Coursework 2: Sequential Circuit Design using Quartus

This exercise contributes 40% of the continuous assessment of the hardware course. It is scheduled during the first two weeks of the spring term, but can be done earlier if desired. Check on CATE for the hand-in date which is normally the monday of week 3.

Introduction

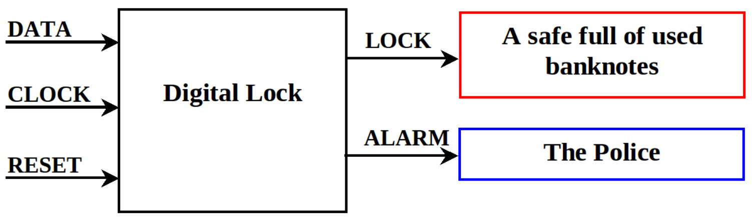
This hardware laboratory exercise is designed to introduce you to a modern computer aided design (CAD) tool for hardware called Quartus II. The Quartus II software is available on the Windows machines in the laboratory. In addition, the equivalent Quartus II is available for free download from:

`http://www.altera.com`
or direct downloads Nov 2018:
`http://download.altera.com/software/acds/11.1sp2/259/standalone/11.1sp2_259_quartus_free_windows.exe`
`http://download.altera.com/software/acds/11.1sp2/259/standalone/11.1sp2_259_quartus_free_linux.sh`

You can set it up on your own machine if you wish. You will need to install the QuartusII web edition and ModelSim starter edition. The exercise was developed using version 11.1, so download that rather than the latest version.
The Quartus II interface can take a little getting used to so make sure you check it out during the first week.

The Design Problem

Design a three-input, two-output sequential digital circuit which functions as a digital locking mechanism.



If a logic level 0 appears on the LOCK output then the safe is unlocked. If a logic level 1 appears on the ALARM line then the police are summoned. When a logic 1 appears on the RESET line, the mechanism is put into the known "reset" state in which the LOCK output is set to logic level 1 (locked). In this state the lock mechanism is waiting for a specific five binary digit sequence of 1s and 0s to appear, one binary digit for each CLOCK pulse. If the correct sequence of digits is sent then the LOCK output becomes logic 0 and the safe opens. The safe remains open as long as the DATA input line is at logic 1. When logic 0 appears on the DATA input the system returns to the reset state. If a wrong sequence of digits is input then the ALARM output signal becomes a logic 1 and it remains so for any sequence of values appearing at the DATA input. The only way to remove the Alarm signal is to set the RESET input line to 1.

Your design should use a personalised sequence number. On the web there is a list of students with a unique number allocated to each. This same number was used for the coursework in term 1. Your individual binary sequence number can be generated using the formula:

Binary Sequence = Binary Equivalent of (1 + (MyNumber mod 30))

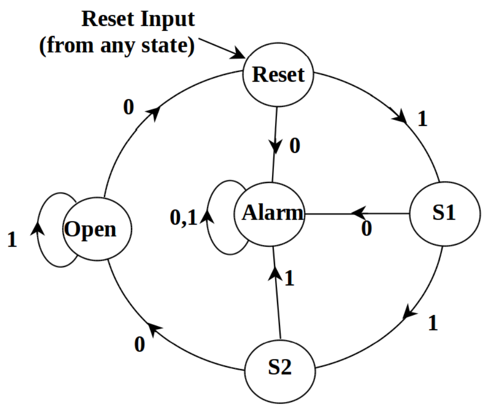
For example, if my unique number is 87, I get (1 + (87 mod 30)) = 1+27 = 28 = 11100 or, if my unique number is 8, I get (1 + (8 mod 30)) = 1+8 = 9 = 01001. The sequence is to be input with the most significant bit coming first.

Digital Design

The design procedure will be demonstrated for a three-digit sequence, say 110 (remember that your design problem has five digits!).

The State Diagram

The operation of the digital lock mechanism can be described by the following Moore state diagram.



Note that a 1 on the RESET input takes the finite state machine to the reset state from any state. We will handle this input separately from the DATA input by using either a preset or clear function on the D-Q flip flops.

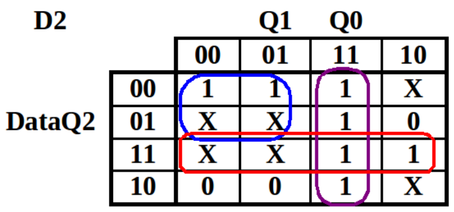
The State Transition Table

From the State Diagram a State Transition Table can be generated. There are five states; therefore, we need a minimum of three flip-flops whose outputs are Q2, Q1, Q0. We have a free choice to assign the five required states to the eight possible ones. The complexity of the final circuit will depend on these assignments but not in an obvious way. We choose the following ones:

