

**Department of Electrical & Electronic Engineering
Imperial College London**

E4.20 Digital IC Design

Laboratory 3: Using IRSIM for Switch Level Simulation

Objectives

By the end of this laboratory session, you should be able to:

- Generate .sim files for switch level simulation from Electric;*
- Run IRSIM switch-level simulator to simulate the ring oscillator;*
- Plot waveform results;*
- Learn how to use some of the more useful commands in IRSIM..*

The following section is relevant if you want to use IRSIM separately from Electric. The latest version of Electric has a built-in capability with IRSIM used as a plug-in. Please read the supplementary tutorial on how to use this plug-in (which is actually better).

Before you begin

- Before you start this laboratory, check that the following are installed on your PC:
(You Skip this part if you are using departmental computers)

- i. Cygwin – to check this, look for the icon



For your own machine, you must download and install cygwin from:
<http://www.cygwin.com>.

- ii. Tcl/Tk – to check this, start Cygwin and enter the command: wish. You should see a pop-up window. If Tcl/Tk is not installed, you must download and install them from:
<http://opencircuitdesign.com/irsim/>.
- iii. IRSIM – to check this, start Cygwin and in the terminal window, enter the command: **which irsimX**. You should get a response: “usr/local/bin/irsimX”.

- Cygwin is effectively a version of Linux that runs with MS-Windows. If you do not know Linux, you will probably have to learn a few useful commands, some are listed below:

Cygwin command	Meaning
pwd	Present working directory
ls -l	List directory
cd <path>	Change directory
cd ..	Move to parent directory
which	Report which version of programme to run

Exercise 1: Generation of .sim files & Invoking IRSIM

Step 1: Open the layout of the 11-stage ring oscillator and generate "**osc11.sim**" in Electric8.07 by: Tools -> Simulation (Others) ->Write IRSIM Deck.

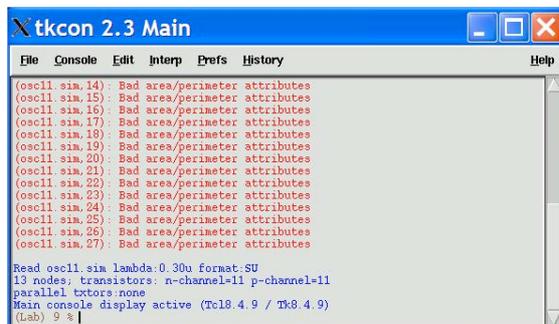
Step 2: Download "**tsmc18_v11.prm**" from http://www.ee.ic.ac.uk/pcheung/teaching/ee4_asic/labs/tsmc18.prm (0.18μ SCMOS technology file from TSMC)

Step 3: put "**osc11.sim**" and "**tsmc18_v11.prm**" in the same directory, e.g. "**h:/electric_lab/lab3**"

Step 4: Start Cygwin: "Start menu" -> "Cygwin" -> "Cygwin Bash Shell"

Step 5: In the console, use the "**cd**" command to set the path to the directory where your "**tsmc18_v11.prm**" and "**osc11.sim**" files are located. e.g. "**cd h:/electric_lab/lab3**".

Step 6: Now type "**irsim tsmc18_v11.prm osc11.sim**" in the console. IRSIM will run and launch the simulation console (tkcon 2.3). You can **ignore** the warnings in **red**.



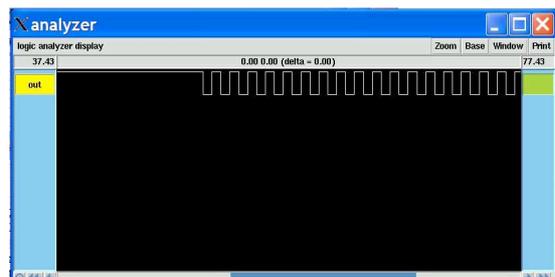
□ Enter the IRSIM command: "**d ring_out**" (display the value on the node "**ring_out**"). It should report that "**ring_out**"=X (which is undefined). This is not surprising. The circuit does not have an input, therefore "**ring_out**" is not initialised to a valid logic level.

□ Now try the IRSIM command: "**h ring_out**" (set "**ring_out**" to high). Followed by: "**s**" (simulate). It will report that time elapses to 10.0ns. By default, IRSIM will simulate for a period of **stepsize=10.0ns**. You may change this with the "**stepsize**" command. Now "**d ring_out**" will report that "**ring_out**"=1. You can switch on the reporting automatically with the watch command: "**w ring_out**"

□ We force the node "**ring_out**" to be high in order to initialize the ring oscillator. This is unique because of the feedback nature of the circuit and the fact that there is no other way that this circuit will ever find a stable initial state. In most other circuits, setting all inputs to some valid value will do the trick. For circuits which are difficult to initial, you may also need to force some outputs to known state first, then release the signal node with the command: "**x ring_out**".

