FED 101 – Fundamentals of Engineering Design

Electrical and Computer Engineering Module

Laboratory Manual and Supplementary Notes

Updated August 2017

John D. Carpinelli
Mohammed Feknous
Marek Sosnowski
Department of Electrical and Computer Engineering
New Jersey Institute of Technology
University Heights
Newark, New Jersey, 07102-1982

with contributions from Wayne Clements, Joseph Frank, and Joseph Strano

Table of Contents

1 Introduction

1.1 Introduction to FED

FED 101 (Fundamentals of Engineering Design) is a required course for all NCE freshmen. It has different modules associated with different NCE departments and this manual is intended for the Electrical and Computer Engineering Module (ECEM). If you have declared that you wish to study electrical or computer engineering, you are assigned to this module. If you are still undecided, taking FED 101 - EEM may be a good way to find out what ECE is all about. If after taking this course you decide that ECE is not for you, so be it. While we hope that this will not happen, it is important to make a good decision about your professional future based on real information, not on myth.

FED 101 is very different than a typical college course focused on studying in depth a specific technical topic. FED was developed as a part on a national initiative aimed at improving engineering education carried out by Gateway Engineering Education Coalition and supported by the National Science Foundation. NJIT was among several top universities, which included Columbia, Cooper Union, Ohio State, Brooklyn Polytechnics and others, which took part in this effort.
This course covers a broad range of subjects and attempts to expose you to information, skills and experience needed for successful collage work and future professional activity. Among the prime course goals are:

- Inform you what electrical and computer engineers do and excite you about the field
- Enhance your ability to learn (learning to learn) and work effectively in peer groups
- Improve your communication skills
- Introduce you to basic electrical engineering concepts, devices and circuits as well as computer software for their simulation.

We hope that taking this course will be enriching and enjoyable experience. Good luck!

1.2 Introduction to ECE

ECE –Electrical and Computer Engineering is a broad field of theoretical and practical knowledge related to operation of devices and systems based electricity and magnetism. It would be impossible to find today any area of modern life that does not rely on extensive use of electrical power, electronic communication, or computers. Just imagine what life would be like without these technologies. That's why getting a degree in computer engineering or electrical engineering is so great. With the knowledge and experience you gain you can work in companies focusing not only on electronics or computers but also in those in communication, transportation, or aerospace industries, as well as in the areas of bioengineering and medical technology, and many others.
The ECE department at NJIT is an organization of faculty and staff devoted to providing an outstanding academic and research experience to students to prepare them to meet the needs and challenges they will face in the world of technology. Department has been recently reorganized into thrust areas to meet the challenges of future technology development, advancement of science and engineering, and industry needs. The Department has over 40 faculty members, over 1100 undergraduate and 450 graduate students with an annual research expenditure of over $6 million and growing.

The academic program in the ECE department is divided into two tracks: Electrical Engineering (EE) and Computer Engineering (CoE), with some overlap in the coursework. Undergraduate students in the senior year of the EE program are able to select from seven specialized areas:

- **Communication Network:** The information revolution is built on an infrastructure of communications and computer networks. This infrastructure has the potential to drastically change the way we live and work. The communications networks track focuses on the analysis and design of wireless & wireline systems for information delivery. In addition to the systems elective, which emphasizes networks, a variety of courses are available in topics such as optical communications networks and wireless communications.
- **Telecommunications:** Wireless Telecommunications is rapidly becoming one of the most rapidly developing and exciting fields in today's technology. The Telecommunications track seeks to prepare individuals to contribute to the diverse fields of wireless communications. The systems elective focuses on wireless personal communications systems, while the electives cover a wide area of topics such as networks and optical communications.
- **Controls:** The mechanism of feedback pervades nature, science, and technology. The curriculum in control teaches how engineers can use the feedback mechanism to design systems for controlling a variety of dynamic processes, ranging from spacecraft, aircraft, and automobile emission systems to heating, ventilating, and air conditioning systems.
- **Computers:** The computer system elective provides an in-depth study of computer system organization and computer system design. Students study CPU design, control unit design, memory organization and I/O processing.
- **Power Systems:** The Power Systems elective includes the study of the economical generation and stable transmission of electrical energy to consumers.
- **RF/Microwave and Fiber Optics:** This area concentrates on radio frequency (rf) microwave and lightwave technologies at the component and the system levels. Applications include communication systems (rf/microwave and fiber optics), remote sensing, radar, sensors, etc.

Areas of specialization in the CoE program include:

- **Computer Networking**
- **Advanced Computer Architecture**
- **Telecommunications**

You are invited to visit various homepages of the ECE department for further information on our academic and research programs, faculty and staff and state-of-the-art facilities and resources to support high quality educational programs with life-long learning opportunities.

1.3 Design Exercise – Paper Drop Competition

In this exercise, you will play the role of the engineer. You are given a goal and must design a solution to achieve that goal.

1.3.1 Design Specification

Each team is required to design and construct a "flying" device. There are two design criteria for this device.

1. The device must stay in the air as long as possible.
2. The device must land as close as possible to a given target.

Each team must construct their device using any or all of the following materials.

- Three sheets of 8½" x 11" paper
- Adhesive tape
- One 3" x 5" index card
- Four paper clips
- A pair of scissors

1.3.2 Scoring

A design competition will be held among all teams in the class. One member of each team will launch the device from a predetermined height toward a target on the lower floor. The time will be recorded from when the device is launched until it hits the ground. Then the distance will be measured from the device to the target. Each team will perform two drop runs; the times and distances will be totaled for each team.

The scoring for this competition emphasizes flight time over accuracy. The length of time before reaching the ground comprises 70% of the overall score, and the distance from the target accounts for the other 30% of the score. The scores are scaled by the slowest and fastest times or closest and farthest distances. The formula for calculating the time portion of the score, a maximum of 70 points, is as follows.

$$\text{Time score} = \frac{(\text{Your team's time} - \text{Shortest team's time})}{(\text{Longest team's time} - \text{Shortest team's time})} \times 70$$

To illustrate how this works, consider three teams with total times of 4, 8, and 11 seconds. The formula becomes

$$\text{Time score} = \frac{(\text{Your team's time} - 4\text{ seconds})}{(11\text{ seconds} - 4\text{ seconds})} \times 70$$

For the three teams, this is

$$\text{Time score} = \frac{(4 \text{ seconds} - 4 \text{ seconds})}{(11 \text{ seconds} - 4 \text{ seconds})} \times 70 = 0 \text{ points}$$

$$\text{Time score} = \frac{(8 \text{ seconds} - 4 \text{ seconds})}{(11 \text{ seconds} - 4 \text{ seconds})} \times 70 = 40 \text{ points}$$

$$\text{Time score} = \frac{(11 \text{ seconds} - 4 \text{ seconds})}{(11 \text{ seconds} - 4 \text{ seconds})} \times 70 = 70 \text{ points}$$

The longest time always earns 70 points and the shortest time receives no points. Other times earn varying numbers of points; the closer they are to the maximum time, the greater the number of points they earn.

The distance scores are calculated in a similar manner using the following formula.

$$\text{Distance score} = \frac{(\text{Longest team's distance} - \text{Your team's distance})}{(\text{Longest team's distance} - \text{Shortest team's distance})} \times 30$$

1.3.3 Acknowledgments

Thanks to Stephen Tricamo, Professor of Industrial and Manufacturing Engineering at NJIT, for allowing us to adapt this experiment from one he developed for his FED 101D class.

2 Electricity

This chapter introduces the physical phenomenon of electricity. First it examines the overall structure of the atom, the most fundamental basis for studying electricity. Next, it examines the basics of electricity – charge, current, and voltage. The two types of current electricity, direct current and alternating current, are introduced, along with differences between conductors and insulators.

Next, this chapter introduces the topic of resistance, and its relation to voltage and current. The color codes for resistors are presented, and circuits using resistors in series and in parallel are examined. Potentiometers, resisters whose values can be varied, and photoresistors, resistors whose values vary based on the amount of light they receive, are introduced. Laboratory Experiments 1 and 2 correspond to the material covered in this chapter.

2.1 Electricity, Charge, and Current

2.1.1 Atomic Structure

To understand current, it is necessary to first understand the basic structure of the atom. An atom is composed of three basic types of particles. The nucleus, or center of the atom, contains some number of *protons* and *neutrons*. The protons are positively charged particles, and the number of protons in an atom determines the type of atom. For example, all hydrogen atoms have exactly one proton, and all atoms with 13 protons are aluminum. Most atoms (except for most hydrogen atoms) have one or more neutrons in their nuclei as well. Neutrons have no charge, and the number of neutrons in an atom may vary. Atoms with the same number of protons but different numbers of neutrons are called *isotopes*.

Outside of the nucleus, negatively charged particles called *electrons* orbit the nucleus. Electrons are much smaller and lighter than protons. The attraction between the positively charged protons in the nucleus and the negatively charged electrons normally keeps the electrons in orbit around the nucleus. The number of electrons is the same as the number of protons, and the positive and negative charges cancel out; the atom has no net charge. A typical atom is shown schematically in Figure 2.1.

Figure 2.1: Basic atomic structure. Negative electrons orbit the positive nucleus at the center.

Sometimes an electron can escape from the orbit of its atom and be captured by another atom. The original atom now has one less electron and the atom has a net positive charge. The atom that captures the electron has one more electron than it has protons, resulting in a net negative charge. These are called *ions*, a general term that applies to all atoms with non-zero net charge, either positive or negative.

This last phenomenon, electrons moving between atoms, is the basis for electricity. Electricity is created by the flow of electrons. We'll examine this in the following subsections.

2.1.2 Charge

In the previous section we mentioned that electrons have a negative charge, but how much is this charge? To quantify charge, scientists have defined a *Coulomb* as the amount of charge contained by 6,250,000,000,000,000,000 (6.25×10^{18}) electrons. Although this sounds like a lot of electrons, keep in mind that one mole of hydrogen atoms, 6.022×10^{23} atoms, weighs only one gram, and the vast majority of that gram comes from the protons!

Charge is typically denoted as *Q*. Coulombs is abbreviated *C*; for example, a charge of 12 Coulombs is denoted as $Q = 12C$..

2.1.3 Current

Electric current is the flow of charged particles in a specific direction. In liquids and gases, these charged particles can be electrons or ions. In solids, such as the wire used in electrical circuits, electrons are the charged particles that cause electric current. Since the charge of a single electron is very small indeed ($q_e = 1.6 \times 10^{-19}$ C), any practical current involves flow of many electrons.

Current is typically denoted as *I* (for intensity). Current is the amount of charge flowing per unit time. This can be denoted by the equation:

$$I = \frac{Q}{t}$$

The same amount of charge flowing over a longer period of time would produce a smaller current, just as having a street where 20 cars pass through an intersection in one minute would be considered to have greater traffic than having the same 20 cars pass through the same intersection in an hour. The basic unit of current is the *Ampere*, or *Amp*, denoted as *A*. One Ampere is defined as the flow of one Coulomb of charge per second.

The following point is really important. *Although the electrons in a metal wire flow from the negative terminal to the positive terminal, the current flows from the positive terminal to the negative terminal*. The electrons carry a negative charge, and the current is defined by convention as the flow of positive charge. This is sort of like subtracting 1 and -1. 1 - (-1) is the same as 1 + (+1). The negative charge of the electrons flowing in one direction gives the same current as the positive charge flowing in the opposite direction. The convention of the current flow from positive to negative terminal was established before electrons were discovered and later people did not bother to change it.

2.1.4 Direct Current and Alternating Current

Two types of electric current are used in everyday life. *Direct current*, or *DC*, always flows in the same direction. This is the type of current created by batteries. The other type of current is *alternating current*, or *AC*. This is the current used to power household appliances and lights. This type of current periodically changes direction, once every 1/120 seconds in the United States (or once every 1/100 seconds in Europe).

2.1.5 Voltage

Voltage, also called the *electromotive force* or *potential difference*, is the force that causes current to flow. It can be helpful to visualize voltage as a difference in potential energy caused by differences in charges. Consider a simple battery; it has positive and negative terminals. Chemicals inside the battery cause positive charges to congregate near the positive terminal and negative charges to collect near the negative terminal. If we connected a wire from one terminal to the other, electrons would flow from the negative terminal to the positive terminal, creating a current in the wire. Eventually the charges on each side of the battery will become more neutral, and the battery will die out.

Voltage is defined as energy per unit charge. The basic unit of voltage is the *Volt*, abbreviated as *V*. One volt is equal to one joule per Coulomb. The more volts a battery has, the more joules of energy it supplies per coulomb.

2.1.6 Power

The flow of charges requires energy, and the energy per unit time, or power *P*, is proportional to both current and voltage

$$P = I \times V$$

Power is measured in watts. One watt is one joule per second, which is equal to one ampere times one volt.

2.1.7 Conductors and Insulators

Some materials allow electrons to flow more freely than others. *Conductors* are materials that give up electrons easily, offering little opposition (resistance) to current flow. Copper is a very good conductor; that is why house wiring is usually made of copper.

Other materials, called *insulators*, do not yield electrons easily. They offer high resistance to current flow. They are not perfect; some electrons do flow in insulators. However, the amount is so small that, for all practical purposes, virtually no current flows. Insulators are useful for wrapping wires, causing all current to flow from one end of the wire to the other and not allowing current to escape from within the wire. This is why an extension cord that is plugged into a wall outlet can be handled safely, as long as there is no break in the insulation!

2.2 Resistors and Ohm's Law

Resistors are fundamental components in electric circuit design. As their name implies, they resist the flow of current in a circuit. The next several sections examine resistors, their color codes, and circuits that use resistors in series, in parallel, or in both configurations. Potentiometers and photoresistors, resistors whose values can be varied, are also described.

To examine the relationship between the voltage, current, and resistance in a circuit, we will start with the simple circuit shown in Figure 2.2.

Figure 2.2: A simple 1-resistor circuit

The circle on the left hand side of the figure is a power source. It has a voltage of 1.5V, the voltage level of a standard battery. The positive terminal of the battery (+) is connected via a wire, represented by straight lines, to one end of a resistor, which is denoted by the zigzag lines. The other end of the resistor is connected to the negative terminal of the battery (-) with wires, completing the circuit.

The value of the resistor is based on how well it resists the flow of electrons. A higher resistance allows fewer electrons to flow through the resistor in a given time, reducing the current. The basic unit of measure of resistance is the *Ohm*, denoted by **Ω**, the Greek letter Omega. One Ohm is defined as the value of the resistance that lets the current of 1 A to flow under a voltage of 1 V (one volt per ampere). The resistor in this circuit has a value of 100 **Ω**.

Ohm's Law defines the relationship between voltage, current, and resistance. It was developed by German physicist Georg Ohm, for whom both Ohm's Law and the unit of measure for resistance were named. It states that the voltage (*V*) in a circuit is equal to the product of the current (*I*) and the resistance (*R*), or

$$V = I \times R$$

Manipulating this equation, we can express the current or resistance as a function of the other terms in the equation as follows.

$$I = \frac{V}{R} \qquad \text{and} \qquad R = \frac{V}{I}$$

A current flowing through a resistance generates power (in form of heat).

$$P = I \times V = \frac{V}{R} \times V = \frac{V^2}{R} = I^2 R$$

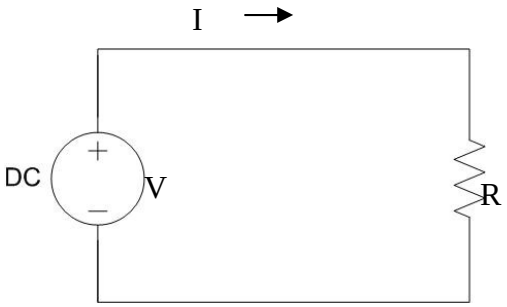
Now let's look back at the circuit in Figure 2.3. With a voltage of 1.5V and a resistance of 100 Ω, we can calculate its current as

$$I = \frac{V}{R} = \frac{1.5\ V}{100\ \Omega} = 0.015\ A = 15mA$$

The symbol *mA* stands for mili Amperes, or one-thousandths of an Ampere.

Worksheet 2.1 – Single-Resistor Circuit

For the circuit shown below, show the missing values for each set of given values.



1. V = 10V I = 0.5A R = _____ P = _____

2. V = 2V I = _____ R = 400 Ω

3. V = _____ I = 0.1A R = 1kΩ (=1000 Ω) P = _____

4. V = 25V I = 0.125A R = _____

5. V = 8V I = _____ R = 640 Ω P = _____

6. V = _____ I = 1.2A R = 120 Ω

2.3 Resistor Color Codes

If you look at a resistor, you won't see a number stamped on it indicating its value. Instead, it has several colored stripes that indicate its resistance. There may be three, four, or five stripes on a resistor. In this section we'll examine how these stripes represent the resistor's value.

Each stripe can be one of several colors. For the first two stripes (or the first three stripes for resistors with five stripes), the colors represent digits from zero to nine. The colors corresponding to each digit are shown in Table 2.1.

Table 2.1: Resistor colors and digit values for resistor magnitudes

Color	Digit
Black	0
Brown	1
Red	2
Orange	3
Yellow	4
Green	5
Blue	6
Violet	7
Grey	8
White	9

These digits give the base value of the resistor. For example, consider a resistor with four stripes with colors red, black, yellow, and gold, respectively. The first color, red, represents the digit 2, and black corresponds to 0. Together they give a resistance value of 20.

However, the next stripe changes this value considerably. The value corresponding to yellow is 4, but this 4 is not added to the end of the 20 to create 204. Instead, it means that 20 is multiplied by 10^4, or 20,000; the actual resistance is 20×10^4, or 200,000 Ω, or 200 kΩ. Table 2.2 shows the exponent values for this stripe. Note that the values are the same as in Table 2.1, except two new colors are available. Gold, representing 10^{-1}, and silver, which corresponds to 10^{-2}, are used for very small resistances. A program that shows the values of a resistor based on its color code is available via the ECE Department laboratory web site.

Table 2.2: Resistor colors and values for resistor multipliers

Color	Multiplier
Black	10^0
Brown	10^1
Red	10^2
Orange	10^3
Yellow	10^4
Green	10^5
Blue	10^6
Violet	10^7
Grey	10^8
White	10^9
Gold	10^{-1}
Silver	10^{-2}

It is very unlikely that this resistor is exactly 200,000 Ω. Manufacturing processes aren't perfect, and the actual resistance may be greater than or less than its stated value. For the four and five stripe resistors, the last stripe indicates the *tolerance* of the resistor. The tolerance specifies the percentage that the actual resistance may vary from its marked value. Table 2.3 shows the more common tolerance values. For our 200 kΩ resistor, the gold stripe indicates a tolerance of ±5%, so our resistor may have any value from 200 kΩ – 5%, or 190 kΩ, up to a maximum of 200 kΩ + 5%, or 210 kΩ.

Table 2.3: Resistor colors and values for resistor tolerances

Color	Tolerance
Brown	±1%
Red	±2%
Gold	±5%
Silver	±10%

Resistors with tolerances of 5% or 10% have four stripes. A resistor with only three stripes does not display a tolerance explicitly. By default, its tolerance is 20%.

There are also resistors made with tighter tolerances of 1% and 2% designated by the last stripes red or gold, respectively. These resistors have five stripes with first three designating a three digit number, the fourth stripe a multiplier (power of ten) and the fifth stripe the tolerance. The reason for an additional stripe is that more digits are required to specify more precisely the values of these resistors, which are made more precisely than the four stripe resistors.

Worksheet 2.2 – Resistor Color Codes

What are the resistances, and tolerances, of resistors with the following colored stripes?

1. Green, Blue, Red

2. White, Yellow, Green, Silver

3. Brown, Orange, Violet, Yellow, Brown

Show the colors found on resistors with the following values and tolerances.

1. 1.2 kΩ ±20%

2. 88 Ω ±5%

3. 3.14 Ω ±2%

2.4 Series Resistors

A typical circuit will have more than one resistor. Resistors in a circuit may be configured in series, in parallel, or in a combination of the two. This section examines resistors connected in series; parallel resistance is examined in the next section.

Resistors that are connected end-to-end are said to be connected in series. Figure 2.3 shows a circuit with two resistors connected in series. One hallmark of series resistance is that the same current that flows through one resistor must flow through the other resistor as well. There is only one path for the current to flow in this circuit.

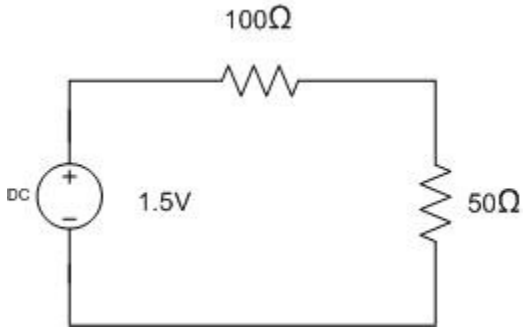


Figure 2.3: Circuit with two resistors in series

When two resistors are connected in series, their overall resistance is the sum of their individual resistances. For the circuit in Figure 2.3, the two series resistors have values of 100 Ω and 50 Ω; their overall resistance is 100 Ω + 50 Ω = 150 Ω. Using Ohm's Law, we can calculate the current in the circuit using this combined resistance.

$$I = \frac{V}{R} = \frac{1.5\,V}{150\,\Omega} = 0.01\,A = 10mA$$

2.5 Parallel Resistors

Resistors are not always connected in series; they can also be connected in parallel. Figure 2.4 shows a circuit with two resistors connected in parallel. Notice that both ends of the two resistors are connected together.

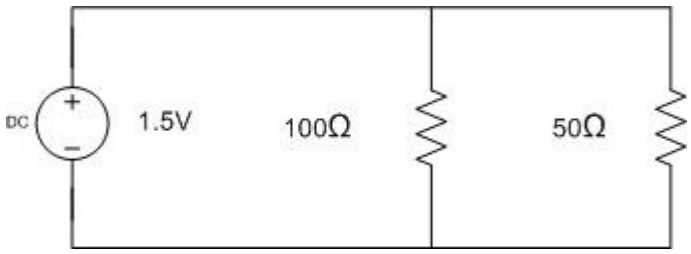


Figure 2.4: Circuit with two resistors in parallel

Although it might not seem to make sense, the overall resistance of two resistors connected in parallel is *less* than the resistance of either resistor! The basic reason this is true has to do with the current flow; adding another resistor in parallel increases the amount of current flowing in the circuit. From Ohm's Law, I = V/R; as current (I) increases and the voltage (V) remains the same, the overall resistance (R) must decrease.

Let's look at the circuit in Figure 2.4. Each resistor is connected directly to the positive and negative terminals of the battery, so each has a voltage of 1.5V. The 100 Ω resistor has a current of 1.5V/100 Ω = 15 mA, and the 50 Ω resistor has a current of 1.5V/50 Ω = 30 mA. Together the circuit has a current of 15mA + 30 mA = 45 mA. For the overall circuit, using Ohm's Law, we find R = 1.5V/45mA = 33.3 Ω.

A standard formula, called the *reciprocal formula*, is used to calculate the net resistance of two or more resistors in parallel. The reciprocal of the overall resistance is equal to the sum of the reciprocals of the individual resistors, or

$$R_{OVERALL} = \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} + \ldots$$

This formula can be simplified for circuits with only two resistors. The formula becomes

$$R_{OVERALL} = \frac{R_1 R_2}{R_1 + R_2}$$

2.6 Series-Parallel Resistors

As their name implies, series-parallel circuits have resistors in series and in parallel. Figure 2.5 shows two series-parallel circuits.

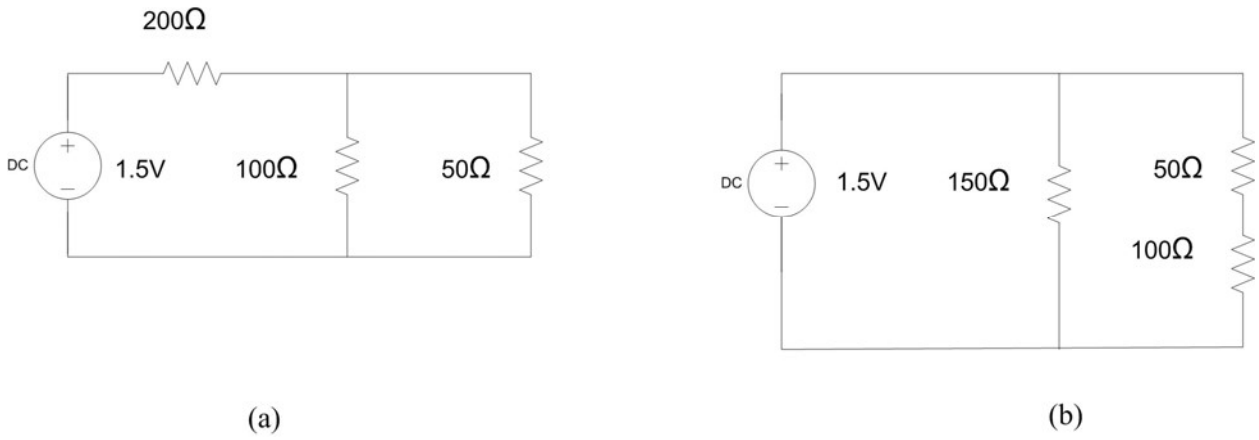


Figure 2.5: Two series-parallel circuits

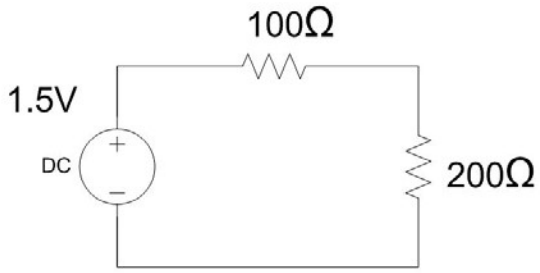
We can analyze series-parallel circuits by breaking them into their individual series and parallel components. For the circuit in Figure 2.5(a), the 50 Ω and 100 Ω resistors are in parallel; their net resistance, calculated using the reciprocal formula, is 33.3 Ω. That equivalent resistance is in series with the 200 Ω resistor, producing a net resistance of 233.3 Ω in the circuit. This yields a current of 6.4mA.

For the circuit in Figure 2.5(b), we first combine the 50 Ω and 100 Ω series resistors, which results in a net resistance of 150 Ω. Combining this in parallel with the 150 Ω resistor yields a net resistance of 75 Ω, and a current of 20mA.

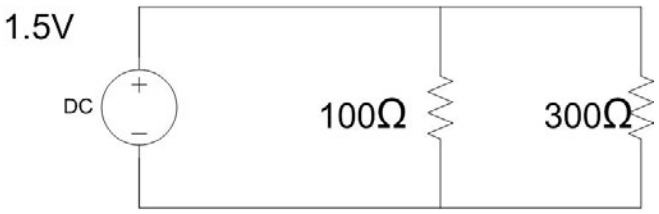
Worksheet 2.3 – Circuit Analysis

What is the net resistance and overall current for the following circuits?

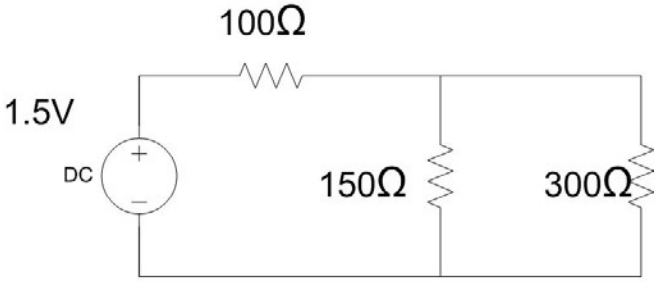
1.



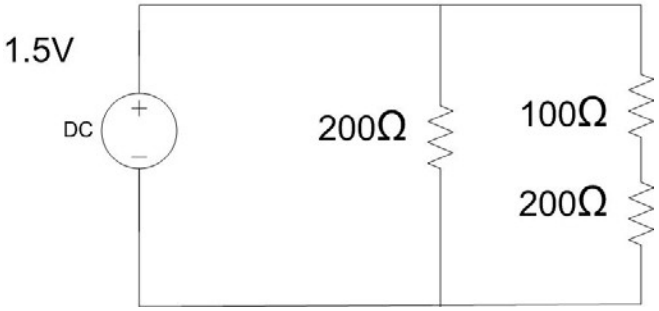
2.



3.



4.



2.7 Potentiometers and Photoresistors

For some circuits, it would be preferable to allow a user to change the value of the resistance without having to re-wire the circuit. This is the role of the *potentiometer*. It is a variable resistor whose value can be changed, typically by turning a shaft or sliding a lever.

There are several applications for potentiometers. Dimmer switches used to vary the intensity of lights are typically potentiometers, or their close relatives *rheostats*. Volume control knobs on older radios and televisions are also potentiometers.

A photoresistor also changes its value, but the user does not directly change the value of the resistance. Instead, the resistance varies depending on the amount of light sensed by the photoresistor. The photoresistor has greater resistance in dim light and darkness. If the light is bright, the resistance decreases. The photoresistor can be used as a light sensor.

2.8 Kirchoff's Laws

Kirchoff's Laws are useful tools for analyzing circuits. There are two distinct laws, Kirchoff's Current Law (KCL) and Kirchoff's Voltage Law (KVL). This section examines both laws and how they can be used to analyze resistor circuits.

2.8.1 Kirchoff's Current Law

Kirchoff's Current Law can be summarized as follows.

Current entering a node = Current leaving a node

A node is a point in the circuit to which at least two elements, for example resistors, are connected. Intuitively this low makes sense if you consider that current flow is generated by electrons. The electrons flowing into a point in a circuit must come out somewhere; just like water flowing into a pipe must flow out of its other end or out of the pipe branches, if there are any.

To illustrate this point, consider the series resistor circuit shown in Figure 2.6. This is the same as the circuit of Figure 2.3, and we had previously calculated the current in this circuit to be 10 mA. For this circuit, this is the current in to and out of point A, as well as the current in to and out of point B.

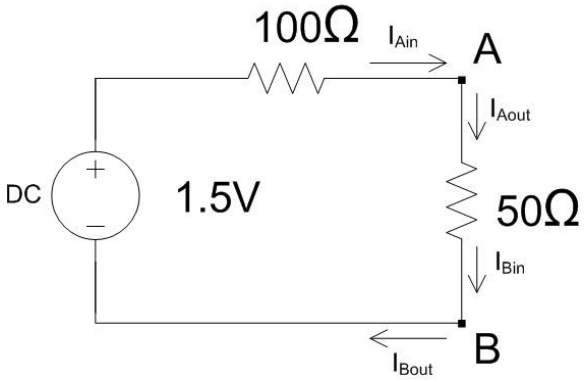


Figure 2.6: Applying Kirchoff's Current Law to a series resistor circuit

Although the KCL is valid for series resistor circuits, it isn't all that useful as an analysis tool. It is much more helpful for analyzing current flow in parallel resistors. For example, consider the circuit shown in Figure 2.7. This is the same circuit as shown in Figure 2.4, and we had calculated its overall current as 45 mA.

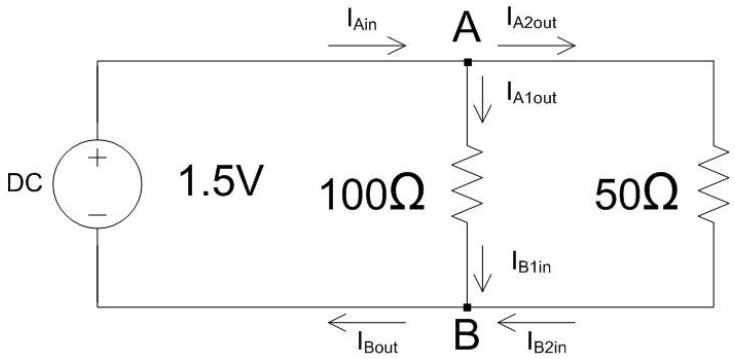


Figure 2.7: Applying Kirchoff's Current Law to a parallel resistor circuit

According to Kirchoff's Current Law, the current entering node A is equal to the current leaving the node, or

$$I_{Ain} = I_{A1out} + I_{A2out}$$

In Section 2.5 we calculated I_{A1out} = 15 mA and I_{A2out} = 30 mA, so

$$I_{Ain} = 45 \text{ mA} = I_{A1out} + I_{A2out}$$
$$= 15 \text{ mA} + 30 \text{ mA} = 45 \text{ mA}$$

Similarly for node B,

$$I_{Bin} = 45 \text{ mA} = I_{B1out} + I_{B2out}$$
$$= 15 \text{ mA} + 30 \text{ mA} = 45 \text{ mA}$$

An important question remains: How do we know how much current flows through each of the parallel resistors? In this example, why did the current split up as 15 mA and 30 mA instead of, say, 40 mA and 5 mA?

Clearly the current values are not selected randomly. In this case there is a straightforward explanation. *When two resistors are connected in parallel, they both have the same voltage drop*. For the circuit of Figure 2.8, the voltage drop across each resistor is 1.5V. Applying Ohm's Law gives us the following current values.

$$I_{A1out} = 1.5\text{V}/100\Omega = 15 \text{ mA}$$
$$I_{B1out} = 1.5\text{V}/50\Omega = 30 \text{ mA}$$

If two resistors are connected in parallel and one resistor has four times the resistance of the other, the larger resistor will have 1/4 the current flow of the other resistor.

2.8.2 Kirchoff's Voltage Law

The second of Kirchoff's Laws, Kirchoff's Voltage Law, is as follows.

The sum of all voltages in a loop is equal to zero

Before examining this law in detail, we first must define a loop. A loop is essentially a closed path within a circuit, consisting of part or all of the circuit. For example, Figure 2.8 shows a series resistor circuit and its one and only loop.

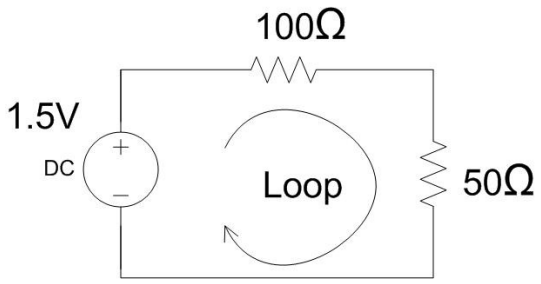


Figure 2.8: A series resistor circuit and its only loop

There are three voltages in this loop: the voltage from the 1.5V power supply and the voltages dropped across each of the two resistors. By convention, we show voltage values as positive or negative based on the direction of the current flow. Since current flows in the same direction as the loop for this circuit, the voltage across each resistor is positive. Since the loop passes from the negative to positive terminal of the power source, its value is negative. The voltage in this loop can be expressed as

$$+V_{100\Omega} + V_{50\Omega} - 1.5V = 0$$

By Ohm's Law, the voltage drop across each resistor is equal to its current multiplied by its resistance, $V = I \times R$. We had previously calculated the current flow through each resistor as 10 mA, so $V_{100\Omega} = 10\text{ mA} \times 100\Omega = 1.0V$ and $V_{50\Omega} = 10\text{ mA} \times 50\Omega = 0.5V$, or

$$V_{100\Omega} + V_{50\Omega} - 1.5V = 1.0V + 0.5V - 1.5V = 0$$

Circuits with resistors in parallel have more than one loop. As shown in Figure 2.9, a circuit with two parallel resistors actually has three loops. The sum of the voltages in each of the three loops must equal zero.

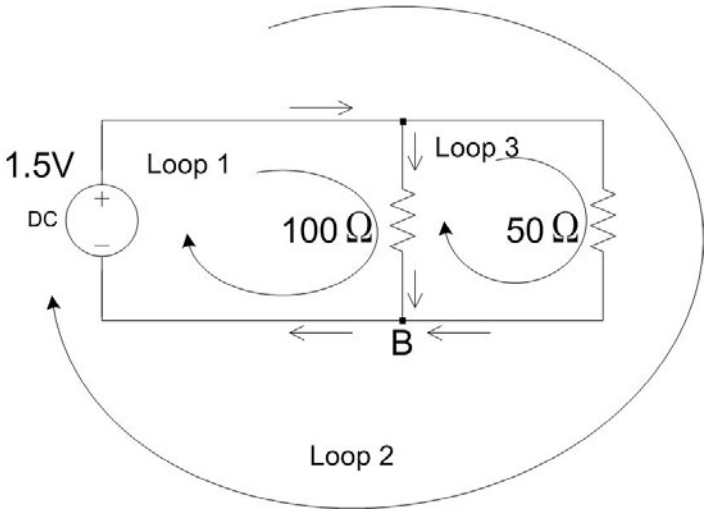


Figure 2.9: A parallel resistor circuit and its three loops

First let's look at Loop 1. It consists of the 1.5V power supply and the 100Ω resistor. We had previously calculated the current for this resistor as 15 mA, so its voltage, calculated using Ohm's Law, is 15 mA × 100Ω = 1.5V. The voltages in this loop are

$$(15\text{ mA} \times 100\Omega) - 1.5\text{V} = 1.5\text{V} - 1.5\text{V} = 0$$

The second loop consists of the 1.5V power supply and the 50Ω resistor. Since this resistor has a current of 30 mA, the voltage equation for this loop is

$$(30\text{ mA} \times 50\Omega) - 1.5\text{V} = 1.5\text{V} - 1.5\text{V} = 0$$

The third loop, consisting of the two resistors, might appear to fail under Kirchoff's Voltage Law since all voltage drops across resistors so far have been positive. If this is true for this loop, we would be adding two positive values and could not obtain a zero result. However, this is not the case here. Note the direction of the arrow for this loop. For Loop 1, the flow of the loop for the 100Ω resistor goes in the opposite direction of the current flow. For this loop, the voltage across this resistor is treated as a negative value, and the loop equation becomes

$$-(15\text{ mA} \times 100\Omega) + (30\text{ mA} \times 50\Omega) = -1.5\text{V} + 1.5\text{V} = 0$$

3 Transistors and Diodes

Transistors and diodes are wonderful devices. They are the basic elements of modern electronic circuits that made possible the rapid development of computer and information technologies we experience today. The TRANSISTOR was invented in 1947 at Bell Laboratories (AT&T), a mere 15 miles from NJIT. Initially, its importance was not fully appreciated, but years later the impact of this invention was recognized by awarding its authors (John Bardeen, Walter Brattain, and William Shockley) a Nobel Price. The full power of the transistor was unleashed with the invention of the INTEGRATED CIRCUIT (Texas Instruments, in 1958), which combines many transistors (a few initially, millions today) in one integrated device. It is this device that literally changed the way we work and live. The inventor of the integrated circuit, Jack Kilby, has been also recognized with the Nobel Price.

Transistors and simpler elements, DIODES, belongs to the class of so called SOLID STATE devices, because they are made of a solid semiconductor material, most of them crystalline silicon (earlier electronics used vacuum tubes). There are two types of semiconductors, n-type and p-type. These symbols stand for negative (n) or positive (p) charges (electrons and holes) that carry electric current in these materials. Electronic devices are made by combining p and n semiconductors We will first examine the simpler device, the diode before moving to more complex transistors.

3.1 Diodes

Diodes, like resistors, have two terminals for connecting to electrical circuit. Unlike resistors, however, diodes are so called NONLINEAR ELEMENTS. It means the current through them is not proportional to the voltage applied across them; they do not obey Ohm's law. Moreover, they have polarity. That is, their terminals can be assigned positive and negative signs since they are connected to p – type and n – type semiconductors, as shown schematically in Figure 6.1.

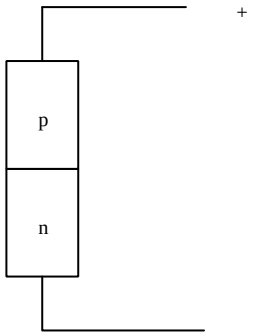


Figure 3.1: A diode as a junction of p – type and n – type semiconductors

When the positive terminal of the diode is connected to the positive terminal of a power source, a current flows with very low resistance. This is called the forward bias of a diode. If the diode is connected in reverse, practically no current flows. The diode acts like a valve, passing the

current in one direction. You may recall a different device, also called a valve, found on bicycle or car tires. The valve there passes the air only in one direction allowing you to pump a tire, while it can not escape in the reverse direction.