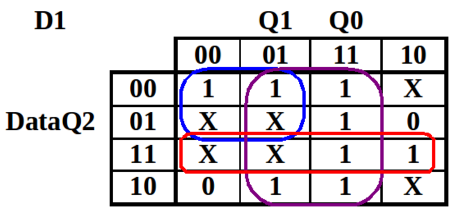
DATA	This State	Q2	Q1	Q0	Next State	D2	D1	D0
0	Reset	0	0	0	Alarm	1	1	1
0	S1	0	0	1	Alarm	1	1	1
0		0	1	0	×	×	×	×
0	S2	0	1	1	Open	1	1	0
0		1	0	0	×	×	×	×
0		1	0	1	×	×	×	×
0	Open	1	1	0	Reset	0	0	0
0	Alarm	1	1	1	Alarm	1	1	1
1	Reset	0	0	0	S1	0	0	1
1	S1	0	0	1	S2	0	1	1
1		0	1	0	×	×	×	×
1	S2	0	1	1	Alarm	1	1	1
1		1	0	0	×	×	×	×
1		1	0	1	×	×	×	×
1	Open	1	1	0	Open	1	1	0
1	Alarm	1	1	1	Alarm	1	1	1

The Karnaugh Maps

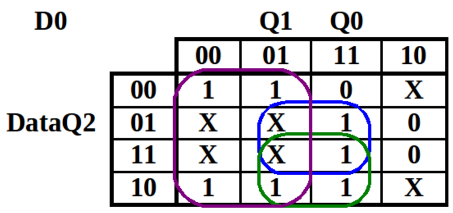
From the flip-flop output table the three design Karnaugh maps can be generated:



	00	01	11	10
00	1	1	1	X
01	X	X	1	0
11	X	X	1	1
10	0	0	1	X



	00	01	11	10
00	1	1	1	X
01	X	X	1	0
11	X	X	1	1
10	0	1	1	X



	00	01	11	10
00	1	1	0	X
01	X	X	1	0
11	X	X	1	0
10	1	1	1	X

$$\begin{aligned} D2 &= DATA' \cdot Q1' + DATA \cdot Q2 + Q1 \cdot Q0 \\ D1 &= Q0 + DATA' \cdot Q1' + DATA \cdot Q2 \\ D0 &= Q1' + Q2 \cdot Q0 + DATA \cdot Q0 \end{aligned}$$

Further factorisations and simplifications may be possible at this stage.

Output Logic Design

You must ensure that the LOCK output signal becomes 0 only when the system is in the Open state, and the safe is to be unlocked. Similarly, the ALARM output should only be 1 when the system is in the Alarm state.

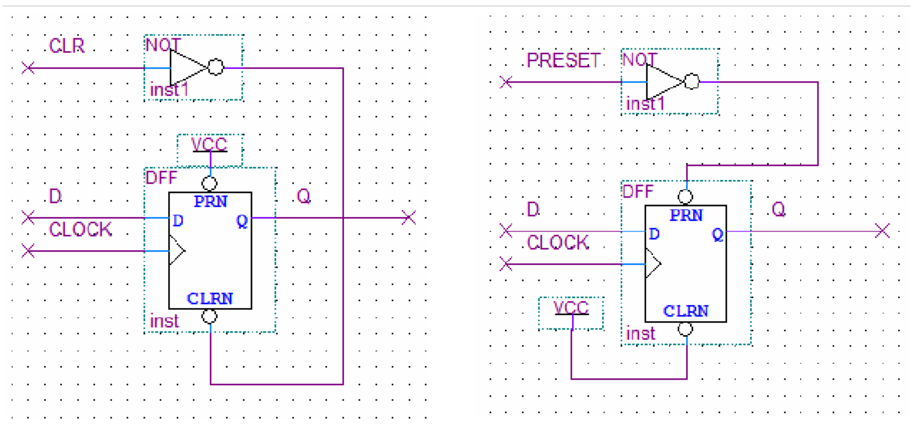
Circuit Design

You are allowed to use the following gate types which are a small subset of those supported by Quartus II.

Name	Function	Area	Name	Function	Area
NOT	Inverter	15	OR2	2 Input OR	28
AND2	2 Input AND	27	OR3	3 Input OR	34
AND3	3 Input AND	34	XNOR2	2 Input XNOR	44
AND4	4 Input AND	40	XOR2	2 Input XOR	44
NAND2	2 Input NAND	20	DFF	D Flip-Flop	100
NAND3	3 Input NAND	28	VCC	Logic 1	0
NAND4	4 Input NAND	32	GND	Logic 0	0
NOR2	2 Input NOR	19	INPUT	Input pin	0
NOR3	3 Input NOR	28	OUTPUT	Output pin	0

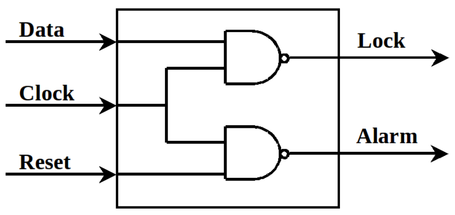
The RESET input to the lock mechanism can be achieved by using the CLEAR or PRESET input for the D-type flip-flops FDC or FDP. Depending on the state (flip-flop outputs) assigned to the RESET state, you may choose to use flip-flops with CLEAR, or PRESET functions only. The flip-flops are positive edge triggered; i.e they latch their inputs on the positive rising edge of the clock pulse. Minimisation here is for silicon area; therefore you are given the approximate area for each gate in the table above. Try to minimise this area as much as possible within your design constraints.

