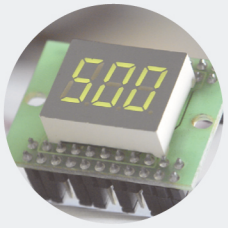
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES






Laboratório Remoto de Eletrónica

INÊS BEATRIZ DA SILVA RODRIGUES
Setembro de 2023

POLITÉCNICO
DO PORTO

Politécnico do Porto
Instituto Superior de Engenharia do Porto

Remote Laboratory For Electronics

Inês Beatriz Silva Rodrigues

Master in Electrical and Computer Engineering
Specialization Area of Automation and Systems

Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto

September, 2023

This dissertation partially satisfies the requirements of the Thesis/Dissertation course of the program Master in Electrical and Computer Engineering, Specialization Area of Automation and Systems.

Candidate: Inês Beatriz Silva Rodrigues, No. 1181192,
1181192@isep.ipp.pt

Scientific Guidance: Prof. André Fidalgo, anf@isep.ipp.pt

Scientific Co-Guidance: Prof. Ricardo Costa, rjc@isep.ipp.pt

Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto
Rua Dr. António Bernardino de Almeida, 431, 4200–072 Porto

September, 2023

Abstract

The realisation of practical experiments in a laboratory environment is essential in the teaching of engineering and other scientific areas. Thus, remote laboratories become very important in universities, since they allow their use from any location, reduce costs and maintenance, allow resource sharing and the use of all equipment by all students.

Therefore, this thesis focuses on the study and development of a remote electronics laboratory. In short, the aim of the laboratory developed is to allow students to carry out electronic experiments remotely and visualise the values recorded by the measuring instruments.

The system developed consists of a circuit with the components needed for the experiments, a microcontroller, power supplies, measuring instruments, two interfaces (student and teacher/technician), and a computer (server).

The circuit is made up of relays which, with their switching capability, connect or disconnect the components, sources and instruments to the circuit. The circuit developed is flexible in that the components can be replaced by the teacher/technician and can be connected to any node in the circuit via jumpers, also by the teacher/technician, taking into account the care required for the changes made. On the other hand, the current measurement method is limiting or implies a high expansion of the circuit for greater flexibility of this method. The power supplies and measuring instruments are all built into a single device, which facilitates their use and makes it possible to interconnect them via software with the interface developed. The interfaces allow the student or teacher to choose the circuit they want to test and visualise the desired results in real-time or after the experiment has been carried out, as chosen by the user. The microcontroller's function is to receive the string sent by the computer (the server where the interfaces are located) and control the relays so that the circuit desired by the user is formed.

A great advantage of this laboratory is the possibility of testing the correct functioning of the circuit's relays via the teacher's interface, making it easier and less time-consuming to check for errors in the event of a circuit malfunction.

The experiments tested in this laboratory are those that are the basis of electronics, namely Ohm's law, current and voltage dividers, half-wave rectifiers and Kirchhoff's laws. Other experiments can be tested given the flexibility of this laboratory, but it is necessary to study the use of components not used in the validation

carried out here beforehand.

The functional prototype developed was assembled on various breadboards and tested under these conditions. Testing the system made it possible to validate the correct operation of the functionalities implemented.

Keywords: Remote laboratory, electronic, relays, VirtualBench, LabVIEW.

Resumo

A realização de experiências práticas em ambiente laboratorial, é essencial no ensino de engenharia e outras áreas científicas. Deste modo, os laboratórios remotos tornam-se bastante importantes nas universidades, uma vez que permitem a sua utilização de qualquer localização, redução de custos e manutenção, permitem a partilha de recursos e o uso de todos os equipamentos por todos os alunos.

Portanto, esta tese foca-se no estudo e desenvolvimento de um laboratório remoto de eletrónica. De um modo resumido, pretende-se que o laboratório desenvolvido permita que o aluno execute experiências eletrónicas remotamente e visualize os valores registados pelos instrumentos de medição.

O sistema desenvolvido é composto um circuito com os componentes necessários às experiências, um microcontrolador, fontes de alimentação, instrumentos de medição, por duas interfaces (aluno e professor/técnico), e um computador (servidor).

O circuito é composto por relés que, com a sua capacidade de comutação, conectam ou disconectam os componentes, fontes e instrumentos ao circuito. O circuito desenvolvido é flexivel, na medida em que os componentes podem ser substituídos pelo professor/técnico e, podem ser ligados a qualquer nó do circuito através de *jumpers*, também pelo professor/técnico tendo em consideração os cuidados necessários para as alterações que efetuar. Em contrapartida, o método de medição de corrente é limitador ou implica uma elevada expansão do circuito para uma maior flexibilidade deste método. As fontes de alimentação e instrumentos de medição estão todos embutidos num único dispositivo, denoninado de VirtualBench, o que facilita o seu uso, e torna possível a sua interligação por software com a interface desenvolvida. As interfaces, desenvolvidas em LabVIEW, permitem que o aluno ou o professor escolham o circuito que pretendem testar e, visualizem os resultados pretendidos em tempo real ou depois de realizada a experiência conforme escolhido pelo utilizador. O microncontrolador tem como função receber a string enviada pelo computador (servidor onde estão as interfaces) e, controlar os relés para que se forme o circuito pretendido pelo utilizador.

Uma grande vantagem deste laboratório, é a possibilidade de testar o correto funcionamento dos relés do circuito, através da interface do professor, facilitando e diminuindo o tempo de verificação de erros em caso de mau funcionamento do circuito.

As experiências testadas neste laboratório são as experiências que estão na base

da eletrónica, sendo elas: a lei de Ohm, divisor de corrente e tensão, retificador de meia onda e leis de Kirchhoff. Outras experiências podem ser testadas dada a flexibilidade deste laboratório, contudo é necessário um estudo prévio para a utilização de componentes não utilizados na validação aqui efetuada.

O protótipo funcional desenvolvido foi montado em várias breadboards e testado nestas condições. O teste ao sistema permitiu validar o correto funcionamento das funcionalidades implementadas.

Palavras-Chave: Laboratório remoto, eletrónica, relés, VirtualBench, LabVIEW.

Contents

List of Figures

Listings

List of Acronyms

AC	Alternating Current
BTH	Blekinge Institute of Technology
COTS	Commercial Off-The-Shelf
CPU	Central Processing Unit
DC	Direct Current
DMM	Digital Multimeter
DPST	Double Pole Single Throw
DUT	Device Under Test
EDA	Electronic Design Automation
FGEN	Function Generator
GND	Ground
GPIB	General Purpose Interface Bus
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
I^2C	Inter-Integrated Circuit
IC	Integrated Circuit
IVI	Interchangeable Virtual Instruments
LMS	Learning Management System
LXI	LAN eXtensions for Instrumentation
NI	National Instruments
PCB	Printed circuit board

PHP	Hypertext Preprocessor
PID	Proportional – Integral – Derivative
PSLab	Pocket Science Lab
PXI	PCI eXtensions for Instrumentation
SPI	Serial Peripheral Interface
SPST	Single Pole Single Throw
TCP/IP	Transmission Control Protocol/Internet Protocol
USB	Universal Serial Bus
VI	Virtual Instrument
VISIR	Virtual Instruments in Reality
XML	Extensible Markup Language

Chapter 1

Introduction

This chapter describes the motivations related to the theme addressed in this thesis. To this end, a contextualisation is presented. Then the proposed objectives are presented and, in the end, the organization of the document is described briefly and objectively.

1.1 Context

An engineer's main goal is to manipulate materials, energy and information to add value to mankind. Experimental work is one of the most crucial features for science, technology and engineering students. This is because some concepts in these areas can be very abstract when studied in theory, and when applied in laboratory experiments, knowledge of the concepts becomes more understandable. Laboratories have been crucial to engineering undergraduate programs. A physical lab allows students to confirm that calculation methods based on mathematical models correspond with real life and also enables them to verify the limitations of these models. Practical work also provides students with the experience to cope with real-life problems [1][2][3].

The traditional way of working in physical experiments is to participate in lab sessions in university laboratories where students work in teams and get help from teachers. However, throughout the years, it has been essential to use technologies that bring a better experience to students that allow them to perform the experiences even if they are not synchronous in time and space with other students and with

the university laboratories. Therefore, many academic institutions offer web-based experimentation environments that sustain virtual or remote experiments [3] [4]. The difference between virtual and remote experiments is that virtual experiments are simulations and remote experiments imply the use of the experimental equipment in a distant way [5].

Recently, due to the global pandemic caused by Covid-19, remote labs have been crucial for students. This is because it has become the only way to be in contact with laboratory practice. In many situations, it was necessary to use expensive devices that were located in university laboratories, and students accessed them remotely.

For all the advantages that these systems bring, a remote electronics laboratory has been developed. This allows the student to build his circuit remotely and to check the real values of the electrical quantities by using real measuring instruments, also remotely controlled.

1.2 Objectives

The main objective addressed in this thesis is the development of an interface that allows the user to design a circuit that is physically constructed.

In the interface, it is intended that the user:

- Choose the components, power supplies and measuring instruments in their circuit and where they will be placed;
- Change the settings of the measuring instruments;
- Visualise in real time the values of the electrical parameters;
- Run the experiment and visualise the results after releasing the resources;
- Features a circuit control mechanism to protect equipment;
- Run the relay operation test if the user is a teacher or a lab technician.

In the case of the hardware, it is intended that:

- Contain a mechanism for selecting components;
- Have a device that contains all the measuring instruments and power supplies that are common in a traditional laboratory;
- Use a microcontroller that receives a string by Universal Serial Bus (USB) from the server software and controls the selection components to build the desired circuit;
- Use methods to connect the measurement instruments using as few relays as possible.

1.3 Plan

Figure 1.1 shows a graph of the work plan. This graph shows how the various stages of this project have been divided over time. From the stages of writing this document to the development of the several parts of the hardware and software project.

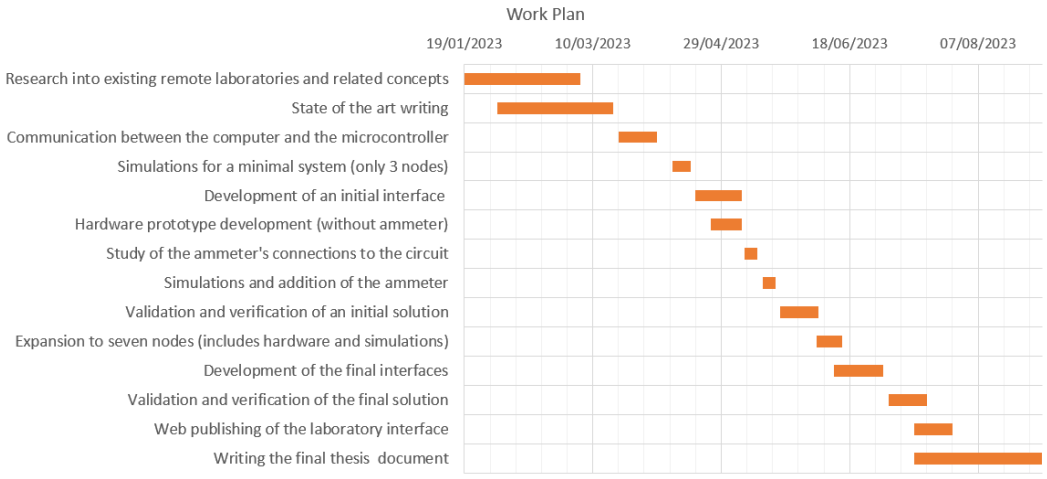


Figure 1.1: Work plan

1.4 Structure

This document is divided into six chapters. In chapter 2 the concepts of the various ways of conducting an experiment are presented. Then, the study of several existing remote electronics laboratories (some more complex than others) is carried out to study the best solutions for the developed laboratory.

Next, chapter 3 first aims to present the planned architecture and give a description of all the hardware and software that is intended to be used. In the end, it is shown a brief study of the circuits that the developed remote laboratory is intended to test.

Chapter 4 presents the entire architecture of the laboratory developed. Then, all the circuit connections of components, instrumentation and power supplies are presented. At the end, all the interface pages are shown and some code extracts related to the developed software are described.

Chapter 5 shows some of the experiment's results and simulations. To validate the results, the obtained values are compared with the theoretical and simulated ones.

Finally, chapter 6 presents the conclusions, summarising what was developed, the advantages and limitations, what can be concluded from the results obtained and what can still be improved.

Chapter 2

State of the art

In this chapter, the different types of laboratories used by students/teachers for practical experiments are presented, with an emphasis on remote laboratories. Some examples of remote electronics laboratories are described to understand the concepts of the existing laboratories. In the end, a literature review of the Virtual Instruments in Reality (VISIR) remote laboratory is presented to understand its operation to support the development of the proposed project. This lab is described in more detail since it is a more robust existing laboratory and is being utilised at ISEP.

2.1 Experimental practices

Engineers mustn't just rely on theory and know how natural phenomena occur. This knowledge of nature is obtained through laboratories used in the education of engineers [6]. Given the theme of the proposed work, this document will cover the various types of laboratories and tools that support laboratory experiments, whether laboratories require students to be in the same geographical space or not. All laboratory practices contribute to improving students' knowledge and that they recognise the difference between theoretical laws and the real world.

The practical experiences can be divided into Hands-on labs, better known as the traditional labs, online labs, which can be considered the remote labs, simulators and virtual labs and local simulators (not online). Each practical experience is used according to the intended objectives since they all have advantages and disadvantages in their use. Often they complement each other [7].

Beyond the scientific knowledge obtained through practical work, skills such as defining hypotheses or interpreting results are also developed. In short, laboratory experiments allow students to learn to apply theoretical concepts in practical situations, handle equipment and instruments and also to analyse and process the data obtained [4].

2.1.1 Definitions

The semantics concerning the term remote laboratories may present several definitions. These definitions are inconsistent and confusing, and sometimes different terms are used to characterise the same concept. To be consistent, according to [8], some metrics have been defined to characterise laboratory experiments:

- Interaction between user and the experiment;
- Experiment nature;
- User and experiment location.

Taking into account the first criterion, the types of user interaction can be classified in two different ways. The first refers to the physical control of the instrumentation devices and equipment which refers to a traditional laboratory. The second is regarding the control of the devices and instrumentation equipment through an interface that uses virtual instrumentation or virtual reality environments. The remaining criteria are usually combined, leading to the definition of four types of experience as presented in Figure 2.1, that is, two types of experiments can be defined concerning their nature, whether they involve physical devices or simulated models for the devices. Two further scenarios can be considered for the location of the user and the experience, that is, these two actors are in the same location or in separate locations [8].

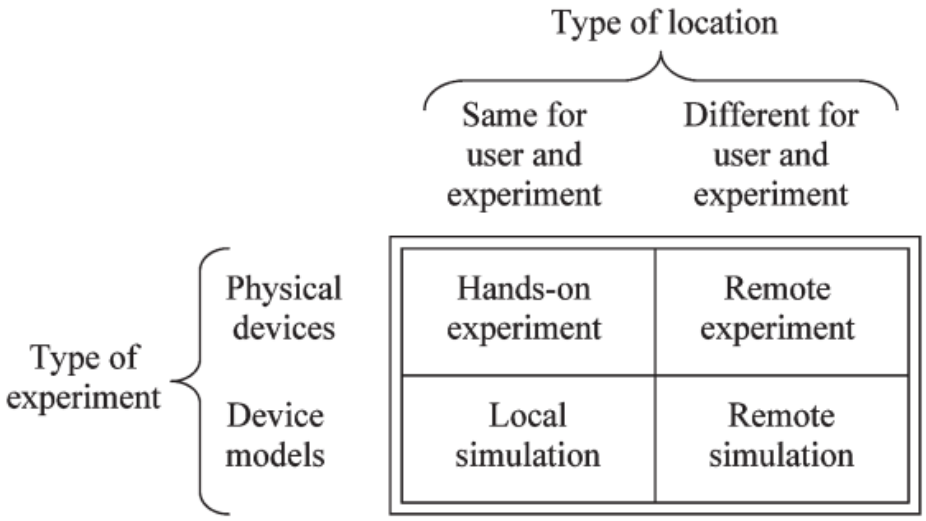


Figure 2.1: Characteristics of experiment types [8]

2.1.2 Traditional Laboratories

Traditional laboratories, which are the oldest, allow students to carry out practical experiments and consequently create technical skills for interaction with physical equipment. In addition, the actual results that appear from practical experience are important as they allow the student to study the difference between the values calculated and those obtained in the experiment [3][4].

In a traditional laboratory exercises are a successful teaching method, once the student performs experiments on his own or in teams and the teacher is in the same space and can help or give useful advice. In this kind of laboratory, the students interact directly with the equipment, which allows them to execute physical actions, such as connecting jumpers, changing the instrumentation settings etc., and receive factual responses. This is very important for students since they learn by doing and learn by failure. In this context, students will get practical notions to achieve the objectives of the experiment. The teacher is also very helpful in checking if the students take every precaution needed since when interacting with physical material care must be taken, such as not short-circuiting any components or equipment. And, they must also pay attention to the ambient conditions, such as magnetic interference, temperature, etc [2][8].

In short, with traditional laboratories, students develop their autonomous work as well as teamwork. However, the main disadvantages of these laboratories are space, time and cost issues. Traditional laboratories imply the use of appropriate space, with the necessary characteristics to install the equipment and accommodate teachers and students. Usually, they are not always open and students have to

follow the class schedule. Another important aspect is the high cost of equipment and equipment maintenance carried out by qualified technicians. These limitations mean that the number of students per class is high, as well as the number of students per group [2][9].

A hands-on solution for traditional laboratories used by undergraduate students is the use of pocket labs. Pocket labs consist of small, cheap and portable hardware devices. This solution provides the student with the opportunity to realise some experiments at home with physical contact with real hardware [4][9].

There are a variety of commercial options for hands-on experimentation outside of a physical laboratory for educational and research purposes. One example is PocketLab, which is a kit that includes various data sensors, laboratory software, and a wide range of lesson libraries. When connected to a computer, it allows users to access a variety of lessons, such as those related to physics [10].

Another commercial example is Pocket Science Lab (PSLab). PSLab develops mobile and desktop applications to receive data measurements. The application incorporates an oscilloscope, multimeter, signal generator, logic analyser, and power supply and is constantly updated to add new instruments. By adding the PSLab board and Open Standards sensors to the mobile phone or computer, a pocket laboratory is formed. It is possible to collect data on air, temperature, water quality, and radioactivity levels, among others.

The PSLab board is USB powered when connected to a mobile phone or computer which adds sensors to the device and allows connecting to hundreds of available sensors. To do this, it is needed to connect two wires to the relevant pins and carry out the measurements. All sensors following the Inter-Integrated Circuit (I^2C) protocol are compatible and there is no need to program them. For programmers, there is also the possibility to implement new functionalities (by changing firmware and/or hardware) [11].

Figure 2.2, represents the PSLab board and the software waiting to receive data as soon as sensors are connected to the board.

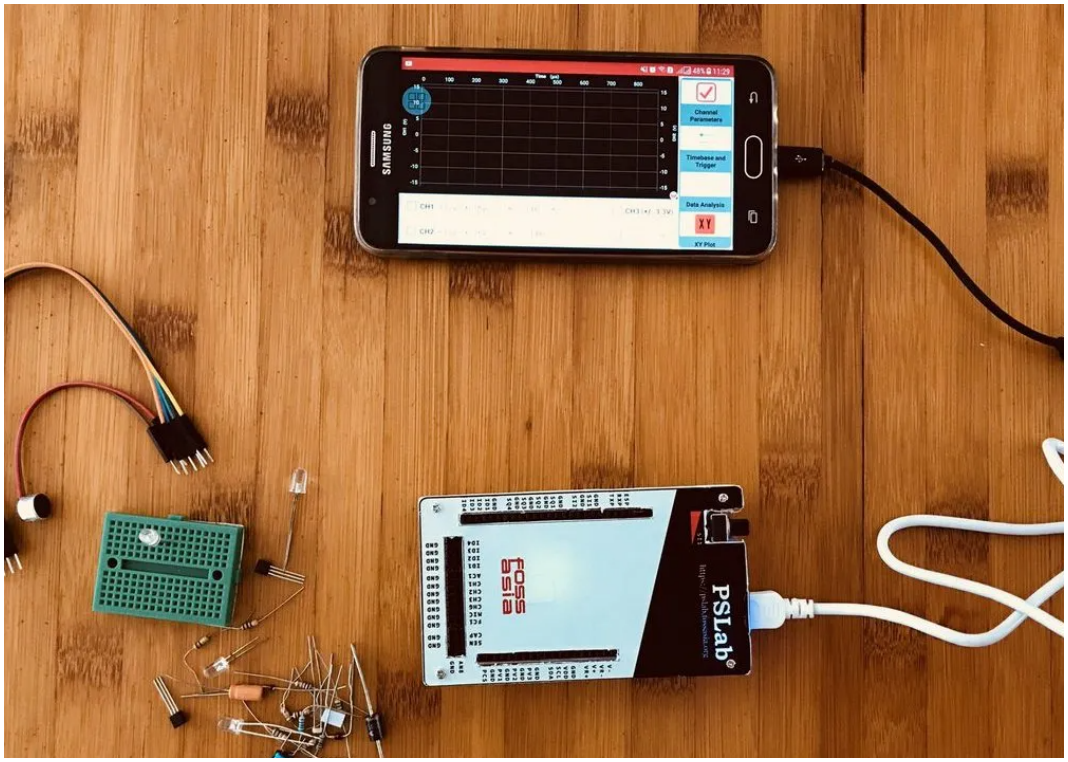


Figure 2.2: Pocket Science Lab hardware and software [11]

2.1.3 Local simulation

Local simulation is the process of using computer software to simulate a real experience. As the name implies, simulation is local since simulation software needs to be installed locally. This process allows tests to be carried out in a virtual environment before a physical circuit is assembled, making it more cost-effective and safer to understand the circuit's behaviour. To carry out the simulation, the user needs to create a representation of their circuit through components and connections and simulate the response of the circuit to different input conditions.

Local simulation is important for university students as it allows them to test circuits before physically building them in the lab. This allows them to save time and resources and to understand theoretical concepts by observing circuit behaviour in a virtual environment.

As this solution is based on mathematical models, the results obtained will always be the same for the same conditions, which in reality is not the case due to errors. Local simulation serves to complement the physical experience, however, it does not replace it [12].

Several open-source or commercial tools are used for local simulation, such as PSpice. PSpice is a simulation tool that belongs to the OrCAD collection of Electronic Design Automation (EDA). This tool can simulate and analyze the behaviour of analogue and mixed-signal circuits and verify their performance when submitted

to different conditions. This allows circuits to be properly checked before being implemented. PSpice provides libraries of standard analogue and digital components. Figure 2.3 shows an example of a transformer circuit designed in PSpice [13].

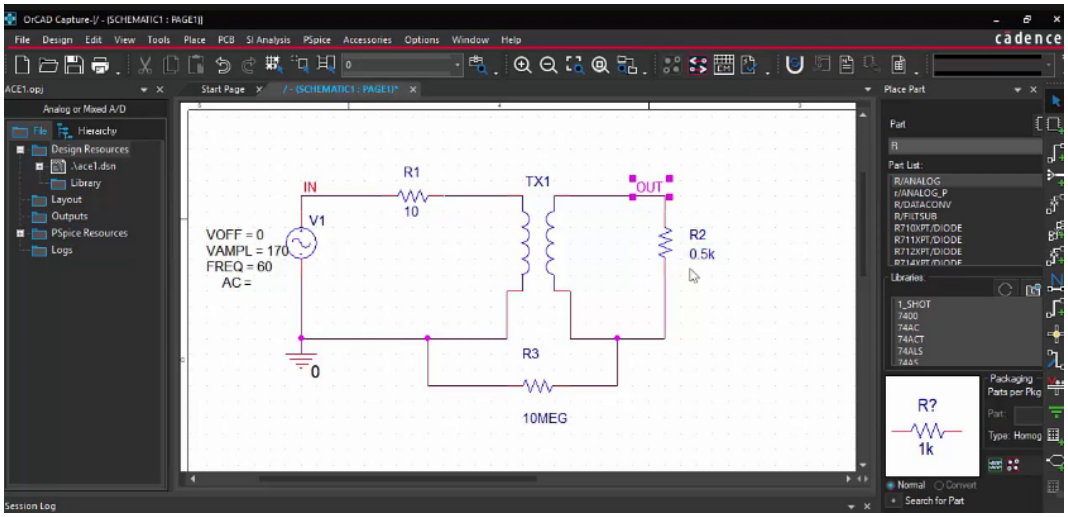


Figure 2.3: Pspice interface's example

2.1.4 Online Laboratories

The aim is to make education more and more flexible to overcome the disadvantages of traditional laboratories presented above, such as limited resources (teachers, lab time, lab stations per number of students). Flexible education means providing students with the freedom to organise their learning through the use of learning resources. Nowadays, many academic institutions provide online laboratory environments (virtual and remote laboratories). These environments allow experiments to be carried out far away from physical materials to give students more lab time. This is a solution to time limit lab sessions with in-person supervision without cost per student.

In summary, the use of online laboratories in educational courses not only reduces operating costs but also enhances student performance across all education levels. This tool is very helpful for teachers once it supports the pedagogical process. In recent years, technological advancements have overcome some of the main limitations, making these labs increasingly crucial.

One of the questions that may arise when addressing this topic is the effectiveness of virtual and remote laboratories (online laboratories) compared with traditional ones. However, most empirical studies prove that online labs and traditional labs are equally effective [14].

Online labs have advantages such as:

- High availability as they can be accessed through any location and at any time. This is important for students who are geographically distant;

- They are accessible to all, promoting their use by handicapped people;
- It can be a good solution to traditional laboratories when it comes to dangerous experiments. It is safer for students and avoids damaging materials and consequently reduces equipment costs;
- Promotes the opportunity to learn by experimenting;
- Ability to manage larger groups of users, experiments and laboratories.

The biggest limitation of virtual labs is that a single experiment typically produces the same solution or set of results, making it more difficult for teachers to verify the originality and origin of the data analysis reported by students.

2.1.4.1 Virtual Laboratories

Virtual Laboratories and local simulations are two different approaches to simulating real-world systems and situations. Virtual laboratories are web-based simulations that can be accessed through a browser, making them easily accessible and shareable by a large number of users. They offer remote access to the simulation environment without needing local software installations, but they typically have a more limited set of resources and capabilities compared to local simulations. On the other hand, local simulations run on the user's computer or local network and offer a higher level of control over the simulation environment. They also provide more computing resources and capabilities compared to virtual labs but require the installation of simulation software on the local machine.

Apart from the agility of simulating experiments from any place as long as there is a computer connected to the internet when running an experiment in these laboratories there are no concerns related to the security of the equipment when compared to remote and traditional laboratories. Although, there are some limitations like these labs don't consider environmental variables and experimental errors, so users aren't able to compare the theoretical values with experimental results since they are the same [8].

In the field of electronics, there are several examples of virtual systems that simulate electronic circuits, but EveryCircuit and Falstad are two of the most widely used.

EveryCircuit runs in a web browser (Chrome, Firefox or Microsoft Edge) and doesn't need any plug-ins, it is just required to draw the schematic with the given components, including the instrumentation. Additionally, EveryCircuit has several pre-designed circuits. Dynamic animations display voltages, currents and charges. It is possible to change the simulation parameters while the simulation is running and see how the circuit responds in real-time [15].

Figure 2.4, shows a half-wave rectifier circuit already pre-designed. It's possible to see the voltage animation in real time in the graphic above the schematic. The course has some animation to show whether there is current or not and also shows the voltage at the entry of every component.

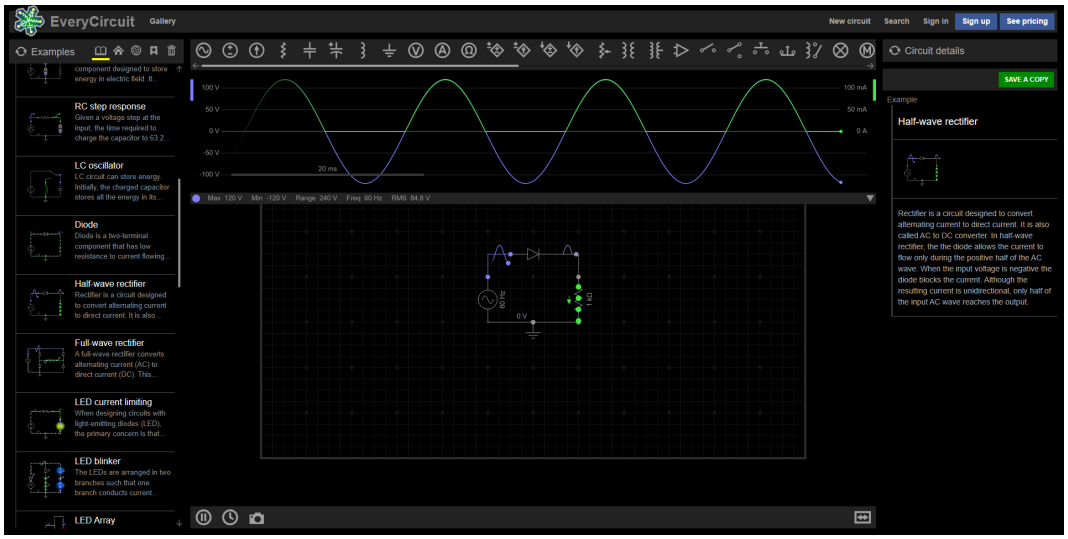


Figure 2.4: EveryCircuit simulation of half-wave rectifier

The Falstad virtual system presents several applets such as waves, acoustics, signal processing, electrodynamics, electricity and others. Focusing on the applet of electronic circuits, the Falstad is a tool to design an electronic circuit, where the user can simulate the circuit and observe how the circuit behaves. The type of components are present in the simulator and the user can design the circuit through these and change the parameters [16].

Figure 2.5, shows a capacitor's circuit and response when an alternating current drives it on Falstad. The user is also able to see the current direction through a circuit animation.

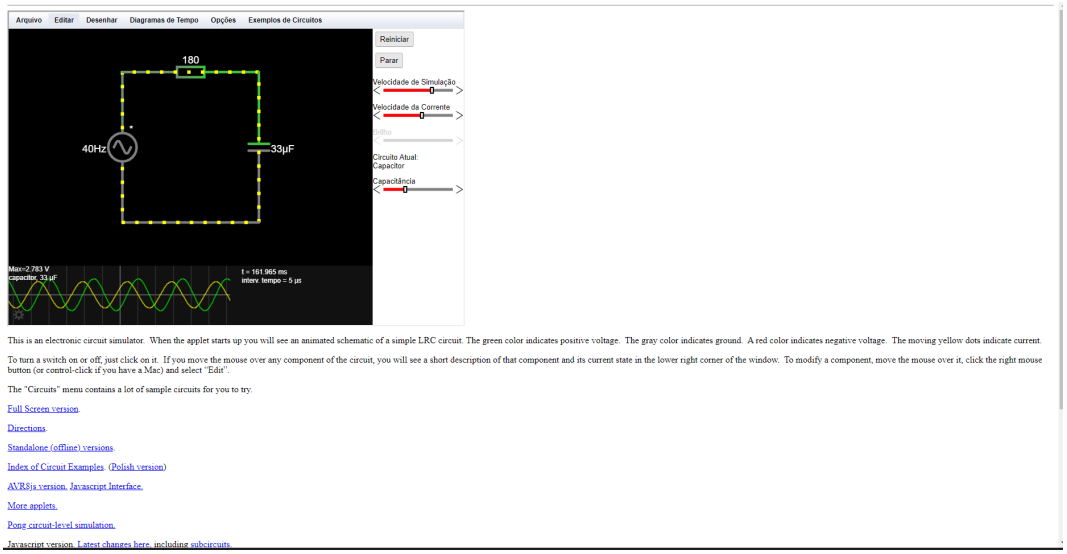


Figure 2.5: Falstad simulation of capacitor driven by alternating current

In addition to the virtual laboratories used in electronics, there are several related to science. The PhET platform brings together several virtual laboratories related to areas such as physics, chemistry, mathematics, earth sciences and biology. Figure 2.6 shows a physics laboratory representing a spring-mass system. This system has simulations that represent Hooke's law, in other words, the spring's elasticity and its displacement as a function of the force applied to it [17].

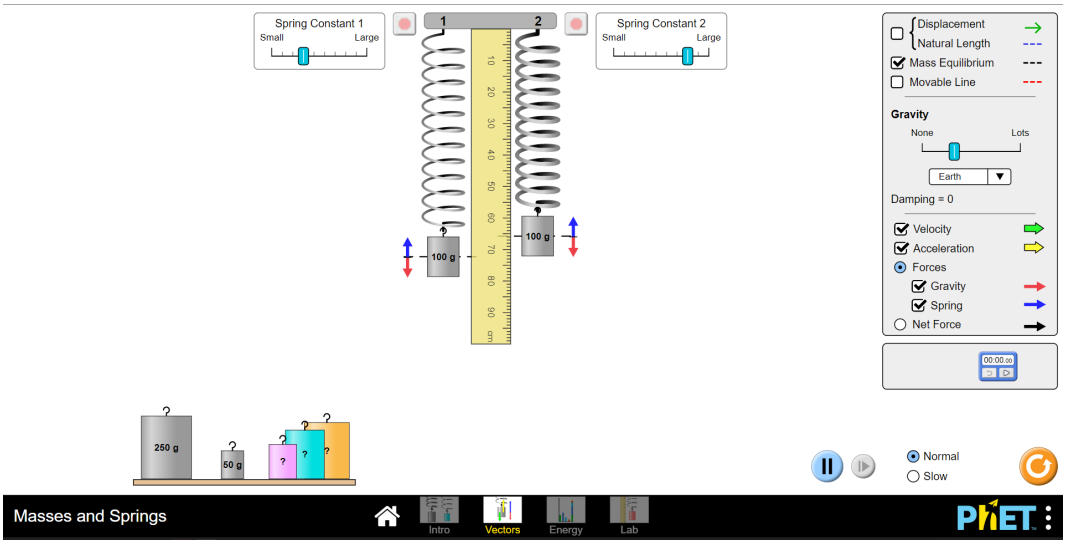


Figure 2.6: PhET spring-mass system simulation

2.1.4.2 Remote Laboratories

There is no doubt that the practical lessons provided by traditional laboratories provide better knowledge for the students, making them more familiar with the

instruments and the real world. It also makes the concepts and theories easier to understand. However, as already mentioned, there are barriers to this teaching such as high costs, administrative burden and low availability of physical laboratory use. With the fast-paced developments in technology, remote laboratories have been a solution and complement to face-to-face classes in several universities. Remote laboratories are systems made up of hardware and software that allow students to access hardware located physically at universities through the Internet as if they were in a face-to-face session. Hence, experiments are performed on actual equipment in physical laboratories that are ideally always accessible, allowing remote control of users via the internet [18].

The user side of the remote laboratory software consists of a virtual interface accessed through a web browser that allows the user to control and run the experiment. This interface also manages laboratory access. The physical part of the system is responsible for monitoring and controlling the devices and ensuring that the experience is ready to run again. Thus, these systems allow the user to carry out the tests on laboratory equipment, making the results obtained real and not calculated, unlike what happens with virtual laboratories. In conclusion, the main difference between remote laboratories and traditional ones is that the user is physically outside the same space where the physical equipment is since the experiment and its observation are performed through a set of instructions made over the internet.

Remote laboratories have advantages over traditional ones, such as the time to access the experiment. In the case of remote laboratories, they can be accessed at any time of the day every day, in the case of traditional laboratories this access already depends on the schedule defined for it. Remote laboratories allow autonomous learning and collaboration with individuals and institutions around the world. Another advantage is the decrease in equipment and maintenance costs.

Given the subject of the proposed project, it is important to make a review of existing remote electronics laboratories. To this end, the following laboratories were analysed.

- **LaboREM** - developed at Bayonne Institute of Technology (IUT), Anglet, France. It is a remote laboratory designed to be used by electronics students to build circuits using a remotely controlled robot arm [19]. Thus, students can control a robotic arm remotely to build their circuits. The learning objectives are the assembly of electronic circuits and the use of measuring instruments. LaboREM follows a Game-Like strategy to motivate students. This consists of quizzes before and after the labwork and a final test. Based on the student's performance, points are awarded which are then translated into a ranking. Figure 2.7 shows LaboREM architecture [20].

  Instead of virtual wiring, LaboREM also allows the construction of circuits by using a robotic arm to simulate the student's arm when experimenting.

The experiments that can be performed are active filters that are based on operational amplifiers, resistors and capacitors. The robot easily places the components in the circuit as they contain a magnet. The user can choose the type of filter via the interface.

LaboREM does not allow the full construction of circuits by students. This lab allows the realization of some predefined experiments that, for now, consist of four passive filters (low-pass, high-pass, band-pass and band-rejection) and four active filters (low-pass, high-pass and band-pass of Sallen-Key type, plus another second-order band-pass). These pre-made experiments can be of two types: plugs that contain all possible components where some are chosen by switches through the user interface, or they can also be plugs that contain the pre-made experiment and the user needs to place the two missing components on a board connected to the plugs with the help of the robot [21].

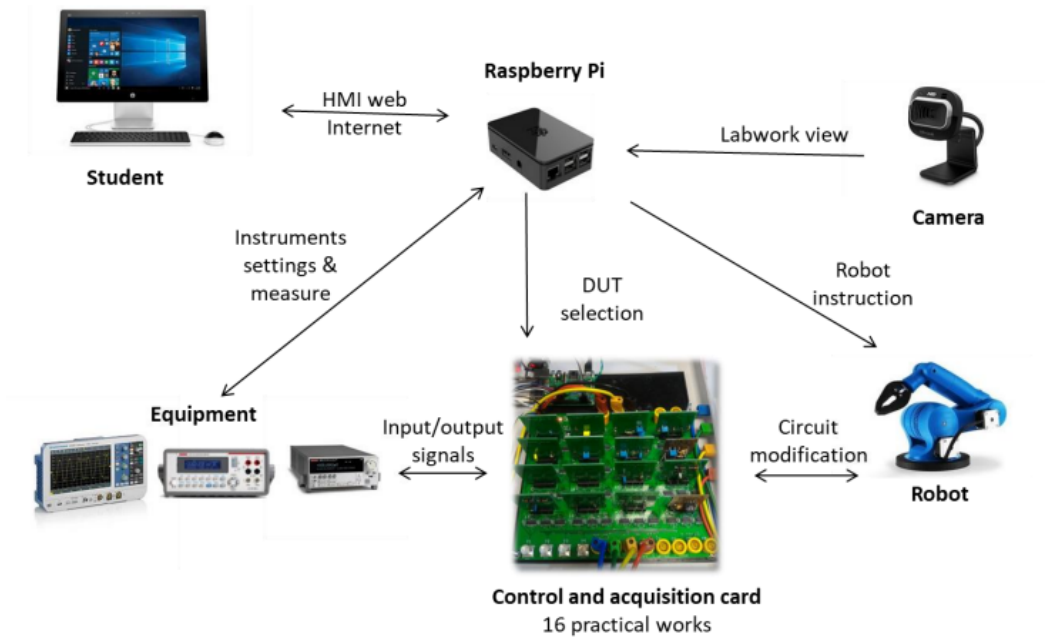


Figure 2.7: LaboREM architecture [22]

The server is embedded in a Raspberry which is a widely used solution in various areas of electronics, such as remote laboratories [22]. The system also contains a motherboard where 16 different plugs can be connected and a robotic extension board that interconnects the mobile elements of the robot. Plugs are electronic boards with pre-prepared experiments and contain switches that activate or deactivate components to change the circuit. Digital components such as an I^2C controlled digital potentiometer can also be inserted into the plugs. This adjustable resistance option is required for some practical work

such as the Wheatstone bridge . The motherboard has the function of receiving and managing all the information of the experiment that the student intends to work on. Each plug represents a Device Under Test (DUT) and can be easily replaced to adapt the experiment. Each plug has the option of being powered by different voltages to connect with circuits that have different power supplies or symmetrical power supplies. Three input voltages, one output voltage and 4 pins are provided that can be connected to external components that will be placed by the robot. The plug between the 16 is selected via a multiplexer controlled by 4 bits from Raspberry Pi GPIO's [20][21]. The motherboard's schematic is represented in Figure 2.8.

