ANKARA YILDIRIM BEYAZIT UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES



DESIGN AND IMPLEMENTATION OF A 400A INDUSTRIAL WELDING MACHINE

M.Sc. Thesis by

Ertuğrul KOCAAĞA

Department of Electrical and Electronics Engineering

March, 2019

ANKARA

DESIGN AND IMPLEMENTATION OF A 400A INDUSTRIAL WELDING MACHINE

A Thesis Submitted to the

Graduate School of Natural and Applied Sciences of

Ankara Yıldırım Beyazıt University

In Partial Fulfillment of the Requirements for the Master of Science in

Electrical and Electronics Engineering, Department of Electrical and Electronics Engineering

by

Ertuğrul KOCAAĞA

March, 2019

ANKARA

M.Sc. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **''DESIGN AND IMPLEMENTATION OF 400A INDUSTRIAL WELDING MACHINE''** completed by **ERTUĞRUL KOCAAĞA** under supervision of **ASSOC. PROF. DR. ENVER ÇAVUŞ** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Enver ÇAVUŞ

Supervisor

Assoc. Prof. Dr. Sinan KIVRAK

Jury Member

Assoc. Prof. Dr. Mehmet DEMİRTAŞ

Jury Member

Prof. Dr. Ergün ERASLAN

Director

Graduate School of Natural and Applied Sciences

I hereby declare that, in this thesis which has been prepared in accordance with the Thesis Writing Manual of Graduate School of Natural and Applied Sciences,

- All data, information and documents are obtained in the framework of academic and ethical rules,
- All information, documents and assessments are presented in accordance with scientific ethics and morals,
- All the materials that have been utilized are fully cited and referenced,
- No change has been made on the utilized materials,
- All the works presented are original,

and in any contrary case of above statements, I accept to renounce all my legal rights.

Date: 20.03.2019 **Signature:……..**

Name & Surname: Ertuğrul KOCAAĞA

ACKNOWLEDGEMENTS

I would like to express my gratitude to my supervisor, Asst. Prof. Dr. Enver Çavuş for his continuous support and motivation during my study. His knowledge, invaluable experiences and priceless advices constructed the milestones of my thesis study. It was a pleasure to work with him.

I would like to express my gratitude to Dr. Gökhan Şen for his continuous support and motivation during my study. His knowledge, invaluable experiences and priceless advices constructed the milestones of my thesis study. It was a pleasure to work with him.

I would like also to thank Linkas Teknoloji for supporting my thesis study.

I express my sincere love and special thanks to my mother, father, sister and brothers for their sincere prayers, supports, courage and patience for me.

This thesis study is supported by Linkas Teknoloji AŞ.

2019, 20 March **Ertuğrul KOCAAĞA**

DESIGN AND IMPLEMENTATION OF A 400A INDUSTRIAL WELDING MACHINE

ABSTRACT

Inverter based welding machines are getting popular and favored by users due to their size, fast response and high efficiency properties. In inverter welding machines, grid signal is rectified, and the line voltage is switched at high frequencies and the power is transferred to the secondary side of a high frequency transformer. There is again rectifier circuit in secondary side. After rectification, a filter inductor is used for current regulation. The filtered voltage is applied between welding electrode and work piece. The duty cycle of the semiconductor switches in primary side adjusts the weld current.

In this thesis, 17.2 kW welding machine is designed and implemented. The designed machine is a part of project of generate new generation of welding machine in cooperation with Linkas Teknoloji AŞ.

A full-bridge topology is used, and smart control tool of simulation program is used to design the controller for welding machine. Then the controller was modelled and tested. The results of simulation were used to program the digital signal processor unit in real circuit.

Finally, experimental results are presented. Experimental results are in good agreement with simulation results.

Keywords: Arc welding machine, pulse width modulation, voltage source inverter, full bridge inverter, constant current, constant voltage.

400A ENDÜSTRİYEL KAYNAK MAKİNESİ TASARIMI VE UYGULAMASI

ÖZ

İnvertör tipi kaynak makineleri, büyüklükleri, hızlı tepkileri ve yüksek verim özellikleri nedeniyle kullanıcılar tarafından popüler hale getirilmekte ve tercih edilmektedir. İnvertör kaynak makinelerinde şebeke sinyali rektifiye edilir ve hat voltajı yüksek frekanslarda anahtarlanır. Güç yüksek frekanslı transformatörün sekonder tarafına aktarılır. Sekonder tarafta tekrar doğrultma devresi vardır. Düzeltmeden sonra, akım regülasyonu için bir filtre indüktörü kullanılır. Filtrelenmiş voltaj kaynak elektrodu ve iş parçası arasında uygulanır. Primer taraftaki yarı iletken anahtarların görev döngüsü kaynak akımını ayarlar.

Bu tezde 17.2 kW invertör tipi kaynak makinesi için simülasyon, güç ve kontrol devresi tasarımı ve deney sonuçları sunulmaktadır. Çalışma, Linkas Teknoloji AŞ ile iş birliği ile yapılmıştır.

Çalışmada tam köprü topolojisi kullanıldı ve simülasyon programının akıllı kontrol uygulaması ile kaynak makinesinin kontrolörü tasarlandı. Daha sonra kontrolör modellenmiştir ve gerçek devrede test edilmiştir.

Son olarak, deney sonuçları sunulmaktadır. Deney sonuçları simülasyon sonuçlarıyla iyi bir uyum içerisindedir.

Anahtar Kelimeler: Ark kaynak makinesi, darbe genişlik modülasyonu, tam köprü evirici, sabit akım, sabit gerilim.

CONTENTS

M.Sc. THESIS EXAMINATION RESULT FORM ii

ETHICAL DECLARATION iii

ACKNOWLEDGMENTS iv

ABSTRACT v

ÖZ vi

NOMENCLATURE x

LIST OF TABLES xi

LIST OF FIGURES xii

CHAPTER 1 - INTRODUCTION 1

1.1 Literature Review 1

1.1.1 Objectives of the Thesis 4

1.1.2 Contribution of Thesis 4

1.1.3 Outline of the Thesis 5

CHAPTER 2 - DEFINITIONS 6

2.1 Voltage Source Inverters 6

2.1.1 Half Bridge Inverter 8

2.1.2 Single Phase Full Bridge Inverter 8

2.2 Control Methods 11

2.2.1 Voltage Mode Control 12

2.2.2 Current Mode Control 12

2.3 PWM Methods 14

2.4 GMAW Modelling Methods 15

2.4.1 Load Behavior of the GMAW System 15

2.4.2 Modelling of GMAW Process 16
2.4.3 Conventional Load Regulation 18
CHAPTER 3 - DESIGN STATE 20
3.1 Power Stage Design 20
3.1.1 Transformer Design 21
3.1.2 Filter Inductor Design 23
3.1.3 IGBT Selection 23
3.1.4 Diode Selection 24
3.1.5 Loss Calculations 24
3.1.5.1 *Semiconductor Loss Calculations* 24
3.1.5.2 *Transformer Loss Calculations* 26
3.2 Input Filter Stage Design 27
3.3 Controller Stage Design 27
3.4 Driver Stage Design 32
CHAPTER 4 - SIMULATION AND CONTROLLER DESIGN 34
4.1 Open Loop Design 34
4.1.1 Full Bridge Open Loop Design 34
4.1.2 Buck Converter Open Loop Design 36
4.2 Closed Loop Design 37
4.2.1 AC Sweep 37
4.2.2 Smart Control 39
4.2.3 Analog Controller Design 40
4.2.4 Digital Controller Design 41
4.3 Arc Force Algorithm 44
4.4 Hot Start Algorithm 46

CHAPTER 5 - RESULTS **47**

5.1 Experimental Results 47

CHAPTER 6 - CONCLUSION **55**

REFERENCES **56**

APPENDICES **58**

Appendix A – Application C Code 59

CURRICULUM VITAE **102**

NOMENCLATURE

Acronyms

AC	Alternating Current
ADC	Analog to Digital Converter
CAN	Controller Area Network
CC	Constant Current
CV	Constant Voltage
DAC	Digital to Analog Converter
DC	Direct Current
FCAW	Flux Cored Arc Welding
FPGA	Field Programmable Gate Array
FRAM	Flash Random Access Memory
GMAW	Gas Metal Arc Welding
GTAW	Gas Tungsten Arc Welding
IEEE	Institute of Electrical and Electronics Engineers
IGBT	Insulated Gate Bipolar Transistor
MCU	Micro Controller Unit
MOSFET	Metal Oxide Semiconductor Field Effect Transistor
NTC	Negative Temperature Coefficient
PLC	Programmable Logic Controller
PSIM	Power Simulation
PWM	Pulse Width Modulation
SMAW	Shielded Metal Arc Welding
SPDT	Single Pole Double Throw
SPI	Serial Peripheral Interface
VMC	Voltage Mode Control
VSI	Voltage Source Inverter
ZVS	Zero Voltage Switching

LIST OF TABLES

Table 2.1 Recommended power source output type by arc welding process 18
Table 2.2 GMAW parameters 19
Table 3.1 Welding machine system parameters 22

LIST OF FIGURES

Figure 2.1 Voltage source inverter 7
Figure 2.2 Three phase two level voltage source inverter 7
Figure 2.3 Half bridge inverter 8
Figure 2.4 Full bridge inverter 9
Figure 2.5 Switching sequence of full bridge inverter 10
Figure 2.6 Secondary side of the inverter 10
Figure 2.7 Voltage mode control 12
Figure 2.8 Current mode control 13
Figure 2.9 MCU realization of current mode control 14
Figure 2.10 Scalar PWM generation 15
Figure 2.11 Electrical model of the GMAW process 16
Figure 2.12 GMAW constant voltage curve 19
Figure 3.1 System block diagram 20
Figure 3.2 Welding machine power stage design 21
Figure 3.3 Picture of the designed transformer 22
Figure 3.4 Picture of the designed filter inductor 23
Figure 3.5 Power stage schematics 27
Figure 3.6 Input filter design 27
Figure 3.7 MCU connection 28
Figure 3.8 CAN driver connection 29
Figure 3.9 Controller power supply schematics 29
Figure 3.10 Secondary current feedbacks 30
Figure 3.11 System feedbacks 31
Figure 3.12 FRAM and watchdog timer block 32
Figure 3.13 Gate driver circuit 33
Figure 4.1 Full bridge open loop schematic 34
Figure 4.2 Open loop simulation result 35
Figure 4.3 Current ripple of weld current 35
Figure 4.4 Constant PWM push-pull mode circuit 36
Figure 4.5 Buck converter current mode control open loop design 37
Figure 4.6 Buck converter AC sweep simulation 38
Figure 4.7 AC sweep parameters 38

Figure 4.8 AC sweep result 39
Figure 4.9 Smart control controller design 39
Figure 4.10 200A output current controller s domain design 40
Figure 4.11 200A output current controller analog controller design 40
Figure 4.12 Digital controller schematic of buck converter 41
Figure 4.13 Controller stage of full bridge inverter 42
Figure 4.14 350A output of full bridge simulation 44
Figure 4.15 Welding short circuits simulation 45
Figure 4.16 Arc force implementation results 45
Figure 4.17 Hot start implementation results 46
Figure 5.1 System overall look 47
Figure 5.2 MCU peak current control method 48
Figure 5.3 Pulse transformer input signals 49
Figure 5.4 Transformer primary current and transformer primary voltage 50
Figure 5.5 Welding current and voltage 51
Figure 5.6 Hot start 51
Figure 5.7 Arc force 52
Figure 5.8 Temperature distribution of welding machine 53
Figure 5.9 Power efficiency chart 53
Figure 5.10 Welded workpiece 54

CHAPTER 1

INTRODUCTION

1.1 Literature Review

History of welding can be traced back to ancient times. So far, the Bronze Age shows the earliest examples. Small gold round boxes were made by force welding lap joints concurrently. It is assumed that these boxes were made more than 2000 years ago. The Egyptians and people in the eastern Mediterranean area learned to weld pieces of iron together during the Iron Age. Many tools were found which were made around 1000 B.C. During the middle Ages, the art of blacksmithing was developed and many items of iron were created which were welded by pounding. It was not until the 19th century that welding, as we know it today was produced. Automatic welding was introduced around the 1920s. It employed bare electrode wire operated on Direct Current (DC) and employed arc voltage as the basis of controlling the feed rate. Automatic welding was developed by P.O. Nobel of the General Electric Company. It was used to build up worn motor shafts and worn lift motors. The automobile also used it to produce rear axle housings. Around the 1920s, several types of welding electrodes were produced. There was significant discussion during the 1920s about the advantage of the heavy-coated rods versus light-coated rods. The heavy-coated electrodes, which were made by extruding, were developed by Langstroth and Wunder of the A.O. Smith Company and were used by that organization in 1927. Lincoln Electric Company manufactured extruded electrode rods that were sold to the public in 1929. Covered electrodes were widely used by 1930s. Welding codes emerged which demanded higher-quality weld metal, which improved the use of coated electrodes. Gas Tungsten Arc Welding (GTAW) had its origins from a design to weld in a nonoxidizing gas atmosphere. The theory was moreover improved in the late 1920s by H. M. Hobart, who used helium for shielding, and P.K. Devers, who used argon. This method was perfect for welding magnesium and for welding stainless and aluminum. It was later patented by Linde Air Products, where the water-cooled torch was developed. The GTAW process has become one of the most important welding processes of all.

Gas Metal Arc Welding (GMAW) process, under the sponsorship of the Air Reduction Company was successfully developed at Battelle Memorial Institute. This extension employed the gas shielded arc like the gas tungsten arc but replaced the tungsten electrode with a continuously fed electrode wire. One of the basic differences that made the process more convenient was the small-diameter electrode wires and the constant-voltage power source. This opinion had been licensed earlier by H. E. Kennedy. The initial presentation of GMAW was for welding nonferrous metals. The high deposition velocity directed users to try the method on steel. The cost of inert gas was comparatively high, and the cost savings were not instantly possible. In 1953 it was announced that the use of welding with consumable electrodes in an atmosphere of CO_2 gas. The CO_2 welding process immediately gained favor since it used tools formed for GMAW but may now be used for economically welding irons. The CO_2 arc is a hot arc and the bigger electrode wires needed high currents. The process developed widely used with the foundation of smaller-diameter electrode wires and strained power supplies later. This correction provided all-position welding on thin metals and soon became the most popular of the GMAW process modifications of all. One of the modifications was the use of inert gas with small amounts of oxygen that provided the spray-type arc transfer. It became popular in the early 1960s. A recent adaptation is the use of pulsed current. The current is switched from a high to a low value at a rate of once or twice the grid frequency all the time. Soon after the enlightenment of CO_2 welding, a variation utilizing a special electrode wire was developed. This process was published in 1954 but was licensed in 1957, when the National Cylinder Gas Company reintroduced it to the welding sector. An inside-outside electrode was presented which did not need external gas shielding in 1959. For noncritical work, the absence of shielding gas gave the process popularity. In 1958, the electroslag welding process was announced by the Soviets at the Brussels World Fair in Belgium. Based on work done in the United States, it had been used in the Soviet Union since 1951 [1].

There was considerable research in shielding the arc and weld area by externally applied gases. The atmosphere of oxygen and nitrogen in contact with the molten weld metal created brittle and seldom acceptable welds. With utilizing gas shielding

techniques many types of research are done. Alexander and Langmuir did work in containers using hydrogen as a welding atmosphere. They utilized two electrodes beginning with carbon electrodes but later converting to tungsten electrodes. The hydrogen was converted to atomic hydrogen in the arc. It was then blown out of the arc forming a profoundly hot flame of thermonuclear hydrogen during to the atomic form and liberating heat. This arc effected half again as much heat as an oxyacetylene fire. This enhanced the atomic hydrogen welding method. Atomic hydrogen never became famous but was used around 1930 and 1940 for special employment of welding and later for welding of tool steels. Later there were similar projects done using atmospheres of argon and helium. These researches lead to arc welding utilizing gas supplied around the arc was a forerunner of the GTAW process. It is also figured out that welding with a concentric nozzle and with the electrode being fed as a wire through the nozzle. Thus, the forerunner of the GMAW process is invented. These processes were developed much later [2].

Design and implementation of a high-frequency 200 A, 5.6 kW welding machine and its controller is designed and analyzed [3].

Mathematical model of welding is studied, and different control algorithms are derived for GMAW process [4].

When designing welding machine from scratch, it is important to select the topology correctly before anything else. Several topologies have been studied and concluded that two switch forward converter and full bridge converter are the best suitable for welding applications [5].

It is mentioned that microcontroller-based arc length control design is more stable comparing to the automatic voltage controllers over wide range [6].

An extended dynamic matrix controller is designed on GMAW process, which is considered non-linear input and output system [7].

Sliding mode control is used in phase shifted full bridge Zero Voltage Switching (ZVS) arc welding inverter power source in order to overcome the contradictory

between the small voltage overshoot and the fast response time of the traditional proportional integral control [8].

Improvements for an arc welding power supply based on resonant inverter are investigated [9]. Also, in the rectifier stage, effects of using Metal Oxide Semiconductor Field Effect Transistor (MOSFET) instead of diodes are studied. An algorithm is proposed for starting and extinguishing the arc has been developed.

Digital controller is done with using Field Programmable Gate Array (FPGA) for arc welding machine [10].

1.1.1 Objectives of the Thesis

The main goals of this thesis are defined in following.

- Introduce the welding machines.
- Use a new algorithm based on digital control of peak current method by using dsPIC33EP128GS806.
- Evaluate the performance of the proposed algorithm of control method with designed welding machine.
- Test the efficiency of the welding machine.

1.1.2 Contribution of Thesis

In literature there are many studies that designs welding machines. However, these studies are mainly based analog controllers that have many components and fixed design because it depends on the passive parts. But in this study, the focus is designing digital controller with 16-bit Micro Controller Unit (MCU). In this thesis, a new algorithm that classifies peak current control using microcontroller is proposed. The proposed algorithm used analog circuits inside the MCU with firmware design. Design approaches of proposed algorithm are represented in detail. Confirmation of new method is investigated with current and voltage waveform.

1.1.3 Outline of the Thesis

The content of the dissertation is arranged as follows:

In Chapter 1, after a comprehensive literature review on welding history and welding properties is presented. The objectives and contributions of the dissertation are expressed.

In Chapter 2, definitions, causes and solutions of designing welding machine are given. Challenges and solutions are given.

In Chapter 3, design of the power stage, controller stage and driver stage are given in details.

In Chapter 4, simulation results and controller design are given. Flowchart of the controller is explained.

In Chapter 5, experimental results are given.

In Chapter 6, conclusions of this dissertation are given.

CHAPTER 2

DEFINITIONS

2.1 Voltage Source Inverters

Modern power electronics converters usually consist of 2 stages. In the first stage, the aim is to produce DC voltage from Alternating Current (AC) utility grid. This stage is called rectifier and generally uses 4 diodes connected bridge form (therefore called full bridge rectifier) and capacitors to rectify the AC utility grid to some certain value. The second stage involves semiconductors to achieve DC/AC transformation. Since the input to the whole system is voltage source based, it is called Voltage Source Inverter (VSI). The basic look of typical VSI is shown below in Figure 2.1. In Figure 2.1, rectifier block represents 6 diodes connected in bridge form which is called three phase full bridge rectifier. After that, a capacitor block connected in order to produce constant DC link voltage for the inverter. The capacitance of the capacitor determines DC link ripple which should be as small as possible in VSI. In the inverter block, the switches should be realized as Single Pole Double Throw (SPDT) since the switches must be capable of conducting both positive and negative current but need block only positive voltage.

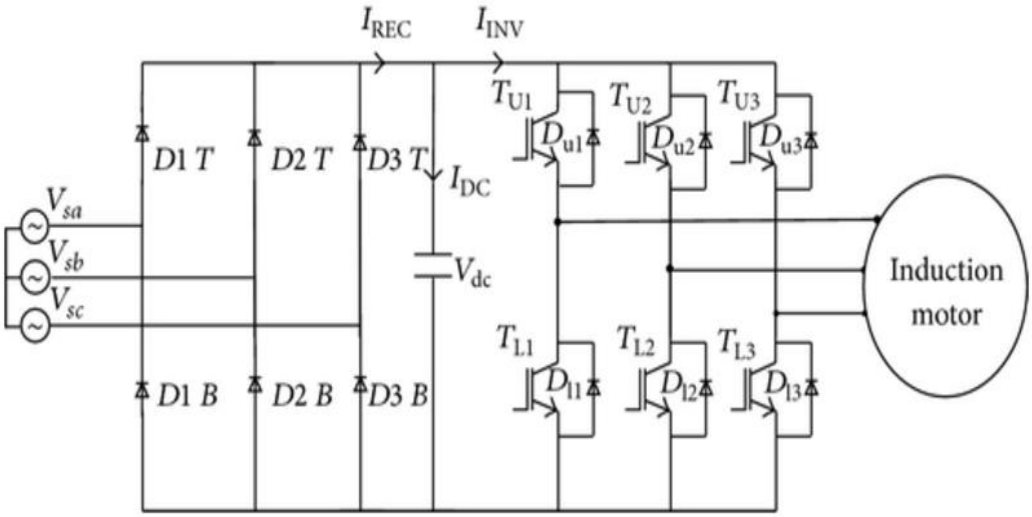

Figure 2.1 Voltage source inverter

VSI contains three two quadrant SPDT switches, one per phase [11]. In Figure 2.2, most widely applied VSI topology three-phase two level VSI is shown.

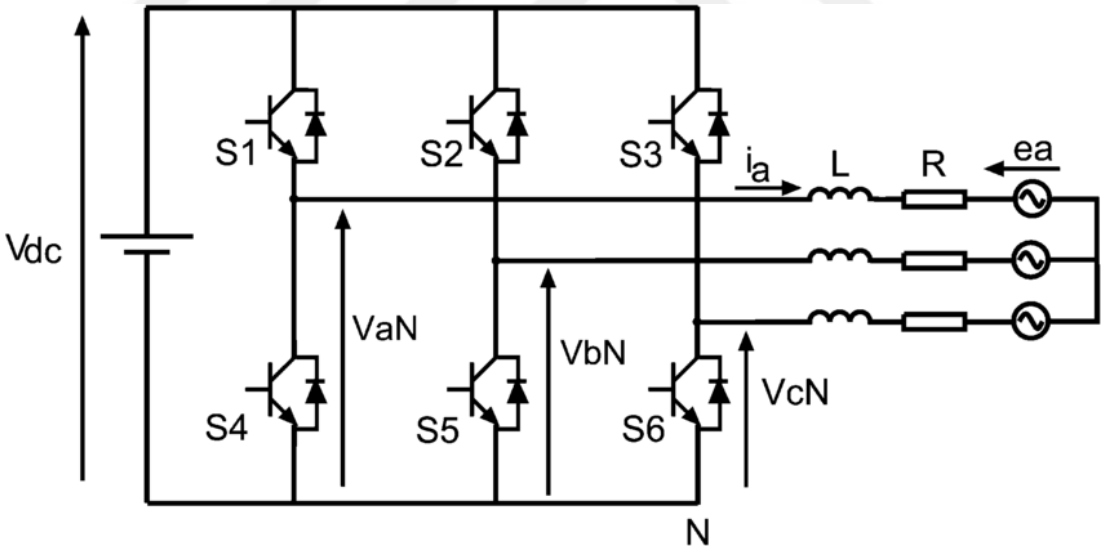

Figure 2.2 Three phase two level voltage source inverter

In VSI, 6 independent controller current bi-directional semiconductors with their body diodes are used to convert the DC Link voltage to variable voltage fixed frequency for different purposes. A phase can see +VDC, -VDC or zero for different switching states. Two level VSI is most common in many applications such as inverters for welding machines.

2.1.1 Half Bridge Inverter

Half bridge inverters are consisting of 2 switches located on the same branch. The switches are pulsed separately and symmetrically, and the output voltage is related to the middle point of the DC link and the middle point of the switches. Figure 2.3 shows the general look of the half bridge converter. In this circuit, the half of the DC link voltage is seen by secondary and to achieve higher powers, the secondary side turns count should be greater than the primary side turns count. Hence both switch currents and losses are greatly increased. Therefore, this circuit is not very effective in high power applications.

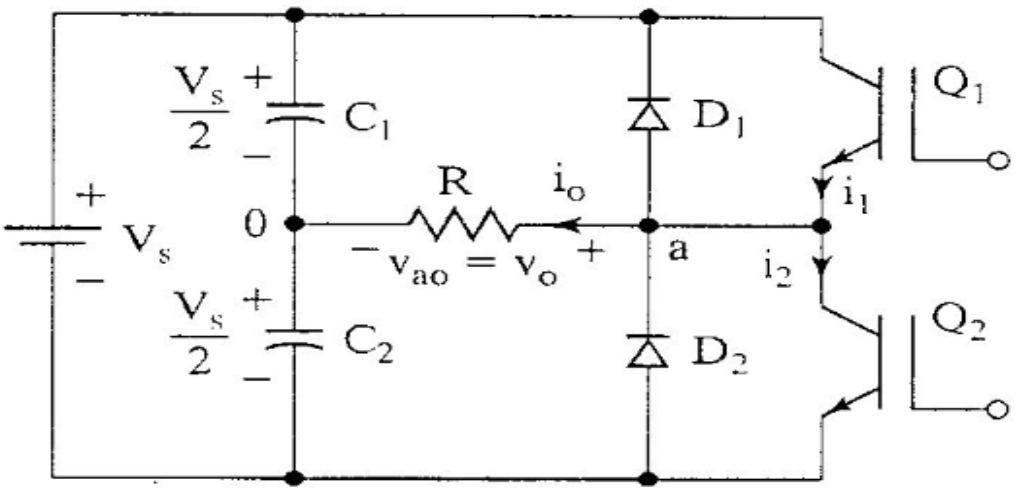

Figure 2.3 Half bridge inverter

2.1.2 Single Phase Full Bridge Inverter

The full bridge transformer isolated buck converter is shown in Figure 2.4. A version containing a center tapped secondary winding is shown. This circuit is commonly used in converters producing low output voltages. The two halves of the center tapped secondary winding may be viewed as separate windings, and hence we can treat this circuit element as a three winding transformer having turns ratio 1:n:n [11].

