IMPLEMENTATION OF SPACE VECTOR MODULATION FOR THREE PHASE INVERTER UTILIZING FIELD-PROGRAMMING GATE ARRAY

HALIMAHANI BINTI ROSLI

BACHELOR OF ELECTRICAL ENGINEERING
(GENERAL)
UNIVERSITY TECHNICAL MALAYSIA MELAKA

“ I hereby declare that I have read through this report entitle “*Implementation Of Space Vector Modulation For Three Phase Inverter Utilizing Field-Programming Gate Array*” and found that it complies the partial fulfillment for awarding the degree of Bachelor of Electrical Engineering”

Signature : ....

Supervisor’s Name : DR. Auzani Bin Jidin

Date : ....

IMPLEMENTATION OF SPACE VECTOR MODULATION FOR THREE PHASE INVERTER UTILIZING FIELD-PROGRAMMING GATE ARRAY

HALIMAHANI BINTI ROSLI

A report submitted in partial fulfilment of the requirements for the degree of Bachelor of Electrical Engineering

Faculty of Electrical Engineering

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2018

I declare that this report entitle "*Implementation Of Space Vector Modulation For Three Phase Inverter Utilizing Field-Programming Gate Array*" is the result of my own research except as cited in the references. The report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature : ....

Name : HALIMAHANI BINTI ROSLI

Date : ....

Special dedication to my parent Murdiati binti Darwis and Rosli bin Saat which give me a lot of love and support to complete my final year project 1. As well as my sibling, Hamidahsyahri, Yahamzah and Yabahalif which give me strength and happiness.

ACKNOWLEDGEMENT

Gratitude to Allah SWT, with His willing that give me this strength and motivation to complete my final year project at University Technical Malaysia Melaka. I would like to take this opportunity to express my gratitude to my supervisor, Dr. Auzani bin Jidin for his constant guidance, advised and encouragement throughout the completion of this report.

I also would like to extend my appreciation and thanks to my friends who have helped share their knowledge and time with me in the completion of this project. As well as my panel supervisor, Dr. Kasrul for his time and effort to evaluate my report.

In addition, to my family for their moral support through the entire progress of this report. Last but not least, I would like to thank each and every one who has helped me directly or indirectly in completing and preparing for this report until realization.

ABSTRACT

Industries that used ac drives are growing rapidly to become more advanced because of high performances drive technology. In industry different there are different type of motor and variable speed. As for that, the three-phase inverters will give a variable voltage and variable frequency supply to the machines. As for that, using Space Vector Modulation (SVM) with Field-Programmable Gate Array (FPGA) can increase the performance of the three-phase inverter. The problem occurs within the SPWM technique and the existing SVM works. The SPWM techniques are difficult to control three-phase quantities such as amplitudes, frequencies and phase angle of three-phase voltage and depending to the torque and flux demands in vector control. This method also has a constrain for output voltage such as it cannot fully utilize the DC link voltage. Next, the problem with the existing SVM works because it totally implement the complex calculation of SVM algorithm and vector control in DSP which can increase the computational burden and hence reduce AC drive performance as the sampling time increase. The main objectives to be achieved in this task is to design and implement a space vector modulator utilizing Field-Programmable Gate Array (FPGA) which produce appropriate switching states, according to the inputs of d- and q- axis of space vector for measuring the output voltage and Fast Fourier Transform (FFT) Analysis. This modulator is suitable to be employed for any type of AC motor controls. Second objective is to verify the effectiveness of the design of SVM modulator through simulation and experiment. The scope of the project is to study various PWM and SVM techniques, especially for AC motor drives. Next, to develop simulation models of SVM using MATLAB or SIMULINK. Besides, implement SVM modulator using FPGA. As well as, performing the simulation and the experiment results to verify the effectiveness of SVM. The simulating and experimental designing the Space Vector Pulse Width Modulation (SVPWM) technique for three phase Voltage Source Inverter (VSI) using MATLAB/Simulink software for simulation as well as to develop the experimental result using FPGA. SVM can operated in high voltage and high frequency. The results that produce by SVM is high quality because it can managed to read in nanosecond.

ABSTRAK

Industri yang menggunakan pemacu ac berkembang pesat untuk menjadi lebih maju kerana prestasi yang tinggi memacu teknologi. Oleh itu, inverter tiga fasa akan memberikan pemboleh ubah voltan dan pemboleh ubah frekuensi kepada mesin. Untuk itu, menggunakan Modulasi Vektor Ruang (SVM) dengan Field-Programmable Gate Array (FPGA) boleh meningkatkan prestasi inverter tiga fasa. Masalah yang berlaku dalam teknik SPWM dan kerja-kerja SVM sedia ada. Teknik SPWM sukar untuk mengawal kuantiti tiga fasa seperti amplitud, frekuensi dan sudut fasa voltan tiga fasa dan bergantung kepada tork dan permintaan fluks dalam kawalan vektor. Kaedah ini juga mempunyai kekangan untuk voltan keluaran seperti tidak dapat menggunakan sepenuhnya voltan pautan DC. Seterusnya, masalah dengan SVM yang sedia ada berfungsi kerana ia benar-benar melaksanakan pengiraan kompleks algoritma dan kawalan vektor SVM di DSP yang boleh meningkatkan beban pengiraan dan dengan itu mengurangkan prestasi pemacu AC apabila peningkatan masa pensampelan. Objektif utama yang akan dicapai dalam tugas ini adalah untuk mereka bentuk dan melaksanakan modulator vektor ruang yang menggunakan litar Field-Programmable Gate Array (FPGA) yang menghasilkan litar bersesuaian sesuai dengan input d- dan q - paksi vektor ruang untuk mengukur voltan keluaran dan Analisis Fast Fourier Transform (FFT). Modulator ini sesuai digunakan untuk sebarang jenis kawalan motor AC. Objektif kedua adalah untuk mengesahkan keberkesanan reka bentuk modulator SVM melalui simulasi dan eksperimen. Skop projek ini adalah untuk mengkaji pelbagai teknik PWM dan SVM, terutamanya untuk pemacu motor AC. Seterusnya, untuk membangunkan model simulasi SVM menggunakan MATLAB atau SIMULINK. Selain itu, melaksanakan modulator SVM menggunakan litar FPGA. Serta, melaksanakan simulasi dan hasil percubaan untuk mengesahkan keberkesanan SVM. Simulasi dan eksperimen merancang teknik Modulasi Lebar Pulse Vector Space (SVPWM) untuk tiga fasa Voltage Source Inverter (VSI) menggunakan perisian MATLAB / Simulink untuk simulasi serta untuk membangunkan hasil eksperimen menggunakan FPGA. SVM boleh dikendalikan dalam voltan tinggi dan frekuensi tinggi. Hasil yang dihasilkan oleh SVM adalah berkualiti tinggi kerana ia dapat dibaca dalam nanosecond.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

PWM	-	Pulse width modulation
SPWM	-	Sinusoidal pulse width modulation
SVPWM	-	Space vector pulse width modulation
SVM	-	Space vector modulation
AC	-	Alternating current
DC	-	Direct current
DTC	-	Direct torque control
FPGA	-	Field-programmable gate array
VHDL	-	VHSIC hardware description language
VHSIC	-	Very high speed integrated circuit
FFT	-	Fast fourier transform
VSI	-	Voltage source inverter
CSI	-	Current source inverter
ASD	-	Adjustable speed drives
VSD	-	Variable speed drives
UPS	-	Uninterruptible power supplies
HVDC	-	High voltage direct current
BJT	-	Bipolar junction transistor
MOSFET	-	Metal oxide semiconductor field effect transistor

IGBT	-	Insulated gate bipolar transistor
GTO	-	Gate turn off
DSP	-	Digital signal processor
MIPS	-	Measurement in millions of instruction per second
HDL	-	Hardware description language
IC	-	Integrated circuit
RAM	-	Random Access Memory
HDMI	-	High Defination Multimedia Interface
USB	-	Universal Serial Bus
PC	-	Personal Computer
LED	-	Light Emitting Diode
SOPC	-	System-on-a-programmable-chip
MATLAB	-	MATrix LABoratory
RMS	-	Root mean square
TNB	-	Tenaga Nasional Berhad

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Research Background

A conventional Sinusoidal Pulse Width Modulation (SPWM) technique is not practical to be adapted in Alternating Current (AC) motor drives due to inflexibility control of three-phase quantities and lower output voltages. Space Vector Modulator Width Modulation (SVPWM) or known as Space Vector Modulator (SVM) technique was intensively used for obtaining high performances of AC motor drives such as produce a constant switching frequency in Direct Torque Control (DTC), fast dynamic torque control with over modulation strategy adopted in SVM and improved torque control capability for a wide-speed range. SVM is one of the approached to improve the SPWM technic for a better performance.

1.2 Motivation

The development of high performance drive technology is a center in the industry because technologies become more advanced from time to time. Industries that used AC drives, mostly are necessary to be conducted at different speed because in industries used different types of drives and machines. As for that, to get a variable speed, this machine is fed from inverters with variable voltage and variable frequency supply. So, using SVM with Field-Programmable Gate Array (FPGA) can increase the performance of the three-phase inverter. This is because FPGA used very high speed integrated circuit VHSIC Hardware Description Language (VHDL), where it is capable to perform Pulse Width Modulation (PWM) control algorithm especially SVM at high speed calculation.

1.3 Problem Statement

The problem is divided into two which is the SPWM technique and the existing SVM works. The SPWM techniques are inflexible control because difficult to control three-phase quantities such as amplitudes, frequencies and phase angle of three-phase voltage. It is according to the torque and flux demands in vector control of induction motor. This technique also has a limit for output voltage such as it cannot fully utilize the DC link voltage. Next, the problem with the existing SVM works because it totally implement the complex calculation of SVM algorithm and vector control in DSP which can increase the computational burden and hence reduce AC drive performance as the sampling time increase.

1.4 Objectives

In that respect are various objectives to be achieved in this task which include:

i. To design and implement a space vector modulator utilizing FPGA which produce appropriate switching states, according to the inputs of d- and q- axis of space vector for measuring the output voltage and Fast Fourier Transform (FFT) Analysis. This modulator is suitable to be employed for any type of AC motor controls.

ii. To verify the effectiveness of the design of the SVM modulator through simulation and experiment

1.5 Scopes of Project

The scope of the project is to study various PWM and SVM techniques, especially for AC motor drives. The scope of work is more to the investigate the performance of SVM for two levels with three-phase inverter. Next, to develop simulation models of SVM using MATLAB or SIMULINK. Besides, implement SVM modulator using FPGA. As well as, performing and comparing the results of simulation and experiment to validate the effectiveness of SVM.

1.6 Report Outline

Chapter 1 Introduction

This chapter will brief explains the main idea of this project is discussed in the overview. The idea is then elaborated in research background, objectives and scopes of work.

Chapter 2 Literature Review

The review of basic inverter and principle PWM method that will used in this project. Besides, this chapter summed up the research information in related previous work and journals.

Chapter 3 Methodology

The overall of this chapter is discussed the principle of SVPWM switching technique for three-phase Voltage Source Inverter (VSI). The flow of the project is explained and illustrated in flow chart. Besides, in this chapter also will this discuss the simulation of the MATLAB/Simulink. In this chapter, the hardware setup will concisely explain.

Chapter 4 Result and Discussion

The software and hardware is used for simulation and experimental is described. This chapter will discuss the comparison result between SPWM and SVM. Besides, this chapter discusses about the switching result and the SVM result from the simulation and hardware, which is the output voltage and FFT Analysis.

Chapter 5 Conclusion and recommendation

This chapter will summarize the idea about this report and there will be recommended for the future works.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This chapter discusses about the study from the journal or technical paper that related to the three-phase inverter which contain Voltage Source Inverter (VSI) and Current Source Inverter (CSI). This review also includes the Pulse Width Modulation (PWM) such as Sinusoidal Pulse Width Modulation (SPWM), SPWM with third harmonic injection and Space Vector Modulation (SVM). This review will briefly explained about the two levels of three-phase inverter to justify the main topic in this report. Besides, this review also discusses about the connection of Field-Programmable Gate Arrays (FPGA).

2.2 Inverter

An inverter is used to convert Direct Current (DC) to Alternating Current (AC) at desired output voltage and frequency. The application that will be used inverter is grid-connected system, Uninterruptible Power Supplies (UPS), High Voltage Direct Current (HVDC) power transmission and adjustable speed motor drive. An inverter function to supply an induction motor and need a switching device capable of being turned off and on through the gate. The output voltage can be controlled with the help of drives and the switches.

Three phase inverter is a large application in the industry. Adjustable Speed Drives (ASD) or known as Variable Speed Drives (VSD) where the motor's supplied voltage and frequency of power is changing by hold at the speed of an AC induction motor. Using this approach the energy can save because the speed of motor can vary according to the situation. Figure 2.1 shows the example application of ASD.

Figure 2.1 : Adjustable Speed Drives (ASD)

Next, three phase inverter can act as an Uninterruptible Power Supply (UPS) or known as a backup power supply. When the main power from the supplier is discontinuous, it is function to provide and supply the energy from the battery stored while protecting the hardware. Figure 2.2 shows the example application of UPS.

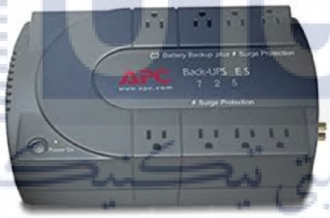


Figure 2.2 : Uninterruptible Power Supply (UPS)

High Voltage Direct Current (HVDC) gives permission between unsynchronized AC transmission systems for power transmission. This system can stabilize a network against disturbances due to rapid changes in power using the force flow over an HVDC link. Besides, it also can go through independently of the phase angle between source and load. Figure 2.3 shows the illustration application of UPS.

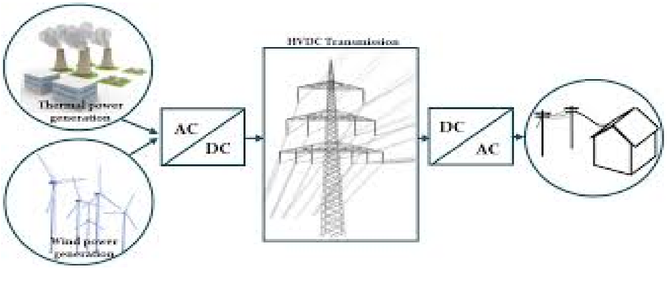


Figure 2.3 : Illustration of High Voltage Direct Current (HVDC)

Next, there are a lot of devices that can be used such as Bipolar Junction Transistor (BJT), Metal Oxide Semiconductor Field Effect Transistor (MOSFET), Insulated Gate Bipolar Transistor (IGBT), Gate Turn Off (GTO) or Thyristor with external commutation circuit [1]. A device that will be used in this project is IGBTs because it is a voltage controlled devices and only requires a small voltage on the gate to maintain conduction through the device.

Besides, there are two kind of inverter which is VSI and CSI. The VSI is when the DC voltage remains constant while the CSI is when the input current is maintained constant. As for that, the chosen inverter is VSI because it is more effective, have higher dependability and rapid in dynamic response [2]. Besides, the amplitude of output voltages from inverter does not depend on the load. So, using VSI is more advantage compared to the CSI and thus the three-phase inverter will be used.

2.3 Pulse Width Modulation (PWM)

The pulse width modulation approach is most usually used to control the output voltage of inverters and improve performances of inverter. This method is used in inverter to obtain a variable voltage and variable frequency of the output AC waveform for use in most variable speed motor drives. It also used to modulate the duration of the pulse or duty ratio in order to achieve voltage or current and frequency. As for that, there are many ways for PWM control technique such as SPWM, SPWM with third harmonic injection and SVM.

SPWM is a popular modulation technique because it can decrease the filter requirements for harmonic reduction and controllability of the amplitude for the output voltage while controlling the switching operation of three-phase inverter. SPWM used simple control strategy by comparing reference signal with carrier signal [3]. SPWM is about the comparison between the high frequency carrier wave with the sinusoidal modulating signals which will be used to produce the appropriate gating signal for the inverter [4]. As for that, each switch is controlled by comparing a sinusoidal reference wave with a triangular carrier wave.

The reference wave and the amplitude of the output waveform is the same with the output of fundamental frequency which can be found by the relative amplitudes of the references and carrier waves. Refer to the figure 2.4, the upper switch is turned on and the lower switch is turned off when the amplitude waveform is greater than the triangular waveform,. On the other hand, the upper switch is off and the lower switch is on when the sinusoidal waveform is less than the triangular waveform,.

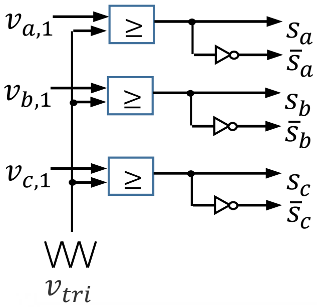


Figure 2.4: PWM control circuit

The performance analysis of output voltage is based on the modulating index, M_I.

$$M_I = \frac{V_a}{V_{tri}} \qquad (2.1)$$

The amplitude modulating waveform, V_a is half of the V_{dc} which multiplies with the modulation index as (2.1) and the V_{tri} is the amplitude of triangular waveform.

$$V_{a,1} = \frac{V_{dc}}{2} M_I \qquad (2.2)$$

The modulation ratio is also known as frequency ratio can be described as:

$$M_R = \frac{f_{tri}}{f_m} \qquad (2.3)$$

There is some benefit for three-phase inverter if M_R Value is odd, so all even harmonic will be eliminated from the pole-switching waveform, but if the value is triplens or multiple number of three, so all triplens harmonic will be eliminated from the line-to-line output voltage.

SPWM with third harmonic injection can lead to the decrement in the peak of the resultant modulating signal and thus can increase the output voltage up to the 15.47%. As for that, the peak of the resultant modulating signal can reach at the peak value of triangular wave. Where the normal SPWM can only reach half value of the V_{dc} with the value of modulation index equal to 1, but with third harmonic injection it can increase up to the $0.575V_{dc}$ [4]. During normal SPWM the value for a DC-link voltage is not fully utilized, so a lot of AC drive needed an output voltage for improving power output and dynamic performances.

SVM has been chosen for this project due to the less voltage or current distortion, higher output voltages and flexibility to be adopted for many AC drive systems. Next, SVM also provides a constant switching frequency and produces higher fundamental AC voltage. SVM is basically based on the figure 2.5 and this hexagon diagram contain six sectors with the two planes at the stator which is direct voltage, v_d and quadrature voltage v_q. Each sector has 60 degrees and have different type of switching. Which sector will be chosen is depend on the voltage references as a red dot in the sector 1 based on the figure 2.5. This voltage reference is given in terms of rotating space vector. The voltage vectors are separated into two sectors, such as active vectors and zero vectors. For active vectors (V_1, V_2, V_3, V_3, V_5 and V_6) and zero vectors (V_0 and V_7) . Next, the SVM method is based on the Clarke Transformation which is the sinusoidal three-phase variables can be indicate on a two orthogonal axis (αβ) [4]. As for that, three-phase voltage vector can be express as:

$$\begin{bmatrix} V_d \\ V_q \end{bmatrix} = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & -\frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix}\begin{bmatrix} V_a \\ V_b \\ V_c \end{bmatrix} \tag{2.4}$$

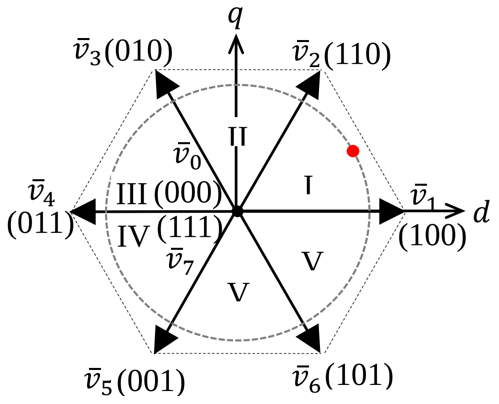


Figure 2.5 : Basic diagram SVM

2.4 Digital Signal Processor and Field-Programmable Gate Array

Digital Signal Processor (DSP) is struggling to capture and process to get the output data without any loss compared to Field-Programmable Gate Array (FPGA). DSP coding using high language such as C language or used the algorithm as a block diagram with a graphic user interface [5].FPGA is clock based, so each clock has the promising ability to perform a mathematical operation on the incoming data stream. A lot of engineers choose FPGA compared to DSPs because of application Measurement in Millions of Instruction per Second (MIPS) needed [6].

An FPGA is a high performance device with the integration of many fuctions. It has a high density to create many complex logic functions. There a lot of available input or output standard and features. Besides, FPGA is a fast programming with Hardware Description Language (HDL) where usually using VHDL and Verilog. FPGA can create many types of configuration which is based on the designer itself to create a digital circuit based on the written HDL. As for that, FPGA is more suitable for this project.

2.5 Review of previous related works

There are many research projects conducted related to SVM for three phase inverter. Three phase inverter is one of the important product in indutry because a lot of machines operated using AC motor type. Besides, SVM has a lot of method for simulation and hardware setup.

Duehee Lee [7] presented the design and implementation of Three-Phase Inverters Using a TMS320F2812 Digital Signal Processor. This thesis explain the method to produces an inverter using TMS320F2812 Digital Signal Processor by Texas Instruments. The puposed for this inverter is for labarotory-scaled wind turbine. This thesis implement the complex calculation of SVM algorithm and vector control in DSP which can increase the computational burden and hence reduce AC drive performance as the sampling time increase. Comparing the FPGA is a high speed controller which can read in nanosecond and conducting the SVM control algorithm using VHDL code.

CHAPTER 3

RESEARCH METHODOLOGY

3.1 Introduction

This chapter interpret about the direction of space vector modulation (SVM) from the start until the end of this project. This is because the research methodology also acts as a guideline to ensure the project is developed systematically and efficiently. As for that, it is briefly explained about the three phase inverter, space phasor, mapping of voltage vectors, space vector modulation technique and the sequences of SVM. So, the basic fundamental of SVM must be discussed clearly so the project can proceed smoothly. Besides, this chapter elaborates the process of the project with the flow chart and the simulation model using Matlab/Simulink.

3.2 Three-phase inverter

Based on figure 3.1, the three-phase inverter circuit consists of six IGBTs components. This topology inverter circuit contains three legs, where each leg has upper and lower IGBTs. The phase terminal is connected in between the upper and lower switches. The switching gate of S_a, S_b and S_c is fed to the upper switch for its leg, while the switching gate of $\bar{S}_a, \bar{S}_b$ and $\bar{S}_c$ is fed to the lower switch in the same leg. Based on the figure 3.2 the simplified circuit of three-phase Voltage Source Inverter (VSI) is connected to the wye-winding of three-phase induction motor and fed by using a DC voltage.