The curve representing the relationship between the current and the voltage across a typical diode is shown in Figure 6.2. With the values of current on the vertical axis and voltage on the horizontal scale it is called the I-V characteristic on the I-V curve. The current increases exponentially for the positive voltage values and the graph really shoots up when the voltage exceeds about 0.7 V. For the negative voltage value the graph shows zero current, but only because of the scale on the vertical axis. Expanding this scale would reveal that there is in fact some current for the reverse bias, the so-called reverse current. Only an ideal diode has zero reverse current but in real diodes this current is usually orders of magnitude smaller than the forward current.

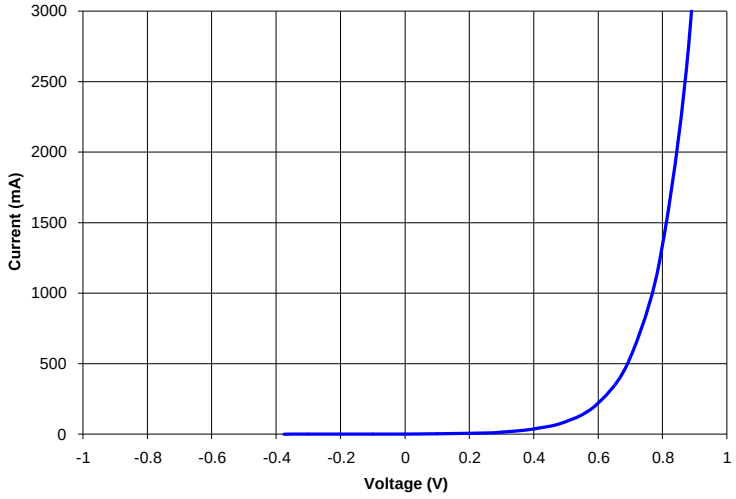


Figure 3.2: The characteristic of a typical diode (I-V curve)

The diodes are very useful in rectifiers, circuits that change AC to DC power. They are used in all electronic devices, such as desktop PCs, which require DC to operate but are supplied by the household current (AC). Battery operated devices, such as quartz watches run on batteries and do not need rectifiers but laptops still need them to recharge their batteries. Diodes can be also used in digital devices to form basic logic gates, as we will see in the laboratory experiments. A special type of diode, the light emitting diode or LED, is used in light communication and display panels. The symbols representing diodes in electric circuit schematics are shown in Figure 6.3. The forward current direction is from left to right. The arrows in the LED symbol represent emitted light.

Figure 3.3: Schematic representation of a diode (left) and an LED (right).

3.2 Transistors

There are two most common transistor types today: the Metal-Oxide-Semiconductor or MOS and the Bipolar Junction Transistor or BJT. The MOS is also designated as MOSFET because it is a field effect transistor (FET). A great majority of both types are made from silicon (Si) and a small fraction (about 2%) from gallium arsenide (GaAs). Depending on the type of the semiconductor materials used in making transistors (n-type or p-type semiconductors) there are n-type or p-type MOSFETs or n-p-n and p-n-p BJTs. The BJT dominated the market initially but now most of the transistors, particularly in integrated circuits, are of the MOS type. The BJT still holds its own, particularly in some analog and high power circuits.

The big difference between transistors and diodes is that transistors have three terminals, not two. The additional terminal controls the current flowing through the other two. This creates important functional possibilities, namely, one circuit can control (or switch on or off) the current flowing in another circuit. Another important feature of the transistor is that the controlling current can be much smaller than the current that is controlled. This leads to amplification, a very useful effect in devices such as stereo amplifiers or the control of the speed of electric motors.

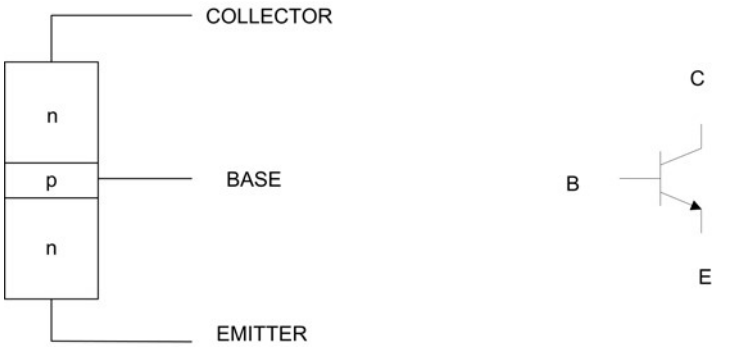


Figure 3.4: The structure of an n-p-n BJT (left) and its representation in electrical schematics (right)

The structure and the schematics of connections to a BJT are shown in the Figure 6.4. An np-n type transistor consists of two n-type regions (the emitter and the collector) with a thin p – type layer (the base) between them. Note that the transistor consists of two p-n junctions connected by the common p-type base. In normal operation the collector is positive and the current flows between the collector (C) and the emitter (E) while the base (B) is the control electrode. There are also less often used p-n-p transistors with opposite types of semiconductor layers. In MOS transistors the electrodes have different designation: source instead of emitter, gate instead of base and drain instead of collector. There are also n-type and p-type MOSFETs. In the n-type, the current flows between source (S) and drain (D), while gate (G) is the control electrode.

Most transistors of any type are made today as elements of integrated circuits (ICs), which can contain many millions of circuit elements. Such a combination of many interrelated circuits can result in an extremely complex and powerful system such as the Pentium microprocessor. Single or discrete transistors, however, are still useful in many applications such as high frequency or power units.

4 Simulation Tools

In this course we will use computer programs to simulate the operation of electrical circuits. We will run two types of software: educational and engineering programs. The purpose of educational programs is to help students understand the principles of a given discipline and to set up simple virtual experiments in which various parameters (such as resistance or voltage) can be easily and quickly changed and the results of "measurements" obtained immediately, without actually building the real system in laboratory. This is a great learning tool and fun to use. Do not believe, however, that playing well with the software will make you a good engineering practitioner. You still need to go to the lab and work with real components and instruments. We live in the real world and here things are usually harder than they look on the screen and just mouse clicking is not enough. You may even find that solving real problem in real world is more satisfying.

We are lucky that a very useful (and fun to use) program for teaching physical principles is available free on the internet at http://phet-web.colorado.edu/web-pages/index.html. PHET stands for Physics Education Technology and consists of the main program PhET SimLauncher and a number of different modules in the form of Java applets. These animated programs cover a wide range of topics from simple physics principles (such as motion and energy) to advanced concepts of modern science. Many were developed by prominent scientists and educators. You may want to explore them in your free time but in this course we need only three modules:

- Battery resistor circuit
- Ohm's law
- Circuit construction kit (DC only) virtual lab version

PHET can be run online but you may want to install the program on your computer and run it without an Internet connection. Following the instructions on the web page, you can download the main program PhET SimLauncher. If you do not have Java installed on your PC use option: Windows PC that does not have Java 1.4 or later on it. After installation Launch Phet; it will show that no modules are installed and guide you to a module installation site. Download the three modules listed above. They are already installed on the computers in our laboratory.

Professional engineering simulation software packages have been developed to assist engineers and scientists in solving design problems using a computer. These software packages rely on mathematical models of circuits (PSpice developed by Microsim, purchased by ORCAD, which in turn was purchased by Cadence Design Systems, and Multisim created by a company named Electronics Workbench, which is now a division of National Instruments), or on the usage of a programming language (Matlab from Mathworks, Mathcad from Mathsoft) to facilitate the resolution of mathematical problems in circuit analysis. The three packages that you will be exposed to in a very limited fashion will be PSpice, Multisim, and Matlab. The students will be shown a limited set of the features of these software packages to have an idea of their capabilities. These and other software packages will be utilized more extensively in the future, in the various courses that require the assistance of these software tools and the students will see how powerful

these packages can be in supporting the design of engineering systems, most notably in electrical and computer engineering.

PSpice is available free of charge as a very limited version of the professional package. It can be downloaded from the ORCAD website, http://www.orcad.com/downloads/form/cdrequest.asp; ORCAD is the parent company of Microsim. An older version of the demo package will be available on the FED webpage. Matlab is available free of charge as part of the software package distributed to the students when they join NJIT.

4.1 Circuit Examples

We will initially analyze on a sheet of paper a circuit that is purely resistive (made of energy sources and resistors only). The students are advised not to be scared by the complexities introduced. The material will become familiar in subsequent courses. The purpose of these choices is to show you how powerful and useful the software packages are. You will however easily master these cases and much more complex ones by next year.

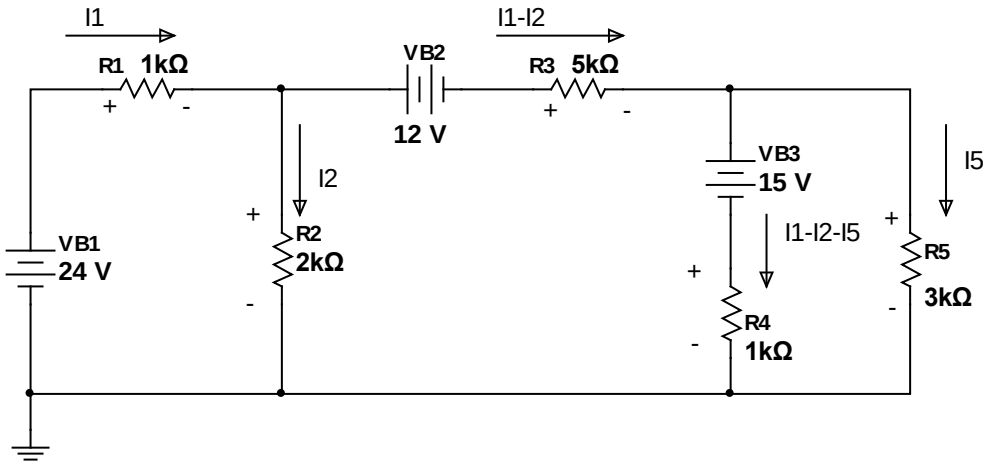


Figure 4.1: Complex Resistive Circuit

We are going to use KVL (Kirchhoff' Voltage Law), KCL (Kirchhoff' Current Law), and Ohm's law, which will be covered more extensively in ECE 231, to obtain some of the currents and voltages in the circuit.

Left loop: $- V_{B1} + R_1\mathbf{I_1} + R_2\mathbf{I_2} = 0$

Center loop: $-R_2\mathbf{I_2} + V_{B2} + R_3(\mathbf{I_1}-\mathbf{I_2}) + V_{B3} + R_4(\mathbf{I_1}-\mathbf{I_2}-\mathbf{I_5}) = 0$

Right loop: $-R_4(\mathbf{I_1}-\mathbf{I_2}-\mathbf{I_5}) - V_{B3} + R_5\mathbf{I_5} = 0$

Rearranging these three equations, we obtain

$$R_1\mathbf{I_1} + R_2\mathbf{I_2} = V_{B1}$$

$$-(R_3 + R_4)\mathbf{I_1} + (R_2+R_3+R_4)\mathbf{I_2} + R_4\mathbf{I_5} = V_{B2} + V_{B3}$$
$$-R_4\mathbf{I_1} + R_4\mathbf{I_2} + (R_4+R_5)\mathbf{I_5} = V_{B3}$$

We will realize that when Matlab is introduced in a subsequent section, the solution to this linear system of equations will be a very simple task.

4.2 PSpice

PSpice is a PC version of the SPICE program (Simulated Program with Integrated Circuit Emphasis). It will be used to simulate an electrical system, and obtain values or graphical plots of some of the responses (entities that we may be interested in such as currents or voltages through time, or frequency domain characteristics).

Much of this tutorial on PSpice will be presented through examples from which the students can extract patterns that they can apply to other cases, or be able to extend their learning capabilities to other features of these packages. The Windows operating system and many other packages before it have shown the appeal that a graphical interface has for the user. PSpice had to go that route to compete with other software packages that understood the need to develop user-friendly software packages. Since this is a demo version of PSpice, its features may be severely limited, but nonetheless adequate for this course and even some subsequent courses. The major advantage is that it is free.

4.2.1 PSpice Installation

Once you download the file named **91pspstu.exe**, double click on it to execute the program. The program will unzip all the necessary files in a temporary folder which I have renamed C:\Users\Mo\PS. It is obvious that your user name may not be Mo, and you can create any folder under your user folder or anywhere in your hard drive.

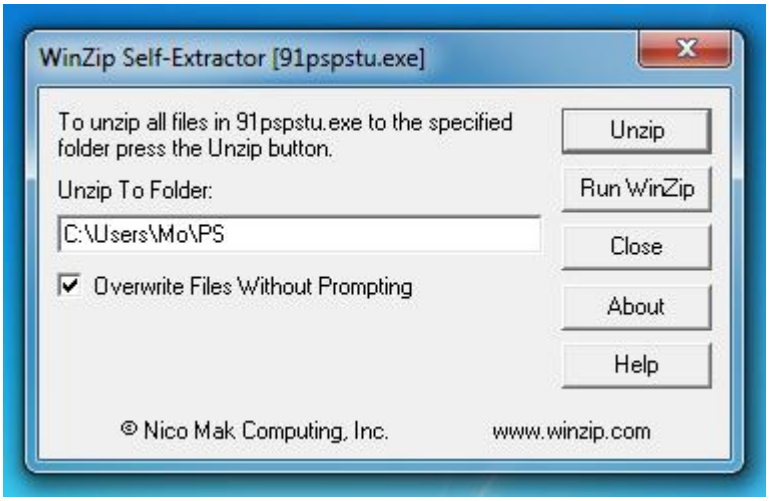


Figure 4.2: Winzip Screen

In the PS folder, execute the file Setup.exe, and the following screen will appear after a screen which requires your approval of the installation and another one related to turning off your antivirus software (you can ignore this one). Before you continue make sure you mark the schematics option in the window

Figure 4.3: Selection Screen

4.2.2 *Getting started*

From **Start, All Programs, Pspice Student, Schematics**, have the PSpice program running in the **Schematics** mode.

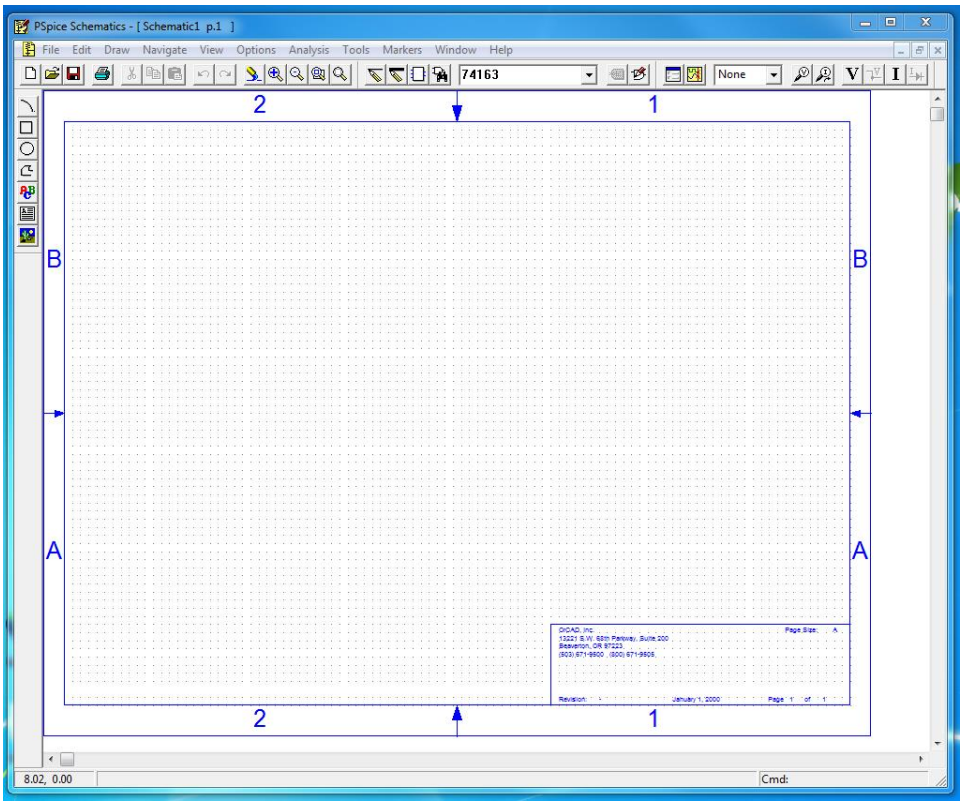


Figure 4.4: PSpice Opening Screen

4.2.3 Simple resistive circuits

We will first revisit the two circuits from chapter 2, shown in Figures 2.3 and 2.4.
We will also learn to use PSpice through these examples.

Figure 4.5: Toolbar from opening screen

To draw a circuit you can either

- go to the menu **Draw, Get New Part** (equivalent to **Control-G**),
- or use the button **Get New Part**  ,

You can then either type in the name of the part that is needed, or obtain from the long list, or open the libraries which will give you a categorized listing of the components. The components are **VDC**

for the dc voltage sources or batteries (source library), **AGND** for the ground (port library), **R** for resistors (analog library). You can use the button **Draw Wire**  to connect the various components. You can also use **Control-R** to rotate components when highlighted (use the left button of the mouse for highlighting components) if needed. You can also add from the parts library **Viewpoint** (special library) to display a nodal voltage at any given node referenced to the ground node. The ground node may not, as in this case, be part of the original circuit. However PSpice needs a reference node labeled as a ground node. All the voltages are referenced to it.

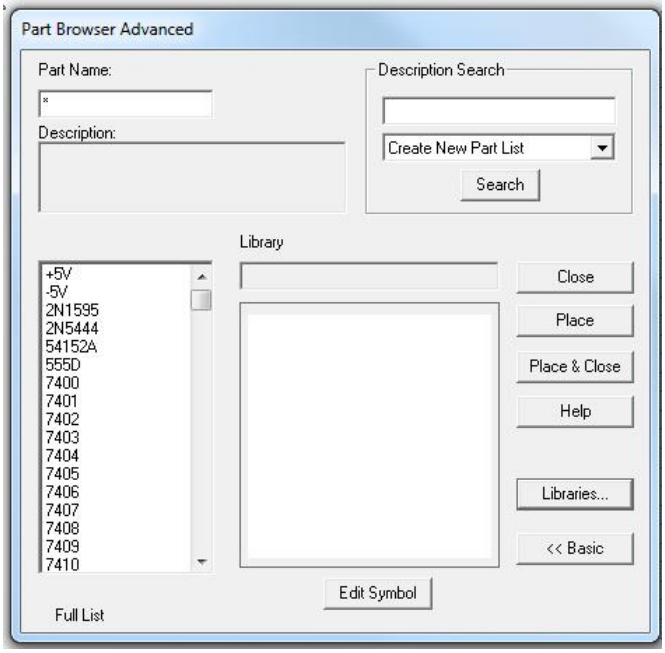


Figure 4.6: Part Browser Screen

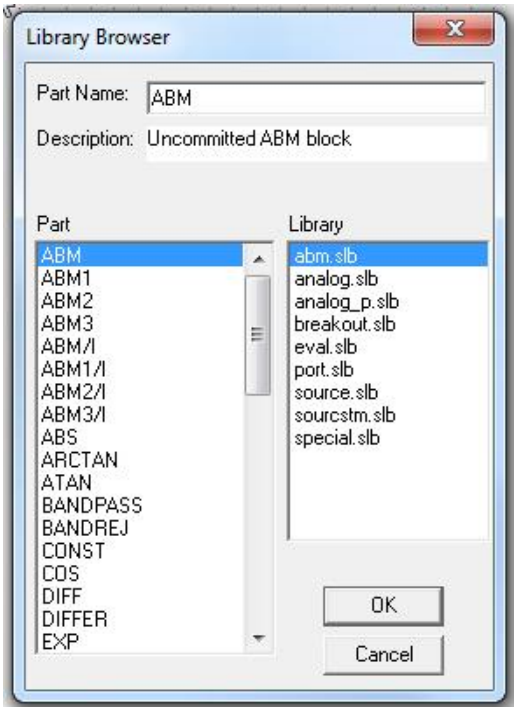


Figure 4.7: Library Browser Screen

Once you select a component and press place you can keep placing as many times the same component until you click the right button. When you select the Draw Wire button, you click the left button of the mouse when the cursor is on top of one of the terminals, and drag the cursor until you reach another node that you need connected and then click the left button again.
A quick demonstration in class will also show you what you need to do when some of these tasks give you trouble, or need to add your own way of doing things.

The Iprobe needs to be inserted in a circuit just like an ammeter is. Though you could have used the following buttons

to read every nodal voltage and every component current, in most cases, that is overkill as only few of those entities are of interest, aside from the fact that too many visual displays of data can overwhelm the user.

Simulation can be executed through **Analysis – Simulate** or clicking on the following button

Note that the measurements shown in Figure 4.8 and obtained through PSpice match exactly the calculated values derived in chapter two.

The same task can be done for the parallel circuit shown in Figure 4.9.

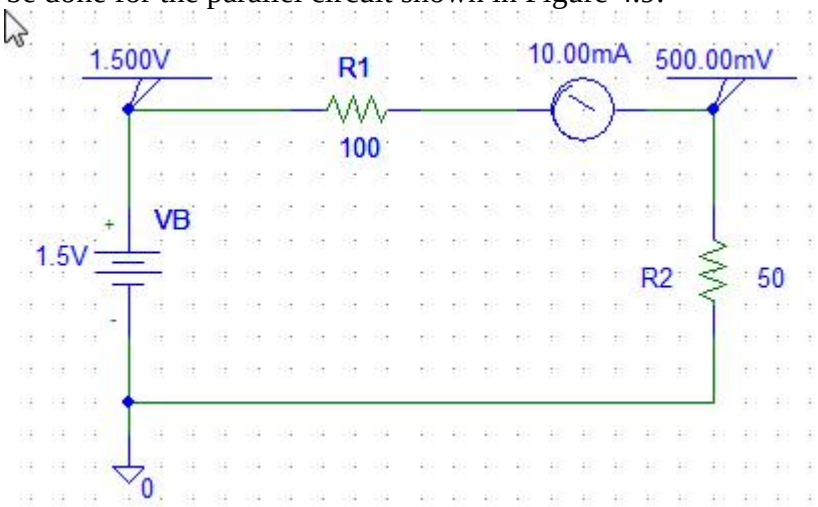


Figure 4.8: Series Circuit Simulation

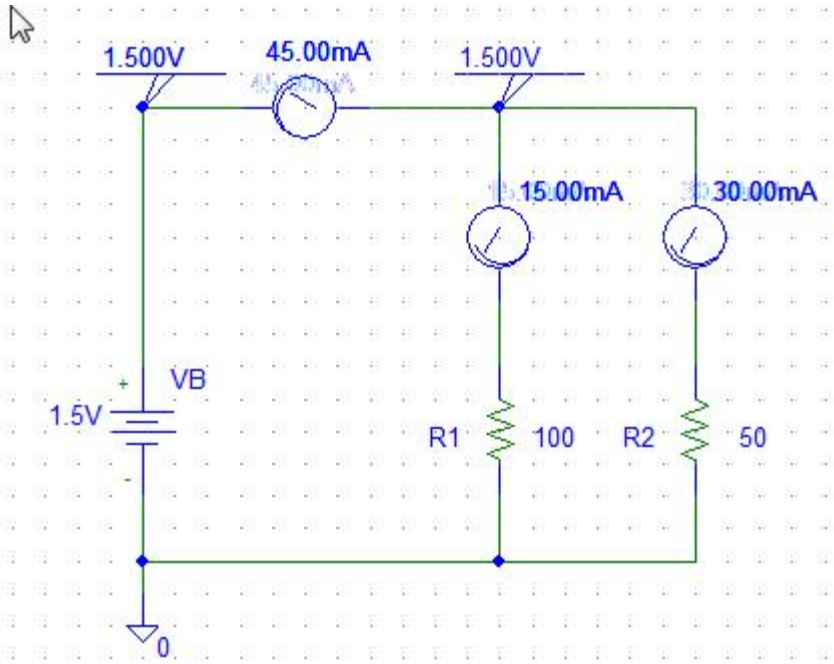
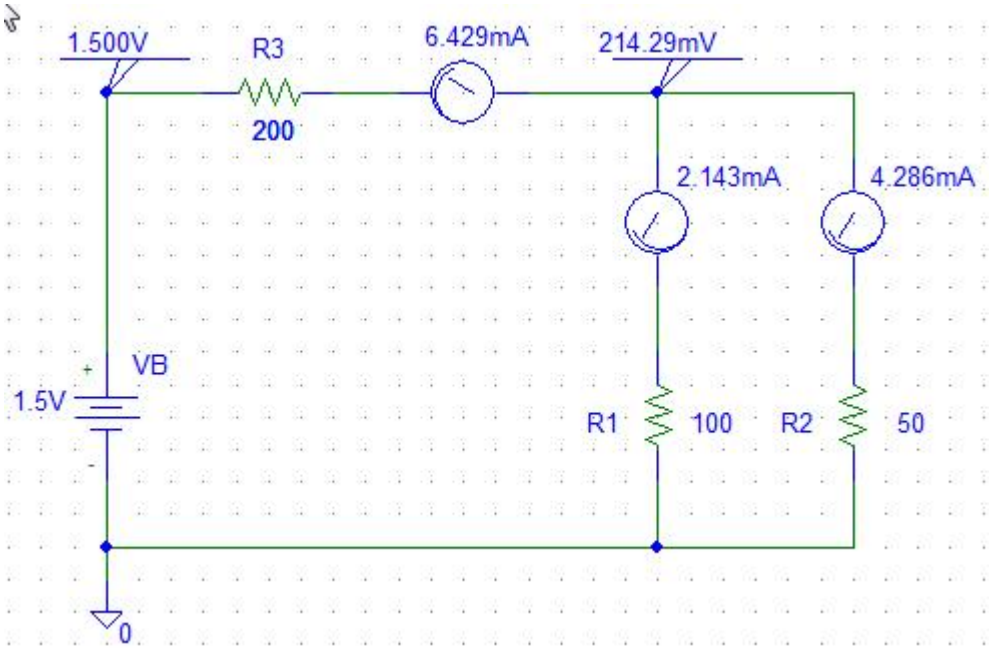


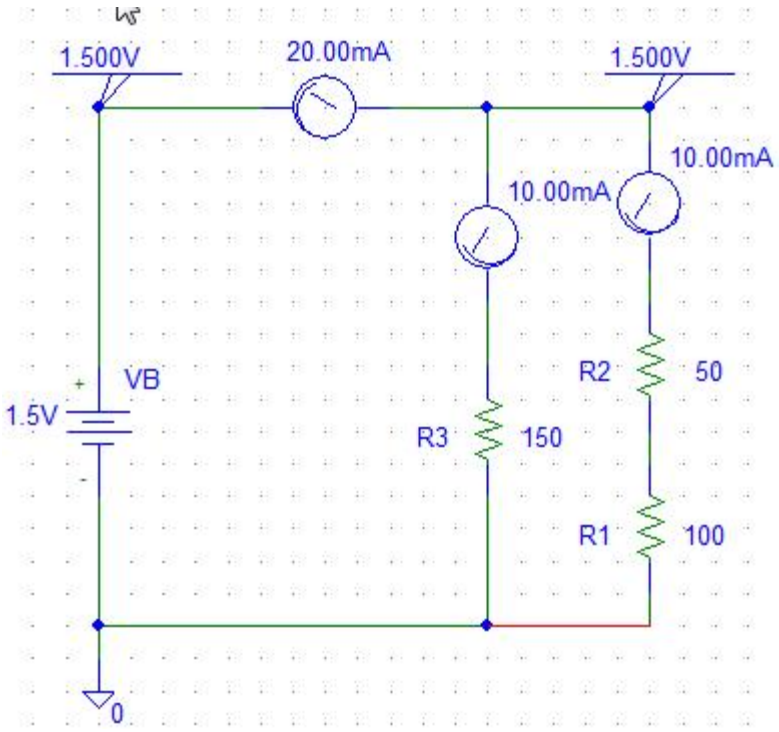
Figure 4.9: Parallel Circuit Simulation

4.2.4 More complex resistive circuit

The circuits shown in Figure 2.5 (a) and (b) are simulated to confirm the validity of the results obtained in that section.



(a)



(b)

Figure 4.10: Two series-parallel circuits

We return to the circuit shown in Figure 4.1 and determine with ease the unknown responses, in this case the currents I_1, I_2, and I_3, and the voltage across the resistor R_3 which we label V_3. Figure 4.11 shows the simulated values of those responses.

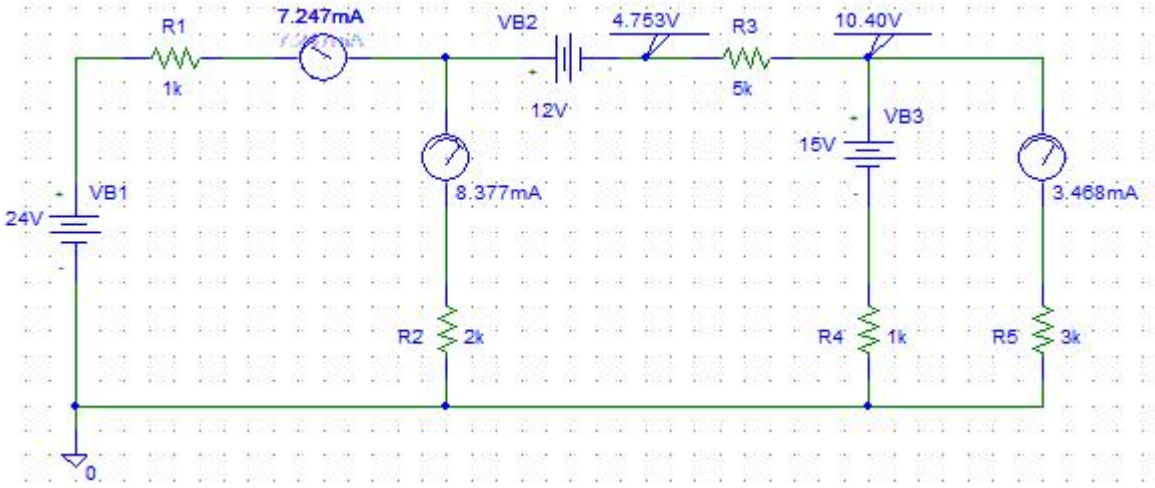


Figure 4.11: Complex Circuit Simulation

We will verify through Matlab that the calculated values will match the ones obtained through PSpice.

4.3 Multisim

The Multisim program (also based on the SPICE engine) is installed on many computers and definitely the computers used in your labs. A student version can be purchased by the students at a good price. The student version is powerful enough to handle all your requirements in all courses related to circuitry or electronics. After you click **Start-All Programs-National instruments-Circuit Design Suite 11.0-Multisim**,

Figure 4.12: Starting Multisim

you will be seeing the screen shown in Figure 4.13.
As of this writing, Multisim 11 is the one you will be working with even though the examples given in this manual were obtained through Multisim 10. We will revisit the circuits shown in Figures 4.8, 4.9, and 4.10. We will also learn to use Multisim through these examples. It will be very easy for the students to extend that learning to other features available in Multisim through their own usage of the software package.

It will be up to the students to decide how many toolbars they want displayed. The most popular toolbars are Standard, View, Main, Components, Simulation Switch and/or Simulation, and Instruments. Note that the instruments toolbar is located in the right section of the screen.
In the components toolbar, the buttons that you will be working with are **Place Source** (ground, sources), **Place Basic** (resistors, and capacitors), **Place Diode**, **Place Transistor**, **Place TTL**, **Place Mixed**, and **Place Indicator**.

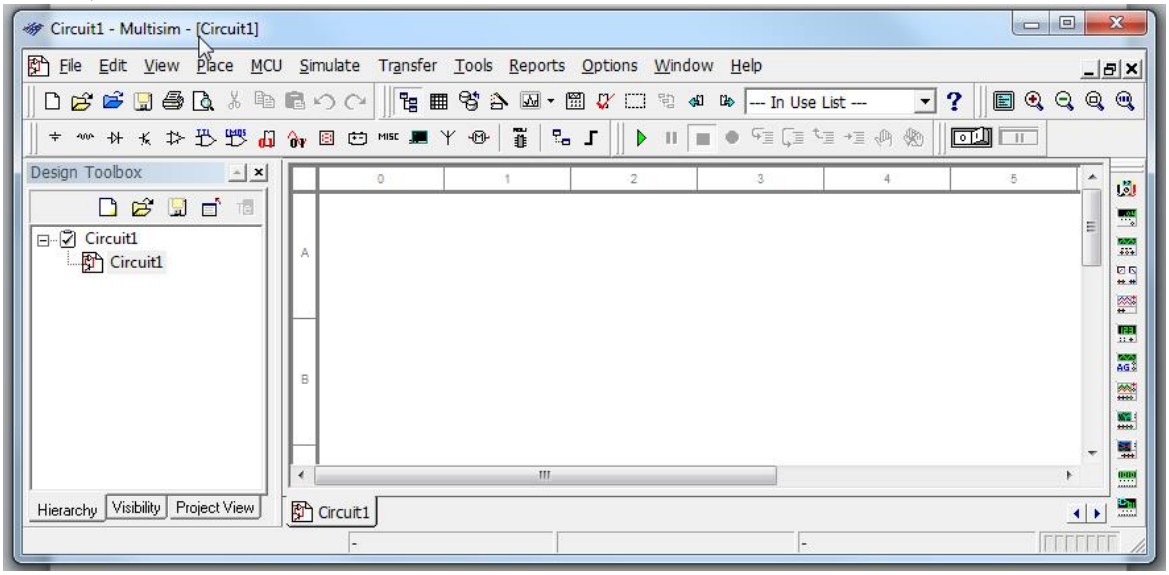


Figure 4.13: Starting Multisim Screen

If you want to select ground or a battery, you click on the **Place Source** button. The screen shown in Figure 4.14 will appear on the monitor from which you can select the desired components. When finished you can close the screen and click on another component button, or you can click on the '**Sources**' window and other libraries will show immediately. You can also display the grid (**View-Show grid**) if you want to have an easy way to line up your components.

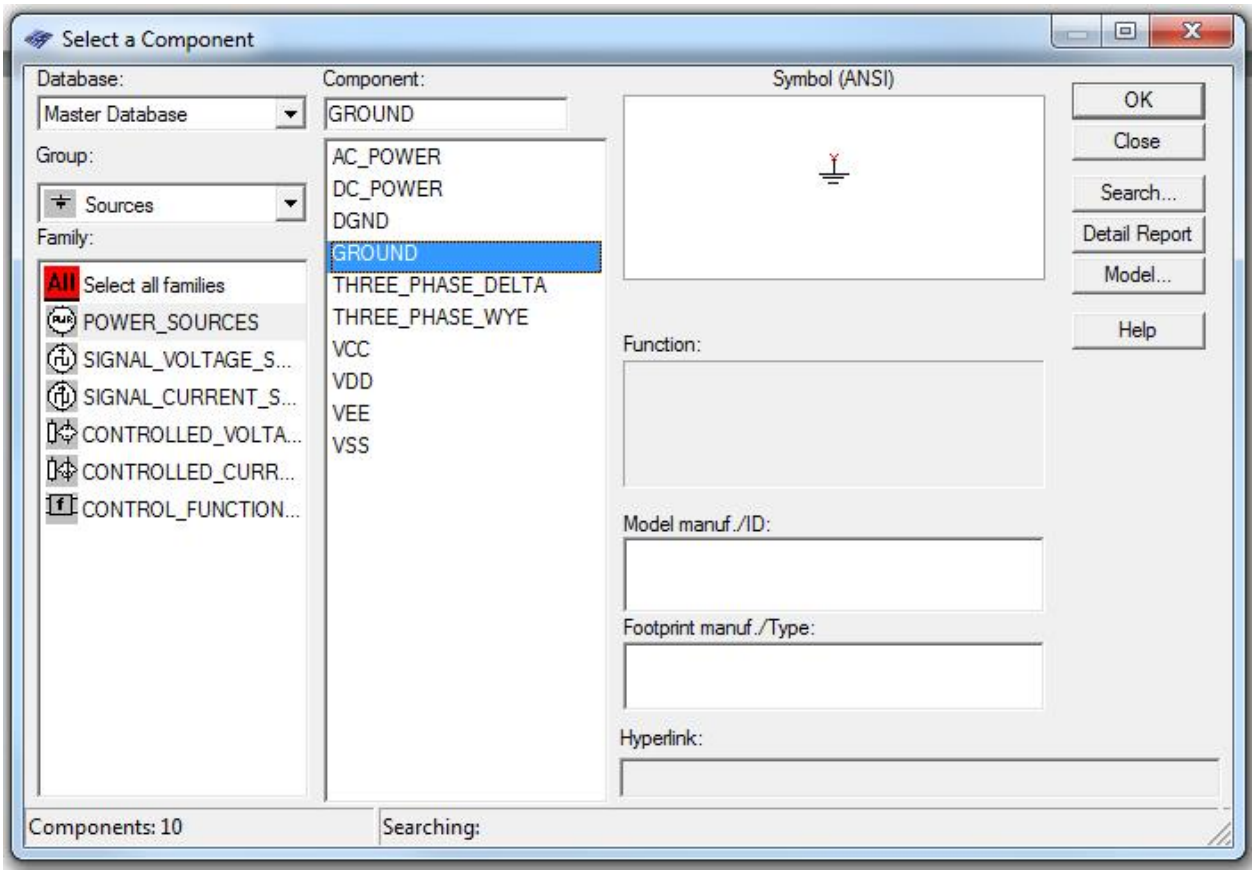


Figure 4.14: Selection from the Sources Library

Let us now look at some of the differences between Multisim and the demo version from PSpice. (To be fair, the demo version has been available for over 2 decades, and it cannot compete with a software package that is more recent. It is most likely that the more current version of PSpice would be a lot more powerful than this demo version.) To connect components together you just move the cursor over one terminal in which case it changes to a different shape (bullet with crosshairs) then click the left mouse button and move the cursor to the desired node and then click again on the mouse left button when you reach it. The instruments (voltmeter, ammeter) are similar in function to the ones you would use in the lab (some choices are related to replicas of Agilent instruments).

You can also click on the mouse right button to have access to a lot more features and operations. Simulation can be executed by pressing the switch or clicking the green play button.
The Figures 4.15 through 4.18 show the Multisim simulation of the circuits presented in the PSpice section.

