage to +/- 3V for the TMS320C30 system board. The signal from the tachogenerator is fed into the A/D converter of the TMS320C30 board. The signal of the D/A converter is fed to the current loop of the BHT controller

to control the servomotor.

The speed performances are measured at the test point of the tachogenerator on the BHT controller board using the HP digital oscilloscope. The waveforms of the oscilloscope are drawn using a drawing package HG.

7.3 Tests of the servo system

7.3.1 Design I Test

The design I system is tested to investigate the design quality. A speed step response from 3000 rpm to 0 is presented to the design I system which can demostrate the overshoot. Figure 7-2 (a) shows the measured result, the overshoot is 10.3% and the settling time is about 0.07 sec. Figure 7-2 (b) is the simulated step response, respectively. The overshoot is about 10% and settling time is 0.064 sec. Both results show the design meets the requirement in the section 3.4.2 where the desired overshoot is 8.1% and the settling time is 0.062 sec. This proves the design is successful since it eliminates steady-state error and has a fast dynamic response.

7.3.2 Design II Test

The design II system is tested to inspect the system dynamic response. The speed step response from 3000 rpm to 0 is presented. Figure 7-3 (a) shows the measure result of the speed dynamic response, the overshoot is 22% and the settling time is about 0.028 sec. Figure 7-3 (b) is the simulated step response, respectively, in which the overshoot is about 20% and the settling time is 0.025 sec. The requirements of the design II system is an overshoot of 20% and a settling time of 0.02 sec. Both results show the design meets the requirements since it has a faster dynamic response than the design I, also it eliminated the steady-state error and it is controlled within the stability margin.

7.3 Sensitivity to controller parameters

For digital PID implement, the system performance is sensitive to changes in the controller parameters K_P, K_I and the DSP sampling rate. Design II system is used to observe the effects. First, we investigated the effect of varying the parameter K_P as show in figure 7-4. K_P was varied from 25 to 20, 18 and 15 with the rest parameters constant: K_I=0.25, K_D=50 and the sample rate=0.12 msec. The figures show that with larger the value of K_P, the overshoot of the responses becomes smaller, hence K_P has significant affect on the system. Second, we investigated the influence of K_I on the system as shown in figure 7-5. K_I was adjusted from 0.15 to 0.3, where the rest of parameters were constant: K_P=15, K_D=50 and sample rate = 0.12 msec. The figures show that with larger K_I, the overshoot of the responses becomes higher, so K_I also has a significant affect on the system. Then, we investigated the influence of the sample rate on the system as show in figure 7-6. The sample rates were varied from 0.06 msec to 1 msec with the rest of the parameters were constant: K_P=15, K_I=0.25 and K_D=50. The figures show that with faster the sample rate, the overshoot of the responses becomes higher, therefore the sample rate affects the system significantly as well.

This sensitivity is a disadvantage of the feedback and forward PID control, but if we use grapho-analytical method of pole-placement, suitable parameters can be selected for a desired performance of the system. Therefore we need not to do much trial and error work to adjust the system.

7.4 Conclusion

The simulation and experimental results show the designed DSP based brushless drive system is able to implement a high performance servo operation. Both designs can eliminate steady-state error, have a fast dynamical response and have stability in the steady-state. However, design II has a much faster dynamical response than design I, this is very important for the high-performance servo applications. Although the PID control is sensitive to the parameters of the controller, we can obtain a desirable performance using the graph-analytical method of pole-placement.

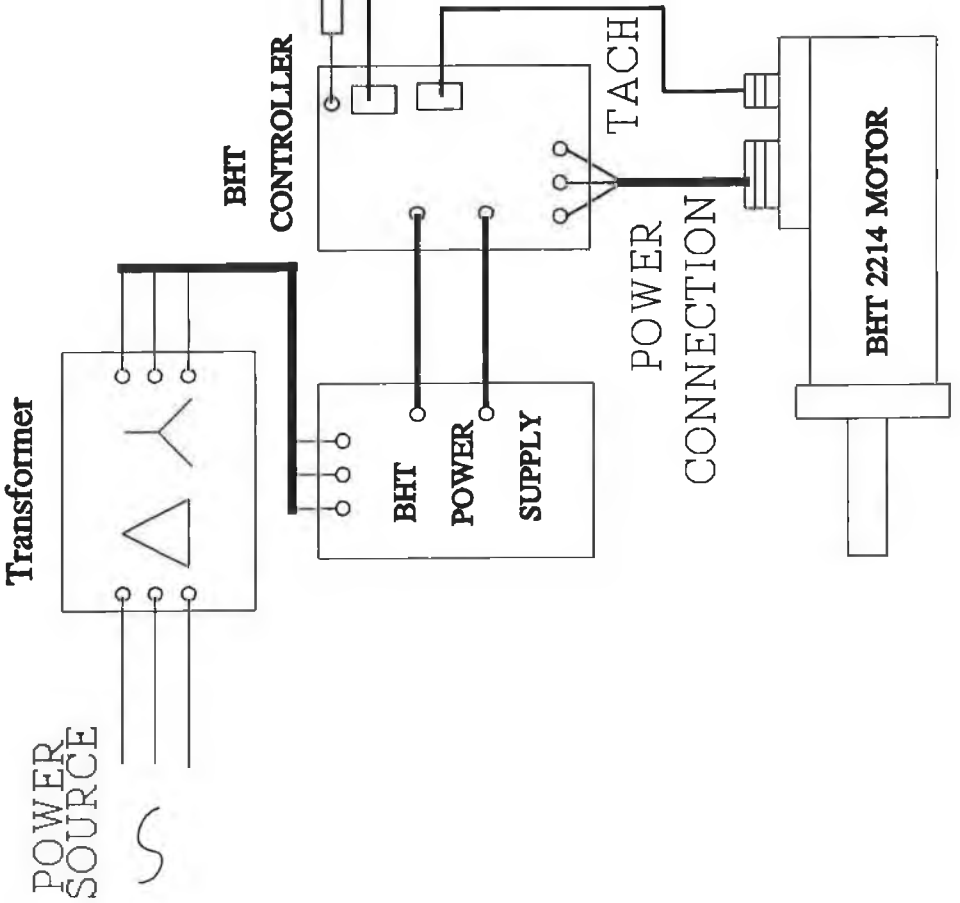
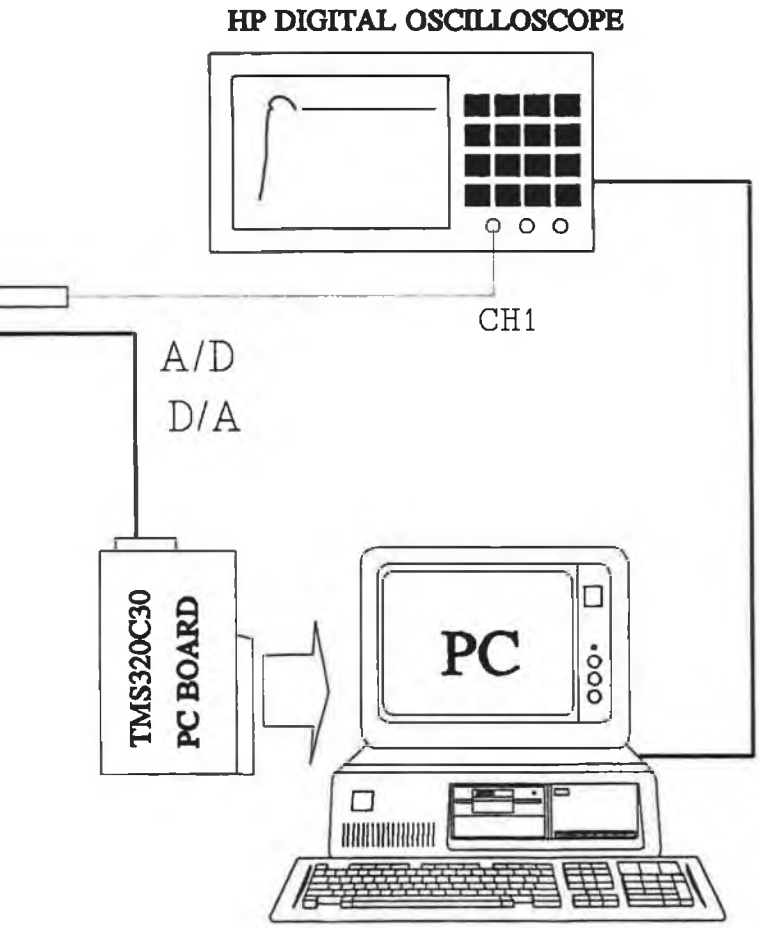


