# INF2270, mandatory exercise 1: Advanced Combinational Logic

P. Häfliger

February 19, 2010

**Abstract**

In this mandatory exercise you will use the professional digital circuit design tool ISE from Xilinx, to implement and test a advanced combinational circuit.

## 1 The Task

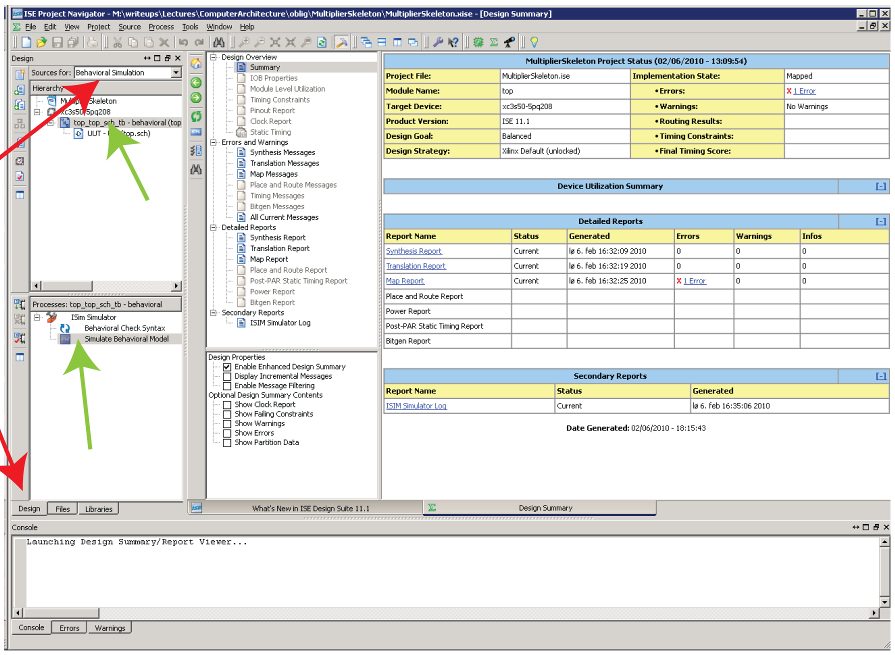
You shall design a *combinational* circuit that multiplies an input number by 10. The input will be represented in two's complement and is 5 bits long (i.e. -16 to 15). The minimal and maximal results are thus 150 and -160. This requires a 9 bit two's complement representation of the result. Assign this result to the output signal os(8:0).

Furthermore the result shall be made ready to be displayed on an hexadecimal LED display as a sign and magnitude hexadecimal number in two steps: convert the result into binary 'sign and magnitude' representation and then produce the correct control signal for the 7-segment LED displays. The sign and magnitude result will also be 9 bits, the MSB for the sign and 8 bits for the magnitude. (Assign it to the output signal o(8:0).) Thus, two LED 7-segment displays for the magnitude and one to display the minus sign will be required.

Generate the correct control signals for these hexadecimal displays (see figure 3 and assign them to the output signals LED1 and LED2. Do not worry about the sign bit, which is simply the output bit o(8) For example, if your sign and magnitude result is 111010010 (-209) the hexadecimal display should show -D2. The bit pattern to display a 'D' (actually displayed as a lower case 'd') is '1011110', the pattern for '2' is '1011011'.

You may verify the correctness of your bit patterns in the simulation display by setting the radix of the output signal o(8:0) to 'hexadecimal' and the signals disp1 and disp2 to 'ASCII'. If your signals LED1(6:0) and LED2(6:0) are correct disp1 and disp2 will display the same characters as the hexadecimal disply of o(8:0).

