

POLITECNICO DI TORINO

Master's Degree in Electrical Engineering



Master's Degree Thesis

**Interconnection Techniques among
Real-Time Simulators for Remote
Co-Simulation**

Supervisors

Dr. Andrea MAZZA

Prof. Enrico PONS

Prof. Ettore BOMPARD

Mr. Lorenzo SOLIDA

Candidate

Riccardo POGGIO

October 2021

Summary

The Electrical system includes a wide variety of technologies and applications. The nature of its components, its interaction with other systems and its geographical extension, make it one of the most complex existing system. Thanks to the presence of sensors, processors and actuators, this system takes on more and more form of Cyber-Physical Systems (CPSs), i.e., a system in which the layer of information and communication technologies (ICT) is closely coupled with the electrical components. The CPSs model must incorporate both the physical components (such as grids, sensors, actuators) and the control parts with the aim of being as much as possible representative of the real system and its relative dynamics.

As future Power System becomes increasingly complex and interconnected to other energy systems, a single research infrastructure may rarely provide the required test-beds to study all of them together. Therefore, the complexity of the current and future Power System requires tools suited for simulations on a large scale. To face up to the challenge posed by this new reality, in recent years researchers have increasingly resorted to the use of Real-Time "co-simulation", based on the use of multiple simulation structures that interact in Real-Time with each other through special software interfaces.

This thesis aims to define an integrated experimental activity that will be used as a launching pad for the multi-site ENET-RTLab (ENSIEL National Energy Transition Real-Time Lab), which involves Politecnico di Torino, Politecnico di Bari, Università degli Studi di Genova and JRC-Ispra. The goal is to create a communication network among the different partners in order to create a Real-Time co-simulation environment in which it is possible to share hardware and software resources.

In the first part of the thesis, some introductory aspects are described, such as the concept of Real-Time simulation, co-simulation and geographically distributed co-simulation, paying particular attention to the advantages and criticalities of these methods of analysis in Power System applications. Subsequently, the tools, made available by VILLAS, were analyzed and applied to perform some communication

tests and evaluate the delays introduced during communication and the quality of the signals exchanged.

In the second part of the thesis, a distribution network was implemented through OPAL-RT[®] hardware and software products, with the aim to carry out some co-simulations with the partners involved in the initiative. The Real-Time co-simulations performed mainly refer to the Power Hardware-in-the-Loop (PHIL) concept with the following aims: 1) verify the feasibility of a Low and Medium Voltage flexible resource dispatching and 2) verify the stability of the proposed Remote co-simulation.

In the last part of the thesis, the European HV CIGRE transmission network was implemented on the OPAL-RT[®] simulator to study the primary frequency response following an abrupt load event. The network, decoupled on several simulator computation cores, was initially analyzed locally and, subsequently, was modified to allow the insertion of the Renewable Energy Sources (RES). In particular, the HV network has been modified to emulate the decommissioning of a traditional generation power plant, replaced by RES-based power plants and storage systems.

Acknowledgements

Mi è doveroso dedicare questo spazio del mio elaborato alle persone che hanno contribuito, con il loro instancabile supporto, alla realizzazione dello stesso.

In primis, un sentito ringraziamento al mio relatore, il Dr. Andrea Mazza, per la sua immensa pazienza, per i suoi indispensabili consigli, per le conoscenze trasmesse durante tutto il percorso di stesura dell'elaborato.

Ringrazio anche ai miei correlatori, ed in particolare il Prof. Enrico Pons, per i suoi preziosi consigli e per avermi dato l'opportunità di lavorare a questo progetto.

Ringrazio i miei genitori, mamma Maria Palma e papà Giuseppe, per il loro costante sostegno ed ai loro insegnamenti senza i quali oggi non sarei ciò che sono. Senza di voi, tutto questo non sarebbe stato possibile.

Ringrazio mia nonna Pia, per l'amore e l'attenzione che mi ha saputo donare e per l'appoggio che non mi ha mai fatto mancare.

Ringrazio infine la mia fidanzata Giorgia, per avermi trasmesso amore e sostegno incondizionato anche nei momenti più bui.

Table of Contents

List of Tables	IX
List of Figures	X
Acronyms	XV
1 Introduction	1
1.1 Real Time Simulation and Power Hardware in The Loop concepts .	2
1.2 Advantages and application to Power Systems	5
1.3 Real-Time Co-Simulation	7
1.4 Geographically Distributed Real Time Co-Simulation	9
1.5 Examples of International cooperation based on remote Real-Time co-simulation	13
2 Numerical methods and construction of the Simulink model for Real-Time execution	16
2.1 One-Step methods	16
2.1.1 Euler's method	17
2.1.2 Runge-Kutta methods	19
2.2 Choice of simulation parameters	19
2.3 Preparing a model for Real-Time execution	20
3 VILLAS Framework	26
3.1 Background and Motivation	26
3.2 VILLAS node	27
3.2.1 Communication between VILLAS nodes	29
3.2.2 Node types and configuration	30
3.3 Local test of VILLAS node	32
3.4 Remote test of VILLAS node	40

4	Application of VILLAS Framework	44
4.1	Test with Savona	44
4.2	Test with Bari	49
4.3	Test with Bari and Savona	54
4.4	Tests with Genoa	61
5	High Voltage Transmission Network	66
5.1	Topology of the high voltage transmission network	67
5.2	High voltage network data	67
5.2.1	Branch data	67
5.2.2	Load data	69
5.2.3	Generator data	70
5.2.4	Shunt capacitor data	70
5.3	Implementation of the Simulink model of the European HV trans- mission network	71
5.3.1	Line model	71
5.3.2	Transformer model	74
5.3.3	Generator model	76
5.3.4	Excitation system	79
5.3.5	Steam Turbine and Governor	81
5.3.6	Load and shunt model	83
5.4	Model initialization and transmission stability concepts	84
6	Test on the HV Transmission Network	87
6.1	Decoupling of HV transmission network	87
6.2	Local test: Primary frequency response	89
6.3	Remote Co-simulation: Inertial response of the system after the insertion of RES	93
6.3.1	Introduction to the dynamics of modern electrical systems	93
6.3.2	Primary response after the introduction of RES	94
6.3.3	Remote Co-simulation: PHIL setup	100
7	Conclusion and Future works	105
A	Tests with sine wave signals	107
A.1	OPAL-RT® Asynchronous Process	107
A.2	Configuration of the VILLAS node for local tests	110
A.3	Configuration of the VILLAS node for the remote test with Bari	112
B	Tests on the MV grid	114
B.1	Configuration of the VILLAS node for the test with Savona	117
B.2	Configuration of the VILLAS node for the test with Bari and Savona	118

B.3	Configuration of the VILLAS node for the test with Genoa	120
C	Tests on the HV transmission network	122
C.1	Simulink model of the HV transmission network	122
C.2	Decoupled Simulink model of the HV transmission network	126
C.3	Configuration of the VILLAS node for the remote tests	127
	Bibliography	131

List of Tables

3.1	Configuration instructions for a node	31
3.2	Statistic on data exchange between Polito OPAL-RT and Polito VILLAS node (10,000 samples)	37
3.3	Results of delays with and without VILLAS node	43
5.1	Per unit system base values of European HV transmission network [26]	68
5.2	Line parameters of European HV transmission network [26]	68
5.3	Transformers parameters of European HV transmission network [26]	69
5.4	Load parameters [26]	69
5.5	Generators parameters of European HV transmission network [26] .	70
5.6	Reference bus (slack) of European HV transmission network [26] . .	70
5.7	Shunt capacitive compensation of European HV transmission net- work [26]	70
5.8	Line parameters of European HV transmission network and the propagation delay	73
5.9	Parameters of the simplified synchronous generator	78
5.10	Parameters of the DC1A excitation system	80
5.11	Parameters of the speed governor system for steam turbine	82
5.12	Parameters of the steam turbines	83
5.13	Load flow results	85
5.14	active power transmitted by the generator and load angle	86
6.1	Variations in the power of the generators	95

List of Figures

1.1	Example of sequential simulation [1]	2
1.2	Example of parallel simulation [1]	3
1.3	A schematic of Target and Host interconnection [2]	3
1.4	Synchronous communication [2]	4
1.5	Asynchronous communication [2]	4
1.6	Real Time simulation framework [1]	6
1.7	HIL Principle Scheme [5]	6
1.8	PHIL Principle Scheme [5]	7
1.9	Example of co-simulation [1]	8
1.10	Illustration of the concept of Geographically Distributed Real-Time Simulation [6]	9
1.11	ERIC-Lab Connections [3]	13
1.12	RT-Superlab Connections [7]	14
1.13	ETMS Connection [3]	15
2.1	Euler's method	17
2.2	RT-Lab [®] interface	21
2.3	An example of top-level Simulink model	21
2.4	An example of a Simulink model with parallel computation	22
2.5	OpComm block	23
2.6	OpAsyncRecv block	24
2.7	OpAsyncSend block	24
2.8	OpIPSocketCtrl block	25
3.1	VILLAS node interfaces [20]	27
3.2	An example of VILLAS interconnection [21]	28
3.3	UDP packet format [22]	30
3.4	Top level Simulink model	32
3.5	SM_Master Simulink scheme	33
3.6	SC_Console Simulink scheme	34
3.7	Sent and received waveform ($t_s = 50 \mu s$)	35

3.8	Sent and received waveform ($t_s = 100 \mu\text{s}$)	35
3.9	Sent and received waveform ($t_s = 200 \mu\text{s}$)	36
3.10	Sent and received waveform ($t_s = 400 \mu\text{s}$)	36
3.11	Sent and received waveform ($t_s = 1 \text{ ms}$)	37
3.12	Error amplitude according to the different sending rate. (Top-right) 50 μs ; (Top-left) 100 μs ; (Centre-right) 200 μs ; (Centre-left) 400 μs ; (Bottom-centre) 1 ms	38
3.13	Architecture of VILLAS Framework implementation	40
3.14	SM_Master Simulink model for the remote test	41
3.15	Addition of the offset to the sine wave in the remote test	42
4.1	Schematic representation of the distributed feeder in the test with Savona	45
4.2	Schematic representation of the communication infrastructure in the test with Savona	45
4.3	SM_Master for the RT co-simulation with Savona	46
4.4	Active power produced by the PV generator in Savona	46
4.5	SC_Console of the Simulink model used for the co-simulation with Savona	47
4.6	Use of the received power in the RT co-simulation with Savona . . .	48
4.7	Total active power of the two <i>Dynamic Load</i> and the voltage response in the RT co-simulation with Savona	48
4.8	Schematic representation of the distributed feeder in the test with Bari	49
4.9	Schematic representation of the communication infrastructure in the test with Bari	50
4.10	SM_Master Simulink model in the RT co-simulation with Bari . . .	51
4.11	First plot: active and reactive power received from Bari. Second plot: voltage response to load variations	52
4.12	Delay introduced by the VILLAS node	53
4.13	Schematic representation of the distributed feeder in the test with Bari and Savona	54
4.14	Schematic representation of the communication infrastructure in the test with Bari and Savona	55
4.15	The SM_Master for the test with Bari and Savona	56
4.16	Active power produced by the PV generator in Savona	57
4.17	Results of the RT co-simulation with Bari and Savona. First plot: active power absorbed by the load in Bari recorded in Bari and Turin. Second plot: Voltage response recorded in Bari and Turin . .	58
4.18	Zoom of the load variation recorded in Bari and Turin	59

4.19	Results of the RT co-simulation with Bari and Savona. First plot: Load variation measured in Bari. Second plot: voltage response measured in Turin	60
4.20	Scheme of the communication infrastructure in the tests with Genoa	61
4.21	RT co-simulation with Genoa. First plot: positive step. Second plot: negative step	62
4.22	Schematic representation of the distributed feeder in the test with Genoa	63
4.23	SM_Master in the test with Genoa	64
4.24	RT co-simulation with Genoa: active and reactive power produced by the Genoa's wind generator	65
4.25	RT co-simulation with Genoa: dynamic behavior of the phase-to-phase voltage	65
5.1	Topology of European HV transmission network [26]	67
5.2	Equivalent impedance network between two terminals [27]	72
5.3	<i>Distributed Parameters Line</i> block and required parameters	73
5.4	<i>Three-Phase Transformer Inductance MatrixType</i> block and required parameters	74
5.5	On the right the transformer connected YNd11. On the left the transformer connected YNyn0	75
5.6	Figure (a): synchronous machine equivalent d-axis circuit. Figure (b): synchronous machine equivalent q-axis circuit [28]	76
5.7	<i>Synchronous Machine pu Standard</i> and the required parameters	77
5.8	Simplified model of the Synchronous generator [30]	78
5.9	<i>Simplified Synchronous Machine pu Units</i> and the required parameters	79
5.10	Type DC1A exciters	80
5.11	Speed governor system for steam turbine	81
5.12	Steam turbine model	82
5.13	<i>Three-Phase RLC Load</i> block used to model capacitive shunts	84
5.14	Trend of the transmitted power as a function of the load angle δ [29]	86
6.1	<i>ARTEMiS Distributed Parameters Line</i> and the required parameters	88
6.2	<i>ARTEMiS guide</i>	88
6.3	Decoupled HV transmission network	89
6.4	Load event at bus 5	90
6.5	Frequencies responses	91
6.6	Frequency of the Center of Inertia	92
6.7	HV transmission network with RES penetration	95
6.8	Connection diagram	96

6.9	Load event at bus 5	97
6.10	Frequencies responses of the traditional generator	97
6.11	Frequency of the Center of Inertia	98
6.12	Active and reactive power received from Bari	99
6.13	Active and reactive power from the wind generator in Genoa	99
6.14	Active power produced by the PV generator in Savona	100
6.15	Load step variation at bus 5	101
6.16	Step variations of the Bari load	102
6.17	Variation of the active power produced by the PV generator	103
6.18	Trend frequencies of the four generators	104
6.19	Trend of f_{COI}	104
A.1	Loading of required files from RT-Lab [®]	107
A.2	Loading of required files from RT-Lab [®] for the connection with VIL- LAS node	108
A.3	Compiler command	109
B.1	Simulink model: HV/MV Transormer	114
B.2	Simulink model: MV Grid	115
B.3	Simulink model: Current and Voltage measurements	116
B.4	Simulink model: Current and Voltage measurements details	116
C.1	Bus 1 (Area 1)	123
C.2	Bus 2 (Area 1)	123
C.3	Bus 3 (Area 2)	124
C.4	Bus 4 and bus 5 (Area 2)	124
C.5	Bus 6 (Area 3)	125
C.6	Decoupled Simulink model of the HV transmission network	127

Acronyms

CPS

Cyber-Physical Systems

HW

Hardware

SW

Software

RT

Real Time

SIL

Software-In-the-Loop

HIL

Hardware-In-the-Loop

PHIL

Power Hardware-In-the-Loop

R-SIL

Remote Software-In-the-Loop

R-PHIL

Remote Power Hardware-In-the-Loop

RTS

Real Time Simulators

DRTS

Digital Real Time Simulators

GD-RTS

Geographically Distributed Real Time Simulators

VILLAS

Virtually Interconnected Laboratories for Large Systems

VPN

Virtual Private Network

UDP

User Datagram Protocol

TCP

Transmission Control Protocol

TD

Time Domain

EMT

Electro-Magnetic Transient

GUI

Graphic User Interface

CPU

Central Process Unit

DUT

Device Under Test

BMS

Battery Management System

SCADA

Supervisory Control And Data Acquisition

LAN

Local Area Network

HVDC

High Voltage Direct Current

JSON

JavaScript Object Notation

PV

Photo-Voltaic

HV

High Voltage

MV

Medium Voltage

LV

Low Voltage

EHV

Extra High Voltage

RMS

Root Mean Square

DC

Direct Current

DC1A

Direct Current 1A

COI

Centre of Inertia

RES

Renewable Energy Sources

PE

Power Electronics

ASCII

American Standard Code for Information Interchange

Chapter 1

Introduction

There are many different types of simulation tools for studying different aspects of Power Systems, from large-scale high voltage transmission systems to low-voltage distribution grids or even small controllers or devices, for a variety of applications, from transient analysis to long-term planning.

Nowadays in Power Systems, renewable energy sources (RES) as distributed generations, are increasingly penetrating, furthermore smart equipment, like smart metering infrastructures, are being utilized. The infrastructures are also growing in terms of complexity and interaction to each other, and the communication system used for information flow are also being extensively developed, as well as operational methods. Therefore the development of advanced management systems, either equipment or strategies, is being more and more important for power networks.

However, the deployment of new systems requires tests and validations before implementation: new equipment models should be tested before making any prototypes, or the physical prototypes should be replaced with virtual models in advance, and tested in a virtual environment or network model. This requires power network simulation as close as possible to the real world. New strategies (e.g. fault location algorithms, demand side management and so on) should be also verified before applying them to the real world [1]. To meet all these requirements, Real-Time simulation of Power Systems seems a promising approach..

1.1 Real Time Simulation and Power Hardware in The Loop concepts

Real-Time simulation provides a virtual environment of the systems in which new control strategies or technologies can be tested *ex-ante*, i.e., before implementing in the real world system, providing reliable real-like information on impacts and benefits. This type of approach, based on Real-Time simulation, is increasingly used, especially thanks to more and more powerful microprocessors and it has a wide range of applications, even outside the world of Power Systems. The fundamental principle on which a Real-Time simulation is based is that it reproduces the behavior of a physical system through running its computer-based model at the same rate as actual time, so that each advance in simulation time is placed to occur in synchrony with an equivalent progress in wall clock time [1].

One of the most important features of Real-Time simulation is its computation parallelism capability, in which different components of the model could be assigned to different calculation core. The most basic case refers to the *sequential simulation* in which the model to be simulated in Real-Time is run on a single computer having a single CPU, while the problem is broken into a discrete series of instructions to be executed one after another, but only one instruction may execute at any moment in time. The concept of sequential simulation is summarized in Figure 1.1 [1].

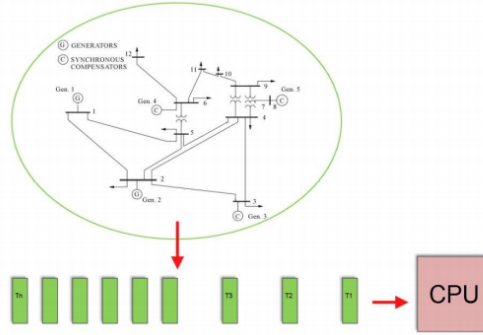


Figure 1.1: Example of sequential simulation [1]

Contrary to sequential simulation, *parallel simulation* uses more processors of the same computer with the aim of reducing the execution time of the model, and provides the ability to run larger models since more memory and more computational power is available. Parallel computing is based on the simultaneous use of multiple computing resources to solve a computational problem, running on multiple CPUs. The problem is broken into discrete parts that can be solved concurrently, and each part is further broken down to a series of instructions. The instructions from each

part then execute simultaneously on different CPUs of the same machine. The concept of parallel simulation is shown in Figure 1.2 [1].

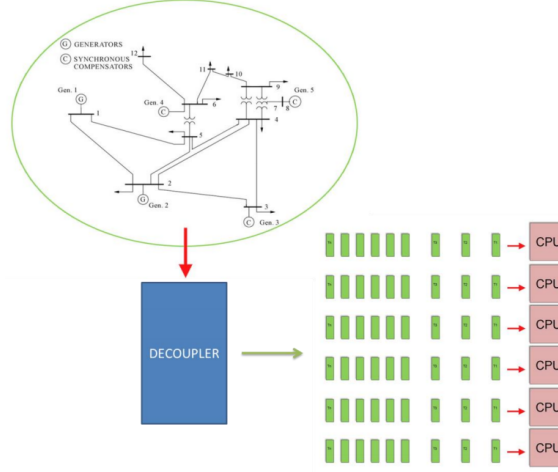


Figure 1.2: Example of parallel simulation [1]

An important topic regarding Real-Time simulation is the distinction between the machine that are called *Target* and *Host Computer*. Physically, the target represents the Real-Time simulator, while the *Host Computer* is a computer that communicates with the *Target* in order to send and receive data from it. *Target* and *Host Computer* are interfaced by dedicated communication protocols, such as UDP/IP and TCP/IP. A schematic of the interconnection between *Target* and *Host Computer* is reported in Figure 1.3 [2].

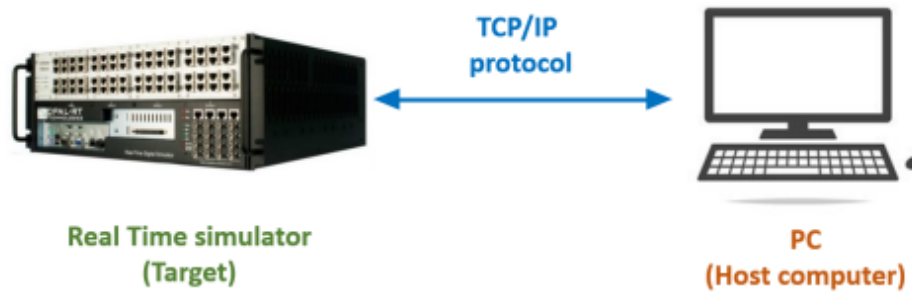


Figure 1.3: A schematic of Target and Host interconnection [2]

Finally, there are two different types of communication within a Real-Time simulation environment: *synchronous* communication (Figure 1.4) and *asynchronous* communication (Figure 1.5).

1. **synchronous** communication:

- Used for communication between CPU cores.
- Extremely high speed.
- Capable of Real-Time simulation.

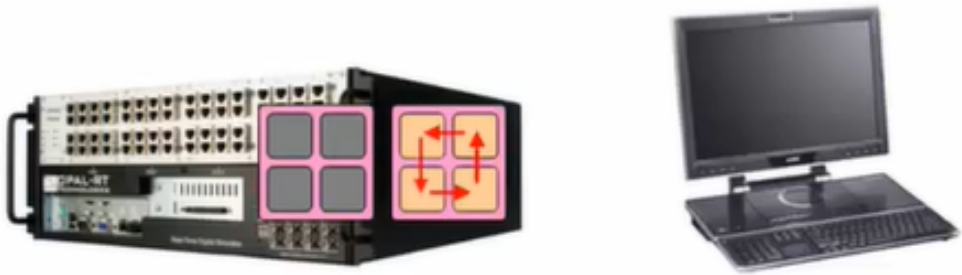


Figure 1.4: Synchronous communication [2]

2. **asynchronous** communication:

- Used for communication between *Target* and *Host*.
- Relatively low speed.
- Not capable of Real-Time simulation, used for feedback to user and data logging.

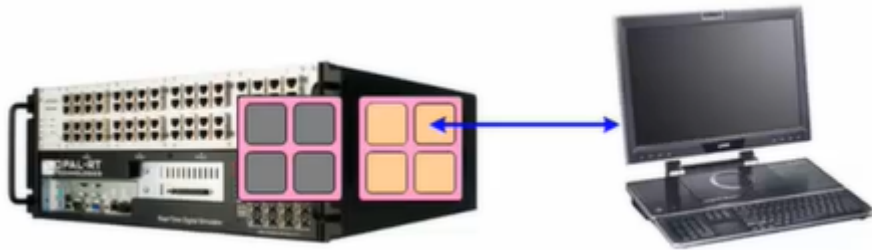


Figure 1.5: Asynchronous communication [2]

1.2 Advantages and application to Power Systems

The main advantage of Real-Time simulation is the possibility of replacing physical devices with virtual devices, which reduces costs and enables more complete and continuous testing of the entire system without interruptions. Many possible configurations without physically modifications, can be tested under possibly dangerous conditions, but safely. This advantage can be mainly gained from two simulation setups, such as Software in-the-Loop and Hardware-in-the-Loop [1]. When Real-Time tools and concepts are applied, it is necessary to define the different feasible trials, which have different objectives:

- **Software-in-the-Loop (SIL)**, where a software that works on a real component or on a control system interacts with the simulator that represents the remaining part of the system where it is necessary to test the new element.
- **Hardware-in-the-Loop (HIL)**, where physical components are also included in the simulation and the interaction between them and the software occurs through the exchange of low power signals.
- **Power Hardware-in-the-Loop (PHIL)**, which are the natural extension of HIL systems in it is not only possible to exchange signals of low power between simulator and physical components, but also electrical power signals that are able to supply a real physical device [3].

The concepts described regarding the differences between SIL, HIL and PHIL are summarized in Figure 1.6.


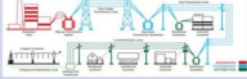






TARGET	ENVIRONMENT	Type	Abb
REAL 	REAL 	No Simulation (REAL-LIFE)	R-L
SIMULATED 	REAL 	Software In-The-Loop (testing a virtual prototype – before manufacturing or a control strategy)	SIL
REAL 	SIMULATED 	(Power) Hardware In-The-Loop (testing manufactured product or control devices)	(P)HIL
SIMULATED 	SIMULATED 	Pure Simulation	P-S

Figure 1.6: Real Time simulation framework [1]

Hardware-in-the-Loop (HIL) testing leverages Real-Time Simulation to connect real equipment and systems, through sensors and actuators, and "fool" them into thinking that they are connected to the real thing. This allows users to perform realistic closed-loop tests without the need to test in a real components. The Device Under Test (DUT) may be a controller device (e.g. SCADA, EMS, DMS, Microgrid controllers), a sensor, an intelligent electronic device (e.g. a protection device, "Smart" sensors) or a Software system. The working scheme is shown in Figure 1.7 [4].



Figure 1.7: HIL Principle Scheme [5]

While HIL typically refers to setups low-voltage level signal connections, Power Hardware-in-the-Loop (PHIL) can be employed for higher power testing. PHIL can be interpreted as an extension of HIL and it involves creating a virtual power interface between the digital simulation and DUTs. Typically, the power interface

involves power amplifiers (Voltage and/or current), which must be selected carefully depending on the application to act as a source or sink. Some examples of DUTs in a PHIL simulation are: power controllers, power converters (e.g. inverters and rectifiers), protection devices, electric machines, batteries and Battery Management System (BMS) [4]. An example of PHIL working principle is shown in Figure 1.8.

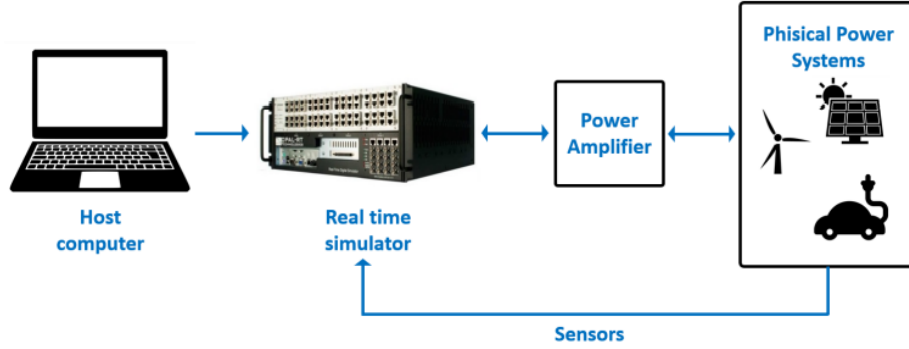


Figure 1.8: PHIL Principle Scheme [5]

1.3 Real-Time Co-Simulation

As already mentioned in the previous paragraph, one of the characteristics of Real-Time simulation is its computation parallelism capability which allows dividing the model to be simulated into different components and assign them to different computational cores. Thanks to the possibility of carrying out co-simulations, it is possible to involve a greater number of hardware and software resources existing in a laboratory.

The fundamental principle on which Real-Time co-simulation is based is that it is possible to perform simulation using computational cores of different machines located in the same area (e.g, a laboratory) which are interconnected via Local Area Network (LAN). Co-simulation can be interpreted as an evolution of parallel simulation (as it exploits the same principles of parallel simulation) but the computing cores belong to different machines and are interconnected via a LAN. An example is reported in Figure 1.9 [1].

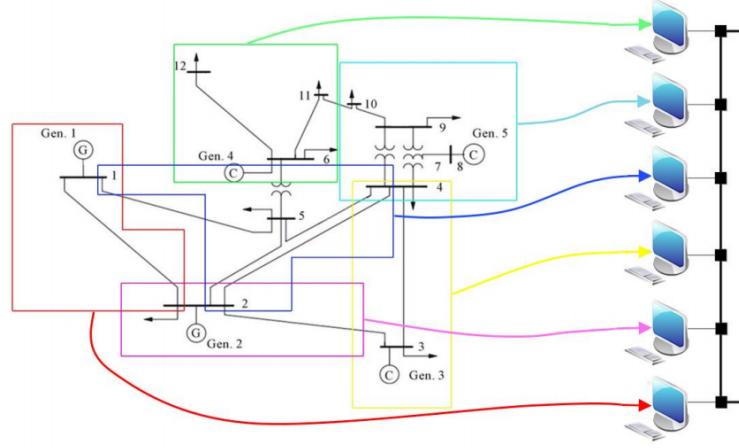


Figure 1.9: Example of co-simulation [1]

Co-simulation applies very well to studies that must take into account the mutual interaction among subsystems that operate on different hierarchical level or operational time frameworks. In the co-simulation, different subsystems that form the complete system are represented and simulated in a distributed manner. The overall simulation of the system is, in general, obtained by launching the subsystem simulations interacting with each other. The separation of the overall simulation in the different subsystems and the partition of the same between the different computing cores (belonging to different target) must be carefully treated.

During the co-simulation, the subsystems thus model exchange data between them to simulate the behavior of the whole system. In this way, each simulator synthesizes a specific element or aspect of the system, allowing a detailed analysis of the iterations and dynamics that occur between the various subsystems. In the co-simulation of very different physical phenomena (such as those existing in the electricity sector), is very important that the dynamics of the subsystems are adequately simulated. For this purpose, Real-Time simulation systems can be useful for carrying out simulations whose evolution is synchronized with the evolution times of a phenomenon [3].

The principles of Real-Time co-simulation can also be used to create distributed virtual environment in which different users or computers, located in different labs (e.g. located in different cities or countries), concurrently execute a simulation. In this case the simulation is called *Geographically Distributed Real-Time Co-Simulation* [1].

1.4 Geographically Distributed Real Time Co-Simulation

The concept of the virtual interconnection of laboratories in Real-Time is being introduced as a geographically distributed real-time co-simulation in which Real-time simulation resources, PHIL setups, novel test benches, and hybrid co-simulation frameworks are interconnected to form a comprehensive research infrastructure that allows the sharing of resources and enables large-scale and multi domain experiments.

Virtually interconnected infrastructures serve as a flexible framework for collaboration on joint experiments and studies based on indirect data sharing without revealing confidential details of individual research groups, industries, and utilities. Therefore, the collaborations and joint studies that leverage competencies across research entities are feasible even with confidentiality constraints.

Geographically distributed Real-Time co-simulation, also called Remote co-simulation, refers to the concept of partitioning a monolithic simulation model into subsystem portions that are simulated concurrently on multiple Digital Real-Time Simulators (DRTS) units located at geographically dispersed facilities. As illustrated in Figure 1.10, a monolithic simulation model is partitioned into subsystems for real-time simulation at multiple DRTS units while only the interface quantities are being exchanged between the units [6].

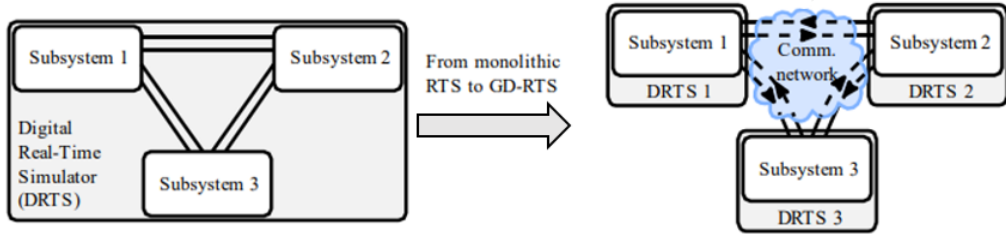


Figure 1.10: Illustration of the concept of Geographically Distributed Real-Time Simulation [6]

Interconnection and data exchange between DRTS systems is typically realized via a shared communication network, such as the Internet. Therefore, comprehensive real-world testing of interconnection between novel technologies and the existing power grid can be performed without the need for the diverse HIL or PHIL setups to

be located at the same facility [7]. This type of approach allows the implementation of Remote Power-Hardware-in-the-Loop (R-PHIL) and Remote Software-in-the-Loop (R-SIL), i.e., the implementation of hardware and software elements present in remote laboratories. In general, the R-PHIL and R-SIL implementation requires the exchange of electrical variables (such as voltage, current and power) between two laboratories through the Internet [3].

There are mainly four conditions for which the interconnection of geographically distributed Real-Time simulators can be a beneficial solution:

- **Soft-sharing of hardware (HW) and software (SW) facilities within a federation.** Utilizing the facilities of other labs if they are not locally available (without the need to move them or go where they are) can be gained as an advantage of multi-site simulation platform. It provides remote testing of devices by integrating (power) Hardware-in-the-Loop, as well as remote Software-in-the-Loop while the target model is simulated in a different lab.
- **Enhancing simulation capabilities for big systems.** If the model is too “large” to be simulated in a single local machine, it needs to be split and co-simulated in more than one machine (or lab). What makes a system “large” is not only the size of the model in terms of number of node, but also the simulation time step. Simulation time step is usually set based on the purpose of the simulation and the required time scale for the analysis. Regarding the interconnection issue, the time step should set at least equal to the round-trip communication delay. Considering this limit, in case the model cannot be run using all available calculation cores of a local machine, it is a “large” case, which needs to be split and run on more than one machine.
- **Soft-sharing of expertise in large knowledge-based virtual environment.** Different labs may need Real-Time simulations for different purposes and applications according to their research interests or available experts, while not necessarily in each single lab experts in all fields of Power Systems are working. In this case, the lab with available HW/SW equipment can take advantages from the “knowledge” or “expertise” of another distant located lab remotely in an agreed cooperation. In other words, the equipped lab may not need to host physically visiting experts or researchers for short-term projects/collaborations.
- **Keeping susceptible data/model/algorithm confidential.** If in one country some data or system models are confidential and sharing them with other labs require long authorization and administrative procedure, or even not allowed, the model can be simulated locally and exchange appropriate data or

simulation results with other interesting labs through Real-Time co-simulation [1].

In addition to the advantages mentioned above, however, there are some critical aspects that can affect the fidelity and reliability of the co-simulation.

Critical aspects of Geographically Distributed Real Time Co-Simulation

The interface quantities exchanged among DRTS units are variables defining power, typically current and voltage. The primary requirement for an accurate geographically distributed Real-Time co-simulation is conservation of energy at the interface. Energy must be neither generated nor stored at the interfaces between simulated subsystems. If conservation of energy is violated, simulation stability and fidelity cannot be ensured. It greatly depends on the partitioning point that is selected to divide the monolithic model for the purpose of distributed simulation [7].

The challenge of ensuring conservation of energy between subsystems increases with longer geographical distances. Data exchange between subsystems in geographically distributed DRTS is characterised by time-varying delay, packet loss, packet reordering and other non-deterministic characteristics of a shared communication network such as the Internet [8]. Regarding the communication delay in the Internet can be decomposed in four types:

1. *signal propagation delay*: it is the time taken for a signal to travel in the physical propagation medium, and depends on the medium itself and the distance between the nodes. Usually the propagation speed of the signal in the medium is about 70 % of the speed of light in vacuum.
2. *Network processing delay*: these delays are incurred when the network gateways, firewalls, and servers determine the path and sequence of all incoming data packets, and decision-making with any incoming packet. The delay depends on the network equipment technology and the specific processing function.
3. *Transmission delay*: it is the define time delay for a packet to be completely pushed on to the physical link layer and depends on the bandwidth of the link and packet size.
4. *Queuing delay*: this kind of delay occurs when multiple packets from an entry port are routed to the same exit port. One packet is transmitted at a time, and a queue is maintained to hold the remaining packets. The time a packet has to spend in a queue is the queuing delay seen by that packet.

The total communication latency is the sum of all the above four delays. Based on the communication network condition, the total communication latency normally ranges from several milliseconds to a few hundred millisecond [9]. Unfortunately, delays are also mainly influenced by the instantaneous network traffic, which can vary depending on the moment on which the test is run and it is difficult to assess how much traffic will impact the Real-Time remote simulation [10]. Namely, communication medium can result in a virtual energy stored or generated at the co-simulation interfaces. As a consequence, simulation stability might be lost or simulation fidelity degraded. Simulation fidelity represents a degree of similarity between simulation results obtained in geographically distributed Real-Time co-simulation and simulation results obtained in a Real-Time simulation of a monolithic model. Co-simulation interface algorithms (IA) aim at eliminating the violation of the energy conservation at the interfaces, allowing to preserving simulation stability and ensuring high-degree of simulation fidelity in geographically distributed Real-Time co-simulation [6].

Finally, it is possible to define three different models to build a geographically distributed co-simulation:

1. *Monolithic model*: Monolithic model represents a reference for characterizing the simulation fidelity. In fact, there are some critical aspect during co-simulations (i.e. delays, noise, packet losses) and monolithic model represent the reference with no time delay for the comparison of the decoupled and the distributed model.
2. *Decoupled model*: Decoupled model, performed locally between two or more simulators connected to the same LAN, represents a first step towards GD-RTS and is directly derived from the monolithic model by introducing an interface algorithm co-simulation at the selected partitioning point. This simulation framework is introduced to differentiate between fidelity degradation caused by model partitioning and communication time delay.
3. *Distributed model*: The distributed model is derived from the decoupled model by moving one subsystem entirely to a separate RTDS simulation. In this case all the critical aspects that characterize distributed simulation exist [11].

By comparing the monolithic model and the distributed one, in which the interfaces are interconnected to each other through simplified models, it is possible to verify the delays introduced by the interconnection network and the number of packets lost during the co-simulation.

The concept implemented in ERIC-Lab, previously limited to the European borders, was then generalized to create a transatlantic co-simulation in September 2017. In fact, the experience called RT-SuperLab saw the integration of some ERIC-Lab partners (PoliTo and RWTH Aachen) with six US partners, DOE and University (Sandia National Laboratory - NREL, Colorado State University, Idaho National Laboratory, University of South Carolina, Washington State University) with the aim of experimenting a potential link in HVDC between United States and Europe through a transatlantic Real-Time co-simulation between laboratories European and North American (Figure 1.12) [3].