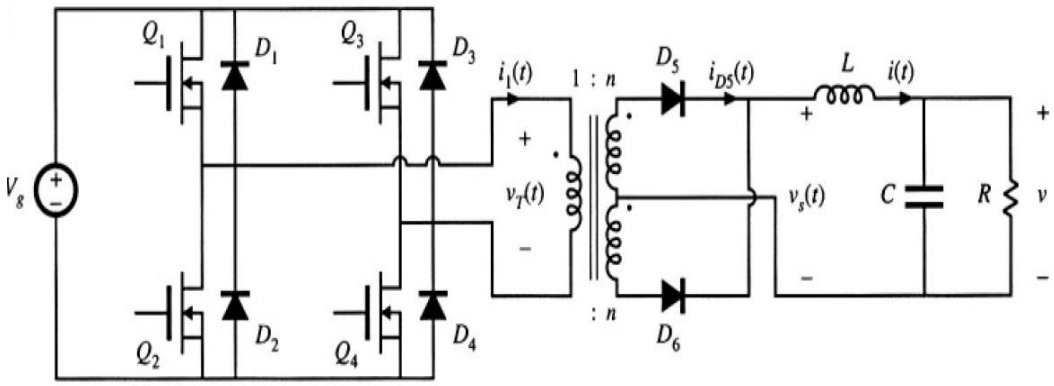

Figure 2.4 Full bridge inverter

During the first subinterval $0 < t < DTs$ transistors Q_1 and Q_4 and conduct, and the transformer primary voltage is equal to the input voltage.

$$VT = Vg \quad \textbf{(2.1)}$$

This positive voltage causes the magnetizing current IM(t) to increase with a slope proportional to Vg inversely to magnetizing inductance.

$$IM(t) = \frac{Vg}{Lm} \quad \textbf{(2.2)}$$

The voltage appearing across each half of the center tapped secondary winding is directly proportional to number of turns n, with the polarity mark at positive potential.

$$Vs(t) = nVg \quad \textbf{(2.3)}$$

D_5 is forward biased, and D_6 is reverse biased. The voltage Vs(t) is then equal to nVg and the output filter inductor current i(t) flows through diode D_5 [11]. Figure 2.5 shows the first interval results of the inverter.

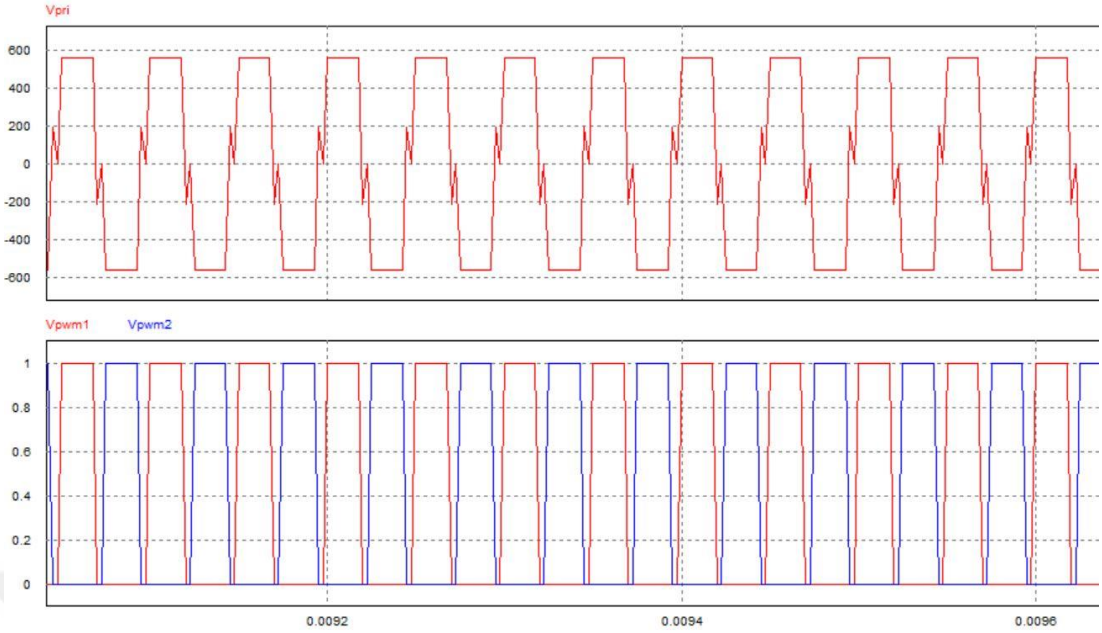

Figure 2.5 Switching sequence of full bridge inverter

Several transistor control schemes are possible for the second subinterval $DTs < t < Ts$. In the most common scheme, all four transistors are switched off, and hence the transformer voltage is VT=0. Alternatively, transistors Q_2 and Q_4 could conduct, or transistors Q_1 and Q_3 could conduct. In any event, diodes D_5 and D_6 are both forward biased during this subinterval; each diode conducts approximately one half of the output filter inductor current [11]. In Figure 2.6, secondary side of the inverter is given.

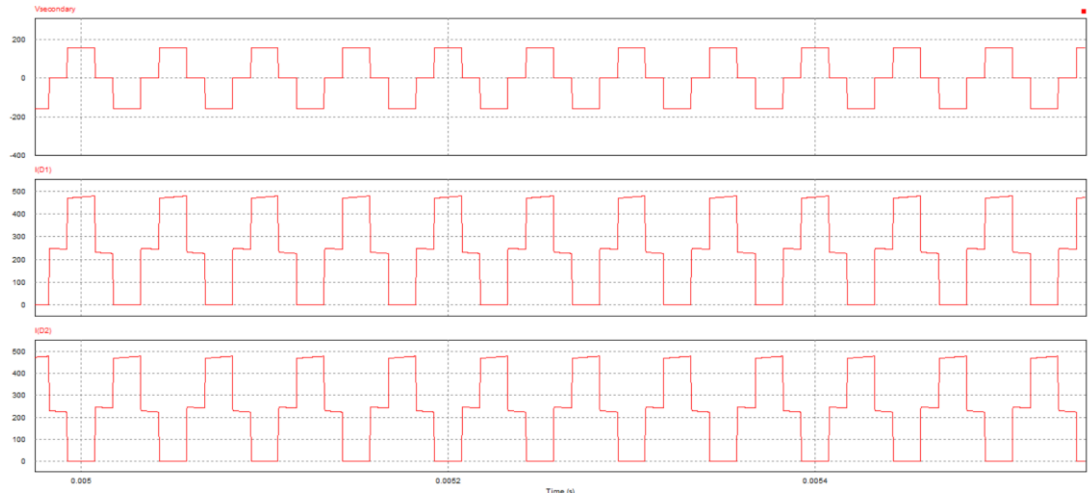

Figure 2.6 Secondary side of the inverter

The average value of the output voltage of this topology, can be found by because the average of the inductor voltage value is zero.

$$< VL > = 0 = D(nVg - Vo) - (1 - D)Vo \quad \textbf{(2.4)}$$

Solving the formula,

$$Vo = nDVg \quad \textbf{(2.5)}$$

can be found.

The inductor current is risen during the first interval of DTs.

$$\frac{diL}{dt} = \frac{\frac{Vg}{n} - Vo}{L} \quad \textbf{(2.6)}$$

using the formula,

$$\Delta I = dT \frac{\frac{Vg}{n} - Vo}{L} \quad \textbf{(2.7)}$$

is found. If output voltage is written according to input voltage, the current ripple is found as:

$$\Delta I = Vg \frac{d(1-D)}{fswLn} \quad \textbf{(2.8)}$$

is found. Using these formulas, the necessary inductance value is found as

$$L = \frac{Vo(1-D)}{fsw\Delta I} = \frac{Vg(1-D)}{fsw\Delta In} \quad \textbf{(2.9)}$$

is found. In this thesis, filter inductor value is chosen 30 micro Henry for the optimum welding model.

2.2 Control Methods

In this section, the most common control methods and their properties are given. Although Voltage Mode Control (VMC) is much common in digital controllers, in this thesis peak current control method is used for its fast response abilities.

2.2.1 Voltage Mode Control

VMC approach is used for first switching regulator designs for many years. The basic voltage mode configuration is shown in Figure 2.7.

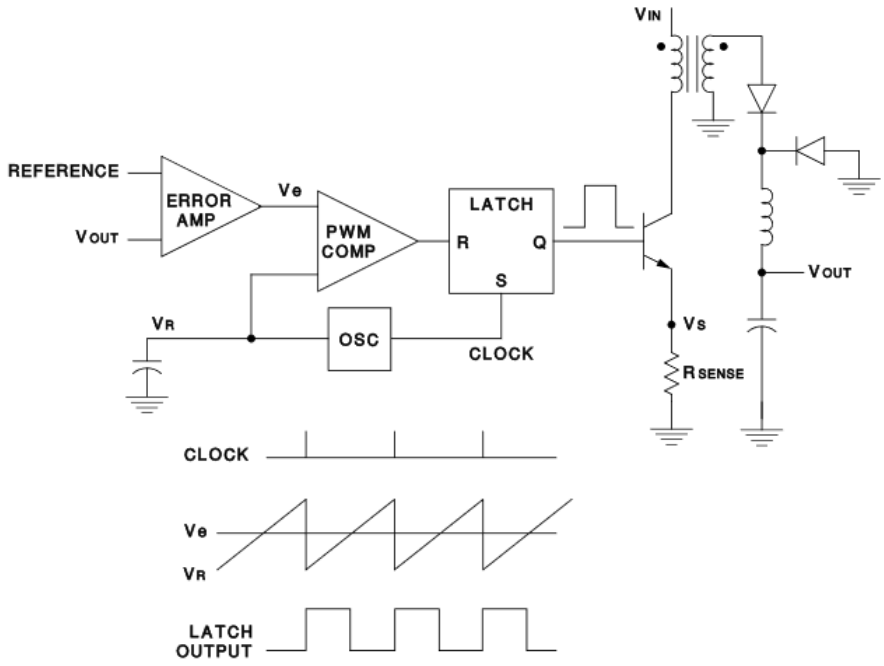

Figure 2.7 Voltage mode control [12]

The major characteristics of this design are that there is a single voltage feedback path, with Pulse Width Modulation (PWM) performed by comparing the voltage error signal with a constant ramp waveform. Although single feedback loop is easier to design and analyze and large amplitude ramp waveform provides good noise margin for stable modulation process, any change in line or load must first be sensed as an output and regulated by the feedback loop, which means slow response. Also output filter adds two poles to the control loop which means -40dB/decade after resonance frequency. Hence, compensation is further complicated by the fact that the loop gain varies with input voltage.

2.2.2 Current Mode Control

The disadvantages of VMC led people to find a new topology. In current mode control, the oscillator used as fixed frequency clock and ramp waveform is replaced with a

signal derived from output inductor current. Figure 2.8 shows the basic diagram of current mode control.

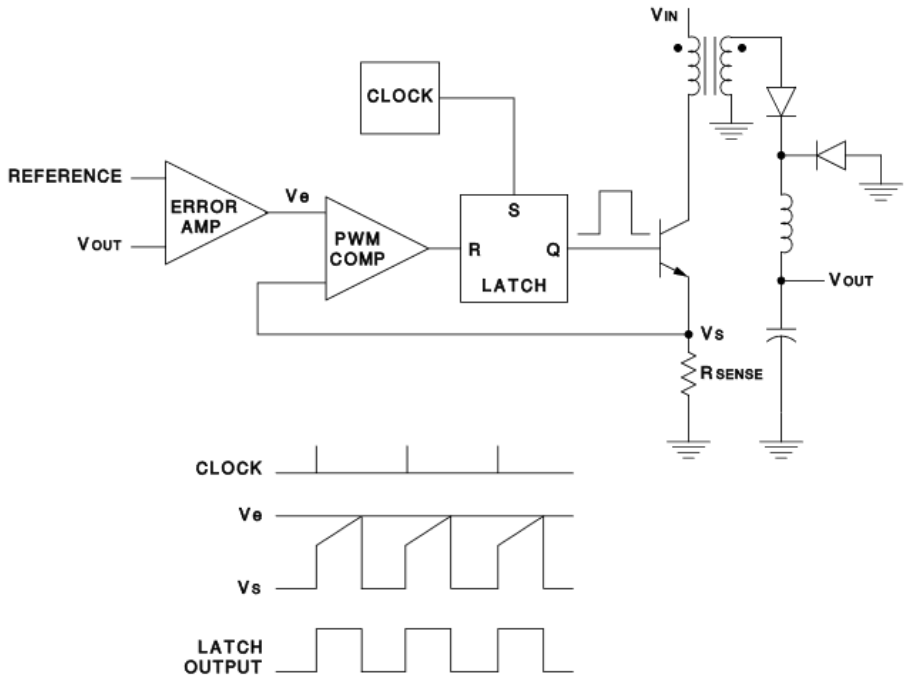

Figure 2.8 Current mode control [12]

Since the inductor current rises with a slope determined by input and output voltage, this waveform is expected to respond immediately to line voltage changes which eliminates both delayed response and gain variation with changes in input voltage. Also, the error amplifier is now used to command an output current rather than voltage, the filter offers only a single pole to the feedback loop. This allows the user both simpler compensation and high gain bandwidth over a comparable voltage mode circuit [13]. Although there are significant benefits, there are certain problems that must be solved in the design process. Since there are two feedback loops in play, the circuit is now much more complex comparing to VMC. Unless slope compensation is added, control loop becomes unstable at duty cycles over 50%. The more compensation added; the more stable circuit becomes but adding slope means that current control is now become VMC. In welding applications, the error amplifier is used to command an output current, and the filter offers only a single pole to the feedback loop. This allows the user both a simpler compensation and high gain

bandwidth over a comparable VMC converter [8]. Also, it is mentioned that microcontroller-based arc length control design is more stable compared to the automatic voltage controllers over a wide range [5]. In this thesis, peak current control application is realized for a welding machine using dsPIC33EP128GS806. The proposed method is shown in dsPIC33EP128GS806 has an inner fast comparator and set reset latch to truncate the PWM signals for current control mode. Also, programmable gain amplifiers are available to filter and buffer analog input signals. The error amplifier part is simply coding part that runs in an interrupt routine.

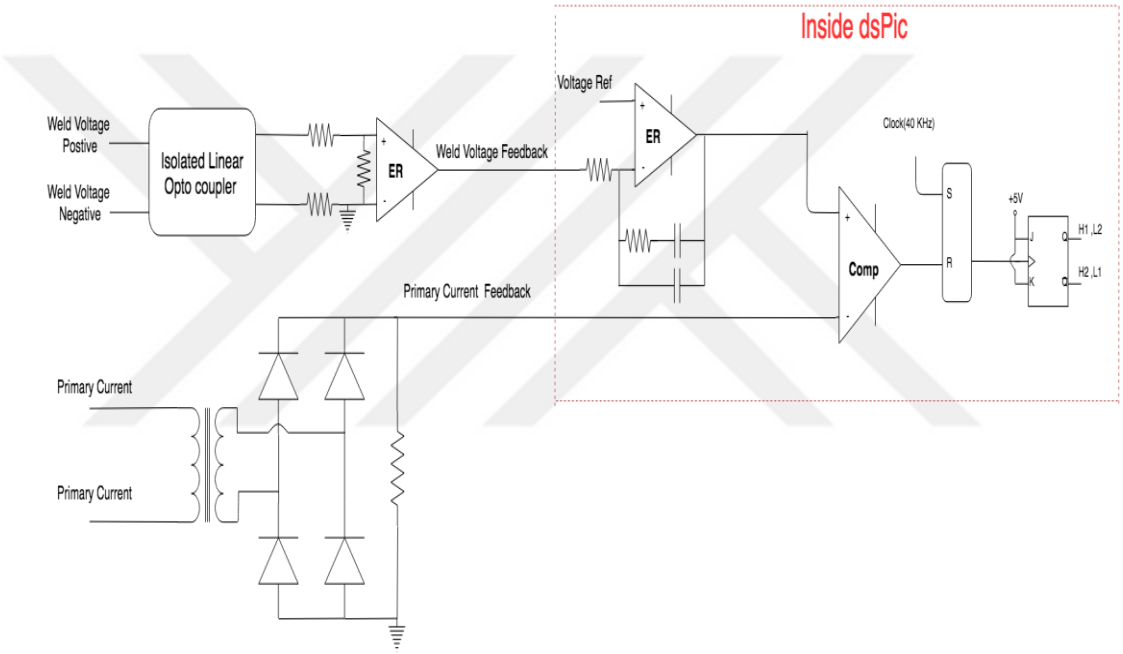

Figure 2.9 MCU realization of current mode control

2.3 PWM Methods

There are two main implementation techniques for carrier based PWM methods, which are scalar implementation and space vector implementation. In the scalar approach, for each inverter phase, a modulation wave is compared with a triangular carrier wave. The intersection points determine the switching instants for the associated inverter leg switches. In the space vector approach the time lengths of the inverter states are precalculated for each carrier cycle by employing space vector theory and the output voltage pulses are directly programmed. The mathematics involved in the scalar

method of modulation waves and the space vector calculations is related. With proper modulation waves, the scalar and space vector PWM pulse patterns can be made identical. Thus, both techniques may have equivalent performance results. But the scalar approach is simpler than the space vector approach from the implementation perspective as the involved computations are generally fewer and less complex [14]. One of the most used PWM approaches called as a per carrier cycle volt second balance principle is generally suitable for power electronics converters. This principle states that in one PWM period the reference value determines the average value of the output voltage [15]. Hence desirable output voltage can be obtained with a reference voltage for each PWM period. In the scalar approach, for each inverter phase, a modulation wave is compared with a triangular carrier wave shown in Figure 2.10.

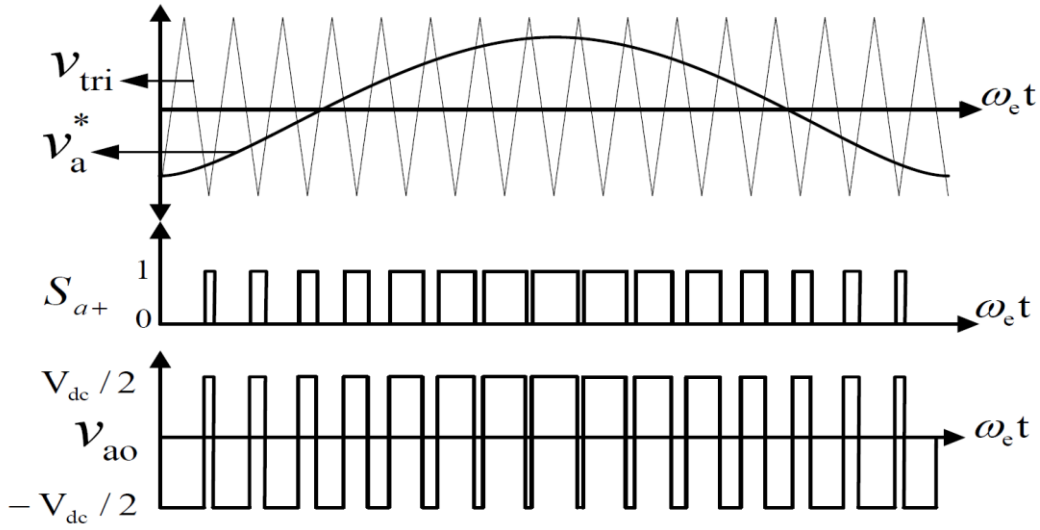

Figure 2.10 Scalar PWM generation

2.4 GMAW Modelling Methods

2.4.1 Load Behavior of the GMAW System

GMAW is one of the several types of welding methods. It is highly efficient and can be used to weld any material of any thickness in any position with the possibility of automatic implementation. GMAW nowadays is considered as the most popular type of arc welding process. Figure 2.11 shows a schematic that represents the GMAW process.

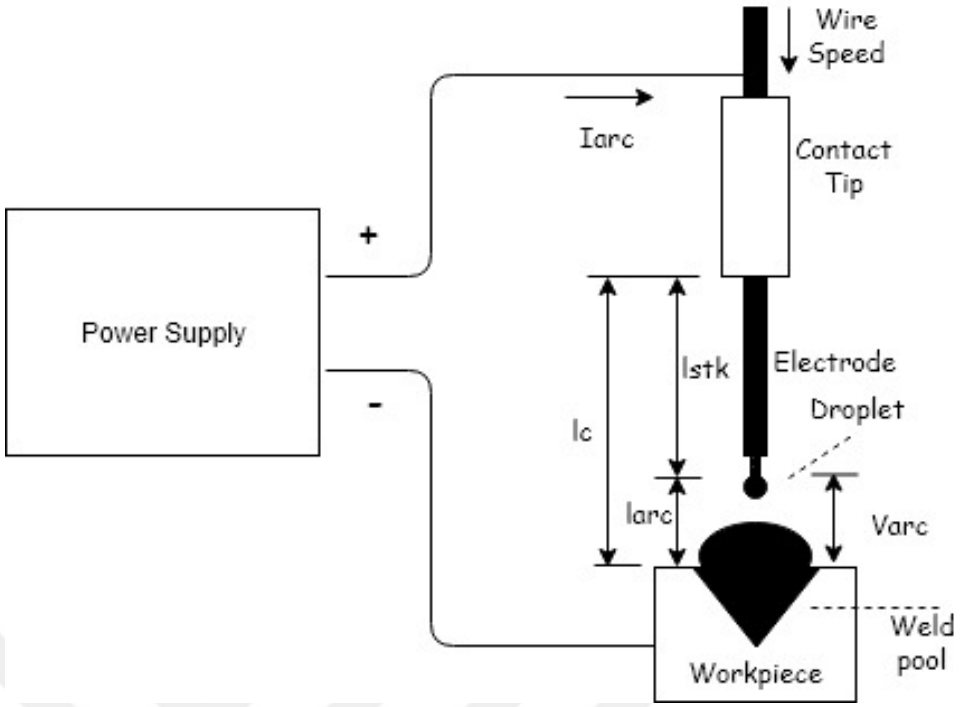

Figure 2.11 Electrical model of the GMAW process

2.4.2 Modelling of GMAW Process

The welding machine for GMAW includes a DC power source. The positive terminal of this power source is connected to the welding electrode while the negative one is connected to the workpiece or vice versa. At the steady state when the arc current (I_{arc}) and the arc length (l_{arc}) are positive, the arc voltage (V_{arc}) and arc power functions that generate arc energy for making a metal joint can be expressed as follows:

$$V_{arc} = V_{ccz} + E_{arc}\, l_{arc} + R_{arc}\, I_{arc} = V_{ccz} + E_{arc}\,(l_c - l_{stk}) + R_{arc}\, I_{arc} \quad \textbf{(2.10)}$$

The power arc (P_{arc}) can be expressed as:

$$P_{arc} = V_{ccz} I_{arc} + (E_{arc} l_{arc} + R_{arc} I_{arc}) I_{arc} = V_{ccz} I_{arc} + (E_{arc}(l_c - l_{stk}) + R_{arc} I_{arc}) I_{arc} \quad \textbf{(2.11)}$$

Four distinct parameters of arc are the constant charge zone voltage (V_{ccz}), which is the sum of cathode and anode drop voltages that are considered constant, the arc length (l_{arc}), the arc length factor (E_{arc}) and the arc resistance (R_{arc}). The distance between contact tube and base plate (l_c) is the sum of the arc length (l_{arc}) and stick out distance (l_{stk}) as shown in Figure 2.11. To define the load voltage and power of the system the electrode resistivity (ρ) needs to be known. The load voltage (V_{load}) and load power (P_{load}) are given as follows:

$$V_{load} = V_{ccz} + E_{arc} l_{arc} + (R_{arc} + \rho l_{stk}) I_{arc} \quad \textbf{(2.12)}$$

$$P_{load} = \rho l_{stk} I_{arc} + P_{arc} \quad \textbf{(2.13)}$$

To define the wire feed speed (v_{fsd}) the electrode melting rate (v_{mr}) and the rate of change of electrode stick out should be known.

$$v_{fsd} = \frac{dl_{stk}}{dt} + v_{mr} \quad \textbf{(2.14)}$$

$$v_{mr} = \frac{4000}{q_m \pi \rho_d d^2} p\, l_{stk} I_{arc} + \eta_{arc} V_{ccz} I_{arc} \quad \textbf{(2.15)}$$

where q_m is quantity of heat needed to melt one gram of weld metal, ρ_d is the density, d is the dimeter of the wire and η_{arc} is the heat transfer efficiency of the process. Equation 2.15 can be rewritten as:

$$v_{mr} = k_1 I_{arc} + k_2 l_{stk} I_{arc}^2 \quad \textbf{(2.16)}$$

where k_1 and k_2 are constant for a specific operating set up.

