A COMPACT, FLEXIBLE CONTROLLER FOR PWM INVERTERS

By

KONRAD MAUCH

B.A.Sc., The University of British Columbia, 1977

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

Department of Electrical Engineering

We accept this thesis as conforming

to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

May 1984

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Electrical Engineering

The University of British Columbia
1956 Main Mall
Vancouver, Canada
V6T 1Y3

Date May 24, 1984

ABSTRACT

A flexible, low cost inverter controller for use in variable speed AC motor drives has been designed. Originally designed for use in a submersible thruster drive, the controller has been found to have potential for use in other drives as well. Low cost and small size are made possible by the use of large scale integrated circuits. Flexibility in a variety of motor drive applications is achieved by using a microcomputer as the heart of the controller.

In comparison to other controllers for PWM inverters, this controller has the following distinctive features:

i) The controller can operate the inverter to output frequencies of 400 Hz or more as opposed to maximum frequency limits of 120 Hz to 200 Hz for other controllers.

ii) The controller is programmed in the C high level programming language instead of the assembly language used in other microcomputer based controllers. This allows the controller to be adapted more rapidly to new applications.

iii) The controller is smaller and has a lower component count than most other controllers. As a result the controller should be more reliable, easier to package, and less expensive.

After a review of AC drive technology, the objectives and the basic alternatives in the design of the controller are presented. The circuit design and software design of the controller are discussed. Two applications of the inverter controller are presented and the success of the design in these two applications is evaluated. Conclusions are presented on the overall success of the design and suggestions are made for improvements and further work.

Table of Contents

List of Figures

Acknowledgements

I wish to thank Dr. M.R. Ito for his advice and comments on this thesis. Thanks are also due to Dr. L.M. Wedepohl for his supervision of the B.C. Science Council grants under which much of the work described in this thesis was performed. My colleague Frank Peabody made many useful suggestions and was a great help in the construction and testing of the inverter controller.

My work on this project has received financial support from the Transportation Development Agency, the National Science and Engineering Research Council, and the B.C. Science Council.

Chapter 1

Introduction

Variable speed drives using AC induction motors are finding increasing use in a variety of applications. The increased cost of energy and the move to increased automation has led to the installation of variable speed AC drives in systems which previously used fixed speed motors and mechanical means of controlling output. At the same time the decline in the cost of the electronics in AC drives has made them competitive with DC drives in many of the applications where variable speed motors have traditionally been used.

AC drives have several advantages over DC drives. An AC induction motor is typically smaller, lighter, and cheaper than a DC motor of the same power rating. The maximum speed of an AC induction motor is not limited by the commutation problems experienced by a DC motor. It is therefore possible to design high speed AC induction motors that have a very high power to weight ratio. The absence of a commutator and brushes reduces the maintenance required for an AC drive and allows it to be used in environments where a DC drive can't be used.

While AC variable speed drives based on stator voltage control, slip energy recovery, or cycloconverters are available [1], the majority of AC drives are based on variable frequency operation

with a three phase inverter. The three phase inverter accepts standard fixed frequency, fixed voltage power from the utility line and converts it to variable frequency, variable voltage power which is fed to the AC motor.

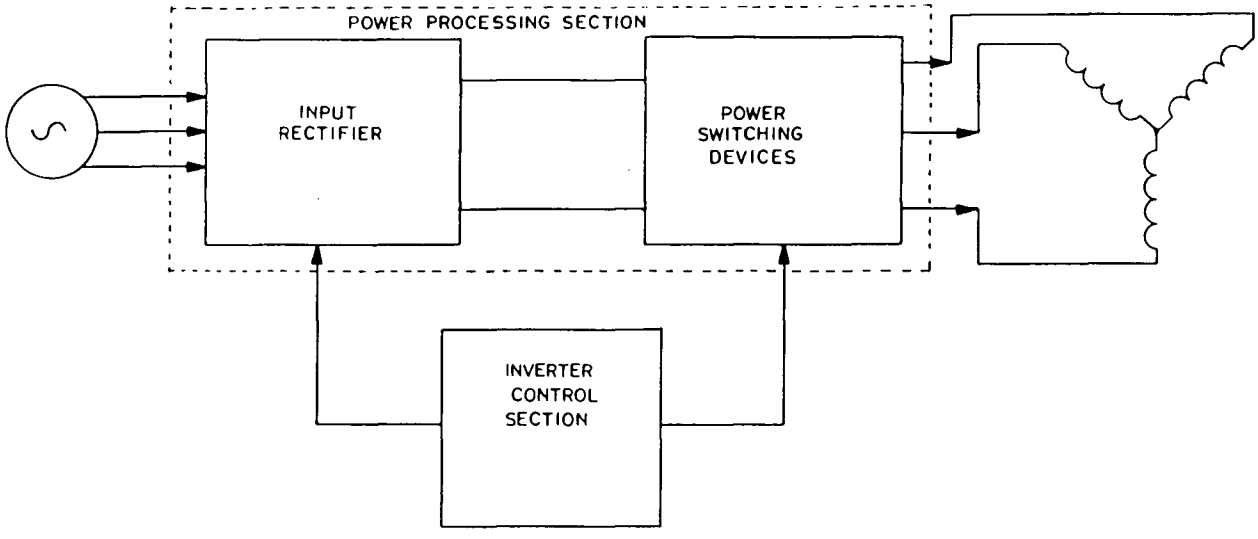


Figure 1 Block Diagram of Typical AC Drive System

An inverter system can be divided into two sections, a power processing section and a control section, as shown in Figure 1. A typical power processing section consists of an input rectifier that converts the input power to DC on an internal bus and a set of power switching devices that switch the DC bus on to the three output lines of the inverter. The switching devices have usually been silicon controlled rectifiers (SCRs) but power transistors and power MOSFETs are increasingly used. The control section controls the opening and closing of the electronic switches in

order to produce the appropriate AC waveforms on the output lines. In addition, it performs necessary sequencing and supervisory functions and accepts control inputs from the system control panel. The controller section has been based on a mix of analog and medium scale digital integrated circuits. However, microcomputers are now being used in some inverter controllers.

1.1 Motivation for Thesis Project

Inverters for AC drives are normally packaged and sold for use in standard industrial environments with standard squirrel cage induction motors. Specialized applications requiring unusual packaging or unusual output voltages and frequencies in most cases require an inverter system that is custom designed. For example, International Submarine Engineering of Port Moody, B.C. had a need for AC drives to power the thrusters on unmanned submersibles produced by the company. Suitable drives were not commercially available. As a result, the Science Council of B.C. funded the development of a suitable drive by the Department of Electrical Engineering at UBC [2,3].

The drive had to be packaged to withstand submersion to depths of up to 1000 meters and yet it also had to be compact since four drives were required on a relatively small submersible. The inverter was required to supply frequencies of up to 400 Hz so that compact, high speed thruster motors could be used. Finally, the inverter controller had to be flexible enough to accomodate

changes in the control strategy due to possible changes in either the power conversion section design or the submersible control system design.

1.2 Scope and Objectives of Thesis Project

This thesis discusses the design of the inverter controller for the submersible thruster drive developed at UBC. The design requirements for this controller (small size, wide frequency range, flexibility in control strategy) were such that it could serve as the basis for an inverter controller applicable to a variety of future applications. Therefore, one of the primary objectives of the design was to allow the maximum amount of flexibility in application wherever this did not conflict with the other objectives that the controller be compact and that it be practical to manufacture in terms of cost and availability of components.

The basic design objectives set for the inverter controller were the following:

The controller should

i. control pulse-width modulated inverters using transistors, power MOSFETs, or thyristors as switches;

ii. operate over an inverter frequency range of 3 Hz to 400 Hz;

iii. perform as many functions as possible by means of computer software written in a high-level programming language;

iv. handle common open-loop and closed-loop motor control strategies;

v. occupy less than 200 cm^2 of printed circuit board;

vi. cost less than $200 in parts and direct labor.

The inverter controller designed to meet the above objectives has several advantages over inverter controllers described in the literature or incorporated in commercially available AC drives. Very few controllers for pulse width modulated inverters that have been described in the literature operate above an output frequency of 250 Hz. There do not appear to be any commercially available pulse width modulated inverters that operate above 250 Hz. Yet the advent of fast power transistors and power MOSFETs allows the design of practical pulse width modulated inverters operating to 400 Hz and beyond. Operation at these high frequencies has advantages in many applications.

Many inverter controller designs presented in the literature are

based on microcomputers and claim to have considerable flexibility in application. However, all of them appear to be programmed in assembly language, which substantially increases the cost of software development for a new application. In addition, many seem to have been designed only for laboratory use and are unsuitable for industrial use in terms of cost, complexity, and size. Controllers in commercially available units meet the cost and size constraints imposed by the market but lack the kind of flexibility that would allow them to be easily reconfigured for a new application. The controller design resulting from the objectives listed above, takes a middle path. It retains the flexibility of having functions performed by microcomputer software, and adds the advantages of high level language programming. But, by judicious partitioning of functions between hardware and software, and by using the minimum number of components, it manages to meet the cost and size requirements for a practical system.

1.3 Outline of Thesis

In order to set the design of the inverter controller in context, and to explain some of the specialized vocabulary and concepts, Chapter 2 is devoted to a review of AC drive technology. The three main types of inverters are briefly described and the characteristics of the semiconductor power switches used in inverters are discussed. Then operation of AC induction motors from a variable frequency source is discussed to provide some

insight into the control aspects of AC drives. Finally, the effects of the non-sinusoidal inverter waveforms on motor operation are discussed along with techniques for reducing these effects.

A detailed discussion of the design requirements for the inverter controller and the possible design tradeoffs is presented in Chapter 3. This is followed by consideration of the alternative methods of implementing the controller functions and by presentation of the basic architecture of the controller.

Chapter 4 describes the circuit design of the inverter controller in detail. Chapter 5 discusses the software design of the controller and presents an evaluation of the success of high level language programming in this application.

Two applications of the inverter controller are described in Chapter 6. The first application is the submersible thruster drive which was the original motivation for the development of the controller. The second application is a drive for an experimental linear induction motor. The success of the inverter controller in these two applications is evaluated.

Chapter 7 ends the thesis and supplies some conclusions on the success of the inverter controller design. Some suggestions for improvements in the design and for future work with the inverter controller are also presented.

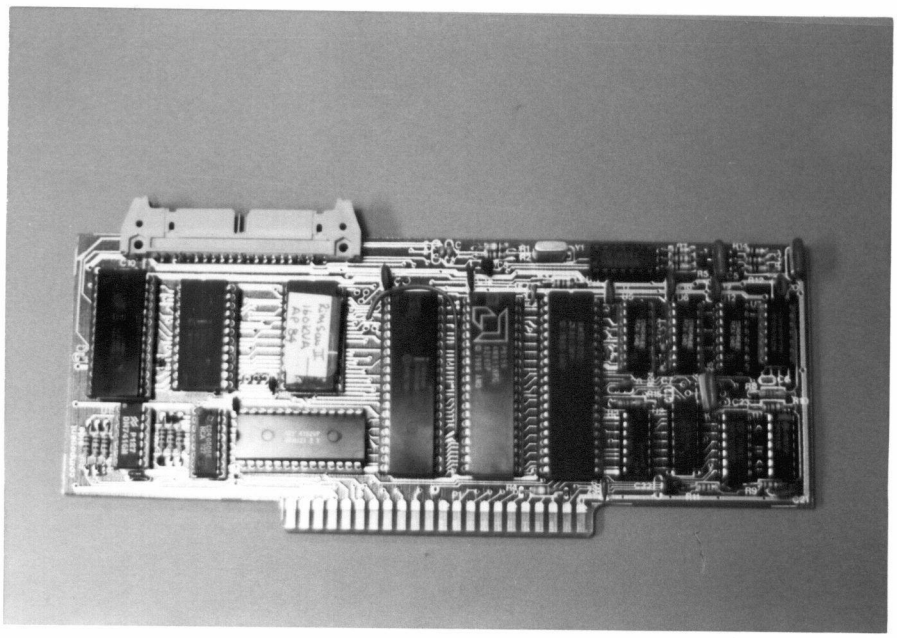


Figure 2 Inverter Controller

1.4 Summary

The end product of this thesis project is shown in Figure 2. The inverter controller is housed on a single printed circuit board approximately 7.5 cm by 21.5 cm. The small size is made possible by the use of large scale integrated circuits. Flexibility in a variety of applications is achieved by using a microcomputer as the heart of the controller. The controller can be adapted to a new application simply by changing the microcomputer's program. The program is written in the C programming language which substantially reduces the program development time and substantially increases the ability to modify and maintain the program when compared to writing programs in assembly language.

The inverter controller meets or exceeds its original design objectives. In comparison to other inverter controllers that have been described, this controller is unique in its combination of good performance and broad range of application with low cost and small size. In addition it is the one of the few pulse width modulated inverter controllers that operates beyond 250 Hz. This makes it useful in applications where high frequency motors or transformers are used to reduce size and weight.

This controller has been used in the prototype submersible drive and in an inverter for a linear induction motor drive. It is probable that both of these applications will end up in production. Further applications of the controller are being planned.

Chapter 2

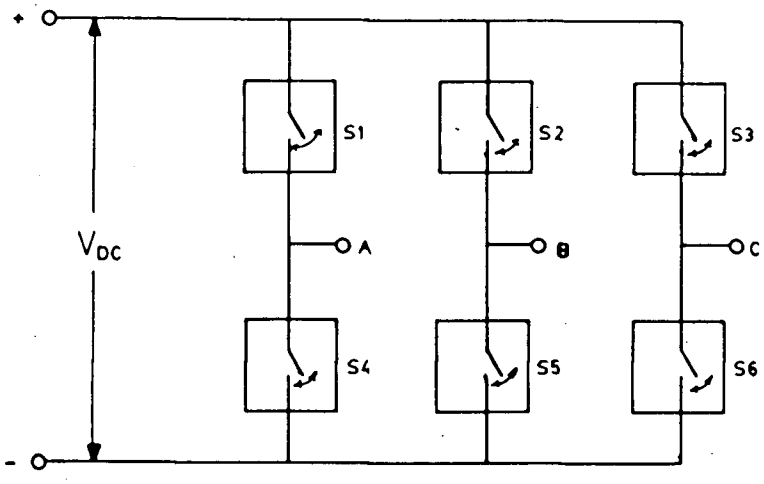
Review of Variable Frequency AC Drive Technology

Variable frequency AC motor drives were made practical by the invention of the silicon controlled rectifier (thyristor) in the late 1950's. Since that time, AC drive technology has progressed significantly. The design of the inverter controller is based on these developments in AC drive technology. This chapter presents a review of some aspects of this technology in order to provide some context for the design of the inverter controller and to introduce concepts and terminology which will be used in later chapters.

2.1 Classification of Inverter Drives

The basic three phase inverter bridge circuit is shown in Figure 3. A set of symmetrical three phase waveforms can be generated by operating the switches in the sequence shown in Figure 4. By varying the switching rate, the output frequency of the inverter can be varied. In motor drive applications the output voltage of the inverter must also be varied. The method used to vary the output voltage provides a convenient means for classifying the different types of inverter motor drives.

In the Variable Voltage Input Inverter (Figure 5-1), the voltage



THREE PHASE INVERTER

Figure 3

Basic Inverter Bridge Circuit

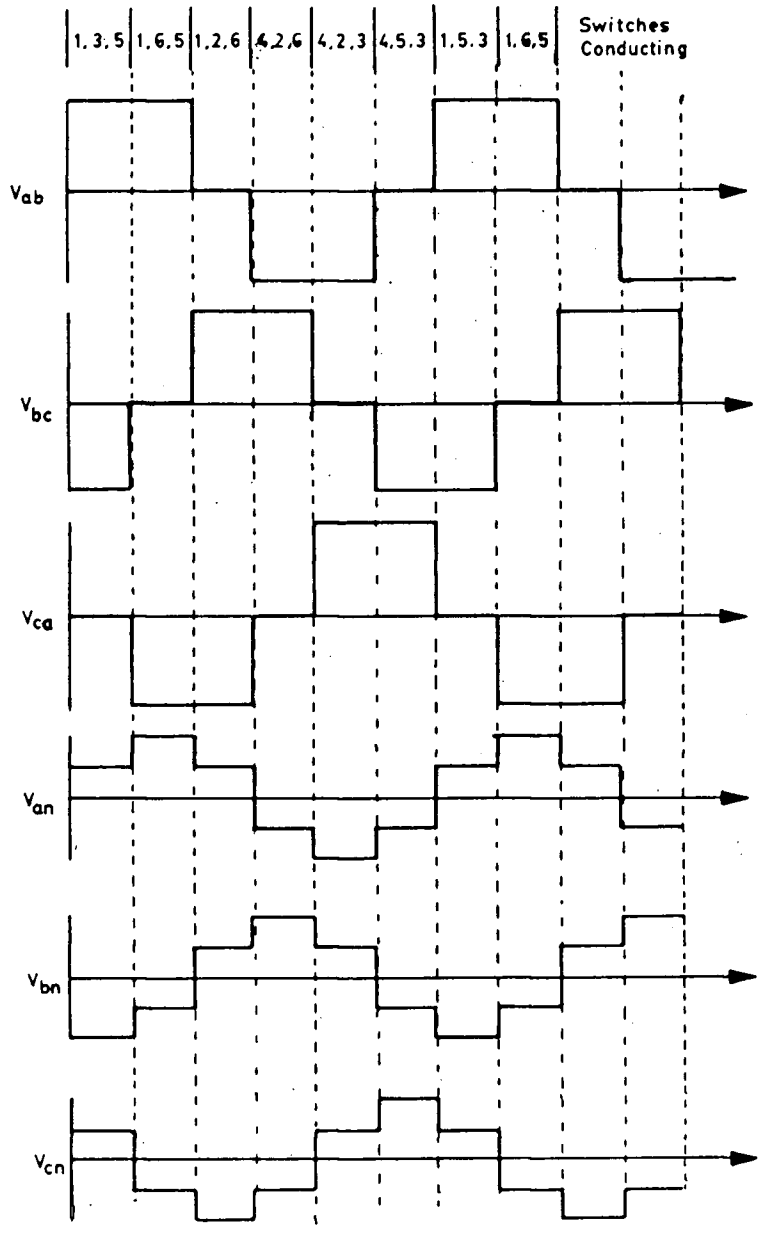


Figure 4

Inverter Switching Sequence

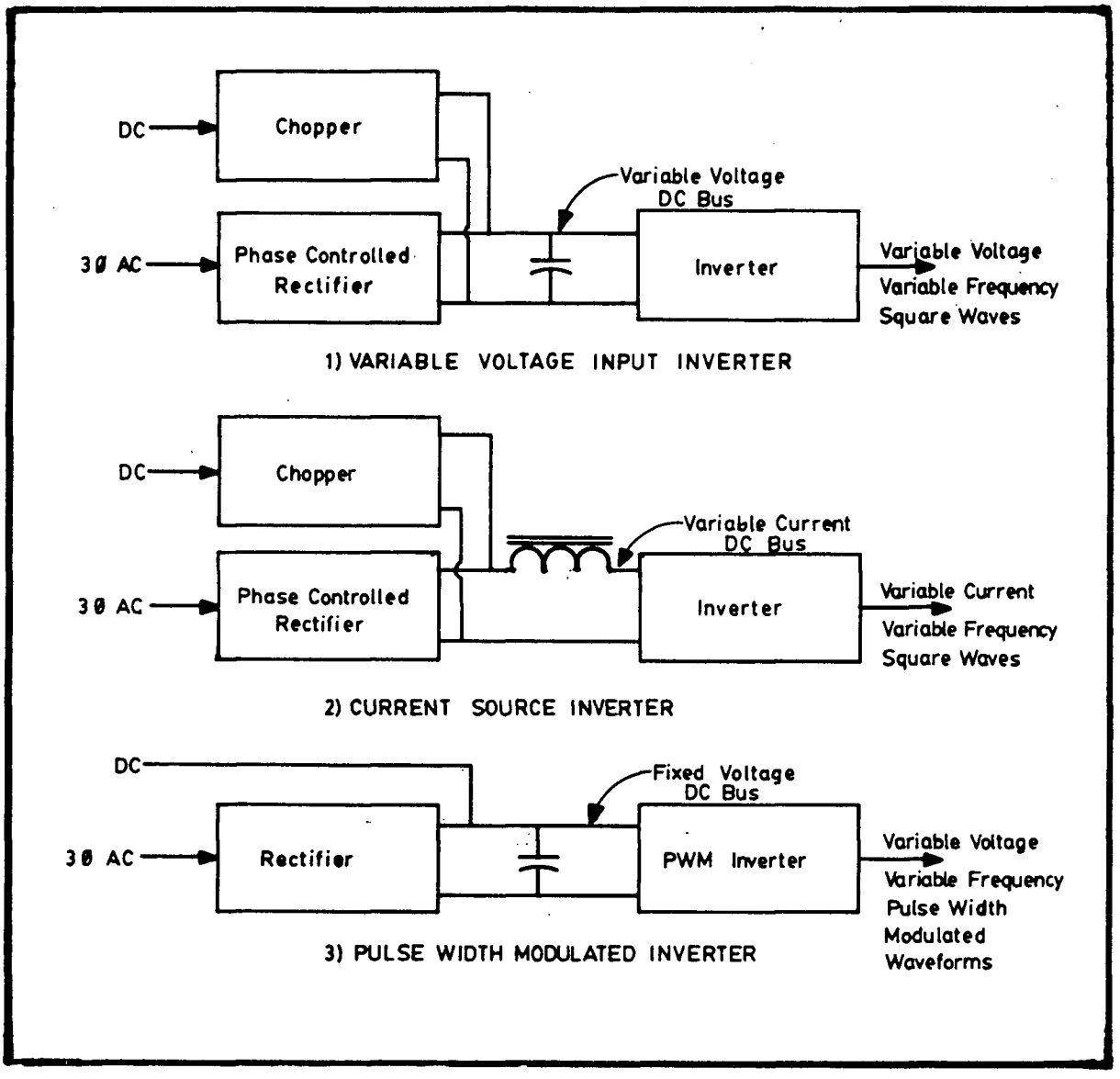


Figure 5 Basic Inverter Types

on the DC bus feeding the inverter bridge is varied in order to vary the output voltage. Depending on the nature of the power source, the voltage control can be performed by a phase controlled rectifier or by a chopper. Since the output of the inverter consists of variable amplitude square waves with a six-step pattern, this type of inverter is also called a "six-step" or "square wave" inverter.

The Current Source Inverter (Figure 5-2) differs from the Variable Voltage Input Inverter in that the DC bus is designed to act as a current source rather than a voltage source. The phase controlled rectifier or chopper at the input regulates the bus current rather than the bus voltage. The output of the inverter consists of square waves of current rather than square waves of voltage.

The Current Source Inverter is attractive when the inverter uses thyristors as switching elements since the commutation circuits are simple and relatively slow thyristors can be used [4]. Also the Current Source Inverter allows regeneration of energy back to the AC line with no extra circuitry. When the machine is acting as a generator, the inverter switching sequence with respect to the machine terminal voltage is adjusted so that the voltage on the DC bus is reversed. The firing angle of the phase controlled rectifier is then retarded so that power flows from the DC bus back into the AC source in the same fashion as a four quadrant DC drive. The Current Source drive is not normally used with transistor

or power MOSFET based inverters since commutation and device switching speed do not pose any problems.

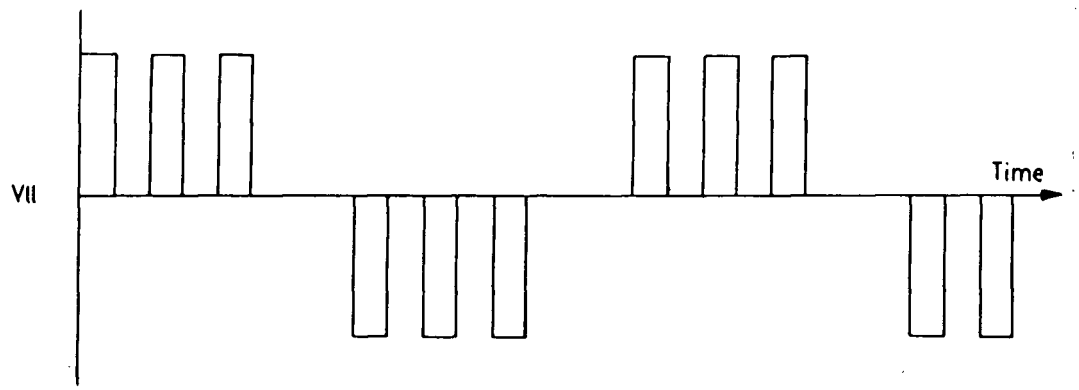


Figure 6 PWM Inverter Output Voltage Waveform

The Pulse Width Modulated (PWM) Inverter (Figure 5-3) operates from a fixed voltage DC bus. The output voltage is controlled by turning the switches on and off many times within a half cycle of the output wave as shown in Figure 6. By varying the ratio of on time to off time, the output voltage can be varied. This variation in the duty cycle can also be used to control the harmonic content of the output waveform. The PWM inverter has the advantages that it has a very simple input power conversion section, if any is required at all, and it can change voltage rapidly since the time constant of the DC bus filter does not

affect the speed of response. A PWM inverter presents a high power factor to the AC power source since a phase controlled rectifier is not required at the input. The PWM inverter does require fast switches which exhibit low switching losses and therefore is better suited to use with transistors and power MOSFETs than with SCRs.

The design of the controller is strongly influenced by the type of drive. In the Variable Input Voltage and Current Source inverters, the inverter switching sequence is simple and can be controlled by something as simple as an up/down counter driven by a variable frequency clock. The input power converters to these drives must be controlled in order to produce the desired DC bus voltage or current. The PWM inverter has a much more complex inverter switching pattern and therefore requires a more complex controller but a separate controller for the input power converter is not required.

2.2 Inverter Power Switching Devices

The power switches used in inverters intended for motor drives are solid state semiconductor devices. In the past, the thyristor has been the most commonly used switching device and it is still used extensively in higher power drives. The thyristor can be switched on by a low power pulse into its gate terminal but it can only be turned off by interrupting the current flow through it for a certain period. In induction machine drives,

this interruption of current flow, called commutation, does not occur naturally as it does in phase controlled rectifiers. As a result, the inverter circuit must include some means to force the current flowing through the thyristor to zero so that the thyristor commutates. These forced commutation circuits add considerably to the complexity of thyristor based inverters. In addition, the period of time required for the complete commutation can be as long as 100 microseconds which places limitations on the maximum switching frequency of the inverter.

A normal thyristor blocks voltage in both directions when it is off. However most inverter circuits require voltage blocking in only one direction. Assymetrical thyristors and reverse conducting thyristors have been designed which have little or no voltage blocking capability in one direction. As a result, other parameters, particularly the time required to commutate the device, can be improved. Inverters using these newer devices are beginning to appear on the market.

Another variation on the standard thyristor is the gate turn off thyristor. This device can be turned off by applying a large reverse gate current. This simplifies the design of the inverter somewhat and eliminates bulky forced commutation components. Again, inverters using gate turn off thyristors as switches are appearing on the market.

In low to medium power applications, the bipolar power transistor

is begining to supplant the thyristor as the inverter power switch. Transistors which can block 1000 volts and conduct 300 amperes are now available so transistorized drives with ratings up to 150 kilowatts can be manufactured without the need to parallel devices. The switching of the transistor is completely controlled by the current applied to the base, therefore no external commutation circuits are required. Transistors switch more quickly than thyristors so they are better suited to inverters which have high switching frequencies.

At power levels below 5 kilowatts, power MOSFETs are sometimes used as the power switching devices. These devices are very easy to control, requiring very little gate power. They also have very high switching speeds which makes them suitable for use in high frequency inverters. Power MOSFETs are still relatively expensive so their application in motor drives is presently limited to experimental systems and to special purpose systems which require their special characteristics.

The design of the inverter controller must take into account the type of power switching devices used. A controller for an inverter using thyristors must properly sequence the firing of the main thyristors and the thyristors in the forced commutation circuit in order to ensure proper commutation of the thyristors.

Inverters based on power transistors are easier to control but the controller must allow for the switching time of the

transistors. In large power transistors there can be as much as 20 microseconds between the time that base current is removed from the device and the time that collector current falls to zero. If a transistor in one leg of an inverter is turned on before the other transistor in the same leg is completely off, a short circuit will occur across the DC bus and potentially damaging "shoot through" currents will flow through the two transistors.

Power MOSFETs are the easiest of the three types of devices to control since they require very little drive power and switch extremely fast. The controller can treat them pretty much as ideal switches.

2.3 Variable Frequency Operation of Induction Motors

The three phase induction motor operating from a fixed frequency voltage source has the familiar torque-speed curve shown in Figure 7. If the source frequency is varied and the motor airgap flux is kept constant, the torque-speed curve remains the same except that it is translated to the left or right. If torque/speed curves are plotted for a range of frequencies, a family of curves as shown in Figure 8 results. These curves show that variable speed operation is possible with the induction machine retaining basically the same properties that it has when it is running with low slip at 50 or 60 Hz.

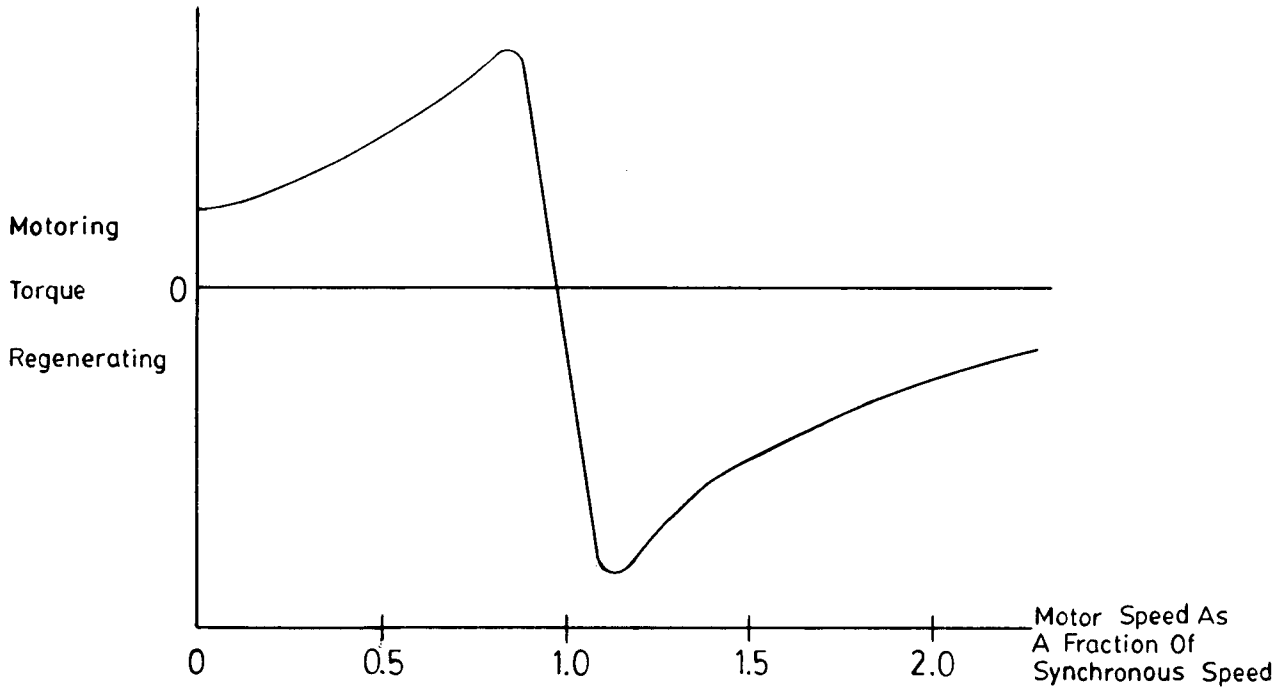


Figure 7 Induction Machine Torque-Speed Curve

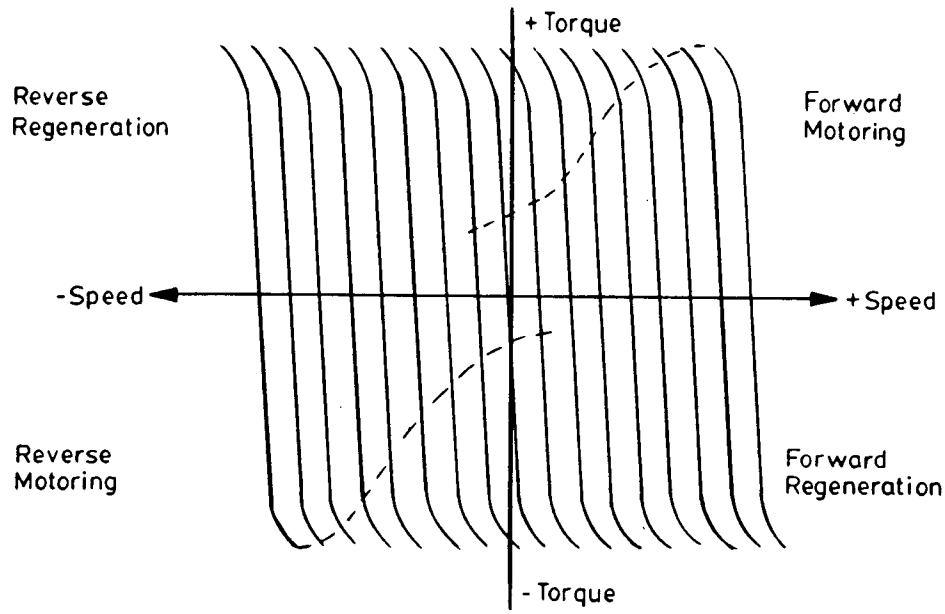


Figure 8 Torque-Speed Curves at Different Stator Frequencies

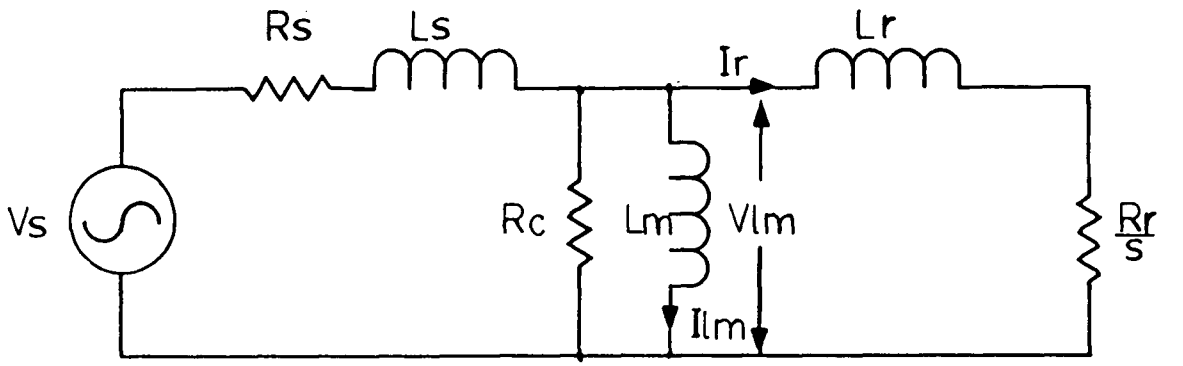


Figure 9 Standard Equivalent Circuit for the Induction Machine

In fact, the torque produced by the induction machine does not fundamentally depend on the stator frequency (w_e). The torque equation for the induction machine operating from a voltage source can be developed from the standard equivalent circuit for the induction machine (Figure 9):

(1) Rotor input power = Ir^2Rr/s

(2) Electromagnetic torque = Te = Rotor input power/w_e [5]

(3) $$Ir^2 = \frac{V_{Lm}^2}{((w_eLr)^2 + (Rr/s)^2)}$$

(4) $|V_{Lm}| = w_eImLm$

(5) Airgap Flux = $\emptyset$ = ImLm

$$(6)\quad s = w_{slip}/w_e$$

$$(7)\quad Te = \frac{\phi^2 Rr/w_{slip}}{(Lr^2 + (Rr/W_{slip})^2)}$$

When w_{slip} is small (the normal operating condition), $Rr/w_{slip} >> Lr$ and the machine torque can be approximated as:

$$(8)\quad Te = \phi^2 w_{slip}/Rr$$

Thus the torque depends on slip frequency and airgap flux but not on the stator frequency.

