ELEC-E3520 Project Guidelines

Marko Kosunen, Antti Ontronen

19.01.2016

Aalto University
School of Electrical
Engineering

Contents

1 Purpose of the project

During this project you will:

- Learn very basics of the unix command line working environment.
- Get acquainted with the important properties of a text editor (that are really different form the unimportant properties.)
- Lear to simulate with Eldo circuit simulator.
- Learn the basics of LaTeXdocument editing.
- Apply your knowledge of digital gate design to practice.

So, we try to teach you lots of things that are useful when you work in Unix/Linux environment. The aim is to teach you something new, so we kindly ask you to be open-minded and try things out as we instruct.

2 The very basics of Unix command line

Table 2.1: This is the minimum set of commands you need to survive

Command	**Operation**
man	Manual pages, this is your first source of guidance.
pwd	Print the path of working (current) directory.
cd	Change directory.
ls	List the contents of the directory.
mkdir	Make directory
rmdir	Remove empty directory.
mv	Move file/directory (rename)
rm	remove file (option -r will recursively remove also directories -f will force to do so without asking, so be VERY careful when issuing *rm -rf **.
less	Browse a file (no editing).
kill	Kill a process
CTRL-Z	Suspend currently running command.
bg	Continue execution of a suspended program in background.
grep	Search for a pattern in a file.
tar	Create (and pack) a tar archive of files. (Do not use zip or rar if you can use tar.)

2.1 Preparations for the project

In coming sections, notation *#>sometext* corresponds to unix command which is always terminated by hitting RETURN if not instructed otherwise.

You may browse through your command history with arrow keys up and down, or *CTRL+p*. Try it out after you have issued some commands. It is important to understand, that most often you do not need to rewrite commands when you need to repeat them. The very principle of unix command line is that you do not retype if you can avoid that. However, avoiding retyping needs some practicing. See Table 2.2 and try to use the shortcuts whenever you can.

Now it's time to do the actual preparations. First, set up your project directory working environment. Ensure that you are in your home directory.
#>cd

Table 2.2: The most important commands to avoid retyping

Command	**Operation**
TAB	Autocompletion of file/directory names.
history	shows you the previously issued commands. For your convenience, this is aliased to *h* .
!!	Repeats the previous command.
!-2	Repeats the command before the previous one. (guess what !-3 does)
!123	Repeats the command number 123 in command history.
Editing shortcuts with Emacs bindings	**Operation**
CTRL+A	Go to beginning of a line.
CTRL+E	Go to end of line
CTRL+K	Kill (cut) to the end of the line.
ESC+D	Delete next word.
ESC+bckspc	Delete previous word
Editing shortcuts with Vi bindings	**Operation**
	Go through vimtutor, and put bindkey -v in your .cshrc file in your home directory.

Command *cd* is used to change the directory. Without arguments it will change to your home directory. *#>cd ..* will take you one level up in directory hierarchy.

pwd prints the working directory.

#>pwd

Check what is in your directory with *#>ls* or *#>ls -l* to see the long version of listing. You may also try *#>man ls* to learn more about options of ls and the man-command.

Copy recursively the directory */home/E3520/ELEC-E3520_Project* to your home directory.

#>cp -r /home/E3520/ELEC-E3520_Project ./ELEC-E3520_Project

There should now be a directory *ELEC-E3520_Project* in your home directory. Check that it exists with *#>ls* .

Still in your home directory, issue command

#>tar xvzf ./ELEC-E3520_Project/vimsettings

This extracts proper settings for vim text editor to your home directory. These are located in directory .vim and file .vimrc. The "." in front of the filename hides the particular file, but you may see them with *#>ls -a* . Try it out. You may also read the contents of the setup file with *#>less .vimrc* . Don't worry if you do not understand the syntax, this is jut to show how *less* works. You may exit less by pressing *q* .

Now you are equipped with elementary tools of unix to try out some simulations.

3 My first simulation

Go to the project directory.
#>cd ./ELEC-E3520_Project
Check its contents *#>ls -l* . There are two directories *ELDO_Simulations* and *Report_Template* . Go to *ELDO_Simulations*
#>cd ./ELDO_Simulations
and check it contents with *ls* , and again you see two directories *Netlists* and *Cmdfiles* . This is an example of a hierarchical design directory structure. Everything has it's own place (directory) which makes it easier maintain the design. Try to avoid storing files without any hierarchy, but create a proper directory structure for your designs. That helps you finding the files in complex designs. This is very simple design, but you may have hundreds of files for various purposes, and in that case flat directory structure results in complete chaos.