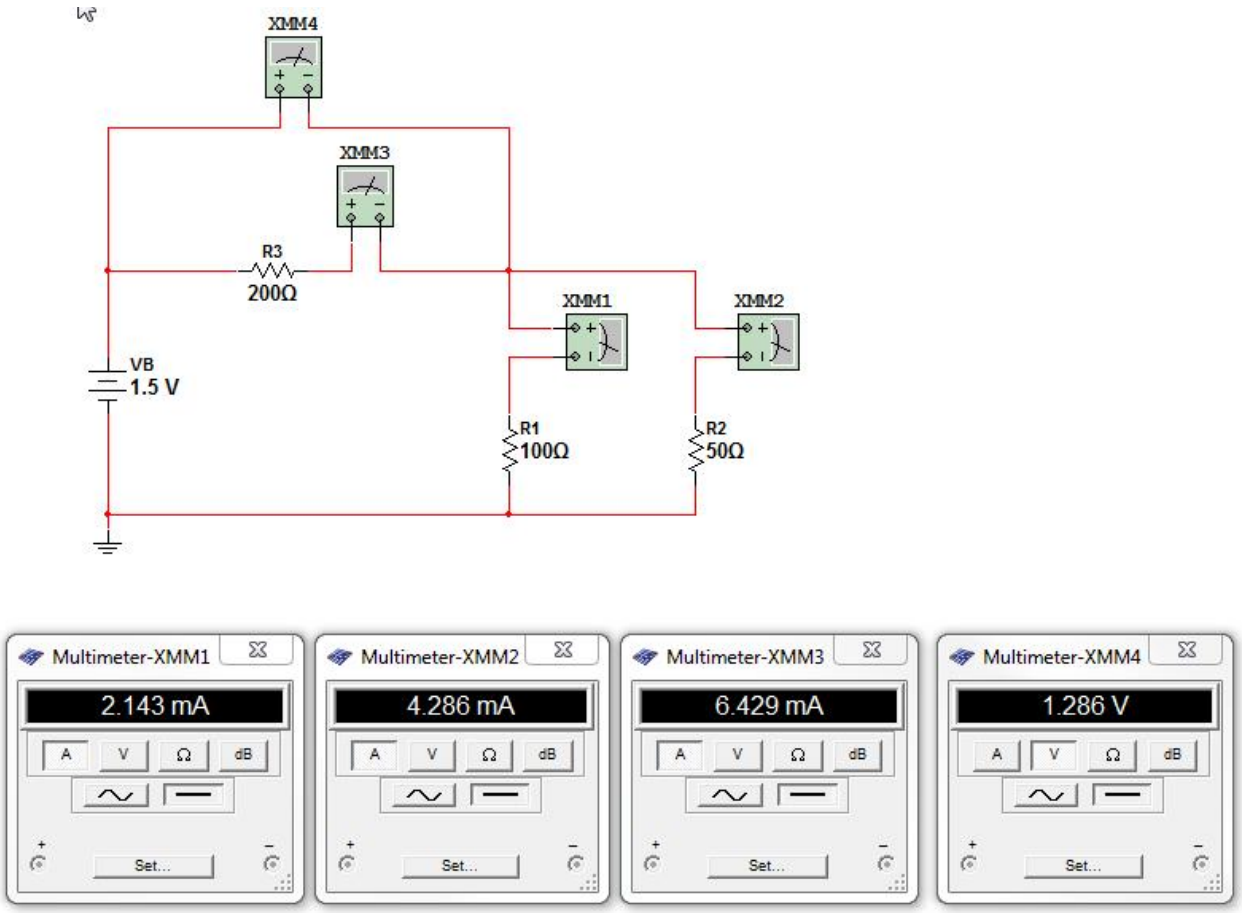


Figure 4.15: Simulation of the series-parallel circuit (a)

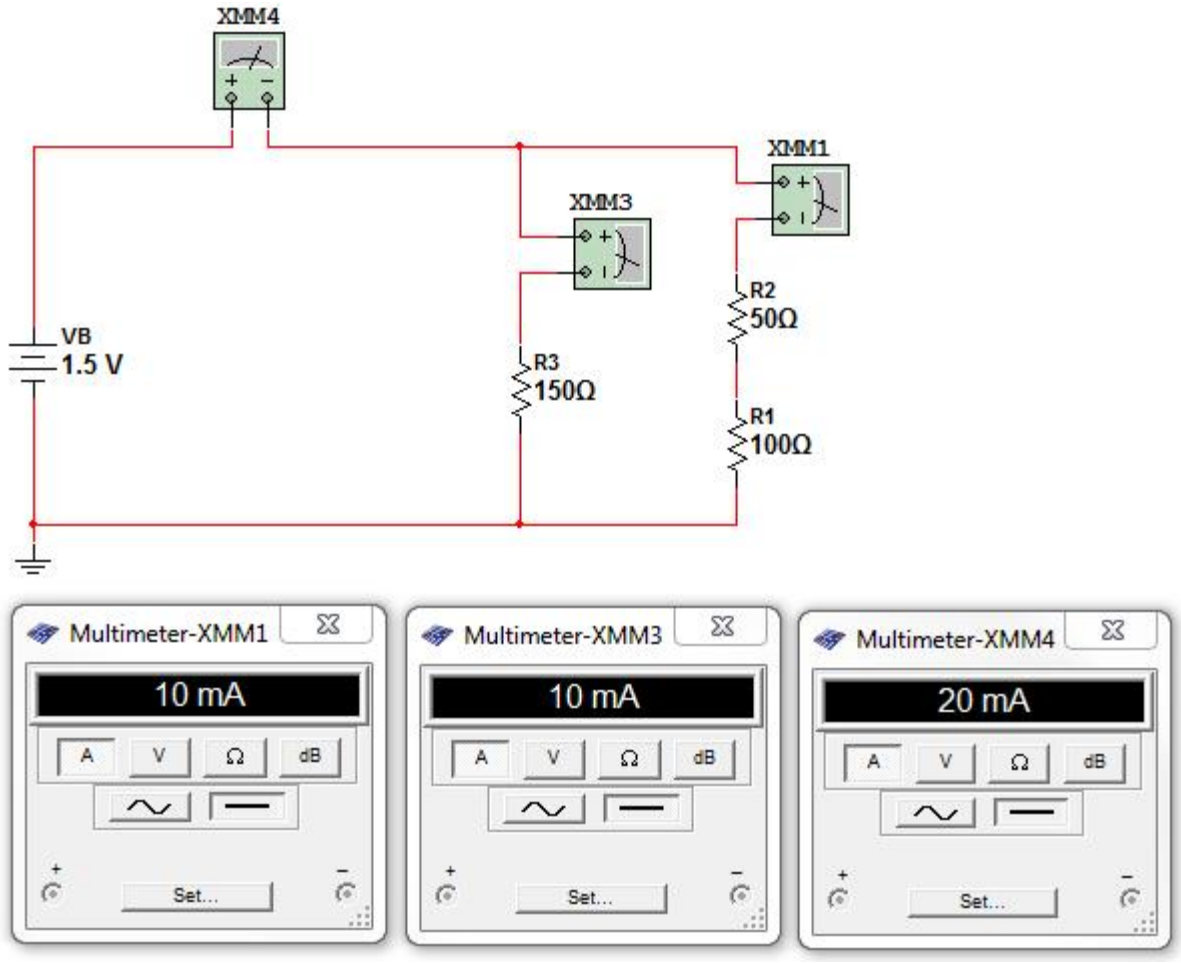


Figure 4.16: Simulation of the series-parallel circuit (b)

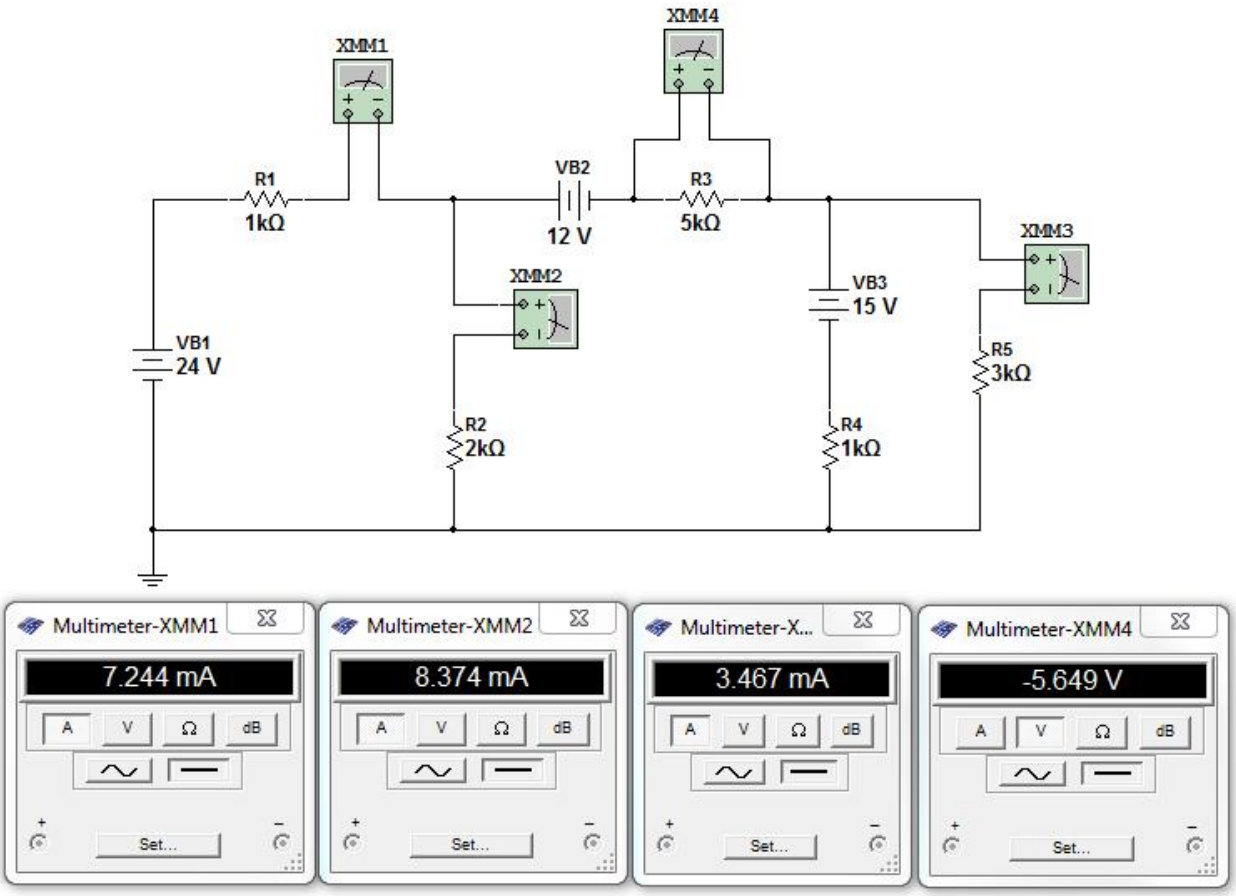


Figure 4.17: Simulation of the complex circuit

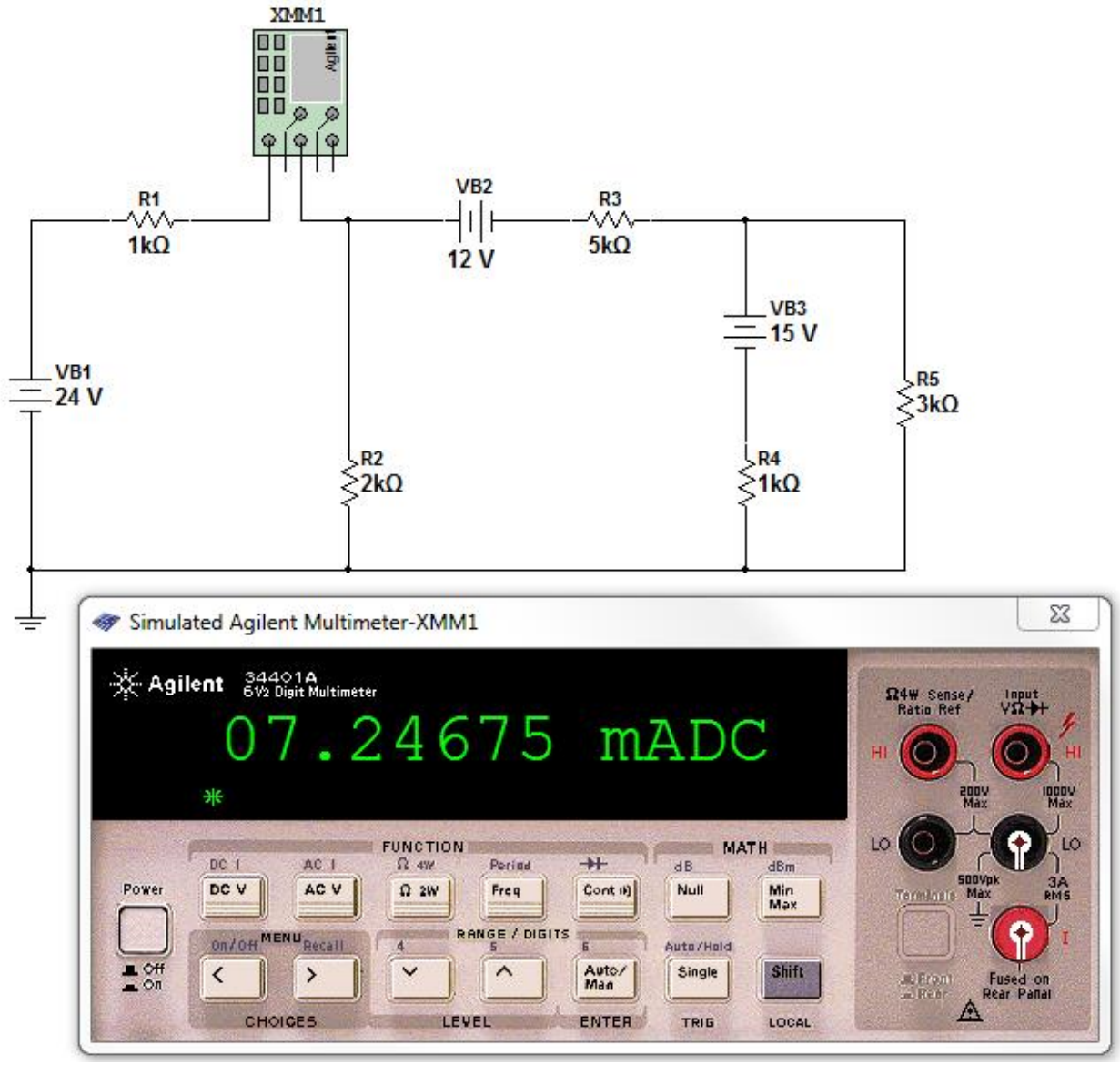


Figure 4.18: Simulation of the complex circuit with Agilent Instruments

4.4 MATLAB

Matlab has become a de facto standard programming language in industry and universities, where scientists and engineers are involved in systems analyses and design.

4.3.1 Getting started

Launch Matlab through the following sequence of steps: **Start > All programs > Matlab > R2013b > Matlab R2013b** or whatever the latest edition that is installed in the lab computers. Multiple windows will be opened by Matlab. The window of interest to us at this point is the command window (center window). In that window, the students will be able to type in the recognizable commands that Matlab will execute. The line command always starts with '>>'.

The students are encouraged to type in 'demo' at the command prompt so that you can see some of the capabilities of Matlab and some of the code that Matlab relies upon to perform the required tasks.

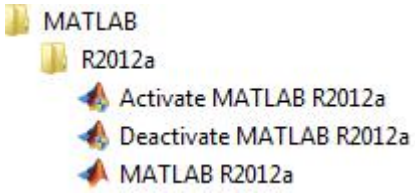


Figure 4.19: Starting Matlab

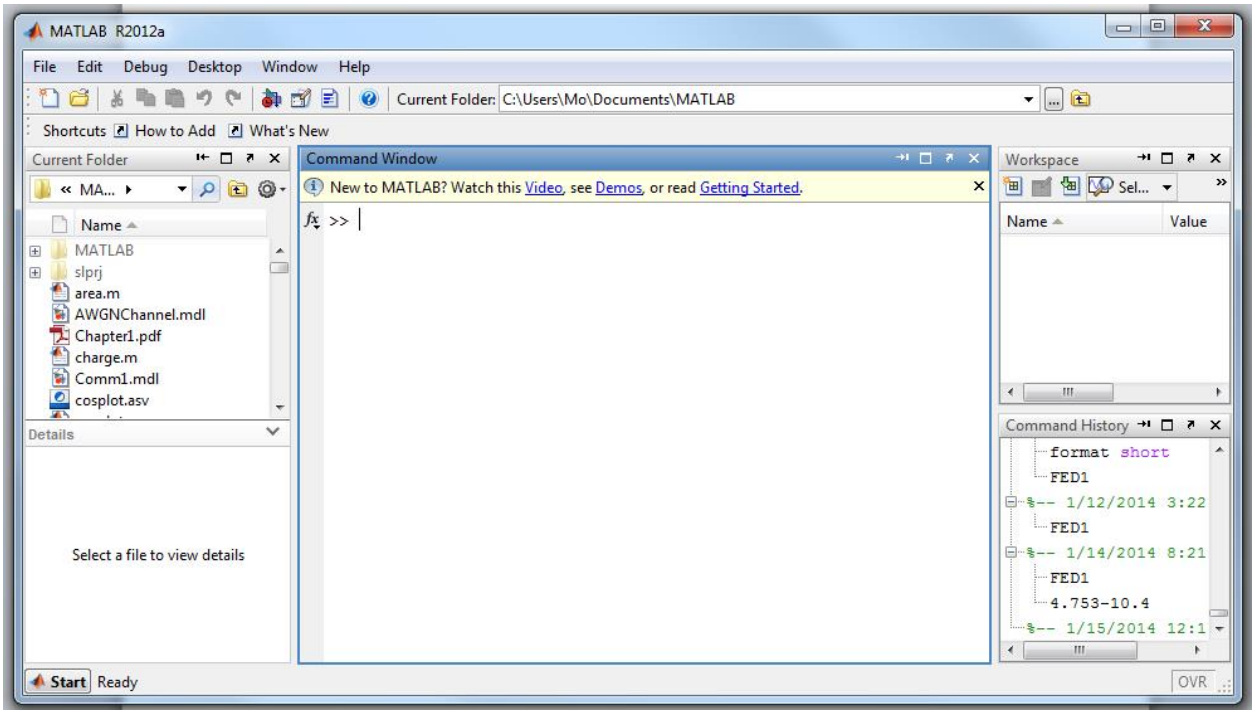


Figure 4.19: Matlab Starting Screen

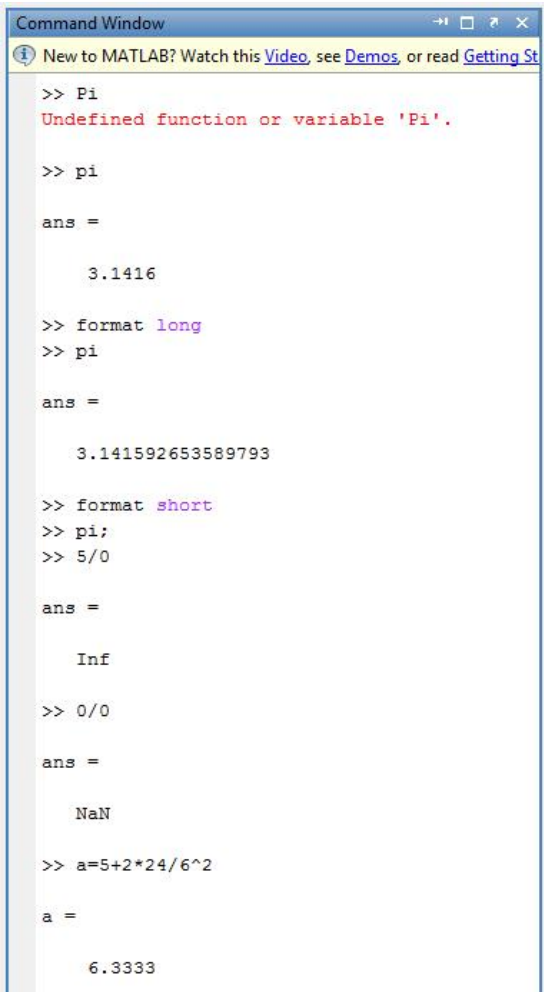
4.3.2 Simple Matlab Tutorial

It is important to remember that Matlab is case sensitive. Whether you are naming a variable, constant, a file, or a command, Matlab will recognize the lower case and the upper case of a letter as two different entities. (Even if the Windows operating system may not distinguish between file and File, Matlab does.) The default naming of the displayed result will be **ans** unless you override that case with your own choice. You can recover earlier typed commands using the up or down arrows (or copy and paste from the Command History window), and use the left and right arrow for moving the cursor for editing purposes. The user can clear the command window by typing **clc** at the prompt.

1. ***Arithmetic operations***

The screen shown in Figure 4.20 exhibits examples of some basic arithmetic operations. Note that if you want to suppress the display of the result of any command, you terminate the command with a semicolon.

The default format for displaying numbers is based on 4 digits after the decimal point (called **format short**). If you type in **format long**, the results will show 16 digits after the decimal point. Some of the constants that you may be familiar with are 'pi', Inf for infinity, and NaN for operations that are indeterminate such as 0/0.



2. ***Elementary Math*** ons

Some of the functions (like **sqrt** or **exp**) are built-in functions that cannot be changed, but other functions exist as m-files (through Mathworks or could be user-defined). Some of these functions are listed in Table 4.1 and examples of using them in Figure 4.21
Note that sin(x) expects x to be in radians whereas sind(x) involves x in degrees. The same is applicable to asin(x) and asind(x).
The complex operator is accepted as i or j (though electrical and computer engineers prefer to use j as i is usually used to represent currents).

exp(x)	Exponential; e^x.
sqrt(x)	Square root; $\sqrt{x}$.
log(x)	Natural logarithm; ln *x*.
log10(x)	Common (base-10) logarithm; log *x* = $\log_{10}x$.
abs(x)	Absolute value or magnitude of a complex number; *x*.
angle(x)	Angle of a complex number *x*.
conj(x)	Complex conjugate.
imag(x)	Imaginary part of a complex number *x*.
real(x)	Real part of a complex number *x*.
sin(x); sind(x)	Sine; sin*x* (*x* in radians or degrees respectively)
cos(x); cosd(x)	Cosine; cos*x* (*x* in radians or degrees respectively)
tan(x); tand(x)	Tangent; tan*x* (*x* in radians or degrees respectively)
asin(x); asind(x)	Inverse Sine; arcsin*x* (result will be in radians or degrees respectively)
acos(x); acosd(x)	Same with Inverse cosine
atan(x); atand(x)	Same with Inverse Tangent

Table 4.1: Elementary Mathematical Functions

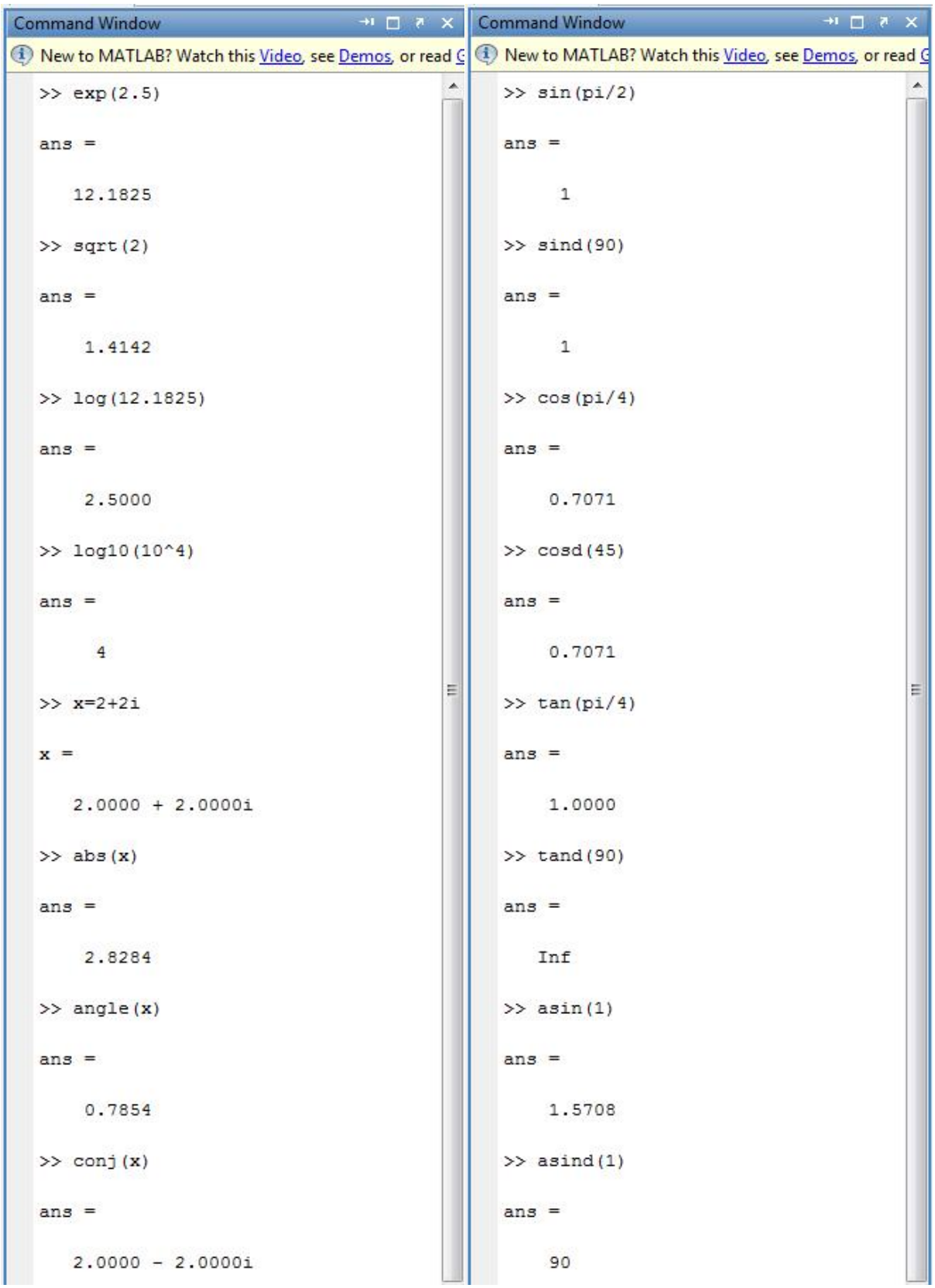


Figure 4.21: Samples of Elementary Mathematical Functions

3. Arrays, Polynomials, and Plotting

Arrays can be a row vector, a column vector, or a 2-dimentional array. Figure 4.21 shows an example of these arrays. **a** is a row vector; **b** is a column vector and **c** is 2-dimentional array. **d** is also a row vector but note that the elements of **d** follow a pattern. The first element is the minimum 2 and the other elements are built based on the increment 3 all the way up to but not exceeding the maximum value 36. Even though the display of the array may occupy multiple rows (due to the number of elements, the size of the monitor, and the number of elements that can be displayed in a single row), **d** is still a row vector. If the increment is 1, it can be omitted.

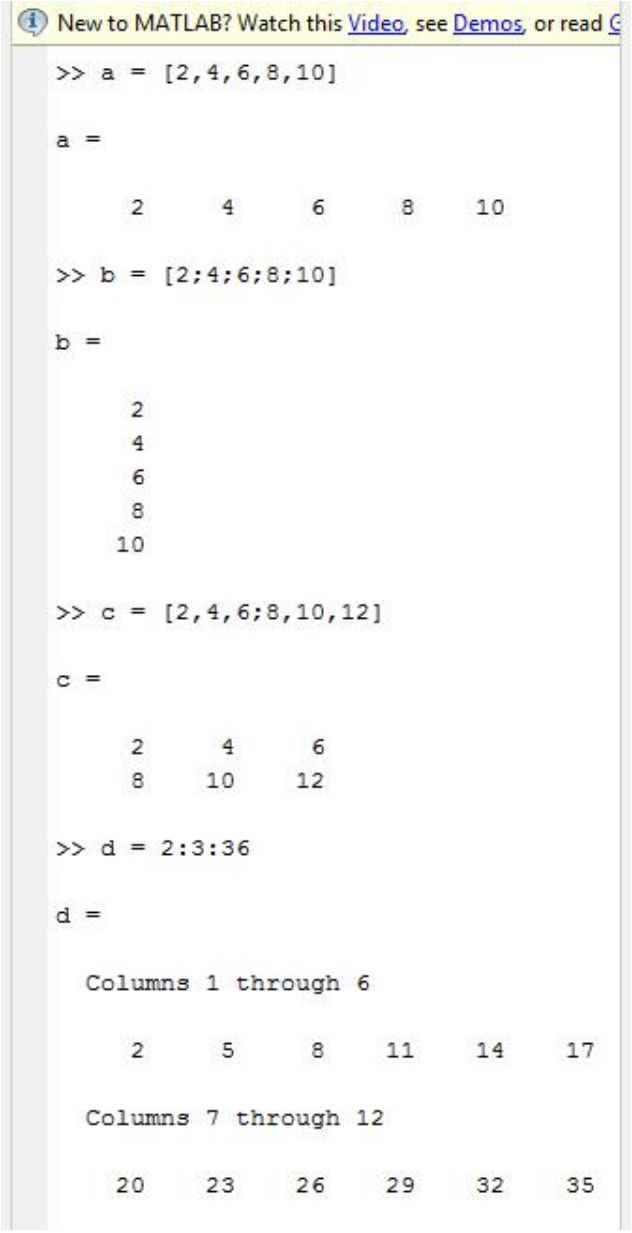


Figure 4.22: Arrays Sample Display

Many Matlab functions rely on the coefficients of a polynomial to perform operations such as the evaluation of the polynomial or the acquisition of a zero or an extremum. For example, given the

polynomial $f(x) = x^4 - 10x^2 + 5x - 1$, the array representing the coefficients of the polynomial is p = [1, 0, -10, 5, -1]. It is clear that the third degree term is absent because its coefficient is zero. For an n^{th} degree polynomial, the array must be made of n+1 elements.

When we need to execute many instructions for the purpose of a given goal, it would be more efficient to create a **script** file made of all the commands needed. If changes are made to some commands, one only needs to call that file once and all the commands will be executed in the order they appear in the script file.

To create the script file press file-new-script or go to the toolbar and press script file, and the file editor will open where one can type all the required commands. Save the file preferably with an m extension (name.m) where the name should be chosen according to Matlab requirements.

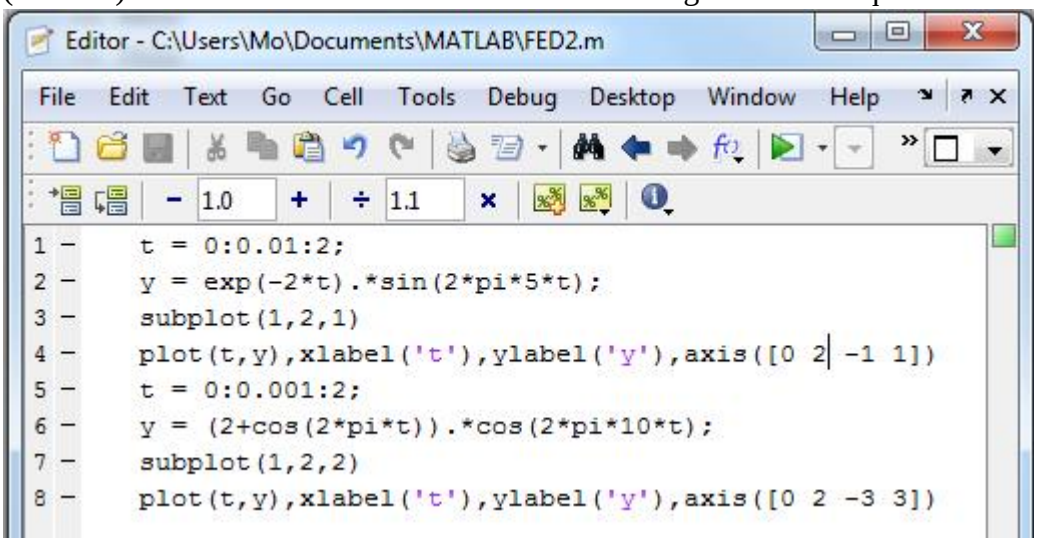


Figure 4.23: Plotting Example Script File

Figure 4.23 shows an example of a file written to exhibit some of the basic plotting capabilities of Matlab. Explanations for the different command are as follows:

- Line 1 sets the range for t for the first plot
- Line 2 establishes the first function to be plotted
- Line 3 divides the plot window into 1 row and 2 columns of panes, and sets the first plot to be displayed in the first pane
- Line 4 shows the plot of y versus t, with the horizontal axis labeled t, the vertical axis labeled y, and the display of the plot over the ranges 0 and 2 for t, -1 and 1 for y
- Line 5 sets the range for t for the second plot
- Line 6 establishes the second function to be plotted
- Line 7 divides the plot window into 1 row and 2 columns of panes, and sets the second plot to be displayed in the second pane
- Line 8 shows the plot of y versus t, with the horizontal axis labeled t, the vertical axis labeled y, and the display of the plot over the ranges 0 and 2 for t, -3 and 3 for y

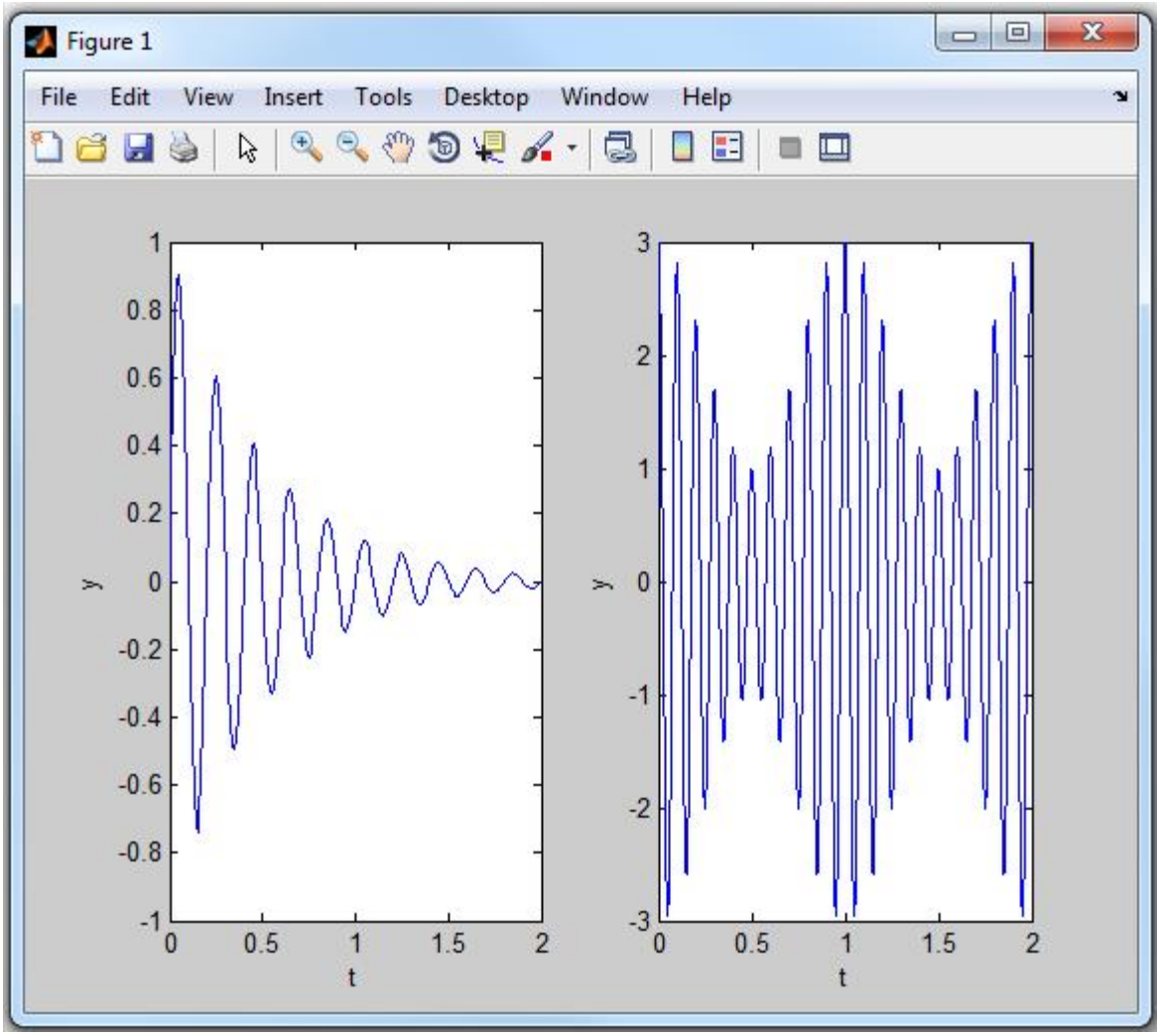


Figure 4.24: Plot of the functions defined in Figure 4.23

4. Back to the complex circuit

We return to the circuit introduced in Fig 4.1 described by the following set of equations:

$$\begin{aligned} R_1\mathbf{I_1} + R_2\mathbf{I_2} &= V_{B1} \\ -(R_3 + R_4)\mathbf{I_1} + (R_2+R_3+R_4)\mathbf{I_2} + R_4\mathbf{I_5} &= V_{B2} + V_{B3} \\ -R_4\mathbf{I_1} + R_4\mathbf{I_2} + (R_4+R_5)\mathbf{I_5} &= V_{B3} \end{aligned}$$

The script file shown in Figure 4.25 describes the solution to this set of equations with unknowns $\mathbf{I_1}$, $\mathbf{I_2}$, $\mathbf{I_3}$, and the desired voltage $\mathbf{V_3} = R_3\,(\mathbf{I_1} - \mathbf{I_2})$

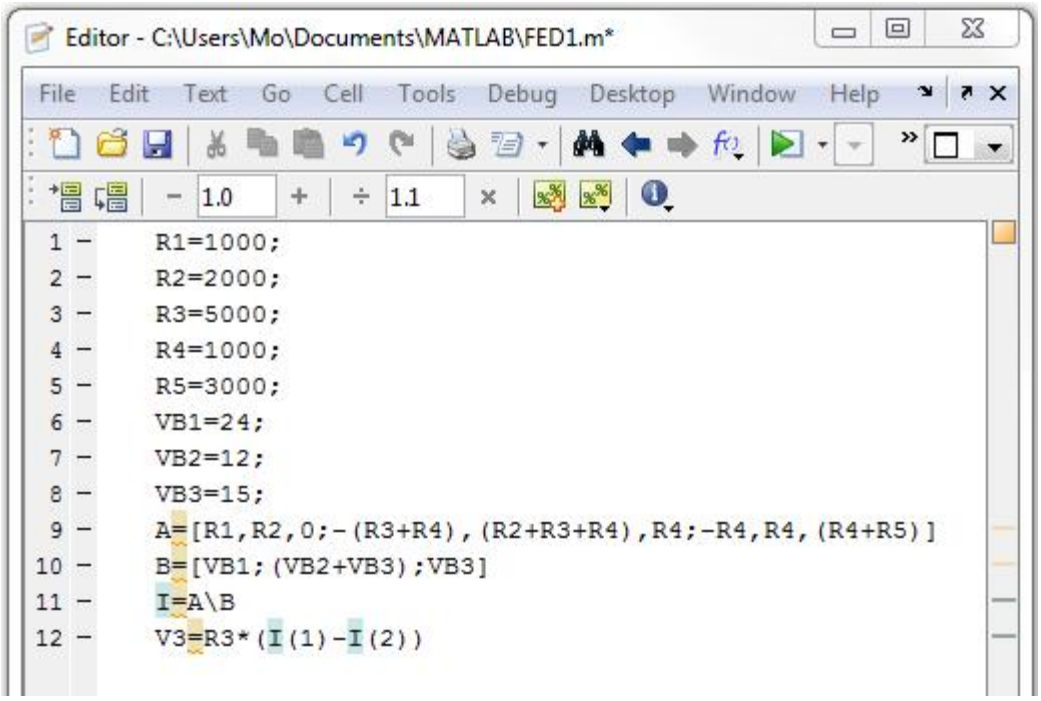


Figure 4.25: Script file for solving the problem shown in Figure 4.1

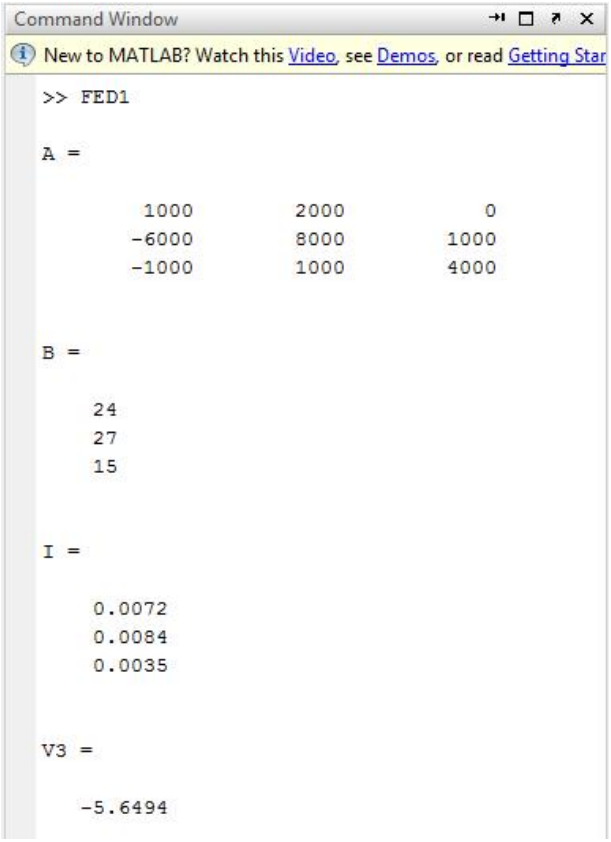
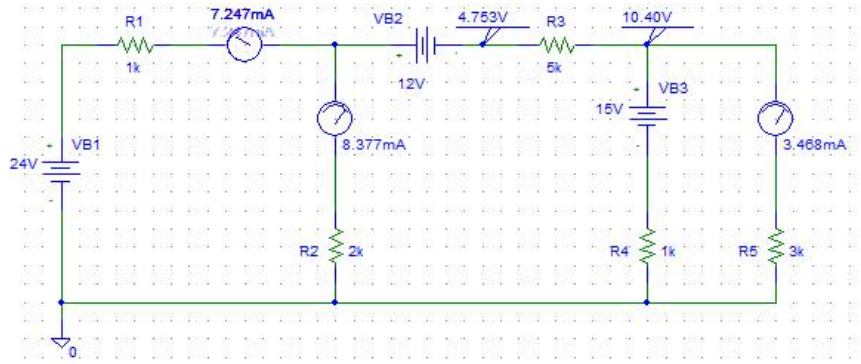
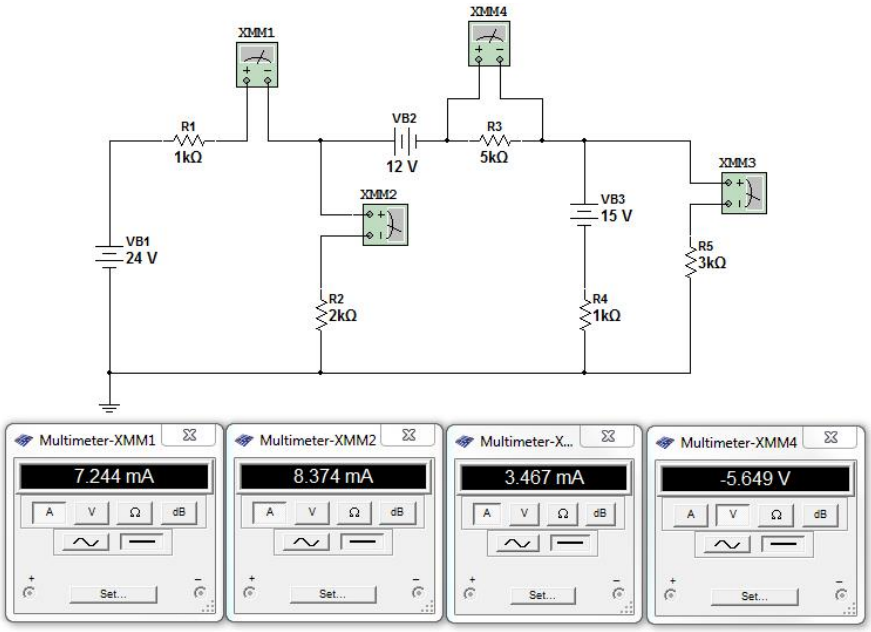


Figure 4.26: Execution of the script file from Figure 4.25

Note that the calculated values match exactly the ones given by PSpice and Multisim reproduced again below for comparison purposes.



