BEE Design Flow Tutorials

Lesson 1
Flow Basics

Chen Chang
Kimmo Kuusilinna
Brian Richards

Lesson Goals

- Basic usage of Simulink with Xilinx System Generator
- Single FPGA routing and assigning I/O gateways
- Compiling the design for FPGA
- Downloading the design bit file to BEE system and run hardware emulation
- Basic ASIC flow operations

What do you need to start?

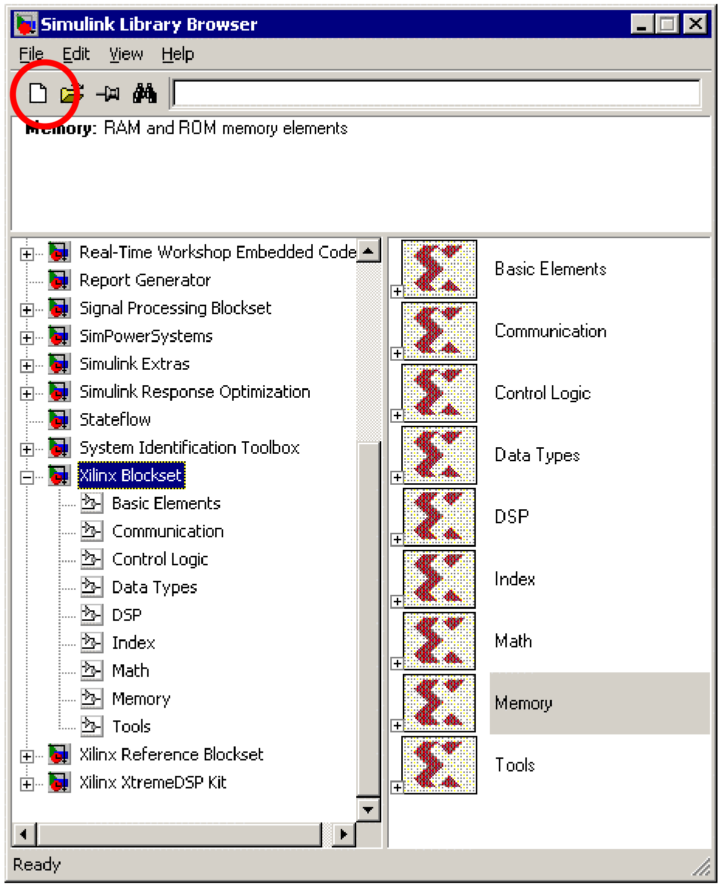
- MS Terminal Service Client software installed on the local PC machine, as well as SSH client and Exceed Xwindows program.
- Use MS Terminal Service Client logon to one of the BWRC terminal servers
  - Intel2650-1.eecs.berkeley.edu through Intel2650-7.eecs.berkeley.edu
- On the terminal server, map T: drive to \\hitz.eecs.berkeley.edu\tools
- For ASIC flow you need
  - Access to the BWRC Unix environment
    - E.g., sunV440-1.eecs.berkeley.edu
  - Signed the ST NDA to be granted access to vendor IP

Starting a new project

- SSH to a Unix machine (e.g., sun450-1)
- $ `tcsh`
- $ `setenv PATH /tools/insecta/current/bin:$PATH`
- $ `setenv DISPLAY` *your_machine*`:0.0`
  - This may be set automatically if SSH X11 tunneling is enabled.
- $ `cd /tools/designs/`*your_work_dir*
  - EG: /tools/designs/BEE/users/*login_name*
  - Avoid using your home directory – space is limited!
- $ `mk_insecta_project tut1`
  - This creates the tut1 directory with several design subdirectories.
- $ `cd tut1`
- The same directory can be accessed from the PC side as //hitz/Designs/*your_work_dir*/tut1
- And then return to the PC side for a little while…

Starting Matlab/Simulink

- First start Matlab, then type "simulink" in the command window
- Type "path" in the command window and verify \\hitz\designs\BEE\mlib is included
- Check the "Xilinx Blockset" toolbox is installed in the library browsing window
- Create a new model
  - File -> New -> Model



Memory: RAM and ROM memory elements

How is this Tutorial Organized

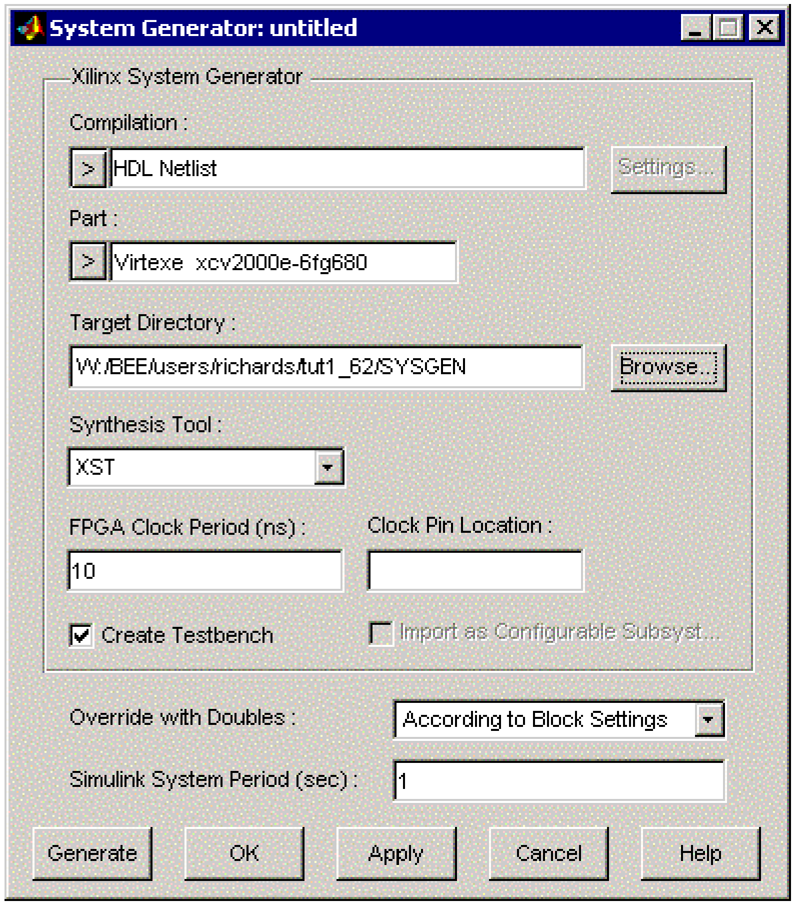
- Section 1: Creating and Simulating a Model
- Section 2: Tag and Route the Design conforming to BEE architecture
- Section 3: Generating FPGA Bit File Using BEE_ISE Tool
- Section 4: Download and Emulate Design on BEE Hardware
- Section 5: ASIC flow basics

Current Section

- Section 1: Creating and Simulating a Model
- Section 2: Tag and Route the Design conforming to BEE architecture
- Section 3: Generating FPGA Bit File Using BEE_ISE Tool
- Section 4: Download and Emulate Design on BEE Hardware
- Section 5: ASIC flow basics

Create a System Generator Model

- Drag a “System Generator” block to the design
- Double click on the block to bring up the property dialog
- Choose VirtexE family, XCV2000E-6-fg680 chip



System Generator: untitled

Xilinx System Generator

Compilation : HDL Netlist

Part : Virtexe xcv2000e-6fg680

Target Directory : W:/BEE/users/richards/tut1_62/SYSGEN

Synthesis Tool : XST

FPGA Clock Period (ns) : 10

Clock Pin Location :

[✓] Create Testbench

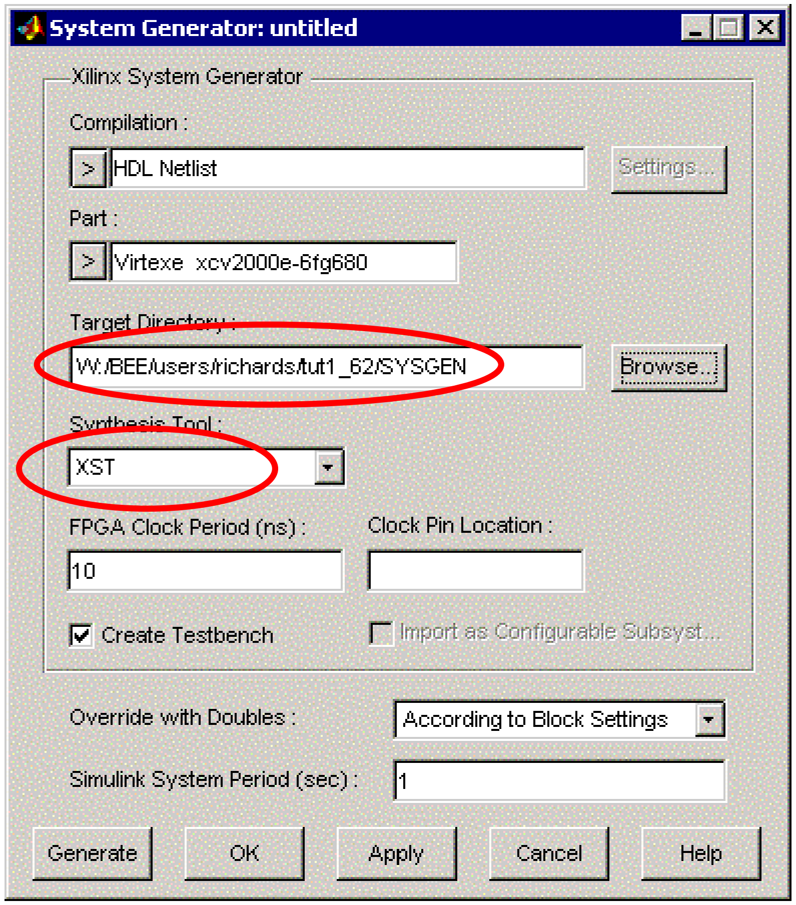
[ ] Import as Configurable Subsyst...

