

Name: _____

COMP125

Sign-Off: _____

Lab #1: Workstation Familiarization

Experimental objectives:




1. Become familiar with the computer workstation environment in the DeVry electronics laboratory.
2. Learn various ways of logging into and out of a lab workstation.
3. Access *Netscape Navigator* to browse the World Wide Web (including the resources available at the DeVry-KC web site.)

Introduction:

Computers will be used extensively in your electronic studies at DeVry. Proper use of workstations is critical to your success in the program. In this exercise, you'll learn how to access the workstation and gain access to the World Wide Web.

Part I: Logging into and out of a workstation with your user name

If you know your username and password for the DeVry network, complete this section. Otherwise, you will need to obtain this information from the IT department, room 140.

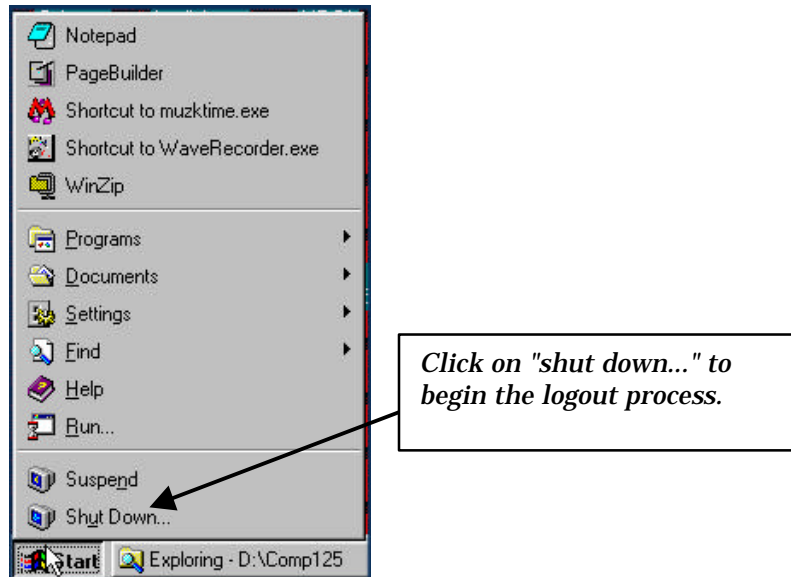
1. Press    to begin the log-in process. A dialog box will appear prompting for your username and password.
2. Type in your username and password. Click the *OK* button to begin the login process. (It may take a few seconds to several minutes for this complete, depending on network conditions.)

In case of trouble ...please check:

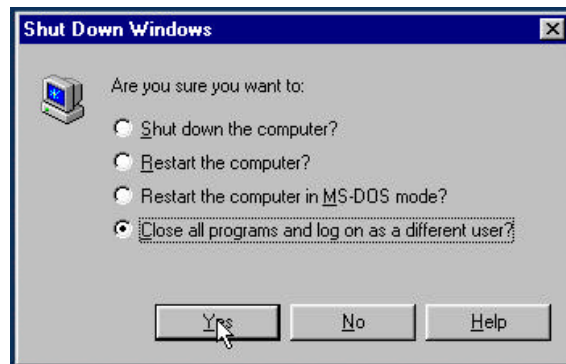
- ✓ Did you type your username and password accurately?
- ✓ Is your *context* set to "students.kc" ? (Use the *advanced* tab of the login dialog box to check.)
- ✓ You can always ask for assistance at the crib (the parts check out counter area) if no instructor is available.

3. If you have logged in successfully, you will see the *Windows NT* desktop appear. Most of the applications you will use in the lab are contained on the desktop. More are also on the *start menu*.

4. Now log out of the computer. *It is important to logoff whenever you have finished with a computer, especially if you're leaving lab.* Click the start button at the lower-left side of your screen and you'll see:



5. Once you have clicked on "Shut Down...", you'll see a new dialog box that looks like this:



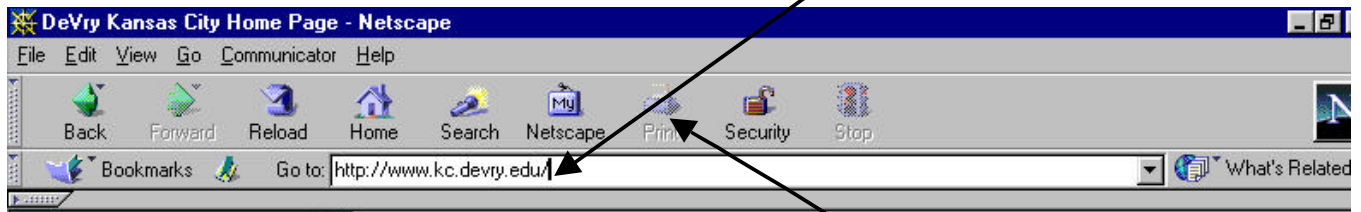
6. Choose "Close all programs and log on as a different user?" to log out, then click on YES.

Part II: Accessing the World Wide Web

1. Log into the workstation with your username and password as you did in part 1.
2. *Netscape Navigator* is installed as the web browser on the laboratory computers. To launch Netscape, *double click* on its desktop icon, which looks like this:



3. When Netscape begins, it will display a *home page*. The home page is the URL (location) that has been preset under the Netscape options. This page can vary from workstation to workstation.
4. To access the KC-DeVry home page, type "www.kc.devry.edu" into the *location bar* of Netscape, and press the RETURN key.



5. Navigate the DeVry home page until you find your professor's page.
6. Print a copy of the professor's home page by pressing the *Print* button. Your output will go to one of the lab printers near the crib.
7. Attach this output to the back of this lab, and obtain an FA (faculty assistant) or instructor sign-off.
8. Log out of the workstation when you're finished, and turn in your work.

DeVry Acceptable Use Policy

- ✓ Make sure that you have read and understand the terms of DeVry's Acceptable Use Policy. You are responsible for adhering to these rules throughout your studies at the Institute. These rules are in the *Student Handbook* and are also posted conspicuously in the electronics laboratory.

Name: _____

COMP125

Sign-Off: _____

Lab #2: Internet Search Engines and E-Mail

Experimental objectives:

1. Utilize Web search engines to locate information on the Internet.
2. Establish and utilize an e-mail account.

Introduction:

There is a vast array of information available from the Internet. There are dedicated computers (usually provided by commercial interests) called *search engines* that maintain databases of on-line content. These databases are not always complete, but they at least provide a starting point. Electronic mail, or e-mail, is an important method of online communication. Although DeVry doesn't provide students with an e-mail account, there are dozens of free e-mail services available through the web. The only requirement for the use of the free services is a connection to the Internet, which is provided by DeVry.

Part I: Using Search Engines

1. Log on to the workstation, and navigate the web browser to any one of the following URLs:

<http://www.altavista.com/>

<http://www.yahoo.com/>

<http://hotbot.lycos.com/>

2. Perform a search for an area of electronics that interests you (what made you want to attend DeVry?)

Some possible areas of interest include robotics, industrial automation, radio frequency (RF) communications, control systems, amateur radio, digital signal processing (DSP), electronic image processing, and so on.

Print out a page from at least *two* sites that cover the topic you're searching for, and attach these pages to the back of the lab.

3. What are the implications of *copyrights* and *ethics* in the on-line world? Use the search engine again to find at least one article on this topic. Print the article and attach it to the lab.

Part II: Accessing E-Mail

1. If you already have an e-mail account, skip to step 3 of this procedure.
2. Launch the web browser and point it to one of the following URLs (others are also possible):


www.hotmail.com
www.yahoo.com

On the screen you will see a prompt to sign up as a new user (or a prompt to sign up for free-email). Click on the prompt and follow the instructions given on each page.

Make sure you read and understand the conditions of use given for the online e-mail service you choose.

3. Write an e-mail to your professor (e-mail address is in the course syllabus). In your e-mail message, write the answers to the following questions. The information can be located in the tutorial presentation on the DeVry-Kansas City *Library* page. Be sure to use a complete sentence for each answer.

1. What tool can be used to find books in the DeVry-KC library?
2. If the DeVry-KC library doesn't have the book you need, what can you do?
3. What amount of time may be required to get a book through an interlibrary loan? (Why do you suppose this is so?)
4. What tool is used to find articles in journals, magazines, trade publications, and newspapers?
5. What information is required to use *ProQuest*, and where can you obtain it?
6. List the eight databases contained in *ProQuest*.
7. If you search online for an article and find that the full text isn't available, the library may have the full article in print (or other media.) What is the URL for the library page that gives the list of *periodical subscriptions*?
8. What is *SIRS Researcher* and where is it available?
9. List three databases that can be used to obtain information about companies.
10. Sometimes a reference book or encyclopedia is a good place to start when searching for information. List three examples that are available in the library.

TIP: You can open two (or more) browser windows to complete this part of the assignment much more quickly. Use -N to force Netscape to open a new browser window.

Upon completion of this lab, your professor should receive from you:

- ✓ The lab handout, with the printed pages of the search results from part I attached.
- ✓ An e-mail with the answers to the ten questions above.

Name: _____

COMP125

Sign-Off: _____

Lab #3: The MS-DOS Command Processor

Experimental objectives:

1. Enter system commands directly into the system command processor.
2. Manipulate files, folders, and other objects directly using the command processor.
3. Compare the file operations available in the command processor to those in *Explorer*.

Introduction:

Although Windows is a GUI (graphical user interface) operating system, there exists within it a text-based command line processor. This processor, *command.exe* (*command.com* in some versions) emulates most of the MS-DOS command structure from Old Testament times. Sometimes the quickest way to get a job done (especially if it involves manipulation of files) is to use the command line interface. Situations will also arise where you *must* use this interface (for example, recovering after a system crash.)

Experimental Procedure:

1. Log on to the workstation, and using the procedure on page 1 of the *MS-DOS Commands* handout, start a command processor session.
2. Start an *Explorer* window by right-clicking on the *Start* button of the taskbar and select "*Explore*."
3. Change the current directory and drive to H: by issuing the following command in the MS-DOS window:

H:

The *H* drive is the place on DeVry's network where you can store files. Each student gets about 10 MB of storage on the *H* drive. If you have trouble in this step, check with the instructor.

3. Are any files currently stored on this drive? Issue a directory command for this drive.

Unless you've stored something here already, drive H should be empty. Drive H contents are "privately owned" by each student and remain on the network after you log out. However, do not rely on drive H as your only method of storing data!

4. Navigate the *Explorer* window to the *H:* drive. Below, describe how the output in the *Explorer* window and the *MS-DOS* window compare:

5. From the command line (still on drive H:) create a text file called *mine.txt* by performing the following sequence of commands:

copy con mine.txt

[After pressing , the command prompt doesn't reappear. DOS is awaiting your input from the CONsole. Type the following lines:]

This is the first line of the text file.

This is the second line of the file.

This is the last line of the file.

Type and Z together. Control-Z is a special character that marks the end of text files. When DOS gets the ^Z (control-Z), it closes the open text file.


6. There are several ways of examining your text file. Execute a *type* command to view the contents of the file. Below, write the exact command line that you used:

7. You can also use *notepad* to examine files. Issue the appropriate command (*notepad* is the correct name of the program), and write your command line below:

8. Create a *copy* of the file "mine.txt" called "copy.txt" by using the *copy* command. Write down the command line that you used:

9. At this stage, you should now have two text files in the root directory of drive H:. Placing a lot of files in the root folder of a drive can lead to confusing situations, so we often want to create *subdirectories* or *subfolders* to help us keep track of what we're working with.

Use the `mkdir` command to create a subfolder called "Lab3". Write down the command line contents:

10. Switch momentarily to the *Explorer* window, and press the  key. This key forces *Explorer* to refresh and repaint its window. What new element has appeared in this window since the execution of the *mkdir* command of step 9?

11. Switch back to the MS-DOS window. Use the *copy* command to copy both of the files you've created into the *Lab3* folder. Write the resulting command line below:

12. Use the *Explorer* window to verify that the copy operation completed. How many copies of each file now exist, and why?

13. Use the *delete* command to remove the two original files from the root directory of drive H:. Write down the command line (or lines) below:

14. We want to create a copy of the folder (and any subfolders within this folder) called *Lab3* on a floppy diskette in drive A. The *xcopy* command can do the job. Place a blank diskette into drive A: and issue the following command line:

```
xcopy h:\lab3 a:\ /s
```

TIP: When you copy folders in Explorer, it normally copies all subfolders as well.

15. We can create another copy of the *Lab3* folder "within" the *Lab3* folder on drive A. Issue the following command, then examine floppy drive A from *Explorer*.

```
xcopy h:\lab3 a:\lab3\ /s
```

Expand the directory tree of drive A in *Explorer*, and describe below how it looks:

16. Switch to the MS-DOS window and create a new folder called *newstuff* at the root directory of drive H. Write the command line that you used here:

17. Use the *xcopy* command to place a copy of the *Lab3* folder from the root directory of drive A into the *newstuff* folder that you just created in step 16. Write the command line here:

18. Make sure that you are still logged into the root directory of the H: drive by issuing the following two commands to the MS-DOS window:

```
h:  
cd \
```

19. Issue the following command:

```
rd Lab3
```

What error message is issued, and why?

