ANALOG DESIGN OF SINUSOIDAL PULSE WIDTH MODULATION

A PROJECT THESIS

Submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRICAL AND ELECTRONICS ENGINEERING

by

A. NAGA JITHENDRA REDDY– 511908

V.PAVAN KRISHNA – 511989

A. KOWSHIK – 511912

Supervisor:

Dr.K. Sri Phani Krishna

DEPARTMENT OF ELECTRICAL ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY ANDHRA PRADESH

TADEPALLIGUDEM

MAY-2023

DISSERTATION APPROVAL FOR B.TECH

This dissertation work entitled "**ANALOG DESIGN OF SINUSOIDAL PULSE WIDTH MODULATION**" by ANKIREDDDY NAGA JITHENDRA REDDY (511908), VEESAM PAVAN KRISHNA (511989) and AVUNURI KOWSHIK (511912), is approved in the partial fulfilment of the requirements for the award of degree of Bachelor of Technology in Electrical and Electronics Engineering form National Institute of Technology, Andhra Pradesh.

Examiners

_____

Supervisor

_____

Dr. K. Sri Phani Krishna
Dept. of Electrical Engineering
National Institute of Technology, Andhra Pradesh

Head of the Department

_____

Dr.V. Sandeep

Asst. Professor

Dept. of Electrical Engineering
National Institute of Technology, Andhra Pradesh

Date:_____ Place:

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the sources. We also declare that We have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will cause disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Ankireddy Naga Jithendra Reddy	**Avunuri Kowshik**	**Veesam Pavan Krishna**
Roll no. 511908	Roll no. 511912	Roll no. 511989
Date: _____	Date: _____	Date: _____

DEPARTMENT OF ELECTRICAL ENEGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY ANDHRA PRADESH

CERTIFICATE

This is to certify that the dissertation work entitled **"ANALOG DESIGN OF SINUSOIDAL PULSE WIDTH MODULATION"** by **ANKIREDDY NAGA JITHENDRA REDDY (511908), AVUNURI KOWSHIK (511912), VEESAM PAVAN KRISHNA (511989)** submitted in the partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology in Electrical and Electronics Engineering** from **National Institute of Technology, Andhra Pradesh**, during 2019-2023. This dissertation work was not submitted earlier at any other university or Institute for the award of degree.

Dr. K.SRI PHANI KRISHNA
Thesis Supervisor
Dept. of Electrical Engineering
National Institute of Technology
Andhra Pradesh

Dr.V.SANDEEP
Head of the Department
Dept. of Electrical Engineering
National Institute of Technology
Andhra Pradesh

May 2023

ACKNOWLEDGEMENT

Foremost, we would like to express our sincere gratitude to our supervisor prof. Dr.K.SRI PHANI KRISHNA for his support, time, and knowledge he shared with us. We would like to express appreciation to our team for helping in doing our project in addition to our supervisor. We would like to thank our institute, the National Institute ofTechnology Andhra Pradesh, Department of Electrical and Electronics Engineering for approving the project and providing us with an opportunity. Finally, We are immensely grateful to everyone who has played a role in making this project a success. Your contribution have been invaluable, and we appreciate all the support and encouragement you have provided us with.

Ankireddy Naga Jithendra Reddy (511908)

Avunuri Kowshik (511912)

Veesam Pavan Krishna (511989)

ABSTRACT

The pulse width modulation (PWM) technique is essential in inverters, power grids, and general-purpose AC supplies. When compared to conventional PWM techniques, the Sinusoidal Pulse Width Modulation (SPWM) technique is one of the most popular PWM techniques used for harmonic reduction. SPWM switching signals are now generated using various FPGAs, microcontrollers, and microprocessors. This report presents the SPWM generator module designed for harmonic reduction and demonstrates how to generate SPWM switching signals using various simple Op-Amp/analog circuits. All the Op-Amp circuits have been simulated, and their outputs are shown. This method is applicable in standalone-grid systems and large industrial machines with and without transformers.

`

CONTENTS

LIST OF FIGURES

LIST OF ACRONYMS

Acronym	Description
OP-AMP	Operational Amplifier
PWM	Pulse Width Modulation
SPWM	Sinusoidal Pulse Width Modulation
FPGA	Field Programmable Gate Array
DAC	Digital to Analog Converter
SPI	Serial Peripheral Interface
ASIC	Application specific Integrated Circuit
CAN	Controller Area Network
HDL	Hardware Description Language
VLSI	Very-Large-Scale-Integration
IDE	Integrated Development Environment

CHAPTER-1

INTRODUCTION

Pulse Width Modulation (PWM) is a technique used to control the amount of power delivered to a load by adjusting the duration of pulses of a fixed voltage. In PWM, a fixed voltage is switched on and off at a high frequency, and the width of the resulting pulses is varied to control the average voltage delivered to the load. PWM is commonly used in power electronics to control the speed of DC motors, regulate the voltage of DC power supplies, and control the brightness of LEDs. PWM is also used in audio amplifiers to convert digital signals into analog signals that can drive speakers. One of the advantages of PWM is that it allows for precise control of the power delivered to the load while maintaining a relatively constant input voltage. This results in better efficiency compared to other voltage regulation techniques, such as linear voltage regulators. PWM can be implemented using various techniques, such as standard PWM, variable frequency PWM, phase shifted PWM, and sinusoidal PWM. Each technique has its own advantages and disadvantages, and the selection de phase shifted specific application requirements.

1.1 PWM-Pulse width Modulation technique

PWM is a technique that is used to reduce the overall harmonic distortion THD in a load current. It uses a pulse wave in rectangular/square form that results in a variable average waveform value ftt, after its pulse width has been modulated. The time period for modulation is given by T. Therefore, waveform average value is given by

$$y = 1/T \int_0^T f(t)dt$$

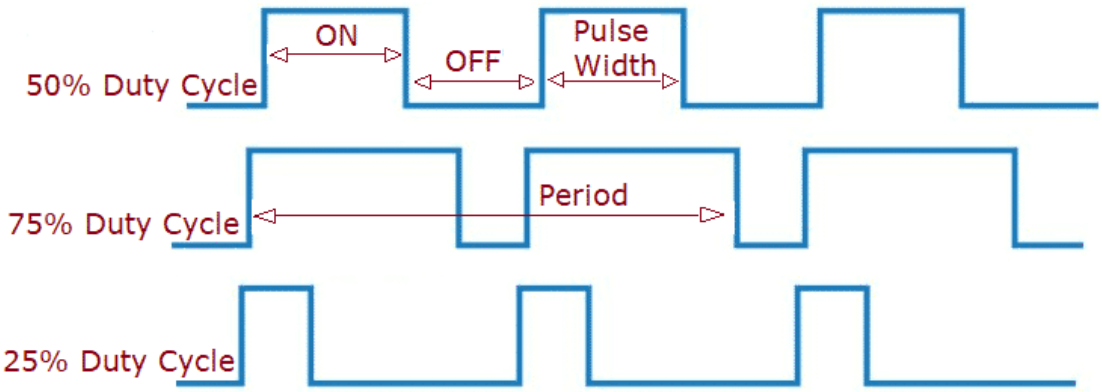


Figure1. 1 illustrates the variation of duty cycle

The proportion of time that the pulse is 'ON' or 'High' is called the **Duty Cycle**.

There are different types of PWM techniques in which mostly practised techniques are

Single pulse width modulation (SPWM)

Multiple pulse width modulation (MPWM)

Sinusoidal Pulse width modulation (SPWM)

1.2 Single pulse width modulation technique

This technique is used for single-phase circuits. Changing the width of the pulse to control the on-time or to control the gate signal of a transistor so that the output voltage of the inverter can be controlled. By changing the width or by changing the width of the courier signal we can change the on-time of the transistor. As we vary the time of the transistor, we can vary the inverter output voltage. So, if the width of the gate pulse that we are getting as an output or comparator after pulse width modulation is the width of the output voltage will be getting is d.

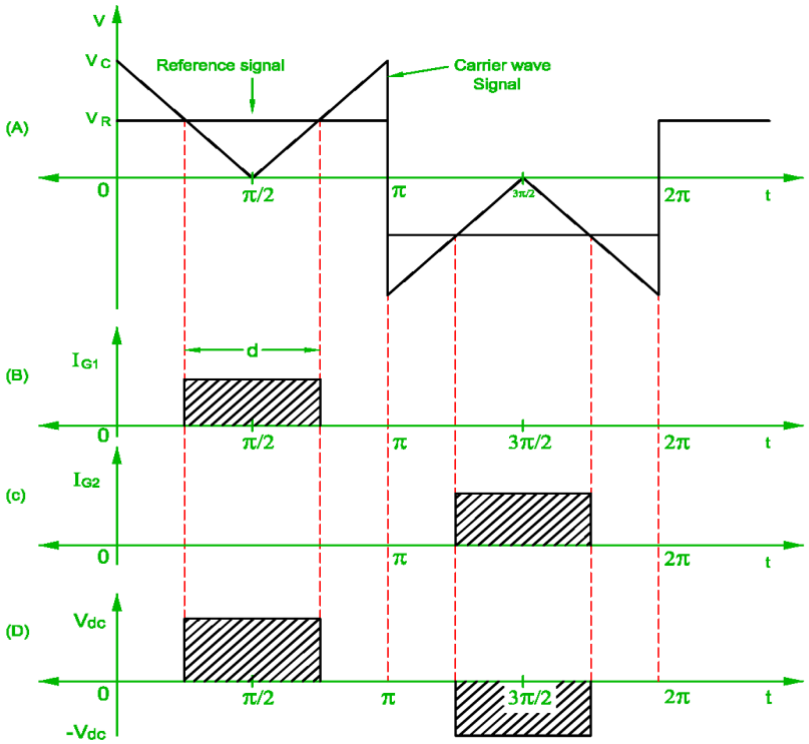


FIG H : SINGLE PULSE WIDTH MODULATION

Figure1. 2 illustrates the single pulse width modulation

1.3 Multiple pulse width modulation

Multiple pulse width modulation consists of multiple numbers of pulses per half cycle of the output voltage. Each pulse can be varied by the carrier signal. The frequency of the triangular wave is greater than that used in single pulse width modulation and it decides the number of gating signals per half cycle. By using multiple pulses, the harmonic content can be reduced. The frequency decides the number of pulses per half cycle. The output voltage can be controlled by controlling the modulation index.

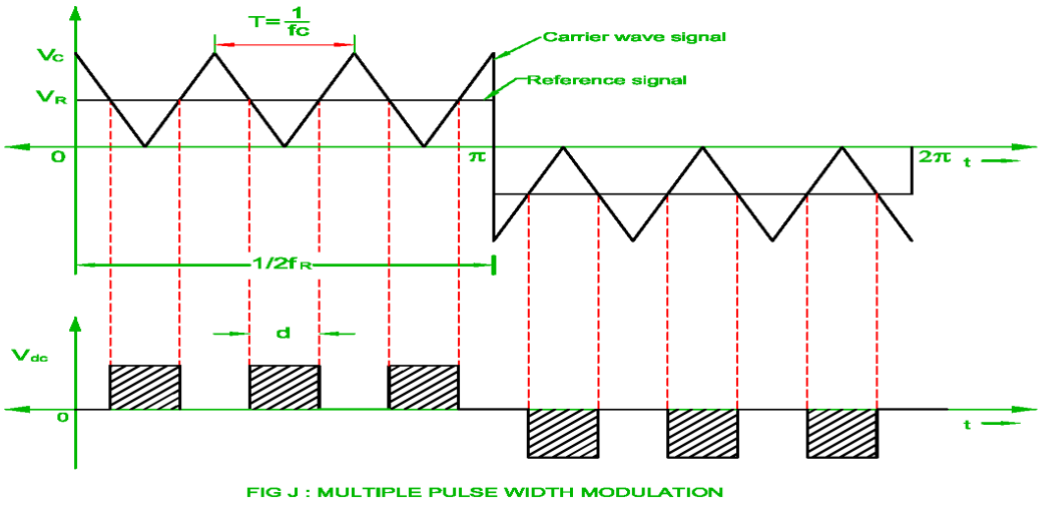


Figure1. 3 illustrates the multiple PWM

1.4 Sinusoidal PWM

In a simple source voltage inverter, the switches can be turned ON and OFF as needed. During each cycle, the switch is turned on or off once. This results in a square waveform. However, if the switch is turned on for a number of times, a harmonic profile that is improved waveform is obtained.

The sinusoidal PWM waveform is obtained by comparing the desired modulated waveform with a triangular waveform of high frequency. Regardless of whether the voltage of the signal is smaller or larger than that of the carrier waveform, the resulting output voltage of the DC bus is either negative or positive. The RMS value of the output voltage can be varied by varying the modulation index. This technique improves the distortion factor and eliminates all harmonics less than or equal to 2p-1.

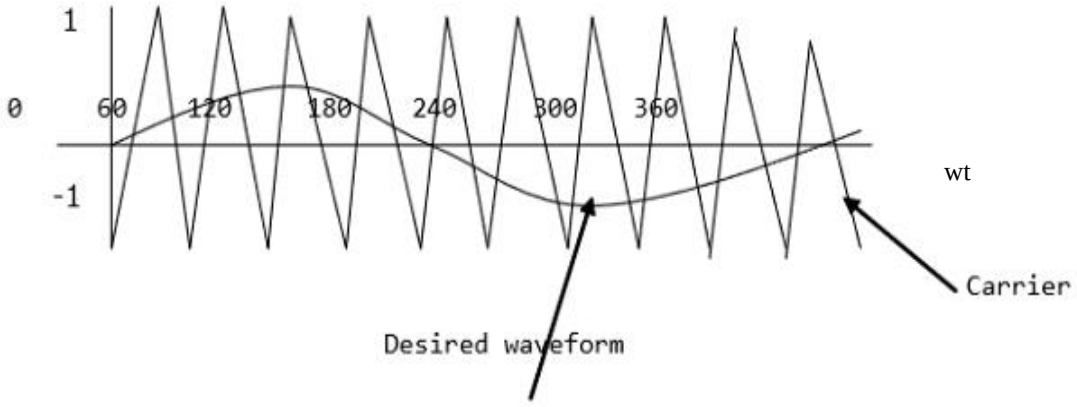


Figure 1. 4 Shows the carrier and reference signal

The sinusoidal amplitude is given as A_m and that of the carrier triangle is give as A_c. For sinusoidal PWM, the modulating index m is given by A_m/A_c.

