11th International Conference on Electrical, Electronics and Computer Engineering (IcETRAN), Nis, 3-6 June 2024

Enhancement of Circuit Theory Course Using Low-Cost Software-Defined Instruments

Nikola Basta
School of Electrical Engineering
University of Belgrade
Belgrade, Serbia
nbasta@etf.bg.ac.rs
https://orcid.org/0000-0001-6141-7414

Nemanja Grujičić
School of Electrical Engineering
University of Belgrade
Belgrade, Serbia
gn200026d@student.etf.bg.ac.rs

Aleksa Madžarević
School of Electrical Engineering
University of Belgrade
Belgrade, Serbia
ma210399d@student.etf.bg.ac.rs

Milka Potrebić Ivaniš
School of Electrical Engineering
University of Belgrade
Belgrade, Serbia
milka_potrebic@etf.bg.ac.rs
https://orcid.org/ 0000-0002-4866-6608

***Abstract*— In this paper, we show use-case examples of the application of affordable programmable measurement hardware in circuit analysis as part of a graduate course in Electric-circuit theory. Besides basic measurement modes, some advanced software-enabled capabilities in experiment control, post-processing and data visualization are shown. In conclusion, we consider different setup realizations, as well as the advantages and limitations of such software-defined equipment.**

***Keywords*— Matlab, computerized instrumentation, oscilloscopes, network analyzers, filters**

I. Introduction

In this digital era, demand for quick acquisition, analysis, and exchange of measurement data has increased manyfold and *computerization* of measurement equipment has followed closely this trend. Even before the epidemic of coronavirus, the advancement of mobile, low-cost and affordable measurement equipment, both consumer-grade and industrial, was evident in recent years. Most of these instruments are based on some kind of processing unit, be it a simple microcontroller, a CPU, or an FPGA. The latter category had a particular impact on the versatility and price of the available equipment. Until recently, the features of the equipment's processing units were either completely inaccessible or only partially accessible to users. With the price decrease of FPGA and ARM chips, but also with the increase of their computing resources (memory and speed) instrumentation products with programmable analysis and post-processing features appeared. The intermediate step to this stage was so-called PC-based equipment, e.g. oscilloscopes and spectrum analyzers, where the instruments were relieved of the display unit and the data analysis. In the PC-based equipment, these two features completely migrated to the PC.

Besides the lower cost, other advantages of the PC-based, but even more of the software-defined instrumentation (SDI), are versatility in the creation and monitoring of tests, ease of integration into different development environments, and connectivity to wired or wireless computer networks, which allows for comfortably and remotely governed experiments. Due to all these advantages, SDI has penetrated both professional and consumer-grade markets.

If we consider oscilloscope-type instrumentation, the current price for professional SDI units, can range from several thousand up to few tens of thousands USD. Some mid- and lower-range oscilloscopes can be purchased for less than one thousand USD. An overview of PC-based oscilloscopes is given in Table I, where BW denotes bandwidth and f_s denotes sample rate.

TABLE I. EXAMPLES OF USB-BASED OSCILLOSCOPES

Price range [USD]	Company	Product	Signal features
≥10,000	Keysignt technologies [1]	P9243A	BW> 200 MHz f_s:>5 GS/s
1,000÷10,000	National Instruments [2]	NI 779970-01	BW> 50 MHz f_s:>100 MS/s
500÷1,000	RS [3]	RS-6254BC	BW> 70 MHz f_s:>1 GS/s
250÷500	OWON [4]	VDS6102	BW> 10 MHz f_s:>1 GS/s
50÷250	Hantek [5]	6004EU	BW> 70 MHz f_s:>1 GS/s

In this work, we consider a particular product at the lower end of the price range, namely, a PC-based platform from Analog Devices, ADALM2000. The platform is essentially a miniature software-defined laboratory and, among other features, offers:

- dual channel analog input (oscilloscope/multimeter), of 25 MHz bandwidth and sample rate of 100 MS/s;
- DC power supply from −5 to 0 V and from 0 to +5 V and 50 mA;
- function generator with amplitudes of ±5 V, 30 MHz bandwidth and 125 MS/s sampling rate;
- 16 digital I/O pins.

