Add Some SPICE to Distance and Blended Learning

Adding SPICE simulation to remote labs for a complete learning experience which integrates theory, test and verification

Ingo Knoblich[1] and Philipp Krauss[1]
[1]National Instruments, Munich, Germany

***Abstract*— In this case study, we utilize the SPICE simulator NI Multisim and the graphical programming environment NI LabVIEW to build a remote lab. The student will be able to simulate a circuit, save the characteristic data as a reference and then test and measure against a real circuit to understand how real-world measurements compare to a simulation model. This real circuit is built upon a prototyping breadboard.**

***Index Terms*— SPICE, Circuit simulation, data acquisition, Education**

I. INTRODUCTION

EE students today, are being prepared for the laboratory experience by learning theory and then building on this knowledge, first through circuit simulation, and then by physically building and testing these circuits. However, the true value of this experience is at that point of realization, when the student truly understands the difference between theory, and the real world behavior of circuits. Professors, instructors and professionals agree that the true meaning of engineering, is in that ability to design, predict and understand real world effects and apply this knowledge to innovative new projects .

In this case study, we utilize the SPICE simulator NI Multisim and the graphical programming environment NI LabVIEW to build a remote lab. The student will be able to simulate a circuit, save the characteristic data as a reference and then test and measure against a real circuit to understand how real-world measurements compare to a simulation model. This real circuit is built upon a prototyping breadboard.

Within a visual interface, the student will understand the difference between real world and simulation providing an appropriate opportunity to use different circuit variants in simulation to choose a circuit adjustment that will eliminate the unwanted real world effects. With NI LabVIEW we will also create a remote panel function that will be used to publish the lab experiment to the internet (as seen in Figure 1).

II. SPICE SIMULATION

[1] Simulation Program with Integrated Circuit Emphasis, or SPICE, has been used for over thirty years. The original implementation of SPICE was developed at the University of California Berkeley campus in the late 1960s. The first widely used version of SPICE was announced in Waterloo, Canada in 1973. Shortly

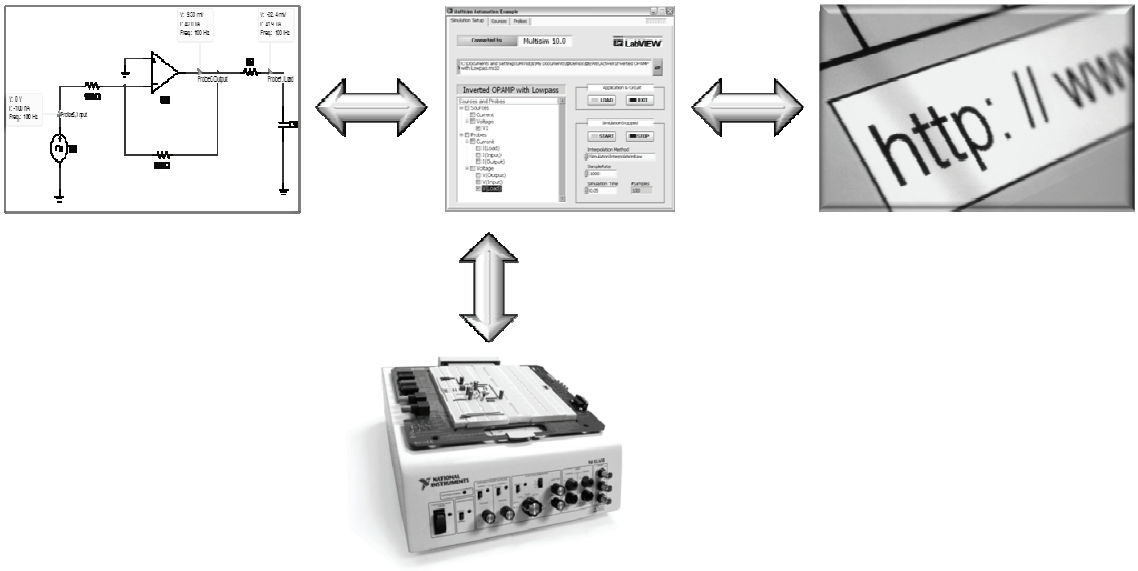

Figure 1. Concept of the remote laboratory

thereafter SPICE was adopted by nearly all major engineering institutions throughout North America. SPICE has evolved into the academic and industry standard for analog and mixed-mode circuit simulation.

