UNIVERSITY OF RHODE ISLAND

ELE 405 - Digital Computer Design

Spring 1997

Project 1

Introduction to the ELE UNIX Environment and the Mentor Graphics Design Tools

Due: Tuesday, February 4, 1997, at the beginning of class.

1 Overview

The main purpose of this project is to become familiar with the ELE department's UNIX computing environment, and to gain experience with the Mentor Graphics system's logic timing/functional simulator. Additionally, you will design a basic element used in future designs, namely a 1-bit register. This document contains brief tutorials in Part I to the ELE system in Section 2, Netscape in Section 3 and four of the Mentor programs in Section 4. Part II, in Sections 5 and 6, has the details on Project 1 itself. Appendix A contains some tips on how to debug a circuit.

Part I
Tutorials

2 Getting Started

By this time you should have an account on the ELE computing systems. You should also get an account on the Engineering Computing Center (ECC) systems of the College of Engineering, located on the third floor of Bliss Hall. The ECC is the main place to work. Alternatively, the machines in our Computing Lab located in Kelley 116 can be used.

Most UNIX programs will also work on character-based terminals, i.e., over the phone lines, but many, including our Mentor Graphics CAD tools, need to be run on workstations. There are 12 SPARCstation 5's in the ECC. Kelley 116 has 5 SPARCstation 5's which are all restricted from undergraduate use, 7 Sun IPC's and a DECstation 5000. Some details on these machines are:

Sun SPARCstation 5's: These machines are the best of our workstations. Because those in Kelley 116 are reserved for faculty and graduate students, the main place for you to work is in ECC where the machines are available to everyone. When running Mentor Graphics on the machines at the ECC, always run them locally, i.e.: you must **NEVER** open a remote window and try to run the tools on Leviathan (the ELE server). This would only increase network traffic, slow down the system, and in general offer decreased performance for all.

Sun IPC's: These computers are available to everyone. They are less capable than the SPARCstation 5's, but they are still able to run the Mentor Graphics Tools. Keep in mind that if you started your design at ECC, you should continue your design there.

DECstation 5000: This is an old machine. It cannot execute the Mentor Graphics tools directly; you must run them on a remote host, such as Leviathan.

To begin with, read the "Introduction to ELE Computing" document distributed in class. Follow the instructions for logging in given in that document (note: you can't do anything on one of these computers without logging in). See the ECC documentation for how to login on their machines; it's similar. NOTE: you need a different account, obtained from the ECC staff, to login on an ECC machine.

In the following steps, the UNIX commands used are (partially) described in the "Intro..." document.

2.1 Getting Your Directory System Setup

There are a few things you should do to setup your account to make life easier on yourself. The following steps should only be done ONCE. The next time you login, everything will operate as it should.

Command Abbreviations: You can get some useful command abbreviations, or *aliases* from my `.cshrc` file, located in my home (~`uht`) directory.
To make sure that you are in your home (root) directory, type:

```
cd
```

(NOTE: all commands end with a carriage return or "enter" keystroke.)
To save what you have now, type:

```
mv .cshrc .cshrc.old
```

Next, copy the file:

```
cp -p ~uht/.cshrc .        (be sure to include that last period!)
```

Let's make sure the file made it over. Type:

```
ls -al
```

to see a directory listing of ALL of the files in this directory, with lots of information on each. See if `.cshrc` is listed. If not, try to copy it again; be careful.

In order for the new aliases to take effect, either logout and then log back in, or type:

```
source .cshrc
```

The latter command causes your command interpreter (what's actually executing your commands) to read in the `.cshrc` file.

To see a directory listing similar to what you saw a minute ago, now type just:

```
ll
```

Configuring the Keyboard for the Mentor Environment: There's one thing you have to do to make using the Mentor tools easier; you have to make a modification to a root directory entry. Type:

```
mv ~/.ctwmrc .ctwmrc.old
```

followed by:

```
ln -s /usr1/generic/WORLDCTWMRC.MG ~/.ctwmrc
```

This modification lets you use the function or "F_" keys at the top of your keyboard to quickly invoke Mentor commands (Mentor calls them "softkeys"). You should now logout and then re-login for this setting to take effect.

ECC Environment Setup: Mentor Graphics requires that the path be modified and certain environment variables be set in order to function properly. On the ELE system, these parameters are automatically set for you whenever you log in; however, on the ECC system, you need to specify the file which includes all of these modifications. The following steps must be done for the ECC system only.

1. From your ECC home directory, type: `<editor> .cshrc` ...where `<editor>` represents your favorite text editor, such as emacs, vi, xedit, etc.
2. Add the following line near the end of the .cshrc file: `source /tmp_mnt/lev1/.mgstart`
3. Save the file and exit your editor.
4. Log out then log back in, or simply type: `source .cshrc` at the Unix prompt to update your system environment variables. You are now ready to start running the Mentor Graphics Tools.

