Mechatronics and Measurement

Laboratory Manual

Rico A. R. Picone
Department of Mechanical Engineering
Saint Martin's University

Contents

Preface

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna.

Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

01

Introduction

Lab Exercise 01 Introduction

The objectives of this lab exercise are for students:

1. to organize into groups,
2. to be introduced to myRIO boards,
3. to be introduced to LabVIEW,
4. to be introduced to LaTeX,
5. to begin reading technical documents, and
6. to begin writing summaries of technical documents.

Lab 01.1 Materials

The following materials are required for each lab station:

1. a PC with LabVIEW installed,[1]
2. a myRIO configured with LabVIEW,[2] and
3. headphones.

Lab 01.2 Groups

Form groups of three.[3] This will be your group for the remainder of the term.

Box 01.1 Task rotation

The following three tasks should be completed individually by rotating group members in 45-minute intervals.[a]

[a] A group of four students will need to shorten these to about 30 minutes and may require additional lab time to complete all tasks.

Lab 01.3 Task I: LabVIEW and the myRIO

Find a Robotics Lab station to perform this task. Log in to your student account.

[1] See Resource 2 for more details on the LabVIEW software configuration.
[2] See Resource 3 for more details on the myRIO software configuration.
[3] In the event of the total number of students not being divisible by three, the maximum number of groups of three should be formed, minimizing groups of two and especially four.

Lab 01.3.1 Introduction to LabVIEW

LabVIEW is a graphics-oriented software and programming language for rapidly controlling measurement and control instruments. It gives students the power to capture data; create "virtual instruments" that can, for instance, graph data acquisition, live; and interface with external instruments. We'll use it in most of our lab exercises! It runs on desktop computers and also on the myRIO boards, described in the next section.

Launch LabVIEW and open in a web browser the National Instruments site *Learn LabVIEW*:

`www.ni.com/academic/students/learn-labview`.

Watch the following videos and follow along in LabVIEW!

Box 01.2 Open-lab tasks

The Robotics Lab is available for you to work on some of these tasks outside of the designated lab times—that is, it's an "open lab." Due to time-and-space constraints, you will need to come in outside of lab time to complete tasks with the OL stamp.

Time estimates for working through the tutorial exclude pause-and-play time.

1. LabVIEW Environment (10 minutes)
   - Introduction to the LabVIEW Programming Environment
   - Charts and Graphs
   - Building a Graphical User Interface (GUI)
2. Loops and Structures (15 minutes)
   - While Loops
   - For Loops
   - Case Structures
   - Shift Registers
3. OL Data Types and Structures (8 minutes)
   - OL LabVIEW Data Types
   - OL Arrays
4. OL Graphical Programming (4 minutes)
5. OL Programming Tools (3 minutes)
6. OL Debugging and Handling Errors (4 minutes)

Lab 01.3.2 Introduction to myRIO

The myRIO is a powerful real-time input-output device that gives you the power to measure data from sensors and control signals to actuators. It is similar to certain high-end Arduino boards,[4] which can also be used for these tasks. We'll use the myRIO for data acquisition in most of our lab exercises!

A distinct feature of the myRIO is that it (and its FPGA) can be programmed in the graphical LabVIEW programming language. In this course, we will use this graphical language, but in certain advanced courses the C programming language is used to program the myRIO.

Download the myRIO *User Guide and Specifications* at

`www.ni.com/pdf/manuals/376047c.pdf`

and skim it (skipping the boring parts).

Visit the National Instruments site *Learn to Use myRIO* at

`www.ni.com/academic/students/learn-rio/applications`.

Watch the first video, *Getting started with NI myRIO* (8 minute).

Plug in the myRIO to power and connect it to the computer via the USB cable. A dialog should appear. Select the Getting Started Wizard. Proceed, but *do not* change the name of the myRIO. Once you have reached the Test Panel, move the myRIO around and view the response of the indicators. Also test the buttons that toggle the myRIO LEDs and the `Button0` button on the myRIO.

Proceed in the wizard until you can select the option to Start My First Project. This will launch a tutorial. Work through the tutorial.

Box 01.3 Go deeper

Although it is not required, one can later explore the other videos on the *Learn to Use myRIO* page. There's even a tutorial (*Data Dashboard and NI myRIO*) on wireless real-time data display on a mobile device. In addition to displaying data, you can *control the myRIO* on wireless mobile devices. This could make a nice user interface for senior design projects! Additionally, the myRIO Project Essentials Guide has several cool project tutorials.

[4] Recently, Arduino has released boards with FPGAs, which improve their competitiveness with the myRIO for high-speed signal processing and control.

Lab 01.4 Task II: LaTeX for reports

If you brought your own laptop, find a seat in the Robotics Lab. If not, log in to a workstation in Cebula 101 (CAD Lab) computer.

LaTeX is a document preparation system for scientific and engineering documents. Read thoroughly and follow along with the instructions in Resource 1.

Lab 01.5 Task III: Writing and reading

For this task, find a seat in the Robotics Lab that isn't at a lab station.

Download and read this article (password: `me316`). It's from a scientific engineering journal, and it will give you a good feel for how technical documents should be written.

Lab 01.6 What to turn in next week

Submit to me a pdf of the report template that you have edited to include the names of your group. You can leave the instructions in the file. Read them.

Also, write a paragraph or two summary of the the Fedrizzi article. Include it under a new, first section in the report named "Summary of Fedrizzi Article". Somewhere in this paragraph, cite the article. Here is the BibTeX citation for it.

```
@article{Fedrizzi2015,
Author = {Marcus Fedrizzi and Julio Soria},
Journal = {Measurement Science and Technology},
Number = {9},
Pages = {095302},
Title = {Application of a single-board computer
as a low-cost pulse generator},
Url = {http://stacks.iop.org/0957-0233/26/i=9/a=095302},
Volume = {26},
Year = {2015}}
```

Resource: 1 LaTeX Tutorial

LaTeX is a document preparation system widely used in science and research-oriented engineering. It allows authors to create high-quality documents with technical content.

Resource 1.1 Overleaf

A great way to collaborate on LaTeX reports is to use Overleaf instead of (or in addition to) installing it locally. This is actually preferred for collaborative projects. A template is available here.

Resource 1.2 The document structure

The following are the highlights of the file structure.

- The `report.tex` file is the one you will mostly be editing.
- The `figures` directory contains a single figure file. This is where you'll put your figure files to be included in your report.
- The `report.bib` file contains the references.
- The `.sty` files load packages and define some macros. You will probably not be editing the `.sty` files. If you would like to add a package, you can do so in the header of the `.tex` file, preferably after the line `\usepackage{mycommands}`.

Resource 1.3 LaTeX basics

My first rule is build often. Error messages are not always helpful, so I recommend building after a complete thought has been expressed. It's a nice cadence, like breathing.

New *paragraphs* can be added with two linebreaks. A single linebreak does not give you a new paragraph.

You can add new sections with the command

```
\section{Name of Section}
```

Subsections and subsubsections are created with the command

```
\subsection{Name of Subsection}
\subsubsection{Name of Subsubsection}
```

The lines that follow these commands are included in that section/subsection/subsubsection.

You can label a section (and many other objects) with the following syntax.

```
\section{Name of Section} \label{sec:name}
```

The label is `sec:name`, and can later be referred to in the document with the command

```
\ref{sec:name}
```

or

```
\autoref{sec:name}
```

The latter command is usually preferred because it returns the type of object, like "Section 3" or (if the object was a figure) "Figure 2". The **\ref** command gives only the number, such as "3" or "2". It is not necessary to label a section with `sec:name`; it could just as well be `name`. However, it is common practice to use prefixes like `sec:` (section), `fig:` (figure), `tab:` (table), and `eq:` (equation) for labels in order to differentiate them.

In the style (`.sty`) files I have included in the template, the `hyperref` package is loaded, so references will be hyperlinked. If you would like to include a url, you can create a hyperlink with the syntax

```
\href{http://ricopic.one}{the coolest website}
```

Resource 1.4 Adding your own figures

The template `report.tex` includes a figure. The figure that is loaded is the external file `figures/data.pdf`, wish is located in the `figures` subdirectory. The syntax we usually use to load figures is

```
\begin{figure}[bt]
  \centering
  \includegraphics[width=1\linewidth]{figures/data.pdf}
  \caption{here's a caption.}
  \label{fig:name}
\end{figure}
```

This creates a `figure` *float* environment, which inserts a figure as close as possible to the location in the document where you've placed it, but at the bottom (`b`) or top (`t`) of a column. Sometimes the placement of a figure is mysterious, and needs to be tweaked at the end by moving it around in your `.tex` file. Don't bother with placement until you've finished the entire document.

You may change the width of the figure by changing the `width` parameter in `\includegraphics[width=1\linewidth]{...}`. It's convenient to leave it in terms of the `\linewidth`, so you may like to use `0.9\linewidth`, for instance. You may also like to use dimensions like `3.2in` or `7cm` or `200pt`, which are all valid.

The file path (`figures/data.pdf` above) can either be relative or absolute. I highly recommend a *relative* path for portability.

The `\caption{}` can always be edited. It should be descriptive, and even redundant with the text. The reason for this is that when many people read technical documents, they read the abstract, look at the figures and captions, and read the conclusions. It should tell a complete story.

The `\label{}` name (`fig:name` above) can be whatever you like (sans spaces or weird characters). I recommend the prefix `fig:` for all figure labels.

Resource 1.5 Adding your own equations

One of the most powerful features of the LaTeX system is the beautiful equations. With a little work, you can master the syntax and write them with ease. There are a few environments for equations.

Resource 1.6 Equation environments

For *inline* equations, include the contents between two `$` signs, like $\cos(x) + 7$. Even when you refer to a variable in your text, you should use an inline environment, like `$x$`. This will ensure the typesetting is consistent for that variable.

For *display* equations, there are a few options. I recommend mostly using

```
\begin{align} \label{eq:name}
  y = x_2
\end{align}
```

Strictly speaking, with one equation, `align` is overkill. Its typical application is when you would like to align two or more equations on consecutive lines. The syntax then is

```
\begin{align}
  y &= x_2  \label{eq:name1}  \\
  z &= x_3  \label{eq:name2}
\end{align}
```

where `&` is the alignment tab that expresses where the equations are to be aligned (in this case at the equal signs). In the given example, both equations will be given a unique equation number. However, this is often undesirable, especially if the second equation is really just a simplification of the first, as in

```
\begin{align}
  y &= 0 - x_2  \\
    &= x_2
\end{align}
```

In this case, it is more common to want to number just one equation, which can be done with the syntax

```
\begin{align}
  y &= 0 - x_2  \label{eq:name}  \\
    &= x_2 \nonumber
\end{align}
```

Note that there is no need to label every equation, even if it is numbered. You'll only want to label equations to which you'll want to refer later.

If you want equations to be numbered as *subequations*, like "(2a)", "(2b)", etc., then use the syntax

```
\begin{subequations}
  \begin{align}
    y &= x_2  \label{eq:name1}  \\
    z &= x_3  \label{eq:name2}
  \end{align}
\end{subequations}
```

Finally, if you want no numbers, use the syntax

```
\begin{align*}
  y &= x_2  \\
  z &= x_3
\end{align*}
```

Resource 1.7 Math syntax

LaTeX math syntax must always be included in a math environment (see above) and is pretty straight-forward. This reference page, beginning with math symbols is very useful. The usual math operators like + and − are the obvious syntax, but others are more subtle. If you'd like an explicit "times" operator or cross product, you'd use the syntax `x \times y`, for instance.

Vectors are best expressed with a boldface font. The best way to do this is to use the include `bm` package with the command `\bm{x}`. One advantage of this is that it works also for Greek symbols like `\alpha`, `\beta`, etc.

Powers and indices use the common syntax `x^2` and `x_2`. If more than one character is in a numerator or denominator, braces `{}` must be used, like `x^{2y}`.

Fractions are usually expressed as either `numerator/denominator` (for inline fractions) or `\frac{numerator}{denominator}` (for display fractions).

Sums and integrals can have the syntax `\sum_{i=1}^{n}` and `\int_0^t`.

Matrices can be included with the syntax (for a three-by-three)

```
\begin{bmatrix}
  m_{11}  &  m_{12}  \\
  m_{21}  &  m_{22}
\end{bmatrix}
```

Resource 1.8 Adding your own references

Resource 1.8.1 Setting up references database

Another powerful feature of LaTeX is the inclusion of references via BibTeX. The references you would like to include in your document should be added to the `.bib` file via a reference manager such as JabRef (Windows) or BibDesk (OS X). You can open your `.bib` reference database in one of these apps and add them manually or import them from the internet.

Google Books includes downloadable BibTeX references for the books it has in its database. Also, if you're referencing an academic paper, usually the journal's website allows you to download the BibTeX reference file.

On university computers you will be unable to install JabRef. However, it can be run from a flash/thumb drive. Download the `.jar` installation file here. Place the file on your flash drive. Double-click the file (now on the flash drive) and navigate to `Options > Preferences`, then, under the `General` section, check the box `Load and save references from/to jabref.xml on startup (memory-stick mode)`. Now you can run JabRef to manage your references on most computers—from your flash drive!

Both JabRef and BibDesk allow you to copy to the clipboard a text reference, like

```
@article{Fedrizzi2015,
Author = {Marcus Fedrizzi and Julio Soria},
Journal = {Measurement Science and Technology},
Number = {9},
Pages = {095302},
Title = {Application of a single-board computer
as a low-cost pulse generator},
Url = {http://stacks.iop.org/0957-0233/26/i=9/a=095302},
Volume = {26},
Year = {2015}}
```

and paste it into the main window. It should create a new entry for you.

Alternatively, you can simply directly edit the `report.bib` file. You can paste in a text reference, like the one shown above, at the end of the file. If you aren't editing it directly in Overleaf, make sure you use a plain text editor like `Notepad` on Windows or `TextEdit` on a Mac. (Don't use `WordPad`, `Word`, or any other "rich text" editor.) This is the quick and dirty method.

Once you have your reference in the database, you must make sure it has a BibTeX *key*. I usually make my keys the author's last name and the year of the publication, like `Picone2015` or `Mathers2000`. Each key must be unique, so sometimes you have to include a letter postfix like `Picone2015a` or `Picone2015b`. These keys are how we will refer to the reference in the text.

Resource 1.9 Including the references database in your document

In the template `report.tex` file I've pointed to the `report.bib` file in the lines

```
\bibliographystyle{plainnat}
\bibliography{report}
```

This bibliography style `plainnat` is a good one, so there's usually no need to tweak it. The location of the **`\bibliography`**`{}` command is where the bibliography section of the document will be included. It is usually after the main contents in the report, and before appendices.

Resource 1.10 Citing sources in the text

Now you're ready to cite sources in the text. The usual way to do this is at the end of a sentence or paragraph with the syntax

```
\citep[p.~77]{Picone2015}
```

The `[p.~77]` is an optional argument, but references a specific page number. Leave it out if you don't want to reference a page number. The key `Picone2015` identifies which source you're citing from your database. This command gives a parenthetical inline citation like (Picone, p. 77).

All the sources that you reference in your text will be listed automatically in the "References" section of your document. If there is a source in your database that you don't reference, it will not be listed.

Overleaf automatically handles updating your references. If you're using TeXShop as your editor with `pdflatexmk` as your build protocol, the references will be updated automatically on each build. If you're using TeXWorks as your editor with `pdflatex` as your build protocol, whenever you add a new reference to the text or change the details on a reference in your reference manager, you will have to build with `pdflatex` once, then build with `BibTeX` once, then build with `pdflatex` once or twice more. This will update all the references in your document. If you encounter a reference in your output pdf with a question mark like "(?)", it usually means you need to run `BibTeX` again.

Resource 1.11 Adding your own tables

LaTeX tables are pretty cumbersome. An example is included in the template file using the `tabularx` environment. Another useful environment is the standard `tabular`. I usually have to review this documentation when I make a table.

The following website provides a nice way to enter and edit your table before putting it into LaTeX syntax:

www.tablesgenerator.com.

Resource 1.12 Including full-page pdf documents

If you need to include one or more pages from a pre-existing, multi-page pdf document, you can use the following process. You need the `pdfpages` package and use the following command to include all pages

```
\includepdf[ % put this where you want the pdf
  pages=-, % all pages
  frame, % frame the pages
  scale=.9, % scale the pages
  pagecommand={\pagestyle{fancy}} % make it sexy
]{
  file.pdf % this is the pdf file name
}
```

If only certain pages are to be included, these can be listed as described in the following example. To include pages 1–3, a blank page, 5, and 9, use the command

```
\includepdf[pages={1-3,{},5,9}]{file.pdf}
```

Box 01.4 Overleaf or local?

If you'd like to build LaTeX documents locally, proceed with the rest of the tutorial. Otherwise, you're done!

Resource 1.13 Installing LaTeX locally

The Cebula Hall 101 computers have LaTeX (TeX Live) installed. The following is for those of you looking to install LaTeX on your personal computer.