A hint to the multiplication: $10x = 2x + 8x$ or $2(x + 4x)$



**MultiplierSkeleton Project Status (02/06/2010 - 13:09:54)**

| | | | |
|---|---|---|---|
| **Project File:** | MultiplierSkeleton.ise | **Implementation State:** | Mapped |
| **Module Name:** | top | **• Errors:** | X 1 Error |
| **Target Device:** | xc3s50-5pq208 | **• Warnings:** | No Warnings |
| **Product Version:** | ISE 11.1 | **• Routing Results:** | |
| **Design Goal:** | Balanced | **• Timing Constraints:** | |
| **Design Strategy:** | Xilinx Default (unlocked) | **• Final Timing Score:** | |

**Device Utilization Summary** [-]

**Detailed Reports** [-]

| Report Name | Status | Generated | Errors | Warnings | Infos |
|---|---|---|---|---|---|
| Synthesis Report | Current | lø 6. feb 16:32:09 2010 | 0 | 0 | 0 |
| Translation Report | Current | lø 6. feb 16:32:19 2010 | 0 | 0 | 0 |
| Map Report | Current | lø 6. feb 16:32:25 2010 | X 1 Error | 0 | 0 |
| Place and Route Report | | | | | |
| Power Report | | | | | |
| Post-PAR Static Timing Report | | | | | |
| Bitgen Report | | | | | |

**Secondary Reports** [-]

| Report Name | Status | Generated |
|---|---|---|
| ISIM Simulator Log | Current | lø 6. feb 16:35:06 2010 |

**Date Generated:** 02/06/2010 - 18:15:43

```
Launching Design Summary/Report Viewer...
```

Figure 1: Screenshot of the ISE project navigator

## 2 The Tool

ISE WebPack, the free lite version of the ISE Design Suite, can be started on the windows servers dixon.ifi.uio.no and wise.ifi.uio.no. You can also download it from ttp://www.xilinx.com/tools/webpack.tm for Windows or Linux (only 32-bit, it seems!), once you have registered a (free) user account.

But before starting the tool you will need to copy a 'skeleton' ISE project that contains all necessary configurations and a basic schematic that defines all inputs and outputs that are required plus a simulation setup that will supply test-inputs for your system. Copy the directory 'MultiplierSkeleton' including all subdirectories from

```
 ~inf2270/programmer/MultiplierSkeleton
```

into a folder on your home directory, under Windows simply by copy-paste after having mapped a net drive to

```
\\hjemme.uio.no\inf2270
```

, under Linux by

```
cp -R ~inf2270/programmer/MultiplierSkeleton <target directory>
```

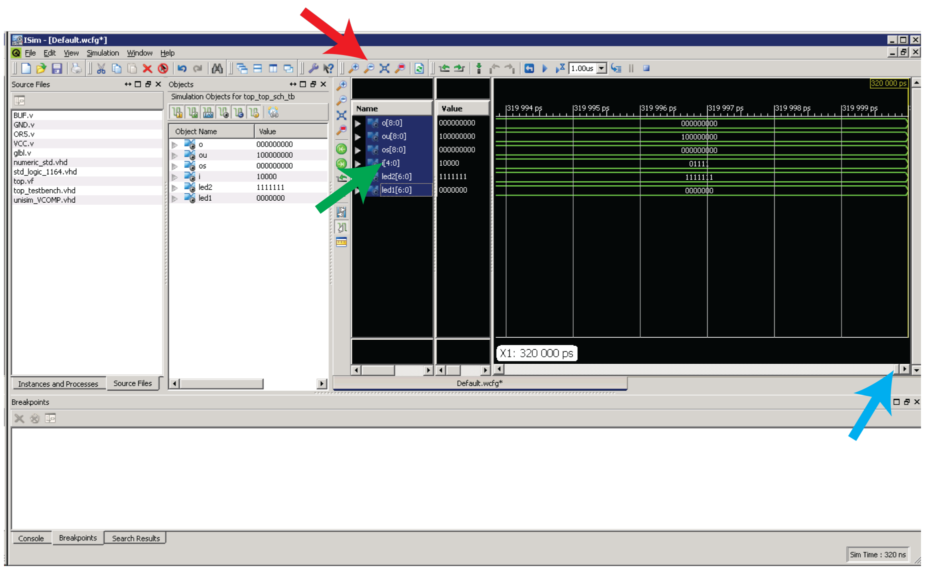


Figure 2: Screenshot of the ISim simulator

You can open a remote desktop on those servers from any windows PC in the 'termstuene' by choosing:

```
Start->All programs->Accessories->Remote desktop connection
```

and then logging onto one of the two servers.