Next, the six bidirectional switches in this circuit are complimentary to each other because of the circuit constraint where the toggle switch at the same leg cannot turn on at the same time because it will lead to the short circuit condition. This means during the upper switching turn on it will be marked as 1 while the lower switching turn on it will be marked as 0. This condition happens because there are only two states possible either 1 or 0. The output voltage of the two-level inverter for each phase can produce two levels of voltage and the switching states can be determined as given in (3.1).

$$v_{xn} = \begin{cases} +\frac{V_{dc}}{2} & \rightarrow (S_x = 1) \\ -\frac{V_{dc}}{2} & \rightarrow (S_x = 0) \end{cases} \quad , n = \text{a-, b- or c-phase} \tag{3.1}$$

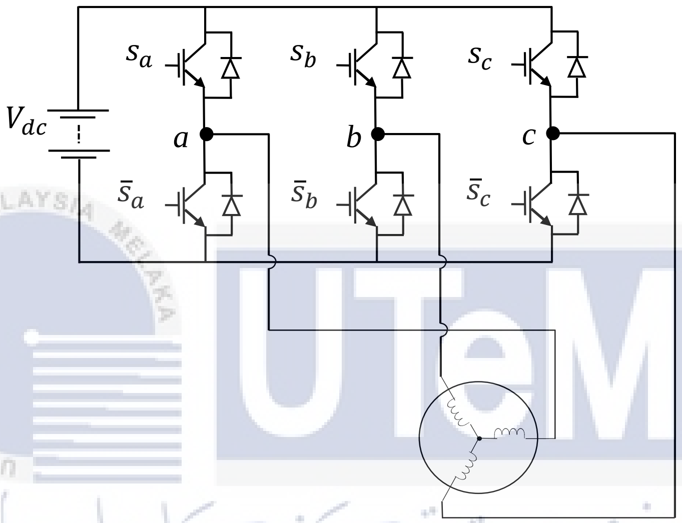


Figure 3.1 : Topology circuit for three-phase VSI

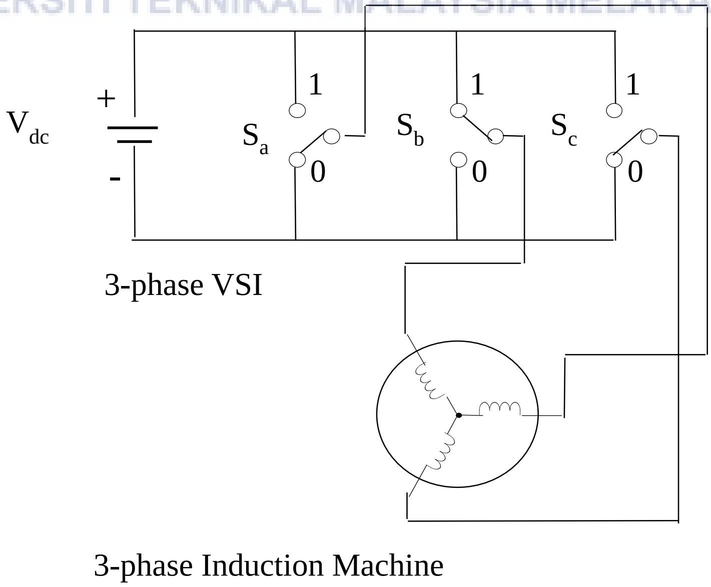


Figure 3.2 : Simplified circuit of three-phase VSI

3.3 Space phasor defination

The voltage space phasor is representing the three-phase voltage by a single vector that in voltage vector plane. The output voltage of a three-phase inverter is expressed in a space vector form for simplifying complex equations that contain three-phase voltage. The space voltage vector for each phase has 120◦ and can be defined as (3.2).

$$\bar{v} = \frac{2}{3}(v_{an} + v_{bn} + v_{cn}) \tag{3.2}$$

Equation (3.3) can be written in terms of a unit vector $\bar{a}$, as shown in (3.3) and shown in figure 3.3.

$$\bar{v} = \frac{2}{3}(v_{an} + \bar{a}.v_{bn} + \bar{a}^2.v_{cn}) \tag{3.3}$$

Space vector can be written in terms of d- and q-axis components of voltage as given in (3.4) and proven through figure 3.3.

$$\bar{v} = v_d + jv_q \tag{3.4}$$

The unit vector $\bar{a}$ can be expressed into an exponent and a rectangular form as represent in figure 3.3.

$$\begin{aligned}\bar{a} &= e^{j2\pi/3} \\ &= \cos\left(\frac{2\pi}{3}\right) + jsin\left(\frac{2\pi}{3}\right) \\ &= -\frac{1}{2} + j\frac{\sqrt{3}}{2}\end{aligned} \tag{3.5}$$

$$\begin{aligned}\bar{a}^2 &= e^{j4\pi/3} \\ &= \cos\left(\frac{4\pi}{3}\right) + jsin\left(\frac{4\pi}{3}\right) \\ &= -\frac{1}{2} - j\frac{\sqrt{3}}{2}\end{aligned} \tag{3.6}$$

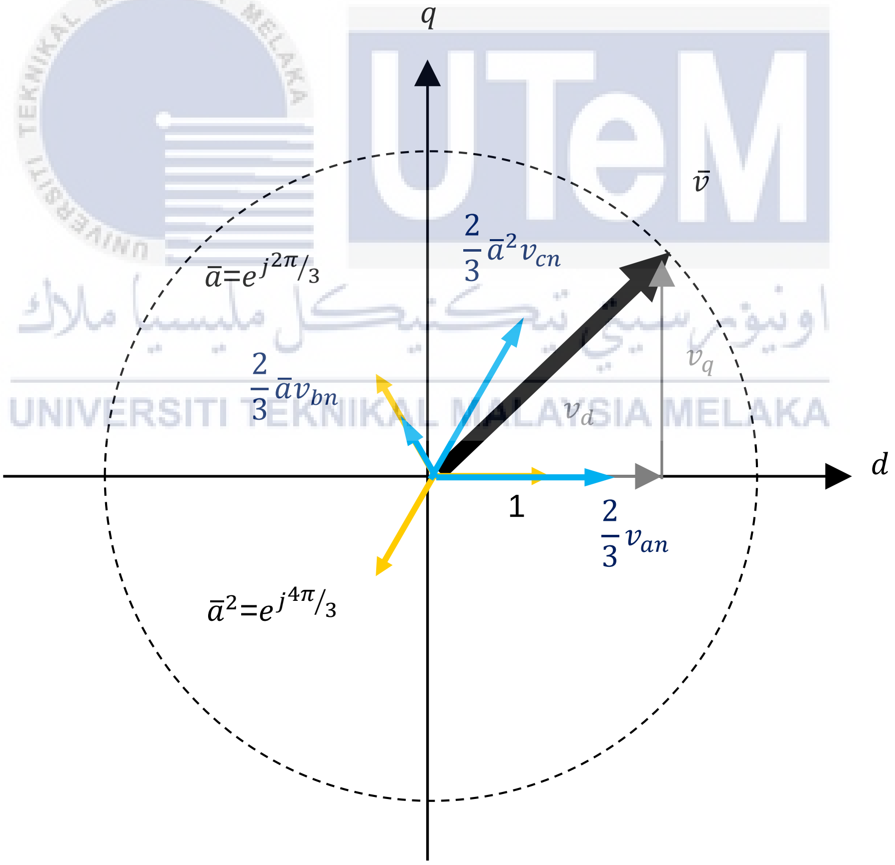
Substitute the rectangular form of unit vectors from (3.5) and (3.6) into (3.3).

$$\bar{v} = \frac{2}{3}\left[v_{an} + \left(-\frac{1}{2} + j\frac{\sqrt{3}}{2}\right)v_{bn} + \left(-\frac{1}{2} - j\frac{\sqrt{3}}{2}\right)v_{cn}\right] \tag{3.7}$$

Then, (3.7) is separated into real and imaginary parts because $\bar{v}$ contain the elements of d- and q-axis as (3.4) which represent the real and imaginary axis respectively.

$$v_d = \frac{2}{3}\left[v_{an} - \frac{1}{2}v_{bn} - \frac{1}{2}v_{cn}\right] \tag{3.8}$$

$$v_q = \frac{1}{\sqrt{3}}\left[v_{bn} - v_{cn}\right] \tag{3.9}$$



q
$\bar{v}$
$\frac{2}{3}\bar{a}^2 v_{cn}$
$\bar{a}$=$e^{j2\pi}/_3$
v_q
$\frac{2}{3}\bar{a}v_{bn}$
v_d
d
1
$\frac{2}{3}v_{an}$
$\bar{a}^2$=$e^{j4\pi}/_3$

Figure 3.3 : Space phasor diagram based on d- and q-axis

The simulation from figure 3.4 is to prove the space phasor definition of the output voltage in a d-q plane where the three-phase quantities (a, b and c) are transforming into two phase quantities (d and q) as given in figure 3.5.

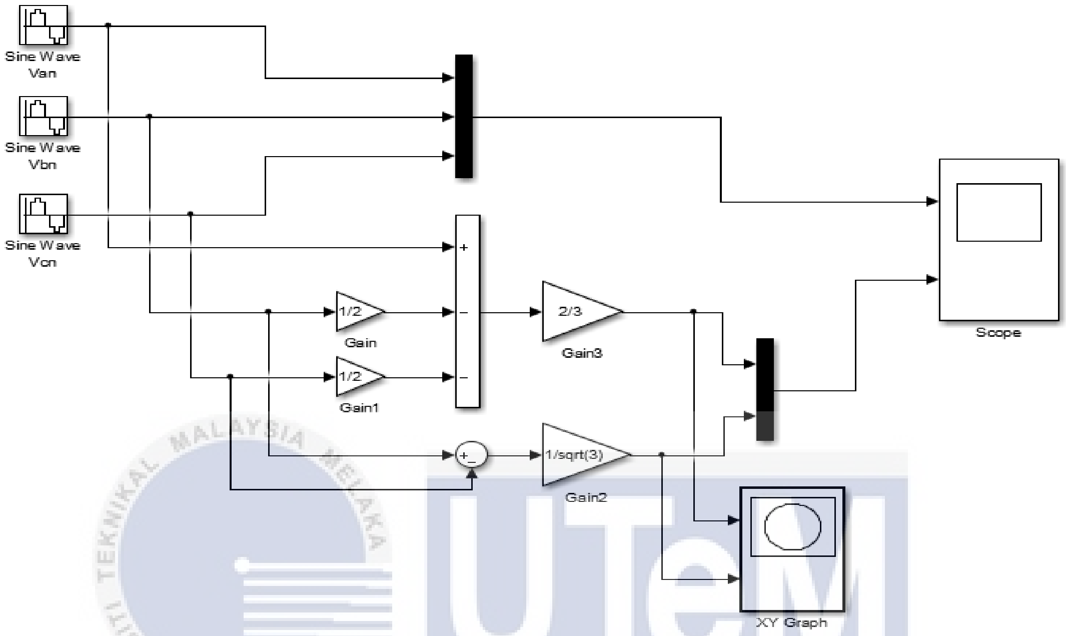


Figure 3.4 : Simulation for space phasor

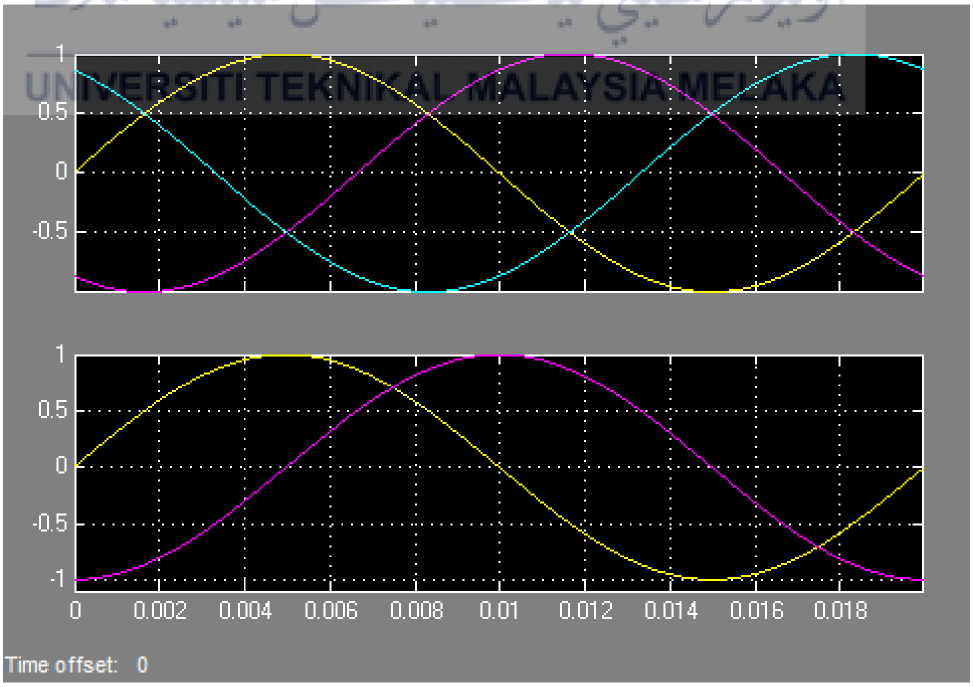


Figure 3.5 : Result transformation of three phase to two phase quantities

3.4 Mapping of voltage vector

Two levels of three-phase voltage inverter contain eight possible switching pattern according to (3.10) and table 3.1, thus it contains two types of vector which is zero vectors (V_7 and V_0) and active vectors ($V_1,\ V_2, V_3, V_3, V_5$ and V_6).

$2^n = 2^3 = 8$ possible switching , $n =$ type of phase (3.10)

Table 3.1 : Switching vectors according to voltage vectors

Voltage vectors	Switching vectors		
	a	b	c
V_7	1	1	1
V_0	0	0	0
V_1	1	0	0
V_2	1	1	0
V_3	0	1	0
V_4	0	1	1
V_5	0	0	1
V_6	1	0	1

Mapping voltage vectors in figure 3.6 can be produced by rewriting (3.3) in terms of switching states.

$$\bar{v} = \frac{2}{3}V_{dc}(S_a + \bar{a}.S_b + \bar{a}^2.S_c) \tag{3.11}$$

Then, (3.8) and (3.9) also can be expressed in terms of switching states.

$$v_d = \frac{2}{3}V_{dc}\left[S_a - \frac{1}{2}S_b - \frac{1}{2}S_c\right] \tag{3.12}$$

$$v_q = \frac{1}{\sqrt{3}}V_{dc}[S_b - S_c] \tag{3.13}$$

So, two-level inverter is during all six active voltage vectors lies along the radial of a hexagon and each sector has 60 degrees. Besides, the six active vectors acts as a DC link voltage which is supplied to the load. The two zero vectors there no voltage is supplied to the load and located at the origin of the hexagon.

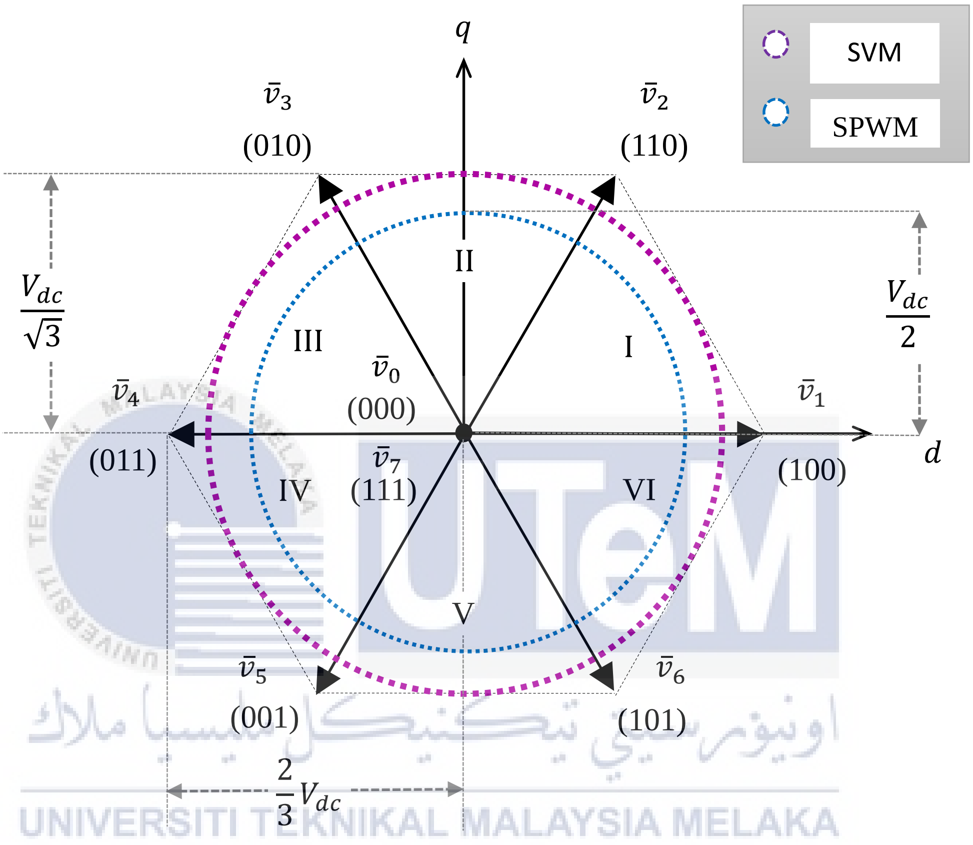


Figure 3.6 : Mapping of voltage vectors

3.5 Space vector modulation technique

The space voltage vector or the magnitude of $\bar{v}$ becomes the voltage references which rotates in the forms of the circular locus at an angular velocity of ω where the movement direction of rotation depends on the sequences of voltages. As for that, the positive sequence will rotate counterclockwise direction while the negative sequence will rotate in the clockwise direction. Next, the total of zero vector and two adjacent voltage vectors to get the resultant space voltage vector for any sectors.

3.5.1 Calculation of on-duration

Diagram in figure 3.7 is general representation which is valid for every sector. Referring the general representation, the position of v_a must always aligned to α-axis. While, the angle within a sector (θ_{sec}) must being in the range where the θ_{sec} more than 0 degrees and less and equal than 60 degrees. The different angle is located between $\bar{v}$ and v_a. Given on the figure 3.7, the resultant space voltage vector for any sector is:

$$\bar{v} = \bar{v}_a \frac{t_a}{T} + \bar{v}_b \frac{t_b}{T} + \bar{v}_z \frac{t_z}{T} \tag{3.14}$$

Where, T is the sampling period.

$$T = t_a + t_b + t_z \tag{3.15}$$

Based on the space voltage vector in (3.14) it can rewrite as a rectangular form.

$$\bar{v} = v_\alpha + jv_\beta \tag{3.16}$$

Used the Trigonometry Identities and the Theorem Pythagoras to get the value of the voltage components of v_α and v_β which act as scalar quantities.

$$v_\alpha = v_a \frac{t_a}{T} + v_b \frac{t_b}{T} cos(60^0) \tag{3.17}$$

$$v_\beta = v_b \frac{t_b}{T} sin(60^0) \tag{3.18}$$

The voltage components of v_α and v_β can expressed in form of voltage space vector $\bar{v}$.

$$v_\alpha = V.\cos(\theta_{sec}) \tag{3.19}$$

$$v_\beta = V.sin(\theta_{sec}) \tag{3.20}$$

So, the angle and the magnitude of the voltage components of v_α and v_β is :

$$V = \sqrt{\left(v_\alpha^2 + v_\beta^2\right)} \tag{3.21}$$

$$\theta_{sec} = tan^{-1}\left(\frac{v_\beta}{v_\alpha}\right) \tag{3.22}$$

Refer to the figure 3.6 to get the magnitude of voltage vectors.

$$v_a = v_b = \frac{2}{3}V_{dc} \tag{3.23}$$

Applying (3.17), (3.18) and (3.23) to get the on-duration of t_b and t_a.

$$t_b = \sqrt{3}.T\frac{v_\beta}{V_{dc}} \tag{3.24}$$

$$t_a = \frac{3}{2}\frac{T}{V_{dc}}\left[v_\alpha - \frac{v_\beta}{\sqrt{3}}\right] \tag{3.25}$$

Rewrite (3.15) to get the on duration of t_z.

$$t_z = T - (t_a + t_b) \tag{3.26}$$

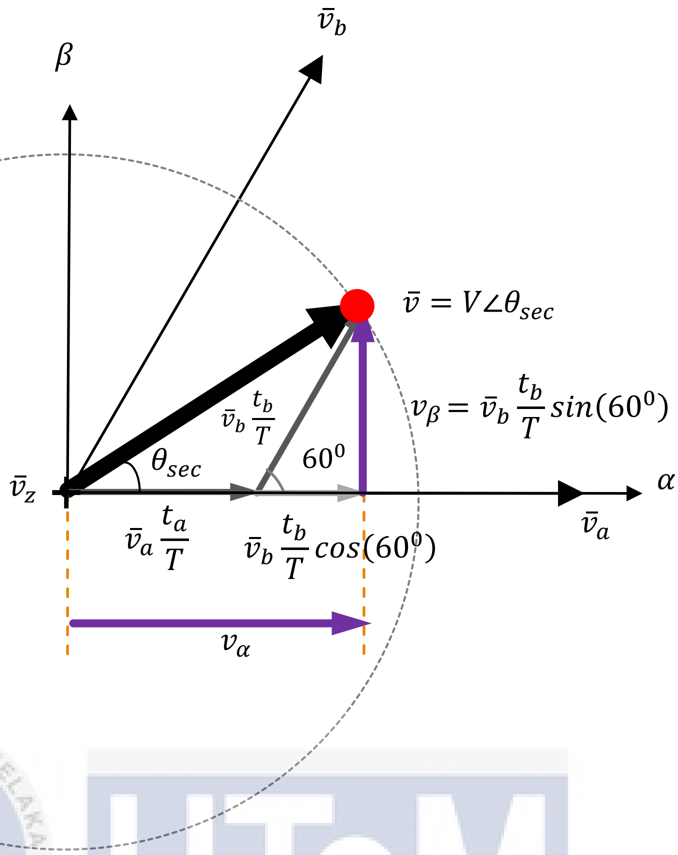

$\bar{v}_b$
β
$\bar{v} = V\angle\theta_{sec}$
$v_\beta = \bar{v}_b \frac{t_b}{T} sin(60^0)$
$\bar{v}_b \frac{t_b}{T}$
θ_{sec}
60^0
$\bar{v}_z$
α
$\bar{v}_a$
$\bar{v}_a \frac{t_a}{T}$
$\bar{v}_b \frac{t_b}{T} cos(60^0)$
v_α

Figure 3.7 : General representation for every sector

The effect of on-time at the $\bar{v}$ location will have three cases. The first cases happen went $\bar{v}$ is closed to v_α based on the figure 3.8(a) and during this condition the time on t_a is more than t_b. Given the second case in the figure 3.8(b), $\bar{v}$ is located at the middle of $\bar{v}_a$ and $\bar{v}_b$. As for that, time on the t_a and t_b is equal. While for the third case on the figure 3.8(c), $\bar{v}$ is being closed to $\bar{v}_b$ and the time on t_b is more than t_a. So all these three cases is based on the position of the space voltage vector.

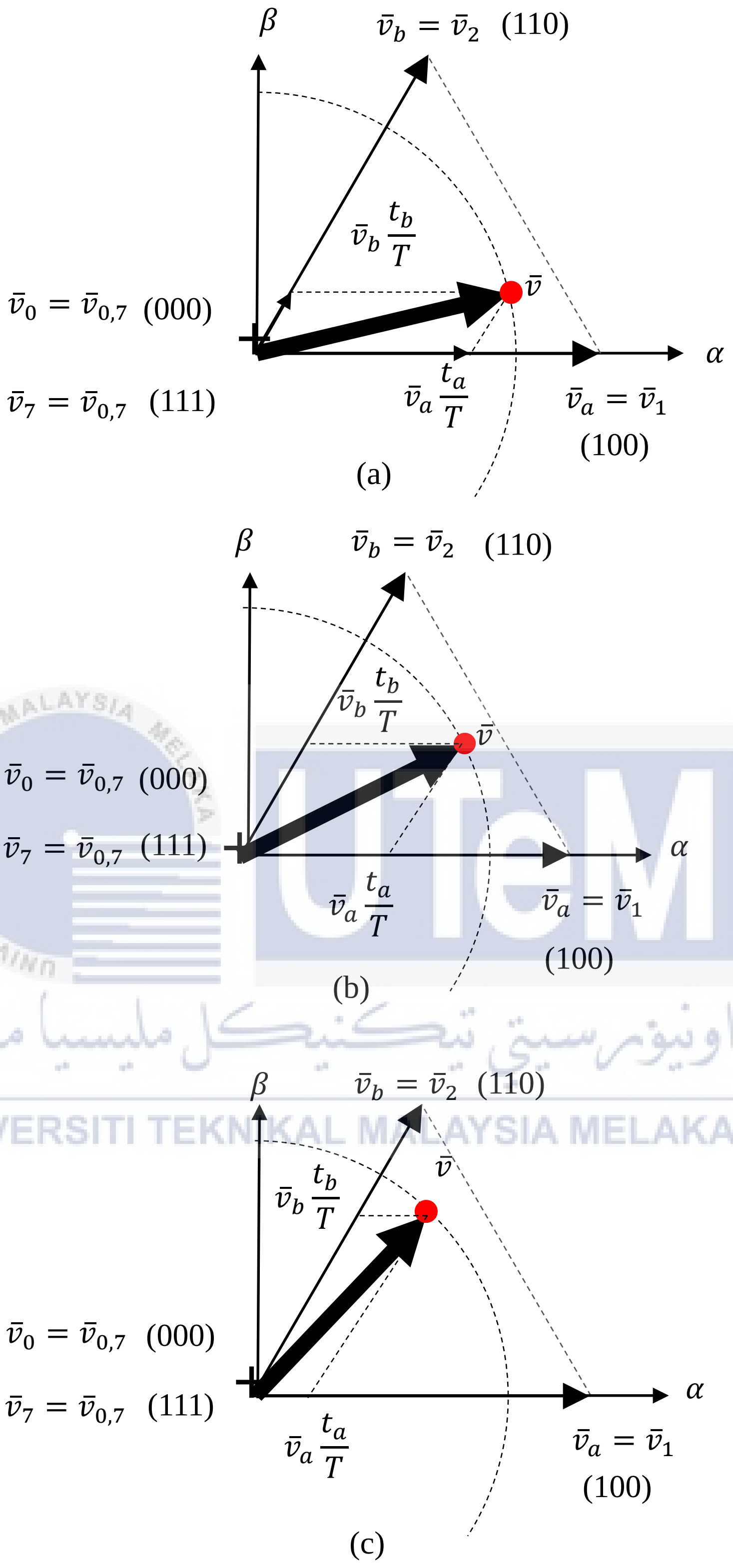

β
$\bar{v}_b = \bar{v}_2$ (110)
$\bar{v}_b \frac{t_b}{T}$
$\bar{v}$
$\bar{v}_0 = \bar{v}_{0,7}$ (000)
α
$\bar{v}_7 = \bar{v}_{0,7}$ (111)
$\bar{v}_a \frac{t_a}{T}$
$\bar{v}_a = \bar{v}_1$
(100)
(a)
β
$\bar{v}_b = \bar{v}_2$ (110)
$\bar{v}_b \frac{t_b}{T}$
$\bar{v}$
$\bar{v}_0 = \bar{v}_{0,7}$ (000)
$\bar{v}_7 = \bar{v}_{0,7}$ (111)
α
$\bar{v}_a \frac{t_a}{T}$
$\bar{v}_a = \bar{v}_1$
(100)
(b)
β
$\bar{v}_b = \bar{v}_2$ (110)
$\bar{v}$
$\bar{v}_b \frac{t_b}{T}$
$\bar{v}_0 = \bar{v}_{0,7}$ (000)
α
$\bar{v}_7 = \bar{v}_{0,7}$ (111)
$\bar{v}_a \frac{t_a}{T}$
$\bar{v}_a = \bar{v}_1$
(100)
(c)

Figure 3.8: Location space voltage vector (a) Case 1 is $\bar{v}$ closed to $\bar{v}_a$, (b) Case 2 is $\bar{v}$ at the middle $\bar{v}_a$ and $\bar{v}_b$, (c) Case 3 is $\bar{v}$ closed to $\bar{v}_b$

3.5.2 Sequences of selection for voltage vectors

Pattern for selection or switching is different for every sector because itrefersr to the location of the reference space voltage vectorr. The sequences, selection of voltage vectors must start at the beginning and ending of the switching period, which is the zero voltage vector $\bar{v}_0$ with states (000) must be selected. Next, it is desirable to minimize the number of switching states commutation and thus to identify the voltage vectors of $\bar{v}_a$ and $\bar{v}_b$. Besides, figure 11 showing the sequence of space voltage vector within the sector I. As for that, the sequence start with zero voltage vector which is $\bar{v}_0$ and goes to the active voltage vectors. Then continue at the zero voltage vector which is $\bar{v}_7$ with the states (111) and goes back to the active voltage vectors and stop at te zero active vector. For instance, $\bar{v}_0 \rightarrow \bar{v}_1 \rightarrow \bar{v}_2 \rightarrow \bar{v}_7 \rightarrow \bar{v}_2 \rightarrow \bar{v}_1 \rightarrow \bar{v}_0$ is the sequence for sector I based on the figure 3.11.

