Week 1 RISC V Tape-Out Report on Yosys Tool

By

Dr M A. Raheem,

Department of Electronics and Communication Engineering

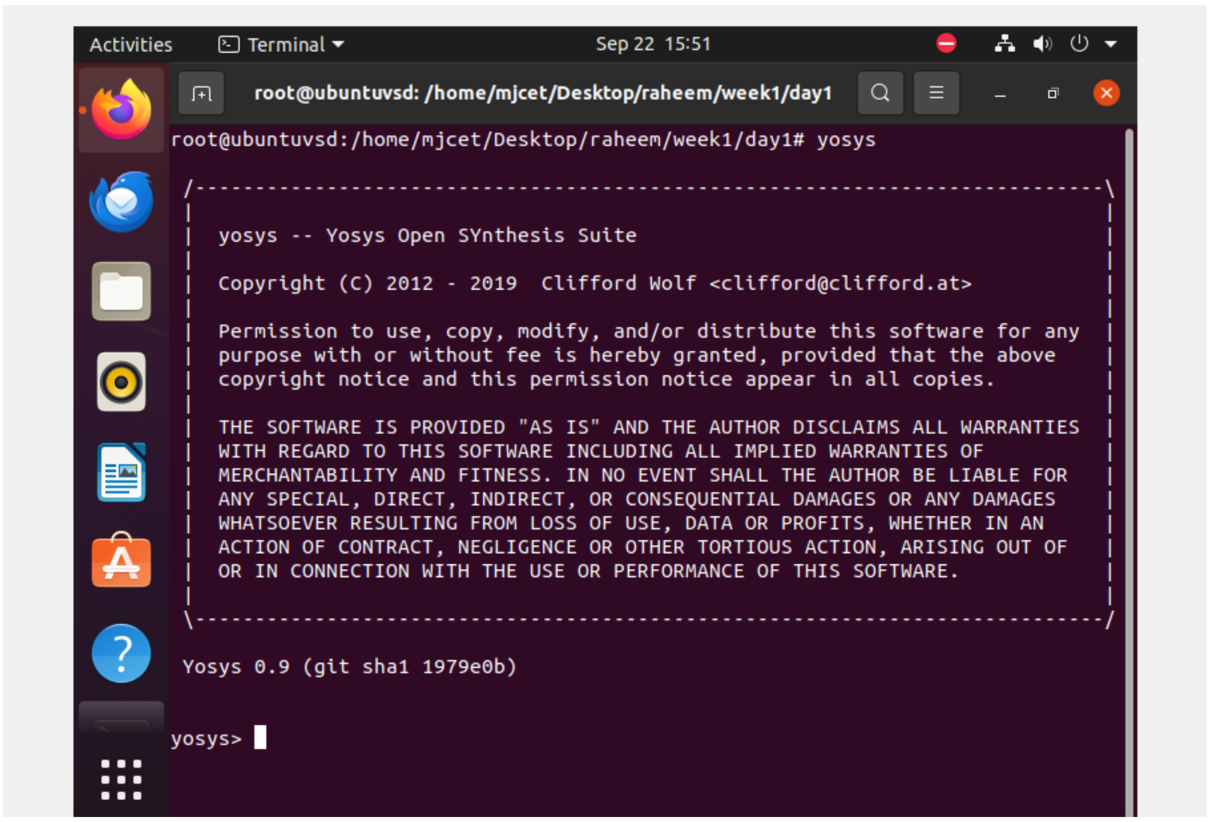
Muffakham Jah College of Engineering and Technology

Banjara Hills, Hyderabad-500 034

Your directory should be something like `.../verilog_files/`, containing both the `lib` and `verilog_files` directories (from a cloned repository). All standard cell libraries reside in `lib`