From Linux, type into an xterm:

```
rdesktop -f dixon.ifi.uio.no
```

or

```
rdesktop -f wise.ifi.uio.no
```

Once you are logged in you can start ISE under:

```
Start->All programs->Xilinx ISE Design Suite 11->ISE->Project navigator
```

and then open the project file 'MultiplierSkeleton.xise'.

You should see the screen in figure 1. Make sure that the 'design panel' (lower red arrow) is active and the 'behavioral simulation' is chosen (upper red arrow). If you click on the file 'top_top_sch-behavioral(top_testbench.vhd)' indicated by the upper green arrow, the link 'simulate behavioral model' indicated by the lower green arrow should appear.
Double-click it!

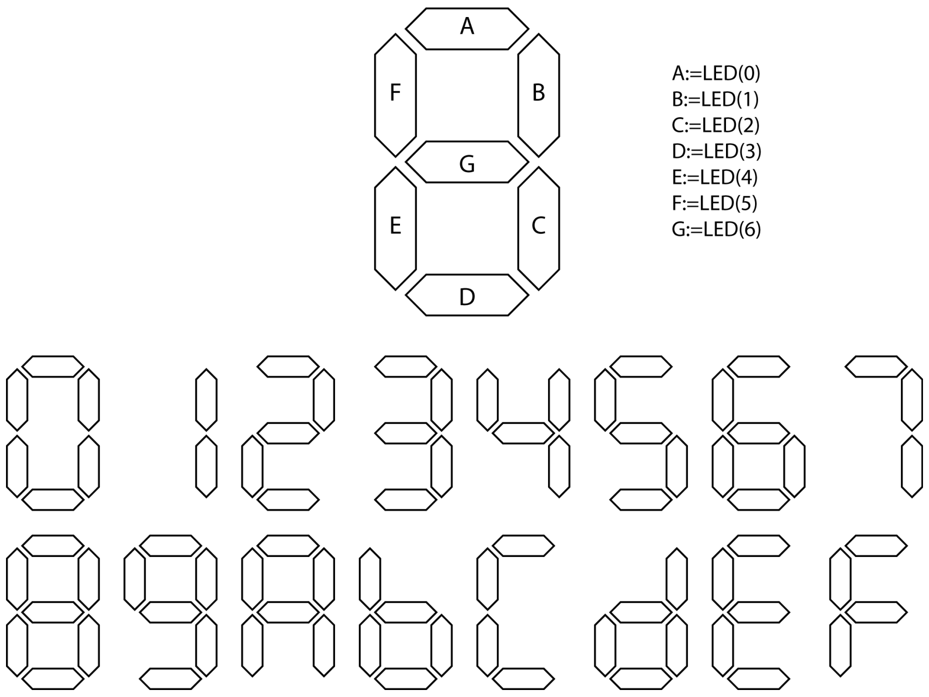
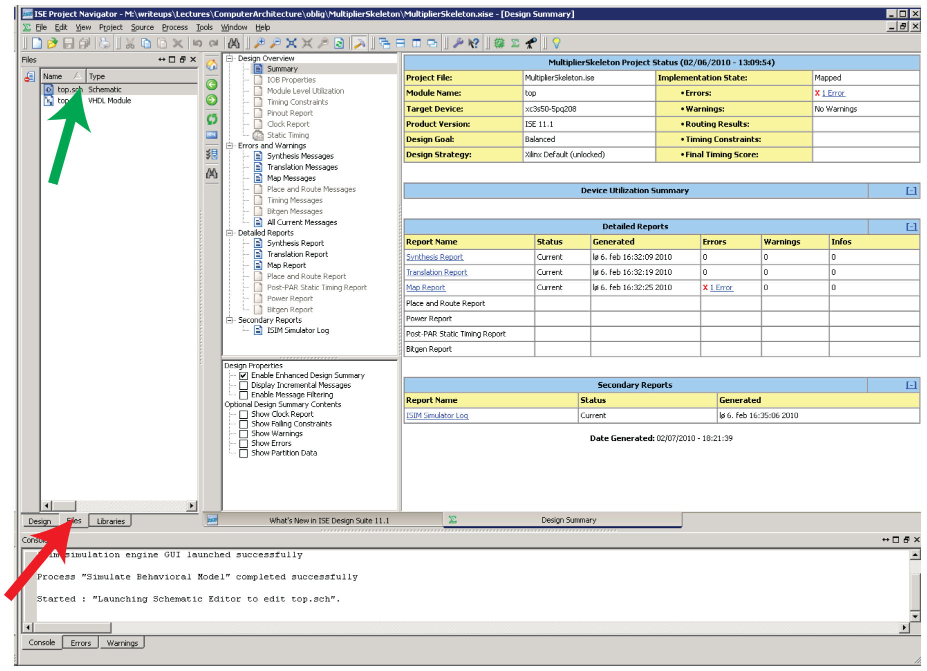