(a)



(b)

Figure 4.27: Results of simulation from PSpice (a) and Multisim (b)

We see that with a few keystrokes, Matlab gave us all the needed answers. You will appreciate the power of Matlab as your technical skills and knowledge increase. The main drawback is that it is not very user-friendly. However, most designs and analyses had been made possible with this package. Like most software packages, one always has to verify that the results obtained digitally make sense. No machine or software package is full proof. However, what would take you a day by hand (if at all possible depending on the complexity of the circuit), would take few minutes of programming, and most likely a fraction of a second of execution with a computer.

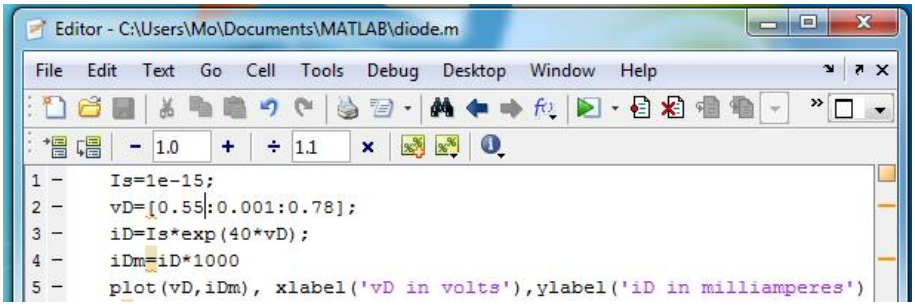
5. Diodes and Curve Fitting

The model that represents the relationship between the voltage across the diode and the current that flows through it is:

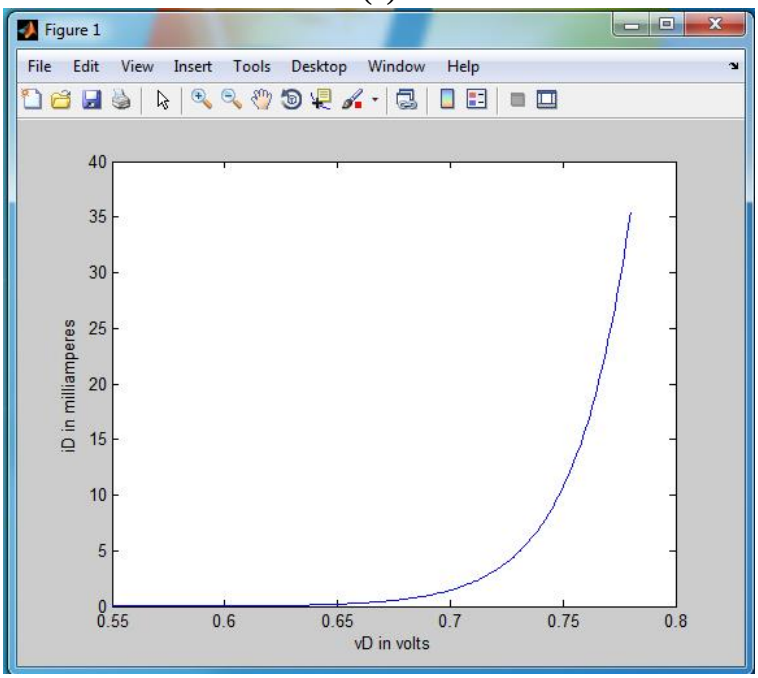
$$i_D = I_s\, e^{\frac{v_D}{V_T}}$$

The diode will be assumed to have a saturation current of $I_s = 10^{-15}$ A, and the thermal voltage V_T will be assumed to be equal to 25 mV at room temperature. Note that for $V_T = 25*10^{-3}$V, that means that $1/V_T = 40\ V^{-1}$, and hence

$$i_D = I_s e^{40 v_D}$$



(a)



(b)

Figure 4.28: Plot of current versus voltage for a diode

It would also be beneficial to learn to extract from a plot obtained through experimental work the characteristics of a diode. Assume that a student conducted an experiment where the current through a diode was measured for different voltage levels across it, an experiment similar to the one described in Figure 4.29.

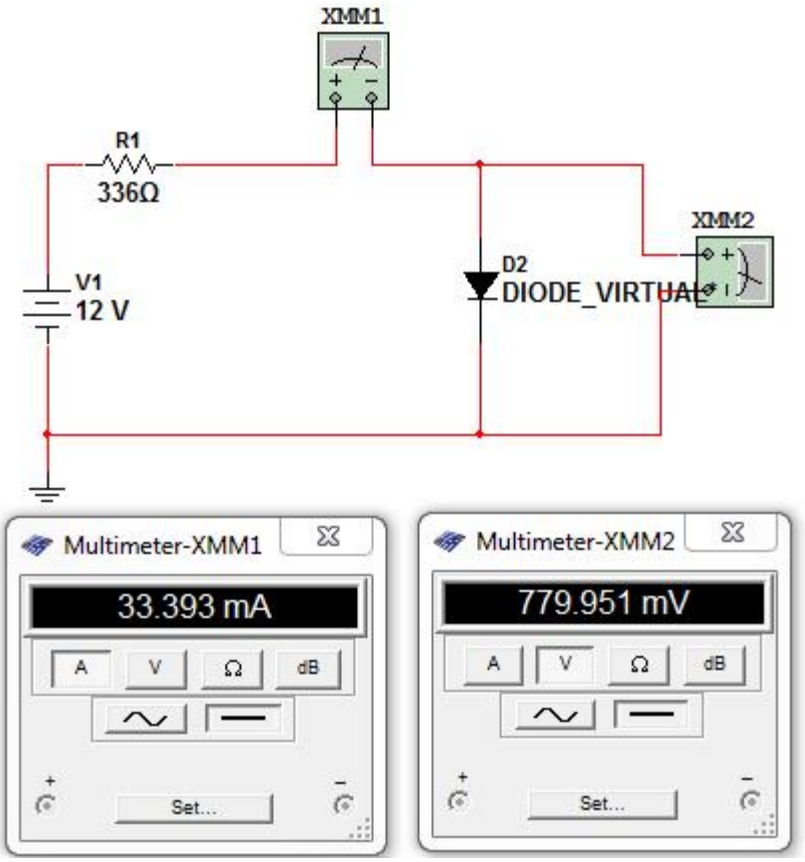


Figure 4.29: Current versus voltage for a diode circuit

The measurements given by simulation for a virtual diode having $I_s = 10^{-15}$ A and at room temperature (say 20.7°C) are shown in Table 4.2

R_1(kΩ)	35.1	16.2	7.45	3.43	1.58	.73	0.336
v_D(V)	0.66	0.68	0.7	0.72	0.74	0.76	0.78
i_D(mA)	0.323	0.699	1.517	3.2289	7.127	15.4	33.4

Table 4.2: Measurements from a diode circuit through simulation

However, due to the real physical characteristics of the diode different from the characteristics of the Multisim diode model, the limitations of the measuring instruments, and discrepancies during those measurements, the values acquired during the experiments are shown in Table 4.3

v_D (V)	0.66	0.68	0.7	0.72	0.74	0.76	0.78
i_D (mA)	0.4	0.8	1.7	3.5	7.8	16.5	35.2

Table 4.3: Measurements from a diode circuit through a lab experiment

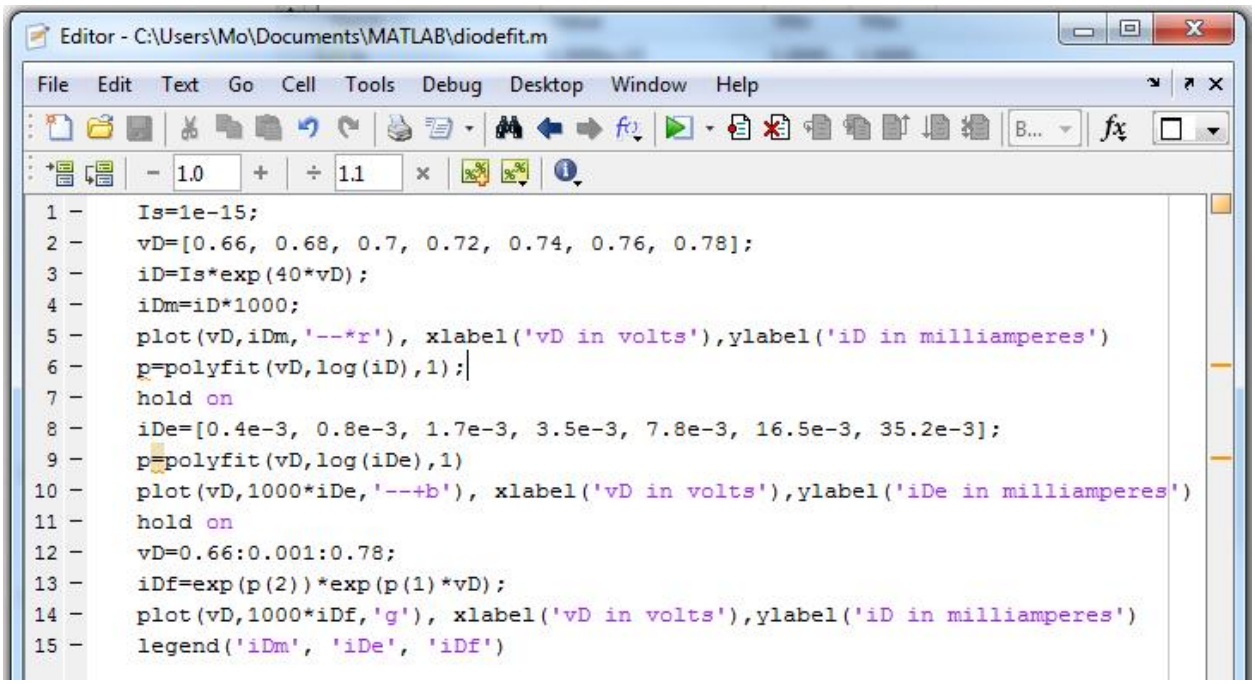


Figure 4.30: Curve fitting applied to current versus voltage for a diode

```
>> diodefit

p =

   37.5152  -32.6241
```

Figure 4.31: Curve fitting polynomial coefficients for the diode plot

This exponential fit shows that $V_T = 1/p(1) = 1/37.5152$

$V_T = 26.7$ mV

and $I_s = e^{(p(2))} = e^{-32.6241}$

$I_s = 6.78\ 10^{-15}$ A

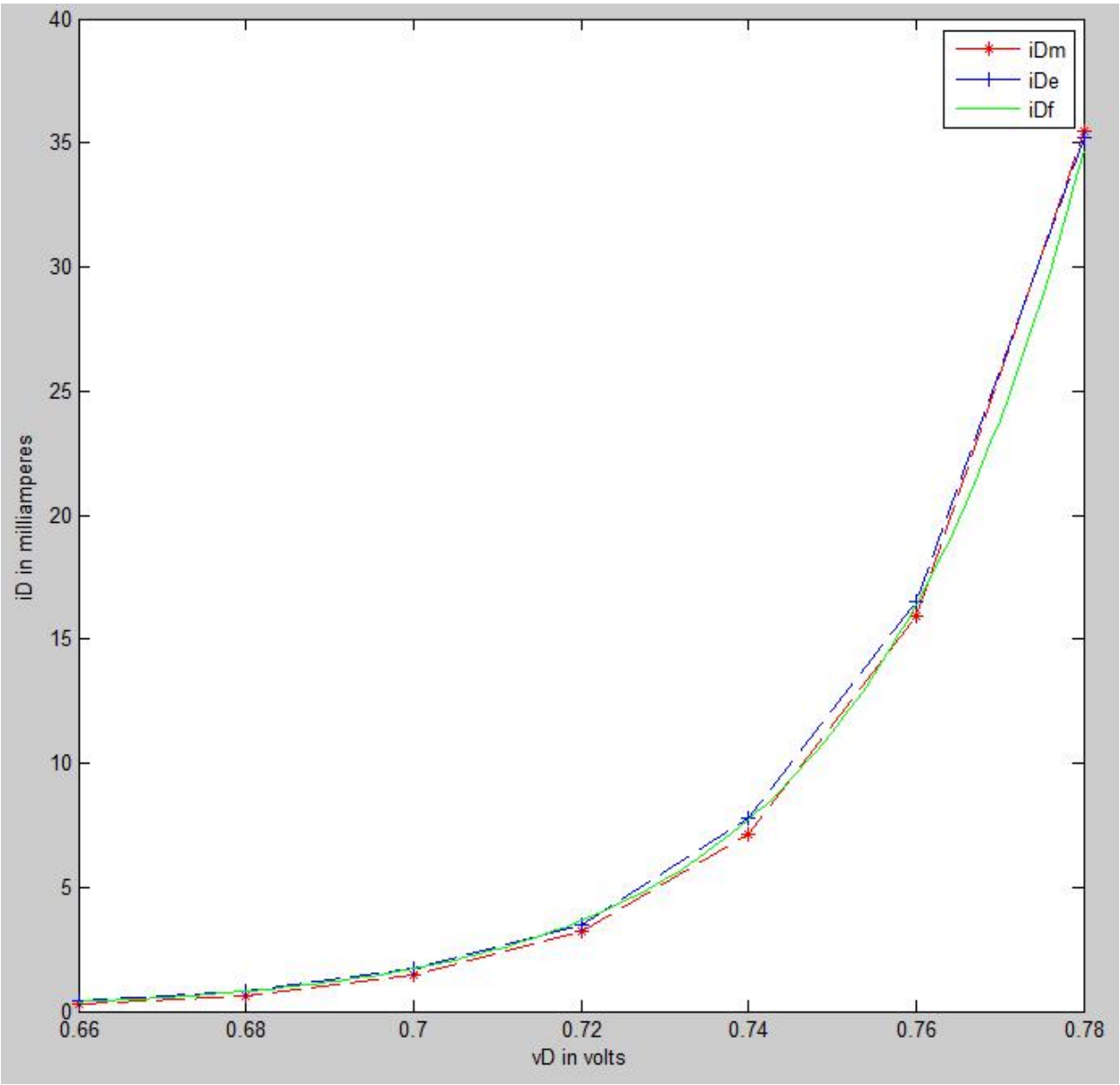


Figure 4.32: Curve fitting plot applied to current versus voltage for a diode

5 How Things Work

5.1 Introduction

Among the goals of this course is gaining understanding of engineering design work, enhancing your communication skills, and practicing team work. For this purpose you are required to pick a topic related to electrical and computer engineering and make a professional presentation. A list of suggested topics is included in this chapter but you may also suggest your own (subject to approval by the instructor). The whole team of students that had been formed at the beginning of the semester will start research using the Internet, the library, and other resources (see below) to prepare the elements necessary to present the information related to the description of a scientific or technological phenomenon, concept, machine, or innovation. Choose your topic wisely as you have to be excited about what you would be presenting to do it well. Your excitement will motivate others, in your team as well as in other teams in the class, to do a good presentation as well.

5.2 List of Suggested Topics

The following is by no means an exhaustive list. It will only help you choose when you still are undecided about your own choice. The listing will appear in no particular order.

1. Cellular technology, or in general, wireless communications
2. CD and/or DVD burners
3. How does an electric motor work?
4. How does a power station generate electricity?
5. Solar power
6. How do computers communicate through the World Wide Web?
7. How does electronics (so-called computer boards) affect the performance of a car?
8. How do we manufacture integrated circuits (ICs)?
9. How does a hard drive store data?
10. How do servomechanisms help in the design of robots?
11. How do we communicate with satellites?

5.3 Suggested resources

NJIT Library (Professional websites such as IEEE)

Websites

ECE website
Encyclopedia.thefreedictionary.com
Electronics.howstuffworks.com
Google.com

Advertising is part of the way these and other sites will support themselves. Concentrate on the material that is of interest to you.

5.4 Presentation

The task of an engineer is not finished unless he or she can achieve a good presentation of his or her work. The outside world (from the peers and employer to clients and other people who may be interested in the results) needs to find out firsthand the importance of that work. Who is better than the author to render justice to the efforts expended? Nobody can be more excited and knowledgeable than the author in presenting it. A presentation of that work starts with a well written text (mostly in bullet form and title rather than sentences). A well written power point is not necessarily a piece of literature. It has to appeal to potential customers of that work (those customers may be the general public). The styles vary depending on the recipients of that information. It cannot be boring, lengthy, or too short (not enough information); and of course it should not contain spelling and grammatical errors.

The next step is to present that information in an informative and exciting way directly to the audience. The most common way of delivering that material is through a power-point presentation. Other means could include transparencies, poster boards, video, or other visual means. Try to master as much of the material you will be presenting as you can. Otherwise, you will be reading your material from flash cards, or worse from your power point (the audience does not get excited by a speaker reading his material and also may be reading with you therefore disconnecting themselves from you and you losing their attention).

A sample of a PowerPoint presentation submitted by your peers in the past is available in appendix A, and an electronic copy of it is also available on the website. The PowerPoint may be improved greatly but it only serves as a reference to build upon.

6 Digital Logic

One of the more recent introductions in electrical devices is the personal computer. Computers rely on voltages and currents to represent binary and Boolean values. In this chapter we look at Boolean values and Boolean logic, and basic digital logic devices that realize Boolean operations.

6.1 Introduction to Boolean Logic

Boolean logic is the basis for digital system design, from the simplest circuits to the most advanced computer systems. From a logic standpoint, a Boolean value can be true or false; on or off; yes or no; "maybe" is not allowed. You can think of this like a light switch that can be either on or off; no dimmer switches allowed. In digital logic, true is usually represented by a 1 and false is denoted as 0.

Having a Boolean value of 0 or 1 isn't very useful on its own. To perform useful work with Boolean values, and ultimately the digital circuits we design, we must be able to perform functions that combine different values to produce desired results. There are several basic functions for Boolean values, including the AND, OR, Exclusive-OR, and NOT functions.

6.1.1 The Logical AND Function

The AND function is one of the most commonly used functions in Boolean algebra. It takes two or more Boolean values as its inputs and generates a single Boolean output. This output is 1 (on) only if all of its inputs are 1; if any input is 0 (off) then the output is 0, even if all of the other inputs are 1. This is similar to what you might see if you serve on a jury. For criminal trials, the jury must return a unanimous verdict to convict the defendant. Here, every juror must vote guilty (1) to produce a verdict (output) of guilty. If any juror votes not guilty (0), there can be no guilty verdict.

As another example, consider the extreme example of an AND function with one million inputs. If 999,999 of these input values are 1, but the other input is 0, the AND function outputs a 0. The AND function can be represented using several notation. The AND of Boolean values A and B can be represented as $A \cdot B$, $A \wedge B$, or simply AB.

One way to visualize the AND function is as a circuit consisting of a battery, a light bulb, and two or more switches in series, as shown in Figure 6.1. The switches represent the input values to the Boolean function; a closed switch represents a 1 (on) and an open switch represents a 0 (off). The light bulb corresponds to the output of the AND function. The bulb is on when the function outputs a 1. Figure 6.1(a) gives an example where input $A = 1$ and input $B = 0$, producing an output of 0. Figure 6.1(b) shows the same circuit, except $A = 1$ and $B = 1$, producing an output of 1 and lighting the bulb.

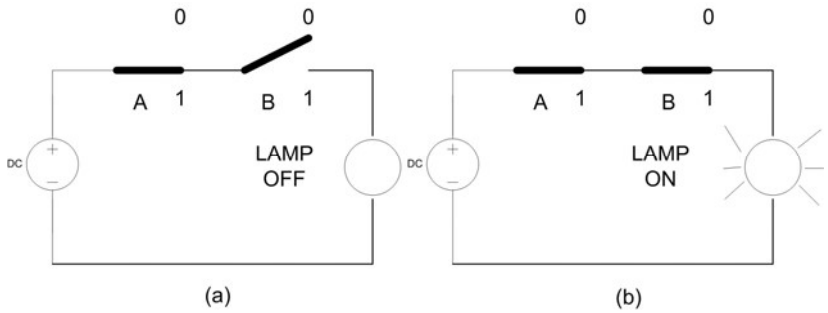


Figure 6.1: Illustration of the AND function producing results of (a) 0, and (b) 1

6.1.2 The Logical OR Function

The OR function is more tolerant of its input values. Its output is 1 if *any* of its inputs are 1. Returning to our extreme, one million input function, if 999,999 of the inputs are 0 but the remaining input is 1, the OR function outputs a 1. The OR of Boolean values *A* and *B* can be represented as $A \square B$ or $A + B$. Be careful with the latter notation. Here, $A + B$ is not arithmetic. In standard arithmetic, 1 + 1 = 2; however, in Boolean logic, 1 + 1 = 1.

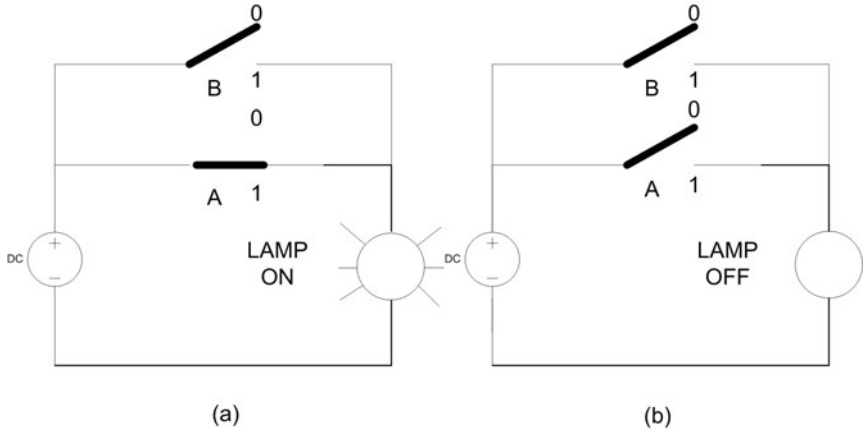


Figure 6.2: Illustration of the OR function producing results of (a) 1, and (b) 0

To visualize this function, consider the circuits shown in Figure 6.2. The switches representing *A* and *B* are connected in parallel; setting either switch to 1 completes the circuit and lights the bulb. Figure 6.2(a) shows input values of $A = 1$ and $B = 0$, which produces an output of 1. Figure 6.2(b) shows inputs of $A = 0$ and $B = 0$, generating a result of 0.

6.1.3 The Logical Exclusive-OR Function

The Exclusive-OR, XOR, function, is based on the OR function. Unlike the OR function, however, for the Exclusive-OR function the number of inputs that are 1 does matter. If two inputs are 1, their XOR is equal to 0, not 1; the logical OR of two inputs with values of 1 is 1. When more than two input values are Exclusive-ORed, the output if 1 if an odd number of inputs are 1

and 0 if the number of inputs with values of 1 is even. The Exclusive-OR of Boolean values *A* and *B* is represented as *A* □ *B*.

The Exclusive-OR function is similar to a 3-way switch you might find at the top and bottom of a stairway in a house. If one switch is up and the other is down, the light is off. If either switches are up, or both are down, the light is lit. (This may be reversed in some houses. My house has two sets of 3-way switches. The upstairs light is lit when the switches are in opposite positions; the downstairs light is only lit when both switches are in the same position.) Figure 6.3 shows an Exclusive-OR configuration in which the switches must be in opposite positions to light the bulb, or output a 1. In Figure 6.3(a), *A* = 0, *B* = 1, and the output is 1. Figure 6.3(b) shows *A* = 1, *B* = 1, and an output of 0.

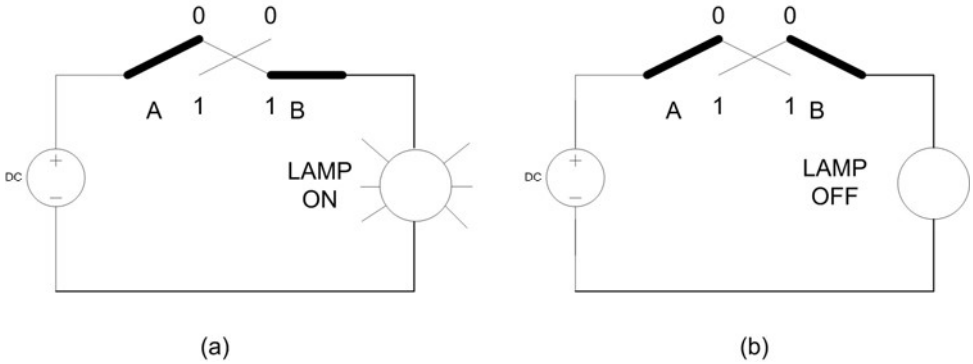


Figure 6.3: Illustration of the XOR function producing results of (a) 1, and (b) 0

6.1.4 The Logical NOT Function

Unlike the AND, OR, and XOR functions, which received two or more inputs, the NOT function operates on a single input. Its output is the opposite of its input value. An input of 0 produces an output of 1, and an input value of 1 generates a 0 output. The NOT of Boolean value *A* can be represented as *A'*, /*A*, or !*A*. Figure 6.4 illustrates a NOT function for input values of 0 and 1.

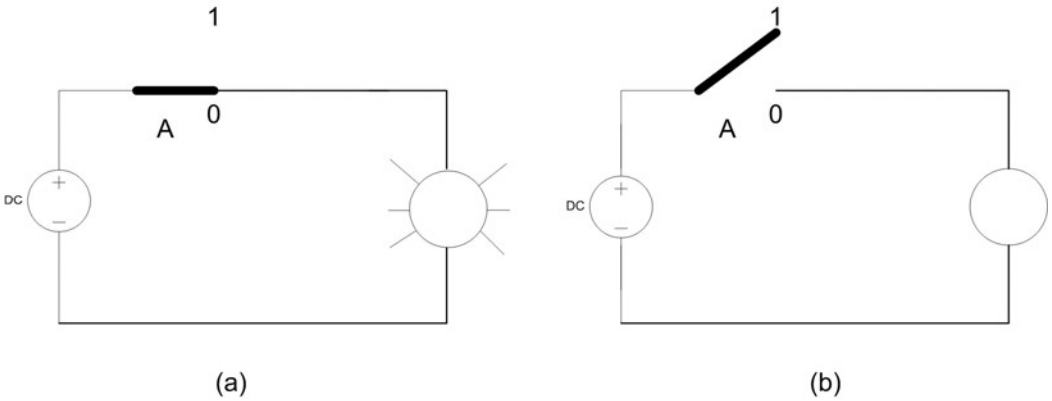


Figure 6.4: Illustration of the NOT function producing results of (a) 1, and (b) 0

Worksheet 6.1 – Boolean Logic Calculations

Show the values of each function for the given values of *A*, *B*, and *C*.

1. $A = 0, B = 0$: $A \wedge B =$ _____ $A \vee B =$ _____ $A \oplus B =$ _____

   $A' =$ _____ $B' =$ _____

2. $A = 0, B = 1$: $A \cdot B =$ _____ $A + B =$ _____ $A \oplus B =$ _____

   $/A =$ _____ $/B =$ _____

3. $A = 1, B = 0$: $AB =$ _____ $A \vee B =$ _____ $A \oplus B =$ _____

   $!A =$ _____ $!B =$ _____

4. $A = 1, B = 1$: $A \wedge B =$ _____ $A + B =$ _____ $A \oplus B =$ _____

   $A' =$ _____ $B' =$ _____

5. $A = 1, B = 0, C = 1$: $A \cdot B \cdot C =$ _____ $A \vee B \vee C =$ _____

   $A \oplus B \oplus C =$ _____

6. $A = 0, B = 1, C = 0$: $A \cdot B \cdot C =$ _____ $A \vee B \vee C =$ _____

   $A \oplus B \oplus C =$ _____

6.2 Truth Tables

Truth tables are a mechanism used to express the operations of a function for all possible input values. In the previous worksheet, we calculated the values of the functions A □ B, A □ B, A □ B, A', and B' for all possible values of A and B. Truth tables show these values in a convenient format. A truth table has two groups of columns, and several rows of data. The left-hand group of columns consists of all inputs. Each row of the table shows a different combination of input values. For example, a table with two inputs, A and B, would have values of $A = 0$ and $B = 0$ in its first row, followed by A and B of 0 and 1; 1 and 0; and 1 and 1, as shown in Table 6.1.

Table 6.1: A generic truth table for a 2-input function

A	B	Function
0	0	
0	1	
1	0	
1	1	

Consider the truth table for the 2-input AND function shown in Table 6.2(a). The left hand side shows the two inputs, A and B, and all possible combinations of values for these inputs, one combination per row. The first row shows $A = 0$ and $B = 0$, followed by rows for $A = 0$ and $B = 1$, $A = 1$ and $B = 0$, and $A = 1$ and $B = 1$. The right hand side of the table shows the output of the AND function for these input values. The AND function only outputs a 1 when $A = 1$ and $B = 1$. It outputs a 0 at all other times. Tables 6.2(b), (c), and (d) show the truth tables for the OR, XOR, and NOT functions.

Table 6.2: Truth tables for the (a) AND, (b) OR, (c) XOR, and (d) NOT functions

A	B	$A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

(a)

A	B	$A \vee B$
0	0	0
0	1	1
1	0	1
1	1	1

(b)

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

(c)

A	A'
0	1
1	0

(d)

6.3 Complementary Functions – NAND, NOR, and Exclusive-NOR

In addition to the AND, OR, XOR, and NOT functions, there are three common, complementary functions. The NAND function is the opposite of the AND function; it may help to think of the NAND as the NOT-AND function. The NAND of two or more inputs is 0 if all of the inputs are 1; if any input is 0, the output of the NAND is 1. Table 6.3(a) shows the truth table for the 2-input NAND function. Note that its outputs are exactly the opposite of those for the AND function of Table 6.2(a)

Table 6.3: Truth tables for the (a) NAND, (b) NOR and (c) XNOR functions

A	*B*	NAND
0	0	1
0	1	1
1	0	1
1	1	0

(a)

A	*B*	NOR
0	0	1
0	1	0
1	0	0
1	1	0

(b)

A	*B*	XNOR
0	0	1
0	1	0
1	0	0
1	1	1

(c)

There are also complementary functions for the OR and XOR functions. The NOR function produces an output of 1 only when all of its inputs are 0. Its truth table is shown in Table 6.3(b); as with the NAND and AND functions, the outputs of the NOR function are exactly the opposite of the OR function. The Exclusive-NOR, or XNOR, function outputs a 1 when an even number of inputs are 1, and 0 when an odd number of inputs are 1. Its truth table is shown in Table 6.3(c). Again, the outputs of the XNOR function are exactly the opposite of those of the XOR function in Table 6.2(c). Note that the output of the 2-input XNOR function is 1 whenever the two inputs are the same. For this reason, the 2-input XNOR function is sometimes referred to as the *equivalence* function.

Worksheet 6.2 – Boolean Logic Calculations

Show the truth tables for the 3-input AND, OR, XOR, NAND, NOR, and XNOR functions.

A	B	C	$A \wedge B \wedge C$
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

A	B	C	$A \vee B \vee C$
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

A	B	C	$A \oplus B \oplus C$
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

A	B	C	$(A \wedge B \wedge C)'$
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

A	B	C	$(A \vee B \vee C)'$
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

A	B	C	$(A \oplus B \oplus C)'$
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

6.4 Digital Logic

Logic gates are digital components that implement the logic functions described earlier. They are usually represented by the symbols shown in Figure 6.5. For functions with more than two inputs, the gate symbols are the same, except more inputs are added to the gate. (The NOT gate is an exception; it can never have more than one input.) Each gate always has one output regardless of the number of inputs.

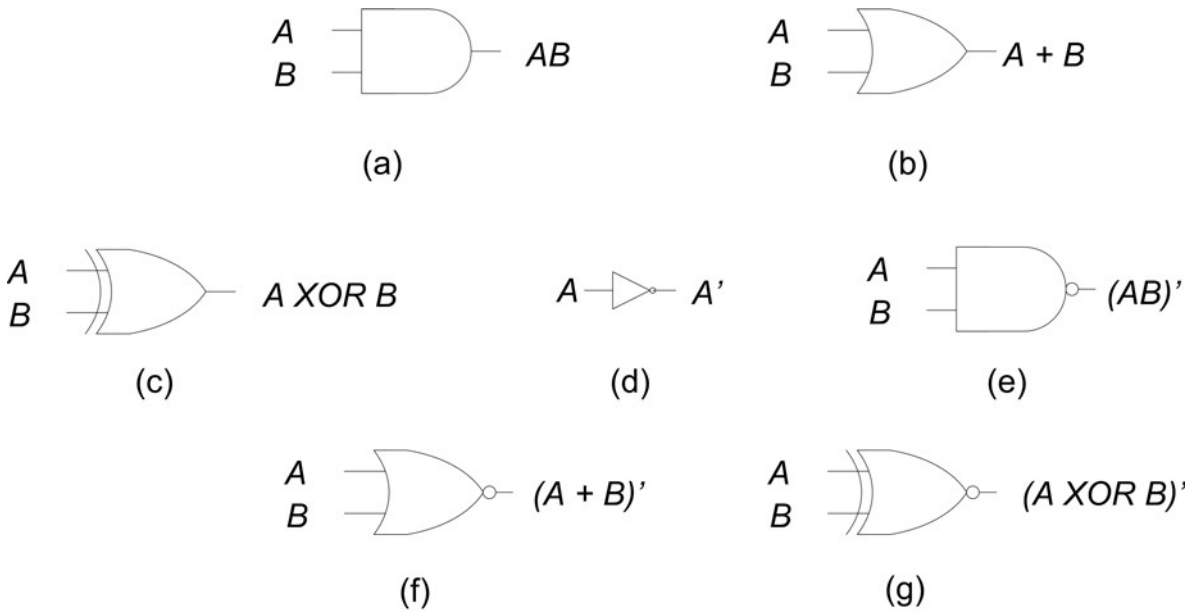


Figure 6.5: Logic symbols for the (a) AND, (b) OR, (c) XOR, (d) NOT, (e) NAND, (f) NOR, and (g)
XNOR gates

Notice that the NOT, NAND, NOR, and XNOR gates have small circles at their outputs. This indicates that the output of the gate is complemented, or inverted. For example, consider the AND and NAND gates. They look the same, except the NAND gate has a small circle at its output. Remember, as shown in the truth tables, the output of the NAND gate is the complement of the output of the AND gate. Logically, you can consider the NAND gate to be an AND gate which sends its output to a NOT gate, as shown in Figure 6.6(a). The reverse is also true; an AND gate has the same logical function as a NAND gate that sends its output to a NOT gate as shown in Figure 6.6(b).

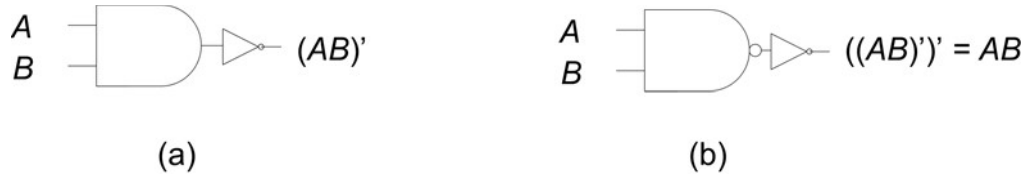


Figure 6.6: Logical implementation of (a) a NAND function using AND and NOT gates, and (b) an

AND function using NAND and NOT gates

6.5 Logic Chips

To build logic circuits, designers can use Integrated Circuit chips, or ICs, or chips. Different ICs are available with the gates we've seen so far, as well as more advanced components. A chip may look like the chip shown in Figure 6.7(a). The black box in the center of the chip contains the circuitry that implements the chip's functions. The metal strips on the sides of the chip are its pins. These pins interface the logic gates in the chip to the rest of the circuit. For a NAND gate with two inputs, two of the chip's pins would connect to the inputs of the NAND gate, and the output of the NAND gate would connect to another pin. By connecting the pins of the chip, we incorporate the NAND gate into a circuit.