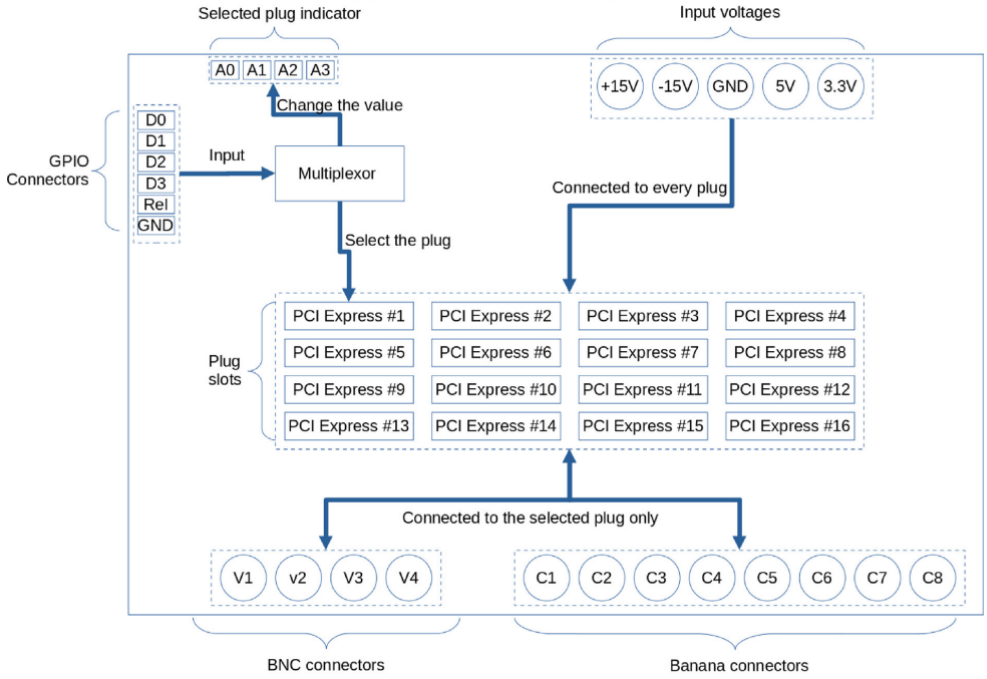


Figure 2.8: Motherboard's schematic [21]

For the students to visualise the experience they are conducting, there is a fixed camera or a drone that makes a live video. Using a robotic arm on the server side, allows the student to perform the experience dynamically through the web interface as if it were the student's arm. Besides the presented devices, the system hardware also includes measurement instruments, such as a waveform generator, multimeter, oscilloscope and power supply [19][20]. Figure 2.9 shows the LaboREM physical station indicating all the elements.

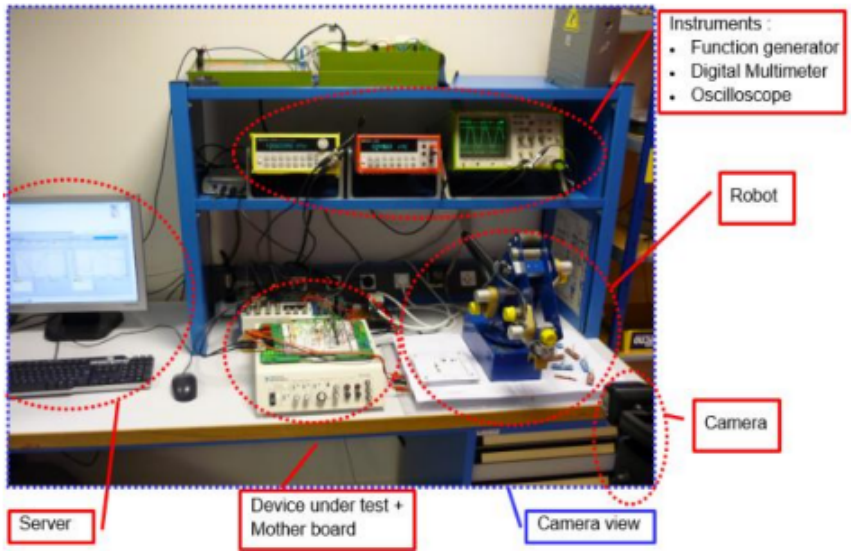
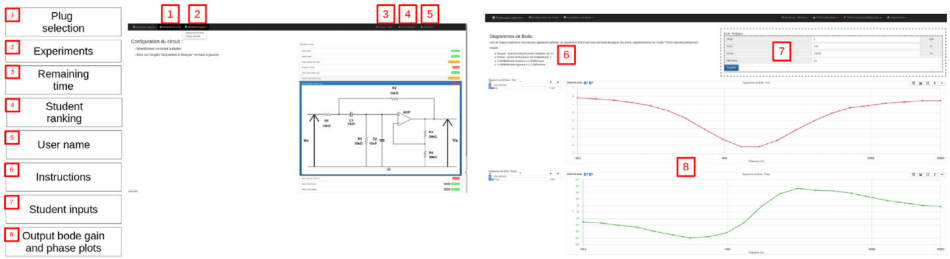


Figure 2.9: LaboREM physical station [20]

The LaboREM interface is accessed by students through the Moodle platform and, for several students connected at the same time, there is queue management. The first student in the row is the one who can access the instruments of the experiment and, as soon as another student appears in the queue, a time limit is assigned to the worker of the moment before it is handed over to the next student in the waiting line. Those in the queue can view the waiting time required to access the equipment. It is also possible for people in the waiting line to observe what the worker is doing, such as instrument configuration and curves visualization [21].

The student needs to choose one of the available experiments, Figure 2.10a, such as studying bode diagrams or analysing the response of a filter. In figure 2.10b, the response of a bode diagram experiment is displayed. It is also possible for the user to change the instrumentation parameters.



(a) (b)

Figure 2.10: Student's interface: (a) Circuit Configuration [21] and (b) Results of a Bode experiment [21].

LaboREM uses a supervisory control and data acquisition system (SCADA). The operating system used on the server is Linux (Debian distribution). The programming language used is Python and the Django framework (Python language framework) is also used, which makes the creation of websites simpler and faster. As a way to communicate with the instrumentation devices, the Django PyScada application and other open-source Python applications are used. An advantage of this software architecture is to provide a web interface compatible with all clients (computers, smartphones, etc.) [20]. Figure 2.11 shows a diagram with the software used in LaboREM.

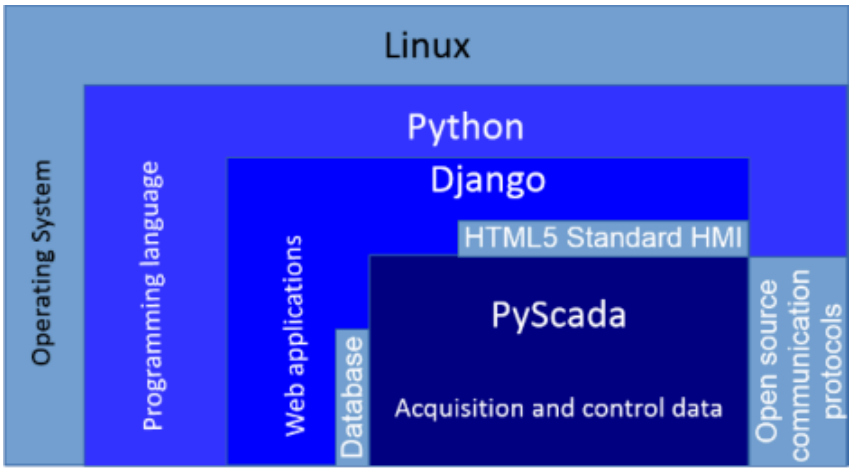


Figure 2.11: LaboREM software [20]

- **MostaLab** - developed at *Université Abdelhamid Ibn Badis Mostaganem* (Mostaganem, Algeria). This laboratory is used for electronic experiments where, briefly, it presents a switching board, controlled by a Raspberry Pi, which connects to a board containing the specific components for each experiment [23].

  Figure 2.12 shows MostaLab's architecture. This architecture is based on three components: the laboratory manager, server, and user interface.

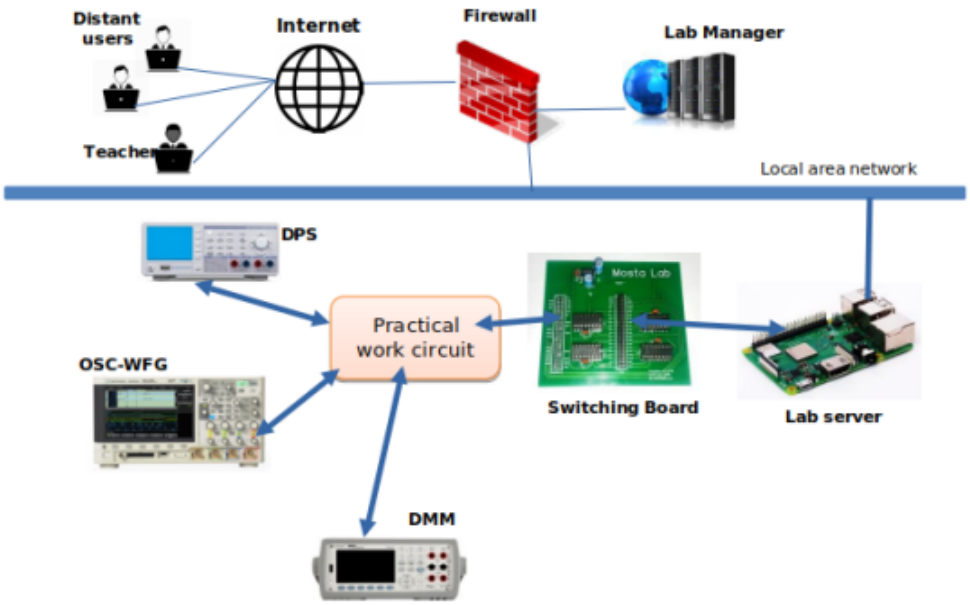


Figure 2.12: MostaLab architecture [23]

The laboratory manager is the entry point of several access requests. This manager has the function of managing and allocating the resources of the remote laboratories, controlling and monitoring the remote laboratories by registering in a file the status of each remote laboratory and registering the actions performed by the users, sending requests to the laboratory server to verify or modify the status of the laboratory.

The lab server (Raspberry Pi), can manage access to remote lab hardware. In this way, it is responsible for sending the requests made to the equipment present in the experiment and returning the answers to the laboratory manager. In addition, the server implements an application for communicating with the instruments.

The user interface is a graphical interface accessed through the internet browser, i.e. it does not require the installation of any software by the user. It is through this interface that users send requests to the laboratory manager. Users have access to this interface through the Moodle platform. The user chooses the experiment and then it's possible to change the values of a possible configuration of the experiment, for example, when connected to a diodes board, the user can change the diode (silicon diode, germanium diode, red LED, green LED, blue LED, yellow LED) and its direction (direct or reverse) [24]. According to [23], MostaLab provides the test of electronic experiments such as the study of fundamental theorem, filters (RC, CR, RLC, etc.), operational amplifier and diode characterization. Figure 2.12 shows MostaLab's graphical interface.

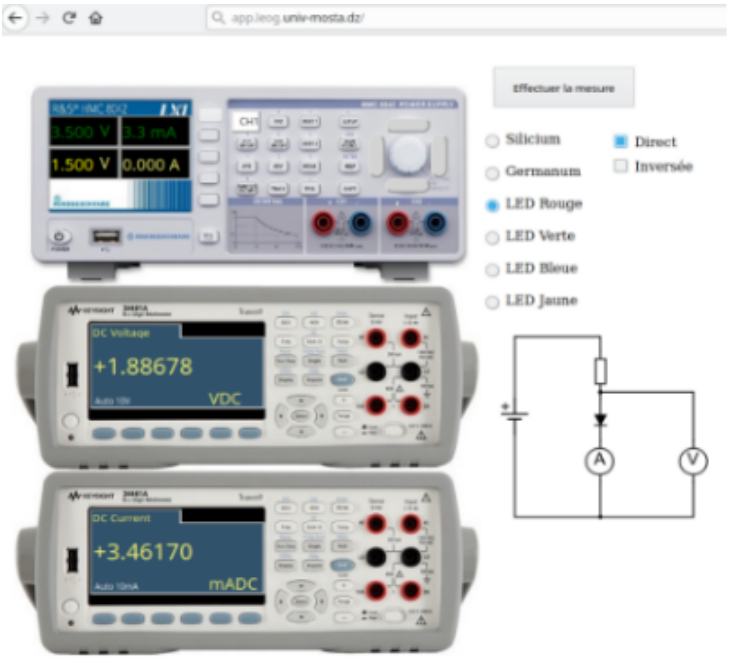


Figure 2.13: MostaLab graphical interface [24]

Students have a maximum connection time to the lab of two hours. When they make a measurement request they have exclusive access to the necessary lab equipment (if this is not in use) and the hardware is released when the measurement value is returned [24].

Figure 2.12 also shows the hardware present in the remote laboratory. That hardware is composed of a lab server hosted in a Raspberry Pi as mentioned before, a switching board, a test circuit and some measurement equipment. The switching board is used to interconnect the measuring instruments and the components of the circuit considering the experience selected by the user via the web interface. In other words, the switching board selects the desired hardware for each experiment, considering the information received by the lab server.

To achieve a multi-configuration experience, there must be electronics and switches that select a certain configuration. In this system, the switches are on the switching board and the components are on another board. Figure 2.14 shows MostaLab's hardware implementation. Note that the switching board is directly connected to Raspberry Pi and connected to an LED board [23].

Figure 2.14: MostaLab hardware implementation

- **E@Slab** - developed by *Cadi Ayyad* University (Morocco). This remote laboratory allows electronics students to carry out remote experiments that are chosen and scheduled by teachers. It is also possible to ask theoretical and practical questions through the interface to assess students' knowledge. In addition, this remote lab also uses artificial intelligence to automatically evaluate and separate students into groups based on their performance. By using this technology, it is possible to monitor students' study behaviour throughout the online practical experience [25].

  There are two types of users handling the user interface: administrators/teachers and students. The teacher is allowed to manage the lab and can divide the students into groups and schedule appointments for them to do practical work. These appointments are reservations that the teacher makes for each student for a specific experiment. The teacher can also define a practical scenario and ask the students questions. On the students' part, it allows them to check if they have any appointments and, when the time comes, the student performs the experiment [25].

  There are two architectural versions of E@Slab, each with its pros and cons. The first version consists of an embedded system using node js technology and Ubuntu as the operating system and the hardware is either a PCduino or a Raspberry Pi. The second version is based on LabVIEW and the National Instruments (NI) ELVIS II board from NI which will be shown later.

  Figure 2.15 shows the student interface of E@Slab. The student can drag and drop devices and wire them and is also able to adjust sources and measuring devices. The available experiments are related to analogue electronics such as circuits with operational amplifiers and RLC circuits [26].

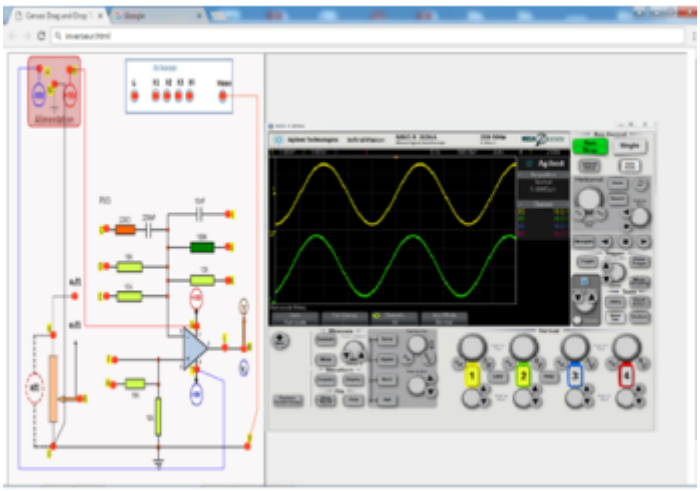


Figure 2.15: E@Slab user interface [25]

The architecture of E@Slab's first version is represented in Figure 2.16. There are two servers in this architecture. On one server are the university information system and the management platform, where students find an online course and appointments for practical work. The second server contains the user interface. It is this server that controls a relay board to select the components that will be used in the experiment and also controls measuring instruments [26].

Figure 2.16: E@Slab architecture version 1 [26]

The second version of E@Slab's architecture is represented in Figure 2.17. This version is based on the NI ELVIS II board shown in Figure 2.18. This board

was designed to be used by electronic engineering students. This board integrates the most used instruments in an electronics laboratory, such as a digital multimeter, a variable power supply, a function generator, an oscilloscope, and a Bode analyzer. This board is connected directly to the computer and all its modules are controlled by LabVIEW software. LabVIEW has an integrated software called NI ELVISmx that is responsible for controlling the NI ELVIS II board.

For the second version of E@Slab, the user needs to install the NI LabVIEW run-time engine that can be found on the NI website. In the first version, no installation is required and the user can use any search engine [26].

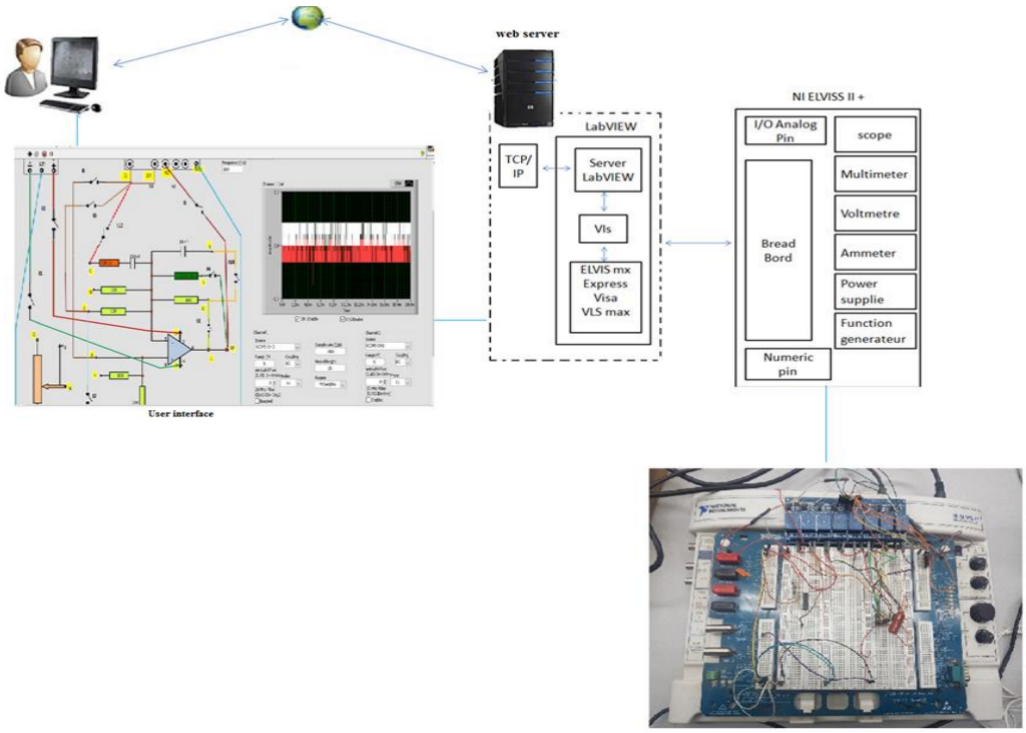


Figure 2.17: E@Slab architecture version 2 [26]

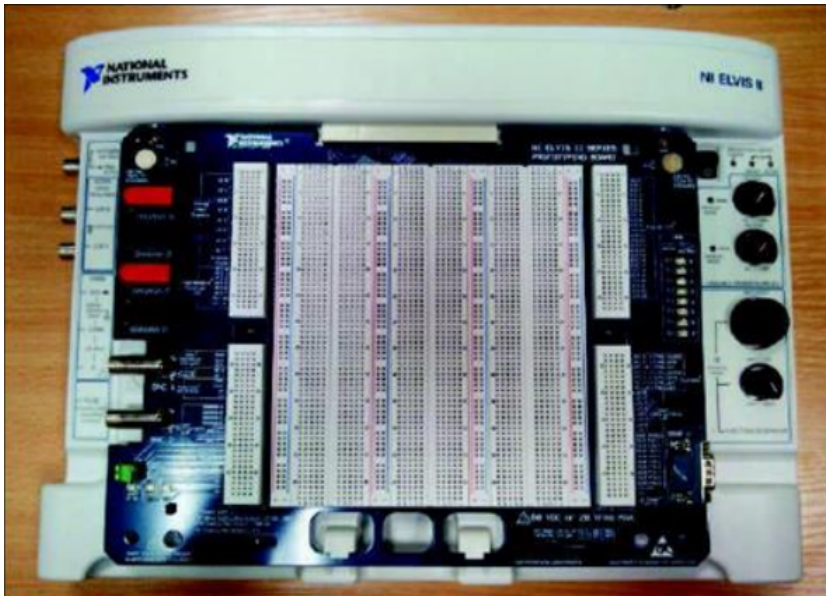


Figure 2.18: NI ELVIS II [26]

- **eLab** - developed at Bordeaux I University (France). This laboratory is a measurement platform that enables more than 130 experiments to be carried out in the field of electronics and microelectronics. This platform is dedicated to characterising electronic components and analogue signal processing circuits. The eLab provides access to several electronic and microelectronic products that meet the needs of undergraduate and graduate engineers [27]. It is possible to carry out experiments containing amplifiers, transistors, etc.

  eLab does not require any commercial software as it is a web-based solution and therefore only needs a web browser. The main objective of developing this platform is availability since it is possible to hold several sessions on the same experience without booking. One person uses the measuring devices at a time for a few seconds and there is a queue for anyone trying to access the devices simultaneously. The waiting time is made available to the user through a real-time view of the progress measurements and provides the list of pending assignments. The measurements in progress are viewed through a camera that zooms in on the active instruments.

  Figure 2.19 represents the eLab architecture. The eLab laboratory runs on two low-end computers with Linux, Apache, MySQL and Hypertext Preprocessor (PHP) setup. The front-end computer runs the web server where the eLab application is located and also has allocated the database where the users' accounts, notebooks and results are stored. Notebooks are files that contain the experimental data of the user. The instrument server is allocated in a local network, it drives the instruments and the two switching matrices.

The matrix used in the time and frequency domain connects a circuit under test to an Alternating Current (AC) Gain-Phase analyzer for frequency domain measurements or to a function generator coupled to a digital oscilloscope for time domain measurements. The circuit under test is also connected to a power supply. The switching matrix for Direct Current (DC) experiments connects 25 building blocks including specific integrated circuits, DC source meters and a power supply [27].

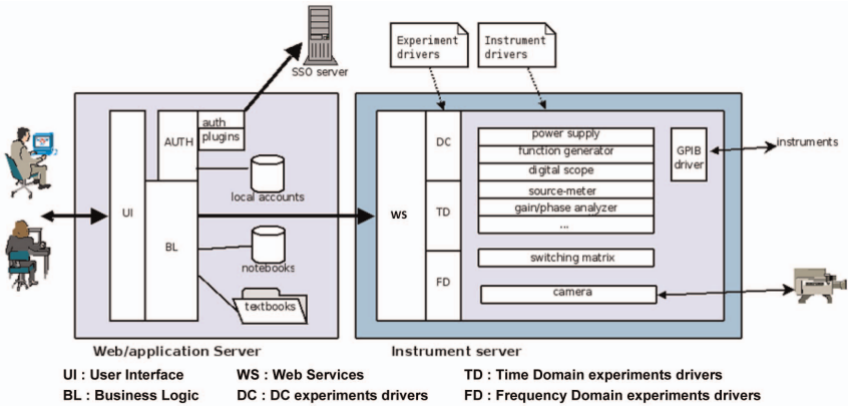


Figure 2.19: eLab's architecture [27]

Figure 2.20 shows the response of a differential amplifier frequency characterization as an example of an eLab measurement result.

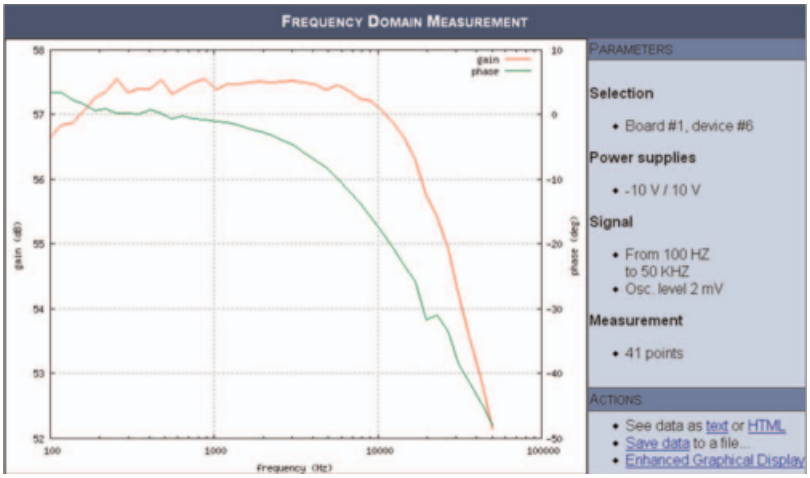


Figure 2.20: eLab's measurement results example [27]

- **eLab3D** - developed at *Universidad Politécnica de Madrid* (UPM) in Spain. The eLab3D remote lab allows electronics students to conduct a real experiment through a web interface that is based on a 3D virtual world. So, the user interacts with a virtual workbench, similar to the traditional one, and the fact

that is in a 3D world brings a sense of realism to the user. Every user's action with cables, components and instruments in the virtual laboratory is updated automatically.

To use this platform, the user must install a free 3D viewer application. The interaction between the objects and other users of this laboratory is carried out using an avatar. The virtual laboratory simulates a building with several rooms in which teachers and students can carry out various activities such as meeting rooms for collaborative learning, video rooms for presentations or discussion forums and laboratory rooms for experiments. Figure 2.21 shows the various workbenches that allow students to carry out practical experiments and interact with each other and with teachers [28].

Figure 2.21: eLab3D's Laboratory room [28]

Figure 2.22 shows a virtual workbench featuring the same components, cables, circuit boards and instruments that are usually found in a hands-on laboratory. This laboratory provides boards with pre-defined experiments and the user has to carry out the assembly of the components and has to connect the cables between the board, power supplies and measurement instruments. The experiment boards allow you to test circuits involving operational amplifiers, diodes, transistors and passive components [28][29].

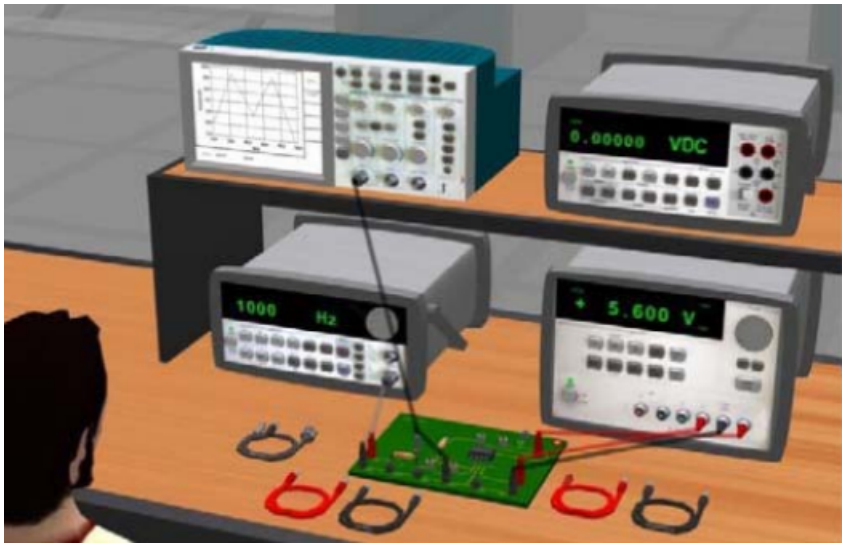


Figure 2.22: eLab3D's workbench [29]

Figure 2.23 shows an example of a test board and it's possible to visualize a drawer (at right) with some components (resistors) for users to assemble. It is also possible to see the cables connecting the board with the equipment [28][29].

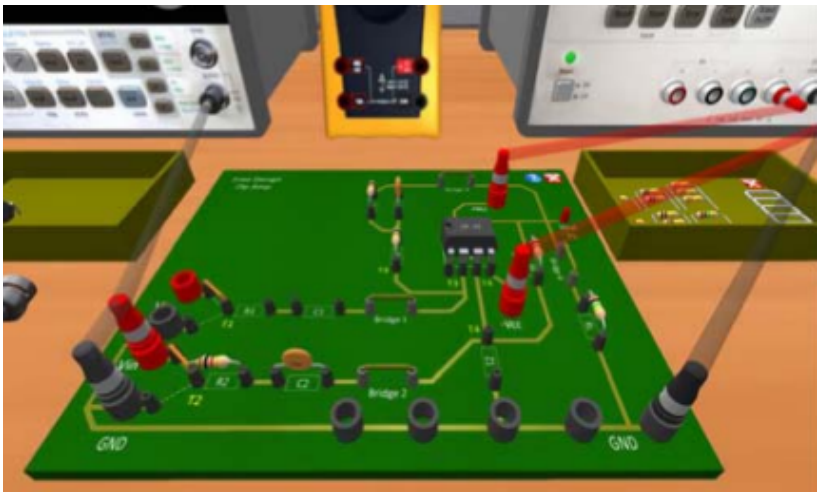


Figure 2.23: eLab3D's test board example [29]

Figure 2.24 shows the architecture of the eLab3D platform, and this architecture is composed of hardware and software elements. The software elements are composed of blocks that refer to the instruments and the experiment. The instruments include the equipment normally used in an electronics laboratory, such as a power supply, a signal generator, an oscilloscope and a multimeter. These are controlled through the laboratory server using the General Purpose Interface Bus (GPIB) and USB communication protocols. The experiment

block is made up of a system that includes a motherboard that communicates via USB to the server and is connected to the various experiments available [29].

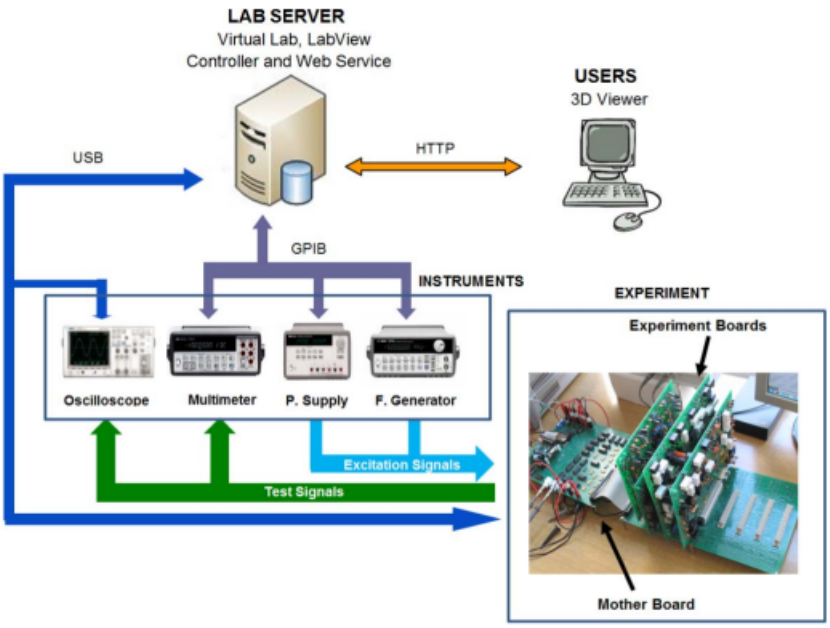


Figure 2.24: eLab3D's architecture [29]

For a student to use the platform, he needs to make a reservation for the workbench, which is always made for two hours. After reserving a bench, it is necessary to choose the test board that he wants to experiment. This platform allows several users to work collaboratively on the same workbench and all the actions performed on a workbench are recorded in a database for later evaluation by the teacher. If a student makes connections that make the circuit work incorrectly, the platform informs the student that they have made a mistake to protect the equipment [28][29].

2.2 VISIR Literature Review

This section will discuss the VISIR remote laboratory. Considering the proposed project, it is necessary to understand the architecture, software, hardware and use of the VISIR web interface. All these topics are presented in detail in this section.

2.2.1 Introduction

In engineering, it is necessary to know the principles of nature. The use of laboratories serves to reproduce that nature and provide the student to recognise and

understand the phenomena observed. Remote laboratories eliminate some of the problems found in traditional laboratories, such as high maintenance costs, low availability, and problems of organising students, equipment and teachers, among others. VISIR is a remote laboratory for designing, wiring and measurement of electronic circuits developed for undergraduate students by The Signal Processing Department (ASB) at Blekinge Institute of Technology (BTH) in Sweden. The VISIR project was started in 1999 and was launched at the end of 2006 under the leadership of Ingvar Gustavsson [30][31]

The VISIR laboratory aims to create a web-based replica of a traditional workbench containing an oscilloscope, a function generator, a multimeter, a power supply, a breadboard and diverse components. The VISIR platform enables users to set up and carry out experiments remotely, and the experiments are executed in real equipment physically distant and the results of these experiments are verified on real instruments and displayed on virtual instruments. On the user interface, a simulation graphically similar to the real equipment and instruments is visible making it easier to familiarise the student with the real equipment [32].

The wiring mechanism is performed by a relay switching matrix that is connected to an instrumentation platform called PCI eXtensions for Instrumentation (PXI). The hardware used in this architecture is controlled by a LabVIEW-based software server and there is also a measurement server software that protects the equipment from dangerous connections since it verifies the circuit connections made by the students before making the same connections in the physical equipment [32].

Figure 2.25 illustrates the overall architecture of the VISIR remote laboratory. The VISIR platform consists of four main parts: the Equipment Server, Measurement Server, Web Server, and Web Interface. The equipment server consists of all the lab equipment, including the PXI platform and the relay switching matrix, which are connected to and controlled by a LabVIEW server. It receives the commands from the measurement server to be executed on the actual instruments. These commands follow the form of the protocol based on Extensible Markup Language (XML) over Hypertext Transfer Protocol (HTTP). The component list is a file located on the equipment server and consists of the components that are installed in the switching matrix. The Measurement Server is a Visual C++ based server. This server acts as a virtual instructor that verifies and controls the commands sent from the web interface to the equipment server. This action is performed to avoid dangerous design circuits to protect the physical equipment. It is programmed by a file called a max list that contains the maximum values that the components can handle and the instrument settings for each experiment. The Web Server hosts the VISIR web interface, which is written for the Apache server against the MySQL database. The Web Interface is the website of VISIR, which is accessed by students through a secured protocol named Hypertext Transfer Protocol Secure (HTTPS) after getting

authenticated. The students design the circuits using the virtual breadboard and the circuit data is sent from the interface to the measurement server to be verified and then to the equipment server to be converted into a real circuit and the required measurements are taken. The measurement results are returned to the virtual instruments in the user interface [33].

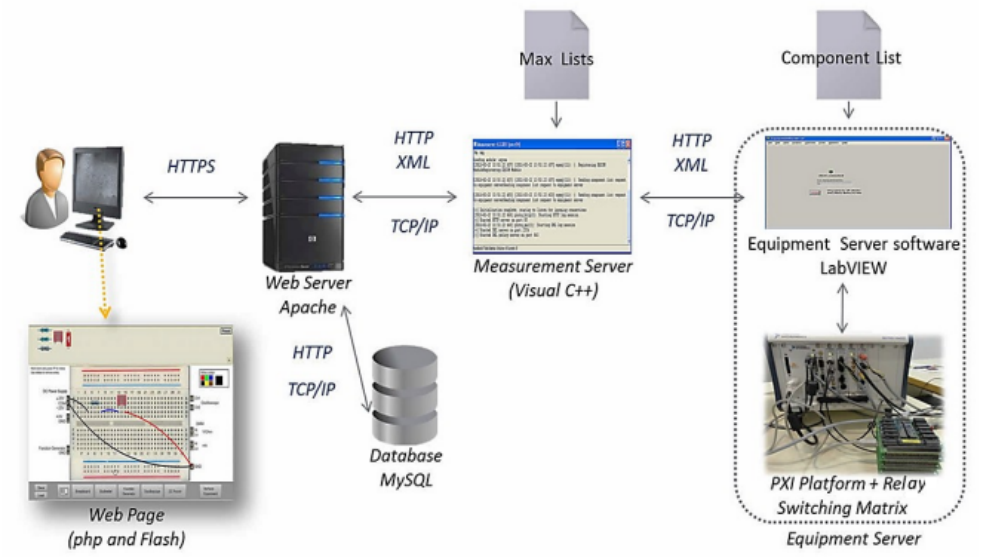


Figure 2.25: Architecture of VISIR [33]

VISIR has its own Learning Management System (LMS), called OpenLabs, which contains various access and administration features such as registration, login, reservations, and account type, among others. Access to the lab content depends on the permissions of the user's account type [33][34]

This remote laboratory is prepared to handle PXI, however, LAN eXtensions for Instrumentation (LXI) can also be used, as they are the most suitable instrumentation platforms for remote control, according to [14]. Due to this ease of interchangeability between instrumentation platforms, all the necessary drivers according to the Interchangeable Virtual Instruments (IVI) standard must be installed. The IVI standard defines open driver architecture, a set of instrument classes and shared software components [14].

2.2.2 Hardware

The hardware consists of a computer, a switching matrix and an instrumentation platform (NI PXI). The computer controls the PXI to make measurements of a circuit located in the switching matrix. The components of this circuit are chosen by closing a set of relays after sending the instruction through the computer via a USB cable.

2.2.2.1 PXI Platform

A PXI system is a PC-based platform for measurement and automation systems. The PXI platform contains instrument module cards, a controller card and a chassis in which all the cards are suitable. For each element of this platform, several models are depending on their technical characteristics [35].

The NI PXI-Chassis contains the instrument modules, the controllers and the communication system between the different parts. NI PXI-Modules or NI PXI-Instruments are the module cards that replace instruments. The modules are connected to the NI PXI-Chassis. These module cards are removable and can be added or removed as required. The NI PXI-Controller is a built-in computer that is connected to the NI PXI-Chassis and serves to manage the existing modules. It has features such as an integrated Central Processing Unit (CPU), hard disk, Ethernet, keyboard/mouse, etc. The fact that these device drivers are already installed eliminates the need for an external computer. [4][35].

Figure 2.26 shows the current PXI-Chassis in use by VISIR at ISEP.

Figure 2.26: NI PXI-1033 [36]

NI PXI-5114 - Oscilloscope

The PXI-5114, Figure 2.27, it is an 8-bit resolution dual-channel oscilloscope featuring a maximum bandwidth of 125 MHz, a maximum sampling rate of 250 MS/s and flexible settings for coupling, impedance, voltage range, and filtering. PXI oscilloscopes also feature multiple triggering modes, deep onboard memory and an instrument driver that includes data streaming and analysis functions [37].

Figure 2.27: NI PXI-5114 [37]

NI PXI-5402 - Function generator

The PXI-5402 function generator, Figure 2.30, features one channel, 14-bit resolution and 20 MHz bandwidth to generate sine, square, triangle and ramp waves as well as other signals such as DC signals. This function generator has an output voltage range of -5V to 5V [38].

Figure 2.28: NI PXI-5402 [38]

NI PXI-4072 - Digital multimeter

The PXI-4072, Figure 2.29, is a digital multimeter with 6½ digit resolution and a voltage range of ± 300 V AC/DC. It features a sampling rate of 1.8 MS/s. It

provides AC/DC voltage and current, 2 or 4-wire resistance, frequency, period, coil and capacitor measurements as well as diode tests [39].

Figure 2.29: NI PXI-4072 [39]

NI PXI-4110 - DC Power supply

The PXI-4110, Figure 2.30, is a programmable DC power supply with three channels (+6V, -20V and +20V) capable of supplying up to 1 A per channel [40].

Figure 2.30: NI PXI-4110 [40]

2.2.2.2 Switching Matrix

The switching matrix was manufactured at BTH and is based on a stack of printed circuit boards designed according to PC/104 standard specifications. The switching

matrix is remotely controlled by the user interface and consists of relays, sockets for components, and instrument connectors [35]. It is sized to meet the needs of most circuits used in electronic experiments in undergraduate education. The teacher or a staff member adds or removes components from the sockets according to the circuits required [2].

Figure 2.31 shows the types of boards that can be inserted in the switching matrix. The bottom three boards (Source board, Digital multimeter board and Oscilloscope board) correspond to the instrument module inserted into the NI PXI Chassis. At the top is the Component board with the relays and the sockets for inserting components [4][35].

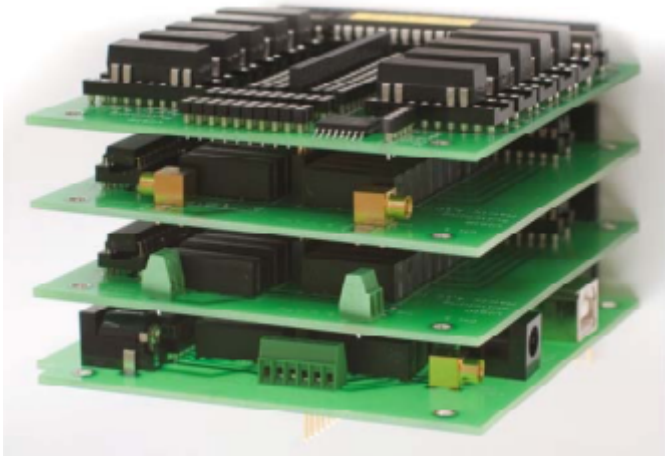


Figure 2.31: VISIR Switching Matrix [31]

More recently, a board called Dual Component Board was developed, Figure 2.32. This board connects up to two two-lead components or a low-frequency instrument such as a multimeter with two of the nodes A, D, F, H and 0 [41].

