# Table of Contents

**Table of Contents** ................................................................................................. iii

**Preface** ...................................................................................................................... vii
- Introduction ........................................................................................................ vii
- Troubleshooting ................................................................................................... vii

**Simple DC & AC Circuits** .................................................................................. 9
- 1.1. Measuring Voltages and Currents ................................................................ 13
- 1.2. Thévenin’s Theorem .................................................................................... 15
- 1.3. Output Impedance and Input Impedance ..................................................... 16
- 1.4. Oscilloscope and Function Generator .......................................................... 17
- 1.5. AC Voltage Divider ....................................................................................... 25
- 1.6. Z_{in}/Z_{out} Calculations ............................................................................... 25

**RC Circuits** .............................................................................................................. 26
- 2.1. RC Circuit ....................................................................................................... 27
- 2.2. Differentiator .................................................................................................. 27
- 2.3. Integrator ......................................................................................................... 28
- 2.4. Low-pass Filter ............................................................................................... 28
- 2.5. High-pass Filter ............................................................................................. 30
- 2.6. Filter Example Using a Composite Signal ...................................................... 30
- 2.7. Blocking Capacitor ......................................................................................... 32

**LC Circuits and Rectifiers** ................................................................................ 34
- 3.1. LC Resonant Circuit ...................................................................................... 34
- 3.2. The Diode ....................................................................................................... 35
- 3.3. Half-Wave Rectifier ...................................................................................... 37
- 3.4. Full-Wave Bridge Rectifier .......................................................................... 37
- 3.5. Ripple .............................................................................................................. 38
- 3.6. Three Terminal Fixed Regulator ................................................................... 38
- 3.7. Resistance Measurement Techniques ............................................................ 39

**Basic Op-Amp Circuits** ....................................................................................... 44
- 4.1. Power Supply, Ground and Common............................................................. 45
- 4.2. Inverting Amplifier ....................................................................................... 46
- 4.3. Non-Inverting Amplifier ................................................................................. 47
- 4.4. Follower .......................................................................................................... 48
- 4.5. Integrator ....................................................................................................... 49

**Op-Amp Applications 1** ..................................................................................... 51
- 5.1. Summing Amplifier – D to A Converter ......................................................... 51
- 5.2. Current-to-Voltage Converter ....................................................................... 52
- 5.3. Comparator and Schmitt Trigger ................................................................. 53
- 5.4. Op-amp Relaxation Oscillator ....................................................................... 56

**Op-Amp Applications 2** ..................................................................................... 58
Introduction

This manual consists of 14 individual chapters, each representing about one week of laboratory work. Although we have tried to spread the workload equally among these chapters, you will find that some chapters require more work than others. For example, chapters 2 and 4 require considerably more work than any other chapter; it is to your advantage to study the material in these chapters ahead of time.

Before doing the exercises in the manual, you should do the assigned reading in Horowitz and Hill or Diefenderfer and Holton or Polnus. Try to predict the behavior of a circuit before building it. These exercises are designed to accompany the material covered in class lectures.

Troubleshooting

The exercises outlined in this manual require a three step process: first, building the circuit, second testing, and finally, troubleshooting or fixing the circuit. Steps one and two are always required and you will quickly discover that step three is required most of the time. You will also learn that you will spend most of your lab time on troubleshooting. Do not regard trouble shooting as a waste of time. It will help you obtain a more complete understanding of the circuit.

Here are some suggestions on how to make step three as efficient as possible.

Understand the Circuit

It is almost impossible, and to say the least -- frustrating, to fix something that you do not understand. Instead of spending an endless amount of time exchanging components and checking wires, go back to your textbook (or TA) and make sure you understand the material. Sometimes the circuit does work, yet since the student does not understand what the circuit is supposed to do he or she may spend needless time and effort modifying it.

Check the Wiring

It is a good idea for one person to build the circuit and for the lab partner to check the wiring. Some tips on wiring:
-- color code the wires (RED for positive supply voltages, GREEN for ground and BLACK for negative supply voltages).
-- use the correct type of test leads; it greatly reduces noise and errors.
-- check the ground wires; no ground wires should be left hanging or unattached.
-- use as few wires as possible; beginners tend to use far too many connecting wires.

Check the Components

Sometimes components get mixed up in the bins. This is particularly true for resistors and capacitors as they are often returned to the wrong bin. Make sure you have the correct
component. If it does not seem to work, try another one. After you have tried three of them, you can be certain that it must be something else in your circuit that is causing the problem. If you find faulty components, do not return them to the bins; throw them in the “dead-components” shoebox or trash.

Check the Supply Voltages

Use a scope or a voltmeter to check the supply voltages. Check these voltages at the point where they are connected to your circuit. Often you will find that for one reason or another you forgot to power your circuit.

Check the Voltages inside the Circuit

After verifying that power is indeed applied to the circuit follow the current path and measure voltages at a few easy to calculate points.

Typos

While we cannot claim that there are no mistakes or errors in this manual, there are definitely no deliberate errors in this manual. If we find any mistakes or errors, we will post them on the blackboard in room 65. Also, the circuits in this manual have been tested and in use for the last 10 years and they do work!

Rules of Thumb

Throughout this manual you will find various "rules of thumb." They are approximations and help you remember what you should look for first when dealing with a particular component or instrument. You should completely understand and memorize these rules and you should also understand the limitations to them.

Acknowledgments

This lab manual is based on the first edition of the lab manual by Horowitz & Robinson. Many of the exercises were copied from that manual and adapted to meet our specific needs. Some of the digital exercises were developed by Prof. Mans, Prof. Zimmermann and Prof. Shupe here at the University of Minnesota. Prof. Ruddick contributed some of the exercises and also spent considerable time and effort making the manual readable. Thanks to Prof. Ganz, Prof. Rusack, Prof. Weyhmann, James Flaten, Michael Krueger, Jens Henrickson and Yaroslav Lutsyshyn, Zvie Razieli for their suggestions and proofreading and to Marty Stevens, Jon Huber, Andrew Stewart and Kienan Trandem, Jason Fuglsby, Jessica Santiago for some of the drawings.

Kurt Wick
Summer 1999 / 2012

Copyright Regents University of Minnesota 2012
Write-up Format: Short

Reading:
(This) Lab Manual: Appendix A and F
E4E: Sections 2.1-2.2
Horowitz&Hill: Sections 1.01 - 1.05
Horowitz&Hill: Sections 1.06 - 1.11
Horowitz&Hill: Appendix A and C
or:
Diefenderfer&Holton: Sections 1-1 - 1.13 (pages 1 to 19)
Diefenderfer&Holton: Sections 6-6 - 6-7 (pages 111 to 115)
Plonus: Sections 1.3 – 1.7 (pages 2 to 37)

Introduction Hardware:

This first chapter introduces you to the lab and its equipment. In the first lab session, you and your lab partner will obtain a "breadboard" from your TA. For the rest of this quarter, you will “plug” various components into the breadboard to assemble and to test your circuits. Therefore, you will keep your breadboard until the end of the semester; at the end of each lab session, store it in one of the drawers in lab and clearly mark the label on the drawer with your names.

Study the diagram below and make sure you understand how the wires are connected inside the breadboard!

![Diagram](image.png)

- Figure 1.1. Top-view of a "breadboard" and its internal connections.

On these breadboards, the two outer strips are usually used for ground and supply voltages. Note, these outer strips are disconnected at the center of the board.

To assemble and to connect your circuits you will need wires, cables, test clips and various connectors. Have your TA show you where to find various cables and how to use them.
**Patch Cords and Connectors**

The lab has a large collection of color-coded single-conductor patch cords of various lengths. These cords are terminated with banana plugs or minigrabbers. (See below). While minigrabbers are attached to bare wires or components, banana plugs can only be used with mating connectors, called banana jacks. Most test instruments or power supplies are equipped with such banana jacks. If you want to make an electrical connection between a wire and a banana plug, you need to attach an alligator clip to the banana plug. Also observe that several banana plugs can be stacked together.

- Banana Plug
- Alligator-Clip
- Minigrabber

Single conductor patch cords tend not to be shielded and, therefore, are susceptible to noise. They should be used only for low frequency signals ($f < 1 \text{ MHz}$).

The second most often used patch cords of the lab are coaxial cables (RG 58-U). They are black, thicker and stiffer than the single conductor patch cords. Coaxial cables contain two conductors: a center conductor and an outer conductor. The outer conductor, also known as the shield, is electrically isolated from the center conductor, which it surrounds. To shield a signal, the outer conductor is usually held at ground while the signal is fed through the center conductor.

Since coaxial cables contain two conductors, they are terminated at each end with either two connectors (two banana plugs, two minigrabbers etc.) or most commonly, a BNC connector. (BNC stands for Berkeley Nucleonics Corporation, the first company to manufacture these connectors.) The sleeve of the BNC connector is always tied to the shield of the coaxial cable while the signal passes through the center pin of the connector.

- Coaxial cable with male BNC connector

BNC connectors come in two genders, male or female. The center conductor determines the gender.
To join two or more BNC cables together, use the following connectors:

- BNC “Tee”
- BNC Barrel

Sometimes it is necessary to join cables with BNC connectors to banana plugs. This can be done with the two adapters shown below. **Note the little extrusion shown on top of the adapters.** It identifies the banana connector connected to the shield of the BNC, which is usually its ground connection.

- Banana (Jack) to Male BNC Adapter
- Banana (Jack) to Female BNC Adapter

The BNC to banana adapter shown below is used very often to connect a BNC cable to a test instrument with banana jacks. The distance between adjacent banana jacks is standardized to allow for such adapters. For example, all our digital voltmeters can be connected this way.

- Female BNC to Banana Plug
A BNC terminator contains a shielded resistor, typically 50 Ω, that connects the shield and the center connector. It is used to reduce noise and reflections of very fast pulses.

- BNC 50 Ω Terminator

### Introduction: Theory

This chapter introduces two fundamental circuit analysis concepts: voltage and current dividers and input/output impedance. The concept of the voltage divider is very important because the analysis of the most complicated circuits is often accomplished by resolving the circuit into combinations of voltages or current dividers. The concept of input and output impedance is fundamental in understanding amplifiers, attenuators and transmission lines; it is explained in appendix A of this manual.

Exercise 1.5 introduces you to the oscilloscope. Since that section contains a lot of information and explanations you may want to glance over it before you come to the lab.

Finally, here is the first rule of thumb that you should remember and a note you should carefully read.

**Rules of Thumb about Volt- and Ammeters:**

- An ideal voltmeter acts like an open circuit; it has an infinite resistance.
- An ideal ammeter acts like a short circuit (i.e. an ideal wire); it has zero resistance.

**Note:** Dealing with the voltages and currents as indicated in the exercises in this book is safe for both humans and instruments. All instruments contain various safeguards to prevent them from being destroyed by you, or from destroying you if they are used improperly; so do not worry too much if you should make a mistake or if you are new to electronics. **However,** one exception is the ammeter or any device that can be set into a current measuring mode, such as a Digital Multimeter (DMM). Little or no protection is built into such an instrument and improper use will simply destroy the instrument. Hence, always stop and think first before turning on the power in any circuit which incorporates an ammeter. Check that the current measuring device is:
  1) ALWAYS hooked in SERIES with a current limiting device such as a resistor, and
  2) NEVER is hooked directly ACROSS a voltage source.
1.1. Measuring Voltages and Currents

Figure 1.2.a. shows a simple circuit containing a power source and a circuit element called the "device under test" or D.U.T. In this particular drawing, the power source is a battery though it also could have been a power supply, a signal generator or a transducer. The D.U.T. selected here is a resistor though it could also have been any other passive or even an active electronic component.

To measure either the voltage or the current characteristics of the D.U.T. the following configurations are used:

In both cases, "voltmeter" and "ammeter" may refer to the same physical instrument. In the first case it is in its "voltmeter" mode, in the second case it is in its "ammeter" mode. Note that for both configurations the previous rules of thumb are satisfied: the voltage is always measured with the meter across the D.U.T., i.e., the voltmeter is in parallel with the D.U.T.; the current is always measured through the device, i.e. the ammeter is in series with the D.U.T.

If you already have a working circuit, measuring the voltage across a D.U.T. is usually less complicated than measuring the current because you can connect the voltmeter directly across the D.U.T. without having to rearrange the circuit. On the other hand, if you want to measure the current you need to disconnect or "break" the working circuit before or after the D.U.T. and insert the ammeter at that point.

To determine the electrical properties of a device, the current and the voltage are measured simultaneously while the power source is adjusted. The data is then plotted as an I-V plot and conventionally the voltage is plotted along the x-axis and the current along the y-axis. You will create such a plot shortly when you observe Ohm’s law.

There are two possible configurations to measure both the current and the voltage of the D.U.T. simultaneously and they are shown below:
Except for the rare case when the resistance of the D.U.T. is very small (on the order of the internal resistance of the ammeter (typically a few Ohms) or smaller) or very large (on the order of the internal resistance of the voltmeter (typically a few M\(\Omega\)s) or larger) it does not matter which configuration you use in Figure 1.2.c. Deciding which of the two arrangements is more suitable for each of the two extreme cases stated is left to you as an exercise.

**Exercise:**

Measure and plot \(I\) vs. \(V\) for a couple of resistors. First use a 33 k\(\Omega\) resistor and then repeat your experiment with a 120 \(\Omega\) resistor. As power source, use the adjustable DC power supply, the Agilent 3630A and connect your circuit to the COM and the +20V output. Adjust the \(\pm 20V\) knob on the power supply and obtain at least 5 different pairs of \(I-V\) readings per resistor using either one of the configurations in figure 1.2.c. You must use one digital meter (DVM) and one analog meter for your measurements. (You may not use two digital meters!)

**Write-up**

1.1.1. Measure and plot at least 5 different pairs of \(I-V\) readings for a 33 k\(\Omega\) resistor. Plot the \(I-V\) data using standard conventions as explained earlier.
1.1.2. Repeat 1.1.1 with a 120 \(\Omega\) resistor. What does its power rating of 1/8 Watt imply in terms of maximum currents and voltages? Calculate these values for the 120 \(\Omega\) resistor and indicate on your plot the region where it is safe to operate the 120 \(\Omega\) resistor.
1.1.3. Graph what the \(I-V\) relationship would look like across:
   a) an open-circuit configuration
   b) a short circuit
   c) a wire with zero resistance

(For b and c, assume that you are using a "real" voltage source with a small, non-zero output impedance.) **Do not measure it**; use your intuition or talk to your lab partner or TA. In your lab report, complete the table below:
### Thévenin's Theorem

<table>
<thead>
<tr>
<th></th>
<th>Current</th>
<th>Voltage</th>
<th>Resistance</th>
</tr>
</thead>
<tbody>
<tr>
<td>a) open circuit</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>b) short circuit</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>c) wire</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Table 1.1.

Of the three cases (a, b, c) which two are identical? Since open and short-circuit concepts will be used continuously throughout the rest of this course be sure that you understand the above table!

1.1.4. You are to simultaneously measure $I$ and $V$ for a D.U.T. which has an extremely large resistance, i.e., a resistance considerably larger than the internal resistance of any voltmeter available to you. Which of the two configurations shown in figure 1.2.c is “better” and why?

#### 1.2. Thévenin's Theorem

Thévenin’s theorem is usually applied to describe a complicated circuit in terms of a much simpler circuit that has similar characteristics. In this exercise we will determine the Thévenin equivalent circuit of the voltage divider shown in figure 1.3. We then will build the Thévenin equivalent circuit and compare its characteristics with the original circuit.

Appendix B of this manual (or section 1.05 of H&H or section 1.11 of D&H) explains how to find the Thévenin equivalent circuit by calculating $V_{TH}$ and $R_{TH}$. The Thévenin equivalent circuit can also be determined by measuring the appropriate currents and voltages which is what we will do in this exercise. Since we are unable to measure $R_{TH}$ directly by connecting a meter to the circuit (because it may produce the wrong value and it also could damage the meter or the circuit) we need to measure $V_{TH}$ and $I_N$; $R_{TH}$ can then be calculated directly from these values.

First, measure $V_{TH}$ or the (open-circuit) output voltage, $V_{out}$. Second, measure the short-circuit current, $I_N$, across the output. Finally, calculate $R_{TH}$. Before you replace the circuit with its Thévenin equivalent circuit measure $V_{out}$ for a load: Attach a load, arbitrarily chosen to be 10 kΩ, and measure $V_{out}$ again.

Now build the actual Thévenin equivalent circuit, using a variable regulated DC power supply as the voltage source for $V_{TH}$ and check that its open-circuit voltage and short-circuit current are similar to the previously measured ones. Then attach a 10 kΩ load, just as you did with the original voltage divider and see if it behaves identically.

#### Write-up

1.2.1. Calculate the Thévenin equivalent circuit for the circuit in figure 1.3. (See appendix B or H&H.) Hand in a drawing of the Thévenin equivalent circuit; specify the values of the components.
1.2.2. Hand in a drawing of the Thévenin equivalent circuit (for the circuit in figure 1.3.) which you obtained from measuring $V_{TH}$ and $I_N$; again specify the values of the components.

1.2.3. Complete the table below with your measured values:

<table>
<thead>
<tr>
<th>Circuit in Figure 1.3.</th>
<th>Vout: Open Circuit</th>
<th>Vout: 10 kΩ Load</th>
<th>Short-circuit current</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thévenin Equivalent</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Table 1.2.

1.2.4. Show (draw a schematic) how you measured the short-circuit current. Explain which assumption you made for the ammeter.

### 1.3. Output Impedance and Input Impedance

Before you start the experiments in this section be sure to read Appendix F & A first!

A "real" battery can be thought of as an "ideal" voltage source with a series resistance; in batteries, this series or output impedance is often referred to as the "internal resistance." Measure $V_{Source}$ and calculate $Z_{out}$ for three of 1.5 V D-type batteries; use $R_{load} = 33 \, \Omega$. From your measurements, can you say what determines whether a battery is "good" or "bad"?

Measure the input impedance of your DMM (Digital Multimeter) using only the DMM itself. (If you use any additional meters you will probably obtain incorrect values!) Adjust your DC power supply to output approximately 10 volts and measure it accurately with your DMM. Now insert a 1 MΩ resistor in series with the power supply and the DMM. Measure the voltage again. (The measurement with the 1 MΩ in place differs from the first measurement because the DMM's input impedance is affecting the measuring process.) For both cases draw a circuit diagram; indicate clearly in each which voltages you measured. (Hint: think of a "real" voltmeter as an "ideal" voltmeter (with infinite input resistance) in parallel with a very large resistor of value $Z_{in}$; for simplicity, assume for the power supply $Z_{out} = 0$, i.e. an ideal power supply.) Compare your circuit diagrams to figure A.4. and calculate $Z_{in}$. Compare your value to the value the manual quotes: 11 MΩ.

By now the following rules of thumb should be "obvious":

- **The output impedance of an ideal voltage source is zero.**
- **The input impedance of an ideal voltmeter is infinite.**

### Write-up

1.3.1. What is $Z_{out}$ for three different 1.5 V batteries? Show how you arrived at the values.

1.3.2. What is $Z_{in}$ for the DMM? Show how you measured this value; specifically, draw circuit diagrams and indicate clearly which voltages you measured.

1.3.3. There is a method for measuring output and input impedance which is suitable for people who abhor calculations: To measure the output impedance, first measure $V_{open}$. Next, attach $R_{load}$ to the circuit and adjust $R_{load}$ until $V_{load} = 1/2 \cdot V_{open}$. Now measure $R_{load}$. Knowing $R_{load}$ one can determine $Z_{out}$ without any further calculations. What is it?
1.4. Oscilloscope and Function Generator

We will be using the oscilloscope ("scope"), a Tektronix 2245A, and the HP33120A function generator frequently. Become familiar with their operation by connecting a BNC-to-BNC cable from the function generator’s OUTPUT to the scope CH1 input. Turn the function generator on and reset it by pressing the Recall button and the up or down arrow keys (\(\uparrow\) or \(\downarrow\)) until the display shows: RECALL 1. Now press the Enter button.

![Figure 1.6. Tektronix 2245A Analog Oscilloscope Front Panel.](image)

In order to display a signal fed into channel 1, you must "initialize" the following five scope settings:

<table>
<thead>
<tr>
<th>Function</th>
<th>Button</th>
<th># (see Fig 1.6)</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vertical Display Source</td>
<td>MODE</td>
<td>1</td>
<td>CH 1</td>
</tr>
<tr>
<td>Vertical Display Mode</td>
<td>COUPLING</td>
<td>2</td>
<td>AC</td>
</tr>
<tr>
<td>Trigger Setting</td>
<td>MODE</td>
<td>3</td>
<td>AUTO</td>
</tr>
<tr>
<td>Trigger Setting</td>
<td>SOURCE</td>
<td>4</td>
<td>CH 1</td>
</tr>
<tr>
<td>Trigger Setting</td>
<td>CPLG</td>
<td>5</td>
<td>DC</td>
</tr>
</tbody>
</table>

* Table 1.3. Initial Scope Settings

If both the scope and the function generator are turned on, adjust:

<table>
<thead>
<tr>
<th>Function</th>
<th>Button</th>
<th># (see Fig. 1.6)</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 CH1 Vertical Position</td>
<td>POSITION</td>
<td>6</td>
<td>Adjust until signal is centered on the screen. (If you still don’t see anything increase INTENSITY!)</td>
</tr>
<tr>
<td>2 Horizontal Sweep Speed</td>
<td>A and B SEC/DIV</td>
<td>7</td>
<td>Adjust until you see a few cycles of the signal.</td>
</tr>
</tbody>
</table>
3 | CH1 Vertical Gain | VOLTS/DIV | 8 | Adjust until signal fills screen.
---|------------------|-----------|---|------------------
4 | Horizontal Position | POSITION | 9 | Adjust until you can see the beginning of the trace. (You rarely care what the end of the trace looks like!)

- Table 1.4.

This procedure works on most scopes, so become very familiar with it. If you do not understand what all these buttons do, read on; their functions are explained in the sections below.

Note that the Tektronix 2245A scope is equipped with an "AUTO SETUP" button. Locate it on the front panel. Activating this button starts a process similar to the one outlined above and it (usually) results in a display of some sort of signal. Most of the time the process works but generally speaking, the scope prefers to display high frequency signals which often represent noise and not the signal that we are interested in. So use this button with caution! In case your next scope does not have an AUTO SETUP function or it does not display the signal you are interested in then you need to become familiar in setting up the scope from scratch; this is outlined in the next section.

1.4.1. Scope Introduction

"Usually" the oscilloscope is used to display one or more input voltage signal vs. time. The voltages are plotted on the vertical, i.e., the y-axis while the time is displayed along the x-axis. Though most scopes can also be used in an XY mode, where one voltage is displayed against another, for the rest, we will only consider the much more commonly used voltage vs. time mode.

There are 4 parts to a scope. Luckily, all the knobs are grouped in easily identifiable sections:

1) **Time Base**: The A and B TIME/DIV setting controls the time axis (x-axis). It specifies the horizontal velocity (sweep rate) at which the electron beam is continuously swept from left to right across the display.

2) **Vertical Section**: The VOLTS/DIV setting(s) controls the vertical gain of the input voltage(s) as they are displayed along the y-axis.

3) **Trigger**: The trigger buttons specify at what point in time the electron beams begins each scan across the display.

4) **Display Adjustments**: This section of knobs is located directly below the screen. These control the visual appearance of the signal on the screen, such as the signal's focus and intensity. Since they should be self-explanatory we will not discuss them any further.

How does it work? Check for yourself and use the following scope settings:

- In the time base (A and B TIME/DIV) select 0.2 s / division.
- Set the (left) vertical amplification VOLTS/DIV for channel 1 to its maximum value, i.e., all the way counter clock wise, 5V / division.
- In the vertical MODE chose Ch1.
- Set the TRIGGER MODE to AUTO.
- Use no input signal, i.e., temporarily disconnect the cable.

Adjust the (vertical) POSITION knob for channel 1 until you see a dot slowly wandering from left to right across the display. This dot represents the voltage at a given time and without any input it should remain constant with respect to the y-axis. What you are seeing are the electrons from the electron gun striking a tiny circular section of the fluorescent screen causing it to glow in the form of a tiny bright dot while the horizontal section is slowly sweeping the beam (horizontally) across the display. Though the screen will glow where the electrons strike it, in the absence of electrons, the glow will quickly fade and then disappear.

Now change the time base to faster sweep velocity, for example, 1 ms / division. As you change the time base, you will observe how the small dot transforms itself into a horizontal line. That event in itself is actually nothing spectacular but it really forms the basis on how the scope works.
It is WRONG to think that at higher speeds the dot spreads itself out into a horizontal line. It is still as small as before but it is now racing rapidly repeatedly across the display. The combination of the afterglow of the phosphorescence and the retina retention makes it appear like a continuous line! (Retina retention is the physiological effect that you will see something for about 1/10 of a second after it has already disappeared.)

1.4.2. Scope Trigger

We start with the scope’s trigger settings because they are essential for operating a scope. To get a “clean” display of a periodic signal it is essential that each trace overlaps with the previous one, i.e. that they are all “in sync.” An example of a periodic signal that is not in synch is shown below.

The synchronization is accomplished through the trigger. The scope starts a trace each time it receives a valid trigger event. It is created when the input signal reaches a user specified (trigger) voltage level. This level is set by the trigger LEVEL knob.

Reconnect the HP 36120A’s OUTPUT (the lower of the two BNC connectors) to the scope’s CH1 and adjust the horizontal sweep speed (i.e. A and B TIME/DIV knob) until you observe a few periods of a sine wave. With the trigger MODE set to AUTO, carefully watch the beginning of the trace and slowly turn the Trigger LEVEL knob. Note that when the trigger level goes past the signal’s max/min values, the signal on the screen appears messy because the scope no longer receives a trigger signal. Instead, when in AUTO mode, if after a preset amount of time no trigger signal has been received, the scope will automatically (and arbitrarily) start a new trace, resulting in
the messy display shown above. In other words, in AUTO mode, the scope will still display a trace even without a trigger signal.

Next move the trigger MODE knob, (#3) to NORM and try the normal triggering mode; again adjust the trigger level. In this mode, the trace is only visible when the input signal crosses the trigger level that has been set with the LEVEL knob. If the trigger level lies outside of the signal’s range, no signal at all is displayed. Hence, it is probably a good idea to leave the scope in AUTO mode most of the time.

Figure: Display of a signal that is in-sync. The trigger point is indicated by the black dots. Note: though the display shows only one trace, it is actually composed of overlapping traces that are in sync. As the old trace fades away, it is “redrawn” by the new ones which are identical.

Note that whenever a specific voltage level is specified within a periodic signal, this level is reached twice within one period, once with a rising slope and once with a falling slope. Hence, to specify the intersection uniquely, the slope of the signal at the trigger level must also be specified. As you probably guessed, this is done by pressing the SLOPE button. Again play with the trigger level and press the SLOPE button and see if it behaves the way you expect it.

1.4.3. Input Channel Selection

Next consider the last two settings in table 1.3, namely the signal source and signal input coupling mode.

The signal source, or vertical MODE button (#1), selects the input channel. Setting the vertical MODE button to CH1 will display the signal fed into the channel 1, from the BNC input labeled CH1. Similarly, the CH2 setting will display the signal from channel 2, from the BNC input labeled CH2. Connect an additional coaxial cable between the SYNC output of the HP function generator and the channel 2 input of the scope. Switch between channel 1 and 2 and you should see a square wave on channel 2. (You may have to adjust the vertical gain i.e., VOLTS/DIV for channel 2.)

The vertical MODE switch can also be used to display both channels 1 and 2 at the same time. Since most scopes have only one electron gun to draw the traces on the screen, two different methods are used to display both traces at the same time. In the ALT mode, the complete trace of
one channel is drawn and then the trace of the other, i.e. the gun alternates drawing the two channels. See the figure below.

Figure: Scope display in the ALT mode.

At high sweep speeds, it will appear as if both channel traces are drawn at the same time but if you lower the sweep speed to 10 msec/div, you can clearly observe the alternating drawing. Try it: set V MODE to ALT and set the sweep speed to 10 msec/div. If you want to compare two signals at sweep speeds even lower than that, the ALT mode can become annoying.

This problem can be overcome by setting the V MODE into the CHOP mode. In this mode, the gun rapidly alternates between drawing small parts of each trace. At high sweep speeds, this mode may result in a trace that appears grainy or "chopped", particularly on older scopes.
1.4.4. Calibration & Measurements

Voltages or time intervals are measured by counting the number of divisions a signal takes up on the display and then multiplying this value by the corresponding knob setting. (Note: a division on the display is one square, about ½” wide.) For example, if a signal is two divisions large and the vertical gain is set at 1 Volt/division, then one should be able to assume that the signal is 2 V. This answer can be wrong if the vertical gain is not in its calibrated mode or if you should use a scope probe.

Unfortunately, the VOLT/DIV value or A and B SEC/DIV is correct only if it is in its calibrated mode, i.e., when the inner knob of the vertical gain adjustment is completely turned cw! Try it: turn the inner knob of the vertical gain (do not turn the outer knob, the VOLT/DIV knob) and turn the A and B SEC/DIV. Note, that whenever these knobs are in their uncalibrated position, a red warning light labeled UNCAL, directly adjacent to it, will turn on to indicate that your readings will be useless!

Therefore, always check that all these red warning lights are turned off! Note that these knobs are rarely ever used. Nevertheless, people love to abuse them and they are the major source of errors in faulty measurements.

For your convenience, you can measure and display the voltage, time and frequency directly from the scope display by activating on the front panel (top left) the appropriate MEASUREMENTS CURSOR, such as VOLTS, 1/TIME, TIME. The cursors can be positioned by using CURSOR /
TIME POSITION knobs. As long as your vertical and horizontal settings are in the “calibrated” mode, the display will calculate and display the values between the cursors.

Your readings can also be incorrect if you measure your signal using one of the scope probes that you may find in the lab. (Ask your TA to show you one if you are unsure what they look like.) The advantage of a scope probe (over a “simple” cable) is that it provides 10 MΩ input impedance, as opposed to the scope’s 1 MΩ. This gain in input impedance comes at a cost: the signals measured are reduced by a factor of 10. (Essentially, the scope probe is a 1/10 voltage divider.) Some fancy scopes are able to sense if a scope probe is being used and then automatically adjust the gain or indicate the 10x loss on the vertical gain knob. The Tektronix scope used in the lab is able to do that as long as you use scope probes manufactured by Tektronix. Nevertheless, if you use any other scope, it might be completely unaware if you use a scope probe or a cable. So if you should ever use a scope probe, be aware of this and adjust your measured value accordingly. Also note that not all scope probes are necessarily divide by 10 circuits! (Some have adjustable gain settings.) In short, to avoid confusion, avoid scope probes for now.

1.4.5. HP 33102A Function Generator

You will use the function generator as a signal source for most of the circuits that you will be building in this course.

The function generator has been configured to remember its settings when it is shut down. This can be annoying because the function generator has many different settings and you can never be certain which ones have been changed by a previous user.

Therefore, when you first turn it on, or if you have problems with it, reset it:
- Press the “Recall” button.
- If necessary, use the up “↑” or down “↓” arrow keys until the display shows: RECALL 1.
- Press the “Enter” key.

In the Recall 1 mode you should get a 1 kHz sine wave with a 200 mVpp amplitude and no offset. If this is not the case, inform your TA of the problem. (For additional troubleshooting, especially if you should measure an amplitude twice the value displayed, see also the web at: http://mxp.physics.umn.edu/f99/trouble)

The signals used for testing the circuits are usually sine, square and triangle waves. Connect the function generator's OUTPUT to CH1 on the scope and SYNC to CH2, as has been described in the previous sections. Push the appropriate buttons in the row labeled “Function / Modulation” and observe the corresponding signals on the scope. What happens to the SYNC output as you change from sine to square to triangle wave? (Do not get misled by the “overshot,” the small spike at the rising edge, present on the SYNC signal; in an ideal world, the SYNC signal would be a perfect square wave. Assume for this exercise it approaches such a square wave.)

You modify the frequency, amplitude or offset of a signal by pressing the corresponding buttons in the bottom row labeled “MODIFY.” Once you have selected an attribute that you want to modify, its current value is displayed. You can then change it by using one of three methods:
- Use the right or left arrow keys to select the digit you want to increment or decrement and then turn the round knob.
- Use the right or left arrow keys to select the digit you want to increment or decrement and then hold the up or down arrow keys.
- Push the “Enter Number” button and then enter a numerical value. (The numerical values are printed in green to the lower left of the buttons.) Complete the entry by pressing the arrow key that corresponds to the appropriate unit which is printed in green to right of it.

Which method you use is up to you.
With the scope CH1 connect to the OUTPUT and CH2 to SYNC, adjust the amplitude of a sine, square and triangle wave signal. In what way does the output signal from OUTPUT and SYNC differ when you adjust the amplitude?

Choose a waveform and change its frequency. In what way does the output signal from OUTPUT and SYNC differ when you adjust the frequency?

Summing up, **describe** how the SYNC signal is affected by the waveform, amplitude and frequency selected.

**Note:** Unless specified otherwise, whenever we talk about the "output signal" from the function generator, we refer to "OUTPUT" output. Therefore, always connect your scope and circuits to the "OUTPUT." The SYNC output is rarely used.

### 1.4.6. Signal Input Coupling

Finally you get ready to measure some "real" signals. A sinusoidal signal can be expressed by:

\[
V(t) = A + B \sin \left( \frac{2\pi t}{T} \right)
\]

(1.1.)

"A" is usually referred to as the DC offset voltage, "B" as the amplitude and "T" is the period. **Draw** the above function and clearly indicate in your picture what \(A\), \(B\) and \(T\) are.

If you followed the instructions, you had your channel 1 input coupling set to AC. In this mode any DC offset is ignored and equation (1.1.) reduces to:

\[
V(t) = B \sin \left( \frac{2\pi t}{T} \right)
\]

(1.2)

Hence, if all you want to measure is the amplitude and frequency of a signal this mode is fine. Measure (with the scope) and **report** the largest sine wave amplitude that can be generated by the function generator OUTPUT at 1 kHz and 15 MHz. (Check that the DC OFFSET is off, i.e., set to zero.) For the two frequencies, what is the percentage error between the frequency set on the function generator and the frequency measured from the scope?

To measure a DC offset is a two-step process. First, you need to know where ground potential is situated on your screen (i.e. where \(V = 0\) is) since its position can be arbitrarily adjusted by the Vertical Position knob. Set the input coupling to GND (ground) by unselecting both AC and DC coupling buttons; this shorts out the input (i.e. physically disconnects the input signal) and turns equation (1.1.) into:

\[
V(t) = 0
\]

(1.3.)

Set your trigger MODE to AUTO and adjust your Vertical Position until you see the corresponding signal on the screen. Position it vertically wherever you consider it convenient and remember where that location is. **Explain** to your lab partner why you cannot get a steady display with trigger MODE set to NORM. (Try it.)

Secondly, switch the input coupling lever to DC. Adjust the function generator to produce a 0.5 V amplitude, 1 kHz sine wave. Now, adjust the function generator DC OFFSET from one extreme position to the other. **Report** the maximum positive and negative DC offset voltages measured.
Remember, we said that with the input coupling set to AC, any DC offset voltages are ignored by the scope. Check that it is true by setting the input coupling to AC and again change the function generator’s DC offset voltage.

**Write-up**

1.4.1. Describe how the SYNC signal is affected by the waveform, amplitude and frequency.
1.4.2. Draw the function (1.1.) and clearly indicate in your picture what A, B and T are.
1.4.3. Measure (on the scope) and report the largest sine wave amplitude that can be generated by the HP function generator’s OUTPUT at 1 kHz and 15 MHz.
1.4.4. Report the maximum positive and negative DC offset voltages measured on the scope for a 0.5 V amplitude, 1 kHz sine wave.

### 1.5. AC Voltage Divider

![Voltage Divider Diagram](image)

Figure 1.11. Voltage divider from Figure 1.3 driven by a function generator.

How would the analysis of a resistive voltage divider be affected by an input voltage that changes with time (for example, a sinusoidal input signal)? Hook up the voltage divider from lab exercise 1.2 and replace the 15 V voltage source with a function generator; see Figure 1.11. Use the scope to see what the function generator’s 1 kHz sine wave does to the output by comparing the input and output signals.

**Write-up**

1.5.1. Draw pictures of $V_i$ vs. $t$ and $V_o$ vs. $t$ as observed on the scope. Draw both pictures on the same graph.
1.5.2. Explain in detail (use equation 1.1.) why it must act that way.

### 1.6. $Z_i / Z_o$ Calculations

Find the answer to any three of the five problems, F.6.1 to F.6.5., in Appendix F.6. at the end of this manual.

**Write-up**

1.6.1. Hand in the solutions to three of the five problems in Appendix F.6.
Write-up Format: Short

Reading:

E4E: Sections 2.3-2.5
Horowitz & Hill: Chapter 1.12 - 1.21
Horowitz & Hill: Appendix B

or:

Diefenderfer & Holton: Sections 2.1 - 2.6 (pages 23 to 40)
Diefenderfer & Holton: Sections 3.1 - 3.4 (pages 43 to 55) (skip material on phasors)
Diefenderfer & Holton: Appendix A-1 (pages A-4 to A-8)
Plonus: Section 1.8 (pages 37 to 40)
Plonus: Section 2.3 (pages 58 to 62)

In the exercises in this chapter, you will observe how various circuits alter the signals supplied to them. To understand the output signal it is essential that you have both input and output signals displayed at the same time on the scope so that you can compare the two. Hence, for all exercises in this chapter (and most later on) set the scope into ALT or CHOP mode. Use a BNC Tee to split the output from the signal generator. Feed one of these signals directly to scope channel 1 and the other to your test circuit. Feed the output signal from your test circuit into channel 2.

Trick: When comparing two signals, always feed the “cleaner” signal (which is almost always the signal from the function generator’s OUTPUT) into channel 1 and the messier signal (usually the one coming out of your test circuit) into channel 2. The reason being that the scope by default always triggers on channel 1 and it is much easier to trigger on a clean signal than a messy one.

Rules of Thumb about Capacitors:

The following approximations are often useful:

- at low frequencies and DC, a capacitor acts like an open circuit (with infinite input resistance) and can be ignored.

- at (very) high frequencies a capacitor acts like a short circuit (with zero resistance) and can be thought of as a wire.
2.1. **RC Circuit**

![RC Circuit Diagram]

Verify that the RC circuit in figure 2.1 behaves in the time domain as described in section 1.12 of H&H (or section 2.5 of D&H.) Drive the circuit with a 500 Hz square wave, and look at the output.

**Write-up**

2.1.1. Measure the time constant by determining the time for the output to drop by 63%.
2.1.2. Calculate the time constant based on the listed component values.
2.1.3. What is the percentage error between the calculated time constant and the measured time constant?
2.1.4. What percentage error should you expect considering the tolerance of the components in your circuit? (Remember that when you multiply independent quantities, the propagation of errors goes like the square root of the sum of the squares of the percentage errors.) Assume that you know value of the capacitor to within 20%; check Appendix C of H&H or D&H p.7 for the tolerance of the resistor.

2.2. **Differentiator**

![Differentiator Diagram]

Construct an RC differentiator (figure 2.2). Drive it with a large 100 kHz triangle wave using the function generator. Carefully observe the output signal on the scope. Next try a 100 kHz square wave.

**Write-up**

2.2.1. Draw the circuit's output from the 100 kHz triangle wave and square wave.
2.2.2. Write down explicitly a function \( f(t) \) for a triangle wave for one period \( T \) with slope \(+A\) for \( 0 < t < T/2 \) and slope \(-A\) for \( T/2 < t < T \); now calculate the function's derivative \( df/dt \). Plot \( f(t) \) and \( df/dt \). Does it look similar to what you observed in 2.2.1?
2.2.3. Write down explicitly \( f(t) \) and \( df/dt \) for a square wave and plot them. How do they compare to what you observed in 2.2.1.
2.2.4. Using the “rules of thumb”, what are the circuit's input and output impedances at zero frequency? At infinite frequency? Questions like this become important when the signal source is less ideal than the function generators you are using.

### 2.3. Integrator

![Figure 2.3. RC integrator.](image)

- Figure 2.3. RC integrator. (Note: Unlike figure 2.2., the ground terminals are not shown at the input and the output. From now on it will be assumed, unless stated otherwise, that all signals are applied and measured with respect to ground.)

Construct an integrator (figure 2.3). Drive it with a 100 kHz square wave at maximum output level. Drive it with a triangle wave. Both the differentiator and integrator circuits are approximations. In Chapter 4 we will see how to make “perfect” differentiators and integrators using operational amplifiers.

#### Write-up

2.3.1. Show the circuit’s output from the 100 kHz triangle wave and square wave; what is the form of the output waveform?

2.3.2. Check your results by integrating the corresponding \( f(t) \) from exercise 2.2.2 and 2.2.3. For both cases, plot \( \int f(t) dt \) and compare with 2.3.1.

2.3.3. What are the input and output impedances of the circuit at DC? At infinite frequency?

2.3.4. Under what circumstances can these circuits be treated as true differentiators and integrators and when does the approximation fail?

### 2.4. Low-pass Filter

![Figure 2.4. RC low-pass filter.](image)

- Figure 2.4. RC low-pass filter.

The Transfer Function of a circuit is defined as the ratio of: \( V_{out}/V_{in} \) (which is just a fancy way of saying "the gain" or "the attenuation" of a circuit). It is a dimensionless ratio and is either expressed in percents or in Decibels, dB, defined as:

\[
\text{Voltage Gain (or Attenuation) in dB} = 20 \log_{10} \left| \frac{V_{out}}{V_{in}} \right|
\]  

(2.1.)

Construct the low-pass filter in figure 2.4.
While adjusting the frequency of the input sine wave from 100 Hz to 20 kHz observe its output, $V_{out}$ on the scope. You should see two distinct regimes: the output remains more or less constant and independent of the frequency; the output changes substantially when the frequency changes.

The -3dB point (also known as the "break-point frequency," $f_B$) is then defined as the frequency for which the filter changes regimes. More specifically it is defined as the frequency for which the transfer function, i.e., $V_{out}/V_{in} = 1/\sqrt{2} \approx 70\%$. (If you use equation 2.1. to convert this ratio into Decibels you will find that this corresponds roughly to a value of -3, hence, the name -3 dB point!)

Now find the -3dB point of the circuit in Figure 2.4. by adjusting the frequency of the input sine wave until you observe the ratio of quoted above. Measure this frequency, $f_{3dB}$. Next measure the output at a frequency 10 times $f_{3dB}$ and an octave above that, i.e. at 20 times $f_{3dB}$. At the same time observe the behavior of the phase shift $\phi$ vs. frequency: what is the phase shift between the input and output signal for $f << f_{3dB}$, $f = f_{3dB}$ and $f >> f_{3dB}$? (Note: to measure a phase shift on the scope, you must display the input and output signals simultaneously using the scope's two input channels!)

Finally, note the similarity between the circuits in figure 2.3 and figure 2.4. Except for the component values the circuits are identical! So whether you call this circuit an integrator or low-pass filter is really arbitrary. Change the input waveform back to a triangle wave and see if you can determine the frequency regime over which the circuit in figure 2.4. behaves like an integrator.

Write-up

2.4.1. What is the measured -3dB frequency?
2.4.2. Calculate explicitly the transfer function for the above low-pass filter, i.e. find $V_{out}/V_{in}$ in terms of $R$, $C$, and $\alpha$. Hint: think of the circuit as a voltage divider with

$$\frac{V_{out}}{V_{in}} = \frac{Z_C}{Z_C + Z_R} \quad (2.1.)$$

where $Z_C$ and $Z_R$ represent the complex impedances of the capacitor and the resistor. Using only the magnitude or "modulus" of the transfer function, (i.e. $|V_{out}/V_{in}|$) does your result agree with the result stated on page 37 of H&H (or page 53 of D&H, equation 3-41)?

2.4.3. Use the definition for the -3dB point, which is:

$$\left|\frac{V_{out}}{V_{in}}\right| = \frac{1}{\sqrt{2}} \quad (2.2.)$$

and the transfer function calculated in 2.4.2 and explicitly solve for $f_{3dB}$.

2.4.4. To what percentage does it agree with the frequency that you measured in 2.4.3?

(Be sure you understand the relationship between the measured and calculated -3dB point and become familiar converting between decibels and "normal" $V_{out}/V_{in}$ ratios! If you have problems with complex numbers read Appendix A-1 of D&H or Appendix B of H&H or bug your TA!)

2.4.5. Now calculate directly from the transfer function how many decibels per octave $V_{out}$ changes for very high frequencies. (Hint: From the transfer function, find $V_{out}(f)/V_{in}(f)$ in dB as $f$ becomes very large; observe how it changes when $f$ increases to $2f$ and note that $2f$ is one octave above the fundamental frequency, $f$.) Can you describe the same thing in plain English without using decibels and octaves; in other words, what happens to $V_{out}$ each time you double the frequency?

2.4.6. Compare your calculated value to your measured outputs at a frequency 10 times $f_{3dB}$ and an octave above that.

2.4.7. What was the observed phase shift for $f << f_{3dB}$, $f = f_{3dB}$, and $f >> f_{3dB}$?

2.4.8. Confirm your observed results with the results calculated from the transfer function; the phase shift, $\phi$, is defined as:
\[
\phi = \text{Arc tan} \left[ \frac{\text{Im} \left( \frac{V_{\text{out}}}{V_{\text{in}}} \right)}{\text{Re} \left( \frac{V_{\text{out}}}{V_{\text{in}}} \right)} \right]
\]  

(2.3.)

where \( \text{Im} \) and \( \text{Re} \) are the imaginary and real parts of the transfer function. Hence, calculate \( \phi \) for \( f = 0, f = f_{\text{3dB}}, f = \infty \).

### 2.5. High-pass Filter

![Figure 2.5. RC high-pass filter.](image)

Construct a high-pass filter with the components shown in figure 2.5. Again notice the similarity to the differentiator circuit of figure 2.2.; except for the component values the two circuits are identical. Measured its -3dB point by adjusting the frequency of the input sine wave until \( \frac{V_{\text{out}}}{V_{\text{in}}} \) reaches the correct ratio, i.e. -3dB. What is the limiting phase shift, both at very low frequencies and at very high frequencies?

**Write-up**

2.5.1. Calculate its -3dB point from first principals (see the previous exercise 2.4.2. and 2.4.3.) or by finding the proper equation in D&H or H&H.

2.5.2. Compare it to the measured -3dB point.

2.5.3. Calculate the limiting phase shifts, both at very low frequencies and at very high frequencies, using the transfer function of the high-pass filter (see page 35 of H&H or page 54 of D&H) and equation 2.3 of this manual.

2.5.4. What phase shifts did you observe at very low frequencies and at very high frequencies? Do they agree with the results from 2.5.3?

### 2.6. Filter Example Using a Composite Signal

![Figure 2.6. Generate a composite signal, consisting of two sine waves using a step-down transformer and a function generator.](image)

(Note: The symbol with the "two coils and two horizontal lines" represents a transformer. You
should not have to connect its center lead, the “center tap.”) The 1 k resistor protects the function generator in case the composite output is accidentally shorted to ground.

In this exercise you will use high- and low-pass filters to retrieve the individual frequency components of a composite signal. An application of this method is currently being used with Internet service providers and with some home networks which use regular power lines (instead of designated lines) to deliver the Internet content. The system is very similar to the one shown in figure 2.6, except that the function generator is replaced by the actual signal source delivering the information.

First produce the composite signal by combining the power line’s 60 Hz sine wave, which is supplied by a step-down transformer, with the function generator’s sine wave output, as shown in figure 2.6. To observe the signal better, set the function generator’s output voltage to its maximum value and its frequency to about 1 kHz.

Next, use a scope and try to observe the composite signal produced by the circuit in figure 2.6. Instead of seeing the two signals with the different frequencies neatly superimposed you will probably only see the lower frequency signal, the 60 Hz; in addition, this signal will be “blurry” and its trace will be very wide. Adjust your scope and try to understand why it must look that way on a scope. In doing so it may be helpful to increase or decrease the amplitude of the function generator’s output signal; also read carefully section 1.4.1. in this lab manual!

Now observe only the high frequency component: Set the frequency of the function generator well above the calculated -3dB point of the high-pass filter of figure 2.7. Attach this filter to the output of the signal circuit shown in Figure 2.6 and observe both the signal sent into the filter and the signal coming out of it.

Finally, replace the high-pass filter with the low-pass filter of figure 2.8. Again, keeping the frequency of the function generator well above the -3dB point of the low-pass filter observe both the composite signal sent into the filter and the resulting output.
Write-up

2.6.1. Show the signal observed on the scope from the output of the circuit in figure 2.6. Draw a picture (V vs. t) of a composite signal made by adding a 60 Hz and 1 kHz Sine wave. Assume the 60 Hz signal’s amplitude to be about 5 times larger than that of the function generator’s. Explain with some diagrams why the scope output looked the way it did.

2.6.2. Show the signal observed after the high-pass filtering and indicate the signal generator’s frequency.

2.6.3. Calculate the high pass filter’s attenuation of the 60 Hz component (no complex arithmetic necessary) by determining how many octaves 60 Hz is below the filter’s -3dB point.

2.6.4. Show the signal after the low-pass filtering.

2.7. Blocking Capacitor

- Figure 2.9. Blocking capacitor application using a (polarized) capacitor. Note: whether or not the capacitor used is polarized or not, has nothing to do with its (DC) blocking action! The reason for using a polarized capacitor is entirely based on its capacitance. Once capacitors reach a certain capacitance (about 1 uF and larger) it is difficult to manufacture them as non-polarized capacitors while maintaining a reasonable physical size. To achieve a large capacitance, manufacturers must use special insulators (typically oxidized films) but these can break down if a reversed polarity should be applied. Therefore, if you accidentally connect a polarized capacitor in reverse, it can act “unpredictably” and may burn out.

In section 1.4.6. you saw that periodic signals can be expressed by:

\[ V(t) = A + B\sin\left(\frac{2\pi t}{T} + \phi\right) \]

where the first term on the right is the DC part of the signal and the second term the AC part. If you were only interested in the DC aspect of the signal, you should by now be able to design a filter circuit that would remove all (or most of) the AC characteristics of the signal. On the other hand, many applications, particularly amplification of small time-varying signals, often require the removal of the DC offset from the signal.

In this exercise you will first learn how to remove (i.e. to "block") a signal’s DC offset by using blocking capacitors; in the second exercise, you will learn how to add a DC offset to an AC signal. Important: to properly observe the signals you must use the scope input coupling throughout this exercise in its DC-coupled mode! (If you do not understand what "DC-coupled" is, reread section 1.4. of this manual.)

Wire up the circuitry of figure 2.9. Drive it with a 1 kHz sine wave with a 1.5 Volt amplitude and a 5 VDC offset (use the DC OFFSET on the function generator.) (If you are having problems getting a 5 VDC offset it is probably because you entered the wrong amplitude and confused it with Vpp!) Observe both the input and output signal on the scope on channel 1 and 2. (Make sure that the ground reference for both channels has been set to the same place on the screen!) Adjust the DC offset on the function generator and observe the output. What type of filter is shown in figure 2.9
and what is the “frequency” of a DC signal? Hence, what part of the signal (AC or DC) should the filter pass?

Incidentally, the very same type of circuit is activated inside your scope (located between your probe and the scope amplifier) whenever it is set in AC coupled mode.

![Figure 2.10. AC signal riding on a DC offset](image)

So far you have used the DC offset on the function generator to produce an AC signal that was “riding” on top of a DC voltage. Now build your own circuit to do just that. It consists of a voltage divider to produce the proper DC voltage and a coupling capacitor to couple an AC signal onto it. Without any input signal applied at point C, what supply voltage must you apply at \( V_{cc} \) to produce \( V_d = 5 \) volts DC? Apply this voltage at \( V_{cc} \) and measure \( V_d \) to confirm your calculation. Turning the DC OFFSET OFF on the function generator, apply a 1 kHz, 1.5 Volt amplitude sine wave at C. Observe again \( V_c \) and \( V_d \). Does the output at \( V_d \) look identical to the output from the function generator in the first half of this exercise before you applied the filter in Figure 2.9? (If not, check your circuit.)

**Write-up**

2.7.1. What filter do you need if you want to see only the DC part of a signal? What if you only want to see the AC part?
2.7.2. Draw the output signal of the circuit in figure 2.9. when a 1 kHz, 1.5 V amplitude sine wave with a 5 VDC offset is applied to it.
2.7.3. Draw the output signal, \( V_d \), of the circuit in figure 2.10. when a 1 kHz, 1.5 V amplitude sine wave with no offset was applied at C. What value of \( V_{cc} \) was applied to the circuit in figure 2.10?
3.1. LC Resonant Circuit

Construct the LRC series circuit of figure 3.1. Drive the circuit with a sine wave and observe $V_A$ while adjusting the frequency from 100Hz up to 1 MHz. (Secondary effects such as the capacitance of the cables will produce secondary minima above 1 MHz.) Measure the frequency where $V_A$ is minimum.

Now see what happens at point $B$ when as $R$ approaches zero resistance. Try it by either removing $R$ or short-circuiting it. Drive the circuit again with a sine wave up to 1 MHz and observe $V_B$ while adjusting the frequency. Measure the frequency where $V_B$ is a maximum; also measure $V_B/V_P$.

Write-up

3.1.1. At what frequency did you observe $V_A$ to be minimum?
3.1.2. Now calculate this frequency: a) find the transfer function at A, i.e. find $V_A/V_P$; b) calculate the frequency for which the transfer function will be at a minimum.
3.1.3. Compare your measured value from 3.1.1. to the calculated one in 3.1.2.
3.1.4. At what frequency did you observe $V_B$ to be a maximum? (Make sure you did not forget to remove or short circuit $R_1$)
3.1.5. At the frequency reported in 3.1.4, what was $V_B/V_r$? In other words, did the circuit exhibit gain or attenuation?
3.1.6. Calculate the transfer function at B, i.e. find $V_B/V_r$ (assume $R=0$); from this find the frequency for which $V_B/V_r$ will be a maximum.
3.1.7. From your calculations in 3.1.6, what maximum should $V_B/V_r$ theoretically reach? Give two reasons why in reality it falls far short of that value.

### 3.2. The Diode

**Rules of Thumb about Diodes:**

If you were to replace a diode with a wire, in which direction would the current flow?
- if it flows in the direction the diode was pointing, then the diode acts like a short circuit
- if it flows in the opposite direction, then the diode acts like an open circuit.

![Diode Polarit y](image)

In this exercise you will measure the I-V characteristics of a diode. Because of its non-linear behavior you cannot apply a voltage source directly across the diode as you did with the resistor in section 1.1. Instead you will use an adjustable resistor to regulate the current that flows through the diode and then measure the resulting voltage across the diode and the current through the diode. (The adjustable resistor acts like an adjustable current source.)

Wire up the circuit in Figure 3.2. For $R_{adj}$ use a 10 turn, 50 kΩ variable resistor, i.e., "a pot" or "trimpot" (short for potentiometer) and connect its terminals as shown in figure 3.3:
Finally attach (digital) meters to measure $V_d$ and $I_d$. (If you forgot how to connect an ammeter, go back to section 1.1 and read the introduction!) In any case, NEVER connect your ammeter in parallel with the diode! Explain why this would be a bad idea.

If everything is connected correctly you should see some rather drastic changes in the current and voltage readings as you adjust the turn pot with a small screwdriver. Record these $V_d$ and $I_d$ readings.

Reverse the direction of the diode and record $V_d$ and $I_d$ again.

**Write-up**

3.2.1. Draw the schematic of Figure 3.2. and clearly indicate how and where you connected your volt and ammeter.

3.2.2. If you were to connect an ideal ammeter in parallel with the diode, what effect would the diode have on the circuit of Figure 3.2?

3.2.3. Plot $I$ vs. $V_d$ for the circuit in figure 3.2. with the diode forward biased (i.e. as shown in figure 3.2.)

3.2.4 Plot $I$ vs. $V_d$ with the diode reverse biased.

3.2.5. Comparing your diode $I_d$ vs. $V_d$ plot with the plots from exercise 1.1.1, how would you summarize the $I$ vs. $V$ behavior of a diode in the forward and reverse direction using only the concept of open and short circuit?

3.2.6. Explain what would happen if you were to put 5 volts directly across the diode. (If you have to try it, do so without the ammeter in the circuit!)
3.3. Half-Wave Rectifier

- Figure 3.4. Half-wave rectifier: "CT" stands for center tap and is a connection point on the secondary winding exactly halfway between the outputs to allow for output signals that are symmetrical with respect to it. It is not used in these exercises.

Construct a half-wave rectifier circuit with a 6.3 VAC transformer and a 1N914 diode (figure 3.2.)

Connect a 2.2 kΩ load, and look at the output on the scope.

Write-up

3.3.1. Draw the output of the half-wave rectifier and indicate the "size" of the observed signal.

3.3.2. The transformer output is rated 6.3 VAC (RMS); what $V_{pp}$ would you expect from it? Compare it to the value observed in 3.3.1. (Do not worry if the calculated $V_{pp}$ value does not agree very well with your measured value; the nominal 6.3 VAC (RMS) is for a worst-case-scenario, i.e. with a very large load attached; the actual VAC (RMS) is considerably larger for no load attached.)

3.3.3. Typical electrical household outlets in the USA are rated 110 volts rms. If you should ever be unlucky enough to touch the "hot" wire in your outlet, what would be the maximum voltage drop across your body and ground?

3.4. Full-Wave Bridge Rectifier

- Figure 3.5. Full-wave bridge

Now construct a full-wave bridge circuit (Fig 3.5). Be careful about polarities -- the band on the diode indicates cathode, as shown in the figure 3.2. Look at the output waveform (but don't attempt to look at the input -- the signal across the transformer's secondary -- with the scope's other channel at the same time; this would require connecting the second "ground" lead of the scope to one side of the secondary, which is impossible).

Write-up

3.4.1. Draw the output from the above circuit and label the size of the signal.
3.4.2. Draw the path of the current through the entire circuit if point A of the transformer’s secondary is more positive than point B. Show what happens to the current if A is more negative than B.

3.4.3. Look at the region of the output waveform that is near zero volts. Why are there flat regions? Measure their duration, and explain.

3.4.4. What would happen if you were to reverse any one of the four diodes (DON’T TRY IT!).

### 3.5. Ripple

Now connect a 15 \( \mu F \) filter capacitor across the output (IMPORTANT - observe polarity) and observe the output. Next put a 500 \( \mu F \) electrolyte capacitor across the output (again, be careful about polarity). This circuit is now a respectable voltage source, for loads requiring low current. To make a "power supply" of higher current capability, you would use heftier diodes (e.g. 1N4002) and a larger capacitor.

#### Write-up

3.5.1. Show the output with the 15 \( \mu F \) and the 500 \( \mu F \) capacitor across the output; on your drawing indicate the size of the DC voltage and the peak-to-peak "ripples", \( \Delta V \).

3.5.2. Calculate what the peak-to-peak ripples should have been for the two capacitors (see section 1.27 H&H or D&H 5-4), then compare it with what you measured in 3.5.1. Does it agree? (If not, have you assumed the wrong discharge time, by a factor of 2?) Note: you do not have to measure the current through the load resistor; you should be able to calculate it directly from your measurements in 3.5.1.

### 3.6. Three Terminal Fixed Regulator

Most electronic circuits use discrete active devices, such as operational amplifiers or digital logic gates. These devices require a clean, fixed DC supply voltage with +5V, +12V, +15V, -12V and -15V being the most common ones.

In the previous exercises you gained an understanding on how to build a DC power supply by rectifying and filtering an AC signal. In this section you will learn how to convert such a DC voltage into another, lower DC voltage that is very clean and stable by using a three terminal fixed voltage regulator. Note that with such devices you can only down-convert voltages, i.e., you can only go from a larger DC voltage to a smaller one. Nevertheless, these devices are extremely simple to use; they are efficient and they filter out most of the noise from the input as long as the input voltage remains sufficiently larger than the output voltage. These voltage regulators are available either with an adjustable output voltage or, like the one being used for this exercise, with a fixed output voltage.

[Diagram of 78L05 regulator with 4.7 \( \mu F \) and 0.01 \( \mu F \) capacitors, 220 ohm resistor]
Hook up the circuit in figure 3.6, which uses an LM78L05 voltage regulator; a description, with "specs" is given in H&H, Table 6.8, p. 342. (For our purposes, the characteristics of the 78L05 and the 7805 are identical.) Use the Wavetek Model 182A or Model 19 function generator and feed a 1 V amplitude, 120 Hz sine wave with a 10 VDC offset into the input. (Note: You must use the WaveTek function generators because the HP 33120A function generators cannot provide the required DC offset and amplitude range for this exercise.) Observe both the input and the output on the scope in its DC coupled mode. Now reduce the function generator's DC offset voltage until the output drops out of regulation. What is the "minimum input voltage" for which the output stays in regulation?

The dropout voltage is defined as the difference between the minimum input voltage (for which the output remains regulated) and the regulated output voltage. Measure it for the 78L05. An important measure of a voltage regulator's performance is its ability to reject ripple at the input when it is in regulation. Test this ripple rejection (the ratio of AC voltage in to AC voltage out) using the function generator with a 120 Hz sine wave (1 V amplitude) and a suitable DC offset to maintain the 78L05 in regulation. To observe the small ripple at the output use the scope into its AC coupled mode.

**Write-up**

3.6.1. Explain briefly what the voltage regulator does. Specifically, explain what the output is when the input voltage is larger than the "minimum input voltage" and what the output does when the input voltage is less than the "minimum input voltage."

3.6.2. What minimum voltage and what dropout voltage did you observe?


3.6.4. What ripple rejection ratio did you measure? Compare this with the specs from the manufacturer.

### 3.7. Resistance Measurement Techniques

**Exercise Overview**

The purpose of this lab is to use the digital voltmeter's built in resistance measurement feature to measure the resistance of a 2 foot piece of copper wire. Before you begin, a brief description of the pitfalls and strengths of resistance measurements is given.

**General Functionality of Resistance Meters**

In lab 1.1., you determined the resistance from the ratio of the measured voltage and current. Though this technique works well, it required a voltage source, two measurement instruments and a small amount of math. However, a simpler approach exists using the DVM's built-in resistance measurement feature. (On the HP or Agilent 34401A or 34410A DVMs it is enabled by pressing the "Ω 2W" or "Ω 4W" buttons.) This approach requires no additional test instruments and provides the user directly with the measured resistance value.

The built in resistance measurement feature works like this: the DVM sends out a predetermined, fixed DC current, $I_m$, and then measures corresponding voltage, $V_m$, across the device under test (DUT.) (See figure 3.7. below.)
Resistance Measurement Techniques

For example, applying a fixed 1 Amp current results in voltage drop across the device (in units of volts) that is identical to its resistance in Ohms. Similarly, a 1 mA current produces voltage drops corresponding to kOhms. A more sophisticated meter may actually calculate the ratio of voltage and current to display the resistance value:

$$R_m = \frac{V_m}{I_m} \quad (3.7.1)$$

The built-in resistance measurement feature is simple and works fine with isolated, passive devices (for example carbon resistors) whose resistance is not dependent on the applied current. For other devices, for example diodes, whose resistance directly depends on the current applied, using the built-in resistance measurement feature results in meaningless values as we are generally unaware of the actual value of the applied current. Furthermore, using the resistance measurement feature directly for devices that are part of a larger circuit can be outright dangerous as the device injects current into the entire circuit and it can potentially affect, or even damage, other components! Finally, most modern devices step through a range of applied currents (or voltages) to get the most accurate reading; in doing so they may exceed the current or voltage ratings of the DUT. Though in this exercise we will use the meter’s built-in resistance measurement feature, you should avoid it if you are not sure how it impacts your device; instead measure current and voltage separately as you have done in chapter 1.

2 Wire Resistance Measurements

Let’s measure the resistance of a passive device, labeled DUT below, using the most common built-in resistance measurement method. The setup shown below is known as a 2 wire (2W) resistance measurement or two point probe measurement. You may assume that the measurement devices themselves are ideal but that the measurement leads and test clips provide some resistance between the sample and the measurement device.

For example, applying a fixed 1 Amp current results in voltage drop across the device (in units of volts) that is identical to its resistance in Ohms. Similarly, a 1 mA current produces voltage drops corresponding to kOhms. A more sophisticated meter may actually calculate the ratio of voltage and current to display the resistance value:

$$R_m = \frac{V_m}{I_m} \quad (3.7.1)$$

The built-in resistance measurement feature is simple and works fine with isolated, passive devices (for example carbon resistors) whose resistance is not dependent on the applied current. For other devices, for example diodes, whose resistance directly depends on the current applied, using the built-in resistance measurement feature results in meaningless values as we are generally unaware of the actual value of the applied current. Furthermore, using the resistance measurement feature directly for devices that are part of a larger circuit can be outright dangerous as the device injects current into the entire circuit and it can potentially affect, or even damage, other components! Finally, most modern devices step through a range of applied currents (or voltages) to get the most accurate reading; in doing so they may exceed the current or voltage ratings of the DUT. Though in this exercise we will use the meter’s built-in resistance measurement feature, you should avoid it if you are not sure how it impacts your device; instead measure current and voltage separately as you have done in chapter 1.

2 Wire Resistance Measurements

Let’s measure the resistance of a passive device, labeled DUT below, using the most common built-in resistance measurement method. The setup shown below is known as a 2 wire (2W) resistance measurement or two point probe measurement. You may assume that the measurement devices themselves are ideal but that the measurement leads and test clips provide some resistance between the sample and the measurement device.
In this setup, the meter reads the total resistance between its inputs, namely:

$$R_{m2W} = \frac{V_m}{I_m} = 2 \times (R_{\text{Lead}} + R_C) + R_{\text{DUT}}.$$  \hspace{1cm} (3.7.2.)

where $R_{\text{Lead}}$ is the resistance of the test leads and $R_C$ is the contact resistance caused by the physical connection between the leads and the DUT.

The result of the measurement, equation 3.7.2., can be better understood if we consider two different cases:

a) $R_{\text{DUT}} \gg R_{\text{Lead}} + R_C$

b) $R_{\text{DUT}} \leq R_{\text{Lead}} + R_C$

In the first case:

$$R_{m2W} \approx R_{\text{DUT}}.$$  \hspace{1cm} (3.7.3.)

In this situation, the resistance of the leads and contacts does not significantly add to the measurement and their effect can be ignored. In this case, a standard 2 wire resistance measurement approach is sufficient and provides a straightforward way to measure the resistance using the DVM's built in functionality.

How do we know if condition a) has been met? Typical test leads of a few feet in length have a resistance of an Ohm or less; the contact resistance between non-oxidized metal surfaces, such as the solder leads attached to most electronic components, is usually also less than one Ohm. Therefore, as long as the DUT has a resistance of 10 Ohm or more condition a) is met. For example, the lab’s carbon resistors with their solder wire leads can be measured with a great accuracy using the standard 2 wire probe method and the DVM's built-in feature.

Measurements do get more involved in the second case, b), when $R_{\text{DUT}} \leq R_{\text{Lead}} + R_C$. In this case, the result of the 2 wire measurement is only meaningful if we know $R_{\text{Lead}}$ and $R_C$ so that their contributions can be subtracted from the resulting measurement. Situations like these arise when the DUT has a very low resistance (comparable or lower than the test leads) or when the contact resistance is large which occurs when the surface of the DUT oxidizes which is typical for semi conductors or reactive metal films. While $R_{\text{Lead}}$ can be determined, $R_C$ is generally very difficult to measure; each contact point is unique and, in addition, $R_C$ can be greatly affected by physical stress or strain. In conclusion, when $R_{\text{Lead}}$ and $R_C$ significantly affect our measurements (as shown in case b) the typical 2 wire resistance measurement technique is inadequate. Instead, you should use the 4-wire resistance measurement technique as explained in the next section.

**4-Wire Resistance Measurements**

The 4-wire (or 4-point probe, 4-terminal sensing, Kelvin sensing) measurement technique measures the DUT’s resistance independent of $R_{\text{Lead}}$ and $R_C$; it is the preferred method when $R_{\text{Lead}} + R_C \geq R_{\text{DUT}}$, i.e., case b).
Resistance Measurement Techniques

How does it work? In the drawing shown above, the outer two wires provide a path for the current $I_m$ to flow through the DUT; the inner pair of wires is used for the voltage sensing, $V_m$. Note that for an ideal voltmeter with $Z_{in} = \infty$ no current flows in the sensing wires once steady state has been reached. Therefore, there are no voltage drops through $R_{\text{LeadV}}$ and $R_{\text{CV}}$. $V_m$ represents the actual voltage drop across the DUT. In this case, $R_m$ is independent of the lead or contact resistance:

$$R_{\text{4W}} = \frac{V_m}{I_m} = R_{\text{DUT}}$$  \hspace{1cm} (3.7.4.)