Control schemes for induction motor drives attempt to control the machine airgap flux or the slip frequency in order to control torque. Most drives used in industrial applications use a simple open loop control strategy called constant volts/Hz control. If the stator impedance is ignored, airgap flux is proportional to the stator voltage divided by the stator frequency. Thus approximate constant airgap flux operation can be maintained by maintaining the ratio of stator voltage to stator frequency constant. The value of the ratio is usually set to the nominal stator voltage (e.g. 440 volts) divided by the normal operating frequency (e.g. 60 Hz). At low stator frequencies the voltage drop across the stator resistance becomes an important factor and reduces the airgap flux level if constant stator volts/Hz is maintained. A boost in the volts/Hz ratio is sometimes added at low frequencies to compensate for the stator resistance drop. However, the magnitude of the stator voltage drop depends on the stator current so a fixed boost in volts/Hz ratio will not

compensate for all load conditions.

Most inverters have a maximum output voltage determined by the inverter supply voltage. Once this maximum voltage is reached in constant volts/Hz operation, any further increase in frequency will result in the airgap flux decreasing linearly with increasing frequency. As a result the peak torque capability of the machine will drop off at a rate proportional to the square of the frequency. The torque envelope for the induction machine operating under constant volts/Hz control is shown in Figure 10.

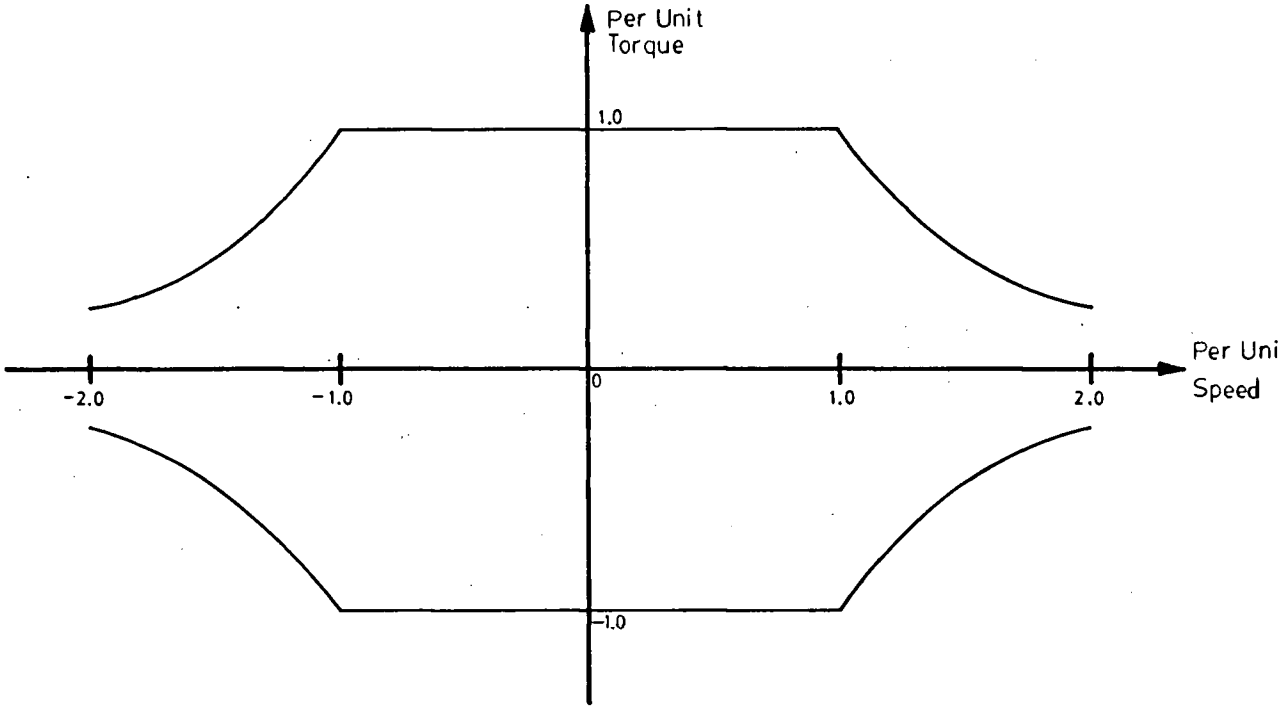


Figure 10 Torque Envelope for Constant Volts/Hz Operation

As long as the load torque of the machine is within this envelope, the speed will be relatively insensitive to load torque variations since the machine torque/speed curve is quite steep

when operating at low slip frequencies. The machine characteristics thus resemble those of a DC shunt wound machine.

Constant volts/Hz operation is only an approximation to true constant airgap flux operation since the effects of the stator impedance are ignored. True constant airgap flux operation offers improvements in both static and dynamic performance over simple constant volts/Hz operation [6]. However some means of determining the airgap flux is required. Direct measurement using Hall-effect devices [7] or flux sensing coils [8] installed in the machine is possible but depends on the availability of machines with these sensors installed. Direct calculation of the airgap flux using terminal voltages and currents and the parameters of the machine's equivalent circuit has also been described in the literature [9]. This technique does not require special machines but does require voltage and current feedback as well as a priori knowledge of the machine's parameters. Due to the increased complexity of true flux control and the relatively modest requirements of most industrial variable speed drives, the constant volts/Hz drive is still the most commonly used AC drive.

AC drive systems which rely on the control of motor slip frequency have also been designed [10]. In the controlled slip frequency drive, the stator frequency is determined by adding the desired slip frequency to the motor mechanical frequency which is measured by a tachometer. Two methods of torque control are commonly used in controlled slip drives. In the first technique,

the airgap flux is maintained approximately constant through constant volts/Hz operation and torque is controlled by varying the slip frequency. Alternatively, the slip frequency is maintained at a fixed value and the torque is controlled by varying the stator voltage which in turn varies the airgap flux. The controlled slip drive has poor speed regulation since the motor basically increases its speed until the load torque matches the machine torque but it has fairly good torque regulation characteristics since torque can be directly controlled by the slip frequency or stator voltage. As a result the slip controlled drive has been used in traction applications such as electric vehicles [11] and the Canadian Advanced Light Rapid Transit System [12] where torque control rather than speed control is desired.

The slip frequency controlled drive requires slip frequency feedback from the motor. Normally a tachometer is used to measure the rotor speed which provides an indirect measure of slip frequency. However, since slip frequency is the small difference between the rotor rotational frequency and the stator frequency, a very accurate tachometer is required. These tachometers add significantly to the cost of the drive system and make the system less rugged. Attempts have been made to use lower cost tachometers [13] or to calculate the slip directly from machine terminal voltages and currents [14].

The controlled slip drive is inherently protected against the

possibility of "pull out" in which an increase in load torque causes the slip frequency to increase beyond the point for maximum torque (pull out torque) and enter the positively sloped portion of the torque/speed curve as shown in Figure 11.

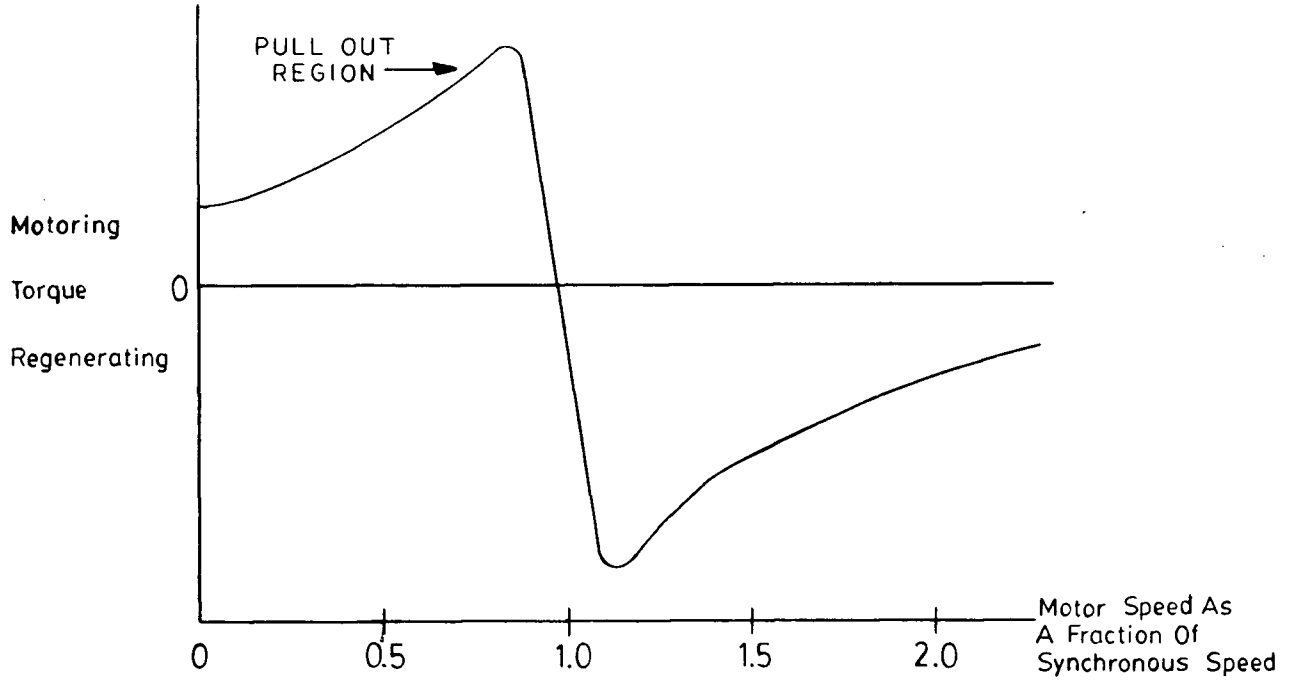


Figure 11 Torque/Speed Curve Showing Pull Out

Operation on the positively sloped portion of the torque/speed curve is characterized by poor efficiency and power factor, high stator currents, and poor speed regulation. Drives not equipped with slip frequency control attempt to prevent pull out in a variety of ways. Most open loop constant volts/Hz drives have adjustable acceleration and deceleration rates so that different

load inertias can be accomodated without exceeding the maximum torque capability of the machine. Stator current limits which cause a decrease in stator frequency when the current limit is exceeded act as an approximate form of slip frequency control. A better approximation is achieved when the real component of stator current is used as the indicator of slip since this eliminates the effect of the magnetizing current [15]. The machine power factor has also been used as an easy to measure indicator of slip [16].

AC drives running from current source inverters present a special control problem. If the induction motor equivalent circuit of Figure 9 is fed by a current source delivering a stator current Is, the machine torque equations are developed as follows:

$$(9)\quad Te = 3/w_e(Ir^2Rr/s)$$

$$(10)\quad Ir^2 = \frac{Is^2w_e^2Lm^2}{w_e^2(Lm + Lr)^2+(Rr/s)^2}$$

$$(11)\quad Te = \frac{3Is^2Lm^2Rrw_{slip}}{w_{slip}^2(Lm+Lr)^2+Rr^2}$$

This torque equation can be rewritten in terms of per-unit slip and with the inductances replaced by their impedances to allow comparison with the standard torque equation for the induction machine operating from a voltage source:

(12) Current source operation

$$T = \frac{3/w_e Is^2 Xm^2 Rr/s}{(Xm+Xr)^2+(Rr/s)^2}$$

(13) Voltage source operation [17]

$$T = \frac{3/w_e Vs^2 Rr/s}{(Rs+Rr/s)^2+(Xs+Xr)^2}$$

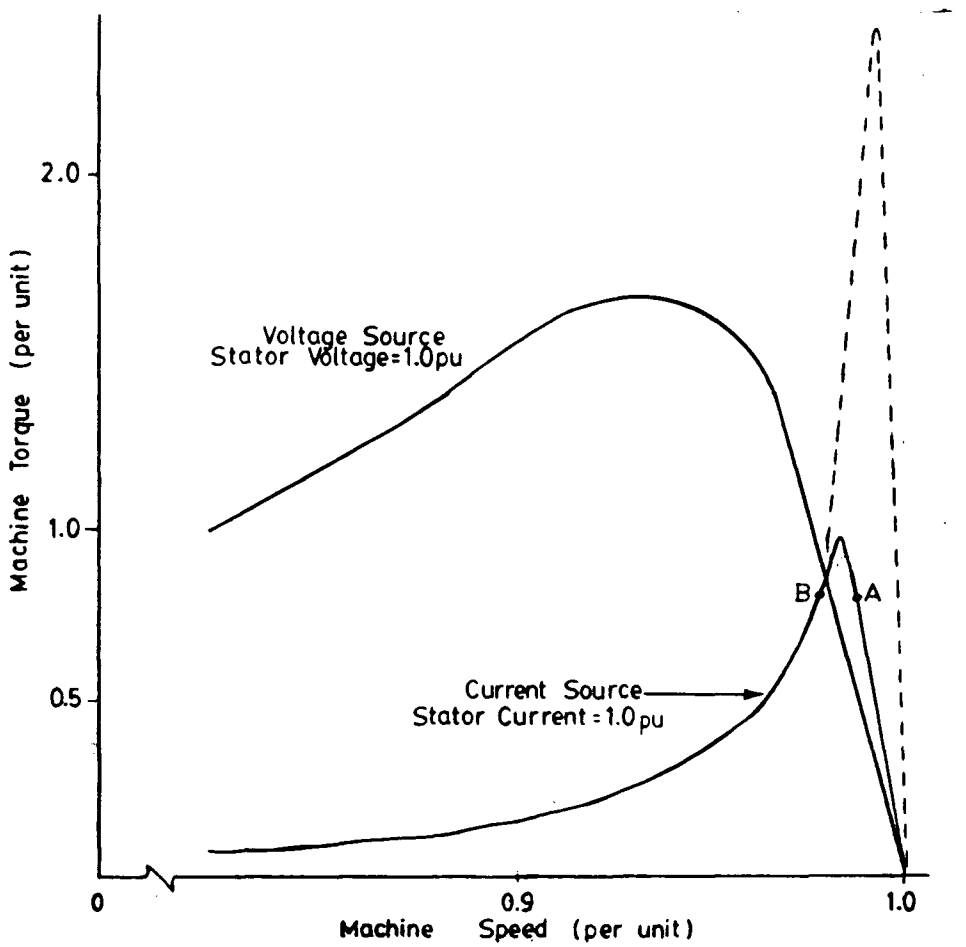


Figure 12 Torque-Speed Curves for Current and Voltage Source Operation

Figure 12 shows torque-speed curves, generated with the equations above, for an induction machine running from a voltage source and a current source. The dotted line on the current source curve

indicates the theoretical curve. However that curve results in very high airgap flux levels when the slip is small since Rr/s becomes very large and most of the stator current flows through the magnetizing inductance Lm rather than into the rotor circuit. In a real machine, the iron saturates and the torque curve shown as a solid line results.

The machine can operate at rated torque at either point A or point B on the constant current torque/speed curve. Operation at point A on the negatively sloped portion of the torque speed curve corresponds to the operating region in which the machine iron is saturated and the machine is running inefficiently. Therefore operation at point B on the positively sloped portion of the curve is desirable. However this curve results in statically unstable operation since any transient perturbation in load torque causes the machine to either slow down to a halt or speed up so that it enters the stable negatively sloped portion of the torque/speed curve. As a result, current source AC drives are usually operated under some type of closed loop control. Controlled slip frequency operation with control of torque by variation of the stator current is a common control technique [18, 19].

The dynamics of AC drives have been intensively studied as engineers attempt to design AC drives with the same fast response as DC drives. The standard per-phase equivalent circuit of the induction motor (Figure 9) is a good representation of the

machine under steady state conditions but is not accurate under transient conditions. The two axis or d-q model used for synchronous machine transient analysis can also be used for induction machine transient analysis [20]. Krause and Thomas have developed the induction machine equations for d-q coordinates in a reference frame rotating with angular velocity w_e [21] as

$$(14)\quad \begin{bmatrix} V_{qs} \\ V_{ds} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} R_s+L_s p & w_e L_s & L_m p & w_e L_m \\ -w_e L_s & R_s+L_s p & -w_e L_m & L_m p \\ L_m p & (w_e-w_r)L_m & R_r+L_r p & (w_e-w_r)L_r \\ -(w_e-w_r)L_m & L_m p & -(w_e-w_r)L_r & R_r+L_r p \end{bmatrix} \times \begin{bmatrix} i_{qs} \\ i_{ds} \\ i_{qr} \\ i_{dr} \end{bmatrix}$$

p = d/dt operator

Ls = Stator leakage plus magnetizing inductance

Lr = Rotor leakage plus magnetizing inductance

w_r = Rotor angular velocity

These differential equations result in a mathematically non-linear state equation since the rotor frequency w_r varies with the operating point. In addition the machine inductances and resistances are not constant parameters since they change due to magnetic saturation and motor heating. As a result general transfer functions for the induction machine are not possible since the pole and zero locations depend on the operating points. Instead, transfer functions are derived by

linearizing around an operating point to obtain a linear small signal model [22,23]. Poles and zeroes for the linear model can be determined at a variety of operating points and plotted on a root locus diagram in order to determine regions of potential instability and to give some insight into the nature of the system response.

The major results of these studies can be summarized as follows:

(1) The rotor speed response to step changes in load torque, input voltage, or input frequency is approximately first order when the machine and load have high inertia. The time constant is [24]

$$T = \frac{Jw_{slip}}{nTo} \tag{15}$$

where

J = total inertia of machine plus load

n = number of machine pole pairs

To = full load machine torque per pole

w_{slip} = slip frequency at full load torque

(2) The trend in rotor speed response is towards a second order type as machine and load inertia decreases. Some lightly loaded machines actually go into steady state rotor speed oscillations when operated at low stator frequency [25]. These oscillations can sometimes be induced in otherwise stable machines

by interaction with the filter elements in the inverter's DC bus [26].

(3) Very good dynamic performance can be achieved by an induction motor drive if the stator currents are controlled in a d-q reference frame rotating at the velocity of the rotor flux vector. The d-axis vector is aligned with the flux vector so that the d-axis stator current effectively controls the machine flux. When the d-axis current is aligned in this fashion, the q-axis current directly controls the machine torque. This "field oriented control" [27] allows independent control of flux and torque since the two current vectors are orthogonal. This is analogous to the control of a separately excited DC machine where the field current controls the machine flux and the armature current controls the machine torque. Most other induction motor control control schemes do not succeed in decoupling the control variables for flux and torque so a change in the setpoint for one will cause a transient perturbation in the other.

2.4 Control of the Inverter Waveform

The output waveform generated by the inverter is usually non-sinusoidal. Since induction machines are designed to operate

with sinusoidal currents, the effects of operation with non-sinusoidal currents must be evaluated to determine how machine performance is affected. For the purpose of analysis, the induction machine can be considered to be a linear system as long as magnetic saturation is ignored [28]. The waveform can therefore be decomposed into a fundamental component and a set of harmonic components by Fourier analysis and the machine response can be determined for each harmonic individually. The total response is obtained by summing the responses to the fundamental and harmonic components.

Waveforms from a properly operating inverter are symmetrical and have no DC offset. As a result the waveform has no DC component and no even harmonics. In addition, the triplen harmonics (harmonics of order 3k where k is a positive integer) in a three phase system are all in phase (zero sequence) and so produce no machine flux. As a result they will not affect the machine's torque. If the machine is connected in a wye configuration, no triplen harmonic will flow in the windings. Of the remaining harmonics, those of order 6k+1 produce machine flux that rotates in the same direction as the fundamental flux (positive sequence) while those of order 6k-1 produce machine flux that rotates in the opposite direction to to the fundamental flux (negative sequence).

The equivalent circuit of the induction machine shown in Figure 9 can be used to develop a simpler circuit which is valid for the

harmonic components of the excitation waveform. The slip for the harmonic components is

$$s = \frac{kw_e - w_{rotor}}{kw_e} \tag{16}$$

where

w_e is the fundamental stator frequency

k is the order of the harmonic component

w_{rotor} is the rotor mechanical frequency

From this equation it is apparent that the slip for the harmonic components is close to 1 since the lowest harmonic component is of order k = 5. This means that the harmonic components of the machine current will not be greatly affected by the load on the machine. At most operating frequencies the input impedance seen by the harmonic components is dominated by the reactances in the circuit since the reactances increase with frequency while the resistances remain constant (ignoring skin effect for the moment). The impedance of the magnetizing branch is very large at harmonic frequencies and as a result its effect on harmonic currents is small. If the machine resistances and its magnetizing branch are neglected, the equivalent circuit can be reduced to the leakage reactances as shown in Figure 13.

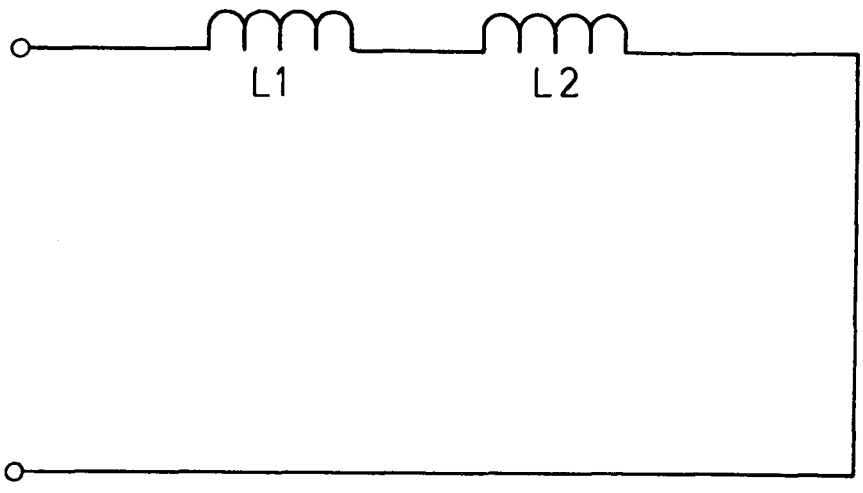


Figure 13 Machine Equivalent Circuit for Harmonics

Using this approximate equivalent circuit, the harmonic currents flowing into the machine can be calculated as follows:

$$(17) \qquad i_k = \frac{v_k}{k(X_1+X_2)}$$

where

i_k = magnitude of current harmonic of order k

v_k = magnitude of voltage harmonic of order k

X_1, X_2 = stator and rotor leakage reactances at fundamental frequency

The induction machine acts as a low pass filter which attenuates higher

order current harmonics. Since the voltage harmonics in inverter waveforms decrease in magnitude as their order increases, the net result is that machine performance is primarily affected by the low order harmonics in the waveform.

In an inverter with a six-step (square wave) waveform, the voltage waveform can be expressed by the following Fourier series:

$$(18) \qquad v(t) = V_1(\sin wt + 1/5\sin 5wt + 1/7\sin 7wt + 1/11\sin 11wt + \ldots)$$

The dominant harmonics are the fifth and seventh. The fifth harmonic produces flux rotating in the opposite direction to the fundamental flux. The flux produced by the seventh harmonic rotates in the same direction as the fundamental flux. Under normal operating conditions, the rotor rotates at approximately synchronous speed and therefore the rotor bars cut the rotating fifth and seventh harmonic fluxes at the sixth harmonic frequency. As a result sixth harmonic currents are induced in the rotor. These currents interact with the fundamental flux to produce a pulsating torque at the sixth harmonic frequency [29]. The torque has zero average value.

This pulsating torque represents the major effect that harmonics have on machine torque. The magnitude of this harmonic torque depends on the magnitude of the fifth and seventh harmonic currents. High leakage reactance machines fed from a voltage source inverter have the smallest torque pulsations since

harmonic currents are attenuated by the motor leakage reactance. On the other hand, motors fed from a current source inverter have the largest torque pulsations since the motor is fed a square wave of current in which the harmonic currents are not attenuated at all. An average machine fed from a voltage source inverter has torque pulsations with a magnitude of about 0.1 p.u. [30].

The effect of the torque pulsations on overall system performance depends on rotor and load inertia and on operating speed. On one hand, a low inertia machine operating at low speed (a few Hertz) may actually rotate in a series of steps or jerks. On the other hand, a high inertia machine running at high speeds (50 Hertz) may not be noticeably affected by the pulsating torque. In intermediate situations, the pulsating torque may be noticeable as a vibration that can increase wear on couplings and gears.

Since harmonic currents add little if anything to the machine's output power, their effect is seen primarily as an increase in the machine's electrical losses. The flux generated by the harmonic currents causes an increase in the core loss and the stray losses in the machine. The increase in core loss has been found to be small; on the order of 10% [31]. The stray losses in the machine have been found to be as much as two or three times greater when the machine is running from a non-sinusoidal source [32]. Fortunately, the stray loss is a small fraction of the machine's total losses so a large increase does not severely

impair the machine's efficiency.

The main source of increased motor losses, however, is harmonic copper losses. Skin effect in most induction machine stators is negligible since the windings in most cases consist of insulated wire of relatively small cross section. As a result, the additional copper losses in the stator are:

$$P_{loss} = 3\sum_{k>1}^{\infty} Ik^2 R_{stator} \tag{19}$$

where I_k = rms value of the kth harmonic current

The skin effect in the rotor bars, on the other hand, is significant, particularly if the rotor is of deep-bar construction. As a result, the rotor resistance increases with frequency. According to Alger [33] the increase in rotor resistance is approximately proportional to the square root of the frequency. As a result, the major increase in copper losses occurs in the rotor.

Kliman [34] calculates an increase of 15% to 60% in stator copper losses depending on the particular waveform applied but calculates an increase of 90% to 270% in rotor copper losses. Kliman's values are valid only for the machine and waveforms he analyzed since losses are very sensitive to the particular harmonic content of the waveform, and to the leakage reactance,

rotor resistance, and rotor bar height of the motor. However, his figures are in line with other studies which indicate that total machine losses when operating from a non-sinusoidal source can be from 1.1 to 3 times the losses when running from a sinusoidal source.

The increased losses in the inverter driven motor results in increased heating of the motor, particularly since the motors often run at reduced speed which lessens the air flow through the machine. The increased heating can result in accelerated aging of the motor insulation and thus shorten the service life of the machine. This problem can be dealt with by choosing a machine with a higher power rating than required or by applying forced air cooling but it is more desirable to control the inverter's output waveform to minimize the harmonic losses.

The output waveshape of a Variable Voltage Input inverter or a standard Current Source inverter cannot easily be changed. However, the six-step square wave supplied by these inverters usually increases machine losses by less than 20%. The main problem with the waveform of these inverters, particularly the Current Source inverter, is the production of sixth harmonic torque pulsations. Some Current Source inverters do modulate their output current at low output frequencies in order to reduce these torque pulsations.

The output waveform of a pulse width modulated (PWM) inverter, on

the other hand, can increase machine losses substantially. Fortunately, by adopting a suitable modulation technique, the harmonic content of the output waveform can be adjusted to reduce losses to an acceptable level.

A variety of modulation techniques are described in the literature. Perhaps the oldest is the "triangle intercept" or "subharmonic" technique [35]. In this technique, a sinusoidal voltage (the reference) is compared with a higher frequency triangular voltage (the carrier). The intersections of the two waveforms are used to determine the switching instants for the pulse width modulated waveform as shown in Figure 14. The

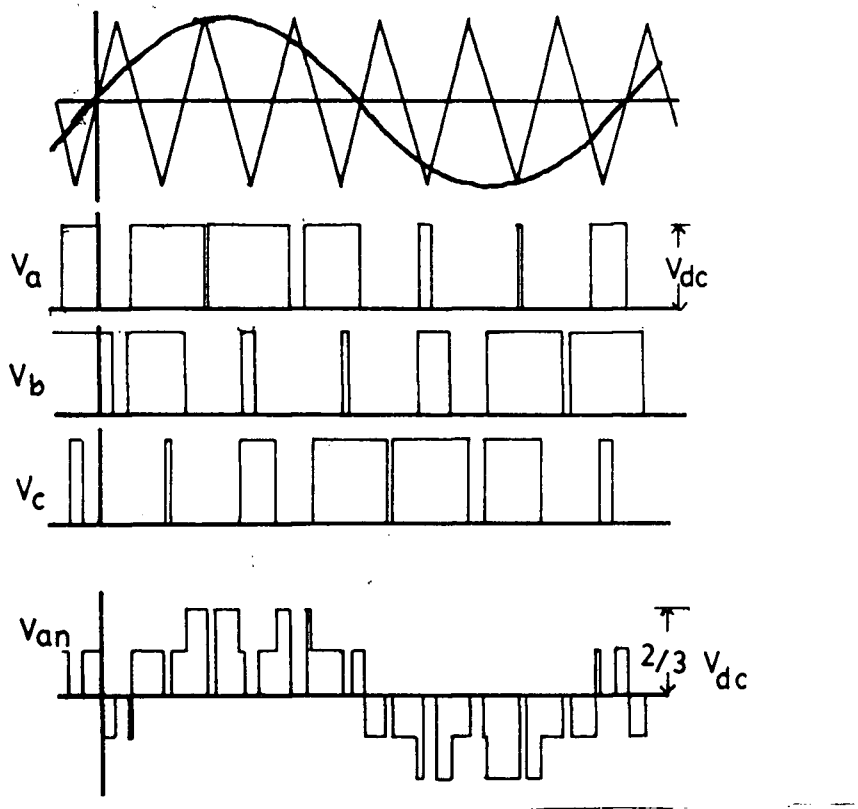


Figure 14 Triangle Intercept Pulse Width Modulation

resulting pulse width modulated waveform has a harmonic spectrum consisting of a fundamental component at the frequency of the reference waveform and harmonic components centered around multiples of the carrier frequency [36]. The amplitude and frequency of the fundamental component is changed by varying the amplitude and frequency of the reference waveform. To produce a three phase output, a three phase set of reference waveforms is compared with the carrier.

The carrier waveform is commonly synchronized to the reference to avoid subharmonic beats caused by the changing phase relationship between the carrier and reference [37]. The carrier frequency is chosen to be an odd multiple of three times the reference frequency (e.g. 21 or 27). This ensures that the modulated waveform is always symmetrical and, as long as the motor neutral isn't grounded, the harmonics at multiples of the carrier frequency won't generate any motor currents since they are triplen harmonics. In that case the harmonics occur in sidebands around the carrier frequency and its harmonics. The most important harmonics occur at:

(19) $$f_c \pm 2f_{ref}$$

(20) and $$2f_c \pm f_{ref}$$

where f_c = carrier frequency

f_{ref} = reference frequency

Harmonics also occur in sidebands around higher multiples of the carrier frequency but they do not have much effect on machine performance.

Obviously, by choosing a suitably high carrier frequency, the harmonic currents can be reduced to any desired level since the machine acts as a low pass filter. However, the switching devices in the inverter may have switching speed limitations. This is particularly true for thyristors which are usually limited to maximum switching frequencies in the range of 500 Hz to 1500 Hz. Transistors, on the other hand, can be used effectively up to 5 or 6 kHz, so PWM inverters using transistors can produce output waveforms with low current harmonics. Power MOSFETs can switch at very high frequencies and so can produce waveforms with essentially no harmonic currents. To deal with the problem of maximum switching frequency limits, multi-mode modulation strategies are often used in which the ratio between carrier and reference frequency is lowered as the reference frequency is increased so that the switching frequency is kept below the maximum limit [38].

The amplitude of the fundamental line to line voltage at full modulation (when the amplitude of the reference equals that of the carrier) is

(21) $$\sqrt{3}\, V_{DC}/2$$

where V_{DC} = inverter DC bus voltage

The magnitude of the fundamental component of the unmodulated six-step square wave is

(22) $$\sqrt{3}\ 2\ V_{DC}/\pi$$

which is 27% greater. In order to make full use of the KVA rating of the inverter it is desirable to make a transition from the fully modulated PWM wavefom to the unmodulated six-step waveform. In order to do this, pulses are merged or "dropped" until the six step waveform is reached. Some care must be taken in the manner in which pulses are dropped to ensure that sudden jumps in the fundamental do not occur. Various strategies for pulse dropping have been described in the literature [39], [40].

The triangle intercept form of modulation is well suited to analog implementation, although the implementation becomes more complex when carrier synchronization to the reference is required and the ratios between carrier and reference frequencies are changed to keep inverter switching frequency under some limit. Digital implementations of this type of modulation are also possible, either by storing pre-calculated switching patterns in memory and reading them out as required or by directly calculating the switching instants during inverter operation. However, digital implementation of pulse width modulated

controllers introduces the possibility of other modulation techniques.