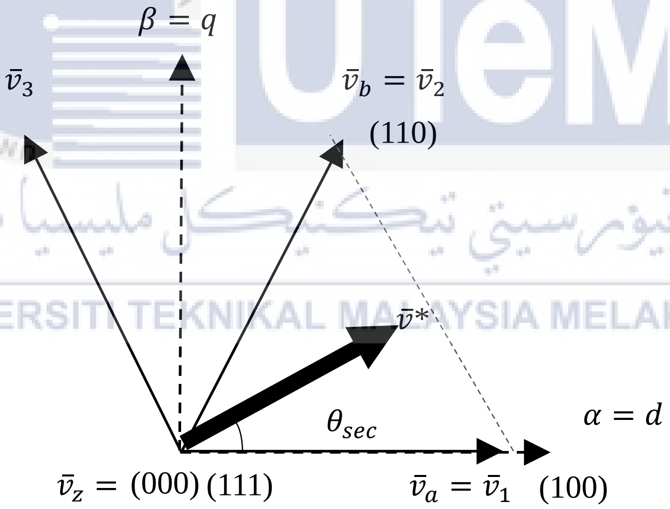


Figure 3.9 : $\bar{v}$ is located within sector I

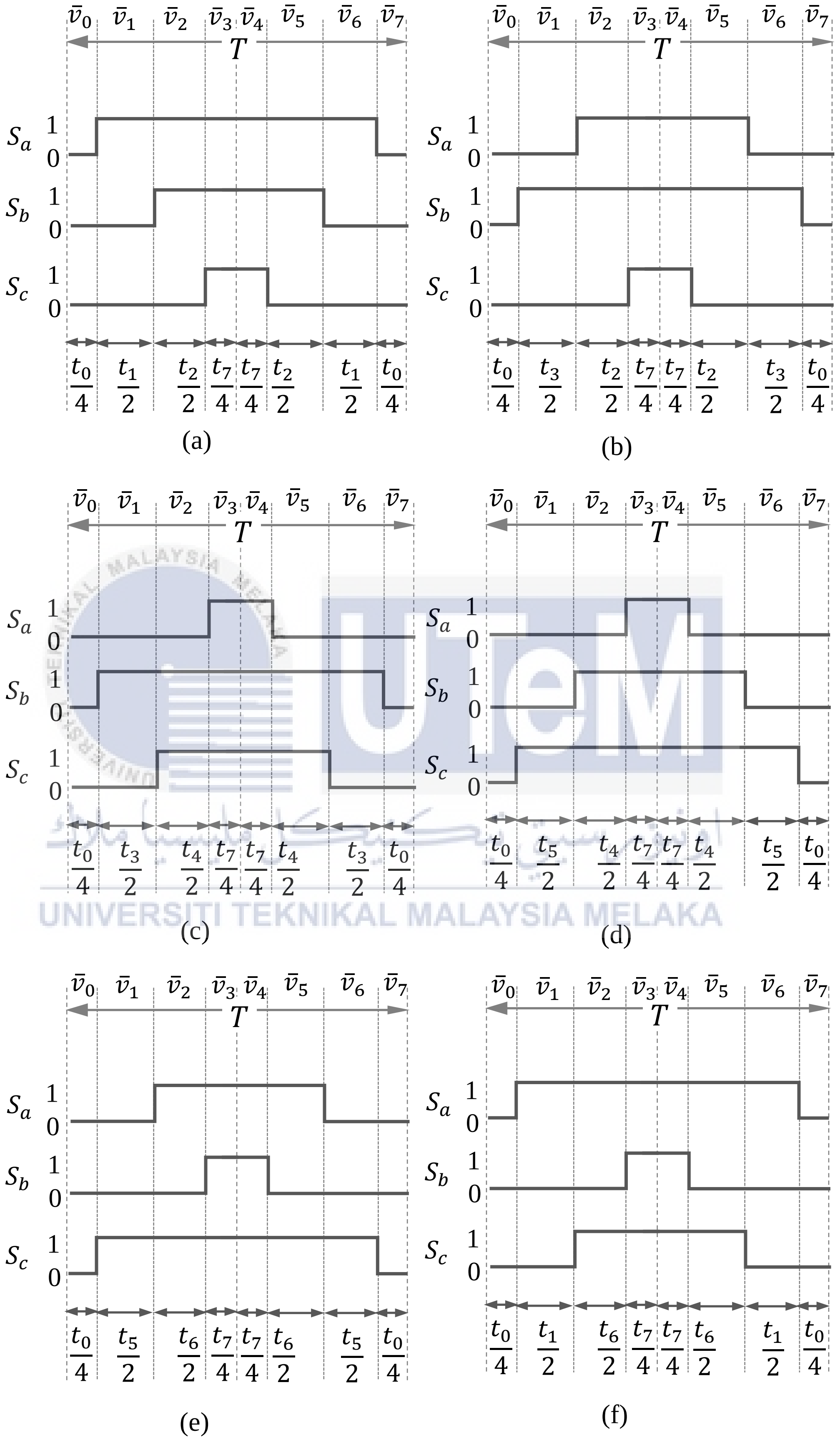

$\bar{v}_0$ $\bar{v}_1$ $\bar{v}_2$ $\bar{v}_3$ $\bar{v}_4$ $\bar{v}_5$ $\bar{v}_6$ $\bar{v}_7$
T
S_a
S_b
S_c
1
0
(a) $\frac{t_0}{4}$ $\frac{t_1}{2}$ $\frac{t_2}{2}$ $\frac{t_7}{4}$ $\frac{t_7}{4}$ $\frac{t_2}{2}$ $\frac{t_1}{2}$ $\frac{t_0}{4}$
(b) $\frac{t_0}{4}$ $\frac{t_3}{2}$ $\frac{t_2}{2}$ $\frac{t_7}{4}$ $\frac{t_7}{4}$ $\frac{t_2}{2}$ $\frac{t_3}{2}$ $\frac{t_0}{4}$
(c) $\frac{t_0}{4}$ $\frac{t_3}{2}$ $\frac{t_4}{2}$ $\frac{t_7}{4}$ $\frac{t_7}{4}$ $\frac{t_4}{2}$ $\frac{t_3}{2}$ $\frac{t_0}{4}$
(d) $\frac{t_0}{4}$ $\frac{t_5}{2}$ $\frac{t_4}{2}$ $\frac{t_7}{4}$ $\frac{t_7}{4}$ $\frac{t_4}{2}$ $\frac{t_5}{2}$ $\frac{t_0}{4}$
(e) $\frac{t_0}{4}$ $\frac{t_5}{2}$ $\frac{t_6}{2}$ $\frac{t_7}{4}$ $\frac{t_7}{4}$ $\frac{t_6}{2}$ $\frac{t_5}{2}$ $\frac{t_0}{4}$
(f) $\frac{t_0}{4}$ $\frac{t_1}{2}$ $\frac{t_6}{2}$ $\frac{t_7}{4}$ $\frac{t_7}{4}$ $\frac{t_6}{2}$ $\frac{t_1}{2}$ $\frac{t_0}{4}$

Figure 3.10: Switching status for all sector (a) sector I (b) sector II (c) sector III (d) sector IV (e) sector V (f) sector VI

3.6 Flow chart

Figure 3.11 shows the flow chart which represents the scope of work in the SVM. The construct of coding and calculation in block diagram of MATLAB/Simulink to generate the switching states. Other than that, for the whole process of work has been shown in the Gantt chart and Milestone from the appendix A.

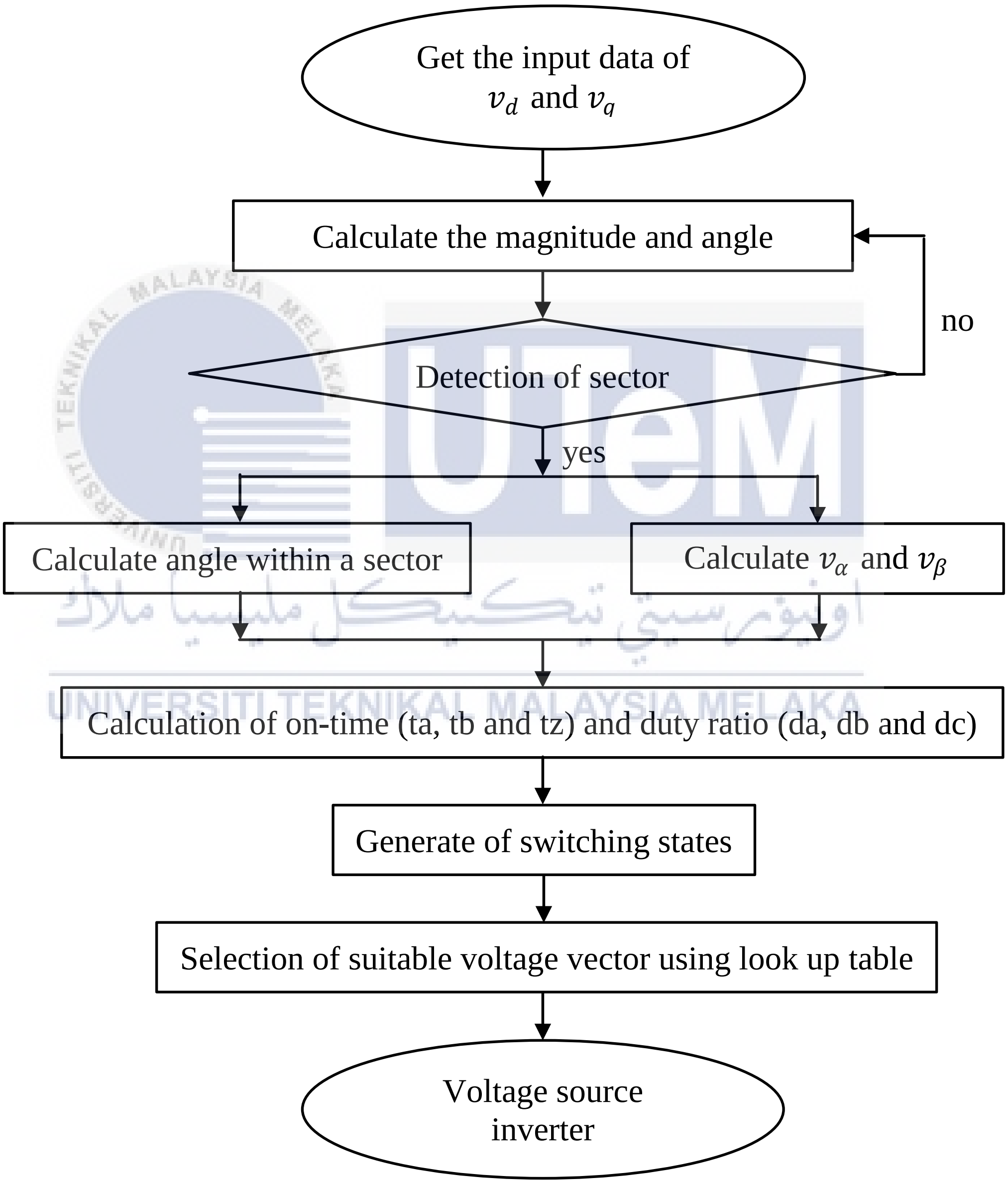

Get the input data of v_d and v_q
Calculate the magnitude and angle
no
Detection of sector
yes
Calculate angle within a sector
Calculate v_α and v_β
Calculation of on-time (ta, tb and tz) and duty ratio (da, db and dc)
Generate of switching states
Selection of suitable voltage vector using look up table
Voltage source inverter

Figure 3.11: Flow chart

3.7 Simulation Model of Space Vector Modulation

This section presents a simulation of SVM using MATLAB /Simulink. Figure 3.12 shows the complete simulation model to control the algorithm of SVM which generates switching status. This simulation model utilizes to generate switching states for driving IGBTs of two level inverter depends on the input values of d- and q- axis compounds of space vector. Basically, the simulation model is constructed using two methods of programming approach utilizing Simulink block diagram and the c-programming approach written in MATLAB function blocks. Based on figure 3.12, in the MATLAB function block there a written code about sector identification about the voltage vector of alpha and beta as Appendix B. Next, the MATLAB Function 1 is about the calculation of duration the coding are also shown in Appendix B is coding about the mapping vector in the MATLAB Function 2.

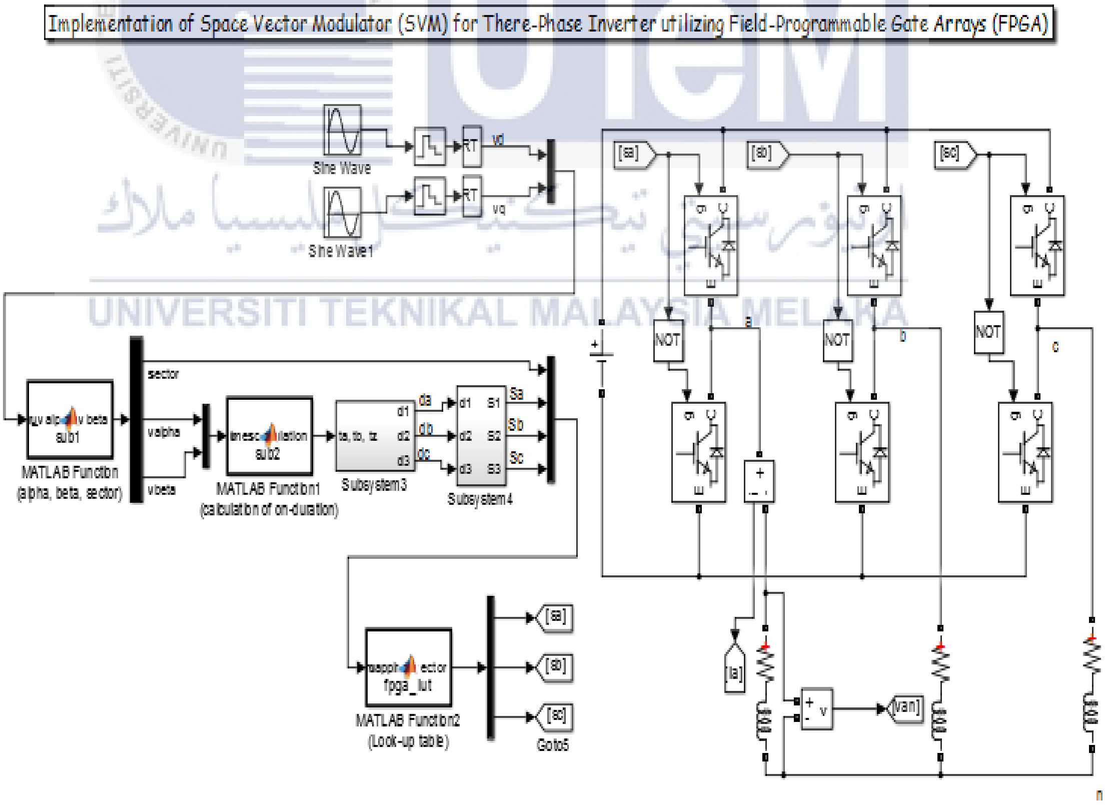


Figure 3.12 : Complete simulation model for space vector modulation

Based on the figure 3.13, the switching period is set at $T = 1/5000$s. The on duration t_a and t_b produced from the MATLAB Function 1 block are used to calculate duty ratios of switching states for each phase such as d_a, d_b and d_c. As figure 3.13 the calculation of duty ratios is performed by Subsystem 3 block. The Simulink block diagram of figure 3.13 is constructed inside subsystem 3.

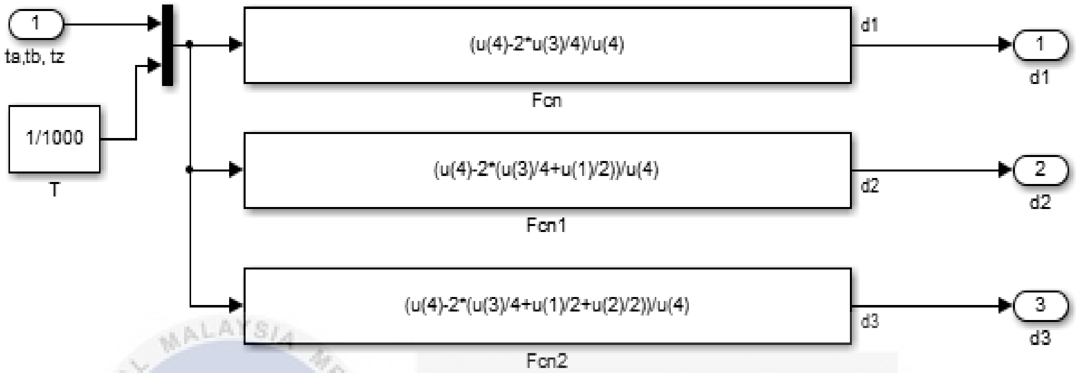

1
ta,tb, tz
1/1000
T
(u(4)-2*u(3)/4)/u(4)
Fcn
(u(4)-2*(u(3)/4+u(1)/2))/u(4)
Fcn1
(u(4)-2*(u(3)/4+u(1)/2+u(2)/2))/u(4)
Fcn2
d1
d2
d3
1
d1
2
d2
3
d3

Figure 3.13: Simulation model of duty ratio calculation at subsystem 3

Figure 3.14 shows a simulation model of switching states which are developed based on the comparison between duty ratios and triangular waveform. The Simulink blocks from figure 3.24 is constructed inside the Subsystem 4.

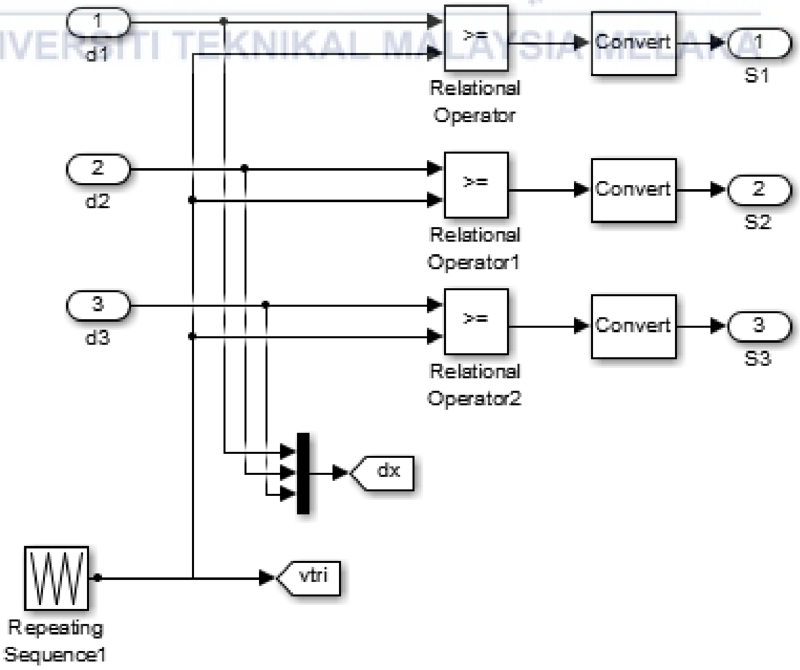


Figure 3.14: Simulation model of switching states at subsystem 4

Lastly, the switching states are used to drive IGBTs of the inverter. Figure 3.15 shows the simulation model of inverter circuit foe two level inverter. The switching of IGBTs is determined by the switching states where the switching of upper and lower IGBTs for each leg must be complimentary to each other. The output of each inverter is associated to the resistive and inductive loads.

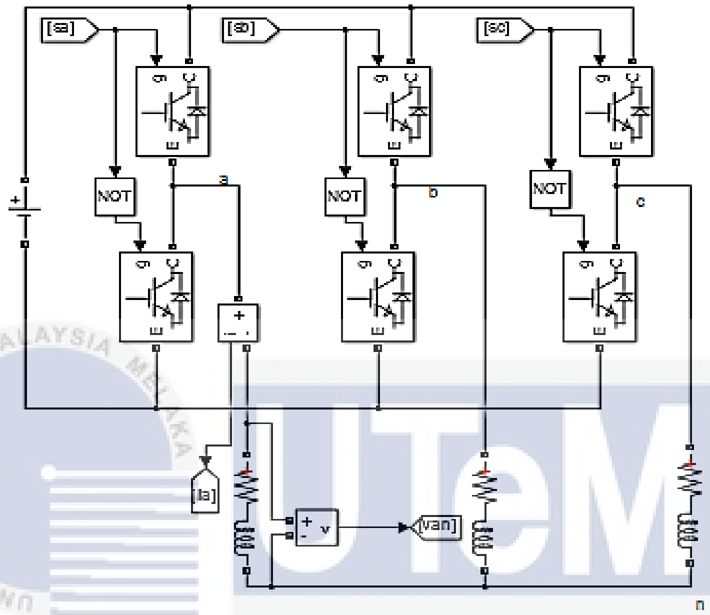


Figure 3.15: Simulation model of inverter circuit foe two level inverter

3.8 Detail of the hardware setup

This segment shows the detail of the explanation about the component which has been set up for the experimental platform for authenticate the capability of SVM control algotrithm for two-level inverter. All the component that assembles on the circuit is based on the schematic diagram designed into simulation part. Figure 3.16 shows the experimental setup while figure 3.17 in the hardware and all hardware setup is stated in table 3.2. Based on the figure 3.17, one of the supply will directly connect to the FPGA board to supply the 5V. The other supply connects to inverter and the voltage is up to the 50V. Next, laptop will be used to upload the program to the FPGA. The FPGA circuit is connected to the inverter using rainbow cable. The output Sa, Sb and Sc are connected to the series connected resistive and inductive loads. Then from that it goes to the oscilloscope.

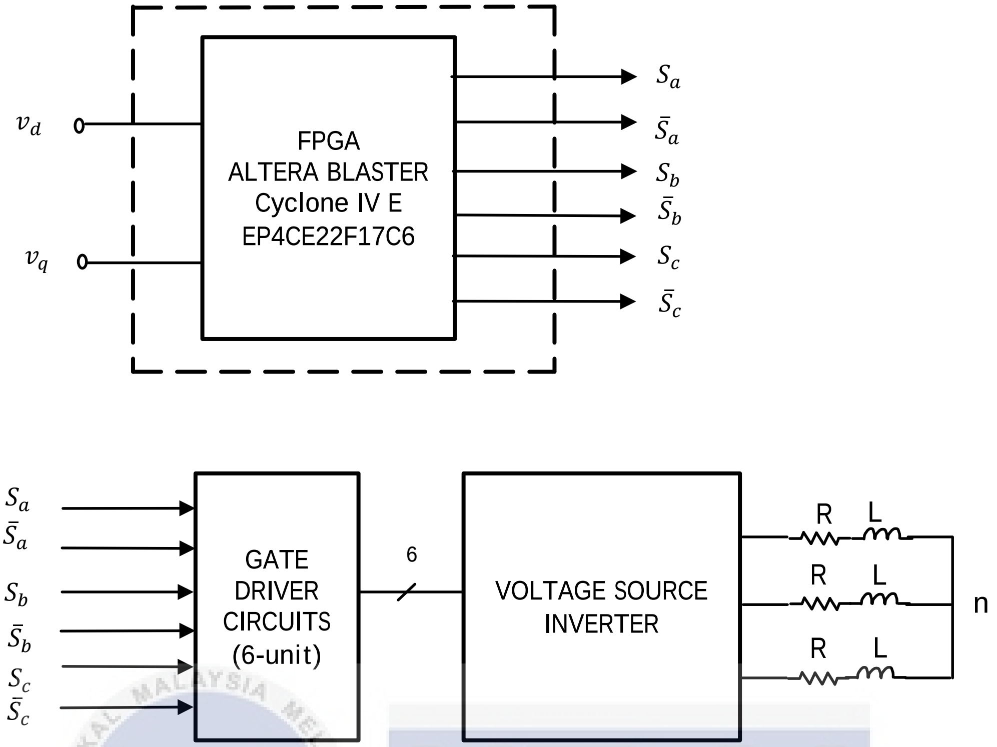


Figure 3.16 : Experimantal setup

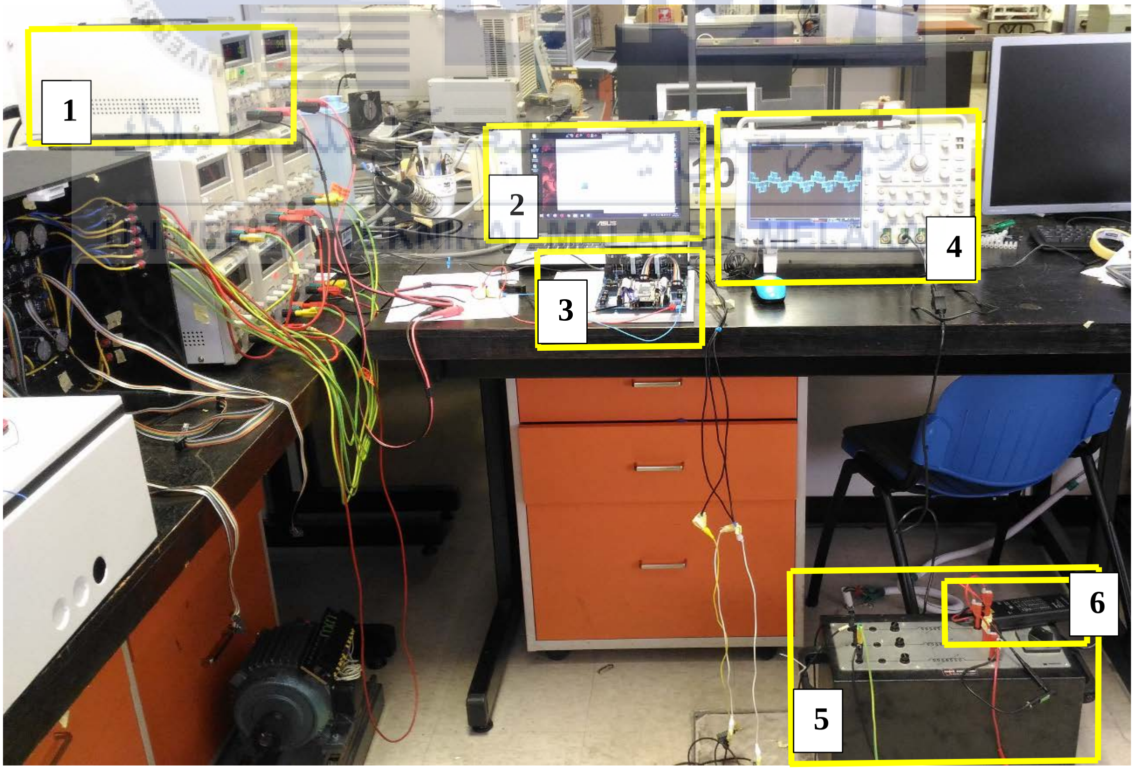


Figure 3.17 : Hardware setup

Table 3.2 : Hardware component description

No	Component	Description
1	Two power supply	Connect to inverter and power circuit of FPGA
2	Laptop	Simulation MATLAB and Quartus
3	Inverter and FPGA	Inverter for convert DC to AC voltage and FPGA to load the program
4	Oscilloscope	The output waveform
5	Series Connected Resistive and Inductive Loads	Connect to the phase a, b and c at inverter.
6	Voltage probe	Measure the voltage and goes to oscilloscope

3.8.1 Inverter hardware

This inverter combines with the gate driver to become one completed circuit as figure 3.18. The inverter side contains a heat sink for absorbing the unwanted heat. There are eight of IGBTs will attach to the to the heat sink, but before it attached the mica sheet will be attached first to avoid the short circuit. This circuit contains two capacitors. There also connector goes to the DC-link and also connector goes to the load which is sa, sb and sc. The gate driver side, there are six DC-DC Converter and six Optocoupler. There are two Integrated Circuit (IC), resistor and also capacitor. There are connector goes to the FPGA circuit using rainbow cable.