Directory Setup: Let's make a structure to hold your designs for 405. First, we'll give 405 it's own subdirectory off of your root directory. Type:

```
cd

mkdir ele405
```

Now we'll make room for your first project (this one!). First, put yourself into the 405 directory by changing directories to it; type:

```
cd ele405
```

Type an

```
ll
```

There should be only two lines shown, one with the subdirectory ".", and one with the subdirectory "..". Actually, these are not really subdirectories; in UNIX command syntax, a single period refers to the directory you are currently in, and the double period refers to the parent directory of the one you are in.

You should now be able to make a directory for your first project; name it whatever you want. HINT: just use a `mkdir` command.

Please adhere to the following rules:

- **Rule 1:** Every component for Project 1 should be in the Project 1 directory that you just created.
- **Rule 2:** All components should be at the same directory level, i.e., this is a "flat" directory structure, even though logically your design is a hierarchy of components.
- **Rule 3:** Except in certain noted cases, ALL manipulations of your Project 1 directory and its contents should be made ONLY within a Mentor CAD tool (such as Design Manager [`dmgr`]). This is true of ALL Mentor designs and their parent directories.

Note: If you do not follow Rules 1 through 3, the Mentor tools may get confused and may not work right, and you may thus wind up having to completely re-enter your design *following* the rules, to keep Mentor happy.

2.2 Sharing Files

In future projects, you will have a partner, and you will probably want to work on different parts of the project independently. Since moving Mentor files around can be treacherous[1], it is recommended that once these files are created, they be left and used where they are. This means that you may wind up with some of a project's files in your partner's directory, and vice versa. In order to share these files, and only with your partner, you will set up the file protections on these files specially.

Topic A. The UNIX File Protection System

Every file and directory in the UNIX system has three levels of protection: *user*, *group* and *world*. You are the user, everyone (other users and you) comprise the world, and some selected subset of the world forms a group.

Within each level, there are three types of accesses that can be enabled or restricted independently: *read*, *write* and *execute*. Usually you set up your protections so that only you (user) are able to perform any access on your files; the group and world have no access .

You can determine the existing privileges of the files and subdirectories in a directory by typing

```
ll
```

There are a group of 9 (three levels by three access types) single-character columns towards the left side of the directory listing. Each character indicates the corresponding privilege. The levels are in the order: user, group and world; and the accesses are listed within each level in the order: read, write and execute. For example:

```
rwxr-x---
```

for a file indicates that the user can do anything with the file, members of the group (indicated in another column of the directory listing) can only read and execute the file, but not write it, and the rest of the world cannot access the file at all.

[1] This is because Mentor keeps links to other files within its files, and these are not changed when you move or copy the files with UNIX commands. Even Mentor file commands may not change all of the links automatically.

What you want to do is to set the protections on your Mentor files so that you (user) and your group have read privileges to the files. You do this with the `chmod` and `chgrp` UNIX commands, as well as with the system administrator's aid. Assuming your Mentor files are in the `~username/ele405` directory, perform the following steps:

1. Tell the system administrator the name of your group, and its members' login ID's (you and your partners). He or she will set the group up on the system.

   Both you and your partner should perform the last steps in your respective directories. Note that you may have to redo these steps after you have added any file or subdirectory to the `ele405` directory.

2. Type
   ```
   cd
   ```
   You are now in your home directory.

3. Set the group name of all of the files and subdirectories:
   ```
   chgrp -R <groupname> ele405
   ```
   and of your home directory:
   ```
   chgrp <groupname> .
   ```

4. Set the protections of all of the files and subdirectories (remember, UNIX is case sensitive):
   ```
   chmod -R g+rX ele405
   ```
   and of your home directory:
   ```
   chmod g+rX .
   ```

You and your partner may now read, and thus use, each other's Mentor files (actually, everything in the `ele405` directory), but not write them.

2.3 Quitting Mentor Graphics Cleanly

Mentor Graphics Tools do not have a "Quit" option from any pull-down menus. You may be tempted to simply log-out or *destroy* the window when you are done using the application. If you do this, there is a chance that not all of the Mentor Graphics processes which are running will receive the *kill* signal, so that even after you log-out, they may continue running in the background, wasting valuable system resources. Therefore, to cleanly exit any of the Mentor Graphics Tools when you are done using them, simply click the middle button of the mouse on the upper-left corner of the window and choose "Exit Window".

3 The Web

OK, let's do a little practice World Wide Web (WWW) surfing. This is done using a Web *browser*; there are many of these, we'll use the hands down favorite, Netscape.

To execute it, type:

```
netscape &
```

The "`&`" character causes Netscape to be executed in the "background", allowing you to continue to enter commands.