For all platforms, I recommend installing the latest version of TeX Live. For each platform, there is a different method to do this.

If you're using a Mac, I recommend installing TeX Live via MacTeX.

If you're using a Windows machine, I recommend installing TeX Live by downloading `install-tl-windows.exe` here and following the directions on the page, which include the quick installation instructions.

If you're running a Unix-based OS, you're on your own, but you're used to it. (And it's actually pretty straightforward.)

Resource 1.14 Building documents

LaTeX provides a markup language that allows you to write a "codey" text document with extension `.tex` that LaTeX *builds* into a `.pdf` (usually). This is a different paradigm from the what-you-see-is-what-you-get (WYSIWYG) word processors (like MS Word) you've used. It requires a little learning curve, but it's well worth it in the end.

Resource 1.15 Editing the template files and your first build

Download the template from Overleaf. Extract the contents of the archive to a convenient directory.

There are multiple options on each platform for editing. On a MS Windows installation, I recommend using the app *TeXWorks*, which will be installed automatically with TeX Live. On OS X, I recommend the app *TeXShop*, which also comes bundled with TeX Live. These apps work in a similar manner that I will now describe.

You will be able to open the provided template file `report.tex` in your editor (TeXWorks or TeXShop). You *should* be able to just "build" immediately: ctrl + T in TeXWorks and cmd + T in TeXShop. You should see in another window a `.pdf` file generated. Congratulations!

Try changing some text from the document and building it again.

Resource 1.16 TeXWorks configurations

Near the green "play" button in the toolbar, select the dropdown menu. You will see several options, like pdflatex. This is the processing tool used

(primarily) to build the `.tex` file into a `.pdf`. If you use pdflatex, you should be fine, unless you need to update your bibliography or index. I will address the bibliography question below. To ensure all your changes have been included in the build, it's best to run pdflatex twice (occasionally thrice).

Resource 1.17 TeXShop configurations

Near the Typeset button in the toolbar, select the dropdown menu. You will see several options, like LaTeX and pdflatexmk. These are processing tools used (primarily) to build the `.tex` file into a `.pdf`. I recommend pdflatexmk for its ease of use. If you choose to use LaTeX, you will have to also have to execute the BibTeX processing tool on the file each time you change your references. You will also need to run LaTeX twice at the end.

Resource: 2 LabVIEW configuration

The LabVIEW configuration described in this resource is installed on the lab station computers in the Robotics Lab. Students need not concern themselves with it unless they would like to replicate it on their own machines, which requires that the (proprietary) LabVIEW software be obtained and is generally not recommended.

Resource 2.1 Software versions

Resource 2.2 Compatibility

Resource 2.3 Installation

Resource 2.4 Debugging

Issues and their resolution are documented in this section.

Resource 2.4.1 Error on myRIO deployment:

Resource: 3 myRIO configuration

MAX > Remote Systems > myrio > Software (right click) > Add/Remove Software

Then install Labview Real-Time **14.0.1** version for NI myRIO 15.0. The Labview Real-Time 15.0.0 version DOES NOT WORK FOR Labview 2014 (which is installed on the lab stations).

02

Circuits and their measurement

Lecture 02.01 Supplying power

voltage sources
current sources
supplying power
supplying signals
supply loading

In the lecture course we have seen that we typically use two ideal models of power supplies or sources: *voltage sources* and *current sources*. In this lecture, we'll survey actual devices used to supply power. It's worth noting that there's typically a distinction made between supplying *power* and supplying *signals*: the latter are usually voltage of a specific waveform, but limited in current. In other words, devices that supply signals typically can't be *loaded*.

02.01.1 DC power supply

dc power supply

A *dc power supply* is an instrument that provides dc (constant) voltage and current to a circuit. These devices are sometimes active and use feedback to hold their outputs at the specified voltage or current—we call such power supplies *regulated*. Note that regulated supplies can be treated as ideal sources (voltage or current, whichever is applicable) within their power specifications.

regulated power supply

We will be using the *RSR HY3005* regulated power supply, with front panel shown in Figure 02.1. Its manual can be found here:

`ricopic.one/resources/HY3005.pdf`.

Figure 02.1: the front panel of the HY3005 dc power supply.

In order to use the HY3005 as a voltage source, use the following procedure, beginning with the POWER button in the OFF position and nothing connected to the output terminals.

1. Turn the CURRENT FINE and COURSE knobs completely clockwise (on).
2. Toggle the POWER button to ON.
3. Adjust the VOLTAGE FINE and COURSE knobs until the desired voltage is generated (as shown on the display).
4. Connect the + and − terminals. (Optionally, jumper the GND and − terminals.)

02.01.2 Function generator

A *function generator* is a device that generates a voltage signal. Most of these devices can produce sinusoidal, square, triangle, sawtooth, and other signals. The *frequency* and *amplitude* of the signals can be adjusted, as well.

Function generators are typically *not regulated*. Their outputs are typically BNC connectors with 50 Ω output resistance. Therefore, a function generator should be modeled as an ideal voltage source in series with its output resistance.

The nominal voltage displayed on the front panel of the function generator *assumes some specific load*, typically either 50 Ω or an infinite resistance. When working with a function generator, either consult the manual or measure its actual voltage output.

Figure 02.2: the front panel of the Tektronix AFG1062 function generator.

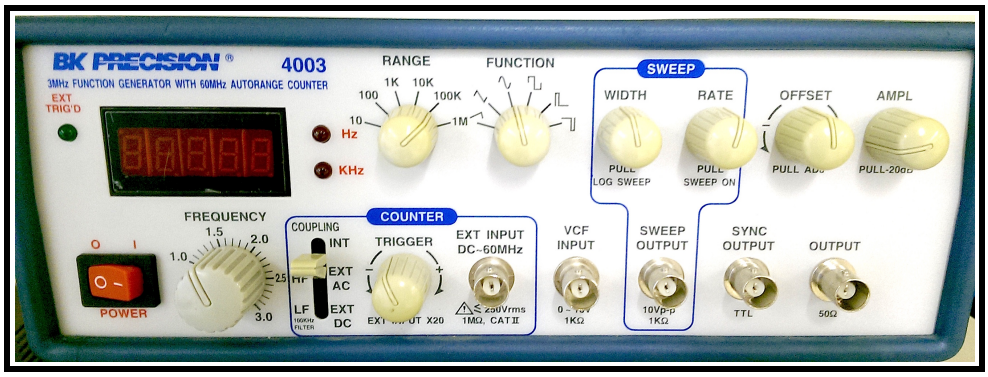


Figure 02.3: the front panel of the BK Precision function generator.

We will be using the *Tektronix AFG1062* function generator, with front panel shown in Figure 02.2. Its manual can be found here:

`ricopic.one/resources/afg1062.pdf`.

Also available in the lab is the *BK Precision 4003* function generator, with front panel shown in Figure 02.3. Its manual can be found here:

`ricopic.one/resources/BK_4003.pdf`.

The BK Precision has a readout of the frequency, but the amplitude is not given. Therefore, we must measure it ourselves!

Lecture 02.02 Measuring voltage

Voltage can be measured with several different instruments, used in different measurement applications.

02.02.1 Multimeter

A multimeter, such as the Fluke shown in Figure 02.4 can be used to measure voltage, current, resistance, and sometimes capacitance. These small, usually hand-held devices have two probes, used in two different connections: high-resistance and low-resistance.

high-resistance connection Each probe is connected to a location on the circuit in order to measure the voltage, resistance, or capacitance *across* the circuit connecting the locations. The hot probe is connected to the V or Ω port.

low-resistance connection In order to measure the current *through* the multimeter, circuit must first be broken and the multimeter placed into the circuit. The hot probe is connected to the A, mA, or μA ports.

The dial selects the *measurement mode*. The multimeter shown in Figure 02.4 is in dc voltage-measurement mode.

Figure 02.4: the front panel of a Fluke multimeter, connected in high-resistance mode.

measurement mode

Going from bottom-left, clockwise, the modes are, with high-resistance connection assumed unless otherwise stated:

1. ac voltage,
2. dc voltage,
3. dc voltage—small signal,

Figure 02.5: the front panel of the Tektronix DPO 2012B oscilloscope.

4. resistance,
5. diode test mode,
6. current (low-resistance connection), and
7. small current (low-resistance connection).

02.02.2 Oscilloscope

An oscilloscope is an instrument that measures and displays a voltage signal through time. It allows us to "zoom-in" on a window of time, through which the signal is diplayed. If the oscilloscope is properly triggered, the signal will appear stationary. This occurs when the window is set to a time interval that is an integer multiple of the signal's period. This allows the signal to trace over itself and appear continuous.

We are using the Tektronix DPO 2012B oscilloscope, with front panel shown in Figure 02.5. Its manual can be found here:

`ricopic.one/resources/dpo2012b.pdf`.

There are two inputs, `1` and `2`. This oscilloscope has advanced functionality, including cursors for precisely measuring waveform properties, signal averaging, signal math, fast-fourier transform (FFT), and trace capturing.

Also in the lab is the BK Precision 2120B oscilloscope, with front panel shown in Figure 02.6. Its manual can be found here:

Figure 02.6: the front panel of the BK Precision 2120B oscilloscope.

`ricopic.one/resources/BK_2120B.pdf`.

There are two inputs, `CH1` and `CH2`. This oscilloscope is old-school and lacks modern features such as cursors and trace capture, but it has everything we need to understand the primary functionality of oscilloscopes.

Lecture 02.03 Circuit prototyping

When one is building a circuit for a one-time application or for a prototype, it is best to use a temporary construction. Production circuits are almost invariably constructed as *printed circuit boards* (PCBs), like that depicted in Figure 02.7, which can be cheaply and reliably mass-produced. PCBs have recently become economical for even low-volume projects, with custom orders costing only tens of dollars per board, or less (plus components).

printed circuit boards

If a lightweight and reliable prototyped circuit is required, constructing the circuit via soldered wires and components is usually best. However, even before this, a more flexible prototype is often required.

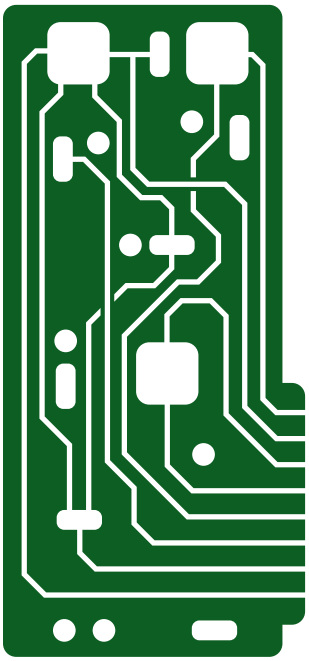


Figure 02.7: a depiction of a printed circuit board, adapted from openclipart Cheeseness (2018).

02.03.1 Breadboards

Breadboards, such as the one shown in Figure 02.8, are plastic-and-metal boards that have several pin holes for easily building temporary circuits. Pins and leads can be inserted into the breadboard holes to make connections.

power rails

The *bus strips* or *power rails* on each side of the boards are marked with

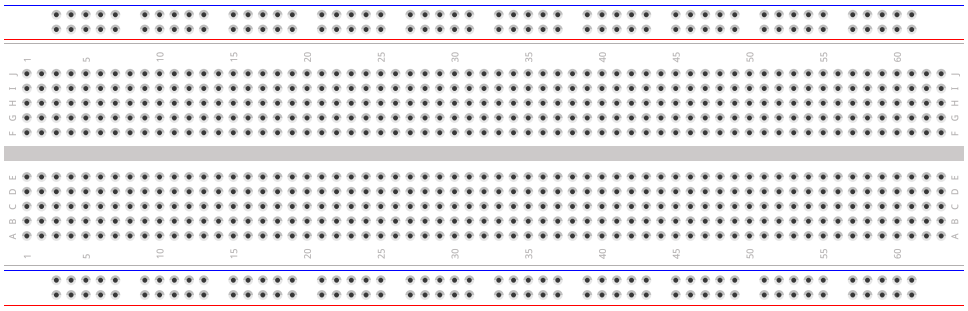


Figure 02.8: a breadboard with two each of blue and red power rails.

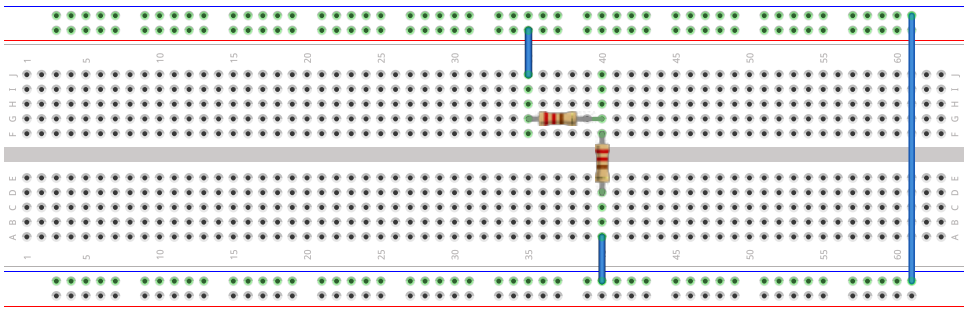


Figure 02.9: a breadboard with a voltage-divider circuit.

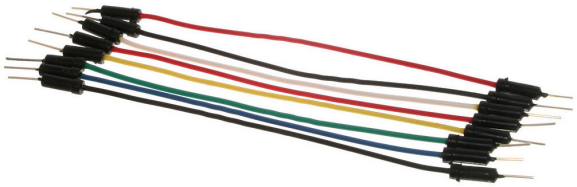


Figure 02.10: jumper wires for wiring breadboard circuits (Commons, 2018b).

red (+, hot) and blue (−, ground) stripes, each being connected along its row.

The numbered *terminal strips* run vertically in the center of the board. Numbered columns are connected, but split into two halves by the center notch such that each side of the notch is disconnected. The center notch is conveniently spaced such that standard *integrated circuits* or *chips* can be connected across it.

terminal strips

integrated circuits

An example of a voltage divider circuit is shown in Figure 02.9. Two resistors are shown, sharing a node at row 40. Power is connected to the upper power rails, and the upper ground rail is bridged to the lower ground rail. In addition to the built-in connections among breadboard holes, *jumper wires*, such as those in Figure 02.10, are often used to make connections among holes.

jumper wires

02.03.2 Coaxial cables

coaxial cables

Connecting instruments such as function generators and oscilloscopes to a breadboard circuit requires *coaxial cables*, which have an inner hot wire surrounded by but insulated from a tubular shield that conducts the ground.

Therefore, when an instrument is connected to a circuit via a cable, it is connected *in parallel*.

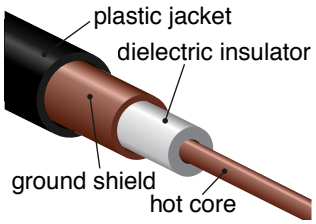


Figure 02.11: coaxial cable, adapted from Commons (2018a).

02.03.3 BNC connectors

BNC connectors

In instrumentation, coaxial cables almost universally have *BNC connectors*, shown in Figure 02.12. On the left is a male connector and in the center and on the right are two views of a female connector. The male connector is spring-loaded such that mating the two with a twist gives a reliable connection that can easily be felt by the user.

Figure 02.12: BNC connectors, adapted from openclipart boudinpg (2018).

The outer "barrels" of both male and female connectors are connected to the coaxial ground shield.

02.03.4 Alligator clips

alligator clips

In order to connect coaxial cables to breadboards, connectors with *alligator clips*, shown in Figure 02.13, are often used. These each have a single conductor, with a black clip typically being used for ground and a red clip for hot. Often, the will clip directly to a jumper wire.

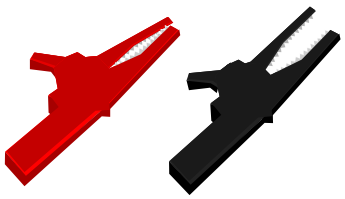


Figure 02.13: alligator clips, adapted from openclipart goios (2018).

Connections with alligator clips are convenient, but relatively insecure. They should be used only in very temporary prototypes and never when the circuit is to undergo acceleration.

Lab Exercise 02 Voltage Dividers

The objectives of this lab exercise are for students:

- ☐ 1. to understand voltage and signal generation,
- ☐ 2. to understand voltage and signal measurement,
- ☐ 3. to become familiar with the instruments used for voltage and signal generation and measurement,
- ☐ 4. to become familiar with prototyping circuits,
- ☐ 5. to deepen an understanding of voltage dividers,
- ☐ 6. to model real circuits,
- ☐ 7. to learn to acquire voltage measurements with the myRIO, and
- ☐ 8. to learn to plot and export plots in MATLAB.