Hint: The DFF (D-Type flip flop) has an asynchronous clear and preset. The two circuits below show how to use these to set the flip flop output to either 0 (CLR) or 1 (PRESET).



Schematic Construction and Testing

We suggest that you implement the simple circuit shown below at first. This is just for practice using the Quartus II tools. It doesn't solve the problem, but you can easily tell whether you have a correct implementation. Go through the schematic construction and simulation steps for this sample circuit, so you get the mechanics out of the way. Decide whether the results make sense.



Getting Started With Altera Quartus II

Quartus II has been installed on the Windows operating system in the labs. You will need to find the location of the two applications "Quartus II web edition" and "ModelSim-Altera". These may be in separate directories called Altera, and could be in the Lab or Programming directories. Use the search facility in the Windows program menu to find them.

Setting up your Machine

1. Ensure that your fileserver space is mounted as the H: drive. Although the main work is usually done in Windows you will need to use Linux for the final submission.

2. Create a directory in your home space for your designs (eg use H:\hwlabs\dlock). The testbench set up for this exercise is named tb_dlock.vhd. It is available on the course web page. Right click the link tb_dlock.vhd and download the file to your directory (H:\hwlabs\dlock).

Starting the Tools

For the first part start the Quartus II web edition application. Decline the offer to upgrade and start a new project.

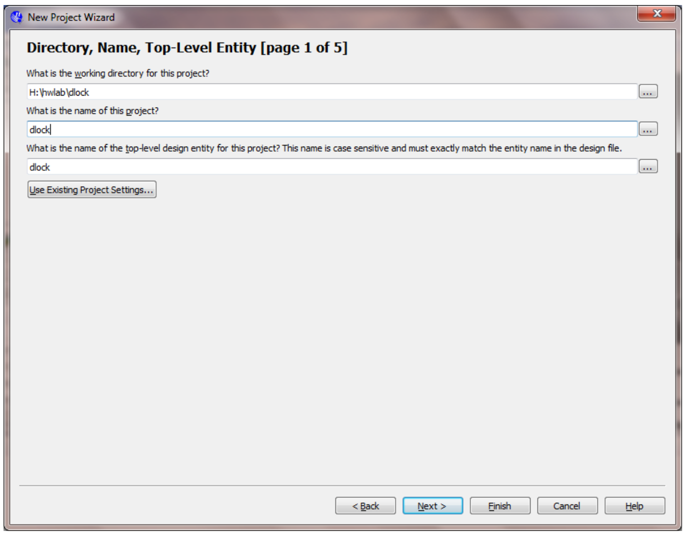
1. Select New Project Wizard from the opening window or from the "File" menu.

2. Click next to go past the introductory dialog and a new dialog box will appear, please enter the following information as shown in the figure:

   (a) Project Location: H:\hwlabs\dlock (your working directory)

   (b) Project name: dlock

   (c) Top level design entity: dlock



New Project Wizard

Directory, Name, Top-Level Entity [page 1 of 5]

What is the working directory for this project?

H:\hwlab\dlock

What is the name of this project?

dlock

What is the name of the top-level design entity for this project? This name is case sensitive and must exactly match the entity name in the design file.

dlock

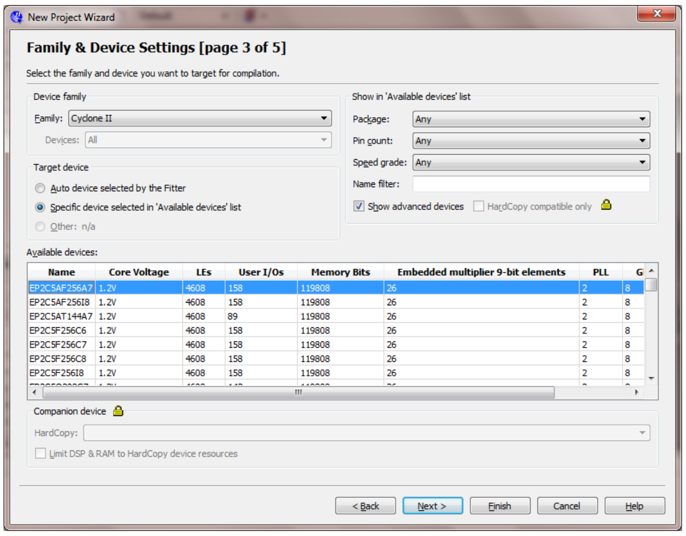
3. Click next, you may be asked to create a new directory if you have not done so already. After this step a dialog will appear for you to add files. As we do not have any files to add right now, just simply click next to skip this step.

4. You will be asked to select the device. Please enter the following information:

   (a) Family: Cyclone II

   (b) Target Device: Specific device selected in ”Available device” list

   (c) Available device: EP2C5AF256A7



New Project Wizard

Family & Device Settings [page 3 of 5]

Select the family and device you want to target for compilation.