Open Yosys from the terminal.

```
yosys
```



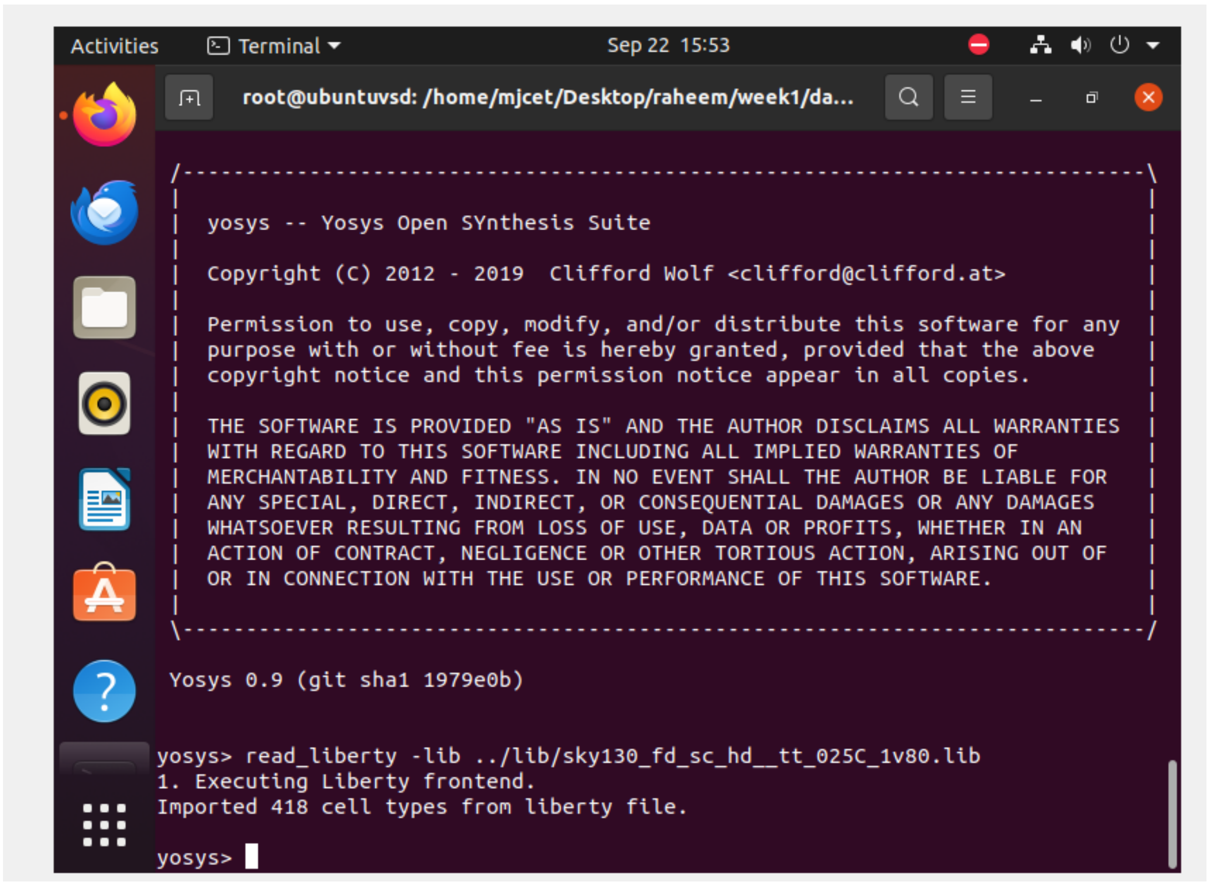
```
root@ubuntuvsd:/home/mjcet/Desktop/raheem/week1/day1# yosys

 /----------------------------------------------------------------------------\
 |                                                                            |
 |  yosys -- Yosys Open SYnthesis Suite                                       |
 |                                                                            |
 |  Copyright (C) 2012 - 2019  Clifford Wolf <clifford@clifford.at>           |
 |                                                                            |
 |  Permission to use, copy, modify, and/or distribute this software for any  |
 |  purpose with or without fee is hereby granted, provided that the above    |
 |  copyright notice and this permission notice appear in all copies.         |
 |                                                                            |
 |  THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES  |
 |  WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF          |
 |  MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR   |
 |  ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES    |
 |  WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN     |
 |  ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF   |
 |  OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.            |
 |                                                                            |
 \----------------------------------------------------------------------------/

 Yosys 0.9 (git sha1 1979e0b)


yosys>
```

Read the standard cell library for synthesis. Paths may be absolute or relative; here we use a relative path:

```
read_liberty -lib ../lib/sky130_fd_sc_hd__tt_025C_1v80.lib
```

Read the standard cell library for synthesis. Paths may be absolute or relative; here we use a relative path:

```
read_liberty -lib ../lib/sky130_fd_sc_hd__tt_025C_1v80.lib
```



```
 /----------------------------------------------------------------------------\
 |                                                                            |
 |  yosys -- Yosys Open SYnthesis Suite                                       |
 |                                                                            |
 |  Copyright (C) 2012 - 2019  Clifford Wolf <clifford@clifford.at>           |
 |                                                                            |
 |  Permission to use, copy, modify, and/or distribute this software for any  |
 |  purpose with or without fee is hereby granted, provided that the above    |
 |  copyright notice and this permission notice appear in all copies.         |
 |                                                                            |
 |  THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES  |
 |  WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF          |
 |  MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR   |
 |  ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES    |
 |  WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN     |
 |  ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF   |
 |  OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.            |
 |                                                                            |
 \----------------------------------------------------------------------------/

 Yosys 0.9 (git sha1 1979e0b)


yosys> read_liberty -lib ../lib/sky130_fd_sc_hd__tt_025C_1v80.lib
1. Executing Liberty frontend.
Imported 418 cell types from liberty file.

yosys> 
```

