IMPLEMENTATION OF CONTROL FOR MMC-BASED ARBITRARY WAVE SHAPE GENERATOR IN THE OPAL-RT SIMULATOR

Sandhyaa Venkateswaran Subramaniam (5122058)

TU Delft

IMPLEMENTATION OF CONTROL FOR MMC-BASED ARBITRARY WAVE SHAPE GENERATOR IN THE OPAL-RT SIMULATOR

By

Sandhyaa Venkateswaran Subramaniam

in partial fulfilment of the requirements for the degree of

Master of Science
in Electrical Engineering

at the Delft University of Technology,
to be defended publicly on Friday August 6th, 2021 at 9:00 AM

Supervisor	: Dr.Ir. Thiago Batista Soeiro
Thesis committee	: Dr. Ir. Thiago Batista Soeiro Prof. Dr. Ir. Pavol Bauer Dr. Aleksandra Lekic
Daily Supervisor	: Ir. Dhanashree Ganeshpure : Ir. Yang Wu
MSc coordinator	: Dr.Ir. Babak Golizad

TU Delft

Table of Contents

List of Figures

List of Tables

ABSTRACT

A High Voltage (HV) Modular Multilevel Converter (MMC) based Arbitrary Wave shape Generator (AWG) is to be designed for a final voltage of 100 kV with 67 submodules in each arm. They can be used to test unconventional stresses experienced by high voltage equipment. Before the realization of a full-scale prototype setup, it is necessary to validate the controller as it involves the complexity of communication and control of a large number of switches in the circuit while demanding accuracy in the generation of gate pulses. OPAL-RT was identified as a good solution for implementing the MMC control.

This thesis intends to investigate two main objectives. Firstly, to evaluate the various capabilities of the OPAL-RT to implement control of MMC by performing simulations and testing it with real MMC with 12- submodules. Secondly, to design a PR controller for the MMC circuit and further validate it in real-time. The topology of MMC, as well as two modulation approaches – Phase-Shifted Carrier (PSC) and Nearest Level Control (NLC) – were explored initially. This was followed by the filter design for 50 Hz and 1 kHz fundamental frequencies. Thereafter, the trade-offs for both control strategies based on the filter design requirement were examined. Furthermore, the various software tools included in OPAL-RT were investigated for potential use in the development of the control algorithm.

To test the accuracy of gate pulses and the ability to generate precise waveforms for higher fundamental frequencies, Model-in-Loop testing with eFPGAsim is used. By realizing, the control algorithm in the CPU and the MMC in the FPGA, results obtained are compared with the offline simulations. It is observed that the minimum CPU time step is incapable of generating accurate waveforms for higher fundamental frequencies, resulting in missing or delayed gate pulses.

Next, the control was implemented in the FPGA using internal PWM generators of the OPAL-RT and the MMC plant was tested both in the FPGA and as actual hardware. In PSC modulation, the phase shift between the upper and lower arm decides the switching pattern of the submodules, contributing to the building of the voltage level at any instant. In this case, the gate pulses' phase difference between the lower and upper arms are noticed to be larger than what is observed from the offline simulations. These phase delays are resulting in the addition of delayed gate pulses at any instant causing strange switching voltage levels at the output, contributing to unexpected carrier harmonics. Furthermore, because the CPU samples the modulating signal before passing it to the FPGA for processing, it compensates for communication latency while also contributing to inaccuracy.

Finally, a PR controller was designed and discretized for performing stability analysis, later implemented on the real-time setup. The PR controller resulted in suppressing the lower order harmonics by reducing them within the expected tolerance limits and the THD obtained was consistent with the results of the offline simulation.

ACKNOWLEDGEMENT

The goal of this master's thesis is to evaluate the OPAL-RT Real-time simulator's capacity to control an MMC-based Arbitrary Wave Shape Generator. I'd like to use this opportunity to express my gratitude to everyone who has helped me complete this project on time.

I am grateful to **Dr. Ir. Thiago Batista Soeiro** for believing in me and encouraging me throughout the process. His optimism has given me the confidence to pursue this as a master's thesis as well. I am also thankful to the other thesis committee members **Prof. Dr. Ir. Pavol Bauer** and **Dr. Alexandra Lekic**. I am grateful to **Dr. Ir. Babak Gholizad** for giving me all the academic support I required. I owe **Ir. Dhanashree Ganeshpure** a huge debt of gratitude for her unwavering support and guidance throughout the project by sharing her ideas and suggestions. **Ir. Yang Wu** has been instrumental in guiding me in the project and helping me realize my potential.
I also owe a debt of gratitude to **Dr.Ir. Jianning Dong** for his prompt assistance in gaining a better insight into the settings and simulators. I would also like to express my gratitude to the **OPAL-RT teams in Europe and Canada**, who have provided quick assistance and suggestions based on their software experience in resolving software-related challenges.

I am grateful to my husband, who has always believed in my potential and encouraged me to pursue a master's degree. Finally, I am grateful to my family and friends for supporting me during this journey, especially during difficult times.

Sandhyaa Venkateswaran Subramaniam
June 2021

List of Abbreviations

Abbreviation	Definition
AWG	Arbitrary Waveshape Generation
DOV	Default Output Value
EPROM	Erasable Programmable Read-only Memory
FFT	Fast Fourier Transform
FIFO	First In First Out
FPGA	Field-Programmable Gate Arrays
GM	Gain Margin
GUI	Graphical User Interface
HDL	Hardware Description Language
HIL	Hardware -In-Loop
HV	High Voltage
KVL	Kirchoff's Voltage Law
LCA	Loss Compensation Algorithm
LUT	Look-Up Table
MIL	Model-In-Loop
MMC	Modular Multilevel Convertors
MSB	Most Significant Bit
MUSE	MUlti-System Expansion
NLC	Nearest Level Control
PI	Proportional-Integrator
PLD	Programmable Logic Devices
PM	Phase Margin
PR	Proportional-Resonant
PSC	Phase- Shifted Carrier
PWMO	Pulse Width Modulated Out
RCP	Rapid Control Prototyping
RES	Renewable Energy Sources
SDO	Static Digital Out
SFP	Small Form-factor Pluggable
SRAM	Static Random-Access Memory
SUT	System Under Test
THD	Total Harmonic Distortion
VHDL	Very-high-speed-integrated circuit Hardware Description Language
ZOH	Zero-Order Hold

1. Introduction

The electrical power network is facing new electrical stresses due to the advent of power electronics and novel methods of Renewable Energy Sources (RES) [1]. These stresses cause effects on high voltage equipment such as transformers, cables, and switch gears. Existing HV testing sources can imitate waveforms such as sinusoidal, lightning, switching impulse, or a DC to understand the effect of the stresses but has their limitations on the signal bandwidth and to be used to test higher voltage levels. Recent studies in power electronics have speculated that multilevel converter topologies are an ideal solution, which can overcome the drawbacks of other traditional HV test sources allowing it to test for higher voltage with large bandwidth. They are still being investigated for their flexibility and accuracy to generate arbitrary waveshapes.

The feasibility of a multilevel converter topology like the Modular Multilevel Converter (MMC) as an Arbitrary Waveshape Generator (AWG) has been mathematically modelled and designed in [2]. This was designed using an offline simulation method that convinces the mathematical computations and control logic of the circuit. In the future, the implemented topology must be built as a single-phase full-scale prototype test source. This prototype shall be created for a final voltage of 100 kV with 67 submodules in each arm. But, it is necessary to first perform some preliminary studies on a smaller number of submodules like chosen in the offline design. This can aid in comprehending the required accuracy of control hardware and its capability to generate output waveforms at higher frequencies. Hence, real-time simulation is a better solution to perform the preliminary studies that will help in studying the control like the way it happens in a real physical system.

The OPAL-RT simulator is a target capable of performing real-time simulations and was chosen as a wise solution for the implementation of control of the MMC. However, the accuracy of the control algorithms and the communication with the real hardware by the OPAL-RT should be verified. This thesis intends to investigate the capabilities of the OPAL-RT to implement the control of MMC by understanding the target as hardware. As the main focus of the thesis is on control hardware, only a sinusoidal wave generated from the MMC- based AWG is presented as an example. For this purpose, multiple software tools in OPAL-RT is used for the preliminary studies.

1.1. Research Objective

As illustrated in Figure 1.1, the thesis is separated into two preliminary study phases. The first objective of the thesis is to investigate the capabilities of the OPAL-RT for the implementation of the control algorithms of MMC. To begin with, an experiment of the open-loop control of MMC using the eFPGAsim software was performed by implementing the MMC in the FPGA of the OPAL-RT and the control algorithm in the target's CPU. This stage also comprises developing the control algorithm using the RT-Events block-set available in OPAL-RT and the analysis of the control algorithm in the FPGA of OPAL-RT Target. The above results are then compared to the data from the offline simulation. The second objective corresponds to studying closed-loop strategies for the HV MMC setup and designing a PR controller for the MMC. The main focus for designing the PR controller is to ensure the stability of the system while achieving zero steady-state error and restricting the lower order harmonic within tolerance limits. This design is further implemented and verified in the Model-In-Loop setup. This thesis intends to achieve the following goals mentioned below:

1. To understand the topology of MMC, control modulation techniques selected for the HV arbitrary waveshape generator application
    - To design a filter to eliminate undesirable harmonics, and examine the trade-offs of the two control techniques (PSC and NLC) based on the filter design requirement.

2. To study and get hands-on with OPAL-RT simulator and its associated software to test the control of MMC in real-time
    - RT-LAB
    - RT-EVENTS
    - eFPGAsim
    - RT-XSG

3. To perform Model-in-Loop (MIL) testing with open-loop control for the MMC
    - Evaluate the accuracy of the gate pulses with control algorithm in CPU and FPGA
    - The capability of the CPU of OPAL-RT target to generate waves with higher fundamental frequencies

4. To design the closed-loop control scheme for the HV MMC Arbitrary waveshape generator with a PR controller and to implement and further validate it with zero steady-state error in the Model-in-Loop setup.

Preliminary tests were carried out on a 12-submodule MMC model with a 300V DC link voltage. Because of various constraints provided by the available target and software, which only permits 48 switches to be modelled in the FPGA, a 12-submodule MMC was chosen for the investigation. The investigations were first performed as a Model-in-Loop test for implementation of the control inside the real-time simulator. Later, some of the Model-in-Loop analysis was applied to an existing Rapid Control Prototyping (RCP) setup with 12 submodules to understand the control performance in the real hardware.

1.2. Thesis Outline

Chapter 2 explains the basic analysis of the circuit and modulation of the MMC is covered in this chapter. The effects of two modulation approaches, Phase Shifted Carrier (PSC) and Nearest Level Control (NLC), are studied in depth. For two fundamental frequencies of 50 Hz and 1 kHz, the filter design for both modulation schemes is also presented. To choose one modulation approach for implementation, a trade-off analysis is undertaken.

Chapter 3 provide some background information on several types of simulations and real-time testing methodologies. A brief overview of the OPAL-RT simulator is also provided. The OPAL-RT software packages, notably eFPGAsim, which was widely used in this Thesis study, are described in-depth, followed by other packages on which hands-on experience was gained. In addition, knowing the architecture of the FPGA is emphasised to become familiar with the RT-XSG software, which requires a basic knowledge of the FPGA.

Chapter 4 portrays the results of the Model-in-Loop testing using both control modulation techniques (PSC & NLC) with control implemented in the CPU and the plant simulated in the FPGA for two different fundamental frequencies. The results of the same experiments with the implementation of RT-EVENTS

blocks for PSC modulation are also documented in this section. Interpretations on the accuracy of the control due to the implementation of control in the CPU is concluded.

Chapter 5 outlines the analysis performed based on the implementation of control in FPGA based environment. It presents two methodologies of experiments performed namely the Model-in-Loop test and testing the model in an RCP setup. The experiment was first conducted in the smaller 4-Submodule MMC model due to the limited availability of FPGA based PWM generators in MIL. Further, the 12-Submodule MMC model was simulated in the RCP prototype to interpret the variations and to draw conclusions.

Chapter 6 portrays the design of a PR controller for the MMC in the continuous domain and further the methods used to implement it in the discrete-time domain. It shows the stability studies performed using bode and obtaining the right controller parameter values. Further, the controller was implemented in the real-time simulator and the results are compared to the Offline data.

Chapter 7 concludes the thesis findings by summarising their outcomes and provides recommendations for future work.

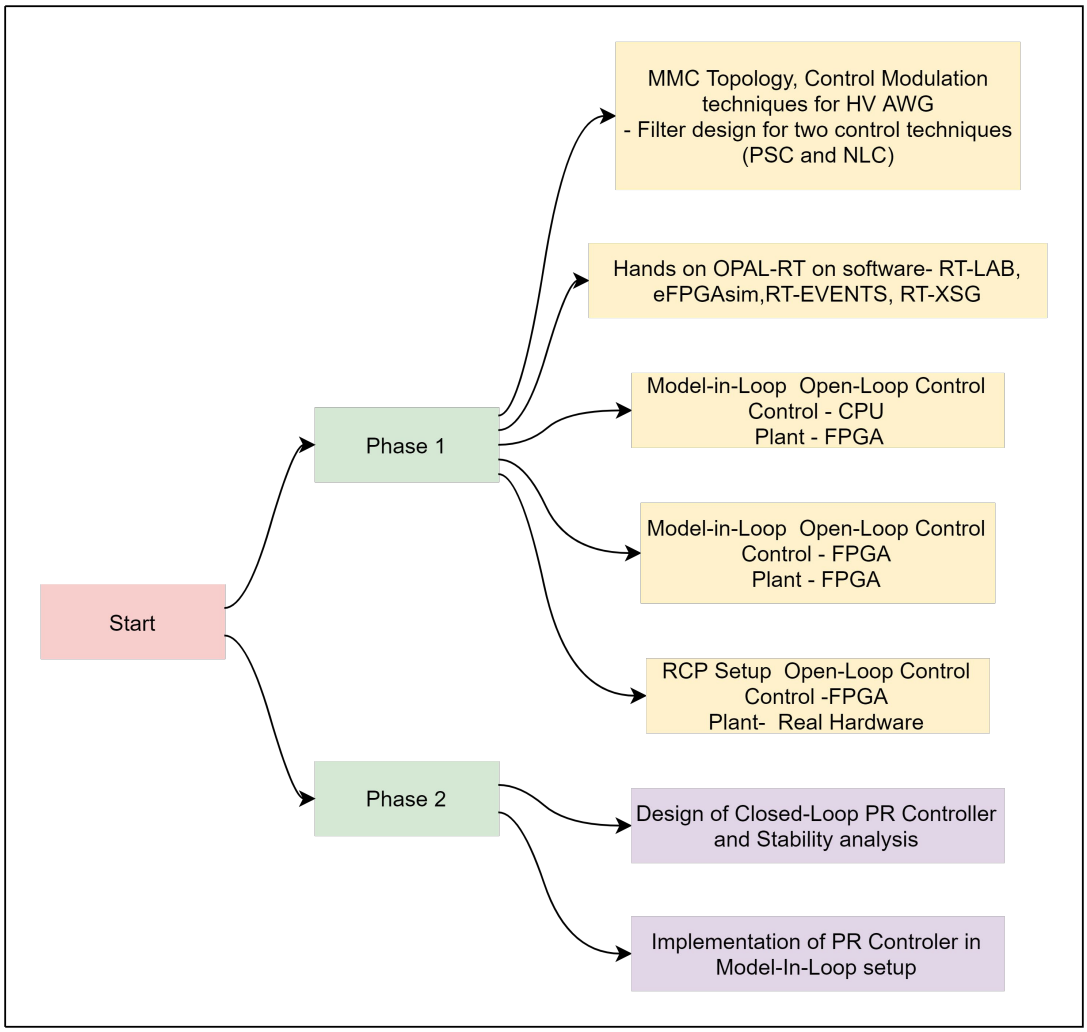


Figure 1.1: Structure of Thesis

2. Literature Study of MMC

2.1. Structure of MMC

A single leg MMC is made up of two arms in parallel, referred to as the upper and lower arms, and each arm is made up of the same number of submodules (N) as shown in Figure 2.1. Each MMC submodule constitutes two semiconductor devices (SW1 and SW2) such as the Insulated-Gate Bipolar Transistor (IGBT) with anti-parallel diodes. A DC capacitor is installed across these two switches to store the charges. During the normal operation, the two semiconductor devices in the lower and upper arms are either inserted or bypassed [2], [3]. When SW1 is Gated On and SW2 is Off, the submodule is inserted, and the voltage across the capacitor C_{SM} is equal to the submodule input voltage V_{AB}, as shown in Figure 2.2(a). When SW1 is Gated Off and SW2 is On, the submodule is bypassed and the voltage across the capacitor C_{SM} is zero, as shown in Figure 2.2(b). Figure 2.2(c) depicts a blocked situation in which both SW1 and SW2 have been turned off. This is used to safeguard semiconductors in the event of a fault. In the absence of a DC circuit breaker, this fault condition is utilized to protect the submodule and MMC against DC fault circumstances [2]. The difference in the number of submodules inserted in the higher and lower submodules, therefore, provides the output voltage[4].

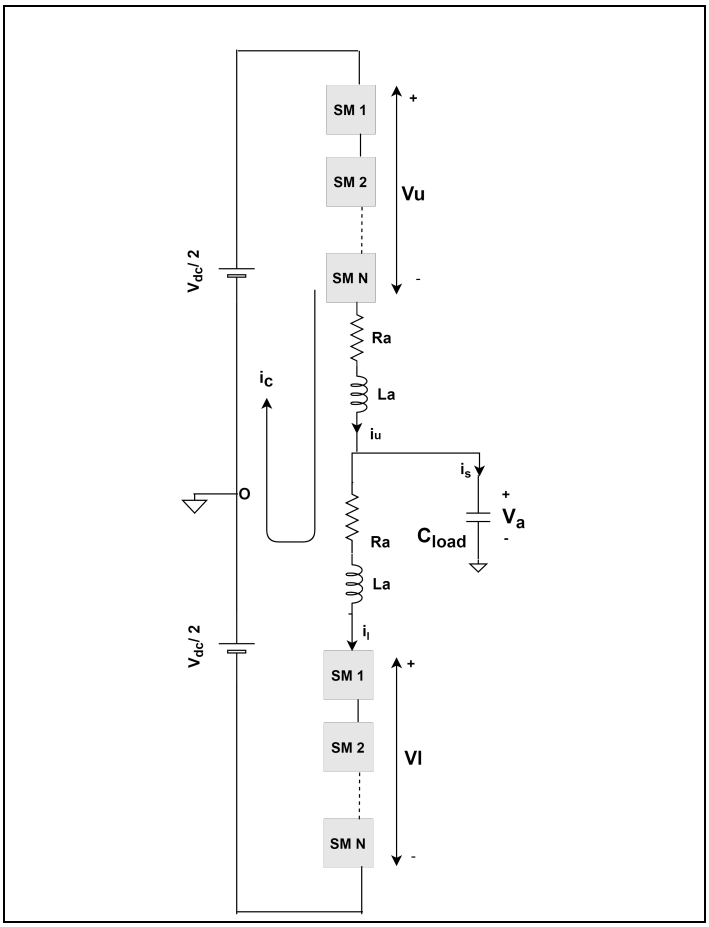


Figure 2.1: Topology of MMC

In an ideal operation of MMC, the capacitors across each of the submodules have an average capacitor voltage of $C_{SM} = V_{dc}/N$, and the DC link is purely capacitive with a value of V_{dc}. In addition to the submodules, the two arms each have an arm inductor (L_a) and an arm resistor (R_a) in series with the submodules to minimize

the high-frequency harmonics components of the arm currents, lowering power converter losses [5]. The difference between the upper and lower arm currents represents the output current of the circuit.

The constant charging and discharging of the capacitors in every submodule due to the arm current may deviate the capacitor voltage from its average value. This deviation between the DC voltage of the source and the total capacitor voltage of both arms results in circulating currents in the MMC [2]. To keep the submodule capacitor values close to their average values, arm controllers or sorting algorithms might be used. This single-phase MMC test setup consists of a capacitive load since most of the high voltage test objects are capacitive in nature.

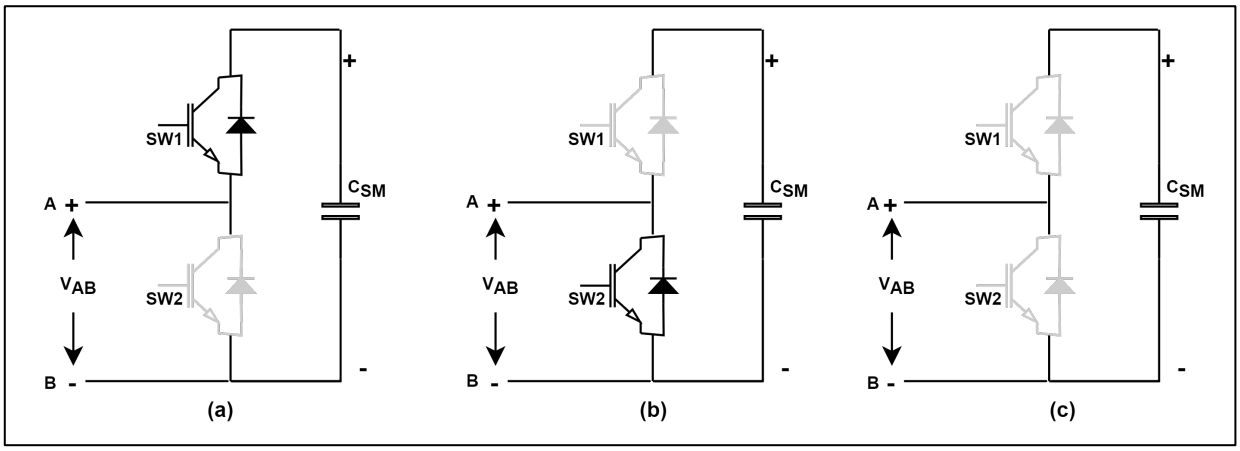


Figure 2.2: States of MMC (a) Inserted (b) Bypassed (c) Blocked

To understand the operation of the MMC, the equations of the arms are derived by applying Kirchhoff's voltage laws (KVL) to the circuit in Figure 2.1.

$$\frac{V_{dc}}{2} - v_u - L_a.\frac{di_u}{dt} - R_a i_u - V_a = 0 \tag{2.1}$$

$$\frac{V_{dc}}{2} - v_l - R_a i_l - L_a.\frac{di_l}{dt} + V_a = 0 \tag{2.2}$$

Thus, subtracting and adding to obtain the equations for switching voltage and circulating currents

$$v_l - v_u + L_a.\left(\frac{di_l}{dt} - \frac{di_u}{dt}\right) + R_a.(i_l - i_u) - 2V_a = 0 \tag{2.3}$$

$$V_{dc} - (v_u + v_l) - La.\left(\frac{di_u}{dt} + \frac{di_l}{dt}\right) - R_a.(i_u + i_l) = 0 \tag{2.4}$$

The equations (2.3) and (2.4) can be simplified further by substituting the following variables as mentioned in equation (2.5) to (2.8),

$$i_s = i_u - i_l \tag{2.5}$$

$$i_c = \frac{i_u + i_l}{2} \tag{2.6}$$

$$v_s = \frac{v_l - v_u}{2} \tag{2.7}$$

$$v_c = \frac{v_u + v_l}{2} \tag{2.8}$$

Here in equation (2.5), i_S represents the output current of the circuit. In equation (2.7), v_s corresponds to the switching voltage of the MMC. Also, in equation (2.6), i_C is the circulating current presented in the MMC circuit which is induced by the circulating voltage v_C as mentioned in (2.8). Substitute equation from (2.5) - (2.8) into equations (2.3) - (2.4), the MMC circuit models can be simplified as given in equations (2.9) - (2.10) illustrated in Figure 2.3(a) and Figure 2.3 (b).

The switching voltages due to the arm currents are obtained by deriving voltage through the charging and discharging of the submodule capacitances. This switching voltage is also responsible to provide the voltage across the load capacitance after passing through the filters of the circuit.

$$v_s - \frac{L_a}{2}\left(\frac{di_s}{dt}\right) - \frac{R_a}{2} i_s - V_a = 0 \tag{2.9}$$

$$V_{dc} - 2v_c - 2R_a i_c - 2L_a . \frac{di_c}{dt} = 0 \tag{2.10}$$

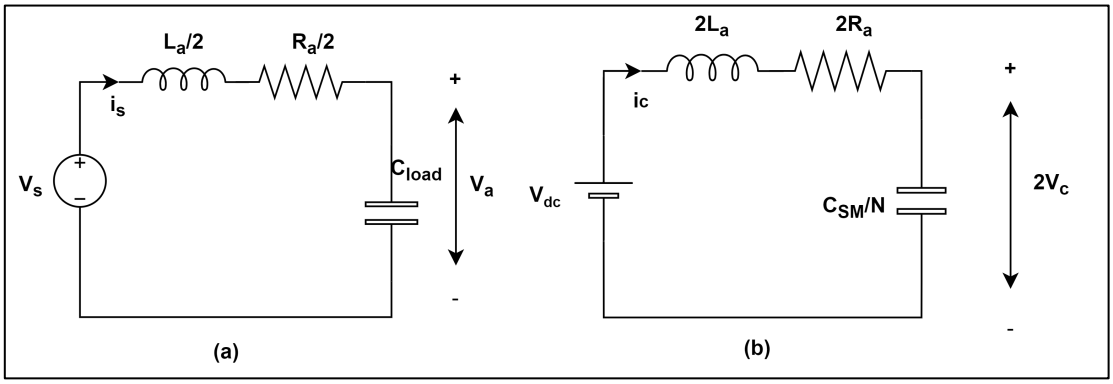


Figure 2.3: Simplified MMC Circuit (a) Output Current Circuit (b) Circulating current circuit

2.2. Modulation Techniques Used for the High Voltage AWG Application

The gate pules for the submodules can be generated using a variety of modulating approaches, which can be classified into high frequency based carrier-based modulation or fundamental frequency based staircase modulation. To control the MMC's gate pulses, the Nearest Level Control (NLC), also known as staircase modulation, and Phase-Shifted Carrier (PSC) modulation techniques, were used. It is noteworthy that only N submodules are inserted at any given time, out of the 2N submodules available from both arms that generate the necessary output voltage.

2.2.1. Phase-Shifted Carrier – PWM

The Phase-Shifted Carrier Pulse-Width Modulation (PSC-PWM) is a high-frequency modulation technique that consists of N carriers corresponding to (N) number of submodules in each arm. Each submodule's carrier is phase-shifted by a factor of $2\pi/N$. Additional phase shifts between the arms carriers are also possible, which aids in pushing the carrier harmonics to higher frequencies. For each arm, these carriers are compared with the one identical reference wave to generate the gate signals. If the reference wave is greater than the carrier signal at any instant the PWM considers a high pulse and if the reference signal is lesser than the carrier signal it generates a low pulse. The same process is performed for every carrier wave which delivers the desired gating pulse for the semiconductor devices present in the submodule respectively.

The PSC-PWM generates either 2N+1 or N+1 levels depending on the phase difference between the lower and the upper arm carrier signals [6]. This modulation technique provides 2N+1 levels, if the carrier signals are the same in the lower and upper arm when N is odd or the carrier signals are phase shifted by π/N for the lower and upper arm when N is even. Similarly, PSC-PWM provides N+1 level if the carrier signals are phase shifted by π/N in both arms when N is odd and having the same carrier signals on both the arms when N is even [4]. Such variations in the system are proven to provide better harmonic eliminations since the carrier harmonics are shifted to 2N times the carrier frequency or N times the carrier frequency depending on the 2N+1 or N+1 technique implemented, respectively.

The below Figure 2.4(a) is an example represented in [7], a PSC-PWM with carriers of the upper arm with 6-Submodules. Hence each Carrier is phase shifted by an angle of $2\pi/6 = 60^{o}$. Figure 2.4(b) indicates the number of submodules to be inserted in the upper arm at any one time.

2.2.2. Nearest Level Control (NLC) Modulation

The Nearest Level Control is also known as the staircase modulation, which is categorized into the low-frequency modulation technique. In this technique, submodules remain switched ON or OFF to generate the output voltage which is determined at every sampling time instant of the reference wave [8], [9]. To understand the modulation technique better as discussed in [7], we assume an MMC with 6-Submodules in each arm and hence it displays 7 levels or bands from (Level 0 to Level 6) as shown in Figure 2.5(a). A sinusoidal reference signal is also present for the modulation technique. These levels or bands indicate the number of submodules that should be inserted in the upper or the lower arm. The total number of submodules to be inserted in each arm to reach the output reference voltage at a sampling instant is provided by the rounding function given below in Equations (2.11) and (2.12).

$$N_u = \text{round}\left(\frac{N(V_{dc} - V_{ref})}{V_{dc}}\right) \tag{2.11}$$

$$N_l = \text{round}\left(\frac{N(V_{dc} + V_{ref})}{V_{dc}}\right) \tag{2.12}$$

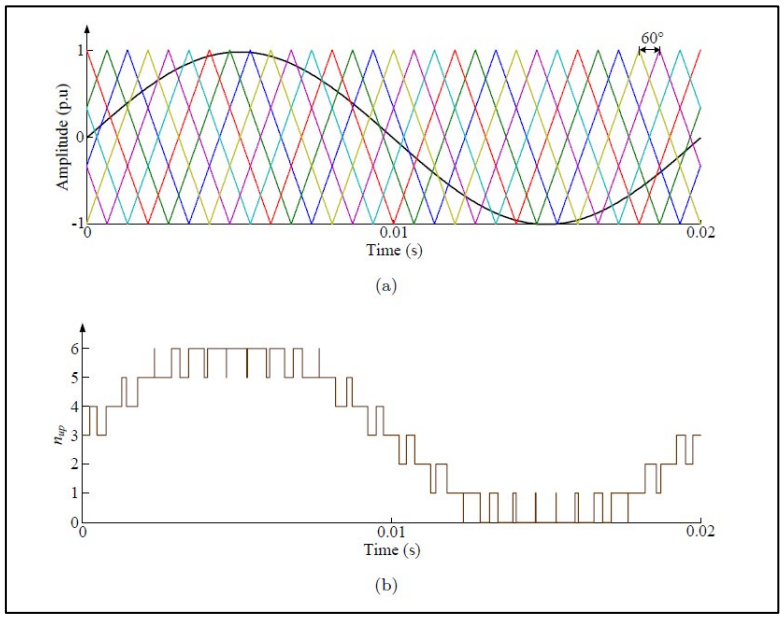


Figure 2.4: Phase-Shifted Carrier Modulation[7] (a) Phase-shifted carriers for N= 6 Submodules (b) Number of Submodules to be inserted at any instant (Upper arm)

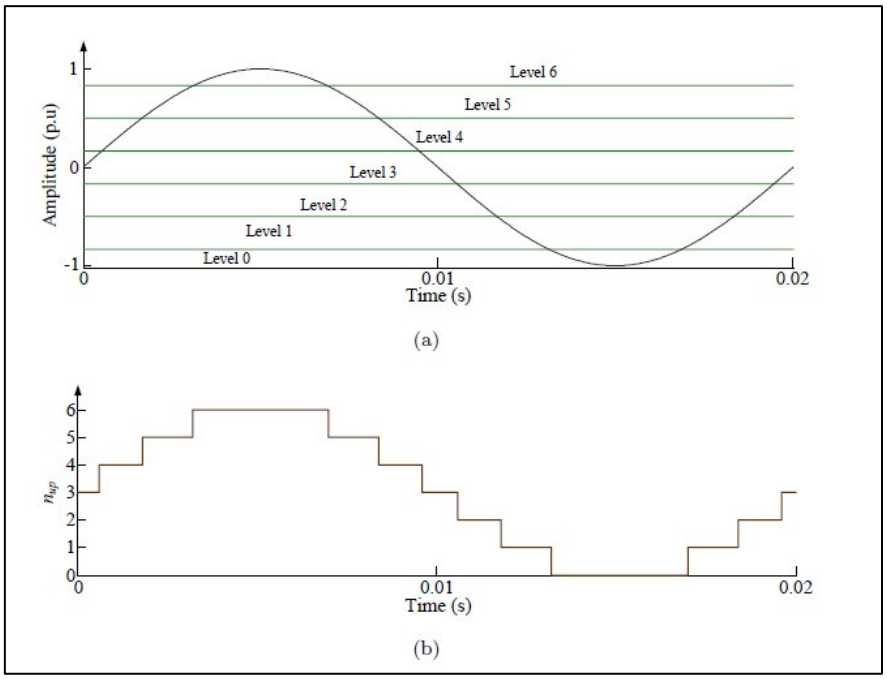


Figure 2.5: Nearest Level Control Modulation for N = 7 submodules[7] (a) Generation of bands and reference (b) Number of Submodules to be inserted at any instant (Upper arm)

At every instant when the sinusoidal reference signal crosses, one of these levels, the number of submodules to be inserted in each arm also changes as seen in Figure 2.5(b). The output voltage resulted from NLC does not have an exact waveshape as that of the reference signal due to the applied rounding off function at each sample instant [8]. As this technique revolves around the low switching frequency, a longer time of insertion or bypass of the submodules occur. This reduction in the switching frequency has a bad influence on the capacitor voltage balancing effect especially in circuits with a lower number of submodules. There are multiple sorting algorithms available to ensure this capacitor voltage balancing efficiently.