This unit is accompanied by *Scopy*, a proprietary software for control, data analysis and visualization. Besides from the fact that this module combines so many functions into a single board, the possibility to program the measurement process itself and the analysis steps, using bindings to common program languages, such as C/C++, Python, LabView or MATLAB [6], makes it very appealing for educational purposes.

ISBN 978-86-6200-001-9 ©2024 AM & ETRAN

This paper aims to demonstrate advantages of such platform by employing it in an integrated software environment that includes theoretical study of electric circuits, but also its experimental verification on the spot, without the need for the actual laboratory, setting up bulky equipment or transferring data from one device to another. In the following, we will showcase analysis of two example circuits that emerge in a typical second-year course in Electric-circuit theory.

II. Description of the Approach

In our approach, we rely on Xilinx FPGA-based platform ADALM2000 and the MATLAB environment and its toolboxes. It should be noted that other environments (e.g. C/C++ or Python) and other hardware platforms (Altera FPGA or ARM) could be used to build a similar setup. However, to the best of our knowledge, such alternative platforms from other vendors are still not available commercially, at least not in an affordable small-form product.

In the first step, the students are required to compute the theoretical time- or frequency-domain response of the defined circuit. The circuit can be analyzed using symbolic computation by hand or using MATLAB's Symbolic Math Toolbox, which is considered in this work. The Toolbox can be used in a regular MATLAB script, bus also in the so-called Live Script format, which is convenient for taking digital and interactive notes [7].

In order to determine the time-domain (TD) response of a linear time-invariant (LTI) system symbolically, in classic circuit theory, we can take two paths: (1) solve system of linear first-order ordinary differential equations (ODE) or (2) find the response in the Laplace domain and perform the inverse Laplace transform. Obviously, if one chooses the second path, the frequency-domain (FD) response is derived easily by substituting the Laplace transform variable s with $j\omega$ [8]. These procedures will be presented in more detail in the following section.

Once the desired response is computed symbolically, with general symbols for excitation, gain, resistances, capacitances, inductances, etc., we can inspect the circuit's response numerically, by substituting symbols with different numerical values, i.e. parameters. This parametric analysis is very quick and effective with a proper analytic model of the circuit.

However, additional numerical verification can also be performed by means of simulation using a SPICE-like tool. In this work, again, for the purpose of an integrated seamless environment, we used Simulink, namely its Simscape [9] component (libraries *Foundation Library* and *Electrical*), which allows for TD causal simulation, but also FD response evaluation of different multi-physics systems, including electric circuits. Since the block-based Simulink and code-based MATLAB are part of the same software family and may establish direct communication, it was only natural to unify the theoretical and experimental analysis in this way.

Theoretical TD and FD responses, obtained either via symbolic computation or simulation, can be saved in one of the convenient formats (e.g. .csv or native .mat). For the final verification, the theoretical results can be compared directly to the experimental data, all within the same script.

To outline our measurement and analysis setup, its architecture is given in Fig. 1. The student's laptop is connected via USB to the ADALM2000 module, and the module requires 8 wires to connect to the breadboard with the analyzed circuit. Two wires are required for the power supply (if needed), one for the excitation signal, another one for the reference node (GND), and four additional wires are required for recording excitation and output voltage signals (CH1 and CH2). The serial bus enables MATLAB to control and read data from the module, using the Analog Devices' *libm2k* library [10]. The module can also be used with an additional breakout board that allows for the use of real oscilloscope probes, instead of plain jumper wires, as shown in Fig. 2.

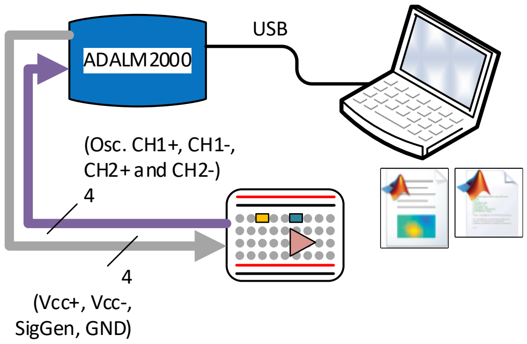

Fig. 1. Architecture of measurement and analysis setup. Through USB, MATLAB scripting can enable triggering and control of the measurement process, as well as convenient processing and visualization of measurement data through commands from the dedicated libm2k library. Both regular scripts (.mat) and Live Scripts (.mlx) can be used.