SPICE is a computer simulation and modeling program used by engineers to mathematically predict the behavior of electronic circuits. Developed at the University of California at Berkeley, SPICE can be used to simulate circuits of almost all complexities. However, SPICE is generally used to predict the behavior of low to mid frequency (DC to around 100MHz) circuits.

SPICE has the ability to simulate components ranging from the most basic passive elements such as resistors and capacitors to sophisticated semiconductor devices such as MESFETs and MOSFETs. Using these intrinsic components as the basic building blocks for larger models, designers and chip manufacturers have been able to define a truly vast and diverse number of SPICE models. Most commercially available simulators include more than 15,000 different components.

The quality of SPICE models can vary, and not all SPICE models are applicable to every application. It is important to consider this when using the models supplied with a SPICE simulation package. Using a SPICE model inappropriately can lead to inaccurate results, or even generate an error in some circumstances. One of the most common errors made by even seasoned engineers is confusing a SPICE model with a PSPICE model. PSPICE is a commercially available program that uses proprietary languages to define components and models.

A circuit must be presented to SPICE in the form of a netlist. The netlist is a text description of all circuit elements such as transistors and capacitors, and their corresponding connections. Modern schematic capture and simulation tools such as Multisim allow users to draw circuit schematics in a user-friendly environment, and automatically translate the circuit diagrams into netlists. Consider as an example the simple voltage divider given as a SPICE netlist and as schematic (Figure 2).

```
* Voltage Divider
vV1 1 0 12
rR1 1 2 1000
rR2 2 0 2000
.OP * perform a DC operating point analysis
.END
```

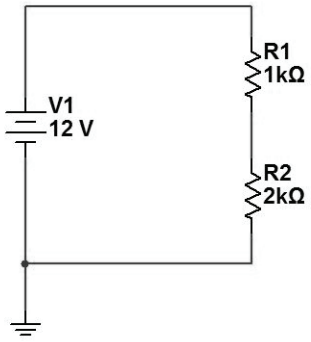

Figure 2. Simple voltage divider

A. NI Multisim

SPICE by itself can be used as a command line, or text-based simulation tool. However, to make it feasible for a broad range of users to utilize the power of SPICE, a graphical user interfaces that wraps around the original engine is highly advantageous. The user interface allows for a more intuitive and interactive approach to SPICE simulation. Although different graphical interfaces, have different approaches to their SPICE simulation environments, most of them offer access to a component database, an integrated schematic capture tool and a way of displaying the simulation results.

National Instruments Multisim, formally known as Electronics Workbench, is the schematic capture and simulation environment used for this case study of this remote lab. Based on professional printed circuit board (PCB) design tools, NI Multisim was designed with the needs of educators in mind, and aids student understanding through features such as integrated quizzes, virtual and rated components, password protected component faults and "black-box" behavior. In addition to the analysis functions offered by the original SPICE engine, NI Multisim enables the user to interact with the simulated circuit the same way a designer would interact with a physical prototype. The user can choose from more than 20 virtual instruments, which emulate their real world counterparts, in order to gain a detailed understanding of the circuit's behavior. The list of virtual instruments covers nearly all possible measurements, from AC/DC signals using multimeters, oscilloscopes and function generators, logic analyzers and pattern generators, all the way to more complex applications using bode-, spectrum and network analyzers. To enable real interactivity with the mixed-signal circuit a user can also place and operate switches, potentiometers, as well as variable capacitors and inductors. LEDs, seven-segment displays, lamps and several probes also deliver immediate feedback to the user.