20. What DOS command will work in place of *rd* in step 19? (See page 4 of the *MS-DOS* handout).

21. With the MS-DOS and *Explorer* windows still open on the display, obtain an FA or instructor sign-off.

22. Summarize what you have learned in this exercise.

Name: _____

COMP125

Sign-Offs: Part I _____

Part II _____

Lab #4: Batch Procedures

Experimental objectives:

1. Create an automated process using the DOS batch language.
2. Utilize the flow control commands of the batch language to make decisions and handle processing exceptions.

Materials required:

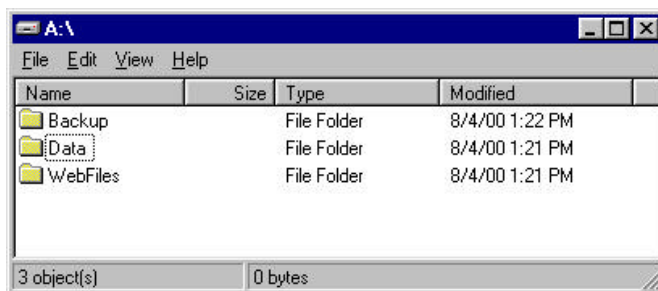
1. Blank diskette
2. Executable files *okdialog.exe*, *makehtml.exe*, *choose.exe*, and data file *phone.txt* from the instructor's web site, contained in the file "lab4.zip."

Introduction:

Batch files are text files that contain sequences of DOS commands. In computing, many tasks require that the user type in repetitive strings of commands - for example, backing up a portion of a hard drive's data, or updating a database. Batch files can be used for these tasks, and more. A special case is the file "autoexec.bat," which is executed each time an MS-DOS or Windows computer is started.

Part I: Copying Files with a Batch Procedure

1. Log on to the workstation, and using the procedure on page 1 of the *MS-DOS Commands* handout, start a command processor session.
2. Open an *Explorer* window by right-clicking on the *Start* button of the taskbar and select "*Explore*."
3. Place a floppy diskette into drive A:, and if it is not blank, format it.
3. Create the following folder structure on the diskette in drive A using the DOS command line. Verify it with *explorer*. Create exactly the same setup on your H drive.

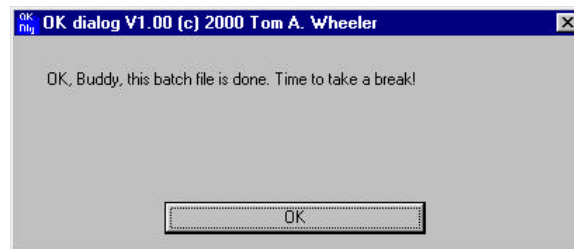


You need to create *exactly* the same structure on both drive A and drive H (if available). If drive H is not available, create a folder called "DriveH" on drive C and use this folder as if it is the root for drive H.

4. Copy the executable file *OKDialog.exe* into the root directory of drive H.
5. Copy the *phone.txt* file into the *Data* folder on drive H. Using a text editor, create a second text file within the *H:\Data* folder called "mine.txt." Place any text you wish within *mine.txt* (but not the same text as that in *phone.txt*). These text files will be manipulated by your batch procedure, and later on, *phone.txt* will get converted to a web page by your batch procedure.
6. Write a procedure called "backup.bat" that performs the following steps:
 - a) Copies all files from the *H:\Data* folder to the *A:\Data* folder, using the wildcard sequence "*" to specify the files copied.
 - b) Copies all files ending with ".txt" from the *H:\Data* folder to the *H:\Backup* folder, renaming the files with a ".bak" extension in the process.
 - c) Deletes all files in the *A:\Backup* folder.
 - d) Copies all files from the *H:\Backup* folder to the *A:\Backup* folder.
 - e) Announces completion of the task with a Windows dialog box using the program "OKDialog.exe." An example of this would be:

OKDialog OK, Buddy, this batch file is done. Time to take a break!

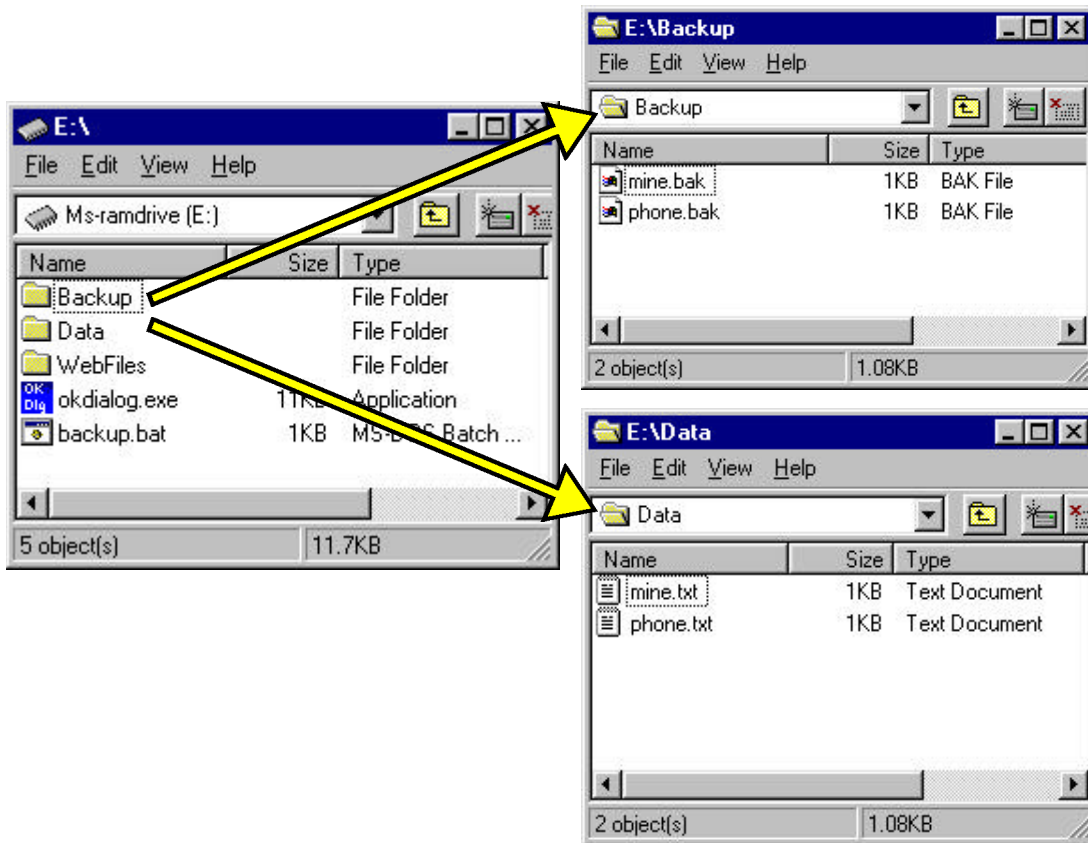
Which will result in the following dialog box display:



Put a message with your name in it within the completion dialog box. (Example: "Harvey Gliptnor's batch file has completed. Press OK to continue.")

7. To test your batch procedure, do the following:
 - a) Erase all files from the folders on drive A. (Do not remove the folders).
 - b) Erase all files from the *Backup* folder on drive H. (Do not remove the folder).
 - b) From the H:> command prompt, execute the following command to run your batch procedure:

```
backup
```
 - c) After running the batch procedure, the folders on drive A and drive H should contain the same contents as the samples on the next page.



(Folder contents after running *backup.bat*)

8. When you've got it working, get your first sign-off (on the first page of this lab.)

Print your batch file and attach it to the back of this lab.

Make sure that your batch file is properly commented, according to the convention shown on page 7 of the *MS-DOS* handout. Use a block of REM statements to comment the file. *Your name must appear in the header comment.*

To print the batch file, open it in *notepad* and select "File, Print".

Part II: Batch File Flow Control and Exception Handling

In this part, you will create a batch procedure that implements a menu of user choices using the utility *choice.exe*. The options on this menu will be as follows:

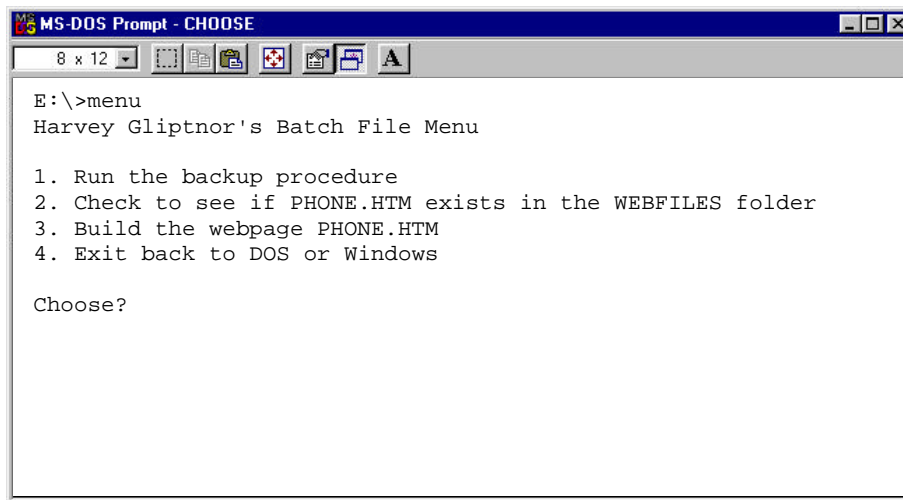
- Backup the data files using the *backup* batch procedure created in part I.
- Check to see if *phone.htm* exists within the *WebFiles* folder on drive H, and report the result.
- Build the web page *phone.htm* from *phone.txt* using the utility program *makehtml.exe*.
- Exit with no further action.

1. Copy the files *choose.exe* and *makehtml.exe* into the root directory of drive H (or its equivalent).

2. Using *notepad*, create a new batch file called "menu.bat" with the following contents:

```
@ECHO OFF
REM menu.bat
REM Menu implemented in batch file
REM
REM Author: (Your name)   Date: (Today's date)
REM Revision History: None
REM
REM
:LOOP
ECHO (Your Name)'s Batch File Menu
ECHO.
ECHO 1. Run the backup procedure
ECHO 2. Check to see if PHONE.HTM exists in the WEBFILES folder
ECHO 3. Build the webpage PHONE.HTM
ECHO 4. Exit back to DOS or Windows
ECHO.
CHOOSE
```

3. Save and run the new batch file (menu.bat). You should see an output that looks like this:



The screenshot shows a window titled "MS-DOS Prompt - CHOOSE". The command prompt shows the command "E:\>menu" and the output of the batch file. The output is a menu with four options: "1. Run the backup procedure", "2. Check to see if PHONE.HTM exists in the WEBFILES folder", "3. Build the webpage PHONE.HTM", and "4. Exit back to DOS or Windows". Below the menu, the prompt "Choose?" is displayed.

4. We now have the basic framework of the batch procedure in place. The only thing left to do is to fill out the "skeleton" with statements to complete the actions in the menu.

Did you figure out why we placed *choose.exe* as the last command in the menu? Right, this program is what produces the "Choose?" prompt. *Choose* patiently waits for a selection from the user. But wait; how can the batch procedure know what choice the user has made in *choose*? That's a very good question!

Choose returns an *ERRORLEVEL* code to DOS. An *ERRORLEVEL* code is a number between 0 and 65535 that a program returns to DOS after it runs. Code 0 usually means *success*, and anything non-zero usually represents a failure of some sort.