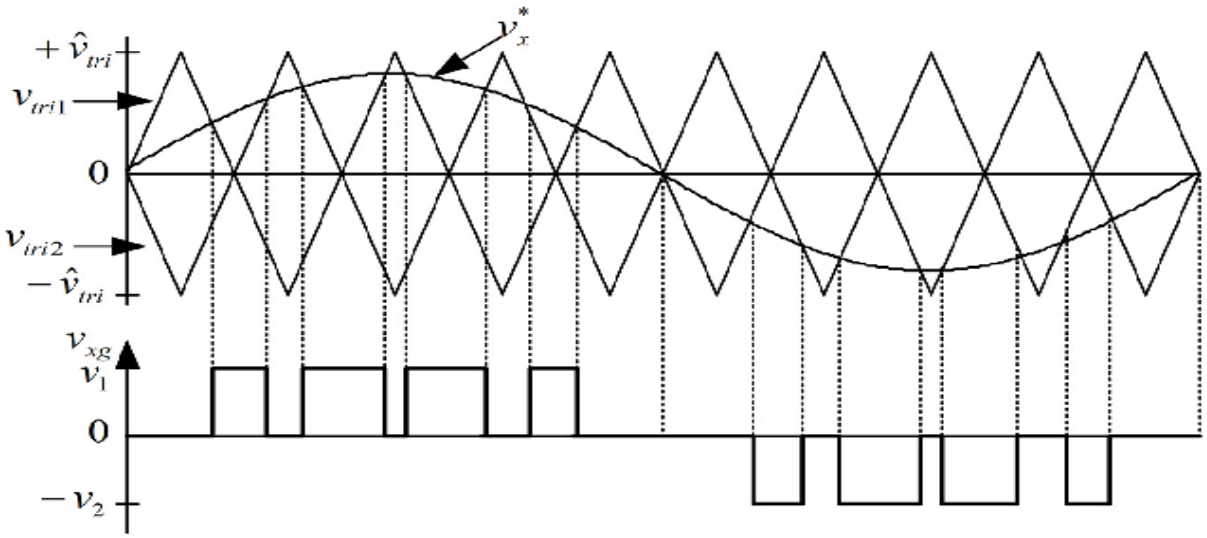


Figure1. 5 illustrates the sinusoidal PWM.

CHAPTER-2

LITERATURE REVIEW

2.1 Proposed Techniques

The analog design of a sinusoidal pulse width modulation (SPWM) generator is a crucial aspect of power electronics and motor control systems. There are several analog techniques for generating SPWM, including the use of operational amplifiers, comparators, and integrators. One common technique involves the use of a triangle waveform generator, which is compared with a sinusoidal waveform using a comparator. The output of the comparator is then used to trigger a pulse width modulator (PWM), which generates the SPWM waveform. Analog SPWM generators have several advantages over digital techniques. Firstly, they provide a more precise control over the amplitude and frequency of the output waveform. Secondly, they are less susceptible to noise and interference, resulting in a cleaner output waveform. Finally, analog techniques are generally more cost-effective than digital techniques, especially for low-frequency applications.

Analog SPWM generators have some limitations. Firstly, they require more complex circuitry than digital techniques, resulting in higher design and manufacturing costs. Secondly, they are less flexible than digital techniques, as they require manual adjustments for changes in the input signal. Finally, they are less accurate than digital techniques, as they are affected by variations in temperature and component tolerances. In conclusion, analog techniques for generating SPWM are widely used in power electronics and motor control systems. They provide a more precise control over the output waveform and are less susceptible to noise and interference. However, they require more complex circuitry and are less flexible and accurate than digital techniques. Overall, the choice between analog and digital techniques depends on the specific application and the requirements of the system.

2.1.1 Early analog techniques

The early analog techniques for generating SPWM were based on the principle of comparing a sinusoidal waveform with a triangular waveform. The first published work on this topic was by E. W. Ernst and H. R. Kemper in 1958, who proposed a

circuit for generating SPWM using a voltage-controlled oscillator and a comparator. This was followed by the work of M. E. Hagen in 1961, who proposed a similar circuit using a voltage-controlled oscillator and a pulse shaper.

2.1.2 Later analog techniques

In the 1970s, several new analog techniques for generating SPWM were proposed, including the use of operational amplifiers and integrators. One notable example is the work of J. G. Kassakian and G. C. Varghese in 1976, who proposed a circuit for generating SPWM using an operational amplifier and an integrator. This technique became widely used in power electronics and motor control systems.

2.1.3 Digital techniques

In the 1980s, digital techniques for generating SPWM became popular due to the availability of microprocessors and digital signal processors. One of the earliest examples is the work of M. R. Habetler and R. G. Harley in 1987, who proposed a digital SPWM generator using a microprocessor. This technique allowed for more flexibility and accuracy than analog techniques and became the standard in modern power electronics and motor control systems.

The history of designing sinusoidal pulse width modulation generators dates to the mid-20th century, with the early analog techniques based on the comparison of a sinusoidal waveform with a triangular waveform. Later analog techniques used operational amplifiers and integrators, while digital techniques using microprocessors became popular in the 1980s. Today, digital techniques remain the standard for generating SPWM, offering more flexibility and accuracy than analog techniques.

2.2 Problem Statement

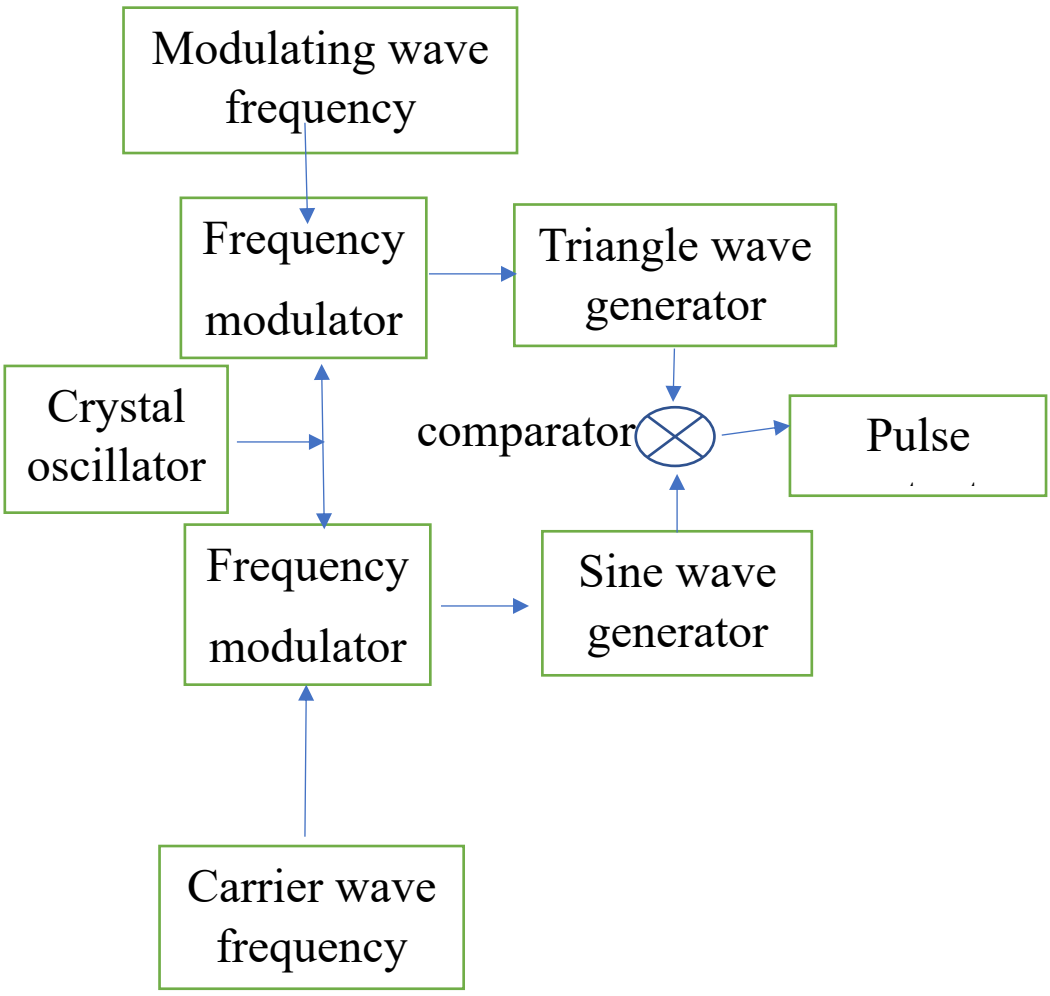
The early proposed techniques are not efficient because of the switching and the controlling techniques. The introduction of FPGA into electronics applications also helped the user to gave the provision to design a customized microcontroller or a single integrated circuit for desired applications. FPGA (Field Programmable Gate Array) helps in designing the analog circuit model of SPWM.

Power electronic inverters are developing rapidly with new designs and more levels. Harmonics have been a major issue from the start in power electronic devices

like inverters. As the switches and there sequence are controlled using microcontrollers. In most of the microcontrollers square wave pulse width modulation technique is used by which only one of the harmonics can be eliminated and therefore filter size required for obtaining pure signal, will be more.

2.3 Proposed Methodology

The below flowchart describes about the methodology of analog design of carrier signal generator and reference signal generator and implementing the PWM signal. The only two input control parameters used here are input clock pulse from crystal oscillator and modulating frequency.



2.4 FPGA

FPGA stands for Field-Programmable Gate Array. It is a type of integrated circuit that allows users to program its internal logic to perform specific functions. Unlike traditional Application-Specific Integrated Circuits (ASICs), which are designed for a specific purpose and cannot be changed after manufacturing, FPGAs can be reprogrammed as needed, making them highly versatile**.** FPGAs consist of an array of programmable logic blocks, which can be configured to perform different logic functions. These logic blocks are connected by programmable interconnects, which allow signals to be routed between the logic blocks as needed. The design of an FPGA is typically specified using a hardware description language (HDL), such as Verilog or VHDL. One of the key benefits of FPGAs is their flexibility. They can be reprogrammed to perform different functions, making them useful in a wide range of applications, including digital signal processing, image processing, machine learning, and cryptography. FPGAs can also be used to implement custom processors, allowing users to design specialized hardware for specific tasks. Another advantage of FPGAs is their high performance. Since they can be optimized for specific applications, FPGAs can often outperform general-purpose processors or ASICs in certain tasks. Additionally, FPGAs can be parallelized, allowing them to process multiple data streams simultaneously.

However, FPGAs also have some limitations. They can be expensive and complex to design, and they may not be as power-efficient as other types of circuits. Additionally, since FPGAs are reprogrammable, they may be more vulnerable to security threats than ASICs, which are typically designed for a specific purpose and cannot be reprogrammed. Overall, FPGAs are a powerful tool for implementing custom digital circuits, offering high performance and flexibility. They are widely used in a range of industries, including telecommunications, aerospace, and automotive, and they are likely to remain an important technology in the future.

2.5 Vivado Xilinx

Vivado Xilinx is a powerful tool used for FPGA programming that is widely used in the industry. It is an integrated development environment (IDE) that provides a suite of tools for FPGA design, synthesis, implementation, and verification. One of the key benefits of Vivado Xilinx is its ability to optimize FPGA designs for performance,

power, and area. It provides a range of advanced features and techniques to ensure that the FPGA design meets the desired specifications while minimizing power consumption and maximizing performance.

Vivado Xilinx also offers a user-friendly graphical interface, which simplifies the process of FPGA programming. It allows users to drag and drop elements, such as logic gates and blocks, and connect them using intuitive tools, making it easy for beginners to get started with FPGA programming. Moreover, Vivado Xilinx supports a wide range of FPGA platforms, from low-cost development boards to high-performance production-grade devices, making it a versatile tool for FPGA design. Overall, Vivado Xilinx is an essential tool for FPGA programming due to its advanced features, user-friendly interface, and support for a wide range of FPGA platforms. It can significantly simplify the FPGA design process and optimize the design for performance, power, and area, making it a valuable tool for FPGA designers and engineers.

CHAPTER-3

ANALOG DESIGN OF SINUSOIDAL PWM

3.1 Triangular wave generation using OP-AMP

We know that the integrating the square wave gives the triangle wave. Therefore, the triangle wave generation can be done by using the two op-amps. The first op-amp act as the square wave generator and the other op-amp act as the integra - tor thus providing the triangle wave.

3.1.1 Square wave

The basic square wave oscillator is based on the charging and discharging of a capacitor. Op-amps inverting input is the capacitor voltage and the noninverting input is a portion of the output fed back through resistors and (refer figure 3.1.1 and 3.1.2). When the circuit is first turned on, the capacitor is uncharged, and thus the inverting input is at 0V. This makes the output a positive maximum, and the capacitor begins to charge towards voltage at VO through resistor R. When the capacitor voltage reaches a value equal to the feedback voltage (Vf) on the non-inverting input, the op-amp switches to the maximum negative state. At this point, the capacitor begins to discharge from +Vf towards –Vf. When the capacitor voltage reaches –Vf, the op-amp switches back to the maximum positive state. This action repeats and a square wave output voltage is obtained.