More recently the possibility of developing and simulating a microcontroller (MCU) alongside the mixed-mode SPICE environment of NI Multisim has been introduced through the NI Multisim MCU Module. This MCU can be programmed in either C or assembly, and co-simulated with interfacing analog and digital circuitry.

The circuit in this case study that will be utilized to demonstrate the comparison of SPICE simulation and real

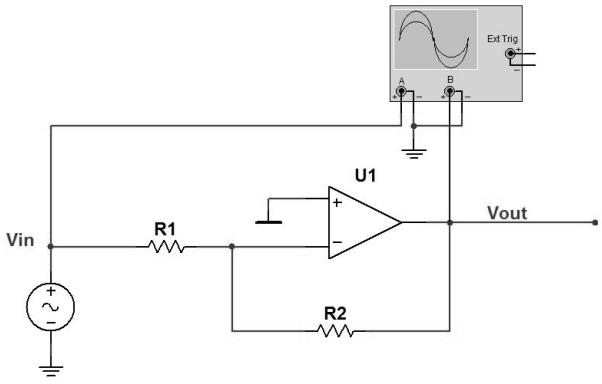

Figure 3. Basic circuit of an inverted operational amplifier

world measurements will be a basic inverted operational amplifier. This amplifier, as designed in NI Multisim is pictured in Figure 3.

By performing KCL (Kirchhoff's current law) or KVL (Kirchhoff's Voltage Law) on an inverted opamp circuit, you can calculate the theoretical output voltage as:

$$V_{out} = -V_{in} * R2/R1 \quad (1)$$

Let us see the output voltage of an inverted amplifier for our experimental values:

Vin = +2V

R1 = 1kOhm

R2 = 10kOhm

Vout = - (+1V) * 10kOhm / 1 k Ohm = -20V

After capturing the example circuit in NI Multisim (as done in Figure 3) and running the simulation the oscilloscope reads the expected simulation result (as seen in Figure 4).

III. Practical Lab Exercise

Repeating the learned theory in a practical lab is the logical next step, and key to a profound understanding of analog and digital circuits. Today, many EE labs are equipped with PC-based measurement systems consisting of hardware and software that leverage the computing power of modern PC-technology and hence leading to powerful yet flexible data acquisition (DAQ) platforms.

Data acquisition involves gathering signals from measurement sources and digitizing the signal for storage, analysis, and presentation on a PC. Data acquisition (DAQ) systems come in many different PC technology forms for great flexibility when choosing your system. Scientists and engineers can choose from PCI, PXI, PCI Express, PXI Express, PCMCIA, USB, IEEE 1394, parallel, or serial ports for data acquisition in test, measurement, and automation applications.

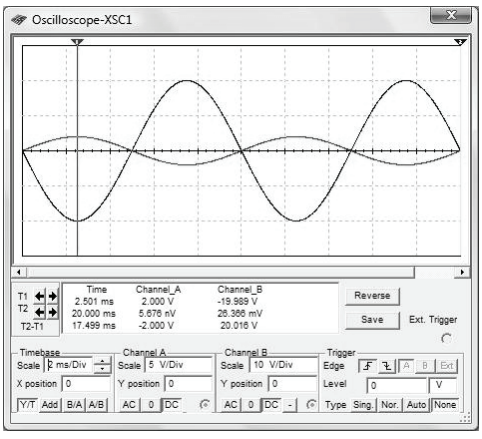

Figure 4: Simulation Results Displayed on the Oscilloscope

There are five components to be considered when building a basic DAQ system

- Transducers and sensors
- Signals
- Signal conditioning
- DAQ hardware
- Driver and application software

A. Transducers and sensors

A transducer is a device that converts a physical phenomenon into a measurable electrical signal, such as voltage or current. The ability of a DAQ system to measure different phenomena depends on the transducers to convert the physical phenomena into signals measurable by the DAQ hardware. Transducers are synonymous with sensors in DAQ systems. There are specific transducers for many different applications, such as measuring temperature, pressure, or fluid flow.