Figure 3: 7 segment LED display and how it is used to display the hexadecimal digits

The ISim simulator opens in another window as displayed in figure 2. It shows a number of signal traces. If you zoom out a couple of times (red arrow), you will discern that one signal has a few signal transitions, the signal named 'i(4:0)' which will be the test input for your circuit. If you select its label (green arrow) and then right click on it there appears a pull down menu. From this menu choose 'radix->signed decimal'. By using the slider you can now look at this signal and will note that it counts from -16 up to 15, i.e. all possible inputs. The other signals will be your outputs but they do not yet work correctly of course. os(8:0) will be your two's complement result of the input multiplied by 10. ou(8:0) will be the inverse two's complement. For the magnitude and sign output o(8:0) you will either choose os(8:0) (if the result is positive) or os(8),ou(7:0) (if the result is negative). Alternatively you might want to convert the input to 'sign and magnitude' before multiplying, in which case you would actually only need ou(4:0) and os(4:0). In short: you may choose to display any intermediate result through ou and os that you find useful. The signals LED1 and LED2 will be the outputs to the 7-segment LED displays for the hexadecimal display. LED1(0) corresponds to the LED 'A' in figure 3 and LED1(6) to 'G'. There are two more signals displayed, disp1 and disp2. If you set their radix to 'ASCII' they will display the hexadecimal digit that



MultiplierSkeleton Project Status (02/06/2010 - 13:09:54)

| | | | |
|---|---|---|---|
| **Project File:** | MultiplierSkeleton.ise | **Implementation State:** | Mapped |
| **Module Name:** | top | **• Errors:** | X 1 Error |
| **Target Device:** | xc3s50-5pq208 | **• Warnings:** | No Warnings |
| **Product Version:** | ISE 11.1 | **• Routing Results:** | |
| **Design Goal:** | Balanced | **• Timing Constraints:** | |
| **Design Strategy:** | Xilinx Default (unlocked) | **• Final Timing Score:** | |

Device Utilization Summary [-]

Detailed Reports [-]

| Report Name | Status | Generated | Errors | Warnings | Infos |
|---|---|---|---|---|---|
| Synthesis Report | Current | lø 6. feb 16:32:09 2010 | 0 | 0 | 0 |
| Translation Report | Current | lø 6. feb 16:32:19 2010 | 0 | 0 | 0 |
| Map Report | Current | lø 6. feb 16:32:25 2010 | X 1 Error | 0 | 0 |
| Place and Route Report | | | | | |
| Power Report | | | | | |
| Post-PAR Static Timing Report | | | | | |
| Bitgen Report | | | | | |

