AN ABSTRACT OF THE DISSERTATION OF

Abdurrahman Ünsal for the degree of Doctor of Philosophy in Electrical and Computer Engineering presented on December 1, 2000. Title: A DSP Controlled Resonant Active Filter for Current Harmonic Mitigation in Three-Phase Power Systems

Abstract approved: Redacted for Privacy

Annette R. von Jouanne

Power quality has become an important concern to both electric utilities and end users due to the increased use of non-linear loads in modern power systems over the past decade. Nonlinear loads inject harmonics into the power system and thus may lead to poor power quality and lower power factor. Current and voltage harmonics can adversely affect the operation of sensitive devices.

A common remedial solution to reduce the effects of harmonic distortion in a power system is filtering. Passive and active filters are two common types of harmonic filters. An active filter, in general, is a controllable current source that injects current at the same magnitude and opposite phase to that of the harmonic current. For this thesis work, a *DSP*-controlled active filter to cancel lower order (5^{th},7^{th},11^{th}, and 13^{th}) harmonics in a three-phase, three-wire power system is designed. The proposed active filter employs a series LC tank tuned to a high frequency, along with a pulse-width modulated (*PWM*) converter topology. The *PWM* control of the active filter is implemented in a TMS320F240 *DSP*. The *DSP* implementation enhances the performance of the filter in real-time and enables the filter to compensate for varying loads. Additionally, the use of *DSP*-control reduces the number of components and therefore reduces the cost and improves the reliability of the overall system. The uniqueness of the filter is in its ability to control each harmonic separately. A laboratory prototype of the proposed active filter has been built and tested to verify the performance of the active filter.

A DSP Controlled Resonant Active Filter for Current Harmonic Mitigation in Three-Phase Power Systems

by

Abdurrahman Ünsal

A DISSERTATION

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Presented December 1, 2000

Commencement June 2001

Doctor of Philosophy dissertation of Abdurrahman Ünsal

presented on December 1, 2000

APPROVED:

Redacted for Privacy

Major Professor, representing Electrical and Computer Engineering

Redacted for Privacy

Head of Department of Electrical and Computer Engineering

Redacted for Privacy

Dean of Graduate School

I understand that my dissertation will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my dissertation to any reader upon request.

Redacted for Privacy

Abdurrahman Ünsal, Author

ACKNOWLEDGEMENT

I would like to express my sincere appreciation to my advising professor, Dr. Annette von Jouanne. Her guidance, support, and motivation has been invaluable during the time I have spent at Oregon State University. I would also like to thank the other members of my committee, Dr. Alan K. Wallace, Dr. Gabor Temes, Dr. Ron Mohler, and Dr. Mark Dolan for their time and input.

I would like to acknowledge the financial support from the Turkish Government and Dumplupınar University. Special thanks to Dr. Virginia Stonick for her support with the Digital Signal Processor.

I also wish to thank Marcel Merk, Evelyn Matheson, and Manfred Dittrich for their friendship and help.

Finally, I would like to thank my parents and members of my extended family for their continued support and encouragement.

TABLE OF CONTENTS

TABLE OF CONTENTS (CONTINUED)

page

TABLE OF CONTENTS (CONTINUED)

page

TABLE OF CONTENTS (CONTINUED)

page

LIST OF FIGURES

LIST OF FIGURES (CONTINUED)

LIST OF FIGURES (CONTINUED)

LIST OF TABLES

LIST OF APPENDIX FIGURES

A DSP-CONTROLLED RESONANT ACTIVE FILTER FOR CURRENT HARMONIC MITIGATION IN THREE-PHASE POWER SYSTEMS

1. INTRODUCTION

1.1 POWER QUALITY

Power quality has become an important concern to both electric utilities and end users due to the increased use of non-linear loads in modern power systems over the past decade. The increasing cost of electrical energy has forced utilities and equipment manufactures to build high-efficiency equipment. The increased emphasis on power system efficiency has resulted in the use of a multitude of power electronic equipment in power systems and end-users' applications, such as high-efficiency adjustable speed drives.

Nonlinear loads deteriorate power quality and affect the sensitivity of modern electronic equipment to power system disturbances. Nonlinear loads inject harmonics into power systems [1]-[28] and thus may lead to poor power quality and lower power factor. Current and voltage harmonics can adversely affect the operation of sensitive devices. The most sensitive devices to harmonic distortions are solid-state devices, measurement devices, transformers, and induction machines and their controllers. Industrial facilities, which rely heavily on electronic-based controls such as timers, can experience heavy losses due to multiple zero-crossings and reduced peak voltages [1]-[4] due to data loss and long production downtime.

Harmonics are voltages and currents at integer multiples of the fundamental frequency caused by nonlinear devices. In single-phase power systems, odd-harmonics such as 3^{rd}, 5^{th}, 7^{th},...(at 180 Hz, 300 Hz, 420 Hz... for a 60 Hz system) are present with the third harmonic being dominant. In three-phase power systems,

only non-triplen odd harmonics such as 5^{th}, 7^{th}, 11^{th}... are present. The severity of the effect of harmonics on equipment depends on their magnitude and the impedance of the supply source.

1.2 SOURCES OF HARMONICS

Nonlinear loads are the main source of harmonics in electric power systems. A nonlinear load is an electrical device which draws current discontinuously or whose impedance varies throughout the cycle of the input ac voltage waveform [17]. Diode rectifiers and switch-mode power converters are two well-known examples of nonlinear loads. These loads are widely used in power electronic applications, such as switch-mode power supplies in PCs, fax machines, and adjustable speeds drives. Due to their non-linearity they can inject large magnitude harmonics into the power system.

1.3 MITIGATION TECHNIQUES

Harmonic problems involve all three parties concerned with power systems: utility companies, equipment manufacturers, and electric power consumers. The IEEE standard 519-1992 establishes recommendations for harmonic control in power systems [29]. It specifies harmonic current limits at the point of common coupling as well as the quality of the voltage that the utility must furnish the user. The European harmonic standard, IEC-555, proposes absolute harmonic limits for individual equipment loads.

A common remedial solution to reduce the effects of harmonic distortion in a power system is filtering. Passive and active filters are two common types of harmonic filters [30]. Parallel passive filters provide a low impedance path for the harmonic currents through the use of passive "LC" elements while series passive filters provide high impedance to the harmonic currents and prevent them from flowing into the power system.

An active filter, in power systems, is a controllable current source that injects current at the same magnitude and opposite phase to that of harmonic current. There are three different types of active filters: parallel, series, and hybrid active filters. Parallel active filters work as controllable current sources and are designed to reduce current harmonics by injecting currents that are equal in magnitude and opposite in phase with system harmonics. Series active filters are controlled in such a way that they can present zero impedance, at the point of common coupling, to the fundamental frequency and high impedance to harmonic frequencies. Hybrid filters are designed as a combination of passive and active filters.

The most challenging part of the design of an active filter is the controller. Currently, various control methods in time- and frequency-domains are used. The frequency-domain control methods are mostly based on the Fourier Transformation. They are mainly limited by the their long response time. However, with the advent of high-performance digital signal processors (DSPs), most of the control commands and signals are easily generated in the time-domain, as well as in the frequency-domain, in real-time.

1.4 OBJECTIVES AND RESEARCH OUTLINE

Power quality problems with the focus on nonlinear load harmonics and mitigation techniques are investigated because of the increased use of nonlinear loads and sensitive electronic equipment in power systems. The objective of this research is to design a *DSP*-controlled active filter to cancel lower order (5^{th},7^{th},11^{th}, and 13^{th}) harmonics in a three-phase three-wire power system to counter their detrimental effects on the power system and sensitive equipment. The filter is designed to be connected at the point of common coupling (*PCC*) with a number of single, multiple, or varying three-phase nonlinear loads. The filter is designed to cancel harmonics from nonlinear loads under varying load conditions. The time for the filter to respond to a varying load is one cycle of 60 Hz frequency.

The filter can be used with the loads that are changing within periods larger than one cycle of 60 Hz cycle such as pump and elevator loads or any other nonlinear loads. The proposed active filter employs a series LC tank tuned to a high frequency, along with a pulse-width modulated (*PWM*) converter topology. The *PWM* control of the active filter is implemented in a TMS320F240 *DSP*. The *DSP* implementation enhances the performance of the filter in real-time and enables the filter to compensate for varying loads. Additionally, the use of *DSP*-control reduces the number of components and therefore reduces the cost and improves the reliability of the overall system. The uniqueness of the filter is in the ability to control each harmonic separately. A laboratory prototype of the proposed active filter has been built and tested to verify the performance of the filter.

The contents of this dissertation are organised as follows:

Chapter 2 through Chapter 5 provides the necessary background information of power quality problems, harmonics, and filtering techniques including active and passive filters. Chapter 2 presents a discussion of power quality. Chapter 3 focuses on nonlinear loads. The harmonics of the different types of nonlinear loads are also presented along with Matlab simulations. Chapter 4 briefly discusses passive filters to provide the necessary background for the design of the LC resonant tank of the proposed active filter. Chapter 5 concentrates on active filters. A detailed description of different topologies of active filters and common control methods are presented.

Chapter 6 presents the proposed active filter topology to cancel lower order (5^{th},7^{th},11^{th},11^{th}) harmonics in power systems. Theoretical analysis of the principle of harmonic cancellation strategy is given along with PSpice and MATLAB simulations to verify the performance of the proposed filter.

The implementation of the *PWM* control scheme of the proposed filter is given in Chapter 7. Chapter 8 summarises the experimental results. The conclusions of this work and recommendations for future work are presented in Chapter 9.

2. POWER QUALITY

2.1 INTRODUCTION

Power quality has become an important concern for utilities, their customers, and the manufacturers of electrical equipment [17]. In addition, the increasing emphasis on overall power systems efficiency has lead to a continued growth in the application of high-efficiency power electronic devices such as adjustable speed drives and switch-mode power supplies. This has resulted in increased harmonic levels in power systems which can affect the operation of various sensitive loads.

Today, the load equipment is more sensitive to power quality variations than the equipment used in the past. Many types of new load equipment use microprocessor-based control techniques and power electronic devices that are sensitive to many types of power system disturbances. Due to the economic losses resulting from power quality problems, end users are challenging utilities to improve the quality of their power. The main driving force is the increased productivity of utility customers and the demand for high-efficiency and lower cost. Manufacturers and end users want more productive and more efficient equipment. The demand for higher productivity and lower energy costs has lead to the installation of sensitive equipment which suffers the most from the power quality disturbances.

The term "power quality" is used widely to include the entire scope of interaction among electrical suppliers, users, environment, and loads. For example, utilities may define power quality as reliability. The manufacturers of load equipment may define power quality as those characteristics of a power supply that enable the equipment to function properly. Power quality commonly has been used to describe variations of voltage, current, and frequency on a power system [2].

Thus, the power quality problem can be defined as "any power problem apparent in voltage, current, or frequency deviations that results in failure or misoperation of customer equipment [17]".

The ultimate reason for the interest behind power quality is economic value. The quality of electric power has economic impacts on utilities, their customers, and manufacturers of load equipment. In addition, the quality of power has a direct economic impact on many industrial consumers. There has recently been a great emphasis on automation of process industries which leads to the use of electronically-controlled and energy-efficient modern equipment which is sensitive to deviations in the supply voltage. Meeting the customer expectations towards high-quality power and maintaining customer confidence is a strong motivator for utilities' concern about power quality. With today's movement towards deregulation of electric utilities and a competitive market, it is important for utilities to meet customer expectations. The loss of a large industrial customer to a competing power supplier can have a significant financial impact on a utility. Power quality also has a direct economic impact on the manufacturers of today's modern load equipment. Reducing the sensitivity of load equipment to power system disturbances will increase the cost of the equipment significantly.

Electric power generated by utilities has nearly perfect sinusoidal waveforms. Throughout processing including generation, transmission, distribution, and consumption, many power quality problems from small deviations to complete outages occur. These deviations deteriorate the quality of power and increase the vulnerability of power system equipment to disturbances. Causes and effects of power systems disturbances on power systems and sensitive loads are discussed next.

2.2 VOLTAGE SAGS

A voltage sag is defined as a decrease in rms voltage of between 10% and 90% of nominal voltage at the fundamental power frequency, for periods from a

half cycle up to a minute long [17], [20]. Common causes of sags are power equipment failures, downed lines, utility recloser operations, power factor correction systems, load changes, and storms. Voltage sags may disrupt the operation of sensitive loads such as adjustable speed drives (ASDs), high-density discharge lighting (HID), programmable logic controllers (PLC), and motor contactors.

2.3 VOLTAGE SURGES

Surges are sudden, microsecond to millisecond increases in the magnitude of the voltage of a power system [18]. Surges may be fast-rise, fast-fall, or damped oscillatory in form. Common causes of surges are lightning, power system faults, switching of power protective equipment, power factor correction equipment, and sudden load changes. Surges can result in insulation breakdown, data errors, and damaged electronic equipment.

2.4 POWER OUTAGE

An outage is a complete loss of power for at least three cycles, lasting up to several hours [2], [18]. Outages are caused by equipment failures involving transformers, generators, and power lines. Outages affect the operation of all power equipment.

2.5 VOLTAGE AND PHASE IMBALANCE

Percent voltage imbalance is defined as the maximum deviation from the average of the three-phase voltage or current divided by the average of the three-phase voltages or currents, multiplied by 100. [17]. Phase imbalance is the deviation from ±120 degrees of angular spacing between three-phase voltages of a

three-phase power system [18]. Unequal loading among three phases in a three-phase power system causes voltage imbalance. Voltage and phase imbalances can cause excessive heating in transformers, motors, and rectifiers. Power converters may experience high ripple in the presence of imbalances.

2.6 NOISE

Noise is defined as distortion from sinewave power due to switching and arcing of industrial equipment [18]. Noise can be grouped into two categories: normal-mode noise and common-mode noise. Normal-mode noise is the voltage differential that appears between the power line and its neutral wire. Common-mode noise is the voltage differential that appears between the ground and the neutral wire.

2.7 VOLTAGE MODULATION

Modulation is a periodic increase or decrease in the amplitude of a voltage waveform [18]. Periodic loads such as pumps and furnaces cause voltage modulation. Voltage modulation can cause malfunction of equipment or can be seen as flickers which is a variation in the illumination level provided by incandescent and fluorescent lamps.

2.8 FREQUENCY DEVIATIONS

Frequency deviations are deviations from 60 Hz which may result from sudden load changes [18]. When a load increases faster than the response of a generator, the generator slows down, hence frequency decreases. Similarly, a sudden decrease in the load results in a rise in frequency. Frequency deviations

affect the operation of timing devices such as clocks and timers, and perhaps synchronous and induction machines.

2.9 HARMONICS

Harmonics are periodic voltages and currents having a frequency that is an integral multiple of the fundamental frequency [1]-[18]. Typical values for a 60 Hz power system are the 5^{th} (300 Hz), 7^{th} (420 Hz), 11^{th} (660 Hz), and 13^{th} (780 Hz). The main causes of harmonics are nonlinear devices such as ASDs and switch-mode power converters. Harmonics have adverse effects on the operation of a large variety of equipment including power electronic devices, microprocessor based control systems, and electric machines. Harmonic sources and their effects on power systems as well as measures used to analyze harmonic distortion are discussed next.

2.9.1 HARMONIC SOURCES

Nonlinear loads in residential, commercial, and industrial levels of power systems are the main sources of harmonics. The following are some common harmonic-generating loads that are widely used in these levels of power systems:

- Adjustable speed drives
- Air conditioners and compressors
- Arc welders
- Battery chargers
- Copy machines/printers
- Switch-mode power supplies
- Elevators
- Fluorescent lights (electronic ballasts)
- Personal or mainframe computers

- Uninterruptible power supplies (UPS)
- Silicon-controlled rectifier (SCR) drives
- X-Ray equipment

2.9.2 EFFECTS OF HARMONICS

The harmonics in a power system can cause problems such as device malfunction, overheating, and communication interference [6], [8]-[10], [12]-[15], [21]. Common harmonic problems that can be observed in a power system are briefly discussed next.

2.9.3 COMMUNICATION INTERFERENCE

Magnetic coupling between power lines and communication lines can cause distortion in communication signals. Current flowing in a power circuit produces a magnetic field, which will induce a current or voltage in the nearby conductors of the communication circuit and thus distort the communication signals. Other communication interference problems are:

- Relay malfunction.
- Induced line noise.
- Interference with power line carrier systems.

2.9.4 HEATING

Heating in a power circuit is referred as I^2R losses. When harmonic currents are present, heating is given as:

$$I^2R = I^2_{60\,Hz}R + I^2_{300\,Hz}R + I^2_{420\,Hz}R + \ldots \tag{2.1}$$

It can be seen that harmonics can cause excessive heating in power systems. In addition, harmonics cause additional eddy losses in transformers and electric motors. Typical harmonic related heating problems in a power system are:

- Excessive losses and heating in motors, capacitors, and transformers.
- Blown capacitor fuses.

2.9.5 SOLID STATE DEVICE MALFUNCTIONS

Common solid-state device related harmonic problems are given below.

- Errors in measurement.
- Nuisance tripping of relays and breakers.
- Unstable operation of zero-voltage crossing firing circuits.
- Interference with motor controllers.

2.9.6 ERRORS IN POWER MEASUREMENTS

Harmonic content of the load current can result in significant errors in power measurement especially with induction power meters [12], [14]. Application of digital microprocessor technology with high frequency sampling can reduce harmonics related power measurement errors significantly.

Other problems include series or parallel resonance, capacitor failure, insulation breakdown, mechanical oscillations of machines, excessive neutral current in three-phase four-wire systems, and audible noise.

2.9.7 HARMONIC MEASURES

The following methods of measurement are used in power systems to analyze and quantify harmonic content and distortion of a waveform and their effects.

Apparent power, (volt-amperes), of a power system is given as

$$\vec{S} = \vec{V}\vec{I} \tag{2.2}$$

Real power (watts) is expressed as

$$P = \mathrm{Re}\left[\vec{S}\right] = |V||I|\cos\Phi \tag{2.3}$$

Reactive power (volt-ampere-reactive, var) is

$$Q = \mathrm{Im}\left[\vec{S}\right] = |V||I|\sin\Phi \tag{2.4}$$

Displacement power factor (DPF) is defined as

$$DPF = \cos\Phi_1 \tag{2.5}$$

where Φ_1 is the phase angle difference between the fundamental voltage and current.

True power factor (PF) includes the harmonic content of the current

$$PF = \frac{I_{s1}}{I_s}\cos\Phi_1 \tag{2.6}$$

where I_s is the total rms current.

Crest Factor is the ratio of peak value to the total rms value

$$Crest\ \ factor = \frac{I_{peak}}{I_s} \tag{2.7}$$

Total Harmonic Distortion (THD)

$$THD = \frac{\sqrt{\sum_{h=2}^{\infty} I_h^2}}{I_1} \tag{2.8}$$

where I_h is the rms value of the harmonic component.

Total Demand Distortion (TDD)

$$TDD_{\%} = \frac{100}{I_{ML}} \sqrt{\sum_{h=2}^{\infty} I_h^2} \tag{2.9}$$

where I_{ML} is the maximum 60Hz demand load current and I_h is the individual harmonic current.

Distortion factor (DF)

$$DF = \frac{100}{I_1} \sqrt{\sum_{h=2}^{\infty} \left[\frac{I_h}{h}\right]^2} \tag{2.10}$$

It can be seen from the equations (2.3)–(2.10) that a current with large harmonic content leads to low active power, high reactive power, high apparent power, and poor power factor. This also implies that utilities have to rate their equipment for higher power ratings. THD is a measure of the effective value (quality) of a current or voltage waveform.

2.9.8 MITIGATION TECHNIQUES

The adverse effects of harmonics distortion can be reduced by several methods [17]-[20]. The load itself can be controlled to reduce harmonic distortion. Loads can be forced to stay at their designed characteristics to reduce the current harmonics. The operation of an adjustable speed drive at different torque and speed levels can change the harmonic content of its current significantly. According to

[17], operating an *ASD* at speed levels lower than the rated speed can generate larger harmonic currents than at the rated-speed. Adding a line reactor in series to a *PWM* controlled power converter will significantly reduce harmonics. Delta-Wye or zig-zag transformers can prevent triplen harmonics from flowing into upper levels of power systems. A phase-shift transformer with power converters can also eliminate the 5^{th} and 7^{th} harmonics in three-phase rectifiers. The sizing of power factor correction capacitors can be changed, or capacitors can be removed or moved to a different location in systems at the expense of poor power factor, higher losses, and lower voltage, in order to reduce the harmonic distortion. In addition, K-rated transformers [22] are also widely used to handle the excessive harmonic currents and thus reduce the resulting adverse effects. Derating neutral wires [10], [21] is also commonly used in power systems in order to operate under a large amount of triplen harmonics in neutral wire without excessive heating. These methods do not really cancel harmonics, but rather reduce the negative effects of the harmonics on the power system. Harmonic filters such as passive and active filters are used to cancel harmonics in power systems. Passive filters and active filters are discussed in detail in Chapter 4 and Chapter 5 respectively.

3. NONLINEAR LOADS IN THREE-PHASE THREE-WIRE AND THREE-PHASE FOUR-WIRE SYSTEMS

3.1 INTRODUCTION

A nonlinear load is an electrical device which draws current discontinuously or whose impedance varies throughout the cycle of the input ac voltage waveform [17]. Nonlinear loads, which are mainly power electronics-based power converters, are known as the primary sources of harmonics in electric power systems. When a sinusoidal voltage with a constant frequency (60 Hz) is applied to a nonlinear load, such as switch-mode power supply, a distorted current waveform results. This distorted current contains the fundamental sinusoidal current plus a number of low order harmonic currents (3^{rd},5^{th}, 7^{th}, etc.) at harmonic frequencies of 180Hz, 300Hz, 420Hz, respectively.

Diode rectifiers and thristors rectifiers are well-known types of nonlinear loads [8]. These devices are primarily used in power electronic applications, such as switch-mode power supplies in PCs, fax machines, and *ASD*s, and inject a large amount of harmonics into power systems. Three-phase nonlinear loads such as *ASD*s primarily inject non-triplen odd harmonics into power systems. Single-phase nonlinear loads such as switch-mode power supplies inject all odd harmonics with the third harmonic being dominant. The triplen harmonics of single-phase nonlinear loads add up in the neutral conductor and cause excessive overheating problems in the neutral conductors and distribution transformers. Nonlinear loads including three-phase current source and voltage source topologies and single-phase topologies are discussed next.

3.2 CURRENT SOURCE NONLINEAR LOADS

Thyristor-type rectifiers are known as current source nonlinear loads due to the nature of their operation [31]. A sufficiently large inductance at the *dc*-side of a thyristor-type rectifier produces a constant current that can be considered as a current source. Therefore, these type of rectifiers are called current-source nonlinear loads. They are used in applications such as *dc* drives, battery chargers etc. Figure 3.1 shows a typical thyristor rectifier with a *dc*-side smoothing inductor.

The current at the input of the rectifier consists of a large amount of harmonics due to the switching operation of the thyristors. The distorted current causes distorted voltage on the line inductance and thus results in a distorted voltage at the terminal of the thyristor which can cause additional distortion on the input current of single-phase nonlinear loads connected to the same power system [30]. The harmonics in the input current of three-phase rectifier are on the order of $h = 6n \pm 1$ where $k = 1,2,3...$ Figure 3.2 shows the input voltage and current waveforms of a thyristor converter with a *dc* smoothing inductor.

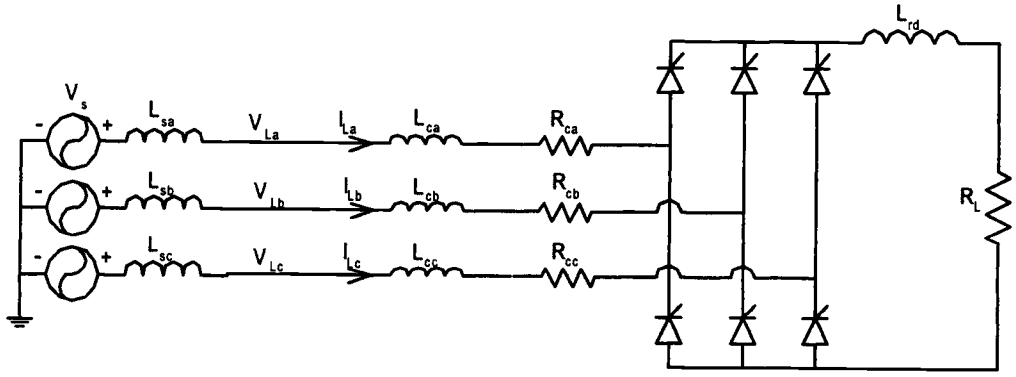


Figure 3.1 A three-phase thyristor rectifier

It can be seen that the current is no longer sinusoidal and thus contains a large number of odd harmonics. This distorted current causes a distorted voltage drop on the supply conductors, leads to voltage distortion in the supply systems (Figure 3.2), and results in poor power quality. It is recommended in [31]-[34] that

for optimum harmonics compensation with these types of loads, parallel, passive and active filters and hybrid filters should be used due to the high *dc* side impedance of the load which will force the compensation current to flow into the source side instead of the load side.

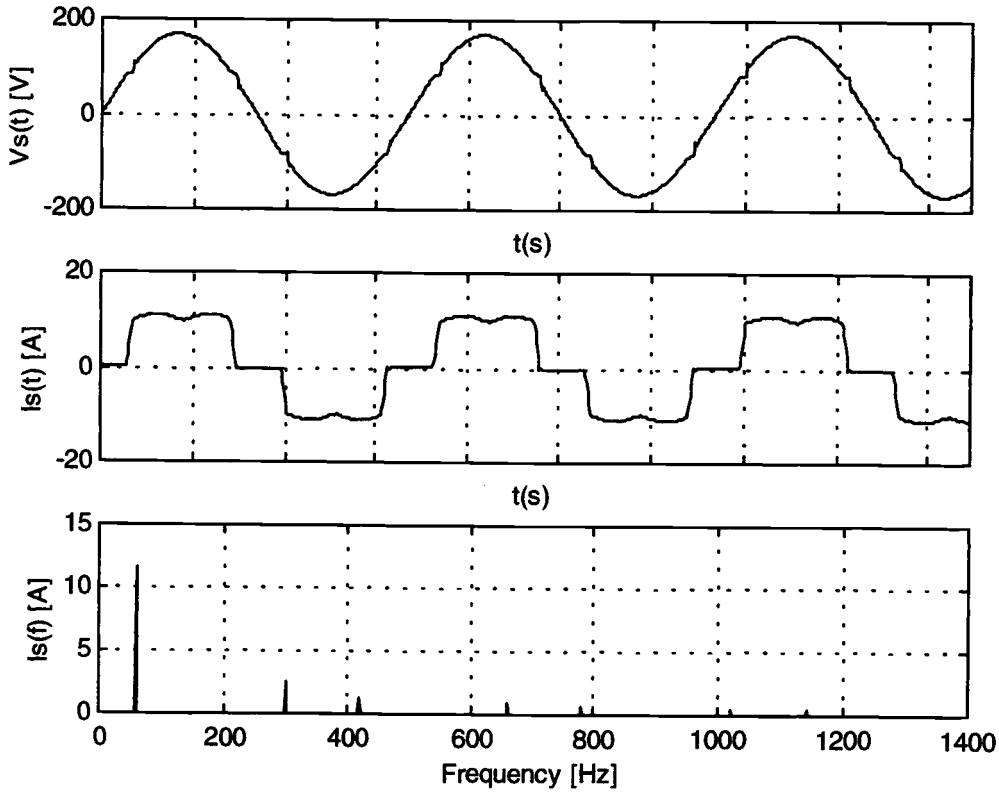


Figure 3.2 Harmonic distortion of a current-source nonlinear load: Voltage waveform and current waveform in time and frequency domains

3.3 VOLTAGE SOURCE NONLINEAR LOADS

Diode-rectifiers are another common producer of harmonics in power systems [31]. A large capacitor at the *dc*-side of the rectifier behaves like a *dc* voltage source as shown in Figure 3.3. Therefore, this type of harmonic source is called a voltage-source nonlinear load. As can be seen from Figure 3.4, the current

and voltage waveforms of diode rectifiers are much more distorted than those of the current-source harmonics source. The large distortion of the voltage is due to discontinuity in the current. It is also important to mention that the impedance of a

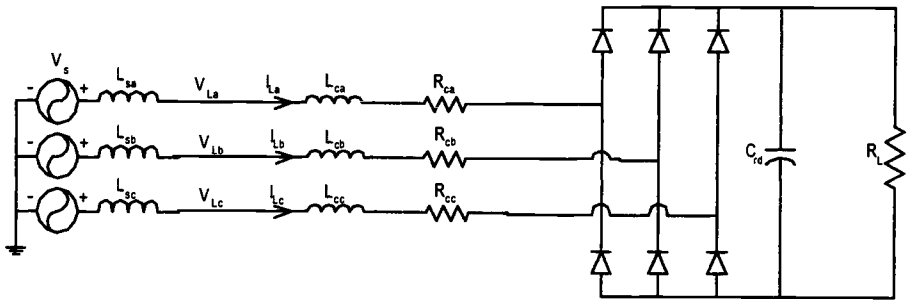


Figure 3.3 A three-phase diode rectifier with a *dc* capacitor

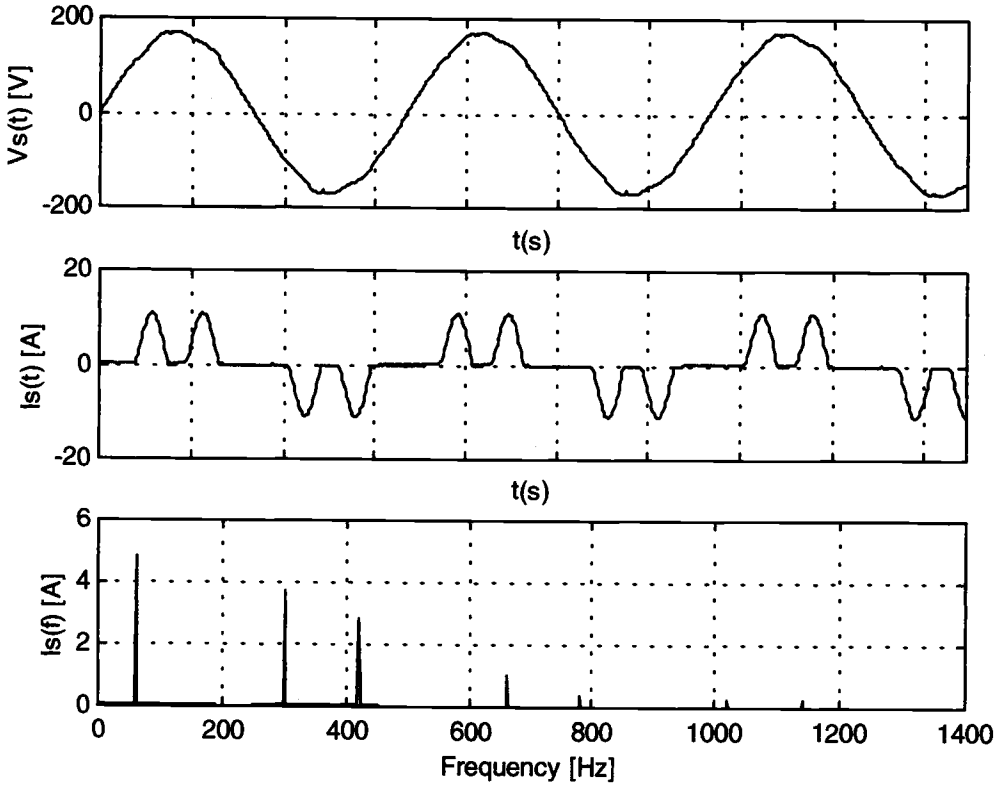


Figure 3.4 Harmonic distortion of a voltage-source nonlinear load: Voltage waveform and current waveform in time and frequency domains

voltage-source nonlinear load is smaller than that of a current-source nonlinear load due to the *dc*-side capacitor. This must be taken into account when designing harmonic compensation filters. If the impedance of the load is less than that of the source, some of the the compensation currents may flow into the load-side [31]. The recommended types of compensation filters for this type of harmonic source are series passive and active filters and hybrid filters [32].

3.4 SINGLE-PHASE NONLINEAR LOADS

Single-phase rectifiers and switch-mode power supplies are the primary sources of harmonic distortion in single-phase as well as three-phase power systems. Single-phase diode rectifiers are widely used in many residential and commercial applications such as PCs, fax machines, copy machines, and as the front-end rectifiers in switch-mode power supplies. Like in three phase diode rectifiers, single-phase rectifiers draw distorted current that contains a large amount

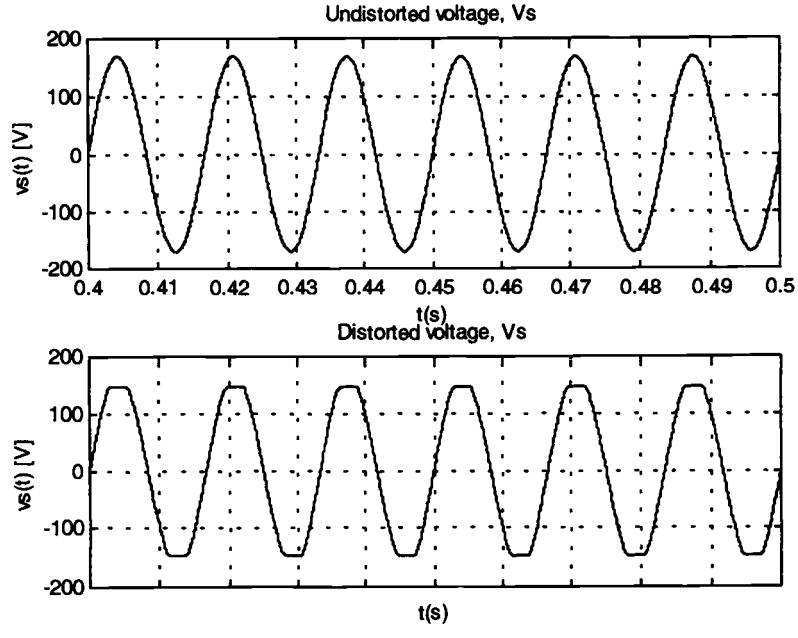


Figure 3.5 Voltage distortion when multiple single-phase loads are present: Undistorted voltage and distorted voltage waveforms

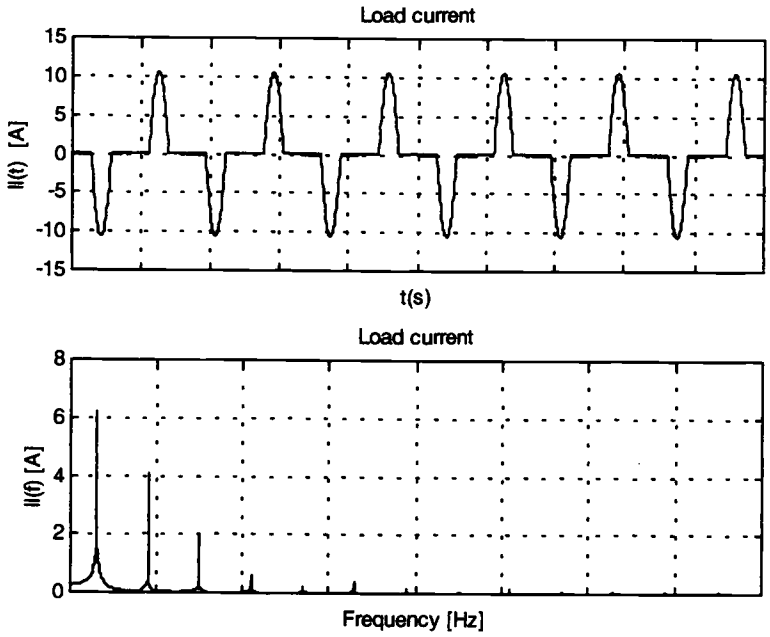


Figure 3.6 A single-phase nonlinear load current

of odd harmonics with the third harmonic dominant. Additionally, single-phase nonlinear loads draw current at the peak of the source voltage. If the source impedance is high enough, the nonlinear load current causes voltage drop at the peak of the source voltage and causes a voltage distortion called "flat-topping" (Figure 3.5). Furthermore, the triplen harmonic currents of single-phase nonlinear loads add up in the neutral conductor of the distribution system and thus cause excessive heating in the neutral conductor and the distribution transformer. This results in higher losses and therefore reduced efficiency [15]. This distortion deteriorates the power quality of the system and can cause data loss in PCs and affect many other sensitive loads. Figure 3.6 shows the current of a single-phase rectifier type nonlinear load. It is interesting to note that the load current contains a large amount of odd harmonics with the third harmonic being dominant.

3.5 PHASE-ANGLE OF HARMONIC CURRENTS

The phase angle of harmonic currents is an important factor for the design and control strategy for harmonic filters. The variation in the phase-angle of harmonic currents as the load changes can affect the performance of the filter significantly. The performance of frequency-domain control methods is mainly dependent on phase and amplitude information of the harmonic currents. If the change in the harmonic phase is large, it must be compensated when designing a controller for an active filter in order to synchronise the compensation currents with harmonic currents.

MATLAB simulations were conducted to evaluate the variations in the phase-angle of the harmonic currents of a diode-rectifier with a resistive load. Figure 3.7 shows the variation in phase-angles as a function of load change for the fundamental, 5^{th}, 7^{th}, 11^{th}, and 13^{th} harmonics.

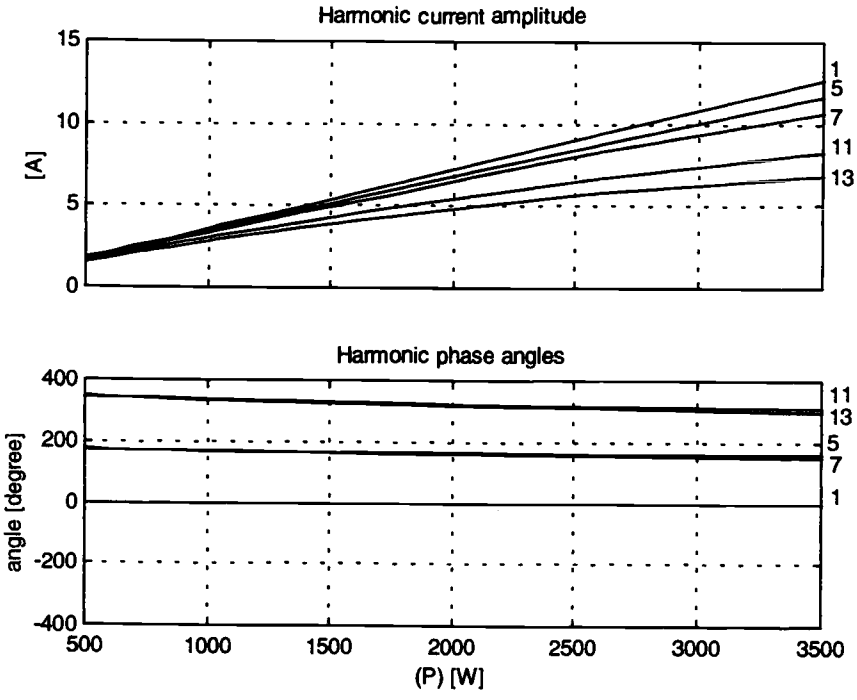


Figure 3.7 The variation of phase-angles of a diode rectifier as a function of load

It can be observed from the simulation results (Figure 3.7) that the change in phase-angle of harmonic currents as the load increases is quite small. The smallest change occurs in the fundamental current phase. The change in the 5^{th} and 7^{th} harmonic currents is comparably small. The fact that small changes occurs in harmonic angles is used in this work for the synchronisation of the compensating currents of the active filter with the load currents. The phase angles of the harmonic currents in this research are determined by Fourier Transformation on the load current. This is discussed in detail in Chapter 6.

3.6 THREE-PHASE FOUR-WIRE SYSTEMS

Large commercial and office buildings are supplied by three-phase four-wire power systems. A three-phase delta/wye distribution transformer is used to step down supply voltage to 120V or 208V to supply single-phase and three-phase loads. The 120V line-to-neutral voltage is used to supply office equipment including computers, fax machines, copiers, fluorescent lighting circuits, and many other sensitive electronic loads. Most of these types of loads are nonlinear and use switch-mode type power converters and produce mainly odd harmonics.

In three-phase four-wire distribution systems an attempt is made to load all three phases equally. If the loads are linear and the voltage and current have undistorted sinusoidal wave-shapes, the neutral current, I_n, is expected to be virtually zero. However, recently a substantial number of single-phase nonlinear loads are being installed. These loads inject a large amount of odd harmonics, with the third harmonic dominant, into power systems. In addition, uneven loading among three phases results in voltage unbalance at the terminal of other loads. The voltage unbalance cause additional triplen harmonics at the input of three-phase nonlinear loads. The triplen harmonics (3^{rd}, 9^{th} ...) add up in the neutral conductor of distribution systems, thus leading to overloading of the neutral conductors, and eventually resulting in failure (Figure 3.8).

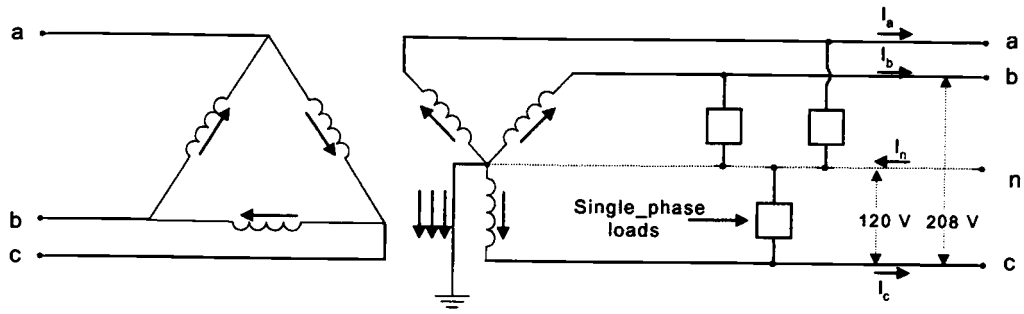


Figure 3.8 A delta/wye connected three-phase four wire distribution transformer

The harmonic currents in a power system can be expressed mathematically as follows: The currents in phase "a" of a distribution transformer are given as

$$\begin{aligned} i_a &= i_{a1} + \sum_{h=2k+1}^{\infty} i_{ah} \\ &= \sqrt{2} I_{s1} \sin(\omega_1 t - \phi_1) + \sum_{h=2k+1}^{\infty} \sqrt{2} I_{sh} \sin(\omega_h t - \phi_h) \end{aligned} \tag{3.1}$$

where k=1,2,3,... and I_{a1} is the fundamental current, I_{sh} is the h^{th} harmonic current. Assuming a balanced three-phase system with equal loading on all three phases, the currents in all three phases are equal and 120° phase shifted at the fundamental frequency (60 Hz). Therefore, the currents in phase b and c are

$$i_b = \sqrt{2} I_{s1} \sin(\omega_1 t - \phi_1 - 120^o) + \sum_{h=2k+1}^{\infty} \sqrt{2} I_{sh} \sin(\omega_h t - \phi_h - 120^o h) \tag{3.2}$$

$$i_c = \sqrt{2} I_{s1} \sin(\omega_1 t - \phi_1 + 120^o) + \sum_{h=2k+1}^{\infty} \sqrt{2} I_{sh} \sin(\omega_h t - \phi_h + 120^o h) \tag{3.3}$$

respectively. Thus, the neutral current is

$$i_n = i_a + i_b + i_c \tag{3.4}$$

$$i_n = 3 \sum_{3(2k-1)}^{\infty} \sqrt{2} I_{sh} \sin(\omega_h t - \phi_h) \tag{3.5}$$

It can be seen in (3.5) that the magnitude of the neutral current is three times the magnitude of triplen harmonic currents in phase line. This current flows into the ground through the neutral conductor of the distribution transformer. It is important to note that the triplen harmonic currents will also induce a voltage at the same frequency in the delta primary of the distribution transformer leading to a circulating current on the delta primary circuit. This circulating current can cause overloading of the distribution transformer and is not noticeable on the three-phase power system feeding the transformer. Therefore the transformer may be operating beyond its rated capacity while it is loaded only with a fraction of its rated load due to the circulating 3^{rd} harmonic current in the delta primary. Furthermore, third harmonic current can cause additional hysteresis, eddy current losses, and exacerbated by skin effect in the distribution transformer.

3.7 MITIGATION OF THIRD HARMONIC CURRENT

The excessive neutral current and its effects on the power systems due to the triplen harmonics in power systems can be reduced by several methods:

- Active and passive filtering
- Dedicated neutral conductors
- Oversized neutral conductors
- Derated distribution transformer

There are specifically designed neutral current filters to compensate for the neutral current harmonics in power systems. Single-phase passive and active filters are used to cancel triplen harmonic currents before it adds up on the neutral conductor. The use of a dedicated neutral wire for each phase and derating of the neutral conductor reduce losses in the neutral conductor [21], but the distribution transformer still experiences excess neutral current in the wye secondary circuit and circulating currents in the delta primary circuit. K-Factor rated transformers which are specifically designed with enlarged primary windings and secondary neutral

conductors [22] are rated in multiples of eddy current losses. Using a K-Factor transformer with an oversized neutral current will reduce the effect of the neutral current significantly but does not cancel the harmonics. However, the cost of the power system increases as the K-Factor and the size of the neutral conductor increases.

4. PASSIVE FILTERS

4.1 INTRODUCTION

Harmonic current distortion in power systems is discussed in detail in the previous chapters. Active and passive filtering are commonly used mitigation techniques to reduce the effects of harmonics on power systems [17]-[18], [36]. Since the filter topology used in this work uses a series LC resonant passive filter, the necessary background information to design passive filters is briefly discussed next.

4.2 TOPOLOGIES

The use of passive filters is one of the common methods to compensate for harmonic current distortion. Passive filters are made of inductors (L), capacitors (C), and resistors (R). The goal of a passive filter is to reduce harmonic distortion by one of two methods: A Passive filter provides either a low impedance path to the harmonic currents to divert them from flowing into the system or high impedance path to harmonic currents to block them from flowing into the power system [37]. Passive filters are tuned to a specific single-frequency or multiple frequencies to accomplish this goal. Single-tuned passive filters are used for low order harmonic currents which are dominant in power systems. There are two passive filter topologies: shunt passive filters and series passive filters. Figure 4.1 shows a shunt and a series passive filter.

A shunt passive filter offers a low impedance path to harmonic currents at its tuned frequency and therefore diverts the flow of harmonic current from flowing into the power system (Figure 4.1 (a)). Additionally, a shunt passive filter can compensate for the reactive power requirements of nonlinear devices [17], [36].

Shunt passive filters are designed to carry only a fraction of load current and therefore are known to cost less compared to series passive filters for the same load.

A series passive filter presents a high impedance at its tuning frequency and thus prevents harmonic currents from flowing into the power system. A series passive filter is designed to carry full load current and is therefore rated for the full load current. In addition, any problems with a series passive filter can affect the power supply to the load and thus leads to low system reliability.

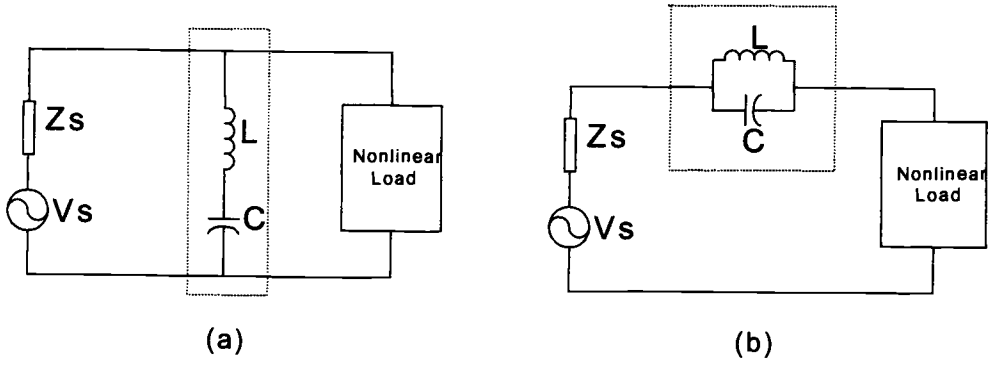


Figure 4.1 (a) A shunt passive filter (b) A series passive filter

4.3 DESIGN CONSIDERATIONS

Single-tuned, first order high-pass, second order high-pass and third order high-pass passive filters are commonly used configurations. One or more passive filters are designed for low order harmonics and then one high-pass filter is designed for the rest of the higher order harmonics [18], [36]-[41]. Figure 4.2 shows the most commonly used passive filter configurations. Quality factor, filter effectiveness, and resonant frequency are the most important parameters of the design of passive filters. The discussion here is limited to the design of a single-

tuned shunt passive filter. The impedance of a single-tuned passive filter (LCR filter) as a function of frequency is given by

$$Z = R + \left[j\omega L - \frac{1}{(j\omega C)} \right] \tag{4.1}$$

where ω is resonant frequency in radians. The resonance occurs when the imaginary part is equal to zero, at that time the impedance of the filter is limited by the value of R.