Lab 02.1 Materials

The following materials are required for each lab station:

1. a PC with LabVIEW installed,[1]
2. a myRIO configured with LabVIEW,[2],
3. a multimeter,
4. a dc power supply,
5. a breadboard,
6. jumper wires,
7. the following resistors:
   - ☐ 1. two 1.5 MΩ,
   - ☐ 2. one 2.2 MΩ,
   - ☐ 3. one 3.3 MΩ, and
   - ☐ 4. one 4.7 MΩ.
8. a function generator,
9. an oscilloscope,
10. two BNC cables, and
11. a BNC Y- or T-connector.

Lab 02.2 Building a voltage divider circuit

- ☐ 1. Measure and record the actual resistance of each resistor (use a multimeter).

[1] See Resource 2 for more details on the LabVIEW software configuration.
[2] See Resource 3 for more details on the myRIO software configuration.

	R_1	R_2	R_3	R_4	R_5
nominal (MΩ)	1.5	1.5	2.2	3.3	4.7
measured (MΩ)					

☐ 2. Connect the two 1.5 MΩ resistors in series.
☐ 3. Connect the dc power supply across the series resistors.
☐ 4. By following the procedure in Section 02.01, set the power supply as a voltage source at 10 V.

Lab 02.3 Measuring with a multimeter voltage dividers powered by a dc power suppy

Use the voltage divider circuit built in the previous section and a multimeter to make measurements.

☐ 1. Measure and record the voltage across each resistor, separately.

	v_{R_1}	v_{R_2}
measured (V)		

☐ 2. Measure and record the voltage across both resistors, together (this is the same as measuring the source voltage).

	$\widetilde{V}_s$
measured (V)	

☐ 3. Record the voltage and current readings on the dc power supply.

	V_s (V)	I_s (mA)
readings		

output resistor

☐ 4. Replace the R_2 resistor (let's call it the *output resistor*) with a 2.2 MΩ resistor and repeat and record (in the table below) the voltage measurement.
☐ 5. Repeat for the 3.3 MΩ and 4.7 MΩ resistors.

When you've finished the above steps, you should have data to fill in Table 02.1.

output res.	R_2	R_3	R_4	R_5
nominal R_i (MΩ)	1.5	2.2	3.3	4.7
measured $\widetilde{R}_i$ (MΩ)				
measured $\widetilde{V}_s$ (V)				
measured $\tilde{v}_{R_i}$ (V)				

Table 02.1: table of voltage divider measurements from the multimeter for each output resistor.

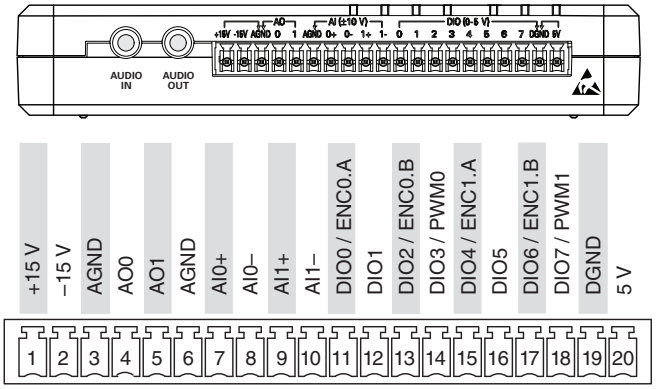


Figure 02.14: myRIO Connector `C` (from Instruments (2013)).

Lab 02.4 Measuring and powering voltage divider circuits with a myRIO

- ☐ 1. Build your original voltage divider circuit with two 1.5 MΩ resistors in series, but this time do not connect the dc power supply to the circuit. As before, define the "free" terminal of R_2 to be ground.
- ☐ 2. Connect a myRIO to power and to your workstation computer via USB.
- ☐ 3. We will be using the MSP connector named `C` on the side of the myRIO shown in Figure 02.14.
- ☐ 4. With jumper wires, connect the analog output ground channel `AGND` (3) to the ground of your circuit (choose ground to be one of the "free" ends of the series resistors).
- ☐ 5. Connect the analog output channel `AO1` (5) to the opposite end of the series resistors such that the analog voltage supplied via `AO1` and

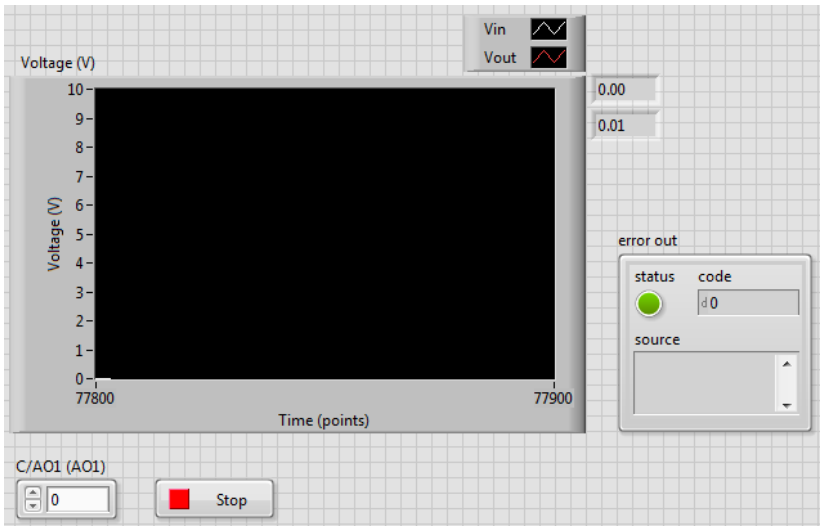


Figure 02.15: how the front panel of your VI might look.

`AGND` is applied across both resistors, together.

☐ 6. Connect both `AGND` (6) and `AI0-` (8) to the circuit's ground.

☐ 7. Connect `AI0+` (7) to the node shared by the two resistors—call the resistor you've just connected between `AI0+` and `AGND` your "output" resistor (it should be R_2 at this point).

☐ 8. Create a new LabVIEW myRIO project. Edit `Main.vi` such that it has the following functionality:

- ☐ a. outputs an analog voltage from `AO0` that can be updated from the front panel, continuously;
- ☐ b. measures the analog voltage from `AIO+`, continuously;
- ☐ c. plots the output and input voltages on the same chart, continuously; and
- ☐ d. displays digital readouts of the current voltage output and input, continuously. The front panel of Figure 02.15 and corresponding block diagram of Figure 02.16 show one way of realizing this VI.

☐ 9. Using your VI, set the source voltage to each of the values $(0, 1, \cdots, 10)$ V. Manually record the corresponding voltage measurements in Table 02.2.

☐ 10. Repeat these measurements for output resistors with nominal resistances 2.2 MΩ, 3.3 MΩ, and 4.7 MΩ. Manually record the corresponding voltage measurements in Table 02.2

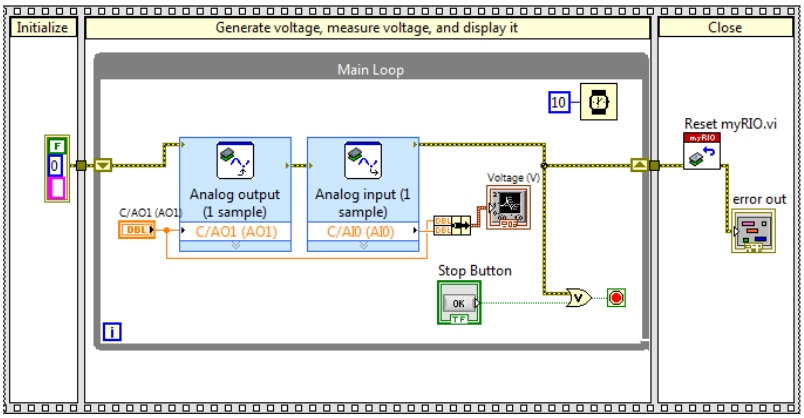


Figure 02.16: how the block diagram of your VI might look.

	V_s (V)	0	1	2	3	4	5	6	7	8	9	10
R_2	$\widetilde{V}_s$ (V)											
	$\tilde{v}_{R_2}$ (V)											
R_3	$\widetilde{V}_s$ (V)											
	$\tilde{v}_{R_3}$ (V)											
R_4	$\widetilde{V}_s$ (V)											
	$\tilde{v}_{R_4}$ (V)											
R_5	$\widetilde{V}_s$ (V)											
	$\tilde{v}_{R_5}$ (V)											

Table 02.2: table of voltage divider measurements from the myRIO for each output resistor.

Lab 02.5 Signals generated and measured

☐ 1. Switch the function generator to the following settings:

- ☐ a. sine wave,
- ☐ b. frequency: 10 kHz, and
- ☐ c. 5 V_{pp}.

☐ 2. Connect the function generator `OUTPUT` to `CH1` of the oscilloscope. Make sure your oscilloscope (especially `CH1`) has the following settings:

- ☐ a. ac coupling,
- ☐ b. `VERT MODE` either `CH1` or `DUAL`,
- ☐ c. `TRIGGER` on `AUTO` and `CH1`, and
- ☐ d. multiplication factor `1X`.

☐ 3. Adjust the oscilloscope settings until a stable wave is found.
☐ 4. Estimate and record the peak-to-peak amplitude and period from the oscilloscope screen. (Use the cursors!)

$\widetilde{V}_s = \tilde{v}_{R_2}$ (V_{pp})	T (ms)

☐ 5. Change the function generator signal waveform to a square wave, then a triangle wave, then a sawtooth. Finally, change it back to a sine wave.
☐ 6. Decrease the function generator frequency to 10 Hz.
☐ 7. Adjust the `TIME/DIV` on the oscilloscope until you can see the dot tracing across the screen.
☐ 8. Profit.

Lab 02.6 Report requirements

Write a report on your laboratory activities using the template given. Include the following elements:

☐ 1. A **circuit analysis** of each circuit configuration (using generic resistor values, e.g. R_1 and R_2). Don't forget to properly model each source!
☐ 2. A **plot of multimeter measurements** of voltage input ($\widetilde{V}_s$) and output ($\tilde{v}_{R_i}$) versus resistor value. Use markers only, like $\times$ or $\circ$. Include on the same plot corresponding **theoretical predictions** based on measured input voltage and measured resistance values (i.e. don't use the nominal values). For these theoretical predictions, use continuous lines without markers.
☐ 3. A **plot of myRIO measurements** (manually recorded) of measured input voltage $\widetilde{V}_s$ versus output voltage $\tilde{v}_{R_i}$ for each output resistor R_i. Include on the same plot corresponding **theoretical predictions** based on $\widetilde{V}_s$ and R_i.

□ 4. Include a **table** of all **multimeter measurements** (resistance and voltage).

It may be helpful to take photos during the laboratory procedure. These can be included in your report.

Resource: 4 Generating and exporting figures in MATLAB

Once you've entered your data into arrays in MATLAB, you can easily generate plots with syntax such as:

```
%% fake some data

x = linspace(0,10,50); % dummy independent variable
y = x.^2; % dummy dependent variable
z = y+sin(y); % another dummy dependent variable

%% plot

figure % opens a new figure
plot(...
  x,y,... % data
  'bx'... % LineSpec
);
hold on % Hodor figure for more plots
plot(...
  x,z,... % more data
  'ro'... % LineSpec
);
hold off % Hodor is sacrificed
grid on % turn on grid
xlabel('R_i (M\Omega)')
ylabel('voltage (V)')
legend(... % add a legend!
  'first data',... % label for first trace
  'second data'... % label for second trace
)
```

For more details on the appearance of plot traces, called `LineSpec` in MATLAB, see its documentation here:

`mathworks.com/help/matlab/ref/linespec.html`.

So you have your plot, but how do you get it into your LaTeX report? You need to export it from MATLAB as a pdf. The advantage of using a pdf and not a rasterized graphic (e.g. jpg) is that the quality of the output is "vector" and doesn't look pixelated. However, if you use MATLAB's

GUI interface to export the figure, you'll be disappointed to find it yields full-page figures—hardly conducive to including in your report!

Fortunately, there's a nice function `save2pdf` available here:

`ricopic.one/courses/me316_2018F/resources/save2pdf.m`.

The following procedure will get you started with this function.

- ☐ 1. Download the m-file (`save2pdf.m`) with the link, above.
- ☐ 2. Copy `save2pdf.m` to the same directory as your main MATLAB m-file. The function `save2pdf` is now available to your main MATLAB m-file.
- ☐ 3. Use the following command in your main MATLAB m-file to save the most recently generated figure (`gcf`) to a pdf. This will save the figure as the file `figure-file-name.pdf` in your current directory.

```
save2pdf('figure-file-name',gcf,300)
```

- ☐ 4. After each figure you'd like to save, call the `save2pdf` function in the same way, changing the filename, appropriately.

Now your figure is a nice pdf `figure-file-name.pdf`. Upload it to Overleaf and include it in your document in the usual way (`\includegraphics{figure-file-name.pdf}`). For more details, review Resource 1.4.

03

RC Circuit Time Response

Lab Exercise 03 RC Circuit Response

In this lab, we will be measuring the time-response of the voltage across a capacitor in an RC circuit, such as that of Figure 03.1. We will apply an input voltage and measure the v_o response with an NI myRIO device, export it to a data file, import that data file into MATLAB, plot the data, derive an analytic model, and compare the analytic model to the data.

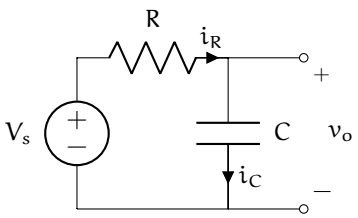


Figure 03.1: RC circuit

The objectives of this lab exercise are for students:

1. to explore transient circuit response,
2. to deepen their understanding of RC circuits,
3. to learn to acquire and log data with a microcontroller board (myRIO),
4. to learn to export the acquired data to MATLAB,
5. to model real circuits and compare the theory and experiment, and
6. to learn to better plot and export plots in MATLAB.

Lab 03.1 Materials

The following materials are required for each lab station:

☐ 1. a PC with LabVIEW installed,[1]
☐ 2. a myRIO configured with LabVIEW,[2],
☐ 3. a multimeter,
☐ 4. a breadboard,
☐ 5. jumper wires,
☐ 6. a 100 kΩ resistor,
☐ 7. a 10 μF capacitor,
☐ 8. one BNC Y- or T-connector,
☐ 9. one BNC cable, and
☐ 10. two BNC-to-alligator cables.

Lab 03.2 Build the circuit

Use the following procedure to build the circuit.

[1] See Resource 2 for more details on the LabVIEW software configuration.
[2] See Resource 3 for more details on the myRIO software configuration.

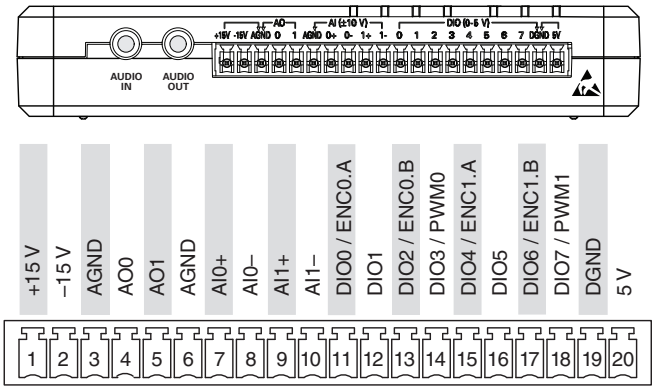


Figure 03.2: myRIO Connector `C` (from Instruments (2013)).

☐ 1. Measure and record the actual resistance R of the resistor and capacitance C of the capacitor with a multimeter.

	R (kΩ)	C (μF)
nominal	100	10
measured		

☐ 2. Build the RC circuit (sans source) on a breadboard. The capacitor is **polarized**; that is, it has a positive and a negative terminal. We must take care to connect such capacitors such that the voltage always drops from the positive terminal to the negative terminal. Sometimes this is indicated by a shorter lead and/or a "−" symbol on the negative terminal. (In our case "−" should connect to the ground node.)

☐ 3. Connect the myRIO to power and to your workstation computer via USB. It will be both the voltage source V_s and measurement of v_o.

☐ 4. We will be using the MSP connector named `C` on the side of the myRIO shown in Figure 03.2.

☐ 5. With jumper wires, connect the analog output ground channel `AGND` (`3`) to the ground of your circuit (choose ground to be the "free" end of the capacitor).

☐ 6. Connect the analog output channel `AO0` (`5`) to the free end of the resistor such that the analog voltage supplied via `AO1` and `AGND` is applied across both resistors, together. You now have V_s supplied by analog output `AO1`.