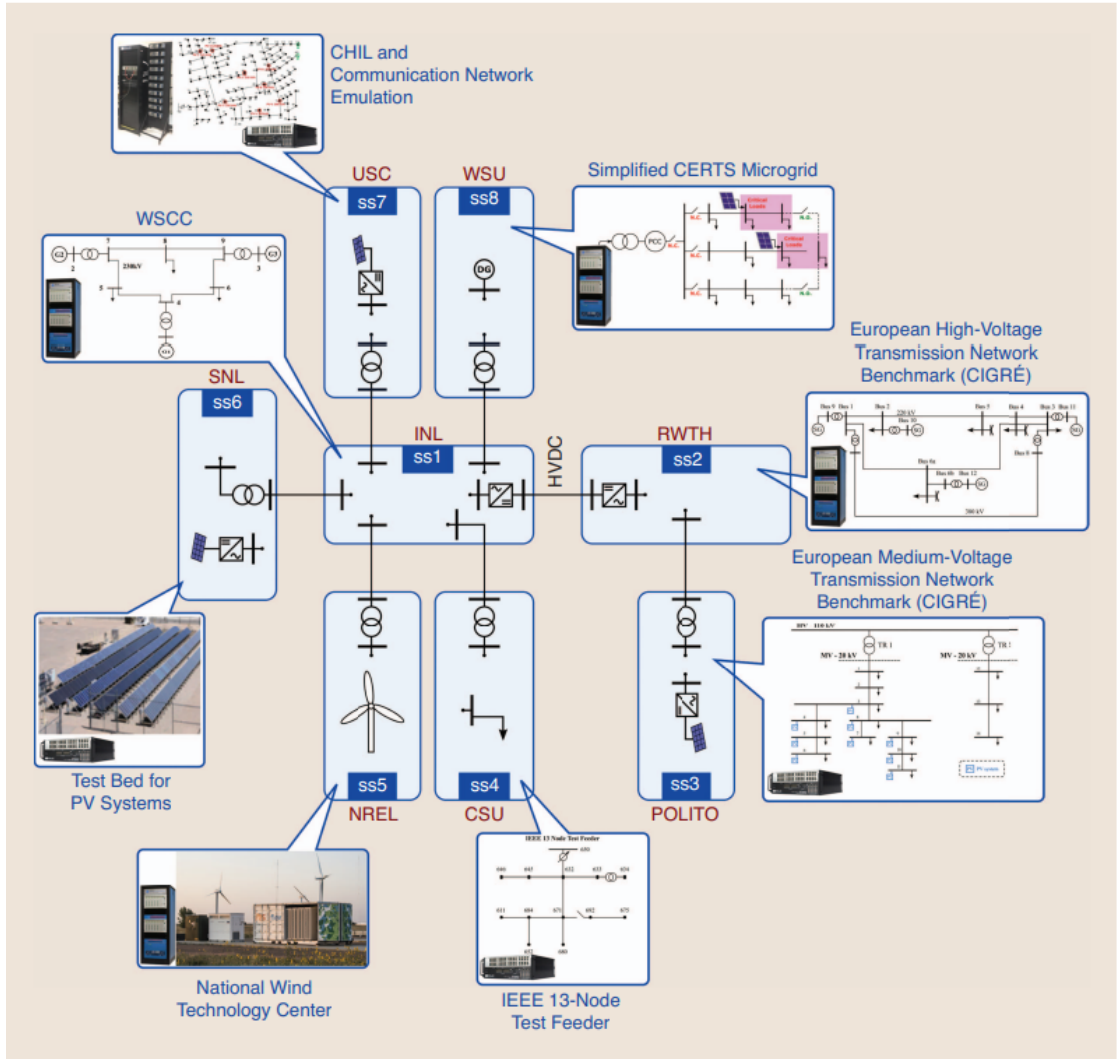


Figure 1.12: RT-Superlab Connections [7]

In the context of the Belt and Road Initiative (BRI), also the paradigm of the “Supergrid” and the role of “global interconnections” in the transition energy, are studied using Real-Time co-simulation, in March 2018, within the Joint Research Center on Energy Transition, Modeling and Simulation (ETMS), realized between Shanghai Jiao Tong University and Politecnico di Torino (Figure 1.13) [3].

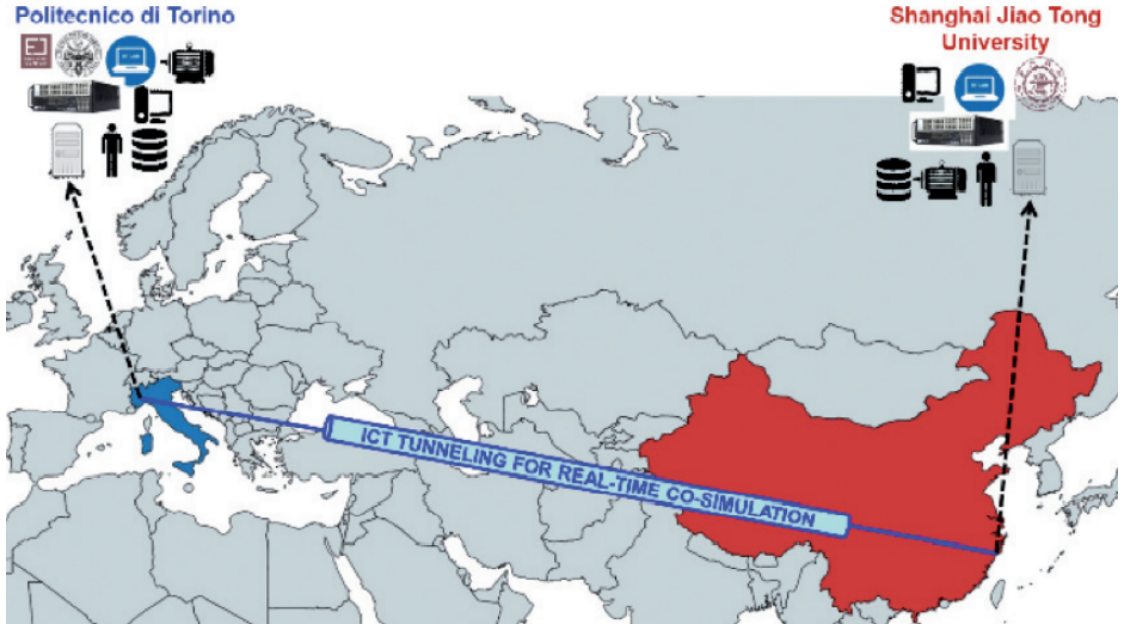


Figure 1.13: ETMS Connection [3]

Previous experiences refer to applications where various simulation programs that interact with each other remotely, enriching the possibility of representing complex systems that can be different from each other. These experiences clearly show the possibility to apply R-SIL and R-HIL technologies to the energy systems sector [3]. A recent example of R-PHIL co-simulation is the collaboration between Politecnico di Bari (PoliBa) and Politecnico di Torino (PoliTo), for the setting up of a permanent cooperative co-simulation platform, that permits to share Real-Time laboratory resources for both research and education applications. The two laboratories, LabZERO (Bari) and G-RTSLab (Turin), are located at a geographical distance of about 1,000 km. The R-PHIL co-simulation architecture is designed to integrate the response of LabZERO’s microgrid in the simulation of a larger electrical power system, simulated in Real-Time, by both units located in Bari and Turin [12]. The various tests between PoliTo and PoliBa are reported in [3],[10],[12], [5]. The architecture of the various laboratories will be presented in detail later in this thesis and new tests will be carried out on the basis of past experiments.

Chapter 2

Numerical methods and construction of the Simulink model for Real-Time execution

This chapter aims to introduce the basics of the numerical methods for solving ordinary differential equations. There are two families of numerical methods: *one-step* methods and *multi-step* methods. Only the one-step methods will be presented as in the following the Euler's method will be used as a solver in the models. Subsequently, some useful concepts for the choice of some parameters for the construction of a Simulink model suitable for Real-Time applications will be briefly presented. Finally, since the OPAL-RT[®] hardware and software products are used in this work, all the blocks necessary to build a Real-Time model in the RT-Lab[®] environment will be described.

2.1 One-Step methods

This section is dedicated to solve ordinary differential equations, in the form:

$$\frac{dy}{dx} = f(x, y) \quad (2.1)$$

Equation 2.1 can be written in discrete form as:

$$actual_value = previous_value + slope \times interval_size$$

and therefore in the mathematical form:

$$y_{i+1} = y_i + \phi h \quad (2.2)$$

As can be verified in this equation, the slope ϕ is used to extrapolate a new value of y_{i+1} from the previous one, that is one step away. Equation 2.2, applied at successive intervals, allows to obtain further values and therefore to plot the function. All the one-step methods can be traced back to this general formulation: they differ in the way in which the slope is calculated.

2.1.1 Euler's method

The first derivative gives directly the slope of the point x_i :

$$\phi = f(x_i, y_i) \quad (2.3)$$

where $f(x_i, y_i)$ is the differential equation calculated in x_i and y_i . By replacing Equation 2.3 in Equation 2.2, the following result is obtained:

$$y_{i+1} = y_i + f(x_i, y_i)h \quad (2.4)$$

The Formula 2.4 is called *Euler's method*: the new value of y_{i+1} is computed using the slope, that is equal to the first derivative calculated at the point of origin (x_0, y_0) to extrapolate linearly along the interval h (Figure 2.1).

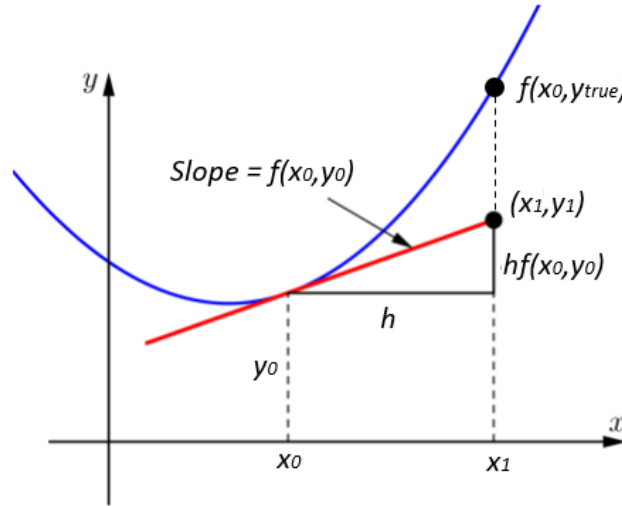


Figure 2.1: Euler's method

In general, the solution of ordinary differential equations is followed by two types of errors:

1. *truncation or discretization error*, related to the techniques used to approximate the variable y
2. *rounding errors*, related to the number of significant digits that can be processed by the calculator

Truncation errors can be broken down into two parts: the first one, called *local truncation error* comes from applying the method to the single interval; the second one, called *propagated truncation error*, depends on the approximations introduced in the previous steps. The sum of the two is called *global truncation error*.

To evaluate the amplitude and the characteristics of the local truncation error introduced by Euler's method it is possible to use the Taylor series. Remembering the Equation 2.1 and defining the first derivative of the function f as $f'(x_i, y_i) = d(f(x_i), y_i)/dx$ it is possible to write:

$$y_{i+1} = y_i + f(x_i, y_i)h + \frac{f'(x_i, y_i)}{2}h^2 + \dots + \frac{f^{(n-1)}(x_i, y_i)}{n!}h^n + o(h^{n+1}) \quad (2.5)$$

Equation 2.5 indicates that the local truncation error is proportional to the width of the interval raised to $(n + 1)$. From the Equation 2.4 appears that the Euler's method is equivalent to Taylor's series expansion to term $f(x_i, y_i)h$ and consequently the local truncation error results:

$$E_l \simeq o(h^2) \quad (2.6)$$

Where E_l is the local truncation error.

Equation 2.5 allows to quantify only the local truncation error accumulated in a single interval and it does not allow us to know the propagated error and therefore the global error. However, it can be demonstrated (Carnahan et al., 1969 [13]) that the global truncation error is proportional to the width of the interval according to:

$$E_g \simeq o(h) \quad (2.7)$$

where E_g is the global error. Equation 2.7 leads to the important conclusion that it is possible to reduce the global error for the Euler's method by reducing the interval h [13].

2.1.2 Runge-Kutta methods

All the Runge-Kutta (RK) methods can be traced back to the equation:

$$y_{i+1} = y_i + \phi(x_i, y_i, h)h \quad (2.8)$$

where $\phi(x_i, y_i, h)$ is called increment function and can be interpreted as the average slope within the interval. The incremental function can be written in the general form:

$$\phi = a_1k_1 + a_2k_2 + \dots + a_nk_n \quad (2.9)$$

where the various function k are repeatedly related to each other, that is k_1 appears in the expression of k_2 and so on. Different types of RK methods can be identified using a different number of terms in the increment function, identified by n . Using an $n = 1$ the RK method is, in fact, Euler's method. The number n of terms corresponds to the order of the method.

Regarding local and global truncation errors, it is possible to generalize the results obtained for the Euler method: for a generic order n the local and global truncation errors result respectively $o(h^{n+1})$ and $o(h^n)$ [13], i.e., again both errors decrease when the interval length decrease.

2.2 Choice of simulation parameters

The concepts introduced in the previous section must be taken into account for the build of Simulink models. First of all it is necessary to choose the type of solver suitable for the application. There are two types of solvers in the Simulink environment:

- Fixed-step solver
- Variable-step solver [14]

This thesis is based on Real-Time applications and in particular on the concept of geographically distributed co-simulation. For this type of application it is necessary to configure the model for fixed-step. The type of fixed-step solver, step size, and number of iterations that are specified affect the speed and accuracy of the Real-Time simulation [15]. To limit the computational time, it was chosen the `ode1 (Euler)` solver: it uses the Euler integration method to compute the model state at the next time step as an explicit function of the current value of the state and the state derivatives. This solver requires fewer computations than a higher order solver although it may be less accurate [14].

It is important to clarify from now on the various times that will occur in the simulations proposed in the rest of the work. The *time-step* T_s is set in the Simulink model and it is "read-only" by the Real-Time simulator: the smaller the T_s , the more precise the simulation result will be. For example to reproduce a sinusoidal signal of period $T = 20$ ms, it is possible to choose different values of T_s . The value of T_s defines the number of samples per period of the signal. To represent the signal as faithfully as possible, according to its period T , it is recommended that $T = NT_s$, where N is the number of samples per period.

In the context of remote co-simulations, it is necessary to send data between the two or more simulators involved. A new time, called *sending-rate*, is then introduced. The sending rate is an integer multiple of T_s and represents how often the data are sent to the remote simulator. Due to the communication problems described in Section 1.4 the shorter is the sending rate, the higher is the number of lost packets related to the communication medium that can cause a degradation of simulation fidelity or stability problems [3]. This will be verified in the Chapter 3 where some tests will be carried out using the Internet network for the communications. The tests will be conducted both locally and at a distributed level involving other universities.

The result is a dilemma between the choice of *Time-Step* which should be small enough to faithfully reproduce the signal and the choice of *sending rate* which, on the contrary, cannot be too short for not losing too much data. For this reason, in the context of geographically distributed co-simulation for Power Systems, it is not recommended to send the signals every *Time-Step* due to the problems described in Section 1.4 that could degrade the fidelity of the co-simulation.

2.3 Preparing a model for Real-Time execution

After introducing some fundamental concepts on Real-Time simulations and co-simulations, it is useful to introduce some application aspects for the realization of Real-Time simulations and co-simulations. For this thesis the OPAL-RT[®] hardware and software products have been used. In particular, the simulator model installed at the Real-Time laboratory of the Department of Energy (DENERG) of Politecnico di Torino is an OPAL-RT[®] 5600.

In order to run in Real-Time a Simulink model, the RT-Lab[®] software is executed. RT-Lab[®] is OPAL-RT[®] 's Real-Time multi-domain simulation software platform, and it is fully integrated with MATLAB[®]/Simulink. The main user interface of RT-LAB[®] is shown in Figure 2.2.

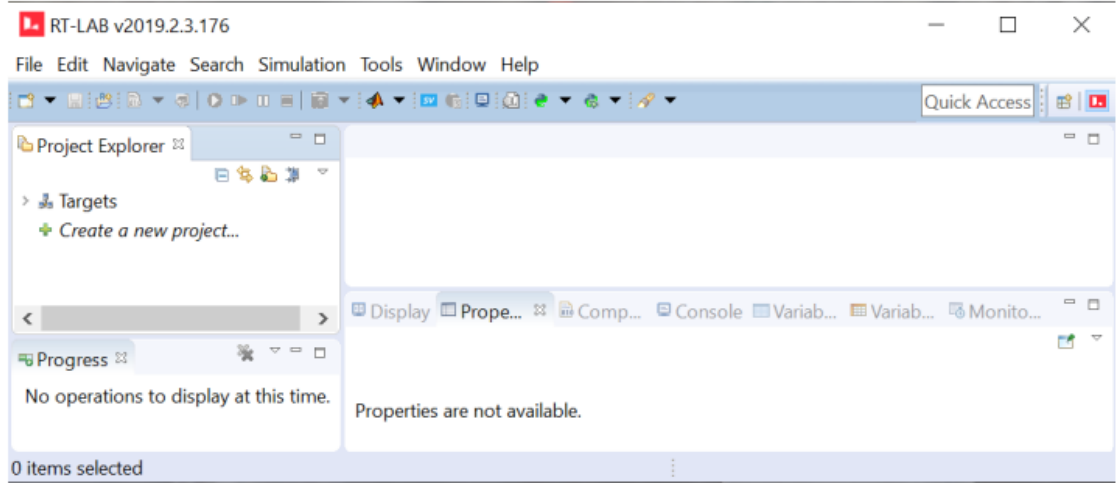


Figure 2.2: RT-Lab[®] interface

From such interface, a new model can be created. The model consists of a Simulink model that requires a certain number of blocks and components in order to run correctly. Firstly, the top-level of the Simulink model requires only subsystem blocks with specific names and functions.

The example in Figure 2.3 shows an elementary case of Simulink model: the *SC_Console* represents the user inputs and feedback with a function of Graphic User Interface (GUI) and it is run by the *Host Computer*. The *SM_Master* represents the computational block, i.e., the mathematical model where the calculations are performed. It will be executed by the Real-Time simulator, also called *Target*. *Target* and *Host Computer* are interfaced by dedicated communication protocols, such as UDP/IP and TCP/IP [2].

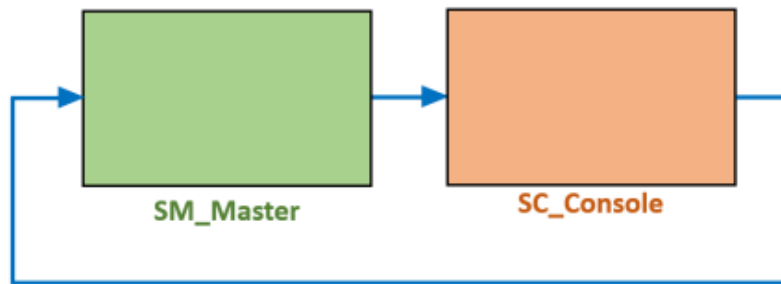


Figure 2.3: An example of top-level Simulink model

Whenever a higher computational capacity is required, e.g. the case of a complex model to be simulated, the OPAL-RT[®] Real-Time simulator has the possibility to implement a parallel computation by using multiple CPU's cores of the simulator itself. Typically, one CPU's core is used for each computational block, the example reported in Figure 2.3 refers to the use of one CPU's core. Another computational block can be created: in fact the parallel computation requires to add a new computational block as shown in Figure 2.4, which is called *SS_Slave* [2].

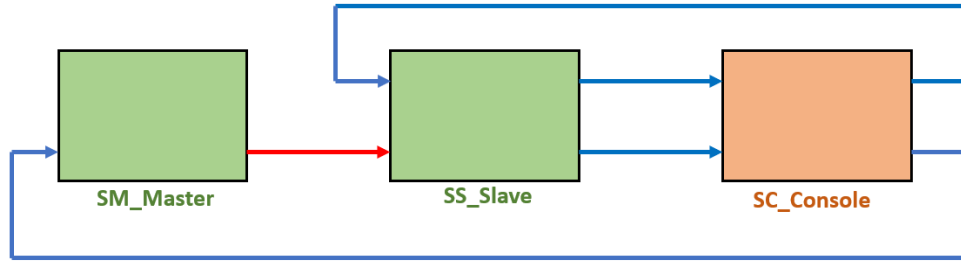


Figure 2.4: An example of a Simulink model with parallel computation

In the example reported above, part of the computation is executed using a different CPU's core with respect to the one used for the *SM_Master*. The amount of *SS_Slave* block that can be implemented is as high as the number of available CPU's cores in the Real-Time simulator.

An important aspect to pay attention to is naming the various blocks correctly in fact the use of the prefixes *SM*, *SC* and *SS* is mandatory: the software uses them to correctly distinguish the GUI application by the computational blocks and applies those at the correct CPU's cores. The concepts described previously can be summarized:

- Only **subsystems** are allowed at the top-level of the Simulink model.
- **SC** subsystems are used as a graphical interface.
- **SM** and **SS** subsystems are used for computation.
- One computation subsystem is used for each CPU core [2].

A further topic related to the implementation of the Real-Time simulation is the communication type between the different top-level subsystems. As already described in Section 1.1 there are two different type of communication: *synchronous* communication and *asynchronous* communication.

The former (red arrow in Figure 2.4) is used among CPU cores, it is a high-speed communication and thus able to support Real-Time simulations. The latter (blue arrow in Figure 2.4) is used between the *Host* and *Target computer*, it is a low-speed communication and thus only used for data logging.

The top-level subsystems are implemented by difference devices. In order to synchronize the signals in the simulation, every subsystem must include the use of the *OpComm* block (shown in Figure 2.5).

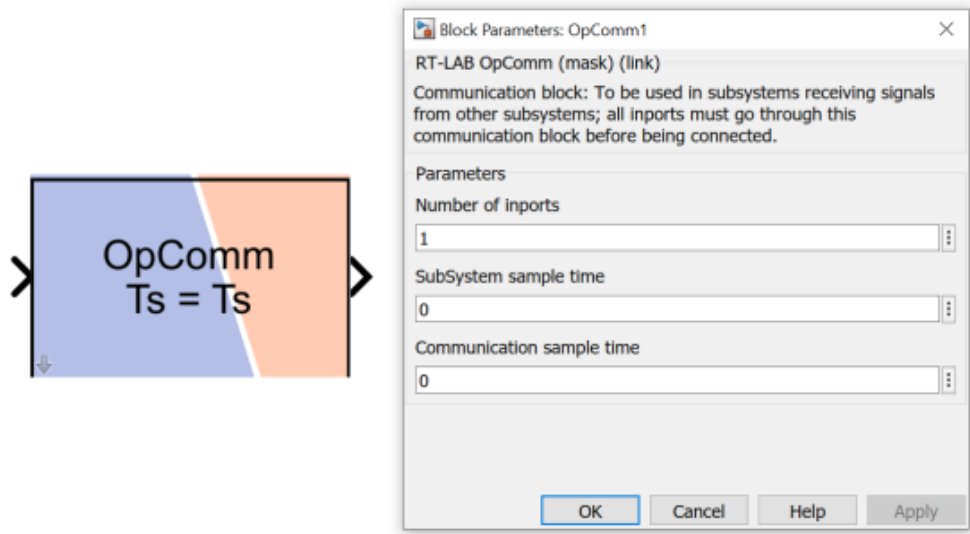


Figure 2.5: OpComm block

The *OpComm* block is used in Console, Master, or Slave subsystems to simulate the behavior of the Real-Time communication link. Any signal entering in Console, Master, or Slave subsystem has to go through the *OpComm* first [16]. The *OpComm* has to be inserted after the subsystem creation and renaming (*SM,SS,SC*). One *OpComm* can accept multiple inputs in one subsystem: each input signal can be either a scalar or vector. To guarantee a correct implementation, an *OpComm* block cannot read both *synchronous* and *asynchronous* signals. Whenever both type of signal enter in a subsystem, two *OpComm* will be required:

- One **OpComm** block receives Real-Time synchronized signals from the other Real-Time subsystems.
- One **OpComm** block receives asynchronous signals from the GUI subsystem [2].

Since this work is primarily concerned with a Remote Software-in-the-Loop (R-SIL) and Remote Power Hardware-in-the-Loop (PHIL), other blocks are required. In order to receive data with a wireless connection from another Real-Time simulation, a model requires the use of the *OpAsyncRecv* block, shown in Figure 2.6 [17].

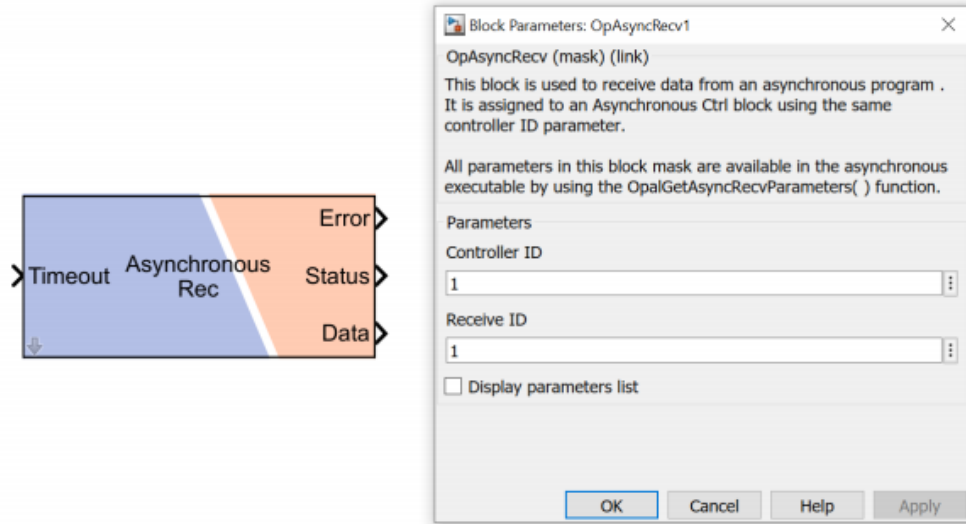


Figure 2.6: OpAsyncRecv block

Similarly, in order to send data with an Internet connection to another Real-Time simulator, a model requires the use of the *OpAsyncSend* block, shown in Figure 2.7 [18].

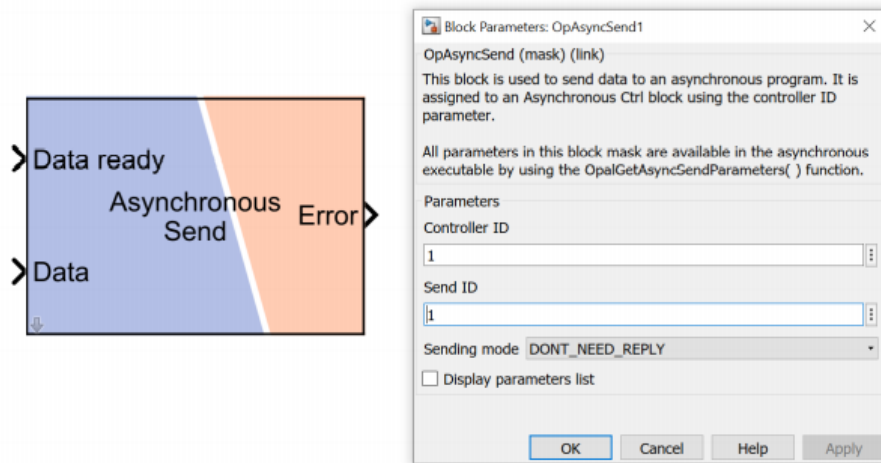


Figure 2.7: OpAsyncSend block

To setup an Internet communication between two Real-Time simulators the *OpIPSocketCtrl* block is required (Figure 2.8).

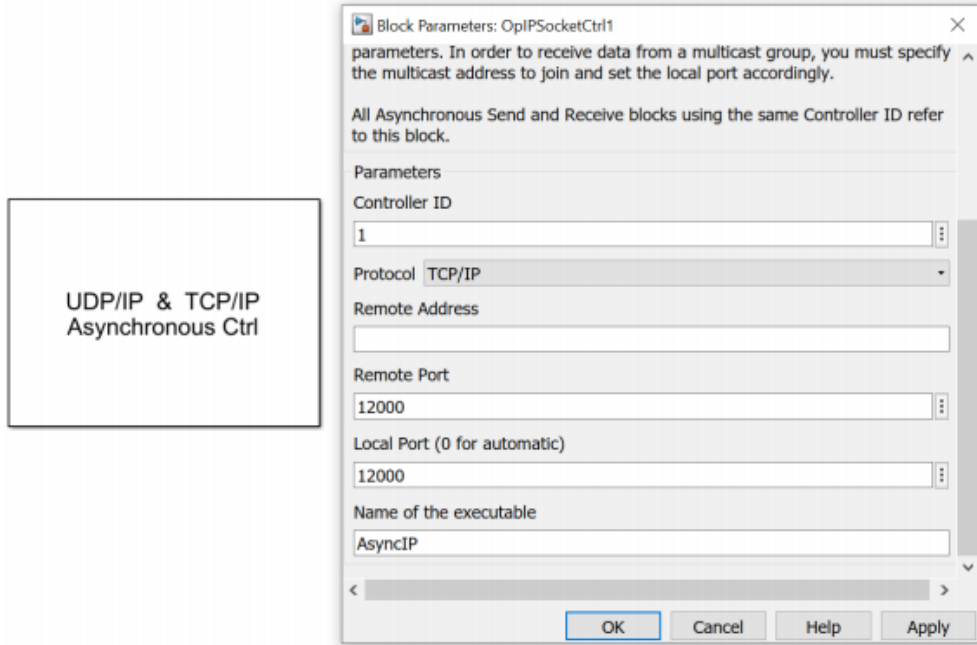


Figure 2.8: OpIPSocketCtrl block

The *OpIPSocketCtrl* block is used to control TCP/IP or UDP/IP communication with third-party applications by the use of an Asynchronous Process launched during model initialization. The block parameters allow the user to set up the specific communication parameters with the remote device. The communication port of this remote device must be enabled before the Asynchronous Process is started, i.e before the RT-Lab[®] model is loaded on the target computer [19]. Data reception and transmission is managed by *OpAsyncSend* and *OpAsyncRecv* blocks.

Chapter 3

VILLAS Framework

3.1 Background and Motivation

This thesis has as main goal to create a co-simulation platform widespread among different sites.

In the recent past, co-simulation tests have already been carried out between the laboratory in Turin and the one in Bari as described in [10] and [12]. In those cases, direct co-simulations were carried out. The two OPAL-RT[®] simulators exchanged data by means of a Virtual Private Network (VPN) tunnel, employing an IPsec encryption key for security reasons. The data was sent from the PoliTo's simulator to the PoliBa's one directly, without any type of interface between them.

In this thesis, the past experiences have been replicated but, instead of the direct connection between the two simulators, the VILLAS framework was used as an interface tool.

VILLAS framework is able to guarantee some benefits during a remote co-simulation process. In fact, it is able to interconnect a greater number of hardware and software resources existing in a laboratory and can connect a greater number of geographically distributed laboratories. Thanks to this tool it is therefore possible to extend the experience between PoliTo and PoliBa involving a greater number of universities in Italy.

This chapter will describe the main tools made available by the VILLAS framework, the first local tests performed at PoliTo to verify the correct functioning of the platform and the first tests aiming to establish the connection between the PoliTo and the PoliBa.

3.2 VILLAS node

VILLAS node is a flexible gateway for co-simulation interface developed and maintained by the Institute for Automation of Complex Power Systems (ACS). It is able to convert protocols and (de-)multiplexes signals to and from multiple sources and destinations, allowing to interface one or more resources available at each laboratory. Currently, around 18 different protocols and interfaces are supported. It collects statistics about the exchanged data, such as packet loss, one-way delay and jitter, and provides a network emulation to simulate the conditions of a geographically distributed co-simulation. VILLAS node is a C/C++ application optimized for Real-time Linux operating systems. The main interfaces of VILLAS node are shown in Figure 3.1.

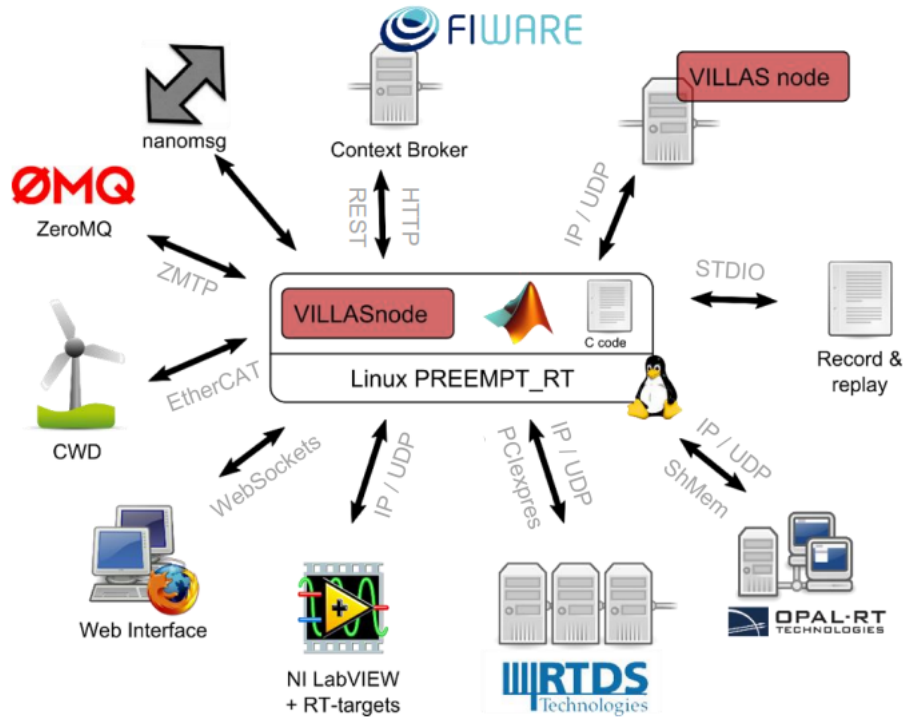


Figure 3.1: VILLAS node interfaces [20]

Generic and modular design of VILLAS node addresses challenges imposed by diversity of required interfaces with the goal to abstract this complexity with respect to simulators and other devices involved in experiment. The underlying concept of VILLAS node is a modular design with the goal to achieve flexibility and portability among laboratories. To this end, the aim is to minimise implementation of the interfaces based on specific devices or simulators, focusing instead of the implementation of required interfaces on the VILLAS node that can be utilized in

different laboratories.

A generic architecture of the VILLAS node enables extension with other interfaces in the future. VILLAS node manages multiple interfaces for data exchange among different nodes in the framework. A node refers to a physical device such as a simulator or a sensor or to any other type of functionality or a service that processes or manages simulation data [8].

Figure 3.2 shows an example of a co-simulation architecture based on the VILLAS node. In this case, a two-site scenario is shown. In the site A, a RTDS[®] simulator and an OPAL-RT[®] simulator are included and they are interfaced by VILLAS node. In the site B another OPAL-RT[®] simulator is available and it is interconnected with the simulators installed in site A via Internet.

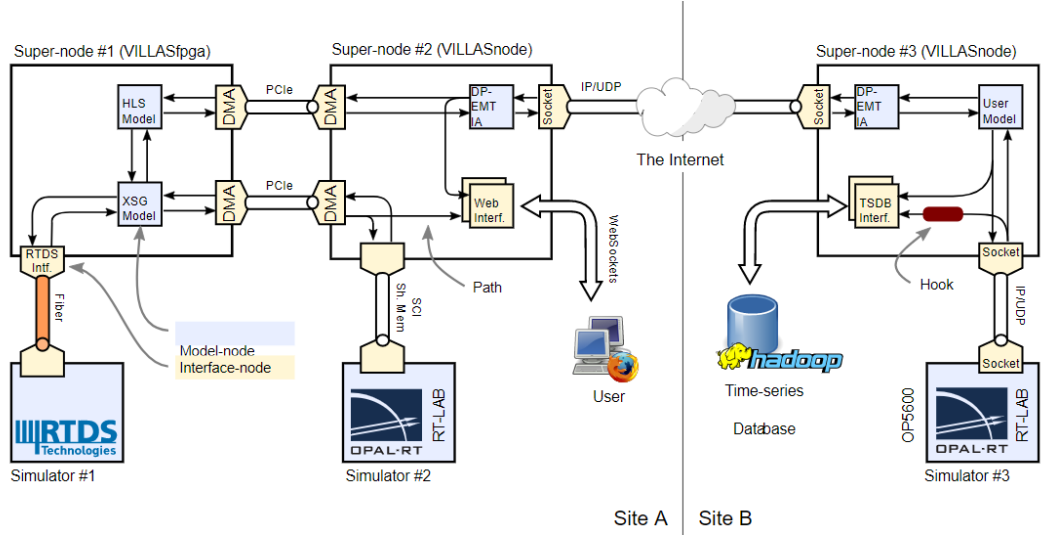


Figure 3.2: An example of VILLAS interconnection [21]

To better understand the architecture of the VILLAS node some definitions must be provided as VILLAS node is designed around the concept of *super nodes*, *nodes*, *paths* and *hooks*:

- a *super node* is a collection of nodes and paths which are interconnected. Usually, one super node is executed in a single operating system process and configured by a single configuration file;
- a *node* is an interface to a local or remote simulator, file, database etc. A *super node* consists of one or more *nodes*. A *node* acts as a sink and as a source of samples;

- a *sample* is an array of floating point or integer values with an associated timestamp and sequence number;
- a *path* connects one or more *nodes* within a *super node* and forwards *samples* from one or more *source nodes* to one or more *destination nodes*.

The flexibility offered by VILLAS allows to insert more *nodes* inside a *super node* and therefore the possibility to use more resources. Furthermore, several remote *super nodes* can be interconnected with each other through dedicated protocols [20].

3.2.1 Communication between VILLAS nodes

During a geographically distributed simulation between two laboratories, packets are sent from each interface to the remote interface with a selected sending rate. The rate at which simulators exchange their interface signals is one of the key factors, which affects accuracy of simulation results. The maximum rate is limited only by the available bandwidth of the communication link. As the link is shared with other users, the available bandwidth varies with time, which may cause congestions. Congestions have to be avoided as they lead to packet loss and a degradation of simulation fidelity.

The most used communication protocol in Real-Time applications is the UDP (User Datagram Protocol) type protocol. The UDP protocol is a connection-less type and, in contrast to TCP (Transmission Control Protocol), it not required the re-transmission of lost packets. This aspect represents an advantage in Real-Time applications, as delay variation is lower compared with connection-oriented protocols such as TCP whose re-transmission of lost packets and required acknowledgment of received packets increase latency and jitter [8].

In the context of communication between VILLAS nodes, the ACS Institute designed a light-weight protocol to facilitate a fast transmission of packets. This protocol, called *VILLAS binary*, is designed to be lightweight and easily readable by machines in order to avoid more complex parsers which are required for human readable formats such as JSON and it is particularly advantageous in applications where sending rates are very high and processing time has an impact [22].

Figure 3.3 illustrates the binary UDP packet format that is used for the interface for sending data between the various laboratories.

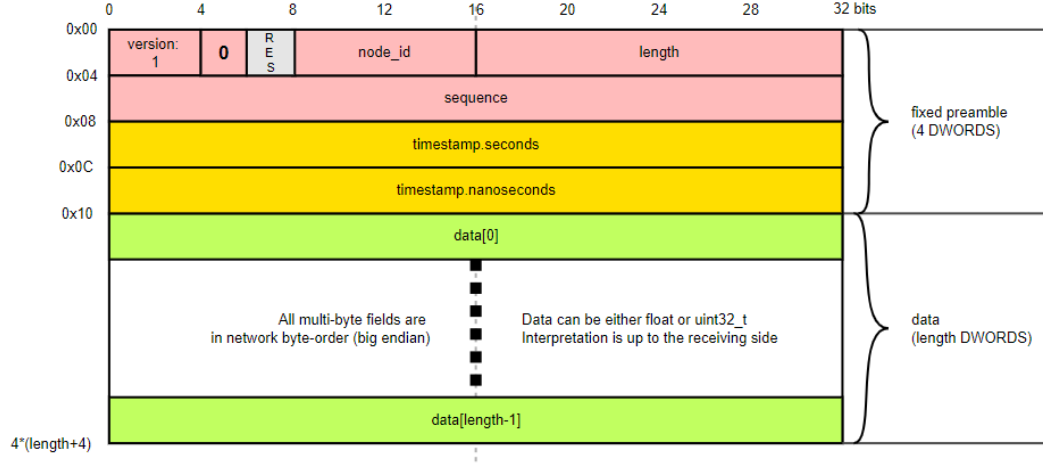


Figure 3.3: UDP packet format [22]

Usually a simulator sends a message that contains a variable number of values per timestep. Each message contains a header with the following fields:

- 32 bit floating-point or integer values;
- 32 bit timestamp (integral seconds);
- 32 bit timestamp (integral nanoseconds);
- 16 bit sequence number;
- 4 bit version identifier [22];

3.2.2 Node types and configuration

All communication partners which are interfaced by the VILLAS node gateway are represented by nodes. These nodes act as sinks or sources for simulation data. VILLAS node supports a great number of node-types, the complete list is contained in [23].