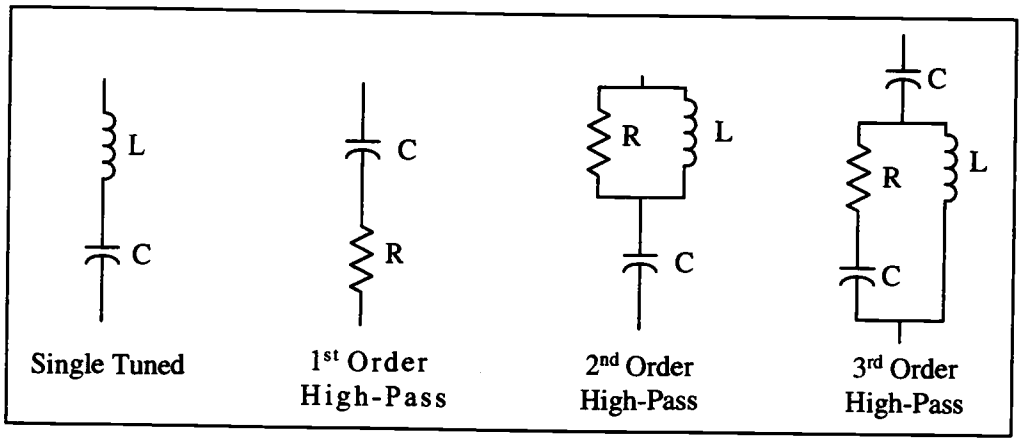


Figure 4.2 Common passive filter structures

The frequency at which the filter is tuned and results in series resonance is given as

$$f_o = \frac{1}{2\pi\sqrt{LC}} \tag{4.2}$$

where f_o = resonant frequency

C = filter capacitance

L = filter inductance

R = resistance representing the ohmic losses of the filter

The quality factor (Q) of a filter is the measure of the sharpness of tuning. It is the ratio of reactance (X_C) or inductance (X_L) to the resistance (R) at tuned frequency Therefore, Q can be expressed as in equation (4.3)

$$Q = \frac{X_L(f_o)}{R} = \frac{X_C(f_o)}{R} \tag{4.3}$$

where f_o = resonant frequency

$X_L(f_o)$ = inductive reactance at the resonant frequency

$X_C(f_o)$ = capacitive reactance at the resonant frequency.

The selection of Q influences the effectiveness of the filter in series and parallel resonance applications, inductor power losses, and cost, as well as the capacitor and inductor ratings. A filter with a high Q has lower losses, narrow bandwidth, and sharper tuning, but may experience stability problems.

The filter effectiveness, an indicator of the performance of the filter, is the ratio of filter-generated harmonic current to the load-generated harmonic current. The filter effectiveness is given mathematically as below.

$$\varepsilon\,\% = \frac{I_{hF}}{I_h} 100 \tag{4.4}$$

where I_{hF} is the filter current at the tuning frequency and I_h is the load harmonic current. The filter size is an important parameter when the filter is required to compensate for reactive power required by the nonlinear load. The size of the filter is known as the reactive power of the capacitor at the fundamental frequency [38]. Therefore, the size of the capacitor determines the cost of passive filters. Passive filters are less expensive than active filters.

To provide stable operation, a passive filter must be connected at the point in a power system where the reactance of the source is fixed and expected to remain constant. Passive filters, in general, are simple but they have the following problems:

- Passive filters may fall in series resonance with source impedance.

- Parallel resonance may occurs between passive filters and source impedance and thus may result in harmonic amplification.
- If the source impedance is not known in advance or is subject to changes with systems configuration, the filtering characteristics are not stable.

Due to the above drawbacks of passive filters, active filters have attracted great attention and are replacing passive filters. This is a result of the developments in solid-state switching devices and control technology in recent years. Active filters are discussed in the next chapter.

Although passive filters have the above mentioned drawbacks when used for harmonic cancellation purposes, a series LC resonant passive filter is used in this work as an energy storage tank and therefore, it does not experience the problems that normally passive filters may have. Figure 4.3 shows the impedance of the resonant filter designed in this work.

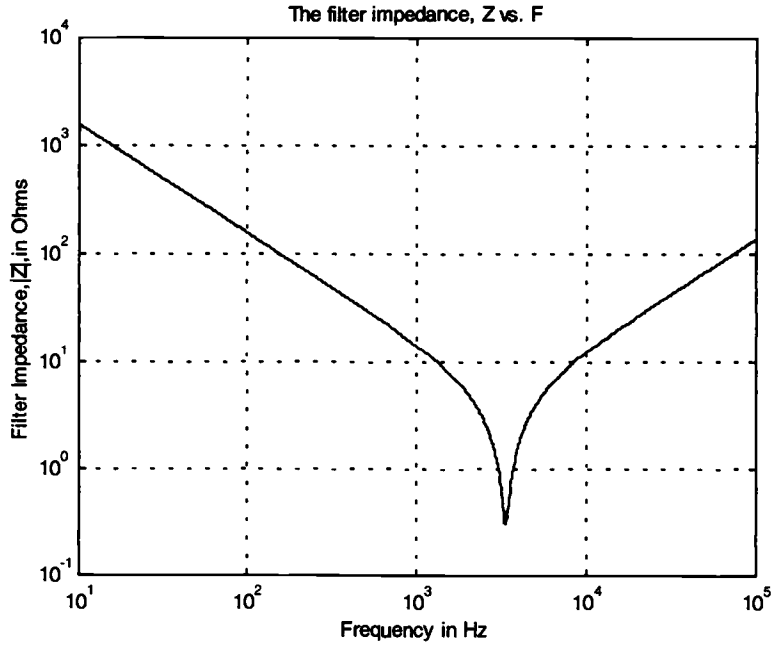


Figure 4.3 The impedance of the resonant filter as a function of frequency

5. ACTIVE FILTERS

5.1 INTRODUCTION

An active filter, in general, is a controllable current source that uses solid-state switches to inject harmonic currents with the same amplitude and opposite phase to that of the load current to maintain a sinusoidal current at the source terminals. With the developments in fast solid-state switching devices, *PWM* inverter technology, and sophisticated instantaneous active and reactive (p-q) theory by Akagi [42]-[54], series and shunt active filters have been developed to compensate for harmonic distortion reactive power requirements [45]-[54]. In 1982, the first commercial shunt active filter with a capacity of 800 KVA, which consists of a current-source inverter using gate-turn-off (GTO) thyristors [45], was put into practical use for harmonic compensation in Japan. At the present time, various active filters topologies are designed to compensate for harmonics, but in many cases they are also designed to have additional features such as compensation of reactive power. In this chapter, the state of art in active filters including various topologies, characteristics, and control methods are reviewed.

Active filters are classified based on converter type, number of phases, topologies, and control methods [33], [42], [51], [55]. A current-fed active filter uses an inductive storage element and a voltage-fed active filter uses a capacitive storage element.

5.2 CONVERTER BASED CLASSIFICATION

Current-fed and voltage-fed *PWM* inverters are two types of inverters used in active filters. A current source inverter uses an inductive element for energy storage. The inductor behaves as a controllable nonsinusoidal current source to

compensate for the harmonic current requirement of nonlinear loads. Insulated-gate bipolar transistors (IGBT's) and *GTO* thyristors are used as switching elements in these active filters [55]. Figure 5.1 (a) shows a current-fed active filter.

A voltage-fed *PWM* converter is the other type of active filter converter. A large capacitor connected to the DC bus of the converter behaves as a voltage source. This type of active filter is popular in uninterruptible power supply (*UPS*) applications because in the presence of mains, the same inverter bridge can be used as an active filter to eliminate harmonics of non-critical loads. This is a low cost solution and can be used in multilevel versions to improve performance. A voltage fed active filter is shown in Figure 5.1 (b). In this topology, the filter input must have a large enough inductor to provide compensation currents.

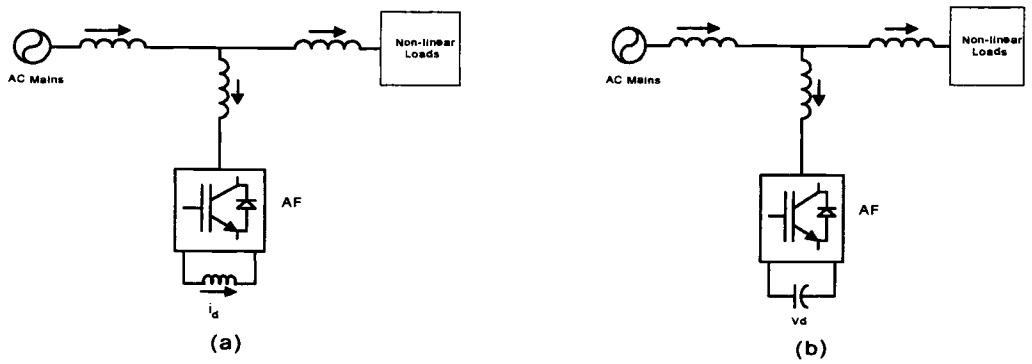


Figure 5.1 (a) A current-fed shunt active filter (b) A voltage-fed shunt active filter

5.3 TOPOLOGY BASED CLASSIFICATION

Shunt, series, and hybrid active filters are three different topologies. A shunt active filter is a *PWM* inverter to be installed in parallel with a nonlinear load to inject compensating harmonic current with the same amplitude and reverse phase of the load harmonic current. The harmonic current and shunt active filter current cancel at the point of common coupling and thus result in a sinusoidal current in the upstream level of the power system. Figure 5.1 (a) and Figure 5.1 (b) are two

examples of shunt active filters. In addition to harmonic cancellation, shunt active filters can be controlled in such a way that they can compensate for reactive power.

The compensation performance of shunt active filters is determined not only by the filter itself but also by the source and load impedances. When the load impedance is lower compared to the source impedance, the source impedance affects the performance of the shunt filter and the filter current may flow into the nonlinear load. Therefore shunt active filters are recommended [33] to be connected with loads that have larger impedance compared to the source impedance such as current source nonlinear loads (such as a *dc* drive system) to successfully compensate for the harmonic distortion.

Series active filters are designed either as controllable voltage sources (voltage-fed converter type) or as controllable current sources (current-fed converter type). A series active filter is connected in series with a nonlinear load through a specially designed transformer in order to cancel harmonic distortion. It is controlled in such a way that it can present zero impedance, at the point of common coupling, to the fundamental frequency and high impedance to harmonic frequencies to prevent harmonic currents from flowing into the system.

Since series active filters isolate the load from the source it is important to notice that they carry full load current and must withstand large power ratings. In the event of a failure of the filter's transformer, the load will lose the power supply. Therefore, series active filters reduce the overall system reliability when used as a stand-alone device. They must be protected against power fluctuations. They are rarely used as a stand-alone device, but are commonly used along with a shunt active or passive filter in hybrid filter applications. Figure 5.2 shows a voltage-source series active filter.

The source and load impedances largely influence the operational performance of series active filters. It is indicated in [33] that the compensation characteristics of a series active filter are optimized when used with a voltage source load (a diode rectifier with a large *dc* bus capacitor). Series active filters do not have reactive power compensation capability.

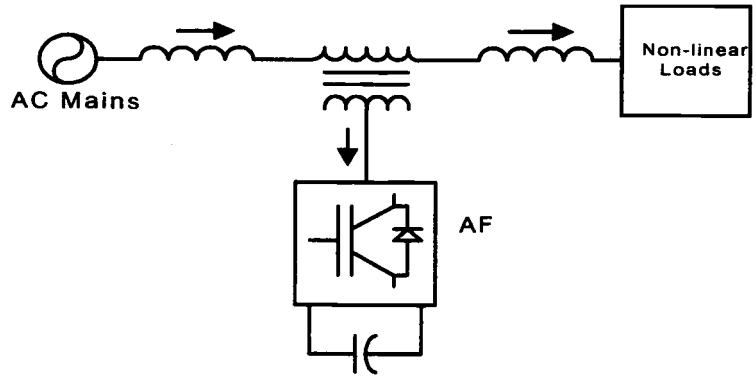


Figure 5.2 A series active filter

From an economical point of view, the cost of active filters is higher than that of passive filters. However, active filters are superior in performance to passive filters. Hybrid filters are proposed to combine the advantages of the simplicity and lower cost of passive filters with the higher performance of active filters to reduce harmonic distortion in power systems. A hybrid filter can be built in a number of different way as a combinations of:

- active series and passive shunt filters
- active series and active shunt filters
- active shunt and passive shunt filters
- active filter in series with shunt passive filters

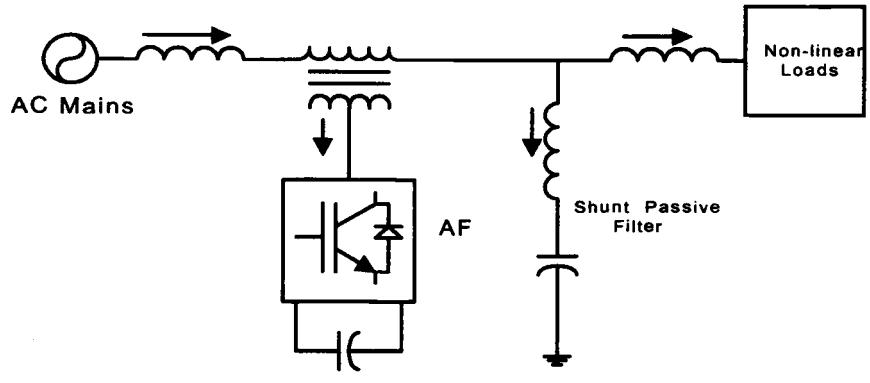


Figure 5.3 A hybrid filter as a combination of a series active filter and a shunt passive filter

Figure 5.3 shows a hybrid filter as a combination of an active series and a passive shunt filter. In this topology, the hybrid filter acts as a harmonic trap. It prevents harmonics from flowing into power system and reduces the resonance problems between passive filter components and the power system.

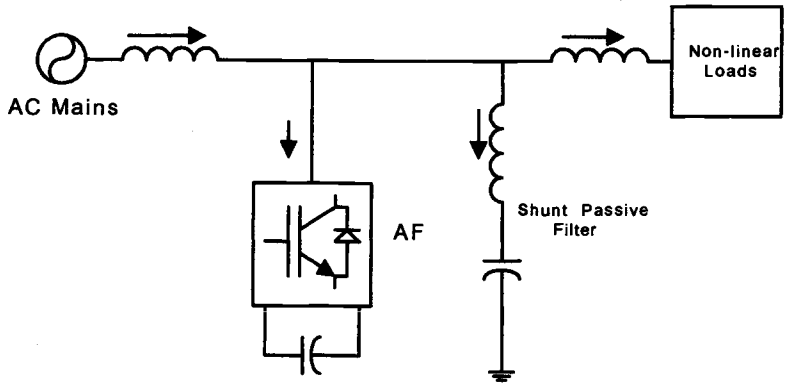


Figure 5.4 A hybrid filter as a combination of a shunt active and a shunt passive filter

In hybrid filter applications, the series active filter needs to be designed only for a fundamental fraction of the total load power and therefore the initial cost is low. However, any voltage surge or large inrush current may damage the switches of series active filters and thus result in low reliability. Figure 5.4 shows a hybrid filter as a combination of active shunt and passive shunt filters. The shunt active filter in this topology can be used to compensate for reactive power requirements. The shunt passive filter may interfere with the source impedance.

Another hybrid topology is shown in Figure 5.5. This topology is not very practical due to high initial cost. The fourth hybrid filter topology is shown in Figure 5.6. An active shunt filter is used in series with a passive shunt filter. The active filter acts as a variable impedance. Since the active filter is in series with the passive filter, the voltage and VA ratings are reduced compared to other hybrid filter topologies. The passive and active filters carry only harmonic current. Therefore, this type of hybrid filter has reduced power losses and higher reliability compared to the other hybrid filter topologies.

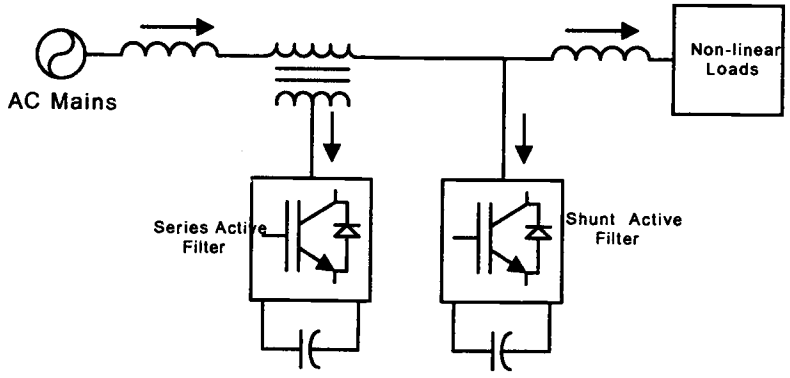


Figure 5.5 A hybrid filter as a combination of a shunt and a series active filter

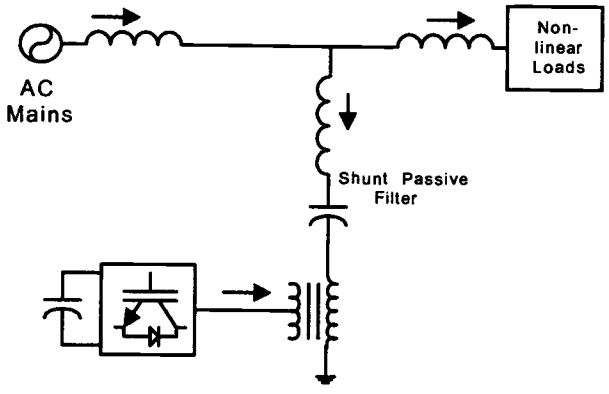


Figure 5.6 A Hybrid filter as a combination of active shunt filter in series with a passive shunt filter

5.4 SUPPLY SYSTEM BASED CLASSIFICATION

Active filters are also classified based on supply system or load system: they are classified as single-phase, three-phase three-wire, and three-phase four-wire active filters. Most nonlinear loads, (PCs, copiers, fax machines), used in residential and small commercial applications are single-phase type loads. Three-phase nonlinear loads, such as ASDs are used in large commercial and industrial

applications. As a result of the distortion of these nonlinear loads, active filters are designed and classified accordingly.

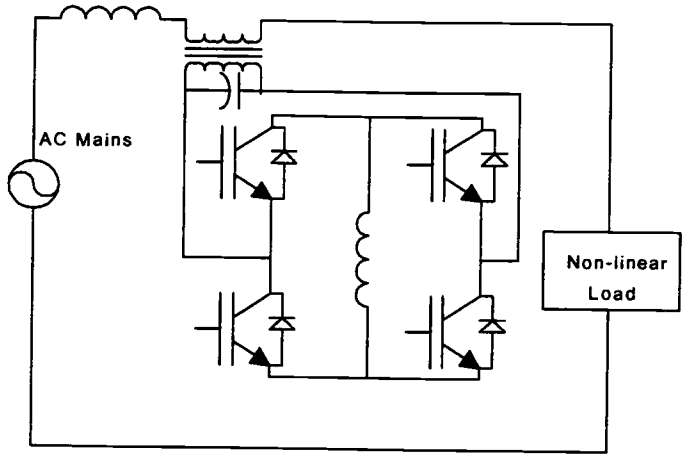


Figure 5.7 A single-phase active filter

Single-phase active filters are used in series, shunt, and hybrid configurations to reduce harmonic distortion. Like three phase active filters, single-phase active filters also use PWM-controlled current-fed or voltage-fed converters with inductive and capacitive energy storage elements respectively. Figure 5.7 shows a single-phase series active filter. In addition to harmonic compensation, single-phase active filters can be used to eliminate voltage spikes, voltage harmonics, sags, notches, and to provide reactive power compensation [55].

Three-phase three-wire active filters are used to eliminate harmonics in three-phase, three-wire nonlinear loads. Three-phase, three-wire active filters can be used in shunt, series and hybrid configurations with current-fed or voltage-fed converter topologies. An example of a three-phase three-wire system is shown in Figure 5.8.

A large number of single-phase nonlinear loads are supplied by a three-phase supply with a neutral conductor. Single-phase nonlinear loads, as mentioned in Chapter 3, draw a large amount of triplen harmonics which add in the neutral conductor. In addition, due to the uneven loading among the three phases of power

systems, the currents in all three phases are not equal and this leads to unbalanced voltage drops on the transmission lines. This results in voltage unbalance which causes additional triplen harmonics in three-phase three-wire type nonlinear loads. The triplen harmonic currents flow through the neutral wire of the power system. To mitigate this problem, three-phase, four-wire active filters are designed.

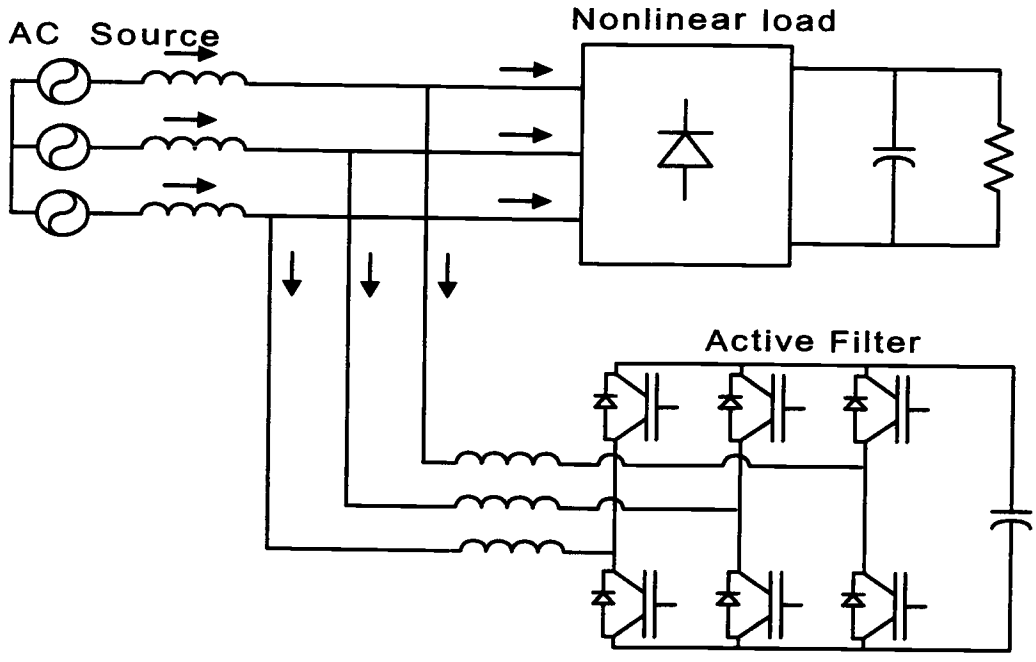


Figure 5.8 A three-phase, three-wire active filter

5.5 CONTROL STRATEGIES

The control of an active filter is the most important part of its design. The control process consists of three phases:

- Sensing of the voltage and current signals
- Extraction of the reference signals
- Generation of gating signals

The first phase of the control includes the sensing of voltage and current signals to extract harmonic content using sensors. The commonly used sensors are potential transformers (PTs), current transformers (CTs), or hall-effect voltage and current transducers (VTs and CTs). The required voltage and current signals for control algorithm implementation are supply voltages, dc bus voltage, load currents, supply currents, compensating current, and the dc link current of the active filter. These signals are used to implement the control algorithm and various performance-monitoring indices such as *THD*, power factor, reactive power, crest factor, etc. However, the harmonic content of the nonlinear load current is detected in one of three methods:

a) load current detection ($I_{AF} = I_{LH}$)

b) supply current detection ($I_{AF} = K_S I_{SH}$)

c) voltage detection ($I_{AF} = K_V V_H$)

The second phase of the control consists of the extraction of the reference signals. The reference signals are derived based on the control methods and active filter configuration. The harmonics content of the load current and the reference control signals can be derived in either time-domain or frequency-domain.

The time-domain control methods are based on the instantaneous derivation of compensating signals. Instantaneous active and reactive power theory (also known as "p-q" theory) [42]-[59], synchronous detection method [80], and sliding-mode method [61]-[63] are commonly used time-domain control techniques. The frequency-domain control methods are mainly based on the Fourier analysis of the load current and/or voltage. Fourier analysis is performed to determine the amplitude and phase of the harmonic currents of the load. Fourier analysis methods, in general, have a slow response time. However, with the advent of *DSP*s, fast Fourier algorithms can be implemented successfully in real-time.

In the third phase of control, final gating signals for the solid-state switches of active filters are generated using PWM (such as space vector modulation or modulation-signal triangular signal intersection). The compensating signals can be

developed in terms of either voltage or current signals. This is realized using discrete analog devices, microprocessors, or DSPs.

The most commonly used time-domain control techniques are instantaneous active and reactive power ("p-q") theory, sliding mode, and synchronous detection methods.

5.6 THE "P-Q" THEORY

The most well-know and widely used time-domain control method is the so-called instantaneous "p-q" theory which was originally developed by Akagi in the early 1970s [42]-[59]. This method is applied to three-phase three-wire systems (Figure 5.9) as well as three-phase, four-wire systems. The control strategy is based on transformation of three-phase current and voltage signals to two-phase orthogonal "α-β" coordinates to generate compensating signals.

The instantaneous active and reactive powers are calculated from the transformed voltage and current signals. From the instantaneous active and reactive powers, harmonic active and reactive powers are extracted through the use of low-pass and high-pass filters. From the harmonic active and reactive powers, compensation signals, in terms of current or voltage signals, are derived by the use of inverse "α-β" transformation. The main advantage of this method is the coupling of current and voltage signals when calculating harmonic current.

Mathematically, the compensating signals for a three-phase three-wire system can be derived as follows. Let the three-phase source-voltages be represented by v_a, v_b, v_c and the three-phase load currents be represented by i_{La}, i_{Lb}, i_{Lc}. Voltage and current signals can be transformed into orthogonal α-β coordinates by the following expressions:

$$\begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} = \sqrt{\frac{2}{3}} \begin{bmatrix} 1 & -1/2 & -1/2 \\ 0 & \sqrt{3}/2 & -\sqrt{3}/2 \end{bmatrix} \begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix} \tag{5.1}$$

$$\begin{bmatrix} i_{L\alpha} \\ i_{L\beta} \end{bmatrix} = \sqrt{\frac{2}{3}} \begin{bmatrix} 1 & -1/2 & -1/2 \\ 0 & \sqrt{3}/2 & -\sqrt{3}/2 \end{bmatrix} \begin{bmatrix} i_{La} \\ i_{Lb} \\ i_{Lc} \end{bmatrix} \tag{5.2}$$

The instantaneous real and reactive (imaginary) power components are defined as in (5.3)

$$\begin{bmatrix} p_L \\ q_L \end{bmatrix} = \begin{bmatrix} v_\alpha & v_\beta \\ -v_\beta & v_\alpha \end{bmatrix} \begin{bmatrix} i_{L\alpha} \\ i_{L\beta} \end{bmatrix} \tag{5.3}$$

The instantaneous active and reactive power components are then decomposed into three components by means of low-pass and high-pass filtering as shown in (5.4).

$$\begin{aligned} p_L &= p_d + p_{Lh} \\ q_L &= q_d + q_{Lh} \end{aligned} \tag{5.4}$$

In equation (5.4) p_d and q_d are dc components corresponding to the fundamental of the load current, and p_{Lh} and q_{Lh} are the ac values corresponding to the harmonic current. Components p_{Lh} and q_{Lh} are extracted by means of low-pass or high-pass filtering. The reference control signals are generated by applying the inverse α-β transformation to p_{Lh} and q_{Lh} as in equation (5.5):

$$\begin{bmatrix} i_{ha} \\ i_{hb} \\ i_{hc} \end{bmatrix} = \sqrt{\frac{2}{3}} \begin{bmatrix} 1 & 0 \\ -1/2 & \sqrt{3}/2 \\ -1/2 & -\sqrt{3}/2 \end{bmatrix} \begin{bmatrix} v_\alpha & v_\beta \\ -v_\beta & v_\alpha \end{bmatrix}^{-1} \begin{bmatrix} p_{Lh} \\ q_{Lh} \end{bmatrix} \tag{5.5}$$

The resulting reference control signals are used to generate the necessary gating signals. The gating signals can be generated by a variety of approaches such as hysteresis based current control or *PWM* current or voltage control through hardware, or software such as *DSP* implementation. This theory can be modified and applied to three-phase, four-wire balanced and unbalanced systems. The detailed discussion of this is beyond the scope of this thesis, more details can be found in [52], [58]-[59].

The other time domain control methods are synchronous detection method [60], and sliding-mode control [61]-[63]. The details of these methods are beyond the scope of this research. More details can be found in the references.

5.7 FREQUENCY-DOMAIN COMPENSATION

Control methods in frequency-domain are based on the frequency spectrum-analysis of load currents and/or voltages and are mostly based on Fourier transformations. Fast Fourier Transformation (*FFT*) are performed on the load current and/or voltage signals to extract the harmonic content and generate the appropriate gating signals for solid-state switches of the active filter [64]. Frequency-domain compensation generally has large response times. However, with the advent of high-speed microprocessors *DSPs*, *FFT*-based algorithms are being implemented in real time successfully.

Cancellation of-M-harmonics method and the predetermined method are the two major frequency-domain compensation techniques [64]. The cancellation of-M-harmonics method can be used to compensate for the varying loads. The periodicity property of harmonic currents is used to calculate the switching pattern of the injected waveform.

Predetermined harmonic injection method injects fixed harmonic currents. In this method, the harmonic currents must be known in advance. This method can successfully be used for fixed loads.

In this research, an active filter control algorithm is designed and implemented in a TMS320F240 EVM fixed-point *DSP* to cancel 5^{th}, 7^{th}, 11^{th}, and 13^{th} harmonics. The details are given in the next chapter.

6. PROPOSED ACTIVE FILTER

6.1 INTRODUCTION

AC power systems have a substantial number of harmonic generating loads i.e. adjustable speed drive systems for motor control and switch-mode power supplies used in a variety of office equipment, which draw nonsinusoidal current primarily consisting of odd harmonics as discussed in Chapter 3. The adverse effects of harmonics on power systems are discussed in Chapter 2. With the increased use of harmonic-generating nonlinear loads in power systems, the control of harmonic currents is becoming increasingly important.

A common remedial solution for reducing the effects of harmonics is through filtering. Traditionally, shunt LC passive filters were used to suppress harmonics in power systems. The filtering characteristics of passive filters, as discussed in chapter 4, are mainly dependent on the power system parameters. In addition, passive filters may interfere with the system impedances and result in harmonic amplification. Due to the limitations and undesirable characteristics of passive filters, active filters are becoming the alternative solution to reduce harmonic distortion in power systems.

In this work, a three-phase active filter is designed and implemented to compensate for lower order 5^{th}, 7^{th}, 11^{th}, and 13^{th} harmonics in a three-phase balanced power system. The proposed filter uses a *PWM* rectifier topology with a series LC tank tuned to a high frequency to cancel the lower order harmonics generated by nonlinear loads. The filter is designed to be connected in parallel with nonlinear loads at the point of common coupling (*PCC*) in order to cancel harmonics before flowing into up-stream levels of the power system. Figure 6.1 shows the block diagram of the proposed active filter system.

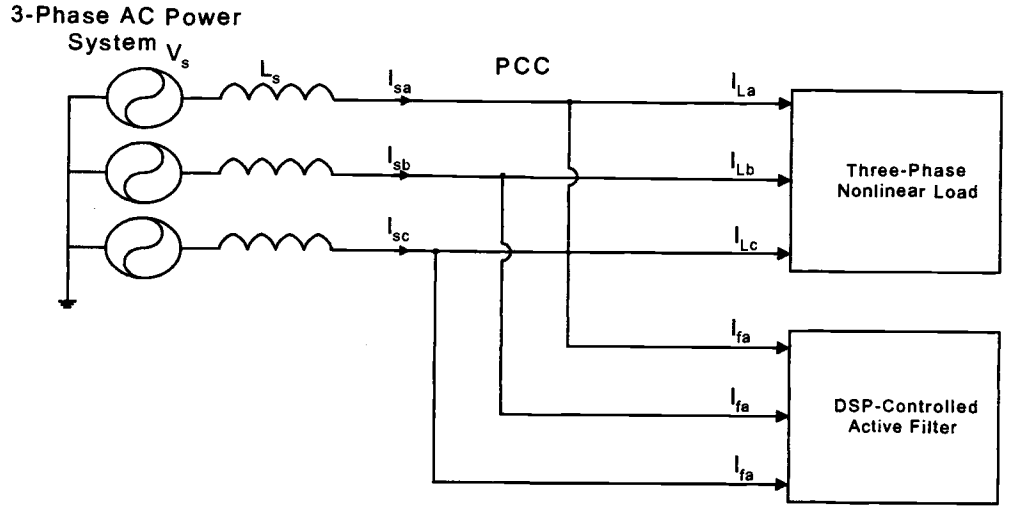


Figure 6.1 The block diagram of the proposed active filter system

6.2 DESCRIPTION OF PROPOSED FILTER

By the proper control of the proposed active filter, several lower order harmonics, i.e. 5^{th},7^{th},11^{th},13^{th}, can be effectively canceled on ac power systems. The active filter is controlled using a *PWM* rectifier topology employing six bi-directional *IGBT* switches as shown in Figure 6.2. One additional bi-directional *IGBT* switch is used as a bypass switch for the resonant tank. The filter injects harmonic currents with the same magnitude and 180 degrees out of phase with the harmonics of the nonlinear load on the ac power system. The series resonant tank circuit consisting of L_o and C_o is tuned to a high resonant frequency, e.g. $f_o = 18*f_l = 1080$ Hz (which will be explained in the next section), where $f_l = 60$Hz. The six-switch converter is *PWM* controlled to generate an output current, I_o, at the resonant frequency. The output current, I_o, is reflected onto the filter input-side, as a function of the *PWM* switching functions controlling the bi-directional switches. Thus, the desired current harmonics are generated at the input of the filter in order to cancel the nonlinear load generated harmonics at the *PCC* with the utility (Figure 6.1).

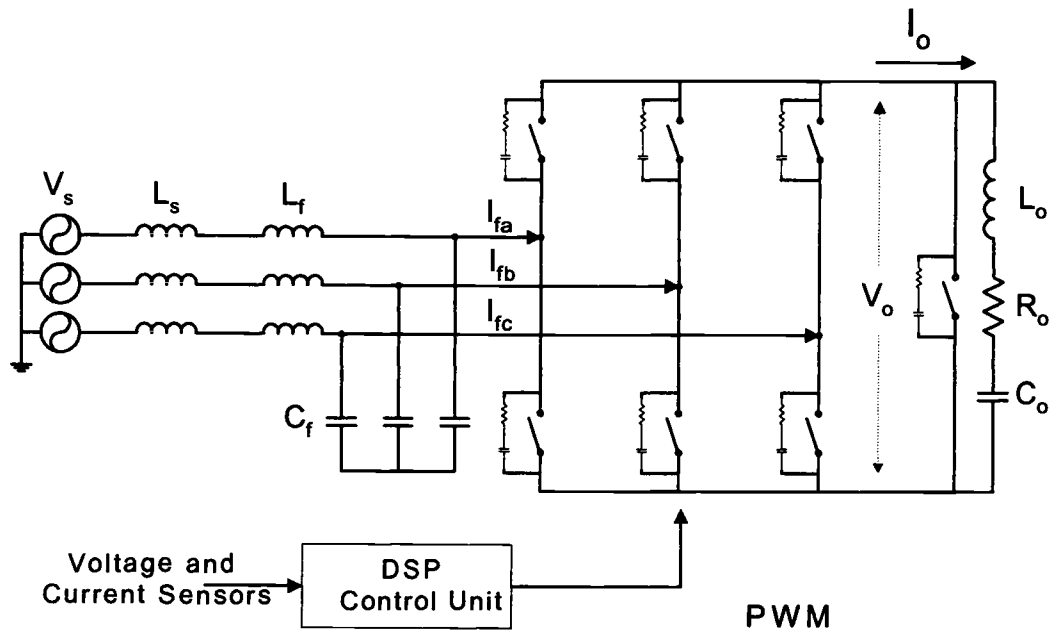


Figure 6.2 PWM series resonant active power filter with *DSP* control

The *PWM* strategy provides active cancellation of lower order harmonics by controlling the input currents I_{fa}, I_{fb}, and I_{fc} to be at the desired harmonic frequencies, i.e. 5^{th}, 7^{th}, 11^{th}, 13^{th}, with variable amplitude, and opposite in phase to the load generated harmonics (I_{La}, I_{Lb}, I_{Lc}). It should be noted that the *PWM* switching frequency is chosen to be high and the switching frequency harmonics in I_{fa}, I_{fb}, and I_{fc} are filtered by the L_f and C_f components composing a small input low-pass filter (Figure 6.2). The losses of the resonant tank circuit are represented by the resistance 'R_o' in Figure 6.2. In order to cancel the high-voltage turn-off spikes on the terminals of the switches RC snubbers are used; thus the active filter does not consume any real power other than that due to switching losses, losses in the resonant tank, and the losses in the snubber circuit. The *PWM* control of the active filter is first simulated in MATLAB to effectively cancel load-generated harmonics The active filter control algorithm is also verified through PSpice simulation as shown in Figure 6.5 using the gating signals generated in MATLAB.

The filter is controlled in frequency domain to cancel 5^{th},7^{th}, 11^{th} and 13^{th} harmonics.

6.3 ANALYSIS

The proposed three-phase series resonant active filtering scheme uses a *PWM* rectifier topology as shown in Figure 6.2. When the filter is supplied by a three phase voltage source, the output voltage, in terms of the input voltage and the modulating functions, can be written as follows

$$v_o(t) = T \cdot v_s(t) \tag{6.1}$$

where $v_s(t)$ is the line-to-line source voltage and given as follows

$$v_s(t)^T = \begin{bmatrix} V_{ab} \\ V_{bc} \\ V_{ca} \end{bmatrix} = \begin{bmatrix} V \cdot \sin(\omega t + 30^o) \\ V \cdot \sin(\omega t - 2\pi/3 + 30^o) \\ V \cdot \sin(\omega t + 2\pi/3 + 30^o) \end{bmatrix} \tag{6.2}$$

and T is the mathematical relation between the input and output voltages of the filter, a vector composed of three modulating functions and is given in equation (6.3).

$$T = [SW_1 \quad SW_2 \quad SW_3] \tag{6.3}$$

SW_1, SW_2, and SW_3 are the sinusoidal line-to-neutral switching functions, in terms of Fourier series representation, with each switching function given by

$$SW_1 = \sum_{k=1}^{\infty} A_k \sin(h_k(\omega t + \alpha)) \tag{6.4}$$

$$SW_2 = \sum_{k=1}^{\infty} A_k \sin(h_k(\omega t - 2\pi/3 + \alpha)) \tag{6.5}$$

$$SW_3 = \sum_{k=1}^{\infty} A_k \sin(h_k(\omega t + 2\pi/3 + \alpha)) \tag{6.6}$$

The resulting output voltage (in equation (6.1)) can be written as

$$v_o(t) = SW_1 \cdot V_{ab} + SW_2 \cdot V_{bc} + SW_3 \cdot V_{ca} \tag{6.7}$$

Since we are only interested in the fundamental and the 5th, 7th, 11th, and 13th harmonics, the modulating functions given in equations (6.4-6.6) reduce to the following equations (6.8-6.10)

$$SW_1 = \begin{pmatrix} A_1 \sin(h_1(\omega t + 30)) + A_5 \sin(h_5(\omega t + 30)) + A_7 \sin(h_7(\omega t + 30)) \\ + A_{11} \sin(h_{11}(\omega t + 30)) + A_{13} \sin(h_{13}(\omega t + 30)) \end{pmatrix} \tag{6.8}$$

$$SW_2 = \begin{pmatrix} A_1 \sin(h_1(\omega t + 270)) + A_5 \sin(h_5(\omega t + 270)) + A_7 \sin(h_7(\omega t + 270)) \\ + A_{11} \sin(h_{11}(\omega t + 270)) + A_{13} \sin(h_{13}(\omega t + 270)) \end{pmatrix} \tag{6.9}$$

$$SW_3 = \begin{pmatrix} A_1 \sin(h_1(\omega t + 150)) + A_5 \sin(h_5(\omega t + 150)) + A_7 \sin(h_7(\omega t + 150)) \\ + A_{11} \sin(h_{11}(\omega t + 150)) + A_{13} \sin(h_{13}(\omega t + 150)) \end{pmatrix} \tag{6.10}$$

where the "h" terms are the harmonic frequency coefficients that determine the switching frequency of the filter and the "A" terms are the harmonic coefficients that determine the amplitude of the filter compensation currents. The goal is to find the appropriate expression for the "A" and "h" terms that can be used to control the compensation currents. The filter output voltage, $v_o(t)$, by using equations (6.8-6.10) in equation (6.7) is as follows:

$$v_o(t) = \frac{1}{2} A_1 V \left(\cos((h_1 - 1)(\omega t + 30)) + \cos((h_1 - 1)(\omega t + 270)) + \cos((h_1 - 1)(\omega t + 150))\right)$$

$$+ \frac{1}{2} A_1 V \left(-\cos((h_1 + 1)(\omega t + 30)) - \cos((h_1 + 1)(\omega t + 270)) - \cos((h_1 + 1)(\omega t + 150))\right)$$

$$+\frac{1}{2}A_5V\left(-\cos((h_5-1)(\omega t+30))+\cos((h_5+1)(\omega t+270))+\cos((h_5+1)(\omega t+150))\right)$$

$$+\frac{1}{2}A_5V\left(\cos((h_5+1)(\omega t+30))+\cos((h_5+1)(\omega t+270))+\cos((h_5+1)(\omega t+150))\right)$$

$$+\frac{1}{2}A_7V\left(\cos((h_7-1)(\omega t+30))+\cos((h_7-1)(\omega t+270))+\cos((h_7-1)(\omega t+150))\right)$$

$$+\frac{1}{2}A_7V\left(-\cos((h_7+1)(\omega t+30))-\cos((h_7+1)(\omega t+270))-\cos((h_7+1)(\omega t+150))\right)$$

$$+\frac{1}{2}A_{11}V\left(-\cos((h_{11}-1)(\omega t+30))-\cos((h_{11}-1)(\omega t+270))-\cos((h_{11}-1)(\omega t+150))\right)$$

$$+\frac{1}{2}A_{11}V\left(\cos((h_{11}+1)(\omega t+30))+\cos((h_{11}+1)(\omega t+270))+\cos((h_{11}+1)(\omega t+150))\right)$$

$$+\frac{1}{2}A_{13}V\left(\cos((h_{13}-1)(\omega t+30))+\cos((h_{13}-1)(\omega t+270))+\cos((h_{13}-1)(\omega t+150))\right)$$

$$+\frac{1}{2}A_{13}V\left(-\cos((h_{13}+1)(\omega t+30))-\cos((h_{13}+1)(\omega t+270))-\cos((h_{13}+1)(\omega t+150))\right) \quad 6.11)$$

It is interesting to notice that the output voltage contains several harmonic components at different frequencies and this will be reflected at the input of the active filter as harmonic components. Since we use a single-tuned LC resonant tank at the output of the filter we would like to tune the resonant tank to the largest harmonic component of voltage/current at the output of the filter in order to generate the largest current at the input of the filter (Figure 6.2). The largest harmonic components in the output voltage (equation 6.11) are at the $(h_1\text{-}1)^{th}$ and $(h_1\text{+}1)^{th}$ frequencies. The resonant tank can be tuned to one of these two frequencies to achieve this goal. Therefore, the resonant tank is tuned to $(h_1\text{-}1)$

frequency for this research. Thus, the output voltage in equation (6.11) reduces to the following equation (equation (6.12)):

$$v_o(t) = \frac{1}{2} A_1 V \left(\cos((h_1 - 1)(\omega t + 30)) + \cos((h_1 - 1)(\omega t + 270)) + \cos((h_1 - 1)(\omega t + 150))\right)$$

$$v_o(t) = \frac{1}{2} A_1 V \left(\cos((h_1 - 1)(\omega t)\cos((h_1 - 1)30)) - \sin((h_1 - 1)(\omega t)\sin((h_1 - 1)30)\right)$$

$$+ \frac{1}{2} A_1 V \left(\cos((h_1 - 1)(\omega t)\cos((h_1 - 1)270)) - \sin((h_1 - 1)(\omega t)\sin((h_1 - 1)270)\right)$$

$$+ \frac{1}{2} A_1 V \left(\cos((h_1 - 1)(\omega t)\cos((h_1 - 1)150)) - \sin((h_1 - 1)(\omega t)\sin((h_1 - 1)150)\right) \quad (6.12)$$

The value of h_1 that results in maximum output voltage, $v_o(t)$ in equation (6.12), must be determined next. This is equivalent to the following quantities (equation (6.13)) to be equal to ± 1.

$$\cos((h_1 - 1) \cdot (30)) = \pm 1$$

$$\cos((h_1 - 1) \cdot (270)) = \pm 1 \quad (6.13)$$

$$\cos((h_1 - 1) \cdot (150)) = \pm 1$$

The value of h_1 that satisfies this condition is $(h_1 - 1) = 6 \cdot k$ where k = 1,2,3,... (h_1 = 7, 13, 19, 25 ...). A larger h_1 increases the frequency of the PWM switching harmonics and decreases the size of the passive components. The value of h_1 that is used for this research is optimized with respect to the size of LC elements and switching frequency and therefore is chosen to be 19. The output resonant circuit is tuned to (h_1 – 1) x 60 = 1080 Hz. The resulting output voltage in the resonant circuit is as follows

$$v_o(t) = \frac{-3A_1V}{2}\cos((h_1 - 1)\omega t) \tag{6.14}$$

and the current is

$$i_o(t) = \frac{v_o(t)}{R} = \frac{-3A_1V}{2R}\cos((h_1 - 1)\omega t) \tag{6.15}$$

where R is the impedance of the resonant circuit since the impedance of the inductor and capacitor in the resonant circuit cancels at the resonant frequency and $h_1 = 19$. R represents the losses in the circuit and V is the amplitude of the line-to-line input voltage.

The input current of the filter can be written as

$$\begin{bmatrix} i_{fa}(t) \\ i_{fb}(t) \\ i_{fc}(t) \end{bmatrix} = \begin{bmatrix} (SW_1 - SW_3)\cdot i_o(t) \\ (SW_2 - SW_1)\cdot i_o(t) \\ (SW_3 - SW_2)\cdot i_o(t) \end{bmatrix} = \begin{bmatrix} S_1 \cdot i_o(t) \\ S_2 \cdot i_o(t) \\ S_3 \cdot i_o(t) \end{bmatrix} \tag{6.16}$$

where S_1, S_2, and S_3 are the line-to-line sinusoidal modulating functions. As can be seen in equation (6.16), the output current, $i_o(t)$, is reflected onto the input side according to the line-to-line switching functions S_1, S_2, and S_3. The input currents for three phases are the same and 120° apart from each other. Only the expression for the current of phase "a" is given here:

$$i_{fa}(t) = (SW_1 - SW_3)\cdot i_o(t)$$

$$= (SW_1 - SW_3)\cdot \frac{3A_1V}{2R}\cos((h_1 - 1)\omega t + 180) \tag{6.17}$$

By using equations (6.17) and (6.8)-(6.10), the current at the input of the filter is given as

$$i_{fa}(t) = \frac{-3\sqrt{3}A_1^{\,2}V}{4R}\{\sin((\omega t) + \sin((2h_1 - 1)\omega t)\}$$

$$+\frac{3A_1A_5V}{4R}\{\sin((h_5 - h_1 + 1)\omega t + h_5 30 + 180) + \sin((h_5 + h_1 - 1)\omega t + h_5 30 + 180)\}$$

$$+\frac{3A_1A_5V}{4R}\{\sin((h_5 - h_1 + 1)\omega t + h_5 150 + 180) + \sin((h_5 + h_1 - 1)\omega t + h_5 150 + 180)\}$$

$$+\frac{3A_1A_7V}{4R}\{\sin((h_7 - h_1 + 1)\omega t + h_7 30 + 180) + \sin((h_7 + h_1 - 1)\omega t + h_7 30 + 180)\}$$

$$+\frac{3A_1A_7V}{4R}\{\sin((h_7 - h_1 + 1)\omega t + h_7 150 + 180) + \sin((h_7 + h_1 - 1)\omega t + h_7 150 + 180)\}$$

$$+\frac{3A_1A_{11}V}{4R}\{\sin((h_{11} - h_1 + 1)\omega t + h_{11} 30 + 180) + \sin((h_{11} + h_1 - 1)\omega t + h_{11} 30 + 180)\}$$

$$+\frac{3A_1A_{11}V}{4R}\{\sin((h_{11} - h_1 + 1)\omega t + h_{11} 150 + 180) + \sin((h_{11} + h_1 - 1)\omega t + h_{11} 150 + 180)\}$$

$$+\frac{3A_1A_{13}V}{4R}\{\sin((h_{13} - h_1 + 1)\omega t + h_{13} 30 + 180) + \sin((h_{13} + h_1 - 1)\omega t + h_{13} 30 + 180)\}$$

$$+\frac{3A_1A_{13}V}{4R}\{\sin((h_{13} - h_1 + 1)\omega t + h_{13} 150 + 180) + \sin((h_{13} + h_1 - 1)\omega t + h_{13} 150 + 180)\} \quad (6.18)$$

Equating the "h" terms to the harmonics in the input current and choosing h_1 = h leads to the following: $h_5 = (h-6)$, $h_7 = (h-8)$, $h_{11} = (h-12)$, and $h_{13} = (h-14)$. The final form of the input current is given as

$$i_{fa}(t) = \frac{3\sqrt{3}A_1^{\,2}V}{4R}[\sin\{(2h-1)\omega t\} + \sin(\omega t)] + \frac{3\sqrt{3}A_1A_5V}{4R}[\sin\{(2h-7)\omega t\} + \sin(-5\omega t)]$$
$$+\frac{3\sqrt{3}A_1A_7V}{4R}[\sin\{(2h-9)\omega t\} + \sin(-7\omega t)] + \frac{3\sqrt{3}A_1A_{11}V}{4R}[\sin\{(2h-13)\omega t\} + \sin(-11\omega t)] \quad (6.19)$$
$$+\frac{3\sqrt{3}A_1A_{13}V}{4R}[\sin\{(2h-15)\omega t\} + \sin(-13\omega t)]$$

The final form of the switching functions in terms of the "h" is given as follows

$$SW_1 = \begin{pmatrix} A_1\sin(h(\omega t + 30)) + A_5\sin((h-6)(\omega t + 30)) + A_7\sin((h-8)(\omega t + 30)) \\ + A_{11}\sin((h-12)(\omega t + 30)) + A_{13}\sin((h-14)(\omega t + 30)) \end{pmatrix} \quad (6.20)$$

$$SW_2 = \begin{pmatrix} A_1 \sin(h(\omega t + 270)) + A_5 \sin((h-6)(\omega t + 270)) + A_7 \sin((h-8)(\omega t + 270)) \\ + A_{11} \sin((h-12)(\omega t + 270)) + A_{13} \sin((h-14)(\omega t + 270)) \end{pmatrix} \quad (6.21)$$

$$SW_3 = \begin{pmatrix} A_1 \sin(h(\omega t + 150)) + A_5 \sin((h-6)(\omega t + 150)) + A_7 \sin((h-8)(\omega t + 150)) \\ + A_{11} \sin((h-12)(\omega t + 150)) + A_{13} \sin((h-14)(\omega t + 150)) \end{pmatrix} \quad (6.22)$$

In equations (6.20)-(6.22), the frequency coefficients, "*h*", determine the suitable *PWM* switching frequency and the series LC tank resonant frequency. The active filter is designed to reflect the output current, I_o, of order (*h*-1) onto the input side. Therefore, the series resonant LC tank is tuned to (*h*-1)*60 Hz. The coefficients A_1, A_5, A_7, A_{11}, and A_{13} determine the magnitudes of the reflected input harmonic currents. By changing terms, A_1, A_5, A_7, A_{11}, and A_{13} the magnitude of 5th, 7th, 11th, and 13th harmonic currents at the input of the filter can be controlled as desired. In addition, the injected harmonics by the proposed filter are made to be negative and positive sequence in order to successfully compensate for the negative sequence 5th and 11th and positive sequence 7th and 13th harmonic currents generated by nonlinear loads. Furthermore, it can be seen in equation (6.19) that I_{fa} consists of the following harmonics: 1st (60Hz), 5th, 7th, 11th, 13th, (2*h*-15), (2*h*-13), (2*h*-9), (2*h*-7), and (2*h*-1). The fundamental (60Hz), and the lower order harmonics (5th, 7th, 11th, 13th), are independently controlled by the modulating function coefficients of A_1, A_5, A_7, A_{11}, and A_{13} respectively. The independent control of the 5th, 7th, 11th, and 13th harmonics on the input side allows for effective cancellation of load generated harmonics under varying load conditions.