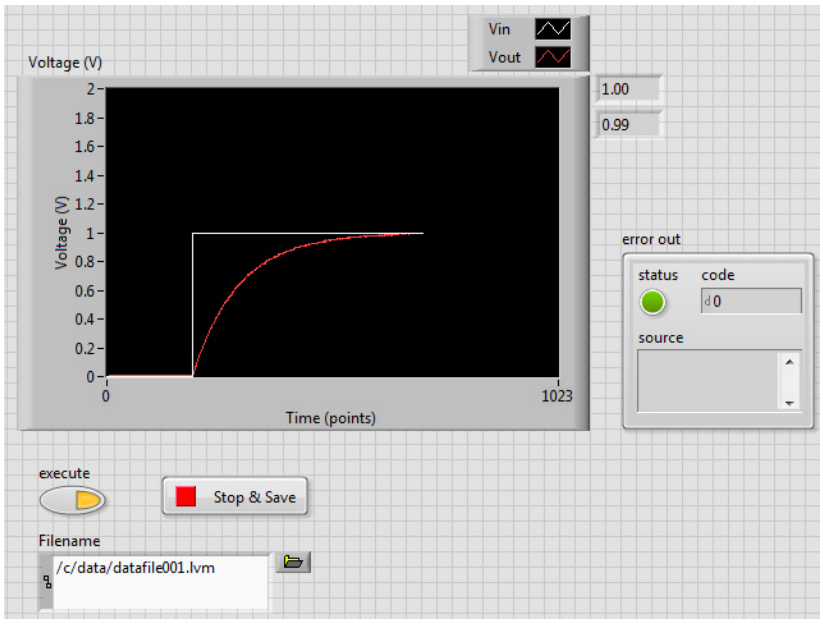


Figure 03.3: how the front panel of your VI might look.

☐ 7. Connect `AGND` (`6`) and `AI0-` (`8`) to the circuit's ground.
☐ 8. Connect `AI0+` (`7`) to the node shared by the resistor and capacitor. You now have a measurement of v_o on analog input `AI0`.
☐ 9. Connect both `AGND` (`6`) and `AI1-` (`10`) to the circuit's ground.
☐ 10. Connect `AI1+` (`9`) to the end of the resitor to which `AO1` is applied. You now have a measurement of V_s on analog input `AI1`.

Lab 03.3 Create a LabVIEW project and Main VI

Create a new LabVIEW myRIO project. Edit `Main.vi` such that it has the following functionality:

☐ 1. outputs an analog voltage of 1 V from `AO0` when the user toggles a button named execute;
☐ 2. measures the voltage across the capacitor using `AI1+`, continuously;
☐ 3. measures the analog output voltage using `AI0+`, continuously;
☐ 4. plots the command, output, and input voltages on the same chart, continuously;
☐ 5. displays digital readouts of the current command, voltage output, and input, continuously;

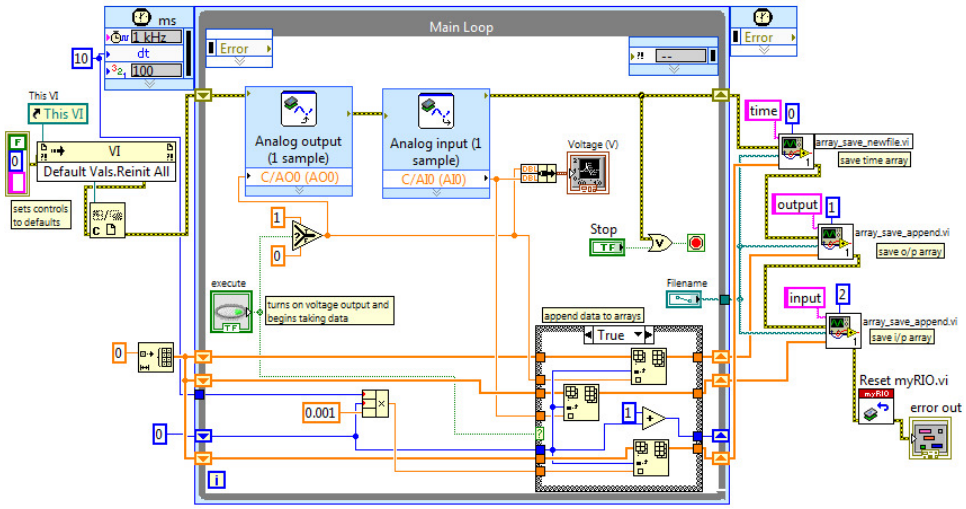


Figure 03.4: how the block diagram of your VI might look. Note that this example does not include a measurement of the source $\widetilde{V}_s$.

- ☐ 6. uses a *timed-loop* that executes once every 10 ms; **timed-loop**
- ☐ 7. after (and only after) the execute button is pressed, store time values (with zero beginning when the button is pressed), measured circuit input voltage $\widetilde{V}_s$, and measured circuit output voltage $\tilde{v}_o$ in arrays; and
- ☐ 8. after the timed loop is exited via a Stop & Save button, write the three arrays to a LabVIEW `.lvm` datafile on the myRIO via the custom VIs I've written for you `array_save_newfile.vi` and `array_save_append.vi`, which are zipped here:

  `ricopic.one/resources/array_savers.zip`.

  (unzip and move them to the same directory as your project);

The front panel of Figure 03.3 and corresponding block diagram of Figure 03.4 show a way of realizing a very similar VI. The difference is that the example has no measurement of $\widetilde{V}_s$ (it assumes the circuit doesn't load the source).

The code that *preceeds* the Main Loop in Figure 03.4 resets the controls on the front panel to their default values. The block labeled This VI is the VI Server Reference VI and is found in the Functions > Application Control palette.

The block labeled `Defaults Vals.Reinit All` is the Invoke Node VI and is found in the Functions ⟩ Application Control palette. After wiring the VI Server Reference to the input of Invoke Node, right-click on the word `Method` on the Invoke Node VI and select Method for VI Class ⟩ Default Values ⟩ Reinitialize All to Default. Finally, the block labeled `C` is the Close Reference VI and is found in the Functions ⟩ Application Control palette.

The Case Structure labeled `append data to arrays` has a `False` panel that simply passes through arrays instead of appending to them.

Box 03.1 See Resource before proceeding

See Resource 5 before proceeding. It will help you prepare the myRIO so you can download data from it to the desktop computer.

Lab 03.4 Capture the capacitor voltage response

The following procedure can be used to capture the charging of the capacitor.

1. Name your data file on the front panel something like `/c/data/datafile001.lvm`.
2. Run your `Main.vi`. You should see a zero voltage V_s and v_0 reading on the chart.
3. Click the execute button to begin application of V_s and data acquisition of v_o.
4. Observe the voltage response on the chart. You should observe an exponentially decaying approach of $\tilde{v}_o$ to 1 V.
5. Once the voltage has leveled off (at least about $6\tau = 6RC$), click the Stop & Save button on the front panel of the VI.
6. Check to see if the data file was saved to the myRIO flash memory. Use Windows Explorer to navigate to the mapped network drive `http://172.22.11.2/files`. Navigate to `/c/data`. Each run, a data file is created (or over-written) here with a name specified on your front panel. Old data files may be present. Be sure that you identify your data file.
7. Copy your data file to a convenient directory on the computer (e.g. `~/Documents/me316/lab03`).

Lab 03.5 Report requirements

Write a report on your laboratory activities using the template given.

- ☐ 1. Include the following plots (use MATLAB to make the plots):
  - ☐ a. The measured $\widetilde{V}_s$ and $\tilde{v}_o$ as functions of time and
  - ☐ b. The measured $\tilde{v}_o$ as a function of time overlayed with your theoretical prediction for v_o.
  - ☐ c. A log-linear plot of the measured $\tilde{v}_o$ as a function of time overlayed with your theoretical prediction for v_o and your linear regression of the data.
- ☐ 2. Include all measurements made by hand (e.g. resistance).
- ☐ 3. Include an analysis of a voltage RC circuit that can be used to produce the theoretical predictions in the plots.
- ☐ 4. Compute your measured time constant $\tau = RC$ by performing a linear regression on your measured $\tilde{v}_o$ data. The data must be processed before a linear regression can be performed! (It's not linear yet, it's exponential.)
- ☐ 5. Include, as always, an abstract, introduction, materials and methods, results, and discussion.

It may be helpful to take pictures during the laboratory procedure. These can be included in your report.

Resource: 5 Preparing the myRIO

The myRIO must be connected to power and to the computer via USB. We will need to access the on-board flash memory of the myRIO from the computer. If under `Computer` in Windows Explorer no network drive `http://172.22.11.2/files` is available, it must be added. If required, map the network drive as follows.

- ☐ 1. Navigate to `Computer` in Windows Explorer.
- ☐ 2. Click (near the toolbar) Map network drive.
- ☐ 3. Enter the following path: `http://172.22.11.2/files`.
- ☐ 4. Check both Reconnect at login and Connect using different credentials.
- ☐ 5. Click Finish. Wait.
- ☐ 6. If asked for a username and password, enter the username `admin` and leave the password blank. (If you're using myRIO #4 for some reason the password also is `admin`.)
- ☐ 7. Click Ok. Wait.
- ☐ 8. A new network drive `http://172.22.11.2/files` should appear.
- ☐ 9. Navigate to `/c/data`. This is where your data files will appear.

Resource: 6 Importing, processing, and representing data in MATLAB

We will import the `.lvm` file into MATLAB with the MATLAB function `lvm_import`.

☐ 1. Download the following `.zip` file:

`ricopic.one/resources/lwm_import_v22.zip`.

☐ 2. Extract the contents of the `.zip` archive. Do this by navigating to the file, right-clicking it, and selecting Extract all... **or** "opening" it (viewing its contents in Windows Explorer) and clicking the button Extract all files, as shown in Figure 03.5.

☐ 3. Copy the file `lvm_import.m` to either your `~/Documents/MATLAB` directory **or** the directory in which your data file (`.lvm`) resides (e.g. `~/Documents/me316/lab03`).

☐ 4. Create a MATLAB script and save it *in the same directory as your data file*. Within the script, we can load the data file with the following command.

```
data_struct = lvm_import('your_file_name.lvm');
```

You now have a `struct` class variable `data_struct` from which we can extract the data with the following commands.

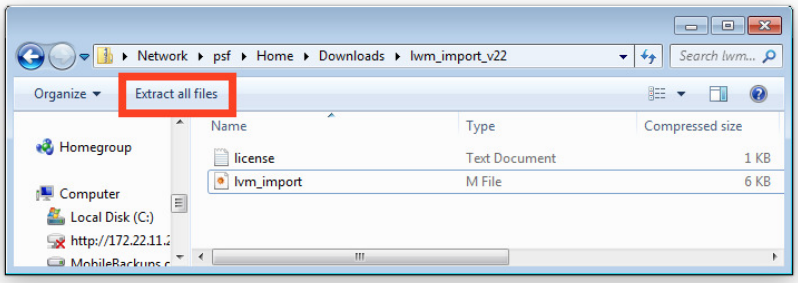


Figure 03.5: extracting files from the `lwm_import_v22.zip` archive.

```
myrio_t = data_struct.Segment1.data; % time array
myrio_output = data_struct.Segment2.data; % myrio output
myrio_input = data_struct.Segment3.data; % myrio input
```

You now have three array variables `myrio_t`, `myrio_output` ($\widetilde{V}_s$), and `myrio_input` ($\widetilde{v}_o$).

Resource 6.1 Plotting the data

Now we can plot the results. Here's an example of how that can be done.

```
figure;                    % open a new figure
plot(myrio_t,myrio_output,'xr'); % plot myrio output (Vs)
hold on;
plot(myrio_t,myrio_input,'xb'); % plot myrio input (vo)
grid on;                   % turn the grid on
xlabel('time (s)')         % label the x-axis
ylabel('voltage (V)')      % label the y-axis
legend('V_s','v_o')
```

04

RLC Circuit Frequency Response

Lab Exercise 04 RLC Frequency Response

In this lab exercise, we will be measuring the steady-state response of a capacitor voltage in the RLC circuit shown in Figure 04.1 with an AC input. We will record the data by hand, enter it into a data file in MATLAB, plot the data, derive an analytic model, and compare the analytic model to the data.

The objectives of this lab exercise are for students:

1. to explore steady state circuit frequency response,
2. to deepen their understanding of RLC circuits,
3. to learn to better measure AC voltage with an oscilloscope,
4. to model real circuits and compare the theory and experiment, and
5. to get a sense of resonance in a circuit.

Lab 04.1 Materials

The following materials are required for each lab station:

- ☐ 1. a function generator,
- ☐ 2. an oscilloscope,
- ☐ 3. a multimeter,
- ☐ 4. a breadboard,
- ☐ 5. four male-male jumper wires,
- ☐ 6. a 78 Ω resistor,
- ☐ 7. a 100 nF capacitor (label: `104`),
- ☐ 8. a 10 mH inductor,
- ☐ 9. one BNC Y- or T-connector,
- ☐ 10. one BNC cable, and
- ☐ 11. two BNC-to-alligator cables.

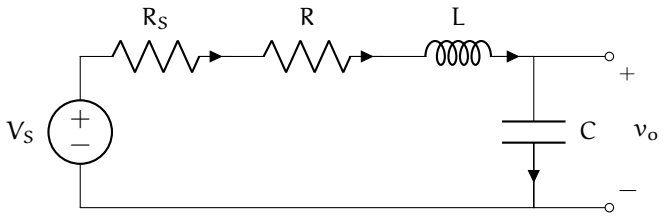


Figure 04.1: a model of an RLC circuit with a source model including the source output impedance R_S. Since the function generator is not a regulated source, we must consider its output impedance.

Lab 04.2 Build the circuit

The following steps describe how to build the RLC circuit of Figure 04.1 on the breadboard. Note that the output resistance R_S of the source is internal to the function generator.

- ☐ 1. Measure and record the actual resistance R of the resistor and capacitance C of the capacitor with a multimeter.
- ☐ 2. Build the passive portion of the circuit on a breadboard.
- ☐ 3. Use the function generator as the voltage source V_S. Its output resistance is $R_S = 50\ \Omega$. Use a BNC T- or Y-connector to split the output of the output such that one cable connects directly to the oscilloscope's Channel 1 and the other connects (via alligator clips and jumpers) across the circuit.
- ☐ 4. Into Channel 2 of the oscilloscope, connect another cable that has alligator clip probes connected to jumpers probing $v_o(t)$.

Lab 04.3 Measure the steady state response at different frequencies

In steady state, this linear circuit with a sinusoidal input will have a sinusoidal output. The only difference between the input and output signals will be in amplitude and phase. It is these two quantities we will measure in each signal.

- ☐ 1. Set the function generator to a sinusoidal output around 100 Hz and record the actual frequency. (Take care that the output channel above the connector is pressed to turn it on!) Make sure you can see the input and output sinusoids, simultaneously. Measure and record the peak-to-peak amplitudes of the *input* ($\widetilde{V}_S - \tilde{v}_{R_S}$) and the *output* ($\tilde{v}_o$). Also measure and record the *time lag* between input and output. (That's four recorded measurements, folks.)
- ☐ 2. Repeat these measurements and record the results in Table 04.1 for inputs near the input frequencies listed in the table (you don't need to be that close, just record the actual frequencies).
- ☐ 3. By fiddling with the input frequency, find a frequency that yields the highest output amplitude and make a final set of measurements at that frequency. Don't forget to include this in your data (it will be obvious in your report plots).

Box 04.1 Tip for those using the BK Precision 2120B oscilloscope

If you're using the BK Precision 2120B oscilloscope, don't use the fine-tuning knobs on the vertical scale knobs of the oscilloscope. Instead, rotate them all the way clockwise until they "click" for calibrated voltage operation.

Box 04.2 Tip for those using the BK Precision 4003 function gen.

If you're using the BK Precision 4003 function generator, your frequency control isn't very precise, but don't worry: you can measure it easily by taking the reciprocal of the period on the oscilloscope!

Lab 04.4 Report requirements

Write a detailed report of your experimental results, as outlined in the report template. Pay special attention to the following.

- ☐ 1. A *thorough* description of the theoretical analysis of the circuit using impedances. Make sure you derive the *amplitude ratio* $r(\omega)$ of output over input as a function of source frequency ω. Also (along the way) derive the theoretical phase shift $\phi(\omega)$ between input and output.
- ☐ 2. A figure with the *theoretical* r plotted versus logarithmic frequency (use the measured R and C values) overlayed with the measured amplitude ratio $\tilde{r}$ data.
- ☐ 3. A figure with the *phase shift* ϕ plotted versus logarithmic frequency overlayed with the measured phase shift $\tilde{\phi}$ data.
- ☐ 4. The other parameters measured in the lab (e.g. R and C).

f(Hz)	$\tilde{f}$ (Hz)	$\widetilde{V}_S - \tilde{v}_{R_S}$ (V_{pp})	$\tilde{v}_o$ (V_{pp})	$\tilde{\Delta}$ (ms)
100				
127				
162				
206				
263				
335				
428				
545				
695				
885				
1128				
1438				
1832				
2335				
2976				
3792				
4832				
6158				
7847				
10000				
peak				

Table 04.1: table of RLC circuit measurements: frequency $\tilde{f}$, input amplitude $\widetilde{V}_S - \tilde{v}_{R_S}$, output amplitude $\tilde{v}_l$, and time delay $\tilde{\Delta}$.