Regarding this thesis and, in general, for all the geographically distributed Real-Time co-simulations, the **socket** node-type is used. The **socket** node-type is the most comprehensive and complex one. It allows to send and receive simulation data through the network. This node-type only supports connection-less based protocol. This means that UDP protocol is supported, while TCP protocol is not supported [24].

The VILLAS node configuration consists of a single file and the recommended file format is JSON. As for the socket node the main settings are summarized in Table 3.1.

Table 3.1: Configuration instructions for a node

Instruction	Description
type	Select the node-type (i.e. " socket ")
layer	Select the network layer which should be used for the socket
format	The payload format which is used to encode and decode exchanged messages
in.address	The local address and port number this node should listen for incoming packets
out.address	The remote address and port number to which this node will send data

If multiple nodes are created, a path between the nodes must be specified in order to properly exchange data.

A path is a uni-directional connection between incoming and outgoing nodes: it forwards messages from one or more incoming nodes to one or more outgoing nodes. Therefore, it represents a n -to- n relation among nodes. For bi-directional communication, a corresponding path in the reverse direction must be added.

By default, message contents are not altered. However, every path supports optional *hook* functions which allow user-defined operations on the sample contents. *Hooks* are simple callback functions which are called whenever a message is processed by a path. There are several built-in hooks for:

- collecting, show and reset statistics;
- drop reordered messages;
- verify message metadata;
- handle simulation restarts;
- remapping values of a sample;
- overwriting / updating timestamps;
- converting data-types;
- down-sampling.

A complete list of all available hooks is available in [25].

3.3 Local test of VILLAS node

The first step on the use of VILLAS node was performed locally at the Politecnico di Torino (PoliTo). In this way, it was possible to check the operation of some configuration functions of the VILLAS node and estimate the delay that cannot be eliminated due to the communication via Internet.

The test involved an OPAL-RT[®] 5600 and a VILLAS node. The VILLAS node was installed and configured on a PC in the same laboratory, as the simulator and connected to the same Internet network. Please note that the configuration file of the VILLAS node used for this test is reported in Appendix A.2.

The data exchanged consists of a sine signal, which is generated in the Simulink model and loaded to RT-Lab[®]. The data is then sent to VILLAS node and then sent back to the OPAL-RT[®] simulator. This process is called *loop-back*. The same test has been implemented a number of times by varying the sample time. The obtained results are shown and discussed later in this section.

In Figure 3.4 is illustrated the top level of the model, merely composed by subsystem blocks. In this specific case, a SC_Console block and only one SM_Master block have been implemented.

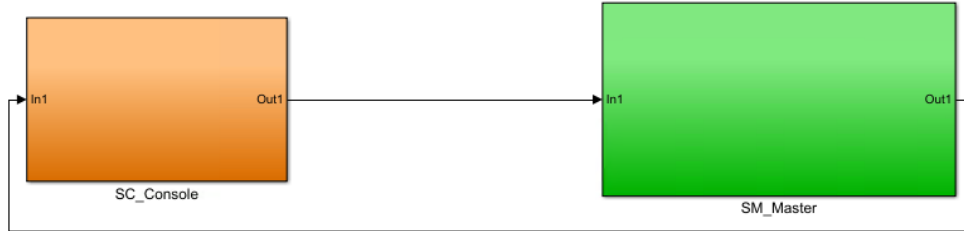


Figure 3.4: Top level Simulink model

The scheme included in the SM_Master block is shown in Figure 3.5. A sine signal, with a frequency of 50 Hz and an amplitude of 1, is sent to VILLAS node through the *OpAsyncSend* block. The output of the *OpAsyncRecv* block corresponds to the return data from the loop-back. Thanks to the *OpWriteFile* block is possible to save the send and receive signals to MATLAB[®] workspace to compare them. The *OpTrigger* block is used to start and to interrupt the data saving-on-file process at the desired time.

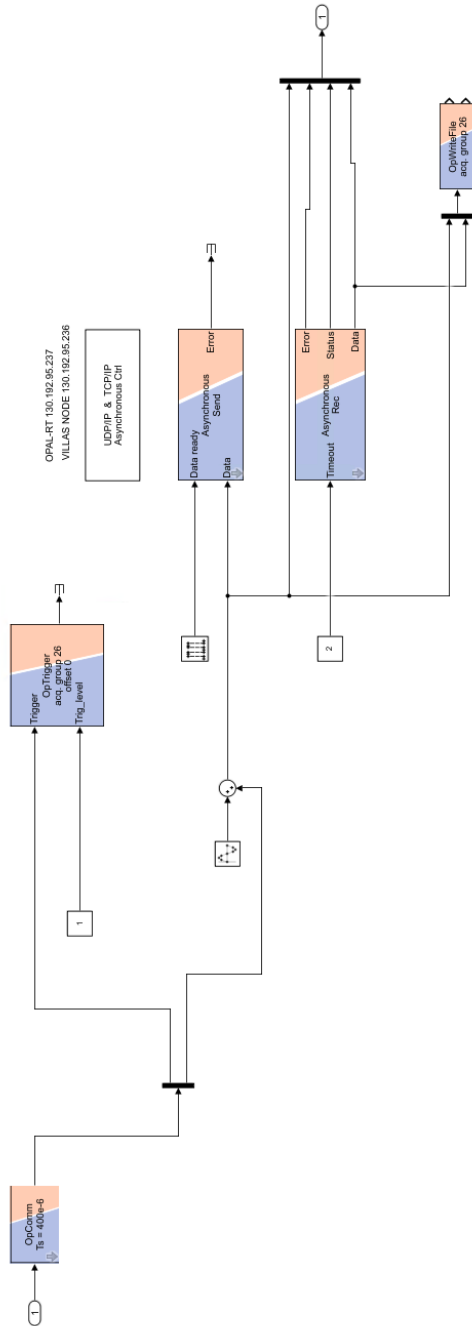


Figure 3.5: SM_Master Simulink scheme

To assess the latency of the communication between the Real-Time simulator and the VILLAS node is necessary to consider a precise time reference. For this reason, an offset of 0.5 has been added to the sine wave at a certain time. The offset is added manually through a switch inserted in the SC_Console, as shown in Figure 3.6.

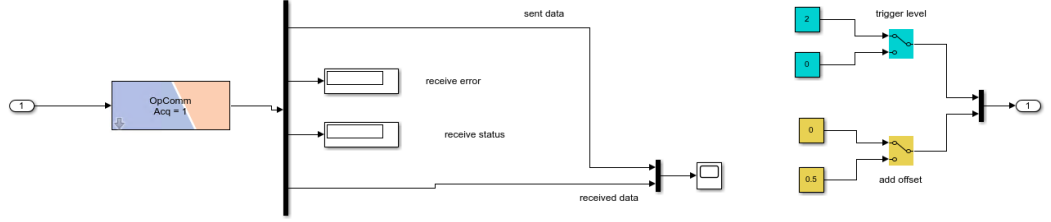


Figure 3.6: SC_Console Simulink scheme

This same test has been implemented a number of times with a variation in the sample time. Particularly, the sample time t_s of the communication has been varied from $50\text{ }\mu\text{s}$ up to 1 ms ($50\text{ }\mu\text{s}$, $100\text{ }\mu\text{s}$, $200\text{ }\mu\text{s}$, $400\text{ }\mu\text{s}$, 1 ms). The results are given below:

- Figure 3.7 represents the sent and received sine signal with $t_s = 50\text{ }\mu\text{s}$.
- Figure 3.8 represents the sent and received sine signal with $t_s = 100\text{ }\mu\text{s}$.
- Figure 3.9 represents the sent and received sine signal with $t_s = 200\text{ }\mu\text{s}$.
- Figure 3.10 represents the sent and received sine signal with $t_s = 400\text{ }\mu\text{s}$.
- Figure 3.11 represents the sent and received sine signal with $t_s = 1\text{ ms}$.

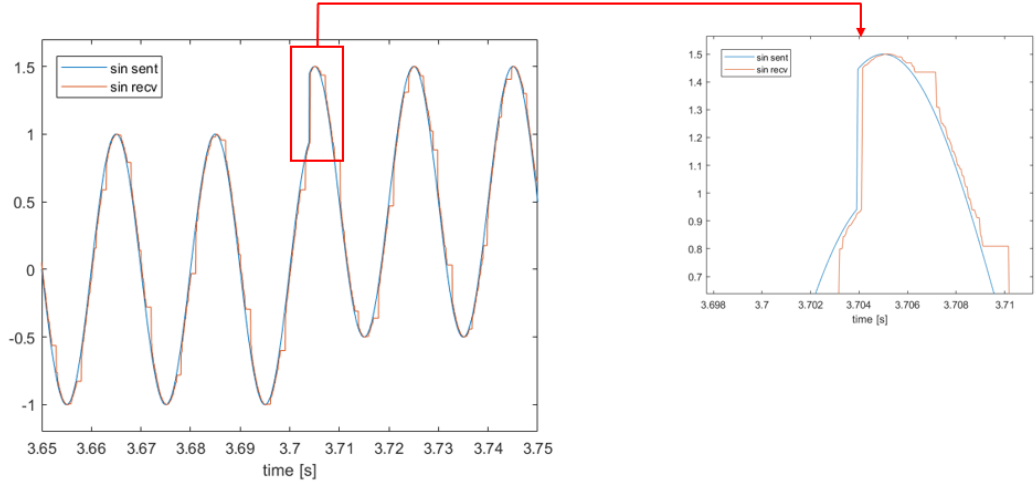


Figure 3.7: Sent and received waveform ($t_s = 50 \mu\text{s}$)

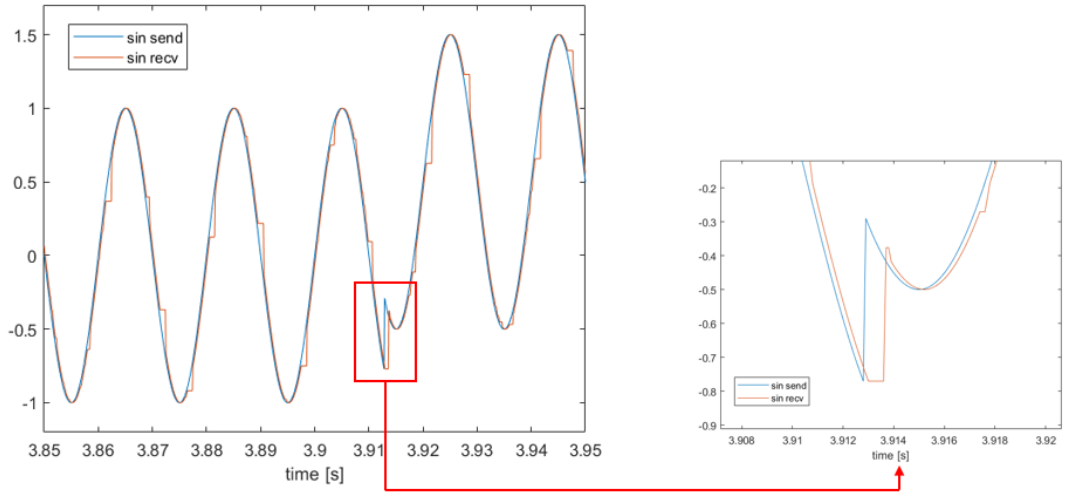


Figure 3.8: Sent and received waveform ($t_s = 100 \mu\text{s}$)

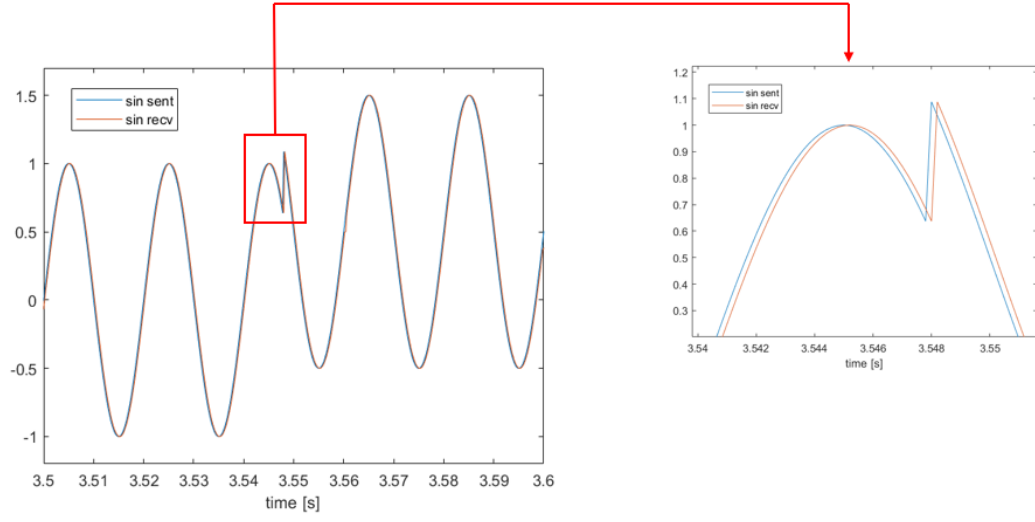


Figure 3.9: Sent and received waveform ($t_s = 200 \mu s$)

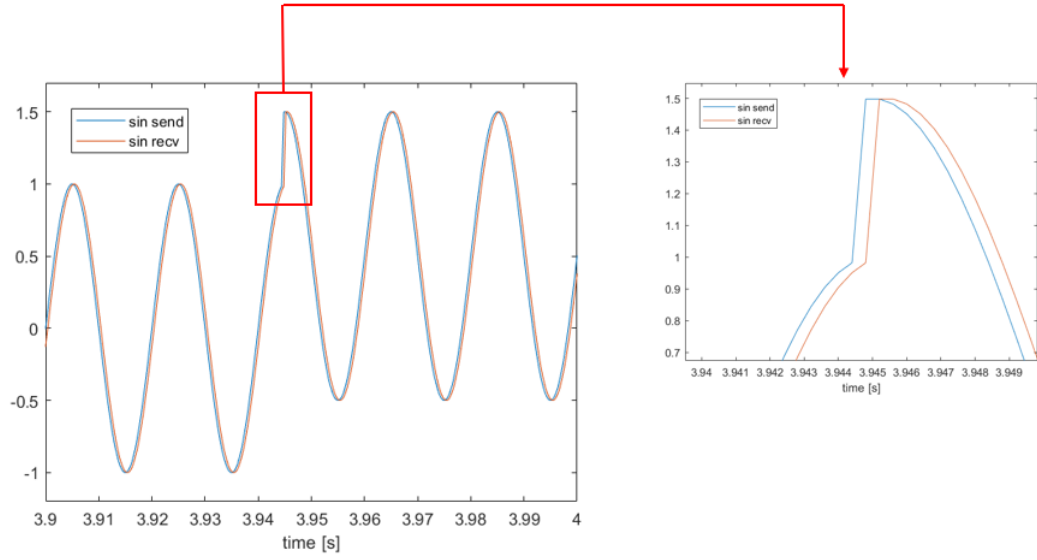


Figure 3.10: Sent and received waveform ($t_s = 400 \mu s$)

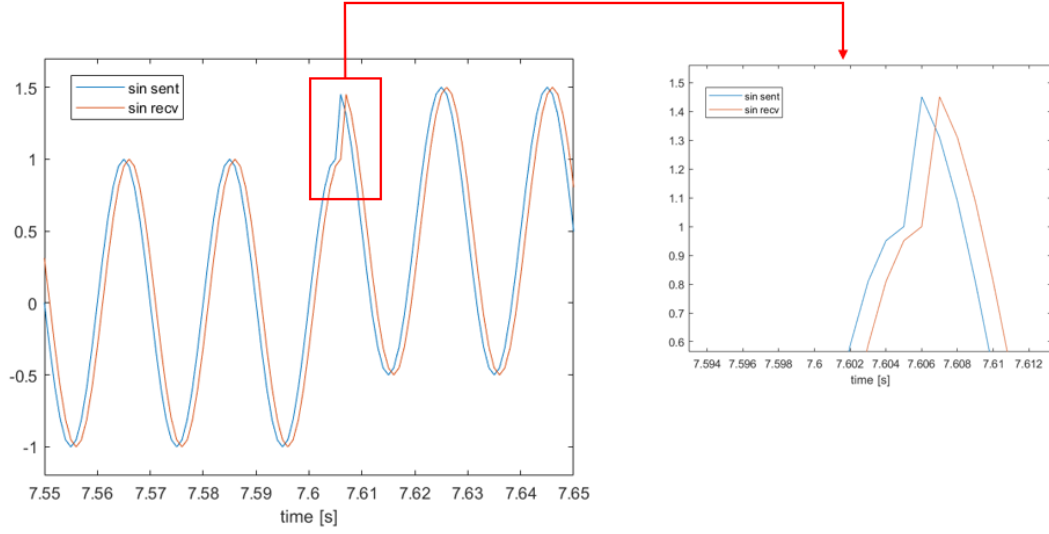


Figure 3.11: Sent and received waveform ($t_s = 1$ ms)

Table 3.2 summarizes the test results, while Figure 3.12 shows the magnitude of the error between the sine wave sent and the one received after the loop-back. The analysis refers to a sample of 10,000 samples and its objective is twofold: 1) to estimate the delay introduced by the communication 2) to verify the fidelity of the simulation according to the time step and the sending rate.

Table 3.2: Statistic on data exchange between Polito OPAL-RT and Polito VILLAS node (10,000 samples)

Time step	Delay	Delay (Time step)	Lost data	Lost data (%)
50 μ s	200 μ s	4	6803	68.03%
100 μ s	200 μ s	2	2671	26.71%
200 μ s	200 μ s	1	36	0.36%
400 μ s	400 μ s	1	12	0.12%
1 ms	1 ms	1	8	0.08%

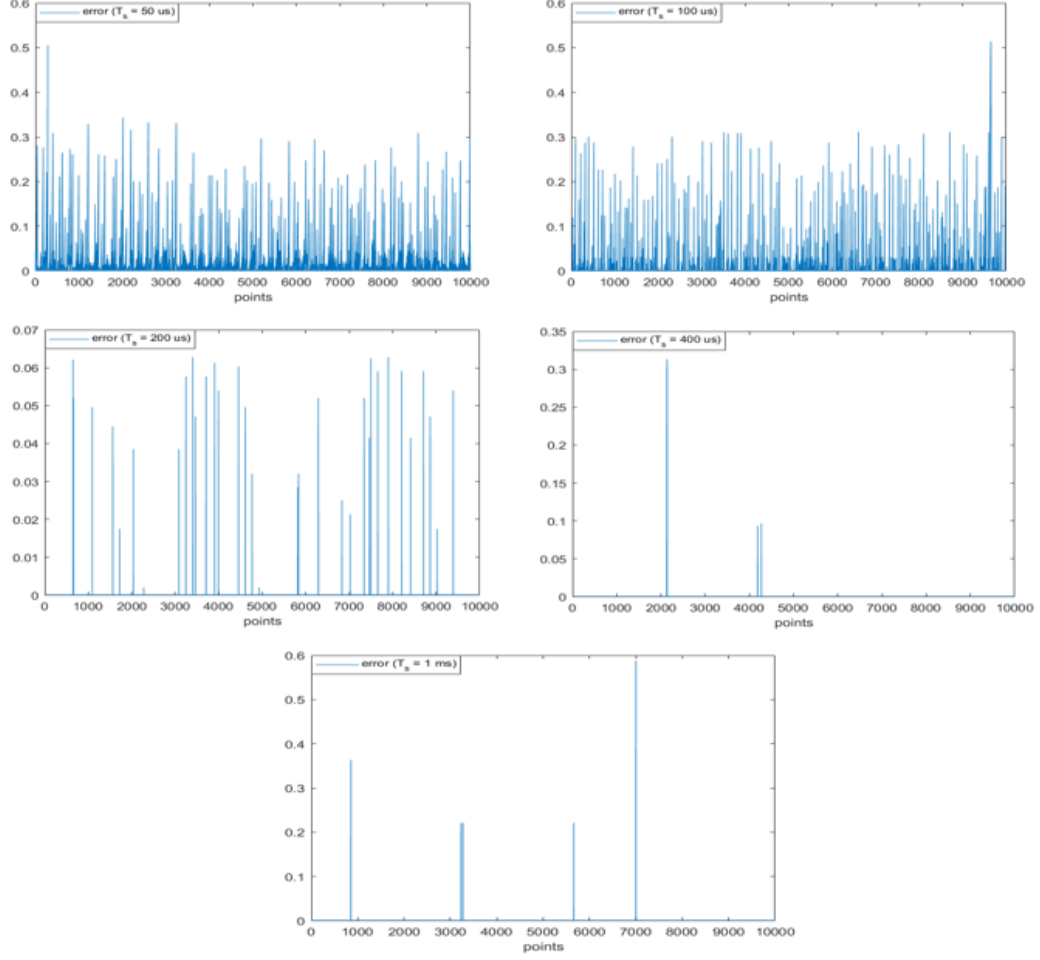


Figure 3.12: Error amplitude according to the different sending rate. (Top-right) $50 \mu s$; (Top-left) $100 \mu s$; (Centre-right) $200 \mu s$; (Centre-left) $400 \mu s$; (Bottom-centre) $1 ms$

First it has occurred that the delay can never be lower than the duration of the time step. This aspect depends on the *OpAsyncSend* block, which sends a signal to VILLAS node at a generic time t_k , the same signal will be received by the *OpAsyncRecv* block at least at the time t_{k+1} ; for this reason, the delay is always greater or equal to the time step. In detail, with smaller time step (i.e. $50 \mu s$ and $100 \mu s$) delay results turns out to be larger than a time step. Conversely, by increasing the duration of the time step, the delay becomes equal to the time step: this means that the time required to VILLAS to receive, process and re-send data to the simulator is less than the time step. Tests have shown that this time is between $100 \mu s$ and $200 \mu s$.

Regarding the lost data, from Figure 3.12, it is highlighted that as the time step increases the number of lost data is drastically reduced. This aspect is linked to latency due to congestion of the ports and occurs when a high number of data are exchanged in a very short time. Furthermore, due to the property of the UDP protocol according to which the last incoming data is considered valid, a large number of data are discarded.

The assumptions made in Section 2.2 are confirmed by this test. In this regard it is possible to notice how the original sinusoidal signal is better represented as the time step becomes shorter. This aspect is appreciable from the last waveform where a 1 ms time-step is used with only 20 samples per period: the sine signal has been re-built using only 20 points per period, which means that it was re-constructed by using only 20 points per period and therefore with lower accuracy than, for example, in the case with a time-step of 50 μ s and 400 points per period. From the point of view of the simulation fidelity and therefore the number of lost data, however, the result is the opposite. Considering that a data has been sent for each time step and therefore the sending-rate is equal to the time-step, the case with the longest time step is definitely the most favorable.

3.4 Remote test of VILLAS node

After having conducted some local tests to verify the functioning and implementation of the VILLAS node, a test was conducted involving also Politecnico di Bari (PoliBa). Although, from a conceptual point of view, the first test with the PoliBa is not very different from the previous one conducted locally, it was important to be able to establish the first connection between the two Universities through the VILLAS node.

The test consists in the exchange of a sine signal. The signal is generated in the Simulink model built at the PoliTo. In this test, in addition to the Real-Time simulators of the two universities, the VILLAS nodes are also involved as shown in Figure 3.13.

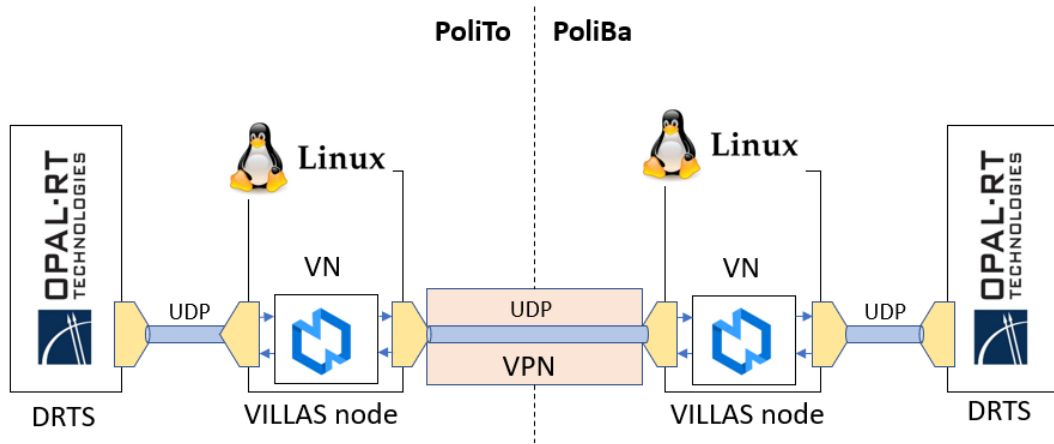


Figure 3.13: Architecture of VILLAS Framework implementation

The same test was performed in [3] with the only difference that, in that case, the connection between the two simulators was direct and the exchanged data did not pass through the VILLAS nodes. In this test both the direct connection and the connection through the VILLAS nodes interface were tested. The configuration function used in this test of the VILLAS node is illustrated in Appendix A.3. Regarding the Simulink model used in the test, the SC_Console is the same as the previous test (Section 3.3) while the SC_Master has been modified as shown in Figure 3.14.

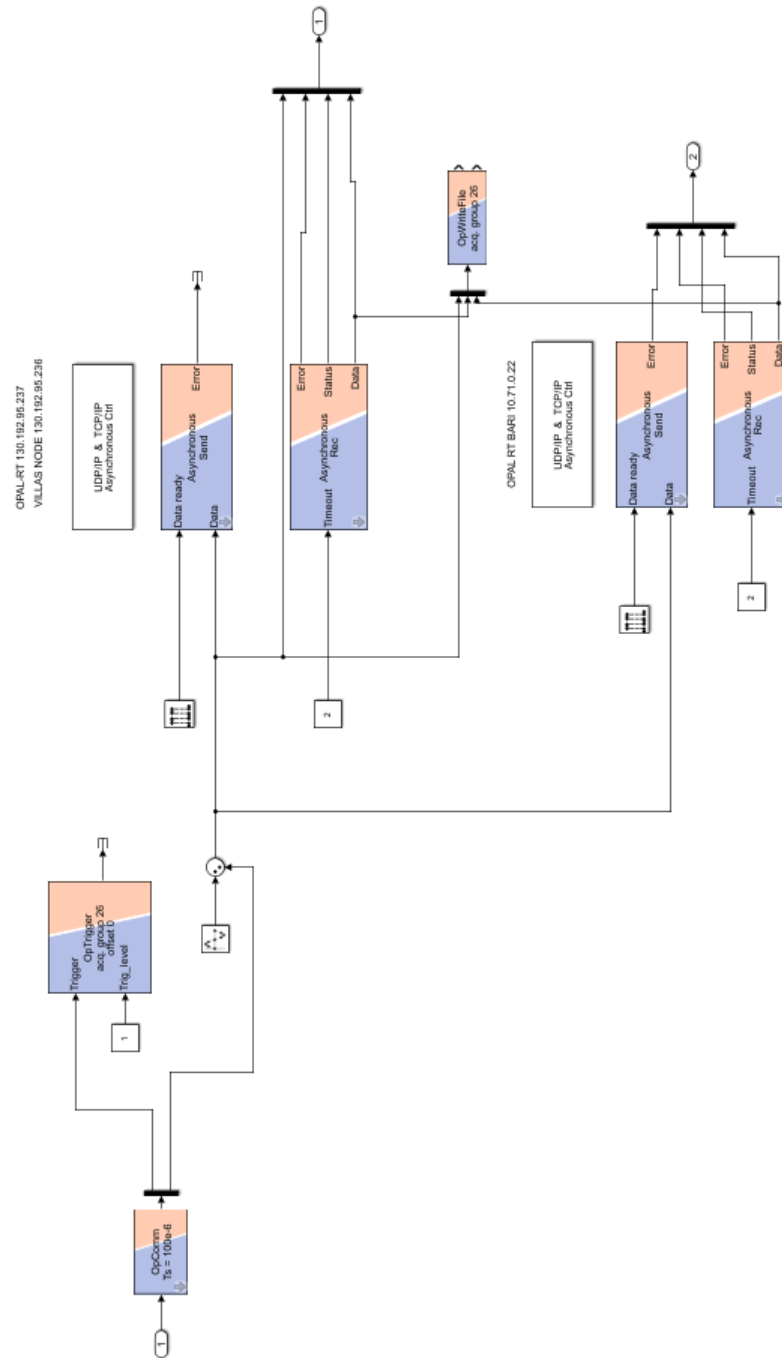


Figure 3.14: SM_Master Simulink model for the remote test

The changes are visible in the left part of Figure 3.13 and are necessary to directly receive the signal from the OPAL-RT[®] simulator located in Bari. Some details on the implementation are given in Appendix A.3.

Compared to the local test, a substantial change has been made: the data is not sent every time step, but one is sent every 10 time steps. The time-step chosen in this test is $100\text{ }\mu\text{s}$ and in this way a data will be sent every 1 ms . The result is shown in Figure 3.15. The red sine is the sent signal, the blue sine is the received signal from VILLAS node and the green signal is signal received directly from the PoliBa's simulator. To evaluate the real extent of the delays in the two cases, as for the local test, an offset of 0.5 to the sent sine signal was manually entered after a certain time.

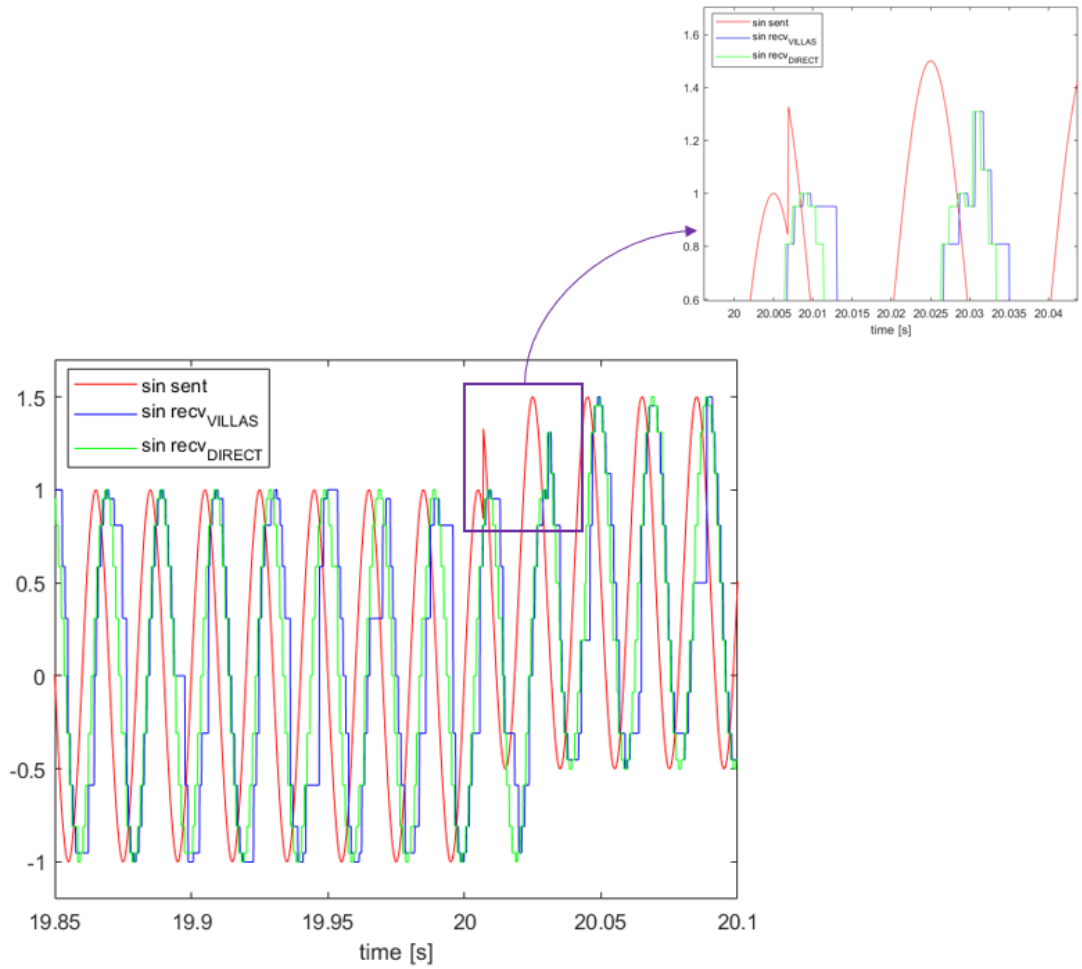


Figure 3.15: Addition of the offset to the sine wave in the remote test

The results of the delays in the two cases with and without VILLAS node are reported in Table 3.3.

Table 3.3: Results of delays with and without VILLAS node

Connection type	Delay
with VILLAS node	23.8 ms
without VILLAS node	23.5 ms

The results shown that the delays obtained using the direct connection and those obtained using VILLAS node are very close. The difference is 300 μ s, which means 3 time-step. The results are in line with the test performed in [3], where the overall delay was estimated to be about 25 ms. Assuming valid the hypothesis that the transmission delay does not depend on the transmission direction, it is possible to estimate a delay of the single transmission as about 12.5 ms. This latency value between the two laboratories, which are about 1,000 km away from each other, turns out to be a good result.

As for the fidelity of the received signal, it can be noted from Figure 3.15 it appears very different from the original sine. In particular, the problems related to latency in communication are evident, especially in the test with the VILLAS node. Please note that it is difficult to make a comparison between the signal sent and those received, as a data is sent every 10 time-steps and therefore the signal arriving at the Real-Time simulator in Bari, and consequently the one received after the loop-back, is different with respect to the one generated by the Real-Time simulator in PoliTo.

The aspects highlighted in this Section suggest not to directly use the received sine waveforms in the co-simulation process, but to use a phasor-based approach. This type of approach will be described and used in all subsequent tests.

Chapter 4

Application of VILLAS Framework

In this chapter some Real-Time remote co-simulation tests will be presented. These tests are part of the initial activity of the ENET-RTLlab, promoted by ENSIEL. The universities currently involved in the project are: Politecnico di Torino (PoliTo), Politecnico di Bari (PoliBa) and Università di Genova (Unige and Savona Campus). In each of the locations geographically distributed on the national territory there are hardware and software elements that can be integrated in a Real-Time co-simulation thanks to the open source protocol VILLAS suitable for the Real-Time distributed co-simulation.

4.1 Test with Savona

The first test on a real grid model was developing by involving Politecnico di Torino (PoliTo) and the University of Genova (Campus Savona). The Real-Time co-simulation was performed to verify the feasibility of the connection between the two universities through the VILLAS node as well as the correctness of the distribution network model implemented in PoliTo following the insertion of a remote hardware element.

The implemented model simulates a MV feeder (22kV) composed of 8 load nodes to which a photovoltaic (PV) generator has been connected. The HV network (220 kV) is considered ideal and it is represented with the Thevenin equivalent circuit. More details about the MV feeder are listed in Appendix B. The transmission grid and the distribution feeder are implemented in Turin, while the PV generator is actually installed in the Savona's Campus. The PV generator of Savona has been inserted in the model of Turin downstream a transformer MV/LV and represented

in the model as a flexible load. The network schematic is shown in Figure 4.1.

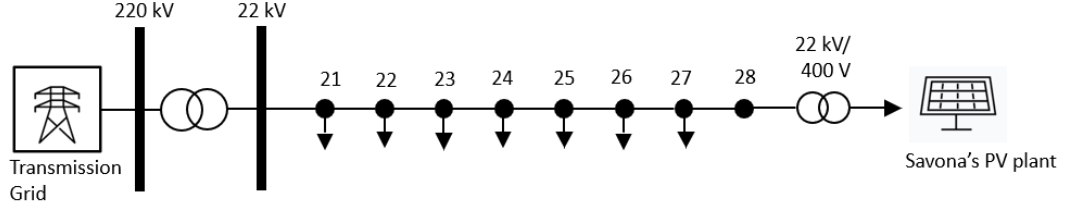


Figure 4.1: Schematic representation of the distributed feeder in the test with Savona

Regarding the communication aspects between the two remote sites, thanks to the use of the VILLAS node it was possible to interface the simulated model of PoliTo with the hardware component installed in Savona's Campus (PV generator). The communication infrastructure is schematized in Figure 4.2.

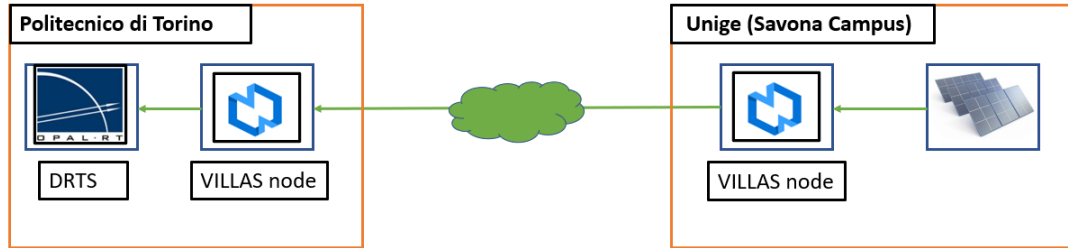


Figure 4.2: Schematic representation of the communication infrastructure in the test with Savona

Figure 4.2 shows that the communication is one-way and this means that the VILLAS node located in Turin is configured with the only scope to receive signals from Savona's VILLAS node without sending any data back. To achieve this goal, the configuration file for the VILLAS node described in Appendix B.1 was used. In this view, in the Simulink model implemented in PoliTo, was inserted only an *OpAsyncRecv* block to receive the signal sent from Savona's node. The SM_Master of the Simulink model is shown in Figure 4.3.

The communication between the two remote nodes starts at time $t=3.5$ s and after this time, the active power produced by the PV generator is almost constant for the duration of entire the co-simulation (e.g., there were no cloud passages).

The data from Savona's plant are received about once per second (0.912 s to be precise) and the value of the active power received is expressed in kW, according to the sign convention of generators. The received power is then multiplied by 1000 to convert kW into W and then sent into the two *Dynamic Load* blocks. For one of the two loads, a gain of 10 has been added.

Since the power received by the PV system is quite constant, to see in Real-Time the dynamic response of the voltage measured at the secondary winding transformer MV/LV a manual switch in the SC_Console of the Simulink model has been added as shown in Figure 4.5 to change the sign of the received power.