Override with Doubles : According to Block Settings

Simulink System Period (sec) : 1

Configure System Generator Block

- Synthesis tool should always be XST, different synthesis tool choices are offered in the BEE_ISE software during the FPGA generation step.
- Make sure that a network drive is mapped to //hitz/designs (eg: W:)
- Set “Target Directory” to W:/<work_dir>/ tut1/SYSGEN



System Generator: untitled

Xilinx System Generator

Compilation : HDL Netlist

Part : Virtexe xcv2000e-6fg680

Target Directory : W:/BEE/users/richards/tut1_62/SYSGEN

Synthesis Tool : XST

FPGA Clock Period (ns) : 10

Clock Pin Location :

☑ Create Testbench

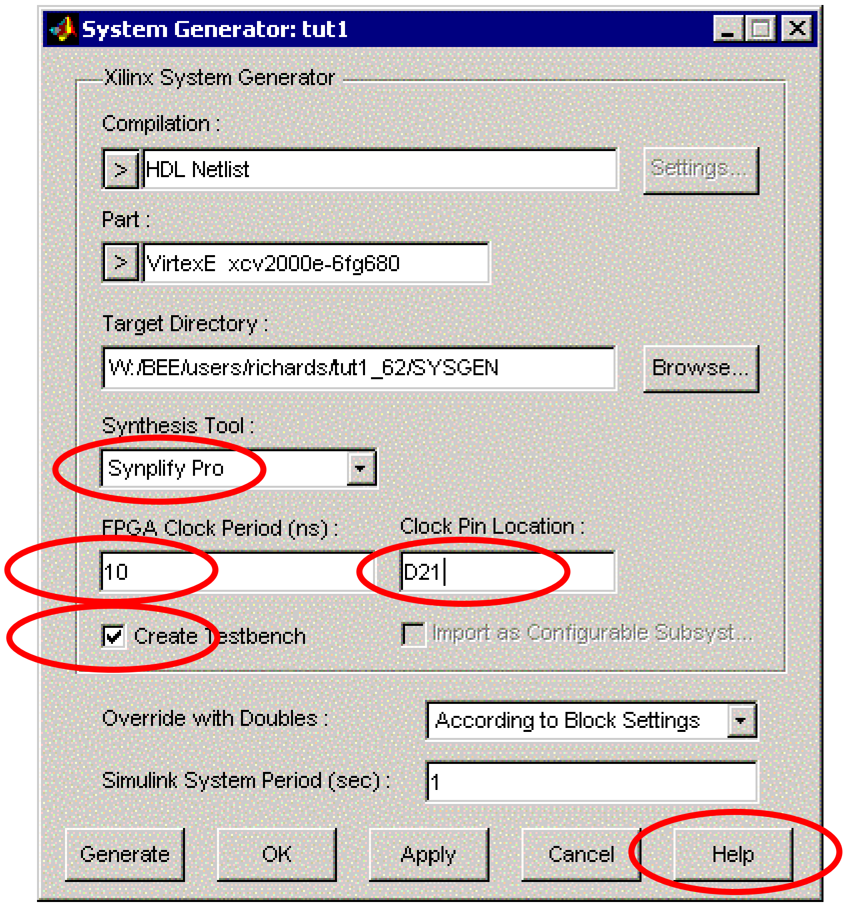
☐ Import as Configurable Subsyst...

Override with Doubles : According to Block Settings

Simulink System Period (sec) : 1

Configure System Generator Block

- Synthesis Tool: Synplify Pro
- FPGA System Clock Period should be the target design's fastest clock rate
- Check the "Create Testbench" box to enable VHDL testbench generation
- The Clock Pin Location is D21 on the BEE units
- More information on the block can be found by clicking on the "Help" button


System Generator: tut1

Xilinx System Generator

Compilation : HDL Netlist

Part : VirtexE xcv2000e-6fg680

Target Directory : W:/BEE/users/richards/tut1_62/SYSGEN

Synthesis Tool : Synplify Pro

FPGA Clock Period (ns) : 10

Clock Pin Location : D21

[✓] Create Testbench

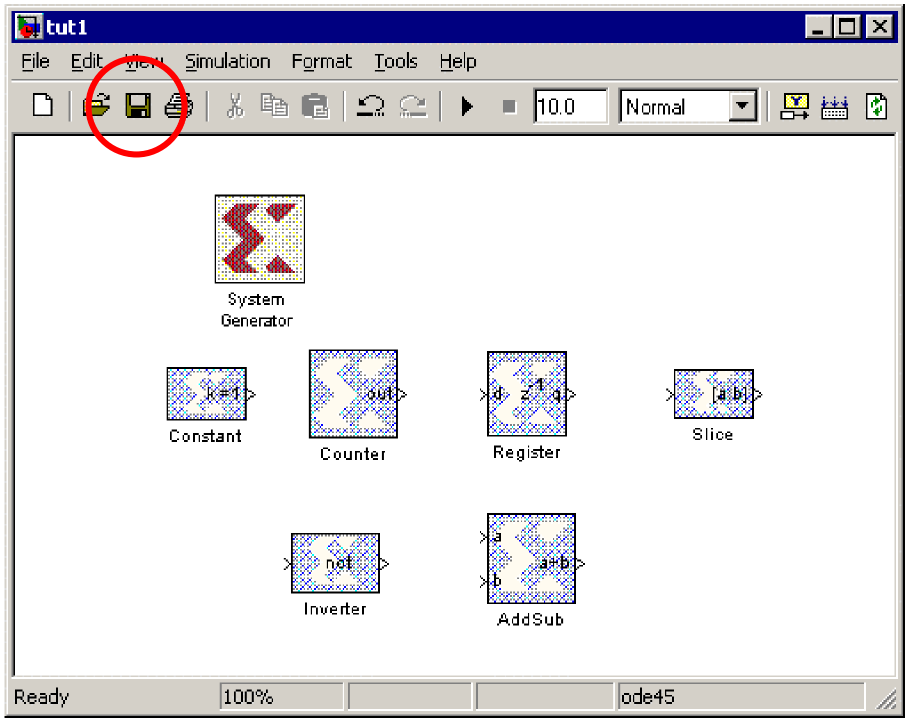
[ ] Import as Configurable Subsyst...

Override with Doubles : According to Block Settings

Simulink System Period (sec) : 1

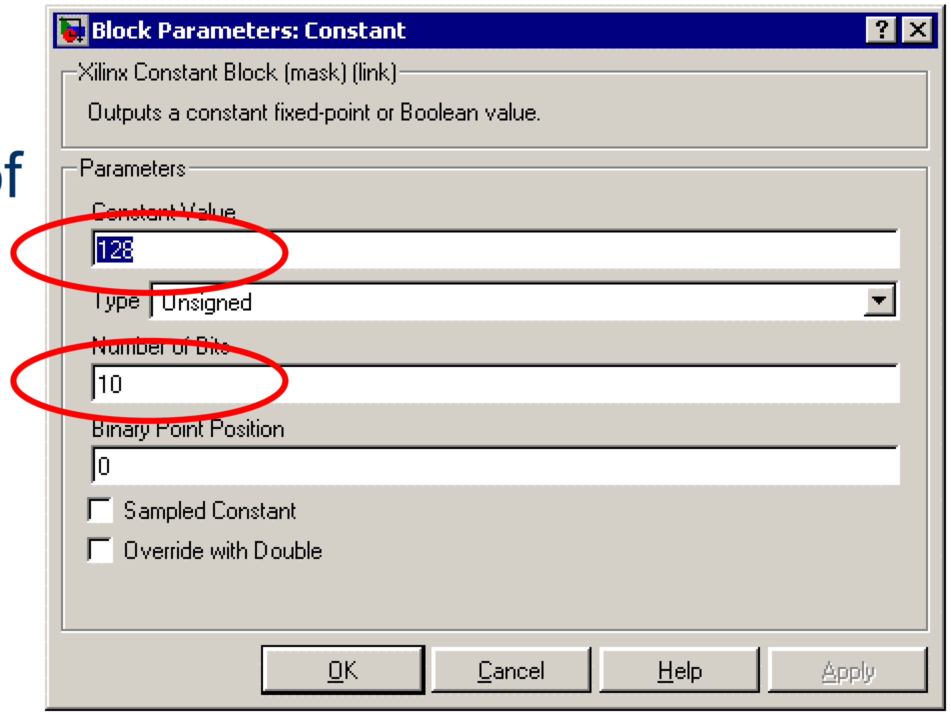
Add More Blocks to the Model

- Drag the “constant”, “counter”, “Register”, “Slice” blocks to the design from the Basic Elements library
- Drag the “Inverter”, “AddSub” block from the Math library
- Now save your model as W:/<work_dir >/ tut1/tut1.mdl



Modify Constant Parameters

- Change the "Constant Value" to 128
- Change the "Number of Bits" to 10
- Click on "OK" to close the dialog


Block Parameters: Constant

Xilinx Constant Block (mask) (link)

Outputs a constant fixed-point or Boolean value.

Parameters

Constant Value: 128

Type: Unsigned

Number of Bits: 10

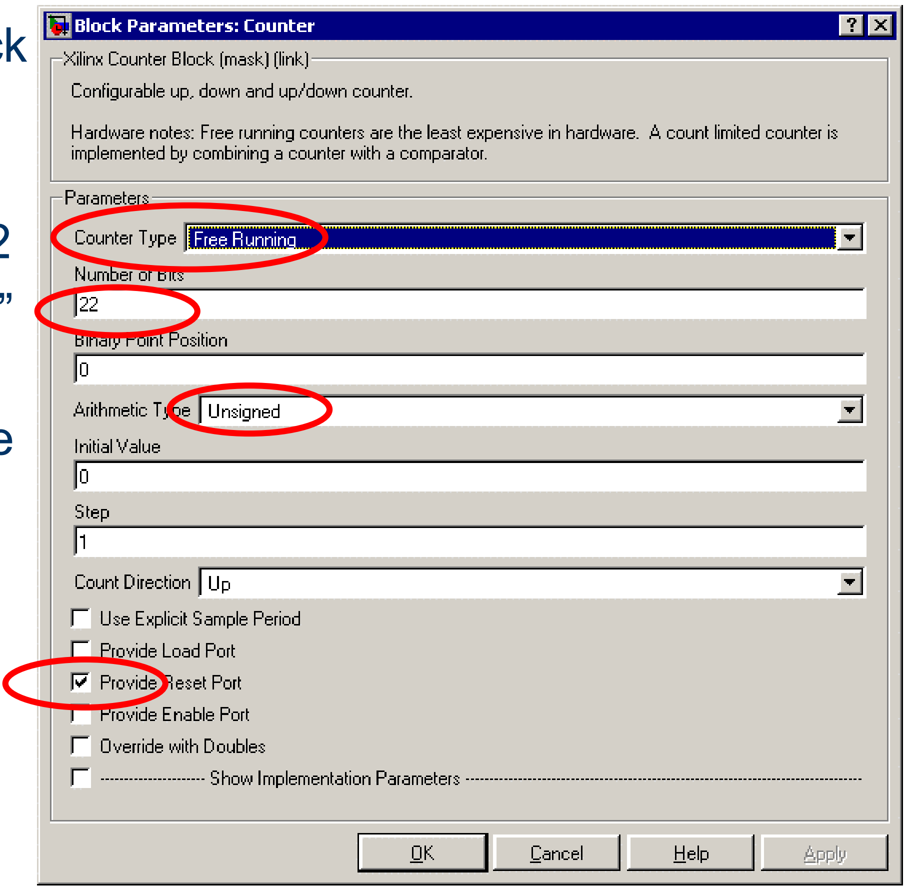
Binary Point Position: 0

☐ Sampled Constant

☐ Override with Double

Modify Counter Parameters

- Double click on the “Counter” block
- Select the Free Running counter type
- Change the “Number of Bits” to 22
- Use Arithmetic Type of “Unsigned”
- Check “Provide Reset Port”, an additional port “rst” appears on the block
- Click on “OK” to close the dialog



Xilinx Counter Block (mask) (link)

Configurable up, down and up/down counter.

Hardware notes: Free running counters are the least expensive in hardware. A count limited counter is implemented by combining a counter with a comparator.