05

AC-to-DC Conversion
Diode Full Wave Rectifier

Lecture 05.01 Analyzing a full-wave bridge rectifier

In Lab Exercise 05, we will build and measure the response of a diode full-wave bridge rectifier circuit. This lecture is intended to introduce this circuit, used to convert AC signals to DC signals.

Figure 05.1 shows a circuit diagram for a full-wave bridge rectifier. Four diodes perform the rectification. A capacitor in parallel with the load effectively smooths this rectified signal.

There are four diodes, so there are $4^2 = 16$ possible diode states! We'll try to make good guesses and avoid invalid states in our analysis.

05.01.1 Analyzing the circuit

Let's begin by thinking about our diode model. Piecewise linear is our go-to. However, we can simplify it even further if we assume the load won't draw much current. The load in Lab Exercise 05 is 1 kΩ—so only small currents will flow through these diodes. We could go through the process of determining the current flow and choosing an operating point for diode resistances, but with such small current flow, we'd still be stuck setting all $R_d = 0$.

So we choose to model the circuit as shown in Figure 05.2. At this point we could perform a single analysis for the entirety of the circuit, but with the number of diodes, this gets messy. Therefore, we choose to reason our way through the potential states and analyze the few that survive.

Consider the case of $V_S > 0$. The only potential pathway for current is through D_1 and returning through D_2. It cannot return through D_4, since its voltage drop will be negative. Similarly, current cannot flow from ground, through D_3, to the positive side of the capacitor. That is, D_1 and D_3 cannot

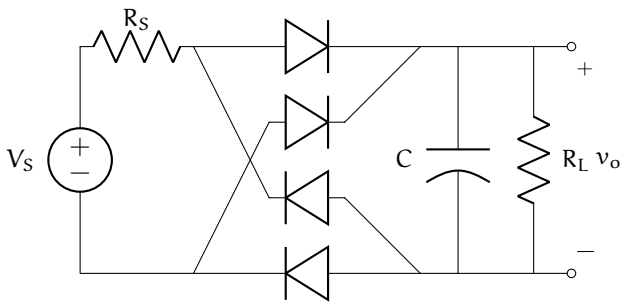


Figure 05.1: full wave bridge rectifier circuit, including a source output resistance R_S.

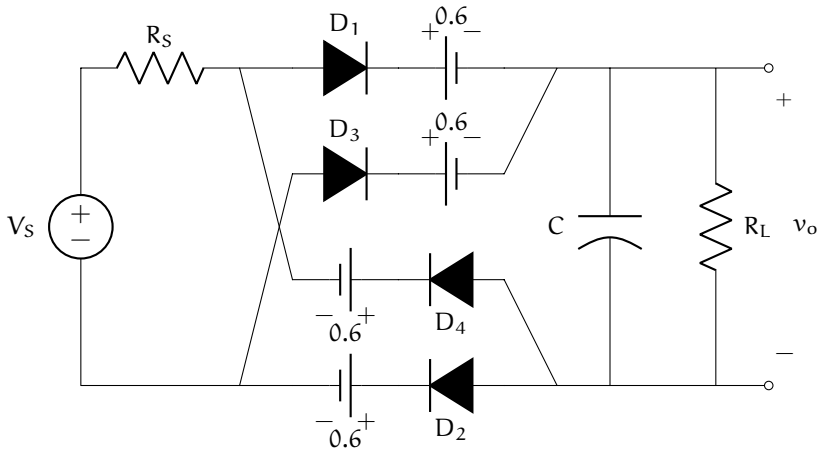


Figure 05.2: full wave bridge rectifier circuit, including a source output resistance R_S.

both be in forward-bias and neither can D_2 and D_4. A similar consideration of $V_S \leqslant 0$ shows that current can only flow to the load via D_3 and return through D_4.

Furthermore, we cannot have current flow to the load side of the circuit and not return, so we can eliminate any such cases. This leaves only three possible states.

primary charge state S_1 When D_1 is ON, current must return via D_2, so it must also be ON. D_3 and D_4 are off, corresponding to Figure 05.3. So, using bold face for ON:

$$S_1 = \{\ \mathbf{D_1},\quad \mathbf{D_2},\quad D_3,\quad D_4\ \}. \tag{05.1}$$

secondary charge state S_2 When D_3 is ON, current must return via D_4, so it must also be ON. D_1 and D_2 are off, corresponding to Figure 05.4. So, using bold face for ON:

$$S_2 = \{\ D_1,\quad D_2,\quad \mathbf{D_3},\quad \mathbf{D_4}\ \}. \tag{05.2}$$

discharge state S_3 When all four diodes are OFF, the charged capacitor discharges into R_L, corresponding to Figure 05.5. So our discharge state is:

$$S_3 = \{\ D_1,\quad D_2,\quad D_3,\quad D_4\ \}. \tag{05.3}$$

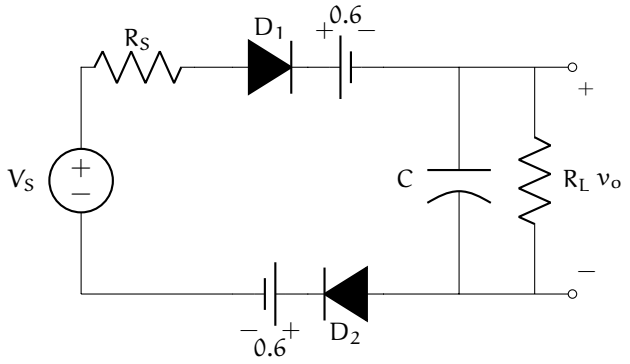


Figure 05.3: the primary charge state S_1 ($V_S > 1.2 + v_o$), in which only D_1 and D_2 are in forward-bias.

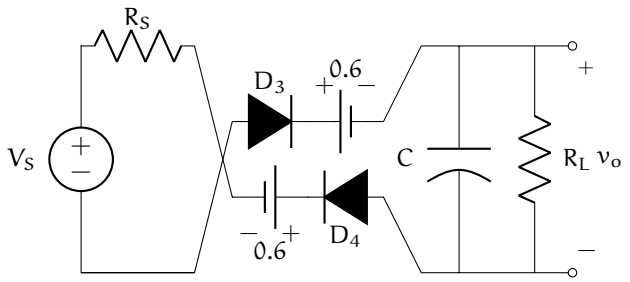


Figure 05.4: the secondary charge state S_2 ($V_S < -1.2 + v_o$), in which only D_3 and D_4 are in forward-bias.

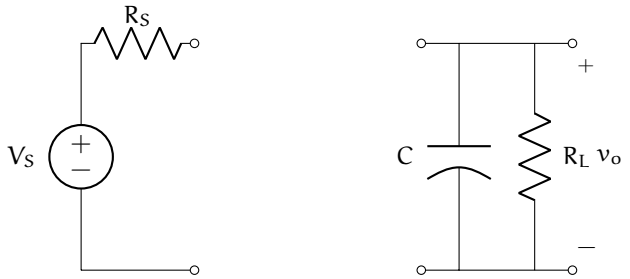


Figure 05.5: the discharge state S_3 ($-1.2 + v_o < V_S < 1.2 - v_o$), in which all diodes are in reverse-bias.

Let's consider each of these states as they arise from the initial time. As we do so, we can examine both the conditions that give rise to each diode state S_1–S_3 and how the circuit behaves during each state.

Let's assume the capacitor begins uncharged. This isn't much of a stretch, considering that it has a load resistor to which it can discharge.

Let's also assume V_S increases sinusoidally from 0 V. At first, there's not enough voltage drop across D_1 to turn it on, so we're stuck in the discharge state S_3 until V_S increases enough to overcome both the 0.6 V associated with D_1 and the 0.6 V associated with the return diode D_2—so when $V_S > 1.2$. At that point, we transition to S_1.

As V_S continues to rise, $v_o = v_C = v_{R_L}$ increases—that is, the capacitor charges. When V_S decreases, the capacitor voltage doesn't immediately disappear, since its discharge into R_L is typically relatively slow. As the decrease in V_S continues, it will switch back to state S_3, but this time before reaching 1.2 V because now *the capacitor is still charged* (although it is slowly discharging). Reconsidering the S_3, we see that $V_S > 1.2 + v_o$ for the diodes to be on. So we have now discovered that an alternative formulation for S_1:

$$S_1 = \{V_S > 1.2 + v_o\}. \tag{05.4}$$

Let's consider development of output voltage during S_1. In this state, we can analyze the circuit as follows. Here are the elemental equations.

$$\begin{array}{r|l} C & \dfrac{dv_C}{dt} = \dfrac{1}{C} i_C \\ R_S & v_{R_S} = i_{R_S} R_S \\ R_L & v_{R_L} = i_{R_L} R_L \end{array}$$

Similarly, KCL and KVL give the following equations.

$$\begin{array}{r|l} \text{KCL} & i_{R_S} = i_C + i_{R_L} \\ \text{KVL}_1 & V_S = v_{R_S} + v_C \\ \text{KVL}_2 & v_C = v_{R_L} \end{array}$$

Let's reduce this to a differential equation in v_C, starting with the capacitor

elemental equation:

$$\begin{aligned}
\frac{dv_C}{dt} &= \frac{1}{C}(i_{R_S} - i_{R_L}) && \text{(KCL)}\\
&= \frac{1}{C}\left(\frac{v_{R_S}}{R_S} - \frac{v_{R_L}}{R_L}\right) && \text{(resistors)}\\
&= \frac{1}{C}\left(\frac{V_S - v_C}{R_S} - \frac{v_C}{R_L}\right) \Rightarrow && \text{(KVL}_{1,2}\text{)}\\
\tau\frac{dv_C}{dt} + v_C &= \frac{1}{R_S C}V_S && (05.5)
\end{aligned}$$

where

$$\tau = \frac{R_S R_L}{R_S + R_L}C. \tag{05.6}$$

Word. The solution to this familiar differential equation for initial capacitor voltage $v_{C0} = v_C(t)|_{t=t_0}$ and input $V_S(t) = A\sin(\omega t)$ is our output $v_o(t)$.

The other charging state S_2 is similar and yields the alternative formulation of S_2:

$$S_2 = \{V_S < 1.2 - v_o\}. \tag{05.7}$$

The discharging state S_3 is the easiest circuit to analyze: it is simply the capacitor C with an initial voltage discharging through the load R_L, with no forcing.

Due to the nonlinearities, cycles through the states may be required before settling into a steady state. This relatively complicated analysis is sometimes bypassed by the method that follows in 05.01.2.

05.01.2 A quick-and-dirty model

A full time-domain analysis of the circuit is preferable to this quick-and-dirty model, but this method is a nice way to quickly estimate the results.

The DC component of a signal is its average value. The average value of the signal across the load can be predicted if we know the maximum value of the rectified signal (sans capacitor) A_r and the ripple voltage Δv_o, which is the steady state voltage from the high-to-low output voltage. The maximum value of the rectified signal should be

$$A_r = A - 2 \cdot 0.6,$$

where A is the input amplitude and the 0.6 V term comes from the voltage drop across a diode. Then the DC component of the signal should be approximately

$$v_{DC} = A_r - \frac{\Delta v_o}{2}.$$

If we assume the ripple voltage Δv_o is relatively small and the AC signal has a high-enough frequency that it changes faster than the capacitor, we can assume that the full wave rectifier's output can be approximated by the expression

$$\Delta v_o = \frac{i_{R_L}}{2fC} \tag{05.8}$$

where f is the input signal frequency.

How do we know the load current i_{R_L} if we don't do the entire circuit analysis? It turns out that most loads for AC-to-DC conversions are actually voltage regulators, which draw steady currents. In the case of a resistor load, we can use a conservative estimate: the max $v_{DC} = A_r$ (no ripple), so from Ohm's law the max $i_{R_L} = v_{DC}/R_L$. Using the max i_{R_L} in Equation 05.8 yields the max ripple.[1]

[1]This reasoning may appear circular: did we not assume the ripple was zero, then derive from that its maximum? In fact, we *dropped* the assumption that the ripple was zero once we derived the maximum current. This is an example of a *bootstrapping method*. We could actually iterate on this to get a better and better approximations, re-using better and better approximations of v_{DC} for better i_{R_L} for better Δv_o for better v_{DC}, etc.

Lab Exercise 05 AC-to-DC Conversion via a Diode Full Wave Rectifier

In this lab, we will design and build a full wave bridge rectifier for AC-to-DC conversion. We will capture the time response data with the NI myRIO device, export it to a data file, import that data file into MATLAB, plot the data, derive an analytic prediction, and compare the analytic prediction to the data.

The objectives of this lab exercise are for students:

1. to understand AC-to-DC conversion,
2. to deepen their understanding of diode circuits,
3. to better learn to acquire and log data with a microcontroller board (myRIO),
4. to learn to export the acquired data to MATLAB,
5. to model real circuits and compare the theory and experiment, and
6. to learn to better plot and export plots in MATLAB.

Lab 05.1 Materials

The following materials are required for each lab station:

☐ 1. a PC with LabVIEW installed,[2]
☐ 2. a myRIO configured with LabVIEW,[3],
☐ 3. an oscilloscope,
☐ 4. a multimeter,
☐ 5. a breadboard,
☐ 6. jumper wires,
☐ 7. four 1N914 diodes,
☐ 8. a 1 kΩ resistor, and
☐ 9. a 100 μF capacitor.

Lab 05.2 Build the circuit

The full wave bridge rectifier circuit we are building can be represented by the circuit diagram of Figure 05.6.

[2]See Resource 2 for more details on the LabVIEW software configuration.
[3]See Resource 3 for more details on the myRIO software configuration.

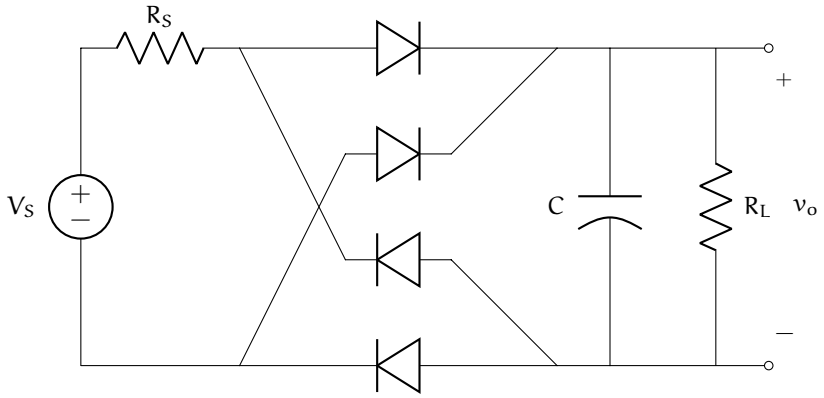


Figure 05.6: full wave bridge rectifier circuit, including the function generator output resistance $R_S = 50\ \Omega$.

☐ 1. Use a function generator as the voltage source. Choose a capacitor of nominal capacitance $C = 100\ \mu F$ and a resistor of nominal resistance $R_L = 1\ k\Omega$. Select four 1N914 diodes.

☐ 2. Measure and record the actual resistance R and capacitance C with a multimeter.

	R_L (Ω)	C (μF)
nominal	1000	100
measured		

☐ 3. Measure each diode voltage drop using the multimeter. The stripe on the diode corresponds to the cathode (the direction toward which current typically flows). The voltage drops should be around 0.6 V.

	$\tilde{v}_{D_1}$ (V)	$\tilde{v}_{D_2}$ (V)	$\tilde{v}_{D_3}$ (V)	$\tilde{v}_{D_4}$ (V)
measured				

☐ 4. Build the circuit on a breadboard. It should look something like Figure 05.7.

Be cautious with the capacitors—they're polarized! Be sure to connect the − terminal to ground.

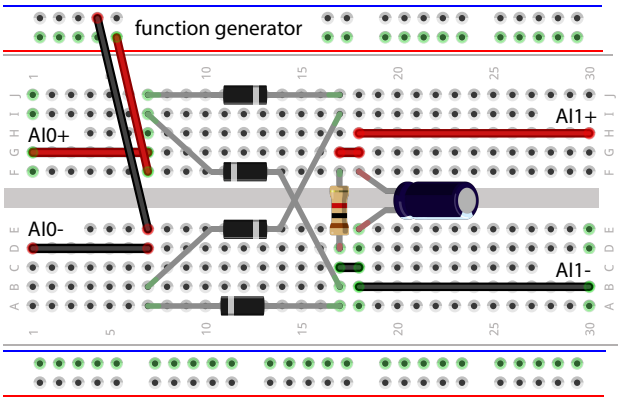


Figure 05.7: an example of how the breadboard circuit might look.

Box 05.1 prepare the myRIO for data collection

Before proceeding, be sure to prepare the myRIO for data collection with the steps of Resource 5.