2.4.3 Conventional Load Regulation

In general, there are two types of regulations in welding machines which are Constant Voltage (CV) and Constant Current (CC) regulations. In CC mode machines, the

current effects the melt off rate or consumption rate of the electrode, whether it is a stick electrode or wire electrode. The higher the current level, the faster the electrode melts or the higher the melt off rate. In CV mode machines, the voltage controls the length of the welding arc, and the resulting width and volume of the arc cone. As the voltage increases, the arc length gets longer. Table 2.1 summarizes the recommended output types by welding process.

Table 2.1 Recommended power source output type by arc welding process

Welding Process	**Shielded Metal Arc Welding (SMAW)**	**GTAW**	**GMAW**	**Flux Cored Arc Welding (FCAW)**
Output Type	CC	CC	CV	CC

Figure 2.12 shows the typical output curve of a CV GMAW, which is chosen in this thesis. The voltage stays in a narrow band although the current may vary in a large range.

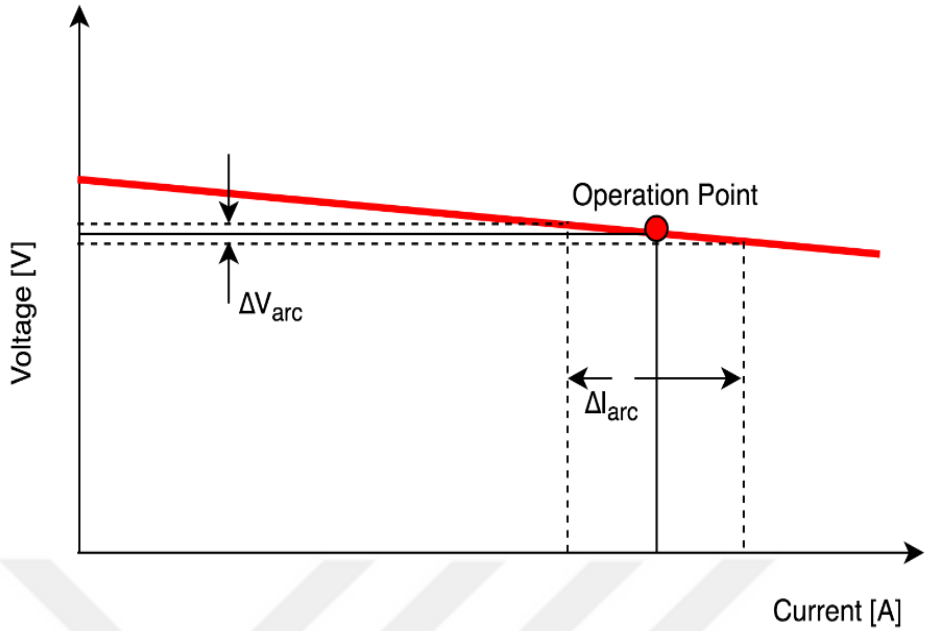

Figure 2.12 GMAW constant voltage curve [16]

In table 2.2, parameters and some variables of GMAW process model used in the simulations for carbon steel is given.

Table 2.2 GMAW parameters

Parameters	Symbol	Value (Unit)
Arc resistance	R_{arc}	0.0237 (Ω)
Arc length factor	E_{arc}	400 (vm^{-1})
Constant charge zone	V_{ccz}	12 (V)
Resistive of the electrode	ρ	0.43 (Ω m^{-1})
Density of the liquid electrode material (steel)	ρ_d	7850 (kg m^{-3})
Electrode dimeter	D	0.01 (m^2)
Melting rate constant 1	k_1	3.3 x 10^{-10} (m^3 S^{-1} A^{-1})
Melting rate constant 2	k_2	0.78 x 10^{-10}(m^3 S^{-1} A^{-1})
Quantity of heat needed to melt one gram of weld metal (Steel)	Q_m	≈1430 (J/gm)
Open circuit voltage	V_{oc}	22 (V)
Constant distance between contact tube and base plate	l_c	23.5 (mm)
Stick out distance	l_{stk}	11 (mm)
Arc length distance	l_{arc}	12.5 (mm)

CHAPTER 3

DESIGN STAGE

In the design stage, it is considered that the output current will be 350A continuously. It is named 400A welding machine because it can give 400A 60% of the working time, which means, after 6 minutes of working in 400A region, the welding machine should stop for at least 4 minutes. Also, when working in 350A continuously, the welding machine should not exceed 2 ℃ per hour, meaning, when the machine works, if IGBT temperatures are 70 ℃, after 1 hour of active working the IGBT temperatures should not exceed 72 ℃. This is called the regime of the welding machine and it can be considered the most important criteria of all. In Figure 3.1, system block diagram of the welding machine is given.

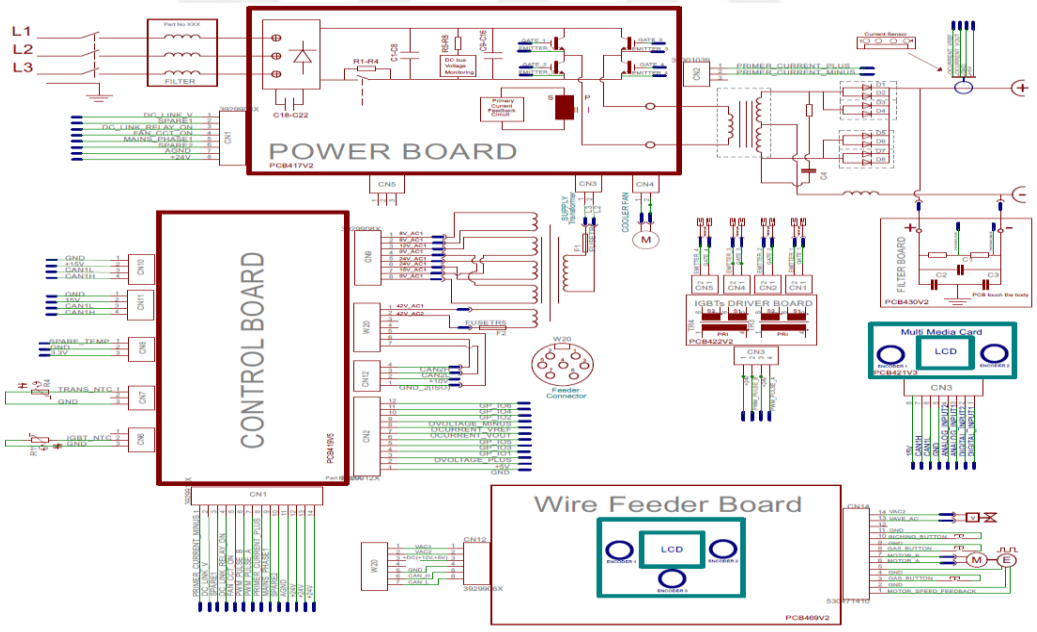

Figure 3.1 System block diagram

3.1 Power Stage Design

The power stage consists of a harmonic filter, an AC/DC rectifier, full bridge DC/DC inverter circuit, a high frequency transformer, a rectifier and a filter circuit respectively. The structure of the system is shown in Figure 3.2.

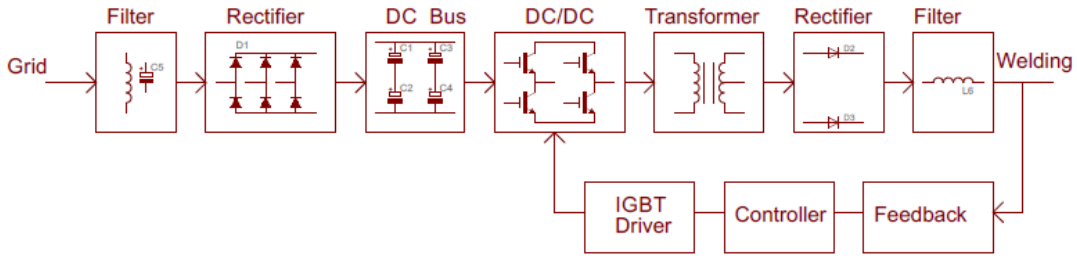

Figure 3.2 Welding machine power stage design

3.1.1 Transformer Design

In the transformer design state, the first thing to do is to specify the specifications of the transformer. The relation between primary and secondary voltages are also given as:

$$n = \frac{Vs}{Vp} \tag{3.1}$$

When designing the transformer, the power of the system, input and output voltages and input and output currents are used. The output of the welding voltage is decided according to EN60974-1 standard which also decided the output power of the welding machine. Below the EN60974-1 standard of the current and voltage relation is given.

$$Vwelding = Iwelding \times 0.04 + 20 \tag{3.2}$$

When the welding current is 400A, the expected welding voltage is found as;

$$Vwelding = 400 \times 0.04 + 20 = 36\,V \tag{3.3}$$

But there is a voltage drop at the secondary rectifier (1.4V), filter inductor (3V) and conduction losses (2.6V) so the expected output power will be;

$$Pwelding = 43 \times 400 = 17200\,W \tag{3.4}$$

The EN60974-1 standard defines the minimum voltage of the level of the welding, but the machine is intended to be used with all kind of electrodes. Also configuring arc length is one of the aims and therefore when welding short circuit conditions occur it is expected to give at least 80V in short period of time. So, when the ratio of the

transformer is calculated, this ratio will be used. The data for the transformer is given as given in Table 3.1.

Table 3.1 Welding machine system parameters

Circuit Topology	Full bridge
Secondary rectifier circuit type	Center tapped
Input DC voltage (V)	Min: 486 V, Max: 600 V
Output voltage	10-36 V
Output current	400 A
Switching frequency	20 kHz
Output current ripple	18 A
Turn ratio Np/Ns	14/2/2
Switching period (us)	50 us
No load secondary voltage (V)	80 V
Output filter inductance (uH)	30 uH
Rated output power	17.2 kW
Primary current (A)	60 A
Max. primary over current (A)	70 A
Primer inductance(mH)	1.8 mH
Operating temperature (℃)	100 ℃
Cooling system	Air cooling

After the specifications, a transformer is designed. Figure 3.3 shows the designed transformer.

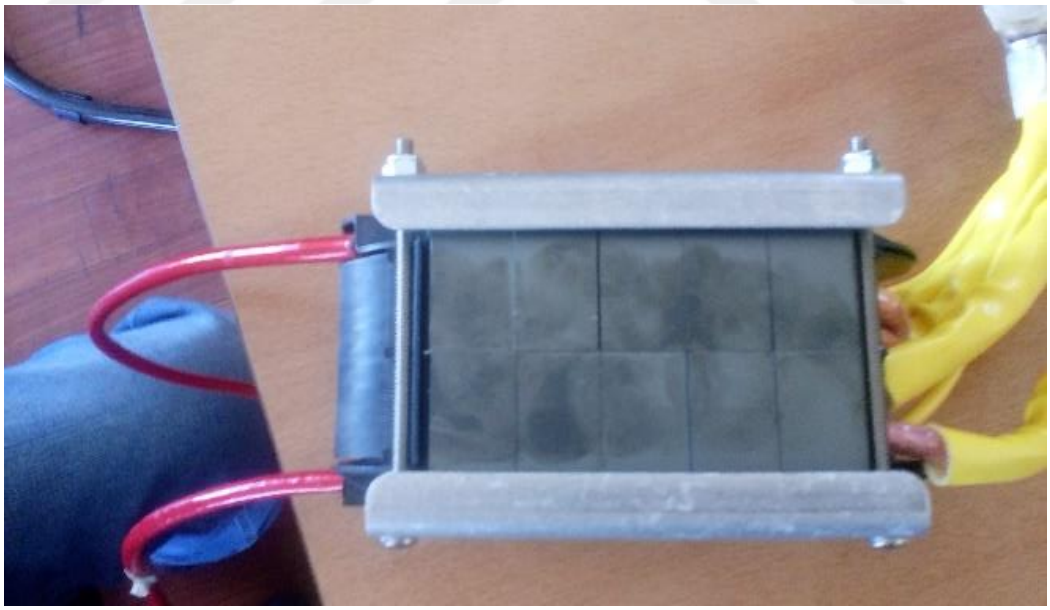

Figure 3.3 Picture of the designed transformer

3.1.2 Filter Inductor Design

The output filter inductor is designed using the formula:

$$L = \frac{\frac{Vg}{n} - Vo}{dI} \, dT \tag{3.5}$$

$$L = \frac{\frac{560}{7} - 43}{61} \times 50 \times 10^{-6} \, uH \tag{3.6}$$

$$L = 30 \, uH \tag{3.7}$$

Although maximum secondary current is 400A, filter inductor is designed for 500A just to be on the safe side. Filter inductor design is done by Nuriş AŞ. Figure 3.3 shows the designed filter inductor.

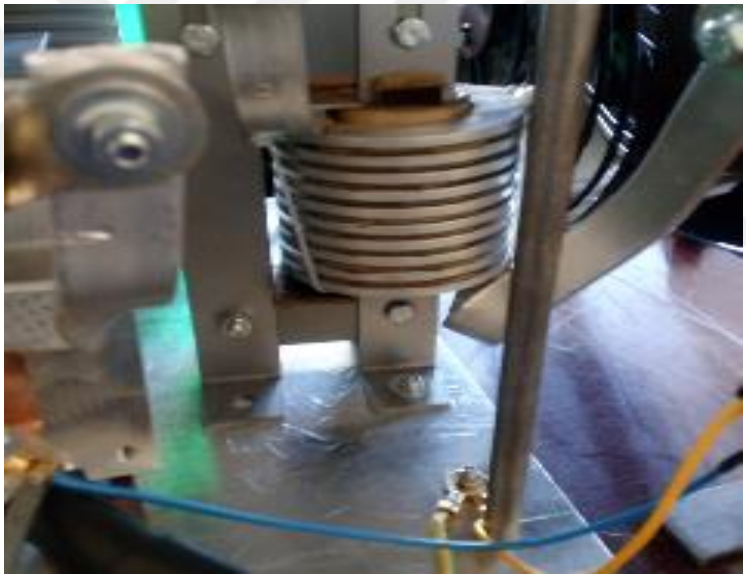

Figure 3.4 Picture of the designed filter inductor

3.1.3 IGBT Selection

IGBT switches will be seeing full DC link when they are off. This is expected to be around 560V. Since transformer turns ratio is 7:1, 80V will be reflected secondary side, which means total of 640V per IGBT. When designing power stage at least 40% margin is added to be on the safe side. In this study, 2 model of Semikron

SKM200GB125D is selected. Every model has 2 ultrafast IGBT with maximum continuous voltage 1200V and current 200A and a voltage drop of 1.5V.

3.1.4 Diode Selection

In primary side, for rectifier diodes the same method as IGBT's is used. Since the maximum reverse voltage on diodes are DC Link plus reflected secondary side, SEMIKRON SKD82/16 selected for three phase rectification circuit. It has voltage drop of 1.5V. On the secondary side the current ratings became crucial and Turbo 4 ultrafast high voltage rectifier STTH200W06TV1 from ST Microelectronics is selected for secondary rectification circuit. Each one has two diodes paralleled and they have voltage drop of 1V.

3.1.5 Loss Calculations

When designing power stage, one of the most important things is to calculate losses and overall efficiency since no considerable losses are expected controller or driver circuit.

3.1.5.1 Semiconductor Loss Calculations

All semiconductor losses can be calculated with the technical information given manufacturers' datasheets.

IGBT switching losses can be calculated as:

$$Ploss, igbt, sw = \frac{fswVswIsw(trise+tfall)}{2} \quad \textbf{(3.8)}$$

where,

$$trise = 36\ ns \quad \textbf{(3.9)}$$

$$tfall = 25\ ns \quad \textbf{(3.10)}$$

$$Vdrop = 1.5\ V \quad \textbf{(3.11)}$$

$$trr = 55\ ns \quad \textbf{(3.12)}$$

Hence,

$$Ploss, igbt, sw = 0.5 \times 20 \times 10\text{^}3 \times 560 \times 57 \times (36 \times 10\text{^}(-9) + 25 \times 10\text{^}(-9) \quad \textbf{(3.13)}$$

$$Ploss, igbt, sw = 19.47\ W \quad \textbf{(3.14)}$$

Since there are 4 IGBT in full bridge topology the total switching loss would be around 78 W.

IGBT conduction losses can be calculated as:

$$Ploss, igbt, cond = DVdropIsw \quad \textbf{(3.15)}$$

$$Ploss, igbt, cond = 0.425 \times 1.5 \times 57 \quad \textbf{(3.17)}$$

$$Ploss, igbt, cond = 36.3375\ W \quad \textbf{(3.18)}$$

Since there are 4 IGBT in full bridge topology the total conduction loss would be around 145 W.

The total losses from IGBT's are calculated as 223 W.

Primary diodes are rectifying at 50 hertz, therefore reverse recovery losses can be ignored. Total loss from primary diodes can be calculated as:

$$Ploss, primary\ diode, cond = Vdrop\ Iprimary \quad (\textbf{3.19})$$

$$Ploss, primary\ diode, cond = 1.5 \times 57 \quad (\textbf{3.20})$$

$$Ploss, primary\ diode, cond = 85.5\ W \quad (\textbf{3.21})$$

Secondary side rectification diodes are rectifying at high frequency so reverse recovery losses are present.

$$Ploss, secondary\ diode, sw = \frac{fIsecondarytrrfswVrrm}{2} \quad \textbf{(3.22)}$$

$$Ploss, secondary\ diode, sw = 0.5 \times 400 \times 55 \times 10^{-9} \times 20 \times 10\text{^}3 \times 80 \quad \textbf{(3.23)}$$

$$Ploss, secondary\ diode, sw = 16.42\ W \quad \textbf{(3.24)}$$

$$Ploss, secondary\ diode, cond = DVdropIsec \quad \textbf{(3.25)}$$

$$Ploss, secondary\ diode, cond = 0.4251 \times 400 \quad \textbf{(3.26)}$$

$$Ploss, secondary\ diode, cond = 170\ W \quad \textbf{(3.27)}$$

Since there 2 diodes in each branch, the total diode loss would be equal to 372.84 W.

3.1.5.2 Transformer Loss Calculations

For primary side, 0.2 mm diameter 240 litz wires are paralled. For the secondary side, 0.1mm diameter 1500 litz wires are paralleled.

$$Bmax = 300\ mT \quad \textbf{(3.29)}$$

$$AL = 5070\ nH \quad \textbf{(3.30)}$$

$$Np = 14\ Turns \quad \textbf{(3.31)}$$

$$Ns = 2\ Turns \quad \textbf{(3.32)}$$

In Figure 3.5, power stage of the selected components is shown.

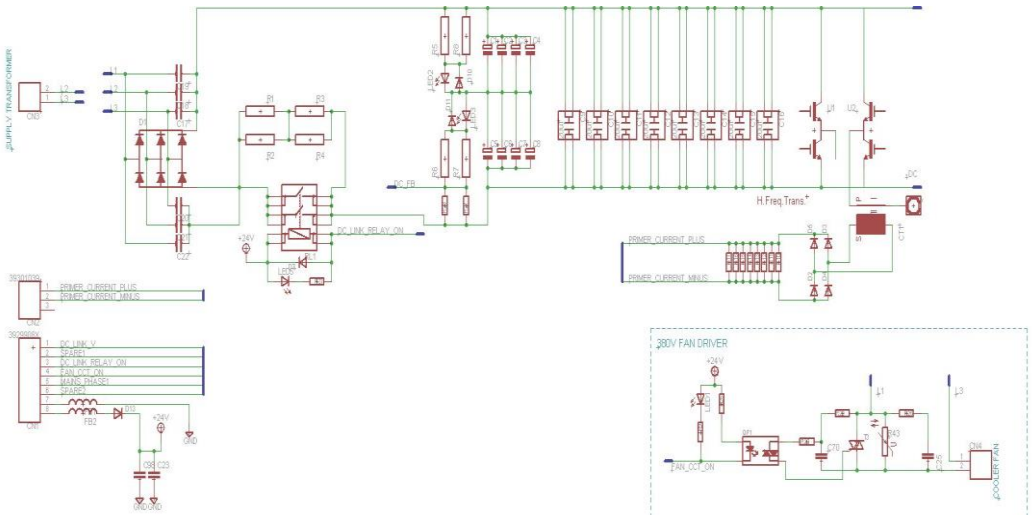

Figure 3.5 Power stage schematics

3.2 Input Filter Stage Design

Input filter is designed to reduce harmonics of the inverter to the grid. A pi filter is used with 0.68uF-310VAC capacitors as X-capacitors and 0.015uF-310VAC capacitors as Y-capacitors. Figure 3.6 shows the input filter schematics.

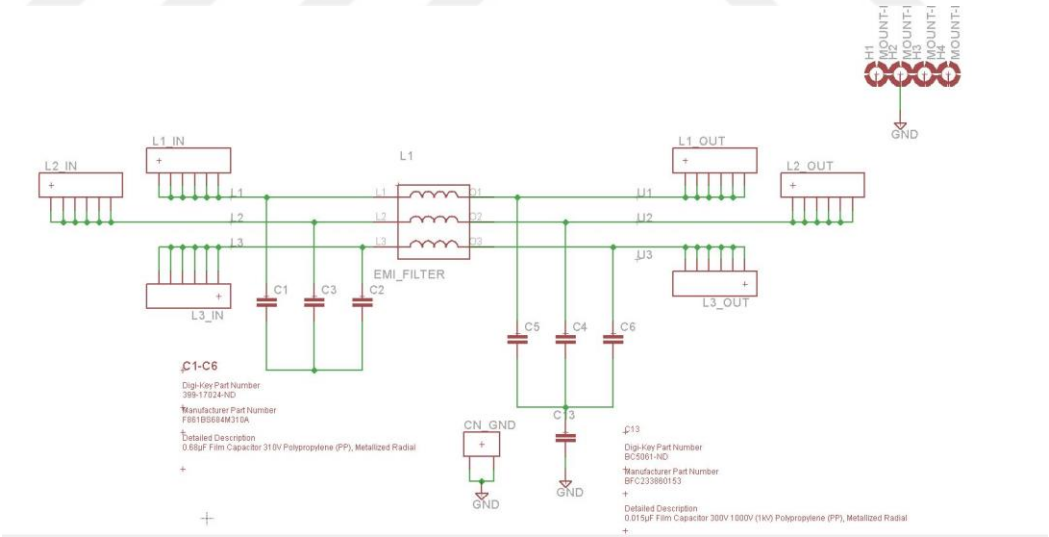

Figure 3.6 Input filter design

3.3 Controller Stage Design

In controller stage design, 16-bit MCU is used. It has interconnected switched mode power supply peripherals and supports cycle by cycle current mode control. It has four

PWM generators (two outputs per generator) and 12-bit dedicated Analog to Digital Converter (ADC) core up to 3.25 mega sample per seconds conversion rate per channel at 12-bit resolution. When designing controller stage, both primary and secondary current feedbacks are considered. Also output voltage is taken in order to execute some specific welding properties. Figure 3.7 shows the dsPIC33EP128GS806 pin connections. As seen in the schematics, voltage, current and temperature signals are taken as analog input. 1 isolated, 1 non-isolated Controller Area Network (CAN) driver module is used to communicate with different machines at the same time. Also, digital outputs such as relay control, fan control, reset latch are configured in the schematics. Both internal and external watchdog is used to maximize security when the control loop takes over.

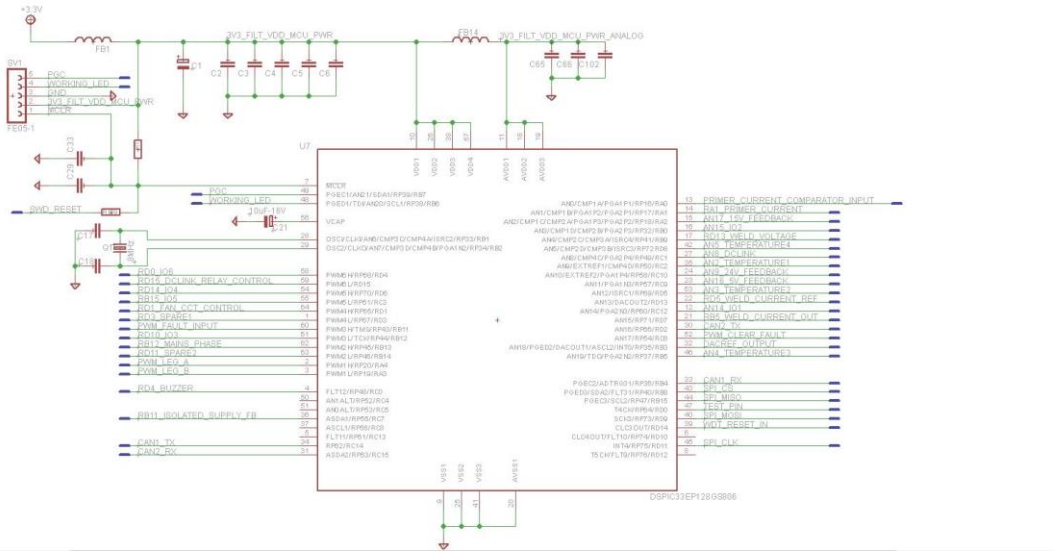

Figure 3.7 MCU connection

In Figure 3.8, CAN driver connection schematics is shown. Both isolated (ISO1050-DUB) and non-isolated (SN65HVD233D) CAN drivers are realized as in the Figure 3.8. A ferrite is used to minimize disturbances from other modules. The connectors are also shown in Figure 3.8.