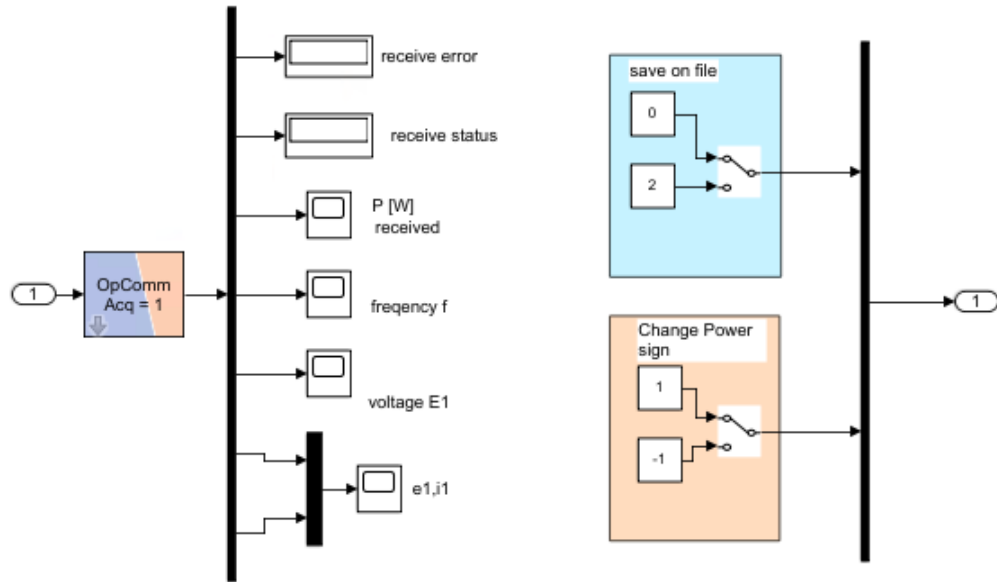


Figure 4.5: SC_Console of the Simulink model used for the co-simulation with Savona

In this way, by manually changing the position of the switch in the SC_Console it is possible to multiply the value of the power received by 1 or -1 and then interpret the received data respectively as load or generator, respectively. A detail on how the incoming power is used is reported in Figure 4.6.

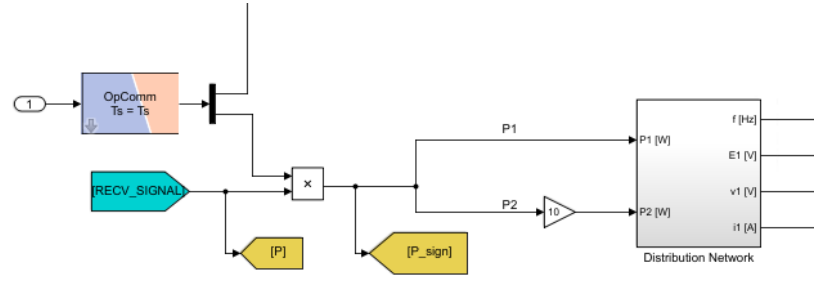


Figure 4.6: Use of the received power in the RT co-simulation with Savona

The result of the co-simulation is presented in Figure 4.7.

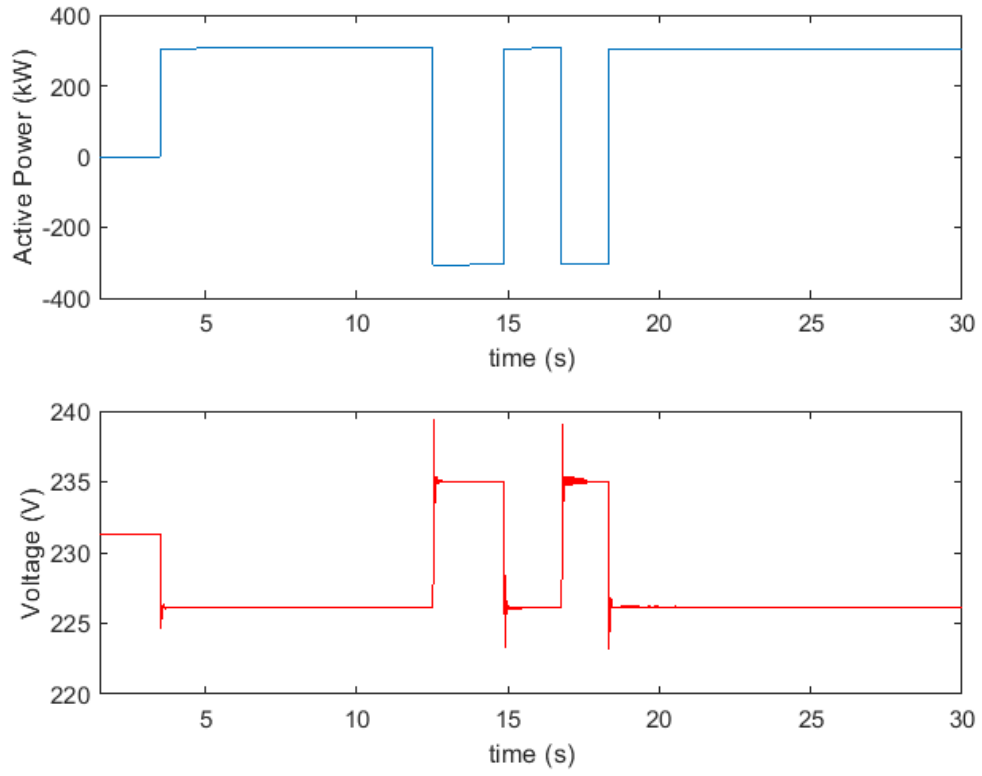


Figure 4.7: Total active power of the two *Dynamic Load* and the voltage response in the RT co-simulation with Savona

Figure 4.7 shows the trend of the active power in blue (expressed in kW) and the variation of the amplitude of the phase voltage (expressed in V) measured at the secondary side of the MV/LV transformer in red, according to the sign and the value of the active power arriving from the Savona's measurements. The response of the phase voltage is as expected. In the first part (before time $t=3.5$ s), the measurements from the remote node are not yet arriving and, therefore, no load was connected to the transformer. Hence, the Voltage remains constant and equal to the base value (230.9 V). Subsequently (after time $t=3.5$ s), when the measurements from Savona's PV system started to arrive, the sign of the power was changed manually, and the measured voltage lowers when the data stream is considered as load and rises in the case when the data stream is considered as generator. The voltage response is a bit under damped because the load variation is abrupt (steps without filters). Since the power values produced by the remote PV system during the test were almost constant, the deviation of the amplitude of the phase voltage with respect to the base voltage is almost symmetrical in the two cases.

4.2 Test with Bari

This simulations has been carried out within a collaboration between Politecnico di Bari (PoliBa) and Politecnico di Torino (PoliTo). This test is based on the implementation of a PHIL model, where the components of a micro-grid located in PoliBa have been used.

The Real-Time co-simulation has been fulfilled by modeling part of the electric network at the PoliTo's facility and the rest at the PoliBa's facility. Specifically, at PoliTo the HV level, the MV level, and part of the LV level have been simulated on software, while at PoliBa the rest of the LV level and the Hardware-in-the-Loop have been implemented [5]. The network schematic is shown in Figure 4.8.

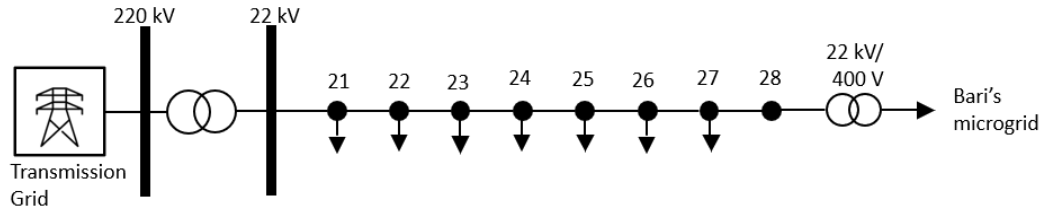


Figure 4.8: Schematic representation of the distributed feeder in the test with Bari

The implementation of the R-PHIL simulation (Remote Power Hardware In the Loop) includes the PoliBa's LabZero facility and the PoliTo's G-RTS LAB facility. The PoliBa's facility is composed by a OPAL-RT[®] 5600 Real-Time digital simulator and a 16 kVA 4-quadrants programmable power source (PMI15A30F60). Such controllable power output can be used to communicate either with a RLC local bank or to exchange active and reactive power with a microgrid, located in the same research facility. In the PoliTo's facility, which is equipped with a long list of device for Real-Time applications, only the digital Real-Time simulator OPAL-RT[®] 5600 has been exploited [5].

Regarding the communication aspects, two approaches were analysed. The first approach, consists in the use of VILLAS node as a communication interface. The second approach, already used in [5], consists in direct communication between the two remote Real-Time simulators. In Figure 4.9 is reported the scheme of the communication infrastructure.

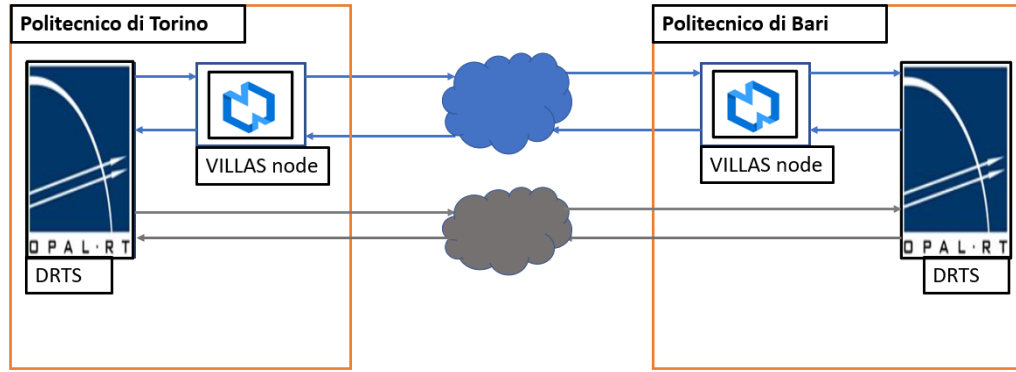


Figure 4.9: Schematic representation of the communication infrastructure in the test with Bari

Figure 4.9 shows with blue arrows the communication with VILLAS node and with grey arrows the direct communication. Unlike of the test with Savona, the communication is bi-directional. The configuration file used in this test is reported in Appendix A.3. Since the sinusoidal signals would be heavily impacted by the data losses, obtaining an unacceptable distortion of the signal itself, an hybrid phasor approach has been used. PoliTo sent to PoliBa, besides the reference frequency f and the RMS value of the phase voltage (V) measured at secondary side of the transformer. Moreover, a loop-back is implemented, with the active power P (W) and the reactive power Q (var) sent back from PoliBa to PoliTo. These values are multiplied by a gain of 100 and then are used to supplied a flexible load.

Please note that the same signal are sent both through VILLAS and with direct communication.

The Simulink model's top level is merely composed by a SC_Console and a SM_Master. Figure 4.10 shows the scheme inserted within the SM_Master.

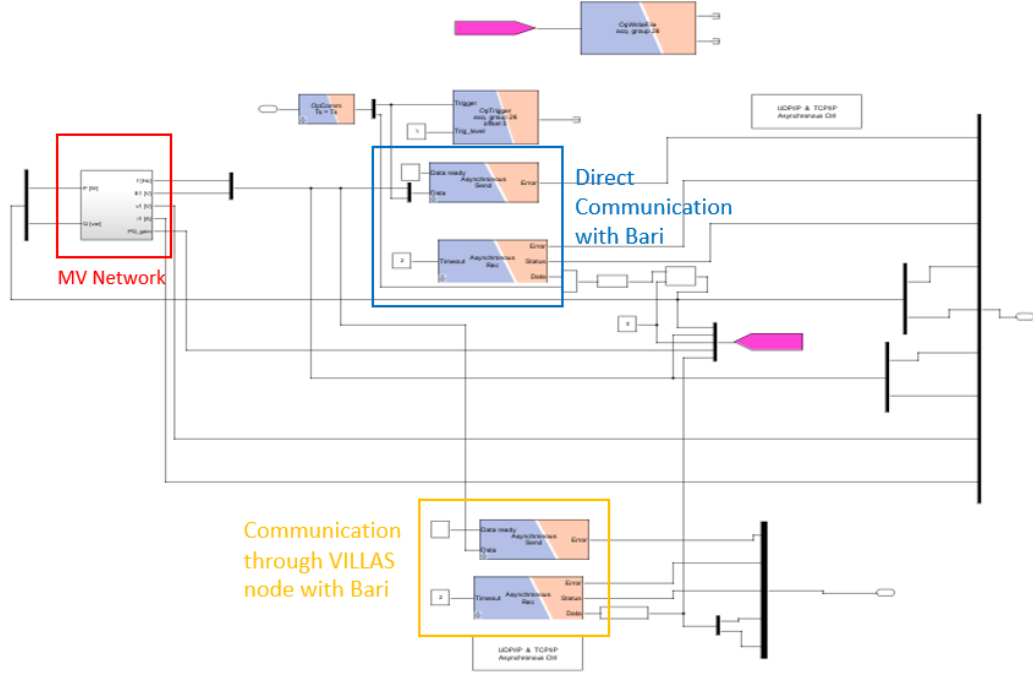


Figure 4.10: SM_Master Simulink model in the RT co-simulation with Bari

The SM_Master block contains the blocks necessary to fulfill the data exchange with PoliBa, the *OpTrigger* to correctly terminate the data saving at a desired instant and the electric grid simulated at PoliTo. Moreover, from Figure 4.10 it can be appreciated the loop-back of P and Q , which are received from PoliBa and sent in the electric grid modeled at PoliTo. Specifically, P and Q amplitudes are sent to the controllable load block simulating the LV flexible resource. From the point of view of the Turin model, the network of Bari is a "black-box" and only the powers absorbed by the RL load are known. For this reason a *Dynamic Load* block is used.

As evidenced by Figure 4.10 two different communication paths were used. The first one directly connects the two remote Real-Time simulators, while the second one also involves the VILLAS nodes configured in PoliTo and PoliBa. The goal is to understand if there were significant differences between the two communication medium.

The results of the remote Real-Time co-simulation with the integration of PHIL are highlighted in Figure 4.11.

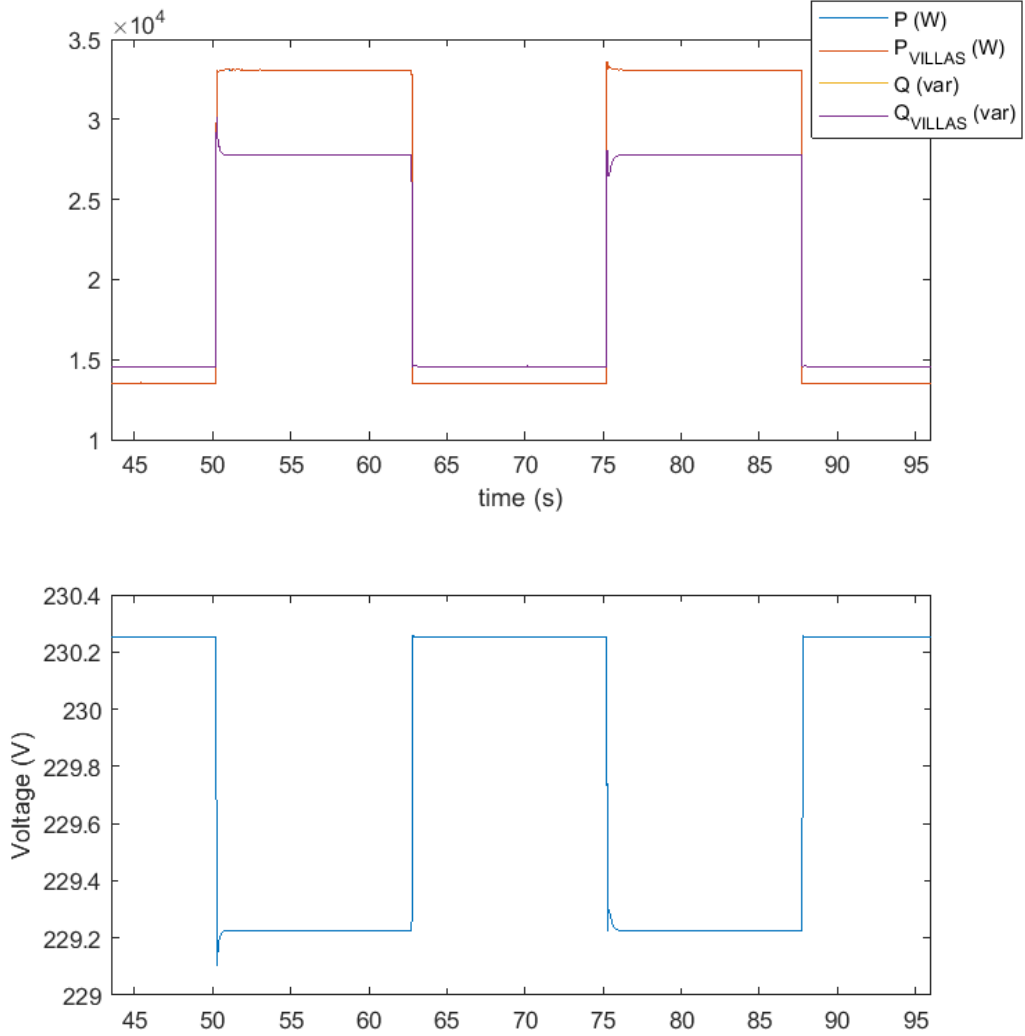


Figure 4.11: First plot: active and reactive power received from Bari. Second plot: voltage response to load variations

The first plot of Figure 4.11 shows the active and reactive powers received both through direct communication and through VILLAS. The second plot of Figure 4.11 refers to the dynamic response of the voltage measured at the secondary side of the MV/LV transformer.

The load is an RL type, so it will absorb an active component and a reactive component of the power. Thanks to the presence of the inductance of the load, the step response of the voltage is smoother than in the case of the co-simulation with Savona. From the test we saw in Real-Time, the variation of the amplitude of the phase voltage as a function of the variation of the power absorbed by the load: the greater is the value of the absorbed power, the greater is the value of the current amplitude and therefore the greater the voltage drop.

Regarding the difference between the case with and without VILLAS node, the differences are imperceptible, as shown in Figure 4.11. By zooming in on the first step of the active power, which occurs at a time of approximately 50 s, the delay added by the communication with the VILLAS nodes can be appreciated.

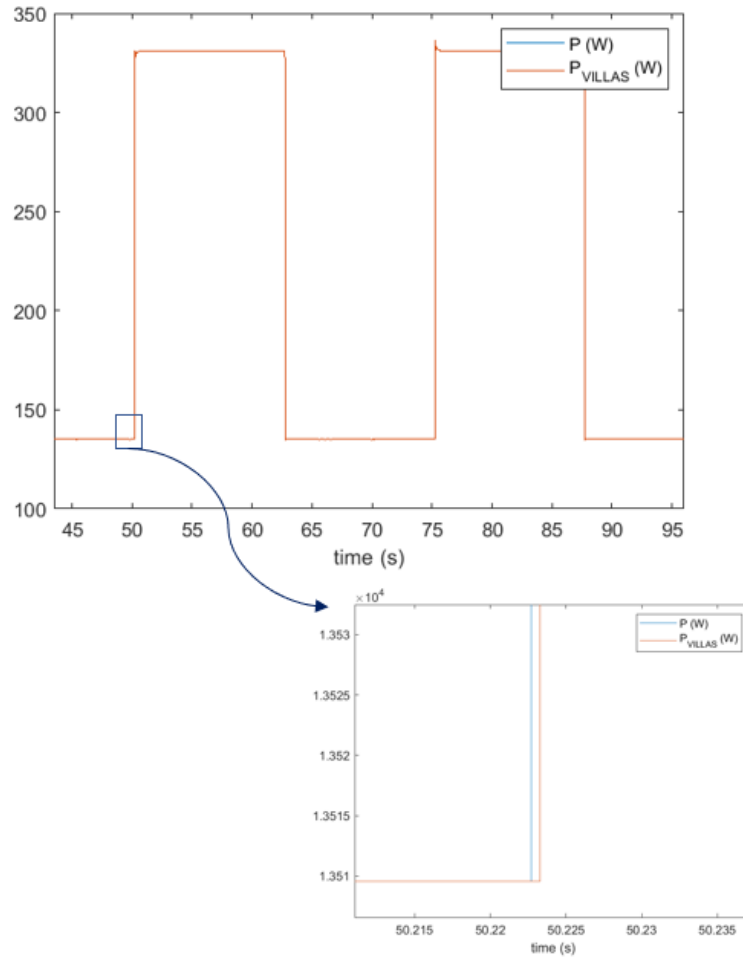


Figure 4.12: Delay introduced by the VILLAS node

Figure 4.12 highlights the delay introduced by the VILLAS nodes. Since the time-step chosen turns out to be $150\text{ }\mu\text{s}$, the delay introduced has occurred to be $600\text{ }\mu\text{s}$, i.e., 4 time-step.

In this test it was tried to highlight how the exchange of phasor quantities during a remote co-simulation can bring great advantages on the reliability of the test compared to the use of waveforms. Please note that despite the time-step being $150\text{ }\mu\text{s}$, the sending-rate is 20 and therefore a data is sent every 3ms. To conclude then the use of phasors might allow larger systems to be simulated in Real-Time with larger time-steps (i.e., milliseconds instead of microseconds) and this is an advantage from the point of view of the reliability and stability of the remote co-simulation.

4.3 Test with Bari and Savona

The remote Real-Time co-simulation proposed in this section involves Politecnico di Torino (PoliTo), Politecnico di Bari (PoliBa) and University of Genova (Campus Savona). In this test are involved two real hardware systems that is the load RL present in PoliBa and the PV generator present in Savona's Campus. Figure 4.13 shows a scheme of the network used for the test.

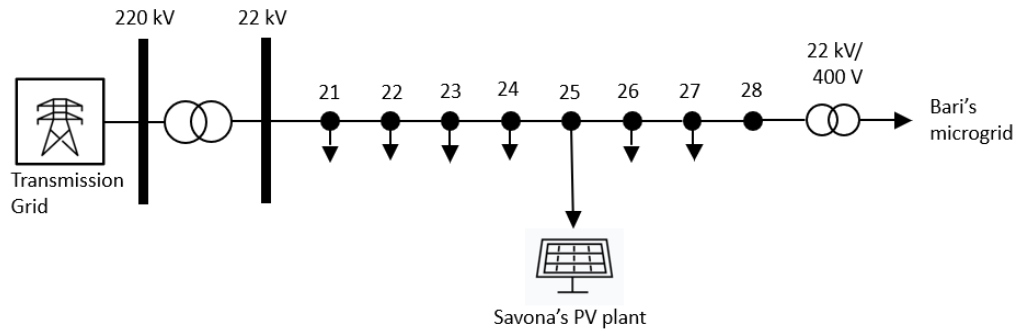


Figure 4.13: Schematic representation of the distributed feeder in the test with Bari and Savona

As in the previous tests, the HV network and a feeder of the MV network are simulated on software at PoliTo, while the PHIL (which includes a variable RL load) is implemented at PoliBa. To include in the simulation also the PV located

in Savona's Campus at node 25 a *Dynamic Load* has been added at node 25 that receives in input the measures of active power in arrival from the PV generator.

During this remote co-simulation, the communication between the various laboratories took place entirely using the VILLAS node. Thanks to the flexibility of the VILLAS node it was possible to interface the model simulated in PoliTo with the hardware components present in PoliBa and the Savona's Campus. For the configuration of the VILLAS node in Turin one can refer to Appendix B.2. The schematic of the communication infrastructure is shown in Figure 4.14.

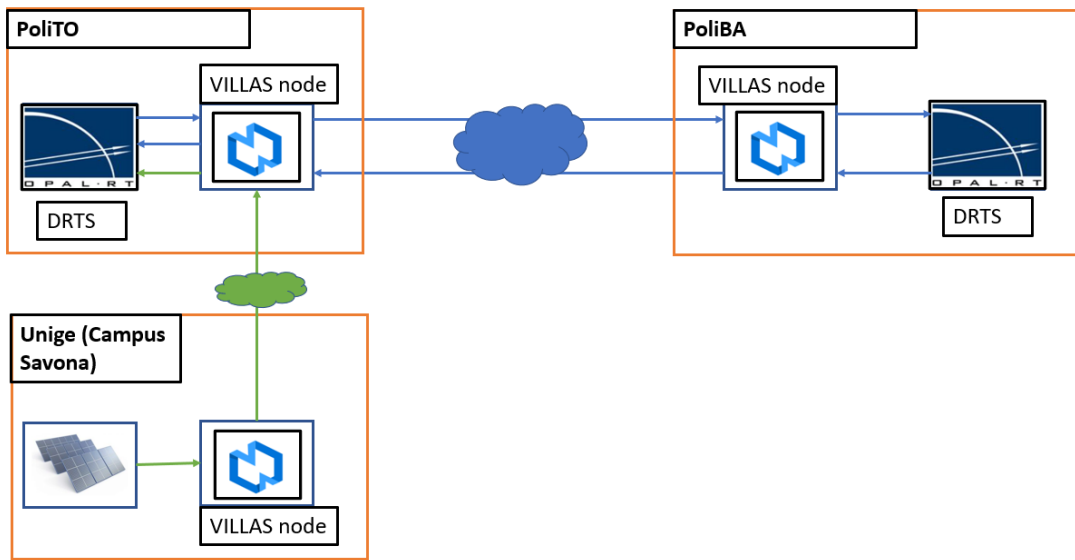


Figure 4.14: Schematic representation of the communication infrastructure in the test with Bari and Savona

This overall test is presented as the conclusion of the two previous tests to establish the connection between the three Italian universities. In this test, therefore, three physical nodes communicate with each other in Real-time and exchange different data. As shown in Figure 4.14 there are two different communication paths:

- **PoliTo-PoliBa** is a bi-directional path, through which the Turin node sends to the Bari node the RMS value of the phase voltage and the frequency necessary to supply the RL load present in Bari and receives from Bari the active and reactive power absorbed by the load.

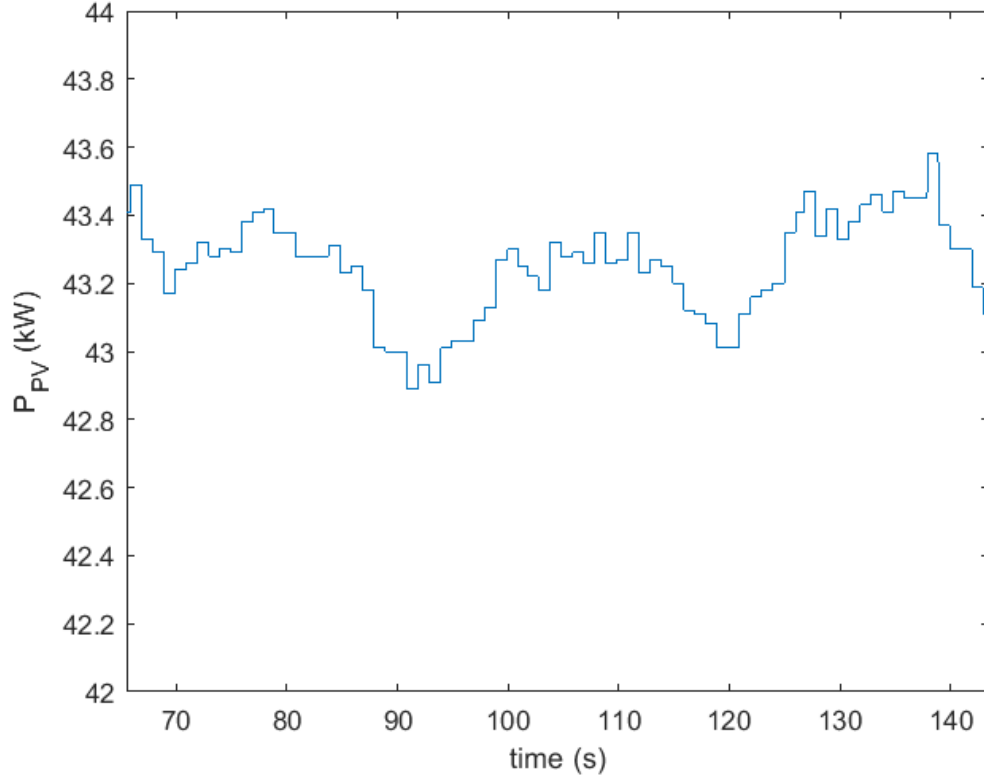


Figure 4.16: Active power produced by the PV generator in Savona

The variation of the active power due to the variation of the load of Bari and the respective response of the voltage are shown in Figure 4.17. Please note that the figures show the data recorded in both PoliTo and PoliBa and that the recording started at the same time and consequently the two trends are the same. However, as the time scale is very long, the two trends seem to overlap.

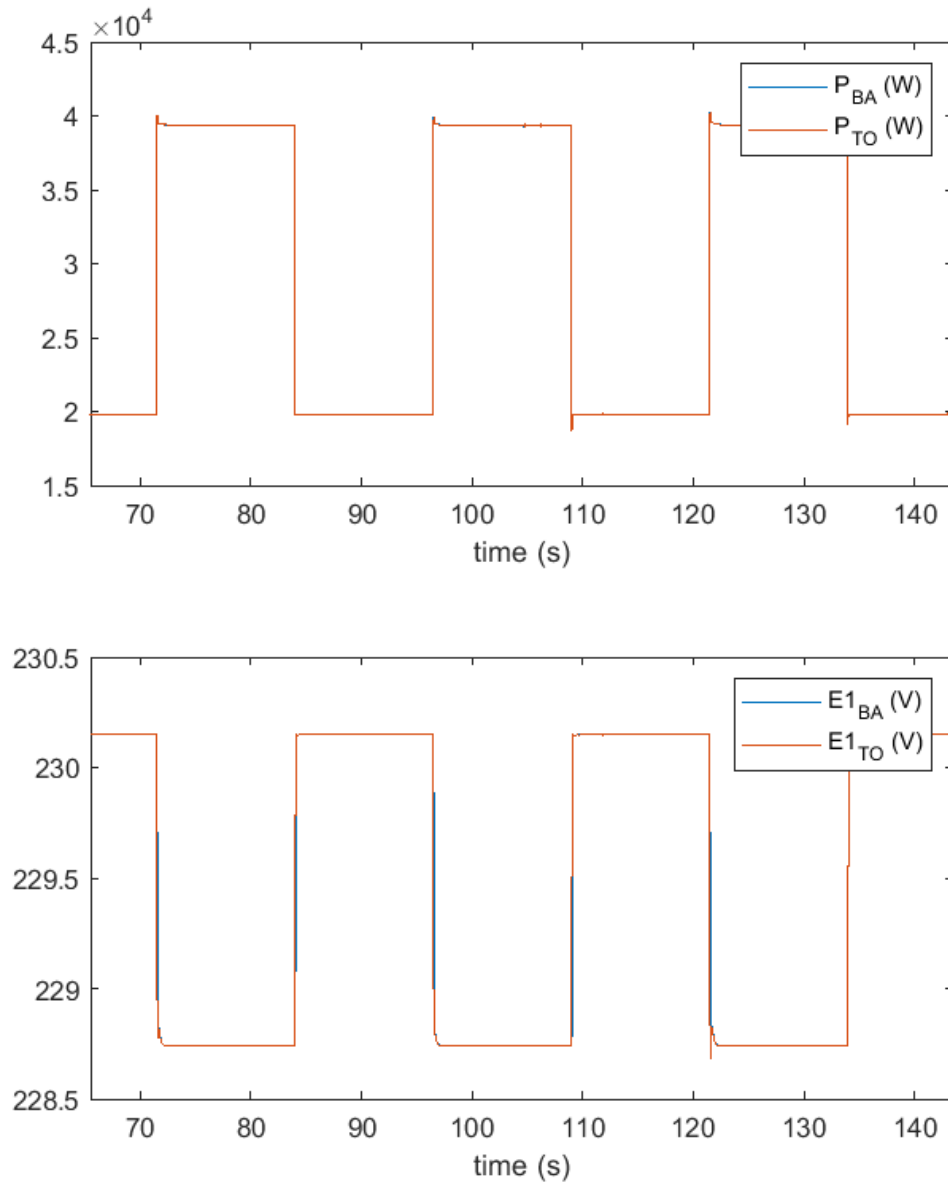


Figure 4.17: Results of the RT co-simulation with Bari and Savona. First plot: active power absorbed by the load in Bari recorded in Bari and Turin. Second plot: Voltage response recorded in Bari and Turin

However, by reducing the time scale, it is possible to see distinctly the powers recorded at the PoliTo and those recorded at the PoliBa and estimate the one-way delay introduced by the communication. The delay in the one-way communication path between PoliBa and PoliTo was observed by zooming in a point where there is a change in active power as shown in Figure 4.18 at a time of about 84 s.

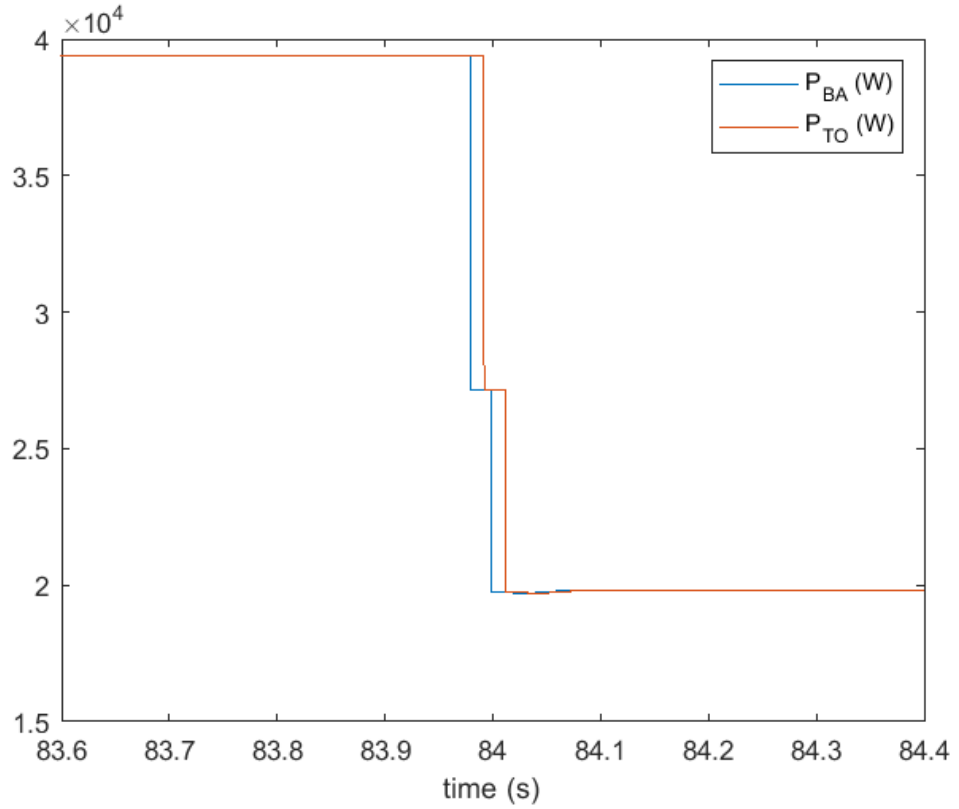


Figure 4.18: Zoom of the load variation recorded in Bari and Turin

It turns out that the one-way delay is about 13.2 ms and therefore, assuming that the total delay is twice the delay of the single path, it appears that the total delay introduced by the communication is 26.4 ms which is a result in line with those found in Section 3.4. Please note that the total communication delay is the sum of several different delays [9] it is highly dependent on congestion and the state of the internet; so, it can not be constant for subsequent tests or for tests performed at different times of the day.

Finally, the dynamic response of the voltage measured in the Turin model compared to the power variation measured in Bari is shown in Figure 4.19.

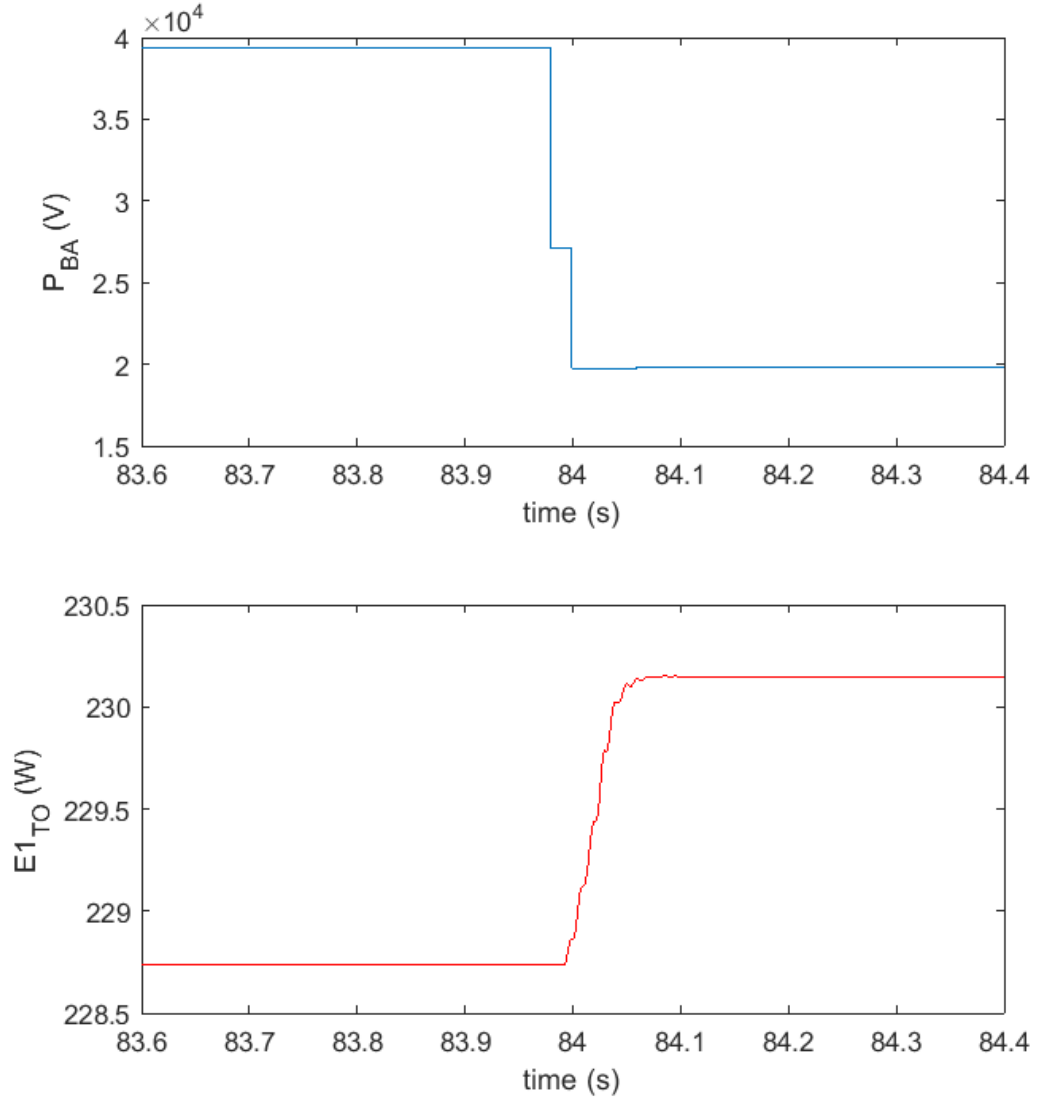


Figure 4.19: Results of the RT co-simulation with Bari and Savona. First plot: Load variation measured in Bari. Second plot: voltage response measured in Turin

The beginning of the voltage response is delayed compared to the beginning of the load variation of 13.2 ms which is the delay of the one-way communication. It is also noted that the voltage response is not a step due to the presence of reactive elements and a low-pass filter inserted in the PoliTo's model in order to reduce the presence of overshoots and make the simulation more stable.

4.4 Tests with Genoa

The two co-simulations presented in this section were conducted thanks to the collaboration between Politecnico di Torino and Università degli Studi di Genova (UniGe). The Real-Time simulator installed in Genoa (speedgoat[®]) and that installed in Turin (OPAL-RT[®] 5600) have been enabled to communicate through the use of the VILLAS node. In contrast to the previous tests, no hardware element, except for the two Real-Time simulators was used in the co-simulation. The communication infrastructure is schematized in Figure 4.20 while the configuration file of the Turin's VILLAS node for these tests is reported in Appendix B.3.