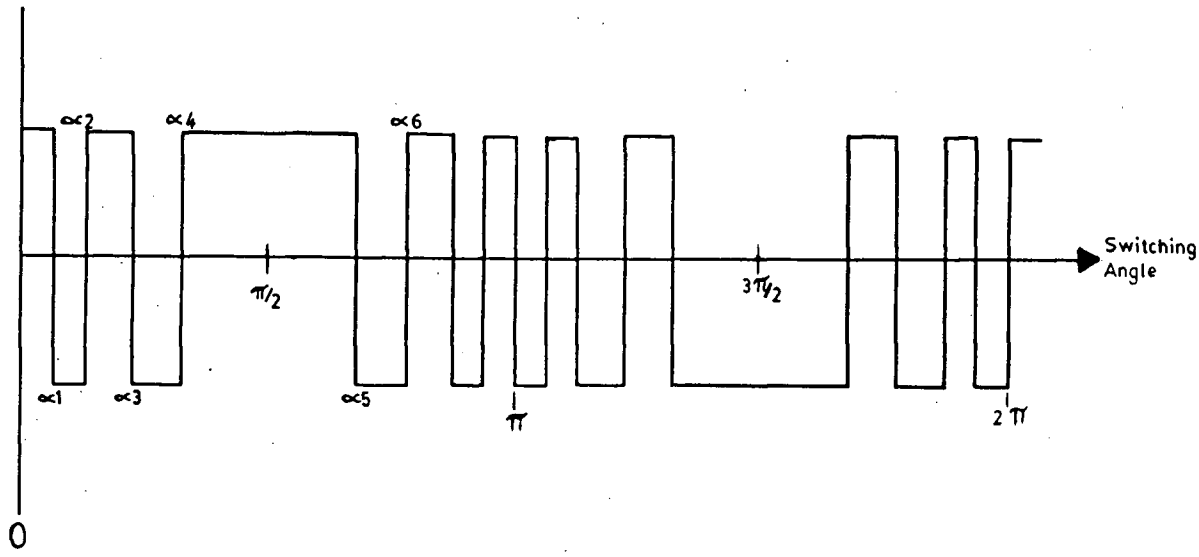


Figure 15 PWM Waveform Having Quarter and Half Wave Symmetry

The harmonic content of a pulse width modulated waveform of the type shown in Figure 15, which has quarter and half wave symmetry, is [41]:

$$a_n = \frac{2V_{DC}}{n\pi}\left[1 + 2\sum_{k=1}^{M}(-1)^k \cos n\alpha_k\right] \tag{23}$$

where a_n = amplitude of the nth harmonic

M = number of switching instants per quarter cycle

α_k = angle of the kth switching instant

Using numerical techniques it is possible to choose the M switching angles per quarter cycle to control the amplitude of the harmonics. For example, Patel and Hoft [42] calculated the switching angles required to control the amplitude of the fundamental while reducing the amplitudes of M-1 harmonics to zero. Buja and Indri [43] and others calculated the switching angles required to control the amplitude of the fundamental while minimizing a performance criterion such as

$$\sum_{n=3}^{\infty} (a_n/n)^2 \tag{24}$$

which is proportional to the total rms value of the harmonic currents in the motor. Minimizing this criterion should also minimize machine copper loss if skin effect is ignored.

Switching angles are calculated for a range of fundamental amplitudes ranging from zero to maximum and are stored in the PWM controller. The controller then looks up the set of angles for the desired inverter output voltage and uses them to generate the required switching frequency [44].

These programmed waveforms have been found to produce lower losses in the machine than the triangle intercept waveforms when the number of pulses per cycle is small [45]. Thus they are useful with inverters employing thyristors which are restricted

in the maximum switching frequency. They are also of use when the inverter must produce high fundamental frequencies (e.g. 400 Hz).

Chapter 3

Inverter Controller Design Considerations

This chapter discusses the design of the inverter controller from a high level or systems perspective. The design requirements for the inverter controller and the possible design tradeoffs are discussed in detail. This is followed by consideration of the alternative methods of implementing the controller functions and by presentation of the basic architecture of the controller. The detailed design of the inverter controller hardware and software is discussed in succeeding chapters.

The controller design is based on a microcomputer in order to give the controller a considerable amount of programmability. The division of controller functions between hardware and software was given careful attention. Some functions, such as the controller sequencing function or the control panel interface function, can be performed by software with no sacrifice in performance and with a considerable gain in flexibility. Other functions are more time critical and had to be analyzed carefully to determine a good tradeoff between hardware and software. In particular, the functions associated with pulse width modulation required close attention. Software intensive PWM generation techniques had difficulty in meeting performance requirements while more hardware oriented techniques required a large number of

components. Fortunately, a solution presented itself in the form of a large scale integrated circuit that generates three phase PWM waveforms.

Curiously, this IC, the Philips HEF4752V, has been almost entirely ignored in the literature on inverter controllers. Papers are still published describing complex PWM waveform generation circuits that appear to have no performance advantages over this IC. This thesis appears to be the first description of a microprocessor based inverter controller that makes use of this IC.

One advantage of the care taken in defining the inverter controller functions and dividing them between hardware and software is that the controller architecture can be kept quite simple without major sacrifice in performance or flexibility. As mentioned, the controller has a microcomputer at its heart. The other major components of the controller are analog and digital input ports to the microcomputer, digital output ports from the microcomputer, and the PWM waveform generation IC. With the exception of the PWM waveform generation, the controller functions are primarily performed by microcomputer software with some assistance from simple hardware components in cases where speed of response is critical.

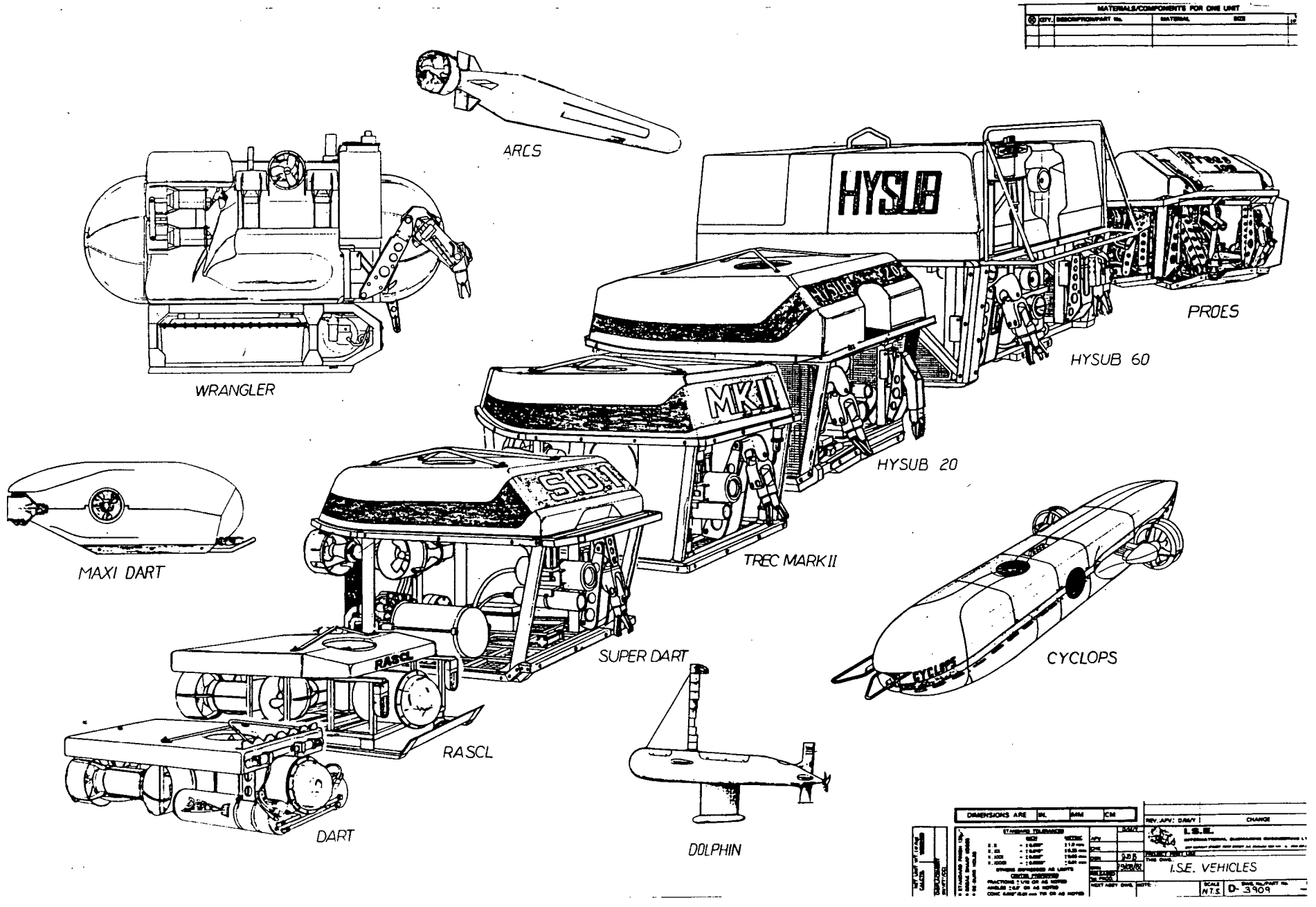


Figure 16 I.S.E. Submersibles

3.1 Basic System Requirements

The initial application of the inverter controller was to control thruster drives for submersibles produced by International Submarine Engineering. Figure 16 shows the submersibles produced by I.S.E. Some of the submersibles are autonomous but most are run with a tether that carries control signals and three phase AC power from the surface and returns telemetry from the submersible to the surface. Each submersible has up to four thrusters to allow complete control of the submersible's motion. Depending on the size of the submersible, the power rating of each thruster is from 1.5 horsepower to 5 horsepower.

International Submarine Engineering has used a variety of motors to power the thrusters. The smaller thrusters have used single phase universal motors with speed control by controlled rectification. Larger thrusters have used three phase AC motors with speed control achieved by a variable pitch propellors. The thrusters on one series of submersibles use variable speed hydraulic motors with one large AC motor running a hydraulic pump. Each of these techniques has disadvantages, either from a reliability or a cost aspect.

Variable frequency speed control of AC motors with an inverter was considered by ISE, but no suitable inverters could be found. The inverters cannot be on the surface because the cable required to carry the power to the individual thrusters would be too heavy

and too costly. Therefore, compact inverters that can withstand submersion to 1000 meters are required so that they can be installed directly on the submersible.

A development project was started in 1980, within the Department of Electrical Engineering at UBC, to develop an inverter drive to meet the requirements of the submersible application. The basic inverter configuration chosen was a PWM inverter using power MOSFETs or power transistors as switches. This results in a minimum number of components in the inverter since a single rectifier bridge can be used for all four inverters on a submersible and no commutation circuits are required. Power transistors and power MOSFETs are available in the power ratings required for this application. Each transistor or MOSFET switch is controlled by a driver circuit which interfaces to the controller via an opto-coupler.

The inverter is designed to be packaged in the same housing as the thruster motor. The entire motor housing is filled with oil to compensate for the external water pressure so the motor housing does not have to withstand any pressure.

It is doubtful that the integrated circuits and other components used in the controller will work submerged in oil. As a result, the controllers for the four inverters have to be installed in the pressure housing used to contain the other electronic circuits in the submersible. This places particularly severe

requirements on the size of the controller since space in the pressure housing is a valuable commodity. The need for high reliability and quick repair in the submersible application also encourages a design that uses a minimum number of components and is compact enough to be placed on one printed circuit card that can be swapped if a controller malfunction occurs.

In order to make use of compact high speed motors, the inverter drive must operate at frequencies up to 400 Hertz. Smooth control is required in both forward and reverse directions from the 400 Hz maximum output frequency down to a few Hertz so that the submersible can be manoeuvered in confined spaces. However, extremely rapid changes in motor speed are not required and there is no need for precise control of motor torque or speed. Therefore, an open-loop constant volts/Hz control strategy is acceptable. The inverter is designed to detect overcurrent conditions which occur if the thruster propeller becomes jammed. The inverter sends an overcurrent signal to the controller, which must respond by shutting down the inverter.

The thruster speed and direction commands are sent down to the submersible on a serial data link. Circuitry on the submersible converts the serial data to parallel data consisting of 8 bit words. The inverter controller must accept this parallel data as its control input. In addition, it is desirable to have some means of controlling the inverter with potentiometers and switches for the purposes of testing and maintenance.

To summarize, the basic requirements for the inverter controller for the submersible application are the following:

1. The controller must supply pulse width modulated switching signals for each of the six inverter switches. Each switching signal must drive the LED in an opto-coupler.

2. The controller must operate the inverter over an output frequency range of 3 to 400 Hz and be capable of reversing the phase sequence for bidirectional operation of the motor.

3. The controller must implement an open-loop constant volts/Hz control scheme for the motor drive.

4. The controller must accept a digital (TTL compatible) overcurrent signal from the inverter and shut down the inverter on receipt of the signal.

5. The controller must accept 8 bit parallel data (TTL compatible) as its speed and direction commands. It should also have provision to accept an analog voltage input as a speed command.

6. The controller must fit within the pressure housing

of an ISE submersible. This means it must consist of one or more of the 218 mm. by 77 mm. circuit cards used by ISE for its own electronic circuits. Ideally the controller should fit on one card.

7. Since the controller is installed in a sealed housing and operates only when the vehicle is submerged, non hermetic, commercial temperature range (0 C to 70 C) components may be used. It is desirable but not mandatory that the controller operate from a single 5 Volt power supply.

3.2 Requirements For Other Applications

If the inverter controller is to be used in applications other than submersible drives, some additional capabilities must be added to the controller. Deciding exactly what capabilities are to be added requires careful consideration of how the additional capability will affect the size and cost of the controller. Also, the addition of some functions requires considerable design effort which may not be warranted by the value of the additional functions. For instance, to add the capability to control Variable Voltage Input and Current Source inverters would substantially increase the complexity of the controller and increase the time required to design it. However, PWM inverters can be used in almost all inverter applications involving power

levels below 200 kW. Therefore the addition of these capabilities is not really warranted by the limited number of additional applications made possible by their inclusion.

Other capabilities can be added at less cost. For example, the ability to control inverters using thyristors as switches would be advantageous in higher power and higher voltage applications where suitable transistors or power MOSFETs are not available. To add this capability, the controller must generate separate control signals to turn on (fire) the thyristor and turn off (commutate) the thyristor. In addition the controller must insert appropriate delays so that sufficient time is allowed for thyristor commutation before the other thyristor in the same inverter leg is fired. These requirements are not difficult to meet, so this is a capability that should be added.

The submersible application places no particular demands on inverter waveshape since torque pulsations don't pose a problem and the seawater surrounding the motor acts as a good sink for the additional heat generated by the current harmonics. However, waveshaping is desirable in many other applications in order to reduce losses and torque pulsations.

Although the submersible drive can operate open-loop, other applications will require some form of closed loop control. Ideally, the controller should be designed so that it can handle high performance control strategies such as field oriented

control. The controller could then be used in servo drives for robots and machine tools. However, controllers for these high performance drives are complex and require a wide bandwidth. Examples described in the literature have required dual 8 bit microcomputers [46] or a single high speed 16 bit microprocessor [47] along with considerable support circuitry. Even with the use of a 16 bit microprocessor, the control bandwidth was only wide enough to allow closed loop operation up to a stator frequency of 100 Hz. The inclusion of a capability for servo type control of AC motors in the controller would make it too large for the submersible application and would require a major design effort. It is better to design a special purpose controller for servo applications than to attempt to squeeze servo capabilities into a general purpose inverter controller.

Fortunately, simple speed control loops or slip frequency control loops are not as demanding on controller bandwidth. They can be implemented fairly easily if the controller is based on a microprocessor. Provision must be made, however, to accept feedback signals such as stator currents or rotor speed.

In some applications, parameters such as the acceleration and deceleration rate or the volts/Hz ratio must be adjusted to match a particular load. The controller should have some provision for manual adjustment of such parameters.

The controller should be equipped with additional protective

features to prevent inverter failure due to conditions such as overcurrent or overheating. For example, when a high inertia load is decelerated, energy is regenerated back on to the DC bus of the inverter. If this energy is not dissipated or returned to the AC source, it will cause an overvoltage on the DC bus which will eventually cause failure of inverter components. To avoid this, the controller can be designed sense the bus voltage and switch in a ballast resistor to absorb the regenerated energy if the bus voltage exceeds a certain limit.

Finally, the controller must meet some cost constraints. Even though many of the applications of the controller will be specialized, economic considerations will still apply. It is difficult to set a target price for the controller since cost breakdowns for commercial inverter drives are not available. However, an estimate can be made based on the wholesale prices of some inverters and some assumptions on the division of costs within these inverters. The target price based on this estimate is $200 for parts and direct labor.

To summarize, the additional capabilities that should be added to the submersible controller to make it usable in other applications are the following:

1) The controller should be able to control thyristor inverters as well as transistor and power MOSFET inverters.

2) The controller should use a PWM strategy that reduces harmonic currents in the motor in order to reduce torque pulsations and harmonic losses.

3) The controller should provide for the implementation of simple closed-loop control schemes that do not require complex circuits or wide controller bandwidth.

4) The controller should allow for manual adjustment of parameters such as acceleration and deceleration rates, volts/Hz ratio, and maximum output frequency.

5) The controller should have provision to protect the inverter against failure due to overvoltage, overcurrent, and overheating.

6) The controller should cost no more than $200 for parts and direct labor.

3.3 Controller Functions

The basic functions to be performed by the PWM inverter controller can be grouped into the following categories:

1) System management and sequencing

2) Torque/speed control

3) PWM waveform generation

4) Control panel interface

5) Monitoring and protection

The system management and sequencing functions perform the operations required to take the drive through its various operating modes such as start up, shut down, acceleration, deceleration, and reversing. During system start up, for example, it is often necessary to turn on power supplies in the inverter in a defined order so that the inverter switches do not become active before their control signals become valid. Also, the DC bus filter capacitor may have to be precharged before the main contactor is closed to apply power to the DC bus in order to prevent large inrush currents. Similarly, shut down of the drive requires that a sequence of operations be performed.

Sequencing is also required during inverter operation. For instance, if the drive is commanded to go from full speed forward to full speed reverse, the controller must first decelerate the motor to a stop at a controlled rate, then reverse the inverter phase sequence, and then accelerate the motor at a controlled rate to the maximum speed.

The torque/speed control functions implement the overall closed loop control strategy for the drive. These functions accept feedback signals from sensors such as tachometers and they receive control setpoints from the system management functions. The difference between the setpoint and the feedback value is applied to a control algorithm. The control algorithm produces setpoint values for the inverter frequency and volts/Hz ratio which are sent to the PWM waveform generation functions.

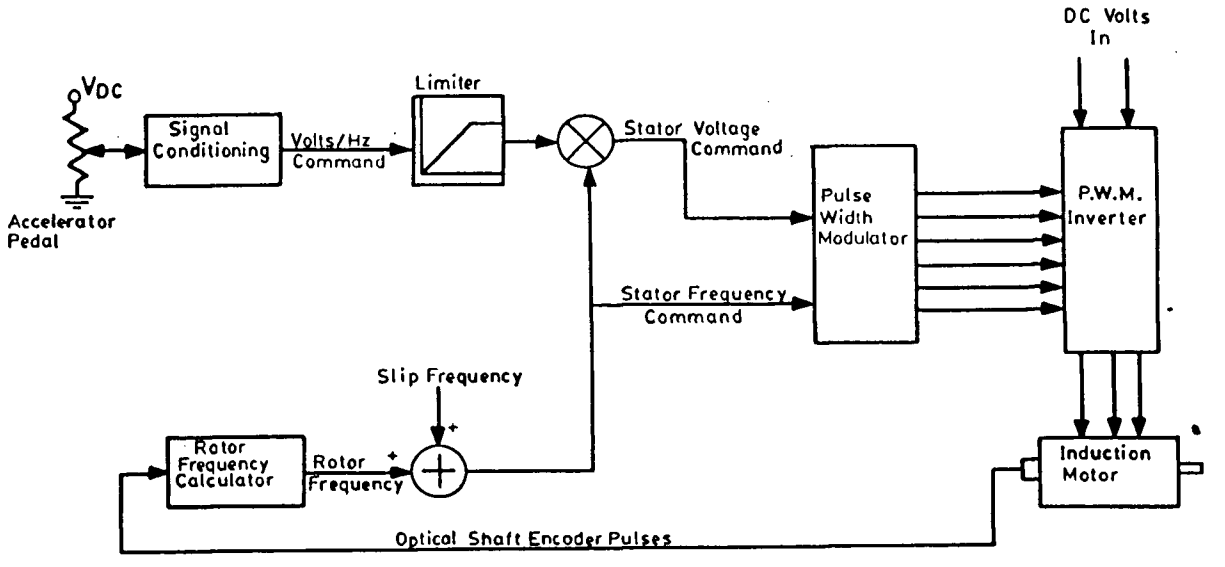


Figure 17 Control Loop for Controlled Slip Drive

As an example of a typical control loop, consider the controlled slip drive shown in block diagram form in Figure 17. This drive has been used as a traction drive in an electric vehicle [48].

The controller measures the rotor frequency using an optical shaft encoder and adds a constant slip frequency to produce the stator frequency setpoint. The controller receives a volts/Hz setpoint signal from a potentiometer attached to the vehicle accelerator pedal. The volts/Hz signal is limited so that it cannot exceed a value which would result in magnetic saturation of the motor. In this simple control loop, motor torque is controlled by the volts/Hz ratio set by the accelerator pedal. Motor speed will increase until the motor torque is balanced by the load torque.

The PWM waveform generation functions produce the control signals that cause the inverter switches to turn on and off. The functions must accept setpoints for the inverter output frequency and volts/Hz ratio and then apply a modulation technique such as the triangle intercept technique to produce an output waveform of the desired frequency and voltage. Since the controller must control six inverter switches at switching frequencies which may be several kilohertz, the PWM generation functions must be very fast.

Due to limitations of the switching devices in the inverter, the PWM functions must also ensure that certain timing requirements are met by the generated control signals. For example, the switching frequency must normally be kept below the limit at which device switching losses begin to overheat the device. At the same time it is desirable to keep the switching frequency as

close to the limit as possible so that the waveform harmonics are at the highest possible frequency. In thyristor inverters, a switch must normally be on for a certain minimum length of time in order to ensure that the commutation circuits can recharge for the next commutation. Thus the PWM generation functions must eliminate all control pulses shorter than this minimum time. Finally, the PWM generation functions must insert delays between the turning off of one switch in a leg of the inverter and the turning on of the other switch in the leg to ensure that no conduction overlap occurs.

The control panel interface functions represent those functions that communicate with the "outside world". In a conventional industrial drive they read setpoints from potentiometers and switches on a control panel and send information to indicator lamps, meters, and displays on the panel. In embedded drives such as the submersible thrusters, they handle the communication between the inverter controller and the system that is supplying the control commands.

The monitoring and protection functions monitor the inverter and take protective action if conditions occur which may cause damage to the drive. The actions must be tailored to the fault condition. For example, if a large and rapidly increasing overcurrent, indicative of a short circuit, is detected, the inverter must be immediately shut down. On the other hand, if a moderate overcurrent is detected when the drive is accelerating,

indicative of too rapid acceleration, the appropriate action is to reduce the rate of acceleration.

The monitoring and protection functions may also perform self checks on the controller and perform diagnostics in the event of a drive fault. For instance, if the inverter shuts down, the diagnostic functions will indicate what conditions caused the shutdown on a control panel display.

3.4 Inverter Controller Design Alternatives

The controllers in many inverters currently in production are based on a mix of CMOS logic circuits and analog integrated circuits. For the past several years however, research and development has centred on microprocessor based designs. Some of these new microprocessor based controllers are beginning to appear in production [49].

Microprocessor based controllers are attractive because they allow more reliable and flexible controllers to be built at a lower cost. The increased reliability comes from the substitution of LSI components for the small scale integrated circuits used previously. The reduction in component count and number of interconnections results in a more reliable system. The increase in flexibility comes from the ability to change controller characteristics by altering the microprocessor software rather than making modifications to hardware. Since a microprocessor

based controller uses less components than a controller based on small scale integrated circuits, it is normally considerably cheaper to manufacture. However, the cost of software development must be considered when comparing the cost of a microprocessor based controller to the cost of a controller based strictly on hardware.

Since the design requirements for our project emphasized a small, flexible and reliable inverter controller, one of the first design decisions was to base the controller on a microcomputer. With this decision made, it is necessary to partition the controller functions between those functions that are to be carried out entirely by software and those functions that are to be carried out by auxiliary hardware under control of the microcomputer. Some functions such as the system management and sequencing functions and the control panel interface functions are well suited to a software implementation. Others, such as the torque/speed control functions, the system protection functions, and the PWM waveform generation functions, require a closer examination of the tradeoffs between software and hardware implementation.

3.5 Torque and Speed Control

Torque/speed control functions have traditionally been implemented as analog feedback control loops with operational amplifier circuits providing proportional or proportional plus integral

control. These circuits are simple and have more than enough bandwidth for motor control applications. Moreover, an analog implementation is inherently parallel; multiple feedback loops can be closed or feedback signals can be filtered simply by adding more components. However, analog circuits tend to drift with time and temperature. Also, they are designed to implement only one control scheme. Adopting a new control scheme requires redesign and rewiring of the circuit. A software based control loop, using digital control algorithms, is drift free and can be changed simply by reprogramming. However it is difficult to duplicate the bandwidth and inherent parallelism of the analog circuit with algorithms running on a single processor.

Fortunately the bandwidth requirements of many motor control loops are fairly low. For example, as discussed in the previous chapter, the open loop speed response of a loaded induction motor is dominated by a single electromechanical time constant. This time constant is on the order of 20 to 60 milliseconds for a small (3 to 15 horsepower), unloaded machine [50] and will increase to several times this value when the machine is loaded. Assuming that the closed loop bandwidth of a speed control loop doesn't exceed the motor's open loop bandwidth, the loop bandwidth will be on the order of 5 to 10 Hertz. For good response, sample rates for digital control loops are usually set at 10 to 20 times the loop bandwidth [51, 52]. Thus a sample interval of 5 to 20 milliseconds can be expected for a speed control loop. This is not beyond the capabilities of a

microprocessor, particularly if the control algorithm is kept simple.

In our case, the inverter controller is intended for use in a variety of applications. It is difficult to predict exactly the control requirements of each potential application. Therefore, the flexibility inherent in the software based control loop is very attractive. As a result it was decided that the basic controller would contain no analog control loops. Instead, an analog to digital converter would be included to accept analog feedback and setpoint signals and the feedback control would be carried out by programs in the microprocessor.

3.6 System Protection

Whether a particular system protection function can be handled by the microprocessor or must be performed by external circuitry is determined by the nature of the fault condition the function is supposed to deal with and by the design of the inverter. Common fault conditions that the protection functions must deal with are the following:

a) Inverter shoot through or motor short circuit. An inverter shoot through occurs when both switching devices in one leg of the inverter are on simultaneously. This can occur because of a defective device or incorrect switching signals. A shoot through or motor short circuit is characterized by a very rapid rise in

current through one or more of the switching devices in the inverter. The magnitude of the current is limited only by the impedance of the wiring.

b) Open motor phase. If a motor phase is opened during operation, the current in the other phases increases and thus some of the inverter switches will carry increased current. The rate of rise and magnitude of the fault current are not as large as for a short circuit.

c) Motor overload. The motor can draw a current greater than the inverter rating if the motor is loaded beyond the drive's rating or if the motor acceleration or deceleration rate is too high. The rate of rise of current is limited by the speed that the motor slip frequency changes.

d) Bus overvoltage on regeneration. When inertial loads are decelerated, the kinetic energy of the load is regenerated back on to the DC bus of the inverter. This energy must either be dissipated or fed back to the AC bus. Otherwise the energy will simply go into charging the bus filter capacitor until the bus voltage is high enough to cause a breakdown in some component.

e) Inverter overheating. Improper installation of the inverter or a failure in the cooling system can cause the inverter to overheat.

The normal protective action when a shoot through, motor short, or open motor phase occurs, is to turn the inverter switches off as rapidly as possible. If the fault currents have a short rise time and the inverter switches are transistors or power MOSFETs, the inverter must be switched off in a few microseconds in order to prevent damage to the switching devices. Even with the use of an interrupt input it is difficult for a microprocessor to respond this quickly. Therefore external protective circuits are required to deal with these faults. However, a signal must be sent to the microprocessor, informing it of the occurrence of the fault.

In the case of a motor overload, the speed of response does not have to be as fast since the rate of rise of fault current is lower and the magnitude of the fault current is lower. The required response depends on the exact nature of the fault. In this case the microprocessor can monitor the motor line current and take the appropriate corrective action. Alternatively, if the monitoring of motor current consumes too much processor time, external circuits can be used to compare motor current to a limit and send an interrupt signal to the microprocessor if the limit is exceeded.

When an AC drive is regenerating, the rate of rise of voltage on the DC bus is limited by the bus filter capacitor. The bus filter capacitor is normally quite large and there is normally a substantial safety margin on the bus voltage. As a result there

is ample time for the microprocessor to take protective action once it detects that the bus voltage has exceeded a limit. Again, an external circuit that interrupts the microprocessor when the bus voltage exceeds the limit can be used if the continuous monitoring of bus voltage places too much of a demand on processor time.

Overheating occurs very slowly when compared to the time span over which the other fault conditions can occur. As a result, the microprocessor can handle the protection function quite easily by sampling the inverter temperature once every few seconds and turning the inverter off if the temperature exceeds a maximum limit.

3.7 Pulse Width Modulation

The pulse width modulation functions present more of a challenge to the microprocessor. Consider the software intensive pulsewidth modulation scheme shown in Figure 18. The microcomputer programs the programmable timer so that it generates an output with a frequency 360 times the output frequency of the inverter. The programmable timer therefore interrupts the microcomputer 360 times per cycle of the inverter output waveform. Each time the microcomputer is interrupted, it sends the switching pattern for the next interval of the inverter output waveform to the parallel output port. The resolution of this PWM scheme is one degree since all pulses must be multiples

of one degree (1/360th of a cycle) long. This is a relatively coarse resolution for a PWM waveform. Nevertheless, the demands on the microprocessor are quite severe. At an inverter frequency of 60 Hertz, the processor is interrupted every 46 microseconds. Within the 46 microsecond interval the microprocessor must service the interrupt, send the new switching pattern to the output port, and determine the switching pattern for the next interval. In addition, some time must be reserved for the other functions that the inverter controller must perform. Casteel and Hoft [53] found that an Intel 8080 based controller using a PWM scheme similar to the one just described would only operate to 55 Hertz despite the fact that all the switching waveforms were pre-calculated and stored in memory and the controller performed no functions other than PWM waveform generation.

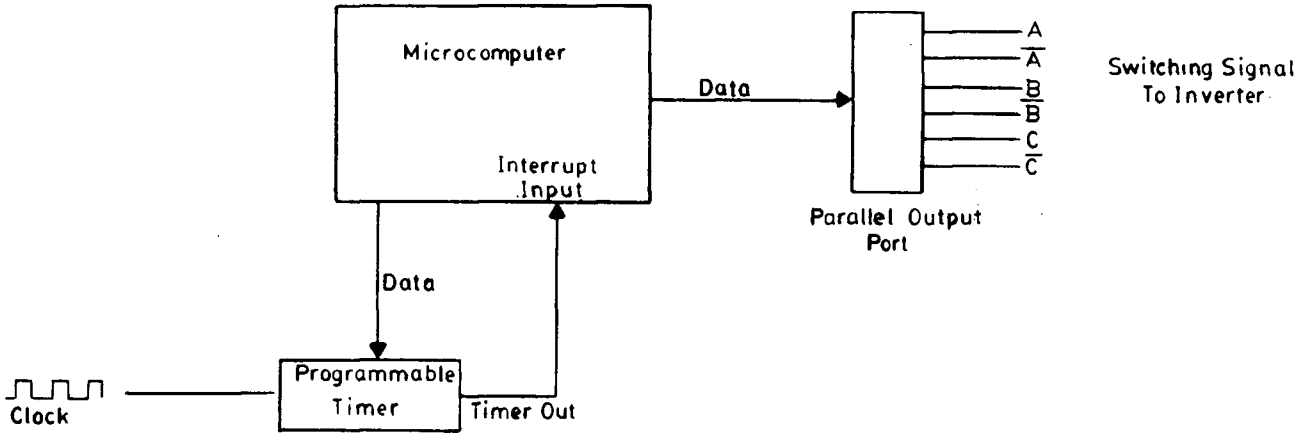


Figure 18 Software Intensive PWM Waveform Generator

Obviously some of the PWM waveform generation must be performed by auxiliary hardware, particularly since inverter output frequencies up to 400 Hertz are required. One approach is to make use of programmable timers to generate the PWM pulses. The block diagram for one such system is shown in Figure 19.

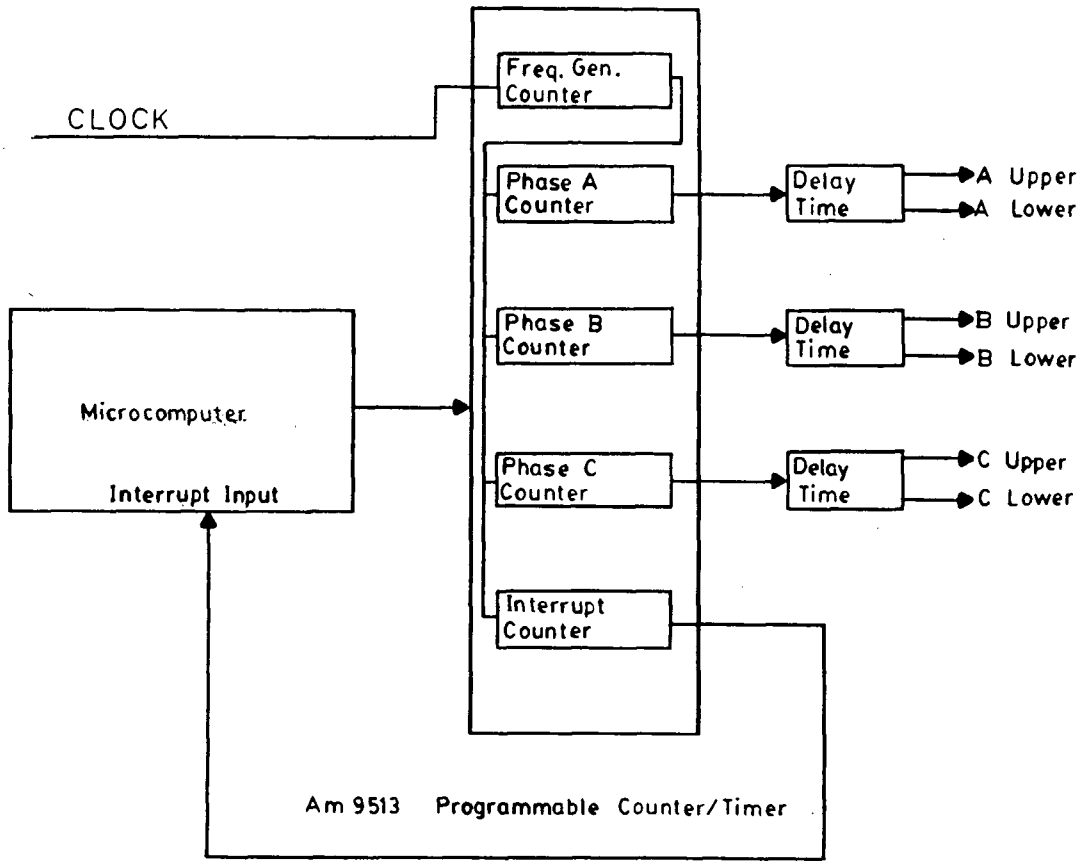


Figure 19 Counter/Timer Based PWM Waveform Generator

The counter/timer IC used in this design is the Advanced Micro Devices Am 9513. The counters in this device can be set up so that each counter is loaded alternately from one of two registers upon reaching terminal count. In addition the counter output line toggles at each terminal count. This allows the counters to be used as pulse width modulators. One register is loaded with the

"on time" of the pulse and the other register is loaded with the "off time", producing PWM waveforms such as those shown in Figure 20.