Parameters

Counter Type: Free Running

Number of Bits: 22

Binary Point Position: 0

Arithmetic Type: Unsigned

Initial Value: 0

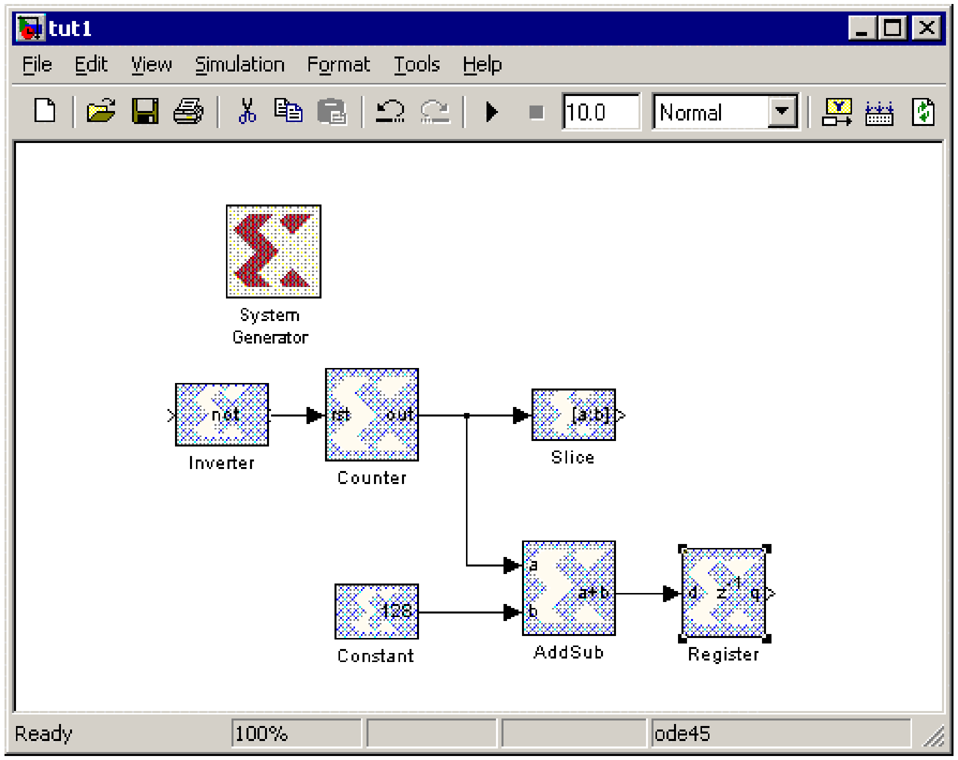
Step: 1

Count Direction: Up

- [ ] Use Explicit Sample Period
- [ ] Provide Load Port
- [✓] Provide Reset Port
- [ ] Provide Enable Port
- [ ] Override with Doubles
- [ ] Show Implementation Parameters

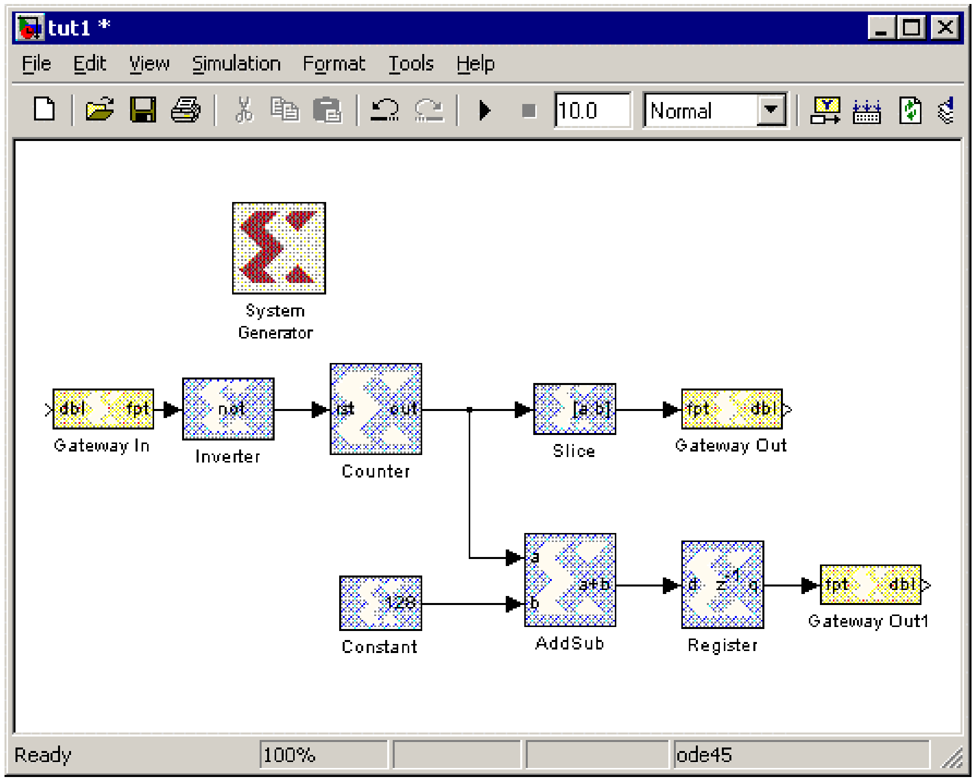
Moving and Connecting Blocks

- Move the blocks and connect them up as shown on the right



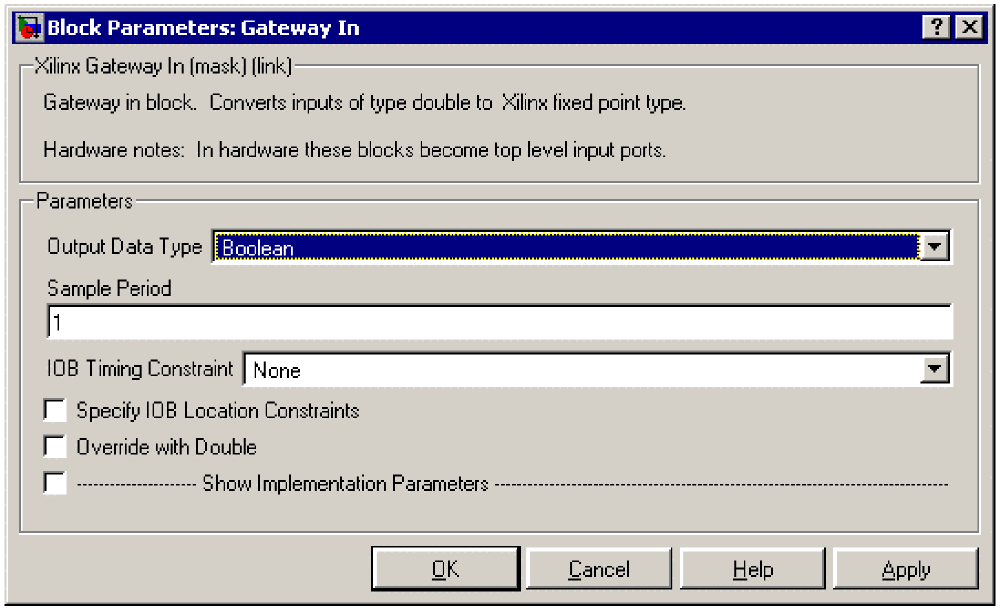
Adding Xilinx Gateway Blocks

- Add one “Gateway In” and two “Gateway Out” blocks from the Basic Elements library
- Connect them with the rest of the blocks as shown on the right
- These gateways are used to connect to native Simulink blocks and also to identify the hardware design boundary



Modify Gateway In Parameters

- Change the “Output Data Type” to Boolean
- Make sure the “Sample Period” is 1



Block Parameters: Gateway In

Xilinx Gateway In (mask) (link)

Gateway in block. Converts inputs of type double to Xilinx fixed point type.

Hardware notes: In hardware these blocks become top level input ports.

Parameters

Output Data Type: Boolean

Sample Period: 1

IOB Timing Constraint: None

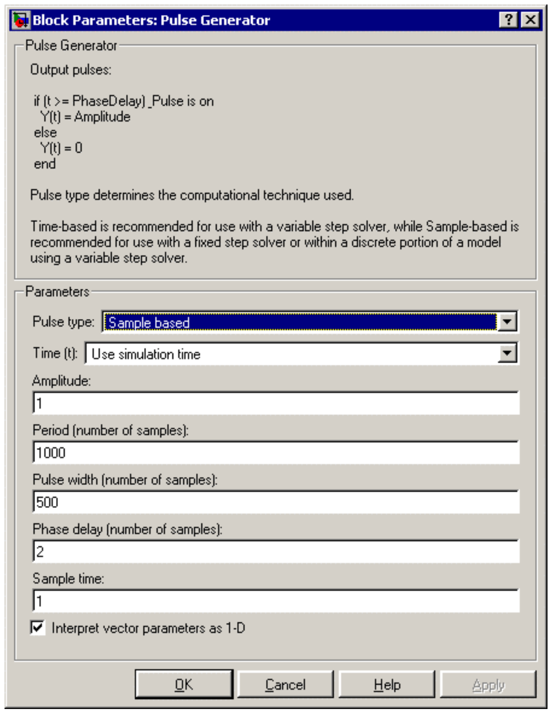
☐ Specify IOB Location Constraints

☐ Override with Double

☐ Show Implementation Parameters

Adding Native Simulink Blocks

- Add one "Pulse Generator" block from Simulink/Sources library, and two "Scope" blocks from Simulink/Sinks library



Block Parameters: Pulse Generator

Pulse Generator

Output pulses:

```
if (t >= PhaseDelay) _Pulse is on
  Y(t) = Amplitude
else
  Y(t) = 0
end
```

Pulse type determines the computational technique used.

Time-based is recommended for use with a variable step solver, while Sample-based is recommended for use with a fixed step solver or within a discrete portion of a model using a variable step solver.

