Basics 01 : Introduction and Requirements

This first document gives a brief introduction to the SLab project and describes the material needed to do the proposed laboratory projects.

Index

Introduction

The Small Lab (SLab) project pretends to ease self learning in electronics. It does not pretend to be a theoretical introduction to electronics. There are a lot of books for that and the author does not aspire to add more to the pile. So, you are supposed to know what is the resistor Ohm's law, what are the Kirchoff circuit laws and how does an inductor or a capacitor behaves. It is good for you to also know basic circui theory in the frequency “s” domain although it is not always needed to follow the SLab documents.
The objective of the project is to learn by working on lab measurements. That is, by building and measuring circuits.

This is not new. Learning electronics by building circuits is quite typical. The usual problem, however, is the instrumentation. You can do things with only a power supply and a multimeter, but in no time you will need an oscilloscope and a function generator. And that costs money. Moreover, on modern laboratories we tend to make use of automation to perform repetitive measurements and that is not easy to do using budget instruments. Things like obtaining an input to output DC voltage response of a circuit is cumbersome and requires taking manual note of the measurements or building a setup with a function generator and an oscilloscope in X-Y mode. Add to that that it is not always easy to extract the measurement data to perform post processing.

This project tries to ease the instrumentation part of the lab problem by using a combination of a low cost microcontroller development board and some Python code. Having some foundation of Python will help but it is not a requirement. Python is quite easy to learn anyway and probably you already know about it.

SLAB for Instrumentation

As we, humans, are not capable of sensing electrical magnitudes like voltages and currents like we sense textures and colors, we depend on instruments to know what is happening inside an electronic circuit. Moreover, things change quickly inside circuits so we need to be able to adapt the timescale where a circuit operates to the timescale where we can think about what is happening.

Instrumentation is needed to set and measure voltage values both constant and variable in time. The basic set for any workbench is four instruments. To set constant voltages we use a **power supply**. To set a variable voltage we use a **function generator**. To measure a constant voltage or current we use a **multimeter**. Finally, to measure a variable voltage we use an **oscilloscope**. There are methods to work also with currents when a multimeter is not enough but with most circuits, using some tricks, we can work out the current from voltage measurements.

If we want to extract information from the instruments or perform any automated measurements we also need a PC computer. This is the 5th element of any modern basic workbench.

Off course the list can grow if you have specific measurement needs but this list of 5 elements is usually enough for low frequency analog circuits.

All those elements cost money and take space. In order to reduce space usage and instrument costs there are several available devices that connect to a PC using USB or a network interface and act as one or several of the four basic instruments. Note that most modern bench instruments can also be connected to a PC, the difference is that the low cost solutions need the PC to be used and, sometimes, make use of its computing power to operate.

By displaying the information on the PC screen, you reduce the cost and space of having screens on the instruments themselves. You can get PC controlled oscilloscope / function generator combos some also add a power supply to the mix.
There are a plethora of examples like the Digilent Analog Discovery (260$), the Analog Devices ADALM1000 (44€) o the Red Pitaya StemLab (From 200€).
In general, the price for a half decent function generator/scope combo put you above 150€ and, although a lot of instruments can be automated it is not always easy. Prices are currently falling so this numbers could be not accurate today.

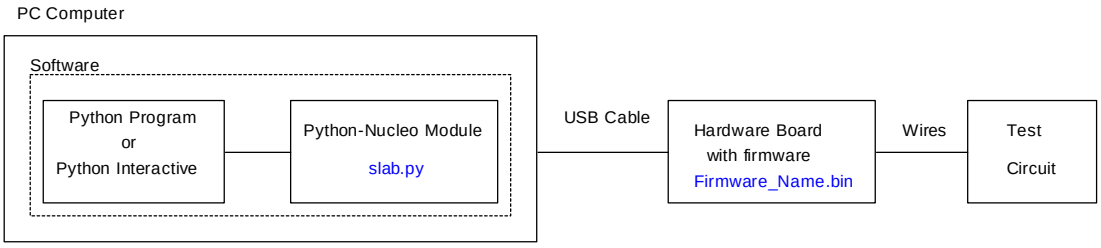
The microcontroller world has come to unbelievable low prices you could not expect several years ago. A lot of microcontroller manufacturers sell cheap demonstration boards at cost price to promote their products. Moreover, some of the featured microcontrollers include not only ADCs, to measure voltages, but also DACs to set voltages. That way you can control the inputs of a circuit and read its outputs. The SLab project proposes to use a cheap microcontroller board to generate the inputs and read the outputs of a circuit under test (CUT).