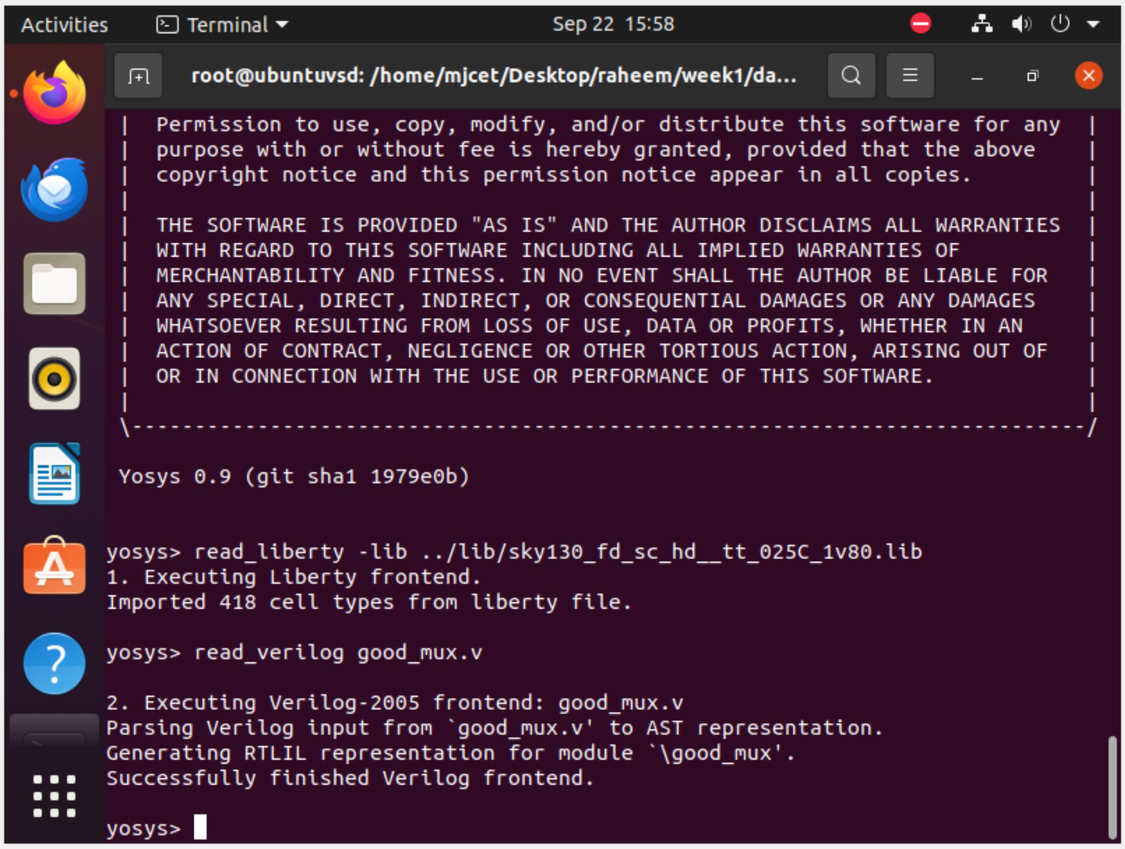
Read your Verilog design (in this case: `good_mux.v`):

```
read_verilog good_mux.v
```

On success, Yosys should report:

```
Successfully finished Verilog frontend.
```

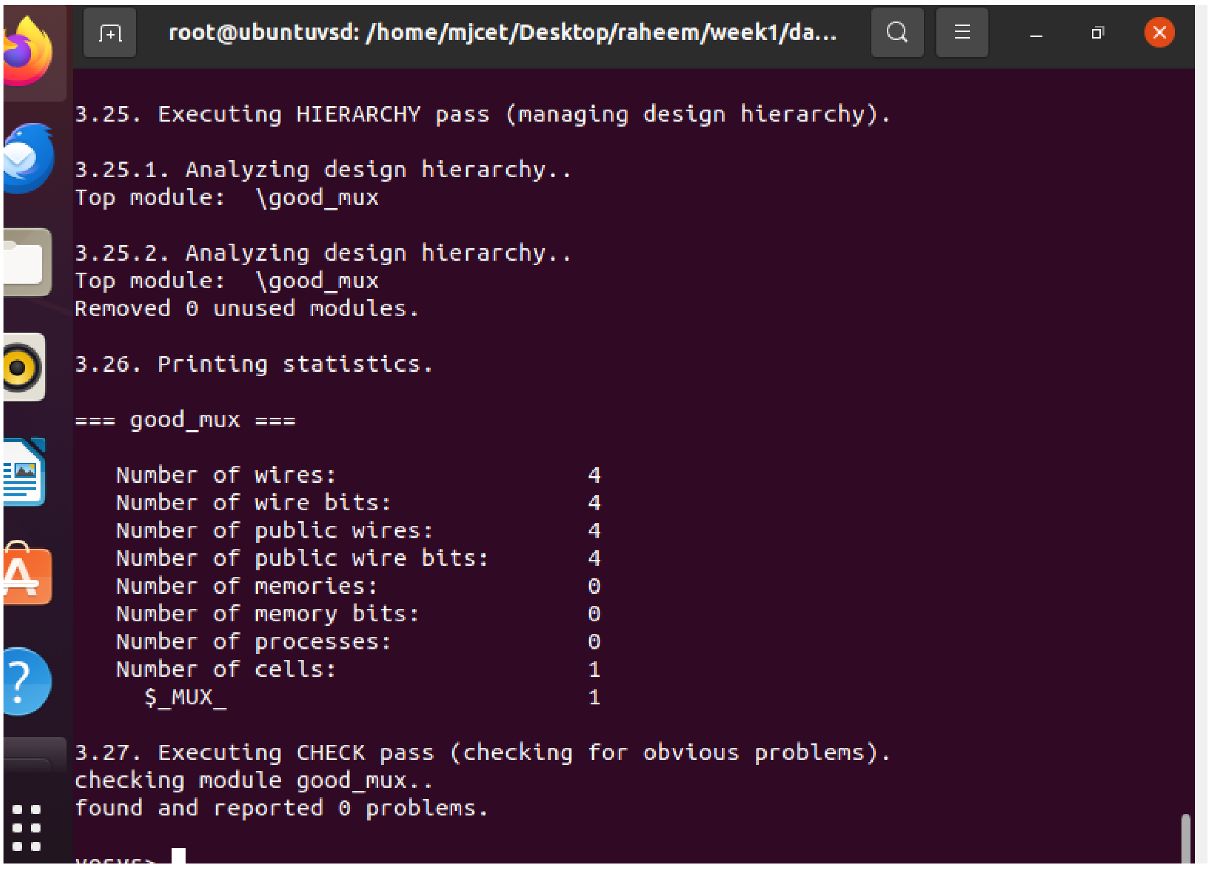
If your design consists of multiple files, repeat the `read_verilog` command as needed for each file.



```
 |  Permission to use, copy, modify, and/or distribute this software for any  |
 |  purpose with or without fee is hereby granted, provided that the above    |
 |  copyright notice and this permission notice appear in all copies.         |
 |                                                                            |
 |  THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES  |
 |  WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF          |
 |  MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR   |
 |  ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES    |
 |  WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN     |
 |  ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF   |
 |  OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.            |
 |                                                                            |
 \----------------------------------------------------------------------------/

 Yosys 0.9 (git sha1 1979e0b)


yosys> read_liberty -lib ../lib/sky130_fd_sc_hd__tt_025C_1v80.lib
1. Executing Liberty frontend.
Imported 418 cell types from liberty file.

yosys> read_verilog good_mux.v

2. Executing Verilog-2005 frontend: good_mux.v
Parsing Verilog input from `good_mux.v' to AST representation.
Generating RTLIL representation for module `\good_mux'.
Successfully finished Verilog frontend.

yosys> 
```