Figure 2.32: Dual Component Board [41]

General outline

The switching matrix was developed with the main purpose of allowing students to build electronic circuits remotely. To do this, switching techniques are used through electromechanical relays that control pre-established connections that enable components to be interconnected to each other as well as to measure equipment and power supplies [4][42]. The switching system of VISIR has limitations as does not provide the switching techniques of a switching matrix architecture. It does not provide the ability to connect any component directly to any other component [43].

Usually, the boards used in the switching matrix can be divided into two large groups: the instrumentation boards, which refer to the measuring instruments and power supplies, and the component boards, which refer to the hardware needed for circuit design, consisting of the components required for a certain experiment [34]. Therefore, the typical configuration of the VISIR switching matrix contains one Source board, two Instrument boards and one or more Component boards [44].

The nodes of the switching matrix are interconnected between the boards creating a node bus, each board has two independent buses which allow them to be stacked and share the same connections. The term node refers to the fact that each conductor added can be a node in a specific circuit, such as resistors, diodes, and transistors, among others. Additionally, short circuits can be added with jumper leads (copper connectors) which in theory consist of a 0 Ω resistor that shorts two nodes. The 17 nodes are divided into two groups [41]:

- The ten nodes from A to I and 0 are called visible nodes because they can be nodes of a circuit that serve to interconnect the components and short circuits.

Node 0 function is if there is an oscilloscope and/or function generator connected, node 0 will be connected with protective earth since these instruments are always connected with protective earth;

- The seven nodes from X1 to X6 and COM are called concealed nodes since the instruments cannot connect directly with them, only sources can since they serve to connect to the Sources board. However, a concealed node may be connected with any other node by a relay switch.

Note that the output of the function generator is hardwired to node A, while the oscilloscope and DMM channels are dynamically connected to any node depending on the circuit design. [32].

All switching matrix boards have a microcontroller (PIC16F767) which is also called Board Controller and is used to identify each board and control the relays on each board. Each PIC16F767 microcontroller uses the Inter-Integrated Circuit (I^2C) serial communication protocol bus to connect to peripherals. The seven-bit I^2C address is written into each board's firmware. Figure 2.33, presents in the left column the name of each board, in the middle one indicates the corresponding board number and in the last one, the I^2C address of each board is presented [32].

Board Type	Board Label	I^2C Address
Component board 1	1	COMP 1
Component board 2	2	COMP 2
Etc.		
Oscilloscope board	16	OSC 16
DMM board	17	DMM 17
Etc.		
Source board	24	SRC 24

Figure 2.33: Corresponding Board Number of Each I^2C Label [35]

All boards contain integrated circuits, called ULN2003A, which are arrays of seven NPN Darlington transistors. These transistors are intended to amplify the current and voltage that comes from the microcontroller and power the electromechanical relays. They supply 500 mA and 50 V at the output.

Electromechanical relays can be Single Pole Single Throw (SPST) or Double Pole Single Throw (DPST) depending on whether they are connected to sockets containing components from two or more leads. These relays can handle currents up to 2 amps and have a minimum life expectancy of 3 x 10^8 operations [32].

Figure 2.34 shows an example of all the possibilities of a circuit in a VISIR system implemented at BTH University. The concept applied in this example is transversal to all VISIR systems. Therefore, in Figure 2.34, several nodes are represented and the possibilities of components can be found between these nodes. Currently, one of the limitations of VISIR is that it is only possible to connect one component

between two nodes, instead of connecting several in parallel. The function generator and the DC power supply are always connected to node A and only one of them can be connected at a time.

Between node A and node B, one of these components can be connected: R1, R2, R3, C3, a shortcut to node B, a shortcut to node C, a shortcut to node D and a shortcut to ground. This concept applies to the remaining nodes. For example, if it's intended to connect a power supply to node B, the shortcut that connects these two nodes is linked. And if it's wanted a low-pass filter (a resistor in series with a capacitor) with the characteristics of components R3 and C2, the relay that connects the DC source to node A is closed along with the relay that connects resistor R3 between nodes A and B, the relay that connects node B to C via a shortcut and the relay that connects C2 between node C and ground.

In the case of components with more than two leads, as is the case of a transistor and an operational amplifier represented there, these have to be connected to the nodes shown there, in case of a change of transistor it has to be positioned in the same nodes, because it may not be possible to have combinations of components to connect it elsewhere.

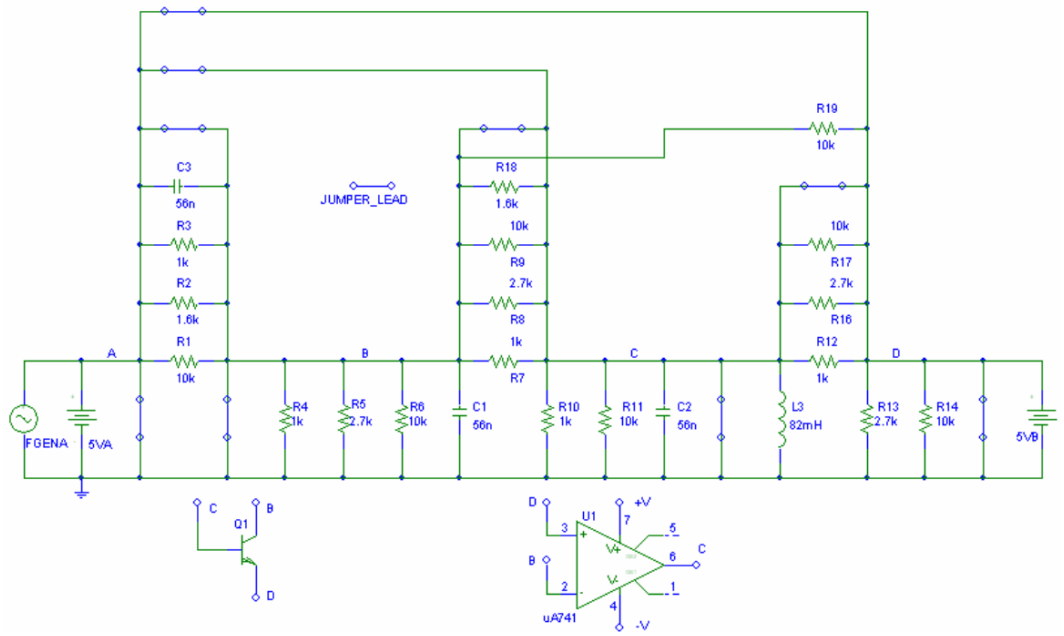


Figure 2.34: Example of circuit possibilities [3]

Component Boards

The switching matrix can have a maximum of 15 independent Component boards. Figure 2.35 shows a Component board with a resistor connected through an electromechanical relay to nodes A and B.

This board provides a double 20-pin Integrated Circuit (IC) socket, which can be seen at the top of the board, Figure 2.35. These sockets are reserved for IC

and other components with more than two leads. This type of component needs to connect to more than one relay since each pin of the device in question is connected to a node and in this way, it is possible to connect components with several leads.

This board also provides a digital potentiometer, AD7376, which can be used in experiments where wiper resistance is not problematic [41].

Note that the node name is written on the Component Board next to the socket pins. The socket pins are electrically linked horizontally, i.e. the whole first line refers to node A.

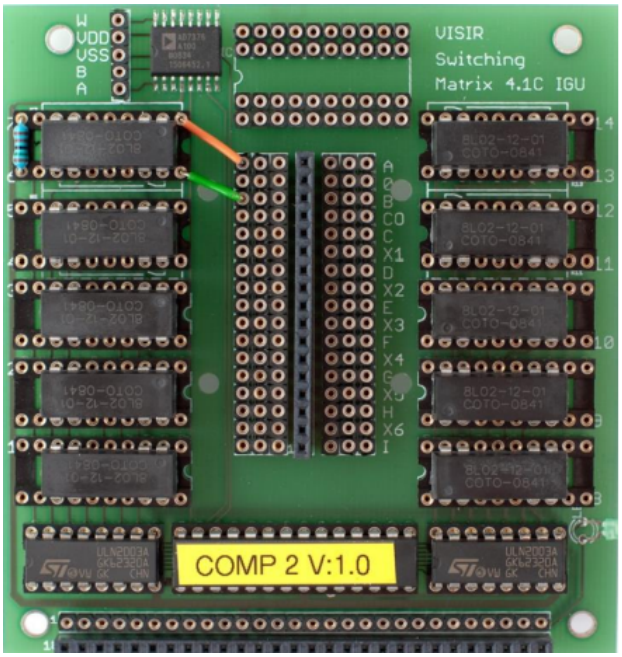


Figure 2.35: Component board [32]

In addition, it is also possible to add an external circuit to the switching matrix to increase the available experiments, as shown in Figure 2.36.

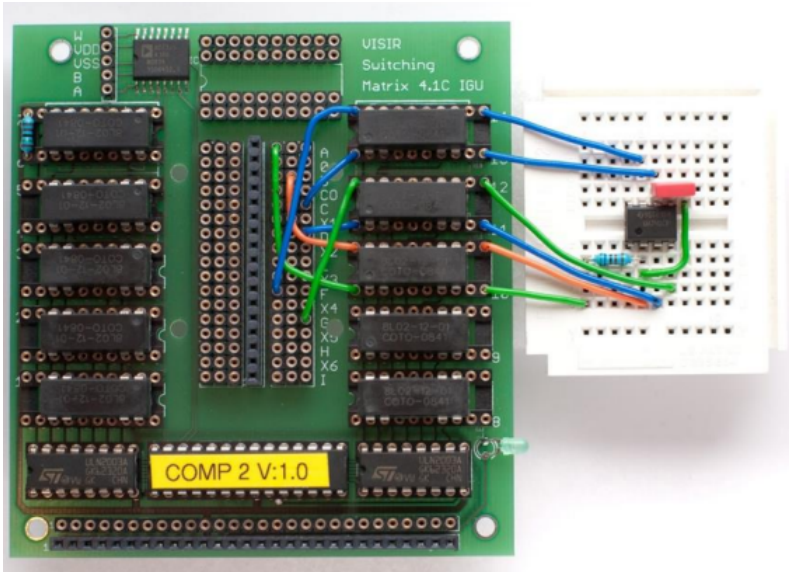


Figure 2.36: External circuit connected to the matrix [41]

Instrument Boards

Each switching matrix has two Instrument boards, an Oscilloscope board, Figure 2.37, and a DMM board, Figure 2.38. The basic function of these boards is to connect to the instrument modules present in the NI PXI Chassis. The DMM board is designed to connect low-frequency equipment such as a digital multimeter and the Oscilloscope board is designed to connect high-frequency equipment such as a dual-channel digital oscilloscope. Figure 2.37 shows the places where the two channels of the oscilloscope have to be connected [41].

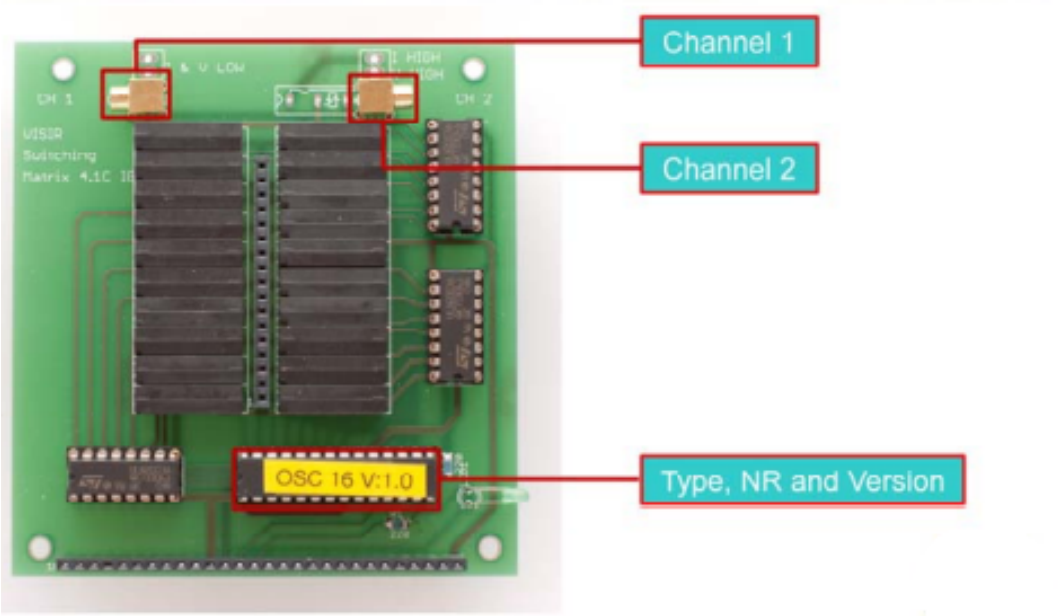


Figure 2.37: Oscilloscope board [31]

The DMM board features two inputs, labelled as Low Side and High Side. One of these inputs is for voltage and resistance measurement and the other is for current measurements. Note that when measuring current it is necessary to place the device in series with the component (like any direct current measurement with an ammeter), this is only possible by adding a short-circuit [41].

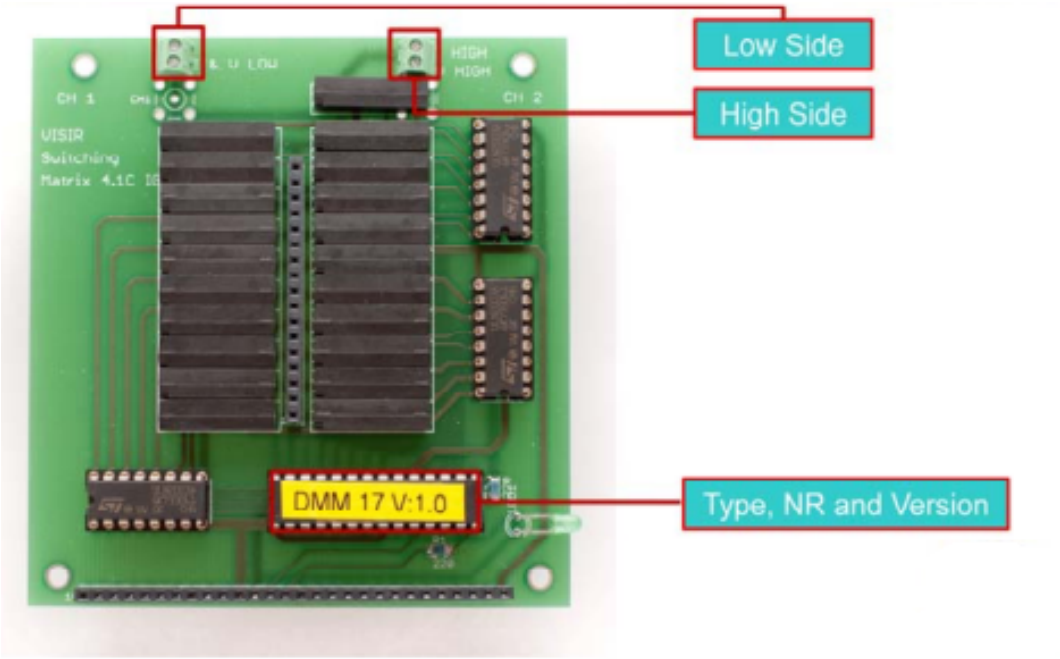


Figure 2.38: DMM board [31]

Source Board

Each switch matrix owns one source board and it is represented in Figure 2.39. This board usually has card number 24 and allows the connection of several power supplies as a function generator or a DC power supply [31]. Figure 2.39 shows a connector at the top right for an external power supply used to supply the relays. It's also visible on the source board the relays that connect to the power supply and function generator, two devices that are placed in NI PXI Chassis.

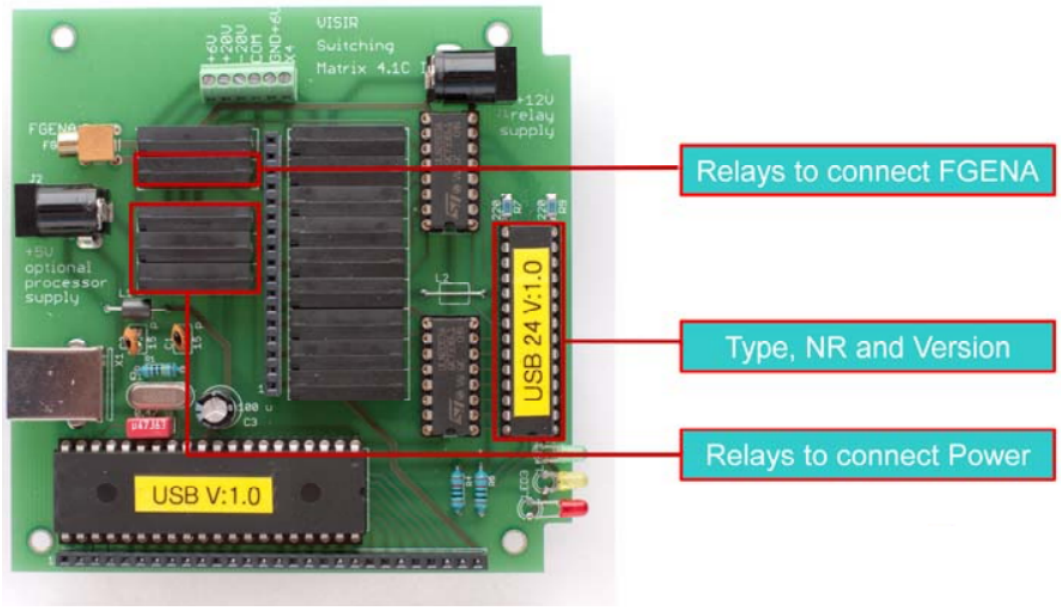


Figure 2.39: Source board [31]

On the Source Board, there is a microcontroller named, PIC18F4550, which communicates with the PIC16F767 microcontrollers on each board through the I^2C protocol to send commands to the relays on each board. These commands cause the relays to open or close according to the designed circuit [32]. In the left part of the board, it's possible to see a USB connector used to communicate with the computer.

2.2.3 Software

The software used in VISIR can be divided into two main topics: the software required to provide a remote electronics laboratory capable of being controlled remotely, and, the configuration and description files required for the experiments and components installed [34] . Therefore, in this subsection, the topics on which the VISIR system is based are described, starting with a general architecture of the system and specialising in the different types of existing servers and configuration files.

2.2.3.1 Architecture

The software architecture is divided into several virtual servers each with specific functions as represented in Figure 2.40. As mentioned before, the software architecture is divided into four parts [14]

- Web interface responsible for user admission, resource scheduling and other administration issues;
- An Equipment Server containing the instrumentation used in the experiments and the relay switching matrix. The software of this server is written in LabView;
- A Measurement Server that handles requests from clients and, through a virtual instructor, checks the requested circuits before sending them to the Equipment Server, so as not to damage the equipment;
- An Experiment client software presented through an HyperText Markup Language (HTML) page that contains the module as an embedded object.

The VISIR workflow starts at the web interface by allowing the student to design the circuit through a web browser while the measurement and equipment server are responsible for translating the virtual circuit into a real one on the switching matrix and providing the user with the measurements obtained from the experiment [45].

All these components mentioned are going to be detailed in the following subsections.

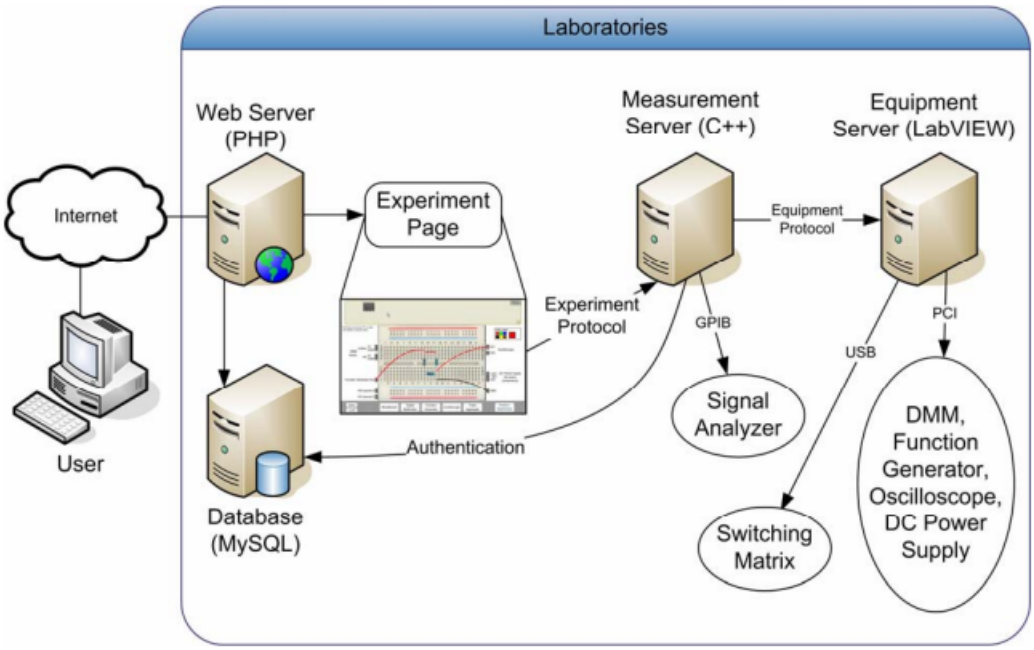


Figure 2.40: System modules and interfaces [46]

Communication

Considering the proposed project, it is important to study the USB communication that occurs between the computer and the switching matrix for future developments.

The switching matrix is controlled via a computer that sends USB messages in USB raw format. The computer sends a message to the matrix controller whenever the value of one or more relays needs to be changed and the matrix controller distributes the message to the board controller concerned. These messages consist of four bytes, the first byte consists of the board number, i.e. its I^2C address, and, the remaining bytes contain the relay numbers or the potentiometer ratio, Figure 2.41. For the bytes related to the relays, if bit 0 is set, the first relay corresponding to this byte must be energized, i.e., for byte 1 it's relay 1, for byte 2 it's relay 8 and for byte 3 it's relay 15 (only in instrument boards) and if bit 0 it's cleared the corresponding relay has to be released. It's the same for the rest of the bits except for bit 7 which has to be 0. For byte 3 corresponding to the component board, the potentiometer ratio is set in bits 0 to 6 and bit 7 has to be 1. For instance, if the string sent to the matrix is 024001000000 it means that the board where the relays will be closed is board number 24 (source board) and relay number 1 of that board is going to be closed. Every byte corresponds to 3 digits of the string once a byte can go from 0 to 255 in decimal format (3 digits).

Board type	Byte 0	Byte 1	Byte 2	Byte 3
Source	Board number	Relay 1 – 7	Relay 8 - 14	Not used
Instrument	Board number	Relay 1 – 7	Relay 8 - 14	Relay 15 - 21
Component	Board number	Relay 1 – 7	Relay 8 - 14	Pot. ratio

Figure 2.41: Board Controller messages [41]

Figure 2.42 shows an example of a USB message and the corresponding binary number of byte 3. This message is targeted to board number 16 (oscilloscope board) and the decimal number corresponds to closing the fifth relay of byte 3 which is relay number 19.

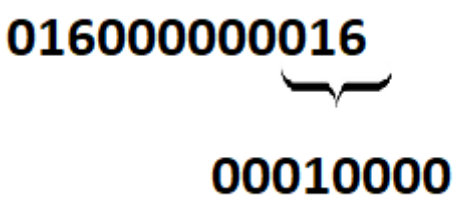


Figure 2.42: Example of a message

2.2.3.2 User Interface

The user interface is a web-based interface, developed in HTML, and it is used to design circuits, with the ability to drag components to a virtual breadboard and wiring components, instrumentation and power supplies, and also allows the performing of measurements and visualising the responses in virtual instrumentation modules. The user can change the instrumentation settings as in real life.

The VISIR platform has the feature of handling multiple users. Each user has to reserve the lab every time they use it and then it redirects to the experimentation page, Figure 2.43. Note that the time students spend designing the circuit is not using the actual equipment. The experimentation page has three areas, the components area has the components needed for every experiment, the implementation area where circuits are designed by placing the components and connecting the instrumentation, and the instrumentation area where the user can set up the instruments [35][47].

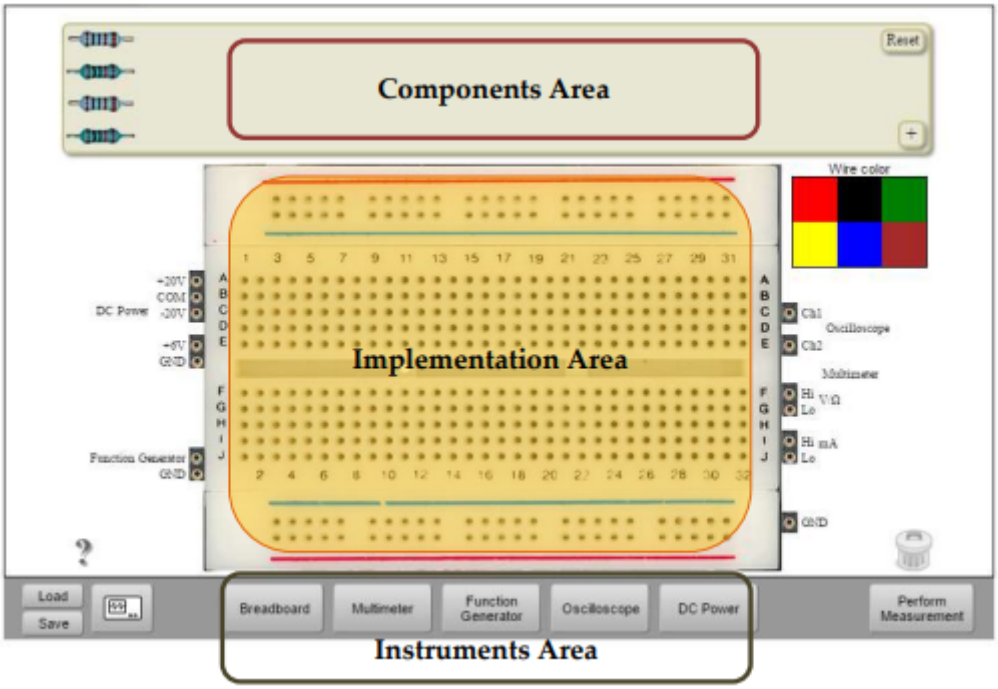


Figure 2.43: VISIR User Interface [47]

There is a + button on the bottom right of the component area, when clicked it shows the list of the available components. These components have their picture as real-life components. After the components selection, they are placed in the implementation area for the user to interconnect them. Note that the virtual breadboard holes have the same interconnections as the real ones. When the components are

placed on the breadboard, there is the option to delete them or to rotate them. The instrumentation connections are placed on the breadboard on both sides.

Figure 2.44 shows an example of a circuit using the function generator and the oscilloscope. The function generator has to be connected to the diode's anode and one channel of the oscilloscope is connected to the circuit's input and the other one is connected to the circuit's output. Note that the function generator ground doesn't need to be connected to the ground, once it's the same as the Ground (GND) terminal on the right side [47].

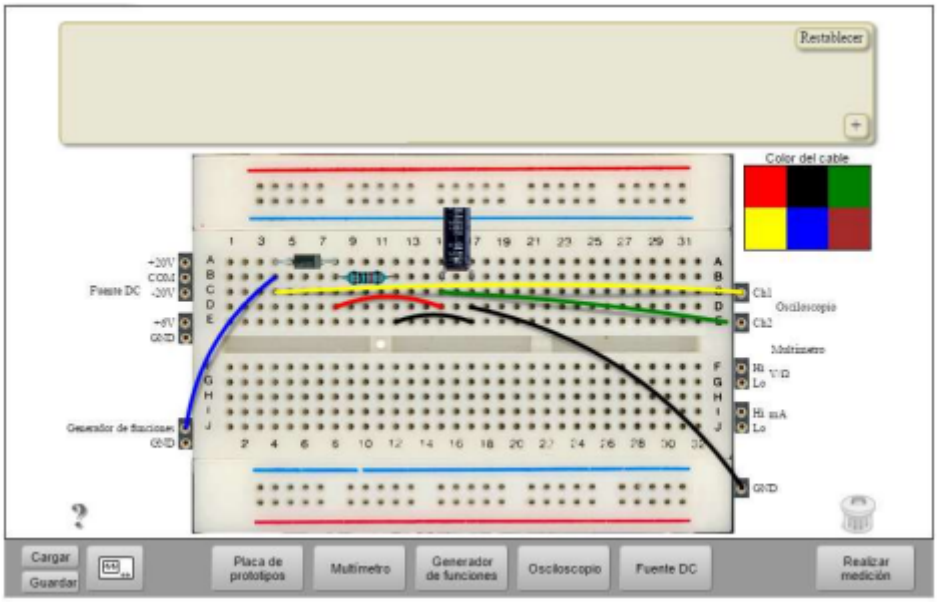


Figure 2.44: Circuit example with the connections [47]

The equipment modules are developed in Adobe Flash . The virtual instruments are similar to those existing in traditional workbenches to better familiarise students with the equipment. The existing modules are a Digital Multimeter (Fluke 23), Function Generator (HP 33120A), Oscilloscope (Agilent 54622A) and DC Power Supply (E3631A) [35].

Figure 2.45 shows the digital multimeter front panel which is selected by the button "Multimeter" at the bottom of the experimentation page. Users can use the digital multimeter to measure resistances, voltages and currents. On this page (Figure 2.45), they can change multimeter settings. For the use of other instrumentation devices users have to do the same procedures, the only difference is the type of settings of each one. To obtain the results of the experiment and for these to be shown in the instrumentation, the user must press the "Perform Measurement" button found on the experimentation page. The physical equipment is only reserved for a user at the moment he presses the "Perform Measurement" button and for the time needed to perform the experiment and obtain the results. [47]

Figure 2.45: Digital multimeter front panel [47]

2.2.3.3 Web Server

The Web Server, also called Web Interface, is the web page that provides the user with access to the experiment's client. It is responsible for the user interface and for receiving the configuration data made by the user. After running the experiment, it displays the data on the screen [30][35]. The Web Server is written in PHP and is connected to a MySQL relational database management system and installed on an Apache HTTP web server. This handles the client login and authentication procedures over the secure HTTPS protocol [32][35].

The Web Server provides features similar to those of a LMS to facilitate the implementation of VISIR in the learning process. These features are associated with the type of user account, which can be divided into three main groups: administrator account, teacher account and student account.

The administrator account refers to the laboratory supplier and has the following privileges: upload files, videos, manuals, etc.; can create, update and delete courses with the association of a start and end date, a maximum number of users and teacher and student accounts; the administrator can change the "Teacher view" so that he has all the privileges of the teacher in that course.

The administrator is the one who creates the teacher accounts and associates each one to a certain course that has the following privileges: add and remove experiences; make a scheduled reservation; the teacher can change the settings to "Student View" to check the same platform content as their students.

Student accounts are created by the teacher and are associated with a course. Students only have permission for the experiences created by their teacher. The student can make a reservation separately or reserve a seat on the scheduled reservation made by the teacher [32].

2.2.3.4 Measurement Server

The Measurement Server was developed in C++ and is connected directly to the Web Server and Equipment Server, responsible for controlling the messages between them. This server is responsible for receiving the requests that come from the user in XML file format and translates the file to be interpreted by the next server. This translation is based on the IVI standard. This standard specifies a set of commands that each instrument (oscilloscope, function generator, etc.) must implement [30]. The XML file is sent over the Transmission Control Protocol/Internet Protocol (TCP/IP) via port 2324. The requests and responses must not exceed 64 kB in size.

At each request, the Measurement Server checks if it is a valid user by validating the client cookie generated by the Web Server with the database. It also acts as a virtual instructor, i.e. it compares the circuit information received with the MaxLists (file created by the teacher with the maximum values allowed for the components and instruments for a certain experiment) before executing the requests on real instruments, to avoid possible damage to the equipment. All MaxLists for all experiments are allocated within the Measurement Server software folder.

The Measurement Server can handle 16 customer requests simultaneously in less than 1 second by queuing them all and executing them sequentially according to priority, reservation, and so on. After validating and queuing the requests, it sends the requests in orderly TCP/IP sessions through port 5001 to the Equipment Server. The Measurement Server acts as a gateway that can provide more than one Equipment Server [32].

Figure 2.46 shows an example of an XML file of a request sent by a function generator. The experiment protocol describes the configurations and functions that each instrument can perform [32]

```
<functiongenerator>
<fg_waveform value="sine" />
<fg_amplidute value="1000.0" />
<fg_frequency value="1000.0" />
<fg_offset value="0.0" />
<fg_startphase value="0.0" />
<fg_triggermode value="continuous" />
<fg_triggersource value="immediate" />
<fg_burstcount value="0" />
<fg_dutycycle value="0.5" />
<fg_userdefinedwave length="20" encod-
ing="BASE64">ABCD1234ABCD1234ABCD
</fg_userdefinedwave>
</functiongenerator>.
```

Figure 2.46: A message based on XML to describe the parameters of a function generator [32]

MaxLists

MaxLists files contain a list that defines all possible circuits and maximum values for components and instruments, such as the maximum output voltage allowed for sources. MaxLists files always end with the extension .max [48]. This check allows material damage to be avoided and, if the check is validated, physical wiring is performed on the switching matrix.

The Switching Matrix in the Measurement Server folder has reserved words, including Resistor (R), Inductor (L), Capacitor (C), Electrolytic capacitor (CE), Diode (D), a Function generator (VFGENA and VFGENB), DC voltage sources (VDC+6V, VDC+20V, VDC-20V, and VDCCOM), Short-circuit (SHORTCUT), Probes (IPROBE, DMM, PROBE, PROBE1, PROBE2, PROBE3, and PROBE4), Generic component called the black box (BLACKBOX), Operational amplifier (OP), Transistor (Q), and Potentiometer (POT). These reserved words can be listed in MaxList files and new types can be defined as well [4].

In the code extract shown, Figure 2.47, it's possible to see that each line indicates the type of component, the component board on which it is located, the number of the relay to which the component is connected and the corresponding nodes, maximum values admitted, as is the case of the function generator (VFGENA) which supplies a maximum of 5 V. For example, *R*_3_6 C D 10 k represents a resistor (R) with a value of 10 kΩ which is connected to relay number 1 of component board number 4 and, is connected between nodes A and H. The SHORTCUTS shown here are referenced in the same way, except for the value since they represent a 0 Ω resistance.

VFGENA_FGENA1	A 0	max:5	
VDC+25V_4	B	max:15	imax:0.5
VDCCOM_24_2		0	
SHORTCUT_1_5	A B		
SHORTCUT_1_5	B E		
SHORTCUT_1_5	B C		
R_3_6	C D	10K	
R_5_2	D E	1K	
SHORTCUT_1_5	C 0		
SHORTCUT_8_9	D 0		

Figure 2.47: Example of a MaxList [34]

For components with more than two leads, more relays are required and, for example, for an operational amplifier, the description in the MaxList is as follows: $OP\_4\_9 : 4\_10 : 4\_14$ NC1 B D G NC2 C F NC3 uA741. The uA741 amplifier is connected to component board number 4. Terminals 1, 5 and 8 are not connected. Pin 2 is connected to node B and relay 9 of component board number 4, and so on [35][41].

The DMM board and Oscilloscope board connections are fixed and don't appear in the MaxList or the Component List [31].

2.2.3.5 Equipment Server

The Equipment Server consists of the NI PXI platform and the switching matrix. The server software is installed on the NI PXI controller, but it can be installed on a separate computer. This consists of a software application developed in LabVIEW to control the instrumentation equipment. As previously mentioned, the Measurement Server validates the experiment protocol requests through sessions over TCP/IP through port 5001 and, the Equipment Server software receives the requests that have been validated and executes them through the connected instruments. After the measurements have been taken, the results are returned to the client computer screen in the same sequence as they were executed. The results are displayed in the form of measurements on the virtual instruments.

VISIR supports instrumentation in the form of the PXI platform, however, some universities would like to use other platforms such as LXI or GPIB. To ensure platform interchangeability VISIR recommends using the functions and attributes defined by the IVI standard to describe the capabilities of the laboratory hardware. All instrument drivers that are installed on the Equipment Server are compatible with the IVI standard. The base capabilities of an IVI instrument are, for example, an oscilloscope containing the IviScope extension with automatic autoconfiguration capability, or the ability of an oscilloscope to calculate waveform measurements such as rise time or frequency, etc [32].

The Equipment Server hosts configuration files, named component list, with all the components available and their location, and an initialization file ("EquipmentServer.ini"). Before running the Equipment Server, it is necessary to configure the address of the instruments and the PXI slot to which each one is connected for the server to recognise the location of each instrument, as represented in the dashed rectangle in Figure 2.48. If other configurations need to be made, such as the communication port, types of components, etc., they are made in this file [35].

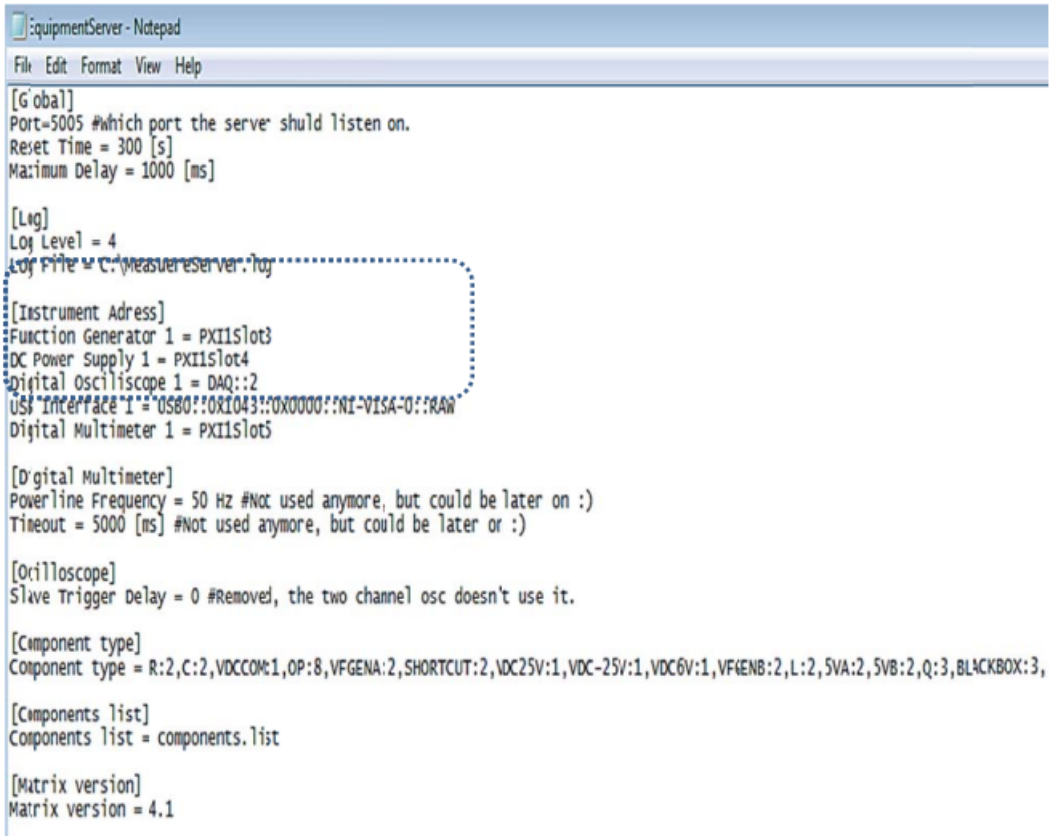


Figure 2.48: EquipmentServer.ini file [35]

Component List

The component list file describes all the components and instruments that are currently installed in the matrix to make them known to the software. There is a single component list file for each switching matrix. This file is located in the Equipment Server software folder and is called "components.list". All the components connected in the component boards of the switching matrix should be listed in this file. Thus, it's only possible to build circuits with the components and the connections described in the file.

The components with two leads are listed as <Component type>_<board label>_<relay number> <Node1><Node2><etc.> <Value>.

This file follows the same notations as a MaxList, for instance, the instruction *C*_2_4 AB 10n represents a capacitor (C) of 10n farad connected to relay 4 to nodes A and B in board number 2 [35].

