

## Article

# Emulation of Circuits under Test Using Low-Cost Embedded Platforms

José-María Guerrero-Rodríguez <sup>1</sup>, Clemente Cobos Sánchez <sup>1,\*</sup>, Ángel Quirós-Olozábal <sup>1</sup>  
and Juan A. Leñero-Bardallo <sup>2</sup>

<sup>1</sup> Area of Electronic, Escuela Superior de Ingeniería, University of Cádiz, 11519 Puerto Real, Spain; josem.guerrero@uca.es (J.-M.G.-R.); angel.quiros@uca.es (Á.Q.-O.)

<sup>2</sup> Electronic and Electromagnetism Department, University of Seville, 41012 Seville, Spain; jlenero@us.es

\* Correspondence: clemente.cobos@uca.es

**Abstract:** Electrical engineering education requires the development of the specific ability and skills to address the design and assembly of practical electronic circuits, as well as the use of advanced electronic instrumentation. However, for electronic instrumentation courses or any other related specialty that pursues to gain expertise testing a physical system, the circuit assembly process itself can represent a bottleneck in a practical session. The time dedicated to the circuit assembly is subtracted both to the measurements and the final decision-making time. Therefore, the student's practical experience is limited. This article presents a reconfigurable physical system based on the Arduino™ shield pin-out, which (after specific programming) can virtually behave as a device under test to carry out measurement procedures on it, emulating any system or process. Although it has been mainly oriented to the Arduino boards, it is possible to add different control devices with a connector compatible. The user does not need to assemble any circuit. Our approach does not only pursue the correct instrument handling as a goal, but it also immerses the student in the context of the functional theory of the proposed circuit under test. Consequently, the same emulation platform can be utilized for other techno-scientific specialties, such as electrical engineering, automatic control systems or physics courses. Besides that, it is a compact product that can be adapted to the needs of any teaching institution.

**Keywords:** electronic instrumentation training; circuit emulation; simulation; embedded platform and Arduino; electronic engineering education; courseware



check for updates

**Citation:** Guerrero-Rodríguez, J.-M.; Cobos Sánchez, C.; Quirós-Olozábal, Á.; Leñero-Bardallo, J.A. Emulation of Circuits under Test Using Low-Cost Embedded Platforms. *Electronics* **2021**, *10*, 1990. <https://doi.org/10.3390/electronics10161990>

Academic Editors: Iouliia Skliarova and Arkaitz Zubiaga

Received: 29 June 2021

Accepted: 15 August 2021

Published: 18 August 2021

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Measurement and instrumentation usage are the essential means of experimentation and checking in the natural sciences as well as in the engineering tasks. For this reason, learning about measurement art is a basic topic of scientific and technical high education. In this sense, the laboratory practices should provide practical competencies destined to give value to the theoretical knowledge acquired by the students. In the case of particular subjects of electronic engineering specialties, these practices must allow them to master the techniques necessary for the assembly of components, as well as to know and handle the different types of measurement equipment. However, in electronic instrumentation subjects teaching, two general limitations must be considered when addressing practical contents in the laboratory:

1. On the one hand, the circuits, the required components and the measurement instruments, with the consequent expenditure of economic resources for the different purchases.
2. On the other hand, the temporal distribution of lab experiences, which requires an optimized learning plan to provide a maximum benefit of practice-time targeted to the electronic instrumentation handling.

Usually, the main problems that we face when performing practical electronic measurements at the laboratory are due to the fact that a device (component, circuit, or equipment)

is required to carry out the measuring. The accumulated experience as teachers allows us to guarantee that one of the main difficulties detected is the excessive time to assembly interesting prototypes. Next, several tests and measurements are required to check the correct assembly and operation of the circuit. Due to this required extra-time, the fundamental objective of the instrumentation laboratory experiences must be to accelerate the circuit-under-test assembly process to accommodate the required measurements stage.

For this reason, different manufacturers of educational training equipment offer a variety of circuits and engineering training kits to facilitate the specific learning of the main functional modules of each specialty (digital, power or analog electronics, control systems, etc.). This method allows decreasing the assembly periods during the lab sessions. On the other hand, other manufacturers of educational equipment or instrumentation have placed on the market several solutions to operate and to learn on electronic instruments such as oscilloscopes or logic analyzers. However, most of the solutions presented are ad-hoc products to provide a set of explicit measurements, but predominantly the device-under-test (DUT) is not configurable.

For example, Naugra<sup>®</sup> offers an oscilloscope demonstrator/trainer kit specifically designed for the study of the oscilloscope as open equipment [1]. This company also offers a box (product code CON-024-NE) [2] which presents a variety of output signals and simulated waveforms for training on digital storage oscilloscope (DSO) measurement techniques.

Moreover, the company Velleman<sup>®</sup> offers a simple printed circuit board (PCB) [3] to learn to use the oscilloscope. This board generates a set of different waveforms which can be used to perform measurements using a DSO. In this case, the developed learning product is a non-configurable DUT.

In addition, STB (discontinued) and STB-3 boards from Siglent<sup>®</sup> [4] are intended for teaching and demonstrating about the use of an oscilloscope. These boards generate several output waveforms for the oscilloscope operation training.

Rigol<sup>®</sup> has presented the DS6000 Demo board [5] for illustrating the basic functions of the oscilloscope. This product can output 25 kinds of signals for the explanation of diverse techniques and oscilloscope setups. This amount of complex output signal allows learning an efficient use of the oscilloscope.

Additionally, GW Intek Co., Ltd. (New Taipei City, Taiwan) [6] has developed the GDB-02 (discontinued) or the updated GDB-03 training kit. This one allows learning both basic and advanced testing operation of the GDS-3000 Series DSO. This demo board can actuate as a sophisticated signal generator module capable of producing stimulus which represents different real-life scenarios typical on electronics.

Tektronix<sup>®</sup> has introduced a software tool to learn oscilloscope operation using Arduino<sup>™</sup> IDE and a few discrete components added to actuate as DUT [7]. The proposed 22 lab experiments, around the Arduino platform, include amplitude and timing measurements, as well as, advanced practice of the DSO. In this learning method, the Arduino board is configured using a computer to select a particular program code to carry out a specific test procedure. However, the card programming pursues to simulate none electronic particular architecture.

Carullo et al. [8] have presented an educational kit based on a computer equipped with a general-purpose audio-card and open-source software to drive this card and a test-board proposed by them. The PC audio-card actuates both as a function generator (outputs) and a software-based basic oscilloscope (inputs). The authors only provide information about blocks of the test-card: different active filters, a mixer, and an astable buffered-oscillator. Therefore, it follows that the test-card is not easily reconfigurable and requires hardware changes.

Potkonjak et al. [9] have reviewed several virtualization proposals to achieve virtual laboratories applying new emerging software topics. However, software-based virtualization can be an excessive solution in the case of electronic instrumentation practices. A high effort (for modeling and programming) is required to achieve a true immersion of the

student in the laboratory experience, and the contact with the real electronic circuits and instruments would be lost.

Grimaldi and Rapuano [10] have analyzed the virtual laboratory (VL) and distributed VL (DVL) concepts for education in instrumentation and measurement using virtual instruments software, physical instruments, and computer networks. However, referring to the fundamentals of the measurement procedure, we consider that direct manipulation of the instruments is mandatory in the first stages of learning.

Angrisani et al. [11] have utilized the reconfigurable features of Field Programmable Analog Array (FPAA) devices embedded in the Anadigm QuadApex commercial board from Anadigm<sup>®</sup> to configure a remote lab method for practices in electronic. In this case, the users can uniquely remotely choose one topology from a set of available pre-configured analog circuits that can be implemented by the FPAA and carry out the measurements, but without direct manipulation of the instruments.

Letowski et al. [12] have proposed a remote hardware laboratory (Laborem) based on the Rasp-Pi board computer, using a multiplexing method that interconnects a large number of functional physical boards, as well as diverse electronic instrumentation. Again, the purpose is orientated to remote education on electronics by making the corresponding measurements, but without particular handling of the instruments nor oriented to the specific learn about the measurement techniques.

Considering the above solutions, we propose a 'dummy' DUT based on a reconfigurable-platform, which can emulate/simulate (in [13], the interrelation of the emulation and simulation concepts is analyzed) any basic circuit to learn measurement methods by operating electronic instrumentation.