One of the common sorting algorithms is the bubble sorting algorithm which uses the arm currents and the dc capacitor voltage at every sample period [10]. The number of submodules to be inserted is calculated using the rounding equation (2.11) and (2.12). Depending on the current direction, the capacitor voltages are sorted ascending or descending [9]. The capacitor voltages are sorted in ascending order to charge the submodules with the lowest voltage if the arm currents are positive. If the arm currents are negative, the capacitor voltages are ordered in order of decreasing voltage to discharge the higher voltage submodules. This traditional sorting method creates a balanced influence on the voltages. However, sometimes few submodules are discharged to a voltage level lower than the average value and are used more than a submodule with a higher charge. As a result, the switching action of the submodules at a particular sampling instant contributes to higher switching losses. This drawback can be overcome by the tolerance band algorithm, introducing an addition of a tolerance value to the average capacitance. In the second case, the submodule will initially compare the submodule capacitor value with the average capacitor voltage plus the tolerance value [11]. The submodule is bypassed only if the capacitor voltage exceeds this tolerance band, thus reducing undesirable switching losses.

For the future simulations based on NLC in this thesis, a sorting algorithm pattern as stated in [2] is utilized, which may be applied regardless of the initial value of capacitor voltages or the direction of arm currents. Instead, the sequence of the submodules to be inserted will be varied every cycle. If, for example, the Nth submodule was utilized the least in the first cycle, it will be used first in the second cycle. In the same way, the Nth submodule present during the second cycle will be used at first in the third cycle. This will ensure that the average capacitor voltage is maintained constant and rapid switching actions can also be reduced. This expression is represented in a matrix form in the equation (2.13)

$$\mathrm{P} = \begin{bmatrix} UA_1 & UA_2 & UA_3 & \cdots & UA_N \\ UA_N & UA_1 & UA_2 & \cdots & UA_{N-1} \\ UA_{N-1} & UA_N & UA_1 & \cdots & UA_{N-2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ UA_2 & UA_3 & UA_4 & \cdots & UA_1 \end{bmatrix} \tag{2.13}$$

2.3. Filter Design

The MMC circuit addressed in this thesis is mostly composed of 12-submodules, and while this is a small-scale prototype, the low number of submodules may result in higher switching harmonics in the output voltage. Because the output voltage waveform impacts the amount of electric stress to be delivered to the high voltage equipment, as stated in [1], the use of an HV arbitrary wave shape generator necessitates careful filter design. Larger tensions in the circuit could put the testing instrument under unnecessary strain. As a result, the arm inductance and resistance must be designed in such a way that switching harmonics are minimized. For the application of HV arbitrary wave shape generation, a capacitive load is chosen as a representation of high

voltage equipment. The MMC model for providing an output voltage waveform is simplified and shown in Figure 2.3(a). The v_S represents the total switching voltage obtained due to the insertion and bypassing of the submodule capacitor, behaving as a staircase type output waveform. The arm current i_S , further passing through the filter results in the desired output voltage across the load. The circuit model clearly shows that the upper and lower arms are in parallel to each other and hence the simplified circuit shows that the arm inductance and arm resistance are halved. The presence of a capacitive load in the MMC and an arm inductance forming an LC circuit result in oscillations at resonant frequencies. However, the arm resistor designed is used for damping the oscillations occurring due to the inductance and the capacitive load. Thus this behaves like an RLC filter circuit in suppressing the switching harmonics of the output voltage waveform.

The filter in this case is designed as presented in [1] where the transfer function of the output current circuit is considered in the Laplace domain as in equation (2.14). This equation results in two equations (2.15) and (2.16) in terms of the arm inductance and arm resistance. The equations are evaluated keeping in mind that the values of L_a and R_a be practical values for the MMC circuit. The frequency spectrum of the switching voltage v_S is analysed while designing the filter to choose the n_1 and n_2 values practically. Note that the n_{1freq} represents the fundamental and the baseband frequencies in which, n_1 stands for the number of baseband harmonics one wants to keep with a 1% error. The n_{2freq} in equation (2.16) stands for the suppression of the undesirable higher harmonics where n_2 represents, which carrier harmonics has to be suppressed by a factor of k.

$$\frac{V_a[s]}{V_s[s]} = \frac{2}{s^2 L_a C_{load} + s R_a C_{load} + 2} \tag{2.14}$$

$$\text{abs}\left(\frac{V_a[s]}{V_s[s]}\right)_{@n_{1freq}} = 0.99 \tag{2.15}$$

$$\text{abs}\left(\frac{V_a[s]}{V_s[s]}\right)_{@n_{2freq}} = k \tag{2.16}$$

In this thesis, the analysed 12-submodule circuit has a capacitive load of 6.8 µF considered as equivalent HV equipment. The FFT is obtained as shown in Figure 2.6(a) for a fundamental frequency of 50 Hz with PSC modulation for 2N+1 levels. Using equation (2.15), n_1 is set to 4, implying that up to four baseband harmonics will be suppressed with a 1 percent error. Similarly, using the equations (2.16) and (2.17), n_2 is determined to be 470 and suppressed by a factor of 0.01. As already known, with 2N+1 levels, the switching harmonics are shifted to 2N times the switching frequency (F_{sw}) and the switching frequency for PSC modulation is chosen to be a non-integer multiple of the fundamental frequency to ensure capacitor voltage balancing [4][1].

$$\text{n}_2 = \frac{2N.\ F_{sw(undesirable)}}{\text{fundamental frequency}(freq)} \tag{2.17}$$

Thus, L_a is calculated as 1.32 mH and R_a is calculated to be 45 ohms. Similarly, for a fundamental frequency of 1 kHz with PSC modulation for 2N+1 levels the FFT was obtained as shown in Figure 2.6(b) and n_1 is

chosen as 3 considering until third baseband harmonics and n_2 as 221 for higher harmonics suppression. The arm inductance and arm resistance for 1 kHz is calculated as 15.2 µH and 3.71 Ω. The simulations performed hereon for the above-mentioned fundamental frequencies will hold the calculated values above for their arm inductance and resistance.

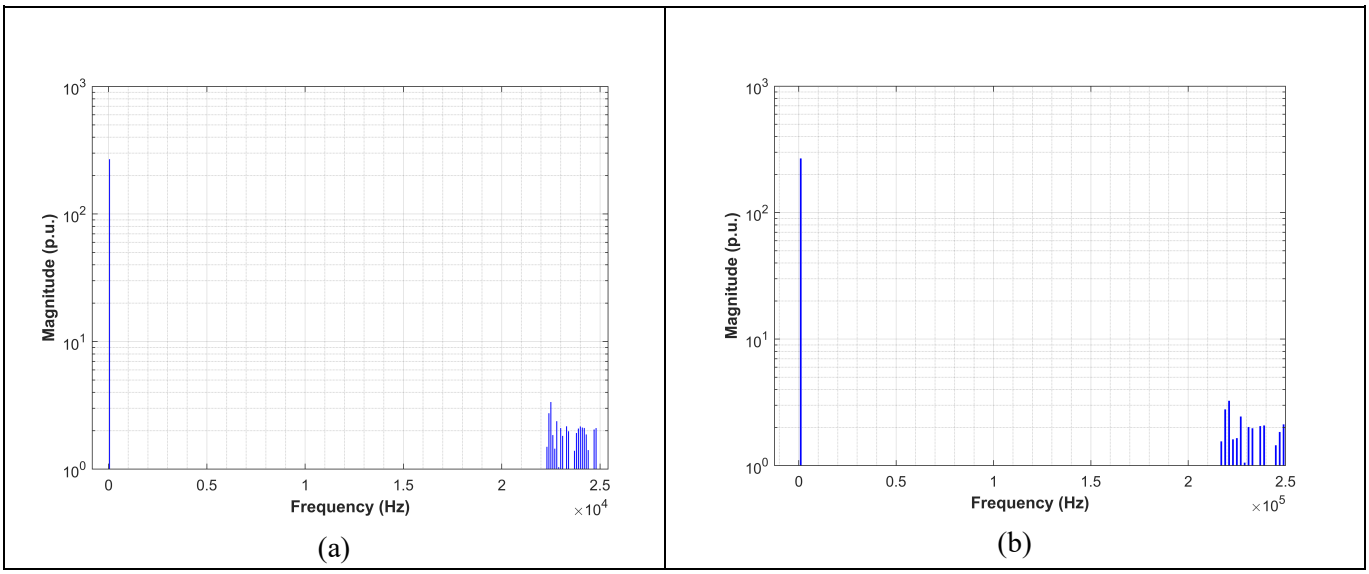


Figure 2.6: Frequency Spectrum of the switching voltage using PSC modulation (a) 50 Hz fundamental for switching frequency of 1002 Hz (b) 1kHz fundamental for switching frequency of 10.5 kHz

Similarly, the same equations from (2.14) - (2.16) for filter design were implemented for the NLC modulation technique as well. The NLC was also chosen for the implementation of 12-submodule MMC for which the filter design was modelled with a capacitive load of 6.8 µF. Figure 2.7 (a) and Figure 2.7(b) illustrate the frequency domain analysis of the switching voltage waveforms using NLC of an offline simulation with open-loop control for 50 Hz and 1 kHz respectively. It is clear from both graphs that NLC has a very non-characteristic spectrum with several lower order harmonics close to the fundamental frequency. However, this is highly noticeable due to the lower number of submodules in the circuit built for this study. Hence, filter values should be designed in such a way as to avoid such lower order harmonics for the NLC modulation algorithm. For the 50 Hz fundamental frequency, the n_1 value is chosen to be 1 keeping in mind no baseband harmonics and the value of n_2 is chosen to be 57 based on the harmonic frequency having the highest magnitude (i.e., 2.85 kHz). The arm inductance and the capacitance for a 50 Hz fundamental frequency are found as 91 mH and 266 Ω respectively. The same case was done for the 1 kHz fundamental frequency where the values of n_1 and n_2 are considered to be 1 and 57 respectively since the highest harmonic frequency was found to be at 57 kHz. Hence using the equations (2.14) - (2.16), the arm inductance and arm resistance were calculated to be 0.22 mH and 13.3 Ω respectively.

2.3.1. Modulation Technique Trade Off's

The switching voltage frequency spectrums for both NLC and PSC modulation algorithms depict a set of trade-offs, as can be seen in the above frequency spectrums. Based on the number of submodules and the phase shift provided for the carrier signals in the PSC method, the switching harmonics are moved to N times the switching frequency or 2N times the switching frequency. In addition, the filter design can reduce the magnitude of carrier harmonics that are located far away in the frequency spectrum. Because the switching harmonics are already present in the frequency spectrum, the circuit can be designed with a smaller arm resistance and arm inductance, resulting in a less bulky and complex circuit. Furthermore, constructing a high

voltage prototype with a larger number of submodules would benefit from the design of reduced filter inductance and resistance. Table 2.1 shows the trade-offs of NLC and PSC modulation techniques, as well as the factors that should be addressed while designing a filter.

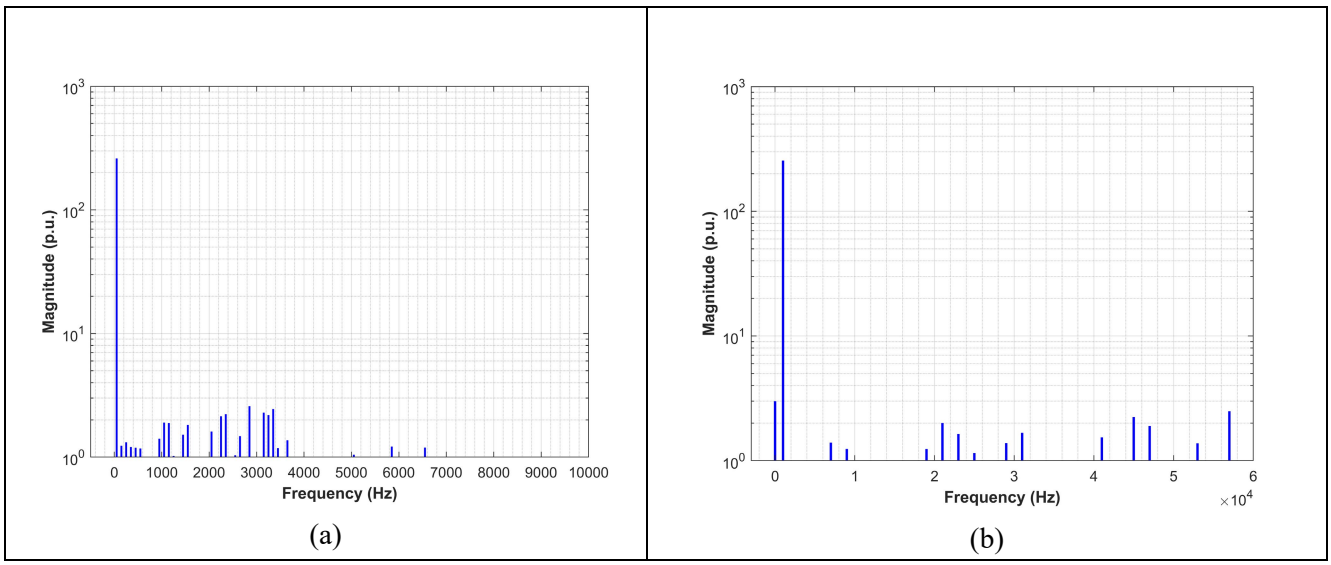


Figure 2.7: Frequency Spectrum of switching voltage using NLC modulation (a) 50 Hz fundamental frequency (b) 1kHz fundamental frequency

However, the NLC algorithm and its frequency spectrum analysis notice providing lower order harmonics and a non-characteristic spectrum. The design of the filter for NLC has shown the inductance and the resistance to be very bulky when compared to that of the PSC requirement. As the number of submodules might increase and there is a requirement to test higher voltages, the NLC might demand a higher and bulkier filter requirement for the circuit. This is not desired for the HV application and requires a less complex setup. Based on the filter need and the harmonic spectrum, PSC is chosen as a better modulation approach for a high voltage prototype. In the next chapters, both PSC and NLC modulation approaches have been explored and their respective waveforms have been realised, while this report only shows experimentation with a low voltage level.

Table 2.1: Trade-Off Table of PSC and NLC

	PSC	**NLC**
Switching Harmonics	$2NF_{sw}$ for 2N+1 Levels NF_{sw} for N+1 Levels	Non-Characteristic Lower order Harmonics
Frequency of commutation	Higher – high switching losses	Lower – low switching losses
Filter Design	Less bulky	Bulky

3. Understanding of the OPAL-RT Target and Software

3.1. Methods of Digital Simulation

The digital simulations are based on the system's ability to perform computations quickly enough to provide the needed outputs. The fixed sampling time determines the precision with which the quickest transients may be recorded [12]. The simulation methods are classified as follows depending on their computational capability based on the simulation time step.

1. Offline Simulation
2. Real-Time Simulation

3.1.1. Offline Simulation

The offline simulation technique of testing enables the creation of even the most complicated circuits by selecting components from specific libraries. In an offline simulation, the CPU conducts the computations quicker or slower than the selected time step, depending on the circuit complexity [12]. This simulation approach cannot imitate the behaviour of a genuine physical system or certify its completeness in control testing since the computing period might be longer than a selected time step. For instance, testing and examining the control algorithm of a power electronics circuit. The offline simulation validates the circuit's control logic and mathematical computations, as well as its fundamental function. However, in this scenario, the variable processing time at each time step makes it time-intensive to test the systems.

3.1.2. Real-time Simulation

The real-time simulators can process even the complex systems in the exact way an offline simulation can do but generate results during real-time. Real-time corresponds to the actual clock time in this case. The time required for the computations during real-time is lesser than the actual simulation time step and hence provide outputs fast enough even before the next sample of the time step [12].

Real-time simulation allows to design and test of prototypes before building an actual system for real. For example, external hardware, also known as a System-Under-Test (SUT), with the help of input and output communication can be interfaced to a virtually present system, modelled inside a real-time simulator, and simulated in real-time.

A converter model is running for a set time step in a real-time simulation, where 10 seconds of simulation equate to the current clock time. As illustrated in Figure 3.1, the fixed time step indicates that the controller should be able to examine the measurements, compute, and send or receive commands within this set period of time. If the above-mentioned considerations are not met within a fixed amount of time, an overrun occurs. This is an undesirable occurrence that occurs during real-time simulation, implying that the controller may not act as intended. This may lead to the simulation taking over the next time step for computations that had to be finished inside the previous fixed time step, which would be a violation of the definition of real-time. In

real-time simulations, the simulator should be able to complete functions in the same amount of time that a physical mimic would [13].

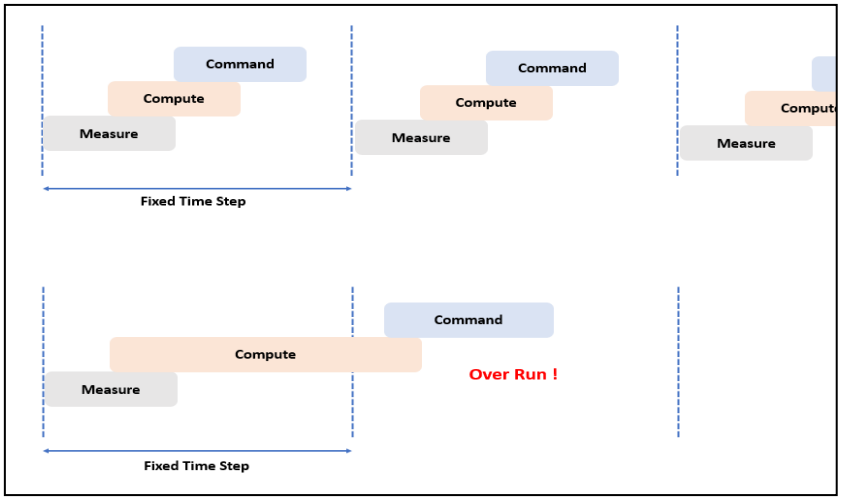


Figure 3.1: Fixed time-step conditions in Real-time simulation

3.2. Methods of Testing

The real-time simulation offers different methods of testing based on the focus for testing:

1) Rapid Control Prototyping (RCP): Facilitates the verification of different algorithms of the controller, where the controller is modelled in the real-time simulator. The simulator is interfaced with the real hardware of the MMC which includes real IGBT's and capacitors. This is a solution when a controller design must be analysed before building physical real hardware.
2) Hardware In Loop (HIL): The MMC which is also known as the plant model is designed in the real-time simulator while a real controller is used to control the simulated MMC. This type of testing supports the verification of the compatibility of the controllers with designs of the plant model. This is a wise solution when the real plant is not ready.
3) Model In Loop (MIL): The Model in Loop allows to provide a proof of concept and perform preliminary studies. In this case, both the MMC plant model and the controller will be simulated inside the simulator itself [14]. This is performed in the absence of I/O ports in the system.

While the initial two methods involve the construction of expensive equipment in terms of the plant or real controller hardware, the third solution entitles a better solution to perform preliminary studies.

3.3. OP4510 Target

The OP4510 target is equipped with a multicore CPU and a Kintex 7 FPGA from Xilinx. The CPU and the FPGA communicate with each other using the PCI express link. The FPGA is capable of executing at a sub-microsecond level and handling the input and output communication between the real hardware and the target. Since the I/O connections are connected through the FPGA, the accuracy of the input and output signals are very high. The basic standard configuration of this target consists of 16 analog input channels, 16 analog output channels, 32 digital input and 32 digital output channels. These input and output ports are connected to the external hardware using DB37 connectors as shown in Figure 3.2. Another additional 12 channels of digital input and output channels can be included using the RS422 ports. They also consist of four Small Form-factor Pluggable (SFP) in the front interface that is used to extend the I/O capability by connecting other MUlti-

System Expansion (MUSE) eligible targets like OP5607, OP4520 and OP4200 [15]. These SFP ports can also be used to connect to the external hardware. This OP4510 target is one of the targets used in processing real-time simulations performed in the further chapters of this thesis. However, no external hardware was connected to the target when the Model-in-Loop simulations were performed.

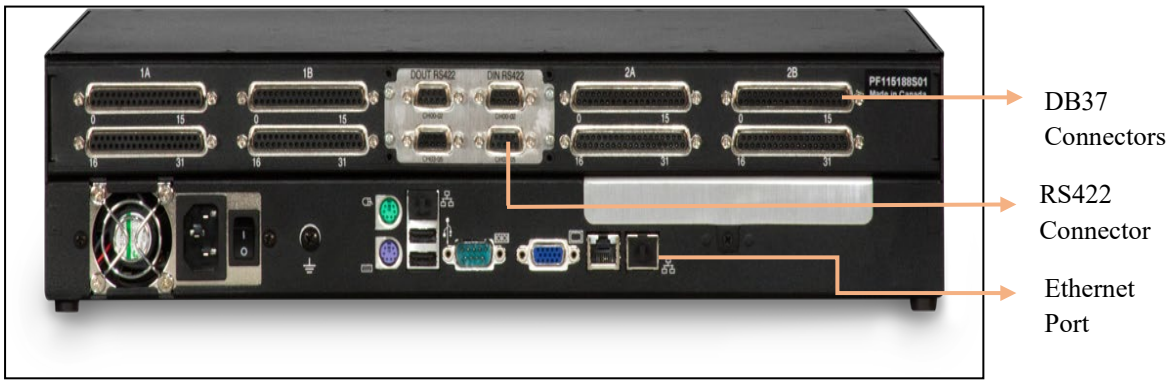


Figure 3.2: Rear Side of OP4510 Target

3.4. RT-LAB

RT- LAB from OPAL-RT is a software tool that runs on the Host PC and their physical connection with the simulator is secured using the local area network via the Ethernet cable. RT-Lab is responsible for opening models designed with compatible software packages like MATLAB/Simulink as its front-end editing platform and consists of a separate Simulink toolbox which is custom made to configure the I/O interfaces [16]. This is similar to any other Simulink toolbox, where the blocks should be dragged, dropped and connected to their inputs and outputs. These blocks function as a medium to generate the C source code automatically and are later translated into the target simulator and executed in real-time. RT-Lab also provides a user-friendly GUI that allows to measure and verify the signal values in real-time. The models to be computed and their corresponding GUI must be categorized into different subsystems with few RT-LAB customized blocks, for the simulator to understand and process as expected. RT-Lab also allows execution of models in different simulation modes namely [16], [17]:

1. Simulation: Accelerated simulation that is run on the simulator but faster than executing on the Simulink interface. No synchronization takes place since there are no external communication links. The model continues with a new time step after the previous time steps' computation is complete.
2. Software Synchronized: The model runs in real-time and the synchronization is done internally in the target simulator using the CPU clock. However, in this case, there is no external I/O's connected but a model in the loop simulation takes place.
3. Hardware Synchronized; The model simulated in the simulator is connected to external hardware with some physical input/ output signals and specific blocks are used to intimate where the external clock is located. The simulation runs in real-time, and the input and output board clock is used to synchronize the simulation.
4. Simulation with Low-Priority: This is used very rarely when one is working on other applications and the simulation is necessarily a low priority at that moment. The simulation will be executed in such a way that the computation of the subsystems is less prioritized.

Functions using RT-LAB:

- Logging data
- Modifying circuits and models in Simulink
- Changing values of parameters in a circuit during real-time
- Monitoring Dynamic signals
- Adding specific blocks to the model to be capable of running in real-time

3.5. RT-Events

Simulation models designed are generally run on different types of solvers which includes a discrete set of events. These events can be only recognized correctly and with minimal errors utilizing variable step solvers, which identify events at even the smallest time steps and then compute them, despite the model's complexity requiring a considerable simulation period. However, loading the model into the real-time simulator does not enable variable step solver types during code generation. Hence real-time simulation systems can only run with a discrete-time step. In addition to this, simulation models that are run on variable step solvers, while ran at a discrete-time step result in some events detected between sample times, evaluated only at the occurrence of the time step after the event has occurred. This causes some delay in the solving of the sampled event, thus causing an erroneous result. To avoid delay of such critical events in real-time, OPAL-RT consists of a separate Simulink block set called RT-EVENTS. This block set has an in-built algorithm that can compensate for the occurrence of events between sampling instants [18]. They are modelled in such a way that the input that they consider to these blocks is a compensated value for an event not occurring at the sampling instant. However, the RT-Events efficiency of how well it can detect events also depend on the time step based on the frequency requirements of the models. RT-Events can support two kinds of data types namely-

1. RTE Boolean
2. RTE Double

It is necessary to note that any block from the RT-Events library can accept an input of both data types for a signal [19]. The output from an RT-Events Block is Boolean by default, while it can be converted to a double type suitable for a normal Simulink model using a data conversion block.

3.6. eFPGAsim

Real-time simulator CPU's are capable of computations that require a sampling time in the range of milliseconds [20]. When dealing with gating signals and attaining high switching frequencies, power electronics models demand a smaller and more precise time step. To overcome this drawback of the simulators and to work with computations requiring sub-microsecond time steps, FPGA based real-time simulations are proposed to solve complex modelling. While FPGA programming skills are required to comprehend writing code and working on FPGA, OPAL-RT has developed a solution that uses eFPGAsim to translate a full model into the FPGA of the real-time simulator. It should also be known that the I/O interfaces connected to the real-time simulators also communicate through the FPGA for faster communication. Building a new model in the FPGA, on the other hand, does not prevent communication with external hardware. The most significant benefit of FPGA is that it performs all of its computations in parallel. Hardware in Loop Testing is supported by eFPGAsim, in which the plant model is simulated in the target simulator and the real controller is external hardware. While the plant is replicated in the FPGA, the controller may be developed and simulated in the CPU instead of a real one, allowing for a preliminary investigation utilizing Model in Loop testing.

The eFPGAsim's Electrical Hardware Solver (eHS) is in charge of reprogramming the FPGA and solving models created with circuit editors like as PLECS, SimPowerSystems, and PSIM [20]. By translating the circuit model into VHDL code, which the FPGA understands, this solution can program the FPGA. This is similar to the automatic code generation done on MATLAB and Simulink. It should also be mentioned that eFPGAsim can test several controller variants, which may be given as actual hardware, as a model from within the CPU, or as an input from the OPAL- RT's internal PWM generators. However, control algorithms cannot be designed inside the FPGA using the eHS solver offered in eFPGAsim. Another software package called RT-XSG of eFPGAsim, which is detailed in Section 3.8, may accomplish this design.

3.6.1. General Structure of Modelling with eFPGAsim

The basic interface of the real-time simulator is as shown in Figure 3.3. The Host PC runs the RT-LAB from OPAL-RT and it is connected via the ethernet to the target computer (Real-Time Simulator). The real-time simulator is made up of a CPU and an FPGA, the latter of which has several I/O ports for connecting to an external controller or real hardware. A PCIe network link connects the CPU with the FPGA, allowing them to communicate.

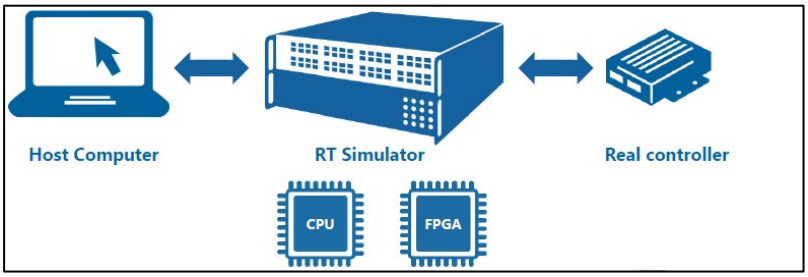


Figure 3.3: General Structure of Real-time Setup [21]

During an eFPGAsim simulation, the CPU may keep slower models like those of a mechanical system or control, while the FPGA can hold models like power electronics that need quick reaction times, as illustrated in Figure 3.4. Although models may be generated in the FPGA, they can only be initialized from the CPU with the aid of the eFPGAsim toolbox's specialized blocks. As a result, two models are necessary to create a model in eFPGAsim: the CPU Model and the FPGA Model, with the former initializing the latter.

3.6.2. eHS CPU Block and OpCtrl Block of the Simulink Library

The two important blocks required for the initialisation of the FPGA model are the eHS CPU Block and the OpCtrl Block. When designing, the CPU model should be created in the same way as an RT-LAB compliant model is created. The OpCtrl block is used when external hardware comes into the picture either using Digital and Analog I/O's or to access Internal PWM generators. In this case, the Hardware synchronised mode should be chosen for simulation in RT-LAB and this allows to set the firmware and provide details to its right configuration file. The eHS CPU Block allows the RT-LAB to recognise the FPGA model that must be translated into the FPGA [22]. The amount of gate signals and input sources required for the associated FPGA Model determines the inputs to the eHS block, as illustrated in Figure 3.5, and the total number of measurement signals received from the electrical model in the FPGA model file determines the outputs.

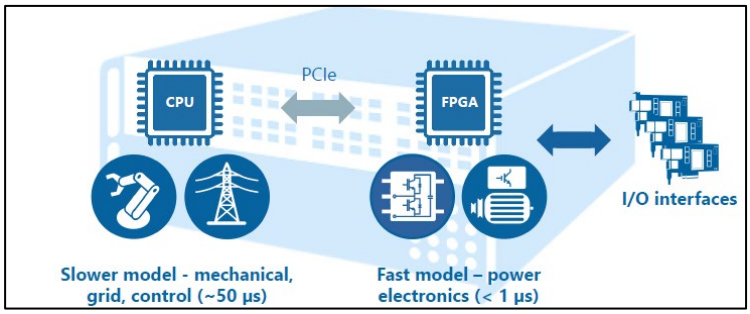


Figure 3.4: Representation of Models built inside the CPU and FPGA[21]

The choice of the eHS Solver time step, which allows for quicker FPGA computing, is another circuit aspect to keep in mind. The input channel of the eHS block can accept three different choices of inputs namely-

1. From the CPU Model
2. External analog input from the real hardware
3. Input from the internal Sine wave generators which the user can configure by oneself.

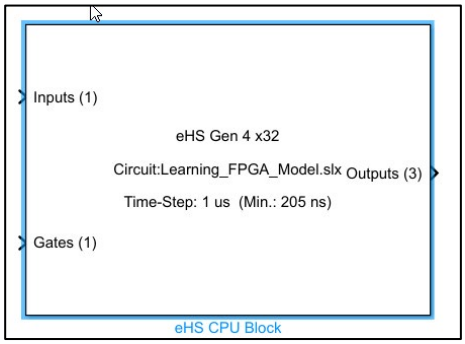


Figure 3.5: eHS Solver Simulink Block used In the CPU Model

Similarly, the Gate inputs accept three choices namely-

1. The CPU Model
2. As a digital input from the external hardware
3. As a PWM generator present internal to the FPGA.

The eHS solver's various form factors provide distinct capabilities for building models. The number of semiconductor switches, inputs, outputs, or states that any plant model in the FPGA can have is determined by these form factors. The electrical solver supports the following form factors: eHSx16, eHSx32, eHSx64, and eHSx128 [23]. More complicated models with a greater number of switches, inputs, and outputs are supported as the form factor increases. In addition, the time step that each form factor may achieve is dependent on the model's complexity.

3.7. Architecture of FPGA

FPGA circuits were first popular in the digital processing business, where they could provide customer-specific solutions. These devices have evolved so quickly that they now have a tiny structure with greater capability. Despite the fact that the structure appears to be modest, it is a technique that has helped to create the most complex circuits at a greater expense. In recent times, there are companies like Intel, Altera, Xilinx and National Instruments have come up with FPGAs in different configurations [24]. The cost of these configurations varies with the number of logic gates and the capacity of computations possible by each FPGA.

The FPGAs are similar to the Programmable Logic Devices (PLD), the former is made up of thousands of logic gates together being the larger version while the latter in hundreds.

As seen in Figure 3.6, the FPGA's structure resembles a two-dimensional array of logic blocks that can be programmed to provide the appropriate solutions. A routing network connects the logic blocks in an array, allowing them to communicate with one another [24]. They can also be connected to external inputs and outputs that can be customized according to the application's needs. Each logic block has logic gates that the user can program. A Lookup Table (LUT) is a list that contains the input states for each of these logic gates, as well as their outputs.

In the market, there are two types of FPGAs: (a) One-time Programmable FPGAs and (b) Reprogrammable FPGAs. A one-time programmable FPGA allows for one-time design and makes a physical connection between the logic blocks, making it impossible to re-program. Reprogrammable FPGAs come in a variety of formats, with SRAM-based FPGAs storing configurations in the SRAM until the FPGA is turned off. Every time these FPGAs are turned on, they should be reprogrammed. EPROM FPGAs, on the other hand, store settings even after they are turned off, avoiding the need for reprogramming, as opposed to SRAM FPGAs [24].