After a while, you'll see a shadow screen with its upper-left hand corner placed where the mouse pointer is. Position the pointer where you want the Netscape window to be, and click the left button of the mouse; the window should now be filled in with the standard Netscape screen (it says "Netscape" on it).

Since the 405 homepage isn't up yet, we'll practice by going to the ELE Department's home page. Position the pointer in the blank line (towards the top of the screen), and left-click the mouse (press the left button). Now type:

```
http://www.ele.uri.edu
```

Make sure you type a carriage return!

After a few seconds, you should see our department's smart-looking homepage. Contiguous underlined blue text indicates a "hyperlink"; invoking it will cause a "linked" page to appear in the window. How do you invoke it? It takes years of laborious and serious practice to do this, and requires a large monetary donation to the department. (Actually, if you'd rather not do that, just left-click on the hyperlink; that's why surfing is soooo easy!)

Click on a few links to try it out. To retrace your steps, left-click the "Back" button in the upper left part of the window. To keep a "permanent" record of a favorite site (page), pull down the "Bookmarks" menu (by holding down the left button while moving the mouse towards yourself [down]), and choosing the "add bookmark" item (by releasing the button while the pointer is over the item).

Pretty easy, huh? Now you're an official Web surfer! (The real fun comes from just poking around different Web pages, especially at different locations, by following hyperlinks. It can be quite addictive.) OK, you can stop now, and we'll go on. I SAID, "stop now, and we'll GO ON"; that's better.

4 Introduction to the Mentor Graphics Tools

Mentor Graphics is a very powerful software package used to design, and more importantly, to simulate designs in order to verify their functionality. The software was donated to the University of Rhode Island, and we are authorized for up to ten copies (floating licenses). This means that if ten people are currently running the software, you will not be able to start the application. Please note that ELE 405 is not the only class which will be using Mentor Graphics, so it may happen more than a few times that all the licenses are being used. If this happens, you will just have to wait until one becomes available. The software itself is being used in industry today and thus is a very practical application for you to learn.

4.1 Bold Browser: Mentor's Online Documentation

One of the key skills you must master in this course is being able to answer your questions on the tools' operation yourself by using the Mentor documentation; there is no way I can go over all of the details in class necessary to operate even a single tool. Virtually all of Mentor's documentation is online; you can print it out, but do so sparingly, if at all, since you have a small print quota.

The online documentation is accessed via a special program called: "Bold Browser". To crank up the browser, type:

```
bold_browser &
```

after the prompt. After a while, its window will appear. What you see initially is the "Bookshelf", which lists the different categories of online documents. We'll start by looking at some introductory documentation for the overall Mentor system user interface, called the "Falcon Framework".

To get to it, double click on the Bookshelf entry: "Falcon Framework". You'll then see a large list of documents, only a few of which are relevant to what we do. A good place to start learning about any Mentor software package is to look at its "Getting Started with... Workbook". So, double click on: "Getting Started with the Falcon Framework Training Workbook".

The manual now appears in a sub-window of Bold Browser. To make it easier to read, resize the various windows by doing the following: 1) resize the Bold Browser window by putting the pointer on its upper right-hand corner (in the grayish region), holding down the middle button, and selecting "Left"; 2) resize the document's window by clicking on the (faint) square in its upper right-hand corner (the blue region); After all this, you should be able to see the entire front page of the "Getting Started with the Falcon ...." in a window taking up about half of the overall space on the screen.

You should now take a look at this document, turning the pages with the grey button(s) on the right side of the top bar of the sub-window. In general, you can also go directly to a document's Table of Contents and Index with the grey button(s) on the left side of the top bar. The Mentor online documents have hyperlinks that look and work pretty much the same way as they did with Netscape. Use the F7 softkey, identified at the bottom of the browser screen, to "Travel Back" (like Netscape's Back button), etc. (If you are displaying the softkey legend, note that the top choice in the key identification is executed with the key pressed by itself; the second with the Shift key pressed; next with the Control key; and last with the Alt key [look at the center of the identification panel].) (To learn more about the capabilities of the Bold Browser, look at its documentation - using, of course, the Bold Browser itself!)

In the Falcon document, read and work through Modules 1, 6, 7, 4 and 5 in that (suggested) order. (You can ignore the info in Module 5 about "configurations".) Modules 2 and 3 may also be useful, if you have questions about operating elements of the user interface, such as the mouse, menus, windows, etc. NOTE: There will be no test on this material; much of it is intuitive; focus on what you need to absorb to get the job done; don't try to remember every detail; the online documentation is always there for you to refer to.

Mentor tools often have other manuals, e.g., the "Design Architect User's Manual", for learning the capabilities of the tool after looking at the "Getting Started..." intro; and the "Design Architect Reference Manual", which is used usually as its name implies, for reference purposes when you can't figure out how to do something in particular.