Lab 05.3 Create a LabVIEW project and Main VI

The Labview VI needed for this lab exercise is very similar to that of Lab Exercise 03. Either copy your project directory from that exercise or create a new LabVIEW myRIO project. Edit `Main.vi` such that it has the following functionality:

- ☐ 1. measures the function generator voltage using `AI0+`, continuously;
- ☐ 2. measures the voltage across the capacitor using `AI1+`, continuously;
- ☐ 3. plots the circuit input (scope output) and circuit output (capacitor) voltages on the same chart, continuously;

timed-loop

- ☐ 4. uses a *timed-loop* that executes once every 10 ms;
- ☐ 5. after (and only after) the user toggles the front-panel button record data pressed, stores
  - ☐ a. time values (with zero beginning when the button is pressed),
  - ☐ b. measured circuit input voltage $\tilde{v}_s$, and
  - ☐ c. measured circuit output voltage $\tilde{v}_o$ in arrays; and
- ☐ 6. after the timed loop is exited via a Stop & Save button, write the three arrays to a LabVIEW `.lvm` datafile on the myRIO via the custom VIs

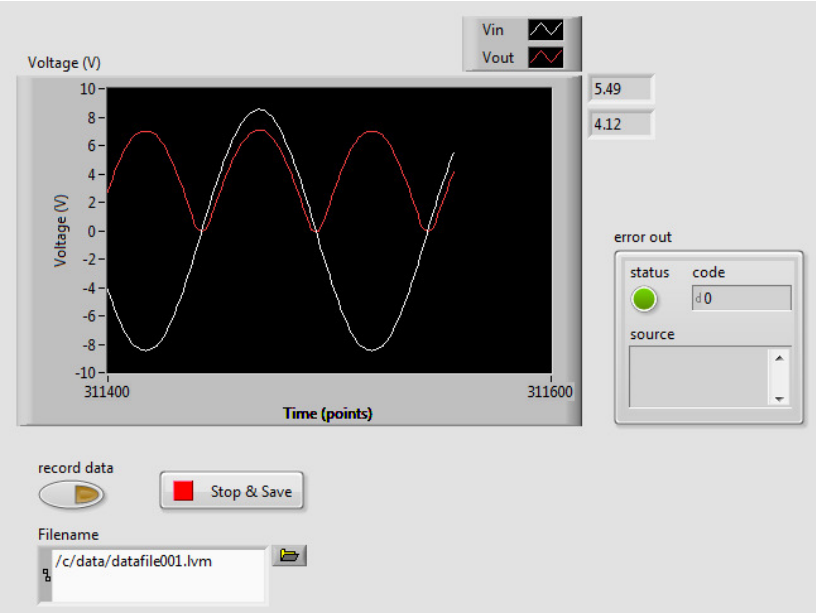


Figure 05.8: how the front panel of your VI might look when the capacitor is disconnected.

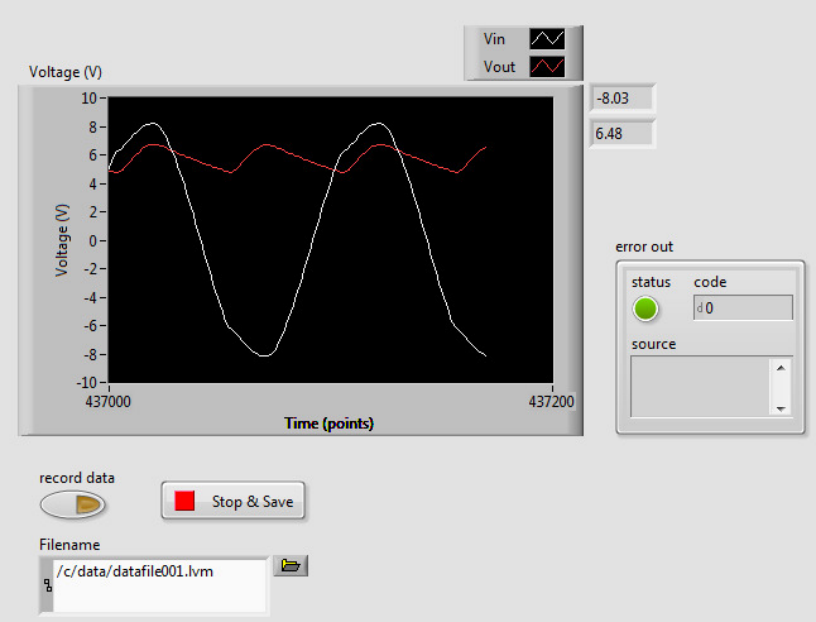


Figure 05.9: how the front panel of your VI might look when the capictor is connected.

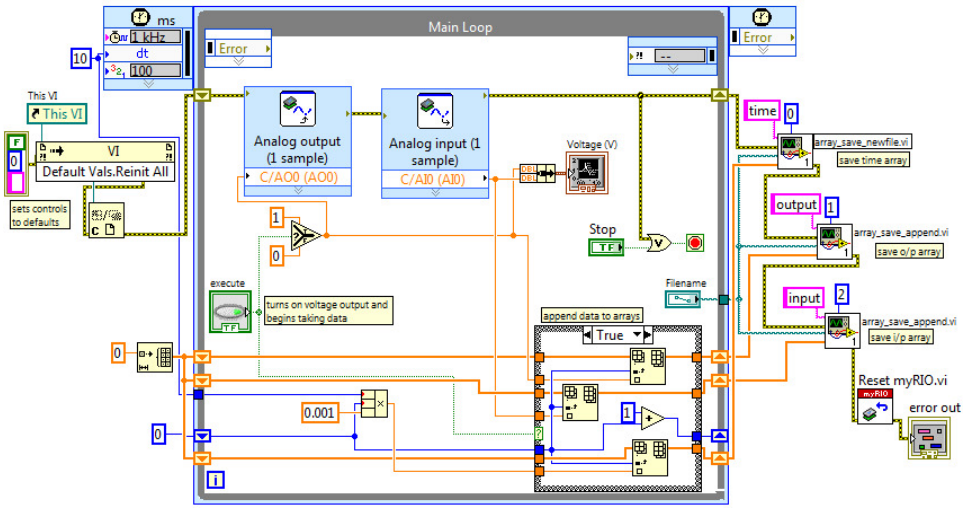


Figure 05.10: how the block diagram of your VI might look. Note that, unlike your VI, this example has an output voltage and does not include a measurement of the source $\widetilde{V}_s$. One should probably use a loop time of `1` ms instead of the `10` shown.

I've written for you `array_save_newfile.vi` and `array_save_append.vi`, which are zipped here:

`ricopic.one/resources/array_savers.zip`.

(unzip and move them to the same directory as your project);

The front panels of Figures 05.8 and 05.9 and corresponding block diagram of Figure 05.10 show a way of realizing a very similar VI. The difference is that the example includes an output voltage and has no measurement of $\widetilde{V}_s$.

Lab 05.4 Measure the signals

Use your VI to measure the source voltage $V_S(t)$ and the output voltage $v\_o(t)$ across the load resistor.

included Set your function generator to 8 V.[4] For two cases, the capacitor *included*
excluded and *excluded* from the circuit, measure and record (save to a data file) each

[4] If you're using a BK Precision function generator, use your VI to hand-tune the function generator voltage to 8 V.

voltage (in and out) over at least three periods of input at 10 Hz.

Repeat for the function generator frequency at the following values:[5]

- ☐ 1. 15 Hz,
- ☐ 2. 20 Hz, and
- ☐ 3. 25 Hz.

Be sure to note which data file is associated with which frequency. Also, be sure to change the filename before each trial is run.

Without the capacitor, the front panel should display something like what is shown in Figure 05.8.

With the capacitor, the front panel should display something like what is shown in Figure 05.9.

Copy your data files to a convenient directory on the computer (e.g. `Documents/me316/lab05`).

Lab 05.5 Report requirements

Write a detailed report of your experimental results, as outlined in the report template. Pay special attention to the following.

- ☐ 1. A *thorough* description of the theoretical analysis of the circuit. As part of this analysis, predict ripple and average DC value for each circuit with the capacitor attached. Also predict the current supplied to the load resistor R_L. Use these and the model of 05.01.2 to give a predicted DC voltage v_{DC} and ripple Δv_o.
- ☐ 2. Four figures that compare each of the capacitor-attached captured data sets (input and output) to its corresponding analytic prediction. It is acceptable to predict only the ripple range and average value of the output, but a full time-analysis is even more fun! In MATLAB, be sure to use `xlim` to scale the x-axis such that around 3–5 periods are visible on each plot (or trim the data). Include somewhere on the plot the approximate measured ripple and DC value. (You can use the plot cursor tool to help with the estimate. "Measure" peak-to-peak and divide by two. Alternatively, use the MATLAB function `cumtrapz` to numerically integrate the voltage and divide it by the time duration, which should be an integer multiple of the signal's period, to find its average.)

[5] If you're using a BK Precision function generator, use your VI or the oscilloscope to measure the actual frequency, which you should record.

- □ 3. Four figures that display the capacitor-detached input and output data sets. Alternatively (preferably), plot these on the corresponding capacitor-attached data set plots. Carefully label and caption these.
- □ 4. The other parameters measured in the lab (e.g. R_L, C).
- □ 5. Be sure to comment on how well your predictions matched the theory.

06

555 Timer Circuit and Soldering

Lecture 06.01 Soldering

soldering
work pieces
solder
solder joint
PCBs

Soldering is the process of joining two pieces of metal (*work pieces*) by introducing molten metal called *solder*, which bonds to both metals, creating a *solder joint*, which joins the work pieces both mechanically and electrically.

In large-scale production of *printed circuit boards* (PCBs), soldering is automated. However, when prototyping for applications that require strong mechanical connections or have space or weight constraints, the mechatronicist must solder the prototype circuit.

Before you begin soldering, please watch Collin Cunningham's video introduction *Soldering*:

https://youtu.be/QKbJxytERvg.

Here are some additional tips.

1. Integrated circuits such as the 555 timer can be easily damaged due to excess heat, so it is important not to overheat it, keeping the iron on the terminal for as little time as possible (< 2 sec). Practice by soldering other components first.
2. Use flux paste, which aids in breaking the surface tension of the melted solder.

tinning

3. *Tinning* the tip of the soldering iron will improve its performance.
4. An oxide layer will periodically form on the tip of your iron, to remove it simply wipe the tip on a wet sponge.
5. Temperature control allows you to find the ideal temperature so that it will solder without overheating the components.
6. The soldering iron is extremely hot so practice caution when using it. Always place it into the holder when it's not in use.

Lab Exercise 06 555 Timer Circuit and Soldering[1]

The objectives of this lab exercise are for students:

- ☐ 1. to understand the 555 timer,
- ☐ 2. to be introduced to digital circuits,
- ☐ 3. to learn to solder, and
- ☐ 4. to analyze and build relatively complex circuits.

Lab 06.1 Materials

The following materials are required for each lab station:

- ☐ 1. a PC with LabVIEW installed,
- ☐ 2. a myRIO configured with LabVIEW,
- ☐ 3. a multimeter,
- ☐ 4. a breadboard,
- ☐ 5. a prototyping board,
- ☐ 6. wire and jumper wires,
- ☐ 7. a 9 V battery,
- ☐ 8. a 9 V battery connector,
- ☐ 9. a 555 timer integrated circuit,
- ☐ 10. two 1 kΩ resistors,
- ☐ 11. a 470 kΩ resistor,
- ☐ 12. an LED, and
- ☐ 13. a 1, a 10, and a 100 μF capacitor.

Lab 06.2 555 timers

555 timer

The *555 timer* is a ubiquitous integrated circuit that can be used to output periodic signals loaded up to around 200 mA. The pinout diagram is shown in Figure 06.1. Note the "dot," which can be used to orient the circuit.

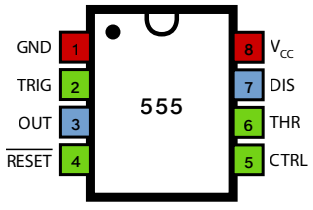


Figure 06.1: 555 timer pinout (contributors, 2018b).

A functional block diagram is shown in Figure 06.2. The `VCC` pin `8` has its voltage divided by three 5 kΩ resistors in green before being grounded by `GND` pin `1`. Pin `2` (`TRIG`) and pin `6` (`THRES`) are compared by *comparators* to the one-third and two-thirds of `VCC`. Comparators

comparator

[1] Portions of this lab exercise were prepared by Jordan Parker, SMU class of 2019.

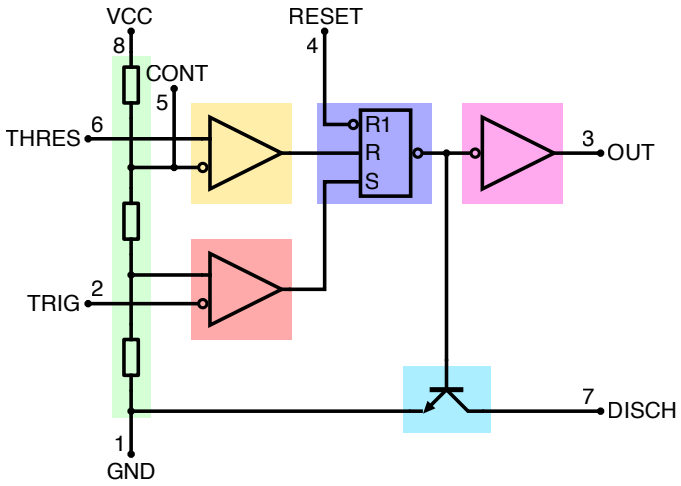


Figure 06.2: a 555 timer functional block diagram (contributors, 2018a).

SR flip-flop

memory

are digital electronic devices that output boolean `TRUE` when the upper terminal has voltage greater than that of the lower terminal—and output `FALSE`, otherwise. Their outputs are connected to an *SR flip-flop* (purple), which gets set to `TRUE` by terminal `S` and reset to FALSE by `R`. This flip-flop has a *memory*. Once `S` sets it, even if `S` goes `FALSE`, it will remain `TRUE`. Only if `R` goes `TRUE` will the flip-flop return to `FALSE`. The `RESET` pin `4` can reset the flip-flop state at any time.

If the flip-flop is set to `TRUE`, two things happen. First, the output is inverted (the symbol ∘ means inversion), and it shuts off the gate for a transistor (blue) to which we will return in a moment. The flip-flop output is inverted again and switches on an output stage (pink), supplying power to the `OUT` pin `3`.

Once the `THRES` pin resets the flip-flop, the gate transistor is turned on, allowing current to flow from pin `7` (`DISCH`) to pin `1` (`GND`).

Lab 06.3 The LED 555 timer circuit

We will be building the circuit diagram of Figure 06.3. This circuit charges, then discharges the capacitor C_1. The capacitor's voltage is applied to both pins `2` and `6`, which control the flip-flop, but which are compared to different voltages. So the capacitor first charges up enough to trigger the output (`3`, which flows through the LED (through R_3, which limits current). Once the capacitor charges up to initiate the reset, the LED turns off and the

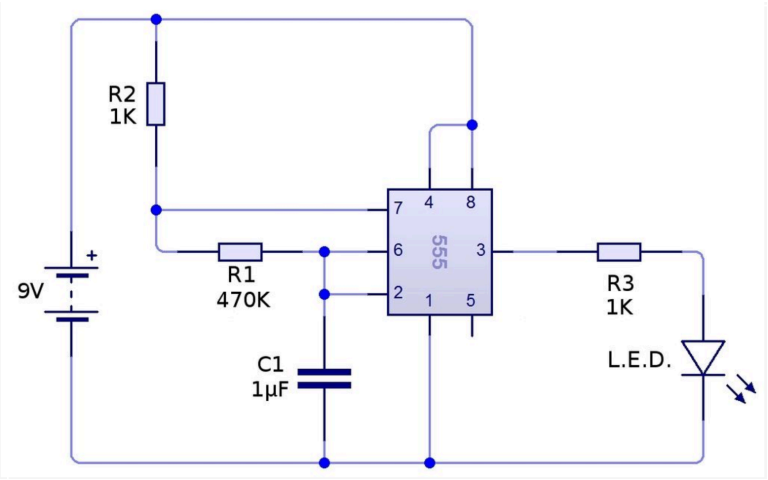


Figure 06.3: 555 Timer circuit diagram (Trilesto, 2018).

discharge transistor turns on. This connects pins 7 and 1, which effectively discharges C_1 to ground through R_1 with time constant $\tau = R_1 C_1$. Once it has discharged to the lower threshold voltage, the flip-flop turns back on and recharges the capacitor.

Lab 06.4 Building the circuit

Use the following procedure to build the LED 555 timer circuit.

- ☐ 1. Gather the components listed above.
- ☐ 2. Measure and record the parameter value for each resistor, each capacitor, the battery, and the LED in Table 06.1.
- ☐ 3. Lay out the circuit in accordance with the circuit diagram of Figure 06.3 on a prototyping board. Use the 1 µF capacitor, but connect it with an auxiliary breadboard.
- ☐ 4. Leaving the battery disconnected from its connector, solder each connection, connecting components that aren't directly next to each other with wires.