Figure 2.49 shows an extract of a component list file. This file allows comments after the * character which the MaxList file does not allow and all the reserved words are the same as in MaxLists files. Dual Component Board components always end with the characters _X.

```
* Source board
        VFGENA_24_1          A 0

*Power supply terminals
*The notations refer to designations of the output terminals of Ni PXI-4110
        VDC+6V_24_3:4_2
        VDC+6V_24_3:4_7              A

        VDC+25V_24_4:4_5     F
        VDC+25V_24_4:4_3     D

        VDC-25V_24_5:4_4     G

        VDCCOM_4_6           E
        VDCCOM_24_2          0

        SHORTCUT_24_6                A 0
        SHORTCUT_24_7                B 0
        SHORTCUT_24_8                C 0
        SHORTCUT_24_9                D 0
        SHORTCUT_24_10               E 0
        SHORTCUT_24_11               F 0
        SHORTCUT_24_12               G 0
        SHORTCUT_24_13               H 0
        SHORTCUT_24_14               I 0

* COMPONENTS CONFIGURED BY THE LAB STAFF

* Configurable component board 1
        R_1_7                C E 1k
        R_1_5                C E 2.7k
        R_1_3                C E 10k
        R_1_10               D E 1k
        R_1_9                D E 2.7k
        R_1_8                D E 10k
        SHORTCUT_1_14                D F
        SHORTCUT_1_13                E F
  (...)
* Dual component board 14
        R_14_18:14_1         A 0 330_X
        R_14_18:14_3         A D 330_X
        R_14_18:14_5         A F 330_X
        R_14_18:14_7         A H 330_X
        R_14_4:14_1          D 0 330_X
        R_14_4:14_5          D F 330_X
        R_14_4:14_7          D H 330_X
        R_14_6:14_1          F 0 330_X
        R_14_6:14_7          F H 330_X
        R_14_8:14_1          H 0 330_X

        R_14_17:14_16        A 0 680_X
        R_14_17:14_14        A D 680_X
        R_14_17:14_12        A F 680_X
        R_14_17:14_10        A H 680_X
        R_14_13:14_16        D 0 680_X
        R_14_13:14_12        D F 680_X
        R_14_13:14_10        D H 680_X
        R_14_11:14_16        F 0 680_X
        R_14_11:14_10        F H 680_X
```

Figure 2.49: Extract of a component list file example [41]

Chapter 3

Definition of the problem

This thesis aims to develop a remote electronics laboratory. It is intended to consist of an instrumentation device that contains the instruments planned for this laboratory (oscilloscope, function generator, digital multimeter and variable power supplies), a microcontroller that receives commands from the computer and controls several relays and a set of electronic components. It is through the closure of a set of relays that the intended circuit is built. In this chapter, it will be explained the hardware and software needed for this project.

3.1 Architecture

Figure 3.1 shows a diagram of the planned architecture. It is intended that the user accesses a web interface remotely. This interface exchanges data with a server that consists of power supplies, measuring instruments and a microcontroller that is part of the circuit. The circuit infrastructure can have several microcontrollers, relays and components. The designed circuit can be expanded by adding microcontrollers, relays or components.

The microcontroller is responsible for controlling the relays that consequently connect or disconnect the components to the circuit. The power supplies have the function of supplying energy to the components, and the magnitudes that pass through them can be measured through the instruments. It is intended to use only one hardware device, which contains all the necessary power supplies and instruments.

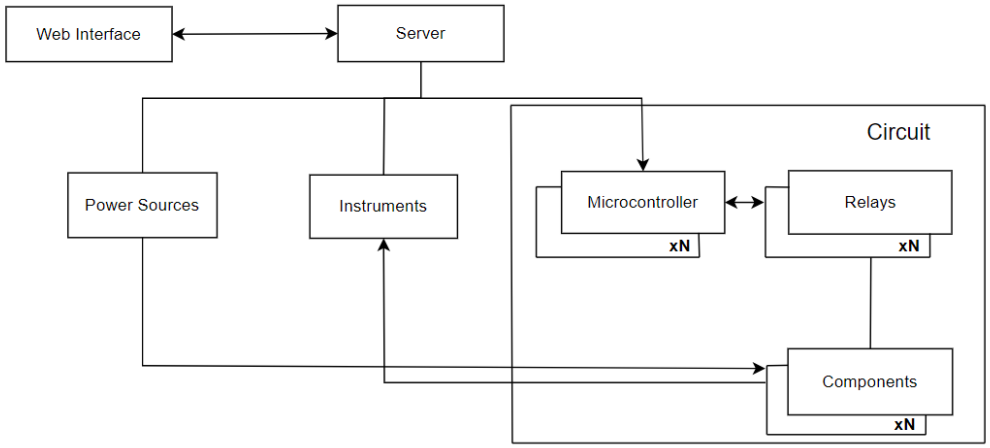


Figure 3.1: Planned architecture

The planned software should allow for the use of different solutions and a variable number of elements. The proposed prototype will represent a possible solution with a defined set of experiments, but both the software and hardware architectures are designed to allow variable setups and be expandable with additional or different hardware.

3.2 Hardware

In this section, the hardware components will be described. As mentioned, this laboratory will consist of a VirtualBench for instrumentation, an Arduino to control the circuit relays, relays and relays drivers. All these components will be explained in more detail in this section.

3.2.1 VirtualBench

The device suggested for instrumentation is VirtualBench VB-8012, Figure 3.2. VirtualBench is a device that has embedded five of the most widely used instrumentation devices, making test and measurement systems cheaper and more efficient. The instruments integrated here are a mixed-signal oscilloscope, an arbitrary waveform generator, a DMM, a programmable DC power supply and digital I/O. This device is capable of connecting to a computer through USB and WiFi [49][50].

The oscilloscope integrated here features two analogue channels and a bandwidth of 100 MHz. It also has an 8-bit resolution and a maximum peak-to-peak voltage of 24 V. The Function Generator (FGEN) can provide various waveforms such as sine, square and ramp/triangular, DC. The update rate is 125 MS/s and has a resolution of 14 bits. The DMM features the measurement functions of DC voltage,

AC voltage, DC current, AC current, resistance, diode and continuity. The DMM has a resolution of 51/2 digits and measures a maximum voltage of 300 V DC or AC_{rms} [51].

Figure 3.2: VirtualBench VB-8012

VirtualBench has software that can be downloaded from the National Instruments website and it can control all instruments present in VirtualBench. Figure 3.3 shows the interface of this software. At the bottom of the image, it's possible to see the controls for the DC power supplies and the multimeter where is chosen the magnitude wanted to measure and the measurement value is displayed. On the right, are the function generator controls, where it's possible to change the wave type, frequency, peak-to-peak voltage and offset. The graph that appears is related to the oscilloscope and it's possible to control the channels displayed and the horizontal and vertical scale [50].

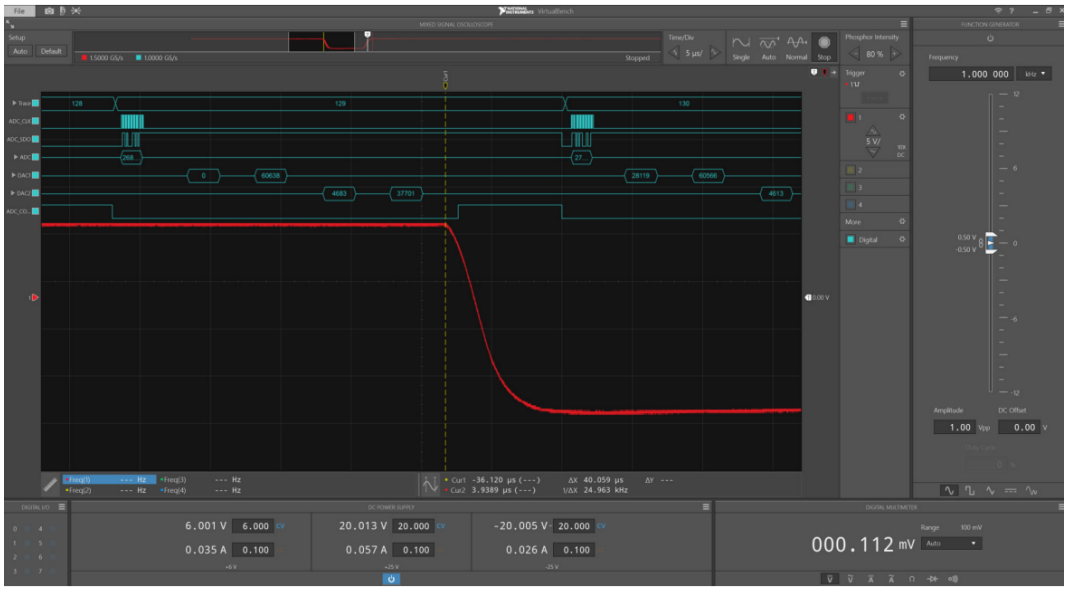


Figure 3.3: VirtualBench Interface [50]

There is also the possibility to access VirtualBench through LabVIEW. For this, it is necessary to install some libraries in LabVIEW. After that, LabVIEW has access to functions related to VirtualBench and it is possible to develop code like the one represented in Figure 3.4. In this example, LabVIEW is accessing the function generator and defining the waveform characteristics [50].

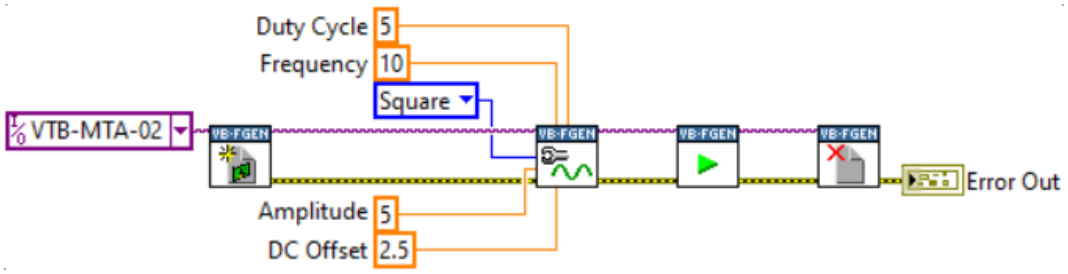


Figure 3.4: Controlling function generator through LabVIEW

3.2.2 Arduino

There are several microcontroller options such as Atmega168, ESP32, etc. Although Arduino is included in the products considered Commercial Off-The-Shelf (COTS). These refer to products designed for general use rather than made for a specific application. Arduino provides pre-designed hardware and software solutions that make it easy to use for a variety of projects. Arduino boards are very affordable and include a user-friendly development environment and well-documented libraries. Arduino also supports a variety of sensors, actuators and communication interfaces, making it suitable for a wide range of applications. For these, and several other reasons, Arduino was chosen as the microcontroller for this project.

For this thesis, it was suggested the Arduino Mega. As mentioned before, the Arduino Mega will be responsible for controlling all the circuit relays. Hence, Arduino digital pins will be physically connected to relays (through relay drivers). This Arduino is the microcontroller (Arduino product family) which has more digital I/O pins (54).

Figure 3.5: Arduino Mega [52]

3.2.3 Relays

As mentioned, the remote lab hardware is based on having all components, power supplies and instrumentation physically connected and will be switched on or off from the circuit.

There are various components responsible for switching circuits, such as transistors, relays, multiplexers and so on. In this situation, it's intended that the switching component does not interfere with the circuit and also has a low cost as this type of circuit will require a considerable number of these components. For example, if it is planned to connect a resistor in series with a source, it's not intended that the switch used changes the value of the current in the branch.

So the component that doesn't generate a significant voltage drop when switched on is the relay. This component can isolate the control circuit (the coil) from the power circuit (contacts).

On the other hand, components such as transistors have an associated voltage drop when they are in conduction caused by the characteristics of these components. Multiplexers also lack electrical insulation and are more limited in terms of the current capacity they can switch. Considering the technical and economical objectives, relays were chosen for this project. Relays are components that behave like switches through a control signal.

In this project were used SPST relays and DPST relays. The SPST relays are used to connect the power supplies and measuring instruments to the nodes. In the case of DPST relays are used to connect components to the circuit (including shortcuts). This separation is made, since the double relays that are connected to the components, are responsible for connecting the relay to two nodes, but these nodes can then be changed by the teacher/technician through two jumpers.

Figure 3.6 shows the pinout of the 3570-1331-123 SPST relay. As can be seen in Figure 3.6 there is a coil inside the relay, which when powered by an external power supply (+12 V), the electric current travels through the coil and creates an electromagnetic field that attracts the contact present between pins 1 and 4 and consequently closes the circuit. When this voltage is no longer present at the coil terminals, the contact opens, opening the circuit [53][54]. So, for example, it can be connected the pin 1 to node A and pin 4 to the positive terminal of a DC power supply (the negative terminal is connected to the ground) and, when the relay is energised, the supply provides voltage at node A.

Between pins 2 and 3, there is a diode, called Flyback diode or Freewheel diode. This component is connected in parallel with the inductive coil to prevent voltage spikes when the power to the coil is switched off. The coil is magnetically charged when powered and stores the energy of this magnetic field. When the power supply is interrupted, the coil current flows in the opposite direction as it decreases.

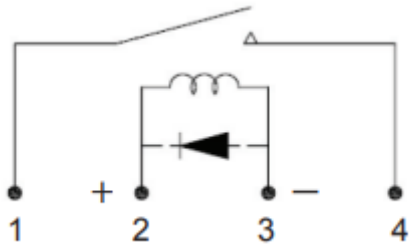


Figure 3.6: SPST Relay [54]

The DPST relay has a similar operation to the SPST with the difference that it has two contacts, i.e. two switches in the circuit. When the coil is energised, the two contacts close simultaneously. For example, if it's placed a component between pins 1 and 14 (Figure 3.7), and connects pins 7 and 8 to nodes A and B respectively, the

component is switched on between these nodes when the contacts are closed [53]. The DPST relay provided has the series number 3572-1220-123.

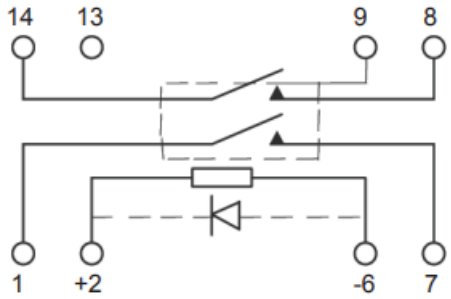


Figure 3.7: DPST Relay [55]

Figure 3.8 shows a diagram explaining how to connect the components to the nodes. The teacher/technician will connect the components to socket pins that are connected to the dual relays and then connect the relay pins to the circuit nodes via jumper wires. The components connected to each relay can be changed by the teacher, as well as the nodes to which they are connected, at any time. In the example presented in Figure 3.8, the resistor is connected between nodes C and 0 and the diode between nodes A and B.

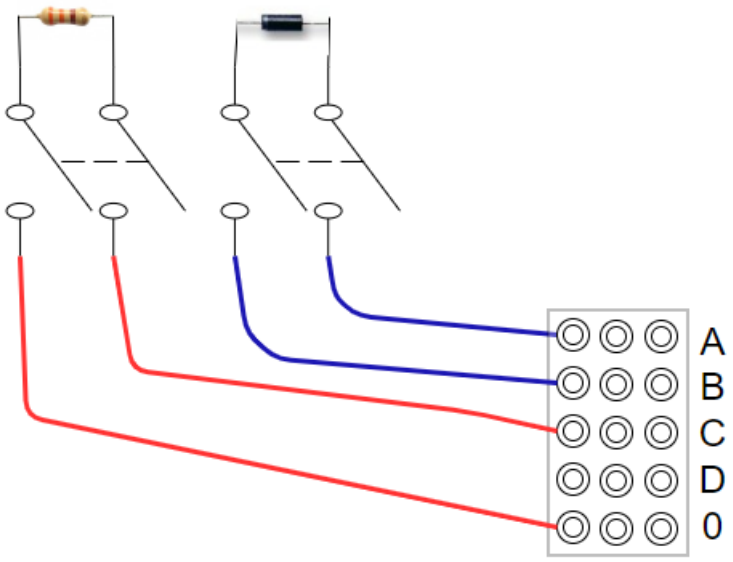


Figure 3.8: Components placement diagram

Figure 3.9 shows an example circuit of the connections made above. There are fixed power supplies between nodes A and 0 and the components are physically

connected to the nodes indicated above. The switch represents the operation of the relays.

If the teacher considers it necessary to have a resistor between nodes A and B, he needs to change the jumpers for those nodes and then the user chooses whether to use that resistor in his circuit by closing the relay.

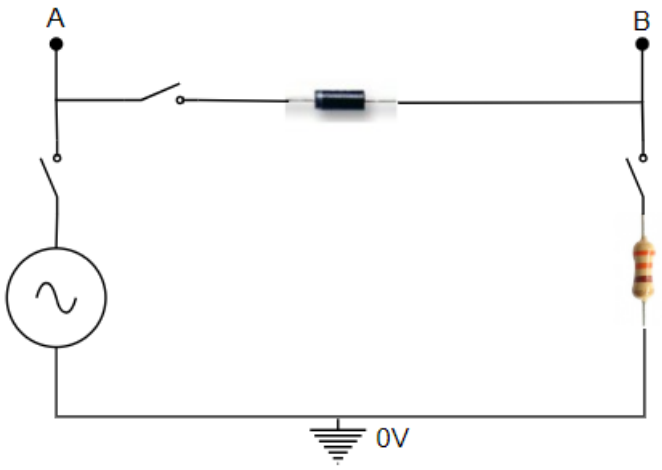


Figure 3.9: Circuit example

3.2.4 Relays Driver

The output power of the Arduino (or other microcontrollers) is too low to drive loads that require higher voltages or currents such as relays. In these situations, relay drivers are used. The driver used is the ULN2003. This driver consists of a set of seven Darlington transistors that have high voltage outputs and feature a clamping diode for switching inductive loads. The nominal collector current of a pair is 500 mA, and pairs can be placed in parallel to obtain a higher current, and the output voltage can be up to 50 V. The input pins can be activated with +5 V [56][57]. Figure 3.10 shows the ULN2003 pinout configuration.

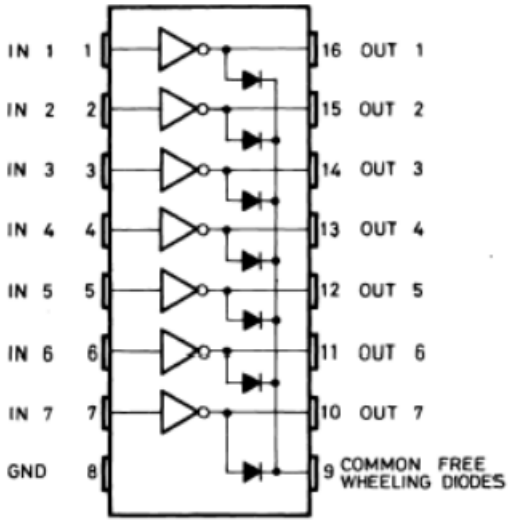


Figure 3.10: ULN2003 pinout [58]

3.3 Software

This section aims to present the software that will be used for the development of the remote laboratory. LabVIEW software will be used for the development of the user interface and Proteus simulation software will be used for the circuit simulations before physical testing.

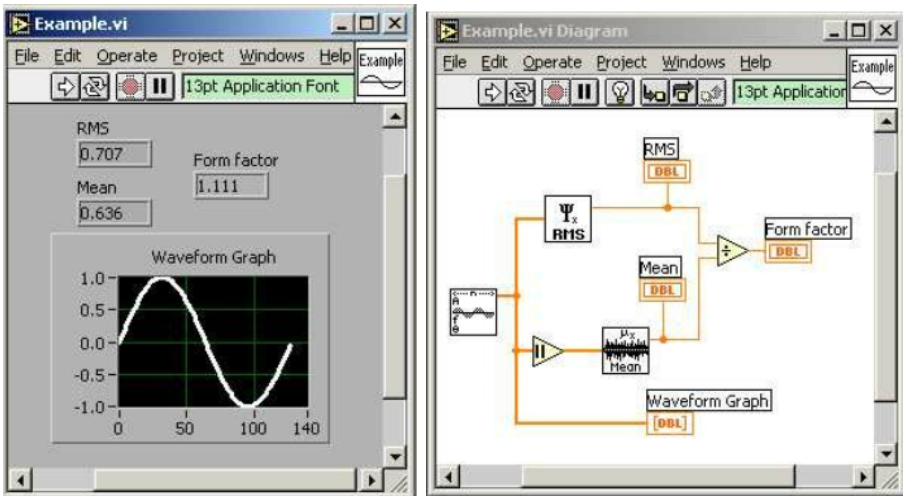
3.3.1 LabVIEW

LabVIEW is a software development environment that uses a graphical interface for programming in G. This programming language was created by National Instruments and is used for automated test applications, data acquisition, and Proportional – Integral – Derivative (PID) control, among others.

Programming in G is carried out using building blocks which, when connected, build up the data flow. Each block performs functions such as basic arithmetic, if/then/else if conditional statements, case statements, filtering, etc. Another feature of LabVIEW is the ability to offer real-time compilation. Each LabVIEW function is associated with a graphical design in the user interface allowing immediate interaction with the written code [59].

A LabVIEW program is called Virtual Instrument (VI). When a VI is created, two windows are displayed: the front panel and the block diagram. The front panel is the user interface and the block diagram is the window where the code is developed [60].

Figure 3.11a and 3.11b show a simple example of a LabVIEW front panel and a block diagram, respectively.



(a) (b)

Figure 3.11: LabVIEW VI [61]: (a) Front panel and (b) Block diagram .

LabVIEW runs on Windows, Apple operating systems and Linux platforms, so it is suitable for almost all computer systems.

LabVIEW was chosen for the development of this remote lab since it has features such as:

- Web publishing tool, a tool that allows the VI to be published on a web server through a server and accessed remotely by clients;
- Functions capable of interacting with VirtualBench instruments, controlling them and reading data;
- Ability to communicate with Arduino by sending a string to control the relays.

3.3.2 Proteus

It is very important to use simulation software before implementation since the software analyses the circuit presenting the predicted values and possible issues that the circuit may cause, thus avoiding the damage of equipment. In addition, discovering an error in the circuit while simulating is much easier than in the practical circuit.

Proteus is a software developed by Labcenter Electronics and is used to design and simulate electronic circuits. This software features several components used in electronics (resistors, relays, integrated circuits, etc.) and several instruments such as oscilloscope, voltmeter, etc.

Proteus was used instead of other simulation software since it allows the integration of a microcontroller and tests the circuit considering the code embedded in the

microcontroller. Figure 3.12 shows the Proteus interface with an example circuit containing an Arduino [62].

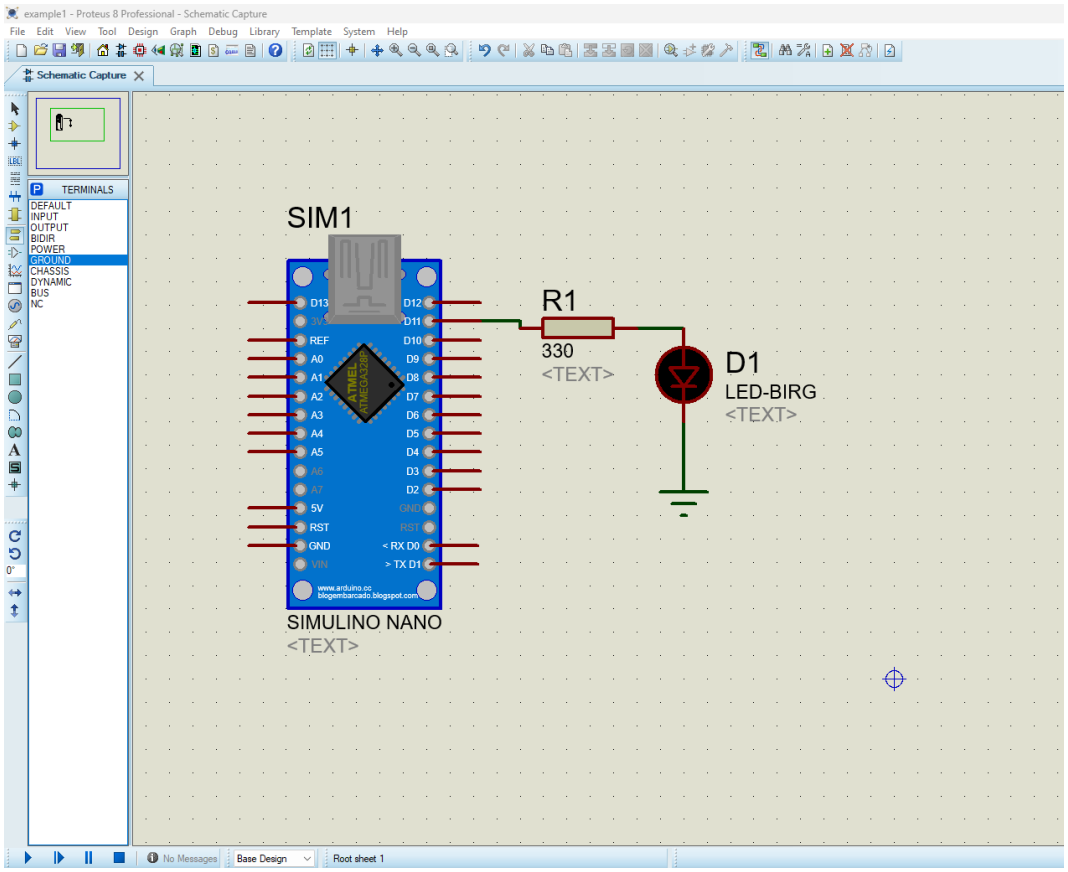


Figure 3.12: Proteus interface

3.4 Circuits to be tested

Some circuits are the basis of electronic engineering and are always used in the life of an engineer. Therefore, engineering students need to understand the workings of these circuits and their concepts. This chapter presents the suggested circuits that are indispensable for an electronics engineering student and will be tested with the developed remote laboratory.

3.4.1 Ohm's Law

Ohm's law is a formula that relates three quantities: resistance (R), voltage (U) and current (I). These three quantities are linked by the equation 3.1. This law is the basis of electronics and is essential for studying the values of these quantities present in circuits.

$$R = \frac{U}{I} \tag{3.1}$$

To test this law is pretended to test a simple circuit as shown in Figure 3.13. This circuit consists only of a source and a resistor and it is intended that the student can measure the three quantities (R, U and I). It is also intended that the student can change values (R and U) so that he always checks the relationship of the equation 3.1 for different values.

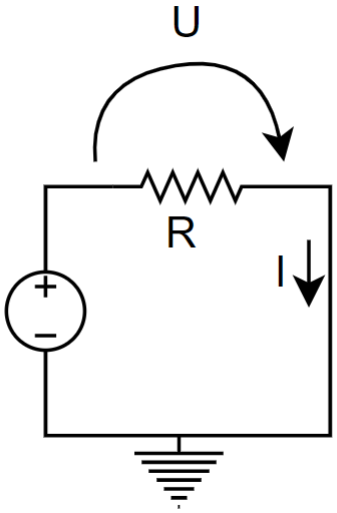


Figure 3.13: Ohm's law circuit

3.4.2 Voltage and Current Divider

The main types of component placement can be divided into series and parallel configurations. A circuit in which the components are in series behaves as a voltage divider, since it divides the total voltage, applied by the power supply, into different voltages across the elements of the circuit. Figure 3.14 shows a voltage divider that divides the total voltage applied by the power supply into the voltages U1 and U2 across the resistors R1 and R2. As the resistors are on the same branch, the current is the same.

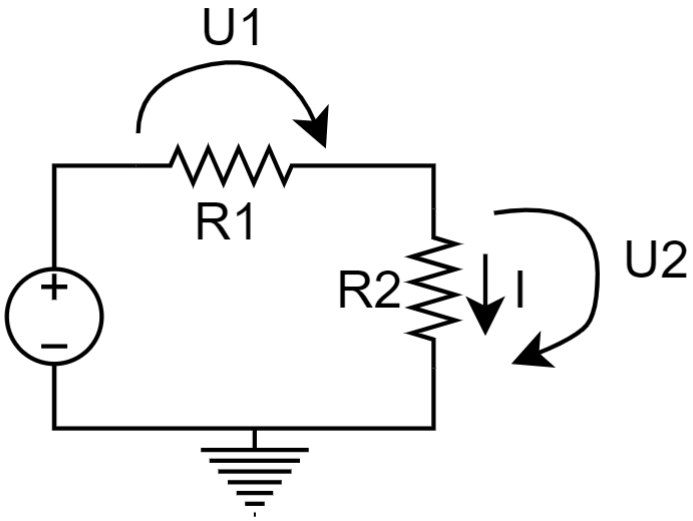


Figure 3.14: Voltage divider circuit

In a circuit where the components are in parallel, the circuit behaves like a current divider, dividing the total current coming from the source across all the branches. Figure 3.15 shows a current divider that splits the total current (I1) into the currents I2 and I3. However, the voltage drop across both resistances (R2 and R3) is the same.

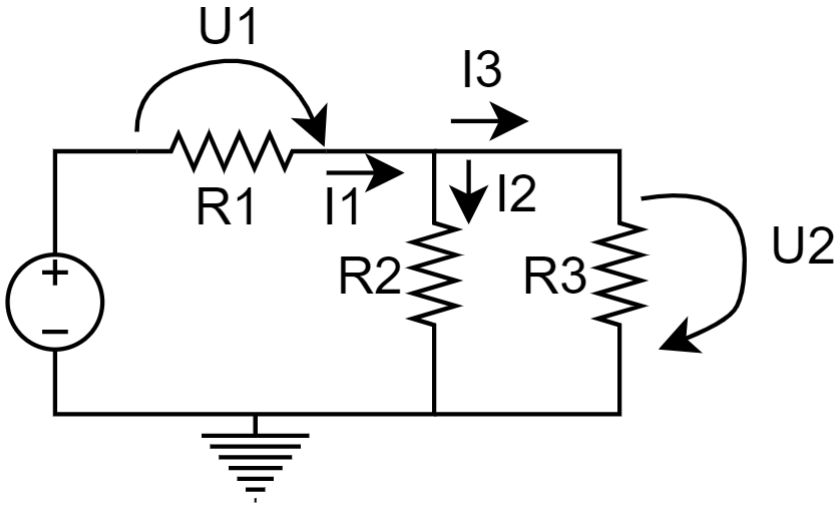


Figure 3.15: Current divider circuit

In these two circuits, it is intended that the student can measure all the quantities shown in Figures 3.14 and 3.15.

3.4.3 Kirchhoff's Laws

Kirchhoff's laws handle the conservation of charge and energy in electrical circuits. The first law states that at any node in an electrical circuit, the sum of the currents flowing into that node is equal to the sum of the currents flowing out of that node, i.e. the current is conserved.

The second law is related to voltage and states that, according to the principles of conservation of energy, the sum of the potential difference around each closed loop is equal to zero. Figure 3.16 shows a circuit where Kirchhoff's laws can be verified. For this example, Kirchhoff's laws are 3.2 and 3.3. The student has to be able to measure all the resistance, voltages and currents represented in Figure 3.16.

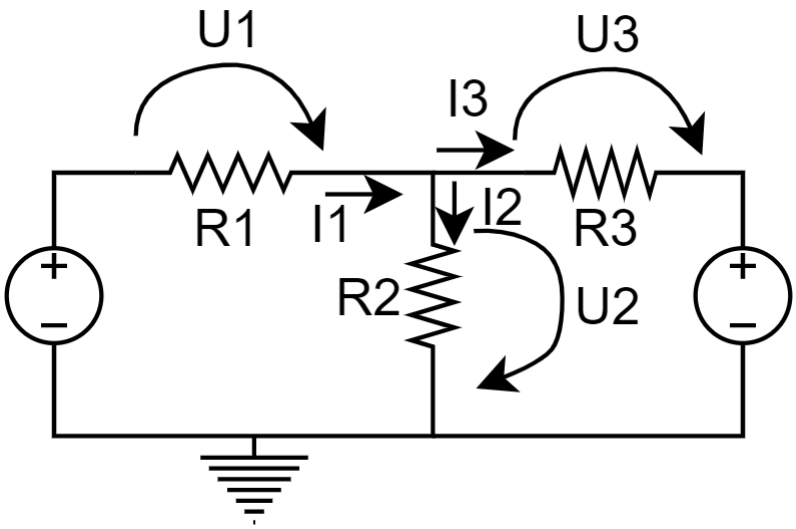


Figure 3.16: Kirchhoff's laws circuit

$$I1 = I2 + I3 \tag{3.2}$$

$$U(Source) - U1 - U2 = 0 \tag{3.3}$$

3.4.4 Half-wave rectifier

A half-wave rectifier has the purpose of converting a AC signal into a DC signal by letting the positive or negative half-cycle pass and blocking the other. This can be achieved with the use of a diode, as this component is only passed through by current in one direction.

Figure 3.17 shows a half-wave rectifier circuit. With a diode and a resistor, it's possible to verify the input waveform with oscilloscope channel 1 connected to the

function generator and the output waveform with channel 2 connected to the resistor [63].

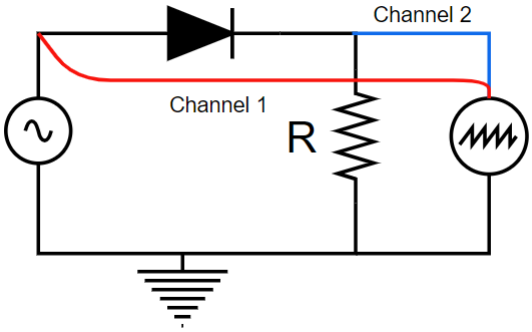


Figure 3.17: Half-wave rectifier circuit

As explained, the input waveform should be similar to Figure 3.18 and the output waveform should look like Figure 3.19.

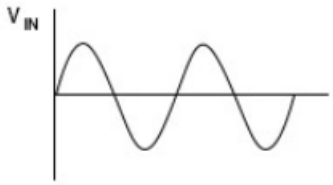


Figure 3.18: Half-wave rectifier input waveform

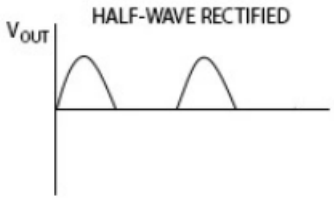


Figure 3.19: Half-wave rectifier output waveform

Chapter 4

Proposed Solution

The proposed solution was developed based on other laboratories already in use, in order to study the best approaches for its development and study its limitations to correct them or add some features.

This laboratory enables students to perform experiments related to electronics such as Ohm's law, Kirchhoff's law, half-wave rectifiers and the measurement of current, voltage and resistance in circuits with resistors.

This system allows the user to change settings and access instrumentation devices (oscilloscope, multimeter, DC power supplies and function generator) embedded in a device called NI VirtualBench. This lab has two different user interfaces, one for the student and one for the teacher/technician. The student interface allows the student to define the characteristics of the measurement instruments, choose the components that are between each node of the circuit and perform the experiment by obtaining real values from the VirtualBench. The student can choose whether to perform the experiment and obtain the desired value, or view in real time the change of values while the experiment is performed. This feature and the ability to choose between the two options do not exist in most remote laboratories. The teacher/technician interface allows them to perform the same functions as the student with the addition of being able to configure between which nodes each component is physically located, and they can also enter a test mode to verify the proper functioning of the relays.

Figure 4.1 shows the global architecture diagram of this remote laboratory. The computer server hosts the two interfaces developed in LabVIEW. As said before, these interfaces will communicate with VirtualBench instruments for sending device

parameters and receiving measurement values. The LabVIEW software also communicates with the relay switching circuit. In short, this circuit has relays that connect or not (if turned or on or not) a component, a source or an instrument between two nodes and then the desired circuit is built by the user. To turn on or off the relays, a string is sent via USB from the LabVIEW interface to an Arduino. This communication will be explained in further detail later. There is a file that defines the circuits that are allowed to be built, in order to protect the equipment, located in the server and checked every time the user performs an experiment.

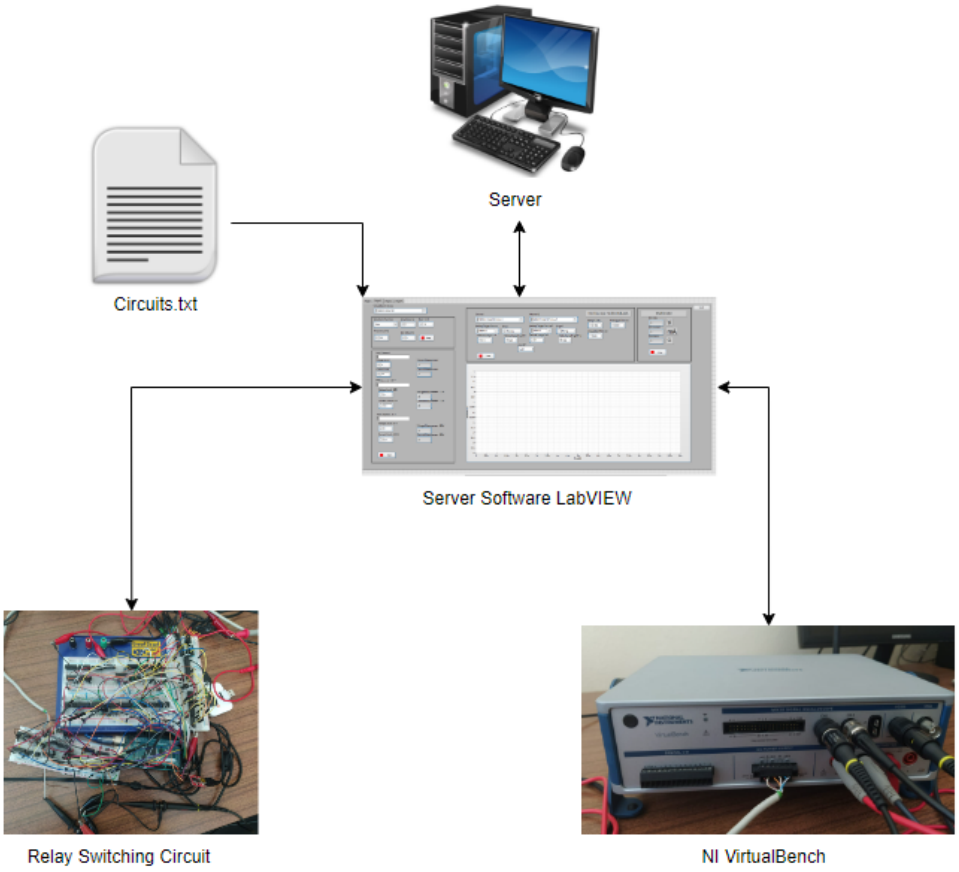


Figure 4.1: Global architecture diagram

4.1 Hardware

This section aims to describe the hardware indicated in chapter 3 and how the hardware components are integrated with the circuit. How the measurements are made and the current circuit configuration are also presented. Finally, it is explained how to build the desired circuit.

4.1.1 Communication circuit

The information related to whether the relays are closed or not is sent in the format of a text string from the computer to the Arduino. This string results from the concatenation of four lines, related to the four types of boards (Component board, DMM board, Oscilloscope board and Source board) that follow the format previously presented in the VISIR communication. This format was selected for a possible future integration of both laboratories.

The computer is physically connected to the Arduino through the programming cable, as such the information could be sent through this cable. However, while testing the Arduino code and then sending the string, there were some conflicts accessing the serial port so it was decided to use a serial port for programming and another to receive the text string. For this, it was necessary to use a USB to serial converter and connect it to pins RX1 and TX1 of the Arduino.

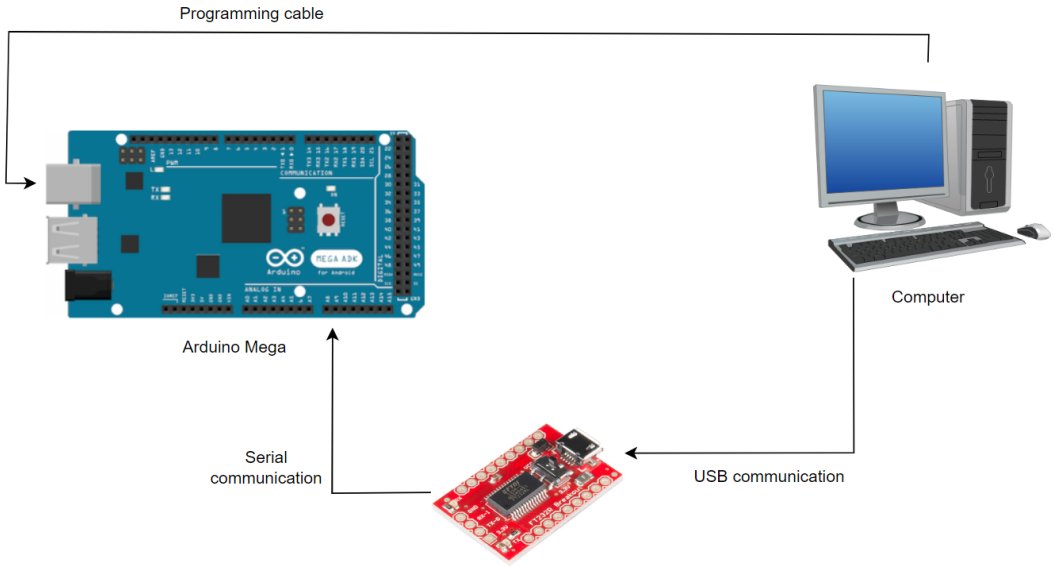


Figure 4.2: Communication circuit diagram

As stated above, the Arduino will control the relays via the relay driver. Each label represented in Figure 4.3 is connected to a pin of a relay driver. As a Printed circuit board (PCB) will be implemented later, and the boards will be divided (for sources, digital multimeter, oscilloscope and components), a driver (or more) has already been used for each functionality. There are two drivers for components and each pin connected to the Arduino starts with RC, two drivers for the multimeter and the pins start with RV, one driver for the sources and one for the oscilloscope and the label starts with RS and RO, respectively.