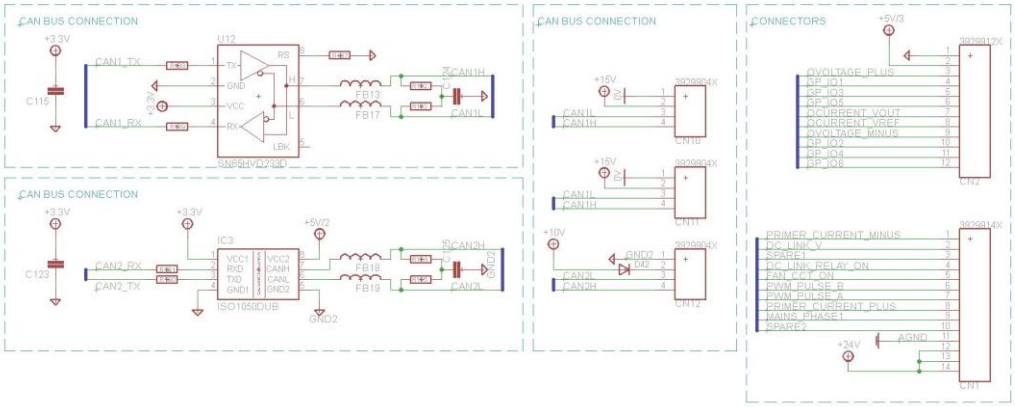

Figure 3.8 CAN driver connection

As the control module power source, a grid frequency transformer is used. The transformer has isolated channels to supply different modules. In Figure 3.9, the power supply and buck converter circuits of the control part is shown.

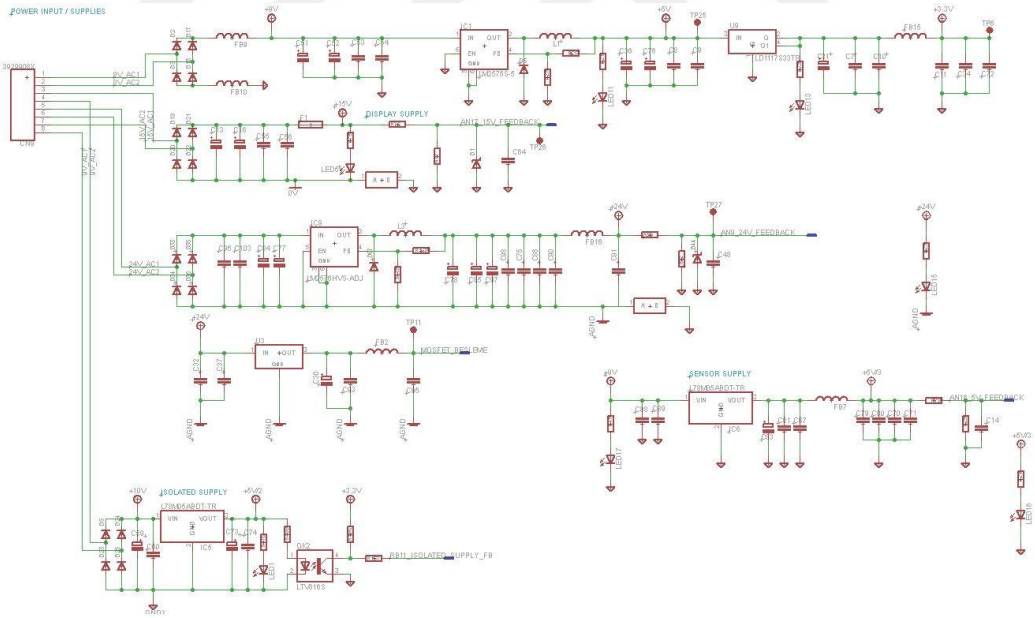

Figure 3.9 Controller power supply schematics

As seen in the schematic, 3 different grounds are present. Gate driver circuit and control circuit has a star connection at the controller side. In the system, at the output there is a current sensor which measures to secondary current. The current sensor also

has a reference voltage. The reference voltage is also read to calculate the secondary current precisely. In Figure 3.10, output current filter schematics is shown.

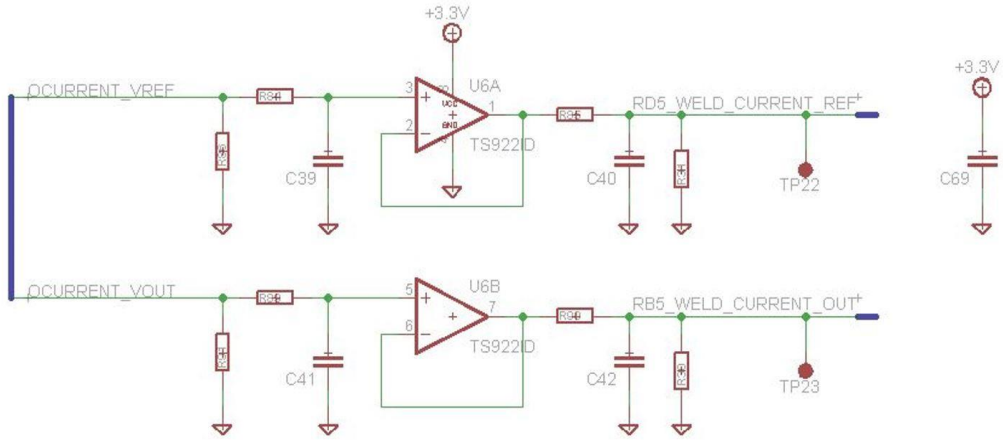

Figure 3.10 Secondary current feedbacks

In this study, DC link is measured for fault conditions and to open the DC link relay. Also, primary current is measured with current transformer ratio of 100:1. A buzzer is used to alarm in fault conditions. Temperatures of Insulated Gate Bipolar Transistor (IGBT), transformer, output diodes and output filter inductor is measured using 10K Negative Temperature Coefficient (NTC). An interpolation algorithm is used to calculate temperatures fast speeds. Also an isolated phase failure circuit is used. Figure 3.11 shows the schematics of the feedbacks.

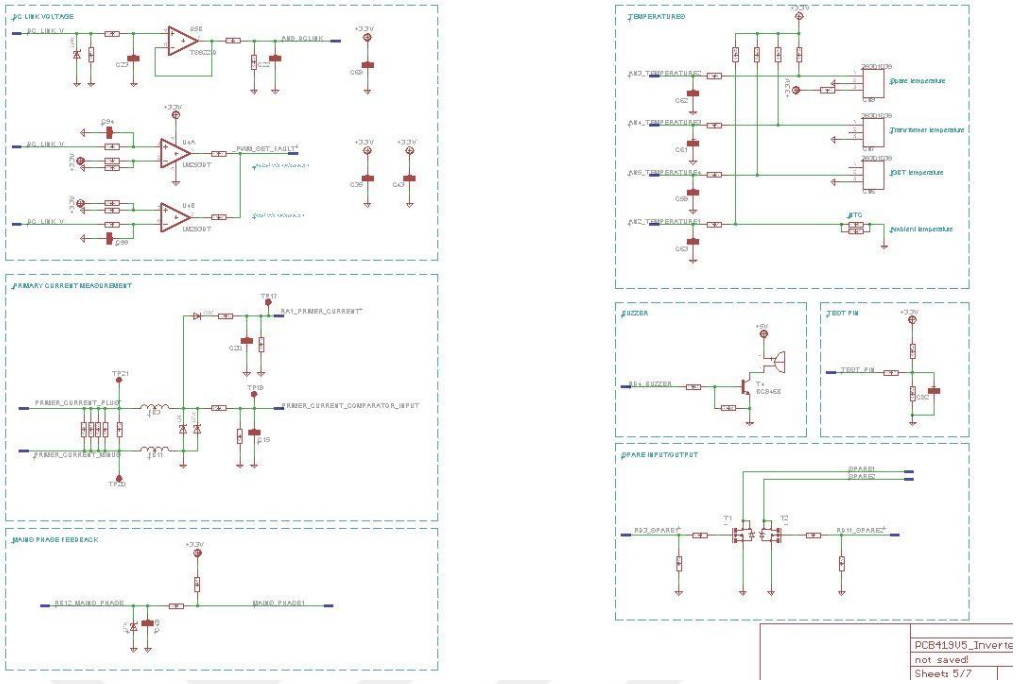

Figure 3.11 System feedbacks

In the welding machine, both external watchdog (STM6321) and internal watchdog (firmware) timers are used for security reasons. Also, there is a Flash Random Access Memory (FRAM) is used to calculate and save some parameters. Welding has different parameters such as arc force, hot start, burn back, soft start, crater fill and many programs depending on the material to be welded and electrode or wire to be used. In Figure 3.12, FRAM block and watchdog block schematics are given. FRAM is communicating with MCU with the Serial Peripheral Interface (SPI) module. SPI module has a master and slave and, in this study, MCU works as a master and the FRAM works as a slave to save the welding related parameters and tells the MCU whenever asked.

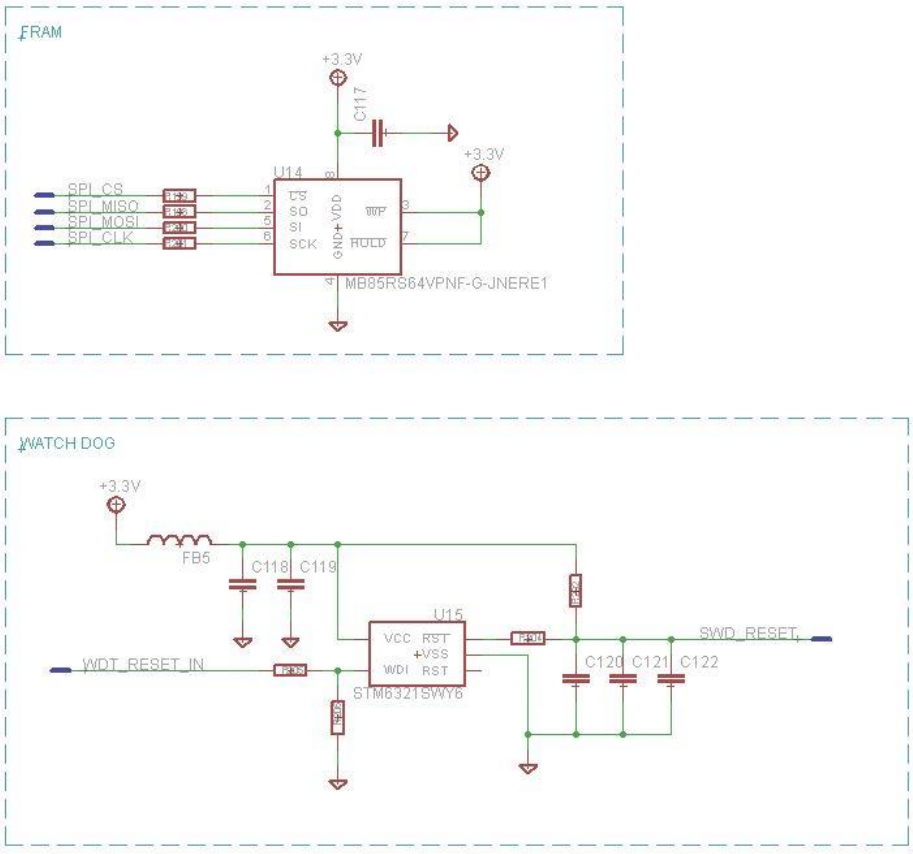

Figure 3.12 FRAM and watchdog timer block

3.4 Driver Stage Design

In driver stage design isolated gate drivers are used since all the IGBT's are n channel and needs to be driven separately. The pulse transformers selected are 60403-F4185-X04. The MCU outputs low voltage value (3. 3V) and in order to drive IGBT's a pulse transformer is needed. In order to drive the pulse transformer, a high voltage driver needed. Figure 3.13 shows the isolated gate driver circuit. The isolated gate driver has an isolation level of 4kV. Whenever the PWM signal is high, the pulse transformer works as a single switch forward converter and the 2 cross IGBT start conducting. When the PWM signal is low, the pulse transformer resets itself through a Zener diode and the U3 starts conducting which drives both T11 and T12 to discharge the gate emitter capacitance of the IGBT. Note that this circuit is for a single IGBT. D26 and D25 is 3.1V Zener diodes and C18 is a 100nF capacitor. When the PWM signal is

high, Zener diode charges C18 to 6.2V, which is used during the PWM low signal to produce a negative voltage in order to turn off the IGBT as fast as possible. Current tailing is minimized with this configuration. C9, R16 and R25 works as a snubber circuit. R51 and R52 are gate resistors to limit charging and discharging current. On the left-hand side there is a high speed MOSFET which is used to drive pulse transformer. The switching speed of the MOSFET also determines the welding quality. Since welding is a process with many short circuits, the turn off time of the IGBT should be small as possible. The MOSFET also has a parallel Zener diode to its drain to source terminals. When the MOSFET is in off state, the pulse transformer resets itself on this Zener diode. As a result, MOSFET and the IGBT gate signals look similar.

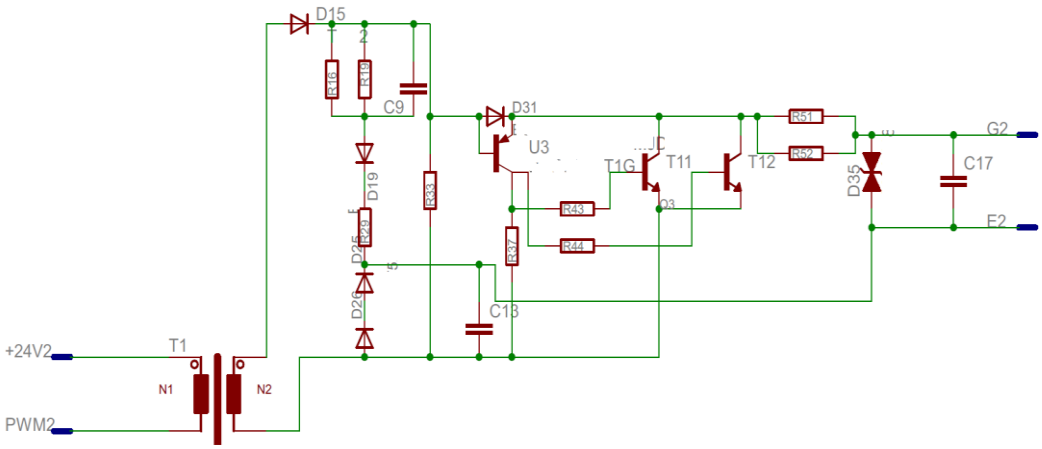

Figure 3.13 Gate driver circuit

CHAPTER 4

SIMULATION AND CONTROLLER DESIGN

In the simulation stage, Power Simulation (PSIM) software is used for its advanced controller design tools. Before any controller design the first thing is model the system of welding machine. Welding process acts as constant resistance at steady state.

4.1 Open Loop Design

4.1.1 Full Bridge Open Loop Design

In power stage open loop design, all the parameters are set as found at previous stage. In Figure 4.1, DC/DC full bridge inverter supply of the welding machine part is shown.

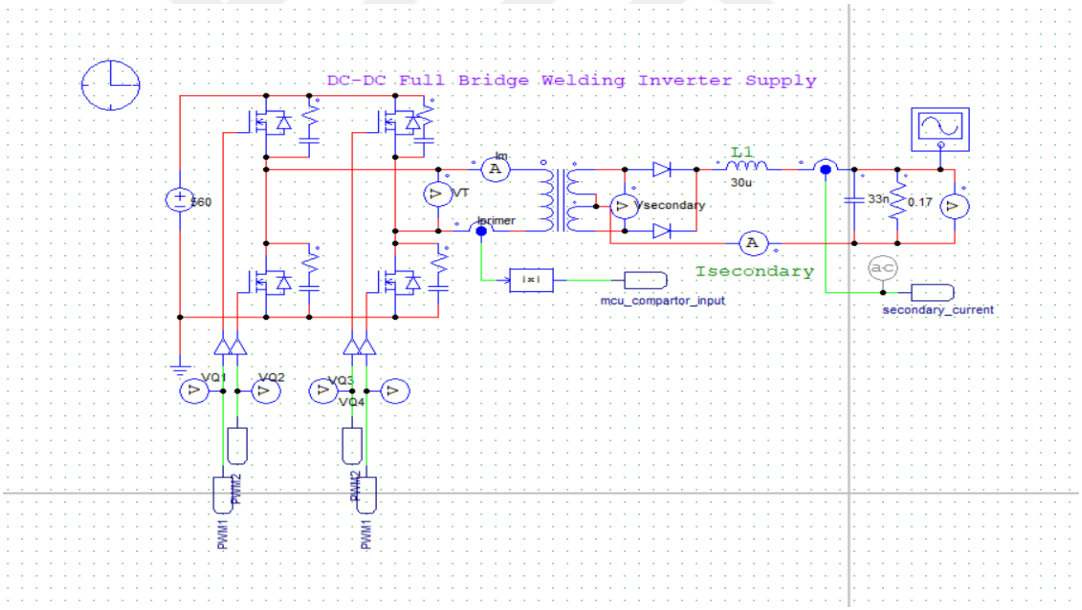

Figure 4.1 Full bridge open loop schematic

In Figure 4.2, open loop simulation result of weld current is given.

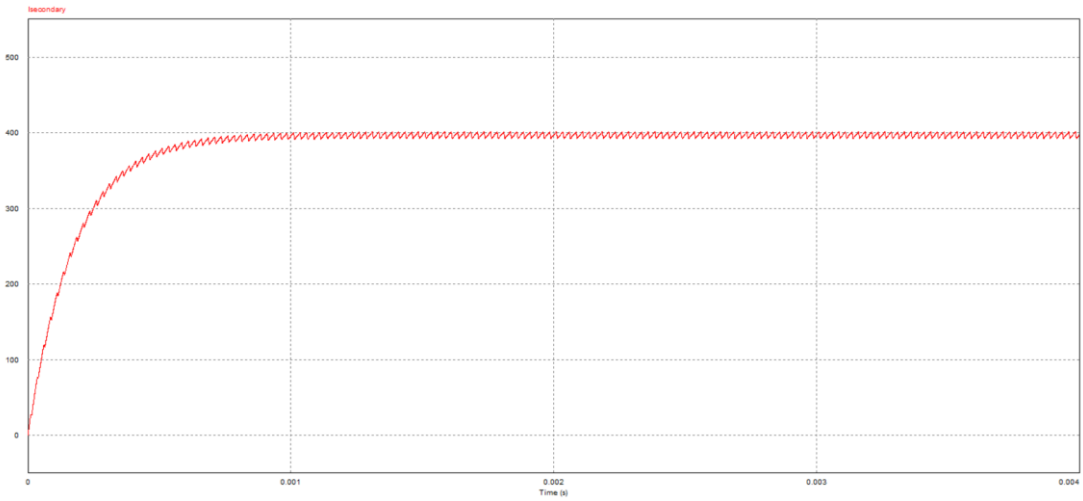

Figure 4.2 Open loop simulation result

Current ripple of the secondary side (output current) approximately 8.37A and is shown in Figure 4.3.

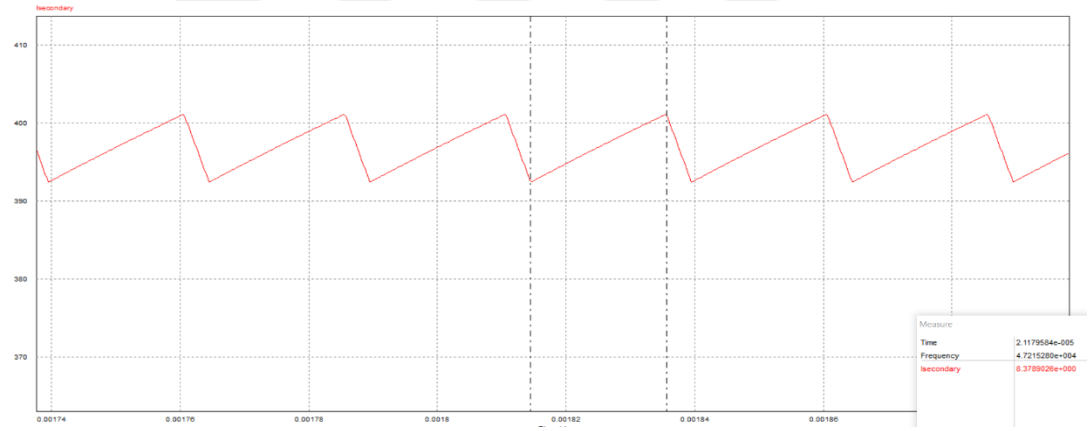

Figure 4.3 Current ripple of weld current

The switches are pulsed in complementary mode and hard switching technique is applied in welding machine. To realize push-pull mode PWM in simulation, a flip flop is used. In Figure 4.4, realization of push-pull mode PWM is given.

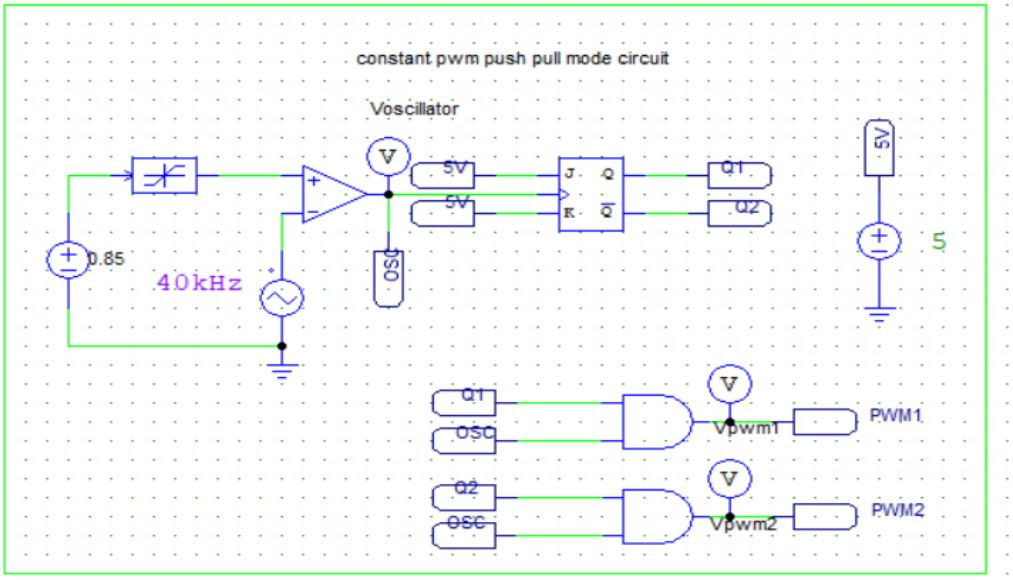

Figure 4.4 Constant PWM push-pull mode circuit

4.1.2 Buck Converter Open Loop Design

Full bridge DC/DC converter is basically a buck converter with transformer, in this thesis the controller is designed according to buck converter. But because of the push-pull configuration in full bridge the filter inductor sees twice of the frequency of the buck converter and to give exact result in buck converter design switching frequency is set to 40 kHz. In Figure 4.5, open loop design of buck converter is given.

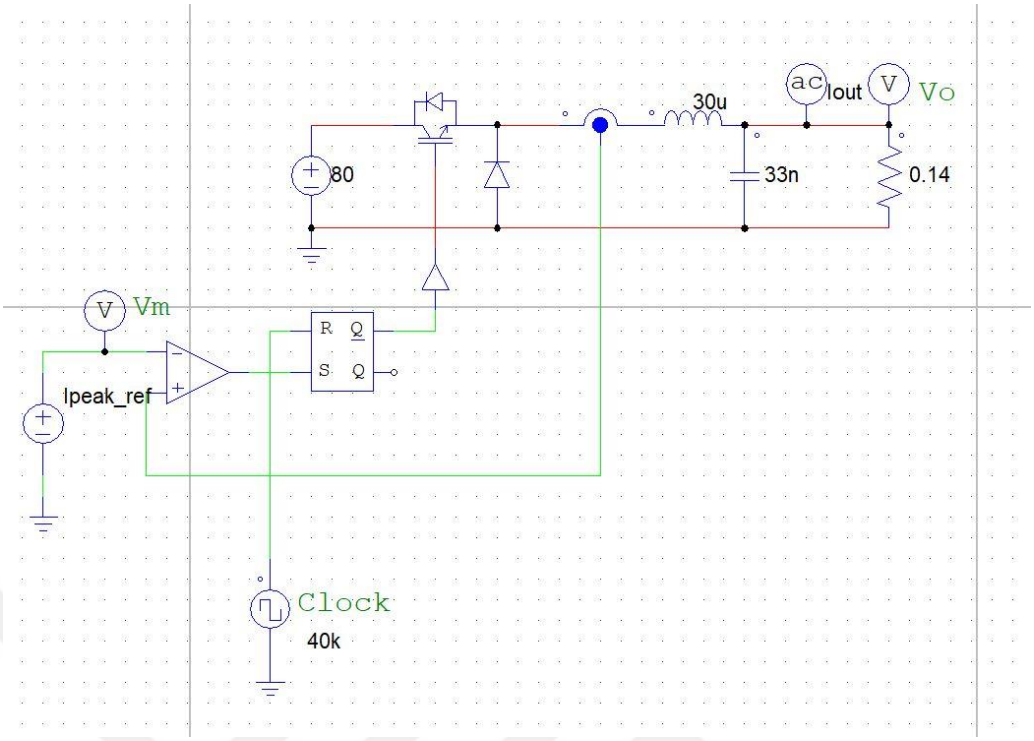

Figure 4.5 Buck converter current mode control open loop design

4.2 Closed Loop Design

4.2.1 AC Sweep

PSIM has an AC sweep property that allows user to perturb duty signal at different frequencies and magnitudes and observe the output. A small disturbance is added to the control signal and this signal is observed in a wide range. The proposed AC sweep method simulation is shown in Figure 4.6.