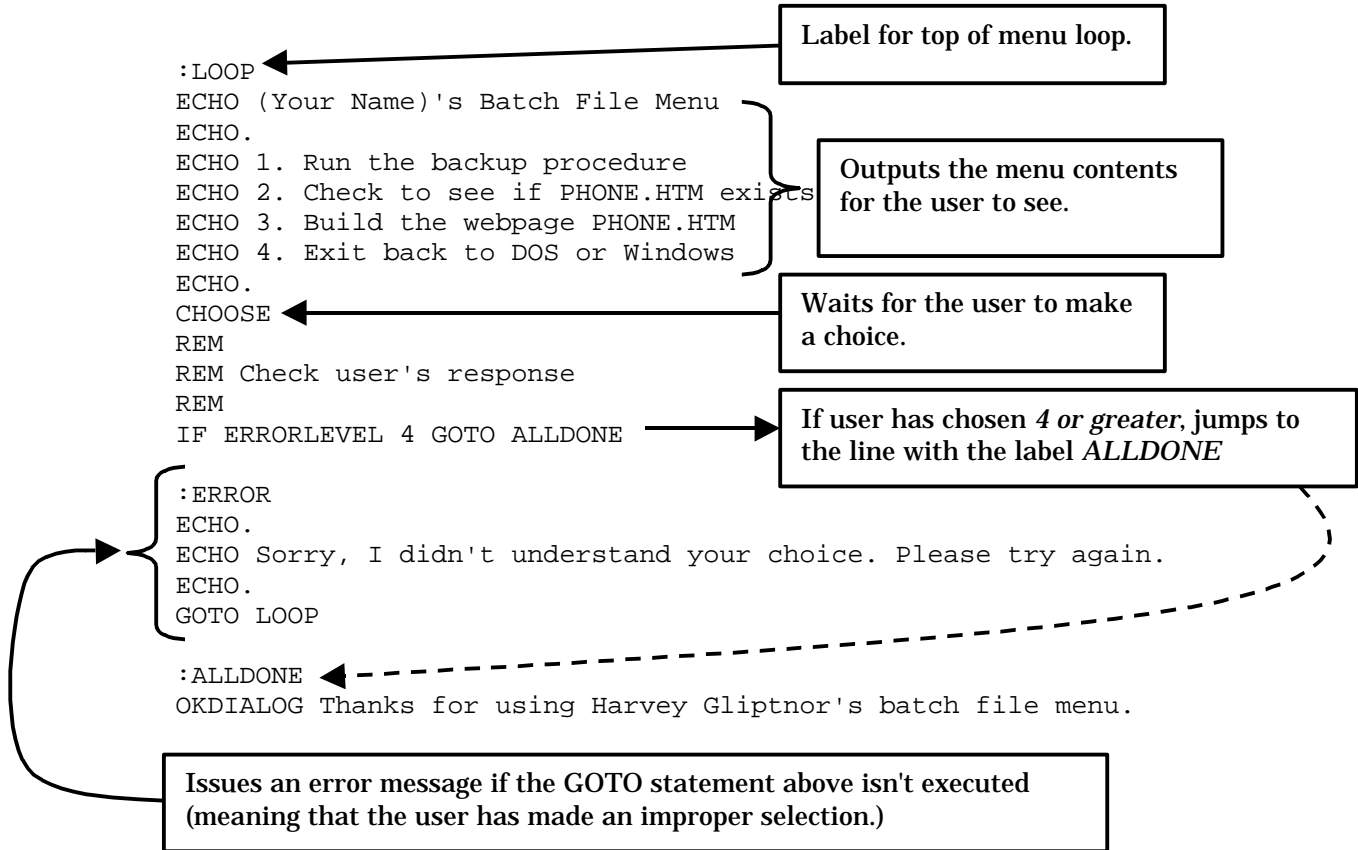
The *ERRORLEVEL* code returned by *choose* corresponds to the numbered selection made by the user. We can test this with the *IF ERRORLEVEL* sequence within our batch file. Add the following lines to the end of *menu.bat*:

```
REM
REM Check user's response
REM
IF ERRORLEVEL 4 GOTO ALLDONE

:ERROR
ECHO.
ECHO Sorry, I didn't understand your choice. Please try again.
ECHO.
GOTO LOOP

:ALLDONE
OKDIALOG Thanks for using Harvey Gliptnor's batch file menu.
```

5. Let's analyze the action of the instructions in the batch file:



6. Let's implement the third menu option. Add the following line to the batch file immediately after the *choose* command, so that it now looks like this:

```

CHOOSE
REM
REM Check user's response
REM
IF ERRORLEVEL 4 GOTO ALLDONE
IF ERRORLEVEL 3 GOTO MAKE_WEB_PAGE
    
```

Existing lines of batch file. (points to the first five lines)

Also, add the following lines immediately after the batch procedure's error handler:

```

:ERROR
ECHO.
ECHO Sorry, I didn't understand your choice. Please try again.
ECHO.
GOTO LOOP

:MAKE_WEB_PAGE
MAKEHTML \DATA\PHONE.TXT \WEBFILES\PHONE.HTM
GOTO ALLDONE
    
```

7. Menu option 3 should now work. Run the batch procedure and select option 3. After doing this, there should now be a file called "phone.htm" in the *WebFiles* folder. Open *phone.htm* with a browser (double-click on *phone.htm* from an *Explorer* window) and compare it to the contents of *phone.txt*.

MAKEHTML.EXE is a simple utility that builds a web page with e-mail addresses from an input text file database. Study *phone.txt* to see how it works.

8. You're ready to implement option #2 of the menu. Add the following instructions to the batch file below

```
CHOOSE
REM
REM Check user's response
REM
IF ERRORLEVEL 4 GOTO ALLDONE
IF ERRORLEVEL 3 GOTO MAKE WEB PAGE
```

Existing lines of batch file.

```
IF ERRORLEVEL 2 GOTO CHECK_PHONE
```

And below the code for building the web page, add the following lines:

```
:MAKE_WEB_PAGE
MAKEHTML \DATA\PHONE.TXT \WEBFILES\PHONE.HTM
GOTO LOOP
```

```
:CHECK_PHONE
IF EXIST \WEBFILES\PHONE.HTM GOTO PHONE_EXISTS
OKDIALOG The PHONE.HTM web page file does not exist.
GOTO LOOP
:PHONE_EXISTS
OKDIALOG The PHONE.HTM web page exists!
GOTO LOOP
```

9. The IF EXIST test in the logic above checks for the existence of the file *phone.htm* in the *WebFiles* folder. Devise a method to test the above section of the batch file, and describe it below.

10. The last thing to be implemented is option 1. This time, you will write the instructions to do the job. When the user selects option 1, the *backup.bat* procedure will be executed. The proper instruction to do this is:

```
call backup
```

This instruction will cause *backup.bat* to execute as a *subroutine*. Control will then be returned to *menu.bat* (the "caller" of the subroutine.)

Add the appropriate lines to the batch file and test its operation. When it's working correctly, get an FA sign off again.

11. Print the listing of *menu.bat* and attach it to the back of this experiment. You're done!

Again, make sure that your batch file is properly commented, according to the convention shown on page 7 of the *MS-DOS* handout.

Your name must appear in the header comment.

12. Summarize what you have learned in this exercise.

COMP125

Lab #5: Introduction to MS-Word

Experimental objectives:

1. Create and print a simple document using Microsoft Word.
2. Use the paragraph tagging features to create a reusable style sheet.
3. Insert stock art objects into a Word document, and use the built-in drawing tools to create text boxes, callouts, and other effects.
4. Convert an existing Word document into a web page.

Materials required:

1. Blank diskette
2. Textbook: *Exploring Microsoft Office 2000 Professional*, Grauer & Barber

Introduction:

Microsoft Word is a central tool in the *Office* suite. It is heavily used throughout the industry for creating memos, technical reports, and even simple web pages. As a student in an engineering technology program, you will be expected to produce professional-quality work. You'll find Word to be an essential tool for this purpose.

Part I: Word Familiarization

1. Read pages 2-9 of the textbook.
2. Perform exercise 1 (page 10), exercise 2 (page 19), and exercise 3 (page 33).

Print the resulting output from each exercise (with your name at the top of each document) and attach it to the back of this report. Do not attempt to perform the e-mail operation of exercise 2.

Part II: Formatting and Style Sheets

1. Read pages 51-58 of the textbook.
2. Perform exercise 1 (page 59), exercise 2 (page 74), and exercise 3 (page 89).

Print the resulting output from each exercise (with your name at the top of each document) and attach it to the back of this report.

Part III: Using Stock Art Objects in a Word Document

1. Read pages 109-114 of the textbook.
2. Perform exercise 1 (page 115), and exercise 2 (page 126).

Print the resulting output from each exercise (with your name at the top of each document) and attach it to the back of this report.

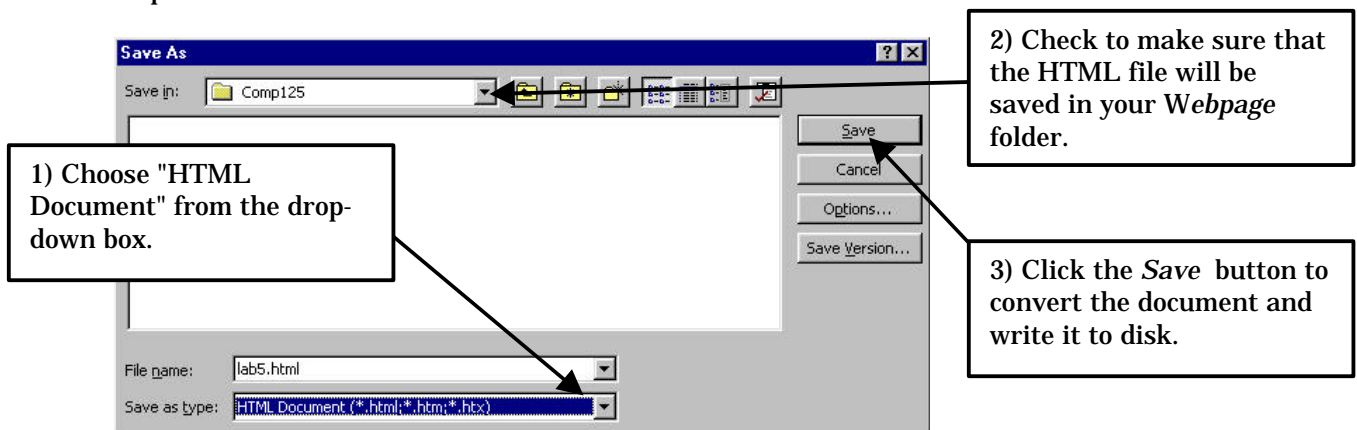
Part IV: Creating a Web Page from a Word Document

Though slow and cumbersome, *Word* has the ability to convert documents into web pages. This is quite handy when a document needs to be disseminated through both hard-copy printing and the Internet. Your course instructor may have used this ability to place his or her syllabi onto the course web site.

This capability is accessed through the "Save As..." option of the *File* menu. Be careful; once you have chosen this option, *Word* will no longer be working with the original document you started with, and changes made will *not* be reflected in the original (only the web page).

Word will also automatically deal with pictures and other objects embedded in your document. It will give these items default names and save them as separate files within the same folder where you choose to store the HTML-formatted web page. Because of this, you must not instruct *Word* to save web pages to the desktop, because the extra generated files will clutter it.

1. Create a folder called "Webpage" on your *H:* drive (or on floppy drive *A:* if the *H:* drive is not available.)
2. Open any of the *Word* documents from the previous exercises (preferably one with a graphic embedded).
3. From the *File* menu, choose "Save As...", and you'll get the following dialog. Follow the steps shown.



4. After saving the document as *HTML*, you will now be directly working with the web page in *Word*. Close *Word* after the save is complete.

Resist the urge to add lubricants to the computer system during the lengthy conversion and disk save process of *Word*.

5. Using *Explorer*, navigate to the *Webpage* folder you just created, and double-click on the HTML file. This file will have the same name as the document you created, but will have an HTM or HTML extension (which may or may not be visible in *Explorer*, depending on the view settings.)

The web page should open up. Print it using the browser, and attach the output to the back of this report.

6. List and describe the files that were present in the *Webpage* folder in step 5. (You can print out the *Explorer* window to support this discussion if you like.)

7. Summarize what you've learned in this experiment.

Name: _____

Sign Off 1: _____

Sign Off 2: _____

COMP125

Lab #6: Electronic Data Gathering

Experimental objectives:

1. Acquire a waveform with the Agilent 54622D Mixed Signal Oscilloscope.
2. Incorporate oscilloscope screen data into an Office document.

Materials required:

1. Blank diskette
2. 1 set of alligator-to-BNC leads (available for check-out from the stockroom/crib).
3. HP 54654A Training Kit (check out from the stockroom/crib).

Introduction:

Like most areas of science, electronics requires the accurate collection and interpretation of data. Oscilloscopes are one of the primary tools of a technologist; they allow the user to "see" the electrical signals in a circuit. You can expect to work a great deal with the oscilloscope in your electronic studies. Technologists must often create written documentation about circuits and systems, and oscilloscope waveforms are usually an important part of this documentation. After completing this exercise, you'll have hands-on experience with a scope, and the ability to create some very impressive reports!

Part I: Oscilloscope Familiarization

An oscilloscope is an instrument that is designed to view signals in the time domain. This means that we see the signals "as they happen," which is very useful for studying changing signals. An example of a signal that is constantly changing is the 120 Volt AC (alternating current) voltage in a household electrical system.

The display of an oscilloscope has two axes. The *horizontal axis* is calibrated in units of *time*, and the *vertical axis* is calibrated in units of *voltage*.

An oscilloscope draws a waveform in much the same manner as the popular "Etch-A-Sketch" toy. The *horizontal deflection circuit* moves the electron beam from left to right, while the *vertical deflection circuit* (amplifier) moves the beam up and down. The result is a time-domain drawing of the waveform, as shown in Figure 6-1.