Parameters

Pulse type: Sample based

Time (t): Use simulation time

Amplitude: 1

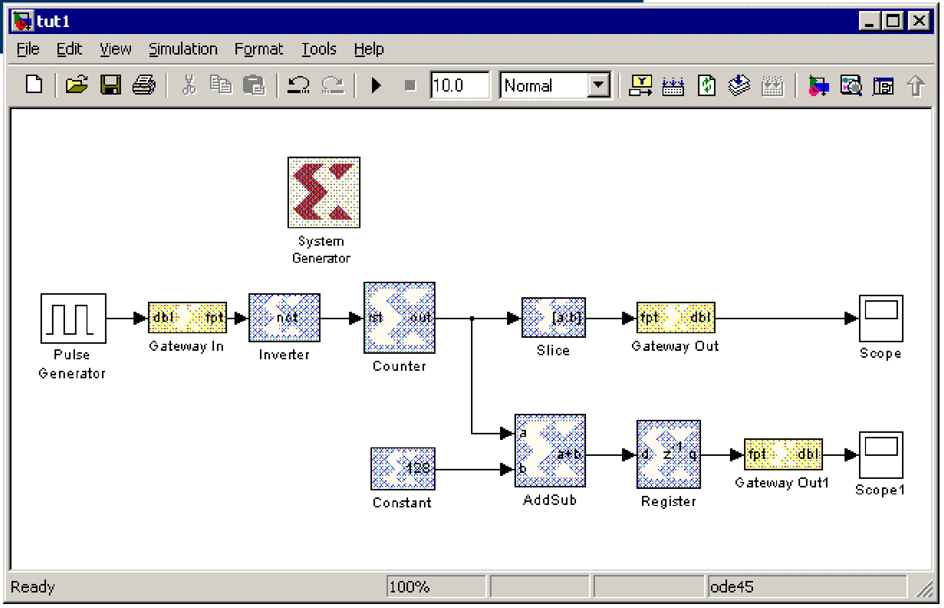
Period (number of samples): 1000

Pulse width (number of samples): 500

Phase delay (number of samples): 2

Sample time: 1

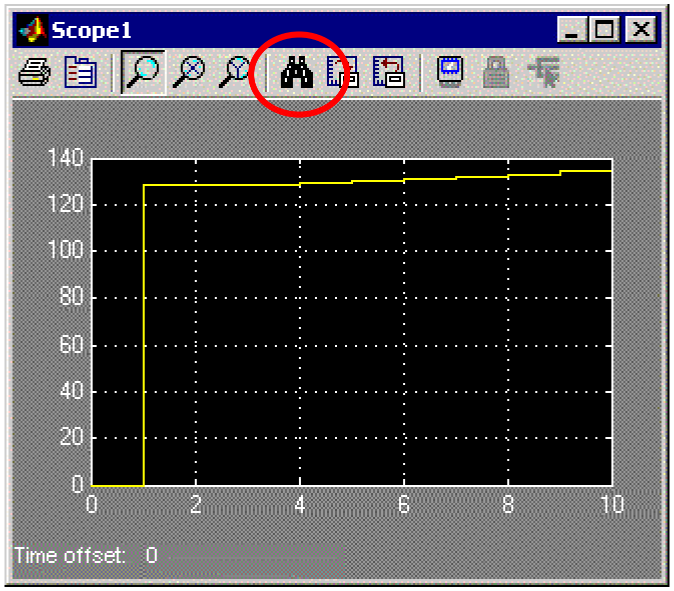
☑ Interpret vector parameters as 1-D



- Configure the Pulse Generator parameters:
  - Pulse Type = Sample Based
  - Period = 1000
  - Pulse Width = 500
  - Phase delay = 2
- Place and connect them as shown above

Simulating the Model

- Start the simulation
  - Simulation -> start
- Double click to open the “Scope1” block, right-click on “Autoscale” (binocular icon), and verify that it looks like the figure on the right

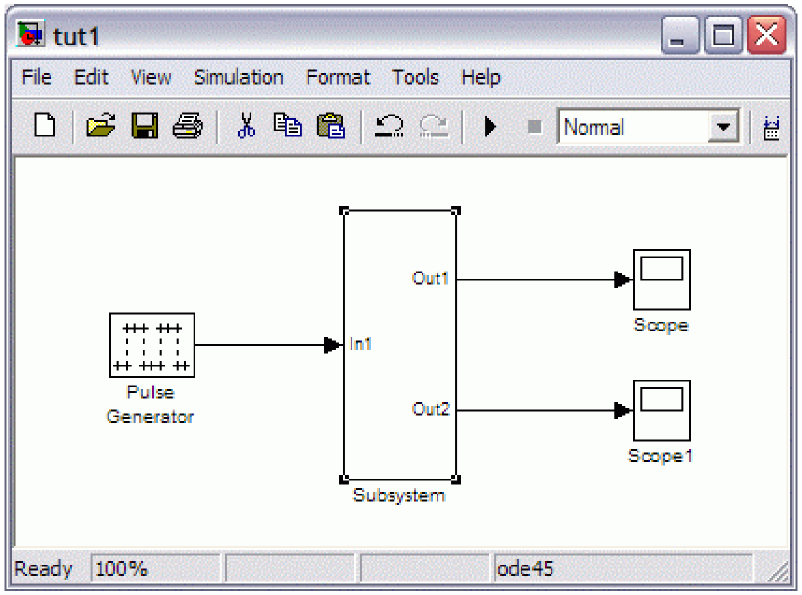


Current Section

- Section 1: Creating and Simulating a Model
- Section 2: Tag and Route the Design conforming to BEE architecture
- Section 3: Generating FPGA Bit File Using BEE_ISE Tool
- Section 4: Download and Emulate Design on BEE Hardware
- Section 5: ASIC flow basics

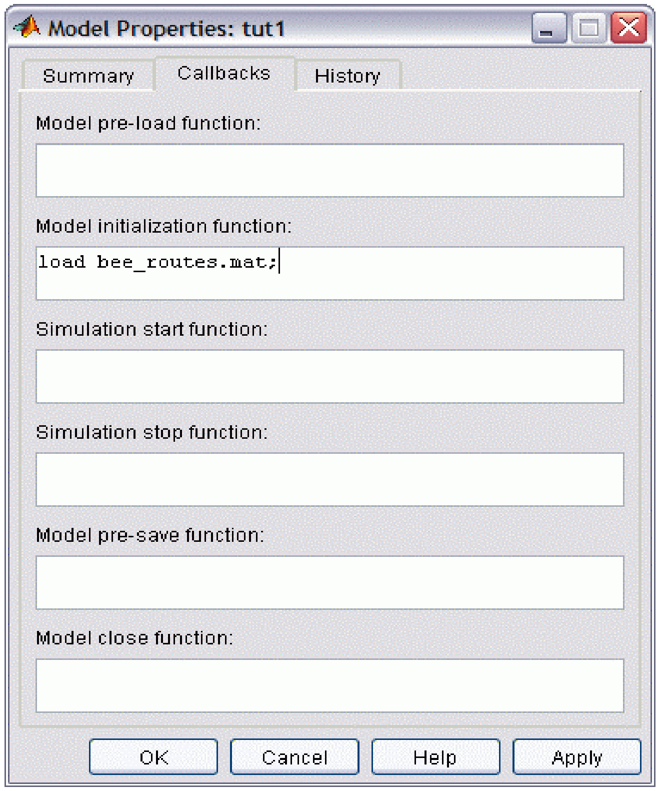
Creating Subsystem

- Click and drag on the model to select all Xilinx library blocks, including the System Generator block, ignoring the Pulse Generator and Scope blocks
- Right click and select "create subsystem"
- This creates a subsystem that will be later targeted to a FPGA



Configure BEE Routes Loading

- Open the Model Properties Dialog
  - File -> Model Properties
- Click on the Callbacks tab, and add “load bee_routes.mat;” to Model initialization function
- This loads the BEE routing specific variables to the model at initialization



Model pre-load function:

Model initialization function:

```
load bee_routes.mat;
```

Simulation start function:

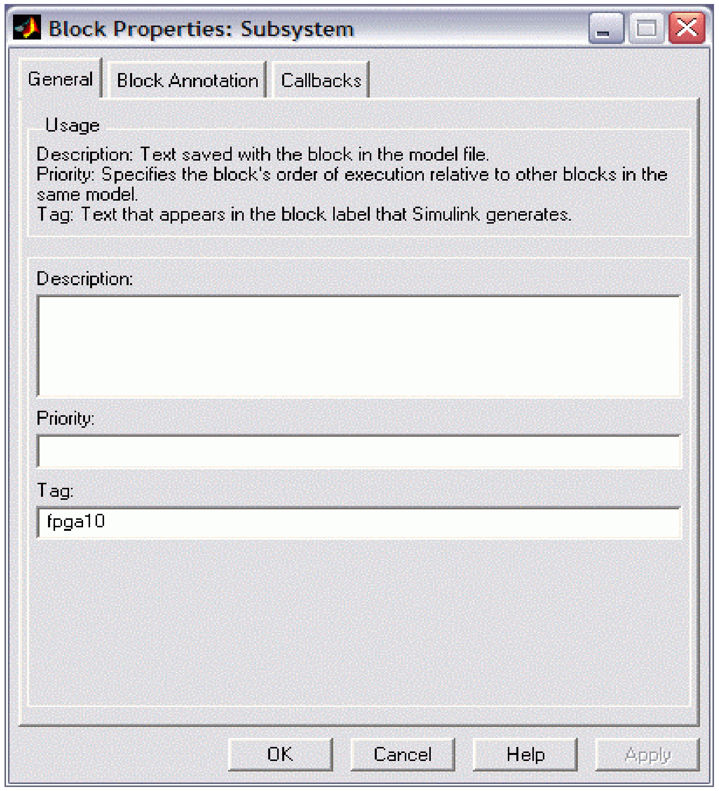
Simulation stop function:

Model pre-save function:

Model close function:

Tag Subsystem

- Right click on the subsystem and select Block properties
- In the “Tag” field type in “fpga10” to indicate that this subsystem will be implemented on FPGA10 of the BEE board



Usage

Description: Text saved with the block in the model file.
Priority: Specifies the block's order of execution relative to other blocks in the same model.
Tag: Text that appears in the block label that Simulink generates.

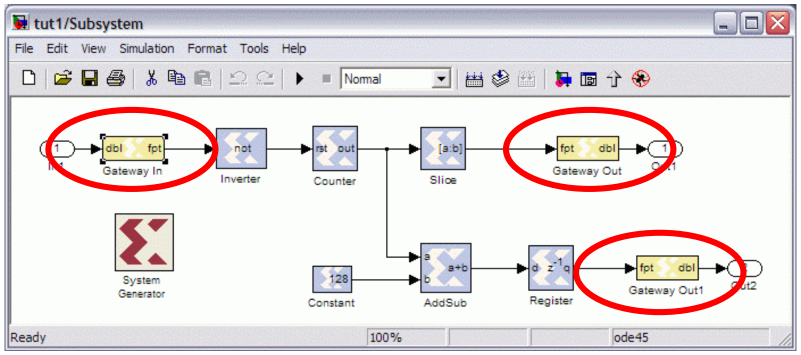
Description:

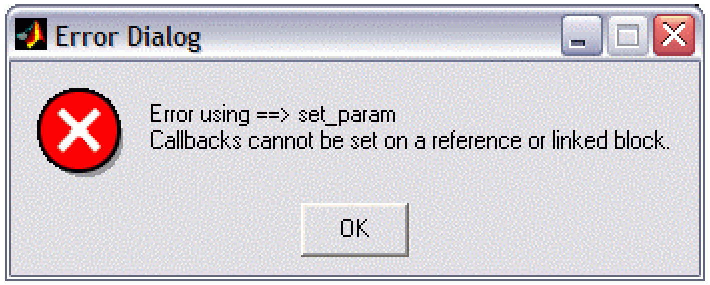
Priority:

Tag: fpga10

Tag Gateways

- Similarly tag “Gateway In” as “rst”, “Gateway Out” as “led”, “Gateway Out1” as “conn00_bus1”
- An error dialog might show up as on shown on the right, just ignore it. This is a minor bug of Simulink R13.
- This indicates the gateways to use “rst”, “led”, and external conn00_bus1 I/O ports





Error using ==> set_param
Callbacks cannot be set on a reference or linked block.

