People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

University M'Hamed BOUGARA – Boumerdès

Institute of Electrical and Electronic Engineering
Department of Electronics

Final Year Project Report Presented in Partial Fulfilment of

the Requirements of the Degree of

'MASTER'

In Electrical and Electronic Engineering

Option: **Computer Engineering**

Title:

Hardware-In-The Loop simulation of Inverter's Control Unit based on OPAL-RT and FPGA

Presented By:

- **BOUYAHIAOUI Hadjer**

Supervisor: **Dr. AZZOUGUI Y.**

Co-Supervisor: **Dr. BENMERABET M.S.**

Registration Number:……/2023

• Abstract:

Due to the increasing cost of power electronics switches and the cost of repairing failures of power electronics systems, semi-physical simulation technology such as hardware-in-the-loop (HIL) simulation is increasingly being used as an important design, development, and testing step in the manufacturing process of many power electronics systems. In this project, HIL simulation for an FPGA-based controller of a three-level power inverter was performed and tested using the OPAL-RT HIL simulation tool. First, an FPGA-based controller for a single-phase three-level inverter was designed and tested. Three different PWM digital control circuits (Bipolar SPWM, Unipolar SPWM, and SHEPWM) were designed from scratch using VHDL. Then Hardware-in-The-loop testing of the controller was implemented using the RT-LAB HIL testing platform. To emulate the three-level inverter, a simulation model based on MATLAB®/Simulink software and Simscap blocks was designed to perform real-time simulation testing of the FPGA-based controller. The obtained test results were compared to pre-simulated results using MATLAB®, the results have shown that the total harmonic distortion (THD) of the difference controlling circuits decreases gradually and thus helps improve the power quality of the inverter, the final HIL results were acceptable and close to the expected and real results especially with the SHEPWM where we obtained errors less than 7% between expected and HIL results.

Keywords: Pulse Width Modulation (PWM), Total Harmonic Distortion (THD), Hardware In the Loop (HIL), Unipolar & Bipolar Sinusoidal Pulse Width Modulation, SHEPWM, VHDL, FPGA, Inverter.

• Acknowledgement:

In the name of Allah, the Most Gracious and the Most Merciful

Alhamdulillah, all praises

to Allah for the strengths and His blessing in completing this project.

I would like to express my deepest and sincere gratitude to my project Supervisor

Dr. AZZOUGUI Yasmina, and Co-Supervisor Dr. BENMERABET Mohamed Salim, it was a great privilege and honor to work and study under your supervision. Thank you very much.

I would give special thanks to Mrs CHIKH Wafa and Mr BOUKERDOUN Haithem.

Finally, I am infinitely grateful to my family members, particularly my parents and my aunt Hafida DEBBAGHI for their patience, unwavering support, continuous encouragement, and belief in me throughout my whole life. I would have never made it this far without them beside me every step of the way.

- Dedication:

I have a great pleasure to dedicate this modest work
To my Dear Father, my Beloved Mother,
To my Dear Sisters, Brother, Aunts, Uncles, and Cousins
To all my Friends
And to all with whom I spent wonderful moments
And I do appreciate you taking the time to read this work, so my dedication is to you.

Hadjer BOUYAHIAOUI

- Table of Content:

- # List of figures:

- # List of Tables:

- # Abbreviation:

A

ASIC: Application Specific IC.

C

CHB:Cascaded H-Bridge.

CPU: Centra Processing Unit.

CSI: Current-Source Inverters.

D

DLL: Dynamic Link Library.

DUT: Device Under Test.

E

EDA: Electronic Design Automation.

F

FC: Flying Capacitor.

FIL: FPGA-In-the-Loop.

FPGA: Field Programmable Gate Array.

G

GA: Genetic Algorithm.

GUI: Graphical User Interface.

H

HDL: Hardware Description Language.

HIL: Hardware In the Loop.

HMI: Human Machine Interface.

I

IC: Integrated Circuit.

L

LE: Logic Elements.

M

MFO: Moth Flame Optimization.

MIL: Model in the Loop.

MLI: Multilevel Inverter Topology.

N

NPC: Neutral Point Clamped.

NR: Newton-Raphson.

P

PHIL: Power Hardware-in-the-Loop.

PIL: Processor in the Loop.

PWM: Pulse-Width Modulation.

R

RCP: Rapid control prototyping.

RE: Renewable Energy.

RTS: Real-Time Simulation.

S

SHEPWM: Selective Harmonic Elimination Pulse Width Modulation.

SIL: Software In the Loop.

SoC: System on Chip.

SPF: Small Form-factor Pluggable.

SPWM: Sinusoidal Pulse Width Modulation.

T

THD:Total Harmonic Distortion.

U

UPS: Uninterruptible Power Supply.

V

VHSIC: Very High-Speed Integrated Circuit.

VHDL: VHSIC Hardware Description Language.

VSD: Variable Speed Drive.

VSI: Voltage-Source Inverters.

Z

ZSI: Impedance-Source Inverters.

• General Introduction:

Renewable energy is becoming increasingly important as we move towards a more sustainable future. Power electronics play a vital role in the generation, transmission, and distribution of renewable energy. Inverters are a type of power electronics device that converts DC power to AC power. They are used in a variety of applications like Uninterruptible Power Supply (UPS), Variable Speed Drive (VSD) and in renewable energy (RE) such as solar photovoltaic systems, wind turbines, and electric vehicles.

The control of inverters is a complex task. The controller must ensure that the inverter output voltage and frequency are accurately controlled, even under varying load conditions. Hardware-in-the-loop (HIL) simulation is a powerful tool that can be used to test and validate inverter controllers. HIL simulation allows the controller to be tested in real time with a simulated inverter model. This can help to identify and correct controller design flaws early in the development process, which can save time and money.

The objective of this report is to investigate the use of HIL simulation for testing and validating inverter controllers. The specific focus of the report is on the development and testing of an FPGA-based controller for a three-level power inverter. The controller was designed using VHDL and tested using the OPAL-RT HIL simulation tool. The results of the HIL simulation were compared to pre-simulated MATLAB® results and expected results. The results showed that the controller was able to control the inverter output voltage and frequency under varying load conditions.

In the first chapter, we provided a brief overview of the theoretical background of power electronics and inverter technology. We also discussed the different modulation topologies for 2-level and 3-level voltage source inverters and the different real-time simulation methods. In the second chapter, we described the hardware used in the HIL simulation, we also discussed the simulation setup and the results of the MATLAB® simulation. In the third chapter, we described the process used to implement the FPGA-based controller and present the results of the HIL simulation and discussed them and compared them with the pre-simulated results and finally we present a general conclusion based on the results.

Chapter I. Theoretical Background and Literature Review

I.1. Introduction:

Power Electronics is a field in electronical engineering that deals with conversion and control of electrical power with the help of switching devices, it involves several theoretical aspects and disciplines including semiconductor physics, control theory, power systems and circuit principles. [1]. There are four basic possible conversions: AC/DC also known as rectifier, AC/AC, DC to AC or inverter and DC to DC conversion. The range of applications for power electronics converters is vast, including motor drive systems, renewable energies, robotics, electrical and hybrid vehicles as well as circuits promoting power quality. -Figure I-1.1 illustrates this by analogy-. Testing and validation of power converters are necessary during the design and improvement phases of these electrical systems and before final implementation on actual real processes. The Inverter is one of the power converters that are widely used in modern electrical systems. (Standalone-Inverters, Motor drives) In order to achieve the required performances of the power converter system and its control: real-time simulation technology is a prominent tool extensively used, aimed at testing functionality and performance of the electrical systems.

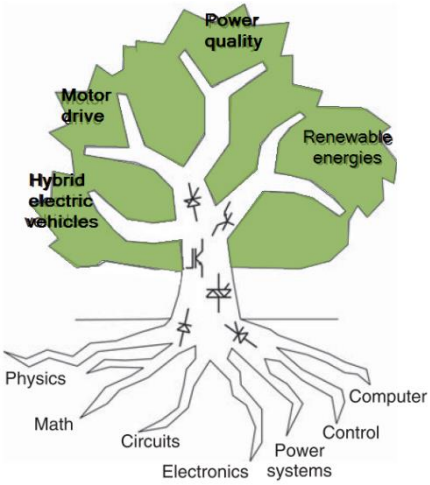

Figure I-1.1 Interdisciplinary nature and applications obtained from power electronics. [1]

In this chapter, we will initiate an examination of power electronics and inverter technology. Our attempt will involve mapping briefly the domain of inverters, covering their classification and operational principles. We will explore a diverse range of inverter structures, configurations, modulation techniques and control strategies as well as real time simulation techniques widely used for testing systems functionality.

I.2. Overview of power electronics and inverter technology:

I.2.1. History of Power Semiconductor Switches and Power Electronics:

The history of power electronics can be traced back to the early 1902, with the development of the mercury-arc rectifier by Peter Cooper Hewitt, despite being large and bulky as shown in Figure I-2, they were used in many applications, including power supplies, electric motors, and lighting. it was the only way to convert high AC power to DC power, two years later until the invention of semiconductor switches, in the 1950s, the first semiconductor power switches were developed, they were much smaller and more efficient than the mercury-arc rectifier, and paved the way for the development of modern power electronics [1].

Figure I-2: Mercury arc rectifier under operation [34]

The development of semiconductor switching devices is essentially a search for the ideal switch. The efforts have been made through years to reduce device power losses, increase switching frequencies and simplify gate drive circuits [2]. some of the key events in the history of power electronics are described in the following timeline in Figure I-3:

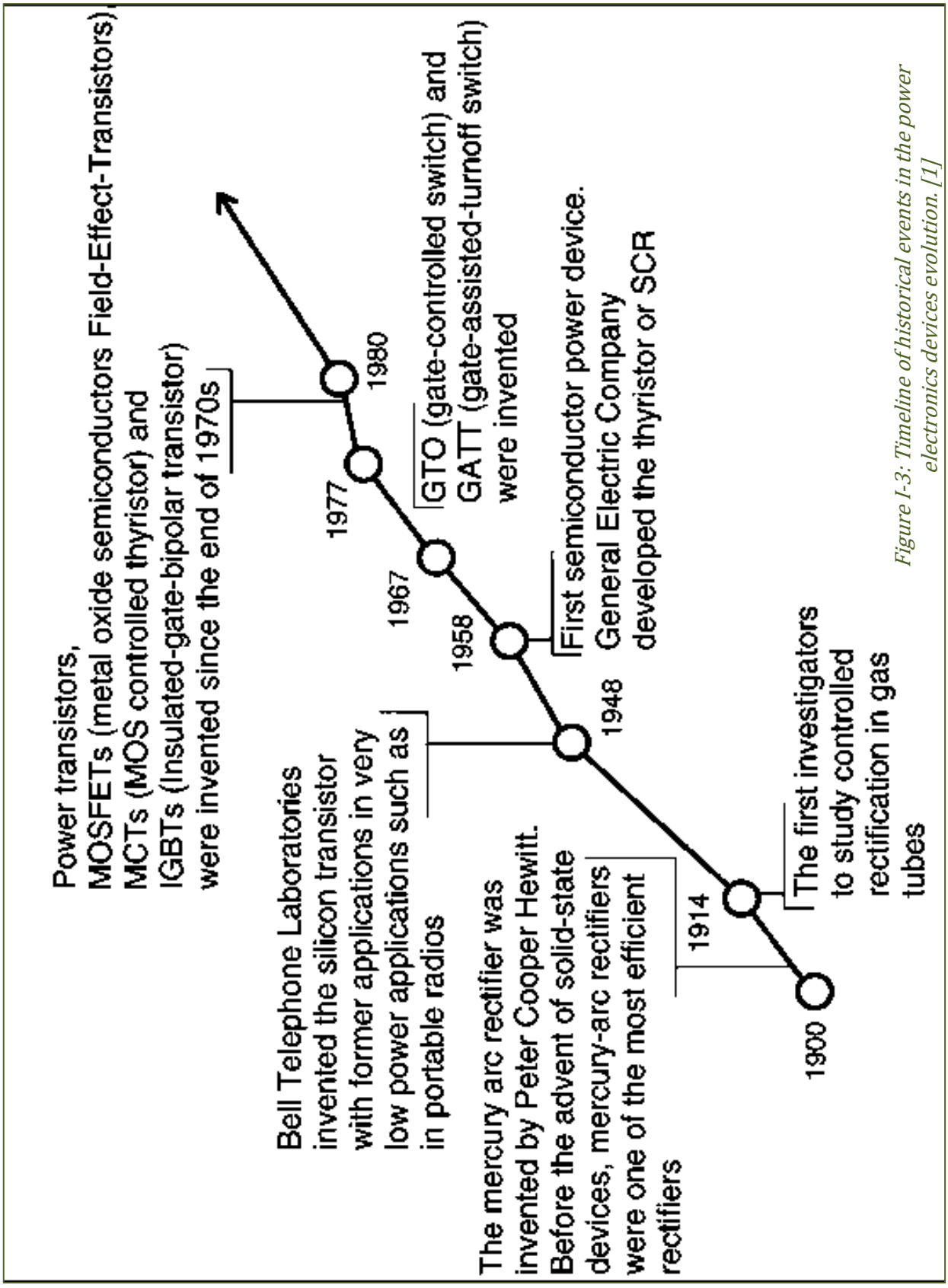

Figure I-3: Timeline of historical events in the power electronics devices evolution. [1]

I.2.2. Inverters:

The inverter also known as DC -AC converter, is a one type of the power electronics that converts DC power to AC power at desired output amplitude and frequency, inverters can be voltage-source inverters (VSIs), current-source inverters (CSIs) and impedance-source inverters (ZSIs) [1]. Figure I-4 represents a typical block diagram representation of VSI, CSI and ZSI, they are mostly used in AC power supplies (power inverters), motor drives (VFD) and systems where the objective is to produce a sinusoidal AC output whose magnitude and frequency both can be controlled. Practically, inverters are used in both single-phase and Multi-phase AC systems.

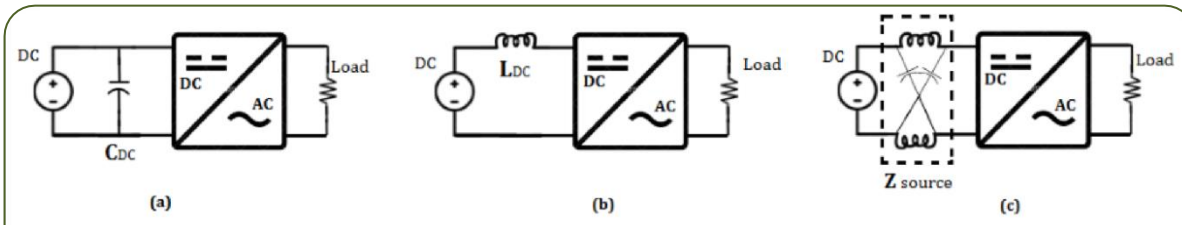

Figure I-4: (a) Typical voltage-source inverter block diagram and (b) current–source inverter block diagram (c) [30]Z-source inverter [1] indicating input energy storage elements

Inverters are available on the market in various configurations and power ratings. Figure I-5 illustrate an abstract representation of inverters in a system where: element 1 represent the DC power source that feeds the inverter, it could be a battery, a rectifier or a DC power supply.

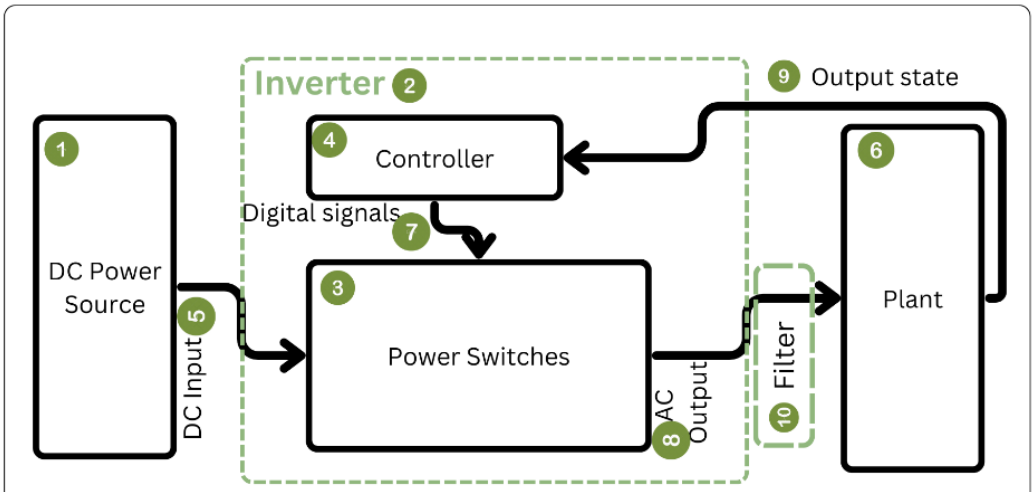

Figure I-5: Block diagram of a DC-AC within a system in an abstract point of view.

Element 2 is the inverter, in within element 3 represents a set of power semiconductor switches that are connected together in specific structure to direct the flow of DC current,

there exist numerous different structures and topologies, some will be discussed later in this chapter, these power semiconductor switches are controlled via element 4 the controller of the inverter, this last generates digital signals based on different control and modulation techniques aimed to turn ON or OFF the power semiconductor switches, these digital signals are called control signals, gating signals and control commands interchangeably in this scope. The controller can be available as a separate off-the-shelf module as 'converter's control unit' for specific application, as it could be a circuit integrated internally within the system, these controllers can be single Integrated Circuit (IC) chips that generate the control commands (either ASIC or FPGA-based controller) like the TL494 PWM IC, as they can be a SoC (System on Chip) with a microcontroller that generates the command signals, integrated with safety circuits and Human Machine Interface (HMI) to manipulate the output signals..etc. The rest of the element are summarized in Table I-1 bellow:

Table I-1:Inverter main component

Element	Name	Description	Example
5	DC input	The Dc input is the DC power that feeds the inverter and will be converted to AC power	220Volts, 350Volts
6	The output load (plant)	The characteristics of the load can greatly affect the performance of the inverter. For instance, resistive loads (like incandescent lights or heaters) are the easiest for an inverter to handle. However, reactive loads (like motors or transformers found in refrigerators) can be more challenging due to their power requirement.	Resistor, incandescent lights, AC motor
7	Digital signals	These digital signals control the switches, they are crucial for the operation of the inverter, they turn high or low depending on the type of the modulation controlling method used.	Pulse Width Modulation,
8	AC output	This output is the desired AC signal which have been converted from a DC signal.	Square wave, PWM signal
9	Output state	Feedback signal, generally used to control actions needed to keep a specific variable of the inverter under control, for instance, in a motor drive system where the speed must be controlled electronically through the inverter. This signal requires sensors to be measured	Phase of the AC signal state,
10	Output filter	The output filter removes the high-frequency components of the PWM wave, to produce a nearly sinusoidal output.	Low pass LC filter

I.2.3. Basic Voltage-Source Inverter's structures:

VSIs basic structures are the simplest inverter topologies that can output an AC signal from single DC source and can be classified as: 1) Single phase half bridge, 2) H-bridge structure and 3) Multi-phase inverter structure as shown in Figure I-6.

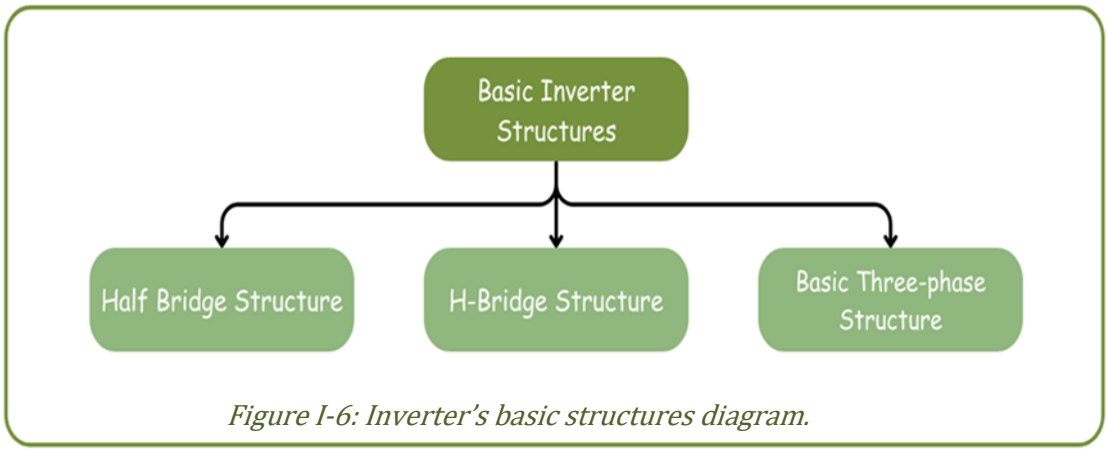

Figure I-6: Inverter's basic structures diagram.

I.2.3.1. Half-Bridge inverter:

The half-bridge structure is the simplest inverter structure that generates AC voltage from a DC source, it is composed of a single DC source Vdc and two capacitors (C1 & C2) that split the source in half in order to obtain 0V midpoint, it is possible to use a center-tapped voltage source supply instead. As can be observed on Figure I-7 it contains two power semiconductor switches (S_1 and S_2) connected in series each with a freewheeling diode in parallel, it is called an inverter leg or an inverter arm, the diodes are used to provide current path when the power transistors are turned off. The DC link capacitors are used to store energy and provide a smooth DC voltage to the inverter. the switches are controlled to provide two-level output ($-\frac{V_{dc}}{2}$ and $\frac{V_{dc}}{2}$) thus it is also named: two-level inverter. Table I.2 and Figure I.8 show the switching states and the output levels respectively.

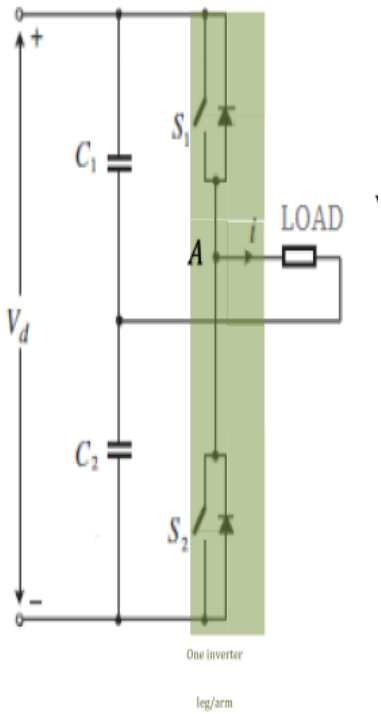

Figure I-7:Half-bridge inverter.

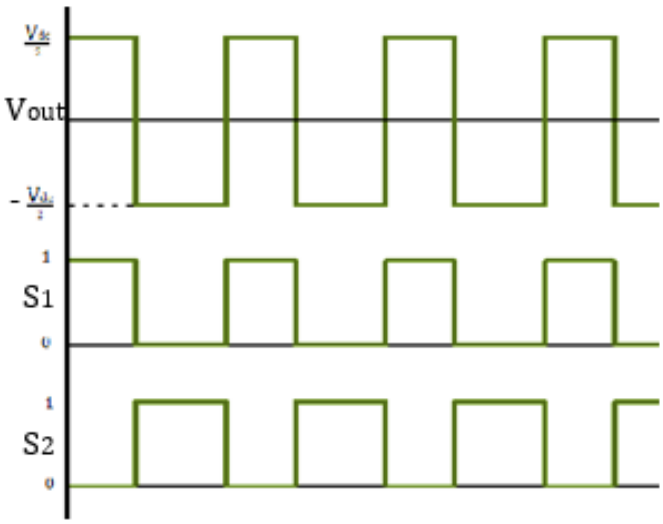

Figure I-8: Switching pattern of half bridge switches along with Voltage waveform across Resistive LOAD.

Switch state		output Voltage
S_1	S_2	
0	1	$-\frac{V_{dc}}{2}$
1	0	$\frac{V_{dc}}{2}$

Table I-2: Half-Bridge state table

DEDUCE THE STATE TABLE: The switching states of the inverter can be deduced using circuit analysis. When both switches in one inverter leg are turned on, it results in a short-circuit state. When both switches are turned off, it results in a floating state. These two states are avoided and forbidden. The remaining two states, when one switch is turned on and the other is turned off, cause a voltage drop of either $-\frac{V_{dc}}{2}$ ***volts*** or $\frac{V_{dc}}{2}$ ***volts*** across the load. Applying certain periodic on and off control commands to the switches will lead to producing an alternating voltage and current signal in the output as illustrated in Figure I-8.

Table I-2 shows the possible switching states of the inverter and the corresponding output voltages.

It is important to note that when using non-ideal semiconductor switches in an inverter leg and transitioning between the two conducting states described in Table I-2, a delay circuit must be added to the output of the control signals that govern these switches. This addition is intended to avoid the occurrence of the short-circuit state [3], as shown in Figure I-9. This requirement applies to every inverter leg in real-world applications, particularly those involving medium and high voltages (MV and HV).