Figure 7-1 Brushless servo motor drive system configuration


HP DIGITAL OSCILLOSCOPE
CH1
A/D
D/A
TMS320C30
PC BOARD
PC

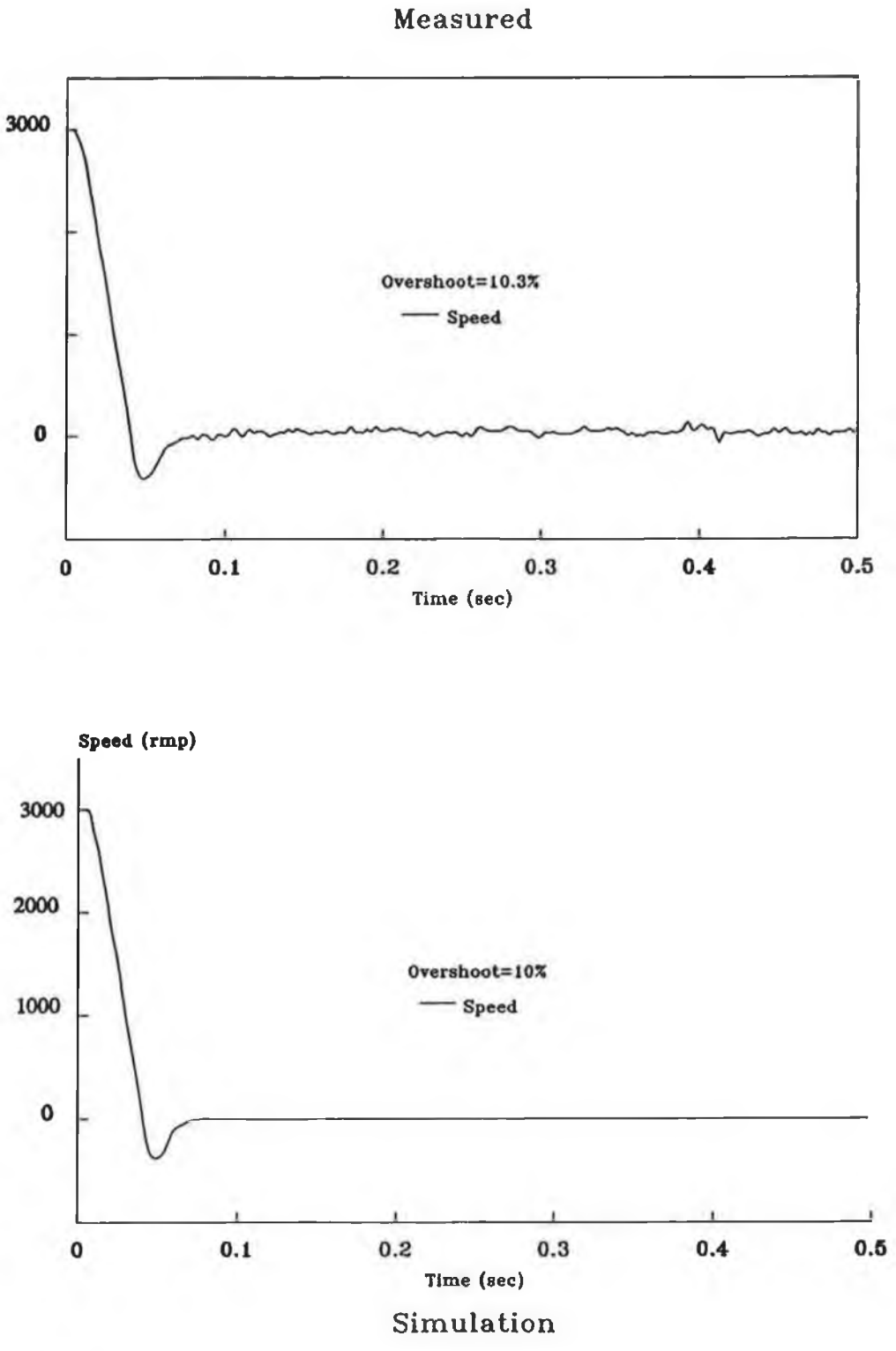


Figure 7-2 Measured & Simulated speed step response of Design I

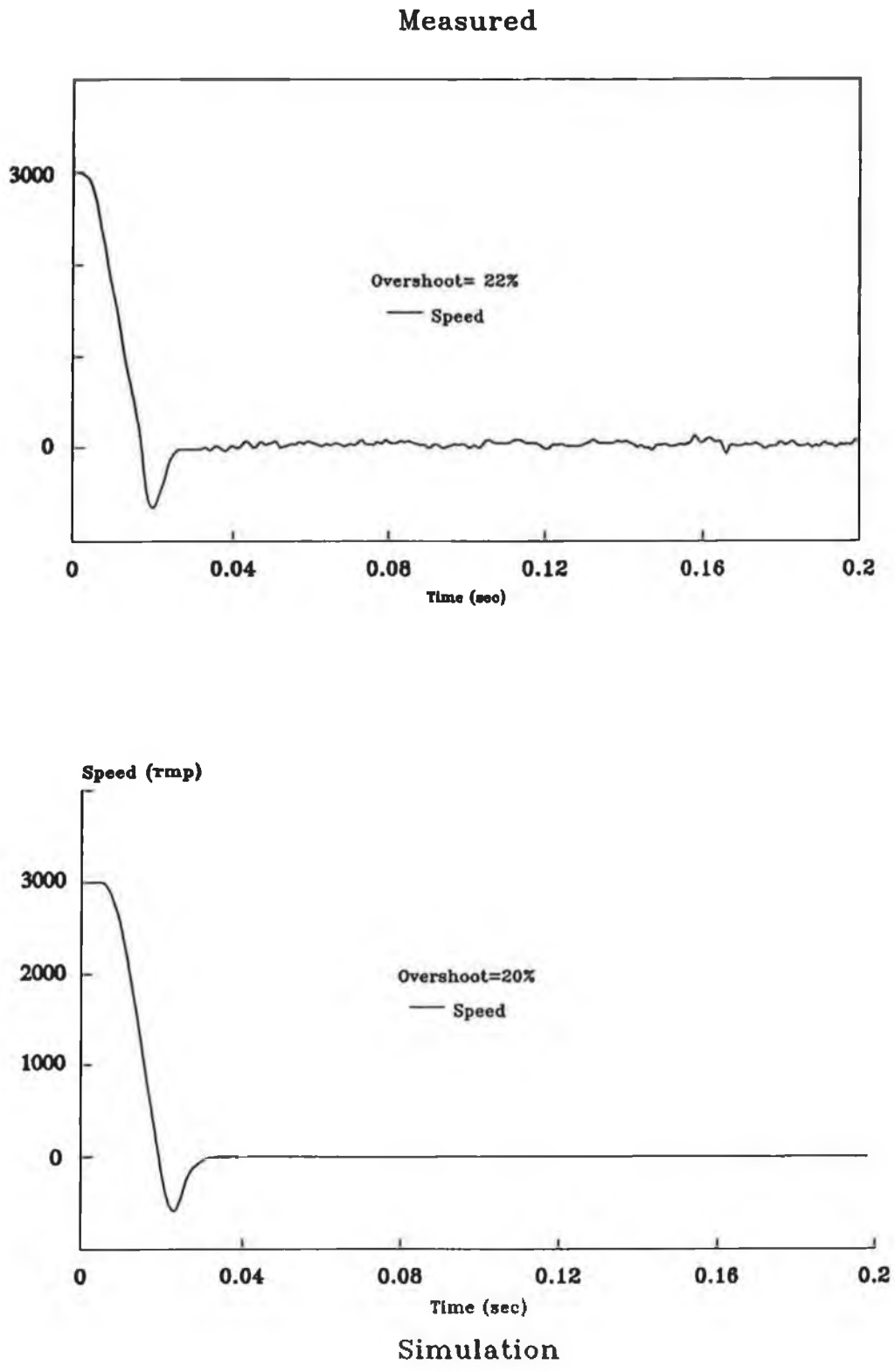


Figure 7-3 Measure & Simulation speed response of the Design II

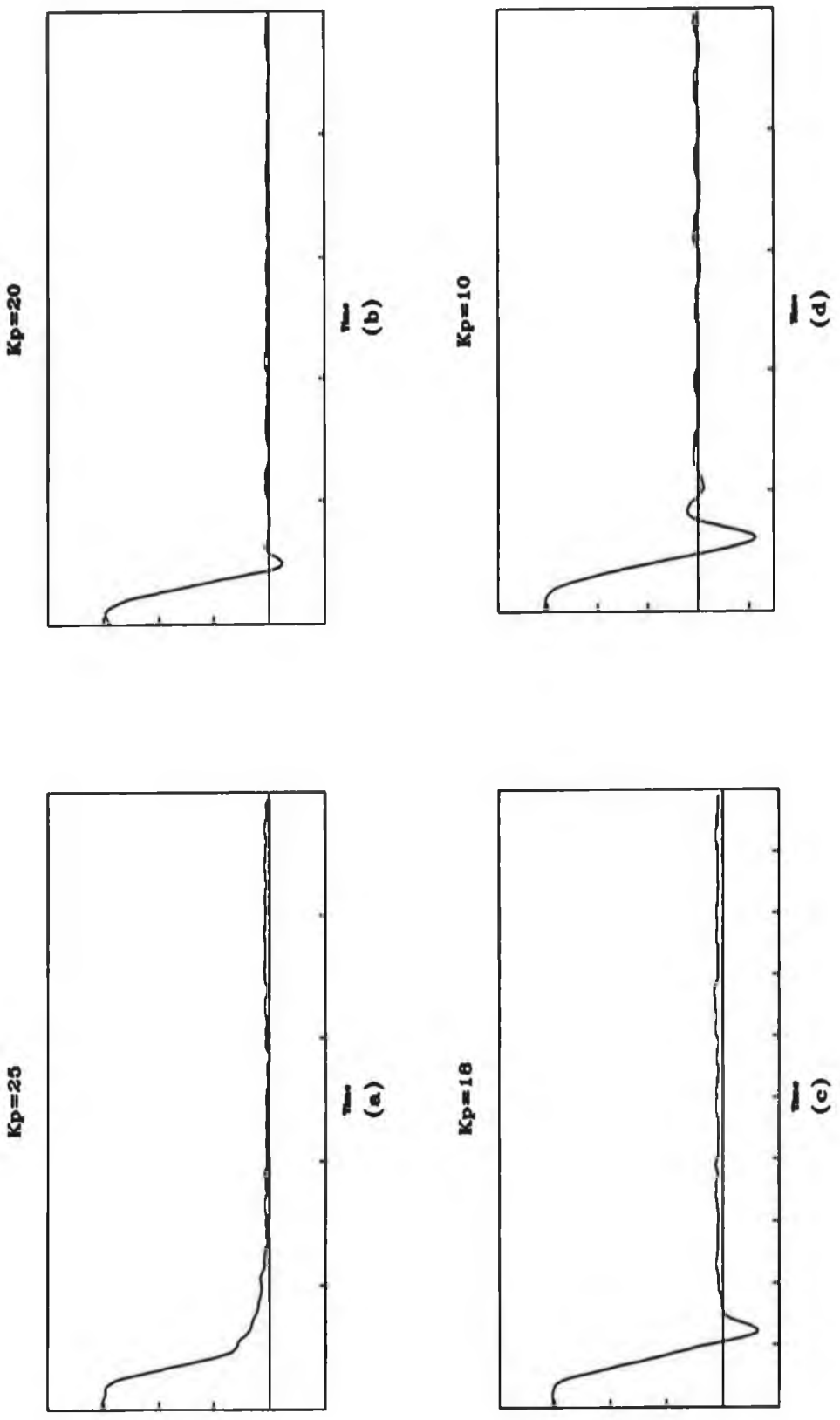


Figure 7-4 The sensitivity to K_P

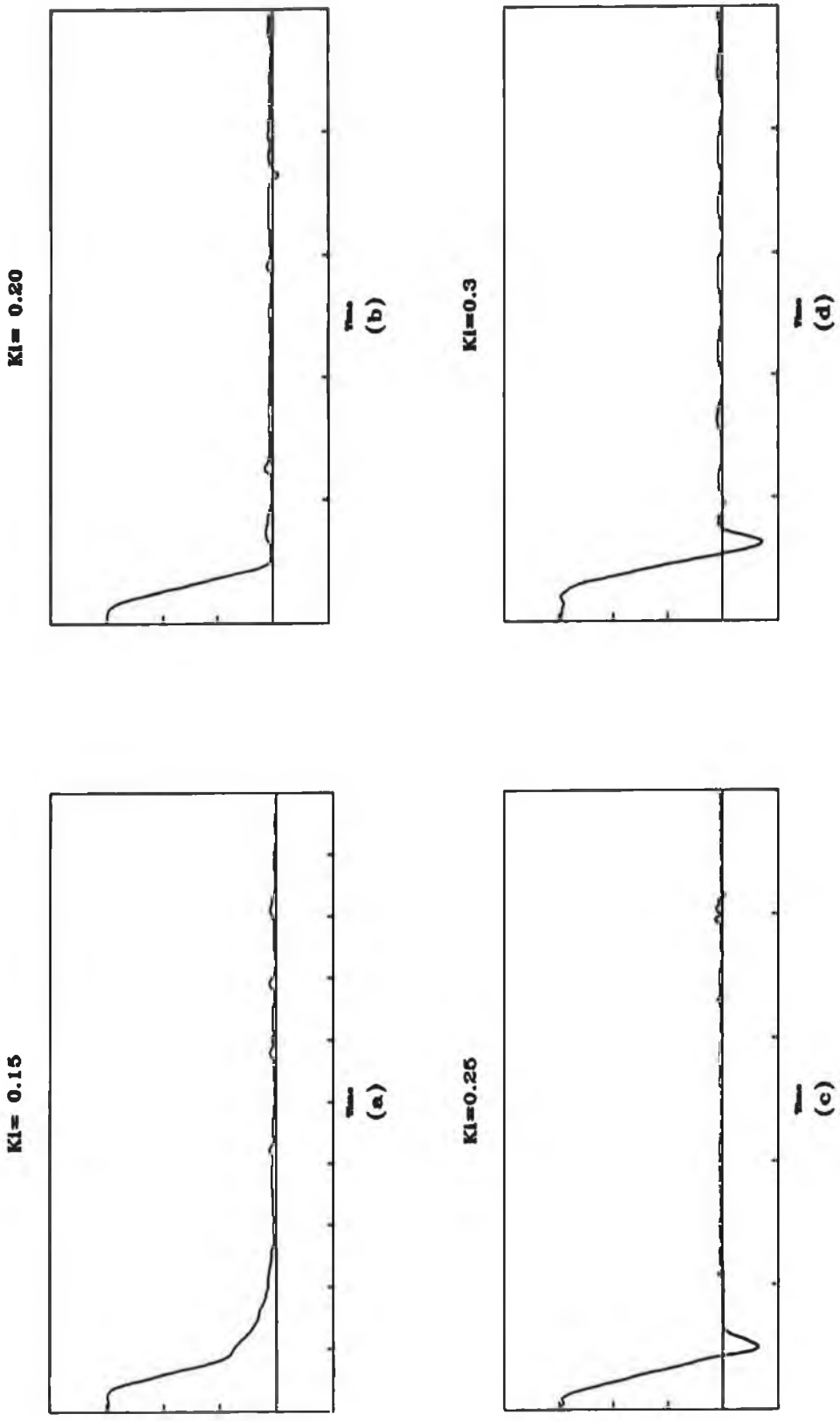


Figure 7-5 The sensitivity to K_i

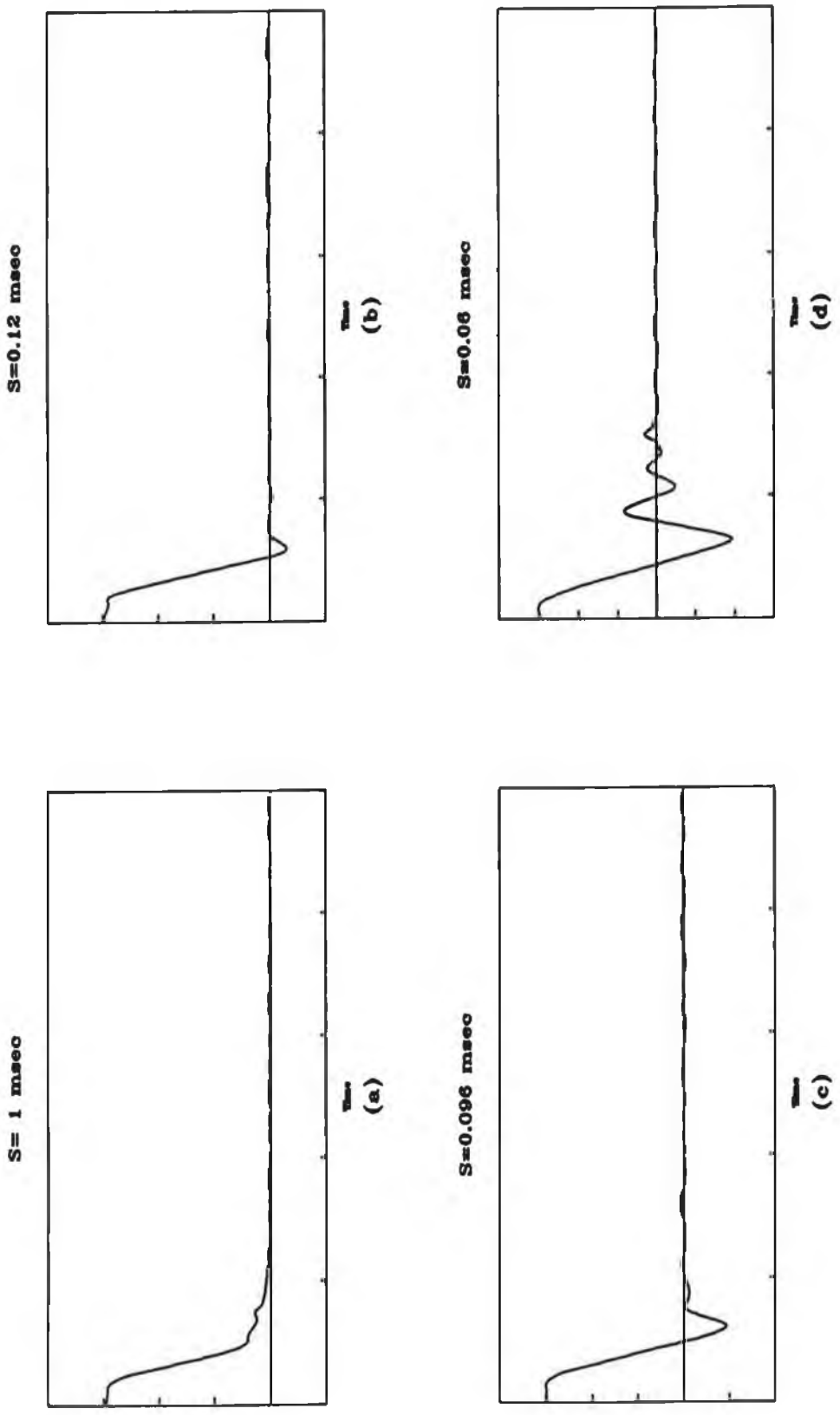


Figure 7-6 The sensitivity to the sample rate

CHAPTER 8

CONCLUSIONS AND RECOMMENDATIONS

8.1 Conclusions

This thesis outlines a design methodology for digital speed servosystems and their thermal protection control using a Texas Instruments TMS320C30 microprocessor. The design methodology applied throughout this thesis is based on following phases:

- System description
- Plant modelling
- Controller design and analysis
- System simulation
- Brushless motor thermal protection control
- TMS320C30 implementation
- Experimental test

A permanent magnet brushless DC servo system was generally described in Chapter 2, it is consist of a brushless motor, an electronic commutator and control system and a sensing system. The control type is a current-controlled brushless DC motor servo system with a PWM operation. The servo system model is based on a commercial PM brushless drive.

The research focused on the digital speed controller design for a brushless drive. Two methods were used, an "analogue design method for a digital controller" and a "grapho-analytical method of pole-placement". The first one used well-known analogue design technology to design an analogue speed controller which is then transformed to a digital

form. For the existing analogue system, using this scheme, the design becomes very simple and easy. The second one used a digital forward and feedback PID controller, in which parameters were adjusted using a graph-analytical method of pole-placement. This controller has a faster response and greater accuracy than the first traditional PI implementation but is more complex. This scheme eliminated the steady-state error of the system and the controller parameters were adjusted for the desired accuracy, response speed and stability margin of the system.