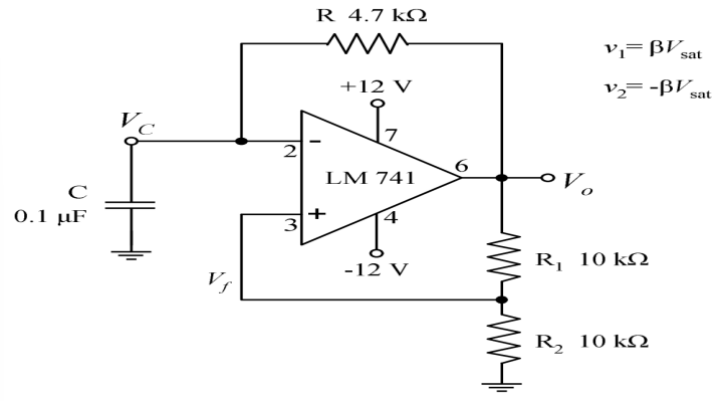


Figure 3.1. 1 schematic circuit diagram of suare wave generator

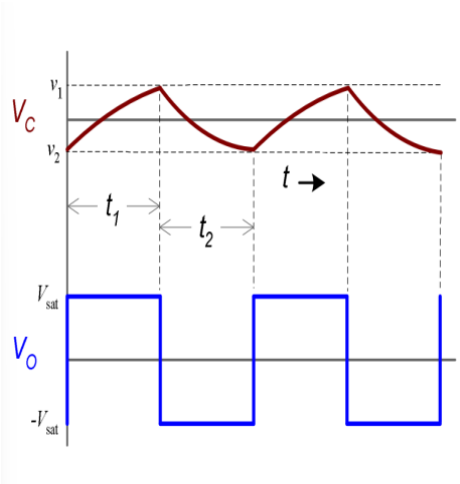


Figure 3.1. 2 illustrates the square wave output

Expression for the period is

$$T = 2RC\ ln\frac{1+\beta}{1-\beta} \qquad \text{----------(3.1)}$$

where $\beta = \frac{R_2}{R_1+R_2}$ ----------(3.2)

If R1=R2 then the equation reduces to T = RCln3 ----------(3.3)

The frequency of oscillation is $f = \frac{1}{Rc\ ln\,3}$ ----------(3.4)

3.1.2 Triangle wave

This circuit (figure 2.3 and 2.4) uses two operational amplifiers. Op-amp A1 functions as a comparator and the op-amp A2 as an integrator. Comparator compares the voltage at point P continuously with respect to the voltage at the inverting input; which as at ground potential. When the voltage at P goes slightly below zero, the output of A1 will switch to negative saturation. Suppose the output of A1 is at positive saturation +Vsat. Since this voltage is the input of the integrator, the output of A2 will be a negative going ramp. Thus, one end of the voltage divider R1-R2 is at +Vsat and the other at the negative going ramp. At time t = t1, when the negative going ramp attains value of –Vramp the effective voltage at point P becomes slightly less than 0 V. This switches output of A1 from positive saturation to negative saturation level –Vsat. During the time when the output of A1 is at –Vsat, the output of A2 increases in positive direction. At the instant t = t2, the voltage at point P becomes just above 0 V, thereby switching the output of A1 from –Vsat to +Vsat. The cycle repeats and generates a triangular waveform.

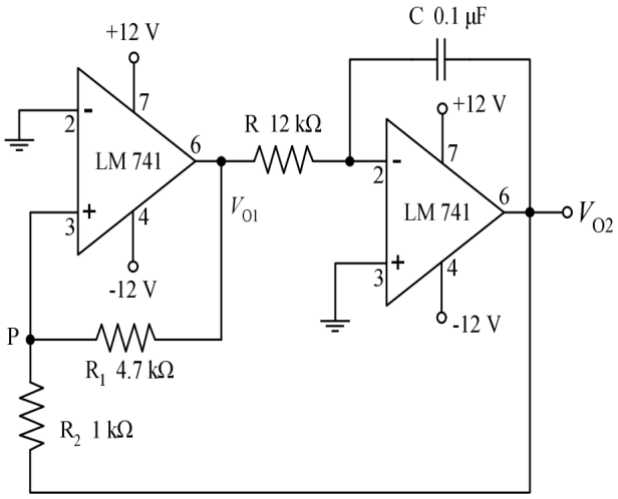


Figure 3.1.2. 1 illustrates the analog circuit using op amp

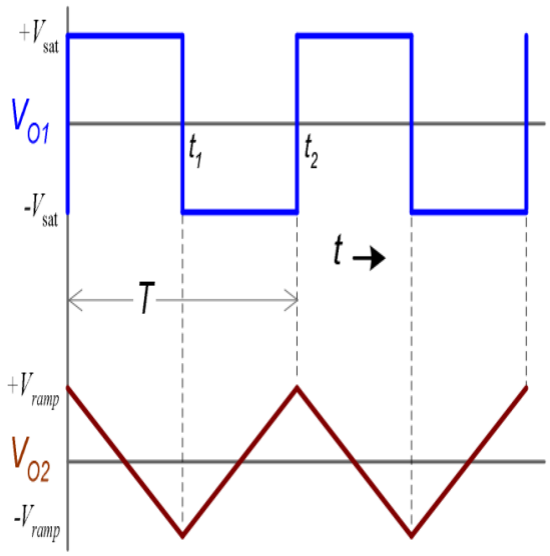


Figure 3.1.2. 2 illustrates the output square and traingle waves

At t =t_1 i.e. $\frac{-Vramp}{R2} = \frac{-Vstat}{R1}$ ---------(3.5)

$$-Vramp = \frac{-R2}{R1}(+Vsat) \quad \text{----------(3.6)}$$

$$+Vramp = \frac{-R_2}{R_1}(+v_{sat}) \quad \text{----------(3.7)}$$

Similarly at t= t_2

The peak-to-peak output is

$$Vo = +Vramp - (Vramp) = 2\frac{R2}{R1}Vsat \quad \text{-----(3.8)}$$

During period 0-t_1, the integrator function as below

$$Vo = \frac{1}{RC}\int_0^{T/2}(-Vsat)dt = (\frac{Vsat}{RC})(\frac{T}{2}) \quad \text{----(3.9)}$$

$$T = RCln(\frac{Vo}{Vsat}) \quad \text{-----(3.10)}$$

Then,

Substituting Vo then T=(4RCR_2)/R_1 ------(3.11)

Then the frequency of oscillation will be f=(1/T)

3.2 Sinusoidal wave generation using OP-AMP

To implement the generation of the sinusoidal there are many ways to generate the sinusoidal wave, one of them is rc phase shift oscillator.

3.2.1 RC phase shift oscillator

R-C phase shift oscillator using op-amp uses an op-amp in inverting amplifier mode. Thus, it introduces the phase shift of 180° between input and output. The feedback network consists of 3 RC sections each producing 60° phase shift. Such an RC phase shift oscillator using op-amp is shown in Fig. 3.2.1.

The output of the amplifier is given to the feedback network. The output of the feedback network drives the amplifier. The total phase shift around a loop is 180° of the amplifiers and 180° due to 3 RC section, thus 360°. This satisfies the required condition for positive feedback and circuit works as an oscillator.

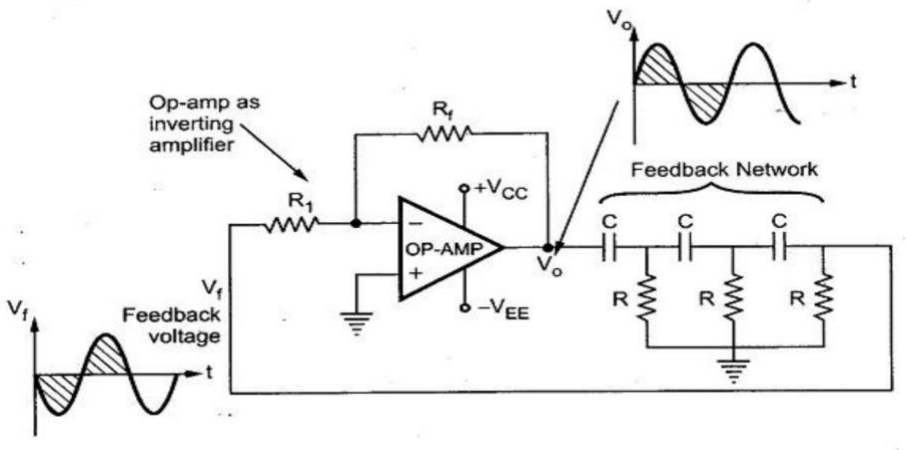


Figure 3.2. 1.1 rc phase shift oscillator

By finding the transfer function of the rc feedback network

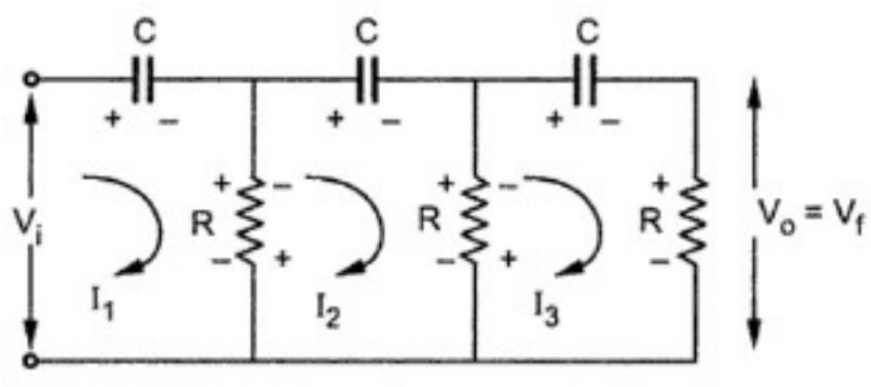


Figure 3.2.1.2 RC phase shift oscillator

Applying kvl to the various loops we get

$$i_1R + \frac{i_1}{j\omega c} - i_2R = v_i \qquad \text{------(3.12)}$$

$$i_1R + 2i_2R + \frac{i_2}{jwc} - i_3R = 0 \qquad \text{------(3.13)}$$

$$0 - i_2R + \frac{i_3}{jwc} + i_3R = 0 \qquad \text{----(3.14)}$$

Replacing jw by s

$$\begin{vmatrix} R+\frac{1}{sC} & -R & 0 \\ -R & 2R+\frac{1}{sC} & -R \\ 0 & -R & 2R+\frac{1}{sC} \end{vmatrix} \begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \begin{bmatrix} V_i \\ 0 \\ 0 \end{bmatrix}$$

Using crammers method to find i_3

$$i_3 = \frac{v_i \cdot R^2 s^3 c^3}{1+5sCR+6s^2C^2R^2+s^3C^3R^3} \qquad \text{---------(3.15)}$$

$v_0 = v_f = i_3R$

So $\beta = \frac{-jw^3R^3c^3}{1+5jwCR-6w^2C^2R^2-w^3C^3R^3}$ -------(3.16)

By replacing the $\frac{1}{wRC}$ $by\ \alpha\ then$

$$\beta = \frac{1}{1-5\alpha^2+j\alpha 6-j\alpha^2} \qquad \text{----(3.17)}$$

For phase shift to be 180 the imaginary part should be 0 so,

$$6\alpha = \alpha^3 \text{ ----------(3.18)}$$

So the frequency of oscillation is

$$f = \frac{1}{2\pi RC\sqrt{6}} \text{ -----(3.19)}$$

3.2.2 Mathematical modelling of sinewave

For mathematical modelling of sinewave .the tailor series is being employed the tailor gives the sine function intern's of a polynomial.

Sin'(x)=cos(x), sin''(x)=-sin(x),sin'''(x)=-cos(x), sin''''(x)=sinx,

And the pattern repeats

The taylor's formula now tells us that

$$sin(x) = 0 + 1x + 0x^2 - \frac{x^3}{3!} + Ox^4 + \cdots \text{ -------(3.20)}$$

Thus equals for

$$sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots \text{ ------(3.21)}$$

Using this taylor's expansion we can model the sine wave

CHAPTER-4

ANALOG CIRCUIT SIMULATION IN MULTISIM

Multisim is a circuit simulation software developed by National Instruments. It is a powerful tool that allows users to design and simulate electronic circuits before building them in hardware. Multisim provides a user-friendly interface and a wide range of components, making it an excellent choice for both beginners and professionals.

One of the key features of Multisim is its intuitive graphical interface, which allows users to drag and drop components onto a virtual breadboard and connect them using virtual wires. This makes it easy to design and simulate complex circuits quickly and efficiently. Multisim also provides a range of analysis tools, such as AC/DC simulation, transient analysis, and Monte Carlo analysis, allowing users to test the performance of their circuits under various conditions. Multisim also offers a library of pre-built components, including resistors, capacitors, inductors, diodes, transistors, and ICs. These components can be easily added to the circuit design, and their properties can be customized as needed. Additionally, Multisim supports a range of industry-standard components and models, making it easy to design and simulate circuits that can be built in hardware.

The implementation of the analog design for triangular wave generation, sine wave generation and the final PWM wave generation are given below:

4.1 Triangular wave generation

The below given figure 4.1.1 is the schematic analog circuit diagram for generating the triangular wave as carrier waveform. In this model, we use two OP-Amps of LM324NG, capacitors, resistors and two voltmeter probes and their simulated results are shown in figure 4.1.2.