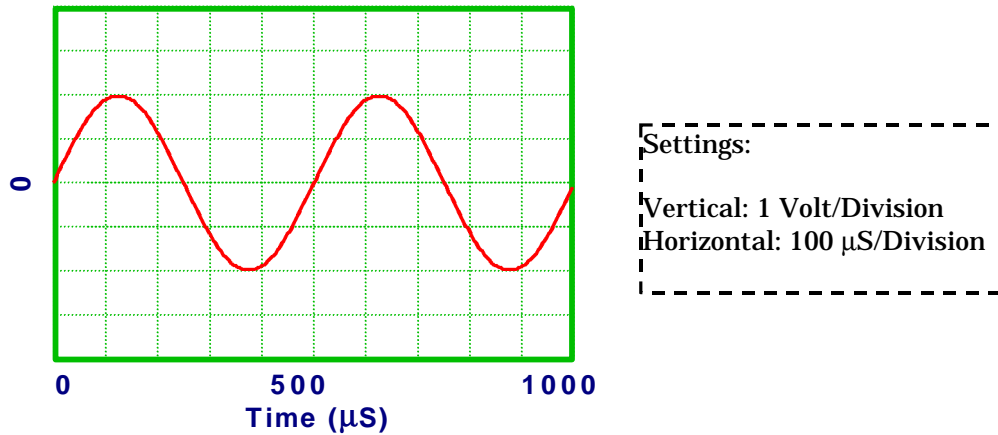


Figure 6-1: An Oscilloscope Tracing of a Sine Wave

The oscilloscope you'll be using in this experiment is a *digital sampling oscilloscope*. Instead of using a sweeping electron beam to draw its display, it uses a small computer. The analog inputs (which provide the Y-axis deflections on the graph) are internally converted to a sequence of numbers in the computer's memory by an *analog to digital converter*. The computer then produces the graph from these numbers. (In a later experiment, you will get to manipulate the numerical contents of a waveform directly.)

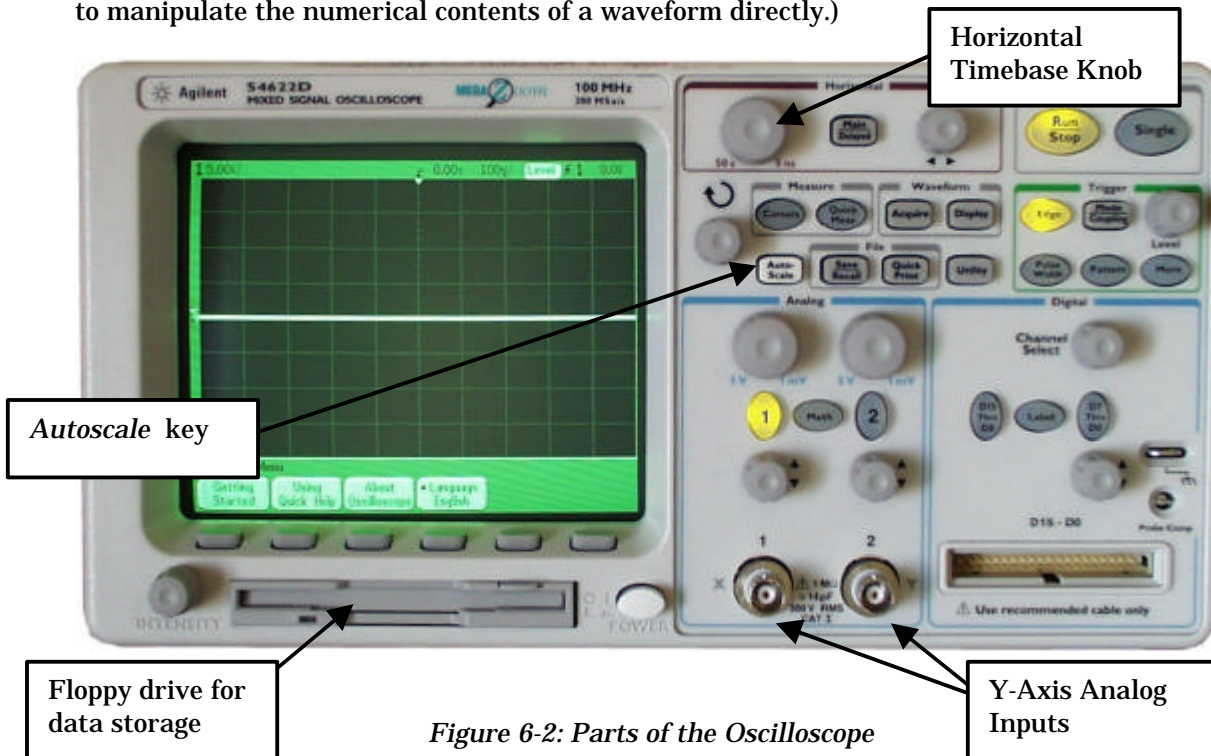


Figure 6-2: Parts of the Oscilloscope

Let's acquire a waveform from a real-world circuit. The HP 54654A Training Kit is designed to produce various waveforms for oscilloscope study. You'll notice that this kit is contained on a printed circuit (PC) board. Handle it carefully, since its components are exposed. This board is shown in Figure 6-3.

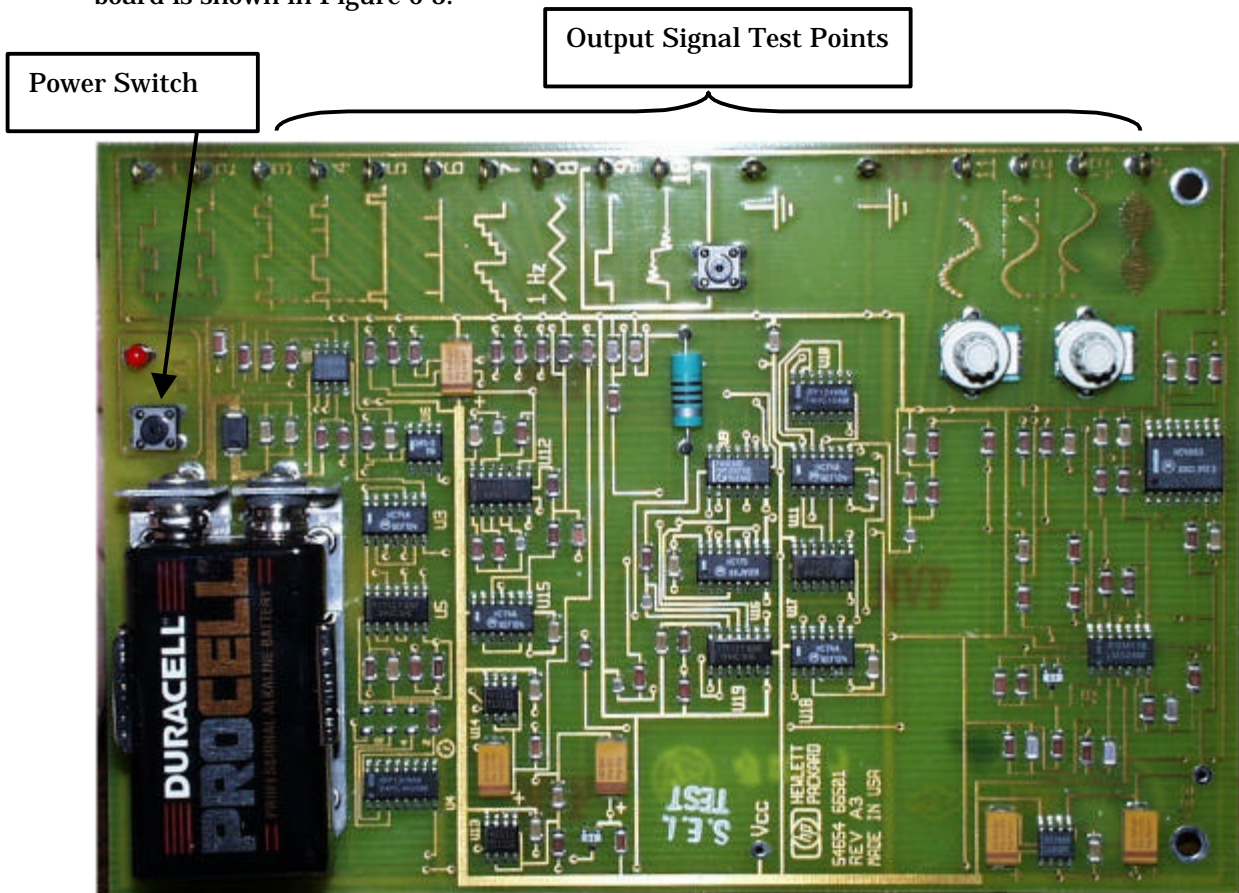


Figure 6-3: HP 54654A Training Kit

1. Reset the oscilloscope to the factory default configuration by doing the following:
 - a) Turn the scope power *off* if the instrument is turned on.
 - b) Hold down the *Autoscale* key.
 - c) Turn the instrument on (while continuing to hold the *Autoscale* key.)
 - d) Release the *Autoscale* key when the screen contents stabilize.

It is important do this whenever you use these scopes. Otherwise, you may be working with the settings used by the last person, which can be very confusing!

2. Connect the alligator-to-BNC cable to the *Analog Channel 1 Y-axis* input connector on the oscilloscope (located at the lower middle in Figure 6-2).

3. Connect the alligator clips of the cable to the *ground* and *12* pins of the trainer board, as shown in Figure 6-4.

IMPORTANT: The black clip is GROUND and the red clip is the SIGNAL (pin 12.)

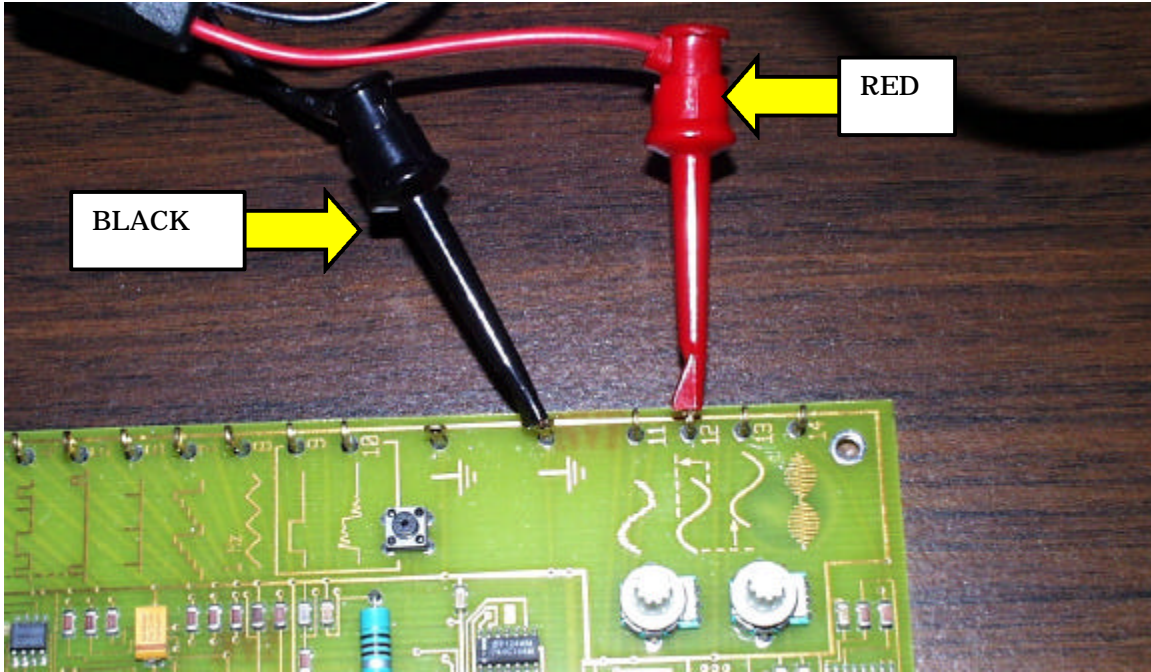


Figure 6-4: Connecting to the Trainer Board

4. Turn on the trainer board by pressing its power switch. The LED power indicator should light. If it doesn't, make sure the battery is good, and that it is securely attached at its terminals.

WARNING: Do not force the battery against the terminals. You can easily break the board.

5. Set the two control knobs on the trainer board (visible in Figure 6-4) to mid-position.
6. Press the *Autoscale* button on the scope. You should now see a display like Figure 6-5. Congratulations - you've just acquired your first waveform!

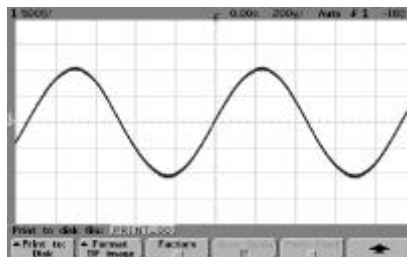


Figure 6-5: Scope Display of Test Point 12 Signal

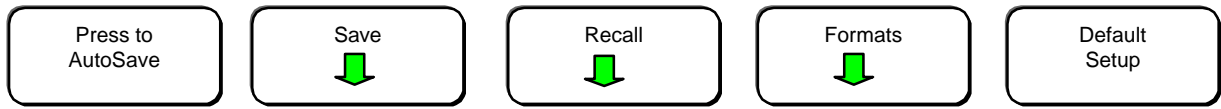
7. Have the instructor or an assistant check your work and provide the first sign-off.