The system works in client-server fashion. You have a hardware board, acting as server, connected to a circuit. You also have a client PC connected to the board. The PC client sends commands to the board, the board processes them and return the results to the PC.

This solution is not ideal. Maximum sample frequencies in most microcontroller boards are in the low MHz range at best and that is using an optimal firmware. Voltage operation is usually low, usually in the 3V range. And measured values need calibration to resemble the real physical values. But you cannot beat the price. As an example, current STM32 Nucleo boards feature an ARM Cortex M4 CPU and include DAC and ADC peripherals for about 10€. Other manufacturers are in the same range of prices and features.

Using generic programmable boards have its pros and cons. The basic inconvenient is that you need to program them to perform any operation. The related advantage is that, as you have developed the software it is running, you have complete control on its operation.

The Slab System, at least, requires the four elements shown in the following figure:



First, on the right side, we have our test circuit we are measuring that is designed to learn something about electronics. On its left we have a hardware board that is basically a cheap microcontroller demonstration board. Inside this board, there is a custom made SLab firmware that enables the board to perform some operations on the circuit as they are commanded from a PC through an USB cable.

The rest of the system is software on a PC. The hardware board communicates with a slab.py Python module. Finally, the user, by the way of interactive or scripted commands, instructs the slab.py module to perform operations on the test circuit. Both the board firmware and the slab.py module are provided to you so you don't have to develop them.

Want to set one circuit input to 1.2V?, just type **slab.setVoltage(1,1.2)** on a Python console.
Want to read the voltage of a node?, type **slab.readVoltage(2)**.
Want to read all four ADC voltages? **slab.printDC( )**
Want to plot the response of the circuit to a waveform? **slab.wavePlot(…)**

The slab.py module provides commands that interact with the hardware board. It is easy to build upon this functionality by adding more sophisticated functions like DC curves or AC analysis. Adding this functionality to the slab.py would make this file too big and less stable. That's why the slab.py module is complemented by a set of sub modules like slab_dc.py, slab_ac.py, slab_meas.py and slab_fft.py. There is also an easy to use slab_ez.py module that hides most of the SLab internals for a less powerfull but more straightforward path to circuit measurements.

Now you can use **dc.curveVV(....)** from the DC module to obtain a Vo(Vi) out circuit curve or **dc.curveVI(....)** to obtain a Io(Vi) circuit curve.
In the case of the AC module, you can use, for instance, the **ac.bodeResponse** to draw the bode plot from a circuit AC response.

As all commands are Python functions, you can script any sequence of commands to automate whatever measurement you want to perform. Also all the code, both the board firmware and the python module, is public and open source, you can modify it if you think you could do better or you can also build new modules over the provided ones. The communication protocol between the board and the python module is standardized, so you can also use any board you like as long as it features at least 2 DACs and 4 ADCs and you are able to develop its firmware.

Is it the best solution? Perhaps it won't always be. Using a microcontroller board with no specific instrumentation hardware has its drawbacks. A better solution could be to use a dedicated solution like the **Analog Discovery** or other PC controllable function generator/oscilloscope combo with some Python glue code. But that put you over the 200€ mark. Moreover, learning how to work out the measurements using modest equipment makes you learn a lot about errors and calibration and you never know enough about those topics on any laboratory environment no matter how expensive the equipment is.

In addition, having full control on both the board firmware and the Python code gives the maximum compatibility and integration in all elements on the chain from the user interface to the circuit to measure.

What SLab is

The SLAB system is a project than enables you to develop electronic learning projects in a cheap way. The main goal of the project is to use a cheap microcontroller board to provide, at least, a supply voltage and six analog I/O lines that can be used to interface an experimental circuit under test (CUT). So, the main elements that the SLAB hardware provides are:

- A supply voltage that gives a constant Vdd voltage (for instance 3.3V) to power the circuit.