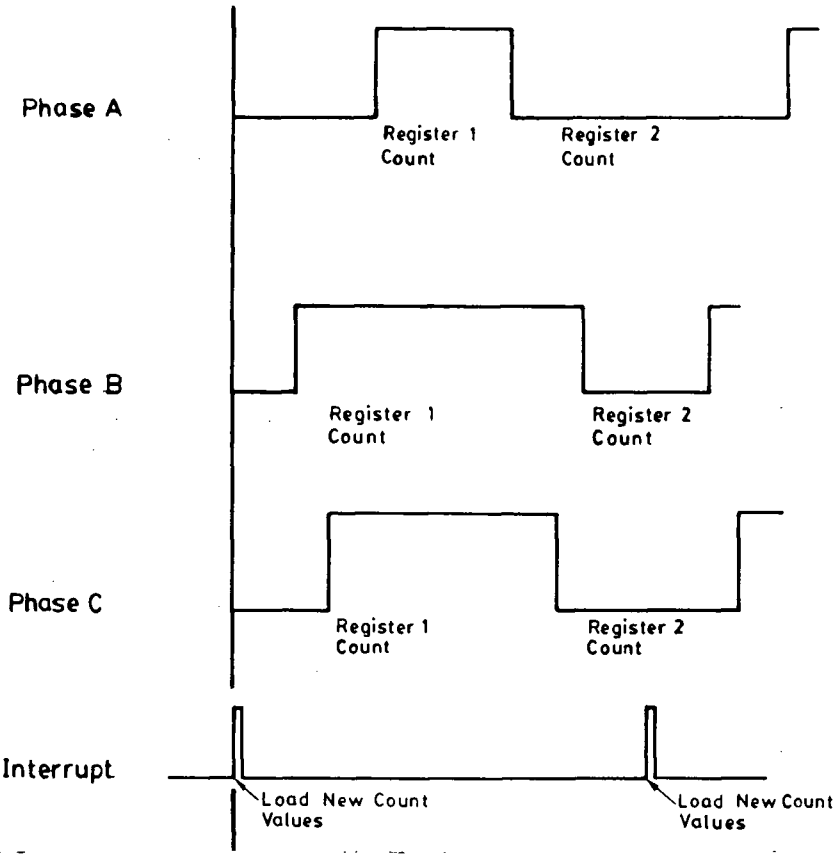


Figure 20 Programmable Counter/Timer Output Waveforms

In order to vary the duty cycle of the PWM waveform over a cycle of the the fundamental frequency, the microprocessor is interrupted by the interrupt counter which generates pulses at the carrier frequency of the PWM waveform. On each interrupt, the microprocessor loads new duty cycle values into the counter registers. The inverter output frequency is determined by the Frequency Generator counter which is programmed to supply a clock waveform which has a frequency which is a multiple of the inverter output frequency. This clock frequency is then counted

by the Phase Counters which generate the PWM signals and the Interupt Counter which generates the processor interrupts. The PWM signals generated by the counter/timer IC have only two states which means that one switch in an inverter leg must always be on. As a result, extra circuits must be added to insert the dead time required between the turning off of one switch in an inverter leg and the turning on of the other switch.

In this PWM generation technique, the microprocessor must respond at a rate determined by the carrier frequency of the PWM waveform. In thyristor based inverters, the carrier frequency is usually kept below one kilohertz and the processor has a reasonable amount of time between interrupts to perform other tasks. Inverters using transistors or power MOSFETs as switches, on the other hand, may have maximum switching frequencies of 5 kilohertz or more. In this case the time between interrupts which can be devoted to other tasks becomes rather short. Despite this problem, several modulators using programmable timers in conjunction with a microprocessor have been described in the literature [54, 55, 56]. For most of these designs, the inverter switching frequency is limited to a maximum of less than two kilohertz. In addition, many of them are designed strictly as PWM modulators since there isn't enough processing time available to carry out the other functions of an inverter controller. In that case, the controller becomes a multimicroprocessor system with one microprocessor dedicated to the generation of the PWM signals and one or more other microprocesors dedicated to the

other controller functions [57]. Such an approach can deliver excellent performance but is likely to be too large and complex for the submersible drive application.

An alternative PWM scheme is shown in Figure 21. In this case the switching waveform for an entire cycle is stored in read/write memory as a bit pattern. This bit pattern is then read out of memory to the inverter switches at a rate proportional to the desired inverter output frequency. By providing two banks of memory, one bank can be generating the inverter switching signals while the other bank is being loaded with a new waveform by the microprocessor.

This approach has the advantage that the microprocessor only has to update the waveform when a change in inverter output voltage is required. Thus, when the inverter is running at constant frequency, the processor is free to carry out other tasks. The microprocessor must generate new waveforms when the inverter is changing frequency in order to maintain the volts/Hz ratio but there is no absolute requirement on processor response time. Slow microprocessor response simply results in a low slew rate for inverter output frequency.

The maximum possible inverter output frequency is determined by the length of the waveform storage memory and by its access time. If we assume the memory is 2048 locations long and the minimum time between accesses to successive locations is 500 nanoseconds,

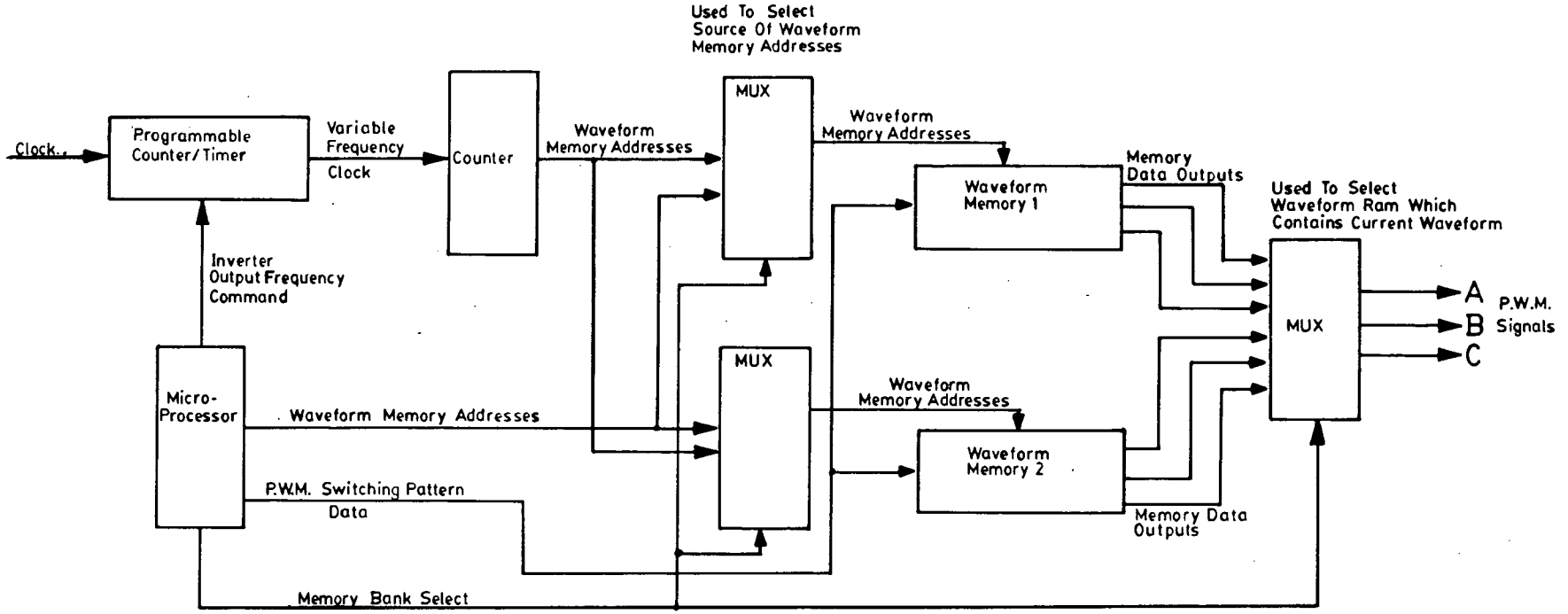


Figure 21 Memory Intensive PWM
Waveform Generator

then the minimum length of one inverter output cycle is 1024 microseconds. This corresponds to an inverter output frequency of approximately one kilohertz.

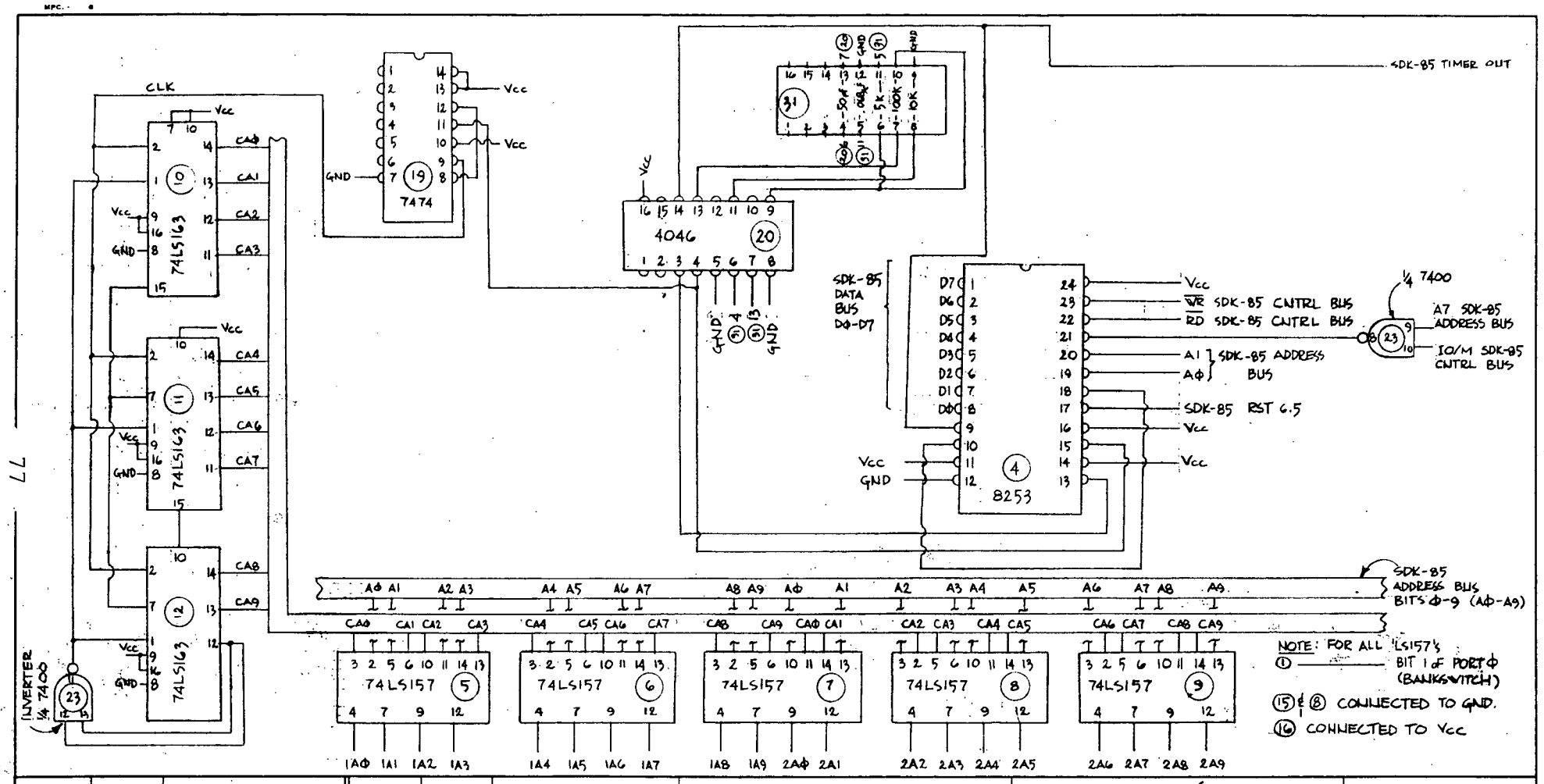
The length of the the waveform storage memory determines the resolution possible in the inverter switching intervals. For example, a memory with 2048 locations provides a resolution of 0.176 degrees in the inverter switching angles and allows a minimum pulsewidth that is approximately .0005 of the period of the fundamental. This kind of resolution is more than adequate over most of the inverter operating range but, surprisingly, may not be good enough at low output frequencies. Consider the case where the inverter output frequency is 5 Hertz and the desired switching (carrier) frequency is 1000 Hertz. There will be 200 pulses per cycle of the inverter output waveform. Each pulse can only take on one of ten (2048/200) discrete widths. This only allows coarse control of the output voltage. By using memories with short access times, the length of the waveform storage memory can be increased substantially (up to 16384 locations) while still allowing inverter operation to 400 or 500 Hertz. This alleviates the resolution problem at low inverter output frequencies but also increases the time required to load a new waveform into the waveform memory.

A number of PWM controllers based on some variation of the basic scheme just discussed have been described [58, 59, 60]. Probably the principal drawback to all of them is the large number of

components required to implement this technique.

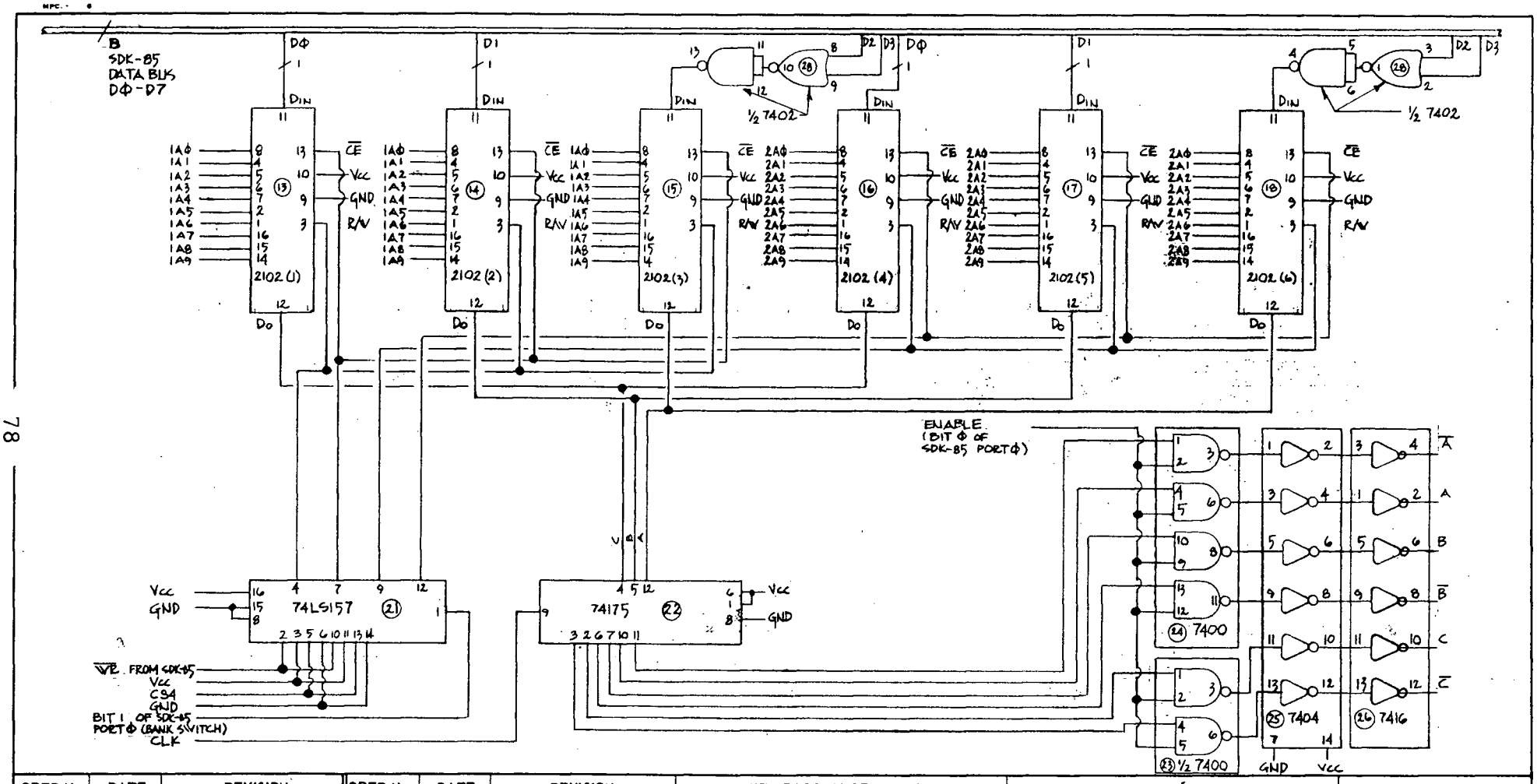
At an early stage in the development of the controller for the submersible drive, a prototype PWM waveform generator of the type just described was constructed in order to gain some familiarity with this technique and to evaluate its suitability for the submersible drive application. The prototype was constructed on an Intel SDK-85 microcomputer evaluation board which uses the Intel 8085 microprocessor. Schematics for the PWM generator portions of the circuit are shown in Figures 22 and 23. The two waveform memory banks each consist of three 1K x 1 RAMs organized as 1024 locations containing three bits each. The memory length is too short for good resolution at low inverter output frequencies but was considered adequate for initial experimentation.

The RAM bank that is currently supplying the inverter switching signals is addressed by a 10 bit counter formed from 74LS163 binary counters. The counter is clocked by a frequency synthesizer circuit consisting of a CD4046 phase locked loop and an Intel 8253 counter/timer. A frequency synthesizer is used so that the output frequency can be changed in small fixed size increments. In this design the increment was 0.5 Hz. Using a simple programmable counter as a frequency generator would result in poor frequency resolution at higher output frequencies where the counter is loaded with small divisors.



OPER.No.	DATE	REVISION	OPER.No.	DATE	REVISION	KEY PROCESS OPERATION			PREPARED BY ARCHIDE	DATE	ORIG. ISSUE
	MARCH 17					SEQUENCE	TOOLING	RELATIONSHIP	APPROVED BY	DATE	SHEET 2 OF 3
	MAY 14/81	REV. PER K. MAUCH R.A.							OPERATION		OPER. NO.
						ELEMENTS	ELEMENTS	ELEMENTS	FREQUENCY SYNTHESIZER & ADDRESS GENERATION		

Figure 22 Prototype PWM Waveform Generator
Part 1



OPER.No.	DATE	REVISION	OPER.No.	DATE	REVISION	KEY PROCESS OPERATION			PREPARED BY ARCHIDÉ	DATE	ORIG. ISSUE
	MARCH 17	GENERAL REV.				SEQUENCE	TOOLING	RELATIONSHIP	APPROVED BY	DATE	SHEET 3 OF 3
	MAY 21	REV. PER K.MAUCH R.A.							OPERATION		OPER. NO.
						ELEMENTS	ELEMENTS	ELEMENTS	WAVEFORM RAMS AND OUTPUT		

Figure 23 Prototype PWM Waveform Generator
Part 2

The 8085 microprocessor selects the RAM bank that is currently supplying the inverter signals with the BANKSWITCH signal. This signal controls a set of 74LS157 multiplexers that apply the counter outputs to the currently active RAM bank's address lines and apply the 8085 address lines to the other RAM bank. Another 74LS157 applies appropriate control signals to the RAM banks.

The RAM bank that is not currently supplying the inverter switching signals is set up so that the microprocessor can write data to it. The microprcessor loads the RAM bank with the next switching pattern to be generated. In this design, the PWM technique employed is the harmonic elimination technique developed by Patel and Hoft. The required switching angles to eliminate the 5th, 7th, and 11th harmonics while controlling the amplitude of the fundamental at 5% intervals from 0 to 100% are listed in Patel's thesis [61]. These values were used to develop switching patterns for the first 90 degrees of the three phase output waveform. These 20 patterns were then stored in EPROM. Due to the quarter and half wave symmetry of the PWM switching patterns, it is easy to generate the switching patterns for the remaining 270 degrees of the three phase waveform from the stored switching pattern for the first 90 degrees. When the direction of rotation must be reversed, the phase sequence of the generated three phase waveform is reversed by loading the switching pattern into RAM in descending order from the highest RAM address to the lowest RAM address.

The PWM modulator was tested with both transistor and power MOSFET inverters. On the whole the performance was satisfactory. However, there were a few problems. At low frequencies there were high peak currents in the inverter switches. This was due to the fact that the PWM technique chosen only eliminated the 5th, 7th, and 11th harmonics and exercised no control over the remaining higher order harmonics. This problem can be resolved by adopting a PWM technique that controls more of the harmonics at low output frequencies. For example, the harmonic elimination technique could be extended to eliminate the 5th, 7th, 11th, 13th, 17th, and 19th harmonics. This would probably require a longer waveforn memory in order to allow for the additional resolution required in the switching angles.

A less tractable problem is the amount of circuitry required to implement this PWM generator. The equivalent (in board space) of approximately 30 16 pin ICs were required for the PWM generator. This does not include the microprocessor and its associated components such as program storage EPROM, RAM, and bus drivers. After adding these components and the components required to support the other inverter controller functions, the total chip count would probably approach the equivalent of 50 16 pin ICs. A controller with this many components would require two or three of the 218 mm. by 77 mm. circuit cards that are to be used in the submersible.

The ideal way to reduce the chip count is to integrate the PWM

generator on to a single chip. At the time that the design of the inverter controller was started (Spring of 1981), facilities were not available at UBC for the design and fabrication of an integrated circuit of the required complexity. PWM circuits that were commercially available were designed for switching power supplies and were completely unsuitable for three phase inverter applications. It appeared that a relatively high component count for the PWM generator would have to be accepted.

Fortunately, just as the design of an improved version of the PWM generator just described was beginning, a PWM generator IC designed specifically for AC motor drive applications was announced. The HEF4752 AC Motor Controller IC [62] is manufactured by Philips and is distributed in North America by Signetics. Samples of this device and all the available documentation on the device were obtained in order to assess its suitability for our application.

The device has the following basic features:

a) It uses a "regular sampling" PWM technique [63]. This produces a waveform and harmonic content similar to that produced by the triangle intercept PWM technique.

b) The carrier frequency to fundamental frequency ratio is changed in integer steps at discrete points in the output frequency range in order to keep the carrier frequency between a minimum and

maximum value. This prevents both very low carrier frequencies which would result in increased harmonic currents and very high carrier frequencies which would result in increased inverter switching losses.

c) Signals can be generated to control thyristor inverters as well as transistor and power MOSFET inverters.

d) Inputs are availabe to control the inverter output frequency, the volts/Hz ratio, the maximum carrier frequency, the phase sequence of the three phase waveform, and the delay between the switching of the upper and lower devices in an inverter leg.

The maximum inverter output frequency that the device can produce is specified as 200 Hz in the documentation. However, in tests on samples from two different production lots, all devices worked to output frequencies well above 400 Hz.

On the whole the HEF4752 seems to meet our needs admirably. There are only two drawbacks to the use of the device. First, one must accept the PWM strategy chosen by the designers of the IC since the device is not programmable. However, the technique used gives very good results so this is only a disadvantage in research applications where the effects of different modulation techniques are being evaluated. Since the inverter controller is designed

primarily for industrial use, this is not a severe handicap in our case.

The second problem is that the control signals for the inverter output frequency, volts/Hz ratio, and maximum carrier frequency are frequencies. For use in a microprocessor based system it would be better if the device had a parallel digital input instead which would accept binary numbers as control inputs. Such a device has been designed by Hitachi [64] but it is not available on the open market. However, digitally controlled frequency generators are relatively easy to design so this interfacing problem is not a major obstacle to the use of the HEF4752.

Since the use of the HEF4752 PWM generator IC results in a substantial reduction in the chip count of the inverter controller and also reduces the software design effort and since the use of the device presents no significant disadvantages, it was decided to base the PWM generation portion of the inverter controller on this device. The primary risk with this decision is that the device is available only from a single source. If the device were to be discontinued or if the design of the device was changed so that it no longer operated to 400 Hz, the inverter controller would have to be completely redesigned. Since the applications of this inverter controller are in specialized, low volume applications it is possible to buy enough of the devices in advance to ensure a sufficient supply for several years of

production.

3.8 Basic Controller Architecture

With the basic tradeoffs made on what controller functions will be performed by software and what functions will be performed by dedicated hardware, it is possible to develop a system architecture for the inverter controller. This architecture is shown in Figure 24. The basic components of the system are:

a) Microcomputer. This is the heart of the controller and consists of the microprocessor, its clock circuit, RAM, EPROM, and any required buffer circuits. It also includes interrupt inputs for a real-time clock and for fault signals from the inverter.

b) Real-time clock. This interrupts the microcomputer at regular intervals. It is used in controller functions that need timing information. For example,when the inverter is started, the DC bus capacitor must be allowed to charge through a resistor for a defined time before the main contactor applies power to the DC bus.

c) A/D converter. This is used to accept inputs such as control setpoints and signals from the inverter and motor.

d)Parallel input port. This can be used either as an input for

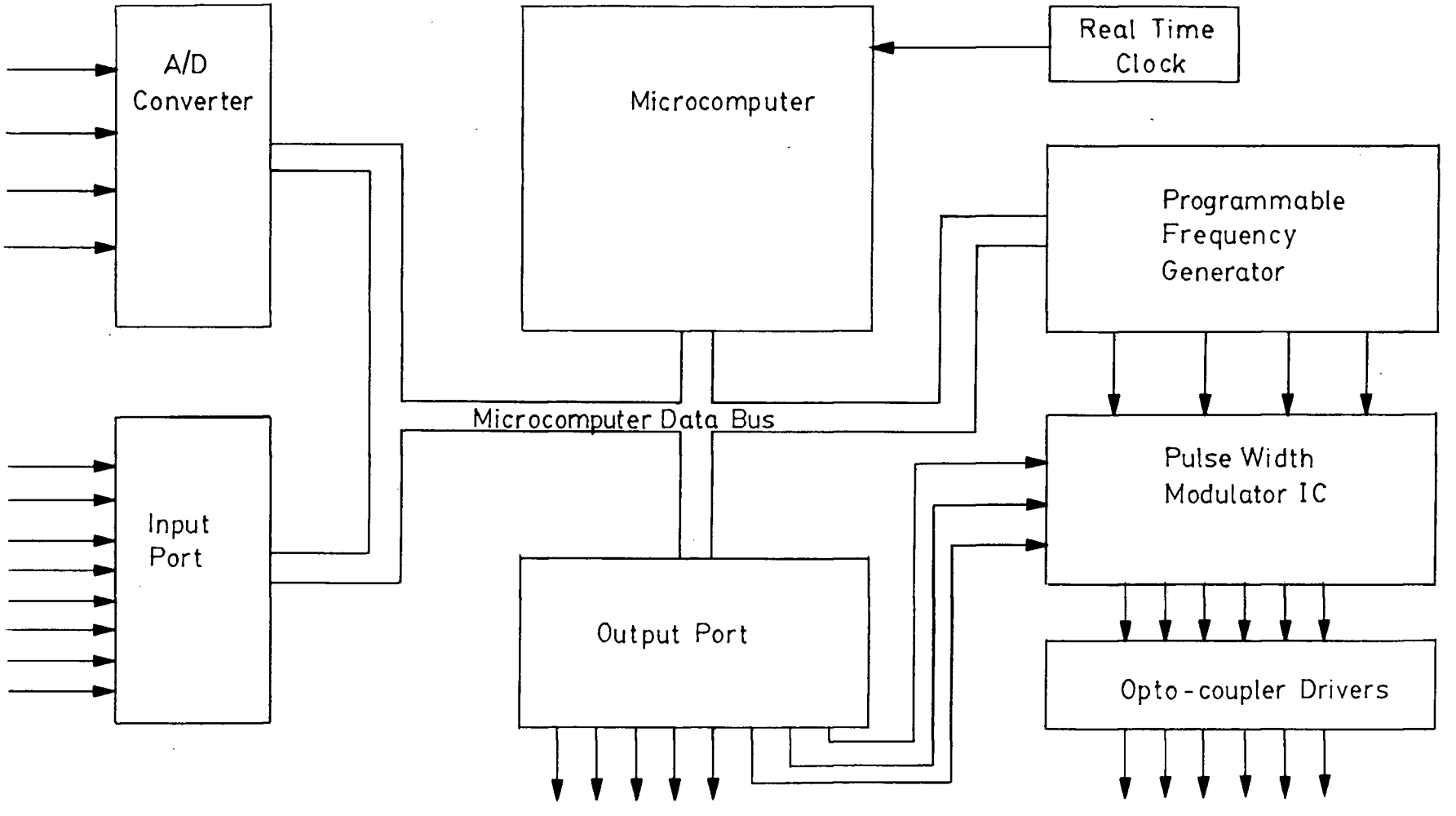


Figure 24 Inverter Controller Architecture

parallel binary data from a higher level controller or as inputs for switches and other individual binary signals.

e) Parallel output port. This is used as an output for control signals for the PWM generator IC and for inverter components such as contactors.

f) Programmable frequency generator. This supplies the command frequencies for inverter output frequency, volts/Hz ratio, and maximum device switching frequency to the PWM generator IC under control of the microcomputer.

g) Pulse Width Modulator IC. The Phillips HEF4752 generates the inverter switching waveforms under the control of the microcomputer.

h) Opto-coupler drivers. These supply the current to drive the opto-couplers on the circuits that control the inverter switching devices.

Given this basic architecture, it is now possible to discuss in detail the design of the inverter controller hardware.

Chapter 4

Controller Hardware Design

With the basic architecture of the inverter controller defined, and the decisions made on implementation of controller functions, the next step was the detailed design of the controller circuit. The design was carried out with close attention to the number of integrated circuits required. The goal was to obtain acceptable performance from a minimum number of ICs so that the controller would be reliable, compact, and inexpensive

The goal was achieved through use of large scale integrated circuits. The microcomputer in the controller consists of an Intel 8085 microprocessor, a 4 kbyte EPROM memory, and an Intel 8156 IC that provides read/write memory and I/O lines. The microcomputer is interfaced to an ADC0808 analog to digital converter IC that provides eight analog input lines to the controller. The microcomputer is also interfaced to the HEF4752V PWM generator IC that actually generates the inverter switching signals.

The controller is very compact, fitting on a single 7.5 cm by 21.5 cm printed circuit board. Only 18 integrated circuits are required, resulting in an estimated parts cost (including the circuit board) of under $160 when the parts are purchased in

small quantities. The direct labor cost to assemble the controller board is estimated to be under $25. Published descriptions of inverter controllers rarely include information on the size or cost of the design. However, based on what information is available, it is apparent that the controller described in this thesis is among the most compact and least expensive.

Despite the small size and low cost of the controller hardware, the performance and flexibility of the controller are more than adequate. The controller can control PWM inverters using transistors, power MOSFETs, or thyristors as switches over a frequency range of at least 2 Hz to 400 Hz with a resolution of 0.5 Hz. The volts/Hz ratio can be adjusted over at least a 10 to 1 range. Analog and digital inputs are available for control and feedback signals. Four interrupt inputs are available for fault signals from the inverter. Nine digital output lines are available to control inverter and control panel components. Finally, there is provision in the design to expand the controller since the microcomputer data lines, control lines, and low order address lines are brought to connectors on the edge of the printed circuit board.

4.1 Microcomputer Circuit

This stage started with the selection of the microcomputer components. Three factors influenced the choice of

microprocessor. In ascending order of importance, they are:

i) the speed and computational power of the microprocessor

ii) the number and size of chips required for a complete microcomputer system

iii) the availability of low cost software and hardware development tools

The faster and more powerful the microprocessor, the more the controller can do before serious degradation of response time occurs. Computational power is most important when the controller is performing closed loop control. Control algorithms make heavy use of multiply and divide operations. If the time required for these operations can be decreased, either the control loop bandwidth or the complexity of the control algorithm can be increased.

Minimizing the number and size of chips required to make up a complete microcomputer is important because of the size constraints imposed by the submersible application. A single chip microcomputer is the best choice in this regard. A microprocessor that can function with a few highly integrated support chips makes an acceptable second choice.

The availability of good, low cost software and hardware development tools for the microprocessor is important for two reasons. First, the budget for the development of the inverter controller was not large enough to allow the purchase of expensive microcomputer development systems. Development had to be performed with equipment already available at UBC or with equipment that could be purchased at low cost. Second, once the design was complete, it was to be transferred to a small company which would manufacture submersible drives and other AC motor drives based on this controller. Since one of the virtues of the controller is its programmability, the company also has to have software development facilities for the microprocessor. Again, money was not available for the company to purchase an expensive software development system.

The combination of the three factors narrowed the list of potential microprocessors considerably. The 16 bit microprocessors readily available at the time (the latter half of 1981) required too many support chips and, with the exception of the TI 9900, were not supported by the microcomputer development facilities at UBC. The situation has changed. Newer microprocessors such as the Intel 80188 have most of their support circuits integrated on the microprocessor chip. In addition, inexpensive personal computers can be used to develop software for the 80188 since many of them use the software compatible Intel 8088 microprocessor.