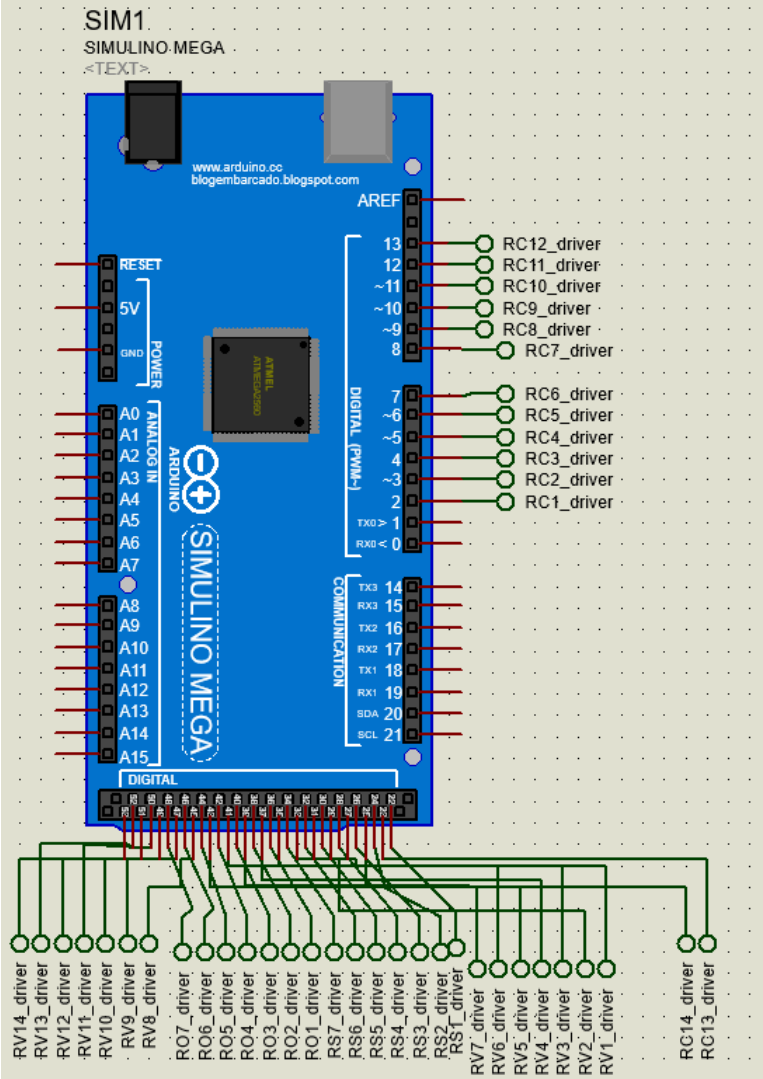


Figure 4.3: Arduino connections in Proteus

4.1.2 Voltmeter connections

Measuring instruments must be able to connect their terminals to any of the nodes in the circuit to be able to make all the measurements in the circuit.

As the voltage is measured in parallel, allowing both voltmeter terminals access to all nodes ensures that all possible voltages in the circuit can be measured.

This is shown in Figure 3.6, using simple relays that allow the positive terminal of the voltmeter to have access to all nodes, as well as the negative terminal of the voltmeter. This approach implies that the number of relays for the voltmeter is twice the number of nodes in the circuit.

This solution is the simplest way to obtain all combinations of nodes connected to the voltmeter while using as few relays as possible. Since the positive and negative terminals can connect to any node, it means that both terminals can connect to the

same node. In the case of the voltmeter, this does not cause any complications for the instrument, since it has a high impedance and will only give 0 V potential difference.

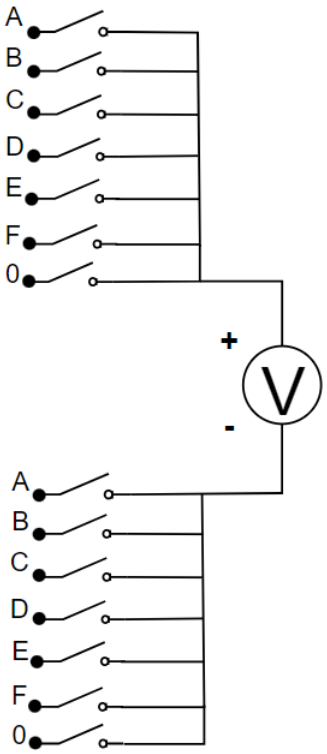


Figure 4.4: Voltmeter connections with SPST relays

4.1.3 Ammeter connections

An ammeter measures the current in a branch when the circuit is opened in that branch and the ammeter is placed in series with the components. Therefore, to measure the current, the ammeter is placed between two nodes where no component is located. For example, Figure 4.5 shows a circuit with two resistors and a DC power source, and if no current was to be measured, only three nodes would be needed. To measure the current, an extra node is needed to place the ammeter between B and C. When the current is not being measured, there is a shortcut between B and C to close the circuit so that it can operate normally. Every two nodes in a row have a shortcut so there is a possibility to measure the current there. This is because it allows current to be measured in all branches, depending on where a component is located. For example, if there was a resistor between nodes B and C, the current would be measured between A and B (by connecting the ammeter there and checking that no component is connected here). If it's intended to test the circuit without the ammeter, the user must connect the shortcut between A and B to close the circuit.

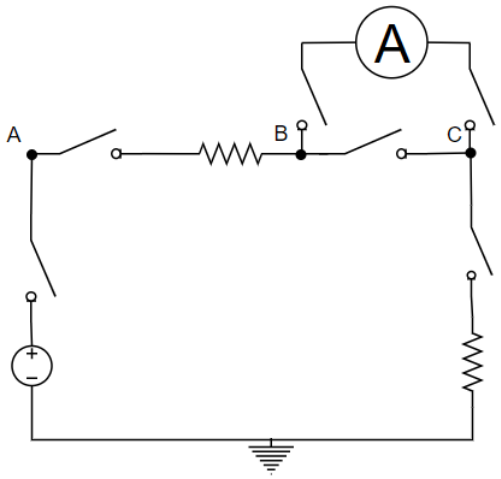


Figure 4.5: Ammeter connection to circuit

So, similar to the voltmeter, the ammeter will also have to come up with the combinations needed for each of the terminals to be able to connect to any node. Instead of replicating the same configuration used for the voltmeter, we opted to reuse the voltmeter relay infrastructure and add a couple of relays to select the multimeter connector (ammeter or voltmeter) being used. Thus reducing from 28 relays to 16 in this situation.

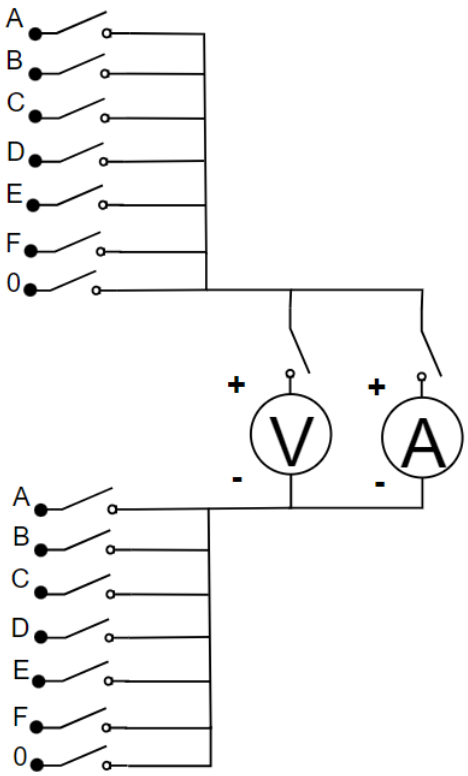


Figure 4.6: Ammeter and Voltmeter connections

Figure A.1 in appendix A shows the voltmeter and ammeter connections in Proteus. The connections are the same as those shown in the diagram in Figure 4.6, however, the relay symbol and its connections are shown here. It is important to note the A label connected to the coil of each relay since it is this connection that is responsible for the relay opening or not. Each relay has a label in the format RVx where x represents a number. Each label means a connection to an output pin of the relay driver. When it's needed to close a relay, the voltage of the Arduino pin is fixed at +5 V and, consequently, the input pin of the relay driver. The operation of the driver causes the output pin to have a voltage of 0 V (or close to 0 V) or disconnected (open). When the pin has 0 V, as it is connected to the coil pin of the relay, the potential difference in the coil is +12 V (since the other coil pin is connected to +12 V) and the relay closes. When the driver output pin is disconnected, the coil (of the relay) pin it is open and there isn't current in the relay coil (relay is open).

and, as this pin is connected to the coil pin of the relay, the potential difference in the coil is +12 V (since the other coil pin is connected to +12 V) and the relay closes. The number entered in the label to which each relay is connected is important for the way the Arduino software activates the pins and will be explained in detail in the software section.

As already mentioned, 16 relays are required for these connections. Node F

was added for the specific situation of measuring currents in more complex circuits. Fourteen (apart from the two connections to node F) relays were connected to the multimeter drivers, the pins of the two drivers being fully occupied. As node F was added later, for practical reasons these two connections were added to the source driver which was not fully occupied.

Figure 4.7 shows the two drivers used to control the relays of the digital multimeter. On the left are the driver input pins that are connected to the Arduino pins. On the right are the driver output pins, connected to a relay coil pin.

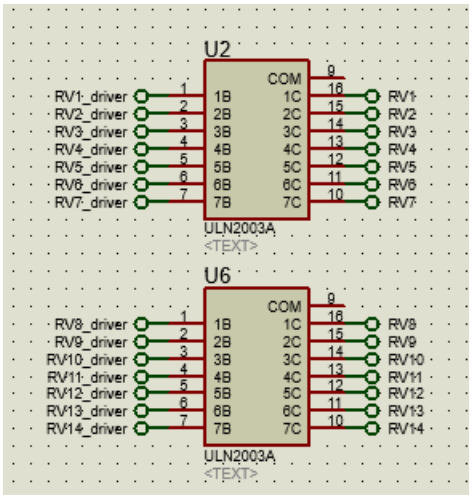


Figure 4.7: Relay drivers for Voltmeter and Ammeter connections

4.1.4 Oscilloscope connections

The oscilloscope connections are similar to those presented above, with some singularities. The oscilloscope has two channels, so it needed to connect to twice as many circuit nodes. Another peculiarity of the oscilloscope is that the ground is directly connected to the ground of the circuit and cannot connect to any other node. Therefore, the oscilloscope needs twice relays than the number of existing nodes (for the two channels) minus two, since none of the channels will connect to node 0 (ground). Figure 4.8 shows the connections of the oscilloscope, with the particularity that the two channels are only connected from node A to C. The oscilloscope should (and could) be connected to all nodes, but in practice, a reduced configuration was implemented, since towards the end of the practical development the digital pins of the Arduino were becoming scarce and priority was given to other applications. It should be noted that it did not influence anything in the experiments carried out since nodes A and B were reserved for experiments that use the oscilloscope. This situation was the only one that needed to implement fewer nodes than necessary due to the lack of pins, and if necessary to use the oscilloscope with more nodes, some other experiments could have been removed. As indicated above, the software

and hardware are prepared to be expanded in order to handle these limitations, in this case requiring an additional Arduino board.

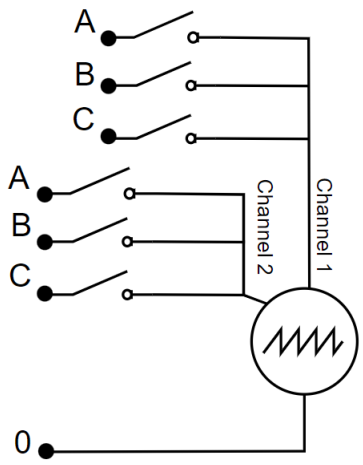


Figure 4.8: Oscilloscope connections

Figure 4.9 shows the oscilloscope connections in Proteus. The Proteus oscilloscope allows you to visualise up to four channels and it is not necessary to make the connection to the ground, it already does it internally. As shown, nodes A, B and C can be connected to both channels. In Figure 4.9 the channels are represented as A and B (the ones being used). Figure 4.10 shows the driver used for the oscilloscope connections.

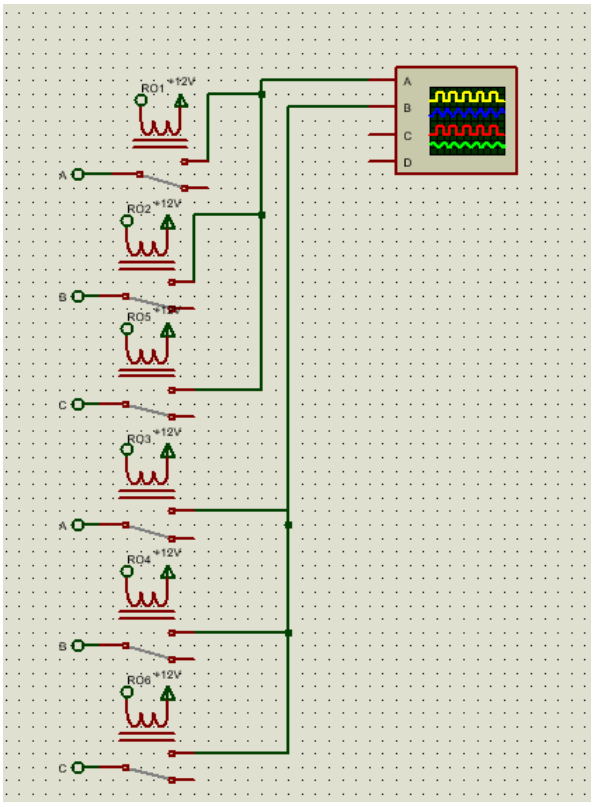


Figure 4.9: Oscilloscope connections in Proteus

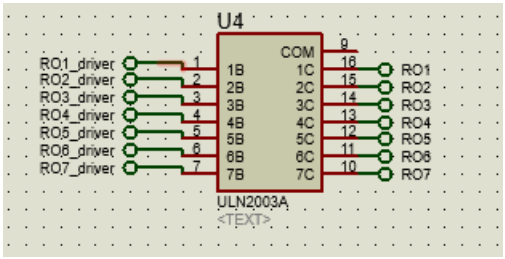


Figure 4.10: Relay drivers for oscilloscope connections

4.1.5 Components connections

Figure 4.16 shows an example of how the components connect to the nodes. As mentioned above, a component is connected between two of the relay's pins and another two pins are linked to the nodes to connect the component to the circuit when the relay is closed. All the wires shown in colour in Figure 4.16 represent jumper wires that allow the teacher/technician to change the nodes to which the component is connected according to the experiments he wants. Components can also be removed and new ones added, depending on the experiments being tested. In the future, when a PCB is implemented, the connection to the nodes will be made via jumpers and the components will be placed in female connectors.

If a shortcut between two nodes is desired, the nodes are connected like the components and a jumper wire is placed in place of the component, as shown in black in Figure 4.16.

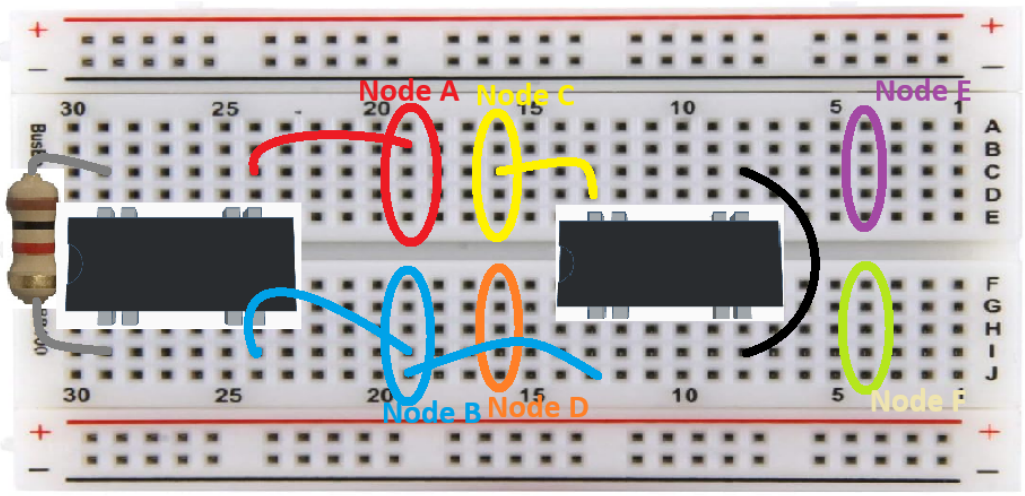


Figure 4.11: Components connections

Figure B.1 in the appendix B shows the circuit (without instrumentation) in Proteus. In this circuit, it is possible to visualise the present connections of the components and shortcuts. Whenever any component or connection to the nodes is changed, the component or node labels must be changed in the simulation circuit.

4.1.6 Circuit without instrumentation

Considering the intended experiments (referenced in chapter 3) the components chosen for the circuit were: two diodes (1N4001), one polarised directly and the other indirectly, and four resistors (330 Ω, 680 Ω, 1 kΩ and 4.7 kΩ). The sources used were: the function generator and the +6 V and ± 25 V variable voltage sources.

The power supplies are connected to specific nodes instead of the components and shortcuts that can connect to any two nodes through jumpers. Figure 4.12 shows the circuit that is currently assembled. The black wires represent the fixed connections and the blue wires represent the jumpers that can connect the component between two nodes of the circuit. It is the teacher/technician who chooses the connections of the components and has to consider the experiments he wants the student to perform.

This circuit has 6 nodes (A, B, C, D, E, 0) where components can connect (apart node F). With this circuit, when the user wants to experiment, for example, the Kirchhoff laws, he cannot measure the current in the middle branch (where the resistance of 1 kΩ or 4.7 kΩ is). For this reason, node F was created so that it would be possible to measure the current in this specific situation of the 1 kΩ resistance

in the Kirchhoff laws. Furthermore, this situation also emphasises the difficulty and complexity of having the possibility to measure current in all branches of the circuit. The more complex the circuit and the more branches it has, the number of nodes has to grow exponentially.

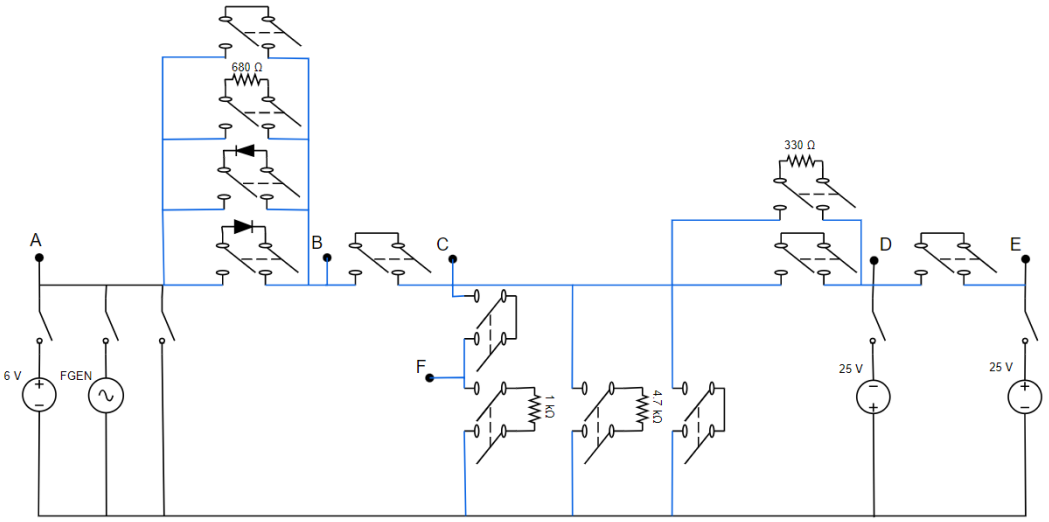


Figure 4.12: Present circuit diagram

Figure B.1 in appendix B shows the circuit with sources, components and shortcuts in Proteus. In addition to the components, the shortcuts are also connected to the drivers of the components so that the teacher can change a shortcut for another component for an experiment he wants, and the nodes where the shortcut is located can also be changed through jumpers. Twelve relays are needed to control the components used, so two relay drivers are used for the components, Figure 4.13.

Four relays were used for the sources plus two relays to connect node F to the multimeter plus a relay for a shortcut between node A and 0. If the user intends to use the sources connected to nodes D and E, it may be necessary to close the circuit between A and 0. That is, seven relays connected to the source driver were used, so it was only necessary to use one driver, Figure 4.14.

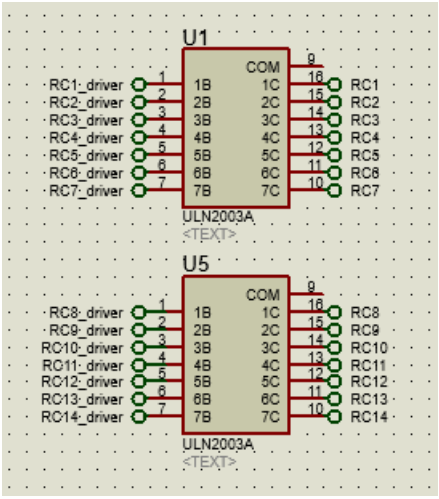


Figure 4.13: Relay drivers for component connections

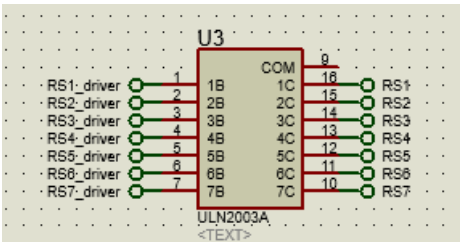


Figure 4.14: Relay drivers for sources connections

Figure 4.15 shows the implementation of the circuit. It indicates where the relays and drivers related to each functionality are located, and the connections made to VirtualBench and the external power supply for the relays. Note the difference in relays from the components (DPST) to the others (SPST).

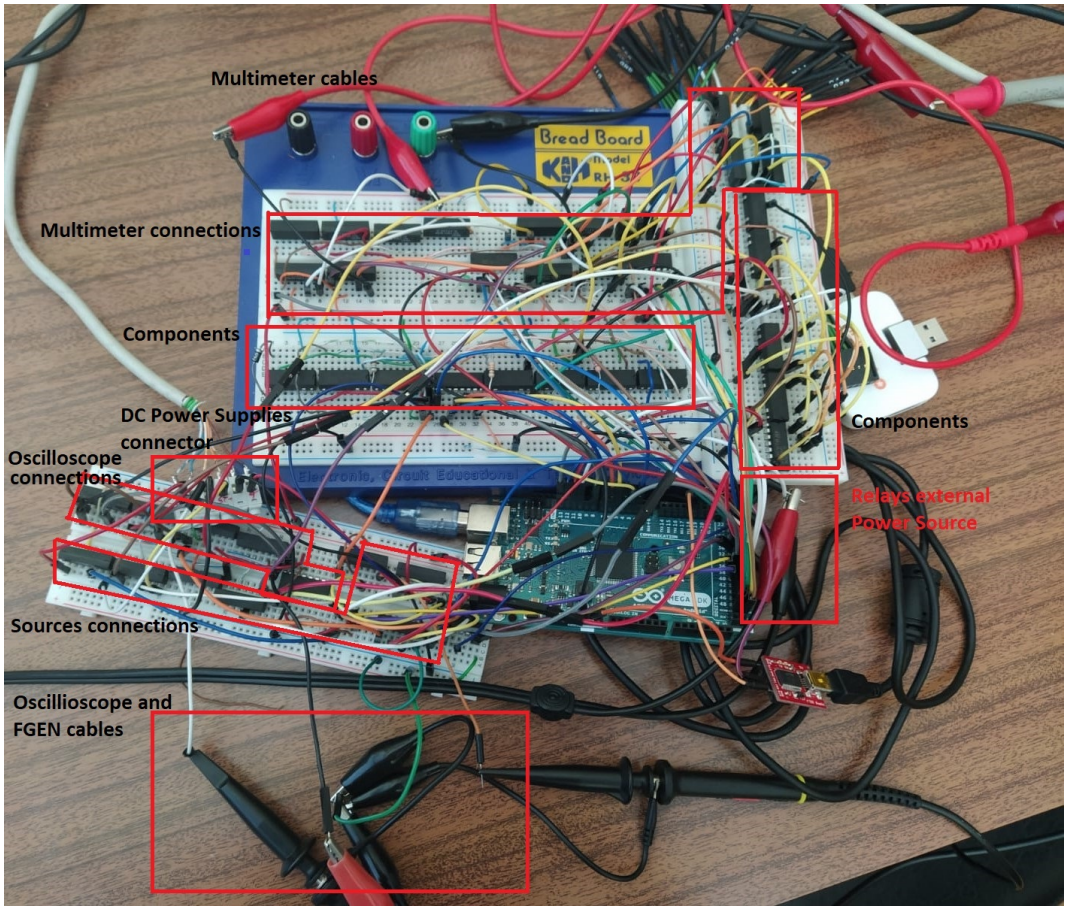


Figure 4.15: Circuit implementation

Figure 4.16 shows an example of a component's connections. The diode highlighted in blue will be connected to nodes A and B through two jumpers (it can be connected to any other node) as highlighted in red in the image. One of the pins related to the relay coil is connected to +12 V and the other one shows a connection to a component relay driver pin.

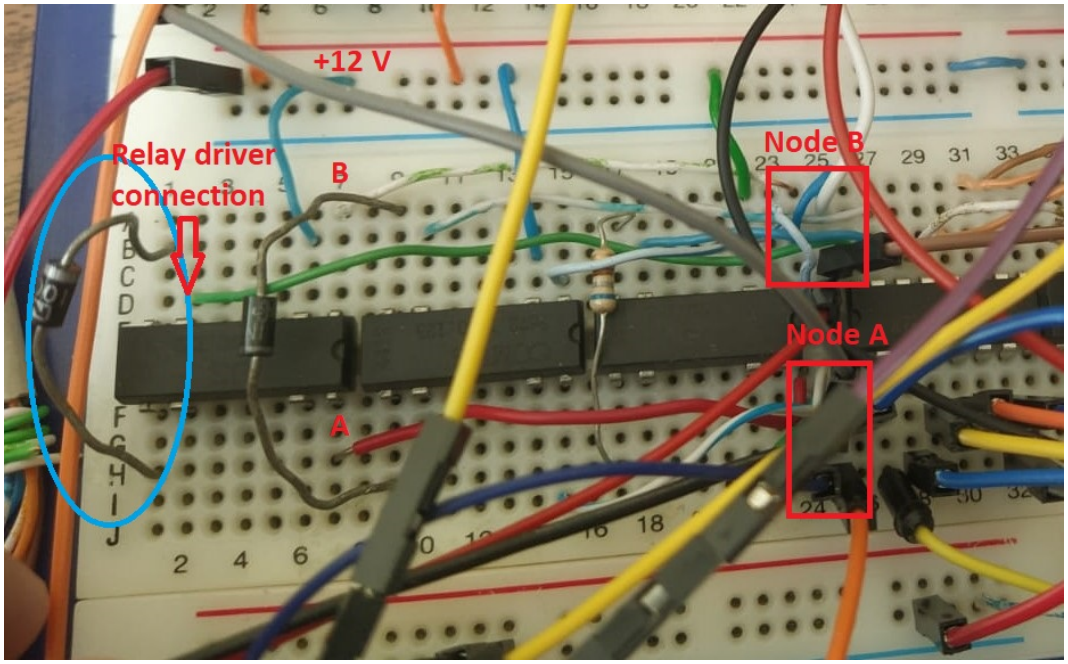


Figure 4.16: Component connections

4.2 Software

This section describes the software developed. Firstly, the interfaces developed are described from the user's point of view. Then the Arduino code responsible for receiving the string and controlling the relays is presented. Finally, the code developed in LabVIEW for the two interfaces is explained.

4.2.1 Teacher/Technician Interface

The teacher/coach interface has four pages. The main page, shown in Figure 4.17, allows the teacher to choose between which node each component, power supply and shortcut is physically located and then select ON or OFF depending on the circuit they wish to test. Next, you have to indicate which and where to connect the measuring instrument(s). In the DMM the user has to select if he wants it to work as voltmeter/ohmmeter or ammeter and then select in which node each of the terminals is connected. In the case of the oscilloscope he only has to indicate which node each channel is connected to.

Taking into account the selected items, the code developed in LabVIEW will calculate the final string that will be sent to the Arduino and is displayed only in the teacher interface for verification and debugging in case of any error.

At the bottom of the page, several buttons are displayed. The button "Instrumentation" is used to navigate to the page of Figure 4.18 where the user needs to choose the settings of each measuring instrument that he wants to use and visualise

the values if the experiment is being carried out in real-time. There are two ways to experiment, as mentioned before, the default mode being in real-time by clicking on the "Close Relays" button and checking the values on the instrumentation page. When the teacher has already obtained the desired values he must click on the "Open Relays" button to open the relays. The other way to experiment is to click on the "Perform Measurement" button and the desired relays are closed, the measurement values are recorded and you are directed to the Figure 4.19 page to visualise the results obtained.

In this interface, there is also the possibility to choose the time, in seconds, needed to perform the experiment and obtain the results when the "Perform Measurement" button is pressed. This feature was built due to the different times needed for different experiments and to make it possible, in the future, to carry out experiments that require a time constant, such as capacitors load and unload.

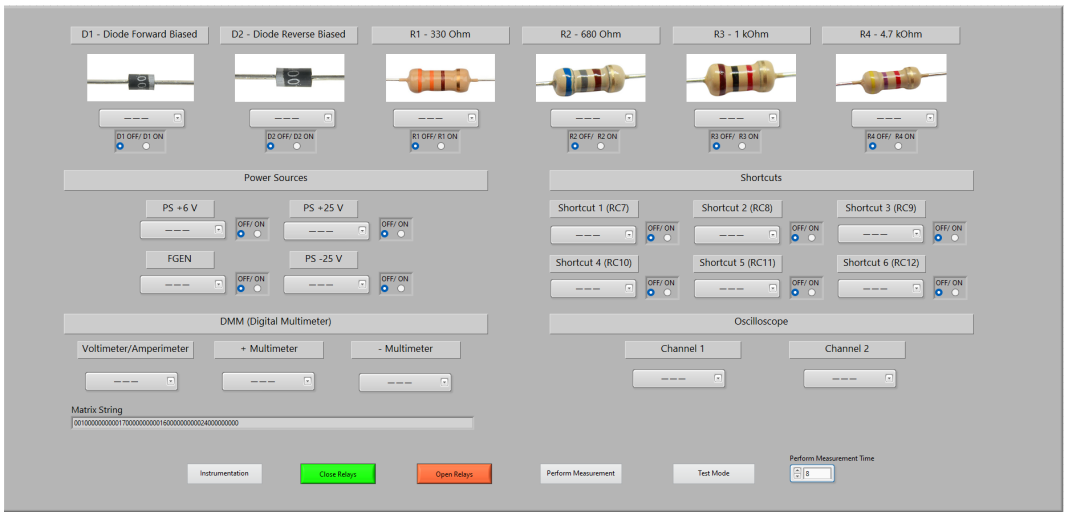


Figure 4.17: Teacher/Technician interface main page

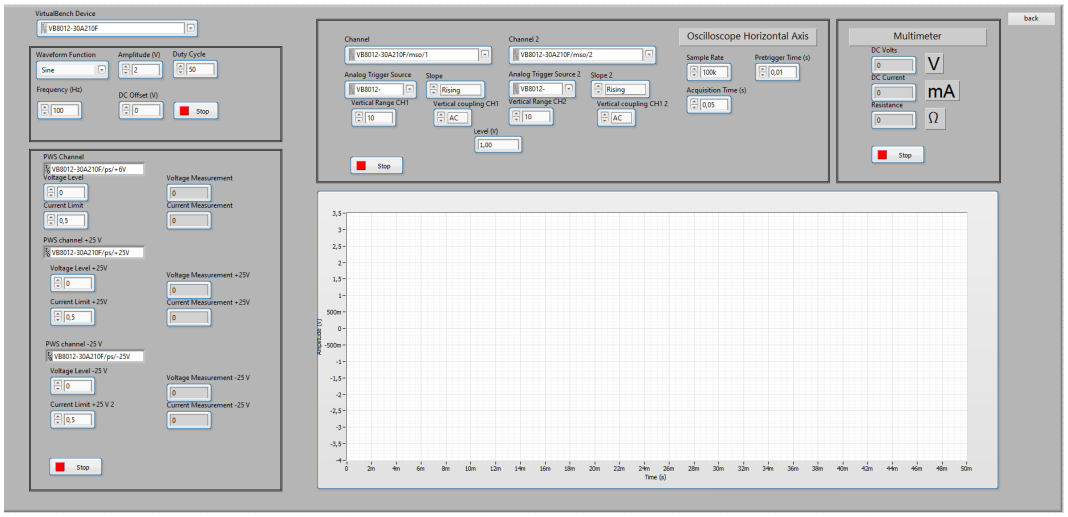


Figure 4.18: Teacher/Technician interface instruments page

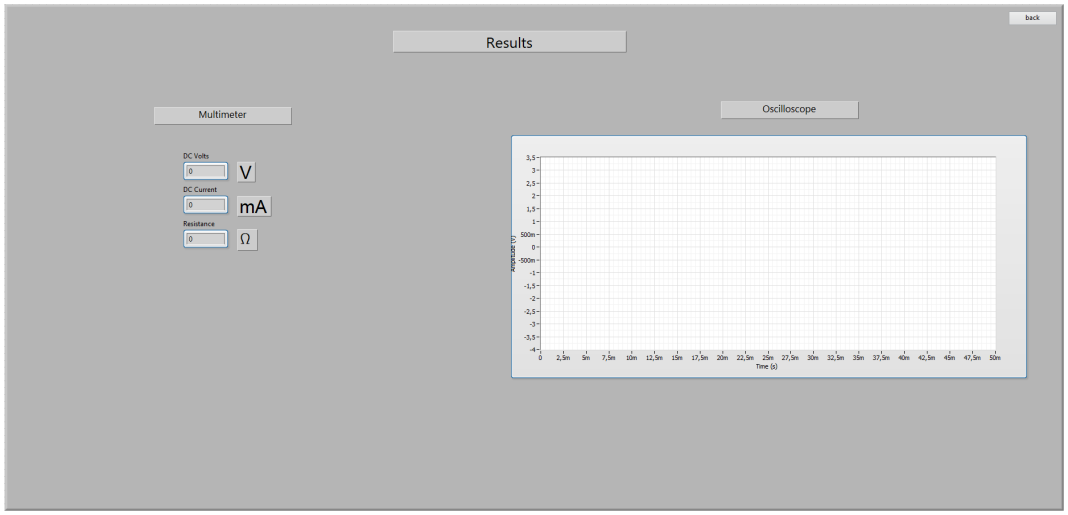


Figure 4.19: Teacher/Technician interface experiment results

One known limitation of relay based circuits is the limited number of physical actuation, which in remote laboratories, used many times each day leads to a relevant risk of relays breaking down, frequently without warning. That is, when there is an anomaly in the circuit created by a relay malfunction, the system has no way of checking which relay(s) is not working. Thus, the technician has the exhaustive task of reviewing the entire circuit to detect the error. To solve this limitation, a test mode was developed, accessed through the "Test Mode" button on the main page and shown in Figure 4.20.

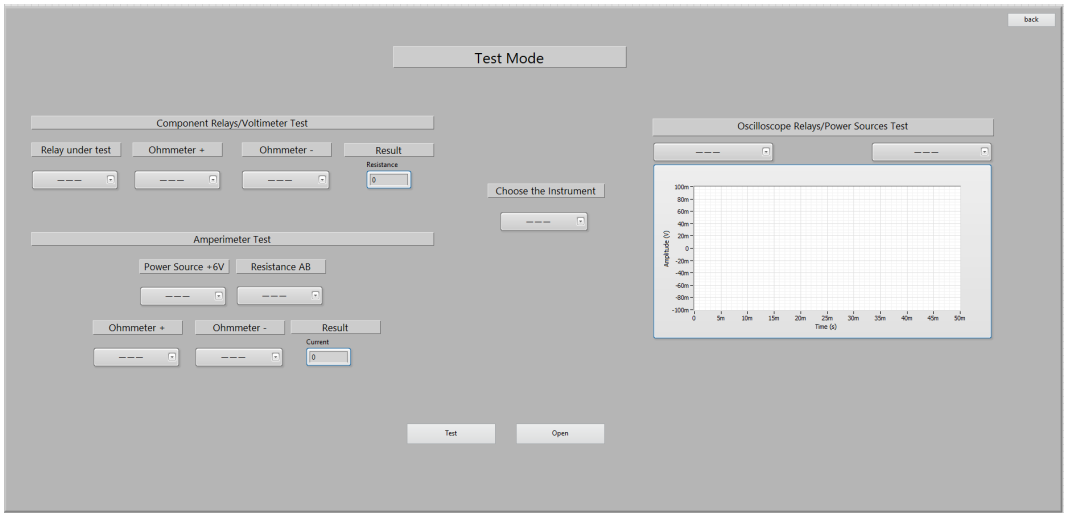


Figure 4.20: Teacher/Technician interface test mode page

4.2.2 Student Interface

The student interface features two pages. The main page, shown in Figure 4.21, allows the student to choose the circuit he wants to test considering the components physically placed between each node. In Figure 4.21 below "Select your circuit" the student chooses the component he wants to connect between each node through a dropdown that hypothesises the parts that are physically connected between each node. On top of each component's dropdown is written "Component X" for the student to relate to the image presented above to somehow have a graphical relationship with the circuit he is choosing.

This image represents a simpler circuit, with components in series and the student can change to another image that represents a more complex circuit with all the nodes by clicking on the "Complete Circuit" button. This interface, with this graphical part, limits the possible circuits to those that are intended to be validated with this thesis. For example, a student can't choose a component between nodes A and C, without the addition of another image for that specific experiment. In the future, it is intended that this interface has a more dynamic graphical part. The places where the instrumentation devices are located are chosen in the same way as in the teacher interface.

When the student presses the "Perform Measurement" button, the experiment is executed and the desired values are displayed on the right under "Measurement Results". In the case of real-time visualisation, it is done in a similar way as in the teacher interface by pressing the "Close Relays" button and then "Instrumentation" to visualise them on the page of Figure 4.22. After analysing them, the "Open Relays" button must be pressed.

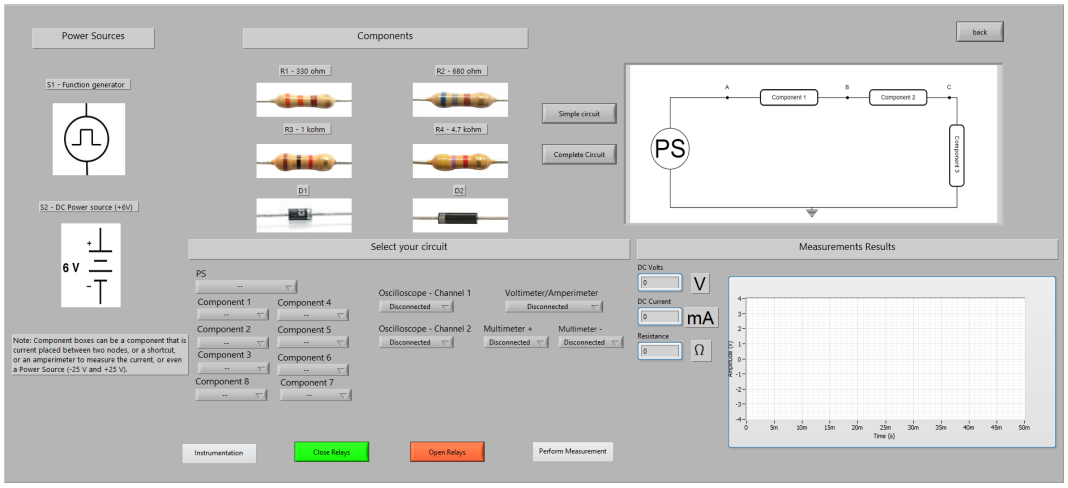


Figure 4.21: Student interface main page

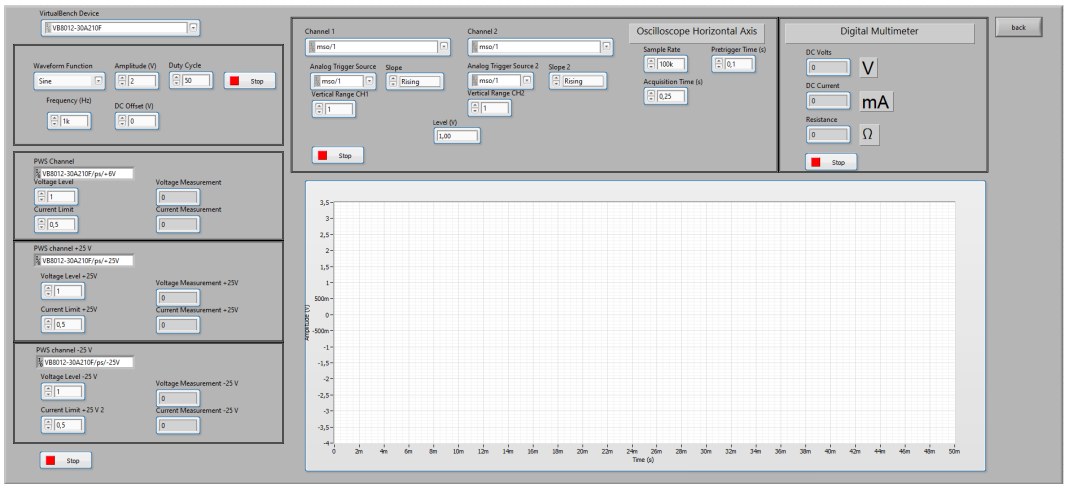


Figure 4.22: Student interface instrumentation page

4.2.3 Communication

Communication is carried out in one direction only. According to the user's choices, the program developed in LabVIEW performs the calculations to form a string that is sent to the Arduino, and the Arduino controls the relays. The format of the strings was developed according to the VISIR strings explained in chapter 2. However, in this case, the strings are all concatenated and only one string is sent as shown in Figure 4.23. Then, the Arduino code separates the strings considering each functionality.