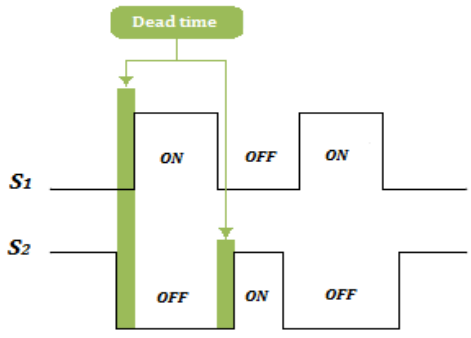

Figure I-9:Dead time representation

I.2.3.2. Full-Bridge Inverter:

Full-bridge inverter also known as H-bridge cell is illustrated in Figure I-10 is another basic inverter structure where the switching circuit consists of two half-bridge legs connected in parallel with a DC link capacitor C_d, The switches can be controlled to provide Three-Level voltages ($-V_{dc}$, 0 and $+V_{dc}$), Table I-3 shows the possible switching states of the inverter and the corresponding output voltages. Figure I-11 presents the output waveform across points A and B: $V_{AB}=V_{A0}-V_{B0}$, along with corresponding gating signals.

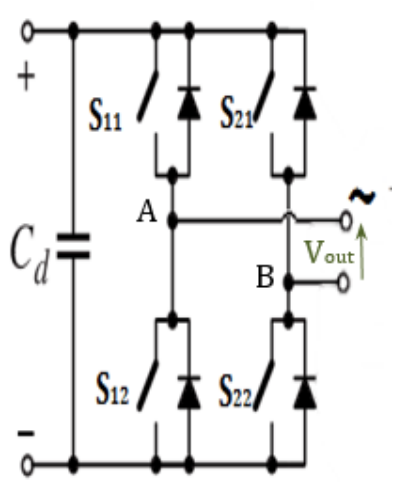

Figure I-10: H-bridge inverter structure [2].

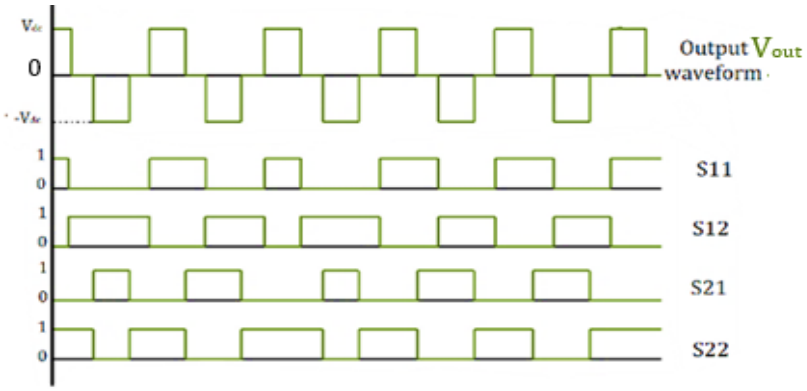

Figure I-11: Switching pattern of H- bridge switches and the outputted three level pattern

Switching state				Output
S_{11}	S_{12}	S_{21}	S_{22}	
0	1	1	0	$-V_{dc}$
0	1	0	1	0
1	0	1	0	0
1	0	0	1	V_{dc}

Table I-3: H-bridge state table

The output voltage of H-bridge can be written as: $V_{out} = (S_{11} - S_{21})V_{dc}$ [1]

I.2.3.3. Multi-Phase Inverter Structure:

There exist numerous Multi-phase inverter structures, the basic one is represented in Figure I-12, it is composed of three inverter legs all connected in parallel with two DC link capacitors and a three-phase load (either Y or Δ connection). The output line-to-line voltages can result up to three-level pole voltages (-Vdc,0 and Vdc) as shown in Figure I-13.

Table I-4 represent the conducting switching stats along with output pole voltages:

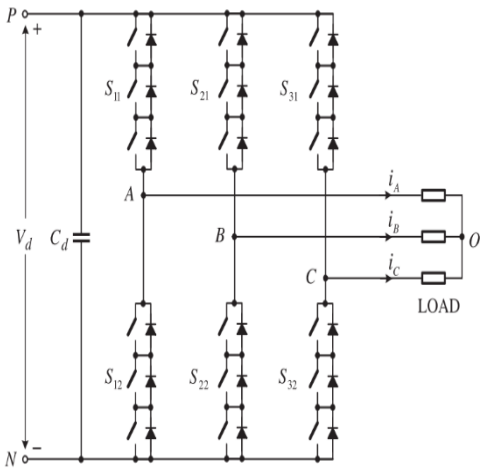

Figure I-12: Basic Three Phase Inverter [2].

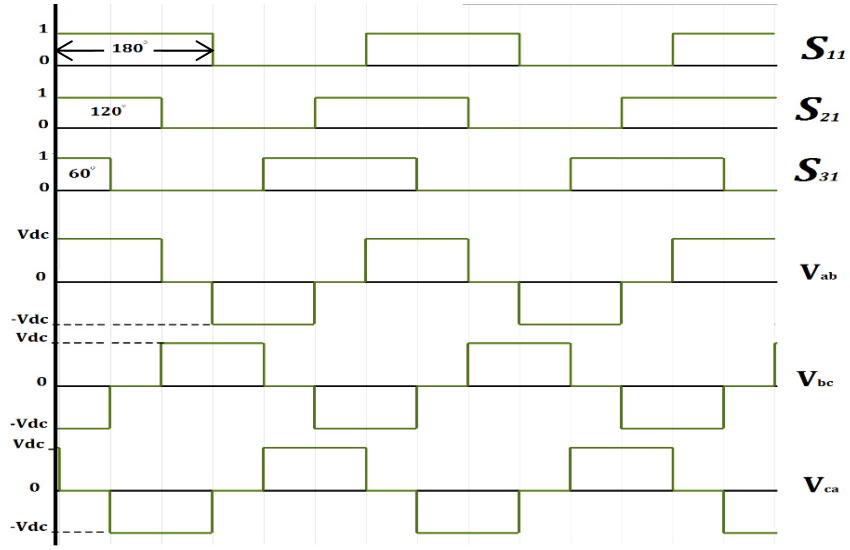

Figure I-13: Line-to-line three phase inverter waveform.

Switching states			Pole voltages		
S_{11}	S_{21}	S_{31}	V_{ab}	V_{bc}	V_{ca}
0	0	0	-Vdc	+Vdc	0
0	0	1	-Vdc	0	+Vdc
0	1	1	0	-Vdc	+Vdc
1	0	0	+Vdc	-Vdc	0
1	1	0	+Vdc	0	-Vdc
1	1	1	0	+Vdc	-Vdc

Table I-4: Three Phase Inverter State table.

I.2.4. Multi-Level Inverter Configuration:

The concept of multilevel inverter topology (MLI) was introduced in the early 1975 with the three-level converters [4] [5]. Figure I-14 illustrates several multilevel waveforms. In multi-level converters, the desired output voltage waveform is composed of a specific number of voltage levels. These levels are typically achieved by combining one or more independent DC sources (depending on the type of the multi-level converter). The most used independent sources include photovoltaic panels, fuel cells, batteries, and ultra-capacitors. As the number of voltage levels increases, the complexity of circuit design grows, thus, more complex switching controller circuits. The multilevel inverters is constructed with a minimum of three levels. As the number of levels approaches infinity, the output waveform approximates a sinusoidal waveform. [6]

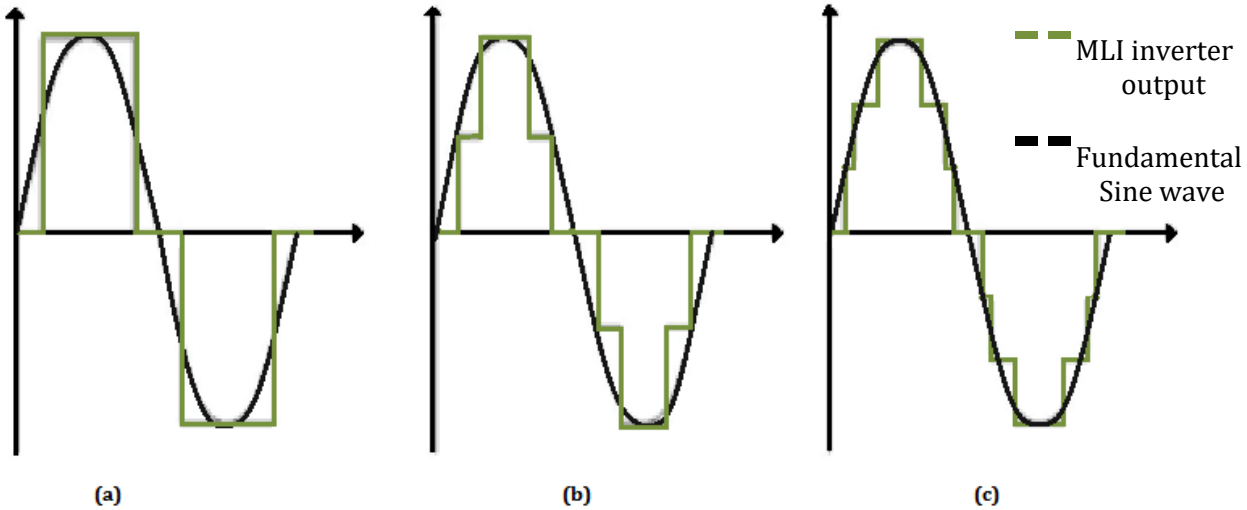

Figure I-14: (a) A three-level, (b) five-level and (c) seven-level output waveform at fundamental switching frequency. [26].

Multi-level inverters are generally categorized into three main commercial configurations: [4] [5]:Neutral Point Clamped (NPC), Flying Capacitor (FC) and Cascaded H-Bridge (CHB) as shown in Figure I-15.

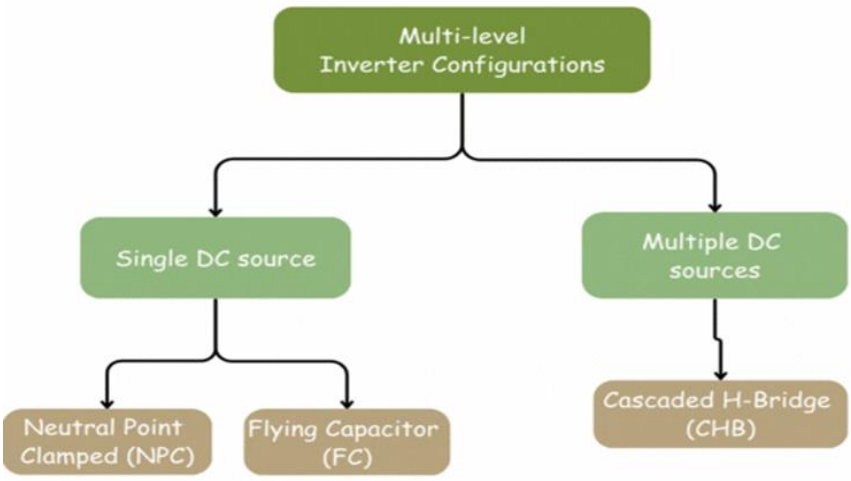

Figure I-15: Block Diagram for MLI Configurations.

I.2.4.1. Neutral-Point-Clamped Configuration:

The neutral-point-clamped (NPC) configuration as shown in Figure I-16, is a type of MLI configuration. In this configuration, the neutral point of the DC source is clamped to a fixed voltage. This means that the output voltage of the inverter is aimed to be balanced, even if the load is unbalanced [1] [5]. The NPC configuration is one of the most common topologies for medium and high voltage applications such as in variable-speed drives.

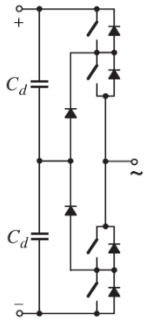

Figure I-16: Neutral point clamped inverter leg [2].

I.2.4.2. Flying Capacitor Configuration:

The flying capacitor (FC) configuration is another type of MLI configuration, it is similar to NPC. In this configuration, the neutral point of the inverter is not clamped to a fixed voltage. Instead, it is connected to a flying capacitor as shown in Figure I-17. This means that the output voltage of the inverter can be unbalanced [1]. The FC configuration is often used in applications where a high-power density is required, such as in electric vehicles.

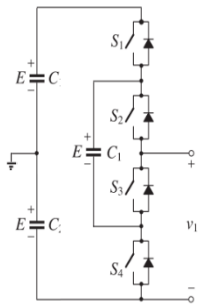

Figure I-17: Flying capacitor inverter

I.2.4.3. Cascaded H-Bridge Configuration:

Cascaded H-bridge is a type of multilevel inverter that is made up of multiple H-bridge cells connected in series. Each H-bridge cell can produce three different output voltages: (-V_{dc} , 0 and +V_{dc}). The output voltage of the cascaded H-bridge is the sum of the output voltages of all the H-bridge cells. The number of output voltage levels in cascaded inverter is defined by: $m = 2s + 1$ where s is the number of sources or cells, Figure I-18 represent a basic cascaded H-bridge configuration with number of cells s=3.

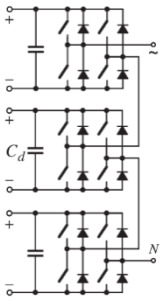

Figure I-18:Cascaded H-bridge configuration.

I.3. Modulation Topologies for 3-level Voltage Source Inverters:

Since the invention of semiconductor power switches, multiple modulation topologies have been developed to control the power converters. Among these is pulse-width modulation (PWM), which was first developed in the 1960s. PWM are classified under three different categories (as shown in Figure I-19) based on the switching frequency used, as 1) High switching frequency, 2) Fundamental switching frequency PWM and 3) variable frequency modulation techniques [6] [7].

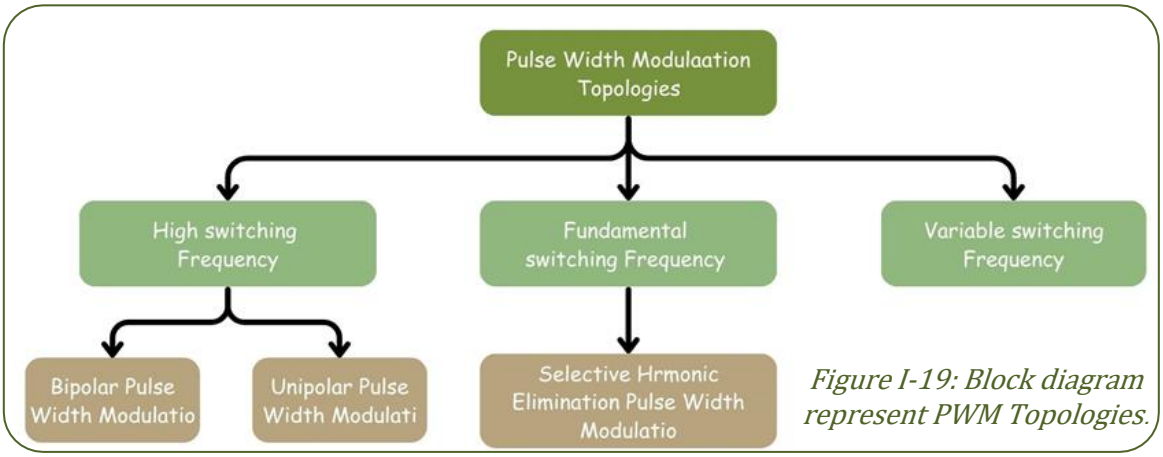

Figure I-19: Block diagram represent PWM Topologies.

I.3.1. Pulse Width Modulation signals:

Figure I-20 represents the main characteristic of PWM, PWM is a technique for generating a variable-width pulse train from a fixed-frequency clock signal. The width of the pulse train is modulated according to a reference signal. The duty cycle is the ratio of the pulse in ON state (T_{on}) to the period of the clock signal T. A duty cycle of 50% means that the pulse is High for half of the period [8], in this case a rectangular waveform is generated. The duty cycle can be adjusted to control the average power delivered to the load [7] .

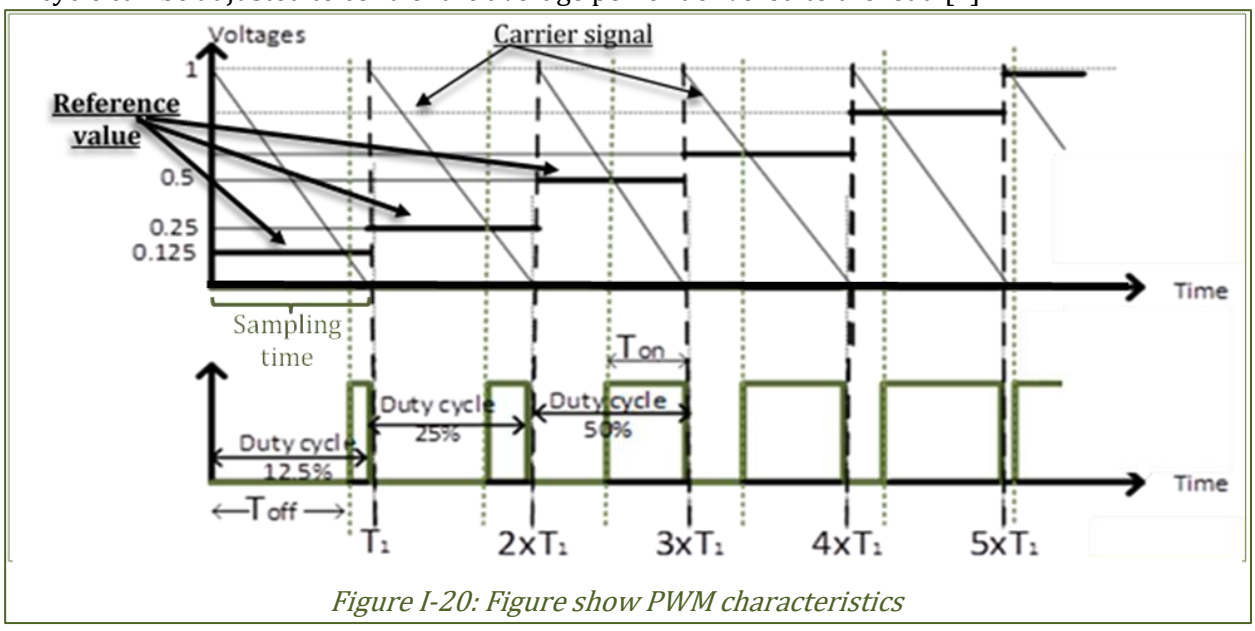

Figure I-20: Figure show PWM characteristics

To generate PWM gating signals, a reference signal, a carrier signal, and a comparator are needed. The maximum amplitude or peak value of the reference and carrier signals are denoted by $V_{r,peak}$ and $V_{c,peak}$, respectively. Their frequencies are denoted by f_c and f_r, respectively. The ratio of $V_{r,peak}$ to $V_{c,peak}$ is known as the modulation index or amplitude modulation index, m_a: $m_a = \frac{V_{r,peak}}{V_{c,peak}}$.

The ratio of f_c to f_r is known as the frequency modulation index, *mf*: $m_f = \frac{f_c}{f_r}$

The PWM gating signals are generated by comparing the reference signal to a triangular carrier signal. When the reference signal is greater than the carrier signal, the comparator outputs a high pulse. When the reference signal is less than or equal to the carrier signal, the comparator outputs a low pulse. The width of the pulses is determined by the modulation index [8].

I.3.2. Sinusoidal Pulse Width Modulation:

Sinusoidal pulse width modulation or SPWM is the most common used method in motor control and inverter application [3] [9]. It is a modulation technique that uses a sinusoidal as a reference signal to generate a pulse width modulated (PWM) signal. The PWM signal is then used to control the switching of power semiconductor devices in the inverter.

In the case of a half-bridge inverter, two switching signals are required to control the two power semiconductor devices S_1 and S_2. from Table I-2 it is possible to notice that $S_1 = \overline{S_2}$ in both conducting states [1], The upper switch S_1 is controlled by the SPWM signal from the comparator, while the lower switch is controlled by the logical inverse of the SPWM signal. This ensures that the two switches are never turned on at the same time (because that would create a short circuit.) therefore the output is a two level SPWM signal (-Vdc/2 and +Vdc/2).

In the case of an H-bridge inverter, four switching signals are required to control the four semiconductor devices. The two switches on each leg are controlled in the same way as the switch in a half-bridge inverter. However, the two upper switches must be carefully synchronized to ensure that the desired output voltage is well generated.

The two types of SPWM signals that can be generated by an H-bridge inverter are bipolar SPWM and unipolar SPWM [1] [3] [10] as shown in Figure I-21.

Switching state				Output	
S_{11}	S_{12}	S_{21}	S_{22}		
0	1	1	0	$-V_{dc}$	0°
1	0	1	0	0	90°
1	0	0	1	V_{dc}	180°
1	0	1	0	0	270°

Bipolar transition

Unipolar transition.

Table I-5: H-Bridge state table

- Bipolar SPWM is generated when S_{11} is the logic-inverse of S_{21} (i.e $S_{11} = \overline{S_{21}}$) this will generate a two level SPWM signal (-Vdc and +Vdc) [1] as shown in Figure I-21 and illustrated in Table I-5.
- Unipolar SPWM is generated when S_{11} is leading S_{21} by 180° as described in Table I-5, this will generate a three level SPWM signal (-Vdc, 0 and +Vdc) as shown in Figure I-21, this is named Unipolar SPWM because in positive half cycle only levels (+Vdc, 0) occur while in negative half cycle only signal (-Vdc, 0) occur [8].

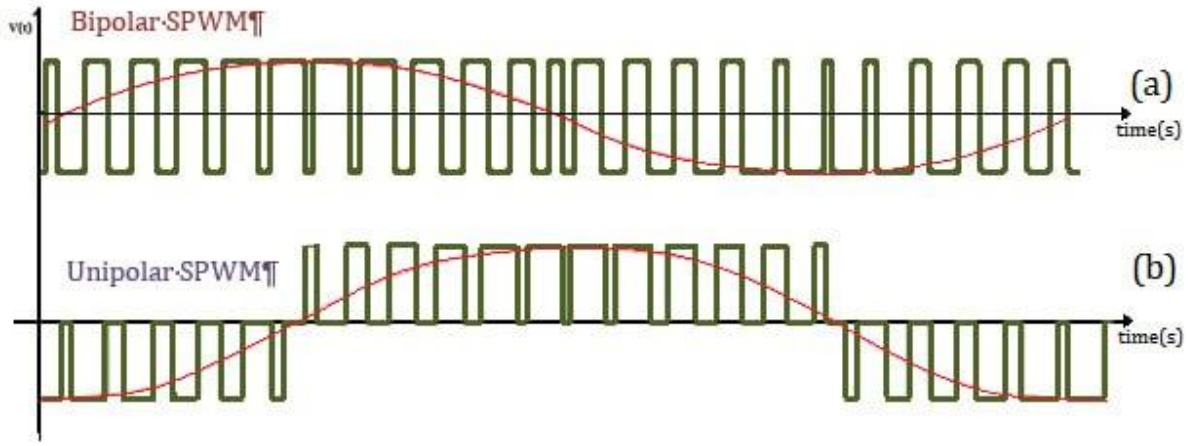

Figure I-21:Bipolar and Unipolar SPWM waveform of Three-level inverter.

Harmonics and THD: In multi-level inverters the output of the DC/AC inverter actually looks more like staircase signal and not ideal sinusoidal because it contains harmonics [11], which are frequencies that are not present in the original sinusoidal waveform and that are integer multiples of the fundamental frequency. Harmonics can cause problems such as noise and vibration. However, the staircase signal has several advantages over the square-wave inverter, such as lower voltage stress and reduced switching losses [2].

THD is a ratio that can be calculated to deduce whether a signal is close to a pure sine wave or not, it is calculated as the ratio of the RMS value of the harmonics to the RMS value of the fundamental frequency:

$$\text{THD} = \frac{\sqrt{\text{RMS of harmonic components}^2 - \text{RMS of fundamental component}^2}}{\text{RMS of fundamental component}} \qquad \text{(a)}$$

A lower THD value indicates a signal that is closer to a pure sine wave. A signal with a low THD value is more likely to be a pure sine wave than a signal with a high THD value.

I.3.3. Selective Harmonic Elimination Pulse Width Modulation:

SHEPWM is a modulation technique mainly used to eliminate any number of harmonics, including the lower-order harmonics that are most harmful to power quality. It can also be used to reduce the switching frequency, which can improve the efficiency of the inverter. Historically, SHEPWM was proposed in the early 1960s, when it was found that low order harmonics could be suppressed by adding several switching angles in a square wave voltage. Years later the idea was extended by Patel and Hoft [12]using Fourier series to mathematically express the harmonic contents of a PWM waveform as follows:

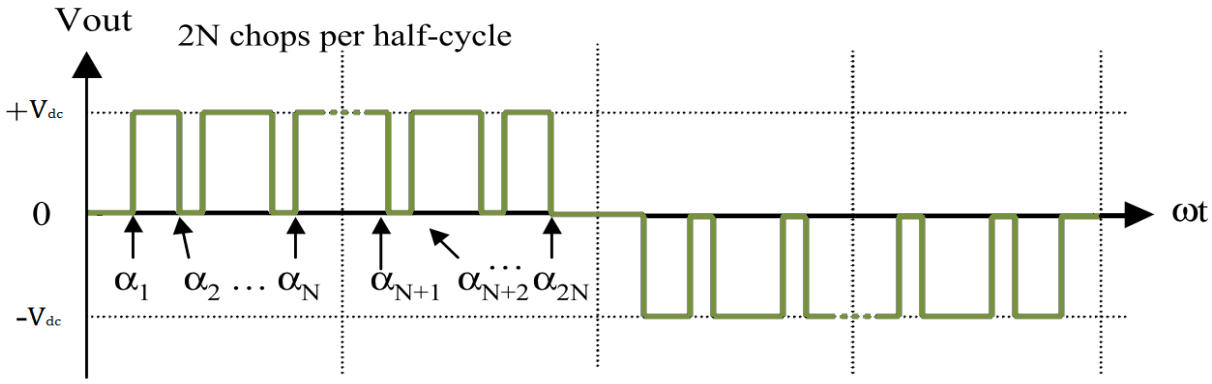

Figure I-22: Generalized three-level SHE PWM waveform. [12]

Figure I-22 represents the generalized three-level SHEPWM waveform where N is the number of switching angles per quarter-cycle. The output waveform is assumed to be odd quarter-wave symmetry, therefor the dc component and the even harmonics are equal to zero and the Fourier series of the three-level SHEPWM can be written as:

$$V_{out}(\omega t) = \sum_{n=1}^{\infty} a_n \sin(n\omega t) \qquad (b)$$

Where:

$$a_n = \frac{4\,V_{dc}}{n\,\pi} \sum_{k=1}^{N} (-1)^{k+1} \cos(n\alpha_k)\,, \text{for } n \text{ odd} \qquad (c)$$

N is the number of switching angles per quarter.