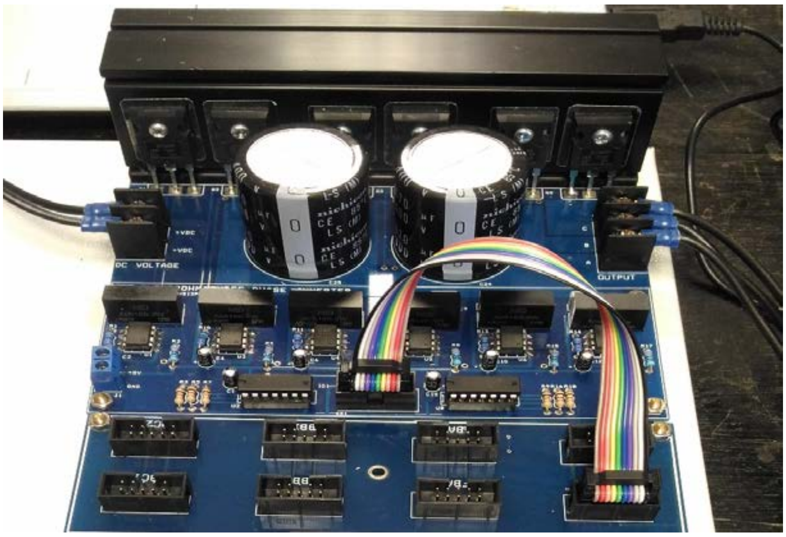


Figure 3.18: Inverter with the gate driver

3.8.2 Field-Programmable Gate Array

FPGA can be used to create a fast low logic devices where it only need to use a code and FPGA chips wil create a logic for designers. As for that, it is suitable for high speed interfaceses because it contain Random Access Memory (RAM), High Defination Multimedia Interface (HDMI) and Universal Serial Bus (USB). Besides, it can create a softcore such as building a microcontroller with a logic at FPGA.

FPGA has power entirely by USB port where during developing it did not need a power supply. The limitation is it possibly not get enough power for Input and Output pin, but it can plug a five volt supply at the FPGA. It comes with a USB Blaster which is a device that connect from a Personal Computer (PC) USB port to the programming header of a device. On the FPGA board there is eight of Light Emitting Diode (LED) to act as an indicator to make sure whether the coding has been uploaded or not. Figure 3.19 shows the Altera FPGA Cyclone IV DEO-Nano Controller Board.

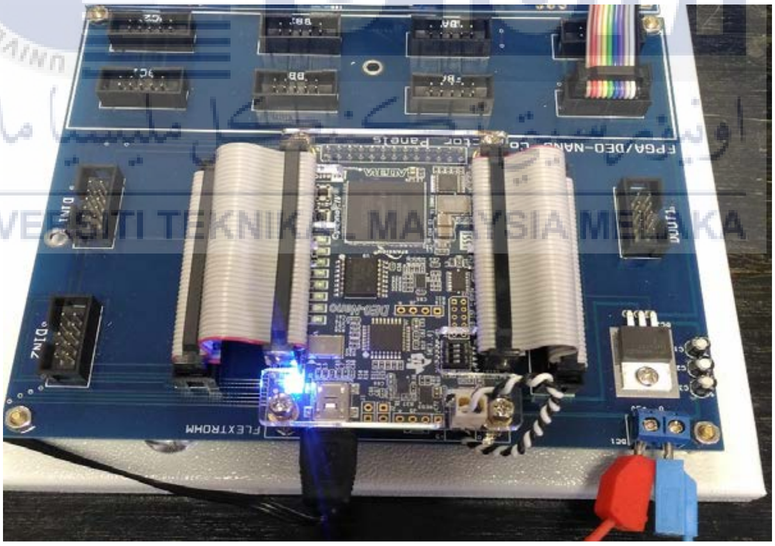


Figure 3.19 : The Altera FPGA Cyclone IV DEO-Nano Controller Board

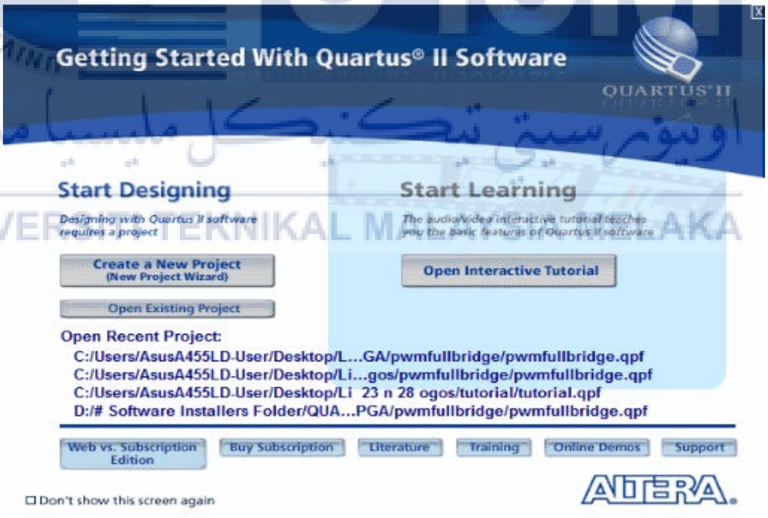
3.9 Software

Software is an important part to make sure that hardware can run properly. Software is a safety precaution before run the hardware. Software also gives a direction to the hardware for running properly.

3.9.1 Quartus II (Altera)

Altera Quartus II is used to run the FPGA as shown in figure 3.20. This software gives a perfect design with a lot of fuction that facilitates the user to adapt their specific design needs. As for that, it is a suitable for system-on-a-programmable-chip (SOPC) design. It allows users to use the Quartus II graphical user interface and command-line for each phase of the design flow. Besides, users also can use one of these interfaces for the overall flow, or they can use another alternative at different phases.

This software also contains a modular Compiler which includes a module such as Analysis and Synthesis, Partition Merge, Fitter, Assembler, TimeQuest Timing Analyzer, Design Assistant, EDA Netlist Writer and HardCopy Netlist Writer [8]. To run all Compiler module as part of a full compilation, on the processing menu, click Start Compilation. User can also run each modules individually by pointing to Start on the processing menu, and then clicking the command for the modules to start.



Getting Started With Quartus® II Software

Start Designing

Designing with Quartus II software requires a project

Start Learning

The audio/video interactive tutorial teaches you the basic features of Quartus II software

Open Recent Project:

C:/Users/AsusA455LD-User/Desktop/L...GA/pwmfullbridge/pwmfullbridge.qpf
C:/Users/AsusA455LD-User/Desktop/Li...gos/pwmfullbridge/pwmfullbridge.qpf
C:/Users/AsusA455LD-User/Desktop/Li 23 n 28 ogos/tutorial/tutorial.qpf
D:/# Software Installers Folder/QUA...PGA/pwmfullbridge/pwmfullbridge.qpf

□ Don't show this screen again

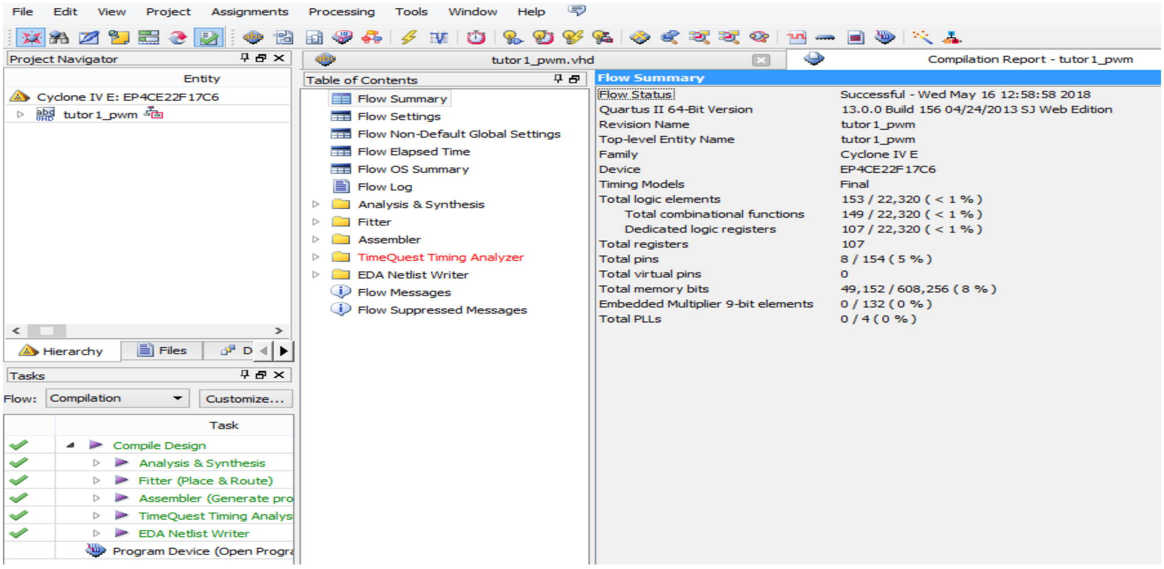
Figure 3.20 : Software Altera Quartus II

The first step to use Altera Quartus II is go to ‘File’ on the menu, click ‘New Project Wizard’ to display Quartus layout. Click ‘Next’ on Introduction page. On page 1 of 5 there will be a Directory, Name and Top-Level Entity, find the directory folder of ‘tuto1_pwm’ and apply the name for the project. Then click ‘Next’ until reach page 3 of 5 about Family & Device Settings. As for that, the selection is set as family select Cyclone IV E, package select FBGA, pin count select 256 and speed grade is any. Then select type of FPGA which is EP4CE22F17C6. Click ‘Next’ until it reaches page 5 of 5 which is Summary thus choose ‘Finish’. Therefore, project’s name is appeared on Quartus layout.

Next, click ‘New’ on menu and select ‘Memory Initialization File’ and type ‘12001’ into no. of words which mean total nombor of data and ‘1’ on word size for take the maximum binary. The purpose for this method is to create .mif file. Based on this project create 12 files and save the file as sa, sb and sc for all switching respectively under ‘tuto1_pwm’ folder. Open the .mif file and also notepad of all files from ‘tuto1_pwm’ folder. On .mif file, copy and paste the excel file to stored the data at the sa. Save and close the both files. The updated .mif file will be displayed at the Quartus. Repeat all step about for sab and sc.

Go to ‘Tools’ on menu and click ‘MegaWizard Plug-In Manager’ as and choose ‘Next’. Select the device family which is Cyclone IV E and choose type of output file, the VHDL. On the left layout of ‘MegaWizard Plug-In Manager [page 2a]’ go the ‘Memory Compiler’ folder and select ‘ROM: 1-PORT’. Meanwhile, give name for the output file similar as .mif file. Click ‘Next’. On ‘ROM: 1-PORT’ page [page 3 of 7], edit q output with 1 bits while memory is 16384 which is near to and more than 12001 words. Click ‘Next’ until it reaches page 5 of 7 and on this page, browse thus insert the sa.mif file and click ‘Next’ followed by ‘Finish’ on the last page. Apply the similar process for sb and sc in the .mif file.

Create VHDL File by selecting ‘New’ on the File menu. Start writing a coding for tuto1_pwm, upper counter, lower counter, comparator, blanking time generators, blanking_mod18 and clk_div as Appendix C into VHDL layout then save the file as ‘tuto1_pwm’.vhd. Modify the entity name into ‘tuto1_pwm’. Save the VHDL File once modification occurs. On the left of FPGA layout, click the ‘Analysis & Synthesis’ on ‘Task’ to analyze the project or click the start compilation on the processing. If the analysis is unsuccessful, thus need to trace the error on ‘Message’ at the bottom of FPGA layout. The compiler will provide error massages, read them throughlt to underdstand what the problem is. Then, the file is saved and analyzed again. Finally, the analysis or compilation is successful as shown in figure 3.21.

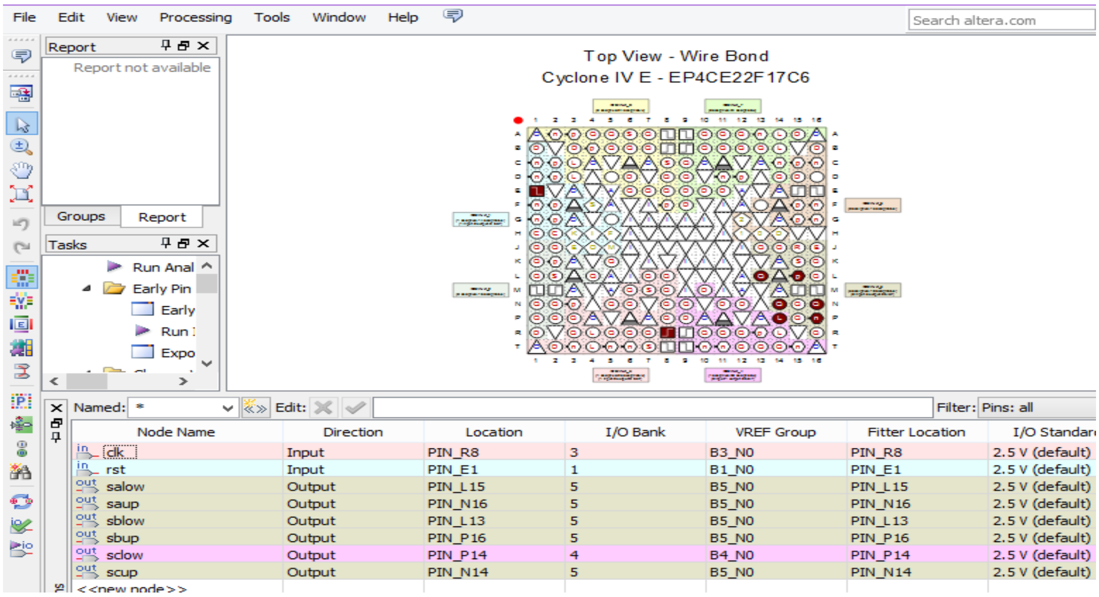


Flow Summary

Flow Status	Successful - Wed May 16 12:58:58 2018
Quartus II 64-Bit Version	13.0.0 Build 156 04/24/2013 SJ Web Edition
Revision Name	tutor1_pwm
Top-level Entity Name	tutor1_pwm
Family	Cyclone IV E
Device	EP4CE22F17C6
Timing Models	Final
Total logic elements	153 / 22,320 (< 1 %)
Total combinational functions	149 / 22,320 (< 1 %)
Dedicated logic registers	107 / 22,320 (< 1 %)
Total registers	107
Total pins	8 / 154 (5 %)
Total virtual pins	0
Total memory bits	49,152 / 608,256 (8 %)
Embedded Multiplier 9-bit elements	0 / 132 (0 %)
Total PLLs	0 / 4 (0 %)

Figure 3.21 : The successful complation of VHDL coding

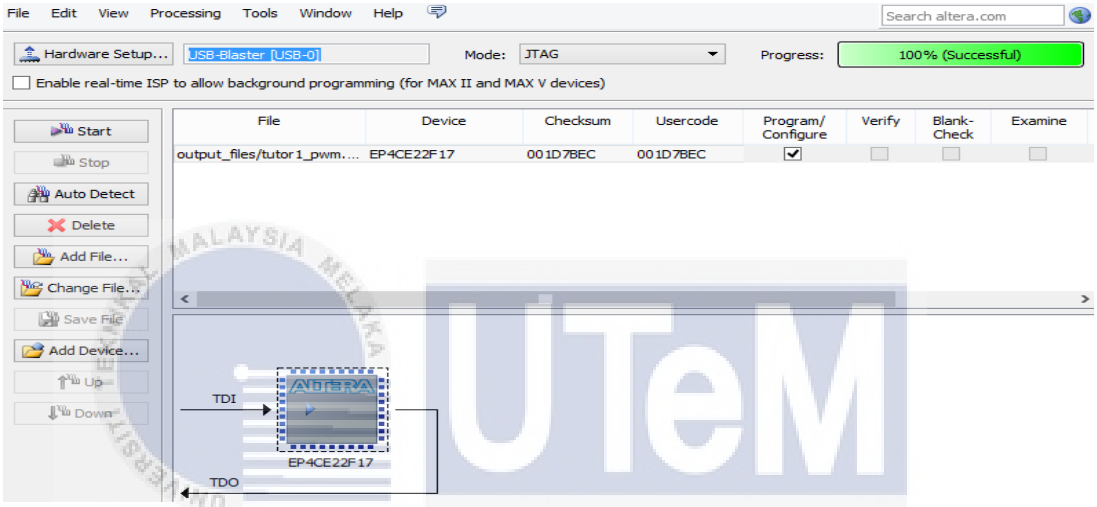
The important step on FPGA is assigning pin, click 'Assignment' on Menu and select 'Pin Planner'. The top view of FPGA is displayed. Click the location of each node to assign the pin according to the particular FPGA datasheet as Appendix D. Pin for each node is as assigned such as the clock select PIN_R8 and the reset select PIN_E1. While other pin are select as figure 3.22. Full compilation is done by clicking button at 'Compile Design' on 'Task' of FPGA layout. Otherwise, it can be done by clicking the 'Start Compilation' at 'Processing' menu. It will compile the whole setting of design. If the design is unsuccessful, thus error and warning appear on compilation report.



Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard
clk	Input	PIN_R8	3	B3_N0	PIN_R8	2.5 V (default)
rst	Input	PIN_E1	1	B1_N0	PIN_E1	2.5 V (default)
salow	Output	PIN_L15	5	B5_N0	PIN_L15	2.5 V (default)
saup	Output	PIN_N16	5	B5_N0	PIN_N16	2.5 V (default)
sblow	Output	PIN_L13	5	B5_N0	PIN_L13	2.5 V (default)
sbup	Output	PIN_P16	5	B5_N0	PIN_P16	2.5 V (default)
sclow	Output	PIN_P14	4	B4_N0	PIN_P14	2.5 V (default)
scup	Output	PIN_N14	5	B5_N0	PIN_N14	2.5 V (default)
<<new node>>						

Figure 3.22 : Assigning the pin at the pin planner

Finally, connect the USB port of FPGA to computer/laptop and select ‘Programmer’ on ‘Tools’ menu. Hence, ‘Programmer’ layout is displays and click ‘Add File’ to choose file from ‘Select Programming File’. Click output_files then select ‘.sof’ file (In this case, ‘tuto1_pwm.sof’ is selected). Therefore, ‘.sof’ is available on ‘Programmer’ layout. Click ‘Hardware Setup’ to select USB-Blaster [USB-0].Then, click ‘Start’ to upload the program into FPGA. Status of successful uploaded the program into FPGA is shown as in figure 3.23.



Hardware Setup... USB-Blaster [USB-0]   Mode: JTAG   Progress: 100% (Successful)

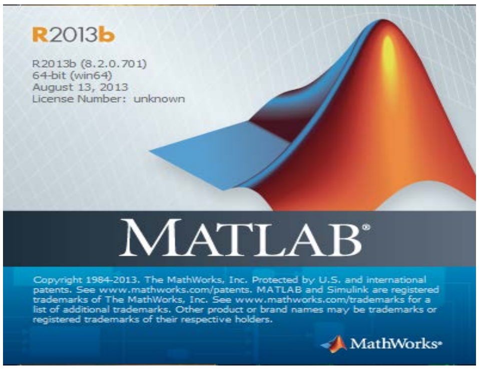
□ Enable real-time ISP to allow background programming (for MAX II and MAX V devices)

File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank-Check	Examine
output_files/tutor1_pwm....	EP4CE22F17	001D7BEC	001D7BEC	☑	□	□	□

Figure 3.23: Status of successful uplođed the program

3.9.2 MATLAB

MATLAB in figure 3.24 is a numerical computation and simulation tools that were efflorescent into a commercial tool with an interface for the numerical function libraries, which basically written in the programming language. Next, the principle in MATLAB is a purely numerical calculation. Basically MATLAB only include a single data structure and all its operations are based. Besides, MATLAB name are standing for MATrix LABoratory. The elementary MATLAB operations can be divided into 5 examples such as arithmetic operations, logical operation, mathematical functions, graphical functions and input/output operations. Simulink in the MATLAB is a tool for simulating dynamic systems with a graphical interface [9]. Before run the program set the end time to ensure the amount of data to be collected. The end time for this project is 0.06s, thus the number of data will equal to 12000 which get from 0.06s divide by 5e-6s which is sampling time in the simulation.



R2013b

R2013b (8.2.0.701)
64-bit (win64)
August 13, 2013
License Number: unknown

MATLAB®

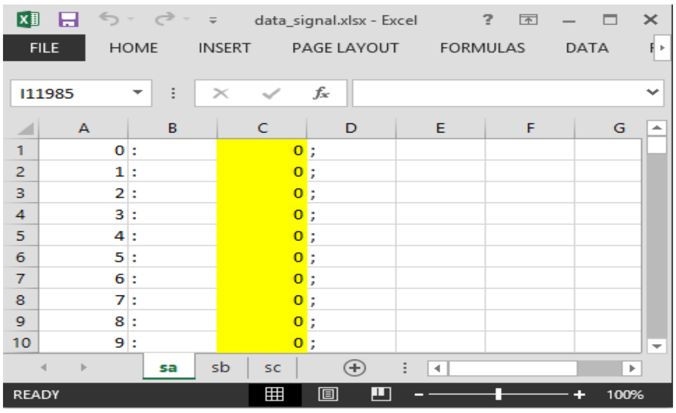
Copyright 1984-2013. The MathWorks, Inc. Protected by U.S. and international patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Figure 3.24: Software Matlab

3.9.3 Microsoft Excel

Excel is a spreadsheet application that allows you to organize data, entire calculations, make decisions, graph data, develop professional-looking reports, organized the data, and access real-time data [10]. There are four part in the Excel such as workbooks and worksheets, charts, tables and Web support. Using this application, it can help to store a lot of data.

Obtain the data from the MATLAB workspace and copied the number of data from 1 to 12001 and paste into a new document of Excel file. In the new document, it is modified by placing the data numbering by sequence (must start from 0 onwards), colon (:), binary number (data obtained from the workspace) and semicolon (;) into column 1, 2 3 and 4 respectively. Save the Excel file into an empty folder, named it as a data_signal because all FPGA's files will be combined into the same folder. The binary data will be used in the tutorial FPGA as figure 3.25.



	A	B	C	D	E	F	G
1	0	:	0	;			
2	1	:	0	;			
3	2	:	0	;			
4	3	:	0	;			
5	4	:	0	;			
6	5	:	0	;			
7	6	:	0	;			
8	7	:	0	;			
9	8	:	0	;			
10	9	:	0	;			

Figure 3.25 : The binary data

CHAPTER 4

RESULT AND DISCUSSION

4.1 Introduction

This chapter presents the process and scope of work to obtain the simulation and the experimental result to show performances of two level inverter utilizing Space Vector Modulation (SVM) technique. The simulation result obtains by the MATLAB/Simulink and Quartus Altera. This chapter also showing and proving the calculation of SVM based on the simulation and the experimental. The comparison waveform between simulation and hardware will be show and illustrated.