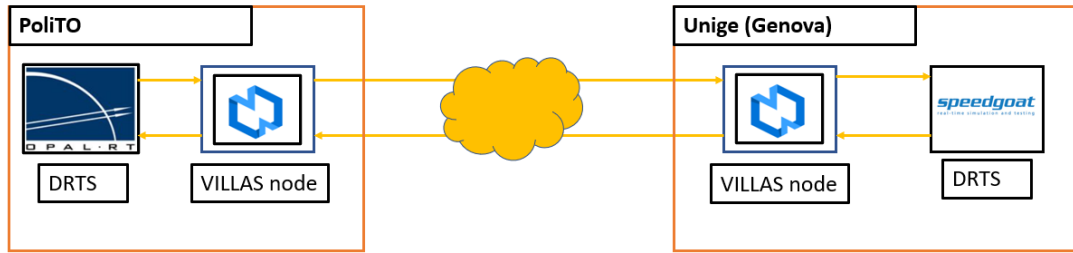


Figure 4.20: Scheme of the communication infrastructure in the tests with Genoa

The first test aims to determine the communication delay between the two remote laboratories. Since it was shown in Section 3.4 that sending sinusoidal signals is not the optimal choice due to the numerous problems affecting the fidelity of the simulation, another type of test was performed for the sole purpose of knowing the delay introduced from the communication between the two universities. To perform the test, two steps were sent from UniGe to PoliTo and re-sent to UniGe without any variation. The first step is a positive step and occurs at time $t = 30$ s of the Unige's model. The second step is a negative step and occurs at time $t = 40$ s. The same step are received by UniGe after the loop-back respectively at the times $t = 40.01$ s. In this way it was possible to define the delay by comparing the step sent from UniGe and the one received after the loop-back, which is about 10 ms. Figure 4.21 shows the results of the test where the blue graph represents the signal sent by UniGe and in orange the same signal received by PoliTo. The overall delay is lower than the one related to the test with Bari as the two universities are geographically closer.

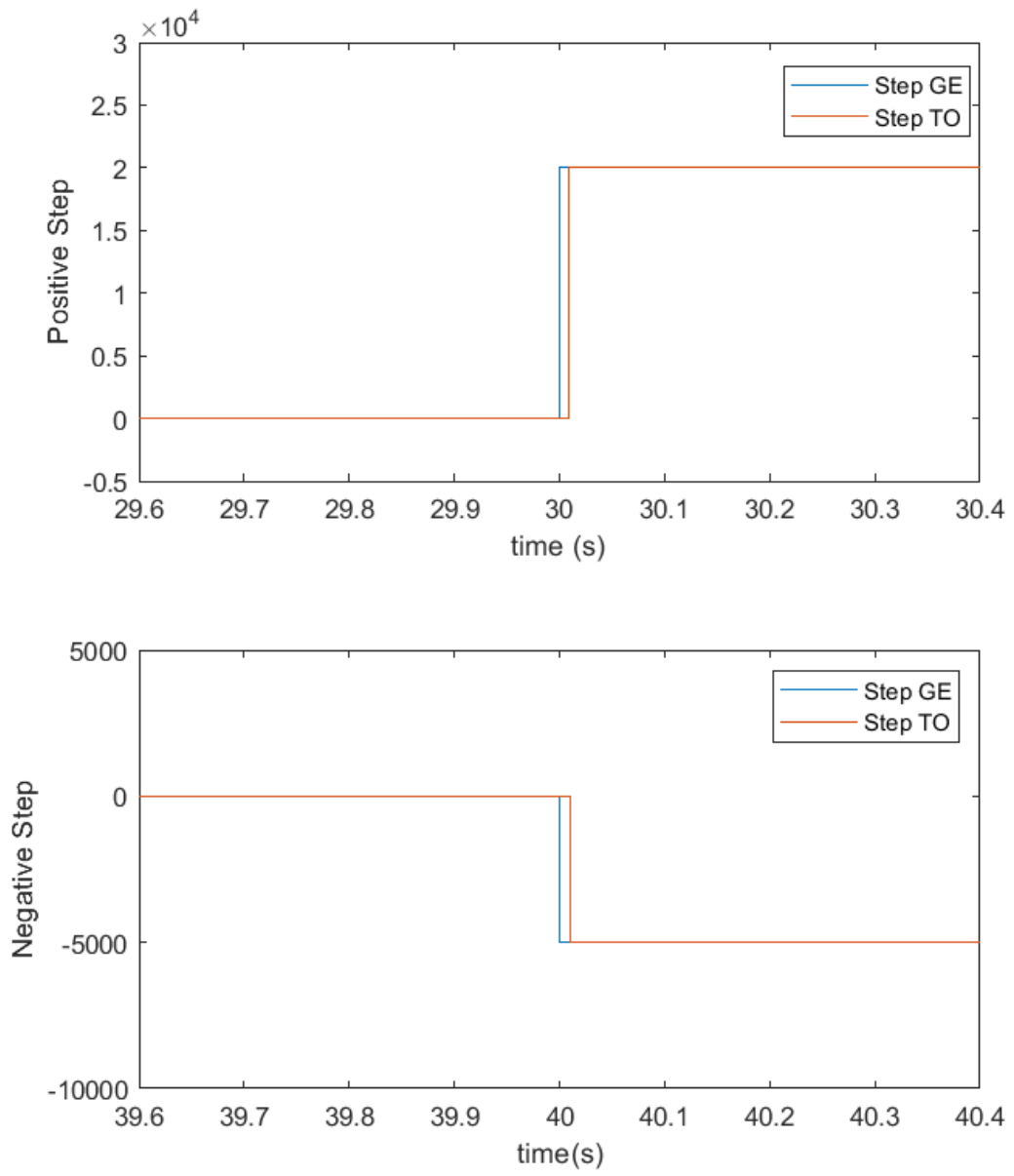


Figure 4.21: RT co-simulation with Genoa. First plot: positive step. Second plot: negative step

The second test was conducted using the network model already used in the previous tests with Bari and Savona with some minor changes. In particular, the Unige model included the model of a wind generator and was inserted in the node 28 of the MV network modeled in PoliTo. The network schematic used for this test is shown in Figure 4.22.

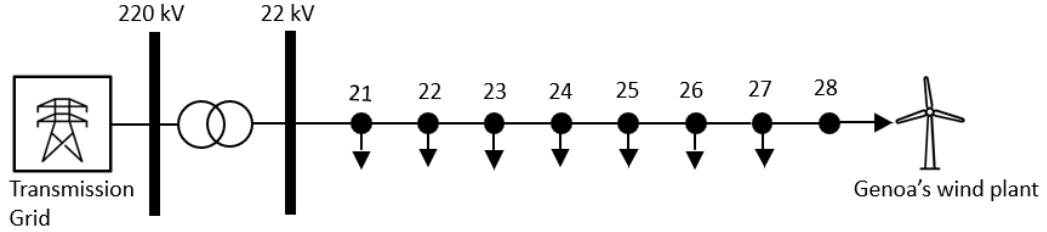


Figure 4.22: Schematic representation of the distributed feeder in the test with Genoa

As for the previous co-simulation that involves the hardware components of the PoliBa, also in this case the communication path is bi-directional. To work, the Genoa's wind generator model needs the values of the amplitude of the phase-to-phase voltage and frequency measured in the node in which it is inserted. To obtain the RMS value of the phase-to-phase voltage, a *Three Phase V-I Measurement* block was inserted upstream of the *Dynamic Load* block that is used to emulate the wind generator. Definitely, the phase-to-phase voltage and the frequency are the signals sent from PoliTo to UniGe, while the signals received in PoliTo from Unige are the active and reactive powers produced by the wind system. The SM_Master of the model used for the remote co-simulation with Genoa is shown in Figure 4.23. As can be seen from Figure 4.23, only communication through the VILLAS nodes was used.

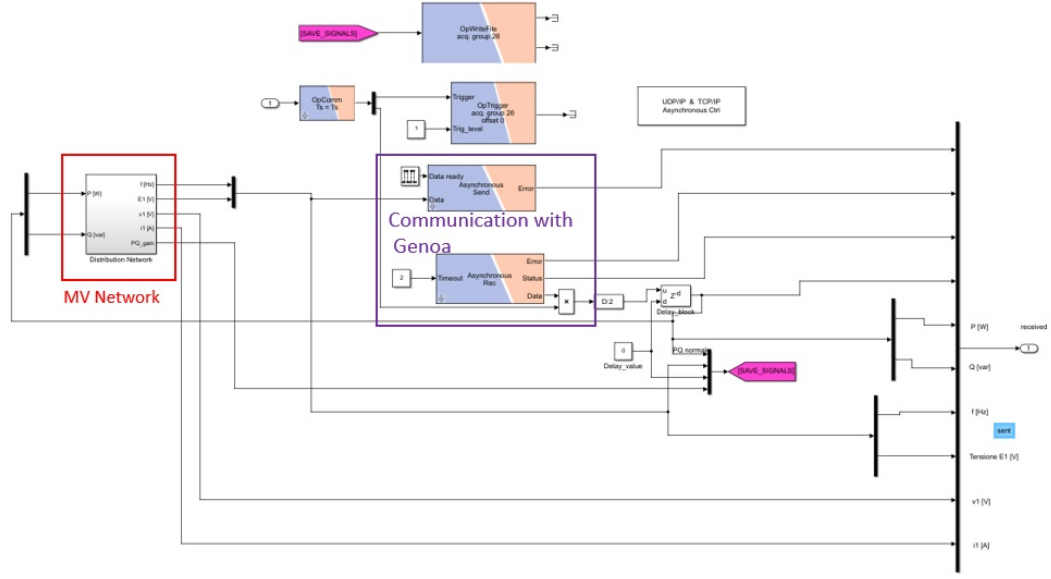


Figure 4.23: SM_Master in the test with Genoa

The results of the test are proposed below. Figure 4.24 shows the trend of the active and reactive powers generated by the wind system. These power trends are derived from a real wind profile over a certain time interval. At the start of the simulation, before the simulation in UniGe was launched, the active and reactive power values were zero. When the simulation of UniGe is launched, the value of the active power starts to increase as a function of the wind value, while the reactive power goes to a very small negative value. To adapt them to the user convention and to be able to use them to control the *Dynamic Load* block, the values of the two received powers were multiplied by a gain of -1.

The dynamic response of the phase-to-phase voltage measured upstream the wind generator is shown in Figure 4.25. As shown in Figure 4.25 at the start of the simulation when the active and reactive power of the wind generator are zero, the RMS value of the phase-to-phase voltage is equal to the base voltage, while it increases and follows the trend of the active power produced by the wind generator when the Unige's simulation is started and then the data are received.

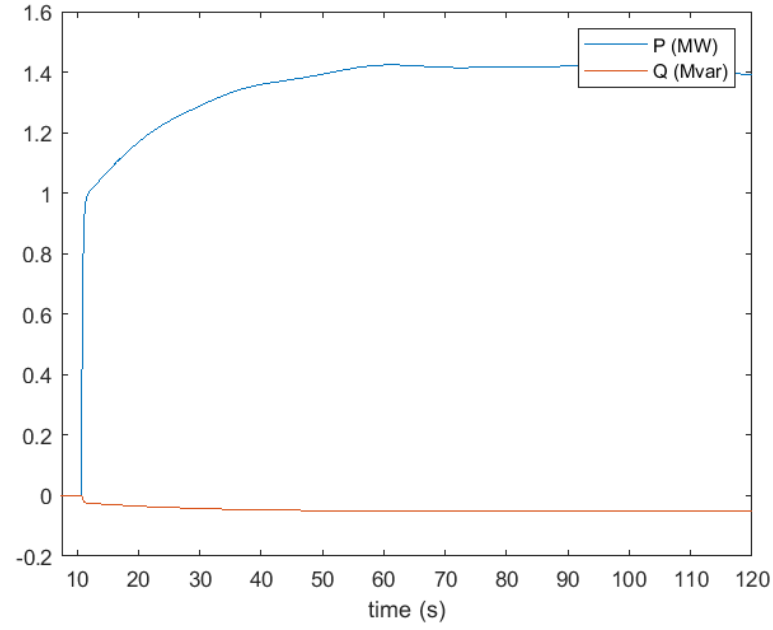


Figure 4.24: RT co-simulation with Genoa: active and reactive power produced by the Genoa's wind generator

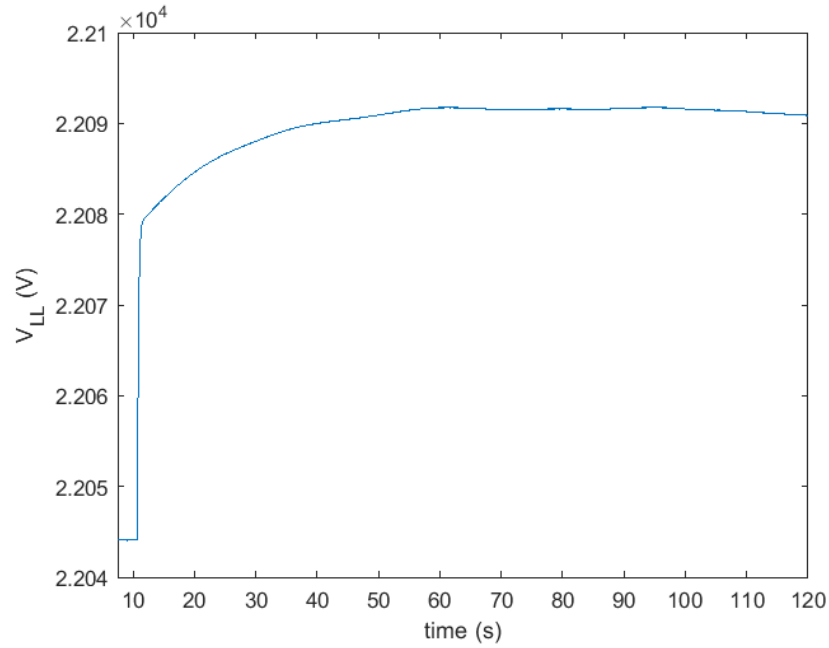


Figure 4.25: RT co-simulation with Genoa: dynamic behavior of the phase-to-phase voltage

Chapter 5

High Voltage Transmission Network

The high voltage transmission network modelled in this thesis is based on a physical HV network located in the North America covering areas of Manitoba, North Dakota, and Minnesota. Compared with the original network, the number of nodes has been reduced such that the essential characteristics are maintained [26]. In particular, the configuration of the network adapted to the European model was used. Some of the most important features of the European network configuration are reported:

- *Structure:* The network transmission voltages used are 220 kV and 380 kV, which are typical in European transmission systems. Generation bus voltages are 22 kV and the rated system frequency is 50 Hz.
- *Symmetry:* This is a balanced three-phase HV transmission network. Ideal line transposition is assumed.
- *Line types:* Overhead lines are constructed with stranded aluminum conductors reinforced with a steel core. The 220 kV lines have one conductor per bundle and the 380 kV line has two conductors per bundle. A standard overhead transmission line tower structure is used. It provides sufficient clearance from ground and between conductors to ensure adequate immunity to switching over-voltages while attempting to minimize the line inductances.
- *Grounding:* It is assumed that ground wires are solidly grounded [26].

5.1 Topology of the high voltage transmission network

The topology of the European like HV transmission network is shown in Figure 5.1. The network consists of 12 buses (bus 6a and bus 6b are considered the same bus) and covers three geographical areas, indicated as area 1, 2, and 3, marked by dashed lines. Area 1 is predominantly a generation center. Area 2, situated about 500 km from Area 1, is a load center with a small amount of generation available. Area 3 lies between the main generation Area 1 and the main load center Area 2 [26].

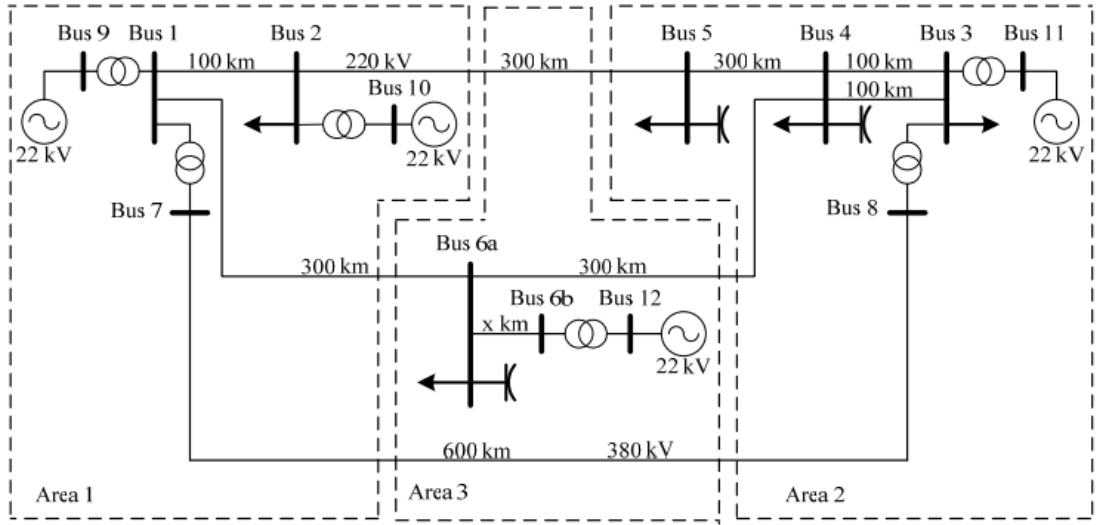


Figure 5.1: Topology of European HV transmission network [26]

Three voltage levels exist in the network: generation bus voltage of 22 kV, primary transmission high voltage of 220 kV, and the long line connecting Area 1 and Area 2 at the extra-high voltage (EHV) level of 380 kV.

5.2 High voltage network data

5.2.1 Branch data

In this section the data and the parameters of the transmission line and the power transformers are reported into a number of tables. In Table 5.1, the base values for the per unit system used are provided. The given base power applies to all three phases and the base voltage is a line-to-line voltage. Table 5.2 gives the resistance,

reactance and susceptance values of the transmission lines and Table 5.3 gives the transformers parameters. All parameters, expressed in p.u., refer to the basic quantities in the Table 5.1.

Table 5.1: Per unit system base values of European HV transmission network [26]

$N.$	S_{base}	V_{base}	I_{base}	Z_{base}
$phases$	(MVA)	(kV)	(kA)	(Ω)
3	100	220	0.262	484
3	100	380	0.152	1444

Table 5.2: Line parameters of European HV transmission network [26]

$Node$ $from$	$Node$ to	R_{ph} (pu/km)	X_{ph} (pu/km)	B_{ph} (pu/km)	R_0 (pu/km)	X_0 (pu/km)	B_0 (pu/km)	l (km)	V (kV)	S_{rated} (MVA)
1	2	1.35e-4	8.22e-4	1.38e-3	4.41e-4	2.72e-3	8.06e-4	100	220	250
1	6	1.35e-4	8.22e-4	1.38e-3	4.41e-4	2.72e-3	8.06e-4	300	220	250
2	5	1.35e-4	8.22e-4	1.38e-3	4.41e-4	2.72e-3	8.06e-4	300	220	250
3	4	1.35e-4	8.22e-4	1.38e-3	4.41e-4	2.72e-3	8.06e-4	100	220	250
3	4	1.35e-4	8.22e-4	1.38e-3	4.41e-4	2.72e-3	8.06e-4	100	220	250
4	5	1.35e-4	8.22e-4	1.38e-3	4.41e-4	2.72e-3	8.06e-4	300	220	250
4	6	1.35e-4	8.22e-4	1.38e-3	4.41e-4	2.72e-3	8.06e-4	300	220	250
7	8	2.27e-5	2.16e-4	5.21e-3	1.25e-4	8e-4	3.64e-3	600	380	500

Table 5.3: Transformers parameters of European HV transmission network [26]

<i>Node from</i>	<i>Node to</i>	<i>Connection</i> (3-ph)	V_1 (kV)	V_2 (kV)	X_{tr} (pu)	X_{gnd} (Ω)	S_{rated} (MVA)
1	7	YNyn0	220	380	0.013	3	1000
9	1	YNyd11	22	220	0.013	3	1000
10	2	YNyd11	22	220	0.013	3	1000
3	8	YNyn0	220	380	0.013	3	1000
11	3	YNyd11	22	220	0.013	3	1000
12	6	YNyd11	22	220	0.026	3	500

5.2.2 Load data

Table 5.4 shows the parameters of the loads at each node expressed in the per unit system. The values given in the table represent maximum active and reactive powers at each bus ($S_{base}=100$ MVA).

Table 5.4: Load parameters [26]

<i>Node ID</i>	P_{MAX} (pu)	Q_{MAX} (pu)
1	0	0
2	2.85	2.0
3	3.25	2.44
4	3.26	2.44
5	1.03	0.62
6	4.35	2.96
7	0	0
8	0	0
9	0	0
10	0	0
11	0	0
12	0	0

5.2.3 Generator data

Parameters of the generator model and operation data are presented in Table 5.5. Table 5.6 identifies the system reference bus ($S_{base} = 100$ MVA). All generators are rated at a phase-to-phase voltage of 22 kV and a frequency of 50 Hz.

Table 5.5: Generators parameters of European HV transmission network [26]

<i>Node</i>	<i>H</i>	δ	X_d	X'_d	X''_d	T'_{d0}	T''_{d0}	X_q	X''_q	T''_{q0}	S_{rated}	P_{out}	V_{out}
<i>ID</i>	(s)	(pu)	(pu)	(pu)	(pu)	(s)	(s)	(pu)	(pu)	(s)	(MVA)	(MW)	(pu)
10	5.0	1.0	1.25	0.333	0.292	5.0	0.002	1.0	0.292	0.002	700	500	1.03
11	3.0	0.0	1.667	0.25	0.233	6.0	0.002	1.125	0.225	0.002	500	200	1.03
12	5.0	1.0	1.25	0.333	0.292	5.0	0.002	1.0	0.292	0.002	500	300	1.03

Table 5.6: Reference bus (slack) of European HV transmission network [26]

Node	Bus voltage angle	Bus voltage magnitude
ID	(deg)	(pu)
9	0	1.03

5.2.4 Shunt capacitor data

Fixed capacitor banks provide voltage support at buses 4,5 and 6. Reactive power injection at the rated voltage for each of those buses is given in Table 5.7.

Table 5.7: Shunt capacitive compensation of European HV transmission network [26]

<i>Node</i>	Q	V_{rated}
<i>ID</i>	(pu)	(kV)
4	1.6	220
5	0.8	220
6	1.8	220

5.3 Implementation of the Simulink model of the European HV transmission network

This section describes the steps and models used for the implementation of the European like HV transmission network in the Simulink environment. The model of the lines, transformers, generators, exciters, steam turbines, capacitive shunts and loads are presented. The ultimate goal is to build the HV transmission network model in the OPAL-RT[®] simulator environment in order to run a co-simulation in which all the universities involved in the project can participate.

5.3.1 Line model

In the distribution networks, the various nodes composing the network are connected to each other typically by short lines (km / tens of km). For this reason, the propagation constants are often negligible and the classic model of the Π line, with lumped parameters, can be used. In the case of very long lines, which are typical of HV transmission networks, it is necessary to take into account the propagation delay of the signal through a suitable model. For this reason, the lines have been represented as distributed parameters lines with a model based on the Bergeron's traveling wave method used by the Electromagnetic Transient. The Bergeron model should generally be chosen over an equivalent Π section whenever the line is long enough to not allow the propagation of waves (at approximately the speed of light) to travel along the entire length of the line within a single time step. This means that for a typical simulation time step of $t_s = 50 \mu\text{s}$, transmission systems roughly over 15 km in length should be represented by the Bergeron model as opposed to a Π section. The advantages are of course that a real world propagation delay is considered in the Bergeron model, and also it is possible to avoid the artificial resonances introduced by consequent Π sections connected in series at high frequencies [27]. Figure 5.2 shows the equivalent impedance model of the line between two generic k and m terminals.

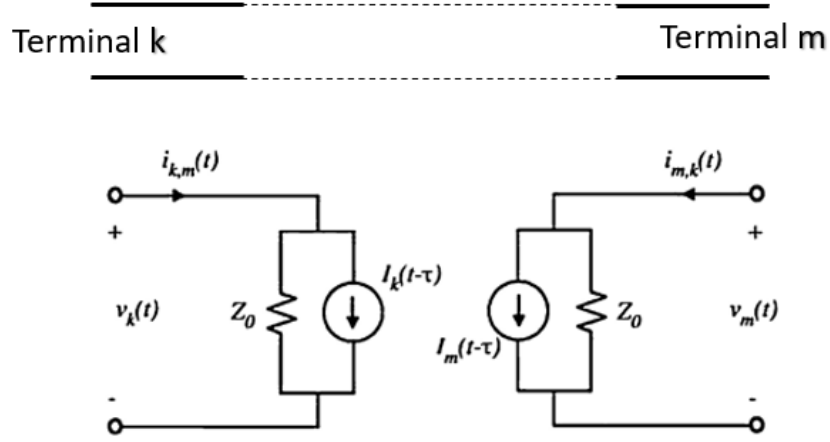


Figure 5.2: Equivalent impedance network between two terminals [27]

Considering the loss-less model, the line is characterized by two parameters: the surge impedance Z_0 (Equation 5.1) and the wave propagation speed ν (Equation 5.2).

$$Z_0 = \sqrt{\frac{l}{c}} \quad (5.1)$$

$$\nu = \frac{1}{\sqrt{lc}} \quad (5.2)$$

where l and c are respectively the inductance per unit of line length (H/km) and the capacitance per unit of line length (F/km). So it is possible to determine the travel time τ from one terminal of the line to the other one from Equation 5.3 knowing the length of the line d [27].

$$\tau = \frac{d}{\nu} = d\sqrt{lc} \quad (5.3)$$

To determine the value of the propagation delay, it is necessary to know the value of the inductance per unit of length (H/km) and of the capacitance per unit of length (F/km) of the lines. From Table 5.2, inductive reactance and capacitive susceptance are expressed in (p.u./ km). Table 5.8 shows the calculated results with the appropriate units of measurement and the calculated value of the propagation delay of the lines.

Table 5.8: Line parameters of European HV transmission network and the propagation delay

<i>Node from</i>	<i>Node to</i>	R_{ph} (pu/km)	X_{ph} (pu/km)	B_{ph} (pu/km)	r_{ph} (Ω /km)	l_{ph} (H/km)	c_{ph} (F/km)	l (km)	τ (ms)
1	2	1.35e-4	8.22e-4	1.38e-3	0.06534	1.267032e-3	9.08038e-9	100	0.339192
1	6	1.35e-4	8.22e-4	1.38e-3	0.06534	1.267032e-3	9.08038e-9	300	1.017576
2	5	1.35e-4	8.22e-4	1.38e-3	0.06534	1.267032e-3	9.08038e-9	300	1.017576
3	4	1.35e-4	8.22e-4	1.38e-3	0.06534	1.267032e-3	9.08038e-9	100	0.339192
3	4	1.35e-4	8.22e-4	1.38e-3	0.06534	1.267032e-3	9.08038e-9	100	0.339192
4	5	1.35e-4	8.22e-4	1.38e-3	0.06534	1.267032e-3	9.08038e-9	300	1.017576
4	6	1.35e-4	8.22e-4	1.38e-3	0.06534	1.267032e-3	9.08038e-9	300	1.017576
7	8	2.27e-5	2.16e-4	5.21e-3	0.327788	0.993325e-3	11.4906e-9	600	2.2027063

To represent lines with distributed parameters according to the Bergeron's model, in the Simulink environment, the *Distributed Parameters Line* block are used. The block is shown in Figure 5.3. The parameters inserted in the blocks for the different lines are those obtained from Table 5.8.

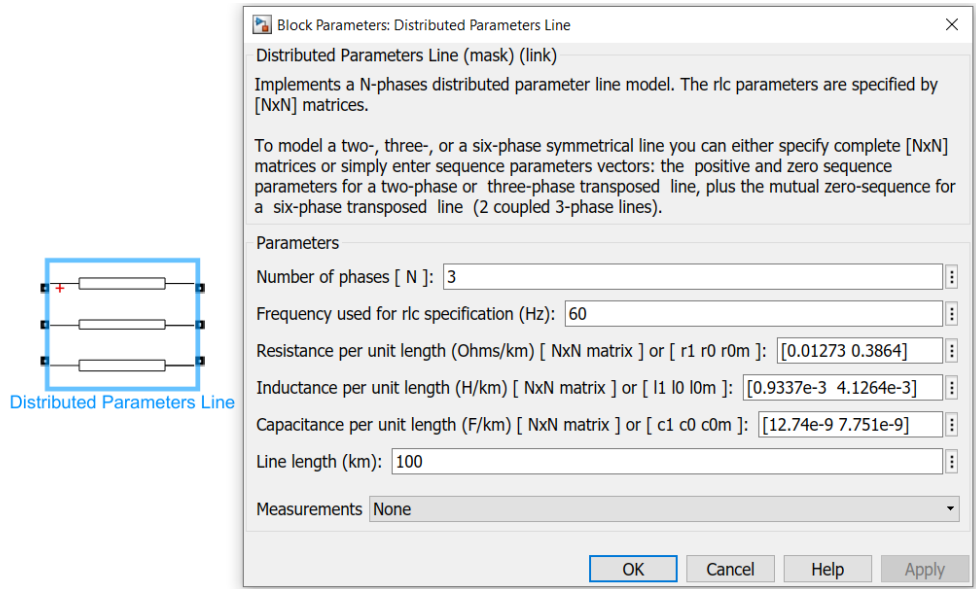


Figure 5.3: *Distributed Parameters Line* block and required parameters

5.3.2 Transformer model

In the HV transmission network, presented in Figure 5.1, there are several transformers to adapt the amplitude of the output voltage from lines and generators to the amplitude of the rated voltage of the different nodes. There are two different types of transformers: the first is used to raise the rated value of the output voltage from the generators (22 kV) to the value of the buses voltage (i.e. 220 kV), while the second one is used to adapt the rated value of the voltage of the buses to the nominal value of the voltage of the line 7-8 (i.e., 380 kV).

The model used to represent the various transformers is represented by the *Three-Phase Transformer Inductance Matrix Type* Simulink block. This block, shown in Figure 5.4, allows modeling a three phase single core transformer.

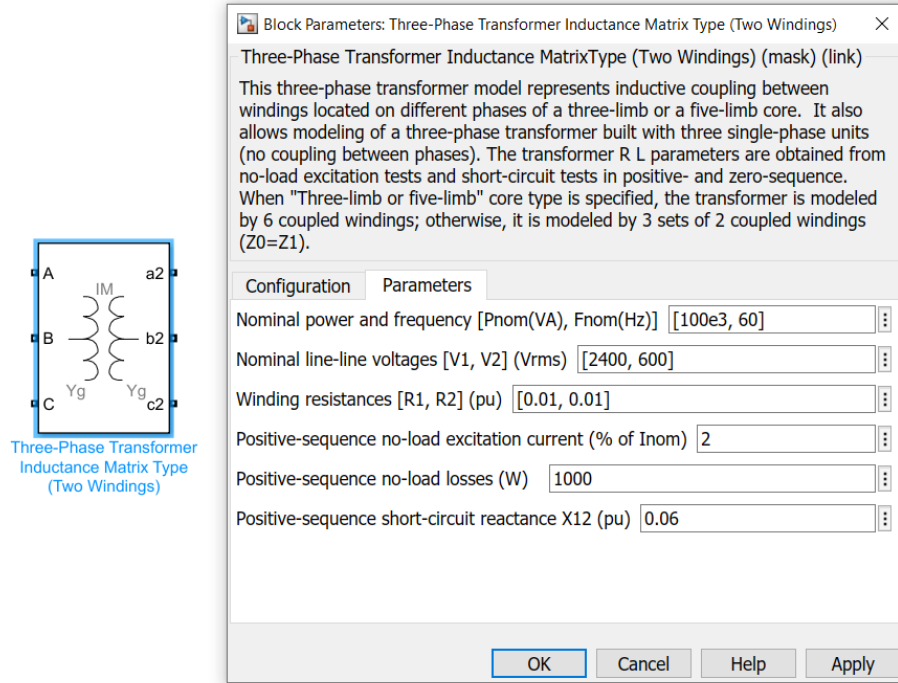


Figure 5.4: *Three-Phase Transformer Inductance MatrixType* block and required parameters

The parameters used for the various transformers are those shown in the Table 5.3. In particular, the transformers that connect node 1 to node 7 and the transformer that connects node 3 to node 8 have the windings connected YNyn0, i.e., both the primary winding and the secondary winding are star-connected with the grounded neutral and the phase shift between the primary and secondary voltage is 0. The other transformers, i.e., those that connect the generators to buses,

have the winding connected YNd11, i.e., the primary winding is star connected with the grounded neutral while the secondary winding is delta-connected and the phase shift between the primary and the secondary voltage is 330 degrees (that is 30×11). Regarding the grounding impedance, a resistance with a value of 3Ω was considered. The two different types of transformers are shown in Figure 5.5.

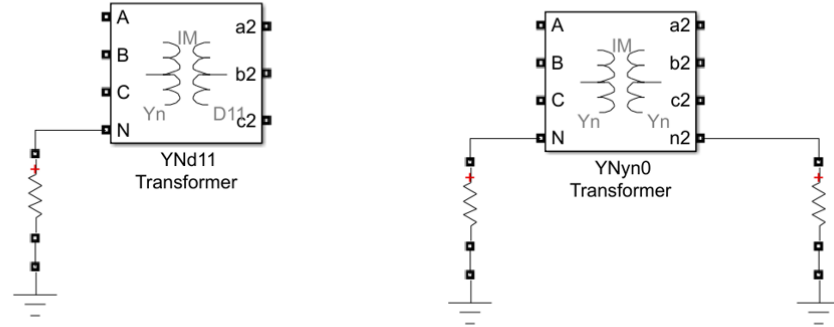


Figure 5.5: On the right the transformer connected YNd11. On the left the transformer connected YNyn0

5.3.3 Generator model

In the HV transmission network adopted in this thesis, there are three generators: generator 2, generator 3 and generator 4. They are connected to nodes 10, 11, 12 respectively. The electric models used to represent these generators are a six-order state-space models and the equivalent circuits for the d-axis and q-axis respectively are shown in Figure 5.6.

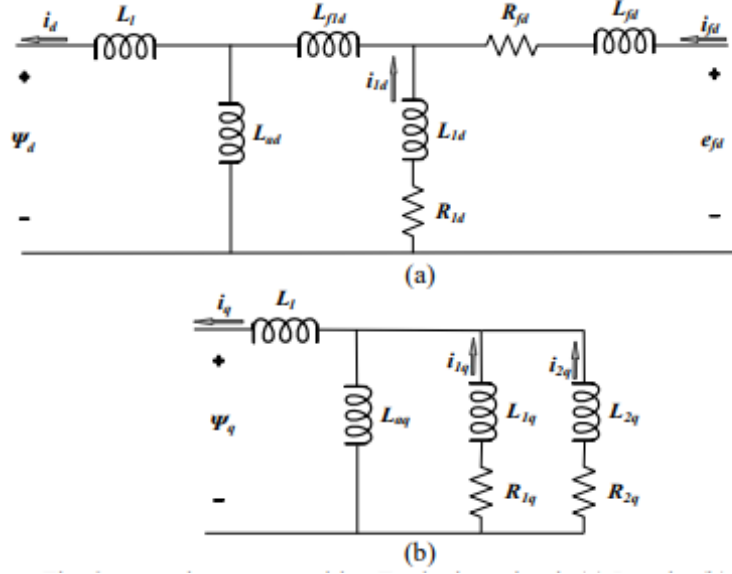


Figure 5.6: Figure (a): synchronous machine equivalent d-axis circuit. Figure (b): synchronous machine equivalent q-axis circuit [28]

To describe the equations of the synchronous generator motion and therefore the mechanical model it is necessary to introduce the following definitions:

- P_m, T_m : the power and mechanical torque generated by the prime mover.
- P_e, T_e : the electrical power supplied to the network by the synchronous machine and the electromagnetic torque.
- $P_a = P_m - P_e$: the accelerating or decelerating power.
- H the inertia coefficient of the synchronous machine and prime mover.
- $\theta(t)$: the angular abscissa of the rotor with respect to a fixed reference axis.
- $d\theta(t)/dt, d^2\theta(t)/dt^2$: the angular speed and angular acceleration of the rotor.
- ω_0/p : the rated angular speed of the rotor (ω_0 = electric pulse, p = pole pairs).

- $\delta(t) = p\theta(t) - \omega_0 t$: load angle between the rotor of the synchronous machine and the rotor of a machine of infinite power rotating at constant speed ω_0 .

The motion of the rotor of the synchronous machine is governed by the Equation 5.4.

$$H \frac{d^2\theta}{dt^2} = T_m - T_e \quad (5.4)$$

The Equation 5.4 can be explained as a function of mechanical power P_m and electric power P_e by multiplying both its terms by $d\theta/dt$, obtaining Equation 5.5 [29].

$$H \frac{d^2\theta}{dt^2} \frac{d\theta}{dt} = T_m \frac{d\theta}{dt} - T_e \frac{d\theta}{dt} = P_m - P_e = P_a \quad (5.5)$$

To represent on Simulink the generators with the sixth-order electrical model shown in Figure 5.6 and from mechanical Equations 5.4 and 5.5, the *Synchronous Machine pu Standard* Simulink block has been used. The Simulink block and the required parameters are shown in Figure 5.7.

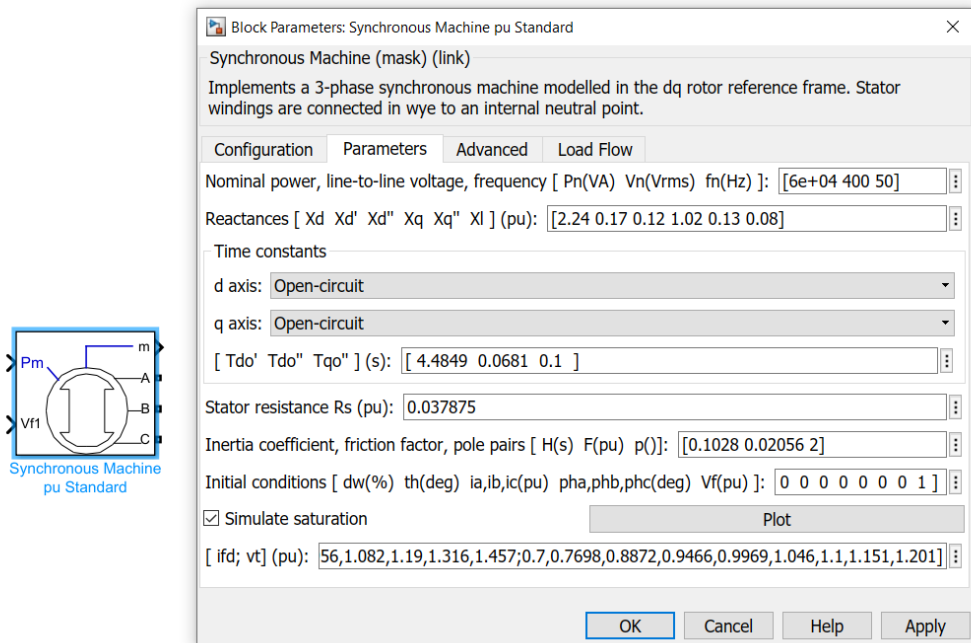


Figure 5.7: *Synchronous Machine pu Standard* and the required parameters

The parameters used for the Simulink implementation are those provided by Table 5.5.