- 2 Output lines, associated to two separate Digital to Analog Converters (DAC), can be set to any value between GND and Vdd.

- 4 Input lines, associated to four Analog to Digital Converter inputs (ADC), can read any voltage in the circuit if it is between GND and Vdd.

Inputs and outputs are always synchronized, so you can always show the DAC and ADC values on a coherent time axis. For instance, you can inject a sine wave on DAC1, a square wave of different frequency on DAC2 and see the response of the circuit on four ADC nodes having good synchronization between inputs and outputs at each time point.

A custom firmware in the microcontroller board interacts both with the ADC and the DAC elements so that you can generate and read voltages on the circuit under test.

In order to ease the interaction, the board firmware communicates with a PC that houses a custom Python set of files. By importing those files in a Python program or by using Python in interactive mode you can control the board operation and take measurements on the circuit.

Operation **is always on demand**. You send a command to the board, the board interacts with the circuit, and you get a response from that interaction. Most interaction results can be shown graphically on the PC computer.

In SLab you setup an experiment, perform the measurements, and interpret the results.

SLab has deep roots on Python automation. Every action you can perform with the system can be written down on Python code. That means that it is highly scalable. For instance, from generic wave response that is implemented on the board firmware, the Python layer code can obtain the circuit response for several different frequencies and obtain a Bode plot, or can also perform a FFT and calculate the harmonic distortion.

You can also design you own Python programs that can perform automatic measurements. If you want to get readings during a very long time following a sequence of predetermined voltage excitation values, it is easy to automate.

As all voltages, both inputs and outputs, are just vector objects from the Python point of view, you can use any of the available Python libraries to process them or you can take readings, process them, and generate excitations from them if you need.

Moreover, you can also add interactivity in the program that drives the board so that it changes its operation depending on the readings. As the program is developed using high level Python language, it is easier and more flexible to test concept this way than programming a microcontroller board.

What SLab is not

SLAB is not an oscilloscope, a function generation or a combo of both. In an oscilloscope or a function generator the instrument is always interacting with the circuit. An oscilloscope shows, in real time, the voltages in a circuit. You setup the scope and, at any time, you can see on its window, a graphical view of the circuit voltages. In a similar way, after you setup a function generator, it continually injects the selected waveform on the circuit.

SLab don't work that way. You can instruct the system to read one second worth of data from all four ADC channels, but you cannot get a continuous data stream. You can also generate 100 cycles of a 1kHz sinewave, and at see the circuit voltages during this time, but you cannot generate a continuous sinewave. Not, at least, if you also want to obtain measurements. Moreover, in order to start a new action on the SLab system, the previous one need to have ended. You give an order, wait for the result, examine it, generate a new order, and so on...

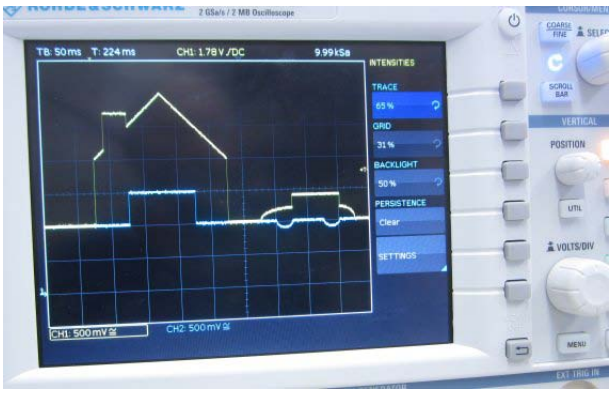
An useful scope needs enough bandwidth and sampling rate, but also a high enough update rate because you expect it to be real time. In the case of a scope implemented with microcontroller board and a PC, that gives a lot of stress on the communication link and both the microcontroller and PC code. SLab doesn't pretend to win this war.

SLab also is not a good circuit troubleshooting tool. Poking at different nodes of the circuit is possible but requires sending a new read command for each node. Due to how the system operates, you only get information on the circuit when you execute a command so it is not a good tool to locate events that happen at unknown times. It cannot substitute a scope with a decent update rate.