4.2 Result analysis between SPWM and SVM

Simulation results performed using MATLAB/ Simulink to obtain the comparison result between Sinusoidal Pulse Width Modulation (SPWM) and SVM. The results from SPWM as figure 4.1 and 4.2 while the result for SVM as figure 4.3 and 4.4. The purpose of this comparison is to show that the SVM method SVM is better than SPWM. This comparison is shown in the form of output voltage and FFT Analysis.

The design specification and simulation specification are given as follows:

- DC link voltage, $V_{dc} = 100\ V$
- triangular carrier frequency, $f_{tri} = 1000\ Hz$
- peak of triangular waveform, $V_{tri,p} = 1\ V$
- fundamental frequency of modulating signal, $f_1 = 50\ Hz$
- simulation sampling time, $t_s = 5\ \mu s$

Figure 4.1 shows the output voltage to the input voltage is 100V and the output voltage is 66.67V.

$$vd = \frac{2}{3}Vdc = \frac{2}{3}(100) = 66.67V \tag{4.1}$$

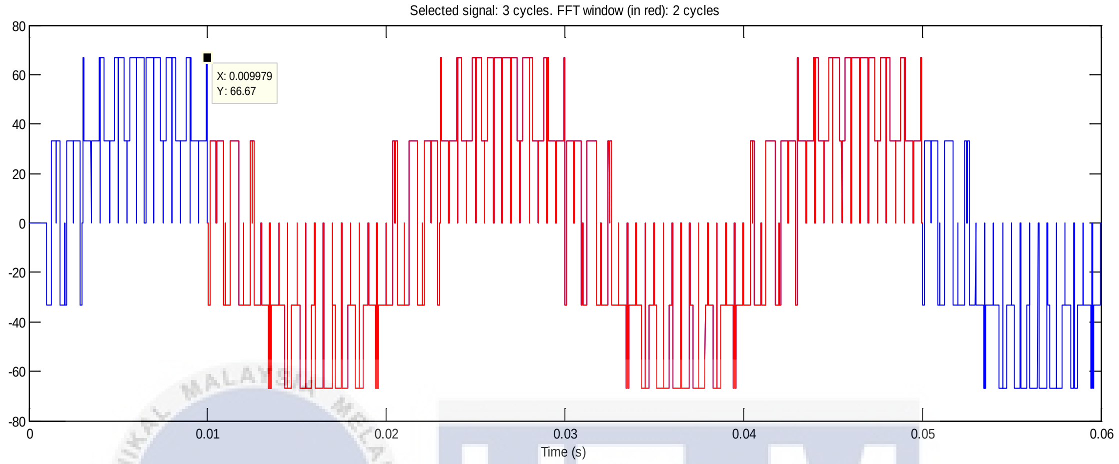


Figure 4.1 : The output voltage for SPWM

The FTT analysis from figure 4.2 can be made based on the frequency spectrum of the output voltage using powergui block from MATLAB.

$$vq = V1 = \frac{V_{DC}}{2} = \frac{100}{2} = 50V \tag{4.2}$$

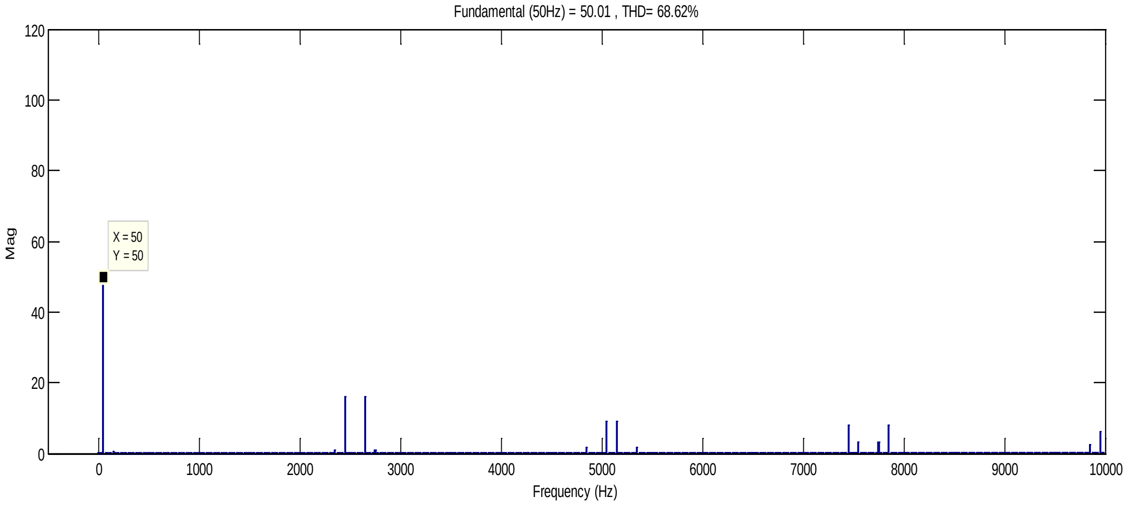


Figure 4.2 : FTT Analysis for SPWM

Figure 4.3 show the v_d with the input voltage is 100V and his can be proved by the calculation of v_d.

$$vd = \frac{2}{3}Vdc = \frac{2}{3}(100) = 66.67V \tag{4.3}$$

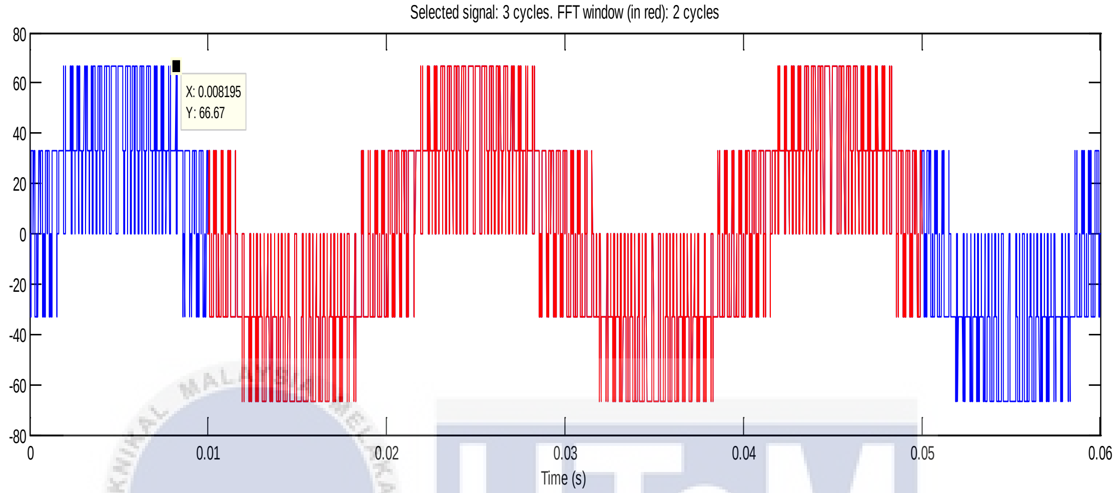


Figure 4.3 : The v_d waveform for SVM

The FTT analysis from figure 4.4 can be made based on the frequency spectrum of the output voltage using powergui block from MATLAB.

$$vq = V1 = \frac{V_{DC}}{\sqrt{3}} = \frac{100}{\sqrt{3}} = 57.735V \tag{4.4}$$

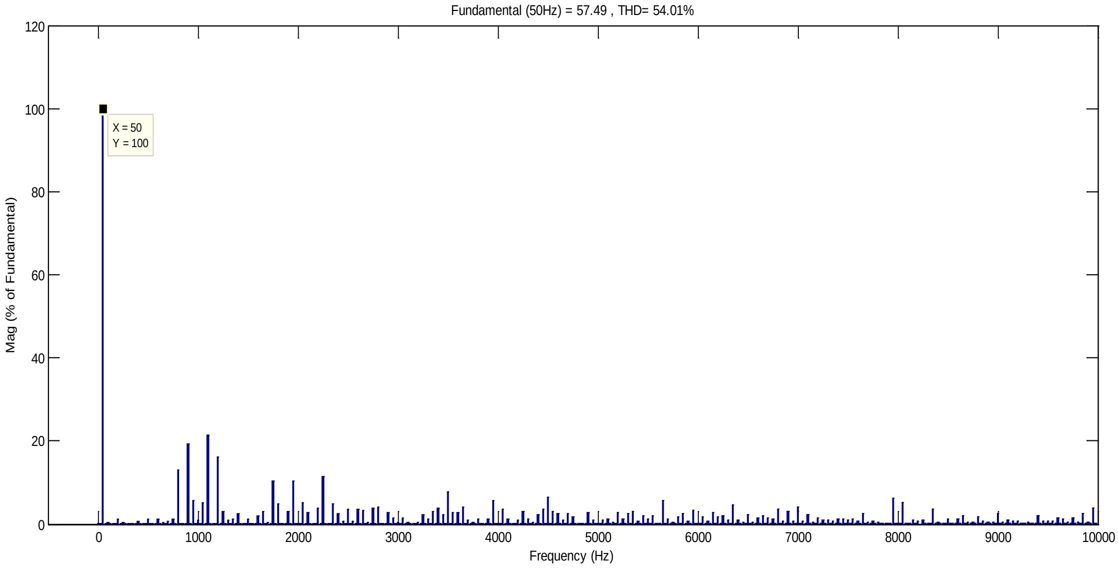


Figure 4.4 : FTT Analysis for SVM

This proving the percentage different between SPWM and SVM.

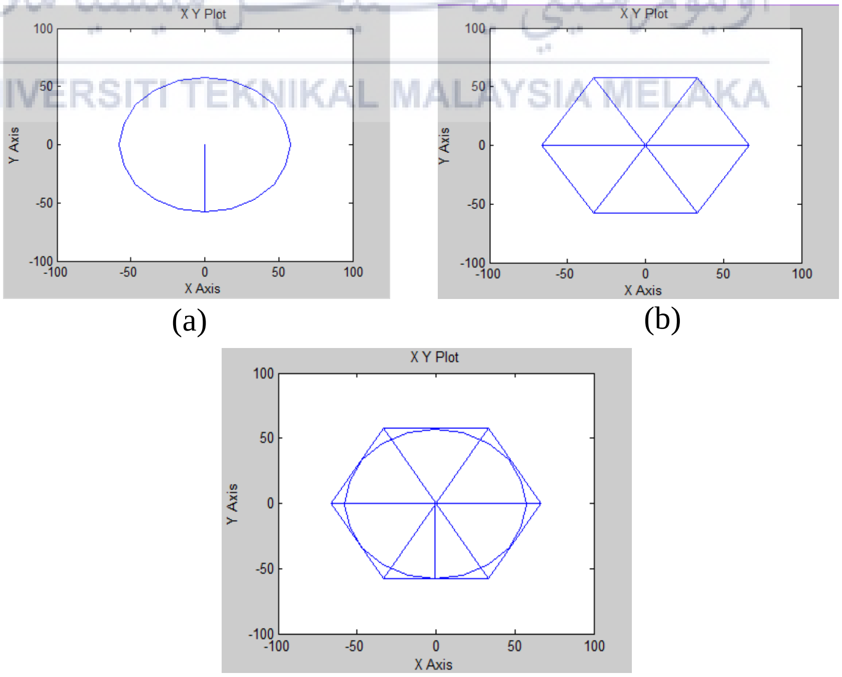
$$\Delta V\% = \frac{\left(\frac{V_{DC}}{\sqrt{3}}\right)-\left(\frac{V_{DC}}{2}\right)}{\left(\frac{V_{DC}}{2}\right)} \times 100\% = \frac{\left(\frac{100}{\sqrt{3}}\right)-\left(\frac{100}{2}\right)}{\left(\frac{100}{2}\right)} \times 100\% = 15.47\% \qquad (4.5)$$

4.3 Result of switching states

Results of switching states showing the approach to prove the calculation of simulation based on the given parameters. The parameters are given as follows:

- DC link voltage, $V_{dc} = 100\ V$
- Triangular carrier frequency, $f_{tri} = 1000\ Hz$
- Simulation sampling time, $t_s = 1\ \mu s$
- References voltage vector, $v^* = 45\angle -130°$

Figure 4.5 shows that the d and q-axis voltage vd. The value vd can be observed by x-axis of the graph. Refer to the figure 4.5 (b) the value of vd is 66.67V while vq can be taken from y-axis of graph which is 57.735V as states in figure 4.5(a).



(a)

(b)

(c)

Figure 4.5 : Switching status (a) d-axis voltage, vd (b) q-axis voltage, vq (c) combination of vd and vq

Figure 4.6 shows the triangle voltage supplied with 1volt. The frequency of triangle voltage is 1kHz.

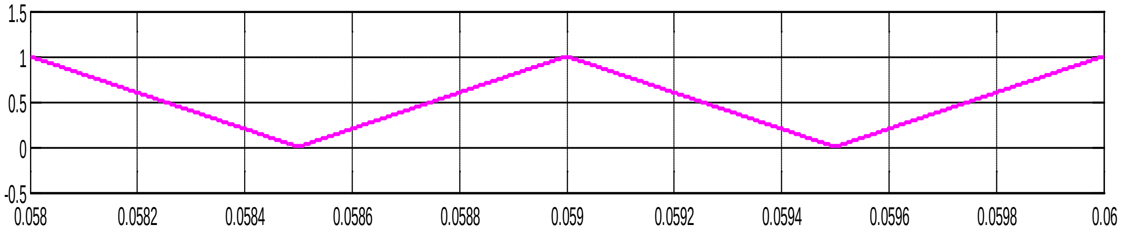


Figure 4.6 : Voltage triangle, V_{tri} waveform

Figure 4.7 is the illustration for the position references voltage vector, v^* based on the parameter. The given angle of the references voltage vector, v^* from the parameter is -130, which can get the 10° for angle theta sector, θ_{sec}. This is because value angle of the references voltage vector, v^* subtract with the angle from the theta sector VI and V which is 60°. Besides, Figure 4.8 showing the general representation for sector IV.

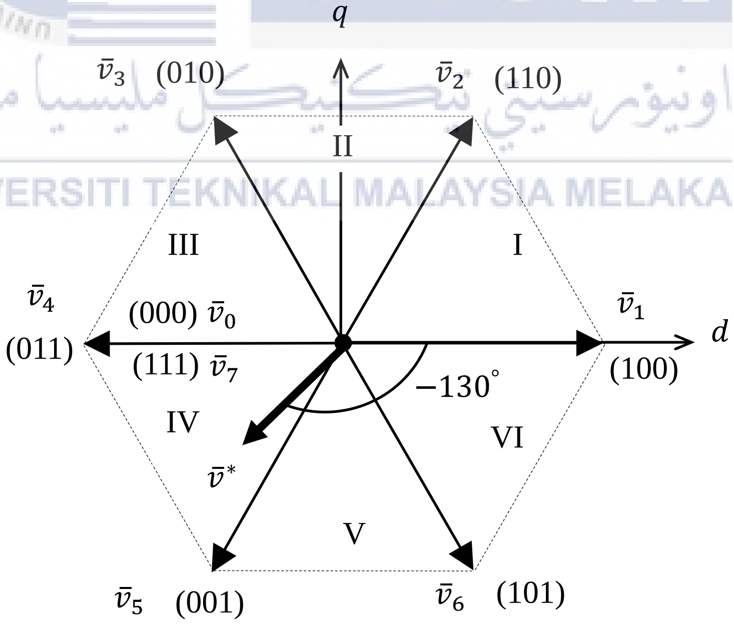


Figure 4.7 : The position references voltage vector, v^*

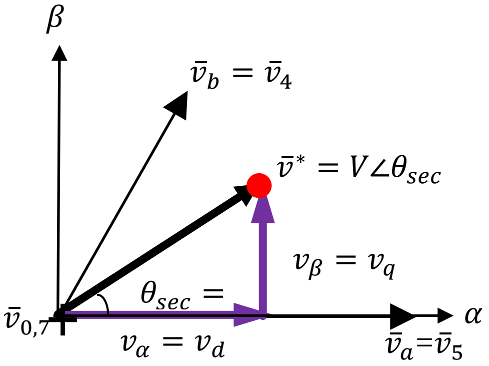


Figure 4.8 : General representation for sector IV

The equation of (4.6), (4.7), (4.8), (4.9) and (4.10) is used to prove the result simulation in figure 4.9.

$$v_\alpha = v_d = Vcos\theta_{sec} = 45\cos(10^{\circ}) = 44.3163V \tag{4.6}$$

$$v_\beta = v_q = Vsin\theta_{sec} = 45\sin(10^{\circ}) = 7.8142V \tag{4.7}$$

$$t_b = \sqrt{3}.T\frac{v_\beta}{V_{dc}} = \sqrt{3}(\frac{1}{1000})\frac{7.8142}{100} = 135.346\mu s \tag{4.8}$$

$$t_a = \frac{3}{2}\frac{T}{V_{dc}}\left[v_\alpha - \frac{v_\beta}{\sqrt{3}}\right] = \frac{3}{2}\frac{({}^{1}/{}_{1000})}{100}\left[44.3163 - \frac{7.8142}{\sqrt{3}}\right] = 597.07\mu s \tag{4.9}$$

$$t_z = T - (t_a + t_b) = \left(\frac{1}{1000}\right) - (597.07\mu + 135.346\mu) = 267.584\mu s \tag{4.10}$$

Figure 4.9 based on the switching status waveform of S_a, S_b and S_c in the sector IV.

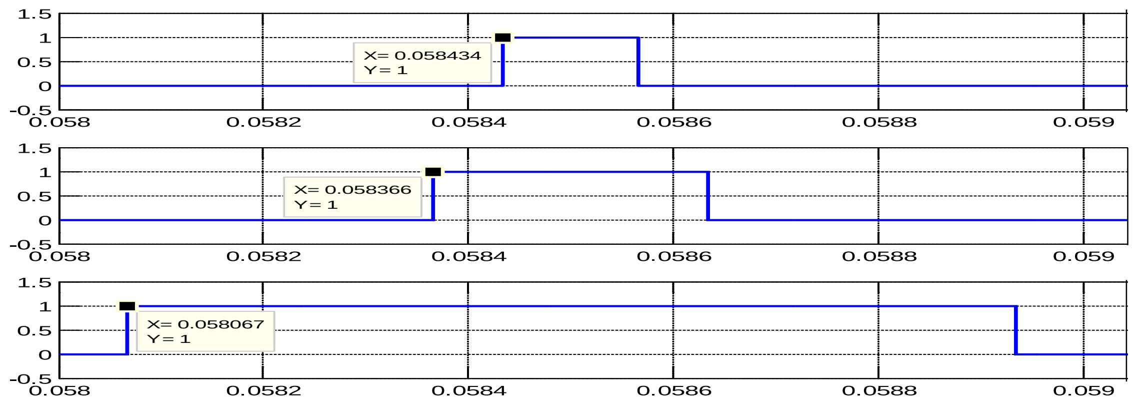


Figure 4.9 : Switching status waveform for S_a, S_b and S_c respectively

4.4 Performance Analysis of AC output voltage at different switching frequency using SVPWM

There are two types of frequency that have been used in this simulation and experimental to get the accurate value which is 1000 Hz and 2550Hz. Table 4.1 is to show the exact value of FFT Analysis based on the formula and calculation. Firstly, the value of DC link voltage, V_{dc} which is 100V is multiplied by the modulation index, M_i and divide with the square root of three ($\sqrt{3}$) for simulation. While, for hardware is the same method with the simulation, but it need to divide with the square root of two ($\sqrt{2}$) because the results that appear on the oscilloscope is in the form of Root Mean Square (RMS).

Table 4.1: Comparison result between simulation and hardware based on the formula

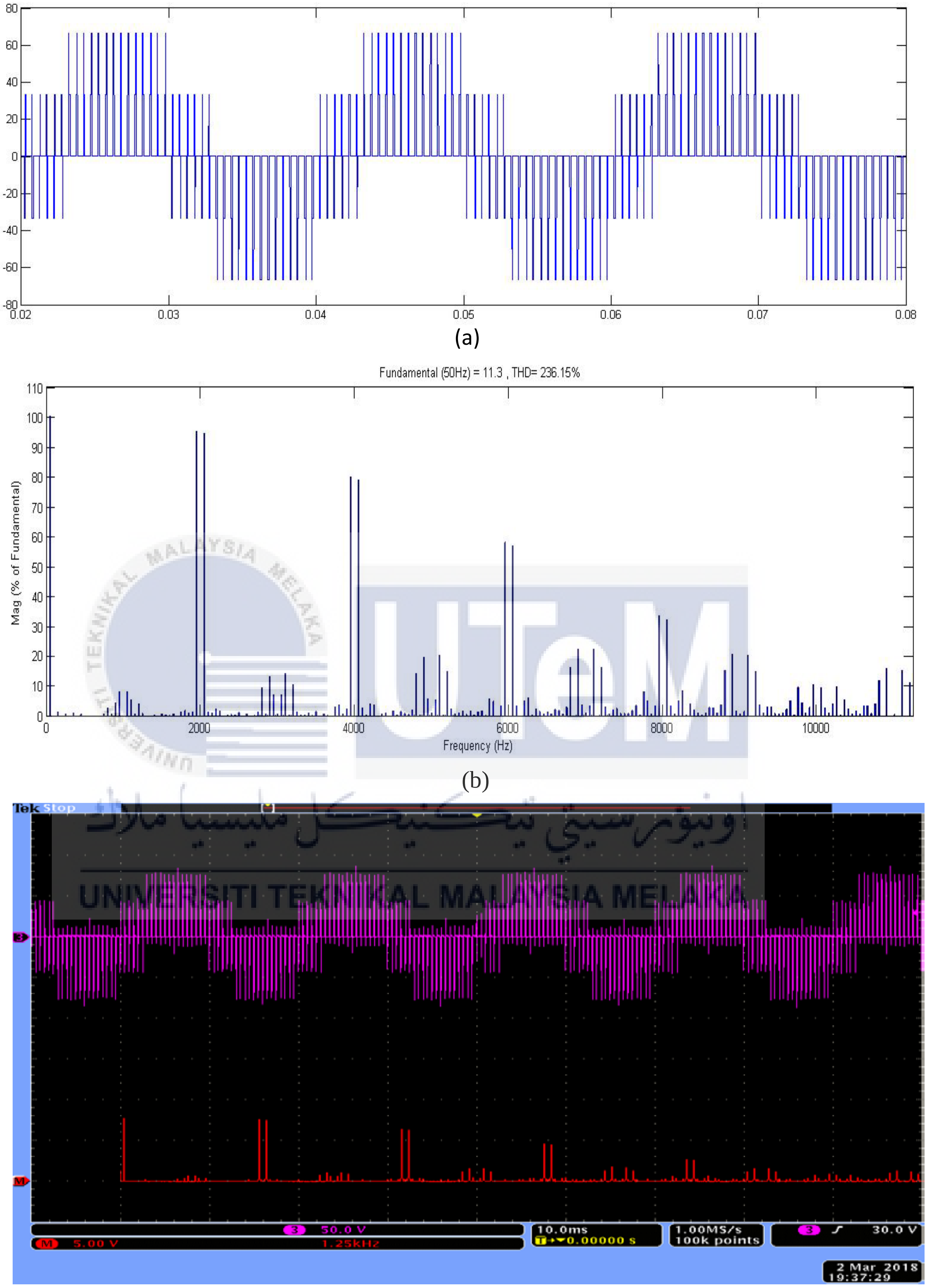
Types M_i, Modulation index	Simulation, $M_i V_{dc}/\sqrt{3}$	Hardware, $(M_i V_{dc}/\sqrt{3})\ /\sqrt{2}$
0.2	11.54701	8.16497
0.4	23.09401	16.3299
0.6	34.64102	24.4949
0.8	46.18802	32.65986
1.0	57.73503	40.82482

4.4.1 Triangular carrier frequency, f_{tri} for 1000Hz

Figure 4.10, figure 4.11, figure 4.12, figure 4.13, and figure 4.14 are generated using $1000Hz$ of triangular carrier frequency, f_{tri}. All this figure contains 3 parts which is a, b and c. Result in figure a and b is obtained from MATLAB simulation which is a from M file and b from FFT Analysis. While the figure in part c shows the result of hardware from the oscilloscope, which is indicated the output voltage on the upper part and the FFT Analysis on the lower part of the oscilloscope. As for that, the result from the simulation part are compare with the hardware to get the precise results. This comparison is proved using formula and calculation approach.