Different transducers have different requirements for converting phenomena into a measurable signal. Some transducers may require excitation in the form of voltage or current. Other transducers may require additional components and even resistive networks to produce a signal. Refer to ni.com/sensors for more information on transducers.

B. Signals

The appropriate transducers convert physical phenomena into measurable signals. However, different signals need to be measured in different ways. For this reason, it is important to understand the different types of signals and their corresponding attributes. Signals can be categorized into analog and digital signals.

An analog signal is measured at a value with respect to time. A few examples of analog signals include voltage, temperature, pressure, sound, and load. The three primary characteristics of an analog signal include level, shape, and frequency.

A digital signal cannot take on any value with respect to time. Instead, a digital signal has two possible levels: high and low. Digital signals generally conform to certain specifications that define characteristics of the signal. Digital signals are commonly referenced as transistor-to-transistor logic (TTL). TTL specifications indicate a digital signal to be low when the level falls within 0 to 0.8 V, and the signal is high between 2 to 5 V. The useful information that can be measured from a digital signal includes the state and the rate.

C. Signal Conditioning

Sometimes transducers generate signals too difficult or too dangerous to measure directly with a DAQ device. For instance, when dealing with high voltages, noisy environments, extreme high and low signals, or simultaneous signal measurement, it becomes essential for a signal conditioning technique to be utilized for an effective DAQ system. Signal conditioning maximizes the accuracy of a system, allowing sensors to operate properly, and guarantees safety.

D. Data Acquisition Hardware

DAQ hardware acts as the interface between the computer and the outside world. It primarily functions as a device that digitizes incoming analog signals so that the computer can properly interpret them. Other data acquisition functionality includes:

- Analog Input/Output
- Digital Input/Output
- Counter/Timers
- Multifunction - a combination of analog, digital, and counter operations on a single device

The heart of a DAQ device is composed of an analog-to-digital converter (ADC) and a digital-to-analog converter (DAC). These two components of the DAQ system are used to generate and read signals. A multiplexer connects the various analog input lines to the ADC. Before an analog signal is converted to a digital signal, it must be sampled. Generally, conversion occurs uniformly in time. Similarly, a DAC can generate a voltage with a maximum specified update rate.

The DAQ board is what dictates the accuracy and speed of any acquired measurement, hence it must be selected with the desired measurement capabilities and performance in mind. The most readily available platforms to house a DAQ device are the desktop and laptop computer. Therefore, a USB-based data acquisition device, the NI USB-6251, has been chosen to study this remote lab, as it can be used with both computer architectures. The DAQ device has the following basic specs:

- 16 analog inputs (16-bit); 1.25 MS/s single-channel sampling rate (1 MS/s aggregate)
- 2 analog outputs (16-bit, 2.8 MS/s)
- 24 digital I/O (8 clocked)
- 32-bit counters

In a typical lab exercise, simple analog and digital I/Os are not sufficient, as the task at hands require sophisticated instruments that measure, quantify and visualize circuit behavior. The NI Educational Laboratory Virtual Instrumentation Suite or NI ELVIS, is a design and prototyping environment for university science and engineering laboratories. Originally conceptualized by Paul Dixon of the Cal State Physics Department, NI ELVIS is now used in educational institutions world-wide. NI ELVIS is a bench-top workstation and protoboard unit that works as a front end to a standard NI multifunction data acquisition board. NI ELVIS (as seen in Figure 5) turns the various I/Os of the connected DAQ device into twelve independent instruments, such as variable power supplies, standard and arbitrary waveform generator, DMM, oscilloscope, dynamic signal analyzer, etc.

The example circuit, an inverted operational amplifier, has also been built on the NI ELVIS breadboard. The appropriate connections have been made in order to drive the circuit with a user defined waveform at the input, and data is being read from its output.

E. Driver and Application Software

The performance of a PC-based DAQ system is not only determined purely by its hardware specifications.