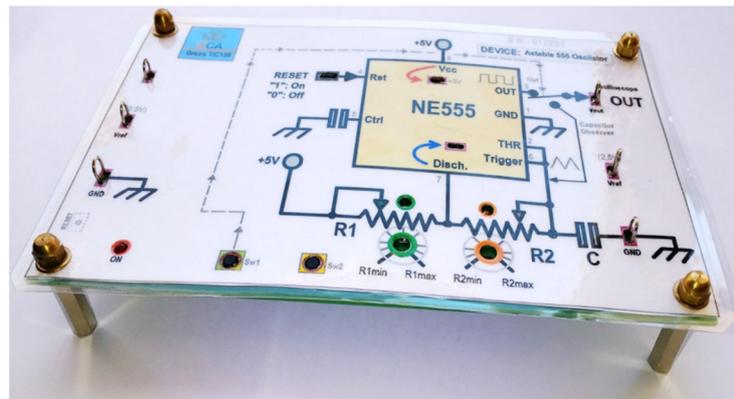
The paper is organized as follows: Section 2 briefly describes the designed circuit and presents the simulation-procedure basics. Section 3 presents a concise review of several suggested programming techniques to implement an emulation project by employing this card. Section 4 explains some typical circuit examples and results. In Sections 5 and 6, future works and conclusions are discussed, respectively.

## 2. Description of the Emulation Platform

### 2.1. Emulator Overview

Our proposal consists of a 16 cm × 10 cm flat work-panel (Figure 1). The different elements (electronic components, knobs, and switches) are mounted below this level, on the PCB bottom-side. The top smooth work surface can support a graphical cover (interchangeable) that shows an illustrated reference of the emulated circuit. The emulation will be running on a microcontroller-based development card. Our choice was the Arduino board for two reasons: first, this is a known worldwide device and supported by numerous publications of technical documentation and programming tutorials. Second, it features an interface connector that has imposed a standard pin-out arrangement, which is now used by other programmable platforms from other companies (e.g., some PSoC based development platform from Cypress Microsystems<sup>™</sup> or Galileo series boards from Intel<sup>®</sup>) [14]. In the same way, many of these programmable boards use the Arduino IDE (Integrated Development Environment).

Another example of the Arduino open-source hardware phenomenon may be the case of TSXpert (an award-winning National Instruments<sup>™</sup> Alliance Partner), which has introduced the Arduino compatible compiler for LabVIEW<sup>™</sup>. This software is a compiler that takes a virtual instrument program and compiles and downloads it to Arduino compatible platforms [15]. National Instrument also facilitates (free) the LabVIEW Interface for Arduino (LIFA) Toolkit which allows developers to acquire data from the Arduino microcontroller and process them in the LabVIEW environment. On the other hand, MathWorks<sup>®</sup> has taken advantage of Arduino's acquisition capabilities to use it as an interface with the real world toward its Matlab<sup>®</sup> and Simulink<sup>®</sup> software [16].



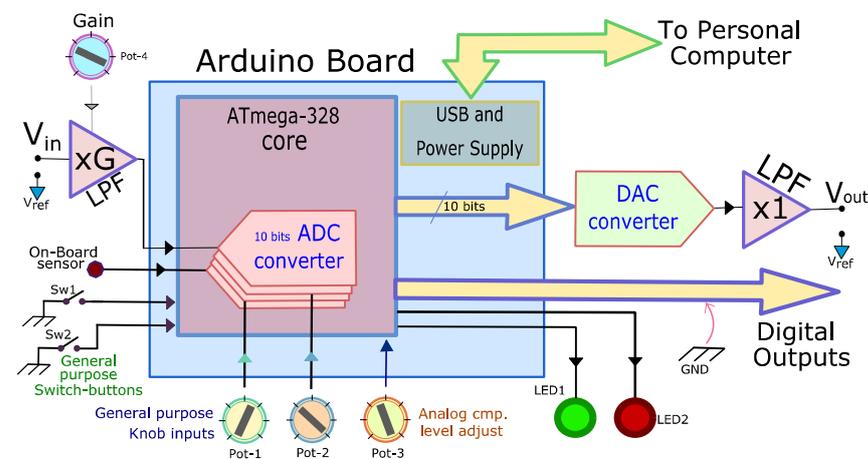
**Figure 1.** A picture of the emulator proposed. In this case, the “dummy” shows a cover-illustration (interchangeable) to emulate the classical 555-IC oscillator circuit.

Given this background, exploiting the Arduino platform as the project core is a justified choice. Based on that, two Arduino development card types could be the project driver, namely:

- The Arduino UNO edition [17], an AVR<sup>®</sup> MCU version (includes the 8 bits ATmega328 microcontroller from Microchip/Atmel running at 16 MHz).
- The Arduino ZERO [17] (or your clone), a product ARM<sup>®</sup>-SAND21 based (a 32 bits Cortex M0 MCU from Atmel) at 48 MHz clock, or Arduino DUE card, powered by a SAM3X8E ARM Cortex<sup>®</sup>-M3 CPU (from Atmel3) running at 84 MHz. These ARM-based cards will be able to run more efficiently the required simulation software, regard to the eight-bit MCU. However, it should be noted that in these two cards, the microcontroller works from a 3.3 V power supply instead of a 5 V one as the original Arduino UNO. This power supply change requires small re-adjustments in the program variables and some hardware reworks (potentiometers and pull-up resistors must be connected to 3.3 V).

## 2.2. Emulation Platform Board

Figure 2 shows a block diagram of the proposed emulator circuit. It consists of an extension of the Arduino card. There are two main groups of input/outputs elements, regarding analog or digital actuation.



**Figure 2.** Block diagram of the training platform proposed.

First, referring us to the analog signals, the emulator can use four of six 10-bits-ADC channels available inside of the microcontroller. Therefore, it is possible to dedicate one ADC input to acquire an external analog signal, two more to enter variables manually (Pot-1 and Pot-2), and the fourth channel is reserved for an on-card analog sensor.

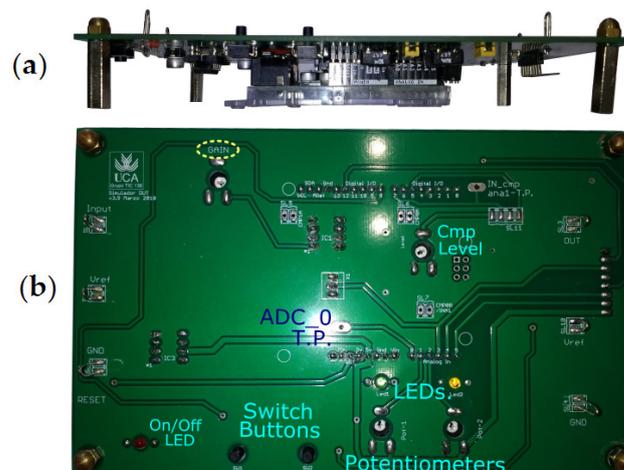
The external analog signal goes to ADC through a manually adjustable gain amplifier (from one to ten, handling Pot-4). This amplifier with two imposed poles works as a second-order anti-aliasing filter. Since the ATmega series does not integrate a DAC module, a converter (10 bits, DAC101S101 from Texas Instrument) and its output buffer have been externally added. Again, two-poles imposing, achieve that this buffer works as a second-order aliasing filter.

The in/out interfacing operational amplifiers are mounted on DIP sockets. In case of damage as a result of mishandling during practice, this provides a fast and easy repair.

Second, referring now to the logic events, two push-buttons (Sw1 and Sw2) are added as discrete interaction inputs. As digital outputs, the board uses some microcontroller specific pins (e.g., timer-count outputs or standard digital ports). In addition, the emulator board presents two signaling LEDs to report particular states during the simulation runtime.

The ATmega328 AVR microcontroller has an embedded analog-comparator. The logic result can decide to execute a specific interrupt routine. The comparator non-inverter input is accessible on the emulation card (OUT-2 pin), while the inverter pin is connected to the Pot-3 potentiometer out for a manual adjustment of the comparator threshold level (from 0 to 5 V DC). This microcontroller multiplexes on the same pin the non-inverter input from the comparator, and a standard digital out. The programmer can decide the identity of this pin at each time by configuring it.

The controller board plugs on the backside of the emulation card (Figure 3a). On this same side are mounted the external components associated with the Arduino board. In addition, a located reset-button on this one side let us to initialize the emulator if necessary.



**Figure 3.** The emulator PCB: (a) lateral view, showing the Arduino UNO plugged. (b) Top side (work area).

The emulator card is raised using four brass 20 mm long separators. This height is enough to host the Arduino board and the taller electronic components mounted under the PCB.