Recall that the first four bytes of the string are associated with the relays of the components. The first byte "001" indicates the board number of the components and the remaining three are associated with the relays used. In this case, it was only necessary to use a maximum of two bytes for the relays of each functionality,

but the third byte with constant "000" was added for later expansion of the circuit and future use.

```
00101900000001700303200001600000000002403200000 0
```

Figure 4.23: String sent from LabVIEW

To construct the string in LabVIEW, it is necessary to associate with each connection (relay) a number associated with the pin to which it is connected that corresponds to a bit of the byte. For example, Figure 4.24 shows the properties of the dropdown button of the multimeter positive terminal. As there are seven relays per drive, only seven bits per word are used, the highest value being 64 in decimal numbers. The positive terminal of the multimeter is connected to node A by the label RV1, i.e. it was associated with the first bit counting from left to right (most significant bit) being the number 64. Node B is connected via the relay associated with the second most significant bit and has therefore been assigned the number 32.

In case the relay belongs to the second or third set of bytes, 64 was added for each byte. That is, if a number is between 1 and 64, it means that it is one of the first seven relays. If it is between 65 and 128, it means that it is related to the second byte, and if it is greater than 128, it is related to the third byte. For example, node C of the positive terminal of the multimeter has the number 96 associated with it, which means that it has been assigned the number 32 of the second byte (96-64).

In short, the LabVIEW code adds each feature selected by the dropdown button to a string, taking into account whether it refers to the components, the multimeter, etc. And, in the end, it concatenates the four strings giving rise to something similar visualised in Figure 4.23

Items	Values
— — —	0
A	64
B	32
C	96
D	80
E	72

Figure 4.24: Bit association in LabVIEW

4.2.4 Arduino software

The Arduino software aims to read the string sent by serial port by the LabVIEW program and divide it into four strings (components, oscilloscope, multimeter, power supplies). It then transforms the string into several bytes and associates each bit with the corresponding Arduino pin.

Figure 4.25 shows the flowchart of the loop() function. This is an Arduino function that runs in an infinite loop. Therefore, it was used to read a string from the serial port whenever it was not empty. Then, it opened all relays, in case the user did not open the relays after an experiment and, to avoid closing relays after another circuit was already closed to avoid malfunctions. Then the matrix string is split into four strings corresponding to each functionality. The four strings are stored in a previously defined vector that sends one by one to the ProcessString(String message) function. When it sends all four, the loop is closed and it checks again if there is any string in the serial port.

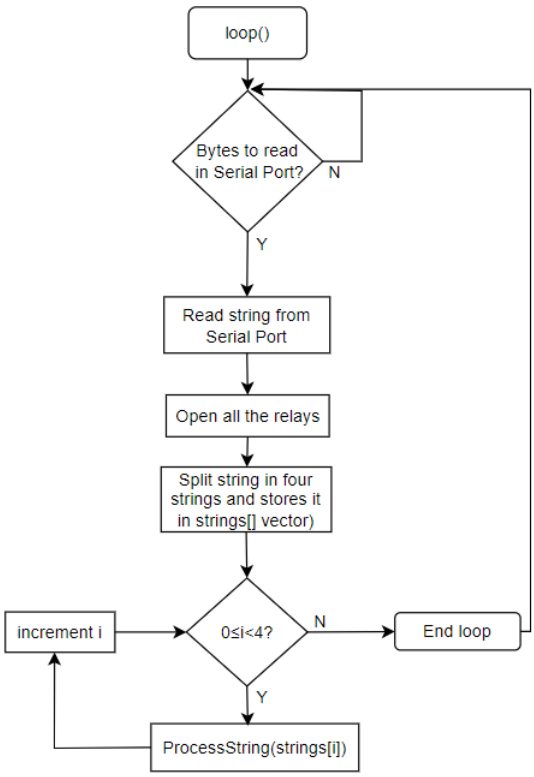


Figure 4.25: Arduino loop function

The ProcessString(String message) function, Figure 4.26 will receive the string sent by the loop() function. First, it checks the beginning of the string to see what string it is. If it starts with "000" it is the string to open all relays. If it starts with "001" it is the components string, with "016" it is the oscilloscope string, with "017" it is the multimeter string and with "024" it is the one for the sources. Then, convert the string into bytes and associate each bit to an Arduino pin defined for each function. A vector has been defined that stores the seven pins that are connected to each driver. In the case, for example, of components that use two drivers, two bytes of the string are converted and assigned one to each vector with the previously defined pins.

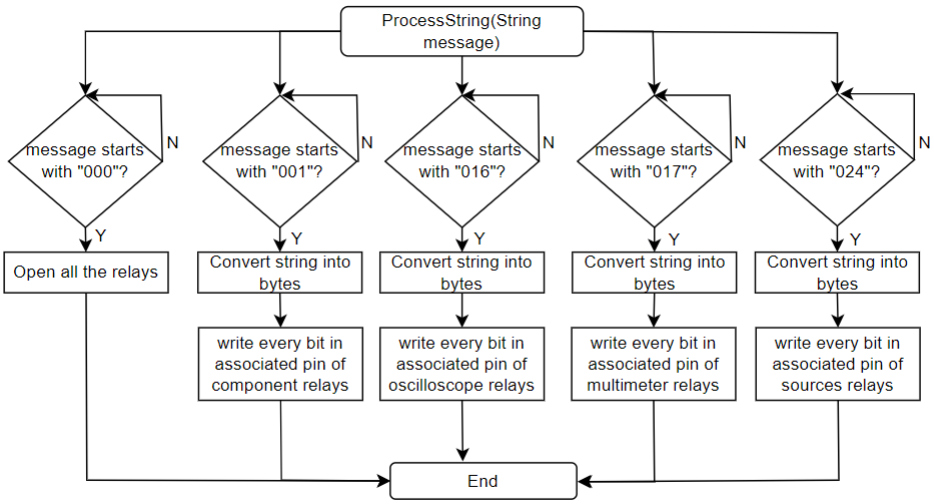


Figure 4.26: Arduino ProcessString function

The code for the functions shown is in appendix C for a better understanding.

4.2.5 LabVIEW Software

This subsection describes slices of code relating to circuit selection in the teacher/technician interface and in the student interface, relating to VirtualBench control and the test mode present in the teacher interface.

Teacher/Technician interface circuit choice

Figure 4.27 shows a code extract of the component string construction. To add, for example, diode 1 to the string, it is necessary for the user to indicate between which nodes this diode is located through the dropdown button (D1) and to indicate that it is ON. If these two conditions are met, the decimal number corresponding to diode 1 (64) is added to an array. In this image, only the code for diode 1 is represented. The same process is performed for the remaining components, but the number that is added to the array depends on the pin to which each component is associated. Then, all the values present in the array are summed and some code blocks are presented in front, responsible for ensuring that the decimal number resulting from the sum of the connected components always maintains the three digits, adding zeros to the left if necessary. In the end, the value is added to the string concatenation block, in the second place, since the first one concerns the string "001" of component string identification. For the third place, the same code described here is carried out, but for the components that are connected to the second relay driver. The last one is added "000" since only two drivers are being used and, in the future, more can be added to increase the number of relays.

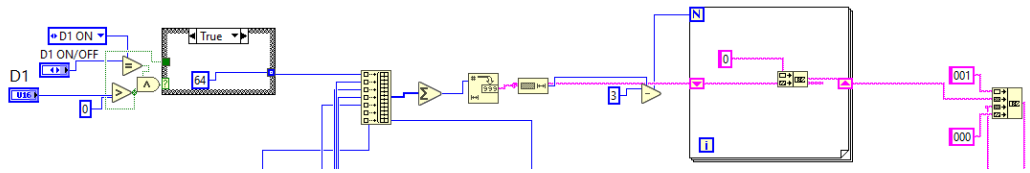


Figure 4.27: Components string construction code

Figure 4.28 shows the code for checking allowed circuits. The circuits.txt file is opened and through a block called "Match Pattern" the final string is compared with all the contents of the file and, if it matches, the match pattern is returned. It is intended that this check is carried out when the user clicks on the "Close Relays" button. If the user clicks on this button and if any pattern is found (the size of the match string is greater than zero), a message is displayed to the user indicating that he has defined a correct circuit and the string is sent through the serial port to the Arduino. If the user presses the button and the circuit is not defined in the file, nothing is sent by the Arduino and an "incorrect circuit" message is displayed. If the user presses the "Open Relays" button, no verification is required and a string of zeros is sent to the Arduino.

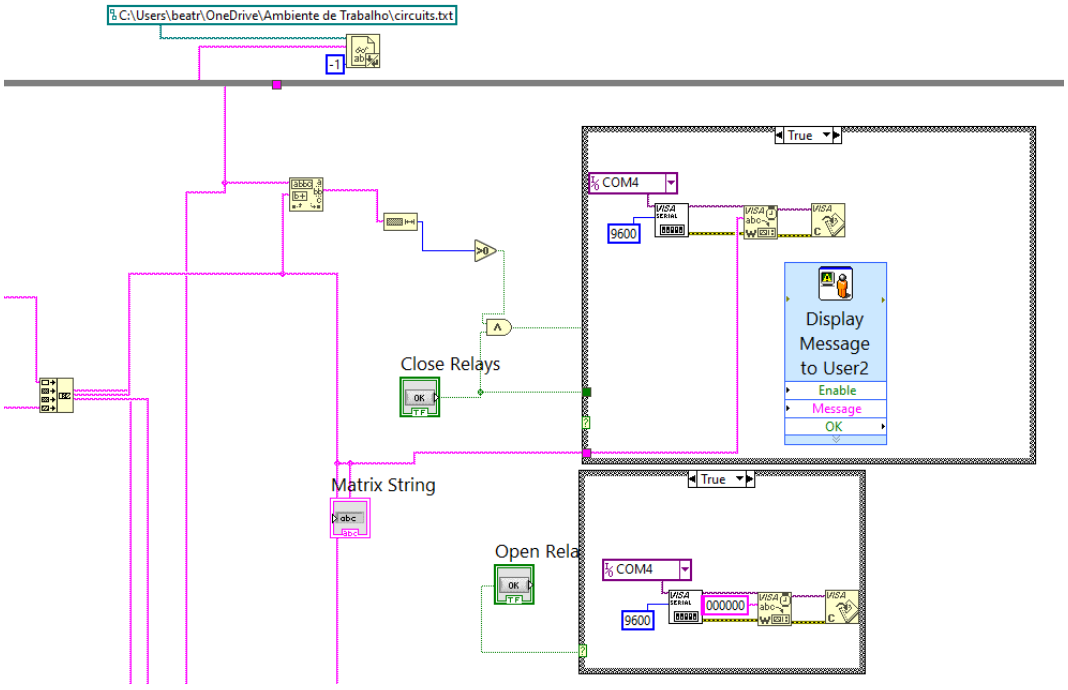


Figure 4.28: Circuit check code

Figure 4.29 shows the code extract for the "Perform Measurement" functionality. When the user clicks on this button, the same file check is performed as explained above. If the chosen circuit is valid, a set of functions are executed within a flat sequence structure. This structure causes the functionalities to be executed in

sequence from left to right. That is, when the user clicks on the "Perform Measurement" button, the string to close the relays is sent to the Arduino, then the time set by the teacher is waited and the values are recorded. Finally, a string is sent to open the relays and automatically switches to the results visualisation page.

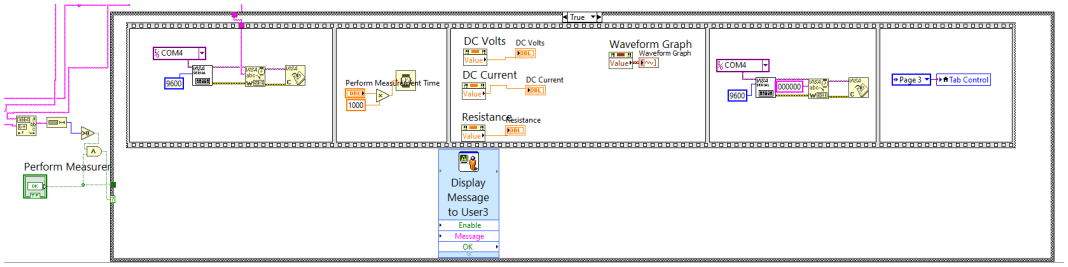


Figure 4.29: Perform measurement code

Student interface circuit choice

The way the component string is constructed in the student interface is slightly different from the teacher interface. As shown above, in the student interface there are dropdown buttons containing the possible components between each node. The student chooses the ones they want to connect and, as shown in the communication section, each component is associated with a number. Figure 4.30 shows part of the string construction code. This consists of checking which components are less than 64 and which are greater (corresponding to the first byte and the second byte, respectively), then adding all those corresponding to the first byte to a vector and all those corresponding to the second byte to another vector, and then adding up all the numbers in the vector to obtain the final value for each byte. The string is then constructed by concatenating all the bytes in the same way as in the teacher interface.

The choice of instruments and sources and their configuration is carried out in a similar way to the teacher interface.

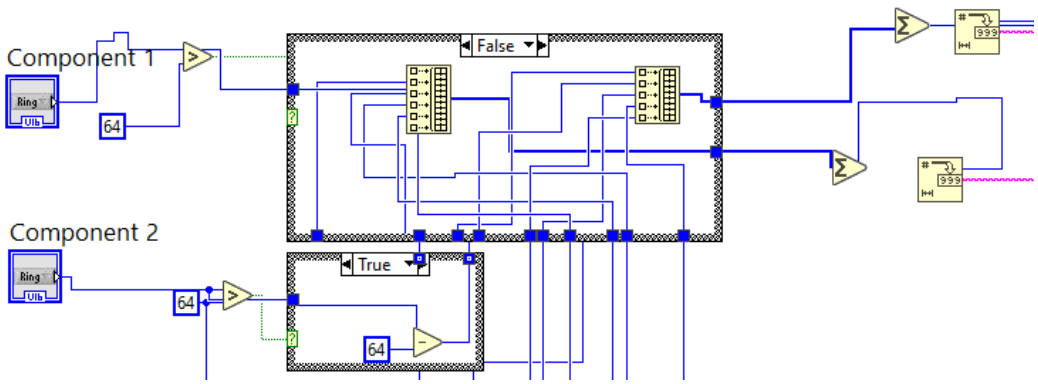


Figure 4.30: Student interface components string

Figure D.3 in the appendix D shows the total code for constructing the string that is sent to the Arduino in the student interface. The rest of the code, relating to navigating between pages, sending the string to the Arduino and controlling VirtualBench, is done in the same way for the teacher/technician interface and is shown in full in Figures D.1, D.4 and D.5, from the appendix D.

VirtualBench control

Figure 4.31 shows the code for FGEN control. Through the predefined blocks for VirtualBench, the FGEN is accessed and the waveform parameters (waveform function, frequency, duty cycle, offset and amplitude) are defined and then applied to the instrument. The rest of the code specifies the limits for the inputs so as not to exceed the range of values allowed by the device. For example, the FGEN allows a maximum amplitude of 24 V and a maximum positive voltage of 12 V. If the user sets 24 V amplitude and 1 V offset, it will exceed the FGEN limits. With the developed code, none of the limits are exceeded.

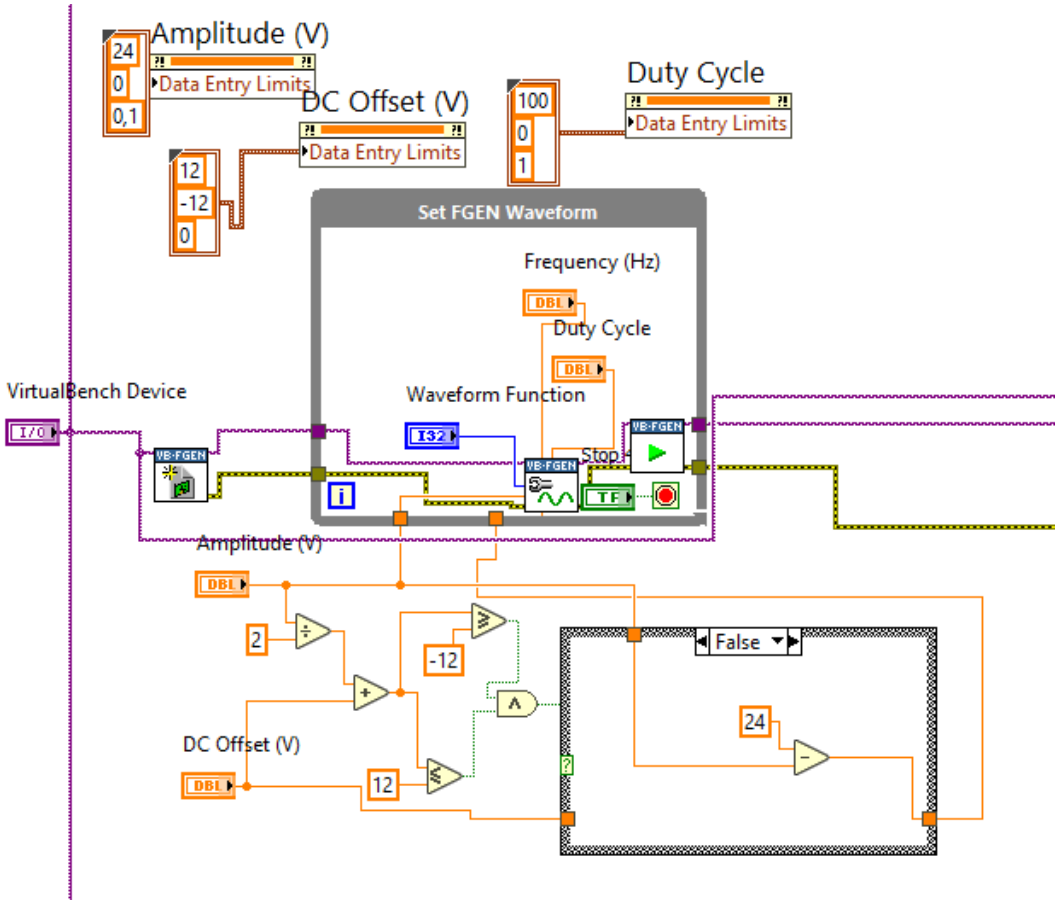


Figure 4.31: FGEN code

Figure 4.32 shows the code where the DC power supplies are configured. Like all other instruments, it is necessary to access the sources through a predefined

block and then use a block for each source to set its parameters (desired voltage and maximum current). Then, the current value of the voltage and current of each source is measured since the device performs these measurements internally. This way, it is possible to check that the sources are working correctly. As in FGEN, the maximum and minimum input values are set so as not to allow the user to exceed the limits of the device.

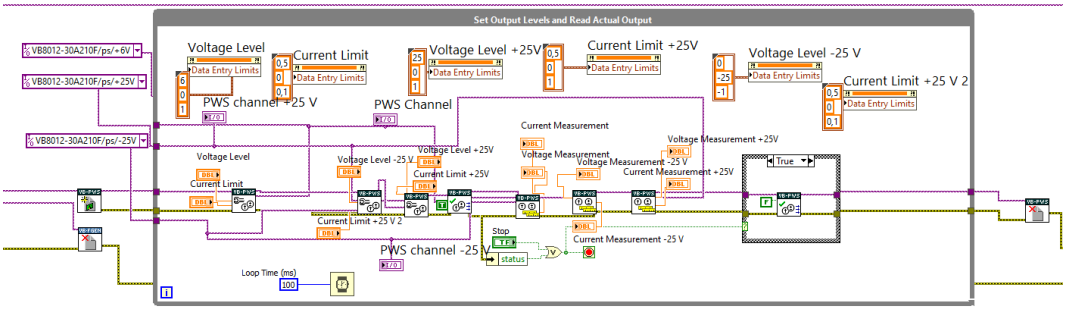


Figure 4.32: DC Power Supplies code

Figure 4.33 shows the code for the oscilloscope's characteristics. The user needs to define the vertical characteristics for each display channel and then the time-related characteristics (horizontal axis). The "Run" block is then executed to take the oscilloscope measurements with the characteristics indicated and, at the end, the graph with the results obtained is displayed.

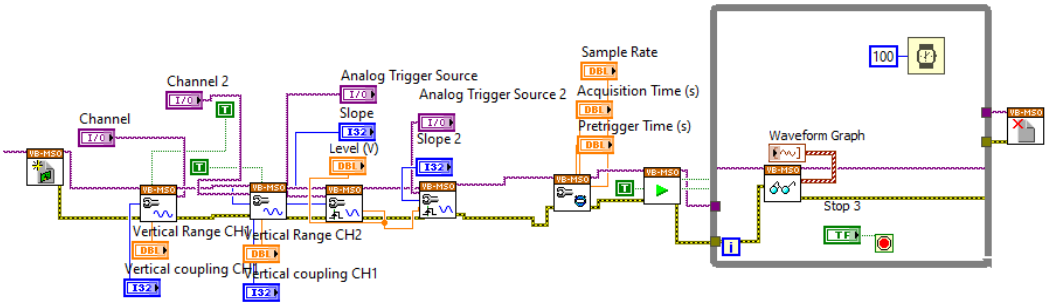


Figure 4.33: Oscilloscope code

Figure 4.34 shows the DMM configuration code. For each measurement to be made, it is necessary to define which quantity is to be measured (voltage, current or resistance). Then, there is a block to perform the measurement and display the value on an indicator present on the front panel. Since the measured value is in amperes and the current values of most electronic circuits are in the order of milliamperes, this value has been multiplied by one thousand.

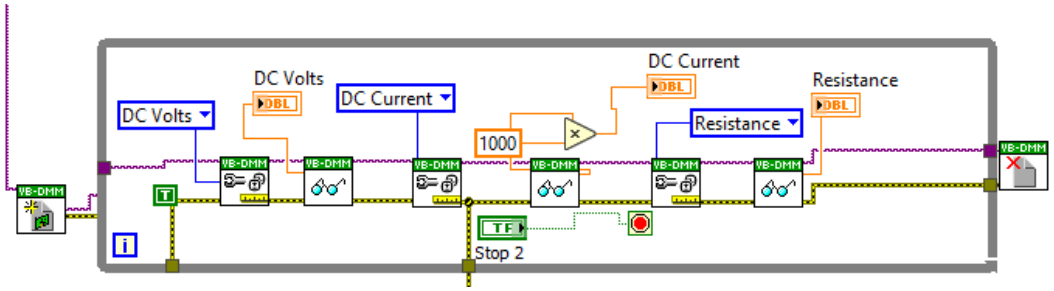


Figure 4.34: Digital multimeter code

Test mode

Figure 4.35 shows a code extract from the relay test mode, which refers to testing the components together with the voltmeter. Firstly, it is checked whether the multimeter is being selected as the voltmeter, and if so, the vectors for positioning the voltmeter terminals in the component test are created. If not, it is checked whether it is working as an ammeter and the vectors relating to the multimeter in the ammeter test are built. Note that two vectors are created which give rise to the two bytes referring to the number of relays used for the multimeter connections.

The thinking behind the test mode code is similar to the normal construction of circuits, with the difference that the same instrument is used in several tests, as is the case, for example, with the choice of multimeter terminals twice (for testing the components and the ammeter), which makes the code slightly more complex.

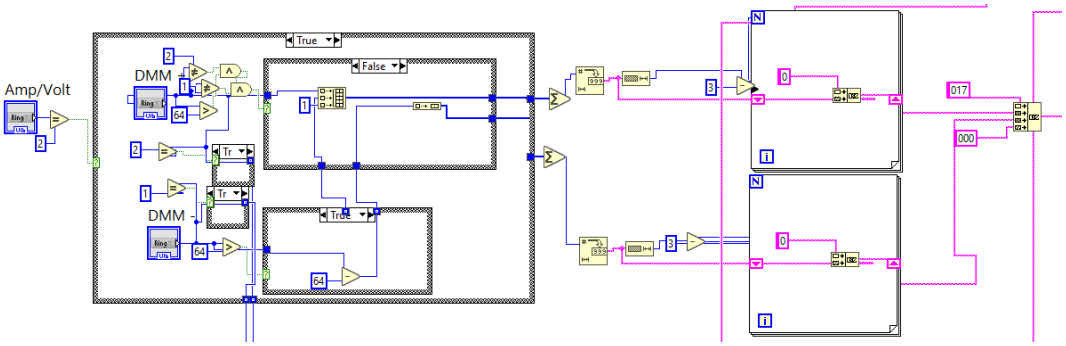


Figure 4.35: Test multimeter relays code

The complete code can be found in the appendix D for a better understanding of the relationship between the several parts presented.

4.2.6 Valid circuits file

The Circuits.txt file is responsible for storing the strings that indicate the allowed circuits. In other words, it serves to limit the circuits chosen by the user so that the equipment cannot be damaged.

When the user clicks on the button to run the experiment, it is checked if the string referring to the designed circuit is present in this file. If it is, a "Correct Circuit" message is displayed and the experiment is run. If not, the message "Incorrect Circuit" is displayed and the relays are not closed.

Figure 4.36 shows an extract of a circuits.txt file. Before each string, a comment is displayed to indicate the elements that make up the circuit. For example, the first line indicates that the +6 V supply is connected, a shortcut between A and B and another shortcut between B and C, the 4.7 kΩ resistor is connected between C and 0 and the voltage between C and 0 is being measured.

```
+6V SHORTCUT AB BC R_4k7 C0 Volt C0
00100306400001700303200001600000000024032000000
SHORTCUT AB BC R_4k7 C0 Ohm C0
00100306400001700303200001600000000024000000000
+6V R_680 AB SHORTCUT BC R_1k C0 Volt C0
00102100400001700303200001600000000024032000000
+6V R_680 AB SHORTCUT BC R_1k C0 Volt AB
00102100400001706900000001600000000024032000000
R_680 AB SHORTCUT BC R_1k C0 Ohm AB
00102100400001706900000001600000000024000000000
R_680 AB SHORTCUT BC R_1k C0 Ohm C0
00102100400001700303200001600000000024000000000
R_680 AB R_680 AB SHORTCUT BC C0 Ohm AB
00101703200001706900000001600000000024000000000
FGEN D1 AB SHORTCUT BC R_1k C0 Chn1_A Chn2_B
00106900400001700000000001607200000024064000000
FGEN D1 AB SHORTCUT BC R_1k C0 Chn1_B
00106900400001700000000001603200000024064000000
```

Figure 4.36: Circuits.txt file extract

Chapter 5

Results

This chapter presents some of the results obtained. As various tests were carried out for several types of experiments, some examples of simulations obtained and some examples of results obtained when the experiment was carried out in practice will be presented. The simulations include an example of current measurement and an example of wave visualisation on the oscilloscope. For the rest of the simulations obtained, the operation of the relays is the same depending on the desired experiment. In the case of empirical measurements, an example measurement is given for each experiment described in chapter 3.

5.1 Simulations

For all the experiments and tests carried out, a simulation was carried out beforehand in Proteus. This is an important step before practical testing to detect errors and avoid damaging the equipment. For all the simulations, the values obtained were equal to the theoretical ones and similar to those obtained in practice.

Figure 5.1 shows a simulation of a current measurement. The +6 V source relay is connected with the 680 Ω, 1 kΩ and 330 Ω resistors. The latter resistor is connected between nodes C and D via labels. Whenever the teacher physically changes the connections of a component, the nodes to which it is connected must be changed in the simulation. In the 330 Ω resistor branch it's connected the +25 V source, regulated to +5 V. The shortcut between B and C is switched off so that

the ammeter can be placed between these two nodes and the current in that branch is measured.

Figure 5.2 shows the ammeter's connections and the current value of +2.41 mA, which is the expected value for these conditions.

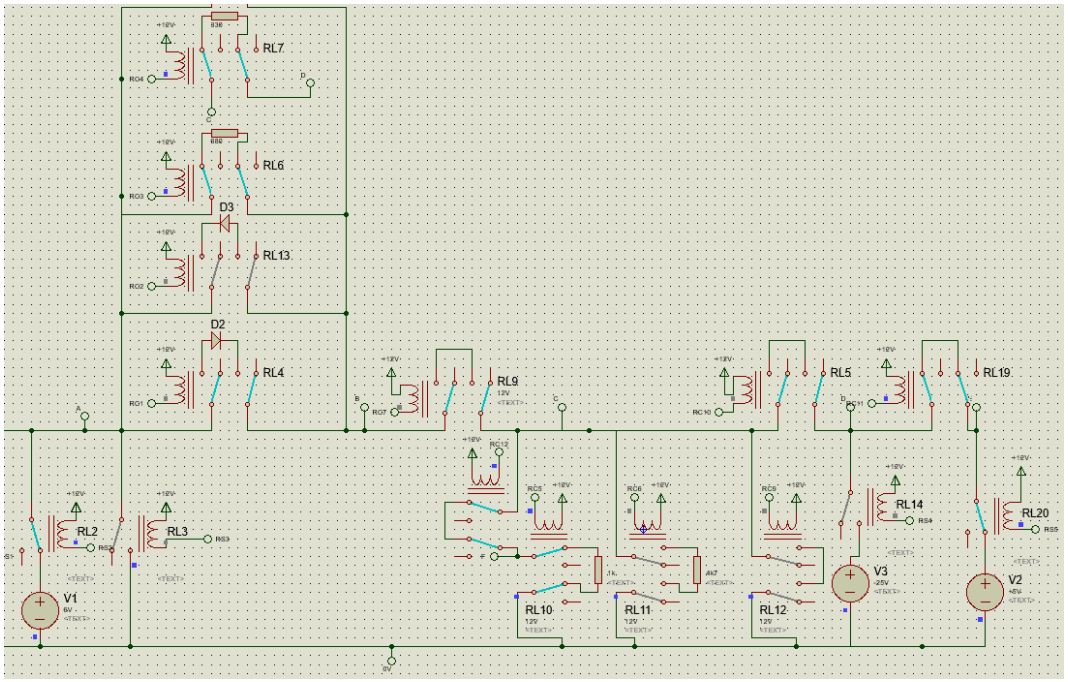


Figure 5.1: Current measurement circuit simulation

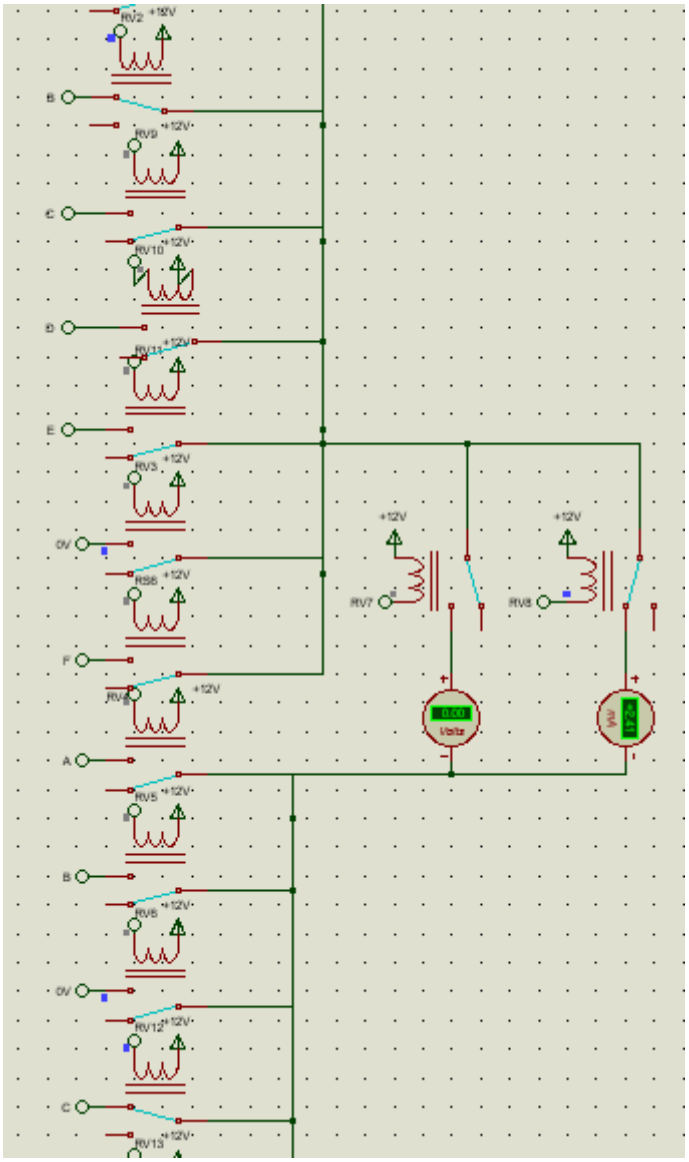


Figure 5.2: Current measurement ammeter result simulation

Figure 5.3 shows an example of a simulation of a half-wave rectifier. The simulator's oscilloscope can display up to four channels and allows the waveform characteristics to be changed, as in real oscilloscopes and the remote laboratory developed.

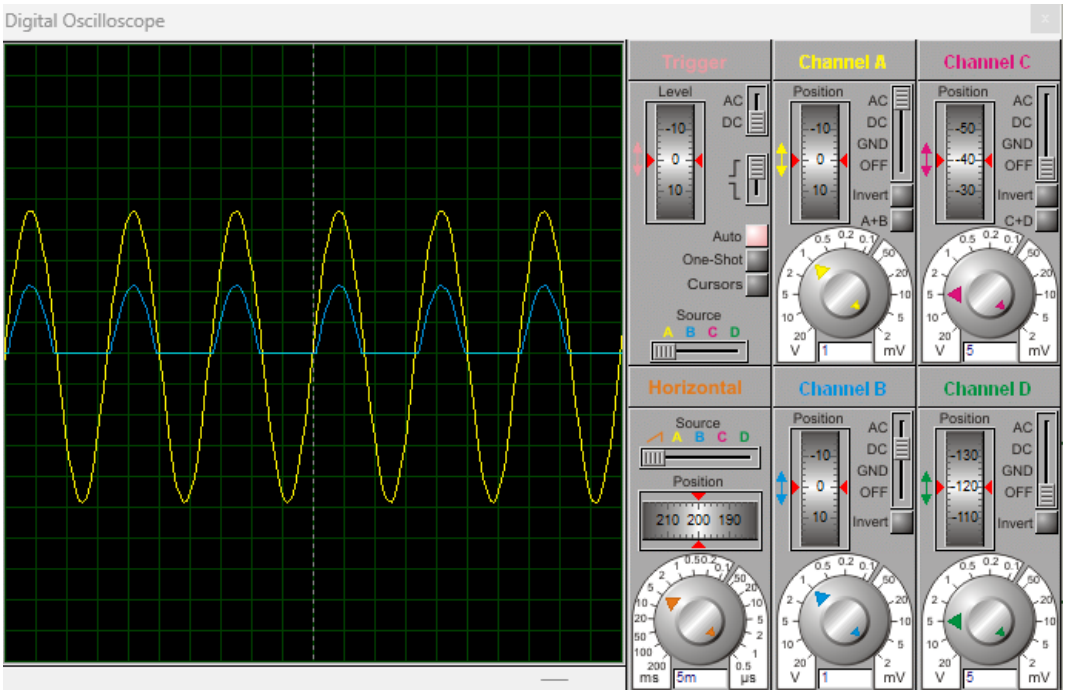


Figure 5.3: Half-wave rectifier simulation

5.2 Teacher/Technician Interface

In the teacher/technician interface, you can measure various circuits and test relays. This section will present results from both modes.

5.2.1 Testing Circuits

Figure 5.4 illustrates how to choose the desired circuit in the teacher/technician interface. It is necessary to define between which nodes each component is placed through a dropdown that makes it possible to set an element between any two nodes of the circuit. Then, it is necessary to define whether the component is ON or OFF. This methodology is also valid for adding sources and shortcuts to the circuit. In the case of the multimeter, it is necessary to indicate whether it works as a voltmeter/ohmmeter or as an ammeter and then indicate which node it's intended to connect the positive and negative terminals to. In the case of the oscilloscope, it is necessary to indicate in which node it's necessary to connect channels 1 and 2.

This interface allows all possible combinations of connections since it is the teacher/technician who will change the physical connections of the components, being able to connect between any two nodes. Note that the shortcuts have an indication in brackets referring to the label assigned in Proteus to which the user of this interface has access. In this way, it is possible to identify the shortcuts.

From Figure 5.4, it can be seen that a circuit is being implemented in which Ohm's Law can be verified. The resistance of 680 Ω between nodes A and B is selected as ON.

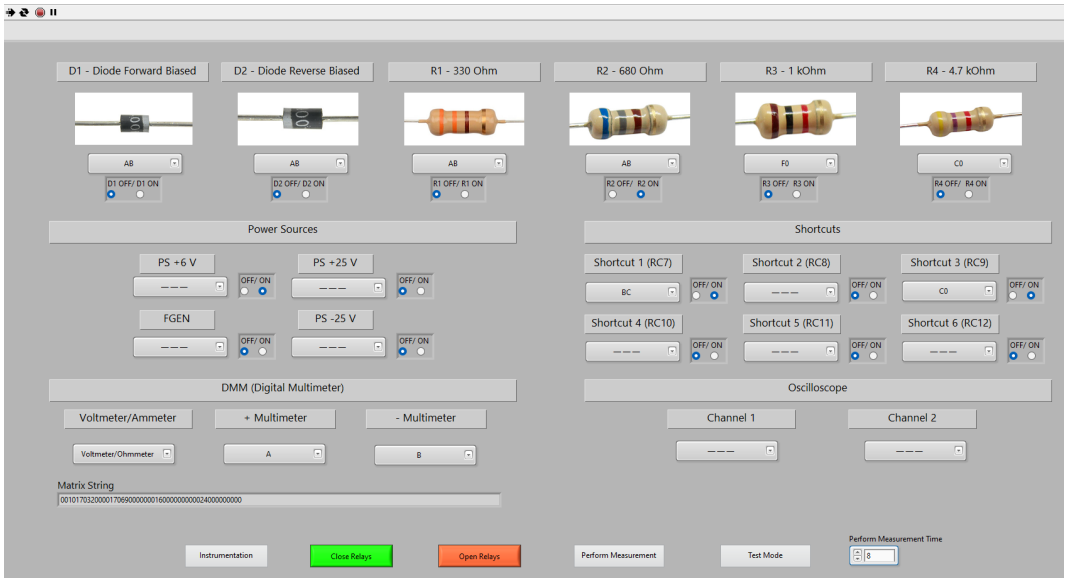


Figure 5.4: Ohm's law circuit main page

Figure 5.5 shows the resistance measurement value of 680 Ω resistor on the instruments page. The actual measurement value is 701.47 Ω.

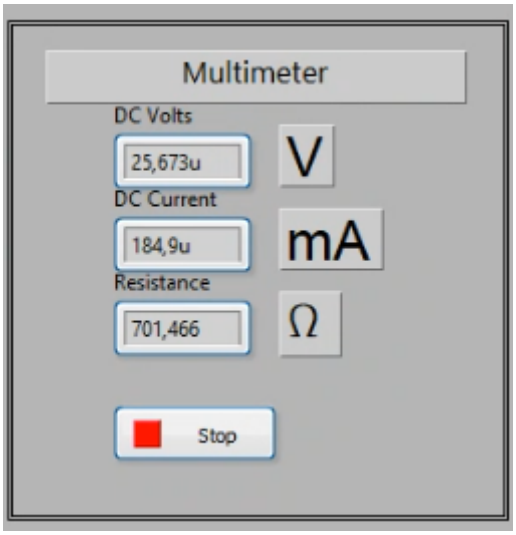


Figure 5.5: Ohm's law circuit resistance measurement

An example of voltage measurement in a voltage divider circuit is shown in Figure 5.6. In this example, 680 Ω and 1 kΩ resistors are being used in series. As the first resistor is located between nodes A and B and the second between nodes C and 0, it is necessary to turn on the shortcut between nodes B and C so that

resistors are connected. Then, the +6 V power supply it's also switched on between nodes A and 0 for the circuit to be complete.