Table 4.2 : Result simulation and hardware of FFT Analysis for 1000Hz

Types M_i, Modulation index	FFT Analysis
0.2	11.30
0.4	22.97
0.6	34.33
0.8	45.99
1.0	57.26

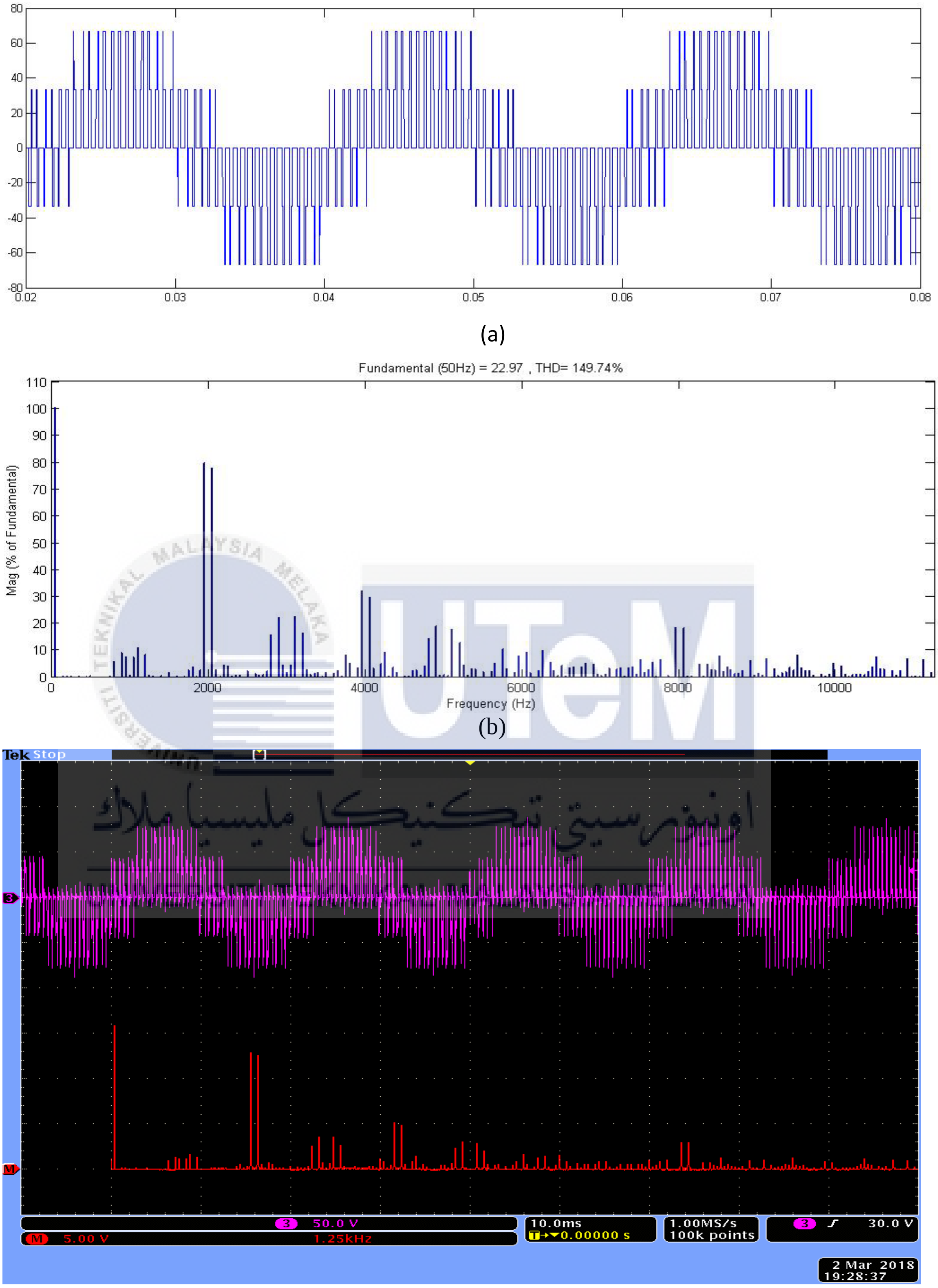


(a)

(b)

(c)

Figure 4.10 : The comparison of simulation and experimental results for amplitude 0.2 with frequency of 1kHz (a) Signal waveform from the simulation (b) Carrier signal from the FFT Analysis from simulation (c) the experimental result from the scope.

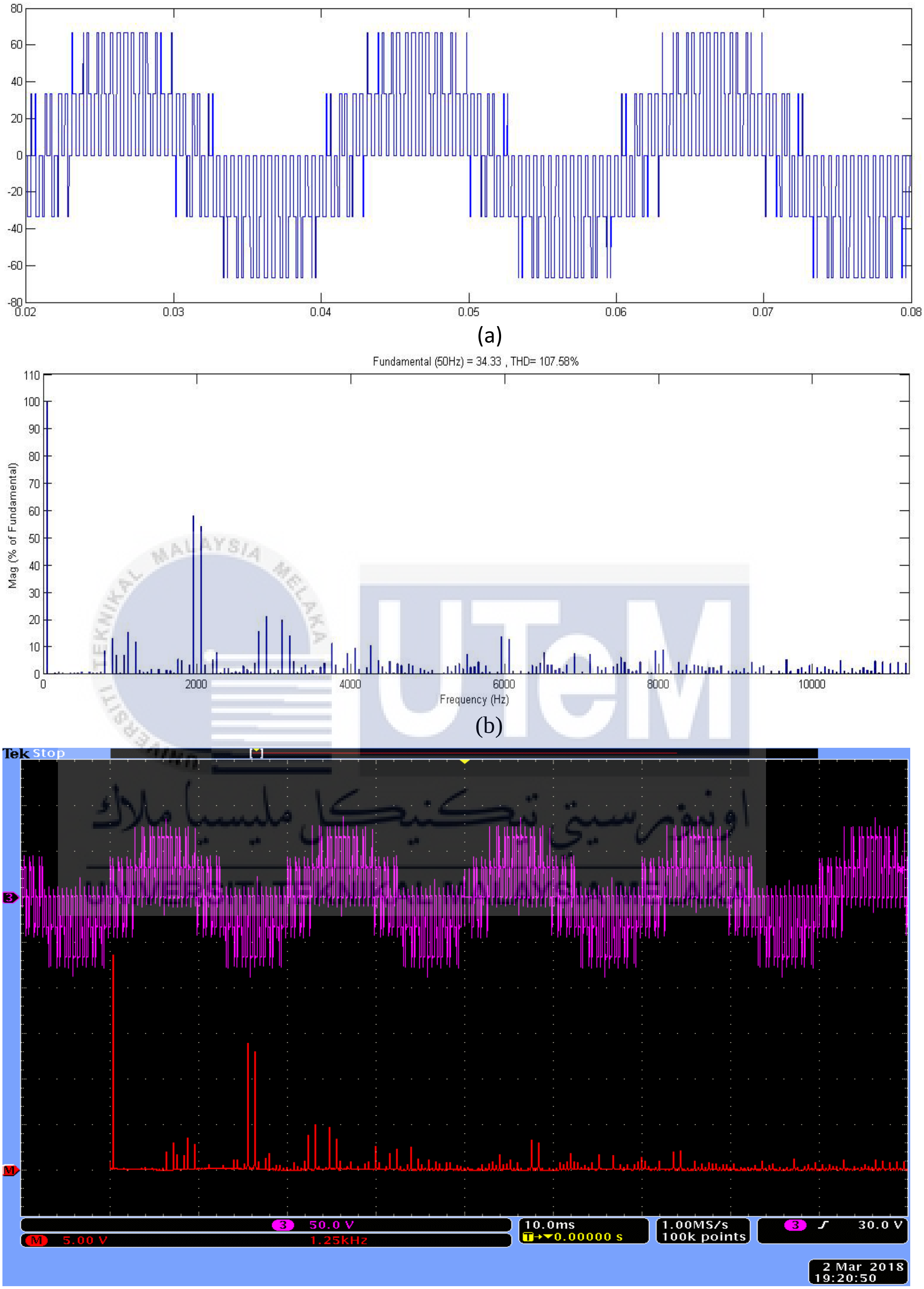


(a)

(b)

(c)

Figure 4.11 : The comparison of simulation and experimental results for amplitude 0.4 with frequency of 1kHz (a) Signal waveform from the simulation (b) Carrier signal from the FFT Analysis from simulation (c) the experimental result from the scope.

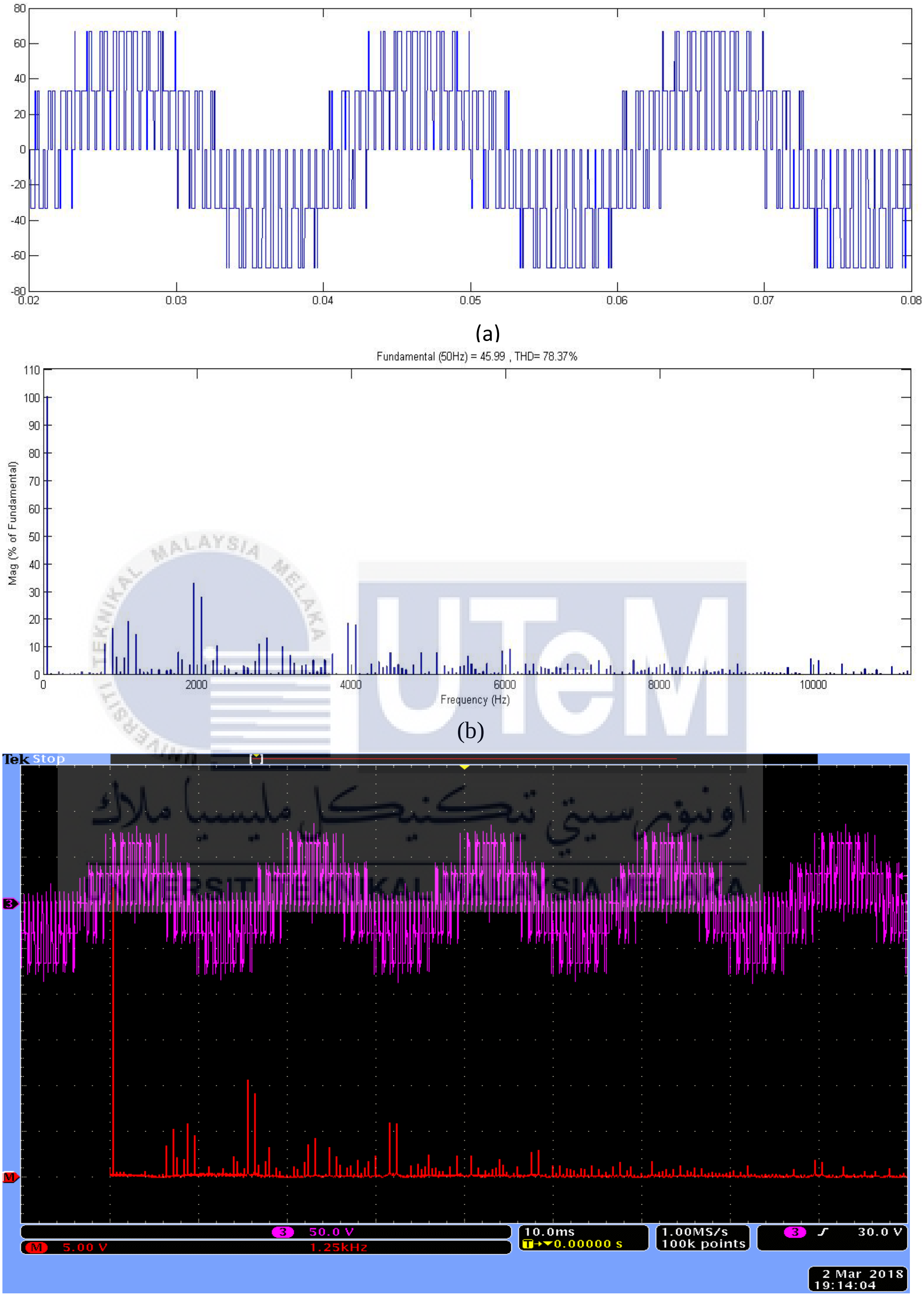


(a)

(b)

(c)

Figure 4.12: The comparison of simulation and experimental results for amplitude 0.6 with frequency of 1kHz (a) Signal waveform from the simulation (b) Carrier signal from the FFT Analysis from simulation (c) the experimental result from the scope.

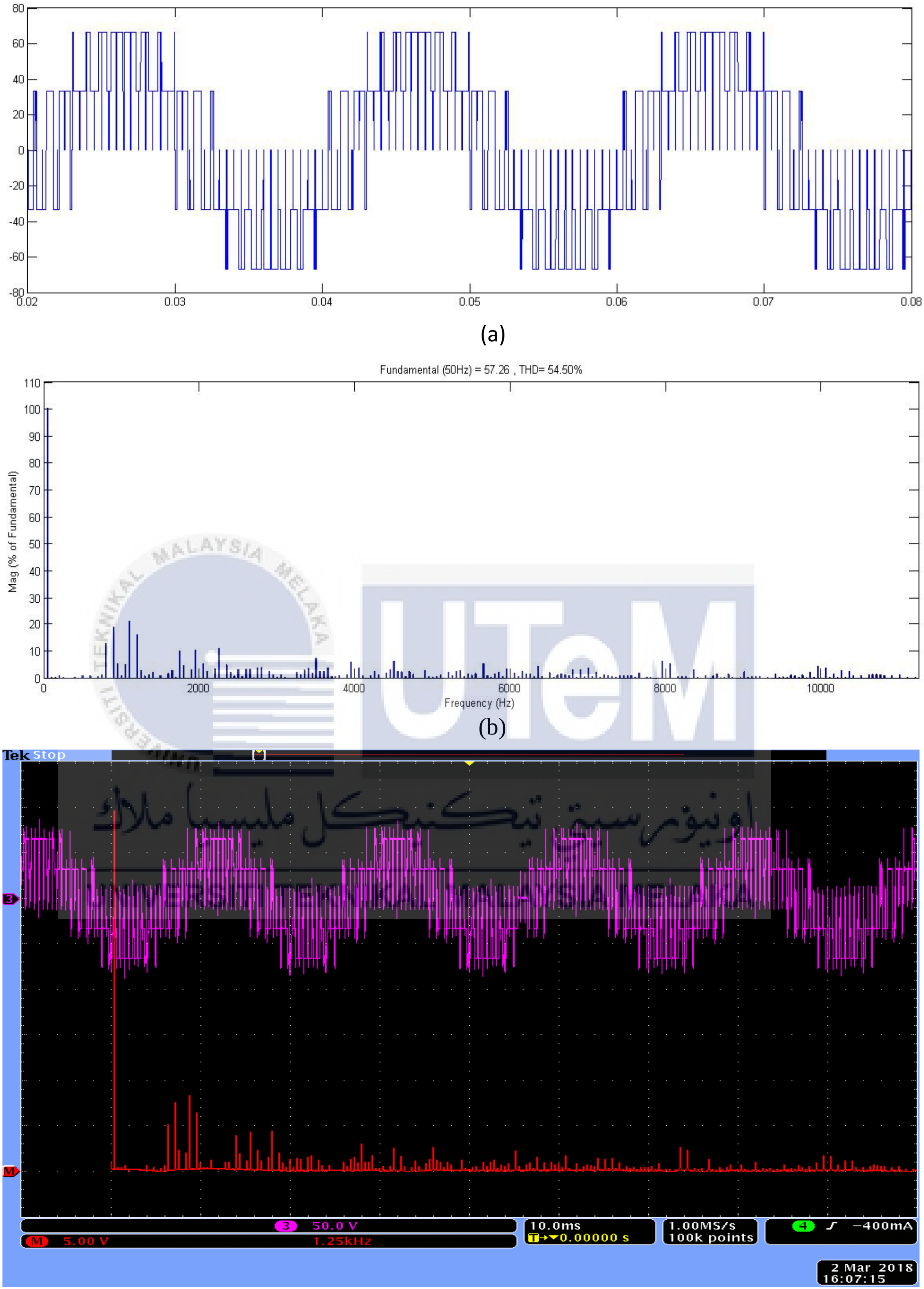


(a)

(b)

(c)

Figure 4.13 : The comparison of simulation and experimental results for amplitude 0.8 with frequency of 1kHz (a) Signal waveform from the simulation (b) Carrier signal from the FFT Analysis from simulation (c) the experimental result from the scope.



(a)

(b)

(c)

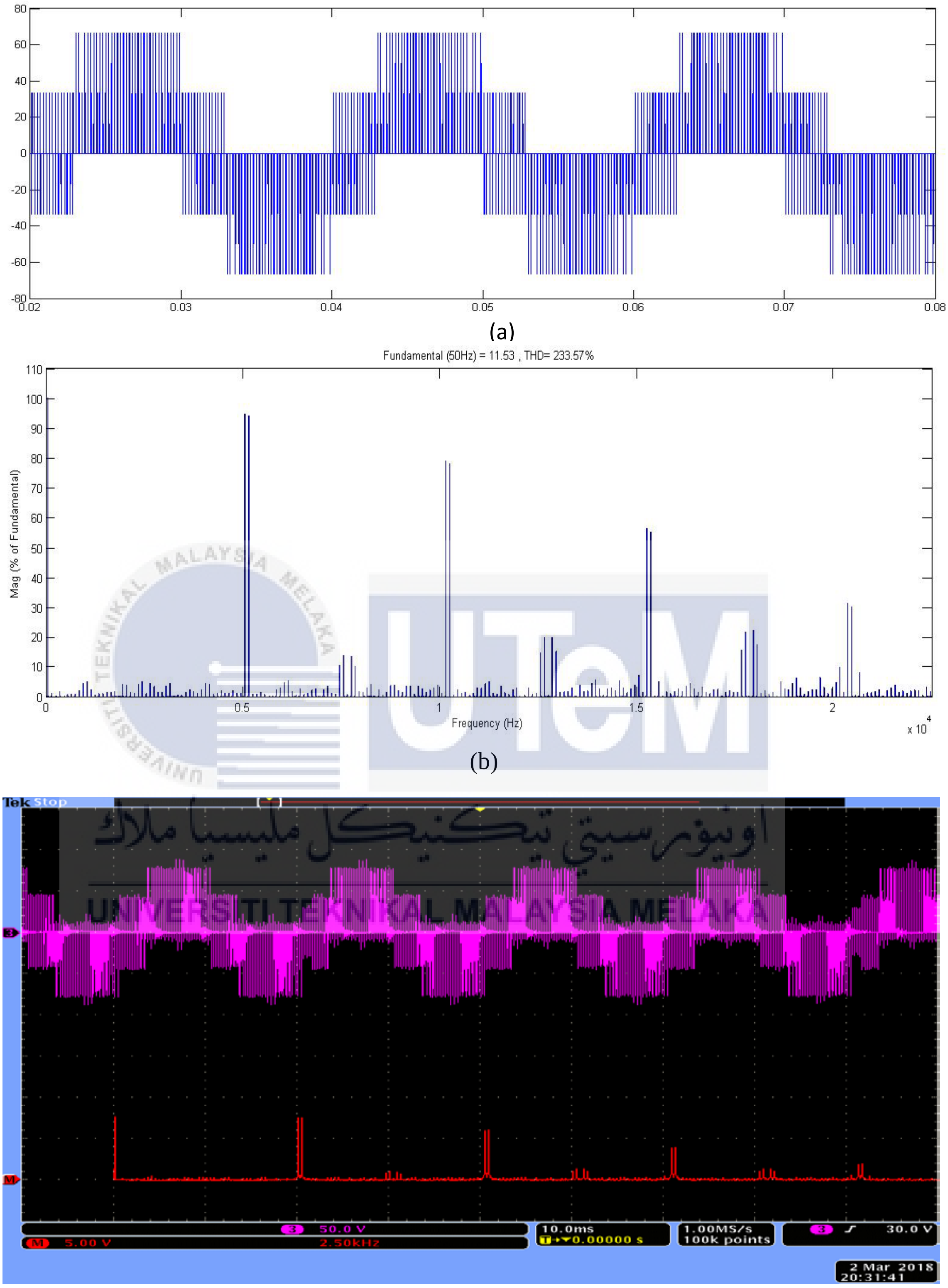
Figure 4.14 : The comparison of simulation and experimental results for amplitude 1.0 with frequency of 1kHz (a) Signal waveform from the simulation (b) Carrier signal from the FFT Analysis from simulation (c) the experimental result from the scope.

4.4.2 Triangular carrier frequency, f_{tri} for 2550Hz

Figure 4.15, figure 4.16, figure 4.17, figure 4.18, and figure 4.19 are generated using $2550Hz$ of triangular carrier frequency, f_{tri}. All this figure contains 3 parts which is a, b and c. Result in figure a and b is obtained from MATLAB simulation which is a from M file and b from FFT Analysis. While the figure in part c shows the result of hardware from the oscilloscope, which is indicated the output voltage on the upper part and the FFT Analysis on the lower part of the oscilloscope. As for that, the result from the simulation part are compare with the hardware to get the precise results. This comparison is proved using formula and calculation approach.

Table 4.3 : Result simulation and hardware of FFT Analysis for $2550Hz$

Types M_i, Modulation index	FFT Analysis
0.2	11.53
0.4	23.12
0.6	34.75
0.8	46.38
1.0	57.79

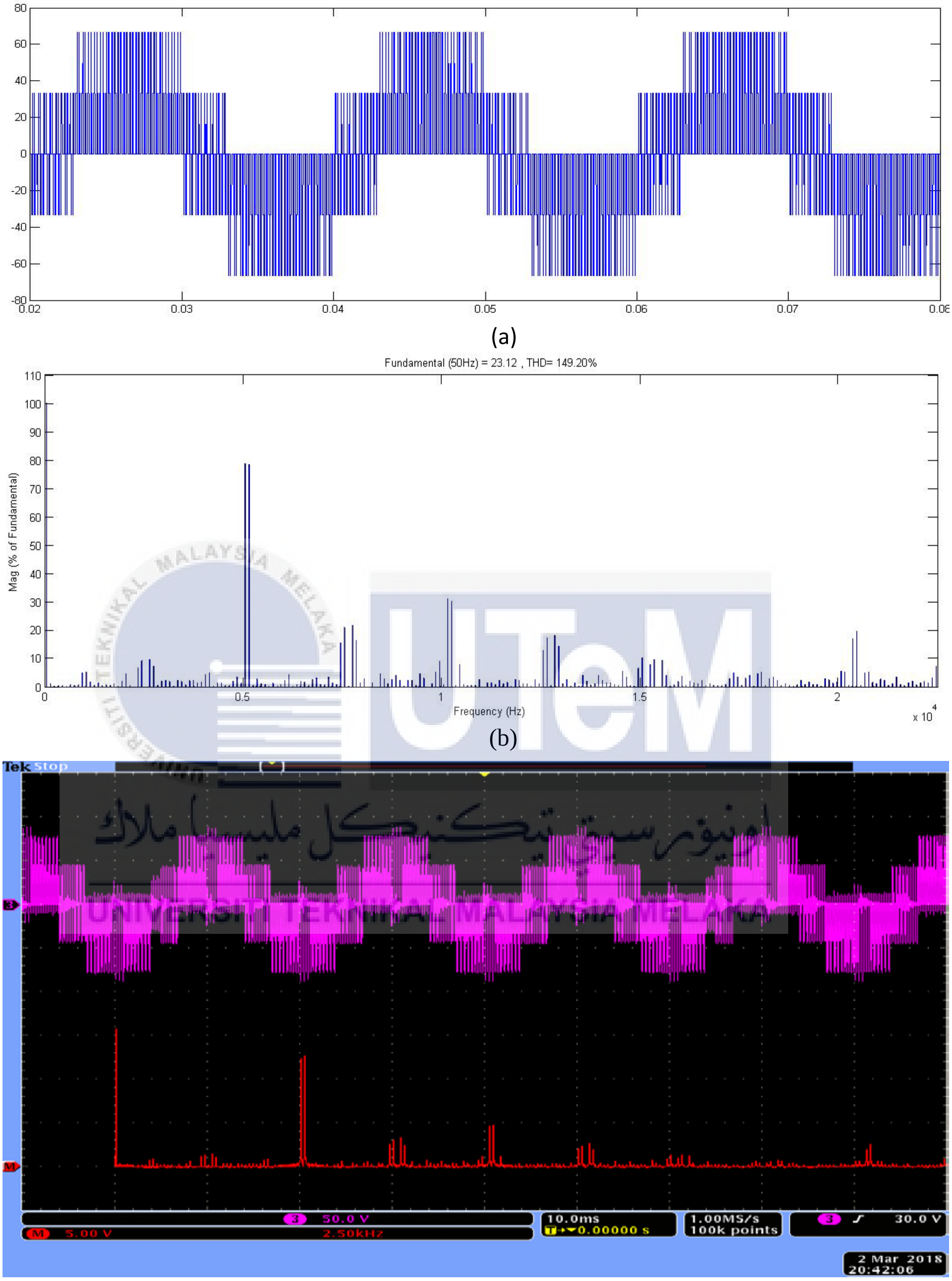


(a)

(b)

(c)

Figure 4.15 : The comparison of simulation and experimental results for amplitude 0.2 with frequency of 2.55kHz (a) Signal waveform from the simulation (b) Carrier signal from the FFT Analysis from simulation (c) the experimental result from the scope.