A simulation scheme, written in C, was developed for the designed digital servo system. For the simulation scheme, the dynamic performance of the system can be investigated and examined simply using the flowchart of the system transfer functions. By comparing the results of the simulation and the experimentation, we can see the scheme accurately demonstrated the performance of the system.

Thermal protection of PM brushless motors is a key problem in motor servo drives used in robots and machine tools. In this thesis, a real-time thermal protection control scheme was presented for the PM brushless servo motor. A thermal model of the motor was established, and the thermal controller used this to predict the temperature of the motor windings. Once the predicted temperature reaches the winding insulation limit, the thermal controller limits the motor current to maintain operation within a maximum allowable speed-torque region. This keeps the winding temperature below the insulation limit, while maximizing the motor power output. The simulation results showed a precise thermal protection control.

The digital speed controller was implemented with a digital signal processor, the TMS320C30. The control software program was developed to perform a fast and high precision speed control for the brushless servo drive.

The TMS320C30 based adjustable speed system was tested on a AEG BHT 2214M brushless servo motor. A series of measurement results showed a successful design for the system.

In summary, this research realizes variable speed control of the brushless servo drive using a digital signal processor. It develops a thermal protection scheme for the DSP based servo system. This makes the brushless servo drive system more flexible and more powerful than the previous analogue system.

8.2 Recommendations

The present DSP-based brushless servo system developed in the thesis may be used for some of the further developments mentioned below:

8.2.1 Sinusoidal Type of Brushless AC Motor

In order to obtain a smoother operation, especially in low speed, the brushless drive can use a sinusoidal type control. The motor is called a self-controlled permanent magnet synchronous motor as shown in figure 9-1.

8.2.2 All-digital control

The controller for the brushless servo drive may be developed further into an entirely digital control with the TMS320C30, including speed loop, current loops and sensor signal converter. The overall integrated module would be more efficient, more reliable and certainly more flexible.

8.2.3 More Advanced Control Algorithms

The control algorithms may be developed further by using adaptive control strategy, including self-tuning control, model reference adaptive control, or variable structure control.

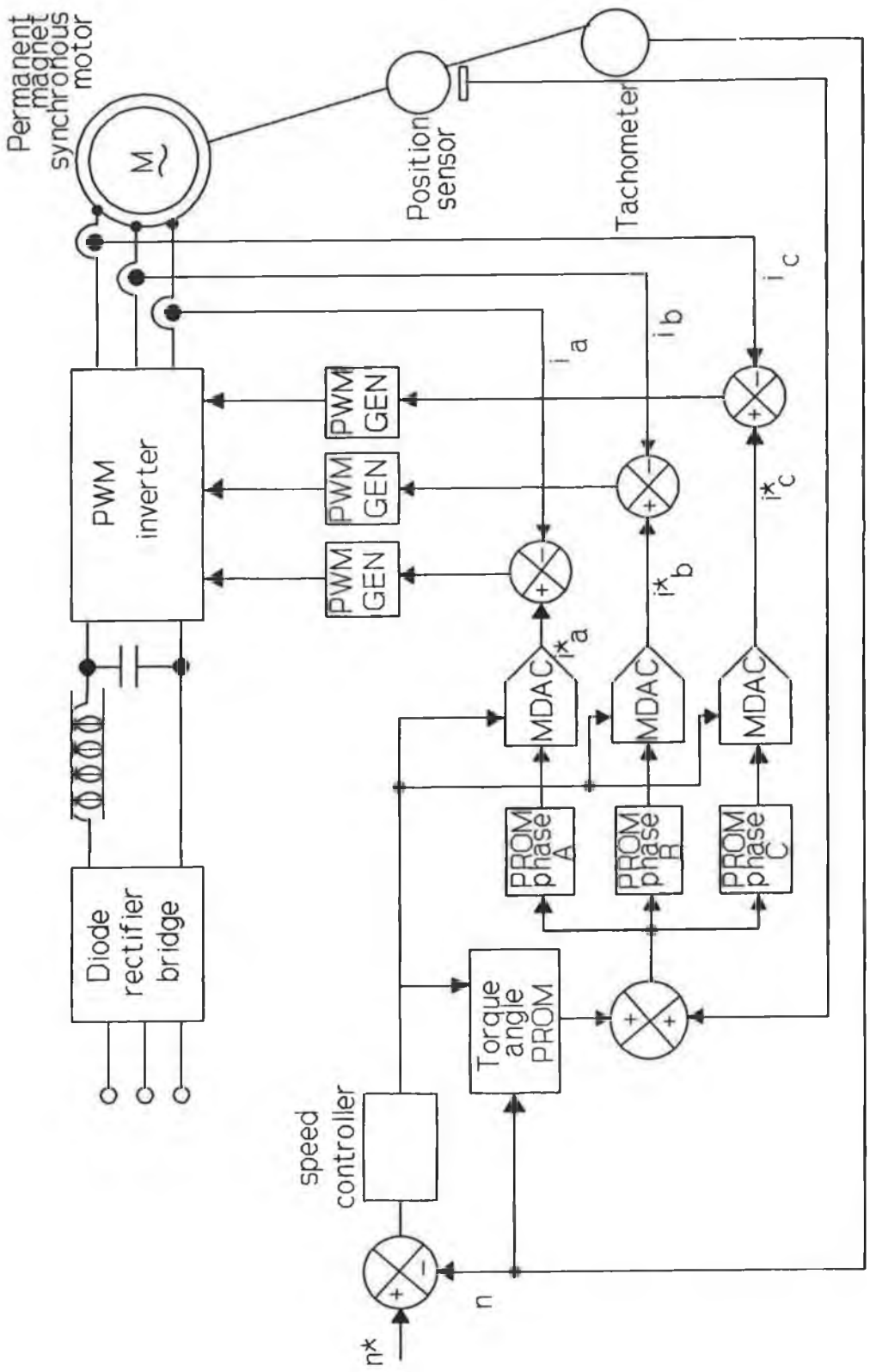


Figure 9-1 Permanent magnet synchronous motor

References

CHAPTER 1:

1-1. M.A.El-Sharkawi, "Development and implementation of high performance variable tracking control for brushless motors",IEEE Transactions on Energy Conversion, Vol.6 No.1, Mar. 1991. pp 114-119

1-2. Edward C. Lee, "Brushless D.C. A modern approach to variable speed drives", IEEE 1990 Annual Textile, Fiber and Film Industry Technical Conference. pp 3/1-5.

1-3. Louis-A. Dessaint, Bernard Hebert and Hoang Le-Huy, "An adaptive controller for a smooth positioning system: analysis and simulation", Conference Record of the 1988 Industry Applications Society Annual Meeting, pp. 562-5 vol.1

1-4. M.F. Brosnan and B.Broon, "Closed loop speed control using an A.C. synchronous motor", IEE proc-B Power Electronics, 1990. pp 373-376

1-5. "AEG AC Servomotors Specifications for the Inland BHT 22XX", AEG Inland, 1989.

1-6. A.M.Zikic, "Practical digital control", Halsted Press, 1989.

1-7. M.R. Stojic, "Design of microprocessor-based digital system for DC motor speed control", IEEE Transaction on Industrial Electronics, Vol. IE-31, no.3, August 1984. pp 243-248

chapter 2:

2-1. J.M.D. Murphy and F.G. Turnbull, "power electronic control of AC motors", Wheaton & Co.,1988.

2-2. P.M. Pelozewshi and U.H. Kunz, "The optimal control of a constrained drive system with brushless dc motor", IEEE Transaction on Industrial Electronics, Vol. 37.No.5, Oct. 1990. pp 342-348

2-3. B.K. Bose,"Adjustable speed AC drive systems", IEEE, New York, NY,1981.

2-4. E.K. Persson and S.Meshket, "Brushless servo system with expanded torque-speed operating range", Proceeding of Motor-Control Conference, Hannovor, 1985. pp 29-37

2-5. A.C. Stone and M.G. Buckley, "Novel design and control of a trapezoidal back emf motor: The smooth transition from brush to brushless dc", Proceeding of Motor-Contral Conference, Hannover, 1985. pp 86-95

CHAPTER 3:

3-1. "AEG AC Servomotors Specifications for the Inland BHT 22XX", AEG Inland, 1989

3-2. A.M.Zikic, "Practical digital control", Halsted Press, 1989.

3-3. P.E. Papamichalis "Digital signal processing application with the TMS320 family, Volume 2", Prentice Hall, 1991.

3-4. Benjamin C. Kuo,"Digital control systems", Holt, Rinehart and Winston, 1980.

3-5. M.R.Stojic, "Design of microprocessor-based digital system for DC motor speed

control", IEEE Transaction on Industry Electronics, Vol. IE-31 No.3, August 1984. pp 243-248

3-6. Millic R.Stojic, "Microprocessor-based control system",Reidel Publishing Company, N.K.Sinha, 1986.

3-7. Jiabing Lu "A new adjustable speed system for application on ships", Proceeding of IMECE'91, 1991. pp101-108

3-8. Baishi Chen, "Automatic control systems", Jiao Tong Press, China, 1981.

3-9. Guangzhou Chen, "The principles of optimal controller", Journal of Electric Drive, China, No.4,1973. pp 305-318

3-10. Hugh F. Vanlandingham, "Introduction to digital control system", MacMillan Publishing Company, 1985.

3-11. KATSUHIKO OGATA " Discrete-time Control System", Prentice-Hall International Inc., 1987.

CHAPTER 4:

4-1. Frank H.S. & Walter L.G. "A guide to using CSMP-The continuous system modelling program." Prentice-Hall, 1976.

4-2. William H.P. & Brian P.F. "Numerical recipes: The art of scientific computing in C." Cambridge university press, 1989.

4-3. URI M. ASCHER & ROBERT M.M. MATTHEIJ & ROBERT D. RUSSELL " Numerical solution of boundary value problems for ordinary differential equations", PRENTICE-HALL Inc, 1988.

4-4. P.D Evans & D Brown "Simulation of brushless DC drives". IEE Proceeding-B, Vol 137, HB No.5, Sep. 1990. pp299-308

CHAPTER 5:

5-1. JMD MURPHY, FG TURNBULL, "Power Electronic Control of AC Motors", Pergamon press, 1988.

5-2. P.C SEN, "Principles of Electric Machines and Power Electronics", John Wiley & Sons, 1988.

5-3. M.G SAY, "Alternating Current Machines",Pitman press, 1976.

5-4. M.G. SAY, E.O TAYLAY, "Direct Current Machines", Pitman press, 1980.

5-5. "BHT 300 Installation and Service Manual", AEG Inland, 1989

5-6. "AEG AC Servomotors Specifications for the Inland BHT 22XX", AEG Inland, 1989.

5-7. "2430A Digital Oscilloscope Operators Manual", Tektronix Inc., 1989.

5-8. P.H MELLOR, D.R TURNER, D. ROBERTS, "Microprocessor based induction thermal protection", Proceeding of 2nd International Conference EMDA, IEE, London, 1985. pp 16-20

5-9. P.H MELLOR, D.R TURNER, "Real time prediction of temperatures in an induction motor using a microprocessor", Electrical Machine Power System, 1988, 13, pp. 333-352.

5-10. P.H MELLOR, D. ROBERTS, D.R TURNER, "Lumped parameter thermal model for electrical machines of TEFC design", IEE Proceedings-B, Vol.138,

No.5, September 1991. pp205-218

5-11. P.G.A WILSON, "Complete protection of motors under variable load conditions", Proceeding of Drive/motors/controls, 1983. pp. 172-178

CHAPTER 6:

6-1. Third-Generation TMS320C30 User's Guide, Texas Instruments, 1988.

6-2. TMS320C30 Assembly Language Tools User's Guide, Texas Instruments, 1988.

6-3. TMS320C30 C Compiler Reference Guide, Texas Instruments, 1990.

6-4. Panos E. Papamichalis, Digital Signal Processing Applications (volume 2), Prentice Hall, 1991.

6-5. Microsoft Quick C Compiler programmer's Guide, Microsoft 1990.

6-6. BHT 300 Installation and Service Manual, AEG Inand, 1989.

6-7. Brian W.Kernighan, The C Programming Language, Prentice-Hall, Inc., 1987.

6-8. Panos E. Papamichalis, The TMS320C30 Floating-Point Digital Signal Processor, IEEE proceeding of MICRO, DEC. 1988. pp13-29

APPENDIX A

The servo system model

The brushless servo system used in this project consists of an AEG BHT 2214 brushless motor and an BHT-300 electronic control system.

The specifications of the brushless driver system are:

Motor:

P=2900 W (rated power);

ω_M=5000 RPM (maximum operating speed);

M_0=11.3 Nm (continuous torque at 40°C ambient);

M_p=32.1 Nm (peak torque);

I_0=21.1 A (rated current at continuous torque);

I_p=63 A (rated current at peak torque);

V_T=300 V (maximum terminal voltage);

K_T=0.536 Nm/A (torque sensitivity);

K_B=56 V/KRPM (back EMF constant);

R_M=0.30 Ω (DC resistance at 25°C);

L_M=2.5 mh (inductance);

τ_{em}=0.001 s (mechanical time constant);

τ_l=0.0083 s (electrical time constant);

J=0.00098 kg•m^2 (rotor inertia);

F=0.04 Nm/KRPM (viscous damping ∞ Z source

=0.04/[(1000•2π)/60]=0.000382 Nm•s

τ_m=J/F=0.536/0.000382=2.57 s

Sensor:

α=0.0435 (10V/230V, speed feedback constant);

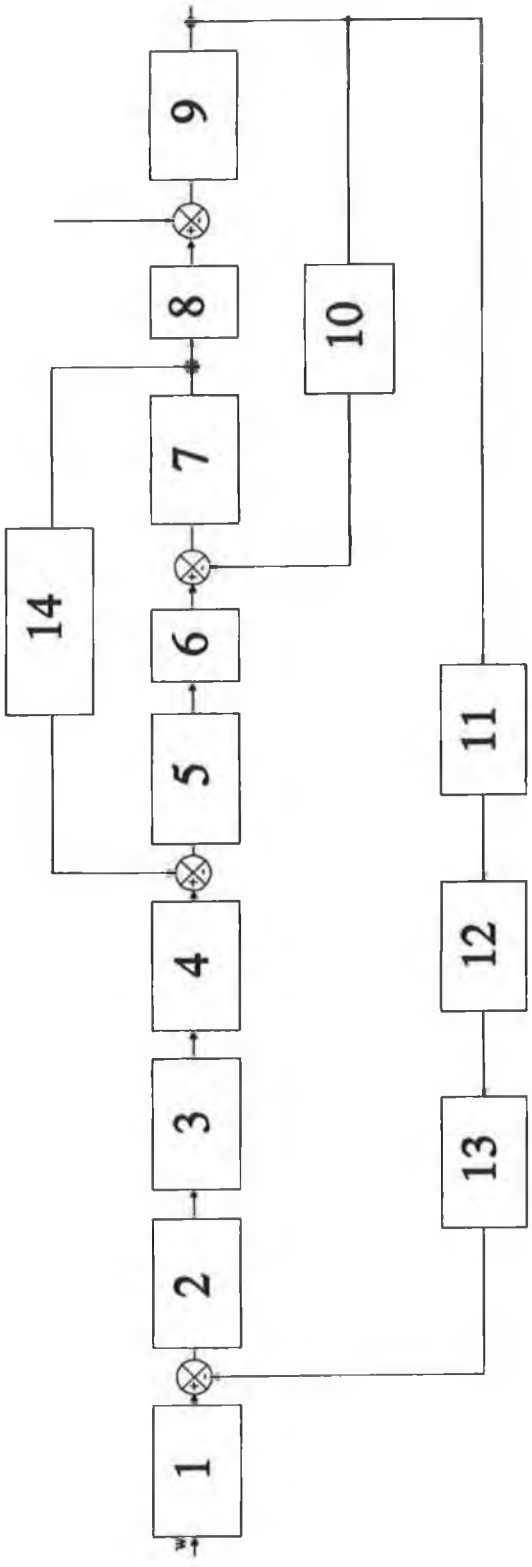


Fig. A-2 The block diagram of adjustable speed servo systems

T_e=0.0003 s (electrical time constant);

Controller:

Resistances:

R3=6.8K; R4=62K; R9=0;

R11=100K R12=100K; R13=100K;

R14=12K

Capacitors:

C1=0.01μF; C2=0.01μF; C3=0.1μF;

C4=0.15μF; C5=0.001μF; C7=0.22μF;

C9=0.022μF;

The circuit diagram of controller is shown in the figure D1. The whole servo system can be divided into 14 blocks as show in Fig A-2. The modelling analysis of each block is described as follows.