The HDL language specifies the logic that must be implemented in the user-specific application. In a simulation environment, a model can be checked for errors and validated if necessary. Additionally, a compiler can translate the model into HDL code, which contains the logic design. If the application requires it, this logic architecture can be written into the FPGA's Logic blocks and memory.

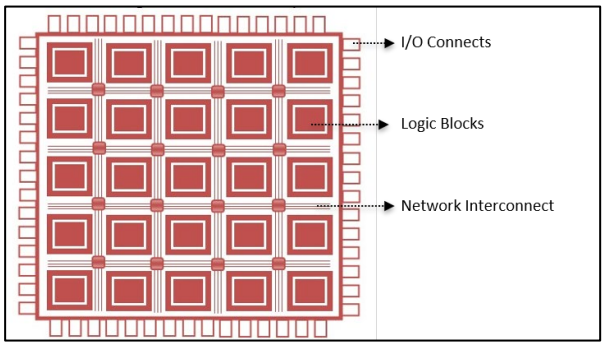


Figure 3.6: FPGA Architecture

3.8. RT-XSG

RT-XSG is a feature of eFPGAsim that allows you to program control algorithms in the FPGA. These electronics circuits only recognize HDL language, which aids in the development of the model inside the FPGA in the form of logic gates and memory. Because generating HDL code for an FPGA is a difficult procedure for a user who is familiar with model creation using MATLAB/SIMULINK, RT-XSG provides a simpler way to generate HDL code that can be used to reconfigure the FPGA. The RT-XSG from OPAL-RT uses the Xilinx system Generator provided by one of the FPGA Developers name Xilinx to generate the HDL Code. This software is linked to Simulink, which converts a model-based design into an HDL description in a way that the FPGA can understand. Hence this makes it suitable for a user with no prior knowledge of HDL to convert models for the FPGA [25].

The Xilinx System Generator is made up of libraries that contain blocks that the FPGA can process. The FPGAs can understand all of their values as binary data rather than decimal data. As a result, the HDL description ensures the generation of a binary file that is readable by the FPGA. Furthermore, once the model is designed and verified for errors, the HDL descriptive code can be automatically generated in the MATLAB/SIMULINK environment.

3.8.1. Compatibility with FPGA Chips

Building models in Simulink for FPGA's is different from the normal simulations as Simulink signals must be converted from double format to a fixed-point format using the dedicated Xilinx blocks. Also, each signal must be defined independently for the input value it can represent. For instance, the total number of bits a signal consists, the number of bits assigned for a fractional part and also if the signal must hold a positive or a negative sign. The Xilinx blocks like **Gateway in** and **Gateway out** helps to convert the Simulink doubles format to a fixed-point format and vice versa respectively. The model blocks present between these two blocks would be the circuit that can be configured onto the FPGA chip [25].

3.8.2. Accessing the External I/O boards

There I/O boards connected to the simulator can be well accessed with the help of the interface blocks present as a Simulink block set provided by OPAL-RT. The most important block for the accessing of the I/O boards is the 'Hardware Configuration' block where all the interfaces of the I/O connected to the target simulator are listed. This block contains a description of the type of I/O mezzanines (Analog In/Out or Digital In/Out) in the target and their location and slot number. It also shows the type of target the FPGA chip is installed into. The communication between the external hardware and the target simulator is accessible through the I/O boards linked through the FPGA. Hence, if a model is built in the FPGA and has some external hardware communication alongside, both their computations will take place parallelly and each of them will not experience any priorities.

3.8.3. Mandatory Blocks of RT-XSG

Some Xilinx and RT-XSG dedicated blocks are important for the proper verification and generation of the VHDL code for the FPGA [26].

1. System Generator Block
2. Synthesis Manager
3. Hardware Configuration
4. Version Block

The Hardware Configuration block is already discussed in Section 3.8.2.

The Synthesis Manager is an OPAL-RT priority block that lets you choose the FPGA development board you want to use to configure your design model. This is also responsible for setting up the bitstream and configuration file generating processes.

The Version Block aids in the management of the various versions of the configuration file and the generated bitstream file. It should be emphasized that every little modification to the model necessitates the regeneration of the bitstream file, making managing multiple versions of the same model difficult. As a result, the user can change the names of the bin files and conf files using this block.

The System Generator is a mandatory block for any model that has to be built onto the FPGA board. It includes information about the type of FPGA device to be used, the FPGA-based language to be utilized, the design's clock frequency, and even the Simulink period. This is a Xilinx block and OPAL-RT ensures that all the required settings to this block are automatically set using the Synthesis Manager block [26].

3.8.4. Communication between CPU and FPGA

Designing in RT XSG allows sending and receiving signals from CPU and FPGA. Some dedicated blocks that permit communication during runtime are DataOut Recv, DataIn Send that is used to receive data from FPGA and send data to the FPGA respectively in the CPU Model. The DataIN and DataOUT block are used in the FPGA Model to receive data from the CPU and send data to the CPU, respectively [25], [26].

The DataIn Send block is used in the CPU model in sending data to the FPGA at the CPU time step. This block allows sending data in the double or uint32 format. These blocks can support only 32 ports for data transfer and hence when larger data must be sent, concatenation of data is done within the 32-bit before sending to the FPGA. However, on the FPGA side, the data must be separated and reinterpreted depending on the data required for further computation. It also can send around 250 words from the same port, each of 32-bit known as time multiplexing. The default mode of this block is without time multiplexing, sending one data per CPU time step.

The DataIN block is present in the FPGA model which receives data from the CPU. This block is similar to the DataIn Send block and can have up to 32 ports with each port having 32 bits. The default option of this port is the synchronous mode where one data is received at one CPU time step [26], [27]. The time multiplexing is related to the asynchronous mode when multiple words can be received at a CPU time step but the exact time in which the FPGA receives the data is not known. These blocks can be provided with inputs that will be used during an offline simulation. When the generation of bitstream takes place, the inputs are disabled and are not used anymore. Further, these inputs are henceforth obtained from the CPU DataIn Send block from the CPU model.

The DataOut Recv blocks obtain the data into the CPU during its run time. They work the same way the above two blocks function consisting of a maximum of 32 ports returning a double or uint32 format. The data types of the received data can be changed using two other library blocks provided which allow converting double to a fixed point or vice versa. They receive words at one data per CPU time step.
DataOut Block in the FPGA sends words back to the CPU and they use two methods of sending data namely FIFO or registers. During the register mode for example, if there are six FPGA time steps before one CPU time step, the last computed value of the FPGA time step will be sent to the CPU. FIFO method is used during time-multiplexing methods where a buffer can store up to 250 words of the data [26], [28]. In this case, the decision of samples that will be stored in the buffer depends on the Most Significant Bit (MSB) of each 32-bit sample. The samples with only an MSB value of 1 is stored in the buffer and the rest are ignored. Hence there is a lot of uncertainty about at what time instant the sample was generated. The block also consists of an output port that allows displaying data that is being sent to the CPU during an offline simulation, while this remains disabled during the generation process.

4. Implementation of Control in CPU with the plant in FPGA

4.1. Open Loop Control in the CPU of OPAL-RT

This section expands on the research into the use of the eFPGAsim software that was addressed in Section 3.6. A Model-in-Loop (MIL) simulation was performed and then the output waveforms were analysed. A MIL simulation here corresponds to the development of the plant (MMC) and open-loop control modulation, both as models but run in real-time. Hence, the MIL simulation is slightly different from the normal simulations. Using the eFPGAsim software, any power electronics circuit may be implemented in the FPGA for a faster response time. The MMC plant is implemented in the simulator's FPGA.

Since the OP4510 target that is used for the simulation consists of an installed eHS with the form factor of (x32) with Gen4 capability from OPAL-RT, there are certain limitations on the number of semiconductor switches, number of outputs and number of inputs that can be modelled in the FPGA. The following limits for building the model in (x32) is given in Table 4.1[23]. With eHS(x32), with the maximum number of switches possible being 48, a 12-submodule MMC was built. The individual submodule capacitor voltages (24 outputs), lower arm, upper arm current and the output current (3 outputs), the output voltage across the load (1 output) and switching voltage (2 outputs) are among the 30 outputs evaluated in the specified model. Other versions of eHS, such as x64 and x128 that allow building a larger number of switches or inputs and outputs, can improve and extend the model's potential to accommodate higher submodules [23]. Though each version has limitations on the number of semiconductor switches, inputs, and outputs, the total number of non-switching devices such as resistors, capacitors, inductors, and ideal transformers that a circuit can have is unrestricted. In addition to this, the half-bridge submodules of the MMC utilise the Loss Compensation Algorithm (LCA) concept developed by OPAL-RT for eFPGAsim. This feature provides an optimised half-bridge circuit that has reduced losses than a normal half-bridge circuit.

Table 4.1: Capability of eHS with form factor (x32)

Feature	Form Factor (x32)
Number of inputs	32
Number of outputs	32
Number of switches	48
Maximum number of states	112

While the MMC is flashed into the FPGA of the target, the control of the MMC can be implemented in the following three ways -

1. As real hardware that can communicate with the MMC through the input and output ports of the OPAL-RT target
2. In the CPU of the OPAL-RT target that runs at the CPU time step
3. In the FPGA of OPAL-RT target that can run at the time step of the FPGA based on the complexity of the model

Further sections of this chapter verify the capability of the control of the MMC, built inside the CPU of the OPAL-RT target. All of the functions are normally conducted inside the CPU, including the modulation approach algorithm for generating gate pulses, stabilizing the various submodule capacitor voltages, and monitoring output voltage and current. The CPU's calculation speed is slower than the FPGA's. The PCIe network establishes communication between the CPU and the FPGA, allowing data to be shared between them with a one CPU time step delay. Figure 4.1 shows the flow of the gate pulses from the CPU to the FPGA that consists of the MMC model designed using eHS [29]. The capability of reaching higher fundamental frequencies with CPU control is evaluated for modulation techniques PSC and NLC. The open-loop control using PSC modulation in the real-time simulator is implemented in two ways-

1. Without RT-EVENTS Blockset
2. With RT-EVENTS Blockset

The PSC modulation technique to generate gate pulses is developed in the CPU model by using relational operators to compare the reference wave and the carrier signals generated. The behaviour of the NLC modulation algorithm is analysed only without RT-EVENTS. The results in sections 4.2, 4.3 and 4.4 further represent the 2N+1 modulation for both PSC and NLC techniques. In the digital control implementation, the sampling frequency cannot be considered infinite and is run on a fixed time step. The offline simulations are also run on a fixed time step to acquire precise waveforms such that, the comparisons can be made of the same simulation mode. As accuracy is the primary goal of this investigation, the real-time simulations using eFPGAsim is compared to the offline simulations at a very small time step of 1 µs.

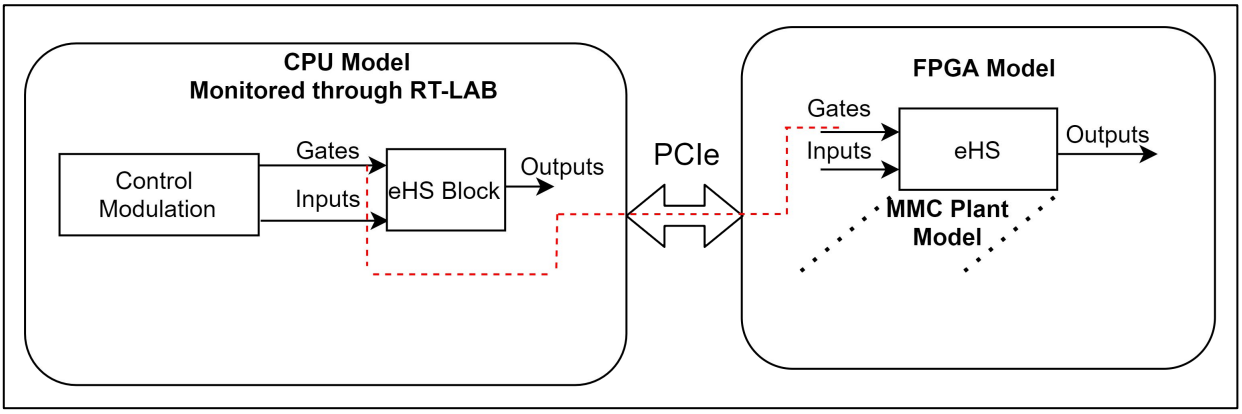


Figure 4.1: Flow of Control in CPU using eFPGAsim

4.2. Phase-Shifted Carrier Modulation with CPU Control – 50 Hz Fundamental Frequency

For a fundamental frequency of 50 Hz, a real-time simulation was performed with open-loop control of PSC modulation to provide 2N+1 levels at the output voltage. The MMC of 12-submodules being built in the FPGA using the eHS, allows the FPGA to process at a minimum time step of 1.48 µs. Despite the FPGA's ability to simulate at smaller time steps of up to 10 nanoseconds, the complexity of the 12-submodule circuit built inside with eFPGAsim lowers the computation performance. The control algorithm can be run at a minimum time step of 10 µs modelled inside the CPU when employing blocks without using RT-EVENTS. However, to ensure synchronization of clocks between FPGA and CPU, it should be ensured that one is the multiple of the other. Thus, to abide by this criterion, the FPGA is set to an explicit time step of 2 µs while the CPU is initially set with a minimum timestep of 10 µs. The DC-link voltage, the submodule capacitance, the arm inductance,

arm resistance and the load capacitance parameters for the MMC model implemented in the FPGA are tabulated below in Table 4.2.

Table 4.2: Parameter and Values of MMC for 50 Hz

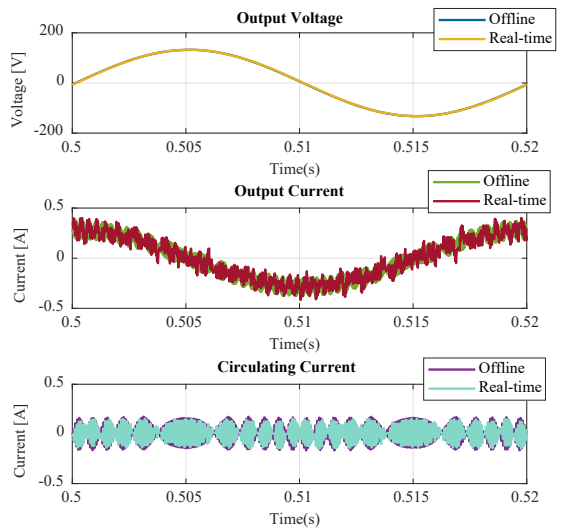
Parameter	Value
Number of Submodules	12
DC Link Voltage (V_{dc})	300 V
Submodule Capacitance	4 mF
Arm Inductance	1.32 mH
Arm Resistance	45 Ω
Load Capacitance	6.8 µF

For the fundamental frequency of 50 Hz, the switching frequency of the carriers for the PSC modulation was chosen as a non-integer multiple of the fundamental frequency to ensure balancing of the submodule capacitor voltages. The parameters chosen for the simulation in the CPU is tabulated below in Table 4.3.

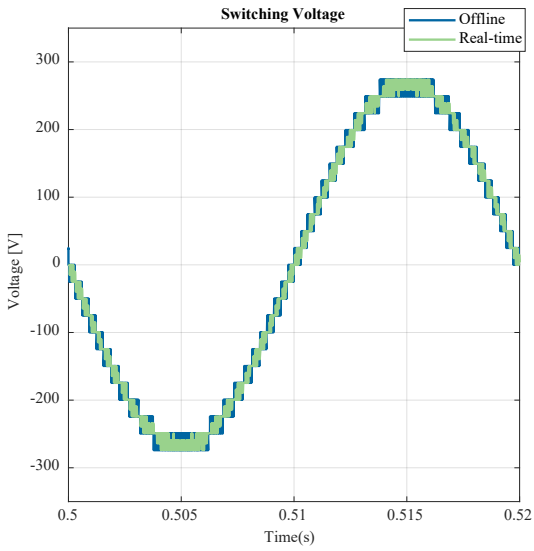
Table 4.3: CPU Model Parameters

Parameters	Values
Modulation Index	0.9
Fundamental Frequency	50 Hz
Switching Frequency (F_{sw})	1002 Hz
CPU Time Step	10 µs
FPGA Time Step	2 µs

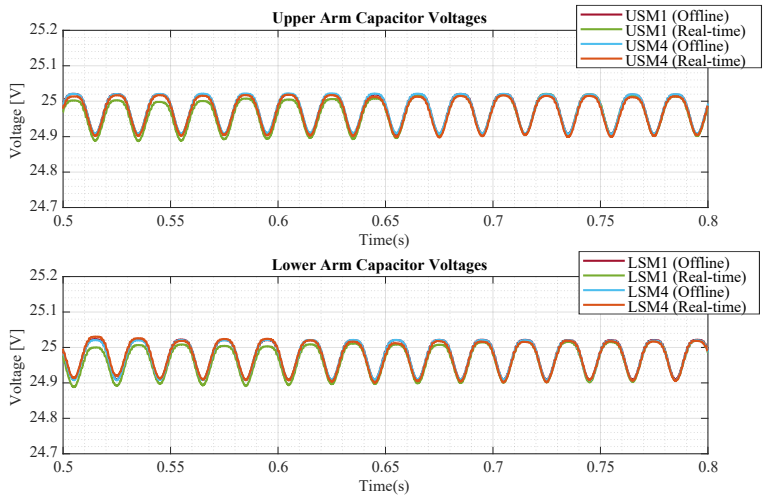
Figure 4.2 below shows the comparison of a discrete offline simulation run at a smaller time step of 1 µs and real-time simulation without RT-EVENTS for 50 Hz fundamental frequency.



(a)



(b)



(c)

Figure 4.2: Comparison of offline and real-time simulation without RT-EVENTS of a 50 Hz fundamental frequency Open-loop control using PSC modulation technique for 2N+1 levels (a) Output Waveforms (b) Switching voltage (c) Capacitor Dynamics of the Upper and Lower Arm

The waveforms clearly illustrate that the real-time results are substantially identical to the values obtained offline. The switching voltage waveforms of real-time simulation from Figure 4.2 (b) also show that they display the 2N+1 levels as expected like in the offline simulation. The small-amplitude variation observed at any level of the switching voltage of the real-time is a result of the sampling time of 10 µs which is higher than the offline simulation value of 1 µs. In addition, a PCIe delay of one sampling time is also expected, with which the plant inside the FPGA receives the gate pulse for the submodules. The circulating current also shows that it is zero when a voltage level change occurs and compensates for any imbalance in voltage at a particular voltage level. The capacitor voltage dynamics from Figure 4.2(c) of two submodules each from the upper (USM1 and USM4) and lower arm (LSM1 and LSM4) notice that the real-time simulation results match with the offline simulations perfectly with time.

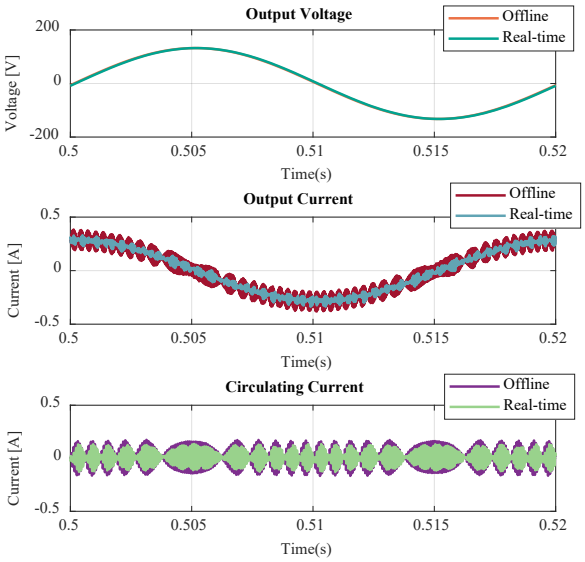
Likewise, the same experiment using real-time simulation was performed using RT-EVENTS for the fundamental frequency of 50 Hz. The changes to the model were made using relational operator blocks by RT-EVENTS block set for generating the gate pulses using PSC modulation and adding logical operator blocks from the same block set to separate gate pulses to the two IGBT's of the same submodule. As already indicated in the previous chapter, the RT-EVENTS blocks initiate an algorithm that can help in minimising the error in progress with time, which occurs due to discrete-time step solvers. The input to the eHS CPU block is a Boolean signal of gate pulses which is recognised by the eHS block after choosing RT-EVENTS in its property dialog box. The CPU can run at a minimum time step of 20 µs while using RT-EVENTS and the FPGA time step remains the same at 2µs considering synchronisation.

Figure 4.3 represents results with RT-EVENTS also notices that the output voltage waveform from real-time matches the offline simulation. This confirms that the computation speed of both the FPGA and the CPU is sufficient to perform the simulations as the considered switching frequency is also low. The output waveforms from Figure 4.3(a) show satisfactory outputs similar to the offline. Figure 4.3(b) presents the switching voltage and notices that the levels are not very clearly visible. These levels could be properly visible if they can be viewed through an oscilloscope. Since the OP4510 has some limitations in connecting to the oscilloscope it could not be further analysed. However, the comparison of Total Harmonic Distortion (THD) of the output

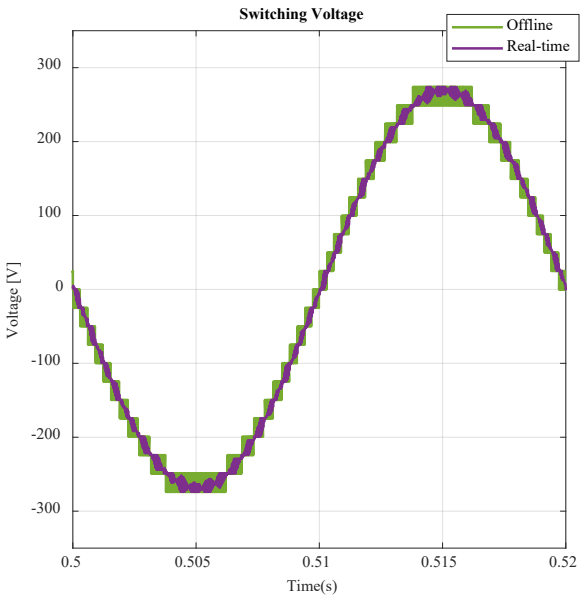
voltage waveforms between the three simulation approaches, namely Offline, Real-time without RT-EVENTS, and RT-EVENTS, shows that RT-EVENTS improves the results as shown in Table 4.4. This proves that the numerical algorithm involved in RT-EVENTS can reduce the harmonics to 0.09% quite closely comparable to the THD obtained from offline results, providing accurate results than the other normal discrete time step solvers. The capacitor voltages can be seen balancing over their average value as they progress over time close to the offline results. An advantage of the RT-EVENTS blocks being used with eHS is that the Boolean signal fed to the eHS has a greater time resolution than what a CPU model has. As a result, these boolean signals can be sampled at the explicit sample time mentioned for eHS [22]. The accuracy of RT-EVENTS is however subjected based on the minimum step size requirement, which is indirectly related to the switching frequency of the circuit.

Table 4.4: THD values of output voltage Open-Loop control simulations for 50 Hz Fundamental Frequency

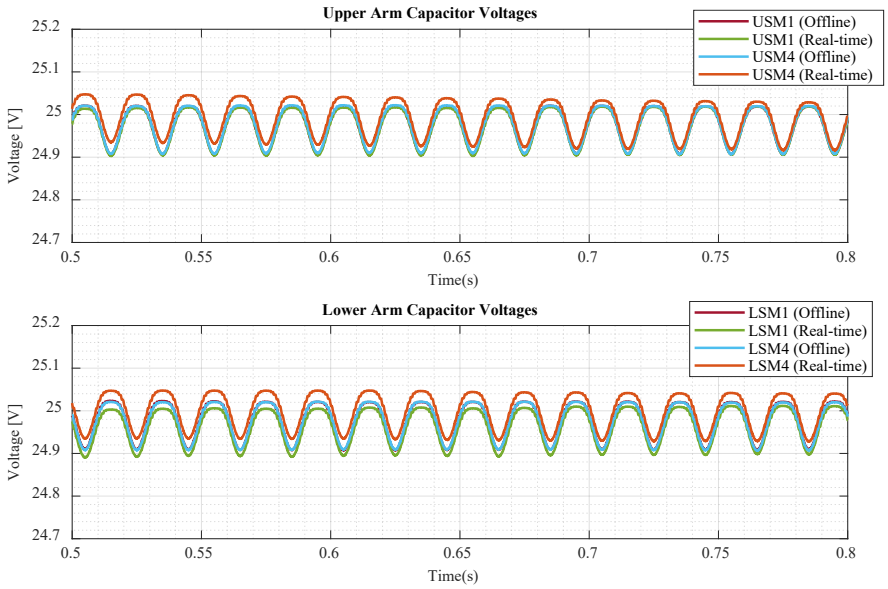
Type of simulation	Offline Simulation	Real-time without RT-EVENTS	Real-time with RT-EVENTS
Total Harmonic Distortion (THD) of Output Voltage(in %)	0.07%	0.14%	0.091%



(a)



(b)



(c)

Figure 4.3: Comparison of offline and real-time simulation with RT-EVENTS results of a 50 Hz fundamental frequency Open-loop control using PSC modulation technique for 2N+1 level (a) Output Waveforms (b) Switching voltage (c) Capacitor dynamics of Upper and Lower Arm

4.3. Phase-Shifted Carrier Modulation with CPU Control – 1 kHz Fundamental Frequency

The simulations performed for the 50Hz were repeated to evaluate the capability of the eHS to generate a 1 kHz fundamental frequency output voltage. An offline simulation was executed and compared with the two real-time simulation experiments without RT-EVENTS and using RT-EVENTS. The parameters set considered for the simulation on the CPU side is shown in Table 4.5. The switching frequency of the carrier wave is chosen as non-integer multiple values, to balance the capacitor voltages.

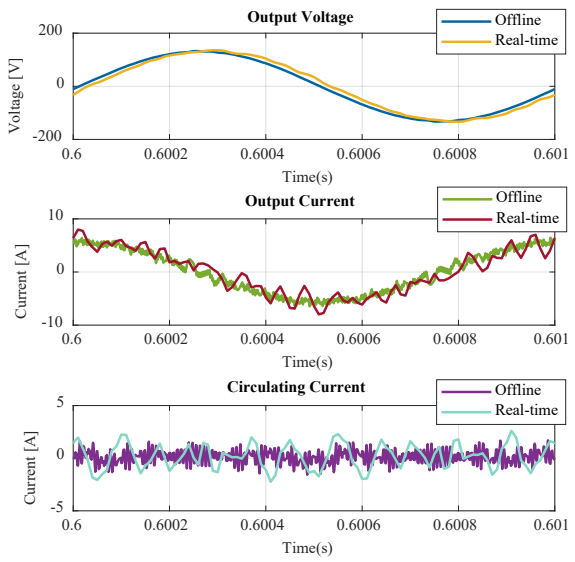
Table 4.5: CPU parameters for a 1 kHz

Parameters	Values
Modulation Index	0.9
Fundamental Frequency	1 kHz
Switching Frequency (F_{sw})	10.5 kHz
CPU Time Step	10 µs
FPGA Time Step	2 µs
Arm Inductance	15.2 µH
Arm Resistance	3.71 Ω
Load Capacitance	6.86 µF

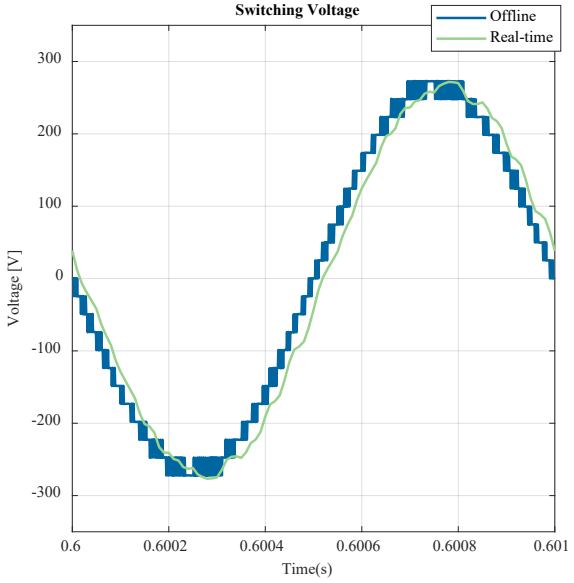
Figure 4.4 shows the waveforms while comparing the real-time simulations without RT-EVENTS with the offline simulation for 1 kHz. The offline simulation run at 1 µs time step indicates the switching voltage in Figure 4.4 (b) with 2N+1 levels but real-time results show the absence of voltage levels at the output. When semiconductor devices fail to receive gate pulses at an expected instant for higher frequencies, the building of a voltage level at that instant is disrupted. By involving its effect at a different time, the delayed switching

becomes accountable for interrupting the impending creation of levels as well. As a result of the gate pulse's involvement at a slightly different time, the switching voltage has a slight phase shift, which can be observed in the output voltage waveform. Around the period when most of the submodules remain inserted from the higher or lower arm, the output current in Figure 4.4 (a) barely exhibits any charging and discharging of the load capacitance, proving that the voltage level is not changing. The three charging and discharging instances seen at the peaks of the output current correspond to the only visible three voltage level changes in the switching voltage. While working with fast switching devices like IGBT in real-time, for better accuracy and stability it is expected to have a sampling time at least 10 times higher than the switching frequency [30]. In this case, the CPU time step can process only at a minimum of 10 µs and the switching frequency is also large. In addition to this, communication delays due to interaction between CPU and FPGA are also responsible for these unexpected results in real-time.

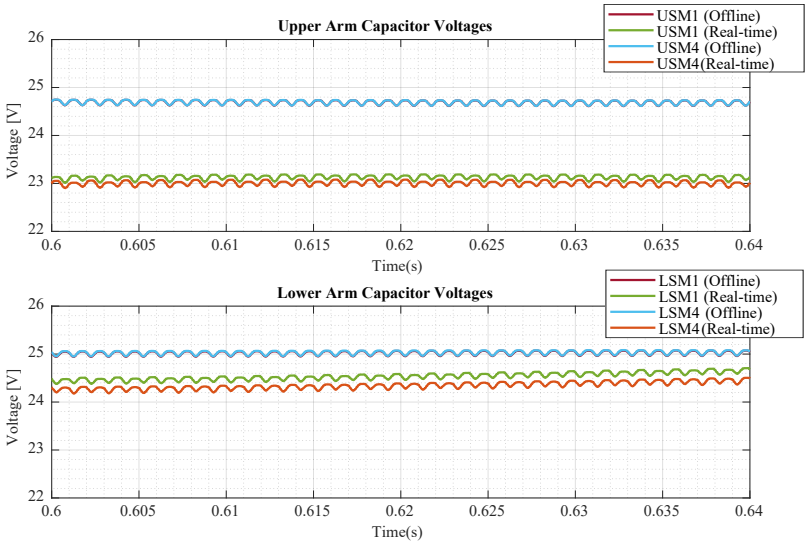
The capacitor voltages as seen in Figure 4.4(c), the offline simulation is attempting to efficiently balance the capacitor voltage. The real-time findings demonstrate that the charging and discharging of the submodule capacitors are not occurring at a faster rate to compensate for the change in their voltages, as shown in the circulating current waveform. Also, as the sampling frequency is insufficient to keep up with these transitions with the switching frequency, the capacitor voltage deviates farther away from its average value, which can be seen when simulated for a longer time.