For $h = 19$, the higher order input harmonics of (2*h*-15), (2*h*-13), (2*h*-9), (2*h*-7), and (2*h*-1), become 23rd, 25th, 29th, 31st, and 37th, and can be easily filtered by the small input filter L_f and C_f components as shown in Figure 6.2. Also notice that for *h*=19, the LC tank resonates at 18*60 Hz = 1080Hz, resulting in small sized passive components.

By using equation (6.19) and the magnitudes of the load harmonic currents, the coefficients of the modulating signals are given as in equation (6.23).

$$A_1 = \sqrt{\frac{4RI_1}{3\sqrt{3}V}} \qquad A_5 = \frac{4RI_5}{3\sqrt{3}A_1V} \qquad A_7 = \frac{4RI_7}{3\sqrt{3}A_1V}$$

$$A_{11} = \frac{4RI_{11}}{3\sqrt{3}A_1V} \qquad A_{13} = \frac{4RI_{13}}{3\sqrt{3}A_1V} \tag{6.23}$$

where R is the resistance of the resonating circuit and I_1, I_5, I_7, I_{11}, and I_{13} are amplitudes of the load fundamental, 5th,7th,11th, and 13th harmonics respectively.

6.4 SIMULATIONS

The PWM switching functions are as given in equations (6.20)-(6.22), where the frequency coefficient variable, *h*, is 19, thus the series LC resonant frequency is (*h*-1)*60, or 1080Hz. The series resonant circuit components of L_o and C_o were selected to be 2.17mH and 10μF respectively, and display a series resonance at 1080Hz.

The simulation of the control algorithm in Matlab and PSpice is as follows. First, a nonlinear load is simulated in PSpice. The load current in the PSpice simulation is read into MATLAB. Then a 128-point FFT is performed on the load current to determine the harmonic content of the load current including phases and magnitudes. The harmonic magnitudes from the FFT are used in equations (6.23) to calculate the modulating function coefficients A_1, A_5, A_7, A_{11}, and A_{13}.

In the Matlab simulation, each modulating function, SW_1, SW_2 and SW_3 (equations (6.20)-(6.22)) is generated using the coefficients in equations (6.23) and it is then intersected with a triangle carrier wave (Figure 6.3 (a)) of a frequency of 7.68 kHz to obtain a bi-level PWM signal as shown in Figure 6.3 (b). The tri-level

line-to-line switching functions S_1, S_2 and S_3 are then obtained by successive subtractions of the line-to-neutral functions (SW_1, SW_2, SW3) as shown in equations (6.16) (Figure 6.3 (c)). Each tri-level line-to-line switching function is

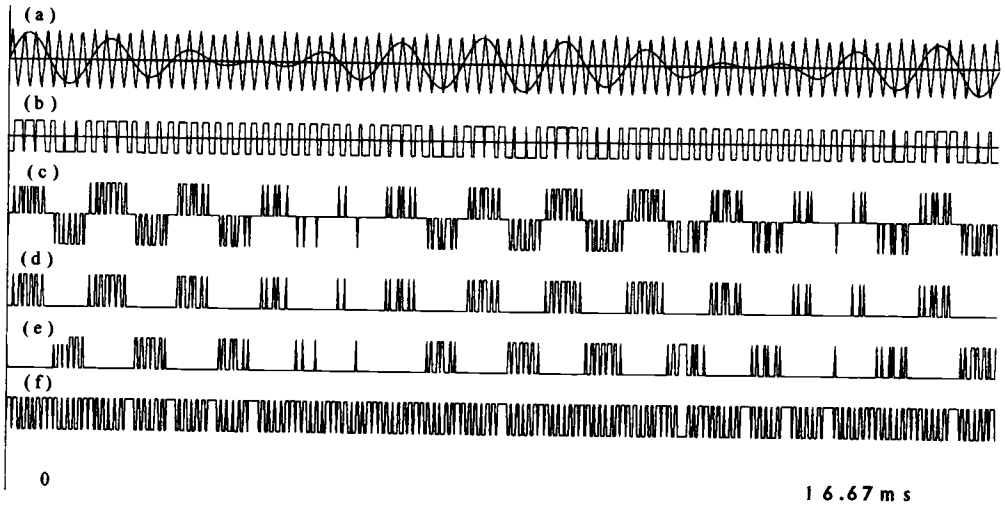


Figure 6.3 PWM control of active power filter. (a) Sine-triangle intersection (b) Bi-level PWM signal (c) Tri-level line-to-line switching function S_1 (d) Positive portion of S_1 (e) Negative portion of S_1 inverted (f) Control signal when all of the switches are in the off state

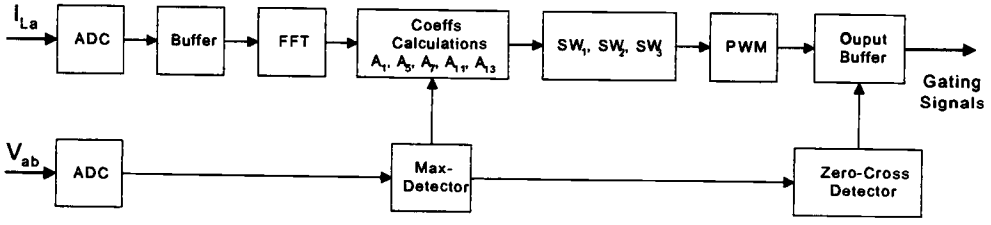


Figure 6.4 Simulation block diagram

then divided into two positive signals, (d) and (e), by inverting the negative portion. The two positive signals generated by each line-to-line switching function S_1, S_2 and S_3, correspond to the top and bottom switch of the first, second and third leg of the active filter bridge respectively. Since there must always be a path for the inductive current to flow, when all the switches are off at the same time, an

error control signal needs to be generated corresponding to the time that all of the switches are in the off state (Figure 6.3 (f)). The error signal is applied to the bypass switch. The simulation block diagram is shown in Figure 6.4.

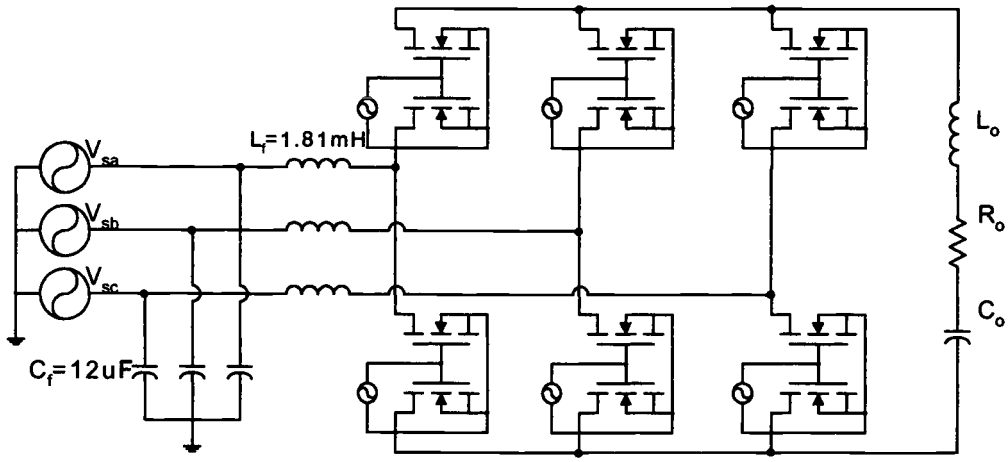


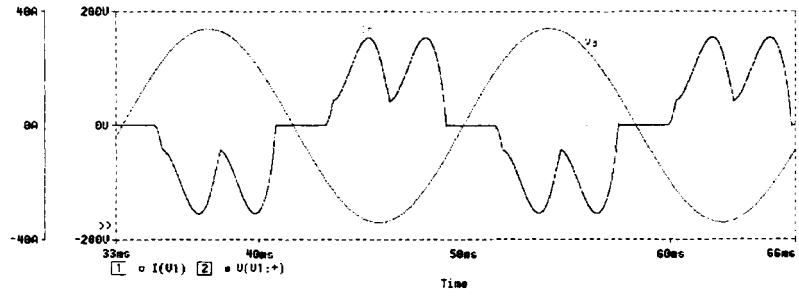
Figure 6.5 The active filter circuit diagram in Pspice

6.5 SIMULATION RESULTS

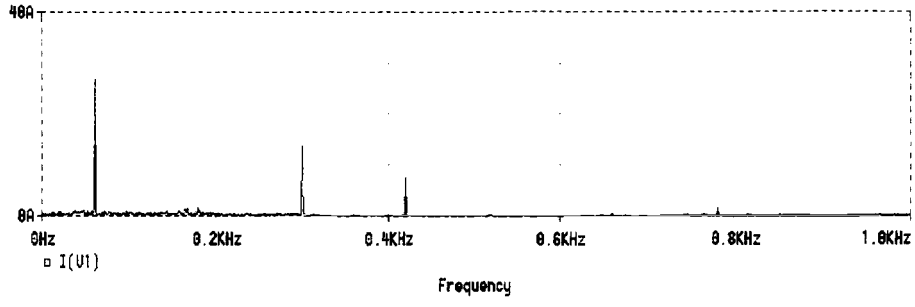
Simulations were conducted using both MATLAB and PSpice to evaluate the load harmonic cancellation performance of the proposed three-phase series resonant active power filter. Figure 6.5 shows the PSpice circuit of the active filter. Power MOSFETs were used in the PSpice simulations.

The nonlinear load system was also simulated in PSpice using a three-phase diode rectifier load. The resulting harmonic currents are given in Figure 6.6(a) and 6.6(b). Notice the nonsinusoidal nonlinear load current that is rich in harmonics as indicated in Figure 6.6(b). Figure 6.6(a) shows the input current, I_{La}, and input voltage, V_{sa}, waveforms for the three-phase diode rectifier nonlinear load. The FFT of the input current, I_{La}, is shown in Figure 6.6(b). The magnitudes of load harmonics are used in Matlab to generate the PWM switching functions.

The switching functions are then used to generate the gating signals (Figure 6.4) where they are then read into PSpice to control the active filter.



(a)



(b)

Figure 6.6 (a) The nonlinear load current and voltage (b) FFT of the nonlinear load current

The nonlinear load current and the corresponding filter and resulting conditioned ac line currents were also simulated in Matlab as shown in Figure 6.7. Matlab was also used to generate the active filter gating signals to control the PSpice circuit shown in Figure 6.5. The FFT of the resulting active filter current and conditioned ac line current are shown in Figure 6.7.

The initial nonlinear load current as shown in Figure 6.6(b) has a THD (total harmonic distortion) of 57%. After applying the active filter control algorithm in MATLAB, the THD of the ac line current is reduced to nearly 0%. After the PSpice simulation, the THD of the input ac line is reduced to 8%.

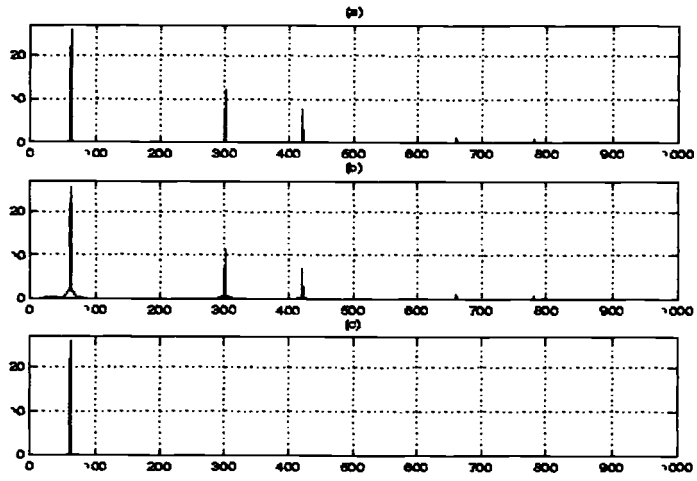
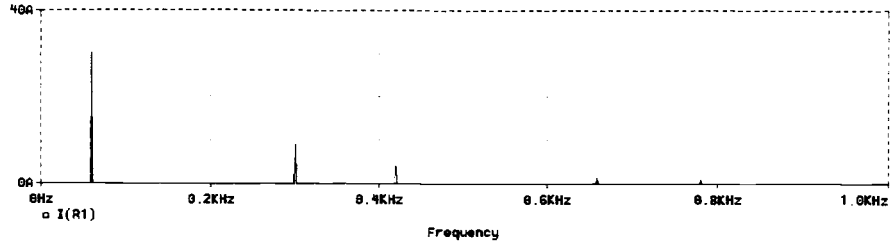
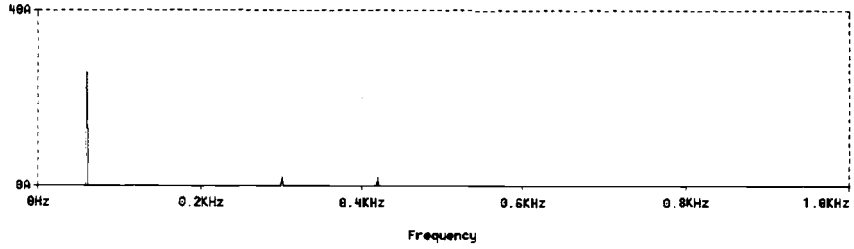


Figure 6.7 The Matlab simulation results (a) FFT of nonlinear load current (b) FFT of the generated active filter current (c) the FFT of conditioned ac line current



(a)



(b)

Figure 6.8 The PSpice simulation results (a) FFT of the generated active filter current (b) FFT of the conditioned ac line current

6.6 DISCUSSION

In this chapter, the proposed three-phase active power filter is designed and simulated in Matlab and Pspice to cancel lower order (5^{th},7^{th},11^{th},13^{th}) harmonics generated by nonlinear loads under varying load conditions (Figure 6.7 and Figure 6.8). The proposed approach employs a PWM rectifier topology with a series resonant LC tank tuned to a high frequency. The PWM control of the active power filter displays the ability to control several lower order (5^{th},7^{th},11^{th},13^{th}) harmonics to effectively cancel load generated harmonics. PSpice and Matlab simulation results verify the concept of the proposed active filter. The hardware implementation of the active filter is given in the next chapter.

7. DSP IMPLEMENTATION

7.1 INTRODUCTION

The advent of programmable *DSPs* in recent years is creating a revolution in control, monitoring and testing of power electronic applications; for example, *ASDs* and active filters. High-performance *DSPs* make it possible to implement a wide variety of control algorithms at high sampling rates and reasonably low cost. Power electronic applications such as *ac* and *dc* drives and active filters are typically complex combinations of linear, nonlinear, and high-frequency switching elements. These applications demand the use of high-speed data acquisition systems and the implementation of complex control methods. *DSPs* along with high speed data converters meet the processing requirements imposed by such systems in real-time. In addition, a *DSP*-based control system compared to a conventional discrete hardware control system, offers great performance improvements including noise filtering, signal conditioning and monitoring which is quite important for diagnostic purposes and trouble shooting [65]-[66]. Therefore, a TMS320F240 *DSP* is used in this research to control the proposed active filter.

7.2 SELECTION PROCESS OF THE DSP

Before selecting a *DSP*, the application at hand must first be thoroughly analyzed and the system requirements must be defined. The capabilities and characteristics of different processors should be evaluated in light of these requirements. The main criteria in selecting a processor for this work were *CPU* clock frequency, data converters (analog-to-digital converters (*ADCs*) and digital-

to-analog converters (*DACs*)), timers, *PWM* capabilities, word-length, and data format. The clock speed is the most critical requirement for real-time applications.

Software development of a *DSP*-based system is very critical to deliver high performance in real-time. Real-time processing requires a fast multi-tasking operating system. Using a high-level language (*HLL*) like C or MATLAB can reduce the software development effort significantly. However, a C compiler does not generate the optimum code for a *DSP*. Compiled code is usually two to four times larger than the assembly code. Due to the complexity of the algorithm and real-time requirements, assembly language is used to implement the algorithm in order to use the full capabilities of the *DSP*'s hardware performance and to prevent loss of calculation power and to minimize quantization errors. The Texas Instruments TMS320F40 *DSP*, [66]-[70], was chosen for this application based on the requirements imposed by the proposed active filter. This processor is a 16-bit fixed point DSP. The code for this application is written in assembly language. Details of this processor are given in Appendix B.

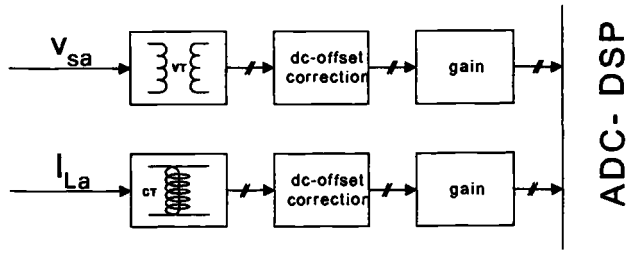


Figure 7.1 Block diagram of the *DSP* input stage

7.3 IMPLEMENTATION

The objective is to digitally control the proposed active filter, which was discussed in detail in Chapter 6, to cancel lower order (i.e. 5^{th},7^{th},11^{th},13^{th}) harmonics generated by nonlinear loads using a TMS320F240 *DSP*. To interface the *DSP* with the power circuit, a signal conditioning and protection circuit was

built. The details of the hardware interface circuits are given in Appendix C through Appendix G. Figure 7.1 shows the block diagram of the interface circuit on the input-side of the *DSP*.

Since the system is balanced, only one current signal and one voltage signal are measured. The source voltage V_{sab} and load current I_{La} are measured using a voltage transducer (*VT*) and a current transducer (CT) respectively. The *DSP* requires an input signal level of 0 - 5V. The measured voltage and current signals are offset corrected and amplified to meet the specifications of the *DSP* (Figure 7.1).

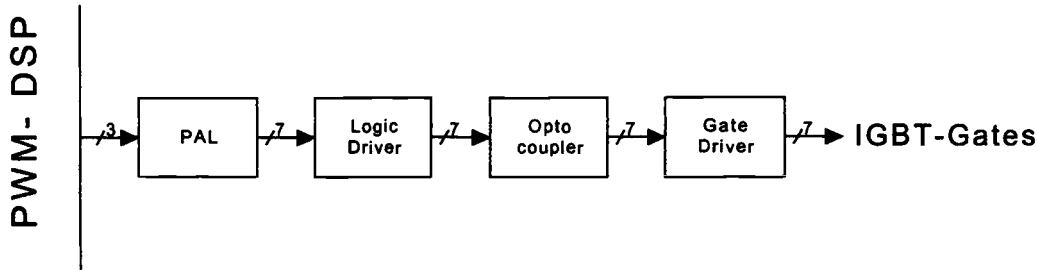


Figure 7.2 Block diagram of the *DSP* output stage

The three full-compare units of the *DSP* (Appendix B) are used for sine-triangle comparison and *PWM* generation. The three modulating functions (equations (6.20)-(6.22)) are internally generated and then intersected with a triangular carrier signal with a frequency of 7680 Hz. Three *PWM* signals corresponding to the three modulation signals are generated by the *DSP*. The active filter has a total of seven *IGBT* switches (one of them is used as a bypass switch for resonant tank, Figure 7.3). The bypass switch is required to compensate for the effects of dead time (the time when the resonant circuit is in an open circuit condition). Since the resonant circuit has a large inductor, the rectifier curcuit has to provide a current path for the resonant circuit current to flow continuously in order to protect the rectifier switches against high voltage spikes. The control signal for the bypass switch is generated when the resonant current path is incomplete.

The three *PWM* signals of the *DSP* are passed through a programmable array logic (*PAL*) to generate seven *PWM* signals to apply to the gate terminals of the active filter switches (Appendix I). The block diagram of the output stage of the *DSP* is given in Figure 7.2.

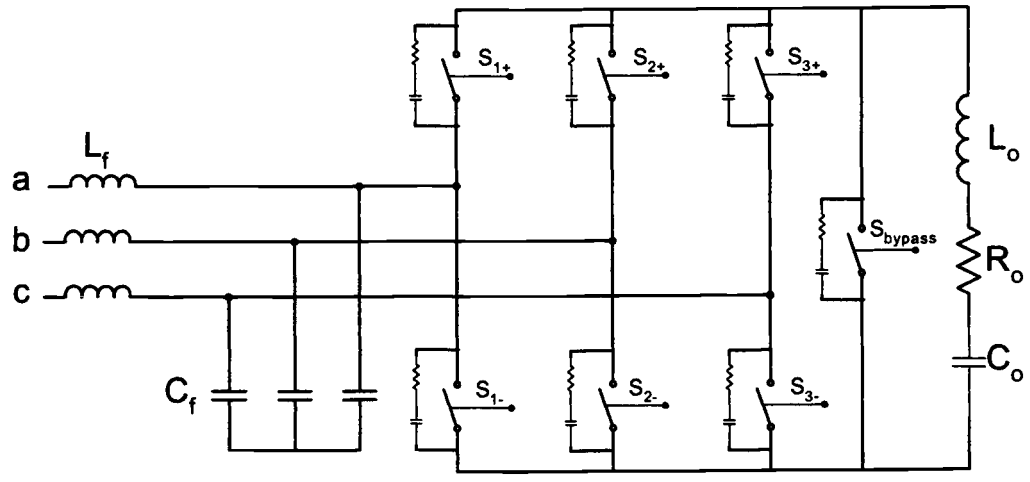


Figure 7.3 The active filter with gating signal descriptions

The generation of the seven *PWM* signals from three *PWM* signals by using a *PAL* is based on the logical operation on the three *PWM* signals. Figure 7.3 shows the descriptions of the gating signals of the active filter. The logical table for the *PAL* to generate six signals from three signals is given in Table 7.1 The code for

Table 7.1 The logical truth table to generate six *PWM* signals from three signals

Input Signals			Output Signals					
SW1	SW2	SW3	S1+	S1-	S2+	S2-	S3+	S3-
0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	1	0	0
0	1	0	0	0	1	0	0	1
0	1	1	1	0	0	0	0	1
1	0	0	0	1	0	0	1	0
1	0	1	0	0	0	1	1	0
1	1	0	0	1	1	0	0	0
1	1	1	0	0	0	0	0	0

the PAL to generate six PWM signals and the error signal for the bypass switch is given in Appendix I. The truth table for the error signal is given in Table 7.2.

The control algorithm for this research is implemented in the frequency-domain. It uses Fourier analysis of the load current to determine the harmonic content. The control algorithm is designed to cancel lower order harmonics (i.e. 5^{th},7^{th},11^{th},13^{th} at 300 Hz, 420 Hz, 660 Hz, and 780 Hz respectively). In order to

Table 7.2 The logical truth table for error signal

Switch Signals						Error Signal
S1+	S2+	S3+	S1-	S2-	S3-	Bypass
0	0	0	0	0	0	1
0	0	0	0	0	1	1
0	0	0	0	1	0	1
0	0	0	0	1	1	1
0	0	0	1	0	0	1
0	0	0	1	0	1	1
0	0	0	1	1	0	1
0	0	0	1	1	1	1
0	0	1	0	0	0	1
0	0	1	0	0	1	0
0	0	1	0	1	0	0
0	0	1	0	1	1	0
0	0	1	1	0	0	0
0	0	1	1	0	1	0
0	0	1	1	1	0	0
0	0	1	1	1	1	0
0	1	0	0	0	0	1
0	1	0	0	0	1	0
0	1	0	0	1	0	0
0	1	0	0	1	1	0
0	1	0	1	0	0	0
0	1	0	1	0	1	0
0	1	0	1	1	0	0
0	1	0	1	1	1	0
0	1	1	0	0	0	1
0	1	1	0	0	1	0
0	1	1	0	1	0	0
0	1	1	0	1	1	0
0	1	1	1	0	0	0
0	1	1	1	0	1	0
0	1	1	1	1	0	0
0	1	1	1	1	1	0

Switch Signals						Error Signal
S1+	S2+	S3+	S1-	S2-	S3-	Bypass
1	0	0	0	0	0	1
1	0	0	0	0	1	0
1	0	0	0	1	0	0
1	0	0	0	1	1	0
1	0	0	1	0	0	0
1	0	0	1	0	1	0
1	0	0	1	1	0	0
1	0	0	1	1	1	0
1	0	1	0	0	0	1
1	0	1	0	0	1	0
1	0	1	0	1	0	0
1	0	1	0	1	1	0
1	0	1	1	0	0	0
1	0	1	1	0	1	0
1	0	1	1	1	0	0
1	0	1	1	1	1	0
1	1	0	0	0	0	1
1	1	0	0	0	1	0
1	1	0	0	1	0	0
1	1	0	0	1	1	0
1	1	0	1	0	0	0
1	1	0	1	0	1	0
1	1	0	1	1	0	0
1	1	0	1	1	1	0
1	1	1	0	0	0	1
1	1	1	0	0	1	0
1	1	1	0	1	0	0
1	1	1	0	1	1	0
1	1	1	1	0	0	0
1	1	1	1	0	1	0
1	1	1	1	1	0	0
1	1	1	1	1	1	0

algorithm are one line-to-line voltage signal and one phase current signal ($v_{sa}[n]$ and $i_{La}[n]$). The filter is designed for a balanced system and therefore only one

voltage signal ($V_{ab}(t)$) and one phase current ($I_{La}(t)$) are used. The output variables of the *DSP* are three-*PWM* signals.

The harmonic content of the load current is extracted through the fast Fourier transformation of the load current providing that the three-phase load is balanced and supplied by a balanced three-phase voltage source.

The TMS320F240 module's on-board Analog-to-Digital converters (*ADC*s) sample the load current and source voltage at a sampling rate of 7680 Hz. The sampled current, $I_L[n]$, is buffered in a 256-word buffer. A 128-point *FFT* is performed on the input current to determine the harmonic content of the load current including magnitude and phase of the 5^{th}, 7^{th}, 11^{th}, and 13^{th} harmonics. Modulating signal coefficients, (A_1, A_5, A_7, A_{11}, A_{13}) and the resulting *PWM* signals are calculated using the result of the *FFT*.

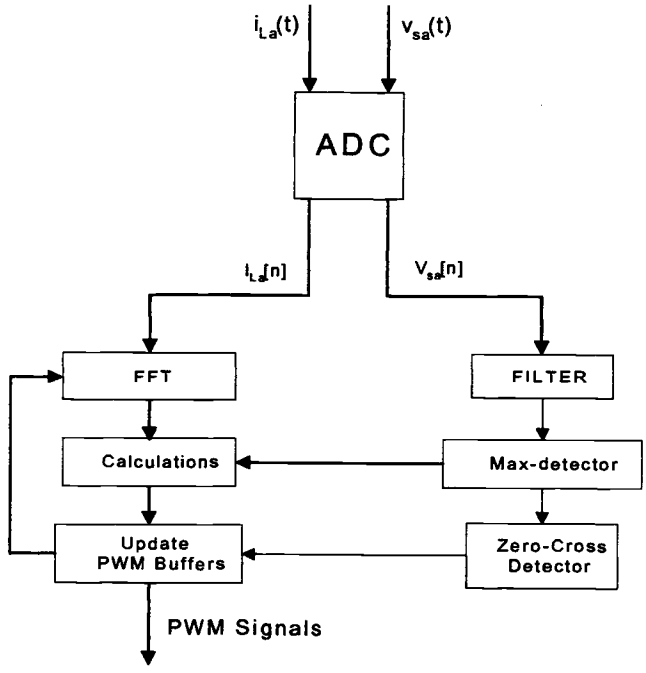


Figure 7.4 The block diagram of the implementation of control algorithm

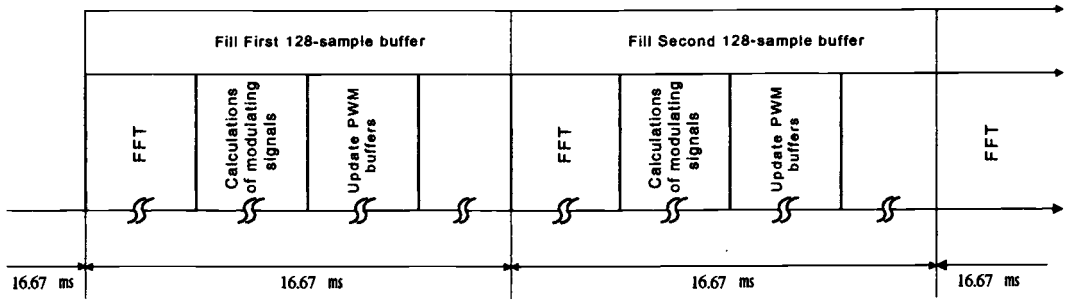


Figure 7.5 The timing diagram of the control algorithm

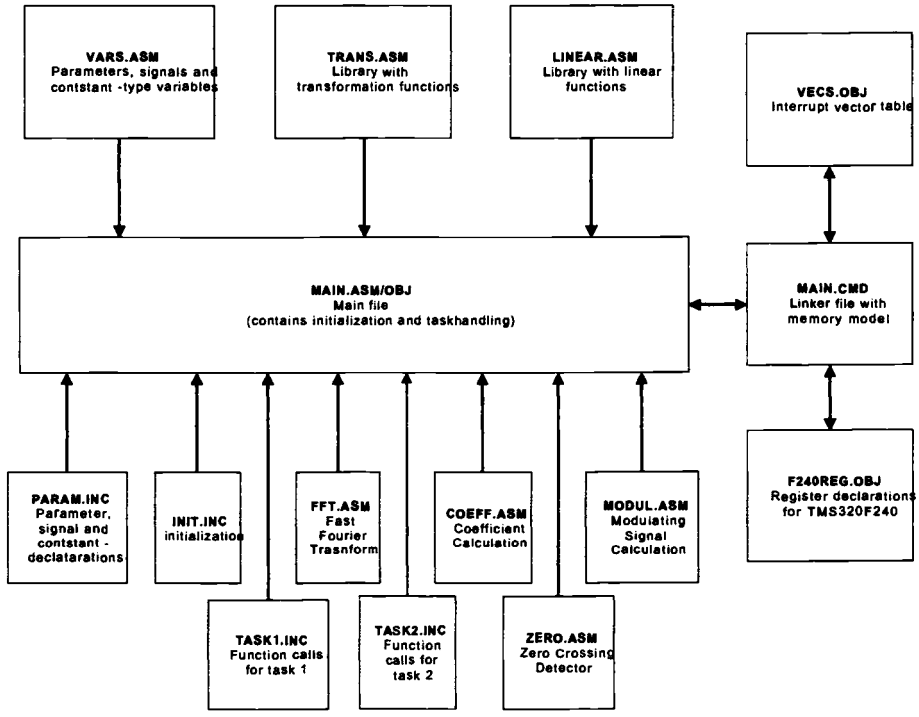


Figure 7.6 The functional organization of the software

7.4 SOFTWARE ORGANISATION

The software concept is based on functional blocks which are organised in different libraries which were originally developed in [79]. There is a central main program, which contains the task-handling process and system initialisation. All of the other programs are called from the main program. All modules are declared as macros and thus are only assembled when they are used. The main file also saves and restores the environment when needed during an interrupt service routine. Figure 7.6 shows the functional organisation of the software blocks. The code and a brief explanation for each of these modules is given in Appendix K.

The experimental results of the implementation are given in the next chapter.

8. EXPERIMENTAL RESULTS

A laboratory prototype of the active filter has been built to evaluate the performance of the proposed filter and its implementation in the TMS320F240 Evaluation Module. The active filter is realized using six bi-directional *IGBT* switches along with a series LC resonant circuit. In order to eliminate the detrimental effects of the dead time which corresponds to the commutation time of the switches a bi-directional bypass switch is connected across the terminals of the resonant circuit. The bi-directional switches are realized by using two *IGBT* switches in a back-to-back configuration. A three-phase diode-rectifier with a *dc* bus capacitor is used as a nonlinear load. The rectifier was loaded with a resistive load. The system is supplied with a voltage of $208V_{LL}$ with a frequency of 60 Hz. Figure 8.1 shows the block diagram of the experimental setup. The active filter is controlled using a TMS320F240 DSP. The DSP is connected to a PC through a XDS510PP emulator.

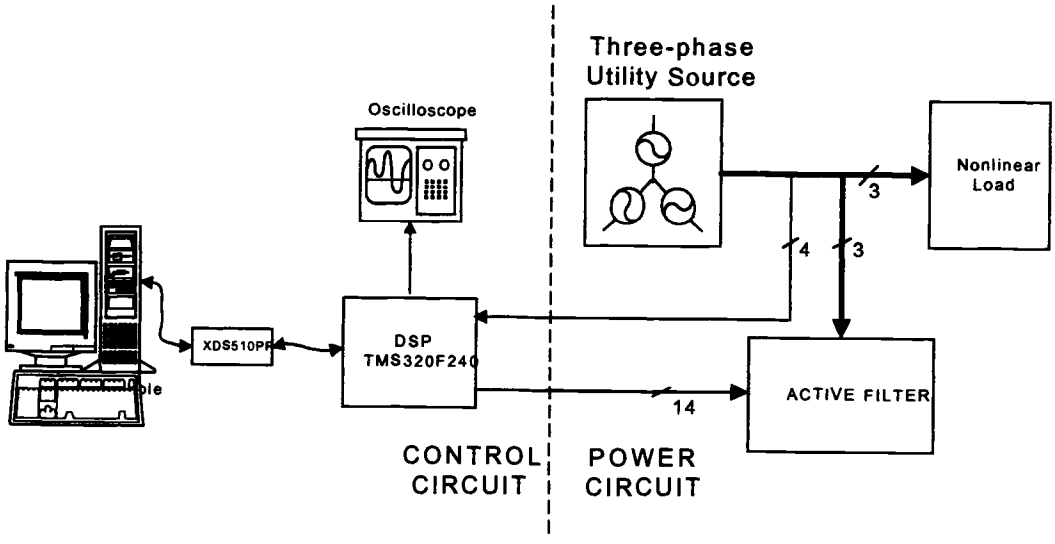


Figure 8.1 The laboratory setup of the hardware

The load for the experimental setup is a 2.6 kW resistive load composed of a three-phase diode rectifier with a *dc* smoothing capacitor of C_{rd}, = 2400 μF. The resonant circuit is realized using an inductor of L_o = 2.2mH and a capacitor of C_o = 9.9 μF with a quality factor of Q = 67. The resonant tank is tuned to 1080 Hz. The input LC low-pass passive filter is realized with an inductor of L_f = 1.81 mH and a capacitor of C_f = 12 μF. The LC passive filter is used to smooth the input voltage and current waveforms of the active filter. Figure 8.2 shows the complete circuit of the compensation system. The measurements were taken at points A, B, C, and D.

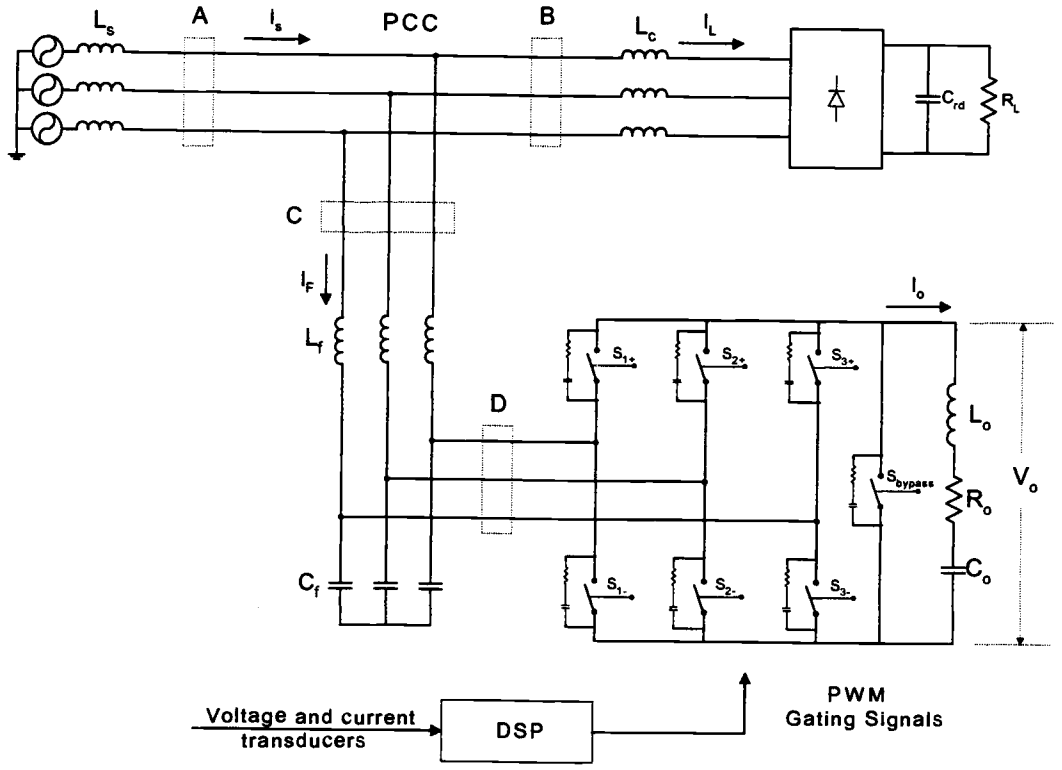


Figure 8.2 The complete circuit of the compensation system

As discussed before, the harmonics in power systems distort the source voltage due to the voltage drop on the power lines. The filter in this work is supplied from a three-phase supply source through a three-phase variable voltage

transformer. The transformer has a large winding impedance and therefore there is a significant voltage drop due to harmonic currents which distorts the sinusoidal shape of the voltage at the PCC. Figure 8.3 shows the source voltage at the PCC measured at point A (Figure 8.2) before and after the application of active filter.

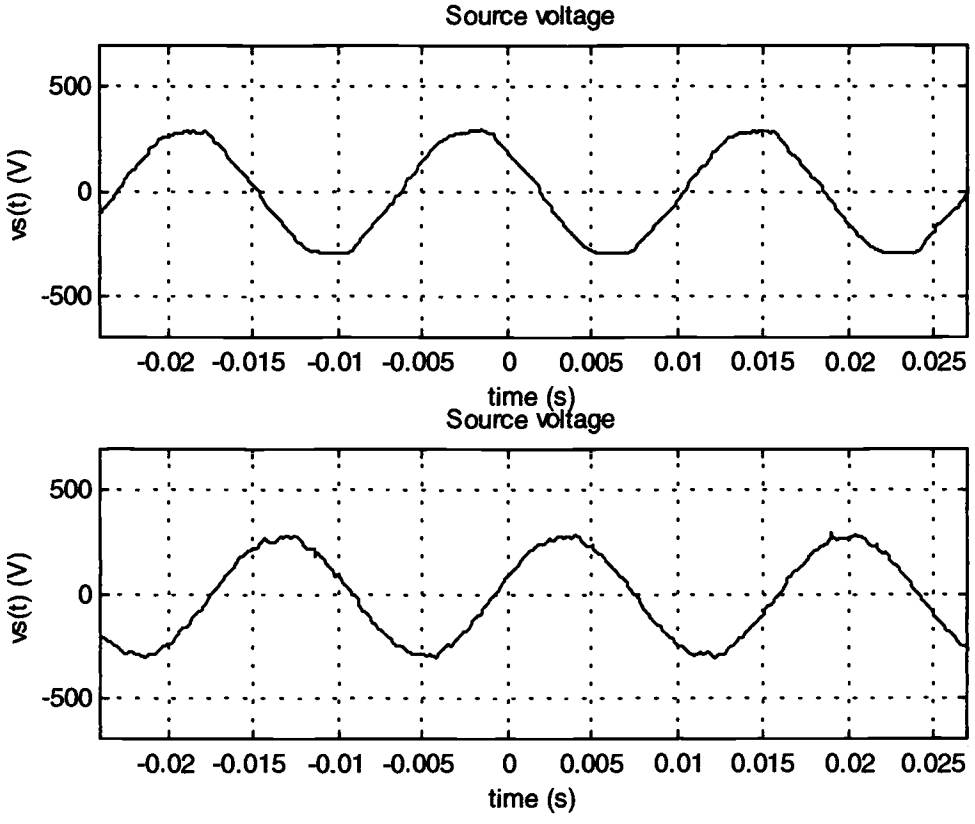


Figure 8.3 The source voltage without (top) and with active filter application (bottom)

It can be seen that the peak of the source voltage is reduced which can affect the operation of sensitive equipment. The active filter cancels harmonic currents and therefore reduces the voltage distortion significantly.

The load and source currents (measured at point A and point B respectively, Figure 8.2) before the application of active filter are shown in Figure 8.4 and Figure 8.5. It is obvious that the source and load currents are the same and contain a large amount of harmonic distortion with a THD of 70.67%.

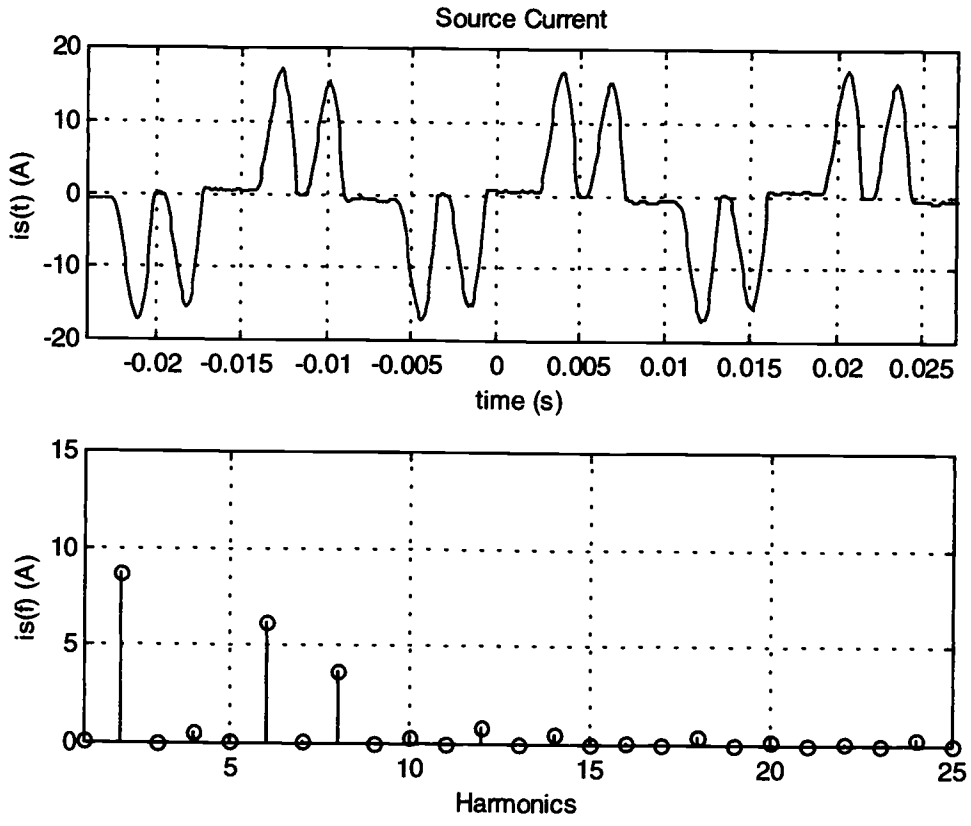


Figure 8.4 Source current before application of the active filter

It is reported in [31]-[32] that the impedance of a voltage source nonlinear load (nonlinear loads with a *dc* capacitor) is low and thus a series filter should be used instead of a shunt filter due to the fact that the compensation current may flow into the load side instead of the source. The nonlinear load that is used in this work is a voltage source with a large *dc* capacitor. The experimental results show that there is little current flowing into the load side. This can be seen by comparing Figure 8.5 to Figure 8.6. The amplitude of the load current after the application of the active filter (Figure 8.6) is slightly higher than before the application of the active filter (Figure 8.5). But the shunt active filter can still be used with a voltage source nonlinear load successfully. Figure 8.5 shows the load current (measured at point B) after the application of the active filter. Comparing Figure 8.5 with Figure

8.6 indicates that there is not a significant change in the harmonic content of the load current.