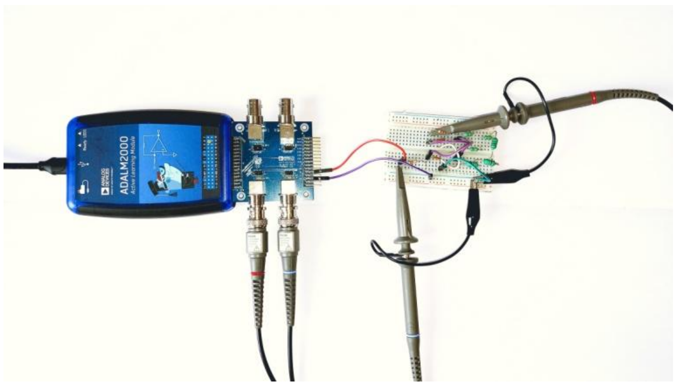

Fig. 2. Photography of the measurement setup. The circuit is assembled on a breadboard and the ADALM2000 is controlled via the USB interface on the left. Professional probes are attached via a breakout interface board.

III. Examples of Experimental Extensions to Typical Theoretical Analysis in Circuit Theory Courses

In this section, we will describe the analysis procedure on two example circuits. As described in Section II, we will start with theoretical analysis and verify it through simulation and measurement. The observables are output voltages when the circuit is excited by canonical sources – approximations of Dirac's delta function in TD analysis and sine signal of variable frequency and constant amplitude in FD analysis.

A. *Example 1: Time-Domain Response of a Passive Low-Pass Filter*

In the first example, we consider a low-pass filter (LPF), given in Fig. 3, and its response in time domain. The filter is fully described with parameters $R_1 = R_2 = 1\,\mathrm{k\Omega}$, $L = 33\,\mathrm{mH}$

and $C_1 = C_2 = 68\,\mathrm{nF}$. In the theoretical consideration, we will take the first of the two solution paths explained in the previous section – namely, direct solving of the system of ODEs that describe the circuit.

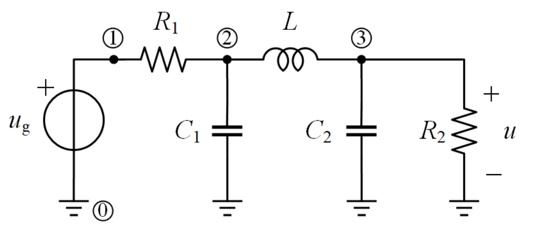

Fig. 3. Passive third-order Chebyshev low-pass filter.

The circuit in Fig. 4 has three voltage nodes, one reference node, and three dynamic elements (L, C_1 and C_2), which are associated to the three system states. Let us apply Kirchoff's current and voltage laws (KCL and KVL) directly in MATLAB Live Script:

```
syms ug iL uC1 uC2 DiL DuC1 DuC2 R1 R2 R L C1 C2 C t T
assume([ R1 R2 R L C1 C2 C t T], 'positive');
% KCL and KVL
eqs = [(uC1-ug)/R1 + C1*DuC1 + iL == 0;...
      -iL + C2*DuC2 + uC2/R2 == 0;...
       uC1-uC2 == L*DiL]
eqs =
```

$$\begin{pmatrix} \mathrm{iL} + \dfrac{\mathrm{uC}_1 - \mathrm{ug}}{R_1} + C_1\,\mathrm{DuC}_1 = 0 \\ \dfrac{\mathrm{uC}_2}{R_2} - \mathrm{iL} + C_2\,\mathrm{DuC}_2 = 0 \\ \mathrm{uC}_1 - \mathrm{uC}_2 = \mathrm{DiL}\,L \end{pmatrix}$$