Invoke BEE Router

- In Matlab command window type “bee_router(‘tut1’)” to invoke the BEE router

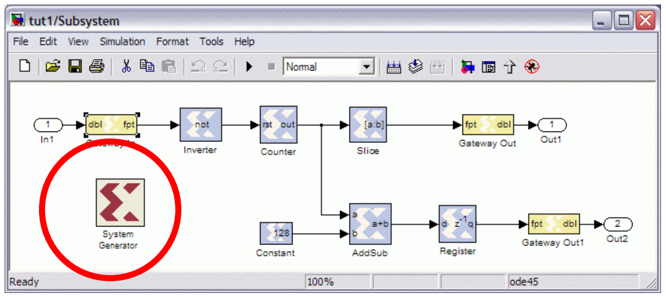
```
>> bee_router('tut1')
Warning: Xilinx System Generator version not specified, assuming V2.2

Routing tut1/Subsystem
Successfully routed link from tut1/Subsystem/Gateway Out(fpga10) to led
Successfully routed link from tut1/Subsystem/Gateway Out1(fpga10) to conn00_bus1
Successfully routed link from rst to tut1/Subsystem/Gateway In(fpga10)

Routing tut1/Subsystem/ System Generator

BEE routing complete.
```

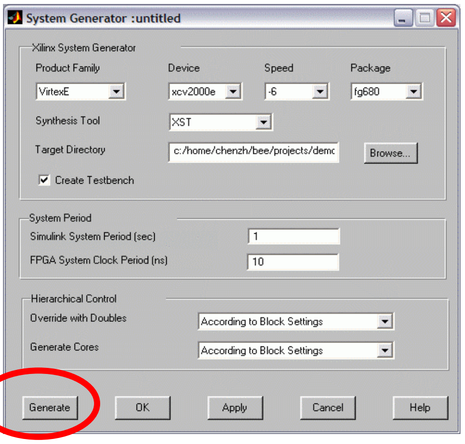
Current Section

- Section 1: Creating and Simulating a Model
- Section 2: Tag and Route the Design conforming to BEE architecture
- Section 3: Generating FPGA Bit File Using BEE_ISE Tool
- Section 4: Download and Emulate Design on BEE Hardware
- Section 5: ASIC flow basics

Invoke System Generator

- Save the Model
- Double click on the System Generator block
- Click on the “Generate” button to invoke System Generator
- This creates the necessary VHDL files and project files
- *NEW for returning users* (12/2003): The bee_mc_gen command was formerly run at this point, and is no longer needed for XSG 3.1 and newer.




System Generator :untitled

Xilinx System Generator

Product Family	Device	Speed	Package
VirtexE	xcv2000e	-6	fg680

Synthesis Tool: XST

Target Directory: c:/home/chenzh/bee/projects/demc

☑ Create Testbench

System Period

Simulink System Period (sec): 1

FPGA System Clock Period (ns): 10

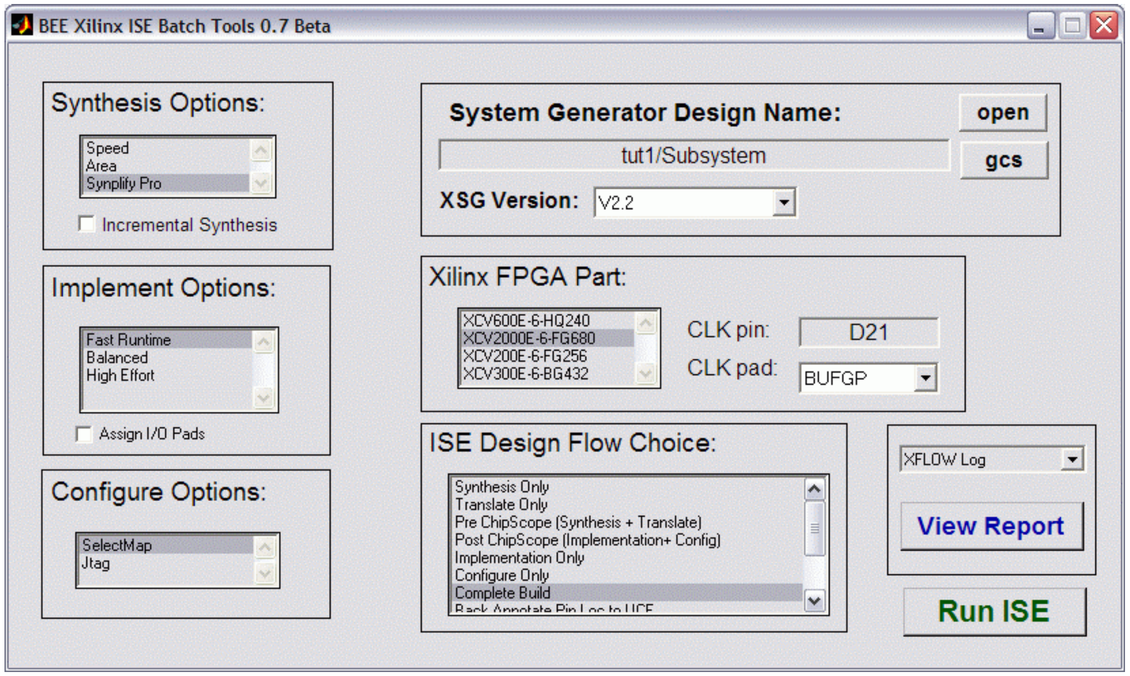
Hierarchical Control

Override with Doubles: According to Block Settings

Generate Cores: According to Block Settings

Open BEE ISE GUI

- In the Matlab command window type “bee_ise”
- Check the Design Name is “tut1/Subsystem”


BEE Xilinx ISE Batch Tools 0.7 Beta

Synthesis Options:
Speed
Area
Synplify Pro
Incremental Synthesis

Implement Options:
Fast Runtime
Balanced
High Effort
Assign I/O Pads

Configure Options:
SelectMap
Jtag

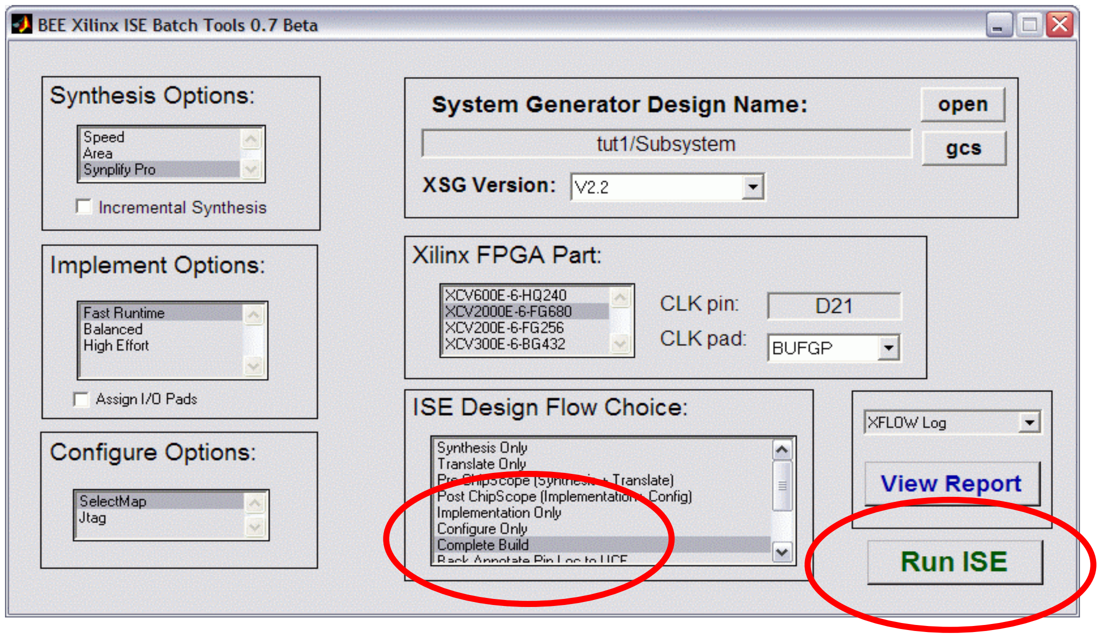
System Generator Design Name: tut1/Subsystem
XSG Version: V2.2

Xilinx FPGA Part:
XCV600E-6-HQ240
XCV2000E-6-FG680
XCV200E-6-FG256
XCV300E-6-BG432
CLK pin: D21
CLK pad: BUFGP

ISE Design Flow Choice:
Synthesis Only
Translate Only
Pre ChipScope (Synthesis + Translate)
Post ChipScope (Implementation+ Config)
Implementation Only
Configure Only
Complete Build
Back Annotate Pin Loc to UCF

Running BEE ISE

- Select “Complete Build” as the Design flow choice, this will synthesize, implement, and generate the FPGA bit file
- Click on the “Run ISE” button to invoke the tools



BEE Xilinx ISE Batch Tools 0.7 Beta

Synthesis Options:
Speed
Area
Synplify Pro
Incremental Synthesis

Implement Options:
Fast Runtime
Balanced
High Effort
Assign I/O Pads

Configure Options:
SelectMap
Jtag

System Generator Design Name: tut1/Subsystem
open
gcs
XSG Version: V2.2

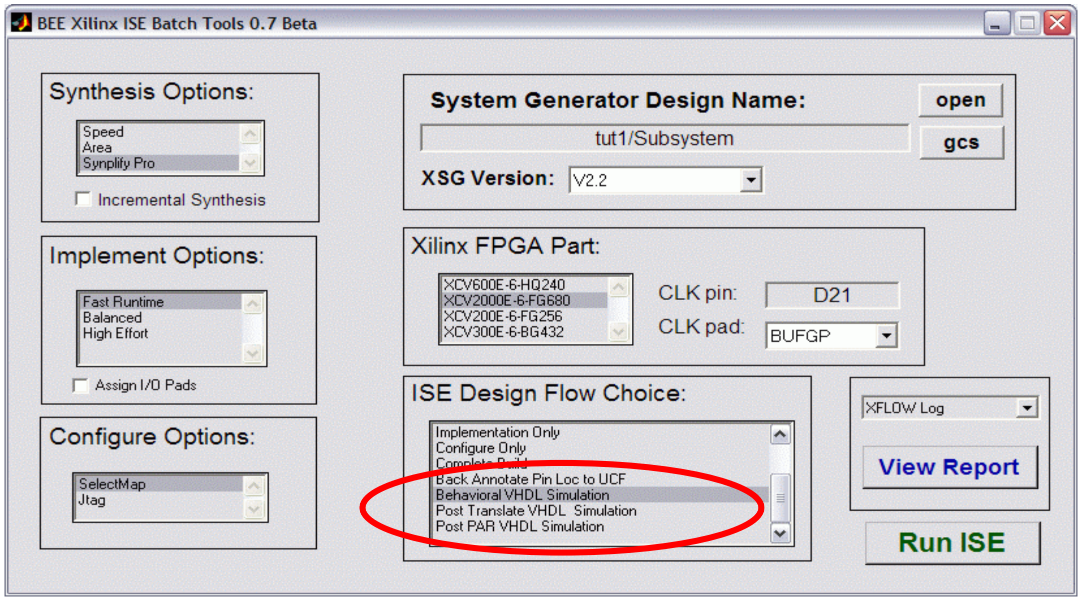
Xilinx FPGA Part:
XCV600E-6-HQ240
XCV2000E-6-FG680
XCV200E-6-FG256
XCV300E-6-BG432
CLK pin: D21
CLK pad: BUFGP