(a)



(b)



(c)

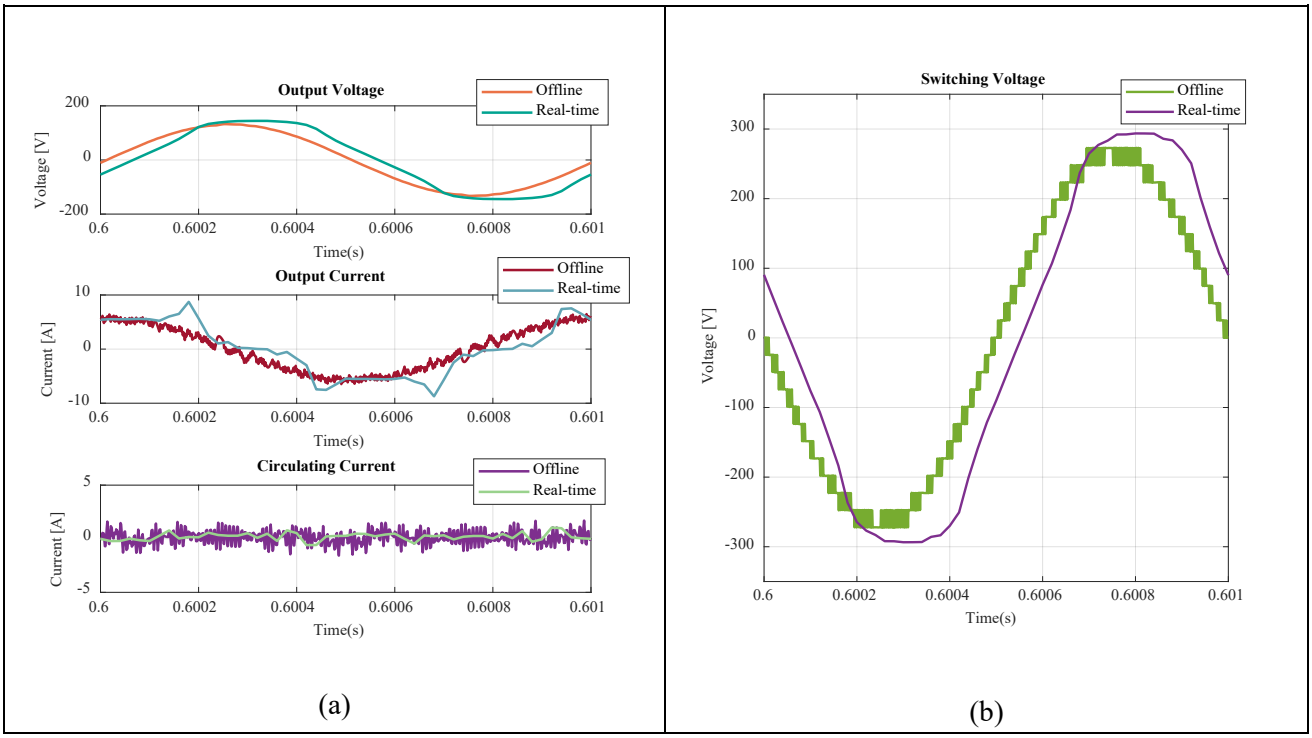
Figure 4.4: Comparison of offline and real-time simulation without RT-EVENTS results of a 1 kHz fundamental frequency Open-loop control using PSC modulation technique for 2N+1 level (a) Output Waveforms (b) Switching voltage (c) Capacitor Dynamics

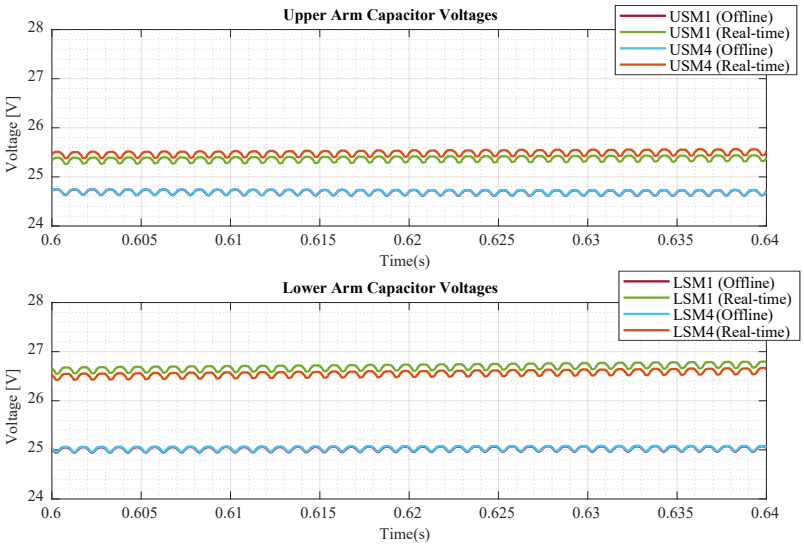
The simulation results with RT-EVENTS noticed that the output voltage has a small distortion at the peaks of the output waveform as seen in Figure 4.5(a). The peak distortions could be caused by poor PWM generation, in which a few submodules do not receive their gate pulses appropriately and are forced to stay inserted or bypassed for prolonged periods. This may also be seen in Figure 4.5 (a) which shows certain peaks that correspond to the moment of distortion in the output waveform. The accuracy of RT-EVENTS also deteriorates as the ratio between the switching frequency and step size decreases which is also a reason for such major discrepancies in results. It is known that RT-EVENTS being an event, or an interrupt-based method sends data every instant when an intersection of a reference wave and carrier wave is detected [19]. As the switching frequency for a 1 kHz signal is around 10 kHz, the simulator is forced to be interrupted multiple times during the same sampling instant due to the higher frequency. In addition to the RT-EVENTS behavioural features, the one-time step delay between the CPU and FPGA has a greater influence in this circumstance, affecting the precision of the findings. This switching delay resulting in a lack of accuracy causes multiple baseband harmonics to occur. Since there is an absence of switching of levels seen from Figure 4.5(b), the carrier harmonics are never visible at $2NF_{sw}$ thus resulting in a higher THD of 3.7% as tabulated in Table 4.6 while the THD of the without RT-EVENTS case was found to be comparatively lesser with 2%. Also, the capacitor voltages reveal a considerable imbalance over time. In addition, the circulating current response shows that the unbalanced submodule voltage is not charged sufficiently seen in Figure 4.5(a). Hence, these errors challenge the confidence of the real-time simulation results for higher fundamental frequencies.

Table 4.6: THD values of output voltage Open-Loop control simulations for 1 kHz Fundamental Frequency

Type of simulation	Offline Simulation	Real-time without RT-EVENTS	Real-time with RT-EVENTS
Total Harmonic Distortion (THD) (in %)	0.47%	2%	3.7%

The waveforms of 1 kHz indicate that both the results in real-time after implementation of control in the CPU show inaccurate results. To understand the issue better on why the switching levels never occurred consistently in both the simulations of 1 kHz, the CPU model was analysed. Figure 4.6 shows that triangular carrier waveforms are not generated completely due to the lack of smaller time steps. Since the carrier frequency is higher, it demands a smaller time step to sample these precise carrier waveforms. A PWM gate pulse is set to 1 at that sampling point when the carrier wave is lower than or equal to the reference wave and 0 when the carrier wave is higher than the reference wave. In OPAL-RT this comparison takes place by the process of linear interpolation by the target's CPU in the case of RT-EVENTS. With RT-EVENTS since multiple interpolations occur within one sample time, the RT-EVENTS is affected with a higher load to compensate for the errors within that step. Also, due to the insufficient sampling requirement, there are some parts of every carrier wave missing and not reaching its designated amplitude limits. Hence, the gate pulses for some submodules to get inserted or bypassed are never generated in the duration of those missing parts of the carrier wave. Figure 4.7 shows that since a comparison never happens at the peaks of the reference wave with a carrier waveform, the gate pulse is either completely turned off or turned on for some time.





(c)

Figure 4.5: Comparison of offline and real-time simulation with RT-EVENTS results of a 1 kHz fundamental frequency Open-loop control using PSC modulation technique for 2N+1 level (a) Output Waveforms (b) Switching voltage (c) Capacitor Dynamics

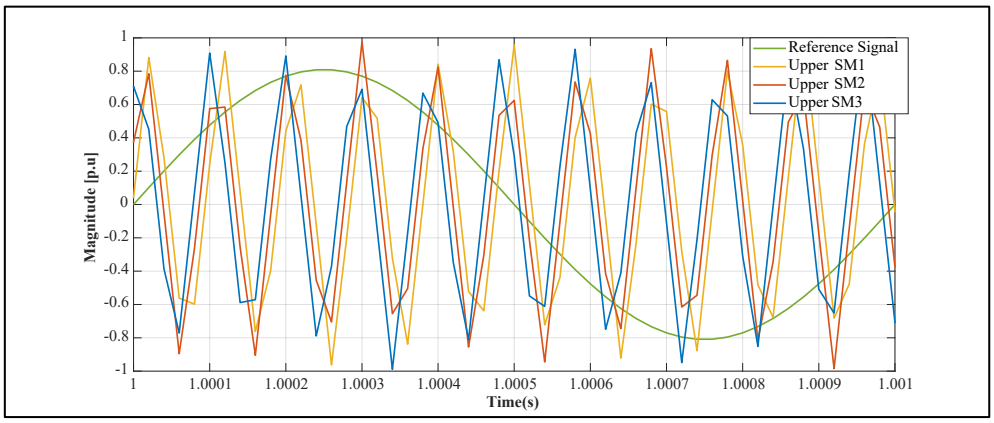


Figure 4.6: Carrier waveforms in real-time simulations for 1 kHz fundamental frequency

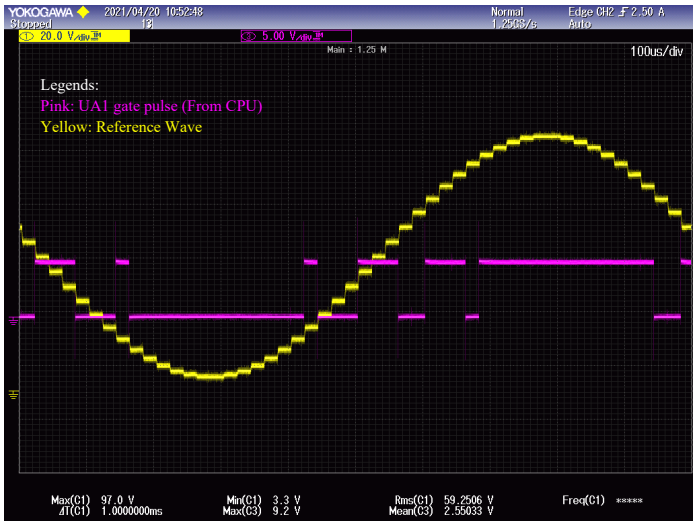


Figure 4.7: Gate Pulse Accuracy from CPU

4.4. Nearest Level Control Modulation – CPU Control

The NLC modulation technique built provides a sorting algorithm that decides the insertion or bypassing of every submodule in the upper and lower arm. This sorting algorithm ensures the balancing of all the capacitor voltages in each submodule. As previously seen in Section 2.3, the filter values have been calculated for the NLC algorithm technique. The NLC technique implemented provides 2N+1 levels and the parameter values used are tabulated in Table 4.7 below. Like the PSC modulation simulations, the modulation technique was implemented with an offline simulation and real-time simulation using eHS to compare the behaviour of the MMC in both cases. This will also provide an analysis of the capability of the real-time simulator and its limitations concerning the efficiency with which it can handle simulations of higher fundamental frequencies.

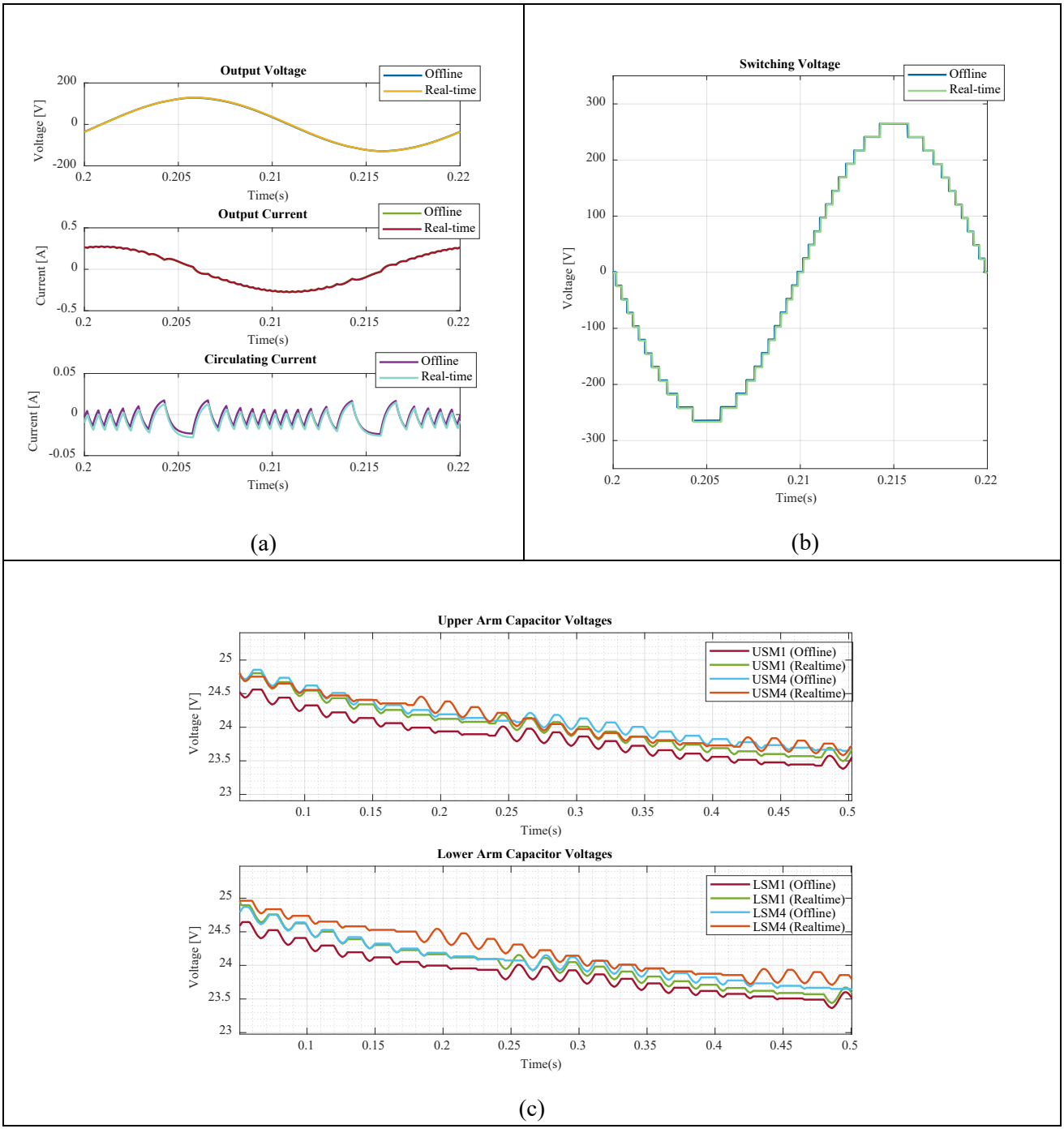


Figure 4.8: Comparison of offline and real-time simulation results of a 50 Hz fundamental frequency Open-loop control using NLC modulation technique for 2N+1 levels (a) Output Waveforms (b) Switching voltage (c) Average Capacitor Voltages

Table 4.7: Parameter values of the Circuit for 50 Hz fundamental frequency using NLC modulation

Parameters	Values
Modulation Index	0.9
Fundamental Frequency	50 Hz
CPU Time Step	10 µs
FPGA Time Step	2 µs
Arm Inductance	91 mH
Arm Resistance	266 Ω
Load Capacitance	6.86 µF

When comparing the offline and real-time simulation in Figure 4.8(a), the output currents and the output voltage simulated in real-time follows the offline simulation results for the 50 Hz fundamental frequency case thus the switching voltage in Figure 4.8(b) also shows the levels of the MMC. The circulating currents try to balance the voltage variation caused by the output current in the submodule by charging and discharging the submodule capacitor. Here, the circulating current is seen slightly larger at the peaks of the output voltage. This could be due to the insertion of the majority of submodules from one of the arms, necessitating compensation for the imbalance. The submodule capacitor voltage during offline and real-time simulation shows that the upper and lower arm voltages are drifting away from their average value and take time to balance the voltages. Since the 50 Hz filter parameter has its arm resistance value on the higher side, the longer it takes to balance the capacitor voltages. The difference in the balancing of the capacitor voltages during the real-time and the offline results are variable and are based on the number of submodules inserted in the arms. The choice of sorting as the 12-cycle sequence, implemented for the NLC is visible in Figure 4.8 (c). It shows that the upper or the lower arm submodule capacitor is charged and discharged over a 12-cycle sequentially trying to maintain a constant capacitor voltage. The THD for 50 Hz was calculated to be 0.7 %.

The parameter values chosen for the 1 kHz open-loop NLC modulation setup is shown in Table 4.8. Figure 4.9 displays a comparison of 1 kHz fundamental frequency results simulated in offline and real-time situations using the NLC technique. The real-time simulation results in Figure 4.9 (b) show that the switching levels are barely visible. They have shifted a little away in time when compared to the offline waveform that is also visible from the output voltage waveforms. This phase shift is due to the delayed gate pulse received by the submodule.

Table 4.8: Parameter values of the Circuit for 1 kHz fundamental frequency using NLC modulation

Parameters	Values
Modulation Index	0.9
Fundamental Frequency	1 kHz
CPU Time Step	10 µs
FPGA Time Step	2 µs
Arm Inductance	0.22 mH
Arm Resistance	13.3 Ω
Load Capacitance	6.86 µF

In this case, the CPU provides the reference signal for NLC sampled at the time step of 10 µs. It is known that in this technique, each submodule contributes to the formation of an output voltage level.

However, a smaller sampling frequency results in approximation errors due to which more than one submodule might correspond to the formation of the same level. The smaller sampling frequency can be responsible for the levels, not visible clearly. In the case of NLC, the requirement of the sampling frequency is of importance similar to that of a switching frequency in PSC modulation [31]. Also, like the previous situation at 50 Hz, since the number of inserted submodules in each arm is different and are not evenly distributed between the two arms, the capacitor voltages might deviate from their average value seen in Figure 4.9 (c). As the arm resistances are smaller for the 1 kHz design when compared to that of the 50 Hz filter, it is noticeable that the capacitor voltages try to balance themselves quicker by 0.3 seconds than in the 50 Hz case. The THD for 1 kHz fundamental frequency was 1.9 %. Thus, since the sampling time is also insufficient the overall comparison shows that the CPU is showcasing difficulty in managing a 1 kHz capacity of the fundamental frequency.

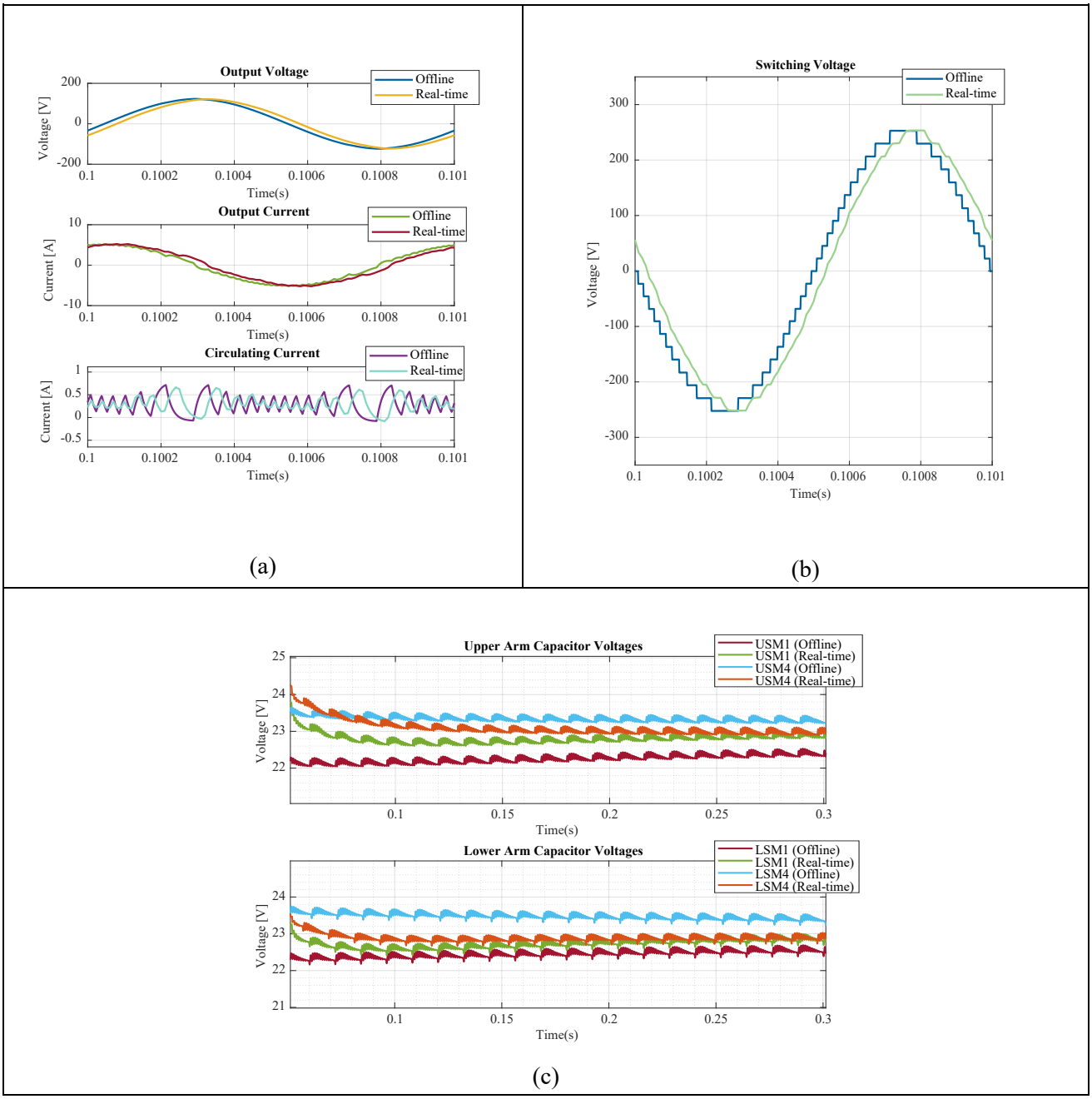


Figure 4.9: Comparison of offline and real-time simulation results of a 1 kHz fundamental frequency Open-loop control using NLC modulation technique for 2N+1 levels (a) Output Waveforms (b) Switching voltage (c) Average Capacitor Voltages

5. Control in FPGA

The simulations done using eFPGAsim and eHS in the previous chapter demonstrated that implementing an open-loop controller for the MMC, in the CPU of the OPAL-RT resulted in a compromise of the accuracy of the gate pulses. This trade-off can be divided into two categories, both of which are CPU-related. To begin with, the PCIe network causes communication delays in the CPU when performing computations and transmitting gate pulses to the plant. Second, the lack of a smaller time step for the sampling of triangular waveforms provided by the CPU. An approach to overcoming both issues is to consider the implementation of the control inside the FPGA of the OPAL-RT itself. The FPGA can handle a minimum time step of 20 nanoseconds which includes performing computations, the model complexity and contributing to the propagation of signals to its connected external hardware as well. As a result, the implementation of control on the FPGA is predicted to improve the precision of the gate pulses to the MMC. To test their capability, two alternatives for implementing the control in the FPGA of the OPAL-RT were discovered namely- the internal PWM generators and the RT-XSG Software to design a new controller for the MMC in the FPGA. The internal PWM generators create in-built triangular carrier waveforms that are compared with a modulating signal given as input, for the generation of gate pulses. The insights of the RT-XSG software, already discussed in Section 3.8 require some understanding of modelling in the FPGA and is comparatively a tedious process than the internal PWM generators. The implementation of the control in the RT-XSG software is not covered in this thesis due to time constraints and the complexity involved in them. Since PSC modulation provides more advantages than the NLC technique seen from Section 2.3.1, the focus is based on only the PSC modulation henceforth and the internal PWM generators were chosen for the analysis in this Thesis.

The implementation of control inside the FPGA avoids the PCIe communication delay contribution between the CPU and the FPGA. Figure 5.1 shows a model flow with the control in the FPGA using the internal PWM generators. The path (1) in Figure 5.1 represents the flow of using internal PWM generators to provide gate pulses to the gate channel of the eHS block using eFPGAsim. The flow (2) represents the flow of gate pulses from the internal PWM generator directly to the external real hardware using optic fibers in the case of an RCP setup. An example that the PWM generators provide gate pulses is illustrated in Figure 5.2. However, the precision with which the PWM takes place has to be verified. Further sections, show the analysis performed on both the flow methods and the interpretations made from them.

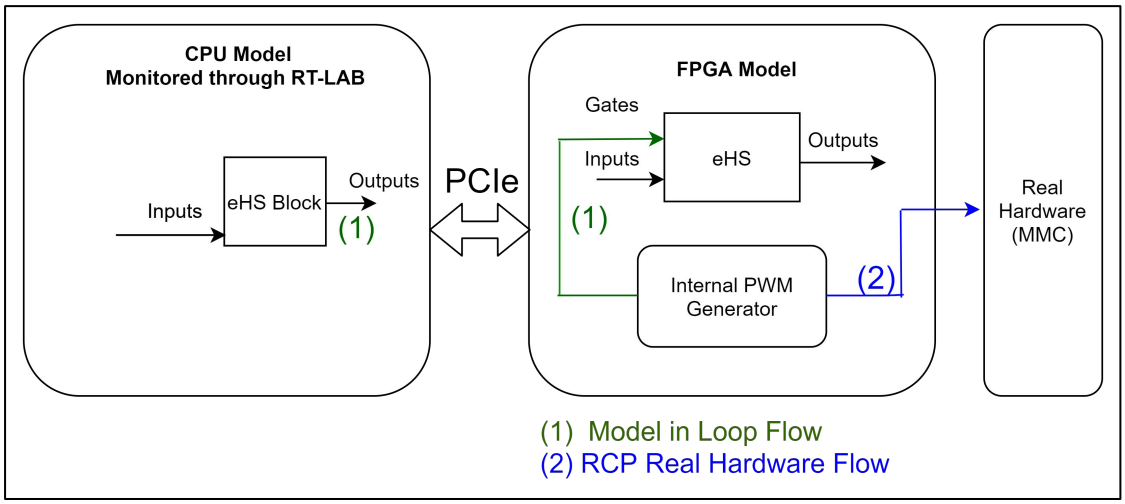


Figure 5.1: FPGA Control Flow in OPAL-RT

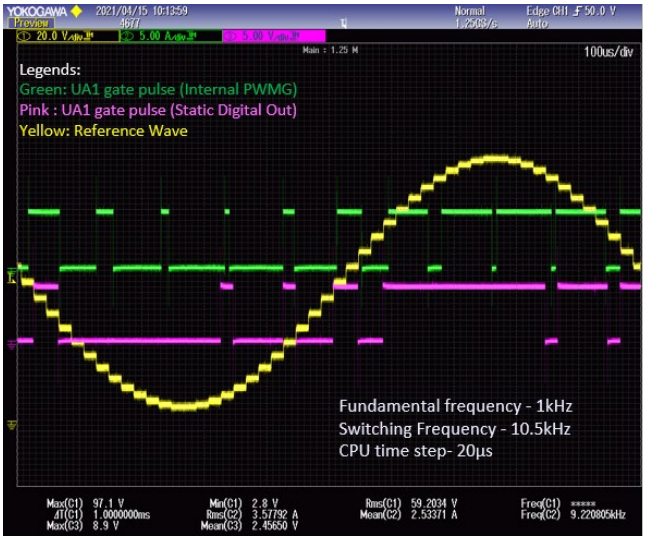


Figure 5.2: Gate pulses using Internal PWM Generator

The internal PWM generators are accessed using the PWMout block provided by OPAL-RT as shown in Figure 5.3(a). One PWMout Block has the capability of generating eight PWM gate pulses each. They also provide an option to choose complementary signals in which case, four signals can be generated along with their complementary signals [32]. The number of PWMout blocks to access the internal PWM generators is decided by the configuration of digital output channels in the OPAL-RT simulator one owns. The availability of the PWMout ports in the simulator can be obtained from the configuration file provided for the target. These digital output boards can either be used as static digital outputs, which allow communication of data from the CPU to the real hardware via the I/O, and as PWMout ports which provide communication of gate pulses from the FPGA itself.

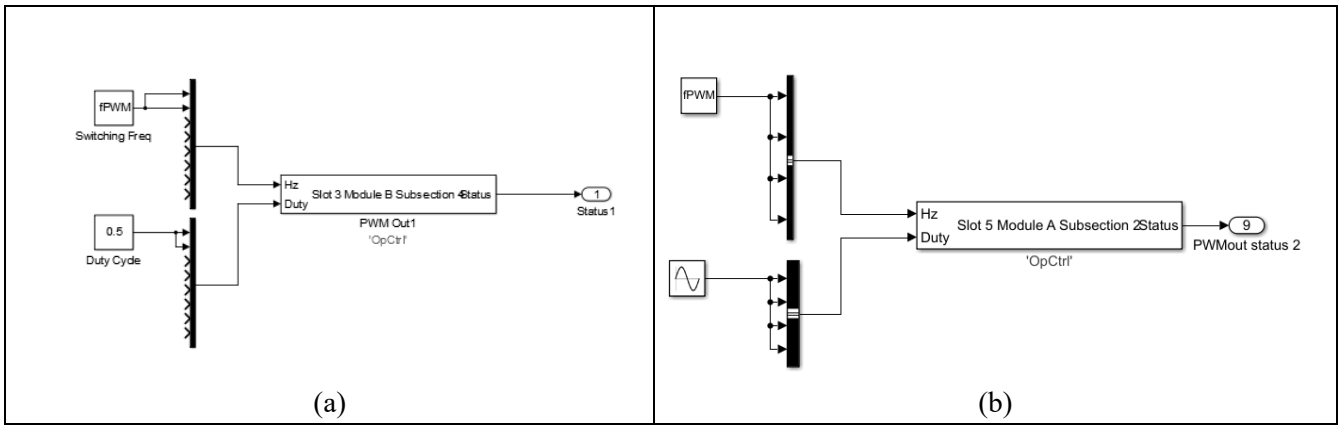


Figure 5.3: (a) PWMout Block with a constant duty cycle (b) PWMout Block with a variable duty cycle

Figure 5.3(a) shows a constant duty cycle of 0.5 for two channels with a switching frequency of 500 Hz. The output of the simulation was obtained through an oscilloscope as shown in Figure 5.4. This shows that the PWMout block provides the gate pulses correctly when a constant duty cycle is provided as an input. It should also be noted that the two gate pulses are not provided with any phase delays between the two gate pulses.

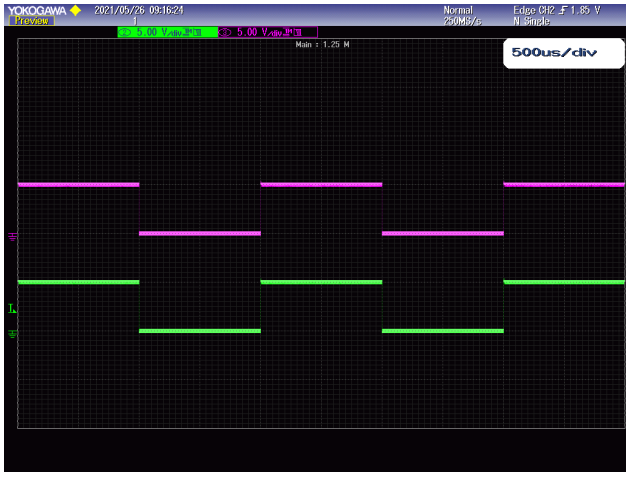
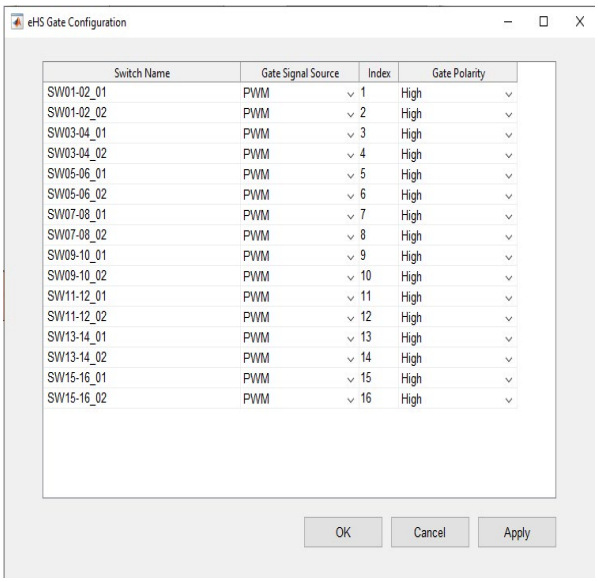


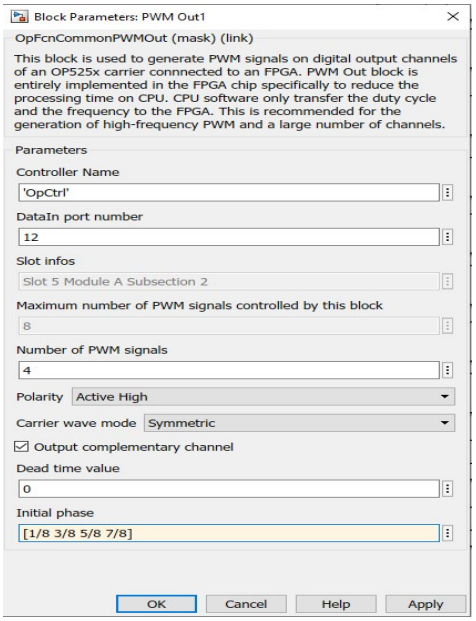
Figure 5.4: Constant Duty Cycle using PWMout Block

5.1. Internal PWM Generators in eFPGAsim for a 4 Submodule MMC

The Model in Loop flow as shown in Figure 5.1 was first considered to implement the control in FPGA using the internal PWM generators. To initially understand the behaviour of the PWM generators for the MMC, a four-submodule model was built using eFPGAsim to test the PWMout block. To access this block in the eFPGAsim, the eHS block was configured to the PWM generator option as shown in Figure 5.5(a) so that the eHS is aware that the gate pulses are sent from FPGA to the IGBT's of the MMC modelled in the FPGA. For the MMC, the sine reference wave as shown in Figure 5.3 (b) is provided as the input for obtaining a variable duty cycle to aid the comparison with the triangular carriers. The PWMout block also allows generating multiple triangular or sawtooth carriers by providing an initial phase delay in its properties box shown in Figure 5.5(b). In this case of the MMC, it is necessary to provide a phase shift of $2\pi/N$ between the carriers representing each submodule in the upper and lower arm respectively.