Part II: Getting Waveform Data onto a Diskette and into a Document

The digital oscilloscope can save waveform data in many different formats. The two primary formats useful to us are *CSV* and *TIFF*. "CSV" stands for *comma separated values* and in short is really nothing more than a text file containing pairs of numbers. These numbers represent the (x,y) coordinate of each point on the waveform, and can be "crunched" directly with a spreadsheet program (the subject of a future experiment).

"TIFF" stands for *tagged image file format*. The TIFF format gives a picture of the screen and all of its contents just as if we used a camera to photograph it. TIFF is one of several "image file" formats (actually, TIFF is a family of somewhat conflicting formats!) Because we want to see the picture of the scope screen in our document, we will need to instruct the scope to save a TIFF image onto our diskette. From there, we can include it in any *Office* document.

1. Place a formatted diskette into the oscilloscope's diskette drive.
2. Press the [SAVE - RECALL] button on the oscilloscope. The following menu will appear at the bottom of the oscilloscope screen:

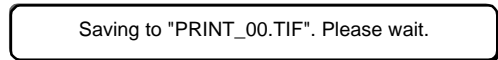


Below this menu you will notice a set of keys. These are *softkeys*. They do whatever the menu above them says, which changes as the scope is operated.

3. Press the [Formats] key.
4. From the menu that appears, choose [TIF →].

This is the standard sequence for choosing that screens be saved as TIFF images.

5. Press the [Quick Print] key (located two keys to the right of the *Autoscale* key.) The scope will display the following message box.



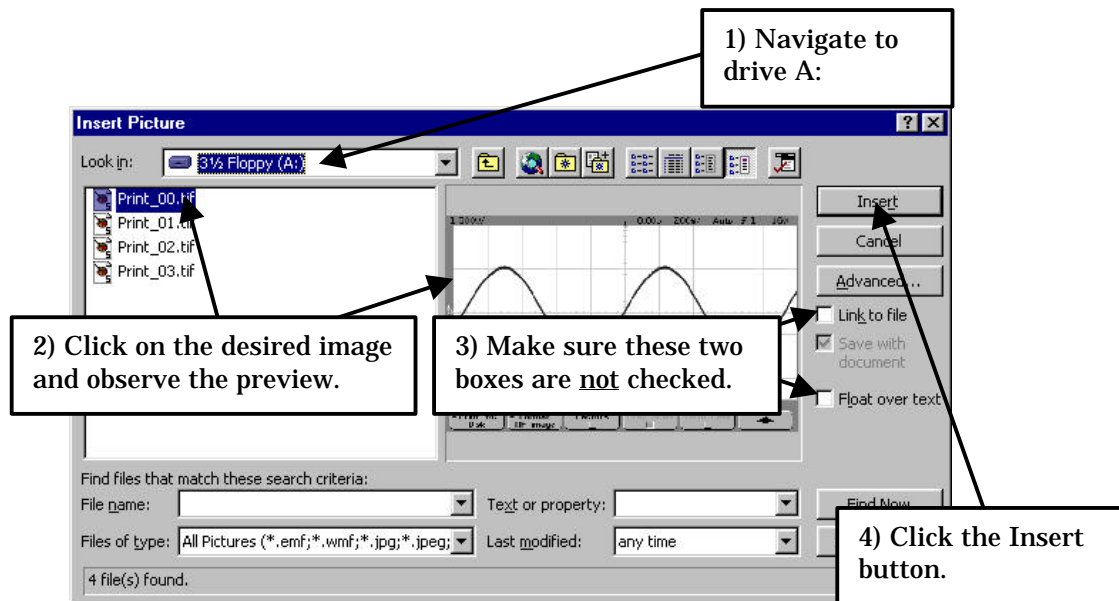
6. The image is now on the diskette, and is ready for insertion into the *Office* document.

Remove the diskette from the oscilloscope and place it into the A: drive on your computer.

7. Launch *Word* and create a new document. At the top of this document, place the following information:

The date
From: Your Name
To: Sr. Professor Wheeler
Subject: Oscilloscope Data Acquisition Exercise

8. On a new line (press the RETURN key at least once), insert the image. From the *Insert* menu of *Word*, choose *Picture* and *From File*. The following dialog will appear. Follow the steps in order as shown.



9. The picture will now appear in your document. It will be part of the text content because the "Float over text" option was disabled. Furthermore, the image will be embedded within your document file because "Link to file" wasn't checked. This last point is very important if you want to send a document by e-mail. *If the images aren't embedded (meaning that the "Link" box is checked), then they aren't saved with the document.*

10. Add a caption below the picture. Center the caption and italicize it. A typical caption might look like this:

Figure 1: Oscilloscope Reading from Test Point 12

11. Get the second FA sign-off at this point.

12. Acquire the waveforms from at least three additional test points and include them within your document. Include an appropriate caption below each waveform.

14. In your document, write several paragraphs summarizing what you've done in this experiment, and what you have learned. Attach a printout to the back of this lab.

COMP125

Lab #7: Introduction to MS-Excel

Experimental objectives:

1. Create and modify a spreadsheet using Excel.
2. Use the decision making functions of Excel.
3. Utilize the Chart Wizard to create a graphical representation of spreadsheet data.
4. Properly annotate charts and graphs.

Materials required:

1. Blank diskette
2. Textbook: *Exploring Microsoft Office 2000 Professional*, Grauer & Barber

Introduction:

Excel is a tool used to create spreadsheets. A spreadsheet is a workspace that can contain any combination of numbers of numbers and formulas. Spreadsheets are widely used in industry for carrying out the mechanical work of business, such as balancing budgets, amortizing assets in an inventory, and so on. In science and electronics, a spreadsheet is often the most convenient way to analyze and design systems, especially where iterative (repetitive) calculations are involved. One of the fastest routes between raw and graphical data is a spreadsheet.

Part I: Excel Familiarization

1. Read pages 1-9 of the green section of the textbook.
2. Perform exercise 1 (page 11), and exercise 2 (page 23).
3. Read pages 41-46 of the text.
4. Perform exercise 1 (page 47), and exercise 2 (page 59).

Print the resulting output from each exercise (with your name at the top of each document) and attach it to the back of this report.

Part II: Decision Making

1. Read pages 85-88 of the textbook.
2. Perform exercise 1 (page 89), exercise 2 (page 98), and exercise 3 (page 115).

Print the resulting output from each exercise (with your name at the top of each document) and attach it to the back of this report.

Part III: Graphs and Charts

1. Read pages 137-148 of the textbook.
2. Perform exercise 1 (page 149), exercise 2 (page 162), and exercise 3 (page 170).

Print the resulting output from each exercise (with your name at the top of each document) and attach it to the back of this report.

Name: _____

Sign Off 1: _____

Sign Off 2: _____

COMP125

Lab #8: Numerical Analysis

Experimental objectives:

1. Use the Agilent 33120A Function Generator to produce a waveform with known frequency and amplitude characteristics.
2. Acquire the data points of a waveform with the Agilent 54622D Mixed Signal Oscilloscope.
3. Incorporate measurement data into an Excel spreadsheet.
4. Generate an Excel graph from waveform measurement data.
5. Use numerical analysis techniques to find the minimum, maximum, and average values of waveform data.

Materials required:

1. Blank diskette
2. 1 BNC-to-BNC coaxial cable (available for check-out from the stockroom/crib).

Introduction:

Digital storage oscilloscopes (DSOs) can provide a wealth of information for the user. In a previous experiment, we demonstrated how the lab DSOs could provide "snapshots" of on-screen information for inclusion in Office documents. However, we can get much more detailed information from the digital storage scope.

How Analog and Digital Oscilloscopes Work

An analog oscilloscope operates by sweeping an electron beam from left to right on a cathode-ray-tube (CRT) display. This sweeping action occurs over and over. The *timebase* portion of the device controls how fast the beam moves, which controls the time displayed per horizontal division. At the same time that the electron beam is being swept left to right, it is also being deflected up and down (vertically) by the input signal being applied to one or both of the *vertical input* connectors. Because this process takes place quite rapidly (many times per second), our eyes fuse the information together into a solid image, and we see a waveform traced on the display.

The digital storage oscilloscope replaces the sweeping electron beam with an analog-to-digital converter and a computer memory circuit. When it's time to begin acquiring a waveform, the DSO simply reads the analog voltage on the vertical (y-axis) input connector, and converts it into a digital number value. This number value (the y-coordinate), and the time value at which it was read (the x-coordinate) is stored in the memory circuit. After a predetermined amount of time (controlled by the *sampling rate* control), the process repeats: Another voltage reading is taken, and the digitized voltage and time values are stored in memory. Figure 8-1 shows how this looks.

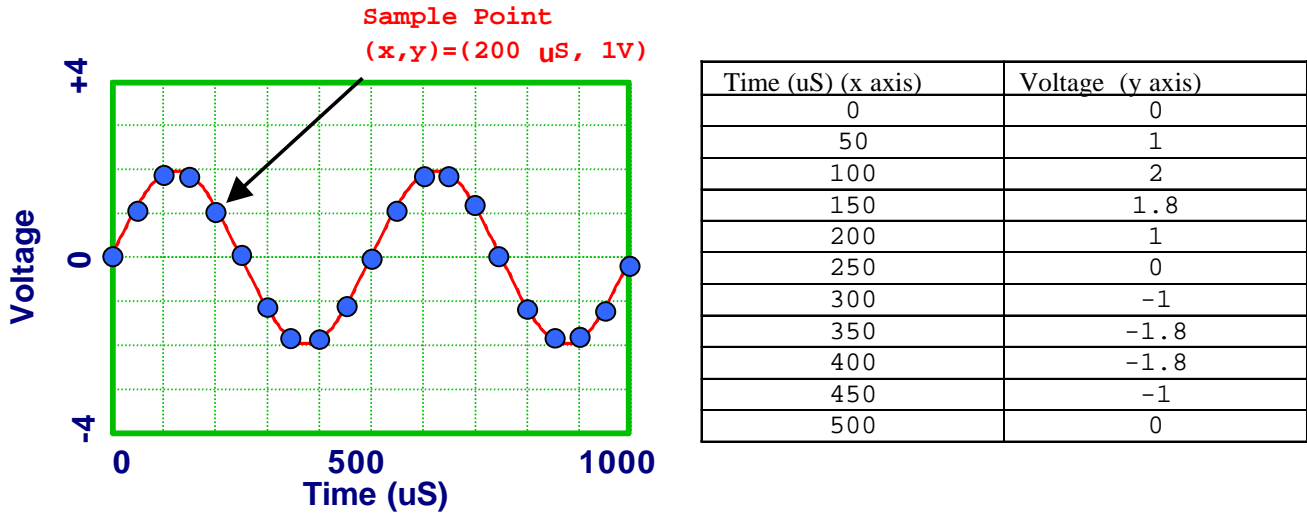


Figure 8-1: How a Digitized Waveform is Represented

The table in Figure 8-1 shows that the scope must store two values for each point on the waveform, the *time* at which the point was measured, and the *voltage* that was measured at that point. The result is an array (list) of x-y coordinates which represent an approximation of the waveform's data.

The Agilent storage oscilloscope we'll be using in this experiment has the ability to store the waveform array data as a CSV file on a diskette. A CSV (Comma Separated Values) file is merely a text file with the printed contents of the array that the oscilloscope collected. Since this file is just a text file, you can certainly open it and examine it with *notepad*, *Word*, or any other text editing tool. You'll see a table of data like the one in Figure 8-1.

Excel, the spreadsheet component of Office, can also directly read CSV files. Opening a CSV file with *Excel* results in the automatic inclusion of the text file's numeric data into the spreadsheet. This means that we can perform direct calculations with the waveform data, which takes us far beyond that capabilities of a mere "snapshot!"

Part I: Function Generator Familiarization

The *function or signal generator* is a device that generates alternating-current (AC) signals in a controlled fashion. It is a very common piece of equipment in an electronics laboratory. They come in many different styles.