Though you can setup a 4-wire measurement from scratch using the individual components (you already did so in lab 1.1., see Figure 1.2.c.) most (sophisticated) DVMs come with this feature built-in simplifying the measurement process. (For details on how to use this feature on the Agilent or HP DVMs see the exercises at the end of this section.) The only (minor) drawback of the 4-wire resistance measurement technique over the 2 wire method is that it requires more hookup wires.

Note one final detail of the 4-wire measurement process: observe in the diagram above that the physical separation between the current leads (point X) and the corresponding voltage sensing leads (point Y) has no effect on the measurements of $R_{\text{4W}}$. In practice, the distance between points X and Y is often minimized by using special Kelvin clips. Outwardly, these resemble alligator clips except that each "jaw" is electrically isolated from the other and while one jaw is used for current the other is used for voltage sensing.

---

*Figure 3.9. 4-Wire resistance measurement setup.*

*Figure 3.10. 4 point probe measurement using Kelvin clips.*
3.7.1. In this exercise, you will measure the resistance of a 2 feet long 22 AWG (American Wire Gauge) 2 copper wire using the Agilent or HP built-in resistance meter feature. Given that the leads of the test wires are of the same material and approximately the same cross-sectional area as the test wire, explain what you expect to measure when you use the 2 wire and the 4-wire technique. Which method will give you a more accurate measurement of the resistance of the wire?

3.7.2. Now use the HP or Agilent 34401A or 34410A digital voltmeter. Set it in its 2 wire resistance measurement mode by pressing the $\Omega_2W$ button and by connecting the test leads to the top rightmost input jacks, labeled "Input." Attempt to measure the resistance of the approximately 2 ft long 22 AWG copper wire. (If the measurement is unstable or produces a value in the kOhms then use some sandpaper and remove the insulation and oxidation at the end of copper wire where the test leads are connected.)

3.7.3. Repeat the resistance measurement of the 2 ft long 22 AWG copper wire. Again, use the HP or Agilent 34401A or 34410A digital voltmeter. Set it in its 4-wire resistance measurement mode by pressing the Shift key and the $\Omega_4W$ button; the upper right hand corner of the display window should show a 4W (indicating a 4-wire measurement, not a 4 Watt measurement).

- First, connect the pair of current injecting test leads: plug one end of the leads into the leftmost input jacks, labeled "$\Omega_4W$"; connect the end with the mini grabber clips to each end of the 22 AWG copper wire.
- Next, connect the pair of voltage sensing test leads: plug one end of the leads into the rightmost jacks on the voltmeter, labeled "Input;" connect the clips at the end of the test leads directly adjacent and to the inside of the current injecting clips, as shown in figure 3.10.

3.7.4. Measure the length of the 22 AWG copper wire (between the sensing contacts) with a ruler and calculate the resistance / ft of the 22 AWG copper wire for the 2-wire and 4-wire probe measurement. How does it compare with the nominal 16.14 mOhms/ft as reported by Wikipedia? [http://en.wikipedia.org/wiki/American_wire_gauge](http://en.wikipedia.org/wiki/American_wire_gauge)
Write-up Format: Short

Reading: Lab Manual: Appendix E
Lab Manual: Appendix H
E4E: Sections 3.1-3; also, first paragraph: Golden Rules are in 3.2 (page 34)
Horowitz&Hill: Sections 4.01 - 4.06 and Section 4.19

Diefenderfer&Holton: Sections 9-1 - 9-6, 9-9 (pages 183 to 194, pages 197 to 198.)
Plonus: Sections 6.1 – 6.3 (pages 203 to 209)

Remember the op-amp "golden-rules" on page 185 of D&H or page 177 of H&H but be forewarned: a lot of confusion about op-amp circuits comes from blindly applying the "golden rules" to all op-amp circuits. It cannot be emphasized enough that the golden rules are valid only under certain specific circumstances and applying them to the wrong op-amp circuits enhances misconceptions about op-amps! Therefore, use the golden rules only when the following two conditions are met:

i) the op-amp circuit uses negative feedback;

ii) the op-amp output voltage is not close to its limiting output voltages, i.e., the op-amp is not “saturated.”

Note: Some of the op-amp circuits you will be building have large gains and high switching speeds and they may exhibit unwanted oscillations. Signals picked up directly on the wires connected to the inputs of the circuit, or power line variations from large switching currents are the most probable culprits. A cure for the former would be to shorten the input and output leads, making sure they are kept as far apart as possible. For the latter, connect decoupling capacitors between each of the power lines and ground, as close to the chip as possible. This is a good practice whenever an integrated circuit is used. 15 μF Tantalum capacitors (or / and 0.01 μF disc capacitors) are usually a good choice and are often the first thing to try when dealing with a “flaky” circuit!
4.1. Power Supply, Ground and Common

These exercises will give you a chance to review your knowledge of DC voltage dividers. In addition, they will expose you to the finicky details of ground and commons. Be sure you have read Appendix E before you start.

Shown below are four voltage divider circuits. They are all identical except for the way common and ground are connected. Build the circuit using an 8.2kΩ resistor in series with an 800 Ω resistor to obtain the nominal 9kΩ. (The 9kΩ and 1kΩ were chosen to simplify the math; recall that a 10% accuracy is all that we strive for in our measurements and calculations.)

Power the circuit with an Agilent E3630A power supply. You will need the adjustable +20V and -20V outputs to produce the two 5V sources. The voltage of the -20V output is related to the +20V by:

\[ V_{-20V} = -k \cdot V_{+20V} \]

where "k" is the "Tracking Ratio."

The +20 V is adjusted by the ±20V knob and the "tracking ratio" should be set to 1 for the rest of the exercises (including op-amp circuits), i.e., all the way clockwise.

You must use an oscilloscope to measure the DC voltages indicated below. (Make sure the scope’s input coupling mode is set to DC mode and not AC!) For the circuits A and B, connect the scope ground leads to point X, i.e., at \( V_X \). For both circuits A and B measure with the scope: \( V_{AX} \), \( V_{BX} \). Confirm your results by adding \( V_{AX} + V_{BX} \). What must it add up to? (If you are not sure, ask your TA.) In each case, calculate what the common voltage (COM) is with respect to ground.
For circuits C and D, measure $V_A$, $V_X$, $V_B$ with respect to ground. Note how that this time the scope grounds are connected to the power supply’s ground. Confirm your results by calculating $V_{AB}$ for each case.

**Write-up**

4.1.1. For circuit A, what voltages $V_{AX}$, $V_{BX}$ were measured?
4.1.2. For circuit A, what are the calculated values $V_{AX}$, $V_{BX}$ and the common voltage (COM) with respect to ground when point X is grounded?
4.1.3. For circuit B, what voltages $V_{AX}$, $V_{BX}$ were measured?
4.1.4. For circuit B, what are the calculated values $V_{AX}$, $V_{BX}$ and the common voltage (COM) with respect to ground when point X is grounded?
4.1.5. For circuit C, what voltages $V_A$, $V_X$, and $V_B$ were measured with respect to ground?
4.1.6. For circuit C, what are the calculated values for $V_A$, $V_X$, $V_B$ and the common voltage (COM) with respect to ground?
4.1.7. For circuit D, what voltages $V_A$, $V_X$, and $V_B$ were measured with respect to ground?
4.1.8. For circuit D, what are the calculated values for $V_A$, $V_X$, $V_B$ and the common voltage (COM) with respect to ground?

**4.2. Inverting Amplifier**

Construct the inverting amplifier in figure 4.3. In these labs, use ±15 volt supplies for all op-amp circuits unless explicitly told otherwise and, if necessary, connect 15 μF Tantalum capacitors to the supply pins as explained in the introduction.
Drive it with a $1 \, V_{pp}$ 1 kHz sine wave. What is the gain? Now measure the gain for a $5 \, V_{pp}$ 1 kHz sine wave. What is the maximum possible output voltage from the circuit? How about the linearity (try a triangle wave)? Try sine waves of different frequencies. At about what maximum frequency does the amplifier stop working well?

An important property of an op-amp is its ultimate "speed," or rate at which its output can change. This is called "slew rate" and is usually measured in units of $V/\mu$sec. Measure the slew rate of the 741 op-amp by driving the input with a square wave and measuring the slope of the falling or rising edges. Compare your result with the manufacturer's specifications of 0.5 $V/\mu$sec. Demonstrate the limiting nature of the slew rate by varying the output signal amplitude over a wide range. (Some op-amps exist with slew rates approaching 1000 $V/\mu$sec). For comparison, measure the slew rate of a 411 op-amp. (The 741 and the 411 have the same pin out so it is easy to exchange them.)

Now go back to 1 kHz sine waves. Measure the input impedance of the amplifier circuit by adding 1 k resistor in series with the input. (For this part you may use either the 741 or the 411 op-amp.)

Measuring the output impedance is a bit tricky: because it is (allegedly) very low one is tempted to use a very small load resistor, i.e., a large load, to better observe the effect of the load. Unfortunately, a typical op-amp is current limited and cannot supply large currents and when connected to a large load its output will quickly deteriorate as it desperately tries to keep up. (Special high current op-amps do exist though.) Therefore, use a very small signal, a load resistor of least 50 $\Omega$ and if that still doesn’t help attach the Tantalum decoupling capacitors as specified in the introduction.

**Write-up**

4.2.1. Calculate the gain of the circuit.
4.2.2. What was measured gain for $1 \, V_{pp}$ 1 kHz sine wave?
4.2.3. What was measured gain for $5 \, V_{pp}$ 1 kHz sine wave?
4.2.4. Which one is the "correct" gain, the one measured in 4.2.2 or 4.2.3? Explain why the gain is different for the two measurements. (Hint: read the next question.)
4.2.5. What was the maximum possible output voltage observed?
4.2.6. How linear is the output, especially as the output reaches its maximum values?
4.2.7. At about what maximum frequency does the amplifier stop working well?
4.2.8. What slew rate did you measure for the 741 and the 411?
4.2.9. Since the inverting input, $v_-$, is a virtual ground, what would be this circuit’s input impedance? What output impedance do you expect? (See D&H page 187 or H&H page 178.)
4.2.10. What were the measured input and output impedances?

**4.3. Non-Inverting Amplifier**

![Non-inverting amplifier diagram](Image)
Wire up the non-inverting amplifier in Figure 4.4. using a 741 (bipolar) op-amp. What is the voltage gain? This circuit has a very high input impedance which is equivalent to the actual op-amp’s input impedance. Measure it by using a 1 kHz sine wave and by putting a large resistor such as a 560k\(\Omega\) in series with the input. Do not use the scope to measure the small voltage drop across the 560 k\(\Omega\) resistor; instead use a DVM - it has a much higher input resistance.

If you would like to increase the input impedance further, then you should replace this bipolar op-amp with a JFet type op-amp, like the LF411. Check out the manufacturer’s input impedance specification for the LF411 at http://cache.national.com/ds/LF/LF411.pdf and see if you can measure such an input impedance with the current setup and equipment. Explain what limits you in these measurements.

(Note: a far better and more accurate way to measure the input impedance of a circuit that has such a large value is to measure the change in the output voltage before and after the 560k\(\Omega\) resistor has been connected. By accounting for the closed loop gain of the circuit, you should be able to calculate the voltage drop across the 560k\(\Omega\) resistor and, from this, finally, the input impedance of the circuit. Optionally, if you have time, test this method and see if it works for the LF411 and yields the results stated in the data sheets.)

**Write-up**

4.3.1. Calculate the voltage gain of the circuit in figure 4.4.
4.3.2. What was the measured voltage gain?
4.3.4. What input impedance did you measure for the 741?
4.3.5. What input impedance would you get using an LF411 op-amp? What series resistor at the input would you need to measure a 50% reduction in output? Even if you had such a resistor, what other factors would seriously affect or hamper your measurements?
4.3.6. In terms of input impedance, how does this circuit differ from the one in figure 4.3.?

4.4. **Follower**

![Follower Diagram](PDF)

*Figure 4.5. If at higher frequencies you get a noisy output, attach the 15uF Tantalum capacitors as described in the introduction of this chapter.*

Build the follower shown in figure 4.5 using an LF411 JFet op-amp. Use a sine wave input and measure its voltage gain for signals below 1 MHz. What is the phase difference between the input and output at signals below 1 MHz?

The LF411 is a “compensated” op-amp, meaning that its transfer function will resemble the behavior of the familiar RC low pass filter with a distinct –3dB point frequency and the 6dB/octave roll-off in gain for frequencies above that. This low-pass filter was built into the op-amp by the manufacturer to keep the op-amp stable even at high frequencies. Observe its low pass filter behavior by increasing the frequency of the signal. At what frequency does the gain begin to diminish, i.e. at what frequency is \(V_{\text{out}}\) only 70% of its low frequency value? What happens to the phase of the output signal vs. the input signal as you approach that frequency?
Write-up

4.4.1. Calculate the voltage gain for the circuit in Figure 4.5.
4.4.2. What voltage gain did you observe for signals below 1MHz? What was the phase difference?
4.4.3. At what frequency was the gain begin to diminish, i.e. at what frequency is \( V_{\text{out}} \) only 70% of its low frequency value?
4.4.4. The frequency at which an op-amp no longer is able to produce unity gain is called the (unity) gain bandwidth product, GBW. How does the result from 4.4.3. compare with the manufacturer's gain bandwidth product? (See http://cache.national.com/ds/LF/LF411.pdf.)
4.4.5. What was the measured phase difference between the input and output signal at the -3dB point? How does that compare with your typical low pass filter? What would happen to the stability of the op-amp if it was not compensated and the phase difference between input and output (and therefore in the negative feedback loop) were to approach to 180 degrees?
4.4.6. What \( Z_i \) and \( Z_{\text{out}} \) do you expect from the follower circuit? (Hint: compare it to the circuit in Figure 4.4.)
4.4.7. Since the gain of this circuit is 1, why not simply replace the entire circuit with a wire?

4.5. Integrator

Construct the active integrator in figure 4.6. using a LF411 op-amp. Drive it with a 2 \( V_{\text{pp}} \) 500 Hz square wave. This circuit is sensitive to small DC offsets of the input wave form (its gain at DC is about 50); if the output appears to saturate near the +/-15 Volt supplies, you may have to adjust the function generator's OFFSET control. Measure the peak-to-peak triangle wave amplitude at the output.

What is the function of the 4.7 M\( \Omega \) resistor? What would happen if you were to remove it? Try it; observe the output with the scope in its DC coupled mode.

Write-up

4.5.1. From the component values and the circuit find an equation that describes the output as a function of time; in your calculations, ignore the effect of the 4.7 M\( \Omega \) resistor.
4.5.2. From 4.5.1 calculate the peak-to-peak triangle wave amplitude at the output that should result from a 2 \( V_{\text{pp}} \) 500 Hz square wave input.
4.5.3. What was the measured peak-to-peak triangle wave amplitude at the output? (If you are off by a factor of two, think carefully how long it takes to go from peak to peak?)

4.5.4. Explain the function of the 4.7 MΩ resistor.
5.1. Summing Amplifier – D to A Converter

The circuit in figure 5.1 is a digital-to-analog converter, a weighted version of figure 9.12 of D&H, page 192 (or H&H, figure 4.19 page 185). The 1-2-4-8 inputs are either +5 V or 0 V. The circuit produces an output voltage that is proportional to the value of the digital number at the input. Circuits of this type are used in digital to analog converters (DAC) where the input is a digital value and the output an analog voltage. The table below lists some binary and some digital equivalents for a few input voltages.

<table>
<thead>
<tr>
<th>Voltage input8</th>
<th>Voltage input4</th>
<th>Voltage input2</th>
<th>Voltage input1</th>
<th>Binary Value</th>
<th>Digital Input Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0000</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0001</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0010</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>5</td>
<td>5</td>
<td>0011</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0100</td>
<td>4</td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0101</td>
<td>5</td>
</tr>
</tbody>
</table>

*Table 5.1.*

The input voltages listed here are +5 V and 0 V. Once we start "really" working with digital circuits, we will refer to them simply as HI or LO.
Figure 5.1. Weighted summing amplifier. There are no 1.6 kΩ resistors in the lab but you can easily obtain such a value using 2 resistors.

Build it and measure the output voltage for the following digital input values: 0, 1, 5, 8, 12 and 15.

Write-up

5.1.1. Calculate the conversion sensitivity of the summing amplifier (i.e., how many volts/digital value).
5.1.2. What additional circuitry is needed in order to get a positive polarity output voltage?
5.1.3. Plot your measured output voltages vs. the digital input values.
5.1.4. How does the slope of your plot compare to the value calculated in 5.1.1.?
5.1.5. To “improve” the linearity, the circuit in Figure 5.1. should really use 8.0k and 4.0k resistors instead of the 8.2k and 3.9k. (Unfortunately these resistor values are harder to come by.) Nevertheless, even for the resistors shown above, your data should still indicate a very linear trend between the input values and the output voltages. Explain why this is the case. In other words, in terms of linearity, how should you select the accuracy of the 4 input resistors? Would this circuit improve if the smaller resistors (1k and 2k) were replaced with high precision resistors rated at an accuracy of 0.1% or should the large resistors (ideally 8.0k and 4.0) be made more accurate? The current lab resistors are only accurate within 5%. Explain your answer.

5.2. Current-to-Voltage Converter

Figure 5.2. Photo transistor photometer circuit. The labels B, E and C on the transistor are traditionally used to identify the transistor’s base, emitter and collector, respectively. For the circuit shown in Figure 5.2., the base is left unconnected.

Connect a GE L14P2 phototransistor as shown in the circuit of figure 5.2. Look at the op-amp output signal. (If the DC level is more than 10 volts, reduce the feedback resistor). Measure the percentage “modulation”, i.e. the ratio of the AC peak-to-peak signal vs. the DC offset. This will be quite large if the laboratory has fluorescent lights. (It would be much smaller in a room with incandescent light bulbs. Why?)
Standard fluorescent light sources emit light in 120 Hz bursts. The light fixtures in the lab have been converted to energy saving devices, which emit light bursts at a different rate; measure that frequency. (If you are unable to observe a "good" signal in the 10's of kHz range, roll a piece of paper into a tube. Position it over the phototransistor and point it towards the nearest fluorescent light source.)

Write-up

5.2.1. What was the measured percentage "modulation"?
5.2.2. What is the frequency of the light bursts for the fluorescent light fixtures in the lab?
5.2.3. From your measurements calculate the average input photo current to the op-amp.
5.2.4. Explain why standard fluorescent light sources emit light in 120 Hz bursts though they are powered by the typical 60 Hz (main) power circuit.

5.3. Comparator and Schmitt Trigger

(Before attempting this and the following exercise, you may want to review Appendix H.)

Op-amp circuits for comparators use either no feedback or positive feedback. Therefore, the "golden rule of op-amps" stating that $V^+ \approx V^-$ cannot be applied for these circuits! As a matter of fact, for an arbitrary input voltage $V_{in}$, $V^+$ and $V^-$ will under most circumstances not be similar at all!

To analyze these circuits one needs to apply the basic principle of op-amps which states:

$$V_{EE} < V_{out} < V_{CC} \quad \quad V_{out} = A (V^+ - V^-) = A \Delta V$$

(5.1.)

where $A$ corresponds to the open loop gain which for an ideal op-amp is infinity. From this equation it can be seen that $V_{out}$ can take on only two values: positive or negative infinity.

For a real op-amp though $V_{out}$ can not exceed its supply voltages and its output will "saturate" close to the supply voltages, $V_{CC}$ or $V_{EE}$. As can be seen from equation 5.1., depending on whether $\Delta V$ is positive or negative, $V_{out}$ will approach either $V_{EE}$ or $V_{CC}$. Except during the time when the output is switching from $V_{CC}$ to $V_{EE}$, or vice versa, its output will remain in one of these two states and, therefore, it is called a "bistable" circuit.

This property is very useful for digital circuits which are only concerned with (binary) states, i.e., either something is ON or it is OFF. Comparators are often used as the first step in converting analog signals to digital ones. A typical application would be in a counter circuit where (analog) pulses from a detector are converted into digital pulses to be counted by a computer. Comparators are also widely used in data transmission where an analog signal or a messy digital one needs to be "cleaned up." In that case, a positive feedback loop is added to the comparator to provide hysteresis to reject noise. You will build such a circuit later in this exercise.
Comparator and Schmitt Trigger

- Figure 5.3. A simple comparator circuit using an external power supply to set the threshold voltage, $V_{Th}$. If you do not have an external power supply you could use a trim pot instead, connected between +12V and -12V to generate such a threshold voltage.

Use a 411 op-amp and build Circuit A in figure 5.3. Connect $V_{Th}$ to the 0 to 6V adjustable output on your power supply but for now set $V_{Th} \approx 0$ V. Drive $V_{in}$ with a 10 Vpp sine wave at 1 kHz and observe the "square wave" output. Is the square wave in phase with the input signal or is it 180 degrees out of phase? Does what you observe agree with equation 5.1.?

Use the previous settings for the power supply and function generator and connect it to Circuit B in Figure 5.3. Observe the output and the phase relationship with respect to the input. Does what you observe make sense, i.e., does it, agree with equation 5.1.?

Once more, go back to Circuit A in Figure 5.3. and study now the effect $V_{Th}$ has on the output. Observe $V_{in}$ and $V_{Th}$ with scope channels 1 and 2, both set to identical (vertical) gain; connect a 10x scope probe from channel 3 to $V_{out}$. (The 10x probe is a 1:10 voltage divider and, hence, reduces the signal by a factor of 10 so that it will not exceed the input range of channel 3.) Slowly adjust $V_{Th}$ from 0 V to 6 V and see how it affects $V_{out}$. Make a drawing of the three signals as seen on the scope; on the graph clearly indicate $\Delta V$ and how it relates to the output. Explain what happens to the output when $V_{Th} > 5$ V? Why is that the case?

Finally, let’s see how noise affects the comparator circuit. Build the following noise simulation circuit by adding a second function generator to your bench and connecting them together as in Figure 5.4.

The first function generator will be our signal source, providing us with $V_{Signal}$. Set it to a 200 Hz, 10 Vpp sine wave. The other function generator, producing a 20 kHz sine wave, will be our noise source, $V_{Noise}$. To start with, set $V_{Noise}$ to its minimum amplitude, 0.1 Vpp, to affect our combined signal minimally. Connect the output $V_x$ to $V_{in}$ of Circuit C in figure 5.5. (This is essentially the same
Op-Amp Applications 1 / 5.3. Comparator and Schmitt Trigger

5.3. Comparator and Schmitt Trigger

Circuit as Circuit B in Figure 5.3. with $V_{th}$ permanently set to 0, i.e., grounded.) On the scope observe $V_x$ and $V_{out}$. You should see the familiar (inverted) square wave with one cycle of the sine wave corresponding to one cycle of the square wave. Now increase the noise amplitude of $V_{noise}$ from 0.1 Vpp up to 4 Vpp. Note the multiple transitions in the output as you increase the amplitude. You now have multiple square waves occurring during one cycle of $V_{signal}$. Draw a picture and indicate what causes these additional transitions. How would they affect a pulse counting experiment?

![Circuit C and Circuit D](image)

- Figure 5.5. Comparator without and with Hysteresis. The circuit on the right is also known as a Schmitt trigger.

Convert circuit C into Circuit D by adding the positive feedback loop. Note how this circuit resembles the standard inverting amplifier in Figure 4.3. How does it differ from it? Again turn on both function generators as specified above and increase the $V_{noise}$ from 0.1 Vpp up 4 V. How is the output affected? To check that the noise is still there briefly disconnect the 10k between $V+$ and the output and note how you’re back to the noisy output signal again! Make a drawing of $V_x$ and $V_{out}$ with the 10k connected.

Circuit D in figure 5.5 has hysteresis, i.e., it has memory. In other words, once it has been “activated” to change its output, it takes a larger signal to change it back to its previous state. This type of circuit is usually referred to as a Schmitt trigger in honor of Otto H. Schmitt who invented it in 1934 here at the University of Minnesota while still a graduate student.

Reconnect the 10k as shown in Circuit D and increase $V_{noise}$ past the 4Vpp. At what voltage do you see multiple transitions appearing again?

To observe the hysteresis in a different way, minimize $V_{noise}$ to 0.1 Vpp; connect $V_x$ to scope channel 1 and $V_{out}$ to channel 2. On the scope, select the X−Y mode and turn off channel 1. What you (should) see now is a direct display of $V_x$ along the X-axis and $V_{out}$ along the Y-axis. Center the image by setting channel 1 temporarily to GND and using the timing position knob to center the image horizontally. Display channel 1 again. Now center channel 2 by using the up / down adjustment after it was grounded. Draw the image observed carefully in your notebook. Can you see the faint vertical transitions? How large is the hysteresis?

To indicate the Schmitt Trigger circuitry, a hysteresis symbol is usually drawn on top of a schematic symbol to indicate that its input stage employs some sort of Schmitt trigger.

Write-up

5.3.1. Using $V_{th} = 0$, what is the phase relationship between input and output voltage for Circuit A in figure 5.3. How does it differ from Circuit B? For the two circuits, draw $V_{in}$, $\Delta V$ and $V_{out}$ vs. time. Explain using equation 5.1. why it must behave that way. For each circuit, draw the corresponding $V_{out}$ (y-axis) vs. $V_{in}$ (x-axis) diagram.

5.3.2. How does adjusting $V_{th}$ from 0 V to 6 V affect $V_{out}$ in Circuit A? Choose an arbitrary $V_{th}$ of less than 5 V and make a drawing of $V_{in}$, $V_{th}$ and $V_{out}$ vs. t as seen on the scope; on the
5.3.3. What will happen to \( V_x \) in the circuit in Figure 5.4. if the 1k resistors were omitted and the function generators were connected together directly. (Please, do not try it!)

5.3.4. Draw \( V_x \) vs. t and \( V_{out} \) vs. t for \( V_{\text{Noise}} \) small (0.1 Vpp) and \( V_{\text{Noise}} \) large (4 Vpp) when the output \( V_x \) of Figure 5.4. has been connected to \( V_{in} \) of Circuit C in Figure 5.5.

5.3.5. Draw \( V_x \) vs. t and \( V_{out} \) vs. t for \( V_{\text{Noise}} \) small (0.1 Vpp) and \( V_{\text{Noise}} \) large but less than 4 Vpp when the output \( V_x \) has been connected to \( V_{in} \) of Circuit D.

5.3.6. With \( V_x \) connected to \( V_{in} \) of the circuit in Figure D, as you increase \( V_{\text{Noise}} \), for what voltage range does it have no effect on the output? At what voltage \( V_{\text{Noise}} \) do you see it affecting the output again, i.e., causing multiple transitions?

5.3.7. For the circuit in Figure D, draw \( V_{out} \) (y-axis) vs. \( V_{in} \) (x-axis) as observed in the XY-mode on the scope. Clearly label the scales and indicate the hysteresis in the input. How many volts was it?

5.3.8. For the circuit in Figure D, calculate the hysteresis. (Hint: calculate at what voltage, \( V_{in} \), the output “switches” either from \( V_{CC} \) to \( V_{EE} \), or vice versa and recall that the output only switches when \( \Delta V \) changes sign, i.e., at \( \Delta V = 0 \).)

5.4. Op-amp Relaxation Oscillator

![Op-amp oscillator diagram]

*Figure 5.4. Op-amp oscillator.*

Hook up the circuit of Figure 5.4 using 10 kΩ resistors and a .01 µF capacitor. (See D&H, section 10-4, page 224 or H&H, p. 285.) Look at the two inputs to the op-amp on the scope and see how it behaves as a comparator. Measure the frequency of the oscillator at the output of the op-amp. How does this frequency change if the feedback resistor, \( R_F \), is increased to 100 kΩ?

**Write-up**

5.4.1. Show that the period of oscillation for the op-amp oscillator is:

\[
T = 2R_C C \ln\left(\frac{1 + \beta}{1 - \beta}\right) \quad (5.2)
\]

where: \( \beta = \frac{R_2}{R_1 + R_2} \) \quad (5.3)

(Here are some comments on how the op-amp oscillator circuit works. The op-amp is used as a comparator with its non-inverting input determining its threshold voltage. The output, as is the case in a typical comparator circuit, will either be at its positive or negative maximum value, i.e. \( \pm V_{out} \).)
This determines the two threshold voltages at the non-inverting input; calculate them in terms of \( R_1 \) and \( R_2 \). To illustrate the operation of the oscillator, assume that initially \( V_{\text{out}} \) is positive and that the capacitor is not charged i.e. \( V_- = 0 \). As time goes on the capacitor charges up and \( V_- \) begins to increase. When \( V_- \) reaches (or exceeds) the positive threshold voltage, \( V_{\text{out}} \) will flip to its negative value; correspondingly, the threshold voltage will also change. The capacitor now discharges through \( R_F \) and \( V_- \) will decrease. When \( V_- \) reaches or exceeds the negative threshold voltage, the output will flip again to its positive value and thereby starts the cycle over again. You should be able to see that the time it takes for \( V_- \) to go from one threshold voltage to the other corresponds to half a period. Calculate it.

5.4.2. What frequency did you measure at the output of the op-amp with \( R_1 = R_2 = R_F = 10 \, \text{k}\Omega \)?

5.4.3. How does this measured frequency compare to what you expect (use equations 5.2. and 5.3.)?

5.4.4. What frequency did you measure with \( R_F = 100 \, \text{k}\Omega \) and \( R_1 = R_2 = 10 \, \text{k}\Omega \) and how does it compare with what you expect?
6.1. Difference Amplifier

The circuit in Figure 6.1 is a difference amplifier also sometimes called a differential amplifier. Unlike our previous amplifier circuits which were only able to amplify a signal with respect to ground, this circuit amplifies signals between any two (arbitrary) points \( V_x \) and \( V_y \). (Of course, the input voltages have to be within the input voltage range of the op-amp which is typically between \( V_{CC} \) and \( V_{EE} \! \).

Being able to amplify a differential voltage is extremely useful in many experimental applications. For example, when a common ground point is difficult to maintain or in long transmission lines it can be used to reject (60 Hz) noise pickup.
Figure 6.2. The detector is located at a large distance from the amplifier. In this case the connecting wires will act like antennas and will pick up (primarily) 60 Hz noise. A difference amplifier will reject signals which are identical on both of its inputs, i.e., it rejects common noise. Instead, it amplifies small difference signals, like the one originating at the detector.

Another application involves blocking the DC bias (or DC offset) that some transducers require to work properly. Capacitive detectors, such as the old style microphones are a good example. In such devices a large (50V to 100V) DC bias voltage has to be applied to the capacitor plates to pick up a signal such as voice or music. The signal itself will be small, much less than a volt and it will ride on top of the large bias voltage. To extract the signal, we can block the DC component with a blocking capacitor as was shown in chapter 3. This approach only works if the signal of interest is at a high frequency, i.e., much higher than the -3dB point of our filter, and if we are not worried about filtering out some of its low frequency components. But if the signal is very slowly changing (as in our next exercise) then we need to use a difference amplifier to amplify only the signal and not its DC bias.

How does the difference amplifier in Figure 6.1. work? To analyze it, use the superposition principle: first ground one input, for example, V_y, and find the transfer function for V_out( V_x, V_y=0); next ground the other input and find its transfer function, i.e., V_out( V_x=0, V_y); finally, add the two transfer functions. (Hints: by grounding V_y, the circuit essentially reduces to the familiar inverting amplifier circuit. When you ground V_x, and temporarily ignore the voltage divider formed by R_1 and R_2 in front of the non-inverting input, you should see that a signal applied directly into the non-inverting input of the op-amp results in the familiar non-inverting circuit; therefore, applying the signal to V_y results first in a (slight) attenuation due to the R_1 and R_2 voltage divider before it is then amplified by the non-inverting circuit.) So what is V_out( V_x, V_y)?

Now build the circuit in figure 6.1; use a 411 op-amp with R_1 = 1k, R_2 = 100k. Measure its gain while grounding one input, for example V_y, and feeding a 100 Hz, 0.1 V_pp sine wave into the other input, V_x. Now reverse the inputs, i.e., ground V_x and feed the input signal into V_y. Since this type of measurement determines the gain of a “difference signal” applied between V_x and V_y, it is called its differential gain measurement, G_{Diff}:

\[ G_{Diff} = \frac{V_{out}}{(V_y - V_x)} \]

Ideally, the results for the two G_{Diff} measurements you just carried out should be identical. Can you explain why that should be so?
6.1. Difference Amplifier

The common mode gain, \( G_{\text{Com}} \), is the amplification obtained when \( V_x = V_y \). Ideally this should be 0 to reject (pick-up) noise that is common to the two inputs completely. Unfortunately, a very low common mode gain can only be achieved if both resistors \( R_1 \) and \( R_2 \) are exactly matched. Since the resistors in the lab are only matched to within 5%, a common mode gain larger than 0 will be observed, though it should be much less than unity! Measure this common mode gain by shorting \( V_x \) and \( V_y \) together as shown in the figure 6.3. You may have to increase the amplitude of the input signal substantially until you are able to see a signal in the output.

The ability of such a circuit to reject common mode while providing for a good differential gain is a very important property. It is mathematically expressed as the Common Mode Rejection Ratio (CMRR) and is defined as:

\[
\text{CMRR} = \frac{G_{\text{Diff}}}{G_{\text{Com}}}
\]

Usually it is expressed in Decibels. What CMRR (in dBs) did you measure for your circuit?

Before proceeding to the next section where we will amplify a very small differential signal, here are some additional comments regarding the differential amplifier. In addition to its modest CMRR due to the poorly matched resistors, the circuit suffers from another major draw back: its input impedance is "only" \( 2R_1 \). (If we want a very large differential gain, for practical reasons \( R_1 \) could become quite low.) Theoretically, we could improve these factors by using higher precision resistors and by preceding each resistor \( R_1 \) with a follower circuit. Fortunately, for us, all these (and other) problems have already been solved and the solution is commercially available in one neat package, called an "instrumentation amplifier," such as the AD625 which we will use in the next section.

**Write-up**

6.1.1. Derive and calculate \( V_{\text{out}}(V_x, V_y = 0) \), \( V_{\text{out}}(V_x = 0, V_y) \). Finally, find the transfer function for the difference amplifier, i.e., calculate \( V_{\text{out}}(V_x, V_y) \). Explain the assumptions you made in finding this answer.

6.1.2. What are the differential mode gains measured for each input and what is the measured common mode gain? Why should the differential mode gain measurement with the signal applied to \( V_x \) be identical to the one where the signal was fed into \( V_y \)? How does the differential mode gain agree with the calculations from the previous question?

6.1.3. What CMRR (in dBs) did you measure for your circuit? What was it limited by?
6.2. Strain Gages and Bridge Circuits

What is a Strain Gage?

Strain gages are used to measure mechanical deformations caused by strain or stress. These deformation produce (very small) changes in the gage's (electrical) resistance and these are proportional to the applied force. Strain gages are used, for example, to measure a person's weight on a bathroom scale, to monitor the cargo in an airplane or to observe structural changes in a building or bridge.

The most common type of strain gage, and also the one we will use in this experiment, is a bonded metallic wire strain gage. It has been glued to a C-shaped stainless steel piece, also knows as a "load cell." By hanging some weights on this load cell, you will observe the resulting change in resistance in the strain gage; you will then amplify and calibrate your setup so that it could be used as a scale. In addition to familiarizing you with this type of detector, this exercise will teach you some tricks commonly used in observing and amplifying small signals amid noise.

How does it work?

The resistance, $R_0$, of a wire with resistivity $\rho$, length $L_0$ and cross-section $A$ is given by:

$$R_0 = \rho \frac{L_0}{A}$$
If opposing forces $F$ are applied at each end, it will change its length by $\delta L$ causing it to change its resistance by $\delta R$ according to:

$$F \propto \frac{\delta L}{L_0} = \frac{\delta R}{R_0}$$

In other words, the change in resistance is directly proportional to the applied force. The strain gage's overall resistance is:

$$R_{st} = R_0 + \delta R$$

The metallic wire strain gage employed consists of a serpentine of very fine wire bonded to a piece of nonconductive plastic, electrically isolating it from the load cell.

Strain gages are small, low cost (a few dollars) and very reliable. By itself, strain gages are fragile and can break easily if they are bent too severely. Therefore, they are permanently attached, usually with epoxy, to the surface of an object to which the stress is applied, i.e., a load cell. Though a solid piece of material can be used for a load cell, most load cells are bent in the shape of a C or S to give it some "spring like" characteristics if a force is applied to their jaws. Additionally, stress applied the jaws will produce much larger deformations against the "back" of the load cell (where the strain gage has been attached) than if the same stress were applied across a solid object.

In this experiment, a C-shaped piece of stainless steel from the junk pile of the machine shop will be used. (See Figure 6.4.) (Note: if the object to which the strain gage is attached is too flexible for the applied load, the gage can easily be damaged or it will not return back to its initial condition after the load has been removed. If the object is too rigid or stiff, the change in resistance will be too small to observe. Therefore, the object to which the stress is applied to and to which the strain...
gage is attached to must be “matched.”) If you want to learn more about strain gages, see one of the manufacturer’s web page: [http://www.omega.com/techref/strain-gage.html](http://www.omega.com/techref/strain-gage.html)

### Strain Gage Limitations and Temperature Effects

Strain gages are affected by temperature in two ways: First, if the thermal expansion coefficient of the material to which they are attached is different than that of the strain gage, any temperature changes will produce an erroneous signal. Fortunately, manufacturers produce strain gages that match the expansion coefficients of the material to which they are affixed. Therefore, selecting the appropriate type of strain gage solves this problem. Second, strain gage are made from thin wires whose resistance is directly affected by temperature changes, i.e., they are good “thermometers.” In this experiment we will find a way to cancel this effect.

### Design Goal

We want to design a circuit that produces a clearly observable signal when the jaws of the load cell are squeezed together by ONE hand applying a MODERATE force. In other words, we would like to observe about a 1 Volt change when we apply about 1 pound of pressure.

### Calculate Overall Gain of System

First we need to calculate the overall gain required for our circuit to get a 1 Volt signal response when the load cell is under pressure. Therefore, select one of the two strain gages attached to the load cell and measure $R_o$, the resistance of the gage without any force applied. Now GENTLY squeeze the jaws of the C-shaped load cell with one hand and observe the resistance change, $\delta R$: WARNING: $\delta R$ will be, very, very small so do not apply inappropriate force to the load cell; instead use a good meter! Assuming a constant current, as is the case with small $\delta R$, we find that $\frac{\delta R}{R_o} = \frac{\delta V}{V_o}$. So what gain is needed to get $\delta V / V_o$ to be unity?

### Design 1: Simple Voltage-Divider

To monitor voltage changes caused by the resistance changes of the strain gage, $R_{st}$, consider the “simple” voltage divider circuit shown in Figure 6.7. on the left.

![Figure 6.7. Voltage divider created by a fixed resistor $R_o$ and a strain gage. The dotted circuit on the right provides a possible amplification scheme.](image-url)

Using a Taylor series expansion in $\delta R$, and noting that $\delta R << R_o$, it can be shown that:

$$V_x \approx V_o + \delta V_x$$

where
\[ V_o = \frac{1}{2}V \text{ and } \delta V_x = \frac{\delta R}{4R_o}V \]

Of the two terms in the previous equation, \( V_o \) is constant offset with little interest to us; \( \delta V_x \), which is directly proportional to the applied force, is the voltage we want to observe.

From your previous measurements of \( \delta R/4R_o \), you can see that \( \delta V_x \) will be exceedingly small and needs a very large amplification to be readily observable. Nevertheless, if you were to connect the simple inverter amplifier, show in its dotted outline in figure 6.7., to \( V_x \), you would find that for even modest gain, the inverter circuit's output would saturate. Why or what causes that? Calculate the maximum gain, \( R_2/R_1 \), that the circuit could possibly accommodate before its output saturates if the supply voltages to the op amp were +12VDC and -12VDC. Note, there is no need to build this circuit as it will fall far short of our design goal.

Finally, observe that the value of the upper resistor in our voltage divider was chosen to be \( R_o \). This is not a coincidence. It can be shown that for a fixed \( \delta R \), the maximum \( \delta V \) is obtained by setting the upper resistor to \( R_o \).

**Design 2: Two V-dividers**

By now you (should) have discovered the caveat in the previous design: while we are only interested in amplifying \( \delta V_x \), the previous design amplifies the entire output voltage at \( V_o \), including the constant offset \( V_o \). Since \( V_o \gg \delta V_x \), the amplifier saturates before any significant amplification of the signal of interest is achieved.

The solution to this problem is to subtract \( V_o \) from the amplification, i.e., to amplify the signal at \( V_x \) with respect to the offset voltage \( V_o \) (and not with respect to ground.)

![Image](image_url)

*Figure 6.8. An external power supply or battery with an output voltage \( V_o \) is used as a reference so that we can subtract the offset voltage from \( V_x \).*

If we define the amplified signal as \( \Delta V \), then:

\[ \Delta V \equiv V_x - V_y = V_o + \delta V_x - V_o = \delta V_x \]

Two issues still need to be addressed: a) how do we obtain a fixed reference voltage \( V_o \) and b) how do we amplify the difference voltage between \( V_x \) and \( V_y \)? Since the solution to part b) has already been discussed in the previous section, regarding the difference amplifier circuit, let’s consider the options for the first issue. We could use:

1) a fixed power supply (or battery);
2) another voltage divider, as shown in Figure 6.9.

Option 2 is better. Not only is it cheaper but it also will reject noise due to fluctuations and drift in the source voltage \( V \) since such drifts affect \( V_o \) directly. Using a second voltage divider forces \( V_o \) at
$V_x$ and $V_y$ to drift in unison and, therefore, the drifts will be canceled out in the amplification of $\Delta V$. That is very important, considering just how small the signal $\delta V_x$ is that we want to amplify!

![Figure 6.9. Strain gage voltage divider with a reference “voltage source” formed by a second voltage divider.](image)

Can we improve or simplify this circuit even more? Yes, especially if we replace the voltage divider on the right with a second strain gage / resistor voltage divider, identical to the one on the left hand side. See Figure 6.10.

![Figure 6.10. Strain gage voltage divider with reference voltage, $V_o$, formed by second identical strain gage circuit.](image)

The tricky part is how to mount the second strain gage on our load cell. For example, if both strain gages are mounted on the same side of the load cell, then they would experience identical stresses and $\Delta V$ would always remain 0. On the other hand, if we mount them on opposite sites of the load cell, then while one gage is compressed the other will experience a stretching strain. (See Figure 6.11.)

![Figure 6.11. Placing strain gages on opposite sites of the load cell causes one to be compressed while the other will be stretched.](image)

This placement causes the resistance of one gage to increase while decreasing it by a similar amount for the other gage when stress is applied. Calculating $\Delta V$ now shows that we have gained a factor of two. It is:
\[ \Delta V = 2 \delta V_x \]

While this set-up improves the gain by a rather modest amount, placing two strain gages on the load cell has a far more important benefit: it makes the voltage to be amplified, \( \Delta V \), immune to temperature changes as long as the two strain gages are held at identical temperatures. Remember, the resistance of a strain gage is greatly affected by temperature:

\[ R_{st} = R_o \pm \delta R + R_T(T) \]

(Note: the +/- sign depends on which side of the load cell the gage is mounted.) If both gages undergo identical temperature changes, which is very likely as they are attached to the same piece of metal, it can be shown that \( \Delta V \) will be independent of \( R_T(T) \) for the circuit in 6.10.

Finally, the circuit above is so "famous" that it is usually drawn in a special way, highlighting the Wheatstone bridge configuration, shown in Figure 6.12.

\[ V_x \approx V_o + \delta V_x \]
\[ V_o = \frac{1}{2} V \] and \[ \delta V_x = \frac{\delta R}{4 R_o} V \]

(Hint: find \( V_x(R_o, \delta R) \) for the voltage divider in Figure 6.7. Next expand it in terms of \( \delta R \), remembering that \( V_x(\delta R) = V_x(0) + \delta R \, dV_x/\delta R(0) + \ldots \) 6.2.3. Observe that the value of the upper resistor in our voltage divider was chosen to be \( R_o \). This is not a coincidence. It can be shown that for a fixed \( \delta R \), the maximum \( \delta V \) is obtained by setting the upper resistor to \( R_o \). Prove that! (Hint: proceed like the previous problem and expand \( V_x(R_1, R_2, \delta R) \) with respect to \( \delta R \). (\( R_1 \) and \( R_2 \) is the bottom, respective, top resistor of the voltage divider.) Inspect the term that depends on \( \delta R \) and calculate the values of \( R_1 \) and \( R_2 \) to maximize it.)

6.2.4. Calculate the maximum gain, \( R_0/R_1 \), that the inverting amplifier circuit in Figure 6.7. could possible accommodate before its output saturates if the supply voltages to the op amp were +12VDC and -12VDC.

6.2.5. Derive \( \Delta V \) in terms of \( \delta R \) and \( R_o \) for the final Wheatstone bridge, Figure 6.12.

6.2.6. Show that \( \Delta V \) in 6.12 is not affected by temperature changes if \( R_x = R_o \pm \delta R + R_T(T) \) and if both strain gages undergo identical temperature changes. (Hint: Assume that \( R_T(T) \) is on the order of \( \delta R \); use this to show that \( \delta V_x = (\delta R + R_T(T)) V/(4 R_o) \).) Show how it is, or is not, affected by changes in \( V \), the source supply voltage.
6.3. Instrumentation Amplifiers – “In-Amps”

Instrumentation amplifiers

We will use an Analog Devices AD625 “instrumentation amplifier” (also called an “in-amp”) to detect and amplify the strain gage signal $\Delta V$ from the Wheatstone bridge. See Figure 6.13. A detailed data sheet can be obtained at:


Figure 6.13. Pin-out for the Analog Devices AD625 instrumentation amplifier.

Note that the terminology “instrumentation amplifier” is somewhat misleading: it implies that they are solely used in instruments. In reality, in-amps are used wherever a differential signal needs to be amplified while providing a high input impedance and a large CMRR without having to resort to accurate external resistor matching! As such they are found in monitor and control electronics, medical instrumentation and in audio to automotive applications. (For an excellent tutorial on in-amps see: http://www.analog.com/UploadedFiles/Associated_Docs/56674312778737Complete_In_Amp.pdf)

How does an instrumentation amplifier work?

The in-amp, see Figure 6.16, for the complete circuit, is a clever extension of the now familiar difference amplifier circuit of the previous section, see Figure 6.1., though with its shortcomings cured. In order to understand the in-amp circuit, let’s look at its design stages.

First, to cure the problem of the low input impedance, a pair of (unity) followers in the form of $A_2$ and $A_3$ is added to the difference amplifier. The input impedance has now increased considerably but the transfer function remains the same as for the difference amplifier.
To achieve a high CMRR, the gain of the difference amplifier is usually held at unity with resistors \(R_1\) and \(R_2\) precisely matched and directly incorporated into the op-amp chip. Therefore, gain must be added at the input stage op-amps, \(A_2\) and \(A_3\). This is accomplished by using \(A_2\) and \(A_3\) as two non-inverting op-amps amplifiers.

Even for \(R_1 = R_2\), this circuit can now produce a very large differential gain when \(R_f \gg R_G\). What is less desirable is that the common mode gain of the left part of the circuit has also been
increased by the same amount as its differential gain. “Small” common mode signal will still be
canceled by the difference amplifier circuit on the right but “larger” signals will saturate the left part
of the circuit making it impossible for the difference amplifier to cancel the common mode signals.

A clever remedy to this problem is found by removing the ground point between resistors $R_G/2$.
See Figure 6.16.

![Figure 6.16. Typical “in-amp” circuit that provides high input impedance, large differential gain and a high CMRR as long as $R_1$ and $R_2$ are equal and accurately matched.]

This change does not alter the differential gain of the inverting circuits formed by $A_2$ and $A_3$ but the
common mode gain for this stage is now unity and can easily be removed by the difference
amplifier formed by $A_1$. Since $R_1$ and $R_2$ have been built into the chip by the manufacturer, the user
only has to provide resistors $R_G$ and $R_F$ to set the gain. What’s even more impressive, these do not
have to be matched accurately for a good CMRR! Typical 5% resistors will do just fine.

Prove that $V_{out} = (V_y - V_x)(1 + 2R_F/R_G)(R_2/R_1)$. (Hint: isolate the right hand circuit (which you
have already solved) from the left. Again apply the superposition principle and find the voltages at
point $A$ and $B$ if $V_x = V_x$, and $V_y = 0$; repeat for $V_x = 0$, and $V_y = V_y$. Combine these results and use
the transfer function for the difference amplifier from the previous section and make the appropriate
notation changes.)

**Offset Trim**

At low frequencies, $f < 100$ Hz, the AD625 can have a differential gain as high as 10000. At such a
gain, even small offset voltages between $V_x$ and $V_y$ can saturate the output. Such offsets can be
caused by non-ideal behavior of the op-amps themselves, or in our set-up, due to the resistors in a
Wheatstone bridge not being matched or from permanent deformations in the load cell. It is
therefore desirable to null any permanent offset, or to “trim it away.” The AD625 provides two pins
for this purpose. The 10k turn pot between pins 3 and 4 can be adjusted until the output is 0 Volts
when $V_x = V_y$. Note that the wiper of the turn pot must be connected to $V_{cc}$. See Figure 6.17.
Figure 6.17. Functional diagram of the AD625 in-amp with the external 10k turn pot to adjust the offset.

Finally, build the circuit in Figure 6.17. (Two 8.2 Ω in parallel will do nicely for R_G.) Short inputs V_x and V_y to ground and adjusted the offset trim so V_out = 0.

Test that everything works and measure the CMRR of the AD625 at 100 Hz. Follow the same general procedure as outlined in section 6.1. for the differential amplifier except:

- To measure the very large G_{Diff} we must first attenuate the signal from the function generator by a factor of 1000 or the output will be driven into saturation. Build a voltage divider with a 47k and 47 Ω resistor and feed the signal across the 47 Ω into the in-amp's inputs. Drive the voltage divider with 0.2Vpp, 100 Hz sine wave. Repeat the G_{Diff} measurement for both inputs.
- When measuring the common mode gain, G_{Com}, you no longer need the voltage divider. Instead you will have to increase V_{in} until you can detect it. Stay below 20 Vpp.

How do your results match the manufacturer's specifications for a CMRR of 120 dB? Since the decibels are not very intuitive, calculate how small a differential signal you still could detect if your common mode noise level was 1V if your CMRR was 120 dB?

Write-up

6.3.1. For the circuit in Figure 6.16., show that t V_{out} = (V_y - V_x) \left(1 + \frac{2 R_F}{R_G} \left(\frac{R_2}{R_1}\right)\right). State the assumption made in obtaining this result. (Hint: isolate the right hand circuit (which you have already solved) from the left. Again apply the superposition principle and find the voltages at point A and B if V_x = V_y, and V_y = 0; repeat for V_x = 0, and V_y = V_y. Combine these results and use the transfer function for the difference amplifier from the previous section and make the appropriate notation changes.)

6.3.2. State the G\text{Diff} and G\text{Com} measured. How does it compare with the calculated values using the equation in the previous question? What CMRR (in dBs) did you obtain and how does it compare with the manufacturer’s rating of 120 dB.

6.3.3. How small a differential signal you still could detect if your common mode noise level was 1V if your CMRR was 120 dB?
Finally, combine the strain gage Wheatstone bridge with the in-amp circuit from the previous section. See Figure 6.18.

![Wheatstone bridge with in-amp](image)

- Figure 6.18. Wheatstone bridge with an in-amp. Note: the 1.5k resistors were added to the Wheatstone bridge to limit the current, and, hence, $I^2R$ heating of the 120 $\Omega$ resistors and the strain gages.

Build the Wheatstone circuit shown above using a load cell with the strain gages. Without squeezing the strain gage, adjust the 10 k$\Omega$ offset trim until $V_{\text{Out}} = 0$V again. Now slightly squeeze the strain gage. Does $V_{\text{Out}}$ change accordingly? (If you do not like the sign of the change, what should you change?) Finally, take 4 weights and attach them to your load cell and measure $V_{\text{out}}$.

Plot and calculate from these results: $\delta V_{\text{out}}/\delta$Weight, i.e. the voltage change as a function of weight. How sensitive is your circuit, i.e., what is the noise voltage (due to drift and other noise) and what is the smallest weight change you could therefore observe with this circuit?

**Write-up**

6.4.1. Calculate $\delta V_{\text{out}}/\delta$Weight, i.e. the voltage change as a function of weight. (Have you converted your masses into weights?)

6.4.2. How sensitive is your circuit, i.e., what is the noise voltage (due to drift and other noise) and what is the smallest weight change you could therefore observe with this circuit?

### 6.5. Transmission Gate / FET Switch

The 4066 contains four analog transmission gates. You may think of them as the semiconductor version of a mechanical switch or an electrical relay. (See the figure below.)
Like a mechanical switch, the 4066 transmission gates are bi-directional. For example, either pin 1 or pin 2 of the bottom left most gate in Figure 6.20 can be used as input or output. **The only limitation is that the input voltage must always remain between $V_{SS}$ and $V_{DD}$, i.e., between GND and +15 V!**

While a mechanical switch is turned on, or off, by flipping its lever, the state of each transmission gate is controlled by its corresponding control input. In the diagrams, the control inputs are marked with an arrow. (For example, for the bottom left most gate, the control input corresponds to pin 13.) The gate acts like a closed switch or a “short” when the control input is at $V_{DD}$ (+15V) and an open circuit when the control input is set to $V_{SS}$ or grounded. Setting the control inputs to voltages other than $V_{SS}$ or $V_{DD}$ should be avoided except during switching.

More information on the gates can be found in H&H on pages 494 and 495; see also H&H section 3.11 (pp. 141 - 142).

Before you begin wiring up the circuit of figure 6.20, compare it to the circuit in figure 6.21. Note the similarities: both consist of only a switch (i.e. the transmission gate) and a resistor. Predict for both cases what happens to the output if the switch or transmission gate is closed, i.e. shorted. What happens to the output if the gate is left open? (You may find it helpful to redraw the circuits, replacing the switches with simple short or open circuits.)

a) Select any one of the four gates on the 4066 chip and build the circuit in figure 6.20. Connect $V_{DD}$ to +15 V and $V_{SS}$ to GND. Since the input signal at all times must remain between $V_{SS}$ and $V_{DD}$ select a suitable input signal from your function generator; for example, a 1 kHz sine wave with 5 Vpp and a 3 V offset would be safe. Connect the function generator to the gate only after you...
have selected this signal. (Feeding a negative going signal into the 4066 will destroy it; if you have
already done so get rid of your evidence as quickly as possible and start over with a new chip!)
Now switch the gate ON and OFF by connecting the control input of the gate selected to \( V_{DD} \),
+15 V, or \( V_{SS} \) (GND). Observe the output of your circuit and compare it to your predictions.

If the gate were an ideal switch, its ON resistance would be zero ohms, i.e. a short circuit between
input and output would exist. The real gate has a finite ON resistance; measure it by "closing the
switch" and treating the circuit as a voltage divider.

b) Using the above specified input signal try the switch of figure 6.21. The choice of a 100 kΩ load
resistance is much better than the 1 kΩ used before (chosen so you could measure the ON
resistance easily) and you should find this switch much cleaner. Test it by switching the gate ON
and OFF.

Write-up

6.5.1. Draw the output of the circuit in figure 6.20. with the gate ON and OFF; use \( V_s \), 1 kHz.
6.5.2. What ON resistance did you measure for the gate in figure 6.20.?
6.5.3. How does your value compare to the value given on page 142 of H&H stating that it is
typically between 25 to 100 Ohms?
6.5.4. Draw the output of the circuit in figure 6.21. with the gate ON and OFF; use \( V_s \), 1 kHz. How
does it compare to 6.5.1?
6.5.5. What OFF resistance did you measure for the gate in figure 6.21.?

6.6. Sample and Hold

Figure 6.22 shows a "sample and hold" circuit. It is used in applications where a signal is not
allowed to change while it is measured, for example by a voltmeter or an Analog-to-Digital-
Converter (ADC). The 4066 FET switch, which has been covered in section 6.5., isolates the
output from the input while it is in its open position. The capacitor holds the input voltage present at the very moment the switch is opened. In an “ideal” circuit, it would retain the voltage indefinitely. In reality, the output voltage begins to drift or “droop” in a matter of a few tens of seconds.

Begin by redrawing the circuit in its two possible states i.e. with the control input lines either grounded (i.e. the “hold” state) or at +15 V (the “sample” state); in your redrawn circuit diagram, omit unnecessary elements: for example, replace the closed switch simply with a wire. For each case, how does $V_{out}$ compare to $V_{in}$? The circuit contains a 411 op-amp follower providing unity gain. Therefore, you could replace it in your analysis with a simple wire; when or why would that be incorrect? For further analysis read D&H page 353 and 354 or H&H page 152.

Construct the circuit of figure 6.22. and drive the input from the +6V output from the Agilent power supply. While continuously changing the input voltage, make sure that it always remains positive and that it does not exceed +15 V. (If you want to know why, reread the first paragraph of section 6.5 of this manual.) On the scope, observe both the input and output signal at the same time with the control input of the 4066 first connected to +15 V, then to GND. Does it behave as predicted?

(If you need to troubleshoot this circuit, split it at the non-inverting input. Set the 4066 control voltage to “sample;” the output signal from the 4066 should look identical to its input signal. Now set the 4066 control voltage to “hold;” you should no longer see the input signal at the output of the 4066. Finally, feed the input signal directly into the non-inverting input of the 411; the output should look identical to the input. Reconnect everything.)

The “hold” function relies on retaining a charge stored in the capacitor; because of the leakage current in the capacitor, this circuit is prone to “droop”. Measure the droop rate (Volts/min.): apply a 7 VDC signal with the control input in the sample mode. Now switch the control voltage to the hold mode, disconnect the input signal and measure the droop rate with a voltmeter and a watch.

**Write-up**

6.6.1. Draw an arbitrary input wave; draw $V_{out}$ for the circuit in figure 6.22. If for the first half of the input wave the control line is held at +15 V and for the second half the control line is held at ground.  
6.6.2. What droop rate did you measure?  
6.6.3. What effects other than capacitor leakage current cause the output to droop?
Digital Output Levels and Logic Families

Unlike an analog signal, the output of a digital signal is defined as HI or LO, or, ON or OFF. Various logic standards exist (see H&H page 475) specifying the exact voltage thresholds which determine if a signal is considered HI or LO.

With the exception of the first exercise, you will be dealing for the rest of the course exclusively with a logic standard called "transistor-transistor logic," abbreviated as TTL. Though it is probably the most widely used logic family, be aware that in physics, especially in High Energy, you will encounter other logic families, notably NIM or ECL. Interconnecting different logic families without suitable converters can lead to disastrous results!

You should also be aware that for some logic families, including TTL, different, newer logic standards may exist. These newer standards have decreased the voltage range to conserve power. For example, traditional TTL is defined as having a 0 to 5V range; the new Low Voltage TTL (LVTTL) which is becoming very popular, has a range from 0 to 3.3V. While some older TTL devices do function at the new LVTTL levels, others do not. Therefore, interfacing TTL with LVTTL devices must be done with care. You will be warned when you have to connect devices from the two families, as will happen towards the end of next chapter.

Observing Digital Output Levels

When dealing with digital logic, one is rarely interested in the exact value of the output voltage. Hence, unless they are specifically requested, values such as $V_{out} = 4.32\, \text{V}$ are generally uninteresting. What we are usually looking for is whether an output is HI or LO; while it is possible to observe a digital output with a voltmeter, it is somewhat overkill. It is much more common to use a "scope" (make sure it is in the DC coupled mode), a Logic Analyzer (a specialized kind of scope that only displays logic levels as a function of time) or a simple LED in series with a $330\, \Omega \pm 100\, \Omega$ resistor.

There are two important facts to remember about LED's. First, as the name Light Emitting Diode implies, an LED is just a diode that glows when it is forward biased. As you learned in chapter 3, one should NEVER connect a diode directly across a voltage source without a current limiting resistor. Hence, ALWAYS use the LED in series with a 200 - 400 $\Omega$ resistor. Second, just like an ordinary diode, an LED is directional; hence, carefully observe which lead is the anode and which
is the cathode. (The cathode lead is usually shorter and the LED is "flat" on the cathode side.) If you plug it in backwards, the LED will obviously never glow.

![LED identification and typical usage](image)

Finally, a word on conventions. Though HI or LO are often used to identify digital levels, the table below lists other commonly used terms:

<table>
<thead>
<tr>
<th>HI</th>
<th>LO</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>TRUE</td>
<td>FALSE</td>
</tr>
<tr>
<td>5 V (only in TTL)</td>
<td>GROUND</td>
</tr>
<tr>
<td>3.3V (only in LWTTL)</td>
<td></td>
</tr>
</tbody>
</table>

![Table 7.1](image)

### 7.1. Diode Gates and TTL

Almost all commercially logic gates use various kinds of transistors to control the output. The transistors provide a high input impedance and a low output impedance and, therefore, act as current amplifiers. This property is extremely useful if one output must drive multiple inputs, a condition which is fairly typical in logic circuits.

Nevertheless, as you will see in the first part of this exercise, you can also construct gates without transistors using only diodes. Though there are not many practical applications for them but they do give us a glimpse into the inner workings of a gate.

![Diode OR gate](image)

Figure 7.2. Diode OR gate. Inputs are A and B, output is Q.
Apply every possible combination of TTL level signals (HI = +5 V, LO = GND) to inputs A and B of the diode OR gate in figure 7.2. Observe the output at Q with an LED (make sure you have read the previous section about LED's.) Does it agree with the truth table in D&H, table 11.2, page 237, or H&H, figure 7.2, page 479?

For this AND circuit check all four entries in the truth table in D&H, Table 11.1, page 236 or figure 7.3 of H&H, page 479. Though this AND gate is very easy to construct it suffers from one major drawback: Its output impedance is rather high which severely limits its applications.

Now compare the performance of the diode gate with the commonly used TTL integrated circuit 74LS08 quad AND gate. Use Appendix C or any TTL Databook for the connection diagrams. (For example, check the 1989 National Semiconductor LS/S/TTL Logic Databook or check the web at: http://www.national.com) Get used to the conventional choice of pins for VCC (+5 V always for TTL) and GND in opposite corners of the package. Also, if you get into the habit of pointing all the chips the same way on the breadboard, the power supplies become very easy to wire up. Finally, when testing TTL circuits, do not expect to find +5.0 V and 0 V for logic levels; the specs allow HI +2 V to +5 V and LO 0 to 0.8 V.

Use an LED to verify the truth table. Determine what happens to the output if an input to a TTL gate is left floating i.e. if it is neither connected to GND or HI. Is such an input considered HI or LO? The answer should explain why unused inputs are conventionally connected to +5 V through a 1 kΩ resistor.

**Write-up**

7.1.1. Make a truth table for the circuit in figure 7.2.
7.1.2. Make a truth table for the circuit in figure 7.3.
7.1.3. Are floating inputs to TTL gates considered HI or LO?

7.2. **Exclusive OR**

The XOR truth table indicates that the output is HI only when the two inputs to the XOR gate are LO and HI, or, HI and LO. This can be expressed as:

\[ A \oplus B = \overline{A}B + \overline{B}A \]

(7.1.)
Convince yourself that the exclusive OR circuit of figure 7.4 is an implementation of equation 7.1, where all the gates have been substituted with their NAND equivalents. Construct it with TTL integrated circuits, 74LS00 (quad NAND) and 74LS04 (hex inverter), and try it out.

Create a design which uses only NAND gates – you may use any number of gates needed. (For a challenge, the minimum number necessary is four.) Often it is more convenient to stock a large number of a single "universal" gate than many different special purpose gates.

**Write-up**

7.2.1. Show your truth table for the circuit in figure 7.4.
7.2.2. Show a design for the XOR which uses only NAND gates.

### 7.3. Binary Adder

The binary adder forms the basis for all modern digital computers. In this exercise you will design and build an adder circuit capable of summing two two-bit binary numbers.

First add two two-bit numbers by hand and note the operations. At each bit position binary addition consists of two steps: the two bits are added, then the sum is added to the carry bit which arises from the sum of the previous two bits. At any stage of the addition a carry bit may be propagated to the next bit position; of course, there is no carry bit to add to the sum of the least significant bits.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>S</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

*Table: Truth table for the half adder*

[Diagram of half adder]

Figure 7.5. Block diagram half adder
A circuit that is able to add two one bit numbers is known as a half adder. Its block diagram is shown in figure 7.5. Complete its truth table. Next design and build a half adder with the available TTL gates and verify its operation.

\[
\begin{array}{ccc}
A & \rightarrow & C_{\text{out}} \\
B & \rightarrow & S \\
C_{\text{in}} & \rightarrow & \\
\end{array}
\]

- Figure 7.6. Block diagram for a full adder

A full one-bit adder sums three (input) bits. (Its block diagram is shown in figure 7.6.) As such it is far more versatile and useful than the half adder because simple addition operations involving only two terms actually require the addition of three numbers if one considers the carry digit to be the third number.

Starting with a new truth table, develop a design for a full one-bit adder (see figure 7.6.). Once your design is optimized, you should construct the circuit and verify its operation.

### Write-up

7.3.1. Complete the truth table for the half adder (shown in figure 7.5.).

7.3.2. Using proper logical symbols, show the circuit that you implemented for the half adder.

7.3.3. Write down the complete Boolean logic expressions using standard Boolean logic notation for \( S \) and \( C_{\text{out}} \) in terms of \( A, B \) and \( C \) for the full adder in figure 7.6. You do not have to simplify it but it has to be correct.

7.3.4. Show your design for the full one-bit adder (see figure 7.6.).

### 7.4. Data Selector / Multiplexer (“MUX”)

\[
\begin{array}{cc}
\text{Data Selector or Address Line(s)} & \rightarrow Q \\
A & \\
\end{array}
\]

- Figure 7.8. Multiplexer

In many data acquisition applications, it is desirable to select data from several sources. A circuit known as a data selector or multiplexer, also known as a “mux,” will perform this function. Such a device has one or more data selector or address lines to select which one of its data inputs is connected to the output. Its function is similar to a telephone switchboard: a user, connected to the output \( Q \), is being connected to a specific user at the inputs after “dialing” his or her number at the address lines.

In this exercise, you will design and construct a two-input data, one-bit data selector as shown in figure 7.8. The output \( Q \) should follow input \( D_0 \) when the control (or address) line \( A \) is LO, and it should follow input \( D_1 \) when the control line \( A \) is HI.
First, write down the truth table for the data selector. Next, design a circuit using available gates. Finally, verify the operation of your circuit by comparing its actual truth table to the one which you first wrote down.

More complicated multi-input, multi-bit data selectors can be built by a straightforward generalization of your design. Design and build a 4 data input ($D_0$, $D_1$, $D_2$, $D_3$) multiplexer. You should strive for an implementation which is easy to understand. How many control (address) lines are needed to select one of the 4 data inputs?

**Write-up**

7.4.1. Write down the truth table for the data selector.
7.4.2. Show your design for the two input data selector circuit using available gates.
7.4.3. Show your design for the 4 data input multiplexer.

### 7.5. Multiplexer: 31-Day Machine

Now construct the 31-day machine of figure 7.9, which is figure 8.41 (p. 500) of H&H. The circuit lights up the LED connected to its output when the month whose number is applied to the control input has 31 days in it. Check it out by applying HI and LO signals at the input corresponding to the first eight months (January = 0001, February = 0010, March = 0011, etc.).

Note that the 74LS151 has only 3 control inputs or address lines, A, B and C. Therefore, the maximum number of data inputs that can be multiplexed is 8, i.e., $D_0$ through $D_7$. Nevertheless, the circuit in figure 7.9 produces the correct output for 12 distinct months. Study it carefully and see how it accomplishes that.

**Write-up**

7.5.1. Show the output of the circuit in figure 7.9, for the first eight months.
7.5.2. Compare the results in 7.5.1 with your knuckles or a calendar.
7.5.3. A typical 3-control line multiplexer such as the one used in the 31-day machine allows for $2^3$ distinct selections. How is it then that the circuit in figure 7.9, is able to make 12 distinct selections?
Write-up Format: Short

Reading:

E4E: Sections 6.3-6.4
Horowitz&Hill: 8.16 - 8.23
Horowitz&Hill: 4.24
Horowitz&Hill, 15.10, (pp. 1019-1021)

or:

Diefenderfer&Holton: Sections 12-7 and 12-8 (pages 287 to 294).
Diefenderfer&Holton: Sections 12-1 and 12-2 (pages 261 to 266).
Diefenderfer&Holton: Sections 12-4 and 12-5 (pages 270 to 284).
Plonus:  Section 7.5  (pages 254 to 265)

Clock

Various circuits in this and the following chapters require a "clock" i.e. a series of TTL pulses. When not stated otherwise, always use the function generator's SYNC output as a clock. Take a second to locate it; it's the upper BNC on the right hand side of the function generator. This output always produces the correct TTL output, independent of the DC OFFSET or Amplitude selected. Do not use the analog output of the function generator (lower BNC) to drive a TTL or CMOS circuit. When powered with voltage below ground, i.e., negative voltages, TTL or CMOS devices can be seriously harmed.