where $D = \partial/\partial t$ represents the differential operator but it is initially considered as algebraic multiplicative magnitude. In Live Script, the output is highlighted in white. The newly formed algebraic symbols Du_{C1}, Du_{C2}, and Di_L will be substituted with actual derivatives, using command `subs` (substitute a set of variables with new values). By defining proper TD functions for the states and source and by employing MATLAB commands `solve()` (solve system of linear algebraic equations for the first derivatives) and `diff()` (differentiate), we arrive quickly at the known system of linear ODEs, i.e. the Cauchy's problem with constant coefficients [11]. In this example, we chose to observe impulse response, thus the voltage generator u_g is substituted with Dirac's delta function, virtually shifted in time (by $T = 0^+$ s), thus imposing a causal source, $u_g = \delta(t)$. In order to solve the system, we use the `dsolve()` command (solve system of linear ODEs), with initial conditions in the argument list:

```
% ODEs in Cauchy's form
vars = [DiL DuC1 DuC2];
Sol = solve(eqs, vars);

syms iL(t) uC1(t) uC2(t)
odes = subs([diff(iL) == Sol.DiL;...
             diff(uC1) == Sol.DuC1; ...
             diff(uC2) == Sol.DuC2],...
             [iL uC1 uC2 R1 R2 C1 C2 ug],...
             [iL(t) uC1(t) uC2(t) R R C C dirac(t-T)])
odes(t) =
```

$$\begin{pmatrix} \dfrac{\partial}{\partial t}\,\mathrm{iL}(t) = \dfrac{\mathrm{uC}_1(t) - \mathrm{uC}_2(t)}{L} \\ \dfrac{\partial}{\partial t}\,\mathrm{uC}_1(t) = -\dfrac{\mathrm{uC}_1(t) - \delta(T-t) + R\,\mathrm{iL}(t)}{C\,R} \\ \dfrac{\partial}{\partial t}\,\mathrm{uC}_2(t) = -\dfrac{\mathrm{uC}_2(t) - R\,\mathrm{iL}(t)}{C\,R} \end{pmatrix}$$

```
% Time domain response
conds = [iL(0)==0, uC1(0)==0, uC2(0)==0];
Resp = dsolve(odes, conds);
g_v3(t) = simplify(rewrite(Resp.uC2, 'heaviside'))
g_v3(t) =
```

$$\frac{e^{\frac{T-t}{C\,R}}\,\sigma_1}{4\,C\,R} - \frac{e^{-\frac{(T-t)\,(\sigma_2-\sqrt{L})}{2\,C\,\sqrt{L}\,R}}\,(\sigma_2-\sqrt{L})\,\sigma_1}{8\,C\,R\,\sigma_2} - \frac{e^{\frac{(\sigma_2+\sqrt{L})\,(T-t)}{2\,C\,\sqrt{L}\,R}}\,(\sigma_2+\sqrt{L})\,\sigma_1}{8\,C\,R\,\sigma_2}$$

where

$$\sigma_1 = \delta_{T-t,0} - 2\,\mathrm{heaviside}(T-t) + 2$$

$$\sigma_2 = \sqrt{L - 8\,C\,R^2}$$

The exact impulse response `g_v3(t)` may be obtained by substituting the symbols with numerical values:

```
values = [R == 1e3, L == 33e-3, C == 68e-9, T == 0];
subs(g_v3(t), lhs(values), rhs(values));
```

The graph of the normalized output voltage is shown in the blue line in Fig. 4. We can observe that the third-order filter has a damped oscillating impulse response.