As for the slack (bus 9), in [26] it is considered as a generator of infinite power. In this thesis the generator of infinite power is replaced with a synchronous generator represented by the simplified model in which the electrical system for each phase consists of a voltage source in series with an RL impedance, which implements the internal impedance of the machine, as shown in Figure 5.8. The simplified generator is connected to bus 9 and it is identified as generator 1.

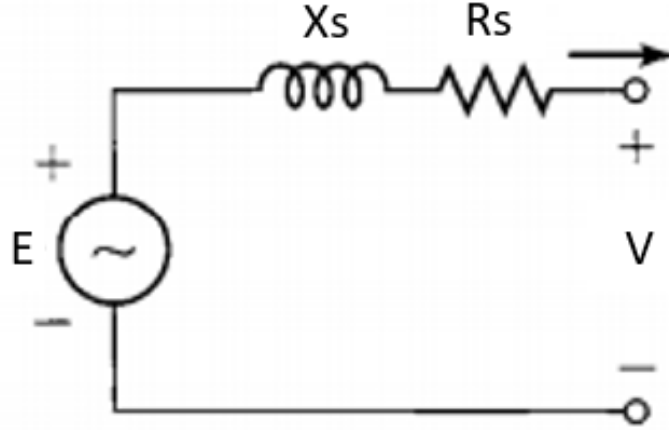


Figure 5.8: Simplified model of the Synchronous generator [30]

The mechanical equations that govern the dynamics of the generator are those reported above (Equation 5.4 and Equation 5.5). In [26] only the parameters shown in the Table 5.6 are provided for the slack; therefore, for completeness, the additional parameters used in this thesis are reported in Table 5.9.

Table 5.9: Parameters of the simplified synchronous generator

<i>Node</i>	<i>H</i>	<i>R_s</i>	<i>X_s</i>	<i>S_{rated}</i>	<i>V_{out}</i>
ID	(s)	(pu)	(pu)	(MVA)	(pu)
9	5.0	0.02	0.3	700	1.03

To represent the simplified model of the synchronous generator in the Simulink environment the *Simplified Synchronous Machine pu Units* are used. The Simulink block and the required parameters are shown in Figure 5.9.

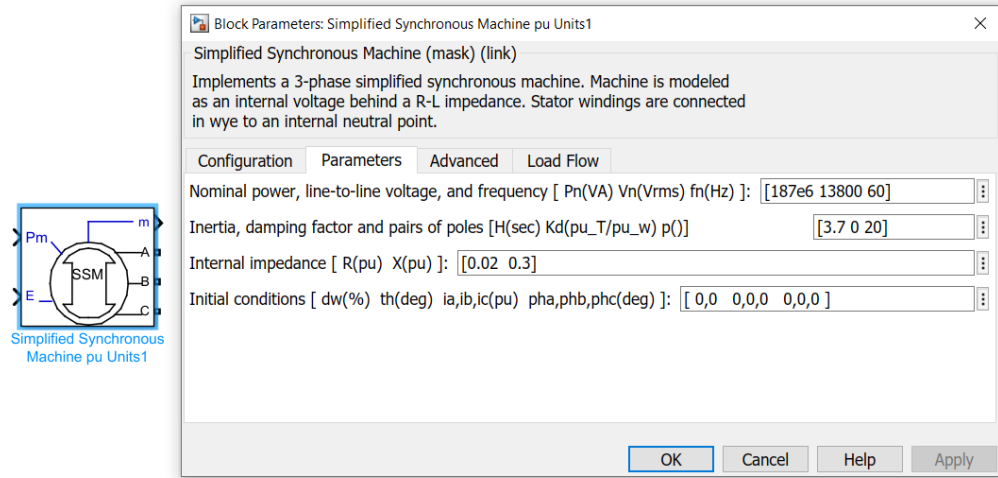
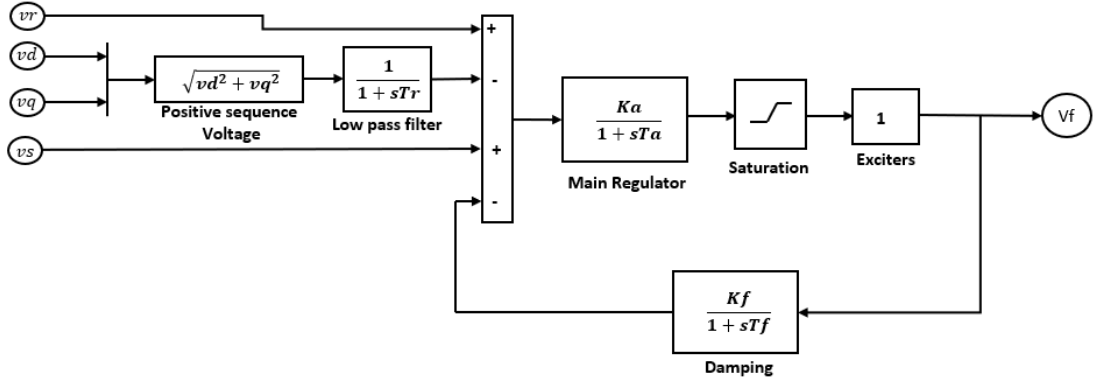


Figure 5.9: *Simplified Synchronous Machine pu Units* and the required parameters

The parameters used for the Simulink implementation are those provided in Table 5.9.

5.3.4 Excitation system

The basic function of an excitation system is to provide direct current to the synchronous machine field winding. In addition, the excitation system performs control and protective functions essential to the satisfactory performance of the power system by controlling the field voltage and thereby the field current [31]. For this thesis, the excitation system that has been implemented is a simplified version of the DC1A Exciter that is presented in [32]. This model is used to represent field voltage with continuously acting voltage regulation. Figure 5.10 shows the block diagram of the control of the DC excitation system implemented. This model, described by the block diagram in Figure 5.10, is used to represent field controlled DC common exciters with continuously acting voltage regulator. In the simplified model, the basic elements is the *Main Regulator*, while the exciter is simplified with a transfer function equal to 1.


Figure 5.10: Type DC1A exciters

The principal input are the the v_d and v_q output voltages from the generator. At the summing junction, the positive sequence voltage ($\sqrt{v_d^2 + v_q^2}$) is subtracted from the reference voltage v_r . The stabilizing feedback v_s , from the Power System Stabilizer (PSS), is added to produce error voltage (v_s is zero if no PSS is used). The resulting signal is then amplified by the *Main Regulator* characterized by a gain K_a and a time constant T_a . The output of the *Main Regulator* is used to control the exciter. The output voltage V_f is the field voltage used to control the Synchronous generator. In addition, a signal derived from field voltage is used to provide excitation system stabilizer via the feedback with gain K_f and time constant T_f [32].

The parameters used for the excitation system are provided in Table 5.10.

Table 5.10: Parameters of the DC1A excitation system

Parameter	Value
T_r (s)	20e-3
K_a	210
T_a (s)	0.001
k_f	0.001
T_f (s)	0.1
$V_{f,MAX}$ (pu)	11.5
$V_{f,min}$ (pu)	-11.5

5.3.5 Steam Turbine and Governor

A steam turbine converts stored energy of high pressure and high temperature steam into rotating energy, which is converted into electrical energy by the generator. The heat source for the boiler supplying the system may be a nuclear reaction or a furnace fired by fossil fuels (coal, oil or gas) [31]. They normally consists of two or more turbine section or cylinders coupled in series. Every turbine consist of a set of moving blades connected to the rotor and a set of stationary vanes. The kinetic energy of this high velocity steams is converted into shaft torque by the buckets. The steam turbine used in this work is a single-mass tandem component turbine, which means that the section are all in on one shaft with a single generator [31].

The model used in this work implements a complete tandem-compound steam prime mover, including a speed governing system and a four-stage steam turbine. The speed governing system consists of a proportional regulator, a speed relay and a servomotor that controls the gate opening c_v [33]. The block diagram of the speed governor system is shown in Figure 5.11.

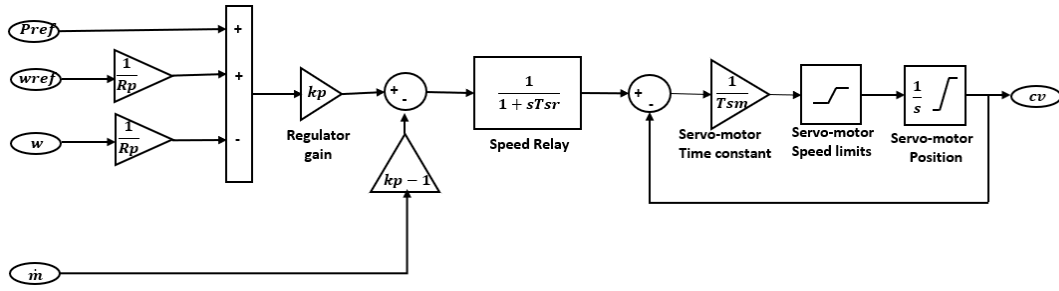


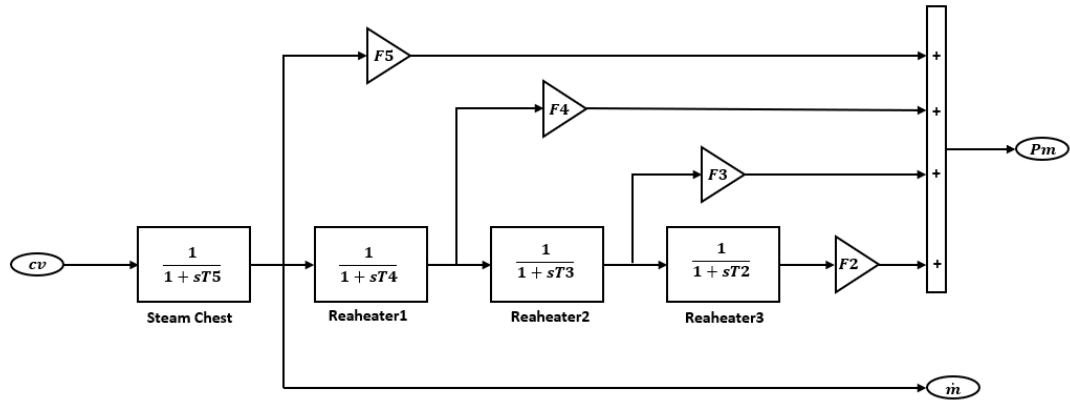
Figure 5.11: Speed governor system for steam turbine

The block diagram of Figure 5.11 shows approximate mathematical relationships for the speed-governing function of the control system with the steam flow feedback operative \dot{m} [33]. The parameters chosen for the implementation of the speed governor are those reported in Table 5.11.

Table 5.11: Parameters of the speed governor system for steam turbine

<i>Parameter</i>	<i>Value</i>
R_p (pu)	0.05
K_p	1.0
T_{sr} (s)	0.001
T_{sm} (s)	0.15
$c_{v,close}$ (pu/s)	-0.1
$c_{v,open}$ (pu/s)	0.1
$c_{v,min}$ (pu)	0.0
$c_{v,max}$ (pu)	4.496

The steam turbine has four stages, each modeled by a first-order transfer function. The first stage represents the steam chest while the three other stages represent either reheaters or crossover piping. Fractions F2 to F5 are used to distribute the turbine power to the various shaft stages [33]. The Figure 5.12 shows the block diagram used to implement the steam turbines.


Figure 5.12: Steam turbine model

The parameters chosen for the implementation of the steam turbine are those reported in Table 5.12.

Table 5.12: Parameters of the steam turbines

<i>Parameter</i>	<i>Value</i>
T_2 (s)	0.0
F_2	0.5
T_3 (s)	10.0
F_3	0.5
T_4 (s)	3.3
F_4	0.0
T_5 (s)	0.5
F_5	0.0

5.3.6 Load and shunt model

In the HV transmission network under analysis the loads are connected to bus 2,3,4,5 and 6. In the Simulink model the *Dynamic Load* block with external control of P, Q are used (the same blocks also adopted in tests with the MV distribution network). The active and reactive power values used as inputs for the blocks representing the load are the maximum values reported in Table 5.4.

The reactive elements (shunts) present in the network are capacitive and are used to locally supply a certain amount of reactive power to the loads to reduce the reactive power value exported between the various areas and consequently reduce the value of the line's current (and losses). There are a total of three capacitive shunts in the network, connected to bus 4,5 and 6. In particular, the most important are those connected to bus 4 and bus 5 as they are the most isolated nodes and connected to the areas where there is generation from very long lines (i.e., 300 km). In the Simulink model the *Three-Phase RLC Load* block is used as shown in Figure 5.13. The active power (P) and the inductive reactive power (Q_L) used are equal to zero while the capacitive reactive power (Q_c) is set with the values shown in Table 5.7.

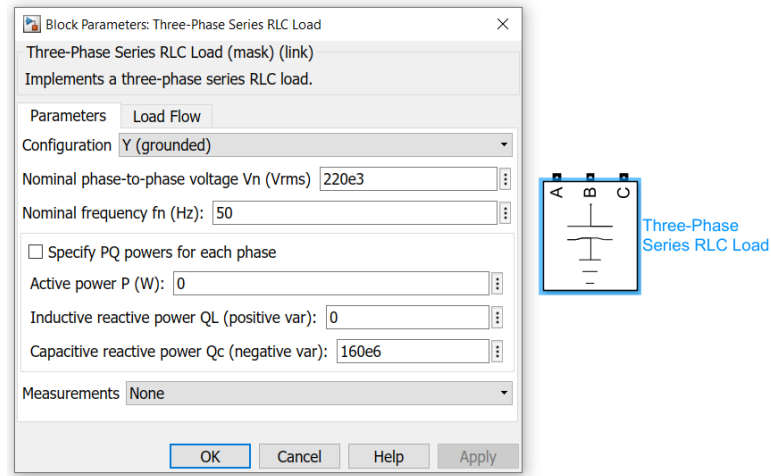


Figure 5.13: *Three-Phase RLC Load* block used to model capacitive shunts

5.4 Model initialization and transmission stability concepts

For the initialization of the Simulink model of the HV transmission network, the load flow was performed. The load flow is used to determine the net active and reactive power injected into the bus and the voltage magnitude and angle of bus positive-sequence voltage. Before solving the load flow, two of the above quantities are known at every bus and the other two have to be determined. Therefore, the following bus types are used:

- **Slack bus:** this bus imposes the voltage magnitude and angle (Table 5.9). The load flow solution returns the active power P and reactive power Q that are generated or absorbed at that bus in order to balance generated power, loads, and losses. For this work, the generator 1 (inserted in bus 9) has been selected as slack.
- **PV bus:** the active power P is generated and the generator terminal voltage V is imposed. The load flow solution returns the machine reactive power Q that is required to maintain the reference voltage magnitude, and the reference voltage angle. The generators 2,3 and 4 (connected respectively to node 10,11 and 12) are considered as PV nodes.
- **PQ bus:** at this bus, the specified active power P and reactive power Q are either injected into the bus (generation PQ bus) or absorbed by a load connected at that bus. The load flow solution returns the bus voltage magnitude and angle. All loads and capacitive shunts are considered as PQ bus.

The tool *Load Flow* of the *powergui* block in Simulink was used to solve the load flow. The results obtained are shown in Table 5.13.

Table 5.13: Load flow results

<i>Node</i>	V_{mag}	V_{angle}	P_{gen}	Q_{gen}	P_{load}	Q_{load}	Q_{sh}
<i>ID</i>	(pu)	(deg)	(MW)	(Mvar)	(MW)	(Mvar)	(Mvar)
1	1.0001	29.60	0.0	0.0	0.0	0.0	0.0
2	0.9965	31.48	0.0	0.0	285.0	200.0	0.0
3	0.9961	2.36	0.0	0.0	325.0	244.0	0.0
4	0.9597	-3.11	0.0	0.0	326.0	244.0	-160.0
5	0.9746	5.91	0.0	0.0	103.0	62.0	-80.0
6	0.9906	2.48	0.0	0.0	435.0	296.0	-180.0
7	1.0006	29.31	0.0	0.0	0.0	0.0	0.0
8	0.9973	2.62	0.0	0.0	0.0	0.0	0.0
9	1.0300	0.0	528.03	-79.95	0.0	0.0	0.0
10	1.0300	1.83	500.0	196.15	0.0	0.0	0.0
11	1.0300	-27.52	200.0	272.70	0.0	0.0	0.0
12	1.0300	-26.66	300.0	163.24	0.0	0.0	0.0

After performing the load flow calculation, it is possible to apply the results to the model to initialize the various blocks with the initial conditions obtained. This procedure is particularly important for the initialization of synchronous generators.

Regarding the concepts of the steady-state stability of the transmission system, it is possible to refer to Figure 5.14. It shows the trend of the transmitted active power as a function of the load angle δ . The value of active power transmitted varies with the sine of the load angle and it is possible to identify the steady-state stability zone in the interval between $-\pi/2$ and $\pi/2$ that is when $dP/d\delta > 0$ [29]. In the curve shown in the Figure 5.14, the P_X represent the maximum transmissible power, the point indicated with *A* it is located within the stability zone while the point *B* is in the instability zone as the load angle results $\delta > \pi/2$.

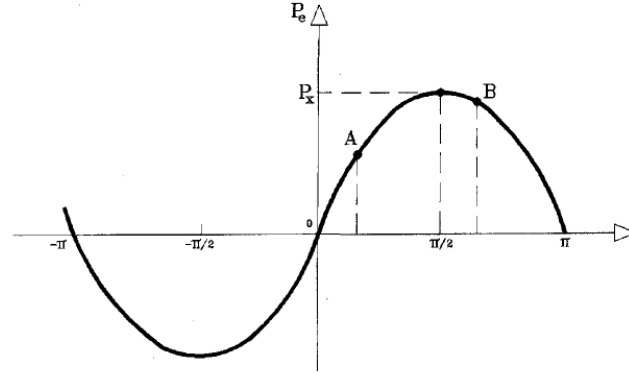


Figure 5.14: Trend of the transmitted power as a function of the load angle δ [29]

The *Synchronous Machine pu Standard* Simulink block allows to measure the load angle value and therefore to have an idea of the load angle value for generators 2,3 and 4. To determine the load angle of the generators, a reference of infinite power must be considered. As a reference for this thesis the slack was considered, that is the generator 1. The values of the load angles and of the electrical power transmitted (expressed in pu) by the generators 2,3 and 4 are shown in Table 5.14.

Table 5.14: active power transmitted by the generator and load angle

<i>Generator</i>	<i>Bus</i>	P_e	δ
<i>ID</i>	<i>ID</i>	(pu)	(deg)
2	10	0.7143	28.11
3	11	0.4	15.1
4	12	0.6	23.4

To conclude, the overall model of the HV network used for the initialization of the various components and in particular of the generators and some considerations on steady-state stability is reported in Appendix C.1.

Chapter 6

Test on the HV Transmission Network

This chapter will describe some tests carried out using the HV transmission network modeled as described in Chapter 5 through the OPAL-RT[®] 5600 Real-Time simulator. The first part of the chapter will describe and explain how the decoupling of the HV transmission network has been performed on multiple calculation cores of the Real-Time OPAL-RT[®] simulator. Subsequently, a local tests have been performed on the primary frequency response following an abrupt load event. To conclude the HV transmission network is modified to insert renewable generation systems (RES) and a remote tests have been carried out in collaboration with the other universities involved in the project.

6.1 Decoupling of HV transmission network

In Chapter 4 the computational burden was limited, and this allows to simulate the model on a single core of the OPAL-RT[®] simulator. The increase of the network extension implies an increase in the computational burden. A Real-Time simulation will then require a decomposition of the network in groups and decouple them in more than one core: in this way a parallel computation will be executed.

For transmission system models, it is possible to take advantage of existing natural delays on long transmission lines due to traveling waves, as used in the Bergeron's line model with losses and wideband frequency-dependent line models [34]. Note that the transmission lines described as “long” are those whose propagation delay is longer than the simulation time-step [35]. Figure 6.1 shows the *ARTEMiS Distributed Parameters Line* block, which allows the decoupling of the transmission network through long lines. The ARTEMiS block can implement an

N -phases distributed parameters transmission line model optimized for Real-Time simulation [34].

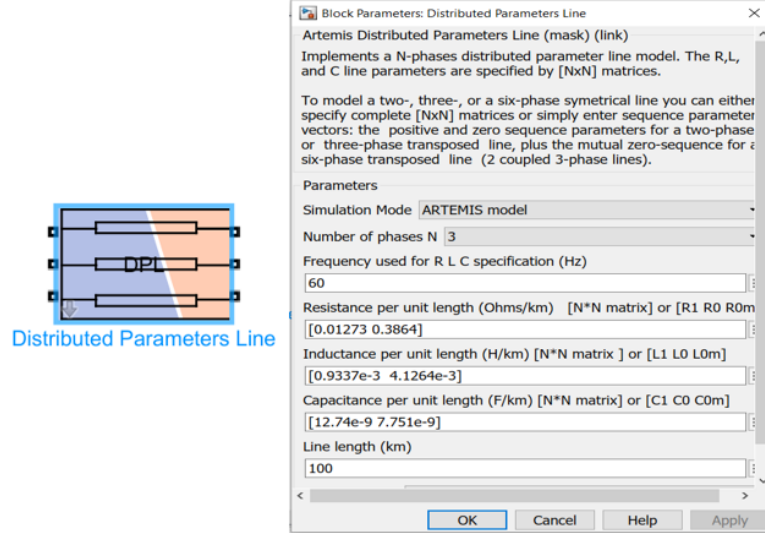


Figure 6.1: *ARTEMiS Distributed Parameters Line* and the required parameters

The parameters required to configure the ARTEMiS blocks are the same as those used for Figure 5.3. In order to correctly use the block, the *ARTEMiS Distributed Parameters Line* is applied to replace the long transmission lines that interconnect the various areas of the HV transmission system and in particular the lines 1-6, 2-5, 6-4 and 7-8. In order to use the ARTEMiS blocks in the Real-Time simulation, the *ARTEMiS Guide* block must be inserted (Figure 6.2).

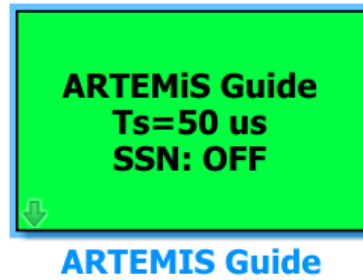


Figure 6.2: *ARTEMiS guide*

Finally, the decoupled network used for local tests on the Real-Time OPAL-RT[®] simulator is schematized in Figure 6.3.

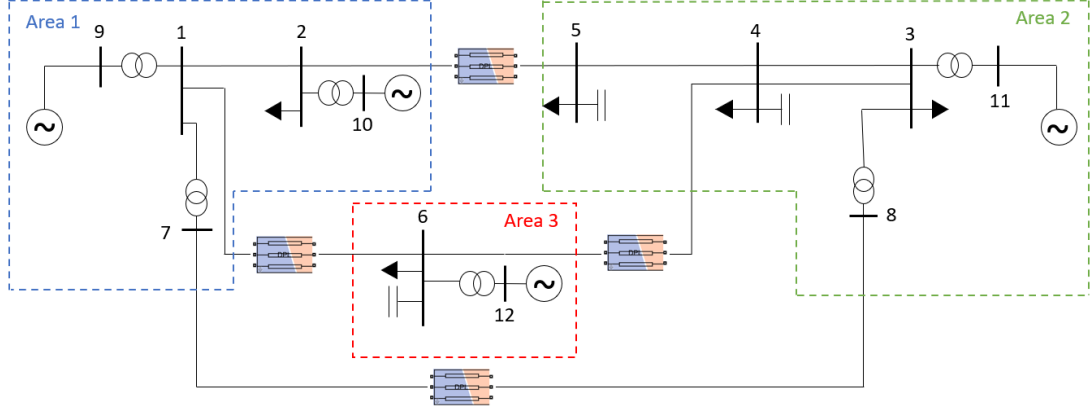


Figure 6.3: Decoupled HV transmission network

6.2 Local test: Primary frequency response

The primary frequency control is automatically performed and in a decentralized way (independently) by all the power resources designed and qualified for this purpose. It aims at stabilizing the frequency around an acceptable quasi-stationary value after a disturbance that causes the frequency to deviate outside the predefined limits. The disturbance consists in the unbalance between generation and load demand, caused by the change in load, by the change in generation, or by the inadvertent change in the power flow on the interconnection lines. Frequency stability refers to the ability of a Power System to maintain steady frequency following a severe system upset resulting in a significant unbalance between generation and load. In the case of slow unbalances, the frequency is the same in every point of the network, while sudden unbalances result in unequal frequency values in different areas during the transient period. The severity of a perturbation depends on the amount of mechanical inertia available in the synchronous generators with respect to the total instantaneous load. Small mechanical inertia may result in an excessively large frequency deviation after a sudden power unbalance, which in the long run may result in frequency instability [36].

The test described in this section was performed using the decoupled transmission network on multiple cores of the OPAL-RT[®] 5600 simulator shown in Figure 6.3. The purpose is to display the primary frequency response of the generators following an abrupt load event. Since the synchronous generators work at the synchronism speed, the frequency has been obtained from the instantaneous values of the mechanical speed measured for the four machines. The load event was triggered by a step change in the value of the active power absorbed by the load present at bus 5 of a value equal to 100 MW. The load event was triggered manually at a time of about 124s thanks to the presence of a switch in the SC_Console of the model and it is shown in Figure 6.4.

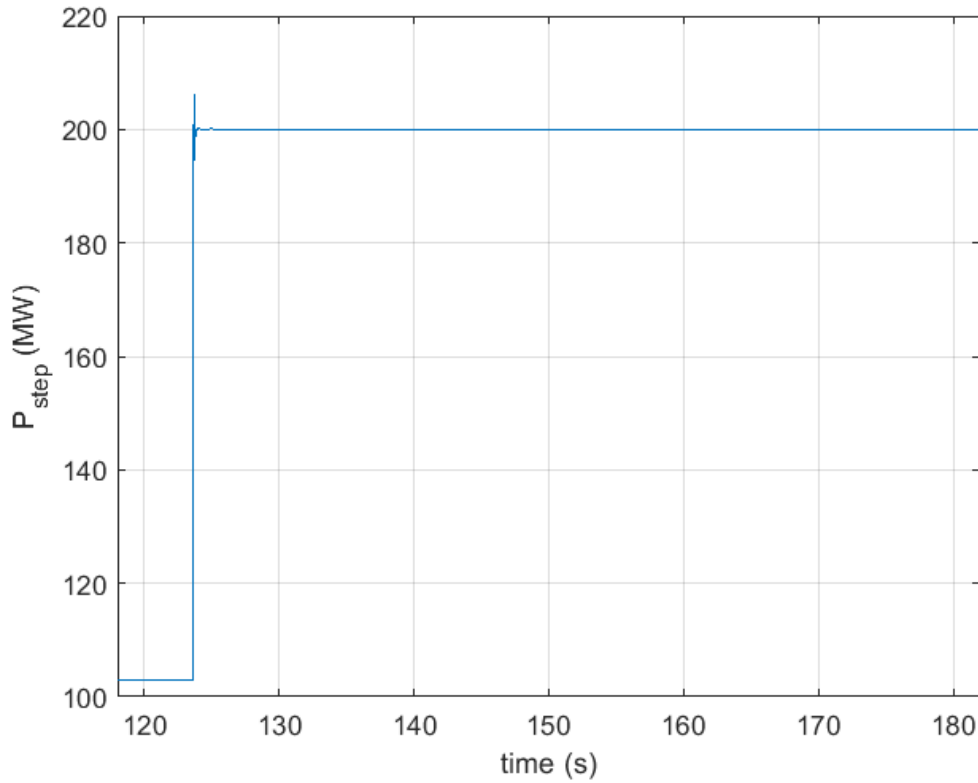


Figure 6.4: Load event at bus 5

The frequency responses measured for each generator is shown in Figure 6.5 where:

- $f1$ (Hz): is the frequency measured for generator 1, i.e., the generator connected to bus 9.

- f_2 (Hz): is the frequency measured for generator 2, i.e., the generator connected to bus 10.
- f_3 (Hz): is the frequency measured for generator 3, i.e., the generator connected to bus 11.
- f_4 (Hz): is the frequency measured for generator 4, i.e., the generator connected to bus 12.

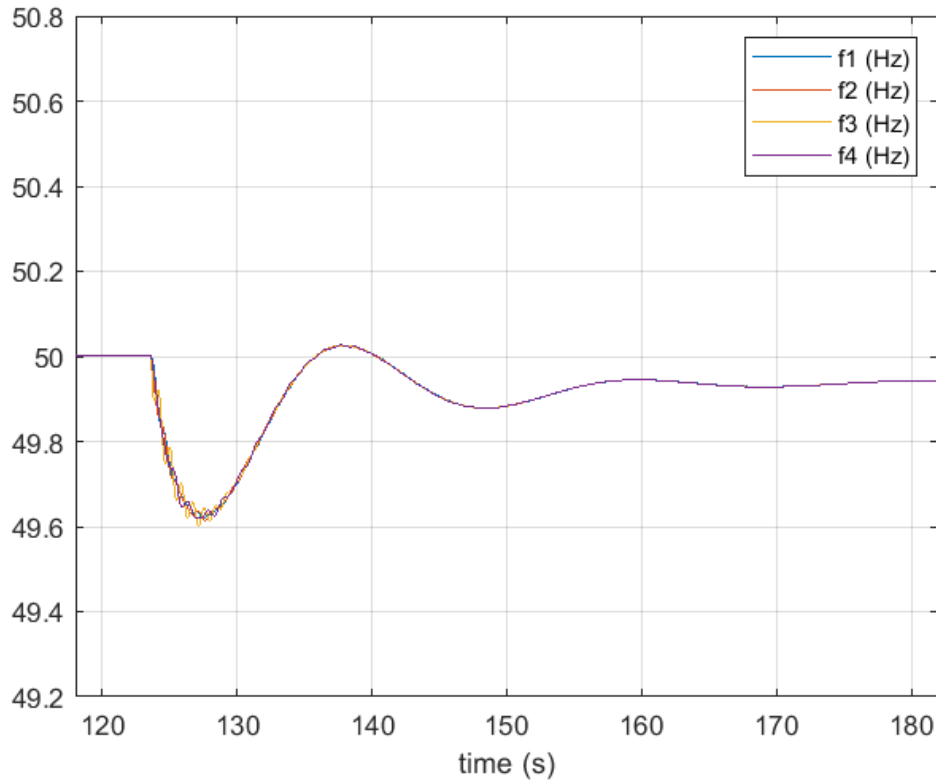


Figure 6.5: Frequencies responses

Figure 6.5 shows the dynamic response of the four generators. Following the load event, the system is stable and the value of the frequency nadir is not so low (about 49.62 Hz, reached about 5 s after the load event). Some local and inter-area oscillation mode can be observed [36]. The frequency of the oscillations is very large (between 1 and 2 Hz). Generator 3 (connected to bus 11) is the one with the most evident oscillations because it is the one that has the least inertia and it is located in the area with the smallest amount of generation available.

To evaluate the overall response of the system, it is possible to refer to the frequency calculated for the Center of Inertia (f_{COI}) defined by the Equation 6.1:

$$f_{COI} = \frac{\sum_{i=1}^{N_{gen}} H_i f_i}{\sum_{i=1}^{N_{gen}} H_i} \quad (6.1)$$

where:

- H_i represents the inertia value of the i -th generator.
- f_i represents the measured frequency of the i -th generator.
- N_{gen} is the total number of generators ($N_{gen} = 4$ in this case).

The f_{COI} for the system under test is shown in Figure 6.6.

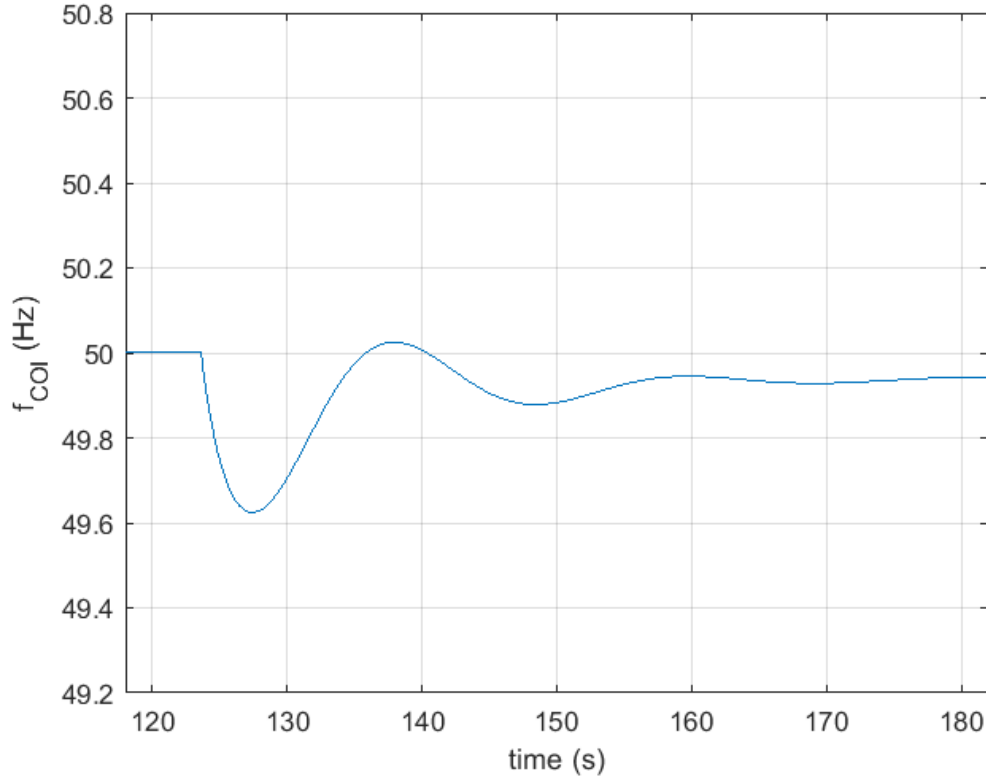


Figure 6.6: Frequency of the Center of Inertia

6.3 Remote Co-simulation: Inertial response of the system after the insertion of RES

6.3.1 Introduction to the dynamics of modern electrical systems

In the last decades, the electrical Power Systems have experienced worldwide substantial changes in the principles and philosophy of operation. Some changes are related to the need of solving climate change, energy shortage, and environmental pollution or to the advancement in technologies. One of the most important innovations was the introduction of RES, such as wind turbines or photovoltaic systems, will dominate the dynamics of future electric power grids. The ongoing transformation of the electric supply from large, centralized power plants based on nuclear or fossil fuels to smaller, decentralized sources based on renewable energy, constitutes an enormous challenge for the stable operation of the grid. Power electronic (PE) converters play a key role in the system, operating as controlled power interface devices, and allowing local control of both directions of flow.

Following these changes happening in the grid, the dynamics and stability of the power system becomes more complex and critical than ever, calling for new control strategies, technologies, and methods. The intermittent and stochastic nature of wind and solar power represents a first main concern based on their integration into the transmission system, as it can have a high impact on the steady-state stability of the system itself. In terms of dynamics, when wind or solar power plants exist in the transmission system, dynamic changes in the level of power can be seen as disturbances to the system. While changes in wind power are usually smooth (except for wind gusts), for solar power the changes could be more sudden, especially as a consequence of cloud formation and movement. Furthermore, in terms of inertia (which strongly influences dynamic stability), wind and solar power have different dynamic characteristics and inertia response compared to conventional rotating machine-based generation. Wind turbines are generally equipped with back-to-back converters, in the form of Doubly Fed Induction Generators or Full Converter Synchronous Generators, which electrically decouple the generator from the grid. Therefore, no inertial response is delivered during a frequency event, although a certain amount of kinetic energy is stored into the blades and the generator. Solar panels are virtually inertia-less. This means that contributions from wind and solar power generators to the grid system's inertia are not direct and mechanical, which can represent a concern for the overall stability of the system. If wind and solar plants were used to replace conventional synchronous generation power plants in the transmission system, this would cause loss of rotating inertia and damping torque, so that the overall transient stability of the system could

be weakened. In this case, proper counteractions should be applied via control methods based on the PE devices [36].

6.3.2 Primary response after the introduction of RES

This test concerns a remote co-simulation carried out in collaboration with all the different Italian universities presented in Chapter 4. In particular, the remote co-simulation was organized as follows:

- **PoliTo:** the HV transmission network shown in Figure 6.3 has been decoupled on several computation cores of the OPAL-RT[®] 5600 simulator.
- **UniGe:** a wind farm equipped with a centralized storage system was implemented through the use of the speedgoat[®] simulator.
- **Campus Savona:** the photovoltaic system actually installed in the Campus of Savona was used to simulate a large PV generator.
- **PoliBa:** a controllable load is inserted, which can be connected or disconnected from the transmission network.

To carry out the remote co-simulation, some changes have been made to the transmission network to allow the insertion of the RES. In particular the network shown in Figure 6.3 has been modified to emulate the partially decommissioning of a traditional generation park. The traditional generation has been replaced by:

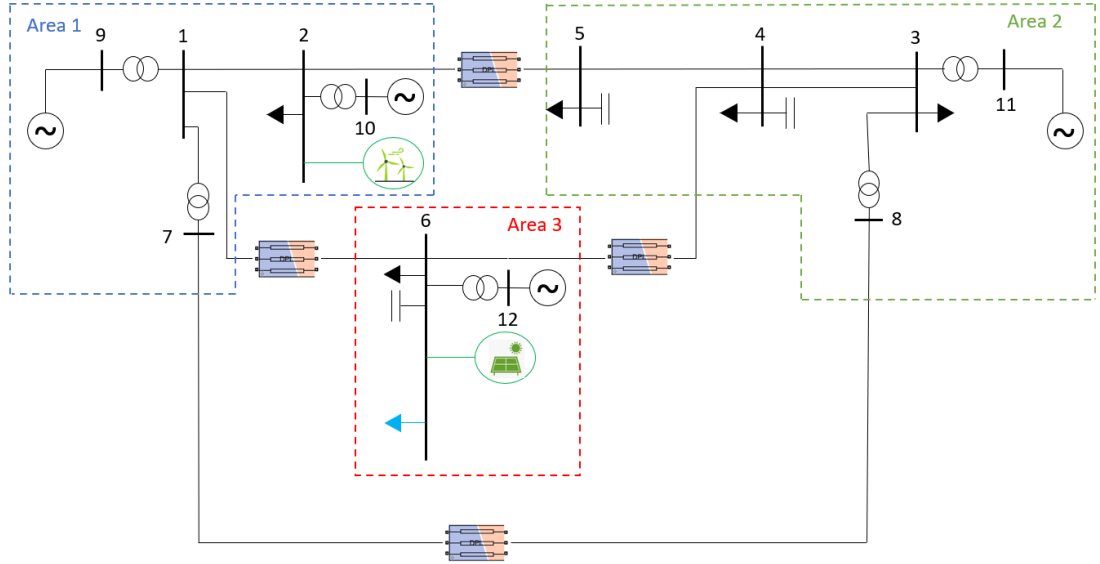
- **wind generation** (simulated in Genoa) connected to bus 2 of the HV transmission network
- **solar generator** (Savona plant) connected to bus 6 of the HV transmission network
- **detachable load** (simulated in Bari) connected to bus 6 of the transmission network

The decommissioning of part of the traditional generation park reduces the global inertia of the network. In particular, since the wind generator was inserted in node 2 and the photovoltaic generator and the micro-grid of Bari at node 6, the apparent power and active power of the generators connected to those nodes have been halved. Table 6.1 summarizes the changes made. The value of loads and capacitive elements has been kept equal to the base case.