The main purpose of this circuit is for the user to make voltage measurements on the resistors. In this example, the voltmeter is connected between terminals A and B to measure the voltage drop across the 680 Ω resistor. Figure 5.7 shows the actual value of approximately 2.40 V. The theoretical value of this quantity under these conditions is 2.43 V.

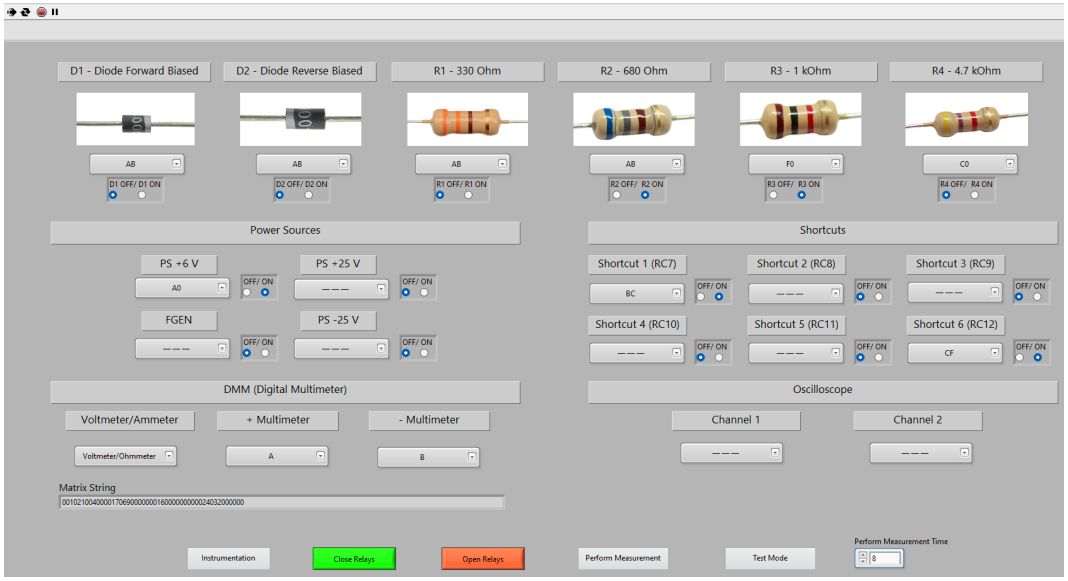


Figure 5.6: Voltage divider circuit main page

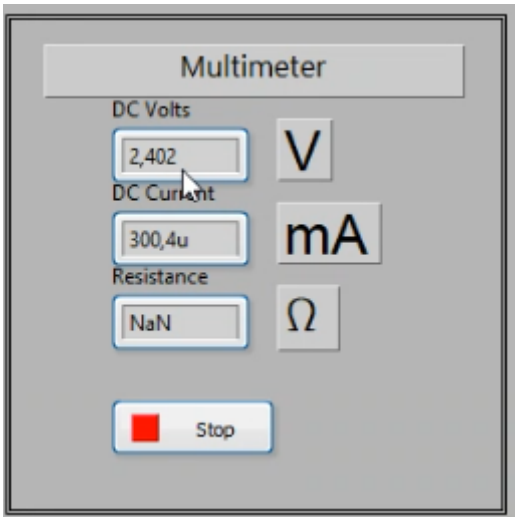


Figure 5.7: Voltage measurement between A and B nodes

Figure 5.8 shows the circuit necessary to carry out the experiment of a half-wave rectifier. The components connected are a diode, directly polarised, and a 1 kΩ resistor connected in series with the diode. For this circuit, it is necessary to

connect the FGEN after choosing the wave characteristics on the instruments page. The connected shortcuts have the functionality to place the components in series considering their location in the circuit. As is characteristic of a half-wave circuit, channel 1 of the oscilloscope is placed on node A to visualise the input wave and channel 2 is placed on node B to visualise the output wave.

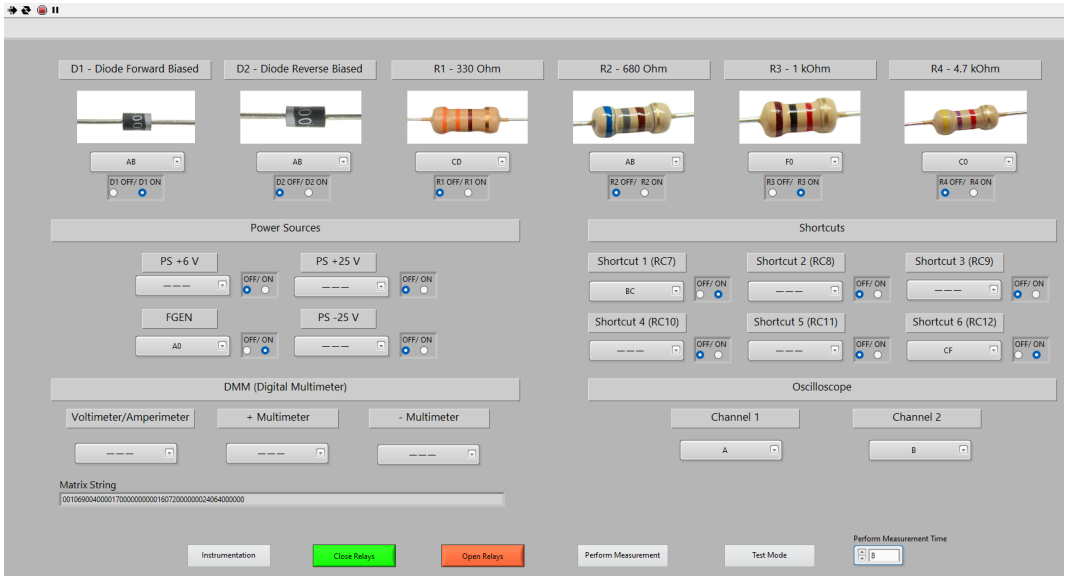


Figure 5.8: Half-wave rectifier main page

Figure 5.9 shows the wave for channel 1 in blue and the wave for channel 2 in red. In addition, it is also possible to check the characteristics defined for the input wave, as well as the characteristics of the horizontal and vertical axis of the oscilloscope.

This image validates the circuit study made previously since the diode lets the positive half-cycle pass while blocking the negative one.

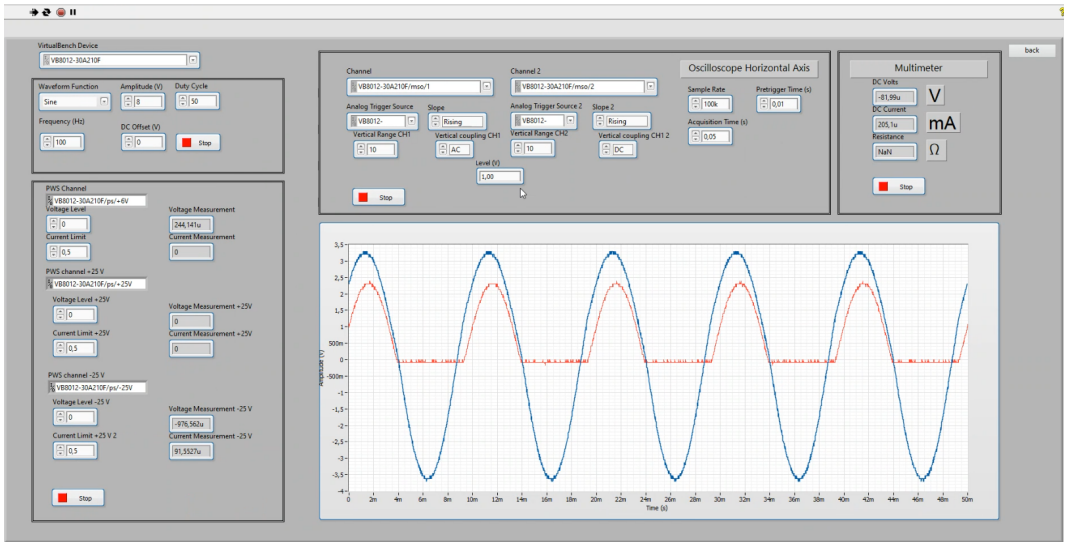


Figure 5.9: Half-wave rectifier instrumentation page

Figure 5.10 shows an example of when the "Perform Measurement" button is pressed and the experiment is executed, and then the results are displayed on the page shown.

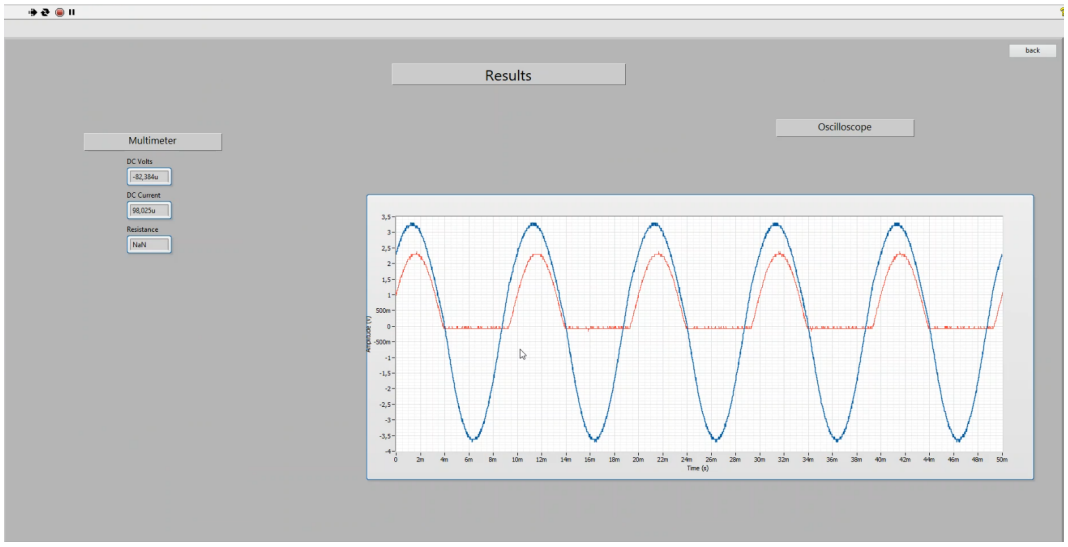


Figure 5.10: Half-wave rectifier "Perform Measurement" page

An example of a current divider circuit is shown in Figure 5.11. It consists of a 680 Ω resistor in series with parallel 1 kΩ and 330 Ω resistors. Each branch of this circuit has a shortcut so that it is possible to measure the current in all branches by replacing it with the ammeter. In this example, the current in the 330 Ω resistor branch is measured, i.e. the shortcut between nodes C and D is replaced by the ammeter.

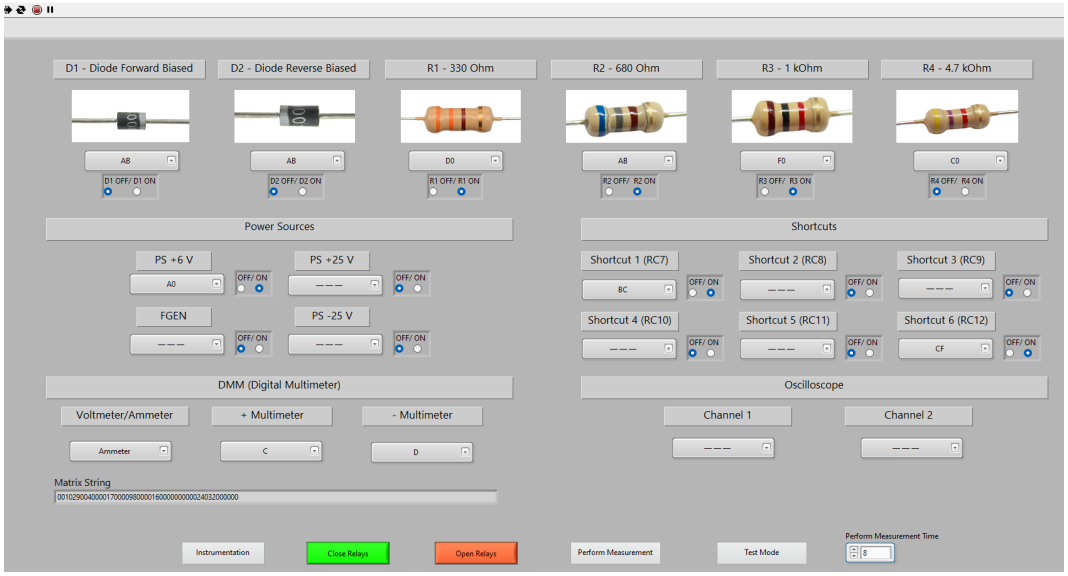


Figure 5.11: Current divider circuit main page

Figure 5.12 shows the actual current value of this branch, which is approximately 4.58 mA. The theoretical value is 4.86 mA.

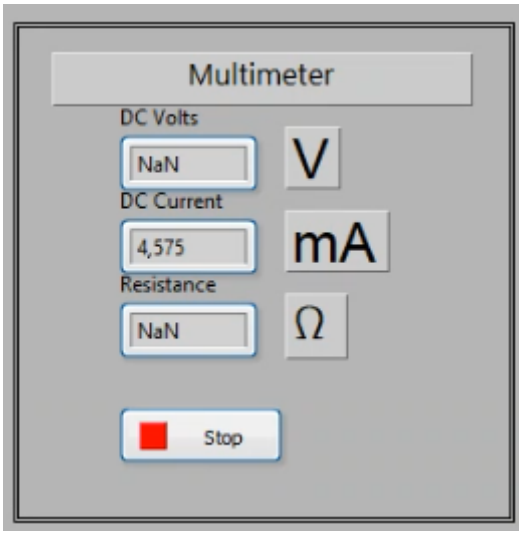


Figure 5.12: Current measurement between nodes C and D

Figure 5.13 shows a circuit with three branches and two power supplies DC, representing a circuit in which Kirchhoff's Laws apply. It can be seen that a power supply is located between node A and 0 (regulated to +6 V) and another between node E and 0 (regulated to +5 V). The 330 Ω, 680 Ω and 1kΩ resistors and the shortcuts that connect the three are being used. In Figure 5.13, it's possible to see that it's intended to measure the voltage on the 680 Ω resistor (between A and B) and the result is shown in Figure 5.14 of approximately 1.7 V. The theoretical value is around 1.64 V. To measure the remaining voltages, it is only necessary to change the nodes where the voltmeter is connected and run the experiment again.

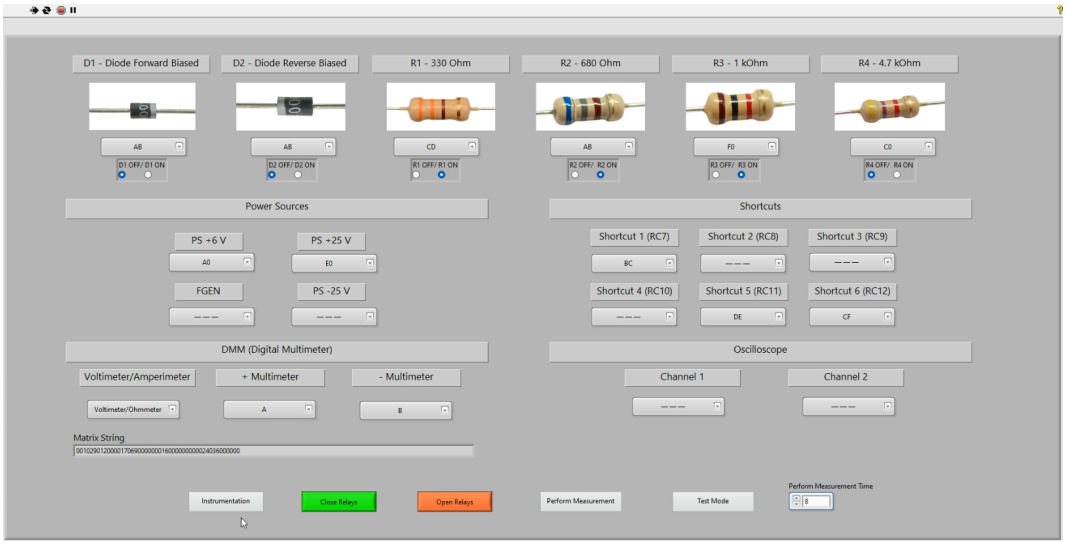


Figure 5.13: Kirchhoff Laws main page

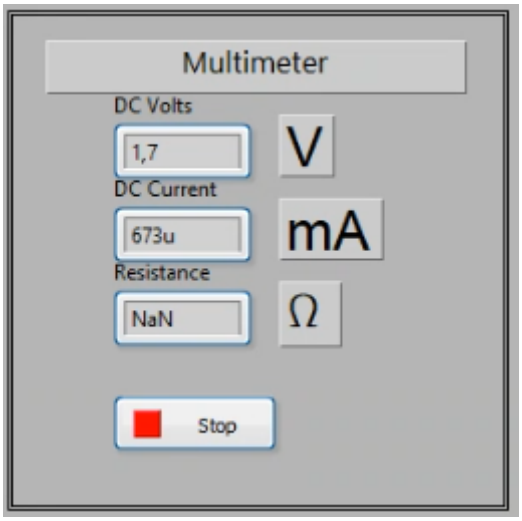


Figure 5.14: Kirchhoff Laws voltage between A and B

To measure the current, it is necessary to change the choice from voltmeter to ammeter, choose between which nodes the ammeter will be positioned and disconnect the shortcut placed there. Figure 5.15 shows the measured result for the current between node B and C (in the branch of the +6 V source and the 680 Ω resistance), of 2.44 mA. The theoretical value is 2.41 mA.

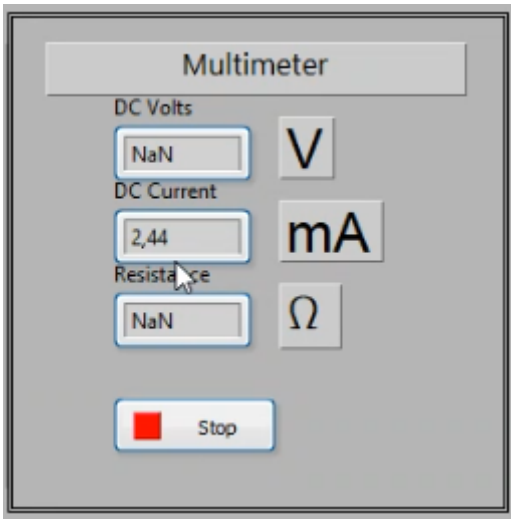


Figure 5.15: Kirchhoff Laws current between B and C

5.2.2 Testing Relays

As mentioned earlier, a relay test mode for each functionality was developed to enable debugging of errors caused by malfunctioning relays.

To test the oscilloscope and power supplies, the user must choose which channel(s) he wants to visualise and which node(s) he wants to connect them to, considering the source he also wants to test and where it is physically located. For

example, Figure 5.16 shows a graph with the waveform of channel 1 of the oscilloscope when connecting this channel to FGEN. Thus, it is possible to verify that the relays related to channel 1 of the oscilloscope and the FGEN are working correctly.

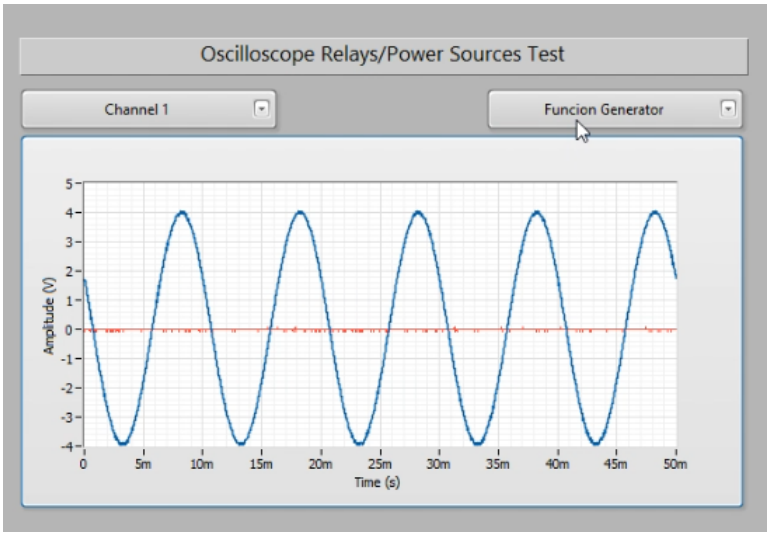


Figure 5.16: Oscilloscope and FGEN relays testing

In Figure 5.17 the oscilloscope is also being tested together with the +6 V power supply. To perform this test, it must be changed the "Vertical coupling" to the "DC" option on the instruments page. In this way, it is possible to check the correct operation of the relays of the power supply and the oscilloscope when a DC wave is present at +6 V. To check the remaining sources, it is only necessary to select the power source to be tested in the dropdown button.

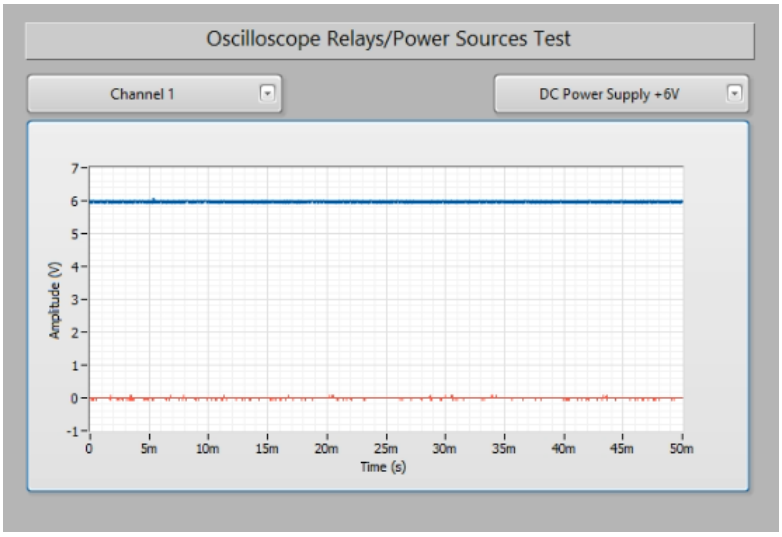


Figure 5.17: Oscilloscope and DC Power Supplies relays testing

The relays of the components (and shortcuts) are tested together with those of the voltmeter. It is necessary to choose the component you want to test (by the labels of the Proteus schematic) and the nodes to which it is connected and the voltmeter will be switched on. This is working as an ohmmeter. If the component is a resistor, it will measure its resistance, as shown in Figure 5.18 the measurement of a resistor of theoretical value 330 Ω. Diodes do not have a specific resistance, but they have a high resistance which is different from being open circuit and thus it is possible to test these relays, Figure 5.19. In the case of shortcuts the measured value will be close to zero (ideally it is zero). For these measurements to be made, it is also necessary to indicate that the multimeter is working as a voltmeter/ohmmeter in order to close the relay which chooses the positive terminal of the voltmeter (instead of the ammeter).

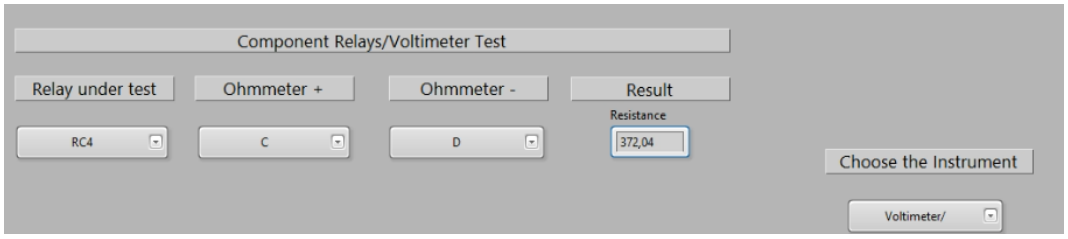


Figure 5.18: Resistors relays testing

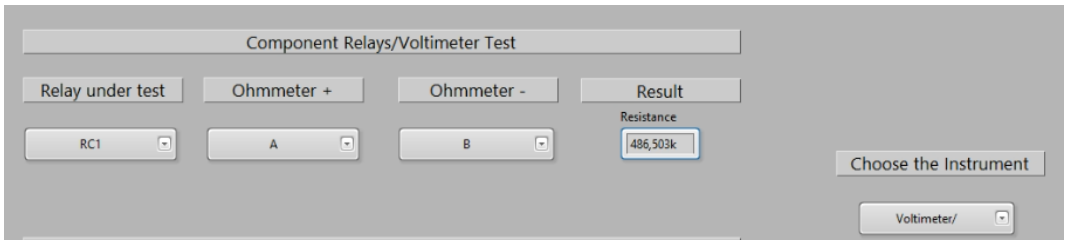


Figure 5.19: Diodes relays testing

The only way to test the ammeter relays is to use a source and a load in series and measure the current. As shown in Figure 5.20, the voltage source of +6 V is being used, which is between A and 0, then a resistor between A and B is chosen and the ammeter is positioned between B and 0. Note that in test mode there is no circuit verification to give more freedom to the user since this user knows the developed circuit (including the connections of the components to the nodes that he makes himself). Only extra care is needed not to have anything connected from other functionalities tests when using power supplies.

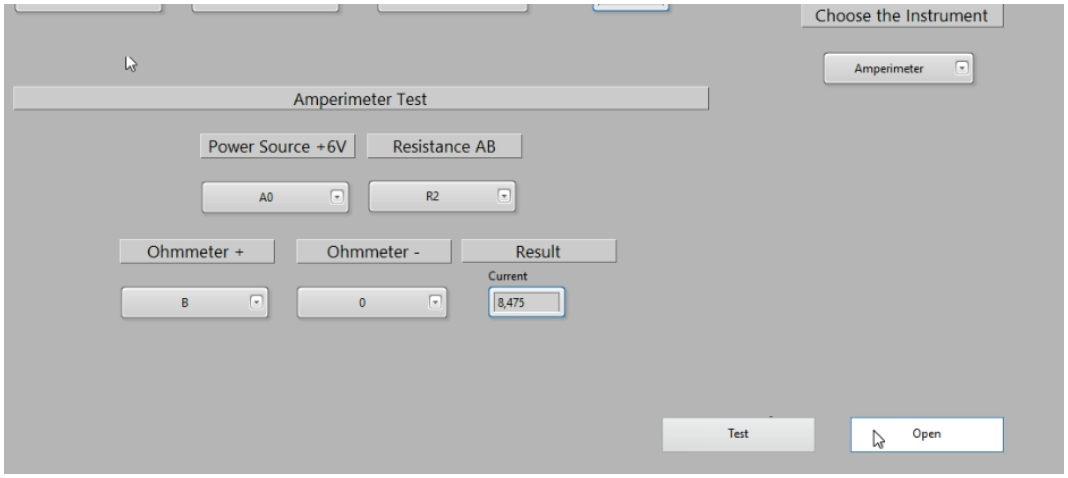


Figure 5.20: Ammeter relays testing

5.3 Student Interface

Figure 5.21 shows how to perform an experiment of a half-wave rectifier in the student interface. In this interface, it is important that there is a graphical part of the circuit so that the student better understands the experiment he is performing.

In this interface, there are two circuit designs, for experiments with series components and for three-branch experiments. These two drawings allow the realisation of all the circuits to be tested. However, it limits the range of the experiments, for example, if the student wants to place a component between A and C, it is not possible through the existing drawings. For other types of experiments, it will be needed to add new drawings.

For example, for a half-wave rectifier, the user has to select the image "Simple circuit" and choose through the dropdown buttons the components (shortcuts, or power supply) that he wants to put in place of the boxes present in the image. And then choose where to place the measuring instrument he wants to use. If the user wants to visualise the change of values in real time, he should press the "Close Relays" button and then the "Instrumentation" button and will be redirected to the instrumentation page just like in the teacher interface. It is also on this page that the student should change the instrumentation characteristics. If the student presses the "Perform Measurement" button, the experiment will be executed and the values will be displayed on the main page, as can be seen in Figure 5.21.

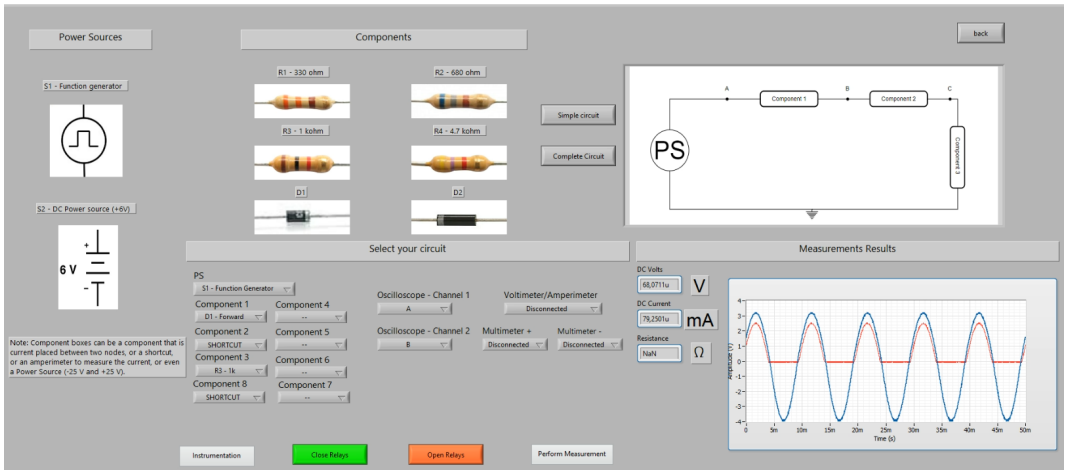


Figure 5.21: Half-wave rectifier at student interface

In case the student wants to test a more complex circuit, he should select the "Complete Circuit" button and the image shown in Figure 5.22 will appear. In this Figure, the current between nodes B and C is being measured when using the +6 V source between A and 0 and the +25 V (regulated to +5 V) between E and 0. The resistors used are 680 Ω between A and B, 1 kΩ between F and 0 and 330 Ω between C and D. When the experiment is performed, the value of approximately 2.38 mA is obtained. The theoretical value is 2.41 mA.

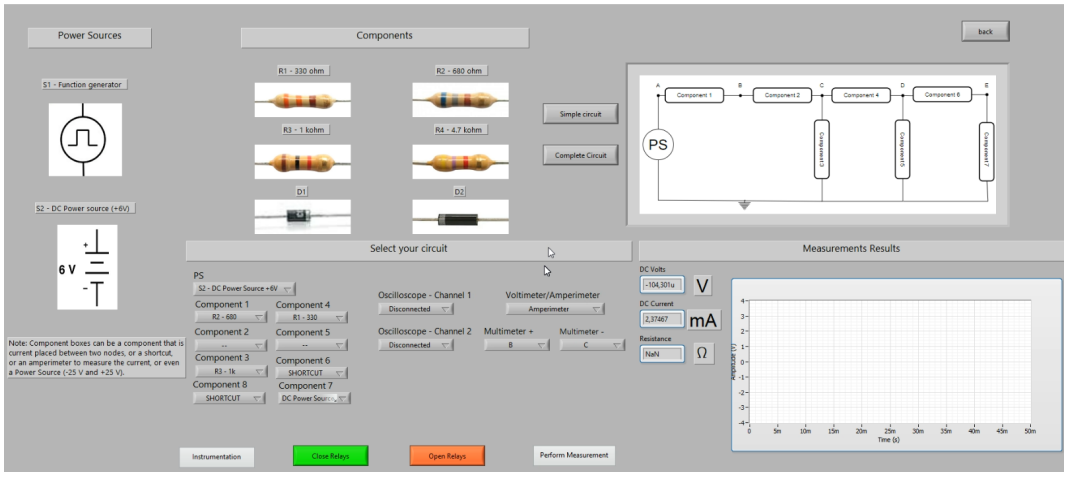


Figure 5.22: Current measurement example

All the theoretical results presented in this chapter have been obtained using the equations obtained in chapter 3 and confirmed through the respective simulations. As the theoretical results are very close to the results obtained, it can be concluded that the results have been validated and the objectives presented have been successfully completed.

Chapter 6

Conclusion

The project developed and described in this dissertation has proved that it is possible to build and empirically test electronic circuits designed using a user interface This system has features that have solved some problems existing in previously developed systems, but it also has its limitations.

The key elements of this project are the relays, the microcontroller (Arduino) and VirtualBench. Therefore, it has been proved that through these main elements, it is possible to test various circuits through the user's choice. The values obtained experimentally were very close to those obtained theoretically and in the simulations, which validates the concept developed.

This project can be divided into two parts: one corresponding to the hardware developed, including all the studies until reaching the final solution. The purpose of this part is to enable the choice of various circuits, including power supplies and measuring instruments. Taking into account what has been developed, if you want to have a greater number of possible experiments at the same time, you only need to add more nodes and, consequently, more relays and more microcontroller(s) (depending on the number of I/Os) starting from the same concept that has already been implemented. The second part consists of the interfaces developed. The purpose of the teacher/technician interface is to enable the teacher to carry out experiments, to test them before allowing the student to carry out a new experiment. He can also indicate which nodes he has physically placed each component between. In this interface, it is also possible to test the correct functioning of the relays, which is a very important feature that was lacking in the laboratories studied. The student

interface allows students to choose the circuit they want to test, taking into account the options allowed. In this interface, it is possible to graphically visualise the circuit possibilities. Both interfaces allow you to control the sources and instruments in VirtualBench.

However, as mentioned earlier, the developed project has limitations. On the hardware side, current measurement in all branches requires a large number of nodes which, in the case of larger circuits will grow exponentially, as well as the number of relays and the number of I/O necessary. On the software side, the interfaces are not very dynamic, as they were developed with a specific set of components and circuits in mind. For new experiences and new components used, some changes need to be made.

As already mentioned, LabVIEW's web publishing tool has been used to enable remote access to the laboratory. However, this functionality was not fully functional as it presented problems due to access issues or software versions and there was no time to solve this issue.

Overall, this project met the requirements initially stipulated, and provided a fully functional prototype with a considerable set of working experiments, however, it has considerable room for progress.

6.1 Future Improvements

As mentioned, this project has room to grow into a more robust and complete laboratory.

In the hardware field, the number of nodes should be increased so that it is possible to measure the current in all the branches in the circuit. It should also be increased to cover a wider range of experiments. Note that the number of relays increases with the number of nodes (connections to the measuring devices). That said, it is impossible to use just one microcontroller due to the lack of I/O pins. It is therefore suggested to use a central Arduino that receives the main string and sends the string for each functionality (components, sources, oscilloscope, multimeter) to each Arduino associated with it, and each one controls the relays on each board. This communication can be carried out over the Serial Peripheral Interface (SPI) protocol, which is an Arduino feature.

Components not used in this project should be implemented, such as capacitors, coils, amps, transistors, etc. The addition of some of these was studied when researching the state of the art, but there was no time to implement them. In the case of components with more than two pins (as transistors), it is suggested that each pin will be connected to a different node. Using this type of component makes measuring the current in each of the branches even more complex. In the case of

capacitors and coils, attention must be paid to the energy storage characteristic of these components so that there are no damaged components.

The two interfaces need to be more dynamic. Firstly, contact must be established between the two so that they are interconnected. In other words, when the teacher indicates on their interface which nodes each component is between, it is not yet done automatically so that the student interface shows the option of components on the dropdown button (this functionality is being developed). The second improvement to the interfaces is that the interfaces were developed for the experiences available at the moment and, for new experiences or new components, there are changes that need to be made. To avoid making these changes every time something is added, the interfaces need to be made more dynamic. For example, in the teacher's interface, there should be an input field so that the teacher can indicate which components are placed in the circuit and can be connected or not. On the student interface, there should be more image options to allow for more combinations of experiments, or the graphical part of the interface should be solved in another way, given LabVIEW's limitations in this area.