Single chip microcomputers also had to be eliminated due to the lack of low cost development facilities. Again this situation has changed. Cross-assemblers which run on personal computers are now available for several single chip computers.

Of the microprocessors for which hardware and software development facilities were available at UBC, the Intel 8085 microprocessor was the most attractive for this application. The 8085 basically has the same instruction set as the Intel 8080 microprocessor. A large amount of development software is available for the 8080 family of microprocessors. At UBC, assemblers and a compiler for the PL/M programming language are available. Relocating macro-assemblers and compilers for the C, Pascal, Forth, Fortran, and Basic programming languages are available on personal computers that use the popular CP/M operating system.

On the hardware side, the 8085 has support chips available which allow the construction of a small microcomputer system with I/O ports, EPROM and RAM memory, and a programmable timer, using only three or four ICs. For an 8 bit microprocessor, the 8085 has an unusually powerful interrupt system which has 5 interrupt inputs and on-chip priority arbitration. This is a useful feature since the design of the inverter protection functions calls for the use of interrupt signals generated by fault detection circuits in the inverter.

A final advantage of the 8085 was familiarity. I had used the 8085 in other microprocessor based systems and was familiar with its assembly language and hardware features. Therefore the design time on this project could be devoted to tasks other than learning about a new microprocessor.

The major drawback to the use of the 8085 microprocessor is its relatively limited processing power. The 8085 is not as powerful as the new 16 bit microprocessors or some of the new single chip microcomputers such as the Intel 8051. Its most notable deficiency is the lack of multiply and divide instructions in its instruction set. However, previous experience with the processor and some preliminary estimates of the processing requirements for the inverter controller application suggested that the processing power of the 8085 would be adequate. This has proved to be true for the applications in which the inverter controller has been used to date.

With the microprocessor chosen, design of the remainder of the microcomputer section of the controller could proceed. The schematic diagram for the microcomputer section is shown in Figure 25.

The 8085 has internal clock generation circuits so only a crystal and two small capacitors are required to complete the clock generator. All versions of the inverter controller built to date have used an 8085 running at a 3 MHz clock frequency. However it

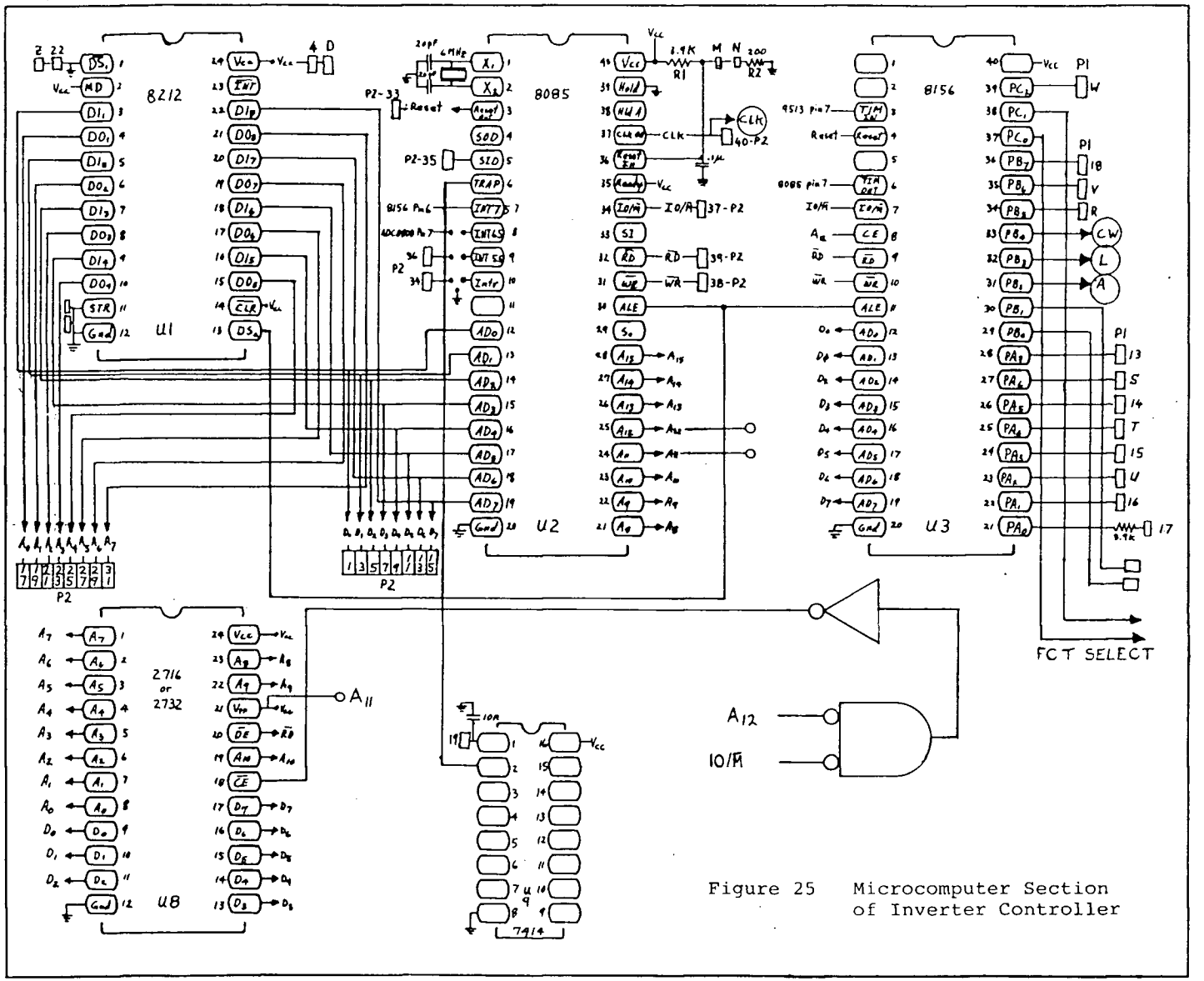


Figure 25 Microcomputer Section of Inverter Controller

would be very easy to upgrade to the 8085A-2 which runs at clock frequencies up to 5 MHz if an application required the extra speed.

The power-on reset circuit in this design is conventional. Provision is made for the addition of a reset switch or some other external reset signal source.

The 8085 has five interrupt inputs. In the inverter controller, the TRAP interrupt input, which is non-maskable and has highest priority, is dedicated to fault signals from the inverter. The INT 7.5 interrupt, which is next highest in priority, is dedicated to the real-time clock signal, as described below. The INT 6.5 input can be connected either to the END OF CONVERSION signal from the A/D convertor or to fault signals from the inverter. The remaining two inputs, INT 5.5 and INTR, can also be connected to fault signals from the inverter.

The Intel 8156 RAM and I/O Expander IC supplies the microcomputer with 256 bytes of read/write memory, 22 bidirectional I/O lines, and a real-time clock. At first glance, 256 bytes of read/write memory appears insufficient. However, the inverter controller has no need to store large amounts of information. During the initial design of the controller it was estimated that the number of variables to be stored at any one time would not exceed 50 and that most of these would be single byte variables. This estimate has been borne out in practice, lack of read/write

memory has not proved to be a problem so far.

The I/O lines on the 8156 are organized into two eight line ports (Port A and Port B) and one six line port (Port C). Port B and Port C are set up as output ports. Five lines from these ports are used for control functions within the controller. The remaining lines are available to control components such as contactors in the inverter or to control indicator lamps on the control panel. Port A is set up as an input port. Its eight lines are available for use as a parallel digital input to accept speed commands from the submersible controller. The eight lines can also be used to accept individual logic inputs from switches or comparators.

The 8156 is equipped with a 14 bit programmable counter/timer. In this design the input to the counter is a clock frequency from an Am9513 counter/timer IC on the controller board. The output of the counter is connected to the INT 7.5 interrupt input on the microprocessor. The counter is programmed to act as a real-time clock which interrupts the microcomputer at regular intervals.

Intel manufactures a companion chip to the 8156, the 8755, which incorporates EPROM memory and more I/O lines. However the chip only contains 2K bytes of EPROM yet is packaged in a 40 pin package. For about the same board area it is possible to include a 2732 EPROM which contains 4K bytes and the 8212 latch required to demultiplex the 8085 address bus. This was the option chosen

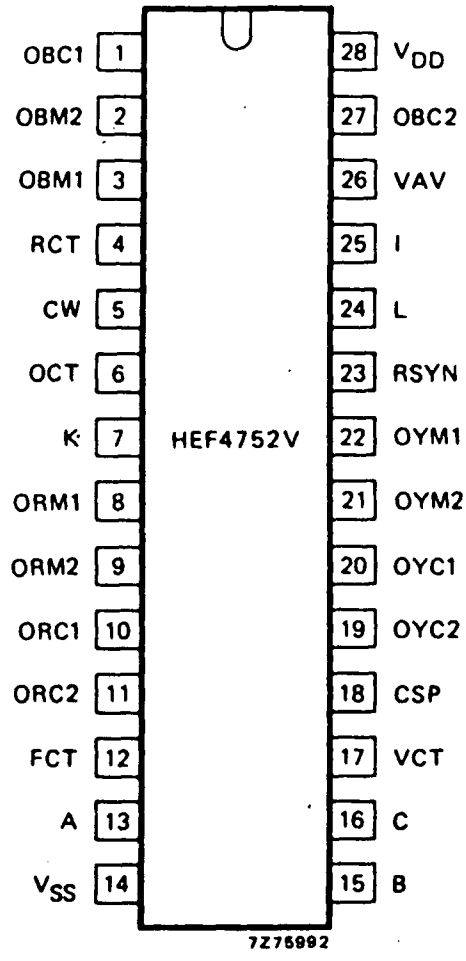
since the extra I/O lines provided by the 8755 were not required.

So far, 4K bytes of EPROM has proved adequate for program storage. However, fitting the program into this space has occasionally been a bit of a squeeze. Programs for more complex applications of the controller are likely to require more storage. At the time, the 2732 was the highest capacity EPROM that was readily available. Since that time, EPROMs with capacities up to 32K byte have appeared on the market. These devices use a 28 pin package that is larger than the 24 pin package used by the 2732. However it should be possible to fit one of these devices on the same board by replacing the 8212 latch, which is housed in a 24 pin package, with a 74LS373 latch which is housed in a smaller 20 pin package.

4.2 Interface to the HEF4752V PWM Generator IC

The microcomputer is interfaced to two other elements of the inverter controller; the PWM generator IC and the analog to digital converter. The interface to the Philips HEF4752V PWM generator IC is the more complex of the two. The HEF4752V (Figure 26 [65]) has 7 single bit binary control inputs and 4 clock frequency control inputs.

The 7 single bit inputs are labelled I, K, L, CW, A, B, and C by the manufacturer. Four of these inputs can be tied permanently high or low. The logic level on input I determines whether



HEF4752VP: 28-lead DIL; plastic (SOT-117).
HEF4752VD: 28-lead DIL; ceramic (SOT-135).

PINNING

Inputs; group I		Inputs; group II	
24 = L	data	12 = FCT	frequency clock
25 = I	data	17 = VCT	voltage clock
7 = K	data	4 = RCT	reference clock
5 = CW	data	6 = OCT	output delay clock
13 = A	data		
15 = B	data		
16 = C	data		

Outputs; group I

23 = RSYN	R-phase synchronization
26 = VAV	average voltage
18 = CSP	current sampling pulses

Outputs; group II

8 = ORM1	R-phase main
9 = ORM2	R-phase main
10 = ORC1	R-phase commutation
11 = ORC2	R-phase commutation
22 = OYM1	Y-phase main
21 = OYM2	Y-phase main
20 = OYC1	Y-phase commutation
19 = OYC2	Y-phase commutation
3 = OBM1	B-phase main
2 = OBM2	B-phase main
1 = OBC1	B-phase commutation
27 = OBC2	B-phase commutation

SUPPLY VOLTAGE

	rating	recommended operating
HEF4752V	−0,5 to 18	4,5 to 12,5 V

FAMILY DATA see Family Specifications

Figure 26
Philips HEF4752V Inputs and Outputs

the output signals produced by the IC are of a type suitable for driving a transistor (or power MOSFET) inverter or of a type suitable for driving a thyristor inverter. In the transistor mode the signal to turn a switching device on is maintained continuously during the interval that the device is on. In the thyristor mode the output signal to turn a device on becomes a pulse train. This facilitates the use of pulse transformers for voltage isolation. In the inverter controller, this input is tied either high or low by a jumper, depending on what type of inverter is to be controlled. Input K is used with one of the clock frequency control inputs to control the delay or interlock period between the switching of two devices in the same leg of the inverter. Again this input can be tied either high or low by a jumper, depending on the interlock period desired. Inputs B and C are used only during the testing of the device by the manufacturer. In the inverter controller they are both tied low.

The remaining three single bit inputs must be controlled by the microprocessor. The L input is an on/off control. When L is low, the output of inverter switching signals from the IC is disabled. The outputs which control inverter switching go low. When L is high, the inverter switching signals are enabled. This input allows all the inverter switches to be turned off in the event of a fault or in cases where the inverter is to be inactive. The CW input controls the phase sequence of the three phase waveform generated by this IC. This is used to control the direction of motor rotation. Finally, the A input is used to

reset the HEF4752V IC to a defined initial state when the controller is first switched on. These three inputs are connected to Port B on the 8156 so that they can be controlled by the microcomputer.

The four clock frequency inputs to the HEF4752V are labelled FCT, VCT, RCT, and OCT by the manufacturer. Two of these clock frequencies are normally kept constant for a particular application. The frequency of the RCT clock sets limits to the maximum and minimum inverter switching frequencies over a wide range of inverter output frequencies. This is done by changing the ratio of carrier (switching) frequency to modulation (output) frequency in discrete steps as the output frequency changes. The ratios used by the chip are 168, 120, 84, 60, 42, 30, 21, and 15. The selection of the RCT clock frequency (f_{RCT}) determines the output frequencies at which the carrier to modulation frequency ratios will be changed. The formula for selecting f_{RCT} is

$$f_{RCT} = 280 \times f_{s(max)} \qquad (25)$$

where $f_{s(max)}$ is the desired maximum switching frequency. The value for f_{RCT} will be different for inverters using different types of switching devices but will remain constant for a particular application.

The OCT clock frequency input is used in conjunction with the

binary input K to set the interlock delay period required between the on time of one switch in an inverter leg and the on time of the other switch in the leg. When the K input is low, the formula for the interlock delay period is

$$(26) \quad \text{Interlock delay (seconds)} = 8/f_{OCT}$$

where f_{OCT} is the frequency of the OCT clock. When the K input is high, the formula for the interlock delay period is

$$(27) \quad \text{Interlock delay (seconds)} = 16/f_{OCT}$$

The interlock delay period required will vary depending on the switching speed of the power devices in the inverter. However, the interlock delay period will not vary for a particular application so f_{OCT} can be kept constant for that application.

The remaining two clock frequency control inputs require variable clock frequencies. The FCT clock input controls the inverter output frequency. The formula relating the clock frequency f_{FCT} to the output frequency f_{out} is

$$(28) \quad f_{FCT} = 3360 \times f_{out}$$

In the submersible drive, f_{out} (and therefore f_{FCT}) varies from about 2 Hz to 400 Hz. This is a range of 200 : 1.

The VCT clock input controls the inverter volts/Hz ratio. The nominal value of the VCT clock frequency f_{VCT} determines the output frequency at which the inverter achieves 100% modulation. This nominal value can be calculated as follows:

$$(29) \qquad f_{VCT}(\text{nom}) = 6720 \times f_{out}(100\% \text{ modulation})$$

With f_{VCT} fixed at this value, the inverter output voltage will be a linear function of the output frequency up to the 100% modulation point. After that, the voltage rises in a non-linear fashion as the inverter waveform makes the transition from a sinusoidal PWM waveform to an unmodulated six step square wave. Depending on the motor being driven, the frequency at which 100% modulation is desired can range from 50 Hz to 400 Hz. Thus VCT must be variable over about a 10 to 1 range. In addition, variation in the volts/Hz setting may be required during operation at low frequencies due to the increased effect of the stator resistance IR drop on the air gap flux.

The method of generating the clock frequencies for the FCT and VCT inputs requires some consideration. The FCT input requires particular attention. The output frequency of the inverter must be controlled over a range of 2 Hz to 400 Hz. In addition, the frequency must be changed in small steps. Each step change in inverter output frequency is also a step change in motor slip frequency. This change in slip frequency causes a change in

motor torque and therefore in motor current. If the step is too large, large transient currents will occur. Step changes in frequency should be kept to a fraction of the normal slip frequency. A 0.5 Hz step size is a good choice for small induction motors. This results in the requirement that the f_{FCT} clock source generate approximately 800 frequencies evenly spaced between 6720 Hz (2 Hz output) and 1,344,000 Hz (400 Hz output).

In the case of the VCT signal it is desirable to be able to change its frequency in small steps (for example, 5%) around the nominal frequency. This ability is required when the controller is performing closed loop control of machine air gap flux or when the controller is compensating for fluctuations in the input voltage to the inverter. The frequency range required for the VCT clock is from 336,000 Hz (100% modulation at 50 Hz output frequency) to 2,688,000 Hz (100% modulation at 400 Hz output frequency).

Generation of the FCT and VCT clock frequencies under control of the microcomputer can be achieved by three techniques:

i) programmable counter dividing a fixed clock frequency

ii) binary rate multiplier dividing a fixed clock frequency

iii) phase-locked loop frequency multiplier operated as a

Using a programmable counter is not a viable alternative, however, because the clock frequency must be several hundred megahertz in order to achieve the required frequency resolution for the FCT signal. Even the coarser resolution of the VCT signal would still require a clock frequency of 40 to 50 MHz. While a high frequency programmable divider circuit could be built using ECL components, it would have a high parts count and would not be easy to interface to the microcomputer.

A binary rate multiplier generates output pulses at a frequency that is related to the input clock frequency by a rational fraction. For example, a 12 bit rate multiplier has the following relationship between input and output frequencies:

$$(30) \qquad f_{out} = f_{in} \times N/4096$$

where N is an integer between 0 and 4095. In this case a high clock frequency is not necessary in order to obtain the required resolution. However, the output pulses generated by a rate multiplier are usually not evenly spaced. This is because the rate multiplier generates its output pulses by letting some of the input pulses pass to the output and suppressing the remainder. Thus the formula for f_{out} given above is only true as an average. This could pose a problem if a rate multiplier is used to generate the control frequencies for the

HEF4752.

The HEF4752V uses the control frequencies to control the timing of the switching pulses it generates. It is possible that the irregular spacing of the pulses produced by a rate multiplier will propagate through to the switching pulses and result in a PWM waveform with noticeable jitter. Unfortunately, the specification sheet supplied for the HEF4752V provides no information on this point. In addition, the application example supplied by the manufacturer consists of an analog inverter controller which generates the control frequencies for the HEF4752V by means of voltage to frequency converters. Thus no information is provided on the requirements for control frequencies generated by digital means.

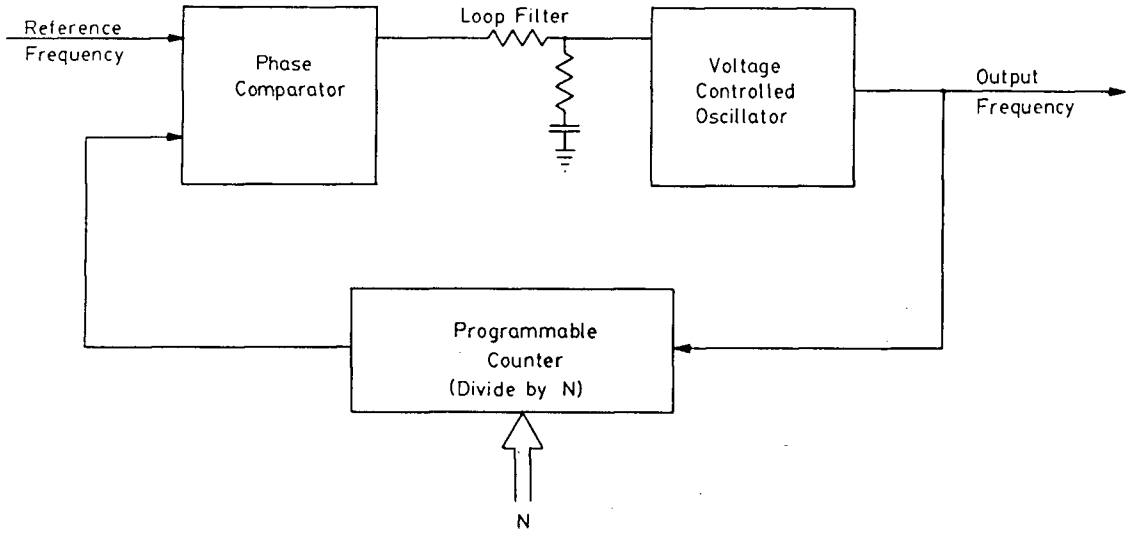


Figure 27 Digitally Controlled Frequency Synthesizer

The digitally controlled frequency synthesizer (Figure 27) consists of a phase-locked loop with a programmable counter in its feedback path. The output frequency f_{out} of the frequency synthesizer is

$$(31) \qquad f_{out} = N \times f_{ref}$$

where N is the divisor programmed into the counter and f_{ref} is the reference frequency applied to the phase comparator. In this case the resolution is dependent on the length of the progammable counter. The range of frequencies that can be generated is basically limited by the range of the voltage controlled oscillator (VCO) in the phase-locked loop. Modern digital phase comparators allow the loop to remain in phase lock over the full VCO frequency range [66].

Quick preliminary designs were carried out for frequency generators based on rate multipliers and on frequency synthesizers, to estimate the number of ICs each approach would require. It was estimated that the rate multiplier approach would require between 7 and 10 16 pin ICs. The estimate for the frequency synthesizer approach was 5 or 6 16 pin ICs plus a 40 pin IC. However, the 40 pin IC used by the frequency synthesizer had some additional capabilities which would be useful in other parts of the controller design. Thus there was not much to choose between the two alternatives in terms of board space

required.

As already discussed, the rate multiplier approach has the disadvantage that it produces an output in which the pulses are unequally spaced, which may cause jitter in the switching signals. The pulses produced by the frequency synthesizer, on the other hand, are equally spaced. The frequency synthesizer approach therefore appeared to be less risky, particularly since it had been used successfully in the prototype controller described in the previous chapter. As a result, the decision was made to use the frequency synthesizer technique to generate the FCT and VCT clock frequencies.

Figure 28 shows the schematic diagram for the frequency synthesizer circuit used to generate the FCT and VCT clock frequencies. The programmable counters which are used to set the output frequencies of the synthesizer are contained in the Advanced Micro Devices Am9513 Counter/Timer IC (see Figure 29 [67]).

This device contains five 16 bit programmable counters which can be loaded by the microcomputer. In addition there is another 4 bit frequency divider (FOUT) which can be set up by commands from the microcomputer. In the inverter controller design, the 9513 is supplied the 3 MHz microcomputer clock signal. The FOUT divider is set up to divide this clock frequency by five to supply a 600 kHz clock signal to the real-time clock in the 8156

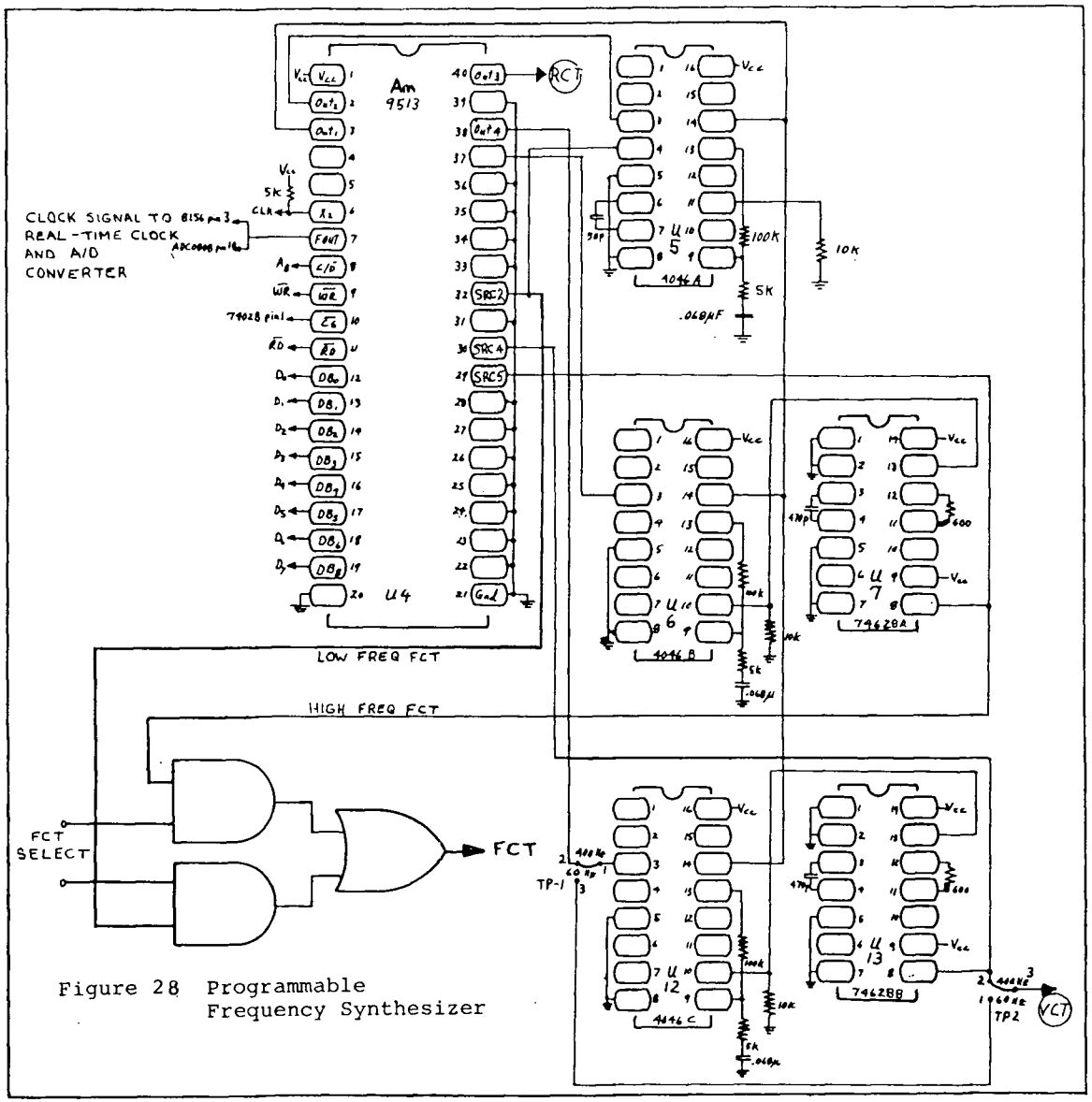


Figure 28 Programmable
Frequency Synthesizer

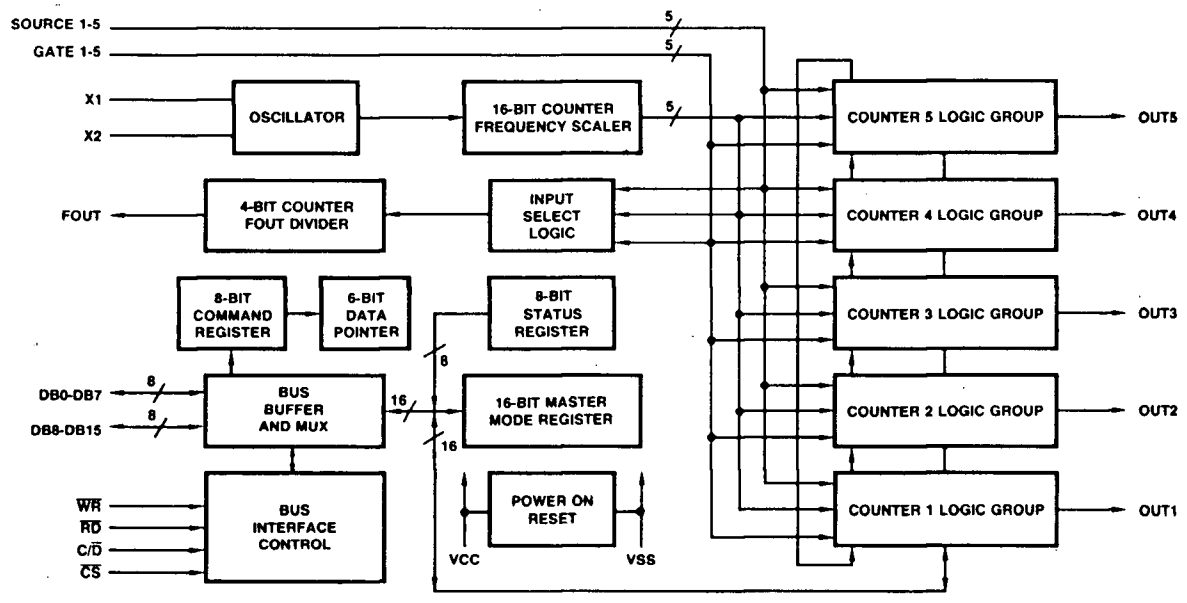


Figure 29 General Block Diagram for Am9513 Counter/Timer IC

and to the A/D convertor.

Counter 1 in the 9513 is programmed to divide the 3 MHz clock signal to produce a 1680 Hz reference frequency for the phase locked loops. Counter 3 is programmed to divide the 3 MHz clock signal to produce the RCT control frequency for the HEF4752V PWM waveform generator IC. The remaining three counters are in the feedback loops of the frequency synthesizers.

Two separate frequency synthesizers are required to generate the FCT clock frequency. This is due to component limitations. The CD4046 phase-locked loop IC contains both a phase comparator and a VCO. The VCO has a frequency range of about 1000 to 1 but is

limited to a maximum output frequency of between 500 kHz and 1 MHz when operating from a 5 volt supply [68]. The maximum frequency required for the FCT signal is 1.344 MHz. It is possible to use the phase comparator in the CD4046 with another VCO such as the 74LS628 TTL VCO IC. The 74LS628 can supply frequencies well in excess of the required 1.344 MHz but its frequency range is limited to about 10 to 1 [69]. The frequency range required for the FCT clock frequency is at least 200 to 1. Thus we have one VCO which has the required frequency range but can't supply the required maximum frequency while another VCO will supply the required maximum frequency but doesn't have the frequency range required.

The solution is to use two frequency synthesizers. The synthesizer associated with Counter 2 in the 9513 uses the VCO in the CD4046 and generates FCT frequencies from 6720 Hz (2 Hz output frequency) to 403,200 Hz (120 Hz output frequency). The other frequency synthesizer, associated with Counter 5 in the 9513, uses the phase comparator in a CD4046 in conjunction with a 74LS628 VCO to generate the FCT frequencies from 403,200 Hz to 1.344 MHz (400 Hz output frequency). Some simple logic circuits, controlled by signals from the microcomputer (FCT SELECT), select signals from one of the FCT sources to send to the PWM generator IC. The operating ranges of the two frequency synthesizers overlap around the frequency where switching occurs (403,200 Hz) and they are phase locked to a common reference frequency so switching from one synthesizer to the other is completely

"glitch" free.

Only a single frequency synthesizer is required for the generation of the VCT clock frequency. However, since a maximum frequency of 2.688 MHz is required, the synthesizer consists of the phase comparator section of a CD4046 used with a 74LS628 VCO. Counter 4 in the 9513 is used as the counter in the feedback path of the VCT synthesizer.

It will soon be possible to considerably simplify the design of the FCT and VCT frequency synthesizers. A high speed silicon gate CMOS version of the metal gate CMOS CD4046 phase-locked loop IC should be available by mid 1984. The new high speed silicon gate CMOS ICs normally run ten times as fast as their older metal gate counterparts. Therefore it should be possible to do away with one of the FCT frequency synthesizers and with the 74LS628 VCO in the VCT synthesizer. Besides reducing the chip count, this change will also make one of the counters in the Am 9513 and two output lines on the 8156 available for other uses.

The design of the loop filters for the phase-locked loops follows the guidelines given by the manufacturer of the CD4046 phase-locked loop IC [70]. The rise time of the VCO control voltage for a step change in the frequency command is about 20 milliseconds. There is no noticeable ringing or overshoot. This speed of response has been more than adequate for all of the applications of the inverter controller to date. The maximum

rate of change of inverter frequency in these applications has been limited to 200 Hz/sec due to limitations on the current that the inverter can supply. It is possible that an application in which the inverter controller is performing fast closed loop speed control of a low inertia motor might encounter some problems due to the lag in the phase-locked loop response. In that case it may be necessary to redesign the synthesizers so that they run at higher frequencies so that loop filters with shorter time constants can be employed. The outputs of the synthesizers would then be divided down to get the frequencies required by the HEF4752V.

With the phase-locked loop reference frequency set at 1680 Hz, the programming of the frequency synthesizers is quite easy. The counters controlling the FCT synthesizers are loaded with an integer equal to twice the desired inverter output frequency. The rationale for this can be seen from the following:

i) Required FCT frequency = $3360 \times f_{inverter}$

ii) Synthesizer output frequency = $N \times f_{reference}$

iii) If $f_{reference} = 1680$ and $N = 2 \times f_{inverter}$
then synthesizer output frequency = $3360 \times f_{inverter}$

This provides a 0.5 Hz resolution in the inverter output frequency.

The counter controlling the VCT synthesizer is loaded

with an integer equal to 4 times the inverter frequency at which 100 % modulation of the PWM waveform is to occur. The rationale for this is similar to that given for the FCT frequency synthesizer.

4.3 Remainder of Controller Circuit

The remainder of the inverter controller circuit is shown in Figure 30. The analog to digital converter used in this design is a National ADC0808. This device has eight analog input channels that accept unipolar voltages between 0 and 5 volts. The convertor has 8 bit resolution and completes a conversion in about 100 microseconds. The primary reason for choosing this converter was that it incorporated an eight input multiplexer and an A/D converter into a single package that interfaced readily to the 8085 microcomputer. The convertor resolution and input voltage range have proved adequate in all applications to date.

