Icarus Verilog and GTKWave

CS141 Spring 2019

Introduction

Some students might find it useful to have access to a Verilog compiler and simulator that is more lightweight than Xilinx ISE and that can run on Mac and Linux operating systems. You won't be able to program the FPGA without Xilinx ISE but Icarus Verilog is often very useful for testing simulations from the command line without having to boot up Xilinx ISE.

Icarus Verilog is an open source Verilog compiler and GTKWave is a nice and simple waveform viewer. You might even find that only Icarus Verilog is necessary for lightweight printing and debugging (without a waveform viewer).

Installation

Mac OSX

The easiest way to install `iverilog` is using Homebrew:

1. Install Homebrew
2. From the command line run

```
$ brew install icarus-verilog homebrew/x11/gtkwave
```

It might take a while especially if Homebrew needs to update.

Linux

On Debian based distros you can simply run `sudo apt install gtkwave iverilog`. For a more detailed guide see here.

Windows

You can get icarus-verilog binaries from this site: http://bleyer.org/icarus/ and gtkwave binaries from here: http://www.dspia.com/gtkwave.html

However, if they don't work it is recommended that they be compiled from source. Directions are on the icarus verilog installation guide and the gtkwave homepage: http://gtkwave.sourceforge.net/.

How to use

As an example let's use the mux code from the Verilog tutorial:

```
module mux(f, a, b, sel);
    input wire a, b, sel;
    output wire f;

    wire n_sel, f1, f2;

    and #2 g1 (f1, a, n_sel),
           g2 (f2, b, sel);
    or #2 g3 (f, f1, f2);
    not g4 (n_sel, sel);
endmodule

module testmux;
    reg a, b, sel;
    wire f;

    mux uut (f, a, b, sel);

    initial begin
        $monitor ($time, " a = %b b=%b sel=%b f=%b", a, b, sel, f);
        a = 1;
        b = 0;
        sel = 0;
        #10 // wait for signals to propagate
        sel = 1; // switch sel to 1
        #6 $finish;
    end
endmodule
```

To use Icarus Verilog you can run the compiler on the command line:

```
$ iverilog *.v -o out.vvp
```

This will compile all Verilog files in the current directory and create an `out.vvp` file. You can run the simulator by executing

```
$ vvp out.vvp
 0 a = 1 b=0 sel=0 f=x
 4 a = 1 b=0 sel=0 f=1
10 a = 1 b=0 sel=1 f=1
14 a = 1 b=0 sel=1 f=0
```

We see the correct simulation output printed at the command line.

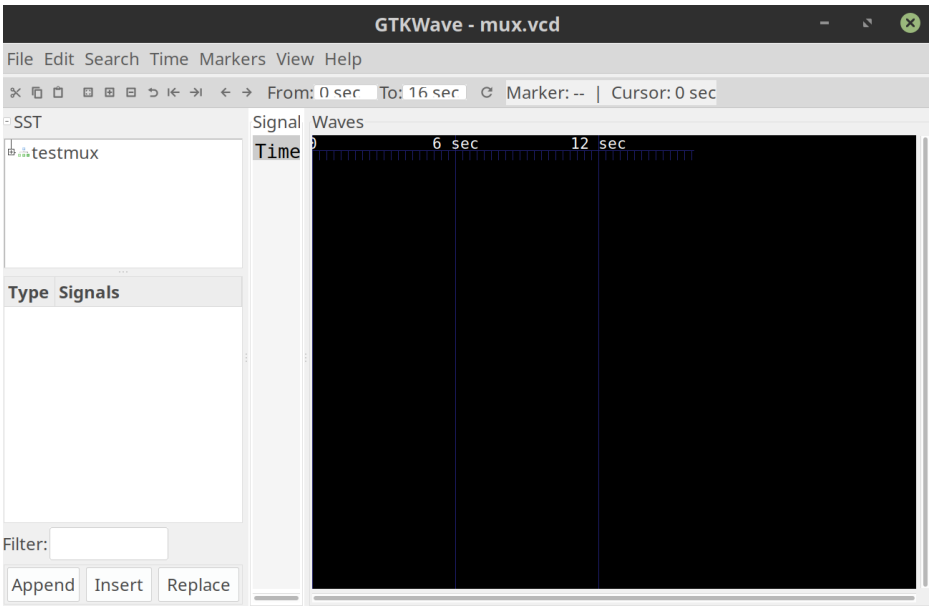
To get a waveform viewer we can add the following code at the beginning of the `initial begin` block:

```
// vcd dump
$dumpfile("mux.vcd");
// the variable 'uut' is what GTKWave will label the graphs with
$dumpvars(0, uut);
```