We have Agilent 33120A signal generators at each operating position, as shown below in Figure 8-2:

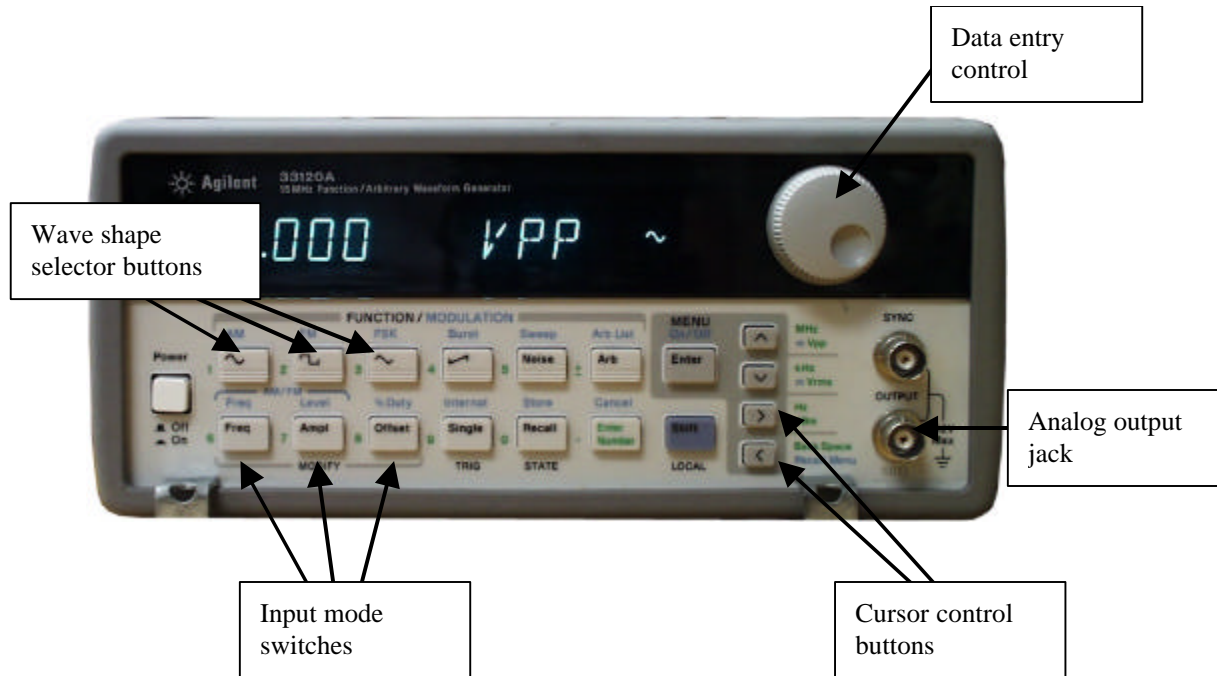


Figure 8-2: The Agilent 33120A Function Generator

There are three basic characteristics of the output signal that we can control from the generator's front panel. These are the wave shape, the amplitude, and the frequency of the signal coming from the analog output jack.

When the '33120 generator is first turned on, it defaults to a 1 kHz (frequency) sine wave (shape) output of 1 Volt peak-to-peak (amplitude). Should you get "lost" during your exploration, simply turn the unit off, wait a second, then turn it back on. These default settings will be restored.

Setting the Wave Shape

To select a different wave shape, simply press one of the *wave shape selector buttons* as shown in Figure 8-2.

Setting the Amplitude

The *amplitude* of the signal from the generator is measured in units of peak -to-peak Volts by default. In circuit analysis courses, you will learn more ways of measuring an AC (alternating-current) waveform.

To set the amplitude, first press the *Ampl* button (second button from the left in the group of *Input Mode* switches). Then rotate the *data entry* control until the display shows the desired result.

You can select which digit of the display is being modified by pressing either one of the *cursor control* buttons.

Setting the Frequency

The *frequency* is the rate at which an AC waveform repeats. A 1 Hertz (cycle-per-second, abbreviated as Hz) waveform repeats once per second. The 60 Hz energy at the wall outlet in your home has 60 sine-wave cycles per second, so it can be said to have a frequency of 60 Hertz.

The hearing range of man is from approximately 20 Hz (cycles per second) to 20,000 Hz (20 kHz). Most of the energy in speech is from 300 Hz to 3,000 Hz.

To change the frequency of the generator, press the *Freq* switch (first of the three *input mode* switches on the lower left), then use the *data entry* knob (and cursor buttons, as needed) to change the frequency.

Part I Procedure:

1. While holding down the *autoscale* key of the oscilloscope, turn the oscilloscope on, and continue to hold the *autoscale* key until the display stabilizes.
2. Connect the *analog output* of the signal generator to the *channel 1* input jack of the oscilloscope. Turn on the function generator and adjust it as follows:

Frequency = 4 kHz
Waveshape = Sine wave
Amplitude = 4 Vpp

Note: Make sure that the generator is set to "Hi-Z" mode. This setting is in the SYS menu (use SHIFT + MENU to enter the menu system, then rotate the data knob to select the SYS menu. Press the down-arrow key twice at the SYS menu to access this item, and press ENTER when it is set correctly.)

3. Press the *autoscale* key on the oscilloscope to get a proper display. (You may manually adjust the scope controls if you wish -- which is actually a superior method. In later work, you will be discouraged from using the *autoscale* key.)

From the oscilloscope, record the following information:

Voltage per vertical division: _____

Peak-to-Peak AC signal voltage: _____

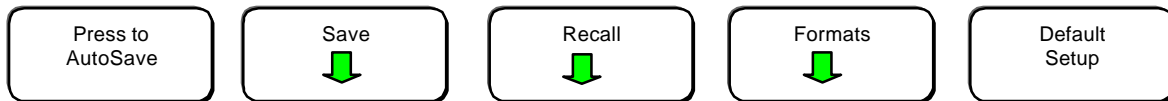
4. How does the information you collected in step 3 compare with the readings from the front panel of the signal generator? Explain using complete sentences.

5. Get your first sign-off.

Part II: Acquiring Waveform Data

In this part, you will take the waveform data from the oscilloscope and bring it into an *Excel* spreadsheet.

1. Place a formatted diskette into the oscilloscope's diskette drive.
2. Press the [SAVE - RECALL] button on the oscilloscope. The following menu will appear at the bottom of the oscilloscope screen:



Below this menu you will notice a set of keys. These are *softkeys*. They do whatever the menu above them says, which changes as the scope is operated.

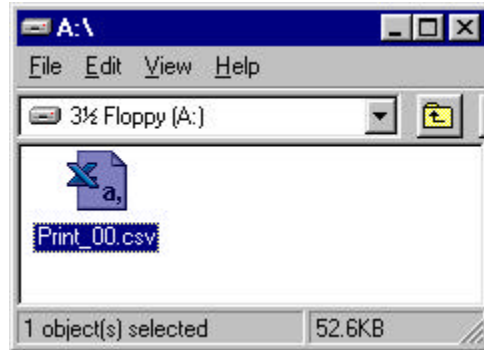
3. Press the [Formats] key.
4. From the menu that appears, choose [CSV ↵].

This is the standard sequence for choosing that waveform data be saved in CSV format.

5. Press the [Quick Print] key (located two keys to the right of the *Autoscale* key.) The scope will display the following message box.

Saving to "PRINT_00.CSV". Please wait.

- The waveform data is now on the diskette, and is ready for insertion into the spreadsheet. Remove the diskette from the oscilloscope and place it into the A: drive on your computer.
- Open an *Explorer* window and point it to drive A. You should see something like this:



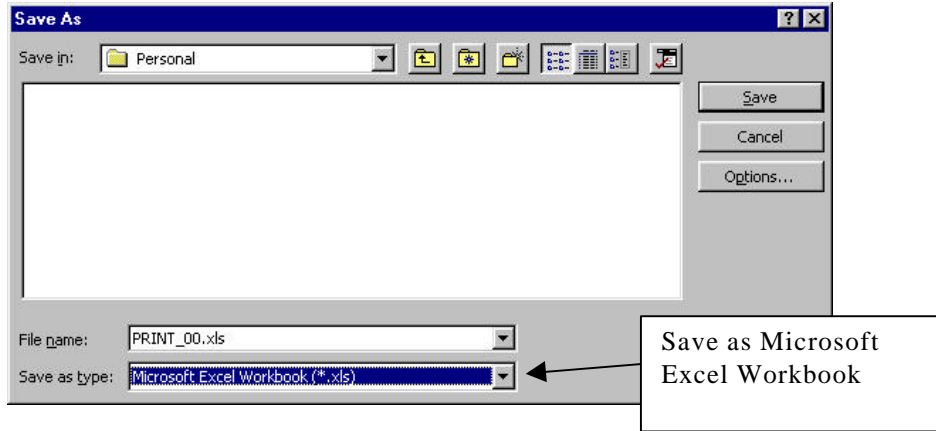
There are at least three ways that you can get this file into an Excel spreadsheet:

- ✂ Double-click the icon as shown (if you don't see an "XL" in the icon, this method will not work, because the ".CSV" entry in the system registry is either missing or corrupt.)
- ✂ Drag the icon on top of the *Excel* application icon (if one is present on your desktop).
- ✂ Manually open it from *Excel* by using the "File...Open" menu sequence.

- Open the CSV file using one of the three methods above. You should see the following:

	1	2	3	4
1	x-axis		1	
2	second	Volt		
3	-1.00E-03	-4.37E-01		
4	-9.99E-04	-4.37E-01		
5	-9.98E-04	-4.37E-01		
6	-9.97E-04	-4.37E-01		
7	-9.96E-04	-4.06E-01		
8	-9.95E-04	-4.37E-01		
9	-9.94E-04	-4.06E-01		
10	-9.93E-04	-4.06E-01		
11	-9.92E-04	-3.75E-01		
12	-9.91E-04	-3.90E-01		
13	-9.90E-04	-3.75E-01		
14	-9.89E-04	-3.44E-01		

- Before you can do any real work with this data, it must be saved as an *Excel* workbook. Use the *File* and *Save As...* menu sequence to save the spreadsheet as a *Microsoft Excel Workbook*, as shown below:



10. Now that the data is in *Excel Workbook* format, we can work with it. The first thing you'll notice is that there is a *lot* of information. In fact, there are 2,000 vertical rows of data, extending from cells A3 to B2002. Take a moment to examine this information before proceeding.

What do the two numbers in each row represent?

11. Let's find the *average* value of our signal. Place the word "Average" into cell D2, and the following formula into cell D3 (right below the word "Average"):

`AVERAGE (B3 : B2002)`

What is the *average* value of all the data points of the waveform? (Write as a decimal number without scientific notation). Think carefully – why is the average a small value?

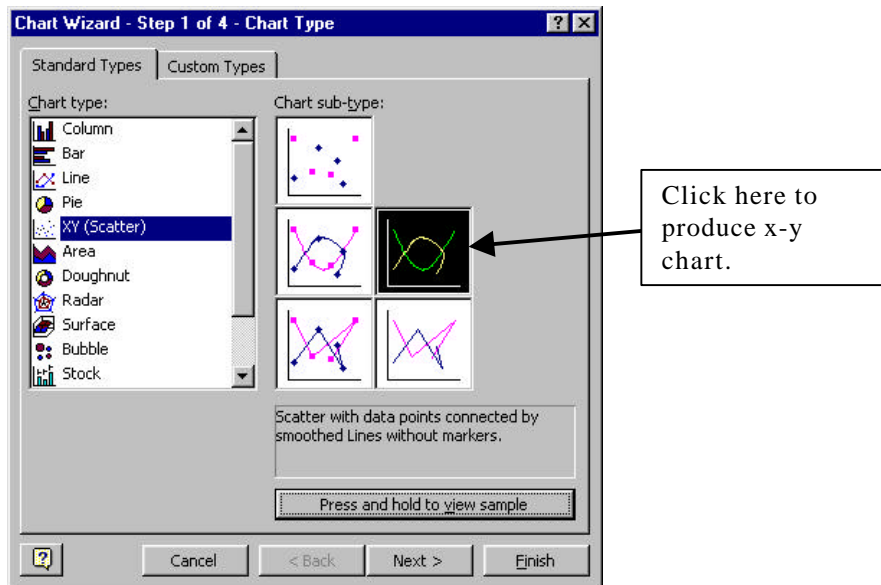
12. We can also find the *maximum* and *minimum* values quite easily. Place the word "Maximum" in cell E2, and the formula `MAX (B3 : B2002)` into cell E3. Place the word "Minimum" in cell F2, and the formula `MIN (B3 : B2002)` into cell F3.

13. Compare the reported *maximum* and *minimum* values that you obtained from step 12. Use complete sentences. Do these values appear to have any relationship with the *amplitude* setting of the signal generator?

14. We can also ask *Excel* to graph our data. Highlight the data in cells A3:B2002 (you should have *two* columns highlighted), and start the Chart Wizard (*Insert, Chart...* menu).

IMPORTANT: IT IS EASIEST TO HIGHLIGHT THIS LARGE RANGE OF CELLS BY USING THE KEYBOARD!

Choose the *scatter* type of chart. This will force *Excel* to use one of the columns for the x-axis data, and the other for the y-axis data. The dialog box should look like this:



15. Follow through the rest of the steps in the Chart Wizard. You should end up with a graph that looks exactly like the waveform on the oscilloscope screen.

Get the second sign-off when you've got the chart displayed.

16. Use the graph formatting commands to add the following to the graph:

An appropriate title, with your name in it.

Appropriate labels for the x and y axes.

Appropriate scale formatting for the x and y axes (for example, the y-axis should *not* be displayed in scientific notation – its number values are ‘normal’ enough for standard display).