The total cost is approximately 31 € (15.6 € for electronic components, and 15.2 € for printed circuit board), excluding the controller board.

### 2.3. The Work Surface

We aim to achieve a smooth work-surface and to show an interchangeable graphical illustration representative of the DUT (equipment, circuit, or component) to be emulated (Figure 3b). Therefore, the printed circuit must be flat to accommodate the plastic or laminate paper cover which includes a DUT illustrative drawing. Only the connection terminals for the test instruments must remain on the upper side. The potentiometer knobs must appear flat respect to the surface and not exert any impediment. In some experiences, the unused components could stay hidden under the illustrated cover. Figure 4 shows the

lower-side of the emulator PCB. The potentiometers are positioned under the PCB plane. A hole permits to adjust the input parameters using a plastic screwdriver. These actuate as variable parameters after being converted by a 10-bit ADC.

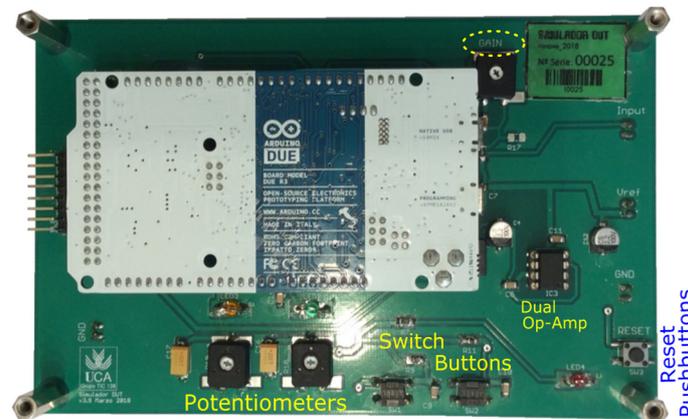


Figure 4. Emulator PCB, bottom side (with a DUE board plugged).

Beginning an emulation procedure requires choosing a symbolic figure of the simulated circuit. The laboratory instructor has the help of a graphical template where the elements appear in their real physical positions (Figure 5). Over this starting template, we must depict the different blocks and labels which describe the device’s functional behavior, using any graphics editor or paint software. The graphical scales must be respected so that the template fits correctly on the work surface by adjusting the different holes over every controller device or connector located on the board.

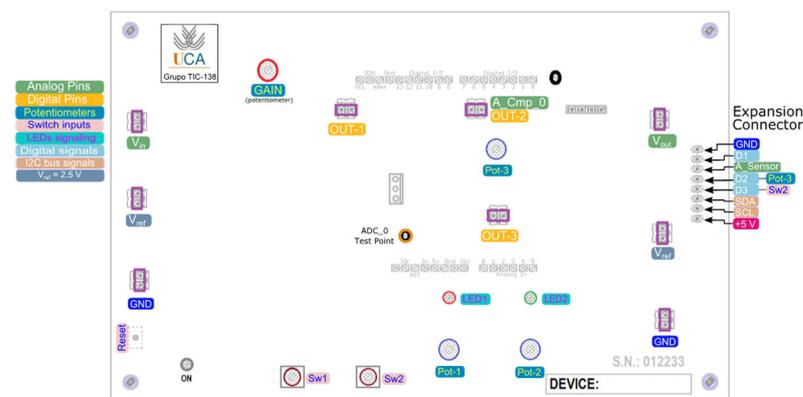
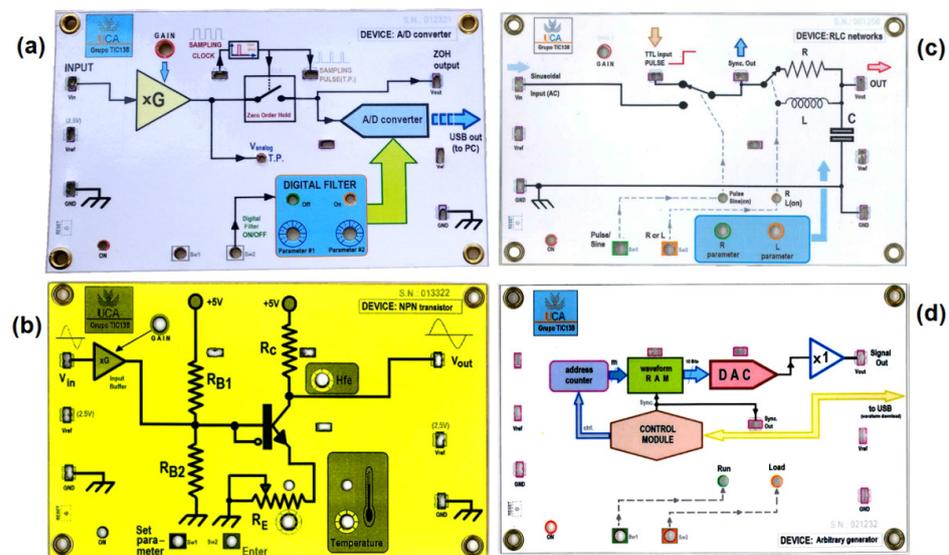


Figure 5. Work area layout showing the input/output elements location.

The designed graphical-cover should be printed in color over pasteboard. A cut precisely near its edges dimension is required. We must advise protecting the paperboard using laminating pouches and a thermal machine. For access to each connector, knob or, LED used on the emulation board and located under the graphical cover, the needed holes must be drilled using drill-pliers or a round-hollow punch set. Figure 6 shows several covers created following the previous indications.



**Figure 6.** Several graphical-covers examples: (a) analog acquisition and basic digital processing experience; (b) the BJT amplifier; (c) first and second order low-pass filter convolution; (d) arbitrary waveform synthesis.

#### 2.4. Emulating and Programming

The DUT behavior must be simulated running real-time algorithms, previously written in the platform selected (Arduino IDE or compatible development system, board FPGA based or low-cost single board computer). Therefore, the lab practices instructor must consider the fundamental aspects of the real device performance and describe them as a program code. Generally, most emulation routines will require either an analog signal (input/output) or several digital outputs. In the case of the analog synthesis, the mathematical description of the waveform signal must be described into a closed-loop routine (for-loop or while-loop structures) and every successive calculated sample, in real-time, must be sent to the DAC. For the digital output cases, several pins are available. A program can be configured as a finite state machine (using if-then structure and time-delay functions) to generate the digital word necessary for emulation on the digital output pins.

Regarding the Arduino UNO board (16 MHz clock), a square waveform of around 91 kHz can be synthesized using directly two commands from IDE. This value is the maximum since the software routine now is only dedicated to generating this digital waveform, and does not execute any additional computation.

The processing and synthesis rate can be improved by using the assembler code from the Arduino IDE (in-line assembler instructions, raising in this case around 2.2 MHz for square wave generation) or the C language. In both cases, the results are quite similar. However, these programming methods imply a more sophisticated level of program writing for less qualified lab instructors.

In the Arduino DUE product, the frequency value increasing to 215 kHz for the synthesis of only a square wave programmed directly in the IDE (by using the Digital-Write() command), and of the order of one or two tens of MHz using the inline assembler editor. It must be remembered that the ARM processor integrated into the DUE card does not have assembly instructions for handling I/O logic levels directly as AVR series. This programming is more complicated and could be written using the 'Atmel Studio' IDP (integrated development platform) for developing and debugging the microcontrollers from Microchip® [18].

The generation of waveforms (sinusoidal, triangular, saw, etc.) requires the execution of an algorithm that calculates successive samples to send them sequentially to the DAC circuit. For instance, the synthesis of a 1024-point sine waveform (executing (int) (512–500 \* sine(phase) instruction repeatedly), running on the Arduino UNO card, provides a 26 Hz real-time output

signal and 98 Hz if implementing on the DUE edition. The generation of other 1024-points functions such as triangle-wave or ramp-wave yields a frequency waveform around 12 Hz and 118 Hz, respectively, using UNO or DUE cards.

It is essential to optimize code execution, between successive samples, to achieve new resulting samples in a shorter time than the I<sup>2</sup>C transfer time and D/A conversion delay. For signal synthesis cases, the look-up tables (LUT) can decrease the execution time of these routines. By using this method, the previously calculated values go directly to the DAC without needing for real-time computing. If the signal waveform is periodic in time, this procedure is even more interesting than the mathematical interpolation techniques.