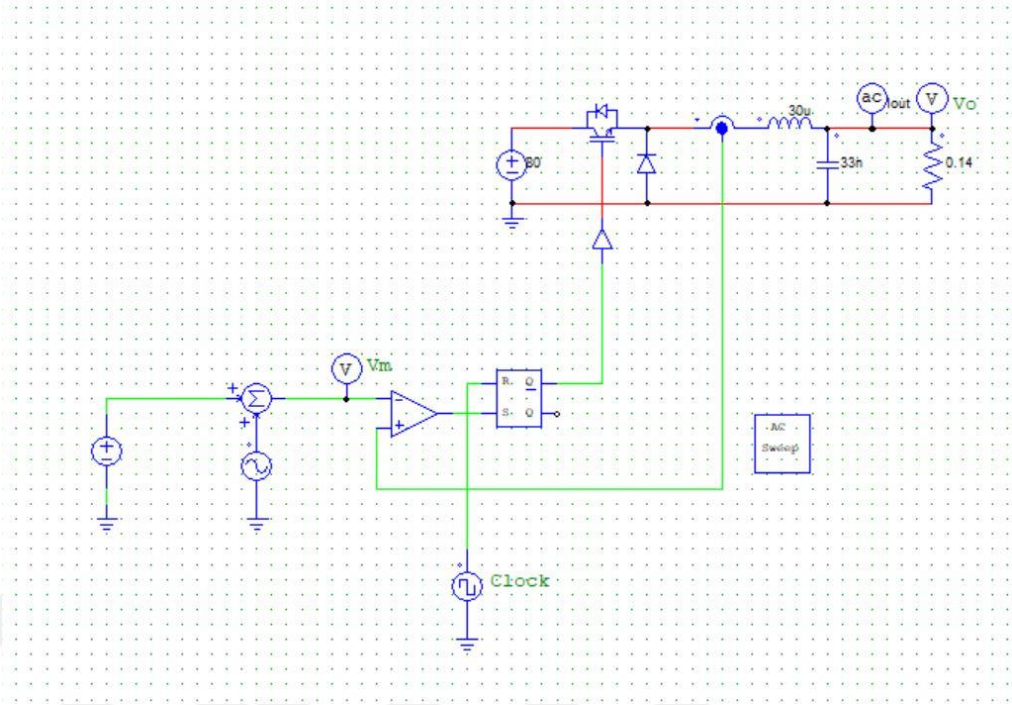

Figure 4.6 Buck converter AC sweep simulation

In Figure 4.7, AC sweep parameters of the simulation are given. Maximum frequency is selected as 16 kHz in order to stay in stable region.

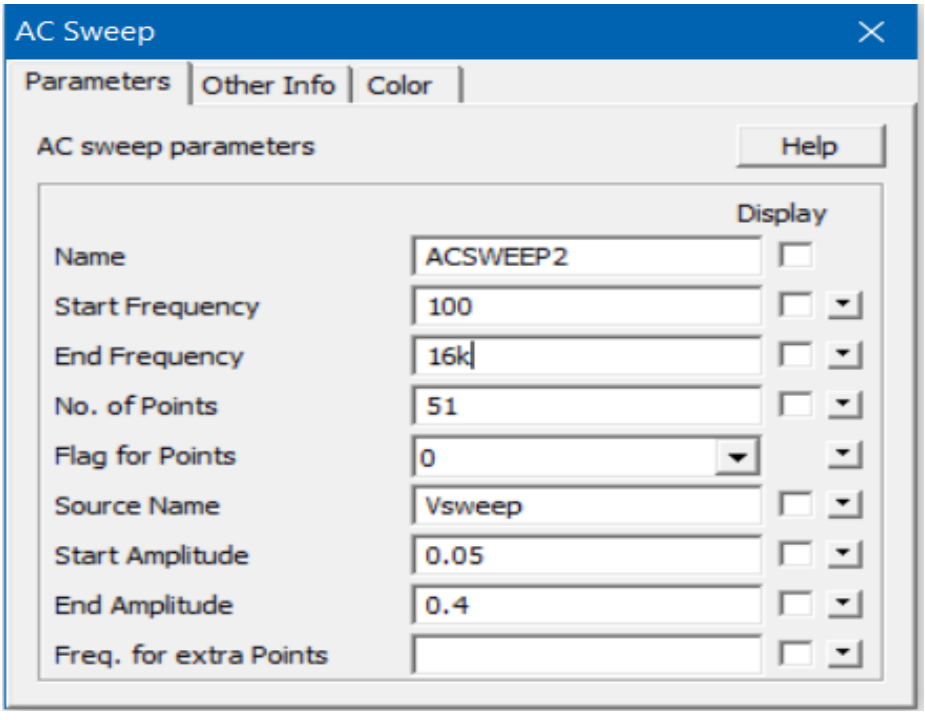

Figure 4.7 AC sweep parameters

After AC sweep the duty to output transfer function result is given. Since the output capacitance is 33nF to achieve the welding quality, the effect of the second pole is far away from the controller bandwidth and the system is stable. In Figure 4.8, the result of AC sweep is given.

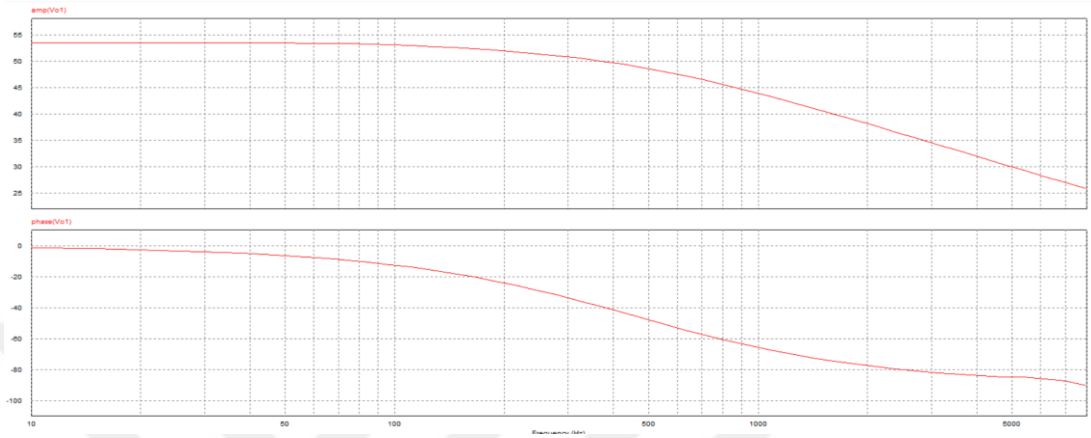

Figure 4.8 AC sweep result

4.2.2 Smart Control

PSIM has a tool, smart control, which takes bode plot result of the circuit and lets you design a controller. In Figure 4.9, designed transfer function and its properties are given.

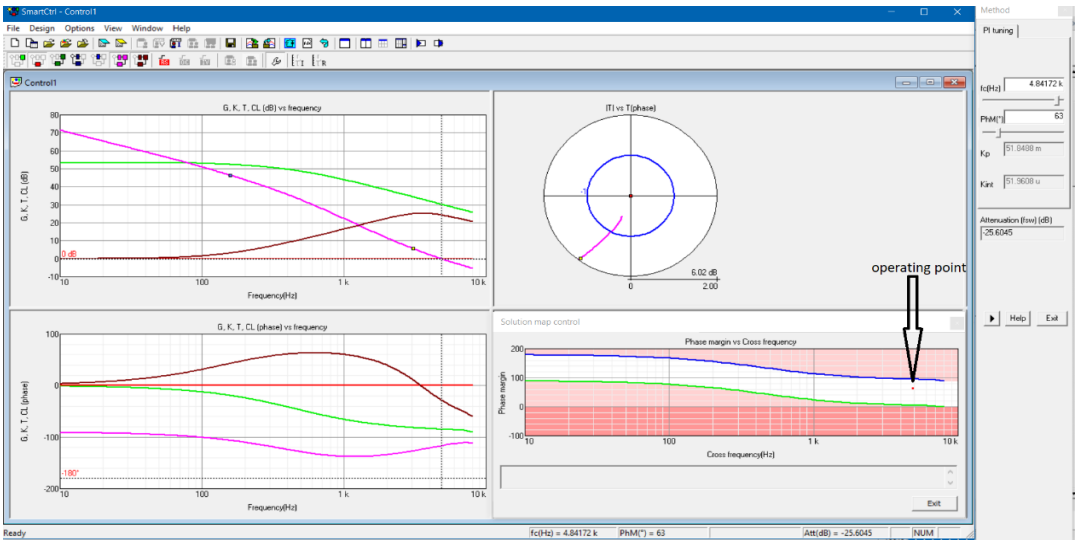

Figure 4.9 Smart control controller design

Smart Control defines parameters in s domain and sends the controller to the power stage. In Figure 4.10, 200A output current graph and parameters are given.

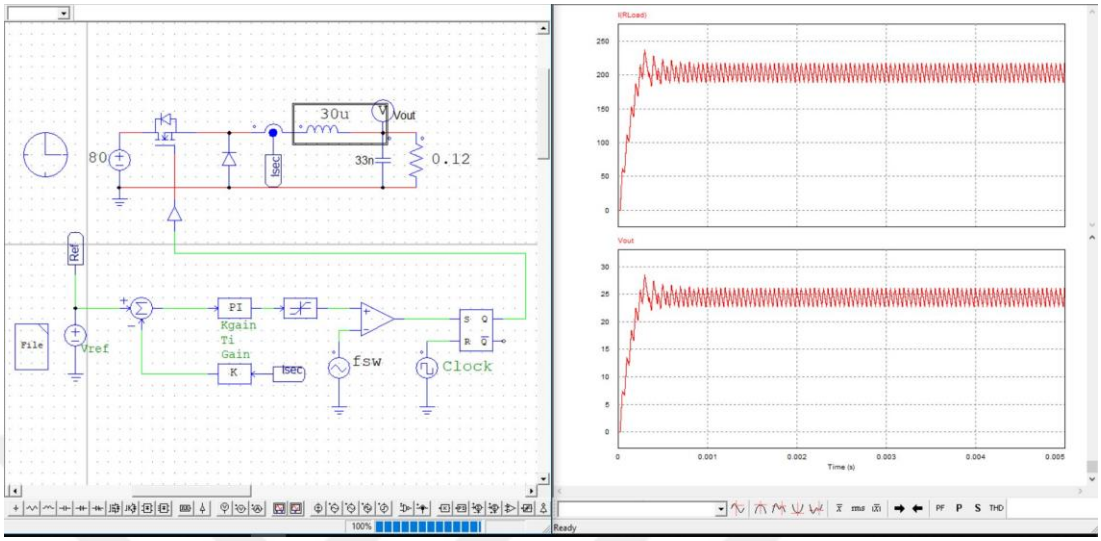

Figure 4.10 200A output current controller s domain design

4.2.3 Analog Controller Design

After designing the controller in s domain, the same controller can be designed with using analog circuit. In Figure 4.11, analog circuit simulation of the inverter.

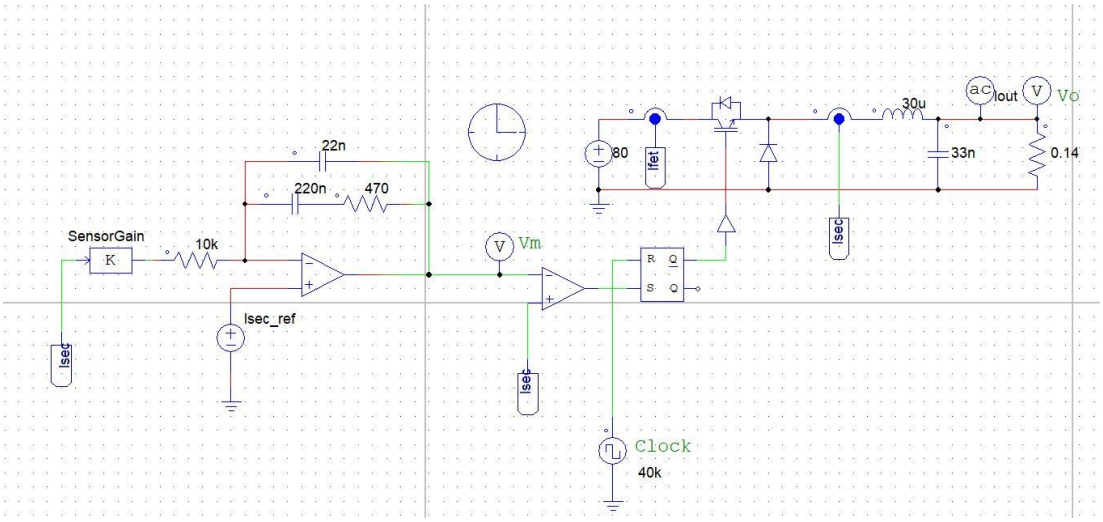

Figure 4.11 200A output current controller analog controller design

4.2.4 Digital Controller Design

After analog design using C-block properties of PSIM a different algorithm can be studied. In this thesis, secondary current feedback is taken and regulated and used as current reference. Figure 4.12 shows the digital controller of the buck converter.

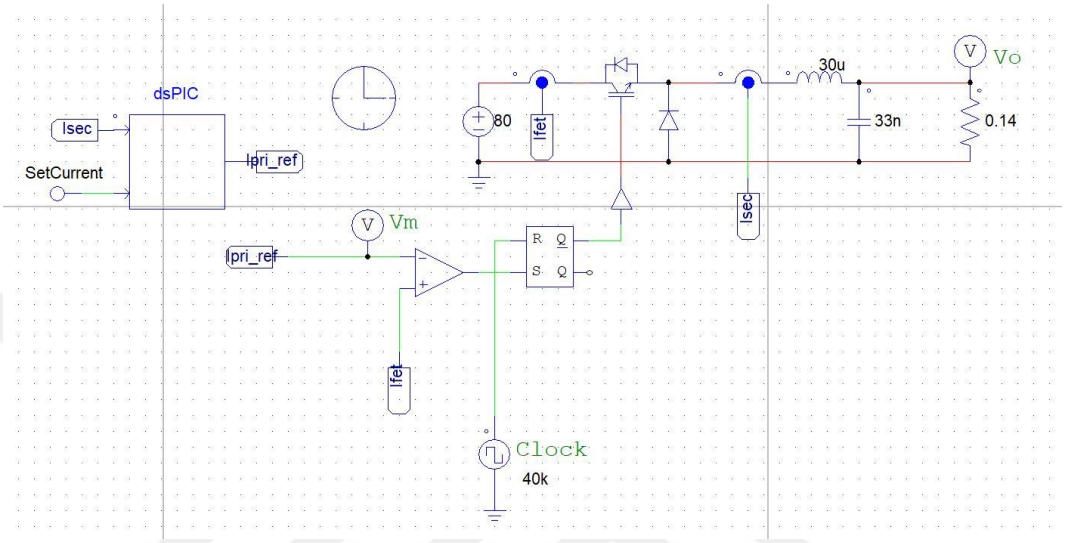

Figure 4.12 Digital controller schematic of buck converter

Since the mathematical model of the full bridge converter and buck converter are the same, similar code can be implemented for the full bridge converter. The simplified C-block in PSIM whose period is 1us is interrupted every 100us and controller is used for producing an IGBT current reference. This current reference is used in Digital to Analog Converter (DAC) pin of the dsPIC33EP128GS806 which is internally connected to the negative pin of the internal fast comparator. The comparator resets an internal latch as soon as the feedback signal passes the reference, which turns off the PWM signals immediately. In Figure 4.13, digital controller part of the whole system simulation is given. Some parts of the schematics are embedded inside dsPIC33EP18GS806 but for the sake of the understandability, a detailed simulation is presented. Also, as mentioned before, output current is sensed with current sensor and the current sensor has an offset which may change in time or temperature. To read the output current precisely, a buffer circuit used, and the subtraction is done inside firmware.

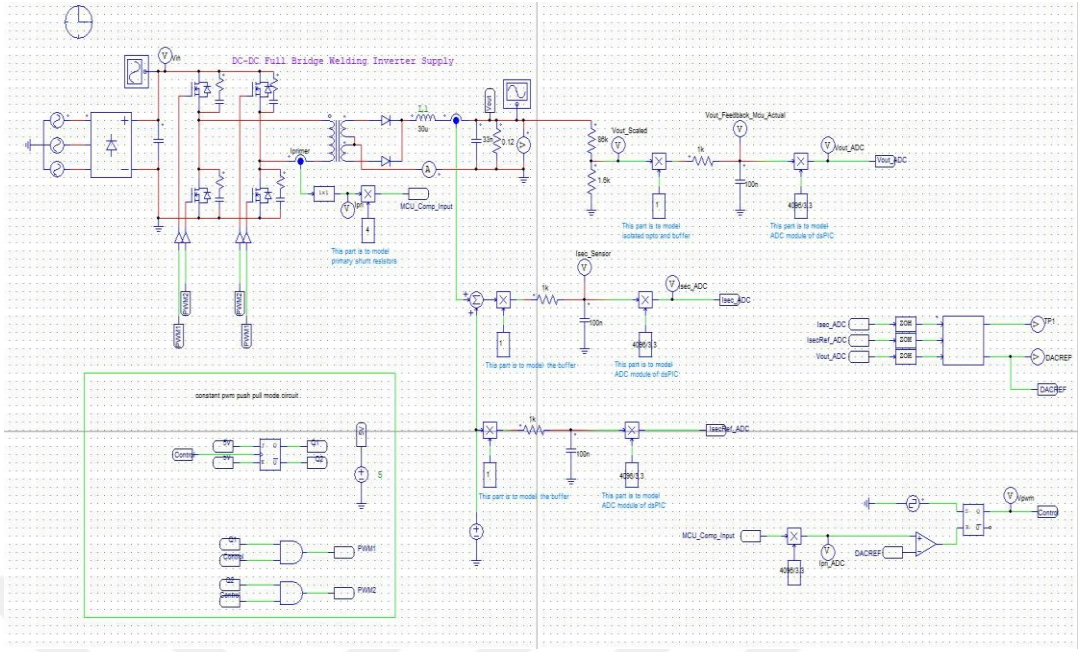

Figure 4.13 Controller stage of full bridge inverter

The control algorithm is same as the buck converter control algorithm. Below detailed code is given.

```
//Variables for Reference Calculation

float error=0;

static float integrator=0;

static float Ki=20;

static float Kp=30.0f;

static float PIDout=0;

static float secondary_current_feedback=0;

static float secondary_current_set_value=0;

#define MAX_PID_OUT 400.0f

#define MIN_PID_OUT 0
```

```
#define MAX_INTEGRATOR_OUT 500.0f

#define MIN_INTEGRATOR_OUT -500.0f

#define ERROR_MAX 50.0f

secondary_current_feedback=x1;

secondary_current_set_value=x2;

//Reference

error = secondary_current_set_value-secondary_current_feedback;

if (error > ERROR_MAX ) error = ERROR_MAX;

integrator = integrator + delt*error*Ki;

if                  (integrator>MAX_INTEGRATOR_OUT)
integrator=MAX_INTEGRATOR_OUT;

else if             (integrator<MIN_INTEGRATOR_OUT)
integrator=MIN_INTEGRATOR_OUT;

PIDout=integrator+ Kp*error;

İf                  (PIDout>MAX_PID_OUT) PIDout=MAX_PID_OUT;

else if             (PIDout<MIN_PID_OUT) PIDout=MIN_PID_OUT;

y1=PIDout;
```

The simulation results of 350A output is given in Figure 4.14. The set current value is adjusted to 350A.

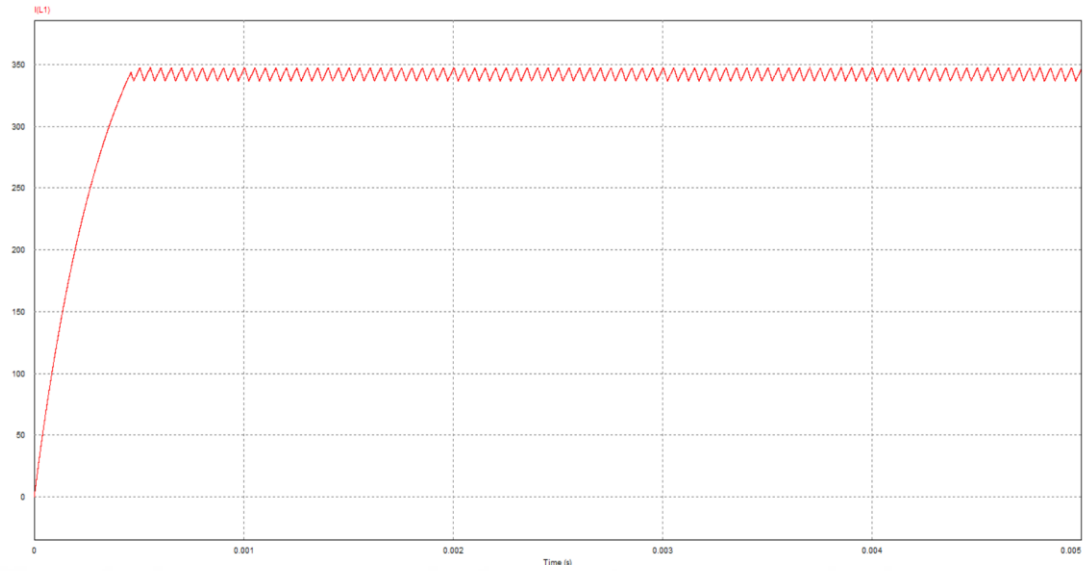

Figure 4.14 350A output of full bridge simulation

4.3 Arc Force Algorithm

Although the simulation results seem okay, in real life welding is complicated and do not show the same resistance all the time. Especially in stick welding mode, short circuits are often present because of the electrode becoming very close to the workpiece, which means that the arc length is smaller than it needs to be for a high quality welding. When a short circuit situation occurs, the controller needs to sense it and must increase the current reference in order to increase the arc length. This process is called arc force. When the arc length reaches its normal length, the current reference goes back to its normal value. The same situation happens in the welding starting process. When the welding starts, the arc length is shorter than its usual length, so the current reference should be set a bigger value for a period of 100ms to 200ms. This process is called hot start. Both the hot start and arc force algorithms are easier and flexible for the user when designing a welding machine with a digital controller. In Figure 4.15, arc force configuration is given.

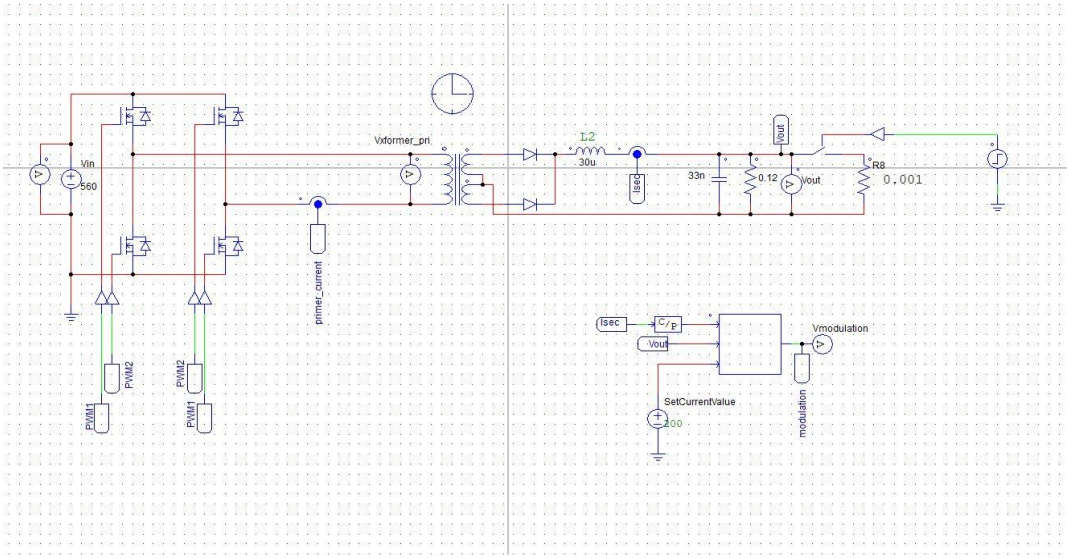

Figure 4.15 Welding short circuits simulation

The simulation starts and after 1ms welding is simulated as short circuit. Then the set current of the welding is increased by 20%. The code is given below.

```
if(outputVoltage<5) {

secondary_current_set_value=1.2*x2;

}
```

In Figure 4.16, the resulted current waveform is given.