Lab 06.5 Measuring the waveform

Use the myRIO with a LabVIEW VI similar to those you've created previously to record at least a few periods of v_{C_1} and v_{R_3}.

Table 06.1: nominal and measured parameter values.

	C_1 (μF)	C_2 (μF)	C_3 (μF)	R_1 (kΩ)	R_2 (kΩ)	R_3 (kΩ)	V_S (V)	ν_D (V)
nominal	1	10	100	470	1	1	9	0.6
measured								

Desolder the capacitor and repeat the previous step with the 10 μF capacitor and the 100 μF capacitor. Be sure to capture several periods of data for each circuit.

Lab 06.6 Report requirements

Write a report on your laboratory activities using the template given. Be sure to pay special attention to the following elements:

- □ 1. In your circuit analysis, predict the entirety of the capacitor voltage waveform for each circuit (different circuits here meaning different capacitors).
- □ 2. Plot your prediction and the measured waveforms on the same graph (one graph for each circuit, three in total). One period is sufficient, but multiple is also fine. A bit better would be to average multiple periods of measured data. However, some difficulty arises when trying to match the period and this is not required.

It may be helpful to take photos during the laboratory procedure. These can be included in your report.

07

Thermal Response

Lab Exercise 07 Thermal Response

In this lab, we will estimate the lumped-parameter thermal resistance and capacitance of a 3D-printed block of PLA by measuring its temperature time-response to internal heating via a resistor. Temperature measurements will be made with thermistors connected to Wheatstone bridge circuits, the output voltages of which which will be measured with a myRIO device. A calibration curve for each thermistor will be generated.

We will be inferring the lumped-parameter thermal resistance R and thermal capacitance C of a block of PLA from its time response. Each block has a 56 Ω resistor embedded within acting as a heater and three thermistors embedded, but we will use only the one closest to the resistor to measure the temperature response of the block.

Lab 07.1 Thermistor calibration

We need to "calibrate" the Vishay NTCLE100E3103JB0 thermistors by measuring their resistances at a measured room temperature. We say "'calibrate'" because we are not fully calibrating the thermistors ourselves, but partially relying on the manufacturer's datasheet. We will develop a calibration curve and a lookup table for each thermistor.

Thermistors are highly temperature-dependent resistors. The calibration curves of many thermistors, including those we will use, can be described by the Steinhart-Hart equation. The following Vishay datasheet gives the equation's parameters based on an R_{25} resistance, which means the measured resistance of the thermistor at a temperature of 25 C:

ricopic.one/resources/thermistor_datasheet.pdf.

However, we don't have a climate-controlled environment at 25 C, so this isn't helpful. Conveniently, the manufacturer provides the following spreadsheet that can be used to calibrate our thermistors at non-25 C-temperatures:

ricopic.one/resources/cc6400c1.xls.

Our thermistors have nominal resistance at 25 C of $R_{25} = 10$ kΩ. As the temperature *increases* from there, the resistance *decreases* (this means we have negative temperature coefficient or NTC thermistors).

The spreadsheet gives a temperature-resistance relation, which is the calibration curve. Additionally, it gives an estimated temperature deviation

ΔT for each point on the curve. It accounts for the tolerance of our temperature measurement.

Lab 07.2 Calibration measurements

The following procedure will yield the required measurements.

- ☐ 1. Locate within the laboratory an accurate temperature measurement device. Be sure to note the model so that you can look up the tolerance/accuracy in its manual.
- ☐ 2. Connect the PLA block to a breadboard across the "spine" that runs down the middle. Be sure not to short the thermistors by connecting them to the same circuit node.
- ☐ 3. Use a multimeter to measure and record in the table below the resistance of the resistor R_L.
- ☐ 4. Use a multimeter to measure and record in the table below the resistance of the thermistor R_{x0} adjacent to the resistor R_L. Note that, when you handled the block to connect the leads to the breadboard, you may have heated it enough to register on the thermistor. If so, the resistance of the thermistor will be climbing as it cools. Once the measurement has reached steady-state, record the resistance in the table below.
- ☐ 5. Use the temperature measurement device to measure and record the temperature T_∞ of the block (or nearby ambient air).

	R_L (Ω)	R_{x0} (kΩ)	T_∞ (C)
nominal	56	10	25
measured			

Lab 07.3 Generate the thermistor calibration curve

The following procedure will yield the required calibration curve.

- ☐ 1. Find the manual for your temperature measurement device. Determine and record the accuracy/tolerance of your temperature measurement.
- ☐ 2. Using the Vishay spreadsheet, enter (in the `RTCOEF` tab) `0.1` for the temperature step, `-10` for the initial temperature, your measured

thermistor temperature T_∞ for the reference temperature, your measured thermistor resistance R_{x0} for the resistance at the reference temperature, the accuracy/tolerance ΔT of your temperature measurement, `3977` for your `B25/85`.

- ☐ 3. Save a copy of the spreadsheet as a comma-separated-values (CSV) file.
- ☐ 4. In MATLAB use the `Import Data` dialog to import the data from the CSV file and select *only the numerical region of the spreadsheet*. Also, give the `Temperature`, `Resistance`, and `DT` columns of each special names—these are the ones we actually care about. Import the data.
- ☐ 5. Use the `save` command to save the `Temperature`, `Resistance`, and `DT` (ΔT) variables for each thermistor to a `mat` file. These can then be loaded much easier into a MATLAB script.
- ☐ 6. Write a MATLAB script that loads the `mat` file and plots the calibration curve (T vs R). On the same plot, use the `errorbar` command to plot ΔT as error bars. Notice how bad this looks. We will do two things to make these plots better.
- ☐ 7. Alter the previous plots so that all the data is still plotted, but only 20 error bars appear. (Hint: when using `errorbar`, plot only some of the points.)
- ☐ 8. Alter the previous plot with `semilogx` instead of `plot`, which is tantamount to taking `log10` of the resistance data (but MATLAB does some nice labeling for you). This should make the plot look quite sharp. Make sure you do a `hold on` before your `errorbar` plot, which needs to appear at the end.

Lab 07.4 Build the circuit

The Wheatstone bridge circuit we are building can be represented by the diagram of Figure 07.1.

- ☐ 1. Select three resistors with nominal resistance $R_1 = R_2 = R_3 = 15\ \text{k}\Omega$.
- ☐ 2. Measure and record the actual resistances $\tilde{R}_1$, $\tilde{R}_2$, and $\tilde{R}_3$ with a multimeter.
- ☐ 3. Build a bridge circuit on a breadboard. Connect the source voltage terminals to a 9 V battery (the DC power supplies are too noisy).
- ☐ 4. Set up, but do not yet connect, a DC power supply. Set it to 5 V and connect the − lead to one side of the block resistor.

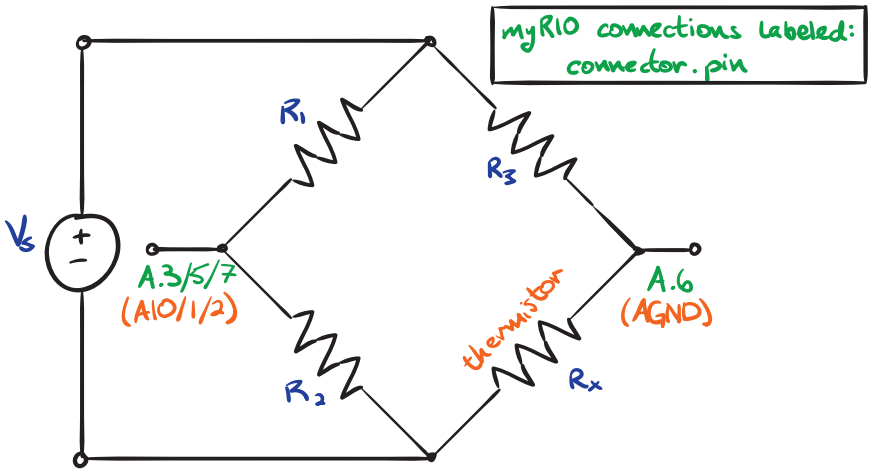


Figure 07.1: the Wheatstone bridge circuit and pinout diagram.

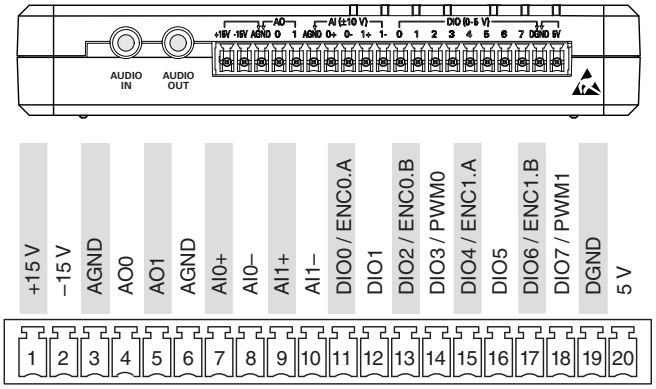


Figure 07.2: myRIO Connector `C` (from Instruments (2013)).

Lab 07.5 Connect the myRIO to the circuit

We are using the myRIO to measure the bridge's output voltage.

We will be using the `MSP` connector named `C` for the bridge output voltage measurement. It is shown in Figure 07.2.

Connect the myRIO to the breadboard as described in the circuit diagram of Figure 07.1. You can use any analog input. The `C` connector differential inputs are slightly better for this application, but any will do. Use the pin numbers shown in the connector diagrams above (or equivalent).

Lab 07.6 LabVIEW program to control the myRIO

Write a LabVIEW program to control the myRIO and save captured data. You will want to be able to measure the bridge output voltage time response with an analog input, plot, and save the data (and associated time array). Previous Lab Exercises' programs may be easily adopted.

Lab 07.7 Measure the temperature response

- ☐ 1. Let the thermal system come to equilibrium with its (room temperature) environment.
- ☐ 2. Begin to capture data and connect the heater power supply. The bridge output voltage (which you should be plotting in your program) should increase. Be careful not to use too high of a sample rate in order to avoid excessive data (1/2 Hz is probably sufficient for this slow system).
- ☐ 3. Once the response has reached something close to steady-state, disconnect the power supply.
- ☐ 4. Observe the slow decay of the temperature to equilibrium.
- ☐ 5. Once the temperature has settled back to room temperature (the bridge voltage will return to its original value), terminate the data capture (and save the data).
- ☐ 6. Copy the `.lvm` data file from the myRIO to a convenient PC location.

Lab 07.8 Report requirements

Write a detailed report of your experimental results, as outlined in the report template. Pay special attention to the following.

- ☐ 1. Include your calibration curve plots (with error bars) that allow you to convert from thermistor resistance R_x to temperature $T_C(t)$.
- ☐ 2. Include a theoretical derivation of the bridge output voltage as a function of the thermistor resistance R_x.
- ☐ 3. Compute and include a plot of $T_C(t)$, your temperature measurement.
- ☐ 4. Include all measured values of resistance.
- ☐ 5. Review the following for guidance before continuing:

  `ricopic.one/resources/thermal_lab_handout.pdf`.

- ☐ 6. Plot the Region II linear plot that has slope $-1/\tau$.
- ☐ 7. Plot the Region I constant plot that has constant value R.
- ☐ 8. Give final estimates for τ, R, and C.

08

Brushed DC Motors

Lab Exercise 08 Brushed DC Motors

Lab 08.1 Introduction

In this lab, we will use a brushed DC motor to drive a rotational mechanical system with a myRIO microcontroller; measure the motor's voltage, current, and angular velocity; and characterize the motor. The electromechanical systems are shown in Figure 08.1.

The objectives of this lab exercise are for students:

1. to characterize the performance of a motor,
2. to apply motor driving,
3. to apply PWM,
4. to apply H-bridge circuits,
5. to apply quadrature encoders, and
6. to estimate electromechanical system parameters such as the motor constant.

Lab 08.2 Materials

The following materials are required for each lab station:

- ☐ 1. a PC with LabVIEW installed;[1]
- ☐ 2. a myRIO configured with LabVIEW;[2];
- ☐ 3. one ECL apparatus (see Figure 08.1) with encoder and most motor driver connections pre-connected as in Figure 08.7, *except C-channel connections*, which are left for the students;
- ☐ 4. a multimeter;
- ☐ 5. a breadboard;
- ☐ 6. jumper wires; and
- ☐ 7. two 330 kΩ resistors R_1 and R_2 (or similar).

Lab 08.3 Motor driving

A DC motor requires DC electrical power provided by a circuit called the "driving" circuit. For industrial motors at least, these circuits must provide significant power, and for this reason a separate (from the control circuit) power supply is often used. There is a quick-and-dirty way to drive a DC motor at variable speed: since its angular velocity is reliably

[1] See Resource 2 for more details on the LabVIEW software configuration.
[2] See Resource 3 for more details on the myRIO software configuration.

proportional to its voltage, place a potentiometer in series with the power supply and motor. However, this has disadvantages that include the power being limited and dissipated at high potentiometer resistance (low speed). For most applications, we will need either a current (or power) amplifier or a microcontroller and an integrated circuit to produce a pulse-width modulation driving signal. The latter is usually more cost-effective, but the former is advantageous in certain applications.

Lab 08.4 Pulse-width modulation

Pulse-width modulation (PWM) is a technique used to deliver an effectively variable signal to a load (in our case a motor) without a truly variable power source. A pulse of full source amplitude is repeated at a high frequency (e.g. 20 kHz), delivering a signal that is effectively averaged by the load dynamics such that its effects on the load are nearly continuous. The fraction of the period T that the signal is high (on) is called the *duty cycle* δ. Figure 08.2 shows a PWM signal $v(t)$ and its average $\bar{v}(t)$ with a few parameter definitions.

The mean of any periodic signal with period T can be computed with the integral

$$\bar{v}(t) = \frac{1}{T}\int_0^T v(t),$$

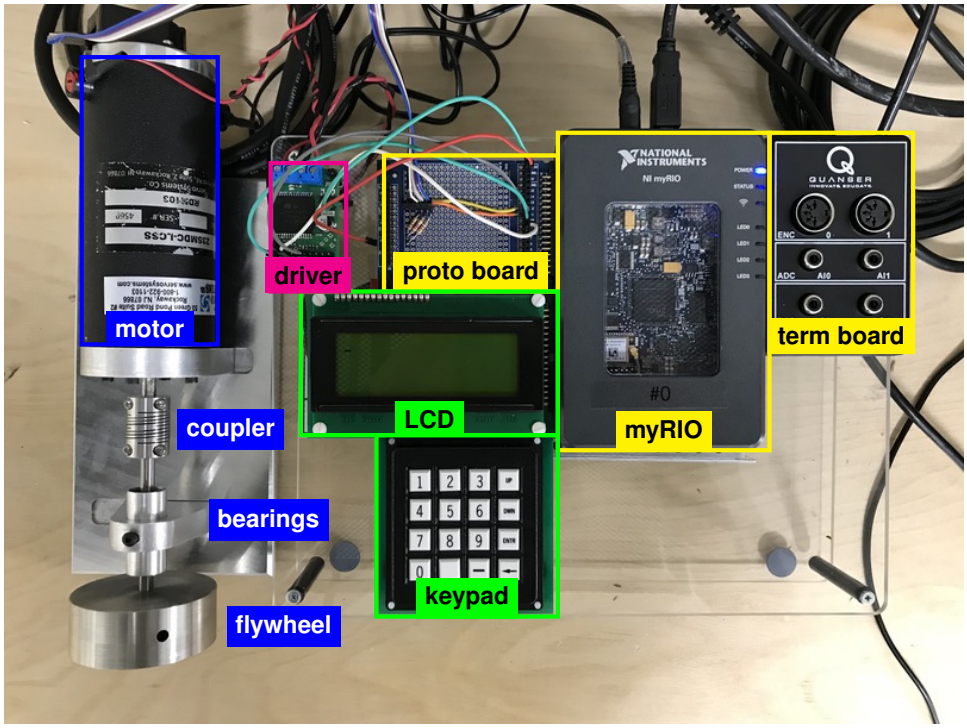
which is easily evaluated for a PWM signal:

$$\bar{v}(t) = \frac{Aw}{T} = A\delta.$$

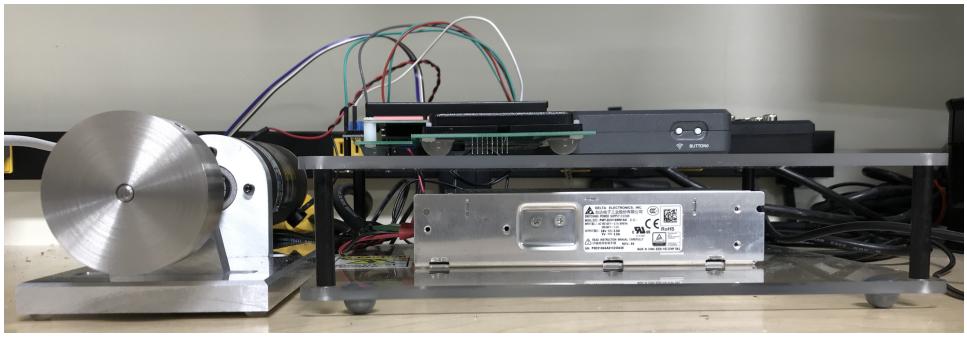
This result shows that if a PWM signal is delivered to a load, such as a DC motor, that is relatively unaffected by high-frequency signals, the effective signal will be simply the product of the source amplitude A and the duty cycle δ. The duty cycle can have values from 0 to 1, so the effective DC signal produced varies linearly with δ from 0 to A.