It is recommendable to execute the required calculations during the pauses between consecutive samples acquisition. In addition, it should be used lock-up-tables (LUT) to avoid real-time long-computations. It must be minimized the code execution times for switches or potentiometers status acquisition. It is better to duplicate similar code blocks (if there is enough memory) because this leads to faster than subroutine jumps.

### 2.5. Operation in the Laboratory

Once the lab instructor has prepared the emulation program and created the corresponding graphical cover, the test procedure in the laboratory is as follows:

1. Students will assemble the emulator card, the Arduino module and the cover, and will connect them to the PC via USB. We suppose that every student has previously all the necessary educative material concerning the current practice.
2. They download the emulation program from the PC with the help of the Arduino IDE.
3. Next, they must follow the present script, which leads to the set of measures and observations that the lab session contemplates.
4. After the measurements, a report should be prepared (readings values and their corresponding tolerance range).

It is possible to particularize the lab practice to every student work-bench, but it requires setting slightly different parameters for each emulation program. Consequently, the readings taken by every student will also be different, which can be of a great help to avoid plagiarism.

## 3. Algorithmic Methods

The Arduino IDE provides a complete set of instructions (structures, essential mathematical functions and operators, analog and digital in-out access, etc.). Using these instructions correctly, any programmer with knowledge of electronics could write an emulation program for a standard circuit. However, we must emphasize the fact that the execution speed is compromised if the code size is excessive.

It is not possible to establish a general programming method that satisfies all emulation situations. Each emulation case will require a particular synthesis code that generates the simulated results solicited and takes advantage of the best available resources on the Arduino platform, the IDE and, the elements of the emulator card.

In general, the emulation program must process the input signals and parameters (knobs and pushbuttons) to generate the aimed behavior that will be manifested by analog or digital output data. The real-time reading of parameters from the potentiometers ADC channels and the switch-buttons status detection introduces a delay within the loops dedicated to the processing algorithms.

The analog input can be captured from the first A/D converter channel executing an IDE immediate instruction. A similar IDE code also handles the auxiliary sensor input. The Arduino-UNO's core lacks an integrated D/A converter. To manage the added DAC (from Texas Instruments, configured via I<sup>2</sup>C), the programmer has a specific routine that facilitates the sending of data (Send\_DAC() function).

For code execution speed-up, we can suggest the use of interruption routines. This programming strategy requires more extensive knowledge of the features microcontroller and more programming skills, but this method can achieve improved performance, tak-

ing advantage of the A/D acquiring times or avoid the polling method by reading the input switch-buttons and the potentiometers. Below, a list of several methodological recommendations to address the simulation of some specific devices is exposed:

### 3.1. State Machines

Basically, decision structures such as the `if{ }` and `else{ }` blocks, and the conditional loops `do{ }-while` and `while{ }` are used. All instructions are fully documented in the Arduino reference page [17].

The decision variables will be the discrete status from buttons SW1 and SW2 (when pressing), as well as a particular level of the input signal captured by the main ADC. The Arduino IDE has direct instructions to read digital (`DigitalRead()`) and analog (`AnalogRead()`) ports.

The outputs will be the available digital ports or then analog output and will depend on the determinate current state, according to the flow chart of the sequential system.

### 3.2. LUT Based Combinational Logic

Combinational circuits can easily be emulated reading a software array-based LUT, which previously contains the corresponding truth tables for all output pins. In order to use this option, it needs to decide the involved input and output pins. A Boolean equation for each digital output must be created (from the truth tables). These values will appear in the different columns of the array. The decoded inputs place a pointer at a particular array position, and the corresponding binary data will directly go to the output pins. This method can also be applied to generate the output values corresponding to the current state in the sequential-circuits simulation cases (Mealy/Moore).

### 3.3. Convolution and Transfer Function

Equation (1) presents the expression for the discrete-time convolution [19]:

$$y(i) = \sum_{k=0}^{N-1} h(k)x(i-k) \quad (1)$$

Basically, this equation allows us to determine the response  $y(i)$  of the circuit to emulate, according to its discrete transfer function  $h(k)$ , from the sampled sequence of discrete excitation  $x(n)$ . The operation is simple and primarily based on the use of a MAC algorithm (Multiply and Accumulate). Therefore, it is possible a real-time execution in microprocessors and programmable devices [20].

When the discrete transfer function  $h(k)$  from a particular circuit is known (impulse response), this algorithm allows the real device response synthesis from the current analog input and the stated parameters. However, this routine could be considerably slow when the emulated circuit requires massive calculations due to a high  $k$  index value.

### 3.4. Digital Filters

The convolution algorithms, which establish the finite impulse response (FIR) or infinite impulse response (IIR) digital filter routines, are well documented in the current extensive literature. There is also a vast bibliography about the implementation of digital processing routines and filters, running at low-level, using simple processors such as those shown in [21,22]. In [23], Fernandez et al. propose some methods for the implementation of digital filters in the Arduino DUE, taking advantage of its ARM SAM3X8E core controller.

Referring us to the FIR case, the general equation that represents this digital filter type is:

$$y = \sum_{i=0}^n b_i x[n-i] = b_0 x[n] + b_1 x[n-1] + \dots + b_n x[0] \quad (2)$$

where  $x[ ]$  is the acquired input signal array,  $y$  is the current output sample to send to DAC,  $n$  is the filter order, and  $b_i$  the different filter coefficients. The complexity of the algorithm depends on the filter order: the number of calculations increases according to the  $n$  index.

For lower order filters, it is possible the real-time computing of all necessary operations to obtain the output data flux by processing the current input sample and the previously acquired input samples. The programmer can simplify this task using specific libraries available for signal processing in Arduino [24].

In order to find the  $b_i$  filter coefficients, there is a diversity of software tools (licensed or freeware) to aid the digital filter design process. After the required filter design and verification process, these tools provide a list of parameters to directly synthesize the digital filter algorithm.

To accelerate the filter algorithms, it is interesting the C-language or assembler code as programming options. For the first case, the lab instructor has available mathematical libraries for the Atmel Studio IPE [25]. It is also possible to directly program filter algorithms in assembler code [26–28] or any filter architecture on FPGA based.

In any case, it is always possible to perform a DUT's emulation by scaling the frequency axis, taking advantage of the impedance and frequency scaling theorems. It is achievable to simulate a DUT or circuit at a different frequency regarding the originally (nominally) required. This allows executing a feasible emulation-code in real-time. Following the required measurements, the results must be reconsidered according to the original frequency, undoing the previous scaling.

Similarly, IIR filters type are expressed by

$$\begin{aligned} y[n] &= \sum_{i=0}^n b_i x[n-i] - \sum_{i=1}^n a_i y[n-i] = \\ &= b_0 x[n] + b_1 x[n-1] + \dots + b_n x[0] - a_1 y[n-1] - a_2 y[n-2] - \dots - a_n y[0] \end{aligned} \quad (3)$$

where now  $x[ ]$  is the acquired input signal array,  $y[ ]$  is the output previous samples array,  $y[n]$  the current sample to send to DAC,  $n$  is the filter order, and  $a_i$  and  $b_i$  the different filter coefficients. In this case, the complexity is higher and the algorithm is, even more, dependent on the filter order.

### 3.5. Wave Synthesis Method

Wave digital filter (WDF) is another method to model classic filters. This technique allows real-time emulation of analog topologies. Although a vast amount of literature exists on this subject, an extensive exposition of this topic may be found in the work of Fettweis [29]. The structures based on LC groups are easily implemented using this method on fixed-point processors [30,31] or configurable devices [32,33]. That represents an advantage to execute the real-time emulation of small circuits.

### 3.6. Custom Libraries

To simplify the emulator's programming tasks, we have generated a particular library with specific functions to optimize the emulation card. For this, a new platform declaration titled DOCENTIA has been created (file package\_docentia\_index.json). After the correct installation, it can be chosen as a single device within the Arduino IDE (Tools -> Boards -> Boards Manager -> DOCENTIA platform). This allows us to particularize some IDE variables to decrease the A/D conversion time, incorporate some specific instructions for handling the external DAC, and simplify the filters programming by adding several useful functions.

### 3.7. Communication between the Emulator and a PC

The Arduino IDE provides an instruction set for sending and receiving data from a personal computer, via USB. The execution of a specific PC application allows data download and the modification, in real-time, of the emulated DUT behavior or status. In addition, this application could evaluate any results from the platform and process them with other entered data from typical computer inputs (keyboard, mouse, etc.).