Device family

Family: Cyclone II

Devices: All

Target device

Auto device selected by the Fitter

[✓ Specific device selected in 'Available devices' list]

Other: n/a

Show in 'Available devices' list

Package: Any

Pin count: Any

Speed grade: Any

Name filter:

[✓] Show advanced devices

[ ] HardCopy compatible only

Available devices:

Name	Core Voltage	LEs	User I/Os	Memory Bits	Embedded multiplier 9-bit elements	PLL	G
EP2C5AF256A7	1.2V	4608	158	119808	26	2	8
EP2C5AF256I8	1.2V	4608	158	119808	26	2	8
EP2C5AT144A7	1.2V	4608	89	119808	26	2	8
EP2C5F256C6	1.2V	4608	158	119808	26	2	8
EP2C5F256C7	1.2V	4608	158	119808	26	2	8
EP2C5F256C8	1.2V	4608	158	119808	26	2	8
EP2C5F256I8	1.2V	4608	158	119808	26	2	8

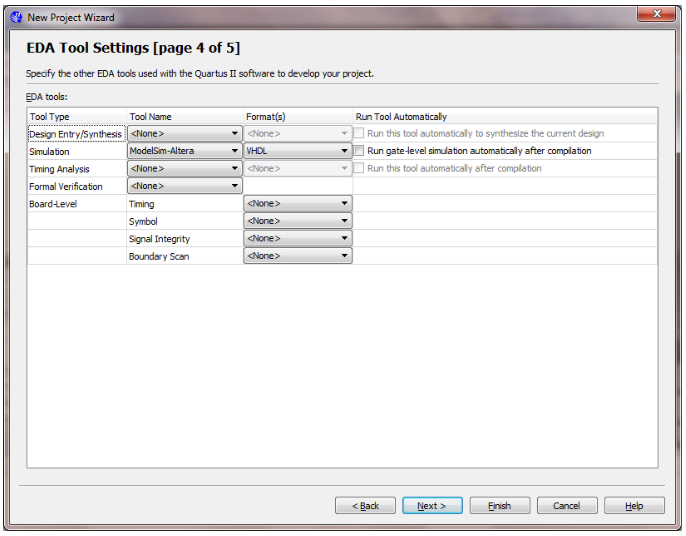
Companion device

HardCopy:

[ ] Limit DSP & RAM to HardCopy device resources

Then click next to continue.

5. You will be asked to specify the EDA tools. We will skip this and use default EDA tools by clicking next



New Project Wizard

EDA Tool Settings [page 4 of 5]

Specify the other EDA tools used with the Quartus II software to develop your project.

EDA tools:

Tool Type	Tool Name	Format(s)	Run Tool Automatically
Design Entry/Synthesis	<None>	<None>	☐ Run this tool automatically to synthesize the current design
Simulation	ModelSim-Altera	VHDL	☐ Run gate-level simulation automatically after compilation
Timing Analysis	<None>	<None>	☐ Run this tool automatically after compilation
Formal Verification	<None>		
Board-Level	Timing	<None>	
	Symbol	<None>	
	Signal Integrity	<None>	
	Boundary Scan	<None>	

6. A summary page will appear. Please click finish to confirm.

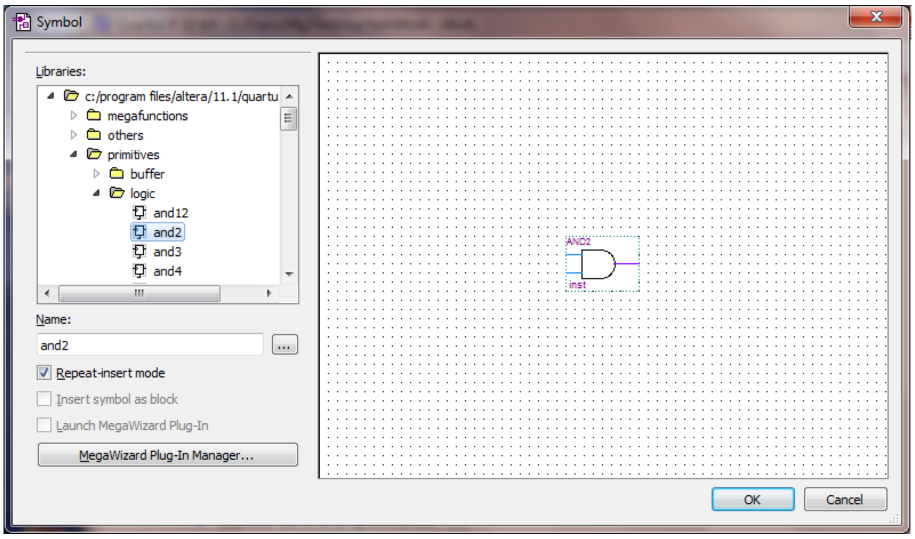
If you experience problems starting the Altera Quartus II tools try another PC and email (help@doc.ic.ac.uk) with the name of the suspect PC.