Create one more directory *Wdbfiles* with *#>mkdir Wdbfiles* . Check the result with *ls* . If you made a mistake, you may rename/move the directories and files with *mv*

The *Netlist* directory contains the design netlist. Take a look at the inverter netlist stored at *./Netlists/Inverter.cir* .
#>less ./Netlists/Inverter.cir
It is very simple, and contains only two transistors. Can you point them from the file? How they are connected. Try to sketch a schematic on a piece of paper.

The *./Cmdfiles* directory contains the Eldo-simulator command files. Read through the file *./Cmdfiles/Inverter.cmd* with
#>less ./Cmdfiles/Inverter.cmd
You see lots of library definitions, parameter definitions, option definitions, component definitions, voltage source definitions measurements ans simulation commands. All of this describes a characterization simulation of an inverter. Your job later on during your project is to modify the netlist (.cir-file) to describe a logic gate, find the dimensions of the transistor, and modify the command file (.cmd) to characterize the functionality and delays of the gate with similar kind of simulations.

As you are now familiar with the netlist and command files, you are ready to simulate the inverter with Eldo. First you have to set up the simulator:
#>use advms_17.1
This sets environmental variables for the eldo simulator and "takes the simulator into use". This has to be done only once per shell session.

For simulator, your main source of information is the Eldo manual at
$MGC_AMS_HOME/docs/pdfdocs/eldo_ref.pdf and

$MGC_AMS_HOME/docs/pdfdocs/eldo_user.pdf .
The environment variable *$MGC_AMS_HOME* (among other required environment variables) is defined by use-script you just issued.

Now, to run the simulation, issue command
#>eldo -use_proc all -64b -outpath ./Wdbfiles ./Cmdfiles/Inverter.cmd
You get plenty of text as output on your screen. You may browse through it to see if any errors occurred (no errors at this point should occur), but this text is also saved in a file *Wdbfiles/Inverter.chi*

Your simulation results are output to directory *Wdbfiles* . Two files are the most important.

The file *./Wdbfiles/Inverter.chi* contains more or less the same text as was output to the window including your extracted results and possible also errors. Browse through it with *less* . You may also try to find errors with
#>grep -i error ./Wdbfiles/Inverter.chi
Try also *#>grep -i tdhl ./Wdbfiles/Inverter.chi* . See *#>man grep* to find out what option *-i* does.

The file *./Wdbfiles/Inverter.wdb* is the wave-form database. Open the database file with
#>ezwave ./Wdbfiles/Inverter.wdb
You notice that *ezwave* reserves the command line. Suspend it with *CTRL+z* and put it to the background by issuing command *bg* . For the future, you may start programs directly to the background by appending "&" after the command (like *#>ezwave ./Wdbfiles/Inverter.wdb &*).

Now take a look at the ezwave window. You may add waveform windows by dragging the waveforms from the left side menu window to the blue waveform window are on the right. You may also add single waveforms to existing waveform window by dragging. Try this out couple of times to learn how it works.

Finally, open the Eldo manuals and take a look at those with *evince* pdf viewer
#>evince $MGC_AMS_HOME/docs/pdfdocs/eldo_ref.pdf &
and *#>evince $MGC_AMS_HOME/docs/pdfdocs/eldo_user.pdf &*

Now you now everything you need to carry out your project. However, you still need some knowledge on text editors.