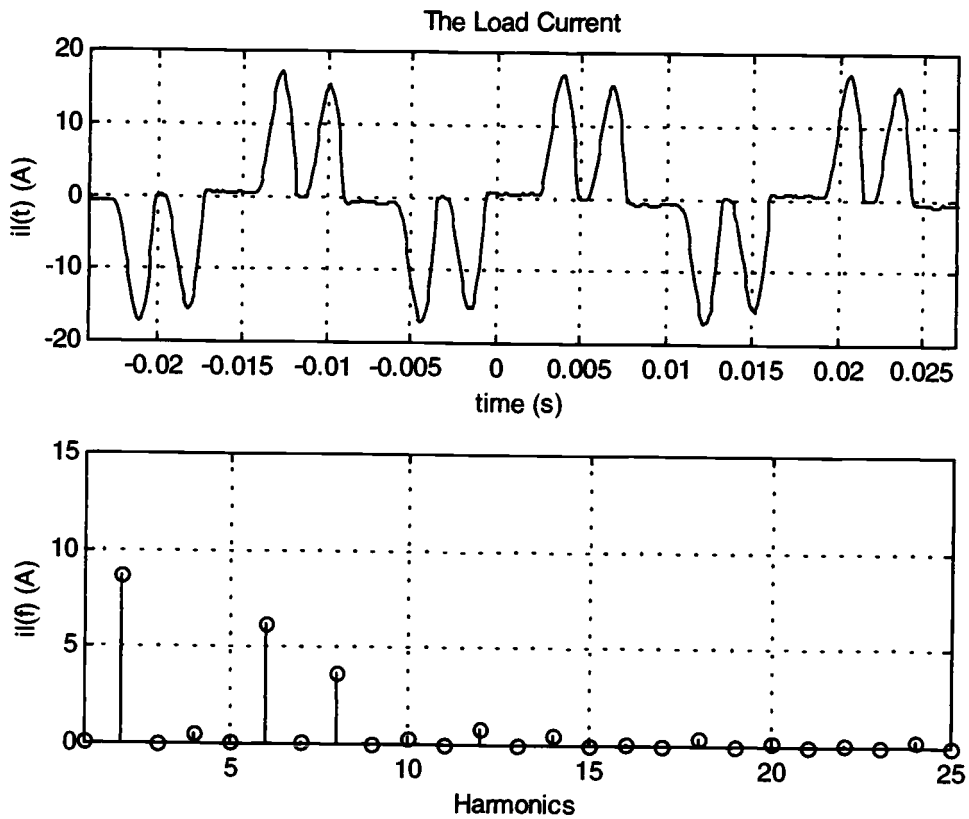


Figure 8.5 Load current before application of the active filtering

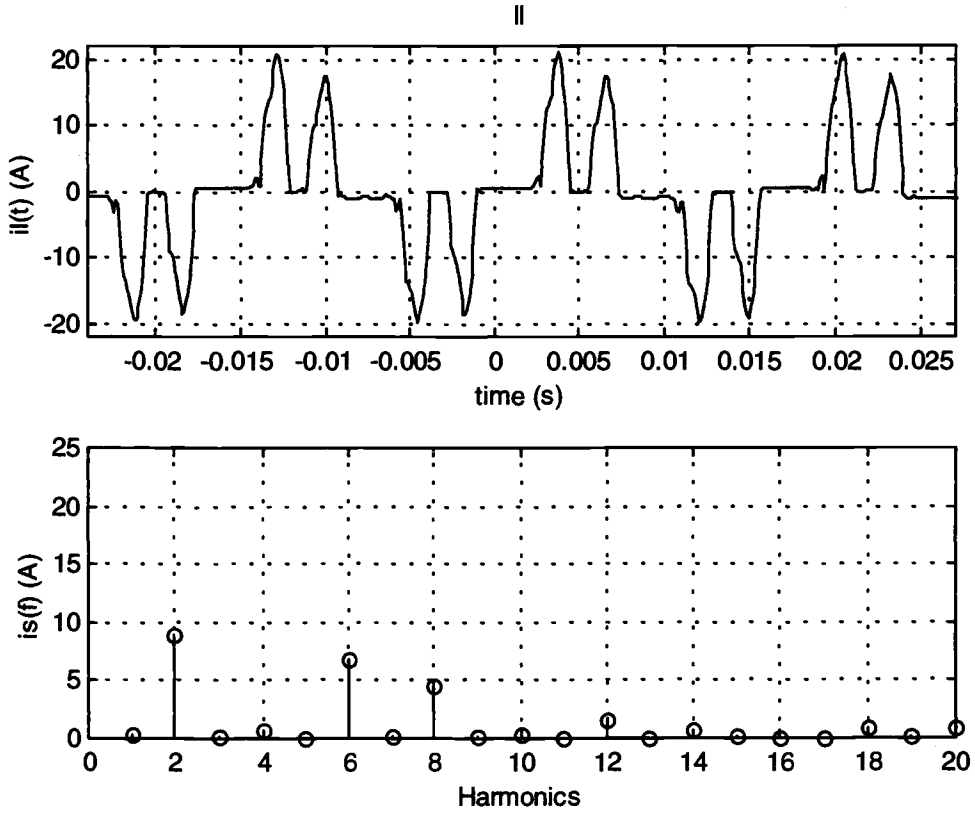


Figure 8.6 Load current after application of the active filter

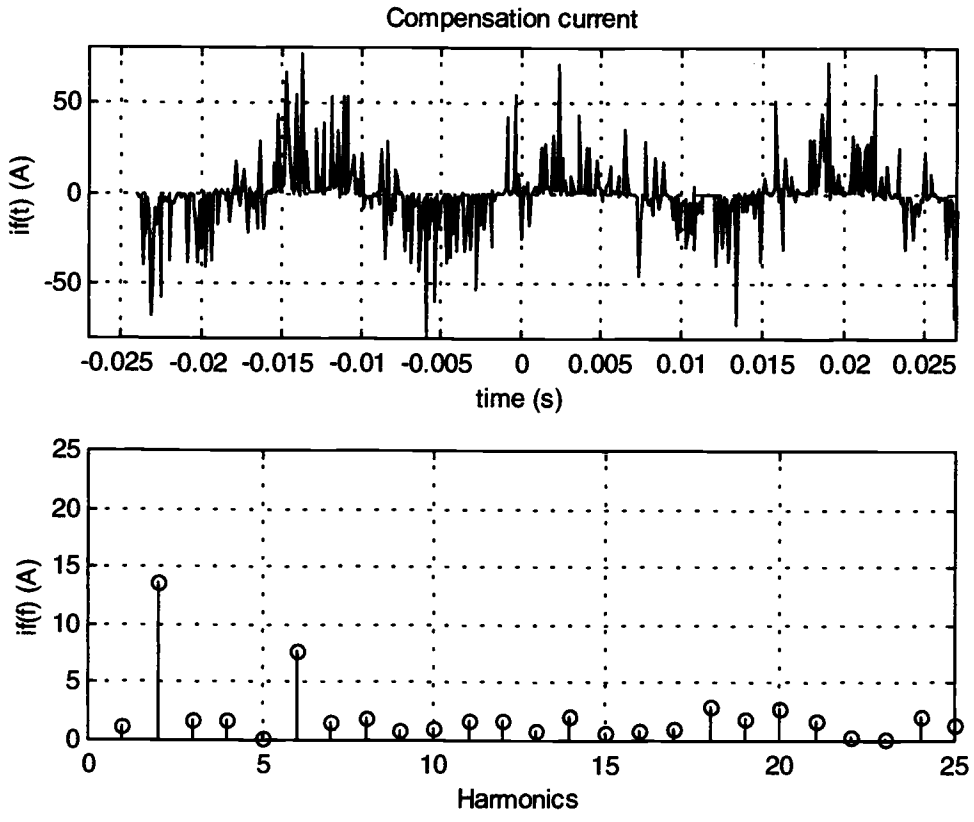


Figure 8.7 Compensation current measured at point D

Figure 8.7 indicates the input current of the active filter measured at point D (without LC input passive filter). In can be seen that the current generated by the active filter is quite discontinuous due to the high switching frequency. In addition, the filter current has very high current spikes due to the effects of the commutation of the current from one pair of switches to another. As mentioned before, the *DSP* generates only three *PWM* output signals and seven signals are generated from three PWM output signals by using a PAL. The PAL introduces more current

spikes due to the additional dead time resulting from delays. Furthermore, the current spikes distort the input voltage and their effects can be seen as notches. (Figure 8.9)

The use of an LC passive filter smoothes the input current of the active filter and eliminates the high switching harmonics at the input of the filter and thus leads to more continuous sinusoidal voltage and current waveforms. Figure 8.8 shows the active filter current with the use of an LC input passive filter (measured at point C). Comparing Figure 8.7 to Figure 8.8 shows that the LC passive filter eliminates high frequency harmonics and thus results in a continuous current with the desired input current magnitudes.

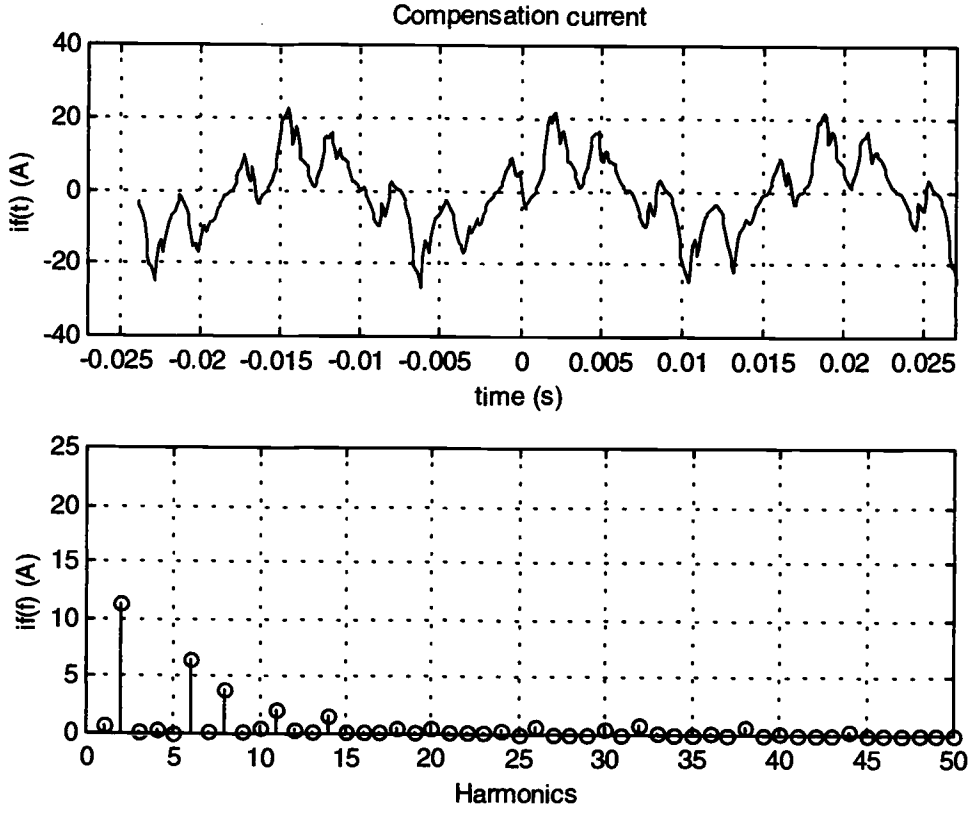


Figure 8.8 Compensation current measured at point C

The switching operation of the active filter also distorts the supply voltage. There are high spikes in the voltage due to the stray inductances of switches and

connectors and the inductor of resonant circuit. In addition, there are voltage notches due to the dead times of the active filter switchings. The use of an LC passive filter improves the quality of the input voltage and results in a sinusoidal waveshape. Figure 8.9 shows the input voltage (measured at point D) without the use of the LC passive filter and with the use of the LC passive filter (measured at point C).

The resonant tank of the active filter is tuned to 1080 Hz in order to generate the desired harmonics at the input of the filter. Figure 10 and Figure 11 show the resonant circuit voltage and current waveforms respectively. It can be observed that the resonant voltage and current mainly contains only 18^{th} harmonic (1080 Hz).

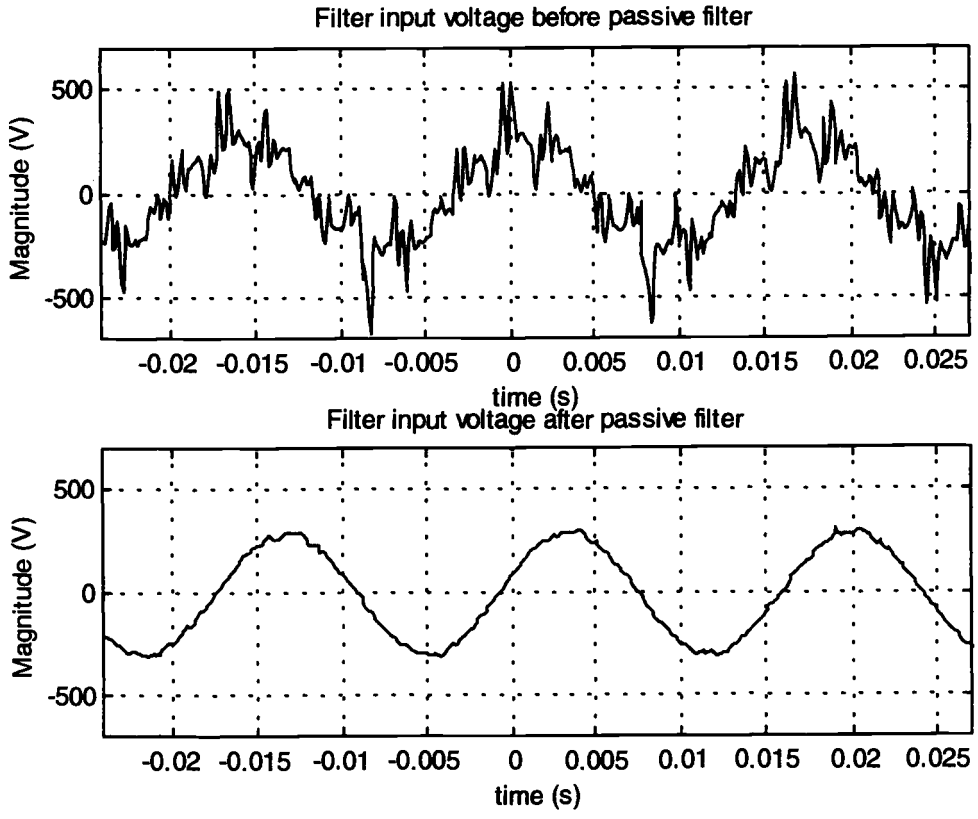


Figure 8.9 The filter input voltage with and without LC passive filter

The compensation performance of the active filter is given in Figure 8.12. It indicates the source current after the compensation currents are injected (measured at point A). The filter compensation currents cancel the load generated harmonics and thus results in a sinusoidal current with only small amounts of the 5^{th},7^{th},11^{th}, and 13^{th} harmonics. It can also be seen that the higher order harmonics such as 17^{th}, 19^{th} etc. do not carry any significant current. The THD of the source current after compensation currents are applied is reduced from 70.67% to 18.51%. Therefore there is a significant reduction in the total harmonic distortion.

Finally, the gating signals are given in Figure 13 through Figure 16.

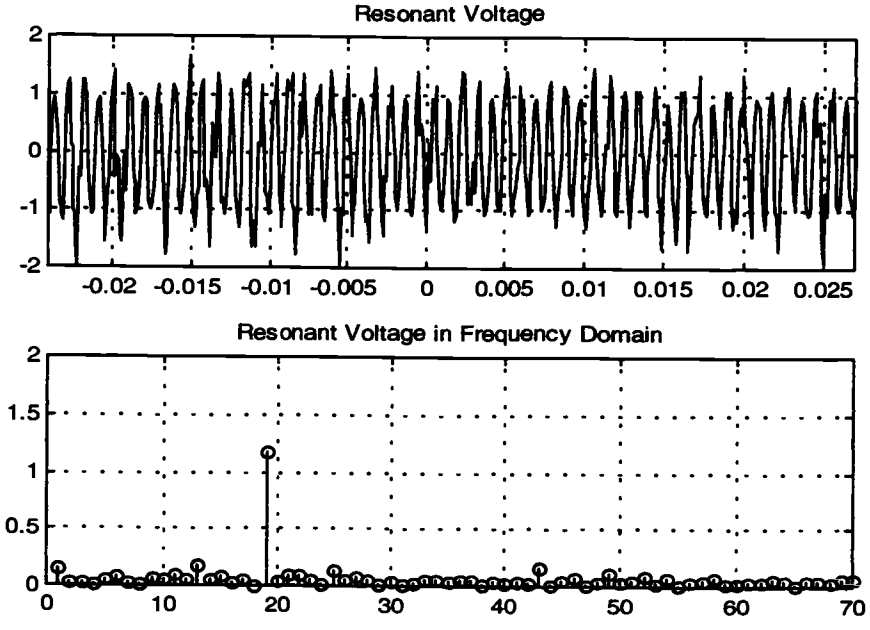


Figure 8.10 Resonant tank voltage

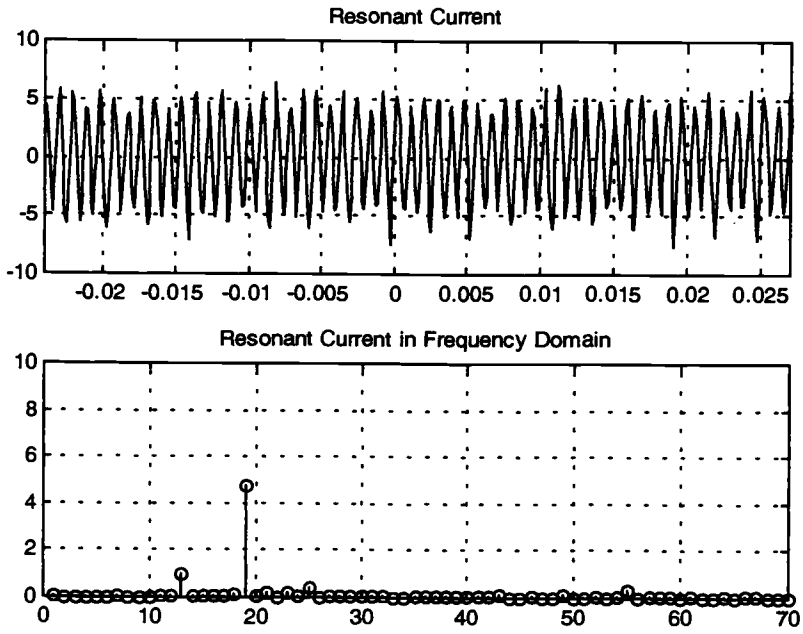


Figure 8.11 Resonant tank current

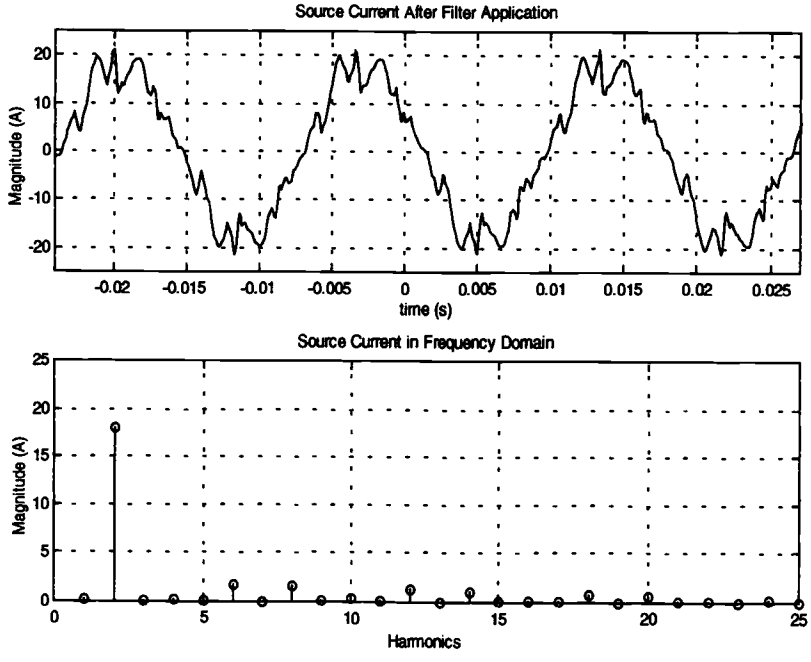


Figure 8.12 The Source current after compensation

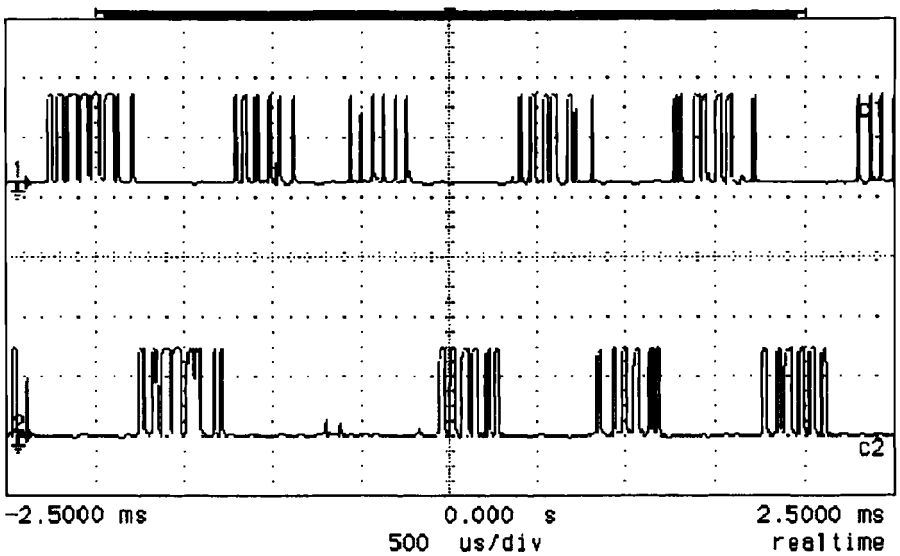


Figure 8.13 Gating signals for leg 1

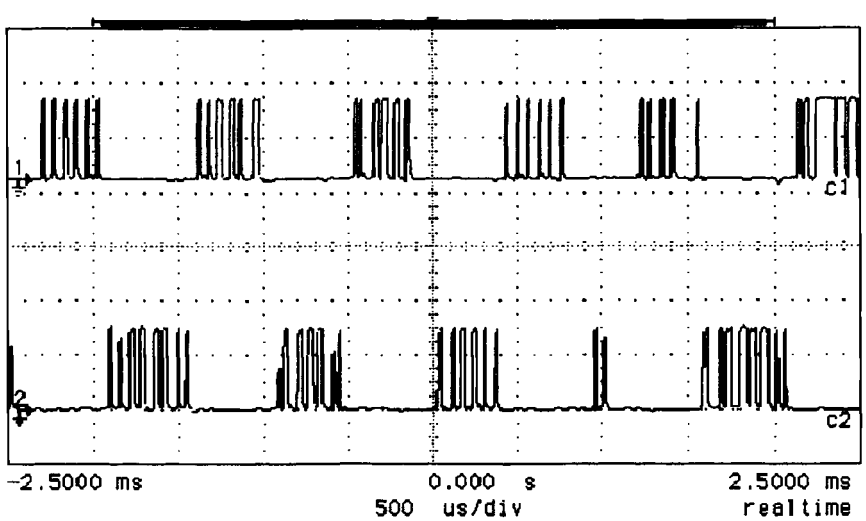


Figure 8.14 Gating Signals for leg 2

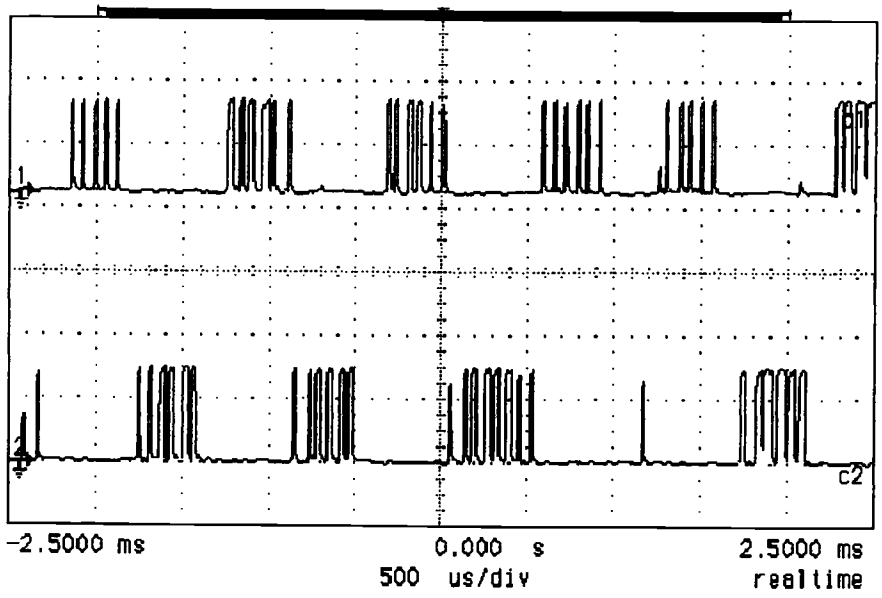


Figure 8.15 Gating signals for leg 3

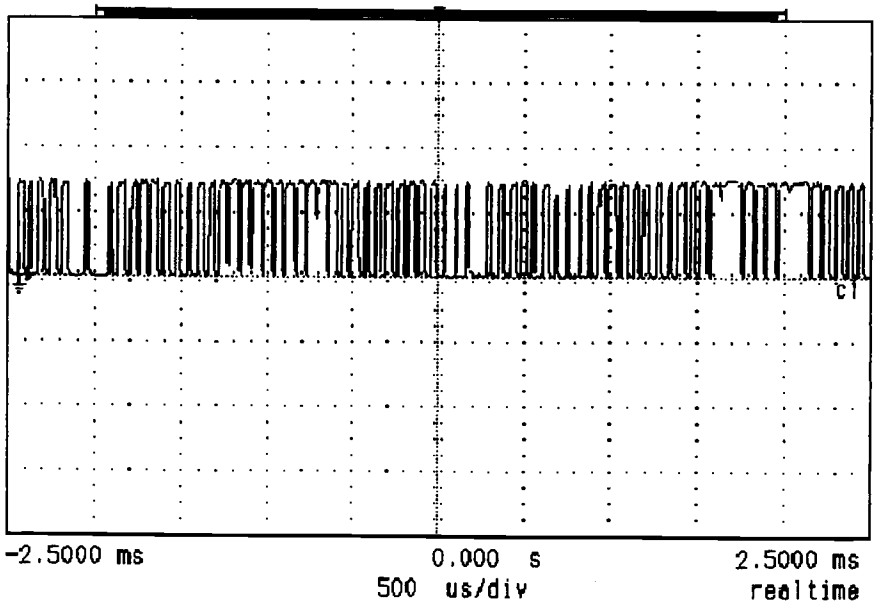


Figure 8.16 Gating signal for bypass switch

9. CONCLUSIONS AND RECOMMENDATIONS

9.1 SUMMARY OF RESULTS

The power quality problem has attracted great attention in the past decade as a result of the installation of more sensitive electronic equipment and the increased use of nonlinear loads. The nonlinear loads generate odd harmonics which deteriorate the power quality and affect the operation of sensitive electronic equipment. Three-phase nonlinear loads such as adjustable speed drives generate non-triplen harmonics with lower order 5^{th}, 7^{th}, 11^{th}, and 13^{th} harmonics being dominant. The higher order harmonics do not carry any significant current and their effect can be neglected.

The focus of this work was on the compensation for lower order 5^{th}, 7^{th}, 11^{th}, and 13^{th} harmonics in three-phase power systems. A three-phase PWM controlled shunt active filter was designed to inject harmonic currents with the same magnitude and opposite phase to that of nonlinear load currents in order to cancel harmonics at the point of common coupling. The filter topology is based on a three-phase rectifier topology with an LC resonant tank. The current in the LC resonant tank is reflected onto the input of the filter as harmonic currents. The main advantage of this topology is the ability to adapt to varying loads in real-time. The control of the filter is implemented in the frequency domain in a TMS320F240 DSP which provides the advantage of reduced hardware components and the use of FFT implementation.

A laboratory prototype has been built to verify the performance of the active filter. This involved the hardware and software design, including active filter, transducers, DSP input and output interface circuits, and the implementation of the control algorithm in assembly language. The feasibility of the approach was proven through the testing of the active filter. The active filter was connected in parallel to

a rectifier-type nonlinear load with a THD of 70.76%. After compensation the THD is reduced to 18.51% under 1.26% unbalanced input voltage conditions.

9.2 RECOMMENDATIONS FOR FUTURE WORK

The active filter draws a significant amount of fundamental current. The contribution to the fundamental current is mainly due to the switching losses, the losses in the passive LC elements and, the losses in the RC snubber circuits. Snubbers are used to reduce the high voltage spikes on the terminals of switches. The snubber circuit needs to be optimized to reduce the losses and fundamental current.

The proposed filter is designed for balanced power systems. It was tested under 1.26% of voltage unbalance successfully. There is always a certain degree of unbalance in the voltage of a power system due to unequal distribution of single-phase loads (typically <3%, as dictated in ANSI standard C84.1-1995). The derivation of modulating signals in terms of negative and positive sequence components needs to be investigated to incorporate effective operation under larger voltage unbalance conditions.

BIBLIOGRAPHY

[1] J.D. Veil, "Protecting digital equipment from power line disturbances," On Power.

[2] W.E. Reid, "Power quality issues-standards and guidelines," IEEE Transactions on Industry Applications, vol. 32, no. 3, pp. 625-632, May/June 1996.

[3] J.K. Phipps, J.P. Nelson, and P.K. Sen, "Power quality and harmonic distortion on distribution systems," IEEE Transactions on Industry Applications, vol. 30, no. 2, pp. 476-484, March/April 1994.

[4] T.H. Ortmeyer, and K.R. Chakravarthi, "The effects of power system harmonics on power system equipment," IEEE Transactions on Power Apparatus and Systems, vol. PAS 104, no. 9, pp. 2555-2563, September 1985.

[5] J.M. Clemmenson, "Power Quality site survey instrumentation and measurement techniques," 1990.

[6] R.D. Henderson, "Harmonics: The Effects on Power Quality and Transformers," IEEE Transactions on Industry Application, vol. 30, no. 3, pp. 528-532, May/June 1994.

[7] M. Cacciari, "On the significance of quality factor selection in harmonic filter design," IEEE 1991.

[8] J.S. Subjak & J.S. McQuilki, "Harmonics-causes, effects, measurements and analysis-update," IEEE 1989.

[9] IEEE Working Group on Power Systems, "Power system harmonics: An overview," IEEE Transactions on Power Apparatus and Systems, vol. PAS-102, no. 8, pp. 2455-2460, August 1983.

[10] T. Landers, "Are harmonics overheating your neutrals and panel boards?" Power Quality Assurance September/October 1997.

[11] J.J. Burke, "Utility characteristics affecting sensitive industrial loads," Power Quality Assurance November/December 1996.

[12] E.B. Makram R.B. Haines, & A.A. Girgis, "Effects of harmonic distortion in reactive power measurement," IEEE Transaction on Industry Applications, vol.28, No.4, July/August 1992.

[13] V.E. Wagner, "Effects of harmonics on equipment," IEEE 1992.

[14] I. Purkayastha & P.J. Savoie "Effects of harmonics on power measurement," IEEE Transaction on Industry Applications, vol.26, No.5, Sept/Oct 1990.

[15] T.M. Gruzs, "A survey of neutral currents in three-phase computer power systems," IEEE Transactions on Industry Applications, vol. 26, no. 4, pp. 719-725, July/August 1990.

[16] J.J Burker, "Power Quality –Two Different Perspectives," IEEE Transactions on Power Delivery, Vol. 5, No.3, July 1990

[17] R.C. Dugan, M.F. McGranaghan, and H. Wayne, "Electrical Power Systems Quality," New York: McGraw-Hill, 1996.

[18] W.E Kazibwe & M.H. Sendaula, "Electric Power Quality Control Techniques," New York: Van Nostrand Reinhold, 1993.

[19] G.T. Heydt, "Electric Power Quality," West LaFayette: Stars In A Circle Publications, 1994.

[20] A. van Zyl, "Voltage Sag and Momentary Interruption Ride-Through for Adjustable Speed Drives", Oregon State University Ph.D. Thesis, 1999.

[21] P.J.A. Ling & C.J. Eldridge, "Double the neutral conductor & derate the transformer what have we learned," Powersmith International Corp., June 1998.

[22] D.W. Egolf & A. J. Flechsig, "Harmonics-Transformer Derating" IEEE 0-7803-1877-3/94.

[23] C.K. Duffey, and P.P. Stratford, "Update of harmonic standard IEEE-519: IEEE recommended practices and requirements for harmonic control in power systems," IEEE Transactions on Industry Application, vol. 25, no. 6, pp. 1025-1030, November/December 1989.

[24] J.M. Clemmensen, "Estimating the cost of PQ", IEEE Spectrum, pp. 40-41, June 1993.

[25] A. Boyd, Michael, "Powering and Protecting the Modern Facility", Power Quality Assurance Magazine, pp. 56-65, January-February 1995.

[26] R. Redl, P. Tenti, J. Van Wyk, "Power Electronics' Polluting Effects", IEEE Spectrum, pp. 33-39, May 1997.

[27] D.E. Rice, "Adjustable speed drive and power rectifier harmonics-their effect on power systems components," IEEE Transactions on Industry Application, vol. IA -22, no. 1, pp. 161-177, January/February 1986.

[28] J.W. Gray & F.J. Haydock, "Industrial Power Quality considerations when installing adjustable speed drive systems," IEEE Transactions on Industry Applications, vol. 32, No. 3 May/June 1996.

[29] M. Rastogi, R Naik, and N. Mohan, "A comparative evaluation of harmonic reduction techniques in three-phase utility interface of power electronic loads," IEEE Transactions on Industry Applications, vol. 30, no. 5, pp. 1149-1155, September/October 1994.

[30] A. Mansoor et al, "Effect of supply voltage harmonics on the input current of a single-phase diode bridge rectifier loads," IEEE Transaction on Power Delivery, vol.10, No.3, July 1995.

[31] F.Z. Peng, "Application issues of active power filters," IEEE Industry Applications Magazine, pp. 21-30, September/October 1998.

[32] F.Z. Peng & D.J. Adams, "Harmonic sources and filtering approaches," IPCC 99.

[33] F.Z. Peng, "Application considerations and compensation characteristics of shunt active and series active filters in power systems,"

[34] S. Bhattacharya, T.M. Frank, D.M. Divan, and B. Banerjee, "Active filter system implementation," IEEE Industry Application Magazine, September/October 1998.

[35] S. Hansen, P. Nielsen, and F. Blaabjerg, "Harmonic cancellation by mixing non-linear single-phase and three-phase loads,"

[36] D.A. Gonzalez & J.C. Mccall, "Design of filters to reduce harmonic distortion in industrial power systems" IEEE Transactions on Industry Application, vol. IA-23, no. 3, pp. 504-509, May/June 1987.

[37] S.D. Upadya & Y.R. Atre, "Determination of the design parameters of passive harmonics filters using nonlinear optimization" 0-7803-4509-6/98 1998 IEEE.

[38] H. Cheng, H. Sasaki & N. Yorino, "A new method for both harmonic voltage and harmonic current suppression and power factor correction in industrial power systems," IEEE 1995.

[39] M.J Nave, "Power line filter design for switch-mode power supplies," New York: Van Nostrand Reinhold, 1991.

[40] I.M. Gotlieb, "Simplified practical filter design," Blue Ridge Summit: Tab Books, 1990.

[41] S. Kim, P.N. Enjeti, & P. Packebush, "A new approach to improve power factor and reduce harmonics in a three-phase diode rectifier type utility interface," IEEE Transactions on Industry Application, vol. 30, no. 6, pp. 1557-15561, December 1994.

[42] H. Akagi, "Trends in Active Power Line Conditioners," IEEE Transactions on Power Electronics, vol. 9, no. 3, pp. 263-268, May 1994.

[43] H. Akagi, Y. Kanazawa, & A. Nabae, "Instantaneous reactive power compensator comprising switching devices without energy storage components" IEEE Transactions on Industry Applications, vol. IA-20, No.3, May/June. 1984, pp.625-630.

[44] H. Akagi, A. Nabae, and S. Atoh, "Control strategy of active filers using multiple voltage-source PWM converters," IEEE Transactions on Industry Applications, vol. IA-22, No.3, May/June. 1986, pp.460-465.

[45] H. Akagi, Y. Tsukamoto, and A. Nabae, "Analysis and design of an active power filter using quad-series voltage source PWM converters," IEEE Transactions on Industry Applications, vol. 26, no. 1, pp. 93-98, January/February 1990.

[46] F. Z. Peng, H. Akagi, and A. Nabae, "A new approach to harmonic compensation in power systems-a combined system of shunt passive and series active filters," IEEE Transactions on Industry Applications, vol. 26, no. 6, pp. 983-989, November/December 1990.

[47] H. Fujita, and H. Akagi, "A practical approach to harmonic compensation on power systems-series connection of passive and active filters," IEEE Transactions on Industry Applications, pp. 1020-1025, November/December 1991.

[48] T. Tanaka, N. Koshio, H. Akagi, and A. Nabae, "Reducing supply current harmonics," IEEE Industry Applications Magazine, pp. 31-37, September/October 1998.

[49] F. Zeng, H. Akagi, and A. Nabae, "Compensation characteristics of the combined system of shunt passive and series active filters," IEEE Transactions on Industry Applications, vol. 29, no. 1, pp. 144-151, January/February 1993.

[50] F.Z. Peng, H. Akagi, & A. Nabae, "A novel harmonic power filter" IEEE-PESC Record 1988, pp.1151-1159.

[51] H. Akagi, "New Trends in Active Filters for Power Conditioning", IEEE Trans. on IA, Vol. 32, No. November/December, 1996, pp. 1312-1322.

[52] H. Akagi, S. Ogasawara, and H. Kim, "The theory of instantaneous power in three-phase four-wire systems: A comprehensive approach", 0-7803-5589-X/99 1999 IEEE.

[53] H. Akagi, H. Fujita, and K. Wada "A shunt active filter based on voltage detection for harmonic termination of a radial power distribution line", 0-7803-4943-1/98 1998 IEEE.

[54] H. Fujita, T. Ymasaki, and H. Akagi, "A hybrid active filter for damping of harmonic resonance in industrial power systems" IEEE Transactions on Power Electronics, vol. 15, no. 2, pp. 215-222, March 2000.

[55] B. Singh, K Al-Hadad, and A. Chadra, " A review of active filters for power quality improvement," ," IEEE Transactions on Industrial Electronics, Vol 46, no 5, October 1999.

[56] L. Rosetto & P. Tenti, "Using AC-Fed PWM converters as instantaneous reactive power compensator," IEEE-PESC Record 1990, pp.885-861.

[57] T. Furuhashi, S. Okuma, & Y. Uchikawa, "A study on the theory of instantaneous reactive power," IEEE Transactions on Industrial Electronics, vol. 37, No.1, Feb. 1990, pp.86-90.

[58] F.Z. Peng, G.W. Ott, & D.J. Adams, "Harmonic and reactive power compensation based on the generalized instantaneous reactive power theory for 3-phase 4-wire systems," Needs to be determined

[59] H. Akagi, A. Nabae, "The p-q theory in three-phase systems under non-sinusoidal conditions," ETEP Vol. 3, No.1, January/February 1993.

[60] C.L. Chen, C.E. Lin, & C.L. Huang "An active filter for unbalanced three-phase system using synchronous detection method," IEEE-PESC Record 1994, pp.1451-1455.

[61] Z. Radulovic & Asif Sabanovic, "Active filter control using a sliding mode approach," IEEE-PESC Record 1994, pp.177-182.

[62] D.A. Torrey, and A. M. A. M. Al-Zamel, "Single-phase active power filters for multiple nonlinear loads," IEEE Transactions on Power Electronics, vol. 10, no. 3, pp. 263-272, May 1995.

[63] S. Saetieo, R. Devaraj, & D.A. Torey, "The design and implementation of a three-phase active power filter based on sliding mode control," IEEE Transactions on Industry Applications, vol. 10, No.5, Sept/Oct. 1995, pp.993-1000.

[64] W.D. Gady, M.J. Smaothy, and A.H. Noyola, "Survey of active power line conditioning methodologies," IEEE Transactions on Power Delivery, vol. 5, no. 3, pp. 1536-1542, July 199***.

[65] B.K. Bose, "Microprocessors in Power Electronics", IEEE PRESS, New York 1987.

[66] L. Ananth & M.M. Morcos, " Power quality monitoring system: A case study in DSP-based solution for electric power industry" IEEE Power Engineering Review, vol. 19, no. 7, pp. 47-50, July 1999.

[67] Texas Instruments- Configuring PWM Outputs of TMS320F240 with Dead Band for different Power Devices, Application Report SPRA289 (1997).

[68] Texas Instruments- TMS320C24x DSP Controllers Reference Set Volume 1 (CPU, System and Instruction Set, March 1997, Revision A).

[69] Texas Instruments- TMS320C24x DSP Controllers Reference Set Volume 2 (Peripheral Library and Specific Devices, March 1997, Revision A).

[70] Texas Instruments- TMS320C1x/C2x/C2xx/C5 Assembly Language Tools, User's Guide (1995).

[71] Texas Instruments- TMS320C24x DSP Controllers Evaluation Modul, Technical References (1997).

[72] Texas Instruments- Dead-Time Generation on the TMS320C24x, Application Report SPRA371 (1997).

[73] Texas Instruments: TMS320C240, TMS320F240 DSP Controllers Data Sheet, 1996.

[74] Y. Tzou, and H. Yeh, "DSP-based adaptive repetitive control of a PWM inverters for UPS with very low harmonic distortion," IEEE 1996.

[75] S. Jeong, and M. Woo, "DSP-based active power filter with predictive current control," IEEE Transactions on Industrial Electronics, vol. 44, no. 3, pp. 329-336, June 1997.

[76] Q. Yu, and D. Shanshan, "DSP based active power filter for line-interactive UPS," pp. 884-888, IEEE IECON Conference 1995.

[77] M. Chang, R. Ou, and Y. Tzou, "DSP-based fuzzy control of bilateral IGBT PWM DC-to-AC and DC-to-DC converters for battery energy storage system," IEEE 1993.

[78] Y. Tang, and L Xu, " Vector control and fuzzy logic control of doubly fed variable speed drives with DSP implementation, "IEEE Transactions on Energy Conversion, vol. 10, no. 4, pp. 661-668, December 1990.

[79] R. Haldeman & Reto Voser, "User manual for TMS320F24X Controller Firmware, Release 1.0" Diploma Thesis, Berner Fachhochschule, Hochschule fuer Technik und Architektur Burgdorf (www.isburg.ch), Switzerland, 1999.

APPENDICES

APPENDIX A: LIST OF MATERIALS

APPENDIX B
TMS320F240 EVM DSP Board
PAL CE22V10Q-25PC (AMD)
Drives for PAL to Optocouplers (74HC04N, 2 Pieces)

APPENDIX C
R1-R8 12 Ohms
R9-R16 1051 Ohm
R17-R24 1151 Ohms
R25-R32 160 Ohms

C9-C16 1000 μF, 25V
C17-C24 474 nF
C25 C32 104 nF
D1-D8 Zener (1N751A)
U17-U24 Drives (IR2121)
U25-U32 Optocouplers (6N136)

APPENDIX D
TX1-TX Signals Transformer (120/15, 60 Hz)
C1-C8 470 μF, 50V
U1-U8 Single-phase diode rectifiers (BR86D)
U9-U16 15 Voltage regulator (7815)

APPENDIX E
R53-20 kOhm
R54 – R64 10 kOhm
R65-R74 9.3 kOhm
R75-R96 32.42 kOhm
U33-U53 Opamp (741, in Quads LM448M)

APPENDIX F
CT1-CT6 Current transformers (CR 8750-230)
R47-R52 12 Ohm

APPENDIX G
R33-R39 50 kOhm (2*25 Ohms - 5 Watts)
R40-R46 160 Ohms
U33-U39 Voltage Transducers (LV-25-P)

APPENDIX B: THE TMS320F240 DSP

TMS320F240 processor is a member of the TMS320 family of digital signal processors. The TMS320F240 Evaluation Module integrates the performance of a 16-bit fixed-point high DSP core and the on-chip peripherals of a microcontroller into a single-chip to meet a wide range of digital control applications. These applications are industrial motor drives, power inverters and controllers, automotive systems (such as anti-lock braking systems, electronic power steering, and climate control), appliance and HVAC blower/compressor motor controls, and robotics and CNC milling machines. At 20 million instructions per second (MIPS), it offers significant performance over traditional 16-bit microcontrollers and microprocessors in real-time applications. System performance can be enhanced through the use of advanced control algorithms for techniques such as adaptive control, Kalman filtering and state control. The TMS320F240 special architecture uses a 16-bit word length along with 32-bit registers for storing intermediate results, and two hardware shifters are available to scale numbers independently of the CPU. This unique structure minimizes the quantization and truncation errors and increases processing power.

The TMS320F240's unique peripheral unit, event manager, is application-optimized and enables the use of advanced control techniques for high-precision and high-efficiency full variable-speed control of all motor types. Included in the event manager are special purpose PWM generation functions, such as programmable dead-band function and space vector PWM state machine for three-phase motors that provides maximum efficiency in the switching of power switches. Three independent up/down timers, each with its own compare register, support the generation of asymmetric as well as symmetric PWM waveforms. Two of the four capture inputs are direct connections for quadrature encoder pulse signals from an optical encoder.