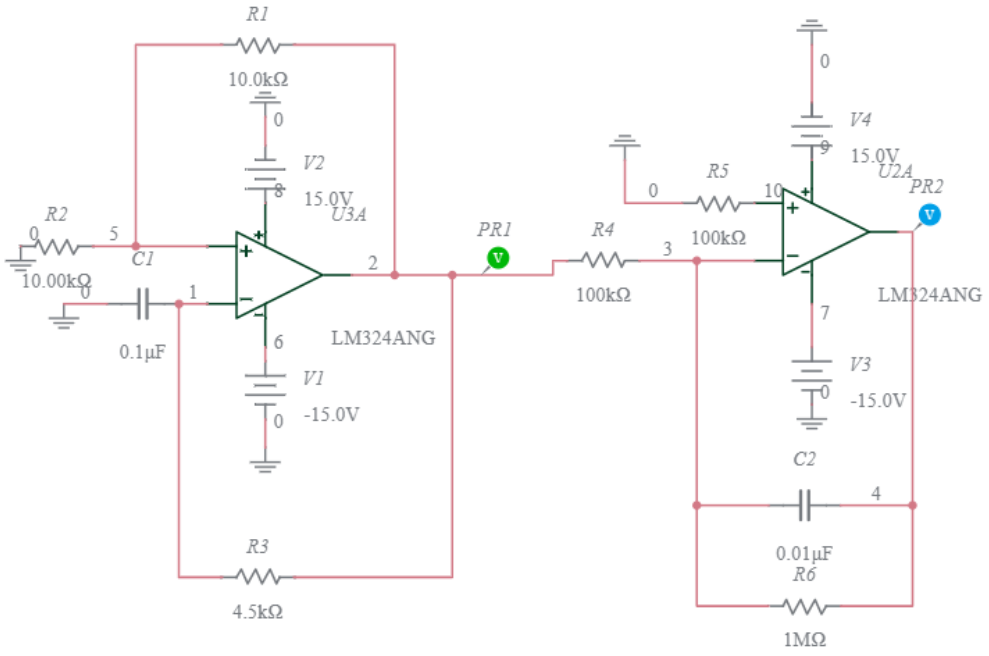


figure 4.1. 1 Trianglewave generation simulink

The simulated results can be seen in the grapher section:

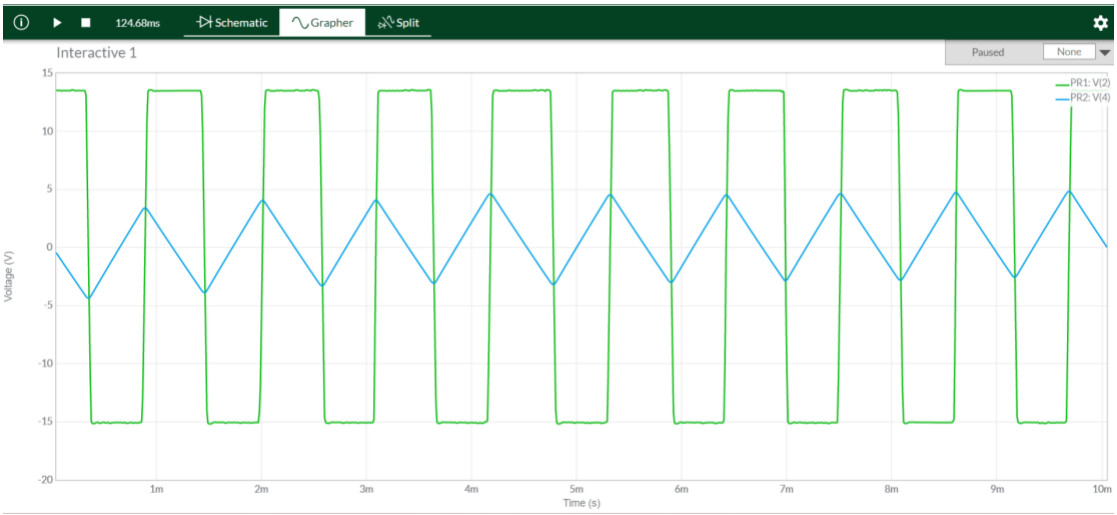


figure 4.1. 2 Trianglewave simulation output

4.2 Sine Wave generation

The below given is the basic schematic analog circuit diagram of sine wave generation which is the modulating pulse used for PWM in figure 4.2.1.

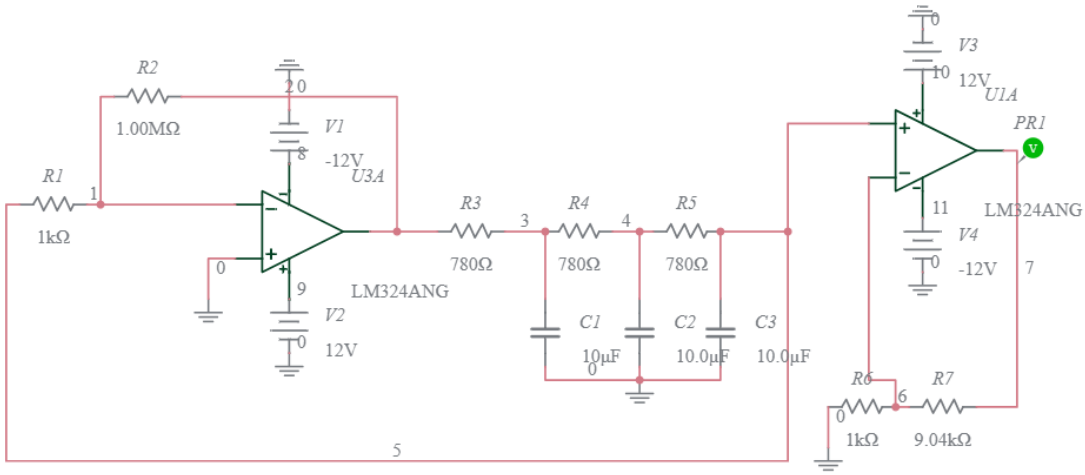


figure 4.2. 1 sinwave generation simulink

The below shown figure is the simulated result in the graph section showing the sinewave in figure 4.2.1

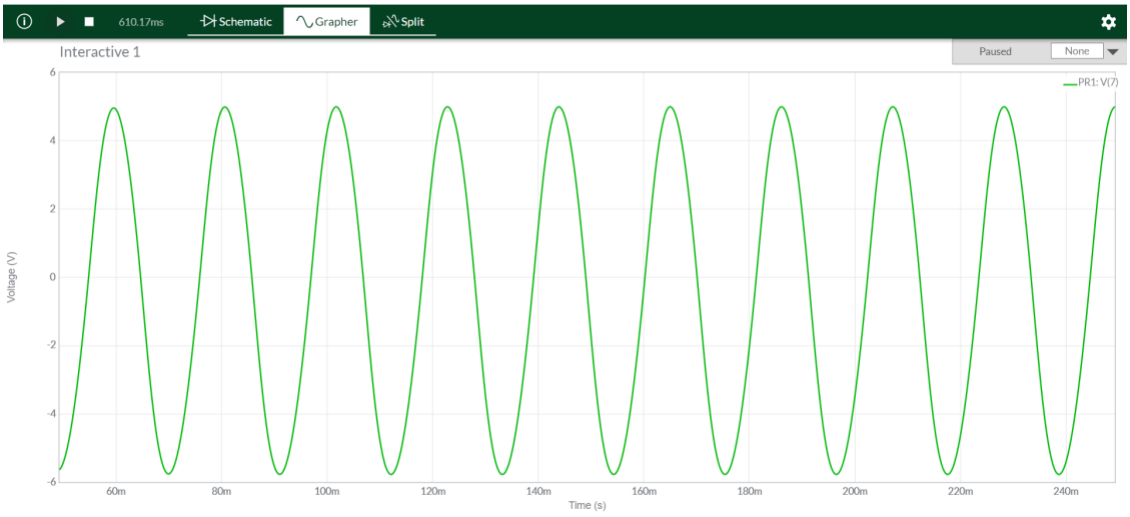


figure 4.2. 2 sinwave simulation output

4.3 SPWM signal generation

This is the final output signal that to be expected when it is running in hardware. In this the comparator block is placed by using an op amp block that compares the two signals between the sinewave and triangular wave. In this, the output signal is represented when the sine wave and triangular wave are compared: when the sine wave is smaller than the triangular wave, output signal becomes 0 and sine wave is greater than the carrier wave then it becomes 1. It acts as sine wave Pulse Width Modulation signal, that can

be used to various applications. The output schematic diagram is the given below in figure 4.3.1

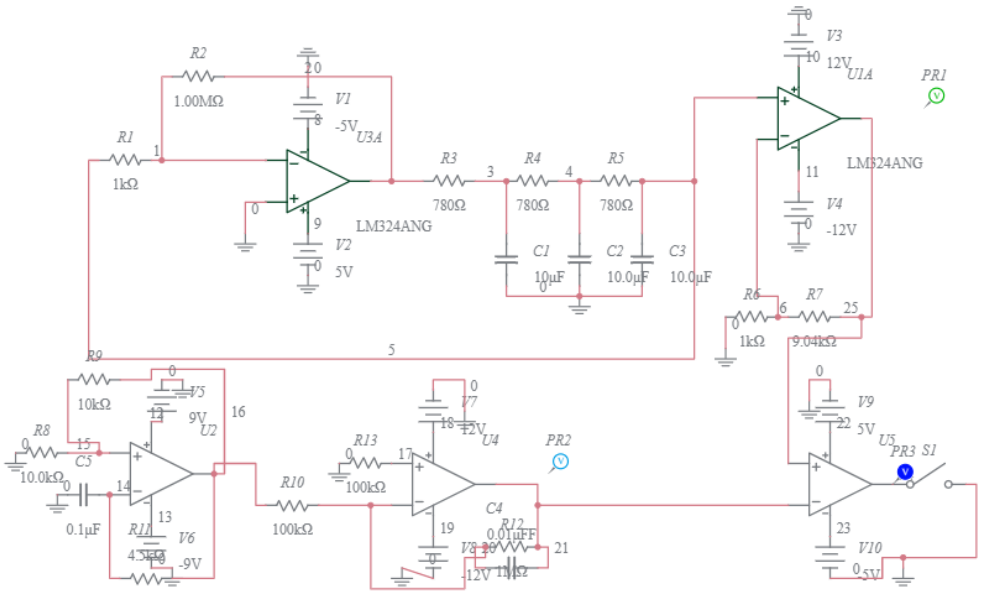


Figure 4.3. 1 spwm simulink

The below output is the related graph of the sinusoidal PWM technique signal generated from the above figure 4.3.1 which is shown below in figure in 4.3.2.

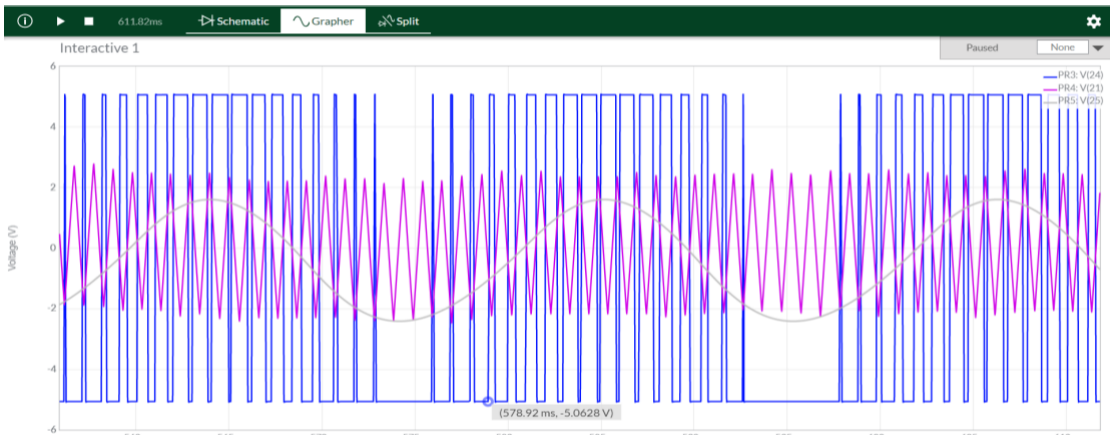


Figure 4.3. 2 spwm simulation output

CHAPTER-5

IMPLEMENTATION ON FPGA BOARD

5.1 Steps for Vivado Simulation

Vivado simulation is the basic open-source FPGA tool used for simulation. The vivado IDE should be installed into the Windows or mac software. It can be programmable and circuit level connection also possible. To simulate a design in Vivado, you typically need to follow these steps:

- Create a testbench: A testbench is a file that provides stimulus to the design and verifies its response. In Vivado, you can create a testbench using the Waveform Editor, which allows you to define input stimulus and expected output values.
- Compile the design: Before you can simulate a design, you need to compile it using the Vivado Synthesis and Implementation tools. This process generates a simulation model that can be used in the simulation.
- Run the simulation: Once the design has been compiled and the testbench created, you can run the simulation in Vivado. To do this, you can use the Simulator tool, which allows you to specify the simulation duration, clock frequency, and other parameters.
- Analyze the results: After the simulation has completed, you can analyze the results using the Waveform Viewer, which displays the signals and values of the design and testbench over time. You can use this information to verify the functionality of the design and identify any issues that need to be resolved.
- Debug the design: If there are issues with the design, you can use the Vivado Debug tools to identify and fix the problems. These tools include the Source Code Debugger, which allows you to step through the design code and set breakpoints, and the Signal Probe, which allows you to monitor the values of specific signals in the design.
- Repeat the simulation: After you have made changes to the design or testbench, you may need to repeat the simulation to verify the changes have been made correctly. You can do this by re-compiling the design and running the simulation again.

- Overall, simulating a design in Vivado is an iterative process that involves creating a testbench, compiling the design, running the simulation, analyzing the results, and debugging any issues that arise.

For detailed understanding, refer to refernece1.