Finally SLab is also a system designed for low frequencies and sample rates. The maximum sample rate with current boards and firmwares is close to 80 kHz in the best case when reading only one channel. When reading four channels it can easily drop below 20 kHz. Future firmwares can optimize the system but the sample rate will always be limited to the one associated to the microcontroller used. As an example, the microcontroller in the STM32 Nucleo F303RE board has a maximum sample rate of 5 MHz for 12 bit ADC readings and 64 kBytes of RAM memory. Most low budget scopes are better than that.

The performance of the SLab system is below average compared with other instruments, but Python automation makes it a quite scalable and versatile tool.

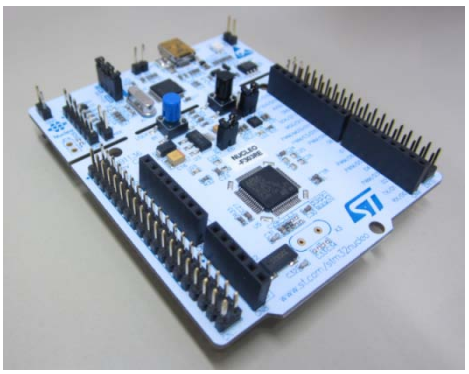
As an example, isn't it pretty this arbitrary dual waveform generation?



The F303RE SLab implementation

Currently the SLab board firmware has been developed for two STM32 Nucleo64 boards. The F303RE and the L152LE boards. As the first one is capable to operate at higher sample frequencies, and has the same cost, it is the recommended one. Moreover, most SLab development is done on this board so this is the one whose firmware is updated more often. Although this description is particular for this board, any other SLab enabled board will work in a similar way.

The following figure shows the Nucleo F303RE board.



The current SLab firmware for the board is developed on MBED. It is not currently fully optimized so it is not yet capable to use the full capabilities of the board.

The board can set the two DAC channels at any value between GND and Vdd (3V3) at 12 bit resolution and can also read the four ADC channels with the same resolution. A calibration procedure improves the accuracy of the values set or read.

As no amplification is provided, you have always the same voltage resolution. Thanks to the 12 bits on the converters, however, the resolution is near 1 mV for a 0 V to 3.3 V range giving a maximum dynamic range of 72 dB.

A part from setting and reading single voltage values, there are commands that can operate on the board using a sequence of values at a fixed sample rate. The sample memory on the board can hold a maximum of 20.000 sample values with 12 bit resolution. This memory is unified both for generating and reading samples.

The firmware implements eight low level commands to operate on the board at a fixed sample rate. You can skip the list at this point if you don't want to know the details:

Asynchronous read: Using this command, you can read one to four ADC channels during a fixed amount of time and report the readings.

Triggered read: This command in similar to the *Asynchronous read* command, but the captured data is centered at the time when the first ADC channel crosses a given voltage value.

Step response: This command gives the response read on one or several ADC channels when the first ADC channel changes between two voltage values.

Wave response: This command give the response read on one or several ADC channels when a waveform is being generated on the first DAC channel.

Dual wave response: This command is similar to the *Wave response* one, but this time we generate two different waveforms on both DAC channels.

Single wave response: This command is similar to the *Wave response* on but only reads a single ADC channel that can be chosen from the four available. As the code is optimized for only one reading, the capture can be much faster.

Wave play: This command generates a waveform on the first DAC channel during a given time without performing any ADC reading.

Dual wave play: This command is similar to the Wave play one, but two signals are generated, at the same time, on the two DAC channels.

The following tables shows the minimum sample time available for each of the different modes and number of ADC channels used.

Command	Number of ADC channels used			
	1	**2**	**3**	**4**
Asynchronous Read	13µs	33µs	43µs	54µs
Triggered Read	13µs	33µs	44µs	54µs
Step Response	13µs	36µs	46µs	56µs
Wave Response	13µs	38µs	47µs	58µs
Dual Wave Response	13µs	40µs	50µs	61µs
Single Wave Response	13µs	Not available		
Wave play	13µs (*)			
Dual Wave Play	13µs (*)			