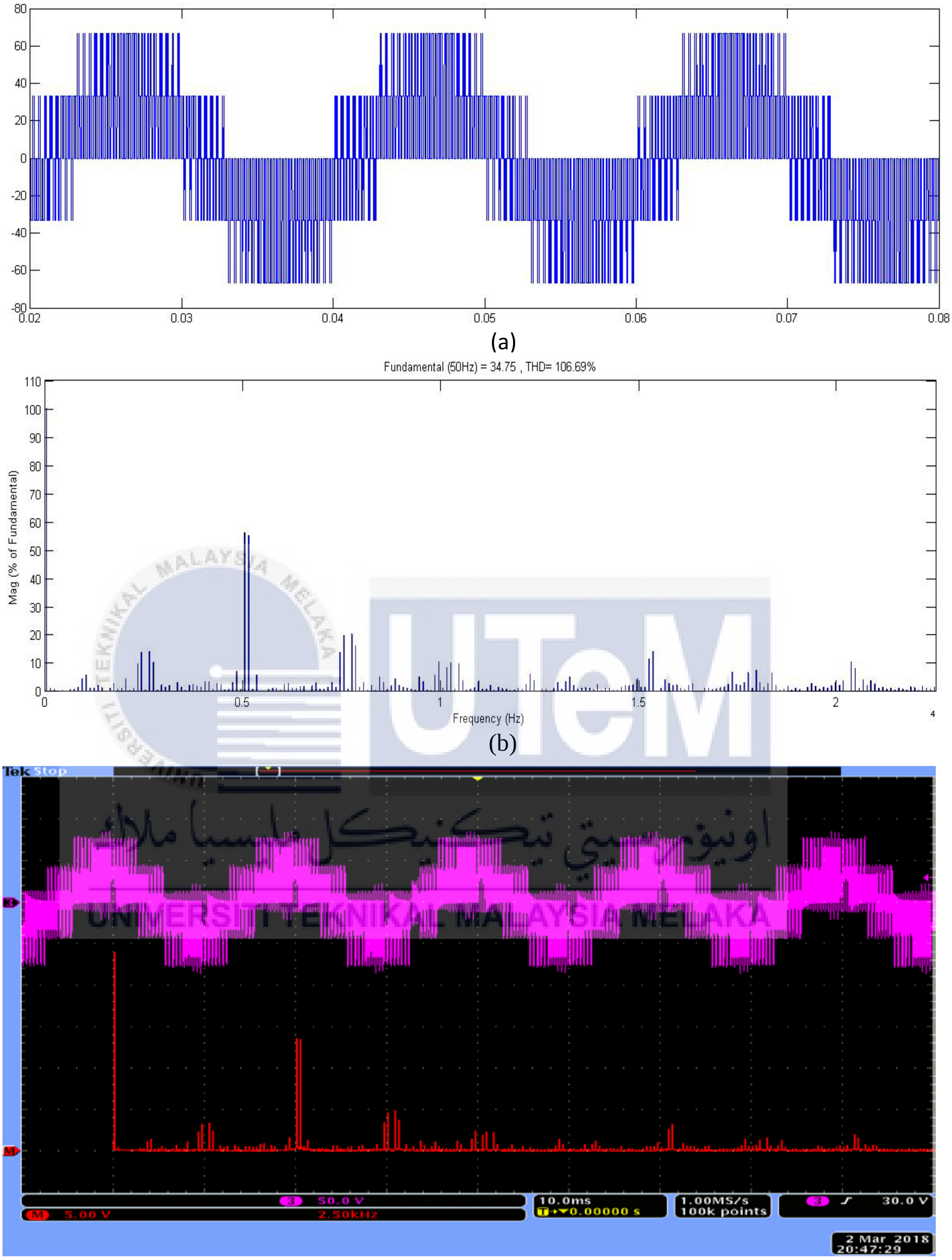


(a)

(b)

(c)

Figure 4.16 : The comparison of simulation and experimental results for amplitude 0.4 with frequency of 2.55kHz (a) Signal waveform from the simulation (b) Carrier signal from the FFT Analysis from simulation (c) the experimental result from the scope.

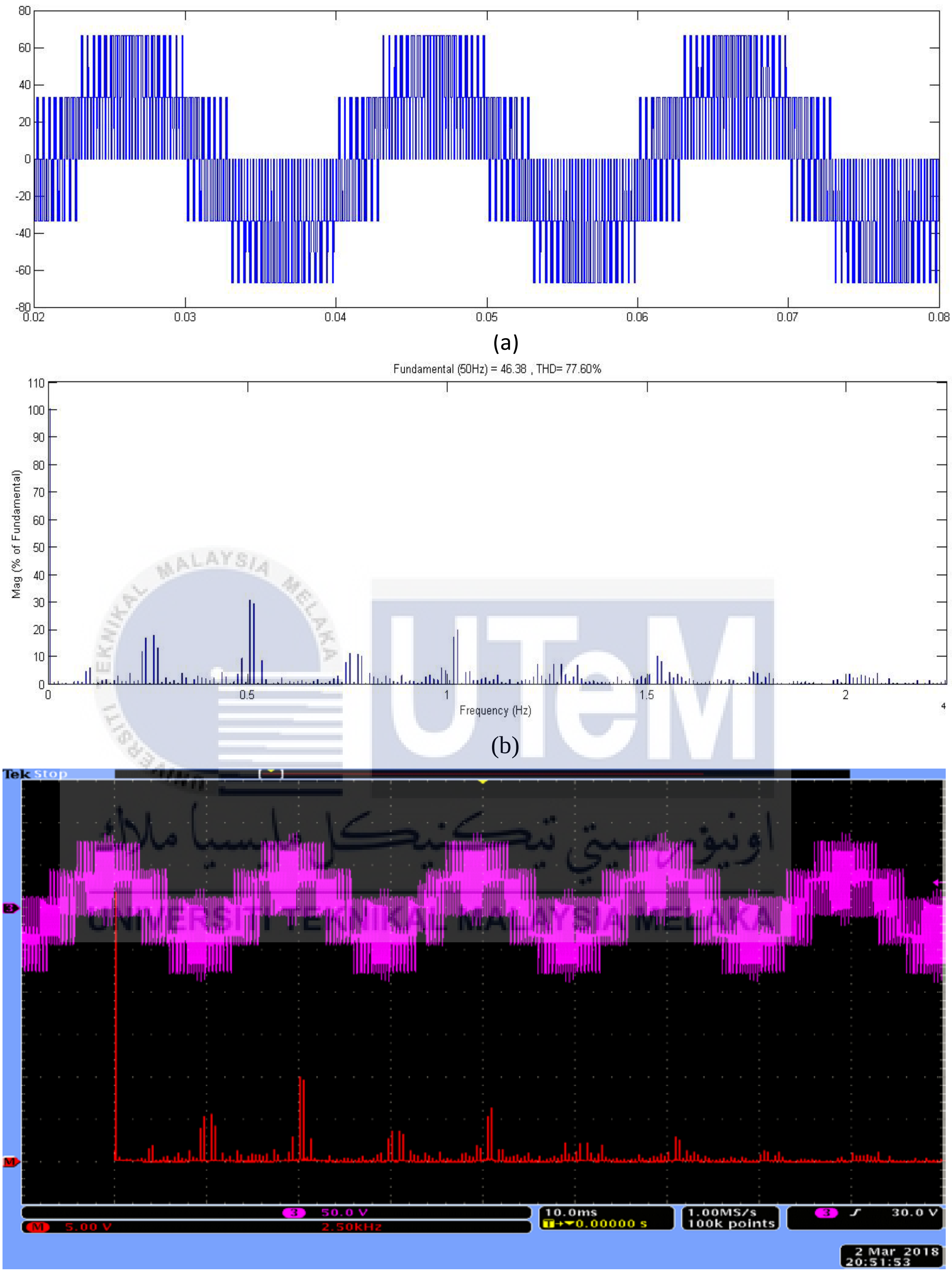


(a)

(b)

(c)

Figure 4.17 : The comparison of simulation and experimental results for amplitude 0.6 with frequency of 2.55kHz (a) Signal waveform from the simulation (b) Carrier signal from the FFT Analysis from simulation (c) the experimental result from the scope.

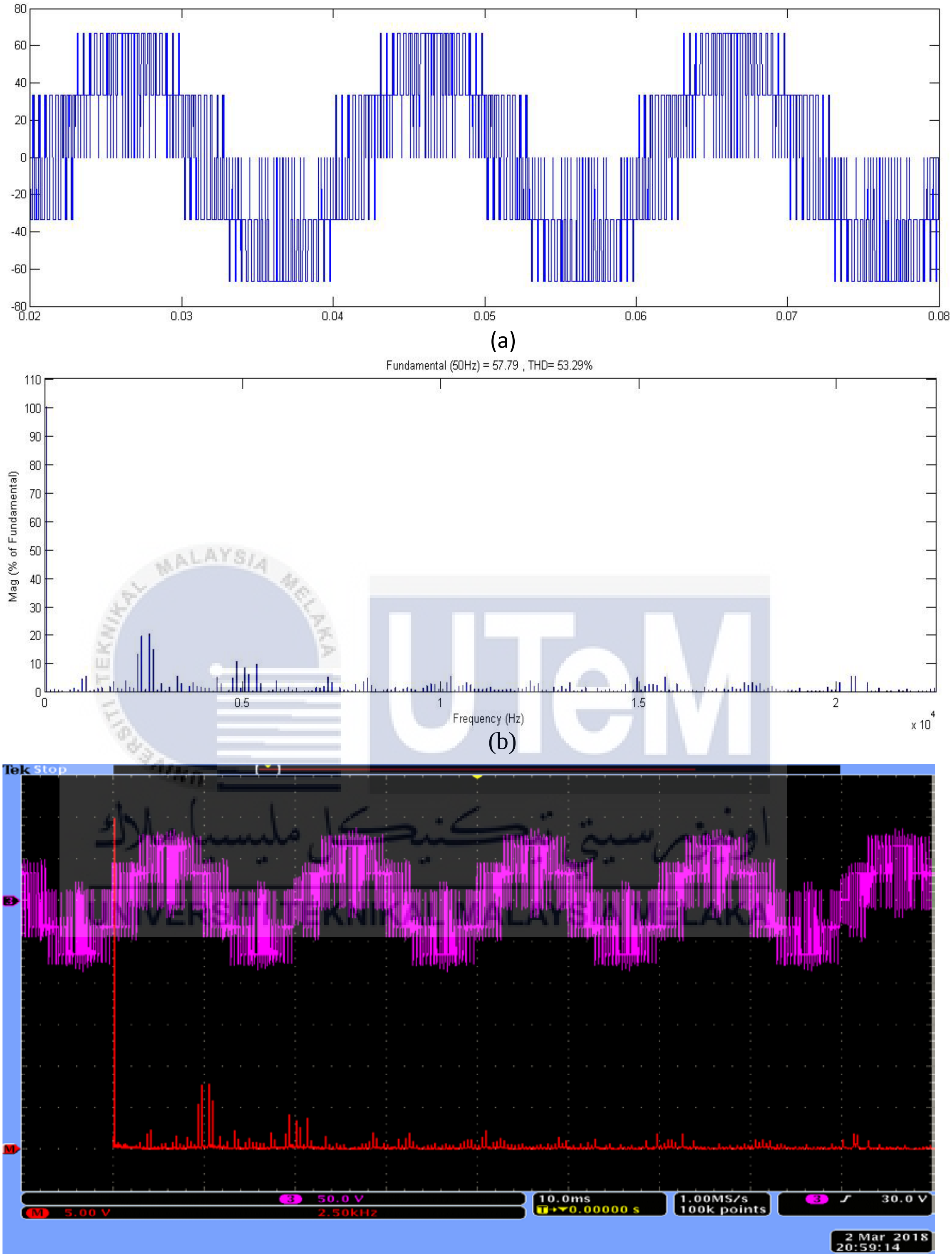


(a)

(b)

(c)

Figure 4.18 : The comparison of simulation and experimental results for amplitude 0.8 with frequency of 2.55kHz (a) Signal waveform from the simulation (b) Carrier signal from the FFT Analysis from simulation (c) the experimental result from the scope.



(a)

(b)

(c)

Figure 4.19 : The comparison of simulation and experimental results for amplitude 1.0with frequency of 2.55kHz (a) Signal waveform from the simulation (b) Carrier signal from the FFT Analysis from simulation (c) the experimental result from the scope.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Introduction

This last chapter will summarize and conclude for this chapter with conclusion and recommendation. Besides, this chapter also has a recommendation for a better design to improved this project for future work.

5.1 Conclusions

Designing a space vector modulator using FPGA controller to produce appropriate switching states, according to the input of $v_d$and v_q. This modulator is suitable for any type of AC motor controls. The design of SVM modulator through simulation and experiment while comparing the resulting simulation for Sinusoidal Pulse Width Modulation (SPWM) and Space Vector Modulation (SVM). The result simulation has been proved that the SVM has better performance compared to the SPWM. This is because the percentage different SVM and SPWM is 15.47%. Besides, the SPWM techniques are inflexible control because difficult to control three-phase quantities such as amplitudes, frequencies and phase angle of three-phase voltage compared to the SVM where it only need to focus at the voltage references and switching states. The design of the space vector modulator using MATLAB/Simulation has successfully produced an appropriate switching state, according to the inputs of d- and q- axis compounds of space vector which is though the output voltage and the FFT Analysis, it is proven that the SVM is the best technique used to control switching for three-phase Voltage Source Inverter (VSI). SVM can operate in high frequency and high voltage which is suitable for the industries that mostly used the three phase inverter. The result is precise and detailed because it can operate in a nanosecond.

5.2 Recommendations

The proposed hardware set-up for future work is adding the battery stored beside the inverter which is will be placed inside the inverter box. The reason for adding the battery stored beside the inverter as a backup plan when the power is shut down or no electrical supply from the supplier such as Tenaga Nasional Berhad (TNB) and Sarawak Energy Berhad. Then, when the supply are discontinuous the inverter still can supply the energy from a battery which has been stored. As for that, the load or user still can receive the power even though the supply is discontinuous. The battery stored and the inverter can join as a one item. Besides, the inverter is functioning to convert the Direct Current (DC) to Alternating Current (AC). As a conclusion, the inverter and battery stored can give the benefit to the users. Figure 5.1 will showing the area for placing the battery store.

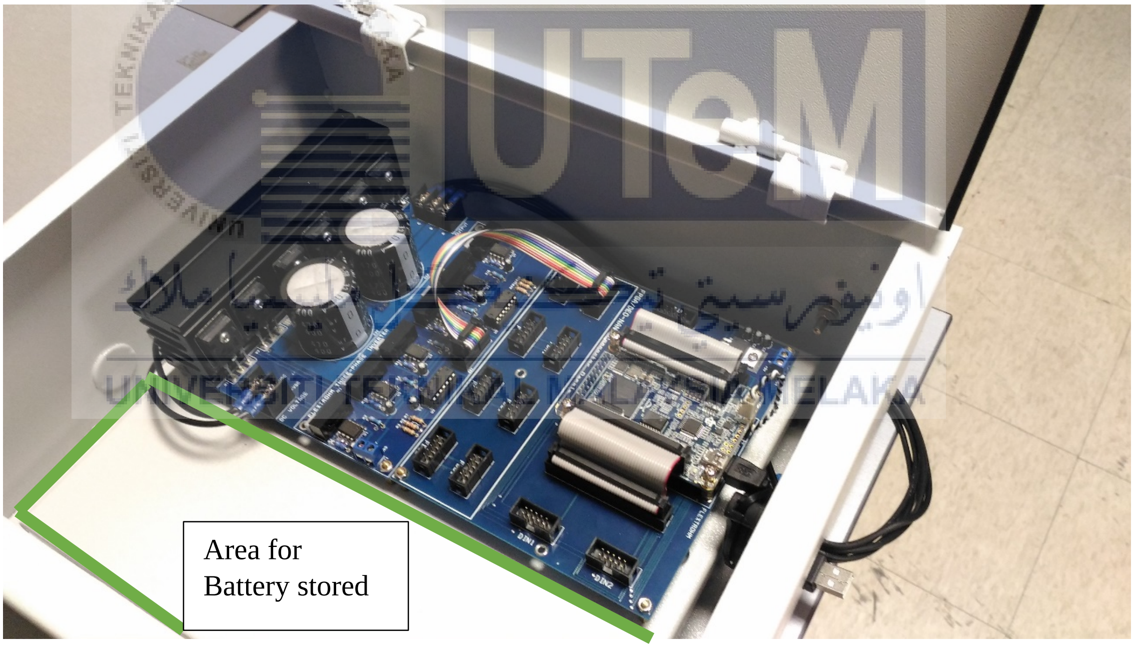


Figure 5.1 : Proposed for future works

REFERENCES

[1] M. Ahmad, *High Performance AC Drives*, vol. 1. 2010.

[2] A. VanderMeulen and J. Maurin, “Current source inverter vs . Voltage source inverter topology,” *Eaton*, no. June, pp. 1–8, 2014.

[3] A. Jidin and T. Sutikno, “MATLAB/SIMULINK based analysis of voltage source inverter with space vector modulation,”,” *TELKOMNIKA Indones. J. …*, no. 1, pp. 23–30, 2009.

[4] J. G. . Haitham Abu_Rub, Atif Iqbal, *HIGH PERFORMANCE CONTROL OF AC DRIVES WITH MATLAB / SIMULINK MODELS HIGH PERFORMANCE CONTROL OF AC DRIVES WITH MATLAB / SIMULINK*. 2012.

[5] L. Wanhammar, *DSP Integrated Circuits*. 1999.

[6] M. Parker, “FPGA versus DSP design reliability and maintenance,” *Design*, no. May, pp. 1–4, 2010.

[7] D. Lee, “Design and Implementation of Three-Phase Inverters Using a TMS320F2812 Digital Signal Processor APPROVED BY SUPERVISING COMMITTEE : Santoso , Surya Grady , W . Mack,” 2009.

[8] Altera Corporation, “Cyclone Device Handbook , Volume 1,” *Memory*, vol. 1, 2008.

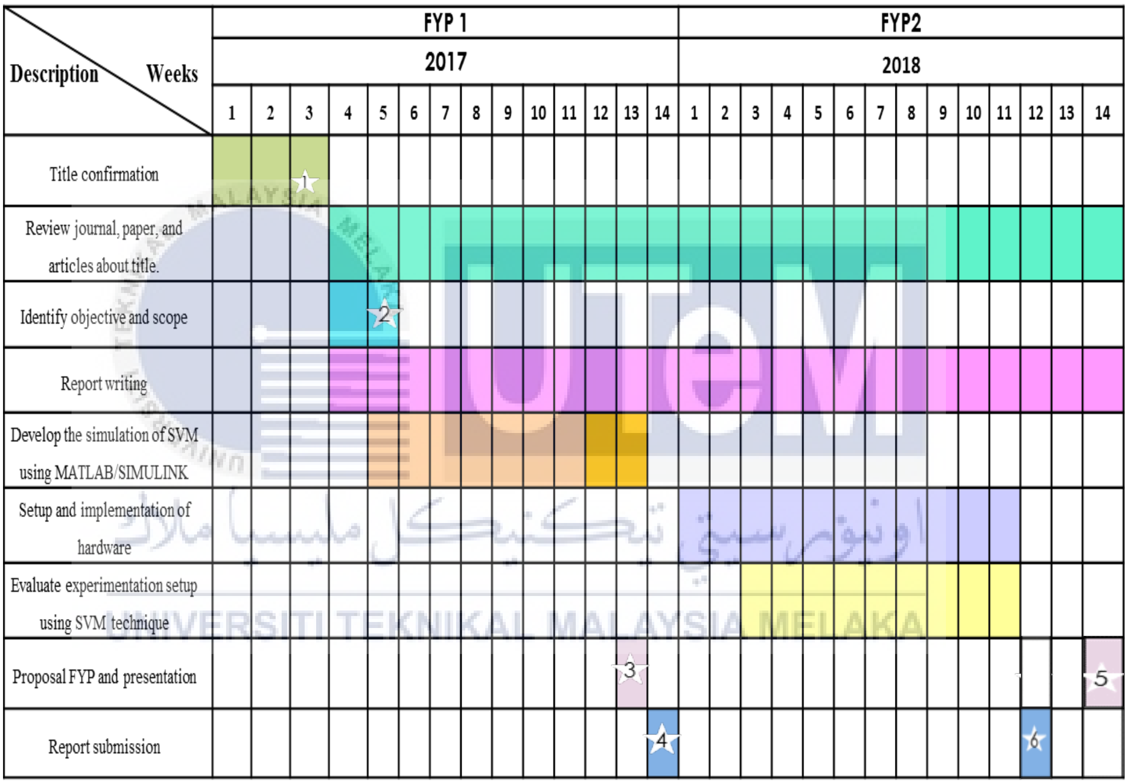
[9] O. Beucher and M. Weeks, *I NTRODUCTION TO MATLAB ® & SIMULINK Approach, A Project*. 2008.

[10] Steven M. Freund, Mali B. Jones, and Joy L. Starks, “Excel 2013 Complete,” 2014.

APPENDIX A

GHANTT CHART AND MILESTONE

This ghantt chart and milestone showing the process and journey to complete the final year project 1 and 2. Each semester contain fourteen weeks.



Description / Weeks	FYP 1														FYP2													
	2017														2018													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Title confirmation	■	■	■ ★1																									
Review journal, paper, and articles about title.				■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
Identify objective and scope				■	■ ★2																							
Report writing				■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
Develop the simulation of SVM using MATLAB/SIMULINK					■	■	■	■	■	■	■	■	■															
Setup and implementation of hardware															■	■	■	■	■	■	■	■	■	■	■			
Evaluate experimentation setup using SVM technique																	■	■	■	■	■	■	■	■	■			
Proposal FYP and presentation													■ ★3															■ ★5
Report submission														■ ★4												■ ★6		

★1	25 SEPTEMBER 2017
★2	13 OCTOBER 2017
★3	6 DECEMBER 2017
★4	22 DECEMBER 2017
★5	18 MAY 2017
★6	21 MAY 2017

APPENDIX B

LIST FOR MATLAB FUNCTION SOURCE CODE

B.1 Sector identification and calculating the alpha and beta voltage vector

This source code is written in MATLAB FUCTION blocks in figure 3.12.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%   WRITTEN BY HALIMAHANI BINTI ROSLI
%   identification sector and calcutation valpha and vbeta

%   Inputs are magnitude u1(:), angle u2(:), and
%   ramp time signal for comparison u3(:)

function sector_valpha_vbeta =sub1(u)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%initialization
sec=1;                      % initial value of sector=1
theta_sec = 0;              % angle within a sector = 0

%define input
vd=u(1);
vq=u(2);

%Calculation of magnitude and angle -reference vector
mag = sqrt(vd^2+vq^2);      % magnitude
theta = atan2(vq,vd);       % angle
theta_deg = theta*180/pi;   % conversion: degrees to theta

%Determination of sector
if((theta_deg >=0)&&(theta_deg < 60))
    sec = 1;                % sector = 1
elseif((theta_deg >=60)&&(theta_deg < 120))
    sec = 2;                % sector = 2
elseif((theta_deg >=120)&&(theta_deg < 180))
    sec = 3;                % sector = 3
elseif((theta_deg >= -180)&&(theta_deg < -120))
    sec = 4;                % sector = 4
elseif((theta_deg >= -120)&&(theta_deg < -60))
    sec = 5;                % sector = 5
elseif((theta_deg >= -60)&&(theta_deg < 0))
    sec = 6;                % sector = 6
```

```
end

%Calculation of valpha and vbeta
%theta_sec is the angle within a sector
%theta_sec is ranging from 0 to 60 degrees
if(sec==1)
    theta_sec = theta;
elseif(sec == 2)
    theta_sec = 2*pi/3-theta;
elseif(sec == 3)
    theta_sec = theta-2*pi/3;
elseif(sec == 4)
    theta_sec = -2*pi/3-theta;
elseif(sec == 5)
    theta_sec = theta+2*pi/3;
elseif(sec == 6)
    theta_sec = -theta;
end
valpha = mag*cos(theta_sec);
vbeta = mag*sin(theta_sec);

% output
sector_valpha_vbeta=[sec, valpha, vbeta];
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

B.2 On times calculation

This source code is written in MATLAB FUCTION 1 blocks in figure 3.12.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   WRITTEN BY HALIMAHANI BINTI ROSLI
%   CALCULATION ON DURATION

function ontimescalculation =sub2(u)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%initialization
Vdc = 100;                    % input DC link voltage
T = 1/1000;                   % sampling period = 1/f

%define input
valpha=u(1);
vbeta=u(2);

%calculation of ta, tb & tz.
ta = 1.5*T/Vdc*(valpha-vbeta/sqrt(3));
tb = sqrt(3)*vbeta*T/Vdc;
```

```
tz = T - ta - tb;

% output
ontimescalculation=[ta; tb; tz];
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

B.3 Mapping of vector

This source code is written in MATLAB FUCTION 2 blocks in figure 3.12.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   WRITTEN BY HALIMAHANI BINTI ROSLI
%   Calculation of on-duration

function mapping_vector = fpga_lut(u)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%initialization
sx=[0; 0; 0];

%define input
sec=u(1);                           % no of sector
s1=u(2);
s2=u(3);
s3=u(4);

%generation of gate pulses s1, s2 & s3;
%For sector=1
**********************************************************
if (sec == 1)
    if (s1 == 0 && s2 == 0 && s3 == 0)
    sx = [0; 0; 0];
    elseif (s1 == 1 && s2 == 0 && s3 == 0)
    sx = [1; 0; 0];
    elseif (s1 == 1 && s2 == 1 && s3 == 0)
    sx = [1; 1; 0];
    elseif (s1 == 1 && s2 == 1 && s3 == 1)
    sx = [1; 1; 1];
    end
elseif (sec == 2)
    if (s1 == 0 && s2 == 0 && s3 == 0)
    sx = [0; 0; 0];
    elseif (s1 == 1 && s2 == 0 && s3 == 0)
    sx = [0; 1; 0];
    elseif (s1 == 1 && s2 == 1 && s3 == 0)
    sx = [1; 1; 0];
    elseif (s1 == 1 && s2 == 1 && s3 == 1)
    sx = [1; 1; 1];
    end
elseif (sec == 3)
    if (s1 == 0 && s2 == 0 && s3 == 0)
    sx = [0; 0; 0];
    elseif (s1 == 1 && s2 == 0 && s3 == 0)
    sx = [0; 1; 0];
    elseif (s1 == 1 && s2 == 1 && s3 == 0)
    sx = [0; 1; 1];
    elseif (s1 == 1 && s2 == 1 && s3 == 1)
    sx = [1; 1; 1];
    end
elseif (sec == 4)
    if (s1 == 0 && s2 == 0 && s3 == 0)
    sx = [0; 0; 0];
    elseif (s1 == 1 && s2 == 0 && s3 == 0)
    sx = [0; 0; 1];
    elseif (s1 == 1 && s2 == 1 && s3 == 0)
    sx = [0; 1; 1];
    elseif (s1 == 1 && s2 == 1 && s3 == 1)
    sx = [1; 1; 1];
    end
elseif (sec == 5)
    if (s1 == 0 && s2 == 0 && s3 == 0)
    sx = [0; 0; 0];
    elseif (s1 == 1 && s2 == 0 && s3 == 0)
    sx = [0; 0; 1];
    elseif (s1 == 1 && s2 == 1 && s3 == 0)
    sx = [1; 0; 1];
    elseif (s1 == 1 && s2 == 1 && s3 == 1)
    sx = [1; 1; 1];
    end
elseif (sec == 6)
    if (s1 == 0 && s2 == 0 && s3 == 0)
    sx = [0; 0; 0];
    elseif (s1 == 1 && s2 == 0 && s3 == 0)
    sx = [1; 0; 0];
    elseif (s1 == 1 && s2 == 1 && s3 == 0)
    sx = [1; 0; 1];
    elseif (s1 == 1 && s2 == 1 && s3 == 1)
    sx = [1; 1; 1];
    end
end