The OCT signal to the HEF4752V PWM generator IC is supplied by the 3MHz microcomputer clock. Since the K input to the HEF4752V can be jumpered high or low, the user can select interlock delay periods of either 2.7 microseconds or 5.3 microseconds. These delays are adequate for inverters based on power MOSFETs or fast power transistors. Provision is made to supply an off-board OCT signal for those cases where a longer interlock delay period is required.

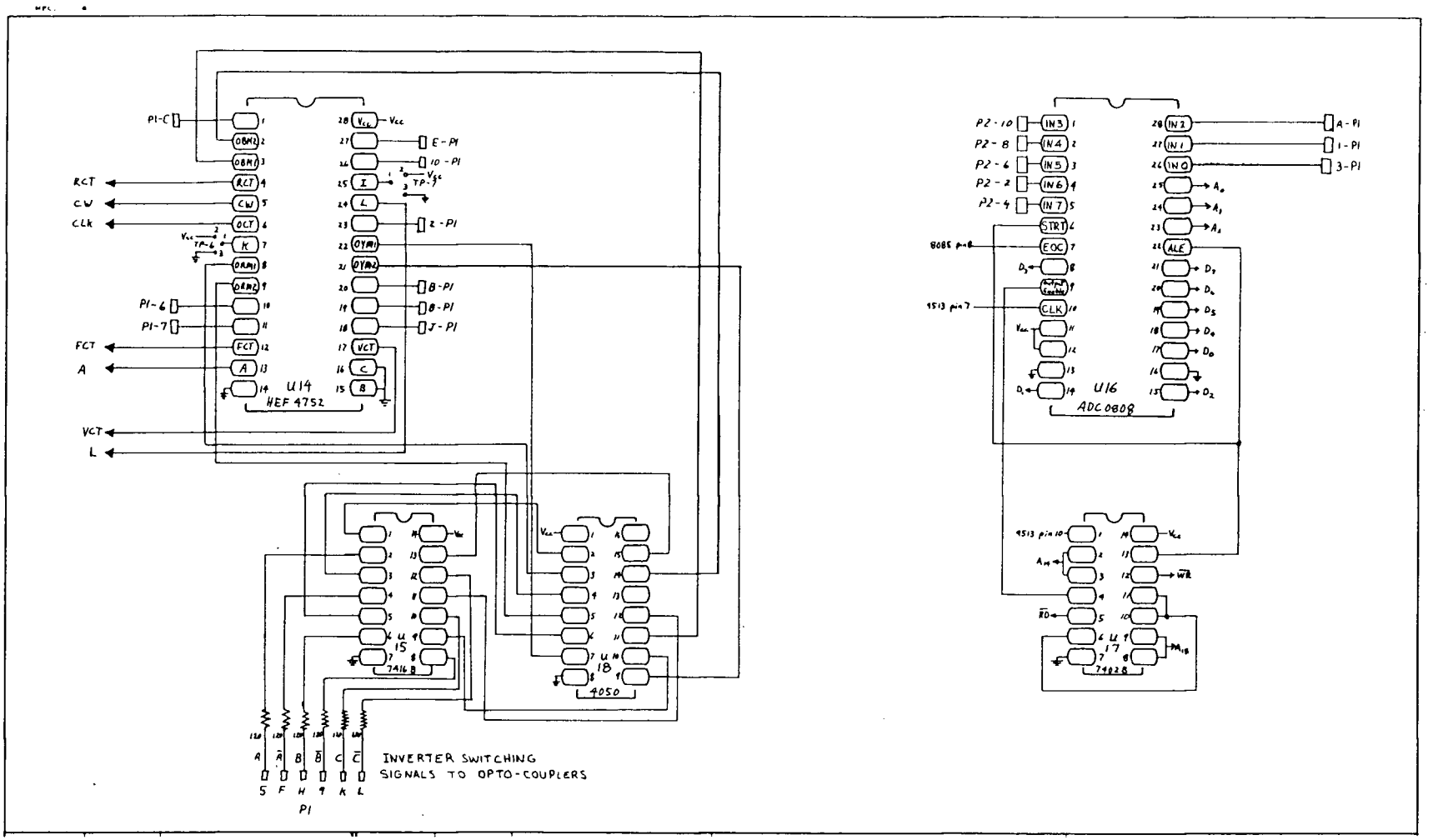


Figure 30 Remainder of Inverter Controller Circuit

The inverter switching signals supplied by the HEF4752 are first buffered by a CD4050 IC and then sent to a 7416 open-collector inverting driver which supplies the current required to drive the LEDs in the opto-couplers on the driver boards for the inverter switches. The 7416 could also drive pulse transformers if required. The commutation signals generated by the HEF4752V for use in thyristor inverters are brought to connectors on the inverter controller board but are not buffered in any way.

Figure 31 shows the memory and I/O maps for the inverter controller. The address decoding was kept very simple in order to minimize the controller chip count. However, some I/O addresses are available for external devices which can be interfaced to the microcomputer in the inverter controller. One application of the inverter controller has made use of this feature to interface a keypad and alphanumeric display to the controller.

4.4 Inverter Controller Circuit Board

After the design of the inverter controller was completed, a wire-wrapped prototype was constructed. This prototype was debugged and then used to test the first version of the inverter controller software. Once it was clear that the design was free of errors and met the basic requirements outlined in Chapter 3, the schematics for the controller, along with specifications on the printed circuit board format used by International Submarine Engineering, were sent to a company which specializes in printed

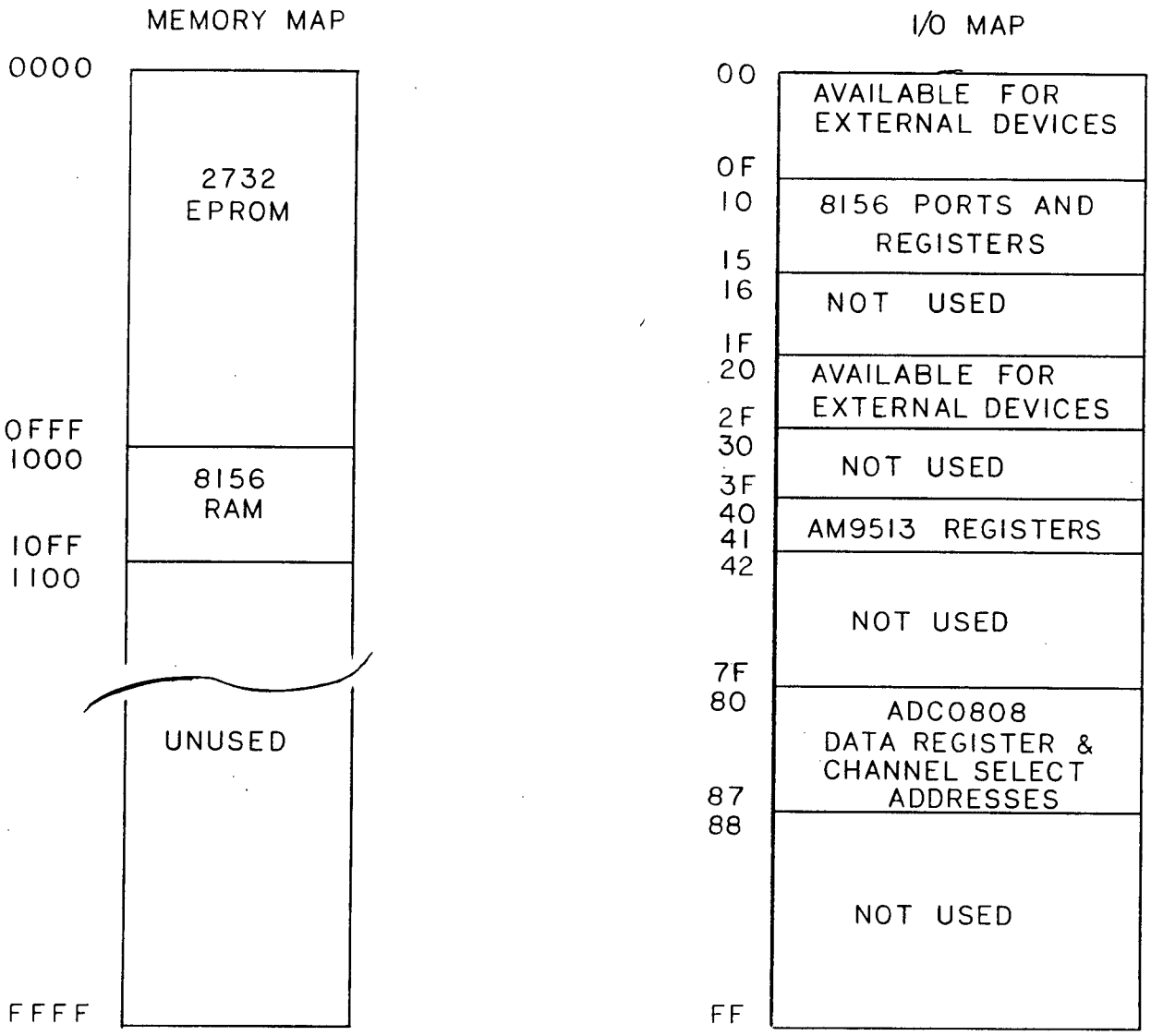


Figure 31
Memory and I/O Map For Inverter Controller

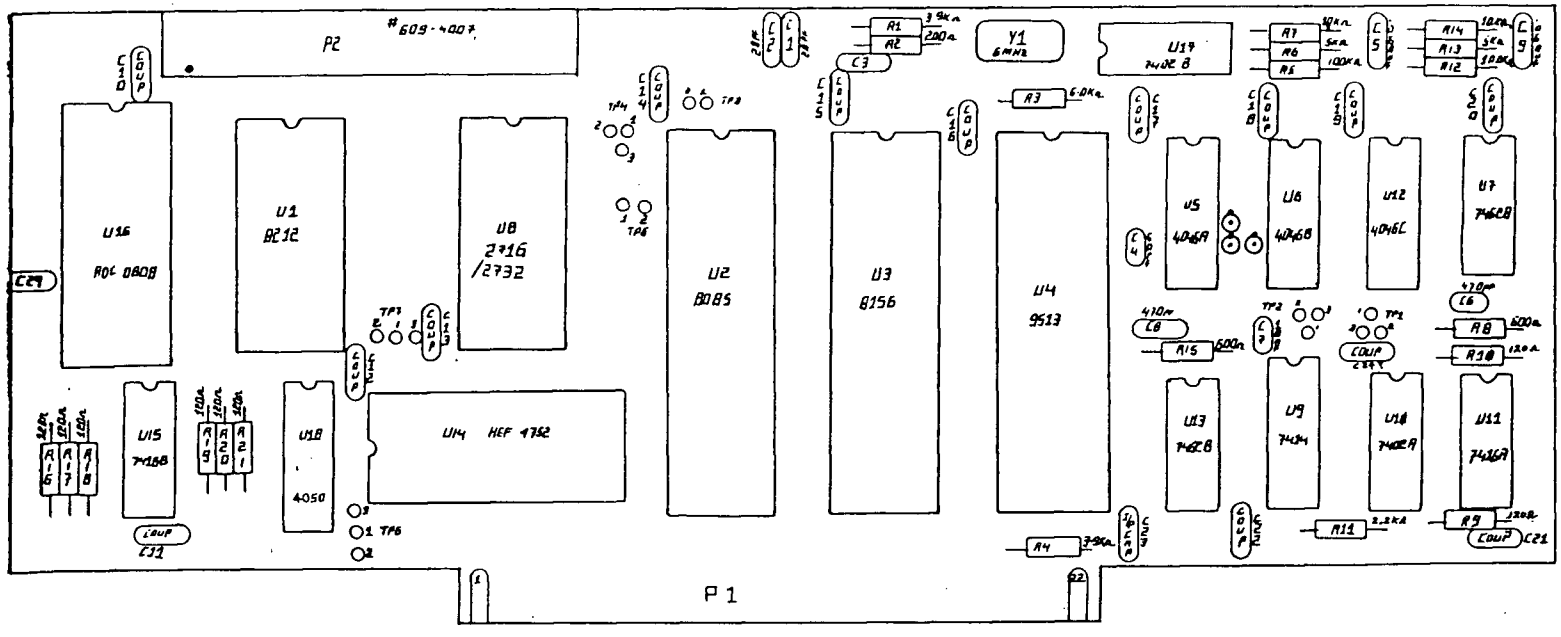


Figure 32
Inverter Controller Circuit Board
Layout

circuit board layout. The company was successful in fitting the entire controller on one printed circuit board. The basic layout of the printed circuit board is shown in Figure 32. The most frequently used lines from the controller are brought to the edge connector P1. Less used lines are brought to the ribbon cable header P2.

While the layout shown is ideal for the submersible application, and is acceptable for other stand-alone applications, the card shape and edge connector format are non-standard and therefore not very good for applications where the inverter controller is part of a larger system. However the area of the inverter controller board is almost identical to that of the printed circuit boards used in the STD microcomputer bus system. Therefore it should be possible to redesign the inverter controller to fit on an STD bus card. The controller could then be used as a peripheral device by a STD bus microcomputer or, alternatively, the inverter controller could make use of the wide range of commercially available STD bus cards to expand its own capabilities.

Chapter 5

Inverter Controller Software Design

The inverter controller software consists of components which are used in virtually all applications and components which are specific to particular applications. This chapter will concentrate on the design of of those components, such as controller and inverter initialization, controller sequencing, and control of motor acceleration, deceleration, and reversing, which are common to all applications. The following chapter discusses applications of the inverter controller and the design of some of the software components specific to these applications.

One of the distinguishing features of the software is that it is written in C, which is a high-level programming language. Other microprocessor based inverter controllers described in the literature have been programmed in assembly language. Besides making software development easier and faster, the use of a high level language results in a controller program that is easier to document, maintain, and modify than the equivalent assembly language program. The controller software is designed, as far as possible, in a modular fashion in order to further enhance the ability to maintain and modify the software.

Another unique aspect of the software design of the inverter

controller is that it is organized as a finite state machine. While the state machine approach to software design has been used in a variety of applications, it does not appear to have been used previously in controllers for power electronic equipment. Designing the inverter controller software as a finite state machine is considerably less prone to error than the more ad hoc techniques commonly used to design the software for power electronic equipment.

5.1 Software Development System

One of the goals of this project is to design the inverter controller so that it can be easily and quickly adapted to a new drive application. Performing many of the controller functions with computer software will help to achieve that goal if the process of developing the software is made quick and easy. In addition the software should be designed so that it is easy to maintain and modify. These requirements provide a strong incentive for the use of a high level programming language. However, the machine language code produced by the translation of the high level language source code must be efficient in terms of execution speed and storage requirements. Reasonable execution speed is required in order that the controller can carry out all its required functions without serious degradation of its speed of response. Fortunately, the use of the Philips HEF4752V PWM waveform generator IC considerably reduces the required execution speed since many of the speed critical controller functions are

peformed by this IC. Compact code is required so that the controller program can fit into the limited EPROM space on the controller card.

A number of high level languages were considered for this project. In fact, some initial software development work was done using the PL/M cross-compiler available on the UBC computer system. However, the C language [71] was finally chosen as the programming language for this project. Benchmark tests of high level language compilers have shown that C compilers are typically among the best in producing efficient machine language code for the 8080 family of microcomputers [72, 73, 74]. The C language has features such as bit manipulation and shifting instructions which make it a good assembly language replacement, yet it also has high level language facilities, such as control structures and data typing, which support a structured approach to program design. A final advantage of the C language is that inexpensive compilers are available for most microprocessors. Thus a change in the microprocessor used in the inverter controller will not result in the obsolescence of all the software developed for the controller.

As discussed in the previous chapter, it was important to have a low cost software development system for this project. The development system used certainly meets this requirement. It is based on an Apple II personal computer equipped with a Z-80 co-processor card that allows it to run programs written for the

CP/M operating system. The development software consists of a C compiler which generates Intel 8080/8085 assembly language, a relocating assembler and linker for the 8080/8085, and a text editor for editing the source files. An EPROM programmer card that plugs into the Apple II is used to store the object code into a 2732 EPROM. The entire system, including the computer, a printer, the EPROM programmer, and the development software, can be purchased for under $5000.

5.2 Description of Controller Software

In any software development project it is helpful to use some form of system model as the basis for the design. The nature of the model will depend on the nature of the project. In the design of a numerical analysis program the model is likely to be the mathematical algorithm that the program is supposed to carry out. In a data processing application, the model may consist of data flow graphs and descriptions of data structures.

A good model for the inverter controller software is the finite state machine [75]. A finite state machine is a system which is in one of a finite number of states at any one time. The machine makes a transition from one state to another as a result of some event. The state that the system moves to on a state transition is determined only by the system's present state and the occurrence of a particular event or combination of events during the present state. The finite state machine model has been used

in a number of control oriented software designs, including an industrial sewing machine controller [76], process control [77], and a controller for a hybrid electric vehicle [78].

The operation of a finite state machine (FSM) can be described graphically by the use of a state transition diagram (Figure 33). The diagram consists of circles, each of which represents a state, and lines indicating transitions among states. Each state represents an operating mode in which the tasks being performed do not change. The lines indicating transitions are marked with the condition that causes the change of state. The condition could be the value of an input or program variable, an interrupt, or simply the completion of all the tasks within the present state.

The state transition diagram explicitly shows the conditions that cause a change of control while suppressing much of the detail about the functions performed within each state. This makes it a useful tool when designing controllers where much of the design involves determining the correct control sequence. It is also possible to identify errors in control flow from this diagram. The diagram at the top of Figure 34 shows an inconsistent FSM where more than one transition is possible from state 1 when condition A is true. The middle diagram shows an incomplete FSM where the transitions for A and B not true are not specified. Finally, the bottom diagram shows an FSM with an unreachable state. State 4 will never be reached since D is always false in

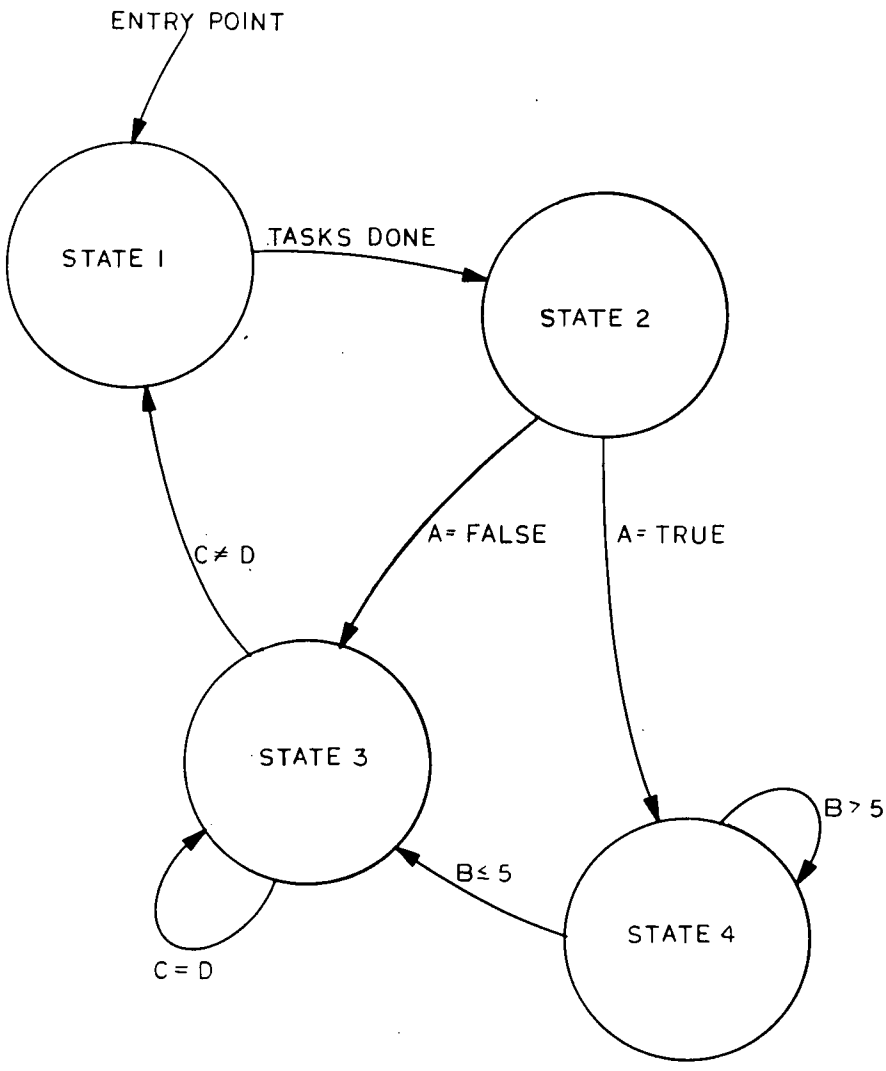
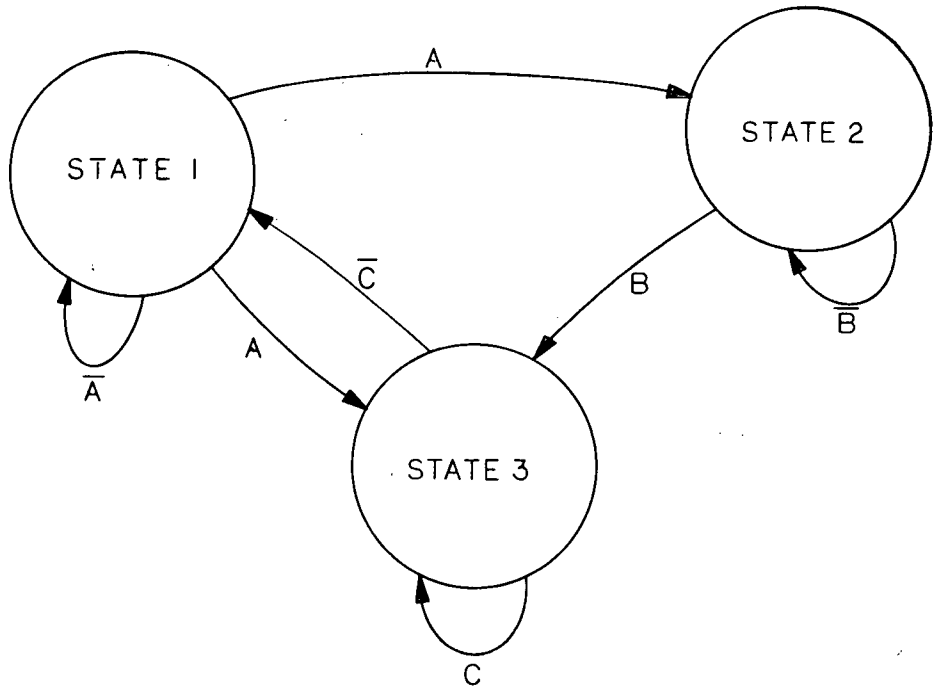
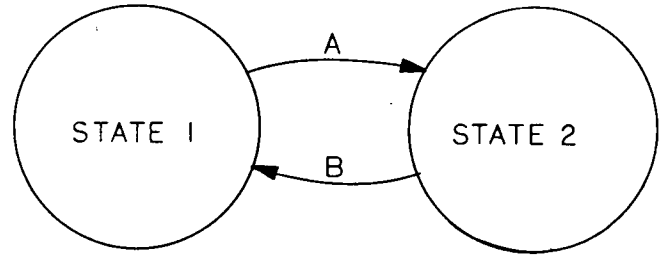


Figure 33 State Transition Diagram





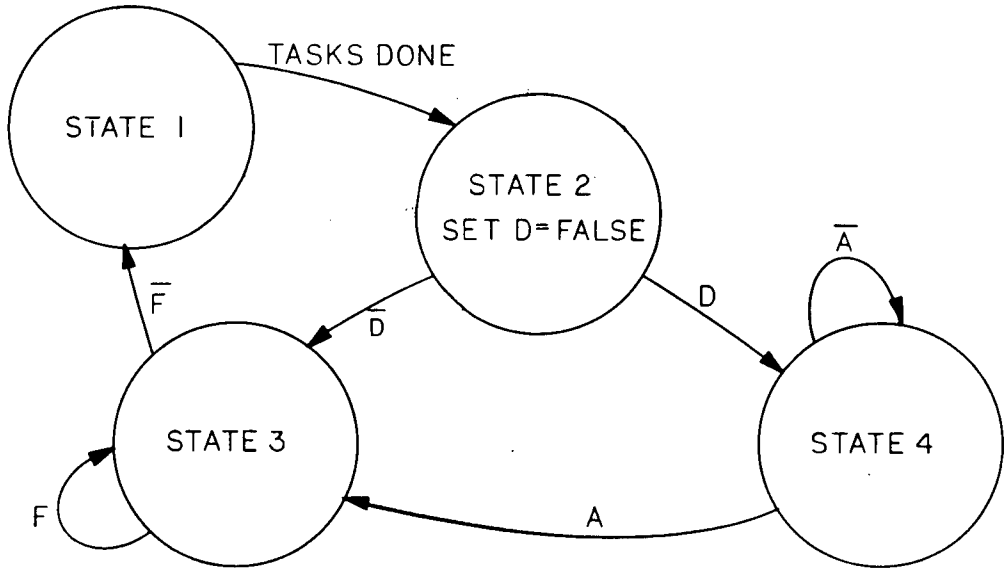


Figure 34 Errors in Control Flow

state 2. Real systems often have many states and many possible transitions among them. It is very easy to to design a system with all of the above errors if a state transition diagram is not used as an aid in detecting and eliminating them.

The state transition diagram for the basic FSM model of the inverter controller is shown in Figure 35. When power is applied to the inverter controller, it enters the INITIALIZATION state. Within this state, the controller software configures the 8156 RAM-I/O chip and the 9513 counter/timer chip to the operating modes required for proper controller operation, loads and starts all the counters, and carries out the reset sequence required for the HEF4752V PWM waveform generator IC. The software also initializes any variables requiring explicit initialization and enables those interrupt inputs that are recognized by the controller. Once this is done the controller can begin initializing the inverter.

The actions required to initialize the inverter depend on the design of the inverter. However they may include closing a contactor to connect the inverter to the AC supply, initializing the fault detection circuits, performing some initial diagnostic checks, and setting the displays on the control panel to their initial condition.

Once the initialization process is complete, the software makes an unconditional transition to the OFF state. In this state the

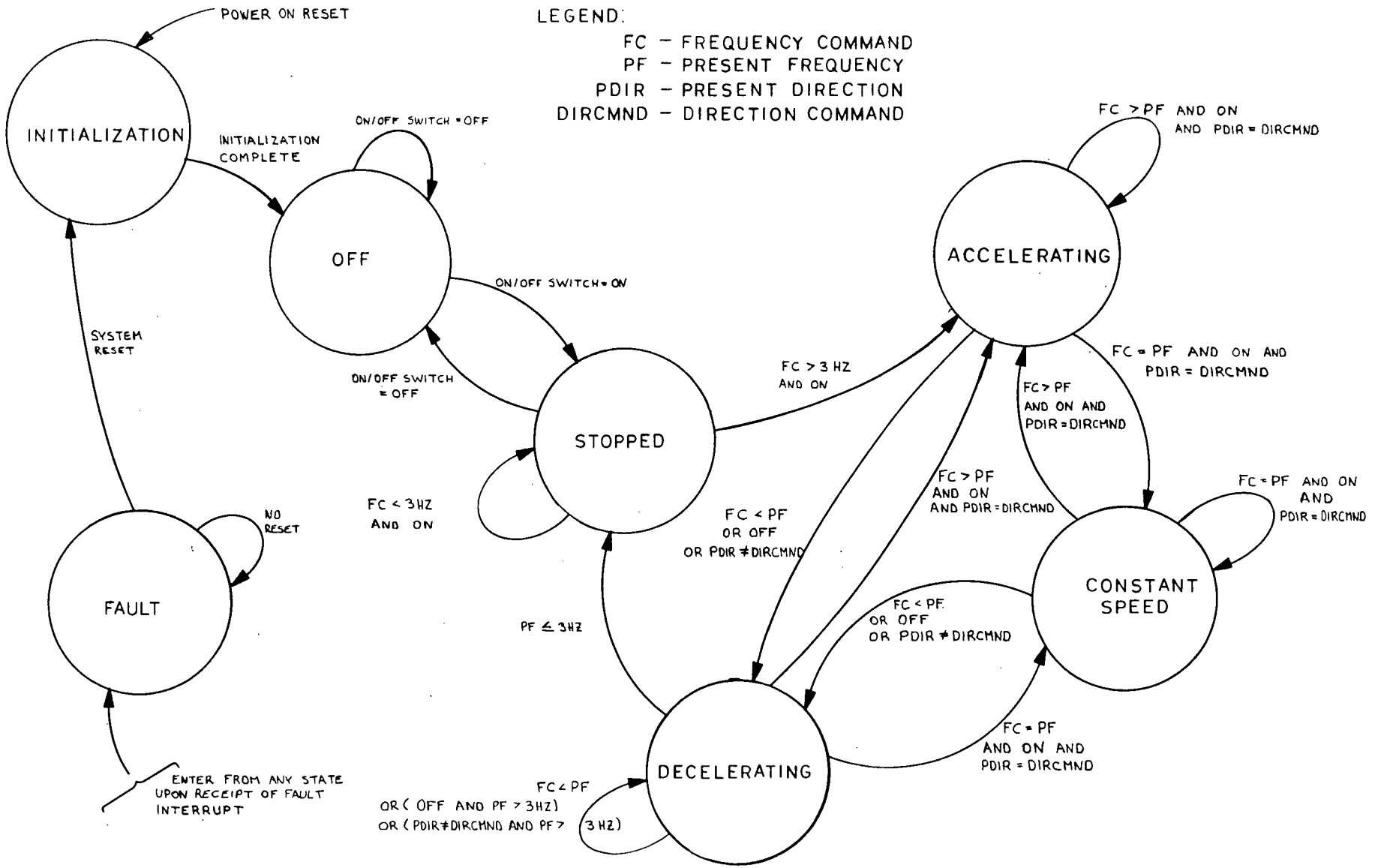


Figure 35 Finite State Machine Model of Inverter Controller

program simply waits for a signal to turn the drive on. This signal, marked as the ON/OFF SWITCH on the diagram, may actually come from a physical switch, or it may come as a command from a higher level controller. When the signal to turn the drive on is received, the software makes a transition to the STOPPED state.

In the STOPPED state the program determines the required direction of motor rotation and sets the CW control input to the HEF4752V accordingly. The program also monitors the frequency command input to determine whether the frequency command is greater than the lower limit of 3 Hz. If it is, the program makes a transition to the ACCELERATING state.

At this point the HEF4752V outputs are enabled by setting the L control input high and the drive begins operation. During the time that the program is in the ACCELERATING state, the inverter frequency is incremented at a constant rate so that the motor accelerates. The inverter frequency is continually compared to the frequency command. If it becomes equal to the frequency command, a transition is made to the CONSTANT SPEED state. Alternatively, if the frequency command is changed so that the inverter frequency exceeds it, or if the drive is commanded to change direction or to turn off, a transition is made to the DECELERATING state.

Assuming that a transition to the CONSTANT SPEED state has been made, the program ceases incrementing the frequency and simply

monitors the frequency command, direction command, and ON/OFF SWITCH commands. If the frequency command is changed so that it is greater than the inverter frequency, the software makes a transition back to the ACCELERATING state. If the frequency command is changed so that it is less than the inverter frequency, or if the drive is commanded to turn off or reverse direction, the software makes a transition to the DECELERATING state.

If the transition to the DECELERATING state is made, the program begins to decrement the inverter frequency. If the drive is being commanded to reverse direction or to turn off, decrementing will continue until the inverter frequency reaches 3 Hz. At that point a transition is made back to the STOPPED state and the inverter switching signals are disabled. Alternatively, if the controller is in the DECELERATING state simply because the frequency command is less than the inverter frequency, decrementing is continued until the inverter frequency is less than or equal to the frequency command. If the inverter frequency is equal to the frequency command, a transition is made to the CONSTANT SPEED state. If the frequency command is changed so that it becomes greater than the inverter frequency, a transition is made to the ACCELERATING state.

Many inverter fault conditions can be dealt with by software within one of the states already discussed. For example, if the acceleration or deceleration rate causes the inverter current

limit to be exceeded, the software in the ACCELERATING or DECELERATING state can reduce the rate at which the inverter frequency is changed. Similarly, if the DC bus voltage exceeds its limit during deceleration because of energy regenerated from the motor, software in the DECELERATING state can take corrective action such as closing a contactor to connect a ballast resistor across the DC bus.

Serious drive faults, such as a motor short circuit, require actions which alter the normal sequencing of the finite state machine. Therefore a FAULT state is added which is entered from any of the other states when a fault interrupt is received on the TRAP input of the 8085. The finite state machine can also be designed to enter the FAULT state on receipt of a "panic stop" signal from the control panel. In the FAULT state, the inverter switching signals are disabled in order to shut the inverter off, the contactor (if any) connecting the inverter to the AC supply is opened, and any fault indicators on the control panel are activated. The controller stays in the FAULT state until it is reset, at which time it enters the INITIALIZATION state.

This finite state machine model is quite simple. However its capabilities are sufficient for applications such as the submersible thruster drive. The finite state machine properly handles sequencing of the thruster drive for any combination of commands. For example, if the direction command is changed while the drive is operating, the drive will decelerate to a stop, the

phase sequence of the switching waveform will be reversed, and the drive will accelerate until the inverter frequency matches the commanded frequency. Similarly, if the drive is commanded to turn off while operating, it will decelerate to a stop, disable the inverter switching signals, and then wait for a command to turn itself on. In a more complex drive, this model serves as a useful framework to which the additional functions required by the drive can be added.

5.3 Software Implementation

A listing of the control program for a submersible thruster drive is given in Appendix 1. Much of the program is devoted to controlling the hardware on the inverter controller board. These device specific portions of the program are not overly interesting and so will not be discussed. However, it is of interest to see how the finite state machine model of the controller is actually implemented in the program.