(a)



(b)

Figure 5.5: (a) eHS Block Gate Configuration Setup (b) PWMout Property Dialog Box

Each submodule of the four-submodule model consists of two IGBT switches that receive complementary gate pulses. Hence the complementary option available in PWMout is chosen to provide the complementary IGBT of each submodule. One PWMout block is capable of generating gate signals for 4 submodules only. In this case, two PWMout blocks were used to generate gate signals for the whole circuit. The first PWMout block was input with a reference wave with no phase shift, corresponding to the upper arm reference. While the second PWMout block with a π phase shift is the reference wave for the lower arm.

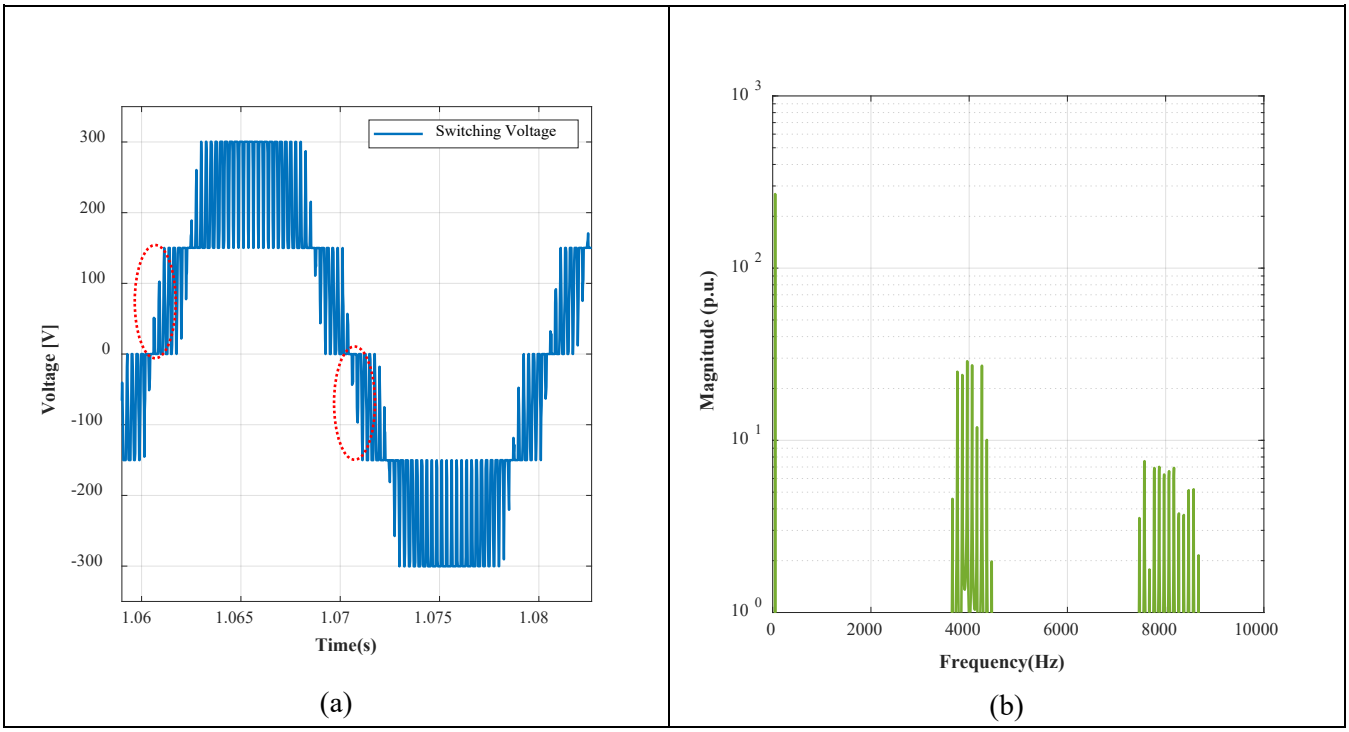


Figure 5.6: Analysis of a Four-Submodule MMC for N+1 Levels (a) Switching Voltage Waveform for 50 Hz fundamental frequency (b) FFT analysis of the Switching Voltage

The first experiment was aimed to generate N+1 levels for a fundamental frequency of 50 Hz and a carrier waveform switching frequency (F_{sw}) of 1002 Hz for balancing the capacitor voltages by choosing a non-integer multiple of the fundamental frequency. For an even number of submodules, the upper and lower arm carriers were phase-shifted by 2π/N (90º) and the initial phase delay between the upper and lower arm carriers was zero. The switching voltage waveform in Figure 5.6(a) reveals the presence of N+1 levels and the FFT of the switching voltage in Figure 5.6(b) shows that the harmonics are concentrated at multiples of the carrier frequency at NF_{sw} (4 x 1002Hz = 4008 Hz), with some sideband harmonics also present. The switching voltage waveform, however, has a few observations that must be examined, even though the FFT appears compelling and as expected for N+1 levels. It is visible from Figure 5.6(a) that some incomplete voltage pulses (circled in red) are evident at each of these levels which is a result of an additional insertion or bypass of the submodules on the upper or the lower arms at an instant it is not expected to occur. Since a voltage level is obtained by adding the gate pulses present at any given time, a delayed generation of gate pulses results in the gate pulse expected at that level not being added.

The second experiment with the eHS was performed again to generate 2N+1 levels for a fundamental frequency of 50 Hz and the switching frequency (F_{sw}) of 1002 Hz as the previous experiment. In this case, the

carriers of each arm remained phase-shifted by 2π/N (90º) and the phase delay between the upper and the lower arm carriers was chosen as π/N (45º).

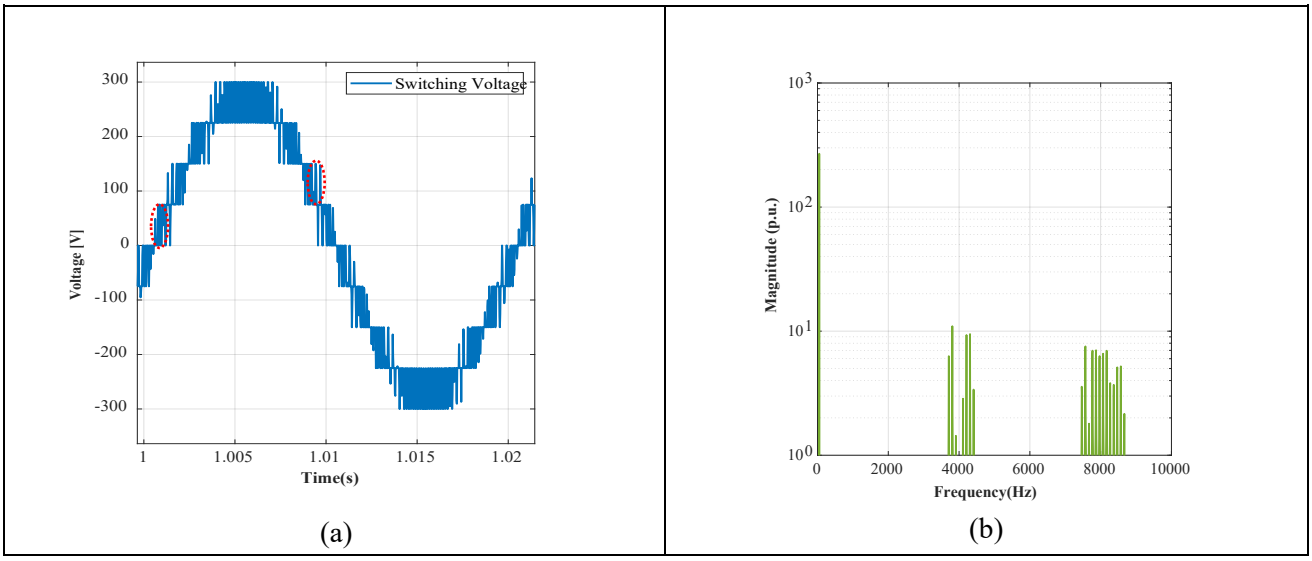


Figure 5.7: Analysis of a Four-Submodule MMC for 2N+1 Levels (a) Switching Voltage Waveform for 50 Hz fundamental frequency (b) FFT analysis of the Switching Voltage

Figure 5.7 (a) depicted the presence of 2N+1 levels; nevertheless, similar to the N+1 scenario, the emergence of pulse patterns reveals inconsistencies in the development of each voltage level because it is influenced by incorrect carrier waveform phase displacements. Figure 5.7(b) shows an atypical expectation of carrier harmonics for the 2N+1 level arrangement based on the FFT of the switching voltage. When 2N levels are formed, the carrier harmonics should be pushed to $2NF_{sw}$ (8016 Hz), but the results reveal carrier harmonics comparable to the N+1 level arrangement. The inconsistencies in waveforms of both N+1 and 2N+1 levels can be explained as the lack of smaller time steps by which the modulating signal is sampled. The variation from the expected results, demands to analyse of the generation of the gate pulses to calculate the phase displacements of the carrier waveforms produced by the PWMout block. The eFPGAsim simulations in the OP4510 target enable access to the PWMout blocks via a virtual present slot. The dedicated port number for using these virtual slots can be found in the target's configuration file. As a result, the gate pulses were observed using an OP5600 target. The oscilloscope may use its RJ45 ports to tap signals from the digital output card with this target.

5.2. Internal PWM Generators in the OP5600 target

To understand the working of the internal PWM generators block, a simple model with just the use of PWMout block and barely any complexities of the MMC was built in the OP5600 target. In this case, the OP5600 target consists of digital output boards, which are linked to the RJ45 jacks present in the front of the simulator. A cable can be routed from the RJ45 connection to the small BNC port provided in the target, via which the output can be collected in the oscilloscope, based on the output of the channel one wants to monitor. In this scenario, the PWMout block was used to generate gate pulses for a four-submodule architecture for N+1 and 2N+1 levels. The gate pulses of the upper arm's first submodule (UA1) and the lower arm's first submodule (LA1) were chosen for examination in both circumstances. The oscilloscope data was then compared to the offline simulation results from MATLAB/Simulink. The simulations ran in real-time and offline were both set with a discrete simulation time step of 20 µs.

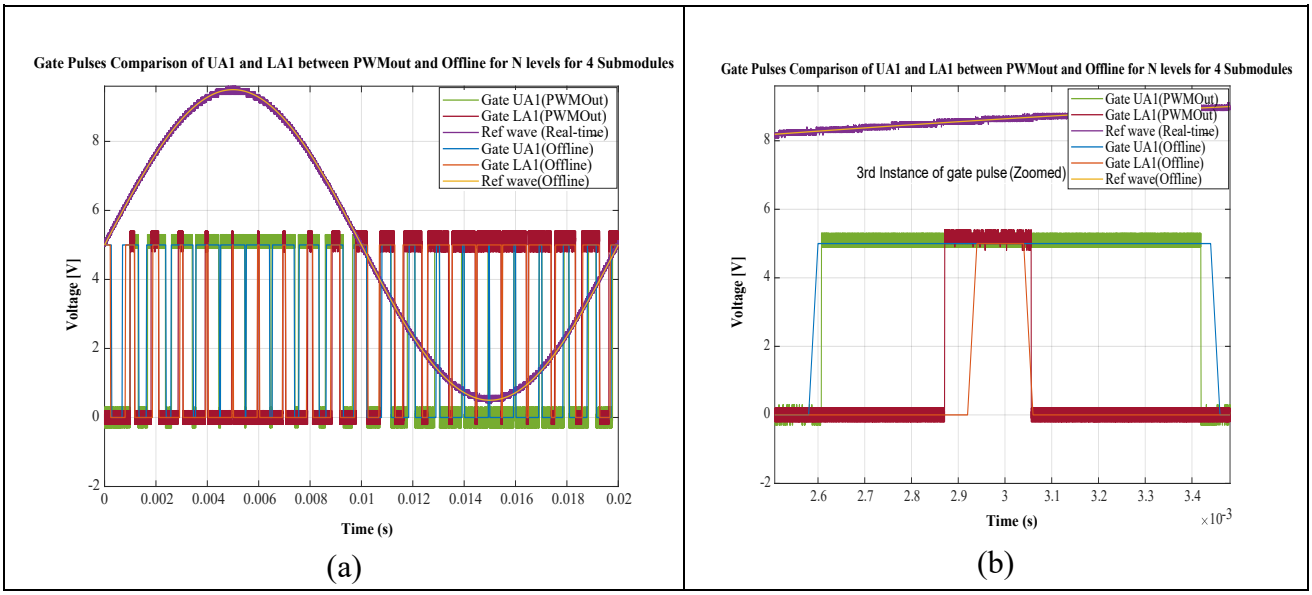


Figure 5.8: Gate Pulse Comparison of UA1 and LA1 for 4 Submodules N+1 Levels (a) Gate pulses of PWMout block Vs Offline simulation (b) Zoomed image of the 3rd instance of the gate pulse of one cycle

Figure 5.8(a) shows the gate pulses obtained from the PWMout block and the offline for UA1 and LA1 for the first cycle of 50 Hz fundamental frequency, right after the start of execution. The reference wave obtained from the real-time corresponds to the upper arm reference wave. Looking closer at it would show how the sampling time of the CPU is also reflected in the reference wave. The PWMout block is chosen to utilise triangular carrier waveforms for the PWM generation and hence it considers symmetric sampling based on how it is configured internally. In general, the carriers of UA1 and LA1 do not have any phase delay and overlap on one another at N+1 levels. But, the modulating wave used for comparison with the carrier wave for the lower arm is phase shifted from the upper arm modulation wave by an angle of π. As a result, both the reference waves meet the carrier wave at slightly different instances of time to generate the PWM gate pulses. Hence, regardless of whether a phase difference is provided to the carriers of the two arms, a slight phase delay develops between the UA1 and LA1, as illustrated in Figure 5.9. Since the symmetric sampling is considered, the phase delay is calculated by the difference between the midpoint of the UA1 and LA1 gate pulse, then multiplying the result with the switching frequency and then by 360°.

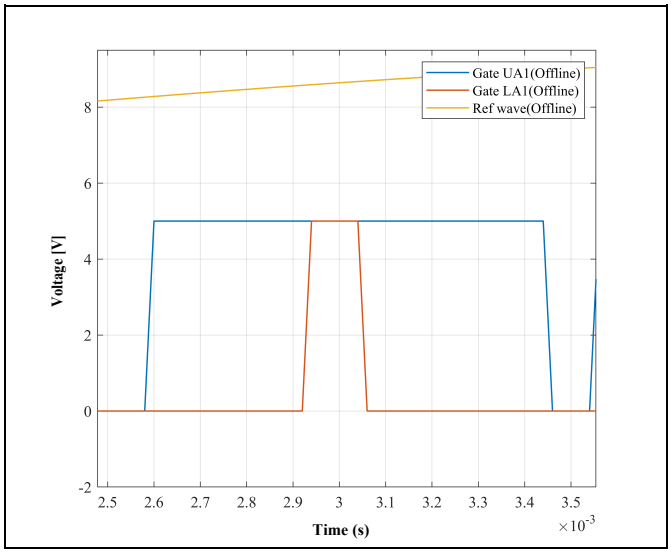


Figure 5.9: Gate pulse comparison of UA1 and LA1 in Offline Simulation for 4 submodules N+1 levels (3rd instance Zoomed)

This way, the phase delay between gate pulse UA1 and LA1 was found to be around 7 to 10 degrees in the offline simulation.

The variable duty cycle makes it difficult to conclude that the phase difference at every instance of the gate pulse is the same. Hence to analyse it further, multiple different instances from Figure 5.8(a) was considered to calculate the phase delay between UA1 and LA1. The 3rd instance as shown in Figure 5.8(b) and the 7th instance from the positive half of the reference cycle and the 11th and the 13th instance from the negative half of the reference cycle were observed. Table 5.1 shows the phase delay values of the offline and the real-time results during PWMout observed for 4-submodule N+1 levels.

Table 5.1: Phase Delay between UA1 and LA1 for 4 Submodule N+1 levels in Offline and Real-time (PWMout)

	3rd Instance	7th Instance	11th Instance	14th Instance
Offline Phase Delay	10.8°	10.8°	10.8°	6.8°
Real-time (PWMout) Phase Delay	18.07°	11.63°	24.16°	11.7°
Difference	7.2°	0.8°	13.4°	4.9°

Table 5.1 shows that the phase delays obtained between the UA1 and LA1 from the PWMout block have an additional phase delay than that seen in the offline simulations. Since the gate pulses provided by the PWMout are moved slightly ahead of what is expected as in the offline simulation, the gate pulses do not properly contribute to every level in the MMC. The misplacement of some gate pulses in time is a result of not building a proper voltage level completely which is seen in the form of some abrupt voltage pulses. This wrongly displaced firing of the gate pulses, in turn, is responsible for the unexpected placement of the carrier harmonics as well.

Table 5.2: Phase Delay between UA1 and LA1 for 4 Submodule 2N+1 levels in Offline and Real-time (PWMout)

	2nd Instance	5th Instance	13th Instance	17th Instance
Offline Phase Delay	54.1°	43.3°	39.6°	50.5°
Real-time (PWMout) Phase Delay	67.7°	49.5°	26.7°	56°
Difference	13.6°	6.2°	13°	5.5°

In the case of 2N+1 levels, the carrier waveform of UA1 and LA1 are phase-shifted for the modulating waveform to deliberately intersect the carrier waveform at a different instant in such a way that it contributes to forming an additional voltage level when compared to what the UA1 and LA1 at N+1 levels would be expected to contribute. The same analysis done for N+1 levels was performed with the 2N+1 levels where Table 5.2 shows the phase delays obtained between the UA1 and LA1 gate pulse from the PWMout block and the offline simulation. The phase delays in the PWMout and the Offline do not match each other and have a difference between them in the range from 5° to 15°. Due to these variable phase shifts, not all the submodules

that should contribute to building the voltage for one particular output level is exactly able to get inserted at the given time instant. Ultimately this leading to the failure of pushing the carrier harmonics to higher frequencies as expected in the case of the 2N+1 levels. Another inference worth noting is that, since the phase difference between two carrier waves in the same arm is almost 90 degrees apart, the additional phase difference added by the PWMout does not drastically modify the output levels that are supposed to be present in both the N+1 levels and 2N+1 levels voltage waveforms as seen from the Figure 5.6. But whether the outcomes are as expected can be verified only by analysing their carrier harmonics.

5.3. Internal PWM Generators for a 12-Submodule MMC

The internal PWM generators that generate gate pulses from the FPGA were implemented for the 12-submodule MMC. Since the eHS used in the OP4510 target had a limitation in the availability of the number of PWMout blocks that can be used which could not supply 48 PWM outputs, an existing RCP setup already available as a lab setup was used for this analysis.

5.3.1. Description of RCP Setup

The RCP setup consists of the target OP5600 from OPAL-RT which is coupled to an external hardware plant that consists of 12 submodules in each arm. The DC link capacitors are two capacitors that are connected at a midpoint and are charged by applying a DC source to them. Every submodule in each arm is a half-bridge and is connected in series with one another. They consist of an IGBT power module with a voltage rating of 600 V and a current rating of 15 A [33]. Capacitors in the submodule consist of a value of 4 mF. The target is made up of Digital I/O ports that are connected to optic fiber receivers, which are subsequently connected to the plant hardware through optic fibers. Individual submodules have two digital inputs, one for enable and the other for receiving a PWM gate pulse, and one digital output that conveys the capacitor voltage. All of the submodules and communication boards linked to the target receive a 5 V power supply. The upper and lower arms are connected to the LEM transducers, which measure the switching voltage and output voltage. The circulating current is measured with a current probe. This test was carried out with no load applied. The OPAL-RT interface was used to control the plant's hardware. The OPAL-RT is connected to a host PC via ethernet, which serves as the control interface. It is important to note that the upper and lower IGBT's have an inbuilt dead-time of 2 µs. Every submodule includes a built-in control for generating complementary gate pulses.

The digital output cards present in the target can be used in two ways by a controller created in the OPAL-RT: Static Digital Out (SDO) ports and PWMout (PWMO) ports. The DataIn Port Number for each of these methods, as specified in the configuration file, will remain the same. In the case of the use of SDO in the control, the computations to generate the PWM gate pulses are performed in the CPU with the sampling time step of the CPU. Furthermore, the Digital Out Block just sends the PWM pulses via the FPGA to the submodule hardware. The communication between the target ports and the real hardware uses the FPGA sampling time. In the case of PWMO ports, the computation for the generation of PWM pulses takes place in the FPGA at the sampling time of the FPGA. But the input of the switching frequency and the duty cycle is inputted from the CPU.

5.3.2. Gate Pulse Comparison for 12 Submodule MMC

The 4 - submodule model was already analysed for the internal PWM generators and noticed that the phase delays were not matching well with the offline simulation. In this section, the 12 submodule MMC gate pulses were analysed in both the methods using Static Digital Out (SDO) and the PWMOut (PWMO) to understand

if the PWMO has any limitations. For the first case, N+1 Levels were observed for a 50 Hz fundamental frequency and a switching frequency (F_{sw}) of 1002 Hz. The carriers were phase-shifted in the upper and the lower arm by $2\pi/N$ (30º) respectively and no phase shift between the upper and the lower arm. As previously mentioned in the previous Section 5.1, though there is no phase shift between the upper and lower arms, there will be an obvious phase shift because the upper and lower reference waves intersect the carrier at different periods in time.

In the second case, the test was done to obtain 2N+1 levels, for the same fundamental frequency and switching frequency. The phase-shifting of carriers for the upper and lower arm remains $2\pi/N$ (30º) and the phase shift between the upper and the lower arms are considered to be π/N (15º). The input to the submodules was given using the PWMO ports and the CPU timestep was initialised to 20 µs. An input DC link voltage of 300V was applied and its corresponding output voltage was measured using the oscilloscope at no load.

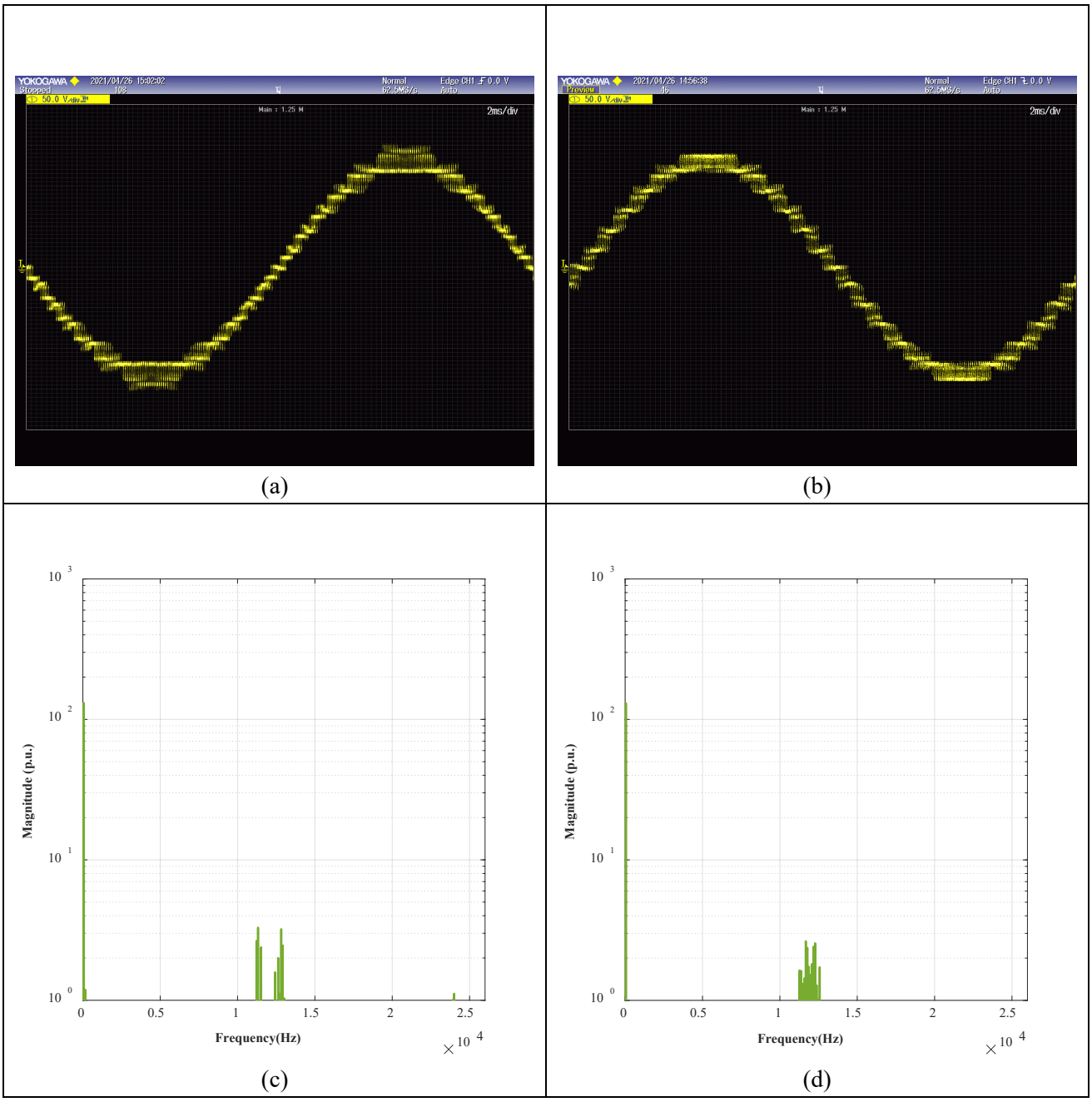


Figure 5.10: Results of 12-Submodule with 50 Hz fundamental frequency (a) Output Voltage N+1 Levels (b) Output Voltage 2N+1 Levels (c) FFT of Output Voltage N+1 Levels (d) FFT of Output Voltage 2N+1 Levels

Figure 5.10 (a) shows the output voltage waveform of one cycle for N+1 levels which shows additional, improperly built levels. They end up contributing neither to the previous level nor the next level but terminates abruptly. Also, the FFT of its output voltage in Figure 5.10 (c) shows the appearance of carrier harmonics at NF_{sw}, but the waveform doesn't convince the functionality of the PWMout. Figure 5.10(b) displaying the output voltage of 2N+1 levels, is unconvincing since it is generating only N+1 levels instead of generating 2N+1 levels. Figure 5.10 (d) supports the output voltage obtained and shows carrier harmonics at NF_{sw} while the expected result is to push the carrier harmonics to $2NF_{sw}$. To analyse better why this strange behaviour occurs three different comparisons are made in both situations. The gate pulses of the first submodule from the upper arm (UA1) and the first submodule from the lower arm (LA1) are extracted from Offline simulations, the static digital out ports (SDO) and the PWMout (PWMO) ports. Similar to what was performed in the 4-submodule model method, the difference between the midpoint of the gate pulses of UA1 and LA1 was further multiplied with the switching frequency and then by 360º to calculate the phase delay between the upper and the lower arm.

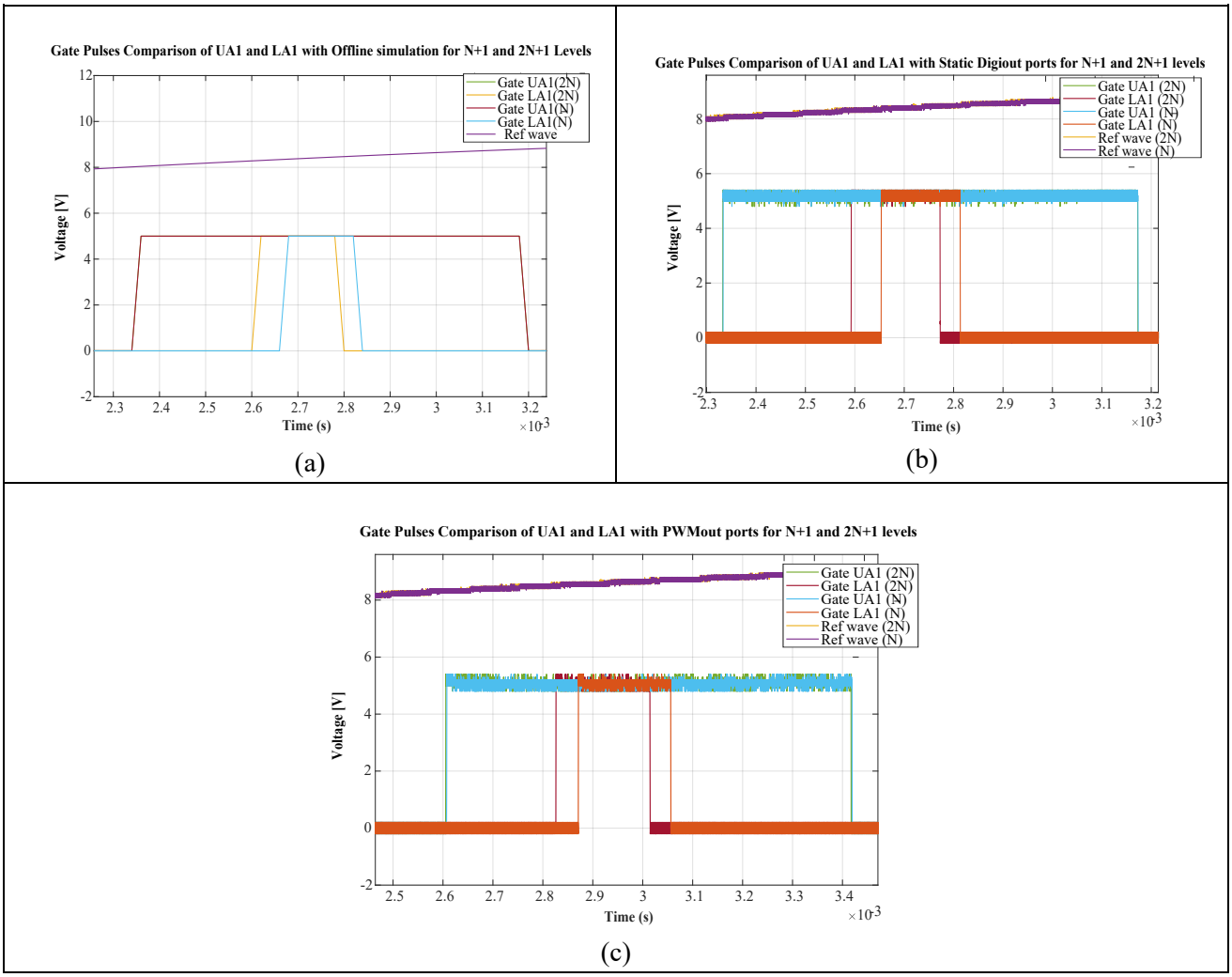


Figure 5.11: 3rd Instance in one cycle of Gate pulse comparison of UA1 and LA1 for 12-Submodule MMC at 50 Hz fundamental frequency for N+1 and 2N+1 levels (Zoomed) (a) With Offline simulation (b) With Static Digital Out Port (c) With PWMout Port

Figure 5.11 shows gate pulses of UA1 and LA1 captured for the third instance of a cycle in three different methods which is the Offline, Static Digital Out and the PWMout cases for both N+1 and 2N+1 levels. Looking closer at Figure 5.11(a) showing Offline simulation and Figure 5.11(b) with Static Digital out presents similar waveforms. The similarity can be because the sampling and computation for the generation

of PWM pulses all are performed inside the CPU itself. Hence the Static digital out can be the choice to perform simulations for a lower fundamental frequency like the 50 Hz requirement. But, a demand for a higher fundamental frequency test might not be of help using Static Digital out since the time step requirement provided by the CPU would not be sufficient to achieve accurate waveforms at higher fundamental frequencies. Figure 5.11(c) shows data from the PWMout ports and looks to have a slight variation from the offline results. This can be verified only by manually checking the phase delays at some instances of gate pulses. For this study, a single cycle of 50 Hz frequency was captured at the start of the execution in real-time and few instances from chosen to calculate the phase delays between the upper and the lower arm.