The following is a summary of F240 features:

CPU:

- 32-bit central arithmetic logic unit (CALU)
- 32-bit accumulator
- 16-bit x 16-bit parallel multiplier with a 32-bit product capability
- Three scaling shifters
- Eight 16-bit auxiliary registers with a dedicated arithmetic unit for indirect addressing of data memory

Memory:

- 544 words x 16 bits of on-chip data/program dual-access RAM
- 16K words x 16 bits of on-chip program ROM or flash EEPROM
- 224K words x 16 bits of maximum addressable memory space (64K words of program space, 64K words of data space, 64K words of I/O space and 32K words of global space)
- External Memory Interface Module with a software wait-state generator, a 16-bit address bus, and a 16 bit data bus
- Support of hardware wait-states

Program Control:

- Four-level pipeline operation
- Eight-level hardware stack
- Six external interrupts: power-drive protection interrupt, reset, NMI, and three maskable interrupts

Instruction Set:

- Source code compatibility with C2x, C2xx, and C5x fixed-point generations of TMS 320 family

- Single-instruction repeat operation
- Single-cycle multiply/accumulate instructions
- Memory block move instructions for program/data management
- Indexed-addressing capability
- Bit-reversed indexed-addressing capability for radix-2 Fast Fourier Transforms (FFTs)

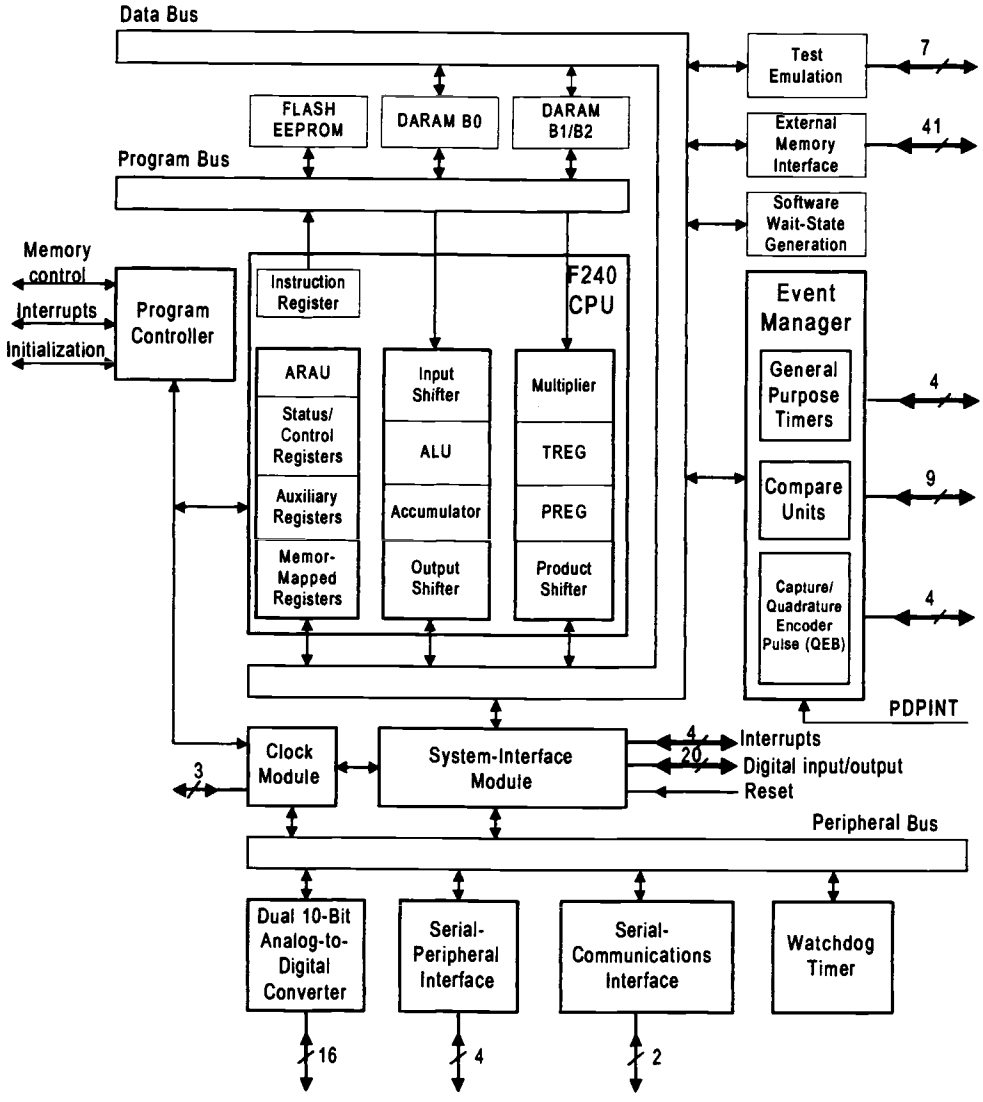


Figure B1 The functional block diagram of TMS320F240

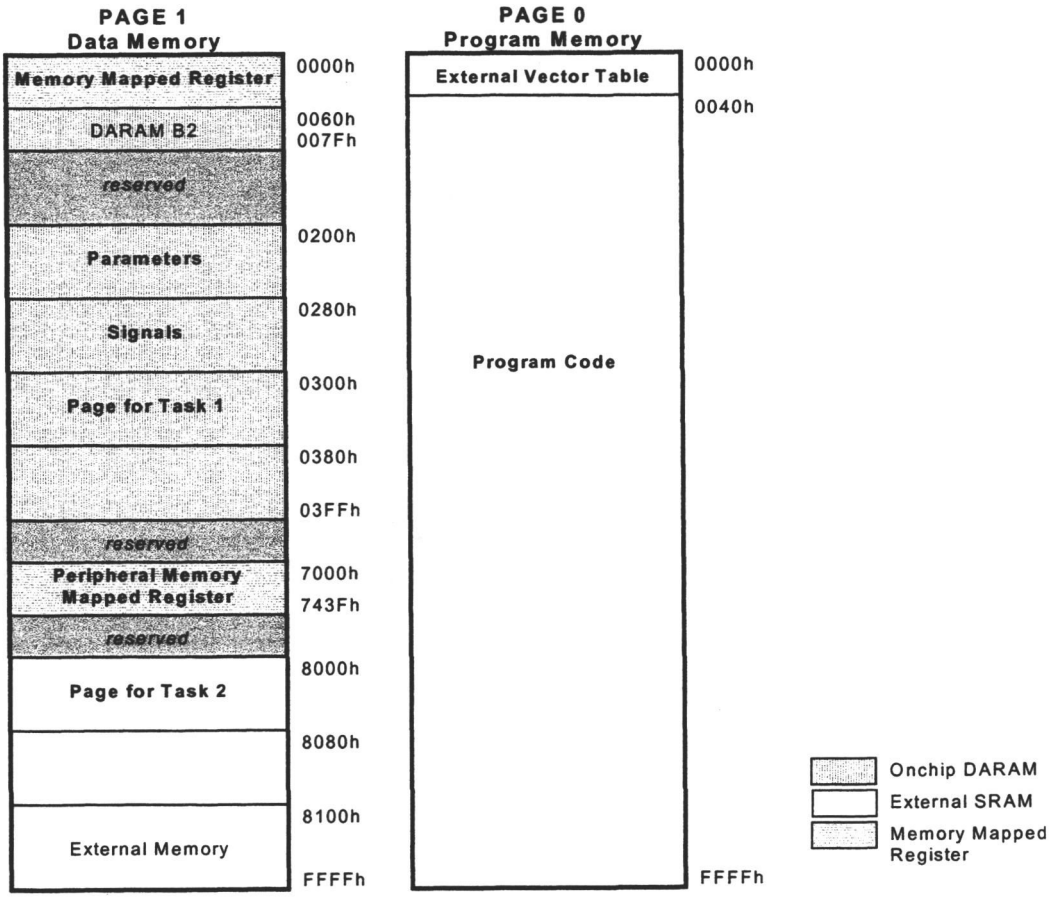


Figure B2 Memory Map of TMS320F240 Evaluation Board

Power:

- Static CMOS technology
- Four power-down modes to reduce power consumption

Event Manager:

- 12 compare/pulse-width modulation (PWM) channels (9 independent)
- Three 16-bit general-purpose timers with six modes, including continuous up counting and continuous up/down counting
- Three 16-bit full compare units with dead band capability

- Three 16-bit simple compare unit
- Four capture units, two of which have quadrature encode-pulse interface capability

Emulation: IEEE Standard 1149.1 test access port interface to on-chip scan-based emulation logic

Speed: 50 ns (20 MIPS) instruction cycle time, with most instructions single-cycle

Others:

- Dual 10-bit analog-to-digital converter
- 28 individually programmable, multiplexed I/O pins
- Phase-locked loop (PLL)-based clock module
- Watchdog timer module with real-time interrupt
- Serial communication interface (SCI)
- Serial peripheral interface (SPI)

APPENDIX C: THE DRIVER CIRCUITS FOR IGBT SWITCHES

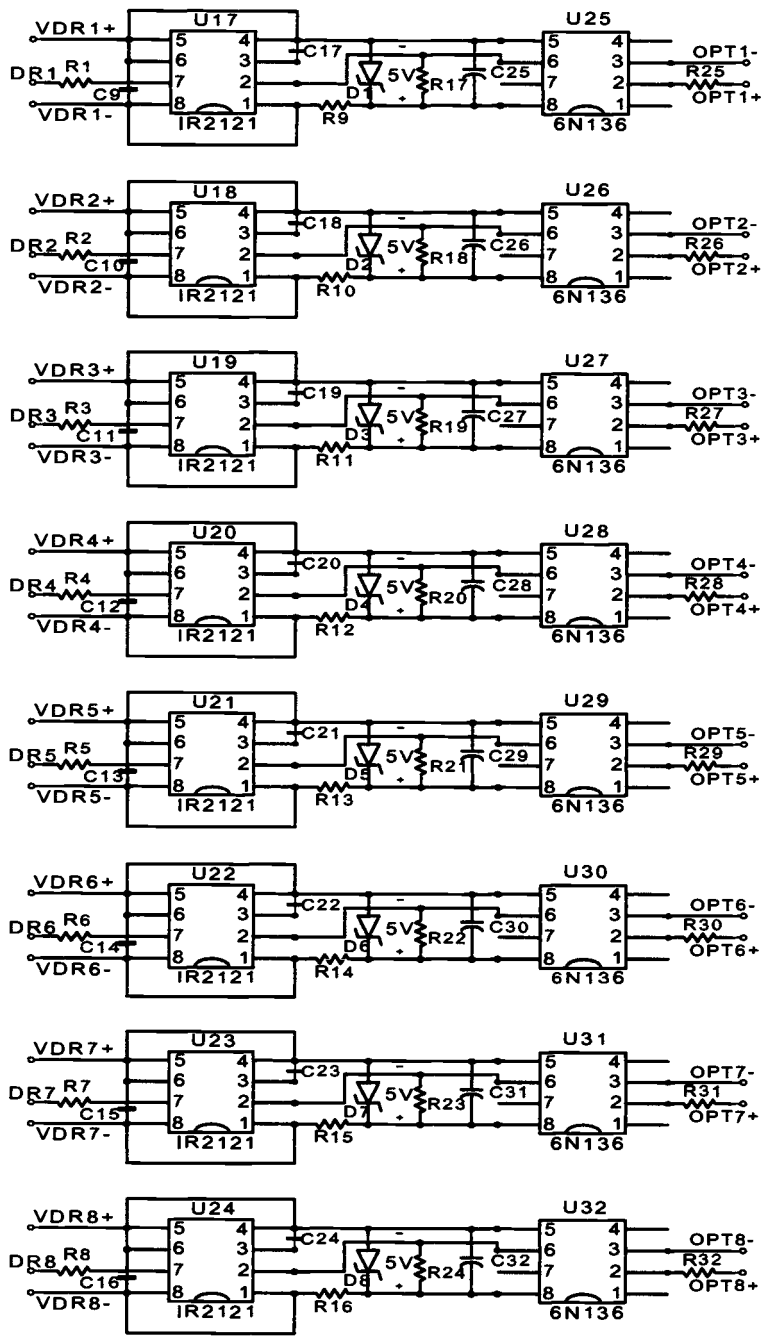


Figure C1 The driver circuits for IGBT switches

APPENDIX D: POWER SUPPLY CIRCUIT

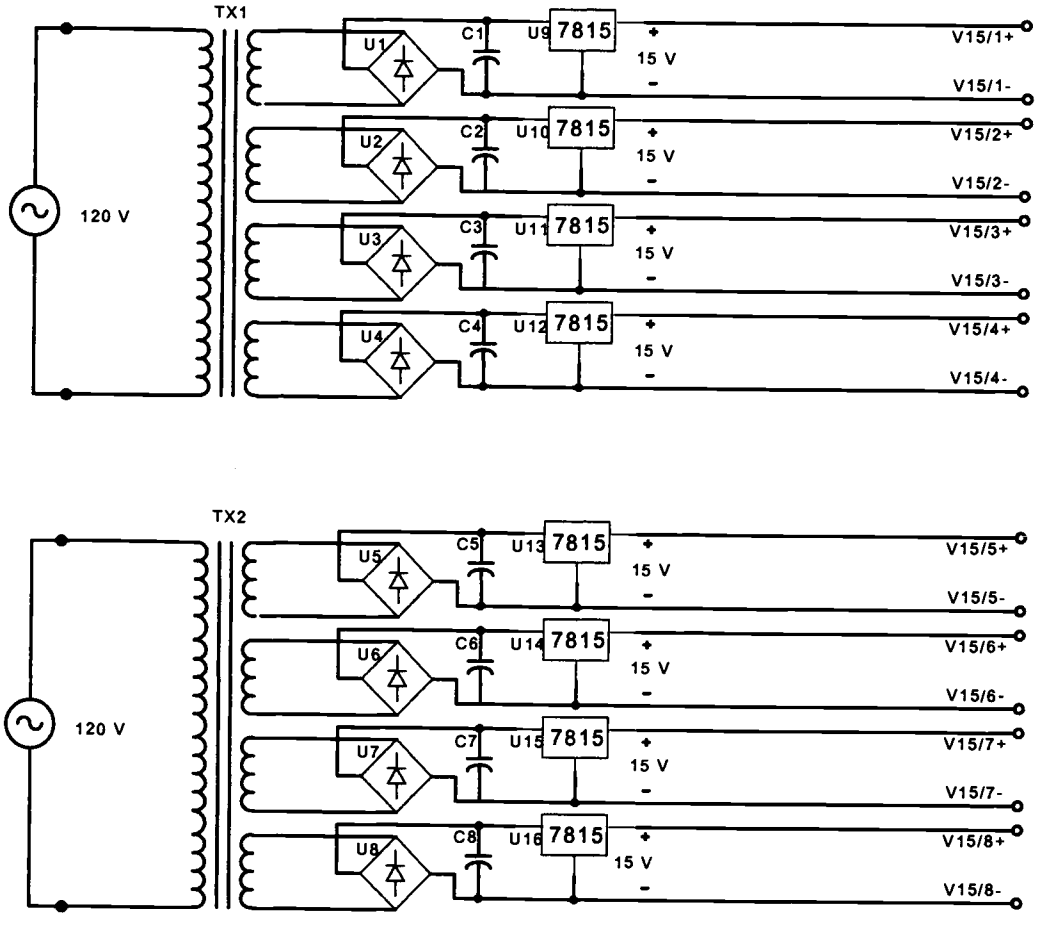


Figure D1 The power supply cicuit

APPENDIX E: CONDITIONING CIRCUIT

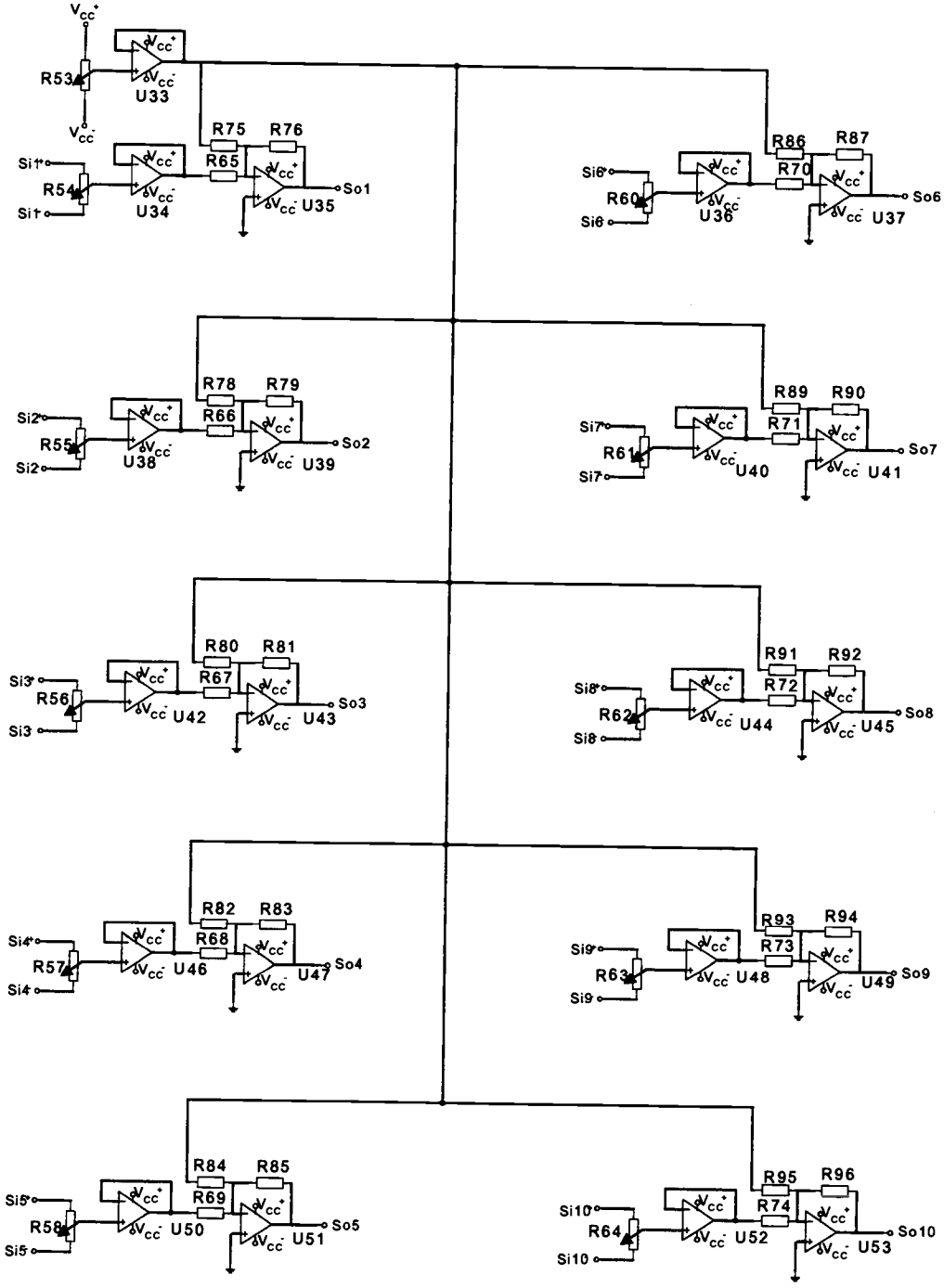


Figure E1 The signal conditioning circuit

APPENDIX F: CURRENT TRANSDUCERS

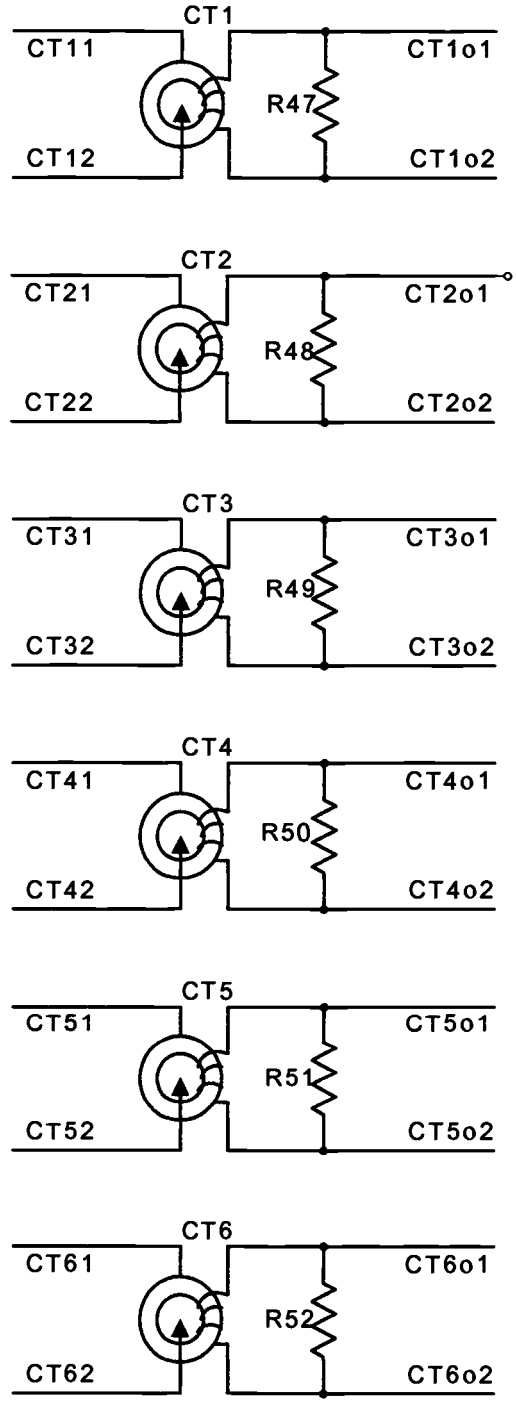


Figure F1 The current Transducers

APPENDIX G: VOLTAGE TRANSDUCERS

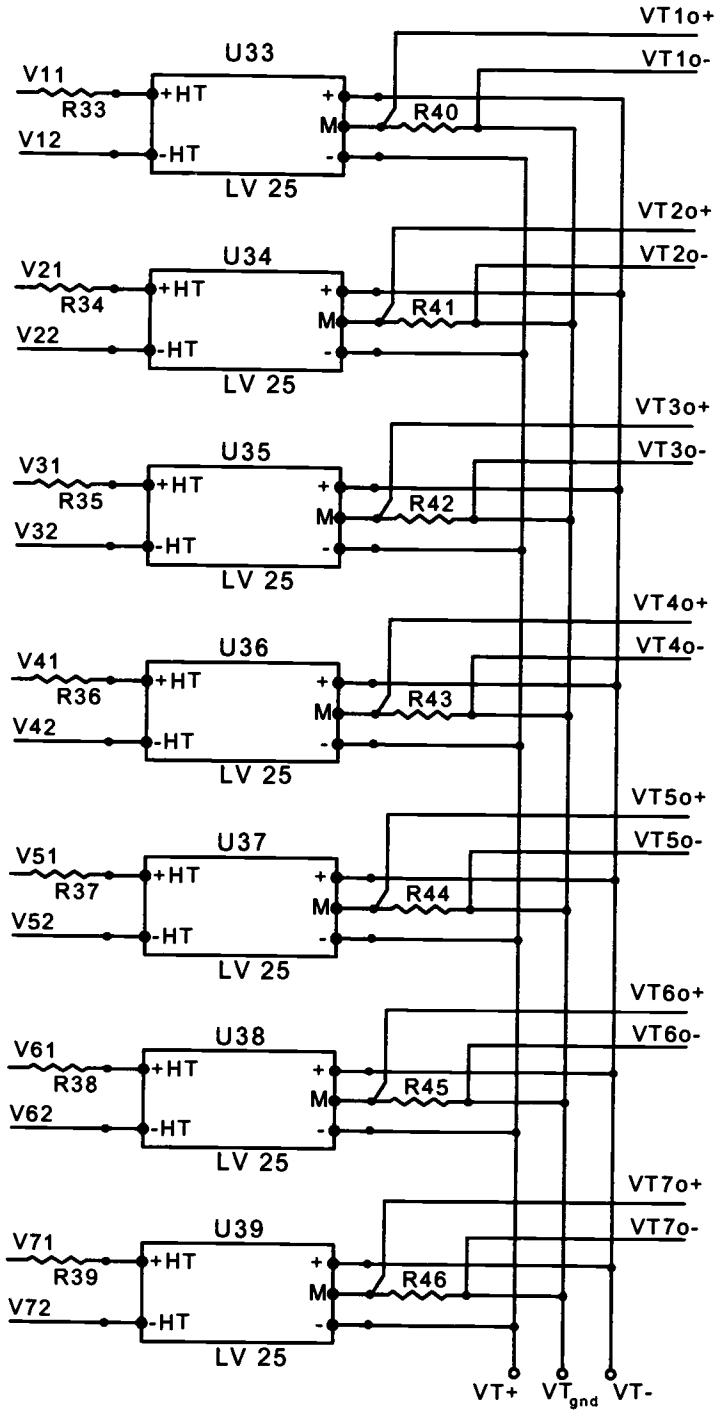


Figure G1 The voltage transducers

APPENDIX H: MATLAB PROGRAM LISTING

```
% Abdurrahman Unsal
% This is the program that generates the Switching signals in matlab for PSpice
% Simulation. The proceedure given here repesents the implementation of the control
% algorithm in DSP.
% Update 7/25/2000
clc;
close all;
clear all;

N=128;        % Size of FFT, number of samples in one period, provided by the programmer
T=1/(7680);   % Sampling period, provided by the programmer

freq=60;      % Fundamental Frequency
R=0.2264;     % The resistance of resonant circuit, provided by the programmer
h=19;         % harmonic coefficient, provided by the programmer
h5=h-6;       % Coefficient of 5th harmonic
h7=h-8;       %      "       of 7th    "
h11=h-12;     %      "       of 11th   "
h13=h-14;     %      "       of 13th   "
p1=pi/6;      % Phase angle of the switching function SW1
p2=p1-2*pi/3;%    "      "   of SW2
p3=p1+2*pi/3;%    "      "   of SW3
p3=p1+2*pi/3;%    "      "   of SW3
p3=p1+2*pi/3;%    "      "   of SW3

k=0:N-1;      % Number of samples
w=2*pi*freq*k;
              %
V=294;        % The amplitude of the line-to-line voltage

% The amplitude of fundamental line current (from a PSPICE simulation
% of nonlinear load
I1=17.85;     % The amplitude of the fundamental current
I5=15.8;      %         "             5th       harmonic
I7=14.48;     %         "             7th          "
I11=10.6;     %         "             11th         "
I13=8.3;      %         "             13th         "

% The simulation of the load current. The load current is assumed to be sensed by CT and
% read to the board to perform FFT to calculate the phase and amplitude of
% load current. Load current = Ila, asumming that the load is a 3-phase balanced
% load.

Ila=Ia1*sin(w*(T)-pi)+Ia5*sin(5*(w*T))+Ia7*sin(7*(w*T-pi))+Ia11*sin(11*(w*T))+Ia13*sin(13*(w*T-
pi));

If=fft(Ila,128);          % The FFT of the input current
Ia1=2*abs(If(2)/128);     % The magnitude of fundamental current.
Ia5=2*abs(If(6)/128);     %         "           5th     harmonic
Ia7=2*abs(If(8)/128);     %         "           7th        "
Ia11=2*abs(If(12)/128); %         "           11th       "
Ia13=2*abs(If(14)/128); %         "           13th       "

% The calculation of moduling coefficients

A=sqrt(4*R*Ia1/(3*sqrt(3)*V));     % The Coefficient corresponding fundamental current
A1=(4*R*Ia5/(3*sqrt(3)*V*A));      %         "           "                5th     harmonic
A11=(4*R*Ia7/(3*sqrt(3)*V*A));     %         "           "                7th        "
A111=(4*R*Ia11/(3*sqrt(3)*V*A));   %         "           "                11th       "
A1111=(4*R*Ia13/(3*sqrt(3)*V*A)); %         "           "                13th       "

% Switching functions;
% SW1, SW2, SW3 are the modulation signals each of these signals has to be
% intersected with a triangular wave of a frequency of 10kHz.  The resulting
% signals sig1, sig2, and sig3 are used to generate the gating signals of the
% power MOSFETs.
% S1 = sig1-sig3,   S2=sig2-sig1,  S3 = sig3-sig2;
%
% Each of these signals is applied to one leg of the active filter. The positive
% part (S1_pos, S2_pos, S3_pos)is applied to the upper switch of the leg, and the
% negative part (S1_neg, S2_neg, S3_neg)is rectified and then applied to the lower
% switch of the filter.
```

```
%
% There is a time when all of the switches are in off state, an error signal must me generated
% corresponding to this time and must be applied to both lower and uppers switches of
% one of the three legs.

SW1=A*sin(h*(w*T+p1))+A1*sin(h5*(w*T+p1))+A11*sin(h7*(w*T+p1))+...
   A111*sin(h11*(w*T+p1))+A1111*sin(h13*(w*T+p1));
%-----------------------------------------------
SW2=A*sin(h*(w*T+(p2)))+A1*sin(h5*(w*T+(p2)))+A11*sin(h7*(w*T+(p2)))+ ...
   A111*sin(h11*(w*T+(p2)))+A1111*sin(h13*(w*T+(p2)));
%-----------------------------------------------
SW3=A*sin(h*(w*T+(p3)))+A1*sin(h5*(w*T+(p3)))+A11*sin(h7*(w*T+(p3)))+ ...
   A111*sin(h11*(w*T+(p3)))+A1111*sin(h13*(w*T+(p3)));

% The generation of the error signal

% The following part of the code is not necessary for the DSP board. This part of the
% code generates the appropriate gating signals for the PSPICE simulation.

% Filter input line-to-line voltages
Vab=Va-Vb;
Vbc=Vb-Vc;
Vca=Vc-Va;

Vo=SW1.*Vab+SW2.*Vbc+SW3.*Vca; % Filter output voltage
Io=Vo/R;                       %    "      "    current

% Filter input currents
Ifa=(SW1-SW3).*Io;
Ifb=(SW2-SW1).*Io;
Ifc=(SW3-SW2).*Io;

%% The fundamental component of the supply current
% The modulating signals for PSpice simulation and gating signals.

SW1=A*sin(h*(2*pi*60*[0:100000]/100000+pi/6))+A1*sin((h-6)*(2*pi*60*[0:100000]/100000+...
pi/6))+A11*sin((h-8)*(2*pi*60*[0:100000]/100000+pi/6))+A111*sin((h-12)*...
(2*pi*60*[0:100000]/100000+pi/6))+A1111*sin((h-14)*(2*pi*60*[0:100000]/100000+pi/6));

SW2=A*sin(h*(2*pi*60*[0:100000]/100000-pi/2))+A1*sin((h-6)*(2*pi*60*[0:100000]/100000+...
5*pi/6))+A11*sin((h-8)*(2*pi*60*[0:100000]/100000-pi/2))+A111*sin((h-12)*...
(2*pi*60*[0:100000]/100000+5*pi/6))+A1111*sin((h-14)*(2*pi*60*[0:100000]/100000-pi/2));

SW3=A*sin(h*(2*pi*60*[0:100000]/100000+5*pi/6))+A1*sin((h-6)*(2*pi*60*[0:100000]/100000-...
pi/2))+A11*sin((h-8)*(2*pi*60*[0:100000]/100000+5*pi/6))+A111*sin((h-12)*...
(2*pi*60*[0:100000]/100000-pi/2))+A1111*sin((h-14)*(2*pi*60*[0:100000]/100000+5*pi/6));

%Generation of triangle wave for the control of the switches.
S=sawtooth(50000*pi*[0:100000]/100000,0.5);
% The gating signals.
sig1=sign((SW1)-max(SW1)*S);
sig2=sign((SW2)-max(SW2)*S);
sig3=sign((SW3)-max(SW3)*S);
ss1=sig1-sig3; % Three-level signals
ss2=sig2-sig1; % Three-level signals
ss3=sig3-sig2; % Three-level signals
% THE GATING SIGNALS
% Positive part phase a
ss1=ss1(1:1667); % One period of 60 Hz cycle
ss1_pos=ss1;
k=find(ss1_pos);
ss1_pos(k)=ss1_pos(k)+2; % The positive part of the signals
% Negative part phase a
ss1=ss1(1:1667);
ss1_neg=ss1;
l=find(ss1_neg);
ss1_neg(l)=ss1_neg(l)-2;
ss1_neg=-1*ss1_neg;
% Positive part phase b
ss2=ss2(1:1667);
ss2_pos=ss2;
m=find(ss2_pos);
ss2_pos(m)=ss2_pos(m)+2;
% Negative part phase b
ss2=ss2(1:1667);
ss2_neg=ss2;
n=find(ss2_neg);
ss2_neg(n)=ss2_neg(n)-2;
```

```
ss2_neg=-1*ss2_neg;
% Positive part phase c
ss3=ss3(1:1667);
ss3_pos=ss3;
o=find(ss3_pos);
ss3_pos(o)=ss3_pos(o)+2;
% Negative part phase c
ss3=ss3(1:1667);
ss3_neg=ss3;
p=find(ss3_neg);
ss3_neg(p)=ss3_neg(p)-2;
ss3_neg=-1*ss3_neg;
% Generating error signal.
error=ss1_pos+ss2_pos+ss3_pos+ss1_neg+ss2_neg+ss3_neg;
error=error-8;
error=-1*error/2;
%FINAL SWITCHING FILES FOR PSpice Simulation.
% Upper Leg a
ss1_pos=ss1_pos;
f11=find(ss1_pos>-1);
y11=[f11/100000;ss1_pos*(15/4)];
fid=fopen('SW11.txt','w');
fprintf(fid,'%1.6f  %2.0f\n',y11);
fclose(fid);
% Lower Leg a
ss1_neg=ss1_neg;
f12=find(ss1_neg>-1);
y12=[f12/100000;ss1_neg*(15/4)];
fid=fopen('SW12.txt','w');
fprintf(fid,'%1.6f  %2.0f\n',y12);
fclose(fid);
% Upper Leg b
ss2_pos=ss2_pos+error;
f21=find(ss2_pos>-1);
y21=[f21/100000;ss2_pos*(15/4)];
fid=fopen('SW21.txt','w');
fprintf(fid,'%1.6f  %2.0f\n',y21);
fclose(fid);
% Lower Leg b
ss2_neg=ss2_neg+error;
f22=find(ss2_neg>-1);
y22=[f22/100000;ss2_neg*(15/4)];
fid=fopen('SW22.txt','w');
fprintf(fid,'%1.6f  %2.0f\n',y22);
fclose(fid);
% Upper Leg c
f31=find(ss3_pos>-1);
y31=[f31/100000;ss3_pos*(15/4)];
fid=fopen('SW31.txt','w');
fprintf(fid,'%1.6f  %2.0f\n',y31);
fclose(fid);
% Lower Leg c
f32=find(ss3_neg>-1);
y32=[f32/100000;ss3_neg*(15/4)];
fid=fopen('SW32.txt','w');
fprintf(fid,'%1.6f  %2.0f\n',y32);
fclose(fid);
% Error signal for bypass switch
ferror=find(error>-1);
yerror=[ferror/100000;error*(15/4)];
fid=fopen('E.txt','w');
fprintf(fid,'%1.6f  %2.0f\n',yerror);
fclose(fid);
% Test for switching functions.
Va=169.7*sin(2*pi*60*[0:100000]/100000);
Vb=169.7*sin(2*pi*60*[0:100000]/100000-2*pi/3);
Vc=169.7*sin(2*pi*60*[0:100000]/100000+ 2*pi/3);
% The output voltage
Vo=(S1.*Va+S2.*Vb+S3.*Vc);
% The output current
Io=Vo/R;
% The input current
Ifa=(SW1-SW3).*Io;
Ifb=(SW2-SW1).*Io;
Ifc=(SW3-SW2).*Io;
Ifb=fft(Ifb,100000);
Figure
plot(abs(Ifb)/100000);
```