Specify the module to synthesize using the `synth` flow. In our example, the module name is `good_mux`:

```
synth -top good_mux
```

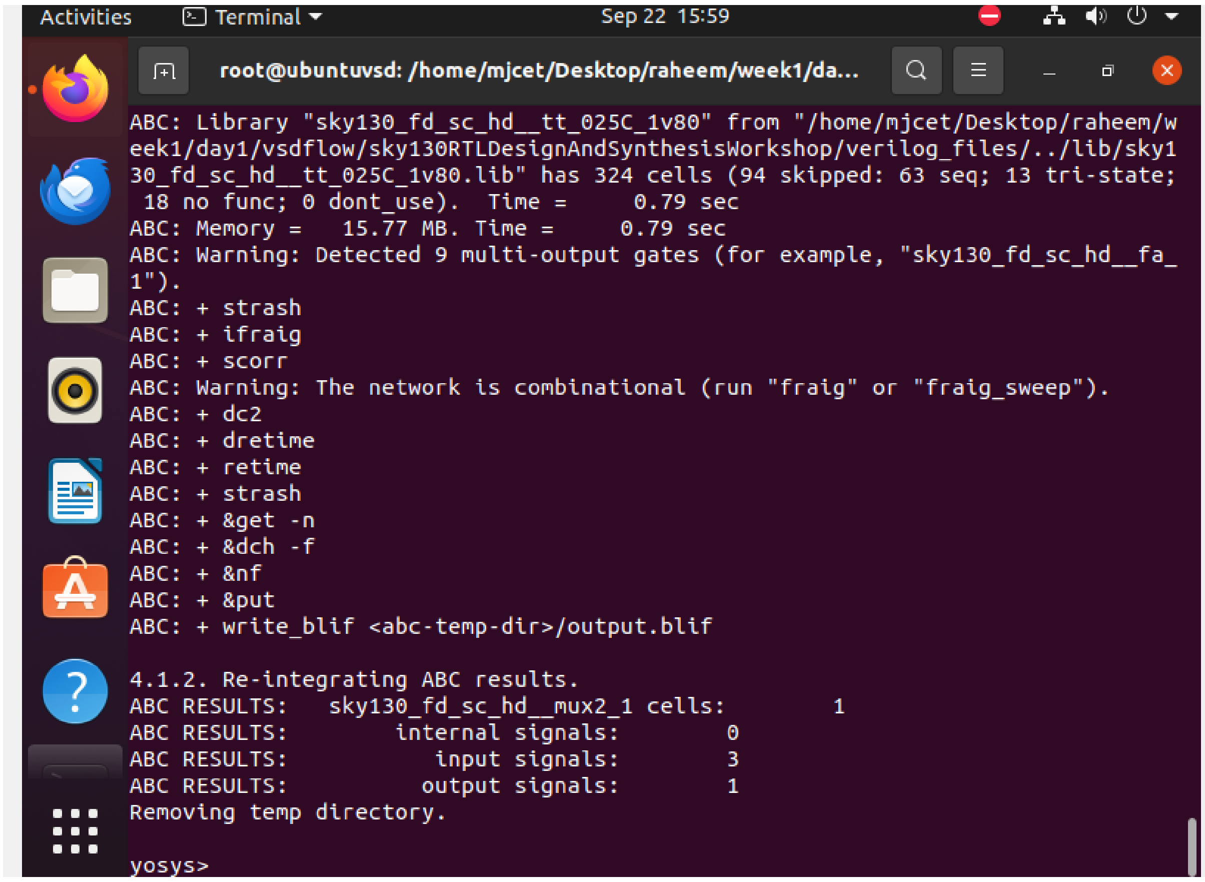


root@ubuntuvsd: /home/mjcet/Desktop/raheem/week1/da...

```
3.25. Executing HIERARCHY pass (managing design hierarchy).

3.25.1. Analyzing design hierarchy..
Top module:  \good_mux

3.25.2. Analyzing design hierarchy..
Top module:  \good_mux
Removed 0 unused modules.

3.26. Printing statistics.

=== good_mux ===

   Number of wires:                  4
   Number of wire bits:              4
   Number of public wires:           4
   Number of public wire bits:       4
   Number of memories:               0
   Number of memory bits:            0
   Number of processes:              0
   Number of cells:                  1
     $_MUX_                          1

3.27. Executing CHECK pass (checking for obvious problems).
checking module good_mux..
found and reported 0 problems.
```

Run technology mapping to your chosen standard cell library:

```
abc -liberty ../lib/sky130_fd_sc_hd__tt_025C_1v80.lib
```



```
ABC: Library "sky130_fd_sc_hd__tt_025C_1v80" from "/home/mjcet/Desktop/raheem/w