Table 5.3: Phase Delay between UA1 and LA1 for 12 Submodule N+1 levels in Offline, Real-time (SDO) and Real-time (PWMout)

	3rd Instance	8th Instance	12th Instance	16th Instance
Offline Phase Delay	7.2°	10.8°	10.8°	3.6°
Real-time (SDO) Phase Delay	7°	10.7°	10.8°	3.3°
Real-time (PWMout) Phase Delay	18°	18°	22.7°	3.8°
Difference between Offline & PWMout	10.8°	7.2°	11.9°	0.2°

Table 5.4: Phase Delay between UA1 and LA1 for 12 Submodule 2N+1 levels in Offline, Real-time (SDO) and Real-time (PWMout)

	3rd Instance	8th Instance	12th Instance	16th Instance
Offline Phase Delay	25.3°	7.21°	3.60°	18°
Real-time (SDO) Phase Delay	25°	7.4°	3.7°	18°
Real-time (PWMout) Phase Delay	32.8°	3°	7.9°	18.6°
Difference between Offline & PWMout	7.5°	4.2°	4.3º	0.6°

Table 5.3 shows the comparison of the three methods for N+1 levels in which the offline and Static digital out values coincide perfectly with each other. When comparing the findings of the offline simulation to the results of the PWMout ports, however, a bigger disparity in phase delay is observed.

Since this variance is greater than what is predicted for an N+1 level generation, the PWMOut pushes the gate signals roughly an additional 5º to 10º, to contribute to the development of additional voltage levels. Hence, the additional phase difference encountered in the PWMout for N+1 levels is giving an unexpected rise to additional levels as seen in Figure 5.10(a). However, since not all instances contribute to a larger variation from the output value as seen from Table 5.3, some levels are normally present.

Similarly, the Static digital out and the offline simulation, as shown in Table 5.4, are still accurate at the 2N+1 level. The data from PWMout still doesn't match the offline simulation perfectly. This minor change in the degree forces the gate pulses to contribute to either the preceding or next level by inserting its submodule, resulting in an abrupt increase in the magnitude of the voltage level as seen in Figure 5.12. PWMout is unable

to generate 2N+1 levels because the phase shift is not precisely set. In comparison to the output voltage waveforms from the 4-submodule, the 2N+1 levels are visible in that case since the phase shift between the upper arm and the lower arm is large (45°). Thus, though the phase shifts are not accurate which is also proved from their FFT's, the additional difference created by the PWMout did not interfere in generating the 2N+1 levels. However, this is not the case with the 12-submodule MMC because the phase shift between the upper and lower arm is very small (15°) and this makes the situation more sensitive towards expecting different results.

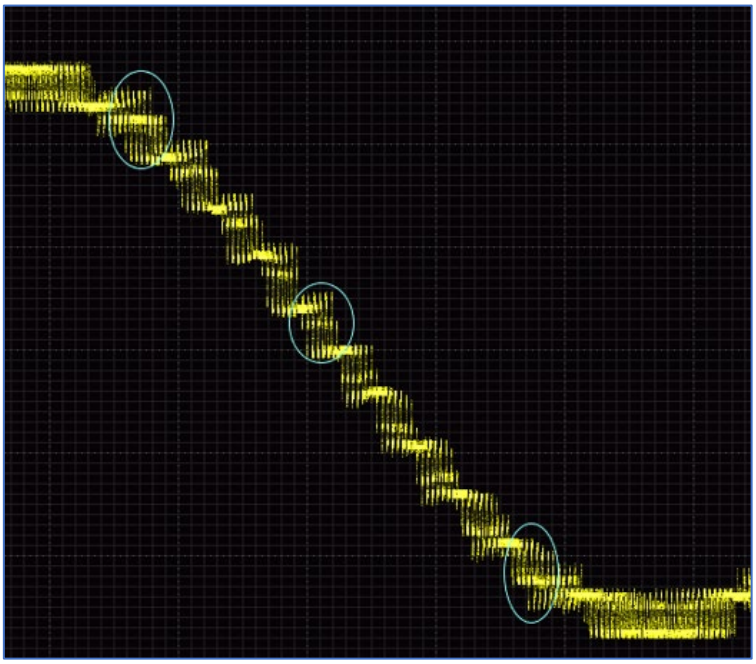


Figure 5.12: Zoomed Image of 2N+1 Levels from PWMout

5.4. Time Delays with PWMout and Static Digital Out Ports

While the gate pulses obtained were observed from the real-time and offline simulations, time delays were noted at the start of the execution of the simulation in generating PWM gate pulses. The Offline simulation runs at a discrete-time step with both control and the plant in the same model.

On the other hand, when employing Static digital out ports and running the simulation in real-time, only the control is embedded within the CPU, and the plant is external hardware. Other measurements, such as capacitor voltage values for all submodules, are obtained in the CPU in addition to the control. This necessitates an increase in the computational complexity of the CPU. However, the static digital out ports generated the PWM gate pulses ahead of the offline simulation by 127 µs for 2N+1 levels and 110 µs for N+1 levels respectively because the CPU has to deal with the complexity of just the control. While the offline simulation includes the plant model also which results in increasing the computational severity of the system. To see if the model's complexity affected the time delay in generating the gate pulses, the model's other measurements and inputs were deleted, leaving only the carrier wave comparison with the reference wave for the calculation to generate the gate pulses. This indicated that the time delay had decreased dramatically to roughly 20 µs implying that model complexity was a contributing factor in the time delays. This 20 µs could be the contribution that happens over the communication PCIe link through which the data is transferred from the CPU to the FPGA and the gate pulses are tapped in the oscilloscope.

The PWMout was also noticed to be starting to generate gate pulses with a time delay when compared to the offline simulations. The PWMout pulses were delayed by around 190 µs from the offline pulses when all the measurement blocks were retained to be computed. A simpler model was also built to analyse the time delay generated without many complex computations. But, this time though the model was simple the time delay was large by a value of 250 µs lagging the offline simulation. The plausible reasons for the time delay were suspected as follows. Firstly, the reference wave input has a variable duty cycle, the modulating signal is still sampled at the CPU sampling time (i.e. 20 µs). This sampled data is sent to the FPGA once every CPU time step, and hence this can contribute to time delays and inaccuracies. If the FPGA time step is considered to be 20 ns and the CPU time step as 20 µs, the data delivered to the FPGA is almost 2000 samples after the last data was sent to the FPGA. Because the PWM pulses are generated in the FPGA, the digital control delay plays a role in ensuring that the PWM signal is generated in the following time step after interpolation. The cumulative delays in this situation are high because the modulation signal is inputted from the CPU with a lag, followed by computation, and then PWM generation. As a result, it was interpreted that the internal PWM generators could not achieve the control accuracy expected from an FPGA implementation.

6. Implementation of Closed-Loop with PR controller

The open-loop control implemented and analysed in the previous chapters is suitable for the choice of HV test setup. In light of this, the output voltage should be able to be regulated in response to changes in the testing object. However, with open-loop control, the output voltage cannot be controlled, and some steady-state error with regard to the reference signal is expected. This contributes to the system's higher harmonics, which is undesirable. The PI controllers are one of the commonly used controllers and are widely used in the design of control for many power electronics circuits because of its simplicity. In the PI controller, generally, the integrator part of the system is responsible for bringing the system to achieving zero steady-state error. However, one significant shortcoming of the PI controller is that it can only monitor a constant reference with a zero steady-state error [4]. When trying to monitor a sinusoidal reference, this creates phase and amplitude inaccuracies [4]. Thus to obtain an exact tracking of the sinusoidal reference, a PR controller implementation is considered which is capable of ensuring a zero steady-state error and aiding in the attenuation of the lower order harmonics [34]. The basic principle of the PR controller is to feed an infinite gain at a particular frequency so that a zero steady-state error can be achieved. The resonant part of this controller can be seen as the AC integrator, similar to the integrator of the PI controller [35]. Moreover, by adding the additional resonators with targeted frequencies to the system, the related harmonics can be reduced consequently. However, this harmonic compensation can only be done up to a limited number of lower-order harmonics before the system becomes unstable [36] [37]. The design of the PR controller and the plant was first considered in the continuous domain and further converted to the discrete-time domain suitable for the implementation in the real-time simulator.

6.1. Design of Control Loop in Continuous-time equivalent

6.1.1. PR Controller Design

An ideal representation of the PR controller with an infinite gain is defined as shown in equation (6.1). However, having an infinite gain may result in stability issues due to the difficulties in designing the gain parameters [34], [38] and hence the non-ideal representation is used for further computations as shown in equation (6.2).

$$G_c(s) = \sum_{h=1,3,5,7\ldots} K_p + \frac{K_h s}{s^2 + (h\omega_1)^2} \tag{6.1}$$

$$G_c(s) = \sum_{h=1,3,5,7\ldots} K_p + K_h \frac{2\omega_c s}{s^2 + 2\omega_c s + (h\omega_1)^2} \tag{6.2}$$

Here K_p corresponds to the Proportional gain, K_h and $h\omega_1$ (i.e. $\omega_1 = 2\pi$* fundamental frequency) represent the resonant gain and the resonant frequency of the h-order harmonic to be compensated. The ω_c is the bandwidth referring to the 3 dB cut-off frequency, which regulates the frequency fluctuation of the resonant component allowed in relation to the fundamental frequency.

To achieve closed-loop stability, it is critical to include the controller's computation delay while designing the control as a continuous-time equivalence that can be easily converted to a discrete-time counterpart [39]. The PWM output generated during any given sampling period is the result of a computation performed during the preceding sampling iteration. Because the modulation input is received one sample time after the computation, there is one sampling time computation delay that must be accounted for.

6.1.2. Design of Plant

The output current circuit in Figure 2.3(a) is used to deduce the transfer function to represent the filter plant in the s-domain. The series LC (L_a and C_{load}) in the circuit causes resonance and the passive damping also exists in the circuit since the arm resistor (R_a) is in series with the LC circuit. The circuit models can be represented by the input voltage (v_s) and the output voltage (V_a) respectively and the transfer function relating the two variables is derived by combining equations (6.3) and (6.4) in the Laplace domain. The block diagram of the control method as a continuous-time equivalent is depicted in Figure 6.1.

$$V_s(s) = \frac{L_a}{2} sI(s) + \frac{R_a}{2} I(s) + \frac{1}{C_{load}} \frac{I(s)}{s} \tag{6.3}$$

$$V_a(s) = \frac{1}{C_{load}} \frac{I(s)}{s} \tag{6.4}$$

$$G_p(s) = \frac{V_a(s)}{V_s(s)} = \frac{2}{L_a C_{load} s^2 + R_a C_{load} s + 2} \tag{6.5}$$

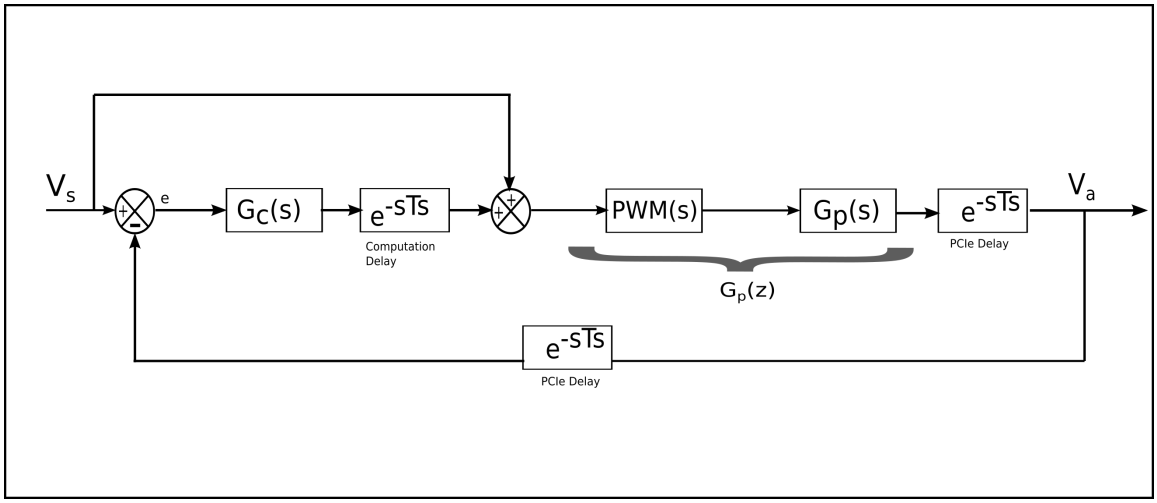


Figure 6.1: Block Diagram of Control Loop as a Continuous-time equivalent

The implementation of this control also shows that the PWM process contributes to some necessary delay which should be included in the block diagram. In the continuous domain, the PWM process is an analog comparator that receives a carrier wave and a modulating signal and then outputs a changing duty cycle. In the discrete domain, on the other hand, a digital comparator is considered, which is fed with a sampled input per sampling period and held constant till the update time instant [39]. This PWM process as a continuous-time equivalent can be represented in the form of $G_{PWM}(s)$ as shown in equation (6.6) [40].
This transfer function is equivalent to a sampler with a gain of 1/Ts, where Ts is the sampling period, followed by a Zero-Order Hold (ZOH) that feeds sampled input data to the comparator, which compares the sampled data to the carrier wave. In this case, the sampler is considered to be generating a multi-sampled modulating signal to the PWM where $T_s = T_{sw}/N$, N is the number of samples per switching period T_{sw}. As a result, it is clear that the larger the number of samples per switching period, the lesser time the PWM contributes to the delay response [39]. The m_{pwm} is also known as the PWM gain, which is the ratio of the modulating signal's peak magnitude to the carrier waveform's peak magnitude. The ZOH transfer function $G_{zoh}(s)$ can be represented in its continuous form as shown in equation (6.7)

$$G_{PWM}(s) = \frac{m_{pwm} G_{zoh}(s)}{T_s} \tag{6.6}$$

$$G_{zoh}(s) = \frac{1 - e^{-sT_s}}{s} \tag{6.7}$$

Further, the block diagram accounts for two other delays called the PCIe delay in the forward path and the feedback path respectively. These delays are present due to the architecture of the real-time simulator in which the CPU implementation of the closed-loop control occurs [29]. This results in one sampling time delay, to send the gate pulses to the plant and the same way the delay occurs when obtaining the output data back to the CPU from the plant to close the loop and for the controller to take the respective corrective actions.

6.2. Discretization of the Closed- Control Loop

The real-time simulator is designed in such a way that it can run on a discrete, predetermined time step. The construction of a continuous-time control structure necessitates many conversions in the system which can be implemented in a discrete domain. The sampling of the data has to be performed when necessary and above all, the sampling of data should be performed in the right place to obtain the expected performance from the system that is designed. In this case, the controller and the plant must be discretized, so that the analysis on the control performance can be as accurate as of the implementation in the simulator. Figure 6.2 shows the discretized representation of the control also called the Z-form representation of the closed-loop.

6.2.1. PR Controller Discretization

The equation (6.2) of the PR controller can be discretised using multiple methods to be converted to the Z-domain. In this thesis, a common method of discretization called the bilinear transformation or the 'Tustin' method is used to transform from the Laplace to the Z-domain[41][42]. This can be achieved by substituting the Tustin transform equivalent $s = \frac{2}{T_s}\frac{z-1}{z+1}$ into Equation (6.2).

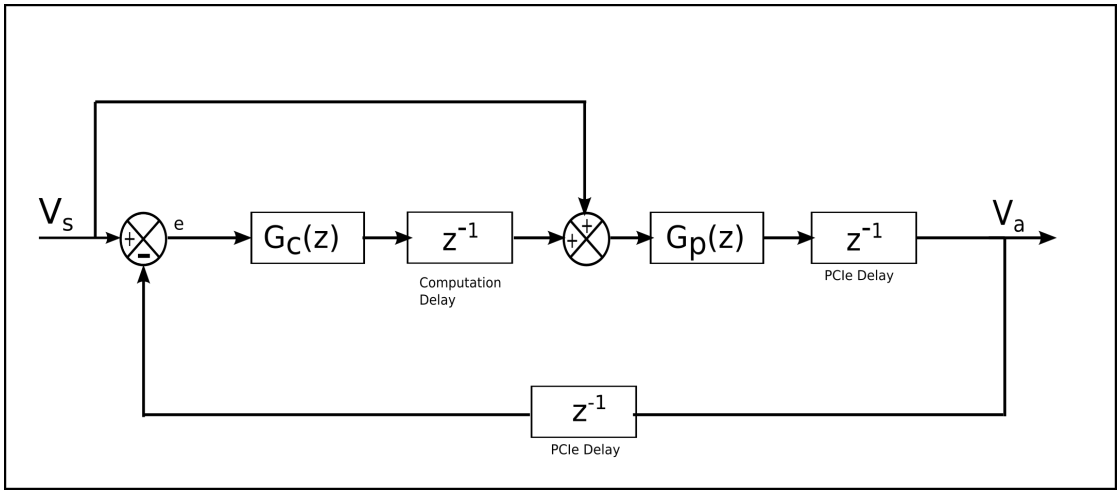


Figure 6.2: Discretised Block Diagram of the Control Loop

$$G_c(z) = \sum_{h=1,3,5,7\ldots} K_p + K_h \frac{2\omega_c\left(\frac{2}{T_s}\frac{z-1}{z+1}\right)}{\left(\frac{2}{T_s}\frac{z-1}{z+1}\right)^2 + 2\omega_c\left(\frac{2}{T_s}\frac{z-1}{z+1}\right) + (h\omega_1)^2} \tag{6.8}$$

Simplifying equation (6.8) by factorising the common denominators which are elaborately shown from Equations (6.9) - (6.11).

$$G_c(z) = \sum_{h=1,3,5,7\ldots} K_p + K_h \frac{\frac{4\omega_c}{T_s}\frac{(z-1)}{(z+1)}}{\frac{4(z-1)^2 + 4\omega_c T_s(z-1)(z+1) + T_s^2(h\omega_1)^2(z+1)^2}{T_s^2(z+1)^2}} \tag{6.9}$$

$$G_c(z) = \sum_{h=1,3,5,7\ldots} K_p + K_h \frac{4\omega_c(z-1)T_s(z+1)}{4(z-1)^2 + 4\omega_c T_s(z-1)(z+1) + T_s^2(h\omega_1)^2(z+1)^2} \tag{6.10}$$

$$G_c(z) = \sum_{h=1,3,5,7\ldots} K_p + K_h \frac{4\omega_c(z^2-1)T_s}{4(z^2-2z+1) + 4\omega_c T_s(z^2-1) + T_s^2(h\omega_1)^2(z^2+2z+1)} \tag{6.11}$$

To reduce the complexity of derivations, Equation (6.11) was rewritten as shown in Equation (6.12).

$$G_c(z) = \sum_{h=1,3,5,7\ldots} K_p + K_h G_{cr}(z) \tag{6.12}$$

Where $G_{cr}(z)$ is illustrated in Equation (6.13).

$$G_{cr}(z) = \frac{4\omega_c T_s(z^2) - 4\omega_c T_s}{z^2(4 + 4\omega_c T_s + (h\omega_1 T_s)^2) + z(-8 + 2(h\omega_1 T_s)^2) + (4 - 4\omega_c T_s + (h\omega_1 T_s)^2)} \tag{6.13}$$

$$G_c(z) = \sum_{h=1,3,5,7\ldots} K_p + K_h.\frac{a_2z^2 + a_1z + a_0}{b_2z^2 + b_1z + b_0} \tag{6.14}$$

This simplified equation (6.14) resembles the discrete transfer function whose coefficients are calculated as follows shown from equation (6.15) to (6.18). Also, it is important to notice that by substituting these values in the equation (6.14) the resonant gain K_h is multiplied by the sampling time in the numerator while the proportional gain K_p remains the same as seen in the continuous domain.

$$a_2 = 4\omega_c T_s \qquad a_1 = 0 \qquad a_0 = -4\omega_c T_s \tag{6.15}$$

$$b_2 = 4 + 4\omega_c T_s + (h\omega_1 T_s)^2 \tag{6.16}$$

$$b_1 = -8 + 2(h\omega_1 T_s)^2 \tag{6.17}$$

$$b_0 = 4 - 4\omega_c T_s + (h\omega_1 T_s)^2 \tag{6.18}$$

Further, in series with the controller, the computational delay considered should be converted into the Z-Domain where $z = e^{sT_s}$ and thus the computation delay can be represented as z^{-1}, an equivalent depiction of a single sampling time step delay in the Z-domain.

6.2.2. Discretization of the Plant

The discretization of the plant, as well as the controller, allows for the discretization of the entire system while analyzing stability. In addition, two scenarios in the real-time simulator necessitate the system's consideration in the digital realm. Firstly, while the plant is implemented in the FPGA, as in the earlier chapters' Model-in-Loop test, and secondly, when the plant is considered as physical hardware that is connected to the target via the FPGA. Though the plant model, like the controller, can be transformed into the discrete domain using a variety of techniques, the most common method for the plant is to use Zero-Order Hold (ZOH). Also, ZOH is the discretization method that resembles the practical sampled system best. As with the continuous-time equivalent, the transfer function for the PWM in Equation (6.6) shows a representation using the ZOH method. The PWM transfer function can simply be expressed as the ZOH and then which is fed to the plant using Equation (6.7) and assuming $m_{pwm} = 1$ for simplicity. However, because the $G_{PWM}(s)$ stated has an internal sampler gain of $1/T_s$, it offers a sampled output variable that will be sent to the plant. Because the plant is currently in the Laplace domain, the $G_{PWM}(s)$ function is multiplied by T_s to bring all functions into the continuous domain [39]. As a result, when the plant transfer function and the assumed ZOH are added together, the Z-transformed transfer function $G_p(z)$ is displayed in Figure 6.1. As a result, equation (6.19) can be used to illustrate this conversion.

$$G_p(z) = \mathbb{Z}\{G_{PWM}(s).T_s.G_p(s)\} \tag{6.19}$$

$$G_p(z) = \mathbb{Z}\left\{\frac{1 - e^{-sT_s}}{sT_s}.T_sG_p(s)\right\} \tag{6.20}$$

Equation (6.20) can be simplified further since $z = e^{sT_s}$ where the equation obtained is as shown in Equation (6.21).

$$G_p(z) = (1 - z^{-1})\mathbb{Z}\left\{\frac{G_p(s)}{s}\right\} \tag{6.21}$$

From the concept discussed in Section 2.3, choosing the filter values L_a and R_a are based on the number of baseband harmonics to be attenuated with 1% error and further filtering of the undesirable higher-order harmonics represented by the equations (2.15) and (2.16). With these equations, it is important to notice that since R_a acts as a passive damping resistor in series to the LC circuit, the value of R_a should be chosen in such a way that the system can be either critically damped or overdamped. The representation of the filter circuit corresponds to a second-order system which can represent a stable system based on either having equal real roots or unequal real roots [43]. The second-order system is represented in a generalised form as mentioned in equation (6.22) where ζ is the damping ratio and the ω_n is the natural damping frequency whose values based on the filter design is shown in equation (6.23) and (6.24) [43]. Considering the damping ratio for an overdamped $(\zeta > 1)$ or a critically damped $(\zeta = 1)$ the situation in equation (6.24) can be modified to calculate R_a such that it respects the equation (6.25) to ensure enough damping in the system.

$$G(s) = \frac{1}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{6.22}$$

$$\omega_n = \sqrt{\frac{L_a C_{load}}{2}} \tag{6.23}$$

$$\zeta = \frac{R_a}{2}\sqrt{\frac{C_{load}}{2L_a}} \tag{6.24}$$

$$R_a \geq 2\sqrt{\frac{2L_a}{C_{load}}} \tag{6.25}$$

For a 50 Hz fundamental frequency, choose n2 = 470 to suppress the unwanted higher-order harmonics and n1 to attenuate till the 4th harmonic resulted in more realistic values of L_a = 1.32 mH and R_a = 45 Ω for a C_{load} = 6.8 μF. The factor for guaranteeing stability based on damping was also respected in these numbers mentioned in equation (6.25). To simplify the equation in (6.21), the corresponding values of the filter design are substituted in Equation (6.5). This transfer function $G_p(z)$ can be written in the form of zeros and poles as illustrated in Equation (6.26).

$$G_p(z) = (1 - z^{-1})\mathrm{Z}\left\{\frac{1}{s}\frac{2.2282e^8}{(s + 2.528e^4)(s + 8816)}\right\} \tag{6.26}$$

Because the poles of the discrete transfer function obtained are real but unequal, the system can be predicted to have an overdamped response. In addition, partial differentiation was used to further obtain the Z-Transform on Equation (6.26).

$$\left[\frac{1}{s}\frac{2.2282e^{8}}{(s+2.528e^{4})(s+8816)}\right]=\frac{A}{s}+\frac{B}{s+2.528e^{4}}+\frac{C}{s+8816} \tag{6.27}$$

$$2.2282e^{8}=\mathrm{A}(s+2.528e^{4})(s+8816)+\mathrm{B}(s)(s+8816)+\mathrm{C}(s)(s+2.528e^{4}) \tag{6.28}$$

Solving (6.28) by substituting s = 0, s = -2.528 e^4 and s = -8816 to obtain values of A = 1, B = 0.535 and C = -1.535 respectively. Then the equation (6.27) can be re-written as equation (6.29)

$$\left[\frac{1}{s}\frac{2.2282e^{8}}{(s+2.528e^{4})(s+8816)}\right]=\frac{1}{s}+\frac{0.535}{s+2.528e^{4}}-\frac{1.535}{s+8816} \tag{6.29}$$

Lastly incorporating (6.29) in equation (6.26) and using the Z-transformation identities to convert from the Laplace domain to the Z-domain to obtain equation (6.30)

$$G_p(z)=(1-z^{-1})\left[\frac{1}{1-z^{-1}}+\frac{0.535}{1-e^{-2.528e^{4}T_s}z^{-1}}-\frac{1.535}{1-e^{-8816T_s}z^{-1}}\right] \tag{6.30}$$

Simplifying the equation by taking common terms and the $(1\text{-}z^{-1})$ terms in the numerator and the denominator cancel each other.

$$G_p(z)=\frac{z^{-1}\left(1-1.535e^{-8816T_s}+0.535e^{-2.528e^{4}T_s}\right)+z^{-2}\left(0.535e^{-8816T_s}-1.535e^{-2.528e^{4}T_s}+e^{-34096T_s}\right)}{\left(1-e^{-2.528e^{4}T_s}z^{-1}\right)\left(1-e^{-8816T_s}z^{-1}\right)} \tag{6.31}$$

Multiplying equation (6.31) with z^2 results in the equation (6.32)

$$\mathrm{G_p(z)}=\frac{\mathrm{z}\left(1-1.535\mathrm{e}^{-8816\mathrm{T_s}}+0.535\mathrm{e}^{-2.528\mathrm{e}^{4}\mathrm{T_s}}\right)+\left(0.535\mathrm{e}^{-8816\mathrm{T_s}}-1.535\mathrm{e}^{-2.528\mathrm{T_s}}+\mathrm{e}^{-34096\mathrm{T_s}}\right)}{\left(\mathrm{z}-\mathrm{e}^{-2.528\mathrm{e}^{4}\mathrm{T_s}}\right)\left(\mathrm{z}-\mathrm{e}^{-8816\mathrm{T_s}}\right)} \tag{6.32}$$

The plant transfer function in discrete form varies depending on the sample time provided to the system because this equation only contains one variable component T_s. For a set of predefined filter parameters, this discretised plant transfer function is generated. If the values of the parameters need to be altered, the same procedure can be used to recalculate the new discretised plant transfer function. In addition, the additional PCIe delays are transformed in the discrete domain, which is denoted by z^{-1} or as $\frac{1}{z}$. Thus, the whole control system after discretization can be represented as shown in the block diagram in Figure 6.2.

6.3. Stability Analysis of the Control System

Before establishing the closed-loop for a circuit, it is vital to ensure the system's stability. In the absence of stability, additional harmonics around the resonance frequency may be induced, and the system may experience many oscillations, which is undesirable. The transfer functions are used to generate the bode plots, which are used to assess the system's stability based on the Nyquist stability criteria. It is necessary

to choose a suitable gain margin and phase margin to ensure that the system can maintain stability even after closing the loop. Thus, open-loop transfer function of the entire system is used to investigate the stability of the close-loop system. The same equations hold good for the s-domain continuous equivalent of the system as well. By simplifying Figure 6.2 in the form of equations (6.33) to (6.35) for the discrete form, the loop gain transfer function in equation (6.36) may be produced from the input reference voltage to the output voltage with the feedback delay. It is worth noting that the loop gain transfer function takes the feedforward PCIe and the feedback PCIe delay into account to correctly analyse the closed loop system stability. The closed-loop transfer function of the system can be represented as shown in Equation (6.37).

$$G_c'(z) = G_c(z)\,.\,z^{-1}(\text{i. e. Computational Delay}) \tag{6.33}$$

$$\mathrm{G_p'(z)\ =\ G_p(z)\ .\ z^{-1}\ (i.e.\,PCIe\ \ Delay-\ \ Feedforward\ Path)} \tag{6.34}$$

$$G_{pFB}'(z) = G_p'(z)\,z^{-1}(\text{i. e. PCIeDelay} - \text{Feedback Path}) \tag{6.35}$$

$$G_{OL}(z) = \big(1 + G_c'(z)\big).\,G_{pFB}'(z) \tag{6.36}$$

$$G_{CL}(z) = \big(1 + G_c'(z)\big)\frac{G_p'(z)}{1 + G_{pFB}'(z)G_c'(z)} \tag{6.37}$$

To use the loop gain bode to analyze closed-loop stability, correct values of K_p, K_h and ω_c must be chosen. This is done by trial-and-error tuning, in which one of the aforementioned two parameters is held constant at first and the impact of changing the third parameter's value is examined. To get a sinusoidal signal in the high voltage test setup, the bandwidth (ω_c) of each of the resonant portion must be lesser than 1% of the rated frequency. As a result, the system is designed such that the bandwidth for each resonator varies around 0.6 rad/sec of the fundamental frequency. The chosen bandwidth of each of these resonators are shown in Table 6.1 while the changes caused by K_p and K_h are monitored for stability.

Table 6.1: Bandwidth Values of Resonators

Resonant Bandwidth (ω_c)	ω_{c3}	ω_{c5}	ω_{c7}	ω_{c9}
Value (rad/sec)	1.8	3	4.2	5.4

To understand the impact of stability owing to discretisation, the bode plots are examined for both the continuous-time equivalent and the discrete form. To analyze bode plots in the continuous domain shown in Figure 6.3 and the discrete domain shown in Figure 6.4 with a sample period of $T_s = 20\mu s$, the initial test is performed by setting the resonant gain values for h-order harmonics as represented in Table 6.2 while adjusting the K_p values.

Table 6.2: Resonant Gain Parameter Values

Resonant Gain (K_h)	K_3	K_5	K_7	K_9
Value	300	150	100	100

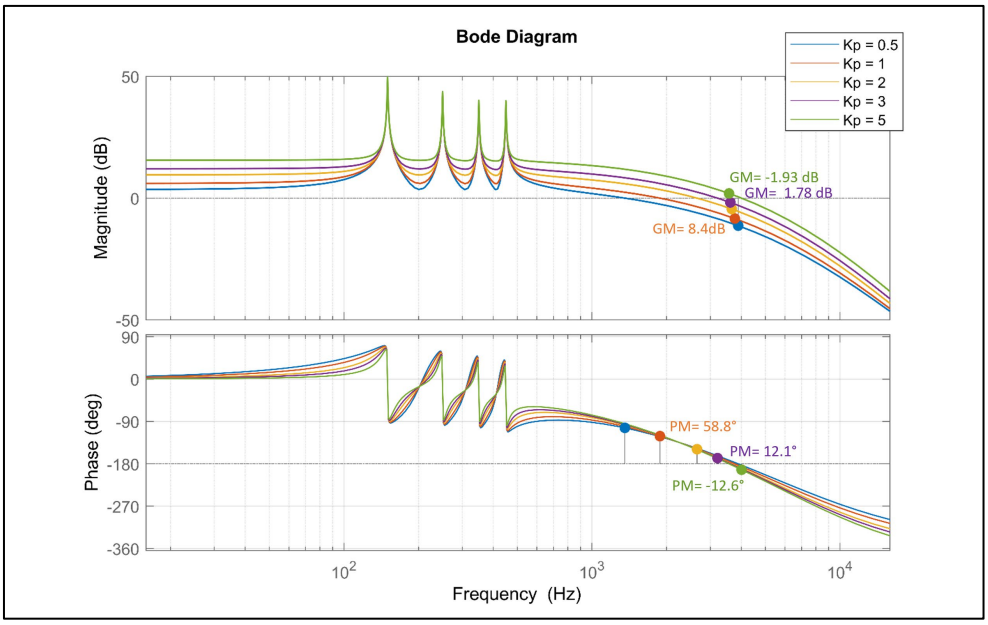


Figure 6.3: Loop gain Bode Plot with Variable Kp in Continuous Domain

The proportional control gain determines the overall system's stability and transient response. Since the resonant gains are intended to attenuate only the lower order harmonics, the system is controlled by the proportional gain during higher frequencies [44]. The cross-over frequency in a bode is the point where the magnitude plot intersects with the 0 dB line. This point is extended to the phase plot in order to estimate the phase of the cross over frequency away from -180º, resulting in the Phase Margin (PM). Similarly, the Gain Margin (GM) of the system is determined by the point where phase equals -180º meets the gain plot and the gain required from that point to reach the 0 dB. The bode plots from Figure 6.3 and Figure 6.4 shows some common observations based on the value of K_p. With a rise in the value of K_p, the magnitude response of the PR controller increases, pushing the plot upwards and away from the 0dB line. On the other hand, a very small value of K_p shall provide a stable system but this might provide a poor transient response to the system. For various values of K_p in Figure 6.3, the behaviour of continuous domain bode shows a higher value of gain and phase margin when compared to that of the discrete domain in Figure 6.4. For example, a $K_p = 3$ in the continuous domain shows that the system is stable with a positive gain margin (GM = 1.78 dB) and a phase margin (PM = 12.1º). While the same chosen K_p value in the discrete domain for a predefined time step results in an unstable system with GM = -2 dB and PM = -21.9º having negative values.