Appropriate placement of the x and y axis scales. (They'll appear in the middle of the graph by default, which is not appropriate.)

Print this graph (*not the entire workbook!*) and attach it to the back of this experiment.

17. Repeat steps 1-16, but this time, set the signal generator as follows:

Waveform = Square

Frequency = 1 kHz

Amplitude = 1 Volt peak

Attach the resulting graph to the back of this experiment.

18. Summarize what you have learned in this experiment.

COMP125

Lab #9: PowerPoint

Experimental objectives:

1. Create a presentation using PowerPoint.
2. Generate items to facilitate a presentation (such as handout notes).

Materials required:

1. Blank diskette

Introduction:

PowerPoint is a very useful tool for creating presentations. It finds two primary uses. First, it is used to make and show slide shows to audiences. Second, it is used to disseminate information electronically. Because a PowerPoint slide show can contain multimedia elements, and because it can be sent by electronic mail, it is an ideal vehicle for this purpose.

Part I: Familiarization

1. Read pages 1-9 of the blue section of the textbook.
2. Perform exercise 1 (page 9), exercise 2 (page 22), and exercise 3 (page 32).

Print the resulting output from each exercise (with your name at the top of each document) and attach it to the back of this report.

Part II: A Complete Presentation

1. Create a presentation on a topic of your choice. The presentation must contain a minimum of 10 slides and must meet the following criteria:
 - The presentation should give a summary or overview of an area of science or technology that you are personally interested in.
 - There should be at least four graphic images in the presentation to illustrate the subject material. Make sure to properly acknowledge the source of any images used that are not your own.
 - The presentation must be spell and grammar checked.
 - The presentation should have handouts, printed 4 per page. Attach these to the back of this lab report.
2. This presentation will be graded on neatness, clarity, and organization. It will constitute 50 points of the 100 points possible for this experiment.

Name: _____

Sign Off: _____

COMP125

Lab #10: OrCAD Capture

Experimental objectives:

1. Generate accurate schematic drawings using OrCAD capture.
2. Incorporate OrCAD schematic diagrams within Microsoft Office documents.

Materials required:

1. Blank diskette
2. Prototype breadboard assembly (or photo likeness)

Introduction:

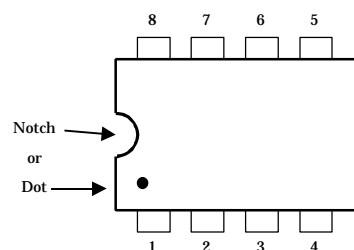
OrCAD is a design *capture* package. It is designed to "capture" (store) all the electronic details associated with a particular electronic design. In OrCAD, a *design* is considered to be the collection of schematic drawings that describe a complete electronic project. Complex electronic systems often require dozens of drawings to completely represent them. OrCAD makes it easy to manage such projects since it keeps all the drawings for a project together. OrCAD provides many support services. It can automatically annotate drawings (filling in the designators for all components - a designator gives the circuit position of a component, such as resistor R_1 or capacitor C_{37} .) It can also generate a bill of materials, which is very useful for planning product production.

Part I: Capturing a Circuit with OrCAD

1. The photograph on the last page of this lab is a prototyped electronic circuit. It has been constructed on a white breadboard. Your task is to convert this into a correct schematic diagram. Your schematic diagram should look very similar to the one on the next page when you're done. (No peeking! ☺)

Please note the following:

- You must read the resistor color codes and put the correct values on the schematic. They have been marked on the photo for your convenience.
- Be careful about the pin numbers on the integrated circuit (IC) chip U_1 . The pin numbers on an IC chip read like this:



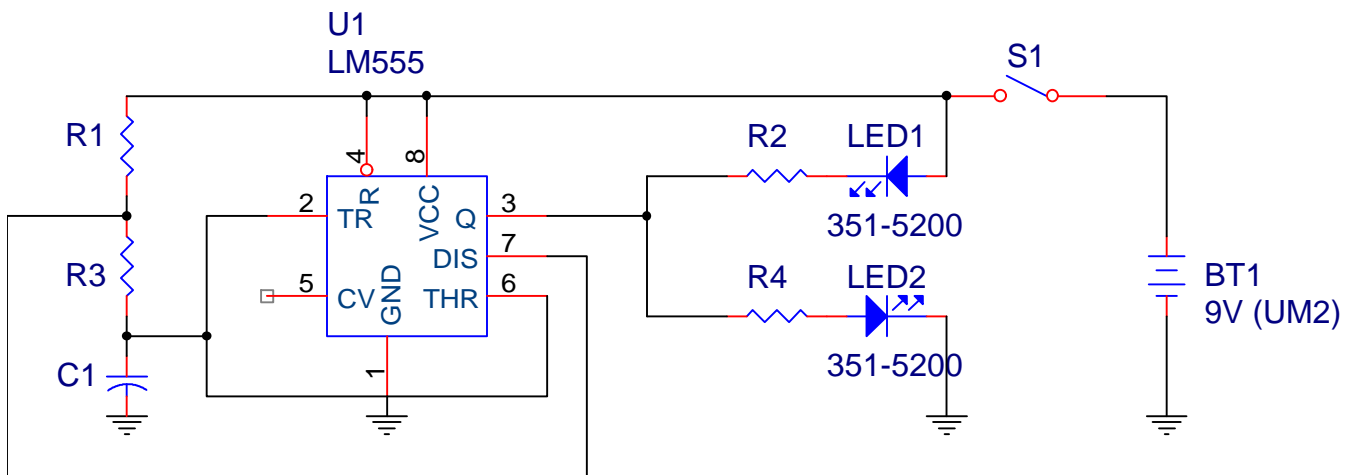


Figure 10-1: The Rough Schematic Diagram of the Project

2. When you've got the schematic worked out, print the page and show it to the instructor (or FA) so that you can get a sign-off. Your schematic *must* be correct to be signed off. The FA or instructor will point out any potential problem areas.

Part II: Incorporating an OrCAD Drawing into an Office Document

1. Create a new *Word* document with the following information at the top:

Your Name
COMP125 Lab 10 for Professor Wheeler

2. Using the Windows clipboard, insert the schematic diagram you created in *OrCAD* into the *Word* document.
3. Write a few paragraphs summarizing:
 - a) The procedures you followed in completing this experiment.
 - b) What you have learned in this experiment.
- Make sure you spell and grammar check this document. Have a classmate proofread it before turning it in.
4. Print this document and attach it to the back of this lab. You're done!

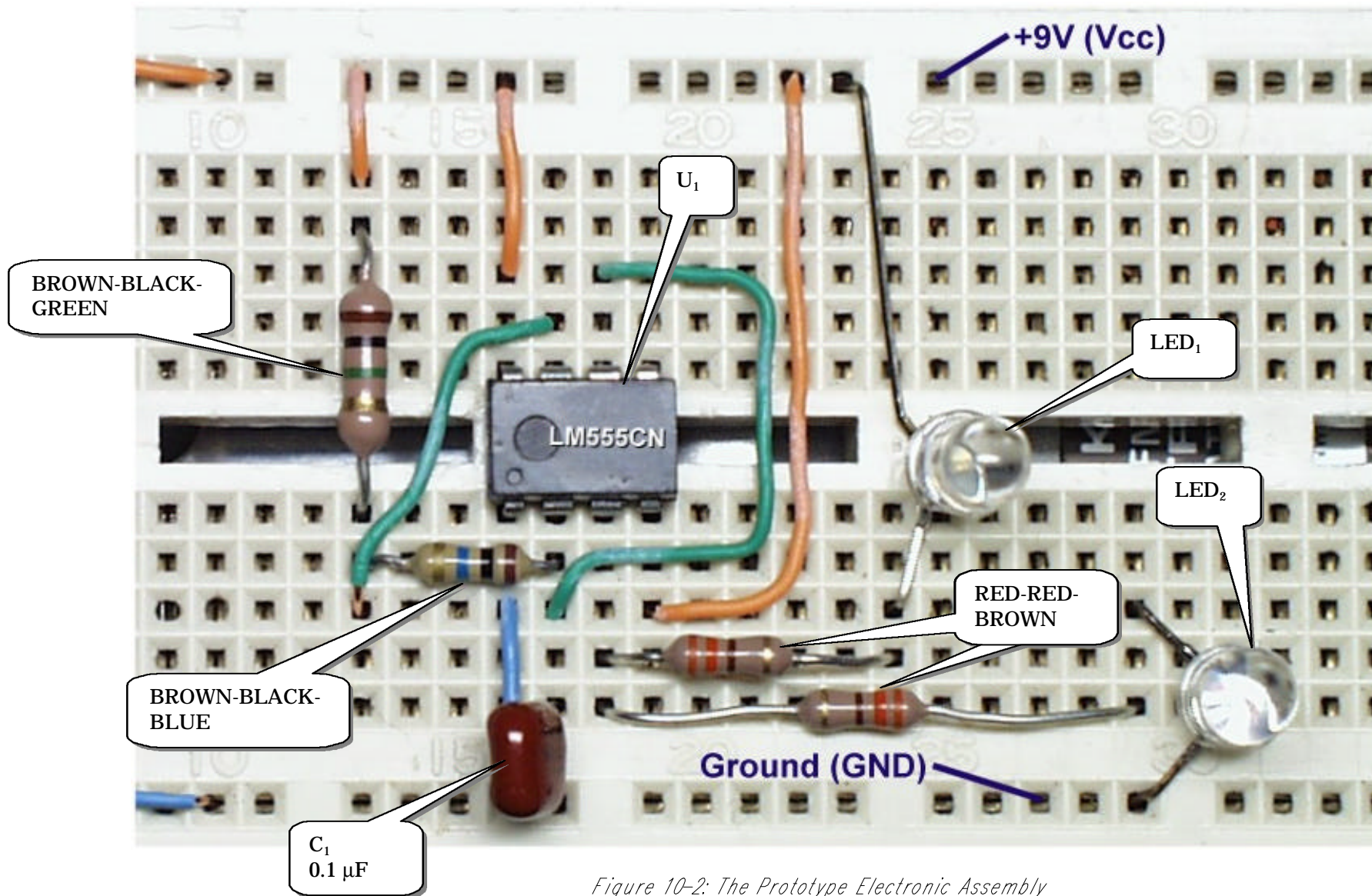


Figure 10-2: The Prototype Electronic Assembly

Name: _____

Sign Off: _____

COMP125

Lab #11: Microsim PSPICE

Experimental objectives:

1. Acquire a schematic diagram using the Microsim schematic capture program.
2. Perform a DC (steady-state) analysis of a series-parallel circuit using the PSPICE simulator.

Materials required:

1. Blank diskette

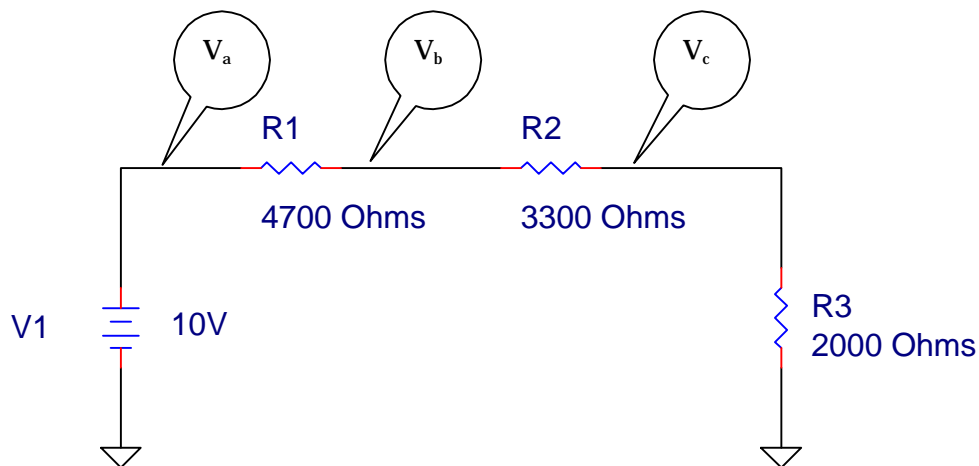
Introduction:

Simulators are an important tool for modern electronics design. Before the introduction of simulator software, engineers had no way of verifying the behavior of systems without actually building (prototyping) them first, short of doing painstaking mathematical analysis. Simulation tools make it much easier to test circuit designs. You should become familiar with as many simulation packages as you can.

The Microsim software we'll be using in this experiment can be used for a wide variety of circuit evaluation tasks. Both steady state (long-term) and transient (short-term) responses can be calculated, and frequency effects can be evaluated as well. Even non-linear effects can be modeled with the software, which goes far beyond what conventional pencil-and-paper circuit analysis tools can do.