A.1 BLOCK 1

Block 1 is a differential input amplifier {A} in figure A-1, which is used to reduce common mode/system ground noise. The circuit of block 1 is shown in figure A-3.

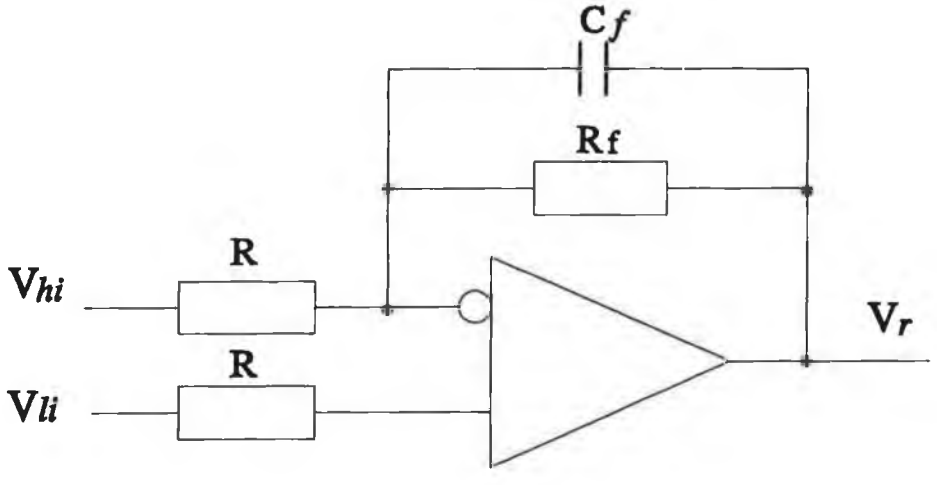


Figure A-3 The circuit of BLOCK 1

The transfer function of block 1 is expressed as:

$$\frac{V_r}{\frac{R_f}{R_f C_f s+1}} = \frac{V_{hi} - V_{li}}{R} \qquad (A.1)$$

where V_{hi} is the command of the high speed input.

V_{li} is the command of the low speed input.

This equation can be simplified to:

$$V_r(s) = K_p \frac{1}{T_a s+1} [V_{hi} - V_{li}] \qquad (A.2)$$

where $K_p = R_f/R$; $T_a = C_f \bullet R_f$

A.2 BLOCK 2 and BLOCK 14

Block 2 and block 14 combine a velocity loop in which block 2 is a PI modulator denotes {C} amplifier and block 14 is a lead modulator in the speed feedback path as shown in the controller circuit diagram, figure A-1. The speed loop compares a REQUIRED speed (input voltage) with an ACTUAL speed. The difference or "error" produces a REQUIRED current signal to current loop. The circuit diagram of block 2 and 14 is shown in figure A-3. The transfer function of the circuit can be expressed as:

$$V_n = K_n \left(\frac{T_n s+1}{T_n s}\right) \left[V_r - V_{f2}\left(1+\frac{T_{no} s}{T_{nf} s+1}\right)\right] \qquad (A.3)$$

where R=10K; $T_{no} = RC_6$; $T_{nf} = R_{10}C_6$; and $K_n = R_{13}/R$; $T_n = R_{13}C_4$;

A.3 BLOCK 3 and BLOCK 4

The block 3 and block 4 are the tack filters {D1, D2} in figure A-1, which is used to eliminate torsional vibrations. Both transfer functions of operational circuits "D1" and

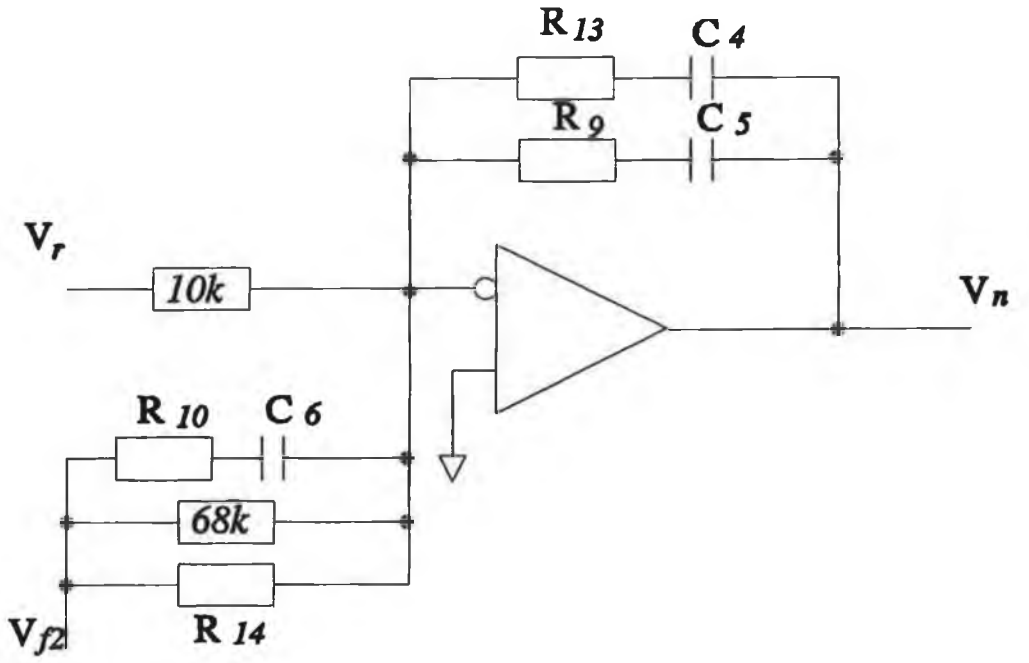


Figure A-4 The circuit diagram of block 2 and block 14

"D2" are expressed as:

$$\frac{V_n'}{V_n} = \frac{20K}{20K} \cdot \frac{1}{20K \cdot C1 \cdot s+1} = \frac{1}{20K \cdot C1 \cdot s+1} \qquad (A.4)$$

$$\frac{V_i}{V_n'} = \frac{20K}{20K} \cdot \frac{1}{20K \cdot C1 \cdot s+1} = \frac{1}{20K \cdot C3 \cdot s+1} \qquad (A.5)$$

where C1=0.01μF;

C3=0.1μF;

T1=20K$\bullet$C1=0.0002 (in block 3) and

T3=20K$\bullet$C3=0.002 (in block 4).

A.4 BLOCK 5 and BLOCK 11

Block 5 and block 11 represent the current loop amplifier {E} in figure A-1 . The circuit diagram is shown in figure A-4.

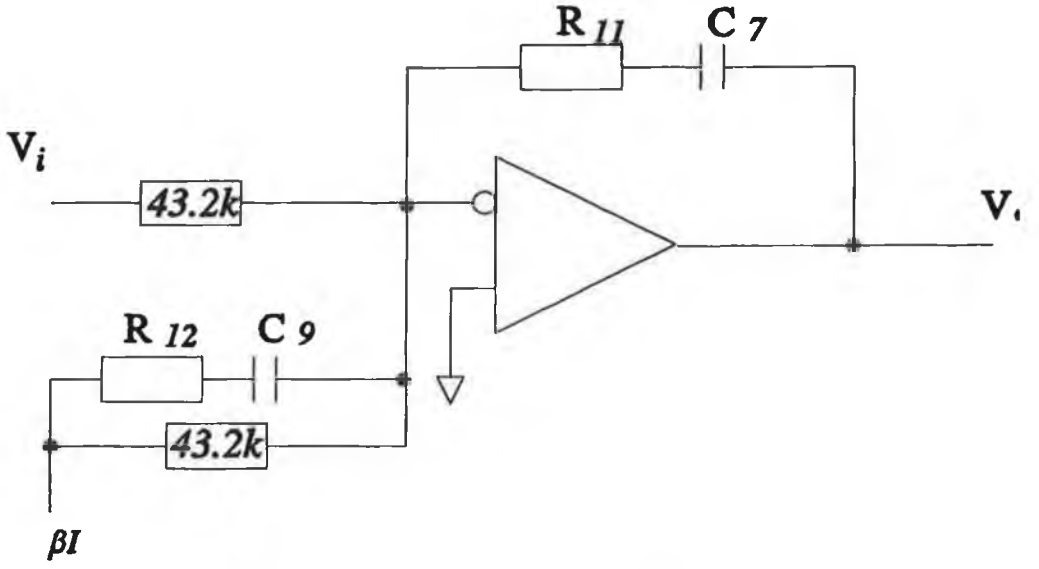


Fig. A-4 The operational amplifier circuit of BLOCK 5 and BLOCK 11

The REQUIRED current V_i from the speed loop compares with the "actual" current signal βI from current sense circuit, where β is the sensitivity. The current loop is used to control the PWM GENERATOR and BASE DRIVE circuit. The larger the current required, the greater the "on" time of the output devices. The transfer function of this current loop is expressed as:

$$\frac{V_0}{R11+\frac{1}{C7\ s}} = \frac{V_I}{43.2K}-\beta I_a\left(\frac{1}{43.2K}+\frac{1}{R12+\frac{1}{C9\ s}}\right) \qquad (A.6)$$

where R_{11}=100K, R_{12}=100K, C_7=0.022μ, C_9=0.022μ and V_o is the output of the current of the current loop, which can be simplified to:

$$\begin{aligned} V_I &= \frac{R11}{43.2K}\cdot\frac{R11\ C7\ s+1}{R11\ C7\ s}\left[V_I-\beta I_a\left(1+\frac{43.2K\ C9\ s}{R12\ C9\ s+1}\right)\right] \\ &= K_I\frac{\tau_i s+1}{\tau_i s}\left[V_I-\beta_a\left(1+\frac{\tau_0 s}{\tau_i+1}\right)\right] \end{aligned} \qquad (A.7)$$

where K_I=100K/43.2K=2.315;

τ_i=R11•C7=0.0022;

τ_0=43.2K•C9=0.00095.

The coefficient β needs to be calculated. For BHT 2214 motor, the current sensitivity is module D (10 A/V) [1]. In the torque-speed characteristic diagram, the maximum torque is 32.1 Nm, so the maximum current which is $T/K_T=32.1/0.536\doteq59.9$ A. The maximum voltage of the controller is 10 V. Therefore ,

$\beta=10$ V/(59.9/10)$\doteq1.67$

A.5 The transfer function of BLOCK 6

Block 6 represents the PWM generator and invertor. They can be approximately modeled to a gain. The input of the PWM is +/- 10V and DC output of the inverter is +/- 300V. So the gain is: $K_A = 300/10 = 30$

A.6 BLOCK 7. Block 8. Block 9 and Block 10

The brushless DC motor can be expressed as:

$$V_A = L\frac{dI_A}{dt} + R \cdot I_A + V_B \qquad (A.8)$$

$$V_B = K_B \cdot \omega \qquad (A.9)$$

$$T_A = K_T \cdot I_A \qquad (A.10)$$

$$T_A = J\frac{d\omega}{dt} + F \cdot \omega + T_L \qquad (A.11)$$

$$\frac{I_A(s)}{V_A(s) - V_B(s)} = \frac{\frac{1}{R}}{\tau_e \cdot s + 1} \qquad (A.12)$$

From equation (A.8), the transfer function of block 7 can be modeled to equation (A.12): where τ_e=L/R (electrical time constant).

From equation (A.11), the transfer function of block 8 can be modeled to:

$$\frac{T_a(s)}{I_a(s)} = K_t \qquad \text{(A.13)}$$

From equation (A.10), the transfer function of block 9 can be derived as:

$$\frac{\omega(s)}{T_a(s)} = \frac{1}{J \cdot s+1} \qquad \text{(A.14)}$$

And from equation (A.9) the transfer function of block 10 is:

$$\frac{V_b(s)}{\omega(s)} = K_b \qquad \text{(A.15)}$$

A.7 BLOCK 12

Block 12 represents the tachogenerator. the transfer function can be expressed as:

$$\frac{V_P(s)}{\omega(s)} = \frac{\alpha}{\tau_f s+1} \qquad \text{(A.16)}$$

where τ_f is the sensor's electrical time constant, and

α is the gain of the sensor.

A.8 BLOCK 13

Block 13 represents the filter of the tachogenerator. The transfer function is expressed as:

$$\frac{V_{P2}(s)}{V_{P1}(s)} = \frac{1}{T_2 s+1} \qquad \text{(A.17)}$$

where $C_2=0.01\mu F$, $T_2=10K \bullet C_2=0.0001$.

A.9 The whole system model

Through out the analysis of each block, we can obtain the block diagram of the total servo system model as shown in figure A-5, which is used to design the speed controller.