Because the real-time simulator simulates discrete-time events, implementing control in the continuous domain may not deliver the exact control performance that the system requires. Furthermore, the sample time affects the delays in the discrete-time domain, which is one of the main reasons for the differences in the margins shown in the bode plots. Because the margins from the bode plots reveal a considerable difference in executing the system in continuous and discrete domains, a discrete control system is preferable for analyzing system stability. The choice of controller parameters for closed-loop is performed using trial and error methods, and for a stable system, a phase margin of at least 45° and a gain margin from 2 -10 dB is expected to be maintained [37]. This reserve of phase margin ensures that the system remains stable even after the loop is closed due to the delay included in the feedback path and due to any effects of sampling time on the delay.

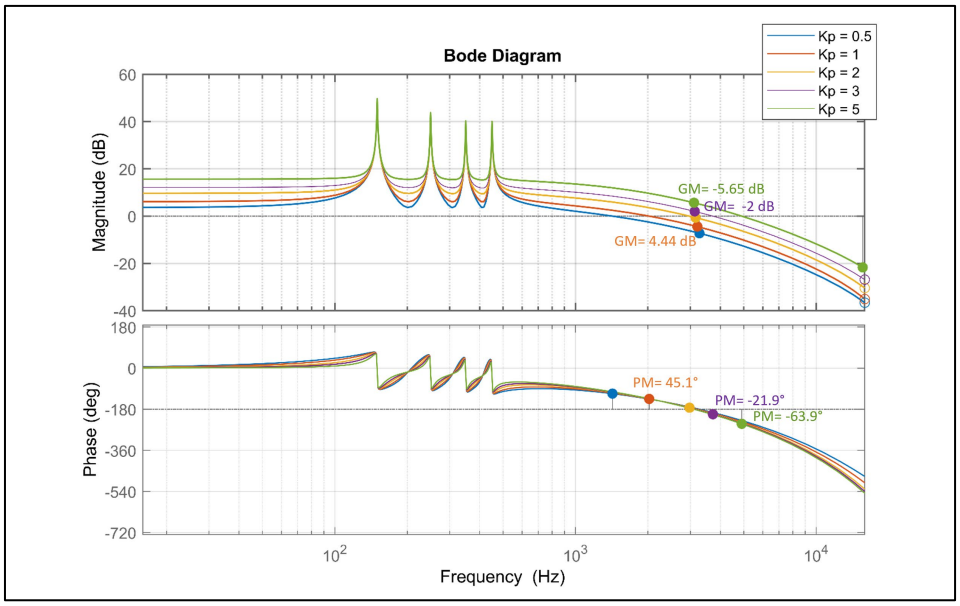


Figure 6.4: Loop-gain Bode Plot with Variable Kp in Discrete Domain

Now that, the K_p has been analysed for its impacts on stability, an appropriate K_h value must be chosen in order to achieve enough attenuation of the h-order harmonics. A bode plot is shown in Figure 6.5 to help visualize the effect of the resonant gain on stability. For a constant $K_p = 1$, it shows varying resonant gain values for the third harmonic (K_3), with the amplitude of the resonant peaks increasing as the resonant gain value rises. Higher resonant gains result in better suppression of harmonics. Variations in resonant gain do not influence overall system stability values significantly, as shown in Figure 6.5, but they do affect phase margins at the resonant frequency for which they are implemented [44].

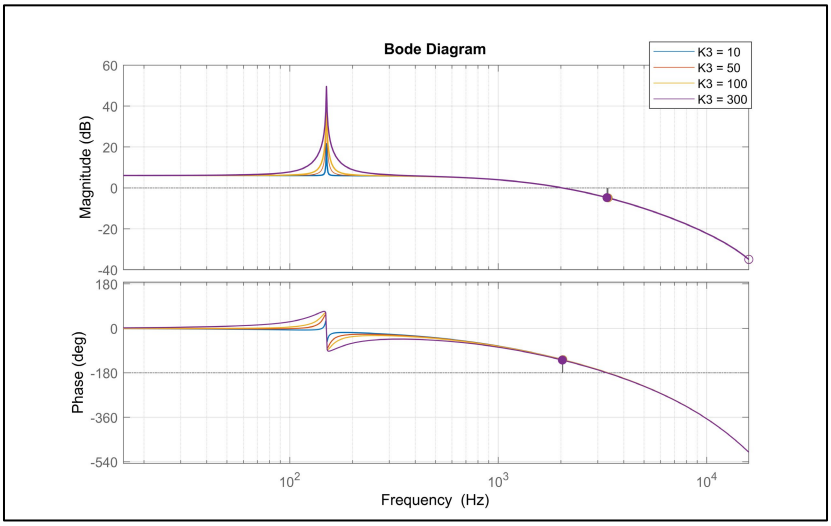


Figure 6.5: Open-Loop Bode Plot with Variable Resonant Gain for 3rd Harmonic in Discrete Domain

After analyzing the magnitude of the lower order harmonics and the stability limits using trial and error, the PR controller implementation was built to aid in attenuation of the lower order harmonics, notably the 3rd, 5th , 7th and the 9th harmonics. The discretised bode is re-evaluated for a 20 µs sampling time, and after careful examination, a K_p value of 1 is chosen for the design, which offers an adequate phase margin of 45º and a gain margin of 4.44 dB. As a result, the bode stability analysis for the loop gain transfer function is plotted in Figure 6.6 for the specified resonant gain values from Table 6.2 and bandwidth values from Table 6.1.

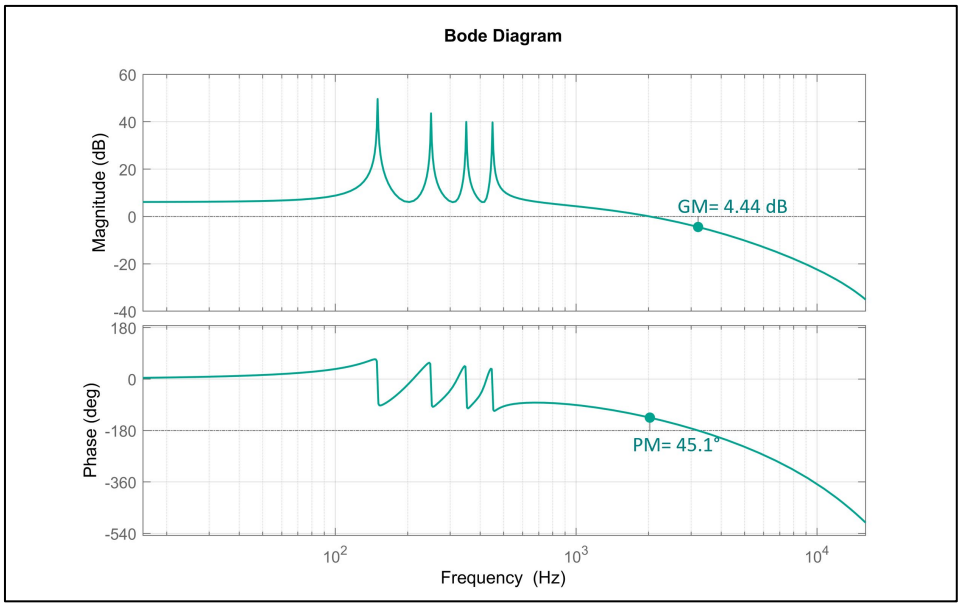


Figure 6.6: Loop gain Bode stability analysis for $K_p = 1$, $K_3 = 300$, $K_5 = 150$, $K_7 = 100$, $K_9 = 100$

6.4. Offline Simulation Verification

The implementation was first done on an offline simulation platform using MATLAB/SIMULINK, with the values checked using the bode plot. A discrete-time step was used to process the simulation, and the parameter values are given in Table 6.3 and Table 6.1.

Table 6.3: Closed-Loop Control Parameters

m_{req}	V_{dc}	T_s	C_{load}	R_a	L_a	freq	K_p	K_3	K_5	K_7	K_9
0.9	300 V	20 µs	6.8µF	45 Ω	1.32 mH	50 Hz	1	300	150	100	100

The output voltage waveform aligns with the reference signal shown in Figure 6.7 (a) with a very small steady-state error of 0.033 V when the prescribed parameters are used. By balancing the capacitor voltages, the closed-loop control enhanced the response, resulting in a better output voltage amplitude of 132.7 V. The filter parameters are already obtained based on equation (2.15) to decide the number of baseband harmonics with a 1% error. For the AWG application based on sinusoidal waveforms, the output signal tolerances with respect to the reference signal are estimated to be around 1% up to the 4th harmonic based on the chosen filter design, around 2% for the 5th and 6th harmonics, and around 4% for the higher harmonics [45] [46]. Furthermore, following correct tuning, the resonators added for the 3rd, 5th, 7th, and 9th harmonics were chosen for attenuation to make the h-order harmonics obey these requirements. For the offline results, the frequency-domain analysis is shown in Figure 6.7 (b) and was closely observed for their tolerances, which are tabulated in Table 6.4. The table indicates that all of the resonators are suppressing harmonics effectively, with errors falling within acceptable bounds. Only the fourth harmonic shows a minor divergence, but it has little impact on the system because it is still about 1%.

Table 6.4: Observed Tolerances of the Offline simulation for Closed Loop PR Control

Harmonic Order	3^{rd}	4^{th}	5^{th}	7^{th}	9^{th}
Expected Tolerance	1%	1%	2%	4%	4%
Observed Tolerance	0.98%	1.5%	0.96%	1%	0.86%

The frequency-domain analysis showed that the THD had reduced to 0.34% compared to the open-loop control simulation with THD 0.4% which was also run on the discrete domain with the same sampling time. In the case of MMC, the open-loop THD in itself is on the lower side due to the functionality that shifts the carrier harmonics to a higher frequency, based on the number of submodules and type of modulation performed on the circuit.

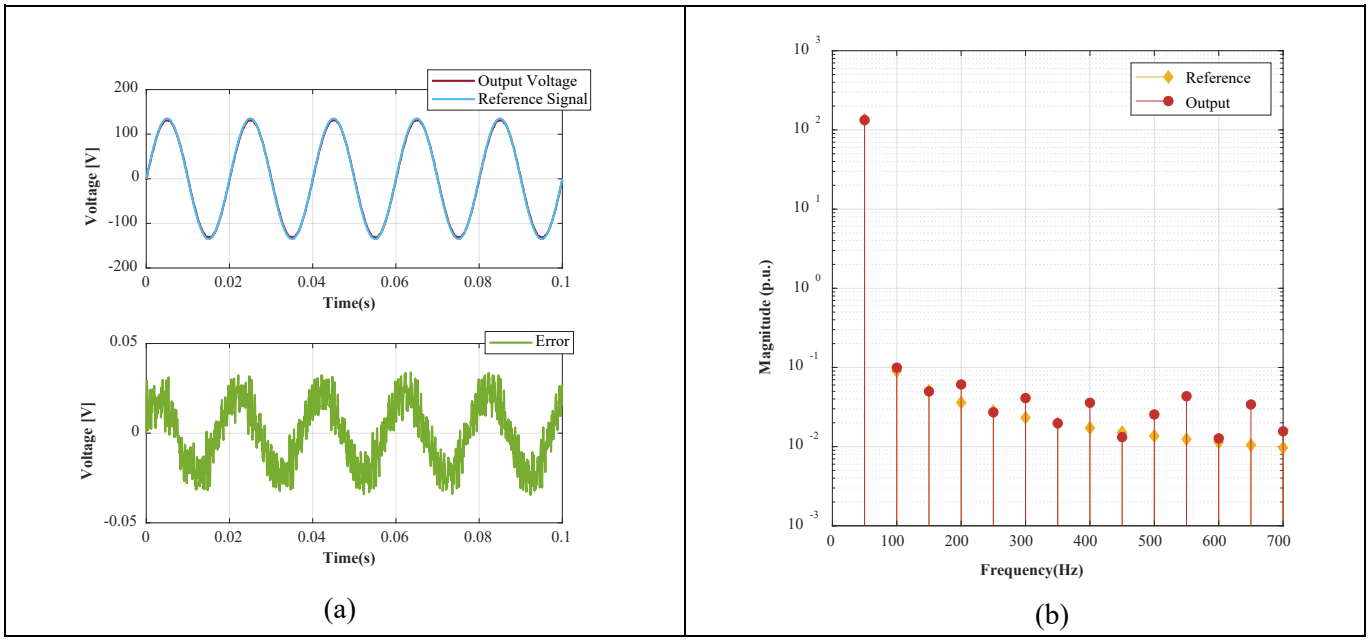


Figure 6.7: Discrete domain offline simulation results of closed-loop control (a) Output voltage and error time-domain analysis (b) Frequency domain analysis

6.5. Real-time Simulation Verification using eFPGAsim

The real-time simulation was performed using eFPGAsim with closed-loop PR control implemented in the CPU at a time-step of 20 µs and the MMC plant present in the FPGA with a sampling time of 2 µs, using the same settings as in Table 6.3. It is worth mentioning that the PCIe delays were already factored into the closed-loop stability graphs when they were developed. The first experiment was performed with a model without any RT-EVENTS blocks and the second experiment included the RT-EVENTS block set for the comparison of the carrier wave with the modulating signal.

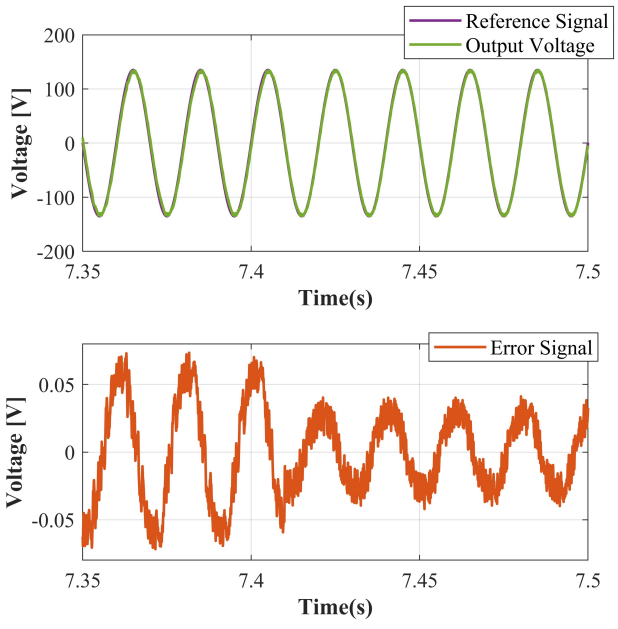
Over the first experiment, K_p and K_h were set to zero at the start of the execution and subsequently varied to the predefined values during the run duration. The resonant gain was initially kept at zero when the execution began, with just the proportional gain being enabled illustrated in Figure 6.8(a). The error between the reference signal and the output voltage has dropped from 0.075 V before activating the K_p to 0.038 V after inputting the K_p value to the controller, indicating that the proportional gain is responsible for overall system stability. The effects of the magnitude of the h-order harmonics without the implementation of PR control is visible in Table 6.5.

Table 6.5: Error Tolerances using eFPGAsim simulation without RT-EVENTS

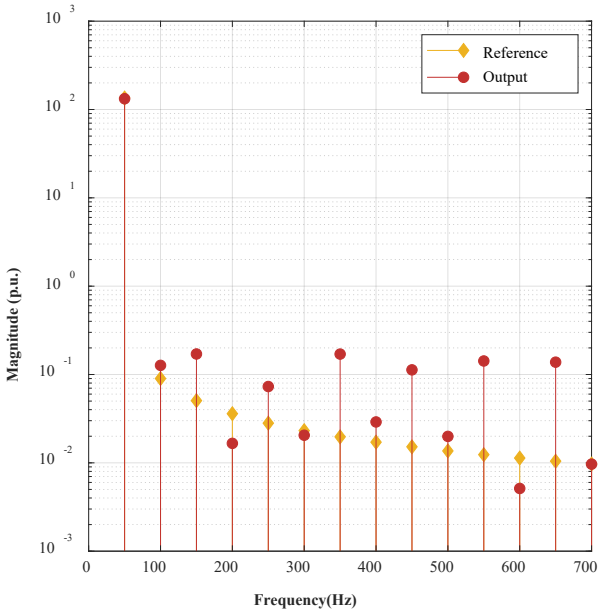
Harmonic Order	3rd	4th	5th	7th	9th
Expected Tolerance	1%	1%	2%	4%	4%
Observed Tolerance (Before Resonator Enable)	3.37%	0.46%	2.6%	8.6%	7.43%
Observed Tolerance (After Resonator Enable)	0.99%	1.21%	1.15%	1.16%	0.96%

To evaluate the impact of suppressing the lower order harmonics, the resonator was also activated by inputting all of the values of the resonant gains shown in Table 6.2. Figure 6.8 (c) illustrates that the error waveform has not altered much as a result of the resonant gain, demonstrating that the resonant gain has little effect on general system stability and instead concentrates on the resonant frequencies. Though the output voltage waveform does not reveal any significant variations from when only the proportional gain was set to after resonant gains were triggered, the frequency domain analysis presented in Figure 6.8 (d) will help to explain this. Table 6.5 shows how the resonant gains have influenced the h-order harmonics and their tolerances. Most of the measured harmonics fluctuate somewhat higher than their projected limits before the resonator is triggered. However, by activating the resonator for particular harmonics, the error magnitude was reduced, and the tolerance limitations were met. Similar to the offline simulation, though the 4th harmonic was also expected to be within 1%, it remains between 1 to 1.5% but does not majorly affect the system. The THD of this experiment is around 0.36% which is quite similar to the results obtained from the offline simulation.

The same verification was performed with RT-EVENTS blocks for the comparators used in the CPU for control which ran at a time- step of 20 µs and the plant in the FPGA at 2 µs. The K_p was initially triggered followed by the enabling of the resonators during the run time. Figure 6.9 (a) shows the results after the resonators were triggered. The frequency-domain in Figure 6.9 (b) shows that the lower order harmonics for which the PR controller was designed namely 3rd, 5th and 7th and 9th have an error tolerance of 0.98%, 0.97% and 0.93% and 1.03% respectively. The 4th harmonic has an error of 0.87% in this case, which respects the considerations taken when designing the filter parameters. The THD obtained was 0.098% and this shows that the simulation with RT-EVENTS delivers better results than that without RT-EVENTS. Because the ratio between the switching frequency and the sample time in this simulation is considerable, it meets one of RT-EVENTS' prerequisites for improved results. Second, between-sampling-time occurrences are compensated using an internal method, which eliminates the inaccuracy produced by evaluating these instants in the next sampling period.



(a)



(b)

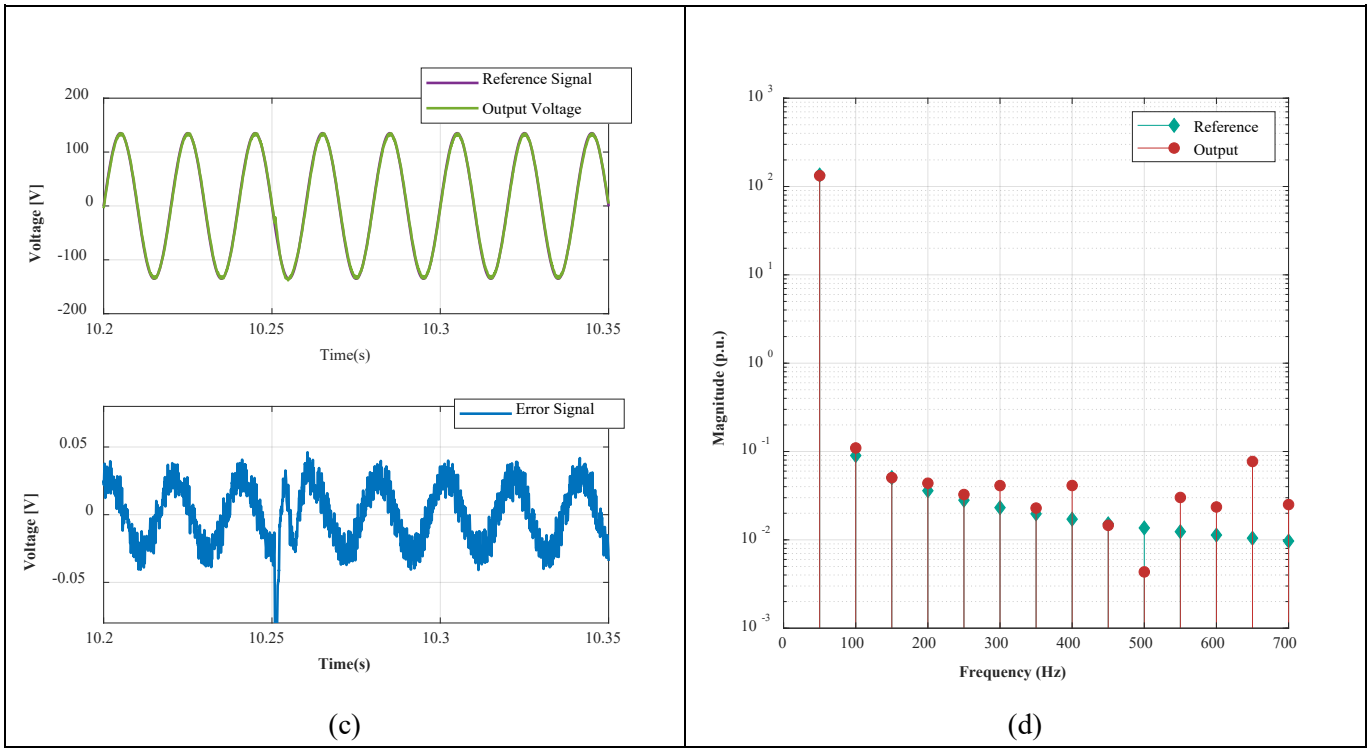


Figure 6.8: Real-time closed-loop with PR control waveforms without RT-EVENTS (a) Output voltage and error in time-domain before PR control implementation (b) Frequency domain analysis before PR control trigger (c) Output voltage and error in time-domain after resonator trigger (d) Frequency domain analysis after resonator trigger

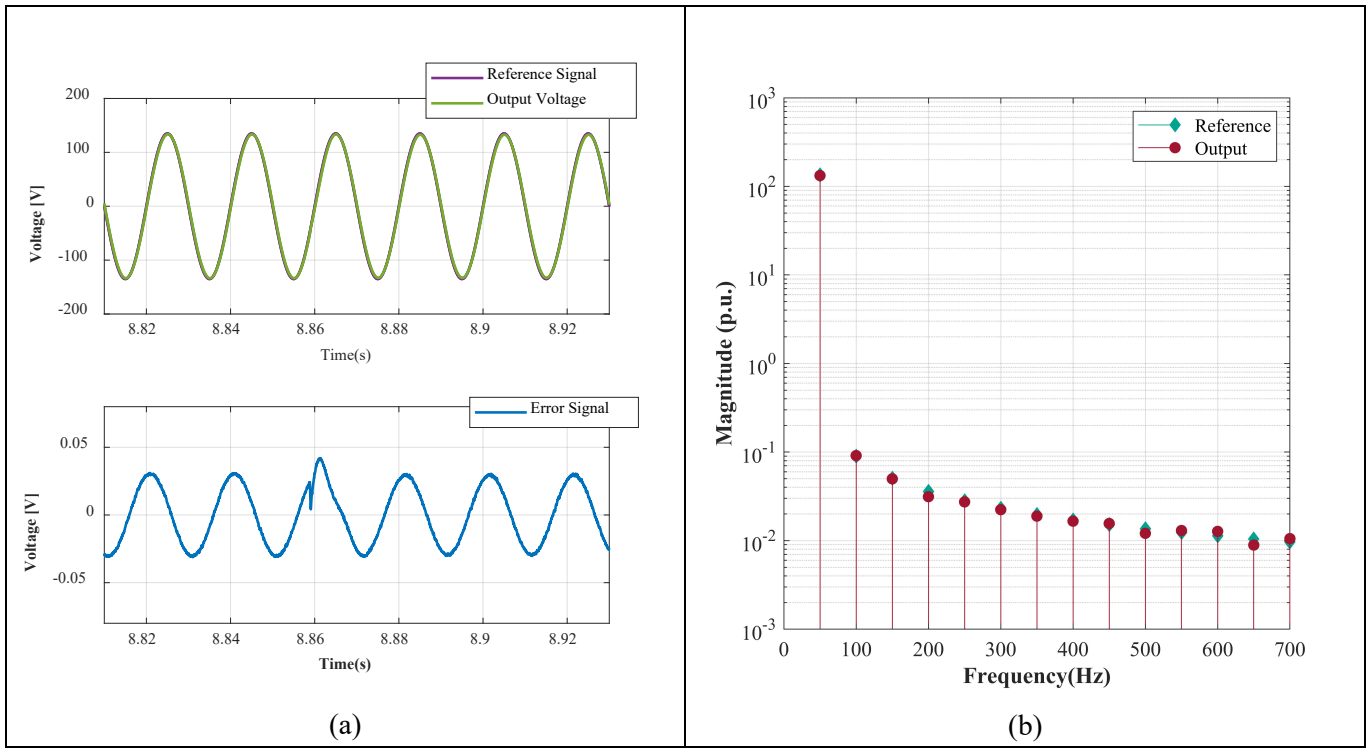


Figure 6.9: Real-time closed-loop with PR control waveforms with RT-EVENTS (a) Output voltage and error in time-domain after resonator trigger (b) Frequency domain analysis after resonator trigger

7. Conclusion and Future Work

The first most important objective of this master's thesis was met by examining the OPAL-RT's capabilities for implementing MMC control algorithms. The MMC topology was studied, and a precise approach for designing the filter was implemented for the PSC and NLC modulation techniques, with their trade-offs analyzed. Despite the fact that simulations were performed for both PSC and NLC modulation strategies, the PSC modulation technique has more advantages, as shown by the trade-offs made during the filter design. Furthermore, the OPAL-RT architecture was explored with hands-on experience of several software that may be utilized to construct control algorithms.

In a Model-in-Loop arrangement employing eFPGAsim, the MMC with a 12-submodule circuit was implemented in the target's FPGA and the control in the CPU. For the fundamental frequency of 50 Hz and 1 kHz, open-loop control for the two types of approaches such as NLC and PSC modulation was done. The capability of the control in the CPU to generate waveforms for higher fundamental frequencies was tested and analysed. A comparison was made to their offline simulation results run in the same simulation mode as real-time, at a smaller time step of 1 µs. The fundamental frequencies of 1 kHz were not achieved for two reasons. The first reason being the PCIe connection latency between the FPGA and the CPU. The second reason is the limitation of the CPU's minimum time step of 10 µs that fails the output to simulate with precision. The sampling frequency should be ten times higher than the switching frequency in order to provide precise results.

Later in the work, control was initiated from the FPGA using the OPAL- RT's internal PWM generators, and the MMC plant was modelled in the FPGA in the MIL experiment or used as real hardware in the RCP configuration. The accuracy of gate pulses generated by the internal PWM generators was observed for a 4-submodule and a 12-submodule MMC. The output waveforms were obtained for both N+1 and 2N+1 levels implemented using PSC modulation. Their accuracy was compared with the gate pulses from offline simulation and also with the gate pulses obtained using Static digital out ports of the OPAL-RT. Because the computation for generating gate pulses in both circumstances occurs in the CPU, the Static Digital out ports supplied correct gate pulses similar to the offline simulation phase delays. While the PWMout generators produced pulses with additional delays than those observed in the offline data, resulting in incomplete voltage levels. The voltage levels had discontinuities with the four submodules, even though the output waveforms indicated N+1 and 2N+1 levels due to the higher phase difference (90º) between the upper and lower arm carriers. Due to the small phase difference between the upper and lower arm carriers (15º) in the 12 submodules, the internal PWM generator induced extra delays, which had a greater impact on the voltage waveforms of the 12 submodules. Also, the sampling of the modulating signal at the CPU sampling time contributes to the time delays in this case. As a result, the internal PWM generators were unable to positively aid in obtaining accuracy through FPGA implementation.

A closed-loop PR controller for the MMC-based HV AWG was developed as part of the master thesis' second phase. The plant and controller were constructed in the continuous domain first, and then discretisation methods were employed to transfer them to the discrete domain. Because the real-time simulator treated both the controller and the plant in the discrete domain, the discretization assisted in drawing accurate stability analysis. The bode stability study resulted in the finalization of the parameter

gain values for the PR controller by guaranteeing that the system remains stable within specific margins. The results of real-time simulations with and without RT-EVENTS blocks were compared to the results of offline simulations. The open loop control ensured the system's overall stability while revealing greater magnitudes of lower-order harmonics that exceeded the tolerance limits. By enabling the PR control, these selected lower order harmonics were attenuated to the tolerance limits established for each harmonic. The practical implementation of the PR controller resulted in obtaining a close to zero steady-state error result. Also, the THD appears to match the expected offline results as well.

7.1. Future Work

The purpose of implementing the control algorithm in the CPU or the FPGA using internal PWM generators did not result in the achievement of the intended aim of gate pulse precision and the potential for higher fundamental frequencies. The research can be expanded further by implementing the control algorithm on the FPGA utilizing OPAL-RT's RT-XSG software. The implementation will eliminate the CPU interference and thus the whole control algorithm can be modelled inside the FPGA with the RT-XSG software. This will also help to improve the ratio maintained between the sampling time and switching frequency, resulting in improved results. However, there are a few things to consider while developing the controller in FPGA:

- Is one FPGA capable of handling the complexity of the entire control algorithm if the final objective is to realize a higher number of submodules?
- Can a single FPGA increase accuracy by taking into account both the computation and communication complexity of the I/Os?
- If two FPGA's are used for the design, should RT-XSG be used to create two different models for each FPGA if the computations are complex?

Appendix A : Bode Response Script

A.1. MATLAB Code to obtain the Discretized Open Loop Bode Response

```
%%-------------------------------------------------------------------
% Controller Parameters
Kp = 1;                    % Proportional Gain

Kp3 = 300;                   % Resonant Gain - 3rd harmonic
Kp5 = 150;                   % Resonant Gain - 5th harmonic
Kp7 = 100;                   % Resonant Gain - 7th harmonic
Kp9 = 100;                    % Resonant Gain - 9th harmonic

wc3 = 1.8;                     % Bandwidth
wc5 = 3;
wc7= 4.2;
wc9 = 5.4;
% Plant Parameters

La = 1.32e-3;                % Filter Inductance
Cl = 6.8e-6;                 % Filter Capacitance
Ra = 45;                     % Filter Resistance


freq = 50;                   % Fundamental Frequency
Tstep = 20e-6;               % Sampling Time
w1 = 2*pi*freq;

% Controller Discretization
G3 = tf([0 2*wc3 0],[1 2*wc3 (3*w1)^2]);
Gc3 = c2d(G3,Tstep,'tustin');

G5 = tf([0 2*wc5 0],[1 2*wc5 (5*w1)^2]);
Gc5 = c2d(G5,Tstep,'tustin');

G7 = tf([0 2*wc7 0],[1 2*wc7 (7*w1)^2]);
Gc7 = c2d(G7,Tstep,'tustin');

G9 = tf([0 2*wc9 0],[1 2*wc9 (9*w1)^2]);
Gc9 = c2d(G9,Tstep,'tustin');

Grc = (Kp3*Gc3) + (Kp5*Gc5) + (Kp7*Gc7)+ (Kp9*Gc9);

Gc = Grc+Kp;

% Computational Delay TF
comp_delay = tf(1, [1 0],Tstep);

% Plant Discretization
```

```
num=[0 2];
den=[La*Cl Ra*Cl 2];
Gs = tf(num,den);

Gz = c2d(Gs,Tstep,'zoh');

% Feedforward and Feedback Delays

PCIe_delay = tf(1, [1 0],Tstep);
PCIe_FB = tf(1, [1 0],Tstep);

% Adding Plant with the PCIe Delay
Gz = Gz*PCIe_delay;     % For Open loop Feedforward delay
Gz_d = Gz*PCIe_FB;      % For Closed Loop Feedback  delay

% Adding the Computation block with the Controller
Gc = Gc*comp_delay;

% Convert to a Zero pole model to obtain a proper bode response
Gc = zpk(Gc);
Gz = zpk(Gz);
Gz_d = zpk(Gz_d);

% Open Loop Bode Response
OL = (1+Gc)*(Gz_d);
figure('Name','Open Loop Bode- Matlab')
bode(OL)
grid on;
h = gcr;
setoptions(h,'FreqUnits','Hz');

% Closed-Loop Bode Response

CL = (1+Gc)*(Gz/(1+(Gz_d*Gc)));
figure('Name','Closed Loop Bode- Matlab')
bode(CL)
grid on;
h = gcr;
setoptions(h,'FreqUnits','Hz');
```

Appendix B : OPAL-RT Errors Debugging

B.1. Error Debugging in eHS

This appendix mostly focuses on offering some insights on how to overcome a few problems that occurred when developing the MMC model with the eFPGAsim software and practising with the RT-XSG Software. The error and warning signals are depicted in the Figures, along with the steps taken to correct the problem.