Data Books

The pinouts of most of the chips that you will use are given in the appendix C at the back. However, you should also become familiar with the data books on the web and on the bookshelf in the lab. For web based access, either check the Texas Instrument site at:
http://www.ti.com/sc/
or the National Semiconductor at:
http://www.national.com

When looking for information about a particular chip, it is not crucial that you select a particular manufacturer; except for details that do not affect us now, chips with identical part numbers do not differ from one manufacturer to the next.

If you are looking for a TTL product (74) and can not find it, try its military equivalent number. For example, if you cannot find information on the 7408 try 5408. This is especially important for the National Semiconductor website. As far as this course is concerned, the difference between the 74 and 54 product family is minute and the pinouts are usually identical.
8.1. Monostable Multivibrator ("One-Shot")

A monostable multivibrator is not a true logic gate. It is probably more similar to an alarm that goes off for a preset time interval (set by $R_{ext}$ and $C_{ext}$) when its inputs are triggered. Its output is a digital pulse and satisfying a logical condition at the (clock) input triggers it. The duration of the output ($t_w$) is fixed by an external resistor, $R_{ext}$ (connected to the $R_{ext}$ pin and HI) and an external capacitor, $C_{ext}$ (connected to the two $C_{ext}$ pins.) There are two complete one-shots on a 74123 chip; each comes with two complementary outputs: $Q$ is normally LO and becomes HI during the pulse, while $\overline{Q}$ is normally HI and becomes LO during the pulse.

Hook up all the necessary connections to the 74LS123 one-shot so that a clock signal fed into input $B$ will trigger the 74LS123 on the positive going edge. The data sheets for the 74LS123 from the 1989 National Semiconductor LS/S/TTL Logic Databook have been reproduced in Appendix C of this manual or they can also be obtained on the web at:

http://mxp.physics.umn.edu/national/htm/nsc04105.htm

In the data sheets mentioned above, find an equation and/or a table that relates the pulse width ($t_w$) to $R_{ext}$ and $C_{ext}$. Keeping $R_{ext} > 5$ k, calculate $C_{ext}$ for $t_w = 500$ nsec. Connect these components to the one-shot. Now measure $t_w$ while clocking the one-shot with a 100 kHz TTL square wave.

---

**Notation**

Be aware that data sheets occasionally display negative logic such as $\overline{Q}$, also as $Q$ or $-Q$. 
Think what happens when the $\overline{Q}$ output of a one-shot is fed into the "trigger input" $B$ of a second one-shot. (See figure 8.2.) Next consider what would happen if the $\overline{Q}$ output from the second one-shot is fed back into the first one; draw a timing diagram for $Q$, $\overline{Q}$ and $\overline{Q}$. From your timing diagram you should clearly see that the one-shots act as an oscillator, producing a rectangular wave at $Q$. The width of the HI pulses ($\tau_H$) at $Q$ is determined by $R_1$ and $C_1$ while the pulse width of the LO pulses ($\tau_L$) is controlled by $R_2$ and $C_2$. Unlike the oscillators from chapter 6, this circuit allows the duty cycle (i.e. the percentage of time a square pulse cycle is HI) to be adjusted. A duty cycle of 50% produces a standard square wave.

From your data above design a 100 kHz, 20% duty cycle pulse generator. Build and test it. (Hint: Depending on which wires were connected last, your circuit may oscillate properly or may hang in its initial state. In the later case, jump-start the circuit by disconnecting the wire from the $A$ input to GND; as soon as the wire is reconnected to GND, the circuit should begin to oscillate properly. If you see short spikes instead of nice pulses, touch the capacitors with your hands.)

Write-up

8.1.1. What value did you measure for $t_w$?
8.1.2. Make a timing diagram of $Q$, $\overline{Q}$ and $\overline{Q}$ for the circuit in figure 8.2.
8.1.3. Draw a schematic for your 100 kHz, 20% duty cycle pulse generator and indicate the values for the resistors and capacitors used as well as the actual $\tau_L$, $\tau_H$ and the overall frequency measured.

8.2. D-Type Flip-Flop

A D-type flip-flop is most often used as a one-bit memory storage device. Similar to a digital camera taking a snapshot when its shutter release is pressed, a D-type flip-flop will store the state of its D-input at the very instant its clock input changes. Depending on the type, the flip-flop will take a "snapshot" either on the positive or the negative going clock cycle but never at both clock edges! The result of the snapshot is then "immediately" displayed through the Q output. From this description, one can see its usefulness as a buffer to store one bit of data till it is needed and read at latter time from its Q output.

Wire up the synchronous divide-by-three circuit of figure 8.9.
The NOR can be either one section of a 74LS02 (quad 2-input NOR) or else a 74LS00 (NAND) cleverly contrived to look like one.

Draw a timing diagram and show the clock, \( Q_1 \) and \( D_1, D_2 \) and \( Q_2 \). To start, arbitrarily choose \( Q_1 = Q_2 = \text{HI} \) and then work out the values for \( D_1, D_2, Q_1 \) and \( Q_2 \) for the next 6 clock cycles. From your timing diagram, you should observe a state that \( Q_1 \) and \( Q_2 \) never reach in the divide-by-three sequence (see also page 514 of H&H). Set the counter in this state by temporarily grounding the \( \text{SET} \) or \( \text{RESET} \) inputs and verify (using LED's) that it does not hang in that state when it is toggled with a very slow clock signal from your function generator. Now try it with the 1 kHz clock and the oscilloscope. Compare the signals seen at \( Q_1, Q_2 \) and the input to your timing diagram.

**Write-up**

8.2.0. From the description given above for the 7474, explain what happens to \( Q \) if \( \overline{Q} \) was directly connected to \( D \)? Draw a timing diagram and show the state of the clock, \( Q \) and \( \overline{Q} \) for three clock cycles.
8.2.1. Draw a timing diagram for the circuit in figure 8.9. and show the clock, \( Q_1 \) and \( D_1, D_2 \) and \( Q_2 \).
8.2.2. Which state can the divide-by-three sequence never reach?

**8.3. Frequency Counter**

**8.3.1. Introduction**

In this exercise you will build a fairly sophisticated frequency and period counter. The principle behind a frequency counter, as the name implies, is to count the number of cycles during a precisely monitored reference or sampling time period, \( t_{\text{sample}} \). For example, if you counted the cycles (on the input signal) for one second, then the total count would represent the frequency of the input signal in Hertz; similarly, if you counted for one microsecond, the count would indicate the frequency in Megahertz.

Other frequency measurement techniques exist; for a list see H&H. Nevertheless, the approach used here, besides its extreme simplicity, provides an amazing accuracy that is only limited by:

- a) the accuracy of the reference timer;
- b) the size of the counter, i.e., how many bits "wide" it is;
- c) for how long one counts.

The first two limitations listed can be solved by selecting the appropriate hardware. If the last constraint is not an issue, i.e., if we have enough time to let the counter run by making \( t_{\text{sample}} \) large, this frequency measurement approach can yield extremely high precision.
This explains why sometimes experimental physicists choose to convert a voltage or current signal into a corresponding frequency signal. (A voltage controlled oscillator, VCO, would do the conversion by providing a frequency that is proportional to some given input voltage.) While in such experiments small changes in voltage or current are monitored, the actual experimental setup is designed to measure small changes in frequency.

8.3.2. Schematic of the Frequency Counter

The frequency counter circuit consists of two major components.

On the left is the frequency counter's reference timer and control logic. The 74193 counter and the attached NOT and NAND gates provide the control signals (reset and store) for the 14 bit cycle counter. This part acts like a state machine. It tells the 14 bit cycle counter module when to count the input signal's cycles and when to reset itself. This part of the circuit is implemented with standard TTL gates that you will wire up on a breadboard.

On the right, the 14 bit cycle counter and hexadecimal display has been implemented as a Verilog Firmware module on a Digilent BASYS Field Programmable Gate Array (FPGA) board. The BASYS board contains over 100 000 digital logic gates. Each of them can be programmed to perform a specific digital logic function such as an AND, NAND, NOR gate etc. The board has been programmed for you to act as a 14 bit counter, a binary to decimal converter and it also will operate the hexadecimal display. More information on the board is given in Appendix L.

(The past, students also had to wire up this part of the circuit using a 74945 chip and an LED display. This year, we decided to use a BASYS board because these chips are becoming hard to...
find and the wiring consists of over 40 wires! In addition, though the board has already been programmed for you, later in this course, you will learn how to program it yourself using a programming language called Verilog. Consider this as an introduction to FPGA board and its powerful capabilities.)

NOTE: the BASYS board can only use LVTTL signals, i.e., 0 to 3.3V signals. Therefore, you must drive all your TTL chips with a 0 and (about) 4.4 V supply voltage (instead of the usual 0 and 5 VDC) to get a 0 to 3.3V output signal! (Note: the output voltage is usually less then the driving voltage because one loses some voltage on the output due to a diode drop or two in the gates; therefore, driving the TTL chips with a 4.4 V supply voltage will result in acceptable LVTTL output voltage levels.) Also make sure that the input signal from the function generator fulfills the LVTTL requirements: do not use the SYNC output; instead use the normal analog output with a square wave and DC offset to get a 0 to 3.3 V signal! (Check on the scope before interfacing!)

8.3.3. Parts of the Frequency Counter

The frequency counter that you will build consists of the following components, both real and virtual:

a) Reference time source: The 8651B programmable pulse generator contains a very accurate (±5 ppm) 100 kHz crystal oscillator and a programmable “divide-by” circuit that allows the generation of 64 different output frequencies. The 8651B will be used as our reference clock and it will be programmed to output a clock signal with period $t_{clock}$.

b) BASYS Board with 14 Bit Cycle Counter and Display: The BASYS board has been programmed to contain a 14-bit counter; it will also drive the 4 digit LED that displays the counter values. See Appendix L for further discussion and pinouts.

The inputs and controls for the BASYS board are:
1) The clock input for the 14-bit cycle counter is at pin JD1. It increments the counter at each negative edge. The 14-bit cycle counter keeps count of the input signal’s clock cycles.
2) The asynchronous $\text{reset}$ input at pin JA4 resets the 14-bit cycle counter to 0 when its input goes LO. The counter will not count as long as it remains LO; it will start counting when it goes HI again. This signal is needed at the start of each $\tau_{\text{sample}}$ period.

3) The $\text{store}$ input at pin JA1 stores (at the negative going edge) the current counter value at the end of $\tau_{\text{sample}}$ in a 14 bit latch so that it can be displayed until the next value is latched. It also starts the conversion of the binary signal to a binary-coded-decimal (BCD) signal so it can then properly be displayed in a decimal format.

Table: BASYS board’s 14 Bit Cycle Counter States and its Inputs (“x” means: don’t care)

<table>
<thead>
<tr>
<th>reset</th>
<th>store</th>
<th>14 Bit Cycle Counter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Increments the counter at each negative edge of the BASYS board clock input, JD1</td>
</tr>
<tr>
<td>0</td>
<td>x</td>
<td>Counter is set to 0 and remains stopped.</td>
</tr>
<tr>
<td>1</td>
<td>↓</td>
<td>Current count value is stored in a memory latch and converted to BCD and displayed</td>
</tr>
</tbody>
</table>

c) **Sequential logic circuit**: The additional logic needed to provide the control signals for the measurement, namely $\text{reset}$ and $\text{store}$ can be implemented by using a 74193 4-bit up counter with some additional combinational logic gates such as NAND and NOT. We will refer to this counter as the state counter because it operates like a simple state machine.

Before you start building the circuit you should make a complete timing diagram; it will be very useful in understanding, designing and trouble shooting this circuit. Start your timing diagram by drawing about 40 clock cycles, i.e. $\tau_{\text{clock}}$ ; show the corresponding $Q_0$, $Q_1$, $Q_2$ and $Q_3$ of the state counter and the $\text{reset}$ and $\text{store}$ lines. In your timing diagram, indicate clearly the sampling period $\tau_{\text{sample}}$ i.e. the time between $\text{reset}$ going LO-HI and $\text{store}$ going HI-LO. Since we want the LCD to indicate the frequency in kHz how long should $\tau_{\text{sample}}$ be? Why? How many clock cycles $\tau_{\text{clock}}$ occur during $\tau_{\text{sample}}$? Hence, what should be the clock frequency, $f_{\text{clock}}$? Using our 4 digit display, what will be the useful frequency range of the function generator?

8.3.4. **Setup**

Now start designing and building the circuit. You will need to consult the appropriate data sheets to work out your design. (Most of them are at the back of this manual.) Do not attempt to assemble the entire circuit without testing each individual part.

Connect the wires for the 8651B programmable pulse generator. Check the data sheets in Appendix C and wire the 8651B so that its output frequency is the desired $f_{\text{clock}}$. Check that the output of the 8615B looks fine on the scope, including the amplitude of the output. Recall that the BASYS needs LVTTL logic signals with a maximum voltage of 3.3V; therefore, power the circuit with 0 and 4.2 Volts instead of the usual 0 to 5V. (Note the spec sheet indicates that the 8651B needs 5 V – our tests have shown it works fine with 4.2 V for VDD.)

Next, you must create the 74193 state counter and associated logic. Check the output of $Q_0$ and compare it to the output of 8651B. What frequency is it? Does it make sense? (Note: a non-working, incorrectly wired 74193 counter has been the main source of trouble with this exercise.)
Next wire up the 7420 4-input NAND gate and the 7404 NOT gates. Finally, measure the time interval between the reset and store signal. Does it make sense? Compare it with your timing diagram.

Before you go on, once more verify that you’re powering ALL your TTL chips with a maximum voltage of 4.2 V to get the appropriate VLTTL signal.

Now connect everything together. Plug the wall outlet into the BASYS board; if it works properly its LED display should show 0000. Connect your reset and store signal to the BASYS board. (You may find the BNC adapters for the board useful.) Last, check that the output from the function generator is a suitable VLTTL square wave; if so, connect it to the clock input of the BASYS board.

Vary the frequency of the input signal from your function generator and check that everything works fine. Take a couple of frequency measurements to determine the accuracy of your frequency counter. Take readings at the low and high end of the frequency range and one or two in the middle. Compare your frequency readings with values:
- from the function generator
- calculated from the scope

When everything works, show your frequency counter to your TA.

### 8.3.5. Period Counter

As you have probably noticed when measuring low frequency signals the accuracy of the frequency counter decreases. As a countermeasure, one could increase the sampling time period, \( \tau_{\text{sample}} \), but then one would also have to wait longer for the display to update, which could be inconvenient especially if one wants to count many signal cycles.

As an alternative, one may prefer to measure the period of one or more signal cycles. This is done with most (good) commercial frequency counters with the push of a button. To convert our frequency counter to a period counter all you need to do is to switch two wires.

### 8.3.6. Theory of Operation of the Period Counter

To determine the period of an unknown signal the most direct method is to count the number of reference cycles, \( \tau_{\text{ref}} \) (or clock ticks) during one signal cycle. For an accurate measurement, \( \tau_{\text{ref}} \ll \tau_{\text{signal}} \).

In our period counter, \( \tau_{\text{ref}} \) is provided by the 8651B which (in addition to switching the two wires mentioned above) needs to be properly adjusted so that after 10 signal cycles the signal period is displayed in milliseconds. While \( \tau_{\text{ref}} \) is directly fed into a counter, the signal is fed into a circuit that will display the clock counter value after 10 signal cycles.

From this description find the appropriate two wires to switch and convert your frequency counter into a period counter. Then adjust \( \tau_{\text{ref}} \) so the output will display the signal period in milliseconds. Test your circuit with a 5 Hz signal. When everything works, show your period counter to your TA.

### Write-up

8.3.1. Hand in a detailed schematic of the frequency counter (show all chips and indicate the relevant pinouts).
8.3.2. Hand in your complete timing diagram (of the frequency counter) showing about 40 clock cycles, $\tau_{\text{clock}}$, and the corresponding Q0, Q1, Q2 and Q3 of the state counter and the reset and store lines. In your timing diagram, indicate clearly the sampling period $\tau_{\text{sample}}$ i.e. the time between reset going LO-HI and store going HI-LO.

8.3.3. Give a short description of what each part of the frequency counter circuit does. What is the range of the frequency counter?

8.3.4. Give a short description of what each part of the period counter circuit does. What is its range?
Write-up Format:  Short

Reading:

E4E: Sections 7.1 - 7.2
This chapter is (mostly) self contained. It contains a lot of information about the Verilog language. You should definitely read it before coming to the lab.

- A short introduction to the Verilog language by Prof. Mans can be found in Appendix K.
- Information about the Xilinx ISE Webpack and the Digilent Adept facility is given in Appendix J.
- Finally, a detailed description of the BASYS board (including pin-outs) is given in Appendix L.

In this and the following chapter we will return to digital logic circuits. No new digital logic concepts will be introduced; instead, we will revisit some of the ideas from chapters 7 and 8. This time however, instead of using simple logic gates connected with wires, we apply the concepts in a new, more powerful way. The tools that we will use are:

a) a programmable logic board by Digilent Inc. called BASYS which uses a Field Programmable Gate Array (FPGA);

b) a hardware description (programming) language (HDL) called Verilog.

With these two tools we are able to turn your knowledge of digital circuits into powerful and (hopefully) useful applications. They will allow you to efficiently implement high speed, large scale digital logic designs that would be far too tedious to build using individual chips!

Introduction to the BASYS FPGA Board and Verilog

While it is possible to use either specific gate chips or universal gates such as the NAND, most modern systems have replaced combinations of “fixed logic” chips (also known as application-specific ICs, or ASIC devices) with programmable logic. The “Basic Systems Board” or “BASYS” board which you have used for the frequency counter in chapter 8 is an example of such a programmable logic system. The large chip in the center labeled “Xilinx” is the programmable logic chip called an FPGA (Field-Programmable Gate Array) – as a rough estimate, it contains about 100,000 gate-equivalent of logic. In addition to the FPGA, the board contains connectors, displays, switches, and the support machinery to let you program the FPGA from your computer. You can find the full documentation for BASYS in Appendix I.

Inside the FPGA there are a large number of memory elements or “lookup tables”. Each lookup table (LUT) has a number of inputs and one output. The (selector) input values select an item in a list of memory elements. It behaves essentially like a multiplexer, familiar to you from section 7.3 and 7.4. The diagram of 4 bit multiplexer and its truth table are shown below.
Consider the multiplexer shown above with two (selector) inputs, \( S_0 \) and \( S_1 \) and one output, \( Q \). If both (selector) inputs \( S_0 \) and \( S_1 \) are low, then the value in the first memory cell, i.e., input \( D_0 \), is selected as the output \( Q \). If the \( S_0 \) is high and \( S_1 \) is low, the value in the second memory, i.e., input \( D_1 \), becomes the output. Similarly for \( S_0 \) low and \( S_1 \) being high, \( Q \) corresponds to \( D_2 \) and so forth. The lookup table behaves as an arbitrary logic function generator: any logic function of two inputs and one output can be implemented using it. Besides the LUTs, most of the rest of the FPGA consists of registers (Flip-Flops) and multiplexers (MUX) which let us connect external pins to the inputs and outputs of the LUTs.

The FPGA is very flexible – we just have to program it! The program defines the contents of the LUTs and the multiplexer settings. We could do it by manually specifying all the LUTs and mux values by hand, but that would be very slow and prone to error. Instead, we will use a computer-aided design (CAD) process where we describe the behavior we want the FPGA to perform, and then we use a (free) software tool from Xilinx called “Webpack” to convert that into the form needed for the programming of the FPGA. We will use a language called “Verilog” to define the behavior of the system. There is a compact guide to Verilog in Appendix K though you will learn it as you work on the exercises in this manual.

In the next section you will learn how to program the BASYS board and how to specify how to get signals into and out of the FPGA. Finally you will learn how to implement combinational and sequential logic structures.

### 9.1. Digital I/O With the BASYS FPGA

#### Verilog Overview: Modules

The Verilog programming language is based on the C language syntax. Similarly to the C-syntax which relies heavily on functions, Verilog uses modules. A very simple Verilog module named `MyModule1` is shown below:

```verilog
module MyModule1(a, b, q);
    input a;
    input b;
    output q;

    assign q = a & b;
endmodule
```

As you can see, the structure of a module is very similar to a C-function. Nevertheless, notice the following similarities / differences:

- There are no curly brackets in Verilog; instead keywords like `endmodule` (or `begin / end`) are used. Note though that semicolons were retained!
- Each module begins with the keyword `module` and ends with the keyword `endmodule`.
- The variables `a, b, q` look like function arguments but they are now called “inputs” or “outputs”; they are then further defined below the (module) header.
• Similar to C, once the function and its inputs and outputs have been declared, the statements that will do “something” useful follow them. For right now you do not have to understand them. (In case you are curious, in the above example, \(a\) and \(b\) perform a simple AND operation whose result is then assigned to the output \(q\).)

• Finally, unlike C, there is no “special” module that needs to be named “main” and which calls all the other modules in your program. Instead, in the Xilinx compiler, you can select any one of your modules and assign it to be the “Top Module” which then functions the same as a “main” module.

For illustrative purposes, a graphical representation of the module MyModule1 is shown below. By convention, inputs are drawn on the left side and the outputs are on the right.

For Pin Assignment

A Verilog module is only useful if we are able to communicate with its inputs and outputs. This communication is accomplished by using the appropriate pins on the actual FPGA chip. Before we can make use of them they must be linked, i.e., assigned, to the corresponding inputs and outputs using the (“LOC = pin number”) command.

For example, in the code shown below we assign the pin labeled P69 (which happens to be connected to the rightmost push button labeled BTN0) to some input named \(a\):

\[
\text{(* LOC = "P69" *) input a; } //\text{NOTE the LOC statement is enclosed in parentheses!}
\]

Though we may name our inputs and outputs anything that pleases us, at least as long as it is not a reserved Verilog keyword, how do we find the actual pin numbers that we want them to link to? They are listed in Appendix L on the last page and they can also be obtained directly from uDigilentInc's Website: [http://www.digilentinc.com/Data/Products/BASYS/BASYS_E_RM.pdf](http://www.digilentinc.com/Data/Products/BASYS/BASYS_E_RM.pdf). Check and see if you can confirm that P69 indeed is connected to the rightmost pushbutton labeled BTN0.

(Warning: Note that there is a bug in the DigilentInc documentation concerning all pins with pin numbers less than 10. In the documentation these pins are listed with a leading 0, as, for example, P06. For Verilog to compile correctly, they must be entered without the 0, i.e., as P6)!

Here are some comments regarding the pins of the FPGA. The chip in the center of the BASYS board, a Spartan 3E, has 144 pins. They are all powered by LVTTL signals, i.e., 0 to 3.3V. Some of its pins have already been assigned by the manufacturer of the BASYS board for a specific task. For example, some pins provide the supply voltage and ground to the FPGA. We cannot and do not need to assign these in our Verilog modules.

A second group of pins has been connected to various inputs (switches, push buttons and clock signal) and outputs (LEDs, Display, VGA connector) We will make use of these. Finally, a third group of pins has been connected to connectors that can act either as inputs or outputs; there are 4 six-pin connectors at the top (labeled JA, JB, JC and JD) and pins 1 through 4 can be assigned to be either outputs or inputs. We will use these later to interface the board to the scope or other devices.
Exercise 1: Simple I/O Verilog Module

To illustrate the topics covered so far, study the module SimpleIO shown below:

```verbatim
module SimpleIO(a, q);
    (* LOC = "P69" *) input a; //use the push button labeled BTN0 at pin P69 as input a
    (* LOC = "P15" *) output q; //use the LED labeled LD0 at pin P15 for the output q
    assign q = a; //assign whatever value input a has to output q
endmodule
```

Again, a graphical representation of the module is given below:

```
SimpleIO

  a  q

```

The module has one input, a, and one output q. The input and output have been assigned to the FPGA pins connected to the right most push button labeled BTN0 and the rightmost LED labeled LD0, respectively. Furthermore, the input has been (in the Verilog module through the LOC statement) connected to the output so pushing BTN0 will illuminate LD0.

Let’s implement this module. For your reference, we have collected the series of steps necessary to create a new project in the programming software in Appendix J. Using the steps in Appendix J.1, create a new project in your personal area called “SimpleIO”. Then, use the steps in Appendix J.2 to make a new source file. Modify the generated skeleton source code (including the “module” statement) and add the statements above. Finally use the steps in Appendix J.3 to synthesize your design and download the bit-file to the FPGA using the Digilent Adept facility. This will be your first assignment. Check that it indeed works and hand in your code.

Write-up

9.1.1. Hand in the Verilog source code needed to turn on LD0 by pushing BTN0. For this and all the subsequent exercises, you must hand in the documented code for all the modules that you used. (Documented means you must add comment statements explaining the purpose of each module and the purpose of the various inputs and outputs as well as the wires and registers within the module.)

9.2. Combinational Logic: 4 Bit Full Adder

In this section you will build a full 4 bit adder. You will learn how to implement combinational logic in Verilog, how to define wires and buses and how to instantiate modules.

Combinational Logic Introduction

The output of a combinational logic circuit depends only on its current input. Therefore, it involves only logic gates that have no memory, i.e., they are amnesic. Typical combinational logic gates are AND, NAND, NOR, OR, NOR and XOR gates. In this exercise you will learn how to use Verilog operators to implement these gates to rebuild your full one bit adder from section 7.3.

Combinational Logic: assign-Statement

Combinational logic is implemented in Verilog by using the assign keyword and the blocking assignment operator represented by an equal sign. Assignments often use Verilog bit operators. They are identical to the ones used in C and are listed below:
### Bit Operator Notation

<table>
<thead>
<tr>
<th>Bit Operator</th>
<th>Notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND</td>
<td>&amp;</td>
</tr>
<tr>
<td>OR</td>
<td></td>
</tr>
<tr>
<td>NOT</td>
<td>! (or ~)</td>
</tr>
<tr>
<td>XOR</td>
<td>^</td>
</tr>
<tr>
<td>Is Equal</td>
<td>==</td>
</tr>
<tr>
<td>Is Not Equal</td>
<td>!=</td>
</tr>
<tr>
<td>Right Shift (by n bits)</td>
<td>&gt;&gt; n</td>
</tr>
<tr>
<td>Left Shift (by n bits)</td>
<td>&lt;&lt; n</td>
</tr>
</tbody>
</table>

Table 9.1.

For example, an AND operation between $a$ and $b$ whose result would be assigned to $q$ would be written in Verilog as:

```verilog
assign q = a & b;
```

Similar to C, multiple operations can be combined in one statement like in the example shown below:

```verilog
assign p = (a & b) ^ !c;
```

An if-statement can be implemented in combinational logic as a conditional assignment using the following notation:

```verilog
assign q = c ? a : b;
```

In this case, if $c$ is true, $a$ will be assigned to $q$; else, $b$ will be assigned to $q$. (Note: this is a one bit multiplexer.)

Another form of this construct that you may encounter is shown below:

```verilog
assign q = (a_input == 3) ? x : y;
```

In this case, $x$ will be assigned to $q$ if $a\text{\_input}$ is 3; else $y$ is assigned to $q$.

### Exercise 1: One Bit Full Adder

Throughout this section you will be building a very basic calculator that ultimately is capable of adding 4 bit numbers. Though the Verilog syntax contains basic mathematical operators, such as addition, subtraction and multiplication, in this exercise you will not make use of them. Instead you will design the adder circuit using the basic logic gates or bit wise operators described earlier. (In case you wondered, this is also how Verilog implements its mathematical operators but it is done in the background, completely hidden from the programmer.)

Create a new Verilog project with a module called MyFullOneBitAdder. The graphical representation of the module is shown in the previous diagram and the table below lists the relevant inputs and outputs and their functions. In the last column of the table below, fill in the corresponding pin numbers from the DigilentInc documentation in Appendix L. Use this information in your Verilog module to assign the inputs and outputs to the appropriate pins using (*) LOC = … *) statements.
<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>BASYS Control</th>
<th>Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Input</td>
<td>Switch SW0</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>Input</td>
<td>Switch SW4</td>
<td></td>
</tr>
<tr>
<td>c_{in}</td>
<td>Carry-in Input</td>
<td>Switch SW7</td>
<td></td>
</tr>
<tr>
<td>q</td>
<td>Output, i.e., a+b+c_{in}</td>
<td>LED LD0</td>
<td></td>
</tr>
<tr>
<td>c_{out}</td>
<td>Carry-out Output</td>
<td>LED LD1</td>
<td></td>
</tr>
</tbody>
</table>

Table 9.2.

Now that you have all the inputs and outputs defined and assigned to the appropriate pins, you need to add the combinational logic statements for the one bit full adder. Find your write-up from section 7.3. Specifically, use the `assign` keyword and convert the logic expression you obtained for 7.3.4., into Verilog syntax using the Verilog bit operators shown in Table 9.1.

Compile your code and download the bit file into the FPGA. Test your one bit adder by operating switches SW0, SW4 and SW7 and check the corresponding LEDs. If everything works, hand in your code.

**Write-up**

9.2.1. Hand in the Verilog source code for your full one bit adder.

**Instantiating a Module**

Instantiating a module is similar to C function calls: once you have a (functioning) module you can use it repeatedly in your code by “instantiating” it. For example, in the previous section you built a working one bit full adder. By “instantiating” it multiple times we can extend it easily into a 4 or even 128-bit full adder!

If that still sounds too theoretical, think of your original module as a blueprint; “instantiating” it creates a working “copy” of the blueprint.

To illustrate this concept, let’s instantiate the previously built one bit full adder twice, i.e., create a two bit full adder that adds the 2 bit word composed of \(a_1a_0\) to the two bit word composed of \(b_1b_0\). (By convention the index 0 indicates the least significant bit of the word.) The result of the addition is then assigned to the three bit output \(c_{out}q_1q_0\). Such a circuit would be able to add the decimal values, for example, of 3 and 2. See its graphical representation below.
The corresponding Verilog statements (minus the pin assignment statements) are:

```verilog
module MyFullTwoBitAdder(a0, a1, b0, b1, q0, q1, c_out);
    input a0, a1, b0, b1; //Note: all pin assignments were omitted for sake of brevity!
    output q0, q1, c_out;

    //Instantiate the full one bit adders:
    MyFullOneBitAdder FullOneBitAdder0(a0, b0, 0, q0, c_out0);
    MyFullOneBitAdder FullOneBitAdder1(a1, b1, c_out0, q1, c_out);
endmodule
```

The statements performing the instantiation of the module `MyFullOneBitAdder` are shown in bold print and it uses the following syntax:

```verilog
    OriginalModuleName InstantiatedModuleName(inputs..., outputs...);
```

Similar to C, the names of the inputs and outputs do not have to be identical to the ones used in the original module but their order, number and type must be preserved! Also, inputs and outputs can be implicitly connected by appropriately naming the inputs and outputs. For example, the `c_out` output in the `FullOneBitAdder0` (instantiation) has been implicitly connected to the `c_in` input of the `FullOneBitAdder1` (instantiation) through the “variable” `c_out0`. (Note: “wire” would be the proper term here, not “variable;” since we have not yet defined what constitutes a wire in Verilog we call them - at least for now - “variables.”)

Also note that the module `MyFullOneBitAdder` was instantiated twice. It could have been instantiated many more times; the only requirement is that each new instantiation has a new, unique name.

A practical issue: how does the compiler know where to locate the module that is to be instantiated? Two different solutions to tackle this issue exist:

a) You can combine all your modules in a single file, one listed below the other. If you choose this approach then you will have to inform the Xilinx ISE program which one you want to use.
as your Top Module: in the upper right panel of the Xilinx ISE Program “Sources” window find the module that you want to be the Top Module; right click on it and select “Set as Top Module.”

b) If you prefer, you can keep each module in a separate file. In this case each file must be added to the project by selecting in the menu bar: Project / Add Source. Again, with multiple modules you need to specify which one represents your Top Module. How this is done has been explained in the instructions in the previous paragraph.

Once a module has been instantiated, Xilinx ISE Webpack adds a plus sign to the Top Module in the project Sources window. See the example below.

![Image of Plus Sign in Sources Window]

Clicking on the plus sign lists the instantiated modules in a tree like fashion. Double clicking on a module name will display its code.

A final comment on the instantiation process concerns the pin assignment information using the LOC statement discussed in the previous section: avoid placing any pin assignments in modules that will be instantiated! Instead, always place all your pin assignment statements in the Top Module!

The reason for this is as follows: each time a module is instantiated a new set of inputs and outputs are created. Therefore, had the inputs and outputs already been assigned to a specific set of pins, instantiating such a module multiple times, which is completely reasonable, would tie the inputs or outputs of each instantiated module to the same pins. This clearly would not be work. (If you’re still not convinced try to visualize what would happen to the outputs of a module instantiated multiple times with all of the outputs connected to the same pin. Clearly the outputs would short each other out.)

Therefore, always place your pin assignment statements in the Top Module! To avoid confusion, it is a good practice to remove all LOC statements from modules that will be instantiated. Warning: failing to remove the LOC statements from (sub) modules does not produce any error messages by the Xilinx compiler when it generates the program files and it usually results in code that simply will not work!

**Exercise 2: 4-Bit Full Adder**

Use the 2 bit adder example and build a complete 4 bit full adder in Verilog by instantiating your previously build full one bit adder module multiple times. See the graphical representation of the 4 bit adder below.
Consider $a_0$ and $b_0$ to be the LSB and $a_3$ and $b_3$ the MSB of each 4 bit term. You may permanently set $c_{\text{in}}$ to 0 using either an assign statement or by setting corresponding input of the instantiated module to 0. Use the table below to assign the inputs and outputs to the correct pins.

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>BASYS Control</th>
<th>Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_0$</td>
<td>Input: LSB of $a$</td>
<td>Switch SW0</td>
<td></td>
</tr>
<tr>
<td>$a_1$</td>
<td>Input: 1st bit of $a$</td>
<td>Switch SW1</td>
<td></td>
</tr>
<tr>
<td>$a_2$</td>
<td>Input: 2nd bit of $a$</td>
<td>Switch SW2</td>
<td></td>
</tr>
<tr>
<td>$a_3$</td>
<td>Input: MSB of $a$</td>
<td>Switch SW3</td>
<td></td>
</tr>
<tr>
<td>$b_0$</td>
<td>Input: LSB of $b$</td>
<td>Switch SW4</td>
<td></td>
</tr>
<tr>
<td>$b_1$</td>
<td>Input: 1st bit of $b$</td>
<td>Switch SW5</td>
<td></td>
</tr>
<tr>
<td>$b_2$</td>
<td>Input: 2nd bit of $b$</td>
<td>Switch SW6</td>
<td></td>
</tr>
<tr>
<td>$b_3$</td>
<td>Input: MSB bit of $b$</td>
<td>Switch SW7</td>
<td></td>
</tr>
<tr>
<td>$q_0$</td>
<td>Result: LSB bit of $q$</td>
<td>LED LD0</td>
<td></td>
</tr>
<tr>
<td>$q_1$</td>
<td>Result: 1st bit of $q$</td>
<td>LED LD1</td>
<td></td>
</tr>
<tr>
<td>$q_2$</td>
<td>Result: 2nd bit of $q$</td>
<td>LED LD2</td>
<td></td>
</tr>
<tr>
<td>$q_3$</td>
<td>Result: 3rd bit of $q$</td>
<td>LED LD3</td>
<td></td>
</tr>
<tr>
<td>$c_{\text{out}}$</td>
<td>Carry-out and MSB of Result</td>
<td>LED LD4</td>
<td></td>
</tr>
</tbody>
</table>

Table 9.2.

Write your code and download it to the BASYS board. Test it. Note that the 4 right most switches represent term $a$ while the 4 left most switches represent $b$. The LEDs should then directly correspond to the result that the addition of the two terms produces.

Write-up

9.2.2. Hand in the Verilog source code for your full four bit adder.

Wires and Buses

For the final exercise in this section you will make two improvements to your previous project. First, you will use the 7-Segment LED display to show the addition results directly in decimal notation. This will require the instantiation of a display module provided to you, courtesy MXP labs. The second improvement concerns the large number of inputs and outputs that our designs are growing into. Clearly, if we were to expand our 4 bit full adder into a 16 or 64 bit adder, the current approach of specifying each input and output wire is inefficient. This problem is remedied by the introduction of the Verilog concept of buses.

The definition of a bus, also sometimes called a vector, is that it represents a collection of wires. (Again an analogy exists with C in the concept of an array being a collection of variables.) But this leads us to ask, what exactly do we mean by wires?
In Verilog, as in reality, wires do not maintain information. They only transmit it from an output to another input. Wires can be explicitly declared in Verilog through the `wire` keyword. (We will discuss this in more detail below.) However, when inputs or outputs are used without further specifications, Verilog implicitly assumes that they are wires. For example, in the `FullTwoBitAdder` module shown previously the wire `c_out0` was used as an input and output. Because it was not explicitly declared, Verilog assumed (correctly) it to be a wire and not something else.

If a wire is not part of an input or output then it must be explicitly declared, using the keyword `wire`, before it can be used in an `assign` operation. See the example below.

```verilog
wire a, b;
wire wLO = 0; //declared and assigned to a permanent value at the same time.
assign a = b; // assign b to a
```

You will find that declaring wires, even when it is not required, will make your code more readable.

Buses, on the other hand, must always be declared. (If you neglect to declare the bus (size), Verilog assumes it to be a wire!) See the bus declaration examples below:

```verilog
wire [3:0] a; //A 4 wire bus consisting of a[0], a[1], a[2] and a[3].
wire [7:0] x = 255; //an 8 bit wire bus permanently assigned to decimal 255 or // binary 1111 1111
```

When a bus is used as an input or output, the declaration can often be combined with the input / output declaration as shown below in a modified version of the previous `MyFullTwoBitAdder` module:

```verilog
module MyBusFullTwoBitAdder(a, b, q);
    input [1:0] a, b; //2 bit input buses
    output [2:0] q; //3 bit output bus
    MyFullOneBitAdder0 FullOneBitAdder0(a[0], b[0], 0, q[0], c_out0);
    MyFullOneBitAdder1 FullOneBitAdder1(a[1], b[1], c_out0, q[1], c_out1);
    assign q[2] = c_out1;
endmodule
```

This module shows how individual wires within a bus are accessed by using the bus name and a bracket, similar to the C-notation for array elements. The graphical representation of the module is given below. Note, that to distinguish the buses from wires, buses are drawn with wide or bold lines.

![MyBusFullTwoBitAdder](image)

Wires and other buses can also be combined into larger buses as shown in the example below. Note the required curly brackets required when combining, i.e., concatenating, bits!
A lot of new material has been covered. Before we go on to the next exercise, let's see how we can simplify our two bit adder design some more by using a bus called Switches, to control the inputs, and a second bus called LEDs, to control the outputs. See its graphical representations below:

MyBusFullTwoBitAdderV1

Its implementation, including the pin assignment statements, is shown in the module below. Note how the LOC statement has been used to assign pins to an entire bus.

module MyBusFullTwoBitAdderV1(Switches, LEDs);
  //control numerical inputs
  (* LOC = "P6 P10 P12 P18 P24 P29 P36 P38" *) input [7:0] Switches;
  //display result of the addition
  (* LOC = "P2 P3 P4 P5 P7 P8 P14 P15" *) output [7:0] LEDs;
  wire [1:0] a, b; //2 bit input buses
  wire [2:0] q;   //3 bit output bus
  assign a = Switches[1:0]; //input a
  assign b = Switches[5:4]; //input b
  // Note Switches 2, 3, 6 and 7 are not being used
  // in this particular exercise.
  assign LEDs = {5'b00000, q}; //send result to LED (NOTE: the notation 5'b00000
  //stands for five (binary) bits all set to 0)
  //It will be explained more in the next section.
  MyFullOneBitAdder FullOneBitAdder0(a[0], b[0], 0, q[0], c_out0);
  MyFullOneBitAdder FullOneBitAdder1(a[1], b[1], c_out0, q[1], c_out1);
  assign q[2] = c_out1;
endmodule

Exercise 3: 4 Bit Full Adder Using Buses

Use the 2 bit adder example above and build a similar 4 bit full adder in Verilog using buses. See the graphical representation of the 4 bit adder shown below.

MyBusFullFourBitAdderV1

Use a bus for the input Switches and one for the output LEDs. Use the LOC statements shown in the module above.

Declare 4 wire buses for $a$ and $b$; assign the 4 right most switches to $a$ and the remaining ones to $b$. Declare a 5 wire bus for the result, $q$, and assign it to the LEDs output bus.
Write-up

9.2.3. Hand in the Verilog source code for your full four bit adder with buses.

Exercise 4: 4-Bit Full Adder Using Buses and 7 Segment Display

For the last exercise in this section, we turn your 4 bit adder design into a crude calculator. We will instantiate an already written module that will display the result of the addition in decimal notation on the 7 segment displays.

Copy the file HexDisplayV1.v from the folder U:\pub_MXP\Verilog\HexDisplay to your current working Xilinx project directory. To do so, go to the menu bar and select Project \ Add Source and then select the HexDisplayV1.v file and copy / paste it. (Note: you must copy the file into your working directory; linking to the (read-only) file in the pub_MXP directory will not work!)

The file should appear in the “Sources” window. Double click on it to open it. You will notice that it contains multiple modules. The one that is of interest to us is the very first one, called HexDisplayV1. (The other modules are support modules that will be instantiated by HexDisplayV1.) It has the following inputs and outputs:

<table>
<thead>
<tr>
<th>Name</th>
<th>I/O</th>
<th>Bus Size (Bits)</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_clk</td>
<td>Input</td>
<td>1</td>
<td>25 MHz BASYS board system clock; required for updating display.</td>
</tr>
<tr>
<td>value_in</td>
<td>Input</td>
<td>16</td>
<td>Binary value to be displayed.</td>
</tr>
<tr>
<td>BCD_enable</td>
<td>Input</td>
<td>1</td>
<td>If this is set to 1, the display is in decimal format, else, it is in hexadecimal.</td>
</tr>
<tr>
<td>Display_Enable</td>
<td>Input</td>
<td>1</td>
<td>If it is 1, the display is turned on or lit; else it is turned off.</td>
</tr>
<tr>
<td>sevenSegLED_out</td>
<td>Output</td>
<td>7</td>
<td>Output signal to the actual 7 segments of each display.</td>
</tr>
<tr>
<td>sevenSegPos_out</td>
<td>Output</td>
<td>4</td>
<td>Output signal to turn one of the 4 displays on.</td>
</tr>
</tbody>
</table>

Modify a copy of your latest version of the 4 bit full adder program so that it represents the graphical representation shown below.

Tips:

FPGA & VERILOG Part I / 9.2. Combinational Logic: 4 Bit Full Adder 101
The final module has 2 inputs and 3 outputs. Of the inputs, only sys_clk is new and it will provide the 25 MHz clock signal to the instantiated HexDisplayV1 module. Add two new buses to the existing output: one for sevenSegLED_out and sevenSegPOs_out. These two buses are directly connected to the pins of the seven segment display. Pin assignments for the three new buses / signals can be found in Appendix K or by studying the first few lines of code in the HexDisplayV1 module.

Compile your code and download the bit file into the FPGA. Test your crude calculator bit operating switches SW0 through SW7 and check the result in the display.

Write-up

9.2.4. Hand in the Verilog source code for your full one bit adder calculator with the HEX display.

Additional Comments: Conditional Assignments Using Buses

Before we race to the next Verilog topic, here are some comments concerning some topics that might be of interest to you in future projects.

In the previous section, we briefly covered the conditional assignment statement using 1 bit wires. With buses, more complex logic conditions, similar to a C if – if else statement, can be implemented this way:

```verilog
wire [11:0] y;
wire [3:0] q;
wire [1:0] c;
assign q = (c == 2'b00) ? y[3:0]: //NOTE Notation: 2'b10
             (c == 2'b01) ? y[7:4]: //2 stands for the number of bits to be assigned
             (c == 2'b10) ? y[11:8]: //'b stands for binary notation
             4'b1111; //10 represents that actual assignment, i.e., 2
```

In this example, when c equals 0, q is set equal to the four least significant bits of (bus) y. When c is 1, the 4 next larger bits are assigned to q, etc. If none of the conditions happens to be met, a default condition, the last statement, is executed and q will be assigned a (decimal) value of 15.

(Note the Verilog notation for assigning values, such as 4'b1111. The first number, 4, represents the number of bits that will be assigned. The value after the single quote stands for the notation of the values following, with ‘b indicating binary, (‘h hex and ‘d for decimal.) Finally, the four ones represent the actual bit values.)

9.3. Sequential Logic: Registers

In sequential logic circuits, the output depends not only on the present input but also on the history of the input. Digital logic applications would be very limited if they were restricted only to combinational logic devices. It is the joining of combinational logic devices with devices that retain their memory that has made digital logic so versatile and ubiquitous in our lives.

The most familiar sequential logic device is the D-type Flip-Flop, a one bit memory device, shown below.
It has two inputs. The \textit{D}-input is for the data while the \textit{clock} input instructs the device when to load and subsequently retain the data. The stored data is accessible at all times through the \textit{Q} output.

The Verilog equivalent of a D-Type Flip-Flop is a \texttt{register}. It is declared with the \texttt{reg} keyword. Similar to wire buses, a collection of registers can be declared to extend its size from 1 bit to multiple bits. The size limit of these collections is only limited by the available memory in the FPGA. With the currently used FPGA on the BASYS board the sum of all registers cannot exceed 8000 (bits.)

Below are two examples that show how to declare and initialize registers. The first line declares a single bit register while the second one declares a 16 bit collection and then initializes them all to 0.

\begin{verbatim}
reg little_r; //one bit registry
reg [15:0] big_r = 0; //16 bit registry, all initialized to 0
\end{verbatim}

The clock function of a Flip-Flop is executed in Verilog through the \texttt{always @} keyword and the statements directly following it, also sometimes called the \texttt{always block}. See the example below:

\begin{verbatim}
reg q = 0;
always @(posedge clk_in) begin
  q <= ~q;
end
\end{verbatim}

In this code segment the \texttt{always block} consists of three lines, highlighted in bold type. The first line of the always block specifies that all statements between “\texttt{begin}” and “\texttt{end}” will be executed at every positive edge of the clock signal called \texttt{clk\_in}. In this particular example, the one bit register named \texttt{q} is negated each time, i.e., it is being toggled.

Note that the clock input signal must always be preceded by the \texttt{posedge} or \texttt{negedge} keyword. Completely omitting these edge descriptors implies dual triggering, i.e., triggering on both the positive as well as the negative edge of the input clock signal. This should be avoided because the FPGA used on the BASYS boards is not capable of dual triggering. Therefore, always include either the positive or the negative edge descriptor with the clock signal!

If you carefully studied the example above, you may have noticed a new Verilog operator: \texttt{<=}. It is called a non-blocking assignment operator - as opposed to the blocking assignment operator previously encountered in the combinational logic section. The non-blocking operator is used when assigning something to a register in an always block.

For now do not worry if you feel confused about the blocking and non-blocking operators. We will have more to say about that topic at the end of this chapter. For now try to think of it this way:

- if you use combinational logic, for example if you assign something to a wire, then you use the \texttt{assign} keyword and the blocking operator, i.e., the equal sign;
- if you use sequential logic, for example if you assign something to a register in an always block, then you should use the non-blocking assignment operator, i.e., \texttt{<=}; in this case also omit the keyword \texttt{assign}. 

D-Type Flip-Flop
Let's look at another example. The module below mimics the simple positive edge triggered D-type Flip-Flop discussed earlier. See if you can understand it entirely.

```verilog
module DTypeSimple( d_in, clk_in, q_out);
    input d_in, clk_in;
    output reg q_out;    //Note how the output and the register specification has been
    //combined into one statement.
    always@(posedge clk_in) begin
        q_out <= d_in;
    end
endmodule
```

The next and also last example of this section illustrates a more complex D-type Flip-Flop. (See the figure below.)

```
module DType( d_in, clk_in, set_in, reset_in, q_out, qnot_out);
    input d_in, clk_in, set_in, reset_in;
    output q_out, qnot_out;
    reg q_out;
    assign qnot_out = !q_out;
    always@(posedge clk_in or negedge set_in or negedge reset_in)begin
        if( set_in == 0) begin
            q_out <= 1;
        end
        else if( reset_in == 0) begin
            q_out <= 0;
        end
        else begin
            q_out <= d_in;
        end
    end
endmodule
```

In addition to the inputs and outputs previously discussed, this D-Type Flip-Flop has negative edge triggered asynchronous *set* and *reset* inputs and complimentary q and !q outputs. Study its Verilog code shown below.

Notice the *clock*, *set* and *reset* inputs: *clk_in*, *set_in* and *reset_in*. They are asynchronous, i.e., they occur independently of each other. Therefore, each of them must be individually listed in the always block's clock signal list, separated by the *or* keyword.

Since every one of them is able to activate the always block, conditional statements in the form of *if*-statements are used to discern which input signal has activated the always block and also what actions are to be taken.

Again compare the sequential logic's branching instructions with those of the combinational logic. While *if* / *else if* statements can be used in sequential logic statements, in combinational logic you must use the conditional assignment operator *c ? a : b* discussed earlier.
Finally, note that the `begin` and `end` keywords function just like the curly brackets in C and, similarly, are optional if they enclose only one statement.

### 9.3.1. Exercise 1: Binary Counter

In addition to storing information, sequential logic devices provide the core of counters and shift registers. Lack of time prevents us to study the use of shift registers which can be found in parallel to serial converters, mathematical logic units and random number generators, to mention a few. Instead, in the following exercises, you will start using the Verilog registers to build various counter applications.

Verilog makes the design of multi-bit counters easy. It includes an addition (or subtraction) operator that increment (or decrements) a register at the transition of a clock input. In the example below, the counter is incremented each time at the rising edge of the `clk` input.

```verilog
reg [3:0] count;  //4 bit register declaration
always @(posedge clk) begin
    count<=count+1;
end
```

So far we have not paid much attention to the “quality” of the input signals and only requested that they adhere to the digital logic standard specified for the BASYS board. Nevertheless, when working with the `clk` signal that controls the sequential logic, such as the one shown in the always-block of the binary counter example above, it is extremely important that it has a clean transition from LO to HI, or vice versa. Physical devices, such as buttons and switches, are especially prone to produce noisy switching signals. In other words, each time you press a push button, instead of a single, clean, low to high transition, multiple such transitions are created as the switch “makes” or “breaks” contact. Various methods for removing this extremely annoying (and very common) effect using hardware and software exist; it is called “debouncing” a signal. For the hardware discussion see H&H pages 576 to 577; for the software solution see the Xilinx compiler under the menu: Edit / Language Templates / Verilog / Synthesis Constructs / Coding Examples / Misc / One-Shot Debounce Circuit.

Though the noisy transitions may only last for a few milliseconds, with the fast digital logic in the FPGA circuits, it can alter the sequential logic results. As you will see, when you implement the example shown above using a push button, the result will be non-sequential counting where the erroneous counts are due to “bounces” of the mechanical button.

Earlier versions of Xilinx did not particularly care about the origin and quality of the `clk` signal used in sequential logic designs. It assumed that the programmer had either chosen a “clean” input signal, such as originates from a digital clock, or that the signal had been debounced appropriately. Unfortunately, with Xilinx Webpack ISE version 12 (and later), i.e., the version we are using in the lab, the compiler began to investigate the pedigree of the (clock) signal used to drive sequential logic circuits. If such a signal originates anywhere else but at the designated digital clock pins, pin 54 and 53 on our BASYS board, Xilinx refuses to compile the code due to the lowly pedigree of the clock signal. Instead it retaliates with the cryptic error message: “1018 – A clokc IOB / clock component...”

The work around is simple: we need to inform the compiler to stop being such a snob for such sequential logic `clk` input pins. By preceding our pin assignments, with `CLOCK_DEDICATED_ROUTE = "FALSE"` we will be able to force the compiler to scale the error message down to a warning, albeit a stern one. Study the example below, where the `clk` input has been assigned to BTN0, pin 69, on the BASYS board.
module BinaryCounter( clk, LEDs);
(* CLOCK_DEDICATED_ROUTE = "FALSE", LOC = "P69" *) input clk; //turn off clk error msg
(* LOC = "P2 P3 P4 P5" *) output [3:0] LEDs
reg [3:0] count; //4 bit register declaration
always @(posedge clk) begin
    count<=count+1;
end
assign LEDs = count;
endmodule

Note that you need the CLOCK_DEDICATED_ROUTE = "FALSE" statement only for inputs that control sequential logic, i.e., the signal following the posedge or negedge statement in the always block!

Finally, for the assignment, in the Xilinx ISE Webpack, create a complete 4 bit binary counter module. Assign the counter outputs to the LEDs LD0 (LSB) through LD3 (MSB.) Use Switch SW0 as your clock input to increment the counter. By operating SW0 you should observe that the counter increments in binary: 0000, 0001, 0010, 0011, 0100 and so on.

Now see what happens to your counter if you use a push button instead of the switch button as your clock signal. Change the clock input in your Verilog program to one of the push buttons on the BASYS board, for example, pin 69. Press the push button and keep track of the counts displayed on by the LEDs. Does your counter occasionally seem to skip some digits? What you are observing is the effect of a push button which has not been properly "debounced."

Write-up

13.3.1. Hand in the Verilog source code for your 4 bit counter.

9.3.2. Exercise II: Decade Counter (Optional)

An n-bit binary (up) counter will count up to $2^n - 1$ and then starts the cycle at 0 again. On the other hand, a decade counter resets itself after 10 cycles. It counts up to 9 and then returns to 0. In other words, it never reaches 10, or for that matter, 11 through 15.

DecadeCounter

<table>
<thead>
<tr>
<th>clock</th>
<th>s_out</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>count(3:0)</td>
</tr>
</tbody>
</table>

Such a counter module is useful. For example, if we were to connect multiple instances together (as shown below) we would have a counter that counts in decimal, or rather in binary coded decimals, BCD, instead of binary. Hence, we no longer would need a binary to decimal conversion module.
Build a (single) decade counter with a clock input, a 4 bit count output and a c_out output. The c_out output goes high for one cycle when the counter is reset to 0. It can be used as clock input for the next higher digit, as shown in the picture above. (Hint: To reset the counter and to set c_out HI, you will need an if-statement somewhere in the always block.)

Instantiate the decade counter in a separate module and use SW0 as the clock input. Display the count and c_out output using LEDs LD0 through LD4. Check that it works and that it resets itself always after 10 consecutive clock cycles.

**Write-up**

9.3.2. Hand in the code for the decade counter.

**9.3.3. Exercise III: Seconds Clock**

Besides its obvious function to keep track of counts, counters are also used to generate timing signals that are slower than their clock input. These timing signals then drive other devices at the new slower frequency.

(In terms of circuit design, you may note the similarity between the counter and the (resistive) voltage divider’s usage. While one operates in the voltage domain, the other operates in the time (or frequency) domain. Both circuits are used to generate signals that are fractions of a reference signal.)

In this exercise you will use the BASYS board’s 25 MHz clock to build a 1 second reference timer. For now, this 1 second timer will be used to turn an LED on every other second for 1 second. In the next chapter, it will be used as a reference timer for a radiation counter monitor.

Use the 25 MHz signal from Pin 54 on the BASYS board and feed it into a n-bit counter. Design the counter so that each time the counter reaches a value corresponding to an elapsed time interval of 1 second, it resets itself to 0. Simultaneously, on each reset, it toggles an output signal, q_out. Assign this output signal to an LED and check that your design works, i.e., every other second, the LED should be lit for exactly one second.
Hints: First “calculate” the number of 25 MHz clock cycles that elapse in a second. Since your counter will have to count up to this number, determine the number of bits required to hold such a large number. Declare the counter register with the required number of bits. Next, in your always block, use if-statements to determine if the counter has reached its 1 second limit and needs to be reset or if it should be incremented further. On each reset, toggle another, separate register which serves to control the LED.

Write-up

9.3.3. Hand in the code for your working second reference clock.

9.4. Radiation Monitor / Counter

In this section we will “recycle” some of the previous modules to build a “pretty good” radiation survey instrument that displays the number of ionizing events measured in a one second time interval.

9.4.1. Radiation Detectors

The radiation detector, a Model RM 60, was manufactured by Aware Electronics. It utilizes a “cigar shaped” (as opposed to a pan-cake shaped) Geiger Mueller (GM) tube to detect ionizing particles. Each time the gas in the GM tube has been ionized by a particle passing through it, the unit generates a 50 μsec negative going LVTTL pulse that is being sent to the BASYS board. It is these pulses that we want to count when they trigger our radiation counter.

Up to four RM 60 units can be powered and interfaced to the BASYS board through the edge connectors JA through JD as shown below.
Figure 9.4. RM 60 Radiation Monitor with Thorium Oxide gas mantle and BASYS board. The RM 60 has been connected to pin 1 of the BASYS board’s JA edge connector which corresponds to pin P81 on the FPGA.

The labels on top of the connector jacks indicate the pin number of the edge connector receiving the pulse signal. For this exercise, you will use only one detector. Connect it as shown in the picture above and place it near a gas mantle. (Please do not unpack the gas mantels.) These gas mantels contain Thorium Oxide. The Thorium itself emits a very low level of radiation as it decays through along decay chain into lead. It is these decays that we will measure.

9.4.2. Step 1: Basic Continuous Counter

In this exercise you will build on the Verilog code that has already been written and tested by you in previous sections. For this first step, you will start out with a very primitive radiation monitor. In future steps, you will adjust or amend your code to improve your design.

Start a new project and implement the simple counting circuit with the 7 segment display shown below.
Before you instantiate your binary counter module from exercise 9.3.1., you probably want to expand its size from a 4 bit counter to a 16 bit counter. Once you have done that, connect it to an instance of the HexDisplayV1 module which you have used previously in exercise 4, section 9.2.

Once everything works, you should be able to observe the ionization events as the counter keeps continuously incrementing. (Of course you must connect the radiation modules to the BASYS board to observe anything.) From looking at the count rate, what do you estimate the count rate per second to be? (Do not use detailed calculations; instead, just look at your counter and notice which digits are updated about every second.)

### 9.4.3. Step 2: Binary Counter with Asynchronous Reset

The previous version of the counter was displaying a continuous count of ionization events. What we really would like to observe is the number of counts in a given time period, for example, counts per second.

This can be achieved with two modifications / additions:

a) Add a reference time source that sends out a “trigger” signal every second.
b) Reset your counter at the beginning of each reference time period. Since the reset and clock signals are independent of each other, this is considered an asynchronous reset. (A synchronous reset would occur always at the same time as the clock signal.)

For a graphical representation on how to implement these ideas, see below.

Implementing part a) is trivial since you have already built a one second reference timer in exercise 9.3.3. You will instantiate in your code and use its output signal to reset your radiation events counter.

Part b) can be realized by modifying your existing binary counter that keeps track of the radiation events. First, add a reset input to the existing module.

Second, since the reset input is asynchronous it needs to be added to the list of clock signals in the always @ block. See the example below for a positive edge reset signal.
Third, since there are now two “clock” signals, clock and reset, an if-statement is required to discern the source of that activated the always block. Specifically, if reset is HI, then the counter is reset to 0, else the counter is incremented by one. Add these changes to your binary counter module.

Instantiate your new binary counter and the one second clock module and connect them as shown in the graphical representation above. Compile your code, load it into the BASYS board and watch its operation. (If it doesn’t work correctly, for debugging, you may want to connect the 1 second clock output to LED LD0 on the BASYS board.) Is the maximum count value close to what you estimated in the previous section?

9.4.4. Step 3: Binary Counter with Asynchronous Reset and Storage Latch

The design still suffers from one flaw: though the counter now resets every second, it still displays a continuously incrementing display. What we are really interested in is only the count value right before it is reset. In other words, we need to store the count value in memory at the very same instant as the counter is reset.

You may add the code for the latching memory module directly to your existing module, or, you may create a stand-alone module (shown below) and then instantiate it in your code. The choice is yours.

The storage latch module consist of an always block that stores the counts_in input in a register of equal size (named counts_out) at each positive edge of the store signal. (Warning: it is tempting to add the code for assigning counts_out to counts_in directly to the binary counter’s module always block, specifically, to the section that handles the asynchronous reset. This will not work! You must create a separate always block (as shown above) with only one clock source that handles the store process.)

The complete circuit with the storage latch is shown below:

Test your new circuit. Are the displayed values similar to the previously observed maximum values?
9.4.5. Step 4: Radiation Monitor Version with Improved “Dead Time”

The time interval during which an instrument is not able to measure or process its input signal is called “dead time.” Dead time occurs when an instrument is busy with other (necessary) tasks. Examples of such tasks are the storing or processing of the acquired data, processing user input or updating its display.

The dead time is usually specified (in percents) as the ratio of dead time to the sum of dead time and “live” time. (“Live” time represents the time during which the instrument is measuring something.)

Percentage dead time = \( \frac{t_{\text{dead}}}{t_{\text{live}} + t_{\text{dead}}} \)

A “good” instrument is one that has a low dead time ratio. A low dead time ratio assures:

a) That one obtains the same amount of data in a shorter overall time interval than if one were to use an instrument with a large dead time ratio;

b) That you will not miss any important events that may occur during the dead time interval.

Let’s apply these concepts to your radiation counter. Note that it only counts radiation events every other second. Can you figure out why? (Hint: make a truth table for your binary counter with the asynchronous reset. Specifically, check what happens when reset remains HI.) So what is the percentage dead time of your current circuit?

We can decrease the dead time (ratio) of the radiation counter to almost 0 by decreasing the time interval that one second timer’s reset signal stays HI. Change the design of the one second timer. Instead of toggling \( q_{\text{out}} \) and keeping it high for an entire second, keep it HI for only one 25 MHz cycle when the second counter is being reset. (After doing that, you may also have to adjust the 1 second maximum count value by one. Why?) Implement these changes and check that your counter still works. If it does, it should now be “a pretty good” radiation counter.

Write-up

9.4.1. Hand in the documented code for all your modules that you used to create the counter in step 4. (Documented means you must add comment statements explaining the purpose of each module and the purpose of the various inputs and outputs as well as the wires and registers within the module.)

9.4.2. What count rate did you observe with the gas mantle positioned close by the detector?

9.4.3. What is the dead time ratio of the radiation counter in step 3? What is it for the design in step 4 after decreasing the reset HI output to one 25 MHz clock cycle?

9.4.4. What is the cause of the dead time for both step 3 and 4. Explain specifically what (signal) causes it and what its effect is on the counter.

9.4.5. Dead time can also affect the results from your data analysis and, therefore, needs to be accounted for. For example, if you summed the displayed counter values over a fixed time period for the counters from step 3 and 4 you would observe a significant difference in the final sum. How does dead time affect it and how would you account for it to obtain the actual values of ionizing events?

9.4.6. What are the similarities / differences of this radiation monitor (use the design of step 4) to the previously build frequency counter in section 8.3.? Which major components are similar, which are different in their functionality?
This section contains no exercises (or write-ups) but tries to sum up some topics briefly touched on previously.

9.5.1. Verilog Summary

Here is a very brief summary on what has been covered so far:

To assign a (single) input (or output) to a specific pin use:
(* LOC = "P54" *) input sys_clk_in;

To override the compiler's refusal to use any input signal for sequential logic clk signals use:
(* CLOCK_DEDICATED_ROUTE = "FALSE", LOC = "P69" *) input clk;

To assign an entire bus of inputs and outputs to specific pins (on the BAYS board) use:
(* LOC = "P6 P10 P12 P18 P24 P29 P36 P38" *) input [7:0] Switches;
(* LOC = "P41 P47 P48 P69" *) input [3:0] Buttons;
(* LOC = "P83 P17 P20 P21 P23 P16 P25" *) output [6:0] sevenSegLED_out;
(* LOC = "P2 P32 P33 P34" *) output [3:0] sevenSegPos_out;
(* LOC = "P2 P3 P4 P5 P7 P8 P14 P15" *) output [7:0] LEDs;

Note that the rightmost pin corresponds to the LSB of the bus and the leftmost to the MSB.

<table>
<thead>
<tr>
<th>Combinational Logic</th>
<th>Sequential Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition:</strong></td>
<td></td>
</tr>
<tr>
<td>Output depends only on the present inputs. Has no memory.</td>
<td>Output depends not only on the present input but also on the history of the input. Has Memory</td>
</tr>
<tr>
<td><strong>Main Elements:</strong></td>
<td></td>
</tr>
<tr>
<td>Uses Wires</td>
<td>Uses Registers</td>
</tr>
<tr>
<td>Examples:</td>
<td>Examples:</td>
</tr>
<tr>
<td>wire one_wire; wire [15:0] many_wires;</td>
<td>reg one_register; reg [15:0] many_registers;</td>
</tr>
<tr>
<td><strong>Assignment:</strong></td>
<td></td>
</tr>
<tr>
<td>Blocking Assignment Operator with assign keyword:</td>
<td>Non-Blocking Assignment Operator without assign keyword:</td>
</tr>
<tr>
<td>Example: assign one_wire = 1;</td>
<td>Example: one_register &lt;= 1;</td>
</tr>
<tr>
<td><strong>Branching Statements:</strong></td>
<td></td>
</tr>
<tr>
<td>assign one_wire = cond ? a : b;</td>
<td>if (cond == 1)</td>
</tr>
<tr>
<td></td>
<td>one_register &lt;= a;</td>
</tr>
<tr>
<td></td>
<td>else</td>
</tr>
<tr>
<td></td>
<td>one_register &lt;= b;</td>
</tr>
</tbody>
</table>

9.5.2. Blocking vs. Non-Blocking Assignment Operators

The topic of blocking vs. non-blocking assignment operators has been named as one of the most misunderstood and confusing of Verilog. There is probably some truth to that and so, if you already feel overloaded with all the Verilog information presented so far, you may skip this section and return to it later when you feel more sure about Verilog's foundations.

Previously, it was stated that register assignments should always be done using the non-blocking assignment operator, the <=. (Note the keyword 'should.') The statement is correct and you may want to use the non-blocking assignment operator most of the time; nevertheless, you are also
allowed to use the blocking assignment operator, the simple equal sign. If you prefer to use the blocking assignment with a register, you should not use the `assign` keyword. The only limitation on using the blocking and non-blocking operator is that you can never mix the two when assigning values to the same register, for example, in a branching instruction.

(Just to clarify things: you can never use the non-blocking operator with wires! For wires, you always have to use the blocking operator with the `assign` keyword.)

So what’s the difference between the blocking and the non-blocking operators? The difference between these two operators is how the timing of the actual assignment operation is carried out.

Since this discussion is only about registers, it follows that these assignment operations will at all times be part of an always block operation!

Let’s look at an example using the blocking assignment operators shown below. As its name implies, a blocking assignment operator blocks the execution of the statement(s) following it until it has completed its assignment. In that sense, its behavior is identical to what we are used to in standard computer languages such as, for example C or BASIC.

```verilog
always @ (posedge clock) begin
  b = a;
  c = b;
  d = c;
end
```

In the example shown here, the assignments will be executed line by line. First, `a` will be assigned to `b`; next `b` (which now contains the value of `a`) to `c` and finally `c` (which also contains the value of `a`) to `d`. The final result, after the always event had been triggered, is that the value of `a` has been assigned to all the other variables in the always block. Also note that if we were to rearrange the order of the three assignment statements, the outcome would be different!

Now let’s see what happens if we use the non-blocking assignment operator as shown in the example below. As the name implies, an assignment is no longer blocked until its previously listed assignment has been completed. Instead, all assignment are carried out concurrently and at the very instance the always block is executed.

```verilog
always @ (posedge clock) begin
  b <= a;
  c <= b;
  d <= c;
end
```

How is that accomplished? Try to think of it as a two step process: the first step occurs prior to the always event becoming true. In this step all assignment operations are calculated and stored in a temporary memory location. The next step happens at the very instant the always block is activated, i.e., at edge of the clock signal. At this point all the previously stored values are assigned simultaneously to the registers.

Here is yet another way to think of this two step process: in the first step, only the right hand side of the assignments has been completed. In the second step, the left hand side of the assignment is carried out.