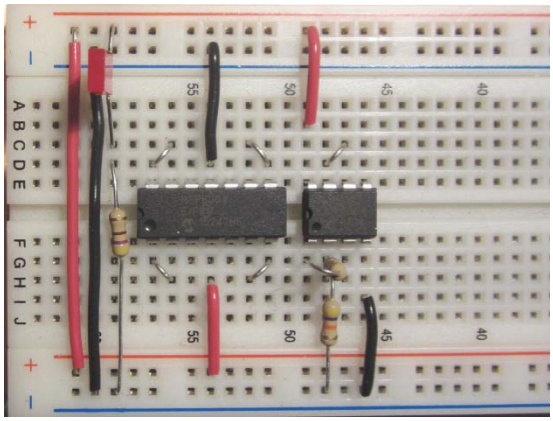
(*) No ADC read is performed

Using a sample time below those limits will generate a ***Sample Overrun*** error indicating that you are trying to read a sample before reading the previous one.

You can see that the system tops at 13 µs sample time (77 kHz sample frequency) when reading one channel or generation waves without any reading. As the current firmware multiplexes the readings when reading several channels, the performance gets worse the more channels you read. That gives a worst case of 61 µs (16 kHz sample rate) when generating two waveforms and reading all four ADC channels.

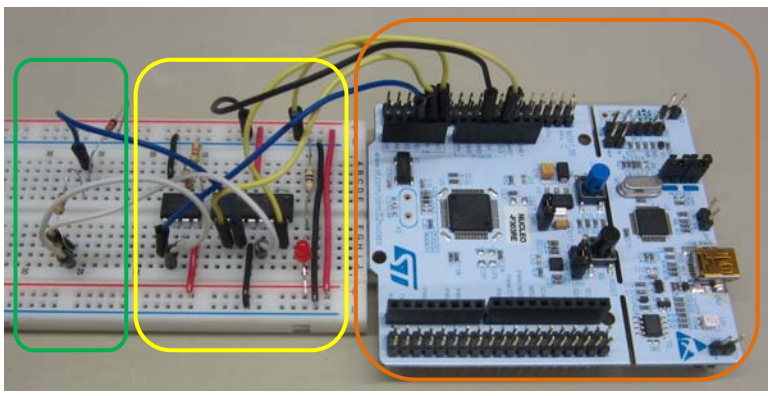
The board communicates with the PC using a mini USB connector and no cable is provided with it, so you will need an USB-mini to USB-A cable like the one below. Probably you have one around.

The board can directly interface with the circuit under test (CUT), but the raw DAC and ADC capabilities of the board are not good enough due to loading effects, so using a driver circuit between the board and the CUT is recommended. For the F303RE board, this circuit requires two opamp ICs (for a total of six opamps) and two resistors. The circuit can be implemented on a breadboard like in the following figure:



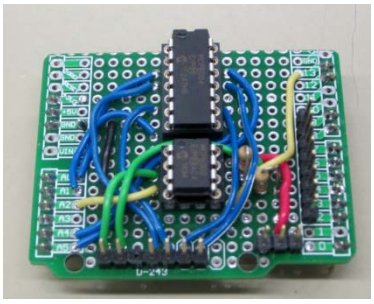
A LED and a resistance have been added to the circuit to show that the circuit is powered, but those components are not really needed for the operation of the SLab system.

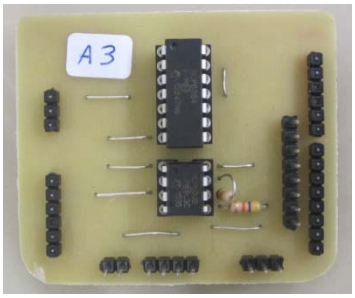
In the following figure you can see the F303RE Nucleo board connected to a breadboard that includes the buffer driver circuit and a Circuit Under Test (CUT).



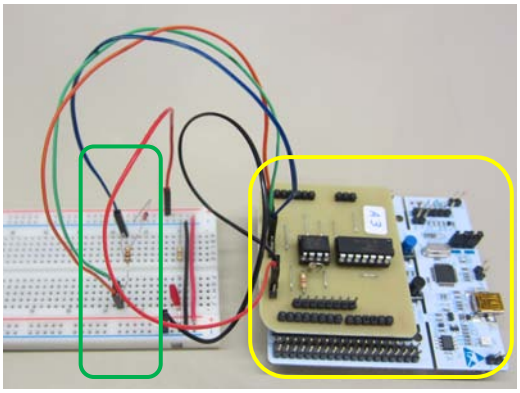
CUT Buffer Nucleo Board

The driver circuit can also be built on a shield that can be mounted over the F303RE board. The following figures show two shield designs. On the left there is a shield built over an Arduino UNO generic shield board and on the right there is a shield built over a custom made single side PCB. There is an specific SLab document about those shields.