Figure 5. NI ELVIS workstation

The driver and application software play an integral role in the overall system performance. In general, the driver is the programming interface or API, between application software and the hardware. Drivers also take care of specific hardware configurations and operating system issues. Depending on the vendor of a particular hardware, the driver is tailored to only one operating system without any open interface for user defined interaction. It is completely programmable through various programming languages across different operating systems.

The driver for the NI USB-6251, as well as most National Instruments DAQ products, is called NI DAQmx. NI DAQmx delivers an API for multiple operating systems (e.g. Windows, MAC, Linux) and programming languages (e.g. LabVIEW, C, C++, Visual Basic).

1. NI LabVIEW

The application software should be open, flexible, easy to use and effortless to learn. These characteristic allow the user to tailor applications specific to the task without lost time and difficult programming. Where as in the past certain languages were reserved for experts, such as in the case of text-based languages, the new paradigm of graphical programming provides easy entry into creating applications that configure PC-based DAQ systems for engineers, scientist and educators around the world.

The NI LabVIEW, graphical dataflow language and block diagram approach, naturally represents the flow of data in the acquisition of data, and intuitively maps the controls of the user interface,to the back-end block diagram code. This means that programmers can easily view and modify the block data (on the block diagram), or control inputs (on the frontpanel) in the creation of their program, also called a Virtual Instrument (VI).

For novice programmers, LabVIEW Express technology transforms common measurement and automation tasks into much higher-level, intuitive VIs. For experienced programmers, LabVIEW delivers the performance, flexibility, and compatibility of a traditional programming language such as C or BASIC. In fact, the fully-featured LabVIEW programming language has the same constructs that traditional languages have, such as variables, data types, objects, looping, and sequencing

structures as well as error handling. Also, with LabVIEW, programmers can reuse legacy code packaged as DLLs or shared libraries and integrate with other software using ActiveX, TCP, and other standard technologies.

2. *NI ELVIS Software*

As mentioned earlier, when it comes to using PC-based DAQ system in a lab environment, it is essential to deliver ready-to-use instruments. Therefore, NI ELVIS not only serves as a frontend to a multifunction DAQ board, but also features ready to use software instruments for any given measurement type. Alternately, the user can use the included LabVIEW API to build custom instruments or take the functionality of NI ELVIS instruments and incorporate them into another LabVIEW program. This is done simply by selecting the appropriate functions from the NI ELVIS Instrument palette within LabVIEW, and placing them on to the block diagram.

IV. REMOTE LAB

A remote lab requires remote control capabilities from all integrated applications. The complete PC-based DAQ system is based on NI LabVIEW, which offers a built-in web server, that allows for multiple users to view the application, as well as allow one user at a time to control the application. Most SPICE simulation applications lack a web interface, therefore making a remote lab set-up essentially impossible. NI Multisim however offers an ActiveX API that enables other ActiveX enabled programs to communicate and control a Multisim-based SPICE simulation. This innovative feature opens up SPICE simulation to the possibility of a remote-lab.

With this understanding of our lab set-up, we can now construct an application in NI LabVIEW such that it becomes the central piece of software that configures, controls and manages the various steps of this remote lab. These steps include:

- Remote control of the SPICE simulation
- Acquisition of real world measurements
- Comparison of simulated and real data
- Web server functionality

A. *Remote control of the SPICE simulation*

As stated above, NI Multisim offers a complete ActiveX API that delivers methods and properties that

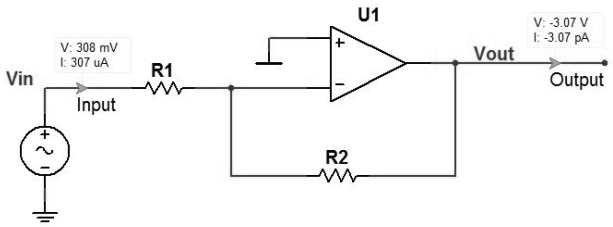

Figure 6: Remote control enabled Multisim circuit

can be used from other applications to control a SPICE simulation. In order to be able to write and read data, sources and probes need to be placed in a schematic that will communicate via our ActiveX API to our web-based remote lab. Input and output data will therefore be transferred via these devices.