The function **main()**, which implements the state sequencer of the finite state machine is listed below.

```
/* Main function first initializes the system and then loops continuously */

main()
{
initialize();  /*implements the INITIALIZATION state */
while (ALWAYS)
        {

        clocktick = 0;  /* clocktick set to 1 by int75() interrupt service
                           routine which handles real-time clock*/
        if (faultflag == TRUE)  /*faultflag set to TRUE by trap() interrupt
```

```
                                service routine which handles fault
                                interrupts */
            nexstate = FAULTSTATE;
        switch (nexstate)  /* State sequencer */
        {
        case OFFSTATE :
            off();
            break;
        case STOPSTATE :
            stopped();
            break;
        case ACCSTATE :
            accelerating();
            break;
        case DECSTATE :
            decelerating();
            break;
        case CONSTSTATE :
            constantspeed();
            break;
        case FAULTSTATE :
            fault();
            break;

        default :
            fault();
            break;
        }

        while (!clocktick)
            ;   /* Loop while waiting for next real-time clock
                   interrupt, then go through state sequencer
                   again*/
        }
}
```

The function **main()** is entered after power is applied to the controller or the controller is reset. It calls the **initialize()** function which carries out the actions required for the INITIALIZATION state. One of these actions is to set up the real-time clock to interrupt the microprocessor every 5 milliseconds. This real-time clock interrupt is used to drive the state sequencer. After every clock interrupt, the sequencer uses the **switch** control statement to select a function to execute depending on the value of the variable **nexstate** which

represents the next state to be entered from the present state of the finite state machine. The function selected carries out the actions required for the new state and determines the new value for the variable **nexstate** so that the proper state transition will be made on the next pass through the state sequencer.

One exception to the practice of determining the next state in the function performing the actions of the present state is the transition to the FAULT state. The interrupt handler routine for the TRAP interrupt input, which receives fault interrupts, sets a flag variable (**faultflag**) that is checked by the state sequencer. If **faultflag** is set, the state sequencer sets **nexstate** to the value required to enter the FAULT state. Determining the transition to the FAULT state within the state sequencer eliminates redundant statements in the functions for each state since the presence of a fault always causes a transition to the FAULT state no matter what the present state of the system.

The **accelerating()** function, listed below, which carries out the actions required in the ACCELERATING state, is representative of the functions for the other states.

```
                /*  accelerating function  */

/* Carries out functions required while inverter is increasing
   its frequency */

accelerating()
{
if (faultflag == FALSE)    /*Activate the inverter switching
        start();             signals if no fault has occurred */
```

```
/* check if transition to another state is required                */

if(off_or_rev())
        nexstate = DECSTATE;
else if(pos_freq_error())
        nexstate = DECSTATE;
else if(zero_freq_error())
        nexstate = CONSTSTATE;

else
        {
/*      nexstate = ACCSTATE; */
        inc_frequency();
        }
}
```

The function first enables the inverter switching signals if no inverter fault has occurred. This action may be redundant since the switching signals may already be enabled but it does no harm. Then a series of checks are performed to determine if a transition to a state other than the present state is required. This is done by a series of calls to lower level functions which each perform a test and return a value of TRUE or FALSE. For instance, the function **off_or_rev()** checks whether the controller is being commanded to turn the drive off or reverse its direction. If a change of state is required, the code for the new state is stored in the variable **nexstate** and the function returns to the state sequencer to wait for the next clock interrupt. If no change of state is required, a function is called to increment the inverter frequency by 0.5 Hz and then the **accelerating()** function returns to the state sequencer.

It should be noted that the controller software is written in a top-down, modular fashion. At the top of the hierarchy is the state sequencer function **main()** which calls the function

associated with the present state of the finite state machine. This function in turn calls lower level functions to perform its tasks. The details of accessing I/O ports, masking out bits in status registers and other hardware and application specific operations are hidden in the low level functions. This makes the program easier to understand since the reader is not inundated with low level detail when trying to follow the overall operation of the program. It also makes it easier to adapt to changes in the hardware or the application since only the low level functions have to be rewritten.

Some assembly language support functions are required in addition to the program listed in Appendix 1, which is written almost entirely in C. One of these functions is called the run-time header. It contains the code, executed immediately after a system reset, which initializes the stack pointer and then calls the function **main()** in the control program. The run-time header also contains the code required upon entry to, and exit from, an interrupt service routine. The other functions give the C program access to the 8085 I/O instructions (IN and OUT) and to the instructions which control the interrupt system (EI, DI, RIM, and SIM).

5.4 Evaluation of Software Development Using the C Language

Microprocessor based inverter controllers that have been described in the literature appear to have been programmed in

asssembly language. Indeed, the use of assembly language to program controllers for power conversion equipment is almost universal. The reason normally given for the use of assembly language is that the maximum possible execution speed is required for the program. Sometimes limitations in memory size or the lack of a high level language compiler are cited as reasons for the use of assembly language. These last two reasons are no longer very valid since memory costs are now very low and high level language compilers are available for many microprocessors. However, the question of execution speed remains.

It is true that a program written in C will execute more slowly than the equivalent assembly language program if the assembly language programmer is reasonably competent. However if the C program is fast enough to meet all the requirements of the application, it is immaterial that the assembly language program would be faster. In that case the advantages of programming in a high level language predominate.

The C program for the submersible thruster drive, described in the previous section, operates with a 5 millisecond real-time clock interval and has some time to spare. A considerably more complicated program, written for a linear induction motor drive application, operates with a 10 millisecond real-time clock interval and also has time to spare. In both cases the response time is more than adequate. Interrupt service routines to handle faults requiring very fast response are

partly coded in assembly language but these constitute a small portion of the entire program.

The primary reason why a high level language like C can be used successfully in this inverter controller is that the PWM waveform generation IC unloads a large number of speed critical operations from the microprocessor. Another contributing factor is the use of the finite state machine as a software design tool. The finite state machine model encourages an organized and structured approach to the software design which in turn results in a simple, clean, and efficient program.

It is likely that some future applications, involving closed loop control of motor torque or speed, will need to make more use of assembly language in order to achieve adequate loop bandwidth. Even in those cases however, it makes sense to write the majority of the program in a language like C and write only the speed critical sections of the program in assembly language.

The control program for the submersible thruster drive consists of about 180 lines of C language code (ignoring comments) and about 40 lines of assembly language code. There are about another 60 lines of assembly language code in the support functions. Of the C language code, about 80 lines consist of definitions and declarations which don't generate any machine language. The machine language program, produced after compiling, assembling and linking the source programs, fits into

less than 2K bytes of EPROM. The program for the linear induction motor drive application barely fits into 4K bytes of EPROM after some rewriting of the program to make it more memory efficient. Future applications will definitely need more EPROM space on the controller card.

The advantages of a high level language are in the time required to develop a program and in the ability to maintain and modify the program. The control program for the submersible thruster drive took about a month to write and debug. This is in line with published estimates that programs are produced at a rate of about 10 lines per day. When the program for the linear induction motor drive was written, much of the program for the submersible thruster drive could be used directly, so this program also took about a month to write despite the fact that it is about twice as long as the program for the submersible thruster drive. If the programs had been written in assembly language it is probable that the development time for the submersible thruster program would have been at least two months and that it would have been considerably more difficult to use this program as a basis for the development of the linear induction motor drive program. Certainly the top down, modular structure of the programs would be more difficult to achieve if the programs were written in assembly language.

Debugging a program written in a high level language is different than debugging a program written in assembly language. In the

absence of high level debugging tools that are integrated with the compiler, there is little to be gained from the use of in-circuit emulators, single stepping, and breakpoints since the results don't mean much in terms of the high level language program. In cases where the target system has an interface to a printer or a terminal, the high level language program can be "instrumented" with statements to print out information on the execution of the program. In the case of the inverter controller this facility was not available. Instead, debugging consisted of thinking carefully about the observed problems in the operation of the controller, relating them to a section of the program which could cause the problem, finding and correcting the error, and then reloading the recompiled program into EPROM to verify that the diagnosis was correct.

The process was actually less painful than it appears. Since the high level language program was shorter than the equivalent assembly language program, and allowed the program design to be expressed more naturally, there were fewer program errors to start with. The high level language program was easier to read and understand, and was better structured than the average assembly language program. Thus it was easier to isolate the section of the program responsible for a problem and easier to find the program error in that section of program. Finally, the need to recompile the program and reprogram the EPROM encouraged an organized, efficient approach to debugging and proper, documented repairs. In contrast, the fast turn-around possible

when debugging in assembly language often encourages trial and error approaches to debugging and the use of quick, undocumented machine language "patches" to correct errors.

On balance, the decision to use a high level language in this project seems to be justified. The programs produced to date meet the speed requirements of the applications. More demanding applications can be dealt with by using a faster version of the 8085 microprocessor, redesigning the controller to use a more powerful microprocessor, or writing some speed critical portions of the program in assembly language. Based on experience with previous assembly language programming projects, development of a high level language program proceeds at least twice as quickly as development of the equivalent assembly language program. The economic implications are obvious.

Chapter 6

Applications of the Inverter Controller

So far the inverter controller has been used in two different applications. The first is the submersible thruster drive which was the original motivation for the design of the controller. The second is a controller for a linear induction motor drive. This chapter is devoted to describing these two drives since they demonstrate the flexibility and practicality of the inverter controller in realistic applications.

Both drives make use of the wide frequency range of the controller. The submersible drive operates over a range of 3 to 400 Hz and the linear induction motor drive runs over a range of 3 to 600 Hz. The small size and low parts count of the inverter controller are particular advantages in the submersible application where compactness and reliability are extremely important. In the linear induction motor drive, the programmability of the controller is a strong asset since the drive is used in experimental work where parameters such as maximum inverter frequency, volts/Hz ratio, current limit setpoint, and acceleration and deceleration rates must be changed for different machines.

The two drives are both viable products that can compete effectively with commercially available alternatives. In the

case of the submersible thruster drive, there do not appear to be any other thruster drives based on induction motors on the market. The closest competitors are drives based on brushless DC motors. The AC motor drive is smaller and cheaper than the brushless DC drive. In addition, the inverter controller allows for considerably more flexibility in drive characteristics than the less sophisticated controllers used by brushless DC drives. The only commercially available inverter that was identified as possibly suitable for the linear induction motor drive is based on outdated technology. In terms of cost and performance it cannot compete with the drive described in this chapter.

Both drives currently exist as prototypes which have been extensively lab tested. Both are in the process of being designed into commercial systems.

6.1 Submersible Thruster Drive

The submersible thruster drive (see Figure 36) is a small drive rated at 1.5 HP. A compact 400 Hz induction motor, originally designed for aircraft applications, is used in conjunction with a power MOSFET inverter. A gear reduction is used between the motor shaft and the propellor shaft of the drive. As mentioned in Chapter 3, the inverter and motor can be mounted in a common oil filled housing. Alternatively, the inverter and motor can be installed in separate housings on the submersible. The controller card is installed in the submersible's electronics

canister which is filled with air at one atmosphere pressure. All signals between the inverter and the controller pass through opto-isolators so the inverter and controller are electrically isolated from each other.

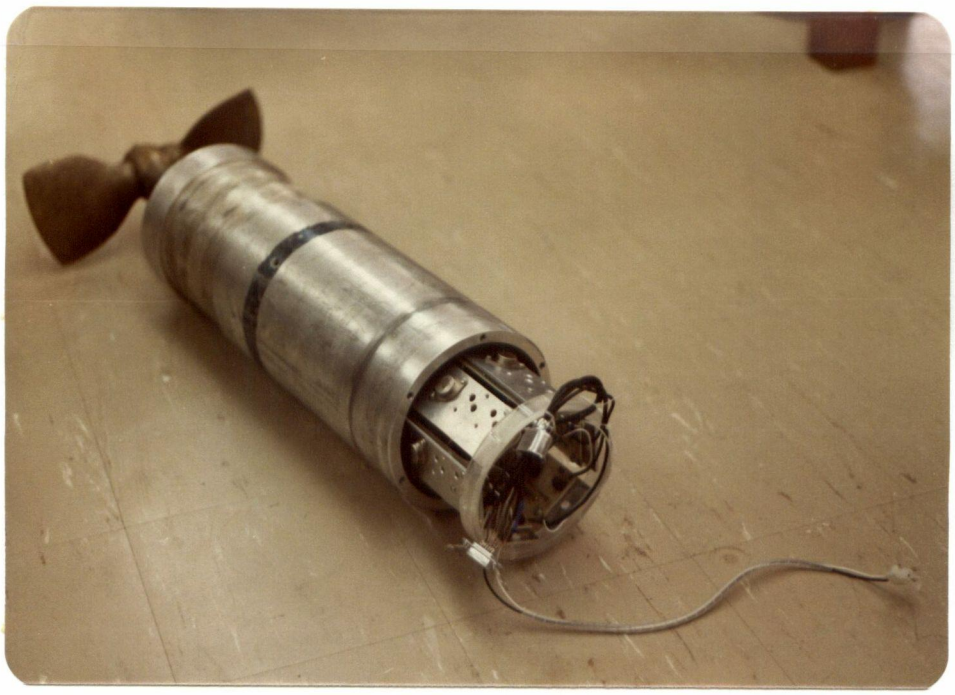


Figure 36 Submersible Thruster Drive

The control strategy for this drive is simple, using open-loop constant volts/Hz control of stator voltage and fixed acceleration and deceleration rates. In a thruster application, a more complex control strategy is unnecessary. In the prototype, the motor speed command is entered via a slide potentiometer connected to the A/D converter. The output of the A/D converter is interpreted as a signed binary number in offset

binary format. Thus the potentiometer is used to command speeds in both directions of rotation. The software that reads and interprets the speed command is in one distinct module which can be modified to deal with speed commands in the format supplied by the designers of the submersible.

In addition to the slide potentiometer, the controller for the prototype drive is also equipped with a switch to turn the inverter on and off and a pushbutton that causes a hardware reset of the microcomputer in the controller. Three LED lamps are installed to indicate that the controller is on, that the inverter is on, and that a fault has occurred.

Since acceleration and deceleration rates are fixed and the inertia of the load does not change, protection against overcurrent due to too rapid a change in inverter frequency is not required. Also, no significant regeneration occurs when the drive decelerates since the load is dissipative rather than inertial. As a result no protection is required against an overvoltage on the DC bus due to regenerated energy. Of the faults which can occur, the most common is jamming of the propellor by a stray piece of rope or fishnet. The inverter is equipped with a fast acting current limit circuit that shuts off the switching devices and sends a fault signal to the inverter controller if the drive jams. The inverter controller software enters the FAULT state on receipt of the fault signal from the inverter and disables the inverter switching signals. In some

versions of the software, the controller exits the FAULT state when the speed command is reduced to zero so that the operator can attempt to restart the drive once the propellor is unjammed.

The thruster drive has been through an extensive series of tank tests. The drive has been operated at a constant speed continuously for two weeks. It has also been continuously cycled from full speed in one direction to full speed in the other direction for a period of three days. Finally, a rope has been thrown into the propellor of the thruster to cause the drive to jam. The drive shut down properly and was restarted once the rope was removed.

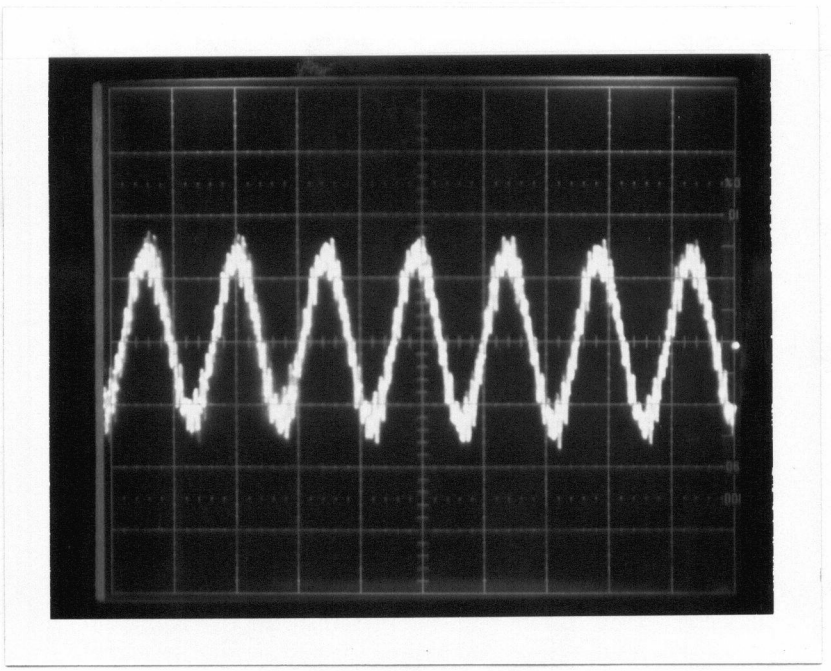


Figure 37 Motor Line Current (Vertical: 5 A/div Horizontal: 2 msec/div)

Figure 37 shows the motor line current for the thruster drive. As can be seen, the waveform is basically sinusoidal. When the

drive is operated at very low speeds no cogging or vibration can be observed, indicating that harmonic torques are very small.

Representatives of International Submarine Engineering who have seen demonstrations of the thruster drive have been impressed by its small size and good performance. They particularly liked the smooth low speed performance of the drive; apparently the drives they are currently using do not perform well at low speeds. At present, International Submarine Engineering is designing a new sumersible which will make use of this drive.

6.2 Linear Induction Motor Drive

Cetec Engineering of Burnaby B.C. is developing linear induction motors (LIMs) for use in industrial equipment such as circular saws, shakers, and grinders. Due to size constraints on the motors, the maximum pole pitch on the LIMs is limited. Therefore, to attain the required speeds on the driven portion of the machine, the LIM stator frequency must be relatively high - up to 600 Hz in some cases. In some equipment the LIM speed must be variable. These two requirements suggest that a variable frequency inverter be used as the LIM power supply.

Cetec attempted to find a commercially available inverter that would meet their requirements. Most inverters were not capable of operating above 120 Hz. One series of inverters was found that operated to 400 Hz but it was based on thyristor switches

and used a rather inefficient complementary commutation scheme to switch off the devices. At 400 Hz the commutation losses were so high that the inverter output power rating was substantialy decreased. As a result the inverters having the required output power rating were large and expensive. Cetec decided to fund development of an inverter, using more modern technology, that would be smaller and less expensive and would have the flexibility required for their development program.

The first version of the inverter was rated at 25 KVA; a 160 KVA version is now being built. The inverter is of the PWM type and uses transistors as the power switching devices. The inverter controller described in this thesis is used to control the inverter. The control strategy employed is basically an open-loop constant volts/Hz type. However, the controller can be switched into a closed-loop speed control mode in which the controller changes the inverter frequency so that the difference between a setpoint level from a potentiometer and a feedback signal from a speed sensor is smaller than some preset error band. This control mode is intended for some LIM applications where the motor speed must be kept approximately constant with changes in load. The speed control loop is not particularly accurate or fast but it should meet the requirement of the LIM application.

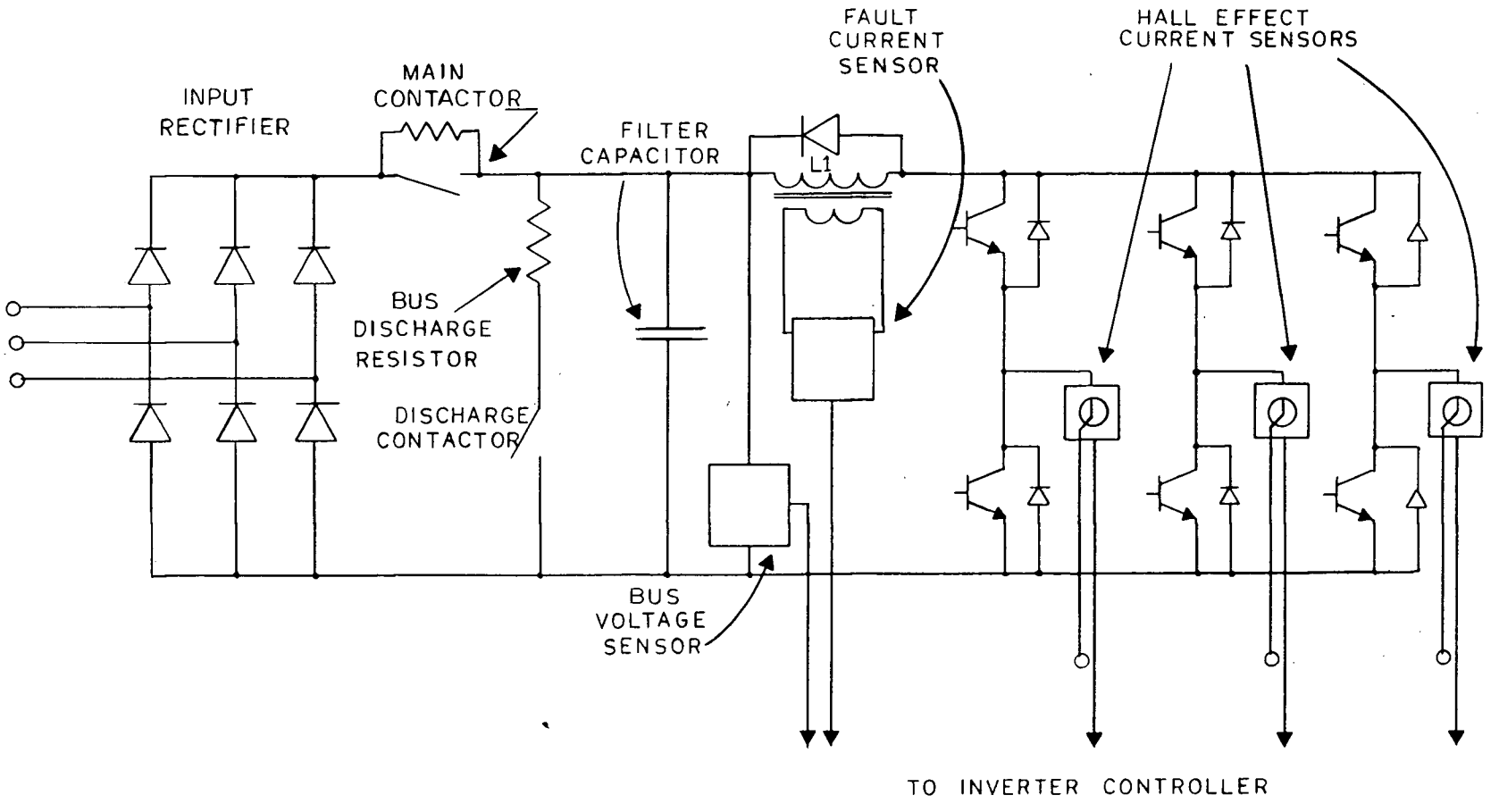


Figure 38 Simplified Schematic of LIM Inverter

6.3 LIM Drive Protective Functions

A simplified schematic of the inverter is shown in Figure 38. The protective circuitry in this inverter is considerably more complex than in the inverter for the submersible drive. A sensor across the DC bus monitors the bus voltage. If the bus voltage exceeds a maximum limit, the sensor sends a fault signal to the inverter controller. The inverter controller then switches load resistors across the DC bus to absorb regenerated energy. Once the bus voltage has dropped below a lower limit, the bus voltage sensor removes the fault signal and the inverter controller disconnects the load resistors from the DC bus.

Protection against short circuits and shoot throughs is provided by inductor L1. It limits the rate of rise of the fault current and, through the secondary coil on the inductor, sends a fault signal to the inverter controller. This fault signal goes to the TRAP interrupt input of the microcomputer. When the TRAP interrupt occurs, the first action taken by the microcomputer is to switch off all the inverter switching devices. After that, the controller enters the FAULT state until the controller is reset. The time from the initiation of a fault to the removal of inverter switching signals is very short - about 15 to 20 microseconds. The design of the current limiting inductor (L1) is coordinated with the overcurrent characteristics of the inverter transistors, and the time required to switch them off, to ensure that fault currents do not rise to damaging levels

before the transistors are able to turn off.

The TRAP interrupt input also receives a signal from an Emergency Stop pushbutton. This pushbutton can be used by the operator in situations where the drive must be shut down immediately. The normal inverter on/off switch causes a controlled shut down in which the inverter controller first decelerates the machine to a standstill. In some emergencies that can take too long.

The Hall Effect current sensors on the inverter output lines provide protection against inverter overloads. The current sensor signals are rectified and summed to produce a single current signal which is sent to one of the A/D channels on the inverter controller. The inverter controller compares this current feedback signal to a preset limit. If an overcurrent is detected, the controller takes corrective action.

The nature of the protective action depends on the magnitude of the overcurrent and the current state of the controller. For example, if the overcurrent is relatively small and the controller is in the ACCELERATING state, the controller will simply decrease the rate of change of inverter frequency. On the other hand, if the overcurrent is large, the controller will shut down the inverter.

6.4 LIM Drive Control Panel Functions

The LIM drive inverter is designed for use in research and development. As a result it has provision for many more adjustments than the submersible thruster drive or a standard industrial variable speed drive. The control panel is equipped with a ten turn potentiometer to set the inverter frequency, a switch to set the phase sequence, and a switch to turn the inverter on or off. In addition, provision is made to set the maximum inverter frequency, the volts/Hz ratio, the acceleration and deceleration rates, and the current limit setpoint. These adjustments are made by potentiometers which are connected to the A/D channels of the inverter controller.

The maximum inverter frequency can be set between 90 and 600 Hz. The potentiometer that sets the inverter frequency has a range from the minimum frequency (3 Hz) to the maximum frequency. The frequency resolution is maintained at 0.5 Hz. internally but, because of the 8 bit resolution of the A/D converters which read the potentiometers, the frequency can only be set with a resolution of 2 or 3 Hz when the maximum frequency limit is 600 Hz. This does not present a problem in the applications in which this inverter is used. The volts/Hz ratio can be adjusted so that the inverter PWM waveform reaches 100% modulation over a range from 50 Hz to 500 Hz. The acceleration and deceleration rates, which are really rates of change of inverter frequency, can be adjusted from 10 Hz/second to 200 Hz/second.

In order to display the various setpoints and data such as the inverter frequency and line current, an alphanumeric display is installed in the control panel. The display is actually a commercially available keypad/display card intended for STD bus computer systems. The card is interfaced to the microcomputer address, data, and control lines which are brought to connectors on the inverter controller card. The keypad on the display card is used to select the data to be displayed. The alphanumeric display is also used to display diagnostic messages when an inverter fault occurs.

6.5 Evaluation of the LIM Drive

The 25 KVA version of the LIM drive inverter has been in use at Cetec's laboratory for the past four months. To date the unit has performed satisfactorily. All the protective functions work as designed and have actually protected the inverter against faults in the linear induction motor. The control panel adjustments and the display also work properly. The closed-loop speed control has not been tested yet because no speed sensor has been installed on any of the linear induction motors.

It is interesting to compare this design to a modern commercial design. The Mitsubishi Freqrol-E [80,81] is a modern transistor inverter that is similar in many respects to the 25 KVA LIM inverter. In fact, the same transistors are used in the two

inverters. The Mitsubishi inverter has been successful in the market due to its compactness and good price/performance ratio. It is representative of the state of the art in general purpose AC drives.

The Mitsubishi inverter operates over a frequency range of 6 to 60 Hz or 6 to 120 Hz with resolutions of 0.5 Hz and 1 Hz respectively. This frequency range is considerably less than that of the LIM inverter but the frequency resolution is comparable. The Mitsubishi inverter allows selection of 16 different volts/Hz ratios. The range of ratios that can be selected is considerably smaller than the range possible with the LIM inverter. The acceleration and deceleration rates of the Mitsubishi inverter can be varied over a wide range from about 120 Hz/second to less than 1 Hz/second. The LIM inverter has a narrower range since the very slow rates of change were not required. However, the inverter controller could easily be programmed to produce lower rates of change.

The protective functions incorporated in the Mitsubishi inverter are very similar to those of the LIM inverter. This is not entirely accidental since the design of the LIM inverter is based on industrial practice. The Mitsubishi inverter does not have the display capabilities of the LIM inverter. An analog output proportional to frequency is supplied which can drive a meter. In addition, some indicator lamps are used to indicate fault conditions.

From the preceding discussion it is clear that the LIM inverter is competitive in performance and features with commercial drives. When the inverter controllers of the Mitsubishi drive and the LIM drive are compared, some of the advantages of the LIM drive become apparent. The Mitsubishi inverter controller is based on a custom LSI IC but the controller board is still about three times larger than the controller board for the LIM drive and contains many more components. Despite the use of LSI circuits, many of the controller functions are still carried out by analog integrated circuits and small scale digital integrated circuits. This reliance on hardware makes the Mitsubishi inverter far less flexible than the LIM inverter. New control modes such as closed-loop speed control can't be added without redesigning the hardware. In contrast the controller in the LIM drive can be reprogrammed rather easily since most control functions are carried out by microcomputer software. Finally, the Mitsubishi inverter has no provision for interfacing to a digital system. As a result it is hard to integrate it into systems which operate under computer control. In contrast, the controller in the LIM inverter can be readily interfaced to digital systems and can be programmed to work under the control of a supervisory computer.

As mentioned, a 160 KVA version of the LIM inverter is currently being built. It uses the same controller as the 25 KVA inverter. This inverter will be used in field tests of Cetec's equipment.

Assuming successful completion of field tests, a simplified version of this inverter, without all the adjustment capabilities, will be used in production versions of Cetec's equipment.

6.6 Conclusion

The two applications discussed in this chapter illustrate the wide range of AC drives that can be designed using the inverter controller described in this thesis. The two applications span a power range from 1.5 KW to 160 KW and involve the use of both power MOSFETs and transistors as the inverter power switches. Control panel and protection functions range from very simple in the submersible thruster to quite complex in the LIM drive. However these two applications do not come close to exhausting the capabilities of the inverter controller. Drives using other control strategies such as controlled slip and other switching devices such as thyristors can be designed using this inverter controller.

Chapter 7

Conclusion

An inverter controller has been described which has the following features:

i. The controller controls PWM inverters using transistors, power MOSFETs, or thyristors as switches.

ii. It operates over an inverter frequency range of at least 2 Hz to 600 Hz.

iii. Most of the control functions, with the exception of the generation of the PWM waveform, are performed by microcomputer software written in the C programming language.

iv. The pulse width modulated inverter switching waveforms are generated by a large scale integrated circuit which is controlled by the microcomputer in the inverter controller.

v. The inverter controller is contained on a single 7.5 cm by 21.5 cm (161 cm^2) printed circuit board. The cost of parts and direct labor for the controller is under $200.

The inverter controller meets or exceeds the design objectives set out for it in Chapter 1 and Chapter 3. It has been used successfully in two applications; a submersible thruster drive and a linear induction motor drive. Overall, this inverter controller design must be considered a success.

The approach taken to the design of this inverter controller is different than the approach taken in much of the literature on PWM inverter controllers published in the last three or four years. Many of these papers (see, for example, papers by Green and Boys [81], Rajashekara and Vithayathil [82], Pollman [83], Varnovitsky [84], and Buja and Fiorini [85]) concentrate on the design of the PWM waveform generator, studiously ignoring the availability of a perfectly satisfactory PWM generator in LSI form. As a result, the PWM generator dominates the design both in terms of component count and design effort. The other functions required in an inverter controller often seem to be added as an afterthought, if they are included at all.

In contrast, a more balanced and "top down" approach was taken in the design described in this thesis. Design started with a careful definition of all the objectives that the design had to meet and then proceeded with an evaluation of the methods available to meet these objectives. The evaluation placed emphasis on flexibility, compactness, and economy rather than on sheer performance or on novelty.

Another aspect of the design which distinguishes it from many of the designs in the literature is the way that the inverter controller can be rapidly adapted to new applications. This capability was achieved by a series of conscious design decisions throughout the design process. During the initial system design, the design was partitioned so that most controller functions are carried out by microcomputer software with the controller hardware perfroming support functions. The microcomputer in the controller was chosen with an eye to the availability of good, low cost software development tools. Finally, the decision was made to write the software in the C programming language in order to speed the program development process.

The flexibility of this inverter controller is not achieved entirely without cost. The software intensive nature of the controller, the use of a microcomputer that has been available long enough to have good software development tools, and the use of a high level programming language, combine to make the controller slower than it otherwise might be. This will primarily affect the controller's performance in applications involving closed-loop control. However the dynamics of many AC drives are dominated by a high inertia load. In these cases a fast control loop is not required so the inverter controller can be used as a closed-loop controller.

On the hardware side, this inverter controller is distinguished by its compactness, achieved through the use of large scale

integrated circuits. As noted in the previous chapter, this controller is approximately one third the size of the controller for a modern commercial transistor inverter, yet is capable of better performance. However, the rapid development in integrated circuit technology since the hardware design was completed now allows for some substantial improvements in the hardware design. As discussed in Chapter 4, the frequency synthesizer that generates control signals for the PWM waveform generator IC can be substantially simplified by using a new silicon gate version of the 4046 phase locked loop IC. In addition, the 4K byte 2732 EPROM can be replaced by a higher density EPROM such as the 2764 or 27128.

A more drastic change would be to replace the Intel 8085 microprocessor with an Intel 80188 microprocessor. The 80188 has better computational capabilities than the 8085, in particular it has multiply and divide operations in its instruction set. These operations are particularly useful in closed-loop control algorithms. In addition, the 80188 has built in counter/timers that would be very useful when interfacing the controller to sensors such as optical shaft encoders. Since the software for the inverter controller is written primarily in the C programming language, the transfer of software to the new microprocessor should be fairly straightforward.

Another possible improvement is to redesign the inverter controller printed circuit board to conform to the configuration

for a microcomputer system bus such as the STD bus. This would allow the controller to be readily interfaced to microcomputer cards and various I/O cards. Some interesting inverter controller designs would then be possible since some of the controller functions could then be transferred from the controller card to another microcomputer, thereby improving system throughput. For example, an AC servo drive could be designed with the controller card carrying out the PWM waveform generation and system protection functions under control of a microcomputer which would carry out the closed-loop control functions. The scalar decoupled control algorithm recently described by Bose [86] would be suitable for a controller with this type of architecture.

There is also considerable potential for further work with the controller operating in its stand-alone form. Since the controller is easily reprogrammed, it can be used in a variety of AC drive applications. For example, a controlled slip control scheme could be implemented for traction applications such as electric automobile or trolley bus drives. The maintenance and reliability advantages of a single board inverter controller would be as attractive in these traction applications as in the submersible thruster drive application.