$\alpha_k$is the switching angles, which must satisfy the following condition: $0 < \alpha_1 < \alpha_2 < \dots \alpha_N < \frac{\pi}{2}$

E is the amplitude of the dc source.

and n is the harmonic order.

The above equation results a group of nonlinear and transcendental equations of the harmonic components:

$$cos(\alpha_1) - cos(\alpha_2) + cos(\alpha_3) - \dots \mp cos(\alpha_N) = \frac{\pi}{4}m \qquad (I\text{-}1)$$

$$cos(3\alpha_1) - cos(3\alpha_2) + cos(3\alpha_3) - \dots \mp cos(3\alpha_N) = \frac{3\pi}{4V_{dc}}h_3 \qquad (I\text{-}2)$$

$$cos(5\alpha_1) - cos(5\alpha_2) + cos(5\alpha_3) - \dots \mp cos(5\alpha_N) = \frac{5\pi}{4V_{dc}}h_5 \qquad (I\text{-}3)$$

$$\vdots$$

$$cos(N\alpha_1) - cos(N\alpha_2) + cos(N\alpha_3) - \dots \mp cos(N\alpha_N) = \frac{N\pi}{4V_{dc}}h_N \qquad (I\text{-}N)$$

To control the amplitude of the fundamental component, the modulation index in the above set of equations, m is given. According to the above nonlinear system, N-1 harmonic can be eliminated from the output waveform by setting equations (I-2) to (I-N) to zero. , therefore, N nonlinear equations will be set up as follows:

$$\cos(\alpha_1) - \cos(\alpha_2) + \cos(\alpha_3) - \dots \mp \cos(\alpha_N) = \frac{\pi}{4}m$$

$$\cos(3\alpha_1) - \cos(3\alpha_2) + \cos(3\alpha_3) - \dots \mp \cos(3\alpha_N) = 0$$

$$\cos(5\alpha_1) - \cos(5\alpha_2) + \cos(5\alpha_3) - \dots \mp \cos(5\alpha_N) = 0$$

$$\vdots$$

$$\cos(N\alpha_1) - \cos(N\alpha_2) + \cos(N\alpha_3) - \dots \mp \cos(N\alpha_N) = 0$$

The switching angles are calculated by solving the above equations. The challenge of SHEPWM is solving nonlinear equations using the best technique. In overall, with the help of Fourier series THD of the output signal can be eliminated by calculating the value of 2N firing angels that make the first 2N-1 harmonics of the signal equal to zero [12].

There are several optimization algorithms such as Newton-Raphson, PSO algorithm Genetic Algorithm (GA), Moth Flame Optimization (MFO) [11] etc. can be used for efficiently solving the nonlinear system of equations , however solving such equations is not in the objective of this project, therefore a set of solution have been taken from references [13] [12] [14] and , compared and tested in MATLAB® then use the best solution (less THD) in the implementation of the FPGA-Based SHEPWM controller.

I.4. Overview of Real Time Simulation:

Real-time simulation (RTS) refers to a computer simulation of a physical system that can run at the same rate as real time. In other words, the computer model operates at the same rate as the real physical system [15], For example, if a tank takes 20 minutes to fill in the real world, in the simulation would also take 20 minutes. RTS is used whenever there is an interaction between a system operating at real-world speed and a model.

Real-time simulation is common in computer games, but it is also important in the industrial market for operator training and offline controller tuning. Computer languages such as LabVIEW, SolidThinking Embed, and Simulink enable the rapid creation of real-time simulations.

. Some commercially available HIL simulators [16]:

- Nvidia: Nvidia HIL simulators are primarily used for the development of driver assistance systems and autonomous driving functions. Using a GPU, a 3D model of the road and environment is calculated, like a racing game on a game console.
- dSPACE: The larger real-time simulators are typically based on a combination of Intel Xeon and FPGA processors. The Model development is performed in MATLAB/Simulink with proprietary libraries for interface definition, while manufacturer-specific software provides the user interface for control and visualization. dSpace systems are widely used in the automotive sector.
- OPAL-RT: Intel Xeon processors are often used in combination with FPGA boards, and similarly, like the simulators from dSPACE, application-specific systems can be created. The development environment is also MATLAB/Simulink, which also serves as the user interface. Manufacturer-specific software is also used for visualization. OPAL-RT real-time simulators are mainly used in the energy industry and in power electronics.
- Typhoon HiL: FPGA in cooperation with coprocessors, for software-side there are proprietary models with interfaces and integration possibilities for C/Python/MATLAB programs.
- PLECS: RT-Box, Xilinx Zynq System-on-Chip with FPGA and multiple CPU cores, software-side solved by MATLAB® connection with additional proprietary add-on libraries or with completely independent software.
- Micronova: Various systems for simulating vehicle components, solutions for electrified drives are available.
- Siemens: Siemens are often used in industry. The company's newer products include the virtual commissioning of fully industrial automated production lines including extensive software libraries.

Real time simulation can be classified into:

I.4.1. Model-In-The-Loop simulation:

Model in the Loop (MIL) is the simulation of an embedded system in an early development phase of modeling in the field of model-based software development. Embedded systems communicate with their environment and often expect sensor signals as input and then stimulate the physical system. In order to function properly, the environment of the embedded system must be simulated. If the embedded system (model) is now simulated in a loop together with the environmental model, this is called Model in the Loop Simulation. [17]

MIL is a cost-effective way to test algorithms for embedded systems. Development and simulation environments for model-based development are, for example, MATLAB/Simulink, PLECS, Dymola, ASCET or the free software Scilab/Xcos.

MIL is typically used in the early stages of embedded system development. It can be used to investigate specified model behaviors. Data collected during MIL simulation helps to validate that the model behaves as expected and can even be used for reference during the next design and testing phase i.e. in later stages, the system may be tested using Software in the Loop (SIL), Processor in the Loop (PIL), or Hardware in the Loop (HIL) simulations. These simulations are more complex than MIL, but they provide a more realistic test of the system's performance.

I.4.2. FPGA-In-The-Loop simulation:

FPGA-in-the-loop (FIL) simulation provides the capability to use soft simulation platforms such as MATLAB® or Simulink® software for testing designs in real hardware for any existing Hardware Description Language (HDL) code.

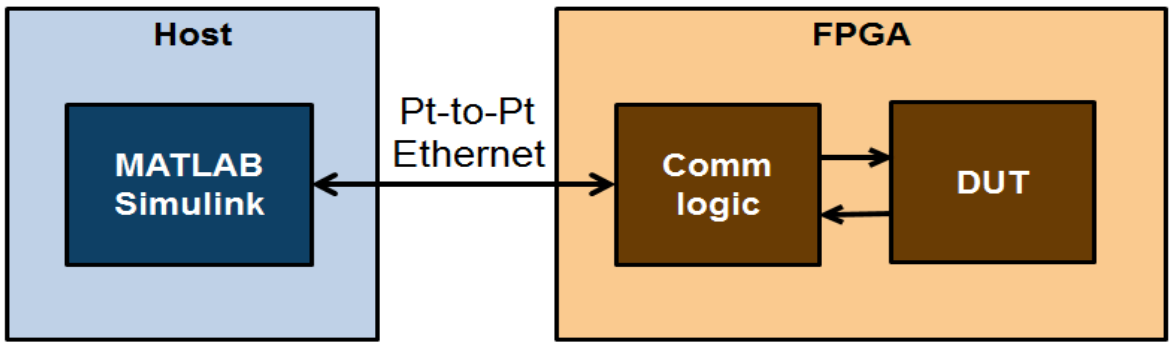

Figure I-23: Diagram represent the communication in FPGA-In-The-Loop block [40]

In MATLAB® for example, the FILSimulation System object™ can be used to connect an FPGA execution to a MATLAB® test bench. It does so by applying input signals to and reading output signals from an HDL model running on an FPGA. This object can be used to model a source or sink device by configuring the object with input or output ports only. The HDL code can be either manually written or software generated from a model subsystem, All Device Under Test (DUT) I/Os are routed to Simulink through the FIL communication logic as it can be observed in Figure I-23.

I.4.3. Rapid Control Prototyping:

Rapid control prototyping (RCP) is a test and development methodology used to accelerate the design process by using model-based design from an RCP platform to test a control strategy on physical hardware i.e. implement a simulated controller early in the design process to test the physical hardware as illustrated in Figure I-24. RCP typically involves the following steps:

- Design a control strategy for the system.
- Implement the control strategy on a rapid prototyping platform.
- Connect the RCP platform to the physical system and send the control signals to the physical system.
- Iterate the design process as needed.

RCP can help to improve the quality of a control system and allows for greater flexibility in the design process, as it is possible to quickly experiment with different control strategies by ensuring that it is well-tested and executed.

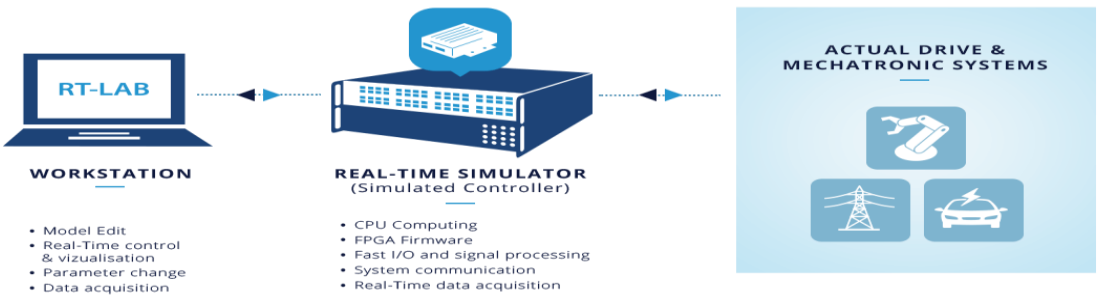

Figure I-24: Schematic representation of RCP [28]

I.4.4. Hardware-In-The-Loop simulation:

Hardware in the Loop (HIL) is a procedure in which an embedded system (e.g., real electronic control unit) is connected to a real time simulator via its inputs and outputs and test the controller on simulated environment, as illustrated in Figure I-25, the HIL simulator serves as a replica of the system's real environment. From the perspective of testing, HIL is a method for securing embedded systems, supporting development, and commissioning machines and plants early on. If the target controller is not used and only a simulation of the software is performed, this is referred to as Software in the Loop (SiL) [15]

Figure I-25: Schematic representation of HIL. [22]

.

I.4.5. Software-In-The-Loop simulation:

In contrast to Hardware in the Loop (HIL), no physical hardware is used in the Software in the Loop (SIL) method [18] as graphically explained in Figure I-26. The original controller code is written into a file using an automated coding process, typically a Dynamic Link Library (DLL). This DLL can be embedded in an environment model in a simulation tool (e.g., MATLAB® and Simulink®). The DLL is then tested in the purely virtual model world. Unlike the HIL method, no real controller is required. SIL makes it possible to test software prior to the initialization of the hardware prototyping phase, significantly accelerating the development cycle.

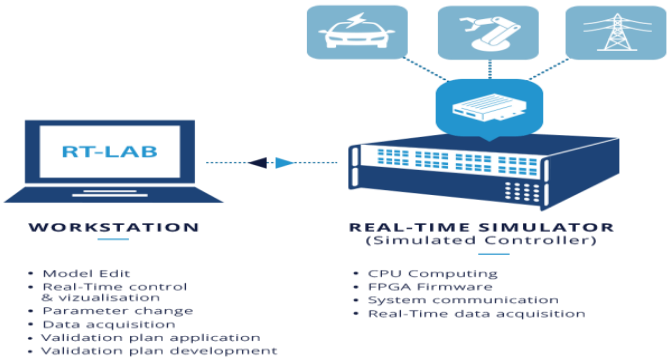

Figure I-26: Schematic representation of SIL. [22]

SIL simulators do not run in real time, and the simulation can be paused at any time for debugging. Hardware-related runtime problems or variable overflows that occur on the real controller can hardly be detected with SIL simulations.

Other advantages of SIL include the fact that the target hardware does not have to be set yet, and that the costs are much lower due to the lack of a real simulation environment. The simulation model used here can also be used in subsequent HIL simulations, which allows the individual test runs to be compared with each other.

I.4.6. Power-Hardware-In-The-Loop simulation:

Power Hardware-in-the-Loop (PHIL) simulation represents a natural extension of HIL, in which the real-time simulation environment can exchange not just low-voltage, low-current signals, but the power required by the Devices under Test (DUT). To bridge this gap, power amplifiers are inserted between DUTs rated for higher power and the low-level simulator I/Os, all while providing the necessary feedback to properly close the loop as it can be observerd in Figure I-27. Power amplifiers are selected for user applications based on their closed-loop performance and ability to generate and absorb power. PHIL also allows the simulation of higher power flows between DUTs, as well as with the simulated electric circuit running on the simulator. This capability enables engineers to test multiple systems, including power converters, generators, motors, and PV loads, while also benefiting from high-fidelity simulation that provides greater flexibility and safety than typical analog benches and dynamometers.

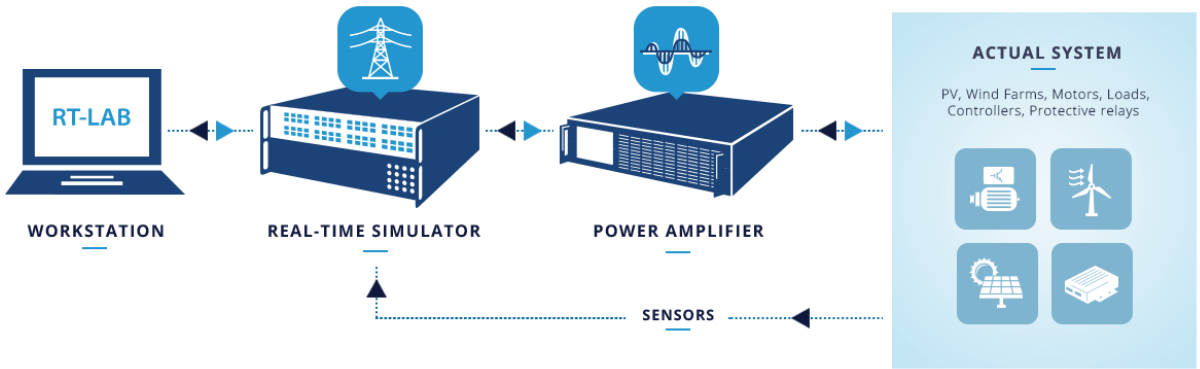

Figure I-27: Schematic representation of PHIL [22]

I.5. Conclusion:

This chapter dealt with the main theoretical background that has been used in the implementation of this project. Starting with the main inverter topologies and inverter modulation techniques, then passed to the most real-time simulation technologies used in research laboratories and industry. Power electronics systems on large ships act as multimegawatt microgrids, powering propulsion and all other onboard electrical equipment. These systems often require multimillion-dollar investments and a year or more to build. As a result, engineering teams that develop the control software often lack access to feature-complete hardware until late in development. For that reason, semi physical simulation has become popular these last years.

In this project, an H-bridge topology has been constructed as the simulated environment using HIL Simulink and Opal-RT simulation tools to test the implemented FPGA-based inverter controller.

Chapter II. Hardware In the Loop and MATLAB® Simulation Setup:

II.1. Overview:

In this chapter we will describe the hardware used in the HIL simulation, we also discuss the simulation setup and the results of the MATLAB® simulation, Early in last year, a group of electrical engineering students designed an industrial H-bridge PWM inverter for research purposes, the power semiconductor switches (Industrial High frequency MV IGBT modules) used in their design where very expensive modules therefore Rapid Control Prototyping was used to control and test the functionality of the designed H-Bridge inverter, Despite being fully operational, the finished product was not ready, the designed inverter needs a real control unit. The objective of this project is to implement an FPGA-Based control unit for inverter's High frequency semiconductor switches, In addition designing a functional PWM control circuits (using Bipolar PWM, Unipolar and SHEPWM) using VHSIC Hardware Description Language (VHDL) Then upload and run the circuit in DE2 Field Programable Gate Array Board then test its functionality using OPAL-RT HIL simulation tools and compare the results with simulated results before test the controller in real hardware. The results of the HIL simulation as well as the VHDL design are discussed in the next chapter.

II.2. FPGA:

FPGA, which stands for **Field Programmable Gate Array** is a type of integrated circuit that can be programmed or reprogrammed after manufacturing its architecture that consists of an array of programmable logic block that can be configured to perform various digital functions, these logic blocks can be configured to perform complex combinational functions, or act as simple logic gates like AND and XOR, they can be seen as Lego basic block that we can use to build several digital circuits . In most FPGAs, logic blocks as shown in Figure II-1 also include memory elements, like simple flip-flops or more complete blocks of memory [19]. FPGA configuration is made using a hardware description language (HDL) namely Verilog or VHDL (Very High-Speed Integrated Circuit Hardware Description Language), Many FPGAs can be reprogrammed to implement different logic functions, allowing flexible reconfigurable computing as performed in computer software.

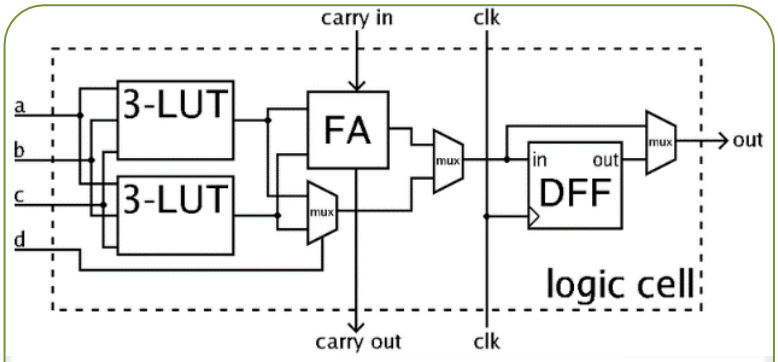

Figure II-1: Simplified example illustration of a basic logic cell in FPGA [19]

In this project, we used the Altera DE2 standard board containing the Cyclone II FPGA chip.

II.3. Altera Cyclone II DE2 Field Programable Gate Array Board:

DE2 is a Development and Educational board conceived for the practical implementation of digital and mixed-signal circuits, even quite complex ones. It includes several I/O devices and interfaces, from simple ones (switches, pushbuttons, LEDs, seven segment displays) to complex devices (LCD matrix display, Ethernet network interface, USB 2.0, SD memory cards, analog audio I/O, analog video input, VGA video output …etc.) [20]. The core of DE2 is an FPGA, composed of more than 30.000 Logic Elements (LE). DE2 may host simple introductory projects, and sophisticated ones that may include one or more microcomputers [20]. Figure II-2 represents DE2 Board top view.

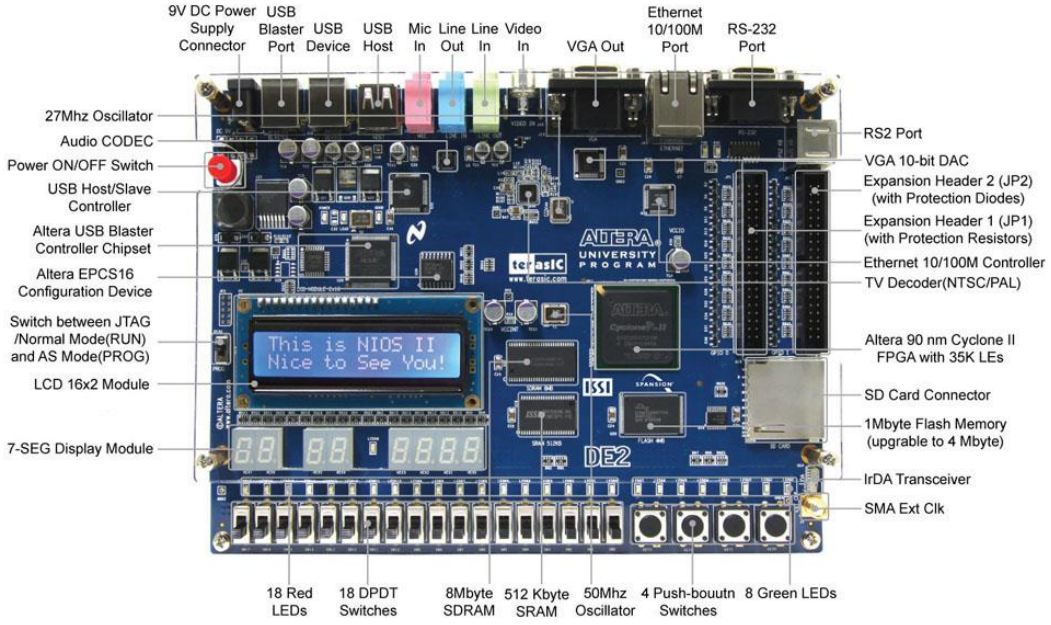

Figure II-2: DE2 Board Top View [35].

II.4. OP4510 RCP/HIL Real Time Simulator:

II.4.1. Overview:

OPAL-RT TECHNOLOGIES is the leading developer of open Real-Time Digital Simulators and Hardware-In-the-Loop testing equipment for electrical, electro-mechanical and power electronic systems. OPAL-RT Simulators are used by engineers and researchers at leading manufacturers, utilities, universities and research centers around the world.

OPAL-RT offers a wide range of simulator platforms as shown in Table II.1 to meet all current industry needs and forthcoming challenges. All simulators are based on a modular and flexible design and are fully customizable and expandable for specific I/O requirements. [21] [22].

Table II-1: Simulator Platforms Comparison [23].

Simulator platform	OP4200	OP4510	OP5600	OP5707	OP5031
Compatible Simulation Systems and Software	RT-LAB, eFPGASIM	RT-LAB, HYPERSIM, eMEGASIM, ePHASORSIM, eFPGASIM	RT-LAB, HYPERSIM, eMEGASIM, ePHASORSIM, eFPGASIM	RT-LAB, HYPERSIM, eMEGASIM, ePHASORSIM, eFPGASIM	RT-LAB, HYPERSIM, eMEGASIM, ePHASORSIM
CPU	ARM	INTEL XEON E3	INTEL XEON E5	INTEL XEON E5	INTEL XEON E5
Number of cores	2	4	4,8,16 or 32	4,8,16 or 32	4,8,16 or 32
XILINX FPGA (standard configuration)	ZYNQ (7030)	Kintex 7 (325T)	Spartan 3	Virtex 7 (485T)	n/a
I/O modules with 16 analog or 32 digital signals	4	4	8	8	n/a
Maximum number of I/O channels	128	128	256	256	n/a

II.4.2. Op4510 Simulation Platform:

The OP4510 as illustrated in Figure II-3 and Table II-1 is an entry-level simulator that contains an FPGA carrier, which can accept four standard OPAL-RT mezzanine boards [24], contains two digital and two analog boards in addition to the RS422 signals, these mezzanine boards interface using a DB37 connector at the back of the chassis as shown in Figure II-4. A, OP4510 has a rear and a front interface as presented in the next Figures and their elements are described in Table II-2 and Table II-3:

Figure II-3: OP4510 Top view [25].

Figure II-4 illustrates the OP4510 Components (back view): [25]

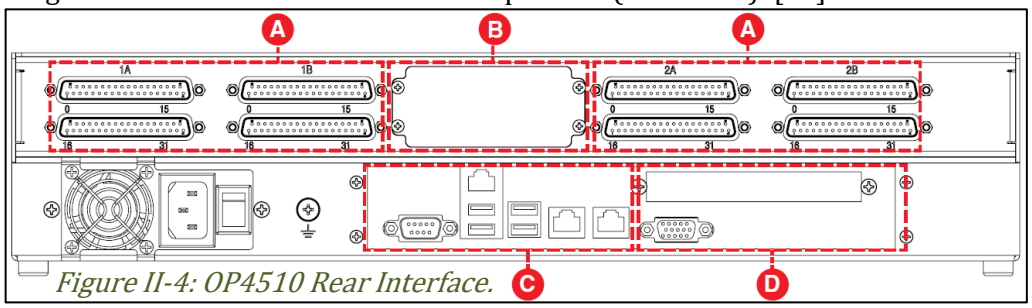

Figure II-4: OP4510 Rear Interface.