Entering your design.

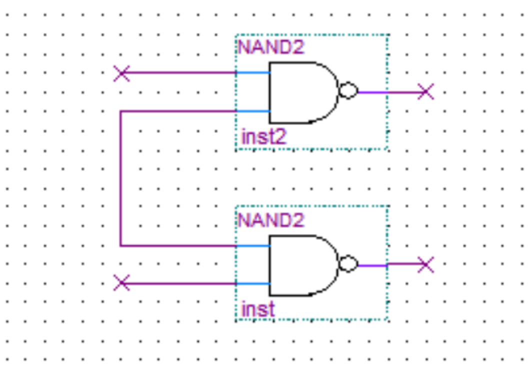
1. We will first create a schematic file to start the design. Select File→New from the pull-down menu, then select ”Block Diagram/Schematic File” from the list. Then click Ok to confirm.

2. A schematic sheet will appear in the main window with an associated toolbar. The icon indicates a symbol tool which allows us to choose different components to construct the circuit. The icon indicates an orthogonal tool which can draw wires to connect the components.

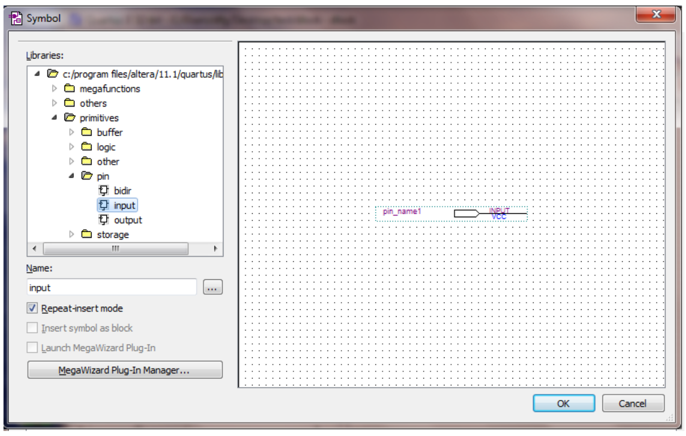
3. To add a component to the schematic diagram, first we click the icon found on the toolbar. A dialog will appear for you to choose the component libraries. Open up the directory tree and choose the logic gates from the primitives/logic directory as shown in the following figure.



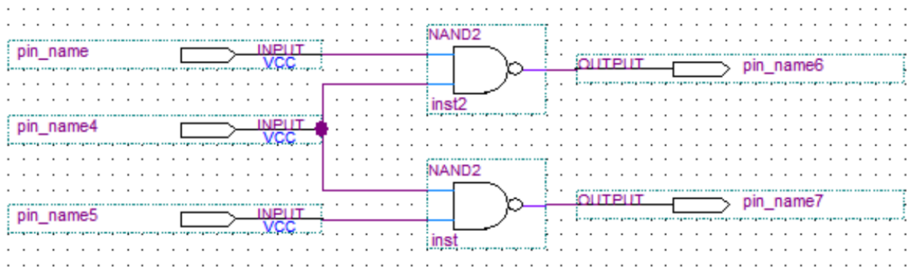
4. Choose the gate you want, press OK and you can place the component on the schematic sheet. For this exercise you are limited to the 16 types of component listed above the handout. The components can be found in these categories: primitive/logic, primitive/pin, primitive/storage and primitive/other. We can now place 2 NAND2 gate on the sheet, then use the orthogonal tool to connect these AND gates as follows:



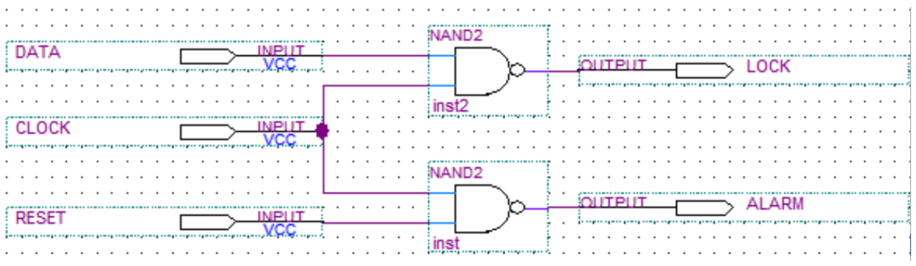
5. We will then give an entity to this circuit which defines the names of input pins and output pins. First select the input pin from symbol dialog



6. Place the input pins into the schematics and use the orthogonal tools to connect them. Repeat it for output pins as shown below:



7. Every I/O pin must be associated with a name. To change the name of an I/O pin (or a gate) right click the symbol and select properties. You can then change the default name.