Part I: Simulating a Simple DC Circuit

1. Launch the Microsim schematic editor, and enter the circuit shown below.



2. Under the *Analysis* menu, select the appropriate items so that the calculated DC voltages and currents will show up on the schematic diagram.
3. Calculate the voltages at each point in the circuit using Ohm's law. Show your work (including all defining equations and algebraic steps) below:

$$V_a = \underline{\hspace{2cm}}$$

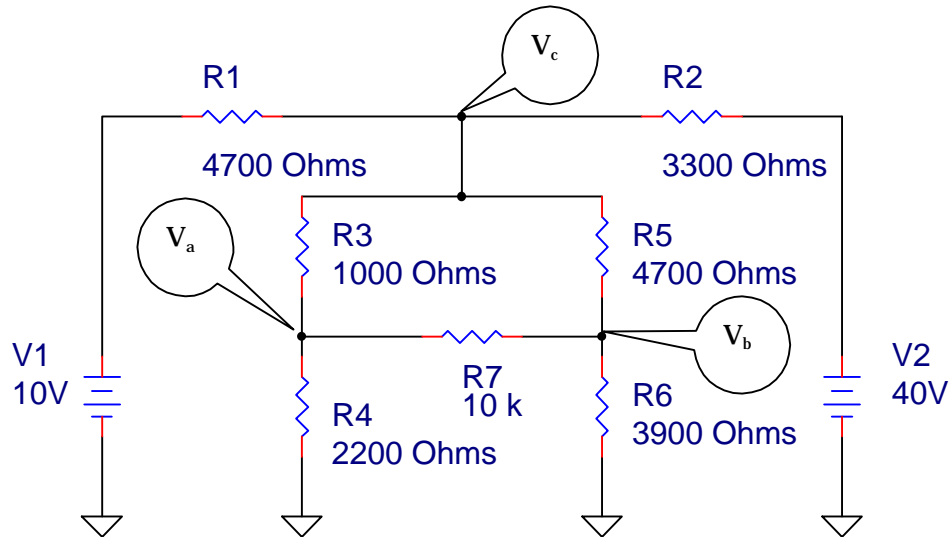
$$V_b = \underline{\hspace{2cm}}$$

$$V_c = \underline{\hspace{2cm}}$$

4. Press the F11 key to simulate the circuit and obtain a sign-off. (Your complete work must be clearly shown in step 3 above to obtain a sign-off.)
5. Print the results and attach them to the back of this lab report.

Part II: A More Complex Circuit

1. Launch the Microsim schematic editor, and enter the circuit shown below.



2. Simulate the circuit (F11) and again print the results.

Note: Only the voltages V_a , V_b , and V_c are needed.

3. Write a summary of what you've done in this experiment using a word processor. This summary should be a minimum of 200 words in length. You should detail:

- The procedures followed during the experiment (you are required to include the schematic diagrams within the document.)
- All the calculations performed in step 3 in part 1 (you must use the equation editor).
- What you have learned in this experiment.
- Make sure you spell and grammar check this document. Have a classmate proofread it before turning it in.

4. Print this document and attach it to the back of this lab. You're done!

Name: _____

Sign Off: _____

COMP125

Lab #12: Introduction to Turbo C++

Experimental objectives:

1. Navigate the Borland Turbo C++ IDE (Integrated Development Environment)
2. Debug a simple C program

Materials required:

1. Blank diskette (for backup and storage of C source code)

Introduction:

C and C++ have become the industry-standard languages for software development. The original version of the language, C, was developed by Brian Kernigan and Dennis Ritchie of Bell Laboratories in the 1970s. The UNIX operating system is largely written in C. C has become popular for three reasons. First, C encourages structured programming techniques, which improves software portability; second, C allows a programmer direct access to hardware features (where necessary) while maintaining a high-level programming interface; and last, C programs run very fast -- nearly as fast as those written in assembly language.

C++ is a superset of C. It allows a programmer to write *object-oriented* (OOP) programs. In OOP, the programmer concentrates on specifying the *behavior* of the objects in the application. *Objects* are groups of data intended to represent a real-world entity. In general, programs written in C will compile under a C++ compiler (though the C++ compiler uses stricter error-checking rules, and errors in the C program that made it past the C compiler may be trapped by C++).

The two C environments available in our lab are Borland Turbo C++ and Microsoft Visual C++. The Borland environment is quite dated -- but it is an excellent platform for learning C concepts. With Visual C++, state-of-the-art Windows applications can be created.

Part I: Compiling and Debugging a C Program

1. Launch Turbo C++ by double-clicking on its desktop icon.
2. Select *File, New* from the Turbo C++ menu bar.
3. Type in the program on the next page.
4. Press [F9] to compile and link the C program. (You may get errors at this point.) Correct the errors as needed!
5. When you get a clean compile, run the program (CTRL [F9]). When it runs correctly, get a sign off.

Don't forget to save your code frequently!

```

////////////////////////////////////
// Program: TempCht.cpp
//
// Purpose: Produces a table for Fahrenheit to Celsius conversions,
//          given the user's choice of input range.
//
// Inputs: Lower and upper limits of Fahrenheit temperature
// Output: A table of temperature conversions.
//
// Author: Harvey Gliptnor
// Version: 1.00
// Revision history: None
////////////////////////////////////

#include <stdio.h>
#include <conio.h>
#include <math.h>

char* szBanner = "\n  Degrees F      Degrees C\n-----\n";

void main(void)
{
    int nLowTemp, nHighTemp, nLineCount; // User input values, and screen line counter
    double dFahrenheit, dCelsius;       // Holds Fahrenheit and calculated Celsius temperatures

    // Inform user about program purpose here

    printf("\n\nFahrenheit to Celsius conversion chart utility.\n\n");

    // Get user inputs

    printf("What is the starting temperature in F?");
    scanf("%d", &nLowTemp );

    printf("What is the ending temperature in F?");
    scanf("%d", &nHighTemp );

    // Bail out if user input is invalid

    if ( nLowTemp > HighTemp)
        {
            printf("\n\nHigh temperature can't be less than the low temperature!\n")
            return;
        }

    printf( szBanner); // Print the table heading (banner)

    nLineCount = 0; // Reset line counter to zero (no lines printed yet)

    for( dFahrenheit = nLowTemp ; dFahrenheit <= nHighTemp ; dFahrenheit++)
        {
            dCelsius = ( dFahrenheit - 32 ) * 5./9. ;
            printf("%10.2lf %10.2lf\n", dFahrenheit, dCelsius );

            if ( (nLineCount) && ( (nLineCount % 15) == 0) ) // Pause every 15 screen lines
                {
                    printf("\nPress a key to continue...");
                    getch();
                    printf("\n\n%s", szBanner ); // Print new copy of table heading banner
                }
            nLineCount++;
        }

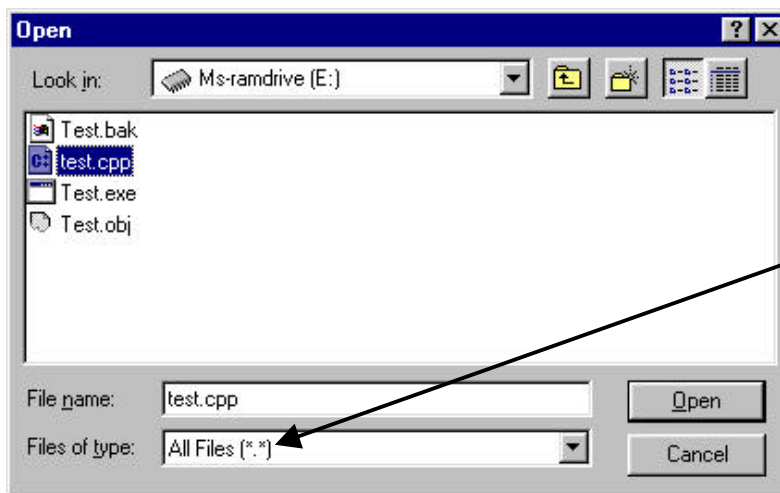
    // Run completed, wait for user to press a key so that last output text remains visible
    printf("\nPress a key to continue...");
    getch();
}

```

Part II: Getting Printed Output under Windows NT

Turbo C++ wasn't intended for operation under Windows NT (though it works quite well in this environment). There's one feature that may not work well, and that is printing. There's an easy way around this, however. You can use either Microsoft Word or Notepad to open the CPP source file, and print its contents.

1. Make sure that you have saved your source code from part I. You can let Turbo C++ remain active with no problems, if you so wish.
2. Open *Notepad*. Choose *File, Open* from the main menu.
3. Navigate the *Open* dialog to the location of your C++ source code:



4. After opening the CPP source (it is merely text), you can do any normal operation on it, including printing.
5. Print your source code, and attach a copy to the back of this experiment.

Name: _____

Sign Off: _____

COMP125

Lab #13: Quadratic Equation Solver

Experimental objectives:

1. Design an algorithm to solve a problem in technology.
2. Implement an algorithm in C and create a simple user interface.

Materials required:

1. Blank diskette (for backup and storage of C source code)

Introduction:

A technologist will often run into problems that can't be solved easily by existing application software, such as spreadsheets or mathematics programs. However, that still doesn't prevent the use of the computer in solution of the problem. The technologist often writes "custom" bits of code to solve unique problems. The steps in doing so are as follows:

1. Write a definition of the problem (which includes pertinent information that might be helpful in its solution.)
2. Derive or otherwise obtain an algorithm that will solve the problem.
3. Implement the algorithm in a computer language, and test its operation.

Note that the above steps are part of the general process of software development.

In this experiment, you'll work out the C code to solve a quadratic equation. As you recall, a quadratic equation has the form:

$$ax^2 + bx + c = 0$$

To derive its solution, we first divide both sides by a to get:

$$x^2 + \frac{b}{a}x + \frac{c}{a} = 0$$

We then solve by completing the square:

$$x^2 + \frac{b}{a}x + \left(\frac{b}{2a}\right)^2 - \left(\frac{b}{2a}\right)^2 + \frac{c}{a} = 0$$

Note that the first three terms of the equation, $x^2 + \frac{b}{a}x + \left(\frac{b}{2a}\right)^2$, are in fact a perfect square that can be expressed as $\left(x + \frac{b}{2a}\right)^2$ which allows us to express the original equation as follows:

$$\left(x + \frac{b}{2a}\right)^2 - \left(\frac{b}{2a}\right)^2 + \frac{c}{a} = 0$$

By rearranging the terms, we can get:

$$\left(x + \frac{b}{2a}\right)^2 = \left(\frac{b}{2a}\right)^2 - \frac{c}{a} = \frac{b^2 - 4ac}{4a^2}$$

Taking the square root of both sides,

$$x + \frac{b}{2a} = \pm \frac{\sqrt{b^2 - 4ac}}{2a}$$

Subtracting $(b/2a)$ from both sides yields the final formula:

$$x = -\frac{b}{2a} \pm \frac{\sqrt{b^2 - 4ac}}{2a} \quad (\text{Note that the } \pm \text{ symbol means that two roots are possible.)}$$

The Algorithm for Solution of a Quadratic Equation

The basic steps your program must take to solve the quadratic equation are as follows:

- ✓ Ask the user for the values of the a , b , and c coefficients. (Store these in `double` variables.)
- ✓ Check the *discriminant* of the solution, which is the fragment $b^2 - 4ac$ in the equation above. If the discriminant is less than zero, the roots are complex (having a real and imaginary part) and are equal to: $x = -\frac{b}{2a} \pm j \frac{\sqrt{|b^2 - 4ac|}}{2a}$ Where j is the square root of (-1) and $| \quad |$ is the absolute value. In this project, you are not required to solve for the complex roots, but your program must report their presence (and stop).
- ✓ If the roots are *real*, calculate them. You might store them in variables $x1$ and $x2$ which would be `double` types.
- ✓ Report the results to the user and end the program.

Program Foundation

Use the following as a foundation for your program. Remember to implement each of the steps of the algorithm above. When you're finished, get a sign-off and attach the finished code to the back of this experiment.

TIP: Use `sqrt()` to find the square root of a number.

```
////////////////////////////////////
// Program: Quad.cpp
//
// Purpose: Solves the quadratic equation
//
// Inputs: Coefficients A, B, and C from user
// Output: The two equation roots, or a warning message if the roots
//         are complex.
//
// Author: Harvey Gliptnor
// Version: 1.00
// Revision history: None
////////////////////////////////////

#include <stdio.h>
#include <conio.h>
#include <math.h>

void main(void)
{
    double a,b,c;           // Place to store the user's coefficients
    double x1,x2;          // The answers are stored here

    // Inform user about program purpose here

    printf("\n\nWelcome to the quadratic equation solver.\n\n");

    // Get user inputs

    printf("What is the value of coefficient A?");
    scanf("%lf", &a );

    // todo: add logic for input of B and C

    // Check discriminant term (b*b - 4*a*c)

    if ( (b*b - 4*a*c) < 0)
        {
            printf("Imaginary roots!");
            return;
        }

    // todo: if roots are real, store them in x1 & x2 and print them.

    // Run completed, wait for user to press a key so that last output text remains visible

    printf("\nPress a key to continue...");
    getch();
}
```