4.2 Design Manager

Ideally, you would start this program first and then invoke all of the actual tools from it (including Bold Browser). However, there is a good chance that you will not want to use it, since it uses a lot of a machine's resources, and may thus slow down your work considerably when the primary tools are also being used. File manipulation can alternatively be performed within a tool, by the using the commands in the tool's "MGC>Design Management" pull-down menu.

Just in case you'd like to use Design Manager (DM): to start it, first

```
cd
```

to your Project 1 directory, and then type:

```
dmgr &
```

Expand the DM (Design Manager) window to take up the whole screen. The left subwindow contains icons to invoke the available Mentor tools. We'll only be using a few of them. The middle window displays the structure of your design. Right now, it's probably empty. The "palette" of available operations that can be performed is on the right. DM is where you may properly manipulate the structure (file system) of your designs, in particular copying and moving components. Recall this should NOT be done in UNIX proper.

4.3 Design Architect: Schematic Capture and VHDL Entry

Design Architect is the tool you'll use to enter your designs. At first this will only be via the schematic capture part; in future labs, we'll get into the VHDL part.

First, change to your project directory:

```
cd ~username/ele405/project1
```

To start Design Architect (DA), either select it from Design Manager (DM), by double-clicking on the "design_arch" icon in DM's left window, or execute it directly from a terminal window by typing:

```
da &
```

after the prompt. Once it has started, expand the DA window to take up almost the whole screen (otherwise, it may be hard to see some of the window's contents).

Topic A. Schematic Capture

"Schematic capture" is the process of entering a design into the Mentor system so that it can be manipulated and, in particular, simulated. This is done by "drawing" a schematic on the DA (Design Architect) screen. This process consists basically of two (intertwined) steps: 1) placing the components, e.g., logic gates and flip-flops, on the sheet; and 2) connecting them with virtual wires (lines on the screen).

At this point, we will step through a whole sequence of events using a trivial running example that should illustrate most of the features you need in order to complete the schematic entry portion of this lab. Keep in mind that DA has much more to offer than just what is described here. You will use these same steps when entering your Project 1 design, so if you have a problem you can always refer back to this section of the handout.

Creating a Sheet: Your circuit must be entered into a page which is called a "sheet". To start a new schematic, you need to open a new sheet for it. Press the "Open Sheet" button on the command palette over on the right hand side of the screen to do this. A window will pop up prompting you for the component name. Make sure you append the directory in the box with a suitable name for your

new circuit (so it looks like ~username/ele405/project1/part1). Click on "OK", and a new blank sheet should appear on the DA main window. You are now ready to place components onto it.

Placing a Component: Placing a component itself has two parts: 1) selecting the component to be placed and 2) actually putting it where you want it to be on the sheet. When you choose a component, not only are you selecting a picture or *symbol* of the component, but more importantly you are calling up the *model* of the component, which is used during simulation to determine the component's output values over time, given its time-varying input signals.

OK, let's get an AND gate on the screen. Click the "Library" square in the "palette" on the lower right part of the screen; a list of possible digital libraries of components will appear where the palette was. Select "ls_lib". This library contains models for the 74LSxx series of SSI/MSI digital logic. This logic family was in great use a while ago, and is still used for "glue" logic. Although we will not be building a physical computer, we'll use models of these real parts to mimic the real thing. In fact, at the end of the course, after you have designed your computer, you could even build the computer, on your own, using real 74LSxx components, and it should work. Back to the AND gate.

You should now see the list of 'LS part numbers available. Select "74ls08". Its symbol will appear in the small window above the palette. Use the mouse to position the symbol on the sheet, then left-click the mouse and the gate symbol will be placed. Oh, you didn't want to place it there; no problem, just make sure it's selected (an object appears dotted in white when selected) and click the right-mouse button. Choose "Move" and place the symbol in a new location. Did you prefer the gate where it was the first time? If so, click the right-mouse button again and select "Undo". If you are happy where it is, either left-click the mouse on the gate again, right-click the mouse and choose "Unselect", or choose "Unselect All" from the palette so it is no longer selected.

One thing you may have noticed is that there are some pretty handy features available when you right-click the mouse. The functions available change depending on whether the gate is selected or not, and whether you are on the palette or the sheet. You should take a look at some of the other items that appear and try to figure out what they do. Most should be very obvious. For example, to return back to the first palette which appeared on the screen, click the right button over the new palette and choose "Display Schematic Palette".

Wiring: OK, let's connect some virtual wires to the inputs and output of the AND gate. Choose the "Add Wire" button from the (Schematic) palette. A small dialog box pops up indicating that you are now adding a wire to your sheet. Now place the cross-hairs of the cursor onto one of the inputs and left-click the mouse. This anchors the wire to the input of the gate. Move the cursor to the left of the gate and extend the wire to an appropriate length. When you think the wire is long enough, double-left-click the mouse to complete that segment of wire. Wire the other input pin in a similar fashion. Let's do the same for the output, but this time instead of double-clicking the mouse to terminate the wire, single-left-click the mouse and move the cursor either up or down. If you did it right, you should have "bent" the wire in one direction or the other. Bend it once more so it is horizontal again. Finally, terminate this wire.