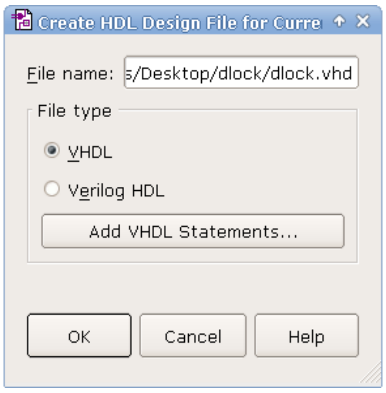
In your proper circuit it is a good idea to name the internal wires so that they can be tested in the simulation. Using the standard crosshair (selected with the arrow button on top of the left hand button bar) point to the wire and click right. A menu will appear, select properties then enter/edit the name. Wires with the same name are assumed connected regardless of whether they are joined on the wiring diagram. This feature is very useful when you find the circuit is complex and confusing if you draw all the wires. Terminal pins will automatically be connected to any wires with the same name.

Note that you must use the names DATA, RESET and CLOCK for the input pins and LOCK and ALARM for the output pins in order to make the simulation work. For all other wires adopt a sensible naming convention to help you when debugging your circuit. For example if as part of your state sequencing logic you have a wire corresponding to a tern $Q1 \cdot Q2'$ choose a name like Q1AndNQ2.

In order to debug the circuit you can use one of the output pins (for example LOCK) as a probe. Simply connect it to a stub of wire, and then name that stub with the wire you want to test. When you are sure that the circuit is working correctly just rename it to connect it to the lock output point.

8. After completing the simple circuit, you can now save the design. Save it using the default name ”dlock.bdf” so that you can edit it later using quartus.

9. In order to simulate your design you need to save it in a different format called VHDL. Select `File` → `Create / Update` → `Create HDL Design File from Current File...` from the application menu to export the design to a VHDL file (`dlock.vhd`).


Create HDL Design File for Curre
File name: s/Desktop/dlock/dlock.vhd
File type
[✓ VHDL] Verilog HDL

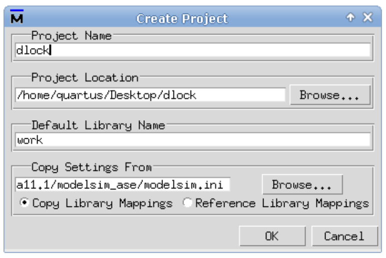
Simulating your design

Once you have entered your design you can use ModelSim to simulate it. This is a separate programme to Quartus II and you start it from the Windows program list.

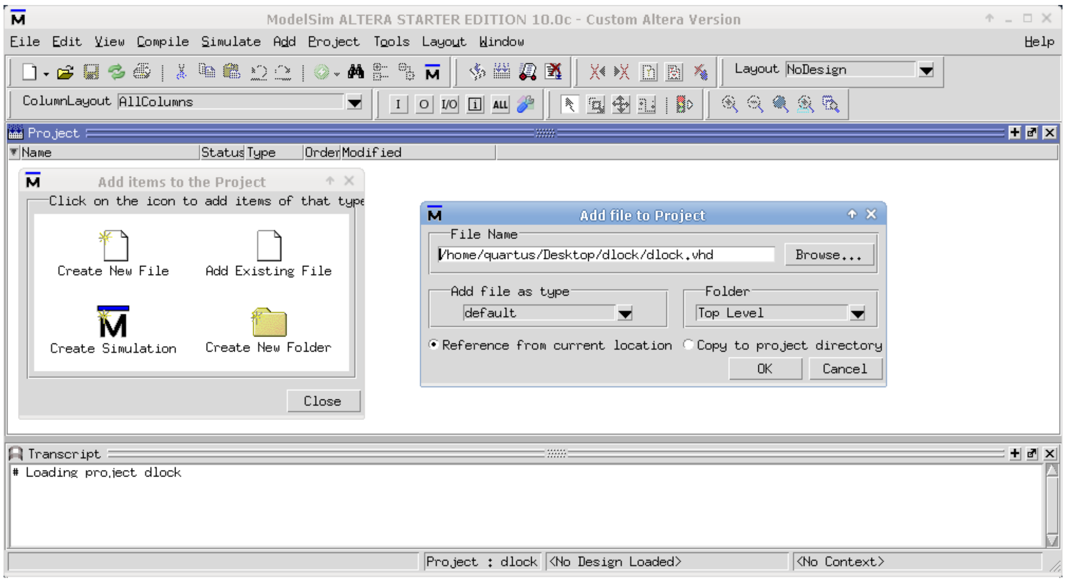
Testing a circuit design is done using a testbench. This is a simulation-only module which can be read by a simulator such as ModelSim, and which wraps around the circuit to be tested in order to generate input data for it and to evaluate the outputs. The testbench available on the course page generates data sequences for the numbers 0, 1, 2, 3...until 31. Each input is separated by a reset signal. You should check that your final circuit places a 0 on the lock output only on the correct sequence, and 1 otherwise.

Setting up the ModelSim project

When you have started ModelSim, create a new simulation project in your `dlock` directory. Make sure to leave the Default Library Name set to `work`.