Table II-2: OP4510 Components (back view)

Element	Description
A	DB37 connectors for digital or analog inputs and outputs.
B	This section contains a blank plate; however, several options are available.
C	Standard computer connectors (left to right): Serial (COM) port, IPMI, USB ports*, network ports**, monitor (VGA)
D	Connector availability depends on the expansion option

Figure II-5 illustrates the OP4510 Components (front view) [25]

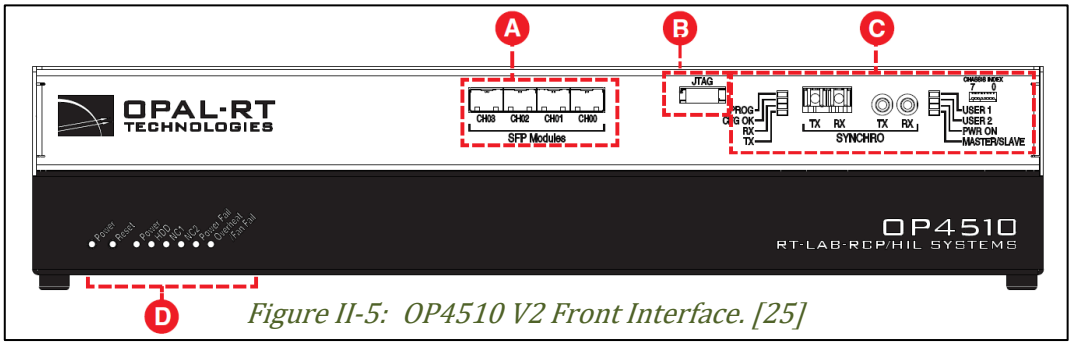

Figure II-5: OP4510 V2 Front Interface. [25]

Table II-3: OP4510 Components (front view)

Element	Description
A	Small form-factor pluggable (SFP) module connectors provide four high-speed communication links between other FPGA simulators or third-party devices. Each socket controls one communication link. SFP transceivers and fiber optic cables must be selected (and purchased separately) according to the type and speed of the communication protocol.
B	JTAG connector (for OPAL-RT technicians' use only) [24].
C	Synchronization connectors status, and user-configurable LEDs
D	Target computer monitoring interface. Two pushbuttons and six LED indicators

The OP4510 contains four Intel Xeon E3 processing cores, and comes with a custom-designed Linux operating system, providing the best real-time performance on the market. The OP4510 also provides the option of user-programmable I/O management, handled by a fast Xilinx® Kintex®-7 FPGA.

In general, the simulator (target simulator) is connected to a computer (host computer) as illustrated in Figure II-6 via ethernet cable with TCP/IP communication protocol. The host computer is supposed to send and control the model, display the graphical results, and receive or read the stored simulation results files in the simulator and some other functions. While the target simulator is supposed to load, build and run the model, during simulation the simulator can store the data without affecting the simulation operation's results and at the end it sends the file to the host computer.

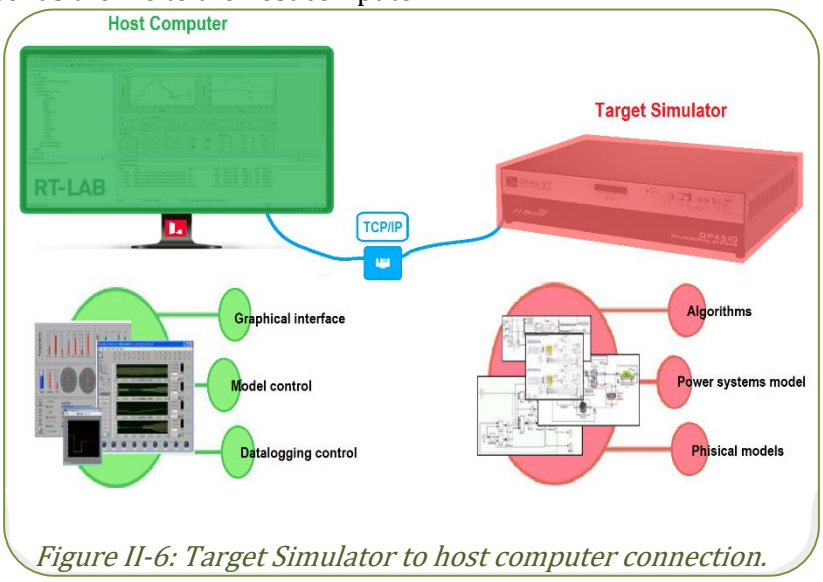

Figure II-6: Target Simulator to host computer connection.

II.5. MATLAB® and Simulink® Software:

MATLAB® is a popular software tool for modeling and simulating dynamic systems from MathWorks® (as shown in Figure II-7) while Simulink is a MATLAB-based graphical programming environment developed for modelling, simulating and analyzing dynamical systems. Its primary interface is a graphical block diagramming tool and a customizable set of block libraries. It offers tight integration with the rest of the MATLAB® environment. Simulink is widely used in automatic control and digital signal processing for multidomain simulation and model-based design. In this project Simulink is used for building and simulating the model circuits.

Figure II-7: MathWorks® Logo. [40]

II.6. Description of the simulation setup:

As mentioned in the theoretical chapter, before moving to HIL, we first implemented an MIL simulation. In this project, the DUT is the three-level inverter's control unit. The testing environment of this DUT contains a single-phase H-bridge inverter, a load, and wire

connections. Therefore, we first implemented a model for the environment (load and H-bridge inverter) and then a model for the DUT, as mentioned previously; a three different PWM controller circuits were designed and tested: Bipolar Sinusoidal PWM, Unipolar Sinusoidal PWM and Selective Harmonic Elimination PWM, the first circuit that have been implemented was the Bipolar SPWM since it is the simplest SPWM modulation circuit, after getting the results we improved the project by designing other complex SPWM controlling circuit which are Unipolar SPWM then SHEPWM.
We used MATLAB/Simulink simulation tools to perform the MIL simulation in this chapter.

The results of the MIL simulation were used as a reference for the hardware-in-the-loop (HIL) simulation.

II.7. The Single-Phase Full Bridge Inverter Model:

To modulate the H-bridge inverter for the MIL simulation, we used the Simulink Blocks, for the switches: we used ideal switch Blocks from the library Simscape, for the load: we used three different loads that have different values as following : Load 1 is R=100Ω as for Load_2 is R=33Ω, L=10mH and Load_3 is L=100mH R=33Ω, as shown in Figure II-8.

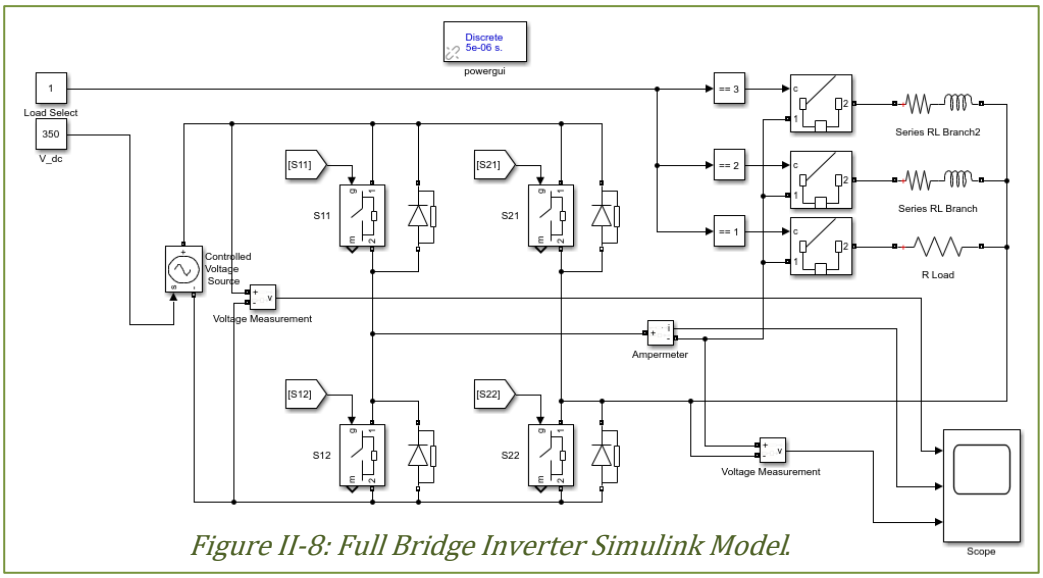

Figure II-8: Full Bridge Inverter Simulink Model.

II.8. MATLAB® Simulation and Results:

The inverter model in Figure II-8 has been controlled using three different modulation strategies, the four impulse signals generated from the controller are applied to the gate input of each switch using (FROM/GOTO) blocks from Simulink library, more about the structures and results are described in the following:

II.8.1. Bipolar SPWM structure:

As mentioned in the theoretical chapter SPWM require a reference of a sinusoidal signal and a sawtooth signal as a carrier, in addition Bipolar SPWM signals in H-bridge switches need to be as follow: $S_{11} = \overline{S_{12}}$ and $S_{21} = \overline{S_{22}}$. To obey these conditions, we used Simulink comparison block to compare between the refence signal from the Simulink's sine wave generator block and the sawtooth generator as shown in Figure II-9, we set the modulation index m_a to 0.9 $m_a = \frac{V_{r,peak}}{V_{c,peak}} = 0.9$, the frequency of the carrier is $f_c = 20kHz$ and the frequency of the refence is $f_r = 50Hz$.The modulation frequency index is $m_f = \frac{20kHz}{50Hz} = 400$

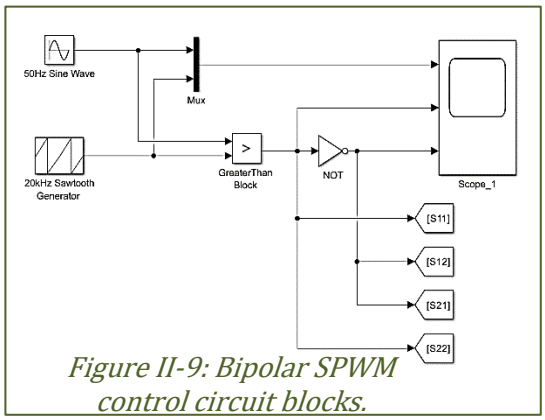

Figure II-9: Bipolar SPWM control circuit blocks.

The output of the scope in figure II-9 is observed in Figure II-10 which represents the expected gating signals. As we can see in Figure II-10, the duty cycle of S11and S22 is proportional to the reference signal, contrary to S12 and S21 which is inversely proportional to the reference signal These values have been saved as a reference to compare them with the output of the real FPGA-Based controller.

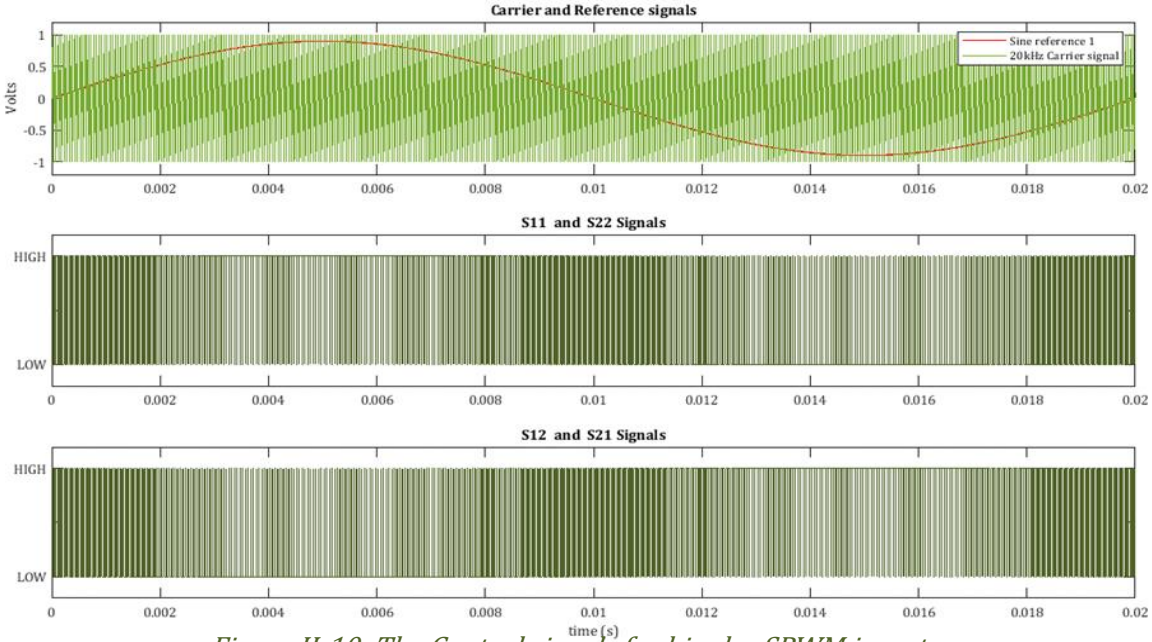

Figure II-10: The Control signals for bipolar SPWM inverter.

The results: after running the Simulation we obtained the results as shown in Figure II-11, due to the high switching frequency, some output signals appear as a rectangle, we calculated their mean value which is inserted and shown in red. the mean value of the output voltage is similar to the reference voltage for the different loads. The THD of the output voltage is high, equal to 121.35% for all different loads, however, the THD of the output current is different for the loads with inductive values and is equal to 23.71% for the lower inductance value, and 8.90% for the high inductance value.

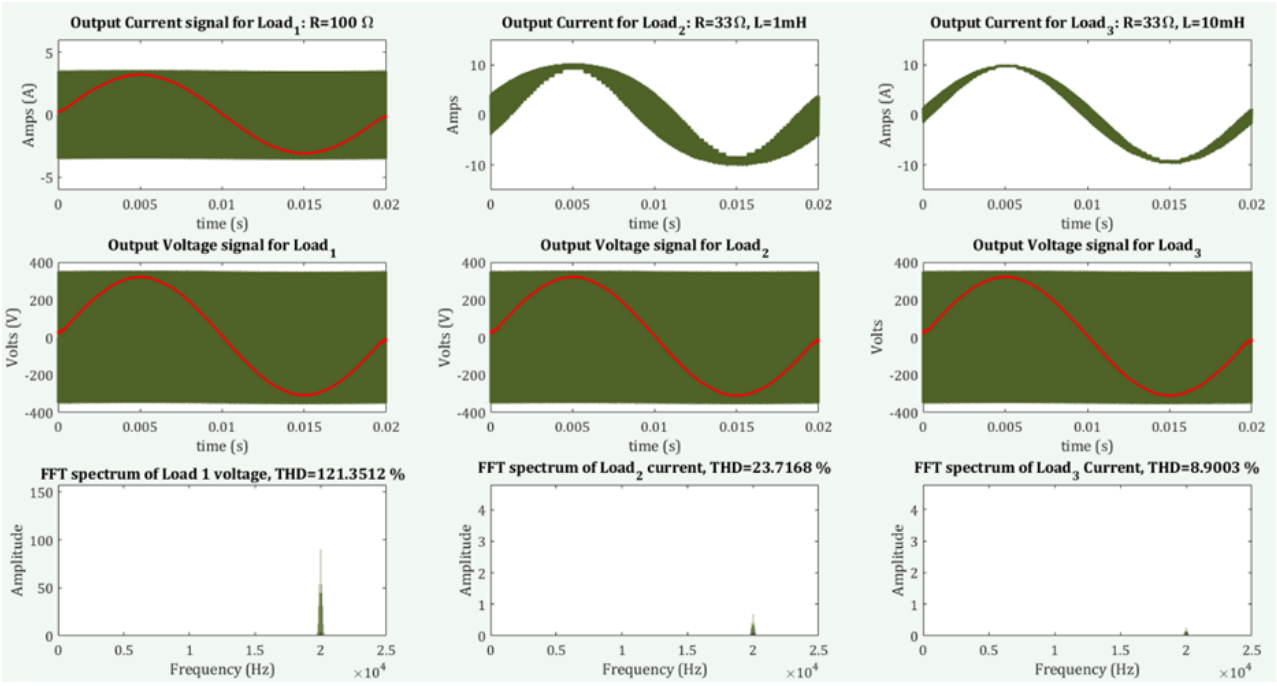

Figure II-11: Results of MIL simulation with Bipolar SPWM controller using Simulink.

II.8.2. Unipolar SPWM structure:

For Unipolar SPWM signals in H-bridge switches S_{11}signal need to be leading S_{21} by 180° *and* $S_{11} = \overline{S_{12}}$ and $S_{21} = \overline{S_{22}}$, these conditions can be satisfied by using two sinusoidal references, reference 1 for S_{11}'s leg signal is leading reference 2 for S_{21} *'s leg* by 180°then compare each reference to a 20kHz sawtooth signal generated from a Simulink generator block as a carrier signal as shown in Figure II-12, we set the modulation index m_a to 0.9 and the modulation frequency index m_f to 400: $m_f = \frac{20kHz}{50Hz} = 400$

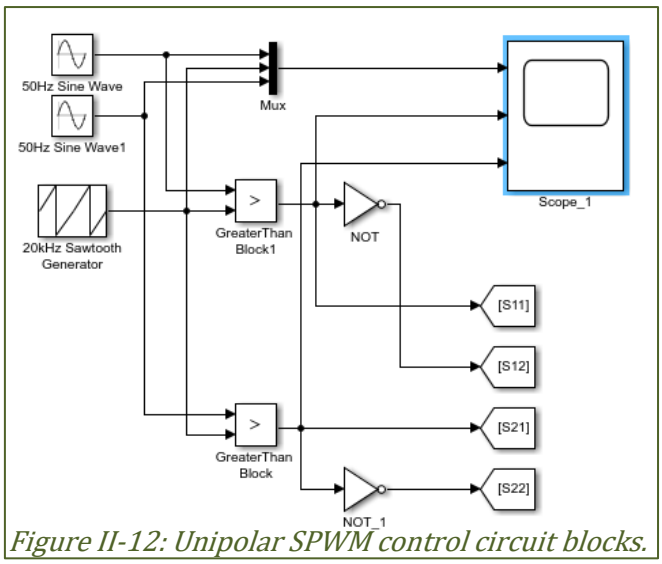

Figure II-12: Unipolar SPWM control circuit blocks.

The output of the scope in Figure II-12 is observed in Figure II-13 As we can see the duty cycle of S11and S22 is proportional to the red reference signal, contrary to S12 and S21 which is proportional to the blue reference signal, these values have been saved as a reference for the output from the real FPGA-Based controller.

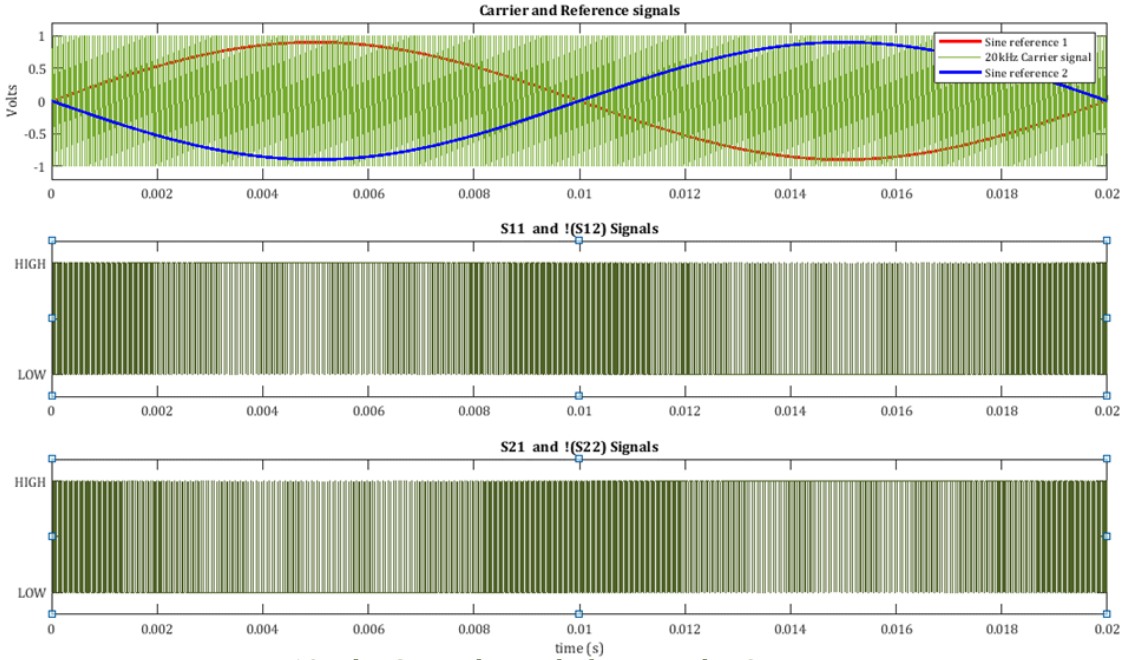

Figure II-13: The Control signals for Unipolar SPWM inverter.

The results: Figure II-14 shows the obtained results, the unipolar SPWM allows the appearance of a third voltage level and is necessary for a three-level inverter using an H-Bridge inverter. The overall THD is lower the bipolar strategy, reaching 63.28% and as always, the loads with inductive component, shows better results for the THD of the output current, 13.5154% and 3.7% , these results were saved as reference for next HIL simulation.

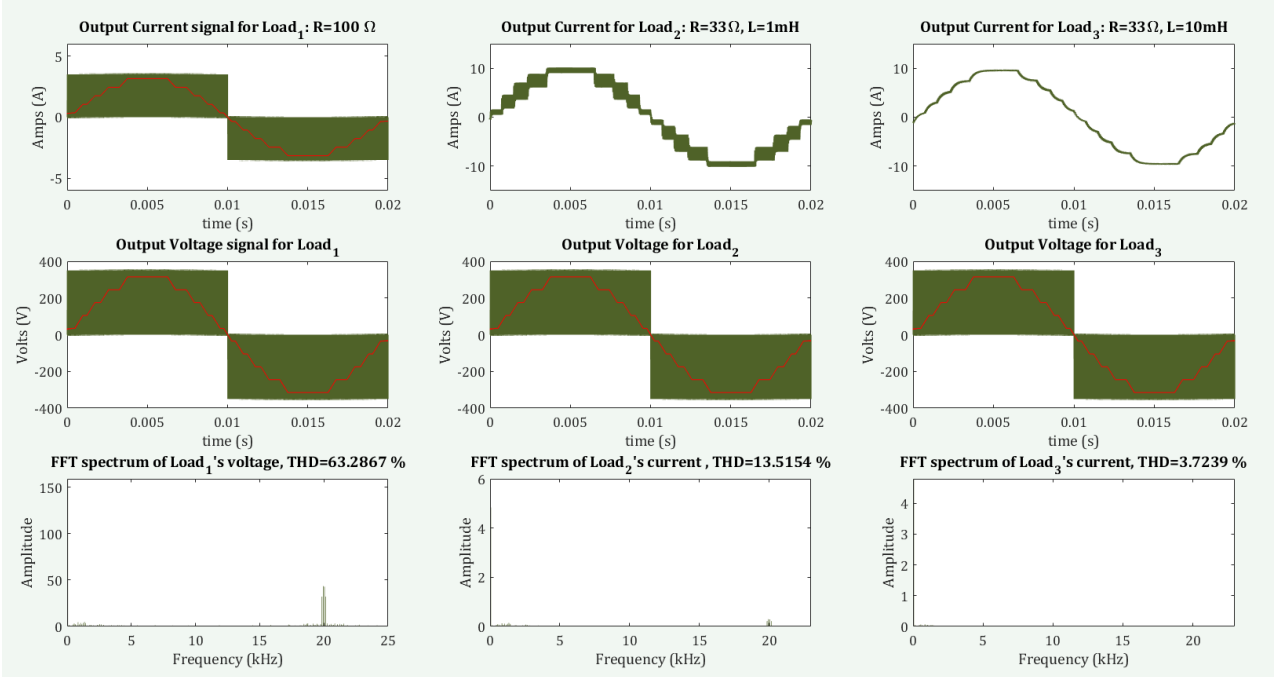

Figure II-14: Results of MIL simulation with Unipolar SPWM controller.

II.8.3. Selective Harmonic Elimination PWM structure:

As mentioned in the theoretical chapter, SHEPWM output is chopped N time every half cycle, thus, to generate SHEPWM output signal 2N switching angles (or firing angles) need to be defined. The challenge in SHEPWM is to solve the set of non-linear equations, in this project we set N=5, we used numerical methods to solve the set of five equations then compare our result with reference [26] and finally use the suitable results,

Figure II-15 (a) is a graph represents calculated Firing Angels versus Modulation Index. and graph (b) represents Modulation Index versus THD values. We can observe that the minimum THD value is at m=1.022 with THD=47.85%. therefore, we will use its Firing angles values in the rest of the simulation because this value is more suitable than the one in [26].