PC and emulator combination facilitates to create smarter trainer equipment. A PC application can supervise different measures sequences and could automatically generate

any simulated failure that the students must solve following the expected troubleshooting process. The use of the  $\text{Rand}()$  function allows particular configurations slightly different for each emulation board. Therefore, the results measured by the students will be different too. In addition, the use of barcodes can help to simplify the identification between the illustrated cover, the corresponding program to load from the IDE and the educational material necessary to perform the lab practice.

#### 4. Examples and Results

The proposed emulator presents a very varied possibilities set. Our group has explored a significant number of feasible applications that include not only circuit simulation for electronic instrumentation training, but also other specialties such as electrical engineering or physical sciences. For sake of brevity, in the following we mention just a few examples:

##### 4.1. Comparator with Hysteresis

The purpose of this experience is to learn the comparator with hysteresis performance (Schmitt-trigger emulation, Figure 7), as well as to get used to the two channels measurements using the digital oscilloscope. Taking advantage of the analog output (via DAC), the simulated output waveform can include the voltage drops for low and high output levels ( $V_{OL}$  and  $V_{OH}$ ) during the synthesis process.

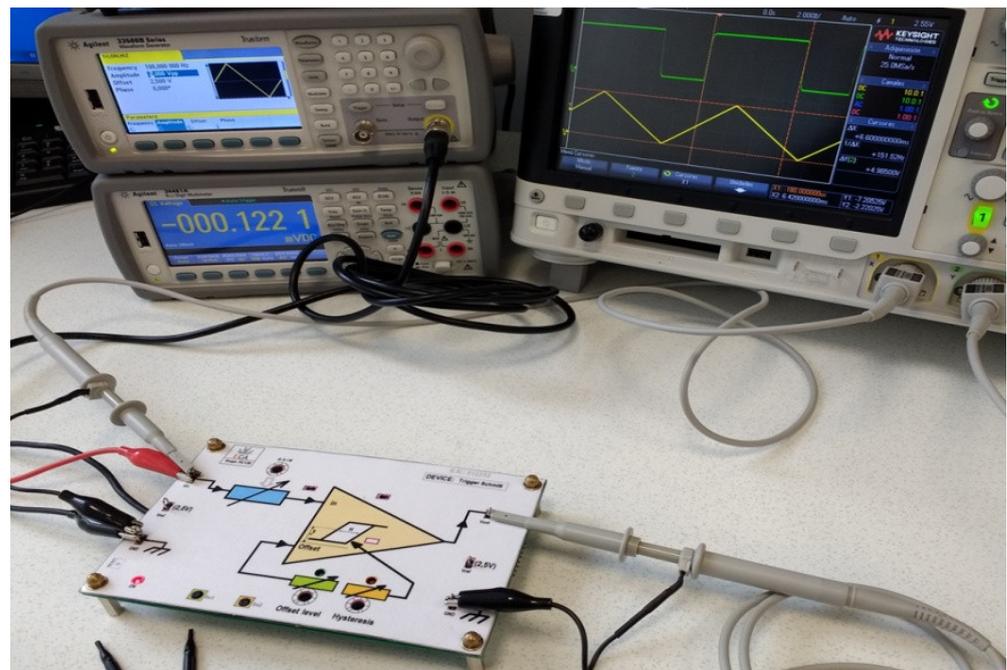


Figure 7. Schmitt-trigger circuit emulation.

The excitation-signal to be used is a triangular waveform, 5 V peak to peak, with 2.5 V offset and frequency around 100 Hz. A very simplified code, which takes into account only the logic states transitions and the output, is shown in Figure 8.

```

void loop()
{
  Input = analogRead(A0); // "Input" save the last input voltage sample

  // Hysteresis & Off_Set computation:
  // UN0: For 5V0, FS and 10 bits, Delta_V=4.88 mV. Offset=>512 (Vref:2.50V)
  // DUE: For 3V3, FS and 10 bits, Delta_V=3.22 mV. Offset=>512 (Vref:1.65V)
  //.....

  Hysteresis = int(analogRead(A2) >> 2); // Read Hysteresis Pot2. (0V to 5V/4)
  Offset      = int(analogRead(A1) >> 2); // Read Off_set Pot1. (0V to 5V/4)

  V1 = Offset + 512 - Hysteresis; // upper-level of triggering
  V2 = Offset + 512 + Hysteresis; // lower-level of triggering

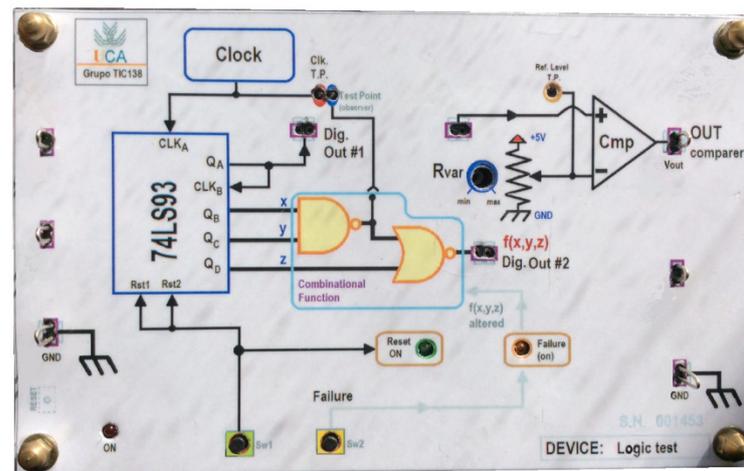
  // TRIGGER EMULATION
  // .....
  if (Input > V2 )
    { Send_DAC(973); } // Output Voh = 95% of Vcc (high level)
  else if (Input < V1 )
    { Send_DAC(51); } // Output Vol = 5% of Vcc (low level)
} // end loop

```

**Figure 8.** Code example for a functional ‘Schmitt-trigger’ emulation.

#### 4.2. Sequential Logic Circuit

A classical TTL version of a counter circuit (Figure 9) was emulated to check the performance when implementing a state machine or a table-based solution, pre-calculating and saving values in a LUT.



**Figure 9.** Graphical-cover for the digital emulation example.

The LUT option requires memory accesses that can slow down the procedure. In return, this method simplifies the programming of more complex combinational functions.

In this proposal, logic values were synthesized using the `digitalWrite()` and `digitalRead()` IDE instructions. This method increases the execution speed since it is only necessary to change the binary states in the output pins. Therefore, this activity is excellent for oscilloscope training since it requires fine synchronizing to visualize the waveforms correctly.

Direct square waveform synthesis involves a logic state alternated on the corresponding output pin. This frequency, in this case, is the maximum, since the software routine is now exclusively dedicated to the waveform synthesis and does not perform any additional processing.

In addition, a combinational function (following two counter output pins) has been implemented to increase interest and the conclusions of the measurements. Now, others functions have been included to implement the truth tables, and therefore, the effective output frequency decreases.

As a reference, the maximum frequencies values for both platforms considered can be seen in Table 1, which shows the results for a state-machine routine using logic functions and structures, and also a LUT-based method.

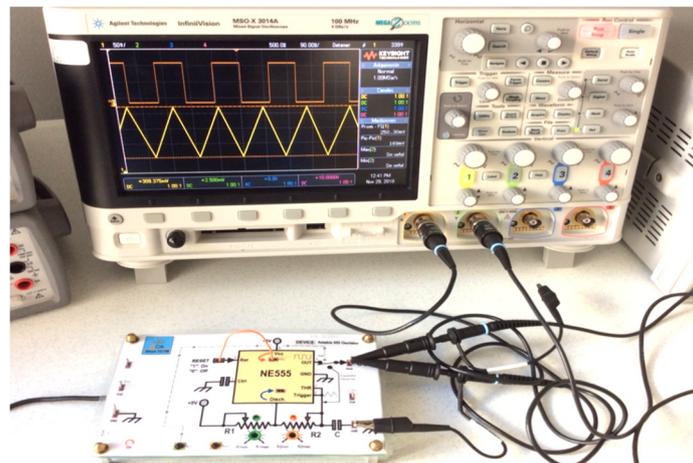
**Table 1.** Digital synthesis: maximum output frequency according to the type of platform.