Each pin should now have a wire connected to it, so I guess we are done using our wiring tool for now. Simply click the "Cancel" button on the small yellow dialog box to put your wiring tool away. The cross-hairs should have disappeared when you returned the cursor over the sheet. Lastly, unselect everything (F2 button).

Adding ports: Now let's add input ports to the input wires and an output port to the output wire. To do this, select the "portin" component in the "ls_lib" library and place it so it is connected to one of the input wires. Now you can left-click the portin symbol in the upper-right corner to grab another one for the second input. Place this one at the other input. For the output port, select "portout" from the library and follow the same procedure. These ports indicate the external connections as seen from outside this sheet, much the same way you only see the external pins which are available on real chips.

Labeling: The default name for all wires is "NET", but since it is not a good idea to have them all labeled the same, we will rename the ports so they are more meaningful. (Mentor logically connects all ports

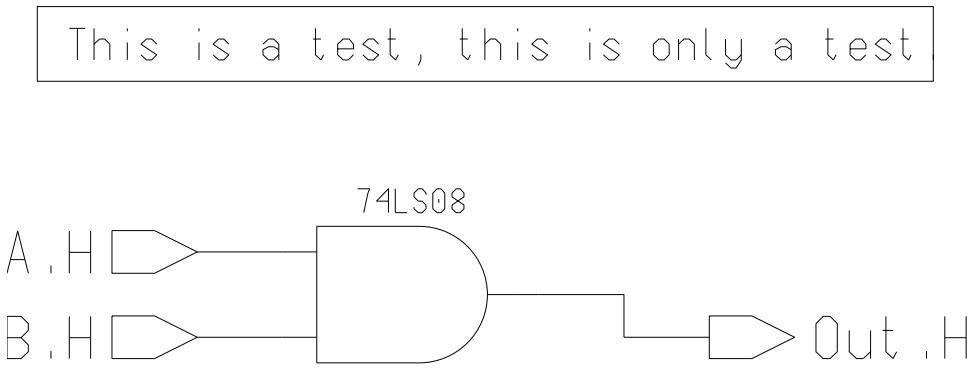

Figure 1: *AND Gate* Schematic Entry Example 1.

with the same name, so if we don't rename the ports, all of the inputs and outputs will be shorted together.) Rename the two inputs "A.H" and "B.H", and call the output, "Out.H". To do this, you must first tell Mentor Graphics that you wish to be able to select component properties. By default, you are not allowed to do so. Press the "Set Select Filter" button on the schematic palette, and when the window appears, click the "Properties" box, then click "OK". Next, select one of the input port names, currently called "NET", then right-click the mouse and choose "Change Values:". A dialog box will pop up at the bottom of the screen. Type the new net name in the "New Value" text window then click "OK". Do the same for the other two ports. Remember to have only one name selected at a time and no components selected (including ports).

This process can be partially automated, when you have a large number of ports to name. This can be done by pressing the "Text" button on the schematic palette, then clicking the "Change Value" button on the new palette. Once you select all of the names to be changed, you will be queried for each new name in turn.

Documentation: We should give our schematic a title so everyone, including yourself, will know what it is we have entered. For this example, it is quite obvious, but for a complex circuit it may not be immediately apparent what the circuit represents. We should also add some information about who entered it, when it was entered (or last modified), and perhaps for what purpose.

To add text such as this to the sheet, click the "Text" button on the main palette, select the "Add Comment" option, and type your comment into the small text window that pops up. When you are done typing, place the information in a convenient location on your sheet. Repeat this for each line of text you wish to add to your sheet.

In addition to text, various shapes can be added to your schematic. As a simple example, draw a rectangular frame around your name, date, and purpose. Accomplish this by selecting "Draw" from the palette and choosing "Add Rectangle". Click and hold the left button of the mouse where you want to start the rectangle, then drag the mouse to the opposite corner and release the button. When you have finished entering your design, it should look something like the circuit shown in Figure 1 (but with better text :-)).

Note: A quicky way to add a title box is to use the "Add Border" command. Select "Add Sheet Border" from the "Edit" pull-down menu.

Topic B. Manipulating Your Schematic

Once you are done entering your design, you can do several things with it. One obvious thing is to *simulate* it to verify that you connected everything together correctly and that your design is correct. Simulation is done using a different application, namely the QuickSim II tool, which is described in Section 4.4. Before we move onto the simulation aspect though, let's discuss some of the less-obvious things that DA provides, and which you need to know in order to get through the whole design process successfully.