ISE Design Flow Choice:
Synthesis Only
Translate Only
Pre ChipScope (Synthesis + Translate)
Post ChipScope (Implementation + Config)
Implementation Only
Configure Only
Complete Build
Back Annotate Pin Loc to UCF

XFLOW Log
View Report
Run ISE

VHDL Simulation

- If VHDL simulation is desired, click on one of the three simulation flow choices, then click on Run ISE to invoke ModelSim for VHDL simulation
  - Behavioral VHDL Simulation can be run right after invoking System Generator
  - Post Translate VHDL Simulation require Synthesis and Translate steps to be run first
  - Post PAR VHDL Simulation require Synthesis and Implementation steps to be run first


BEE Xilinx ISE Batch Tools 0.7 Beta

Synthesis Options: Speed, Area, Synplify Pro (selected); Incremental Synthesis

Implement Options: Fast Runtime (selected), Balanced, High Effort; Assign I/O Pads

Configure Options: SelectMap (selected), Jtag

System Generator Design Name: tut1/Subsystem

XSG Version: V2.2

Xilinx FPGA Part: XCV600E-6-HQ240, XCV2000E-6-FG680 (selected), XCV200E-6-FG256, XCV300E-6-BG432

CLK pin: D21

CLK pad: BUFGP

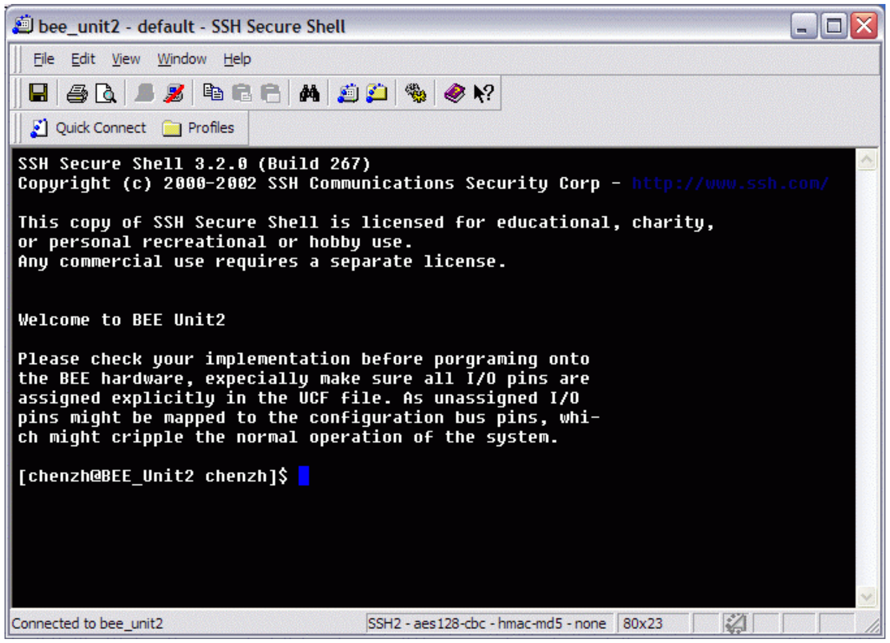
ISE Design Flow Choice: Implementation Only, Configure Only, Complete Build, Back Annotate Pin Loc to UCF, Behavioral VHDL Simulation (selected), Post Translate VHDL Simulation, Post PAR VHDL Simulation

Current Section

- Section 1: Creating and Simulating a Model
- Section 2: Tag and Route the Design conforming to BEE architecture
- Section 3: Generating FPGA Bit File Using BEE_ISE Tool
- Section 4: Download and Emulate Design on BEE Hardware
- Section 5: ASIC flow basics

Connect to BEE

- Log on to one of the four BEE systems using SSH
  - Bee_unit#.eecs.berkeley.edu (# can be 1,2,3,4)
- Access to BEE systems requires a BEE-specific account.
  - See the BEE system administrator to create an account.

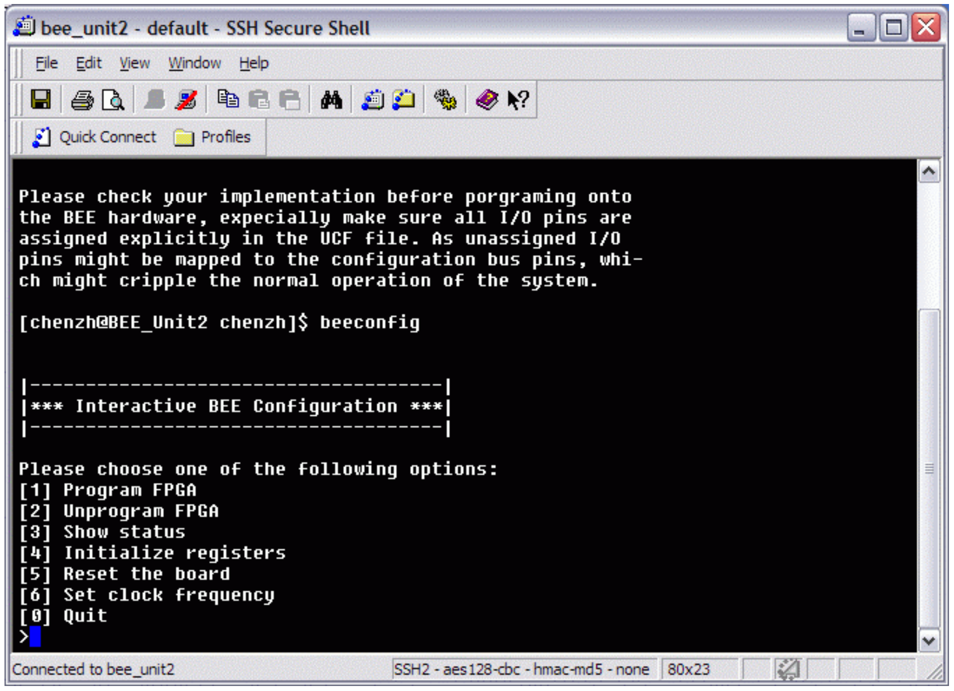

bee_unit2 - default - SSH Secure Shell
File Edit View Window Help
Quick Connect
Profiles

```
SSH Secure Shell 3.2.0 (Build 267)
Copyright (c) 2000-2002 SSH Communications Security Corp - http://www.ssh.com/

This copy of SSH Secure Shell is licensed for educational, charity,
or personal recreational or hobby use.
Any commercial use requires a separate license.


Welcome to BEE Unit2

Please check your implementation before porgraming onto
the BEE hardware, expecially make sure all I/O pins are
assigned explicitly in the UCF file. As unassigned I/O
pins might be mapped to the configuration bus pins, whi-
ch might cripple the normal operation of the system.

[chenzh@BEE_Unit2 chenzh]$
```

Connected to bee_unit2 | SSH2 - aes128-cbc - hmac-md5 - none | 80x23

Run BEEConfig

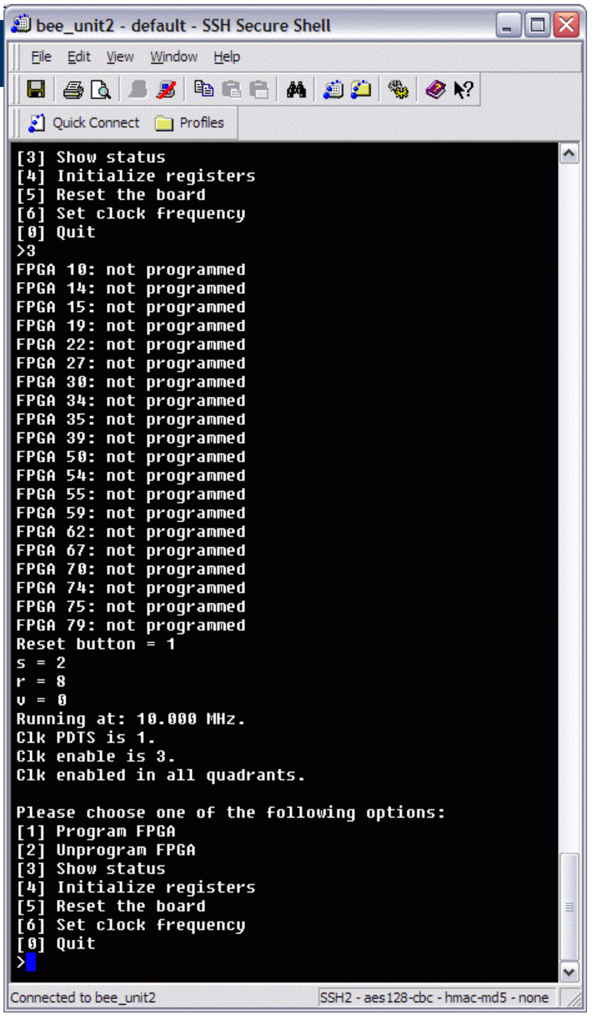
- Type “beeconfig” in the shell prompt
- This is a menu driven interface for configuration and runtime control of the BEE system


bee_unit2 - default - SSH Secure Shell
File Edit View Window Help
Quick Connect Profiles
Connected to bee_unit2
SSH2 - aes128-cbc - hmac-md5 - none
80x23

```
Please check your implementation before porgraming onto
the BEE hardware, expecially make sure all I/O pins are
assigned explicitly in the UCF file. As unassigned I/O
pins might be mapped to the configuration bus pins, whi-
ch might cripple the normal operation of the system.

[chenzh@BEE_Unit2 chenzh]$ beeconfig


|--------------------------------------|
|*** Interactive BEE Configuration ***|
|--------------------------------------|

Please choose one of the following options:
[1] Program FPGA
[2] Unprogram FPGA
[3] Show status
[4] Initialize registers
[5] Reset the board
[6] Set clock frequency
[0] Quit
>
```