Parameter	UNO	DUE
Square wave synthesis:		
- CLK output	125 kHz	4 MHz
State machine algorithm:		
- CLK output	19.23 kHz	51.28 kHz
- Q <sub>A</sub> output	9.59 kHz	25.97 kHz
LUT based algorithm:		
- CLK output	22.2 kHz	46.51 kHz
- Q <sub>A</sub> output	11.1 kHz	23.53 kHz

#### 4.3. Classical 555 Timer Emulation

We propose the simulation of the well-known timer NE555 (Signetics<sup>®</sup>), one of the most transcendental ICs in the history of electronic technology [34], implemented as an astable oscillator (Figure 10). For this case, the emulation routine should consider the equation that represents the period for the 555 astable-oscillator [35]:

$$T = \frac{(R_a + R_b)C}{1.44} + \frac{R_b C}{1.44} \quad (4)$$



**Figure 10.** The 555 astable oscillator emulation the oscilloscope shows the square output, and the capacitor charge-discharge (taking advantage of reference channel screen memory).

The code considers the main functions of the timer 555: the reset pin is simulated by keeping a low-level output permanently; the triangular waveform generated by the capacitor charge/discharge through  $(R_a + R_b)$  and  $R_b$  respectively, is simulated using the analog output (DAC), the output waveform can again include the  $V_{OL}$  and  $V_{OH}$ .

#### 4.4. Filter Implementation Example

In this case, the emulator implements a discrete RC or LC filter from its impulse response  $h(k)$ . The representative graphical-cover is shown in Figure 6c. To obtain the

filter response must be used the Equation (1). The program analyzes the input waveform samples and calculates the convolution, that is, the output signal.

However, in the frequency domain, this same RC/LC circuit can be analyzed as a first-order or second-order filters, respectively, where the coefficients depend on the values of the resistor, the inductor and the capacitor. In this way, it could be used the corresponding expression to an FIR filter (2).

The analysis of the LC circuit can be addressed by applying the Kirchoff's voltage law (KVL) along an RLC closed-network and thus obtain the expression [36]:

$$LC \frac{d^2 V_C(t)}{dt^2} + RC \frac{dV_C(t)}{dt} + V_C(T) = V_{in} \quad (5)$$

where  $V_C(t)$  represents the output voltage (low-pass filter) when the RLC circuit is excited from an input source  $V_{in}$ . For a discrete-time approach, this equation can be modified to achieve a difference equation referred to  $T_s$  sampling period:

$$V_O(n) = \frac{1}{1 + A + B} V_{in}(n) + \frac{A + 2B}{1 + A + B} V_O(n - 1) - \frac{B}{1 + A + B} V_O(n - 2) \quad (6)$$

with  $A = R \cdot C / T_s$  and  $B = L \cdot C / T_s$ , and  $T_s$  the sampling time. Equation (6) allows us to synthesize a general filter function that may to accommodate RLC, LC, and RC filters, depending on the  $R$ ,  $L$ , and  $C$  values. On the one hand, it is possible to annul the  $B$  value to obtain an RC group. In this case, the capacitor charge/discharge in an RC network will be emulated. The  $A$  value represents the product of resistance and capacitance real values. For this motive, it is easy to simulate the RC phenomenon implementing (6) as a MAC algorithm.

On the other hand, for the implemented simulation (Figure 6c) when  $R = 0$ , we obtain a second-order LC-filter, and the response uniquely depends on  $L$  and  $C$  parameters ( $A = 0$ ). This case is shown in Figure 11 emulating an  $R = 0 \Omega$ ,  $C = 68 \mu\text{F}$  and  $L = 100 \text{ mH}$  network. In the same way, an RC integrator may be synthesized for  $L = 0$  and consequently  $B = 0$ . Optimizing the algorithm, the possible minimum sampling time,  $T_{Smin}$  are shown in Table 2 and represent the value for each point ( $N = 1$ ). For  $N$  points, the maximum frequency will be obtained from the inverse  $1/(N \times T_{Smin})$ .

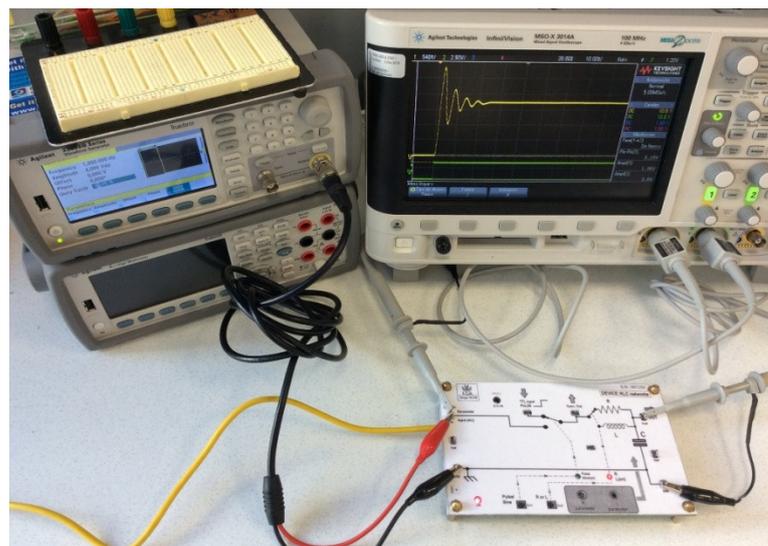


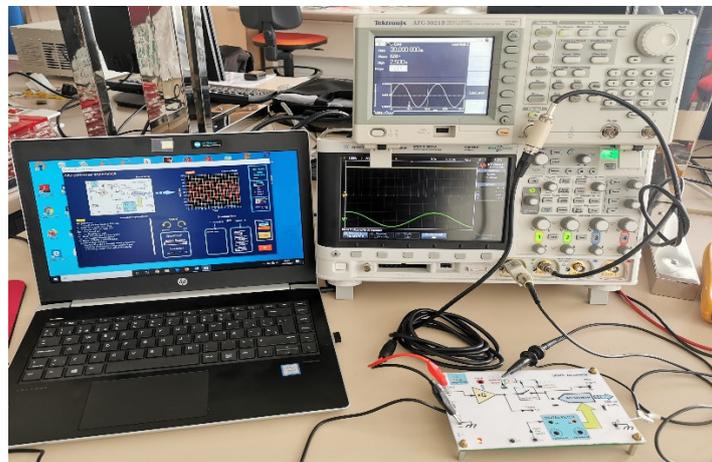
Figure 11. Second-order response implementing (implementing Equation (6)).

**Table 2.** FIR algorithms execution times according to the type of platform.

Parameter	UNO	DUE
LC filter algorithm:		
Min. sampling time $T_s$	264.5 $\mu\text{s}$	75.6 $\mu\text{s}$
RC filter algorithm:		
Min. sampling time $T_s$	222 $\mu\text{s}$	72 $\mu\text{s}$

#### 4.5. The Analog–Digital Converter and a PC Interactive Application for Intelligent Aid

To study the possibilities of interaction between a personal computer (PC) and the emulator card, we present a laboratory experience to check the process of the A/D conversion and perform a set of measurements with the oscilloscope (Figure 12). In this case, a specific C++ application using the LabWINDOWS<sup>TM</sup> software (from National Instrument<sup>TM</sup>) was programmed.

**Figure 12.** The interactive practice concerning the analog/digital converter in the laboratory.

It was also necessary to design a graphical-cover to present the idea of the experiment. This lab practice takes advantage of the embedded A/D converter into the Arduino card microcontroller to understand the sampling frequency concepts, the resolution effect (the binary word length of the converter), and the basic implementation of the digital filters (applying (2) or (3) equations, for low-order stages).

The PC application allows two support levels. In the first mode (Normal), the software reports instructions sequentially to the students, actuating as an interactive script to suggest when to execute the different tasks and measurements (Figure 13). In this mode, the computer presents the values which must be measured and evaluated by the students (for instance, sampling frequency, pulse wide, etc.). For this option, students must complete the tasks and obtain acceptable reading values according to the measurable parameters and the tolerances of the measurement procedure. In this case, the computer only acts as the task sequencer: students must execute the script, performing the indicated readings, but they must annotate and interpret the corresponding results.

In the other mode (interactive), the PC program could randomly modify the emulator card parameters via the Arduino Programming USB. In this case, the students also should execute the measurement sequence, but now they will enter in the computer every achieved result (reads, calculations, conclusions, etc.).