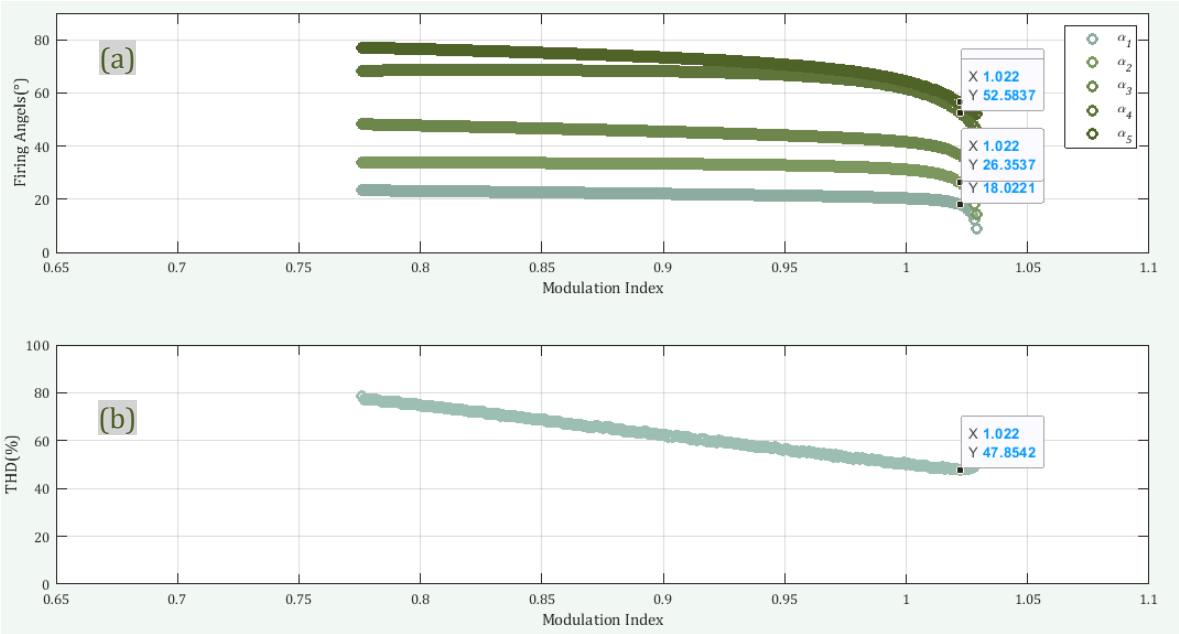

Figure II-15: Graph represents the data obtained after solving the firing-angles equations
a) Graph represent calculated Firing Angels versus Modulation Index.
b) Graph represents Modulation Index versus THD values

For SIMULINK simulation to create and generate the control signals that feed the gates S11, S12, S21, S22 we used the obtained firing angles as reference and compare each with sin wave carrier, the output of comparison is summed together to obtain the SPWM control signals as seen in Figure II-16

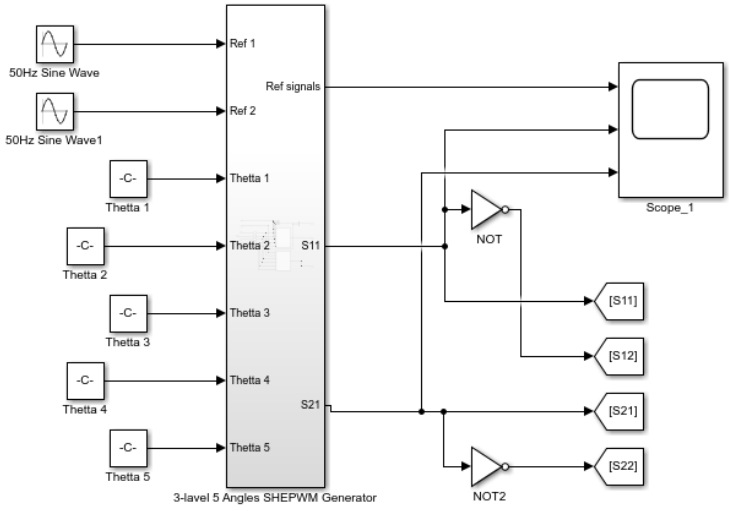

Figure II-16 SHEPWM control circuit blocks.

The output of the scope in figure II-16 is observed in Figure II-17, As we can see in Figure II-17, the signals S11and S22 reflect the positive part of the reference waveform while S12 and S21 reflect the negative part of the waveform. which represents the expected control signals, these values have been saved as a reference for later.

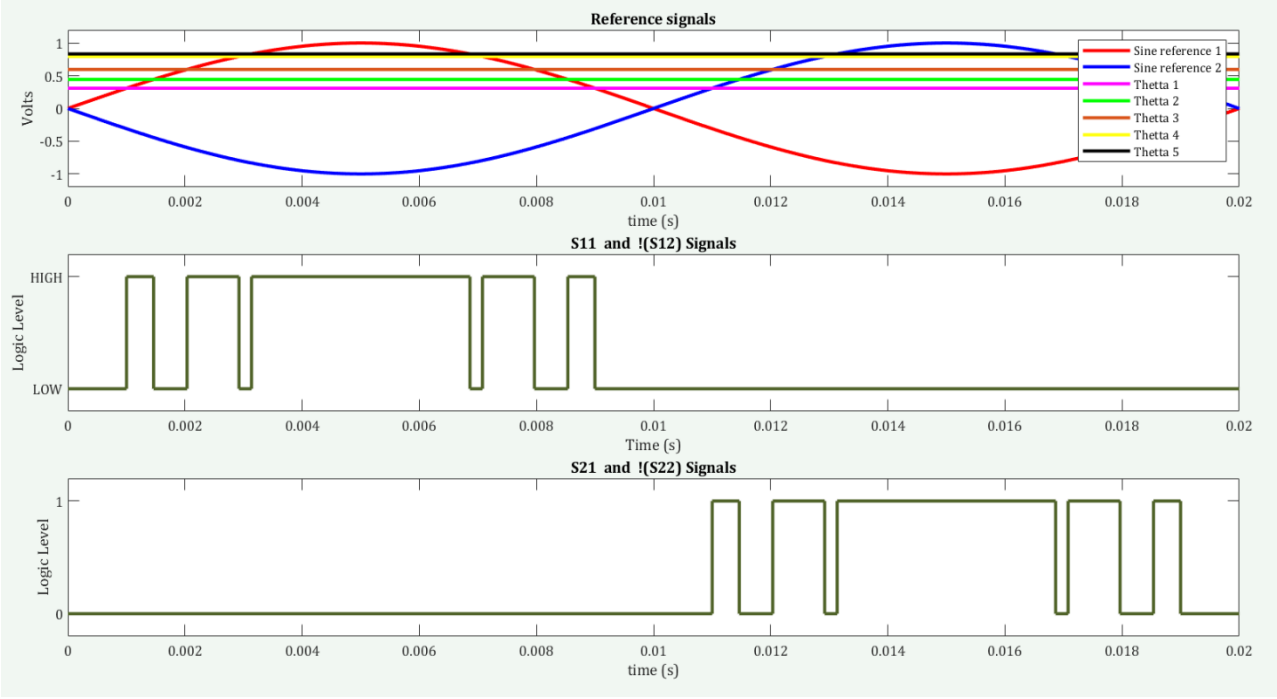

Figure II-17: The Control signals for SHEPWM inverter.

The results: Figure II-18 shows the obtained results of the simulations, this technique allows the voltage's THD to reach 47% and the current's THD to reach 3,4% when the inductance value is 10mH.

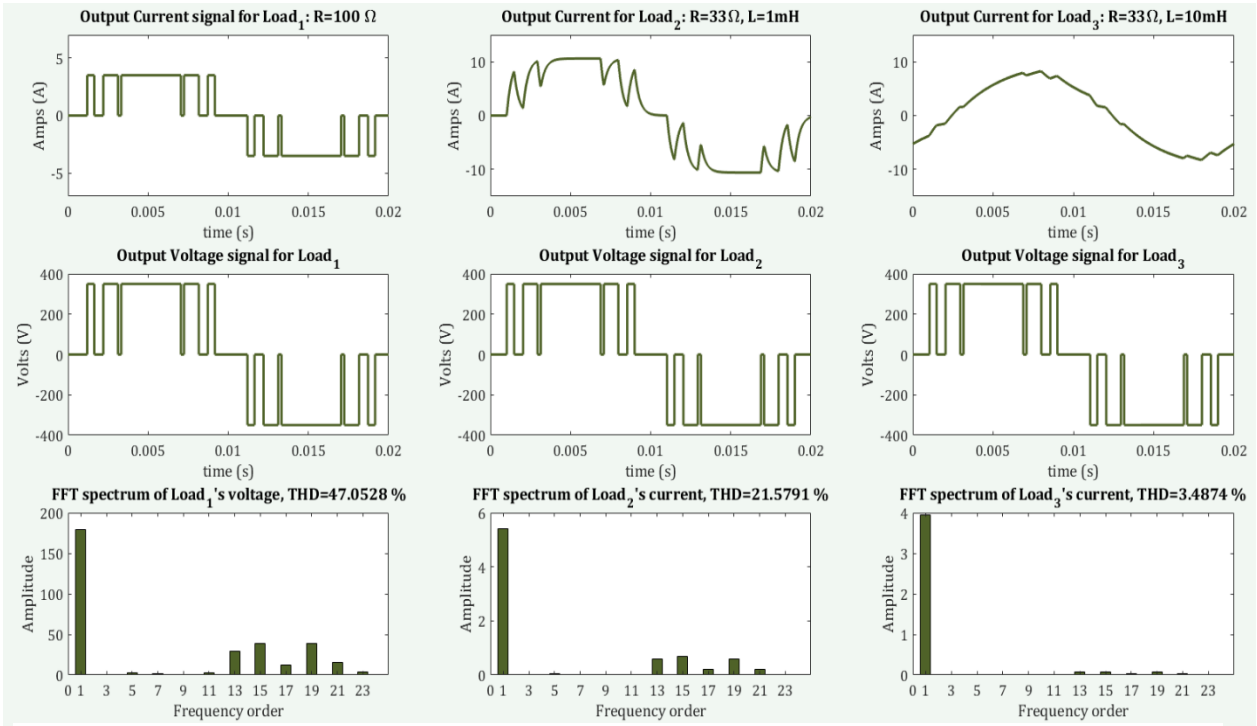

Figure II-18: Results of MIL simulation with SHEPWM controller.

II.9. Hardware Setup of The Experiment:

The next step is to generate real gating signals using FPGA board. These days most small and medium-sized three-level inverters' switching frequency is more than 10 kHz [41]. If we want to carry out accurate and real-time hardware-in-the-loop simulation test, the simulation based on CPU cannot meet the requirements, while the simulation based on FPGA is needed, the minimum simulation step length is 2us [42] which can easily be generated by FPGAs, after generating these real-time signals we used the oscilloscope as shown in Figure II-19 to test and compare the output signals with the MIL results that were presented earlier in this chapter.

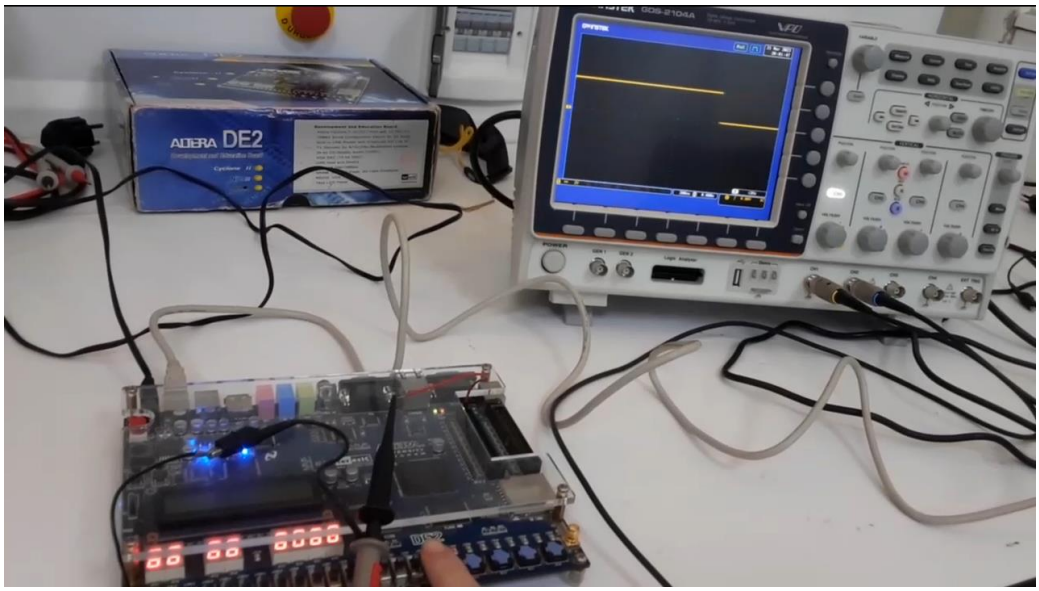

Figure II-19: Testing FPGA output using oscilloscope.

After testing the controller's output, the hardware setup shown in Figure II-20 for HIL simulation has been adopted: as mentioned previously we used op4510 as the Real-time simulator, we used a standard computer as host, we used FPGA as a programable chip to design the controller of the three-level inverter,

We used standard jumping wires to connect the output of the controller to the simulator, the controller basically will generate PWM signals therefore we used the DB37 digital inputs in op4510 as shown in Figure II-4. The DB37 digital inputs pins theoretically consider the signal between 5 to 22 volts as High voltage level but in practice it detects 2 volts and consider it as digital High voltage level.

Figure II-20: The Final Hardware Setup of the experiment.

The cable connection shown in Figure II-20 is presented in Table II-4 also it is clear shown in Figure II-21:

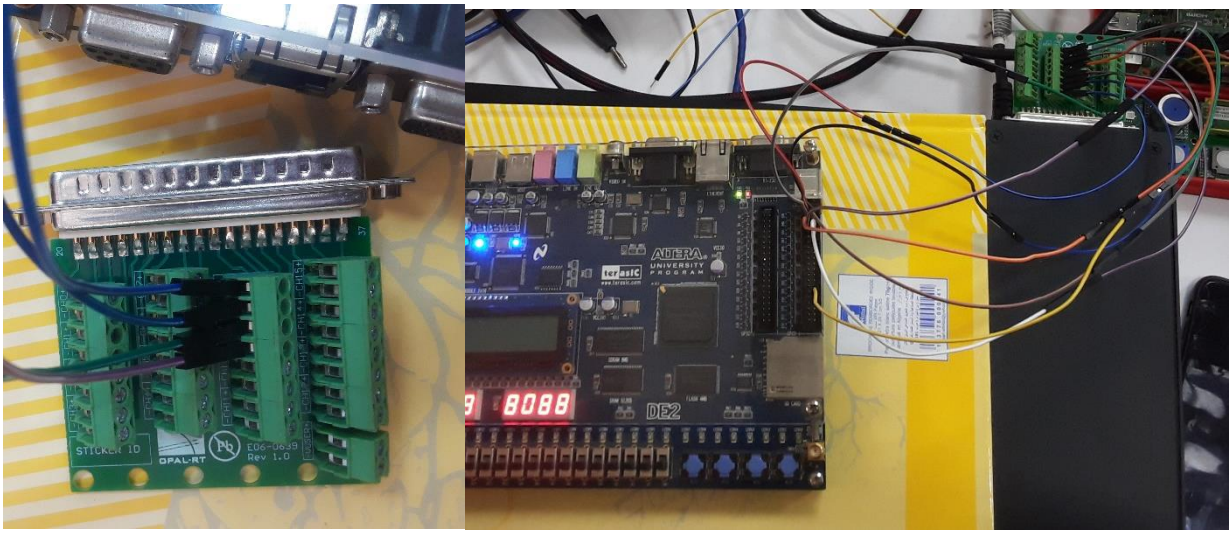

Figure II-21: the cable connections of the OP4510 and DE2 Board

Table II-4: Pin connections of the OP4510 and DE2 Board

Pin connection	
DE2 Board Pin	**OP4510 pin**
GPIO_1[0], PIN_K25	Slote A, CH8+
Gnd	Slote A, CH8-
GPIO_1[2], PIN_M22	Slote A, CH9+
Gnd	Slote A, CH9-
GPIO_1[4], PIN_M19	Slote A, CH10+
Gnd	Slote A, CH10-
GPIO_1[6], PIN_N20	Slote A, CH11+
Gnd	Slote A, CH11-

II.10. Conclusion:

This chapter has dealt with the main hardware used for the implementation of this project. The devices are available at the electronic laboratory of the Direction Centrale Recherche et Développement (DC-RD) Boumerdes.

In this chapter we presented: MATLAB and Simulink, this software was used for the realization of MIL.

In this chapter we presented: the results of the MIL simulation that were used as reference for the next HIL implementation in this project.

Chapter III. Hardware In the Loop implementation and Results:

III.1. Overview:

In this chapter we will describe the software used in the HIL simulation, we will also discuss the results of the HIL simulation as well as the VHDL design of the three controllers design that we implemented and highlight the best design base on the results.

III.2. Quartus II:

The Altera Quartus II design software is a multiplatform design environment that easily adapts to your specific needs in all phases of FPGA and CPLD design. Quartus II software delivers the highest productivity and performance for Altera FPGAs, CPLDs, and HardCopy ASICs (Application Specific IC) [27].

Figure III-1: Quartus II Logo [37]

Quartus II software delivers superior synthesis and placement and routing, resulting in compilation time advantages. Compilation time reduction features include:

- Multiprocessor support.
- Rapid Recompile.
- Incremental compilation.

The design can begin as HDL or a schematic. The MegaWizard™ Plug-In Manager helps creating or modifying design files that contain custom megafunction variations, which can then instantiate in a design file.

In Quartus II it is possible to perform functional and timing simulation of design with the ModelSim-Altera Edition software, or any EDA simulators supported by Quartus II software. The Quartus II NativeLink feature allows to run third-party simulator and other EDA tools from within Quartus II software [27]. Figure III-1 shows Quartus II's logo.

III.3. RT-LAB Software:

RT-LAB is OPAL-RT's real-time simulation software combining performance and enhanced user experience. Fully integrated with MATLAB/Simulink, RT-LAB offers the most complex model-based design for interaction with real-world environments. It provides the flexibility and scalability to achieve the most complex real-time simulation applications in the automotive, aerospace, power electronics, and power systems industries. RT-LAB handles everything, including code generation, with an easy-to-use interface. any Simulink® model can become an interactive real-time simulation application [28] [21] Figure III-2 shows RT-LAB's logo.

Figure III-2: RT-LAB Logo [38]

RT-LAB's acquisition system acts like a virtual oscilloscope, by allowing the user to visualize waveforms in real-time without glitches and data loss. It can run continuously or based on user-configured triggers to capture specific events. Its core engine provides accuracy and the bandwidth for the most demanding real-time applications with hundreds of channels and microsecond precision. With the possibility of recording thousands of measurement points, RT-LAB provides users with a complete data history. The format is compatible with other simulation and post-processing software. These advanced acquisition features are done without disturbing the real-time simulation and by keeping a maximal availability for executing the user's model [21].

III.4. Description of The Software Setup:

In the software designing part, we used Quartus II for coding, compiling and synthesis generation of HDL code, we used VHDL as hardware description language then we route, and load the code to the FPGA.

To test the output and debug the VHDL code we used the oscilloscope as shown in Figure II-3.

After checking that the signals behave as expected we connected the FPGA outputs to the output ports of the OP4510 as shown in Figure II-19

III.5. Building HIL Model on RT-LAB

III.5.1. Overview

Building an HIL simulation in RT-LAB involves creating a model that includes Simulink blocks, RT-LAB library's blocks and respect some specific requirements. In general, Any Simulink model can be implemented in RT-LAB, but some modifications must be made to distribute the model and transfer it into the simulation environment.

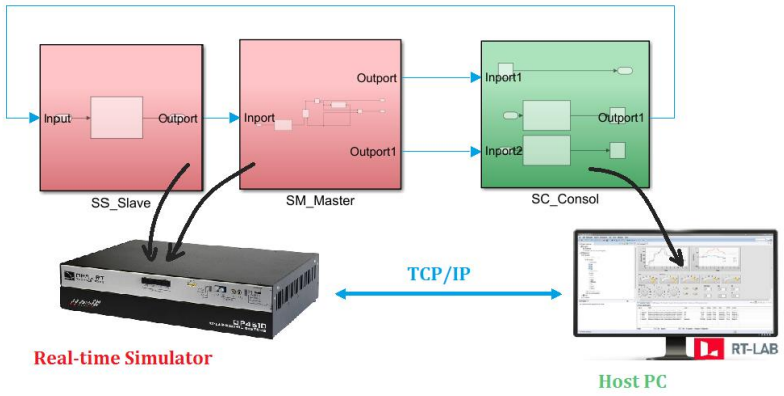

Figure III-3: Opal RT simulator's model subsystems.

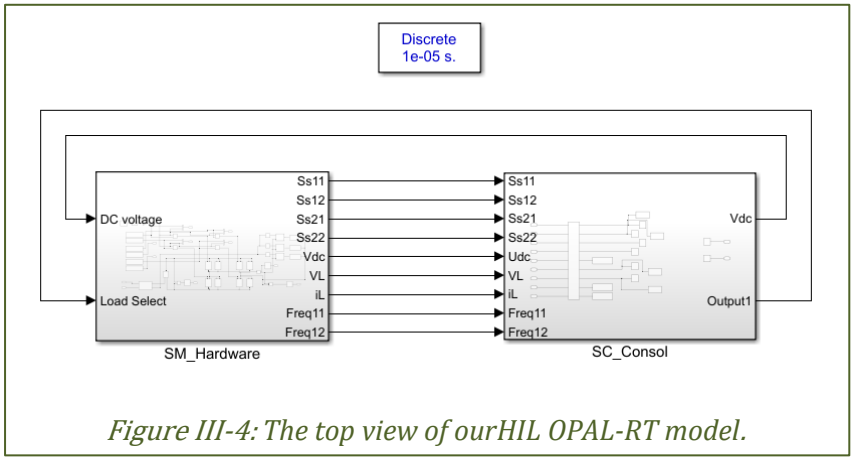

Figure III-4: The top view of ourHIL OPAL-RT model.

In RT-Lab the model Top-view needs to be composed of subsystems, these subsystems serve two purposes: separating computational blocks from graphical interface blocks and assigning computational subsystems to individual CPU cores.

RT-LAB's model consists of two types of subsystems: as shown in Figure III-3, a one that contains the computational elements of the model, (named Master & Slaves) and a consol subsystem that is used as a graphic interface subsystem.

The computation subsystem will be executed in real-time on the CPUs of Realtime simulator, Each computational subsystem will run on different CPU, where Consol subsystem will be displayed on the Host PC as illustrated in Figure III-3.

In top-view there must be only one Master subsystem but multiple Slave subsystems could be added; depending on the number of targets in the model;

The data between the computational subsystems and GUI subsystem will be exchanged asynchronously through the TCP/IP link [25].

The top view of our RT-Lab model is shown in Figure III-4, we used only the master subsystem because we have only one target and it is sufficient to use only one CPU core.

III.5.2. Designing The H-bridge model for HIL simulation:

In our case we used the same H-bridge model (the switches and loads): Load 1 is R=100Ω , Load2 is R=33Ω, L_1=10mH and Load3 is L_2=100mH R=33Ω, as shown in Figure II-8, all inside the master subsystem, instead of using GoTo block we used the OpInput blocks from RT-LAB library as shown in Figure III-5, these last calculates the duty-cycle and the frequency of the signal received from the PWM digital input pins of the op4510,

The duty-cycle values range from 0 to 1 maximum; to turn these values back to the SPWM signal we are able to use multiple methods, the one that we used is to set the duty cycles signals from OpInput blocks as a reference to SPWM and compare it with sawtooth carrier that have the same frequency as the original SPWM signal coming from the controller then perform the comparison to generate the SPWM signals as shown in Figure III-5,

As mentioned previously we tested 3 different loads, to switch between them we used an input-port to select the load easily during simulation from the console, also there is another input-port to change the value of the supply's voltage, all the inputs of the subsystems need to be connected to OpComm block as shown in Figure III-5.

All the outport in Master subsystem (numbered from 1 to 9) where sent to Consol subsystem as inputs and inside the subsystem they were connected to a scope

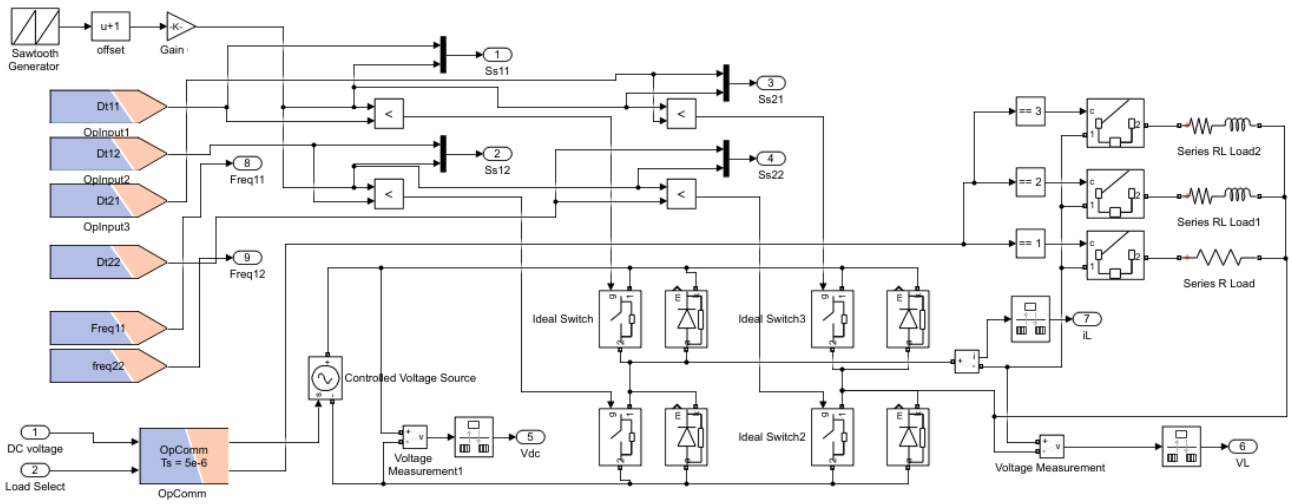

Figure III-5: The design of the Master subsystem

and to a 'To File' block to save their values as shown Figure III-6.