The following figure shows the F303RE Nucleo board fitted with a custom made driver shield and connected to a Circuit Under Test (CUT) that is mounted on a breadboard.



CUT

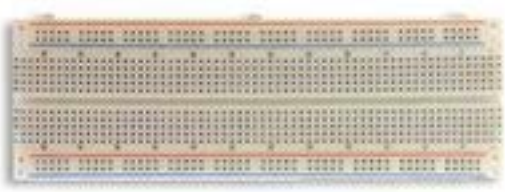
Nucleo Board
and Driver Shield

As you can see you have several options to build the complete SLab hardware system. In the shown case, it is composed of a Nucleo 64 F303RE board and a driver circuit. The SLab specification is open, so there will be backward compatible improved designs in the future.

It is possible to develop a driver board to operate at higher currents or voltages than the ones available with the proposed configuration. I leave it for the future as, for now, I want to keep the system cost as low as possible.

Prototyping board and wiring

In order to implement and measure circuits, we need to build them. There are many options to build circuits but most of them require soldering components making them inadequate to testing the effect of changing components. A solderless prototyping board or breadboard like the one on the figure is enough for most Small Lab projects.



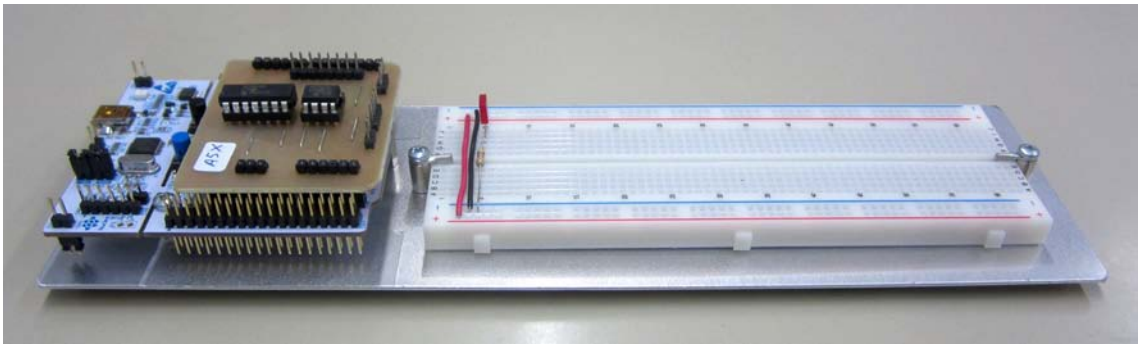
We will also need some cables to connect together the components. We can use jumper wires. They are flexible with rigid pins at the ends. We can also use rigid wire and cut to the proper size. Better yet you can use a combination of both: Rigid cables for short distances and jumper wire for longer distances.

Jumper Wires

Rigid cables

The same jumper wires can also be used to connect the hardware microcontroller board to the circuit under test. Doing that with rigid cables could be more difficult.

You can join the MCU board and the prototyping board in one support to ease the handling of the system. The following figure shows the SLab ***Long Board*** that includes a base support with rubber feet, a prototyping board, a Nucleo MCU board and an opamp driver shield. A basic electronic lab in 28 x 8 cm space.



Passive components

Circuits are composed of passive and active components. Active components are the ones that can provide a power gain and passive the rest. The passives in this section will be useful for several different projects. If a passive is specific only for a reduced list of projects, it will be requested when it is needed.

Resistors

We will need two full E12 series of 1/4W resistors starting at 10Ω and ending at 1MΩ.