(a) (b)

Figure 6.7: (a) A 7400 Quad 2-input NAND gate, and (b) its logic diagram

Figure 6.7(b) shows the logical organization of a 7400 TTL chip. This chip has four, 2-input NAND gates. For each NAND gate, two pins are connected to the gate's inputs and one pin connects to its output. This accounts for 12 of the chip's 14 pins. The other two pins are used for power connections. Notice that the pins are numbered counterclockwise around the chip starting in the upper left corner. Chips usually have a dot by pin 1 or a notch at the top of the chip, such as the notch shown in the figure, so you can tell how to connect it in your circuit.

The logic gates require power in order to function; we use the two power pins to supply power to the chip and its gates. Recall from the circuits covered earlier, those consisting of switches and light bulbs, that a circuit must be completed from the battery's positive terminal to its negative terminal in order for current to flow. One of the two power pins, denoted Vcc, is

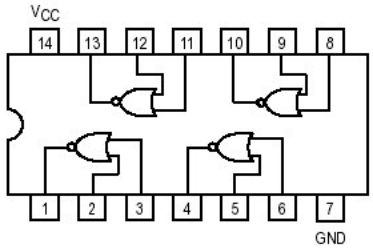
connected to the positive terminal. The other power pin, Gnd or ground, is connected to the negative terminal. For TTL chips, the voltage of the battery or power supply is 5 volts, denoted 5V.

6.6 Where Boolean Logic Meets Digital Logic

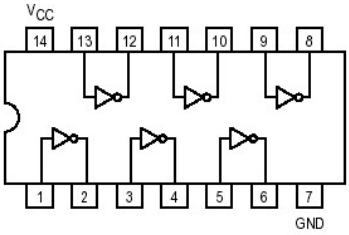
A logic chip doesn't know what binary values 0 and 1 are. To input a 1 to a gate, we supply a voltage at a given level. For TTL chips, logic 1 is a voltage at or near 5V, the voltage of the chip's power supply. A voltage near 0V is used for logic 0. The same voltage values are used for the inputs and outputs.

Other Logic Chips

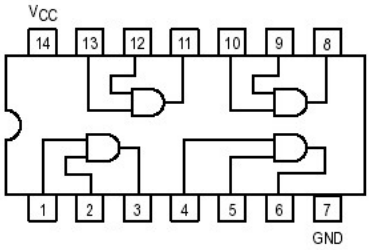
Now let's look at some other chips. The 7402 chip has four NOR gates; its logic diagram is shown in Figure 6.8(a). The 7404 is a hex inverter, having six NOT gates. It is shown in Figure 6.8(b). Figure 6.8(c) shows the 7408 chip, which has four AND gates. The quad OR, XOR, and XNOR chips, the 7432, 7486, and 74266, respectively, are shown in Figures 6.8(d), (e), and (f).



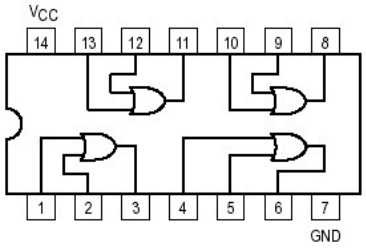
(a)



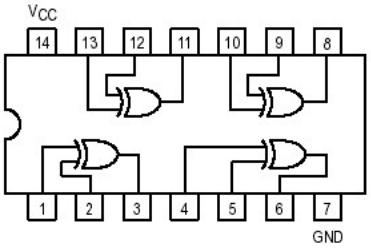
(b)



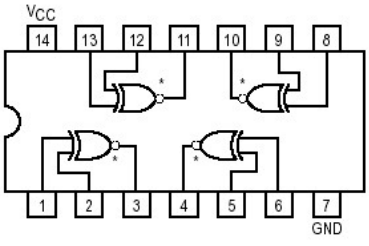
(c)



(d)



(e)



(f)

Figure 6.8: Logic diagrams for the (a) 7402, (b) 7404, (c) 7408, (d) 7432, (e) 7486, and (f) 74266 chips

6.7 Combining Logical Operations

The AND, OR, XOR, and NOT functions introduced in the previous sections are useful in their own right, but they are most useful when combined to form more complex functions. In the real world, situations that cannot be described with a single AND or OR function often can be modeled by combining logical operations.

For example, consider the following scenario. A teenager wants to buy a car but doesn't have any money for a down payment. Ignoring options that involve borrowing money from his parents, winning the lottery, or robbing a bank, the teenager has two options. One is to earn more money, possibly by working more hours or taking a second job. The other is to earn the same amount of money and spend less, maybe by giving up his or her cell phone. We could express these options by combining the logical AND and OR operations as follows.

Get $ to buy car = (*Earn more $*) OR (*Earn same $* AND *Spend less $*)

6.8 Modeling Real-world Situations

We can design a circuit to model this function. We would need three switches, one each for *Earn more $*, *Earn same $*, and *Spend less $*. We would also need one light bulb to represent the output *Get $ to buy car* and a battery to supply power to the circuit. The circuit would light the bulb if *Earn more $* is TRUE (1), or if both *Earn same $* and *Spend less $* are true. The circuit to do this is shown in Figure 6.9. Notice that this circuit combines elements of the AND and OR circuits from the previous chapter. The light can be lit if both *Earn same $* and *Spend less $* are on (TRUE). This is the AND function. The light can also be lit if *Earn more $* is true. Having either option light the bulb is an OR function.

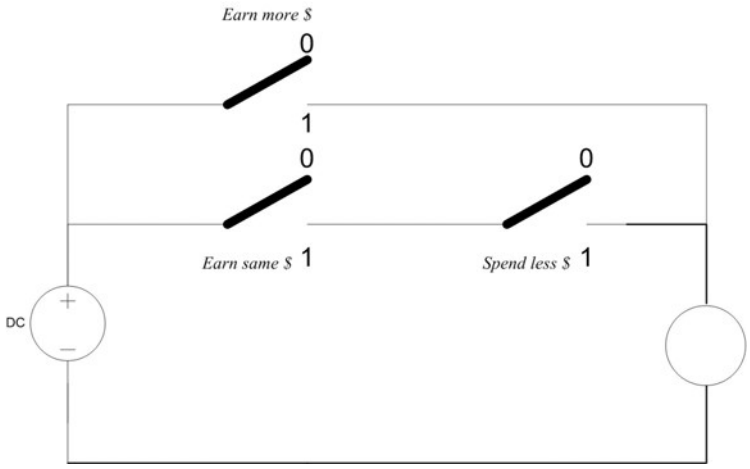


Figure 6.9: Circuit to generate the *Get $ to buy car* function

6.9 Truth Tables for Combined Functions

Just as with any logic function, we can create a truth table for our car-buying example. Here the variables are the three conditions *Earn more $*, *Earn same $*, and *Spend less $*, which we represented as switches in our circuit. There is one output for this function, *Get $ to buy car*; this was represented as a light bulb in our circuit. Each input, and the output, can be either TRUE or FALSE. The truth table for this scenario is shown in Table 6.4. Table 6.4(a) shows the truth table with values of TRUE and FALSE. Table 6.4(b) shows the same table with binary values 1 and 0; 1 represents TRUE and 0 corresponds to FALSE.

Table 6.4: Truth tables for the *Get $ to buy car* function

Earn more $	*Earn same $*	*Spend less $*	*Get $ to buy car*
FALSE	FALSE	FALSE	FALSE
FALSE	FALSE	TRUE	FALSE
FALSE	TRUE	FALSE	FALSE
FALSE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	TRUE
TRUE	FALSE	TRUE	TRUE
TRUE	TRUE	FALSE	TRUE
TRUE	TRUE	TRUE	TRUE

(a)

Earn more $	*Earn same $*	*Spend less $*	*Get $ to buy car*
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

(b)

6.9.1 Creating the Exclusive-OR Function

Let's look at another way of creating the exclusive-OR function. Instead of using an XOR gate, we will construct this function using AND, OR, and NOT operations. First we start with the truth table for the exclusive-OR function, shown in Table 6.5.

Table 6.5: Truth table for the XOR function

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

To create the XOR function, we will develop functions that generate each 1 output separately and then logically OR these functions together. Looking at the table, we can see that $A \oplus B = 1$ twice, when $A = 0$ and $B = 1$, and when $A = 1$ and $B = 0$. First we will create the function to generate the first 1. This function is 1 when $A = 0$ and $B = 1$, or when $A' = 1$ and $B = 1$. (Remember that A' is the NOT of A; if $A = 0$ then $A' = 1$, and if $A = 1$ then $A' = 0$.) We can implement this function as *(A' B)+(A B')*. Its truth table is shown in Table 6.6(a).

Table 6.6: Truth tables for the functions (a) $A'B$, (b) AB', and (c) $A'B + AB' = A \oplus B$

A	B	$A'B$
0	0	0
0	1	1
1	0	0
1	1	0

(a)

A	B	AB'
0	0	0
0	1	0
1	0	1
1	1	0

(b)

A	B	$A'B + AB' = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

(c)

Next we calculate the function for the second 1. This occurs when $A = 1$ and $B = 0$, or if $A = 1$ and $B' = 1$. Its function is AB', and its truth table is shown in Table 6.6(b).

Finally, we logically OR these two functions together to produce a single, combined function. Here this function is $A'B + AB'$, which is true whenever either $A'B$ or AB' is true. Its truth table is shown in Table 6.6(c). Note that this truth table is identical to the truth table for $A \oplus B$ given in Table 6.5. Therefore, $A \oplus B = A'B + AB'$.

Worksheet 6.3 – Truth Tables

Show the truth tables for the following functions.

A	B	C	$A + B + C$
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

A	B	C	AB	BC	$AB + BC$
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

A	B	C	$A'BC$	$A'C'$	$A'BC + A'C'$
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

A	B	C	$A'C$	BC	$A'B$	$A'C + BC + A'B$
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

A	B	C	ABC	ABC'	$A'B$	$ABC + ABC' + A'B$
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

6.9.2 Determining Functions from Truth Tables

Besides using functions to create truth tables, we can also work in the other direction. We can use truth tables to determine its function. Consider, for example, the truth table shown in Table 6.7. This function has three inputs, *A*, *B*, and *C*, and one output function.

Table 6.7: Truth table for an unknown function

A	*B*	*C*	*Function*
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Three different combinations of input values cause this function to be 1. The first occurs when $A = 0$, $B = 0$, and $C = 0$, or $A' = 1$, $B' = 1$, and $C' = 1$. This can be expressed as $A' \wedge B' \wedge C'$, or $A'B'C'$. The function is also 1 when $A = 0$, $B = 1$, and $C = 0$, or $A' = 1$, $B = 1$, and $C' = 1$. This is $A'BC'$. Finally, the function is 1 when $A = 1$, $B = 1$, and $C = 1$, or ABC. We can combine them to form a single function that matches this truth table, $A'B'C' + A'BC' + ABC$. Table 6.8 shows a truth table for each individual component of the function and the overall function.

Table 6.8: Truth table for the function and its components

A	*B*	*C*	*A'B'C'*	*A'BC'*	*ABC*	*A'B'C' + A'BC' + ABC*
0	0	0	1	0	0	1
0	0	1	0	0	0	0
0	1	0	0	1	0	1
0	1	1	0	0	0	0
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	0	0	0	0
1	1	1	0	0	1	1

Worksheet 6.4 – Truth Tables

Show the functions for the following truth tables. Express each entry in the truth table that generates an output of 1 as a separate term.

A	*B*	*C*	*Function*
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Overall function: _____

A	*B*	*C*	*Function*
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Overall function: _____

A	*B*	*C*	*Function*
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Overall function: _____

Overall function: _____

Overall function: _____

Overall function: _____

6.9.3 Simplifying Functions

Sometimes we can simplify a function by combining some of its terms. For the truth table in Table 6.8, take a look at the first two lines that produce outputs of 1. The inputs for these two lines are $A = 0$, $B = 0$, and $C = 0$, and $A = 0$, $B = 1$, and $C = 0$. Notice that both $A = 0$ and $C = 0$ in both cases. As long as A and C are both 0, the function should produce a 1, regardless of the value of B. If $B = 0$, then $A = 0$, $B = 0$, and $C = 0$, and $A'B'C'$ is 1. If $B = 1$, then $A = 0$, $B = 1$, and $C = 0$, and $A'BC'$ is 1. We can replace these two terms by the single term $A'C'$. Our function would then become $A'C' + ABC$; its truth table is shown in Table 6.9.

Table 6.9: Truth table for the simplified function

A	B	C	$A'C'$	ABC	$A'C' + ABC$
0	0	0	1	0	1
0	0	1	0	0	0
0	1	0	1	0	1
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	0	0	0
1	1	1	0	1	1

In general it is desirable to simplify functions. If you implement a function using logic gates, simplifying the function usually results in simpler circuits.

Worksheet 6.5 – Truth Tables

Show the simplified functions for the following truth tables. Express each entry in the truth table that generates an output of 1 as a separate term.

A	*B*	*C*	*Function*
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Original function: _____

Simplified function: _____

A	*B*	*C*	*Function*
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Original function: _____

Simplified function: _____

A	*B*	*C*	*Function*
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Original function: _____

Simplified function: _____

Simplified function: _____

Original function: _____

Simplified function: _____

Original function: _____

Simplified function: _____

6.10 Binary Numbers

Many digital circuits deal with data. To understand how these circuits work, we must first understand how binary data works. In this section we will examine the binary representations of data. We will study the direct storage of data in binary, as well as the binary coded decimal, BCD, data format.

6.10.1 Binary Values

In decimal notation, each digit has a specific value. This value depends on the digit itself, as well as its position. For example, the 1 in 12 is much less than the 1 in 1,234,567.

The decimal number 1,234,567 can be thought of as 1,000,000 + 200,000 +30,000 + 4,000 + 500 + 60 + 7. Each digit is followed by some number of zeroes, which means that the digit is actually multiplied by 10 raised to some power. That power depends on the location of the digit. For example, the 7 is multiplied by 10^0, or 1, since it is the rightmost digit. The 6 is one position to the left, so it is multiplied by 10^1, or 10, to produce a value of 60. The five is multiplied by 10^2, and so on. We could also express 1,234,567, as follows.

$$(1\times10^6) + (2\times10^5) + (3\times10^4) + (4\times10^3) + (5\times10^2) + (6\times10^1) + (7\times10^0)$$

The same holds true for binary values. However, since they use base 2 instead of base 10, we multiply the individual digits by powers of 2 instead of powers of 10. Consider the binary value 10111. This can be expressed as $(1\times2^4) + (0\times2^3) + (1\times2^2) + (1\times2^1) + (1\times2^0)$, which is 16 + 0 + 4 + 2 + 1 = 23 in decimal.

Binary values can get very large very fast. For convenience, we sometimes use *hexadecimal* notation to express binary values. Each group of four bits, starting at the right hand

side of the number, is represented as a single hexadecimal digit. For binary values from 0000 to 1001, we use the standard decimal digits 0 to 9; 0 = 0000, 1 = 0001, 2 = 0010, 3 = 0011, 4 = 0100, 5 = 0101, 6 = 0110, 7 = 0111, 8 = 1000, and 9 = 1001. However, binary values from 1010 to 1111 do not have single-digit decimal equivalents. For that reason, we use the letters A through F to represent these values; A is 1010, B is 1011, C is 1100, D is 1101, E is 1110, and F is 1111.

For convenience, large binary numbers can be broken down into groups of bits called *bytes*. Each byte is 8 bits wide. Computers often access several bytes of data simultaneously when processing instructions.

6.10.2 Binary Coded Decimal (BCD)

Some applications require decimal data. Typical of these applications are those with digital displays. We could store such data in binary format, and convert the binary values to decimal to display the data, but there is a better way. *Binary Coded Decimal*, or BCD, notation stores each decimal digit as four binary bits. The binary value only uses the bit patterns 0000 to 1001 to represent 0 to9. The binary patterns corresponding to hexadecimal values A to F are not used. In this way we can store decimal data directly and operate on the data in decimal.

To illustrate the differences between binary and BCD, consider the binary value 0010 0111. As a binary value, this would be $(0\times2^7) + (0\times2^6) + (1\times2^5) + (0\times2^4) + (0\times2^3) + (1\times2^2) + (1\times2^1) + (1\times2^0)$, or 39. However, let's say this is actually a BCD value. The low four bits, 0111, would represent a single decimal digit, 7. The next four bits, 0010, also represent a decimal digit, 2 in this case. Concatenating these values, we see that this represents 27 in BCD, which is quite different from the value of 39 of the binary representation.

Worksheet 6.6: Binary and BCD values

What are the decimal equivalents of the following binary values?

1. Binary: 01011001 Decimal: _____

2. Binary: 10011001 Decimal: _____

3. Binary: 00000001 Decimal: _____

What are the decimal equivalents of the following BCD values?

1. Binary: 01011001 Decimal: _____

2. Binary: 10011001 Decimal: _____

3. Binary: 00000001 Decimal: _____

Show the 8-bit binary values for the following decimal values.

1. Decimal: 43 Binary: _____

2. Decimal: 71 Binary: _____

3. Decimal: 53 Binary: _____

Show the 8-bit BCD values for the following decimal values.

1. Decimal: 43 Binary: _____

2. Decimal: 71 Binary: _____

3. Decimal: 53 Binary: _____

6.11 More Complex Digital Components

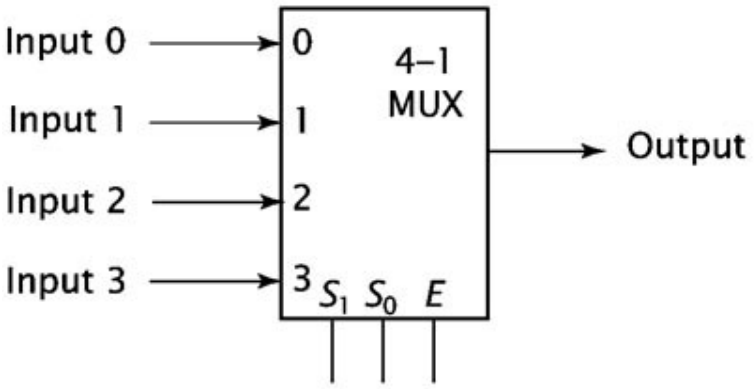
The digital components we've seen so far are fairly basic. Although they can perform some useful work on their own, they are rather limited. This section examines some more complex digital components that can be used to perform more complex functions. It looks at some components

that change their outputs as the inputs change; these components include multiplexers, decoders, and encoders. It also presents some components that can store data for future use, including registers and counters.

Note that some of the material in this chapter, and Figures 6.10 – 6.13, were adapted from *Computer Systems Organization and Architecture*, by John D. Carpinelli (Addison Wesley, Boston, MA, 2001).

6.11.1 Multiplexers

A multiplexer, or MUX, is a selector. It chooses one of its data inputs and passes it through to its output. To illustrate its operation, consider the 4 to 1 multiplexer shown in Figure 6.10. Four binary data values are passed to the inputs of the multiplexer. Two select signals, S_1 and S_0, determine which of the four inputs is passed through to the output. If $S_1 = 0$ and $S_0 = 0$, input 0 is passed through to the output. Similarly, setting S_1 and S_0 to 01, 10 or 11 would pass input 1, 2, or 3, respectively to the output. Note that the E signal is an *enable* signal. If this input is 1, the chip is enabled and performs as described. However, if the E input is 0, the chip is *disabled* and does not produce a usable output. This output, called the *high-impedance state*, or *Z*, does not represent a valid binary value.



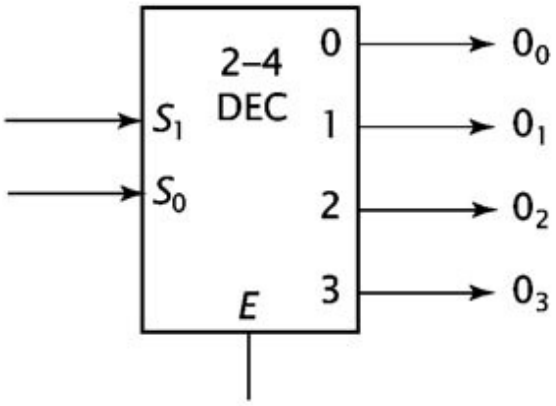
S_1	S_0	E	Output
X	X	0	Z
0	0	1	Input 0
0	1	1	Input 1
1	0	1	Input 2
1	1	1	Input 3

Figure 6.10: A 4-1 multiplexer

6.11.2 Decoders

A decoder, as its name implies, accepts a value and decodes it. It has n inputs and 2^n outputs, numbered from 0 to $2^n - 1$. Each output represents one possible value of the inputs. The output corresponding to the value of the n inputs is activated. For example, a decoder with 3 inputs and 8 outputs will activate output 6 when the input values are 110.

Figure 6.11 shows a 2 to 4 decoder. For inputs S_1S_0 = 00, 01, 10 and 11, outputs 0, 1, 2 and 3, respectively, are high when E is 1.

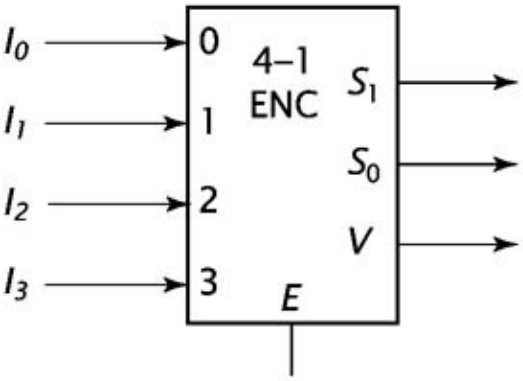


S_1	S_0	E	0_0	0_1	0_2	0_3
X	X	0	0	0	0	0
0	0	1	1	0	0	0
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	1

Figure 6.11: A 2-4 decoder

6.11.3 Encoders

The encoder is the exact opposite of the decoder. It receives 2^n inputs and outputs an *n*-bit value corresponding to the one input which has a value of 1. The 4 to 2 encoder is shown in Figure 6.12. Notice that there is a third output, *V*. This value indicates whether any of the inputs are high. As with the previous chips, the output is invalid when E is 0.



I_0	I_1	I_2	I_3	E	S_1	S_2	V
X	X	X	X	0	Z	Z	Z
0	0	0	0	1	0	0	0
1	0	0	0	1	0	0	1
0	1	0	0	1	0	1	1
0	0	1	0	1	1	0	1
0	0	0	1	1	1	1	1

Figure 6.12: A 4-2 encoder

This encoder works if E is 1 and exactly zero or one inputs are high, but fails if more than one is high. For instance, if inputs 1 and 2 are high, the encoder might set S_1S_0 = 01, S_1S_0 = 10, or some other value which represents neither input. When more than one input can be active, a different type of encoder must be used. This is the role of the *priority encoder*.

A priority encoder works just like a regular encoder, with one exception. Whenever more than one input is active, the output is set to correspond to the highest active input. For example, if inputs 0, 1 and 3 are active, the output is set to $S_1S_0 = 11$, corresponding to input 3

6.11.4 Registers

The components presented so far in this section are *combinatorial*. Once their inputs are changed, their outputs may also change. When values must be stored for later use, *sequential* logic is needed. Unlike combinatorial logic, sequential components can retain their output values even when their inputs change.

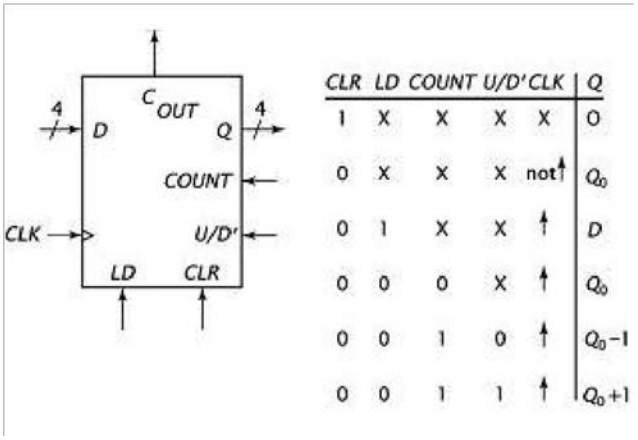
The most fundamental sequential components are the *registers*. They store one or more bits of data and make them available to other components. A 1-bit register is often called a *latch* or *flipflop*.

Most registers have a *clock* input. The clock input is usually derived from an oscillator or other circuit which alternates its output between 0 and 1. It is used to synchronize the flow of data in a digital system. When the clock input changes from 0 to 1, the data on the input of the register is loaded into it. The data is made available via output *Q*. Some variants also have a load signal (*LD*), which must be high as the clock changes from 0 to 1 in order for data to be loaded into the flip-flop.

6.11.5 Counters

A *counter* does exactly what its name implies, it counts. It stores a binary value and, when signaled to do so, arithmetically increments or decrements its value. Some counters do only one or the other, while other counters can do both. Like other registers, counters can be loaded with an externally supplied initial value. Some counters can also be cleared.

Figure 6.13 shows one implementation of a 4-bit counter, along with its truth table. It counts pulses at CLK input and stores their number as a 4-bit binary value. The *CLR* control signal clears the counter. Setting U/D to 1 or 0 increments or decrements the counter as the clock changes from 0 to 1. Signal C_{out} is set to 1 as the count increments from 1111 back to 0000. Note that a slash symbol below 4 at D and Q represents that there are four lines for four bits at the D input and Q output.



CLR	LD	COUNT	U/D'	CLK	Q
1	X	X	X	X	0
0	X	X	X	not↑	Q_0
0	1	X	X	↑	D
0	0	0	X	↑	Q_0
0	0	1	0	↑	Q_0-1
0	0	1	1	↑	Q_0+1

Figure 6.13: A 4-bit counter and its truth table

Sometimes we wish to have a counter that does not count in binary. Rather, it would count in decimal, outputting BCD values instead. There are special counters, called *decade counters*, designed just for this purpose. They count from zero to nine, and then go back to zero. Decade counters are useful in devices which require counting in decimal, such as digital clocks. We'll see how these counters can be used in the following section.

6.12 Application Note: Decimal Counter

In this section we will examine a simple decimal counter circuit. Our decimal counter will have the ability to count from 00 to 99 seconds, and its output will be displayed in BCD on LEDs.

We have already seen decade counters in the previous section. They count from 0 to 9, and then go back to zero. We could wire up an oscillator to cause this chip to count up once every second, but we'll use a pushbutton switch instead for simplicity.

This counter will work well for the one's digit of our counter, but what about the 10's digit? What can we use for that digit?

We can actually use the same type of counter for the tens digit. We just have to increment it differently. If we increment it using the pushbutton, as we did for the one's digit, our sequence would be 00, 11, 22, 33… 88, 99, which is not what we want at all. Clearly we need a different way to increment the ten's digit counter.

The key to solving this problem is to notice that the ten's digit is incremented whenever the ones digit goes from nine to zero. The decade counter has a carry output signal that is set to logic 1 when this occurs. We can then use that signal on the 1's counter to cause the counter of the 10's digit to increment. This is exactly how we want our counter to function. The circuit to do this is shown in Figure 6.14.

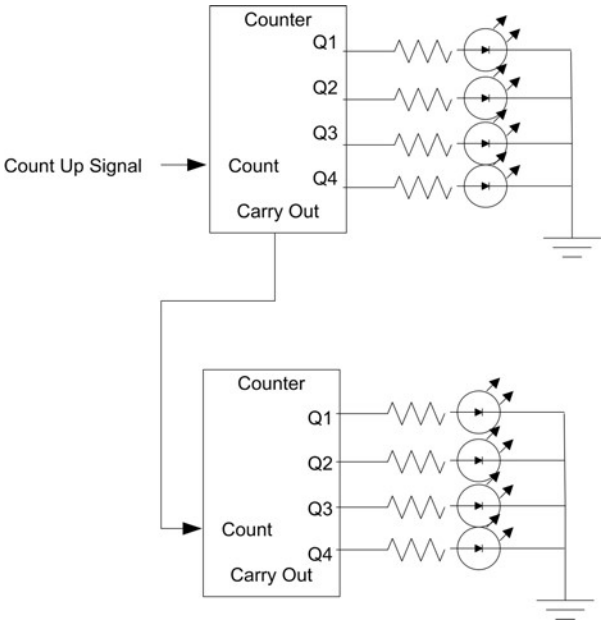


Figure 6.14: 2-digit decimal counter and display using LEDs

6.13 *Some Helpful information and Circuits*

6.13.1 Constant level pulse train

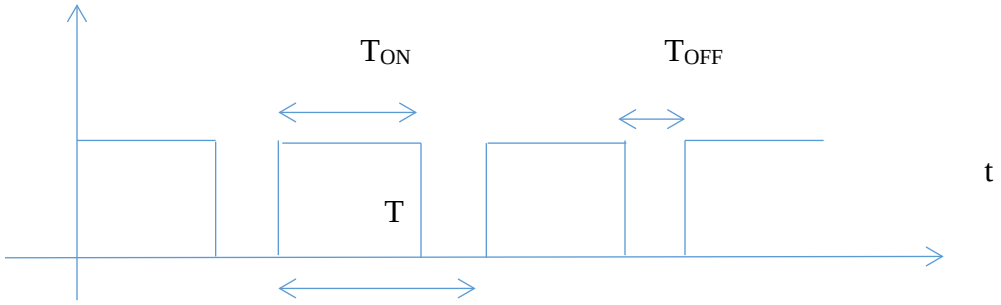


Figure 6.15: Constant Level Pulse Train

As shown in Figure 6.15, the signal is repetitive and has the form of a pulse train. The duration of a cycle (the smallest interval of time after which the signal repeats itself) is called a period, T. The unit for a period is a second (abbreviated sec or s) or its fraction, such as a millisecond (ms). The inverse of a period is the frequency f = 1/T. The unit for frequency is Hertz (Hz) equal to one period per second. Thus, if the period is equal to 1 ms, the frequency is 1 kHz. For example, the signal that brings electrical energy to our homes is a sinusoidal signal (also periodic) with frequency 60 Hz. Its period would then be T = 1/60 sec.
The duration over a cycle where a signal is high is labeled T_{ON}. The duration over a cycle where a signal is low (in the case shown above 0V) is labeled T_{OFF}. The ratio T_{ON}/T is defined as the duty cycle of the signal.

$d = (T_{ON}/T)*100\%$ (usually the duty cycle is given in percent).

In this case, if T_{ON} = 0.14 s and T = 0.21s, the duty cycle d would then be:

$d = (0.14/0.21)*100\% = 66.7\%$

6.13.2 The 555 timer

The 555 timer is a key component of multivibrators. The two most popular forms of a multivibrator are the monostable multivibrator and the astable (or free-running) multivibrator. At this point we will concentrate on the astable multivibrator. This configuration will be useful in generating clock signals (constant level pulse train). That signal could act as a clock signal for some of the integrated

circuits that you may need in your project, and could also help generate an audible tone through a speaker.

6.13.3 Astable Multivibrator

The 555 timer is at the heart of building these multivibrators. The circuit that generated the previous pulse train shown in Figure 6.15 is given in Figure 6.16

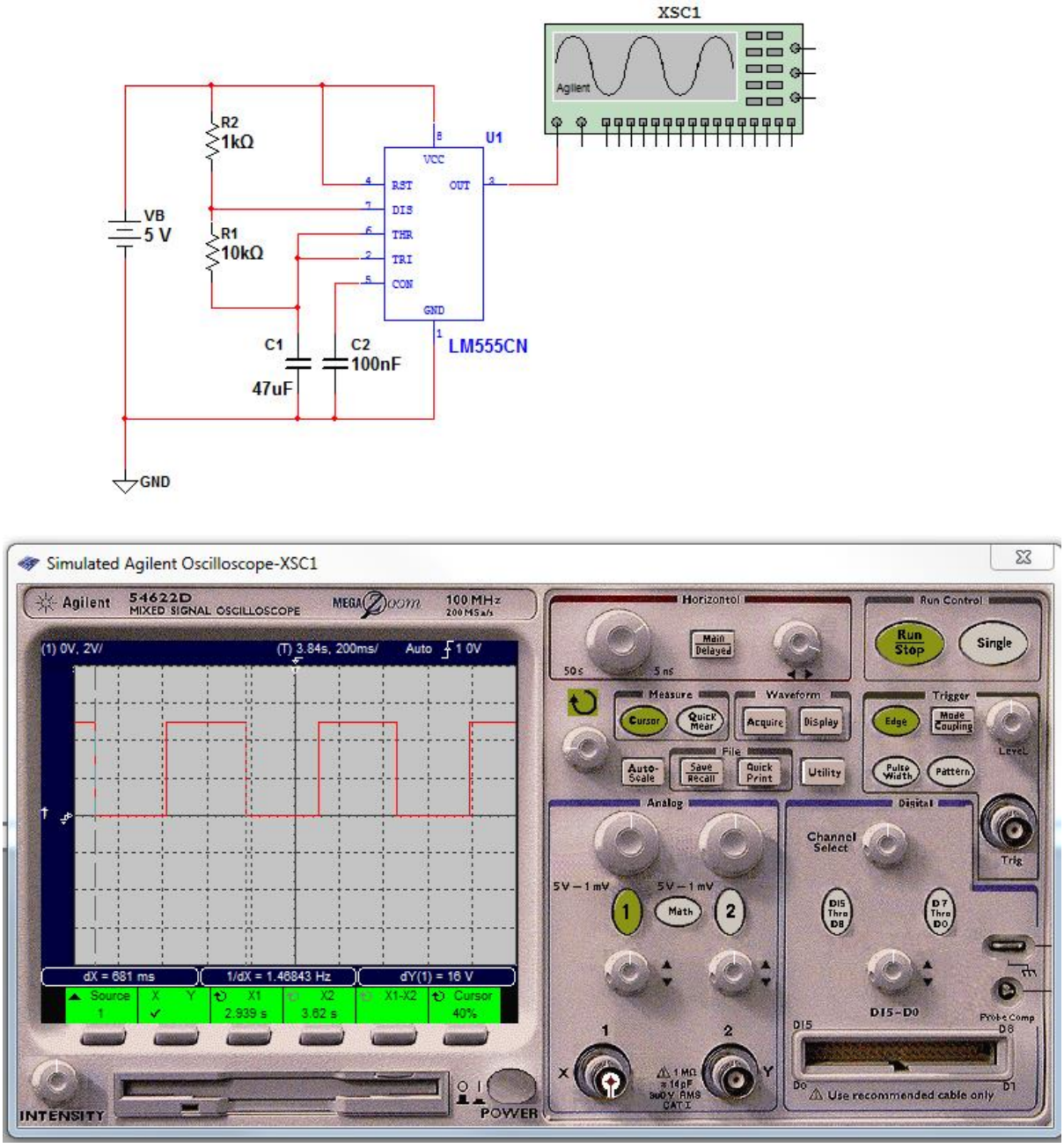


Figure 6.16: A possible astable multivibrator implementation

Note that the cursors X1 and X2 are set at 2.939s and 3.62s, covering one cycle. Hence

$T = 3.62 – 2.939 = 0.681s$

The timer circuit presented in Figure 6.16 obeys the following relationships:

$T_{ON} = 0.693 (R_2 + R_1) C_1$
$T_{OFF} = 0.693 R_1 C_1$
$T = 0.693 (R_2 + 2R_1) C_1$
$d = T_{ON}/T = (R_1 + R_2)/ (R_2 + 2R_1)*100\%$

Note that this circuit leads to a duty cycle that can never be less than 50%. If $R_1 >> R_2$ (>> means much greater than and 10-fold would qualify), then we can approach d = 50%, and the expression for T simplifies to

$T = 1.4R_1C_1$

Evaluating T for the values of the components from the circuit given in Figure 6.16, we obtain

$T = 0.693(10^3 + 2*10*10^3)*47\ 10^{-6}$
$T = 0.684s$

The values given by the calculation and simulation agree. Since in the case of the circuit given in Figure 6.16 R_1 can be considered much greater than R_2, the approximation to the period can be evaluated as follows:

$T = 1.4*10\ 10^3*47\ 10^{-6}$
$T = 0.658s$

This value is within $\frac{0.684 - 0.658}{0.684} * 100\%$ = 3.8% of the exact value.
It would also be beneficial to remember that the tolerance related to the components will have an effect on the accuracy of the achieved results. In industry, no component value will be obtained through measurement. We rely on the fact that the nominal values will lead to final results within an acceptable margin decided by the designer.

Other configurations will offer the possibility of a duty cycle covering all possible practical values (from 0 to 100%). An example of a circuit performing that task is shown in appendix B, with displays of 2 choices: 25% and 75%.

6.13.4 4-bit Binary Counter

The 4-bit binary counter that was selected to be used in this course is the 74LS163. We have presented in appendix C a portion of the information that the manufacturer (in this case Motorola) would be making available. In this course, you will ignore the inputs D to A (equivalent to Motorola

P_3 to P_0) and the output RCO (equivalent to Motorola TC). You will be able to understand most of the features and design parameters by the time you have taken the ECE 251 course.
At this level, this chip is capable of counting from 0 to 15 in binary format (from 0000 to 1111), and repeats itself thereafter.
The astable multivibrator that was introduced in Figure 6.16 generates a train of pulses which are counted by the 4-bit binary (chip 74163). The output of the counter is at the four terminals QD to QA, with QD representing the most significant bit. Note that the Motorola manual labels the outputs Q_3 to Q_0 (corresponding to QD to QA, respectively). LEDs attached to these terminals and connected through a resistor to ground (Figure 6.17) will display the state of the four outputs.

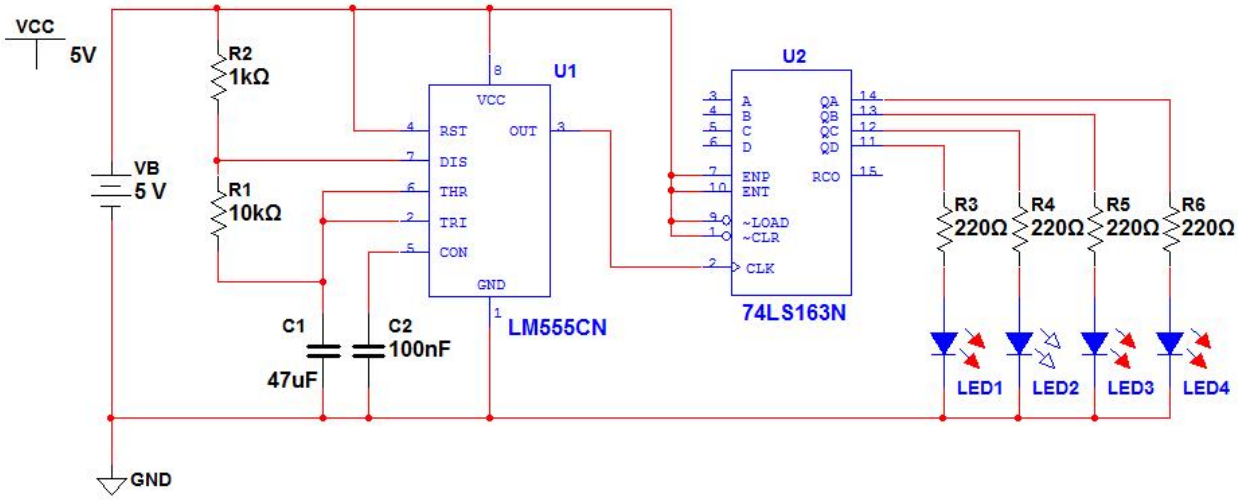
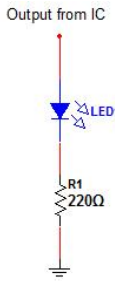


Figure 6.17: 4-bit Binary Counter based on the 74LS163

6.13.5 LED Protection

An LED is a diode that turns ON when a current flows through it (in this case, turning ON means lighting up). As all devices, and most particularly semiconductor devices, they are very sensitive to current limitations. If you exceed the maximum current that the manufacturers specify, you will destroy those components. Hence a **diode**, an **LED** or a **seven-segment display** is to always be connected **in series with a resistor** (usually 100 to 470 Ω, within the available standard values); the higher the resistance, the dimmer the display.