□ Use command: "**t ring_out**" (Trace) followed by "**s 30**". You will see a list of events showing that "**ring_out**" is oscillating. Work, out the period of oscillation and compare it with that found with SPICE. What is the difference?

□ You can also display the waveform in a waveform window with the command: "**ana ring_out**". You should see:



□ You can perform normal window operations such as zoom in and out on the waveform. Using the appropriate mouse keys, you may also measure period on the waveform.

- You will find a full list of IRSIM commands at the back of the original LAB 3 handout from Prof. Cheung.

Exercise 2: Using Command File in IRSIM

- Use a text editor, create a file: osc11.cmd which contains the following collection of commands:
- Exit and re-enter IRSIM. Use command: `source osc11.cmd`, followed by `ana out`. You will now see the same results as what you have done in the previous exercise.
- It is always a good idea to test circuits with command files. Not only can you then perform regression testing easily (i.e. testing a circuit which is evolving or improving to make sure that it still works), you may also be able to generate the command file automatically from a high-level simulation of your chip. For example, it is often possible to create the stimuli and check for expected results from a Matlab simulation. Such a command file is often called a “test-bench”.

Exercise 3: Simulating ring oscillator with enable

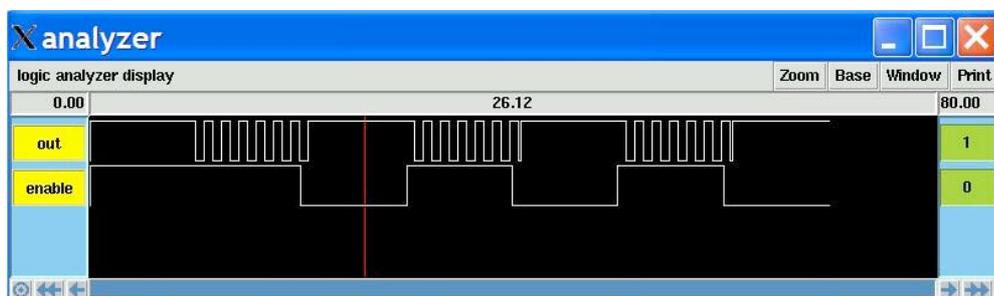
- Generate the .sim file for the osc11_nand circuit. Think about how you may simulate this with IRSIM. What values you should set enable to? How would you initialize this circuit?
- Attempt to simulate this yourself without looking at the solution.
- You need to drive the enable signal with a stimulus. You may set this to alternate 1,0,1... each lasting, say, 10ns. The commands sequence would be:

```
stepsize 10; h enable; s; l enable; s; h enable; s; l enable s;
ana out
```

- Defining a repetitive sequence of a stimulus can be achieved easier with the clock command. This sequence has the same effect as before:

```
stepsize 10          (Set each simulation step to be 10ns)
clock enable 1 0     (Define enable as clock with 1 followed by 0)
c 2                  (Run this sequence twice, i.e. two clock cycles)
```

- You should see the following results:



Commonly Used IRSIM Commands

The most frequently used commands in IRSIM are those that set the values of inputs and those that are used to view signals.

h <i>node1 node2 ...</i>	set list of nodes to logic 1 (high)
l <i>node1 node2 ...</i>	set list of nodes to logic 0 (low)
u <i>node1 node2 ...</i>	set list of nodes to "X" (undefined)
x <i>node1 node2 ...</i>	stop setting (release) the list of nodes
clock <i>node 1 0 1 0</i>	stop setting (release) the list of nodes
d <i>node1 node2 ...</i>	display current value of nodes
t <i>node1 node2 ...</i>	trace any changes in nodes
w <i>node1 node2 ...</i>	watch nodes (or auto-display)
ana <i>node1 node2 ...</i>	display current value of nodes in waveform viewer
clear	clear waveform viewer display
c <i>n</i>	run clock sequence <i>n</i> times
s <i>tm</i>	run simulation for <i>tm</i> nanosecond or one step
stepsize <i>tm</i>	set step size to <i>tm</i> nanoseconds

IMPORTANT. When you set a node high or low using the h or l commands, the node *keeps being set to high or low* (no matter what the circuit is trying to do to the node!) until you use the x command to stop setting the node. The combination of h/l and x is useful for forcing an output node to a defined state for initialization purposes.

Vectors

Since nodes typically are grouped into vectors, it is usually easier to look at *N*-bit quantities as single vector entities. The following commands can be used to define vectors and display them.

vector <i>name node1 node2 ...</i>	define a new vector called <i>name</i> consisting of the list of nodes
d <i>name</i>	display a vector as an array of bits
ana <i>name</i>	add vector <i>name</i> to the waveform viewer
set <i>name value</i>	set the bits of vector <i>name</i> using the binary string <i>value</i>

To set a vector to a hexadecimal number, use:

```
set name %x<hexstring>
```

The prefix %x says that what follows is a hex constant. For instance: **set counter_out %xff** force the 8-bit vector counter_out to have be all '1's.