eek1/day1/vsdflow/sky130RTLDesignAndSynthesisWorkshop/verilog_files/../lib/sky1
30_fd_sc_hd__tt_025C_1v80.lib" has 324 cells (94 skipped: 63 seq; 13 tri-state;
 18 no func; 0 dont_use).  Time =     0.79 sec
ABC: Memory =    15.77 MB. Time =     0.79 sec
ABC: Warning: Detected 9 multi-output gates (for example, "sky130_fd_sc_hd__fa_
1").
ABC: + strash
ABC: + ifraig
ABC: + scorr
ABC: Warning: The network is combinational (run "fraig" or "fraig_sweep").
ABC: + dc2
ABC: + dretime
ABC: + retime
ABC: + strash
ABC: + &get -n
ABC: + &dch -f
ABC: + &nf
ABC: + &put
ABC: + write_blif <abc-temp-dir>/output.blif

4.1.2. Re-integrating ABC results.
ABC RESULTS:   sky130_fd_sc_hd__mux2_1 cells:        1
ABC RESULTS:          internal signals:        0
ABC RESULTS:             input signals:        3
ABC RESULTS:            output signals:        1
Removing temp directory.

yosys>
```

Yosys will print details of inferred cells and IOs, such as:

- Number of input signals: 3
- Number of output signals: 1
- Number of internal signals: 0

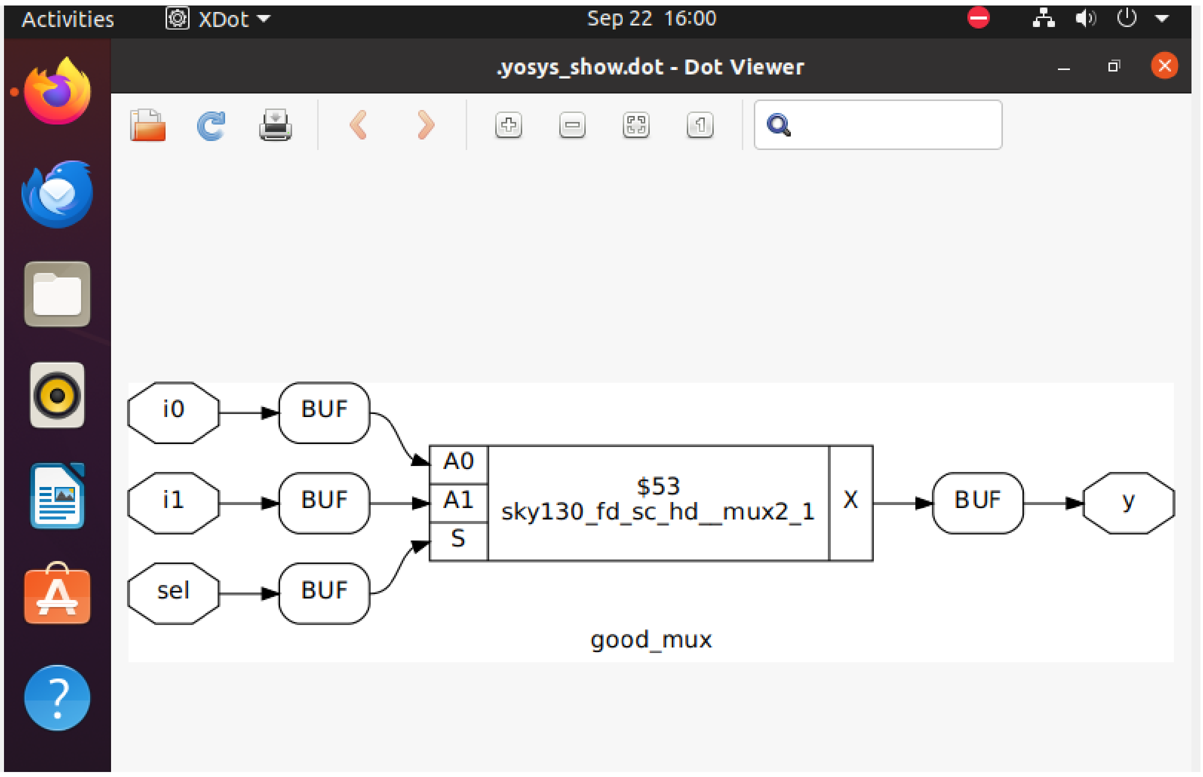
It also lists cell usage, such as:

- 1 clock inverter
- 1 NAND gate
- 1 O2AI gate (OR-AND-Invert) (or)
- 1 mux2_1 (In my case)

Compare these mappings to your original RTL code to ensure expected logic cell usage.

To view the synthesized logic structure:

```
show
```

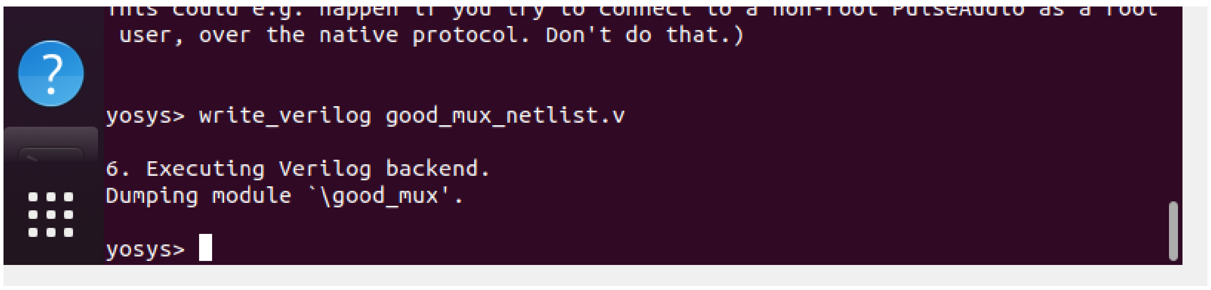


Export the synthesized netlist in Verilog format:

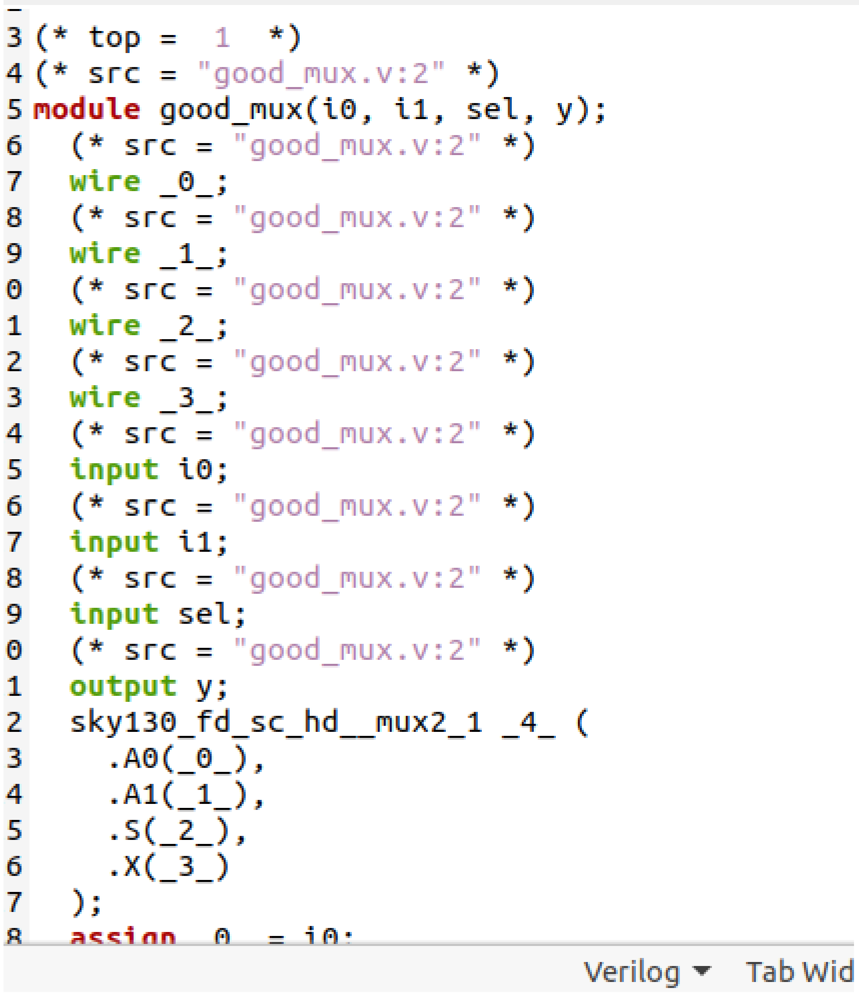
```
write_verilog good_mux_netlist.v
```

T his initial output may contain attributes and extra info. For a cleaner, attribute-free netlist, use:

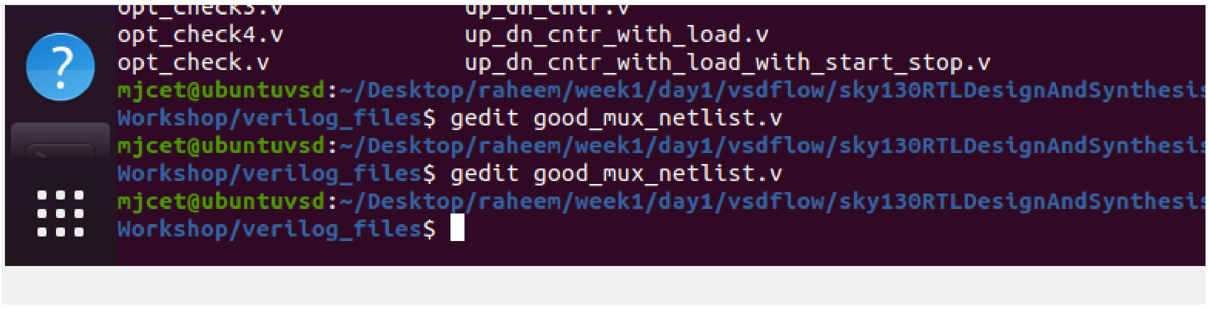
```
write_verilog -noattr good_mux_netlist.v
```



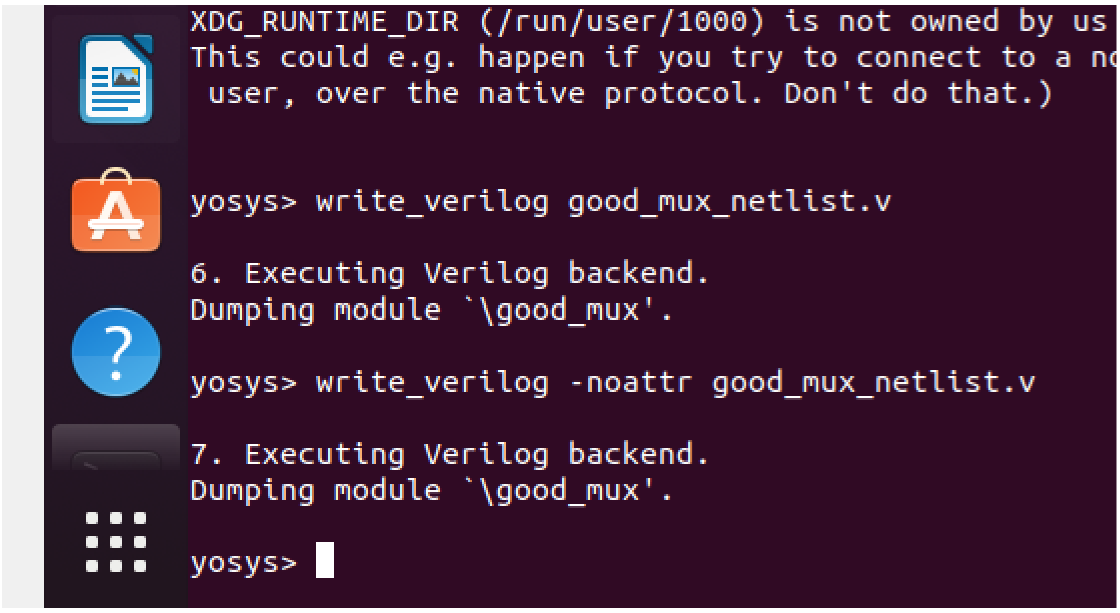
```
This could e.g. happen if you try to connect to a non-root PulseAudio as a root
 user, over the native protocol. Don't do that.)


yosys> write_verilog good_mux_netlist.v

6. Executing Verilog backend.
Dumping module `\good_mux'.

yosys> 
```



```
(* top =  1  *)
(* src = "good_mux.v:2" *)
module good_mux(i0, i1, sel, y);
  (* src = "good_mux.v:2" *)
  wire _0_;
  (* src = "good_mux.v:2" *)
  wire _1_;
  (* src = "good_mux.v:2" *)
  wire _2_;
  (* src = "good_mux.v:2" *)
  wire _3_;
  (* src = "good_mux.v:2" *)
  input i0;
  (* src = "good_mux.v:2" *)
  input i1;
  (* src = "good_mux.v:2" *)
  input sel;
  (* src = "good_mux.v:2" *)
  output y;
  sky130_fd_sc_hd__mux2_1 _4_ (
    .A0(_0_),
    .A1(_1_),
    .S(_2_),
    .X(_3_)
  );
  assign _0_ = i0;
```



```
opt_check3.v                up_dn_cntr.v
opt_check4.v                up_dn_cntr_with_load.v
opt_check.v                 up_dn_cntr_with_load_with_start_stop.v
mjcet@ubuntuvsd:~/Desktop/raheem/week1/day1/vsdflow/sky130RTLDesignAndSynthesis
Workshop/verilog_files$ gedit good_mux_netlist.v
mjcet@ubuntuvsd:~/Desktop/raheem/week1/day1/vsdflow/sky130RTLDesignAndSynthesis
Workshop/verilog_files$ gedit good_mux_netlist.v
mjcet@ubuntuvsd:~/Desktop/raheem/week1/day1/vsdflow/sky130RTLDesignAndSynthesis
Workshop/verilog_files$ 
```



```
XDG_RUNTIME_DIR (/run/user/1000) is not owned by us
This could e.g. happen if you try to connect to a no
 user, over the native protocol. Don't do that.)


yosys> write_verilog good_mux_netlist.v

6. Executing Verilog backend.
Dumping module `\good_mux'.

yosys> write_verilog -noattr good_mux_netlist.v

7. Executing Verilog backend.
Dumping module `\good_mux'.

yosys> 
```