1. ***Active High Turn ON***



For this LED configuration, since the cathode is connected to ground (0 potential), a zero potential at the anode will keep the LED off, and a higher potential, for example 5 V, will turn on the LED. The higher the potential difference, the higher the current flowing into the LED will be generated, and the brighter the display. Keep in mind that if you exceed the maximum allowable current, the LED will be turned off permanently (break down).

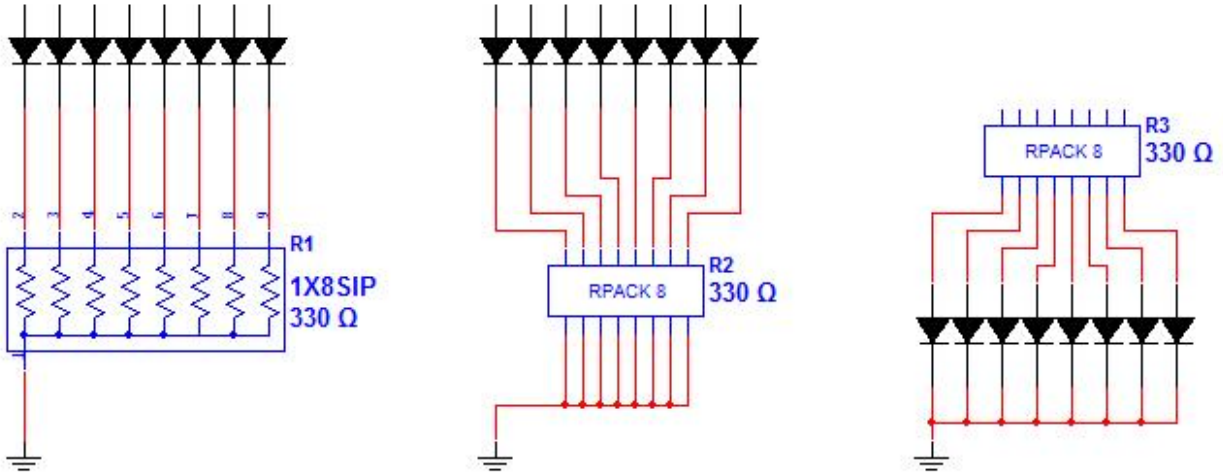
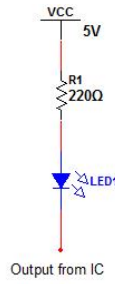


Figure 6.18: Active high turn ON with resistor packs (SIP, DIP)

2. Active Low Turn ON



When the anode is already connected to a higher potential, in this case 5 V, the way to turn on the LED is to apply a lower potential to the cathode, 0 V for example. Some experiments or projects will call for this configuration when a low needs to correspond to an LED turned ON.

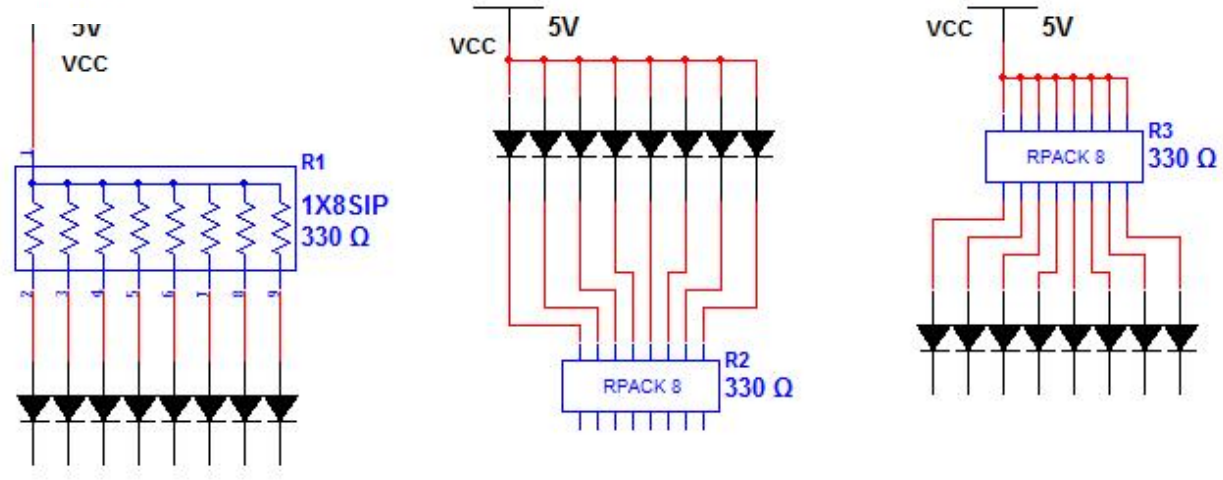


Figure 6.19: Active low turn ON with resistor packs (SIP, DIP)

7 Design

7.1 *Design Process*

Design is a systematic and creative application of scientific and mathematical principles to practical ends such manufacture, and operation of efficient and economical structures, machines, processes, and systems. Design is the most creative, challenging, and often satisfying task that engineers do. Quality of design determines usefulness of a product, process, or system, as well as its economic success. It also often impacts consumers and general public safety.

1. **Define the problem or need**. Identify what problem is to be solved, the shortcomings in current methods, and what is needed in new solutions.
2. **Explore previous or available methods**. Research existing technologies and methods that could be adapted to the design.
3. **Constraints**. Define the target values of the requirements, and what product attributes limitations, such as power, speed, etc., should be considered in the design.
4. **Design and analysis**. Design alternative solutions and analyze each to determine its fit within the requirements.
5. **Evaluation**. Carefully compare the results of each design analysis to the project's requirements and select the one that best meets them.
6. **Delegation**. Define and assign tasks for the design, management, and production of the product.
7. **Specification**. Define manufacturing parameters.
8. **Test**. Identify developmental testing procedures as well as qualification and acceptance tests.
9. **Prototype**. Build a prototype test it according to testing procedures, and determine what works, what doesn't, and what could work better.
10. **Final Product.** Determine if you have an acceptable result with the prototype. If not, redesign and retest.

In this course, you have an opportunity to participate with your team in design of an electrical circuit with specifically defined functionality. Your design process will be based on the knowledge you gained through lectures and assignments and on experience from the laboratory experiments. Of course, in school work you cannot follow all the design steps listed above, which are applicable to industrial and manufacturing environment. You will, however, use some of the most important among them. Hopefully, the process will be enjoyable and will convince you that overcoming design challenges and obtaining a working system is a very satisfying experience that you will want to repeat many times in your future engineering career.

7.2 *Design Project: The Traffic Light Controller*

The design project is an affirmation and a confirmation of your abilities as a designer and a creative engineer. Given a set of specifications that you or somebody else will set, you are to design a device that will satisfy them. You are to work as a team. The whole team will fail or succeed. Every team has a leader. A leader knows how to delegate the different tasks involved in achieving the desired

results. Each member is responsible for the success or failure of the team. The end result is worth all the pain and frustration that may arise in the process.

Your project is to design a traffic light controller made of 4 parts:

- Traffic light controller
- Audio signal
- Seven-segment display of remaining time
- Platform prototype construction

1. Traffic Light Controller

The traffic light controller satisfies the following specifications: given an intersection with EW (East-West) and NS (North-South) 2-way traffic, we want the lights to turn on in the following sequence

Time	EW	NS
0-6	Red	Green
7	Red	Yellow
8-14	Green	Red
15	Yellow	Red

Table 7.1: Traffic Light Sequence

The sequence lasts a total of 16 seconds. The numbers are chosen to simplify some of the design and implementation steps. It is obvious that a real practical case will have a longer sequence and also will allow more complex modules to be incorporated. The lights are red (stop) for the East-West direction and green (go) for the North-South direction during the first 0 to 6 seconds (a total of 7 seconds). The lights are still red (stop) for the EW direction and yellow (speed up? I don't think so) for the NS direction during the 8^{th} second (second 7). Table 7.1 describes the remaining steps. Time has been chosen to start with 0 to easily correlate time with the outputs of a counter.

A model platform representing the intersection will be available at the laboratory, represented with LEDs of appropriate colors. Your task is to design a circuit providing power from your protoboard to different LEDs in an appropriate sequence.

The next circuits are to be utilized as references but you can propose your own implementations as long as you can explain their function and their design within an acceptable margin.

Extra Credit: *Once you have developed a digital circuit to control the traffic lights and demonstrated the working circuit to your instructor, you are going to create a circuit and program using the Arduino Uno to control the lights. You will follow basically the same procedure you used for the boardwalk wheel, except this time you will be creating your own program from scratch. Demonstrate your working circuit and program to the instructor.* ***In your lab report, describe how the two designs differ and the relative advantages of each design.***

2. Audio signal

The audio sound circuit shown in Figure 7.1 is presented as a reference. The tone generated by U1 at a frequency

$$f = 1/(1.4R_2C_1) = 714.3 \text{ Hz}$$

will drive a speaker in series with a capacitor (1 μF to block the DC component) and a resistor (1kΩ for example to dim the sound) connected between the output (pin 3 of U1) and V_B. The timer based on U2 is used to generate the beeping effect where U2 will be enabled when the output of U2 is high, and the sound stops when the output of U2 is low. In reality, the frequency of U2 should have been set to be 1 Hz or T = 1s. In fact, the U2 timer is not needed since you already have built the timer related to the project based on the 1s period. Hence, you can use the output of that timer to control U1.

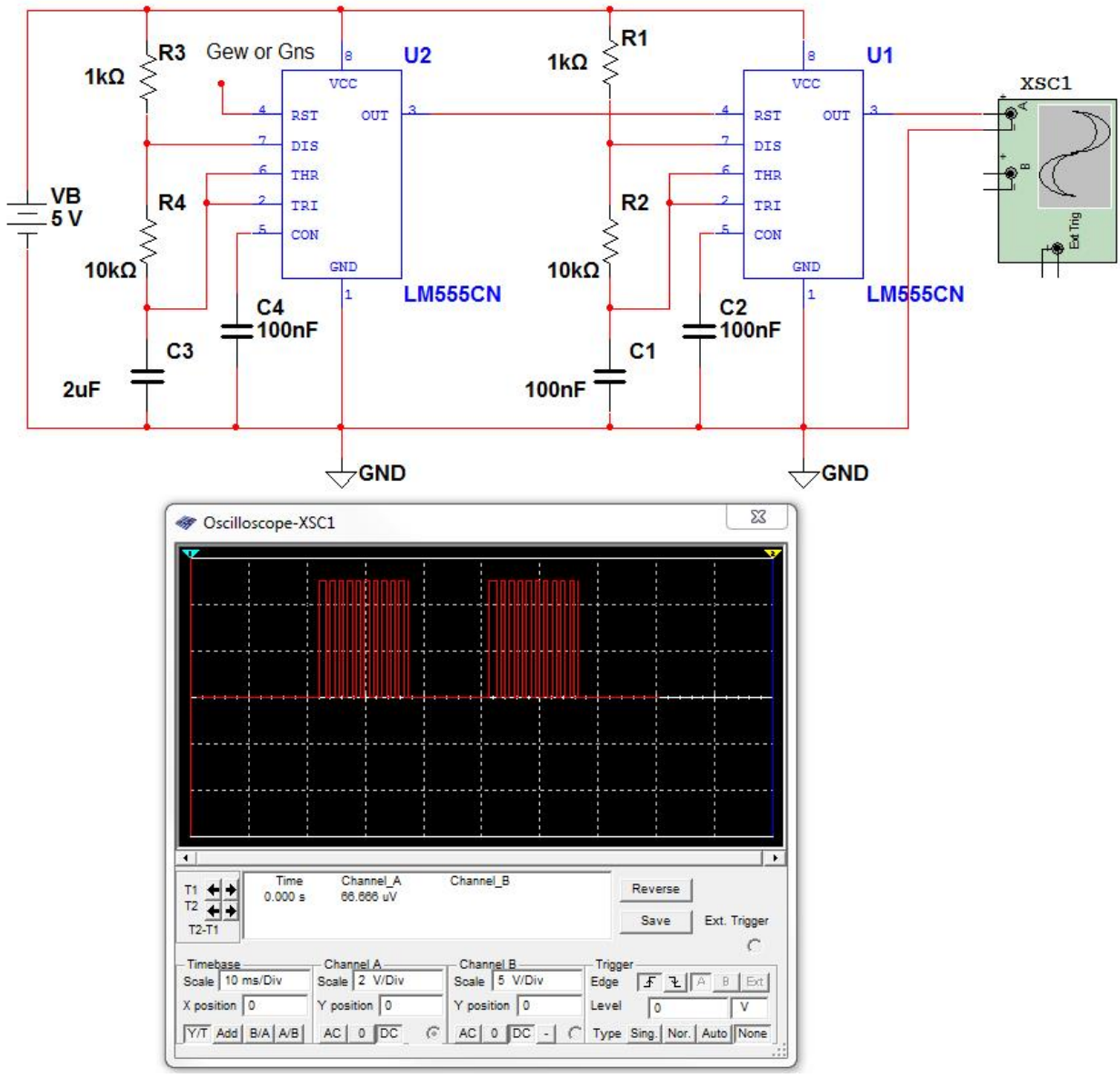


Figure 7.1: Beeping sound circuit

3. Seven-segment display of remaining time

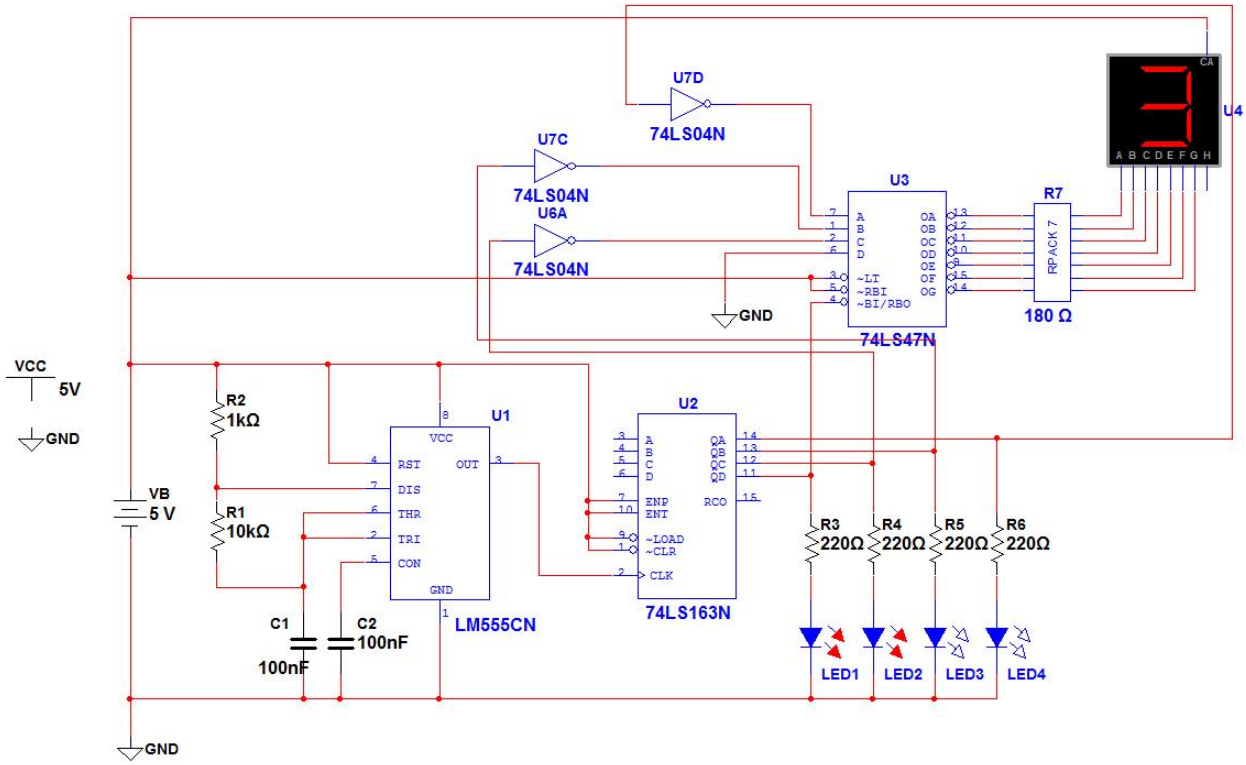
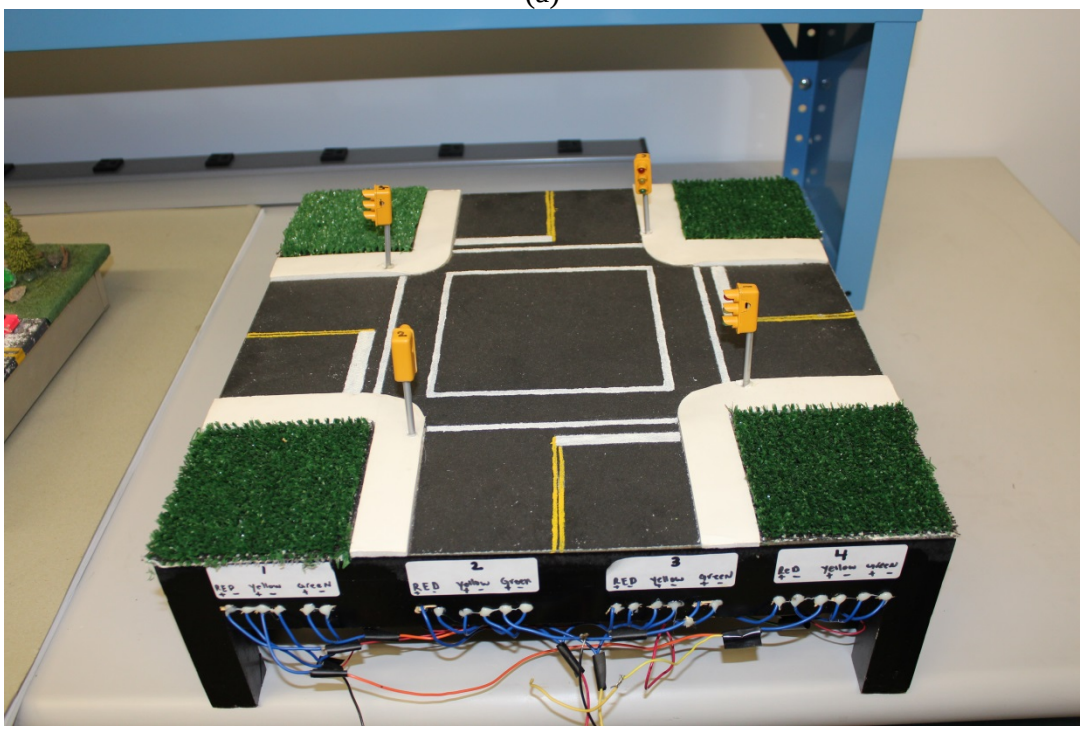


Figure 7.2: Counting circuit with seven-segment display

4. Platform prototype construction

(a)



(b)

Figure 7.3: Samples of platform prototypes

Each group needs to build to the best of their ability a platform that may represent the traffic light intersection. Only 2 lights are necessary (the limit is the sky though).

8 Laboratory Experiments

Experiment 1 – Resistors and 1-Resistor Circuits

In this experiment you will examine resistors and build and analyze a simple resistor circuit. You will also become acquainted with the digital multimeter that you will be using throughout this course.

Background material for this experiment can be found in chapter 2 of this manual.

PARTS LIST:

- Protoboard with built-in power supply
- Digital multimeter
- Various resistors and wires

Background - Protoboards

To perform this experiment, students will need to become familiar with the protoboard and the logic probe they will use. Both are described in this section.

We can construct a circuit using digital logic chips, but we need a surface to secure the chips and wires that comprise the circuit. This is the function of the protoboard. Chips and wires are plugged into the protoboard and can be rearranged easily as the design is modified. (Final circuits are wired on breadboards. Protoboards are typically used during the design of prototypes, hence its name.) A typical protoboard is shown in Figure L1.1.

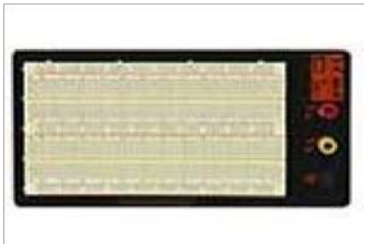


Figure L1.1: A typical protoboard

The terminals at the top of the protoboard are used to connect the power supply (Vcc and Gnd) to the circuit. One pair of cables connects the power supply to these terminals, and wires connect the terminals to the protoboards.

The main protoboard surface consists of a grid of holes. The pins of the chips in a circuit are plugged into these holes; for reasons that will be explored shortly, the chips must straddle the relatively large channels, or gutters, separating columns of holes. The chips cannot be placed in the top or bottom rows. These special power rows are reserved for Vcc and Gnd.

The protoboard has connections embedded within its structure so that certain holes are always connected to each other. As a result, wires and pins plugged in to connected holes are also connected. This greatly simplifies the task of wiring up circuits. Any hole is connected to all other

holes in the same row between the same two channels; this is illustrated in Figure L1.2. When dealing with integrated circuit chips later in the course, you will see that this is the reason that chips must straddle the gutters, to avoid inadvertently connecting pins on the same chip. The power columns are also connected, but in a different way. Here, holes are connected to other holes in the same long rows. Designers can connect wires from the two terminals connected to Vcc and Gnd to these rows, and then connect wires from these columns to the power pins of each chip. Note that the power rows are divided in two separate halves (not connected) in the middle of the protoboard; watch out for this when wiring up your circuits for this and future experiments. They also are not connected to the power posts; you have to do it with wires.

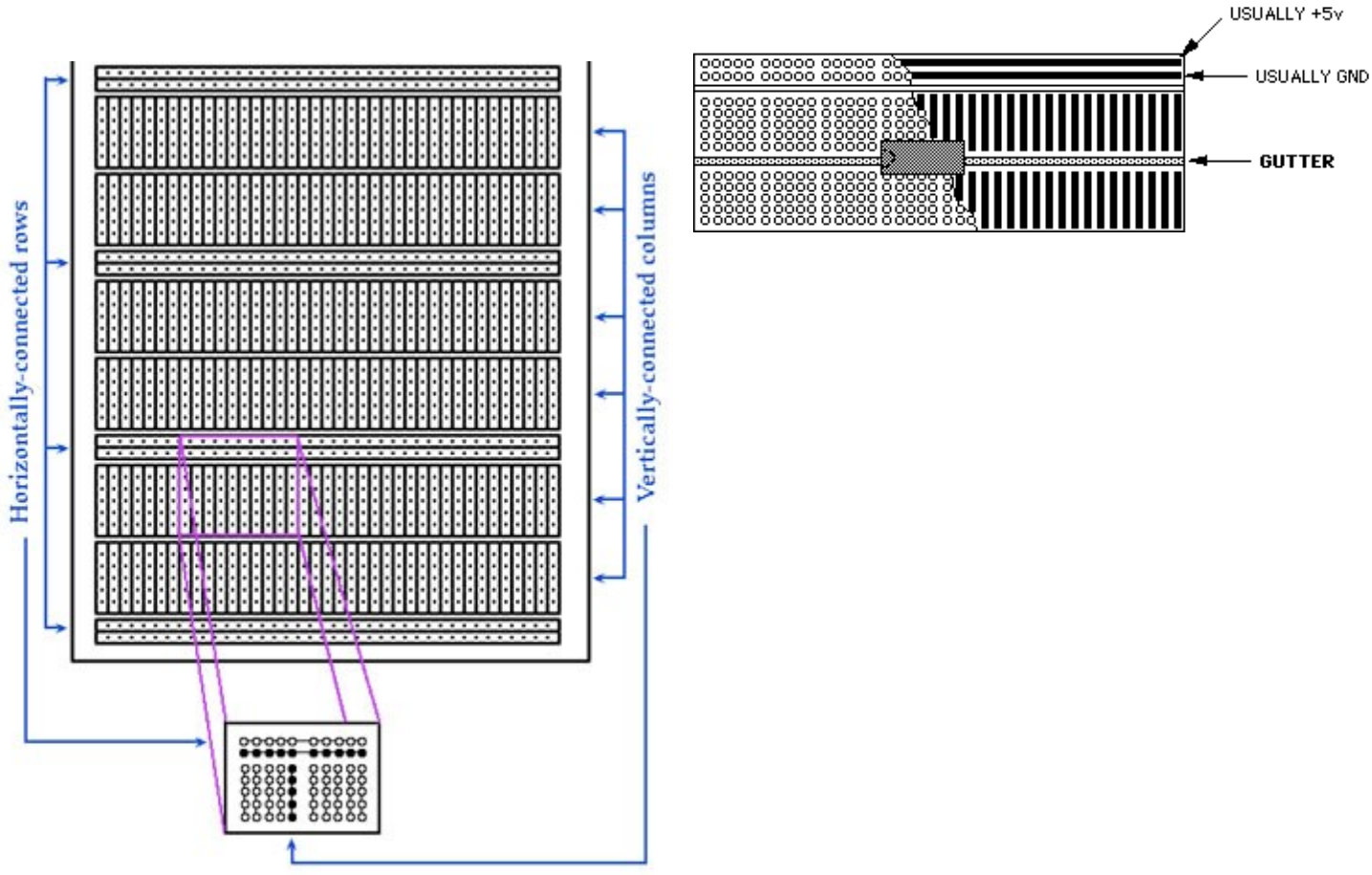


Figure L1.2: Protoboard internal connections. Connect two top horizontal rows(detail shown on the right) to the positive (red) and negative (black) power terminals.

Basic rule of wiring circuits:

DO NOT MAKE CONNECTIONS WHEN THE POWER IS ON – YOU CAN DAMAGE BOTH THE CIRCUIT PARTS AND THE PROTOBOARD.

Wiring steps:

- Make sure that the power switch of the protoboard is in the off position. Turn voltage adjustment knobs to minimum (counterclockwise).
- Connect long power rows with Vcc and Ground terminals.
- Place parts in the protoboard and connect them with wires
- Check the connection following the circuit schematic (always have it in front of you).
- If connections are correct, turn the power switch on. If adjustable power supply is used, connect the voltmeter to the power terminal and set the required voltage level by turning the voltage adjustment knob.
- If you need to change wiring, first turn the power off. **Remember the basic rule of wiring circuits (above).**

Part I – Resistor Measurements

Using the multimeter, measure the resistance of ten different resistors. Make a table showing a nominal value for each resistor (the value indicated by its color bands) and its measured value. In the last column of the table indicate if a resistor value is within its tolerance specification.

Part 2 – 1-Resistor Circuit

Wire up the circuit shown below. Using the multimeter, measure the **voltage across** the resistor. Also measure the **current flowing through** the resistor. Using Ohm's Law, verify that your measurements are valid for this circuit.
Make clear in your notes which are measured and which are calculated values.

Figure L1.3: 1-Resistor circuit

Note: Do not worry if you have trouble stetting the voltage at exactly 5.00 V, values such as 5.07 or 4.85 are close enough. It is unlikely that your resistor will have an exact value of 1.00 kΩ. No problem; measure it with the multimeter (on a resistance scale) and in the calculations use the real measured values of the voltage and resistance, not the nominal values shown in the figure.

Show the record of the experiment in the laboratory notebook at the end of the class.

Experiment 2 – Series and Parallel Resistance

In this experiment you will examine different resistor circuits. Using an ammeter, you will measure the current in the circuits to determine their overall resistances. You will also determine values for unknown resistances in several pre-wired circuits.

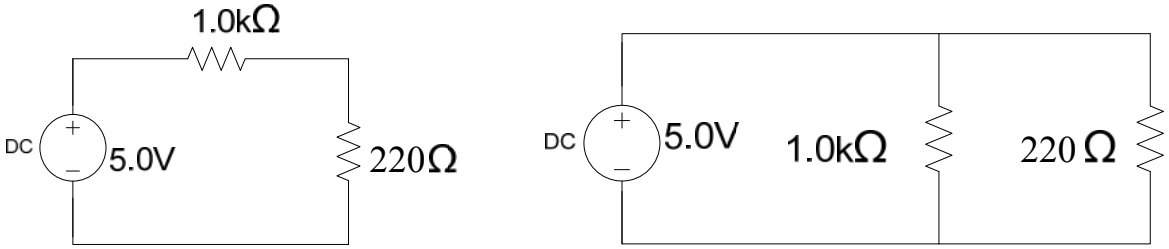
Background material for this experiment can be found in chapter 2 of this manual.

PARTS LIST:

- Protoboard
- Digital multimeter
- Various resistors and wires
- Black tape to cover "unknown" resistors

Part I – Circuits with known elements

In this part of the experiment, you will examine, wire, and test two known circuits. The schematics for these circuits are shown in Figure L2.1. Do not be concerned with setting the source voltage to 5.0 V exactly but measure the voltage precisely.



(a) (b)

Figure L2.1: Circuits to be tested

After you have wired up and powered up each circuit, connect the leads of the ammeter to measure the **current flowing through** the entire circuit. Record the current reading from the ammeter on the laboratory worksheet, and then use Ohm's Law to calculate the overall resistance of each circuit. Also measure the **voltage drop across** each resistor and note these values in your laboratory notebook. **The purpose of these measurements is to verify Kirchhoff's Current and Voltage Laws using your measured (not calculated) currents and voltages.**

Remember that a voltmeter is place across (in parallel) the component whose voltage you are trying to measure, and an ammeter is to be inserted (in series) in the path through which the current is flowing.

Part II – Circuits with an unknown element

This part uses the same circuits as the first part of this laboratory exercise, but with one of the resistors that is unknown to you. The unknown resistor will be covered with tape so you can't see its color code. Put the unknown resistor in series with a known resistor and measure only voltages across each resistor. You are not allowed to measure the current. Next, put the same unknown resistor in parallel with the known resistor and measure only the current through each resistor. You are not allowed to measure voltages.
Calculate the value of the unknown resistor from each of the two measurements separately and discuss how well they agree.

Show the record of the experiment in the laboratory notebook at the end of the class.

In the report, use measured values of currents and voltages (Part I) to confirm KVL and KCL**. How well your measurements agree with these laws (express any discrepancies in %)?** For Part 2 show calculations of the unknown resistor in the series and parallel circuits. **Compare the two results; how closely (%) they agree?**

Experiment 3 – Simulation Tools

In this experiment, students will use Multisim and MATLAB to simulate circuits previously analyzed in this class. You will need a PC running Multisim and MATLAB software

Background material for this experiment can be found in chapter 4 of this manual.

Part I – 1-Resistor Circuit

Using Multisim, "construct" the 1-resistor circuit you built and analyzed in Experiment 1. The schematic for this circuit is repeated below in Figure L3.1. The voltage and resistance in your experiment were most likely different than those shown in the figure; **use the values measured in your experiment.** Simulate the functioning of the circuit and "measure" the current flowing in the circuit and the voltage drop across the resistor. How do your simulation results compare to those you measured in Experiment 1?

Figure L3.1: 1-resistor circuit. The values of voltage and resistance are only examples; those in your experiment were most likely different.

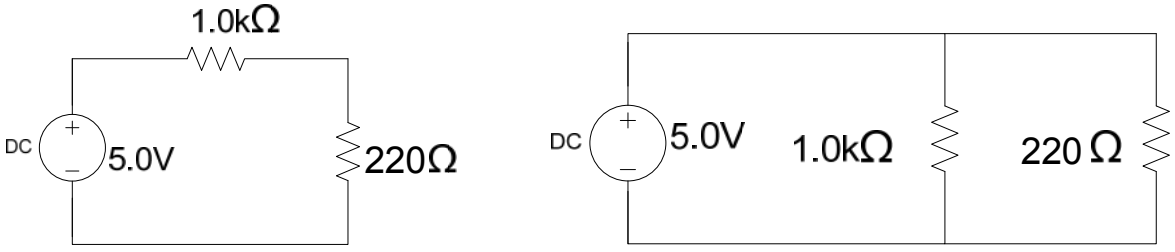


Figure L3.2: (a) Series and (b) parallel resistor circuits. The values of voltage and resistance are only examples; those in your experiment were most likely different.

Part 2 – Series and Parallel Resistor Circuits

Next simulate the series and parallel resistor circuits that you circuits you built and tested in Experiment 2. They are shown in Figure L3.2 with nominal resistors and voltage values. In simulations, use the values form your experiments. Again simulate the **current through** and **voltage across** each resistor. In the parallel circuit simulate also the current flowing from and back to the source.

In the report discuss how close are the results of the simulation to the values you measured in Experiment 2. Check also if the simulations confirm KVL and KCL.

Part 3 – Simulation of Complex Circuits

In Part 1 and Part 2 of this exercise the advantage of computer simulation was not so clear. You had to learn a new program to get results that you could obtain using pen and pencil, without too much difficulty. But most today's electrical circuits are much more complex than the examples above. They may contain thousands or millions of different elements. To get a taste of the power of simulation, build a more complex circuit using 6-10 resistors, or so, in a configuration that would be difficult to analyze "by hand". Show through Multisim the measured of a current through one resistor, and a voltage across another resistor.

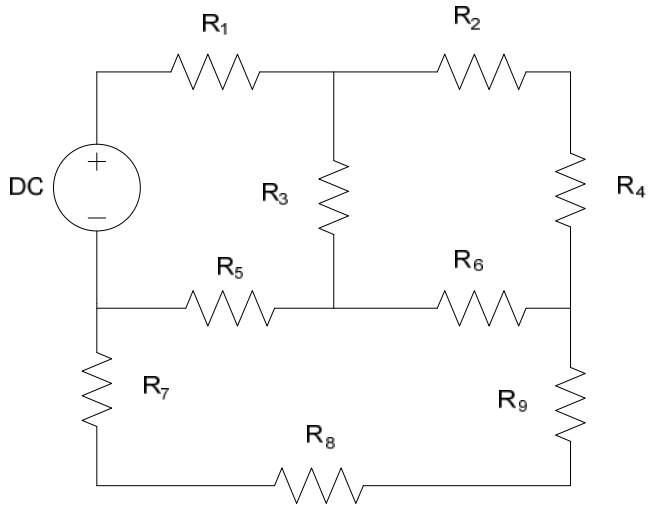


Figure L3.3 An example of a complex 9 resistor circuit.

Experiment 4 – Transistors and Diodes

In this laboratory exercises, students will examine operation of diodes and use them in constructing simple logic gates. Operation of BJT and MOSFET will also be demonstrated.
Background material for this experiment can be found in chapter 3 of this manual.

PARTS LIST:

- Protoboard with built-in power supply
- Digital multimeter
- Light bulb
- Resistors and a potentiometer
- Rectifier diode, LEDs, BJT (npn) and n-type MOSFET.

Part 1 – Diodes - forward and reverse current

Set the voltage on one of the 15 V terminal to 6 V. Connect a light bulb a diode in series and connect them to the power supply terminal through an am-meter. Measure the current with the diode connected in the forward direction (the bulb lights up) and the reverse directions (the lamp is dark). To change the diode polarity, simply exchange the diode terminals on the protoboard (flip the diode position). What is the ratio of the reverse to the forward current? In an ideal diode it should be zero. Repeat the experiment using an LED in series with a 220 Ω or 330 Ω resistor (not with the light bulb!). The resistor is necessary to limit the current, preventing the diode from burning out. **What polarity is needed for emitting light, forward or reverse?**

Part 2 – Operation of a BJT transistor

In this laboratory exercise we will examine both switching operation and amplification of transistors. Schematics of connections to MOS and BJT transistors are shown in Figure L4.1.

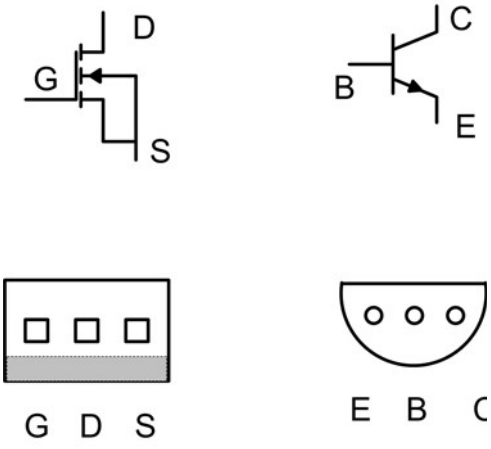


Figure L4.1: Schematics (top) and terminal connections (bottom) seen from the leads side of MOS (left) and BJT (right) transistors

Assemble a circuit (shown below) in which an n-p-n BJT is used to switch on and off a small incandescent lamp. Resistor R_1 should be a few k ohms. Switching a switch S should turn on or off the lamp.

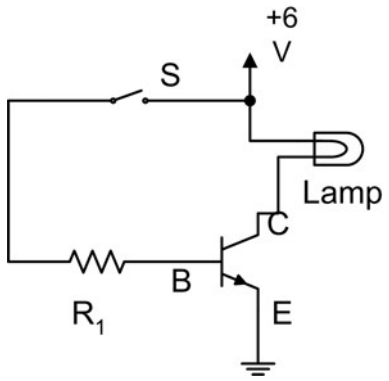


Figure L4.2: A circuit switching current to a lamp with a BJT

In the BJT, the current flows between the collector (C), which is positive, and the emitter (E) while the base (B) is the control electrode. Note that the schematic really represents two circuits (or circuit loops) connected by a transistor. One of them, the base circuit, consists of the power supply, the switch (S), the resistor R_1, and the base and the emitter of the transistor, which is connected to ground closing the loop to the power supply. The other circuit uses the same 6V power supply, connected to the lamp and through the collector (C) and emitter (E) of the BJT, closing the loop to ground. The switch turns on or off only the current to the base (in the first circuit) but the current through the lamp (in the second loop) responds to the condition of the first circuit.

After successful demonstration of the circuit, measure and compare currents in the two loops (the base current and the collector current. **How many times greater is the collector current then the base current?** You should be able to demonstrate GAIN, a great asset of the transistor. Big currents (for example, motors of an electric train) can be adjusted by small currents from electronic control circuits.

A simple demonstration of such control can be accomplished using our lamp and a potentiometer in the circuit shown in Figure L4.3 below. The potentiometer (P) in the base circuit controls the light output from the lamp in the collector circuit. Resistor R_2 (330 ohm) protects the base circuit from excessive current when the potentiometer is turned all the way up.

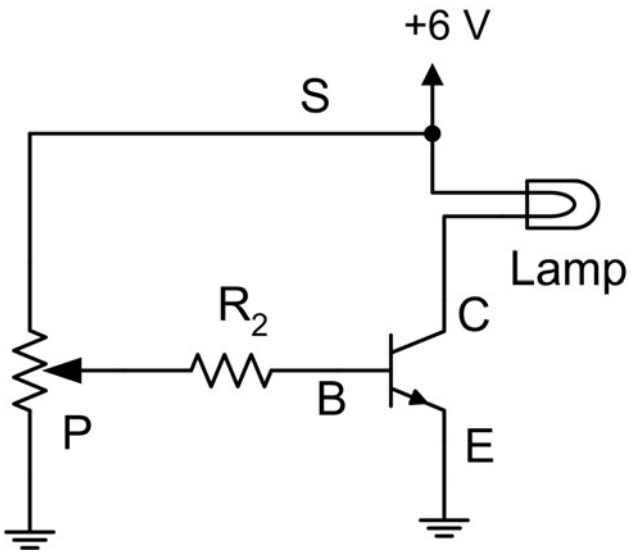


Figure L4.3: A light dimmer circuit with a potentiometer and a BJT

Part 4 – MOSFETs

In the circuit represented in Figure L4.2, replace the BJT with an n-channel enhancement mode MOSFET: the gate terminal replaces the base, the source replaces the emitter and the drain the collector, so that the drain is positive (see Figure L7.2). You do not need a switch and the resistor R_1, which greatly simplifies the circuit. Turn the lamp "on" and "off" by connecting the wire from the transistor gate either to the positive power supply terminal or to the ground.