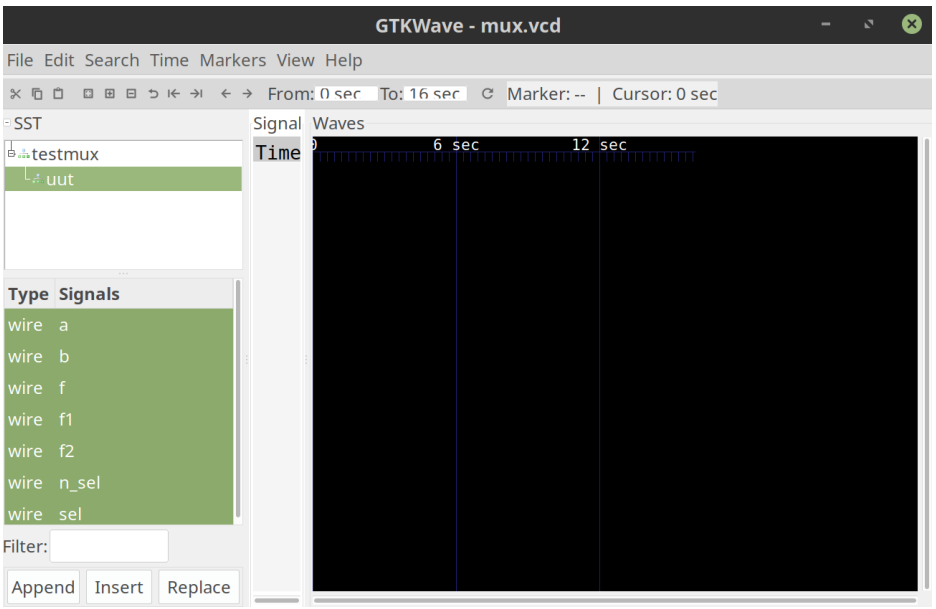
Now we recompile and run the simulation again and we get a file called `mux.vcd` which contains information for GTKWave. We can open this file with GTKWave:

```
$ gtkwave mux.vcd
```

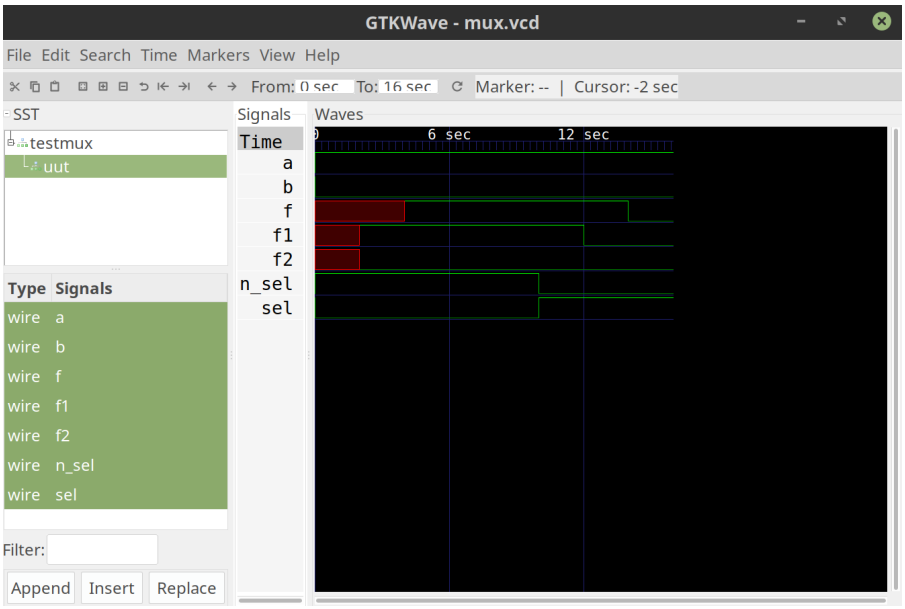
You should see an empty waveform viewer appear:



Expand `testmux` and select `uut` and you should see a list of signals in the module in the bottom left:



Select all the signals and click append:



Now you can see the signals as they appear over time.