References

[1] I. Gustavsson, "User-defined electrical experiments in a remote laboratory," *2003, American Society for Engineering Education*, 2003. [Cited on page 1]

[2] I. Gustavsson *et al.*, "Traditional lab sessions in a remote laboratory for circuit analysis," *15th EAEEIE Annual Conference on Innovation In Education for Electrical and Information Engineering*, 2004. [Cited on pages 1, 7, 8, and 34]

[3] I. Gustavsson, J. Zackrisson, H. Åkesson, and L. Håkansson, "A flexible remote electronics laboratory," 2006. [Cited on pages x, 1, 2, 7, and 37]

[4] F. Jacob, *VISIR remote laboratory utilization and improvement guidelines*. PhD thesis, Univ. Nacional de Educación a Distancia (UNED), Madrid, 2022. [Cited on pages 2, 6, 7, 8, 31, 34, 35, and 48]

[5] "Mobile laboratories as an alternative to conventional remote laboratories," 2017. [Cited on page 2]

[6] L. D. Feisel and A. J. Rosa, "The role of the laboratory in undergraduate engineering education," *Journal of Engineering Education*, pp. 1–2, 2005. [Cited on page 5]

[7] A. Pester and T. Klinger, "Distributed experiments and distributed learning," *International Journal of Online and Biomedical Engineering (iJOE)*, vol. 16, no. 6, pp. 19–32, 2020. [Cited on page 5]

[8] L. Gomes and S. Bogosyan, "Current trends in remote laboratories," *IEEE Transactions on industrial electronics*, vol. 56, no. 12, pp. 4744–4756, 2009. [Cited on pages ix, 6, 7, and 11]

[9] N. Lima, *Fostering Experimental Competences Using Complementary Resources*. PhD thesis, Univ. of Salamanca, Salamanca, 2020. [Cited on page 8]

[10] www.thepocketlab.com. [Cited on page 8]

[11] www.pslab.io. [Cited on pages ix, 8, and 9]

[12] I. Gustavsson, "A traditional electronics laboratory with internet access," 2003. [Cited on page 9]

[13] www.pspice.com. [Cited on page 10]

[14] I. Gustavsson, J. Zackrisson, L. Håkansson, I. Claesson, and T. Lagö, "The visir project – an open source software initiative for distributed online laboratories," *Remote Engineering and Virtual Instrumentation (REV 2007)*, 2007. [Cited on pages 10, 30, and 42]

[15] www.everycircuit.com [Cited on page 11]

[16] www.falstad.com [Cited on page 12]

[17] phet.colorado.edu [Cited on page 13]

[18] P. Orduña, D. Zutin, S. Govaerts, I. Lequerica Zorrozua, P. Bailey, E. Sancristobal, C. Salzmann, L. Rodriguez-Gil, K. Delong, D. Gillet, M. Castro, D. López-de Ipiña, and J. Garcia-Zubia, "An extensible architecture for the integration of remote and virtual laboratories in public learning tools," *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 10, pp. 223–233, 2015. [Cited on page 14]

[19] F. Luthon and B. Larroque, eds., *LaboREM—A Remote Laboratory for Game-Like Training in Electronics*. IEEE, 2015. [Cited on pages 14 and 16]

[20] B. L. Camille Lavayssière, Bastien Letowski and F. Luthon, eds., *LaboREM—A Network of Open Source Remote Laboratories for Learning*. Anglet, France: Univ. Pau Pays Adour/ E2S UPPA, IUT Bayonne Institute of Technology. [Cited on pages ix, 14, 16, 17, and 18]

[21] C. Lavayssière, B. Larroque, and F. Luthon, "Laborem box: A scalable and open source platform to design remote lab experiments in electronics," *HardwareX*, vol. 11, pp. 5–14, 2022. [Cited on pages ix, 15, 16, and 17]

[22] B. Letowski, C. Lavayssière, B. Larroque, and F. Luthon, eds., *An Open Source Remote Laboratory Network Based on a Ready to Use Solution: LaboREM*. Seville, Spain: 12th annual International Conference of Education, Research and Innovation, 2020. [Cited on pages ix and 15]

[23] M. Moussa, A. Benachenhou, A. Mebrouka, and A. Adda-Benattia, "Design of a low cost switching board enabling a reconfigurable remote experiment," *International Journal of Online and Biomedical Engineering (iJOE)*, vol. 15, no. 12, pp. 32–41, 2019. [Cited on pages ix, 18, 19, and 20]

[24] M. Moussa, A. Benachenhou, A. Boumehdi, and A. Adda-Benattia, "Mostalab: Performance evaluation of simultaneous access in analog remote laboratories," *International Journal of Online and Biomedical Engineering (iJOE)*, vol. 17, no. 6, pp. 98–109, 2021. [Cited on pages ix, 19, and 20]

[25] F. Ouatik, F. Ouatik, H. Fadli, A. Elgorari, M. E. Mohadab, M. Raoufi, B. Bouikhlene, and M. Skouri, "E-learning decision making system for automate students assessment using remote laboratory and machine learning," *Journal of E-Learning and knowledge society*, vol. 17, no. 1, pp. 90–100, 2021. [Cited on pages ix, 21, and 22]

[26] F. Ouatik, M. Raoufi, F. Ouatik, and M. Skouri, "Online instrument systems in reality for remote wiring and measurement of electronic in e-learning from labview+ni elvis ii vs embedded system+web services," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 11, no. 2, pp. 1178–1185, 2021. [Cited on pages ix, 21, 22, 23, and 24]

[27] N. Lewis, M. Billaud, D. Geoffroy, P. Cazenave, and T. Zimmer, "A distance measurement platform dedicated to electrical engineering," *IEEE Transaction on learning technologies*, vol. 2, no. 4, 2009. [Cited on pages ix, 24, and 25]

[28] A. Carpeño, S. López, and J. Arriaga, "Using remote laboratory elab3d for a broader practical skills training in electronics," *11th International Conference on Remote Engineering and Virtual Instrumentation (REV)*, 2014. [Cited on pages ix, 26, 27, and 28]

[29] S. López, A. Carpeño, and J. Arriaga, "Remote laboratory elab3d: A complementary resource in engineering education," *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 10, no. 3, pp. 160–167, 2015. [Cited on pages ix, 26, 27, and 28]

[30] J. G. Zubía, I. Gustavsson, U. H. Jayo, P. Orduña, I. Angulo, and J. R. de Garibay, "El proyecto visir en la universidad de deusto: Laboratorio remoto para electrónica básica," *Madrid, Spain*, pp. 653–664, 2010. [Cited on pages 29, 46, and 47]

[31] I. Gustavsson, "A tutorial on the visir open laboratory platform and an invitation to join the visir community," *Remote Engineering and Virtual Instrumentation (REV 2009)*, 2009. [Cited on pages ix, x, 29, 34, 40, 41, and 49]

[32] M. Tawfik, E. Sancristobal, S. Martin, R. Gil, G. Diaz, A. Colmenar, J. Peire, M. Castro, K. Nilsson, J. Zackrisson, L. Hakansson, and I. Gustavsson, "Virtual instrument systems in reality (visir) for remote wiring and measurement of electronic circuits on breadboard," *IEEE Transaction on learning technologies*, vol. 6, no. 1, pp. 60–72, 2013. [Cited on pages x, 29, 36, 38, 41, 46, 47, 48, and 49]

[33] M. Tawfik, E. Sancristobal, S. Martín, C. Gil, A. Pesquera, P. Losada, G. Díaz, J. Peire, M. Castro, J. García-Zubia, U. Hernández, P. Orduña, I. Angulo,

M. C. C. Lobo, M. A. Marques, M. C. Viegas, and G. R. Alves, "Visir deployment in undergraduate engineering practices," *2011 Frontiers in Education Conference (FIE), Rapid City, SD, USA*, 2011. [Cited on pages ix and 30]

[34] F. Garcia-Loro, E. S. Cristobal, G. Diaz, M. Castro, A. Fidalgo, G. Alves, P. Orduña, U. Hernandez-Jayo, J. Garcia-Zubia, C. Garcia, R. Tavio, W. Kulesza, K. Nilsson, C. Kreiter, A. Pester, K. Valtonen, and E. Lehtikangas, "Pilar: Sharing visir remote labs through a federation," *2019 IEEE Global Engineering Education Conference (EDUCON), Dubai, United Arab Emirates*, pp. 102–106, 2019. [Cited on pages x, 30, 35, 41, and 49]

[35] M. Tawfik, "Visir installation start-up guide v.1," *Electric, Electronic and Control Engineering Department Spanish University for Distance Education (UNED)*, 2011. [Cited on pages x, 31, 34, 36, 44, 45, 46, 49, 50, and 51]

[36] www.ni.com/pt-pt/support/model.pxi 1033.html [Cited on pages ix and 31]

[37] www.ni.com/pt-pt/support/model.pxi 5114.html [Cited on pages ix, 31, and 32]

[38] www.ni.com/pt-pt/support/model.pxi 5402.html [Cited on pages ix and 32]

[39] www.ni.com/pt-pt/support/model.pxi 4072.html [Cited on pages ix and 33]

[40] www.ni.com/pt-pt/support/model.pxi 4110.html [Cited on pages ix and 33]

[41] I. Gustavsson, K. Nilsson, W. Kulesza, F. Garcia, and M. Castro, "Visir relay switching matrix v4.1 – user's manual v.8," pp. 1–48, 2019. [Cited on pages x, 34, 35, 38, 39, 40, 43, 49, and 51]

[42] L. Rodriguez-Gil, P. Orduña, J. Garcia-Zubia, and D. L. de Ipifia, "Advanced integration of openlabs visir (virtual instrument systems in reality) with weblab-deusto," *Remote Engineering and Virtual Instrumentation (REV 2012)*, 2012. [Cited on page 35]

[43] Y. Larbaoui and A. Naddami, "Description, analysis and characterization of visir system toward extending its use to various fields of experimenting," *2021 World Engineering Education Forum/Global Engineering Deans Council (WEEF/GEDC), Madrid, Spain*, pp. 584–593, 2021. [Cited on page 35]

[44] I. Gustavsson, J. Zackrisson, J. S. Bartunek, K. Nilsson, L. Håkansson, I. Claesson, , and T. Lagö, "A flexible remote electronics laboratory," *Remote Engineering and Virtual Instrumentation (REV 2008)*, 2008. [Cited on page 35]

[45] J. García-Zubía, U. Hernández, I. Gustavsson, and G. Alves, "Academic effectiveness of visir remote lab in analog electronics," *Erasmus+ Programme of the European Union.* [Cited on page 42]

[46] J. Zackrisson, I. Gustavsson, and L. Håkansson, "An overview of the visir open source software distribution 2007," *Remote Engineering and Virtual Instrumentation (REV 2007)*, 2007. [Cited on pages x and 42]

[47] U. Hernández, "Visir remote lab user manual," *LabsLand*, 2016. [Cited on pages x, 44, 45, and 46]

[48] I. Gustavsson, K. Nilsson, J. Zackrisson, J. Garcia-Zubia, U. Hernandez-Jayo, A. Nafalski, Z. Nedic, O. Gol, T. L. Jan Machotka, Mats I. Pettersson, and L. Hakansson, "On objectives of instructional laboratories, individual assessment, and use of collaborative remote laboratories," *IEEE Transaction on learning technologies*, vol. 2, no. 4, pp. 263–274, 2009. [Cited on page 48]

[49] www.ni.com/en/shop/electronic-test-instrumentation/virtualbench/what-is virtualbench.html [Cited on page 54]

[50] National Instruments, *VirtualBench All-in-One Instrument*, 2019. [Cited on pages x, 54, 55, and 56]

[51] National Instruments, *NI VirtualBench VB-8012 Specifications*, 2014. [Cited on page 55]

[52] www.botnroll.com/pt/arduino-controladores/53-arduino-mega 2560-8058333490083.html [Cited on pages x and 57]

[53] S. Miskam and S. M. Yamin, "Mechatronic devices switches and relays," p. 55, 2021. [Cited on pages 58 and 59]

[54] COMUS International, *3570-1331*, 2015. [Cited on pages x and 58]

[55] COMUS International, *3570-3572-3563-Series*, 2015. [Cited on pages x and 59]

[56] S. G. Hassan, "Relay driver and power factor correction concept with application in microcontrollers," p. 12, 2015. [Cited on page 60]

[57] STMicroelectronics, *Seven Darlington arrays*, 2019. [Cited on page 60]

[58] A. Rahnamei, F. Khoshnevis, M. Vajdi, and P. Farhadi, "A design for car anti-theft system using cell phone," *International Journal of Advanced Scientific and Technical Research*, vol. 1, p. 5, 2012. [Cited on pages x and 61]

[59] www.viewpointusa.com/labview/what-is-labview-used for [Cited on page 61]

[60] learn.ni.com/learn/article/labview tutorial [Cited on page 61]

[61] S. Zurek, *Two-dimensional magnetisation problems in electrical steels.* PhD thesis, Cardiff University, Wales, United Kingdom, march 2005. [Cited on page 62]

[62] E. Jaara, "Introduction to proteus," p. 6, 2016. [Cited on page 63]

[63] M. A. Rashid, "Rectifier – half wave rectifier and full wave rectifier," p. 16, 2020. [Cited on page 67]

Appendix A

Multimeter connections in Proteus

Figure A.1 shows the connection circuit for the voltmeter and ammeter in Proteus. This Figure is presented here to make it easier to read, given its large size.

It is very important to understand these connections in order to understand which relays need to be switched on to connect the desired instrument to the necessary nodes.

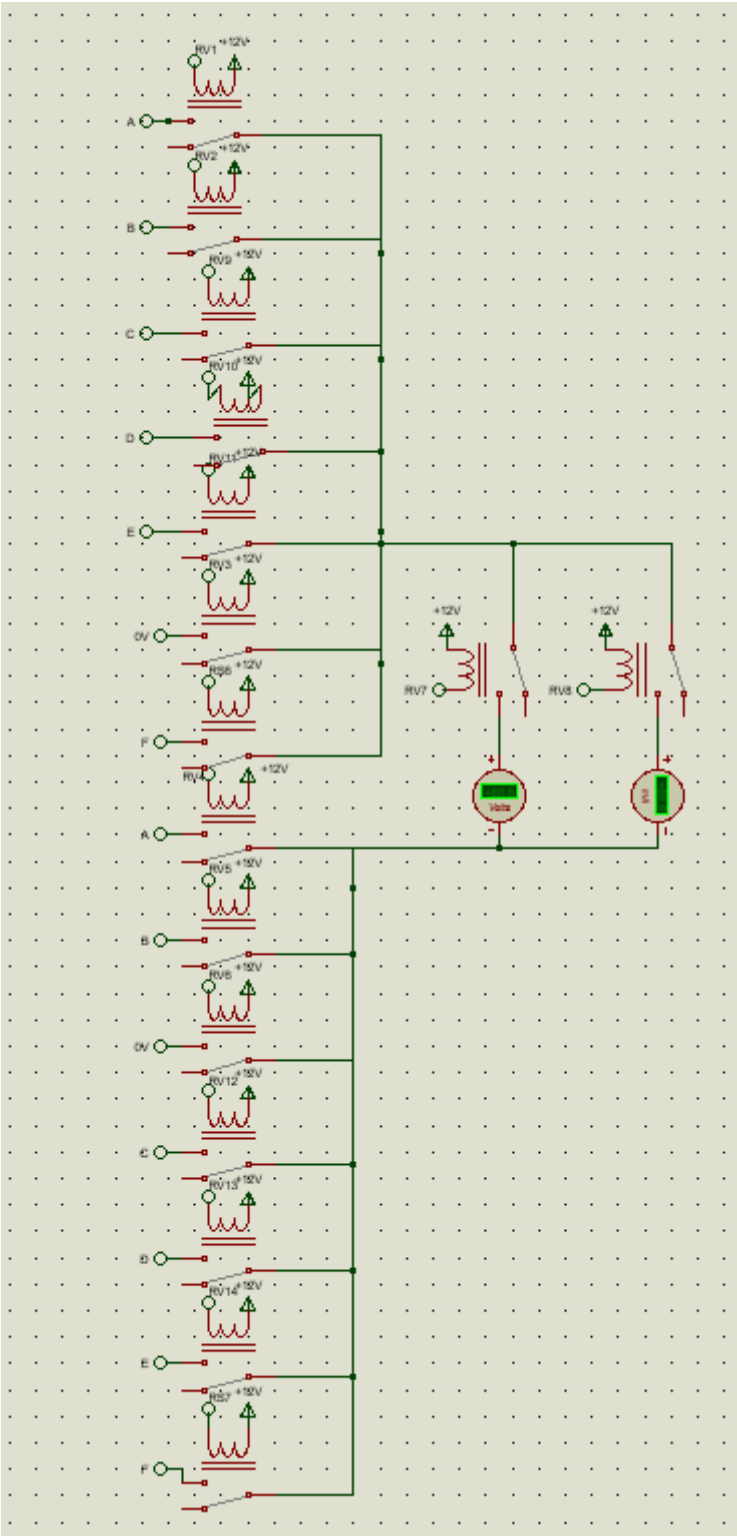


Figure A.1: Voltmeter and Ammeter connections in Proteus

Appendix B

Actual circuit in Proteus

Figure B.1 shows the actual circuit in Proteus (components, shortcuts and power supplies), which is presented here due to its large dimensions.

It is very important to understand the operation of the circuit presented in order to understand the basic dynamics of the laboratory developed. It is important to realise which relays need to be closed in order to obtain the desired circuit and the necessary measurements, and also to understand that the connections of the components and shortcuts to the nodes must be changed in the simulation if they are physically changed.

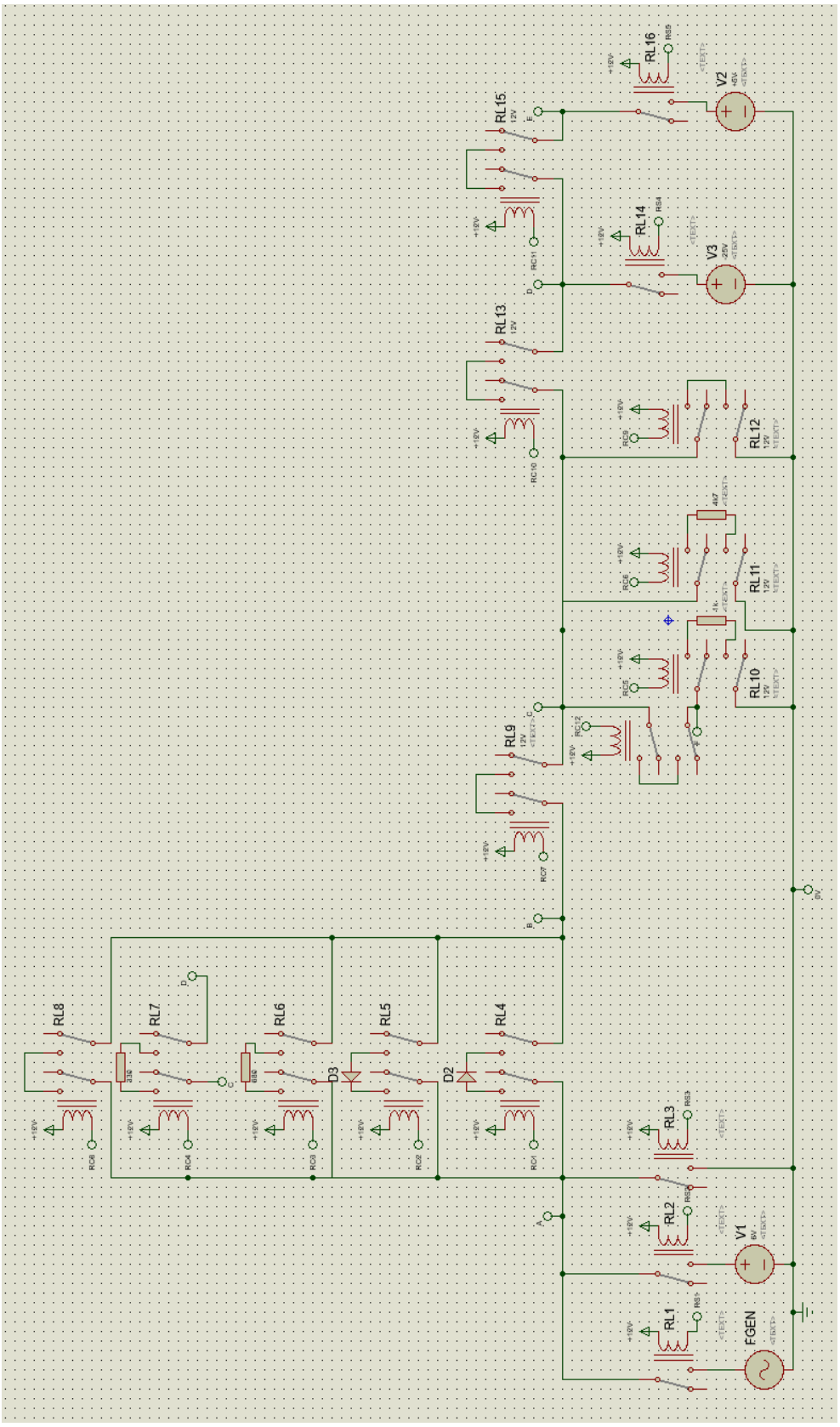


Figure B.1: Present circuit in Proteus

Appendix C

Arduino code

```
int relays_comp1[7]={8, 7,6, 5, 4, 3, 2};
int relays_comp2[7]={25,23,13,12,11,10,9};
int relays_source[7]={34,32,30,28,26,24,22};
int relays_osc[7]={48,46,44,42,40,38,36};
int relays_dmm[7]={43,41,39,37,35,33,31};
int relays_dmm2[7]={53,52,51,50,49,47,45};

void setup()
{ Serial.begin(9600);
  Serial1.begin(9600);
  int relay_comp1, relay_source, relay_osc,
      relay_dmm,relay_dmm2, relay_comp2;
  for(int i=0; i<7; i++)
  {
    relay_comp1=relays_comp1[i];
    pinMode(relay_comp1, OUTPUT);
    relay_comp2=relays_comp2[i];
    pinMode(relay_comp2, OUTPUT);
    relay_source=relays_source[i];
    pinMode(relay_source, OUTPUT);
    relay_osc=relays_osc[i];
    pinMode(relay_osc, OUTPUT);
    relay_dmm=relays_dmm[i];
    pinMode(relay_dmm, OUTPUT);
    relay_dmm2=relays_dmm2[i];
    pinMode(relay_dmm2, OUTPUT);
  }
```

Listing C.1: Setup code

```
1  void ProcessString(String message)
2  {uint8_t reles_comp1, reles_comp2, reles_source, reles_osc,
       reles_dmm, reles_dmm2;
3     if(message.startsWith("000"))
4    {  for(int i=0; i<7; i++)
5      { digitalWrite(relays_source[i], 0);
6      }
7      for(int i=0; i<7; i++)
8      { delay(200);
9        digitalWrite(relays_osc[i], 0);}
10     for(int i=0; i<7; i++)
11     { delay(200);
12       digitalWrite(relays_dmm[i], 0);
13       digitalWrite(relays_dmm2[i],0);}
14     for(int i=0; i<7; i++)
15     { delay(200);
16       digitalWrite(relays_comp1[i], 0);
17       digitalWrite(relays_comp2[i], 0);}
18   }
19   if(message.startsWith("001"))
20   { reles_comp1=message.substring(3,6).toInt();
21     reles_comp2=message.substring(6,9).toInt();
22     for(int i=0; i<7; i++)
23     { digitalWrite(relays_comp1[i], bitRead(reles_comp1,i))
         ;
24       digitalWrite(relays_comp2[i], bitRead(reles_comp2,i))
           ;}
25   }
26   else if(message.startsWith("024"))
27   { reles_source=message.substring(3,6).toInt();
28     for(int i=0; i<7; i++)
29     { digitalWrite(relays_source[i], bitRead(reles_source,i
         ));}
30   }
31   else if(message.startsWith("016"))
32   { reles_osc=message.substring(3,6).toInt();
33     for(int i=0; i<7; i++)
34     { digitalWrite(relays_osc[i], bitRead(reles_osc,i));}
35   }
36   else if(message.startsWith("017"))
37   { reles_dmm=message.substring(3,6).toInt();
38     reles_dmm2=message.substring(6,9).toInt();
39     for(int i=0; i<7; i++)
40     { digitalWrite(relays_dmm[i], bitRead(reles_dmm,i));
41       digitalWrite(relays_dmm2[i], bitRead(reles_dmm2,i));}
42   }
```

Listing C.2: ProcessString(String Message) function code

```
1  void loop()
2  {
3    String str, comp1,dmm,osc,src;
4    String strings[4]={comp1,dmm,osc,src};
5    if (Serial1.available()>0)
6    {
7      str=Serial1.readString();
8      delay(10);
9      Serial.println(str);
10     if(str.startsWith("000"))
11     {
12       ProcessString(str);
13     }
14     else{
15           ProcessString("000");
16           delay(200);
17           strings[0]=str.substring(0,11);
18           strings[1]=str.substring(12,23);
19           strings[2]=str.substring(24,35);
20           strings[3]=str.substring(36,47);
21           for(int i=0; i<4;i++)
22           {
23             delay(i*200);
24             ProcessString(strings[i]);
25           }
26     }
27   }
28 }
```

Listing C.3: loop() function code

Appendix D

LabVIEW's Code

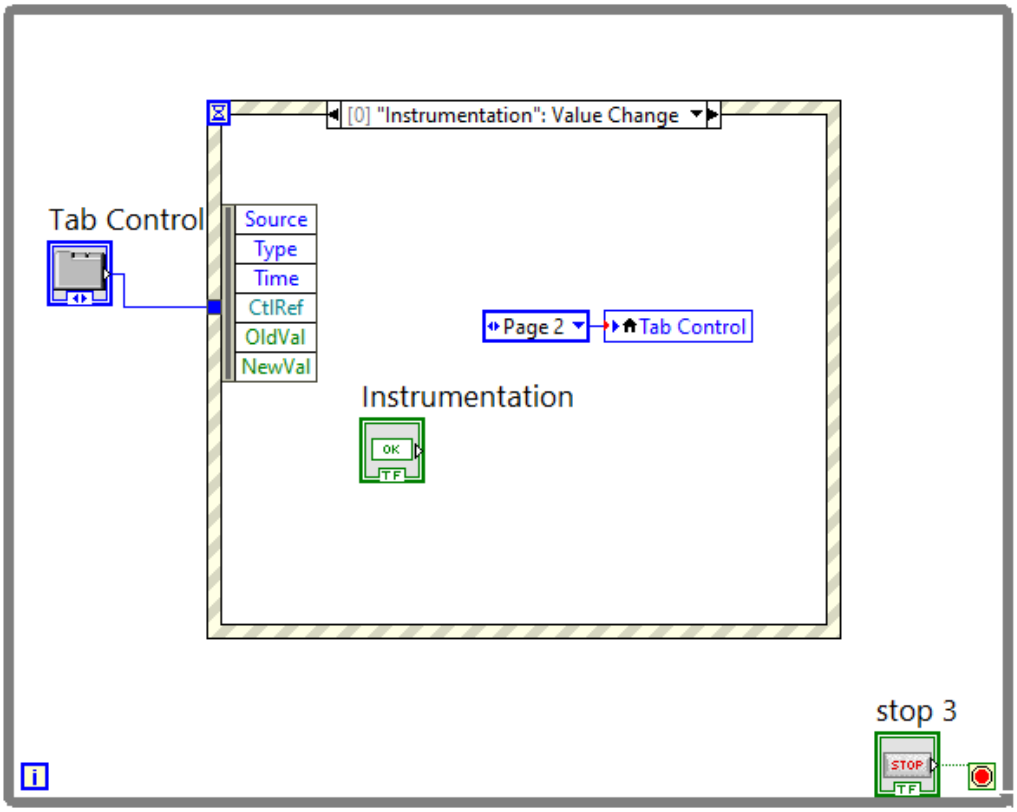


Figure D.1: Tab control code

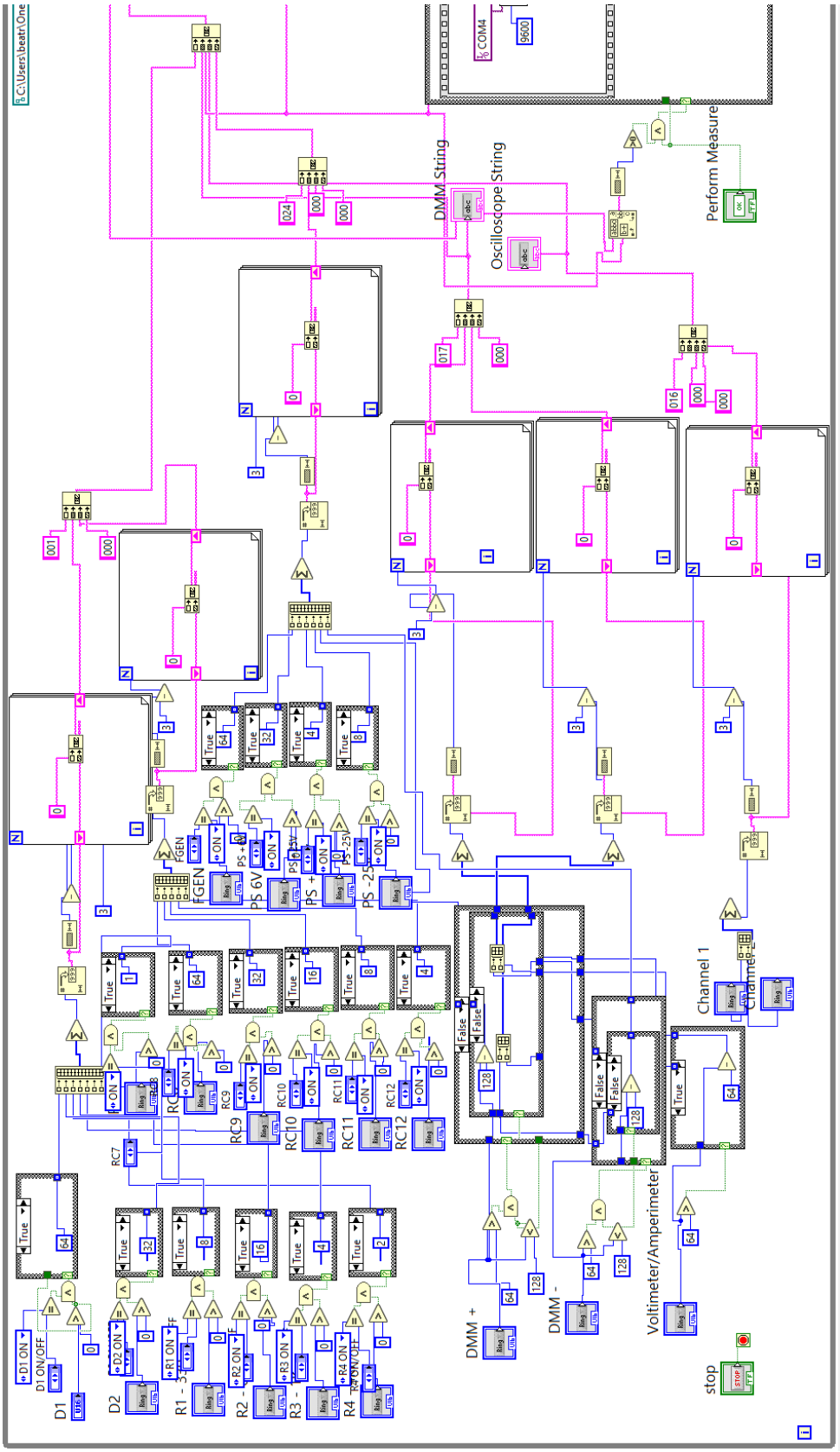


Figure D.2: Final string development code in teacher interface

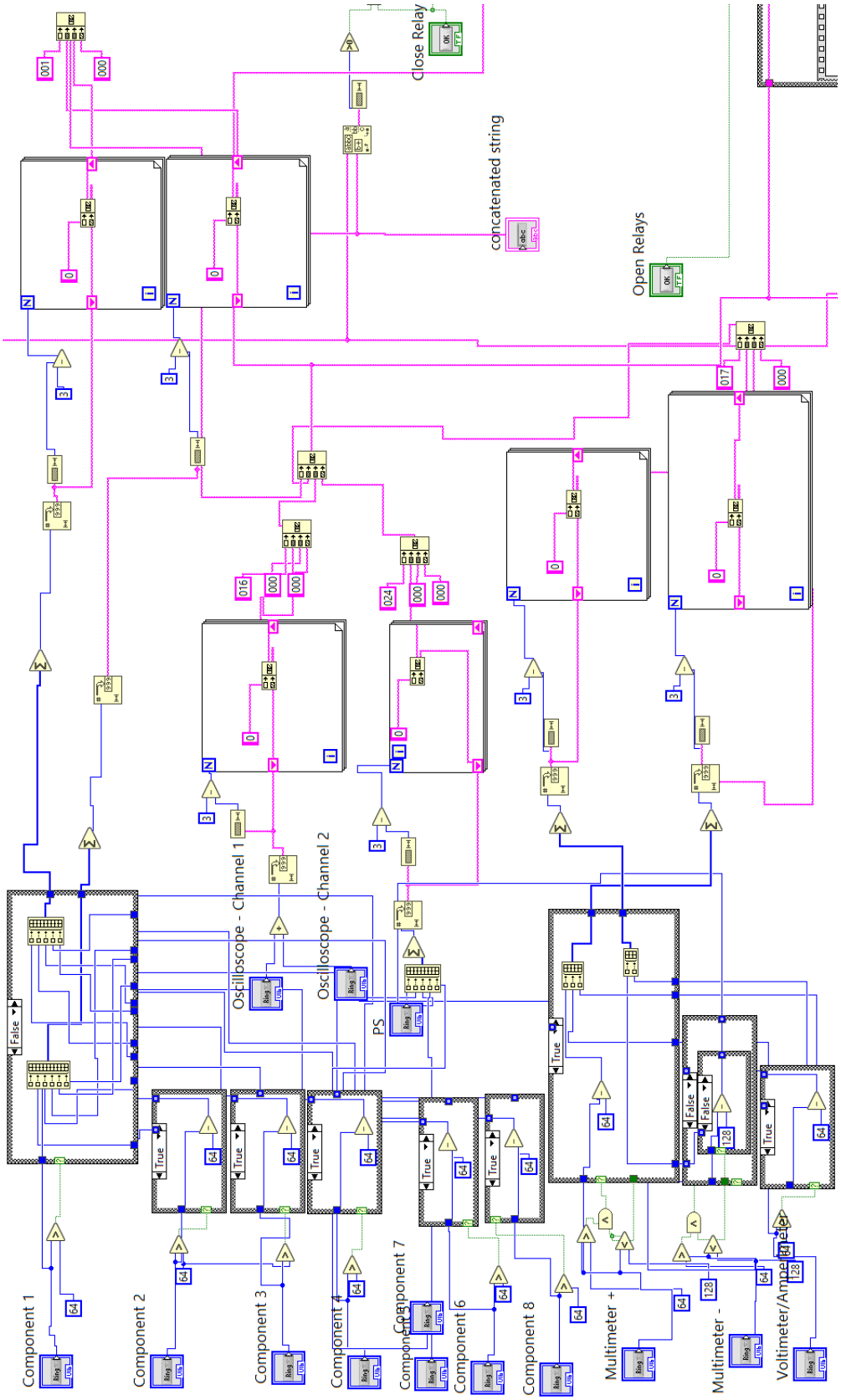


Figure D.3: Final string development code in student interface

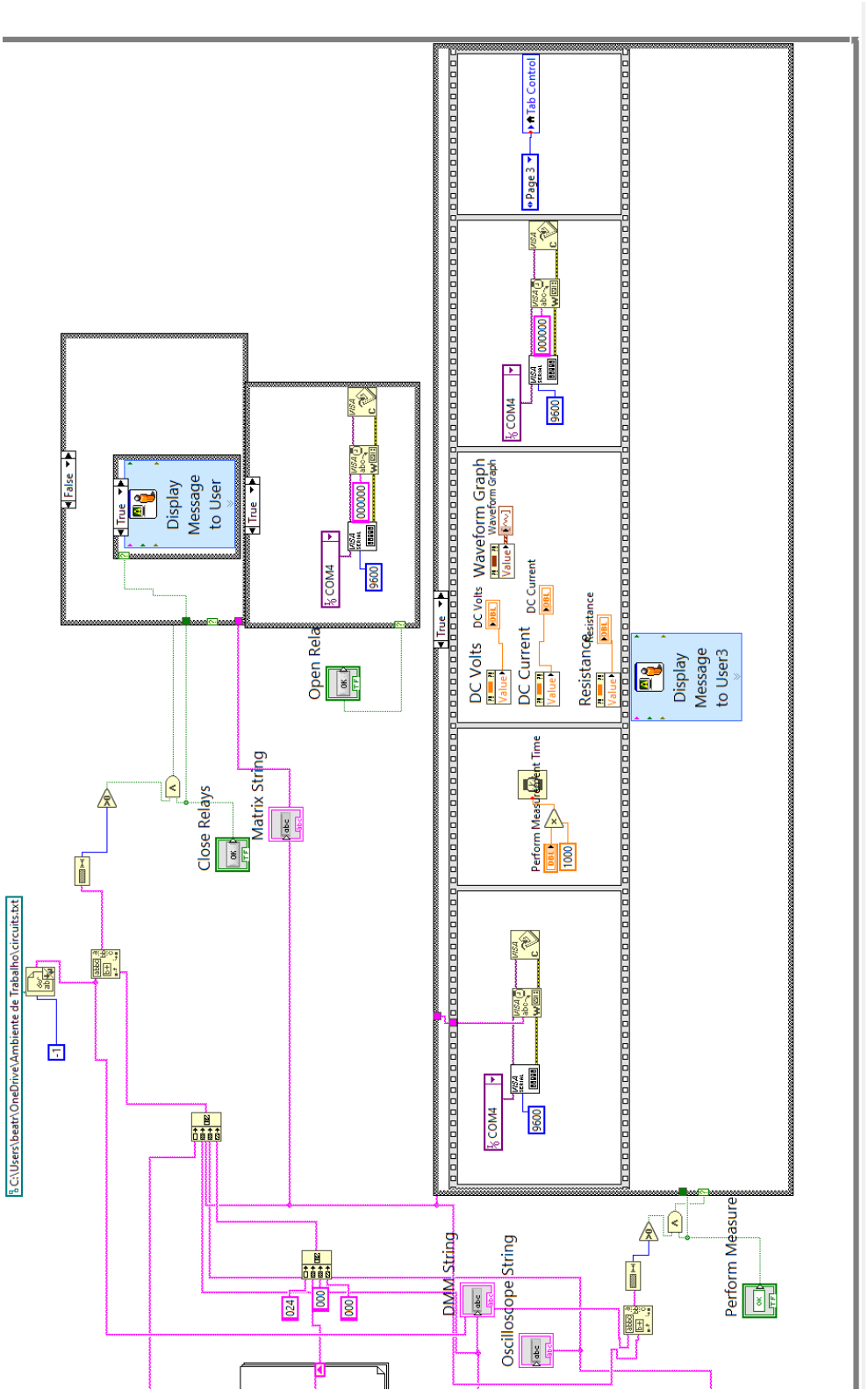


Figure D.4: Send string code

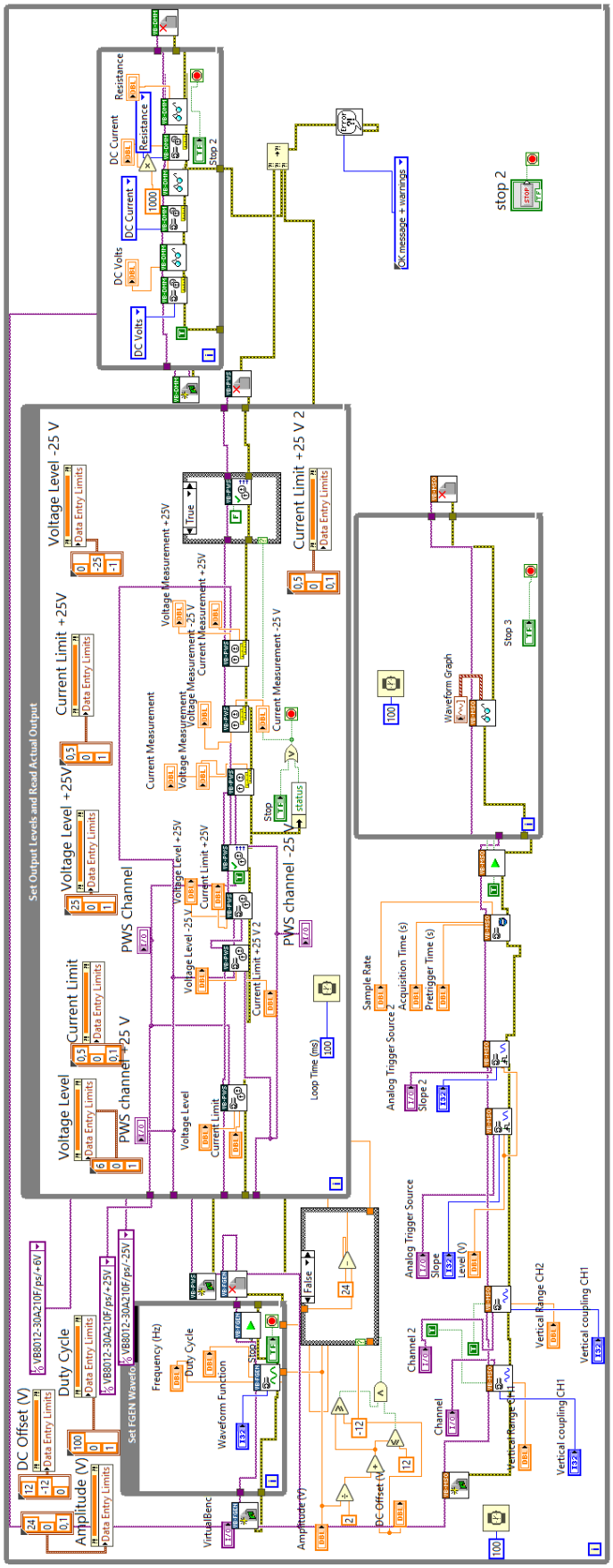


Figure D.5: Instrumentation control code

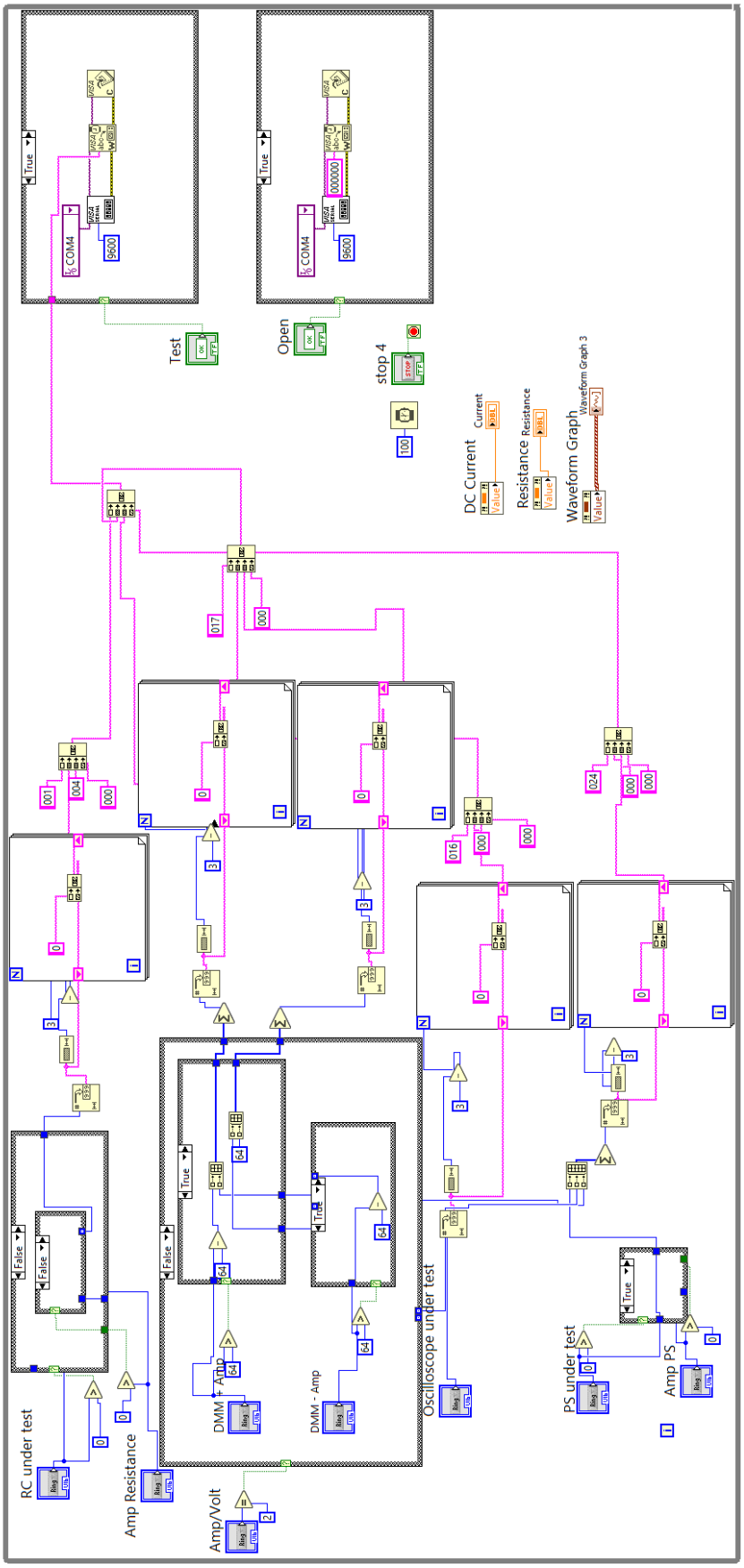


Figure D.6: Test mode code