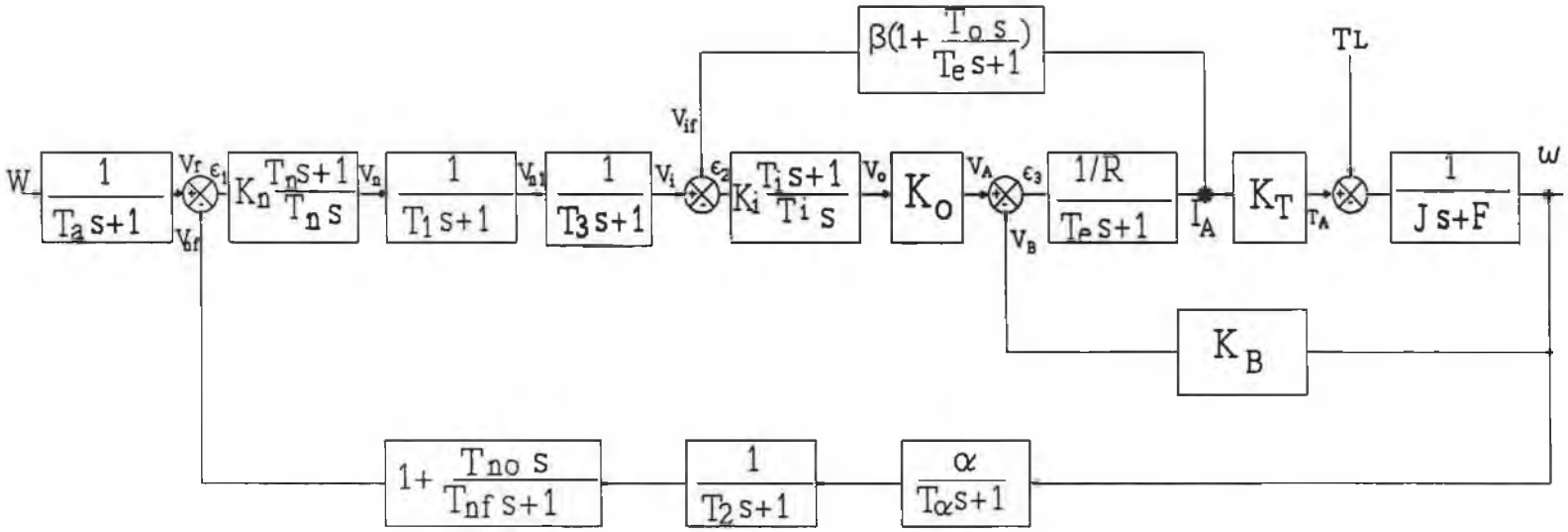
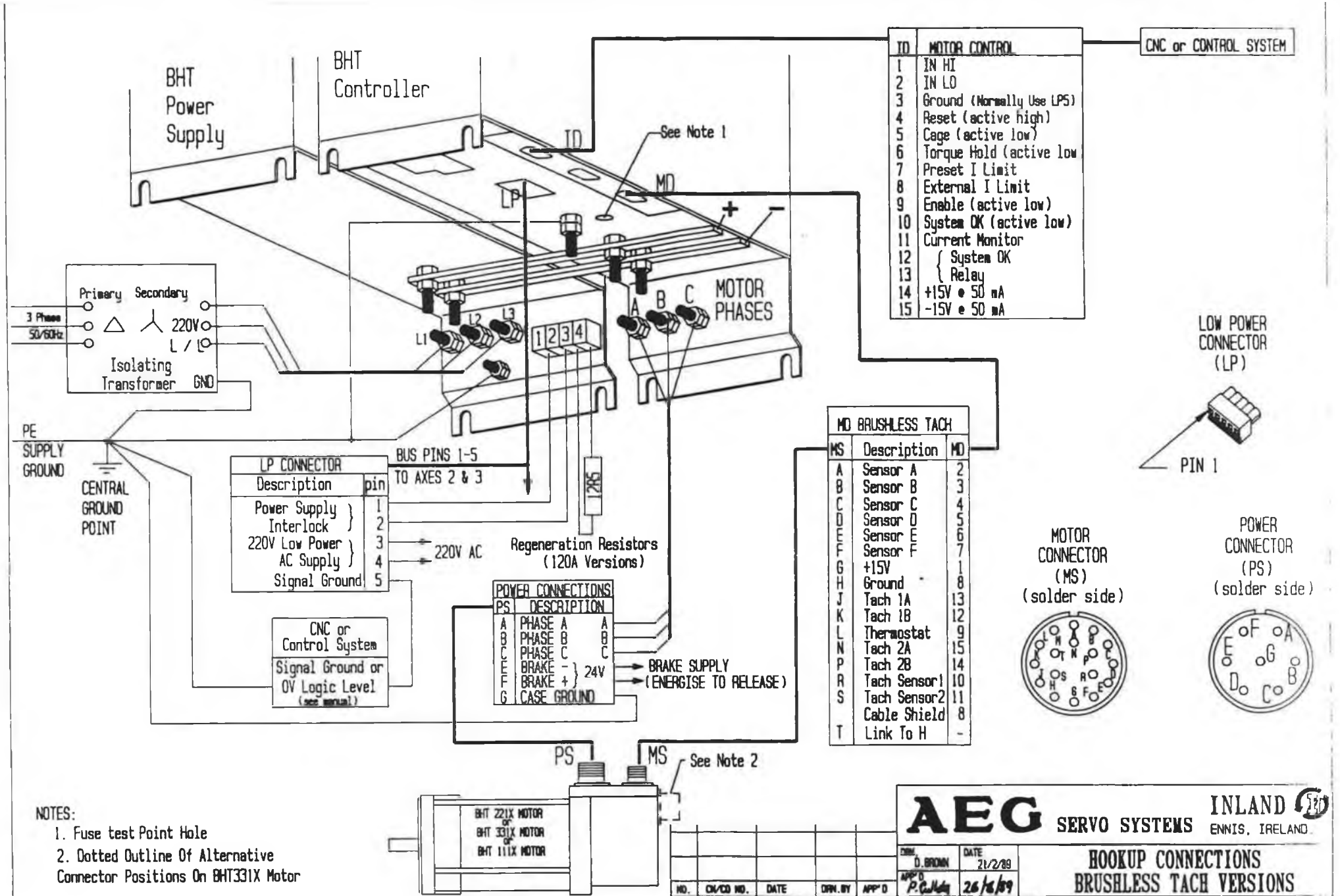


Figure A-5 Block diagram of the speed servo system



NOTES:

1. Fuse test Point Hole
2. Dotted Outline Of Alternative Connector Positions On BHT331X Motor

NO.	CN/CO NO.	DATE	DRN. BY	APP'D

TOLERANCES: I DEC. PLACES +/- 0.3mm; II DEC. PLACES +/- 0.1mm; ANG. DIMENSION +/- 1 DEGREE

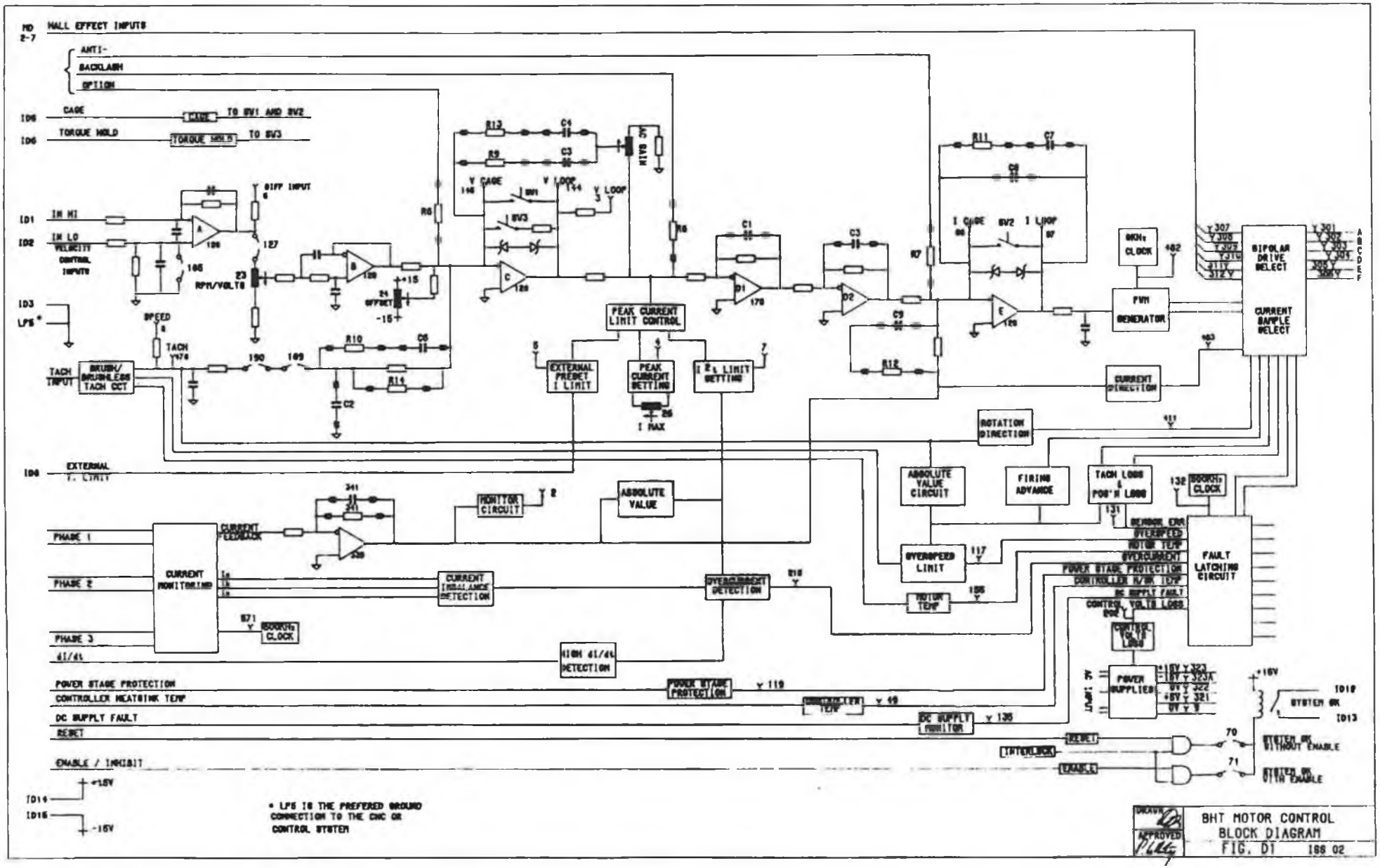
SCALE

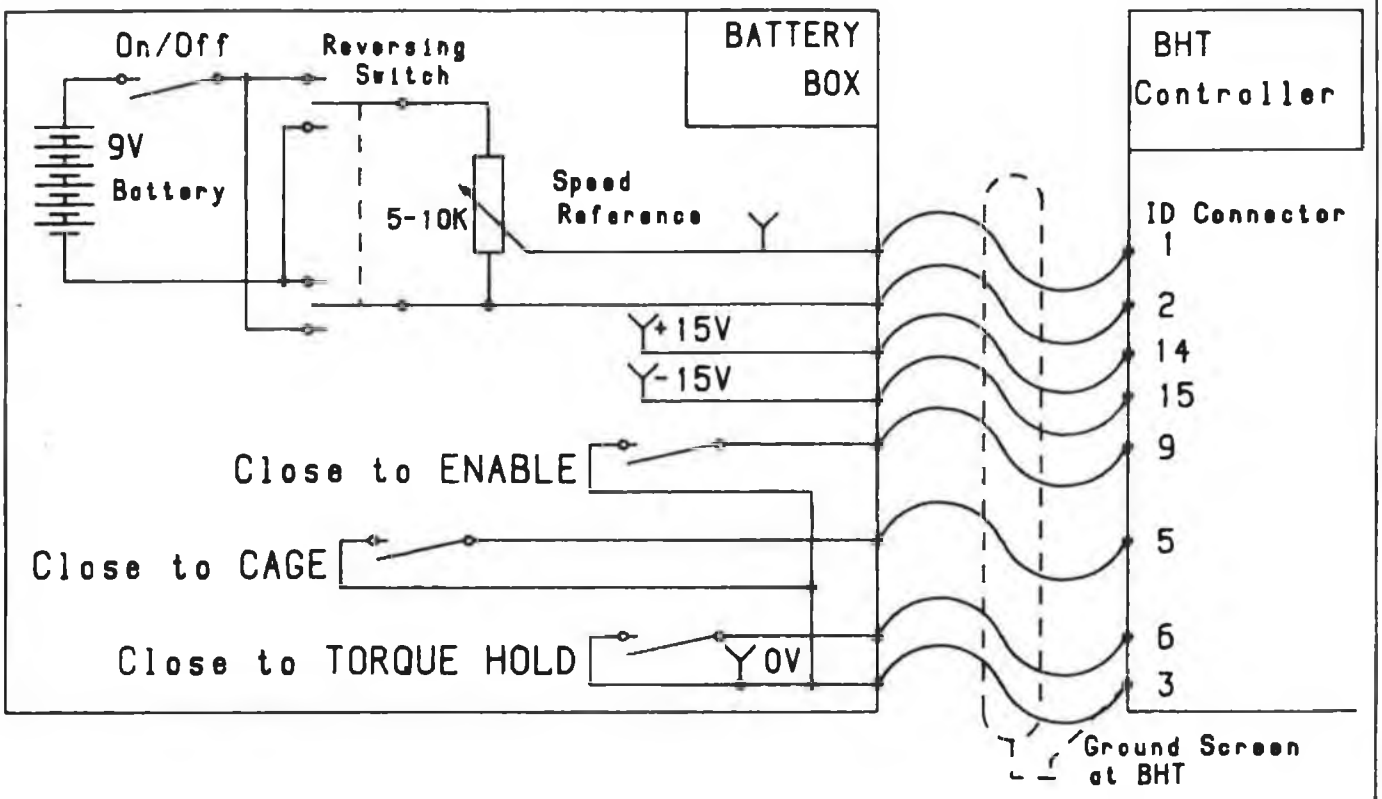
AEG SERVO SYSTEMS INLAND ENNIS, IRELAND

DRN.	DATE
D. BROWN	21/2/89
APP'D P. Cahill	26/2/89

HOOKUP CONNECTIONS BRUSHLESS TACH VERSIONS

DRG. NO. FIG. C2 ISS.03


HALL EFFECT INPUTS
ANTI-BACKLASH OPTION
CAGE
TORQUE HOLD
TO BV1 AND BV2
TO BV3
IN HI
IN LO
VELOCITY CONTROL INPUTS
DIFF INPUT
RPM/VOLTS
OFFSET
+15
-15
SPEED
TACH
TACH INPUT
BRUSH/ BRUSHLESS TACH OCT
V CAGE
V LOOP
I CAGE
I LOOP
AC GAIN
PEAK CURRENT LIMIT CONTROL
EXTERNAL PREDET I LIMIT
PEAK CURRENT SETTING
I MAX
I²t LIMIT SETTING
EXTERNAL I LIMIT
PWM GENERATOR
8KHz CLOCK
BIPOLAR DRIVE SELECT
CURRENT SAMPLE SELECT
CURRENT DIRECTION
ROTATION DIRECTION
ABSOLUTE VALUE CIRCUIT
FIRING ADVANCE
TACH LOSS & POS'N LOSS
500KHz CLOCK
FAULT LATCHING CIRCUIT
OVERSPEED LIMIT
MOTOR TEMP
OVERCURRENT
POWER STAGE PROTECTION
CONTROLLER H/SK TEMP
DC SUPPLY FAULT
CONTROL VOLTS LOSS
RESOLVER ERR
MONITOR CIRCUIT
ABSOLUTE VALUE
CURRENT FEEDBACK
PHASE 1
PHASE 2
PHASE 3
CURRENT MONITORING
CURRENT IMBALANCE DETECTION
OVERCURRENT DETECTION
HIGH dI/dt DETECTION
dI/dt
POWER STAGE PROTECTION
CONTROLLER HEATSINK TEMP
DC SUPPLY FAULT
RESET
ENABLE / INHIBIT
POWER SUPPLIES
SYSTEM OK
SYSTEM OK WITHOUT ENABLE
SYSTEM OK WITH ENABLE
INTERLOCK
ENABLE
+15V
-15V
* LP5 IS THE PREFERED GROUND CONNECTION TO THE CNC OR CONTROL SYSTEM
DRAWN
APPROVED
BHT MOTOR CONTROL BLOCK DIAGRAM
FIG. D1 ISS 02