Create Project
Project Name: dlock
Project Location: /home/quartus/Desktop/dlock
Default Library Name: work
Copy Settings From: a11.1/modelsim_ase/modelsim.ini
[✓ Copy Library Mappings] Reference Library Mappings

ModelSim will then ask for files to be added to the project. Add the file `dlock.vhd`, and the testbench `tb_dlock.vhd`, which you downloaded together with this specification. Make sure to leave all other settings at their defaults.


ModelSim ALTERA STARTER EDITION 10.0c - Custom Altera Version
File Edit View Compile Simulate Add Project Tools Layout Window Help
Layout NoDesign
ColumnLayout AllColumns
Project
Name Status Type Order Modified
Add items to the Project
Click on the icon to add items of that type
Create New File
Add Existing File
Create Simulation
Create New Folder
Close
Add file to Project
File Name
Browse...
Add file as type
Folder
Reference from current location
Copy to project directory
OK
Cancel
Transcript

File Name: /home/quartus/Desktop/dlock/dlock.vhd
Add file as type: default
Folder: Top Level
● Reference from current location ○ Copy to project directory

```
# Loading project dlock
```

Project : dlock <No Design Loaded> <No Context>

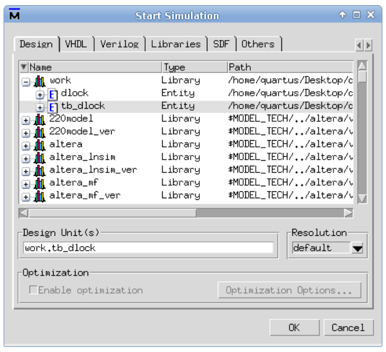
Compilation

Every time the design is changed, the internal ModelSim simulation files will have to be rebuilt. To perform this step, select `Compile` → `Compile All` from the menu.

Running the simulation

Now you need to tell ModelSim which of the loaded modules to use as the testbench, and thus as the module that wraps and tests your `dlock` design. Imagine the testbench as "pushing the buttons" of your digital lock and trying every possible combination for you.

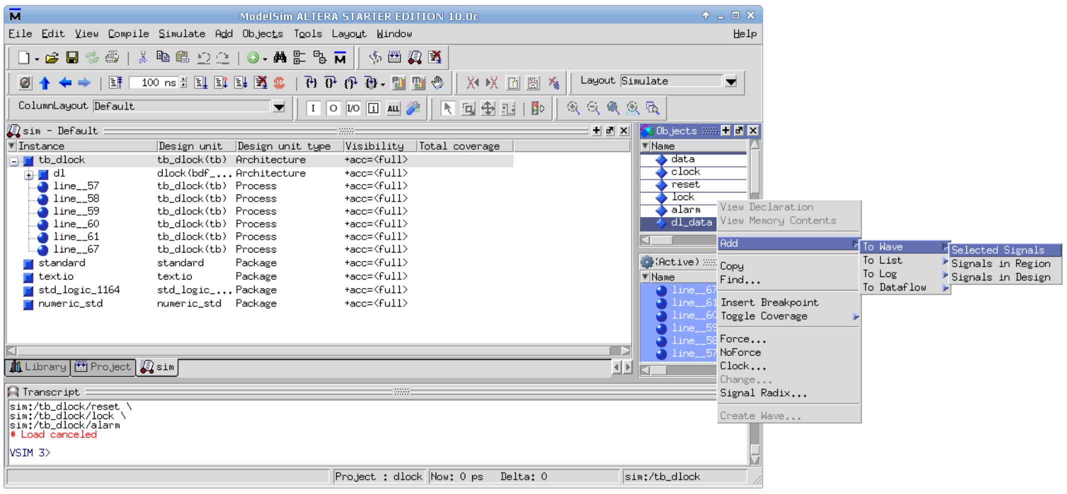
Select `Simulate` → `Start Simulation...` from the menu. The window *Start Simulation* will open, in which you need to choose `work` → `tb_dlock`.



Name	Type	Path
work	Library	/home/quartus/Desktop/c
dlock	Entity	/home/quartus/Desktop/c
tb_dlock	Entity	/home/quartus/Desktop/c
220model	Library	$MODEL_TECH/../altera/v
220model_ver	Library	$MODEL_TECH/../altera/v
altera	Library	$MODEL_TECH/../altera/v
altera_lnsim	Library	$MODEL_TECH/../altera/v
altera_lnsim_ver	Library	$MODEL_TECH/../altera/v
altera_mf	Library	$MODEL_TECH/../altera/v
altera_mf_ver	Library	$MODEL_TECH/../altera/v

Design Unit(s): work.tb_dlock
Resolution: default

After clicking OK, the simulation will be prepared. Once it is ready to run, the pane *Objects* will appear on the right hand side of the window. Select the signals `data`, `clock`, `reset`, `lock` and `alarm` (hold down CTRL to select multiple entries), right click on them and select `Add` → `To Wave` → `Selected Signals`.