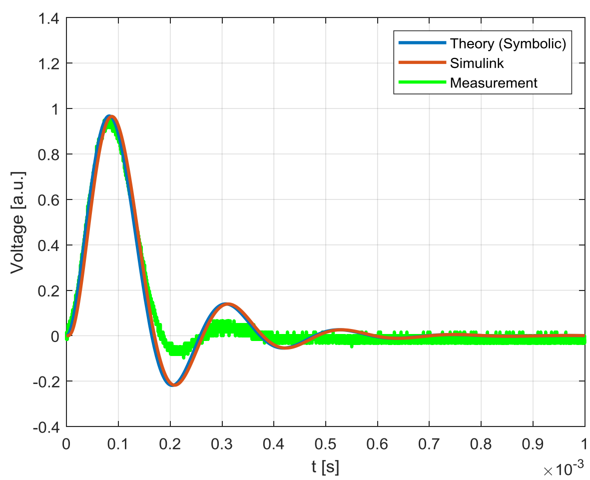

Fig. 4. Comparison of normalized time-domain responses for the analytical expression (theory), simulation and measurement on ADALM2000.

The first verification of the obtained analytic expressions is performed by simulating the circuit in Simulink. The Simulink/Simscape model that includes components of the *physical* system (meaning the electrical circuit) and *pure* Simulink components (meaning mathematical sources and sinks) is given in Fig. 5. In order to quickly realize impulse response, the source is defined as `Repeating Sequence` of very short trapezoidal (or other piecewise-defined) pulses, the period of which is much larger than then the observation interval (i.e. *Stop Time* in Simulink), thus simulating a single pulse. Moreover, the integral of this pulse over time (i.e. the surface below the pulse function) must be equal to one so that it agrees with the main property of Dirac's delta function. The source and output voltage signals are recorded with the `To Workspace` blocks (Fig. 5.).

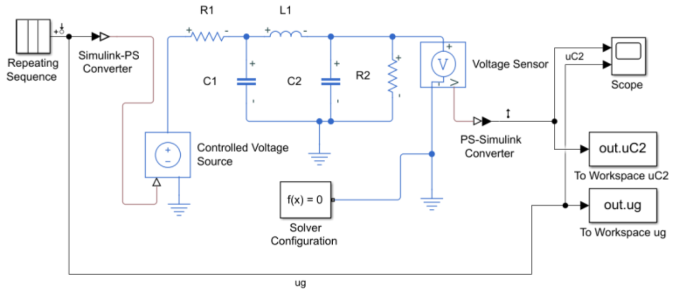

Fig. 5. Simulink realization of passive third-order LPF from Fig. 1. Voltages u_g and u_{C2} are defined as input and output ports for Simulink's system-level simulation.

The simulation can be called directly from a script using the following commands:

```
% Simulink simulation
simtemp = sim('LPF.slx');
time = simtemp.tout;
uC2Sim = simtemp.uC2;
ugSim = simtemp.ug;
```

where `sim()` runs the Simulink simulation on the file named `'LPF.slx'`, and the variables `time`, `uC2Sim` and `ugSim` are the time vector, response and source voltages extracted from the simulation, respectively. The simulated impulse response is given in Fig. 4 in red. Note that the result matches almost perfectly to our theoretical prediction (blue line).

Additional verification is performed by assembling the real-life LPF circuit on a breadboard and configuring the ADALM2000 for the oscilloscope function. In a MATLAB function, we can define a similar pulse stimulus as in the Simulink simulation and place it in a buffer variable, use it to excite the circuit and, finally, capture the output voltage in a variable or a file. Such function would typically consist of the following steps [12]:

(1) initialization of a *libm2k* object,

```
import clib.libm2k.libm2k.*
m2k = context.m2kOpen();
```

(2) calibration of ADCs and DACs,

```
m2k.calibrateADC();
m2k.calibrateDAC();
```

(3) creating analog input, analog output and trigger objects,

```
ain = m2k.getAnalogIn();
aout = m2k.getAnalogOut();
trig = ain.getTrigger();
```

(4) enabling oscilloscope analog input channels, creating channel objects, and setting up their properties (sample rate and range),

```
ain.enableChannel(0,true);
ain.enableChannel(1,true);
ain.setSampleRate(sampleRateIn);

c1 = analog.ANALOG_IN_CHANNEL.ANALOG_IN_CHANNEL_1;
c2 = analog.ANALOG_IN_CHANNEL.ANALOG_IN_CHANNEL_2;
ain.setRange(c1,-5,5)
ain.setRange(c2,-5,5)
```