Now let’s look at the example. In the first step, prior to the always block being activated, the values of `a`, `b` and `c` are stored in a temporary location. If we indicate the temporary registers with a prime then:

- `a → b'`
- `b → c'`
- `c → d'`
At the next step, when the always event has been activated, the previously stored values are assigned to the actual registers.

\[
\begin{align*}
  b' & \rightarrow b \\
  c' & \rightarrow c \\
  d' & \rightarrow d
\end{align*}
\]

This ultimately results in:

- \(b\) contains the value of what \(a\) had been previously;
- \(c\) contains the value of what \(b\) had been previously;
- \(d\) contains the value of what \(c\) had been previously.

With every further clock period the values of \(a\), \(b\), \(c\) and \(d\) are shifted “down,” i.e., it acts like a shift register.

It has been stated that non-blocking assignment operations are all carried out at the same instant. The result of this is that the ordering of the non-blocking assignment statements within an always block has no effect on the final result!

A more detailed paper on the assignment operators can also be found at: http://csg.csail.mit.edu/6.375/papers/cummings-nonblocking-snug99.pdf
Write-up Format: Long

Reading: E4E: Section 8.1 - 8.2

Introduction: Pulse Width Modulation

The exercises in this chapter will guide you towards the final project where you will be building a 16 bit audio player. The audio signal itself will be generated using a pulse width modulation (PMW) technique. Therefore, most of this chapter is devoted to this technique which is used in DC to DC (voltage) conversions and digital to analog applications. It is widely used because it is energy efficient and uses few electronic components.

Shown above is a graph of an analog Sine wave and its (Sigma Delta) pulse width modulated equivalent output.

10.1. Simple Pulse-Width Modulation (PMW) Technique

Introduction: Light (LED) Dimmer

In this exercise you will use the BASYS board to build a simple PWM circuit to control the brightness of an LED. Switches SW0 through SW7 will be used to adjust the intensity of the LED on the board.
To illustrate the concept of PWM, consider the following circuit: an LED is connected through a switch to a fixed (DC) supply voltage, $V_{on}$.

![Figure 10.1.1. LED with manually operated switch.](image)

The time it takes for entire switching cycle, $\tau_{swc}$, consists of the time that the LED is on, $\tau_{on}$, and off, $\tau_{off}$:

$$\tau_{swc} = \tau_{on} + \tau_{off} \quad (10.1.1.)$$

The switching (cycle) rate or speed, $f_{swc}$, is defined as:

$$f_{swc} = \frac{1}{\tau_{swc}} \quad (10.1.2.)$$

At first, let us operate the switch by hand. As long as the $\tau_{on}$ and $\tau_{off}$ are on the order of seconds our eyes will observe the LED either at its full brightness or off.

Next consider what happens when we connect the switch control to a function generator. Assume for now that the time interval that the light remains on, $\tau_{on}$, is equivalent to the time it is off, $\tau_{off}$.

At a low switching rate, at a few Hertz, our eyes observe the familiar full brightness / off behavior. As the switching speed is increased we begin to notice the blinking LED. As the switching is speed is further increased, we reach a frequency where the response of our eyes, specifically the retina retention, is too slow to register the LED’s distinct on and off states. Instead, at this frequency and beyond it, our eyes will act like an “averager” or low pass filter. We perceive an average intensity that is somewhere between the LED’s full brightness and it being completely turned off. In the current case, with $\tau_{on} = \tau_{off}$, we probably perceive the LED at half its maximum brightness.

How does the ratio of $\tau_{on}$ and $\tau_{off}$ affect the “average” brightness perceived? Assume that the switching rate, $f_{swc}$, is kept constant and remains at a frequency exceeding our eyes’ response time. As mentioned previously, at this switching rate, the perceived intensity will be an average based on how long the LED is on - or off - during each switching cycle. For example, if $\tau_{on} \gg \tau_{off}$, the LED is on most of the time and a bright LED will be seen; for $\tau_{on} \ll \tau_{off}$, the LED is mostly off and a dimly lit LED will be observed. It follows that by controlling the ratio of $\tau_{on}$ and $\tau_{off}$ we can obtain any desired level of intensity.

Let us determine mathematical relationship between $\tau_{on}$ and $\tau_{off}$ and the “perceived” average intensity. An analogous model of the eye and its response can be represented by the circuit shown in Figure 10.1.2. In this model, $V_{on}$ would be analogous to the maximum light intensity; the RC low pass filter acts like an averager, representing our eyes; finally, $<V_{out}>$ is the average output voltage and it corresponds to the perceived intensity. Therefore, we would like to calculate $<V_{out}> (RC, \tau_{on}, \tau_{off})$. 

FPGA and VERILOG Part II: / 10.1. Simple Pulse-Width Modulation (PWM) Technique 117
Simple Pulse-Width Modulation (PMW) Technique

As will be calculated in write-up 10.1.1, as long as RC is much larger than the time for one switching cycle, i.e., RC ≫ τ_{swc}, \( <V_{out}> \) is:

\[
<V_{out}> = \frac{\tau_{on} V_{on}}{\tau_{on} + \tau_{off}} \tag{10.1.3.a}
\]

\[
= \frac{\tau_{on} V_{on}}{\tau_{swc}} \tag{10.1.3.b}
\]

Before we proceed to implementing the PWM circuit in Verilog let’s consider the implications of the results above.

First, these results show that with the aid of a timing signal controlling a switch, a fixed (DC) voltage can be converted into a voltage that is less than its maximum input voltage, \( V_{on} \). In other words, for \( RC ≫ \tau_{swc} \), the circuit acts very much like a voltage divider! Unlike the “normal” 2 resistor voltage divider, this circuit has the advantage that (ideally) no power is wasted in the regulation.

Second, if the system to which the PWM is applied responds much slower than the switching speed, then the low pass filter that provides the averaging is no longer required. For example, when using the LED and our eyes, the eyes will provide the low pass filtering. This is true for many other cases. For example, in “power” applications where the PWM technique is used to control speakers, motors, solenoids or other actuators, as long as the response time of the device being powered is much slower than \( \tau_{swc} \), the low pass filter is omitted because the slow response of the system that is being controlled provides it indirectly.

Combining these two observation leads to a very important implication: since the PWM circuit acts like a voltage divider and if the low pass filter is part of the “load” then all the power will be absorbed by the load and none will be wasted by the regulating circuitry! In other words, a PWM circuit can be far more efficient in regulating power than a regular two resistor voltage divider where a lot of power is being wasted in the regulating resistors. (See Exercise 10.1.3.) This is one of the reasons that PWM are so popular in many low power applications such as portable electronics devices.

Verilog Implementation of a Simple PWM Technique

A critical element in the PWM technique is the switch control algorithm or module; it controls the time interval that the switch stays on, \( \tau_{on} \), or off, \( \tau_{off} \). (See the diagram in Figure 10.1.3.) This module has two inputs. An external clock input, \( f_{clock} \), provides a trigger signal for the algorithm and also acts a reference clock signal. The \( x_{in} \) input specifies the number of reference clock cycles.
that the switch stays on, or off. In other words, $x_{in}$ will be directly proportional to $\tau_{on}$ and, hence, $<V_{out}>$. Its output, $PWM_{out}$, is the pulse width modulated signal. It can be used directly to control low power applications, such as dimming an LED but generally this signal is fed into some sort of physical switch, for example a transistor or relay, to provide a power amplification, as shown below.

![Simple Pulse-Width Modulation (PMW) Technique](image)

Figure 10.1.3. The complete PWM circuit. The actual switching element can be a physical switch, such as an electrical relay. Typically though transistors are used for switching, most commonly in the configuration of silicon controlled rectifier, i.e., an SCR. The low pass filter is not required if the system to which the PWM signal is applied to responds much slower than one complete switching cycle.

Now let's look at the actual implementation of the switch control algorithm. After this lengthy discussion it may come as a pleasant surprise to discover just how straightforward its implementation is in Verilog. The complete module is shown below.

```
module SimplePWM(clk_in, x_in, PWM_out);
    parameter MAXBITS = 8;  //maximum number of bits for input value and counter
    input clk_in;         //clock for counter
    input [MAXBITS-1:0] x_in;   //control value that defines pulse width
    output reg PWM_out = 1;  //PWM signal out
    reg [MAXBITS-1:0] counter = 0;

always@ (posedge clk_in)begin
    if (counter < x_in)
        PWM_out <= 1;
    else
        PWM_out <= 0;
    counter <= counter+1;
end
endmodule
```

Module 10.1.1. Simple PWM Technique. (You do not have to type this code in; instead it can be copied from the file PWM_Modules.v from the folder U:\pub_MXP\Verilog\PWM to your current project directory.)
Except for the “parameter” statement, all other Verilog commands should look familiar to you. The parameter statement is similar to a preprocessor or constant in C. It allows presetting values such as register size so that, at a later time, they can easily be adjusted throughout the code.

The main components of the PWM technique consist of a counter and a digital comparator. Whenever the continuous count value exceeds the input, \( x \), the module outputs a LO, else it is HI. In other words, its output, \( PWM_{out} \), acts like the switch output from the previous discussion.

The input, \( x \), is a user selectable value. It controls how long the output remains HI or LO, i.e., how long the “switch” stays “on” and “off.” Thereby, it determines the length of \( \tau_{on} \) and \( \tau_{off} \). The range of \( x \) must never exceed the largest count value of the counter. Therefore, its range depends on the size of the counter specified in the module listed by the parameter \( MAXBITS \). For an \( n \)-bit counter, its range is: \( 0 \leq x < 2^n \).

Note that the counter runs continuously. Once it reaches its maximum value, it starts over again at 0. Since the counter is incremented at a frequency, \( f_{clock} \), the duration of an entire switching cycle for an \( n \)-bit counter is:

\[
\tau_{swc} = 2^n / f_{clock} \quad (10.1.4)
\]

From this it follows that the switch remains “on” for the following time interval:

\[
\tau_{on} = x / f_{clock} \quad (10.1.5)
\]

If the output from the module were connected to a low pass filter with a large RC time constant, (see figure 10.1.2.) then we would find that \( <V_{out}> \) is:

\[
<V_{out}> = x V_{on} / 2^n \quad (10.1.6)
\]

In other words, the averaged output voltage is directly proportional to \( x \). Since \( x \) is a digital value and \( <V_{out}> \) an analog value, we see that the PWM circuit acts like a digital to analog converter, i.e., a D2A!

**Exercise**

In this exercise you will measure the response time of your eyes by connecting an LED to the output of a PWM circuit. You will adjust the PWM module’s clock frequency, \( f_{clock} \), until you are no longer able to observe the discrete blinking of the LED.

We do not want to use external components. Instead, we will implement a circuit (in Verilog) that generates an adjustable clock frequency, \( f_{clock} \) from signals available on the BASYS board. The method we will use is to drive an \( n \)-bit counter with the BASYS board’s 25 MHz system clock. The LSB of the counter will then correspond to a frequency of 25 MHz / 2; the next higher bit has a frequency of 25 MHz / 4 and correspondingly, the \( n \)th bit of the counter will have a frequency of 25 MHz/2\(^{(n-1)}\). (Note the LSB corresponds to \( n = 0 \).) Therefore, by selecting the \( n \)th counter bit and assigning it to the output \( clk_{out} \) will produce a discretely adjustable clock frequency, \( f_{clock} \) to clock the PWM module.

“Build” the entire PMW circuit using the switches, clock and an LED on the BASYS board using Verilog. (Except for the BNC connectors to observe the output signal, you will not need any external components.) See the diagram below and read its caption for the additional hint.
Use a 20-bit counter as a frequency divider to control the frequency of \( f_{\text{clock}} \). To start with, assign the 12th bit to \( \text{clk}_\text{out} \).

Instantiate the module called \textit{SimplePWM} shown in module 10.1.1. Use Switches SW7 (MSB) through SW0 (LSB) to create the 8 bit control value, \( x \), for the PWM module. Select any one of the LEDs LD0 to LD7 as the output. Additionally, also connect the \( \text{PWM}_\text{out} \) to pin 1 and \( \text{clk}_\text{out} \) to pin 2 of connector JA to observe the output signal and \( \text{clk}_\text{out} \) on the scope.

Upload the compiled code into the BASYS board. When you operate switches SW0 through SW7 can you control the blinking LED's brightness? While observing the LED, also study the PWM output on the scope.

Next assign \( \text{clk}_\text{out} \) to the counter’s next lower bit, i.e., \( n = 11 \). Decrease the value of \( n \) until you can barely discern the blinking of the LED when the switches are set to provide the LED with half its intensity, i.e., \( x\_in = 128 \) or the switches set to 1000 0000. The rate at which the LED blinks is the response time of your eye, or its \( f_{3dB} \). Calculate it from the measured \( \text{clk}_\text{out} \) signal.

\section*{Write-Up}

10.1.1 For \( RC = \tau_{\text{swc}} \), calculate \( <V_{\text{out}}(RC, t_{\text{on}}, t_{\text{off}})> \) using the circuit shown in Figure 10.1.2. You may use either of the two methods outlined below:

\begin{itemize}
  \item \textit{Hint 1:} find a (differential) equation for the change in voltage, \( dV \), across the capacitor during \( dt = \tau_{\text{on}} \) and \( dt = \tau_{\text{off}} \). Note that \( V_{\text{in}} \) during \( \tau_{\text{on}} \) is \( V_{\text{on}} \) and it is 0 during \( \tau_{\text{off}} \). Finally, for the output to remain around a fixed (steady state) value, \( <V_{\text{out}}>, dV \) during the on time must be equal to \(-dV\) during the off time.
  \item \textit{Hint 2:} calculate the average by using the general form: \( f(t) = \int_{t_0}^{t_0 + \tau_{\text{on}} + \tau_{\text{off}}} \frac{1}{\tau_{\text{on}} + \tau_{\text{off}}} f(t) \ dt \) where \( f(t) \) corresponds to the voltage fed into the low pass filter during one entire cycle, i.e., \( t_0 = \tau_{\text{on}} \) and \( \tau_{\text{on}} + \tau_{\text{off}} \). Since the low pass filter is an averaging circuit, the result obtained by this method must be identical to the one obtained using an ideal low pass filter.
\end{itemize}
10.1.2 The graph above shows $V_{out}/V_{on}$ as a function of time for two different values of RC for the circuit in 10.4.3. As you can see, adjusting the value of RC has two major effects on the output: how quickly the output responds and the size of the ripple. Describe how increasing (or decreasing) the value of RC affects these two properties.

10.1.3 To conserve energy, a person decides to dim the light in the house by adding a regulating resistor (in series) to a lamp. For now assume that the resistance of the regulating resistor is 5 times the resistance of the lamp. How much power overall was saved by the entire circuit? How much (more) power is absorbed by the regulating resistor then by the lamp? Why would a PWM circuit be a better choice?

10.1.4 What was the measured response time of your eye, or its $f_{3\text{dB}}$?

10.1.5 What's the resolution of the simple PWM digital to analog converter?

10.2. Sigma Delta Algorithm to Produce Audio Clock Signal

### Introduction

In this section, we study the Sigma Delta algorithm and then use it to generate the 44.1 kHz clock signal for our audio player. In the next section, the Sigma-Delta algorithm will be used to improve the simple PWM technique from the previous section.

### Clock Divider Algorithms

Most digital circuits rely on multiple clock signals. Generally, a master clock signal, like the BASYS board’s 25 MHz system clock signal, is selected as a reference signal. All other clock signals are then generated by “slowing” down the reference clock signal to the particular frequency of interest by using some sort of counter circuit.

You have already used two different methods for generating clock signals from a (faster) reference signal:

- In the previous section, the MSB of a continuous n-bit counter was used to divide the reference frequency by $2^n$; this method is extremely straightforward but limits the range of new frequencies to: $f_{\text{reference}} / 2^n$.
- In section 9.3.3, a counter was reset each time a particular time interval (a second) was reached; this method works fine as long as the reference frequency is an integer multiple of the desired clock frequency.

When the reference frequency is not an integer multiple of the desired clock frequency then these methods cannot be used generate an accurate clock signal. Here is an example: assume that you have a reference clock signal of 256 Hz and that you need to generate a new clock signal, arbitrarily chosen, at 6 Hz. The ratio of these frequencies is 42.667. Since the flip-flops can only be triggered an integer number of clock cycles, the new output signal can either have 42 or 43 reference clock cycles. What is needed is some algorithm that corrects itself. For example, two cycles with 43 reference cycles will always be followed by a cycle with 42 clock cycles. Over time, the cycles will average out to 42.667 cycles, i.e., a frequency of 6 Hz. The Sigma Delta algorithm, shown below, achieves exactly that. (Its name originates from the fact that it continuously adds (Sigma) and subtracts (Delta).)

Here is a (C-style) version of the Sigma Delta algorithm with line numbers. It will be executed at each clock cycle at a frequency $f_0$. Its input values are $y$ and $z$. The range of $z$ and $y$ is: $0 \leq y \leq z$.

```c
1. if ( Sigma >= z )
2. {
```
3.  \( \text{Sigma} \leq \text{Sigma} + y - z; \)
4.  \( \text{Out} \leq 1; \)
5.  }
6.  \text{else}
7.  \{
8.  \( \text{Sigma} \leq \text{Sigma} + y; \)
9.  \( \text{Out} \leq 0; \)
10.  }

Figure 10.2.1. Sigma Delta algorithm: Note ‘z’ acts as “Delta,” i.e., the value that is subtracted if the sum (Sigma) exceeds ‘z.’ NOTE: Due to a bug in Verilog, make sure to keep the order of the terms in line 3 as shown above!

A graph of \( \text{Sigma} \) (at line 12) and \( \text{Out} \) as function of clock cycles is shown below.

![Graph of Sigma Delta Algorithm](image)

Figure 10.2.2. Graph of the variables “Out” and” Sigma” during 256 consecutive clock cycles.

The graph shows that the 6 \( \text{Out} \) signal pulses were (more or less) evenly distributed within the 256 clock cycles.

As you will show in write-up 10.2.1., if the algorithm is executed at a frequency \( f_o \), then the average frequency for the \( \text{Out} \) pulses, \(<f_{out}(f_o, y, z)>, \) with \( 0 < y < z \), is:

\[
<f_{out}> = y f_o / z
\]  

(10.2.1.)

If the value for \( z \) is substituted by the actual clock frequency, \( f_o \), then this becomes

\[
<f_{out}> = y
\]

(10.2.2.)

This trick comes in very handy if we want to generate some arbitrary clock frequency and, therefore, this property is often applied in voltage-to-frequency conversion schemes.

To sum up, given a reference clock signal, \( f_o \), the Sigma Delta algorithm allows us:

a)  to generate a number of pulses that are evenly distributed in time;
b) to generate a new clock signal with an average frequency \( y \) when \( z = f_o \).

Property a) will be very useful in the next section where it will be applied towards a much improved PWM technique while property b) will be used in the subsequent exercise.

**Exercise**

Use the Sigma Delta algorithm and write a Verilog module to generate the 44.1 kHz audio clock signal using the BASYS board’s 25.0 MHz system clock as its input. The module will output a positive going pulse for one system clock cycle (on the average) every 44.1 kHz.

![AudioClock](image)

*Figure 10.2.3. Schematic of AudioClock module.*

When you are implementing the code, check that the register size for Sigma and Delta is adequate to hold their maximum values. (Recall that \( z = f_o = 25'000'000 \) and \( y = 41000 \).) In other words, calculate the bit size required for these registers! (If you want to implement a more efficient version of your code, note that as long as the ratio of \( z \) and \( y \) remains constant the code works the same; therefore, dividing \( z \) and \( y \) by 1000 will not change its behavior but will substantially lower the number of bits required!)

Test your module with the scope. Check that your module produces a positive going pulse at a repetition rate of 44.1 kHz. (Note: The 25 MHz system clock is only accurate to one or two percent of its nominal value. So don’t be alarmed if you are off by one or two percent from the expected frequency.)

**Write-Up**

10.2.1. Derive equation 10.2.1. (Hint: one approach would be to calculate the average number of \( \text{Out} \) signals, \( n_{\text{out}} \), generated during some arbitrary time interval, \( \tau \). The ratio of \( n_{\text{out}} / \tau \) then corresponds to the average frequency of the \( \text{Out} \) pulses, \( \langle f_{\text{out}} \rangle \). Assume that the time interval, \( \tau \), lasts \( n \) clock cycles, \( \tau = n / f_{\text{clock}} \). At each clock cycle \( \text{Sigma} \) is incremented by \( y \). From this you can calculate by how much \( \text{Sigma} \) was incremented over the entire time interval, \( \Sigma_{\text{Tck}} \). Each time \( \text{Sigma} \) reaches or exceeds \( z \), \( z \) is subtracted from it and an \( \text{Out} \) pulse is generated. Since over the entire time interval, \( \tau \), \( \text{Sigma} \) was incremented by \( \Sigma_{\text{Tck}} \), you should be able from this to calculate the number of \( \text{Out} \) pulses, \( n_{\text{out}} \), generated during the time interval, \( \tau \).

10.2.2. The 44.1 kHz rate for sampling and playing audio signal is an industry standard. How does it relate to the range of human hearing?

**10.3. Sigma Delta PWM**

**Introduction**

In this section we will use the Sigma Delta algorithm to improve the simple PWM technique used in the first section. The resulting circuit will then be used for the audio player’s A2D converter.
Audio Player Specifications

Before we continue building our audio player, let's look at the specifications that a “decent” audio player must meet. First, it must be able to cover the audible frequency range which is: \(20 \, \text{Hz} < f_{\text{audible}} < 20 \, \text{kHz}\). Second, the human hearing is able to distinguish changes in the audible intensity on the order of 1:10000. In other words, its resolution is \(1.2^{14}\) or better; we will aim for a \(1.2^{16}\) resolution. We now want to apply these specifications our simple PWM circuit from section 10.1. and see if they can be met using the BASYS board and its 25 MHz system clock.

Resolution and Frequency Response of the Simple PWM Circuit

The resolution of a D2A converter is defined as the smallest voltage change that the converter is able to output. In other words, it represents the “step size” in its output voltage when its input changes by the least significant bit, i.e., \(\Delta x_n = \pm 1\). We prefer these steps to be small. This provides for “smoother” transitions between values and thereby reduces the noise associated with large steps. See the graph below for a Sine wave digitized at different resolutions.

![Vout vs. Time for Different D2A Resolution Levels](image)

Figure 10.3.1. \(V_{\text{out}}\) for different resolution levels.

According to equation 10.1.6., the simple PWM D2A converter from the previous section is capable of producing \(2^n\) distinct (averaged) output voltage levels ranging from 0 to \(V_{\text{on}}\). Therefore, its resolution is \(V_{\text{on}} / 2^n\). For a specific output voltage range, its resolution is entirely defined by \(n\), i.e., the (bit) size of its internal counter. By adjusting the size of the counter register into a 16, 32 or even 64 bit counter we can easily adjust its resolution to obtain a more precise and smoother output. Achieving the specified 16 bit resolution certainly presents no problem for this design.

Next we need to determine the minimum time during which the input, \(x_n\), to the PWM circuit must be maintained to obtain \(<V_{\text{out}}>\) as specified by equation 10.1.3. Recall that \(<V_{\text{out}}>\) must be averaged over at least one full switching cycle, \(T_{\text{swc}}\); changing \(x_n\) before at least one switching cycle has elapsed could lead to unpredictable results because we are no longer able to average over an entire cycle. From this it follows that the shortest time interval over which \(x_n\) is allowed to change is, \(T_{\text{swc}}\); applying equation 10.1.4. it follows that the maximum input frequency for simple PWM circuit is:

\[
 f_{\text{MAX Simple PWM}} = 1 / T_{\text{swc}} = f_{\text{clock}} / 2^n \quad (10.3.1.)
\]

For a 16 bit counter with a 25 MHz clock, \(f_{\text{MAX}}\) is a paltry 400 Hz. Clearly, it does not meet the audio frequency range specified previously.

Why is \(f_{\text{MAX}}\) so slow? Because it takes \(2^{16}\), i.e. approximately 65000, clock cycles to complete just one switching cycle. Even at 25 MHz, this takes a long time. It is tempting to reduce the
number of clock cycles for a given switching cycle but this can only be accomplished at the cost of reducing the A2D’s resolution. What is needed is some method to reduce the cycle time without affecting the resolution!

**Resolution and Frequency Response of the Sigma Delta PWM Circuit**

The Sigma Delta algorithm allows us to spread a signal over some arbitrary time interval by breaking it up into evenly spaced pulses. We will use this feature to reduce the time it takes for one switching cycle to repeat itself.

The key to understanding the Sigma Delta PWM technique lies in the following observation: the averaged voltage of a PWM system, \(<V_{\text{out}}\rangle\) only depends on the actual number of clock cycles that the switch remains on, or off, during one switching cycle \(\tau_{\text{swc}}\). (See equation 10.1.3b.) Since we are observing an averaged value, and averages are computed by addition which is commutative, it does not matter in what order the switch was turned on or off as long as the number of on and off cycles is preserved. To illustrate this idea, consider the examples shown below.

First, let us revisit the simple PWM technique from the previous section, see Figure 10.1.3. A reference clock with frequency \(f_{\text{clock}}\) triggers the switching algorithm, shown in module 10.1.1. The algorithm, based on its input value \(x_{\text{in}}\), then determines at each clock cycle whether the switch remains in the open or closed position, resulting in \(V_{\text{out}}\) (before the low pass filter) either being \(V_{\text{on}}\) or 0 V.

For example, let us assume (arbitrarily) that the algorithm is fed an input value of \(x_{\text{in}} = 4\) and that it uses a 4 bit counter. (The module in 10.1.1. uses an 8 bit counter but such a timing diagram would be too lengthy to draw.) The timing diagram of \(V_{\text{out}}\) (before the low pass filter) is shown below. Specifically, for the conditions listed, the switch output would remain HI (or on) for 4 reference clock cycles and then LO (or off) for 12 cycles. As long as the input value \(x_{\text{in}}\) does not change, the process repeats itself after these 16 reference clock cycles. In other words, the time it takes for the switching cycle to repeat itself, \(\tau_{\text{swc}}\), would be 16 (reference) clock cycles. To obtain a meaningful \(<V_{\text{out}}\rangle\), we would have to average \(V_{\text{out}}\) for over at least one (or more) complete switching cycles. Since on the average, \(V_{\text{out}}\) is HI (or on) for 4 cycles and then off for 12, \(<V_{\text{out}}\rangle = 4 \frac{V_{\text{on}}}{16} = \frac{V_{\text{on}}}{4}\). (See equation 10.1.3b.)

![Figure 10.3.2a. V_out (before the low pass filter) vs. time for the circuit in Figure 10.1.2.](image)

Let’s consider the timing diagram in Figure 10.3.2b. The timing intervals were randomly rearranged within the switching cycle. Nevertheless, the number of (reference) clock cycles during which the switch remained closed, or open, is identical to the previous case: during a complete switching cycle, the switch still remains on for 4 cycles and off for 12 and, therefore, \(<V_{\text{out}}\rangle\) remains \(\frac{V_{\text{on}}}{4}\), just as in the previous case.

![Figure 10.3.2b.](image)
Since the arrangement of the on and off cycles does not matter as long as their ratio is conserved, is there any arrangement that is superior over another one? Consider the time diagram below. While the number of on and off cycles is identical to the previous cases, note that the switching cycle repeats itself twice within the original switching cycle, $\tau_{swc}$, resulting in an effective switching cycle, $\tau_{swc\_effective}$, of only 8 reference clock cycles.

$$\text{Figure 10.3.2c.}$$

Can we do better than that? Specifically, can we find some algorithm that redistributes the on and off cycles in such a manner that we always obtain the smallest effective switching time while maintaining the ratio of on and off cycles? What method will give us a timing diagram as the one shown below?

$$\text{Figure 10.3.2d.}$$

As you probably guessed, applying the Sigma Delta algorithm to the switching circuit of Figure 10.1.3. will accomplish that. The Sigma Delta algorithm will provide us with an effective switching cycle time, $\tau_{swc\_effective}$, that is "usually" far shorter than $\tau_{swc}$. Reducing the switching cycle time allows us to shorten the time interval over which we need to average and, hence, allows us to update $x\_in$ at a much faster rate.

How much shorter is $\tau_{swc\_effective}$ than $\tau_{swc}$? In the example above, we only saved a factor of 4. That does not seem very impressive but keep in mind that we only had a maximum of 16 clock cycles to begin with because of the 4 bit counter used in the example. Being able to reduce the repetition rate down to just a few clock cycles when we start out with $2^8$ or $2^{16}$ clock cycles for the 8 and 16 bit counters will have a profound effect of reducing $\tau_{swc}$. It is beyond the scope of this discussion but you may assume that for an 8 bit counter 95% of all input values have a $\tau_{swc\_effective}$ of fewer than 10 clock cycles. In other words, the switching cycle was reduced by a factor of 25! To put this in perspective, when we use the system clock of the BASYS board, $f_{MAX}$ can now easily exceed the 20 kHz audio specifications most of the time even in the 16 bit case.

The Sigma Delta technique performs as poorly as the simple PWM for the extreme low or high input values, $x\_in$. At these values, the $\tau_{swc\_effective}$ from Sigma Delta algorithm will approach $\tau_{swc}$ of the simple PWM switching technique. (For example, for an input value of 1 the switch stays on for only one reference clock cycle; you don’t have much freedom in arranging the pulses anymore.) This does not have to concern us as we will place the bulk of the power of our audio signal into the center of the input range. For most of the input range, $f_{MAX}$ far exceeds the audio frequency range specifications listed earlier.
Summary

Compared to the simple PWM from the previous section, the Sigma Delta algorithm preserves the ratio of the time that the switch stays on vs. remains off and, therefore, it will not affect the value of $<V_{out}>$ or its resolution. Using the Sigma Delta algorithm from section 10.2 and $z = f_{\text{clock}}$, if follows that:

$$<V_{out}> = \frac{y}{V_{on}} f_{\text{clock}}$$  \hspace{1cm} (10.3.2)

On the other hand, by spreading out the on and off cycles over one entire switching cycle we have decreased the time over which a switching cycles repeats itself.

Exercises

10.3.1. Use the code from exercise 10.1 to dim the LED, including the n-bit counter to divide the 25 MHz system clock frequency by $2^n$. See below.

![Diagram of Counter and SigmaDeltaPWM modules](image)

Figure 10.3.4. Dimming the LED using the Sigma Delta PWM technique. Again, assign an output port (not shown above) for clk_out.

Apply the following modification to your original code: replace the instantiation of the *SimplePWM* module with the *SigmaDeltaPWM* module shown below or your own version based on the pseudo-code of Figure 10.2.1. (You do not have to type this code in; instead copy the file PWM_Modules.v from the folder U:\pub_MXP\Verilog\PWM to your current project directory.)

```verilog
//Sigma Delta PWM unsigned version KW 12/4/12
module SigmaDeltaPWM(clk_in, x_in, PWM_out);

parameter MAXBITS = 8;       //maximum number of bits for input value
input clk_in;                //clock for counter
input [MAXBITS-1:0] x_in;   //control value that defines pulse width
output PWM_out;             //PWM signal out

reg [MAXBITS:0] Sigma = 0;  //summer (NOTE: needs one extra bit)

always@ (posedge clk_in) begin
    Sigma <= Sigma[MAXBITS-1:0] + x_in;  //Adds all bits except MSB of Sigma
end

assign PWM_out = Sigma[MAXBITS];  //MSB of Sigma is PWM_out (only works in $2^\text{MAXBIT}$ counter cases.)
endmodule
```

Code 10.3.1. Optimized Verilog implementation of the Sigma Delta PWM algorithm.

The code in 10.3.1 may look unfamiliar to you but it is just a different implementation of the algorithm shown in Figure 10.2.1. It has been optimized to work for cases where $z$ is a power of 2 and, therefore, will work fine with our 8 bit counter. Again use the switches to control the control variable $x_{in}$. Use the Counter module to clock the SigmaDeltaPWM module again with the frequency for which you previously observed blinking for the SimplePWM module. Do you still observe blinking when you set your switches so their input value is near the center of their 8 bit range, i.e., at 128? How about if you change the control value to a very small or large value? Now
see to what frequency you can reduce $\text{clk}_{\text{in}}$ by increasing the value of $n$ in the Counter module until you do not observe blinking for an input value of 128. What happens when you use different values for the $x_{\text{in}}$? Change your switch settings and try $x_{\text{in}} = 1, 2, 3$ and 4.

**Write-up**

10.3.1 What is the slowest $f_{\text{clk}}$, i.e., $\text{clk}_{\text{out}}$, at which you were still able to observe the blinking LED when $x_{\text{in}} = 128$? How does it compare to the result obtained in the previous write-up 10.1.4?

10.3.2 Assign the 8th counter bit to $\text{clk}_{\text{out}}$ and observe $\text{PWM}_{\text{out}}$ and $\text{clk}_{\text{out}}$ on the scope. Describe what happens when your input control value $y$ is: 128, 64, 32 and 127.

10.3.3 Draw the graphs of $\text{PWM}_{\text{out}}$ vs. $t$ for $y = 128, 64, 32$ and 127. For each of these measure or calculate the “effective switching cycle time,” $t_{\text{swc effective}}$. Measure the frequency of $\text{clk}_{\text{out}}$. In each of these cases, how does $\text{clk}_{\text{out}}$ relate to the effective switching time, i.e., for the different cases, how many cycles of $\text{clk}_{\text{out}}$ are required for effective switching time?

### 10.4. Audio Player Using Sigma Delta PWM Technique

We are now ready to apply the Sigma Delta PWM technique to build a basic audio player. The audio information has been stored in both 8 and 16 bit words on 2 Megabyte flash memory cards that plug directly into the BASYS board. Since it is easier to work with 8 bit data, we will first implement an 8 bit word audio player. Once it works, it will be straightforward to convert the code to accommodate the flash memory cards with the 16 bit audio data. Using both the 8 and 16 bits cards will allow you to compare the sound quality between the two.

The hardware consists of:
- 2 MB flash memory cards that contains the audio data;
- the BASYS board which will provide PWM circuit;
- an audio amplifier to boost the power of the audio signal;
- a speaker.

These components are already assembled and can be plugged into the BASYS board’s 6 pin edge connectors; see below.

10.4.1. Picture on the left: Digilent PModSF 2 Megabyte flash memory module; Picture on the right: Digilent PmodAMP1 Speaker/Headphone amplifier module to boost the audio signal strength.

The Verilog code can be broken down into the following components:
- The 44.1 kHz audio clock module, named AudioClock in Figure 10.4.2., provides the trigger signal for reading the audio data from the flash memory modules; the trigger signal also increments the address counter.
- The module named AddressCounter specifies the current flash memory address to be read.
- The module ReadOneByte reads one (buffered) byte of digital audio data from the PModSF flash memory module shown in Figure 10.4.1. The reading is triggered by the enable input and the byte is read from the memory location specified by the address input.
- The module SigmaDeltaPWM uses the Sigma Delta PWM technique to convert the digital audio data into an analog output signal that is then be sent to the PModAMP1 audio amplifier module shown in Figure 10.4.1.
The modules for the audio clock and the Sigma Delta PWM technique have already been written or supplied to you in sections 10.2. and 10.3.

The module for reading the audio data from the flash memory, ReadOneByte, has already been written for you. It is found in the file FlashMemModulesV6.v which must be copied from the folder U:\pub_MXP\Verilog\FlashMemory to your current project directory. Add this file to your project (after copying it to your project folder) by using the menu bar and selecting Project \ Add Source and then choosing the FlashMemModulesV6Basic.v file.

The only new module that you will supply is the AddressCounter module. It consists of a simple 24 bit (register) that is incremented at each positive edge of its input signal. Additionally, the counter is reset to zero each time it exceeds the address range of the flash memory, i.e., at Hex 1FFFFF or, in Verilog notation, 24'h1f_ffff.

After you get the 8-bit version working, make these changes to your code for the 16 bit version:
- Increment the address counter by 2 (instead of 1) at each positive edge audio clock input to account for the two bytes you are now reading at each audio clock cycle.
- Replace the module ReadOneByte with the module ReadTwoBytes, also located in the FlashMemModulesV6.v file.
- In the SigmaDeltaPWM module, change "parameter MAXBITS = 8;" to "parameter MAXBITS = 16;" to allow for the 16 bit input signals.

Enjoy! (For your own curiosity you may want to see if the previously stated concern about bandwidth was justified and that we indeed needed the Sigma Delta PWM module instead of the Simple PWM. Replace the SigmaDelta PWM module with the Simple PWM from section 10.1 for both the 8 and 16 bit version and see if you can tell the difference.)

What follows is a more detailed description of some of the hardware and software components.

**Details**

**Audio Data**: the audio data was created from 8 and 16 bit (mono) WAV files sampled at 44.1 kHz. WAV files are easy to work with for data manipulation because they have not been compressed; the audio information is simply stored in a sequential binary file which can easily be read by a C-program. Unfortunately, since the information has not been compressed, it uses a lot of memory. This is the reason why we are able to play such short audio pieces. (Of course, it would far more rewarding if we were able to use a MP3 compression / decompression algorithm for our data. Unfortunately, these codes are copyrighted and not easily obtainable.)
**Flash Memory:** after the binary data has been read by the C-program it is offset to the middle of its dynamic range (to compensate for signed values) and then it has been stored (via an RS232 interface and the BASYS board) in the flash memory.

The flash memory can contain up to 2 Megabytes of data and it communicates with the BASYS board through a serial peripheral interface (SPI) using the following 4 pins:

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>Direction</th>
<th>Purpose</th>
<th>6 Pin Edge Connector JX Pin Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q</td>
<td>Input</td>
<td>Read Data From Memory Line</td>
<td>3</td>
</tr>
<tr>
<td>D</td>
<td>Output</td>
<td>Send Data to Memory Line</td>
<td>2</td>
</tr>
<tr>
<td>NCS</td>
<td>Output</td>
<td>Chip Select Line (Negative Logic)</td>
<td>1</td>
</tr>
<tr>
<td>CLK_OUT</td>
<td>Output</td>
<td>Clock Line</td>
<td>4</td>
</tr>
</tbody>
</table>

Though it does not matter which of the four 6-pin edge connectors (JA through JD) you will be using for the flash memory cards, it is crucial, that once you have selected one, you assign the signals above to the appropriated pins. (See also page 169 in lab manual for BASYS board pin numbers.)

**ReadOneByte or ReadTwoBytes Modules:** these modules will read either one or two consecutive bytes of data from the flash memory at the address specified at its input. The modules use the following ports:

<table>
<thead>
<tr>
<th>Direction</th>
<th>Name</th>
<th>Size (Bits)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>clk_in</td>
<td>1</td>
<td>Serial interface clock for transmitting data; use the 25 MHz system clock.</td>
</tr>
<tr>
<td>Input</td>
<td>enable</td>
<td>1</td>
<td>Hardware Trigger: at the positive edge, the new data is retrieved and subsequently stored in registers “data.”</td>
</tr>
<tr>
<td>Input</td>
<td>address</td>
<td>24</td>
<td>Address where data is retrieved from in the flash memory.</td>
</tr>
<tr>
<td>Input</td>
<td>Q</td>
<td>1</td>
<td>Data read from memory is transmitted through this line.</td>
</tr>
<tr>
<td>Output</td>
<td>NCS</td>
<td>1</td>
<td>(Not) Chip-Select line.</td>
</tr>
<tr>
<td>Output</td>
<td>D</td>
<td>1</td>
<td>Data sent to the memory is transmitted through this line.</td>
</tr>
<tr>
<td>Output</td>
<td>clk_out</td>
<td>1</td>
<td>Clock signal from memory; required when receiving data.</td>
</tr>
<tr>
<td>Output</td>
<td>busy</td>
<td>1</td>
<td>While HI, flash memory is busy retrieving data; when it is LO again, data has been stored in “data;” in our application, this output can be ignored.</td>
</tr>
<tr>
<td>Output</td>
<td>data</td>
<td>8 or 16</td>
<td>Depending on version, either 8 or 16 bit register holding the data value read from memory.</td>
</tr>
</tbody>
</table>

**Audio Hardware:** The Digilent PmodAMP1 Speaker/Headphone Amplifier amplifies low power audio signals to drive either a stereo headphone or a monophonic speaker. The speaker is driven from the left stereo input and works only when plugged into the left input, i.e., the input closest to the round turn pot! The PmodAMP1 module is connected to the BASYS board through any of the four 6 pin edge connectors JA through JD. Use pin 1 for the PWM signal. The round turn pot on the PmodAMP1 module can be used to adjust the audio volume; turning it all the way counter clock wise produces the maximum output power.

Additional information regarding the flash memory and audio amplifier modules can be found here:

**Memory:**

**Audio:**
[http://www.digilentinc.com/Data/Products/PMOD-AMP1/PmodAMP1_rm_RevB.pdf](http://www.digilentinc.com/Data/Products/PMOD-AMP1/PmodAMP1_rm_RevB.pdf)
Write-Up

10.4.1. Describe the major components (and their function) of the audio player.
10.4.2. Hand in a well documented version of your Verilog code for both the 8 and 16 bit audio player.
10.4.3. What does the difference in audio quality between the 8 and 16 bit samples imply about the human hearing?
Write-up Format: Short


Read and work through all the in-text exercises for chapters 1, 2 and 3.

**Introduction**

LabView is a graphical programming software developed by National Instrument. It is widely used in science to interface instruments and to analyze data. This week you will become familiar with its graphical interface.

A large amount of information will be covered in the first three chapters of the LabView book. To avoid getting lost we recommend that you prepare yourself by reading a chapter before going to the lab. When you are in the lab, re-read the chapter again and complete all the assigned in-text exercises using the computers provided.

Note: For the rest of this semester, we request that you will work individually on these exercises, i.e., one person per computer! You certainly may discuss the exercises with your peers but each person must write his or her own code!

**Submitting Block Diagrams and Screen Shots**

In this and the following chapters you will be asked to hand in a large number of print outs and screen shots. Instead of printing individual hard copies for each one of them it will be far more efficient (and faster) if you capture your screens and then *copy / paste* them into a word document. Additionally, you may annotate those screen shots to describe what they represent. Using this approach your word document will serve as your lab note book, which, after some minor editing and cleaning up, can be submitted directly to your TA.

LabView Block Diagrams and Front Panels can be captured using *Edit >> Select All* followed by *Edit >> Copy* from the LabView menu bar. Next paste them into your word document using Word’s *Paste* command from the menu bar or press *CTRL-V*. To capture a screen, typically the front panel of a running program, press the *Alt and PrtScn (Print Screen)* button at the same time; paste it into your word document using the *Paste menu command* or *CTRL-V*. Depending on the size and details of the screen shots, you may want to resize them so that you can fit more than one on a page.
11.1. LabView Chapter 1

Read and work through all the in-text exercises of chapter 1.

Assignment

11.1.1. Complete the “Do It Yourself” exercise on page 42 and hand in its Block Diagram and Front Panel. See the hints below.

(Note: if you need to print the Block Diagrams, select File / Print from the LabView menu bar and press “Next;” select “VI Documentation” (again press “Next”) and then select “Block Diagram” and then press “Print.” Use this approach only for debugging; when handing in your assignments instead copy / paste the material into a word document as explained previously.)

Hints:
The output of the wait function (millisecond timer value) does not represent the number of milliseconds elapsed since you started your program or VI. Instead, it is related to how many milliseconds have elapsed since the computer was turned on. In other words, it's not very useful for displaying the time elapsed since the program has been started. On the other hand, the index variable (i) in the while-loop corresponds to the number of times the loop has iterated. By knowing this fact and the time delay between iterations, you should - with the help of the multiplication icon (Functions >> Programming >> Numeric) – be able to calculate the elapsed time accurately.

11.2. LabView Chapter 2

Read and work through all the in-text exercises of chapter 2.

Assignments

11.2.1. Complete the “Do It Yourself” exercise on page 74 and hand in its Block Diagram and Front Panel. See the additional comments below.

The Histogram VI’s Graph Output cannot be directly connected to the Graph Input. This problem can be fixed two ways:
   a) use an XY Graph control instead of the Waveform Graph (preferred method);
   b) connect the Histogram’s h(x) output to the Waveform Graph’s input.

If you’re using option b) be aware that h(x) represents the counts in each bin starting with the first non-zero bin. In other words, if you ran this exercise with a loop executing 10 times and you would get (the highly unlikely) outcome of 6 times 3 heads and 4 times 4 heads, h(x) would be {6, 4} and not {0, 0, 0, 6, 4}.

For the graph to display a bar graph you must change the LabView default settings. Right Click on the Graph and change under the Properties / Plots the following settings: the line type (see Figure 1 below), colors, and fill to zero.
11.2.2. Complete the “Random Numbers” exercise below and hand in its Block Diagram and Front Panel.

Random Number Exercise

Use the knowledge from the previous exercise to investigate random numbers and their distributions.

Part 1:

First, determine what a histogram of 100000 random numbers binned in 100 intervals looks like: create a for-loop that generates a 100000 random numbers; feed these into the Histogram VI (with the “interval” input set to 100) and plot its output in an XY Graph. Similar to the previous example, change the default display to a bar graph. Print the resulting graph. Explain in a sentence the shape of the histogram.

Part 2:

For the second part, explore how the distribution changes if we add some of the random numbers together before we bin them. Specifically, sum 11 additional random number functions inside of the for-loop; (remember the neat trick about holding down the CTRL key while dragging on a control to duplicate it?) inside the for-loop, use the compound arithmetic control to sum all 12 random numbers; subtract the (constant) value 6.0 from the compound sum. Again feed the resulting array (outside of the loop) to the histogram function. Run the loop 100000 times and display the histogram output. It should look similar to a Gaussian distribution. (Actually the distribution is a Lorentzian distribution, a close “cousin” of the Gaussian.) Explain in a sentence or two why the distribution looks different from the one in part 1.

11.2.3. Complete Homework 4, page 74. Hand in its Block Diagram and Front Panel.

Optional: if you’re curious, you could explore how adding additional harmonic terms to the series (for example (1/7)sin(7x)) improves the output.
11.3. LabView Chapter 3

You should read and work through all the in-text exercises of chapter 3. Note while the material in sections 3.1 through 3.10 is crucial to using LabView, sections 3.11 through 3.14 are a bit technical and teach you how to turn your VIs into custom controls, including how to make your own icons. While it would be tempting to skip sections 3.11 through 3.14, the text book will refer to the examples created in these sections in later chapters and, therefore, you must complete these in-text exercises. Make sure to save them in a place where you will be able to retrieve them at a later date.

Assignments

11.3.1. Complete the “Do It Yourself” exercise on page 119 and hand in its Block Diagram and Front Panel. See the additional comments below.

Use 200 samples at a sampling frequency of 10000 Hz, signal frequency, $f_{\text{sig}}$ of 2000 Hz and a modulation frequency, $f_{\text{mod}}$ of 50 Hz and amplitude of 0.5.

11.3.2. Complete Homework 4, page 120 and hand in its script and a graph of the resulting data. This exercise can be done entirely in Mathscript (remember to go to Tools >> Mathscript Window…). You do not have to create a VI if you figure out how to plot it in Mathscript.

11.3.3. Complete Homework 8, page 120 and hand in its script and write down the final fit equation in the form of a polynomial using the fit values $a_2$, $a_1$ and $a_0$. This exercise can be done entirely in Mathscript (remember to go to Tools >> Mathscript Window…) and you do not have to create a VI.
This chapter covers computer interfacing using a **data acquisition device**, or DAQ card. It converts an analog signal, typically a voltage, into a digital representation, (A2D conversion,) or vice versa, from a digital into an analog one, (D2A conversion.) These concepts are extremely important: while today’s technology is inherently driven by digital technology, most sensors in the physical world, including the ones we humans rely on such as eyes, ears, touch etc. are entirely analog. In other words, when we conduct a physics experiment that involves a computer, most likely we will make use of A2D or D2A conversions.

The two most important concepts involving D2A and A2D conversions are:

1. **sampling frequency** (for A2D), **conversion frequency** (for D2A)
2. **resolution** or **sensitivity**.

The **sampling frequency** specifies the rate at which we sample or acquire a signal using a A2D conversion. In its most simple form, it dictates the maximum frequency of the signal that we are able to detect, or in the case of an A2D conversion, to produce. For example, if you acquire a signal every millisecond, i.e., at a sampling frequency of 1 kHz, then you will not be able to deduce from your data correctly events that occur at much shorter time intervals, let’s say at microsecond intervals. If you want to detect such events you need to sample at least at twice the frequency of the fastest signal you want to observe. This is in essence the gist of the Nyquist theorem of which you will hear more about throughout the course.

While in theory it may seem straightforward to satisfy Nyquist’s theorem, in reality, extraneous signals, such as noise, can severely distort the acquired data. If the frequency of the noise is larger than the sampling frequency it still will be sampled; worst of all, it will appear in our digitized data at a frequency that is much lower than its original value through a process called **aliasing**. To avoid this effect, low pass filters with suitable cut-off frequencies need to be applied between the signal and the DAQ.

The second most important concept about DAQ cards involves the **resolution** or **sensitivity** of the device. It specifies the smallest incremental (voltage) value that the device will be able to resolve. In the absence of any noise, no matter what clever technique you apply, you will not be able to detect a signal change that is smaller than the resolution of the device. (Strangely, with added random noise you can actually go beyond the resolution through a process called **dithering** which will be covered in the next chapter.)

The resolution of a device depends on two factors: the DAQ’s voltage input or output voltage range and the number of bits used in the conversion. For example, a device capable of operating over a voltage range from +5V to -5V would cover a total range of 10 V. If such device were to have 16 bits then it would be able to resolve $2^{16}$ discrete levels over the entire range. In other words, its resolution will be $10V/2^{16} = 0.153$ mV.
12.1. LabView Chapter 4: Data Acquisition Using DAQ Assistant

Please read and work through the in-text exercises up to and including section 4.8. Our hardware differs from the one presented in the text book; therefore, read the supplemental notes below for the sections indicated.

Section 4.2

All the lab computers have National Instrument NI 6034E or NI 6036E data acquisition (DAQ) boards installed. These boards are very similar to the PCI-6251 boards mentioned in the book except that the NI 6034E lack the D2A conversion capabilities. (That’s the reason we will skip the sections after 4.8.)

The DAQ boards are connected through a 68-pin cable to a National Instrument I/O Terminal Block, or Brake-Out Box. It is either a National Instruments CB-68LP or a BNC-2110. The purpose of the Terminal Block is to allow for easy hardware connections through its BNC, screw, or crimp-type connectors. In other words, these devices themselves do not contain any electronics except (maybe) for fuses and some protective diodes.

Of the two break-out-boxes, the CB-68LP, shown below on the left, is the one that is completely compatible with the DAQ boards. If you use it, you will be able to follow all the instructions in the book. The drawback is that you will have to consult the pin out diagram in LabView (or below) to find the appropriate terminals and that it uses screw type terminals.
Figure 12.3: Pin out diagram for terminal block CB-68LP when connected to an NI 6036E or NI 6034E DAQ board. Note that pins 21 and 22 (AO 0/1 outputs) are not present on the NI 6034E.

The BNC-2011 boards, shown in Figure 12.2. on the right, uses BNC and push down crimp connectors. All its inputs and outputs have already been conveniently labeled. Unfortunately, National Instruments has recently revised some of the terminal names. While the new names are used in LabView and the text book, the labels on the box have not been updated. In case of uncertainty, consult the table shown below listing the terminal name equivalents.

While this box makes "cleaner" hardware connections then the CB-68LP, you should be aware that it does not have connectors for all the 68 terminals of the DAQ card. Some of the more obscure signals are not wired to the box. Luckily this will not affect us, at least for now.

**BNC-2011 Terminal Name Equivalents**

With NI-DAQmx, National Instruments has revised its terminal names so they are easier to understand and more consistent among National Instruments hardware and software products. The revised terminal names used in this document are usually similar to the names they replace. Refer to the following table for NI-DAQmx equivalents for some Traditional NI-DAQ (Legacy) terminal names.

<table>
<thead>
<tr>
<th>New NI-DAQmx</th>
<th>Old Traditional NI-DAQ (Legacy)</th>
<th>New NI-DAQmx</th>
<th>Old Traditional NI-DAQ (Legacy)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI #</td>
<td>ACH#</td>
<td>CTR 0 GATE</td>
<td>GPCTR0_GATE</td>
</tr>
<tr>
<td>AI # +</td>
<td>ACH# +</td>
<td>CTR 0 OUT</td>
<td>GPCTR0_OUT</td>
</tr>
<tr>
<td>AI # -</td>
<td>ACH# -</td>
<td>CTR 0 SOURCE or CTR 0 SRC</td>
<td>GPCTR0_SOURCE</td>
</tr>
<tr>
<td>AI GND</td>
<td>ACHGND</td>
<td>CTR 1 GATE</td>
<td>GPCTR1_GATE</td>
</tr>
<tr>
<td>PFI #</td>
<td>ACK#</td>
<td>CTR 1 OUT</td>
<td>GPCTR1_OUT</td>
</tr>
<tr>
<td>AI GND</td>
<td>AIGND</td>
<td>CTR 1 SOURCE or CTR 1 SRC</td>
<td>GPCTR1_SOURCE</td>
</tr>
<tr>
<td>AI SENSE</td>
<td>AISENSE</td>
<td>P0.#, P1.#, P2.##...</td>
<td>PA#, PB#, PC#...</td>
</tr>
</tbody>
</table>
Section 4.7.

There is a mistake in the text book on page 136: The DAQ Assistant is to be found under: Functions >> Express >> Input.

On page 140, LabView will not list the BNC 2110 Terminal Block in the connection diagram. If you use the BNC 2110, use the labels directly on the Terminal Block to connect your signals. If you cannot find the appropriate label, consult the table above.

Section 4.8.

The method for handling the DAQ Assistant’s error out, shown in the text book on page 148, is lacking. While it may display the error it will not terminate the while loop and can, therefore, create potentially more serious errors. A better method for handling this kind of error is shown in the text book on page 158. It OR’s the error with the STOP button and terminates the while loop and the DAQ Assistant in the event of an error.

Assignments

(You may skip the Do It Yourself section for this chapter since some of the boards do not have D2A capabilities.)

12.1.1. You are given the job of designing a measurement instrument such as a DVOM. The instrument has an n-digit decimal display similar to the ones you are familiar with. Your job is to
figure out the minimum number of data bits \( m \) that the ADC must have to satisfy the accuracy of the \( n \)-digit display. (For example, the accuracy of a \( n = 3 \) (decimal) digit display should be 1 in 999 but for simplicity assume the accuracy to be 1 in 1000). Assume that no sophisticated signal processing techniques are used and that the data displayed is acquired in a single ADC conversion!

a) Find an equation that relates the number of \( n \) decimal display digits to the number of bits \( m \) needed in the ADC.

b) If you want to build a 8 (decimal) digit display unit, how many bits must the ADC have?

12.1.2. Figure out the “important” specs for 6034E (or 6036E) DAQ board using the link in the text or wherever else you can find it. From the data obtained, explain:

a) What is the frequency of the fastest signal that can be detected without aliasing?

b) With its maximum input voltage range, what is the smallest voltage change that can be measured in a single measurement? (Though the DAQ card has built in amplifiers that can be programmed to extend the range of the DAQ card, we assume for now a unity gain.)

12.1.3. Complete Homework 4.2, page 169. Hand in its Block Diagram and Front Panel. Use the program you wrote in in section 4.8. and input a 0.2 Vpp, 1 Hz sine wave. Use 100 samples and set the DAQ sampling rate to its maximum. Answer the questions in the text. Compare the answer to previous exercise.

12.2. **LabView Chapter 5: Data Files**

Please read and work through the in-text exercises up to and including section 5.5. Glance at the remaining sections in this chapter in case you are required to write more complex data to a file.

**Additional Comments and Hints Regarding Writing to Data Files**

1) **Delimiter**: By default, the LabView *Write To Spreadsheet* function separates data columns with Tab characters; it’s easier to work with numerical data if the columns are delimited instead with a coma. (You’ll see why this is the case in the next comment.) Click on the bottom right tab of the *Write To Spreadsheet.vi* and enter in the delimiter field a “coma.” See the example below.
2) **Data File Type:** When saving your comma delimited files, save them with the CSV extension; for example, mydata.csv. When you double click on this file, it automatically will open in Excel. There’s no need to go through the cumbersome Excel import function.

3) **Formatting:** If you simply use %E you will get the maximum precisions in your output data file and you do not have to worry about losing accuracy. If you don’t like this type of formatting, you can always change it later within Excel.

4) **Path Name:** if you do not want the user interface to pop up each time you save a file, specify its file and path name as shown in Figure 12.4. using the path constant as shown in figure 12.4. The path name to your folders is:

   U:YOURX500_USERNAME\Documents\LabVIEW Data\FILENAME.CSV

   Of course, you must replace YOURX500_USERNAME with your personal information and it helps to select a meaningful name for your FILENAME.

**Assignments**

Please read and work through the in-text exercises up to and including section 5.5. but you don’t have to hand anything in.

When everything works correctly continue with the next exercise.

**12.3. Physical Pendulum Data Acquisition Exercise**

In this exercise, you will measure and record the angle of rotation of a physical pendulum as a function of time after it has been released from its rest position. From the data you will then calculate the free period of the pendulum and compare it to the expected value.

There are three aspects to this exercise: first you will need to write a LabView program to acquire and store the data in a CSV file. Second, you will need to connect to physical pendulum to the power supply and computer to acquire the data. Third, and last, you will need to analyze the data.

**Part 1: LabView Program**

Note: there are currently only three physical pendulums in the lab. If none is free for this part of the exercise, use a function generator with a 1 Hz, 1 Volt sine wave amplitude to simulate the signal from the physical pendulum.
Figure 12.5. Simple version of physical pendulum data acquisition program.

The data acquisition program uses the DAQ Assistant and the Write to Spreadsheet File functions. You should be familiar with them from the previous exercises. In addition, the (optional) Waveform Graph displays the acquired data after all the data has been acquired. Because all the data will be acquired and stored in one single step once the user starts the program the program is not enclosed within a while-loop and, therefore, needs no stop button.

The DAQ Assistant has been configured to a +5 to -5V range, a 50 Hz data rate and a timeout value of 100 seconds. (The timeout value must be larger than the entire acquisition process time; therefore, the default value of 10 seconds will be too short for this application and must be replaced by 100 seconds.) The number of samples is set by a control and determines how many periods of oscillations the physical pendulum data will be acquired. Set it to a value that allows acquiring 10 to 20 periods of oscillations.

You should build the above program and test it. If there are no physical pendulums available, simulate the data with a function generator with a 1 Hz, 1 V amplitude sine wave.

Once you have acquired the data from either the pendulum or the function generator, you will notice one short coming of the program: while the graph displays a nice XY graph, the spreadsheet data only shows one column, V(t), and it omits unfortunately the actual time values. While this could easily be remedied in Excel we want you to modify the above program so that the data written to the spreadsheet consists of two columns, the time and voltage values.

Therefore, you will need to create an array consisting of the corresponding time values. This can be accomplished with the aid of a for-loop. From the value for the sampling rate you will be able to calculate the appropriate time value for a specific array element. This time array is then combined with the voltage array using the Build Array function. An incomplete diagram of how you might go about it is shown below.
Figure 12.6. Incomplete Block Diagram of physical pendulum data acquisition program: the broken wires indicate the signals to be used to generate the time array.

Test your program and once you have convinced yourself that it works proceed to the next section.

**Part 2: Physical Pendulum or Rotational Measurements**

We will now monitor the voltage across a low friction turn "pot" to measure rotational angles. The physical pendulum is directly connected to the "wiper" (pin 2 in the diagram below) of a 5kΩ precision turn pot. A constant voltage from a lab power supply is applied across the entire pot: pot terminal 1 is connect to +5V and pot terminal 3 is connected to -5V. The voltage measured at the wiper, pot terminal 2, $V_{wiper}(t)$, is proportional to, $\theta(t)$, the angle of rotation. The relationship between $V_{wiper}(t)$ and $\theta(t)$ can be calculated from knowing that a 360 degree rotation corresponds to the voltage drop applied between terminal 1 and 3.

Since there are currently only two physical pendulums in the lab, work on this part only after completing the previous part successfully. Also, be careful handling the pendulums; they are fragile.
**Part 3: Rotational Data Analysis using Excel**

Open the data file obtained in part 2) in Excel. Plot the data quickly to confirm that it corresponds to what you expect to see. In Excel, convert the voltage data into degrees and now plot \( \theta(t) \) vs. t of the physical pendulum.

**Assignments**

12.3.1. Hand in the Block Diagram and Front Panel for part 2 of the exercise above.

Also hand in a hard copy of the Excel plot of \( \theta(t) \) vs. t. From the plot, measure the (average) period of oscillation and state it directly on the plot. With the ruler provided, measure the length of the pendulum and calculate the free period. If you want to be more exact, you should have used the physical pendulum equation but for this exercise the simple pendulum approximation will suffice. Show how you calculated the free period of the pendulum and compare it with the value obtained from Excel plot!
13.1. LabView Chapter 6: Shift Registers

Reading

Chapter 6 of the LabView text deals (briefly) with shift registers, manipulating arrays and the creation of “Sub Vis.” For the sake of brevity we will concentrate only on shift registers. Carefully read the introduction to chapter 6, pp. 211 – 213. Time permitting you may also want to browse through sections 6.4 to 6.7., pp. 220 – 241.

Introductions to Shift Registers

Shift registers are variables that depend on their history. The analog in digital logic circuits are sequential logic gates, such as flip flops. The current state of such a gate can only be determined if the previous one is known.

To explain the use of shift registers consider the following example. Assume that we want to evaluate the following sum in LabView where \( N \) is a user controlled value:

\[
\sum_{i=0}^{N-1} i
\]

This expression can be evaluated inside of a loop, for example in a for-loop. (As an additional benefit, a for-loop executes \( N-1 \) times and automatically generates the required \( i \) values.)

The sum can be computed by breaking the process down into \( N-1 \) individual summing steps, each involving only the sum of two terms, namely:

\[
sum_i = sum_{i-1} + i
\]

with \( sum_0 = 0 \).

In other words:

- \( sum_0 = sum_1 + 0 = 0 + 0 = 0 \)
- \( sum_1 = sum_0 + 1 = 0 + 1 \)
- \( sum_2 = sum_1 + 2 = 1 + 2 = 3 \) etc. up to \( sum_{N-1} \).

This approach works but it wastes resources. We are only interested in the final sum, \( sum_{N-1} \), and we do not want to store all the intermediate results such as \( sum_0 \) through \( sum_{N-2} \). This shortcoming is remedied by using a temporary variable, or shift register. For now, let’s name this temporary variable...
variable acc (short for accumulator.) (Actual shift registers in LabView have no name assigned to them.)

Rewriting our pseudo code from before we obtain:

\[
\begin{align*}
acc &= 0 \\
acc &= acc + 0 \\
acc &= acc + 1 \\
acc &= acc + 2 \text{ etc. up to } acc = acc + N-1.
\end{align*}
\]

This approach is more efficient than the previous one because we require only one additional variable, acc, and it is how we will implement summing in LabView. See the sample code below:

Here are a couple more comments regarding the pseudo code shown above, specifically the statement:

\[
acc = acc + i
\]

To understand its meaning you must not apply mathematical reasoning to it because it would produce non-sense, i.e., 0 = i. Instead you need to approach it from a sequential viewpoint which breaks down into three distinct states:

1) The right hand side of the expression is evaluated: \( acc + i \). In the LabView diagram shown above, the value of acc is obtained from the left shift register and then \( i \) is added to it.

2) The result from step 1) is stored in a temporary variable. You may visualize this as the result of the addition being temporarily stored in the shift register on the right loop border.

3) The temporary value is finally assigned to the left hand side of the expression, i.e., acc is replaced by its new value. In other words, before the loop iterates again, the value in the right shift register is copied to the left shift register.

4) The loop continues with the next iteration at step 1 again.

We will have more to say about shift registers in the next section; for now let's practice using them.

**Assignments**

13.1.1. Write a LabView program that calculates the following sum:

\[
sum = \sum_{i=0}^{N-1} i \quad (13.1.1)
\]

In your program, let the user control \( N \) and then display the result. You may only use the Programming >> Numeric >> Add and Subtract function, a shift register and a comparison operator.

Implement the problem in two different ways: for one version use a for-loop; for the other version use a while loop. In each case display the resulting sum. (If your code works correctly, the sum
from the two implementations should be identical.) You may write two different programs or implement both approaches in the same program, using two displays.

Run the program a couple times and notice what happens if you do not initialize the shift register: try it with a constant of 0 connected to the left register and without it.

You may also notice a difference in the sums for the two different implementations: the for-loop executes N-1 times while the while-loop executes N times. This issue was addressed in section 2.9 of the LabView book. One way to remedy this issue is to use a “subtract” or “decrement” operator for the index variable, i, in the while-loop.

Hand in the Block Diagram and Front Panel for both versions; in each case use $N = 1\,000\,000$ and check that the answer is correct, i.e., $\text{sum} = (N/2)/(N-1)$. If it is not, check the data representation of your input values, including the shift registers and use suitable data types!

13.1.2. Calculate the $i^{th}$ term of a Fibonacci series, which is defined as:

$x_i = x_{i-1} + x_{i-2}$ with $x_0 = 0$, $x_1 = 1$, $x_2 = 1$, etc.

In other words the series is: 0, 1, 1, 2, 3, 5, 8… etc. Let the user enter the value for $i$ (assume that it is always 3 or larger) and then calculate and display $x_i$.

Hand in the Block Diagram and Front Panel displaying the result for $i = 100$. (The answer should be in the $10^{20}$ range; if not, check the data representation of your input values, including the shift registers and use suitable data types.)

13.1.3. Average and Standard Deviation:
In this exercise, you will write LabView code to calculate the average and the standard deviation of a set of random values contained in an array. To check the results, you will compare these values to the ones provided by LabView’s advanced analysis functions.

You may find the following equations for the average, $\bar{x}$, and the standard deviation, $\sigma$, useful:

$$\bar{x} \equiv (x) = \frac{1}{N} \sum_{i=0}^{N-1} x_i \quad (13.2.\, a)$$

$$\sigma = \sqrt{\frac{1}{N-1} \sum_{i=0}^{N-1} (\bar{x} - x_i)^2} \quad (13.2.\, b)$$

a) Generate an array with N elements and assign each array element a random value between 0 and 1. You may think of this array as a data set that was acquired during some measurement. Implement it using a for-loop and the Mathematics >> Numeric >> Random Number function. Use a control to allow the user specify the value for $N$. For the rest of this exercise set $N$ to 100000.

b) Calculate and display the average of the data in the array using LabView’s Mean function, found under Mathematics >> Probability and Statistics.

c) Write your own code to calculate the mean using only the simple addition and division functions found under Functions >> Programming >> Numeric and shift registers. You may either place your code into the for-loop that you used to create the random values, or, (preferably) into a second for-loop whose sole function is to calculate the mean. Display your result and compare it to the one obtained in b); they should be identical if your code works.
d) Calculate and display the standard deviation of the data in the array using LabView’s **S.D. & Variance** function, found under **Mathematics >> Probability and Statistics**. (Note: that the **standard deviation** VI has an output for the mean; so theoretically, we could have used it also for part b.)

e) Write your own code to calculate the standard deviation (see equation 13.2.b.) using only the simple addition, subtraction, division, square and square root functions found under **Functions >> Programming >> Numeric**. Of course, you will need some shift registers too. You may either place your code into the for-loop that you used to create the random values, or, (preferably) into a third for-loop whose sole function is to calculate the standard deviation. Display your result and compare it to the one in d); if your code works, the values should be identical.

f) The predicted standard deviation for a set of uniformly distributed random numbers, like the ones generated in this exercise is: \( \sqrt{\frac{1}{12}} \). Calculate and display the relative percentage error of your result from part e) using the equation: \( \frac{\Delta x}{x} \) where \( \Delta x \) represent the difference between the expected and the obtained value and \( x \) is the expected value.

Hand in the Block Diagram and Front Panel displaying the results for \( N = 10^7 \).

13.1.4. Do the “Do it yourself,” part a) only, of chapter 6, on page 241, to calculate the **factorial**. You may find that you will need to use the **Select-function** found in **Programming >> Comparison**. It functions similarly to an if-statement in procedural languages or like a multiplexer in digital logic: depending on Boolean value of the center (control) input, it will either select the top or bottom input and pass it on to the output.

Hand in the Block Diagram and Front Panel for part a) only of the “Do It Yourself” displaying the result for 15!

### 13.2. Data Acquisition & Signal Processing I: Data Averaging Simulation

In this and later sections we will discuss techniques for signal processing. These methods address the fact that experimental data is almost always “contaminated” with noise. By applying the appropriate signal processing method we would like to remove the noise without affecting the actual signal.