Table 6.1: Variations in the power of the generators

<i>Generator</i>	<i>Bus</i>	S_{old}	P_{old}	S_{new}	P_{new}
<i>ID</i>	<i>ID</i>	(MVA)	(MW)	(MVA)	(MW)
2	10	700	500	350	250
4	12	500	300	250	150

The structure of the HV transmission network used is presented in Figure 6.7 where the RES generation systems are represented in green and the detachable load of Bari is represent in cyan.


Figure 6.7: HV transmission network with RES penetration

Regarding the communication layer between different systems, the VILLAS framework was used as for the tests presented in Chapter 4. Signals exchanged between different geographically distributed laboratories are summarised in Figure 6.8: the Turin model sent to the models of Genoa and Bari the frequency and the RMS value of the voltage measured in the areas where they are connected whereas the Genoa and Bari models sent the value of the active and reactive powers (generated and absorbed, respectively) to Turin. Since in Savona there is a physical photovoltaic system, this site sent only the value of the active power produced by the photovoltaic generator without receiving any feedback. The configuration file

for the VILLAS node used in Turin is shown in Appendix C.3.

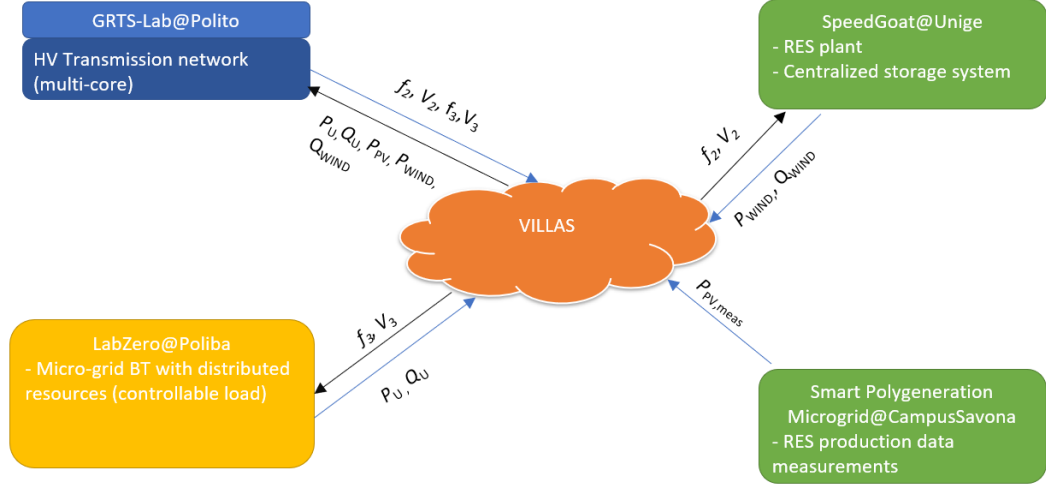


Figure 6.8: Connection diagram

Locally, the various simulated systems are represented by the *Dynamic Load* block in the Simulink model which is commanded with the respective values of active and reactive power received by remote sites.

The frequency transient was activated by a 100 MW-load step at bus 5, as for the test performed locally (Figure 6.9). Figure 6.10 shows the frequency trend of the four generators (divided into three areas), while Figure 6.11 shows the frequency trend of the Center of Inertia f_{COI} . Compared to the case shown in Figure 6.5, it is possible to note that the local and inter-area oscillations mode are more evident because the size of generators 2 and 4 has been reduced and, consequently, the local oscillation of these generators are greater than the ones related to the local test. This aspect can also be observed in the representation of the f_{COI} that shows more oscillation than in the base case. In particular, the worst cases are represented by generators 3 and 4. The generator 3 is located in Area 2, which is the main load centre, with a small amount of generation available, and its constant of inertia is the smallest of the four generators present. The generator 4 is located in Area 3, between the main generation Area 1 and the main load centre Area 2: its inertia constant turns out to be sufficiently large but its size has been halved. The same considerations are valid for generation 2, located in Area 1: although its size has also been halved, the local oscillations are not as evident as those of generator 4 because a large wind farm has been inserted in the same node where generator 2 is present, providing the necessary power to replace the share subtracted from the traditional generation. Concerning the frequency nadir, this does not differ significantly from the base case (49.59 Hz).

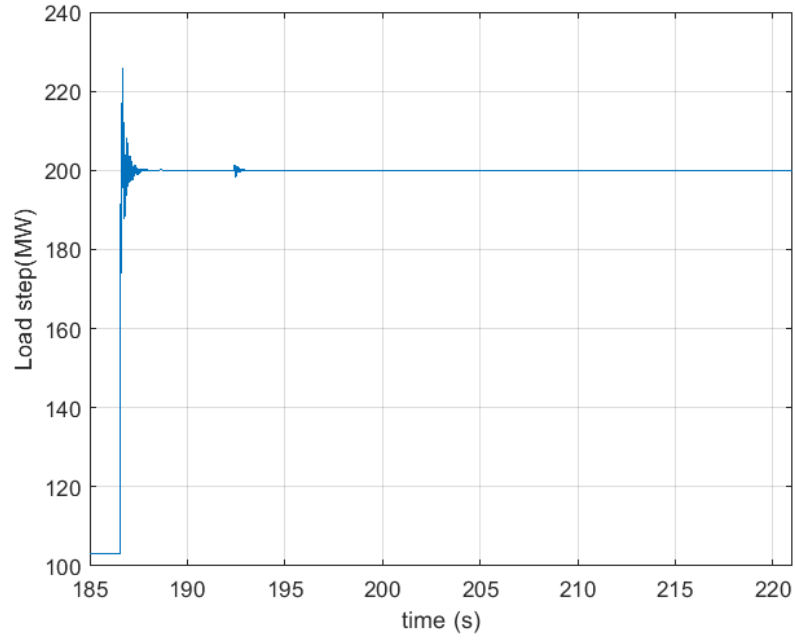


Figure 6.9: Load event at bus 5

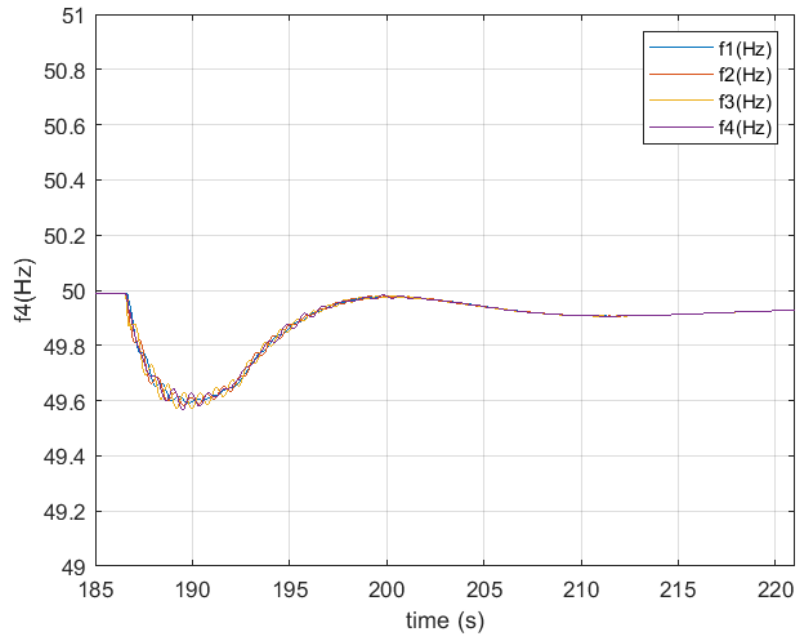


Figure 6.10: Frequencies responses of the traditional generator

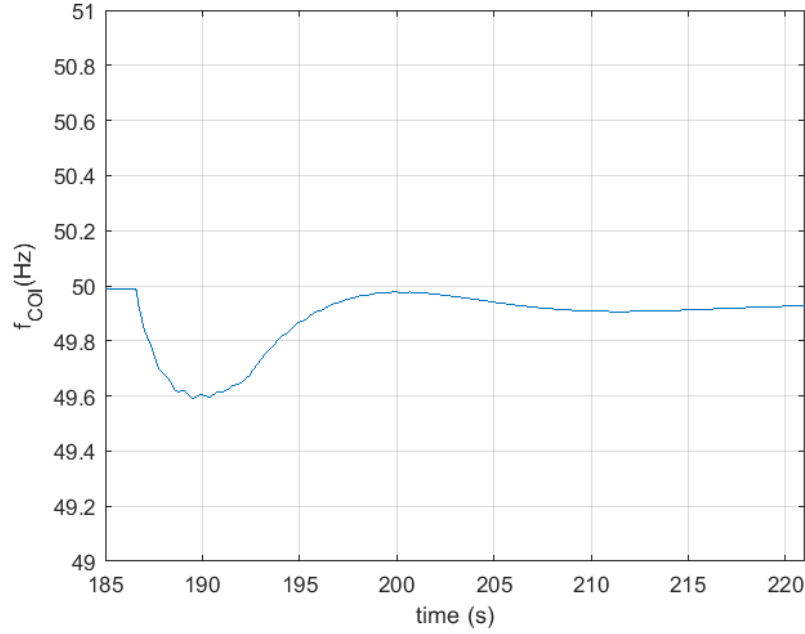


Figure 6.11: Frequency of the Center of Inertia

The disconnection of the Bari load is introduced to a time of approximately 192s and the effect on the frequency can be noted especially by observing the f_{COI} , whose slope change after that time-step. Overall, the frequency response is mainly given by the traditional generators and the wind system simulated in Genoa while the photo-voltaic resource, actually installed in Savona, remains connected during the transient and provides the real active power produced. The disconnection of the Bari load is shown in Figure 6.12. The active and reactive power generated by the simulated wind system in Genoa are shown in Figure 6.13, while the active power received from the PV generator of Savona is shown in Figure 6.14. Please note that the value of the active power generated by the PV of Savona is much lower than the one provided for by the test (30 MW, obtained by using a proper gain) and remains almost constant throughout the time the data was recorded.

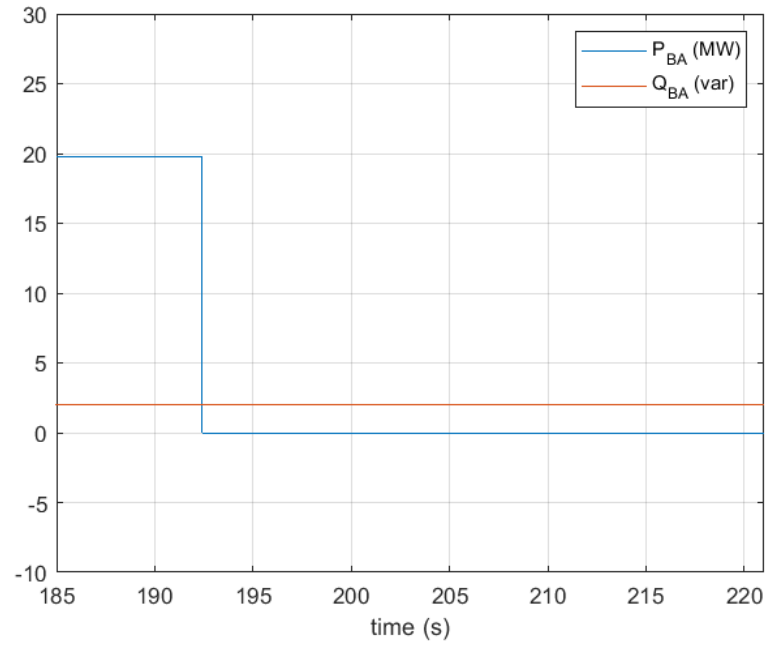


Figure 6.12: Active and reactive power received from Bari

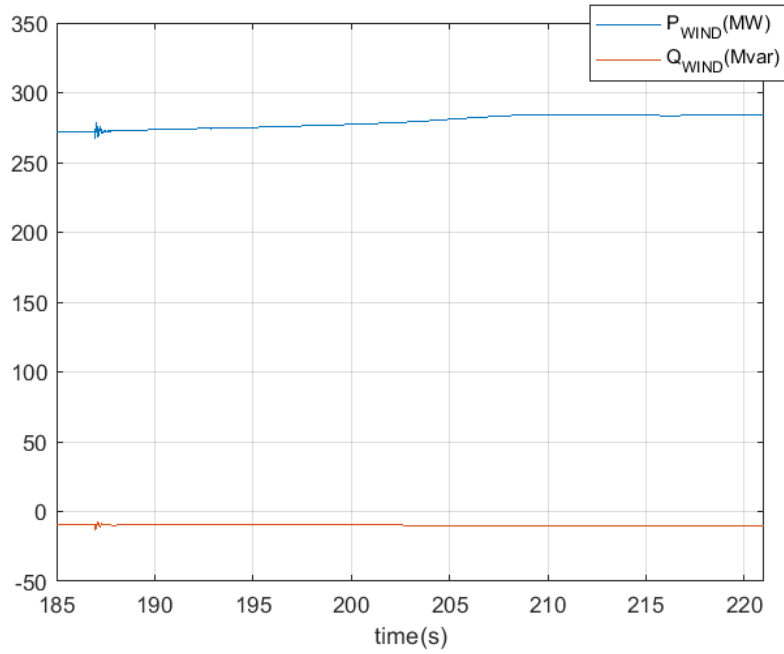


Figure 6.13: Active and reactive power from the wind generator in Genoa

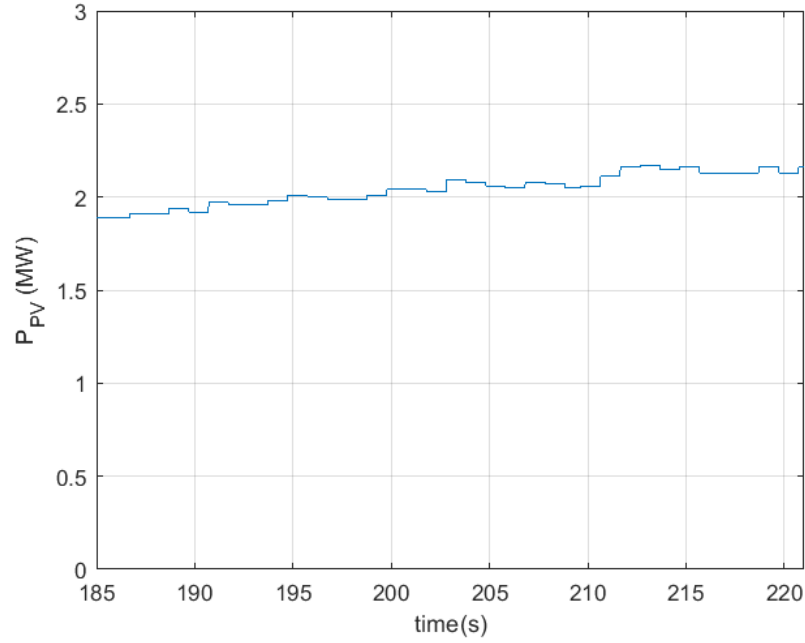


Figure 6.14: Active power produced by the PV generator in Savona

6.3.3 Remote Co-simulation: PHIL setup

This test refers to a remote co-simulation in carried out with the Politecnico di Bari (PoliBa) and the University of Genova (Campus Savona), where only the hardware elements exist. Concerning the wind system simulated in Genoa, it has been replaced with a flexible load that provides the expected power of the system. The remote co-simulation was organized as follow:

- **PoliTo:** the HV transmission network already implemented for the previous tests was used.
- **Campus Savona:** the PV system actually present in Savona was used to emulate the behaviour of a large PV farm.
- **PoliBa:** PHIL components are implemented to emulate the behaviour of a distribution network in which detachable loads are present.

The configuration of the transmission network remains unchanged from the previous case, shown in Figure 6.7 where the PV plant and the PHIL components are connected to node 6.

Also the communication aspects remained unchanged from the previous case. As regards for the signal exchange, PoliTo sent to PoliBa the RMS value of the phase voltage and the value of the frequency measured in the area where the Bari's PHIL is connected, and received the active and reactive power absorbed by the load. Campus Savona sent to PoliTo the value of the active power produced by the PV generator without receiving any feedback. The configuration file for the PoliTo's VILLAS node is the same used for the previous test and is described in Appendix C.3.

The test results are reported below. Contrary to the previous tests, described in Section 6.2 and in Section 6.3, the frequency transients were activated by a series of load connections and disconnections at node 5, with a value of 100 MW. Figure 6.15 shows the load variation of bus 5.

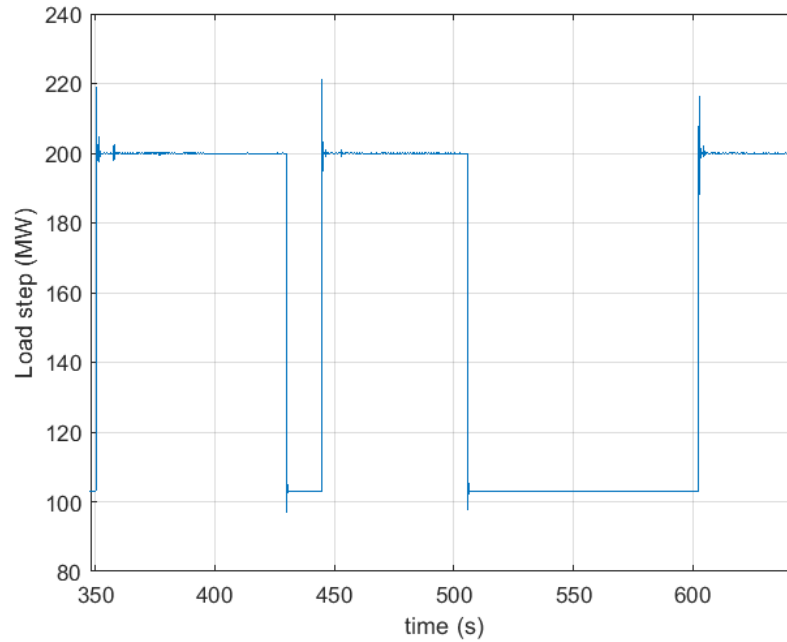


Figure 6.15: Load step variation at bus 5

The load in PoliBa's laboratory has been implemented to automatically switch off when the frequency value reached the value of 49.7 Hz and to automatically switch on when the frequency exceeds the value of 49.9 Hz. During the test, the load step values were varied and in the last part (after time 600 s), the disconnection was manual without subsequent re-connection. Figure 6.16 shows the step variations of the Bari load.

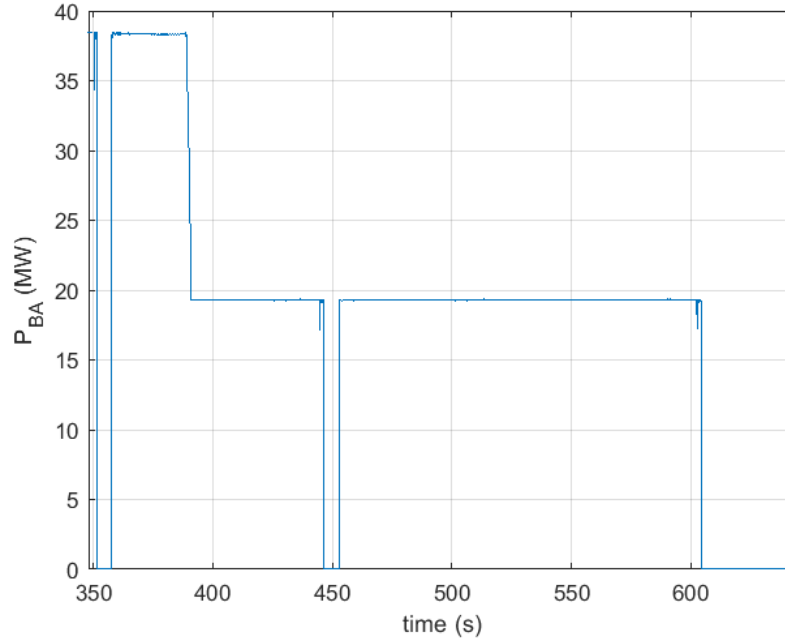


Figure 6.16: Step variations of the Bari load

Since the duration of the test was considerable, it was also possible to appreciate some variations related to the power produced by the PV generator due to the sudden weather variations (passage of clouds) after time $t=350$ s and time $t=570$ s. The recorded power variations are not abrupt as in the case of Bari, but still influence the dynamics of the system. Figure 6.17 shows the trend of the active power generated by the PV system.

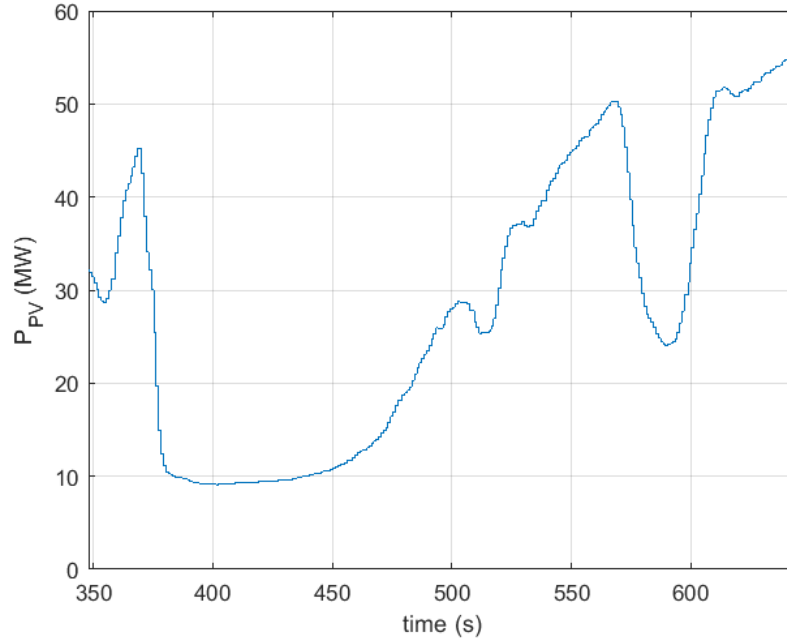


Figure 6.17: Variation of the active power produced by the PV generator

Finally, the frequencies responses of the four generators and of the f_{COI} are shown. Figure 6.18 shows the trend of the frequencies of the different generator connected to the HV system, while Figure 6.19 shows the trend of the equivalent frequency of the COI. The frequency trend is mainly influenced by the load events at bus 5 (red circles in Figure 6.19), and in correspondence with them, the greatest deviations from the reference frequency can be noted. The events related to the disconnection and subsequent reconnection of the load in Bari produce smaller variations of the frequency and they can be observed mainly in the first part of the Figure 6.18 and Figure 6.19 (green circles) where the load variations are higher (i.e., 38.5 MW). The variation of the power produced by the PV generator of Savona can be appreciated in the frequencies plots after time 370s and time 570s (orange circles in Figure 6.19) when there are sudden drops in solar generation due to the passage of clouds.

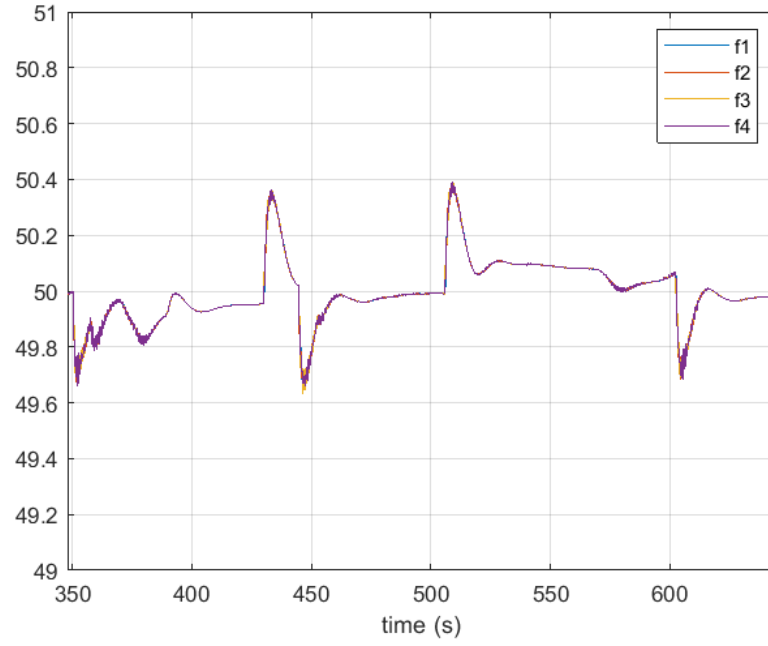


Figure 6.18: Trend frequencies of the four generators

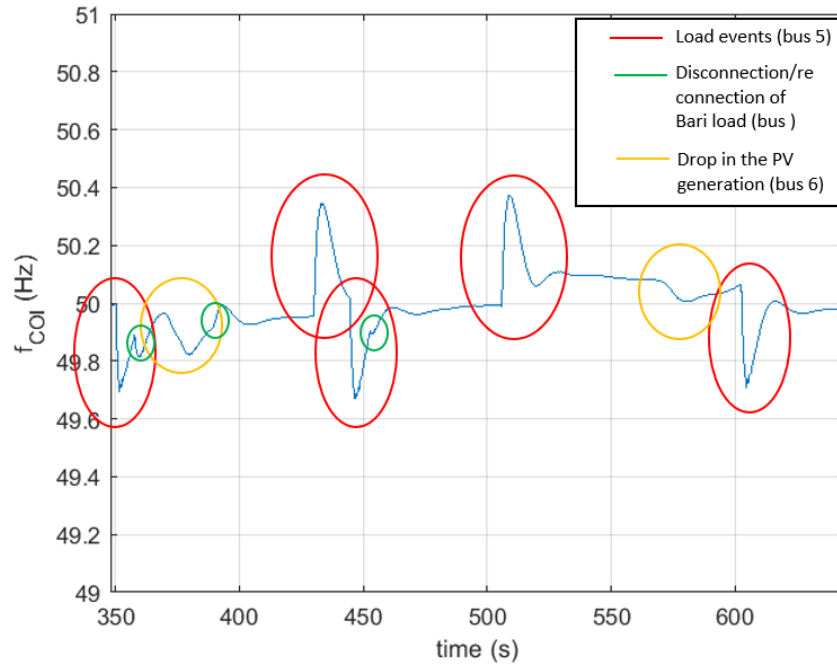


Figure 6.19: Trend of f_{COI}

Chapter 7

Conclusion and Future works

The Real-Time Remote co-simulations realized within the project promoted by ENSIEL show the validity of a geographically distributed research facility. Specifically, this thesis has presented a framework for virtual integration of laboratories across Italy for Real-Time co-simulation. An architecture of the framework was designed with the goal of enabling flexible integration of RT digital simulators. Laboratories are interfaced over a high-bandwidth national research and education networks interconnected with the Internet network while data exchange is based on UDP protocol. The communication link utilized for RT co-simulation is characterized empirically based on loop-back communication tests. The framework for virtual integration of laboratories (VILLAS) is applied for co-simulation of transmission and distribution systems demonstrating the benefits of geographically distributed simulation for large-scale simulations.

First, some local and remote tests were performed, in collaboration with Politecnico di Bari, to demonstrate how the type of data exchanged can affect the reliability of remote co-simulations. This analysis has led to the result that it is not convenient to exchange instant signals, defined in time domain, but it is preferable to rely on a phasor approach. For this reason, a hybrid phasor approach was used in the rest of the work.

Secondly, a distribution network was implemented using the OPAL-RT[®] products in order to integrate in the co-simulation Hardware (Campus Savona) and Software (Unige) elements in addition to the PHIL implemented in PoliBa. These tests showed the feasibility to investigate problems related to the distribution network, such as the dispatchment of LV and MV flexible sources.

In the last part of the thesis an European like HV transmission network was implemented on the OPAL-RT[®] simulator to study the primary frequency response following an abrupt load event. The network, decoupled on several simulator computation cores, was initially analyzed locally and, subsequently, was modified to allow the insertion of RES. In particular, the HV network has been modified to emulate the decommissioning of traditional generation parks, replaced by RES-based power plants and storage systems. The tests carried out on the transmission grid involved all the partners participating in the project.

Future extension of this thesis may be twofold. First of all, the flexibility and portability of the VILLAS communication protocol may allow to extend the lab by adding new participants, both at national and international level. Moreover, the infrastructure may be used to widen the case studies, by adopting larger power systems, and new equipment, with the aim to test real devices (e.g., Phase Measurement Units) or new control strategies (e.g., related to the combined use of wind and storage) before the real implementation in the actual system.

Appendix A

Tests with sine wave signals

A.1 OPAL-RT[®] Asynchronous Process

In order to correctly execute such remote Real-Time co-simulation, a series of ASCII files are required. Among those there is the *AsyncIP.c* file. This file is started by the asynchronous controller and allows to send and receive data to and from the asynchronous icons and a UDP or TCP port. Following the Build operation of the model, the required files can be added by the Files window of RT-Lab[®] as shown in Figure A.1.

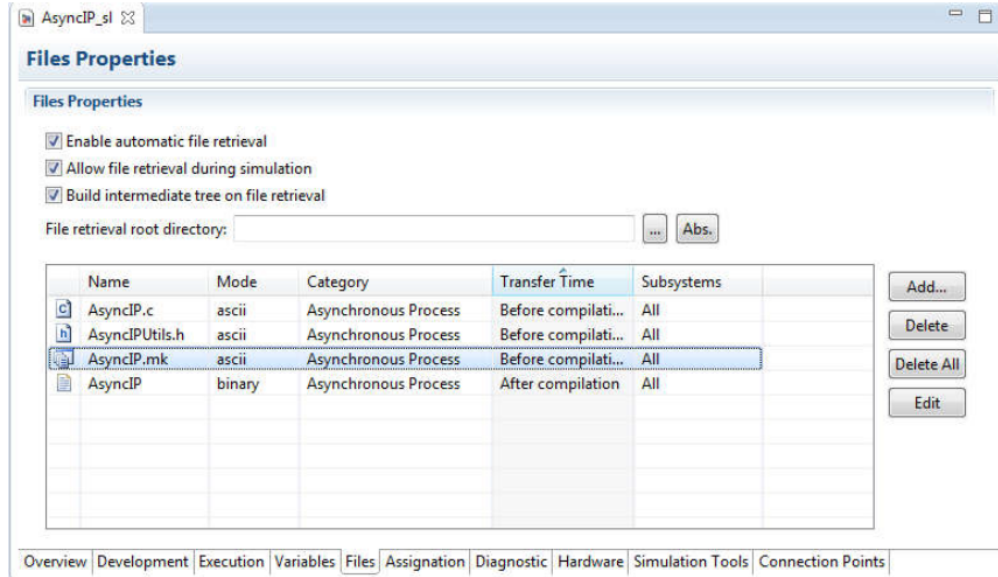


Figure A.1: Loading of required files from RT-Lab[®]

Similarly, the communication between OPAL-RT[®] model and VILLAS node is established by using Asynchronous programs. To activate the communication between OPAL-RT[®] simulator and a VILLAS node the procedure needs to be modified a bit, but with the same principles. In fact, the *AsyncIP* function is not used directly but another configuration file is used which calls the *AsyncIP* function inside it. This new function is called *Makefile.mk* and the file needed for its building are the following shown in Figure A.2. All of these files are available in [37].

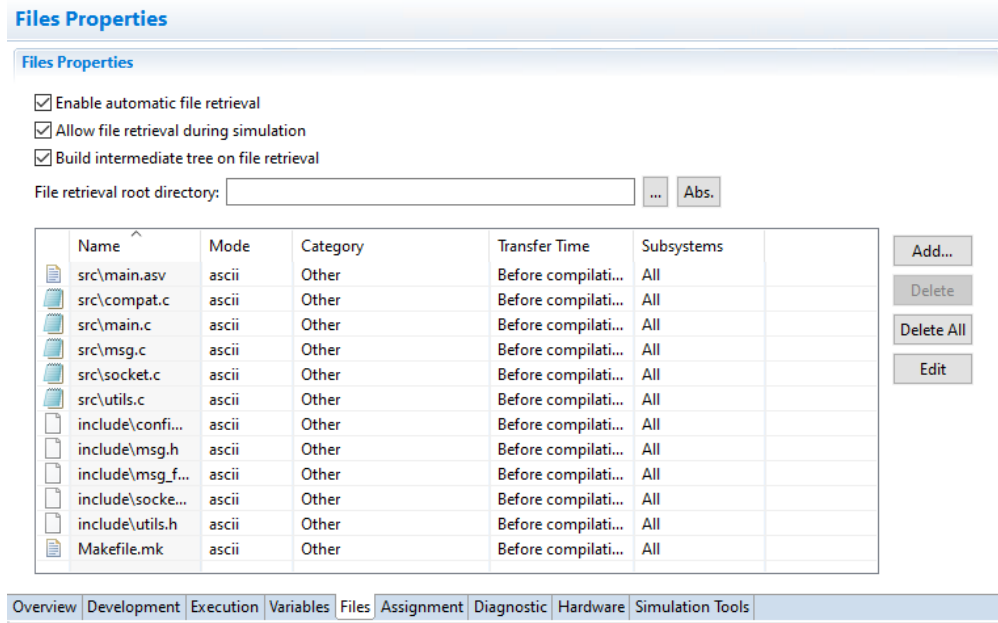


Figure A.2: Loading of required files from RT-Lab[®] for the connection with VILLAS node

To complete the configuration in RT-Lab[®] must be specified the following command in the Compiler command text box as shown in Figure A.3:

```
1 /usr/bin/make -f /usr/opalrt/common/bin/opalmodelmk
```

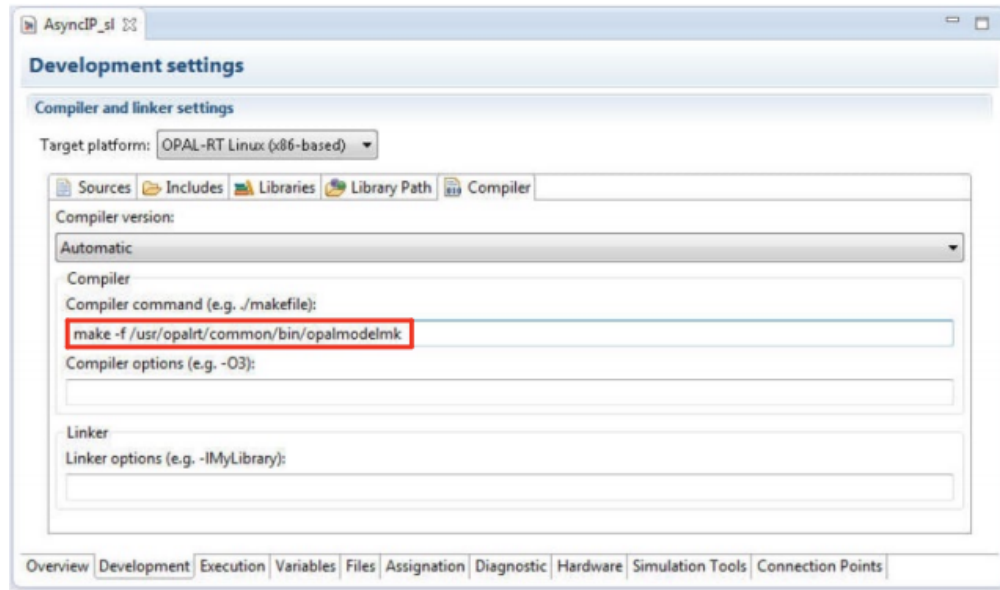


Figure A.3: Compiler command

At this point it is possible to Build the model. At the end of the Build a new file with the *.llm* extension will be generated. Please note that the *.llm* file must contain the following lines of code:

```

1 [ExtraPutFilesComp]
2 Makefile.mk=Ascii | Other
3 include\config.h=Ascii | Other
4 include\msg.h=Ascii | Other
5 include\msg_format.h=Ascii | Other
6 include\socket.h=Ascii | Other
7 include\utils.h=Ascii | Other
8 src\compat.c=Ascii | Other
9 src\main.asv=Ascii | Other
10 src\main.c=Ascii | Other
11 src\msg.c=Ascii | Other
12 src\socket.c=Ascii | Other
13 src\utils.c=Ascii | Other

```

A.2 Configuration of the VILLAS node for local tests

The Appendix A.2 describes in a deeper way the configuration file used for the configuration of the VILLAS node for local tests (Listing A.1). Please note that IP addresses are not shown for security reasons.

```

1 nodes = {
2
3     politico-opal = {
4         type = "socket"
5         layer = "udp"
6         format = "villas.binary"
7
8         in = {
9             address = "*:12001"
10            hooks = ({type = "stats"} )
11        }
12        out = {
13            address = "IP_address_polito-opal:12001"
14        }
15    }
16 }
17
18 paths = (
19     {
20         in = "politico-opal"
21         out = "politico-opal"
22     }
23 )

```

Listing A.1: Configuration file of the VILLAS node for sine local tests

During this test a single node called `politico-opal` was defined. The node `politico-opal` is a `socket` type and it use a `UDP` network `layer` protocol to exchange data. The `format` represents the format in which the data is exchanged. In this case the `villas.binary` format was used. This format is explained in detail in Section 3.2.1. The `in.address` represents the local address and port number this node should listen for incoming packets, the character `*` is used to listen on all the input interfaces. The `out.address` represents the remote address and port number to which this node will send data: in this case, the address of the OPAL-RT[®] simulator present at the DENERG of the Politecnico di Torino has been entered.

The path was set in such a way that the VILLAS node received the data from the OPAL-RT[®] simulator and sent it back directly. For this reason the `in.node` and the `out.node` are the same.

A.3 Configuration of the VILLAS node for the remote test with Bari

This section describes more accurately the configuration file used in the remote test with Bari. The configuration file is shown in Listing A.2.

```

1 nodes = {
2     politico-opal = {
3         type = "socket"
4         layer = "udp"
5         format = "villas.binary"
6
7         in = {
8             address = "*:12001"
9         }
10        out = {
11            address = "IP_address_polito-opal:12001"
12        }
13    }
14    poliba-villas = {
15        type = "socket"
16        layer = "udp"
17        format = "villas.binary"
18
19        in = {
20            address = "*:12000"
21            hooks = ( { type = "stats" } )
22        }
23        out = {
24            address = "IP_address_poliba-villas:12000"
25        }
26    }
27 }
28
29 paths = (
30     {
31         in = "politico-opal"
32         out = "poliba-villas"
33     },
34     {
35         in = "poliba-villas"
36         out = "politico-opal"
37     }
38 )

```

Listing A.2: Configuration file of the VILLAS node for sine remote test with Bari

Compared to the configuration file used for the local test, there are two substantial differences: the number of nodes implemented and the path. The configuration of the `polito-opal` node is the same of the previous case, while the `poliba-villas` represents the VILLAS node configured in Bari with which the data will be exchanged. In particular the `out.address` is the IP address of the machine on which the VILLAS node has been installed. Please note that the port used locally in Torino and the one used to communicate with Bari must be different. Between the OPAL-RT[®] simulator and the VILLAS node in Turin, port 12001 was used. Between the VILLAS node in Turin and the VILLAS node in Bari, the port 12000 is used. This is mandatory, if the same port is chosen both for local and remote communication an error will be reported: **Fail to bind socket**.

In this case the OPAL-RT[®] simulator located in Turin send data to the VILLAS node of Turin, this data are sent to the VILLAS node located in Bari, which in turn sends the data to the OPAL-RT[®] simulator in Bari. In order to perform the loop-back a change in the path must be done: the reverse path must be specified. In this way, the data received from the Bari simulator can be re-sent to the Turin simulator to be displayed.

Appendix B

Tests on the MV grid

In Chapter 4 a number of remote Real-Time co-simulations are discussed. Such simulations have been carried out within a collaboration among University of Savona, Politecnico di Torino, Politecnico di Bari and Università degli Studi di Genova. As regards for the models, in the various tests shown in this chapter, the distribution network model and the measurements has been implemented in Turin and it is the same for each test. The model of the distribution network used is shown in the following.