5.2 Step to upload into Artrix-7 Board

- To upload Verilog code to a Vivado Artix-7 board, you typically need to follow these steps:
- Create a new project in Vivado: Open Vivado and create a new project. Select the Artix-7 board as the target platform and specify the design parameters (e.g., clock frequency, I/O pins, etc.).
- Create a new Verilog file: In the project, create a new Verilog file and enter your design code.
- Simulate the design: Before uploading the design to the Artix-7 board, you may want to simulate it using Vivado's built-in simulation tools to verify its functionality.
- Synthesize and implement the design: Use Vivado's Synthesis and Implementation tools to convert the Verilog code into a bitstream file that can be uploaded to the Artix-7 board. This process involves synthesizing the design into a gate-level netlist and then mapping the netlist to the Artix-7 FPGA architecture.
- Generate the bitstream file: After synthesizing and implementing the design, you can generate a bitstream file that contains the configuration data for the Artix-7 FPGA. This file will be used to program the FPGA on the board.
- Configure the Artix-7 board: Connect the Artix-7 board to your computer using a programming cable and open the Hardware Manager in Vivado. Select the Artix-7 board as the programming target and program the FPGA using the generated bitstream file.
- Verify the design on the board: Once the programming is complete, the Artix-7 board should be running your Verilog design. You can use the board's LEDs, switches, and other peripherals to verify the design's functionality.
- Overall, uploading Verilog code to a Vivado Artix-7 board involves creating a new project, writing Verilog code, simulating, and synthesizing the design,

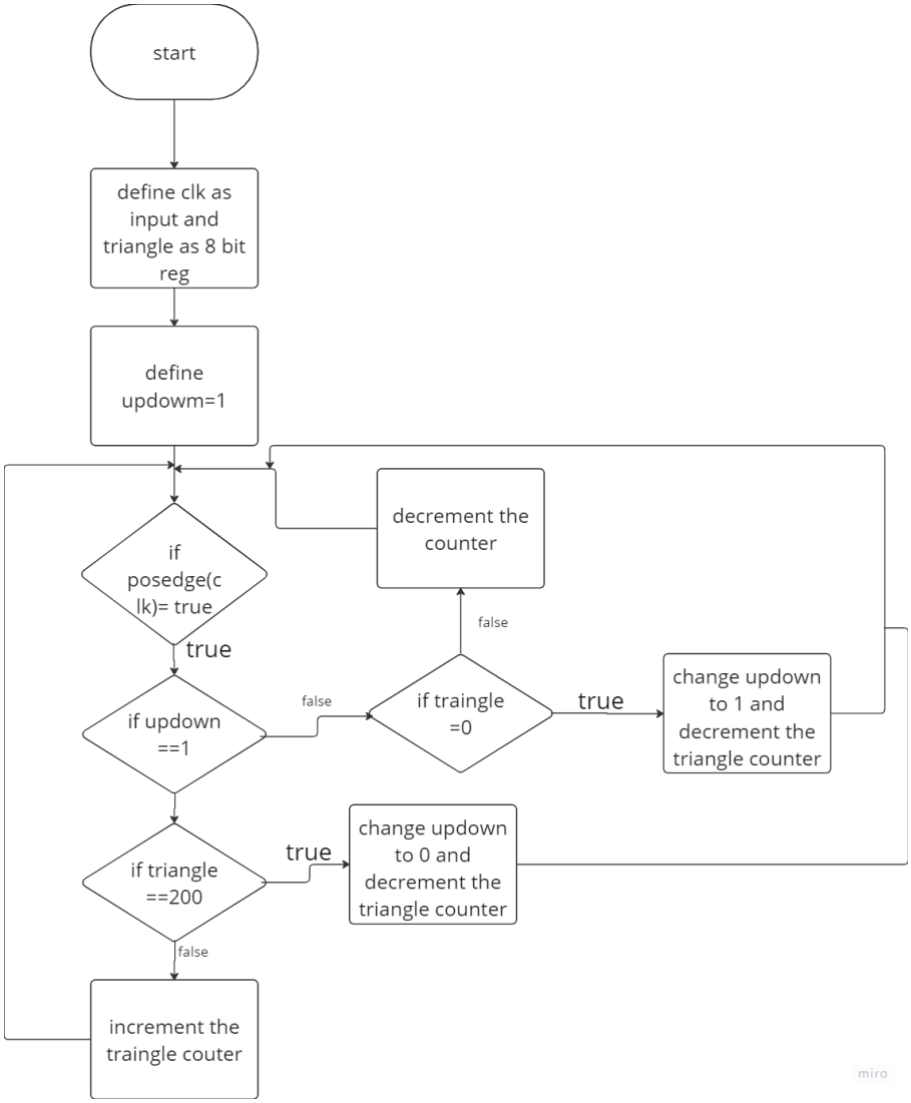
generating a bitstream file, programming the FPGA on the board, and verifying the design's functionality on the board.

5.3 Implementation of Triangular wave in FPGA

5.3.1 Algorithm

- Initialize the registers and internal clock pulse.
- Give two inputs in the form of data type “wire” i.e., clock pulse and reset value named as ‘cclk’ and ‘rstb’. Create a function named triangle using these parameters.
- Initialize the output register of 8 bit starting 0 to 7 that takes the initial value as zero.
- Initialise the up-down count register with values required i.e., 1 for incrementing up and 0 for decrementing down. This acts as status of the slope of signal.
- Now, begin the if-else loop with a condition that positive edge of the clock is initiated or triggered initially. i.e., //always(posedge(cclk))
- Initially the rstb is zero. If the rstb is zero initially, the triangle value (triangle is the amplitude value assumed) is incremented by one bit and the up counter is also incremented simultaneously.
- When the up-down counter is one and the triangle value reached the limit of 127, then the up down counter resets the value to zero and the triangle decrements by one bit. Or else the limit is not reached, then triangle is incremented.
- When the up down counter becomes zero and the triangle continues to decrement upto a value of zero. When the triangle reaches to zero, then the up down register again flips to 1 and the triangle starts incrementing again to reach the maximum limit.
- This process continues in an iterative loop manner.

5.3.2 Flowchart



5.3.3 Code in FPGA

```
`timescale 1ns / 1ps
`default_nettype none
//////////////////////////////////////////////////////////////////////////////////
//
// Module Name:    triangle
// Description:
//
//////////////////////////////////////////////////////////////////////////////////
module triangle(cclk, rstb,triangle);
        //port definitions
```

```
        input  wire cclk;
        input  wire rstb;
        output reg signed [7:0] triangle=0;

  reg updown=1; // 1 = going up, 0 = going down
  always @(posedge(cclk)) begin
    if (~rstb) begin
      triangle <= 8'b0;
      updown <= 1'b1;
    end else begin
        if (updown == 1'b1) begin
          if (triangle == 8'h80) begin
            updown <= 0;
            triangle <= triangle - 1;
          end else begin
            triangle <= triangle + 1;
          end
        end else begin
          if (triangle == 8'h00) begin
            updown <= 1;
            triangle <= triangle + 1;
          end else begin
            triangle <= triangle - 1;
          end
      end
    end
   end
endmodule
```

** The detailed code and source files are available Github , reference 2.

5.3.4 Vivado Result

The below given figure 5.3.1 is the corresponding result for the triangular wave run in the simulation.

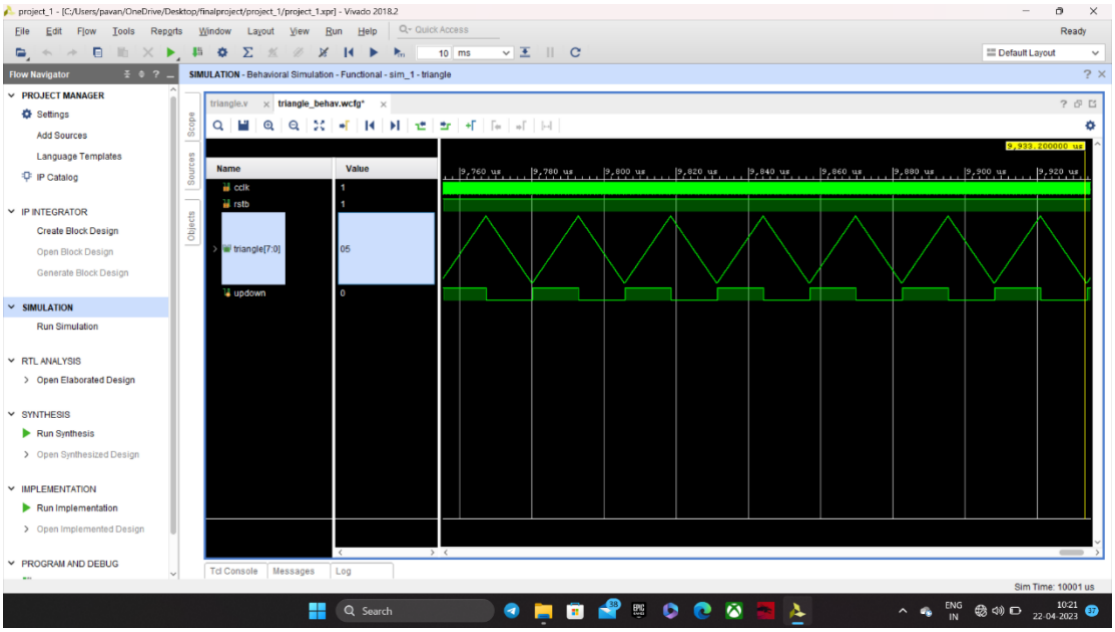


figure 5.3. 1 Triangular wave simulated output

5.4 Implementation of Sinusoidal Wave in FPGA

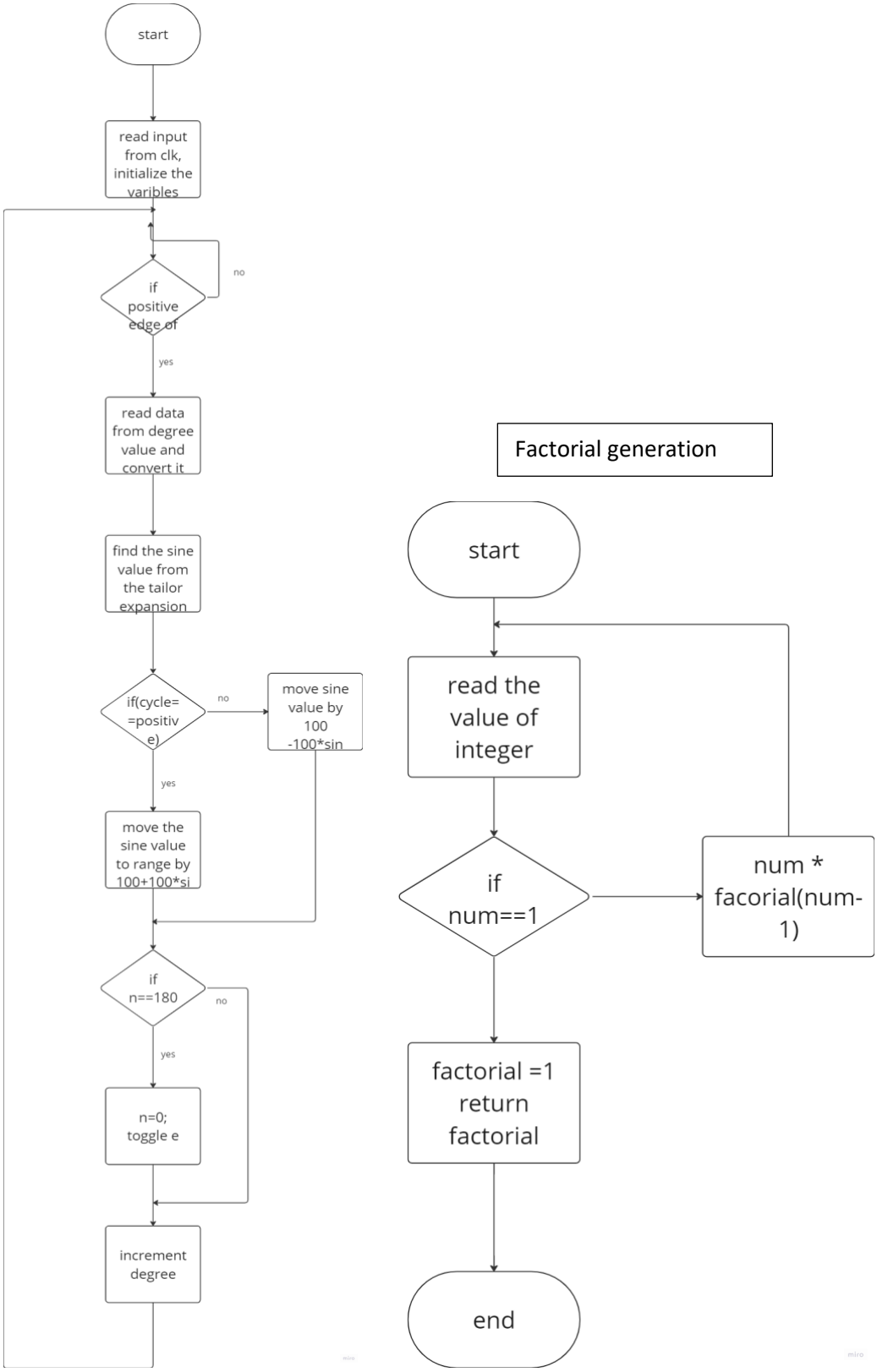
5.4.1 Algorithm

- Set the Verilog timescale to 1ns / 1ps.
- Define a module called "sine" with two ports: clk as the input clock signal and out2 as the output signal.
- Define an automatic function called "factorial" that takes an integer input "a" and returns a real value. This function calculates the factorial of "a" recursively.
- Define several variables: n as an integer initialized to 0, k as a real initialized to 0, p as a real initialized to 0, and e as a register initialized to 0.
- Define an always block that triggers on the positive edge of clk.
- Inside the always block, calculate the value of k as (3.14 * n) / 180. This converts the current value of n (which ranges from 0 to 180) into an angle in radians.
- Calculate the value of p using the sine function Taylor series. The series is evaluated up to the 9th term using k, and the resulting value is stored in p.

- If e is 0, add a DC offset of 100 to the value of p. Otherwise, subtract 100 from the value of p.
- If n has reached 180 (a full cycle of the sine wave), reset n to 0 and toggle the value of e.
- Set the value of out2 to p.
- Increment n by 1.
- End the always block.
- End the module definition.

Overall, this Verilog code implements a sine wave generator that uses a Taylor series approximation of the sine function to generate the output waveform. The DC offset of the waveform can be toggled by changing the value of the e register. The output waveform has a period of 360 clock cycles (180 cycles for the positive half, 180 cycles for the negative half).