This technique, discussed in this section, involves the averaging of the data, explained in the steps below:

1) Data of a repetitive signal is acquired, usually over multiple signal cycles.
2) The acquired data array is added, element by element, to an accumulator array.
3) Step 1 and 2 are repeated multiple times and care is taken that each data acquisition always starts at the identical phase with respect to the original signal.
4) Finally, each element of the accumulator array is divided by the number of data acquisition cycles performed in step 3 to produce an average value.

What does it do? First consider a complex sinusoidal signal without any noise with amplitude \( V_s \) and phase \( \phi \). If we acquire the signal always at the same phase and then add it to the accumulator signal, the amplitude of the signal in the accumulator will be \( N V_s \), where \( N \) represents the number acquisition cycles. See a phasor diagram of this process shown in Figure 13.2.
13.2. Data Acquisition & Signal Processing I: Data Averaging Simulation

If we divide the added signal by \( N \), we essentially retrieve our original signal and find that:

\[
\langle V_s \rangle = V_s \quad \text{(true only if phase-locked)} \quad (13.3.)
\]

It may seem that we have not much accomplished but in fact we did: at least we haven’t distorted our original signal which is no small feat!

To understand the power of this method it must be considered in the presence of noise. Assume for now that no “real” sinusoidal signal is present, only Gaussian random noise with an average amplitude \( V_n \). As is the nature of random noise, no phase can be associated with it since after all noise is random in its nature.

Applying again the averaging procedure outlined above results in the noise signal increasing in the accumulator like \( V_n \sqrt{N} \). (The reason for the increase being proportional to \( \sqrt{N} \) can be found in the statistical treatment of the random walk problem where it is found that after \( N \) identical sized steps in random directions, an object will have moved (on the average) \( \sqrt{N} \) steps away from the origin.)
Dividing the accumulator sum by \( N \) yields after \( N \) data acquisition cycles an average noise amplitude of:

\[
\langle V_n \rangle = \frac{V_n}{\sqrt{N}} \quad (13.3.)
\]

Let’s combine what we have learned so far and consider a signal, \( V \), composed of signal and noise:

\[
V = V_s + V_n \quad (13.4.)
\]

As long as the noise and the signal are not correlated, we then find that after \( N \) phase-locked averages, we get:

\[
\langle V \rangle = V_s + \frac{V_n}{\sqrt{N}} \quad (13.5.) \text{ (This is only true only for phase locked signals!)}
\]

In other words, the averaging technique results in an output signal that is identical to its noise free input signal and that has a noise component proportional to \( 1/\sqrt{N} \). Therefore, its signal-to-noise-ratio (SNR) will improve as \( \sqrt{N} \). Ideally, if we averaged over an infinite number of data cycles, we could (theoretically) filter out the noise entirely.

When using this technique keep in mind that all of the following conditions must be satisfied for it to work:

a) The signal to be detected is repetitive and does not change during the averaging process;
b) We are able to “synch” or “phase-lock” with the signal, i.e., we know its phase;
c) Its noise component is random not correlated to the signal.

In an experimental setting, the condition that might be the most difficult to satisfy is b): often we have no information about the signal’s phase. If condition b) is not satisfied the summed signal will grow in the accumulator proportional to \( \sqrt{N} \). In other words, the signal will grow at the same rate as the noise and nothing will be gained from this technique! (In the next chapter, you will learn a method for overcoming this obstacle.) For now it suffices to say that averaging a non-phase locked signal in the time domain is simply meaningless.

**Part 1: Display the Waveform (No Averaging)**

Create a LabView program to display our noisy test signal, similar to the one shown below:
Your block diagram should contain the following elements:
1) Create a while-loop with a stop button to terminate the loop.
2) Inside of the while-loop, add a Wait timer to delay the loop execution and set it to 100 msec.
3) Add a numeric indicator to display how many times the loop has executed. Label it "N". You will use this value later to calculate the average. It might be helpful to test your program at this point; is "N" updating continuously? Can you stop your program by pressing “STOP.”
4) Create a simulated signal source with noise: select Express >> Input >> Simulate Signal and drop it into the while-loop. Configure it as shown below:

![Simulate Signal property window](image)

- Figure 13.5. Simulate Signal property window.

5) Expand the “Simulate Signal” icon by dragging at its bottom and wire a numerical control to its “Standard Deviation” input to control the amount of noise in the signal. Again place that control inside the while loop.
6) Wire a (Boolean) control to the Simulate Signal “Reset Signal” input. Change its label to “Phase-lock” and place this control inside the while loop.
7) On the front panel, add Modern >> Graph >> Waveform Graph; make sure it is has been placed inside of the while loop and wire its input to the output of the Simulate Signal icon.
8) Test your program. Can you adjust the amount of noise in the signal?

You will not have to hand in anything for this part.

**Part 2: Summing the Waveforms**

Now add the functionality to your program that will perform continuous summing of the waveform data.

You have previously built programs that sum and average values relying on shift registers; see assignment 13.1.1. and 13.1.2. What makes this case different is that we are now summing an entire array to a second array, element by element. In other words, while in the previous section we were dealing with a single shift register we now need an entire array of shift registers to accommodate every single one of the 1000 array elements created by the Simulate Signal function.
Lucky for us, by wiring the output of an array to a single shift register, LabView “somehow” senses that we are requesting an entire array of shift registers and without further ado takes care of that part. On the other hand, what LabView still does not do, is to initialize all the shift registers. You still have to tell it explicitly to do that. Unfortunately, popping up on the left shift register and creating an (array) of constants initialized to 0 does not seem to work either. Instead, you need to create and initialize a dynamic data array (DDT) with 1000 elements (outside of the loop) and connect that to the left shift register.

A simple way of creating such an array is use another instance of the Simulate Signal function. Copy the existing Simulate Signal function and paste it outside of the while-loop. Since we want the shift register to be initialized to 0, we wire the Amplitude and Standard Deviation input of the newly created Simulate Signal to a constant value of 0. Finally, if you haven’t done so already, create a shift register on the while loop and wire the output from the newly created Simulate Signal icon to the left shift register. (See the partial block diagram below.)

Similar to assignment 13.1.3., part c), you can now wire the left shift register to one of the inputs of the addition function; wire the output from the Simulate Signal function that is inside the loop to the other input of the addition function. Finally connect the addition function’s output to the right shift register.

Add a second Waveform Graph inside the loop and connect it to the accumulator. (See above.)

Run your program for about 100 cycles with a noise amplitude of 0.1 Standard Deviations and observe the summed data.

First, run the program with the “Phase-lock” control turned on. Press Alt-PrtScn and paste the resulting front panel screen shot into your lab document. Check the summed signal amplitude; what should it be? (Hint: see the Simulate Signal properties.)

Second, run the program with the “Phase Lock” control turned off. Press Alt-PrtScn and paste the resulting front panel screen shot into your lab document. Check the summed signal amplitude; what should it be? As the program runs, you may observe the original signal in the summed signal disappearing and reappearing. What do you think causes these fluctuations?
**Assignment**

13.2.1. Hand in the screen printouts of about 100 averaging cycles with the Phase-lock set to on and off. Explain in each case the expected signal amplitude. Explain the strange case of the signal appearing and disappearing when the Phase-lock is turned off.

**Part 3: Averaging the Waveforms**

After all the work from the previous sections, the final averaging process consists simply of dividing the summed output by \(N+1\) where \(N\) represents the number of loop iteration. Implement this function in your program. (If you want to be picky, you should connect the display indicating the numbers of averages performed to the \(N+1\) output instead of the loop index.)

Disconnect the second Waveform Graph's input from the accumulator and instead wire it now to your averaged sum. Test your program and once you're confident that it works hand in the Block Diagram and Front Panel.

If you don’t want to measure the amplitude of the averaged signal by eye, you could connect the Tone Measurement VI (found under Signal Processing >> Waveform Measurements) to the averaged waveform. Connect its Amplitude output to a display. (If you prefer peak-to-peak values instead, multiply the amplitude value by 2.) Run it and check that it agrees with the amplitude value set in the Simulate Signal function.

**Assignments**

13.2.2. Hand in a print-out of the Block Diagram and the Front Panel of your program.

13.2.3. Run your program for about 100 cycles with a noise amplitude of 1 Standard Deviations and Phase-lock turned on. Press Alt-PrtScn and paste the resulting front panel screen shot into your lab document. Check the averaged signal amplitude; what should it be?

(For your own curiosity, repeat this exercise but turn the Phase-lock off. You do no need to hand in this result.)

13.2.4. Finally, ask yourself how many phase-locked averaging cycles you will need to perform to discern a signal; assume that you happen to know the signal's original signal-to-noise ratio (SNR). (Of course the term “discernable” is subjective but for now assume that the final signal we want to observe will have a noise amplitude equal to its signal amplitude.)

For this exercise, generate an input signal with an SNR of 1/33, i.e., set the “Standard Deviation” of the Simulate Signal generator to 33. Calculate the number of phase-locked averaging cycles that will be required to obtain an averaged signal with an SNR of 1. (You may want to decrease the Wait Timer's delay time to 50 msec to speed things up.) Run your program and Press Alt-PrtScn when you observe this signal; paste the resulting front panel screen shot into your lab document and explain how you calculated the required signal averaging cycles.

**13.3. Data Acquisition & Signal Processing II: Data Averaging Using the DAQ**

In the previous section you applied phase-locked averaging and reduced the random noise on a simulated signal. In this section, you will use the same technique but instead of “playing” with a simulated signal, you will employ a “real” signal from your data acquisition card.
The programming aspect for this exercise will require one modification of your previous code: you will replace the Simulate Signal VI inside of the while-loop with the DAQ Assistant which you have previously encountered. Before we get to the details of programming let’s explain a few concepts in more detail:

**Phase-locking With Signals**

In the previous exercise, we achieved phase-locking by forcing the signal simulator to start generating the signal always at the same starting point, for example at \( t = 0 \). Though there are (some) real world signal sources that can be triggered externally to start their signal generation, more commonly one instead “locks-in” on a generated signal. The 3 most common scenarios involve:

- a) The signal source outputs a trigger or a sync signal that is in phase with the signal;
- b) The signal is large and “clean” enough so we can trigger on a threshold value;
- c) We employ a fancy technique called phase-locked loop, PLL for short.

Method a) is by far the simplest to employ; that’s why most function generators, such as the ones used in the lab, have a SYNC out signal that represents a digital signal in phase with its analog signal. Since the digital signal transitions are very abrupt, it is straightforward to obtain a clean trigger source to lock-on. This is the method we will use in this exercise. (Note: generally it is not important at what phase the digital transitions are with respect to the analog signal. What matters is that they, independent of frequency and amplitude, remain constant with respect to the analog signal.)

Method b) is based on the assumption that you already have a fairly clean signal and that you will trigger at some preselected level and slope, i.e., either when the signal is raising or falling. This is the method employed by most oscilloscopes to bring you a stable display. Its drawback is that it only works well with clean signals, thereby, making signal processing techniques, such as the data averaging discussed here, superfluous.

Method c), PLL, involves the creation of second, clean, simulated signal that is continuously being compared and adjusted to match the original input signal. It involves the use of a negative feedback technique, similar to the one we encountered with op-amps. We will not discuss the specifics of PLL technique any further but it is used in instruments, such as Lock-In Amplifiers, where a very stable and clean signal source is required that stays in synch with the input signal.

**Creating a Tiny Signal**

When testing or calibrating an experiment, it is useful to have a well-known test signal. In many cases this involves a very tiny signal on the order of a few micro Volts or even less. On the other hand, the output signal from a function generators is usually 0.1 V or larger. You may ask yourself why this is the case and why device manufacturers are not more accommodating in making signal sources that output very small signals.

There are two good reasons for this. First, if you start with a large signal, then you can monitor or measure it with standard equipment such as oscilloscopes and your typical DVM. Second, it is relatively simply to create a smaller signal with a simple voltage divider or an RC circuit. This method works very reliably and accurately and though we may no longer be able to measure or observe the reduced signal, we know what the original one is.

In this experiment we will use the signal from the function generator and reduce it by a factor of 1000 using a 1 kΩ and a 1 MΩ resistor.
Limitations on Small Signals Detection Due to A2D Resolution

In section 13.2., you were able to detect a signal with a SNR of 1/33 using a large number of phase-locked averaging cycles. You may ask yourself if there is a limit to this technique. In other words, if you were patient enough and had an unlimited amount of (phase locked) data, could you average the noise away to detect an infinitesimal signal? It turns out that for the simulated data, the limit is set by the numerical accuracy of the averaging process. Therefore, if your code had variables with an infinite accuracy, yes, you could theoretically detect an infinitesimal signal, though it may take you an infinite amount of time.

In contrast, the exercise in this section uses a signal that has been converted from analog-to-digital by the A2D card. Since digital literally means discrete, we may wonder what the smallest signal is that we are able to detect when acquiring the signal with an analog-to-digital converter.

You may recall from the previous chapter that a unique property of a DAQ is its resolution. It is defined as the smallest voltage change that results in a change in the A2D’s output. It is calculated by dividing the A2D’s input voltage range by $2^n$, where $n$ corresponds to the number of bits that the A2D is capable of outputting. For the settings employed in this exercise, i.e., a 16 bit A2D with an input voltage range of 10V this corresponds to about 153 μV. To repeat: it takes at least a voltage change fo 153 μV for the A2D’s output value to change; for voltage changes smaller than 153 μV the A2D’s output value will remain constant.

From this we may conclude:

a) as long as our input signal is larger than the DAQ’s resolution we should be able to detect it;

b) if the input signal is smaller than the resolution than we will not be able to discern it.

It turns out things are a bit more complicated. While conclusion a) is essentially correct, it turns out that the accuracy of our measurement is proportional to the ratio of the input signal to the resolution of the DAQ. For example, a signal that is only 10 times as large as the resolution will produce an error of 1/10, i.e., 10%, no matter how long we average. For signals that are even smaller, the error in the detected signal will drastically increase.

Contrary to common sense, conclusion b) is neither correct nor incorrect. As you will see in the ensuing exercise, we are indeed able to detect signals that are smaller than the DAQ’s resolution but we are only able to detect them because of the presence of noise in the system. This is one of the few cases where the addition of noise, a process called “dithering,” actually aids in detecting a signal. Why adding random noise to a signal improves the sensitivity of your equipment is not obvious and for a more scientific discussion you should read the original paper by the inventors of this process.1 Nevertheless, be forewarned: while one is able to detect such tiny signals, the information about its amplitude is not very accurate.

---

LabView Program for Phase-Locked DAQ

Open your program from the previous section und use “Save As” to save it under a different name.

Replace the Simulate Signal VI inside of the while-loop with the DAQ Assistant. Also remove the associated Phase Lock and Standard Deviation controls. (Leave the Simulate Signal VI outside the loop as it is.) Open the DAQ Assistant in Express >> Input >> DAQ Assist and drop it inside the while-loop. Since it is an Express VI it will ask you for the various selection parameters. Choose an input channel: Acquire Signals >> Analog Input >> Voltage >> a0.

Configure the DAQ’s input settings as shown. Make sure the Samples to Read and the (Sampling) Rate correspond to the values used in the Simulate Signal VI and are identical to ones as shown on screen to the right, i.e., 1k and 10 kHz.

Phase locking is achieved by setting the trigger settings as shown in the panel on the left.

After having made these changes hit the “OK” button and reconnect the DAQ Assistant’s Data output wire.

Hardware Setup

For testing purposes, feed a 5 VPP, 100 Hz, sine wave from the function generator’s output into channel 0 of your terminal block. (Consult the DAQ Assistant’s Connection Diagram to find the appropriate pins on the CB-68LP terminal block. On the BNC 2110 blocks, set the analog input switch to FS, Floating Source.)

To phase lock the DAQ card, feed the function generator’s SYNC output into the PFI 0 input on the terminal block. (On the CB-68LP terminal block PFI 0 corresponds to pin 11 while the corresponding digital ground should be connected to pin 12.)

Turn on the function generator and then run your program. If all is well, your averaged signal display should correspond to a sine wave of 5 VPP, independent of the number of averages.
Small Signal Generation & Measurement: Part 1

First we will measure a few signals that are larger than the resolution of the DAQ. Reduce the function generator's 5 VPP analog signal by a factor of 1000. Build a voltage divider composed of a 1k Ω and a 1 MΩ resistor and feed the voltage divider's output into the A2D input at the terminal block.

Run your program and measure the phase-locked averaged signal's Peak-to-Peak voltage after approximately 100 averages. Use ALT-PrtScn to capture the image and paste it into your lab document. Explain what amplitude you expected to read and what the percentage error was in the reading. Calculate how many times the input signal was larger than the DAQ's resolution.

Repeat the previous process two more times: reduce the function generator's output voltage to 1.5 VPP and to 0.5 VPP.

Small Signal Generation & Measurement: Part 2

Finally, let's measure an input signal that is smaller than the DAQ's resolution. Reduce the voltage divider to a 1:10000 ratio by replacing the 1k Ω with a 100 Ω resistor. Set the function generator's output voltage to 1 VPP and average the signal over at least 200 cycles. Use ALT-PrtScn to capture the image and paste it into your lab document. Again measure the amplitude of the detected signal. Compare it to the expected value and calculate how many times the input signal was smaller than the DAQ's resolution. Finally, repeat it with a 0.1 VPP signal.

Assignments

13.3.1. Hand in the Block Diagram of your program.

13.3.2. Hand in the screen shots of the phase locked signal with the 1:1000 voltage divider and the 5 VPP, 1.5 VPP and 0.5 VPP output voltage. In each case, what was the size of the observed signal amplitude? How does it compare with the expected amplitude? How does the percentage error relate to the ratio of DAQ resolution / DAQ input signal?

13.3.3. Hand in the screen shots of the phase locked signal with the 1:10000 voltage divider and the 1 VPP and 0.1 VPP output voltage. In each case, what was the size of the observed signal amplitude? How does it compare with the expected amplitude? How many times smaller was the input signal than the DAQ's resolution?

13.4. LabView Chapter 7: Case Structure

Reading

Chapter 7 covers the "case structure." Only read the introduction to chapter 7, pp. 249 – 251 and skip the rest on numerical integration.

Assignments

13.4.1. Complete the “Do It Yourself” exercise on page 268 and hand in its Block Diagram and Front Panel. (Hints: of the many ways to implement this, one method is using a nested loop: place a for-loop...
inside of the timed while-loop; another way is to use the “Remainder” operator from the Quotient & Remainder Function which can be found under Numeric Functions. It calculates the remainder of the integer division of \( x/y \) and hence, it will output a value that goes from 0 to \( y-1 \).

13.4.2. Complete Homework 7.1, page 269 and hand in its Block Diagram and Front Panel.

13.5. LabView Chapter 8: Sequence Structure

Reading

Read all of Chapter 8; work through the in-text exercises.

Assignments

13.5.1. Problem 8.5. on page 297 introduces an important concept called Property Node Values. They are used when one user control is required to control various objects at multiple places in a program. Work through problem 8.5. and note that for part c) you will yet need one more property node with the “Value (Signaling)” property.

Hand in its Block Diagram and Front Panel for part c and explain briefly why part a) does not work as intended.

13.5.2. Problem 8.5. on page 297 uses Property Node Values. Some LabView users consider them inferior in performance and instead prefer Local Variables. However, this might be, from a programming viewpoint they two are very similar and it is up to you which method you want to use. In this exercise, we teach you how to use Local Variables to complete Problem 8.5. So delete all instances of Property Node Values from your program.

To create a local variable, pop up on an existing indicator or control and select: Create >> Local Variable. Drop the newly created icon in the appropriate place in your program. What makes local variables interesting, and at times confusing, is that they can either be in read mode (and act like a control) or in write mode (and act like an indicator), independent of the original object that they represent. So pop up on the newly created local variable and set it to the appropriate type by either selecting “Change to Read” or “Change to Write.” Now wire it up just like the previous property nodes and make sure that your code still works. (Note: the same limitations also apply to Local Variables: they cannot be associated with switch controls in latched modes!)

Hand in the Block Diagram of your modified program.

13.5.3. Do the “Do It Yourself” on page 294. For this exercise, you will again need either local variables or Property Node Value; the choice is yours. You may add as many or as few sequence structure frames as you deem important. What follows are a couple additional hints: (You may want to try programming first, without reading them.)

To start with get the simple sequence of steps 1 and 2 working, i.e., initialize the display and LED, wait a random time interval and then light the LED. For now, don’t worry how many sequence structure frames this takes.

The trickiest part is the implementation of the STOP button function. As you have probably noticed by now, just dropping the STOP button by itself into a sequence frame does not prevent the program from continuing on to its next sequence frame. As a matter of fact, LabView will read the state of the button and make a note of it but it will not wait indefinitely until the button’s state has
been changed. To achieve what we want, we need “something else” that keeps on running repeatedly (within the sequence structure frame) until the STOP button is pressed. Hint: the key word is “repeatedly.”

Finally, at the place where you’re turning on the LED, you may also want to reset the STOP button to prevent cheaters from pushing it ahead of time!

Hand in the Block Diagram of your program and a screen shot of your shortest reaction time recorded.

(If this was not challenging enough try the following optional exercise: display your reaction times in a histogram. Enclose the entire program above within another while-loop with yet another stop button. To create the histogram, we first need to assemble the individual reaction time readings into an array. Place the Build Array function (Programming >> Array >> Build Array) outside the sequence structure (but inside the outer while-loop) and wire its input to the calculated reaction time. Drop the Create Histogram Express Vi (Mathematics >> Probability & Statistics >> Create Histogram) next to it and set the appropriate “Configuration Properties” for it. Wire its Signal Input to the Build Array output. Finally, right click on the Create Histogram Histogram’s output and select Create >> Graph Indicator. This optional assignment is great excuse for the procrastinator in you, especially if have urgent assignments due.)
Chapter 10: Fourier Transform and Spectrum Analyzer


Introductory Concepts

So far you have used one of the most versatile scientific instruments, the oscilloscope, to observe signals in the time domain. This instrument is particularly well suited for analyzing the characteristics of periodic signals. By observing the type of periodic signal present and by measuring its frequency, phase and amplitude we obtain a complete set of information to describe the signal so that it could be reproduced by a function generator.

Nevertheless, even for simple signals, the oscilloscope lacks one crucial piece of information: it is not able to provide us with quantitative information regarding the shape of the signal. As far as that goes, we are left to our own (subjective) judgment to describe how well the signal matches standard wave forms such as a sinusoid, square wave, triangle wave etc. This task can become daunting for an arbitrary signal that does not match any of the commonly used waveforms. Furthermore, recreating such an arbitrary signal form the information gathered from observing it in the time domain will be nearly impossible unless we have an arbitrary signal generator, in essence a D2A converter, capable of creating any shape we program it to do.

The problem of complex, rich signals is found widely in nature and these signals are often very pleasing to us; for example, consider musical sounds. Fortunately, these signals can completely be described as a composite of simple sinusoids each with its own specific frequency, amplitude and phase. While this may at first sound like a daunting task, in real life, it turns out that only a small number of such sinusoids are required to recreate the original signal. Generally, the lowest frequency component, called the fundamental (frequency), is often the one with the largest amplitude. If this is the case then this component dominates the original signal. For example, in music, it is the fundamental frequency that defines (the frequency of) a musical note. The other frequency components that make up the original signal are usually integer multiples of the fundamental and are called harmonics. Hence, they are always of a higher frequency than the fundamental and generally, but not always, are of decreasing amplitude, i.e., significance. While the fundamental defines the musical scale, the harmonics give the tone its unique “flavor” or “accent.” The harmonics allow us, for example, to distinguish an identical note of one instrument from another or between vocalists all humming the same note.

Therefore, when we recreate the original signal by summing its frequency components, we do not have to sum an infinite number of them, as some (mathematical) text books may lead us to believe. Instead, by selecting the fundamental and the two or three most significant harmonics a
fairly accurate rendition of the original signal is achieved. (See for example problem 9.3.2., i.e., problem 3.4 on page 120 of the LabView text book where, in an earlier exercise, you created a square wave by summing sine waves; a square wave is particularly difficult to recreate since its signal is discontinuous.)

The (mathematical) technique that allows us to decompose a signal into the individual frequency components is called the Fourier Transform. Its mathematical description is given in the first sections of chapter 10 of the text book but a more detailed description, including the Fast Fourier Transform (FFT) algorithm which is usually applied to discrete signals, can be found in chapter 12 of the book Numerical Recipes².

The scientific instrument that performs the frequency decomposition and then display signals in the frequency domain is called a Spectrum Analyzer. Paired with an oscilloscope, these two scientific instruments provide us with a wealth of pertinent information and in the following sections you will build just such an instrument.

However, before we start on this task consider the significance of the frequency domain analysis and its application using an example from physiology: if you are able to hear, than you already are acquainted with one extremely sophisticated spectrum analyzer that nature provides, namely the cochlea inside of your ear. Without it, your brain would have to sample, analyze and process the incoming audio signals at roughly 40 kHz which is beyond its capacity. In contrast, having a cochlea simplifies this process greatly. It does so through its snail like shape that acts like a physical spectrum analyzer. When sound enters the cochlea, depending on its frequency components, it stimulates only certain physical regions inside the cochlea and the nerves that are connected to it. In other words, it physically performs a frequency decomposition of the audio signal and only the nerves assigned to the particular area of the cochlea will respond. Note that the speed at which the brain needs to process these signals is now determined by the speed at which audio signals change, typically in the tens of Hz range, yet we’re able to hear signals in the (lower) kHz range.

Hopefully, this introduction has roused your interest as you will now build a fairly sophisticated spectrum analyzer using LabView and, later, the DAQ card. The vast majority of the programming will be done in this section; in the remaining sections you will put your program to work to study some interesting data acquisition and signal processing techniques.

LabView Text Book Sections 10.1 through (and including) 10.8

These sections cover the mathematical aspects of the Fourier transform and they form the prolog to building the spectrum analyzer. Read it and do all the exercises, especially the program FFT (Magnitude Only) on page 353.

Addition 1: Digital Oscilloscope

After you are done with section 10.8, improve the versatility of your FFT (Magnitude Only) program and add the functionality of a digital oscilloscope to it. This way you will be able to observe the input signal in both the time and frequency domain which may come in handy should you ever have to debug your code.

1) The oscilloscope feature is implemented by adding a Waveform Graph to your front panel. Label it Waveform Graph t-domain. (For a partial block diagram, see Figure 14.1. below.)

2) Since the Waveform Graph requires three inputs in the form of \( t_0, \Delta t \) and data array, add a Bundle function to its input. The Bundle function is found under Programming >> Cluster, Class & Variant.

3) Wire the top input of the Bundle, \( t_0 \), to the numerical constant 0.

4) Expand the Digitizing Parameters’ Unbundle By Name function by dragging on it till it reveals the Sampling Frequency output. Wire its inverse to the middle tab of the Bundle function as shown below. The inverse operator can be found under Programming >> Numeric.

5) Finally, wire Wave Sim’s Displacement output to the bottom input of the Bundle.

![Figure 14.1. Partial Block Diagram of the time domain display components.](image)

Test your program and check that it displays the signal now also in the time domain.

**Addition 2: Calculation of the Power and RMS Voltage**

Instead of second guessing the amplitude of the signals in the frequency domain by staring at the corresponding XY Graph, let’s calculate their Root-Mean-Square (RMS) Voltage from the data in your frequency spectrum. Add a numerical display to your front panel and label it \(<V RMS>\).

The (average) \( V_{RMS} \) voltage is related to the total power by:

\[
\langle V_{RMS} \rangle = \sqrt{P} \quad (14.1)
\]

The power itself is related to the signal in the frequency spectrum by:

\[
P = \sum_{k=0}^{N-1} |A_k|^2 \quad (14.2)
\]

where, borrowing the book’s notation, \( A_k \), is defined as the individual Fourier component divided by the number of samples, i.e., (see equation 15 on page 314 of the text.)

\[
A_k \equiv \frac{X_k}{N} \quad (14.3)
\]

Combining these equations yields:

\[
\langle V_{RMS} \rangle = \sqrt{P} = \sqrt{\sum_{k=0}^{N-1} |A_k|^2} \quad (14.4)
\]

Note that this expression includes the \( k = 0 \), or DC component in its summation. Therefore, it represents a value that is a composite of DC offset and signal amplitude. In the extreme cases, where the DC offset is much smaller than the signal’s (AC) amplitude, the RMS voltage represents the “averaged” (rectified) amplitude; for the opposite case, where the offset is much larger than the AC amplitude, it measures essentially the DC offset.

This representation of the RMS voltage is fairly standard. For example, it agrees the way LabView calculates the RMS voltage in its Basic Averaged DC RMS.vi. Nevertheless, there are times (see
the next section) when it is advantageous to suppress the DC component. This can be done by setting it to zero or by summing form \( k = 1 \) to \( N-1 \). Under such circumstances, the RMS voltage will be a representation of the AC part only and it will be independent of offset voltages or DC drift. Keep this mind even though for now, we will use the expression shown above.

This expression can be implemented in your program by a newly created for-loop with a shift register. Since the magnitude of \( A_k \) has been calculated in the Complex to Polar data conversion VI, wire its output directly into the for-loop where it is squared using the Programming >> Numeric >> Square function. The values can then be summed using the method described in section 13.1 of this manual. Feed the output from the shift register into the Square Root function and from there into the numerical display created earlier.

(A final warning before you test this program feature: for whatever reason, the author of the LabView text has chosen to display the frequency domain data in units of amplitude, which is rather unconventional. The only advantage to the book’s notation is that when we observe purely sinusoidal signals, their amplitudes will correspond directly to the value shown on the frequency domain’s XY Graph. More commonly, the frequency components are displayed in \( V_{RMS} \) units, which means that they have been reduced by a factor of \( 1/\sqrt{2} \). By displaying them in RMS units their square can then directly be summed to obtain the power contained in the signal. If on the other hand you were to sum the data in the display, you would be off by a factor of \( \sqrt{2} \)!

Test your program and check that the display agrees with the “Amplitude” value set in the “Waveform Parameters” for a simple Sine wave. (Did you remember that the RMS to amplitude voltage conversion factor of \( \sqrt{2} \)?) Does it agree with other waveforms? Why not?

**LabView Text Book Sections 10.9 through (and including) 10.12**

These sections deal extensively with the problem of “leakage” when performing a discrete Fourier transform. You may think of leakage as a spreading, or smearing of the resulting function transform due to edge effects and (frequency) bin misalignment. However, it does not, as the text may at times lead you to think, cause incorrect results. While this effect is certainly real, and while you should understand its cause, what the significance of the effect is, remains a different question.

Before examining this issue further, let’s approach it with from an entirely different viewpoint invoking Parseval’s theorem. In the discrete case Parseval’s theorem is:

\[
P = \frac{1}{N} \sum_{j=0}^{N-1} |x(t_j)|^2 = \sum_{k=0}^{N-1} |A_k|^2 \quad (14.5.)
\]

It states that the power in a signal must remain the same whether we compute it in the time domain or the frequency domain. You may regard it as another instance of the energy conservation principle.

Using this approach, you will observe that for an input signal with a fixed amplitude, the measured \( V_{RMS} \) must also remain constant, no matter how much leakage we observe. Having implemented the \( V_{RMS} \) display in the previous section, you will be able to monitor this fact while you work through the different parts of sections 10.10 through 10.12.

(Note: the amplitudes on the graphs on pages 364 and 365 of the LabView text are wrong; it shows 1000, they should be 4).
LabView Text Book Sections 10.13

Section 10.13 introduces a remedy for the leakage effect by transforming the input data using a technique called "windowing." The method works by attenuating the input signal at the beginning and end of the acquisition cycle, thereby, reducing the way edge effects affect the Fourier transform.

After you have completed section 10.13, you may have noticed that our $V_{\text{RMS}}$ display from the previous section no longer works correctly. Its value now depends on the type of window applied. This should come as no surprise. Since applying a window alters the power of the input signal in the time domain, the power of the output signal in the frequency domain is equally affected. We will see below how this can easily be fixed by dividing the $V_{\text{RMS}}$ voltage by a window specific conversion factor. However, before we get to this, let’s add a small change to the program that will allow us to graph both the original input data and the windowed input data.

**Addition 3: Display the Windowed Data**

Unless you are familiar with signal processing, the particular names attached to the various windowing functions will have little meaning. It would be far more instructive if we could directly observe their effect on the data in the time domain. To do so, we will add another trace to the time domain graph created in addition 1. For the second trace to work, we again need to bundle the windowed data and then create a 2 dimensional array for the graph display. Detailed instructions are given below.

After having completed the exercises in section 10.13, add the following functions to your block diagram.

1) Disconnect the wire between to Waveform Graph and its Bundle function.
2) Add a Build Array function (from Programming >> Array) near the graph. Drag on the Build Array VI until it has two inputs. Reconnect its output to the input of the Waveform Graph.
3) Reconnect the Bundle’s function output to upper input of the Build Array VI.
4) Add a second Bundle function for the windowed data below the existing one. (Select the existing Bundle and CTRL-drag on it to create a copy.)
5) Similar to the previous instructions in Addition 1, wire the top input of the Bundle, to, to the numerical constant 0.
6) Wire the middle tab of the Bundle function to the inverse of the Sampling Frequency.
7) Wire the windowed time domain data to the bottom input of the Bundle VI and, finally, connect the Bundle’s output to the remaining input of the Build Array function.

If you get stuck check the block diagram for the complete program in Figure 14.2. The components for the two trace Waveform Graph is located at its bottom.

**Addition 4: ENBW Correction for the RMS Voltage Display**

We will now fix the RMS voltage display from the previous section so that its value will remain independent of the window applied. To obtain the power (or rather the RMS voltage) of the original signal we need to multiply it by a correction factor. This correction factor, which depends on the type of window used, is called the effective noise bandwidth of the window, or short ENBW. Therefore, our original equation for calculating the RMS voltage needs to be modified to take this effect into consideration:

$$\langle V_{\text{RMS}} \rangle = \frac{P}{ENBW} = \frac{1}{ENBW} \sum_{k=0}^{N-1} |A_k|^2$$  \hspace{1cm} (14.6.)
For a rectangular window, which is equivalent to no window being applied, $ENBW = 1$. $ENBW$ values for other shapes will be directly supplied by the LabView Scaled Time Domain Window function.

After implementing the windowing function in section 10.13, update your $V_{RMS}$ calculation in your program:

1) Obtain the $ENBW$ value for the particular window applied. In Programming $\rightarrow$ Cluster, Class and Variant select Unbundle by Name.
2) Wire the Window Property output from the Scaled Time Domain Window function to the input of the newly created Unbundle by Name function. You should see a "eq Noise BW" label. This represents the $ENBW$ correction factor for the particular window that has been selected.
3) Remove the wire between the square root function and the output from the for-loop created in the previous section to calculate the signal power.
4) Insert the Divide function and divide the signal power the by $ENBW$ value before passing its output to the square root function.

Again, if you have problems consult the block diagram shown in Figure 14.2.

Run your program a few times, using a different window each time. Observe the shape of the windowed data and check that $<V_{RMS}>$ display remains (within round off errors) constant, independent of the window applied.

**LabView Text Book Sections 10.14**
You should read section 10.14 but do not add any of its features to your program.

In the first case, you have already implemented the “Estimated Amplitude” function in the previous sections.

Second, the “Estimated Frequency” calculations (as presented in the text) are of very limited benefit. They will only work with input data consisting of a single frequency component, such as a pure sine or cosine signal. For all other input signals, for example, a square wave, it will produce utter non-sense. (There is no physical meaning attached to a weighted average of the frequency components.) If you need to measure the numerical values of the individual frequency components instead use LabView’s *Extract Multiple Tone Information.vi* or Harmonic Distortion Analyzer functions; they are found under Signal Processing $\rightarrow$ Waveform Measurements.

**LabView Text Book Sections 10.15**
Read it and work through the exercises.
Assignments

14.1.1. Hand a complete block diagram of your program, including the additions listed above. Also hand in a screen capture of your front panel with the following settings:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Samples</td>
<td>2048</td>
</tr>
<tr>
<td>Sampling Frequency</td>
<td>20000</td>
</tr>
<tr>
<td>Frequency</td>
<td>19500</td>
</tr>
<tr>
<td>Amplitude</td>
<td>1</td>
</tr>
<tr>
<td>Shape</td>
<td>Sine</td>
</tr>
<tr>
<td>Window</td>
<td>Hanning</td>
</tr>
</tbody>
</table>

At what frequency did you observe the input signal in your graph; does it agree with your calculations regarding aliasing?

14.2. Spectrum Analyzer Using the DAQ Assistant

We will now make some minor modification to the program from the previous section and turn it into a “real” spectrum analyzer by replacing the Simulate Signal source with the DAQ Assistant.
1) Make sure you save your program from section 14.1. and create a new version using the “Save As” menu command.

2) Delete the following components: Wave Sim, Waveform Parameters and Unbundle by Name for the Displacement component.

3) Set up the DAQ Assistant and configure it to perform an analog input voltage operation on a particular differential AI channel with Acquisition mode >> N Samples selected. Do not set up external triggering; instead use the DAQ Assistant’s default settings.

4) Connect the Digitizing Parameter’s Number of Samples and Sampling Frequency to the DAQ Assistant’s Number of Samples and Rate input.

5) If during this process some of the wires for the Waveform Graph display appear to be broken, reconnect them as shown below; check that the (former) Displacement data is connected after the Convert from DDT function.

-- Figure 14.3. Block diagram of the spectrum analyzer with the DAQ Assistant. Not shown is the enclosing while loop and delay timer. --
6) Finally enclose the entire block diagram of Figure 14.3 within a while-loop that can be terminated with a stop button control.

7) To prevent locking up the computer with the program, add a 151 msec wait timer (Programming >> Timer >> Wait) inside the while loop.

8) For the rest of the exercises in this section we will use the input settings shown in the table below. To avoid having to enter them each time when you load your program set them to the default values after you have entered them: right click on the control and select Data Operations >> Make Current Value Default.

<table>
<thead>
<tr>
<th>Number of Samples (Samples / second)</th>
<th>1024</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sampling Frequency (Hz)</td>
<td>2000</td>
</tr>
</tbody>
</table>

First verify that your program works. Set the function generator to a 100 Hz, 1 V<sub>RMS</sub> sine wave without any offset voltage. (To set the function generator into V<sub>RMS</sub> mode select: Enter Number >> 1 >> mVrms.) Connect the function generator’s OUTPUT to your ADC’s terminal block’s input channels.

Run your LabView program using a rectangular window. Does the time domain data display the correct amplitude for the input signal? (It should be √2 times larger than the V<sub>RMS</sub> value.) Now read your V<sub>RMS</sub> display. It should agree with the value set by the function generator.

Now switch your window to a Hanning window. Does your V<sub>RMS</sub> display still agree with the value set on the function generator? Continue only if your program passes all these checks; if you encounter problems, see your TA.

Assignments

**14.2.1.** In the first assignment investigate the effect the DC offset has on the program. Specifically, how does the DC offset affect the FFT and V<sub>RMS</sub> calculations? In your program select the Hanning window and set the function generator to produce a 100 Hz, 1 V<sub>RMS</sub> sine wave. Slowly adjust the function generator’s DC offset from 0 to +1 V; what effect does it have on FFT? Capture some screen shots and indicate the DC offset. Now change the DC offset from 0 to -1V; compare this with the previously captured image. How does or doesn’t it differ and why?

As you adjust the DC offset, observe your V<sub>RMS</sub> display. Show that for an input signal, \( V(t) = a \sin(\omega t) + b \), the RMS voltage observed corresponds to: \( V_{\text{RMS}} = \sqrt{\frac{a^2}{2} + b^2} \)

**14.2.2.** Now let’s explore the Fourier Transform of various waveforms. Remove the DC offset, i.e., set it to 0 again. First, set the function generator to produce a 40 Hz, 1 V<sub>RMS</sub> square wave. Observe the effect of windowing on the frequency domain data when a rectangular window and when a Hanning window is applied. Get some screen shots and explain the effect of “leakage.”

**14.2.3.** With the Hanning window applied, compare the harmonics from a 33 Hz square, triangle and sawtooth input signal? For each input signal create a screen capture. What can you conclude from the harmonics for each input signal? Specifically, how quickly do the harmonics in each case diminish? For example, in which case is a large part of the signal contained in the harmonics, i.e.,
for which case is there very little power in the harmonics? By looking at the time domain data, can you explain why this is the case?

14.2.4. Finally explore the effect of aliasing. Use a 1 VRMS sine wave (without any offset) and start increasing its frequency from form 100 Hz towards 3000 Hz. (Window your input data with a Hanning window.) Increment the input signal in steps of a 100 Hz and observe the signal in both the frequency and time domain. Explain what you saw. Pay attention to the signal between 900 and 1100 Hz and 1900 and 2100 Hz.

Take a screen shot for a 100 Hz and 2100 Hz signal. From these two screen shots alone, can you conclude which one corresponds to the 100 Hz signal and which one corresponds to the 2100 Hz?

Take a screen shot at 2000 Hz. Explain.

Finally use a square wave: start at 40 Hz and increase to about 400 Hz. Does it make sense? Take a screen snapshot of the square wave at 40 Hz and at 300 Hz. What is the “observed” fundamental frequency in those two cases?

14.3. Data Acquisition & Signal Processing III: Data Averaging In the Frequency Domain Using the DAQ

When you average data, presumably in order to reduce noise, you have two options: You may either average the data in the time domain as we have done in the previous chapter, or in the frequency domain, as we will do in this section. It may seem that the two methods should yield the same result and constraints but as you will discover that is not necessarily true, especially if we average the amplitude data.

Save your program from the previous section and change it so it will allow averaging. This can be achieved with the method used in section 13.2. You will need to:

- Add shift registers to the while-loop enclosing most of your code.

- Initialize the left shift register using the Initialize Array function from the Programming >> Array functions. Place it out outside of the while-loop, and connect its output to the left shift register. Initialize all its elements by setting the element input to a constant value of 0. To specify the number of array elements create a local variable of the Digitizing Parameters: right click on Digitizing Parameters and select Create >> Local Variable; drag and drop it outside of the while-loop. Right click on the newly created local variable and select Change to Read. Finally, connect its output to the Unbundle by Name function; wire its Number of Samples to Initialize Array dimension size 0 input. Connect the Initialize Array function to the left shift register. (Note: if you want to be picky, we actually only need half of the Number of Samples; for simplicity we ignore this fact for now as it will have no bearing on the program performance.)

- Now accumulate the amplitude values: inside the while loop, add the Programming >> Numeric >> Add function. Connect one of its inputs to the left shift register; connect the other to the Amplitude values right before they are bundled for the f-domain graph. (See the partial block diagram below.) Connect its output to the right shift register.

- To obtain the average, you need to divide the accumulated data by the loop index variable, $i+1$. 

Finally, add another trace to the f-domain graph to display the averaged data: To do so, you will need to disconnect the existing wire between the **bundle** and the frequency domain **XY Graph**; add the **Build Array** function; drag on it till it has two inputs; connect its output to the **XY Graph**. Add another **Bundle** function and bundle the averaged data with the positive frequencies; wire its output to the remaining input of the **Build Array** function. If you get stuck, check the partial block diagram below.

**Figure 14.4.** Partial Block diagram of the spectrum analyzer with the DAQ Assistant and the averaging function.

Note: in this exercise, we do not calculate the $V_{\text{RMS}}$ value for the averaged data. It would be nice to add such a feature but for the sake of brevity we will omit it. Nevertheless, you should always be aware that your $V_{\text{RMS}}$ display indicates non-averaged values!

Input a 93 Hz, 1 $V_{\text{RMS}}$ sine wave and check that both the averaged and the original data overlap on the display.

**Small Signal Generation & Measurement without Phase Locking**

Similar to section 13.3, we will again (attempt) to measure small signals by using an averaging technique. As a quick refresher, we learned in section 13.3 that signal averaging in the time domain only makes sense if we use a phase locked signal. However, in experimental settings it is fairly common that we do not have access to a phase locked signal. It is the purpose of this exercise to show that even in the absence of such vital information, we are still able to detect it.

There are two differences between this exercise and the one in section 13.3: First, we average the amplitude data in the frequency domain, as opposed to the time domain. Second, we will not use phase locking. In other words, in the DAQ Assistant’s properties, no trigger is specified and no (trigger) cable is connected from the function generator’s SYNC out to the DAQ board.

Start by reducing the function generator’s 1 $V_{\text{RMS}}$ analog 93 Hz sine wave signal by a factor of 1000. Again build a voltage divider composed of a 1k $\Omega$ and a 1 M$\Omega$ resistor and feed the voltage divider’s output into the A2D’s input at the terminal block.

Run your program using a Hanning window and measure from the frequency domain graph the averaged amplitude of your signal. Use ALT-PrtScrn to capture the image and paste it into your lab document. Explain what amplitude you expected to read and what the percentage error was in the reading. Calculate how many times the input signal was larger than the DAQ’s resolution.

Repeat the previous process two more times: reduce the function generator’s output voltage to 0.2 $V_{\text{RMS}}$ and to 0.1 $V_{\text{RMS}}$.

(Hints: don’t forget that the frequency domain graph’s unit are in amplitude; in other words you need to divide the measured values by $\sqrt{2}$ to convert to $V_{\text{RMS}}$. When calculating how many times larger the input signal was than the DAQ’s resolution, do not forget to convert to $V_{\text{pp}}$.)

**Fourier Transform / 14.3. Data Acquisition & Signal Processing III: Data Averaging In the Frequency Domain Using the DAQ**
As you learned in section 13.3., when the signal detected is smaller than the resolution of ADC, the accuracy of the measured amplitude becomes suspect. Since we have already observed this effect in section 13.3, we will not proceed further and we will not try to detect even smaller signals than the ones indicated above. (Though you are certainly free to give it a try, if you choose so.)

**Assignments**

14.3.1. Since you will no longer have to modify your program, please hand in a copy of of your front panel and block diagram.

14.3.2. Hand in the screen captures obtained from the 1.0, 0.2 and 0.1 V\textsubscript{RMS} 93 Hz sine wave signals into the 1:1000 voltage divider. State the measured amplitude, its expected V\textsubscript{RMS} before the 1:1000 voltage divider and how many times larger than the resolution of the DAC the signal into the DAC was.

14.3.3. If we compare this averaging technique to the one from section 13.3 we see that they are roughly equivalent in terms of accuracy. Nevertheless, when using the technique explored in the current exercise, we are no longer required to phase lock our signal. **Important:** can you explain why this is the case? Specifically, when you look at the non-averaged frequency domain signal, how does the signal amplitude compare to the average noise amplitude nearby? Also, what phase information does the amplitude contain?

### 14.4. Frequency Response of an RC Filter Using White Noise

A typical physics experiment studies how a device responds to a change or stimulus. More specifically, a large group of these experiments concern themselves with how the device reacts when its input is changed as a function of frequency. For example, this approach is used in spectroscopic measurements to study absorption, transmission and resonances of physical systems as a function of energy. In electronics, we obtain important operating parameters if we understand how a device behaves as a function of frequency and we may, accordingly, classify the device as a type of filter, resonator or neither.

In these experiments, there are two different approaches in determining the response of a device to a changing input frequency.

a) **Input Frequency Sweep:** a signal with a single frequency component, such as a sine wave, is applied to a device and the device’s response is monitored while the input frequency is swept over the region of interest.

b) **White Noise:** white noise is defined as a signal that contains every frequency component (in the region of interest) and all its frequency components are of the same (average) amplitude. Applying white noise to a device means that we apply all frequency components to the device, more or less, at the same instant. If we then monitor its response and Fourier transform it, we then can detect how the device responds to the frequencies applied.

The first method, a), requires that we have access to a signal source whose output frequency can be controlled. A programmable function generator usually works well. Note that this approach does not require a spectrum analyzer to obtain the frequency data; the method itself works in essence like a Fourier decomposition. This semester, you will not learn how to program the function generator using LabView; therefore, we will not use this method, at least for now.
The second method requires a spectrum analyzer and also relies on a white noise source. Two different types of sources are available. In the first category you will find commercial white noise generators. For example, the functions generators in the lab are capable of producing white noise. Even if you would not have access to such a generator, you could create your own generator using a Thermal or Johnson noise source. It produces clean white noise and the white noise spectrum can easily be obtained by amplifying a signal across, for example, a simple carbon resistor. You will be studying such a white noise source next semester.

An entirely different source of white noise generators relies on the impulse function. Just as the Fourier transform of a DC signal results in a delta function in the frequency domain, by symmetry, the Fourier transform of a white noise signal is an impulse function in the time domain. In other words, by applying an impulse that is on the order of, or slightly less than, one sampling interval we have created a white noise source. This method is analogous to hitting a bell with a hammer: during the brief impulse all frequency components are present, but only the ones producing a resonance will ultimately survive and can be heard in the ringing bell.

While you will use the impulse method in the next section to study the characteristics of an unknown device, in this section you will be studying the frequency response of a simple RC filter using the built in white noise source of the Agilent function generator. To better observe the signal, you will be using signal averaging in the frequency domain. The code for this (and the next) exercise has already been written by you in the previous section.

White Noise Source

In this exercise, you will observe the frequency response of an RC filter using the function generator in the lab. The Agilent 33120A has a built in white noise generator that can be activated by pressing the Noise button on its front panel. Activate it with a 500 mVRMS amplitude and connect the function generator’s OUTPUT directly to DAC terminal block’s input.

Before you run your program from the previous section you may want to adjust it to the following input values. So that you will not have to enter them each time you start your program right click on the controls and select Data Operations >> Make Current Value Default after you have entered the values listed below.

<table>
<thead>
<tr>
<th>Number of Samples (Samples / second)</th>
<th>2048</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sampling Frequency (Hz)</td>
<td>200'000</td>
</tr>
<tr>
<td>Window</td>
<td>Hanning</td>
</tr>
</tbody>
</table>

Run your program and check that the input indeed produces a white noise source response as shown in figure 14.5. By comparing the averaged vs. the non-averaged data in the frequency domain graph, you should understand now why averaging is crucial in observing the white noise spectrum. (You may want to turn auto scaling the y-axis off by right clicking on it and setting the y-axis...
axis to an appropriate maximum value; try 10 mV.) After about 20 averaging cycle, capture the screen and paste it into your document.

**White Noise Source with RC Low-Pass Filter**

On a bread board wire an RC low-pass filter using $C = 0.1 \ \mu F$ and $R = 50 \ \Omega$. Connect the function generator's output to the filter's input; feed the output from the filter into the ADC. Run your program again and let it average the frequency domain data till you see a clear trend in the data (about 30 seconds); capture the screen.

From your screen capture measure the $f_{3dB}$ point; calculate the $f_{3dB}$ point from the component values. (Note: if you are using the method from chapter 2 of this manual and calculated the $f_{3dB}$ point from $f_{3dB} = 1/2\pi RC$ then you will be off by a factor of 2. For the correct answer, you must take the 50 $\Omega$ (series) output resistance of the function generator into consideration!)

**White Noise Source with RC High-Pass Filter**

Switch the low-pass filter components so that they form an RC high-pass filter. Run the averaging and notice the very poor rendition of the high-pass filter response. The response can be improved by appending the output of the high-pass filter with another low pass filter with $C_2 = 0.01 \ \mu F$ and $R_2 = 150 \ \Omega$. (See Figure 14.6.) Try it and observe the output with the newly added low-pass filter in the frequency domain. Again, capture your screen.

![Figure 14.6. RC high-pass filter with low-pass stage.](image)

Leave the circuit set-up for testing the next assignment.

**Assignments**

14.4.1. Show a screen capture of your white noise signal.

14.4.2. Show a screen capture of your RC low-pass filter after it has been exposed to white noise. Indicate the $f_{3dB}$ point on the plot. Compare it with the values you calculated from the component values, including the 50 $\Omega$ (series) output resistance of the function generator.

14.4.3. Show a screen capture of your RC high-pass filter after it has been exposed to white noise with and without the added 2nd stage low-pass filter. Indicate the $f_{3dB}$ point on the plot. Compare it with the values you measured in 14.4.2. for the low-pass filter. Why was the RC low-pass filter appended to the high-pass filter? Specifically, what effect is it supposed to suppress? (Hint: see assignment 14.2.3.)
14.5. Frequency Response of an Unknown Source Using the Impulse Function

Introduction

You will again use a white noise technique to characterize a device in the frequency domain. The “mystery” device has already been placed in the boxes labeled “Fourier Exercise: Black Box” and they can be found in the lab.

Though you will again rely on the function generator to produce the white noise, this time we will not make use of its built-in white noise generator. Instead, we will set it up to produce very brief pulses in the time domain. Such a pulse, or delta function, corresponds to white noise in the frequency domain. (See figure 14.5.)

To characterize the device you will apply the pulse to the black box supplied. You will need to take two measurements: a calibration measurement and the actual device response measurement.

Creating and Observing the Impulse Function

You will use the Agilent function generator in the “burst mode” to produce a single pulse for each data acquisition cycle. As the duration of the pulse approaches $1/sampling$ rate, it will resemble a delta function in the time domain. It can be shown that the Fourier transform of a delta function corresponds to a constant. In other words, a short duration pulse acts like a white noise source.

It is instructional to consider when a pulse of finite width, such as the one you will be applying to your mystery device, can be considered a delta function.

![Amplitude vs. Time and Frequency](image)

- Figure 14.7. Left Image: pulse of finite width $t_0 = 1$ in the time domain; Right Image: same pulse in the frequency domain.

The Fourier transform for a single square pulse with unit amplitude and width $t_0$, such as the one shown on the left in figure 14.7, is:

$$F[\omega] = \frac{1}{\pi} \frac{\sin\left(\frac{\omega t_0}{2}\right)}{\frac{\omega}{2}} \quad (14.7.)$$

The absolute value of this function is plotted in Figure 14.7. on the right and it corresponds to what you would observe with your FFT program in the XY Graph frequency domain. As you can see, distinct “lobes” with a width of $\Delta f = 1/t_0$ form. Therefore, as $t_0$ is decreased to $2/sampling$ rate, the
center lobe’s width spans of the entire Nyquist frequency range. Decreasing $t_p$ further to $1/sampling\ rate$ expands the center lobe to such as an extent that, for all practical purposes, it appears to be of a constant value. Before you will be able to observe this behavior yourself, you need to make a couple minor modifications to your program.

1) Since we only want to capture only one pulse per data acquisition cycle, we need to synchronize the DAQ card with the function generator. Right click on the DAQ Assistant and under the Properties set the trigger values as shown below. Be sure to set the trigger edge to “Falling.”

![Figure 14.8. Trigger setting for the DAQ Assistant.](image)

2) If you are auto-scaling your y-axis in the frequency domain, you may find the influence of the DC offset annoying. To turn it off, set its multiplication factor to 0 in the loop where the amplitudes are calculated. (See figure on the right.) This is not crucial but it will simplify things.

3) To observe the input pulse in the time domain, turn auto scaling for the x-axis off. Instead, set it to display the input data for the first 1 msec.

4) The impulse appears at the very start (or at the edge) of the data acquisition cycle. The purpose of the windowing function is to diminish edge effects. Therefore, leaving it on would essentially diminish our input signal to the point where it no longer can be detected. Therefore, it is essential that the windowing feature is turned off, i.e., use the Rectangle window. For the other program parameters see below:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Samples</td>
<td>2048</td>
</tr>
<tr>
<td>(Samples / second)</td>
<td></td>
</tr>
<tr>
<td>Sampling Frequency (Hz)</td>
<td>200'000</td>
</tr>
<tr>
<td>Window</td>
<td>Rectangle</td>
</tr>
</tbody>
</table>

The only thing left before we can observe the signal is to set up the HP 33120A function generator in its burst modulation mode. On the function generator select: Shift >> Burst. In this mode, the generator no longer produces a continuous train of waves but instead produces a number of pulses which are repeated at a specified “burst rate.” The number, shape, frequency and amplitude of these pulses can be set with the standard controls on the function generator by pressing Shift >> MENU buttons. Doing so will display: A: MOD MENU. Select the down arrow
Adjust the function generator’s front panel to produce a 0.3 V<sub>pp</sub> square wave. Start out with a “frequency” of 5 kHz. Note that frequency in this case pertains to 2/t<sub>o</sub>, where t<sub>o</sub> is the pulse width. (In case you wondered, the factor of 2 comes from the fact that we only use half of the square wave cycle; for the other half of the cycle and for the rest of the entire data acquisition cycle, the signal remains “low.”)

Finally, connect the function generator's (analog) output, OUT, to your ADC board's analog input. Connect a cable from the SYNC output of the function generator to your DAC’s terminal block PFI 0 input; it provides the external trigger of the ADC.

Now run your program. (Did you turn the window selection to Rectangle?) Observe the signal in the time and frequency domain. Does it look similar to Figure 14.7? If so, capture the screen and then continue and increase the function generator’s frequency from 5 kHz to 10 kHz, 25 kHz, 50 kHz and finally 100 kHz. (Note: either restart the program after each frequency change to clear the averaging effect in the frequency domain graph or disable the averaged trace for now.) Take a screen snap shot at each frequency, especially the 100 kHz. It will serve as your calibration measurement.

To test that everything works, you may want to connect one of the RC filters from the previous section between the function generator’s output and the DAC’s input. With the function generator’s frequency set at 100 kHz, does the frequency domain signal look identical to the one obtained in the previous section where we used the function generator’s white noise generator?

**Calibration Measurement**

If you haven’t already done so take the calibration reading. Remove any components between the output of the function generator and the DAC’s input and connect the function generators OUTPUT directly to your ADC’s input channel. (Leave the function generator’s SYNC OUT connected to your PFI 0 input.) Once more, confirm that the function generator’s output voltage has been set to a 0.3 V<sub>pp</sub> square wave and that its frequency is set to 100 kHz.

Run your program with the settings specified earlier in this section, i.e., use only the rectangular window.

You should again observe the constant white noise signal in the frequency domain. Measure its amplitude in the frequency domain. You may want a print out of this dataset for your long report. (If you have not done so in the past, hit Alt + Print Screen and paste the screen shot of your GUI with the white noise into a word document.)

**Device Characterization Measurement**

The black box must be powered with an external +5V DC signal. Connect the banana jacks on the black box to an external power supply. (Alternatively, if you are using the CB-68LP terminal block, you may instead use the +5V from pin 34, and the digital ground, pin 33, that is supplied by the DAC to power the black box.) Now connect the function generator’s output to the input of the black box labeled “IN;” connect the output of the black box, “OUT,” to your ADC’s input.

Finally, measure the device characteristics using exactly the same parameters as for the calibration measurement. Again take a screen shot of your GUI or make a print-out for your long report. (The results may look visually more appealing if you disable auto scaling for the x-axis for
both graphs. Manually set the range in the time domain graph to 1 msec and in the frequency domain graph to $10^4$ Hz.)

Specifications for filters are conventionally displayed using a Bode plot. In such a plot, both the x-axis (displaying the frequency) and the y-axis (displaying the filter's response) are plotted on a log scale. You can easily obtain such a plot by stopping your program and then right clicking on each axis of the frequency domain plot and selecting Mapping >> Logarithmic. Again capture the screen with the Bode plot.

You should now be able to answer the following questions:

a) If you were to describe the black box in terms of a filter what type is it: a low-pass, high-pass, band-pass or band-stop?

b) Calculate the bandwidth of the filter: measure the frequencies at which the output changes by 50% from its minimum to its maximum values, or from its maximum to its minimum values. The bandwidth is defined as the difference between these two frequencies. (Note: in “real” life you would measure the $f_{3\text{dB}}$ points but for the sake of brevity we will use the simplified approach of the 50% change.)

c) Measure the gain (or attenuation) of the device around 2000 Hz. This can be calculated by comparing the calibration data set with the one obtained with the mystery device connected.

d) Measure the filter's attenuation in decibels/octave (dB/octave) using your Bode plot. On your Bode plot you should be able to discern 3 distinct regions: first, the center region, next, the attenuation region on either side of it, and finally the noise pedestal (on either side) where the filter's action is buried in the background noise. On your Bode plot screen print-out draw two straight lines indicating the filter's attenuation. (The absolute value of the slope of the two lines should be roughly identical.) From the slope, measure the filter's attenuation: you may recall that one octave is defined as a doubling of frequency; therefore, along the line you drew, measure two amplitudes, $A_1$ and $A_2$ at $f_1$ and $f_2$, where $f_2 = 2 f_1$; finally convert the ratio of $A_1$ and $A_2$ to decibels: $\text{dB} = 20 \log(A_1/A_2)$. How does it compare with the typical RC filter's response of 6 dB/octave?

Note: the answers to the questions above should provide the “central theme” of your long report for chapter 14.

Assignments

14.5.1. Show screen captures with the function generator's frequency set to 5 kHz, 10 kHz, 25 kHz, 50 kHz and 100 kHz. Briefly describe why the data looks that way. Calculate the pulse width, $t_o$, for the 5 kHz square wave and the width of the corresponding nodes in the frequency domain, $\Delta f$. Do they agree with your plot?

14.5.2. Make sure you that you have a good screen shot of the calibration measurement.

14.5.3. Answer all the questions in the “Device Characterization Measurement” section.
When you design or analyze all but the most trivial circuits it is useful to break them up into a number of simpler circuits that can be more easily understood. Because the voltage divider (or the current divider) is very basic, we would like to treat any complex circuit like an assembly of voltage dividers. Therefore, if we know the "equivalent" resistance of parts of a circuit then we can predict how each part affects the other. Furthermore, we can also predict how the entire circuit would affect a source or a load attached to it.

When we simplify a circuit and quote its resistance, or as in the AC case, its impedance, we need to know exactly where in a given circuit this value refers to. Though the impedance could be measured anywhere, usually we are only interested in the impedance at the input and output of a circuit.

- Figure A.1. Not all circuits have an input and an output. For example, a battery has only an output terminal.

Sometimes it is obvious which part of the circuit is the input terminal and which one the output. If this is not the case it is important that the points are clearly identified; generally the circuits are drawn with the left side as the input and the right side the output.

- Figure A.2.

To predict how a circuit affects a source, we replace the circuit with a resistor (and maybe a capacitor) that corresponds to the measured or calculated input impedance of a circuit. The same
Input and Output Impedance Measurements and Calculations

A.1. INPUT IMPEDANCE

**GENERAL**

holds true for how a load affects a circuit except that in this case we replace the actual circuit with a resistor equal to the value of the output impedance.

While the input and output impedance represent a real physical characteristic of a circuit they usually do not describe every electronic property of a given circuit. Therefore, we cannot expect that replacing an entire circuit with its equivalent input or output impedance will reproduce every physical property of a circuit. It should be understood that replacing the circuit with its equivalent input and output impedance is merely a very powerful tool for circuit analysis. This concept is used in modeling "real" devices as "ideal" ones and for maximum power transfer applications (i.e. impedance matching).

Because this method is very important you will be asked to calculate or to measure the input and output impedances many times during this course. In order to be able to calculate the impedances it is assumed that you have enough information about the circuit and that you know how to calculate it; this is often nontrivial especially if active elements such as transistors and op-amps are part of the circuit. Measuring the impedances can be tricky if the impedances are very large or very small.

What follows are various recipes and conventions for finding the input and output impedances and we will use these during the entire course. Nevertheless, you should understand that while these conventions are the most common ones they are not the only ones that people use. Therefore, when you are quoting your results you should always indicate how you arrived at them. You should always specify which impedance you measured (input or output); in addition, if there are any other terminals in the circuit indicate if they were kept open- or short-circuited.

**A.1. INPUT IMPEDANCE GENERAL**

The input impedance of a circuit is the impedance seen by a signal source driving a circuit. It is generally an indication of how much a circuit affects a signal source. Note: if the circuit has an output terminal, then all our calculations and measurements are done with NO load attached, i.e. the input impedance is always measured with the output open-circuited.

**A.2. INPUT IMPEDANCE: Calculation**

- Figure A.3.

1) Identify the input and output terminals of the circuit.
2) If the circuit has an output terminal, there should be no load attached to it.
3) Short-circuit all independent voltage sources and open-circuit all independent current sources in the circuit.
4) Calculate $R_{\text{Thevenin}} = Z_{\text{IN}}$ looking into the input terminal; this is the input impedance of the circuit.

**A.3. INPUT IMPEDANCE: Measurement**

To determine the input impedance, $Z_{\text{IN}}$, treat the circuit in figure A.4. as a voltage divider. The current in the circuit is:

$$ I = \frac{V_R}{R} = \frac{V_{\text{in}}}{Z_{\text{IN}}} $$

(A.1.)

Solving for $Z_{\text{IN}}$, we find:

$$ Z_{\text{IN}} = \frac{V_{\text{in}} R}{I} $$
To measure the input impedance, $Z_{\text{in}}$:
1) Remove any load attached to output of the circuit.
2) Insert a series resistor $R$ between a signal source and the circuit under test and measure the voltage drop across the source, $V_s$.
3) Measure the voltage drop across the circuit under test, $V_C$.
4) Calculate $V_R$: $V_R = V_s - V_C$

Finally use equation A.2 to calculate the input impedance.

\[ Z_{\text{in}} = R \frac{V_C}{V_R} \quad \text{(A.2.)} \]

You may wonder why the extra steps 3 and 4 were introduced to calculate $V_R$ instead of directly measuring it. The reason is subtle: some measurement instruments require that a voltage is always measured with respect to the same point in the circuit. Using the method above measures both voltages with respect to point $A$ often referred to as “ground.”

### A.4. OUTPUT IMPEDANCE GENERAL

The output impedance of a circuit is the impedance seen by a load attached to the output of the circuit as it looks back into the circuit. If the circuit has an input terminal then the output impedance is calculated and measured with an ideal voltage source attached to the input terminal. Since an ideal voltage source has zero output impedance (i.e. its impedance is that of a short circuit) we can say that we measure the output impedance with the input short-circuited.

### A.5. OUTPUT IMPEDANCE: Calculation

1) Identify the input and output terminal of the circuit.
2) If the circuit has an input terminal attach an ideal voltage source to it.
3) Short-circuit all independent voltage sources and open-circuit all independent current sources in the circuit.
4) Calculate $R_{\text{Thévenin}}$ looking into the output terminal; this is the output impedance of the circuit.

(Remember that for finding $R_{\text{Thévenin}}$ ideal voltage sources are replaced with short circuits.)
A.6. OUTPUT IMPEDANCE: Measurement

Notice the circuit in figure A.6. is just a voltage divider with:

\[ V_{\text{Load}} = V_{\text{Open}} \frac{R_{\text{Load}}}{R_{\text{Load}} + Z_{\text{Out}}} \]  \hspace{1cm} (A.3.)

where \( V_{\text{Open}} \) is the voltage measured at the output with no load attached.

Solving A.3. for \( Z_{\text{Out}} \) results in:

\[ Z_{\text{Out}} = R_{\text{Load}} \frac{V_{\text{Open}} - V_{\text{Load}}}{V_{\text{Load}}} \]  \hspace{1cm} (A.4.)

To measure the output impedance:
1) Attach a signal source to the input of the circuit.
2) Measure the open-circuit output voltage \( V_{\text{Open}} \) without any load attached.
3) Attach a "reasonable" load resistor \( R_{\text{Load}} \) across the output; measure the output voltage with the load attached, i.e. measure \( V_{\text{Load}} \). Now use equation A.4. and calculate \( Z_{\text{Out}} \).

Here are a couple of exercises to test your skills. Calculate the input and output impedances (actually, in this case resistance would be appropriate) for the following circuits:

1. Figure A.7. Circuit 1
2. Figure A.8. Circuit 2
3. Figure A.9. Circuit 3
4. Figure A.10. Circuit 4
A.7. Answers:

Circuit 1: \( Z_{in} = 10 \, k \), \( Z_{out} = 0 \)
Circuit 2: \( Z_{in} = R1 \), \( Z_{out} = R2 \)
Circuit 3: \( Z_{in} = 2R \), \( Z_{out} = R/2 \)
Circuit 4: \( Z_{in} = R \), \( Z_{out} = R \)
The currents and voltages in simple circuits are most easily determined by repeated application of the voltage-divider principle. Nevertheless, for more complex circuits, two basic methods exist for the mathematical analysis. They are nodal and mesh or loop analysis and they are applicable in most circumstances. Sometimes, the mathematics involved in solving for the individual voltages and currents can become tedious; in such cases, applying the superposition theorem and Thévenin or Norton theorem may simplify the analysis.

### B.1. Nodal Analysis:

A node is defined as a point where three or more circuit elements are joined together. In nodal analysis the voltage at each node is calculated by summing all the currents, \( i_k \), flowing into and out of each node. From charge conservation, it follows that for each node: \( \sum i_k = 0 \). This simply means that the current flowing into a node must equal the current flowing out of that node.

Here are the individual steps for performing a nodal analysis:

1) Identify and label each node in a circuit as \( v_1, v_2 \) etc.

2) Label the current that flows between each node, for example, \( i_1, i_2 \), etc.