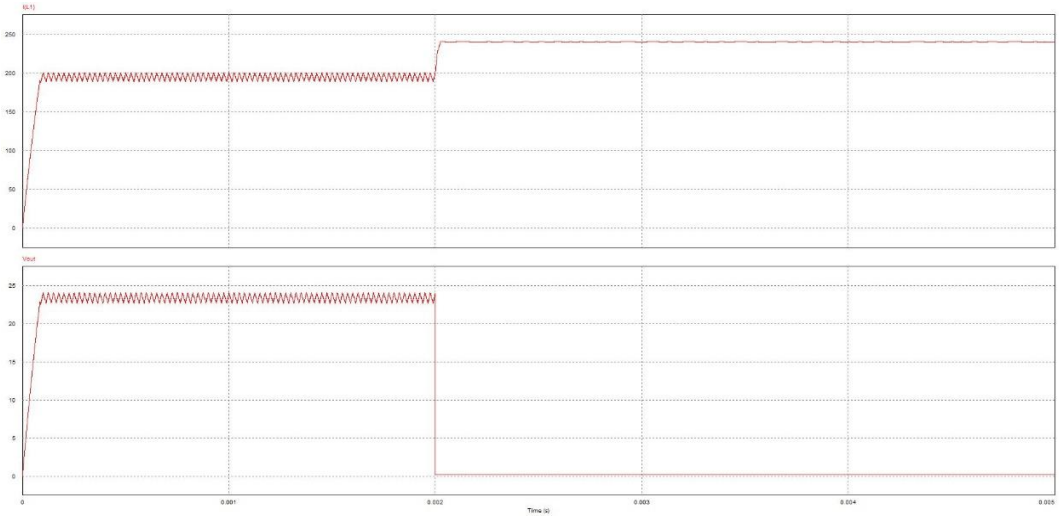

Figure 4.16 Arc force implementation results

4.4 Hot Start Algorithm

When welding is at starting process to anti sticking of electrode to workpiece it needs more current then set value to create enough arc length. So, the algorithm is changed. At start the welding set value is increased and after certain time arc force algorithm is implemented. In Figure 4.17, the hot start and arc force algorithm results are given.

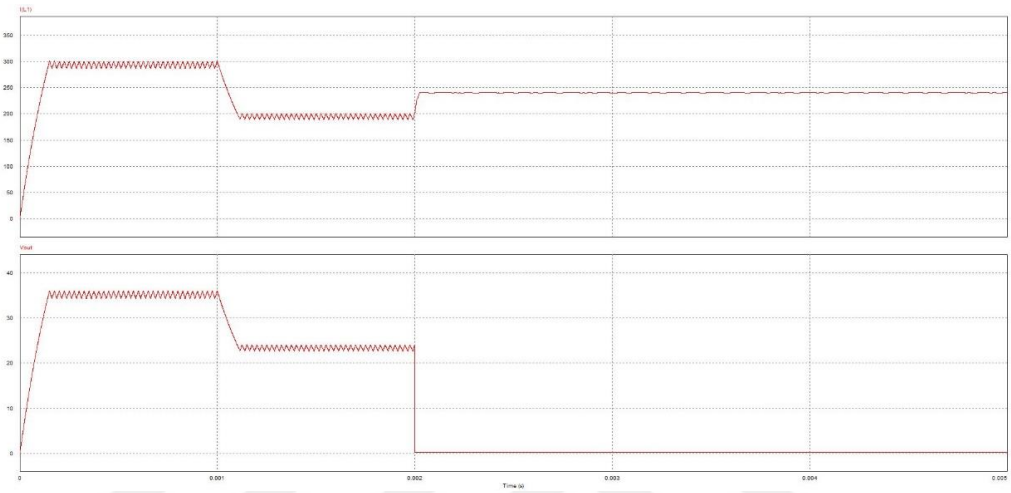

Figure 4.17 Hot start implementation results

The code is again changed.

```
if(t<0.001) secondary_current_set_value=1.5*x2;

else

{

if(outputVoltage<10) secondary_current_set_value=1.2*x2;

}
```

The algorithm doesn't look at output voltage at first and set current value is 50% is increased. After 1ms the arc force algorithm is executed. One important note is that, when the welding occurs even though short circuits are occur, as soon as arc force algorithm applied the arc length becomes normal again.

CHAPTER 5

RESULTS

5.1 Experimental Results

In this chapter, simulation results are implemented on a real time MCU and various experiments are made. Figure 5.1 shows the system overall look.

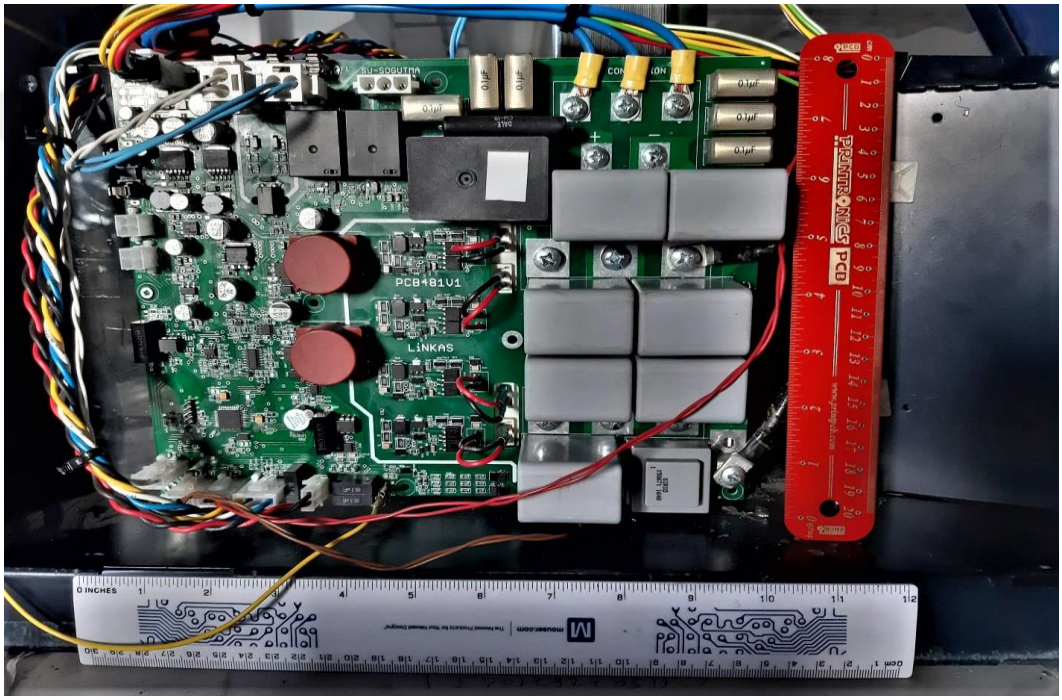

Figure 5.1 System overall look

The control method which is tested in simulation environment is also implemented using MCU which controls the implemented welding machine. The welding machine works in output current control mode. The used MCU has an internal configurable logic cell and current limiting properties in its PWM module. Figure 5.2 shows the PWM signal, current reference from DAC out pin of MCU which realizes the current limiting. Whenever the current feedback from power circuit exceeds the current reference value from the MCU, which is compensated output of output current loop, the latch resets and PWM signals are set low. Each period, PWM clock sets the latch and PWM signals are high again. The ultra-fast comparator has also a blanking

capability. When the switches are on state, the current may go to a higher value than expected. In order not to set the latch incorrectly, a small blanking time (around 100ns) is added to the fast comparators in the code.

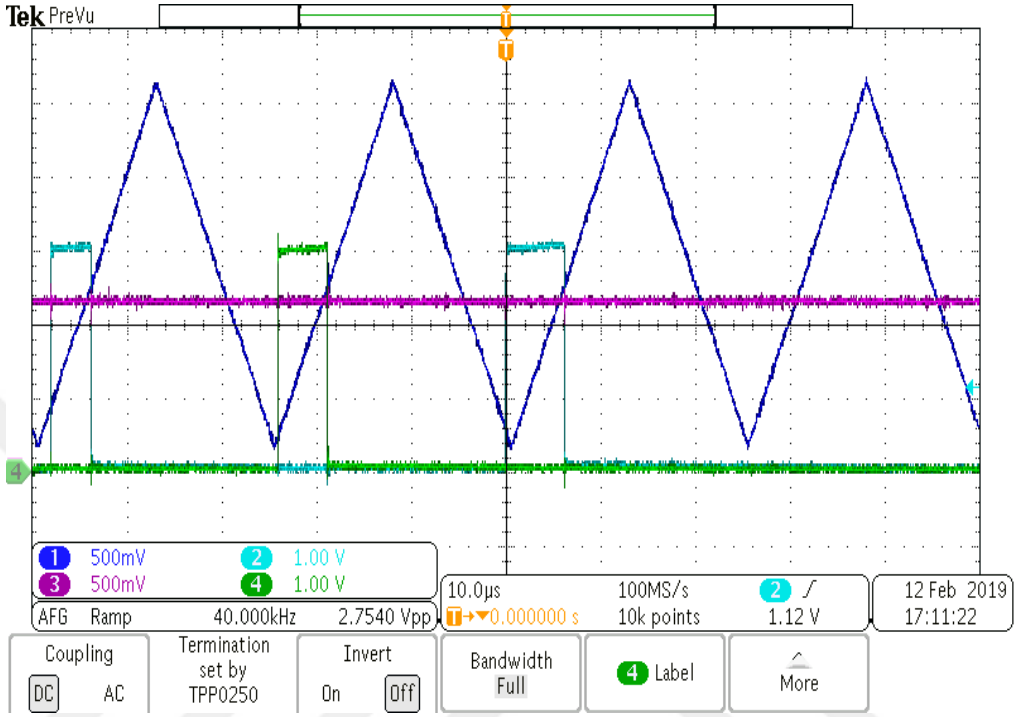

Figure 5.2 MCU peak current control method

In Figure 5.2, two separate push-pull mode PWM signals are presented. Whenever the current feedback from power circuit exceeds the current reference value from MCU, which is compensated signal of secondary current, the latch resets and PWM signals are sets low. Each period PWM clock sets the latch and PWM signals generated again. In Figure 5.3, pulse transformer input signals are shown.

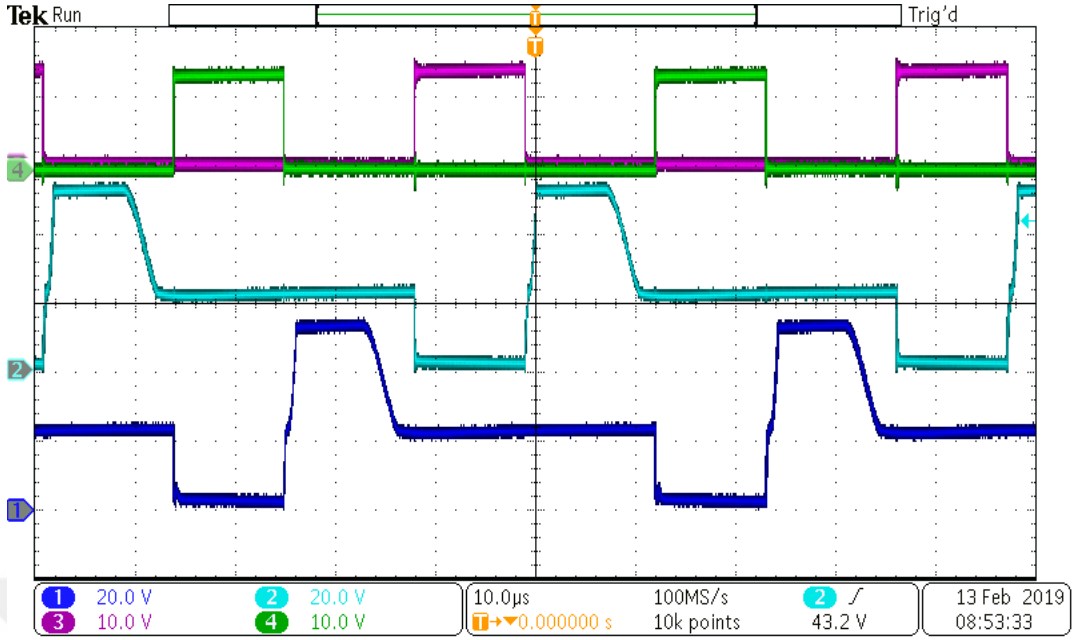

Figure 5.3 Pulse transformer input signals

In this study, the switching is realized with hard switching technique. The upper left and the bottom right IGBT's are controlled with one PWM signal. The bottom left and the upper right signals are controlled using 180 degrees phase shifted version of the first PWM signal. Figure 5.4 shows the primary terminal voltage of the full bridge transformer. A strong ringing is present because of the hard switching technique. Magnetizing inductance of the high frequency transformer is calculated as 2mH, and the transformer works in continious conduction mode. Figure 5.4 also shows the primary current of the transformer which is sensed using a current transformer. The output of the current transformer is recitifed with a diode bridge. Than the current is converted to a voltage signal using through current shunts equivalent to 3ohms. The current shunt resistors are selected to stay in safe region without sacrifising the current resolution. On the secondary side, filter inductor is selected as 30uH allowing a harder current ripple in order to achive a good quality welding. Otherwise a bigger filter inductor with a higher value would slow down the welding response. There is a small capacitance at the output, which is 33nF to filter the high frequency signals of the welding. Conventional welding machines are usually around 50% to 70% efficient whereas in this particular inverter based welding machine 84% efficiency has been observed. Using the isolated CAN module the inverter welding machine can be used

with robotics arms and many Programmable Logic Controller (PLC) modules with a CAN module of 250 kHz baudrate.

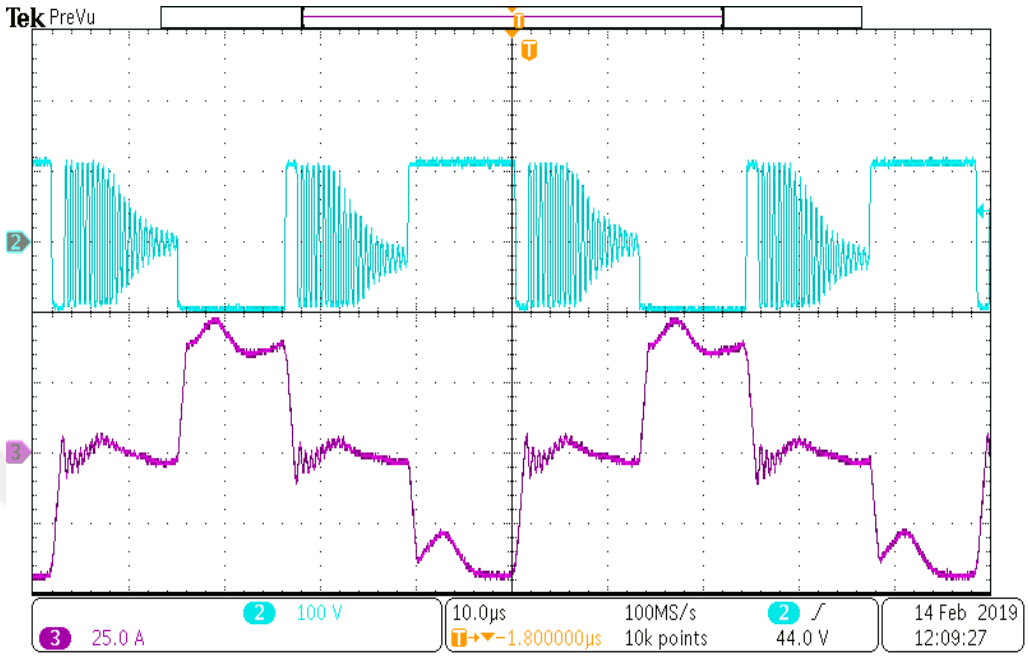

Figure 5.4 Transformer primary current and transformer primary voltage

IGBT gate driver circuit, control circuit and power circuit are implemented as close as possible to each other so that noise effects are minimized. The input six pulse diode rectifier, IGBT and output diode rectifier circuit are all connected to same heatsink. The control card and the power card have different grounds and all the feedback signals are either isolated with a transformer or using linear optocouplers. Figure 5.5 shows the 400A output current of the real circuit. Since there is no capacitor at the output in order to control the output current directly there should be no bulky capacitors at the welding terminals, the output voltage is not stable but in order to achieve a good quality welding this process is necessary. The filter inductor works as expected to minimize current ripples at the output. Too much filtering would lead the slow response, and this is a trade off in welding machines.

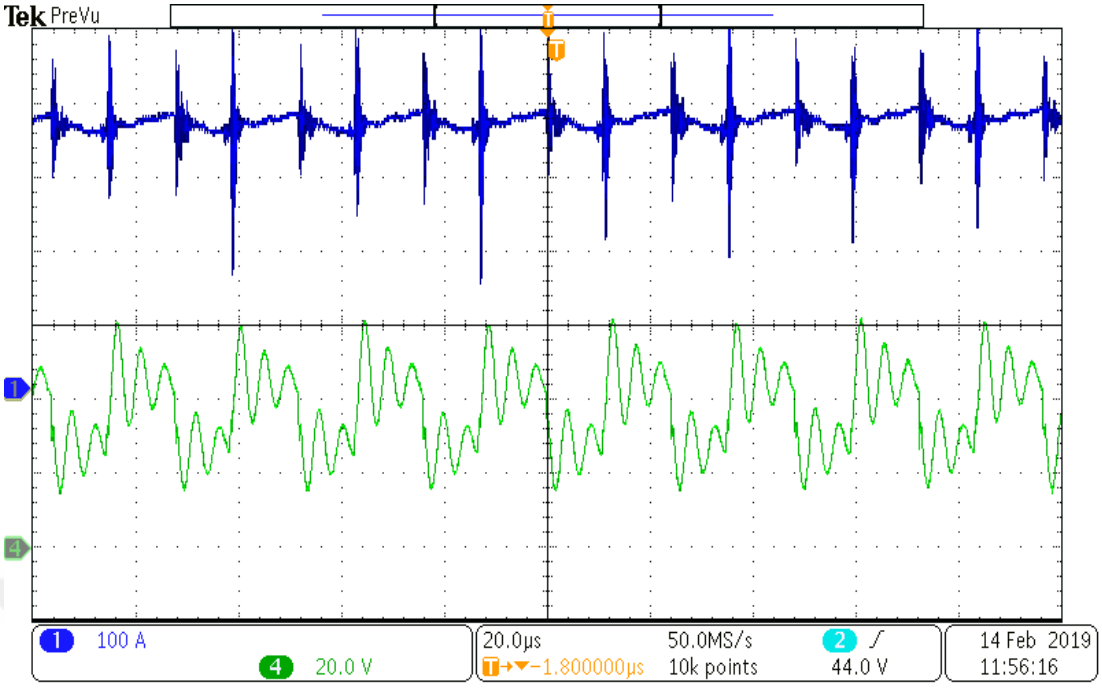

Figure 5.5 Welding current and voltage

As mentioned before, before the welding starts, there is a process called hot start. In Figure 5.6, hot start of the welding process is shown.

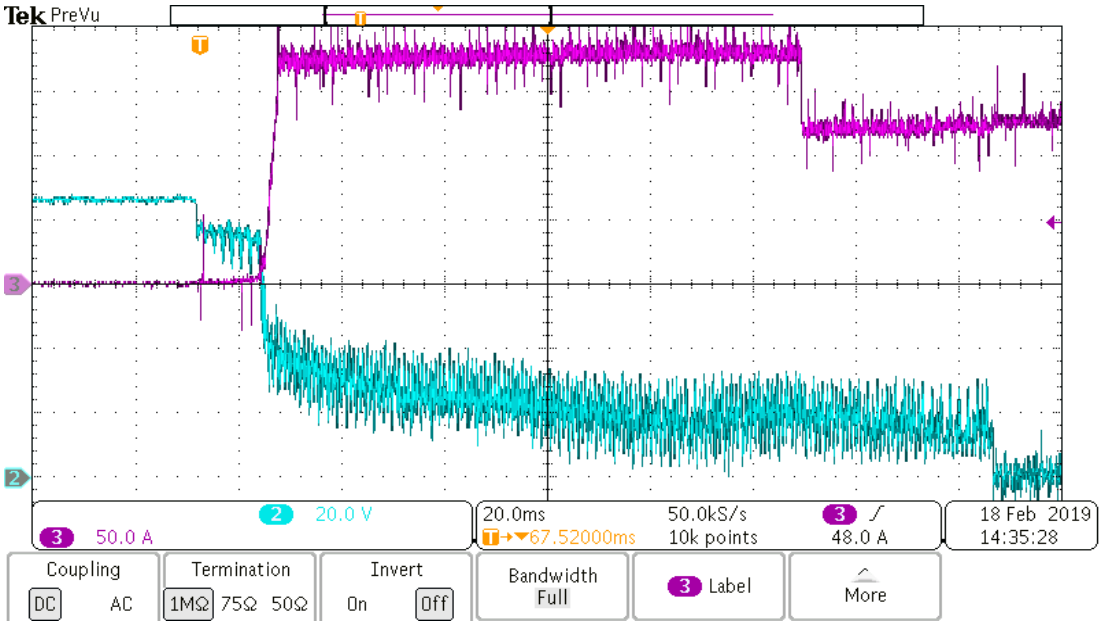

Figure 5.6 Hot start

As seen in Figure 5.6, the welding process starts around 80V and the current reference is much more than what is set in the machine. After 100ms, control loop takes over and current is regulated to its set value. In Figure 5.7, another welding parameter, arc force is shown.

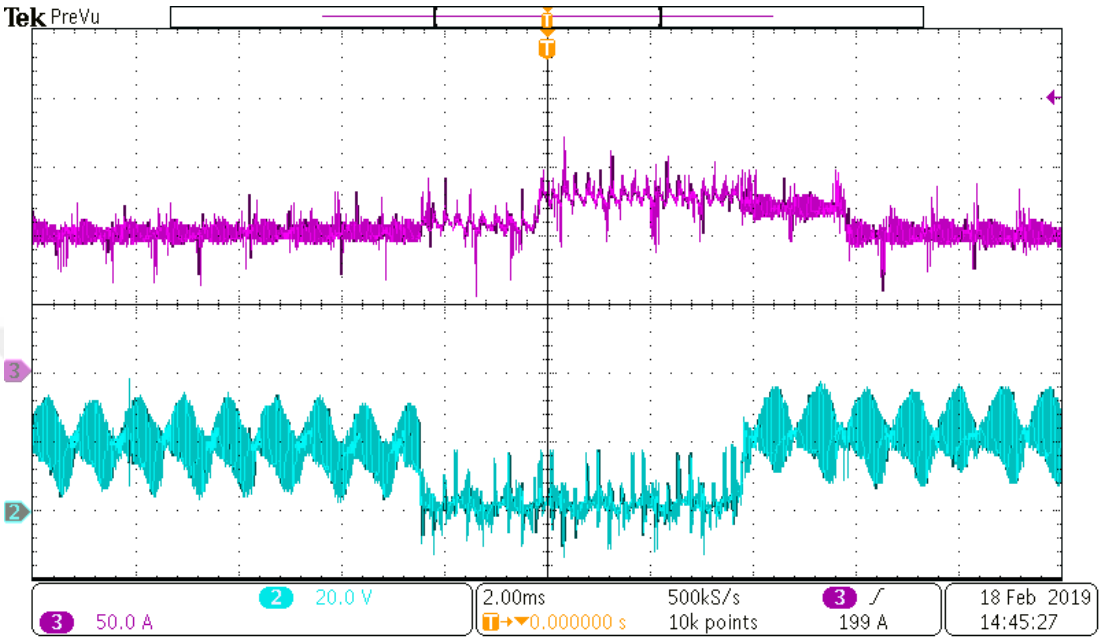

Figure 5.7 Arc force

Whenever the controller senses that output voltage is lower than what it needs to be (when short circuit like situation occurs) the current reference is set higher in order to increase to arc length to achieve good quality welding. The response time is around 1ms which is expected since the crossover frequency is around 1 kHz. A faster response would lead more current peaks in welding, which is an unwanted situation.

In this thesis, the nominal current of the welding machine is selected to be 400A at 65% machine operating duty cycle. This means that, the machine can give 400A for 65% of the time and the needs a rest for 65% of the time. The machine is operated for 6 minute 30 seconds during a total operation time of 10 minute. This cycle is repeated for a full day. On the other hand, the machine should be able to give 300A without the need of a rest. Figure 5.8 shows the thermal image of the welding machine circuit after 1 hour of continuous 300A operation. The ambient temperature was around 20℃ for this test.