Portable Battery Box - BHT Connections

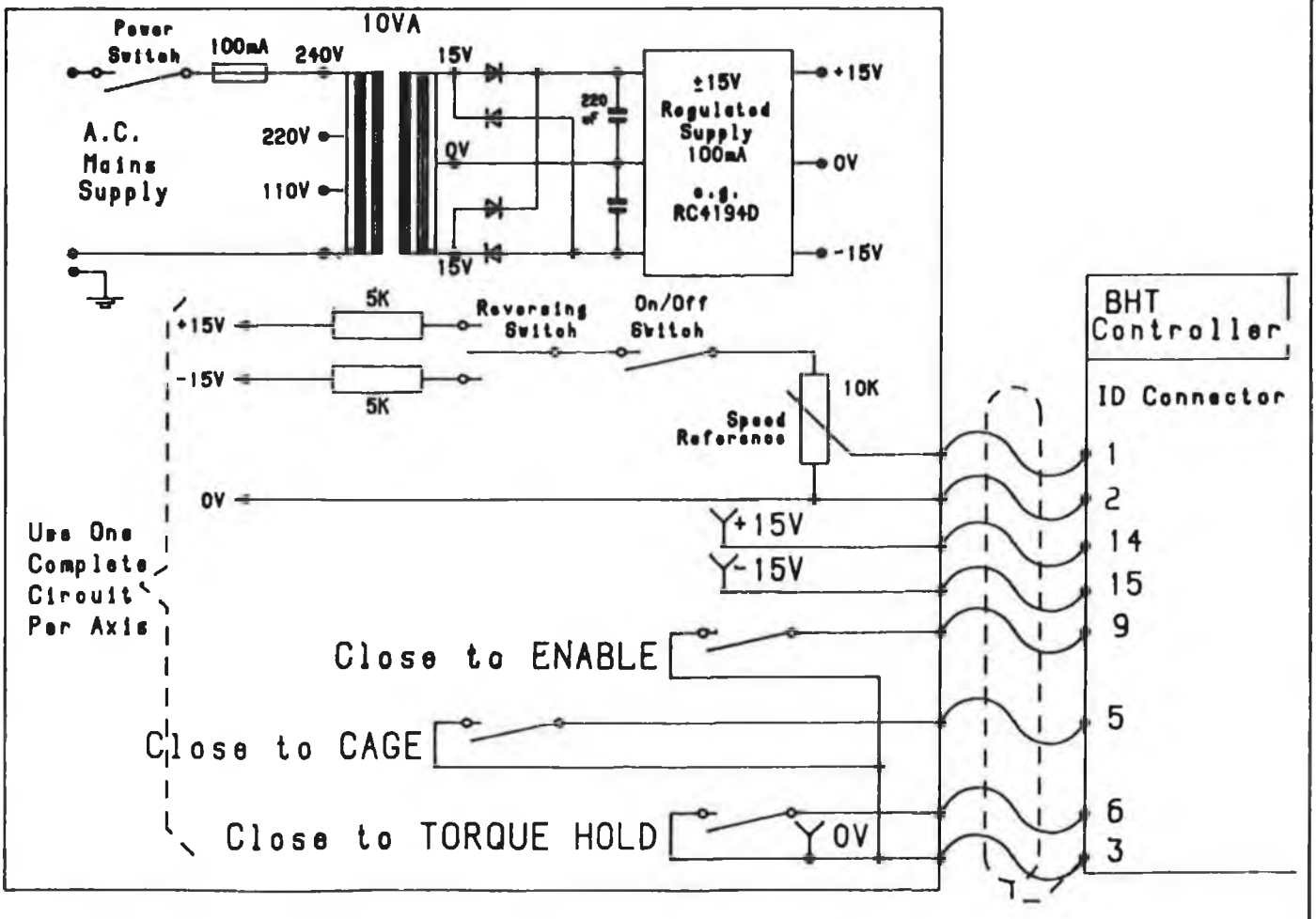


Figure D4 Mains Powered Battery Box - BHT Connections

APPENDIX B

B-I. The simulation program for DESIGN I

```
/*************************************************************
*       Simulation program for disign I speed controller      *
*                                                             *
*       Author: Jiabing Lu                                    *
*************************************************************/


#include <stdlib.h>
#include <stdio.h>
#include <math.h>

#define KT    0.536   /* V/rpm */
#define KB    0.056   /* Nm/A  */
#define N     10

float  w,ki,kp,kd;
int    n;

char    name[10];

main()

{
      float     x[18]={0.0};
      float     error[5]={0.0};
      float     xin[12]={0.0},tran[12]={0.0};
      float     h=0.0001;
      float     time=0.0, output=0.0, current=0.0;
      int       j,i,m,l;

/*parameter of digital controller*/

      float     u1=0.0,er1=0.0,u=0.0,er=0.0,in=0.0, w1=0.0;

      FILE      *dat;
```

```
      inputname();

/*numbers of output points */

      m=40;
      dat=fopen(name,"w");
      for(j=0;j<m;j++)
      {
        fprintf(dat,"\n%f  %f  %f  %f",time,x[9],output,power);
        for(i=0;i<500;i++)
        {



/********** OPTIMIZING DIGITAL CONTROLLER **********/

          er=w1-in;
          u=u1+5.3348*(er-0.999335*er1);
          u1=u;
          er1=er;
          x[2]=u;

/*sample time is T=n*h*/

      for(l=0;l<N;l++)
      {

/******* CONTINUOUS TIME PARTS *******/

          input();

          time+=h;
/*block 4 =x[4]/x[2] = 1/(1+0.002s) filter */

          block(x[2],1.0,0.002,1.0,0.0,h,xin[3],&tran[3],&x[4]);
          xin[3]=tran[3];


/** current loop **/

          error[2]=x[4]-x[16];

/* Ki=2.315 */

          x[5]=2.315*error[2];

/*block 5 = x[6]/x[5] = 0.0022s/(1+0.0022s) */
          block(x[5],0.0,0.0022,1.0,0.0022,h,xin[4],&tran[4],&x[6]);
```

```
        xin[4]=tran[4];

/*block 6 = x[7]\x[6] = ko=60.0*/

        x[7]=60.0*x[6];

        error[3]=x[7]-x[11];

/*block 7 = x[9]/error[3] = 1/(0.3+0.0025s) */

        block(error[3],0.3,0.0025,1.0,0.0,h,xin[5],&tran[5],&x[9]);
        xin[5]=tran[5];

/*block 8 = x[9]/x[10] = KT =0.536 , current to torque*/

        x[10]=KT*x[9];

        error[4]=x[10]-load;

/*block 9 = output/error[4] = (1/F)/(1+J/F s) */

        block(error[4],1.0,24.5,25000.0,0.0,h,xin[6],&tran[6],&output);
        xin[6]=tran[6];

/*block 10 = feedback e.m.f constant = KB = 0.056 (V/rpm) */

        x[11]=KB*output;

/*block 11 = speed feedback constant = 0.001 */

        x[12]=0.001*output;

/** block 12 and 13 is speed feedback filter **/

/*block 12 = speed feedback =1/(1+.0003s) */

        block(x[12],1.0,0.0003,1.0,0.0,h,xin[7],&tran[7],&x[13]);
        xin[7]=tran[7];

/*block 13 =  1/(1+0.001s) */
        block(x[13],1.0,0.001,1.0,0.0,h,xin[8],&tran[8],&x[14]);
        xin[8]=tran[8];

/*  speed feedback to digital controller */

        b2=x[14];

/*bklock 14 = current feedback filter = (1+0.00315s)/(1+0.0022s) ; b=0.1 */
```

```
        x[15]= 0.1*x[9];
        block(x[15],1.0,0.0022,1.0,0.00215,h,xin[9],&tran[9],&x[16]);
        xin[9]=tran[9];
        current=x[9];

      }
      }
     }
     fclose(dat);
}

/*subroutine of function  y(s)/u(s) = (c+ds)/(a+bs)*/
block(u,a,b,c,d,h,xin,tran,y)
float  u,a,b,c,d,h,xin,*tran,*y;

{
       float   m=(-a/b),n=(1/b),z;
       rk4(u,m,n,h,xin,&z);
      *y=(c-a*d/b)*z+(d/b)*u;
      *tran=z;
}

/*subroutine of the fourth-order Runge-Kutta*/
rk4(u,m,n,h,xin,xout)
float  u,m,n,h,xin,*xout;
{
       float   k[4],p[4],xm=0.0;
       int     i,j=0;
       k[0]=0.0;
       p[1]=0.0;
       p[2]=p[3]=h/2.0;
       p[4]=h;
       for(i=1;i<5;i++)
         {
          j=i-1;
          k[i]=m*(xin+p[i]*k[j])+n*u;
         }
        xm=xin+p[4]*(k[1]+2.0*k[2]+2.0*k[3]+k[4])/6.0;
        *xout=xm;
}

inputname()
{
        printf("Type the name of output file:\n");
        scanf("%s",name);
}
```

B-II. The simulation program for DESIGN II

```
/*************************************************************
*       Simulation program for disign II speed controller     *
*                                                             *
*       Author: Jiabing Lu                                    *
*************************************************************/


#include <stdlib.h>
#include <stdio.h>
#include <math.h>

#define KP    18
#define KI    0.1
#define KD    55
#define KT    0.536   /* V/rpm */
#define KB    0.056   /* Nm/A  */
#define N     10

char    name[10];

main()

{
      float     x[18]={0.0};
      float     error[5]={0.0};
      float     xin[12]={0.0},tran[12]={0.0};
      float     h=0.0001;
      float     output=0.0,current=0.0;
      int       j,i,m,l;

/*parameter of digital controller*/

      float     b1=0.0,b2=0.0,u1=0.0,u2=0.0,e=0.0,u=0.0;

      FILE      *dat;

      inputname();

/*numbers of output points */

      m=40;
      dat=fopen(name,"w");
      for(j=0;j<m;j++)
      {
```

```
        fprintf(dat,"\n%f  %f  %f  %f",time,x[9],output,power);
        for(i=0;i<500;i++)
        {


/********** PID DIGITAL REGULATOR **********/

/*e(kT)=e(kT)-b(kT)      */

        e=speed-b2;

/*I*/

/*ul(kT) = KI*e(kT) + ul[(k-1)T]  */

        u2=KI*e+u1;
       u1=u2;

/************************************************************
*P,D                                                        *
*                                                          *
* u(kT) = -KP * b(kT) - KD * {b(kT) - b[(k-1)T]} + ul(kT)   *
************************************************************/
        u=(-KP)*b2-KD*(b2-b1)+u2;
        b1=b2;

/* Limit of maximum current */

        if (u>6.3)
        u=6.3;
        if (u<-6.3)
        u=-6.3;



/* Digital controller output */

        x[2]=u;

/******* continue time system *******/


/*sample time is T=n*h*/

    for(l=0;l<N;l++)
    {
```

```
        input();

        time+=h;
/*block 4 =x[4]/x[2] = 1/(1+0.002s) filter */

        block(x[2],1.0,0.002,1.0,0.0,h,xin[3],&tran[3],&x[4]);
        xin[3]=tran[3];



/** current loop **/

        error[2]=x[4]-x[16];

/* Ki=2.315 */

        x[5]=2.315*error[2];

/*block 5 = x[6]/x[5] = 0.0022s/(1+0.0022s) */

        block(x[5],0.0,0.0022,1.0,0.0022,h,xin[4],&tran[4],&x[6]);
        xin[4]=tran[4];

/*block 6 = x[7]\x[6] = ko=60.0*/

        x[7]=60.0*x[6];

        error[3]=x[7]-x[11];

/*block 7 = x[9]/error[3] = 1/(0.3+0.0025s) */

        block(error[3],0.3,0.0025,1.0,0.0,h,xin[5],&tran[5],&x[9]);
        xin[5]=tran[5];

/*block 8 = x[9]/x[10] = KT =0.536 , current to torque*/

        x[10]=KT*x[9];

        error[4]=x[10]-load;

/*block 9 = output/error[4] = (1/F)/(1+J/F s) */

        block(error[4],1.0,24.5,25000.0,0.0,h,xin[6],&tran[6],&output);
        xin[6]=tran[6];

/*block 10 = feedback e.m.f constant = KB = 0.056 (V/rpm) */

        x[11]=KB*output;
```

```
/*block 11 = speed feedback constant = 0.001 */

        x[12]=0.001*output;

/** block 12 and 13 is speed feedback filter **/

/*block 12 = speed feedback =1/(1+.0003s) */

        block(x[12],1.0,0.0003,1.0,0.0,h,xin[7],&tran[7],&x[13]);
        xin[7]=tran[7];

/*block 13 =  1/(1+0.001s) */
        block(x[13],1.0,0.001,1.0,0.0,h,xin[8],&tran[8],&x[14]);
        xin[8]=tran[8];

/*  speed feedback to digital controller */

        b2=x[14];

/*bklock 14 = current feedback filter = (1+0.00315s)/(1+0.0022s) ; b=0.1 */

        x[15]= 0.1*x[9];
        block(x[15],1.0,0.0022,1.0,0.00215,h,xin[9],&tran[9],&x[16]);
        xin[9]=tran[9];
        current=x[9];

      }
      }
     }
     fclose(dat);
}

/*subroutine of function  y(s)/u(s) = (c+ds)/(a+bs)*/
block(u,a,b,c,d,h,xin,tran,y)
float  u,a,b,c,d,h,xin,*tran,*y;

{
       float   m=(-a/b),n=(1/b),z;
       rk4(u,m,n,h,xin,&z);
      *y=(c-a*d/b)*z+(d/b)*u;
      *tran=z;
}

/*subroutine of the fourth-order Runge-Kutta*/
rk4(u,m,n,h,xin,xout)
float  u,m,n,h,xin,*xout;
{
       float   k[4],p[4],xm=0.0;
```

```
        int     i,j=0;
        k[0]=0.0;
        p[1]=0.0;
        p[2]=p[3]=h/2.0;
        p[4]=h;
        for(i=1;i<5;i++)
          {
           j=i-1;
           k[i]=m*(xin+p[i]*k[j])+n*u;
          }
         xm=xin+p[4]*(k[1]+2.0*k[2]+2.0*k[3]+k[4])/6.0;
         *xout=xm;
}

inputname()
{
        printf("Type the name of output file:\n");
        scanf("%s",name);
}
```