A useful shortcut to defining arrays as long vectors is:

```
vector name a.b[31:0]
```

Scripts

A list of commands can be stored in a text file and executed within IRSIM together. If you create a file called **do_it.cmd**, you can read it into IRSIM by:

```
source do_it.cmd
```

Some other commands for scripts are given below.

# <i>comment</i>	# defines the rest of the line to be a comment. Comments must be on separate lines.
assert <i>name value</i>	This checks if name has value value. If this is true, the command does nothing.
print <i>text</i>	Otherwise, the command prints an error message. Echoes text on the output. Useful to separate the outputs of different major tests.

Built in IRSIM Simulator for Electric (Tutorial Sheet)

To Install (Note this will be done for you on the machines on Level 9):

1) Download electricIRSIM-8.05.jar from <http://www.staticfreesoft.com/jmanual/mchap01-05.html>. Save it to the same directory you have your electric-8.0X.jar file.

2) Create a batch file called 'electric.bat' containing the following:

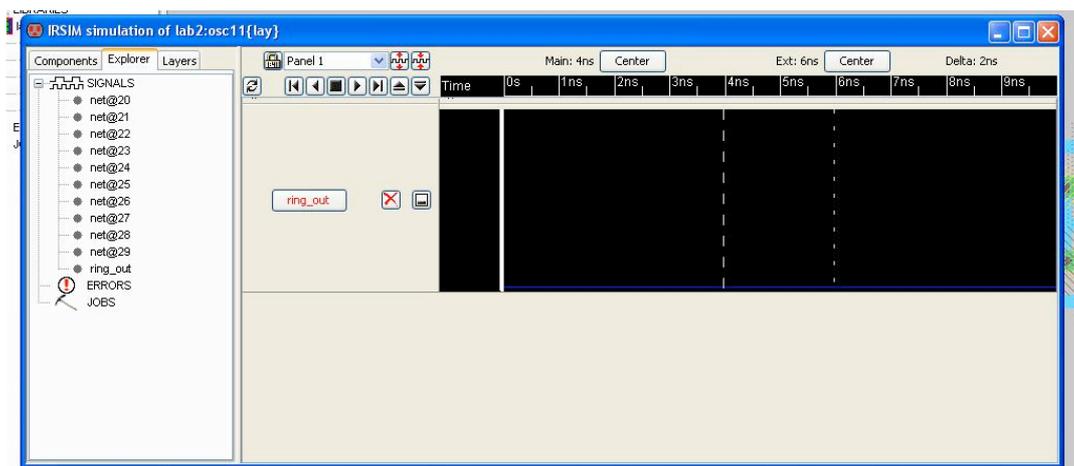
```
cd c:\electric
java -Xmx1000m -classpath electric-8.05.jar;electricIRSIM-8.05.jar
com.sun.electric.Launcher
```

Note: Replace c:\electric with your path to the directory where you have saved the two electric files (executable and IRSIM plug in). Also replace electric-8.05.jar with the executable jar file name that you have.

Basic Guide:

Open up your lab2 project

- 1) Go to File->Preferences->tools->Simulators
- 2) Set the parameter file to the scmos25.prm file you can download from Peter Cheung's course homepage (http://www.ee.ic.ac.uk/pcheung/teaching/ee4_asic/index.html)
- 3) Select multistate display and read up on what it does in the above user guide.
- 4) Select ok
- 5) Select osc11 {lay} – so that it displays the layout in the window
- 6) Select tools->simulation (built-in)-> IRSIM: Simulate Current Cell. The following dialog should appear (you may need to resize this):



- 7) Use the user-guide to familiarise yourself with how to zoom, pan and move the main and Ext markers.
- 8) Select ring_out – the box in main window – look at the layout window and notice it is highlighted the net here.
- 9) Move the main marker to near 0ns and chose tool-> simulation (built-in) -> Set Signal High at Main Time or press V
- 10) Move main marker to 2ns
- 11) Set the signal to undefined (tool-> simulation (built-in) -> Set Signal Undefined at Main Time or press X)
- 12) Observe how the waveform oscillates as expected.
- 13) Chose tool-> simulation (built-in) -> Save Stimuli to disk...
- 14) Look at the file it creates
- 15) Close the waveform window and restart the simulation (step 6)
- 16) Chose tool-> simulation (built-in) -> Restore Stimuli from disk... select lab2.cmd file generated in step 12.
- 17) Observe waveform window
- 1) As an extension try and do the same with osc11_nand – thinking about how you need to set the signals to make your circuit work.

Further Information

There is a good source of help on the internet at:

<http://www.staticfreesoft.com/jmanual/mchap09-05-01.html>

Please read this along with the waveform window help:

<http://www.staticfreesoft.com/jmanual/mchap04-12-01.html#mchap04-12-01>

before asking any questions.