Now a surprise: touch the end of the wire from the MOSFET gate terminal with one hand and touch your other hand to the ground and then to the positive terminal. This demonstrates a big advantage of MOSFET. **Do you need much current to turn the MOSFET "on"?**

In the report address questions shown in bold print in this part of the manual. Comment on the role of diodes and transistors in electrical circuits and on the important difference between BJT and MOS transistors.

Experiment 5 – Digital Logic Fundamentals

In this laboratory exercise, students will test several logic chips to verify their functions. Students will also test an "unknown" chip to determine the Boolean function it realizes.
Background material for this and the following experiments, as well as chips connections layout, can be found in chapter 6 of this manual.

PARTS LIST:

- Protoboard with built-in power supply
- Digital multimeter
- Logic probe
- 7408 (AND gates), 7432 (OR gates), and 7486 (XOR gates) chips and wires
- One additional TTL chip, either 7400, 7402, 7404, 7408, 7432, or 7486 Black tape to cover one "unknown" chip

Background – Logic Probe

To perform this experiment, students will need to become familiar with the logic probe they will use. This is described in this section.

We will use a logic probe to test the functions of our chips. First we must connect the logic probe's Vcc and Gnd clips so the probe has power. We can clip them directly to the terminals of the protoboard. Then we touch the tip of the logic probe to the pin we wish to test. The logic probe has two LEDs that will be used in this experiment. When the probe detects a 0, its "Low" LED lights up; the "High" LED lights when a logic 1 is detected. We can use the logic probe to check the input and output values of a chip. Figure L5.1 shows a typical logic probe.

Figure L5.1: Logic probe

To test our chips, we will need to input values to the gates. To input a 1, we will connect a wire between the chip's input and Vcc. We can input a 0 by connecting the wire from the chip's input to Ground.

<u>An important point students sometimes forget</u>:
Each chip has to be supplied with power, which means that designated pins (not only inputs and outputs) must be connected to Vcc and Ground. For example in chips such as 7408 pin 7 is ground and pin 14 is Vcc. If these connections are not made the chip will not work.

Part 1 – Testing Known Chips

In the first part of this exercise, students will test several logic chips. Each group of students has a protoboard and three known chips: a 7408 (AND gates), 7432 (OR gates), and 7486 (XOR gates). Place the chips into the protoboard, ensuring that each chip straddles one of the channels. This ensures that there is no direct connection between the leads on the two sides of the chips. Connect Vcc and Gnd for each chip.

Starting with the AND gate, connect both inputs to Gnd, producing a logic 0 for each input. Use the logic probe to verify these input values, and then check the output values for the gate. Record your output values on the worksheet for this exercise.

Next, change one of the inputs to Vcc, changing it to logic 1. Again verify your input values and determine your output value using the logic probe, and record your output value on the worksheet. Repeat this for all possible combinations of the input values, and then verify that your values match those in the truth table for the AND function.

Repeat this procedure for the OR and XOR gates.

Part 2 – Testing an Unknown Chip

The protoboard for this exercise also contains a chip which has tape covering its surface. Here you will determine its function without removing the tape.

To do this, follow the same procedure you used in the first part of this exercise. Vary the values of the inputs to the chip and record its outputs on the worksheet. Then compare the truth table for your unknown chip to those given earlier in this chapter and find the table that matches the unknown chip's table. What is the function (type) of your unknown chip?

Test Results

AND gate

A	*B*
0	0
0	1

OR gate

A	*B*
0	0
0	1

A	*B*	O
0	0	
0	1	

XOR gate

A	*B*	Output
0	0	
0	1	

Unknown gate

1	0		1	0		1	0			1	0
1	1		1	1		1	1			1	1

GATE TYPE _____

Experiment 6 – Guess the Pattern Game

In this experiment, we will design a "guess the pattern" game. One player will set three inputs, *A*, *B*, and *C*, to any combination of binary values. This player does not show the pattern to the other player. The other player sets three other inputs, *D*, *E*, and *F*, to some combination of binary values. If *A*,*B*,*C* = *D*,*E*,*F*, that is, *A* = *D*, *B* = *E*, and *C* = *F*, then the game lights a single light, an LED in this case. If not, the LED is not lit.

PARTS LIST:

- Protoboard with built-in power supply
- Digital multimeter
- One 7486 chip and one 7408 chip and wires
- One LED and 330Ω resistor

To design this game, we first design a circuit to check whether or not *A* and *D* are equal. First complete the truth table in Table L6.1(a). Enter all possible combinations of values for *A* and *D*; then enter an output value of 1 if *A* and *D* are the same, or 0 if they are not. Next, determine a function for this truth table. Repeat this to develop functions that check if *B* and *E* are equal, and if *C* and *F* are identical, using Tables L6.1(b) and (c).

Table L6.1: Truth tables for the three comparisons (a) *A* = *D*, (b) *B* = *E*, and (c) *C* = *F*

A	*D*	(*A*	*B*	*E*	(*B*	*C*	*F*	(*C* = *F*)

(a) (b) (c)

Now that you have an equation for each of our three comparisons, the next task is to design three logic circuits to generate the functions *A* = *D*, *B* = *E*, and *C* = *F*. You should be able to generate each of these functions using either:

- two AND gates, two NOT gates, and one OR gate, or
- one XOR gate and one NOT gate.

(You could also use one XNOR gate, but we won't do that here. The 74266 chip is a special type of chip that uses *open collector* outputs. We could use that chip, but we would have to connect additional resistors to make them work with our other chips. For now, consider it sufficient that we will use only regular, non-open collector chips to simplify our design.)

The design of the three circuits should be identical, except for their different input and output values, since each compares two different inputs to see if they are equal.

After designing and building these three circuits, test them using a logic probe. Vary the inputs to check every combination in the truth tables. Verify that each circuit correctly outputs a 1 whenever and only when it inputs are the same.

Now we must take the three circuit outputs and combine them, outputting a 1 only when ($A = D$), ($B = E$), and ($C = F$) are all 1. First complete the truth table in Table L6.2. The inputs ($A = D$), ($B = E$), and ($C = F$) would be generated by the circuits we just designed. The truth table should have output ($ABC = DEF$) equal to 1 only when all three inputs are 1.

Table L6.2: Truth table for the overall function ($ABC = DEF$)

$(A = D)$	$(B = E)$	$(C = F)$	$(ABC = DEF)$

Next we design a circuit to take these three inputs and output a 1 only when all three are 1. This is a relatively straightforward design. After designing this circuit, test all possible combinations of input values with a logic probe to verify that it generates an output of 1 only when all three inputs are 1.

Now connect the sections of your circuits together as shown in Figure L6.1. The first three circuits check whether $A = D$, $B = E$, and $C = F$. The outputs of these circuits are input to the last circuit, which checks whether all three pairs of bits are the same. It outputs a 1 if the player has guessed the pre-set pattern, or 0 otherwise.

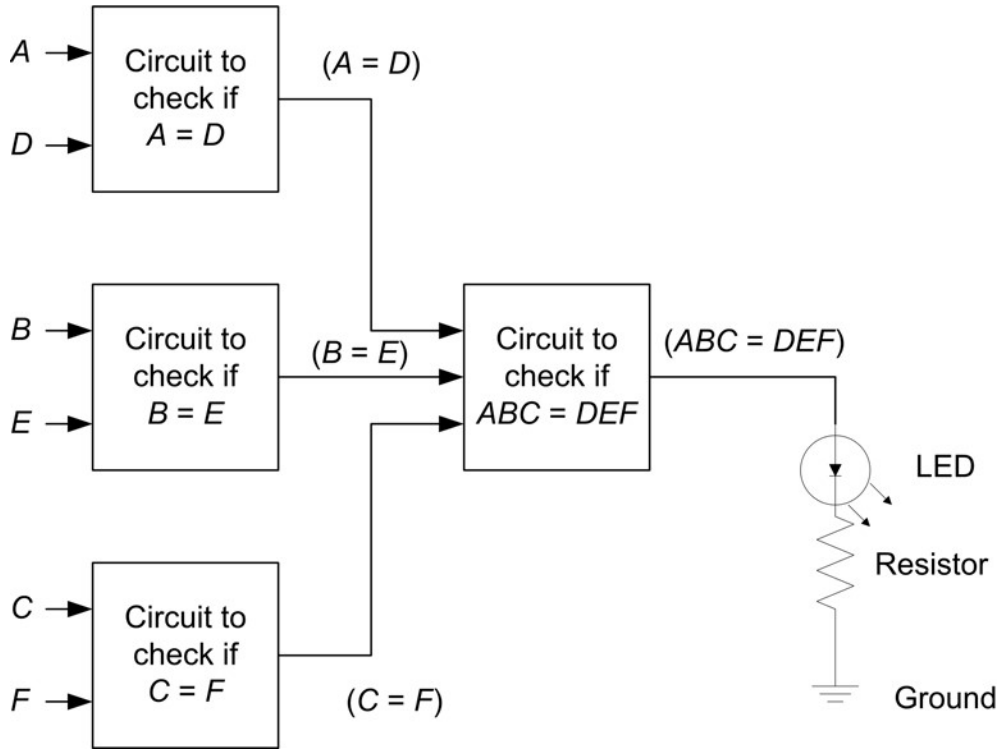


Figure L6.1: Block diagram of the "guess the pattern" game

Finally, we connect the (*ABC = DEF*) output to the LED, as shown in Figure L6.1. Note that the direction of the LED is important. The LED has two different length leads. The longer lead should be connected to the circuit that checks if ABC = DEF and the shorter lead should be connected to the resistor. Notice the resistor in this circuit. This is needed to keep the LED from receiving too much current and burning out.

After designing and building the circuit, demonstrate it to the instructor.

In the report describe briefly the design procedure, from the truth table to the logic functions and finally the logic gates and their connections. Include a schematic of the final circuit.

Experiment 7 – Boardwalk Wheel

The Boardwalk wheel is a game of chance. It is a wheel with a pointer at its center; the outside of the wheel is divided into many different spaces. Players place their bets on which space the pointer will point to when it is done, and the pointer of the wheel is spun. It eventually slows down and comes to a stop. Whatever space the pointer points to when it stops is the winner. Here in New Jersey, such wheels are common in the amusement areas of boardwalks near the beaches, but they are also found at carnivals and amusement areas throughout the world.

In this experiment, we will create an electronic version of the Boardwalk wheel. There will be only eight spaces on our wheel. Instead of using a pointer, each space will have a single LED.

PARTS LIST:

- Protoboard with built-in power supply
- Digital multimeter
- 4-bit counter
- One chip described in section 6.12 of this manual (to be selected by the student)
- Eight LEDs and 330Ω resistors and wires
- One 555 timer chip and components to generate a 1-second clock signal

To design this wheel, we must create a circuit that lights one LED, then the next, and so on, until the "wheel" eventually stops with one LED lit. This design can be created using only two chips: a counter and one other chip.

A counter can generate a sequence of outputs that varies from 0000 to 1111. If we only concern ourselves with the three low-order bits, it will generate the sequence 000, 001, 010, 011, 100, 101, 110, 111, and then start the sequence over again. To do this, we need to generate a clock input to the counter. This can be done using the 555 timer; see section 6.14 of this manual for information on how the 555 timer can be configured to generate a train of pulses (such as in Fig. 6.17). It is recommended to use initially the values R1 = 1k, R2 = 10k and C1 = 47μ which are different than those shown in Fig. 6.16 and Fig. 6.18. Check that the timer period agrees approximately with the equation in section 6.14.3.

The counter chip (74163) should be connected to the power supply (VCC) through pin 16 and to ground through pin 8. Use pin 3 as the input to the counter. Pins11 to 14 give four bits of the counter output. To check that the counter works correctly, initially connect 4 LEDs to these pins (with protective resistors to ground, as usual). The LEDs should light up in binary bit sequence representing decimal values 0 to 15.

The circuit we are building has eight LEDs, so we need only 3 bits form the counter. The LEDs, however, should light up sequentially one at a time, therefore we also must use one of the components described earlier in this chapter to convert the three-bit value into eight separate signals. Only one signal should be high when the inputs are 000, another is when they are 001, and so on. These eight signals will each light one LED. What component can perform this task? *Hint: examine the chips described in section 6.11*

Design and wire your circuit. Connect it to the pre-wired LED "wheel" and verify that it works properly.

Demonstrate working circuit to the instructor.

Once your digital circuit functions properly and has been demonstrated to the instructor, you are going to create a different design to accomplish the same function. Instead of using a digital circuit, you will program an Arduino Uno to accomplish the same task. The Arduino Uno is a small computer board that is well-suited to embedded applications such as the boardwalk wheel. Refer to Appendix E for a description of the Arduino Uno, its programming software, and how to interface it to external hardware. Note that the Arduino IDE software has been installed previously on the computers in the FED 101 labs; the installation instructions are provided for students who wish to download and use the software on their own computers.

This appendix also provides the code needed to control the boardwalk wheel, as well as the connections you will need to make between the Arduino and the LEDs for this design. Program the Arduino using the supplied code, connect the I/O pins of the Arduino to the LEDs as specified in the Appendix, and demonstrate the working boardwalk wheel circuit to your instructor.

In your lab report, describe how the two designs differ and the relative advantages of each design.

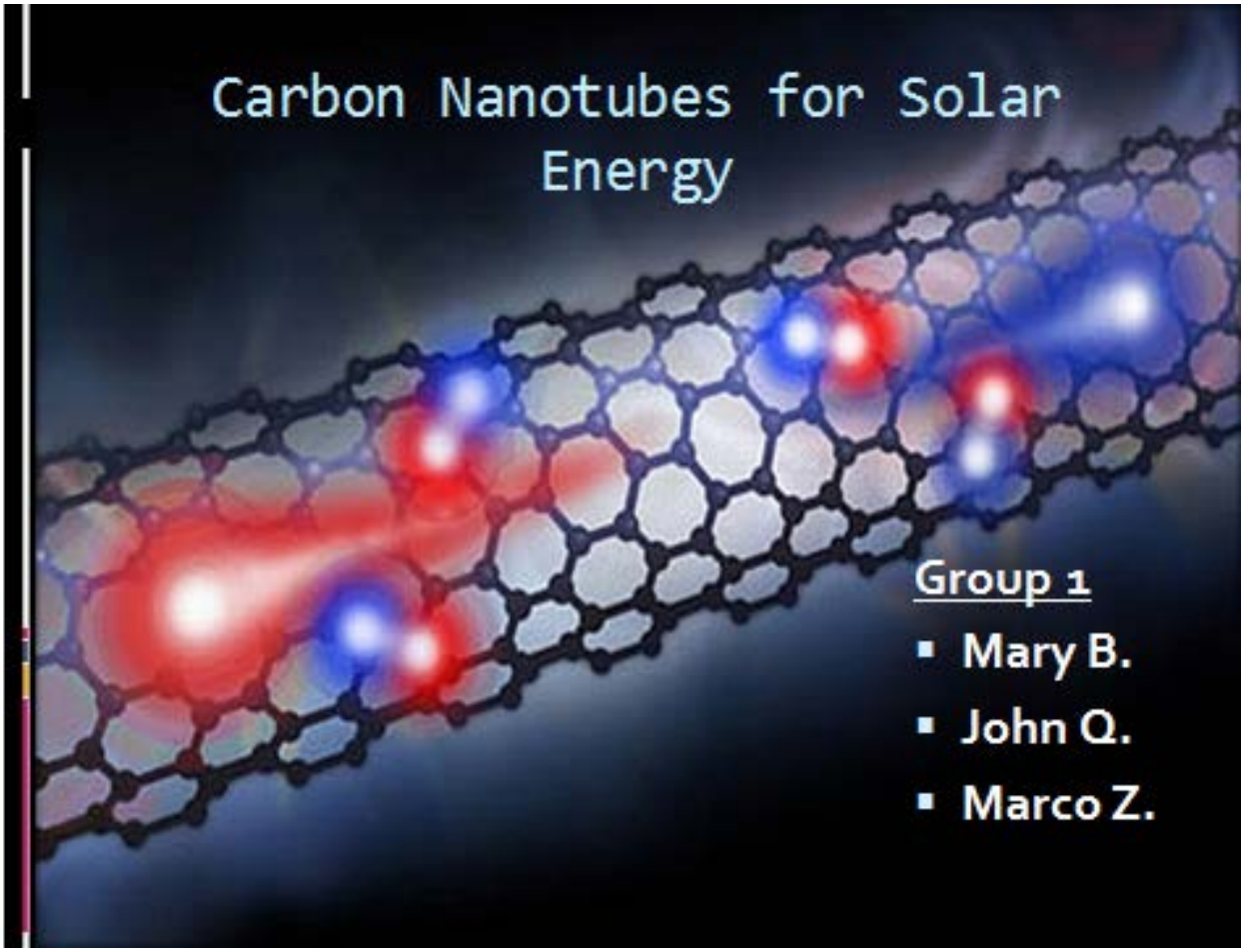
Extra Credit: After you have gotten this circuit working, modify the code to create a different lighting sequence for the LEDs. Be creative when developing this pattern, and describe the pattern to the instructor before demonstrating this design.

APPENDIX A

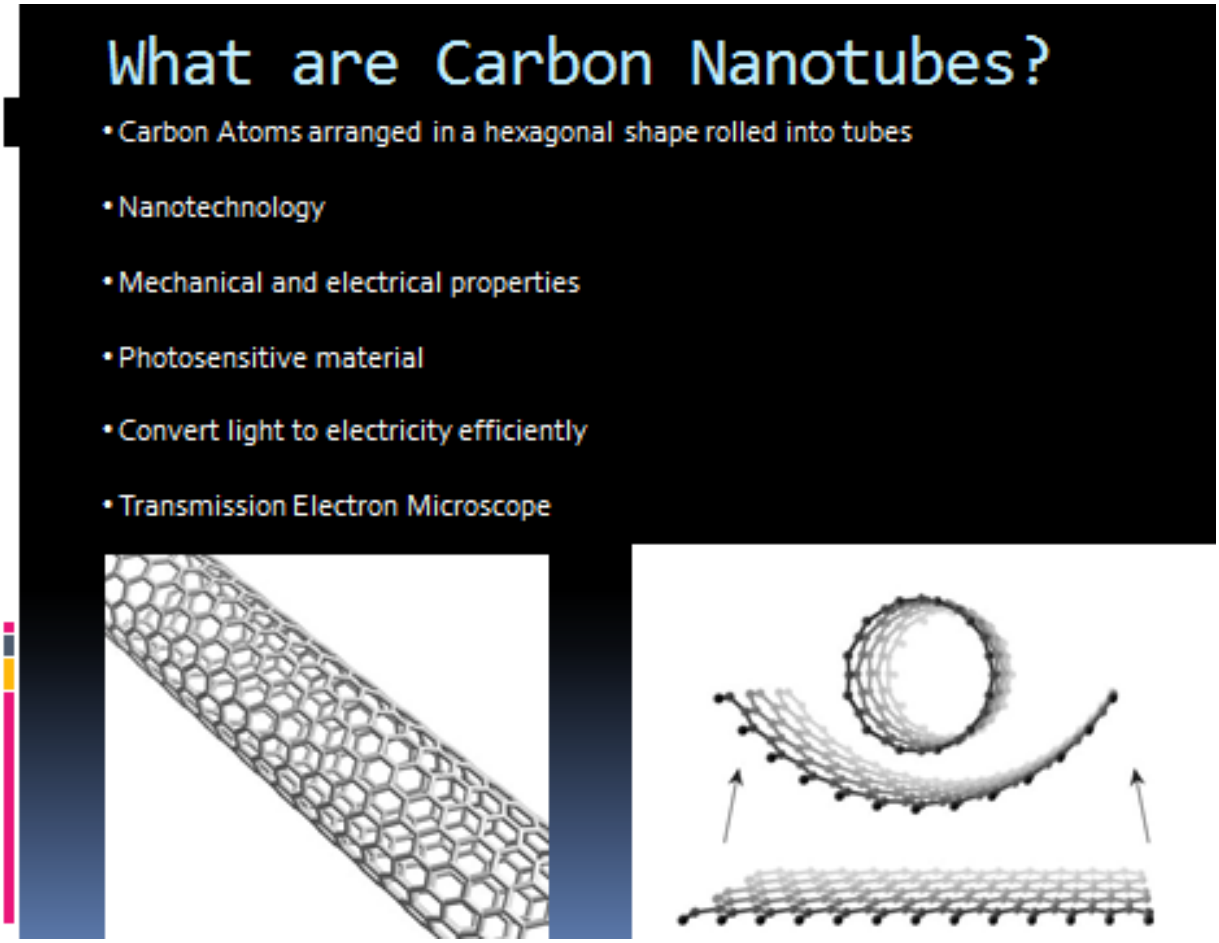
Example of a power point presentation

The following power point presentation has been written by some of your peers in the past, and deals with the utilization of carbon nanotubes to manufacture solar panels. The presentation in this case relies on nine slides, where each slide contains mostly information through bullets. You may have additional text in flash cards to have more details to present to the audience. During the presentation, do not read from the flashcards but use your own words as much as you can. The text is just a guide and a resource. If you use the original version of the material prepared by somebody else, you have to reference it appropriately, with the text included within quotation marks. For presentation outside the classroom, often obtaining the author's permission is necessary to comply with the copyright laws.

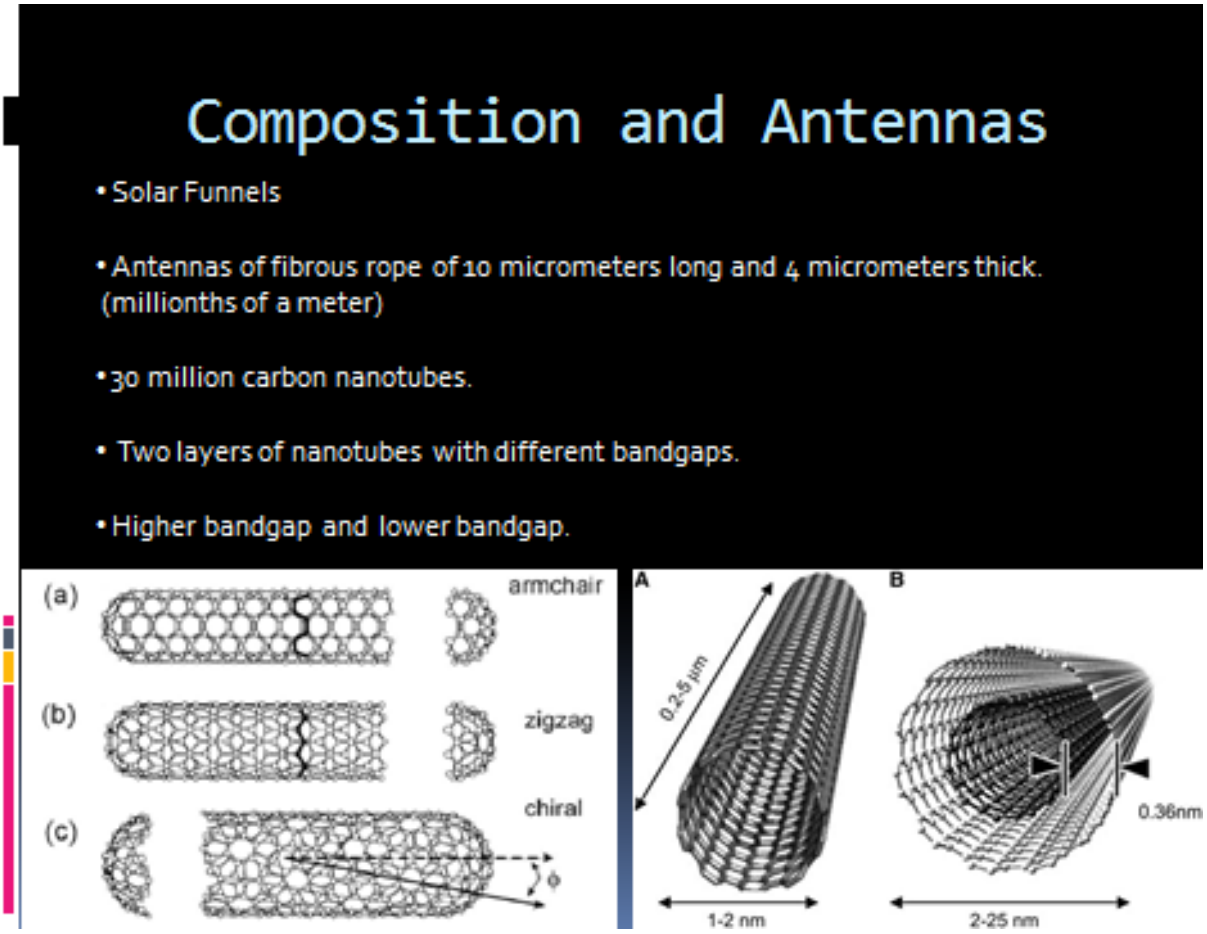
Slide 1



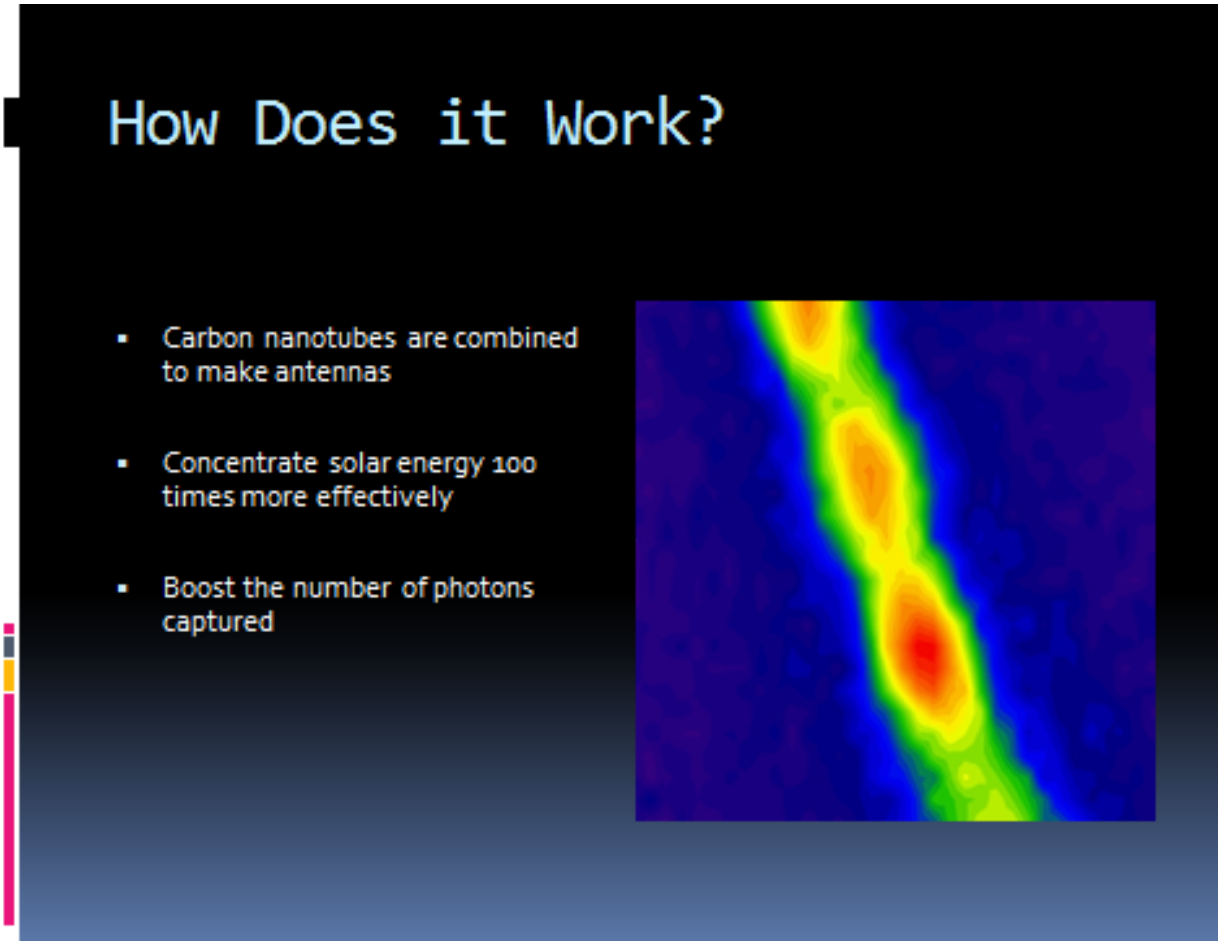
Slide 2



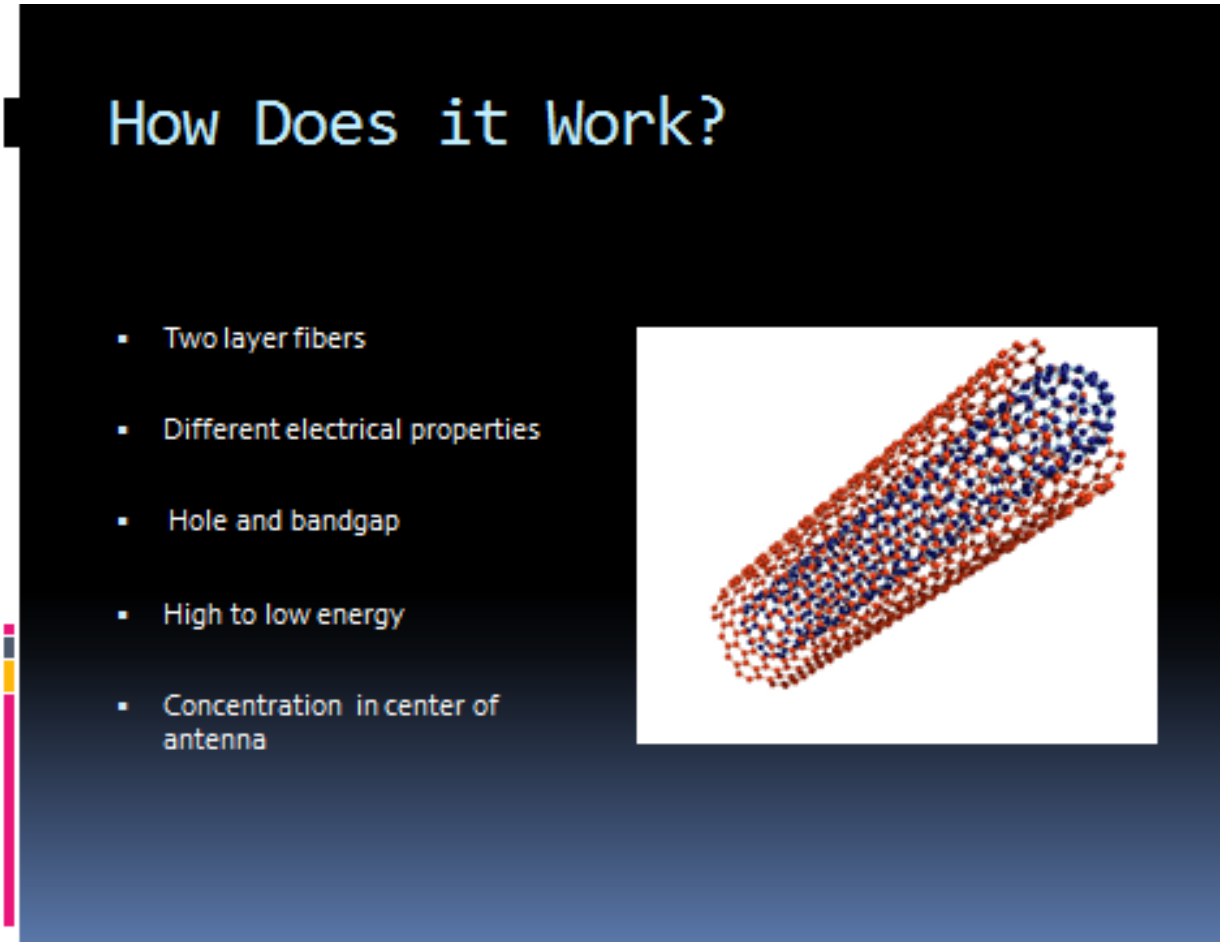
Slide 3



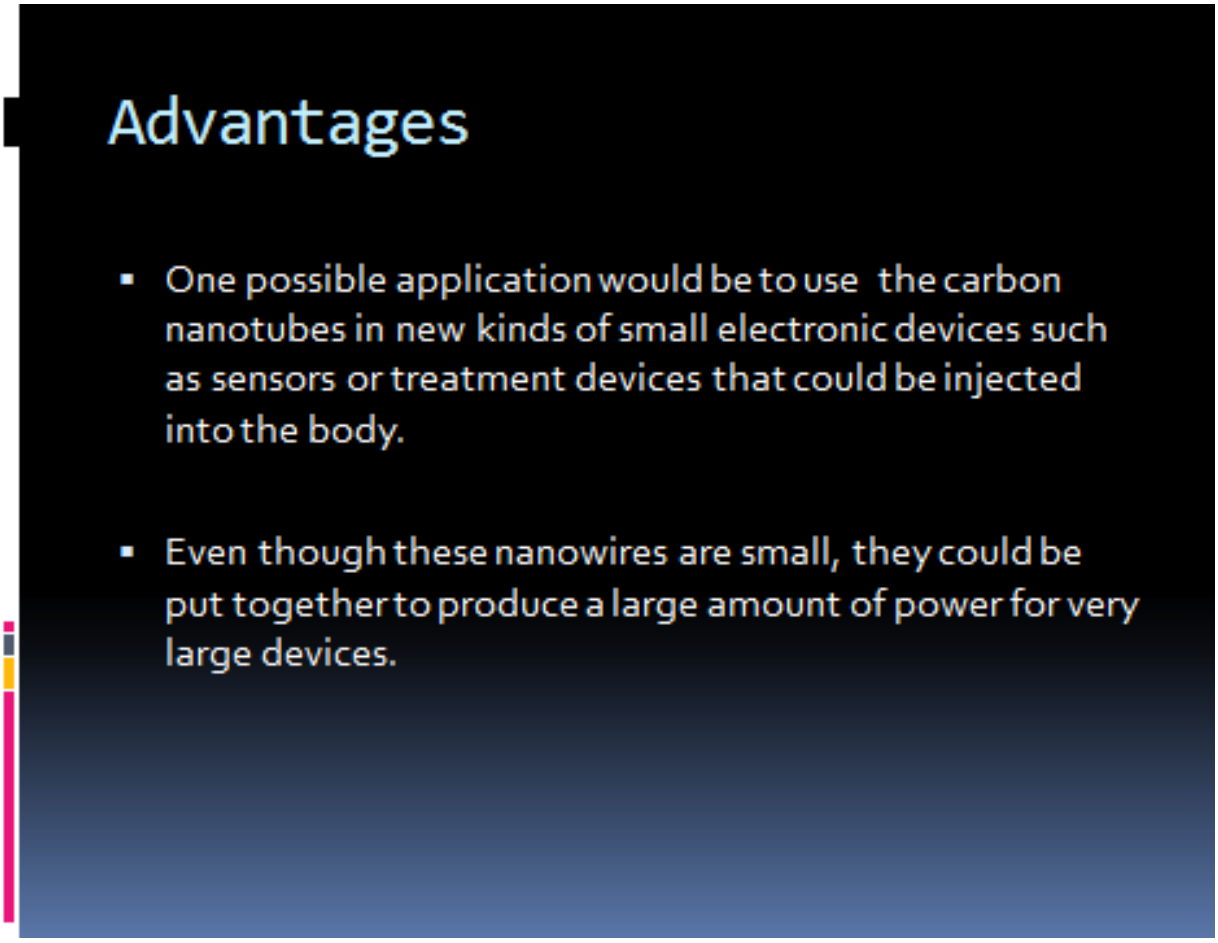
Slide 4



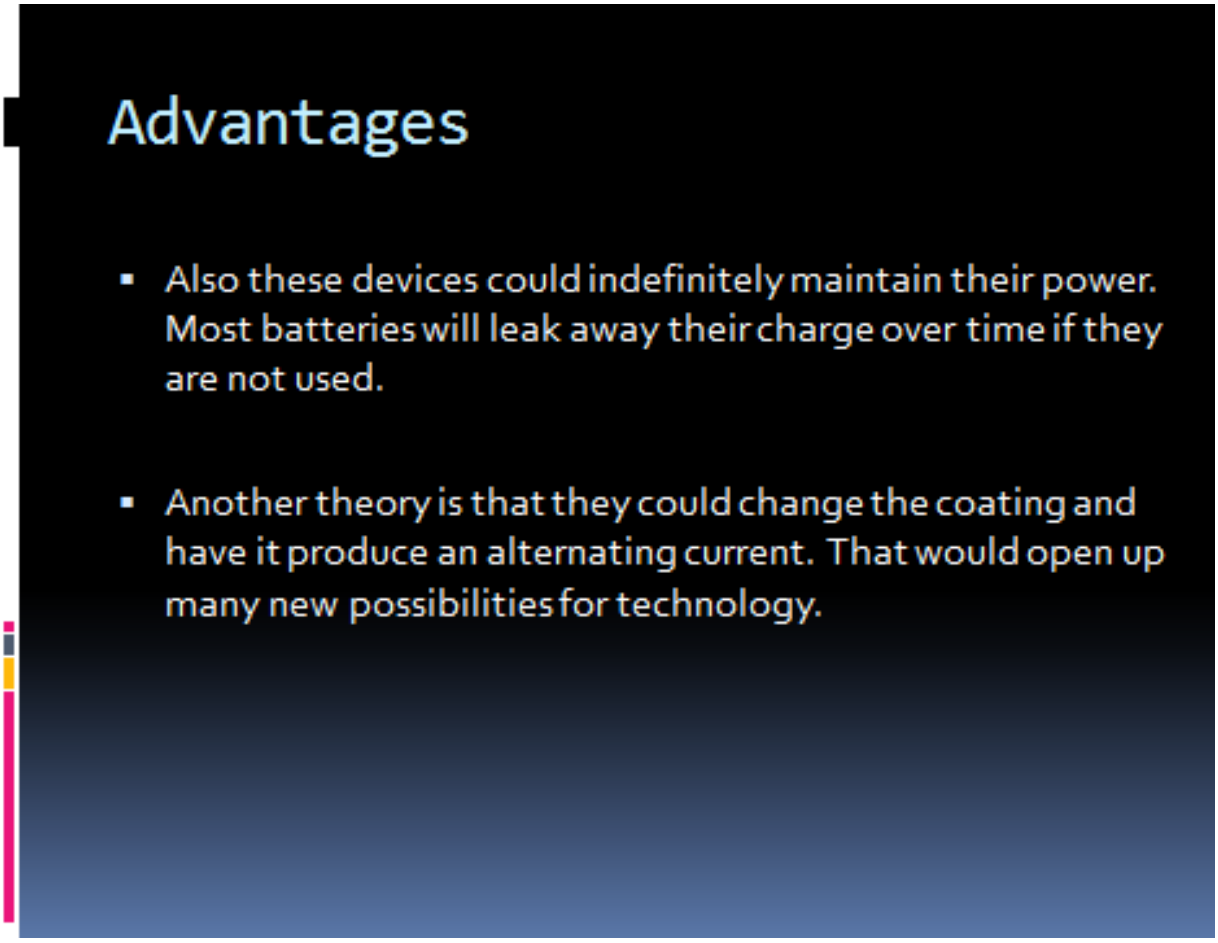
Slide 5



Slide 6



Slide 7



Slide 8

Slide 9

APPENDIX B

Astable Multivibrator with Varying Duty Cycle

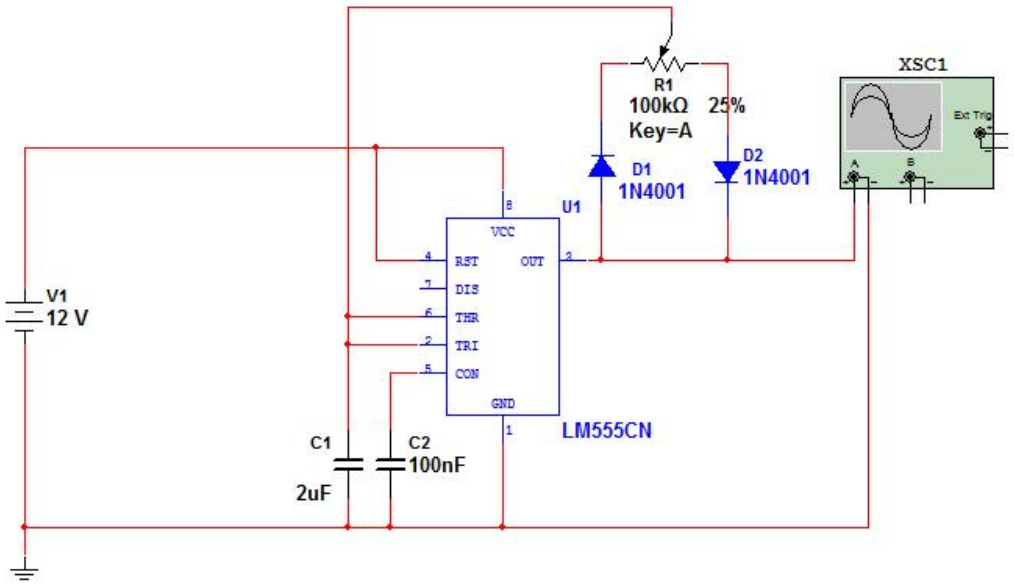


Figure B.1: Astable Multivibrator with Varying Duty Cycle

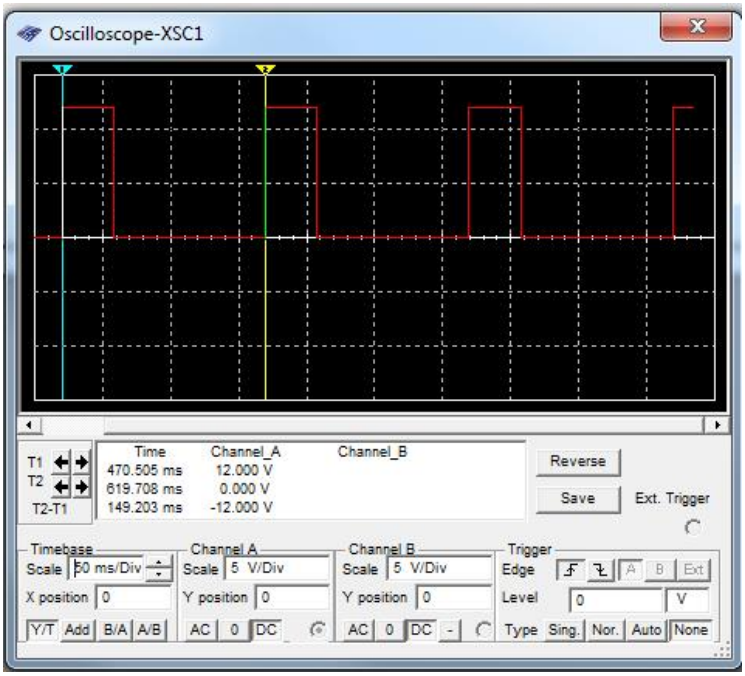


Figure B.2: Constant Level Pulse Train with 25% Duty Cycle

From the display on the oscilloscope and the usage of the scope cursors, we can see that