5.4.2 Flowchart



5.4.3 Code in FPGA

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 19.04.2023 19:38:56
// Design Name:
// Module Name: sine
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module sine(input wire clk,output reg [7:0] out2);
function automatic real factorial (input integer a);
   begin
     if (a > 1) begin
       factorial = a * factorial(a - 1);
     end
     else begin
       factorial = 1;
```

```
    end
  end
endfunction
integer n=0;
real k=0;
real p=0;
reg e=0;
always@(posedge clk)

   begin
          k=(3.14*n)/180;
          p= (k) - ((k**3) /factorial(3)) +((k**5) /factorial(5))-((k**7) /factorial(7))+((k**9) /factorial(9));
    if(e==0)
     begin
       p=100+100*p;
     end
  else if(e==1)
      begin
        p= 100-100*p;
      end
    if (n==180)
          begin
          n=0;
          e=~e;
          end
          out2=p;
          n=n+1;
   end
   endmodule
```

5.4.4 Result in Vivado

The figure 5.4.1 illustrates the simulated output for the written Verilog ccode implemented in Vivado IDE.

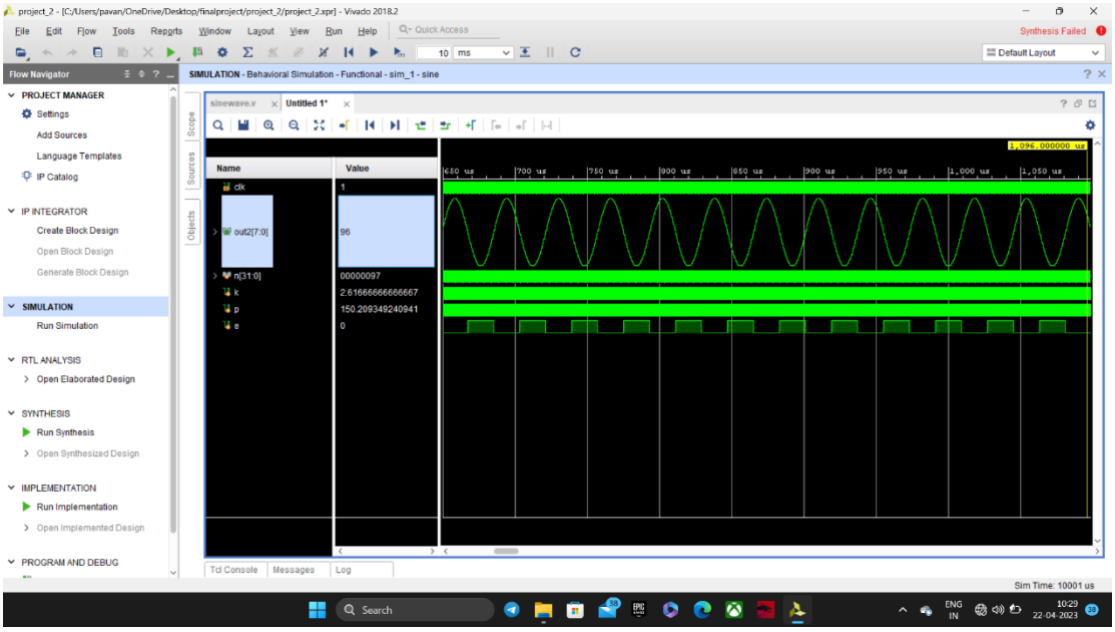


figure 5.4. 1 Sine wave output in FPGA simulation

5.5 SPWM wave

5.5.1 Algorithm

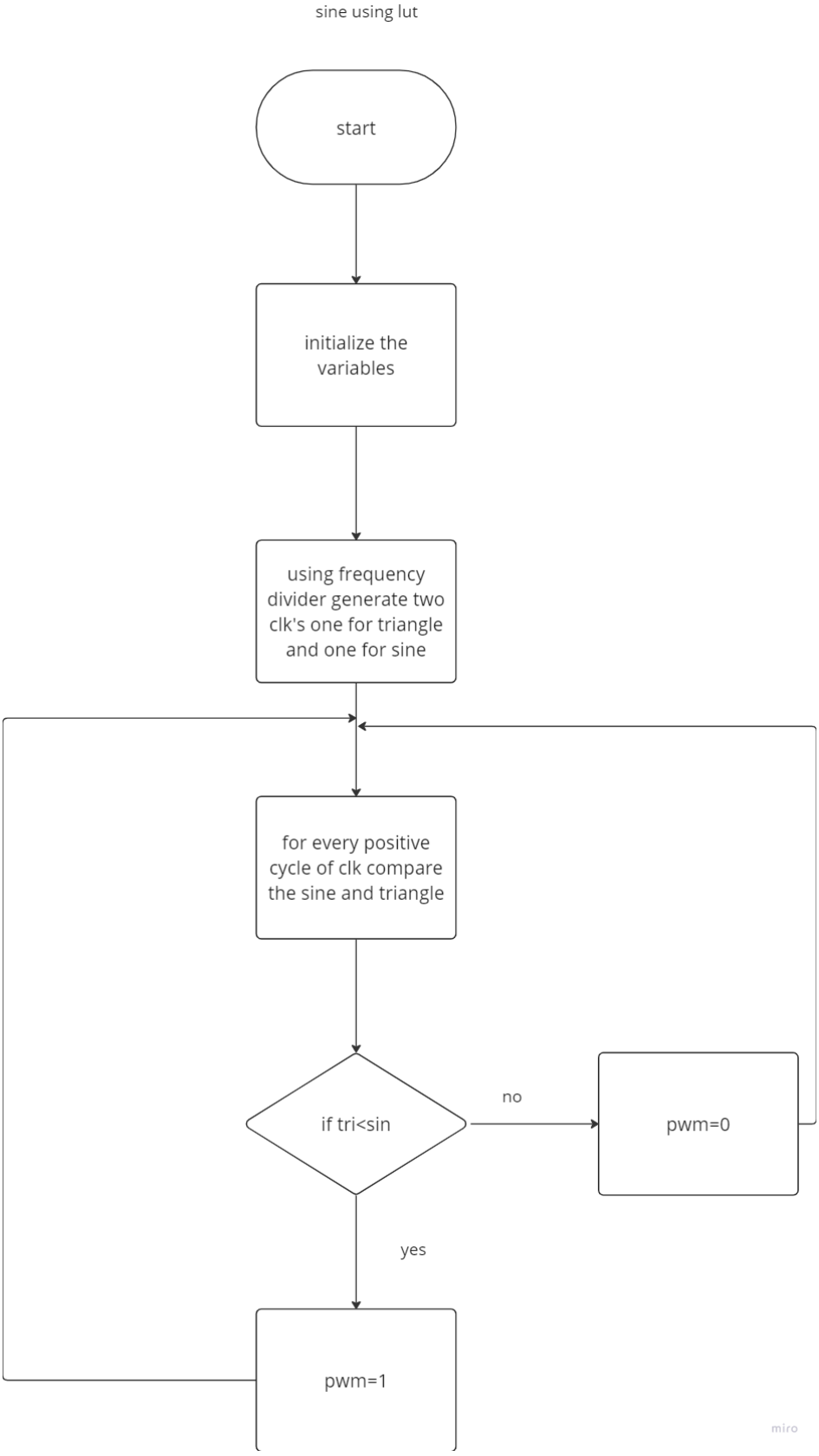
- Define a function named 'factorial' that takes an input integer 'a' and returns the factorial of 'a'.
- Declare and initialize some variables of type 'real' and 'reg' that will be used in the code later.
- Use an always block with a posedge clock to generate a clock output signal that is toggled every 10 clock cycles.
- Use another always block with a posedge of the clock output signal to generate a sine wave using the Taylor series approximation for sine. The 'p' variable represents the approximation of sine value at each angle 'k'. 'n' variable is used to increment the angle value by 0.5 degrees every clock cycle. 'e' variable is used to switch between positive and negative amplitude for the sine wave.
- Use another always block with a posedge of the clock signal to generate a triangular wave. The 'updown' variable is used to switch between incrementing

and decrementing the value of the triangular wave 'triangle'. The triangular wave value is incremented or decremented based on the 'updown' value.

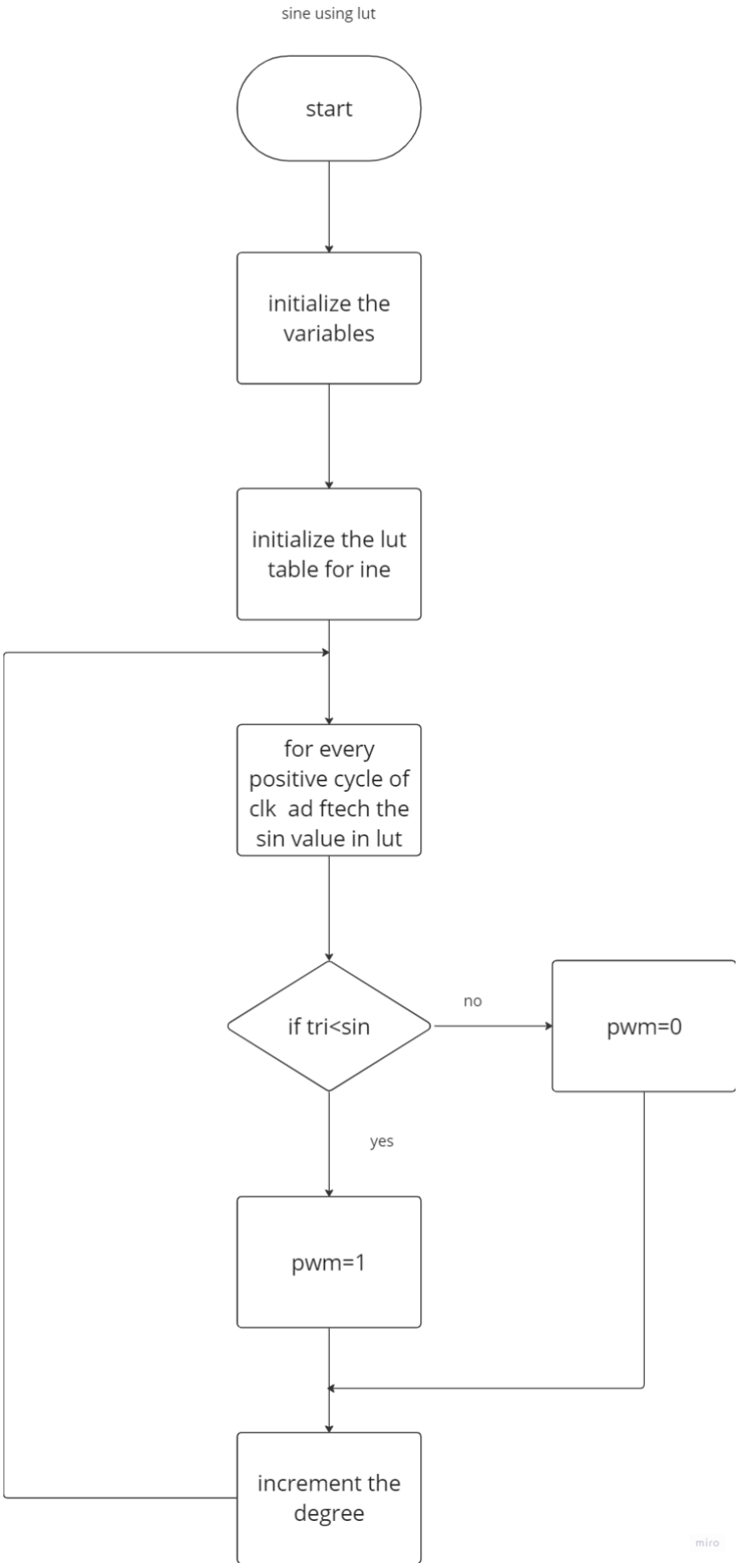
- Use a conditional statement to set the value of 'PWM' based on whether the triangular wave value is less than the sine wave value. The 'PWM' value is set to 1 if the triangular wave value is less than the sine wave value and 0 otherwise.
- Assign the values of the triangular wave 'triangle', the sine wave 'sin', and the PWM signal 'PWM' to the corresponding output ports.
- End the module definition.

Overall, the Verilog code generates a triangular wave and a sine wave using separate always blocks. The triangular wave is used to generate a PWM signal based on whether its value is less than the value of the sine wave. The sine wave is generated using the Taylor series approximation and the angle value is incremented every clock cycle. The triangular wave value is incremented or decremented based on the 'up down' variable, which switches between 0 and 1 every time the triangular wave reaches its minimum or maximum value. The clock output signal is used to toggle an LED or to drive other signals.

5.5.2 Flowchart



SPM using LUT method:



5.5.3 Code in FPGA

```
`timescale 1ns / 1ps

module sine(input wire clk,output reg  pwm,output reg [7:0] triang,output reg [7:0] sin);

function automatic real factorial (input integer a);
  begin
    if (a > 1) begin
      factorial = a * factorial(a - 1);
    end
    else begin
      factorial = 1;
    end
  end
endfunction
real triangle=0;
real out2=0;
real n=0;
real k=0;
real p=0;
reg e=0;
reg updown=1;
reg clk_out=0;
reg [15:0] counter=0;
always@(posedge clk)
begin
```

```
 if(counter==16'd10)
 begin
 counter<=16'd0;
 clk_out <= ~clk_out;
 end
 else
 begin
 counter<=counter+1;
 end
 end

 always@(posedge clk_out)

        begin
                k=(3.14*n)/180;
                p= (k) - ((k**3) /factorial(3)) +((k**5) /factorial(5))-((k**7) /factorial(7))+((k**9) /factorial(9))-((k**11) /factorial(11));
          if(e==0)
        begin
          p=100+100*p;
        end
     else if(e==1)
        begin
          p= 100-100*p;
        end
```