APPENDIX C

C-1. The program for the Torque-Speed characteristics

```
/*************************************************
*                                                *
*       This is program for T-S characteristic graphic      *
*                                                *
*************************************************/

#include <stdlib.h>
#include <stdio.h>
#include <math.h>

/* Back e.m.f constant K_b is 0.056 V/rpm */

#define   KB   0.056
/* Torque constant K_T is 0.536 Nm/A */

#define   KT   0.536

/* Equivalent DC armature resistance is 0.3 Ω */

#define   RA   0.3

/* Thermal resistance derived from section 5.2.5 is 748 °C/W*/

#define   RH   748

/* Maximum allowable thermal power */

#define   POWER   135.56      /* W */


char   name[10];

main()
{

/* Initialize variable */
```

```
      float  torque=0.0;
      float  speed=5000.0;
      float  current=0.0;
      float  emf=0.0;
      float  y=0.0;
      int    n;
      FILE *dat;
      input();
      dat=fopen(name,"w");
        for(n=0;n<44;n++)
        {
        fprintf(dat,"%f  %f\n",torque,speed);

/* Increase current 0.5 A each time */

        current=current+0.5;

/* Torque = I x K_T */

        torque=current*KT;


/*  E^2 = (P - I^2 x R_a) x R_h

      y=(POWER-current*current*RA)*RH;

/*  E^2 ≥ 0*/

        if(y<0.0)
        y=0.0;
        emf=sqrt(y);

/*  Speed = E/K_b  */

        speed=emf/KB;

/*  Maximum speed is 5000 rpm  */

        if(speed>5000.0)
        speed=5000.0;
        }
     fclose(dat);
}
input()
{
      printf("Type the name of output file:\n");
      scanf("%s",name);
}
```

C-2. Measurement of the BHT motor PWM power loss

The BHT 2214 motor has a constant PWM power loss because a fixed frequency PWM method is used. This power loss is a core loss due to the high frequency PWM current. When the motor is stalled with no load, the input power can be considered as an approximation of the power loss of PWM. This loss can be measured using an oscilloscope Tektronix 2430A and the method is as the following procedure:

1. Set up channels

Select MULT Vertical mode to measure the motor input power, where a current probe amplifier and 100× passive probe are used. The current output is connected to CH 1 which uses a 10 mV/div scale factor. CH 1 VOLTS/DIV control is set to a 500 mV/div scale factor. The voltage output is connected to CH 2, which scale is the 100 V/div.

2. Calculate the value of the MULT vertical mode

1) Compute the MULT scale factor displayed on screen:

$$Scale\ Factor\ Multiplier = \frac{Current\ Amplifier\ Scale\ Factor}{CH\ 1\ Volts/Div}$$

$$= \frac{10mA/div}{500mV/div} = 0.02\ A/V$$

where the current scale factor is 10 mA/div and CH 1 is 500 mV/div.

2) Compute the RMS value of power waveform

$$Power = 0.02A/V \times 25.3V^2 = 0.506\ W$$

where the value measure from the scope is 25.3 V^2.

3) the PWM power loss

This loss is three times of the phase power loss. So the PWM loss is 1.5 W. In order to be safe, we set this loss up to 2W.

C-3. The simulation program for the motor temperature

```
#include <stdlib.h>
#include <stdio.h>
#include <math.h>

#define KP    10
#define KI    0.1
#define KD    100
#define KT    0.536   /* V/rpm */
#define KB    0.056   /* Nm/A  */
#define N     10
#define RA    0.3  /*ohm*/
#define RH    748  /*degree/W*/
#define PWM         2    /* W */
#define SAMPLE              1.0   /* min */
#define THERM       45.0  /* min */
#define THERMALRESIS    1.03  /* degree/W */
#define TEMP                140   /* Insulation temperature */

char    name[10];

float   time=0.0;
float   speed=5.0;
float   load=0.0;
float   power=0.0;

main()
{
     float     x[18]={0.0};
     float     error[5]={0.0};
     float     xin[12]={0.0},tran[12]={0.0};
     float     h=0.0001;
     float     output=0.0,current=0.0;
     float     emf=0.0,power=0.0;
     int       j,i,m,l;

/*parameter of digital controller*/

     float     b1=0.0,b2=0.0,u1=0.0,u2=0.0,e=0.0,u=0.0;
     float     temp,temp1=25;   /* ambient temperature is 25 degree */
     FILE      *dat;
     inputname();

/*numbers of output points */
```

```
        m=40;
        dat=fopen(name,"w");
        for(j=0;j<m;j++)
        {
          fprintf(dat,"\n%f  %f  %f  %f",time,x[9],output,power);
          for(i=0;i<500;i++)
          {


/********** DSP REGULATOR **********/

/*   e(kT)=e(kT)-b(kT)    */

          e=speed-b2;

/* I loop; ul(kT) = KI*e(kT) + ul[(k-1)T]  */

          u2=KI*e+u1;
          u1=u2;

/* P,D loops  ,                                              *
* u(kT) = -KP * b(kT) - KD * {b(kT) - b[(k-1)T]} + ul(kT)  */

          u=(-KP)*b2-KD*(b2-b1)+u2;
          b1=b2;

/* Limit of maximum current */

          if (u>6.3)
          u=6.3;
          if (u<-6.3)
          u=-6.3;

/* Digital controller output */

          x[2]=u;

/* thermal power calculating */

          emf=x[11];
          power=current*current*RA+emf*emf/RH+PWM;

/* temperature predicting */

temp=temp1*exp(-SAMPLE/THERM)
     +THERMALRESIST*(1-exp(-SAMPLE/THEM)*power;
   temp=temp1;
```

```
/******* continue time system *******/

/*sample time is T=n*h*/

    for(l=0;l<N;l++)
    {
       input();
       time+=h;

/*block 4 =x[4]/x[2] = 1/(1+0.002s) filter */

       block(x[2],1.0,0.002,1.0,0.0,h,xin[3],&tran[3],&x[4]);
       xin[3]=tran[3];

/** current loop **/

       error[2]=x[4]-x[16];

/* Ki=2.315 */

       x[5]=2.315*error[2];

/*block 5 = x[6]/x[5] = 0.0022s/(1+0.0022s) */

       block(x[5],0.0,0.0022,1.0,0.0022,h,xin[4],&tran[4],&x[6]);
       xin[4]=tran[4];

/*block 6 = x[7]\x[6] = ko=60.0*/

       x[7]=60.0*x[6];

       error[3]=x[7]-x[11];


/************** MOTOR ************************/

/*block 7 = x[9]/error[3] = 1/(0.3+0.0025s) */

       block(error[3],0.3,0.0025,1.0,0.0,h,xin[5],&tran[5],&x[9]);
       xin[5]=tran[5];

/*block 8 = x[9]/x[10] = KT =0.536 , current to torque*/

       x[10]=KT*x[9];
       error[4]=x[10]-load;

/*block 9 = output/error[4] = (1/F)/(1+J/F s) */
```

```
        block(error[4],1.0,24.5,25000.0,0.0,h,xin[6],&tran[6],&output);
        xin[6]=tran[6];

/*block 10 = feedback e.m.f constant = KB = 0.056 (V/rpm) */

        x[11]=KB*output;


/********* Feedback signal ************/

/*block 11 = speed feedback constant = 0.001 */

        x[12]=0.001*output;


/** block 12 and 13 is speed feedback filter **/

/*block 12 = speed feedback =1/(1+.0003s) */

        block(x[12],1.0,0.0003,1.0,0.0,h,xin[7],&tran[7],&x[13]);
        xin[7]=tran[7];

/*block 13 =  1/(1+0.001s) */

        block(x[13],1.0,0.001,1.0,0.0,h,xin[8],&tran[8],&x[14]);
        xin[8]=tran[8];

/*  speed feedback to digital controller */

        b2=x[14];

/*bklock 14 = current feedback filter = (1+0.00315s)/(1+0.0022s) ; b=0.1 */

        x[15]= 0.1*x[9];
        block(x[15],1.0,0.0022,1.0,0.00215,h,xin[9],&tran[9],&x[16]);
        xin[9]=tran[9];
        current=x[9];
      }
      }
    }
    fclose(dat);
}


/******* Subroutine ******/

/* Function of y(s)/u(s) = (c+ds)/(a+bs)*/
```

```
block(u,a,b,c,d,h,xin,tran,y)
float  u,a,b,c,d,h,xin,*tran,*y;
{
       float   m=(-a/b),n=(1/b),z;
       rk4(u,m,n,h,xin,&z);
      *y=(c-a*d/b)*z+(d/b)*u;
      *tran=z;
}


/*subroutine of the fourth-order Runge-Kutta*/

rk4(u,m,n,h,xin,xout)
float  u,m,n,h,xin,*xout;
{
       float   k[4],p[4],xm=0.0;
       int     i,j=0;
       k[0]=0.0;
       p[1]=0.0;
       p[2]=p[3]=h/2.0;
       p[4]=h;
       for(i=1;i<5;i++)
          {
           j=i-1;
           k[i]=m*(xin+p[i]*k[j])+n*u;
          }
         xm=xin+p[4]*(k[1]+2.0*k[2]+2.0*k[3]+k[4])/6.0;
         *xout=xm;
}


/* Input load variable and speed command */

input()
{
         if(time>2.0 )
         speed=0;
         if(time>2.5 )
         speed=-5;
         if(time>4.0)
         load=11.1;
         if(time>4.5)
         {
           speed=0;
           load=15.1;
         }
         if(time>8.0)
         {
```

```
        load=5.2;
        speed=4;
      }
      if(time>12)
      load=10;
      if(time>16)
      load=0;
      if(time>18)
      load=12;
}

/* Input a name of output file */

inputname()
{
      printf("Type the name of output file:\n");
      scanf("%s",name);
}
```