APPENDIX I: ASSEMBLY PROGRAM LISTING

```
;-------------------------------------------------------------------
;                    Oregon State University
;-------------------------------------------------------------------
;
;          File :          MAIN.ASM
;
;          Orginator :     Abdurrahman Unsal
;
;          Date :          05.26.2000
;
;          Target System : C24x Evaluation Board
;
;-------------------------------------------------------------------
;  Description:
;
;  This project-file contains the task-handling, the
;  System initialization and following modules
;
;                - Save environment for interrupt routine 'reg_save'
;                - Restore environment from interrupt.     'reg_rest'
;
;  All modules are declared as a macro and thus are only assembled when
;  they are used.
;
;-------------------------------------------------------------------
; declaration of global and temporary variables
;-------------------------------------------------------------------
          .globl _main,_task1,_task2
;-------------------------------------------------------------------
          .include vars.asm
          .include IO.asm     ; Input/Output-routines
          .include logic.asm  ; logical-routines
          .include Trans.asm
          .include Linear.asm
;-------------------------------------------------------------------
;      Imported Constants & Variables
;-------------------------------------------------------------------
  .globl _CKCR0, _T1CNT, _T2CNT, _CKCR1, _SYSCR, _WDCR, _WDKEY
  .globl _GPTCON, _T1PR, _T2PR, _T3PR, _T3CNT, _DBTCON, _CAPCON
  .globl _ACTR, _CMPR1, _CMPR2, _T1CON, _T2CON, _T3CON, _COMCON
  .globl _IFR, _IMR, _EVIFRA, _EVIMRA, _EVIFRB, _IFR,_ IMR,_ EVIFRA
  .globl _EVIMRA, _EVIFRB, _EVIMRB, _EVIMRC, _ADCFIFO1, _ADCFIFO2
  .globl _ADCTRL1, _ADCTRL2, _OCRA, _OCRB, _PADATDIR, _PBDATDIR
  .globl _PDDATDIR, _WSGR, _PCDATDIR, _SCMPR1, _SCMPR2, _SCMPR3
  .globl _SACTR, B0_SADR, BUF_SADR, B1_SADR, B2_SADR, TWID_TBL
  .globl I_BUF_IN, I_BUF_OUT, I_BUF_IN_PTR, I_BUF_OUT_PTR, I_BUF_WR_PTR
  .globl ZERO_CROSS, PS_TWID_STRT, COS45K, COS45, OFF_SET, temp_t3_1
  .globl positive_zero_crossing, old_sample,new_sample, PTR_A, PTR_B,
  .globl PTR_C, temp_t3_2, temp_t3_3, COPY_BUFFER, FIND_MAX_MIN, MIN,MAX
  .globl zero_counter, counter_125, counter_126, counter_127,counter_128
  .globl counter_129, counter_130, counter_131, Ia1, Ia5, Ia7, Ia11,Ia13
  .globl K5, K7, K11, K13, sqrtIa1, A, A1, A11, A111, A1111, C1, C2, R1,
  .globl R2, R3, R4,R5,FIRST_SINA,FIFTH_SINA,SEVENTH_SINA, ELEVENTH_SINA
  .globl ELEVENTH_SINA,THIRTEEN_SINA,FIRST_SINB,FIFTH_SINB, FIRST_SINC,
  .globl SEVENTH_SINB, ELEVENTH_SINB,FIFTH_SINC,THIRTEEN_SINB,vin, vout
  .globl SEVENTH_SINC,ELEVENTH_SINC,THIRTEEN_SINC ,MODUL_BUF_OUTA, vold
  .globl MODUL_BUF_OUTB,MODUL_BUF_OUTC,MODUL_BUFA,MODUL_BUFB,MODUL_BUFC
;-------------------------------------------------------------------
; MAIN PROGRAM
;-------------------------------------------------------------------
_main:
;-------------------------------------------------------------------
;         System Initialization
;-------------------------------------------------------------------
          SETC      INTM                ; disable maskable interrupts
          CLRC      SXM                 ; disable sign extension mode
          CLRC      OVM                 ; disable overflow mode
          CLRC      CNF                 ; DARAM configuration
          LACK      00BBh               ; CPU clock configuration
          LDPK      _CKCR1              ;       "
          SACL      _CKCR1              ;       "
          LACK      00C3h               ;       "
```

```
            LDPK        _CKCR0                  ;       "
            SACL        _CKCR0                  ;       "
            LDPK        _SYSCR                  ;       "
            SPLK        #40C0h,_SYSCR           ;       "
            LACK        006Fh                   ; disable watchdog
            LDPK        _WDCR                   ;       "
            SACL        _WDCR                   ;       "
            LACK        0055h                   ; kick watchdog
            LDPK        _WDKEY                  ;       "
            SACL        _WDKEY                  ;       "
            LACK        00AAh                   ;       "
            SACL        _WDKEY                  ;       "
            LDP         #temp_t1_1              ;
            SPLK        wait_state,temp_t1_1    ;
            OUT         temp_t1_1,_WSGR         ; configure wait-state-
                                                ; generator
;---------------------------------------------------------------------
        LDPK     _GPTCON      ;
        SPLK    #0000h,_GPTCON
        SPLK    #0000h,_T1CON
        SPLK    #0000h,_T2CON
        SPLK    #0000h,_T3CON
        SPLK    #0000h,_COMCON
        SPLK    #0000h,_ACTR
        SPLK    #0000h,_SACTR
        SPLK    #0000h,_DBTCON
        SPLK    #0000h,_CAPCON

        SPLK    #0FFFFh,_EVIFRA
        SPLK    #0FFFFh,_EVIFRB
        SPLK    #0000h,_EVIMRA
        SPLK    #0000h,_EVIMRB
        SPLK    #0000h,_EVIMRC

;---------------------------------------------------------------------
; Include Parameter-, Constant- and Signaldeclaration of application
;---------------------------------------------------------------------
        .include param.inc
;---------------------------------------------------------------------
; Taskinitialization
;---------------------------------------------------------------------
           CALL task_init                   ; initialize interrupt-sources,
                                            ;  timer 1 & 2 and start them
;---------------------------------------------------------------------
;   Task Cycle Visualisation via LED-port (only when enabled)
;---------------------------------------------------------------------

           .if task_LED = 1         ; compile only when enabled
LED        .usect      "param",1    ; memory allocation
           LDP         #_OCRA                ;
           SPLK        #0000h,_OCRA     ; IO-Port A configuration (8 Output)
           LDPK        LED
           SPLK        #0000h,LED   ; clear LED
           OUT         LED,000Ch    ;      "
           .endif

;---------------------------------------------------------------------
; Initialization of application
;---------------------------------------------------------------------
           .include init.inc    ; include of the init-calls
loop:   CALL task24_handling    ; handling of task 3 & 4
           register_update      ; update registers with new parameter
                                ; contents
           B           loop     ; loop forever

;---------------------------------------------------------------------
; MACROS AND ROUTINES
;---------------------------------------------------------------------

;**********************************************************************
;  task_init (procedure)
;  ---------
;  Description :
;  Initialize GP-timer 1 and 2 for continous up/down-counting. Timer 1
;  is used as timebase for task 1, which is also the timebase for
;  space-vector-modulation. Timer 2 handles as timebase for the slower ;  task (2)
;  Then the event-manager is configured to use the GPT1 periode
;  interrupt (T1PINT) as interrupt source for taskl and GPT2 periode
```

```
;  interrupt (T2PINT) as interrupt source for task 2
;  The corresponding service routines are called '_task1' and '_task2'.
;  The vectors to this routines are set in the vector-table 'VECS.ASM'.
;
;***********************************************************************
task_init:
    LDPK    _GPTCON             ;
    SPLK #0000101010101b,_GPTCON     ; Got them from application # 4

;-----------------------------------------------------------------------
;           Load timer-registers and counters
;-----------------------------------------------------------------------
   LDPK     _T1PR                       ; Load T1PR-register with init-value
   BLKD     #task1_periode,_T1PR
   LDPK     _T1CNT                              ; Clear counter 1
   SPLK     #0000h,_T1CNT
   LDPK     _T2CNT                              ; Clear counter 2
   SPLK     #0000h,_T2CNT
   LDPK     _T1CON                      ; cont. up/down-count of GPT1
   SPLK     #0A842h,_T1CON              ; internal clock clk/1 (prescale)

;-----------------------------------------------------------------------
;           Set interruptsource and enable corresponding interrupt
;-----------------------------------------------------------------------

    DINT          ; Disable unmasked  interrupts
    LDPK  _IFR         ; Clear all interruptflags in IFR
    LACL _IFR
    SACL _IFR
    LDPK _EVIFRA       ; Clear all interruptflags in EVIFRA
    LACL _EVIFRA
    SACL _EVIFRA
    LDPK _EVIFRB       ; Clear all interruptflags in EVIFRB
    LACL _EVIFRB
    SACL _EVIFRB
    LDPK _IMR
    SPLK #0002h,_IMR   ; Enable only INT2
    LDPK _EVIMRA       ; Enable T1PINT from EV-module -> task1
    SPLK #0080h,_EVIMRA
    LDPK _EVIMRB
    SPLK #0000h,_EVIMRB
    LDPK _EVIMRC       ; Disable all other int.sources
    SPLK #0000h,_EVIMRC

;-----------------------------------------------------------------------
;        Start Timer 1 and 2
;-----------------------------------------------------------------------
  LDPK      _T1CON
  LACL      _T1CON
  ORK       #0040h                                    ; Start GPT1
  SACL      _T1CON

;-----------------------------------------------------------------------
;       setup task counters
;-----------------------------------------------------------------------
   LDPK    task2_flag
   SPLK    #0,task2_flag                ; reset task2 flag
   LDPK    task2_counter
   BLKD    #task2_periode,task2_counter  ; load counter for task 3
   LDPK    task2_error
   SPLK    #0,task2_error               ; reset task 3 error
   LDPK    task4_counter
   BLKD    #task4_periode,task4_counter  ; load counter for task 4
   EINT                                              ; Enable interrupts
   RET

; ------------------- THE FIR FILTER -------------------------------
;
;
;    MACRO CALL
;    fir8: input, y, delayline, b_coeff
;
;    MACRO DESCRIPTION
;    calculates a fir filter according to:
;    y(n) = b(1)*x(n)+ b(2)*x(n-1)+ b(3)*x(n-2)+...+ b(16)*x(n-15)
;
;    Delay line consists of 16 delay tabs:
;    [x(n),x(n-1),...,x(n-15)]
```

```
;
;    Filter coefficient are stored as .word in PM in reverse order:
;    [b(16),b(14),b(13),...,b(1)]
;
;    IMPORTANT NOTE
;
;    INPUT PARAMETERS
;     input     : new input value x(n) [16/2]
;     delayline: adress to delay line [16/2]
;     b_coeff   : filter coefficients  [16/-4]
;
;    OUTPUT PARAMETERS
;     y         : output value y(n) [16/2]
;
;    CHANGED REGISTERS
;     ACC, TREG, PREG
;
;    Author                : Marcel Merk
;    Starting Date         : 03/10/2000
;    Update                : Abdurrahman Unsal
;
;-----------------------------------------------------------------------
  .include fir.dat                      ; fir filter parameter
fir8 .macro input, y, delayline, b_fir
fir8:
  LDP #input                            ; DP = input location
  BLDD input,#delayline                 ; x(n)=input
  MAR *,AR2                             ; set ARP -> AR2
  LAR AR2,#delayline+15                 ; AR2 -> end of delay line

  SPM 3                                ; [16/2]*[16/-4]SR 6 = [32/4]
  MAC b_fir,*-                         ; perform first multiplication
  LAC #0                               ; ACC = 0
  RPT #15                              ; multiply line with coeff.
  MACD b_fir+1,*                       ; and move data in delay line
  APAC                                 ; add last multiplication [32/4]
  ADD #1,13                            ; round(x)=trunc(x+0.5)
  LDP #y
  SACH y,2                             ; save result in [16/2]
  .endm
;***********************************************************************
;
; This module contains the following definitions :
;
; reg_save - Save environment for interrupt routine.
; reg_rest - Restore environment from interrupt.
;
; This is the list of registers preserved by these routines:
;
; P, T, ST1, ST0, ACC, AR0, AR2, AR3, AR4, AR5, AR6, AR7
;
; Additionally, the entire hardware stack is preserved.
;
; NOTE : The register AR1 is NOT preserved, therefor this register is
; not used in tasks
;
;***********************************************************************
; MACRO DEF : reg_save
;               This macro is used at the entry to an interrupt
;               handler, to save machine status, and set up a known
;               environment for the interrupt handler.
;***********************************************************************
reg_save .macro addr_of_regbackup
reg_save?:
   DINT                            ; disable maskable interrupts
   LARP  AR1                       ; CURRENT ARP = STACK POINTER
   LAR       AR1,#addr_of_regbackup ; start address of register and stack
                                   ; backup
   SST1      *+                    ; SAVE STATUS REGISTERS
   SST       *+
   SACH      *+                    ; SAVE ACCUMULATOR
   SACL      *+
   ROVM                            ; turn off overflow mode
   SPM           0                 ; product shift count of 0
   SPH       *+                    ; SAVE P REGISTER
   SPL       *+
   MPYK       1                    ; SAVE T REGISTER
   SPL       *+
```

```
    SAR     AR0,*+              ; SAVE AUXILIARY REGISTERS
    SAR     AR2,*+
    SAR     AR3,*+
    SAR     AR4,*+
    SAR     AR5,*+
    SAR     AR6,*+
    SAR     AR7,*+
    LAC     *                   ; PUT RETURN ADDRESS IN ACCUMULATOR
    RPTK    7                   ; SAVE HARDWARE STACK (8 ELEMENTS)
    POPD    *+
   .endm

;***********************************************************************
; MACRO DEF : reg_rest
;               This macro is used at the end of an interrupt
;               handler to restore the environment of the interrupted
;               code.
;             In order to ensure correct restoration of the T register,
;             this macro assumes that interrupts are disabled during
;             its execution.
;***********************************************************************
reg_rest .macro addr_of_regbackup
reg_rest?:
  DINT                              ; disable maskable interrupts
  LARP AR1
  LAR  AR1,#addr_of_regbackup+21    ; top address of register and stack
                                    ; buffer
  RPTK    7                         ; RESTORE TOP 8 ELEMENTS OF STACK
  PSHD    *-
  LAR     AR7,*-                    ; RESTORE ALL AUX REGISTERS EXCEPT AR1
  LAR     AR6,*-
  LAR     AR5,*-
  LAR     AR4,*-
  LAR     AR3,*-
  LAR     AR2,*-
  LAR     AR0,*-
  MAR     *-                        ; SKIP T REGISTER (FOR NOW)
  LT      *+
  MPYK    1                         ; RESTORE LOW PRODUCT REGISTER
  LT      *-                  ; RESTORE T REGISTER
  MAR     *-                  ; SKIP LOW PRODUCT REGISTER VALUE
  LPH     *-                  ; RESTORE HIGH PRODUCT REGISTER
  ZALS    *-                  ; RESTORE ACCUMULATOR
  ADDH    *-
  LST     *-                  ; RESTORE STATUS REGISTERS
  LST1    *-                  ; OLD ARP IS RESTORED *NOW*
  EINT                        ; RESTORE INTERRUPTS
  RET                         ; RETURN TO INTERRUPTED CODE
 .endm


;-----------------------------------------------------------------------
; TASK HANDLING
;-----------------------------------------------------------------------
;***********************************************************************
;  _task1 (interrupt-service-routine)
;  Description :
;  This macro is the service-routine of task 1. It will be called at
;  every timer 1 period match from the T1PINT-interrupt of the Event
;  Module. The corresponding application-code is included in TASK1.INC.
;
;***********************************************************************
_task1:
   reg_save registers_t1  ; save stack and registers (except AR1)
                          ; (interrupt requests are suppressed, INTM=1)
;-----------------------------------------------------------------------
;        start of code for task 1
;-----------------------------------------------------------------------
        .include task1.inc        ; include application code for task 1
;-----------------------------------------------------------------------
;      task cycle visualization via LED-port (only when enabled)
;-----------------------------------------------------------------------
  .if task_LED = 1          ; compile only when enabled
  LDPk      LED             ;
  LACC      LED             ;
  XOR       #0001h          ; task1 -> toggle bit 0
  SACL      LED             ;
  OUT       LED,000Ch       ; write to LED-port
```

```
  .endif
;-----------------------------------------------------------------------
;      modified by merkm
;-----------------------------------------------------------------------
    SSXM                        ;set sign extension
    LDPK task2_counter          ;
    LACC task2_counter          ;load counter of task 2 in ACCU
    BGZ elseT1_1                ;IF task2_counter < 0 THEN
      LDPK task2_flag           ;
      LACC task2_flag           ;  load flag of task 2 in ACCU
      BNZ elseT1_2              ;  IF task2_flag == 0 THEN
      SPLK #1,task2_flag        ;    task2_flag=1 (let task2 run)
        LDPK task2_counter      ;
        BLKD #task2_periode,task2_counter   ; reset task2_counter
        B endifT1                           ; ENDIF
elseT1_2:                               ;  ELSE
           LDPK task2_error             ;
           SPLK #1,task2_error          ;    task2_error=1; (time overflow)
           B endifT1                    ;  ENDELSE
elseT1_1:                               ; ELSE
          SUB   #1                      ;  task2_counter=task2_counter-1;
          SACL  task2_counter           ;
endifT1:                                ; end of code for task 1
;-----------------------------------------------------------------------
           LDPK       _EVIFRA      ; clear interrupt flags of interrupt
           LACL       _EVIFRA      ; group A in event-module
           SACL       _EVIFRA
           reg_rest registers_t1   ; restore stack and registers
                                   ; (except AR1), enables interrupts
                                   ; and return to interrupted code
;***********************************************************************
;  task2_handling (procedure)
;
;  Description :
;    This macro tests the two counters of task 3 & 4 whether they are
;    greater than zero or not. If a counter is zero or less, the
;    corresponding task is executed and the counter with the parameter
;    taskx_periode re-loaded.
;
;***********************************************************************
task2_handling:
;-----------------------------------------------------------------------
;         Check if task2 must be run
;-----------------------------------------------------------------------

        SSXM                            ; set sign extension
        LDPK    task2_flag
        LACC    task2_flag              ; load flag of task 2 in ACCU
        BZ elseT34_1                    ; IF task2_flag == 1 THEN

;-----------------------------------------------------------------------
;           application code for task 2
;-----------------------------------------------------------------------

        .include task2.inc                ;  doTask2

;-----------------------------------------------------------------------
;        task cycle visualisation via LED-port (only when enabled)
;-----------------------------------------------------------------------

          .if task_LED = 1              ; compile only when enabled
          LDPK LED                      ;
          LACC LED                      ;
          XOR  #0004h                   ; task2 -> toggle bit 2
          SACL LED                      ;
          .endif                        ;

;-----------------------------------------------------------------------
;          end of task2 code
;-----------------------------------------------------------------------

          LDPK task2_flag               ;
          SPLK #0,task2_flag            ;  task2_flag=0 (show that task is done)
elseT34_1:                              ;ENDIF

;-----------------------------------------------------------------------
;         chek if task2 error occured
;-----------------------------------------------------------------------
```

```
        SSXM
        LDPK    task2_error
        LACC    task2_error             ;load error flag of task 3 in ACCU
        BZ elseT34_2                    ;IF task2_flag <> THEN

;-----------------------------------------------------------------------
;       run error handler for task2
;-----------------------------------------------------------------------
          LDPK LED                      ;
          SPLK #10000000b,LED           ;  set MSB of LED to 1
doT34:    B doT34                       ;  stay endless in this loop
elseT34_2:                              ;ENDIF

;-----------------------------------------------------------------------
;      task cycle visualization via LED-port (only when enabled)
;-----------------------------------------------------------------------

           .if task_LED = 1                 ; compile only when enabled
           LDPk        LED                  ;
           LACC        LED                  ;
           XOR         #0008h               ; task4 -> toggle bit 3
.endif
t34e2:  RET                         ; return from subroutine

;-----------------------------------------------------------------------

 .def PHANTOM1, PHANTOM2, PHANTOM3, PHANTOM4, PHANTOM5, PHANTOM6
 .def PHANTOM7, PHANTOM8, PHANTOM9, PHANTOM10,PHANTOM11, PHANTOM12
 .def PHANTOM13, PHANTOM14, PHANTOM15, PHANTOM16, PHANTOM17,PHANTOM18
 .def PHANTOM19, PHANTOM20, PHANTOM21, PHANTOM22, PHANTOM23

;-----------------------------------------------------------------------

PHANTOM1 B PHANTOM1
PHANTOM2 B PHANTOM2
PHANTOM3 B PHANTOM3
PHANTOM4 B PHANTOM4
PHANTOM5 B PHANTOM5
PHANTOM6 B PHANTOM6
PHANTOM7 B PHANTOM7
PHANTOM8 B PHANTOM8
PHANTOM9 B PHANTOM9
PHANTOM10 B PHANTOM10
PHANTOM11 B PHANTOM11
PHANTOM12 B PHANTOM12
PHANTOM13 B PHANTOM13
PHANTOM14 B PHANTOM14
PHANTOM15 B PHANTOM15
PHANTOM16 B PHANTOM16
PHANTOM17 B PHANTOM17
PHANTOM18 B PHANTOM18
PHANTOM19 B PHANTOM19
PHANTOM20 B PHANTOM20
PHANTOM21 B PHANTOM21
PHANTOM22 B PHANTOM22
PHANTOM23 B PHANTOM23
```

```
;-------------------------------------------------------------------
;       Oregon State University
;-------------------------------------------------------------------
;
;           File :          COEFF.ASM
;
;           Orginator :     Abdurrahman Unsal
;
;           Date :          05.26.2000
;
;           Target System : C24x Evaluation Board
;
;-------------------------------------------------------------------
  .include Trans.asm
  .include Linear.asm
;-------------------------------------------------------------------
;             Defined symbols, variables, and labels       |
;-------------------------------------------------------------------
 .def mag_start, mag_end, coeff_end
;-------------------------------------------------------------------
;             Imported symbols, variables, and labels      |
;-------------------------------------------------------------
 .ref B1_SADR, temp_t3_1, temp_t3_2, temp_t3_3, sqrtIa1, C1, C2 A, A1
 .ref Ia1, Ia5, Ia7, Ia11, Ia13, K5, K7, K11, K13, A11, A111, A1111

;-------------------------------------------------------------------
;
;  Calculation of the coefficients
;
; The following code calculates the magnitude of the harmonics from the
;  fft. The input format is in the lower LSB bits of buffer the output
;  is in the (16,6) format.
; The result of FFT is as follows:
;              input  -> x
;              output -> y = 4*x^2 before taking square root.
;        If x = 3
;        y = 4*3^2 = 36
;        y = 24h
;  taking square root (in the following) => sqrt(24h) = c4ch which
;  corresponds to 3.074.
;
;  All the above results assume that the max hardware input is 16 A,
;  and the dsp adc has 5 v.
;
;
; The results of the FFT are stored in B1_SADR buffer
;-------------------------------------------------------------------
mag_start:
 LDP #(B1_SADR)             ; Load data page for B1_SADR (The FFT Output
                            ; buffer
 sqrt 3,(B1_SADR+1),Ia1     ; sqrt((Ia1)) = Ia1
 LDP #(B1_SADR)             ; data page for B1_SADR, FFT Output buffer
 sqrt 3,(B1_SADR+5),Ia5     ; sqrt((Ia5)) = Ia5
;-------------------------------------------------------------------
 LDP #(B1_SADR)             ; data page for B1_SADR, FFT Output buffer
 sqrt 3,(B1_SADR+7),Ia7     ; sqrt((Ia7)) = Ia7
;-------------------------------------------------------------------
 LDP #(B1_SADR)             ; data page for B1_SADR FFT Output buffer
 sqrt 3,(B1_SADR+11),Ia11 ; sqrt((Ia11)) = Ia11
;-------------------------------------------------------------------
 LDP #(B1_SADR)             ; data page for B1_SADR FFT Output buffer
 sqrt 3,(B1_SADR+13),Ia13 ; sqrt((Ia13)) = Ia13
mag_end                     ; End of magnitude calculation

constants_start:            ; The start of constants

;-------------------------------------------------------------------
;  A = C'*sqrt(Ia1) where C' = sqrt((4*R)/(3*sqrt(3)*V))
;  C' in floating point = 0.0455251
;  C' in (16,6) fixed format = 47 = 2Fh -> to get the correct result
;  the constant C is scaled (shifted four bit to the left -> 02F0h
;
;-------------------------------------------------------------------
;C1 .usect ".buffers",1  ;  C1
;    LDP #(C1)
;    SPLK #05d0h,C1
;    Sqrt(Ia1) in (16,4) format.
```

```
;     sqrt_Ia1 is a modified version of sqrt to get the correct results
;     in (16,4) format.
;     C2 .usect ".buffers",1  ;  C2
;     LDP #(C2)
;     SPLK #0174h,C2
;
 sqrt_Ia1 3,(Ia1),sqrtIa1     ; Sqrt(Ia1),The square root of Fundamental
;
; The result is in (16,6) format.
;-------------------------------------------------------------------------
; The division, output is in (16,6) format

DV: ;The start of  divisions
 div2 3, Ia5,sqrtIa1,K5        ; Constant5 in (16,4) format)
 div2 3, Ia7,sqrtIa1,K7        ; Constant7 in (16,4) format)
 div2 3, Ia11,sqrtIa1,K11      ; Constant11 in (16,4) format)
 div2 3, Ia13,sqrtIa1,K13      ; Constant13 in (16,6) format)
constants_end                  ; End of constants

coeff_start:                   ; Coefficient Calculations

;                              Coeff A in (16,6) format
  mult2 C1,sqrtIa1,A       ; A = C1*sqrt(Ia1)
;
;  C1 is shifted four bits to the right.
;
;  The coefficient A in (16,4) format
;-------------------------------------------------------------------------

 mult2 C2,K5,A1            ; A1
 mult2 C2,K7,A11           ; A11
 mult2 C2,K11,A111         ; A111
 mult2 C2,K13,A1111        ; A1111

coeff_end    RET           ; End of calculations

; The results are in (16,4) format.
```

```
;-------------------------------------------------------------------------
;       Oregon State University
;-------------------------------------------------------------------------
;           File :  MODUL.ASM
;
;           Orginator :     Abdurrahman Unsal
;           Date :          05.26.2000
;
;
;           Target System : C24x Evaluation Board
;
;-------------------------------------------------------------------------
;      Defined variables
   .def modul_start, COPY_BUFFER

;  Imported variables, symbols and labels
  .ref A, A1, A11, A111, A1111,R1,R2,R3,R4,R5, FIRST_SINA,
  .ref FIFTH_SINA, SEVENTH_SINA, ELEVENTH_SINA,  THIRTEEN_SINA
  .ref FIRST_SINB,FIFTH_SINB,SEVENTH_SINB,ELEVENTH_SINB,THIRTEEN_SINB
  .ref FIRST_SINC,FIFTH_SINC,SEVENTH_SINC,ELEVENTH_SINC,THIRTEEN_SINC
  .ref MODUL_BUF_OUTA, MODUL_BUF_OUTB, MODUL_BUF_OUTC, MODUL_BUFA
  .ref MODUL_BUFB,MODUL_BUFC
;-------------------------------------------------------------------------
;       Modulating Sin signals for phase A
;-------------------------------------------------------------------------

modul_start:

;============= Calculation of the Modulation signal A ==================

   LAR AR0,#127               ; AR0=127
   LAR AR2,#(FIRST_SINA)      ; AR2 = Fundamental Frequency
   LAR AR3,#(FIFTH_SINA)      ; AR3 = Fifth harmonic freq
   LAR AR4,#(SEVENTH_SINA)    ; AR4 = Seventh harmonic freq
   LAR AR5,#(ELEVENTH_SINA)   ; AR5 = Eleventh harmonic freq
   LAR AR6,#(THIRTEEN_SINA)   ; AR6 = Thirteenth harmonic freq
   LAR AR7,#(MODUL_BUF_OUTA) ; AR7 = Modulation buffer
   SOVM                              ; Set Overflow mode
   SETC     SXM                   ; Set sign-extension mode
;-------------------------------------------------------------------------
MODUL_BEGINA:
    MPYK #0h      ; Clear PREG
    LACC #0000h   ; Clear Acc.
    LDP #(A)      ; Load data page for A
    LT A          ; Load with A, TREG = A
    MAR *,AR2     ; ARP = 2, points to FIRST_SINA
    MPY *+        ; PREG = A*FIRST_SINA
    APAC          ; ACC = PREG
    LDP #(R1)     ; Load data page for R1
    SACH R1       ; Store Accumulator to R1
;-------------------------------------------------------------------------
   ZAC            ; Clear ACC
   LDP #(A1)      ; Load data page for A1
   LT  A1         ; Load with A1, TREG = A1
   MAR*,AR3       ; ARP = 3, points to FIFTH_SINA
   MPY *+         ; PREG = A1*FIFTH_SINA
   APAC           ; ACC = PREG
   LDP #(R2)      ; Load data page for R2
   SACH R2        ; Store Accumulator to R2
;-------------------------------------------------------------------------
   ZAC            ; Clear ACC
   LDP #(A11)     ; Load data page for A11
   LT A11         ; Load with A11, TREG = A11
   MAR*,AR4       ; ARP = 4, points to SEVENTH_SINA
   MPY *+         ; PREG = A11*SEVENTH_SINA
   APAC           ; ACC = PREG
   LDP #(R3)      ; Load data page for R3
   SACH R3        ; Store Accumulator to R3
;-------------------------------------------------------------------------
   ZAC            ; Clear ACC
   LDP #(A111)    ; Load data page for A111
   LT A111        ; Load with A11, TREG = A111
   MAR *,AR5      ; ARP = 5, points to ELEVENTH_SINA
   MPY *+         ; PREG = A111*ELEVENTH_SINA
   APAC           ; ACC = PREG
   LDP #(R4)      ; Load data page for R4
   SACH R4        ; Store Accumulator to R4
;-------------------------------------------------------------------------
```

```
    ZAC           ; Clear ACC
    LDP #(A1111)  ; Load data page for A1111
    LT A1111      ; Load with A11, TREG = A111
    MAR *,AR6     ; ARP = 6, points to THIRTEENTH_SINA
    MPY *+        ; PREG = A1111*THIRTEENTH_SINA
    APAC          ; ACC = PREG
    LDP #(R5)     ; Load data page for R5
    SACH R5       ; Store Accumulator to R5
;-------------------------------------------------------------------------
    ZAC           ; Clear ACC
    LDP #R1       ; Load data page for R1
    ADD R1        ; Add R1 to the accumulator
;-------------------------------------------------------------------------
    LDP #R2       ; Load data page for R2
    ADD R2        ; Add R2 to the accumulator
;-------------------------------------------------------------------------
    LDP #R3       ; Load data page for R3
    ADD R3        ; Add R3 to the accumulator
;-------------------------------------------------------------------------
   LDP #R4        ; Load data page for R4
   ADD R4         ; Add R4 to the accumulator
;-------------------------------------------------------------------------
   LDP #R5        ; Load data page for R5
   ADD R5         ; Add R5 to the accumulator
;-------------------------------------------------------------------------
   MAR *,AR7              ; ARP = 7, points to MODUL_BUF_OUTA
   SACL *+,0,AR0          ; Store the accumulator to Modul. buffer
   BANZ MODUL_BEGINA,*-   ; Go back to modulation calculation
   CLRC     OVM           ; Disable overflow mode

;----------------- End of Modulation Signal A -------------------------

;============= Calculation of the Modulation signal B =================


    LAR AR0,#127               ; AR0=127
    LAR AR2,#(FIRST_SINB)      ; AR2 = Fundamental Frequency
    LAR AR3,#(FIFTH_SINB)      ; AR3 = Fifth harmonic freq
    LAR AR4,#(SEVENTH_SINB)    ; AR4 = Seventh harmonic freq
    LAR AR5,#(ELEVENTH_SINB)   ; AR5 = Eleventh harmonic freq
    LAR AR6,#(THIRTEEN_SINB)   ; AR6 = Thirteenth harmonic freq
    LAR AR7,#(MODUL_BUF_OUTB) ; AR7 = Modulation buffer
    SOVM                            ; Set Overflow mode
    SETC     SXM                  ; Set sign-extension mode
;-------------------------------------------------------------------------
MODUL_BEGINB:
    MPYK #0h      ; Clear PREG
    LACC #0000h   ; Clear Acc.
    LDP #(A)      ; Load data page for A
    LT A          ; Load with A, TREG = A
    MAR *,AR2     ; ARP = 2, points to FIRST_SINB
    MPY *+        ; PREG = A*FIRST_SINB
    APAC          ; ACC = PREG
    LDP #(R1)     ; Load data page for R1
    SACH R1       ; Store Accumulator to R1

;-------------------------------------------------------------------------
   ZAC           ; Clear ACC
   LDP #(A1)     ; Load data page for A1
   LT  A1        ; Load with A1, TREG = A1
   MAR*,AR3      ; ARP = 3, points to FIFTH_SINB
   MPY *+        ; PREG = A1*FIFTH_SINB
   APAC          ; ACC = PREG
   LDP #(R2)     ; Load data page for R2
   SACH R2       ; Store Accumulator to R2
;-------------------------------------------------------------------------
    ZAC          ; Clear ACC
    LDP #(A11)   ; Load data page for A11
    LT A11       ; Load with A11, TREG = A11
    MAR*,AR4     ; ARP = 4, points to SEVENTH_SINB
    MPY *+       ; PREG = A11*SEVENTH_SINB
    APAC         ; ACC = PREG
    LDP #(R3)    ; Load data page for R3
    SACH R3      ; Store Accumulator to R3
;-------------------------------------------------------------------------
    ZAC          ; Clear ACC
    LDP #(A111)  ; Load data page for A111
    LT A111      ; Load with A11, TREG = A111
```

```
    MAR *,AR5    ; ARP = 5, points to ELEVENTH_SINB
    MPY *+       ; PREG = A111*ELEVENTH_SINB
    APAC         ; ACC = PREG
    LDP #(R4)    ; Load data page for R4
    SACH R4      ; Store Accumulator to R4
;---------------------------------------------------------------------
    ZAC          ; Clear ACC
    LDP #(A1111); Load data page for A1111
    LT A1111     ; Load with A11, TREG = A111
    MAR *,AR6    ; ARP = 6, points to THIRTEENTH_SINB
    MPY *+       ; PREG = A1111*THIRTEENTH_SINB
    APAC         ; ACC = PREG
    LDP #(R5)    ; Load data page for R5
    SACH R5      ; Store Accumulator to R5
;---------------------------------------------------------------------
    ZAC          ; Clear ACC
    LDP #R1      ; Load data page for R1
    ADD R1       ; Add R1 to the accumulator
;---------------------------------------------------------------------
    LDP #R2      ; Load data page for R2
    ADD R2       ; Add R2 to the accumulator
;---------------------------------------------------------------------
    LDP #R3      ; Load data page for R3
    ADD R3       ; Add R3 to the accumulator
;---------------------------------------------------------------------
   LDP #R4       ; Load data page for R4
   ADD R4        ; Add R4 to the accumulator
;---------------------------------------------------------------------
   LDP #R5       ; Load data page for R5
   ADD R5        ; Add R5 to the accumulator
;---------------------------------------------------------------------
   MAR *,AR7              ; ARP = 7, points to MODUL_BUF_OUTB
   SACL *+,0,AR0          ; Store the accumulator to Modul. buffer
   BANZ MODUL_BEGINB,*-   ; Go back to modulation calculation
   CLRC     OVM             ; Disable overflow mode

;----------------- End of Modulation Signal B -------------------------

;============= Calculation of the Modulation signal B ================

   LAR AR0,#127              ; AR0=127
   LAR AR2,#(FIRST_SINC)     ; AR2 = Fundamental Frequency
   LAR AR3,#(FIFTH_SINC)     ; AR3 = Fifth harmonic freq
   LAR AR4,#(SEVENTH_SINC)   ; AR4 = Seventh harmonic freq
   LAR AR5,#(ELEVENTH_SINC)  ; AR5 = Eleventh harmonic freq
   LAR AR6,#(THIRTEEN_SINC)  ; AR6 = Thirteenth harmonic freq
   LAR AR7,#(MODUL_BUF_OUTC) ; AR7 = Modulation buffer
   SOVM                           ; Set Overflow mode
   SETC     SXM                 ; Set sign-extension mode

;---------------------------------------------------------------------
MODUL_BEGINC:
    MPYK #0h     ; Clear PREG
    LACC #0000h  ; Clear acc.
    LDP #(A)     ; Load data page for A
    LT A         ; Load with A, TREG = A
    MAR *,AR2    ; ARP = 2, points to FIRST_SINC
    MPY *+       ; PREG = A*FIRST_SINC
    APAC         ; ACC = PREG
    LDP #(R1)    ; Load data page for R1
    SACH R1      ; Store Accumulator to R1

;---------------------------------------------------------------------
   ZAC           ; Clear ACC
   LDP #(A1)     ; Load data page for A1
   LT  A1        ; Load with A1, TREG = A1
   MAR*,AR3      ; ARP = 3, points to FIFTH_SINC
   MPY *+        ; PREG = A1*FIFTH_SINC
   APAC          ; ACC = PREG
   LDP #(R2)     ; Load data page for R2
   SACH R2       ; Store Accumulator to R2
;---------------------------------------------------------------------
    ZAC          ; Clear ACC
    LDP #(A11)   ; Load data page for A11
    LT A11       ; Load with A11, TREG = A11
    MAR*,AR4     ; ARP = 4, points to SEVENTH_SINC
    MPY *+       ; PREG = A11*SEVENTH_SINC
    APAC         ; ACC = PREG
```

```
    LDP #(R3)    ; Load data page for R3
    SACH R3      ; Store Accumulator to R3
;-------------------------------------------------------------------
    ZAC          ; Clear ACC
    LDP #(A111)  ; Load data page for A111
    LT A111      ; Load with A11, TREG = A111
    MAR *,AR5    ; ARP = 5, points to ELEVENTH_SINC
    MPY *+       ; PREG = A111*ELEVENTH_SINC
    APAC         ; ACC = PREG
    LDP #(R4)    ; Load data page for R4
    SACH R4      ; Store Accumulator to R4
;-------------------------------------------------------------------
    ZAC          ; Clear ACC
    LDP #(A1111); Load data page for A1111
    LT A1111     ; Load with A11, TREG = A111
    MAR *,AR6    ; ARP = 6, points to THIRTEENTH_SINC
    MPY *+       ; PREG = A1111*THIRTEENTH_SINC
    APAC         ; ACC = PREG
    LDP #(R5)    ; Load data page for R5
    SACH R5      ; Store Accumulator to R5
;-------------------------------------------------------------------
    ZAC          ; Clear ACC
    LDP #R1      ; Load data page for R1
    ADD R1       ; Add R1 to the accumulator
;-------------------------------------------------------------------
    LDP #R2      ; Load data page for R2
    ADD R2       ; Add R2 to the accumulator
;-------------------------------------------------------------------
    LDP #R3      ; Load data page for R3
    ADD R3       ; Add R3 to the accumulator
;-------------------------------------------------------------------
   LDP #R4       ; Load data page for R4
   ADD R4        ; Add R4 to the accumulator
;-------------------------------------------------------------------
   LDP #R5       ; Load data page for R5
   ADD R5        ; Add R5 to the accumulator
;-------------------------------------------------------------------
   MAR *,AR7               ; ARP = 7, points to MODUL_BUF_OUTC
   SACL *+,0,AR0           ; Store the accumulator to Modul. buffer
   BANZ MODUL_BEGINC,*-    ; Go back to modulation calculation
   CLRC     OVM               ; Disable overflow mode

;---------- End of Modulation Signal B -------------------------

modul_end      RET

COPY_BUFFER:
;------- Copy MODUL_BUFFER_OUTA to MODUL_BUF_A ------------

 LAR AR0,#127                ; AR0=127
 LAR AR2,#(MODUL_BUF_OUTA)   ; Load AR2 with Value in the MODUL_BUF_OUTA
 LAR   AR4,#(MODUL_BUFA)     ; Load AR4 with Value in the MODUL_BUFA

START_COPYA:
 MAR   *,AR2          ; ARP=2
 LACL  *+,AR4         ; Acc=data pointed by AR2, ARP=4
 SACL  *+,0,AR0       ; Store the accu.
 BANZ START_COPYA,*- ; Go back to the beginning of the loop.

;-------- End of copy MODUL_BUFA ----------------------------

;------- Copy MODUL_BUFFER_OUTB to MODUL_BUF_B --------------

 LAR   AR0,#127                ; AR0=127
 LAR   AR2,#(MODUL_BUF_OUTB)   ; Load AR2 with Value in the
 LAR   AR4,#(MODUL_BUFB)       ; Load AR4 with Value in the MODUL_BUFB

START_COPYB:
 MAR   *,AR2           ; ARP=AR2
 LACL  *+,AR4          ; Acc=data pointed by AR2
 SACL  *+,0,AR0              ; Store the accu.
 BANZ START_COPYB,*-  ; Go back to the beginning of the loop.

;-------- End of copy MODUL_BUFB ----------------------------

;------- Copy MODUL_BUFFER_OUTC to MODUL_BUF_C --------------

 LAR   AR0,#127                ; AR0=127
```

```
 LAR   AR2,#(MODUL_BUF_OUTC)    ; Load AR2 with Value in the
 LAR   AR4,#(MODUL_BUFC)        ; Load AR4 with Value in the MODUL_BUFC

START_COPYC:
 MAR   *,AR2           ; ARP=AR2
 LACL  *+,AR4          ; Acc=data pointed by AR2
 SACL  *+,0,AR0              ; Store the accu.
 BANZ START_COPYC,*- ; Go back to the beginning of the loop

;-------- End of copy MODUL_BUFC ----------------------------------
copy_end  RET
```

```
;-------------------------------------------------------------------

;-------------------------------------------------------------------
;       Oregon State University
;-------------------------------------------------------------------
;
;           File :          FFT.ASM
;
;           Orginator :     Abdurrahman Unsal
;
;           Date :          05.26.2000
;
;           Target System : C24x Evaluation Board
;
;-------------------------------------------------------------------
;
; Description:          The program perfoms a 128 point FFT.  The data samples
; are gathered from the ADC of the F240.
;
;-------------------------------------------------------------------
; Debug directives
;-------------------------------------------------------------------

 .def fft128r, PS_TWID_STRT,COS45K, COS45, I_BUF_OUT, B0_SADR, B1_SADR
 .def B2_SADR, TWID_TBL

N                       .set        128                     ;FFT length
COS45K      .set        05A82h      ;Constant for COS(45)
                        .bss        COS45,1     ;Value for COS(45)

;-------------------------------------------------------------------
; FFT Macro Declaration
;-------------------------------------------------------------------
;
;     MACRO 'ZEROI'     number of words/number of cycles: 10
;
;           ARP=2 FOR INPUT AND OUTPUT
;           AR2 -> QR,QI,QR+1,...
;           AR6 -> PR,PI,PR+1,...
;
;           CALCULATE Re[P+Q] AND Re[P-Q]
;           QR'=(PR-QR)/2
;           PR'=(PR+QR)/2
;           PI'=(PI+QI)/2
;           PI'=(PI-QI)/2
;
;  version 1.00     from Manfred Christ         update: 2. July 90
;
;*******************************************************************
ZEROI     .macro                          ;  AR6  AR2  ARP
          LACC     *,15,AR6  ; ACC := (1/2)(QR)            PR    QR     1
          ADD      *,15      ; ACC := (1/2)(PR+QR)         PR    QR     1
          SACH     *+,0,AR2  ; PR  := (1/2)(PR+QR)         PI    QR     2
          SUB      *,16      ; ACC := (1/2)(PR+QR)-(QR)    PI    QR     2
          SACH     *+        ; QR  := (1/2)(PR-QR)         PI    QI     2
          LACC     *,15,AR6  ; ACC := (1/2)(QI)            PI    QI     1
          ADD      *,15      ; ACC := (1/2)(PI+QI)         PI    QI     1
          SACH     *+,0,AR2  ; PI  := (1/2)(PI+QI)         PR+1  QI     2
          SUB      *,16      ; ACC := (1/2)(PI+QI)-(QI)    PR+1  QI     2
          SACH     *+        ; QI  := (1/2)(PI-QI)         PR+1  QR+1   2
              .endm

;*******************************************************************
;
;     MACRO 'PBY2I'     number of words/number of cycles: 12
;
;           ARP=2 on entry to macro
;           AR2 -> QR,QI,QR+1,...
;           AR6 -> PR,PI,PR+1,...
;
;           PR'=(PR+QI)/2       PI'=(PI-QR)/2
;           QR'=(PR-QI)/2       QI'=(PI+QR)/2
;
;  version 1.00     from Manfred Christ         update: 02. July 90
;
;*******************************************************************
PBY2I       .macro                  ;  AR6  AR2  ARP
         LACC     *+,15,AR5 ;                            PR    QI     5
```

```
        SACH    *,1,AR2    ; TMP=QR                      PR    QI     2
        LACC    *,15,AR6   ; ACC := QI/2                 PR    QI     1
        ADD     *,15       ; ACC := (PR+QI)/2            PR    QI     1
        SACH    *+,0,AR2   ; PR  := (PR+QI)/2            PI    QI     2
        SUB     *-,16      ; ACC := (PR-QI)/2            PI    QR     2
        SACH    *+,0,AR6   ; QR  := (PR-QI)/2            PI    QI     1
        LACC    *,15,AR5   ; ACC := (PI)/2               PI    QI     5
        SUB     *,15,AR6   ; ACC := (PI-QR)/2            PI    QI     1
        SACH    *+,0,AR5   ; PI  := (PI-QR)/2            PR+1  QI     5
        ADD     *,16,AR2   ; ACC := (PI+QR)/2            PR+1  QI     2
        SACH    *+         ; QI  := (PI+QR)/2            PR+1  QI+1   2
           .endm

;*********************************************************************;
;     MACRO 'PBY4I'    number of words/number of cycles: 16
;
;           T=SIN(45)=COS(45)=W45
;
;           PR'= PR + (W*QI + W*QR) = PR + W * QI + W * QR     (<- AR6)
;           QR'= PR - (W*QI + W*QR) = PR - W * QI - W * QR     (<- AR2)
;            PI'= PI + (W*QI - W*QR) = PI + W * QI - W * QR     (<- AR6+1)
;           QI'= PI - (W*QI - W*QR) = PI - W * QI + W * QR     (<- AR6+2)
;
;
;  version 1.00     from Manfred Christ          update: 02. July 90
;
;*********************************************************************
PBY4I       .macro          ; TREG=                  AR5 PREG AR6 AR2 ARP
        MPY       *+,AR5    ; PREG= W*QR/2            -  W*QR/2 PR   QI   5
        SPH       *,AR6     ; TMP = W*QR/2     -W*QR/2 W*QR/2 PR   QI   1
        LACC      *,15,AR2  ; ACC = PR/2       -W*QR/2 W*QR/2 PR   QI   2
        MPYS      *-        ; ACC = (PR-W*QR)/2 -QR/2 W*QI/2 PR   QR   2
        SPAC          ; ACC = (PR-W*QI-W*QR)/2W*QR/2 W*QI/2 PR   QR   2
        SACH  ,AR6 ; QR  = (PR-W*QI-W*QR)/2  W*QR/2 W*QI/2 PR   QI   1
        SUB *,16    ; ACC = (-PR-W*QI-W*QR)/2 W*QR/2 W*QI/2 PR   QI   1
        NEG         ; ACC =  (PR+W*QI+W*QR)/2 W*QR/2 W*QI/2 PR   QI   1
        SACH   *+   ; QR  =  (PR+W*QI+W*QR)/2 W*QR/2 W*QI/2 PI   QI   1

        LACC *,15,AR5 ; ACC = (PI)/2          W*QR/2 W*QI/2 PI   QI   5
        SPAC        ; ACC = (PI-W*QI)/2       W*QR/2   -    PI   QI   5
        ADD,16,AR2  ; ACC = (PI-W*QI+W*QR)/2     -     -    PI   QI   2
        SACH *+,0,AR6 ; QI = (PI-W*QI+W*QR)/2    -     -    PI   QR1  1
        SUB *,16    ; ACCU= (-PI-W*QI+W*QR)/2    -     -    PI   QR1  1
        NEG         ; ACCU=  (PI+W*QI-W*QR)/2    -     -    PI   QR1  1
        SACH *+,0,AR2 ; PI  =  (PI+W*QI-W*QR)/2 -     -    PR1  QR1  2
           .endm

;*********************************************************************
;
;     MACRO 'P3BY4I'   number of words/number of cycles: 16
;
;     version 1.00     from: Manfred Christ   update: 02. July 90
;
;           ENTRANCE IN THE MACRO: ARP=AR2
;                                          AR6->PR,PI
;                                          AR2->QR,QI
;                                          TREG=W=COS(45)=SIN(45)
;
;           PR'= PR + (W*QI - W*QR) = PR + W * QI - W * QR     (<- AR6)
;           QR'= PR - (W*QI - W*QR) = PR - W * QI + W * QR     (<- AR2)
;           PI'= PI - (W*QI + W*QR) = PI - W * QI - W * QR     (<- AR6+1)
;           QI'= PI + (W*QI + W*QR) = PI + W * QI + W * QR     (<- AR6+2)
;
;           EXIT OF THE MACRO:    ARP=AR2
;                                          AR6->PR+1,PI+1
;                                          AR2->QR+1,QI+1
;
;*********************************************************************
P3BY4I .macro p,m ; TREG= W                 AR5    PREG    AR6   AR2 ARP
                  ;                         ------ ------ ---   --- ---
      MPY    *+,AR5 ; PREG= W*QR/2              -   W*QR/2 PR    QI   5
      SPH    *,AR6  ; TMP = W*QR/2           W*QR/2 W*QR/2 PR    QI   1
      LACC   *,15,AR2 ; ACC = PR/2           W*QR/2 W*QR/2 PR    QI   2
      MPYA   *-        ACC = (PR+W*QR)/2     W*QR/2 W*QI/2 PR    QR   2
      SPAC             ACC = (PR-W*QI+W*QR)/2 W*QR/2 W*QI/2 PR   QR   2
      SACH   *+,0,AR6;QR' = (PR-W*QI+W*QR)/2  W*QR/2 W*QI/2 PR   QI   1
      SUB    *,16    ; ACC = (-PR-W*QI+W*QR)/2 W*QR/2 W*QI/2 PR   QI   1
      NEG            ; ACC =  (PR+W*QI-W*QR)/2 W*QR/2 W*QI/2 PR   QI   1
```

```
      SACH   *+      ; PR' =  (PR+W*QI-W*QR)/2 W*QR/2 W*QI/2 PI   QI   1
      LACC   *,15,AR5 ;ACC = (PI)/2             W*QR/2 W*QI/2 PI   QI   5
      APAC           ;ACC = (PI+W*QI)/2         W*QR/2   -    PI   QI   5
      ADD    *,16,AR2 ;ACC = (PI+W*QI+W*QR)/2     -      -    PI   QI   2
      SACH   *:m:+,0,AR6 ;QI' =(PI+W*QI+W*QR)/2 -        -    PI   QR5  1
      SUB    *,16      ; ACCU= (-PI+W*QI+W*QR)/2 -        -    PI   QR5  1
      NEG            ; ACCU= (PI-W*QI-W*QR)/2    -       -    PI   QR5  1
      SACH   *:m:+,0,AR:p: ; PI' = (PI-W*QI-W*QR)/2  -      PR5  QR5  7
      .endm

;***********************************************************************
;
;  MACRO:  'BFLY'       general butterfly radix 2 for 320C2xx/5x
;
;  version 1.00     from Manfred Christ          update: 02. July 90
;
;  THE MACRO 'BFLY' REQUIRES 18 WORDS AND 18 INSTRUCTIONS
;
;  Definition: ARP -> AR2  (input)  ARP -> AR2                (output)
;
;  Definition: AR6 -> QR   (input)  AR6 -> QR+1      (output)
;  Definition: AR2 -> PR   (input)  AR2 -> PR+1      (output)
;  Definition: AR3 -> Cxxx (input)  AR3 -> Cxxx+1(output)--> WR=cosine
;  Definition: AR4 -> Sxxx (input)  AR4 -> Sxxx+1(output)--> WI=sine
;  Definition: AR5 -> temporary variable (unchanged)
;
;  uses index register
;
;           PR' = (PR+(QR*WR+QI*WI))/2        WR=COS(W)    WI=SIN(W)
;           PI' = (PI+(QI*WR-QR*WI))/2
;           QR' = (PR-(QR*WR+QI*WI))/2
;           QI' = (PI-(QI*WR-QR*WI))/2
;
;  Note: AR0 determines Twiddle Pointers (AR3 & AR4) step increments
;
;***********************************************************************
BFLY .macro p
;                                    (contents of register after exec.)
;                                                TREG AR6  AR2  AR3 AR4 ARP
;                                                --- ---  ---  --- --- ---
  LT    *+,AR3     ;TREG:= QR                     QR PR   QI    C   S   3
  MPY   *,AR2      ;PREG:= QR*WR/2                QR PR   QI    C   S   2
  LTP   *-,AR4     ;ACC := QR*WR/2                QI PR   QR    C   S   4
  MPY   *,AR3      ;PREG:= QI*WI/2                QI PR   QR    C   S   3
  MPYA  *0+,AR2    ;ACC := (QR*WR+QI*WI)/2        QR PR   QR    C+n S   2
                   ;PREG:= QI*WR
  LT    *,AR5      ;TREG = QR                     QR PR   QR    C+n S   5
  SACH  *,1,AR6    ;TEMP:= (QR*WR+QI*WI)          QR PR   QR    C+n S   1

  ADD   *,15       ;ACC := (PR+(QR*WR+QI*WI))/2   QR PR   QR    C+n S   1
  SACH  *+,0,AR5 ;PR   := (PR+(QR*WR+QI*WI))/2    QR PI   QR    C+n S   5
  SUB   *,16,AR2 ;ACC := (PR-(QR*WR+QI*WI))/2     QR PI   QR    C+n S   2
  SACH  *+,0,AR6 ;QR  := (PR-(QR*WR+QI*WI))/2     QR PI   QI    C+n S   1

  LAC        *,15,AR4  ;ACC := PI  /PREG=QI*WR    QI PI   QI    C+n S   4
  MPYS  *0+,AR2        ;PREG:= QR*WI/2             QI PI   QI    C+n S+n 2
                       ;ACC := (PI-QI*WR)/2
  APAC             ;ACC := (PI-(QI*WR-QR*WI))/2     QI PI   QI    C+n S+n 2
  SACH  *+,0,AR6  ;QI  := (PI-(QI*WR-QR*WI))/2      QI PI    QR+1  C+n S+n 1
  NEG             ;ACC :=(-PI+(QI*WR-QR*WI))/2      QI PI    QR+1  C+n S+n 1
  ADD         *,16     ; ACC := (PI+(QI*WR-QR*WI))/2 QI PI    QR+1  C+n S+n 1
  SACH  *+,0,AR:p:    ;PI := (PI+(QI*WR-QR*WI))/2   QI PR+1  QR+1  C+n S+n 2
  .endm

;***********************************************************************
; MACRO 'COMBO'
;
; R1  := [(R1+R2)+(R3+R4)]/4          INPUT               OUTPUT
; R2  := [(R1-R2)+(I3-I4)]/4  --------------------  ------------------
; R3  := [(R1+R2)-(R3+R4)]/4          AR0 =  7
; R4  := [(R1-R2)-(I3-I4)]/4          AR6 -> R1,I1        AR6 - > R5,I5
; I1  := [(I1+I2)+(I3+I4)]/4          AR2 -> R2,I2        AR2 - > R1,I6
; I2  := [(I1-I2)-(R3-R4)]/4 ARP -> AR3 -> R3,I3 ARP - > AR3 - > R7,I7
; I3  := [(I1+I2)-(I3+I4)]/4          AR4 -> R4,I4        AR4 - > R8,I8
; I4  := [(I1-I2)+(R3-R4)]/4
;
;***********************************************************************
;
```

```
;                                                          ARP AR6 AR2 AR3 AR4 AR5
COMBO       .macro                  ;                      --- --- --- --- --- ---
      LACC    *,14,AR4     ; ACC := (R3)/4                  4  R1  R2  R3  R4  T1
      SUB     *,14,AR5     ; ACC := (R3+R4)/4               5  R1  R2  R3  R4  T1
      SACH    *+,1,AR4     ; T1   = (R3-R4)/2               4  R1  R2  I3  R4  T2

      ADD     *+,15,AR5    ; ACC := (R3+R4)/4               5  R1  R2  R3  I4  T2
      SACH    *,1,AR2      ; T2   = (R3+R4)/2               2  R1  R2  R3  I4  T2

      ADD     *,14,AR6     ; ACC := (R2+R3+R4)/4            1  R1  R2  R3  I4  T2
      ADD     *,14         ; ACC := (R1+R2+R3+R4)/4         1  R1  R2  R3  I4  T2
      SACH    *+,0,AR5     ; R1   := (R1+R2+R3+R4)/4        5  I1  R2  R3  I4  T2
      SUB     *,16,AR3    ; ACC := (R1+R2-(R3+R4))/4        3  I1  R2  R3  I4  T2
      SACH    *+,0,AR5    ; R3   := (R1+R2-(R3+R4))/4       5  I1  R2  I3  I4  T2

      ADD     *,15,AR2     ; ACC := (R1+R2)/4               2  I1  R2  I3  I4  T2
      SUB     *,15,AR3     ; ACC := (R1-R2)/4               3  I1  R2  I3  I4  T2
      ADD     *,14,AR4     ; ACC := ((R1-R2)+(I3))/4        4  I1  R2  I3  I4  T2
      SUB     *,14,AR2 ; ACC := ((R1-R2)+(I3-I4))/4         2  I1  R2  I3  I4  T2
      SACH    *+,0,AR4 ; R2  := ((R1-R2)+(I3-I4))/4         4  I1  I2  I3  I4  T2
      ADD     *-,15,AR3; ACC := ((R1-R2)+ I3+I4 )/4         3  I1  I2  I3  R4  T2
      SUB     *,15,AR4 ; ACC := ((R1-R2)-(I3-I4))/4         4  I1  I2  I3  R4  T2
      SACH    *+,0,AR6 ; R4  := ((R1-R2)-(I3-I4))/4         1  I1  I2  I3  I4  T2

      LACC    *,14,AR2     ; ACC := (I1)/4                  2  I1  I2  I3  I4  T2
      SUB     *,14,AR5     ; ACC := (I1-I2)/4               5  I1  I2  I3  I4  T2
      SACH    *,1,AR2      ; T2  := (I1-I2)/2               2  I1  I2  I3  I4  T2
      ADD     *,15,AR3     ; ACC := ((I1+I2))/4             4  I1  I2  I3  I4  T2
      ADD     *,14,AR4     ; ACC := ((I1+I2)+(I3))/4        4  I1  I2  I3  I4  T2
      ADD     *,14,AR6  ACC := ((I1+I2)+(I3+I4))/4          1  I1  I2  I3  I4  T2
      SACH    *0+,0,AR3 ;I1  := ((I1+I2)+(I3+I4))/4         3  R5  I2  I3  I4  T2
      SUB     *,15,AR4  ;ACC := ((I1+I2)-(I3+I4))/4         4  R5  I2  I3  I4  T2
      SUB     *,15,AR3  ;ACC := ((I1+I2)-(I3+I4))/4         3  R5  I2  I3  I4  T2
      SACH    *0+,0,AR5 ;I3  := ((I1+I2)-(I3+I4))/4         5  R5  I2  R7  I4  T2

      LACC    *-,15        ; ACC := (I1-I2)/4               5  R5  I2  R7  I4  T1
      SUB     *,15,AR2  ;ACC := ((I1-I2)-(R3-R4))/4         2  R5  I2  R7  I4  T1
      SACH    *0+,0,AR5 ; I2 := ((I1-I2)-(R3-R4))/4         5  R5  R1  R7  I4  T1
      ADD     *,16,AR4  ; ACC :=((I1-I2)+(R3-R4))/4         4  R5  R1  R7  I4  T1
      SACH    *0+,0,AR7; I4 := ((I1-I2)+(R3-R4))/4          7  R5  R1  R7  R8  T1
            .endm

;=================================================================================
; M A I N   C O D E  - starts here
       .global REPEAT
;=================================================================================
; Fetch 128 Data sample points from Input Buffer
;=================================================================================

fft128r:

FTCH_DATA
  LAR AR0, #128     ;AR0= 28; # of Samples for Bit Reversed Addressing
  LAR AR6, #B0_SADR ;AR6 = Start Address of B0; Data Buffer
  LAR       AR2, #(I_BUF_OUT)       ;AR2 = Value Stored in BUFFER
  LAR       AR3, #(B1_SADR+128)     ;AR3 = B1_SADR + 128
  LAR       AR7, #127               ;AR7 = 128 - 1
  MAR  *, AR2                                ;ARP = AR2
  B FTCH_LP
;--------------------------------------------------------------------------------
;  The following code sets the imaginary part of input to zero
;--------------------------------------------------------------------------------

ZER0_IMAG:
  LACC #0
  LAR AR0, #2
  LAR AR6, #B0_SADR+1
  LAR AR7, #127
  MAR *,AR6
ZERO_LOOP:
  SACL *0+,0,AR7
  BANZ ZERO_LOOP,*-,AR6


;=================================================================================
; Stages 1 & 2 - using the Radix 4 COMBO Macro
;=================================================================================
                       MAR         *, AR3
```

```
                        LAR         AR0, #7h    ;Increment for Data pointers
                        LAR         AR6, #(B0_SADR)
                        LAR         AR2, #(B0_SADR+2)
                        LAR         AR3, #(B0_SADR+4)
                        LAR         AR4, #(B0_SADR+6)
                        LAR         AR5, #B2_SADR    ;Gen purp reg @ 60h
                        LAR         AR7, #(N/4-1)    ;Loop 32 times
STAGE1_2_LP:
                        COMBO
                        BANZ    STAGE1_2_LP,*-,AR3

;=====================================================================
; Stage  3 - using ZEROI, PBY4I, PBY2I, P3BY4I Macros
;=====================================================================
STAGE3:
                        MAR         *, AR2                  ;ARP-->AR2
                        LAR         AR0, #9h                ;
                        LAR         AR6, #(B0_SADR)         ;-->P
                        LAR         AR2, #(B0_SADR+8)       ;-->Q
                        LAR         AR5, #B2_SADR           ;Gen purp reg @ 60h
                        LAR         AR7, #(N/8-1)           ;Loop counter (32 times)
                        LT          COS45

STAGE3_LP:
                        ZEROI
                        PBY4I
                        PBY2I
                        P3BY4I 7,0              ;-->AR7 at end, use *0+ modify.
                        BANZ    STAGE3_LP,*-,AR2

;=====================================================================
; Stage  4 - using ZEROI, PBY4I, PBY2I, P3BY4I, BFLY Macros
;=====================================================================
STAGE4:
                        MAR         *, AR2                              ;ARP-->AR2
                        LAR         AR0, #16                            ;Used to inc Twiddle
pointers
                        LAR         AR6, #(B0_SADR)                     ;-->P
                        LAR         AR2, #(B0_SADR+16)      ;-->Q
                        LAR         AR5, #B2_SADR                       ;Gen purp reg @ 60h
                        LAR         AR7, #(N/16-1)                      ;Loop counter (8 times)
STAGE4_LP:
                        ZEROI
                        LAR         AR3, #(TWID_TBL+8+N/4)
                        LAR         AR4, #(TWID_TBL+8)
STG4_B1     BFLY 2
                        LT          COS45
                        PBY4I
STG4_B2     BFLY 2
                        PBY2I
STG4_B3     BFLY 2

                        LT          COS45
                        P3BY4I 2,                   ;-->AR2 at end
STG4_B4     BFLY 6
                        ADRK        #16
                        MAR         *, AR2
                        ADRK        #16
                        MAR         *, AR7
                        BANZ        STAGE4_LP,*-,AR2
;=====================================================================
; Stage  5 - using BUTTFLYI Macro
;=====================================================================
STAGE5:
                        MAR         *, AR2              ;-->AR2
                        LAR         AR0, #4             ;Used to Inc Twiddle pointers
                        LAR         AR6, #(B0_SADR)     ;-->P (0-->15)
                        LAR         AR2, #(B0_SADR+32)  ;-->Q (16-->31)
                        LAR         AR5, #B2_SADR       ;Gen purp reg @ 60h
;---------------------------------------------------------------------
STG5_BLK1   LAR         AR3, #(TWID_TBL+N/4)            ;COS(angle)
                        LAR         AR4, #(TWID_TBL)    ;COS(angle+pi/4)
                        LAR         AR7, #(N/8-1)       ;Loop counter (16 times)
STAGE51_LP:
                        BFLY 7
                        BANZ        STAGE51_LP,*-,AR2
;---------------------------------------------------------------------
STG5_BLK2:
```

```
                        LAR         AR6, #(B0_SADR+64)      ;-->P (32-->47)
                        LAR         AR2, #(B0_SADR+96)      ;-->Q (48-->63)
                        LAR         AR3, #(TWID_TBL+N/4)    ;COS(angle)
                        LAR         AR4, #(TWID_TBL)        ;COS(angle+pi/4)
                        LAR         AR7, #(N/8-1)           ;Loop counter (16 times)
STAGE52_LP:
                        BFLY 7
                        BANZ        STAGE52_LP,*-,AR2
;---------------------------------------------------------------------
STG5_BLK3:
                        LAR         AR6, #(B0_SADR+128)     ;-->P (64-->79)
                        LAR         AR2, #(B0_SADR+160)     ;-->Q (80-->95)
                        LAR         AR3, #(TWID_TBL+N/4)    ;COS(angle)
                        LAR         AR4, #(TWID_TBL)        ;COS(angle+pi/4)
                        LAR         AR7, #(N/8-1)           ;Loop counter (16 times)
STAGE53_LP:
                        BFLY 7
                        BANZ        STAGE53_LP,*-,AR2
;---------------------------------------------------------------------
STG5_BLK4:
                        LAR         AR6, #(B0_SADR+192)     ;-->P (96-->111)
                        LAR         AR2, #(B0_SADR+224)     ;-->Q (112-->127)
                        LAR         AR3, #(TWID_TBL+N/4)    ;COS(angle)
                        LAR         AR4, #(TWID_TBL)        ;COS(angle+pi/4)
                        LAR         AR7, #(N/8-1)           ;Loop counter (16 times)
STAGE54_LP:
                        BFLY 7
                        BANZ        STAGE54_LP,*-,AR2
;=====================================================================
; Stage  6 - Using BFLY Macro
;=====================================================================
STAGE6:
                        MAR         *, AR2                  ;-->AR2
                        LAR         AR0, #2     ;used to Inc Twiddle pointers
                        LAR         AR6, #(B0_SADR)         ;-->P (0-->31)
                        LAR         AR2, #(B0_SADR+64)      ;-->Q (32-->63)
                        LAR         AR5, #B2_SADR           ;Gen purp reg @ 60h
;---------------------------------------------------------------------
STG6_BLK1   LAR         AR3, #(TWID_TBL+N/4)    ;COS(angle)
                        LAR         AR4, #(TWID_TBL)        ;COS(angle+pi/4)
                        LAR         AR7, #(N/4-1)           ;Loop counter (32 times)
STAGE61_LP:
                        BFLY 7
                        BANZ        STAGE61_LP,*-,AR2
;---------------------------------------------------------------------
STG6_BLK2:
                        LAR         AR6, #(B0_SADR+128)     ;-->P (64-->95)
                        LAR         AR2, #(B0_SADR+192)     ;-->Q (96-->127)
                        LAR         AR3, #(TWID_TBL+N/4)    ;COS(angle)
                        LAR         AR4, #(TWID_TBL)        ;COS(angle+pi/4)
                        LAR         AR7, #(N/4-1)           ;Loop counter (32 times)
STAGE62_LP:
                        BFLY 7
                        BANZ        STAGE62_LP,*-,AR2
;=====================================================================
; Stage  7 - Using BFLY Macro
;=====================================================================
STAGE7:
                        MAR         *, AR2                  ;-->AR2
                        LAR         AR0, #1      ;Used to Inc Twiddle pointers
                        LAR         AR6, #(B0_SADR)         ;-->P (0-->63)
                        LAR         AR2, #(B0_SADR+128)     ;-->Q (64-->128)
                        LAR         AR5, #B2_SADR           ;Gen purp reg @ 60h
                        LAR         AR3, #(TWID_TBL+N/4)
                        LAR         AR4, #(TWID_TBL)
                        LAR         AR7, #(N/2-1)           ;Loop counter (64 times)
STG_7_LP    BFLY 7
                        BANZ    STG_7_LP,*-,AR2
;=====================================================================
; Convert Real/Img into Magnitude ( Mag = sqr(Xr[n]) + sqr(Xi[n]) )
;=====================================================================
                        LAR         AR6, #B0_SADR
                        LAR         AR2, #B1_SADR
                        LARK        AR3, #20                ;loop 128 times
                        LAR         AR4, #(B1_SADR+128)
                        MAR         *, AR6
MAG_LP              ZAC
                        MPYK        0
```

```
                    SQRA        *+                      ;Xr[n]**2
                    SQRA        *+,AR2                  ;Xi[n]**2
                    APAC
                    SACH        *+,0,AR4
                    SACH        *+,0,AR3    ;XOUT(I) = X(I)**2 + Y(I)**2
                    BANZ        MAG_LP,*-,AR6
                    RET
;=======================================================================
; TWIDDLES FOR N=128 FFT (96 entries)
;=======================================================================
PS_TWID_STRT:
     .word 00000h     ;    0.000 ø
     .word 00648h     ;    2.812 ø
     .word 00c8ch     ;    5.625 ø
     .word 012c8h     ;    8.438 ø
     .word 018f9h     ;   11.250 ø
     .word 01f1ah     ;   14.062 ø
     .word 02528h     ;   16.875 ø
     .word 02b1fh     ;   19.688 ø
     .word 030fch     ;   22.500 ø
     .word 036bah     ;   25.312 ø
     .word 03c57h     ;   28.125 ø
     .word 041ceh     ;   30.938 ø
     .word 0471dh     ;   33.750 ø
     .word 04c40h     ;   36.562 ø
     .word 05134h     ;   39.375 ø
     .word 055f6h     ;   42.188 ø
     .word 05a82h     ;   45.000 ø
     .word 05ed7h     ;   47.812 ø
     .word 062f2h     ;   50.625 ø
     .word 066d0h     ;   53.438 ø
     .word 06a6eh     ;   56.250 ø
     .word 06dcah     ;   59.062 ø
     .word 070e3h     ;   61.875 ø
     .word 073b6h     ;   64.688 ø
     .word 07642h     ;   67.500 ø
     .word 07885h     ;   70.312 ø
     .word 07a7dh     ;   73.125 ø
     .word 07c2ah     ;   75.938 ø
     .word 07d8ah     ;   78.750 ø
     .word 07e9dh     ;   81.562 ø
     .word 07f62h     ;   84.375 ø
     .word 07fd9h     ;   87.188 ø
     .word 07fffh     ;   90.000 ø
     .word 07fd9h     ;   92.812 ø
     .word 07f62h     ;   95.625 ø
     .word 07e9dh     ;   98.438 ø
     .word 07d8ah     ;  101.250 ø
     .word 07c2ah     ;  104.062 ø
     .word 07a7dh     ;  106.875 ø
     .word 07885h     ;  109.688 ø
     .word 07642h     ;  112.500 ø
     .word 073b6h     ;  115.312 ø
     .word 070e3h     ;  118.125 ø
     .word 06dcah     ;  120.938 ø
     .word 06a6eh     ;  123.750 ø
     .word 066d0h     ;  126.562 ø
     .word 062f2h     ;  129.375 ø
     .word 05ed7h     ;  132.188 ø
     .word 05a82h     ;  135.000 ø
     .word 055f6h     ;  137.812 ø
     .word 05134h     ;  140.625 ø
     .word 04c40h     ;  143.438 ø
     .word 0471dh     ;  146.250 ø
     .word 041ceh     ;  149.062 ø
     .word 03c57h     ;  151.875 ø
     .word 036bah     ;  154.688 ø
     .word 030fch     ;  157.500 ø
     .word 02b1fh     ;  160.312 ø
     .word 02528h     ;  163.125 ø
     .word 01f1ah     ;  165.938 ø
     .word 018f9h     ;  168.750 ø
     .word 012c8h     ;  171.562 ø
     .word 00c8ch     ;  174.375 ø
     .word 00648h     ;  177.188 ø
     .word 00000h     ;  180.000 ø
     .word 0f9b8h     ;  182.812 ø
     .word 0f374h     ;  185.625 ø
     .word 0ed38h     ;  188.438 ø
     .word 0e707h     ;  191.250 ø
     .word 0e0e6h     ;  194.062 ø
     .word 0dad8h     ;  196.875 ø
     .word 0d4e1h     ;  199.688 ø
     .word 0cf04h     ;  202.500 ø
     .word 0c946h     ;  205.312 ø
     .word 0c3a9h     ;  208.125 ø
     .word 0be32h     ;  210.938 ø
     .word 0b8e3h     ;  213.750 ø
     .word 0b3c0h     ;  216.562 ø
     .word 0aecch     ;  219.375 ø
     .word 0aa0ah     ;  222.188 ø
     .word 0a57eh     ;  225.000 ø
     .word 0a129h     ;  227.812 ø
     .word 09d0eh     ;  230.625 ø
     .word 09930h     ;  233.438 ø
     .word 09592h     ;  236.250 ø
     .word 09236h     ;  239.062 ø
     .word 08f1dh     ;  241.875 ø
     .word 08c4ah     ;  244.688 ø
     .word 089beh     ;  247.500 ø
     .word 0877bh     ;  250.312 ø
     .word 08583h     ;  253.125 ø
     .word 083d6h     ;  255.938 ø
     .word 08276h     ;  258.750 ø
     .word 08163h     ;  261.562 ø
     .word 0809eh     ;  264.375 ø
PS_TWID_END:
     .word 08027h     ;  267.188 ø
```

```
;-----------------------------------------------------------------------;
;           Oregon State University
;-----------------------------------------------------------------------
;
;           File :  TASK1.INC
;
;           Author :      Abdurrahman Unsal
;
;           Date :           05.26.2000
;
;           Target System : C24x Evaluation Board
;
;-----------------------------------------------------------------------
;-----------------------------------------------------------------------
; TASK 1
;
; Sampling of input current

adoffset: .equ 8140h
read_vt:
  LDP #_ADCTRL1
  SPLK #9801h,_ADCTRL1  ; Convert ch 0 & 8

  CLRC  SXM             ; Disable sign extension
vt_if1:                 ; repeat
  BIT  _ADCTRL1,8
  BBNZ vt_if1           ; until end of conversion

  LDP #_ADCFIFO1        ; Load data page for ch. 1
  LACL _ADCFIFO1        ; Load ch. 1
  SFR                   ; Shift accumulator to right by one bit

;-------------------------------------------------------------
;               BUFFERING THE INPUT                           |
;-------------------------------------------------------------

   LDP     #(I_BUF_IN_PTR)     ; Load data page for I_BUF_IN_PTR
   LAR     AR3,I_BUF_IN_PTR    ; Load AR3 with Value in the I_BUF_IN_PTR
   MAR   *,AR3                 ; ARP = AR3
   SACL  *+,0                          ; Store accu.
   LAR   AR0,#(I_BUF_IN+255)   ; Load AR0 to check the end of the buffer
   CMPR  2                     ; Compare the content of AR0 and AR3
   BCND  jump,NTC              ; Branch to the jump if AR3 ~= end of
                               ; buffer
   LAR   AR3,#(I_BUF_IN)       ;Load AR3 with I_BUF_IN jump
   LDP   #(I_BUF_IN_PTR)       ; Load data page for I_BUF_IN_PTR
   SAR   AR3,I_BUF_IN_PTR      ; Save the auxiliary register

;================= Reading Current Buffer ========================

   LDP   #(I_BUF_OUT_PTR)      ; Load data page for I_BUF_OUT_PTR
   LAR   AR2,I_BUF_OUT_PTR     ; Load AR2
   LDP   #(I_BUF_WR_PTR)       ; Load data page for I_BUF_WR_PTR
   LAR   AR4,I_BUF_WR_PTR      ; Load AR4 with Value in the I_BUF_WR_PTR
   MAR   *,AR2                 ; ARP=AR2
   LACL  *+                    ; Acc=data pointed by AR2
   LDP #OFF_SET                ; Load data page for OFF_SET
   SUB OFF_SET                 ; OFF_SET the accumulator

   CMPR  2     ; Compare the content of AR0 and AR2
   BCND  jump1,NTC          ; Branch to the jump1 if AR2 ~= end of
                            ; buffer
   LAR   AR2,#(I_BUF_IN)          ; Load AR2 with I_BUF_IN

jump1 LDP   #(I_BUF_OUT_PTR)      ; Load data page for I_BUF_OUT_PTR
      SAR   AR2,I_BUF_OUT_PTR     ; Save the auxiliary register
      MAR   *,AR4                 ; ARP=AR4
      SACL  *+,0                ; Store the accu.
      LAR   AR0,#(I_BUF_OUT+127) ; Load AR0
      CMPR  2                     ; Compare the content of AR0 and AR3
      BCND  jump2,NTC             ; Branch to the jump2 if AR4 ~= end of
                                  ; buffer
      LAR   AR4,#(I_BUF_OUT)      ; Load AR4 with I_BUF_OUT

jump2 LDP   #(I_BUF_WR_PTR)         ; Load data page for I_BUF_WR_PTR
      SAR   AR4,I_BUF_WR_PTR      ; Save the auxiliary register

;==================  Voltage Sampling with ADC2 ======================
```

```
      CLRC  SXM         ; Disable sign extension
      ZAC               ; Zero acc
      LDP #_ADCFIFO2    ; Load data page for Ch.2
      LACL _ADCFIFO2    ; Load ch. 2
      LDP #OFF_SET      ; Load data page for OFF_SET
      SUB OFF_SET       ; OFF_SET the accumulator

      LDP #vin          ; Load data page for input voltage sample
      SACL vin          ; Save input voltage sample

      fir8 vin, vout,fir_dl, b_fir ; FILTER the input voltage sample
                                   ; using 16-tap FIR Filter

jumper CALL ZERO_CROSS             ; Call for zero crossing


;=====================================================================
;
;             PWM COMPARATORS ARE LOADED DATA FROM MODUL_BUF
;
;=====================================================================
      LDP   #(PTR_A)        ; Load data page for PTR_A, pointer for PWM1
      LAR   AR3,PTR_A       ; AR3= Value in the PTR_A
      MAR   *,AR3           ; AR=AR3
      LACL  *+                    ; Load the ACCL
      LDPK _SCMPR1          ; Load data page for Simple compare unit1
      SACL _SCMPR1          ; Store the accumulator
      LAR   AR0,#(MODUL_BUFA+127); Load AR0
      CMPR  2                     ; Compare the content of AR0 and AR3
      BCND  jump6,NTC             ; Branch conditionally
      LAR   AR3,#(MODUL_BUFA)     ; AR3=Start address of MODUL_BUFA
jump6 LDP   #(PTR_A)             ; Load data page for PTR_A
      SAR   AR3,PTR_A             ; Save the auxiliary register

;------------------ Read data for phase B -------------------------

      LDP   #(PTR_B)              ; Load data page for PTR_B, PWM2
      LAR   AR2,PTR_B             ; AR2=PTR_B
      MAR   *,AR2                 ; AR=AR2
      LACL  *+                    ; Load the ACCL
      LDPK _SCMPR2     ; Load data page for Simple compare unit2
      SACL _SCMPR2     ; Store the accumulator into Simple compare unit2
      LAR   AR0,#(MODUL_BUFB+127); Load AR0
      CMPR  2                     ; Compare the content of AR0 and AR2
      BCND  jump7,NTC             ; Branch conditionally
      LAR   AR2,#(MODUL_BUFB)     ; AR=Start address of MODUL_BUFB
jump7 LDP   #(PTR_B)             ; Load data page pointer for PTR_B
      SAR   AR2,PTR_B             ; Save the auxiliary register

;------------------ Read data for phase C  ---------------------------

      LDP   #(PTR_C)              ; Load data page for PTR_C, PWM3
      LAR   AR2,PTR_C             ; AR2=PTR_C
      MAR   *,AR2                 ; AR=AR2
      LACL  *+                    ; Load the ACCL
      LDPK _SCMPR3     ; Load data page for Simple compare unit3
      SACL _SCMPR3     ; Store the accumulator into Simple compare unit3
      LAR   AR0,#(MODUL_BUFC+127); Load AR0
      CMPR  2                     ; Compare the content of AR0 and AR4
      BCND  jump8,NTC             ; Branch conditionally
      LAR   AR2,#(MODUL_BUFC)     ; AR=Start address of MODUL_BUFC
jump8 LDP   #(PTR_C)             ; Load data page pointer for PTR_C
      SAR   AR2,PTR_C             ; Save the auxiliary register

;---------------------------------------------------------------------
;          END OF PWM COMPARATOR UPDATE
;---------------------------------------------------------------------
```

```
;-------------------------------------------------------------------------
;             Oregon State University
;-------------------------------------------------------------------------
;           File :           TASK2.INC
;
;           Author :      Abdurrahman Unsal
;
;           Date :           05.26.2000
;
;           Target System : C24x Evaluation Board
;
;-------------------------------------------------------------------------
; TASK 3
;--------------- FFT and Sine Generation for PWM ----------------------
        .ref  fft128r, mag_start, modul_start, _WDKEY,
;-------------------------------------------------------------------------
  CALL FIND_MAX_MIN
  CALL COPY_BUFFER       ; Copy Modulation buffer to readout.
  CALL fft128r
  CALL mag_start         ; magnitude calculations
  CALL modul_start       ;
;-------------------------------------------------------------------------
```

```
;-------------------------------------------------------------------
;             Oregon State University
;-------------------------------------------------------------------
;          File :          PARAM.INC
;
;          Author :     Abdurrahman Unsal
;          Date :          5.26.2000
;
;-------------------------------------------------------------------
;  Description:
;  This include file contains all declarations of the parameter and
;  the signals which are used in the application.
;
;-------------------------------------------------------------------
; SYSTEM PARAMETER AND CONSTANTS
;-------------------------------------------------------------------
  parameter_16     task1_periode,1302 ; [16.0u] task1 and PWM period
                                      ; (=100ns*value)
  parameter_16           task2_periode,127  ; [16.0u] task2 period time
                                            ; (=task2*value)
  constant_16            task_LED,1         ; [16.0u] enables visualization
                                            ; of task cycles
;-------------------------------------------------------------------
; TMS320F240-Evaluation Board)
; The CPU Frequency (clock frequency) is 20 Mhz.
; MEMORY ALLOCATION FOR SPECIAL MODULES
;-------------------------------------------------------------------
; INPUT CURRENT MAGNITUDES
;-------------------------------------------------------------------
  parameter_16                    Ia1        ,0
  parameter_16                    Ia5        ,0
  parameter_16                    Ia7        ,0
  parameter_16                    Ia11       ,0
  parameter_16                    Ia13       ,0

; TEMPORARY IMPUT COEFFICIENT PARAMETERS
;-------------------------------------------------------------------
  parameter_16                    K5         ,0
  parameter_16                    K7         ,0
  parameter_16                    K11        ,0
  parameter_16                    K13        ,0
  parameter_16                    sqrtIa1    ,0

; MODULATION COEFFICIENTS
;-------------------------------------------------------------------
  parameter_16                    A          ,0
  parameter_16                    A1         ,0
  parameter_16                    A11        ,0
  parameter_16                    A111       ,0
  parameter_16                    A1111      ,0

; MODULATION TEMPORARY PARAMETERS
;-------------------------------------------------------------------
  parameter_16                    R1         ,0
  parameter_16                    R2         ,0
  parameter_16                    R3         ,0
  parameter_16                    R4         ,0
  parameter_16                    R5         ,0

; MODULATION COMPARATOR LOAD POINTERS
;-------------------------------------------------------------------
  parameter_16                    PTR_A      ,0
  parameter_16                    PTR_B      ,0
  parameter_16                    PTR_C      ,0

;  INPUT CURRENT READ_WRITE POINTERS
;-------------------------------------------------------------------
  parameter_16          I_BUF_IN_PTR     ,0
  parameter_16          I_BUF_OUT_PTR    ,0
  parameter_16          I_BUF_WR_PTR     ,0
  parameter_16          V_BUF_IN_PTR     ,0

;  OFF_SET PARAMETERS
;-------------------------------------------------------------------
  parameter_16           MIN           ,0
  parameter_16           MAX           ,0
  parameter_16           OFF_SET       ,0
```

```
;  ZERO CROSSING PARAMETERS
;-------------------------------------------------------------------------
  parameter_16          zero_counter ,0
  parameter_16          counter_125  ,0
  parameter_16          counter_126  ,0
  parameter_16          counter_127  ,0
  parameter_16          counter_128  ,0
  parameter_16          counter_129  ,0
  parameter_16          counter_130  ,0
  parameter_16          counter_131  ,0
;-------------------------------------------------------------------------
; CONSTANT DECLARATIONS FOR COEFFICIENT CALCULATIONS    |
;-------------------------------------------------------------------------
  constant_16      C1         ,05d0h
  constant_16      C2         ,0174h
;-------------------------------------------------------------------------
;       SAMPLED VOLTAGE FOR ZERO-CROSSING VOLTAGE       |
;-------------------------------------------------------------------------
  signal_16     old_sample                ,0
  signal_16     positive_zero_crossing    ,0
  signal_16     new_sample                ,0

;     FIR FILTER PARAMETERS
;-------------------------------------------------------------------------
 signal_16     vin     ,0
 signal_16     vout    ,0
 signal_16     vold    ,0

;-------------------------------------------------------------------------
;               BUFFERS DECLARATIONS
;-------------------------------------------------------------------------
;     FIR FILTER BUFFERS
;-------------------------------------------------------------------------
fir_dl:          .usect "param" ,16   ;x(n),x(n-1)..x(n-15)

;      FFT BUFFERS
;-------------------------------------------------------------------------
B0_SADR  .usect "buffers",256  ;B0 start address HAS TO START AT 2*N
B1_SADR  .usect "buffers",256  ;B1 start address
B2_SADR  .usect "buffers",32   ;B2 start address
TWID_TBL .usect "buffers",96   ;twiddle table

; INPUT DATA BUFFERS
;-------------------------------------------------------------------------
I_BUF_IN         .usect  ".buffers",256  ; Input current buffer
V_BUF_IN         .usect  ".buffers",256  ; Input current buffer
I_BUF_OUT        .usect  ".buffers",128  ; Output current buffer

; MODULATION BUFFERS (SIN TABLES PHASE A)
;-------------------------------------------------------------------------
FIRST_SINA        .usect ".buffers",128 ; The fund freq. sin function
FIFTH_SINA        .usect ".buffers",128 ; The fifth har. sin function
SEVENTH_SINA      .usect ".buffers",128 ; The seventh har sin function
ELEVENTH_SINA     .usect ".buffers",128 ; The eleventh har sin function
THIRTEEN_SINA     .usect ".buffers",128 ; The thirteenth har sin funct.

; MODULATION SIGNALS (SIN TABLES PHASE B)
;-------------------------------------------------------------------------
FIRST_SINB        .usect ".buffers",128 ; The fund freq. sin function
FIFTH_SINB        .usect ".buffers",128 ; The fifth har. sin function
SEVENTH_SINB      .usect ".buffers",128 ; The seventh har sin function
ELEVENTH_SINB     .usect ".buffers",128 ; The eleventh har sin function
THIRTEEN_SINB     .usect ".buffers",128 ; The thirteenth har sin funct

; MODULATION BUFFERS (SIN TABLES PHASE C)
;-------------------------------------------------------------------------
FIRST_SINC        .usect ".buffers",128  ; The fund freq. sin function
FIFTH_SINC        .usect ".buffers",128  ; The fifth har. sin function
SEVENTH_SINC      .usect ".buffers",128  ; The seventh har sin function
ELEVENTH_SINC     .usect ".buffers",128  ; The eleventh har sin funct
THIRTEEN_SINC     .usect ".buffers",128  ; The thirteenth har sin func

; THREE PHASE MODULATION BUFFERS
;-------------------------------------------------------------------------
MODUL_BUF_OUTA    .usect ".buffers",128  ; The modulating function
MODUL_BUF_OUTB    .usect ".buffers",128  ; The modulating function
MODUL_BUF_OUTC    .usect ".buffers",128  ; The modulating function
```

```
; THREE PHASE MODULATION BUFFERS TO LOAD TO THE COMPARATORS
;-----------------------------------------------------------------------
MODUL_BUFA          .usect ".buffers",128  ; The modulating function
MODUL_BUFB          .usect ".buffers",128  ; The modulating function
MODUL_BUFC          .usect ".buffers",128  ; The modulating function
```

```
;-------------------------------------------------------------------------

;-------------------------------------------------------------------------
;               Oregon State University
;-------------------------------------------------------------------------
;           File :          INIT.INC
;
;           Author :      Abdurrahman Unsal
;
;           Date :          05.26.2000
;-------------------------------------------------------------------------
;  Description:
;
;  This include file contains the macro-calls for the initialization of
;  a specific application. Therefor only macros should be used but no
;  assembler code. All the macros will be executed only once at the
;  start of the program and that will happen in the listed order.
;  All the buffers/pointers are initialized in this file.