Show BEE System Status

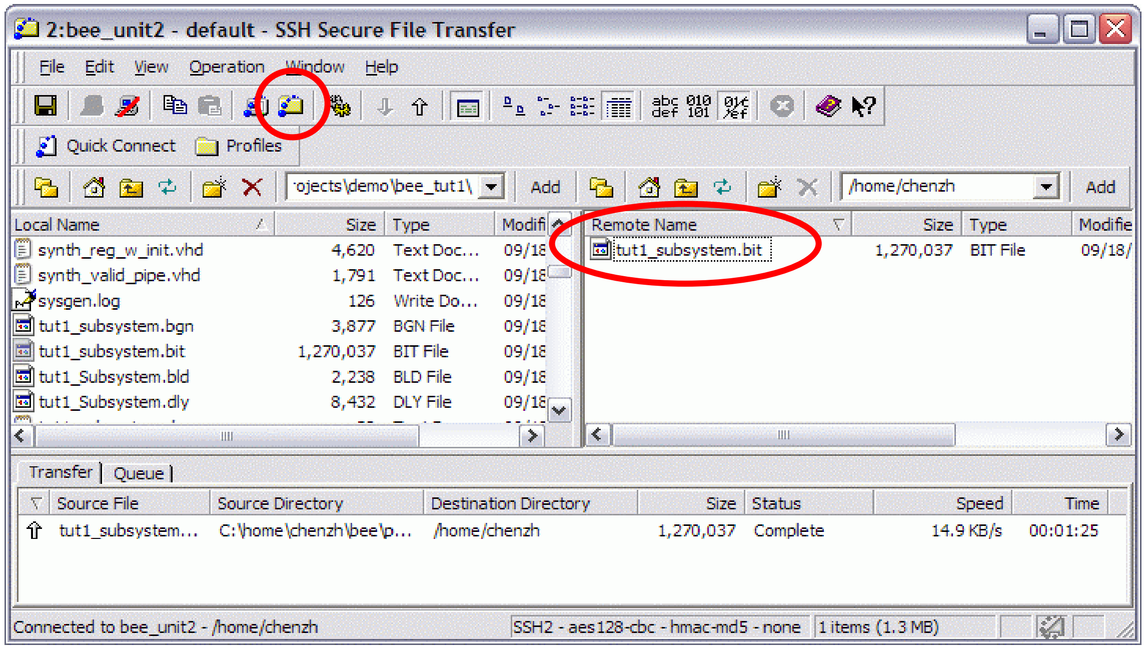
- Choose option 3 from the main menu
- Programming status of each FPGA and the clock rate are displayed


bee_unit2 - default - SSH Secure Shell
File Edit View Window Help
Quick Connect
Profiles
Connected to bee_unit2
SSH2 - aes128-cbc - hmac-md5 - none

```
[3] Show status
[4] Initialize registers
[5] Reset the board
[6] Set clock frequency
[0] Quit
>3
FPGA 10: not programmed
FPGA 14: not programmed
FPGA 15: not programmed
FPGA 19: not programmed
FPGA 22: not programmed
FPGA 27: not programmed
FPGA 30: not programmed
FPGA 34: not programmed
FPGA 35: not programmed
FPGA 39: not programmed
FPGA 50: not programmed
FPGA 54: not programmed
FPGA 55: not programmed
FPGA 59: not programmed
FPGA 62: not programmed
FPGA 67: not programmed
FPGA 70: not programmed
FPGA 74: not programmed
FPGA 75: not programmed
FPGA 79: not programmed
Reset button = 1
s = 2
r = 8
v = 0
Running at: 10.000 MHz.
Clk PDTS is 1.
Clk enable is 3.
Clk enabled in all quadrants.

Please choose one of the following options:
[1] Program FPGA
[2] Unprogram FPGA
[3] Show status
[4] Initialize registers
[5] Reset the board
[6] Set clock frequency
[0] Quit
>
```

Upload Bit File

- Upload the bit file of the tut1 design to the BEE system through secure FTP
- The Bit file name should be “tut1_subsystem.bit”


2:bee_unit2 - default - SSH Secure File Transfer
File Edit View Operation Window Help
Quick Connect
Profiles
Add
Add

ojects\demo\bee_tut1\

Local Name	Size	Type	Modifi
synth_reg_w_init.vhd	4,620	Text Doc...	09/18
synth_valid_pipe.vhd	1,791	Text Doc...	09/18
sysgen.log	126	Write Do...	09/18
tut1_subsystem.bgn	3,877	BGN File	09/18
tut1_subsystem.bit	1,270,037	BIT File	09/18
tut1_Subsystem.bld	2,238	BLD File	09/18
tut1_Subsystem.dly	8,432	DLY File	09/18

/home/chenzh

Remote Name	Size	Type	Modifie
tut1_subsystem.bit	1,270,037	BIT File	09/18/

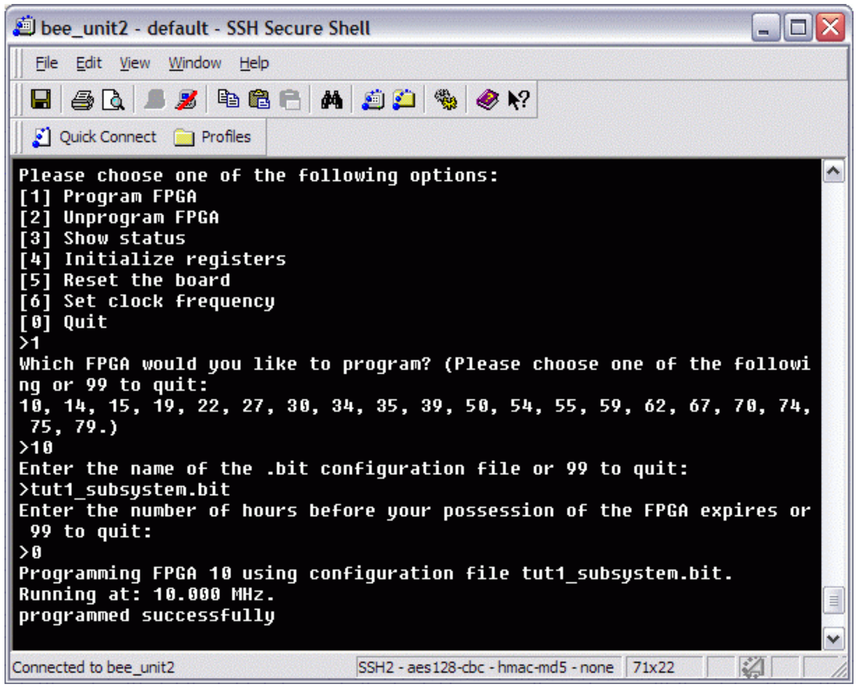
Transfer | Queue

	Source File	Source Directory	Destination Directory	Size	Status	Speed	Time
⇧	tut1_subsystem...	C:\home\chenzh\bee\p...	/home/chenzh	1,270,037	Complete	14.9 KB/s	00:01:25

Connected to bee_unit2 - /home/chenzh | SSH2 - aes128-cbc - hmac-md5 - none | 1 items (1.3 MB)

Program the FPGA

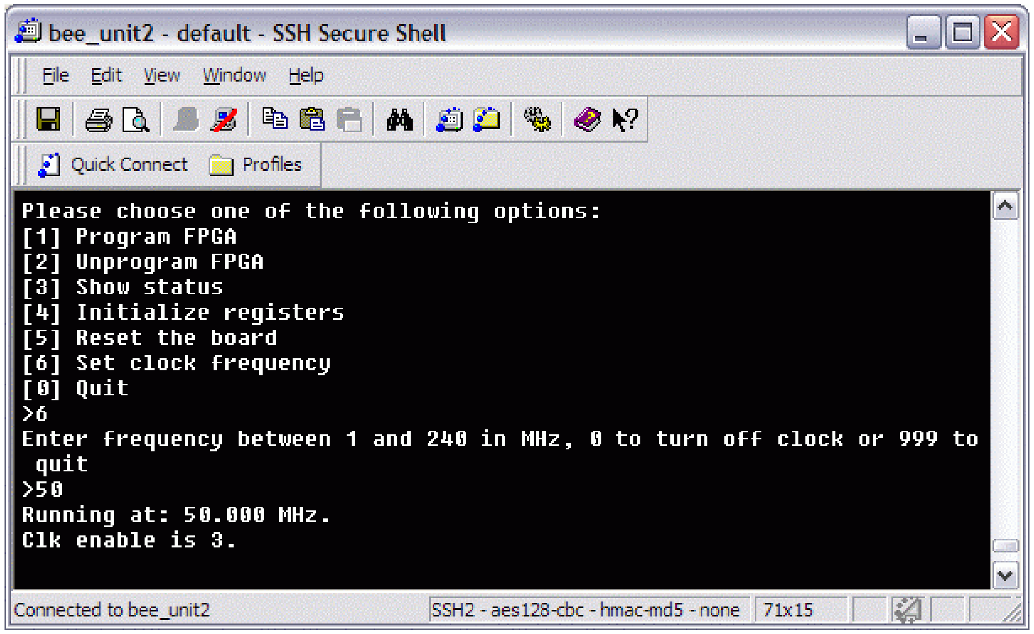
- Select option 1 from the main menu
- Choose the FPGA to be programmed, “10” in this case
- Enter 1 for one hour till expatriation
- After successful programming, the LED on the BEE system FPGA10 will flash


bee_unit2 - default - SSH Secure Shell
File Edit View Window Help
Quick Connect Profiles
Connected to bee_unit2
SSH2 - aes128-cbc - hmac-md5 - none
71x22

```
Please choose one of the following options:
[1] Program FPGA
[2] Unprogram FPGA
[3] Show status
[4] Initialize registers
[5] Reset the board
[6] Set clock frequency
[0] Quit
>1
Which FPGA would you like to program? (Please choose one of the followi
ng or 99 to quit:
10, 14, 15, 19, 22, 27, 30, 34, 35, 39, 50, 54, 55, 59, 62, 67, 70, 74,
 75, 79.)
>10
Enter the name of the .bit configuration file or 99 to quit:
>tut1_subsystem.bit
Enter the number of hours before your possession of the FPGA expires or
 99 to quit:
>0
Programming FPGA 10 using configuration file tut1_subsystem.bit.
Running at: 10.000 MHz.
programmed successfully
```

Change BEE Clock Frequency

- Select option 6 from main menu
- Enter a new clock frequency in MHz
- The LED on BEE system will be flash at different rate now


bee_unit2 - default - SSH Secure Shell
File Edit View Window Help
Quick Connect
Profiles
Connected to bee_unit2
SSH2 - aes128-cbc - hmac-md5 - none
71x15

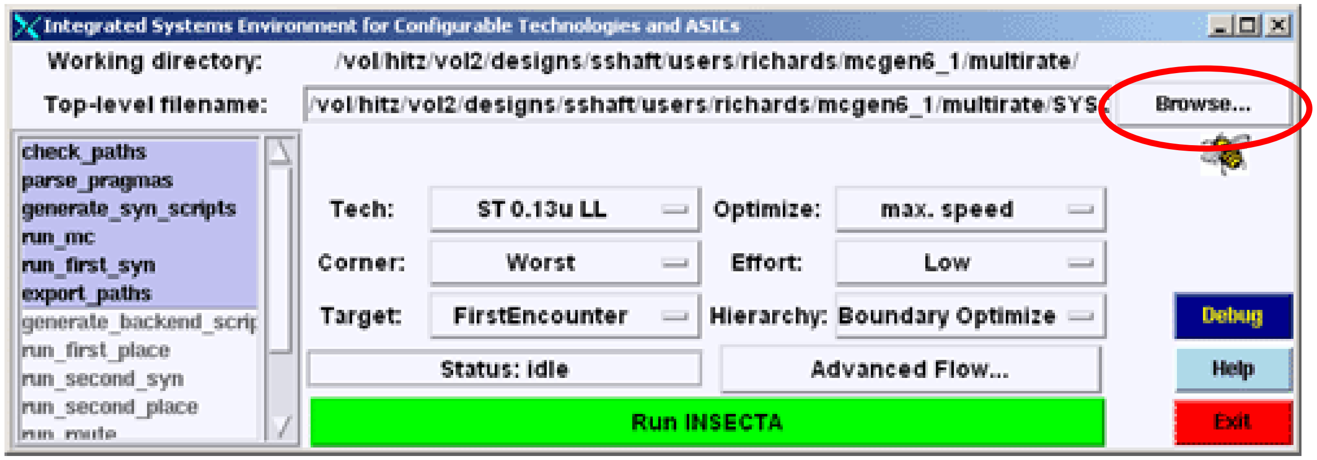
```
Please choose one of the following options:
[1] Program FPGA
[2] Unprogram FPGA
[3] Show status
[4] Initialize registers
[5] Reset the board
[6] Set clock frequency
[0] Quit
>6
Enter frequency between 1 and 240 in MHz, 0 to turn off clock or 999 to
 quit
>50
Running at: 50.000 MHz.
Clk enable is 3.
```