Checking: The design has to pass certain checks before it can be successfully registered (and saved). Do this by using the "Check" pulldown menu, and selecting "sheet". Once DA has checked your design, a window will pop up giving a report of the check. All "errors" must be fixed. "Warnings" may or may not need to be fixed for your design to work, depending on the warning. For example, an unconnected output triggers a warning, but this may be perfectly fine for your circuit (like having an unused Q output from a flip-flop). You should look at the warnings and make your own determination.

Registration: Once you have the schematic entered (it's on the screen), you have to "register" it, i.e., make it known to the rest of the world. This is done automatically when you save the sheet, which you should do often DURING design entry, in case something goes wrong (the system dies), so that you don't lose all of your work. Save the sheet by selecting: "Save" from the "File" pulldown menu.

Making a Symbol: In order for your component's design to be used by some other component, you need to give your design a "symbol"; this is a box (typically) with labeled pins coming out representing the component's inputs and outputs. Making a symbol is like packaging your design into a chip so that all the implementation details are hidden, and all that is seen to the outside world are the inputs and outputs (exactly like real IC's).

Symbols can be generated manually (which is highly tedious), or automatically (which takes seconds). To do the latter, select "Generate Symbol" in the "Miscellaneous" pulldown menu. The symbol will appear in its own window. It must also be checked and registered. Note that the first time you check a symbol, it may generate warnings about non-existent pins. Just save the symbol and recheck. You should also give the symbol a name with the "Text" palette.

NOTE: a component must have no more than one (1) symbol! Any more, and DA gets confused, and your circuit won't work. If you add or remove an input or output to the component, you must recreate the symbol. Do it the same way as you originally created it, but MAKE SURE the "replace old symbol" button is active in the dialog box when you do it.

Using Your Designs: To use a component you created earlier in another component (like a one-bit register invoked 8 times to form an 8-bit register), just call up the symbol of the old component, and place it, as you did with the canned parts (e.g., the AND gate), in a sheet of the new component.

THIS IS VERY IMPORTANT: In order to keep the amount of work down, and to keep your designs logically organized, you should reuse components when possible, and construct your design hierarchically, with one component (say the data path) using another component (say an adder), which is in turn composed of other components (say full adders), and so on. This has another VERY IMPORTANT consequence: you can (and should) simulate each component separately, as you build each component, thereby making your final system debugging job MUCH easier, if not trivial.

Symbol Variations (DeMorganizing): To use a DeMorgan'd version (negative logic) of a standard (library) component, to achieve a clean mixed-mode notation, select the component, press and hold the right button of the mouse, and select: "Replace/Alternate Symbol". You'll have to do this operation possibly a few times to get the right alternate symbol.

Printing in Mentor: You will normally use the "hagar" laser printer in ECC, or the "qms" laser printer in Kelley 116. Normally, when you need to print something from a UNIX shell (command interpreter), a variation of the command: `lpr -Pqms <myfile>` is used. In Mentor tools, before you print anything, you must setup the printing system by selecting "Setup/Printer..." from the MGC pulldown menu. Select "Postscript" for printing text, or to a file, and set the name of the printer to: "`mghagar`" or "`mgqms`"; NOTE that this differs slightly from the actual name of the printer.

4.4 QuickSim II: The Logic Simulator

Once you have entered your design in Design Architect, it has passed all of its checks, and it is registered, it is time to see if it actually does what you designed it to do. This is done using the "QuickSim II" (QS) Mentor Graphics Simulator Tool to verify your design. Invoke it by typing:

```
quicksim ~/ele405/project1/part1 &
```

or the equivalent.

(Alternatively invoke QS by double-clicking on the "QuickSimII" icon of the *Tools* window in DM. Do NOT select the "QuickSimII-FPGA" icon. A window will pop up prompting you to enter the name of the design you wish to simulate. Click on the "Navigator" button located all the way to the right on the window [it may not be visible, so you'll have to move the cursor back and forth over the right edge of the window to encourage Mentor to move the window to the left so it is visible]. When the Navigator window comes up, navigate to your design directory and select the part you just entered. Press "OK", drop the new window on the desktop, and expand it so it takes up most of the screen.)

Select "Open Sheet" from QS's palette, and the sheet containing your design should pop up. Before we can simulate the design, we must set up the timing diagram. Once that is done, we can run the simulator and verify that the circuit behaves as expected.

Configuring QS for Maximum Simulated Delay: When QS is first invoked, it is set to simulate logic without delays. This simulates quickly, but it's not that useful, given the real world :-). Set up QS properly by selecting "Kernel/Analysis" from the "Setup" pull-down menu. In the resulting dialog box, press the "Delay" and "Visible" buttons, then change "Timing Mode" to "(Full) Max", and click the "Transport" button, then "OK".