3) Choose an arbitrary direction for the currents and indicate that in your circuit.

4) Choose a node and write down all currents that flow into and out of that node. Remember that the sum of currents flowing into the node must equal the sum of currents flowing out!

5) Apply Ohm's law and replace each current in step 4 with the corresponding \( \Delta V/R \); keep your sign convention consistent with your current direction!

6) Repeat step 4 and 5 for all other nodes and currents.

7) Solve for the individual \( v_i \).

Here is an example:

![Circuit Diagram](image)

*Figure B.1. Circuit to be analyzed*
First, we identify and label all nodes and indicate the currents in the circuit. The direction of the currents are chosen arbitrarily. Nevertheless, once a current direction has been chosen for a particular node, the direction must be maintained for the entire loop! For example, by selecting \( i_1 \) to flow out of \( v_1 \) we imply that it must flow into \( v_2 \).

![Figure B.2. Circuit with nodes and currents.](image)

The circuit in figure B.2 has only two nodes. Picking node \( v_1 \) and applying step 4, we get:

\[
-i_1 + i_2 + i_3 = 0 \quad \text{(B.1.)}
\]

Note, that we used a positive sign for currents flowing into a node and a negative sign for currents flowing out of a node.

Working through step 5, applying Ohm's law, we end up with:

\[
\begin{align*}
  i_1 &= \frac{v_1 - v_2 - V_b}{R_2} \quad \text{(B.2.)} \\
  i_2 &= \frac{v_2 + V_a - v_1}{R_3} \quad \text{(B.3.)} \\
  i_3 &= \frac{v_2 - v_1}{R_1} \quad \text{(B.4.)}
\end{align*}
\]

Again observe the sign convention: for a current flowing from node \( A \) to node \( B \), \( \Delta V \) is: \( V_A - V_B \). Furthermore, current flowing through a voltage source from the negative to the positive terminal, causes a positive voltage drop; current flowing through a voltage source in the opposite direction is considered a negative voltage drop.

Finally, we substitute equations B.2., B.3 and B.4. into B.1. At this point we could solve for either \( v_1 \) or \( v_2 \) though they are dependent on each other. If we arbitrarily ground \( v_2 \), i.e. \( v_2 = 0 \) then we find that:

\[
\text{for } v_2 = 0, \quad v_1 = \frac{V_a R_1 R_2 + V_b R_1 R_3}{R_1 R_2 + R_1 R_3 + R_2 R_3} \quad \text{(B.5.)}
\]

**B.2. Mesh or loop Analysis:**

A mesh is defined as a loop of a circuit that does not contain any other loops within it. In mesh analysis Kirchhoff's law is applied which states that the voltage changes in a complete circuit loop must add up to zero, i.e. \( \sum v_i = 0 \).

The individual steps for performing a mesh analysis are:

1) Find each mesh and assign a "mesh" current to it.

2) Calculate the voltage drop or increase at each point in the mesh and recall that \( \sum v_i = 0 \).

3) Apply Ohm's law to every resistive term of step 2; if a component is shared with another mesh, then the current through that component is the difference between the two mesh currents.

4) Repeat step 2 and 3 for all other meshes.

5) Solve for the individual \( i_k \).
As an example we use the same circuit as in the nodal analysis, figure B.1. First, we indicate the mesh currents \(i_1\) and \(i_2\) as flowing in a clockwise direction.

![Circuit Diagram]

- Figure B.3. Notice how the mesh currents are drawn in clockwise fashion to increase symmetry.

The voltage drops going around mesh 1 and 2 are:
\[
V_a - V_{R_3} - V_{R_2} = 0 \\
-V_{R_3} - V_b - V_{R_1} = 0
\]

Again note the conventions. If the mesh current goes through a voltage source from the negative to the positive terminal, then we indicate it as a positive voltage source. If the current flows the other direction, we label it as a negative source. All voltage drops across resistors are negative.

Next we apply Ohm's law to equations B.6.:
\[
V_a - i_1R_3 - (i_1 - i_2)R_1 = 0 \\
-i_2R_2 - V_b - (i_2i_1)R_1 = 0
\]

Finally, we are able to solve for \(i_1\) and \(i_2\):
\[
i_1 = \frac{R_1V_a + R_3V_a - R_1V_b}{R_1R_2 + R_1R_3 + R_2R_3}, \quad i_2 = \frac{R_1V_a - R_1V_b - R_3V_b}{R_1R_2 + R_1R_3 + R_2R_3}
\]

Assuming that \(v_2 = 0\) and solving for \(v_1 = (i_1 - i_2)R_1\) results again in equation B.5.

Situations do at times arise when it is difficult to apply the nodal or the mesh analysis. For example, the circuit below, which is very similar to the one above, can not be analyzed using the nodal analysis method. (At least not to the extent that it is covered here.)

![Circuit Diagram]

- Figure B.4. Circuit to be analyzed

If a situation arises where the nodal analysis does not work, then generally a mesh analysis will work and vice versa.

### B.3. Superposition Principle

The superposition principle states: In any linear resistive circuit containing several sources, the voltage across any resistor or source may be calculated by adding algebraically all the individual voltages caused by each independent source acting alone, with all other independent voltage sources replaced by short circuits and all other independent current sources replaced by open circuits.
Let's apply the superposition principle and determine $v_1$ in the circuit in figure B.1. (For simplicity, we assume that $v_2 = 0$.) Since there are two independent voltage sources, $V_a$ and $V_b$, we need to apply the superposition principle twice. First, we find the voltage at $v_1$ due to $V_a$. This is done by shorting $V_b$.

![Figure B.5: First application of the superposition principle to the circuit in figure B.1.](image)

We can see that $R_1$ in parallel with $R_2$ forms a voltage divider with $R_3$. This allows us to write immediately:

$$v_1|_{V_b=0} = V_a \left( \frac{R_1 R_2}{R_3 + R_1 R_2} \right) = V_a \left( \frac{R_1 R_2}{R_3 (R_1 + R_2) + R_1 R_2} \right)$$  \hspace{1cm} (B.9.)

Next, we find the voltage at $v_1$ due to $V_b$; therefore, we shorten $V_a$.

![Figure B.6: Second application of the superposition principle to the circuit in figure B.1.](image)

Again applying the equation for a voltage divider allows us to write down immediately:

$$v_1|_{V_a=0} = V_b \left( \frac{R_1 R_3}{R_2 + R_1 R_3} \right) = V_b \left( \frac{R_1 R_3}{R_2 (R_1 + R_3) + R_1 R_3} \right)$$  \hspace{1cm} (B.10.)

Finally, from the superposition principle, $v_1$ is:

$$v_1 = v_1|_{V_a=0} + v_1|_{V_b=0}$$  \hspace{1cm} (B.11.)

$$v_1 = \frac{V_a R_1 R_2}{R_3 (R_1 + R_2) + R_1 R_2} + \frac{V_b R_1 R_3}{R_2 (R_1 + R_3) + R_1 R_3}$$  \hspace{1cm} (B.12.)

Simplifying this yields again equation B.5.

Below is a circuit example that was used in a final exam in this course. Applying the superposition principle you should see without any further ado that $V_{out}$ is -1 V.

![Figure B.7: Sample circuit.](image)
### B.4. Thévenin Circuit Theorem

Thévenin's theorem is used to represent a circuit by a voltage source $V_{th}$ and a series resistor $R_{th}$ (or impedance $Z_{th}$). As in the previous appendix on input and output impedance, the Thévenin equivalent circuit can either be determined by measurement or by calculation. The method selected depends on the particular situation though both methods yield the same results. As you will see, determining the input impedance of a circuit is identical to finding its Thévenin resistance.

To calculate the Thévenin equivalent of a circuit:

1) **Identify and label two points (A and B) in the circuit across which you will determine the Thévenin equivalent circuit.**

2) **To find $V_{th}$, calculate the voltage across these two points.**

3) **To determine $R_{th}$, first reduce all independent sources in the network to zero by short-circuiting all voltage sources and by open-circuiting all current sources; now calculate the resistance seen across points A and B; this is $R_{th}$.**

To determine the Thévenin equivalent circuit by measurement, first measure the voltage that appears across the two points A and B; this $V_{th}$, $R_{th}$ is identical to $Z_{th}$ and can be determined by the method explained in the previous appendix.

For example, the Thévenin equivalent of the circuit in figure B.7. is:

![Sample circuit](image)

### B.5. Norton Circuit Theorem

Norton's theorem is the dual of Thévenin's theorem. It is used to represent a circuit by a current source, $I_{N}$, and a parallel impedance $R_{N}$ or $Z_{N}$.

The Norton equivalent circuit can be determined by first finding the Thévenin equivalent circuit. The Norton equivalent circuit is then be found from: $I_{N} = V_{th} / R_{th}$ and $R_{N} = R_{th}$.

The Norton equivalent circuit can also be obtained by either measuring (or calculating) the short-circuit current between the two points of interest in the circuit; this short-circuit current corresponds to $I_{N}$. The Norton impedance, $R_{N}$, is found in the same manner as $R_{th}$. (Read the previous section.)

For example, the Norton equivalent circuit of figure B.7. is:

![Norton equivalent circuit](image)
Diode and LED Polarity

Transistors:

Op-Amps:

For additional information on op-amps, see a “linear data” book.
CMOS and Linear Devices:

TTL:
SEALCTABLE-OUTPUT CRYSTAL OSCILLATOR

SPG series

- Capable of selecting 57 varieties of frequency output.
- Low current consumption.
- Easy to mount DIP 16-pin package.

Specifications (characteristics)

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscillation source</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>frequency</td>
<td></td>
<td>60kHz</td>
<td>1MHz</td>
<td>768kHz</td>
<td>3MHz</td>
<td>5kHz</td>
</tr>
<tr>
<td>Pinouts &amp; Data Sheets</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Specifications

- Power source voltage: 3V to 7.2V
- Operating voltage: 3V to 7.2V
- Temperature range: -55°C to +70°C
- Power consumption: 0.1mA
- Shock resistance: 5kΩ max.
- Aging: ±5ppm/year max.
- No load condition

Electric characteristics

- Input voltage: 0.8V
- Input current: 0.1μA
- Input current (Reset): 0.2μA
- Input current (Input terminal except for Reset): 0.3μA
- Input voltage: 3V
- Output voltage: 1.8V
- Output current: 0μA
- Output rise time: 0μs
- Output fall time: 0μs
- Duty: 40% to 60%
- Max. output pulse width: 1μs
- Reset delay time: 1μs
- Reset command time: 1μs
- External signal input pulse width: 1μs
- Oscillation start up time: 0.2μs

Divider IC (without quartz crystal)

- Model name: 8650-3
- Input clock frequency: 1 MHz max.
- Current consumption: 0μA
- No load condition

RESET timing

- Input voltage: 3.3V
- Output waveform: 30% duty
- Input voltage: 3V
- Output voltage: 2.0V
- Reset command time: 1μs
- External signal input pulse width: 1μs
- 8650-3 only

External dimensions

- [Dimensions and drawings]

Block diagram

- [Diagram with pin connections]

Pinouts & Data Sheets / TTL:
DM54123/DM74123 Dual Retriggerable One-Shot with Clear and Complementary Outputs

General Description
The '123 is a dual retriggerable monostable multivibrator capable of generating output pulses from a few nanoseconds to extremely long duration up to 100% duty cycle. Each device has three inputs permitting the choice of either leading-edge or trailing edge triggering. Pin (A) is an active-low transition trigger input and pin (B) is an active-high transition trigger input. A low at the clear (CLR) input terminates the output pulse, which also inhibits triggering. An internal connection from CLR to the input gate makes it possible to trigger the circuit by a positive-going signal on CLR as shown in the truth table.

To obtain the best and trouble-free operation from this device please read the operating rules as well as the NSC one-shot application notes carefully and observe recommendations.

Features
- DC triggered from active-high transition or active-low transition inputs
- Retriggerable to 100% duty cycle
- Direct reset terminates output pulse
- Compensated for $V_{CC}$ and temperature variations
- DTL, TTL compatible
- input clamp diodes

Functional Description
The basic output pulse width is determined by selection of an external resistor ($R_X$) and capacitor ($C_{EXT}$). Once triggered, the basic pulse width may be extended by retriggering the gated active-low transition or active-high transition inputs or be reduced by use of the active-low transition clear input. Retriggering to 100% duty cycle is possible by application of an input pulse train whose cycle time is shorter than the output cycle time such that a continuous "HIGH" logic state is maintained at the "Q" output.

Connection Diagram

Order Number DM54123J-MIL, DM54123W-MIL or DM74123N
See NS Package Number J16A, N16A or W16A

Triggering Truth Table

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>L</td>
<td>X</td>
</tr>
<tr>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>H</td>
<td>X</td>
</tr>
<tr>
<td>L</td>
<td>H</td>
</tr>
</tbody>
</table>

H = HIGH Voltage Level
L = LOW Voltage Level
X = Immaterial
Switching Characteristics at $V_{CC} = 5\,\text{V}$ and $T_A = 25^\circ\text{C}$

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>From (Input)</th>
<th>To (Output)</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_{PLH}$</td>
<td>Propagation Delay Time Low to High Level Output</td>
<td>$A$ to $Q$</td>
<td>$33$</td>
<td>$33$</td>
</tr>
<tr>
<td>$t_{PHL}$</td>
<td>Propagation Delay Time Low to High Level Output</td>
<td>$B$ to $Q$</td>
<td>$28$</td>
<td>$28$</td>
</tr>
<tr>
<td>$t_{PHL}$</td>
<td>Propagation Delay Time High to Low Level Output</td>
<td>$A$ to $Q$</td>
<td>$40$</td>
<td>$40$</td>
</tr>
<tr>
<td>$t_{PLH}$</td>
<td>Propagation Delay Time High to Low Level Output</td>
<td>$B$ to $Q$</td>
<td>$36$</td>
<td>$36$</td>
</tr>
<tr>
<td>$t_{PLH}$</td>
<td>Propagation Delay Time Low to High Level Output</td>
<td>Clear to $Q$</td>
<td>$40$</td>
<td>$40$</td>
</tr>
<tr>
<td>$t_{PHL}$</td>
<td>Propagation Delay Time High to Low Level Output</td>
<td>Clear to $Q$</td>
<td>$27$</td>
<td>$27$</td>
</tr>
<tr>
<td>$W_{(pul)}$</td>
<td>Output Pulse Width*</td>
<td>$A$ or $B$ to $Q$</td>
<td>$3.08$</td>
<td>$3.76$</td>
</tr>
</tbody>
</table>

*$C_{EXT} = 1000\,\text{pF}, R_{EXT} = 10\,\text{k}\Omega$

Operating Rules

1. An external resistor ($R_X$) and external capacitor ($C_X$) are required for proper operation. The value of $C_X$ may vary from 0 to any necessary value. For small time constants high-grade mica, glass, polystyrene, or polycarbonate, or polystyrene material capacitors may be used. For large time constants use tantalum or special aluminum capacitors. If the timing capacitors have leakages approaching 100 nA or if stray capacitance from either terminal to ground is greater than 50 pF, the timing equations may not represent the pulse width the device generates.

2. When an electrolytic capacitor is used for $C_X$, a switching diode is often required for standard TTL, one-shots to prevent high inverse leakage current (Figure 1). However, its use in general is not recommended with retriggerable operation.

3. The output pulse width ($T_W$) for $C_X > 1000\,\text{pF}$ is defined as follows:

$$T_W = K \frac{R_X C_X}{(1 + 0.7/R_X)}$$

where $[R_X$ is in Kilo-ohm]$[C_X$ is in pico Farad]$[T_W$ is in nano second]$[K \approx 0.28]$

![FIGURE 1](TLF/18532-3)

4. For $C_X < 1000\,\text{pF}$ see Figure 2 for $T_W$ vs $C_X$ family curves with $R_X$ as a parameter:

![Pulse Width vs $R_X$ and $C_X$](TLF/19539-4)

![FIGURE 2](TLF/19539-4)

5. To obtain variable pulse width by remote trimming, the following circuit is recommended:

![FIGURE 3](TLF/19539-5)

Note: "Remote" should be as close to the one-shot as possible.
Operating Rules (Continued)

6. The retriggerable pulse width is calculated as shown below:

\[ T = T_{\text{WT}} + t_{\text{BILH}} + K \times r_{X} \times c_{X} + t_{\text{BILH}} \]

The retrigged pulse width is equal to the pulse width plus a delay time period (Figure 4).

7. Under any operating condition \( C_{X} \) and \( R_{Y} \) must be kept as close to the one-shot device pins as possible to minimize stray capacitance, to reduce noise pick-up, and to reduce \( I \times R \) and \( L \text{di/dt} \) voltage developed along their connecting paths. If the lead length from \( C_{X} \) to pins (6) and (7) or pins (14) and (15) is greater than 3 cm, for example, the output pulse width might be quite different from values predicted from the appropriate equations. A non-inductive and low capacitive path is necessary to ensure complete discharge of \( C_{X} \) in each cycle of its operation so that the output pulse width will be accurate.

8. \( V_{CC} \) and ground wiring should conform to good high-frequency standards and practices so that switching transients on the \( V_{CC} \) and ground return leads do not cause interaction between one-shots. A 0.01 \( \mu \text{F} \) to 0.10 \( \mu \text{F} \) bypass capacitor (disk ceramic or monolithic type) from \( V_{CC} \) to ground is necessary on each device. Furthermore, the bypass capacitor should be located as close to the \( V_{CC} \) pin as space permits.

*For further detailed device characteristics and output performance please refer to the NSC one-shot application note, AN-366.
7474, Flip-Flops
Dual D-Type Flip-Flop

**MODE SELECT — FUNCTION TABLE**

<table>
<thead>
<tr>
<th>OPERATING MODE</th>
<th>INPUTS</th>
<th>OUTPUTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asynchronous Set</td>
<td>L</td>
<td>X</td>
</tr>
<tr>
<td>Asynchronous Reset</td>
<td>H</td>
<td>X</td>
</tr>
<tr>
<td>(Clear)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Undetermined(1)</td>
<td>L</td>
<td>X</td>
</tr>
<tr>
<td>Load &quot;11&quot; (Reset)</td>
<td>H</td>
<td>T</td>
</tr>
<tr>
<td>Load &quot;00&quot; (Reset)</td>
<td>H</td>
<td>T</td>
</tr>
</tbody>
</table>

- H = HIGH voltage level
- L = LOW voltage level
- X = Don't care

**NOTE:**
(1) Both outputs will be HIGH while both R0 and R2 are LOW, but the output states are unpredictable if R0 and R2 go HIGH simultaneously.

---

74LS373, 74LS374, S374
Latches/Flip-Flops

'373 Octal Transparent Latch With 3-State Outputs
'374 Octal D Flip-Flop With 3-State Outputs

---

**MODE SELECT — FUNCTION TABLE '374**

<table>
<thead>
<tr>
<th>OPERATING MODES</th>
<th>INPUTS</th>
<th>INTERNAL REGISTER</th>
<th>OUTPUTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load and read register</td>
<td>L</td>
<td>T</td>
<td>L</td>
</tr>
<tr>
<td>Load register and disable outputs</td>
<td>H</td>
<td>T</td>
<td>L</td>
</tr>
</tbody>
</table>

- H = HIGH voltage level
- L = LOW voltage level
- X = Don't care

**NOTE:**
(1) All voltage levels are HIGH except for the LOW-to-HIGH clock transition of HIGH-to-LOW CP transition.
(2) All voltage levels are LOW except for the LOW-to-HIGH clock transition of HIGH-to-LOW CP transition.
Students enrolled in phys4051 have computer accounts on the Methods spa-home\mxpuser file server. The account gives you access to MS Office and other course related software. Your account should be used for class related activities such as writing lab reports or obtaining information about the course. Students who have taken phys2605 should be familiar with the accounts. If you are going to enroll in phys4051, you will need your U of MN X500 username and password to log-in.
Introduction

The following paragraphs are intended to clear up some issues regarding the concepts of “common” and “ground.”

E.1. Notation

Some of the confusion about the concepts discussed here may come from sloppy notation. For example, Ohm’s law is usually written as $V = IR$. While this is (mostly) correct we know that “$V$” always refers to energy, i.e., a potential difference. Therefore, more properly, Ohm’s law should be written as $\Delta V = IR$.

By convention, the voltage difference, $\Delta V$, between two points in a circuit, $A$ and $B$, is defined as:

$$\Delta V = V_{AB} = V_A - V_B.$$  

Using this notation, Ohm’s law reduces to its familiar form of $V_A = V = IR$ only when $V_B = 0$, a situation that is by no means always given!

To illustrate the dangers of carelessly applying $V$, instead of $\Delta V$, a simple voltage divider circuit is shown in the picture below. It is a slightly modified version of a problem used on a recent final exam. Students were asked to calculate $V_{XY}$.

![Voltage divider circuit](image)

- Figure E.1. Voltage divider powered by two batteries.

Students remembered and applied the equation for a voltage divider, derived from Ohm’s law:

$$V_{\text{out}} = V_B \frac{R_1}{(R_1+R_2)}$$

They (correctly) assumed that $V_B$ corresponds to $V_{AB} = 10V$ and obtained $V_{\text{out}} = V_{XY} = 9V$. But is this correct?

Unfortunately, the right answer is $V_{XY} = 4V$. So what went wrong? The voltage divider equation quoted above was derived from Ohm’s law with the assumption that $V_B = 0$. In the situation shown above, $V_B$ is 5 Volts below $V_Y$ and, therefore, this offset must be included in the derivation of the voltage divider equation.

If $V_B$ is not 0 you will get a different expression for the voltage divider.
To sum up what we have been written so far: when analyzing circuits one should be very careful to understand with respect to what (reference) voltage one is analyzing the circuit. Since the math gets easier when the reference voltage is 0 V, circuit designers prefer to designate an arbitrary reference point in the circuit called a "common" and then arbitrarily assume that it always is at 0 V.

**E.2. Floating Circuits: Battery Powered Circuits and Devices**

The simplest example of a "floating" circuit (or a floating device) is a circuit (or device) powered only by batteries and not connected to the building's wiring. Though the voltages within such a circuit are well defined with respect to other points within the circuit, the voltage with respect to the (lab) environment or the building's wiring is unknown or unimportant; it may even change over time, i.e., it is "floating."

Contrary to what one might think, this is a very desirable property because, if required, it is very simple to turn a floating circuit into one that has a fixed voltage with respect to the other instruments or the building's wiring. (Going the other way can be extremely difficult.) Floating circuits also are less affected by electrical noise, especially if that noise comes from devices connected to the wiring in the lab.

An example of a floating circuit was shown in Figure E.1. When analyzing it we paid special attention to specify the reference voltage to which a voltage was calculated or measured. For example, for the circuit in Figure E.1., $V_X$ really has no meaning, $V_{XY}$ does.

On the other hand, we can pick and identify an arbitrary point in the circuit, i.e., a common reference point or a "common," and define that it is at 0 V. In the previous example, point Y was (arbitrarily) chosen as our reference point. (See Figure E.2. below and note the symbol.) Now writing $V_X$ is no longer ambiguous since $V_{XY} = V_X - V_Y = V_X - 0 = V_X$. Of course, selecting a different common may result in voltages that are off by a constant.

![Figure E.2. Voltage divider with a "common" reference point.](image)

Setting the common point at Y allows us now to derive the answer to the final exam question in the previous section. Using the fact that in any voltage divider the current remains constant, $I_1 = I_2$, and applying (the full version of) Ohm's law to calculate $I_1$ and $I_2$ yields: $(V_A - V_Y) / R_1 = (V_X - V_B) / R_2$.

Noting that $V_A = +5V$, $V_B = -5V$ and solving for $V_X$ gives us $V_{XY}$.

Since the terms "common" and "ground" (more on "grounds" in the next section) are often incorrectly used it is important to point out one important, but subtle, fact. Remember that we assumed that the battery operated circuit is floating, i.e., since it is not connected to the building's wiring it has no fixed or predetermined voltage relationship to it. By the same reasoning, the "common" also is floating and has no fixed relation ship to the building's wiring.

Can a circuit have more than one common? Sure, but each common is at the same voltage, i.e., by convention also at 0 V. If you recall that (ideal) wires have no voltage drop across them, then it follows that all points connected to the common with wires must therefore also be at 0 V.
Applying these ideas shows that the two circuits shown below are identical. In the right drawing, the wires between the two common points have been omitted because by convention all common points are at 0 V. Whether one prefers the first diagram or the second diagram is really a matter of taste. (If you do not like the second representation, you can always draw the connecting wires between the common points back in.) Generally, the method pictured on the right is preferred because it allows grouping of circuit elements into “easily” recognizable circuit blocks, such as filters, voltage dividers, amplifiers etc., though it has the following danger: when building such circuits, the (required) connecting wires between the commons are often overlooked! Be aware of this mistake and try to avoid it!

![Diagram of two circuits](image)

- Figure E.3. Floating circuits with common

To sum up, floating battery operated circuits and devices:
- a) can have a common point at any arbitrary point;
- b) though a common point is assumed to be 0V, with respect to the lab it is floating and, therefore, at some arbitrary potential to it.

## E.3. Grounded Circuits: Devices and Circuits Powered by the Building’s Main Power

Things do get a bit more complicated, and sometimes murky, when using devices such as power supplies, oscilloscopes and function generators. These devices are all plugged into the building’s main power. For them to work safely they must be built so that their casing is physically connected to a common point in the building which itself is connected to “ground,” i.e., short for “common ground”.

This ground is not some arbitrary (virtual) reference point but an actual physical point in a building's power circuit. It is either a long stake driven into the ground or a clamp tied to the main water inlet of a house. In either case, it is always connected to the house wiring with a very heavy, thick copper wire able to withstand 100’s of amps, and it would be dangerous to think one could (or should) override or ignore it! Furthermore, this ground point is at 0 Volts with respect to all other devices that are connected to the power in that building. Unlike a floating circuit where the common could be offset to any arbitrary voltage, this common ground always remains, or should remain, at 0 V.
For the first example consider measuring the voltage $V_{AX}$ in the (floating) circuit above using an oscilloscope. Before doing so carefully examine the connectors to the scope inputs: the outer conductor of each coaxial connector is directly mounted to the casing of the scope which, as mentioned previously, is itself connected to the building ground. In other words, the outside, by convention the black connector, of a coax cable is always connected to ground when it is hooked up to the oscilloscope. Once you connect one of the ground connectors to the floating circuit shown above, a ground point in your circuit has been established, and no other ground point can exist or be at a different potential!

Now consider what happens if we try to measure $V_{AX}$ with the scope's channel 1 and $V_{XB}$ with channel 2 as shown in the picture below.

The circuit shown above has now two common ground points, $X$ and $B$. This causes both points $X$ and $B$ to be 0 volts. To visualize this effect, remember that we can replace the connection between ground points with wires; an equivalent drawing of previous circuit is shown below.
E.4. Floating Inputs and Outputs in Power Devices

The device in the previous section, the oscilloscope, had grounded inputs. Though the casings of all devices and instruments powered by the building’s electrical power are (hopefully) always grounded, its outputs or inputs are not necessary. Unfortunately, aside from examining the device as we did in the previous section there is no simple and easy method for determining if a device’s outputs or inputs are grounded short of consulting the manufacturer’s specifications or testing it. To illustrate this, here is an example using the lab’s HP3630A power supply.
The last picture in the previous explained how to measure $V_{AX}$ and $V_{BX}$ using the battery-powered circuit. Could we still use the same setup if we replaced the floating batteries with a power supply? The answer depends on how the power supply was constructed. Is one of its outputs grounded or were both they left floating?

![Figure E.8. The adjustable power supply on the left has one of its outputs internally grounded to the case while the outputs on the power supply to the right were left floating. (Note, the 110 AC input line to the power supply has been omitted from the drawing for simplicity. The arrow across the 'battery' symbol indicates that it is an adjustable voltage source.)](image1)

It turns out that the lab power supply, the HP3630A has floating outputs. The COM output is not connected to the ground. Therefore, the power supply outputs behave like a battery-operated floating circuit. A more detailed schematic of all the outputs of the HP3630A is shown below.

![Figure E.9. Agilent E3630A power supply outputs. (Note the symbol used for ground and that the casing (and one output) is still connected to the building ground for safety reasons.)](image2)

When powering the lab circuits, which output should you use, COM or ground?

- If you use only one of the adjustable voltage sources from the E3630A power supply then COM must always be connected to your circuit to complete the current path.
- On the other hand, if two or more voltage sources are used in series, for example the +20V and the –20V, then the circuit is complete without connecting COM. Whether and how you want to connect COM in such a case depends on if you want to leave COM floating or if you want to tie it to ground.
- Connecting the ground output to your circuit is optional; it is mainly used to reduce noise or for safety reasons, for example, if the voltages applied are very large, 50 V or more.

A final comment on notation: As we have seen now, it is possible for a circuit to have both a ground and a common with the common point being offset from ground. This results in confusion interpreting voltages such as $V_X$, i.e., was $V_X$ measured with respect to ground or the common? Since $V_X$ only makes sense if the reference voltage is 0, it must be concluded that under such circumstances the voltage is with respect to ground, and not the common. To make sure though, it’s best to specify that!
To test how well you understand the previous concepts, calculate for the four circuits shown below the voltages $V_{AX}$. Assume that you use an oscilloscope to measure $V_{AX}$ and that its ground lead is connected to point X.

The voltages observed on the scope would be: a) +1V, b) undetermined, i.e. incomplete circuit, c) 10V d) 0V.

**E.5. Other Devices and Circuits Powered by the Building’s Main Power**

If you’re a bit dazzled by all this it is not surprising. First, we said that devices plugged into the building’s main power usually have grounded outputs (and inputs), and then we showed that the lab power supply has floating outputs. So what can be said about all the other devices that are plugged into the building’s power supply? The short answer is: not much. The safe answer is that unless you know everything about the device, most likely one of its outputs (or inputs) is grounded and, therefore, deal with them accordingly.

Below are some rules that should be taken with a grain of salt:
- **Power supplies:**
  Outputs of “cheap” power supplies are usually not floating; expensive ones almost always are but only over some manufacturer’s specified range!
- **AC signal sources and function generators:**
  Since it is easy to float the output from an AC source, a large capacitor or a transformer will do the job, you may assume that most such devices use floating outputs.
• Voltmeters Powered by the Building’s Main Power: They should have floating inputs if they really want to measure accurate voltages but the question is over which range they can be floated, especially when measuring DC voltages. So if you’re not sure, use a handheld meter.

**E.6. Ground Loops**

It was stated that all the (black) ground leads should be connected to a single point in your circuit. Unfortunately, if there is more than one (ground) path to this common ground point, ground loops exist. As far as circuit design goes, though ground loops are not “illegal,” they are more of a nuisance because they typically manifest themselves through oscillations and noise in your circuit.

For example, in the picture below a function generator has been connected to a device under test (DUT) and to an oscilloscope. Ch 2 is used to observe the signal before it goes through the DUT and CH 1 shows the signal after the DUT. This is a typical setup and easy to implement using a BNC Tee.

![Diagram of a ground loop](image)

• Figure E.11. Example of a ground loop.

Notice that the oscilloscope’s ground is connected to point X through two different paths, A and B. Current could flow from the scope through A into X and back through B, causing oscillations. Disconnecting the ground clips from either cable A or B (but not both) would be the proper thing to do.

**E.7. Building Wiring Conventions**

Finally, here are some closing comments regarding the building’s wiring system and conventions.

The circuits treated so far were powered by two wires; usually one held at ground and the other providing the signal path. If that is the case why do then most (safe) power cords use three wires?

The answer to that question is related to the very small, but finite resistance of real wires which we have so far ignored. We were able to ignore this effect because the currents in the circuits in the lab exercises are on the order of milliamps. Therefore, a resistance of a few Ohms, which is typical, results in a negligible voltage drop. In instruments, appliances and especially in accidents, large currents up to 100 amps and more can flow before a fuse or a circuit breaker will interrupt the current flow. Under such circumstances a “small” resistance of 1 Ohm can quickly lead to a large voltage difference of 100 Volt or more between one end of the wire and the other. The assumption we have made so far, namely that all wires connected to a common point are at the same...
potential, is no longer true in such a situation! Touching a device carrying large currents while
coming in contact with one carrying little current could be lethal!

- Figure E.12. Main power with an electrical outlet (female connector) and wire color-coding conventions.

Therefore, a third wire, called “earth,” or “ground,” has been added to most power cords. One end
is connected to the building’s ground; the other makes contact with the casing of the instrument.
Under normal circumstances no or very little current will flow through the ground wire. Therefore,
even with a small resistance in the wire, the casing will remain very close to the ground potential.
Meanwhile any current used by the device is supplied through the “hot” wire and returned through
the “neutral” wire. Hot and neutral can carry large currents forcing to the actual device inside the
casing to a voltage different from ground. Since the entire device is shielded by the case this
should pose no danger to the user.
are there for!
F.1. Introduction

F.1.1. Overview

The input and output impedance $Z_{\text{in}}$ and $Z_{\text{out}}$ describes how a device is affected when it is connected to another device. Specifically, the devices discussed here are voltage sources such as power supplies, detectors and transducers connected to a load, for example, a scope, a voltmeter or an amplifier.

The input / output impedance is viewed under three different conditions. First, its effect on slowly varying, including DC signals, or steady state signals is considered and “ideal” input / output conditions are derived. Next, the optimal power transfer is explained. Finally, the effect of the input / output impedance on very “fast” signals in a transmission line is discussed.

F.1.2. Mechanical Analogy

Input and output impedance is not limited to electronic circuits. It can exist in mechanical devices. Therefore, it may be helpful to start with a mechanical example.

As mentioned earlier, $Z_{\text{out}}$ (and $Z_{\text{in}}$) play important roles in power transfer when one device, for now called the “transmitter,” is connected to a “load.” See Figure 1 below.

![Figure 1: Mechanical "transmitter" with a load.](image)

The transmitter consists of a (infinitely powerful) motor or oscillator bolted to the floor and connected to a spring with stiffness $k_{\text{out}}$. The load consists of a spring with stiffness $k_{\text{load}}$ attached to a rigid wall. Our main interest is to observe the amplitude of the horizontal displacement, $dx$, between the two devices.
It should be clear that the relative stiffness of the two springs has a great impact on the displacement amplitude. For example, if the load spring is significantly stiffer than the $k_{\text{out}}$ spring then the observed amplitude will vanish entirely. (If you are not convinced then consider the extreme case with the load spring being a rigid bar attached to the wall and the $k_{\text{out}}$ spring a very weak, easily compressible spring.)

How does this simple mechanical analogy relate to electrical circuits? The motor represents a voltage (or current) source and the stiffness of the spring attached to it is inversely proportional to the device’s output impedance, i.e., $Z_{\text{out}} \propto 1/k_{\text{out}}$. (The stiffer the spring, the “lower” its impedance.) The displacement, $dx$, is the voltage detected at the load.

Studying how the load and the output impedance affects the signal allows to predict how much of a signal we are able to transmit to another device, what its optimal power match would be and how much of it might reflect back into the transmitter.

**F.2. DC and Slowly Varying Signals**

**F.2.1. Introduction**

At first we will examine how the output and input impedance affects the amplitude of a signal transmitted for slowly varying signals, $f < 1 \text{ MHz}$. (Note: At this point we are not interested in what input and output impedance will give us the best power transfer; this is a separate issue, and one which provides a different answer and which will be addressed in the next chapter.) Instead we approach this topic from an experimental viewpoint where the signal source is usually some sort of detector whose strength (amplitude) we want to measure accurately at the load without having to worry about the attenuation introduced either by $Z_{\text{out}}$ or any load attached to the transmitter. Additionally, we might also ask ourselves, what makes an “ideal” transmitter?

**F.2.2. Output Impedance: $Z_{\text{out}}$**

**F.2.2.1. “Ideal” Transmitters**

To answer the question what constitutes and “ideal” transmitter we once more use the mechanical analogy from the previous section. We ask ourselves, what type of spring constant, i.e., $k_{\text{out}}$ would be “ideal” for such an “ideal” transmitter? By “ideal” we mean a transmitter whose output signal is the least affected by any other device attached to it and where the entire amplitude of the driver is directly applied to a load and not “wasted” by $k_{\text{out}}$.

Clearly, an infinitely stiff spring attached to the motor, or if you prefer a metal bar, satisfies this condition. As long as the motor is powerful enough, the displacement of any device attached to it will be identical to that of the motor. In electrical terms, this corresponds to $Z_{\text{out}}$ being ideally zero or “small.” If this condition is fulfilled, then any device attached to the voltage source will get the full voltage drop across its inputs.

At this point you may wonder what catch is and why we would spend any effort discussing voltage sources with finite or large output impedances or why anyone would use a non-ideal source. There are two situations when it is important to consider the (finite) output impedance: when dealing with “real” voltage sources (as opposed to “ideal” ones) and when building a voltage source with passive components, such as resistors.

**F.2.2.2. Real Voltage Sources**

Unfortunately, ideal voltage sources do not exist though a good (= expensive) power supply will come close to it. The limitation of most “real” sources is that they are not able to supply an infinite
amount of power even over a short time to maintain a fixed voltage. A battery is a good illustration of this limitation.

Though the exact behavior of each real voltage source is complex, for mathematical purposes, one can approximate its behavior by thinking of the real device as composed of an “ideal” device (with infinite power) and with a finite output resistance. See the figure below.

![Figure 2: 9 Volt battery and its circuit representation](image2)

If the device is attached to a “simple” resistor, as shown below, you can see how $Z_{out}$ and $R_{load}$ form a voltage divider. All the power dissipated by $Z_{out}$ is essentially “wasted” in the battery and never reaches the load.

![Figure 3: 9 Volt battery with load](image3)

**$Z_{out}$ Example 1: Batteries**

Most car batteries are nominally 12 VDC. Though stringing 8 AA batteries in series will also produce 12 VDC there is no way you will be able to substitute your heavy (and expensive) car battery with the (cheap) AA batteries if you want your car to start even in a balmy Minnesota summer day. Why is this the case since both configurations provide 12 VDC?

Certainly, to judge from its size and weight alone, a car battery can store more (chemical) energy and, therefore, provide power for an extended time. Additionally, we also have to consider the output impedance of the two types of batteries.

Some additional information to analyze this problem: Typically, the starter on a car requires up to 100 Amps. In other words, at 12 VDC, you may think of the load resistance as $V_{load}/I_{load} = 0.12 \, \Omega$.

Lets apply our previous knowledge and assume that both types of batteries have identical (infinitely powerful) voltage sources, at least for a short time, (both at 12 VDC) but substantially different output impedances. Though we don’t know what the output impedance of the car battery is, we...
know it must be substantially less than 0.12 \( \Omega \). Assuming that we want to receive at least 90\% of the 100 Amps required, then \( Z_{\text{out}} \) must be 0.012 \( \Omega \) or less.

What is the \( Z_{\text{out}} \) for the AA batteries? As you will measure in one of the lab exercises, it is on the order of a few ohms. Even if it were as low as 10 \( \Omega \), the batteries could at best only supply 0.12 Amp, far short of the required 100 Amps.

**F.2.2.3. Real Voltage Sources: Transducers and Detector**

Another category of “real” voltage sources that play an important role in physics experiments are transducers and detectors. They too can be modeled on voltage or current sources whose output changes with respect to physical conditions such as pressure, temperature, light, etc. Unfortunately, most of these sources not only provide small signals but their output impedance is extremely large! Usually we can not alter the actual physical process that produces these signal so we have to learn to live with these high output impedances and keep this very fact in mind, especially when dealing with input impedance!

**F.2.2.4. Voltage Sources Created with Passive Elements**

It has already been stated that a (good) power supply approaches the behavior of ideal voltage sources. Unfortunately, electronic components require various different supply voltage levels; +15, +12, +5, -12, -15 VDC are some of the more typical ones. When using a large number of electronic components, it is therefore unavoidable that many different supply voltages will be required. Instead of employing a large collection of power supplies, each set to a specific supply voltage, most circuits are usually driven at most by only two power supplies: one providing the largest positive and, if required, another the most negative voltage. Any voltage level in between can then be obtained using a voltage divider built from a pair of resistors, (passive components.) (More fancy solutions do exist but are too complicated to discuss here.) As you will see in the following example, considering \( Z_{\text{out}} \) for this type of voltage source will be very important.

**Example 2: Resistive Voltage Divider**

Assume that you are designing a circuit that needs two different supply voltages (in addition to ground) to power its devices, let’s say +20 VDC, +10 VDC and that you want to use only one, though very good, power supply which for all practical purposes has a \( Z_{\text{out}} = 0 \).

Since it is very difficult to step up (increase) a DC voltage, your only choice is to set the power supply to +20 VDC and then to use a (resistive) voltage divider to obtain the required +10 VDC. Since \( Z_{\text{out}} = 0 \), we will ignore the output impedance of the actual voltage source and assume that \( V_2 \) always remains at +20 VDC.

![Circuit 1](Figure 4: Voltage source created from resistive components)
Close inspection will reveal that, without any load, \( V_1 = +10 \text{ VDC} \) for \( R_1 = R_2 \). Applying the voltage divider equation, it appears that the actual values of \( R_1 \) and \( R_2 \) are unimportant as long the two resistors are identical.

Therefore, let’s first study this circuit by selecting large resistors, arbitrarily chosen at \( R_1 = R_2 = 100 \text{ k\Omega} \). It should come as no surprise that when you connect a “good” voltmeter to this circuit you should read \( V_1 = 10.0 \text{ VDC} \).

This exercise, as it has been stated, is still incomplete. Since we are building a primitive “power supply” we also want to connect some load to it, or what’s the point? Arbitrarily, for illustration, we use a “medium” load, 1 k\( \Omega \) resistor which would draw 10 mA of current at the expected 10 VDC. (In reality we would like to specify the maximum current drawn from this voltage divider.) Measuring \( V_1 \) again with the 1 k\( \Omega \) load shows that it has now dropped to about 0.2 V, far from the 10 V required. (Please confirm these numbers for yourself!)

What happened? Our simple initial calculation neglected to take the \( Z_{\text{out}} \) of \( V_1 \) into consideration. (Note: if \( Z_{\text{out}} \) of \( V_1 \) were not 0 we would have to take that into consideration too!) For now, let’s state that the output impedance of \( V_1 \), \( Z_{\text{out1}} \), is \( R_1 \) and \( R_2 \) in parallel, i.e., \( R_1 R_2 / (R_1 + R_2) \) with \( V_{\text{out}} = V_1 \). (For a more detailed analysis see the section following this one.) In our case, \( Z_{\text{out1}} = R_1 / 2 = 50 \text{ k\Omega} \), a rather large value. In other words, \( V_1 \) is the result of the voltage divider formed by \( Z_{\text{out1}} \) and \( R_{\text{Load}} \), \( V_1 = V_{\text{out}} R_{\text{Load}} / (Z_{\text{out1}} + R_{\text{Load}}) \approx 0.2\text{ V} \). By choosing large values for \( R_1 \) and \( R_2 \) we created a voltage divider that is easily affected by \( R_{\text{Load}} \). So how do we fix this problem?

Since we started with \( R_1 \) and \( R_2 \) being large, we chose in our second approach small values, say 100 \( \Omega \). Applying the load resistor used previously we find that \( V_1 \) with a load = 9.5 V. That’s not great but it is within 5% of the requested 10.0V, i.e., within the accuracy of typical electronic components.

Continuing along the same path of reasoning, one might wonder what prevents us from selecting even smaller values for \( R_1 \) and \( R_2 \). Doing so would decrease \( Z_{\text{out}} \) even more and make the voltage divider even “stiffer”. Before proceeding, let’s quickly calculate the power being dissipated, i.e., wasted, by \( R_1 \) and \( R_2 \) when no load is connected. (The current flowing through the voltage divider resistors without any load attached is often referred to as the \textit{quiescent} current, i.e., the current that flows when all is “quiet.”) Using two 100 \( \Omega \) resistors corresponds to “wasting” 1 Watt in each resistor, or a total 2 Watts. On the other hand, if we were to decrease \( R_1 \) and \( R_2 \) to 10 \( \Omega \), we get a far stiffer voltage divider; \( V_1 \) with the 1 k\( \Omega \) load would be 9.95V, but the power being wasted without any load attached amounts to 10 W in each resistor! At that point, even if the power supply can handle that, you are starting to create a small electric heater instead of an electric circuit. (If those figures do not mean anything, consider how hot a 40 W light bulb gets as it burns twice the amount of power as our 10 \( \Omega \) circuit.)
Conclusion: generally, we would like to have the smallest possible output resistance. Nevertheless, when designing circuits, especially when using only resistive elements we have no choice but find a compromise between how low we want the $Z_{out}$ to make and how much power we are willing to waste. Once you learn about active components, transistors op-amps and ICs in general, you will learn of different ways to create a low output impedance without having to waste quite as much power though as a general rule, you will always be faced with this dilemma.

A final comment before concluding this section: if you read the analysis carefully you may have wondered why we do not use an entirely different approach. Since we already (arbitrarily) selected a 1 kΩ load, we could solve the voltage divider equations more carefully for $R_1$, $R_2$ and $R_{Load}$ so that $V_1$ is exactly 10 V. As you can easily prove to yourself, in such a case $R_2 = R_{Load} R_1 / (R_{Load} + R_1)$ would fulfill this condition. We did not use this approach because $V_i$ without any load attached could greatly deviate from its “nominal” value. For example, if we keep $R_2 = 100$ kΩ, and $R_1 = 909$ Ω, then only with a 1 kΩ, $V_i = 10$ V but without a load it would be at 18.3 V.

Since loads attached to a circuit usually vary, the choice between this approach and the former one is whether we want $V_i$ to exceed or to fall short of the nominal value. Since devices are more easily damaged if the voltage is exceeded then falls short of it, we stay with our original approach. Also, once you have managed the concept of output impedance, you will get a pretty good feeling what type of loads you can “safely” attach to a device. Just keep the simple fact in mind that when your load is identical to the output impedance, the “nominal” voltage will drop by half!

$Z_{out}$ Calculations for the Previous Example

When replacing circuit 2a with the equivalent one, circuit 2b, how do we determine $V_{s1}$ and $Z_{out}$? A (correct) approach would be to find circuit 2a’s Thevenin equivalent but for completeness, we determine it here by first principle.

What do we mean by the circuits “being equivalent”? Essentially two conditions must hold:
1) $V_i$ for circuit 2a must be identical to $V_i$ for circuit 2b when no load is attached. 2) $V_i$ for circuit 2a must be identical to $V_i$ for circuit 2b when identical loads are attached to each circuit. Note, condition 2 must hold for any arbitrary load!

Let’s start with condition 1, the open circuit voltage:
$V_{1open}$ for circuit 2a is the product of the voltage divider formed by $R_1$ and $R_2$, i.e., $V_{1open} = V_{s2} R_1 / (R_1 + R_2)$.
$V_{1open}$ for circuit 2b is simply $V_{s1}$. From this:
$V_{s1} = V_{s2} R_1 / (R_1 + R_2)$

Next we consider condition 2, the voltage with a load, $R_{Load}$, attached:
For circuit 2a: $V_{iLoad} = V_{s2} R_1 / (R_1 + R_{Load} R_2)$ where $R_1 / R_{Load}$ means $R_1$ in parallel with $R_{Load}$.
For circuit 2b: $V_{iLoad} = V_{s1} R_{Load} / (R_{Load} + Z_{out})$
Combining these last two equations leads to: $V_{s2} R_1 / (R_1 + R_{Load} R_2) = V_{s1} R_{Load} / (R_{Load} + Z_{out})$
Finally combining this equation with the last one from condition 1 and solving for $Z_{out}$ yields after some tedious algebra:
$Z_{out} = R_1 R_2 / (R_1 + R_2)$

F.2.3. Input Impedance: $Z_i$

F.2.3.1. Introduction

$Z_i$ is the “inverse” of the $Z_{out}$ concept. If our main concern is to observe the largest possible signal amplitude then we would like the input impedance to be as large as possible. Going back to our
mechanical analogy, it represents the stiffness of the spring that is connected to some existing system. You may think of this spring as the “recorder” observing the motion of a system we are monitoring. Clearly, if the monitored system is very weak, (it already exhibits a large output impedance) or if it exhibits very minute oscillations, hooking it to a stiff spring might entirely alter or even destroy the behavior we want to observe. For this reason, we want to connect it to a spring that extremely floppy so that the original system is perturbed as little as possible. Again, we are not interested in the optimal power transfer from the output device to the input but what we want is that the original signal from the source reaches the receiver with no attenuation.

**F.2.3.2. Transducers, Detectors and Followers**

As already mentioned, transducers are notorious for their large output impedance. From an engineering viewpoint, it is often impossible to reduce the output impedance significantly by changing the physical characteristics of the device without sacrificing sensitivity. Therefore, such devices are usually directly connected to an amplifier. Contrary to what one might think, these amplifiers provide very little or no voltage gain. (An amplifier with a voltage gain of 1 is called a “follower.”) Instead their purpose is to provide an impedance change. They output the original input while providing a very low \( Z_{\text{out}} \). For now we skip on how this is accomplished but generally it involves active components such as transistors or op-amps. From an energy conservation viewpoint, one can see that such a device will only work if it receives some (additional) external power.

**Figure 6: Follower circuit: The external power inputs that are required to power \( V_s \) are not shown.**

We have now specified the ideal output impedance of a follower but what should its input impedance be? Since \( V_{\text{out}} = V_{\text{in}} \), we want \( V_{\text{in}} \) to be as large as possible. Similarly, we don’t want to perturb, load down, or attenuate \( V_{\text{in}} \). Therefore, \( Z_{\text{in}} \) should be as large as possible.

**F.2.3.2.1. Example**

The circuit below is deceivingly simple consisting only of an (ideal) voltage source and a resistor.

**Figure 7:**

Without any calculations, you should see that \( V_{\text{out}} \) (without any load) attached should be 10 V. Nevertheless, when you actually measure it with a decent digital DVM you will observe a value that
is far below the 10 V; depending on the actual DVM used in the lab, it will indicate 5 volts or less. So what’s going on?

When using a (good) measuring device, it is tempting to assume that its input impedance is infinite so that it does not perturb the output signal. Digital voltmeters and oscilloscopes have finite input impedances in the 1 to 10 MΩ range. Most of the times, we get away ignoring the effects that such a large input impedance has on the measurements, which is exactly why the desired input impedance was chosen to be large. Nevertheless, there are situations, this example being one of them, where we have to consider the effect the input impedance has on the measurements. This is certainly the case when the device’s output impedance is comparable to the input impedance, as in this example. Assuming that the DVM’s input impedance is 10 MΩ, then the measured signal will be half of $V_s$.

![Figure 8: Same circuit as figure 7 but with a Digital Volt Meter (DVM) attached.](image)

Another word of caution: the input impedance for a measurement device is usually not constant and can vary when the input range (sensitivity) of the device is changed! This may explain (sometimes) why you obtain different readings when switching between different range scales!

### F.3. Power Transfer

In the previous sections, our intention was to observe the largest possible voltage signal and we concluded that under ideal conditions our “transmitter” should have a $Z_{\text{out}}$ close to 0. Though we approached the problem only in terms of voltage, it still would hold if were to calculate the power transmitted to the “receiver” or load. In such a case, see the circuit below: $P_{\text{Load}} = I_{\text{Load}} V_{\text{Load}} = V_s^2/R_{\text{Load}}$ i.e., all the power from the source gets absorbed by the load, regardless of the size of the load.

![Figure 9: Source with $Z_{\text{out}} = 0$. Though $Z_{\text{out}}$ is 0, it has been drawn for completeness; it has really no effect on the circuit.](image)

In reality, a $Z_{\text{out}} = 0$ is something to be wished for but usually is not achievable. Especially when working with AC or high frequency (RF) signals the inductance and capacitance of the cables and connectors will add a finite $Z_{\text{out}}$ to any driver, even an “ideal” one. Under such circumstances, i.e.,
with a finite $Z_{\text{out}}$, what is the “optimal” power transfer that can be achieved between the transmitter and the load?

![Circuit Diagram](image)

**Figure 10: Power transfer with a finite source impedance.**

From the circuit above, you can see that the power in the resistive load is:

$$P_{\text{Load}} = \frac{V_s^2 R_{\text{Load}}}{(R_{\text{Load}} + Z_{\text{out}})^2}.$$  Differentiating this with respect to $R_{\text{Load}}$ and setting the result to zero shows that the optimal power transfer is achieved when $R_{\text{Load}} = Z_{\text{out}}$. In such a case, half of the power is “wasted” in the output device and half is transmitted. Any other combination will result in a decrease of the power transmitted to the load.

To recapitulate, ideally we want a device’s output impedance to be as low as possible; if $Z_{\text{out}}$ is finite, then the optimal power transfer is achieved if $Z_{\text{out}} = R_{\text{Load}}$.

If you paid close attention, we seemed to have arrived at a contradiction. Borrowing terminology from the previous section, we can consider $R_{\text{Load}}$ as the $Z_{\text{in}}$ of the “receiver.” If this is the case, then aren’t we saying that in the ideal situation (for power transfer) $Z_{\text{in}} = 0$, which is exactly the opposite of the conclusion in the previous chapter?

Not if you keep the following fact in mind: in the previous chapter we were not interested in transferring power. Instead, our aim was to “sense” or measure the voltage signal without affecting the signal source, usually a transducer of some sort. For example, in our mechanical analog, we connected another spring as a monitor to our “transmitter” without perturbing the original system. When measuring something, we want the “monitor” to remove as little power from the system as possible, so our argument still holds.

Yet there are some good reasons (as you will see in the next chapter) that some devices have low input impedances. Such devices are only used with others that have identical, i.e., “matched,” input and output impedances - an impedance matching value of 50 $\Omega$ is probably the most common. Therefore, you should be careful when using such devices and never mix them with devices that are not matched. Also keep in mind that for such matched devices, the input signal is always half of the source signal. To confuse things further, some devices, like our HP function generator, will compensate for this by sending out twice the nominal signal when it is set up for matching 50 $\Omega$ impedances! (Generally, we want to disable this setting and use its “High Z” output setting. For more information see: [http://mxp.physics.umn.edu/f99/hp%2033120a%20setup.htm](http://mxp.physics.umn.edu/f99/hp%2033120a%20setup.htm))

### F.4. Pulses and Terminating Transmission Lines

So far we have only considered DC or steady state signals, which covers 99% of what you will work with in the MXP lab. Input and output impedance considerations do get a bit more complicated when dealing with transients or short pulses as, for example, are very common in particle experiments.

We use again a mechanical analogy to illustrate this effect. You should be familiar with the mechanical concept of a displacement pulse traveling down a piece of rope. As you may recall, if
the end of the rope is rigidly held down, the pulse will be reflected back at the end of rope 180 degrees out phase. If the end is not held down, then the pulse will be returned in phase.

This analogy holds for electrical signals as well. A short voltage pulse will travel down a cable (or transmission line) and then may reflect at the end of the cable. If the end is an open circuit, the reflected signal will be in phase with the original signal and the reflected pulse will maintain the original amplitude. In contradiction to the “slow” signal case, short-circuiting the end of the cable will still reflect the pulse, i.e., it will have the original amplitude, but it is now 180 degrees out of phase.

This behavior can have troublesome consequences. Consider this example: a particle passes through a detector (typically a scintillator attached to a photomultiplier tube) creating a very short (on the order of a few nanosecond) voltage pulse that is then sent through a cable to a counter. If the input impedance of the counter is infinity, the pulse will be reflected back to the photomultiplier, where depending on its output impedance, it might very well be reflected back into the counter, resulting in erroneous counts. This is a serious problem and we need therefore something to destroy, absorb or “terminate” the pulse once it arrives at the input of the counter.

Figure 11: Pulse transmitted from source to a detector. As long as $Z_{\text{out}}$ is not identical to $R_{\text{load}}$, pulses will be reflected.

Since reflections are caused whenever a pulse encounters a change in impedance, one solution to this problem is to send the pulse (once it has been received by the counter) down and onto an infinitely long cable from which it never will return. Another, more practical one, is to determine the “characteristic impedance,” $Z_o$, of such an infinite long cable and then to construct an electronic device that has the same electrical properties as the “infinite” cable. When this device is then attached to the finite cable, it will appear to the pulse as an infinitely long cable and it will annihilate the pulse completely. Note: the characteristic impedance is not the DC resistance of the cable with which we are familiar but the impedance of the cable at a particular frequency based on its resistance, capacitance and inductance:

$$Z_o = \frac{V_{x-xo}}{I_{x-xo}} = \frac{R + \omega L}{\sqrt{G + \omega C}}$$

Where $R$ represents the resistance along the cable, $G$ the conductance between the two conductors, $C$ and $L$ the capacitance or inductance respectively. (All quantities are divided by unit length.)

At high frequency, i.e., for very short duration pulses, the characteristic impedance, $Z_o$, will approach:

$$Z_o = \frac{L}{\sqrt{C}}$$

i.e., it will become purely resistive. Therefore, the electronic gadget which we mentioned previously that will have the same characteristic as an infinitely long cable, becomes simply (matching) resistor. Below is a table with the characteristic impedances of the most frequently used cables:
<table>
<thead>
<tr>
<th>Name</th>
<th>Characteristic Impedance (Z₀)</th>
<th>Type &amp; Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>RG-58</td>
<td>50 Ω</td>
<td>Typical coaxial cable used, for example, in the lab</td>
</tr>
<tr>
<td>RG-59</td>
<td>75 Ω</td>
<td>Coaxial cable used in some High Energy Applications</td>
</tr>
<tr>
<td>10BaseT</td>
<td>100 Ω</td>
<td>Twisted (2 wire) pairs in computer networks</td>
</tr>
<tr>
<td>Twinlead</td>
<td>300 Ω</td>
<td>Parallel 2 wires used in FM antennas</td>
</tr>
</tbody>
</table>

You might wonder how attaching a 50 Ω terminating resistor, a “terminator,” to an RG-58 fulfills our initial discussion of an ideal device having an infinite input impedance, 50 Ω being anything but a small impedance. Note that such terminators are only used either when short pulses are present (in which case the output is most of the time at ground anyhow) or when very high frequency signals are used in which case the output is usually matched for an optimal power transfer, with \( Z_{\text{out}} = Z_{\text{in}} = Z_0 = 50 \ \Omega \).

Figure: Picture of a BNC 50 Ω Terminator

F.5. Terminology: Practical Resistor Values

For all practical purposes, what are considered a low or large resistance in circuits?

Resistances below 10 Ω are for all practical purposes treated as short circuits, (short) wires, or 0 Ω. 50 to 100 Ω resistors are very low resistors and you should be careful when using these because they absorb so much power, they can burn up, literally!

On the other end of the spectrum, resistances above 10 MΩ are considered infinite resistance. Though you might find resistors that are larger than 10 MΩ, grease and dirt from your hands, moisture in the air can easily reduce their effective resistance. This is why people tend to avoid these whenever possible.

In conclusion, most circuit designs rely on resistors values between 100 Ω to 1 MΩ.

F.6. Problems

F.6.1. Design a (resistive) power supply that will provide the following output voltages: \( V_3 = +15 \), \( V_2 = +12 \), \( V_1 = 5V \). (See the figure below.) Use an “ideal” power supply with \( Z_{\text{out}} = 0 \) and use it for the \( V_3 = +15V \) output.

F.6.1.1. Select values for \( R_1 \), \( R_2 \) and \( R_3 \) to obtain above specified output voltages. Furthermore, choose them so that the quiescent power from all the resistors is 1 Watt.

F.6.1.2. Calculate \( Z_{\text{out1}} \) and \( Z_{\text{out2}} \) i.e., the output impedances at \( V_1 \) and at \( V_2 \). Assume (as usual) that no load is attached.

F.6.1.3. The design below has one shortcoming: any load applied to \( V_1 \) (or \( V_2 \)) will affect \( V_2 \) (or \( V_1 \)). To decouple the two outputs we try a different setup consisting of two voltage dividers connected in parallel to an (ideal) power supply. Additionally, we specify that the output impedances of the two voltage dividers are identical, i.e., \( Z_{\text{out1}} = Z_{\text{out2}} \). Calculate the values for the 4 resistors required for this new setup that satisfy all the conditions specified above, including that the quiescent power from all the resistors is 1 Watt. Also calculate \( Z_{\text{out1}} \) and \( Z_{\text{out2}} \).
F.6.2. You are told that a (high impedance) detector’s output voltage for a given condition will be on the order of a few volts but you observe only a 0.1 V signal on an oscilloscope whose input impedance is 1 MΩ. When you then disconnect the detector and attach it to a digital voltmeter (DVM) with an input impedance of 10 MΩ the DVM displays 0.5 Volt reading.

F.6.2.1. What is the detector’s voltage signal when no load is attached and what is its output impedance?

F.6.3. The earths (vertical) electric field is about 100 V / m near sea level. (See also problem 4). We would like to measure it directly with a DVM to study how it changes with weather conditions and as function of day and night. We build a detector consisting of two 1 m² metal sheets mounted vertically 1 m apart, held in place with “perfect” insulators. We then attach the sheets to a very good DVM with $Z_{in} = 10 \, \text{M} \Omega$. The output impedance of the “detector” will be comparable to the resistance of air, which is $4 \times 10^{13} \, \Omega$.

F.6.3.1. Draw the circuit, including the DVM, and indicate the voltages and resistances.

F.6.3.2. What voltage will be displayed by the DVM?

F.6.3.3. Assume that our “perfect” mounts are real and have a resistance of 1 GΩ. How is the answer in 6.3.2. affected, if at all? (Draw a new diagram.)

F.6.4. You have bought the most powerful sound system on the market but find that the lights in your house dim each time you turn it on. You want to measure how much electrical power the system uses and find that with the sound system on, the household voltage drops from 110 VAC to 100 VAC. From your physics class, you remember that a typical output impedance of a household circuit is 1 Ohm.

F.6.4.1. What is the input impedance of your sound system?

F.6.4.2. How much current does it draw?

F.6.4.3. What is its power consumption?

F.6.4.4. What is the maximum current that you could (theoretically) draw from the given output impedance. (Note: “typical” households are rated 100 Amps.)

F.6.5. A person finds himself shipwrecked on a tiny, uninhabited island with an advanced game boy, tin cans and an introductory physics E&M textbook. After the batteries to the game boy run out, having no other diversions, he reads the E&M book and learns to his maddening delight that the earths (vertical) electric field near sea level is about 100 V / m. After removing the two dead AA batteries and vertically mounting two flattened tin cans about 3 cm apart on well isolated supports, he then connects two metal strips between the tin cans and to the battery terminals. Frustrated by the lack of success, he tosses the E&M book in the ocean but hangs on to the game boy. What the E&M book neglected to state was that such a system (i.e., air) has a resistivity, $\rho = 4 \times 10^{13} \, \text{Ohm m}$. (See the Handbook of Physics and Chemistry.)
F.6.5.1. Assume a realistic value for the surface area of the flattened cans and calculate the $Z_0$ of this system. (Remember that: $R = \rho L / A$, where $L$ is the distance between the cans and $A$ the top (or bottom) surface area.)

F.6.5.2. What’s the maximum power to be drawn from this tin can system?

F.6.5.3. What surface area is needed if one wants to get at least 1 mW out of the system, the bare minimum power requirement for some low power chips? Assume that the electric field does not change as a function of the large number of tin cans required.
Introduction:

Most of the problems collected here are actual test problems given during the previous 15 years in Phys4051. Solutions to some of them are posted on our web site though you should always try to solve them first without consulting the solutions.

G.1. Voltage Dividers/Thevenin Equivalent Circuits

1.1) All the voltages in the circuit in Figure 1.1 are with respect to ground, which is not shown explicitly. Determine the value of $R_1$ such that $V_x$ will be at +2 V with respect to ground.

1.2) a) For the circuit in Figure 1.2 calculate $V_{\text{Thevenin}}$ and $R_{\text{Thevenin}}$ between points A and B.
   
   b) Draw the Thevenin equivalent circuit of the circuit shown in Figure 1.2 and label $V_{\text{Thevenin}}$ and $R_{\text{Thevenin}}$ with the appropriate values.

1.3) Given the circuit shown in Figure 1.3:
   a) With the resistor $R_{\text{Load}}$ removed, derive equations for the Thevenin voltage and Thevenin resistance across AB.
   
   b) Make a sketch of the equivalent circuit.
   
   c) With $V_1 = 10$ Volts, $V_2 = 20$ Volts, $R_1 = 10k\Omega$ and $R_2 = 30k\Omega$, find the numerical values for $V_{Th}$ and $R_{Th}$.
   
   d) Using the values given in 1.3c) calculate the current through $R_{Load} = 7.5k\Omega$. 

Figure 1.1.

Figure 1.2.

Figure 1.3.
1.4)
   a) For the circuit in Figure 1.4 calculate $V_{AB} = V_A - V_B$.
      
   b) For the circuit in Figure 1.4 calculate $I_1$.

1.5) For the circuit in Figure 1.5 calculate $V_{AB} = V_A - V_B$.

1.6) Find the Thévenin equivalent circuit for the circuit shown in figure 1.6. Specifically calculate $V_{\text{Thévenin}}$ and $R_{\text{Thévenin}}$ when looking into the circuit at V1 and ground. Draw the Thévenin equivalent circuit.

1.7) For the circuit given in Figure 1.7:
   a) Find the voltage $V_{AB}$.
   b) Find the output resistance $R_{AB}$.
   c) Write down the Thévenin equivalent circuit.
1.8) a) For the circuit in Figure 1.8, write the equations from which $I_1$, $I_2$ and $I_3$ can be determined.
b) Find the magnitude and direction of $I_3$.

1.9) a) For the circuit in Figure 1.9 write down the equations, but do not solve them, that you would use to determine, $i_1$ and $i_2$.

1.10) Find the Thévenin equivalent circuit for the circuit shown in Figure 1.10 between points A and B. Draw the Thévenin equivalent circuit and calculate the values for $V_{Th}$ and $R_{Th}$.

Use the following values:
$V_1 = V_2 = V$

1.11) You have bought the most powerful sound system on the market but find that the lights in your house dim each time you turn it on. You want to measure how much electrical power the system uses and find that with the sound system on, the household voltage drops from 110 VAC to 100 VAC. From your physics class, you remember that a typical output impedance of a household circuit is 1 Ohm.
a) What is the input impedance of your sound system?
b) How much current does it draw?
1.12) The circuit in Figure 1.11 is a Wheatstone bridge which can be used to measure small differences in resistances.

a) Calculate \( V_+ \) and \( V_- \) (with respect to point A) for the Wheatstone bridge.

b) Calculate \( V_{\text{out}} \), where \( V_{\text{out}} = V_+ - V_- \) when \( R_1 = R \).

c) Calculate and draw the Thévenin equivalent circuit for the Wheatstone bridge across \( V_{\text{out}} \) again assuming that \( R_1 = R \).

d) Given \( R_1 = R + \delta \) (where \( \delta << R \)) show that \( V_{\text{out}} \) is proportional to \( \delta \). (Use reasonable approximations.)

e) Given \( R_1 = R + \delta \) (where \( \delta << R \)) calculate \( R_{\text{Thevenin}} \) across \( V_{\text{out}} \). (Use reasonable approximations.)

f) From the information in d) and e) draw the Thévenin equivalent circuit for the Wheatstone bridge across \( V_{\text{out}} \) with \( R_1 = R + \delta \) (where \( \delta << R \)).

1.13) You built a temperature sensor using the circuit in Figure 1.12, where the left top resistor, \( R_T \), is temperature sensitive. The resistance is \( R \) when the temperature is 20°C but above that, the resistance increases 1% every 1°C, and below 20°C, it decreases 1% every 1°C.

When the temperature is 30°C, how much current will flow in the ammeter in the middle? Would it be from the left to the right, or the other way around? Assume that the internal resistance of the ammeter is negligible compared to \( R \).

1.14) For the circuit shown in Figure 1.13, find the equivalent circuit as shown in Figure 1.14. In particular, find \( R_{\text{eq}} \) and the noise voltage \( V_n \) in terms of \( R_1, R_2 \) and \( R_3 \) and a coefficient \( a_n \) that depends on \( T \) and bandwidth \( B \), thereby showing that \( R_{\text{eq}} \) characterizes both the net circuit impedance between A and B and the noise between A and B.

b) Give an expression in terms of \( T, B, R \) and constants for the Johnson noise generator \( V_n \) associated with a resistance \( R \).

c) If \( R_1 = 1 \text{k}\Omega \) and \( R_2 = R_3 = 2 \text{k}\Omega \), what is the noise voltage between A and B in a bandwidth of 10 kHz with the resistors at room temperature.
1.15) a) What is the Thevenin equivalent of the circuit shown in Figure 1.15 (as seen between $V_{Out}$ and Ground). Find both the $V_{Th}$ and $R_{Th}$. Draw the circuit’s Thevenin equivalent circuit diagram and clearly label $V_{Th}$ and $R_{Th}$.

b) What is the value of $V_{Out}$ if a 10k load was attached between $V_{Out}$ and Ground?