Resistor sizes depend on their capability to dissipate power. 1/4W resistors give a good compromise between size and power and is one of the most useful sizes.
An E12 series of resistors includes 12 different values for each ohm decade. Our series of resistors will be: 10, 12, 15, 18, 22, 27, 33, 39, 47, 56, 68, 82, 100, 120, ... you get the idea.
There are 5 decades and the 1MΩ value so each series is 61 resistors.
Fortunately full series are usually sold in packs so you don't need to buy each separate resistor. If you have a more limited pack that starts at 100Ω or is in the E6 series (6 values for decade), it could be enough as a start. You can always buy individual resistors if you don't have the proper value for an experiment.

Resistors are coded with colors. Take some time to learn about them.

Potentiometers

Potentiometers are resistors that are adjustable in value by twisting a knob. They are sold in different sizes and power capabilities, but we will restrict to the smaller ones, known as trimming potentiometers or trimpots. You can see two of them in the following figure. As they are quite small, you will need a small screwdriver to turn its adjustment knob.

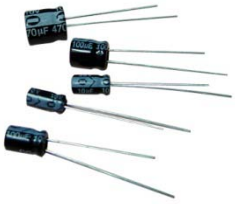
The one on the left is has less than one full turn adjustment range while the one on the right needs several turns to a full value change. Is more precise, slower to adjust and usually more expensive than the first one.
You will need only a pair of 10kΩ trimpots to do the proposed circuits.

Capacitors

We will need several specific capacitor values.
SLab projects try to use the smallest possible set of capacitors.
Those components can be fabricated in different technologies depending on the capacitance value and the intended used. We won't use the capacitors to their spec limits so any kind will do if it is the proper value.


Ceramic disk


Multilayer


Electrolytic

We propose you get:

4x 1nF Capacitors (Ceramic disk or Multilayer)
4x 10nF Capacitors (Ceramic disk or Multilayer)
4x 100nF Capacitors (Multilayer)
2x 1µF Capacitors (Electrolytic)
2x 10µF Capacitors (Electrolytic)

Notice that all capacitors in the figure feature both terminals on the same side. Those are known as **radial components** and are better for breadboard use than the axial ones that have one terminal at each side.

We recommend aluminum electrolytic capacitors for capacitances of 1µF and over. It is easy to find 1µF film capacitors, not so much 10µF. In any case we need to know how to deal with electrolytics.

The value of the capacitor can be directly written on their body or can also be written with a three digit code. In that case, the first digits give a number from 0 to 99 and the third one a power of ten. The total capacitor value is, in this case, in pF. As an example, a 100nF capacitor is coded:

$$Code = 104 \qquad C = 10 \cdot 10^4 pF = 100nF$$

Diodes

Diodes are devices that only conduct in one direction. For our circuits we will use the venerable 1N4140 signal diode.

Those components are cheap, although we don't need too many of them, grab half a dozen so that you have spares if you burn any of them.

There is a special kind of diode that dissipates part of the consumed power emitting light. Those are known as Light Emitting Diodes (or LED) and you can obtain different color versions. We will use infrared, red, green and blue. The specific model is not important as long as they are 3mm or 5mm leaded versions like the one in the figure:

We will need three red ones and two of the each of the other colors.

Active components

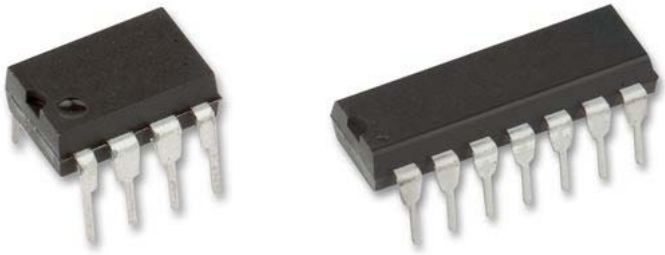
Active components are able to provide power gain. That is, they can provide a load more power that it obtained from a signal source. Here we will include the most basic active components if more specific components are needed, they will be introduced in the relevant projects.

BJT Transistors

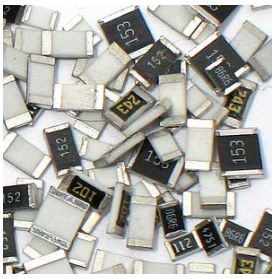
Bipolar Junction Transistors are the most basic active components, they feature three terminals and come in two types NPN and PNP. We will grab 4 NPN BC547B transistors and 4 PNP BB557B transistors.