Secondary Reports [-]

| Report Name | Status | Generated |
|---|---|---|
| ISIM Simulator Log | Current | lø 6. feb 16:35:06 2010 |

**Date Generated:** 02/07/2010 - 18:21:39

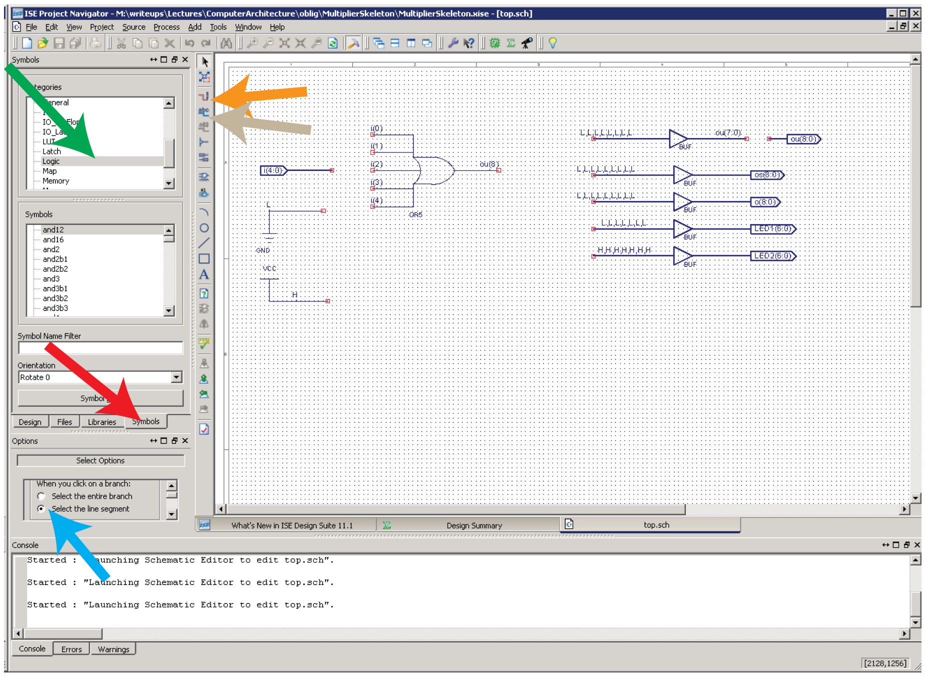
```
ISim simulation engine GUI launched successfully

Process "Simulate Behavioral Model" completed successfully

Started : "Launching Schematic Editor to edit top.sch".
```

Figure 4: Screenshot of the ISE project navigator

corresponds to the 7-segment bit patterns of your signals LED1 and LED2 *if the bit patterns are correct*. If you put out a bit pattern that is not recognized ar all the disp-signals will be X.

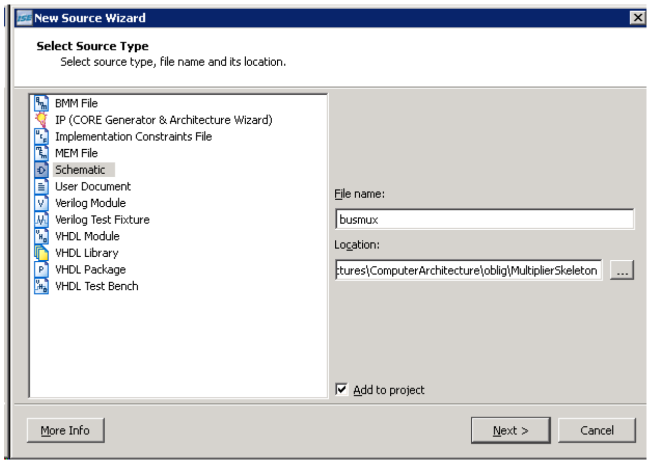
Close the ISim window for now. You will be asked to save your waveform file, but we will not do so: choose 'no'.