;-------------------------------------------------------------------------

; INITIALIZATION
;-------------------------------------------------------------------------
;  AD-converter
;-------------------------------------------------------------------------

 LDPK       #(_ADCTRL1)                     ; AD control reg 1
 SPLK       #9800h,_ADCTRL1
 LDPK       _ADCTRL2                        ; AD control reg 2
 SPLK       #0403h,_ADCTRL2
 LDPK       _OCRA
 SPLK       #8703h,_OCRA

;-------------------------------------------------------------------------
; Simple compare (SCU)
;-------------------------------------------------------------------------

 LDPK       _SACTR
 SPLK       #0015h,_SACTR
 LDPK       _COMCON
 SPLK       #0100h,_COMCON
 SPLK #8100h,_COMCON
;-------------------------------------------------------------------------
 LDP  #(PTR_A)                  ; Data page pointer A
 SPLK #(MODUL_BUFA),PTR_A       ; Load the PTR_A
 LDP  #(PTR_B)                  ; Data page pointer B
 SPLK #(MODUL_BUFB),PTR_B       ; Load the PTR_B
 LDP  #(PTR_C)                  ; Data page pointer C
 SPLK #(MODUL_BUFC),PTR_C       ; Load the PTR_C
;-------------------------------------------------------------------------
 LDP  #(I_BUF_IN_PTR)                ; Data page for input current ptr
 SPLK #(I_BUF_IN+128),I_BUF_IN_PTR   ; Load I_BUF_IN_PTR
 LDP  #(V_BUF_IN_PTR)                ; Data page ptr for input ptr
 SPLK #(V_BUF_IN),V_BUF_IN_PTR     ; Load I_BUF_IN_PTR with input
 LDP  #(I_BUF_OUT_PTR)             ; Output current page pointer
 SPLK #(I_BUF_IN),I_BUF_OUT_PTR    ; Initial value of output current
 LDP  #(I_BUF_WR_PTR)              ; Output Write Current page pointer
 SPLK #(I_BUF_OUT),I_BUF_WR_PTR    ; Initial value of output Current
 LDPK _T2PR                          ; Load T2PR-register with init-value
 BLKD #task1_periode,_T2PR         ; Load task1_periode it T2PR
 LDP #old_sample            ; Load data page for old_sample
 SPLK #0, old_sample        ; Store 0 to the old_sample
 LDP  #new_sample           ; Load data page for new_sample
 SPLK #0h,new_sample        ; Store zero to new_sample
 LDP  #OFF_SET              ; Load data page for new_sample
 SPLK #8140h,OFF_SET        ; Store 8140h to the OFF_SET
 LDP  #positive_zero_crossing ; Load data page for positive_zero_crossing
 SPLK #1h,positive_zero_crossing ; Store 1h to positive zero_crossing
 LDP #zero_counter      ; Load data page for zero_counter
 SPLK #0,zero_counter   ; Store 0 to zero_counter
 LDP #counter_126       ; Load data page for counter_126
 SPLK #0,counter_126    ; Store 0h to the counter_126
 LDP #counter_127       ; Load data page for counter_127
 SPLK #0,counter_127    ; Store 0h to the counter_127
 LDP #counter_128       ; Load data page for counter_128
 SPLK #0,counter_128    ; Store 0h to the counter_128
 LDP #counter_129       ; Load data page for counter_129
 SPLK #0,counter_129    ; Store 0h to the counter_129
 LDP #counter_130       ; Load data page for counter_130
```

```
 SPLK #0,counter_130    ; Store 0h to the counter_130

;      This is the code for fft initialization
;-----------------------------------------------------------------------
fft128r_init
 LACC       #COS45K
 SACL       COS45                    ;Init location with constant.
 LAR        AR0, #95          ;set # of Twiddles to Xfer
 LAR        AR6, #TWID_TBL         ;AR6 points to 1st dest DM addr
 LACC       #PS_TWID_STRT          ;ACC points to 1st Src PM addr
TWD_XFR MAR*,AR6                      ;select AR6
 TBLR       *+, AR0           ;Mov Prog word --> Data mem
 ADD        #1                       ;inc ACC
 BANZ       TWD_XFR           ;continue Xfer until AR0=0

;-----------------------------------------------------------------------
;      Zero Input Buffer, I_BUF_IN                |
;-----------------------------------------------------------------------
 LACC #0
 LAR AR2, #(I_BUF_IN)
 LAR AR7, #255
 MAR *,AR2
ZERO_I_BUF_IN:
 SACL *+,0,AR7
 BANZ ZERO_I_BUF_IN,*-,AR2

;-----------------------------------------------------------------------
; Zero Input Buffer, I_BUF_OUT
;-----------------------------------------------------------------------
  LAR AR2, #(I_BUF_OUT)
  LAR AR7, #127
  MAR *,AR2

ZERO_I_BUF_OUT:
  SACL *+,0,AR7
  BANZ ZERO_I_BUF_OUT,*-,AR2
;-----------------------------------------------------------------------
```

```
;-----------------------------------------------------------------------
;                   Oregon State University
;-----------------------------------------------------------------------
;
;           File :        ZERO.ASM
;
;           Author :      Abdurrahman Unsal
;
;           Date :           05.26.2000
;
;-----------------------------------------------------------------------
; Defined symbols
  .def ZERO_CROSS
;-----------------------------------------------------------------------
; Imported variables, symbols, and labels
;-----------------------------------------------------------------------
  .ref positive_zero_crossing, new_sample, vout, vold, COPY_BUFFER
  .ref   _ADCFIFO2,OFF_SET,PTR_A, PTR_B, PTR_C, MODUL_BUFA, MODUL_BUFB,
  .ref MODUL_BUFC, zero_counter, counter_125, counter_126, counter_127
  .ref counter_128, counter_129,counter_130,counter_131
;-----------------------------------------------------------------------
ZERO_CROSS:

   ZAC                  ; Zero Accml
   SETC SXM             ; Set sign Extention mode
   LDP #vold            ; Load data page for vold
   LACC vold            ; Load accumulator with vold
   BCND DONE,GEQ        ; Branch to done if acc is GEQ

   LDP #vout            ; Load data page for vout
   LACC vout            ; Load accumulator with vout
   BCND DONE,LT         ; Branch to done if acc is less than zero

   LDP #vold            ; Load data page for vold
   SACL vold            ; Store accumulator to vold

   LDP  #positive_zero_crossing
   SPLK #1h,positive_zero_crossing  ; Store 1 to positive_zero_crossing
   B  RESET_MODUL_POINTERS          ; Branch to Reset_Modul_Pointers

DONE:
   LDP #vout         ; Load data page for vout
   LACC vout         ; Load accumulator with vout
   LDP #vold         ; Load data page with vold
   SACL vold         ; Store accumulator with vold

   LDP #zero_counter ; Load data page for zero_counter
   LACL zero_counter ; Load accumulator with zero_counter
   ADD #1h           ; Add 1h to the accumulator
   SACL zero_counter ; Store the accumulator to zero_counter
   RET               ; Return

RESET_MODUL_POINTERS:
  CALL COPY_BUFFER

 LDP #zero_counter    ; Load data page with zero_counter
 LACL zero_counter    ; Load accumulator with zero_counter

 SUB #125             ; Substract 125 from acc.
 BCND  end_COUNT,LT
 SUB #7               ; Substract 7 from acc.
 BCND  RESET_CONTR,GT ;
 ADD #7               ; Add 7 to the accumulator
 BCND RST_CNTR1,EQ    ; Branch to RST_CNTR1 if the acc is equal to zero
 SUB #1               ; Substract 1 from accumulator
 BCND RST_CNTR2,EQ    ; Branch to RST_CNTR2 if the acc is equal to zero
 SUB #1               ; Substract 1 from accumulator
 BCND RST_CNTR3,EQ    ; Branch to RST_CNTR3 if the acc is equal to zero
 SUB #1               ; Substract 1 from accumulator
 BCND RST_CNTR4,EQ    ; Branch to RST_CNTR4 if the acc is equal to zero
 SUB #1               ; Substract 1 from accumulator
 BCND RST_CNTR5,EQ    ; Branch to RST_CNTR5 if the acc is equal to zero
 SUB #1               ; Substract 1 from accumulator
 BCND RST_CNTR6,EQ    ; Branch to RST_CNTR6 if the acc is equal to zero
 SUB #1               ; Substract 1 from accumulator
 BCND RST_CNTR7,EQ    ; Branch to RST_CNTR7 if the acc is equal to zero

RST_CNTR1:
```

```
 LDP #(counter_125)  ; Load data page for counter_125
 LACL counter_125    ; Load accumulator with counter_125
 ADD #1              ; Add 1 to the accumulator
 SACL counter_125    ; Store accumulator to counter_125
 B RESET_CONTR       ; Branch to RESET_CONTR

RST_CNTR2:
     LDP #(counter_126)  ; Load data page for counter_126
     LACL counter_126    ; Load accumulator with counter_126
     ADD #1              ; Add 1 to the accumulator
     SACL counter_126    ; Store accumulator to counter_126

      B RESET_CONTR       ; Branch to RESET_CONTR
RST_CNTR3:
 LDP #(counter_127) ; Load data page for counter_127
 LACL counter_127   ; Load accumulator with counter_127
 ADD #1             ; Add 1 to the accumulator
 SACL counter_127   ; Store accumulator to counter_127
 LDP  #(PTR_A)
 SPLK #(MODUL_BUFA+38),PTR_A
 LDP  #(PTR_B)
 SPLK #(MODUL_BUFB+38),PTR_B
 LDP  #(PTR_C)
 SPLK #(MODUL_BUFC+38),PTR_C
 B RESET_CONTR    ; Branch to RESET_CONTR

RST_CNTR4:
     LDP #(counter_128) ; Load data page for counter_128
     LACL counter_128   ; Load accumulator with counter_128
     ADD #1             ; Add 1 to the accumulator
     SACL counter_128   ; Store accumulator to counter_128

; If the zero cross is 128 samples apart.
 LDP  #(PTR_A)
 SPLK #(MODUL_BUFA+38),PTR_A
 LDP  #(PTR_B)
 SPLK #(MODUL_BUFB+38),PTR_B
 LDP  #(PTR_C)
 SPLK #(MODUL_BUFC+38),PTR_C
      B RESET_CONTR  ; Branch to RESET_CONTR

RST_CNTR5:
 LDP #(counter_129) ; Load data page for counter_129
 LACL counter_129   ; Load accumulator with counter_129
 ADD #1             ; Add 1 to the accumulator
 SACL counter_129   ; Store accumulator to counter_129

 LDP  #(PTR_A)
 SPLK #(MODUL_BUFA+38),PTR_A
 LDP  #(PTR_B)
 SPLK #(MODUL_BUFB+38),PTR_B
 LDP  #(PTR_C)
 SPLK #(MODUL_BUFC+38),PTR_C
 B RESET_CONTR  ; Branch to RESET_CONTR

RST_CNTR6:
     LDP #(counter_130)  ; Load data page for counter_130
     LACL counter_130    ; Load accumulator with counter_130
     ADD #1              ; Add 1 to the accumulator
     SACL counter_130    ; Store accumulator to counter_130
     B RESET_CONTR       ; Branch to RESET_CONT

RST_CNTR7:
 LDP #(counter_131)  ; Load data page for counter_131
 LACL counter_131    ; Load accumulator with counter_131
 ADD #1              ; Add 1 to the accumulator
 SACL counter_131    ; Store accumulator to counter_130
 B RESET_CONTR       ; Branch to RESET_CONT

RESET_CONTR
     LDP #zero_counter     ; Load data page for zero_counter
     SPLK #1h,zero_counter

     LDP  #positive_zero_crossing
     SPLK #0h,positive_zero_crossing
end_COUNT  ; End
 RET       ; Return
;-----------------------------------------------------------------------
```

```
;-----------------------------------------------------------------------
;          Oregon State University
;-----------------------------------------------------------------------
;
;          File :          MAX_MIN.ASM
;
;          Author :     Abdurrahman Unsal
;
;          Date :          05.26.2000
;
; The program finds the offset for the input.
;-----------------------------------------------------------------------
   .ref MIN, MAX, OFF_SET,I_BUF_IN
   .def FIND_MAX_MIN

FIND_MAX_MIN

   LAR AR2, #(I_BUF_IN)
   LAR AR7,#255
   LDP #(MAX)
   BLKD #I_BUF_IN,MAX
   LDP #(MIN)
   BLKD #I_BUF_IN,MIN
   MAR      *,AR2                          ;ARP = AR2

FIND_MAX    LACC        *+,0,AR7           ;ACC = Value pointed by AR2
                        LDP      #(MAX)
                SUB     MAX                ;Subtract MAX
                        BCND        RESUME_MAX,LEQ          ;
                        LDP     #(MAX)
                        ADD         MAX
                        SACL        MAX    ;Store the new MAX value

RESUME_MAX  BANZ FIND_MAX,*-,AR2

        LAR AR2, #(I_BUF_IN)
        LAR AR7,#255
        MAR *,AR2

FIND_MIN    LACC *+,0,AR7
            LDP  #(MIN)
            SUB  MIN                ;Subtract MIN
            BCND RESUME_MIN,GEQ
            LDP #(MIN)
            ADD   MIN
            SACL  MIN               ;Store the new MIN value

RESUME_MIN  BANZ FIND_MIN,*-,AR2

FIND_OFF_SET
        ZAC
        LDP #(MAX)
        LACL MAX
        LDP #(MIN)
        ADD MIN
        SFR
        LDP #(OFF_SET)
        SACL OFF_SET
        RET
;-----------------------------------------------------------------------
```

```
;-------------------------------------------------------------------
;               F240REG.ASM
;-------------------------------------------------------------------
;
;           File :          F240REG.ASM
;
;           Orginator :     S. Lindegger / D. Weyermann
;
;           Date :          04.01.1998
;
;           Version:        1.0
;
;           Target System : C240 Evaluation Board
;
;-------------------------------------------------------------------
;
;  Modification:        Abdurrahman Unsal
;
;-------------------------------------------------------------------
;  Description:
;
;-------------------------------------------------------------------
;On Chip Peripheral Register Definitions (All registers mapped into data
;  space unless otherwise noted)
;-------------------------------------------------------------------

;C2xx Core Registers
;~~~~~~~~~~~~~~~~~~~
  .def _IMR, _GREG, _IFR
_IMR                .set        0004h               ;Int Mask Reg.
_GREG               .set        0005h               ;Global Memory Allocation Reg.
_IFR                .set        0006h               ;Int Flag Reg.
;_ABRPT             .set        01fh                ;Analysis BreakPoint Reg.

;System Module Registers
;~~~~~~~~~~~~~~~~~~~~~~
  .def _SYSCR, _SYSSR, _SYSIVR

_SYSCR              .set        07018h      ;System Module Control Reg.
_SYSSR              .set        0701Ah      ;System Module Status Reg.
;_DIN               .set        0701Ch      ;Device Identification Reg.
_SYSIVR             .set        0701Eh      ;System Interrupt Vector Reg.

;External Interrupt Registers
;~~~~~~~~~~~~~~~~~~~~~~~~~~~
  .def _XINT1, _NMI, _XINT2, _XINT3

_XINT1      .set        07070h              ;Int1 (type A) Control Reg.
_NMI        .set        07072h              ;Non maskable Int (type A) Control Reg.
_XINT2      .set        07078h              ;Int2 (type C) Control Reg.
_XINT3      .set        0707Ah              ;Int3 (type C) Control Reg.

 .page

;Digital I/O
;~~~~~~~~~~~
  .def _OCRA, _OCRB, _ISRA, _ISRB, _PADATDIR, _PBDATDIR, _PCDATDIR, _PDDATDIR
_OCRA               .set        07090h              ;Output Mux Control Reg. A
_OCRB               .set        07092h              ;Output Mux Control Reg. B
_ISRA               .set        07094h              ;Input Status Reg A.
_ISRB               .set        07096h              ;Input Status Reg B.
_PADATDIR   .set        07098h      ;I/O port A Data & Direction Reg.
_PBDATDIR   .set        0709Ah      ;I/O port B Data & Direction Reg.
_PCDATDIR   .set        0709Ch      ;I/O port C Data & Direction Reg.
_PDDATDIR   .set        0709Eh      ;I/O port D Data & Direction Reg.

;Watch-Dog(WD) / Real Time Int(RTI) / Phase Lock Loop(PLL) Registers
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  .def _RTICNTR, _WDCNTR, _WDKEY, _RTICR, _WDCR, _CKCR0, _CKCR1

_RTICNTR    .set        07021h              ;RTI Counter Reg.
_WDCNTR     .set        07023h              ;WD Counter Reg.
_WDKEY              .set        07025h              ;WD Key Reg.
_RTICR      .set        07027h              ;RTI Control Reg.
_WDCR       .set        07029h              ;WD Control Reg.
_CKCR0              .set        0702Bh              ;PLL Clock Control Reg 0.
_CKCR1              .set        0702Dh              ;PLL Clock Control Reg 1.
```

```
;Analog-to-Digital Converter(ADC) registers
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  .def _ADCTRL1, _ADCTRL2, _ADCFIFO1, _ADCFIFO2

_ADCTRL1    .set       07032h                 ;ADC Control Reg. 1
_ADCTRL2    .set       07034h                 ;ADC Control Reg. 2
_ADCFIFO1   .set       07036h                 ;ADC FIFO Data Reg. 1 for ADC 1
_ADCFIFO2   .set       07038h                 ;ADC FIFO Rata Reg. 2 for ADC 2

;Digital-to-Analog Converter(DAC) registers
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  .def _DAC0, _DAC1, _DAC2, _DAC3, _DAC_update

_DAC0                  .set       00000h                 ;Input Data Reg. for DAC0
_DAC1                  .set       00001h                 ;Input Data Reg. for DAC1
_DAC2                  .set       00002h                 ;Input Data Reg. for DAC2
_DAC3                  .set       00003h                 ;Input Data Reg. for DAC3
_DAC_update .set       00004h                 ;DAC update Reg.

.page

;Serial Peripheral Interface (SPI) Registers
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  .def _SPICCR, _SPICTL, _SPISTS, _SPIBRR, _SPIEMU
  .def _SPIBUF, _SPIDAT, _SPIPC1, _SPIPC2, _SPIPRI

_SPICCR     .set       07040h                 ;SPI Config. Control Reg. 1
_SPICTL     .set       07041h                 ;SPI Operation Control Reg. 2
_SPISTS     .set       07042h                 ;SPI Status Reg.
_SPIBRR     .set       07044h                 ;SPI Baud Rate Reg.
_SPIEMU     .set       07046h                 ;SPI Emulation Buffer Reg.
_SPIBUF     .set       07047h                 ;SPI Serial Input Buffer Reg.
_SPIDAT     .set       07049h                 ;SPI Serial Data Reg.
_SPIPC1     .set       0704Dh                 ;SPI Port Control Reg. 1
_SPIPC2     .set       0704Eh                 ;SPI Port Control Reg. 2
_SPIPRI     .set       0704Fh                 ;SPI Priority Control Reg.

;Serial Communications Interface (SCI) Registers
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  .def _SCICCR, _SCICTL1, _SCIHBAUD, _SCILBAUD, _SCICTL2, _SCIRXST
  .def _SCIRXEMU, _SCIRXBUF, _SCITXBUF, _SCIPC1, _SCIPC2, _SCIPRI

_SCICCR     .set       07050h                 ;SCI Communication Control Reg.
_SCICTL1    .set       07051h                 ;SCI Control Reg. 1
_SCIHBAUD   .set       07052h                 ;SCI Baud Rate Reg. High Bits
_SCILBAUD   .set       07053h                 ;SCI Baud Rate Reg. Low Bits
_SCICTL2    .set       07054h                 ;SCI Control Reg. 2
_SCIRXST    .set       07055h                 ;SCI Receiver Status Reg.
_SCIRXEMU   .set       07056h                 ;SCI Emulation Data Buffer Reg.
_SCIRXBUF   .set       07057h                 ;SCI Receiver Data Buffer Reg.
_SCITXBUF   .set       07059h                 ;SCI Transmit Data Buffer Reg.
_SCIPC1                .set       0705Dh                 ;SCI Port Control Reg. 1
_SCIPC2                .set       0705Eh                 ;SCI Port Control Reg. 2
_SCIPRI     .set       0705Fh                 ;SCI Priority Control Reg.

;Waite State Generator Registers (mapped into I/O space)
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  .def _WSGR
_WSGR                  .set       0FFFFh                 ;Waite State Generator Reg.

 .page
;General Purpose Timer Registers - Event Manager (Base = 7400h)
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
EV_BASE                .set       7400h                  ;Event Manager Base Address
  .def _GPTCON, _T1CNT, _T1CMPR, _T1PR, _T1CON, _T2CNT, _T2CMPR, _T2PR, _T2CON
  .def _T3CNT, _T3CMPR, _T3PR, _T3CON

_GPTCON                .set       00h + EV_BASE          ;General Purpose Timer Control Reg.
_T1CNT                 .set       01h + EV_BASE          ;GP Timer 1 Counter Reg.
_T1CMPR                .set       02h + EV_BASE          ;GP Timer 1 Compare Reg.
_T1PR                  .set       03h + EV_BASE          ;GP Timer 1 Period Reg.
_T1CON                 .set       04h + EV_BASE          ;GP Timer 1 Control Reg.
_T2CNT                 .set       05h + EV_BASE          ;GP Timer 2 Counter Reg.
_T2CMPR                .set       06h + EV_BASE          ;GP Timer 2 Compare Reg.
_T2PR                  .set       07h + EV_BASE          ;GP Timer 2 Period Reg.
_T2CON                 .set       08h + EV_BASE          ;GP Timer 2 Control Reg.
_T3CNT                 .set       09h + EV_BASE          ;GP Timer 3 Counter Reg.
_T3CMPR                .set       0ah + EV_BASE          ;GP Timer 3 Compare Reg.
```

```
_T3PR                   .set        0bh + EV_BASE          ;GP Timer 3 Period Reg.
_T3CON                  .set        0ch + EV_BASE          ;GP Timer 3 Control Reg.

; Full & Simple Compare Unit Registers - Event Manager (Base = 7400h)
;-------------------------------------------------------------------------
  .def _T3CNT, _T3CMPR, _T3PR, _T3CON, _COMCON, _ACTR, _SACTR
  .def _DBTCON, _CMPR1, _CMPR2, _CMPR3, _SCMPR1, _SCMPR2, _SCMPR3

_COMCON                 .set        11h + EV_BASE          ;Compare Control Reg.
_ACTR                   .set        13h + EV_BASE          ;Full Compare Action Control Reg.
_SACTR                  .set        14h + EV_BASE          ;Simple Compare Action Control Reg.
_DBTCON                 .set        15h + EV_BASE          ;Dead-Band Timer Control Reg.
_CMPR1                  .set        17h + EV_BASE          ;Full Compare Unit Compare Reg. 1
_CMPR2                  .set        18h + EV_BASE          ;Full Compare Unit Compare Reg. 2
_CMPR3                  .set        19h + EV_BASE          ;Full Compare Unit Compare Reg. 3
_SCMPR1                 .set        1ah + EV_BASE          ;Simple Compare Unit Compare Reg. 1
_SCMPR2                 .set        1bh + EV_BASE          ;Simple Compare Unit Compare Reg. 2
_SCMPR3                 .set        1ch + EV_BASE          ;Simple Compare Unit Compare Reg. 3

;Capture & QEP Registers - Event Manager (Base = 7400h)
;-------------------------------------------------------
  .def _CAPCON, _CAPFIFO, _CAPFIFO1, _CAPFIFO2, _CAPFIFO3, _CAPFIFO4

_CAPCON                 .set        20h + EV_BASE          ;Capture Control Reg.
_CAPFIFO    .set        22h + EV_BASE          ;FIFO1-4 Status Reg.
_CAPFIFO1   .set        23h + EV_BASE          ;Capture FIFO Stack 1 Top
_CAPFIFO2   .set        24h + EV_BASE          ;Capture FIFO Stack 2 Top
_CAPFIFO3   .set        25h + EV_BASE          ;Capture FIFO Stack 3 Top
_CAPFIFO4   .set        26h + EV_BASE          ;Capture FIFO Stack 4 Top

;Interrupt Registers - Event Manager (Base = 7400h)
;--------------------------------------------------
  .def _EVIMRA, _EVIMRB, _EVIMRC, _EVIFRA, _EVIFRB, _EVIFRC
  .def _EVIVRA, _EVIVRB, _EVIVRC

_EVIMRA                 .set        2ch + EV_BASE          ;Group A Interrupt Mask Reg.
_EVIMRB                 .set        2dh + EV_BASE          ;Group B Interrupt Mask Reg.
_EVIMRC                 .set        2eh + EV_BASE          ;Group C Interrupt Mask Reg.
_EVIFRA                 .set        2fh + EV_BASE          ;Group A Interrupt Flag Reg.
_EVIFRB                 .set        30h + EV_BASE          ;Group B Interrupt Flag Reg.
_EVIFRC                 .set        31h + EV_BASE          ;Group C Interrupt Flag Reg.
_EVIVRA                 .set        32h + EV_BASE ;Group A Int. Vector Offset(AD dec only)
_EVIVRB                 .set        33h + EV_BASE ;Group B Int. Vector Offset(AD dec only)
_EVIVRC                 .set        34h + EV_BASE ;Group C Int. Vector Offset(AD dec only)

 .page

;-------------------------------------------------------------------------
; Constant defines
;-------------------------------------------------------------------------
;Data Memory Boundary Addresses
;-------------------------------------------------------------------------
  .def _B0SADDR, _B0EADDR, _B1SADDR, _B1EADDR, _B2SADDR, _B2EADDR

_B0SADDR    .set        00100h              ;Block B0 start address
_B0EADDR    .set        001FFh              ;Block B0 end address
_B1SADDR    .set        00300h              ;Block B1 start address
_B1EADDR    .set        003FFh              ;Block B1 end address
_B2SADDR    .set        00060h              ;Block B2 start address
_B2EADDR    .set        0007Fh              ;Block B2 end address

;Bit codes for Test bit instruction (BIT)
;----------------------------------------------------------------------
  .def _BIT15, _BIT14, _BIT13, _BIT12, _BIT11, _BIT10, _BIT9, _BIT8
  .def _BIT7, _BIT6, _BIT5, _BIT4, _BIT3, _BIT2, _BIT1, _BIT0

_BIT15                  .set        0000h              ;BIT Code for 15
_BIT14                  .set        0001h              ;BIT Code for 14
_BIT13                  .set        0002h              ;BIT Code for 13
_BIT12                  .set        0003h              ;BIT Code for 12
_BIT11                  .set        0004h              ;BIT Code for 11
_BIT10                  .set        0005h              ;BIT Code for 10
_BIT9                   .set        0006h              ;BIT Code for 9
_BIT8                   .set        0007h              ;BIT Code for 8
_BIT7                   .set        0008h              ;BIT Code for 7
_BIT6                   .set        0009h              ;BIT Code for 6
_BIT5                   .set        000Ah              ;BIT Code for 5
_BIT4                   .set        000Bh              ;BIT Code for 4
```

```
_BIT3                   .set        000Ch               ;BIT Code for 3
_BIT2                   .set        000Dh               ;BIT Code for 2
_BIT1                   .set        000Eh               ;BIT Code for 1
_BIT0                   .set        000Fh               ;BIT Code for 0

;External Data Space Registers
;-------------------------------------------------------------------------
  .def _EXTDATA

_EXTDATA    .set        8000h

;-------------------------------------------------------------------------
; M A C R O  -  Definitions
;-------------------------------------------------------------------------

;Watch-Dog Macros
;~~~~~~~~~~~~~~~~
  .global _Kick_WD, _Dis_WD

;Watchdog reset
_Kick_WD    .macro
                        LDP         #00E0h              ;DP --> 7000h-707Fh
                        SPLK        #0055h, WDKEY       ;WDCNTR is enable to reset
                        SPLK        #00AAh, WDKEY       ;WDCNTR is reset
                        LDP         #00h                ; DP --> 0000h-007Fh
                        .endm

;Watchdog disable
_Dis_WD                 .macro
                        LDP         #00E0h              ;DP --> 7000h-707Fh
                        SPLK        #006fh, WDCR        ;Disable WD if VCPP=5V (JP5 in pos. 2-
3)
                        Kick_WD
                        .endm
.end
;-------------------------------------------------------------------------
```

```
;-------------------------------------------------------------------
;                   Oregon State University
;-------------------------------------------------------------------
;
;          File :          VARS.ASM
;
;          Author :     Abdurrahman Unsal
;
;          Date :          5.26.2000
;
;-------------------------------------------------------------------
;  Description:
;  This library contains the declaration of the temporary variables and
;    provides a number of macros which helps to declare specific para-
;    meter and signals for an application. These macros are :
;
;          'parameter_16'                    keyword for parameterdeclaration, 16Bit
;          'signal_16'             keyword for signaldeclaration, 16Bit
;          'constant_16'                     keyword for constantdeclaration, 16Bit
;          'number_of_ctr'          allocates memory for controllers
;          'number_of_wt'           allocates memory for wt_phi modules
;
;    There is also a macro that updates some special function register
;    with the content of the parameter. This macro is called :
;       'register_update'
;
;-------------------------------------------------------------------
; Declarations of Constants
;-------------------------------------------------------------------
true                    .equ        0FFFFh                  ; Boolean true
false                   .equ        0000h                   ; Boolean false
;-------------------------------------------------------------------
; Declarations of temporary Registers and  Register backup
;-------------------------------------------------------------------

task2_flag      .usect  "param", 1    ;1 -> run task3, 0 -> don't run it
task2_counter   .usect  "param", 1    ;counter for task 3 handling
task2_error     .usect  "param", 1    ;0: no error; else: error

temp_t1_1   .usect      "task_1", 1 ;temp1 variable for task 1
temp_t1_2   .usect      "task_1", 1 ;temp2       "
temp_t1_3   .usect      "task_1", 1 ;temp3       "
temp_t1_4   .usect      "task_1", 1 ;temp4       "
temp_t1_5   .usect      "task_1", 1 ;temp5       "
temp_t1_6   .usect      "task_1", 1 ;temp6       "
registers_t1            .usect      "task_1", 22;preserved registers and hw-stack

temp_t2_1   .usect      "task_2", 1 ;temp1 variable for task 2
temp_t2_2   .usect      "task_2", 1 ;temp2       "
temp_t2_3   .usect      "task_2", 1 ;temp3       "
temp_t2_4   .usect      "task_2", 1 ;temp4       "
temp_t2_5   .usect      "task_2", 1 ;temp5       "
temp_t2_6   .usect      "task_2", 1 ;temp6       "
registers_t2            .usect      "task_2", 22;preserved registers and hw-stack

temp_t3_1   .usect      "task_3", 1 ;temp1 variable for task 3
temp_t3_2   .usect      "task_3", 1 ;temp2       "
temp_t3_3   .usect      "task_3", 1 ;temp3       "
temp_t3_4   .usect      "task_3", 1 ;temp4       "
temp_t3_5   .usect      "task_3", 1 ;temp5       "
temp_t3_6   .usect      "task_3", 1 ;temp6       "

temp_t4_1   .usect      "task_4", 1 ;temp1 variable for task 4
temp_t4_2   .usect      "task_4", 1 ;temp2       "
temp_t4_3   .usect      "task_4", 1 ;temp3       "
temp_t4_4   .usect      "task_4", 1 ;temp4       "
temp_t4_5   .usect      "task_4", 1 ;temp5       "
temp_t4_6   .usect      "task_4", 1 ;temp6       "

;-------------------------------------------------------------------
; Macros for Signal- and Parameter types
;-------------------------------------------------------------------

;*******************************************************************
;  number_of_ctr (macro)                    ISB,10.11.98 M.Voser
;  -------------
;  Description :
;    Memory-allocation for historical values in task 1 and 2. How many
```

```
;    'old-values' were needed depends on the number of controllers.
;
;  Call-parameter :
;    number            number of used controllers in whole application
;
;  Return-parameter :
;    none
;
;************************************************************************
number_of_ctr .macro number
               .if number > 0
                       .eval 0,i
                       .loop
             .eval i+1,i
y_old_t1_:i:             .usect    "task_1", 2 ; historical values from controller in task 1
y_old_t2_:i:             .usect    "task_2", 2 ; historical values from controller in task 2
                         LDPK      y_old_t1_:i:
                         SPLK      #0000h,y_old_t1_:i:
                         SPLK      #0000h,y_old_t1_:i:+1
                         LDPK      y_old_t2_:i:
                         SPLK      #0000h,y_old_t2_:i:
                         SPLK      #0000h,y_old_t2_:i:+1
                         .break i=number
                        .endloop
                       .endif
                .endm


;************************************************************************
;  number_of_wt (macro)                ISB,10.11.98 M.Voser
;  ------------
;  Description :
;    Memory-allocation for historical values in task 1. How many
;    'old-values' were needed depends on the number of used moduls.
;
;  Call-parameter :
;    number number of used wt_phi moduls in whole application
;************************************************************************
number_of_wt  .macro number
                .if number > 0
                        .eval 0,i
                        .loop
              .eval i+1,i
w_old_t1_:i:             .usect    "task_1", 1   ; historical values from wt_phi in task 1
                         LDPK      w_old_t1_:i:
                         SPLK      #0000h,w_old_t1_:i:
                         .break i=number
                        .endloop
                .endif
                 .endm


;************************************************************************
;  signal_16 (macro)                          ISB,10.11.98 M.Voser
;  ---------
;  Description :
;  Allocates and initialize memory-space for a 16Bit-Signal in the page
;    called 'signal' (see MAIN.LIN).
;
;  Call-parameter :
;    name_of_signal    name of the signal
;    initial_value     initial value of signal at startup
;
;************************************************************************
signal_16               .macro name_of_signal, initial_value
:name_of_signal: .usect           "signal", 1 ; declaration of a word
                        LDPK      :name_of_signal:
                        SPLK      #:initial_value:,:name_of_signal:
                        .endm

;************************************************************************
;  parameter_16 (macro)                                  ISB,10.11.98 M.Voser
;  ------------
;  Description :
;Allocates and initialise memory-space for a 16Bit-Parameter in the
; page
;    called 'param' (see MAIN.LIN).
;
```

```
;  Call-parameter :
;    name_of_param     name of the parameter
;    initial_value     initial value of the parameter
;
;*************************************************************************
parameter_16            .macro name_of_param, initial_value
:name_of_param:  .usect          "param", 1             ; declaration of a word
                        LDPK     :name_of_param:
                        SPLK     #:initial_value:,:name_of_param:
                        .endm
;*************************************************************************
;  constant_16 (macro)                          ISB,10.11.98 M.Voser
;  ------------
;  Description :
;
;  Call-parameter :
;    name_of_const     name of the constant
;    initial_value     initial value of the constant
;
;  Return-parameter :
;    none
;
;  Calling convention :
;
;*************************************************************************
constant_16 .macro name_of_const, initial_value
:name_of_const:  .equ  :initial_value:
                        .endm


;-------------------------------------------------------------------------
; Servicemacro
;*************************************************************************
;  register_update (macro)                              ISB,3.9.98 R.Haldemann
;  ---------------
;  Description :
;  Copies the contents of parameter into the special function register.
;
;*************************************************************************

register_update  .macro
register_update?:
          LDPK       dead_band_time                  ; restore parameter
          LACC       dead_band_time,8                ; load db-time and
          OR         #_c_dead_band_conf    ; configure dead-band generation
          LDPK       _DBTCON
          SACL       _DBTCON

                                           ; restore timer-registers
          LDPK       _T1PR             ; Load T1PR-register with init-value
          BLKD       #task1_periode,_T1PR
       LDPK          _T2PR                ; Load T2PR-register with init-value
          BLKD       #task1_periode,_T2PR
          .endm
```

```
;-----------------------------------------------------------------------
;               VECS.ASM
;-----------------------------------------------------------------------
;
;           File :          VECS.ASM
;
;           Orginator :     R. Haldemann / M. Voser
;
;           Date :          1.11.99
;
;           Version:        1.0
;
;           Target System : C24x Evaluation Board
;
;-----------------------------------------------------------------------
;  Modification: Abdurrahman Unsal
;
;-----------------------------------------------------------------------
;  Description:
;
;    interrupt vector table
;
;-----------------------------------------------------------------------
 .ref _main, _task1, _task2, PHANTOM1, PHANTOM2, PHANTOM3, PHANTOM4
 .ref  PHANTOM5, PHANTOM6, PHANTOM7, PHANTOM8, PHANTOM9, PHANTOM10,
 .ref PHANTOM11, PHANTOM14, PHANTOM15, PHANTOM16, PHANTOM17, PHANTOM18,
 .ref PHANTOM19, PHANTOM20, PHANTOM21, PHANTOM22, PHANTOM23, PHANTOM13,
 .ref PHANTOM12
;-----------------------------------------------------------------------
; Vector address declarations
;-----------------------------------------------------------------------
                    .sect       ".vectors"

RSVECT                  B    PHANTOM1          ; Reset Vector
INT1                    B    PHANTOM2          ; Interrupt Level 1
INT2                    B    _task1 ; Interrupt Level 2 Task 1
INT3                    B    PHANTOM3          ; Interrupt Level 3 Task 2
INT4                    B    PHANTOM4          ; Interrupt Level 4
INT5                    B    PHANTOM5          ; Interrupt Level 5
INT6                    B    PHANTOM6          ; Interrupt Level 6
RESERVED    B     PHANTOM7          ; Reserved
SW_INT8     B     PHANTOM8          ; User S/W Interrupt
SW_INT9     B     PHANTOM9          ; User S/W Interrupt
SW_INT10    B     PHANTOM10         ; User S/W Interrupt
SW_INT11    B     PHANTOM11         ; User S/W Interrupt
SW_INT12    B     PHANTOM12         ; User S/W Interrupt
SW_INT13    B     PHANTOM13         ; User S/W Interrupt
SW_INT14    B     PHANTOM14         ; User S/W Interrupt
SW_INT15    B     PHANTOM15         ; User S/W Interrupt
SW_INT16    B     PHANTOM16         ; User S/W Interrupt
TRAP                B    PHANTOM17         ; Trap vector
NMINT               B    PHANTOM18         ; Non-maskable Interrupt
EMU_TRAP    B     PHANTOM19         ; Emulator Trap
SW_INT20    B     PHANTOM20         ; User S/W Interrupt
SW_INT21    B     PHANTOM21         ; User S/W Interrupt
SW_INT22    B     PHANTOM22         ; User S/W Interrupt
SW_INT23    B     PHANTOM23         ; User S/W Interrupt

                    .end
;-----------------------------------------------------------------------
```

```
;-----------------------------------------------------------------------;
;          File :          TRANS.ASM
;
;          Orginator :     R. Haldemann / M. Voser
;
;          Date :          26.11.98
;
;          Version:        1.0
;
;          Target System : C24x Evaluation Board
;
;-----------------------------------------------------------------------
;  Modification: Abdurrahman Unsal
;-----------------------------------------------------------------------
;  Description:
;  This library contains following modules :
;
;                      - sinus approximation              'sin'
;                      - cosinus approximation                      'cos'
;                      - magnitude                                  'mag'
;                      - square root extractor            [2.14]    'sqrt'
;                      - square root extractor [3.13]     'sqrt2'
;  All modules are declared as a macro and thus are only assembled when
;  they are used.
;
;***********************************************************************
;  sin (macro)                                  ISB, 17.8.98 Reto Haldemann
;
;
;  Description:
;   sinus-function y=sin(x)
;
;  Call-parameter :
;    nr                                                    number of task
;    x                                          variable
;
;  Return-parameter :
;    y                                                     variable
;
;  Calling Convention:
;
;  Variables                      on Entry                              on Exit
;  ------------------------------------------------------------------
;   DP                            XX                                    task nr
;   ARP                           XX                                    xx
;   ACC                           XX                                    XX
;   nr_sin             Task nummer                          un-touched
;   x_sin [1.15]       value in special degrees             un-touched
;   y_sin [1.15]       XX                                   touched
;  ------------------------------------------------------------------
;
;  Note : nr contains the number of the task
;
;***********************************************************************

sin         .macro      nr_sin, x_sin, y_sin
sin?:
            CLRC        OVM                     ; disable overflow mode
            SETC        SXM                     ;Enable sign extension
                                                ;begin temp_tx_2 = sin(phi)
            LDP         #x_sin                  ;DP for Global variables
            LAC         x_sin                   ;Load phi in ACC
            ADD         #4000h
            BCND        sin_if2?,LT             ;if2 x =< -0.5
            ADD         #8000h
            BCND        sin_if3?,GEQ            ;if3 x > 0.5

            LAC         x_sin                   ;else if3
            B           sin_end2?               ;end if3

sin_if3?:
            LAC         x_sin                   ;Load x in ACC
            ADD         #8000h                  ;then if3
            NEG                                 ;x=1 - x
            B           sin_end2?               ;end if2

sin_if2?:
            LAC         #8000h                  ;if2
```

```
            SUB         x_sin                       ;x= -1 - x

sin_end2?:
            LDP         #temp_t:nr_sin:_1           ;DP for Temp variables
            SACL        temp_t:nr_sin:_1
            sin90       temp_t:nr_sin:_1, temp_t:nr_sin:_2, temp_t:nr_sin:_3
                                                    ;temp_t:nr_sin:_2 = y = sin(x) =
sin(temp_t:nr_dq:_1)
            BLDD    temp_t:nr_sin:_2,#y_sin         ; y = sin(x)
            .endm


;***********************************************************************
;  cos (macro)                                  ISB, 31.8.98 Reto Haldemann
;
;
;  Description:
;   cosinus-function y=cos(x)
;
;  Call-parameter :
;    nr                                         number of task
;    x                                          variable
;
;  Return-parameter :
;    y                                          variable
;
;  Calling Convention:
;
;  Variables                    on Entry              on Exit
;  ---------------------------------------------------------------------
;   DP                          XX                    task nr
;   ARP                         XX                    xx
;   ACC                         XX                    XX
;   nr_cos              Task nummer                   un-touched
;   x_cos [1.15]        value in special degrees      un-touched
;   y_cos [1.15]        XX                            touched
;  ---------------------------------------------------------------------
;
;  Note : nr contains the number of the task
;
;***********************************************************************

cos         .macro      nr_cos, x_cos, y_cos
cos?:
            CLRC        OVM                     ; disable overflow mode
            SETC        SXM                     ;Enable sign extension
                                                ;begin temp_tx_2 = cos(x)
            LDP         #x_cos                  ;DP for Global variables
            LAC         x_cos                   ;Load phi in ACC
            BCND        cos_if1?,LT             ;if x==neg
            SUB         #4000h
            NEG                                 ;x=0.5 - x
            B           cos_end1?               ;end if1
cos_if1?:
            ADD         #4000h                  ;else if1  x=x - 0.5
cos_end1?:
            LDP         #temp_t:nr_cos:_1       ;DP for Temp variables
            SACL        temp_t:nr_cos:_1
            sin90       temp_t:nr_cos:_1, temp_t:nr_cos:_2, temp_t:nr_cos:_3
                                                ;temp_t:nr_cos:_2 = y = sin(x) =
sin(temp_t:nr_cos:_1)

            BLDD    temp_t:nr_cos:_2,#y_cos     ; y = cos(x)
            .endm

;***********************************************************************
;  sqrt (macro)                                 ISB, 17.12.98 Reto Haldemann
;
;
;  Description:
;    call y=sqrt(x), 0 < x < 2
;
;  Call-parameter :
;    x                                          temp variable or variable
;
;  Return-parameter :
;    y                                          temp variable or variable
;
```

```
;  Calling Convention:
;
;  Variables                    on Entry              on Exit
;  ------------------------------------------------------------------
;   DP                          tl_var               tl_var
;   ARP                         XX                   xx
;   ACC                         XX                   XX
;   nr_sqrt          Task nummer                     un-touched
;   x_sqrt [2.14]    Input value                     un-touched
;   y_sqrt [2.14]    XX                              touched
;  ------------------------------------------------------------------
;
;  Note : nr contains the number of the task
;
;*********************************************************************

sqrt        .macro      nr_sqrt, x_sqrt, y_sqrt
sqrt?:
            LAR     AR2,#temp_t:nr_sqrt:_2
            LAR     AR3,#temp_t:nr_sqrt:_3
            MAR     *,AR3                           ; ARP = 3
            LDP         #x_sqrt
            LACC        x_sqrt
; ADD BY UNSAL
        BCND zero_input?,EQ
            ABS
        SACL            *                           ; temp3 = x0
            SUB         #0CCDh
            BGZ         then_sqrt?
            LACC        *,1,AR2                     ; x <= 0.2      ARP=2
            ADD         #05C3h
            SACL        *,AR3                       ; temp2 = 2*x0 + 0.09 ARP=3
            B           end_sqrt?
then_sqrt?:                                         ; x > 0.2
            LACC        *,15,AR2                    ;   ARP=2
            ADD         #6666h,15
            SACH        *,AR3                       ; temp2 = x0/2 + 0.8 ARP=3
end_sqrt?:
            LACC        *,14,AR2                    ;   ARP=2
            RPT         #15                         ; ACCL = x0/temp2
        SUBC            *
        ADDS            *                           ; ACCL = temp2 + x0/temp2
            AND         #0FFFFh
            SFR
            SACL        *,AR3                       ; temp2 = ACCL/2        ARP=3

            LACC        *,14,AR2                    ;   ARP=2
            RPT         #15                         ; ACCL = x0/temp2
        SUBC            *
        ADDS            *                           ; ACCL = temp2 + x0/temp2
            AND         #0FFFFh
        SFR
; Added by Unsal to get the correct magnitude in fft
        SFL
        SFL

;------------------
; The output is in (16,6) format)
; This label used when zero input
zero_input?:
            LDP         #y_sqrt
            SACL        y_sqrt                      ; y_sqrt = ACCL/2
            .endm


;*********************************************************************
;  sqrt2 (macro)                              ISB, 17.12.98 Reto Haldemann
;
;
;  Description:
;    call y=sqrt(x), 0 < x < 4
;
;  Call-parameter :
;    x                                  temp variable or variable
;
;  Return-parameter :
;    y                                  temp variable or variable
```

```
;
;  Calling Convention:
;
;  Variables                        on Entry              on Exit
;  ------------------------------------------------------------------
;   DP                              tl_var                tl_var
;   ARP                             XX                    xx
;   ACC                             XX                    XX
;   nr_sqrt2                        Task nummer           un-touched
;   x_sqrt2 [3.13]      Input value                       un-touched
;   y_sqrt2 [2.14]      XX                                touched
;  ------------------------------------------------------------------
;
;  Note : nr contains the number of the task
;
;*********************************************************************

sqrt2       .macro      nr_sqrt2, x_sqrt2, y_sqrt2
sqrt2?:
            LAR     AR2,#temp_t:nr_sqrt2:_2
            LAR     AR3,#temp_t:nr_sqrt2:_3
            MAR     *,AR3                           ; ARP = 3
            LDP         #x_sqrt2
            LACC        x_sqrt2
            ABS
         SACL           *                           ; temp3 = x0
            SUB         #0400h
            BGZ         then_sqrt2?
            LACC        *,2,AR2                     ; x <= 0.2 ARP=2
            ADD         #051Fh
            SACL        *,AR3                       ; temp2 = 2*x0 + 0.09 ARP=3
            B           end_sqrt2?
then_sqrt2?:                                        ; x > 0.2
            LACC        *,14                        ;       ARP=2
            ADD         *,13,AR2                    ; ACC = 3*x0
            ADD         #1FECh,15
            SACH        *,1,AR3                     ; temp2 = x0*3/8 + 1.36 ARP=3
end_sqrt2?:
            LACC        *,15,AR2                    ;  ARP=2
            RPT         #15                         ; ACCL = x0/temp2
         SUBC           *
         ADDS           *                           ; ACCL = temp2 + x0/temp2
            AND         #0FFFFh
            SFR
            SACL        *,AR3                       ; temp2 = ACCL/2 ARP=3

            LACC        *,15,AR2                    ;     ARP=2
            RPT         #15                         ; ACCL = x0/temp2
         SUBC           *
         ADDS           *                           ; ACCL = temp2 + x0/temp2
            AND         #0FFFFh
         SFR
            LDP         #y_sqrt2
            SACL        y_sqrt2                     ; y_sqrt = ACCL/2
            .endm



;*********************************************************************
;  mag (macro)                          ISB, 17.12.98 Reto Haldemann
;
;
;  Description:
;   magnitud from recangular coordinate
;   call A = sqrt( x^2 + y^2 )
;
;  Call-parameter :
;    nr                                 number of task
;    x                                  variable
;    y                                  variable
;
;  Return-parameter :
;    A                                  variable
;
;  Calling Convention:
;
;  Variables                    on Entry    on Exit
```

```
;  ------------------------------------------------------------------
;   DP                              XX          task nr
;   ARP                             XX          xx
;   ACC                             XX          XX
;   nr_mag              Task nummer             un-touched
;   x_mag [2.14]        x-coordinate            un-touched
;   y_mag [2.14]        y-coordinate            un-touched
;   A_mag [2.14]        XX                      touched
;  ------------------------------------------------------------------
;
;  Note : nr contains the number of the task
;
;*****************************************************************************

mag         .macro      nr_mag, x_mag, y_mag, A_mag
mag?:
            SETC        SXM                         ; Enable sign extension
            LDP         #x_mag                      ; DP for Global variables
            SQRA        x_mag                       ; P_Reg = x^2
         ZAC                                        ; ACC = 0
            SQRA        y_mag                       ; ACC = x^2       P_Reg = y^2
            APAC                                    ; ACC = x^2 + y^2
            LDP         #temp_t:nr_mag:_1           ; DP for Temp variables
            SETC        OVM                         ; Enable sign extension
         ABS
            SUB         #8000h,15
            ADD      #8000h,15
            CLRC        OVM                         ; Enable sign extension
            SACH        temp_t:nr_mag:_1,1          ; temp1 = x^2 + y^2
            sqrt2       nr_mag, temp_t:nr_mag:_1, A_mag  ; A_mag = sqrt( x^2 + y^2 )
            .endm

;*****************************************************************************
;  sqrt (macro)                                     ISB, 17.12.98 Reto Haldemann
;
;
;  Description:
;    call y=sqrt(x), 0 < x < 2
;
;  Call-parameter :
;    x                                  temp variable or variable
;
;  Return-parameter :
;    y                                  temp variable or variable
;
;  Calling Convention:
;
;  Variables                         on Entry              on Exit
;  ------------------------------------------------------------------
;   DP                               t1_var                t1_var
;   ARP                              XX                    xx
;   ACC                              XX                    XX
;   nr_sqrt             Task nummer                        un-touched
;   x_sqrt [2.14]       Input value                        un-touched
;   y_sqrt [2.14]       XX                                 touched
;  ------------------------------------------------------------------
;
;  Note : nr contains the number of the task
;
;*****************************************************************************
sqrt_Ia1    .macro      nr_sqrt, x_sqrt, y_sqrt
sqrt?:
            LAR      AR2,#temp_t:nr_sqrt:_2
            LAR      AR3,#temp_t:nr_sqrt:_3
            MAR      *,AR3                          ; ARP = 3
            LDP         #x_sqrt
            LACC        x_sqrt

; ADD By UNSAL
;         BCND zero_inputted?,EQ

            ABS
         SACL           *                           ; temp3 = x0
            SUB         #0CCDh
            BGZ         then_sqrt?
            LACC        *,1,AR2                     ; x <= 0.2       ARP=2
            ADD         #05C3h
            SACL        *,AR3                       ; temp2 = 2*x0 + 0.09       ARP=3
```

```
            B           end_sqrt?
then_sqrt?:
            LACC        *,15,AR2                    ; x > 0.2
            ADD         #6666h,15                   ;          ARP=2
            SACH        *,AR3                       ; temp2 = x0/2 + 0.8        ARP=3
end_sqrt?:
            LACC        *,14,AR2                    ;                           ARP=2
            RPT         #15                         ; ACCL = x0/temp2
        SUBC            *
        ADDS            *                           ; ACCL = temp2 + x0/temp2
            AND         #0FFFFh
            SFR
            SACL        *,AR3                        ; temp2 = ACCL/2          ARP=3

            LACC        *,14,AR2                     ;              ARP=2
            RPT         #15                          ; ACCL = x0/temp2
        SUBC            *
        ADDS            *                           ; ACCL = temp2 + x0/temp2
            AND         #0FFFFh
        SFR
; Added by Unsal to get the correct magnitude in fft
        ;SFL
        ;SFL
        ;SFR
        ;SFR
        SFR
        SFR
        ;SFR
;------------------
; The output is in (16,6) format)
; USED FOR ZERO INPUT
;zero_inputted?:
            LDP         #y_sqrt
            SACL        y_sqrt                      ; y_sqrt = ACCL/2
            .endm
;-------------------------------------------------------------------------
```

```
;-------------------------------------------------------------------------
;               LINEAR.ASM
;-------------------------------------------------------------------------
;
;          File :          LINEAR.ASM
;
;          Orginator :     R. Haldemann / M. Voser
;
;          Date :          26.11.98
;
;          Version:        1.0
;
;          Target System : C24x Evaluation Board
;
;-------------------------------------------------------------------------
;  Modification: Abdurrahman Unsal
;-------------------------------------------------------------------------
;  Description:
;
;  This library contains following modules :
;
;                      - gain a signal                              'gain'
;                      - multiply  2 signals                        'mult2'
;                      - add 2 signals                              'add2'
;                      - add 3     signals                          'add3'
;                      - sub a signal                               'sub2'
;                      - absolut value from a signal                'abs1'
;                      - negate the signal                          'neg1'
;                      - divides two signals                        'div2'
;
;  All modules are declared as a macro and thus are only assembled when
;  they are used.
;
;-------------------------------------------------------------------------
; constants and parameter
;-------------------------------------------------------------------------
; _c_sqrt3_div_2:.equ  6EDAh  ; constant sqrt(3)/2 in 1.15-format


;*************************************************************************
;  gain (macro)                                ISB, 11.11.98 Reto Haldemann
;
;
;  Description:
;    y = K * x
;
;  Call-parameter :
;    K                                          parameter
;    x                                          signal
;
;  Return-parameter :
;    y                                          signal
;
;  Calling Convention:
;
;  Variables                        on Entry    on Exit
;  ------------------------------------------------------------------
;   DP                              XX          signals
;   ARP                             XX          xx
;   ACC                             XX          XX
;   x_gain [2.14]      Input                    un-touched
;   K_gain [7.9]        factor                  un-touched
;   y_gain [2.14]      xx                       touched
;  ------------------------------------------------------------------
;
;*************************************************************************

gain       .macro     x_gain ,k_gain ,y_gain
gain?:
           SOVM                                ; Set Overflow mode
           CLRC       SXM
           LDP        #k_gain                  ;
           LT         k_gain                   ; factor [7.9]
           LDP        #x_gain                  ;
           MPY        x_gain                   ; factor [7.9] * Input [2.14]
           PAC                                 ; Output [9.23]
           ADD     #0FE00h,15                  ;
           SUB     #0FE00h,15                  ; restrict
```

```
            SUB     #0FE00h,15              ;
            ADD     #0FE00h,15              ;
            SACH        y_gain,7            ; Output [2.14]
            CLRC        OVM                 ; disable overflow mode
            SETC        SXM
            .endm

;*********************************************************************
;  mult2 (macro)                            ISB, 11.11.98 Reto Haldemann
;  -----
;
;  Description:
;    y = x1 * x2
;
;  Call-parameter :
;    x1                                      signal
;    x2                                      signal
;
;  Return-parameter :
;    y                                       signal
;
;  Calling Convention:
;
;  Variables                        on Entry    on Exit
;  ------------------------------------------------------------------
;   DP                              XX          signals
;   ARP                             XX          xx
;   ACC                             XX          XX
;   x1_mult [2.14]      Input                   un-touched
;   x2_mult [2.14]       Input                  un-touched
;   y_mult [2.14]       xx                      touched
;  ------------------------------------------------------------------
;
;*********************************************************************

mult2       .macro      x1_mult ,x2_mult ,y_mult
mult2?:
            SOVM                                ; Set Overflow mode
            CLRC        SXM
            LDP         #x1_mult                ;
            LT          x1_mult                 ; factor [7.9]
       ; put by Unsal
            LDP         #x2_mult                ;
            MPY         x2_mult                 ; factor [7.9] * Input [2.14]
            PAC                                 ; Output [9.23]
            ADD     #0C000h,15                  ;
            SUB     #0C000h,15                  ; restrict
            SUB     #0C000h,15                  ;
            ADD     #0C000h,15                  ;

       ; put by Unsal
            LDP         #y_mult                 ;
;           SACH        y_mult,2                ; Output [2.14]
            SACH        y_mult,3                ; Output [2.14]
;           SACH        y_mult,4                ; Output [2.14]
;           SACH        y_mult,5                ; Output [2.14]

            CLRC        OVM                     ; disable overflow mode
            SETC        SXM
            .endm


;*********************************************************************
;  add2 / add3 (macro)                      ISB, 12.11.98 Reto Haldemann
;  ----   ----
;
;  Description:
;    y = x1 + x2 ( + x3 )
;
;  Call-parameter :
;    x1, x2                          signal
;
;  Return-parameter :
;    y                                       signal
;
;  Calling Convention:
;
```

```
;  Variables                          on Entry    on Exit
;  ---------------------------------------------------------------------
;   DP                                XX          signals
;   ARP                               XX          xx
;   ACC                               XX          XX
;   x1_add2 [2.14]      Input                     un-touched
;   x2_add2 [2.14]       Input                    un-touched
;   x3_add2 [2.14]       Input                    un-touched
;   y_gain [2.14]       xx                        touched
;  ---------------------------------------------------------------------
;
;*********************************************************************

add2        .macro      x1_add2 ,x2_add2 ,y_add2
add2?:
            SOVM                                  ; Set Overflow mode
            SETC        SXM
            LDP         #x1_add2                  ;
            LAC         x1_add2,16                ; Input [2.14]
            ADDH        x2_add2
            SACH        y_add2                    ; Output [2.14]
            CLRC        OVM                       ; disable overflow mode
            .endm


add3        .macro      x1_add3 ,x2_add3 ,x3_add3 ,y_add3
add3?:
            SOVM                                  ; Set Overflow mode
            SETC        SXM
            LDP         #x1_add3                  ;
            LAC         x1_add3,16                ; Input [2.14]
            ADDH        x2_add3
            ADDH        x3_add3
            SACH        y_add3                    ; Output [2.14]
            CLRC        OVM                       ; disable overflow mode
            .endm


;*********************************************************************
;  sub2  (macro)                              ISB, 12.11.98 Reto Haldemann
;  ----
;
;  Description:
;    y = x1 - x2
;
;  Call-parameter :
;    x1, x2                          signal
;
;  Return-parameter :
;    y                               signal
;
;  Calling Convention:
;
;  Variables                         on Entry              on Exit
;  ---------------------------------------------------------------------
;   DP                               XX                    signals
;   ARP                              XX                    xx
;   ACC                              XX                    XX
;   x1_sub2 [2.14]      Input                              un-touched
;   x2_sub2 [2.14]       Input                             un-touched
;   y_sub2 [2.14]       xx                                 touched
;  ---------------------------------------------------------------------
;
;*********************************************************************

sub2        .macro      x1_sub2,x2_sub2,y_sub2
sub2?:
            SOVM                                          ; Set Overflow mode
            SETC        SXM
            LDP         #x1_sub2                          ;
            LAC         x1_sub2,16                        ; Input [2.14]
        SUBH            x2_sub2
            SACH        y_sub2                            ; Output [2.14]
            CLRC        OVM                               ; disable overflow mode
            .endm


;*********************************************************************
```

```
;  absl  (macro)                          ISB, 12.11.98 Reto Haldemann
;  ----
;
;  Description:
;    y = | x1 |
;
;  Call-parameter :
;    x1                                  signal
;
;  Return-parameter :
;    y                                   signal
;
;  Calling Convention:
;
;  Variables                             on Entry              on Exit
;  ------------------------------------------------------------------
;   DP                                   XX                    signals
;   ARP                                  XX                    xx
;   ACC                                  XX                    XX
;   x1_abs [2.14]       Input                                  un-touched
;   y_abs [2.14]        xx                                     touched
;  ------------------------------------------------------------------
;
;*********************************************************************

absl        .macro      x1_abs ,y_abs
absl?:
        SOVM                                                   ; set overflow mode
            LDP         #x1_abs                                ;
            LAC         x1_abs,16                              ; Input [2.14]
        ABS
            SACH        y_abs                                  ; Output [2.14]
            .endm

;*********************************************************************
;  negl  (macro)                        ISB, 12.11.98 Reto Haldemann
;  ----
;
;  Description:
;    y = - x1
;
;  Call-parameter :
;    x1                                  signal
;
;  Return-parameter :
;    y                                   signal
;
;  Calling Convention:
;
;  Variables                             on Entry              on Exit
;  ------------------------------------------------------------------
;   DP                                   XX                    task nr
;   ARP                                  XX                    xx
;   ACC                                  xx                    xx
;   x1_neg [2.14]       Input                                  un-touched
;   y_neg [2.14]        xx                                     touched
;  ------------------------------------------------------------------
;
;*********************************************************************

negl        .macro      x1_neg ,y_neg
negl?:
        SOVM                                                   ; set overflow mode
            LDP         #x1_neg                                ;
            LAC         x1_neg,16                              ; Input [2.14]
        NEG
            SACH        y_neg                                  ; Output [2.14]
            .endm


;*********************************************************************
;  div2 (macro)                         ISB, 11.12.98 Reto Haldemann
;  ----
;
;  Description:
;    y = x1 / x2
;
;  Call-parameter :
```

```
;    nr                          task number
;    x1                        signal
;    x2                        signal
;
;  Return-parameter :
;    y                         signal
;
;  Calling Convention:
;
;  Variables                     on Entry              on Exit
;  ------------------------------------------------------------------
;   DP                           XX                    signals
;   ARP                          XX                    xx
;   ACC                          XX                    XX
;   x1_div2 [2.14]     Input                           un-touched
;   x2_div2 [2.14]      Input                          un-touched
;   y_div2 [2.14]      xx                              touched
;  ------------------------------------------------------------------
;
;*********************************************************************

div2        .macro      nr_div, x1_div ,x2_div ,y_div
div2?:
            SETC        SXM
            SOVM                                        ; Set Overflow mode
            LAR   AR3,#temp_t:nr_div:_1
        MAR   *,AR3

            LDP         #x1_div                         ;
        LACC            x2_div,16
            BGEZ        div_if1?
            ABS                                         ; x2 < 0
            SACH        *
            LACC        x1_div,14
            BGEZ        div_then1?
            ABS                                         ; x2 < 0 and x1 < 0
        B   div_else1?

div_if1?:                                               ; x2 >= 0
            SACH        *
            LACC    x1_div,14
            BGEZ        div_else1?
            ABS                                        ; x2 >= and x1 < 0

div_then1?:                                            ; x1/x2 < 0
            RPT         #15
            SUBC        *
            SACL        *
            LACC        *,16
            CLRC        SXM
            SFL
            SFR
            NEG

;  add by unsal
        LDP #y_div

            SACH        y_div
            B           div_end?

div_else1?:                                     ; x1/x2 >= 0
            RPT         #15
        SUBC            *
            SACL        *
            LACC        *,16
            CLRC        SXM
            ;SFL
            SFR
;   add by Unsal
        SFR

;  add by unsal
        LDP #y_div

            SACH        y_div
div_end?:
            .endm
;-------------------------------------------------------------------
```

```
;-----------------------------------------------------------------
;               SINUSOIDAL TABLES, Fundamental Signal
;-----------------------------------------------------------------

  .data
  .word    16000                                    .word    16000
  .word    28851                                    .word     3149
  .word    31311                                    .word      689
  .word    21390                                    .word    10610
  .word     7111                                    .word    24889
  .word       19                                    .word    31981
  .word     5850                                    .word    26150
  .word    19888                                    .word    12112
  .word    30782                                    .word     1218
  .word    29724                                    .word     2276
  .word    17568                                    .word    14432
  .word     4145                                    .word    27855
  .word      307                                    .word    31693
  .word     9159                                    .word    22841
  .word    23542                                    .word     8458
  .word    31827                                    .word      173
  .word    27314                                    .word     4686
  .word    13652                                    .word    18348
  .word     1889                                    .word    30111
  .word     1536                                    .word    30464
  .word    12879                                    .word    19121
  .word    26745                                    .word     5255
  .word    31923                                    .word       77
  .word    24226                                    .word     7774
  .word     9877                                    .word    22123
  .word      479                                    .word    31521
  .word     3632                                    .word    28368
  .word    16785                                    .word    15215
  .word    29304                                    .word     2696
  .word    31065                                    .word      935
  .word    20645                                    .word    11355
  .word     6469                                    .word    25531
  .word        0                                    .word    32000
  .word     6469                                    .word    25531
  .word    20645                                    .word    11355
  .word    31065                                    .word      935
  .word    29304                                    .word     2696
  .word    16785                                    .word    15215
  .word     3632                                    .word    28368
  .word      479                                    .word    31521
  .word     9877                                    .word    22123
  .word    24226                                    .word     7774
  .word    31923                                    .word       77
  .word    26745                                    .word     5255
  .word    12879                                    .word    19121
  .word     1536                                    .word    30464
  .word     1889                                    .word    30111
  .word    13652                                    .word    18348
  .word    27314                                    .word     4686
  .word    31827                                    .word      173
  .word    23542                                    .word     8458
  .word     9159                                    .word    22841
  .word      307                                    .word    31693
  .word     4145                                    .word    27855
  .word    17568                                    .word    14432
  .word    29724                                    .word     2276
  .word    30782                                    .word     1218
  .word    19888                                    .word    12112
  .word     5850                                    .word    26150
  .word       19                                    .word    31981
  .word     7111                                    .word    24889
  .word    21390                                    .word    10610
  .word    31311                                    .word      689
  .word    28851                                    .word     3149
;-----------------------------------------------------------------
;               SINUSOIDAL TABLES, Fifth Harmonic signal
;-----------------------------------------------------------------

  .data
  .word    24001                                    .word    26353
  .word    30682                                    .word    17047
  .word    31584                                    .word     7329
                                                    .word     1024
```

```
  .word      613
  .word     6259
  .word    15738
  .word    25321
  .word    31235
  .word    31153
  .word    25107
  .word    15476
  .word     6052
  .word      543
  .word     1118
  .word     7550
  .word    17308
  .word    26551
  .word    31641
  .word    30576
  .word    23773
  .word    13911
  .word     4871
  .word      212
  .word     1766
  .word     8922
  .word    18865
  .word    27679
  .word    31897
  .word    29858
  .word    22365
  .word    12366
  .word     3798
  .word       32
  .word     2551
  .word    10363
  .word    20394
  .word    28695
  .word    32000
  .word    29007
  .word    20895
  .word    10856
  .word     2842
  .word        6
  .word     3466
  .word    11858
  .word    21881
  .word    29589
  .word    31949
  .word    28031
  .word    19378
  .word     9396
  .word     2013
  .word      135
  .word     4501
  .word    13393
  .word    23311
  .word    30352
  .word    31744
  .word    26939
  .word    17829
  .word     7999
  .word     1318
  .word      416
  .word     5647
  .word    14953
  .word    24671
  .word    30976
  .word    31387
  .word    25741
  .word    16262
  .word     6679
  .word      765
  .word      847
  .word     6893
  .word    16524
  .word    25948
  .word    31457
  .word    30882
  .word    24450
  .word    14692
  .word     5449
  .word      359
  .word     1424
  .word     8227
  .word    18089
  .word    27129
  .word    31788
  .word    30234
  .word    23078
  .word    13135
  .word     4321
  .word      103
  .word     2142
  .word     9635
  .word    19634
  .word    28202
  .word    31968
  .word    29449
  .word    21637
  .word    11606
  .word     3305
  .word        0
  .word     2993
  .word    11105
  .word    21144
  .word    29158
  .word    31994
  .word    28534
  .word    20142
  .word    10119
  .word     2411
  .word       51
  .word     3969
  .word    12622
  .word    22604
  .word    29987
  .word    31865
  .word    27499
  .word    18607
  .word     8689
  .word     1648
  .word      256
  .word     5061
  .word    14171

;-------------------------------------------------------------------
;              SINUSOIDAL TABLES, Seventh Harmonic signal
;-------------------------------------------------------------------
  .data
  .word    16000
  .word    24226
  .word    30111
  .word    31981
  .word    29304
  .word    22841
  .word    14432
  .word     6469
  .word     1218
  .word      173
  .word     3632
  .word    10610
  .word    19121
  .word    26745
  .word    31311
  .word    31521
  .word    27314
  .word    19888
  .word    11355
  .word     4145
  .word      307
  .word      935
  .word     5850
  .word    13652
  .word    22123
```

```
  .word    28851                                .word     5255
  .word    31923                                .word      689
  .word    30464                                .word      479
  .word    24889                                .word     4686
  .word    16785                                .word    12112
  .word     8458                                .word    20645
  .word     2276                                .word    27855
  .word        0                                .word    31693
  .word     2276                                .word    31065
  .word     8458                                .word    26150
  .word    16785                                .word    18348
  .word    24889                                .word     9877
  .word    30464                                .word     3149
  .word    31923                                .word       77
  .word    28851                                .word     1536
  .word    22123                                .word     7111
  .word    13652                                .word    15215
  .word     5850                                .word    23542
  .word      935                                .word    29724
  .word      307                                .word    32000
  .word     4145                                .word    29724
  .word    11355                                .word    23542
  .word    19888                                .word    15215
  .word    27314                                .word     7111
  .word    31521                                .word     1536
  .word    31311                                .word       77
  .word    26745                                .word     3149
  .word    19121                                .word     9877
  .word    10610                                .word    18348
  .word     3632                                .word    26150
  .word      173                                .word    31065
  .word     1218                                .word    31693
  .word     6469                                .word    27855
  .word    14432                                .word    20645
  .word    22841                                .word    12112
  .word    29304                                .word     4686
  .word    31981                                .word      479
  .word    30111                                .word      689
  .word    24226                                .word     5255
  .word    16000                                .word    12879
  .word     7774                                .word    21390
  .word     1889                                .word    28368
  .word       19                                .word    31827
  .word     2696                                .word    30782
  .word     9159                                .word    25531
  .word    17568                                .word    17568
  .word    25531                                .word     9159
  .word    30782                                .word     2696
  .word    31827                                .word       19
  .word    28368                                .word     1889
  .word    21390                                .word     7774
  .word    12879
;-------------------------------------------------------------------------
;                SINUSOIDAL TABLES, Eleventh Harmonic signal
;-------------------------------------------------------------------------
.data                                           .word    31311
  .word    16000                                .word    31981
  .word    21390                                .word    30782
  .word    26150                                .word    27855
  .word    29724                                .word    23542
  .word    31693                                .word    18348
  .word    31827                                .word    12879
  .word    30111                                .word     7774
  .word    26745                                .word     3632
  .word    22123                                .word      935
  .word    16785                                .word        0
  .word    11355                                .word      935
  .word     6469                                .word     3632
  .word     2696                                .word     7774
  .word      479                                .word    12879
  .word       77                                .word    18348
  .word     1536                                .word    23542
  .word     4686                                .word    27855
  .word     9159                                .word    30782
  .word    14432                                .word    31981
  .word    19888                                .word    31311
  .word    24889                                .word    28851
  .word    28851                                .word    24889
```

```
  .word    19888
  .word    14432
  .word     9159
  .word     4686
  .word     1536
  .word       77
  .word      479
  .word     2696
  .word     6469
  .word    11355
  .word    16785
  .word    22123
  .word    26745
  .word    30111
  .word    31827
  .word    31693
  .word    29724
  .word    26150
  .word    21390
  .word    16000
  .word    10610
  .word     5850
  .word     2276
  .word      307
  .word      173
  .word     1889
  .word     5255
  .word     9877
  .word    15215
  .word    20645
  .word    25531
  .word    29304
  .word    31521
  .word    31923
  .word    30464
  .word    27314
  .word    22841
  .word    17568
  .word    12112
  .word     7111
  .word     3149
  .word      689
;-----------------------------------------------------------------------
;              SINUSOIDAL TABLES, Thirteenth Harmonic signal
;-----------------------------------------------------------------------
.data
  .word    16000
  .word    19888
  .word    23542
  .word    26745
  .word    29304
  .word    31065
  .word    31923
  .word    31827
  .word    30782
  .word    28851
  .word    26150
  .word    22841
  .word    19121
  .word    15215
  .word    11355
  .word     7774
  .word     4686
  .word     2276
  .word      689
  .word       19
  .word      307
  .word     1536
  .word     3632
  .word     6469
  .word     9877
  .word    13652
  .word    17568
  .word    21390
  .word    24889
  .word    27855
  .word    30111
  .word    31521
  .word       19
  .word     1218
  .word     4145
  .word     8458
  .word    13652
  .word    19121
  .word    24226
  .word    28368
  .word    31065
  .word    32000
  .word    31065
  .word    28368
  .word    24226
  .word    19121
  .word    13652
  .word     8458
  .word     4145
  .word     1218
  .word       19
  .word      689
  .word     3149
  .word     7111
  .word    12112
  .word    17568
  .word    22841
  .word    27314
  .word    30464
  .word    31923
  .word    31521
  .word    29304
  .word    25531
  .word    20645
  .word    15215
  .word     9877
  .word     5255
  .word     1889
  .word      173
  .word      307
  .word     2276
  .word     5850
  .word    10610
  .word    16000
  .word    21390
  .word    26745
  .word    30782
  .word    31827
  .word    31923
  .word    31065
  .word    29304
  .word    26745
  .word    23542
  .word    19888
  .word    16000
```

```
.word    12112
.word     8458
.word     5255
.word     2696
.word      935
.word       77
.word      173
.word     1218
.word     3149
.word     5850
.word     9159
.word    12879
.word    16785
.word    20645
.word    24226
.word    27314
.word    29724
.word    31311
.word    31981
.word    31693
.word    30464
.word    28368
.word    25531
.word    22123
.word    18348
.word    14432
.word    10610
.word     7111
.word     4145
.word     1889
.word      479
.word        0
```

```
.word      479
.word     1889
.word     4145
.word     7111
.word    10610
.word    14432
.word    18348
.word    22123
.word    25531
.word    28368
.word    30464
.word    31693
.word    31981
.word    31311
.word    29724
.word    27314
.word    24226
.word    20645
.word    16785
.word    12879
.word     9159
.word     5850
.word     3149
.word     1218
.word      173
.word       77
.word      935
.word     2696
.word     5255
.word     8458
.word    12112
```

