Lab 1.1

Lab 1.1

- In this lab, you will:
  - Learn how to invoke a Verilog simulation
  - Recognize common syntax errors

Lab 1.1

- Using a text editor, enter the following Verilog modules in separate files.
  - Fix any syntax errors or warnings in the specification or any that you may introduce.

hello.v

```
module HELLO
initial $write("Hello World!\n");
endmodule;
```

hi.v

```
module \HI;
initial $write("Hi there!\n");
end module;
```

Lab 1.1

- Using a single Verilog simulator, simulate each module separately.
  - Interpreted simulator (e.g. XL):

```
% verilog hello.v
% verilog hi.v
```

  - Compiled simulator (e.g. ModelSim):

```
% vlog hello.v
% vsim HELLO
% vlog hi.v
% vsim HI
```

  - Compiled simulator (e.g. VCS):

```
% vcs -R hi.v
```

Lab 1.1

- Using the same Verilog simulator, simulate both modules together.
  - Interpreted simulator (e.g. XL): `% verilog hello.v hi.v`
  - Compiled simulator (e.g. ModelSim): `% vsim HELLO HI`
    (no need to recompile)
  - Compiled simulator (e.g. VCS): `% vcs -R hello.v hi.v`
- Reverse the order in which the modules are specified on the command line.
  - Are the results different?
  - Should you care?

Lab 1.1: Optional

- Rename module "HI" to "HELLO", then try to compile/simulate both modules at the same time again.
- What happens?
- What guideline can you deduce from this exercise?

Lab 1.2

Lab 1.2

- In this lab, you will:
  - Work with *always* blocks
  - Work with *initial* blocks
  - Use the *$finish* statement

Lab 1.2

- Enter the following Verilog module in a file:

```
module HELLO;
  always
  begin
  end
endmodule
```

- Simulate the module.
  - Some simulators appear to hang, others core dump or crash. To interrupt the simulation, use Control-C (^C) or Control-Backslash (^\).
  - What is happening?
  - If a warning was issued when compiling the module, it may provide a clue as to what is happening.

Lab 1.2

- To help diagnose what is going on, add a **$write** statement inside the always block.

```
module HELLO;
  always
  begin
    $write("Hi!\n");
  end
endmodule;
```

- Try to simulate the module again.
  - Does the output confirm your suspicions?
  - To interrupt the simulation, press ^C or ^\.

Lab 1.2

- Edit the module and insert a **$finish** statement at the end of the *always* block.
- Simulate the module again.
  - How is the simulation different?
- Edit the module and replace the "always" keyword with "initial". This change turns the *always* block into an *initial* block.
- Simulate the module again.
  - Is the simulation behavior different?

Lab 1.2

- Edit the module and delete the **$finish** statement at the end of the initial block.
- Simulate the module again.
  - Is the simulation behavior different?
  - Why did the simulation terminate by itself?