Back on the ISE project navigator window we will now have a look at the schematic editor in which you will draw your circuits. Click on the flag 'files' indicated by the red arrow (figure 4)and the left column of the window will change as displayed in figure 4. If you double-click on the file 'top.sch', a schematic editor will open with the top schematic cell as displayed in figure 5. The 'symbol' flag will have been chosen automatically (red arrow), otherwise, do so. From the 'categories' list choose 'logic' (green arrow). Note that for this exercise you will only be allowed to use the symbols from this category, the category 'general' and the simple buffers already used here from the category buffers! From the symbols list below you can now select and drop a number of different logic gates into your design. The vertical toolbar allows you for example to draw wires and buses (orange arrow) and to name wires and buses (gray arrow: the name has to be entered in the window where the blue arrow points to now before you click on a wire. Its title will change to 'add net name options'. Scroll down to the 'Name' field and write there.). The blue arrow indicates another selection that I



```
Started : "Launching Schematic Editor to edit top.sch".

Started : "Launching Schematic Editor to edit top.sch".

Started : "Launching Schematic Editor to edit top.sch".
```

Figure 5: Screenshot of the ISE project navigator



**New Source Wizard**

**Select Source Type**
Select source type, file name and its location.

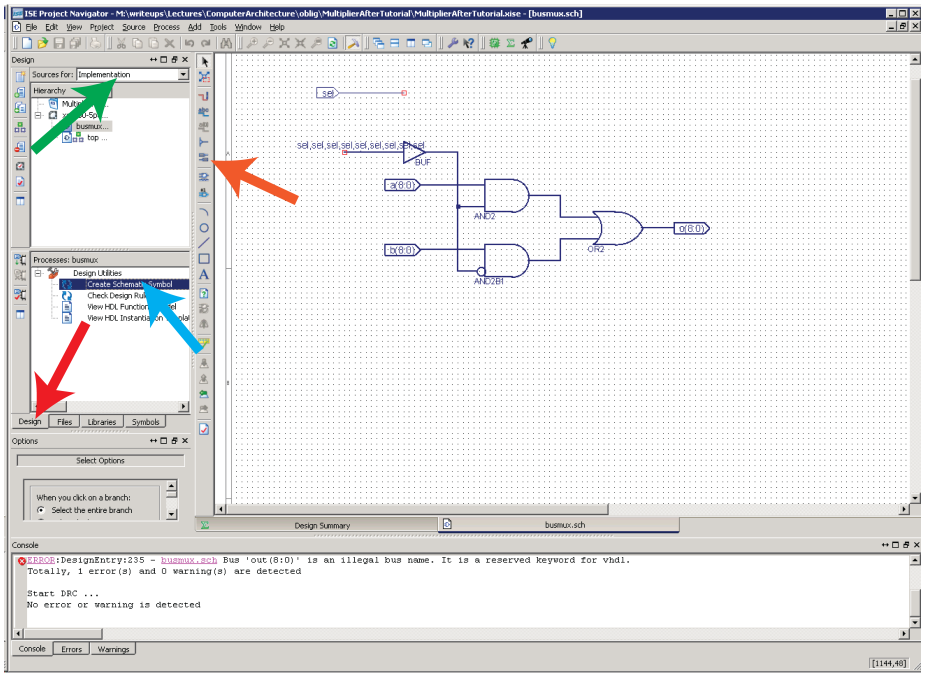
- BMM File
- IP (CORE Generator & Architecture Wizard)
- Implementation Constraints File
- MEM File
- Schematic
- User Document
- Verilog Module
- Verilog Test Fixture
- VHDL Module
- VHDL Library
- VHDL Package
- VHDL Test Bench