APPENDIX K PAL PROGRAM LISTING

```
TITLE PWM_INTERFACE
PATTERN Design File
REVISION 1.0
AUTHOR Abdurrahman Unsal
COMPANY Oregon State University, ECE Department
DATE June 26, 2000
CHIP Decoder PAL22V10

; This program generates seven PWM signals from three PWM signals

; PIN Declarations

PIN   1    Sig1    COMB
PIN   2    Sig2    COMB
PIN   3    Sig3    COMB    ; inputs to the PAL defined as signals
PIN   12   GND             ; required to be on pin 12

PIN   23   TEST1   COMB
PIN   14   TEST2   COMB
PIN   15   TEST3   COMB
PIN   16   S1_Pos  COMB    ; output Leg1+
PIN   17   S1_Neg  COMB    ; output Leg1-
PIN   18   S2_Pos  COMB    ; output Leg2+
PIN   19   S2_Neg  COMB    ; output Leg2-
PIN   20   S3_Pos  COMB    ; output Leg3+
PIN   21   S3_Neg  COMB    ; output Leg3-
PIN   22   BYPASS  COMB    ; Bypass Switch
PIN   24   VCC             ; required to be on pin 24

EQUATIONS ; Logical expressions

S1_Pos=( (Sig1*/Sig2*/Sig3) + (Sig1*Sig2*/Sig3) );
S1_Neg=( (/Sig1*/Sig2*Sig3) + (/Sig1*Sig2*Sig3) );

S2_Pos=( (/Sig1*Sig2*/Sig3) + (/Sig1*Sig2*Sig3) );
S2_Neg=( (Sig1*/Sig2*/Sig3) + (Sig1*/Sig2*Sig3) );

S3_Pos=( (/Sig1*/Sig2*Sig3) + (Sig1*/Sig2*Sig3) );
S3_Neg=( (/Sig1*Sig2*/Sig3) + (Sig1*Sig2*/Sig3) );

BYPASS=((/S1_Pos*/S2_Pos*/S3_Pos*/S1_Neg*/S2_Neg*/S3_Neg)+(/S1_Pos*/S2_Pos*/S3_Pos*/S1_Neg*/S2_Neg*S3_Neg)+(/S1_Pos*/S
2_Pos*/S3_Pos*/S1_Neg*S2_Neg*/S3_Neg)+(/S1_Pos*/S2_Pos*/S3_Pos*/S1_Neg*S2_Neg*S3_Neg)+(/S1_Pos*/S2_Pos*/S3_Pos*S1_Neg
*/S2_Neg*/S3_Neg)+(/S1_Pos*/S2_Pos*/S3_Pos*S1_Neg*/S2_Neg*S3_Neg)+(/S1_Pos*/S2_Pos*/S3_Pos*S1_Neg*S2_Neg*/S3_Neg)+(/S1
_Pos*/S2_Pos*/S3_Pos*S1_Neg*S2_Neg*S3_Neg)+(/S1_Pos*/S2_Pos*S3_Pos*/S1_Neg*/S2_Neg*/S3_Neg)+(/S1_Pos*S2_Pos*/S3_Pos*/S
1_Neg*/S2_Neg*/S3_Neg)+(/S1_Pos*S2_Pos*S3_Pos*/S1_Neg*/S2_Neg*/S3_Neg)+(S1_Pos*/S2_Pos*/S3_Pos*/S1_Neg*/S2_Neg*/S3_Ne
g)+(S1_Pos*/S2_Pos*S3_Pos*/S1_Neg*/S2_Neg*/S3_Neg)+(S1_Pos*S2_Pos*/S3_Pos*/S1_Neg*/S2_Neg*/S3_Neg)+(S1_Pos*S2_Pos*S3_
Pos*/S1_Neg*/S2_Neg*/S3_Neg));
```