Objective

Your objective is to develop a 7-segment decoder module to convert a 4-bit binary value into a 7-bit control code for the 7-segment display device on the DE2-115, and to use 8 instances of your this decoder to display values on the 8 HEX displays on the DE2-115 board. For this project, you will connect each set of 4 consecutive switches across the bank of 18 switches at the bottom of the DE2-115 to the inputs of these decoders. This will allow you to use the switches to "key in" a value from `0x00000000` to `0x0003FFFF` for display.

Specific Tasks

1. Download and extract the project skeleton.
2. Run the shell command `./csce611.sh compile && ./csce611.sh program` to make sure your DE2-115 board is working properly. You should see a bouncing "comet" on the 26 LEDs. If not, or if the script fails, ask a TA for help.
    - You can use this same command to re-compile your project and program the board again at any time.
    - If you see a message like `./csce611.sh: line 47: QUARTUS_ROOTDIR: unbound variable`, then you have not set up your shell session to access the Quartus tools properly. Try running `source /usr/local/3rdparty/cad_setup_files/altera.bash`
3. Develop your 7 segment decoder. This will be a standalone Verilog module which takes one 4 bit input, and has one 7 bit output. You can find information about the 7 segment display in the DE2-115 user guide.
    - **TIP**: a skeleton for a decoder is provided in the file `hexdriver.sv`. You are not required to use it, but may find it a helpful starting point.

4 Modify `top.sv` to instantiate 8 instances of your decoder, one for each 7 segment display.

5. Modify `top.sv` to connect each group of 4 switches to one of the 7 segment decoders. Note that there won't be enough for all 8 decoders, so the "upper" ones can be hard-wired to display 0.
6. Modify `simtop.sv` to implement a self-checking testbench.
7. Run your testbench using `./csce611.sh testbench`.
8. Deploy your design on the DE2-115 development board using `./csce611.sh compile ; ./csce611.sh program`.
9. When you are satisfied with your design, pack it up using `csce611.sh pack`.

Deliverables

- You will turn in **one** file, it should be named `CSCE611_Fall2025_hex_yourusername.7z`. You should replace `yourusername` with your actual username that you use to log into the Linux lab computers with. The script will automatically generate a correct file name for you – you usually will not need to modify it.
  - You must turn in your code for this project. Your submission **must** be packed by using the command `./csce611.sh pack`.
  - See also: Appendix 1.
- Each group only needs to submit once via either partner.

Rubric

- (A) 20 points – The hexadecimal value corresponding to the state of each group of four switches, ordered from right to left (LSB to MSB), must be converted into a 7-segment encoding appropriate for the 7-segment displays on the DE2-115 board.
- (B) 20 points – The inputs to the three most significant (left most) decoders must be hard wired to 0-bits. The two most significant inputs to the fourth most significant decoder must also be hard wired to 0-bits.
- (C) 30 points – A corresponding self-checking testbench must drive and check at least two different output values for each of the 5 decoders.
  - In other words, your testbench should include test-cases which make each non-zero hex display show at least two distinct digits.
- (D) 30 points – The design must be deployed and functional on the DE2-115 board.

Maximum score: 100 points.

Grades will be assigned by demos during lab, during office hours, or by appointment. The TA will test your design on the DE2-115, and you should also show them your testbench. You will also submit your code via Moodle. At our sole discretion, we may verify that code submitted via Moodle behaves consistently with what was demoed, adjusting your grade if appropriate.

Additionally, the following may cause you to lose points: * Academic honesty violation, such as turning in another student's code, or demoing someone else's code, but turning in your own. * Code which does not compile.

Project structure

The structure of the provided skeleton is as follows:

- `DE2_115.htm`: log of pin assignments used for the DE2-115 board
- `DE2_115.qpf`: Quartus Project File for this project
- `DE2_115.qsf`: Quartus Script File for this project – this is where project settings such as device, top-level entity, and pin mappings are configured.
- `DE2_115.sdc`: Synopsis Design Constraint file defining information about the system clock.
- `top.sv`: Top-level System Verilog module for the project.
- `simtop.sv`: Top-level System Verilog module for your testbench.
- `hexdriver.sv`: empty module to write your hex decoder in should you so choose
- `csce611.sh`: front-end wrapper for course-specific scripts.
- `scripts/`: course-specific scripts.

You should not directly modify any of the above files except for `top.sv`, `simtop.sv` and `hexdriver.sv`. The QPF and QSF files may be updated using the Quartus GUI or TCL commands. You may add additional SystemVerilog files if you would like.

Appendix 1 – Example Output of `csce611.sh pack`

This shows an example of a terminal session where an assignment was successfully packed.

```
$ ./csce611.sh pack
Gtk-Message: GtkDialog mapped without a transient parent. This is
    discouraged.
Gtk-Message: GtkDialog mapped without a transient parent. This is
    discouraged.
cleaned simulation files
attempt 1/10 to clean project... OK
cleaned hardware build files

7-Zip [64] 9.20  Copyright (c) 1999-2010 Igor Pavlov  2010-11-18
p7zip Version 9.20 (locale=en_US.UTF-8,Utf16=on,HugeFiles=on,8 CPUs)
Scanning

Creating archive /acct/cad3/CSCE611_Fall2025_hex_cad3.7z

Compressing  lab_hex/README.md
Compressing  lab_hex/DE2_115.htm
```

```
Compressing  lab_hex/csce611.sh
Compressing  lab_hex/scripts/jtag.sh
Compressing  lab_hex/scripts/listfiles.tcl
Compressing  lab_hex/scripts/help.txt
Compressing  lab_hex/README.pdf
Compressing  lab_hex/DE2_115.qpf
Compressing  lab_hex/DE2_115.qsf
Compressing  lab_hex/DE2_115.qws
Compressing  lab_hex/DE2_115.sdc
Compressing  lab_hex/DE2_115.sv
Compressing  lab_hex/hexdriver.sv
Compressing  lab_hex/testbench.sv

Everything is Ok
```