Lab 08.5 Motor driving with PWM

A microcontroller such as the myRIO or Arduino can easily produce a PWM signal; however, this signal is typically *low-power* and cannot drive even small DC motors. Therefore it is common to include a special kind of integrated circuit (IC) that uses the microcontroller's low-power PWM signal to gate a high-power DC source signal for delivery to the motor.



(a) top view of most of the ECL apparatus.



(b) front view of most of the ECL apparatus.

Figure 08.1: the electromechanical system from the Embedded Computing Laboratory (ECL) used in this lab exercise.

These are called *motor drivers*. They deliver power from a high-power source in accordance with a PWM signal, and they often include many additional features such as

motor drivers

1. compact forms;
2. forward- and reverse-driving (see Lab 08.6)
3. protection against reverse voltage, overcurrent, and overheating; and
4. output pins that monitor delivered current and voltage.

Lab 08.6 H-bridge circuits

We want to drive DC motors at different effective voltages *and* different directions. An H-bridge circuit allows us to reverse the direction of the PWM signal delivered to the motor. Figure 08.3 is a diagram of the H-bridge circuit.

The switches S1-S4 are typically instantiated with MOSFET transistors. As shown in the figure below, during the high duration of the PWM pulse, either S1 and S4 (Figure 08.3(b)) or S2 and S3 (Figure 08.3(c)) are closed and the others are open.

Recall that the electronics side of a DC motor can be modeled as a resistor and inductor in series with an electro-mechanical transformer. The inductance of the windings make it an "inductive" load, which presents the following challenge. We can't rapidly change the current flow through an inductor without a huge spike in voltage, and the switches do just that, leading to switch damage. Therefore, during the low or "off" duration of the PWM signal, S1-S4 cannot all be simply opened. There are actually a few options for switch positions that allow the current to continue to flow without inductive "kickback."

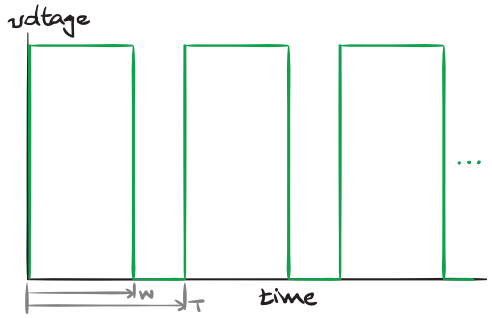
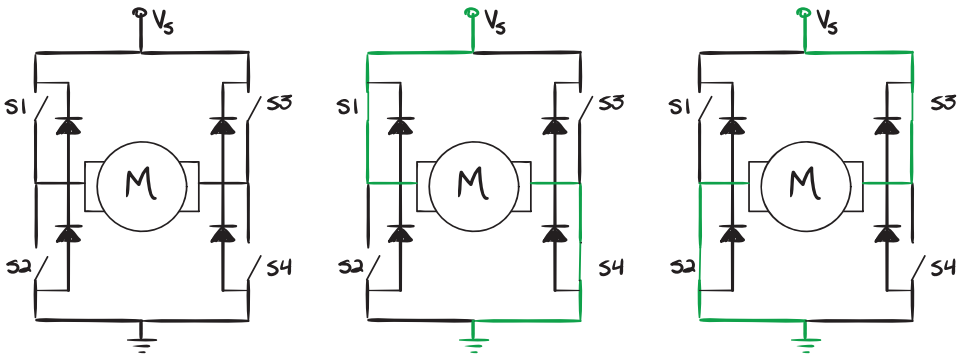


Figure 08.2: a pulse-width modulation (PWM) signal.



(a) off. (b) on one direction. (c) on the other direction.

Figure 08.3: H-bridge operation.

What's up with the diodes? Technically, they could be used to deal with the kickback. But since the diodes dissipate power, the proper switching is the primary kickback mitigation technique. However, the diodes ease the transition between switch flips, which are never quite simultaneous.

Lab 08.7 The ECL instantiation

We use a connectorized printed circuit board (PCB, e.g. a PC motherboard)—the Pololu motor driver carrier:

`pololu.com/product/1451`
`ricopic.one/resources/pololu_VNH5019.pdf` (manual)

This includes an STMicroelectronics VNH5019 H-bridge motor driver integrated circuit:

`ricopic.one/resources/vnh5019.pdf`

Lab 08.8 Voltage dividing

The myRIO's C-connector has analog differential inputs that accept voltages in the interval $[-10, +10]$ V. However, the DC power supply used can deliver more than (but we will not exceed) 20 V, which is the

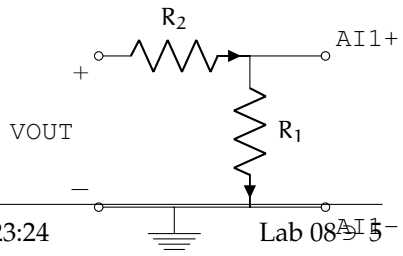


Figure 08.4: voltage divider for measurement of `VOUT`.

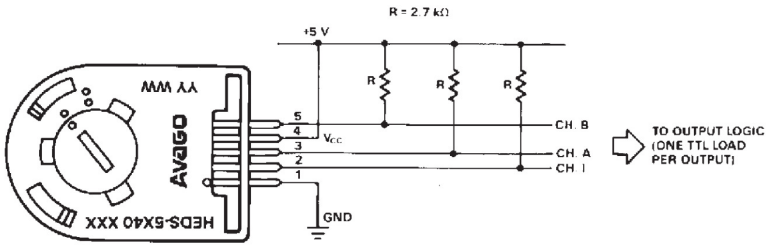


Figure 08.5: the encoder used (source: HEDM-55xx/560x & HEDS-55xx/56xx Data Sheet).

maximum we expect to measure at the motor via pin `VOUT`. This is a common situation, that the analog inputs will not contain the voltage range that we would like to measure. The easiest method to mitigate this challenge is to use a simple voltage divider circuit with known resistances R_1 and R_2. We do not want to saturate the myRIO input, nor do we want the input to be so small that we lose resolution, so we choose our resistance ratio accordingly—in this case, two nominally 330 kΩ resistors will work just fine, dividing the signal in two.

Lab 08.9 Measuring motor speed

Motor position and angular velocity are best measured by rotational quadrature encoders. Rotational encoders are made from a wheel with alternating dark and light stripes. The encoder we have affixed to the rear shaft—the HP HEDS-5640-A06 with manual

`ricopic.one/resources/encoder_manual.pdf`

—has black stripes on clear plastic. A light source either reflects differently off the stripes or, as in our case, passes the light through the clear plastic wheel into a photodiode or is blocked by the black stripes. Each time a stripe passes by, the photodiode detects a "blink," which is passed on to the myRIO via digital channels of the myRIO configured for detecting encoder outputs.

The encoder pinout is shown in Figure 08.5, from the manual.

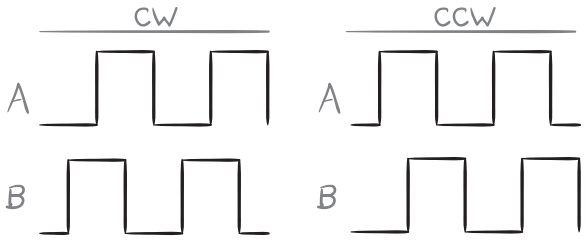


Figure 08.6: quadrature encoding with channels `A` and `B`.

Lab 08.10 Quadrature encoders

The only issue remaining is that a given "blink" doesn't give one important piece of information: which direction the encoder is rotating. However, a clever technique called *quadrature encoding* can be used to determine direction. The idea is that if we double the number of stripes, but offset the second set by half of a stripe width, then measure both "channels" `A` and `B`, then the direction can be determined by which channel "leads" the other. For instance, in Figure 08.6, the encoder output is high when light is detected and low when it is blocked by a stripe. Channel `A` leads `B` when the encoder is rotating clockwise (CW) and `B` leads `A` when it is rotating counter-clockwise (CCW).

Lab 08.11 Step response measurements

In this laboratory, a step voltage input will be supplied to the motor until the flywheel has reached steady-state speed. Then, the step voltage will be removed and the flywheel will slow to a stop. With a single continuous measurement in time, this step input and free response will be repeated several times, such that the step responses can be averaged in post-processing.

These step responses characterize both transient *and* steady-state for the system. The Mechatronics lecture course shows how such data can be used to estimate, as we will here, both the dominant time constant τ_1 and the motor constant (transformer ratio) TF (with estimates $\tilde{\tau}_1$ and $\hat{\mathsf{TF}}$).

Lab 08.12 Measurement setup

Check to make sure the PC, myRIO, proto board, motor driver, and motor are properly connected before proceeding with the final measurement setup steps, below.

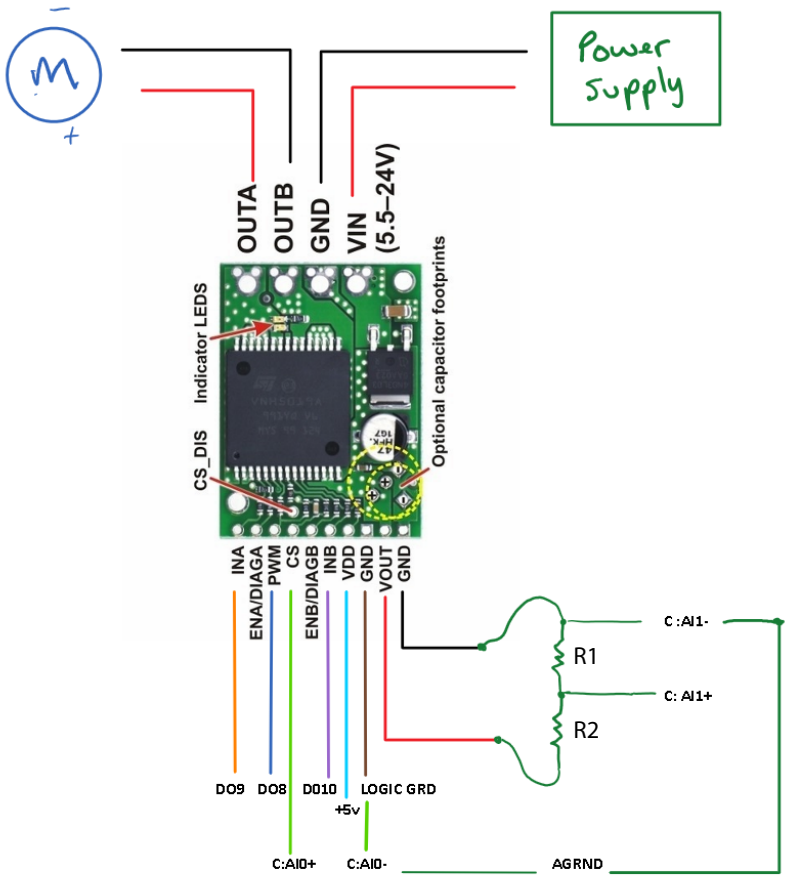


Figure 08.7: the motor driver with labeled pinout connections to the myRIO, power supply, and motor. The motor driver pinout is from pololu.com.

☐ 1. Unplug the myRIO power supply and the motor power supply (large yellow, cylindrical plug).
☐ 2. Measure and record the actual resistances of R_1 and R_2 with a multimeter.

	R_1 (kΩ)	R_2 (kΩ)
nominal		
measured		

☐ 3. Using a breadboard and two 330 kΩ resistors, build a voltage divider to divide the motor driver voltage `VOUT` across resistors R_1 and R_2.
☐ 4. Use Figure 08.7 to make the remaining connections of the motor driver carrier to the myRIO C-connector, with the voltage measurement across R_1.
☐ 5. Plug in the myRIO power supply and the motor power supply.

Lab 08.13 Measurement procedure

☐ 1. Download the following Labview VI and make sure to unzip it to a convenient directory:

`ricopic.one/resources/motorLab.zip`.

☐ 2. Once the VI is open, navigate to the front panel and enter the measured values for the voltage divider resistors R_1 and R_2.
☐ 3. Zero the duty cycle control slider and run the VI.
☐ 4. Play around with the motor speed by sliding the duty cycle control.

Box 08.1 Caution

The direction toggle switch should only be used when the duty cycle has been set to zero! Failure to observe this safety precaution can cause damage to the apparatus and excessive negligence may result in disciplinary action.

☐ 5. After everyone has had a chance to change the motor speed, set the duty cycle to zero.
☐ 6. Use the execute button to run at least 6 cycles with the default parameters, such as 40% duty cycle. The motor should speed up to a steady speed, then slow to a stop, each cycle. If the motor does not slow to a complete stop in each cycle, consider increasing the interval divisor and running it again.

☐ 7. Press the stop button to save your data. Use Matlab to verify your data before you leave the lab.

Lab 08.14 Report requirements

Write a detailed report of your experimental results, as outlined in the report template. Pay special attention to the following.

☐ 1. Estimate the nonzero steady-state value of the angular velocity $\Omega_{J\infty}$. Include a 95% confidence interval.
☐ 2. Use $\Omega_{J\infty}$ to transform the mean step response to decay to zero, take its natural logarithm, and perform a least-squares regression on the resulting data with a linear fit. (If any negative values are input to the natural logarithm, the result will be complex. Mitigate this with the method described in the corresponding Mechatronics lecture. Furthermore, ignore the first five or so of the "fast" time constant τ_2, estimated in the Mechatronics lecture.)
☐ 3. Identify an estimate for the dominant time constant τ_1 from the linear regression. Assign a 95% confidence interval.
☐ 4. From a steady-state analysis of the differential equation (or its solution) governing Ω_J (as described in the corresponding Mechatronics lecture), solve for a transformer ratio estimate $\widetilde{TF}$ in terms of $\Omega_{J\infty}$. Assign a 95% confidence interval.
☐ 5. Plot the sample mean step response with error bars $\pm$ one standard deviation of the means for each point (which is itself a mean). On the same figure, plot the analytic step response (described in a Mechatronics lecture) using all specification sheet parameter values. Also on the same figure, plot the analytic step response using, where appropriate, your estimates $\tilde{\tau}_1$ and $\widetilde{TF}$.
☐ 6. Report all measured parameters (e.g. resistance of resistors).

09

Bibliography

Wikimedia Commons. File:coaxial cable cutaway.svg—wikimedia commons, the free media repository. Online, 9 September 2018a. https://en.wikipedia.org/wiki/File:Coaxial_cable_cutaway.svg.

Wikimedia Commons. File:a few jumper wires.jpg—wikimedia commons, the free media repository. Online, 10 September 2018b. https://commons.wikimedia.org/w/index.php?title=File:A_few_Jumper_Wires.jpg&oldid=292012220.

Wikipedia contributors. 555 internal block diagram. Online, 21 October 2018a. https://en.wikipedia.org/wiki/File:NE555_Bloc_Diagram.svg.

Wikipedia contributors. 555 pinout. Online, 21 October 2018b. https://en.wikipedia.org/wiki/File:555_Pinout.svg.

Marcel B Finan. Existence and uniqueness proof for nth order linear differential equations with constant coefficients. Web, August 2018. http://sections.maa.org/okar/papers/2006/finan.pdf.

Carl Hoefer. Causal determinism. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, spring 2016 edition, 2016.

National Instruments. *User Guide and Specifications NI myRIO-1900*. National Instruments, 376047a-01 edition, August 2013.

E. Kreyszig. *Advanced Engineering Mathematics*. John Wiley & Sons, 2010. ISBN 9780470458365. URL http://books.google.com/books?id=UnN8DpXI74EC.

openclipart boudinpg. Bnc male and female (three images). Online, 9 September 2018. https://openclipart.org/detail/192557/bnc-male-side-view.

openclipart Cheeseness. Circuit board. Online, 9 September 2018. https://openclipart.org/detail/275110/circuit-board.

openclipart goios. Alligator clips. Online, 9 September 2018. https://openclipart.org/detail/159307/alligator-clips.

Derek Rowell and David N. Wormley. *System Dynamics: An Introduction*. Prentice Hall, 1997.

Trilesto. Flashing led using 555 timer. Online, 21 October 2018. https://cdn.instructables.com/FKT/W6NN/H0KKFY8O/FKTW6NNH0KKFY8O.LARGE.jpg.