Operational amplifiers

Operational amplifiers (opamps) are one of the most basic building blocks of analog circuits, especially at low frequencies. Moreover, we will need some of them to improve the performance of the hardware board. As our projects work on a 3,3V supply, we will need low voltage full rail opamps. We will need four microchip MCP6002 (two opamps on one 14 pin chip) and two MCP6004 (four opamps on one8 pin chip).



Beware that we need the dip versions like the ones above, don't get any SMD version.



Note about SMD components

Today, most of the modern circuits are fabricated using surface mount tecnlogy (SMD). Unfortunately its components are difficult to use in basic prototyping. We will use the older leaded through hole (TH) technology. That means that some modern components will be unavailable for breadboard use.

Base Boom

So, this is the base bill of materials (BOM) for the SLab project. Some additional components, if they are too specific, will be added in each set of lab modules.

- (1) STM32 Nucleo64 F303RE Board
- (1) Mini USB to USB-A male cable
- (1) Solderless Breadboard
- (1) Set of male-male jumper Wires
- (2) 10 Ω to 1 M Ω E12 1/4W resistor series
- (4) 1nF Ceramic capacitor or film
- (4) 10 nF Ceramic capacitor or film
- (4) 100 nF Film capacitor
- (2) 1 µF Electrolytic capacitor
- (2) 10 µF Electrolytic capacitor
- (2) 10 k Ω Trimpots
- (4) 1N4148 Diode
- (2) Infrared LED
- (3) Red LED
- (2) Green LED
- (2) Blue LED
- (4) BC574B NPN Transistor
- (4) BC557B PNP Transistor
- (4) MCP6002 Dual Opamp chip
- (2) MCP6004 Quad Opamp chip

In the following table we have the base BOM with reference codes for Farnell. You don't need to buy at Farnell and they don't give me any money if you do, but, although their prices are not always the cheapest they are usually fair and it is good place to search for components. Moreover each component is linked to its specifications and data sheets so you have plenty of information for each component.
Some low price components have minimum order quantities, in that case quantities are bigger than the minimum suggested in the BOM.
Unfortunately they don't seem to provide E12 series resistor sets in quantities of less than 100 resistors of each value (Ref 1171087 at 42,16€), so you will probably need to get them elsewhere.

Component	Quantity	Farnell Code	Unit Price (€) March 2017	Price (€)
STM32 Nucleo64 F303RE	1	2467271	10,63	10,63
Mini USB to USB-A male cable	1	1651027	3,07	3,07
Solderless Breadboard	1	2213346	8,74	8,74
Set of male-male jumper wires	1	2396146	3,64	3,64
10 Ohm to 1 MOhm E12 1/4W resistor series	2	Search Elsewhere		
1nF Capacitors	4	1694097	0,157	0,628
10 nF Capacitors	4	2309024	0,0737	0,2948
100 nF Capacitors (min 10)	10	1141777	0,0758	0,758
1 uF Electrolytic Capacitors	2	9451072	0,0415	0,083
10 uF Electrolytic Capacitors	2	9451056	0,0479	0,0958
10 kOhm Trimpots	2	1703794	2,05	4,1
1N4148 Diode	6	2677463	0,0172	0,1032
Infrared LED	2	1328299	0,283	0,566
Red LED (min 5)	5	1581136	0,0499	0,2495
Green LED (min 5)	5	1581123	0,053	0,265
Blue LED	2	2373467	0,177	0,354
BC547B NPN Transistor (min 10)	10	2101372	0,06	0,6
BC557B PNP Transistor (min 10)	10	2453804	0,0632	0,632
MCP6002 Dual Opamp (DIP8)	4	1332117	0,374	1,496
MCP6004 Quad Opamp (DIP14)	2	1332121	0,486	0,972
			Total	**37,28**

As you can see, you can get the full base SLab BOM that includes the needed instrumentation and components for less than 40€. It's difficult to learn electronics hands-on in a lower budget.

References

SLab Python References

Those are the reference documents for the SLab Python modules. They describe the commands that can be carried out after importing each module.
They should be available in the **SLab/Doc** folder.