- T = 149.203 ms
- T_{ON} = 37.585 ms

- d = duty cycle = (T_{ON}/T)*100% = 25.19%

That value of the duty cycle matches almost exactly the ratio related to the potentiometer

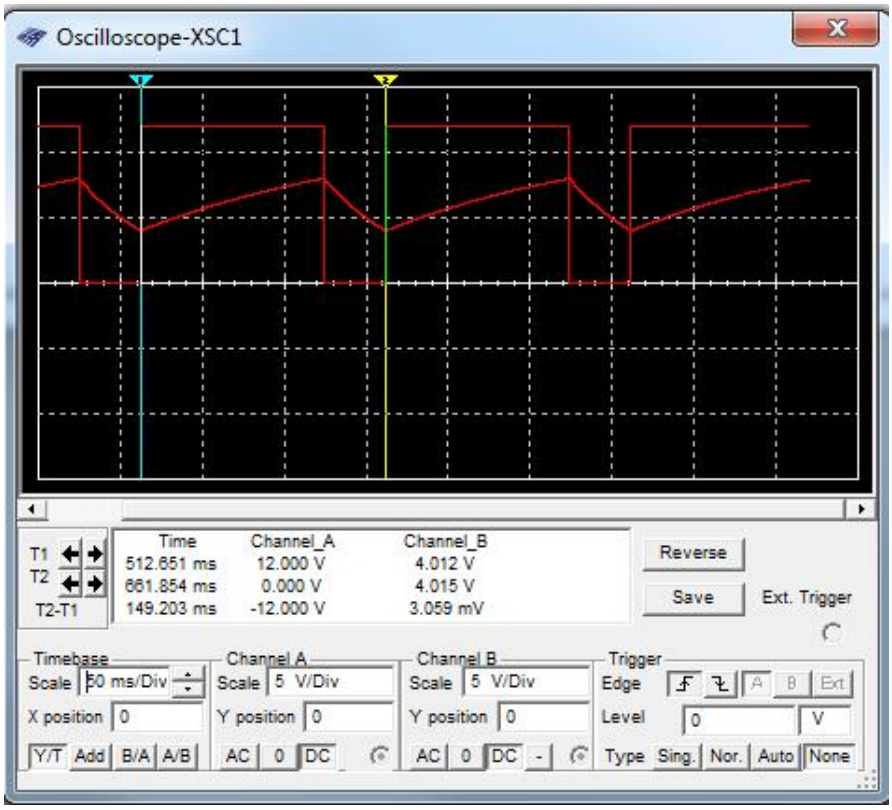


Figure B.3: Constant Level Pulse Train with 75% Duty Cycle

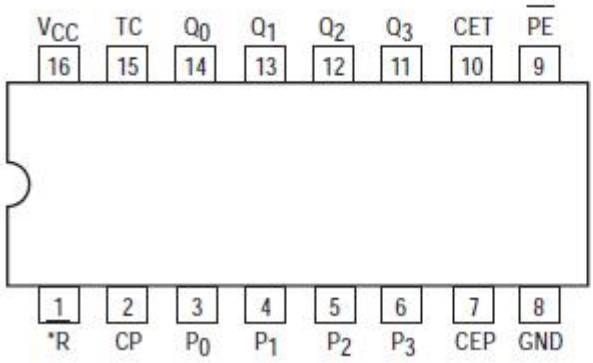
A similar result is obtained when the potentiometer is set to 75%

- T = 149.203 ms
- T_{ON} = 111.617 ms
- d = duty cycle = (T_{ON}/T)*100% = 74.81%
- Figure B.3 also shows a second signal which represents the charge and discharge of the capacitor C_1 during the process of generation of the output

APPENDIX C

C.1 Manufacturer's (Motorola) Specifications for the 74163

CONNECTION DIAGRAM DIP (TOP VIEW)



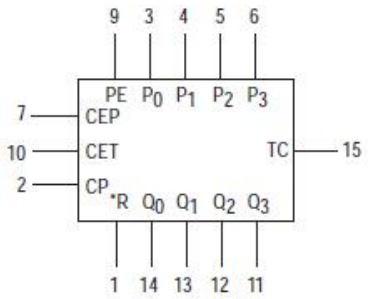
NOTE:
The Flatpak version has the same pinouts (Connection Diagram) as the Dual In-Line Package.

*MR for LS160A and LS161A
*SR for LS162A and LS163A

PIN NAMES

$\overline{PE}$	Parallel Enable (Active LOW) Input
P_0-P_3	Parallel Inputs
CEP	Count Enable Parallel Input
CET	Count Enable Trickle Input
$\underline{CP}$	Clock (Active HIGH Going Edge) Input
$\underline{MR}$	Master Reset (Active LOW) Input
SR	Synchronous Reset (Active LOW) Input
Q_0-Q_3	Parallel Outputs (Note b)
TC	Terminal Count Output (Note b)

LOGIC SYMBOL



*MR for LS160A and LS161A
*SR for LS162A and LS163A

MODE SELECT TABLE

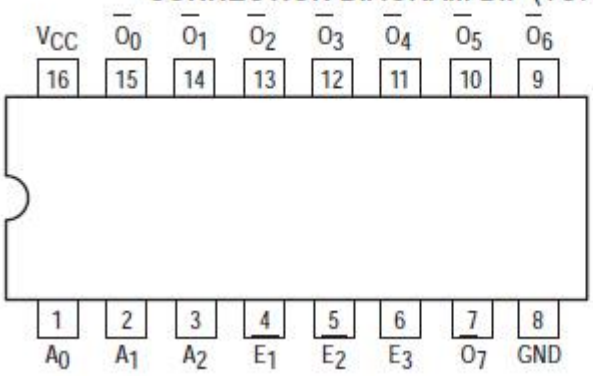
*SR	PE	CET	CEP	Action on the Rising Clock Edge (⌐)
L	X	X	X	RESET (Clear)
H	L	X	X	LOAD ($P_n \rightarrow Q_n$)
H	H	H	H	COUNT (Increment)
H	H	L	X	NO CHANGE (Hold)
H	H	X	L	NO CHANGE (Hold)

*For the LS162A and LS163A only.

H = HIGH Voltage Level
L = LOW Voltage Level
X = Don't Care

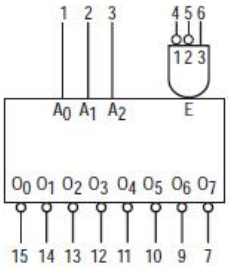
C.2 Manufacturer's (Motorola) Specifications for the 74138

CONNECTION DIAGRAM DIP (TOP VIEW)



NOTE:
The Flatpak version has the same pinouts (Connection Diagram) as the Dual In-Line Package.

LOGIC SYMBOL



V_{CC} = PIN 16
GND = PIN 8

$\overline{A_0}$–$\overline{A_2}$	Address Inputs
E_1, E_2	Enable (Active LOW) Inputs
E_3	Enable (Active HIGH) Input
O_0–O_7	Active LOW Outputs (Note b)

APPENDIX D

THE LABORATORY REPORT

The report you turn in after completion of experiments is the main product of the team's work which will be used for grading. Therefore you should devote enough attention to this final but critical step in the laboratory experience. Writing good technical reports is a valuable skill, which in the future will help advance you professional career. Start practicing it now.

The purpose of the laboratory report is to provide information on the measurement procedure, obtained results, analysis, and interpretation and discussion of these results. The discussion and conclusions are very important in a report because they show what knowledge you gained by doing the experiments.

There is no one best format for all technical reports but there are a few simple rules concerning technical presentations which should be followed. Adapted to this laboratory they may be summarized in the following recommended report format:

1. Cover page
2. Introduction
3. Experimental Procedure
4. Experimental Data and Calculations
5. Discussion
6. Conclusions

Detailed descriptions of these items are given below.

1. Cover page should have the names of the team members, and a designation such as Team 3, and course and section number (e.g. FED 101-015). It should also contain the number and the title of the experiments, such as "Experiment 7 – Transistors and Diodes". Cover page should also have the dates of the experiment and of the report delivery (not the due date).
Copy the cove page at the end of this section and fill the required information.
The cover page should be signed by all team members, confirming that they are familiar and agree with the report contents.

2. Introduction should contain a brief statement in which you state the objectives, or goals of the experiments. It should also help guide the reader through the report by stating, for example, that experiments were done with three different circuits or consisted of two parts etc. or that additional calculations or data sheets can be found in the appendix.

3. **Procedure** describes the experimental setup and how the measurements were made. Include here circuit schematics with the values of components. List actual (measured) values of the resistors used, not only their nominal values given by the color code. Mention instruments used, their settings and describe any special measurement procedure that was used. Remember: giving more information is not a mistake, less may be.

4. **Experimental** Data and Calculations section should be presented clearly with a reference to the procedure and the circuit's schematics. Tables are often a good way of presenting results. This section can also include some calculations or data analysis. For example, in describing the measurements of currents or voltages in a circuit make a table with symbols in the first column (such as V_1, V_2, I_1), measured values in the second and the values calculated from theory in the third. The fourth column might contain the calculated percent difference between the two. This may help to determine if the measurements agree with calculations.

5. **The Discussion** is a critical part of the report which testify to the student's understanding of the experiments and its purpose. In this part of the report you should compare the expected outcome of the experiment, such as measured voltages or currents with those calculated from theory or computer simulation. The table described above will be useful. In comparison of theory and experiment you may not get and usually do not get a perfect agreement. Sometimes the agreement is poor. It does not necessarily mean that your experiment was a failure. Comment if the difference between the experiment and theory is reasonable (1 – 3%) or may be something went wrong with the measurements (for example the difference is 20%). In the latter case do not hide this fact but try to explain what might have happen. You may also say what could have been done differently, how experiments may be improved, or make other comments on the laboratory. Constructive and original statements are highly valued.

6. **The Conclusions**, should contain several short statement closing a report. Was the experiment generally successful? What have you learned from it?

FED 101 – ___
Fundamentals of Engineering Design
For Electrical and Computer Engineers

Laboratory Report – Experiment No ….

Experiment Name: **…….**

…….

<u>On my honor, I pledge that I have not violated the provisions of the NJIT Student Honor Code</u>

STUDENT TEAM N**o** ____

Name	Signature
……..	_____
……..	_____
……..	_____
……..	_____

Experiment performed on date _____

Report submitted on date _____

Returned for corrections on date _____ Grade _____

Returned after corrections on date _____ Grade _____

FINAL

APPENDIX E

AN INTRODUCTORY GUIDE TO ARDUINO

I. Introduction

Arduino is an open-source hardware company that is famous for its easy-to-use, versatile, and relatively inexpensive microcontrollers. Microcontrollers are being used more and more in projects, and, as an aspiring engineer, it is necessary to expose yourself to such technology.

II. Download and Set-Up

Step 1: Visit the official Arduino website at https://www.arduino.cc/
Step 2: Click on the "software" tab located at the top of the home page, and download the Arduino IDE. Figure 1 displays the options for downloading. Please select your computer's operating system (as seen in the list on the right) when downloading the Arduino IDE.
*The following page will ask for a donation to the Arduino software. Ignore this, and press "just download" instead.

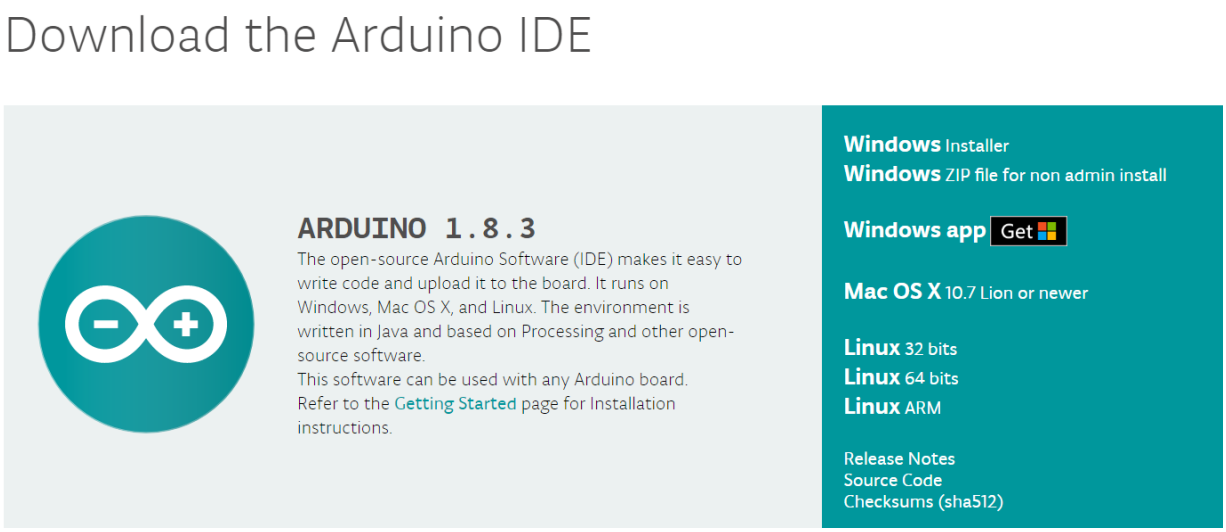


Figure 1. Arduino IDE download options.

Step 3: Open the file that was downloaded. Allow the application to make changes on your device. Figure 2 displays an updated step. Press "I Agree" to the license agreement.

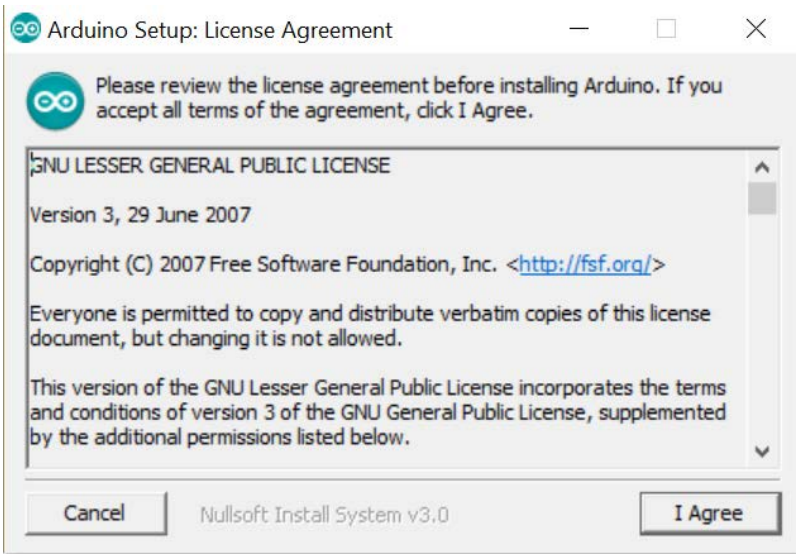


Figure 2. License agreement window.

Step 4: Keep all components selected to install (as seen in Figure 3), press "Next," and then install the Arduino software in an appropriate folder (e.g. Program Files). You will need about 400 MB of space to install the Arduino IDE.

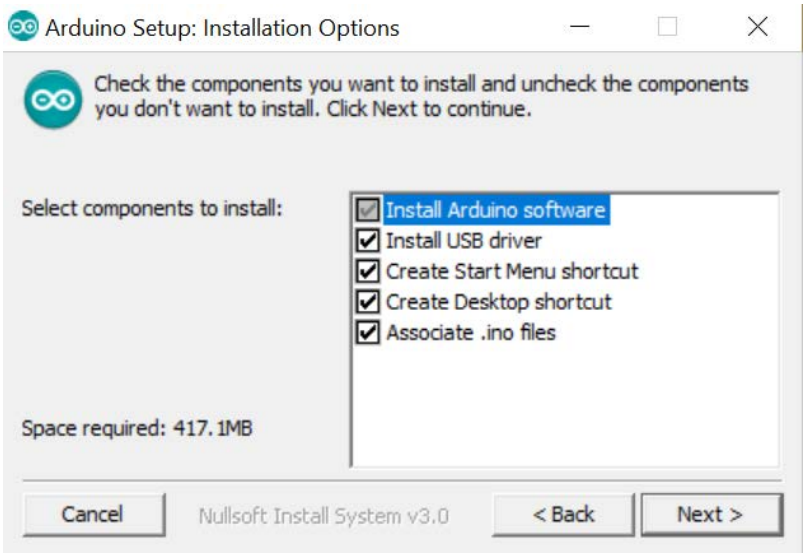


Figure 3. Installing Arduino components.

Two windows will appear during installation, asking you to install certain devices. Click "**Install**" for both. Figure 4 is an example of what one of the windows may look like.

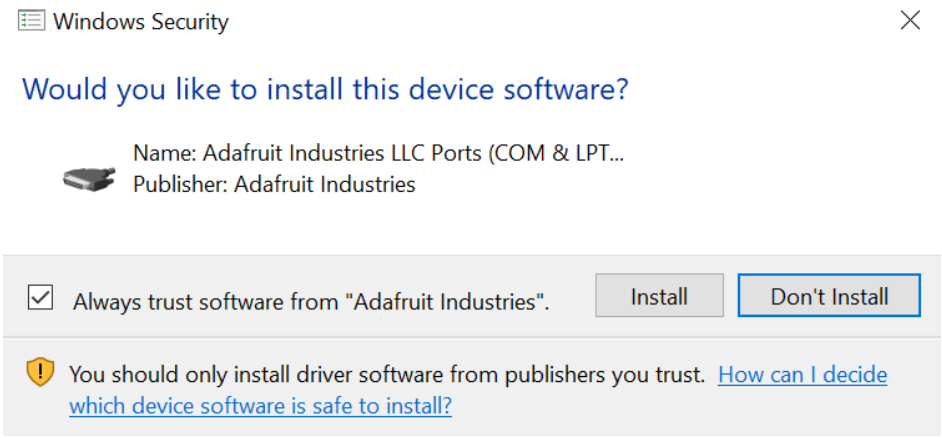


Figure 4. Installation window example, asking to install "Adafruit Industries."

Step 5: Once installation is complete, press "Close." The software has now been added to your computer. Look for the "Arduino IDE" icon on your desktop, and open it.

III. Running the Program and Syntax

1. Note that every program you write should be stored in a folder carrying the same name as the program file.
2. When you start Arduino for the first time or on a different computer, make sure that your laptop or PC recognizes the board. See figure 5 for the procedure to verify or correct the configuration of the appropriate serial port.
3. Most of your programs should be stored in Arduino folder in the documents folder.

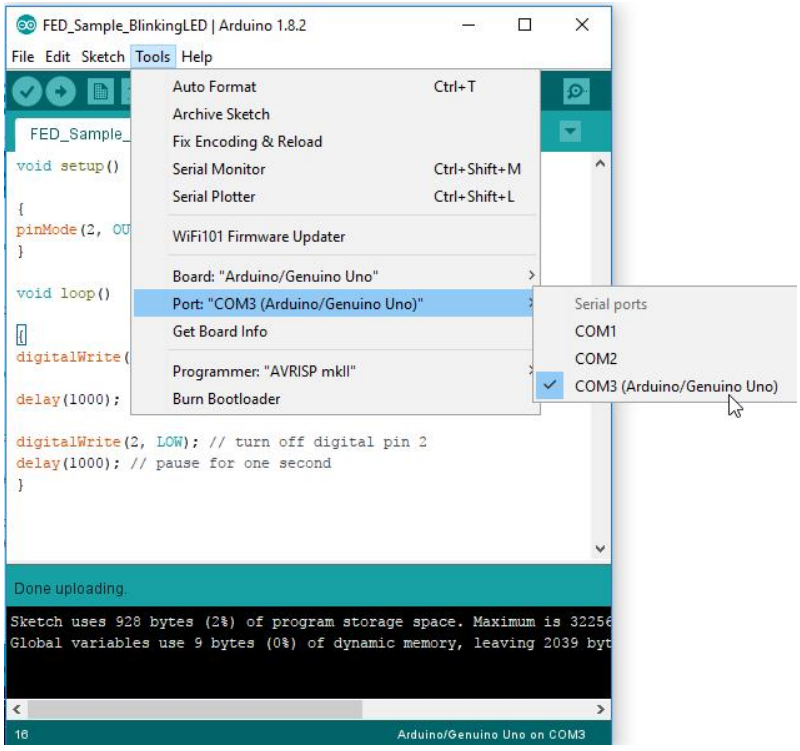


Figure 5. Using the 'Tools' menu to verify the Arduino can communicate with the PC.

4. Click on file then open any desired file such as in this case FED_Sample_BlinkingLED.ino from the folder that carries the same name (without the extension), see figure 6.

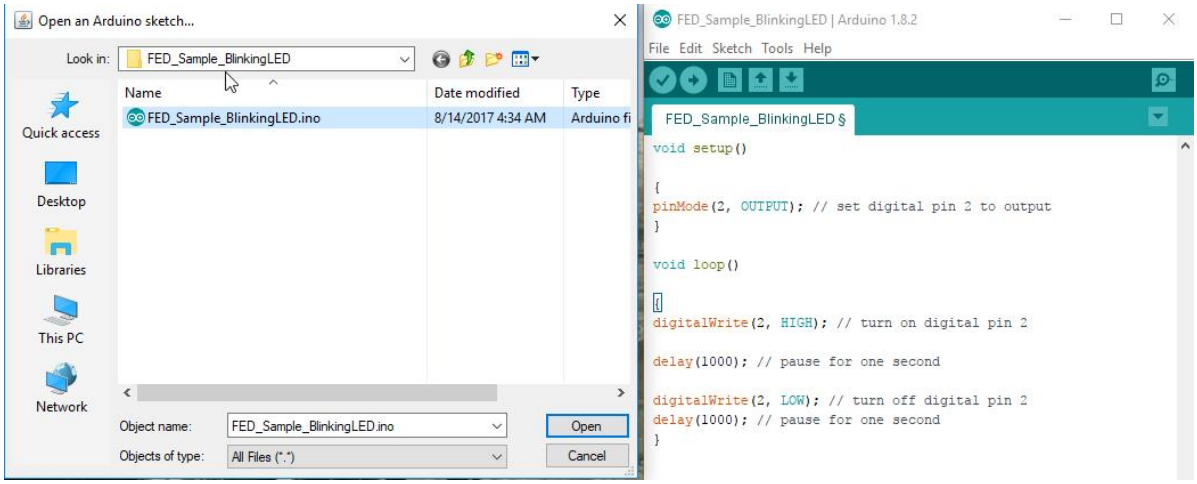


Figure 6. Opening a file from the appropriate folder.

5. The source code (figure 7) and the implementation of the circuit related to the blinking LED (figure 8) are shown here. Explanations about the code used will be provided in the next section.

Figure 7. A sample Arduino file.

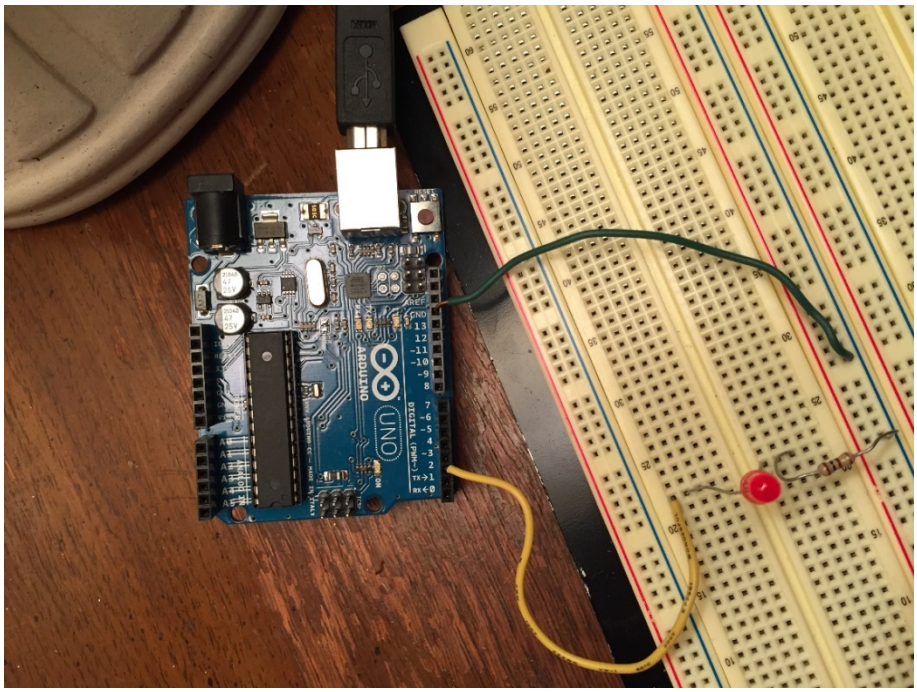


Figure 8. Implementation of the blinking LED program.

IV. Basic Code Explanations for the Blinking LED

1. int timer = 1000; allows us to declare timer as a constant to be used in the program. The advantage of the declaration is that the timer constant may appear often in the code, and if we want to change its value, we only do it once in the declaration statement. The value 1000 is equivalent to 1000 milliseconds (or one second)
2. void setup() is a directive to allow us to program some of the characteristics of the Arduino such as declaring pin 2 of the Arduino as an output pin using the pinmode command. All the setup commands (in this case there is only one) are within a pair of braces.
3. Void loop() gives us the ability to set the following command to be executed in an endless loop unless we disconnect the Arduino from power, we upload a different program, or we keep the reset button pressed (top right red button next to the printer cable connector).
4. The command digitalWrite allows us to set the required output (in this case pin 2) to a HIGH voltage (usually 5V) or a LOW voltage (usually 0V) as desired. In addition the command delay leads to the previous status to be prevalent for the duration of the amount specified (in this case one second). With the LED setup shown in figure 8, a high will turn it ON and a low will turn it OFF.

V. Boardwalk Wheel Program and Implementation

1. The code that is provided in figure 9 is a sample program to execute with the boardwalk hardware as shown in figure 10. Note that in this case we have used only 6 LEDs whereas the experiment in the manual calls for 8 LEDs. You have the ability to use any of the pins 2 to 13 (total of 8 in your case). It is clear that the program is based on turning ON one LED

for a time decided by the choice of the value of the timer, and then turning OFF the LED without delay. The process is repeated for the other LEDs. Though the program could have been written in a more elegant way (see subsequent sections), we want to show you a direct approach that could be useful when you will deal with the traffic light controller project.

Figure 9. Boardwalk Wheel Code using 6 LEDs version 1.

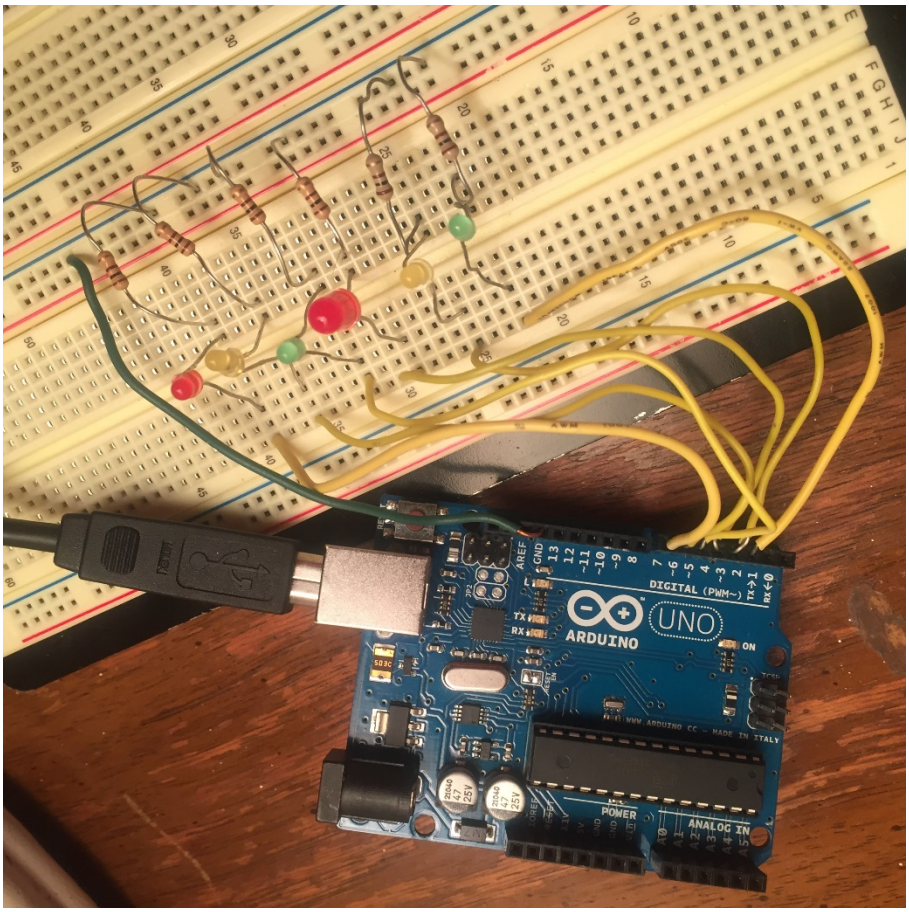


Figure 10. Implementation of the boardwalk wheel program.

2. The code that is provided in figure 11 is a sample program to execute with the same boardwalk hardware as shown in figure 10. The difference is that it is a shorter version because we use a loop since we repeat the same process of lighting the LEDs one at a time, up to as many LEDs as can be handled by the number of output pins that we have access to. In this case, the timer was set to half a second. The first '*for*' loop was used to programs pins 2 to 7 (Ardpin < 8) as outputs. Ardpin++ means increment the variable Ardpin at each execution of the loop. The second '*for*' loop sets a pin HIGH to turn on an LED for a time set by the timer, and then turn it off without delays. The process is repeated as Ardpin is incremented.

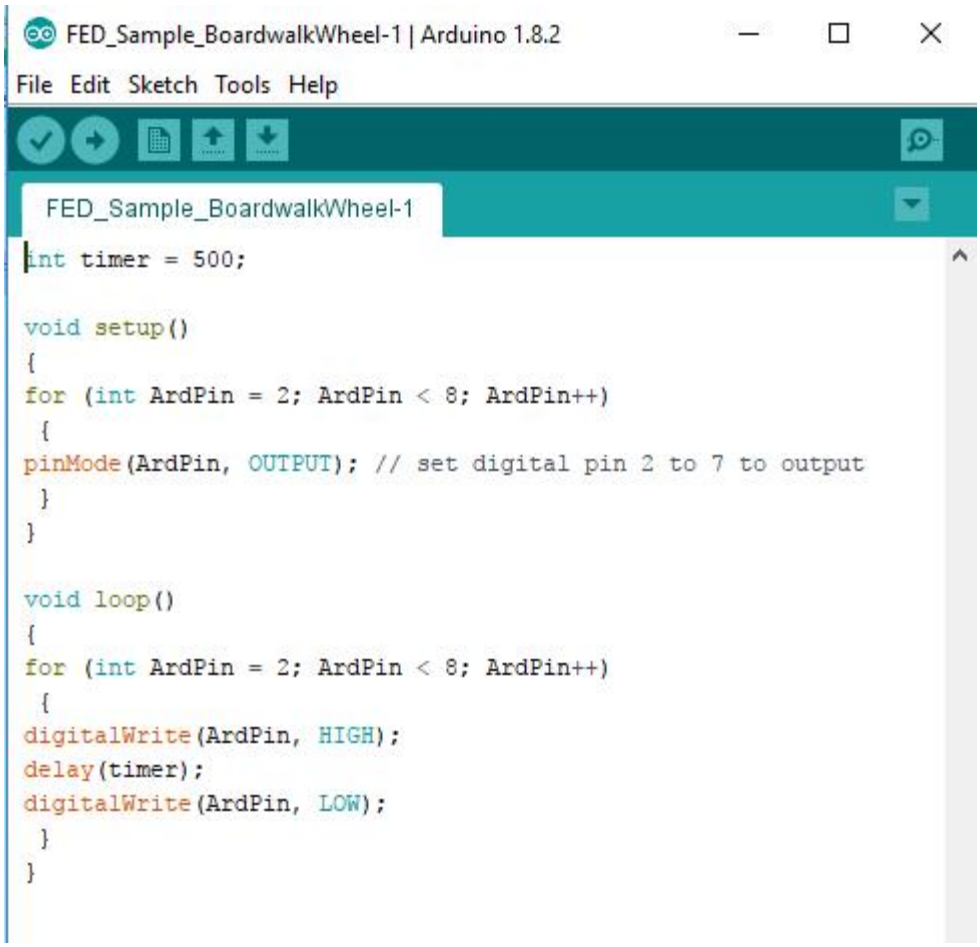


Figure 11. Boardwalk Wheel Code using 6 LEDs version 2.

3. The code shown in figure 12 allows us to turn ON LEDs two at a time, based on the following pattern: LEDs 1 and 3 are turned ON, then followed by LEDs 2 and 4, 3 and 5, and finally 4 and 6 (these numbers are related to the LEDs location, not the pin numbers). The process is repeated indefinitely until the power is turned OFF or a different program is loaded into the Arduino or the reset button (button next to the printer cable connector) is kept down. Note that if pin = 2, pin + 2 means pin 4.

You can inspire yourselves to show a more creative pattern using 8 LEDs for extra credit. You can only earn the credits if you demonstrate to your instructor the pattern you devised. He may need your code to make sure you wrote it and you understood the operations involved.

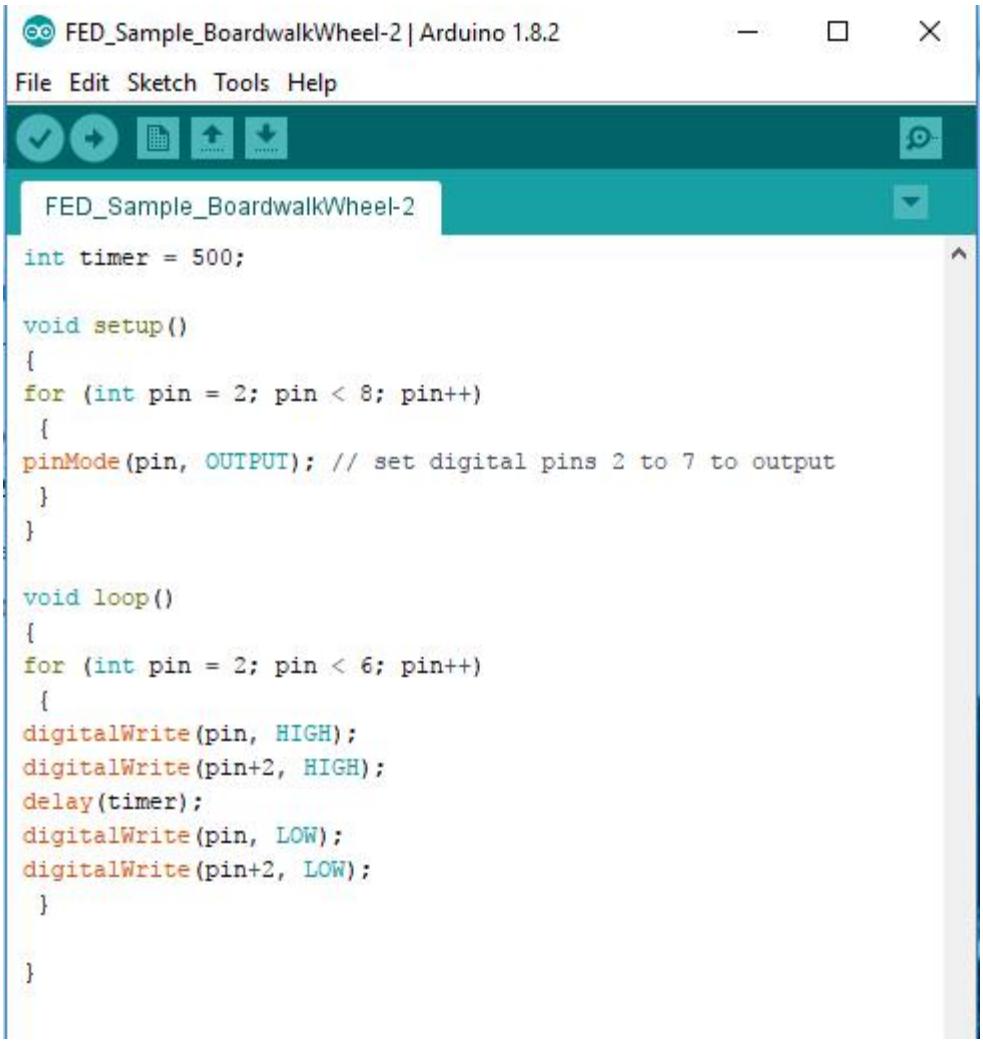


Figure 12. Boardwalk Wheel Code using 6 LEDs version 3.