References

1. B.K. Bose, "Adjustable Speed AC Drives - A Technology Status Review", IEEE Proceedings, vol. 70, No. 2, pp. 116-135, February 1982

2. F. Peabody and K. Mauch, "A.C. Variable Speed Drives For Specialty Applications", Final Report on B.C. Science Council Project #56 (RC-2), September 1981

3. F. Peabody, "A.C. Variable Speed Drives For Specialty Applications", Final Report on B.C. Scince Council Project #37 (RC-5), September 1982

4. K.P. Phillips, "Current Source Converter for AC Motor Drives", IEEE Transactions on Industry Applications, vol. IA-8, no. 6, pp. 679-683, Nov./Dec. 1972

5. M.G. Say, Alternating Current Machines, New York: John Wiley & Sons, 1976, p. 254

6. V.R. Stefanovic "Static and Dynamic Characteristics of Induction Motors Operating Under Constant Airgap Flux Control", in Conference Record of IEEE 1976 Industry Applications Society Annual Meeting, pp. 436-444

7. F. Blaschke, "The Principle of Field Orientation as Applied to the New TRANSVEKTOR Closed-Loop Control System for Rotating-Field Machines", Siemens Review, vol. 34, pp. 217-220, May 1972

8. A. Plunkett, "Direct Flux and Torque Regulation in a PWM Inverter-Induction Motor Drive", IEEE Trans. Ind. Appl., vol. IA-13, no. 2, pp. 139-146, March/April 1977

9. A. Abbondanti, "Method of Flux Control in Induction Motors Driven By Variable Frequency, Variable Voltage Supplies", in Conference Record of IEEE/IAS 1977 International Semiconductor Power Conversion Conference, pp. 177-184

10. B. Mokrytzki, "The Controlled Slip Static Inverter Drive", IEEE Trans. Ind. Gen. Appl., vol. IGA-4, no. 3, pp. 312-317, May/June 1968

11. P.D. Agarwal, "The GM High-Performance Induction Motor Drive System", IEEE Trans. Power Apparatus and Systems, vol. PAS-88, no. 2, pp. 86-93, Feb. 1969

12. A.K. Wallace, J.H. Parker, and G.E. Dawson, "Slip Control for LIM Propelled Transit Vehicles", IEEE Transactions on Magnetics, vol. MAG-16, no. 5, pp. 710-712, September 1980

13. S.B. Dewan and S.A. Mirbod, "Slip Speed Control in an Induction Motor Drive with a Phase Locked Loop", in Conference Record of IEEE 1979 Industry Applications Society Annual Meeting,

pp. 952-955

14. A. Abbondanti and M.B. Brennan, "Variable Speed Induction Motor Drives Use Electronic Slip Calculator Based on Motor Voltages and Currents", IEEE Trans. Ind. Appl., vol. IA-11, no. 5, pp. 483-488, Sept./Oct. 1975

15. R.G Schieman, E.A. Wilkes, and H.E. Jordan, "Solid-State Control of Electric Drives", IEEE Proceedings, vol. 62, no. 12, pp. 1643-1660, Dec. 1974

16. S.A. Rosenberg, S.B. Dewan, and G.R. Slemon, "Inverter Fed Induction Motor Drive Using Power Factor Control", in Conference Record of IEEE 1976 Industry Applications Society Annual Meeting, pp. 810-813

17. A.E. Fitzgerald, C. Kingsley, and A. Kusko, Electric Machinery 3rd ed., New York: McGraw-Hill, 1971, p. 350

18. R.B. Maag,"Characteristics and Application of Current Source Slip Regulated AC Induction Motor Drives" in Conference Record of IEEE 1971 Industry and General Applications Society Annual Meeting, pp. 411-413

19. E.P. Cornell and T.A. Lipo, "Modelling and Design of Controlled Current Induction Motor Drive Systems," IEEE Trans. Ind. Appl., vol. IA-13, no. 4, pp. 321-330, July/Aug. 1977

20. B. Adkins, The Generalized Theory of Electrical Machines, London: Chapman & Hall, 1957

21. P.C. Krause and C.H. Thomas, "Simulation of Symmetrical Induction Machinery", IEEE Trans. Power App. Syst., vol. PAS-84, no. 11, pp. 1038-1053, Nov. 1965

22. T.A. Lipo and A.B. Plunkett, "A Novel Approach to Induction Motor Transfer Functions", in Conference Record of IEEE 1973 Industry Applications Society Annual Meeting, pp. 451-457

23. M.L. MacDonald and P.C. Sen, "Control Loop Study of Induction Motor Drives Using DQ Model", in Conference Record of IEEE 1978 Industry Applications Society Annual Meeting, pp. 897-903

24. R. Stern and D.W. Novotny, "A Simplified Approach to the Determination of Induction Machine Dynamic Response", IEEE Trans. Power App. Syst., vol. PAS-97, no.4, pp. 1430-1439, May 1978

25. F. Fallside and A.T. Wortley, "Steady State Oscillation and Stabilisation of Variable Frequency Inverter-Fed Induction Motor Drives", Proceedings of the IEE, vol. 116, no. 6, pp. 991-999, July 1969

26. T.A. Lipo and P.C. Krause, "Stability Analysis of a Rectifier-Inverter Induction Motor Drive", IEEE Trans. Power App. Syst., vol. PAS-88, no. 1, pp. 55-66, January 1969

27. A. Kawamura and R. Hoft, "An Analysis of Induction Motor Field Oriented

or Vector Control", in Conference Record of IEEE 1983 Power Electronics Specialists Conference, pp. 91-101

28. E.A. Klingshirn and H.E. Jordan, "Polyphase Induction Motor Performance and Losses on Nonsinusoidal Voltage Sources", IEEE Trans. Power App. Syst., vol. PAS-87, no. 3, pp. 624-631, March 1968

29. S.D.T. Robertson and K.M. Hebbar, "Torque Pulsations in Induction Motors with Inverter Drives", IEEE Trans Ind. and Gen. Appl., vol. IGA-7, no. 2, pp. 318-323, March/April 1971

30. J.M.D. Murphy, Thyristor Control of AC Motors", Oxford: Pergamon Press, 1973, p. 100

31. B.J. Chalmers and B.R. Sarkar, "Induction Motor Losses Due to Nonsinusoidal Supply Waveforms", Proceedings of the IEE, vol. 115, pp. 1777 - 1782, December 1968

32. V.B. Honsinger, "Induction Motors Operating from Inverters", in Conference Record IEEE 1980 IAS Annual Meeting, pp. 1276-1285

33. P.L. Alger, Induction Machines 2nd ed., New York: Gordon and Breach, 1970, pp. 265-272

34. G.B. Kliman, "Harmonic Effects in Pulse Width Modulated Inverter Induction Motor Drives", in Conference Record IEEE 1972 IAS Annual Meeting, pp. 783-790

35. A. Schonung and H. Stemmler, "Static Frequency Changers With "Subharmonic" Control in Conjunction With Reversible Variable Speed Drives", The Brown Boveri Review, pp. 555-577, Aug./Sept. 1964

36. J.W.A. Wilson and J.A. Yeamans, "Intrinsic Harmonics of Idealized Inverter PWM Systems", in Conference Record IEEE 1976 IAS Annual Meeting, pp. 967-973

37. J. Zubek, A. Abbondanti, and C.J. Nordby, "Pulse Width Modulated Inverter Motor Drives With Improved Modulation", IEEE Trans. Ind. Appl., vol IA-11, no. 6, pp. 695-703, Nov./Dec. 1975

38. J.W.A. Wilson, "Adaptation of Pulse Width Modulation Theory For Use in AC Motor Drive Inverters", in Conference Record of IEEE 1977 IAS International Semiconductor Power Converter Conference, pp. 193-197

39. G.B. Kliman and A.B. Plunkett, "Development of a Modulation Strategy For a PWM Inverter Drive", IEEE Trans. Ind. Appl., vol. IA-15, no. 1, pp. 72-79, Jan./Feb 1979

40. D.A. Grant, "A Technique for Pulse Dropping in Pulse-Width-Modulated Inverters", Proc. IEE, B, Electr. Power Appl., vol. 128, pp. 67-72, 1981

41. H.S. Patel and R.G. Hoft, "Generalized Techniques of Harmonic

Elimination and Voltage Control in Thyristor Inverters: Part I - Harmonic Elimination", IEEE Trans. Ind. Appl., vol. IA-9, no. 3, pp. 310-317, May/June 1973

42. H.S. Patel and R.G. Hoft, "Generalized Techniques of Harmonic Elimination and Voltage Control in Thyristor Inverters: Part II - Voltage Control Techniques", IEEE Trans. Ind. Appl., vol IA-10, no. 5, pp. 666-673, Sept./Oct. 1974

43. G.S. Buja and G.B. Indri, "Optimal Pulsewidth Modulation for Feeding AC Motors", IEEE Trans. Ind. Appl., vol. IA-13, no. 1, pp. 38-44, Jan./Feb 1977

44. A. Pollman, "A Digital Pulsewidth Modulator Employing Advanced Modulation Techniques", IEEE Trans. Ind. Appl., vol IA-19, no. 3, pp. 409-414, May/June 1983

45. J.M.D. Murphy and M.G. Egan, "A Comparison of PWM Strategies for Inverter Fed Induction Motors", IEEE Trans. Ind. Appl., vol. IA-19, no. 3, pp. 363-369, May/June 1983

46. R. Gabriel, W. Leonhard, and C. Nordby, "Field-Oriented Control of a Standard AC Motor Using Microprocessors", IEEE Trans. Ind. Appl., vol. IA-16, no. 2, pp. 186-192, March/April 1980

47. R. Gabriel and W. Leonhard, "Microprocessor Control of Induction Motor", in Conference Record 1982 International Semiconductor Power Converter Conference, pp. 385-396

48. P. van der Gracht and K. Mauch, "A Microprocessor Controlled Three-Phase Power Inverter", Computer Design, pp. 120-122 March 1977

49. A. Kusko and D. Galler, "Survey of Microprocessors in Industrial Motor Drive Systems", in Conference Record IEEE Industry Applications Society 1982 Annual Meeting, pp. 435-438

50. R. Stern and D.W. Novotny, "A Simplified Approach to the Determination of Induction Machine Dynamic Response", IEEE Trans. Power App. Syst., vol. PAS-97, no.4, pp. 1430-1439, May 1978

51. G.F. Franklin and J.D. Powell, Digital Control of Dynamic Systems, Reading MA: Addison-Wesley, 1980, pp. 285-287

52. P. Katz, Digital Control Using Microprocessors, Englewood Cliffs NJ: Prentice-Hall, 1981, p. 236

53. J. Casteel and R. Hoft, "Optimum PWM Waveforms of a Microprocessor Controlled Inverter", in Proceedings 1978 IEEE Power Electronics Specialists Conference, pp. 243-250

54. S.R. Bowes and M.J. Mount, "Microprocessor Control of PWM Inverters", IEE Proceedings, vol. 128, part B, no. 6, pp. 293-305, November 1981

55. M. Varnovitsky, "A Microcomputer-Based Control Signal Generator for a

Three-Phase Switching Power Inverter", IEEE Trans. Industry Applications, vol. IA-19, no. 2, pp. 228-234, March/April 1983

56. B.K. Bose and H. Sutherland, "A High-Performance Pulsewidth Modulator for an Inverter-Fed Drive System Using a Microcomputer", IEEE Trans. Industry Applications, vol. IA-19, no. 2, pp. 235-243, March/April 1983

57. K. Mauch and M.R. Ito, "A Multimicroprocessor AC Drive Controller", in Conference Record IEEE Industry Applications Society 1980 Annual Meeting, pp. 634-640

58. W.J. Tuten, "Microprocessor Controller for Integrated Power Module Inverter", in Conference Record of the 1977 IEEE/IAS International Semiconductor Power Converter Conference, pp. 471-476

59. E. Dwyer and B.T. Ooi, "A Lookup Table Based Microprocessor Controller for a Three Phase PWM Inverter" in Conference Record of the IEEE/IECI 1979 Conference on Industrial Applications of Microprocessors, pp. 19-22

60. L. Humblet, F. De Buck, B. Verbecke, P. De Valck, " A Realisation Example of a Microprocessor Driven PWM Transistor Inverter", in Conference Record of IEE Power Division 2nd International Conference on Electrical Variable Speed Drives, pp. 151-156, 1979

61. H.S. Patel, "Thyristor Inverter Harmonic Elimination Using Optimization Techniques", Ph.D. Dissertation, University of Missouri-Columbia, 1971

62. Signetics Corporation, Phillips AC Motor Controller Product Information, April 1981

63. S.R. Bowes and B.M. Bird, "Novel Approach to the Analysis and Synthesis of Modulation Processes in Power Convertors", Proceedings of the IEE, vol. 122, no. 5, pp. 507-513, May 1975

64. S. Morinaga et. al., "Microprocessor Control System with I/O Processing LSI for Motor Drive PWM Inverter", in Conference Record of IEEE/IAS 1981 Annual Meeting, pp. 1197-1202

65. Philips LOCMOS HE4000B I.C. Family, Signetics Corp., Sunnyvale CA, p. 689, September 1981

66. F.M. Gardner, Phaselock Techniques, 2nd ed., New York: Wiley, 1979, Chapter 6

67. Am9500 Peripheral Products Interface Guide, Advanced Micro Devices, Sunnyvale CA, p. 4-48, 1980

68. COS/MOS Integrated Circuits, RCA Corp., Somerville NJ, p. 179, 1980

69. 1981 Supplement to the TTL Data Book for Design Engineers, Texas Instruments Inc., Dallas TX, p. 149, 1981

70. "The RCA COS/MOS Phase-Locked-Loop: A Versatile Building Block for Micro-Power Digital and Analog Applications", RCA Applications Note ICAN-6101, RCA Corp., Somerville NJ, 1981

71. B.W. Kernighan and D.M. Ritchie, The C Programming Language, Englewood Cliffs NJ: Prentice-Hall, 1978

72. J. Gilbreath, "A High-Level Language Benchmark", BYTE, vol. 6, no. 9, pp. 180-198, September 1981

73. J. Gilbreath and G. Gilbreath, "Eratosthenes Revisited: Once More through the Sieve", BYTE, vol. 8, no. 1, pp. 283-326, January 1983

74. G.E. Anderson and K.C. Shumate, "Selecting a Programming Language, Compiler, and Support Environment: Method and Example", Computer, vol. 15, no. 8, pp. 29-36, August 1982

75. S.T. Allworth, Introduction to Real-Time Software Design, New York: Springer Verlag, 1981, pp. 100-109

76. J.V. Landau, "State Description Techniques Applied to Industrial Machine Control", Computer, vol. 12, no. 2, pp. 32-40, February 1979

77. J.G Gander and H.U. Liechti, "State Language for Real-Time Process Control", Microprocessors and Microsystems, vol. 5, no. 1, pp. 27-28, Jan/Feb 1981

78. H.A. Sutherland, B.K. Bose and C.B. Somuah, "A State Language for Sequencing in a Hybrid Electric Vehicle", IEEE Trans. Ind. Electronics, vol. IE-30, no. 4, pp. 318-322, November 1983

79 Mitsubishi VVVF Transistor Inverter Freqrol-E Service Manual, Mitsubishi Electric Corp., Tokyo

80. E. Ohno et. al., "General Purpose Variable Frequency Inverter Using Integrated Power Module and LSI", in Conference Record IEEE 1982 Power Electronics Specialists Conference, pp. 478-487

81. R.M. Green and J.T. Boys, "Implementation of Pulsewidth Modulated Inverter Modulation Strategies", IEEE Trans. Ind. Applications, vol. IA-18, no. 2, pp. 138-145, March/April 1982

82. K.S. Rajashekara and J. Vithayathil, "Microprocessor Based Sinusoidal PWM Inverter by PWM Transfer", IEEE Trans. Ind. Electronics, vol. IE-29, no. 1, pp. 46-51, Feb. 1982

83. A. Pollman, "A Digital Pulsewidth Modulator Employing Advanced

Modulation Techniques", IEEE Trans. Ind. Applications, vol. IA-19, no. 3, pp. 409-413, May/June 1983

84. M. Varnovitsky, "A Microcomputer-Based Control Signal Generator for a Three-Phase Switching Power Inverter", IEEE Trans Ind. Applications, vol. IA-19, no. 2, pp. 228-234 March/April 1983

85. G.S. Buja and P. Fiorini, "Microcomputer Control of PWM Inverters", IEEE Trans. Ind. Electronics, vol. IE-29, no.3, pp. 212-216, August 1982

86. B.K. Bose, "Scalar Decoupled Control of Induction Motors", IEEE Trans. Ind. Applications, vol. IA-20, no. 1, pp. 216-225, January/February 1984

Appendix

Control Program for 1 kW MOSFET Inverter for Submersible Thruster Drives

/* This program controls the operation of a PWM inverter used to supply power to a variable speed induction motor drive. The actual pulse width modulation is performed by a special purpose integrated circuit manufactured by Signetics. The control program performs the overall sequencing and control functions required by the drive.

On power up, the control program enters an initialization routine [intialize()] which performs various initialization functions. The PWM IC is controlled by a set of variable frequency pulse trains which determine the inverter output frequency (FCT), volts per Hertz ratio (VCT) and maximum device switching frequency (RCT). These frequencies are supplied by a computer controlled frequency synthesizer consisting of an AMD 9513 counter/timer, 4046 phase locked loop chips, and 74LS628 VCO chips. The first part of the initialization simply sets up the 9513 counter/timer IC so that the proper frequencies can be generated. The timer in the 8156 RAM-I/O chip is set up to interrupt the 8085 microprocessor every 5 milliseconds. The PWM chip is also reset in the initialization routine and some variables are initialized.

After the initialization routine, the controller acts as a simple state machine. The controller is always in one of the following states:

1. OFF
   The on/off switch is off and the inverter is off. The speed control has no effect.

2. STOPPED
   The on/off switch is on and the speed control is in the zero position. The inverter is off.

3. ACCELERATING
   The on/off switch is on and the speed control is commanding an inverter frequency higher than the present inverter frequency. The inverter frequency is incremented by 0.5 Hz every time this state is entered. Since the state machine sequencing is controlled by the 5 millisecond clock interrupt this corresponds to an acceleration rate of 100 Hz/sec.

4. CONSTANT SPEED
   The on/off switch is on and the speed control is commanding an inverter frequency equal to the present inverter frequency.

5. DECELERATING

This state is entered when

a) The on/off switch is turned off while the inverter is operating

b) The on/off switch is on and the speed control is requesting a motor speed in the opposite direction to the present direction.

c) The on/off switch is on and the speed control is commanding an inverter frequency lower than the present inverter frequency.

The inverter frequency is decreased by 0.5 Hz on every entry into this state. This corresponds to a deceleration rate of 100 Hz/sec.

6. FAULT

This state is entered after the TRAP interrupt on the 8085 has received FAULTNUM of overcurrent signals from the inverter. To prevent a widely spaced set of overcurrent signals (possibly created by noise) from eventually causing the inverter to enter the fault state, the counter keeping track of the number of overcurrent signals is decremented on every 5 millisecond timer interrupt. As a result, the count will only increase if there is a rapid succession of overcurrent signals as would occur in a real fault. The inverter is off in the fault state. The controller waits until the speed control is reset to zero and then causes a transition to the stopped state so that the inverter can be restarted if desired. */

```
/* Definition of Constant Values */

/* Intel 8156 RAM/IO chip register addresses (I/O mapped) */

#define COM_STATUS_8156         0x10

#define PORTA           0x11

#define PORTB           0x12

#define PORTC           0x13

#define TIMER_LSB               0x14

#define TIMER_MSB               0x15

/* AMD 9513 Counter/Timer chip register addresses (I/O mapped) */

#define COMMAND_REG_9513        0X41

#define DATA_REG_9513           0x40

/* AMD 9513 internal registers */
```

```
#define MASTER_MODE             0x17

#define CNTR1MODE               0x01

#define CNTR1LOAD               0x09

#define CNTR2MODE               0x02

#define CNTR2LOAD               0x0a

#define CNTR3MODE               0x03

#define CNTR3LOAD               0x0b

#define CNTR4MODE               0x04

#define CNTR4LOAD               0x0c

#define CNTR5MODE               0x05

#define CNTR5LOAD               0x0d

#define REFREG                  CNTR1LOAD

#define FREQREG1                CNTR2LOAD

#define FREQREG2                CNTR5LOAD

#define VCTREG                  CNTR4LOAD

#define RCTREG                  CNTR3LOAD

#define START_9513              0x7F  /* 9513 start command */

/* National ADC 0808 analog to digital converter addresses (I/O mapped)*/

#define ADC                     0x80

#define ADC0                    0x80            /* A/D Converter */
                                                /* Channel 0 Address */

/* Some macros to handle various inputs and outputs */


#define FREQ_SPEED_POT_SETTING  adcin(ADC0)
        /* Frequency/Speed control pot connected to A/D channel 0 */

#define FORWARD         output(PORTB,portb=0x10 | portb)
#define REVERSE         output(PORTB,portb=0xEF & portb)
        /* CW input (phase sequence) to PWM chip is from pin 4 of port B */

#define RESET_PWM_CHIP  output(PORTB,portb=0x4 | portb)
#define PWM_CHIP_RESET_OFF      output(PORTB,portb=0xFB & portb)
        /* A input (reset) to PWM chip is from pin 2 of port B */
```

```
#define SELECT_LO_FREQ_FCT      output(PORTC,0x01)
#define SELECT_HI_FREQ_FCT      output(PORTC,0x02)
/* Active frequency synthesizer for generation of FCT is selected by */
/* pins 0 and 1 on Port C */


/* Macros for switch states */

#define ON      0x1     /* ON/OFF switch must be at +5V when ON */
#define OFF     0x0

/* Macros for inverter operation states */

#define OFFSTATE        0x0
#define STOPSTATE       0x1
#define ACCSTATE        0x2
#define DECSTATE        0x3
#define CONSTSTATE      0x4
#define FAULTSTATE      0x5

#define INTERRUPT_MASK  0x1B    /* Only Int 7.5 enabled */

/* some macros for various wait times */

#define HALFSEC         50000    /* app. 0.5 seconds */
#define HUNDREDMICROSEC 10       /* app. 100 microseconds */


/* some simple macros for flags and logical switches */

#define TRUE            1

#define FALSE           0

#define FWD             1

#define REV             0

#define FAULTNUMBER     5       /* number of overcurrent interrupts
                                   required to shut down the inverter */

/* External Variables */

char clocktick; /* flag to indicate that a 5 millisecond clock interrupt
                   has occurred */
char nexstate;  /* contains number corresponding to the state the inverter
                   controller is to enter */
char faultflag; /* flag to indicate that the inverter is to enter the
                   fault state */
char ovccount; /*  contains count of the number of overcurrent interrupts
                   received from the inverter on the 8085 TRAP input */
char pr_direction; /* flag indicating present phase sequence of inverter */
char directioncommand; /* flag indicating phase sequence setpoint */
int pr_frequency;       /* contains present inverter output frequency */
int freqload;           /* contains value to be loaded into 9513 to set
```

```
                          inverter output frequency   */
int frequencycommand;   /* contains inverter frequency setpoint */
int oldvalue;           /* contains previous value of speed pot setting */
char portb;             /* contains status of Port B in 8156 I/O RAM chip */


/* Main function first initializes the system and then loops continuously */


main()
{
initialize();
while (ON)
        {

        clocktick = 0;  /* clocktick set to 1 by int75() */
        if (faultflag == TRUE)
                nexstate = FAULTSTATE;
        switch (nexstate)  /* State sequencer */
        {
        case OFFSTATE :
                off();
                break;
        case STOPSTATE :
                stopped();
                break;
        case ACCSTATE :
                accelerating();
                break;
        case DECSTATE :
                decelerating();
                break;
        case CONSTSTATE :
                constantspeed();
                break;
        case FAULTSTATE :
                fault();
                break;

        default :
                fault();
                break;
        }

        while (!clocktick)
            ;   /* Loop while waiting for next clock
                                        interrupt*/
        }
}


initialize()
{

                /* Initialize 9513 Counter/Timer */
```

```
output(COMMAND_REG_9513, 0xFF);
        /* Master reset for 9513 */

/* Master Mode Register */

setup_9513(MASTER_MODE,0xc300);


/* Set up 9513 for:                                          */
/*       - BCD division on the scaler                        */
/*       - 8 bit data bus                                    */
/*       - FOUT = F1/3                                       */
/*       - Comparators and Time of Day disabled              */


/* Counter 1 */

setup_9513(CNTR1MODE,0xB23);


/* Counter 1 set up for repetitiive count from load register with */
/* source = F1                                                    */

setup_9513(CNTR1LOAD,1784);

/* Counter 1 set to divide by 1784 to produce a 841 Hz reference */
/* frequency for the PLLs                                        */


/* Counter 2 */

setup_9513(CNTR2MODE,0x223);

/* Counter 2 set up for repetitive count from load register with  */
/* source = SRC2.  Acts as control register for the low frequency */
/* FCT oscillator.                                                */
/* Inverter output frequency = Counter 2 Load Register/2          */


/* Counter 3 */

setup_9513(CNTR3MODE,0xB23);

/* Counter 3 set up for repetitive count from load register with  */
/* source = F1.  Counter 3 is the RCT frequency source. It is     */
/* set to 280*fswitch                                             */

setup_9513(CNTR3LOAD,1);

/* RCT initialized to 1.5 MHz.  Inverter switching frequency is   */
/* about 5.5 kHz.                                                 */

/* Counter 4 */
```

```
setup_9513(CNTR4MODE,0x423);

/* Counter 4 is set up for repetitive count from the load register  */
/* with source = SRC4.  Counter 4 is the control register for the   */
/* VCT oscillator.  VCT frequency = 1682 * Counter 4 load register  */

setup_9513(VCTREG, 1600); /* Set VCT for 100% modulation at 400 HZ */

/* Counter 5 */

setup_9513(CNTR5MODE,0x523);

/* Counter 5 is set up for repetitive count from the load register   */
/* with source = SRC5.  Counter 5 is the control register for the    */
/* high frequency FCT oscillator.                                    */
/* Inverter output frequency = Counter 5 load register/2             */


            /* 8156 I/O - Timer Initialization */

output(TIMER_LSB, 0x88);
output(TIMER_MSB, 0xD3);
output(COM_STATUS_8156, 0xCE);

/* 8156 Timer set to divide by 5000 (200 Hz out)                     */
/* 8156 Port A = input, Port B = output, Port C = output             */

              /* initialize some variables */

portb = ovccount = 0;
faultflag = FALSE;

                /* initialize inverter frequency to 3 Hz */

pr_frequency = 3;
setup_9513(FREQREG1,6);
setup_9513(FREQREG2,6);
SELECT_LO_FREQ_FCT;

              /* Start 9513 */

output(COMMAND_REG_9513, START_9513);

                  /* Initialize PWM chip */

RESET_PWM_CHIP;
wait(HALFSEC);
PWM_CHIP_RESET_OFF;

                  /* Set up initial state */

nexstate = OFFSTATE;

        /* Set up interrupt mask and enable interrupts */
```

```
sim(INTERRUPT_MASK);
enable();


}


                    /*   adcin Function   */

/*  Returns 8 bit value (as int) from A/D channel whose address is */
                      /* specified by "num"  */

adcin(num)
char num;

{
output(num,0);
wait(HUNDREDMICROSEC);
return(input(ADC));
}


                    /*  wait Function   */

/* Acts as a delay function.  Delays approximately "time" * 10          */
/* microseconds when used with a 3 MHz 8085 .                           */

wait(time)
unsigned time;

{
#asm
        POP     H
        POP     D
        PUSH    D
        PUSH    H
LOOP:   XRA     A
        DCX     D
        NOP
        ORA     D
        ORA     E
        JNZ     LOOP
#endasm
}


                      /* int75  function */

/* Interrupt service routine.  Responds to timer interrupt and gets */
/* speed command from A/D converter. Then calculates the frequency */
/* and direction setpoints */
int75()

{
```

```
int newvalue, raw;
newvalue=FREQ_SPEED_POT_SETTING;
if(abs(newvalue-oldvalue) > 1)  /* Check for significant change in */
        oldvalue=newvalue;      /* value.   */

/* Frequency setpoint is input as an 8 bit signed number in offset
   binary format */

if ((raw = oldvalue - 128) > 0) /* convert to standard format and */
        directioncommand = FWD; /* determine direction setpoint */
else
        directioncommand = REV;

if ((raw = abs(raw) - 9) < 0)
        frequencycommand = 3;  /* A deadband is left around the zero point */
else
        frequencycommand = (raw*10)/3 + 3; /* max frequency = 400 Hz */

if (ovccount > 0)       /* Make sure that a widely spaced set of overcurrent*/
        ovccount--;     /* interrupts don't shut down the inverter */

clocktick = 1;
}

                        /* trap function */

/* Trap interrupt service routine counts the number of overcurrent interrupts
   received from the inverter.  If the number exceeds a limit, the inverter is
   shut off and the fault flag is set  */

trap()
{
if (++ovccount > FAULTNUMBER)
        {
        stop();
        faultflag = TRUE;
        }
}

                        /* off function */

/* Carries out the functions required for the inverter off state        */

off()
{
stop();
if(on_off_switch() == ON)
        nexstate = STOPSTATE;
/*else
        nexstate = OFFSTATE;*/
}

                    /*  stopped function  */

/* Carries out the functions required for the inverter stopped state    */
```

```
stopped()
{
stop();
if(directioncommand == FWD)
        {
        FORWARD;
        pr_direction = FWD;
        }
else
        {
        REVERSE;
        pr_direction = REV;
        }
freqload = 6;           /* Set inverter frequency to 3 Hz */
setfrequency(freqload);


/* determine next state */

if(on_off_switch() == OFF)
        nexstate = OFFSTATE;
else if(frequencycommand>3)
        nexstate = ACCSTATE;
else
/*      nexstate = STOPSTATE*/;
}

                /*  accelerating function  */

/* Carries out functions required while inverter is increasing its frequency */
accelerating()
{
if (faultflag == FALSE)
        start();

/* check if transition to another state is required            */

if(off_or_rev())
        nexstate = DECSTATE;
else if(pos_freq_error())
        nexstate = DECSTATE;
else if(zero_freq_error())
        nexstate = CONSTSTATE;

else
        {
/*      nexstate = ACCSTATE; */
        inc_frequency();
        }
}

                /*  constantspeed function  */

constantspeed()
```

```
{
if (faultflag == FALSE)
        start();

if(off_or_rev())
        nexstate = DECSTATE;
else if(pos_freq_error())
        nexstate = DECSTATE;
else if(neg_freq_error())
        nexstate = ACCSTATE;
else
/*      nexstate = CONSTSTATE;*/;
}

                /*  decelerating function   */

decelerating()
{
if (faultflag == FALSE)
        start();

if(pr_frequency < 3)
        nexstate = STOPSTATE;
else if(off_or_rev())
        {
/*      nexstate = DECSTATE;  */
        dec_frequency();
        }
else if(pos_freq_error())
        {
/*      nexstate = DECSTATE; */
        dec_frequency();
        }
else if(neg_freq_error())
        nexstate = ACCSTATE;
else
        nexstate = CONSTSTATE;
}

/* Functions implementing various tests required to determine the next */
/* state or actions to be carried out */

off_or_rev()
{
return(on_off_switch()==OFF||directioncommand!=pr_direction);
}

pos_freq_error()
{
return(frequencycommand < pr_frequency);
}
zero_freq_error()
{
return(frequencycommand == pr_frequency);
```

```
}
neg_freq_error()
{
return(pr_frequency < frequencycommand);
}


                /*  inc_frequency function  */

/* Increases inverter frequency by 0.5 Hz  */

inc_frequency()
{
freqload++;
setfrequency(freqload);
}

                  /* dec_frequency function  */

/* As for inc_frequency except frequency is decreased          */

dec_frequency()
{
freqload--;
setfrequency(freqload);
}

/*  setfrequency function  */
/* Programs 9513 for desired frequency and selects appropriate VCO for */
/* FCT generation.  */

setfrequency(value)
int value;
{
pr_frequency=value/2;
disable();              /*disable interrupts */
sync1();                /* synchronize to 9513 counter to avoid contention
                           over access to load register */
setup_9513(FREQREG1,value);
sync2();                /* synchronize again */
setup_9513(FREQREG2,value);
enable();               /* reenable interrupts */

if(value > 250)                 /* if inverter frequency > 125 Hz */
        SELECT_HI_FREQ_FCT;     /* use high frequency VCO */
else
        SELECT_LO_FREQ_FCT;
}

                    /* sync1() function  */
/* Synchronizes loading of new count for counter #2 (FREQREG1) with start
   of new count in counter #2, thus avoiding conflict over use of Load
   register */

sync1()
```

```
{
#asm
        IN      41H     ; READ 9513 STATUS REG
        ANI     4       ; MASK OFF COUNTER 2 OUTPUT STATUS BIT
        MOV     B,A     ; SAVE IT
M1:     IN      41H
        ANI     4       ; KEEP CHECKING COUNTER 2 OUTPUT FOR A
        CMP     B       ; CHANGE
        JZ      M1      ; CHANGE INDICATES START OF NEW COUNT
#endasm
}

                    /* sync2() function */
/* As for sync1() except checking counter #5 (FREQREG2) */

sync2()
{
#asm
        IN      41H
        ANI     20H
        MOV     B,A
M2:     IN      41H
        ANI     20H
        CMP     B
        JZ      M2
#endasm
}


                    /*  fault function   */

/* Wait until the frequency setpoint is reduced to 3 Hz.  Then reset
   the fault variables (faultflag & ovccount) and set up the controller
   to enter the "stopped" state.   */


fault()
{
while ( frequencycommand > 3 )
        on_overcurrent_indicator();
faultflag = FALSE;
ovccount = 0;
off_overcurrent_indicator();
nexstate = STOPSTATE;
}

/* Functions to read switch status or to output control signals */

on_off_switch()         /* Inverter on-off switch connected to pin 0 */
{                       /* of Port A */
#asm
        IN      11H
        ANI     1
        MOV     L,A
```

```
        MVI     H,0
#endasm
}

start()                 /* L input (on/off) to PWM chip is from pin 3 of */
{                       /* Port B. Green LED (on/off indicator) is  */
                        /* connected to pin 0 of Port B */
output(PORTB,portb=0x9|portb);
}

stop()
{
output(PORTB,portb= 0xF6 & portb);
}

on_overcurrent_indicator()      /* Yellow LED connected to pin 1 of Port B */
{
output(PORTB,portb=0x2|portb);
}

off_overcurrent_indicator()
{
output(PORTB,portb= 0xFD & portb);
}

setup_9513(reg,val)     /* Loads a 16 bit value into the frequency */
int reg,val;            /* control registers of the 9513 */
{
#asm
        POP     H       ; return address
        POP     D       ; val
        POP     B       ; reg
        PUSH    B
        PUSH    D
        PUSH    H
        MOV     A,C
        OUT     41H     ;9513 command register
        MOV     A,E
        OUT     40H     ;9513 data register
        MOV     A,D
        OUT     40H     ;9513 data register
#endasm
}
```