Adding an Input Stimulus: We need to verify the function of the AND gate. In order to do this, we need to vary the two inputs so we obtain all four possible combinations. The output will then react to these "stimuli", and we can compare that reaction to the known response of an AND gate. If we've entered everything correctly, we should be able to verify that we do indeed have an AND gate.

Select input "A.H" on the schematic; the wire connected to "A.H" becomes a dashed-white line. Now select the "WF Editor" in the QS palette. Choose "Edit Waveform". A new *'Trace'* window pops up and the input signal, "A.H" has been added. Currently, there is no stimulus for this input, so the trace shows a dashed-blue line indicating an unknown value. Let's apply a clock at this input with a 100ns period. Select "Stimulus" in the QS palette, then choose "Add Clock". When the window pops up, enter 100 for the clock period then choose OK. The clock signal has been added to the trace. Now unselect the "A.H" input and do the same for the "B.H" input, except give "B.H" a period twice that of "A.H" (200ns).

Note: A quick way to enter arbitrary timing stimuli is to use the "Toggle" tool in the "WF Editor". Once invoked, put the mouse pointer where you want a transition, and left click; repeat as desired.

Simulating Your Design: Now that you have a stimulus for both inputs, we can add the output to the trace and simulate the design. Select the output of the AND gate then press the "Trace" button on the QS palette. This adds the output to the *Trace* window. The last thing to do is choose "Run" from the QS palette and enter 500 for the time to simulate. Click OK and the output should now display the response to the stimuli for every instant of time up to 500ns. If you've done everything correctly, your trace should resemble that of Figure 2. Note the delay in the output of the AND gate.

Saving Your Results: With even slightly complex designs, re-entering stimuli for every simulation becomes tedious and error prone. Fortunately, QS lets you save and recall sets of waveforms contained in *waveform database* (`.wdb`) files.

The displayed waveforms are usually treated as two different types, stimuli ("forces") and outputs ("results"). To save the stimuli, go to the "Stimulus" palette and press the "Save WDB" button. Choose "Forces", and enter the name of a file (with its full pathname) to hold the data; then hit "OK". To save the results, do the same, just select "Results" above instead of "Forces".

Now say you wanted to recall the saved stimuli, to resume a debugging session begun earlier, for example. You must first disconnect the current stimuli by selecting the "Unload/Waveform DB" choice in the "File" drop-down menu. You should also clear the waveform display by selecting "Reset" in the palette, and choosing "State" and "OK" (as is the norm, don't select "Setup" instead of "State"; that would erase the whole waveform display). Now load the saved stimuli by selecting "Load WDB" on the "Stimulus" palette, select both of the input signals in the window that appears, and then press

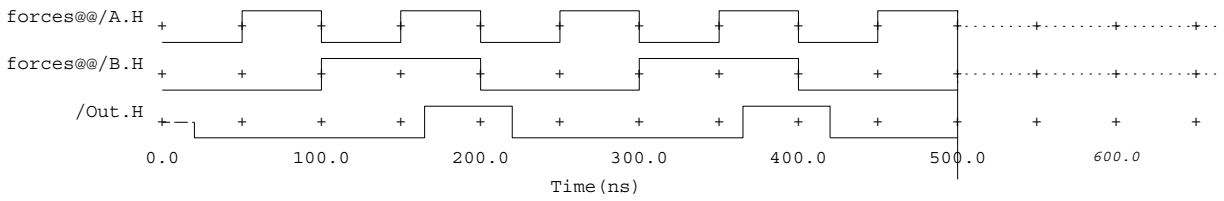

Figure 2: *AND gate* Trace Simulation.

the "Edit Waveform" button in the "WF Editor" palette. The old stimuli should now be back in the *Trace* window, and you can simulate as before.

Interactive Design Changes/Simulations: Often you will want to change a design, and then resimulate it. This can be done rapidly. As an example, let's say that we've discovered we really wanted the complement of the output, so we're going to change the AND gate to a NAND gate, and retest.

Leaving QS alone, and keeping it running, make the DA window active, and edit the sheet to replace the AND gate with a NAND gate (74ls00). As always, check and register (save) the sheet when you're done.

Now go back to the QS window. Update its view of the design by pressing the "Reload Model" button in the "Design CHG" palette. You should see the updated schematic with the NAND gate in the QS window. The results waveform will have been reset, so now re-simulate. The output should have been flipped. (Note that the delay has decreased, since the NAND gate is faster than an AND gate.)

Part II
Project 1

5 Requirements

Your design is to be entered using the Design Architect schematic entry tool. The Mentor Graphics QuickSim II simulator is to be used to simulate the operation of your design, and will be used to verify that your design functions according to the specification. The Mentor system also provides a framework for organizing designs, which we will make use of.