(5) setting up analog output (stimulus generator) object and periodically pushing the stimulus samples,

```
aout.setSampleRate(0,sampleRateOut);
aout.enableChannel(0,true);
aout.setCyclic(true); % periodic excitation
aout.push(0,buffer); % output channel 0 is excitation
```

(6) measurement of input and output voltages in time domain (samples of the two input channels are saved interleaved in `data` variable),

```
% Acquisition
data = ...
ain.getSamplesInterleaved_matlab(2*Nsamples).double;

% Separate vectors for both channels:
data_ch1 = data(1:2:end); % excitation
data_ch2 = data(2:2:end); % response
```

where `Nsamples` is the number of acquired samples. In the acquisition step we use double the number since we record interleaved both excitation and response voltages.

The elaborated steps enable us to record two data sets – one for the measurement of the generator voltage, in our case, a short rectangular pulse of 5 V amplitude and 20 µs duration, and the other for the output voltage, v_3. The chosen sample rates for the input and output signals are 10 MHz and 7.5 MHz, respectively (variables `sampleRateIn` and `sampleRateOut`). This choice has proven to be optimal for the recording of the impulse response, for the given LPF. Note that in this TD experiment, the DAC and ADC settings are critical. In order to mimic the Dirac delta – the duration of the source pulse must be short with respect the circuit's time constant, but also long enough so that the DAC can convert it faithfully.

The measured result shown in green in Fig. 4 and is normalized to match the peak-to-peak outputs of the theoretical and simulated curves. We see that there is some discrepancy in the measurement data with respect to theory and simulation. The excess damping and a slight time shift can be explained by the fact that the theoretical and simulated models consider ideal inductors, resistors and capacitors, thus disregarding their manufacturing tolerances and parasitic components. The presence of noise is a consequence of ADALM2000 amplitude limitation to ±5V power supply and of the fact that the theoretical impulse-response energy is much lower than the source-pulse energy.

B. *Frequency-Domain Response of an Active Band-Pass filter*

In our second example, we consider the frequency-domain response of an active second-order band-pass filter, the schematic of which is shown in Fig. 6. The component values are: $R_1 = 220\,\Omega$, $R_2 = 1\,\mathrm{k}\Omega$, and $C_1 = C_2 = 0.1\,\mu\mathrm{F}$. In our theoretical analysis let us first assume an ideal operational amplifier (op-amp) that operates in a stable and linear regime.

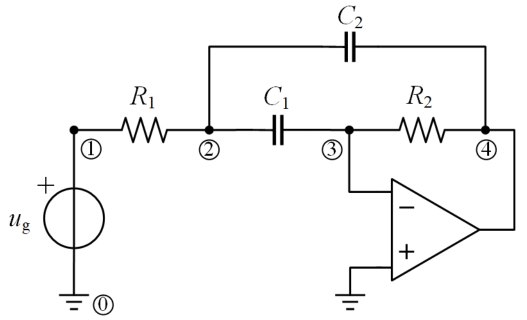

Fig. 6. Active second-order band-pass filter with an ideal operational amplifier.

Let us now compute symbolically the transfer function of the output voltage with respect to the input stimulus in the Laplace domain, $H(s) = V_4(s)/U_g(s)$. This way we only need to solve a system of algebraic equations, using the MATLAB command `solve()`. We can approach this problem with modified nodal analysis (MNA), starting with KCL expressions for each of the four nodes and solving for all of their potentials:

```
syms R1 R2 C1 C2 V1 V2 V3 V4 Ug s w
assume([R1 R2 C1 C2, w], 'positive');

% Transfer function
eqs = [V1 == Ug;...
      (V2-V1)/R1 + s*C1*(V2-V3) + s*C2*(V2-V4) == 0;...
       s*C1*(V3-V2) + (V3-V4)/R2 == 0;...
       V3 == 0]
```

eqs =

$$\begin{pmatrix} V_1 = \mathrm{Ug} \\ C_1\,s\,(V_2 - V_3) - \dfrac{V_1 - V_2}{R_1} + C_2\,s\,(V_2 - V_4) = 0 \\ \dfrac{V_3 - V_4}{R_2} - C_1\,s\,(V_2 - V_3) = 0 \\ V_3 = 0 \end{pmatrix}$$

```
vars = [V1 V2 V3 V4];
Sol = solve(eqs, vars);

H(s) = Sol.V4/Ug
```