The simulation sources, such as an AC or DC source, placed in a circuit count as place holders that expect user defined waveforms which will act as a stimulus in a circuit. Probes need to be placed at specific nodes of interest (as done in Figure 6) to make current and voltage readings available to the ActiveX application.

The process of initiating a remote SPICE simulation via ActiveX can be described in five simple steps:

- Call the Multisim Application
- Open a circuit
- Detect sources (inputs) and probes (outputs)
- Start the simulation
- Set source signals and read probe data

To make the ActiveX API more accessible for LabVIEW users, a simple library for the most commonly used functions was created (as seen in Figure 7) so that all LabVIEW users can utilize this remote control of SPICE simulation.

B. *Comparison of simulated and real data*

As you will recall, the goal of this remote laboratory, is to compare real-world measurements with simulation data, which is to effectively combine theory, simulation and practical lab assignment and improve the overall learning experience for a student. This comparison and the associated analysis is what create sustained and practical knowledge.

Having completed the rest of our remote lab, we now

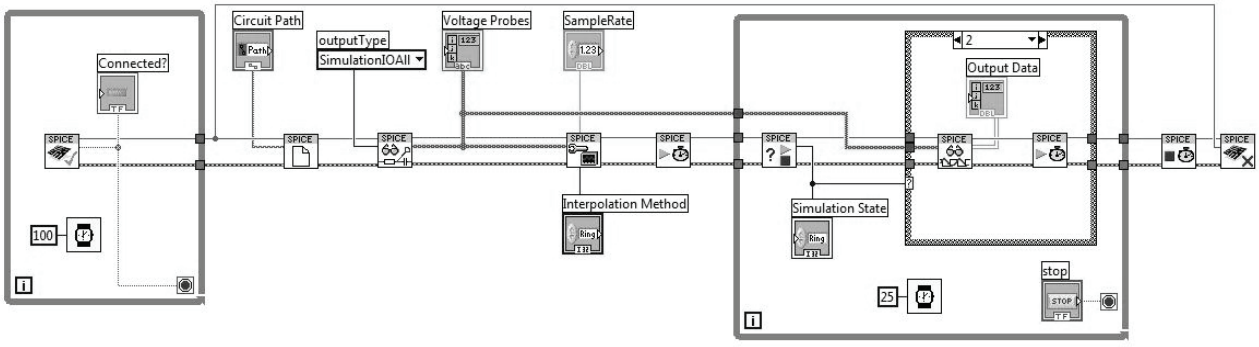

Figure 7. NI Multisim ActiveX API for NI LabVIEW

require a method of exchanging data between the SPICE simulation performed in NI Multisim and the data acquired with NI LabVIEW.

SPICE by nature is somehow limited in the complexity of signals and waveforms that can be applied to a circuit. The provocative question here is how "real" are the simulation results if input signals that drive a circuit are simple periodic wave-forms (such as a sine wave) and does not characterize either the sophisticated source which will be used for the physical circuit, or does not contain the various elements such as noise and phase-shift which are representative of a real signal.

There are several ways data can be exchanged between Multisim and LabVIEW

- Exchange of data files *.lvm (ASCII) or *.tdm (binary)
- Use LabVIEW-based instruments within Multisim in order to connect to hardware I/Os
- Use the Multisim ActiveX API

This case study utilizes data exchange via ActiveX. Data destined for circuit source and readings from probes are exchanged in the form of arrays of numbers and additional timing information.

Simulated data and acquired measurements can be combined within a view created in a LabVIEW VI to measure and display the difference between various signals. The smallest differences in amplitude and frequency, as well as more obvious differences are now easy to detect as they can be analyzed within the LabVIEW application. Not only this, but also design mistakes are quickly exposed, such as in our example.