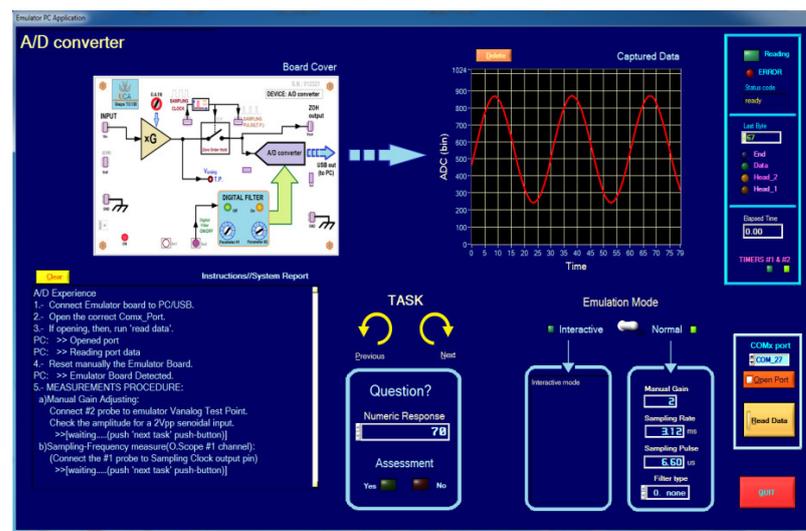


Figure 13. PC application screen cover for the interactive A/D converter practice.

Once the measurement results are entered, the system (which knows the previously loaded parameters and consequent solutions) will validate the obtained readings. If the results are incorrect, the computer could also order a new reading set. In addition, it is automatically possible to evaluate conclusions, applying a Boolean-response method.

## 5. Future Works and Improvements

Regarding the improvement of performance, the Arduino DUE card offers more advantages respect to the previous Arduino UNO version. Although the standard connector of both Arduino cards is similar and compatible with the presented emulation card, the ARM SAM3X-based card features a higher number of digital input/output pins, as well as two on-chip built-in DACs. Therefore, a redesign of the emulator card is necessary to take advantage of all ARM type characteristics. However, it is not possible to standardize the specific Arduino DUE connector: the connection to Arduino UNO would be irreversible.

An additional possible choice is to use an FPGA development card as the emulator engine. This option permits us a higher simulation speed range. In this sense, one possible via of study is the use of the Arduino-Compatible FPGA card XLR8™ or SNOR20M16V3 (joined to the SnōMākr™ shield) presented by Alorium Technology, LLC [37]. In addition, the DuePrologic USB-FPGA from Earth People Technology [38] that offers an Arduino-Due shield layout, or the Spartan Edge Accelerator Board—Arduino FPGA Shield with ESP32 from Seeed [39]. In this case, it is possible to obtain a higher execution speed for more complex and genuine simulations. In addition, it is feasible to develop a solid cover containing other additional hardware to complement a more realistic view of the experience. For example, we can suggest an experience based on a PID controller tuning, where the work surface would include an electric motor, mechanical load, and speed sensor. Similarly, any added hardware (PCB or OEM module) could increase the features available on the emulator platform and the Arduino core.

The association of the computer, the software, and the emulator card provides the necessary support to develop courseware dedicated to electronic engineering. The writing of specific software, which able real-time re-program the emulation card, allows to improving the experience and to adapt the measurement sequence to the knowledge and skill levels of each student.

This interactive learning technique can be enhanced by applying some artificial intelligence techniques (IA) as the Markov chain. In this scenario, the computer could lead the experience depending on the knowledge level, wrong answers and conclusions from any student.

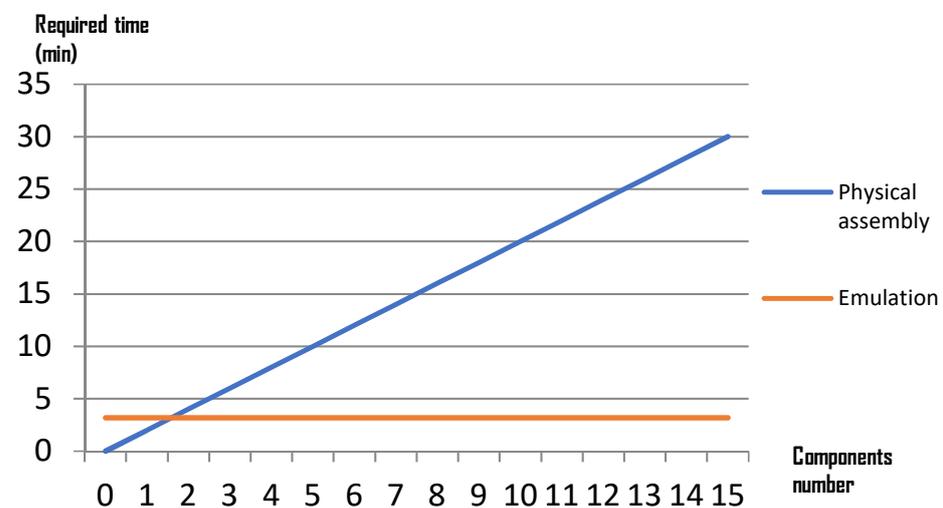
## 6. Conclusions

This work presents the emulation equipment and programming methods to address the real-time simulation of electronic circuits (DUTs) to test in bench. The possible uses that offer this emulator are varied. We highlight especially its utilization for electronic instrumentation training. The performed tests with a set of emulated varied circuits (Schmitt-trigger, NE555 timer, sequential logic circuits, Nyquist and the AD/DA conversion, low-order digital filter, common emitter amplifier, RFID principles, quasi-random source and serial data generator for bit error rate (BER) test, etc.) have validated that this platform is a very powerful and versatile aid to learn and practice laboratory instrumentation.

Although these essays have been carried out with boards UNO and DUE from Arduino, we want to emphasize that they could be carried out with any other type of card Arduino-Shield compatible (based on a microprocessor, microcontrollers, or FPGA development boards).

As an assay, a comparative laboratory practice was carried out around the 555-timer integrated circuit, in its hardware (classic lab practice) and then its corresponding emulated option, and obtained an important result about times: with a fixed time minimum for preparing the emulator card and for download the corresponding software, the emulated-practice option maintains a constant preparation time regardless of the complexity of the experience. In contrast, the preparation and assembly time of a physical device strongly depends on the complexity of the assembly.

Extrapolating the average times in carrying out one and another laboratory practice kind, and considering the number of components involved in the physical assembly, we determine a relationship of two components/min (required time for component selection, assembly, wiring, and inspection). On the other hand, the average time for preparing the emulator (including the software download) was 3.18 min. Using this relative result, it is possible to draw the graph shown in Figure 14.



**Figure 14.** Required extrapolated time comparing the physical assembly of the circuit (classical lab practice) case and the board preparation for an emulated lab practice case.

In addition, it is an open platform to address any other type of emulation of electrical circuits. For instance, the study of transmission lines, magnetic induction by emulating the coupling factor, cores materials, the number of turns of windings, as well as physical phenomena like ionizing radiation absorption versus density/thickness of materials, simulated with magnets, ferromagnetic plates, and Hall sensors, or the beta decay phenomenon.

The low cost of emulator and Arduino boards allows the acquisition of practical skills at student's own homes. They can take advantage of any free software that converts the PC sound card into an oscilloscope and a function generator.