Current Section

- Section 1: Creating and Simulating a Model
- Section 2: Tag and Route the Design conforming to BEE architecture
- Section 3: Generating FPGA Bit File Using BEE_ISE Tool
- Section 4: Download and Emulate Design on BEE Hardware
- Section 5: ASIC flow basics

Running INSECTA

- Start the Exceed program on the PC side
- Return to your SSH to Unix (sun450-1)
- Confirm that you current directory is still /tools/designs/<work_dir>/tut1
- $ insecta &
- Click “Browse…” in Insecta GUI and select /tools/designs/<where ever you work>/tut1/SYSGEN/tut1_subsystem.vhd
- **To get early power, area and speed estimates, highlight the first six entries, from check_paths to export_paths**


Integrated Systems Environment for Configurable Technologies and ASICs

Working directory: /vol/hitz/vol2/designs/sshaft/users/richards/mcgen6_1/multirate/

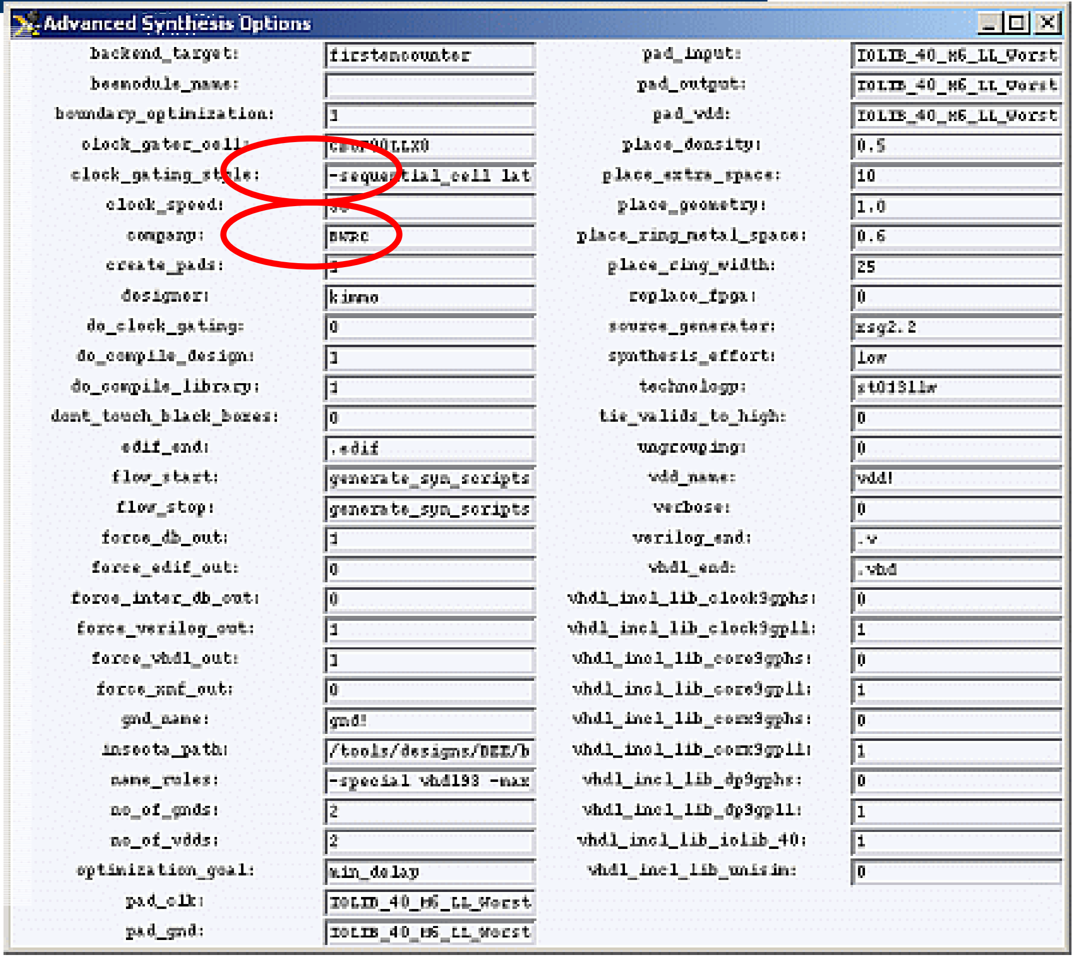
Top-level filename: /vol/hitz/vol2/designs/sshaft/users/richards/mcgen6_1/multirate/SYS

check_paths
parse_pragmas
generate_syn_scripts
run_mc
run_first_syn
export_paths
generate_backend_scrip
run_first_place
run_second_syn
run_second_place
run_route

Tech: ST 0.13u LL
Optimize: max. speed
Corner: Worst
Effort: Low
Target: FirstEncounter
Hierarchy: Boundary Optimize
Status: idle

Running INSECTA (cont'd)

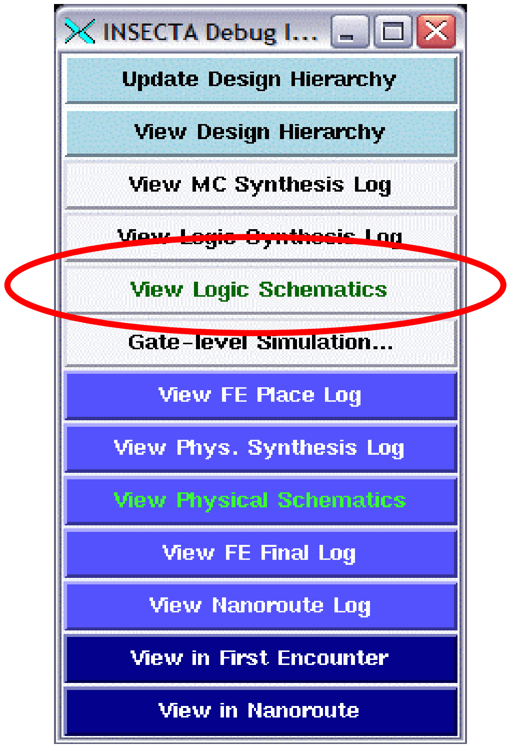
- **Press "Advanced Flow".**
  - This is a list of user-modifiable variables that control the design flow.
  - Most top level menus set defaults for these values
  - All settings will be saved in "insecta_save.tcl" on exit.
- **Set clock_speed to the desired period.**
  - Yes, it should be 'clock_period'
- **Set create_pads to 0 to turn off I/O pad insertion.**
- **Click "Run INSECTA".**
- **Enjoy the show or go get a cup of coffee…**


Advanced Synthesis Options

backend_target:	firstencounter	pad_input:	IOLIB_40_M6_LL_Worst
beemodule_name:		pad_output:	IOLIB_40_M6_LL_Worst
boundary_optimization:	1	pad_vdd:	IOLIB_40_M6_LL_Worst
clock_gater_cell:	[illegible]00LLX0	place_density:	0.5
clock_gating_style:	-sequential_cell lat	place_extra_space:	10
clock_speed:	[illegible]	place_geometry:	1.0
company:	BWRC	place_ring_metal_space:	0.6
create_pads:	1	place_ring_width:	25
designer:	kimmo	replace_fpga:	0
do_clock_gating:	0	source_generator:	xsg2.2
do_compile_design:	1	synthesis_effort:	low
do_compile_library:	1	technology:	st013llw
dont_touch_black_boxes:	0	tie_valids_to_high:	0
edif_end:	.edif	ungrouping:	0
flow_start:	generate_syn_scripts	vdd_name:	vdd!
flow_stop:	generate_syn_scripts	verbose:	0
force_db_out:	1	verilog_end:	.v
force_edif_out:	0	vhdl_end:	.vhd
force_inter_db_out:	0	vhdl_incl_lib_clock9gphs:	0
force_verilog_out:	1	vhdl_incl_lib_clock9gpll:	1
force_vhdl_out:	1	vhdl_incl_lib_core9gphs:	0
force_xnf_out:	0	vhdl_incl_lib_core9gpll:	1
gnd_name:	gnd!	vhdl_incl_lib_corx9gphs:	0
insecta_path:	/tools/designs/BEE/b	vhdl_incl_lib_corx9gpll:	1
name_rules:	-special vhdl93 -max	vhdl_incl_lib_dp9gphs:	0
no_of_gnds:	2	vhdl_incl_lib_dp9gpll:	1
no_of_vdds:	2	vhdl_incl_lib_iolib_40:	1
optimization_goal:	min_delay	vhdl_incl_lib_unisim:	0
pad_clk:	IOLIB_40_M6_LL_Worst		
pad_gnd:	IOLIB_40_M6_LL_Worst		

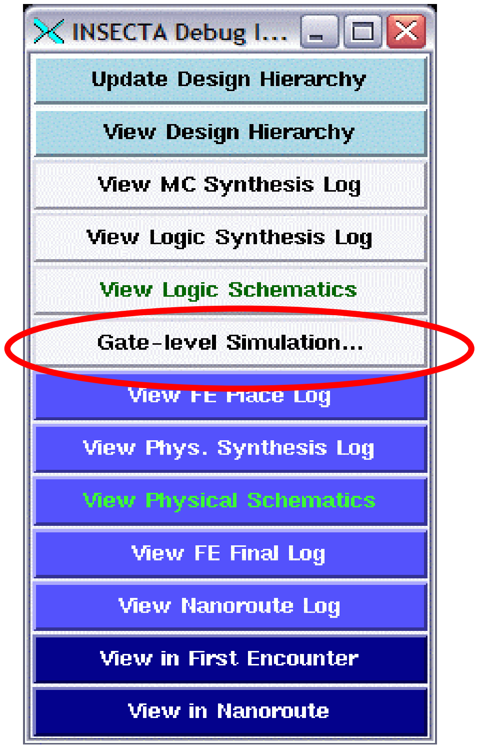
Checking your design

- Click on the “Debug” button to bring up the debug window
- Use the “View” options in the Debug interface to execute a variety of CAD tools and log file viewers.
  - Synthesis log: Synopsys dc_shell output
  - Logic Schematics: design_analyzer
  - Gate level Simulation: Modelsim
- For more INSECTA usage info, click “Help”
  - Detailed usage info about INSECTA



Gate-level VHDL Simulation

- Run ModelSim gate-level VHDL simulation by click on the “Gate-level Simulation …” button
- ModelSim windows will pop up and automatically run the same simulation as in Simulink.
- In the first couple of cycles, the VHDL simulation result will not match Simulink result; this is due to unknown initial value of components, such as registers.
- This mismatch is OK in this case, but more attention is needed if your design contains feedback loops, which require all registers to be reseted properly
- For real designs, you also need to check all the logs



Congratulations

- You have just finished the first BEE Design Flow tutorial lesson
- For more detailed instruction on how to use the tools introduced, please read the user's guide or the manual of the tools from the BEE web site