For this project, you are to simulate and verify a very simple digital circuit: a 1-bit register, described in class. You must:

1. Generate a clean design, using mixed-mode notation. Estimate the performance of the design. How fast can you pump data through the register? Do this BEFORE you simulate it! Use the data sheets of the gates and flip-flop to determine this.

2. Create the design with Design Architect, using hierarchical methods and schematic capture. Specifically, your design should have the following components:

   (a) the overall "system", consisting of a multiplexor component (see the next item) and a 74LS74A flip-flop;

   (b) the multiplexor component, designed with NAND gates (74LS00) and an inverter (74LS04);

   (c) any other ELEMENTARY components needed, from ls_lib (NANDs, NORs, ANDs, ORs, XORs, XNORs, inverters, buffers, and simple flip-flops [not registers]).

3. Determine a set of verification data and tests, i.e., you must determine how you are going to demonstrate that your design works. Note that, in general, it is infeasible to test all the possible combinations of

a digital circuit's inputs; therefore, each component should be tested separately, as well as together, using test inputs designed to exercise all aspects of the component.

4. Using the scheme derived in 3, simulate the system to demonstrate functionality and performance.

5. Make a rudimentary cost and performance evaluation of the system: how fast does it actually go (in MHz)? Does this achieve the speed predicted by the design? How much hardware does it use, in 2-input NAND gate equivalents?

6. Document everything. This basically means including computer-drawn schematics, along with timing charts demonstrating the functionality of the design, and a written section discussing the design and the simulations. Include the information from item 5. The format of the report will be discussed in class.

This may sound like a lot, but each item above is a small amount of work for this project.

6 System to be Designed - A 1-bit Register

You have available a 1-bit positive edge-triggered D-type flip-flop (a 74LS74A). You'd like to use it for all of your data registers in the remainder of the course, but it won't work as is, since data is always being latched into it whenever the clock goes from low to high. An Enable line is needed. When high, and the clock transition occurs, new data will be latched into the FF (flip-flop); when low, the old data will be retained. You can't gate the clock. The suggested circuit will be given in class.

(NOTE: I'll give you the circuit this time, so you can learn about the CAD tools without worrying about design that much. This is the last time I'll do this! The real fun begins with the next project...)

A Debugging Notes

Even though our systems are not being physically built, they must still be debugged, and standard debugging methods (slightly augmented) may be used.

Given a bug, the task is to:

1. adequately characterize the symptoms,

2. isolate the fault,

3. determine whether it is a failed component or a design flaw (only the latter in our case - we hope),

4. replace component (you don't have to worry about that), or revise logic,

5. verify the fix.

Most of the time is spent in the first two steps, which together usually happen iteratively; i.e., in the process of characterizing the symptoms, the location of the fault will be narrowed down, which in turn further characterizes the symptoms, and so on. When the symptoms are fully characterized, the fault has been isolated, and in many cases the solution may also be evident at that point. Therefore we concentrate our attention on the first two steps, which we will treat indistinguishably.

There are basically two methods to be used:

- *Divide And Conquer*, or DAC;

- *Cut The Loop*, or CTL.

A.1 Divide and Conquer

DAC is pretty much what the name implies, but let's discuss it a bit anyway. The procedure is:

1. Find a point A in the circuit that has a correct signal; keep going back in the circuit (with large jumps) until you have found one.
2. If a bad signal is obtained at point C in the circuit (point C is initially at the circuit output), and the signal at a previous point A in the circuit is known to be good, then look at a third point B in the circuit; B should be roughly half way between A and C.
3. If the signal at point B is correct, then the fault is between B and C; otherwise, the fault is between A and B.
4. In the former case, set A equal to B. In the latter case, set C equal to B.
5. Repeat, goto 1.

The size of the circuit possibly containing the fault is halved at each step. The bisection method wins again. If you have a lot of probes available, then you can divide the circuit into multiple sections and perform multiple tests at once; this may be particularly nice when performing simulations, in that it may cut down on the simulation time.

A.2 Cut the Loop

In most interesting circuits, digital as well as analog, there is feedback. The feedback gives circuits interesting and complex properties, which may be hard to analyze straight off. In an analog circuit, if you break the feedback loop, the circuit can completely disfunction and signals can go out of bounds; not so with digital circuits! Therefore, if we have a digital circuit with feedback (like a computer) which is not working, and is acting in a strange fashion, one way of temporarily simplifying things is to *break the feedback loop*. Once the loop is broken, the circuit input (if there wasn't one, there is one now) can be force fed data, repetitively if necessary, and the open loop circuit's outputs examined to characterize the symptoms; note that you may have to do some thinking to determine exactly what you should get. Then DAC is applied.

A.3 Summary

I've discussed two basic methods which have always helped me, but be forewarned: they are not foolproof; sometimes debugging must be performed closed loop. Intuition helps, but so does hard analysis of the operation of the circuit, i.e., exercising those neurons (biological, that is).

Good luck, and have fun.