- **Figure B.1** shows the Thevenin equivalent model of the transmission grid and the transformer HV/MV;
- **Figure B.2** shows the distribution grid with the MV/LV transformer;
- **Figure B.3** shows the block from which the electrical quantities are measured
- **Figure B.4** shows the details of the measurements

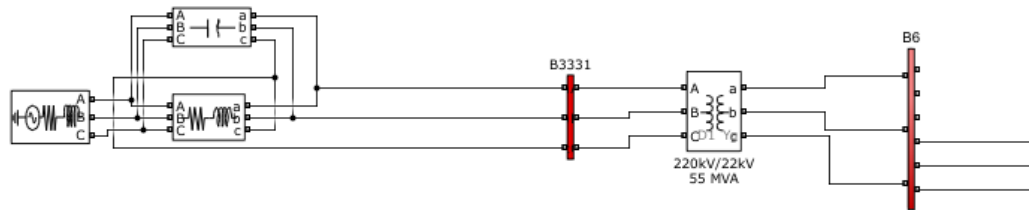


Figure B.1: Simulink model: HV/MV Transormer

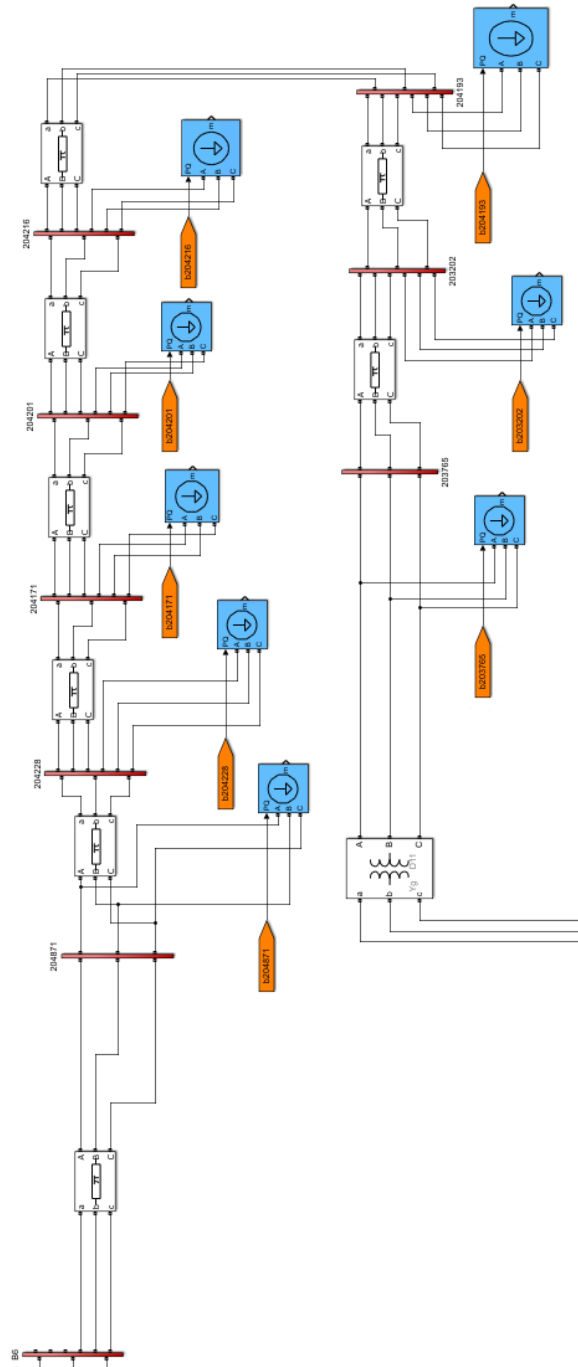


Figure B.2: Simulink model: MV Grid

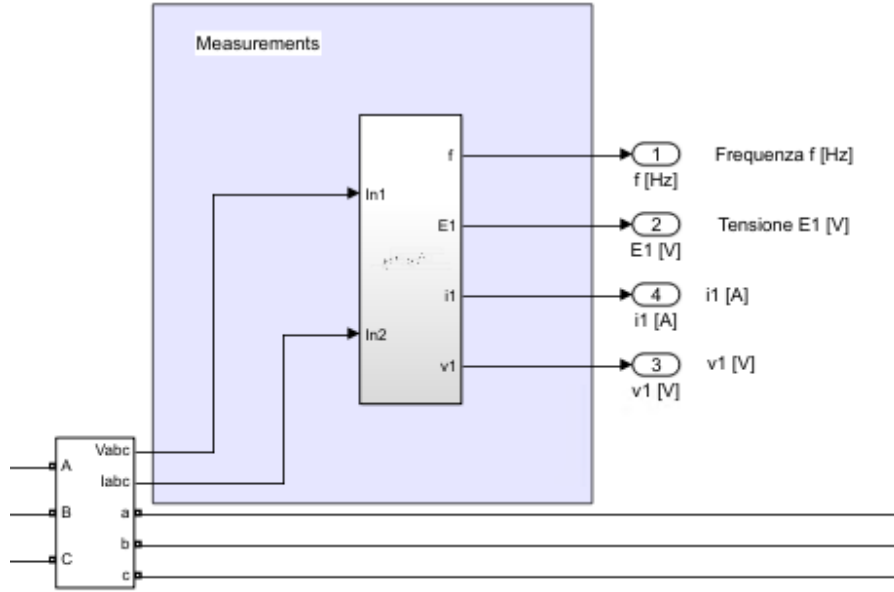


Figure B.3: Simulink model: Current and Voltage measurements

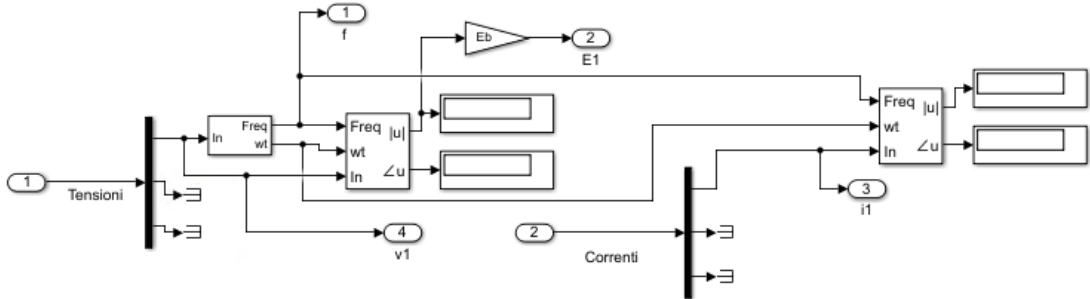


Figure B.4: Simulink model: Current and Voltage measurements details

A flexible load (*Dynamic Load* in Simulink) is used to emulate the behaviour of the of loads. The *Dynamic Load* block is the one used in Figure B.2 (blue blocks). The load varies in each of the co-simulations presented in Chapter 4 as it can be active or passive.

B.1 Configuration of the VILLAS node for the test with Savona

The configuration file used to set up the VILLAS node for the test with Savona (Section 4.1) is shown in Lstlisting B.1.

```

1 nodes = {
2
3     politico-opal = {
4         type = "socket"
5         layer = "udp"
6         format = "villas.binary"
7
8         in = {
9             address = "*:12000"
10        }
11        out = {
12            address = "IP_address_polito-opal:12000"
13        }
14    }
15
16    savona-villas = {
17        type = "socket"
18        layer = "udp"
19        format = "villas.binary"
20
21        in = {
22            address = "*:12001"
23
24            hooks = ( { type = "stats" } )
25        }
26        out = {
27            address = "IP_address_savona-villas:12001"
28        }
29    }
30 }
31
32 paths = (
33     {
34         in = "savona-villas"
35         out = "polito-opal"
36     }
37 )

```

Listing B.1: Configuration of the VILLAS node for the test with Savona

Two nodes have been configured: the `polito-opal` node and the `savona-villas` node. As can be seen from the file, the `polito-opal` node only receives the data arriving from the `savona-villas` node and it does not re-send it because communication is one-way.

B.2 Configuration of the VILLAS node for the test with Bari and Savona

In section the configuration file used in the test with Bari and Savona (Section 4.3) is shown in Listing B.2.

```

1 nodes = {
2     savona-villas = {
3         type = "socket"
4         layer = "udp"
5         format = "villas.binary"
6         in = {
7             address = "*:12001"
8             hooks = ( { type = "stats" } )
9         }
10        out = {
11            address = "IP_address_savona-villas:12001"
12        }
13    }
14
15    polito-opal1 = {
16        type = "socket"
17        layer = "udp"
18        format = "villas.binary"
19
20        in = {
21            address = "*:12002"
22        }
23        out = {
24            address = "IP_address_polito-opal:12002"
25        }
26    }
27
28    polito-opal2 = {
29        type = "socket"
30        layer = "udp"
31        format = "villas.binary"
32        in = {
33            address = "*:12003"
34        }

```

```

35         out = {
36             address = "IP_address_polito-opal:12003"
37         }
38     }
39
40     poliba-villas = {
41         type = "socket"
42         layer = "udp"
43         format = "villas.binary"
44         in = {
45             address = "*:12000"
46             hooks = ( { type = "stats" } )
47         }
48         out = {
49             address = "IP_address_poliba-villas:12000"
50         }
51     }
52 }
53
54 paths = (
55     {
56         in = "savona-villas"
57         out = "polito-opal2"
58     },
59     {
60         in = "polito-opal1"
61         out = "poliba-villas"
62     },
63     {
64         in = "poliba-villas"
65         out = "polito-opal1"
66     }
67 )

```

Listing B.2: Configuration of the VILLAS node for the test with Bari and Savona

In this configuration file four nodes are defined:

- **savona-villas:** represents the remote VILLAS node installed in Savona from which the active power of the PV generator is sent;
- **polito-opal1:** represents the port 12002 of the OPAL-RT[®] in Turin that receives data from Savona;
- **polito-opal2:** represents the port 12003 of the OPAL-RT[®] simulator in Turin that exchanges data with Bari;
- **poliba-villas:** represents the remote VILLAS node installed in Bari that exchanges data with Turin.

Please note that the two nodes referred to the OPAL-RT[®] simulator of Turin physically represent the same machine but with a different port. In this way, it is possible to be sure of where the datum is received and then use it correctly in the Simulink model. As for the path section, communication with Savona is one-way while that with Bari is bi-directional.

B.3 Configuration of the VILLAS node for the test with Genoa

The configuration file used for the remote co-simulation with UniGe (Section 4.4) is shown in Listing B.3.

```

1 nodes = {
2
3     politico-opal = {
4         type = "socket"
5         layer = "udp"
6         format = "villas.binary"
7
8         in = {
9             address = "*:12001"
10        }
11
12        out = {
13            address = "IP_address_político-opal:12001"
14        }
15    }
16
17    genova-villas = {
18        type = "socket"
19        layer = "udp"
20        format = "villas.binary"
21
22        in = {
23            address = "*:12000"
24
25            hooks = ( { type = "stats" } )
26        }
27
28        out = {
29            address = "IP_address_genova-villas:12004"
30        }
31    }
32 }
33

```

```

34 paths = (
35     {
36         in = "polito-opal"
37         out = "genova-villas"
38     },
39     {
40         in = "genova-villas"
41         out = "polito-opal"
42     }
43 )

```

Listing B.3: Configuration of the VILLAS node for the test with Genoa

This configuration file is similar to the one described in Appendix A.3 and used for the tests with Bari. The main differences lie in the different remote IP address and the ports used to exchange data between the two remote sites. The port used for the communication between the Turin's VILLAS node and `polito-opal` is the port 12001, the port used to receive data from the Genoa's VILLAS node is the port 12000 and the port used to send data to Genoa's VILLAS node is the port 12004. This configuration made it possible to verify the flexibility of the ports that can be used for communication.

Appendix C

Tests on the HV transmission network

C.1 Simulink model of the HV transmission network

In this section the overall Simulink model of the HV transmission network, described in Chapter 5 is presented.

- **Area 1:** in this area the main buses are bus 1 and bus 2. Generator 1 is connected to bus 1 via bus 9 and generator 2 is connected to bus 2 via bus 10. The connection between the two buses consists of a 100 km long line. Figure C.1 and Figure C.2 show the implementation of the buses included in area 1.
- **Area 2:** in this area the main buses are bus 3, bus 4 and bus 5. The generator 3 is connected to bus 3 via bus 11. The connection between bus 3 and bus 4 consists of a 100 km long line while bus 5 is connected to bus 4 by a 300 km long line. Figure C.3 and Figure C.4 show the implementation of the buses included in Area 2.
- **Area 3:** in this area the main bus is the bus 2. The generator 4 is connected to bus 2 via bus 12. The Figure C.5 shows the implementation of Area 3.

The three areas are connected to each other through long transmission lines modeled according to Section 5.3.1. In particular, the bus 2 of Area 1 and the bus 5 of Area 2 are connected through a 300 km long line. Bus 1 of Area 1 and bus 6 of Area 3 are connected through a 300 km long line. Bus 6 of Area 3 and bus 3 of Area 2 are connected through a 300 km long line and finally bus 2 of Area 1 and bus 5 of Area 2 are connected through a 600 km long line. Please note that the rated

voltage of the lines is 220 kV except for the longest line that connect bus 7 (Area 1) to bus 8 (Area 2) which is sized for a rated voltage of 380 kV.

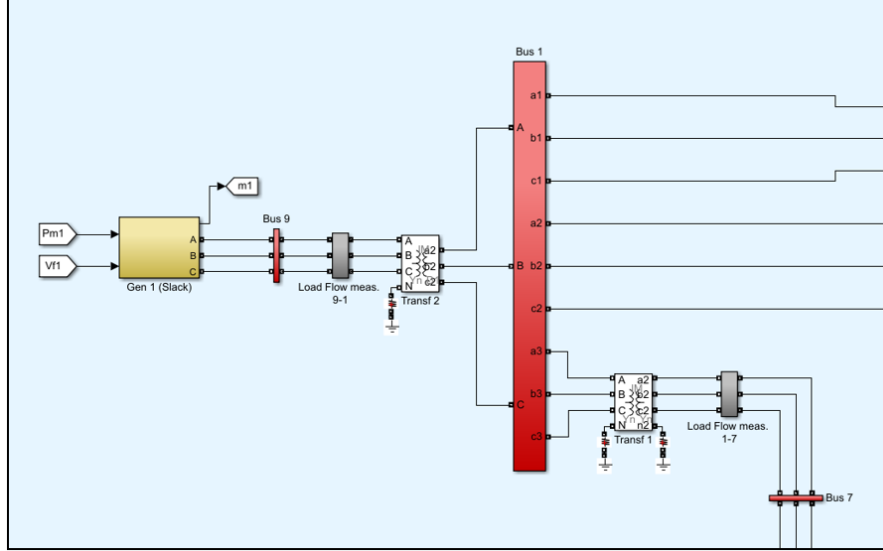


Figure C.1: Bus 1 (Area 1)

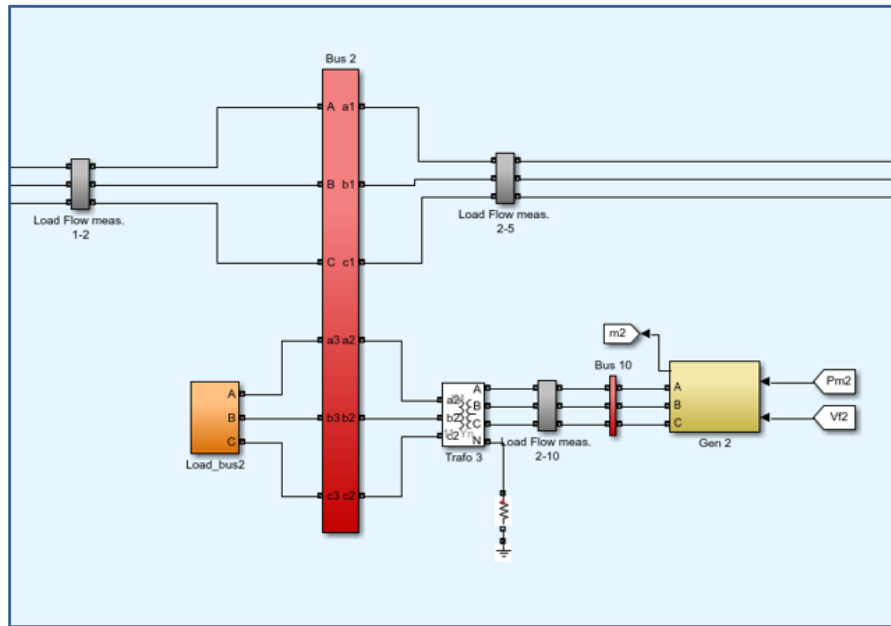


Figure C.2: Bus 2 (Area 1)

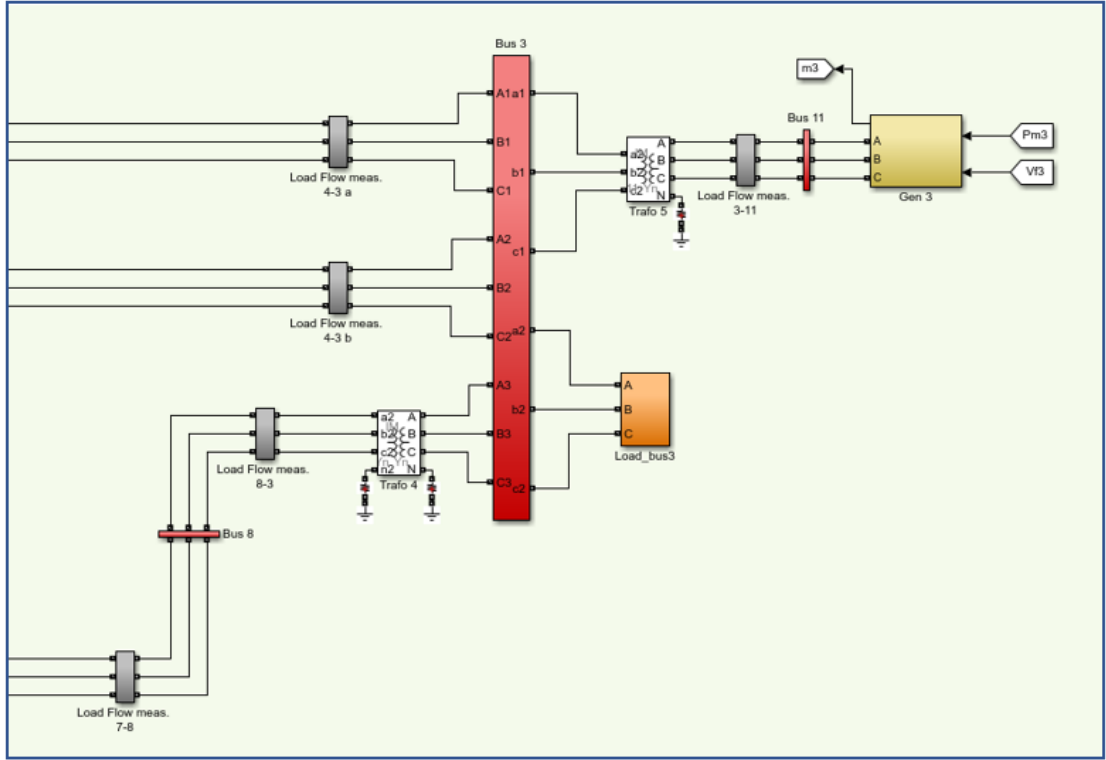


Figure C.3: Bus 3 (Area 2)

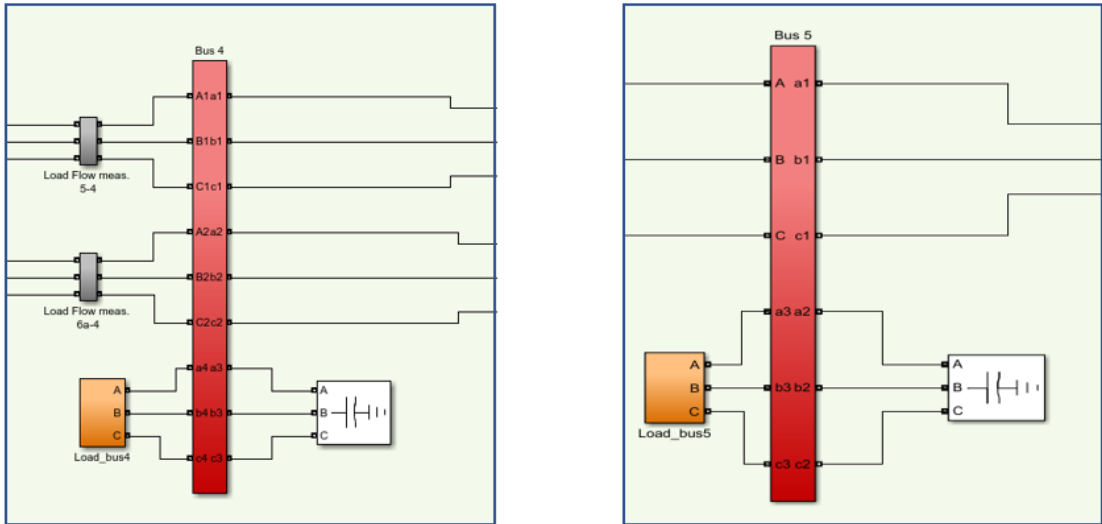


Figure C.4: Bus 4 and bus 5 (Area 2)

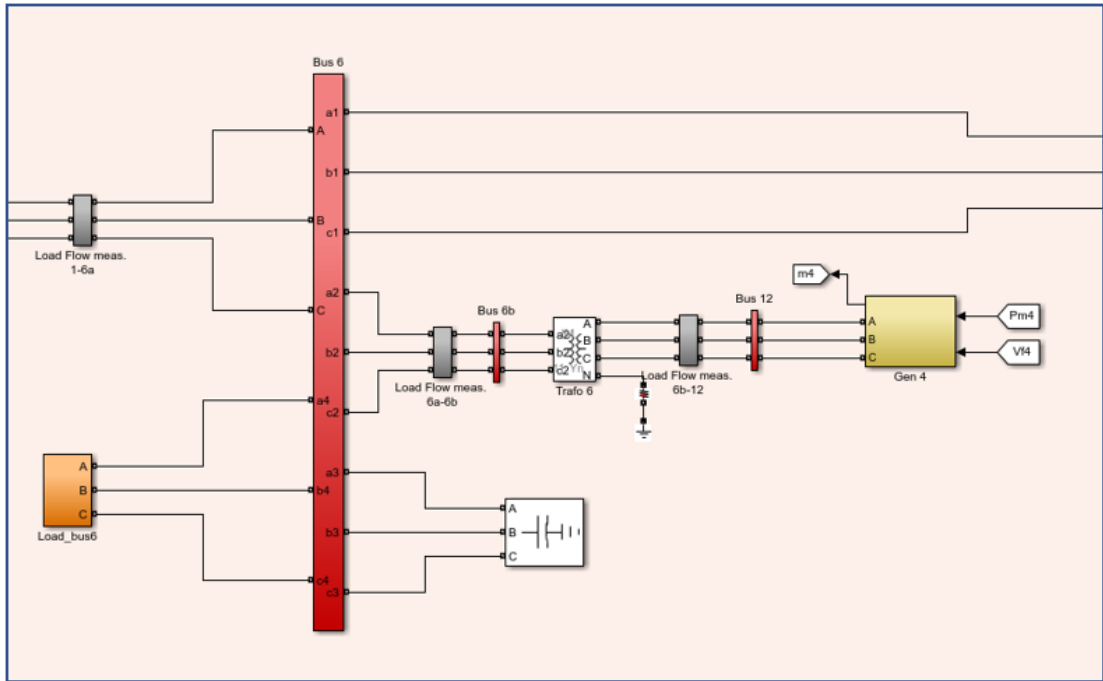


Figure C.5: Bus 6 (Area 3)

C.2 Decoupled Simulink model of the HV transmission network

The Simulink model implemented to run the simulations described in Chapter 6 is here reported. Figure C.6 shows the Simulink model decoupled into several computation cores of the OPAL-RT[®] simulator. Three computation cores are used, in particular:

- **SS_Area1** contains the portion of the network belonging to Area 1 and described in Section C.1.
- **SS_Area2** contains the portion of the network belonging to Area 2 and described in Section C.1.
- **SS_Area3** contains the portion of the network belonging to Area 3 and described in Section C.1.
- **SM_Ctrl** contains the excitation systems, the steam turbines and the governors of the generators implemented according to Section 5.3.3. In this sub-system also the measurements of the frequencies are obtained.
- **SC_Console** contains the elements to view some simulation data and the manual switches to activate the load event and the recording of the measurements.

A greater number of computational cores allows for smaller time-step. The time-step used to simulate the HV transmission network is $t_s = 50 \mu\text{s}$. The Slave Subsystems (indicated with the prefix *SS*) are decoupled using the *ARTEMiS Distributed Parameters Line* block as described in Section 6.1 while the Master Subsystem (indicated with the prefix *SM*) and the Slave Subsystems (*SS*) are decoupled through the *OpComm* block as no physical connector is needed. Please note that physical connectors are used whenever it is necessary to decouple physical systems where there are blocks of Simulink Simscape library. In all the co-simulations performed in Chapter 4, as the physical part of the model was all contained in a single subsystem, it was not necessary to use special blocks for the decoupling.

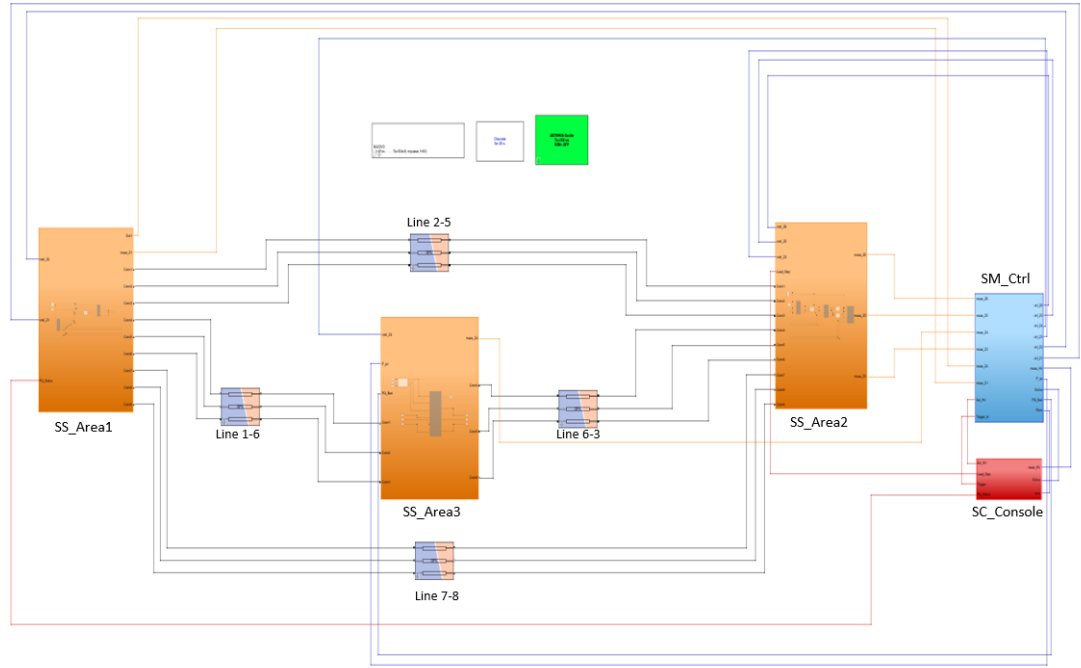


Figure C.6: Decoupled Simulink model of the HV transmission network

C.3 Configuration of the VILLAS node for the remote tests

The configuration file for the VILLAS node located in Turin used for the remote co-simulation described in Section 6.3 is reported in Listing C.1.

```

1 nodes = {
2     savona-villas = {
3         type = "socket"
4         layer = "udp"
5         format = "villas.binary"
6         in = {
7             address = "*:12002"
8             hooks = ( { type = "stats" } )
9         }
10        out = {
11            address = "IP_address_savona-villas:12002"
12        }
13    }
14    politico-opall = {

```

```

15     type = "socket"
16     layer = "udp"
17     format = "villas.binary"
18     in = {
19         address = "*:12006"
20     }
21     out = {
22         address = "IP_address_polito-opal:12006"
23     }
24 }
25 politico-opal2 = {
26     type = "socket"
27     layer = "udp"
28     format= "villas.binary"
29     in={
30         address = "*:12001"
31     }
32     out = {
33         address = "IP_address_polito-opal:12001"
34     }
35 }
36 politico-opal3 = {
37     type = "socket"
38     layer = "udp"
39     format= "villas.binary"
40     in={
41         address = "*:12005"
42     }
43     out = {
44         address = "IP_address_polito-opal:12005"
45     }
46 }
47 bari-villas = {
48     type = "socket"
49     layer = "udp"
50     format = "villas.binary"
51     in = {
52         address = "*:12003"
53         hooks = ( { type = "stats" } )
54     }
55     out = {
56         address = "IP_address_bari-villas:12003"
57     }
58 }
59 genova-villas = {
60     type = "socket"
61     layer = "udp"
62     format = "villas.binary"
63     in = {

```

```

64         address = " *:12000 "
65         hooks = ( { type = "stats" } )
66     }
67     out = {
68         address = "IP_address_genova-villas:12004"
69     }
70 }
71 }
72
73 paths = (
74 {
75     in = "savona-villas"
76     out = "polito-opal1"
77 },
78 {
79     in = "polito-opal2.data[0-1]"
80     out = "genova-villas"
81 },
82 {
83     in = "genova-villas"
84     out = "polito-opal2"
85 },
86 {
87     in = "polito-opal2.data[2-3]"
88     out = "bari-villas"
89 },
90 {
91     in = "bari-villas"
92     out = "polito-opal3"
93 }
94 )

```

Listing C.1: Configuration of the VILLAS node for the remote tests on the HV trasmission network

In this test six nodes are implemented:

- **savona-villas** is the remote VILLAS node installed and configured in Savona that sends the active power value produced by the PV system.
- **polito-opal1** is the first VILLAS node configured in Turin and includes an *OpAsyncRecv* in the Simulink model implemented in Turin, this node is used to receive the value of the active power measured in Savona.
- **polito-opal2** is the second VILLAS node configured in Turin and is represented by an *OpAsyncSend* and an *OpAsyncRecv* in the Turin Simulink model. This node is used to send the amplitude of the voltage and the frequency to

the remote node configured in Genoa and Bari and to receive the value of the active and reactive power produced by the wind system modelled in Genoa.

- **polito-opal3** is the third node configured in Turin and includes an *OpAsyn-cRecv* in the Turin Simulink model. It is used to receive the active and reactive power from the Bari's microgrid.
- **genova-villas** is the remote VILLAS node installed and is configured in Genoa. It sends the value of the active and reactive power produced by the wind farm.
- **bari-villas** is the VILLAS node installed and configured in Bari used to send the value of the active and reactive power absorbed by the Bari microgrid.

Please note that the paths between the various nodes are all bi-directional with the exception of the Savona's VILLAS node that is one-way.

Bibliography

- [1] C. F. Covrig et al. *A European Platform for Distributed Real Time Modelling and Simulation of Emerging Electricity Systems*. Tech. rep. EUR 27941 EN. Westerduinweg 3, 1755 LE, Petten, the Netherlands: JRC Science Hub, 2016 (cit. on pp. 1–3, 5–8, 11).
- [2] OPAL-RT Technology. *RT Lab - Online Course*. URL: https://www.opal-rt.com/opal_tutorial/preparing-simulink-model-real-time-execution/ (cit. on pp. 3, 4, 21–23).
- [3] S. Bruno, C. Rodio G. Giannoccaro M. La Scala, E. Bompard, G. Chicco, A. Mazza, and E. Pons. «Le opportunità della simulazione Real-Time, Cyber-fisica e remota». In: 7-8 (July 1999), pp. 38–55 (cit. on pp. 5, 8, 10, 13–15, 20, 40, 43).
- [4] Thomas Kirk. *Real-Time Simulation for Energy Storage Applications including Battery Management System Testing*. Tech. rep. University of California, UcRiverside: Winstone Chung - Global Energy Center, 2019 (cit. on pp. 6, 7).
- [5] S. Frittoli. «Remote interconnection of real-time simulators: performance analysis and case studies». MA thesis. Turin: Politecnico di Torino, 2021 (cit. on pp. 6, 7, 15, 49, 50).
- [6] M. Stevic, S. Vogel, and A. Monti. «From Monolithic to Geographically Distributed Simulation of HVdc Systems». In: *2018 IEEE 19th Workshop on Control and Modeling for Power Electronics (COMPEL)*. Padua, Italy, 2018, pp. 1–5 (cit. on pp. 9, 12).
- [7] A. Monti et al. «A Global Real-Time Superlab:Enabling high penetration of power electronics in the electric grid». In: vol. 5. 2018, pp. 35–44 (cit. on pp. 10, 11, 14).
- [8] M. Stevic, A. Estebarsari, S. Vogel, E. Pons, E. Bompard, M. Masera, and A. Monti. «Multi-site European framework for real-time co-simulation of power systems». In: *IET Journals* (Apr. 2017), pp. 1–10 (cit. on pp. 11, 28, 29).

- [9] R. Liu, M. Mohanpurkar, M. Panwar, Rob Hovsapian, A. Srivastava, and S. Suryanarayanan. «Geographically distributed real-time digital simulations using linear prediction». In: *International Journal of Electrical Power and Energy Systems* 84 (2017), pp. 308–317 (cit. on pp. 12, 59).
- [10] E. Bompard, S. Bruno, S. Frittoli, G. Giannoccaro, M. La Scala, A. Mazza, E. Pons, and C. Rodio. «Remote PHIL Distributed Co-Simulation Lab for TSO-DSO-Customer Coordination Studies». In: 2020 AEIT International Annual Conference (AEIT) (2018), pp. 1–6 (cit. on pp. 12, 15, 26).
- [11] S. Vogel, V. S. Rajkumar, H. T. Nguyen, M. Stevic, R. Bhandia, K. Heussen, P. Palensky, and A. Monti. «Improvements to the Co-simulation Interface for Geographically Distributed Real-time Simulation». In: vol. IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society, Lisbon, Portugal. 2019, pp. 6655–6662 (cit. on p. 12).
- [12] E. Bompard, S. Bruno, A. Cordoba-Pacheco, C. 13 Diaz-Londono, G. Giannoccaro, M. La Scala, A. Mazza, and Enrico Pons. «Connecting in Real-time Power System Labs: an Italian Test-case». In: *2020 IEEE International Conference on Environment and Electrical Engineering and 2020 IEEE Industrial and Commercial Power Systems Europe* (2020), pp. 1–6 (cit. on pp. 15, 26).
- [13] Raymond P. Canale Steven C. Chapra. *Metodi Numerici per l'ingegneria*. Ed. by EDIGEO. First. Milan, Italy: McGRAW-HILL, 1988 (cit. on pp. 18, 19).
- [14] MathWorks. *Matlab/Simulink - Solver*. URL: <https://it.mathworks.com/help/simulink/gui/solver.html> (cit. on p. 19).
- [15] MathWorks. *Matlab/Simulink - Solvers for Real-Time Simulation*. URL: <https://it.mathworks.com/help/physmod/simscape/ug/solvers-for-real-time-simulation.html> (cit. on p. 19).
- [16] OPAL-RT Technology. *User Documentation*. URL: <https://wiki.opal-rt.com/display/RD/OpComm> (cit. on p. 23).
- [17] OPAL-RT Technology. *User Documentation*. URL: <https://wiki.opal-rt.com/display/RD/OpAsyncRecv> (cit. on p. 24).
- [18] OPAL-RT Technology. *User Documentation*. URL: <https://wiki.opal-rt.com/display/RD/OpAsyncSend> (cit. on p. 24).
- [19] OPAL-RT Technology. *User Documentation*. URL: <https://wiki.opal-rt.com/display/RD/IPSocket+OpIPSocketCtrl> (cit. on p. 25).
- [20] FEIN Aachen. *VILLASframework - VILLASnode*. URL: <https://www.fein-aachen.org/en/projects/villas-node/> (cit. on pp. 27, 29).
- [21] FEIN Aachen. *VILLASframework - Concept*. URL: <https://villas.fein-aachen.org/doc/node-concept.html> (cit. on p. 28).

- [22] FEIN Aachen. *VILLASframework - Custom VILLAS binary*. URL: <https://villas.fein-aachen.org/doc/node-format-villas-binary.html> (cit. on pp. 29, 30).
- [23] FEIN Aachen. *VILLASframework - Node-types*. URL: <https://villas.fein-aachen.org/doc/node-types.html> (cit. on p. 30).
- [24] FEIN Aachen. *VILLASframework - Socket*. URL: <https://villas.fein-aachen.org/doc/node-type-socket.html> (cit. on p. 30).
- [25] FEIN Aachen. *VILLASframework - Hooks-types*. URL: <https://villas.fein-aachen.org/doc/hook-types.html> (cit. on p. 31).
- [26] Kai Strunz et al. *Benchmark Systems for Network Integration of Renewable and Distributed Energy Resources*. Tech. rep. 575. CIGRE, 2014 (cit. on pp. 66–70, 78).
- [27] H.W. Dommel. «Digital Computer Solution of Electromagnetic Transients in Single- and Multiphase Networks». In: *IEEE Transactions on Power Apparatus and Systems* PAS. 88.4 (1969), pp. 388–399 (cit. on pp. 71, 72).
- [28] A. Moeini and I. Kamawa. «Synchronous Machine Stability Model,an Update to IEEE Std 1110-2002 Data Translation Technique». In: *2018 IEEE Power and Energy Society General Meeting (PESGM)* (2018), pp. 1–5 (cit. on p. 76).
- [29] Francesco Iliceto. *Impianti Elettrici*. 1. Bologna: Patron Editore, 1984 (cit. on pp. 77, 85, 86).
- [30] Krause. *P.C. Analysis of Electric Machinery*. New York: McGraw-Hill, 1986 (cit. on p. 78).
- [31] P. Kundur. *Power System Stability and Control*. New York: McGraw-Hill, 1994 (cit. on pp. 79, 81).
- [32] «Recommended Practice for Excitation System Models for Power System Stability Studies». In: *IEEE Std 421.5-2016 (Revision of IEEE Std 421.5-2005)* (2016), pp. 1–207 (cit. on pp. 79, 80).
- [33] IEEE commitee report. «Dynamic models for steam and hydro turbines in power system studies». In: *IEEE Transactions on Power Apparatus and Systems* PAS-92 No. 6 (1973), pp. 1904–1915 (cit. on pp. 81, 82).
- [34] OPAL RT Technology. «ARTEMiS Distributed Parmeters Line». In: *User Documentation* (). URL: <https://wiki.opal-rt.com/display/Artemis/ARTEMiS+Distributed+Parameters+Line> (cit. on pp. 87, 88).
- [35] H. Hooshyar, L. Vanfretti, and C. Dufour. «Delay-free parallelization for real-time simulation of a large active distribution grid model». In: *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society* (2016), pp. 6278–6284 (cit. on p. 87).

- [36] A. Monti, F. Milano, E. Bompard, and X. Guillard. *Converter-Based Dynamics and Control of Modern Power Systems*. 1. Academic Press, 2020 (cit. on pp. 89, 91, 94).
- [37] FEIN Aachen. *VILLASframework*. URL: <https://villas.fein-aachen.org/doc/node-client-asyncip.html> (cit. on p. 108).