H(s) =

$$-\frac{C_1\,R_2\,s}{C_1\,R_1\,s + C_2\,R_1\,s + C_1\,C_2\,R_1\,R_2\,s^2 + 1}$$

In order to get amplitude and phase FD response we need to substitute Laplace variable s with $j\omega$ and employ `abs()` and `angle()` commands. Furthermore, numerical evaluation of the response is obtained by substituting the resistances and capacitances with actual values, using command `subs`:

```
% Amplitude and phase
A(w) = abs(Hw(w));
values = [R1 == 220, R2 == 1000,...
            C1 == 0.1e-6, C2 == 0.1e-6];
Anum(w) = subs(A(w), lhs(values), rhs(values));
phinum(w) = angle(subs(Hw(w), ...
         lhs(values), rhs(values)));
```

where `w` is the frequency in rad/s. The theoretical FD response is given in blue in Fig. 7.

A simulated FD response of the ideal BPF circuit can be performed in TD based Simulink via *model linearization* and extraction of its parameters, namely the **A**, **B**, **C**, and **D** matrices of the state-space model [13]. However, since the components are ideal, the linearization will lead us to exactly the same transfer function and FD response as foreseen by theoretical expression, since the linearized model is based on the same differential/algebraic equations as our MNA analysis.

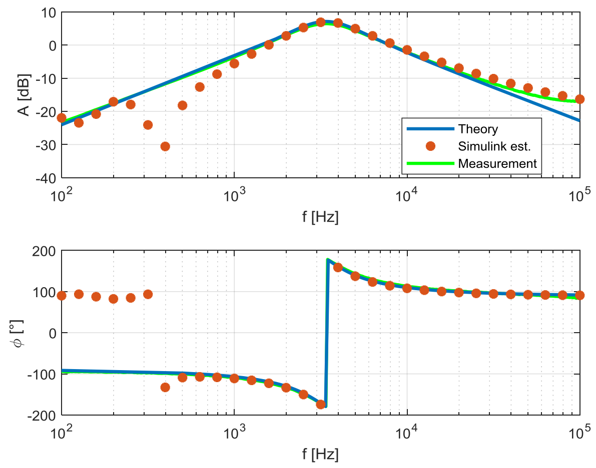

Fig. 7. Comparison of amplitude and phase frequency-domain responses according to theory, Simulink estimation, and measurement on ADALM2000.

In order to have a slightly more realistic simulation, the ideal op-amp may be replaced by a parameterized Simcape model or an imported SPICE-amp model. In our experiment, we used a parameterized Simscape model of a band-limited op-amp taking into consideration certain catalogue properties of the integrated circuit µA741CP [14] (gain, input and output resistance, bandwidth, and slew rate), which we intend to use in the experiment. Therefore, it is suggested to use TD simulation in estimation of the frequency response by feeding the circuit with a series of sine signals of different frequencies. This can be easily performed in Simulink/Simscape with the built-in function for steady-state FD estimation, `frestimate()`. The simulation model is given in Fig. 8 and the estimation result is shown in red markers in Fig. 7.

Obviously, the FD response estimation of the band-limited model is sensitive to the choice of amplitudes, frequencies, settling periods, sampling frequencies, etc., which is why we can sometimes observe errors, as in the range of 300 ÷ 900 Hz. Optimization of the sine-stream excitations is out of the scope of this work.

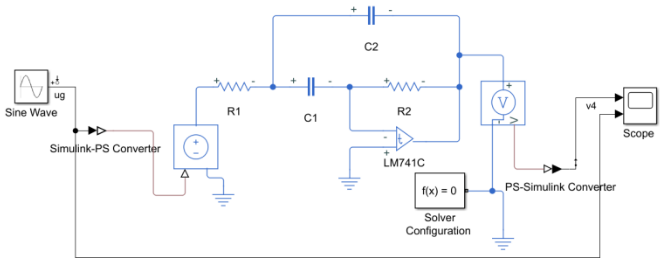

Fig. 8. Simulink realization of active second-order BPF from Fig. 6. Voltages u_g and v_4 are defined as input and output ports for Simulink's system-level simulation.