The inverted opamp uses a 3-pin model that does not simulate the voltage supplies, Vs+ and Vs-.

There are various reasons why one would use a 3-pin model. It is quicker to capture in the schematic, as fewer wires have to be placed. There is also a didactic reason for this design decision, in that it reduces the complexity of the circuits and helps the student to focus on the actual

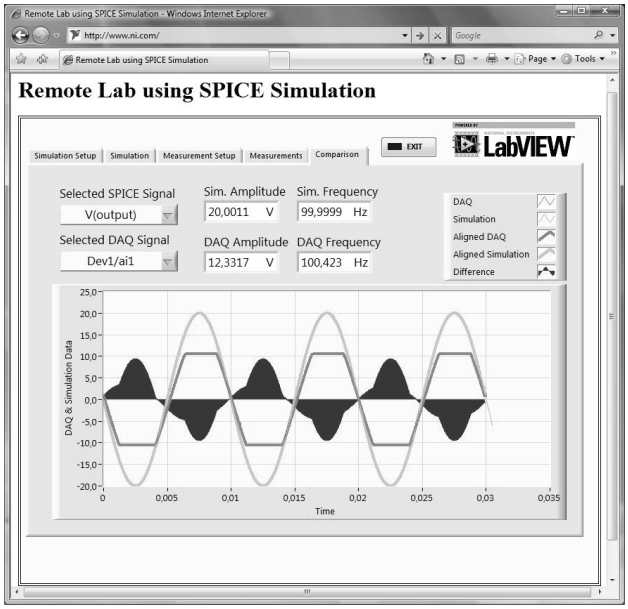

Figure 8: Remote lab showing the comparison of data

base functionality of the component. However, the student must make the logical leap into the physical lab, and understand that power must be supplied to make the design function.

The student will be able to clearly see that depending on the supplied voltage (for example if Vs+ = 15V, Vs- = -15V, Vin = 2V) the previously calculated (1) and simulated voltage level of Vout = -20V can't be achieved, due to the clipping effects.

C. Web Server Functionality

Due to the web-server functionality of LabVIEW, the actual remote-lab virtual instrument can be housed on a single machine and then controlled remotely. Therefore the lab virtual instrument can be embedded into a web page and operated from within a common web browser. All that is required of the client machine to execute in a web page is a browser, with a browser plug-in and machines that has the LabVIEW run-time engine.

The built-in LabVIEW web publishing tools assists the user to create a web page with the embedded application (as pictured in Figure 8) with a few simple steps. Several security feature such as blocking of certain IP addresses, browser access and the visibility of VIs are also managed through the LabVIEW environment.

V. CONCLUSIONS

The chosen example of an inverted operational amplifier is a simple example with an obvious mistake. However it is effective in demonstrating the concept of combining simulation, data acquisition and real-world measurements to uncover a design mistake. This is all made possible by our remote-lab set-up.

The same concepts, steps and technologies can be applied to reveal other real world effects introduced through noise on a signal line, phase shits of power supplies or component tolerances.

Introducing SPICE simulation to remote labs and blended learning, offers an integrated and seamless learning experience for students in electronics education. In the past, SPICE has also been integrated in some remote laboratories via command line calls or XML. However this case study of a remote SPICE lab is different, as it offers for the first time the utilization of build in control capabilities of a SPICE simulation environment.

REFERENCES

[1] E. Robinson, "SPICE Simulation Fundamentals," *National Instruments tutorial, http://zone.ni.com/devzone/cda/tut/p/id/5413 from* 2006.

AUTHORS

I. Knoblich is with National Instruments, Konrad-Celtis-Str. 79, 81369 Munich, Germany (e-mail: Ingo.Knoblich@ni.com).

P. Krauss is with National Instruments, Konrad-Celtis-Str. 79, 81369 Munich, Germany (e-mail: Ingo.Knoblich@ni.com).