File name: busmux

Location: tures\ComputerArchitecture\oblig\MultiplierSkeleton

[✓] Add to project

Figure 6: Screenshot of the 'new source' pop-up


ISE Project Navigator - M:\writeups\Lectures\ComputerArchitecture\oblig\MultiplierAfterTutorial\MultiplierAfterTutorial.xise - [busmux.sch]
Sources for: Implementation
Hierarchy
Processes: busmux
Design Utilities
Create Schematic Symbol
Check Design Rules
View HDL Functional Model
View HDL Instantiation Template
Design
Files
Libraries
Symbols
Options
Select Options
When you click on a branch:
Select the entire branch
sel
sel,sel,sel,sel,sel,sel,sel,sel,sel
BUF
a(8:0)
b(8:0)
AND2
AND2B1
OR2
o(8:0)
Design Summary
busmux.sch
Console
Errors
Warnings
[1144,48]

```
ERROR:DesignEntry:235 - busmux.sch Bus 'out(8:0)' is an illegal bus name. It is a reserved keyword for vhdl.
Totally, 1 error(s) and 0 warning(s) are detected

Start DRC ...
No error or warning is detected
```

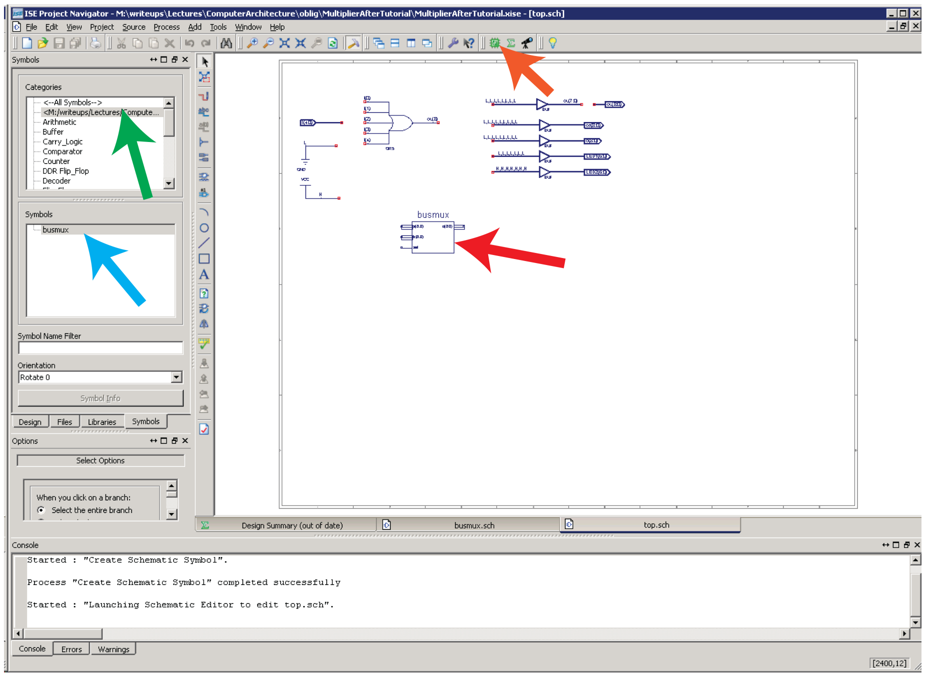
Figure 7: Screenshot of the ISE project navigator

find quite useful: by selecting 'select the line segment' you will be able to select wire segments only instead of the whole electric node. This is especially useful if you made a wrong connection and want to delete only your mistake and not the entire net.

The schematics now is quite basic. For illustration, a 5-input or-gate has been placed to receive all the input bits and the output of the or-gate is connected to output ou(8). Note how connections can be made by simply naming buses. So bit ou(8) can be merged onto the bus ou(8:0) simply by calling it that. Another example that you might find useful: if you name a 9 wire bus "i(4:0),L,L,L,L" then that is equivalent to a 4 bit left shift of the input.