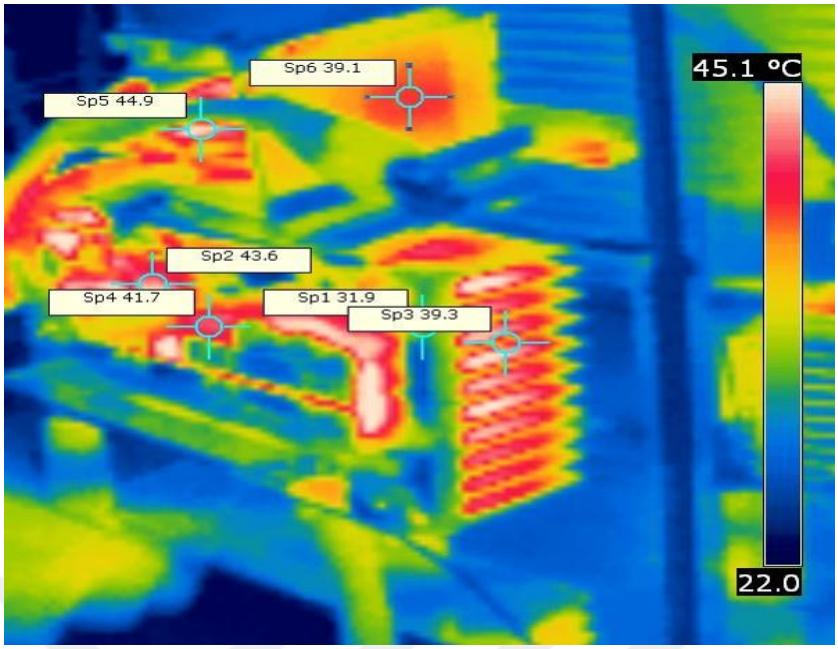

Figure 5.8 Temperature distribution of welding machine

The machine is designed to give its maximum efficiency at 400A which results in 17.2 kW of power. Figure 5.8 shows the power vs efficiency graph of the welding machine.

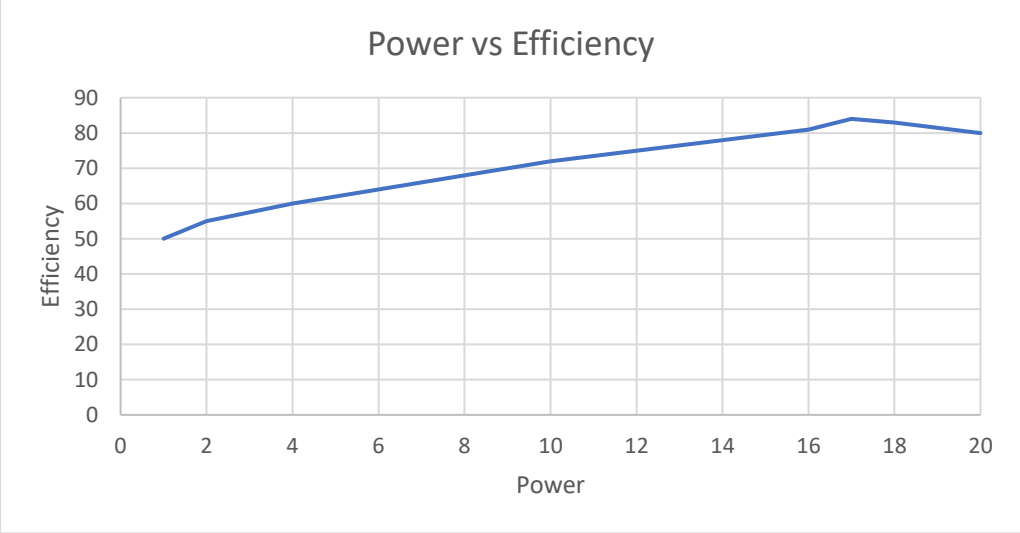

Figure 5.9 Power efficiency chart

The welding quality is defined based on some certain requirements. The first requirement is, that the finished welded product should be in accordance with the aimed joint dimensions. The second one is that the joint point should be strong and there should be no cracks on the surface. Although these properties are tested with

welding inspection machines, the appearance of the welded workpiece also gives an idea to the user. Figure 5.10 shows a workpiece welded with GMAW technique.



Figure 5.10 Welded workpiece

The welding workpiece has a uniform wavy surface, where width and height are very similar. Also, full thermal penetration has been achieved since the edges are in a good shape. For these reasons the welding can be considered as a good quality welding.

CHAPTER 6

CONCLUSION

Inverter type welding machines popular issue nowadays due to high efficiency solutions and expected to be more popular in the next years. In this thesis, simulation power and control circuit design and experimental results are presented for 17.2 kW inverter type welding machine. The controller design is both simulated analog and digital manners. The efficiency of the machine is tested in many power values and the maximum efficiency is 84%. The controller of dsPIC33EP128GS806 was used is a new generation of the controller for DC/DC converter that has peripheral of high speed feedback response that not need extra analog parts to control the PWM according to the dynamic of resistance of the welding itself. The method of peak current control is implemented to project it works successfully for welding machine application. Designed machine that uses digital control. In the market a few machines use this technology to full bridge topology because most of the welding machines use analog circuit or fixed PWM modules like current mode PWM controller for the controller.

REFERENCES

[1] The History of Welding (n.d.). Retrieved March 12, 2018 from https://www.millerwelds.com/resources/article-library/the-history-of-welding.

[2] Welding. (n.d.). Retrieved April 2, 2018, from http://www.wikizero.biz/index.php?q=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnL3dpa2kvV2VsZGluZw.

[3] Koparan, A., Implementation of 200 Ampere, High Frequency Switching DC and AC/DC Arc Welding Machine, M. Sc. Thesis, Gazi University, 2010.

[4] Thomesen, J. S., Advanced Control Methods for Optimization of Arc Welding, PhD Dissertation, Aalborg University. 2005.

[5] Blasco N., Martinez A., F. J. Perez, J. E. Vicuna, I. Lacamara, J. A. Oliva, Evaluation of Power Converters for MMA Arc Welding, IEEE International Symposium on Industrial Electronics, 2007.

[6] Bjorgvnsson J. B., Cook G. E., Andersen K., Microprocessor Based Arc Voltage Control for Gas Tungsten Arc Welding Using Gain Scheduling, Int. J. Modelling, Identification and Control, 1 (2), 133-139, 2006.

[7] Sartipizadeh H., Haeri M., Control of Gas Metal Arc Welding by an Extended DMC, IEEE International Conference on Control Applications (CCA), 2012.

[8] Xiao B-L, Wang S-H, Lui Z., & Zhu G-R, Bai L., The Sliding Mode Control for Arc Welding Inverter Power Source, IEEE Conference on Industrial Electronics and Applications, 2008.

[9] Azcondo F. J., Casanueva R., & Navarro-Crespin A., Performance Improvements in an Arc-Welding Power Supply Based on Resonant Inverters, IEEE Transactions on Industry Applications 48 (3), 1-6, 2010.

[10] Azcondo F. J., Casanueva R., Navarro-Crespin A., & Lopez V. M., Digital Control for an Arc Welding Machine Based on Resonant Converters and Synchronous Rectification. IEEE Transactions on Industrial Informatics, 9 (2), 839-842, 2013.

[11] Erickson, R., &Maximovic D. *Fundamentals of Power Electronics* (2nd ed.) Boston: Kluwer, 2004.

[12] Mammono, R. *Switching Power Supply Topology Voltage vs Current Mode.* Retrieved August 8, 2018, from http://www.ti.com/lit/an/slua119/slua119.pdf, 1999.

[13] Ridley, R. A New Small-Signal Model for Current Mode Control, PhD Dissertation, Virginia Polytechnic Institute and State University. 1990.

[14] Çetin, N. Design and Implementation of Advanced Pulse Width Modulation Techniques and Passive Filters for Voltage Source Inverter Driven Three-Phase AC Motors, Published master's thesis, Middle East Technical University. 2010.

[15] Magnetics in Switched-Mode Power Supplies (n.d.). Retrieved August 29, 2018, from https://www.onsemi.com/pub/Collateral/TND350-D.PDF.

[16] O'Brien, R. (Ed.). *Welding Handbook Volume*-2, (8th ed.), Miami: American Welding Society, 1991.

APPENDICES

Appendix A: Application C Code

Appendix A – Application C Code

NOTE: Some part of this code is generated using the MPLAB X IDE tool MCC

```
//------------------------------------------------------------------------------

// Description: MAIN APPLICATION

//

// LINKAS TEKNOLOJI SAN. TIC. A.S.

// ANKARA-TURKIYE

//

// (c) Copyright 2018, LINKAS TEKNOLOJI SAN. TIC. A.S.

//

// Programmers: Ertuğrul Kocaağa

//------------------------------------------------------------------------------

#include "main.h"

//------------------------------------------------------------------------------

//        CONSTANT & DEFINITIONS

//------------------------------------------------------------------------------

//------------------------------------------------------------------------------

//        FUNCTION PROTOTYPES

//------------------------------------------------------------------------------

void system_init (void);

void adc_init(void);
```

```
void cmp1_init(void);

void interrupt_init(void);

void pin_manager_init(void);

void pwm_init(void);

void spi1_init(void);

void timer1_init(void);

void ADC1_SharedCorePowerEnable(void);

void ADC1_SharedCoreCalibration(void);

void ADC1_Core2PowerEnable(void);

void ADC1_Core2Calibration(void);

void ADC1_Core1PowerEnable(void);

void ADC1_Core1Calibration(void);

void CMP1_Tasks(void);

bool CMP1_StatusGet(void);

bool CMP1_DACOutputStatusGet(void);

void CMP1_SetInputSource(CMP1_INPUT inpSrc);

void CMP1_Enable(bool enable);

void TMR1_Period16BitSet( uint16_t value );

uint16_t TMR1_Period16BitGet( void );

void TMR1_Counter16BitSet ( uint16_t value );

uint16_t TMR1_Counter16BitGet( void );
```

```
void TMR1_CallBack(void);

void TMR1_Start( void );

void TMR1_Stop( void );

bool TMR1_GetElapsedThenClear(void);

int TMR1_SoftwareCounterGet(void);

void TMR1_SoftwareCounterClear(void);

//---------------------------------------------------------------------------

//        VARIABLES

//---------------------------------------------------------------------------

typedef enum

{

        CMP1_INPUT_CMP1D  = 0x3,

        CMP1_INPUT_CMP1C  = 0x2,

        CMP1_INPUT_CMP1B  = 0x1,

        CMP1_INPUT_CMP1A  = 0x0,

        CMP1_INPUT_PGA2  = 0x1,

        CMP1_INPUT_PGA1  = 0x0

}CMP1_INPUT;

typedef struct _TMR_OBJ_STRUCT

{

   /* Timer Elapsed */
```

```
    bool                                timerElapsed;
    /*Software Counter value*/
    uint8_t                              count;
} TMR_OBJ;
static TMR_OBJ tmr1_obj;
//------------------------------------------------------------------------------
//        FUNCTIONS
//------------------------------------------------------------------------------
int main(void)
{
    // initialize the device
    system_init();
    // Functions init
    functionsInit();
    while (1)
    {
        startTimeScanFlags();
        // Telemetry Functions
        telemetryTask();
        // Application Functions
        functionsTask();
```

```
        clearTimeScanFlags();
    }
}
//---------------------------------------------------------------------------
void system_init void
{
        adc1_init();
        cmp1_init();
        interrupt_init();
        pin_manager_init();
        pwm_init();
        spi1_init();
        timer1_init();
}
//---------------------------------------------------------------------------
void adc1_init void
{
        // ADSIDL disabled; PUMPEN disabled; ADON enabled;
    ADCON1L = (0x8000 & 0x7FFF); //Disabling ADON bit
    // FORM Integer; SHRRES 12-bit resolution;
    ADCON1H = 0x60;
```

// SHRADCS 2; REFCIE disabled; SHREISEL Early interrupt is generated 1 TADCORE clock prior to data being ready; REFERCIE disabled; EIEN disabled;

ADCON2L = 0x0;

// SHRSAMC 0;

ADCON2H = 0x0;

// SWCTRG disabled; SHRSAMP disabled; SUSPEND disabled; SWLCTRG disabled; SUSPCIE disabled; CNVCHSEL AN0; REFSEL disabled;

ADCON3L = 0x0;

// SHREN enabled; C3EN disabled; C2EN enabled; C1EN enabled; C0EN disabled; CLKDIV 1; CLKSEL FOSC/2;

ADCON3H = (0x86 & 0xFF00); //Disabling C0EN, C1EN, C2EN, C3EN and SHREN bits

// SAMC3EN disabled; SYNCTRG3 disabled; SAMC0EN disabled; SYNCTRG2 disabled; SAMC1EN disabled; SAMC2EN disabled; SYNCTRG1 disabled; SYNCTRG0 disabled;

ADCON4L = 0x0;

// C3CHS AN3; C0CHS AN0; C1CHS AN1; C2CHS AN2;

ADCON4H = 0x0;

// SIGN0 disabled; SIGN4 disabled; SIGN3 disabled; SIGN2 disabled; SIGN1 disabled; SIGN7 disabled; SIGN6 disabled; DIFF0 disabled; SIGN5 disabled; DIFF1 disabled; DIFF2 disabled; DIFF3 disabled; DIFF4 disabled; DIFF5 disabled; DIFF6 disabled; DIFF7 disabled;

ADMOD0L = 0x0;

// DIFF15 disabled; DIFF14 disabled; SIGN8 disabled; DIFF13 disabled; SIGN14 disabled; DIFF12 disabled; SIGN15 disabled; DIFF11 disabled; DIFF10 disabled; SIGN9 disabled; DIFF8 disabled; DIFF9 disabled; SIGN10 disabled; SIGN11 disabled; SIGN12 disabled; SIGN13 disabled;

ADMOD0H = 0x0;

// DIFF19 disabled; DIFF18 disabled; SIGN20 disabled; DIFF17 disabled; SIGN21 disabled; DIFF16 disabled; SIGN16 disabled; SIGN17 disabled; DIFF21 disabled; SIGN18 disabled; DIFF20 disabled; SIGN19 disabled;

ADMOD1L = 0x0;

// IE15 disabled; IE1 disabled; IE0 disabled; IE3 disabled; IE2 disabled; IE5 disabled; IE4 disabled; IE10 disabled; IE7 disabled; IE6 disabled; IE9 disabled; IE13 disabled; IE8 disabled; IE14 disabled; IE11 disabled; IE12 disabled;

ADIEL = 0x0;

// IE17 disabled; IE18 disabled; IE16 disabled; IE19 disabled; IE20 disabled; IE21 disabled;

ADIEH = 0x0;

// CMPEN10 disabled; CMPEN11 disabled; CMPEN6 disabled; CMPEN5 disabled; CMPEN4 disabled; CMPEN3 disabled; CMPEN2 disabled; CMPEN1 disabled; CMPEN0 disabled; CMPEN14 disabled; CMPEN9 disabled; CMPEN15 disabled; CMPEN8 disabled; CMPEN12 disabled; CMPEN7 disabled; CMPEN13 disabled;

ADCMP0ENL = 0x0;

// CMPEN10 disabled; CMPEN11 disabled; CMPEN6 disabled; CMPEN5 disabled; CMPEN4 disabled; CMPEN3 disabled; CMPEN2 disabled; CMPEN1 disabled; CMPEN0 disabled; CMPEN14 disabled; CMPEN9 disabled; CMPEN15

disabled; CMPEN8 disabled; CMPEN12 disabled; CMPEN7 disabled; CMPEN13 disabled;

ADCMP1ENL = 0x0;

// CMPEN21 disabled; CMPEN20 disabled; CMPEN18 disabled; CMPEN19 disabled; CMPEN16 disabled; CMPEN17 disabled;

ADCMP0ENH = 0x0;

// CMPEN21 disabled; CMPEN20 disabled; CMPEN18 disabled; CMPEN19 disabled; CMPEN16 disabled; CMPEN17 disabled;

ADCMP1ENH = 0x0;

// CMPLO 0;

ADCMP0LO = 0x0;

// CMPLO 0;

ADCMP1LO = 0x0;

// CMPHI 0;

ADCMP0HI = 0x0;

// CMPHI 0;

ADCMP1HI = 0x0;

// OVRSAM 4x; MODE Oversampling Mode; FLCHSEL AN0; IE disabled; FLEN disabled;

ADFL0CON = 0x400;

// OVRSAM 4x; MODE Oversampling Mode; FLCHSEL AN0; IE disabled; FLEN disabled;

```
    ADFL1CON = 0x400;

    // HIHI disabled; LOLO disabled; HILO disabled; BTWN disabled; LOHI disabled; CMPEN disabled; IE disabled;

    ADCMP0CON = 0x0;

    // HIHI disabled; LOLO disabled; HILO disabled; BTWN disabled; LOHI disabled; CMPEN disabled; IE disabled;

    ADCMP1CON = 0x0;

    // CHOFFSET 0;

    ADBASE = 0x0;

    // LVLEN9 disabled; LVLEN8 disabled; LVLEN11 disabled; LVLEN7 disabled; LVLEN10 disabled; LVLEN6 disabled; LVLEN13 disabled; LVLEN5 disabled; LVLEN12 disabled; LVLEN4 disabled; LVLEN15 disabled; LVLEN3 disabled; LVLEN14 disabled; LVLEN2 disabled; LVLEN1 disabled; LVLEN0 disabled;

    ADLVLTRGL = 0x0;

    // LVLEN20 disabled; LVLEN21 disabled; LVLEN17 disabled; LVLEN16 disabled; LVLEN19 disabled; LVLEN18 disabled;

    ADLVLTRGH = 0x0;

    // SAMC 0;

    ADCORE0L = 0x0;

    // SAMC 0;

    ADCORE1L = 0x0;

    // SAMC 0;

    ADCORE2L = 0x0;
```

// SAMC 0;

ADCORE3L = 0x0;

// RES 12-bit resolution; EISEL Early interrupt is generated 1 TADCORE clock prior to data being ready; ADCS 2;

ADCORE0H = 0x300;

// RES 12-bit resolution; EISEL Early interrupt is generated 1 TADCORE clock prior to data being ready; ADCS 2;

ADCORE1H = 0x300;

// RES 12-bit resolution; EISEL Early interrupt is generated 1 TADCORE clock prior to data being ready; ADCS 2;

ADCORE2H = 0x300;

// RES 12-bit resolution; EISEL Early interrupt is generated 1 TADCORE clock prior to data being ready; ADCS 2;

ADCORE3H = 0x300;

// EIEN9 disabled; EIEN7 disabled; EIEN8 disabled; EIEN5 disabled; EIEN6 disabled; EIEN3 disabled; EIEN4 disabled; EIEN1 disabled; EIEN2 disabled; EIEN13 disabled; EIEN0 disabled; EIEN12 disabled; EIEN11 disabled; EIEN10 disabled; EIEN15 disabled; EIEN14 disabled;

ADEIEL = 0x0;

// EIEN17 disabled; EIEN16 disabled; EIEN19 disabled; EIEN18 disabled; EIEN20 disabled; EIEN21 disabled;

ADEIEH = 0x0;