% output
mapping_vector = sx;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

APPENDIX C

LIST FOR VHDL SOURCE CODE LISTING

C.1 The main VHDL coding structure

```
--==========================================================
-- Entity: main coding
-- Author: HALIMAHANI BINTI ROSLI
--==========================================================
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;

entity tutor1_pwm is
port(
     clk: in std_logic;
     rst: in std_logic;

     saup, salow: out std_logic;
     sbup, sblow: out std_logic;
     scup, sclow: out std_logic

);
end entity;

architecture rtl of tutor1_pwm is

     signal address : std_logic_vector (13 downto 0);
-- "n-1 downto 0" 2^n=16384 < num_data=12001, where n=14
     signal clk_200k: std_logic;
-- clk freq FPGA = 50MHz, clock freq is divided to 200 kHz
(1/5us)
     signal sa, sb, sc: std_logic_vector (0 downto 0);
```

```
begin

     process (clk, rst) is
     begin
          if (rst = '0') then
               address <= (others =>'0');
          elsif (rising_edge(clk)) then
               if (clk_200k = '1') then
                    if (unsigned(address) < 12001) then
     -- num_data = 12001 in simulation
                         address <=
std_logic_vector(unsigned(address) + 1);
                    else
                         address <= (others =>'0');
                    end if;
               end if;
          end if;
     end process;

     --============================================
     -- instantiate PWM table (ROM) to the code
     --============================================

     lut_sa:
     entity work.sa (SYN) port map (address => address,
clock => clk, q => sa);
     lut_sb:
     entity work.sb (SYN) port map (address => address,
clock => clk, q => sb);
     lut_sc:
     entity work.sc (SYN) port map (address => address,
clock => clk, q => sc);
```

```
     --===========================================
     -- generating sampling clock with f = 200 KHz
     --===========================================
     gen_200KHz_clk:
     entity work.clk_div (rtl) port map (clk => clk, rst
=> rst, clkout => clk_200k);

--=======================================================
--Generating blanking time function for induction machine
--=======================================================
     blanking_time_2microsec:
     entity work.blankingtime_main (behavioral) port map
(clock =>clk, sa1 => sa, sb1 =>sb, sc1 => sc, saupper =>
saup, salower => salow, sbupper => sbup, sblower => sblow,
scupper => scup, sclower => sclow);
     --x <= address;

end rtl;
--========================================================
```

C.2 Blanking Time Generator

```
--========================================================
-- Entity: Blanking Time Generator
-- Author: HALIMAHANI BINTI ROSLI
--========================================================

library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity blankingtime_main is
     port ( clock : in std_logic;
             --INPUT PORT--
             sa1, sb1, sc1: in std_logic_vector (0 downto
0);
             --OUTPUT PORT--
             saupper, sbupper, scupper: out std_logic;

             salower, sblower, sclower: out std_logic);
end blankingtime_main;

                --blanking_time_signal--
architecture behavioral of blankingtime_main is
signal a1,a2, b1, b2, c1, c2: std_logic_vector (12 downto
0);

--clock_out
signal T : std_logic;

begin
--==========================================================
--clock_divide
--==========================================================
u1 : entity work.blanking_mod18(behavioral) port map (clk
=> clock, clkout => T);
--==========================================================
--upper_counter
--==========================================================
u2 : entity work.blanking_upper_counter(behavioral) port
map (clk => T, clear1 => sa1, clear2 => sb1, clear3 =>
sc1, uc1 => a1, uc2 => b1, uc3 => c1);
--==========================================================
--lower_counter
--==========================================================
u3 : entity work.blanking_lower_counter(behavioral) port
map (clk => T, clear1 => sa1, clear2 => sb1, clear3 =>
sc1, lc1 => a2, lc2 => b2, lc3 => c2);
--==========================================================
--Comperator
--==========================================================
u4 : entity work.blanking_comparator(behavioral) port map
(Cin1 => a1, Cin2 => b1, Cin3 =>c1, Cin13 => a2, Cin14 =>
b2, Cin15 =>c2, Cout1 => saupper, Cout2 => sbupper, Cout3
=> scupper, Cout13 => salower, Cout14 => sblower, Cout15
=> sclower);

end behavioral;
--==========================================================
```

C.3 Clock division

```
--==========================================================
-- Entity: clk_div
-- Author: HALIMAHANI BINTI ROSLI
-- Brief : to generate xMHz clock from 50MHz sys_clk
--==========================================================

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;

entity clk_div is
port (
clk       : in std_logic;
rst       : in std_logic;
clkout    : out std_logic
);
end entity;
```

```
architecture rtl of clk_div is

constant cnt_max : integer range 0 to 250 := 250;
-- 250 = Ts_matlab / Ts_FPGA = 5us/20ns, 20ns = 1/FPGA clk freq
signal   cnt     : integer range 0 to cnt_max-1;
begin

process (clk, rst) is
begin
if (rst = '0') then
cnt <= 0;
elsif rising_edge (clk) then
if (cnt = cnt_max-1) then
cnt <= 0;
else
cnt <= cnt + 1;
end if;
end if;
end process;
-- divide by 5, result a pulse in every 250 cycles
clkout <= '1' when (cnt = cnt_max-1)  else '0';

end rtl;
--==========================================================
```

APPENDIX D

FPGA DATA SHEET

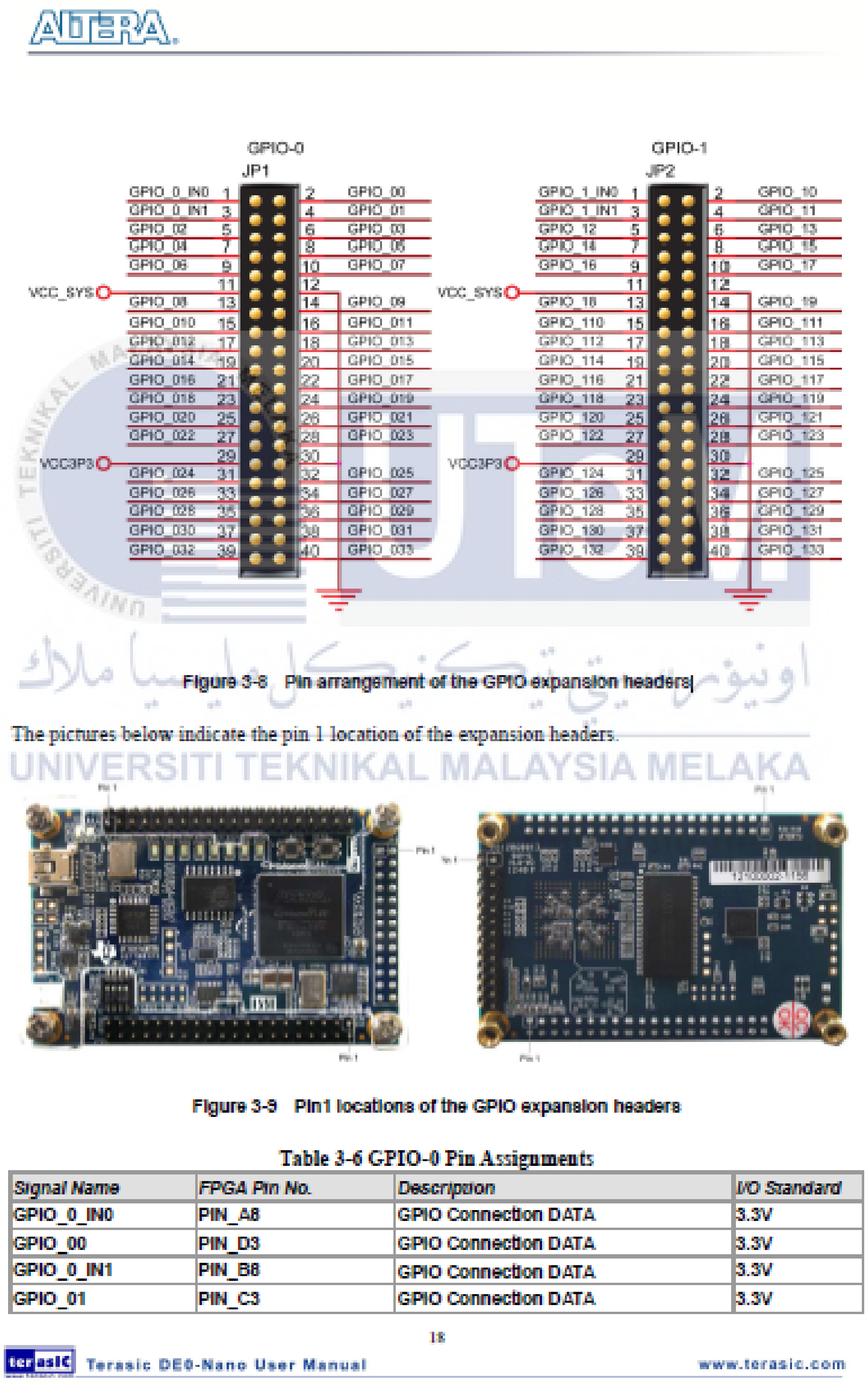

ALTERA
GPIO-0
JP1
GPIO_0_IN0 1
GPIO_0_IN1 3
GPIO_02 5
GPIO_04 7
GPIO_06 9
11
VCC_SYS
GPIO_08 13
GPIO_010 15
GPIO_012 17
GPIO_014 19
GPIO_016 21
GPIO_018 23
GPIO_020 25
GPIO_022 27
29
VCC3P3
GPIO_024 31
GPIO_026 33
GPIO_028 35
GPIO_030 37
GPIO_032 39
2 GPIO_00
4 GPIO_01
6 GPIO_03
8 GPIO_05
10 GPIO_07
12
14 GPIO_09
16 GPIO_011
18 GPIO_013
20 GPIO_015
22 GPIO_017
24 GPIO_019
26 GPIO_021
28 GPIO_023
30
32 GPIO_025
34 GPIO_027
36 GPIO_029
38 GPIO_031
40 GPIO_033
GPIO-1
JP2
GPIO_1_IN0 1
GPIO_1_IN1 3
GPIO_12 5
GPIO_14 7
GPIO_16 9
11
VCC_SYS
GPIO_18 13
GPIO_110 15
GPIO_112 17
GPIO_114 19
GPIO_116 21
GPIO_118 23
GPIO_120 25
GPIO_122 27
29
VCC3P3
GPIO_124 31
GPIO_126 33
GPIO_128 35
GPIO_130 37
GPIO_132 39
2 GPIO_10
4 GPIO_11
6 GPIO_13
8 GPIO_15
10 GPIO_17
12
14 GPIO_19
16 GPIO_111
18 GPIO_113
20 GPIO_115
22 GPIO_117
24 GPIO_119
26 GPIO_121
28 GPIO_123
30
32 GPIO_125
34 GPIO_127
36 GPIO_129
38 GPIO_131
40 GPIO_133
Pin 1
Pin 1
Pin 1
Pin 1
Pin 1
Pin 1

Figure 3-8 Pin arrangement of the GPIO expansion headers|

The pictures below indicate the pin 1 location of the expansion headers.

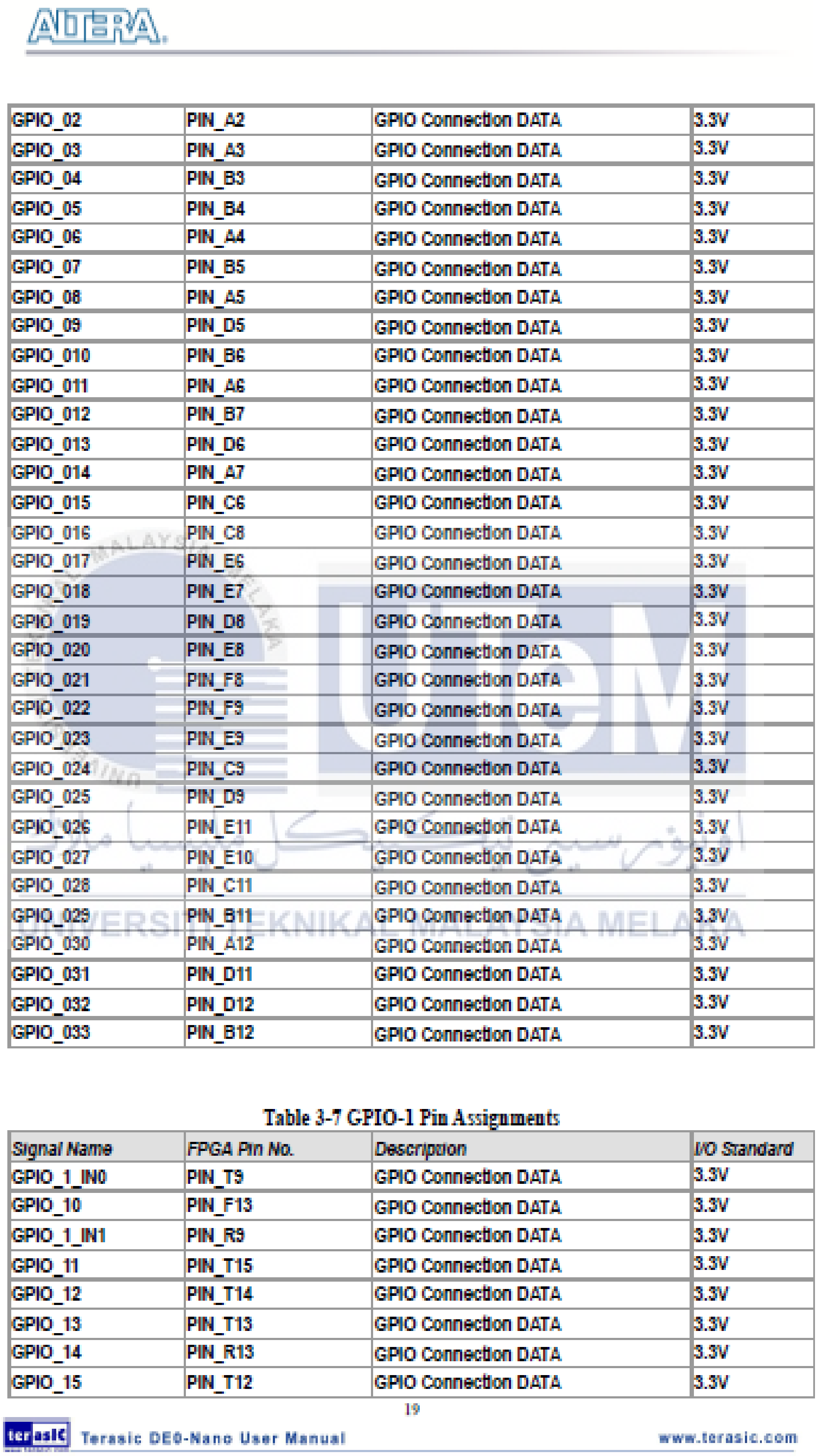
Figure 3-9 Pin1 locations of the GPIO expansion headers

Table 3-6 GPIO-0 Pin Assignments

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
GPIO_0_IN0	PIN_A8	GPIO Connection DATA	3.3V
GPIO_00	PIN_D3	GPIO Connection DATA	3.3V
GPIO_0_IN1	PIN_B8	GPIO Connection DATA	3.3V
GPIO_01	PIN_C3	GPIO Connection DATA	3.3V

18

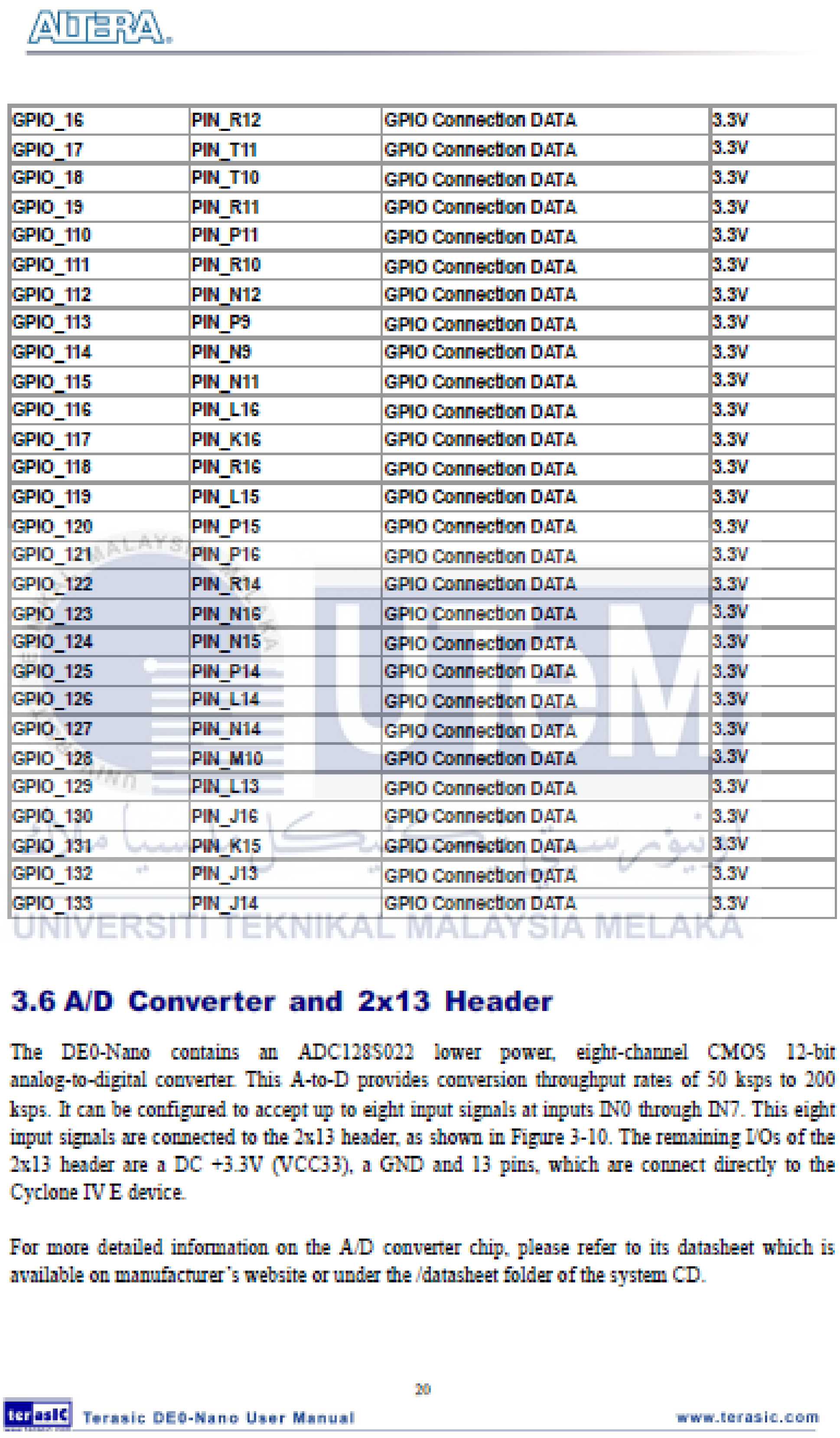
Terasic DE0-Nano User Manual www.terasic.com



GPIO_02	PIN_A2	GPIO Connection DATA	3.3V
GPIO_03	PIN_A3	GPIO Connection DATA	3.3V
GPIO_04	PIN_B3	GPIO Connection DATA	3.3V
GPIO_05	PIN_B4	GPIO Connection DATA	3.3V
GPIO_06	PIN_A4	GPIO Connection DATA	3.3V
GPIO_07	PIN_B5	GPIO Connection DATA	3.3V
GPIO_08	PIN_A5	GPIO Connection DATA	3.3V
GPIO_09	PIN_D5	GPIO Connection DATA	3.3V
GPIO_010	PIN_B6	GPIO Connection DATA	3.3V
GPIO_011	PIN_A6	GPIO Connection DATA	3.3V
GPIO_012	PIN_B7	GPIO Connection DATA	3.3V
GPIO_013	PIN_D6	GPIO Connection DATA	3.3V
GPIO_014	PIN_A7	GPIO Connection DATA	3.3V
GPIO_015	PIN_C6	GPIO Connection DATA	3.3V
GPIO_016	PIN_C8	GPIO Connection DATA	3.3V
GPIO_017	PIN_E6	GPIO Connection DATA	3.3V
GPIO_018	PIN_E7	GPIO Connection DATA	3.3V
GPIO_019	PIN_D8	GPIO Connection DATA	3.3V
GPIO_020	PIN_E8	GPIO Connection DATA	3.3V
GPIO_021	PIN_F8	GPIO Connection DATA	3.3V
GPIO_022	PIN_F9	GPIO Connection DATA	3.3V
GPIO_023	PIN_E9	GPIO Connection DATA	3.3V
GPIO_024	PIN_C9	GPIO Connection DATA	3.3V
GPIO_025	PIN_D9	GPIO Connection DATA	3.3V
GPIO_026	PIN_E11	GPIO Connection DATA	3.3V
GPIO_027	PIN_E10	GPIO Connection DATA	3.3V
GPIO_028	PIN_C11	GPIO Connection DATA	3.3V
GPIO_029	PIN_B11	GPIO Connection DATA	3.3V
GPIO_030	PIN_A12	GPIO Connection DATA	3.3V
GPIO_031	PIN_D11	GPIO Connection DATA	3.3V
GPIO_032	PIN_D12	GPIO Connection DATA	3.3V
GPIO_033	PIN_B12	GPIO Connection DATA	3.3V

Table 3-7 GPIO-1 Pin Assignments

Signal Name	*FPGA Pin No.*	*Description*	*I/O Standard*
GPIO_1_IN0	PIN_T9	GPIO Connection DATA	3.3V
GPIO_10	PIN_F13	GPIO Connection DATA	3.3V
GPIO_1_IN1	PIN_R9	GPIO Connection DATA	3.3V
GPIO_11	PIN_T15	GPIO Connection DATA	3.3V
GPIO_12	PIN_T14	GPIO Connection DATA	3.3V
GPIO_13	PIN_T13	GPIO Connection DATA	3.3V
GPIO_14	PIN_R13	GPIO Connection DATA	3.3V
GPIO_15	PIN_T12	GPIO Connection DATA	3.3V



GPIO_16	PIN_R12	GPIO Connection DATA	3.3V
GPIO_17	PIN_T11	GPIO Connection DATA	3.3V
GPIO_18	PIN_T10	GPIO Connection DATA	3.3V
GPIO_19	PIN_R11	GPIO Connection DATA	3.3V
GPIO_110	PIN_P11	GPIO Connection DATA	3.3V
GPIO_111	PIN_R10	GPIO Connection DATA	3.3V
GPIO_112	PIN_N12	GPIO Connection DATA	3.3V
GPIO_113	PIN_P9	GPIO Connection DATA	3.3V
GPIO_114	PIN_N9	GPIO Connection DATA	3.3V
GPIO_115	PIN_N11	GPIO Connection DATA	3.3V
GPIO_116	PIN_L16	GPIO Connection DATA	3.3V
GPIO_117	PIN_K16	GPIO Connection DATA	3.3V
GPIO_118	PIN_R16	GPIO Connection DATA	3.3V
GPIO_119	PIN_L15	GPIO Connection DATA	3.3V
GPIO_120	PIN_P15	GPIO Connection DATA	3.3V
GPIO_121	PIN_P16	GPIO Connection DATA	3.3V
GPIO_122	PIN_R14	GPIO Connection DATA	3.3V
GPIO_123	PIN_N16	GPIO Connection DATA	3.3V
GPIO_124	PIN_N15	GPIO Connection DATA	3.3V
GPIO_125	PIN_P14	GPIO Connection DATA	3.3V
GPIO_126	PIN_L14	GPIO Connection DATA	3.3V
GPIO_127	PIN_N14	GPIO Connection DATA	3.3V
GPIO_128	PIN_M10	GPIO Connection DATA	3.3V
GPIO_129	PIN_L13	GPIO Connection DATA	3.3V
GPIO_130	PIN_J16	GPIO Connection DATA	3.3V
GPIO_131	PIN_K15	GPIO Connection DATA	3.3V
GPIO_132	PIN_J13	GPIO Connection DATA	3.3V
GPIO_133	PIN_J14	GPIO Connection DATA	3.3V

3.6 A/D Converter and 2x13 Header

The DE0-Nano contains an ADC128S022 lower power, eight-channel CMOS 12-bit analog-to-digital converter. This A-to-D provides conversion throughput rates of 50 ksps to 200 ksps. It can be configured to accept up to eight input signals at inputs IN0 through IN7. This eight input signals are connected to the 2x13 header, as shown in Figure 3-10. The remaining I/Os of the 2x13 header are a DC +3.3V (VCC33), a GND and 13 pins, which are connect directly to the Cyclone IV E device.

For more detailed information on the A/D converter chip, please refer to its datasheet which is available on manufacturer's website or under the /datasheet folder of the system CD.