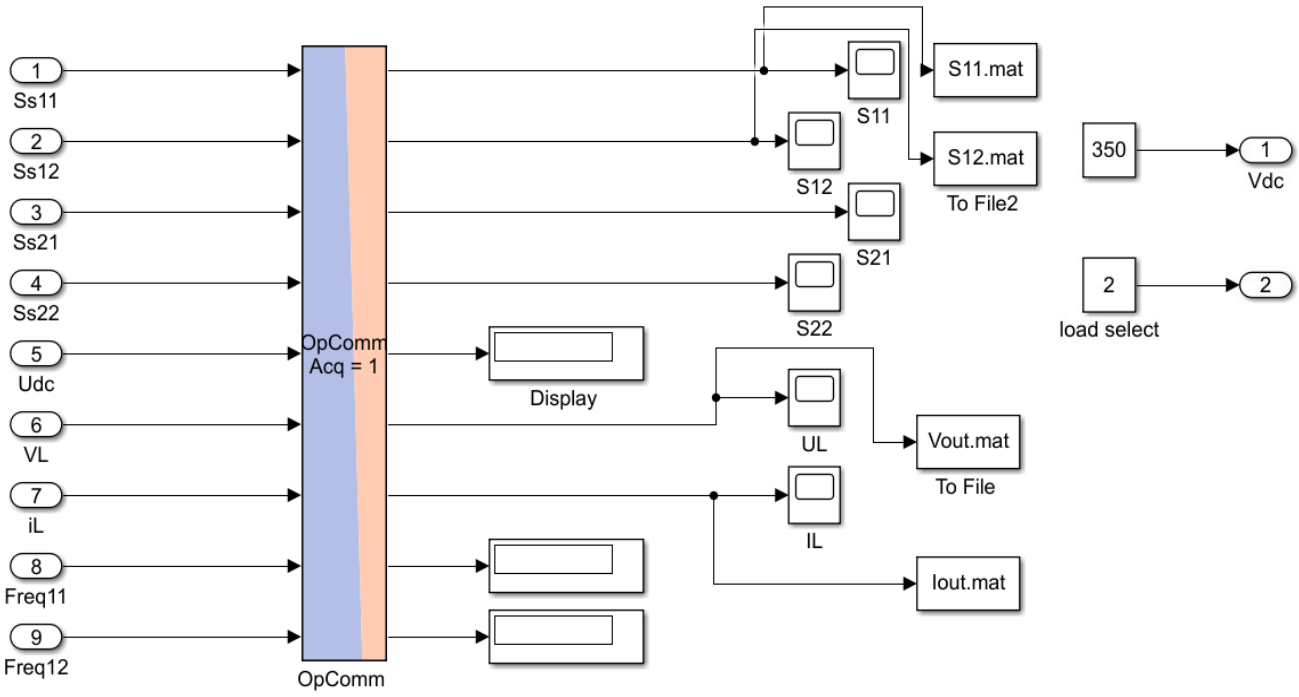

Figure III-6: The design of the Consol subsystem

III.5.3. The implementation of HIL simulation in RT-LAB:

After finish editing the model on Simulink we can run it off line this means without the preset of the real time signals from the RT-LAB library blocks, when everything run without errors we return to RT-Lab software then build and compile the model from the main interface as shown in Figure III-7, after correcting the compilation errors we assign the subsystems by selecting which target nodes each subsystem is executed, then we load and execute the model in real-time, when our model and RT-LAB are running, we can interact with our model by using the Simulink Console.

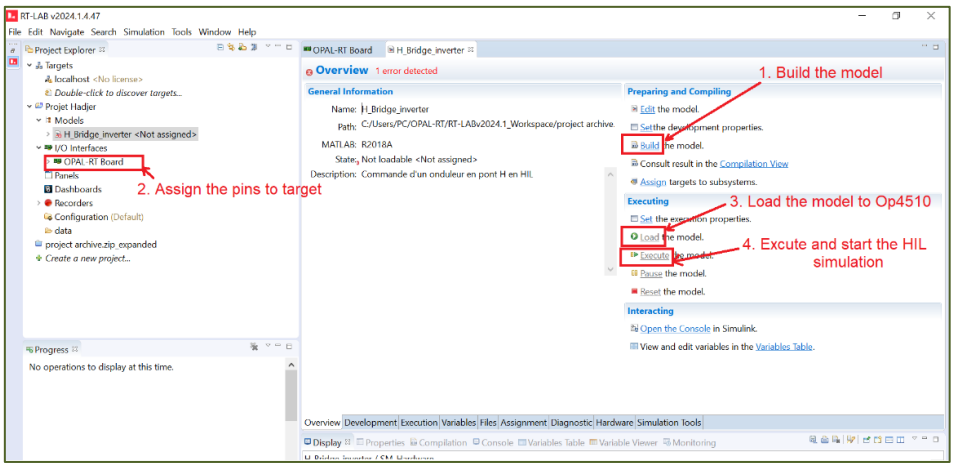

Figure III-7: RT-LAB main interface

III.6. The Software designing Setup:

In the software designing part of the controller, we used VHDL as hardware description language, we used Quartus II for coding, building, synthesis generation and loading the code to FPGA board, for testing and debugging the VHDL code: first we plotted the output gating signals on an oscilloscope as shown in Figure II-19 then compare it with the signals obtained in MIL simulation, when we found similarity between the two waveforms we connected the output signals from the FPGA to the input pins of the op4510 and start the HIL simulation from the RT-LAB software.

III.6.1. Bipolar Sinusoidal Pulse Width Modulation:

As mentioned previously, the goal is to generate a Bipolar SPWM control circuit for H-bridge controller with

$$m_f = \frac{20kHz}{50Hz} = 400 \qquad \text{and} \qquad m_a = \frac{V_{r,peak}}{V_{c,peak}} = 0.9$$

we decided to build a Bipolar SPWM circuit similar to the one in Figure II-9 using VHDL, both designs contain same elements: the final design contain four main components (blocks) in top-level:

'Clock_Divider' (or inst_1), 'STEP' (inst_2), 'LUT' (inst_3) and 'pwm' (inst_4) as shown in Figure III-8, we divided the system into four sub-blocks to simplify the design and facilitate the debugging process moreover every block can be used for later project, the blocks are presented one by one in the following figure:

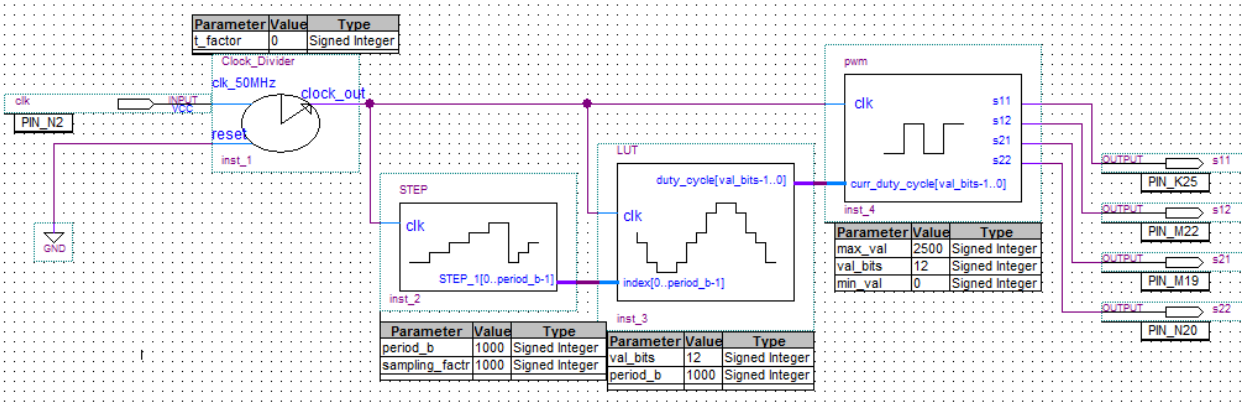

Figure III-8: Bipolar SPWM circuit schematic

The 'pwm' block: as shown in Figure III-9, 'pwm' block contains two inputs labeled clk and curr_duty_cycle[val_bits-1..0] of type STD_LOGIC and STD_LOGIC_VECTOR, and have four outputs: s11, s12, s21 and s22 of type STD_LOGIC, this block contains internal counter that increases every clock cycle (every clk' event) between max_val and min_val then at every clock event it compares the counter's current value with the input signal curr_duty_cycle[val_bits-1..0] then generate PWM signal on s11 output based on the comparison; this block generates Bipolar PWM gating signals: the outputs . s11 and s22 are the same while s12 and s21 are their inverse as mentioned previously.in theoretical chapter, the period of the internal counter or digital sawtooth signal is determined by max_val and min_val and the input clk. Figure III-10 represents the functional simulation results of the pwm block with clk=50Mhz, and max_val =500 and min_val =0

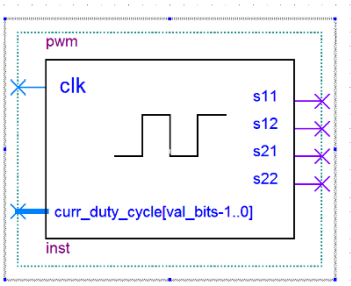

Figure III-9: pwm block diagram

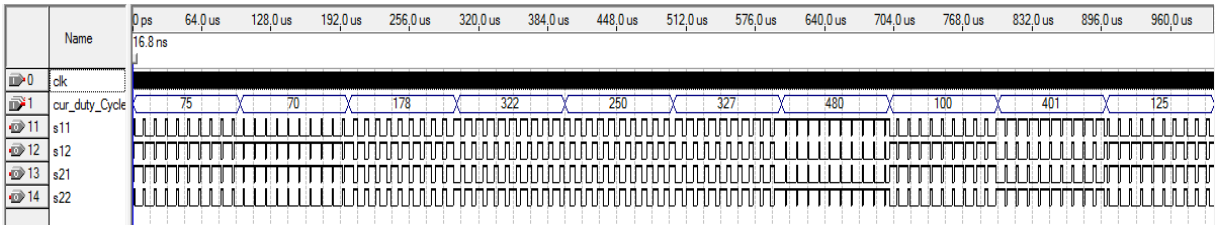

Figure III-10: Functional simulation results for pwm block

The 'STEP' block: as shown in Figure III-11, this block contains one input clk and one output bus: STEP_1[0..period_b-1] this block is a counter that increases from 0 to period_b-1and output the value every clk event multiplied by sampling_factr. Figure III-12 a) b) c) represents the functional simulation results of the STEP block when the pair (period_b, sampling_factr) equals= (1000,1000), (1000,5000) and (4,5000) respectively.

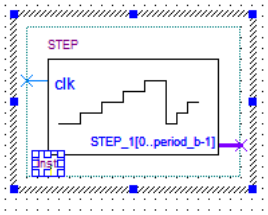

Figure III-11: Block diagram of STEP Block

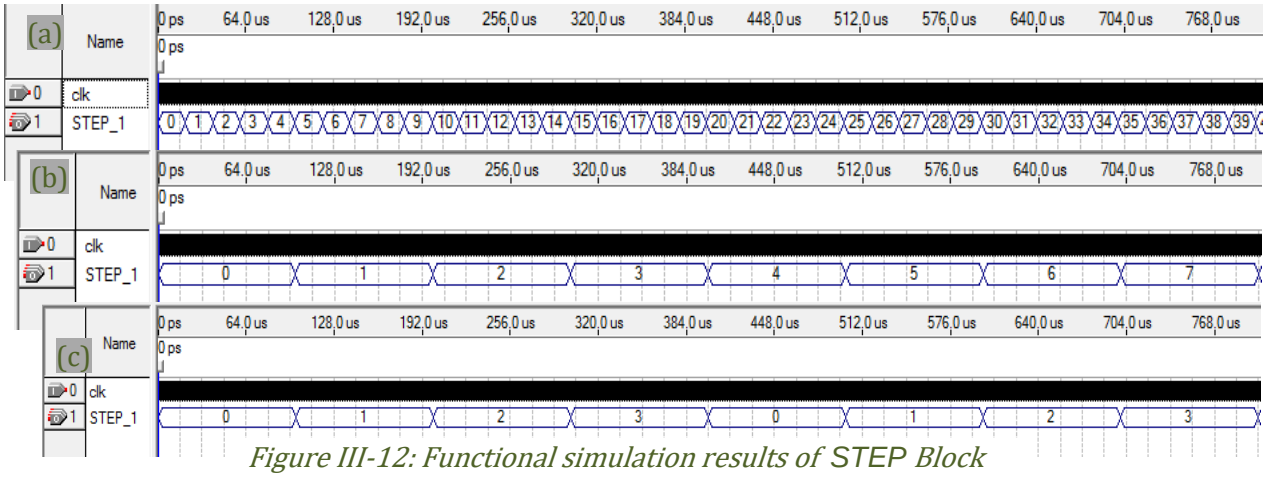

Figure III-12: Functional simulation results of STEP Block

The 'LUT' block: this block as shown in Figure III-13 contains two inputs clk and index[0..period_b-1] of type STD_LOGIC and SIGNED INTEGER respectively and one output bus: duty_cycle[val_bits-1..0] of type STD_LOGIC_VECTOR this block represent a direct Look-Up Table, i.e. a block of index-addressable memory of width equals to val_bits and a period_b depth as shown in Figure III-8 , in our case we stored digital sine wave samples, we calculated these samples using MATLAB. At every clock cycle (every clk' event) the block outputs the value stored at the index presented at the input bus index. Figure III-14 represents the functional simulation results of the LUT block with val_bits=12 and period_b=1000, and the digital sine values range from -1125 to 1125, and we add to them an offset equals to 1250 ⇒ the values range from 125 to 2375

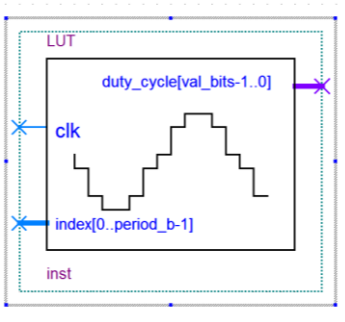

Figure III-13: LUT block diagram

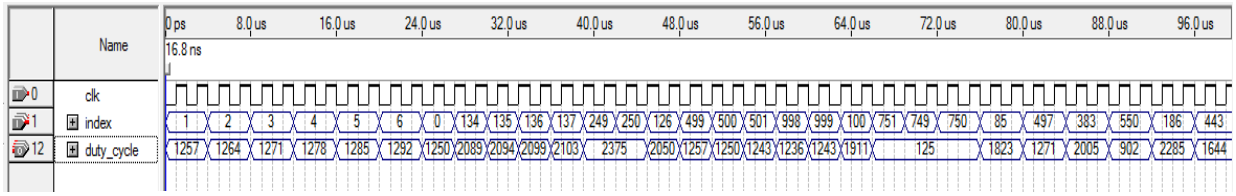

Figure III-14: Functional simulation results of LUT block

The 'Clock_Divider' block: as shown in Figure III-15, this block contains two inputs clk_50MHz and reset and one output: clock_out this block divides the frequency of the input clk_50MHz by twice the factor: t_factor and outputs the new clock at clock_out pin, when t_factor=0 the output is the same clk_50MHz signal. Figure III-16 (a) (b) and (c) represent the functional simulation results of the Clock_Divider block when t_factor equals 0, 1 and 5 respectively.

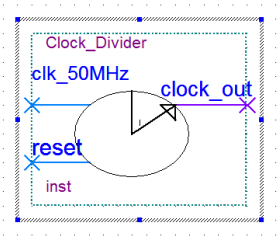

Figure III-15: Clock_Divider Block diagram

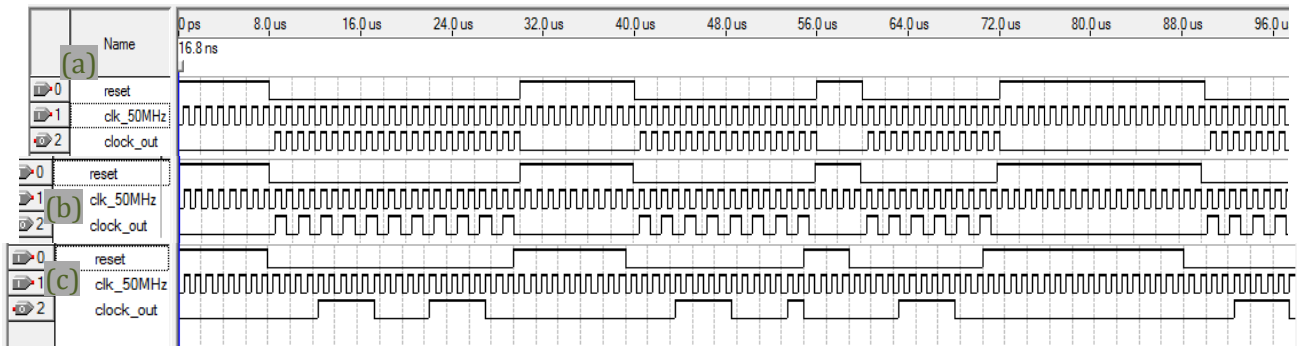

Figure III-16: Functional simulation results for Clock_Divider block

The pseudo code: we summarize the main functionalities of the above VHDL's Bipolar SPWM controller design in the next table:

Table III-1: The Pseudo code of the system:

1. Generate main clock signal
2. Increase STEP_1 internal counter every clock signal
2. In parallel update the output of the LUT table.
3. Compare the counter in pwm block with output from Look-Up Table block.
4. Generate the Bipolar PWM gating signal based on the output of the comparison.

The final Bipolar SPWM controller design as shown in Figure III-8 has two inputs labeled 'clk' and 'reset' and four outputs s11, s12, s21 and s22, In the design we refactored multiple parameters that we need to define them carefully to meet the requirements easily, for example in order to generate 20kHz digital sawtooth carrier: the frequency of 'clk_out' signal divided by the difference between max_val and min_val must equal 20kHz, Additionally, if we need to maintain modulation index m=0.9: the peak-to-peak value of the digital sinusoidal reference need to equal 0.9'time the difference between max_val and min_val and to have the same offset…etc., some other parameters have been chosen as reasonable as necessary such as the depth of LUT table and sampling_factr of STEP block since here performance is more important than precision [29]. In the first trials we used samples with m=1 to facilitate the calculations.

	To	Location	I/O Bank	I/O Standard	General Function	Special Function	Reserved	Enabled
1	clk	PIN_N2	2	3.3-V LVTTL	Dedicated Clock	CLK0, LVDSCLK0p, In…		Yes
2	s11	PIN_K25	5	3.3-V LVTTL	Row I/O	LVDS121p		Yes
3	s12	PIN_M22	5	3.3-V LVTTL	Row I/O	LVDS122p		Yes
4	s21	PIN_M19	5	3.3-V LVTTL	Row I/O	LVDS123p		Yes
5	s22	PIN_N20	5	3.3-V LVTTL	Row I/O	LVDS124p		Yes
6	<<new>>	<<new>>						

Figure III-17: Quartus II Pins Assignment settings table

After finish editing we compiled the VHDL design and assign the pins, we set the input clk to the internal 50MHz clock of the DE2 board and reset to GND, we set the outputs to the expansion Headers of the DE2 board as shown in Figure II-2 ,the final Pin assignment setting is shown in Figure III-17, then we compiled again then run the code using the programmer tool in Quartus II, the outputs gating signals s11, s12, s21 and s22 are presented in Figure III-18:

The gating signals appear to resemble the gating signals generated using Simulink shown in Figure II-10 , we can observe in the above graph that the frequency of the gating signal is 50Hz, we can also observe that s11 and s22 are similar while s12 and s21 are the inverse of s11 and that's exactly as expected, therefore we connected these gating signals to the op4510 to test theme in HIL simulator.

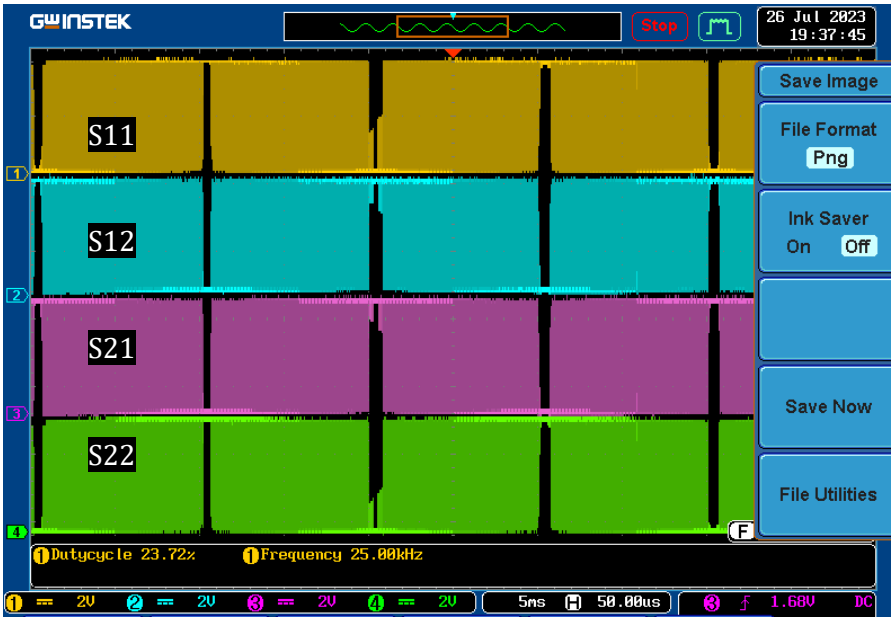

Figure III-18: Gating signals generated from the FPGA-Based controller

After connecting the controller, we executed the model from RT-LAB in real-time. The results are presented in the following:

The Results: As shown in Figure III-6, we putted multiple scopes in the consol subsystem to help us view the behavior of the signals during the HIL simulation.

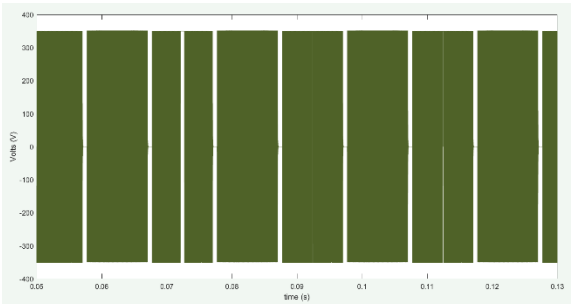

Figure III-19: waveform of the Output voltage across Load_1

Figure III-19 shows the waveform of the output voltage signal from the Load_1, the output appears as rectangular blocks. because the output waveform has a huge number of pulses. These pulses have either positive or negative amplitude and the time between pulses is small therefore

it appears as a continuous rectangular block. To easily detect the change in the duty cycle of the pulses in the output we calculated the mean of the output waveform using a window of 20kHz and plot it with original waveform as shown in Figure II-20, the figure also contains all the HIL simulation results of the Bipolar SPWM controller; we can observe the THD value of the output voltage of the R load is 145.80% while in MIL it was 112.38%, similarly in the R load the harmonics are significant at 20kHz. Similar to MIL, the THD value get decreased when we switch to other loads but it decreases up to 22.8454% instead of 3.497%, Additionally we can notice that the mean waveform is not pure sinusoidal as in the MIL, also in both the RL loads we can easily notice a strange peak in middle the negative cycle that repeats in every period , these phenomenon can appears due to multiple reasons, for hypothesis we can expect it is either a glitch in the LUT table or in STEP block of the FPGA controller or a noise or interference of an external electrical signals is corrupting the data.