// C0CIE disabled; C1CIE disabled; C2CIE disabled; C3CIE disabled; SHRCIE disabled; WARMTIME 32768 Source Clock Periods;

```
ADCON5H = (0x1500 & 0xF0FF); //Disabling WARMTIME bit


// Setting WARMTIME bit
ADCON5Hbits.WARMTIME = 0xF;
// Enabling ADC Module
ADCON1Lbits.ADON = 0x1;
// Enabling Power for the Shared Core
ADC1_SharedCorePowerEnable();
// Calibrating the Shared Core
ADC1_SharedCoreCalibration();
// Enabling Power for Core2
ADC1_Core2PowerEnable();
// Calibrating Core2
ADC1_Core2Calibration();
// Enabling Power for Core1
ADC1_Core1PowerEnable();
// Calibrating Core1
ADC1_Core1Calibration();
//TRGSRC0 None; TRGSRC1 TMR2;
ADTRIG0L = 0xD00;
//TRGSRC3 None; TRGSRC2 TMR2;
```

```
    ADTRIG0H = 0xD;
    //TRGSRC4 TMR1; TRGSRC5 TMR2;
    ADTRIG1L = 0xD0C;
    //TRGSRC6 None; TRGSRC7 None;
    ADTRIG1H = 0x0;
    //TRGSRC8 TMR2; TRGSRC9 TMR2;
    ADTRIG2L = 0xD0D;
    //TRGSRC11 TMR2; TRGSRC10 TMR2;
    ADTRIG2H = 0xD0D;
    //TRGSRC13 TMR2; TRGSRC12 TMR2;
    ADTRIG3L = 0xD0D;
    //TRGSRC15 TMR2; TRGSRC14 None;
    ADTRIG3H = 0xD00;
    //TRGSRC17 None; TRGSRC16 None;
    ADTRIG4L = 0x0;
    //TRGSRC19 TMR2; TRGSRC18 None;
    ADTRIG4H = 0xD00;
    //TRGSRC20 None; TRGSRC21 None;
    ADTRIG5L = 0x0;
}
//------------------------------------------------------------------------
```

```
void ADC1_Core0PowerEnable ( )
{
   ADCON5Lbits.C0PWR = 1;
   while(ADCON5Lbits.C0RDY == 0);
   ADCON3Hbits.C0EN = 1;
}
//------------------------------------------------------------------------
void ADC1_Core0Calibration ( )
{
   ADCAL0Lbits.CAL0EN = 1;
   ADCAL0Lbits.CAL0DIFF = 0;
   ADCAL0Lbits.CAL0RUN = 1;
   while(ADCAL0Lbits.CAL0RDY == 0);
   ADCAL0Lbits.CAL0EN = 0;
}
//------------------------------------------------------------------------
void ADC1_Core1PowerEnable ( )
{
   ADCON5Lbits.C1PWR = 1;
   while(ADCON5Lbits.C1RDY == 0);
   ADCON3Hbits.C1EN = 1;
```

```
}
//---------------------------------------------------------------------------
void ADC1_Core1Calibration ( )
{
   ADCAL0Lbits.CAL1EN = 1;
   ADCAL0Lbits.CAL1DIFF = 0;
   ADCAL0Lbits.CAL1RUN = 1;
   while(ADCAL0Lbits.CAL1RDY == 0);
   ADCAL0Lbits.CAL1EN = 0;
}
//---------------------------------------------------------------------------
void ADC1_Core2PowerEnable ( )
{
   ADCON5Lbits.C2PWR = 1;
   while(ADCON5Lbits.C2RDY == 0);
   ADCON3Hbits.C2EN = 1;
}
//---------------------------------------------------------------------------
void ADC1_Core2Calibration ( )
{
   ADCAL0Hbits.CAL2EN = 1;
```

```
    ADCAL0Hbits.CAL2DIFF = 0;
    ADCAL0Hbits.CAL2RUN = 1;
    while(ADCAL0Hbits.CAL2RDY == 0);
    ADCAL0Hbits.CAL2EN = 0;
}
//------------------------------------------------------------------------
void ADC1_Core3PowerEnable ( )
{
    ADCON5Lbits.C3PWR = 1;
    while(ADCON5Lbits.C3RDY == 0);
    ADCON3Hbits.C3EN = 1;
}
//------------------------------------------------------------------------
void ADC1_Core3Calibration ( )
{
    ADCAL0Hbits.CAL3EN = 1;
    ADCAL0Hbits.CAL3DIFF = 0;
    ADCAL0Hbits.CAL3RUN = 1;
    while(ADCAL0Hbits.CAL3RDY == 0);
    ADCAL0Hbits.CAL3EN = 0;
}
```

```
//---------------------------------------------------------------------------

void ADC1_SharedCorePowerEnable ( )

{

   ADCON5Lbits.SHRPWR = 1;

   while(ADCON5Lbits.SHRRDY == 0);

   ADCON3Hbits.SHREN = 1;

}

//---------------------------------------------------------------------------

void ADC1_SharedCoreCalibration ( )

{

   ADCAL1Hbits.CSHREN = 1;

   ADCAL1Hbits.CSHRDIFF = 0;

   ADCAL1Hbits.CSHRRUN = 1;

   while(ADCAL1Hbits.CSHRRDY == 0);

   ADCAL1Hbits.CSHREN = 0;

}

//---------------------------------------------------------------------------

void ADC1_Tasks ( void )

{

   // clear the ADC1 interrupt flag

   IFS0bits.ADCIF = 0;
```

```
}

//---------------------------------------------------------------------------

void cmp1_init(void)

{

   // Disable the CMP module before the initialization

   CMP1_Enable(false);

   // DACOE enabled; HYSPOL Rising Edge; ALTINP Comparator; FLTREN enabled; FCLKSEL System Clock; CMPSTAT disabled; EXTREF AVDD; CMPPOL Not Inverted; CMPSIDL disabled; CMPON enabled; HYSSEL No hysteresis; INSEL CMP1A; RANGE AVDD is the maximum DAC output voltage;

   CMP1CON = 0x8501;

   // CMREF 2047;

   CMP1DAC = 0x7FF;

   CMP1_Enable(true);

}

//---------------------------------------------------------------------------

void CMP1_Enable(bool enable)

{

   (CMP1CONbits.CMPON) = enable;

}

//---------------------------------------------------------------------------

void CMP1_SetInputSource(CMP1_INPUT inpSrc)
```

```
{
    (CMP1CONbits.INSEL) = inpSrc;
}
//---------------------------------------------------------------------------
bool CMP1_DACOutputStatusGet(void)
{
    return (CMP1CONbits.DACOE);
}
//---------------------------------------------------------------------------
bool CMP1_StatusGet(void)
{
    return (CMP1CONbits.CMPSTAT);
}
//---------------------------------------------------------------------------
void CMP1_Tasks(void)
{
    // clear the CMP1 interrupt flag
    IFS1bits.AC1IF = 0;
}
//---------------------------------------------------------------------------
void interrupt_init (void)
```

```
{
    //    TI: Timer 1
    //    Priority: 1
    IPC0bits.T1IP = 1;
}
//---------------------------------------------------------------------------
void pin_manager_init(void)
{

/**************************************************************************
*********
     * Setting the Output Latch SFR(s)

**************************************************************************
*******/
    LATA = 0x0000;
    LATB = 0x0000;
    LATC = 0x0000;
    LATD = 0x0000;

/**************************************************************************
*********
     * Setting the GPIO Direction SFR(s)
```

```
*************************************************************************
*******/
    TRISA = 0x0007;
    TRISB = 0xFAF7;
    TRISC = 0xF7BE;
    TRISD = 0x75FC;
/*************************************************************************
*********
     * Setting the Weak Pull Up and Weak Pull Down SFR(s)

*************************************************************************
*******/
    CNPDA = 0x0000;
    CNPDB = 0x0000;
    CNPDC = 0x0000;
    CNPDD = 0x0000;
    CNPUA = 0x0000;
    CNPUB = 0x0000;
    CNPUC = 0x0000;
    CNPUD = 0x0000;
```

```
/*********************************************************************
*********
    * Setting the Open Drain SFR(s)

*********************************************************************
*******/
   ODCA = 0x0000;
   ODCB = 0x0000;
   ODCC = 0x0000;
   ODCD = 0x0000;

/*********************************************************************
*********
    * Setting the Analog/Digital Configuration SFR(s)

*********************************************************************
*******/
   ANSELA = 0x0007;
   ANSELB = 0x022D;
   ANSELC = 0x1676;
   ANSELD = 0x21A4;
}
//------------------------------------------------------------------------
```

```
void pwm_init (void)

{

   // PCLKDIV 1;

   PTCON2 = 0x0;

   // PTPER 1535;

   PTPER = 0x5FF;

   // SEVTCMP 0;

   SEVTCMP = 0x0;

   // MDC 255;

   MDC = 0xFF;

   // CHOPCLK 0; CHPCLKEN disabled;

   CHOP = 0x0;

   // PWMKEY 0;

   PWMKEY = 0x0;

   // MDCS Primary; FLTIEN disabled; CAM Edge Aligned; DTC Positive dead time; TRGIEN disabled; XPRES disabled; ITB Master; IUE disabled; CLIEN disabled; MTBS disabled;

   PWMCON1 = 0x0;

   // MDCS Primary; FLTIEN disabled; CAM Edge Aligned; DTC Positive dead time; TRGIEN disabled; XPRES disabled; ITB Master; IUE disabled; CLIEN disabled; MTBS disabled;

   PWMCON2 = 0x0;
```

```
    // MDCS Primary; FLTIEN disabled; CAM Edge Aligned; DTC Positive dead time; TRGIEN disabled; XPRES disabled; ITB Master; IUE disabled; CLIEN disabled; MTBS disabled;

    PWMCON3 = 0x0;

    // MDCS Primary; FLTIEN disabled; CAM Edge Aligned; DTC Positive dead time; TRGIEN disabled; XPRES disabled; ITB Master; IUE disabled; CLIEN disabled; MTBS disabled;

    PWMCON4 = 0x0;

    // MDCS Primary; FLTIEN disabled; CAM Edge Aligned; DTC Positive dead time; TRGIEN disabled; XPRES disabled; ITB Master; IUE disabled; CLIEN disabled; MTBS disabled;

    PWMCON5 = 0x0;

    // MDCS Primary; FLTIEN disabled; CAM Edge Aligned; DTC Positive dead time; TRGIEN disabled; XPRES disabled; ITB Master; IUE disabled; CLIEN disabled; MTBS disabled;

    PWMCON6 = 0x0;

    // MDCS Primary; FLTIEN disabled; CAM Edge Aligned; DTC Positive dead time; TRGIEN disabled; XPRES disabled; ITB Master; IUE disabled; CLIEN disabled; MTBS disabled;

    PWMCON7 = 0x0;

    // MDCS Primary; FLTIEN disabled; CAM Edge Aligned; DTC Positive dead time; TRGIEN disabled; XPRES disabled; ITB Master; IUE disabled; CLIEN disabled; MTBS disabled;

    PWMCON8 = 0x0;
```

```
    //FLTDAT PWM1L Low, PWM1H Low; SWAP disabled; OVRENH disabled; PENL enabled; PMOD Push-Pull Output Mode; OVRENL disabled; OSYNC disabled; POLL disabled; PENH enabled; CLDAT PWM1L Low, PWM1H Low; OVRDAT PWM1L High, PWM1H High; POLH disabled;

    __builtin_write_PWMSFR(&IOCON1, 0xC8C0, &PWMKEY);

    //FLTDAT PWM2L Low, PWM2H Low; SWAP disabled; OVRENH disabled; PENL disabled; PMOD Complementary Output Mode; OVRENL disabled; OSYNC disabled; POLL disabled; PENH disabled; CLDAT PWM2L Low, PWM2H Low; OVRDAT PWM2L Low, PWM2H Low; POLH disabled;

    __builtin_write_PWMSFR(&IOCON2, 0x0, &PWMKEY);

    //FLTDAT PWM3L Low, PWM3H Low; SWAP disabled; OVRENH disabled; PENL disabled; PMOD Complementary Output Mode; OVRENL disabled; OSYNC disabled; POLL disabled; PENH disabled; CLDAT PWM3L Low, PWM3H Low; OVRDAT PWM3L Low, PWM3H Low; POLH disabled;

    __builtin_write_PWMSFR(&IOCON3, 0x0, &PWMKEY);

    //FLTDAT PWM4L Low, PWM4H Low; SWAP disabled; OVRENH disabled; PENL disabled; PMOD Complementary Output Mode; OVRENL disabled; OSYNC disabled; POLL disabled; PENH disabled; CLDAT PWM4L Low, PWM4H Low; OVRDAT PWM4L Low, PWM4H Low; POLH disabled;

    __builtin_write_PWMSFR(&IOCON4, 0x0, &PWMKEY);

    //FLTDAT PWM5L Low, PWM5H Low; SWAP disabled; OVRENH disabled; PENL disabled; PMOD Complementary Output Mode; OVRENL disabled; OSYNC disabled; POLL disabled; PENH disabled; CLDAT PWM5L Low, PWM5H Low; OVRDAT PWM5L Low, PWM5H Low; POLH disabled;

    __builtin_write_PWMSFR(&IOCON5, 0x0, &PWMKEY);
```

```
    //FLTDAT PWM6L Low, PWM6H Low; SWAP disabled; OVRENH disabled; PENL disabled; PMOD Complementary Output Mode; OVRENL disabled; OSYNC disabled; POLL disabled; PENH disabled; CLDAT PWM6L Low, PWM6H Low; OVRDAT PWM6L Low, PWM6H Low; POLH disabled;

    __builtin_write_PWMSFR(&IOCON6, 0x0, &PWMKEY);

    //FLTDAT PWM7L Low, PWM7H Low; SWAP disabled; OVRENH disabled; PENL disabled; PMOD Complementary Output Mode; OVRENL disabled; OSYNC disabled; POLL disabled; PENH disabled; CLDAT PWM7L Low, PWM7H Low; OVRDAT PWM7L Low, PWM7H Low; POLH disabled;

    __builtin_write_PWMSFR(&IOCON7, 0x0, &PWMKEY);

    //FLTDAT PWM8L Low, PWM8H Low; SWAP disabled; OVRENH disabled; PENL disabled; PMOD Complementary Output Mode; OVRENL disabled; OSYNC disabled; POLL disabled; PENH disabled; CLDAT PWM8L Low, PWM8H Low; OVRDAT PWM8L Low, PWM8H Low; POLH disabled;

    __builtin_write_PWMSFR(&IOCON8, 0x0, &PWMKEY);

    //FLTPOL disabled; CLPOL disabled; CLSRC CMP1; CLMOD enabled; FLTMOD Fault input is disabled; IFLTMOD disabled; FLTSRC FLT1;

    __builtin_write_PWMSFR(&FCLCON1, 0x350B, &PWMKEY);

    //FLTPOL disabled; CLPOL disabled; CLSRC FLT1; CLMOD disabled; FLTMOD Fault input is disabled; IFLTMOD disabled; FLTSRC FLT1;

    __builtin_write_PWMSFR(&FCLCON2, 0x40B, &PWMKEY);

    //FLTPOL disabled; CLPOL disabled; CLSRC FLT1; CLMOD disabled; FLTMOD Fault input is disabled; IFLTMOD disabled; FLTSRC FLT1;

    __builtin_write_PWMSFR(&FCLCON3, 0x40B, &PWMKEY);
```

```
    //FLTPOL disabled; CLPOL disabled; CLSRC FLT1; CLMOD disabled; FLTMOD Fault input is disabled; IFLTMOD disabled; FLTSRC FLT1;

    __builtin_write_PWMSFR(&FCLCON4, 0x40B, &PWMKEY);

    //FLTPOL disabled; CLPOL disabled; CLSRC FLT1; CLMOD disabled; FLTMOD Fault input is disabled; IFLTMOD disabled; FLTSRC FLT1;

    __builtin_write_PWMSFR(&FCLCON5, 0x40B, &PWMKEY);

    //FLTPOL disabled; CLPOL disabled; CLSRC FLT1; CLMOD disabled; FLTMOD Fault input is disabled; IFLTMOD disabled; FLTSRC FLT1;

    __builtin_write_PWMSFR(&FCLCON6, 0x40B, &PWMKEY);

    //FLTPOL disabled; CLPOL disabled; CLSRC FLT1; CLMOD disabled; FLTMOD Fault input is disabled; IFLTMOD disabled; FLTSRC FLT1;

    __builtin_write_PWMSFR(&FCLCON7, 0x40B, &PWMKEY);

    //FLTPOL disabled; CLPOL disabled; CLSRC FLT1; CLMOD disabled; FLTMOD Fault input is disabled; IFLTMOD disabled; FLTSRC FLT1;

    __builtin_write_PWMSFR(&FCLCON8, 0x40B, &PWMKEY);

    // PDC1 1200;

    PDC1 = 0x4B0;

    // PDC2 0;

    PDC2 = 0x0;

    // PDC3 0;

    PDC3 = 0x0;

    // PDC4 0;
```

```
PDC4 = 0x0;

// PDC5 0;

PDC5 = 0x0;

// PDC6 0;

PDC6 = 0x0;

// PDC7 0;

PDC7 = 0x0;

// PDC8 0;

PDC8 = 0x0;

// PHASE1 0;

PHASE1 = 0x0;

// PHASE2 0;

PHASE2 = 0x0;

// PHASE3 0;

PHASE3 = 0x0;

// PHASE4 0;

PHASE4 = 0x0;

// PHASE5 0;

PHASE5 = 0x0;

// PHASE6 0;

PHASE6 = 0x0;
```

```
// PHASE7 0;

PHASE7 = 0x0;

// PHASE8 0;

PHASE8 = 0x0;

// DTR1 296;

DTR1 = 0x128;

// DTR2 0;

DTR2 = 0x0;

// DTR3 0;

DTR3 = 0x0;

// DTR4 0;

DTR4 = 0x0;

// DTR5 0;

DTR5 = 0x0;

// DTR6 0;

DTR6 = 0x0;

// DTR7 0;

DTR7 = 0x0;

// DTR8 0;

DTR8 = 0x0;

// ALTDTR1 0;
```

```
ALTDTR1 = 0x0;

// ALTDTR2 0;

ALTDTR2 = 0x0;

// ALTDTR3 0;

ALTDTR3 = 0x0;

// ALTDTR4 0;

ALTDTR4 = 0x0;

// ALTDTR5 0;

ALTDTR5 = 0x0;

// ALTDTR6 0;

ALTDTR6 = 0x0;

// ALTDTR7 0;

ALTDTR7 = 0x0;

// ALTDTR8 0;

ALTDTR8 = 0x0;

// SDC1 0;

SDC1 = 0x0;

// SDC2 0;

SDC2 = 0x0;

// SDC3 0;

SDC3 = 0x0;
```

```
// SDC4 0;

SDC4 = 0x0;

// SDC5 0;

SDC5 = 0x0;

// SDC6 0;

SDC6 = 0x0;

// SDC7 0;

SDC7 = 0x0;

// SDC8 0;

SDC8 = 0x0;

// SPHASE1 0;

SPHASE1 = 0x0;

// SPHASE2 0;

SPHASE2 = 0x0;

// SPHASE3 0;

SPHASE3 = 0x0;

// SPHASE4 0;

SPHASE4 = 0x0;

// SPHASE5 0;

SPHASE5 = 0x0;

// SPHASE6 0;
```

```
SPHASE6 = 0x0;

// SPHASE7 0;

SPHASE7 = 0x0;

// SPHASE8 0;

SPHASE8 = 0x0;

// TRGCMP 0;

TRIG1 = 0x0;

// TRGCMP 0;

TRIG2 = 0x0;

// TRGCMP 0;

TRIG3 = 0x0;

// TRGCMP 0;

TRIG4 = 0x0;

// TRGCMP 0;

TRIG5 = 0x0;

// TRGCMP 0;

TRIG6 = 0x0;

// TRGCMP 0;

TRIG7 = 0x0;

// TRGCMP 0;

TRIG8 = 0x0;
```

```
// TRGDIV 1; TRGSTRT 0; DTM disabled;

TRGCON1 = 0x0;

// TRGDIV 1; TRGSTRT 0; DTM disabled;

TRGCON2 = 0x0;

// TRGDIV 1; TRGSTRT 0; DTM disabled;

TRGCON3 = 0x0;

// TRGDIV 1; TRGSTRT 0; DTM disabled;

TRGCON4 = 0x0;

// TRGDIV 1; TRGSTRT 0; DTM disabled;

TRGCON5 = 0x0;

// TRGDIV 1; TRGSTRT 0; DTM disabled;

TRGCON6 = 0x0;

// TRGDIV 1; TRGSTRT 0; DTM disabled;

TRGCON7 = 0x0;

// TRGDIV 1; TRGSTRT 0; DTM disabled;

TRGCON8 = 0x0;

// STRGCMP 0;

STRIG1 = 0x0;

// STRGCMP 0;

STRIG2 = 0x0;

// STRGCMP 0;
```

```
STRIG3 = 0x0;

// STRGCMP 0;

STRIG4 = 0x0;

// STRGCMP 0;

STRIG5 = 0x0;

// STRGCMP 0;

STRIG6 = 0x0;

// STRGCMP 0;

STRIG7 = 0x0;

// STRGCMP 0;

STRIG8 = 0x0;

// PWMCAP 0;

PWMCAP1 = 0x0;

// PWMCAP 0;

PWMCAP2 = 0x0;

// PWMCAP 0;

PWMCAP3 = 0x0;

// PWMCAP 0;

PWMCAP4 = 0x0;

// PWMCAP 0;

PWMCAP5 = 0x0;
```

```
// PWMCAP 0;

PWMCAP6 = 0x0;

// PWMCAP 0;

PWMCAP7 = 0x0;

// PWMCAP 0;

PWMCAP8 = 0x0;

// BPLL disabled; BPHH disabled; BPLH disabled; BCH disabled; FLTLEBEN disabled; PLR disabled; CLLEBEN disabled; BCL disabled; PLF disabled; PHR disabled; BPHL disabled; PHF disabled;

LEBCON1 = 0x0;

// BPLL disabled; BPHH disabled; BPLH disabled; BCH disabled; FLTLEBEN disabled; PLR disabled; CLLEBEN disabled; BCL disabled; PLF disabled; PHR disabled; BPHL disabled; PHF disabled;

LEBCON2 = 0x0;

// BPLL disabled; BPHH disabled; BPLH disabled; BCH disabled; FLTLEBEN disabled; PLR disabled; CLLEBEN disabled; BCL disabled; PLF disabled; PHR disabled; BPHL disabled; PHF disabled;

LEBCON3 = 0x0;

// BPLL disabled; BPHH disabled; BPLH disabled; BCH disabled; FLTLEBEN disabled; PLR disabled; CLLEBEN disabled; BCL disabled; PLF disabled; PHR disabled; BPHL disabled; PHF disabled;

LEBCON4 = 0x0;
```

```
    // BPLL disabled; BPHH disabled; BPLH disabled; BCH disabled; FLTLEBEN disabled; PLR disabled; CLLEBEN disabled; BCL disabled; PLF disabled; PHR disabled; BPHL disabled; PHF disabled;

    LEBCON5 = 0x0;

    // BPLL disabled; BPHH disabled; BPLH disabled; BCH disabled; FLTLEBEN disabled; PLR disabled; CLLEBEN disabled; BCL disabled; PLF disabled; PHR disabled; BPHL disabled; PHF disabled;

    LEBCON6 = 0x0;

    // BPLL disabled; BPHH disabled; BPLH disabled; BCH disabled; FLTLEBEN disabled; PLR disabled; CLLEBEN disabled; BCL disabled; PLF disabled; PHR disabled; BPHL disabled; PHF disabled;

    LEBCON7 = 0x0;

    // BPLL disabled; BPHH disabled; BPLH disabled; BCH disabled; FLTLEBEN disabled; PLR disabled; CLLEBEN disabled; BCL disabled; PLF disabled; PHR disabled; BPHL disabled; PHF disabled;

    LEBCON8 = 0x0;

    // LEB 0;

    LEBDLY1 = 0x0;

    // LEB 0;

    LEBDLY2 = 0x0;

    // LEB 0;

    LEBDLY3 = 0x0;

    // LEB 0;

    LEBDLY4 = 0x0;
```

```
// LEB 0;

LEBDLY5 = 0x0;

// LEB 0;

LEBDLY6 = 0x0;

// LEB 0;

LEBDLY7 = 0x0;

// LEB 0;

LEBDLY8 = 0x0;

// CHOPLEN disabled; HRDDIS disabled; CHOPHEN disabled; BLANKSEL No state blanking; CHOPSEL No state blanking; HRPDIS disabled;

AUXCON1 = 0x0;

// CHOPLEN disabled; HRDDIS disabled; CHOPHEN disabled; BLANKSEL No state blanking; CHOPSEL No state blanking; HRPDIS disabled;

AUXCON2 = 0x0;

// CHOPLEN disabled; HRDDIS disabled; CHOPHEN disabled; BLANKSEL No state blanking; CHOPSEL No state blanking; HRPDIS disabled;

AUXCON3 = 0x0;

// CHOPLEN disabled; HRDDIS disabled; CHOPHEN disabled; BLANKSEL No state blanking; CHOPSEL No state blanking; HRPDIS disabled;

AUXCON4 = 0x0;

// CHOPLEN disabled; HRDDIS disabled; CHOPHEN disabled; BLANKSEL No state blanking; CHOPSEL No state blanking; HRPDIS disabled;
```

```
    AUXCON5 = 0x0;

    // CHOPLEN disabled; HRDDIS disabled; CHOPHEN disabled; BLANKSEL No state blanking; CHOPSEL No state blanking; HRPDIS disabled;

    AUXCON6 = 0x0;

    // CHOPLEN disabled; HRDDIS disabled; CHOPHEN disabled; BLANKSEL No state blanking; CHOPSEL No state blanking; HRPDIS disabled;

    AUXCON7 = 0x0;

    // CHOPLEN disabled; HRDDIS disabled; CHOPHEN disabled; BLANKSEL No state blanking; CHOPSEL No state blanking; HRPDIS disabled;

    AUXCON8 = 0x0;

    // SYNCOEN disabled; SEIEN disabled; SESTAT disabled; SEVTPS 1; SYNCSRC SYNCI1; SYNCEN disabled; PTSIDL disabled; PTEN enabled; EIPU disabled; SYNCPOL disabled;

    PTCON = 0x8000;

}

//------------------------------------------------------------------------------

void spi1_init (void)

{

    // AUDEN disabled; FRMEN disabled; AUDMOD I2S; FRMSYPW One clock wide; AUDMONO stereo; FRMCNT 0; MSSEN disabled; FRMPOL disabled; IGNROV disabled; SPISGNEXT not sign-extended; FRMSYNC disabled; URDTEN disabled; IGNTUR disabled;

    SPI1CON1H = 0x0;

    // WLENGTH 0;
```

```
    SPI1CON2L = 0x0;

    // SPIROV disabled; FRMERR disabled;

    SPI1STATL = 0x0;

    // SPI1BRGL 15;

    SPI1BRGL = 0xF;

    // SPITBFEN disabled; SPITUREN disabled; FRMERREN disabled; SRMTEN disabled; SPIRBEN disabled; BUSYEN disabled; SPITBEN disabled; SPIROVEN disabled; SPIRBFEN disabled;

    SPI1IMSKL = 0x0;

    // RXMSK 0; TXWIEN disabled; TXMSK 0; RXWIEN disabled;

    SPI1IMSKH = 0x0;

    // SPI1URDTL 0;

    SPI1URDTL = 0x0;

    // SPI1URDTH 0;

    SPI1URDTH = 0x0;

    // SPIEN enabled; DISSDO disabled; MCLKEN FOSC/2; CKP Idle:Low, Active:High; SSEN disabled; MSTEN Master; MODE16 disabled; SMP Middle; DISSCK disabled; SPIFE Frame Sync pulse precedes; CKE Idle to Active; MODE32 disabled; SPISIDL disabled; ENHBUF disabled; DISSDI disabled;

    SPI1CON1L = 0x8020;

}

//---------------------------------------------------------------------------
```

```
void TMR1_Initialize (void)
{
    //TMR1 0;
    TMR1 = 0x0;
    //Period = 0.0009984 s; Frequency = 60000000 Hz; PR1 235;
    PR1 = 0xEB;
    //TCKPS 1:256; TON enabled; TSIDL disabled; TCS FOSC/2; TSYNC disabled;
TGATE disabled;
    T1CON = 0x8030;
    IFS0bits.T1IF = false;
    IEC0bits.T1IE = true;
    tmr1_obj.timerElapsed = false;
}
//---------------------------------------------------------------------------
void __attribute__ ( ( interrupt, no_auto_psv ) ) _T1Interrupt (  )
{
    /* Check if the Timer Interrupt/Status is set */
    //***User Area Begin
    // ticker function call;
    // ticker is 1 -> Callback function gets called everytime this ISR executes
    TMR1_CallBack();
```

```
    //***User Area End

    tmr1_obj.count++;

    tmr1_obj.timerElapsed = true;

    IFS0bits.T1IF = false;

}

//---------------------------------------------------------------------------

void TMR1_Period16BitSet( uint16_t value )

{

    /* Update the counter values */

    PR1 = value;

    /* Reset the status information */

    tmr1_obj.timerElapsed = false;

}

//---------------------------------------------------------------------------

uint16_t TMR1_Period16BitGet( void )

{

    return( PR1 );

}

//---------------------------------------------------------------------------

void TMR1_Counter16BitSet ( uint16_t value )
```

```
{
    /* Update the counter values */
    TMR1 = value;
    /* Reset the status information */
    tmr1_obj.timerElapsed = false;
}
//---------------------------------------------------------------------------
uint16_t TMR1_Counter16BitGet( void )
{
    return( TMR1 );
}
//---------------------------------------------------------------------------
void __attribute__ ((weak)) TMR1_CallBack(void)
{
    // Add your custom callback code here
}
//---------------------------------------------------------------------------
void TMR1_Start( void )
{
    /* Reset the status information */
    tmr1_obj.timerElapsed = false;
```

```
    /*Enable the interrupt*/
    IEC0bits.T1IE = true;
    /* Start the Timer */
    T1CONbits.TON = 1;
}
//------------------------------------------------------------------------------
void TMR1_Stop( void )
{
    /* Stop the Timer */
    T1CONbits.TON = false;
    /*Disable the interrupt*/
    IEC0bits.T1IE = false;
}
//------------------------------------------------------------------------------
bool TMR1_GetElapsedThenClear(void)
{
    bool status;
    status = tmr1_obj.timerElapsed;
    if(status == true)
    {
        tmr1_obj.timerElapsed = false;
```

```
    }

    return status;

}

//---------------------------------------------------------------------------

int TMR1_SoftwareCounterGet(void)

{

    return tmr1_obj.count;

}

//---------------------------------------------------------------------------

void TMR1_SoftwareCounterClear(void)

{

    tmr1_obj.count = 0;

}
```

NOTE: Some part of this code is generated using the MPLAB X IDE tool MCC

CURRICULUM VITAE

PERSONAL INFORMATION



Name Surname : Ertuğrul KOCAAĞA
Date of Birth : 12.11.1991
Phone : +90 554 900 6034
E-mail : ertugrul.kocaaga@gmail.com

EDUCATION

High School : Ankara Atatürk High School
Bachelor : Hacettepe University
Master's Degree : Ankara Yıldırım Beyazıt University

WORK EXPERIENCE

R&D Engineer : Linkas Teknoloji, 2017-…
R&D Engineer : RKC Kontrol, 2015-2017

TOPICS OF INTEREST

- Power Electronics
- Embedded Systems Firmware Design
- Embedded Systems Hardware Design
- Simulation