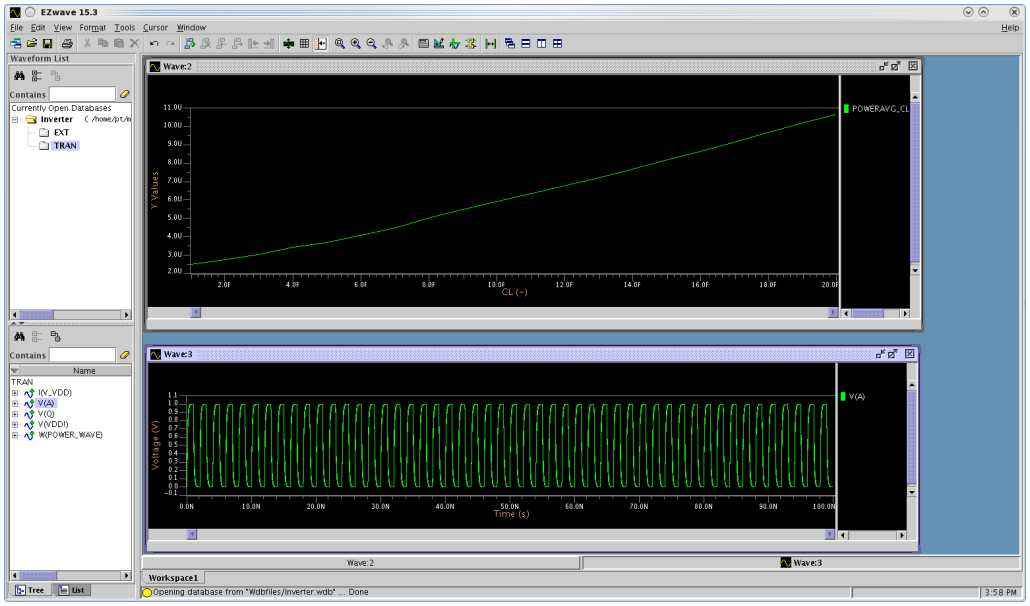

Figure 3.1: Ezwave waveform browser

4 Text editors

There are two excellent text editors in unix: Vi, more recently Vim (vi improved), and Emacs. To get some idea of differences between these two editors, read *https://en.wikipedia.org/wiki/Editor_war.*

The rest of the editors are not worth of mentioning because they do not support the key philosophy of text editors: *maximize the editing speed for a person capable of touchtyping.*

Only guideline that can be given at this point is *pick one and learn that well.* To do that, go through tutorials.

Tutorial for vim: *#>gvimtutor*
(*gvim* is an graphically enhanced version of *vim*). If you pick vim to be your editor, you may launch it with *#>gvim* .

Tutorial for emacs: *#>emacs*
then type C-h t, that is, Ctrl-h followed by t.

To edit a file, just issue a command
#>gvim filename
or
#>emacs filename

Just in case: You may quit *gvim* with *:q!* and *emacs* with *CTRL-x CTRL-c*

The best part: Unix command line supports both emacs and vi keybindings. To try them out you may change the bindings by issuing command *#>bindkey -v* for vi and *#>bindkey -e* for emacs. Especially vi bindings require some practicing. However, the goal is that when you have the bindings in your muscle memory, command line working is very efficient.

You may make this selection permanent by adding a line *#>bindkey -v* (for vim keybindings) or *bindkey -e* (for emacs keybindings) in the file *.cshrc* in your home directory. The emacs bindings are the default.

5 Project report

Now you are familiar with the text editors and have picked one. Next we prepare you for editing the project report with LaTeX. Go to the directory */ELEC-E3520_Project/Report_template*. Then issue the command
#>./makearticle ReportTemplate.tex
Compilation should go through without errors.

The command produces a pdf file *./pdffiles/ReportTemplate.pdf* . Take a look at that with
#>evince ./pdffiles/ReportTemplate.pdf & .

To edit you report, copy the template under name, *ELEC-E3520-YourLastName_YEAR.tex* . Replace YourLastName with your lastname and YEAR and compile it with
./makearticle ./ELEC-E3520-YourLastName_YEAR.tex
Open the resulting pdffile with *evince* . If the compilation was successful, you are ready to start editing your report with your favourite text editor.

6 Project guidelines and grading principles

The topic of the design project will be the characterization of a logic gate (anything more complex than just an inverter) to be implemented in 65nm CMOS process. Characterization includes the simulation of propagation delay and power consumption as a function of supply voltage and external load capacitance.

- Project grading will be based on the study diary.
- The most effective way of providing study diary is to continuously write it by describing arising problems and how you solved those. *Study diary is not a design report, although it may also describe the design*
- Study diary must be written by using template provided to you. Study diaries with layout different from the template will not be graded.

When writing your study diary, please consider at least the following:

- How did you size the transistors in your logic gate (rule of thumb? Switching threshold, tp,LH/tp,HL? Other?)
- Simulate the propagation delay and total power consumption as a function of
  1. Supply voltage
  2. External load capacitance

  (i.e. sweep one quantity at a time, while keeping the other one constant)
- Does the choice of input patterns affect the results? Demonstrate the effect.

The following properties of the document will be graded

- Transistor sizing principles 0-5p
- Simulation setup 0-5p
- Presentation and analysis of the results 0-5p
- Does the document reflect the thinking process during the design? 0-5p

Return the study diary pdf to the MyCourses by the due date.