B.1.1. Error Message 1:

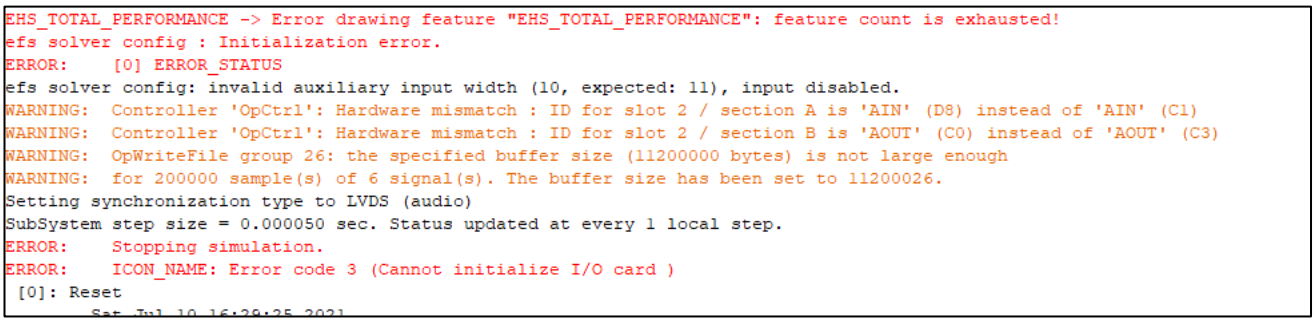


Figure B.1: Error while loading the model in RT-LAB

Error while loading an eFPGAsim based model in RT-LAB shown in Figure B.1 which mentions performance error in eHS

Solution: Open the eHS Block in the CPU model and check if the right form factor which is compatible with the user's target is chosen. In addition to this, the bin file should be chosen such that it is compatible with the form factor installed in the target.

B.1.2. Error Message 2:

Figure B.2 provides an error called ‘Netlist Parsing Error’

Solution: eFPGAsim does not allow two current measurements to be connected in series since it runs on a discrete-time domain. Hence include the snubber resistance of the switches in the model with a higher finite value rather than assigning it to infinity. Secondly, these errors also imply that the FPGA model was initially not run successfully in MATLAB/SIMULINK before giving its reference to the eHS block.

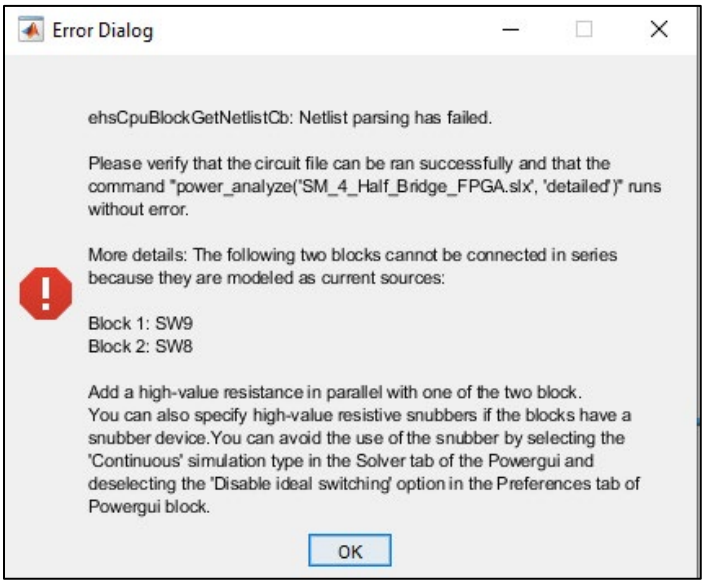


Figure B.2: Error 1 while loading FPGA model in eHS block in CPU

B.1.3. Error Message 3:

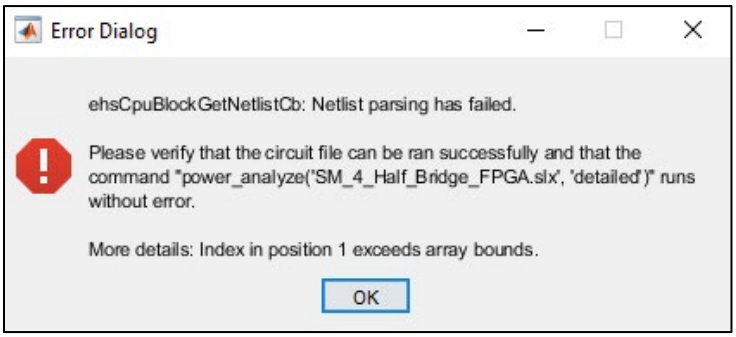


Figure B.3: Error 2 while loading FPGA model in eHS block in CPU

Another 'Netlist Parsing Error' shown in Figure B.3 mentions that Index Positions 1 exceeds array bounds

Solution: The FPGA model should be changed to a fixed-step solver with one of the numerical solving methods. Also, change the Power GUI block to discrete with the FPGA time step.

B.1.4. Error Message 4:

Error: The clock block was utilized in the Console subsystem while developing the model to operate on the real-time simulator, as shown in Figure B.4. The simulation will run without problems, however, the output will produce incorrect results when analyzed.

Solution: Include the clock block in the Master subsystem of the model to get a real-time simulation accuracy. The console is said to be used only to monitor scopes that are not real-time rather they obtain data from the Master subsystem in an asynchronous manner.

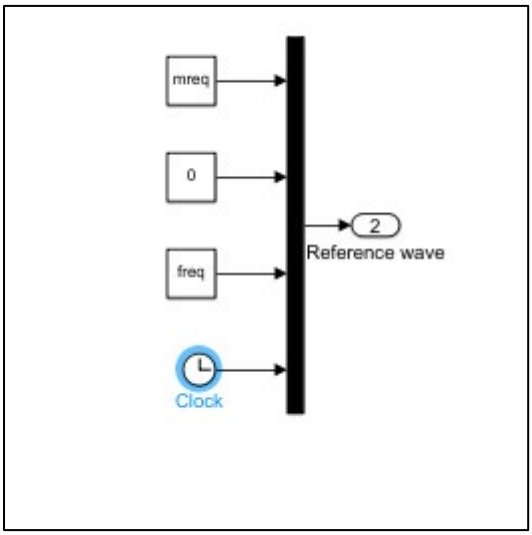


Figure B.4: Error using clock block in the Console

B.2. Error Debugging in RT-XSG to generate Bitstream file

These warnings were obtained when performing Hands-on with RT-XSG to learn the tool. These warnings occurred when an already present RT-XSG model was executed to generate a bitstream and a configuration file.

B.2.1. Warning 1:

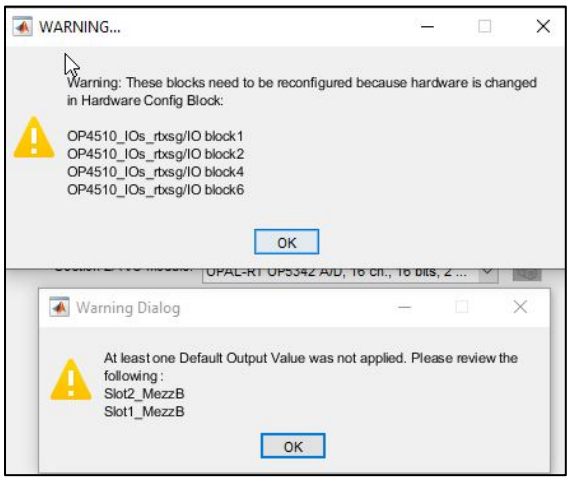


Figure B.5: Warning in RT-XSG model due to wrong I/O modules in Hardware Config Block

The Warning Dialog as shown in Figure B.5 is displayed on opening the RT-XSG Model

Solution: This warning occurs, when the Hardware configuration block includes other I/O card references that are not embedded in the target the user is using. To generate the bitstream it is necessary that this warning is cleared.

Steps :

1) Open the model example of RT-XSG given by OPAL-RT

2) Open Synthesis Manager (Block) -> Hardware Configuration. A dialog box opens as shown in Figure B.6
3) Change the I/O module and carrier type in this window based on the target the user has. The I/O modules and their configuration can be obtained from RT-LAB
   (Right Click on the target name -> Tools -> Get I/O Infos)
4) Changing the I/O modules here will automatically change the hardware interface selection of RT-XSG I/O blocks used to obtain or send data from the analog and digital ports

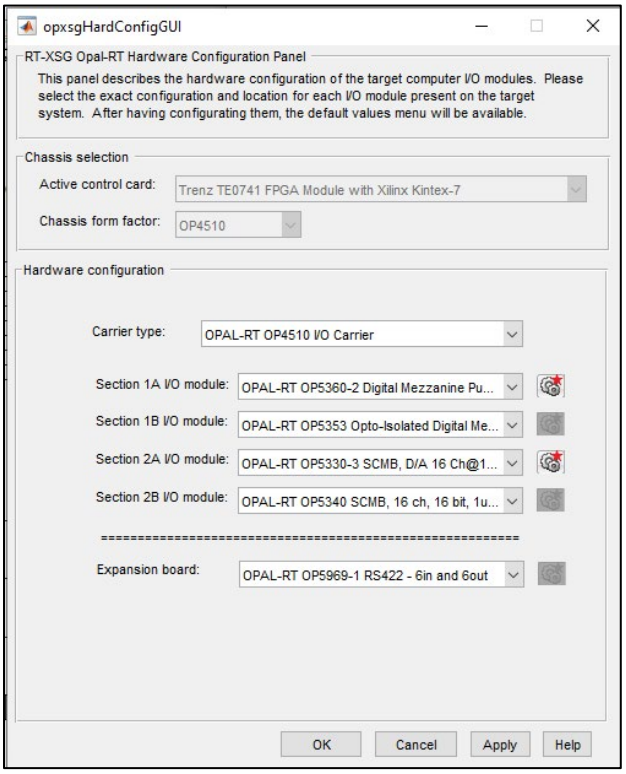


Figure B.6: Hardware Configuration Block Window

B.2.2. Warning 2:

The Warning Dialog as shown in Figure B.7 is displayed on opening the RT-XSG Model. Initially to protect all the I/O ports from sending any signals or receiving signals while designing an RT-XSG model, the default value 'Take value from RT-XSG Model' is assigned. However to generate the bitstream file this should be changed to another parameter to be compatible with the target I/O hardware.

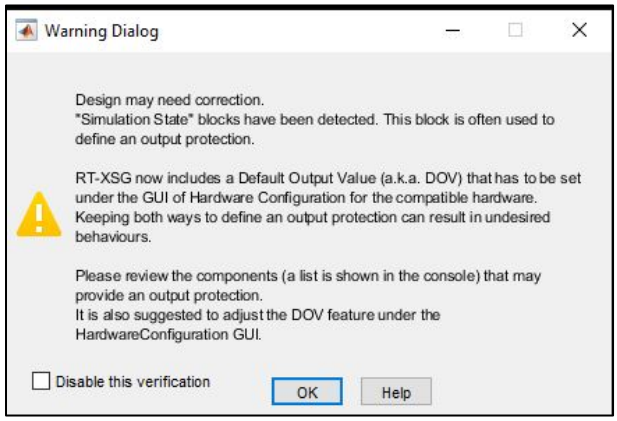


Figure B.7: Warning is shown on 'Default Output Value'

Steps:

1) Click on the settings icon seen beside the Section-wise I/O module seen in Figure B.6
2) In the apply to all channels are shown in Figure B.8, choose the ‘ Set the output Value to “Assigned Value” ’option for both the pause and reset behaviour from the drop-down lists. Assign the value to 0 for the moment.
3) Apply it to all the channels in that section.

Following these steps will allow generating the bitstream file successfully.

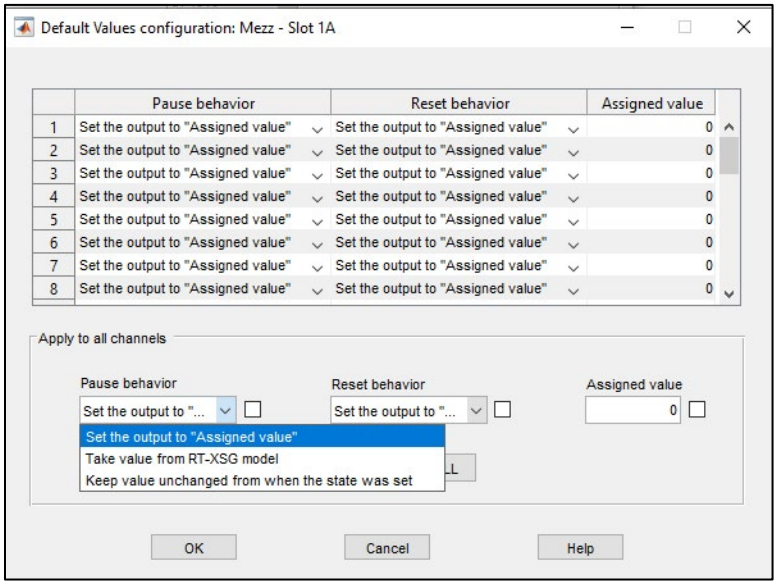


Figure B.8: Window to change Default Output Values (DOV)

Appendix C : Control Architectures

C.1. Types of Control Architectures

The control of MMC is a complex system since it must manage several objectives of the system. Managing many submodules in the MMC is what makes the control complex. The control is known to manage the following objectives

1. Monitor the output voltage and currents
2. Generate gate pulses for the MMC using the modulation techniques
3. Regulating the capacitor voltages
4. Minimising circulating currents

Thus, these objectives can be achieved through two different types of control namely, centralised, and decentralised control.

C.1.1. Centralised control

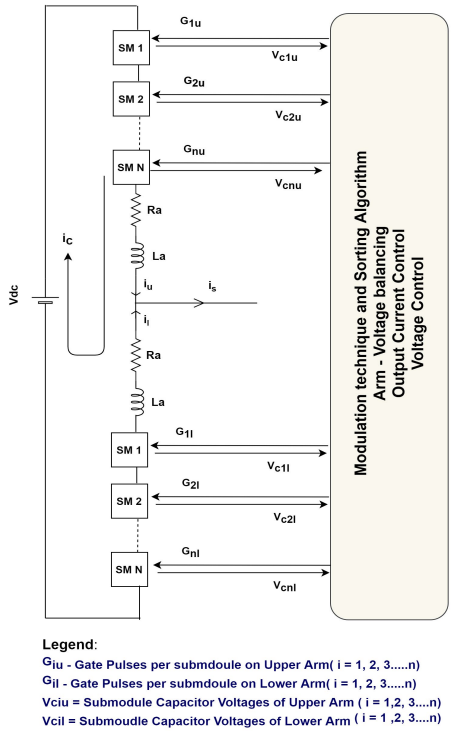
Centralised control for an MMC works in such a way that the control algorithm and modulation technique process is performed only by a single central controller as shown in Figure C.1(a). Thus, the central controller handles all the control processes of the MMC namely- the modulation algorithm to provide gating signals to the submodules, monitoring the voltage and current of the circuit and balancing of the submodule capacitor voltages as a single unit. The centralised control can be a suitable solution for a small number of submodules since it does not require a very complex control for the system and the computation bandwidth necessary is also sufficient. However, this type of control demands a huge communication and computation burden on the controller in terms of the processing power and execution for a larger number of submodules [47].

C.1.2. Distributed Control

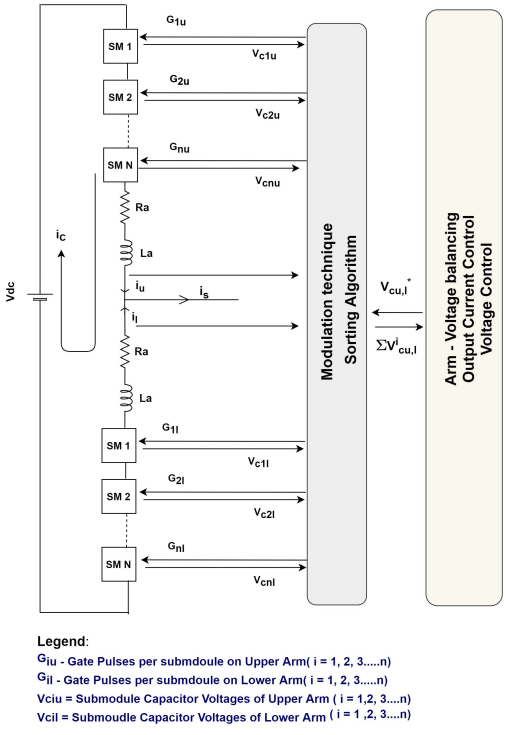
The decentralised control reduces the computation load of the central controller and allows for division of work. This minimises the need for a complicated communication strength for managing the control of the MMC, even with many submodules built in the circuit. In this case of control strategy, work is divided among many different controllers, where there exists a master and a local controller. It depends on the designer to assign which part of data can be sent and shared, to and by the master controller, respectively. The type of communication to be established between the master and the local controller can also vary based on the application of the MMC. The type of communication chosen decides the bandwidth of communication between the master and the slave and the speed at which data can be shared between the two controllers. It is worth noting that, there are different types of local controllers possible with distributed control.

1) Leg Level
2) Arm Level

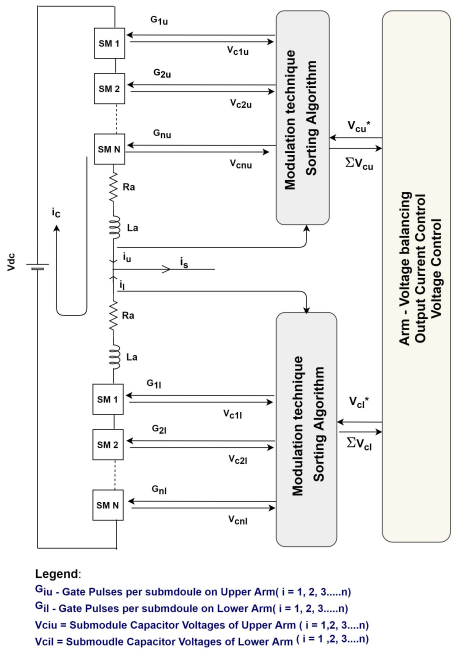
3) Submodule Level



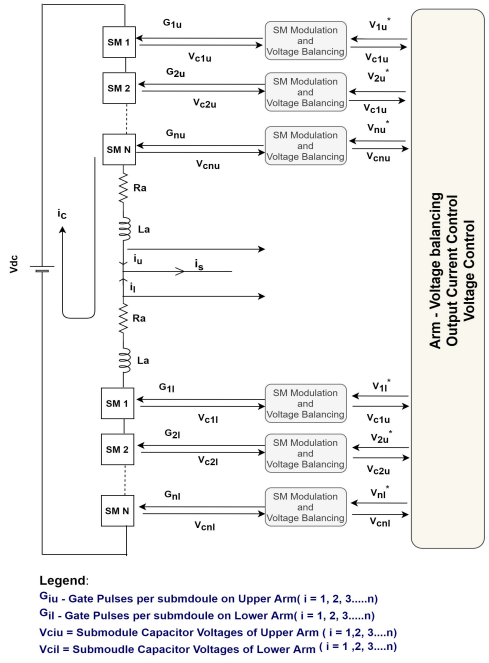
(a)



(b)



(c)



(d)

Figure C.1: (a) Centralised Control (b) Distributed Leg-Control (c) Distributed Arm-Control (d) Distributed Submodule-Control

The leg level control is a solution while an MMC has n phases, and each phase corresponds to one leg. With this control, there exists a central controller and a leg controller for every phase based on the

application. The leg controller receives individual capacitor voltage references for the upper and the lower arm from the main controller and is responsible for balancing these submodule capacitors as shown in Figure C.1(b). Along, with this functionality, it manages the gate pulse generation for the submodules using the modulation technique as well for both the arms. The main controller receives information about the sum of the capacitor voltages after being calculated by the local leg controller [48].

The arm level control is another breakdown of the control system of the MMC where the upper arm and lower arm have their local controllers shown in Figure C.1(c). These slave controllers are linked to a central controller using high-speed communication. The responsibility that the central controller takes, and the slaves take depends on the designer and the speed of the processing device used for control. A simple method of arm-based control is similar to leg control, with submodule capacitor voltage balancing and pulse generation is dedicated to the individual arm, local controllers. Breaking down functions between different local controllers reduces execution complexity and improves the processing speed of the system.

A submodule level control localises the complete control even further ensuring each submodule has its local control for itself as displayed in Figure C.1 (d). This also implies that the local control for every submodule will perform modulation to generate gate pulses and balancing of the submodule capacitance internally. However, the reference for modulation is communicated to the local submodule control by the central controller. Similarly, the individual capacitor voltage values are signalled back to the central controller by the local control, while the sum of capacitor voltages are calculated centrally.

References

[1] D. A. Ganeshpure, T. B. Soeiro, M. G. Niasar, P. Vaessen, and P. Bauer, "Modular Multilevel Converter-based Arbitrary Wave shape Generator used for High Voltage Testing," *Proc. - 2021 IEEE 19th Int. Power Electron. Motion Control Conf. PEMC 2021*, pp. 124–131, Apr. 2021, doi: 10.1109/PEMC48073.2021.9432544.

[2] D. A. Ganeshpure, "Modular Multilevel Converter (MMC) based High Voltage Test Source Demonstration of a proof of concept with a mathematical model and simulation study for arbitrary wave shapes generation," Delft University of Technology, 2018.

[3] S. Purkait and R. Pandey, "Modular Multilevel Converters an Overview : Basic Concepts , Design and Control along with Applications," vol. 5, no. 7, pp. 835–843, 2018.

[4] K. Sharifabadi, L. Harnefors, H. P. Nee, S. Norrga, and R. Teodorescu, *Design, control and application of modular multilevel converters for HVDC transmission systems*. .

[5] K. Ilves, L. Harnefors, S. Norrga, and H. P. Nee, "Analysis and operation of modular multilevel converters with phase-shifted carrier PWM," *2013 IEEE Energy Convers. Congr. Expo. ECCE 2013*, pp. 396–403, 2013, doi: 10.1109/ECCE.2013.6646728.

[6] M. Moranchel, F. Huerta, I. Sanz, E. Bueno, and F. J. Rodríguez, "A Comparison of Modulation Techniques for Modular Multilevel Converters," *Energies 2016, Vol. 9, Page 1091*, vol. 9, no. 12, p. 1091, Dec. 2016, doi: 10.3390/EN9121091.

[7] R. Darus, "Modulation and Control of Modular Multilevel Converters," The University of New South Wales, 2015.

[8] A. Marquez, J. I. Leon, S. Vazquez, L. G. Franquelo, and M. Perez, "A comprehensive comparison of modulation methods for MMC converters," *Proc. IECON 2017 - 43rd Annu. Conf. IEEE Ind. Electron. Soc.*, vol. 2017-January, pp. 4459–4464, Dec. 2017, doi: 10.1109/IECON.2017.8216768.

[9] Y. H. Park, D. H. Kim, J. H. Kim, and B. M. Han, "A new scheme for nearest level control with average switching frequency reduction for modular multilevel converters," *J. Power Electron.*, vol. 16, no. 2, pp. 522–531, Mar. 2016, doi: 10.6113/JPE.2016.16.2.522.

[10] J. Zhang, J. Liu, J. Liu, W. Fang, J. Hou, and Y. Dong, "Modified capacitor voltage balancing sorting algorithm for modular multilevel converter," *J. Eng.*, vol. 2019, no. 16, pp. 3315–3319, Mar. 2019, doi: 10.1049/JOE.2018.8910.

[11] V. Hofmann and M. M. Bakran, "A capacitor voltage balancing algorithm for hybrid modular multilevel converters in HVDC applications," *Proc. Int. Conf. Power Electron. Drive Syst.*, vol. 2017-December, pp. 691–696, Feb. 2018, doi: 10.1109/PEDS.2017.8289176.

[12] M. M. A. M. Bayoumi, "An FPGA-Based Real-Time Simulator for the Analysis of Electromagnetic Transients in Electrical Power Systems," University of Alberta, 2010.

[13] J. Bélanger, P. Venne, and J. Paquin, "The What , Where and Why of Real-Time Simulation," 2010.

[14] L. A. Gregoire, K. Al-Haddad, and G. Nanjundaiah, "Hardware-in-the-Loop (HIL) to reduce the development cost of power electronic converters," *India Int. Conf. Power Electron. IICPE 2010*, 2011, doi: 10.1109/IICPE.2011.5728153.

[15] "OP4510 V2 - Hardware Products Documentation - Wiki OPAL-RT." https://wiki.opal-rt.com/display/HDGD/OP4510+V2 (accessed Jan. 13, 2021).

[16] OPAL_RT Academy, "OP-101 Real-Time Simulation Fundamentals with RT-LAB-Online Training." https://www.eventbrite.ca/o/opal-rt-technologies-30339399500 (accessed Sep. 08, 2020).

[17] S. Mikkili, A. K. Panda, and J. Prattipati, “Review of Real-Time Simulator and the Steps Involved for Implementation of a Model from MATLAB/SIMULINK to Real-Time,” *J. Inst. Eng. Ser. B 2014 962*, vol. 96, no. 2, pp. 179–196, Jul. 2014, doi: 10.1007/S40031-014-0128-6.

[18] C. A. Rabbath, M. Abdoune, and J. Belanger, “Effective real-time simulations of event-based systems,” *Winter Simul. Conf. Proc.*, vol. 1, pp. 232–238, 2000, doi: 10.1109/WSC.2000.899723.

[19] “RT-EVENTS- Introduction - Wiki OPAL-RT.” https://wiki.opal-rt.com/display/RD/Introduction (accessed Jan. 18, 2021).

[20] “Pragmatic Real-Time Simulation on FPGA for Modern Electronic Systems.” https://www.opal-rt.com/systems-efpgasim/ (accessed Dec. 12, 2020).

[21] OPAL-RT Academy, “Application-Oriented Course OP-205 Power Electronics Real-Time Simulation with eHS- Online Training.” https://www.eventbrite.ca/o/opal-rt-technologies-30339399500 (accessed Sep. 21, 2020).

[22] “eHS Building Models with the eHS Solver - eFPGASIM- Wiki OPAL-RT.” https://wiki.opal-rt.com/display/EDS/eHS+Building+Models+with+the+eHS+Solver (accessed Feb. 11, 2021).

[23] “FPGA Based Power Electronics Toolbox (eHS).” https://blob.opal-rt.com/medias/L00161_0483.pdf (accessed Jan. 09, 2021).

[24] A. Myaing, “FPGA-Based Real-Time Simulation of Variable Speed AC Drive,” 2010, doi: 10.7939/R3TC84.

[25] “RT-XSG 3.x Toolbox User Guide - RT-XSG Documentation - Wiki OPAL-RT.” https://wiki.opal-rt.com/display/RUH/RT-XSG+3.x+Toolbox+User+Guide (accessed Feb. 22, 2021).

[26] OPAL_RT Academy, “OP-207 Real-Time Simulation with eFPGAsim- RT-XSG I/O-Online Training.” https://www.eventbrite.ca/o/opal-rt-technologies-30339399500 (accessed Sep. 21, 2020).

[27] “DataIn - RT-XSG Documentation - Wiki OPAL-RT.” https://wiki.opal-rt.com/display/RUH/DataIn.

[28] “DataOut - RT-XSG Documentation - Wiki OPAL-RT.” https://wiki.opal-rt.com/display/RUH/DataOut (accessed Sep. 21, 2020).

[29] OPAL-RT Academy, “eHS Gen4 Wire Diagram-Online Training.” https://www.eventbrite.ca/o/opal-rt-technologies-30339399500 (accessed Sep. 21, 2020).

[30] C. Graf, J. Maas, T. Schulte, and J. Weise-Emden, “Real-time HIL-simulation of power electronics,” *IECON Proc. (Industrial Electron. Conf.*, pp. 2829–2834, 2008, doi: 10.1109/IECON.2008.4758407.

[31] M. R. Haddioui, “Control and modulation strategies for MMC - based HVDC,” Aalborg University, 2015.

[32] “Real-time HIL RCP FPGA Knowledge Base │ OPAL-RT.” https://www.opal-rt.com/support-knowledge-base/?article=AA-00697 (accessed Apr. 05, 2021).

[33] M. Longhin, “Design and construction of a lab-scaled MMC-VSC converter,” 2017.

[34] S. Salimin, A. F. M. Noor, and S. A. Jumaat, “Proportional resonant current controller strategy in inverter application,” *Int. J. Power Electron. Drive Syst.*, vol. 10, no. 4, p. 2238, Dec. 2019, doi: 10.11591/IJPEDS.V10.I4.PP2238-2244.

[35] R. Teodorescu, F. Blaabjerg, M. Liserre, and P. C. Loh, “Proportional-resonant controllers and filters for grid-connected voltage-source converters,” *IEE Proc. Electr. Power Appl.*, vol. 153, no. 5, pp. 750–762, 2006, doi: 10.1049/IP-EPA:20060008.

[36] G. Shen, X. Zhu, J. Zhang, and D. Xu, “A new feedback method for PR current control of LCL-filter-based grid-connected inverter,” *IEEE Trans. Ind. Electron.*, vol. 57, no. 6, pp. 2033–2041, Jun. 2010, doi: 10.1109/TIE.2010.2040552.

[37] M. Liserre, R. Teodorescu, and F. Blaabjerg, “Stability of photovoltaic and wind turbine grid-connected inverters for a large set of grid impedance values,” *IEEE Trans. Power Electron.*, vol. 21, no. 1, pp. 263–271, 2006, doi: 10.1109/TPEL.2005.861185.

[38] A. B. Narayana, P. Raju, and K. V. Ramana, “Proportional Resonant Controller with Resonant Harmonic Compensators for Three-Phase Multilevel Inverter,” 2018.

[39] S. Buso and P. Mattavelli, *Digital control in power electronics*. .

[40] Y. Wu, T. B. Soeiro, A. Shekhar, J. Xu, and P. Bauer, “Virtual Resistor Active Damping with Selective Harmonics Control of LCL-Filtered VSCs,” *Proc. - 2021 IEEE 19th Int. Power Electron. Motion Control Conf. PEMC 2021*, pp. 207–214, 2021, doi: 10.1109/PEMC48073.2021.9432569.

[41] H. Cha, T. K. Vu, and J. E. Kim, “Design and control of proportional-resonant controller based Photovoltaic power conditioning system,” *2009 IEEE Energy Convers. Congr. Expo. ECCE 2009*, pp. 2198–2205, 2009, doi: 10.1109/ECCE.2009.5316374.

[42] P. Chittora, A. Singh, and K. Singh, “Design, analysis, and implementation of proportional-resonant filters for shunt compensation,” *Int. Trans. Electr. Energy Syst.*, vol. 27, no. 10, p. e2388, Oct. 2017, doi: 10.1002/ETEP.2388.

[43] “MASSACHUSETTS INSTITUTE OF TECHNOLOGY DEPARTMENT OF MECHANICAL ENGINEERING 2.151 Advanced System Dynamics and Control Review of First-and Second-Order System Response 1 1 First-Order Linear System Transient Response.”

[44] D. N. Zmood and D. G. Holmes, “Stationary frame current regulation of PWM inverters with zero steady-state error,” *IEEE Trans. Power Electron.*, vol. 18, no. 3, pp. 814–822, May 2003, doi: 10.1109/TPEL.2003.810852.

[45] “IEC 61869-103 Instrument transformers – The use of instrument transformers for power quality measurement,” 2016.

[46] “IEC 61869-6 Instrument transformers - Part 6: Additional general requirements for low-power instrument transformers,” 2016.

[47] B. Xia *et al.*, “Decentralized Control Method for Modular Multilevel Converters,” *IEEE Trans. Power Electron.*, vol. 34, no. 6, pp. 5117–5130, Jun. 2019, doi: 10.1109/TPEL.2018.2866258.

[48] M. López, A. Rodríguez, E. Blanco, M. Saeed, Á. Martínez, and F. Briz, “Design and implementation of the control of an MMC-based solid state transformer,” *Proceeding - 2015 IEEE Int. Conf. Ind. Informatics, INDIN 2015*, pp. 1583–1590, Sep. 2015, doi: 10.1109/INDIN.2015.7281970.