sim - Default

Instance	Design unit	Design unit type	Visibility	Total coverage
tb_dlock	tb_dlock(tb)	Architecture	+acc=<full>	
dl	dlock(bdf_...	Architecture	+acc=<full>	
line__57	tb_dlock(tb)	Process	+acc=<full>	
line__58	tb_dlock(tb)	Process	+acc=<full>	
line__59	tb_dlock(tb)	Process	+acc=<full>	
line__60	tb_dlock(tb)	Process	+acc=<full>	
line__61	tb_dlock(tb)	Process	+acc=<full>	
line__67	tb_dlock(tb)	Process	+acc=<full>	
standard	standard	Package	+acc=<full>	
textio	textio	Package	+acc=<full>	
std_logic_1164	std_logic_...	Package	+acc=<full>	
numeric_std	numeric_std	Package	+acc=<full>	

Transcript

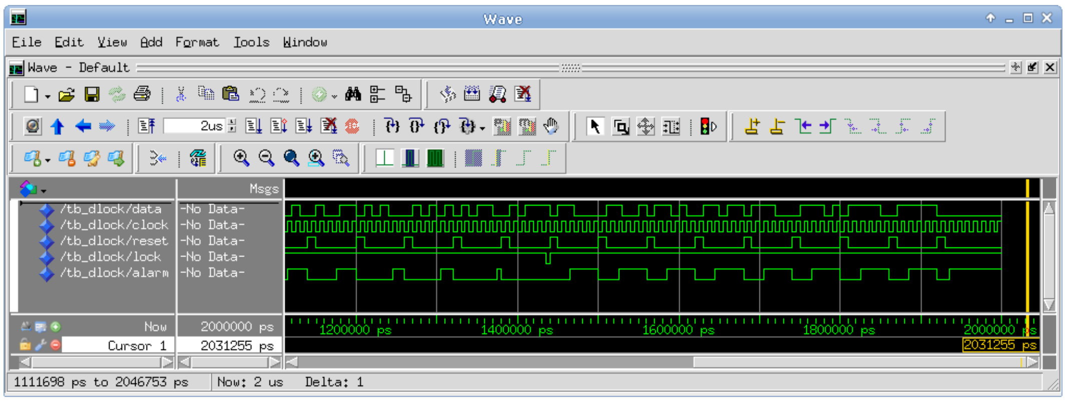
```
sim:/tb_dlock/reset \
sim:/tb_dlock/lock \
sim:/tb_dlock/alarm
# Load canceled

VSIM 3>
```

Project : dlock Now: 0 ps Delta: 0 sim:/tb_dlock

The wave viewer will appear and show the selected signals. You can separate it from the main window by dragging its title bar.

Set the simulation timer (the text box in the toolbar) to two microseconds by typing “2 us” in the box. A larger value will make it harder to see the relevant part of the output waveforms, a smaller value will abort the simulation before it is done.

To run the simulation, first clear any possible remainders of a previous simulation by clicking the button to the left of the simulation timer, `Restart`. Then, click the button to the right of the timer, `Run`. The waveform window should fill with the result of the simulation, which should look similar to this:



1111698 ps to 2046753 ps Now: 2 us Delta: 1

If the waveform graph is not shown properly, adjust the zoom using `View` → `Zoom` → `Zoom Full` in the waveform viewer menu. You can also click on the waveform and use + or - to zoom.

More simulations...

Congratulations, you have just simulated your first hardware device! If you need to change the block design file in Quartus, all you need to do to simulate the new design is *(provided you keep the ModelSim project open)*:

1. Export the changed block design file to a VHDL file
2. Recompile the ModelSim project
3. Restart and run the simulation

Required Submission

The submission is in two parts, both submitted electronically.

Report

You should produce a report in which you describe your solution to this design problem, showing your finite state machine, state transition table, Karnaugh maps, final circuit printout, waveform printouts and your calculation of the total silicon area. Include two versions of the wave forms, one zoomed in to show your binary sequence in detail, and the other zoomed out to show that the lock does not open on any other sequence. Both Quartus and ModelSim have the facility of saving the circuit/waveforms to an image file. Alternatively you can do the same using screen capture. For the other diagrams, such as the finite state machine diagram and the Karnaugh maps it is quickest to draw these by hand, scan them and then incorporate them into your report. You can trim diagrams to size using gimp on linux. Prepare the report using any standard software such as Libre Office or Latex, and submit one pdf file called report.pdf.

Design

The design submission must be done in the Linux environment. Open a terminal, change to your design directory ~/hwlabs/ and type:

```
tar zcf design.tar.gz dlock
```

This command will compress your entire directory into one .tar.gz file. Upload the file - design.tar.gz - using the CATE system.

Mark Scheme

The exercise is marked out of 40 and contributes 40% of the hardware continuous assessment. The breakdown of the marks is as follows:

1. Following the design procedure correctly (deducing the correct finite state machine and transition table, and minimising the state sequencing logic using Karnaugh maps) - 12 marks
2. Calculating the silicon area of your design - 4 marks
3. Using Quartus II to enter the correct schematic wiring diagram - 6 marks
4. Using ModelSim to produce a timing diagram - 6 marks
5. Producing the correct timing diagram for your particular combination - 8 marks
6. Producing a clear and concise report - 4 marks.