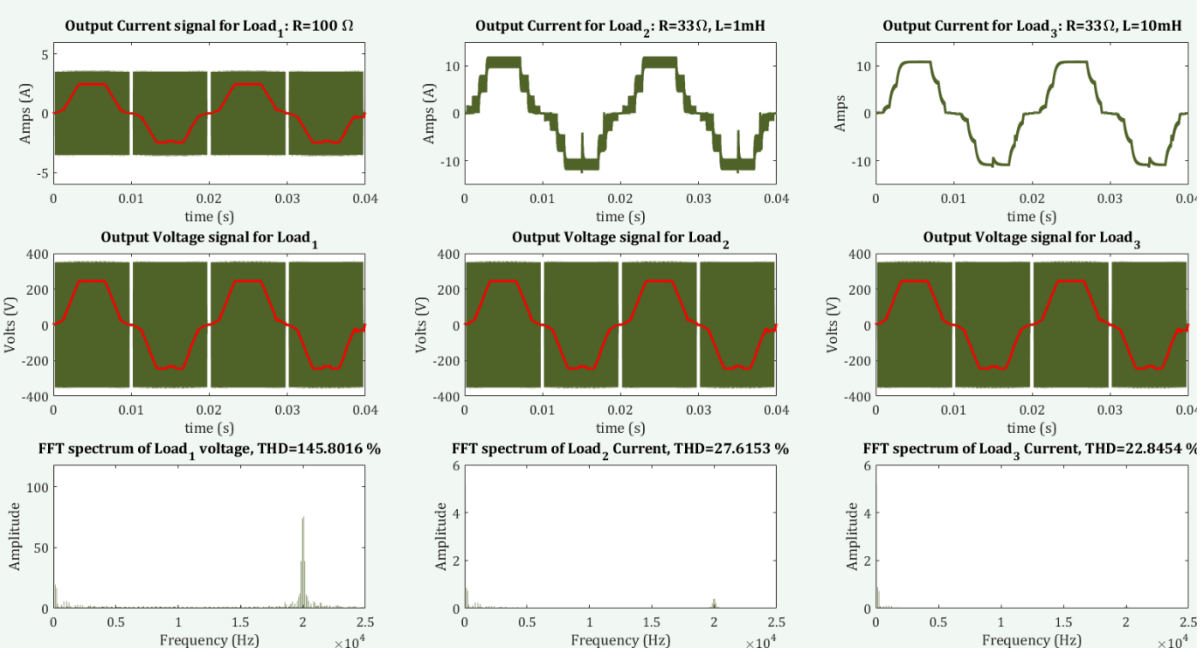

Figure III-20: Results of HIL simulation with Bipolar SPWM controller.

After obtaining these results we designed the Unipolar SPWM control circuit to enhance the THD values of the output voltage additionally we decrease the main clock frequency by a factor = 50 and use the same LUT block with minor changes, more details are in following,

III.6.2. Unipolar Sinusoidal Pulse Width Modulation:

We designed the Unipolar SPWM controlling circuit after obtaining the results from Bipolar SPWM circuit: as mentioned previously the Unipolar SPWM controlling circuit require two sinusoidal references that are out-of-phase, additionally each leg of the H-bridge inverter is controlled based on the comparison between the sawtooth and one of the references; we noticed that there are similarities between the Unipolar and Bipolar SPWM circuits except that the Unipolar SPWM have an addition reference than Bipolar SPWM circuits, therefore we used the same blocks from the Bipolar SPWM that we designed earlier to create Unipolar circuit using VHDL as shown in Figure III-21 : the final design contains four blocks in top-level: 'Clock_Divider' , 'STEP', 'pwm' and 'ULUT' . the blocks 'Clock_Divider' and 'STEP' are the same as previous design, the other blocks are described in the following:

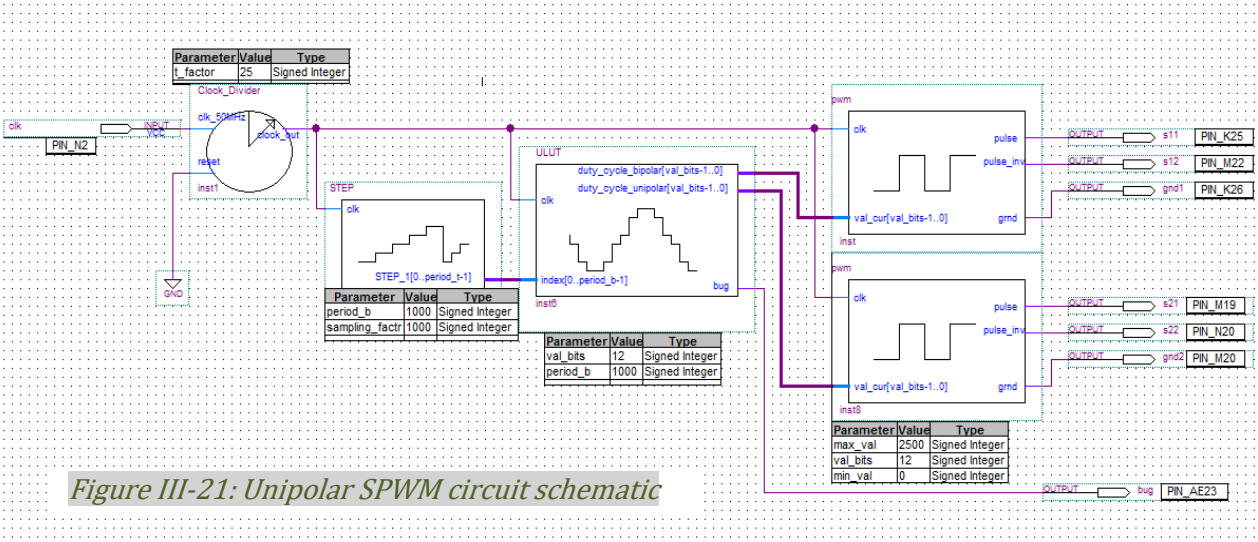

Figure III-21: Unipolar SPWM circuit schematic

The 'pwm' blocks as shown in Figure III-21, 'pwm' block contains two inputs labeled: clk and curr_duty_cycle[val_bits-1..0] of type STD_LOGIC and STD_LOGIC_VECTOR, and have three outputs: pulse, pulse_inv and gnd of type STD_LOGIC, this block contains internal counter that increases every clock cycle (every clk' event) between max_val and min_val then at every clock event it compares the counter's current value with the input signal curr_duty_cycle[val_bits-1..0] then generate PWM signal on pulse output based on the comparison; the output 'pulse' is either High or Low with duty cycle equal to curr_duty_cycle over the difference between max_val and min_val, while the output pulse_inv is the logical inverse of pulse signal, the gnd is always digital Low level and we used it to ground some pins in OP4510. Figure III-22 represents the functional simulation results of the pwm block with clk=50Mhz and max_val =2500 and min_val =0

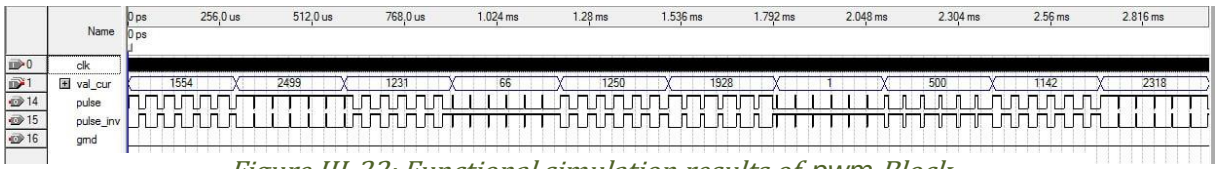

Figure III-22: Functional simulation results of pwm Block

The 'ULUT' block: this block as shown in Figure III-21 contains two inputs labeled clk and index[0..period_b-1] of type STD_LOGIC and INTEGER respectively and two output buses: duty_cycle_bipolar[val_bits-1..0] and duty_cycle_unipolar[val_bits-1..0] of type STD_LOGIC_VECTOR and one STD_LOGIC output labeled bug, this block is similar to the previous LUT, it represent a direct Look-Up Table of width equals to val_bits and a period_b depth and we stored inside it sine wave samples. At every clock cycle the block outputs in duty_cycle_bipolar the value stored at the index presented at the input bus index and outputs in duty_cycle_unipolar the value stored at the index period_b-1 – index.

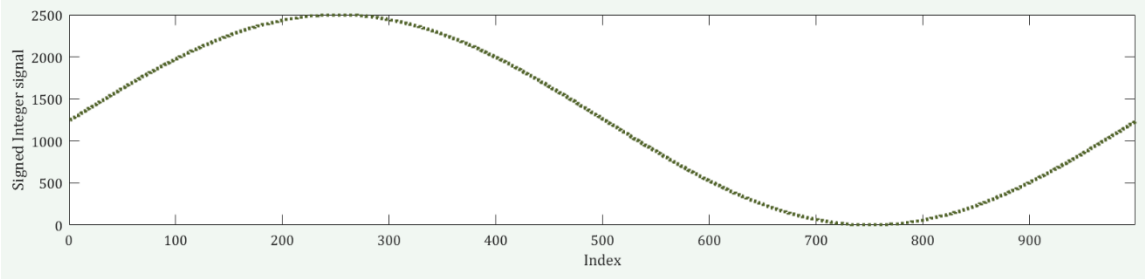

Figure III-23: Graph represent the samples stored in ULUT table versus their indices

In the outputs of the previous controller design we noticed that the output waveforms always have an odd peak in second half cycle, the first hypothesis we guessed was: maybe LUT block contain errors in the samples therefore we checked the sinewave samples multiple times and plot the samples values versus their index as shown in Figure III-23, inside this block there is only one case that might create outputs that is outliner and deviate from the other samples of LUT is when the input index have a value strictly greater than period_b or less than 0 , therefore we put the output bug to toggle each time the block goes to that condition. Figure III-24 represents the functional simulation results of the ULUT block with val_bits=12 and period_b=1000, and the digital sine values range from -1250 to 1250, we add to them an offset equals to 1250 ⇒ the values range from 0 to 2500.

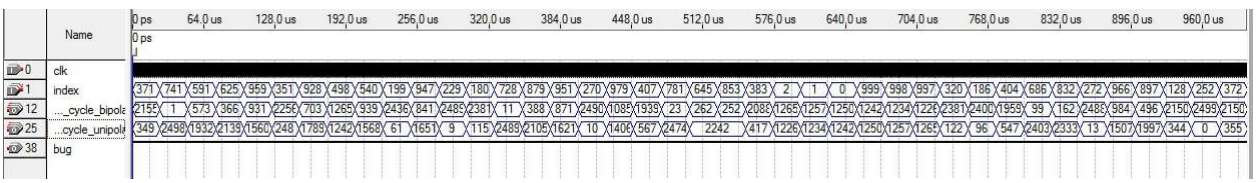

Figure III-24: Functional simulation results of ULUT Block

The Pseudo code: we summarize the main functionalities of the above VHDL's Unipolar SPWM controller design in the next table:

Table III-2: Psuedo code of the Unipolar SPWM system:

1. Generate main clock signal.
2. Increase STEP block's internal counter every clock signal.
2. In parallel update the output of the ULUT table.
3. Compare the counter in pwm blocks with outputs from Look-Up Table block.
4. Generate the Unipolar PWM gating signal based on the output of the comparison

As shown in Figure III-21 the final Unipolar SPWM controller design has two inputs labeled 'clk' and 'reset' and five outputs s11, s12, s21 and s22, and bug, in this design we also refactored multiple parameters that we can change to meet the requirements easily: this time we used samples with m=1 instead we changed the amplitude of the sawtooth carrier from the Gain block as shown in Figure III-5 until the duty cycle's peak-to-peak value is 0.9 times the sawtooth peak-to-peak value. Also we slowed the frequency of the system by a factor = 50 by setting t_factor=25 as shown in Figure III-21 thus the frequency of the FPGA-Based controller is reduced to 1Hz instead of 50Hz as shown in Figure III-25 (a), this is to insure that the errors that we got in the previous controller are not due to the fast frequencies of the system i.e. because the OP4510 require a dead time to perform calculation therefore if we slow down the whole processes: the simulator OP4510 will have enough time to calculate and present the output, additionally the RT-LAB have the option to control the time displayed during the simulation by a factor called Time Factor as shown in Figure III-26 in other words if we set Time Factor equals to 50 then one second in real time will be displayed as 50 seconds in RT-LAB , thus to calibrate the time we decelerated from controller's frequency by a factor =50 we set Time Factor=50 therefore the 1Hz frequency of the controller should be displayed as 50Hz in RT-LAB. Figure III-25 (b) shows that S11 is leading S21 by 180°.

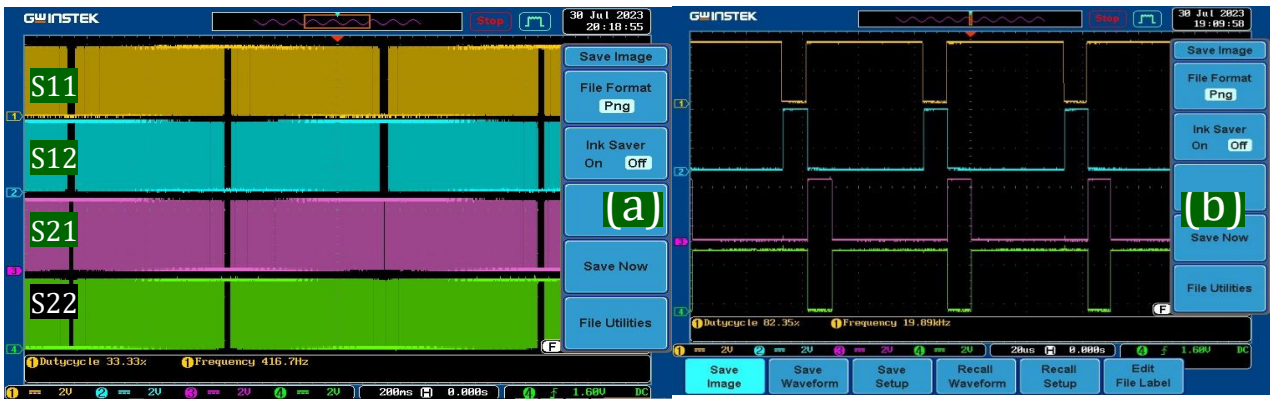

Figure III-25: Unipolar SPWM gating signals generated from the FPGA-Based controller

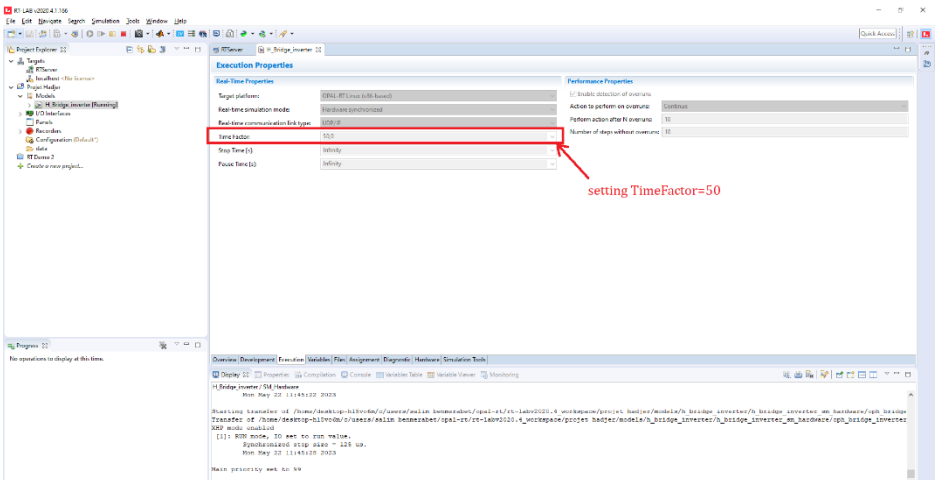

Figure III-26: Execution interface in RT-LAB platform

The Results: Figure III-27 represent the results of HIL simulation with Unipolar SPWM controller, we plotted in red color the mean value of some wave forms to help us observe the change of duty cycle of the output as shown in Figure III-27, we can observe the THD value of the voltage output across the Load_1 is reduced to 69.3599% it is close to the reference which was 63.286%, The THD of Load_2's current is 69.03% it is significantly larger than the reference MIL simulation, for Load_3's current's THD value was 27.475% beside 3.723%., although we calculated the sin samples stored in ULUT several times we still face the same noise of strange peaks in the outputs waveform but this time in both positive and negative half cycles although the bug LED did not blink during the Real-time simulation.

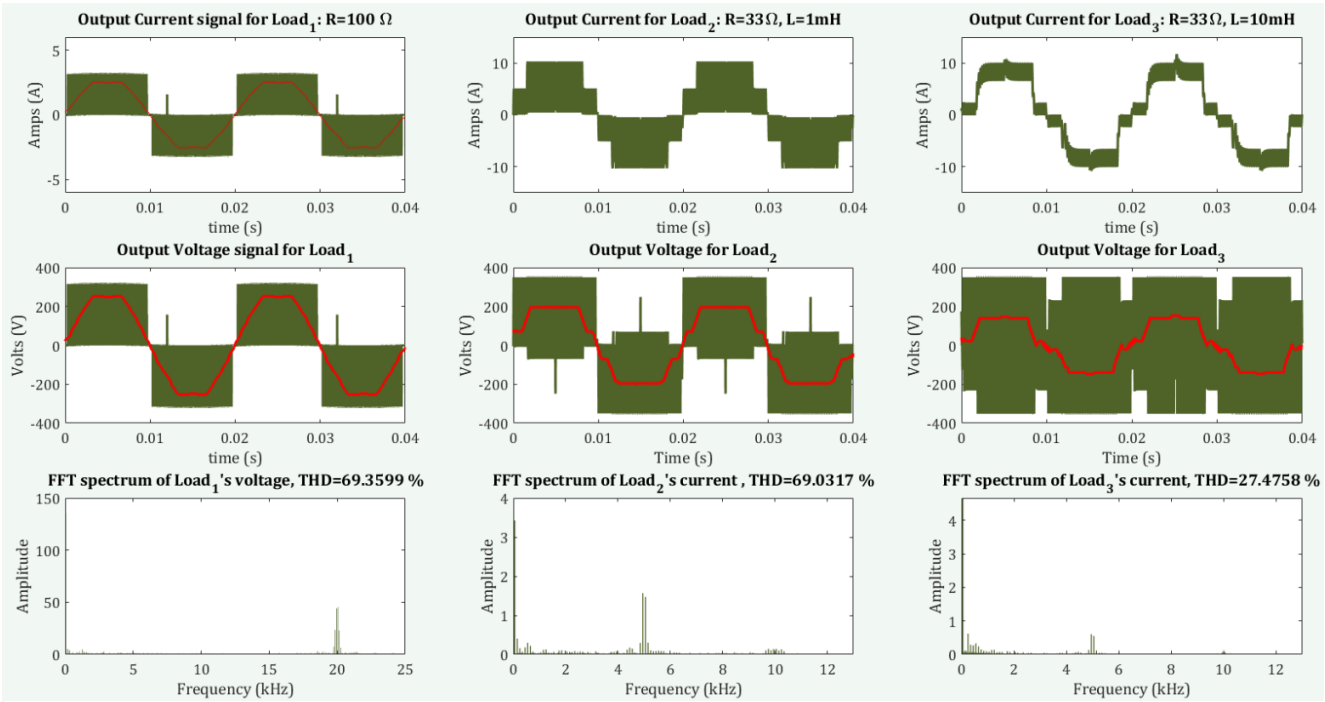

Figure III-27: Results of HIL simulation with Unipolar SPWM controller.

III.6.3. Selective Harmonic Elimination Pulse Width Modulation:

To design the SHEPWM control circuit we used the same calculated firing angles in the previous chapter. Unlike to the two previous circuits SHEPWM doesn't contain high frequencies carrier, the final design contain three components in top-level:, 'STEP' (inst_1), 'LUT_SHEPWM_3L' (inst_2) and 'Gating_Signals' (inst_3) as shown in Figure III-28, the STEP block is the same as in the previous designs, the rest of the blocks are described in the following:

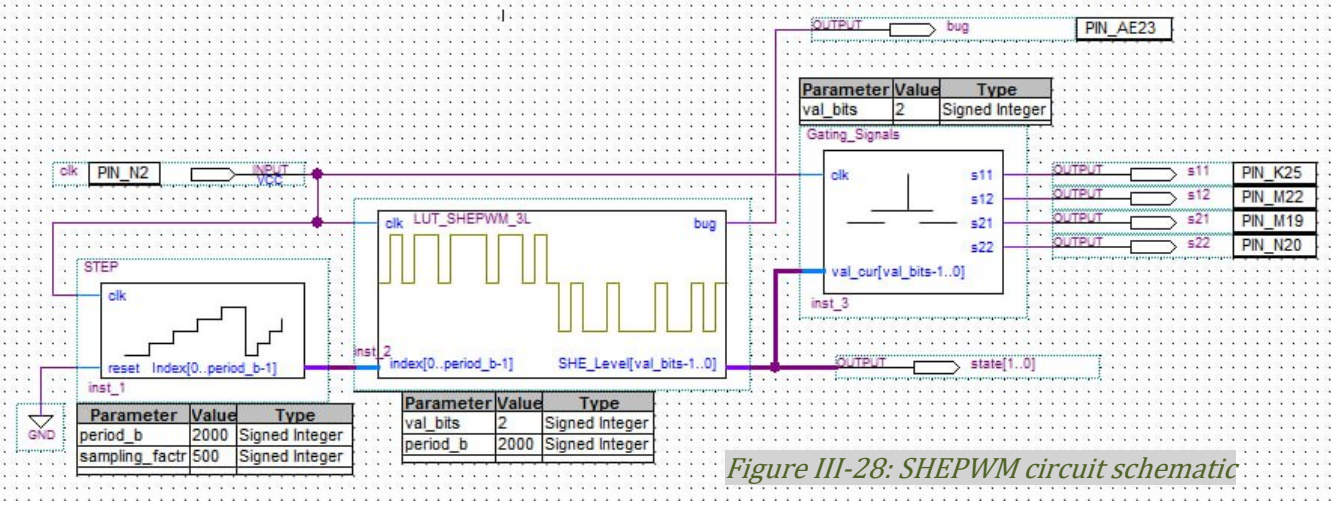

Figure III-28: SHEPWM circuit schematic

The 'LUT_SHEPWM' block: this block as shown in Figure III-29 contains two inputs clk and index[0..period_b-1] of type STD_LOGIC and SIGNED INTEGER respectively and one output bus: SHEP_Level[val_bits-1..0] of type STD_LOGIC_VECTOR this block represent a direct Look-Up Table, of width equals to val_bits and a period_b depth, we stored inside it the states of the SHEPWM reference signal, we calculated these samples using MATLAB. At every clock cycle the block outputs the value stored at the index presented at the input bus index.

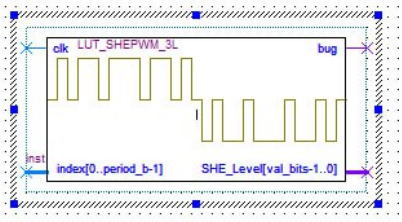

Figure III-29: LUT_SHEPWM Block diagram.

The 'Gating_Signal' block: as shown in Figure III-30, 'Gating_Signal' block contains two inputs labeled clk and curr_val[val_bits-1..0] of type STD_LOGIC and STD_LOGIC_VECTOR, and have four outputs: s11, s12, s21 and s22 of type STD_LOGIC, this block contains an internal multiplexer and an internal memory that contain n different sets of gating signals states, at every clock cycle the block selects a set of {s11, s12 ,s21, s21} outputs from the internal memory based on the state presented in the curr_val port.

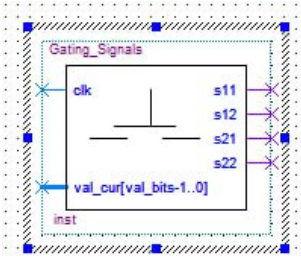

Figure III-30: Gating_Signal Block diagram

The Pseudo code: we summarize the main functionalities of the above VHDL's SHEPWM controller design in the next table

Table III-3: Psuedo code of the system:

1.	Increase STEP block's internal counter every clock signal
1.	In parallel update the output of the LUT table.
2.	Generate the SHEPWM gating signal based on the current state of the system

As shown in Figure III-28 the final SHEPWM controller design has two inputs labeled 'clk' and 'reset' and five outputs s11, s12, s21 and s22, and bug.

To generate the digital reference first we calculated the references samples since the output have low frequencies we can notice it has three different states each corresponds to one of the three-Level voltages ($-V_{dc}$, 0 and $+V_{dc}$), therefore we transformed the digital reference values to its corresponding states sequence of one period of the SHEPWM graph labeled as ('2', '1' and '0'} respectively, we stored the digital samples inside a Lookup table, to ensure that the frequency of the output is 50Hz we set period_b =2000 , sampling_factr =500 and we used a clock of 50MHz for the clk shown in Figure III-28. Finally we stored the sets of output pins {s11, s12 , s21, s21} with the help of the table II-3. Figure III-31 represents the functional simulation results of the final design with (clk, period_b, sampling_factr, val_bits) = (50MHz, 2000, 500,2), we added an output state[1 ..0] to easily observe the states sequence as shown in Figure III-28, Figure III-31 shows that the design outputs gatting signals exactly resemble to the reference in Figure II-17.

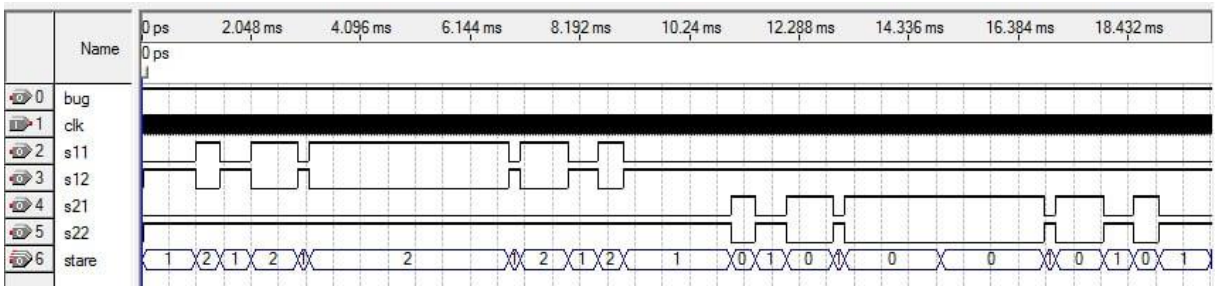

Figure III-31: Functional simulation results of SHEPWM circuit design

After editing, compile and run the simulation of the VHDL design, we assign the pins, we set the input clk to the internal 50MHz clock of the DE2 board and reset to GND, we set the outputs to the expansion Headers of the DE2 board, the final Pin assignment setting is shown in Figure III-32, then we compiled again then run the code using the programmer tool in Quartus II, the outputs gating signals s11, s12, s21 and s22 are presented in Figure III-33.