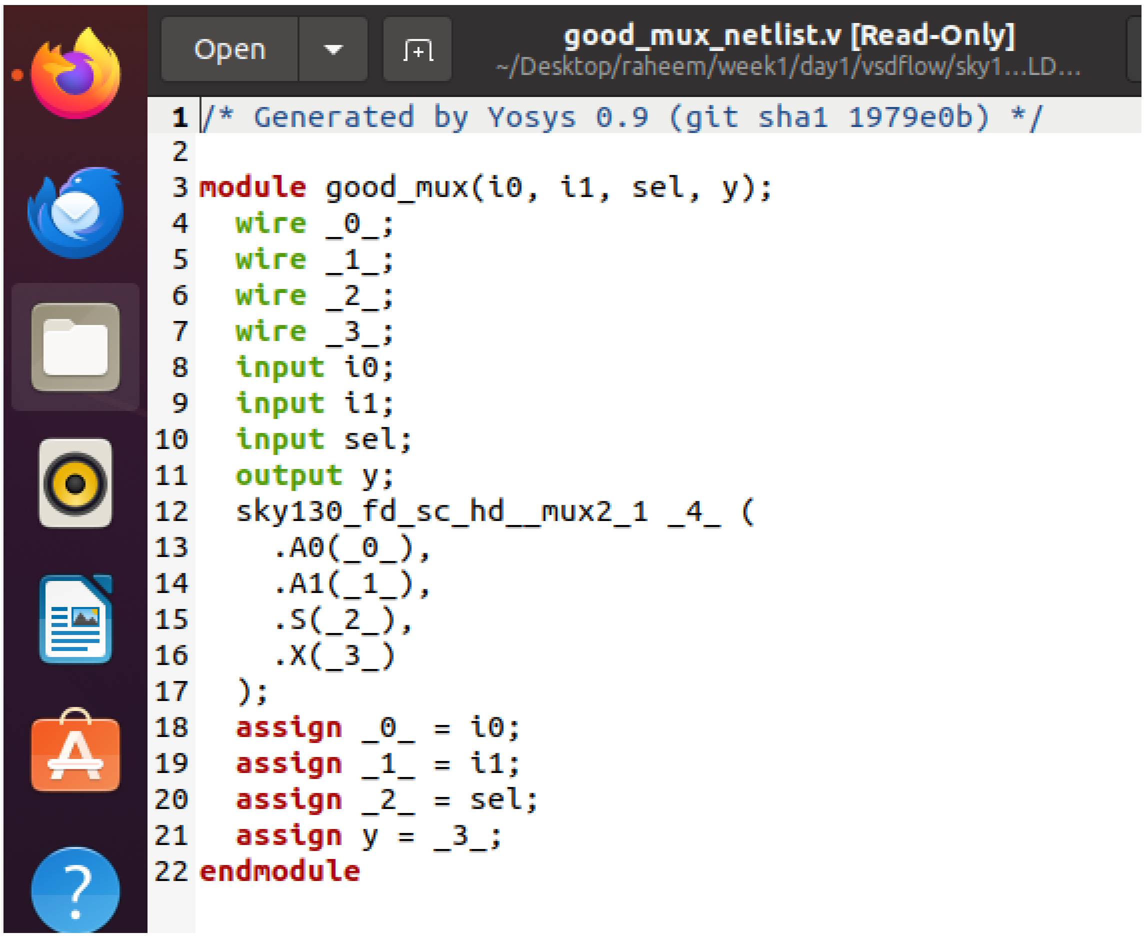
Open and inspect your netlist (example using `gvim`):

```
Gedit good_mux_netlist.v
```

Netlist Structure

- Module name at top matches what was set in `synth -top`.
- Instantiations for each logic gate (inverter, NAND, O2AI).
- Signals (nets) internally connect gate outputs to subsequent gate inputs.
- Primary inputs and outputs clearly mapped to original Verilog I/O.

Trace through each net in your netlist to verify logic connectivity as seen in the schematic.


Open
good_mux_netlist.v [Read-Only]
~/Desktop/raheem/week1/day1/vsdflow/sky1…LD…

```
/* Generated by Yosys 0.9 (git sha1 1979e0b) */

module good_mux(i0, i1, sel, y);
  wire _0_;
  wire _1_;
  wire _2_;
  wire _3_;
  input i0;
  input i1;
  input sel;
  output y;
  sky130_fd_sc_hd__mux2_1 _4_ (
    .A0(_0_),
    .A1(_1_),
    .S(_2_),
    .X(_3_)
  );
  assign _0_ = i0;
  assign _1_ = i1;
  assign _2_ = sel;
  assign y = _3_;
endmodule
```