```
            if (n==180)
                    begin
                    n=0;
                    e=~e;
                    end
                    out2=p;
                    sin=out2;
                    n=n+0.5;
end
  always@(posedge clk)
  begin
     if (updown == 1'b1) begin
        if (triangle == 8'hcb) begin
          updown <= 0;
          triangle <= triangle - 1;
        end else begin
          triangle <= triangle + 1;
        end
end
else begin
        if (triangle == 8'h00) begin
          updown <= 1;
          triangle <= triangle + 1;
        end else begin
```

```
            triangle <= triangle - 1;
          end
     end
   triang=triangle;
if(triangle < out2)
begin
  pwm=1;
end
else begin
pwm=0;
end
end
endmodule
```

5.5.4 Vivado Result

The below figure 5.5.1 illustrates the simulated output generated when the above code is implemented in Vivado IDE.

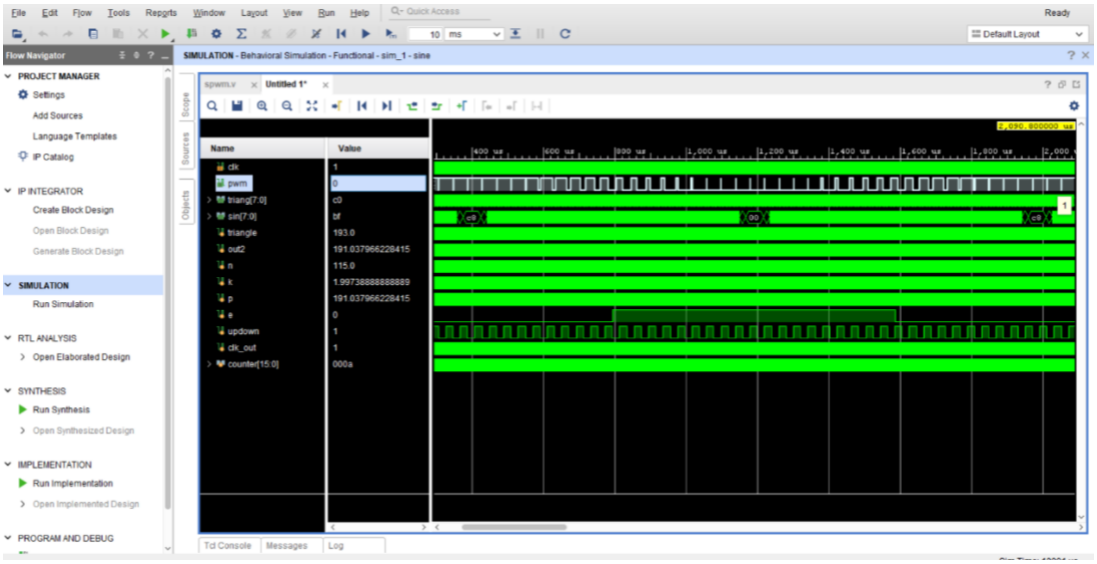


figure 5.5.1 Sinusoidal PWM output simulation

CHAPTER-6

HARDWARE IMPLEMENTATION ON FPGA BOARD

6.1 Real time simulation in ARTIX-7 Microcontroller

The Xilinx Artix-7 FPGA is a high-performance, low-power FPGA that offers a range of features including high-speed serial connectivity, DSP performance, and low-cost integration. The Artix-7 FPGA is ideal for a wide range of applications such as communication systems, aerospace and defense, industrial automation, and scientific instrumentation. The Aryabhata microcontroller is built around the Xilinx Artix-7 FPGA and includes a range of peripherals such as GPIO, UART, SPI, I2C, and Ethernet. The microcontroller also includes an ARM Cortex-M3 processor, which provides processing power for embedded applications. Figure 6.1.1 shows the hardware architecture of Artix -7 Aryabhata Board of Vanix solutions.

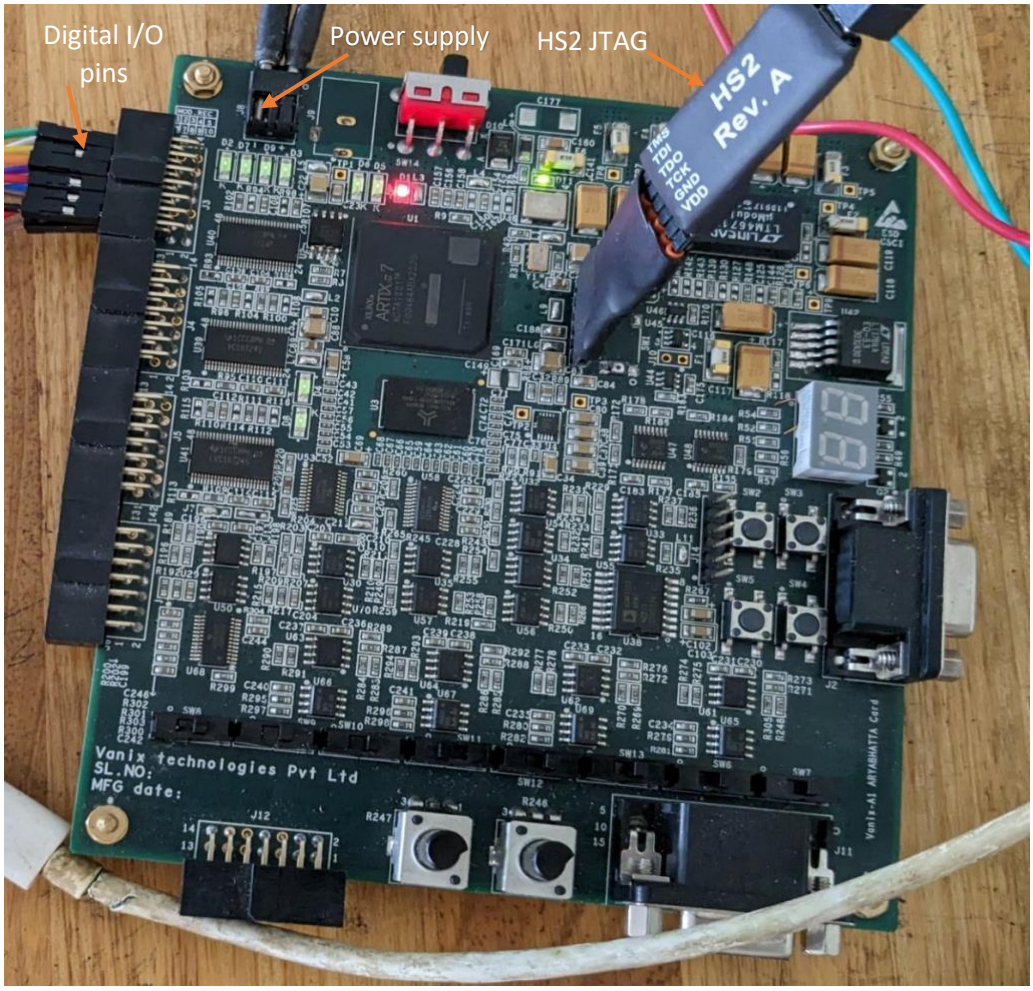


Figure 6.1. 1 Artix-7 Aryabhatta Board

The Verilog code debugged from the Vivado IDE is synthesized and uploaded into the Artix-7 Board using the USB-cable, a voltage of 5V is given through Vcc and Gnd pins and the CRO probes are connected using digital pins. The resultant Sinusoidal pulse width modulated signal in whose carrier frequency is five times the sinusoidal wave frequency which is shown in figure 6.1.2.

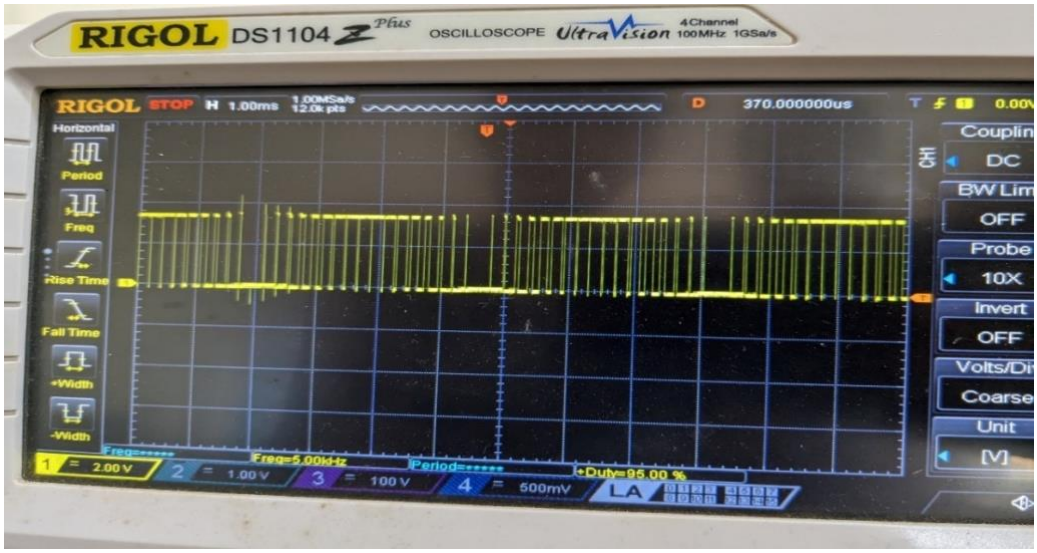


Figure 6.1. 2 illustrates the SPWM output signal in CRO

The FPGA algorithm is very flexible with the frequency so that it can be used for various range of modulated signals. The below figure 6.1.3 shows the carrier frequency eight times the sinusoidal wave frequency.

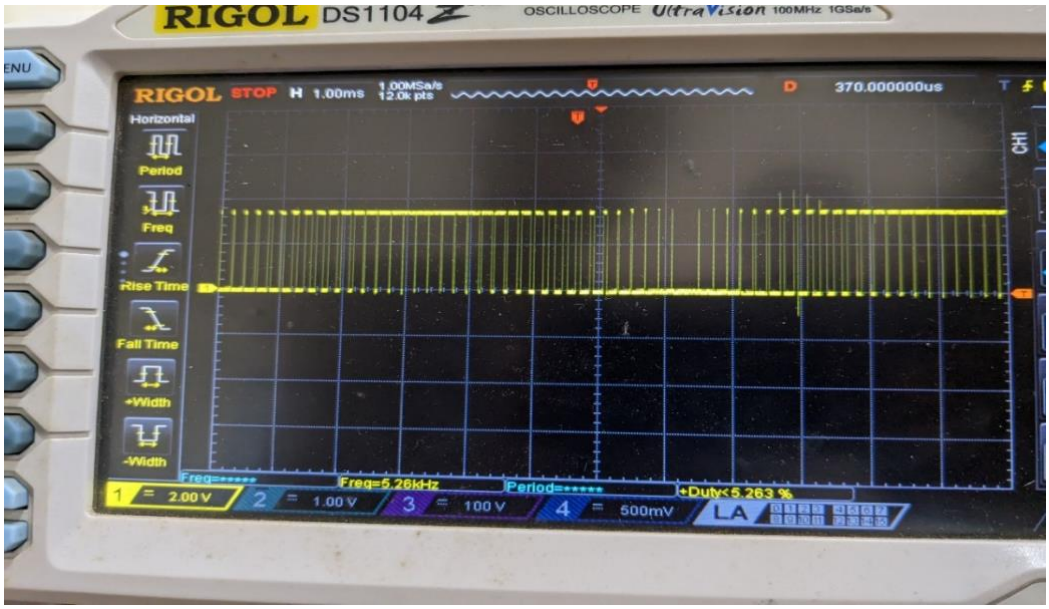


Figure 6.1. 3 SPWM signal in CRO with carrier frequency 8 times the reference

The hardware result is visible, and can be used further for other applications, but the individual carrier and reference waves cannot be analysed in the CRO screen or any analog display as the algorithm generates the digital bit data. To encounter this situation, Digital to analog converter is used. This can also be done by using the logging the output data into any serial port environment. This gives the user proper control of comparability, and it also increases efficiency as the user has the provision to analyse all the signals all at once with high precision.

While interfacing the DAC converter with the Artix-7 Board using SPI communication, an error generated i.e., due to the poor architecture of Artix-7 whose pins are not configured separately for the DAC interfacing. The formal method of pin testing of Artix-7 board consumes time. The second approach of analysing the signals is using external Serial port communication as there is no available internal console logs or CAN communication. Due to the inherited baud rate difference that was limited to the microcontroller, the serial port logging gives you disrupted results. The below figures 6.1.4 and 6.1.5 shows the serial logged deviated results of individual and triangular and sine waves respectively.

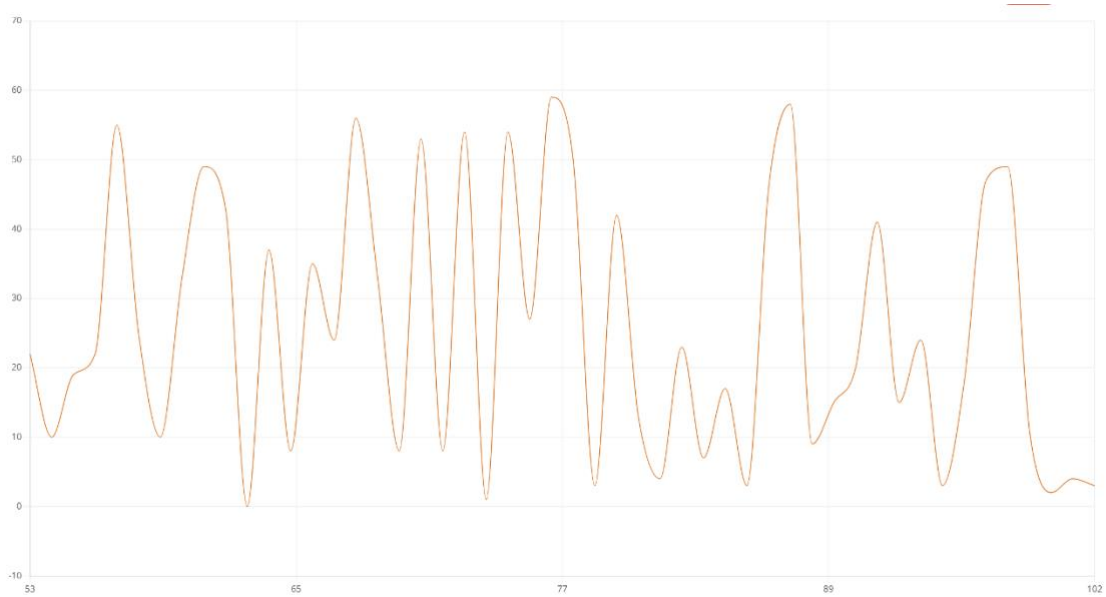


Figure 6.1. 4 shows the disrupted signal of triangular wave in Putty

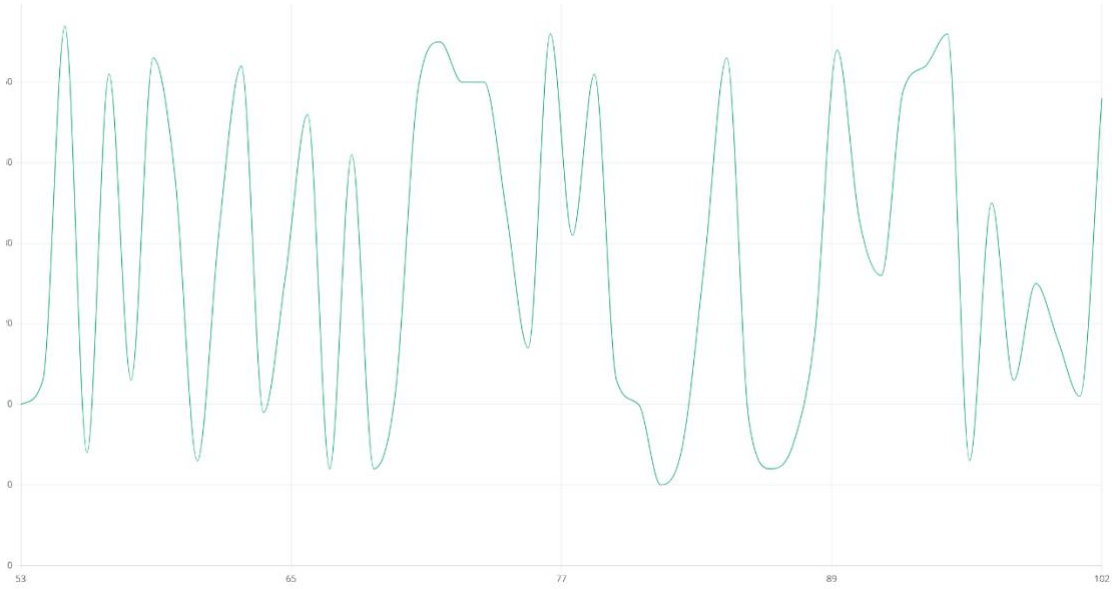


Figure 6.1. 5 disrupted signal of Sine wave in Putty

The below algorithm shows the steps to interface with the digital to analog converter using SPI (serial peripheral interface).

- Declare a module named sine with an input wire clk and an output register pwm.
- Declare three registers, sine, triangle, and sin, with a width of 8 bits.
- Initialize a counter, n, to zero and a flag, updown, to 1.
- Declare a flag, clk_out, and a 16-bit counter, counter, both initialized to zero.
- Initialize the first five values of the sine array.
- Implement an always block for the counter that increments the counter by 1, and when the counter reaches 10, it sets the counter to zero and toggles the clk_out flag.
- Implement an always block for the clk_out flag that resets the n counter to zero when it reaches the maximum value of 720, assigns the value of sine[n] to the sin variable, and increments the n counter by 1.
- Implement an always block for the clk flag that controls the up-down counter. If the updown flag is set to 1, increment the triangle counter until it reaches the maximum value of 8'hcb. Then, set the updown flag to 0 and decrement the triangle counter until it reaches the minimum value of 8'h00. If the updown flag is set to 0, decrement the triangle counter until it reaches the minimum value of 8'h00. Then, set the updown flag to 1 and increment the triangle counter until it reaches the maximum value of 8'hcb.

- Use an if-else statement to compare the triangle value with the sin value. If the triangle value is less than the sin value, set the pwm flag to 1, else set it to 0.
- End the module.

6.2 Usage of DAC (Digital to Analog Converter)

As in our FPGA board the interfacing of the dac and the main Processor using the SPI(Serial Peripheral Interface) protocol. It involves transmitting the data serially using the SPI protocol. the one used is DAC8554.