1.16) The circuit in the figure below uses a three terminal adjustable resistor, also know as a trim pot, familiar to you from the lab. The trim pot is used to obtain an adjustable output voltage, $V_{out}$.

The resistance at the wiper terminal, pin 2, with respect to pin 1 is:

$$R_{12} = \alpha R_o$$  \hspace{1cm} (1)

The resistance at the wiper terminal, pin 2, with respect to pin 3 is:

$$R_{23} = (1 - \alpha) R_o$$  \hspace{1cm} (2)

Where:

- $R_o$ is the resistance between pins 1 and 3 and in this case it is 1 k;
- $\alpha$ is proportional to the angle that the trim pot has been adjusted and its range is: $0 \leq \alpha \leq 1$.

a) Without any load attached, what is the range of the output voltage?
b) Calculate the Thevenin Resistance, $R_{Th}$, for the circuit at $V_{out}$ with respect to ground. (No load attached.)
c) Find an expression for $V_{out}$ with respect to ground in terms of $R_o$ and $\alpha$ when no load has been attached. (Hint: check your answer with the one obtained in a.)
d) Find an expression for $V_{out}$ with respect to ground in terms of $R_o$ and $\alpha$ with a load $R_L$ attached between $V_{out}$ and ground.
2.1) Consider a typical 4 terminal black box with two inputs and two outputs as shown in Figure 2.1. Design a simple RC filter circuit using only one resistor and one capacitor that, when put into the black box, results in the following characteristics at the frequencies \( f \) indicated:

a) \( Z_{\text{in}} \big|_{f=0} = \infty \) (no load attached)

b) \( Z_{\text{in}} \big|_{f=\infty} = 1k\Omega \) (no load attached)

c) \( Z_{\text{out}} \big|_{f=0} = 1k\Omega \) (with an ideal voltage source connected to the input)

d) \( Z_{\text{out}} \big|_{f=\infty} = 0 \) (with an ideal voltage source connected to the input)

e) \[ \frac{V_{\text{out}}}{V_{\text{in}}} \big|_{f<1kHz} = 1 \]

f) \[ \frac{V_{\text{out}}}{V_{\text{in}}} \big|_{f>1kHz} = \frac{1}{\sqrt{2}} \]

In your drawing, clearly label the values for all the components.

2.2.) A square wave with frequency \( f_0 \) is applied to the circuit in Figure 2.2 at \( V_{\text{in}} \).

a) Draw the input and the corresponding output observed at \( V_{\text{out}} \) over two periods of the square wave, when: \( f_0 << f_{3dB} \).

a) Draw the input and the corresponding output observed at \( V_{\text{out}} \) over two periods of the square wave, when: \( f_0 >> f_{3dB} \).

2.3.) Figure 2.3 and 2.4 show the input and output voltages vs. time of a simple RC circuit. (Note the different time scales for each plot.) From these figures determine:

a) Find the time constant \( \tau \) (20% error is acceptable)

b) Is the RC circuit a highpass or lowpass filter?

c) Is it an integrator or differentiator when driven as shown in Figure 2.4?

d) Draw the RC circuit that would produce outputs as shown in Figure 2.3 and 2.4 when hooked up to an oscilloscope. (You do not need to specify the component values.)

e) What is the \( f_{3dB} \) point of this circuit?
2.4.)

a) For the circuit in Figure 2.5, find the complex transfer function, i.e., calculate $V_{out}/V_{in}$ as a function of $\omega$.

b) From part a) find the absolute value or modulus of the transfer function.

c) What is the output impedance of this circuit when looking into $V_{out}$ (with respect to ground) at $\omega = 0$ and $\omega = \infty$?

2.5.) With the holidays just a few weeks away, Figure 2.6 shows a circuit which could be used to produce blinking lights.

It consists of a 155 VDC power supply, a resistor, a capacitor and a neon light bulb.

The neon bulb has the following characteristics:

- It takes 60 Volts ($V_{on}$) to turn the bulb on; once turned on the bulb acts like resistor with $R_{on} = 1k\Omega$. 

Selected Problems / G.2. RC Circuits
• When the voltage falls below 40 V, \(V_{\text{off}}\) the bulb turns off; its resistance is that of an open circuit.

The voltage across the bulb \(V_{\text{Ne}}\) as a function of time is shown in Figure 2.7, and is represented by solid lines.

(Note that the switch \((S_1)\) was closed at \(t = -t_0\) and that it will remain closed indefinitely.)

Determine how long the bulb stays on as follows:

(a) During the charging cycle, while the bulb is OFF, what voltage would \(V_{\text{Ne}}\) reach if the bulb were never to turn on, i.e., find \(V_a\)?

(b) During the discharge cycle when the bulb is ON, what voltage would \(V_{\text{Ne}}\) approach if the bulb were to stay on forever, i.e. find \(V_b\)?

(c) Write down the differential equation for the voltage at \(V_{\text{Ne}}\) during the discharge cycle when the bulb is ON, i.e, find \(dV_{\text{Ne}}/dt\) in terms of \(V_{DC}, R_1, R_{\text{On}}\) and \(C\).

(d) What is the DC impedance (or the Thévenin resistance, \(R_{\text{Thévenin}}\)) that the bulb sees when it is ON and looking into the circuit with the switch closed. (Be sure to include \(R_{\text{On}}\).)

(e) Find a mathematical expression for how long the bulb stays on in terms of \(V_{DC}, V_{\text{on}}, V_{\text{off}}, R_{\text{Thévenin}}, R_1, R_{\text{On}}\) and \(C\). (Hint, you can either solve the previous differential equation or, use the fact that you already know that the voltage across the bulb follows the typical exponential decay, which has a time constant specified by the capacitor and the impedance that the bulb sees. Furthermore, from knowing the voltages at \(t = 0, t = \tau\) and \(t = \infty\) you should be able to determine the various constants in the decay expression and solve for \(\tau\).)

(f) Find a numerical value for \(\tau\) if \(C = 400 \mu\text{F}\).

2.6. Figure 2.8. shows one of many types of capacitance bridges. \(C_x\) and \(R_x\) form the equivalent circuit of a real capacitor. Note that the two variable components to achieve balance are \(R_1\) and \(R_3\).

a) Find the transfer function at point A.
b) Find the transfer function at point B
c) Find the values of \(C_x\) and \(R_x\) (in terms of the other 4 components) for which the bridge is balanced, that is, \(V_A - V_B = V_{AB} = 0\). Be sure to balance both the real and imaginary parts.
Note that this bridge has the advantage that the balance condition is frequency independent.

2.7.) a) For the circuit in figure 2.9, find the voltage between points A and B, $V_{AB}(\omega)$, as a function of $V_s(\omega)$.
b) For $\omega = 0$ and $\omega = \infty$ find $V_{AB}(\omega)$. Do these values make sense? Explain briefly.
c) Include a plot of $\frac{V_{AB}}{V_s}$ and $\varphi_{AB}$ as a function of frequency.
d) What is the output impedance of this circuit when looking into terminals A and B at $\omega = 0$ and $\omega = \infty$?

2.8.) Show that a leaky capacitor $C_p$ in which the leakage resistance is represented by a shunt $R_p$ (Figure 2.10) is equivalent to a capacitance $C_s$ in series with a resistance $R_s$ (Figure 2.11). Find the values of $C_s$ and $R_s$ in terms of $C_p$, $R_p$ and the angular frequency, $\omega$, of the applied alternating voltage $V_s = V_0 \cos(\omega t)$.

2.9.) The circuit in the figure 2.12 represents a typical scope probe (R1 and C1) and a scope input (R2 and C2).
a) Calculate the impedance for a resistor R in parallel with a capacitor C.
b) Assume $R_1 C_1 = R_2 C_2$. Calculate the transfer function $(|\frac{V_{out}}{V_{in}}|)$ for the circuit on the right.
c) What is the frequency dependence of the transfer function calculated in part b?
2.10) a) Sketch how a 15k resistor and a 0.001uF capacitor can be connected to act as a high-pass filter. Show the input and output connections of the filter in your sketch. What is meant by the -3dB or “half-power” point for a high-pass filter and where is it for this filter?

b) A square wave is applied to this circuit with a frequency 10 times less than the -3dB point. Draw the input and output waveforms.

2.11) While in principle one can measure an impedance by measuring the voltage, current and the phase differences, this has the disadvantage of requiring very precise measurements with a stable, accurate power source. Further, it is difficult to measure small changes in the unknown impedance (such as the resistance of a strain gauge or a temperature dependent resistor.)

This situation can be greatly improved in a way analogous to weighing objects with a beam balance, i.e., null (zero) the output by balancing the voltage drop in one arm of a bridge with that in an arm containing the unknown. One does not even need the same types of devices in the two arms, as is shown in Figure 2.14b which represents a Maxwell bridge. In this type of bridge one fixed, calibrated capacitor $C_1$ and two variable and one fixed, calibrated resistors are used to measure an inductor $L_x$ (which has some loss given by $R_x$).

![Figure 2.13](image1)

![Figure 2.14](image2)

a) For the bridge shown in Figure 2.13, obtain the condition for balance ($V_{AB} = 0$) in terms of the four $Z_i$'s.

b) Apply this result to Figure 2.14 and find $L_x$ and $R_x$ in terms of the other four components. (Note: If all the $Z_i$'s are purely resistive, then this becomes a Wheatstone bridge.)

2.12) Suppose that the output impedance of your uncle’s amp is $8\Omega$. Your aunt gives you speakers called tweeters, which work well at high frequencies. The manual says its input impedance is $8\Omega$, and that you should attenuate signals of frequencies less than 1 kHz. You remember a high-pass filter from your physics class, and you decide to use it to attenuate lower frequency components. Knowing the 3dB point is given by $f = 1/(2\pi RC)$, thinking that $R = 8\Omega$, and $f = 1000$ Hz, you find a capacitor of $C = 1/(2\pi f R) = 20 \mu F$. See figure 2.15 on the right.
a) In order to check its performance, you measure the output voltage (with a constant AC signal input) at 10 kHz and 1000 Hz. Without the speaker (and capacitor) connected, the output voltage was the same at both frequencies: 6 $V_{pp}$. When one of the speakers was connected, the output was very close to 3$V_{pp}$ at 10 kHz. Explain why this should be expected instead of 6 $V_{pp}$. Feel free to make reasonable approximations if it makes your calculation easier.

b) At 1 kHz, you measure 2.7 V. You expected that at the 3dB point, the output would be reduced by a factor $1/\sqrt{2} = 0.707$ (compared the 10 kHz reading of 3 $V_{pp}$), which is 2.1 $V_{pp}$. Why does this happen? Or calculate the expected voltage for the circuit shown.

### G.3. LRC Circuits

#### 3.1.) Find the transfer function for the circuit in Figure 3.1. in terms of $L$, $R$ and $C$.

You do not need to simplify your answer and you do not have to calculate the modulus or absolute value of the function!

![Figure 3.1.](image)

#### 3.2.) Consider the circuit in Figure 3.2:

a) Assume $V_{in} = V_0 \cos(\omega t)$. What is the (time-dependent) current $I$, flowing to ground that you would measure?

b) If $L = 40 \mu H$ and $C = 5 \times 10^{-10} F$, at what frequency (in Hz) does the current $I$ vanish?

![Figure 3.2.](image)

#### 3.3.) Given the RLC circuit shown in Figure 3.3:

a) Derive the equation for the complex impedance $Z$ at the input of the circuit as shown in Figure 3.3.

b) Show that the expression for the magnitude of the "transfer function" is:

![Figure 3.3](image)
\[ A(\omega) = \frac{|V_{\text{out}}/V_{\text{in}}|}{\sqrt{\frac{R}{\sqrt{R^2 + 1 - \left(\frac{\omega_r}{\omega}\right)^2}}}}, \text{ where } \omega_r^2 = \frac{1}{LC} \]

c) Give the values of \( A(\omega) \) at:
1) \( \omega = 0 \)
2) \( \omega = \omega_r = \frac{1}{\sqrt{LC}} \)
3) \( \omega \to \infty \)
and make a sketch of \( A(\omega) \). Indicate in the sketch the bandwidth \( \Delta \omega \).

d) Show that the quality factor \( Q = \frac{\omega r}{\Delta \omega} = \omega_r RC \) for this circuit. (Hint: At \( \omega = \omega_r \pm \frac{1}{2} \Delta \omega, \)
\( A(\omega) = \frac{1}{\sqrt{2}} \))

3.4) a) An LRC circuit, as shown in figure 3.4, is driven with an AC wave of the form \( V(t) = V_0 \cos(\omega t) \). Give an expression for \( V_{\text{out}}(t) \) including real amplitude and real phase in terms of \( L, R, \) and \( C. \)

b) Write the absolute value of the transfer function of this circuit.

c) Consider the limit \( R \to 0 \), and the limit \( (RC)^2 \gg LC \). Is the circuit a low-pass, high-pass, notch, or resonant filter in any one of these limits? Give a one or two sentence justification for your answer.

3.5) Consider the filter on the right:

a. Write an expression for the transfer function of this circuit.

b. Discuss the asymptotic behavior of this filter, at high and low frequency. What type of filter is it? Find the characteristic frequency of the filter.

c. What phase shift is introduced by this filter at very small frequencies, very large frequencies and at the characteristic frequency?

d. On a log-log scale, sketch the magnitude of the transfer function and the phase shift as a function of \( \omega. \)
G.4. Diodes and Transformers

4.1.) Determine $V_{AB} = V_A - V_B$ for each circuit in Figure 4.1. In each case assume that $V_{Diode,On} = 0.6$ V.

4.2.) For the circuits in Figures 4.2, 4.3 and 4.4 calculate $V_x$ (with respect to ground.) You may assume that the diode forward voltage drop, $V_{DiodeON} = 0.6$V.

4.3.) For the circuit in Figure 4.5 assume that $V_{in} = 2 \sin(2000 \pi t)$ Volts. Clearly draw and label the output voltage $V_{AB}$ (specify its
amplitude(s)) as a function of time for the following conditions:

a) Assume that $V_{\text{Diode\_On}} = 0$ V, i.e., no forward diode voltage drop and unlimited forward bias current. Draw the output for the circuit in Figure 4.5 at $V_{AB}$ vs. $t$ over a 2 msec time interval.

b) Assume that $V_{\text{Diode\_On}} = 0.7$ V. Draw the output for the circuit in Figure 4.5 at $V_{AB}$ vs. $t$ over a 2 msec time interval.

4.4.) For the circuit in Figure 4.6, assume that the diodes are ideal (i.e. reverse current = zero) with forward drops of 0.6 V. $R_L = 100$ k$\Omega$. The output impedance of the function generator is 600 $\Omega$, and with no load connected it outputs a 6.0 V peak-to-peak sine wave.

![Figure 4.6.](image)

a) Plot $V_{out}$ vs. $t$ for at least one full period. Indicate clearly (with voltages!) the important features of the signal.

b) Assume, $R_L = \infty$, what are the magnitudes of the largest currents $I_{1\text{max}}$, $I_{2\text{max}}$, and $I_{3\text{max}}$ that flow in each of the branches 1, 2 and 3?

4.5. For each of the following 5 circuits imagine you are connecting a scope between ground and $V_{out}$. Draw one cycle of what you see on your scope. The drawings can be qualitative (though accurate), but label important voltage levels. Note: The voltage across the whole transformer secondary is 20 V$_{pp}$ in all cases, the frequency is 100 Hz. Also, $R_L = 1$ k$\Omega$ and $C = 100$ $\mu$F. Assume that the forward voltage drop across the diode is 0.5 V.

![Figure 4.7](image)  ![Figure 4.8](image)
4.6.) A transformer is used to match the impedance of a 400 Ohm source to a 100 Ohm load.
   a) What turns ratio is needed?
   b) Does the primary or secondary have more turns?
   c) What is the voltage ratio of the primary to the secondary?
   d) What is the current ratio?

4.7.) Consider the diode circuit in Figure on the right. The diodes are semi-ideal with a voltage threshold of 0.7 Volt. Determine the voltage at point A, V_A.

5.1.) The modified common emitter amplifier circuit, shown in figure 5.1 on the right, suffers from a large dependence on the transistor’s β. Relying on the assumption that the transistor has a β close to 100, the components for the circuit were selected to produce a quiescent output voltage of 8 V to give the amplifier its largest dynamic range (the maximum amplification that can be achieved without distorting the output.)
Unfortunately, after building the circuit it was found to have quiescent output voltage of only 2 V (making the dynamic range substantially smaller than planned). Based on that information, what was the "real" $\beta$ of the transistor?

Assume that the transistor operates in its unsaturated regime and that the transistor has a base-emitter drop of 0.6 V. You may use appropriate approximations; i.e., if your results are accurate to within 10%, i.e., $\beta + 1 \approx \beta$, and the methods are appropriate, you get a full credit.

a) First calculate the quiescent collector current, $I_C$ given that $V_{out} = 2.0 \text{ V}$.

b) What is the actual $\beta$ of the transistor if the quiescent output voltage, $V_{out} = 2.0 \text{ V}$? (Hint: What can be calculated easily after you get the collector current? Then what?)

5.2.) Assume that the transistor in Figure 5.2 operates in unsaturated regime and that it has a $h_{fe} = \beta = 100$ and a base-emitter drop of 0.6 V. You may use appropriate approximations; i.e., your results should be accurate to within 5%.

a) For $V_0$ very small ($V_0 << 2.6 \text{ V}$), find the following DC voltages and currents: $V_1, V_2$ and $I_2$.

b) Using the information previously stated, calculate the DC values $I_3$ and $V_3$.

c) Calculate the AC voltage gain of the circuit, i.e., find $V_{out}/V_0$ for $V_0$ small.

5.3.) Assume that the transistor in Figure 5.3 does not operate in the saturated mode and that it has a $h_{fe} = \beta = 100$ and a $V_{Diode_{On}} = 0.6 \text{ V}$. You may use appropriate approximations, i.e., your results should be accurate to within 5%.

a) For $V_s$ very small, find the following DC voltages and currents: $V_E, V_C$ and $I_E$.

b) Using the information previously stated, calculate $I_C$ and $I_E$.

c) Calculate the AC voltage gain of the circuit, i.e., find $V_{out,AC}/V_s$ for $V_s$ small.
5.4.) The circuit in Figure 5.4 provides a constant current over a wide range of $R_L$, i.e., it acts as a constant current source. You may make the following assumptions in answering a through d: $V_{\text{Diode On}} = 0.6 \text{ V}; h_{fe} = \beta = 100; V_{\text{sat}} = 0.1 \text{ V}$. Your result should be accurate to within 5%.

a) Find $V_B$

b) Find $V_E$

c) Assume $R_L = 1 \text{k}\Omega$. Find $I_C$.

d) What is the range of $R_L$ for which the current calculated in c) will remain constant? Specifically for what value of $R_L$ will the transistor be in its maximum saturation mode? Is there a minimum value for $R_L$?

5.5.) You are given the current source shown in the figure 5.5. A relay coil that draws 10 mA at 4.0 V is being driven from a 10 V supply as shown. You know that both transistors are in their active mode and:

\[
\begin{align*}
V_{c2} &= 1.9 \text{ V} \\
I_{c2} &= 10 \text{ mA} \\
I_1 &= 3.0 \text{ mA}
\end{align*}
\]

Assume that the transistors have $h_{fe} = 100$ and base-emitter drops of 0.5 V. You are required to determine all voltages, currents, and resistances to within 5%.

(a) What are $V_{B1}, V_{C1}, V_{E1}, V_{B2},$ and $V_{E2}$?

(b) What are $I_{B1}$ and $I_{B2}$?

(c) What are the resistances $R_1, R_2,$ and $R_3$?

(d) What is the input impedance of this circuit measured at point A?

(e) In two sentences or less, explain the reason for the diode across the relay coil. Hint: What happens when the circuit is switched off?

5.6.) Analyze the circuit in figure 5.6: (Use appropriate approximations to obtain voltages to 0.1V and the currents to 10%, assume $h_{fe} = 50$)

a) Determine the values for $R_1$ and $R_2$ such that the voltage at the base is 4.0V and that the current that flows into the base ($I_B$) is about 1% of $I_R$. 

Figure 5.4.

Figure 5.5

Figure 5.6.
b) With no sources or loads connected, what are the values of $I_B$, $I_E$, $I_C$, $V_E$, and $V_C$?

c) An AC signal is coupled to the input through a large capacitor $C$. What is the voltage gain of output1? Of output2?

d) What is the input impedance of the circuit as shown at $f >> 0$?

e) Find the capacitance $C$ required to place the lower frequency -3dB point at 10Hz.

5.7.) Analyze the circuit in figure 5.7a: (Give the voltages to 0.1V and the currents to 10%, assume $h_{fe} = 100$)

a) With no sources or loads connected what are the values of $I_E$, $I_B$, $I_C$?

b) With no sources or loads connected what are the values of $V_E$, $V_B$, and $V_C$?

c) An AC signal is coupled to the input through a large capacitor; what is the voltage gain of output1?

d) An AC signal is coupled to the input through a large capacitor; what is the voltage gain of output2?

e) What is the input impedance of the circuit as shown?

f) Replace the emitter circuit with the circuit shown in figure 5.7b, what are the DC ($\omega = 0$) and the high frequency ($\omega > 1/\tau$) gains of the circuit?

5.8.)

a) What are the relationships between the base current, $i_B$, the collector current, $i_C$, and the emitter current, $i_E$, for a transistor in the active mode?

b) An NPN transistor with a $\beta$ of 100 is connected as shown in Figure 5.8. Make a sketch of the circuit and show the direction of the currents $i_B$, $i_C$, and $i_E$ flowing through the transistor.

c) Calculate the currents $i_B$, $i_C$, and $i_E$ and the voltages at points $X$ and $Y$.

d) What is the operating mode of the transistor in this circuit?

e) A small sinusoidal signal $v$ is applied at $Y$ (through a large blocking capacitor) with an amplitude of 0.01 V. What is the amplitude and phase of the voltage signal at point $X$?
5.9.) For the following transistor circuit in Figure 5.9., what are the values of: (you may make reasonable approximations)

a)  $V_B$?

b)  $V_E$?

c)  $I_E$?

d)  Current through the Lamp?

5.10.) Calculate the following values for the circuit in Figure 5.10: Assume: $V_{sat} = 0$, $V_{BE} = 0.6V$, $h_{fe} = 100$ and the quiescent (output) operating point is 7.5V.

a)  The quiescent collector current is 5 mA. What value should $R_c$ be?

b)  The circuit should have a gain of -10; what value should $R_e$ be?

c)  Using the values for $R_c$ and $R_e$ previously calculated, what is the voltage at the base?

d)  Neither $R_1$ nor $R_2$ can exceed 20k; calculate the largest possible values for $R_1$ and $R_2$.

e)  What is the input impedance at $In$?

5.11) The circuit in Figure 5.11 provides a constant current over a wide range of $R_L$, i.e., it acts as a constant current source. You may make the following assumptions in answering a through f: $V_{Diode,On} = 0.6V$; $h_{io} = \beta = 100$; $V_{sat} = 0.1V$. Your result should be accurate to within 10%.

a)  Find $V_B$

b)  Find $V_E$

c)  Assume $R_L = 1 k\Omega$. Find $I_C$

d)  Assume $R_L = 1 k\Omega$. Find $I_B$

e)  Assume $R_L = 1 k\Omega$. Find $I_D$

f)  Assume that $V_{CC}$ is “noisy”, i.e., it may fluctuate as much as 10% above or below the stated 12 V. Calculate or state how $I_C$ will be affected with $R_L = 1 k\Omega$.
5.12) Given Figure 5.12:

(a) $V_{CC}$ is:
   (a) 8 V
   (b) 10 V
   (c) 0 V
   (d) -2 V

(b) $V_B$ the voltage at the base is
   (a) 8 V
   (b) 2 V
   (c) 0 V
   (d) -2 V

(c) Which of the following statements is correct?
   (a) since there is no resistor near the base the current into the base would be very large and the transistor will be in saturation for all values of $R_C$
   (b) since there is no resistor near the base the current into the base would be very large and the transistor will burn very quickly after power-on regardless of $R_C$
   (c) the transistor will be in an active mode regardless of $R_C$
   (d) the transistor is either in the active mode or in saturation, depending on the value of $R_C$

(d) $V_E$, the emitter voltage
   (a) is undefined since the transistor is not operating
   (b) is one diode drop below the collector
   (c) is one diode drop below the base
   (d) depends on $R_C$
6.1.) Assume the op-amp in Figure 6.1 to be ideal, i.e., \( A = \infty \), \( V_{cc} = +15 \text{ V} \), \( -V_{cc} \leq V_{out} \leq +V_{cc} \).

The input voltage, \( V_{in}(t) \), in Volts, is:
\[ V_{in}(t) = 5.0 \sin(2000 \pi t) - 3.0. \]

a) Plot \( V_{in}(t) \) vs. \( t \) over two periods. Clearly indicate the voltages and times on your plot.

b) What is the voltage at the non-inverting input, \( V_{+} \)?

c) On the plot from part a), superimpose \( V_{out}(t) \) vs. \( t \). Clearly label all voltage levels.

6.2.) For the circuit shown in Figure 6.2, you may assume that the op-amp is "ideal", i.e. that \( A = \infty \) and that \( V_{out} \) saturates at \( \pm V_{cc} = \pm 15 \text{ V} \).

a) Which type of feedback does the circuit in figure 6.2 employ?

b) In terms of \( V_{+} \), over what range of the inverting input voltage, \( V_{-} \), does \( V_{out} = +V_{cc} \)? In terms of \( V_{+} \), for which range of \( V_{-} \) will \( V_{out} = -V_{cc} \)?

c) Find \( V_{+} \) as a function of \( V_{in} \), \( V_{out} \), \( R_{1} \) and \( R_{2} \). (You may assume \( V_{out} = \pm V_{cc} \).)

d) The waveform in figure 6.3 is connected at the input of the circuit in figure 6.2. Superimpose the output waveform on the separate sheet provided and hand it in together with your blue-book. Assume \( V_{out} = +V_{cc} \) at \( t = 0 \);

e) Draw \( V_{out} \) vs. \( V_{in} \) (i.e. draw \( V_{in} \) along the \( x \)-axis and \( V_{out} \) along the \( y \)-axis.) Label \( V_{THL} \) and \( V_{THH} \).
6.3.) Design a circuit that has the following characteristics:
   a) if \( V_{\text{in}} > 5 \text{ V} \) then \( V_{\text{out}} = V_{\text{CC}} \)
   b) if \( V_{\text{in}} < 5 \text{ V} \) then \( V_{\text{out}} = V_{\text{EE}} \).

Assume that you are using an ideal op-amp with \( V_{\text{EE}} \leq V_{\text{out}} \leq V_{\text{CC}} \), connected to \( V_{\text{CC}} = +15\text{ V} \) and \( V_{\text{EE}} = -15\text{ V} \).

a) Draw \( V_{\text{out}} \) vs. \( V_{\text{in}} \) for this circuit. Be sure to label the axes of your graph with numerical values.

b) Draw a circuit that behaves as stated above. For your design you can use two power supplies, each providing \(+15\text{ V}\) plus any number of resistors in the \(1\text{ k}\Omega \) to \(10\text{ k}\Omega \) range.

6.4.) For the circuit shown in Figure 6.4, you may assume that the op-amp is “ideal”, i.e. that \( A = \infty \), and that \( V_{\text{out}} \) saturates at \( \pm V_{\text{CC}} = \pm 15\text{ V} \).

d) Which type of feed back does the circuit in figure 6.4 employ?

e) Over what range of the non-inverting input voltage, \( V_{+} \), does \( V_{\text{out}} = +V_{\text{CC}} \)? For which range of \( V_{+} \) will \( V_{\text{out}} = -V_{\text{CC}} \)?

f) Find \( V_{+} \) as a function of \( V_{\text{in}}, V_{\text{out}}, R_{1} \) and \( R_{2} \). (You may assume \( V_{\text{out}} = \pm V_{\text{CC}} \).)

g) Using the information obtained in b) and c), show that the \( V_{\text{out}} \) switches from \( +V_{\text{CC}} \) to \( -V_{\text{CC}} \) at \( V_{\text{in}} = V_{\text{THL}} = -5\text{ V} \), where \( V_{\text{THL}} \) is the lower threshold voltage. Similarly, show that \( V_{\text{out}} \) switches from \( -V_{\text{CC}} \) to \( +V_{\text{CC}} \) at \( V_{\text{in}} = V_{\text{THH}} = +5\text{ V} \), where \( V_{\text{THH}} \) is the higher threshold voltage.

h) The wave form in figure 6.5 is connected at the input of the circuit in figure 6.4. Copy the wave form in figure 6.5 and then superimpose on it the output wave form.

i) Draw \( V_{\text{out}} \) vs. \( V_{\text{in}} \) (i.e. draw \( V_{\text{in}} \) along the x-axis and \( V_{\text{out}} \) along the y-axis.) Label \( V_{\text{THL}} \) and \( V_{\text{THH}} \).

6.5.) Calculate the period, \( T \), for the relaxation oscillator shown in Figure 6.6. Assume the op-amp to be ideal, i.e., \( A = \infty, -V_{\text{CC}} \leq V_{\text{out}} \leq +V_{\text{CC}} \).

a) Draw the relationship of the following voltages over a couple of cycles on the same graph:
   \( V_{\text{out}} \) vs. \( t \),
   \( V_{X} \) vs. \( t \),
   \( V_{Y} \) vs. \( t \)

   Label the various curves clearly.

b) Write down an equation for \( V_{Y} \) in terms of \( V_{\text{out}}, R_{1} \) and \( R_{2} \).

c) Write down a general expression for \( V_{X}(t) \) vs. \( t \);
you do not have to determine the values of the individual constants, i.e., keep it general.

d) Using the information from a, b and c, determine \( T \) for the circuit shown in Figure 6.6.

![Figure 6.7](image1.png) ![Figure 6.8](image2.png)

6.6. The circuit shown in Figure 6.8 is a variation of the familiar inverting amplifier circuit shown in Figure 6.7. Circuit 6.8 is often preferred over 6.7 because for identical input impedance and gain, circuit 6.8 requires much lower resistor values and, therefore, is less affected by (Johnson) noise.

(For the subsequent questions, you may assume that you are working with ideal op-amps and since you are using negative feedback, the Golden Rules of op-amps do apply.)

a) Use the circuit in Figure 6.7 and calculate the value for \( R_F \) if the gain, \( V_{out}/V_{in} \), of this circuit is to be \(-10^4\).

b) For the circuit shown in Figure 6.8 calculate \( V_{out}/V_{in} \) in terms of \( R_1, R_2, R_3 \) and \( R_4 \). (If you want to check your answer, for the components shown in Figure 6.8, \( V_{out}/V_{in} = -10^4 \)).

6.7. You have a load which draws a very large current. A typical experimental practice to measure the current \( I_L \) through the load is to put a small resistance \( R_s \) in series and then measure the voltage across \( R_s \). One side of the load must be attached to ground as shown in Figure 6.9. The experiment is not working well, and you need to look at the current as a function of time.

a) Assuming ideal op-amp behavior, compute \( V_{out} \) in terms of \( I_L \) (\( V_{out} \) is a linear function of \( I_L \) by design).

b) In three sentences or less, explain the role of the transistor in this circuit. In

![Figure 6.9](image3.png)
particular, what fundamental property of transistors is being exploited in order to make this circuit work?

c) Assuming $V_{in} = 10 \, V$, $R_s = 0.2 \, \Omega$, $R_1 = 100 \, \Omega$, and $R_2 = 1 \, k\Omega$, what is the largest current $I_L$ for which this circuit will work properly?

6.8.) A photodiode is hooked up as shown in Figure 6.10:

Assume that the light power incident on the photodiode is 10 nanowatts and that the sensitivity of the photodiode is 0.5 A/W (i.e. it produces 0.5 amps of current for every watt of incident power). You can assume that the amplifier is an ideal voltage amplifier with a variable gain $G$. The equivalent noise input of the amplifier at the frequency of your measurement is $e_n = 5 \, nV/\sqrt{Hz}$ (ignore any dependence on source resistance).

The only other noise sources you need to consider are:
1) the shot noise due to the photocurrent itself, which produces a voltage noise across the load resistor,
2) the Johnson noise due to the load resistor.

a) Compute the noise output voltage of this circuit in a 1 Hz bandwidth if $R_L$ is 10 Ohms and the amplifier gain is $10^5$. What is the ratio of output signal to noise under these conditions?

b) Compute the noise output voltage of this circuit in a 1 Hz bandwidth if $R_L$ is $10^6 \, \Omega$ and the amplifier gain is $G = 1$. What is the ratio of the output signal to noise?

c) Explain in a few sentences why one of the above configurations is much better than the other. (Do not just say that “the noise is smaller.” Specify the reason.)

6.9.) Assume that the op-amps U1 and U2 in figure 6.11 are ideal and that the input voltage $V_{in}$ is small.

a) For the circuit in figure 6.11 find $V_{out} / V_{in}$. (Make sure your result has the correct sign.)

b) Can you explain in a sentence or two what the purpose or advantage is, if any, of having op-amp U1 in this circuit?
6.10.) Assume that the op-amp in Figure 6.12 is ideal and that its supply voltages are +/- 15 VDC.

Before time \( t = 1 \) second, the switch S1 is closed. At time \( t = 1 \) second the switch S1 is opened. Sketch the output for \( 0 \leq t \leq 5 \) seconds. Clearly label the time axis and voltage levels.

6.13.)

a) Find the output voltage \( V_{\text{out}} \) in terms of the input voltage \( V_{\text{in}} \) for the circuit given in Figure 6.13 when the diode is biased in the forward direction. In this case, the diode I-V curve is given by \( I = I_a \left( e^{\frac{qV}{kT}} - 1 \right) \), but in practical situations the exponential factor is much greater than 1.

6.14.) In a measurement of absorption in a semiconductor, the amount of light absorbed increases exponentially over a very small wavelength range. It is sometimes necessary to measure a signal that varies over four or five orders of magnitude. This is very difficult to do with an ordinary amplifier. In this problem, we consider the following alternative:
You can assume that the op-amp is ideal. The purpose of the resistor R is to ensure that the feedback diode D is always in its conducting state. The photo-diode PD produces a current I_s that is proportional to the intensity of the incident light. The current I_s flows in the reverse direction as shown in the drawing.

a) Explain why all of the bias current I_B flows through the diode D. Indicate clearly how the assumptions of ideal op-amp behavior lead you to this conclusion.

b) Sketch the function \( I = I_0 (e^{\alpha V} - 1) \) and explain in a few sentences why it is a reasonably good approximation for the I-V curve of a typical diode. In terms of \( \alpha \) and \( I_0 \), what are the reverse current and forward voltage drop (i.e. "turn-on voltage") of the diode?

c) Assuming the I-V relation of part b), compute the output voltage \( V_{out} \) of this circuit for a given \( I_B \) and \( I_s \). In a few sentences, explain why this circuit is a good choice for measuring light signals that vary over many orders of magnitude.

d) Alas, your light signal is very small, and you have to worry about the noise in your circuit. You can assume that that the diode D and the battery are noiseless. On the other hand, the manufacturer of the op-amp specifies an input noise current of \( 2 \times 10^{-15} \, \text{A/Hz}^{1/2} \). The manufacturer of the photodiode specifies an equivalent noise current of \( 2 \times 10^{-15} \, \text{A/Hz}^{1/2} \) for the photodiode "in the dark", i.e. when there is no light shining on it. The bias resistor \( R \) is \( 2.0 \times 10^9 \, \Omega \). Given these three possible sources of noise, compute the rms noise output voltage of the circuit in the dark, assuming that you are running the output through a flat-top band-pass filter with a low-frequency cut-off of 2.0 kHz and a high-frequency cut-off of 4.0 kHz. You can assume that \( I_0 = 10^{-12} \, \text{A} \) and that \( \alpha = 15 \, \text{V}^{-1} \). Boltzman’s constant is \( 1.38 \times 10^{-23} \, \text{J/K} \).

e) You now turn on a light source, producing a photocurrent \( I_s \). At what light intensity will the shot noise due to the photocurrent be equal to the noise you calculated in part d)? The manufacturer specifies a sensitivity of 0.2 A/W, meaning that 1 watt of light produces a photocurrent \( I_s \) of 0.2 A.
6.15.) Consider the op-amp circuit in Figure 6.15 with inputs \( V_A \) and \( V_B \). (Assume the op-amp to be “ideal.”)

a) What is the voltage at the non-inverting input, \( V_+ \)?

b) What is \( V_{out} \) in terms of \( V_A \) and \( V_B \)? Describe your analysis using a few short sentences and show all of your work.

![Figure 6.15](image-url)

6.16.) Consider the amplifier circuit as shown in Figure 6.16.

For parts a, b, and c, assume the op-amp is ideal.

a) What will be the voltage at the non-inverting input of the op-amp in terms of \( R_2 \) and \( V_2 \)?

b) Write an expression for \( V_{out} \) in terms of \( V_1 \) and \( V_2 \).

c) What should be the relationship between \( R_1 \) and \( R_3 \) if you want to maximize the common mode rejection of this amplifier? Explain your reasoning in one or two sentences.

d) Assume that \( R_1 = R_2 = R_3 \). You have a real op-amp, a 355, with \( V_{offset} = 3 \text{ mV} \). Assume it to be otherwise ideal. Compute \( V_{out} \) in terms of \( V_1, V_2 \) and \( V_{offset} \).

![Figure 6.16](image-url)

6.17.) The op-amp circuit in figure 6.17 is a differential amplifier. Assume an ideal op-amp and calculate its differential gain using the superposition principle. Specifically:

a) What type of feedback is being used in the op-amp circuit in figure 6.18?

b) Calculate \( V_{out} \left( V_a \right) \bigg|_{V_b=0} \)

c) For \( V_a = 0 \) and \( V_b \neq 0 \), calculate \( V_+ \) and \( V_- \) as a function of \( V_b \).

d) Calculate \( V_{out} \left( V_b \right) \bigg|_{V_a=0} \). (Hint: use the information from 2.c.)

e) Apply superposition to b and d and calculate \( V_{out} \left( V_a, V_b \right) \).

f) Show that the circuit has a differential gain, \( \frac{V_{out}}{V_a - V_b} \) of: \( \frac{V_{out}}{V_a - V_b} = \frac{R_1}{R_2} \)

![Figure 6.17](image-url)
6.18.) On a space flight a defective feedback resistor in an inverting op-amp circuit failed, threatening the crew and its mission. An inventive / desperate astronaut, not having a kit of replacement resistors, substituted an ideal tungsten light bulb to serve with the ideal op-amp. The bulb had the following ideal properties:

1. The resistance was linearly proportional to the absolute temperature, i.e., \( R_f = \beta T \).
2. The power generated in the resistor was radiated into space (\( T \approx 0 \)) such that \( P = \alpha T^4 \).

(Everyone knows, ideal light bulbs have no heat conduction down the leads; and this particular bulb was evacuated, so convection was negligible.)

a) Find the \( I(V) \) relationship for \( R_f \).
b) Find \( V_{out} \) as a function of \( V_{in}, R_i, \alpha, \) and \( \beta \). (If you cannot complete part A, then use \( I = aV^2 \) as the \( I-V \) relationship for \( R_f \).)

6.19.) Given a “perfect” op amp (\( Z_{in} = \infty, Z_{out} = 0 \), Open Loop Gain: \( A_v = \infty \) and noiseless. Assume: \( V_{CC} = +15V, V_{EE} = -15V, V_{EE} \leq V_{out} \leq V_{CC} \) ) answer the following design questions:

a) Design (give a circuit diagram) of an inverting amplifier with an input impedance of 10k and a gain of 20.
b) Design (give a circuit diagram) of a non-inverting amplifier with infinite input impedance and 1mA maximum drain in the feedback loop and a gain of: 20.

c) The forward current through a diode is given by:
\[
I_f = I_S \left( e^{qV/kT} - 1 \right)
\]
where \( I_s \) is a device dependent constant. When \( e^{qV/kT} \gg 1 \) the forward current can be approximated by:
\[
I_f = I_S e^{qV/kT}.
\]
For the circuit in Figure 6.18, find \( V_{out} \) as a function of \( V_{in} \) in the forward biased region where the approximation is valid.
d) If \( V_{in} > 0 \), redraw the circuit in figure 6.18 with the box replaced by the appropriate diode symbol getting the polarity right so that the circuit works as calculated in part c.

6.20.) Figure 6.19 shows an amplifier circuit which uses an op-amp having open-loop gain of \( 10^5 \) and unity-gain bandwidth of 1 MHz. It has a slew-rate of 0.2V/\( \mu \)sec.

a) What is the gain of the amplifier at low frequencies? Over what range of frequencies, approximately, will the amplifier have this gain?
b) What is the input impedance of the amplifier?
c) Draw the output you would expect to see on an oscilloscope if a 1 volt peak-to-peak, 10kHz square wave is applied to the input. (Pay attention to the amplifier specifications listed above.)
d) Presumably, you have been treating the inverting input to the op-amp as a virtual ground in your estimations. How well is this justified? What is the voltage at the inverting input for the input with \( V_{in} = 1VDC \)?
6.21.) For the following questions, you may assume that you are working with an ideal op-amp and that $V_{in}$ is small. For the circuit shown in Figure 6.20:

a) Find $V_{out}/V_{in}$ with switch S1 in position 1.

b) Find $V_{out}/V_{in}$ with switch S1 in position 2 (as is shown in figure 6.20).

c) From the results from a) and b) how would you relabel position 1 and position 2 in such a way that it clearly describes the behavior of the circuit? (UP and DOWN does not count, neither does ON/OFF or OPEN/!OPEN)

6.22.) The input into each of the following circuits is a 1 kHz sine wave with an amplitude of 2.0 V$_{pp}$. Sketch 2 cycles of this input signal and 2 cycles of the corresponding output signal for each of the circuits below. Your graph should show clearly the shape and relative phase of the output signal. Indicate the numerical values of the output voltage at the extreme (maximum and minimum) points of each cycle. Assume that all of the op-amps are ideal and powered by +15 and -15 VDC.
6.23) 
\[ R_x = 100.0 \, \Omega + \delta R \]  
\[ V_s = \text{ac signal} \]  
(Give answers in \( V_{\text{rms}} \) where appropriate.)

The bridge in Figure 6.26 is balanced (\( V_0 = 0 \)) when \( \delta R = 0 \). We want to calculate the minimum detectable \( \delta R \) assuming the system has a bandwidth of 1 Hz. The resistor \( R_x \) is a platinum resistance thermometer (PRT) with a temperature coefficient of \( 4 \times 10^{-3} \, /^\circ C \) at room temperature.

a) The PRT is in a capsule. In order to keep the temperature sufficiently close to that of the bath being measured, no more than 1 \( \mu \text{W} \) of Joule heating can be tolerated. Calculate \( V_s \).

b) Calculate \( V_o/\delta R \).

c) If the amplifier is noiseless, calculate the noise due to the bridge at its input.

d) What \( \delta R \) can be detected?

e) What \( \delta T \) is this?

f) Suppose that the amplifier has a noise referred to input of \( e_n = 2nV/\sqrt{\text{Hz}} \) and \( i_n = 10pA/\sqrt{\text{Hz}} \). Now what \( \delta R \) and \( \delta T \) can be detected?

6.24.) 
\( V_{\text{in}} \) for each of the circuits on the right is a 1.0 kilohertz sine wave with a peak-to-peak amplitude of 5.00 V. For each circuit, make a sketch showing the input and output signals on the same graph as a function of time for at least two full periods of the input signal. Indicate clearly the amplitude of the output signal. You may assume that any op-amps are ideal and that they saturate at \( \pm 15 \, \text{V} \) and the forward voltage drop on the diodes, \( V_{\text{DiodeOn}} \), is 0.5 V. Assume that the output impedance of the source driving these circuits is 0 \( \Omega \).
6.25.)

a) Shown in Figure 6.31 is a summing amplifier with three inputs $V_0$, $V_1$, and $V_2$. Calculate $V_{\text{out}}$ as a function of $R_0$, $R_1$, $R_2$, and $R_F$ for any combination of (analog) input voltages $V_0$, $V_1$, $V_2$, i.e., find $V_{\text{out}}(V_0, V_1, V_2, R_1, R_2, R_3, R_F)$

b) Now turn the circuit in Figure 6.31 into a D-to-A converter: assume that $V_0$, $V_1$, and $V_2$ take on only TTL voltage levels, i.e., 0 V (LO) or 5.0 Volts (HI). Furthermore, if $V_0$ represents the LSB and $V_2$ the MSB, then the binary equivalent of an integer value $X$, where $0 \leq X \leq 7$, applied to $V_0$, $V_1$, $V_2$ should produce an output voltage that is proportional to $X$:

$$V_{\text{out}} = -kX$$

Of course, this relation holds true only if the appropriate values for $R_0$, $R_1$, $R_2$ are chosen. Determine $R_0$, $R_1$, and $R_2$ for $k = 0.1$ Volts and $R_F = 1 \, k\Omega$.

6.26) Figure 6.32 shows a two-switch version of the summing amplifier / DAC circuit previously shown in class.

a) For the circuit in Figure 6.32, calculate each of the currents $I_0$, $I_2$, $I_3$, and $I_4$ as a function of $I_0$ when both switches, $Sw_0$ and $Sw_1$, are in the position as shown in the diagram, i.e., in the up (U) position

b) Calculate $V_{\text{out}}$ in terms of $V_s$ and $R$ when both $Sw_0$ and $Sw_1$ are in the down (D) position. Assume the op-amp to be ideal and that the Golden Rules apply.
6.27) Consider the op-amp circuit shown in Figure 6.33. (Assume the op-amp to be ideal)
a) What is $V_{Out}$ as a function of $V_A$, $R_1$ and $R_2$ if $V_B = 0$ V?
b) What is $V_{Out}$ as a function of $V_B$, $R_1$ and $R_2$ if $V_A = 0$ V?
c) Find the differential gain of the circuit, $V_{out}/(V_A - V_B)$, as a function of $R_1$ and $R_2$.

6.28) The circuit shown in Figure 6.34.B is a variation of the familiar inverting amplifier circuit shown in Figure 6.34.A. Circuit 6.34.B is often preferred over 6.34.A because for identical input impedance and gain, circuit 6.34.B requires much lower resistor values and, therefore, is less affected by (Johnson) noise.

Your task is to select appropriate resistor values for the circuits in Figure 6.34.A and 6.34.B that result in identical input impedances and gain for the two circuits, i.e., $Z_{in} = 100$ kOhm and a (closed loop) gain $V_{out}/V_{in} = 10^4$.

(For all subsequent questions, you may assume that you are working with ideal op-amps.)

a) For the circuit in Figure 6.34.A what should $R_1$ be if the $Z_{in} = 100$ k?

b) Based on your previous answer, calculate the value for $R_F$ if the gain, $V_{out}/V_{in}$ of this circuit is to be $-10^4$. Is this a “realistic” resistor value?

c) For the circuit in Figure 6.34.B what should $R_1$ be if the $Z_{in} = 100$ k?

d) For the circuit shown in Figure 6.34.B calculate $V_{out}/V_{in}$ in terms of $R_1$, $R_2$, and $R_3$. (Hint: you may want to express some of your equations using the voltage at the node formed by $R_2$ and $R_3$.)
If none of the resistors in Figure 6.34.B can exceed 300k what are the values for \( R_2 \) and \( R_3 \) if \( V_{out}/V_{in} \) of this circuit is to be \(-10^4\) and \( Z_in = 100 \, k\Omega \).

6. 29) In class you were shown the output of a photo multiplier tube when a particle passes through a detector. So far none of the circuits discussed allowed for both, an adjustable hysteresis setting and an adjustable threshold level. The circuit on the right fulfills this requirement. Figure out how it works.

Specifically, on the graph below, draw \( V_{out} \) vs. \( V_{in} \), \( V_+ \) vs. \( V_{in} \) and \( \Delta V_{in} \) vs. \( V_{in} \) for this circuit. \( (\Delta V_{in} = V_+ - V_-) \).

Find the numerical values for the lower and upper thresholds voltages. Use your own approach to solve this problem but if you get stuck, suggestions will be posted on our website. The link to the suggestions can be found on the announcement page at: http://mxp.physics.umn.edu/f04/Announcements.htm

Use the following information to solve this problem:
- \( V_{++} = V_{--} = 12 \, V \).
- \( R_1 = 1k \Omega \)
- \( R_2 = 3k \Omega \)
- \( V_- = -3V \)

Assume that it is an ideal op-amp with infinite (open) loop gain and infinite input impedance and zero output impedance.

6.30) The op-amps circuits on the right all use ideal op-amps powered by \( Vcc = +12 \, V \) and \( VEE = -12V \). (Ideal means: infinite input impedance, infinite open loop gain, very low output impedance.)

Calculate \( V_{out} \) for each circuit with \( V_{in} = +3V \) for all circuits.
7.1) For the circuit in Figure 7.1, assume that the 74LS193 (used in the frequency counter lab) 4-bit binary counter's TTL output voltages are exactly: LO = 0 V, HI = 5 V and that it is configured to count continuously receiving pulses at 1 kHz on its clock input. Your assignment will be to draw \( V_{\text{out}} \) vs. \( t \) for two periods of the output signal, specifically:

a) What is the period of the output signal? (How long does it take until \( V_{\text{out}} \) repeats the same voltage pattern?)

b) What is \( V_{\text{out}} \) when all the counter outputs \( Q_0, Q_1, Q_2 \) and \( Q_3 \) are LO?

c) What is \( V_{\text{out}} \) when all the counter outputs \( Q_0, Q_1, Q_2 \) and \( Q_3 \) are HI?

d) What is \( |\Delta V| \), the change in \( V_{\text{out}} \) when the counter value decreases by one from, for example, 15 to 14, or 7 to 6.

e) From the information above, draw \( V_{\text{out}} \) vs. \( t \) for two periods of the output signal. Clearly label all voltages and times involved. (If you were not able to calculate all parts, at least try to draw a qualitative picture.)

7.2) The multiplexer circuit shown in Figure 7.2 is a different implementation of a circuit used recently in one of your lab exercises. It uses the same 8-bit multiplexer that you used for the 31-day machine. Note that \( A_0 \) is the least significant address bit.

a) Write down the complete truth table \( Q(A_2, A_1, A_0) \) (i.e. the table of all possible combinations) for the values of \( A_2, A_1 \) and \( A_0 \), and the corresponding results for \( Q \).

b) Write down in Boolean logic notation an expression that implements the truth table obtain in a) for all values of \( A_2, A_1 \) and \( A_0 \), i.e., write a Boolean expression for \( Q(A_2, A_1, A_0) \).

c) Draw a simple logic circuit using standard symbols for one or two-input gates (AND, NAND, OR, NOR, XOR, NOT) that performs the same function as the circuit shown in Figure 2.

7.3) A Majority Voting Circuit produces a HI if two or more of its inputs A, B or C are HI, otherwise its output is LO. (See Figure 7.3.)
a) Give the truth table for the Majority Voting Circuit.
b) Using standard combinational logic express $Q$ as a function of the inputs. Simplify your expression as much as you can.
c) Using a circuit diagram implement the Majority Voting circuit using standard two input gates (standard gates include AND, NAND, OR, NOR, NOT or XOR gates.)

7.4) In lecture you saw how a parallel-encoded ("flash") analog to digital converter works. A simplified two-bit version with an analog input range from 0 to 4 V is shown in Figure 7.4.

It contains a series of voltage dividers to set the various threshold voltages for the comparators and a decoder chip that converts the information from the comparators into the corresponding binary signal. Design the logic circuitry for the decoder chip which outputs a two bit binary value that corresponds to the truncated (analog) input voltage. For example, an analog input voltage of 2.0 V and less than 3.0 V will always result in a binary output of: 10.

a) Write down the truth table for the decoder chip and the Boolean logic expressions for $Q_0$ and $Q_1$ in terms of the comparator outputs, $C_{Out2}$, $C_{Out1}$ and $C_{Out0}$. (You may assume that the comparator outputs are TTL level and that if the analog input voltage exceeds a comparator’s threshold voltage ($V_-$), its output will go HI.)

b) Implement it using only standard 1 and 2-input gates such as NOT, AND, NAND, OR, NOR and XOR gates and draw a circuit diagram.

7.5) Figure 7.5 shows a pseudo random bit generator.
Assume that it has been initialized with: $Q_3 = HI$, $Q_2 = LO$, $Q_1 = HI$ and $Q_0 = LO$. Determine the decimal equivalent of its outputs $Q_3$ (MSB), $Q_2$, $Q_1$ and $Q_0$ (LSB) during the next 4 cycles.

7.6) For the circuit shown in figure 7.6, sketch a timing diagram. Clearly show in your timing diagram the propagation delay time, $\tau$, which is the same for both the NOT and the NOR gates.

7.7) In many applications 2 address lines are used to select the routing of a data line, with the unselected lines set LO. They often come with an $\text{ENABLE}$ input which sets all outputs LO. This device is known as a demultiplexer; see figure 7.7. Design such a 2-address line demultiplexer with an $\text{ENABLE}$ using the following gates: AND, NAND, OR, NOR, XOR, NOT.
7.8) For the circuit shown in figure 7.8:
   a) Describe Y in terms of A and B using Boolean algebra.
   b) Simplify this expression and determine with which single gate could you replace the circuit in figure 7.8?
   c) The propagation time through each gate is 25nsec and the rise and fall times are zero nsec. Describe using a timing diagram what happens at inputs to 3, 4, 5 and Y with B set LO, when A changes state from LO to HI and then returns to LO some time (> 100nsec) later.

7.9) Design a majority voting circuit which will give an output of HI only when 2 or more of the three inputs A, B, C are 1.
   a) Show a truth table for the majority voting circuit
   b) Show a Boolean expression corresponding to the truth table
   c) Implement it assuming that you have AND, NAND, OR, NOR and XOR two input gates available. For maximum points you must reduce the implementation to a minimum number of gates (5 or less). You must clearly show all steps in any logic reduction that you may use.
   d) Implement it assuming that you have only one multiplexer, such as the 74LS151, see Figure 7.9, available; you may not use any other chips but you are free to tie any of the inputs HI or LO.

7.10) Your lab partner wires up the multiplexer as shown in figure 7.10 as a joke. Note that it is the same multiplexer that is used for the 31 day machine and that A0 is the least significant address bit.
   a) Write down a truth table
   b) Draw the simple logic circuit using only two input gates which performs the same function as your partner’s circuit.
7.11) Shown below are two pictures A, B.

Assume black to be the LO state and white to be HI. Combine the two pictures (point by point) by applying the following Boolean operations:

a) A AND B
b) A OR B
c) A XOR B
d) A XOR A

7.12) Write the binary number \( A = 0111 \ 1010 \ 0101 \ 1111 \) as a:

a) 4 digit hexadecimal number
b) decimal (base 10) number
c) 5 digit BCD representation of the number obtained in 4.b
d) Find the (binary) 2's complement \( C_2 \) to \( A \). Add \( A \) and its 2's complement, i.e., \( A + C_2 = Z \). Is the answer what you expected?
e) Convert the decimal (base 10) number 1001 into binary
f) Convert the decimal (base 10) number 2.375 into binary

7.13)

a) For \( \text{\_\_\_\_\_} \) write out the truth table and show that \( \text{\_\_\_\_\_} \).
b) Draw a circuit of \( \text{\_\_\_\_\_} \) using 2 input AND, OR and NOT gates.
c) Using Boolean algebra show that \( \text{\_\_\_\_\_} \).
d) Using a truth table, show that \( \text{\_\_\_\_\_} \).

7.14) A 4-digit LCD display, similar to the one used in the frequency counter circuit shows 0110. Assume the digits are displayed in one of the following “base” systems.

Case 1: 0110 represents a decimal, i.e. base 10 notation, number.
Case 2: 0110 represents a binary, i.e. base 2 notation, number.
Case 3: 0110 represents a hexadecimal, i.e. base 16 notation, number.

a) Given the three cases, which one represents the smallest decimal number? Which is the largest?
b) If each digit is shifted one position to the left, and a zero is added in the right most position, then by what factor (in decimal notation) would its value have changed in each of the three cases?
c) Assume that a decimal point and a additional digit was added to the original display so it now reads 0110.1 By how much (in decimal notation) does this differ from the original display for each case?

7.15) The circuit in Figure 7.15 is a variant of a synchronous counter circuit. Unlike your typical counter that starts at 0 and then counts up or down, this counter, if properly seeded, continuously
repeats a number sequence that should look familiar to you. In this problem we will find this sequence. Note: $Q_2$ is the most significant bit (MSB) and $Q_0$ is the least significant bit, i.e., the LSB.

![Figure 7.15](image)

a) Determine the (decimal=base 10) values of the sequence if the counter starts with the numerical value 4, i.e., $Q_2 = 1, Q_1 = 0, Q_0 = 0$. To confirm your answer, you must provide a timing diagram showing at least 5 clock periods.

b) Determine what happens if the counter is seeded with any value that is not part of its sequence, for example, 7? (Note: you only need to determine the result for one such value; the result is identical for all other values that are not part of the counting sequence.)

7.16) Shown on the next page is a 7 segment display similar to the one used in the Frequency Counter lab. It consists of 7 bar-shaped LEDs labeled A through G. By turning the appropriate combination of the LEDs on and off, all numerals and some letters can be displayed. For example, to display the number 0 all LEDs except G are turned on; for the number 1, only B and C are turned on, etc. (See the diagram on the next page on the right.) You will design part of the logic driver circuit for such a 7 segment display.

![Diagram of 7 segment display](image)

Your driver must be able to display the numerals 0 through (and including) 9. Each numeral is represented by its 4 bit binary number representation, $n_3$ $n_2$ $n_1$ $n_0$ with $n_0$ representing the LSB and $n_3$ the MSB. For example, the numeral 5 is represented by 0101.
a) Complete the truth table shown below for the remaining numerals and indicate the LED segments that have to be turned on, or off.

<table>
<thead>
<tr>
<th>INPUTS</th>
<th>OUTPUTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>n₃ n₂ n₁ n₀</td>
<td>A B C D E F G</td>
</tr>
<tr>
<td>0 0 0 0</td>
<td>1 1 1 1 1 1 0</td>
</tr>
<tr>
<td>0 0 0 1</td>
<td>0 1 1 0 0 0 0</td>
</tr>
</tbody>
</table>

Etc.

Fill in the rest of the truth table.

b) From your complete truth table, write down the Boolean Logic expression that represents the LED segment labeled E in terms of its inputs n₃, n₂, n₁, and n₀. (You do not have to simplify this expression.)

7.17) A decade counter counts in binary from 0000 (decimal 0) to 0001, 0010 etc. up to 1001 (decimal 9) before it repeats this sequence. An incomplete circuit diagram of a synchronous decade up-counter with 4 D-type Flip-Flops is shown on the next page. Note that Q₀ represents the LSB and Q₃ the MSB.

a) Write down the complete truth table for such a synchronous decade up-counter, i.e., show all the Qᵢ outputs (i.e., Q₃, Q₂, Q₁, and Q₀) and the Dᵢ inputs (i.e., D₃, D₂, D₁, and D₀) which will produce the sequence specified previously.

b) Work out the Boolean logic for the D₃ input in terms of Q₃, Q₂, Q₁, and Q₀, i.e., find the appropriate Boolean logic expression to implement D₃( Q₃, Q₂, Q₁, Q₀).

c) Draw the logic circuit for the D₃ input (from part b) using standard 1 or 2 inputs logic gates such as AND, NAND, OR, NOR, NOT and XOR.

d) Some counters have a “roll-over” output that goes HI when the counter output is 0. Such a feature is very useful because it can be employed as a clock input to another decade counter to form a decade ripple counter. How would you implement such a “roll-over” output circuit that goes HI whenever the counter is 0000? Draw the “roll-over” output circuit using standard 1 or 2 inputs logic gates such as AND, NAND, OR, NOR, NOT and XOR.
7.18) a) Use a Venn diagram to display an XOR function. Specifically show Q, where
\[ Q = A \oplus B \]
In your diagram, clearly indicate and label A, B and Q.

Show that the two Boolean Expressions below are true:

b) \[ A + \overline{AB} = A + B \]
c) \[ AB + \overline{AB} = A \oplus B \]

You will receive full points if you use Boolean logic to prove it. If you are unable to do so, you can use a Venn diagram to prove it but you will lose half a point for each proof. If you are still unable to do it with a Venn diagram then you may use a complete truth table but you will lose 1 point for each proof.

7.19) General Questions / Essay Questions:
Please answer the questions below. Read the questions carefully! Some require more than one answer!

a) You need to amplify a sensor’s small output voltage. When selecting an amplifier for the job, what should (ideally) its input impedance be? What should be its ideal output impedance?
b) What is the function of a blocking capacitor; specifically what type of (voltage) signal does it block?
c) An unwanted signal at 60 Hz must be reduced by a factor of 64. If we use a 6 dB / octave, also known as a 20 dB / decade, RC filter, to what frequency must we go to reach the factor of 64 reduction in the 60 Hz signal? (There are two answers, pick only one.)
d) Draw the type of RC filter that allows you to differentiate a signal? Would you set the filter’s f<sub>3dB</sub> frequency above, below or at the frequency of the input signal, f<sub>input</sub>, to observe its differentiated output?
e) An op-amp follower circuit exhibits unit gain. In other words, when connected to a sensor, the follower’s output signal will be identical to its input voltage. So why would you use it? Specifically, list two important benefits that the op-amp follower provides.
f) You used an instrumentation op-amp to build a strain gage circuit. The instrumentation op-amp was used because it provides a large differential gain and a large common mode rejection ratio, CMRR. How do you measure these gains? Draw the circuit for measuring the differential gain and the circuit for measuring the CMRR using an AC voltage source. (In your drawing, you may substitute the instrumentation op-amp with a difference amplifier circuit.)
g) Name at least 3 good reasons why or what you may want to use the PWM (pulse width modulation) technique for?
h) When describing a particular analog to digital converter, or digital to analog converter, aside from its operating range, which are the two most important parameters of its characteristics or performance?
i) Assume that you want to convert a voltage signal from one type of voltage to another type. What circuit, circuit element or technique would be the most suitable, i.e., simplest for the task:
- Large DC voltage to a smaller DC voltage.
- Small AC voltage to a larger AC voltage.
- Large AC voltage to a small AC voltage.
- AC voltage to DC voltage.
- DC voltage to AC voltage.
j) What simple digital logic circuit element allows us to change the frequency of a signal by a factor of 2? (Note: simple does not mean some sort of PWM techniques.)
k) Write down the Boolean expression for DeMorgan’s theorem.
l) What is a parity encoder? Specifically, can you draw one and why or where would it be used and what for?

### G.8. Solutions to Some of the Problems

#### G.1. Voltage Dividers / Thevenin Equivalent Circuits

1.1) [http://mxp.physics.umn.edu/f02/images/F02Quiz1NSol.pdf](http://mxp.physics.umn.edu/f02/images/F02Quiz1NSol.pdf)

1.2) a) \(V_{ab} = 10\, \text{V}, \quad R_{Th} = 1 \, \text{k}\)

b) Schematic of a resistor of value \(R_{Th}\) in series with a voltage source of value \(V_{ab}\)

1.3) a) \(V_{TH} = \frac{(V_1 R_1 + V_2 R_2)}{(R_1 + R_2)}; \quad R_{TH} = R_1 R_2/(R_1+R_2)\)

b) Schematic of a resistor of value \(R_{Th}\) in series with a voltage source of value \(V_{Th}\)

c) \(V_{TH} = 17.5\, \text{V}; \quad R_{TH} = 7.5\, \text{k}\Omega\)

d) \(I = 1.17\, \text{mA}\)

1.4) a) \(V_{AB} = 0\, \text{V}\)

b) \(I_1 = \frac{(V/R)}{5} \, \text{mA}\)

1.5) \(V_{AB} = -5\, \text{V}\)

1.6) \(V_{TH} = 22\, \text{V}; \quad R_{TH} = 2.91\, \Omega\)

1.7) a)-c) [http://mxp.physics.umn.edu/f98/midterm97/F97midterm3.html](http://mxp.physics.umn.edu/f98/midterm97/F97midterm3.html)

1.8) a) \(i_1 + i_2 + i_3 = 0; \quad V - 5V + i_1(20k) = 0; \quad V + i_2(40k) - 10V = 0; \quad V + i_3(40k) = 0\)

b) \(i_3 = -2.85\, \text{mA}\)

1.9) \(V_1 = i_1 R_1 + V_2 + i_2 R_2 + i_2 R_2; \quad V_2 = i_2 R_2 + i_2 R_2 + i_1 R_2 + i_2 R_4\)

1.10) \(R_{TH} = R/3; \quad V_{TH} = 2 \, \text{V} / 3; \quad \text{Schematic of a resistor of value } R_{Th}\) in series with a voltage source of value \(V_{Th}\). (Hint: use superposition, i.e., find \(V_{AB}\) first with one of the voltage sources shorted and then with the other shorted; add your answers to find \(V_{AB}\).)