APPENDIX D

D-1. TMS320C30 Assembly Language Program

```
*****************************************************
*     This is a MOTOR RUNNING program .             *
*...........................................................*
*     The A/D and D/A conversions on Channel A      *
*     They are initiated by timer1. The sample      *
*     time is SAMPLE*0.12(usec) .                   *
*  ..........................................................*
*     Author: Jiabing Lu                            *
*     Date: 20,June 1991                            *
*****************************************************


*** Interrupt Vectors at address 0 ***

        .sect ".init"

RESET       .word  START
INT0  .word  START
INT1  .word  ISR
INT2  .word  START
INT3  .word  START
XINT0       .word  START
RINT0       .word  START
XINT1       .word  START
RINT1       .word  START
TINT0       .word  START
TINT1       .word  START
DINT        .word  START

*** Data Section at address 30000 ***

        .data
```

```
PRIMCTL    .word 00808064h  ; Primary bus control
EXPCTL     .word 00808060h  ; Expansion bus control
PRIMWD     .word 00000800h
EXPWD      .word 00000000h

ADCADR     .word 00804000h  ; Address of A Chan
TIMECTL    .word 00808030h  ; Timer1 control register
PERIOD     .word 00808038h  ; Timer1 period register
RSTCTRL    .word 00000601h
SETCTRL    .word 000006C1h

REF        .word 0003e000h  ; Reference of input
IN         .word         0        ; Input from ADC    IN(n)
IN1        .word         0      ; Previous input from ADC IN(n-1)
UN         .word         0        ; Output to controller ,UN(n)
UN1        .word         0        ; Output of integrator  UN1(n)
UN2        .word         0  ; Previous output of integator UN1(n-1)
ERROR      .word         0        ; Error sample

KP         .word         0003e005h  ; Coefficient of KP
KI         .word         0003e006h  ; Coefficient of KI
KD         .word         0003e007h  ; Coefficient of KD

VIEW       .word         0003e010h
POINT      .word         0003e110h  ;Set point for PC communication
CLEAR      .word 0003e002h

FLAG       .word 0      ; Decides whether sample obtained.

SAMPLE     .word 2000      ; Time period = SAMPLE*0.12 usec

RSP        .word STACK
STACK      .word 0


*** Program Section at address 0D0h ***

        .text

START:
; Set up page pointer for direct addressing.

        LDP PRIMCTL

; Set up stack pointer.

        LDI @RSP,SP
```

```
; Set up primary bus wait states.

        LDI @PRIMCTL,AR0
        LDI @PRIMWD,R0
        STI R0,*AR0

; Set up expansion bus wait states.

        LDI @EXPCTL,AR0
        LDI @EXPWD,R0
        STI R0,*AR0


; Set up  timer1.

        LDI @TIMECTL,AR0 ; Reset control reg
        LDI @RSTCTRL,R0
        STI R0,*AR0

        LDI @PERIOD,AR1  ; Set period reg
        LDI @SAMPLE,R0
        STI R0,*AR1

        LDI @SETCTRL,R0  ; Set control reg
        STI R0,*AR0

;Initial motor
        LDI 1h,R3
        LDI @CLEAR,AR1
WAIT1:
        LDI @ADCADR, AR2
        LDF 1.0,R0
        STI R0,*AR2
        LDI *AR1,R3
        BNZ  WAIT1

;Set pointer for PC communication.

        LDI @VIEW,AR1
        LDI @REF,AR3
        LDI @KP,AR4
        LDI @KI,AR5
        LDI @KD,AR6
        LDI @POINT,AR7

;Enable interrupt for timer1. Set 1 to 1 bit of IE reg.

        OR 2h,IE
```

```
;Set global interrupt enable in status reg. Set 1 to 13 bit of ST.

        OR 2000h,ST


*** Executing program ***
Set up interrupt pointer.
HERE:
        LDI 0h,R1       ;Set flag=0
        STI R1,@FLAG

; Wait for interrupts.
WAIT:
        LDI @FLAG,R1
        BZ  WAIT        ;Is flag=1 ?  If No, repaet loop.


*** PID routine                                          ***
***  U(n)=KI*E(n)+UN1(n-1)-KP*IN(n)-KD*{IN(n)-IN(n-1)}  ***

; Process input sample E(n)=REF-IN(n)

        LDF @IN,R0
        LDF *AR3,R1
        SUBF3 R0,R1,R4   ;R4=REF-IN(n)
        STF R4,@ERROR

; Process integrate UN1(n)=KI*E(n)+UN1(n-1)

        LDF *AR5,R1
        LDF @ERROR,R2
        LDF @UN2,R3
        MPYF3 R1,R2,R5   ;R5=KI*E(n)
        ADDF R3,R5       ;R5=KI*E(n)+UN2
        STF R5,@UN1
        STF R5,@UN2      ;UN2=UN1(n-1)

; Process Proportion KP*IN(n)

        LDF *AR4,R1
        MPYF3 R0,R1,R6   ;R6=KP*IN(n)

; Process differential KD*{IN(n)-IN(n-1)}

        LDF *AR6,R1
        LDF @IN1,R2
        SUBF3 R2,R0,R7   ;R7=IN(n)-IN1
        MPYF R1,R7       ;R7=KD*{IN(n)-IN(n-1)}
```

```
        STF R0,@IN1      ;IN1=IN(n-1)

; UN=UN1-KP*E(n)-KD*{IN(n)-IN(n-1)}

        ADDF R6,R7       ;R7=KP*IN(n)+KD*{IN(n)-IN(n-1)}
        SUBF R7,R5       ;R5=UN1-R7
        STF R5,@UN       ;UN=UN1-KP*IN(n)+KD*{IN(n)-IN(n-1)}

; Transfor data

        LDF @UN,R2    ;Read imformation from UN register.
        BN  OUT
        ADDF 2,R2
OUT:
        SUBF 1,R2
        CMPF 2.0,R2
        LDFGT 1.9999999999,R2
        CMPF -2.0,R2
        LDFLT -1.9999999999,R2

; Output to PC for view
        LDI 0,R7
        CMPI AR1,AR7
        LDIN 1,R7
        STI R7,*AR7

; Output Chan A
        STI R2,*AR2     ; Out Chan A.

        LDF 1,R2        ; Clear R2.

        BR  HERE


*** Interrupt Service Routine ***

ISR:
        LDI @ADCADR,AR2  ; Set pointer
        LDF 1,R3         ; Clear R3.
        LDI *AR2,R3      ; In Chan A


        SUBF 1.0,R3      ;If R3 is positive,converts from range {2.0,1.0}
        BNN  STORE       ;       to {1.0,0.0}. Store the data to memory.
        ADDF 2.0,R3      ;If R3 is negative.Converts from {-2.0,-1.0}
                         ;       to {-1.0,0.0}. Stare the data.

STORE:
```

```
        STF R3,*AR1++
        STF R3,@IN

; Set flag=1

        LDI 1,R0
        STI R0,@FLAG

        RETI

        .end
```

D-2. The PC control program

```
*************************************************
*     This is a TMS320C30 board control program          *
*  ..................................................*
*     Auther: Jiabing Lu                                 *
*     Date: 20,June 1991                                 *
*************************************************

#include "c:\tmsc30\lib\tms30.h"
#include <stdio.h>
#include <stdlib.h>

#define KP 4
#define KI 0.08
#define KD 7
#define N 10

void main(void)
{
int  loadstat;
int  i;
float  view[255];
float  ref_value,ref[N];
float  kp_val,ki_val,kd_val;
kp_val=KP;
ki_val=KI;
kd_val=KD;

/* Initialize board:   */
SelectBoard(0x290);

/* Special load function; required   *
 * with -cr (RAM) linker option.   */

loadstat = coffLoad("motorout.out");
if (loadstat != 0)
  {
  printf("\n\nError During Program Load!!!!\n");
  printf("coffLoad() returned %x\n\n", loadstat);
  exit (0);
  }

Put32Bit(0x3e110l,DUAL,0l);
Put32Bit(0x3e002l,DUAL,1l);
```

```
Reset();
printf("\nSet a Reference:");
scanf("%f",&ref_value);
ref_value=ref_value;
PutFloat(0x3e000l,DUAL,ref_value);    /*DUAL is Bank 3 from 30000h to 3fffh*/

PutFloat(0x3e005l,DUAL,kp_val);
PutFloat(0x3e006l,DUAL,ki_val);
PutFloat(0x3e007l,DUAL,kd_val);

Put32Bit(0x3e002l,DUAL,0l);
while(!(Get32Bit(0x3e110l,DUAL)));

printf("\n\nC30 Program (motorout.out) is running.\n");
for(i=1;i<N;i++)
{
    printf("\nSet a new Reference:");
    scanf("%f",&ref[i]);
    ref[i]=ref[i];
    PutFloat(0x3e000l,DUAL,ref[i]);
}
}
/*  End of main()     */
```

D-3. Interface circuit design

The servo system used for this research is based on an AEG BHT servo driver system which has been described in Appendix A. The BHT servo drive system is mounted as shown in figure C2. If using TMS320C30 board as the speed controller, An interface board has to used to connect the DSP board and the motor control system.

In the motor control system circuit diagram, figure D1, both locations {189} in the feedback path and {94}(a resister) in the forward path are disconnected. The analogue speed loop is taken away. Then connect {187} to the A/D converter of the TMS320C30 board and {94} to the D/A converter. However, the signals at the left point of {187} and also the right point of {94} are +/-10V, but the signals input A/D converter and output from D/A converter are +/-3V. The interface board was built between the BHT controller and the TMS320C30 board to ensure the speed feedback signal from +/-10 to reduce to +/-3V and speed-loop output signal from +/-3V to +/-10V. This interface circuit consists of two groups of linear amplifiers.

D-3.1 Interface board using inverting amplifiers

An operational amplifier is shown in figure D-1. The noninverting input v_i^+ is grounded,

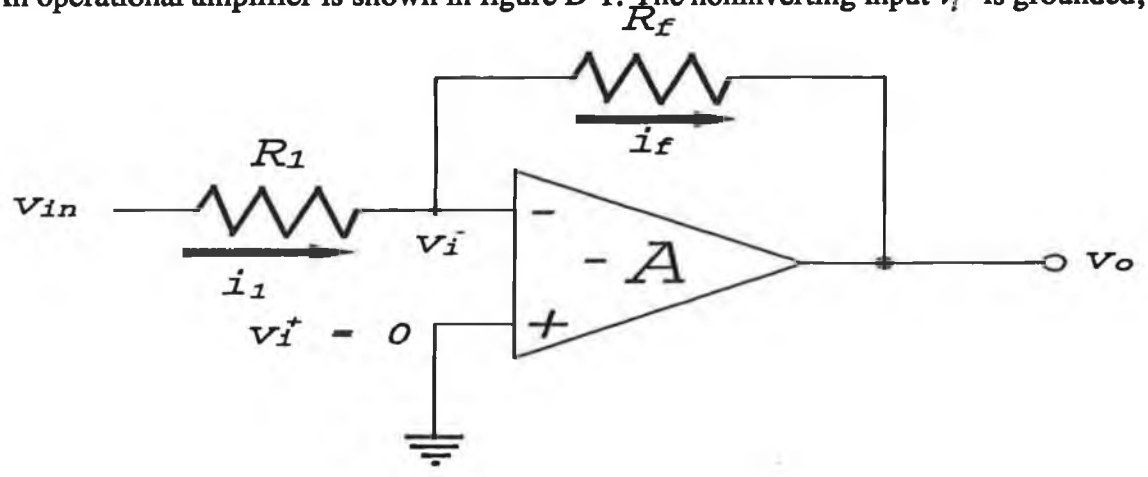


Figure D-1 The Inverting Amplifier

v_{in} is connected through R_1 to the inverting input, and feedback resistor R_f is connected between the output and v_i^-. Since we using the amplifier in an inverting mode, we denote the voltage gain by - A, which is:

$$v_0 = -A \times v_i^- \qquad (1)$$

Using Ohm's law, the current i_1 can be obtained as:

$$i_1 = \frac{v_\epsilon - v_i^-}{R_1} \qquad (2)$$

Similarly, the current i_f is:

$$i_f = \frac{v_i^- - v_o}{R_1} \qquad (3)$$

Because $i_1 = i_f$, we get:

$$\frac{v_\epsilon - v_i^-}{R_1} = \frac{v_i^- - v_0}{R_f} \qquad (4)$$

From the definition (equation 1), $v_i^- = -v_o/A$. If we invoke the assumption that $|A| = \infty$, we see that $-vo/A = 0$, and therefore $v_i^- = 0$ (ideal amp, with $|A| = \infty$). Substituting $v_i^- = 0$ into equation (4) gives:

$$\frac{v_\epsilon}{R_1} = -\frac{v_0}{R_f}$$

or

$$\frac{v_0}{v_\epsilon} = -\frac{R_f}{R_1} \qquad (5)$$

We see that the gain is negative, signifying that the configuration is an inverting amplifier. Equation (5) also reveals that the magnitude of v_o/v_{in} depends only on the ratio of the resistor values.

D-3.2 Interface circuit

Figure D-2 shows the interface circuit.

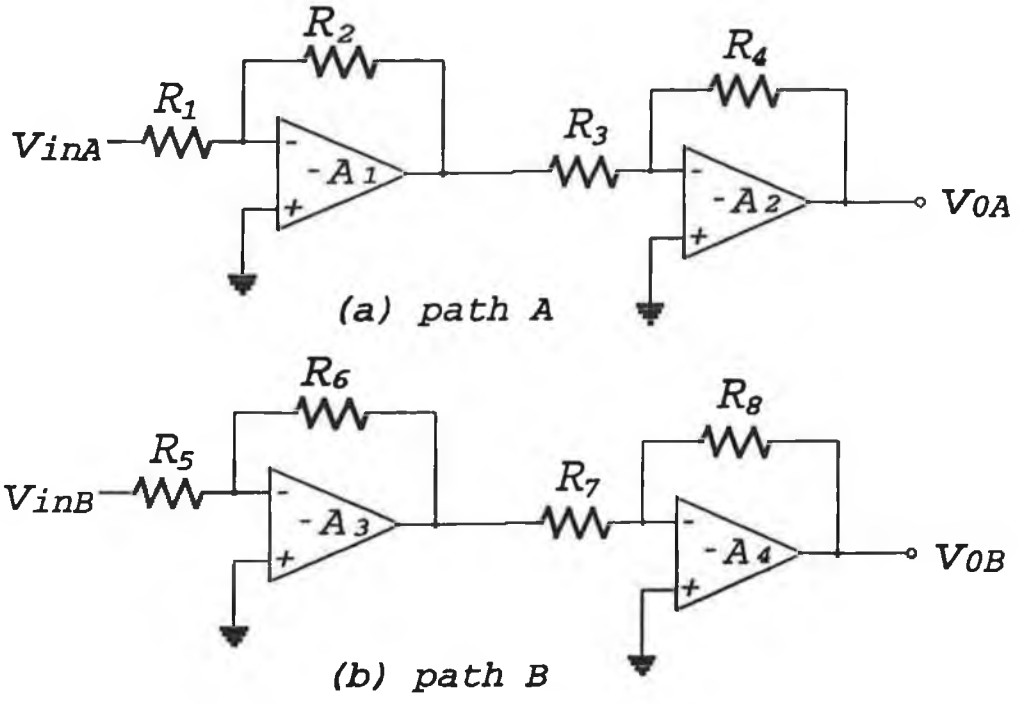


Figure D-2 The diagram of the interface circuit

The figure D-2(a) is path A for system feedback signal from +/-10V to +/-3V. The value of resister R_1 is 33.3K, the R_2 is 10K, the R_3, R_4 are 10K. The gain G_1 of amplifier A_1 is:

$$G_1 = -\frac{R_2}{R_1} = -\frac{3}{10}$$

The gain G_2 of amplifier A_2 is:

$$G_2 = -\frac{R_4}{R_3} = -1$$

The gain G_A of path A is:

$$G_A = \frac{V_{o1}}{V_{in1}} = G_1 \times G_2 = -\frac{3}{10} \times -1 = \frac{3}{10}$$

Figure D-2(b) is path B for speed-loop output signal from +/-3V to +/10V. The R_5 is 10K, the R_6 is 33.3K, the R_7, R^8 is also 10K. The gain G_3 of amplifier A_3 is

$$G_3 = -\frac{R_6}{R_5} = -\frac{10}{3}$$

The gain G_4 of amplifier A_4 is -1 same as A_2. The gain G_B of path B is:

$$G_B = \frac{V_{o2}}{V_{in1}} = G_3 \times G_4 = -\frac{10}{3} \times -1 = \frac{10}{3}$$

The ID connecter (shown in figure C2) is used as the output connect of the BHT controller. Because the analog speed-loop of the BHT controller is not used, pins *5* and *6* of the ID connecter can be employed for the TMS320C30 controller. Pin *5* of ID is used to connect {94}, and pin *6* of ID is used to connect {187}.

The ID connecter pins is shown in table 6.

Table 6. ID pins assignments

ID	MOTOR CONTROL CONNECTER
1	Not used
2	Not used
3	Ground
4	Not used
5	Current-loop command
6	Speed feedback
7	Not used
8	Not used
9	Enable
10	Not used
11	Not used
12	Not used
13	Not used
14	+15V & 50 mA
15	-15V & 50 mA

The connector of interface circuit board is denoted J1, which is described in table 7.

Table 7. J1 pins assignments

J1	INTERFACE CONNECTER
1	Out1
2	In1
3	Ground
4	-15V
5	Not used
6	+15V
7	In2
8	Out2
9	Not used
10	Not used
11	Not used
12	Not used
13	Not used
14	Not used
15	Not used

The connection of ID, J1 and Channel A of TMS320C30 board is shown Table 8.

Table 8. The connection of ID, J1 and Channel A

J1	1	2	3	4	6	7	8
ID	/	6	3	15	14	/	5
Chan A	IN	/	GND	/	/	OUT	/