	To	Location	I/O Bank	I/O Standard	General Function	Special Function	Reserved	Enabled
1	clk	PIN_N2	2	3.3-V LVTTL	Dedicated Clock	CLK0, LVDSCLK0p, In...		Yes
2	s11	PIN_K25	5	3.3-V LVTTL	Row I/O	LVDS121p		Yes
3	s12	PIN_M22	5	3.3-V LVTTL	Row I/O	LVDS122p		Yes
4	s21	PIN_M19	5	3.3-V LVTTL	Row I/O	LVDS123p		Yes
5	s22	PIN_N20	5	3.3-V LVTTL	Row I/O	LVDS124p		Yes
6	bug	PIN_AE23	7	3.3-V LVTTL	Column I/O	LVDS151n		Yes

Figure III-32: Pin assignments of the SHEPWM final design

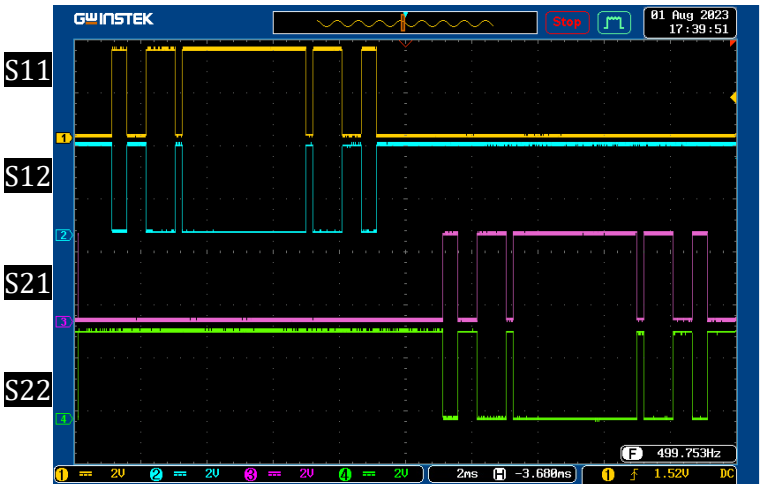

Figure III-33: SHEPWM gating signals generated by the FPGA-Based controller

Figure III-33 illustrates the gating signals generated by the FPGA, as observed in the above Figure the signals closely resemble the gating signals generated using Simulink shown in Figure II-17 , we can also observe that the frequency of the gating signal is 50Hz and that s11 is the logical inverse of s12 and they reflect the positive part of the reference signal and s21 is the logical inverse of s21 and they reflect the negative part of the signal and that's exactly as expected, therefore we connected these gating signals to the op4510 and execute the HIL simulator, their results are presented in the following:

Results: Figure III-34 represent the results of HIL simulation with SHEPWM controller, we can evaluate the THD value of the voltage output across the Load_1 is reduced to 48.42% it is very

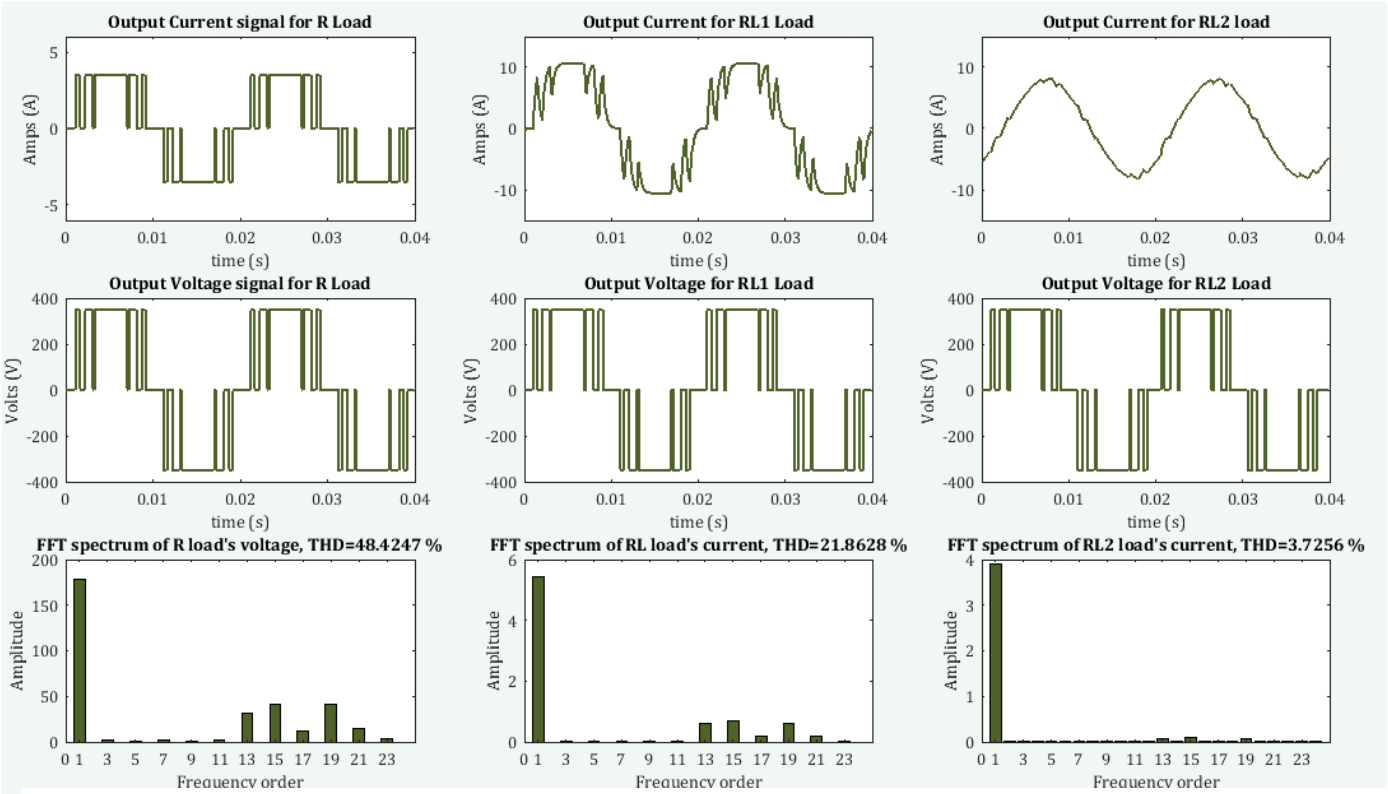

Figure III-34: Results of HIL simulation with SHEPWM controller

close to the reference 47.05%, The THD of Load_2's current is 21.86% while the reference was 21.57%, the THD of Load_3's current is 3.72% it is also close to the reference that was 3.735%. From the THD values above we can observe that it is successfully close to the reference values.

III.7. Results' Discussion:

After obtaining the HIL results we calculated the percentage of errors of the THD to see which design is better, using the following formula and presented the values in Table III-2:

$$Error\ \% = \frac{|Reference\ values - Experimental\ values|}{Reference\ values} * 100$$

Table III-2 demonstrates that the Bipolar PWM has a large error percent of 156.68% with Load_3 similarly the Unipolar PWM has significant large error percents of 410.76% through Load_2 and 637.82% through Load_3, these large errors can be due to multiple reasons like interference of external noise or Instrument malfunction or due to human error like mislabeling the gathered samples, Unfortunately, due to time constraints, we were unable to investigate all potential causes, Additionally the table shows that the SHEPWM controller's largest error was only 6.82%, a remarkably small value compared to the other error percentages presented. This indicates that this design closely approximated the reference, making it a suitable choice for implementation with the real IGBT H-bridge inverter as shown in Figure III-35.

Table III-4: Table present the error percent between Experiment and reference's THD

		Reference values	Experimental HIL results	Error percent %
Bipolar SPWM	THD of Voltage across the Load_1: R=100Ω	121.3512%	145.8016%	20.14
	THD of Current through Load_2: R=33Ω, L=1mH	23.7168%	27.6153%	16.43
	THD of Current through Load_3: R=33Ω, L=10mH	8.9003%	22.8454%	156.68
Unipolar SPWM	THD of Voltage across the Load_1: R=100Ω	63.286%	69.3599%	9.59
	THD of Current through Load_2: R=33Ω, L=1mH	13.5154%	69.0317%	410.07
	THD of Current through Load_3: R=33Ω, L=10mH	3.7239%	27.4758%	637.82
Selective-Harmonic-Elimination PWM	THD of Voltage across the Load_1: R=100Ω	47.0528%	48.4247%	2.91
	THD of Current through Load_2: R=33Ω, L=1mH	21.5791%	21.8628%	1.31
	THD of Current through Load_3: R=33Ω, L=10mH	3.4874%	3.7256%	6.82

The following Figure demonstrates the real H-bridge inverter that we previously mentioned, as well as the output signal from the real inverter in the oscilloscope:

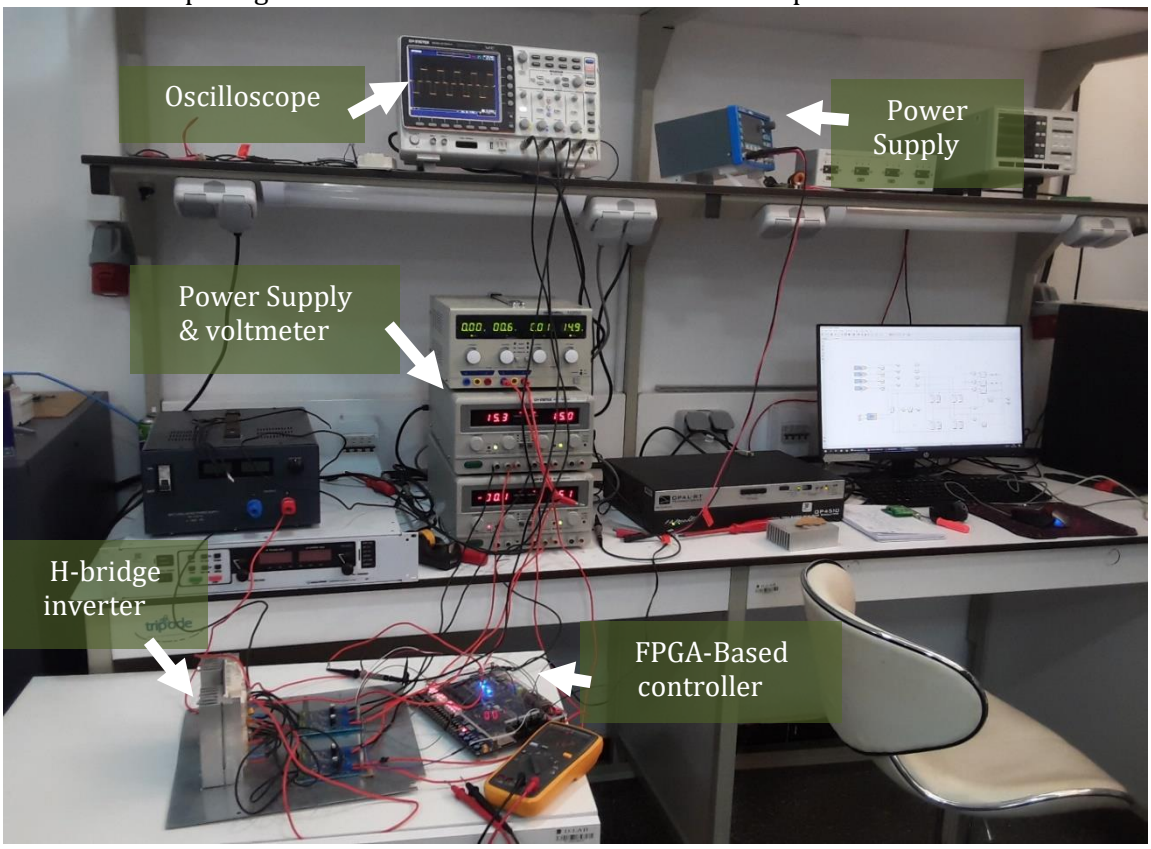

Figure III-35: Installation of the FPGA-Based controller in Real inverter

As shown in Figure III-35 and Figure III-36, the H-bridge inverter successfully generates the desired output waveform. The output signal closely matches the reference signal, indicating that the SHEPWM controller effectively drives the inverter.

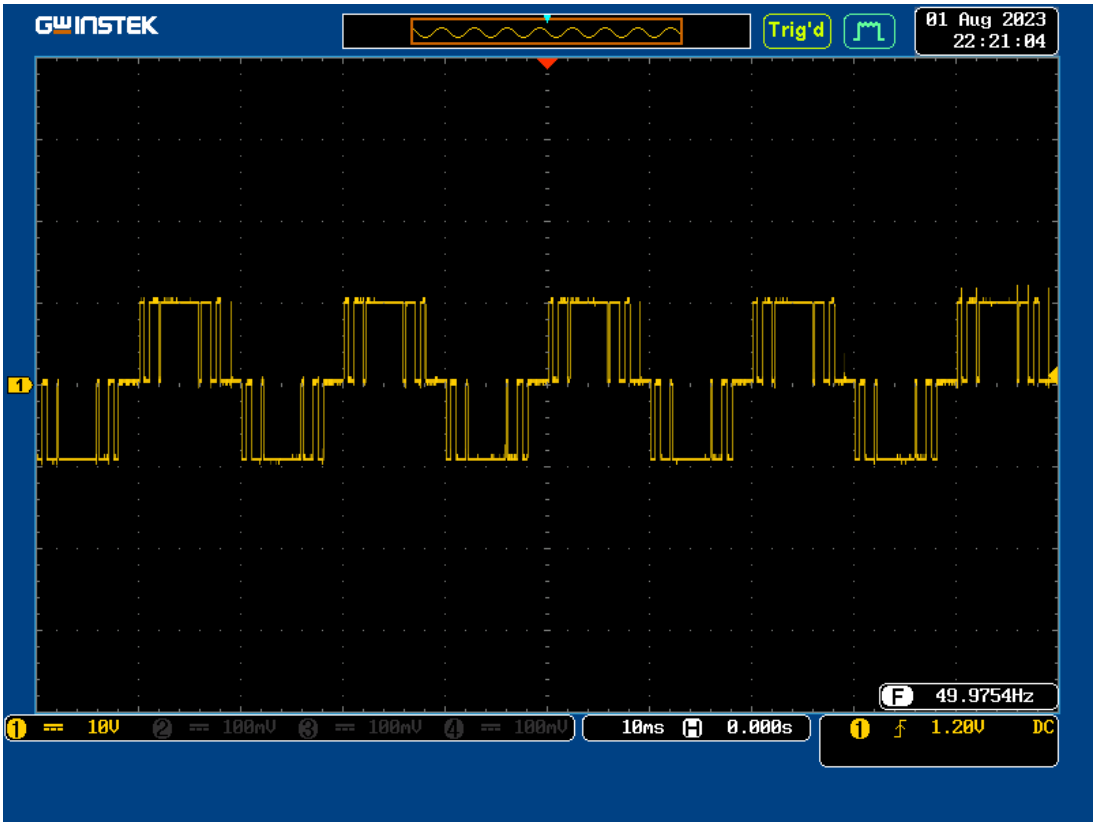

Figure III-36: The voltage output waveform of the real H-bridge inverter

III.8. Conclusion:

This chapter has dealt with the design of the three controllers (Bipolar PWM, Unipolar PWM, and SHEPWM) and presented them in detail, highlighting their unique characteristics and implementation process. The HIL simulation results demonstrated the effectiveness of each control strategy in regulating the inverter output.

Based on the obtained results, the SHEPWM controller emerged as the most promising design due to its performance in terms of error reduction and overall stability.

Overall, this chapter has provided valuable insights into the implementation of HIL simulation for different inverter control algorithms.

General Conclusion:

The objective of this project was the development and testing of an FPGA-based controller for a three-level power inverter and investigate the use of HIL simulation for testing and validating the inverter controllers. using the OPAL-RT HIL simulation tool. The results of the HIL simulation were the research successfully demonstrated the practicality of using HIL simulation to evaluate FPGA-based inverter controllers. Through extensive testing with various PWM control algorithms (Bipolar SPWM, Unipolar SPWM, and SHEPWM), it was found that the SHEPWM controller exhibited significantly lower error percentages compared to the other two algorithms. This suggests that SHEPWM is a promising control strategy for achieving high-quality inverter performance.

Limitations & Future works:

Limitations:

- Time Constraints: Due to limited time, a more in-depth analysis of the factors leading to the large error percentages observed with Bipolar PWM and Unipolar PWM was not possible.
- Hardware Limitations: The available hardware resources, such as the older Altera DE2 board, may have imposed limitations on the complexity of the controller.
- Simulation Model Accuracy: The accuracy of the simulation model used to represent the three-level inverter could potentially influence the obtained results.

Future Works:

- In-Depth Error Analysis: To gain a deeper understanding of the factors contributing to the large error percentages observed with Bipolar PWM and Unipolar PWM, future research could focus on conducting detailed error analysis.
- Comparison with Other Control Strategies: Exploring additional control algorithms and develop more complex control systems, such as Model Predictive Control, Phase-Locked-Loop,
- Real-World Implementation and Testing: To validate the findings of this study in a real-world environment, future research could focus on implementing the SHEPWM controller in Power-Hardware-In-the-Loop (PHL) simulation.
- Optimization of SHEPWM parameters: Investigating the impact of different SHEPWM parameters on inverter performance.

References

[1] E. C. Dos Santos and E. R. Cabral Da Silva, ADVANCED POWER ELECTRONICS CONVERTERS:PWM Converters Processing AC Voltages, Canada: IEEE Press, 2015.

[2] M. N. Bin Wu, HIGH-POWER CONVERTERS AND AC DRIVES, Hoboken, New Jersey and Canada: John Wiley & Sons, Inc, 2017.

[3] M. S. BENMERABET, "Contribution à l'intégration des énergies renouvelables dans les réseaux électriques en utilisant les convertisseurs multi niveaux," 29/12/2020.

[4] L. P. S. Anjali Krishna R, "A Brief Review on Multi Level InverterTopologies,," *International Conference on Circuit, Power and Computing Technologies,* pp. 1-6, 2016.

[5] A. C. A. A. S. M. Obbu Chandra Sekhar, "Simulation of Three-Level Diode Clamped Multilevel Inverter Using SPWM Technique, Space Vector Strategy & SHE Technique," *RECENT,* vol. 17, no. 2, pp. 107-115, 2016.

[6] K. C. J. M. K. D. Akanksha Sinha, "An inclusive review on different multi-level inverter topologies, their modulation and control strategies for a grid connected photo-voltaic system," *Elsevier Ltd,* p. 633–657, 1 June 2018.

[7] J. S. H. B. Jana, "A review of inverter topologies for single-phase grid-connected photovoltaic systems.," *Renew. Sustain. Energy Rev,* vol. 1, no. 71, p. 1256–1270, 2017.

[8] P. K. SACHIN MAHESHRI, "Simulation of single phase SPWM (Unipolar) inverter," *International Journal of Innovative Research in Advanced Engineering (IJIRAE),* vol. 1, no. 9, pp. 2349-2163, October 2014.

[9] Y. S. ,. R. M. Mohamed A.Ghalib, "Design and Implementation of a Pure Sine Wave Single Phase Inverter for Photovoltaic Applications," *Conf. Informatics, Electron,* pp. 1-8, 2014.

[10] Y. Z. ,. C. Han Zhou, "Hardware-in-the-loop simulation of inverter power supply controller based on StarSim and DSP," *Journal of Physics: Conference Series,* pp. 1-7, 2022.

[11] Y. Azzougui and A. Recioui, "Application of the moth flame optimisation to the selective harmonic elimination in multilevel converters," *International Journal of Smart Grid and Green Communications,* vol. 2, no. 1, pp. 1-18, 2020.

[12] H. S. PATEL and R. G. HOFT, "Generalized Techniques of Harmonic Elimination and Voltage Control in Thyristor Inverters: Part I--Harmonic Elimination," *IEEE Transactions on Industry Applications,* Vols. IA-9, no. 3, pp. 310-317, May/Jun., 1973.

[13] A. Kulkarni, "Implementation of SHEPWM in Single Phase Inverter," *International Journal of Scientific and Research Publications,* vol. 6, no. 1, pp. 431-435, 2016.

[14] A. Kulkarni, "Implementation of SHEPWM in Single Phase Inverter," *International Journal of Scientific and Research Publications,* vol. 6, no. 1, pp. 431-435, 2016.

[15] J. Belanger, P. Venne and J. N. Paquin, "The what, where and why of real-time simulation.," *Planet Rt,* vol. 1, no. 1, pp. 25-29, 2010/10/4.

[16] "wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Hardware-in-the-loop_simulation. [Accessed 9 2023].

[17] C. D. .. I. B. Menno Mennenga, "Fahrdynamik-Regelung: Modellbildung, Fahrerassistenzsysteme, Mechatronik," *Elektronik automotive,* pp. 45-50, 2009.

[18] T. Z. Jörg Schäuffele, Automotive Software Engineering:Grundlagen, Prozesse, Methoden und Werkzeuge effizient einsetzen, Springer-Verla, 2010.

[19] B. T. Cem Unsalan, Digital System Design with FPGA Implementation Using Verilog and VHL, United States: Mc-Graw-Hill Education, 2017.

[20] H. M. AnkitaGoel, "Embedded Systems Design Flow Using Altera's FPGA Development Board," (DE2-115 T-Pad), 2012.

[21] B. Yazid, "Study and Implementation of Real-Time Testing with Hardware in the Loop for Protection Relays," 2020.

[22] O.-R. Academy, "Fundamental Course OP-101 Real-Time Simulation Fundamentals with RT-LAB Software, Models & Hardware," fabruary 2019. [Online].

[23] O.-R. Technologies, "Simulator Platform Comparison Chart," 2020.

[24] Opal-rt Academy, "RT-LAB Documentation," 4 2023. [Online]. Available: https://opal-rt.atlassian.net/wiki/spaces/PRD/pages/144249433/Mezzanines. [Accessed 9 2023].

[25] Academy, OPAL RT, "OP4510 RT-LAB-RCP/HIL SYSTEMS User Guide," 2016. [Online]. Available: https://opal-rt.atlassian.net/wiki/spaces/PHDGD/pages/144689955/OP4510+V2. [Accessed 12 2022].

[26] G. K. V. G. A. Mohamed S. A. Dahidah, "A Review of Multilevel Selective Harmonic Elimination PWM: Formulations, Solving Algorithms, Implementation and Applications," *IEEE Transactions on Power Electronics,* 2013.

[27] A. Corporation, Quartus II Handbook, San Jose, 2014.

[28] O.-R. Academy, RT-LAB Quick Start Guide, 2018.

[29] H. So, "Introduction to Fixed Point Number Representation," 28 2 2006. [Online]. Available: https://inst.eecs.berkeley.edu/~cs61c/sp06/handout/fixedpt.html. [Accessed 9 2023].

[30] G. E. Eralp Sener, "Design of a Half-Bridge Current-Source Inverter Topology for Avionic Systems," *Aerospace,* p. 16, 2022.

[31] J. L. M. F. K. C. H. Sepahvand, ""Capacitor Voltage Regulation in Single-DC-Source Cascaded H-bridge Multilevel Converters UsingPhase-Shift Modulation," *IEEE Transactions on Industrial Electronics,* vol. 60, no. 9, pp. 3619-3626, 2013.

[33] T. A. H. A. M. T. I. Laith A. Mohammed, "Implementation of SHE-PWM technique for single-phase inverter based on Arduino," *International Journal of Electrical and Computer Engineering ,* vol. 11, no. 4, p. 2907~2915, August 2021.

[34] "mercury arc rectifier," [Online]. Available: https://www.oobject.com/category/mercury-arc-rectifiers/. [Accessed 9 2023].

[35] "Altera DE2 Board," [Online]. Available: https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=. [Accessed 9 2023].

[36] S. K. 2. ,. M. H. H. T. G. B. A. H. M. T. Ezzidin Hassan Elmabrouk Aboadla 1, "Selective Harmonics Elimination Technique in Single Phase Unipolar H-Bridge Inverter," *IEEE Student Conference on Research and Development,* 2016.

[37] EE Web, "Design Suite Accelerates System Development," [Online]. Available: https://www.eeweb.com/design-suite-accelerates-system-development/. [Accessed 2023].

[38] OPAL-RT, "OPAL-RT," 2023. [Online]. Available: https://www.opal-rt.com/software-rt-lab/. [Accessed 5 2023].

[39] H.-l. ZHENG , B.-m. GE and D.-q. BI, "RT-LAB based real-time simulation of photovoltaic power generation system," *ADVANCED TECHNOLOGY OF ELECTRICAL ENGINEERING AND ENERGY,* vol. 4, no. 29, pp. 62-66, 2010.

[40] Mathworks, "What is FPGA-In-the-Loop," Mathworks, [Online]. Available: https://www.mathworks.com/help/hdlverifier/ug/fpga-in-the-loop-fil-simulation.html. [Accessed 7 4 2024].