1.11) a) \(10\, \text{Ohms}\)

b) \(10\, \text{Amps}\)

1.12) a) \(V_2 = (V_{in} R) / (R_1 + R); \quad V_s = V_{in} / 2\)

b) \(V_{out} = 0\)

1.13) [http://mxp.physics.umn.edu/f03/Quizzes/Final2.pdf](http://mxp.physics.umn.edu/f03/Quizzes/Final2.pdf)

1.14) a) \(V_n = a[R_1 + (R_2 R_3 / (R_2 + R_0))]\)

c) \(V = 0.5\, \mu\text{V}\)

1.15)

1.16) a) With no load: \(-3V < V_{out} <= 7V\)

b) \(R_{Th}\) can be found by shorting all V-sources; in this case \(R_{Th}\) is just \(R_{12}\) in parallel with \(R_{23}\):

\[R_{Th} = (\alpha (1 - \alpha)) \, R_0\]

c) Current through \(R_o\): \(7V - V_{out})/R_{12} = (V_{out} - (-3V))/R_{23}; \quad \text{solve for } V_{out}: V_{out} = 7(1 - \alpha) - 3\alpha\)

(\text{Check answer } \alpha = 0, V_{out} = 7V, \alpha = 1, V_{out} = -3V)

d) Since answer c) is the same as the Thevenin voltage, \(V_{Th}\), \(V_{out}\) will just be the voltage divider formed by \(R_{Th}\) and \(R_{load}\)

\[V_{out} = V_{Th} R_{load}/(R_{Th} + R_{load}) = (7(1 - \alpha) - 3\alpha) R_{load}/(R_{load} + (\alpha (1 - \alpha)) R_0)\]
G.2. RC Circuits

2.1 a)-f) http://mxp.physics.umn.edu/f03/Quizzes/F03Quiz1.pdf

2.2 a) It's a High-Pass filter or differentiator with \( V_{\text{out}} = \frac{\text{d}V_{\text{in}}}{\text{d}t} \), i.e., positive and negative spikes.

b) \( V_{\text{out}} = V_{\text{in}} \)

2.3 a)-e) http://mxp.physics.umn.edu/f02/images/F02Quiz1NSol.pdf

2.4 a)-c) http://mxp.physics.umn.edu/f02/images/F02Quiz1NSol.pdf

2.5 a) \( V_{\alpha} = V_{DC} = 155 \text{ V} \)

b) \( V_{\beta} = 38.75 \text{ V} \)

c) \( (R_1 R_{ON}/(R_1 + R_{ON})) C \left( \frac{\text{d}V_{Ne}}{\text{d}t} \right) + V_{Ne} = (R_{ON} V_{DC})/(R_1 + R_{ON}) \)

d) \( R_{Th} = 750 \text{ Ohm} \)

e) use the most general expression: \( V(t) = A + B \exp(-t/RC) \) and then find:

\( V(0) = A + B = V_{ON} \);

\( V(\infty) = V_{\beta} \);

\( V(\tau) = V_{Off} \)

\( \tau = RC \ln \left( \frac{V_{Off} - V_{\beta}}{V_{ON} - V_{\beta}} \right) \) where \( R = R_{Th} \).

f) \( \tau = 0.85 \text{ sec} \)

2.6 a)-c) http://mxp.physics.umn.edu/f98/midterm97/F97midterm3.htm

2.7 a) \( V_{AB} = V_s \frac{(2 - j\omega RC)}{(4 + \omega^2 R^2 C^2)} \)

b) \( V_{AB}(\omega \to \infty) = 0; \ V_{AB}(\omega \to 0) = \frac{V_s}{2} \)

c) \( Z_{in}(\omega \to \infty) = 0; \ Z_{in}(\omega \to 0) = \infty \)

2.8) http://mxp.physics.umn.edu/f96/F95MidtSol.html

2.9 a)

b)

c)

2.10 a)

b)

2.11 a) \( Z_2 Z_3 = Z_1 Z_4 \)

b) \( R_s = R_2 R_3 / R_1; \ L_x = R_2 R_3 C_1 \)

2.12 a)-b) http://mxp.physics.umn.edu/f04/Quizzes/F04Midterm1wSol.pdf

G.3. LRC Circuits

3.1) \( (j\omega L - \omega^2 RCL) / (R+j\omega L - \omega^2 RCL) \)

3.2 a) \( I(t) = (V_s/(3 \omega L))(1 - 2 \omega^2 C L) \ \sin \omega t \)

b) \( f = 7.9 \times 10^5 \text{ Hz} \)

3.3 a) \( Z = [j / \omega C (1 - (1 / \omega^2 L C))] + R \)

b) \( Z_{in}(0) \to 1; \ A(0 \to \omega) \to 0; \ A(\omega \to \infty) \to 1 \)

c) \( \omega = \omega_c \to H \sim 1 \to \theta(j\omega) = \pi/2 \)

3.4 a)-c) http://mxp.physics.umn.edu/f05/Quizzes/F05Q1.pdf

3.5 a) \( H(j\omega) = jL\omega / [(\omega L + R(1-LC\omega^2))] \)

b) \( \omega \to 0; \ H \sim 0; \omega \to \infty; \ H \sim 0 \) It is a band-pass filter. \( H = 1 \) when \( (1-LC\omega^2) = 0 \to \omega_c = 1/(LC)^{1/2} \)

c) \( \omega \to 0; \ H \sim jL\omega / R \to \theta(j\omega) = \pi/2 ; \omega \to \infty; \ H \sim j/(RC\omega) \to \theta(j\omega) = -\pi/2 \)

G.4. Diodes and Transformers

4.1) http://mxp.physics.umn.edu/f02/images/F02Quiz1NSol.pdf

4.2) http://mxp.physics.umn.edu/f03/Quizzes/F03Quiz1.pdf

4.3 a) If \( V_{in} < 0 \) then the diode is off, \( V_{out} \) is a simple voltage divider with \( V_{out} = V_{in}/2 \);

b) If \( V_{in} > 0 \) then the diode is on, \( V_{out} = V_{in} \);

If \( V_{in} < 0 \) then the diode is off, \( V_{out} \) is a simple voltage divider with \( V_{out} = V_{in}/2 \);

b) If \( V_{in} > 0 \) then the diode is on, \( V_{out} = V_{in} - 0.7 \text{V} \)

4.4 a) http://mxp.physics.umn.edu/f00/F99MidTerm1Sol.htm#Problem%202:

b) \( I_{1\text{max}} = 3 \text{ mA}; \ I_{2\text{max}} = I_{3\text{max}} = 2 \text{ mA} \)
4.5) http://mxp.physics.umn.edu/f98/midterm97/F97midterm3.htm
4.6) a) $N_1 / N_2 = 2 / 1$
   b) $N_1$ or primary
   c) $V_1 / V_2 = 2$
   d) $I_1 / I_2 = 1 / 2$
4.7) $V_A = 1 \text{mA} \times 10^3 - 10 = 0 \text{Volt}$

G.5. Transistors
5.1) a)-b) http://mxp.physics.umn.edu/f03/Quizzes/F03Quiz1.pdf
5.2) a)-c) http://mxp.physics.umn.edu/f02/images/F02Quiz1NSol.pdf
5.3) a) $V_B = 3.1 \text{V}; V_E = 2.5 \text{V}; I_E = 2.5 \text{mA}$
   b) $I_C \approx I_E = 2.5 \text{mA}; I_B = 25 \mu \text{A}$
   c) $G = (V_{out} - AC) / V_S = 1$; current amp not voltage amp.
5.4) a)-d) http://mxp.physics.umn.edu/f04/Quizzes/F04Midterm1wSol.pdf
5.5) a) $V_{E1} = V_{C2} = 1.9 \text{V}, V_{B1} = 2.4 \text{V}, V_{C1} = 6 \text{V}, V_{E2} = 1 \text{V}, V_{B2} = 1.5 \text{V}$
   b) $I_{B1} \approx I_{E} = 2.5 \text{mA}; I_B = 25 \mu \text{A}$
   c) $I_C = I_E / (\beta + 1) = 2 \text{mA}$
   d) $Z_{in} = 0.4 \text{k}$
   e) Protects the collector of transistor 1 when the voltage suddenly drops across the coil.
5.6) a) $R_1 = 8 \text{k}; R_2 = 2 \text{k}$
   b) $V_B = 4.0 \text{V}; V_E = V_B - 0.6 = 3.4 \text{V}; I_E = V_E / 3.4 = 1.0 \text{mA} = I_C; V_C = 20 - I_C; 10 \text{k} = 10 \text{V}$
   c) $V_{out} = 1; Out_2 = 3$
   d) $Z_{in} = 1.6 \text{k}$
   e) $C = 10 \mu \text{F}$
5.7) a)-f) http://mxp.physics.umn.edu/f96/F95MidtSol.html
5.8) a) $I_C = \beta I_B; I_E = I_B + I_C; I_D(\beta + 1) = I_E$
   b) $I_B = 11.5 \mu \text{A}; I_C = 1.15 \text{mA}; I_E = 1.15 \text{mA}; V_x = 5.53 \text{V}$
   d) Active mode
   e) $V = 4.76 \text{mV}$; phase = $180^\circ$
5.9) a) $V_B = 2 \text{V}$
   b) $V_E = 1.4 \text{V}$
   c) $I_E = 10 \text{mA}$
   d) $I_E = I_C = 10 \text{mA}$
5.10) a) $R_{C2} = 1.5 \text{k}$
   b) $R_e = 150 \Omega$
   c) $V = 1.35 \text{V}$
   d) $R_1 = 20 \text{k}; R_2 = 1.98 \text{k}$
   e) $1.62 \text{k}$
5.11) a) $V_B = 3 \text{V}_{Diode\ On} = 1.8 \text{V}$
   b) $V_E = V_B - V_{Diode\ On} = 1.2 \text{V}$
   c) $I_C = I_E = V_E / 0.6 \text{Ohm} = 2 \text{mA}$
   d) $I_B = I_E/\beta = 0.02 \text{mA}$
   e) $I_D = (12 \text{V} - 1.8 \text{V}) / 3.14 \text{kOhm}$
   f) no effect since IC is approximately equal to $I_E$
5.12) a)-d) http://mxp.physics.umn.edu/f05/Quizzes/F05Q1.pdf

G.6. Op-Amps
6.1) b) $V_+ = -3 \text{V}$
   c) if $V_{in} > V_+$ then $V_{out} = -15 \text{V};$ if $V_{in} < V_+$ then $V_{out} = +15 \text{V}$.
6.2) a) Positive Feedback - Golden Rules do not apply here!
   b) if $V_+ > V_-$ then $V_{out} = +Vcc$; if $V_+ < V_-$ then $V_{out} = -Vcc$
   c) $V_+ = 3 \text{VDC}$
   d-e) http://mxp.physics.umn.edu/f00/F99MidTerm1Sol.htm#Problem%203:
6.3) a) b) Comparator circuit with $V_{in}$ into $V_+$ and $V_-$ at $+5 \text{V}$.
6.4) a) positive feedback
   b) $V_{out} = +V_{cc}$ for $V_s > 0$; $V_{out} = -V_{cc}$ for $V_s < 0$;
   c) $V_s = (V_oR_2 + V_{out}R_1) / (R_1 + R_2)$
   d) $V_{THL} = +V_{cc} (R_1 / R_2) = -V_{cc} (R_1 / R_2)$
   e) Hysteresis with Vout switching at Vin +/- 3 V and for Vin = Vcc, Vout = -Vcc; similarly for Vin = -Vcc, Vout = Vcc.

6.5) a) F00
   b) $V_I = (V_{out}R_1) / (R_1 + R_2)$
   c) $V_s(t) = A + B \exp(-t/R C)$
   d) $T = 2R_C = C \ln[(1 + \beta) / (1 + \beta)]$

6.6) a)-b) [link]
   a)-c) [link]
   a)-c) [link]

6.7) a)-b) [link]
   a)-c) [link]

6.8) a)-c) [link]

6.9) a) -10V
   b) [link]

6.10) a) F00
   b) $V_{out} = (kT/q) \ln[(V_{ref} / I_bR) + 1] \approx (kT/q) \ln[V_{in} / (I_bR)]$

6.11) a) The op-amp inputs drain no current, and the V- input is at virtual ground. The Bias current $I_b$ and the photocurrent, $I_{rev}$, must therefore flow through D.
   b) $I_{rev} = I_b$; $V_{threshold} = (1/\alpha)$
   c) $V = (1/\alpha) \ln(I_b/I_o)$ “logarithmic amplifier”
   d) 0.012V
   e) 2.5 E-10 W

6.12) a)-b) [link]

6.13) $V_{out} = (kT/q) \ln[(V_{ref} / I_bR) + 1] \approx (kT/q) \ln[V_{in} / (I_bR)]$

6.14) a) The op-amp inputs drain no current, and the V- input is at virtual ground. The Bias current $I_b$ and the photocurrent, $I_{rev}$, must therefore flow through D.
   b) $I_{rev} = I_b$; $V_{threshold} = (1/\alpha)$
   c) $V = (1/\alpha) \ln(I_b/I_o)$ “logarithmic amplifier”
   d) 0.012V
   e) 2.5 E-10 W

6.15) a)-b) [link]

6.16) a) $V_+ = V_2 / 2$
   b) $V_{out} = (V_2 / 2) (1 + R_3 / R_1) - V_1 R_3 / R_1$
   c) For max. CMRR we want Vout to be strictly a function of $V_2 - V_1$. Therefore, $R_1 = R_3$ will give $V_{out} = V_2 - V_1$.
   d) $V_{out} = 2 V_{offset} + V_2 - V_1$

6.17) a) negative
   b) $V_{out} = -(R_2/R_1) V_1$ inverting
   c) $V_s = (R_2 / (R_1 + R_2)) V_o$; $V_o = (R_2 / (R_1 + R_2)) V_0$;
   d) $V_o = (R_2/R_1) V_o$
   e) $V_o = (R_2/R_1) (V_o - V_a)$
   f) [link]

6.18) a) [link]

6.19) a) Inverter with $R_{in} = 20$ k; $R_{feedback} = 200k$
   b) Non-inverting amplifier with $R_1 = 500$ Ohm and $R_{feedback} = 9.5k$
   c) $V_o = (kT/q) \ln(V) + \text{const.}; \text{const.} < 0$
   d) [link]

6.20) a) $G = -10 < 10^5$ Hz
   b) 1k
   c) F93
   d) $V = (V_{ref} / 10^5)$

6.21) a)-c) [link]

6.22) Fig 6.21) $V_{out} = V_{in}$.
   Fig 6.22) Comparator with $V_{in} = 0.5V$, i.e.; if $V_{in} > .5V$, $V_{out} = -15V$; if $V_{in} < .5V$, $V_{out} = +15V$.
   Fig 6.23) $V_{out} = (1 / 2 \omega R C f) \cos \omega t$, i.e., $-1 / \omega \Omega = Vout <= 1/\Omega$.
   Fig 6.24) $Vout = -2.2 \sin \omega t$
   Fig 6.25) $Vout = -10 \sin \omega t$

6.23) a)-f) [link]

6.24) [link]

6.25) a) $V_{out} = -R_f (V_o/R_o) + (V_f/R_f) + (V_2/R_2)$
   b) $x = 1$; $V_{out} = -0.1V = (1k)(5V) / R_o$; $R_o = 50k$
Selected Problems / G.8. Solutions to Some of the Problems

\[ x = 2; V_{out} = -0.2V = \frac{(1k)(5V)}{R_1}; R_1 = 25k \]
\[ x = 1; V_{out} = -0.4V = \frac{(1k)(5V)}{R_2}; R_2 = 12.5k \]

6.26) a-b) [link to file]
6.27) a-c) [link to file]
6.28) a) 100k
   b) \(10^5\) Ohms; no
   c) 100k
   d)
   e)

6.29)
6.30) Figure 4a: \(V_{out} = +12V\) (it's a comparator, \(\Delta Vin > 0\))
   Figure 4b: \(V_{out} = +12V\) (it's a comparator, \(\Delta Vin > 0\))
   Figure 4c: \(V_{out} = +9V\) (it's a non-inverting amp with gain 1+2/1)
   Figure 4d: \(V_{out} = -12V\) (it's a positive feedback, comparator with hysteresis; the switching voltage is \(=+/-(12/11)V\). Since the input exceeds both of these, and since \(\Delta Vin < 0\), \(Vout = -12 V\))
   Figure 4e: \(V_{out} = -12V\) (it's an inverter that has been "saturated.")

G.8. Digital/Logic Circuits

7.1) a-e) [link to file]
7.2) a-c) [link to file]
7.3) a-c) [link to file]
7.4) a)
   b)
7.5) 1010, 1101, 1110, 0111..., i.e., 10, 13, 14, 7... etc.
7.6)
7.7) \(Q_0 = !A_0 !A_1 !EN; Q_1 = A_0 !A_1 !EN, etc.\)
7.8) a & b) \(Y = !(!(A !(A+B)) !(B !(A+B))) = !(A+B)\)
   c)
7.9) a)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

b) \(Q = AB + AC + BC\)
   c,
   d) \(D_0 = D_1 = D_2 = D_4 = 0; D_3 = D_5 = D_6 = D_7 = 1;\)
7.10) a)

<table>
<thead>
<tr>
<th>A2</th>
<th>A1</th>
<th>A0</th>
<th>Input</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>D0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>D1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>D2</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>D3</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>D4</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>D5</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>D6</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>D7</td>
<td>1</td>
</tr>
</tbody>
</table>

b) \(Q = A_0 (A_1 + A_2)\)
7.11)  
7.12) a) 7 A 5 F  
   b) 31327  
   c) 0011 0001 0011 0010 0111  
   d) 1000 0101 1010 0001  
   e) 3E9 0011 1110 1001  
   f) 10.011  
7.13)  
7.14)  
7.15) a)-b) http://mxp.physics.umn.edu/f05/Quizzes/F05quiz2.pdf  
7.16) a) 

<table>
<thead>
<tr>
<th>INPUTS</th>
<th>OUTPUTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>n3</td>
<td>n2</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

b) $E = \overline{n_3 n_2 n_1 n_0} + \overline{n_3 n_2 n_1 n_0} + \overline{n_3 n_2 n_1 n_0} + \overline{n_3 n_2 n_1 n_0}$

7.17 a)  

<table>
<thead>
<tr>
<th>Dec. Value</th>
<th>Q3</th>
<th>Q2</th>
<th>Q1</th>
<th>Q0</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

b) $D_3 = \overline{Q_3 Q_2 Q_1 Q_0} + \overline{Q_3 Q_2 Q_1 Q_0}$  
c)
d)  

\[ \text{Roll} - \text{Over}_{\text{out}} = \overline{Q_3} \overline{Q_2} \overline{Q_1} \overline{Q_0} \]

7.18) a)  

\[ A \oplus B = \overline{A} \overline{B} + \overline{A} B, \]  

then shaded area is XOR.

\[ A + \overline{A} B = A \cdot 1 + \overline{A} B = A \cdot (1 + B) + \overline{A} B = A + \overline{A} B + \overline{A} B = A + B(A + \overline{A}) = A + B \cdot 1 = A + B \]

b)  

\[ AB + \overline{A} \overline{B} = \overline{A} B + \overline{A} \overline{B} = \overline{A} \overline{B} \cdot AB = (\overline{A} + B) \cdot (A + \overline{B}) = \overline{AB} + \overline{A} B + \overline{B} A + \overline{B} B = \\
\overline{A} B + \overline{A} \overline{B} = (\overline{A} B + \overline{A} \overline{B}) = A \oplus B \]

c)  

7.19)  
a) Input impedance: large; output impedance: small  
b) DC voltage  
c) 3840 Hz or .94 Hz  
d) Highpass Filter with \( f_{\text{dB}} \gg f_{\text{signal}} \)  
e) impedance transformation, i.e., will provide large input impedance and small output impedance.  
f) in differential mode, signal source is connected to one input and the other is grounded; in CMRR measurement, both op-amp inputs are tied together and to the signal source.
g) energy saving, DC-DC conversion, A2D converter, DC to AC conversion, can provide large power with suitable switch and voltage source, voltage to frequency conversion.

h) resolution and conversion speed.
   i) Large DC voltage to a smaller DC voltage. A: resistive voltage divider
   ii) Small AC voltage to a larger AC voltage. A: transformer
   iii) Large AC voltage to a small AC voltage. A: transformer or resistive voltage divider
   iv) AC voltage to DC voltage. A: rectifier
   v) DC voltage to AC voltage. A: PWM technique

j) Flip Flop

k) $(A+B)! = A! B!$ (You could also state $(AB)! = A! + B!$, either is ok.)

l) it used for error detection in (serial) data transmissions; can be constructed by daisy chaining XOR gates together.
Introduction

The notes here are intended to guide you in analyzing general op-amp circuits. Hopefully they will clear up some confusion about one of the biggest misconceptions in understanding op-amps, namely the application of the golden rules from Horowitz & Hill to all op-amp circuits. It cannot be emphasized enough that the golden rule only applies in certain specific circumstances and applying it to the wrong op-amp circuit often enhances misconceptions about op-amps. On the other hand, the “general rules” below are true for any op-amp circuit.

KW 2006

H.1. General Rules

The following two conditions define the rules that follow:

First, an amplifier “amplifies” the difference between its input voltages, $\Delta V_{in}$:

1: $V_{out} = A \Delta V_{in}$ where $\Delta V_{in} = V_+ - V_-$.

“$A$” is the “open loop gain” of the amplifier and for an ideal op amp is infinity and for a typical op amp is between $10^6$ to $10^8$.

Second since we are dealing with an actual physical device its output voltages can never exceed its supply voltages, $V_-$ and $V_+$.

2: $V_- < V_{out} < V_+$

Typical supply voltages are $V_- = -15V$ and $V_+ = +15V$.

Applying the previous rules leads to 3 new cases:

2a: If $A \Delta V_{in} > V_+$ then $V_{out} = V_+$

2b: If $A \Delta V_{in} < V_-$ then $V_{out} = V_-$

2c: If $V_- < A \Delta V_{in} < V_+$ then $V_{out} = A \Delta V_{in}$
Conditions a) and b) merely restate rule 2 above, namely that if the predicted gain of the op-amp exceeds the supply voltages, it will “max out” (or “saturate”) at the supply voltages.

Finally, op-amps approach nearly ideal input and output impedance conditions. For completeness, the following rules are added:

<table>
<thead>
<tr>
<th>3a</th>
<th>$Z_{out} = 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3b</td>
<td>$Z_{in-} = \infty; Z_{in+} = \infty.$</td>
</tr>
</tbody>
</table>

From 3b, it follows that:

| 3c | $I_{in-} = 0; I_{in+} = 0.$ |

For most applications, these assumptions hold since typical input currents are $10^{-9}$ to $10^{-12}$ Amps and the input impedances are in the $10^6$ to $10^9 \Omega$ range.

**H.2. Feedback**

*The most important step in analyzing an op-amp circuit is to determine the type of feedback that is being employed!*

Feedback refers to connecting the output of the op-amp to its input, usually through resistors. There are three basic ways to do that, shown below:

- **No Feedback**
- **Positive Feedback**
- **Negative Feedback**

*Figure H.1. All three feedback methods.*

Determining which feedback method is used will tell us in what function the op-amp is being used.
H.3. No Feedback: Comparator

If no feedback is employed and applying the general rules for an ideal op-amp with \( A = \infty \) shows that only cases 2a) and 2b) need to be considered, i.e.,

\[
\begin{align*}
\text{If } \Delta V_{\text{in}} > 0, & \quad V_{\text{out}} = V_{+} \\
\text{If } \Delta V_{\text{in}} < 0, & \quad V_{\text{out}} = V_{-}
\end{align*}
\]

In other words, an ideal op-amp without any feedback will always saturate either at its positive or negative supply voltage! Its output will change its state when \( \Delta V_{\text{in}} \) changes its sign.

- Figure H.2: \( V_{\text{out}} \) vs. \( \Delta V_{\text{in}} \). Note: the output switches at \( \Delta V_{\text{in}} = 0 \) and otherwise remains at either \( V_{+} \) or \( V_{-} \).

This type of feedback, or rather the lack thereof, is used in comparators circuits where one is interested only in discreet voltages, i.e., outputs that take on one of two states, i.e., \( V_{+} \) or \( V_{-} \).

While the first circuit in Figure A switches its output each time \( V_{\text{in}} \) crosses 0 volt, a small modification, shown below, allows the circuit to switch its output when \( V_{\text{in}} \) crosses a certain preset voltage level, often called the threshold voltage, \( V_{\text{th}} \).

- Figure H.3: Op-Amp comparator circuit with threshold voltage. Note the new switching point at \( V_{\text{th}} \).
Typical applications of this circuit are crossover detectors, analog to digital converters or counting applications where one wants to count pulses that exceed a certain voltage level.

**Comparator Analogy**

A different way of looking at this type of circuit is to think of it in a gravitational analog which involves balls dropped on a hill shaped potential (see below.) Assume that the x-axis position corresponds to a voltage level: the position from which ball is dropped represents $V_-$ and the center of the hill is $V_+; V_{out}$ corresponds to final state of the ball along the x-axis. Depending on the two input variables, $V_-$ and $V_+$, the ball will always end up either all the way to the left or right of the center of the hill, i.e., the system is said to be **bistable**.

![Figure H.4. Gravitational analogy of a comparator circuit.](image)

So far only comparator circuits with ideal op-amps have been considered. Luckily, for all practical purposes non-ideal (but real) op-amps behave likewise. Given the typical open loop gain of $10^6$ to $10^8$ shows that only when $-15 \mu V < V_{in} < +15 \mu V$, case 2c) would have to be considered, i.e., the op-amp is no longer saturated and it behaves like a "real" amplifier. Since this range is so small, for all practical purposes it can, or should, be ignored.

**Exercises:**

Draw $V_{out}$ vs. $V_{in}$ if $V_{in}$ was connected to $V_-$ and $V_{in}$ to $V_+$, i.e., the reverse of the previous example.

Design an op-amp circuit that produces TTL signals, i.e., $V_{out}$ is either 0 or 5V, for a given input signal within that range. Make the switching threshold voltage adjustable.

**H.4. Positive Feedback: Comparator with Hysteresis, Oscillators**

The middle circuit in Figure H.1 has positive feedback yet it still acts like a comparator but, in addition, it now has memory, i.e., it exhibits hysteresis. Though positive feedback is generally associated with oscillation, this circuit is not an oscillator because the input to the circuit ($V_-$) is independent of the output. To create a real oscillator, the input to the circuit must also be in some
way connected to the output as, for example, in the RC relaxation oscillator circuit shown in the lab manual.

To understand what causes the hysteresis let’s analyze the circuit in Figure H.5 using the same rules as in the previous section for the comparator. (Though at this point we have no proof that it will behave exactly like a comparator but our results will ultimately show that these assumptions were indeed consistent and hence, correct.) The key in understanding this circuit will again be in calculating the voltages that cause its output to switch.

The output will still be $V_+$ or $V_-$, depending on whether $\Delta V > 0$ or $\Delta V < 0$. Switching of the output will occur when the value $\Delta V$ changes sign, i.e., at $\Delta V = 0$.

![Figure H.5: Positive Feedback](image)

For the particular circuit above, $\Delta V = V_+$ since $V_- = 0$. So,

- if $V_+ > 0$, $V_{out} = V_+$,
- if $V_+ < 0$, $V_{out} = V_-$.

Since $V_{out}$ changes its state whenever $V_+$ crosses $0V$, we need to calculate the value of $V_{in}$ that results in $V_+ = 0$. Since $V_+$ is a voltage divider formed by $R_{1}$ and $R_{2}$ between $V_{in}$ and $V_{out}$ it follows that:

$V_+ = (V_{in} R_2 + V_{out} R_{1}) / (R_{1} + R_{2})$

Combining these equations and solving for $V_+ = 0$ yields $V_{in} = -V_{out} R_2 R_1 / R_{2}$. Since $V_{out}$ can only be either $V_+$ or $V_-$. $V_{out}$ switches from $V_+$ to $V_-$ when $V_{in} = -V_+ R_1 R_{2} / R_{2}$ and it switches from $V_-$ to $V_+$ when $V_{in} = -V_- R_1 / R_{2}$. In other words, it always takes “more effort” to reverse the output than was initially required.
Gravitational Analogy

If this approach was too mathematical, consider a gravitational analogy again. Assume that $V_{in}$ corresponds to the x-position along the x-axis of balls being dropped on a hill potential from which they either roll down on the left or the right side. While in the previous example the mountain (and its peak) where rigid, assume that it itself can move along the x-axis and remain in one of two positions: if a ball previously rolled down its right side, it will be pushed all the way to the left, see figure 3 and remain there till a ball rolls down its left side at which point it will move all the way to the right as shown in Figure 6. From this analogy you can see that it takes some extra effort to reverse the output state.

With some small modifications to the above circuit, positive feedback circuits can also be used for oscillator circuits.


Only when negative feedback is applied to the op-amp it (may) act(s) as a linear amplifier. To be precise, it can only act as a linear amplifier as long as its output is not saturated, i.e., if it does not exceed its supply voltages.
Nevertheless, when these two conditions are met, (negative feedback AND not saturated) a new rule, called the “Golden Rule of Op-Amps” can be applied. It says:

4: \( V_+ \equiv V_- \)

As already mentioned, be careful when applying this rule that both conditions stated above are met!

The Golden Rule of Op-amps makes the analysis of negative feedback circuits straightforward:

1. Remove the op-amp altogether from your circuit
2. State that \( V_+ = V_- \)
3. Solve.

![Figure H.8: The circuit shown is the negative feedback circuit from Figure H.1. Step 1 is shown in the middle picture and step 2 is at right.](image)

After applying steps 1 and 2, all that is usually left for the analysis is some sort of resistive voltage divider, as shown in Figure H.8 at the right. Noting that the current through \( R_1 \) must be identical to the current through \( R_2 \) leads as immediately to the application of Ohm’s law, namely:

- \( (V_{in} - V_-)/R_1 = (V_- - V_{out})/R_2 \)
  Since \( V_- = 0 \)
- \( V_{out} = (-R_2/R_1) \cdot V_{in} \)

Before applying the golden rule blindly, even where it applies, one should ask where does it come from and does it make sense?

To start with the second question, consider once more the inverting amplifier solved previously. We found by applying the golden rules that \( V_{out} = (-R_2/R_1) \cdot V_{in} \). On the other hand, when applying these rules to obtain this result, they were used to draw the conclusion that \( V_+ = V_- = 0 \, \text{V} \). Since \( \Delta V = V_+ - V_- \) it follows then that for the above case \( \Delta V = 0 \). Applying now our primary rule, namely that \( V_{out} = A \Delta V \), results in \( V_{out} = 0 \) which is in direct contradiction to the result obtained by applying the golden rules of op-amps. So which calculation is right?

Since we stated that rule #1, \( V_{out} = A \Delta V \) always holds, (at least if \( V_{out} \) is not in saturation) we should really apply it to solve for \( V_{out} \). Also for now assume that \( A \) is finite. Solving the circuit in Figure G using only rule #1, (and NOT applying the Golden Rules of Op-Amps) one would find: (please check for yourself)
We see that when $A \gg R_2/R_1$ we arrive at the same answer that was previously obtained by applying the Golden Rules. In other words, the Golden Rule assumes $A$ being infinite, or at least much larger than the negative feedback gain of the circuit. The negative feedback is usually called the "closed loop gain."

The infinite value of $A$ also explains the contradiction we arrived at previously. Solving Rule #1 for $\Delta V$, we find that $\Delta V = V_{out}/A$. If we assume that $A$ is infinite, well then the $\Delta V$ also will approach 0 V. In other words, we are dealing now with the singularities of infinity and 0. Using "realistic" values for $A = 10^8$, we find that $\Delta V = 10^{-8} V_{out}$ under any circumstances, a very, very small signal but not quite 0!
J.1. Creating a New Project

1a. Either, on your desktop, double-click on the icon shown on the right::
1b. or from the Start Menu, open “All Programs” → “Xilinx ISE Design Suite 12.1” →
“ISE Design Tools” → “64-bit Project Navigator.”

2. When the Project Navigator opens, hit “OK” to get clear the “Tip of the Day” box
and on the left side, select “New Project” from the “File” menu.

3. The “New Project Wizard” opens and requests a name for the project and a location to store the
files.
WARNING: by default the program wants to store your data on the C: drive. This will not work and your files will not be available on any other computer! Therefore, click on the browse button (i.e., the three dots) at the far right of “Location” and create a new folder under your “My Documents” folder. (See the picture above on the right.) The name of this folder is your choice.

4. Next specify the type of FPGA board you are working with. Specifically, enter the values shown below for: Family, Device, Package and Speed.

Click “Next” and a project summary window, shown on the left below, will open:
Hit “Finish” and you should see the project window, shown above on the right.

**J.2. Creating a New Source**

1. To create a new Verilog source file, right click on the “xc3s1004-4tq144” icon in the Hierarchy window (see below); select “New Source.”

The window shown below opens.
2. Select “Verilog Module” as the type of source to create and enter a file name (which will also be used as the module name). Do not put spaces in the filename. Press “Next” to continue.

3. If you wish, you can enter the input and output ports in the “Define Module” dialog. You can always add, remove, or adjust ports afterwards by editing the Verilog file directly. Press “Next >” to continue and “Finish >” to complete the process and you should see a display similar to the one shown in the picture below providing you with a design summary.

4. The previous process generated a “skeleton” source file. To see it click on the tab with the Verilog module (it has an “.v” file extension.) See the picture below.
You should now see the source code window on the right. Do not worry if you do not (yet) understand most of the statements.

Since we have not yet specified what exactly the project should accomplish, only an "outline" or "skeleton" source code has been generated to which you then will add the appropriate Verilog statements so that it will do whatever you intend it to do, hopefully "something" useful.

Specifically, note the two commands "module testGate()" and "endmodule" which in the file above are to be found on lines 21 to 25: it is between these two statements where you will place most of your own code (at least for now) and which determines how the FPGA behaves!

J.3. Synthesis and Downloading of a Project

The synthesis process takes awhile, so please be patient! During this very involved process, the software first checks your code for mistakes; next, it finds a way to implement the purpose of your code with the appropriate LUT and muxs and it determines ways to "route" your signal from the various input and output pins in and out of the various FPGA cells. After optimizing it, it will create a "map" or "bit" file which is specific to the particular FPGA employed. Finally, you can then "transmit" or upload this bit file into the FPGA where it will remain until it is overwritten again or the board is turned off or has been reset. Therefore, do not forget to save your code file before you start the synthesis process. System crashes do (sometimes) occur!

1. **Important**: Before beginning this process, you must check the "Programming File" properties.

In the Sources window, shown below, verify that the Top Most module has been selected. (The Top Most module is the one with the "three boxes" icon.) Make sure it is highlighted.
Directly below the “Hierarchy” window, in the “Processes” window, right-click on “Generate Programming File” and choose “Process Properties”.

(Note: if the “Generate Programming File” option is not listed, then you probably have not selected the Top Module as described earlier.)

2. The window shown above opens; choose “Startup Options” and set “FPGA StartUp Clock” to “JTAG Clock”. Click “OK” to save. These two steps will only need to be done once per project.
3. Double-click on “Generate Programming File” to start the synthesis process. If there are any errors (such as syntax errors or “typos”), they will appear in the box at the bottom of the screen. Synthesis, implementation, and generation of the programming file may take some time (a few minutes).

Once it completes successfully, a Xilinx Webtalk dialog window may open. This is way for the manufacturer to collect information on how their products are used. You may decline participating in this by clicking the “Decline” option.

If everything worked ok, you should see the following message in the status window, at the bottom of the Xilinx Navigator: “Process Generate Programming File completed successfully.” Congratulations, you are almost done!

4. Before you can transfer the “firmware” or bit file generated by the compiler to the BASYS board, connect the USB cable from the BASYS board to a USB port on the computer – the red “power” LED on the BASYS board should be on. To transfer the file, from the Start menu, open “All Programs” → “Digilent” → “Adept” → “Adept”. You should see the dialog box shown below.
5. Click on the “Browse…” button to locate your “.bit” file in your Xilinx project folder. The bit file was created when you generated the programming file in Step 3. After selecting it, press the “Program” button to download your program!

Digilent’s Adept program will remember your recently used files, which can be very convenient.

**J.4. Uploading Program Code into Volatile or Non-Volatile Memory**

The previous instructions explained how to upload your bit file to the BASYS board’s volatile memory, i.e., to its RAM (Random Access Memory.) However, this implies that as soon as the board is powered off, or reset, the contents of the bit file will be lost. (Of course, the code still exists in your computer account and can be downloaded again through the Adept facility.) This approach is fine as long as we are learning to program and we do not care to store the programs indefinitely on the boards.

Nevertheless, once you have mastered the basics of Verilog you may want to write code and have it stored in the board’s non-volatile ROM (Read Only Memory.) It will reside there indefinitely or until is overwritten by new code. Each time the board is powered up (or reset) it will remember the stored code in the ROM and will begin executing it immediately.

The hardware setting that controls the RAM vs. ROM mode is controlled by the BASYS board’s ROM / JTAG jumper. If you want to store the code in RAM set the jumper to “JTAG;” if you want to store the code indefinitely in the ROM, set it to “ROM.”

In addition, you must also compile your code for the appropriate setting. Specifically, you must identify the appropriate “Startup Clock.” In the previous section, step 2, you were told to set the “Startup Options” for the “FPGA StartUp Clock” to “JTAG Clock.” This is the correct setting for
storing your code in RAM. However, if you want to store it in ROM you must recompile your code with following setting: in the “Startup Options” set the “FPGA StartUp Clock” to “CCLK.” See the table below for a summary.

<table>
<thead>
<tr>
<th>BASYS Board Jumper Setting</th>
<th>Volatile (RAM)</th>
<th>Non-Volatile (ROM)</th>
</tr>
</thead>
<tbody>
<tr>
<td>JTAG</td>
<td>JTAG Clock</td>
<td>ROM</td>
</tr>
<tr>
<td>Startup Options / FPGA Startup Clock</td>
<td>JTAG Clock</td>
<td>CCLK</td>
</tr>
</tbody>
</table>

### J.5. Books on Verilog

Here are some books that provide a more detailed description of the Verilog language:


K.1 Structure of a Verilog Module

```verilog
module moduleName(a,b,c,d);
    input a;
    input b;
    input c;
    output d;
    // This is a comment
    wire e; // a wire is the result of combinatorial logic
    reg f; // a reg is a register (a D-flipflop)
        wire [4:0] num0; // this wire is a bus
        reg [3:0] num1; // this register is a bus
endmodule
```

K.2 Operators and Constructs

Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND</td>
<td>&amp;</td>
</tr>
<tr>
<td>OR</td>
<td></td>
</tr>
<tr>
<td>NOT</td>
<td>! or ~</td>
</tr>
<tr>
<td>XOR</td>
<td>^</td>
</tr>
<tr>
<td>Is equal</td>
<td>==</td>
</tr>
<tr>
<td>Is not equal</td>
<td>!=</td>
</tr>
</tbody>
</table>

Combinatorial Logic Assignment

Simple Assignment

```verilog
assign d = a | b;
```

Conditional Assignment

```verilog
assign d = ( c ) ? (a) : (b);
```
In this example, the wire \( d \) would have the value of \( a \) when \( c \) was true, and it would have the value of \( b \) when \( c \) was false.

**Sequential Logic Assignment**

With sequential logic (logic involving one or more flip-flops), we must specify both the actual logic behavior and the clock which is used for the flip-flops. This is done with an `always` block as seen in the examples below.

**Simple Clocked Assignment**

```
module dflop(clk, a, q)
    input clk;
    input a;
    output reg q;

    always @(posedge clk) begin
        q<=a;
    end
endmodule
```

This example represents a single D flip-flop with an input \( a \) and output \( q \) with the clock signal \( clk \). The name of the clock signal must be given inside "always @( ...)" block. The term "posedge" means that the flip-flop will transfer the value of \( a \) to \( q \) when the clock goes from low to high (a positive edge). Sometimes it may be useful to use "negedge" instead, which transfers the value when the clock goes from high to low.

**Conditional Clocked Assignment**

```
module counterEnabled(clk, enable, reset, count)
    input clk;
    input enable;
    input reset;
    output reg [3:0] count;

    always @(posedge clk) begin
        if (reset) count<=0;
        else if (enable) count<=count+1;
        else count<=count;
    end
endmodule
```

This example represents a four-bit counter which can be enabled or reset. At each rising clock edge, if \( reset \) is high then the counter is set to zero. If \( reset \) is low but \( enable \) is high, then the counter will increment. If both \( reset \) and \( enable \) are low, then the counter will hold its value and neither increase nor decrease. This example shows the use of the `if/else` construct in Verilog.
Asynchronous Set or Reset

```verilog
module withReset(clk, a, reset, q)
    input clk;
    input a;
    input reset;
    output reg q;

    always @(posedge clk or posedge reset) begin
        if (reset) q<=0;
        else q<=a;
    end
endmodule
```

This design provides a D-flipflop with an asynchronous positive-high reset.

K.3 Pin assignment

Assignment of pins is not strictly a function of Verilog itself, so the process for doing so is to add comments into the code with a specific format.

To assign a single port signal to an input or output pin: (In this case, the input signal “a” to pin 30)

(*) LOC = "P30" *) input a;

To assign all the lines of a bus signal at once:

(*) LOC = “P34 P45 P84 P100" *) output [3:0] b;
This assigns b[3] to P34, b[2] to P45, b[1] to P84 and b[0] to P100.

(Note: in some older programs you may come across the “synthesis attribute loc” statement to assign pins. Use the (*) LOC = *) statement shown above instead; nevertheless, just for reference, the syntax for the “synthesis attribute loc” is shown below:

To assign a single port signal to a pin: (In this case, the signal “a” to pin 30)

// synthesis attribute loc of a is P30

To assign all the lines of a bus signal at once:

// synthesis attribute loc of b is “P34 P45 P84 P100”
This assigns b[3] to P34, b[2] to P45, b[1] to P84 and b[0] to P100.)

K.4 Using other modules

For more complicated projects, it is often helpful to divide a project into several modules and connect them together. This is also helpful when sharing code with others. Verilog is designed to support this functionality.

Consider a project where one wishes to use the module below which converts a four-bit number to drive a “7-segment” display and connect it to a simple counter.

```verilog
module display(clk, value, leds)
    input clk;
    input [3:0] value;
```
output [6:0] leds;

... // implementation is complex!
endmodule

Code Example: display.v

module counterDisplay(ledClk, counterClk, ledSet)
    input ledClk;
    input counterClk;
    output [6:0] ledSet;

    reg [3:0] theCount;

always @(posedge counterClk) begin
    theCount<=theCount+1
end

display myDisplay(.clk(ledClk), .value(theCount), .leds(ledSet));
endmodule

Code Example: counterDisplay.v

The main module in this example is `counterDisplay`, which creates an instance of the `display` module which is called `myDisplay`. Each connection between a signal within `myDisplay` and `counterDisplay` is made explicitly. The `clk` signal in `myDisplay` is provided by the `ledClk` signal from `counterDisplay`, the `value` signal is provided by the `theCount`, and the `leds` signal from `myDisplay` becomes the output `ledSet` from `counterDisplay`. Pin assignment occurs only within the main module – only inputs and outputs of the main module can be connected directly to pins.
Introduction

The following reference manual and copyright permissions were obtained from Digilent Inc at http://www.digilentinc.com/index.cfm
Introduction

The Basys board is a circuit design and implementation platform that anyone can use to gain experience building real digital circuits. Built around a Xilinx Spartan-3E Field Programmable Gate Array and a Cypress EZ-USB controller, the Basys board provides complete, ready-to-use hardware suitable for hosting circuits ranging from basic logic devices to complex controllers. A large collection of on-board I/O devices and all required FPGA support circuits are included, so countless designs can be created without the need for any other components.

Four standard expansion connectors allow designs to grow beyond the Basys board using breadboards, user-designed circuit boards, or Pmods (Pmods are inexpensive analog and digital I/O modules that offer A/D & D/A conversion, motor drivers, sensor inputs, and many other features). Signals on the 6-pin connectors are protected against ESD damage and short-circuits, ensuring a long operating life in any environment. The Basys board works seamlessly with all versions of the Xilinx ISE tools, including the free WebPack. It ships with a USB cable that provides power and a programming interface, so no other power supplies or programming cables are required.

The Basys board can draw power and be programmed via its on-board USB2 port. Digilent’s freely available PC-based Adept software automatically detects the Basys board, provides a programming interface for the FPGA and Platform Flash ROM, and allows user data transfers at up to 400Mbytes/sec (see www.digilentinc.com for more information). The Basys board can also be programmed from within the Xilinx ISE tool environment using a Digilent JTAG3 (or suitable Xilinx cable) and a separate power supply.

The Basys board is designed to work with the free ISE WebPack CAD software from Xilinx. WebPack can be used to define circuits using schematics or HDLs, to simulate and synthesize circuits, and to create programming files. Webpack can be downloaded free of charge from www.xilinx.com/ise/.

The Basys board ships with a built-in self-test stored in its ROM that can be used to test all board features. To run the test, set the Mode Jumper (see below) to ROM and apply board power. If the test is erased from the ROM, it can be downloaded and reinstalled at any time. See www.digilentinc.com/basys for the test project as well as further documentation, reference designs, and tutorials.
Board Power

The Basys board is typically powered from a USB cable, but a power jack and battery connector are also provided so that external supplies can be used. To use USB power, set the power source switch (SW8) to USB and attach the USB cable. To use an external wall-plug power supply, set SW8 to EXT and attach a 5VDC to 9VDC supply to the center-positive, 2.1/5.5mm power jack. To use battery power, set SW8 to EXT and attach a 4V-9V battery pack to the 2-pin, 100-mil spaced battery connector (four AA cells in series make a good 6+/− volt supply). Voltages higher than 9V on either power connector may cause permanent damage. SW8 can also be used to turn off main power by setting it to the unused power input (e.g., if USB power is used, setting SW8 to EXT will shut off board power without unplugging the USB cable).

Input power is routed through the power switch (SW8) to the four 6-pin expansion connectors and to a National Semiconductor LP8345 voltage regulator. The LP8345 produces the main 3.3V supply for the board, and it also drives secondary regulators to produce the 2.5V and 1.2V supply voltages required by the FPGA. Total board current is dependant on FPGA configuration, clock frequency, and external connections. In test circuits with roughly 20K gates routed, a 50MHz clock source, and all LEDs illuminated, about 100mA of current is drawn from the 1.2V supply, 50mA from the 2.5V supply, and 50mA from the 3.3V supply. Required current will increase if larger circuits are configured in the FPGA, or if peripheral boards are attached.

The Basys board uses a four layer PCB, with the inner layers dedicated to VCC and GND planes. The FPGA and the other ICs on the board have large complements of ceramic bypass capacitors placed as close as possible to each VCC pin, resulting in a very clean, low-noise power supply.

Configuration

After power-on, the FPGA on the Basys board must be configured before it can perform any useful functions. During configuration, a “bit” file is transferred into memory cells within the FPGA to define the logical functions and circuit interconnects. The free ISE/WebPack CAD software from Xilinx can be used to create bit files from VHDL, Verilog, or schematic-based source files.

Digilent’s PC-based program called Adept can be used to configure the FPGA with any suitable bit file stored on the computer. Adept uses the USB cable to transfer a selected bit file from the PC to the FPGA (via the FPGA’s JTAG programming port). Adept can also program a bit file into an on-board non-volatile ROM called “Platform Flash”. Once programmed, the Platform Flash can automatically transfer a stored bit file to the FPGA at a subsequent power-on or reset event if the Mode Jumper is set to ROM. The FPGA will remain configured until it is reset by a...
A power-cycle event or by the FPGA reset button (BTN) being pressed. The Platform Flash ROM will retain a bit file until it is reprogrammed, regardless of power-cycle events.

To program the Basys board, attach the USB cable to the board (if USB power will not be used, attach a suitable power supply to the power jack or battery connector on the board, and set the power switch to VEXT). Start the Adept software, and wait for the FPGA and the Platform Flash ROM to be recognized. Use the browse function to associate the desired .bit file with the FPGA, and/or the desired .mcs file with the Platform Flash ROM. Right-click on the device to be programmed, and select the “program” function. The configuration file will be sent to the FPGA or Platform Flash, and the software will indicate whether programming was successful. The “configuration done” LED (LD_D) will also illuminate after the FPGA has been successfully configured. For further information on using Adept, please see the Adept documentation available at the Digilent website.

**Oscillators**

The Basys board includes a primary, user-settable silicon oscillator that produces 25MHz, 50MHz, or 100MHz based on the position of the clock select jumper at JP4. A socket for a second oscillator is provided at IC7 (the IC7 socket can accommodate any 3.3V CMOS oscillator in a half-size DIP package). The primary and secondary oscillators are connected to global clock input pins at pin 54 and pin 53 respectively.

Both clock inputs can drive the clock synthesizer DLL on the Spartan 3E, allowing for a wide range if internal frequencies, from 4 times the input frequency to any integer divisor of the input frequency.

The primary silicon oscillator is flexible and inexpensive, but it lacks the frequency stability of a crystal oscillator. Some circuits that drive a VGA monitor may realize a slight improvement in image stability by using a crystal oscillator installed in the IC7 socket. For these applications, a 25MHz (or 50MHz) crystal oscillator, available from any catalog distributor, is recommended (see for example part number SG-8002JF-PCC at www.digikey.com).
User I/O

Four pushbuttons and eight slide switches are provided for circuit inputs. Pushbutton inputs are normally low and driven high only when the pushbutton is pressed. Slide switches generate constant high or low inputs depending on position. Pushbuttons and slide switches all have series resistors for protection against short circuits (a short circuit would occur if an FPGA pin assigned to a pushbutton or slide switch was inadvertently defined as an output).

Eight LEDs and a four-digit seven-segment LED display are provided for circuit outputs. LED anodes are driven from the FPGA via current-limiting resistors, so they will illuminate when a logic ‘1’ is written to the corresponding FPGA pin. A ninth LED is provided as a power-indicator LED, and a tenth LED (LD-D) illuminates any time the FPGA has been successfully programmed.

Seven-segment display

Each of the four digits of the seven-segment LED display is composed of seven LED segments arranged in a “figure 8” pattern. Segment LEDs can be individually illuminated, so any one of 128 patterns can be displayed on a digit by illuminating certain LED segments and leaving the others dark. Of these 128 possible patterns, the ten corresponding to the decimal digits are the most useful.

The anodes of the seven LEDs forming each digit are tied together into one common anode circuit node, but the LED cathodes remain separate. The common anode signals are available as four “digit enable” input signals to the 4-digit display. The cathodes of similar segments on all four displays are connected into seven circuit nodes labeled CA through CG (so, for example, the four “D” cathodes from the four digits are grouped together into a single circuit node called “CD”). These seven cathode signals are available as inputs to the 4-digit display. This signal connection scheme creates a multiplexed display, where the cathode signals are common to all digits but they can only illuminate the segments of the digit whose corresponding anode signal is asserted.

A scanning display controller circuit can be used to show a four-digit number on this display. This circuit drives the anode signals and corresponding cathode patterns of each digit in a repeating, continuous succession, at an update rate that is faster than the human eye response. Each digit is
illuminated just one-quarter of the time, but because the eye cannot perceive the darkening of a digit before it is illuminated again, the digit appears continuously illuminated. If the update or "refresh" rate is slowed to a given point (around 45 hertz), then most people will begin to see the display flicker.

For each of the four digits to appear bright and continuously illuminated, all four digits should be driven once every 1 to 16ms (for a refresh frequency of 1KHz to 60Hz). For example, in a 60Hz refresh scheme, the entire display would be refreshed once every 16ms, and each digit would be illuminated for ¼ of the refresh cycle, or 4ms. The controller must assure that the correct cathode pattern is present when the corresponding anode signal is driven. To illustrate the process, if AN1 is asserted while CB and CC are asserted, then a “1” will be displayed in digit position 1. Then, if AN2 is asserted while CA, CB and CC are asserted, then a “7” will be displayed in digit position 2. If A1 and CB, CC are driven for 4ms, and then A2 and CA, CB, CC are driven for 4ms in an endless succession, the display will show “17” in the first two digits. Figure 8 shows an example timing diagram for a four-digit seven-segment controller.

**PS/2 Port**

The 6-pin mini-DIN connector can accommodate a PS/2 mouse or keyboard. Most PS/2 devices can operate from a 3.3V supply, but some older devices may require a 5VDC supply. A jumper on the Basys board (JP1) selects whether 3.3V or VU is supplied to the PS/2 connector. For 5V, set JP1 to VU and ensure that Basys is powered with a 5VDC wall-plug supply. For 3.3V, set the jumper to 3.3V. For 3.3V operation, any board power supply (including USB) can be used.

Both the mouse and keyboard use a two-wire serial bus (clock and data) to communicate with a host device. Both use 11-bit words that include a start, stop and odd parity bit, but the data packets are organized differently, and the keyboard interface allows bi-directional data transfers (so the host device can illuminate state LEDs on the keyboard). Bus timings are shown in the figure.
The clock and data signals are only driven when data transfers occur, and otherwise they are held in the "idle" state at logic ‘1’. The timings define signal requirements for mouse-to-host communications and bi-directional keyboard communications. A PS/2 interface circuit can be implemented in the FPGA to create a keyboard or mouse interface.

![PS/2 Connector and Basys PS/2 Circuit](image)

**Figure 9. PS/2 connector and Basys PS/2 circuit**

**Keyboard**

The keyboard uses open-collector drivers so the keyboard or an attached host device can drive the two-wire bus (if the host device will not send data to the keyboard, then the host can use input-only ports).

PS2-style keyboards use scan codes to communicate key press data. Each key is assigned a code that is sent whenever the key is pressed; if the key is held down, the scan code will be sent repeatedly about once every 100ms. When a key is released, a “F0” key-up code is sent, followed by the scan code of the released key. If a key can be “shifted” to produce a new character (like a capital letter), then a shift character is sent in addition to the scan code, and the host must determine which ASCII character to use. Some keys, called extended keys, send an “E0” ahead of the scan code (and they may send more than one scan code). When an extended key is released, an “EO F0” key-up code is sent, followed by the scan code. Scan codes for most keys are shown in the figure. A host device can also send data to the keyboard. Below is a short list of some common commands a host might send.

- **ED** Set Num Lock, Caps Lock, and Scroll Lock LEDs. Keyboard returns “FA” after receiving “ED”, then host sends a byte to set LED status: Bit 0 sets Scroll Lock; bit 1 sets Num Lock; and Bit 2 sets Caps lock. Bits 3 to 7 are ignored.
- **EE** Echo (test). Keyboard returns “EE” after receiving “EE”.
- **F3** Set scan code repeat rate. Keyboard returns “F3” on receiving “FA”, then host sends second byte to set the repeat rate.
- **FE** Resend. “FE” directs keyboard to re-send most recent scan code.
- **FF** Reset. Resets the keyboard.

The keyboard can send data to the host only when both the data and clock lines are high (or idle). Since the host is the “bus master”, the keyboard must check to see whether the host is sending data...
before driving the bus. To facilitate this, the clock line is used as a “clear to send” signal. If the host pulls the clock line low, the keyboard must not send any data until the clock is released.

The keyboard sends data to the host in 11-bit words that contain a ‘0’ start bit, followed by 8-bits of scan code (LSB first), followed by an odd parity bit and terminated with a ‘1’ stop bit. The keyboard generates 11 clock transitions (at around 20 - 30KHz) when the data is sent, and data is valid on the falling edge of the clock.

<table>
<thead>
<tr>
<th>ESC</th>
<th>76</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>~ OE</td>
<td>16</td>
</tr>
<tr>
<td>2 @</td>
<td>1E</td>
</tr>
<tr>
<td>3 #</td>
<td>26</td>
</tr>
<tr>
<td>5 $</td>
<td>25</td>
</tr>
<tr>
<td>6 ^</td>
<td>36</td>
</tr>
<tr>
<td>7 &amp;</td>
<td>3D</td>
</tr>
<tr>
<td>8 *</td>
<td>46</td>
</tr>
<tr>
<td>9 (</td>
<td>45</td>
</tr>
<tr>
<td>0 = +</td>
<td>55</td>
</tr>
<tr>
<td>4E</td>
<td>BackSpace</td>
</tr>
<tr>
<td>14</td>
<td>66</td>
</tr>
<tr>
<td>TAB</td>
<td>0D</td>
</tr>
<tr>
<td>Q</td>
<td>15</td>
</tr>
<tr>
<td>W</td>
<td>1D</td>
</tr>
<tr>
<td>E</td>
<td>24</td>
</tr>
<tr>
<td>R</td>
<td>2D</td>
</tr>
<tr>
<td>T</td>
<td>3C</td>
</tr>
<tr>
<td>Y</td>
<td>43</td>
</tr>
<tr>
<td>U</td>
<td>44</td>
</tr>
<tr>
<td>I</td>
<td>54</td>
</tr>
<tr>
<td>O</td>
<td>5B</td>
</tr>
<tr>
<td>P</td>
<td>5D</td>
</tr>
<tr>
<td>[</td>
<td>5A</td>
</tr>
<tr>
<td>{</td>
<td>Enter</td>
</tr>
<tr>
<td>Command</td>
<td>12</td>
</tr>
<tr>
<td>Ctrl</td>
<td>14</td>
</tr>
<tr>
<td>Shift</td>
<td>12</td>
</tr>
<tr>
<td>Z</td>
<td>1Z</td>
</tr>
<tr>
<td>X</td>
<td>22</td>
</tr>
<tr>
<td>C</td>
<td>21</td>
</tr>
<tr>
<td>V</td>
<td>2A</td>
</tr>
<tr>
<td>B</td>
<td>32</td>
</tr>
<tr>
<td>N</td>
<td>31</td>
</tr>
<tr>
<td>M</td>
<td>3A</td>
</tr>
<tr>
<td>,</td>
<td>41</td>
</tr>
<tr>
<td>.</td>
<td>49</td>
</tr>
<tr>
<td>/</td>
<td>4A</td>
</tr>
</tbody>
</table>
| ? | Shift

Figure 11. Keyboard scan codes

Mouse

The mouse outputs a clock and data signal when it is moved; otherwise, these signals remain at logic ‘1’. Each time the mouse is moved, three 11-bit words are sent from the mouse to the host device. Each of the 11-bit words contains a ‘0’ start bit, followed by 8-bits of data (LSB first), followed by an odd parity bit, and terminated with a ‘1’ stop bit. Thus, each data transmission contains 33 bits, where bits 0, 11, and 22 are ‘0’ start bits, and bits 11, 21, and 33 are ‘1’ stop bits. The three 8-bit data fields contain movement data as shown in the figure above. Data is valid at the falling edge of the clock, and the clock period is 20 to 30KHz.

The mouse assumes a relative coordinate system wherein moving the mouse to the right generates a positive number in the X field, and moving to the left generates a negative number. Likewise, moving the mouse up generates a positive number in the Y field, and moving down represents a negative number (the XS and YS bits in the status byte are the sign bits – a ‘1’ indicates a negative number). The magnitude of the X and Y numbers represent the rate of mouse movement – the larger the number, the faster the mouse is moving (the XV and YV bits in the status byte are movement overflow indicators – a ‘1’ means overflow has occurred). If the mouse moves continuously, the 33-bit transmissions are repeated every 50ms or so. The L and R fields in the status byte indicate Left and Right button presses (a ‘1’ indicates the button is being pressed).

Figure 12. Mouse data format
VGA Port

The Basys board uses 10 FPGA signals to create a VGA port with 8-bit color and the two standard sync signals (HS – Horizontal Sync, and VS – Vertical Sync). The color signals use resistor-divider circuits that work in conjunction with the 75-ohm termination resistance of the VGA display to create eight signal levels on the red and green VGA signals, and four on blue (the human eye is less sensitive to blue levels). This circuit, shown in figure 13, produces video color signals that proceed in equal increments between 0V (fully off) and 0.7V (fully on). A video controller circuit must be created in the FPGA to drive the sync and color signals with the correct timing in order to produce a working display system.

**VGA System Timing**

VGA signal timings are specified, published, copyrighted and sold by the VESA organization (www.vesa.org). The following VGA system timing information is provided as an example of how a VGA monitor might be driven in 640 by 480 mode. For more precise information, or for information on other VGA frequencies, refer to documentation available at the VESA website.

CRT-based VGA displays use amplitude-modulated moving electron beams (or cathode rays) to display information on a phosphor-coated screen. LCD displays use an array of switches that can impose a voltage across a small amount of liquid crystal, thereby changing light permittivity through the crystal on a pixel-by-pixel basis. Although the following description is limited to CRT displays, LCD displays have evolved to use the same signal timings as CRT displays (so the “signals” discussion below pertains to both CRTs and LCDs). Color CRT displays use three electron beams (one for red, one for blue, and one for green) to energize the phosphor that coats the inner side of the display end of a cathode ray tube (see illustration). Electron beams emanate from “electron guns” which are finely-pointed heated cathodes placed in close proximity to a positively charged annular plate called a “grid”. The electrostatic force imposed by the grid pulls rays of energized electrons from the cathodes, and those rays are fed by the current that flows into the cathodes. These particle rays are initially accelerated towards the grid, but they soon fall under the influence of the much
larger electrostatic force that results from the entire phosphor-coated display surface of the CRT being charged to 20kV (or more). The rays are focused to a fine beam as they pass through the center of the grids, and then they accelerate to impact on the phosphor-coated display surface. The phosphor surface glows brightly at the impact point, and it continues to glow for several hundred microseconds after the beam is removed. The larger the current fed into the cathode, the brighter the phosphor will glow.

Between the grid and the display surface, the beam passes through the neck of the CRT where two coils of wire produce orthogonal electromagnetic fields. Because cathode rays are composed of charged particles (electrons), they can be deflected by these magnetic fields. Current waveforms are passed through the coils to produce magnetic fields that interact with the cathode rays and cause them to transverse the display surface in a “raster” pattern, horizontally from left to right and vertically from top to bottom. As the cathode ray moves over the surface of the display, the current sent to the electron guns can be increased or decreased to change the brightness of the display at the cathode ray impact point.

Information is only displayed when the beam is moving in the “forward” direction (left to right and top to bottom), and not during the time the beam is reset back to the left or top edge of the display. Much of the potential display time is therefore lost in “blanking” periods when the beam is reset and stabilized to begin a new horizontal or vertical display pass. The size of the beams, the frequency at which the beam can be traced across the display, and the frequency at which the electron beam can be modulated determine the display resolution. Modern VGA displays can accommodate different resolutions, and a VGA controller circuit dictates the resolution by producing timing signals to control the raster patterns. The controller must produce synchronizing pulses at 3.3V (or 5V) to set the frequency at which current flows through the deflection coils, and it must ensure that video data is applied to the electron guns at the correct time. Raster video displays define a number of “rows” that corresponds to the number of horizontal passes the cathode makes over the display area, and a number of “columns” that corresponds to an area on each row that is assigned to one “picture element” or pixel. Typical displays use from 240 to 1200 rows and from 320 to 1600 columns. The overall size of a display and the number of rows and columns determines the size of each pixel.

Video data typically comes from a video refresh memory, with one or more bytes assigned to each pixel location (the Basys uses three bits per pixel). The controller must index into video memory as the beams move

![Figure 15. VGA system signals](image)
across the display, and retrieve and apply video data to the display at precisely the time the electron beam is moving across a given pixel.

A VGA controller circuit must generate the HS and VS timings signals and coordinate the delivery of video data based on the pixel clock. The pixel clock defines the time available to display one pixel of information. The VS signal defines the “refresh” frequency of the display, or the frequency at which all information on the display is redrawn. The minimum refresh frequency is a function of the display’s phosphor and electron beam intensity, with practical refresh frequencies falling in the 50Hz to 120Hz range. The number of lines to be displayed at a given refresh frequency defines the horizontal “retrace” frequency. For a 640-pixel by 480-row display using a 25MHz pixel clock and 60 +/-1Hz refresh, the signal timings shown in the table at right can be derived. Timings for sync pulse width and front and back porch intervals (porch intervals are the pre- and post-sync pulse times during which information cannot be displayed) are based on observations taken from actual VGA displays.

A VGA controller circuit decodes the output of a horizontal-sync counter driven by the pixel clock to generate HS signal timings. This counter can be used to locate any pixel location on a given row. Likewise, the output of a vertical-sync counter that increments with each HS pulse can be used to generate VS signal timings, and this counter can be used to locate any given row. These two continually running counters can be used to form an address into video RAM. No time relationship between the onset of the HS pulse and the onset of the VS pulse is specified, so the designer can arrange the counters to easily form video RAM addresses, or to minimize decoding logic for sync pulse generation.
Expansion Connectors (6-pin headers)

The Basys board provides four 6-pin peripheral module connectors. Each connector provides Vdd, GND, and four unique FPGA signals. Several 6-pin module boards that can attach to this connector are available from Digilent, including A/D converters, speaker amplifiers, microphones, H-bridge amplifiers, etc. Please see www.digilentinc.com for more information.

FPGA Pin Definitions

The table below shows all pin definitions for the Spartan-3E on the Basys board. Pins in grey boxes are not available to the user.

| Grey | Not available to user |
| Green | User I/O devices |
| Yellow | Data ports |
| Tan | Pmod connector signals |
| Blue | USB signals |

### Basys Spartan-3E pin definitions

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PROG25</td>
<td>25</td>
<td>CA</td>
<td>49</td>
<td>VDDO-2</td>
<td>73</td>
<td>GND</td>
<td>97</td>
<td>PS2D</td>
</tr>
<tr>
<td>2</td>
<td>LD7</td>
<td>26</td>
<td>AN3</td>
<td>50</td>
<td>GRN2</td>
<td>74</td>
<td>JD-3</td>
<td>98</td>
<td>NC</td>
</tr>
<tr>
<td>3</td>
<td>LD6</td>
<td>27</td>
<td>GND</td>
<td>51</td>
<td>GRN1</td>
<td>75</td>
<td>JD-1</td>
<td>99</td>
<td>NC</td>
</tr>
<tr>
<td>4</td>
<td>LD5</td>
<td>28</td>
<td>VDDO-3</td>
<td>52</td>
<td>GRN0</td>
<td>76</td>
<td>JC-3</td>
<td>100</td>
<td>VDDO-1</td>
</tr>
<tr>
<td>5</td>
<td>LD4</td>
<td>29</td>
<td>SW2</td>
<td>53</td>
<td>CLK2</td>
<td>77</td>
<td>JC-1</td>
<td>101</td>
<td>NC</td>
</tr>
<tr>
<td>6</td>
<td>SW7</td>
<td>30</td>
<td>VDDAUX</td>
<td>54</td>
<td>CLK1</td>
<td>78</td>
<td>NC</td>
<td>102</td>
<td>VDDAUX</td>
</tr>
<tr>
<td>7</td>
<td>LD3</td>
<td>31</td>
<td>NC</td>
<td>55</td>
<td>GND</td>
<td>79</td>
<td>VDDO-1</td>
<td>103</td>
<td>NC</td>
</tr>
<tr>
<td>8</td>
<td>LD2</td>
<td>32</td>
<td>AN2</td>
<td>56</td>
<td>NC</td>
<td>80</td>
<td>VDDINT</td>
<td>104</td>
<td>U-SLWR</td>
</tr>
<tr>
<td>9</td>
<td>VDDINT</td>
<td>33</td>
<td>AN1</td>
<td>57</td>
<td>MODE2</td>
<td>81</td>
<td>JA-1</td>
<td>105</td>
<td>U-SLCS</td>
</tr>
<tr>
<td>10</td>
<td>SW6</td>
<td>34</td>
<td>AN0</td>
<td>58</td>
<td>JD-4</td>
<td>82</td>
<td>JA-3</td>
<td>106</td>
<td>U-SLCS</td>
</tr>
<tr>
<td>11</td>
<td>GND</td>
<td>35</td>
<td>VS</td>
<td>59</td>
<td>JD-2</td>
<td>83</td>
<td>CG</td>
<td>107</td>
<td>NC</td>
</tr>
<tr>
<td>12</td>
<td>SW5</td>
<td>36</td>
<td>SW1</td>
<td>60</td>
<td>MODE1</td>
<td>84</td>
<td>NC</td>
<td>108</td>
<td>TMS</td>
</tr>
<tr>
<td>13</td>
<td>VDDO-3</td>
<td>37</td>
<td>GND</td>
<td>61</td>
<td>GND</td>
<td>85</td>
<td>JC-4</td>
<td>109</td>
<td>TDO</td>
</tr>
<tr>
<td>14</td>
<td>VDDO-4</td>
<td>38</td>
<td>SW0</td>
<td>62</td>
<td>MODE0</td>
<td>86</td>
<td>JC-2</td>
<td>110</td>
<td>TCK</td>
</tr>
<tr>
<td>15</td>
<td>LD1</td>
<td>39</td>
<td>HS</td>
<td>63</td>
<td>DIN</td>
<td>87</td>
<td>JB-1</td>
<td>111</td>
<td>NC</td>
</tr>
<tr>
<td>16</td>
<td>LD0</td>
<td>40</td>
<td>INIT</td>
<td>64</td>
<td>VDDO-2</td>
<td>88</td>
<td>JB-3</td>
<td>112</td>
<td>U-PKTD</td>
</tr>
<tr>
<td>17</td>
<td>CB</td>
<td>41</td>
<td>BTN3</td>
<td>65</td>
<td>VDDAUX</td>
<td>89</td>
<td>NC</td>
<td>113</td>
<td>U-FAD1</td>
</tr>
<tr>
<td>18</td>
<td>CF</td>
<td>42</td>
<td>VDDO-2</td>
<td>66</td>
<td>NC</td>
<td>90</td>
<td>GND</td>
<td>114</td>
<td>NC</td>
</tr>
<tr>
<td>19</td>
<td>SW4</td>
<td>43</td>
<td>BLUE1</td>
<td>67</td>
<td>RED2</td>
<td>91</td>
<td>JA-2</td>
<td>115</td>
<td>VDDINT</td>
</tr>
<tr>
<td>20</td>
<td>SW3</td>
<td>44</td>
<td>BLUE0</td>
<td>68</td>
<td>RED1</td>
<td>92</td>
<td>JA-4</td>
<td>116</td>
<td>U-FAD0</td>
</tr>
<tr>
<td>21</td>
<td>CE</td>
<td>45</td>
<td>VDDINT</td>
<td>69</td>
<td>BTN0</td>
<td>93</td>
<td>JB-2</td>
<td>117</td>
<td>U-SLDE</td>
</tr>
<tr>
<td>22</td>
<td>DP</td>
<td>46</td>
<td>GND</td>
<td>70</td>
<td>RED0</td>
<td>94</td>
<td>JB-4</td>
<td>118</td>
<td>GND</td>
</tr>
<tr>
<td>23</td>
<td>CC</td>
<td>47</td>
<td>BTN2</td>
<td>71</td>
<td>CCLK</td>
<td>95</td>
<td>NC</td>
<td>119</td>
<td>NC</td>
</tr>
<tr>
<td>24</td>
<td>SW2</td>
<td>48</td>
<td>BTN1</td>
<td>72</td>
<td>DONE</td>
<td>96</td>
<td>PS2C</td>
<td>120</td>
<td>NC</td>
</tr>
</tbody>
</table>

Note: Every 6-pin connector has a power supply jumper and ESD diodes, although they are only shown for JPA.
Built in Self Test

A demonstration configuration is loaded into the Platform Flash ROM during manufacturing. This demo, also available on the resource CD and on the Digilent website, can serve as a board verification test since it interacts with all devices and ports on the Basys board. To configure the FPGA from a bit file stored in Platform Flash, set the Mode Jumper to ROM and cycle power or press the FPGA reset button (BTN R).

The self-test connects the switches to the LEDs, the buttons and PS/2 keyboard (if attached) to the seven-segment display, and a VGA monitor (if attached) will show a color pattern. By interacting with the buttons and switches on the Basys board (and the keys on a PS/2 keyboard if attached), and watching the LEDs, seven-segment display, and VGA monitor (if attached), any hardware problems with the Basys board can be readily identified.

If the self test is not resident in the Platform Flash ROM, it can be programmed into the FPGA or reloaded into the ROM using the Adept programming software.