The signals 'H' and 'L' are connected to vcc and gnd respectively and are thus the constant signals '1' and '0' ('H' for high and 'L' for low). The buffer cells are needed if you want to connect nodes that already have a name to another node with a different name, e.g. the constants 'L' and 'H' to an output port.

Symbols too can be representative for a whole bus width of the same symbol: if you right click on one of the buffers and select 'object properties' from the pop-up menu, you will see that its instance name is something like XLXI_66(7:0). This means that there are 8 buffers in parallel, each connected to one wire of the bus. You can do the same with any logic gate: just right-click on it and



```
Started : "Create Schematic Symbol".

Process "Create Schematic Symbol" completed successfully

Started : "Launching Schematic Editor to edit top.sch".
```

Figure 8: Screenshot of the ISE project navigator

edit the instance name such that it ends in ...(7:0) and you'll have 8 parallel copies of that gate to which you can connect bus-lines. The symbol will also be displayed with thick lines to indicate the fact that it is to be connected to buses.

For tidiness you will want to define subcircuits. To create a subcircuit, chose from the top menu: project->new source... A pop-up will appear as shown in figure 6. Chose the type to be schematic and enter a name for your subcircuit. For illustration we will construct a bus multiplexer for 9 bits bus width (9 1-bit multiplexers in parallel that can route one of two buses). Press 'next' and then 'finish'. A new empty schematic window will open. Draw the circuit as shown in figure 7. To add I/O pins use the tool indicated by the orange arrow. You may change the name of those pins by double-clicking on the pins or right-clicking and choosing 'object properties'. In the pop-up choose 'Nets' and edit the name. The logic gates have also been expanded to serve a bus by right-clicking on them, choosing 'object properties' and editing the instance name to end in ...(8:0).

Now you must create a symbol of this cell that you can use in your top schematic cell. For this, choose 'Implementation' in the drop down list indicated by the green arrow (figure 7) and the 'design' flag (red arrow). Select the file 'busmux.sch' in the list below and double click on 'create schematic symbol (blue arrow) and a symbol is created automatically.

If you now open your 'top.sch' cell again and choose the 'symbols' flag, there should be a new symbol category: the path at which your new symbol has been stored (figure 8, green arrow). (If it does not appear immediately, as has happened in the demo in the lecture, restart the ISE project navigator. This seemed to help.) Select it, select the only symbol therein (blue arrow) and place it in your top design (red arrow). If you do not like the automatically generated symbol, you may edit it by right-clicking on it and choosing 'symbol->edit symbol'.

If you have edited your circuits, think that they should work and want to run the simulation again, hit the button indicated by the orange arrow in figure 8 while in 'implementation' mode (chose the 'design' flag and 'implementation' from the drop down menu). This will compile your circuit for simulation. You might get error messages that you have to correct. One last error message concerning the 'mapping' of the schematic is expected, though and you need not heed it. After successful compilation, choose the 'design' flag and the 'behavioral simulation' from the drop down menu. Chose the file 'top_top_sch-behavioral(top_testbench.vhd)' as before and double click on 'simulate behavioral model'. Always close the ISim window before running another simulation as you will get an error otherwise.

## 3 Deliverables

Submit a report in which you include plots of your circuit, all subcircuits and the signal trace from the simulation. Make multiple plots of the simulation traces (e.g. from 0ns to 160ns and from 160ns to 320ns) such that the entire sequence can be checked. Choose 'o' to be displayed as hexadecimal number, which should be easiest to decipher (the topmost bit will be the sign), and LED1 and LED2 as binary. Document and describe your specific approach and any design choices you may make in the process. Explain what the other signals ('ou','os') are showing, as there are several possibilities as to how you may use them.

Good luck!