6.2.1 SPI

In order to use the SPI protocal there are mainly four pins sck(serial clock), cs(chip select),mosi(master output slave input), miso(master input slave output). The communication starts by the master by giving the pulse to slave using the sck pin and then selecting the chip using the cs pin . it depends on the chip that giving low will select the pin and giving high on the pin will select the chip.

The data is transmitted with the clk synchronisation.as it does not have start and stop bits but bit should transmit from msb to lsb only.

6.2.2 DAC Working

The main architecture is that it has a resistor string DAC followed by an output buffer amplifier

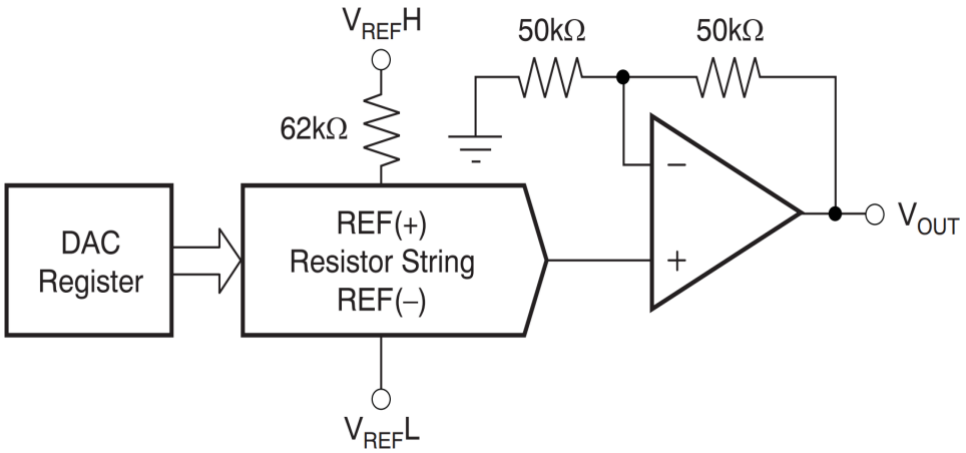


figure 6.2.2. 1 internal schematic circuit diagram of DAC

The resistor string is a simple divide-by-2 resistor followed by a string of resistors . the code is loaded from the data register. The applied voltage is then applied at the output voltage.

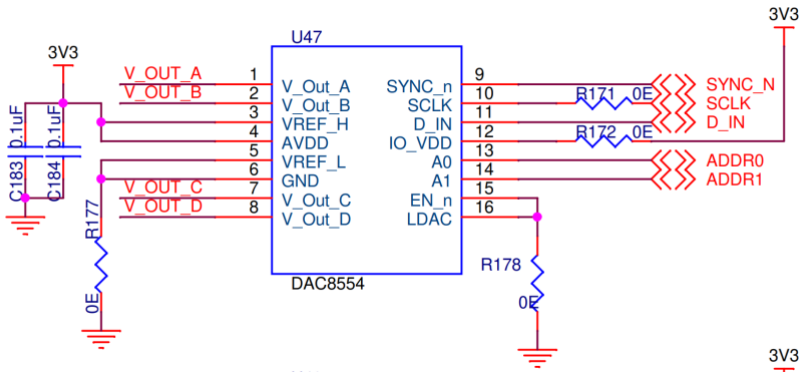


figure 6.2.2. 2 internal architecture of DAC

And the SYNC_N will act as the chip select and sck is the serial clock and din as the Mosi as slave will not transmit any data to the master miso is not given as spi. The master initiates the transfer of data..

The data in order must be from MSB (most significant bit) to LSB (least significant bit).

The following is the order of the bits to be transmitted.

A1- A0 -LD1- LD0- X- DAC Select 1- DAC Select 0 -PD0- D[15:0]

The PD0 tells about the power down mode. And the DAC select will selec the which dac to activate.

And LD0-LD1 will tell the channel update (00-single channel store, 01- single channel update, 10- simultaneous mode, 11- broad cast update).

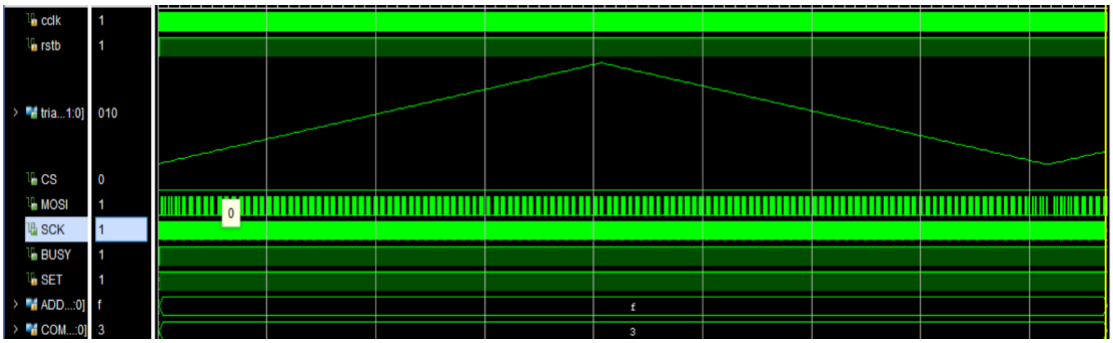


figure 6.2.2. 3 Ouput simulated signal is stored in MOSi

CHAPTER-7

CONCLUSION & FUTURE WORK

7.1 Conclusion

This Sinusoidal Pulse Width Modulation generator model is initially aimed at briefly on the basic analog design. Simulation overcomes the chances of connection failures and flexibility in analysis. Multisim played an important role in making the ideology of analog design using OP-Amps successful. This was further proceeded to the second level hardware language i.e., FPGA (Field Programmable Gate Array). In FPGA, the generation of sinewave and triangular carrier wave are also implemented using tailor series expansion and counter incrementing algorithms respectively. The SPWM signal is also generated and observed in the simulation graph. While synthesizing the Verilog code of FPGA Vivado to the Artix-7 Aryabhata Board, Debugger ran with error in Microcontroller because of the "real" data type which is not in the data size of the Artix-7. The significant shift to generation of lookup table overcame the error from the previous approach. This gave the desired results for the output Sine wave Pulse Width Modulated signal in the CRO (Cathode Ray Oscilloscope). However, the individual carrier and modulated signals cannot be observed in the CRO. The reason for this error is the inherited baud rate difference between the serial port and the microcontroller besides the failure of DAC. To overcome this problem, the frequency should be decreased to very low levels. This can also be overcome by using the external digital to analog converter and high baud rate serial communication portal.

7.2 Future Scope

This model can be still developed by taking the motives i.e., Integrating the entire blocks of circuit on single IC. The logging data method can also be improved by setting the efficient baud rate for the communication. It can be designed in high level hardware implementation using VLSI. This design can also be implemented by synthesizing the circuit model in FPGA by replicating the working of analog Op-amp circuit.

References

1. Design, M.B.D., 2012. Vivado design suite user guide.

2.https://github.com/Jithuan91/B.-Tech_project

3.Ilhami Colak , Ersan Kabalci," Developing a novel sinusoidal pulse width modulation (SPWM) technique to eliminate side band harmonics"

4.Matthew C Trigg , Hooman Dehbonei, C. V. Nayar," Digital Sinusoidal PWM Generation using a Low-cost Micro-controller Based Single-Phase Inverter"

5.RishirajSarker, AsimDattab,SudiptaDebnatha," FPGA-based variable modulation-indexed-SPWMgenerator architecture for constant-output-voltage inverter applications"

6. DAC8554 manual by Texas instruments

7.Quan, J., Sun, Y., & Shu, H. (2019). FPGA-Based Real-Time Implementation of Sinusoidal Pulse Width Modulation for Motor Control Applications. IEEE Transactions on Industrial Electronics, 66(11), 8865-8874.

8. Zhang, X., Lin, W., & Shao, H. (2017). FPGA-based design and implementation of sinusoidal pulse width modulation. International Journal of Circuit Theory and Applications, 45(5), 679-695.

9. Masoudi, H., Soltani, M., & Khajehpour, A. (2017). FPGA-based real-time implementation of a three-phase inverter with sinusoidal pulse width modulation. International Journal of Electrical Power & Energy Systems, 92, 111-122.

10.Afzalian, A., & Niazi, M. (2019). FPGA-Based Real-Time Implementation of Sinusoidal Pulse Width Modulation Technique for Three-Phase Inverters. Journal of Electrical and Computer Engineering Innovations, 7(3), 263-272.

11. Nasiri, M., Gholamzadeh, M., & Lashgari, R. (2016). FPGA-based implementation of SVPWM for three-phase voltage source inverters. In 2016 7th Power Electronics, Drive Systems & Technologies Conference (PEDSTC) (pp. 333-338). IEEE.

BIO DATA

NAME : ANKIREDDY NAGA JITHENDRA REDDY

DATE OF BIRTH : 21-09-2001

NATIONALITY : Indian

EMAIL : jithendra92911@gmail.com

CONTACT : 9121952521

NAME : VEESAM PAVAN KRISHNA

DATE OF BIRTH : 16-06-2001

NATIONALITY : Indian

EMAIL : pavankrishna2321@gmail.com

CONTACT : 6309639277

NAME : AVUNURI KOWSHIK

DATE OF BIRTH : 31-12-2001

NATIONALITY : Indian

EMAIL : avunurikoushik4575@gmail.com

CONTACT : 8106367650