**Author Contributions:** Conceptualization and validation, J.-M.G.-R., C.C.S., Á.Q.-O. and J.-A.L.-B.; methodology, Á.Q.-O.; software, J.-M.G.-R.; writing—original draft preparation, J.-M.G.-R.; writing—review and editing, C.C.S.; visualization and supervision, J.-A.L.-B.; funding acquisition, J.-M.G.-R. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was performed as an innovation and teaching improvement project and supported by grant SOL-201700083174-TRA from *Vicerrectorado de Recursos Docentes y de la Comunicación*, University of Cadiz.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. NAUGRA®, P. Code: CON-008-NE. India. Available online: <https://www.naugraexport.com/oscilloscope-trainer-oscilloscope-lab-kit-for-training-for-vocational-training-and-didactic-labs> (accessed on 20 June 2021).
2. NAUGRA®, P. Code: CON-024-NE. India. Available online: <https://www.naugraexport.com/digital-storage-oscilloscope-trainer-lab-demonstration-kit-for-vocational-training-and-didactic-labs> (accessed on 20 June 2021).
3. VELLEMAN®, Oscilloscope Tutor Board, France. Available online: <https://www.velleman.eu/products/view/?id=387510> (accessed on 20 June 2021).
4. SIGLENT®, STB-3Signal Test Board, The Netherlands. Available online: <https://www.siglent.eu/product/1142628/siglent-stb-3-oscilloscope-demo-training-board> (accessed on 20 June 2021).
5. RIGOL®, DK-DS600 Oscilloscope Trainer Board, USA. Available online: <https://www.rigolna.com/education/>; <https://www.rigol-uk.co.uk/product/rigol-ds6000-dk-demo-board/> (accessed on 20 June 2021).
6. GW INSTEK®, GDB-03 Oscilloscope Education and Training Kit, Taiwan. Available online: <https://www.gwinstek.com/en-global/products/detail/GDB-03> (accessed on 20 June 2021).
7. TEKTRONIX®, Learn Digital Oscilloscope Operation Using Arduino Board as DUT Signal Generator, USA. Available online: [https://download.tek.com/courseware/ST\\_Arduino\\_Labs\\_Package.zip](https://download.tek.com/courseware/ST_Arduino_Labs_Package.zip) (accessed on 26 June 2019).
8. Carullo, A.; Parvis, M.; Vallan, A. An Audio Card-Based Kit for Educational Purposes. *IEEE Trans. Instrum. Meas.* **2003**, *52*, 733–737. [CrossRef]
9. Potkonjak, V.; Gardner, M.; Pasi Mattila, V.; Guetl, C.; Petrović, V.M.; Jovanović, K. Virtual laboratories for education in science, technology, and engineering: A reviews. *Comput. Educ.* **2016**, *95*, 309–327. [CrossRef]
10. Grimaldi, D.; Rapuano, S. Hardware and software to design virtual laboratory for education in instrumentation and measurement. *Meas. J. Int. Meas. Confed.* **2009**, *42*, 485–493. [CrossRef]
11. Angrisani, L.; Bonavolontá, F.; D’Arco, M.; Liccardo, A. A Flexible Remote Laboratory with Programmable Device under Test. *Measurement*. **2020**, *156*, 1–15. [CrossRef]
12. Lavayssière, C.; Larroque, B.; Schröder, M.; Luthon, F. A Fully Open Source Remote Laboratory for Practical Learning. *Electronics* **2020**, *9*, 1832. [CrossRef]
13. McGregor, I. The Relation Between Simulation and Emulation. In Proceedings of the 2002 Winter Simulation Conference, San Diego, CA, USA, 8–11 December 2001; pp. 1683–1688. [CrossRef]
14. List of Arduino Boards and Compatible Systems (Wikipedia). Available online: [https://en.wikipedia.org/wiki/List\\_of\\_Arduino\\_boards\\_and\\_compatible\\_systems](https://en.wikipedia.org/wiki/List_of_Arduino_boards_and_compatible_systems) (accessed on 20 June 2021).
15. TSXPERS™. Arduino™ Compatible Compiler for LabVIEW, USA. Available online: <https://www.tsxperts.com/arduino-compatible-compiler-for-labview/> (accessed on 20 June 2021).
16. MATHWORKS®, Arduino Support from MATLAB. USA. Available online: <https://es.mathworks.com/hardware-support/arduino-matlab.html> (accessed on 20 June 2021).
17. Arduino Official Products list. From the Company Home Page. Available online: <https://www.arduino.cc/en/main/products> (accessed on 20 June 2021).
18. Atmel STUDIO 7®, Integrated Development Platform (IDP) for AVR® and SAM Microcontroller Applications from Microchip®. USA. Available online: <https://www.microchip.com/mplab/avr-support/atmel-studio-7> (accessed on 20 June 2021).
19. Smith, S.W. *Convolution and Properties of Convolution. The Scientist and Engineer’s Guide to Digital Signal Processing*, 2nd ed.; California Technical Publishing: San Diego, CA, USA, 1999; pp. 140–147. ISBN 0-9660176-7-6.
20. Tertel, P.; Hedrich, L. Real-time emulation of block-based analog circuits on an FPGA. *Integration* **2018**, *68*, 373–382. [CrossRef]
21. Ibrahim, D. *Practical Digital Signal Processing Using Microcontrollers*, 2nd ed.; Elektor Verlag International Media: Aachen, Germany, 2013; ISBN 978-1907920219.
22. Reay, D.S. *Digital Signal Processing Using the ARM® Cortex®-M4*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 2016; ISBN 978-1-118-85904-9.
23. Fernández, J.; Gemin, W.; Rivera, R.; Revuelta, M.; Kuzman, M.; Hidalgo, R. Digital filter design with Arduino DUE and Matlab. In Proceedings of the XVI Workshop on Information Processing and Control (RPIC), Cordoba, Argentina, 6–9 October 2015; pp. 1–6. Available online: <https://ieeexplore-ieee-org.bibezproxy.uca.es/document/7497060> (accessed on 20 June 2021). [CrossRef]
24. Molina-Cantero, A.J.; Castro-García, J.A.; Lebrato-Vázquez, C.; Gómez-González, I.M.; Merino-Monge, M. Real-Time Processing Library for Open-Source Hardware Biomedical Sensors. *Sensors* **2018**, *18*, 1033. [CrossRef] [PubMed]

25. Microchip, AVR Libc Reference Manual Modules (<math>h</math>: Mathematics). USA. Available online: [https://www.nongnu.org/avr-libc/user-manual/group\\_avr\\_math.html](https://www.nongnu.org/avr-libc/user-manual/group_avr_math.html) (accessed on 20 June 2021).
26. Microchip, AN-AVR223: Digital Filters with AVR. USA. Available online: <http://ww1.microchip.com/downloads/en/AppNotes/doc2527.pdf> (accessed on 20 June 2021).
27. Microchip, AN852: Implementing FIR and IIR Digital Filters Using PIC18 Microcontrollers. USA. Available online: <http://ww1.microchip.com/downloads/en/appnotes/00852a.pdf> (accessed on 20 June 2021).
28. Varshney, V.; Tiwari, M. Realization of an FIR filter using ATMEGA32 microcontroller. In Proceedings of the 2017 ICETCCT, Dehradun, India, 17–18 November 2017; pp. 1–4. Available online: <https://ieeexplore.ieee.org/document/8280325> (accessed on 20 June 2021). [CrossRef]
29. Fettweis, A. Wave digital filters: Theory and practice. *Proc. IEEE* **2018**, *74*, 270–327. [CrossRef]
30. Karjalainen, M.; Erkut, C. Digital Waveguides vs. Wave Digital Filters in physical modeling: Theoretical and computational aspects. In Proceedings of the EUROSIPCO, Vienne, Austria, 6–10 September 2004; pp. 289–292.
31. Davidek, V.; Antosova, M. Design and Implementation of Wave Digital Filters. *Radioengineering* **2001**, *34*, 11–14, ISSN 1210-2512 (Print); 1805-9600 (Online).
32. Wei, W.; Yen-Lung, C.; Yue, M.; Chien-Nan, J.L.; Jing-Yang, J.; Sudhakar, P.; Lei, H. Wave Digital Filter based Analog Circuit Emulation on FPGA. In Proceedings of the 2016 ISCAS, Montreal, QC, Canada, 22–25 May 2016; pp. 107–120. [CrossRef]
33. Mita, R.; Palumbo, G. Modeling of analog blocks by using standard hardware. *Analog Integr. Circuits Signal Process* **2006**, *48*, 107–120. [CrossRef]
34. Santo, B.R. 25 microchips that shook the world. *IEEE Spectrum*. **2009**, *46*, 34–43. [CrossRef]
35. Camenzind, H.R.; Kash, R.B. A low-voltage IC timer. *IEEE J. Solid-State Circuits* **1978**, *13*, 847–852. [CrossRef]
36. Agarwal, A.; Lang, J.H. Foundations of Analog and Digital Electronic Circuits. In *Morgan Kaufmann Publishers*; Elsevier: San Francisco, CA, USA, 2005; pp. 640–664.
37. Alorium Technology, LLC. XLR8™ Arduino-Compatible FPGA Development Board. Available online: <https://www.intel.com/content/www/us/en/programmable/solutions/partners/partner-profile/alorium-technology--llc/board/xlr8-max-10-development-board.html?wapkw=xlr> (accessed on 20 June 2021).
38. Earth People Technology. DuePrologic, FPGA Development Board (Arduino-Due Shield Layout). Available online: <https://earthpeopletechnology.com/products-page-2/modules/dueprologic> (accessed on 16 June 2021).
39. Seeed Studio. Spartan Edge Accelerator Board—Arduino FPGA Shield with ESP32. Available online: <https://www.seeedstudio.com/Spartan-Edge-Accelerator-Board-p-4261.html> (accessed on 16 June 2021).