Similar to our Simulink estimation of the FD response, in the experiment, we can define a series of sine excitations of particular frequencies and durations, and feed them to our assembled circuit, just like we did in the TD measurement of the LPF. The TD responses to sine stimuli are then recorded and used as input in the amplitude- and phase-estimation step, which is performed according to the process described in [15]. The measurement results are also shown in Fig. 7, in green color. We can observe that the experiment gives the expected BPF response, in a close accordance with theory. The discrepancy of the amplitude response at the higher frequencies can be explained by the non-linear effects that cannot be included in the purely analytical model.

IV. Discussion and Conclusion

Pure theoretical consideration of electric-circuit theory can be quite extensive for second-year students. Usually, in electric-circuit courses, we introduce the basics of LTI systems. The supporting mathematics behind the definitions and derivations is relatively cumbersome, when observed in its entirety. A hybrid theoretical-experimental approach, including numerical and physical experiments, can offer new insights for young undergraduates.

Miniaturization and computerization of the measurement instrumentation allow for diversity in the analysis of electric circuits within the circuit-theory frame. Software-defined instrumentation platforms allow for different levels of students' engagement adapted to their experience: from basic voltage measurements to playing around with sampling theorem, algorithms, data buffers, memory and other CPU resources, etc.

Although affordable platforms like ADALM2000 have certain technical limitations in comparison to professional laboratory equipment, they offer a plethora of opportunities for lab and homework assignments, where, besides performing basic TD and FD measurements, the students can really take part in defining test procedures, thus employing, but also broadening their knowledge on math, programming, signal processing and general electrical engineering.

In order to further extend the proposed practical approach, students may be assigned to refine their theoretical circuit model so that the response matches the measurements better, e.g. including parasitic components and tolerances (e.g. obtained using an LCR-meter or a multi-meter). Additionally, a parameter-sweep analysis can aid the students in understanding the system sensitivity and impact of each element on the response. Finally, instead of analysis, students may be introduced to a basic engineering process, by assigning them to design a simple circuit that fulfils a set of moderate requirements, e.g. a lumped-element filter.

Note that the aforementioned tasks can also be performed on other standard or SDI equipment. However, with the low-cost, small-factor, and mobile SDI platforms, the learning goals seem to be achievable more quickly, saving time and workspace, with much flexibility that inspires both students and teachers.

Acknowledgment

This work was financially supported by the Ministry of Science,Technological Development and Innovation of the Republic of Serbia under contract number: 451-03-65/2024-03/200103.

References

[1] https://www.keysight.com/
[2] https://www.ni.com/
[3] https://uk.rs-online.com/web/
[4] https://www.owon.com.hk/
[5] https://www.hantek.com/
[6] https://www.analog.com/en/resources/evaluation-hardware-and-software/evaluation-boards-kits/adalm2000.html
[7] https://www.mathworks.com/help/matlab/matlab_prog/what-is-a-live-script-or-function.html
[8] W. M. Siebert, *Circuit, Signals and Systems*, Cambridge, USA: MIT Press, 1985.
[9] https://www.mathworks.com/products/simscape.html
[10] https://analogdevicesinc.github.io/libm2k/
[11] L. Hörmander, *Partial Differential Operators*, Berlin, Germany: Springer, 1963.
[12] https://github.com/analogdevicesinc/libm2k-matlab
[13] C.-T. Chen, *Linear System Theory and Design*, New York, USA: Oxford University Press, 1999.
[14] https://www.ti.com/lit/ds/symlink/ua741.pdf
[15] S. Ye, J. Sun and E. Aboutanios, "On the Estimation of the Parameters of a Real Sinusoid in Noise," in *IEEE Signal Processing Letters*, vol. 24, no. 5, pp. 638-642, May 2017.