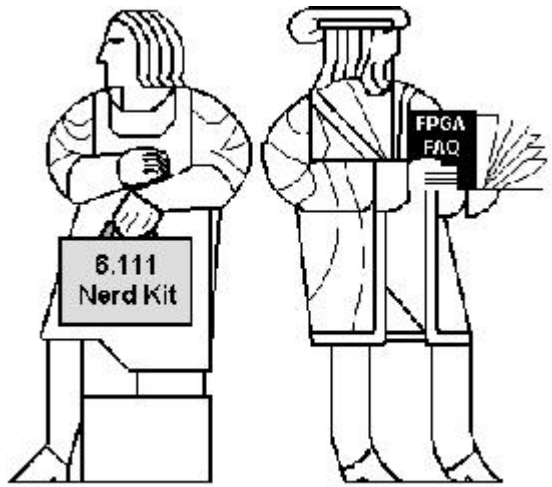
6.111 Lecture 7

Today:

Demo! (or die): An Electronic Lock

1. Design FSM
2. Implement in Verilog
3. Compile: Xilinx tool-chain
4. Program labkit



Demo!

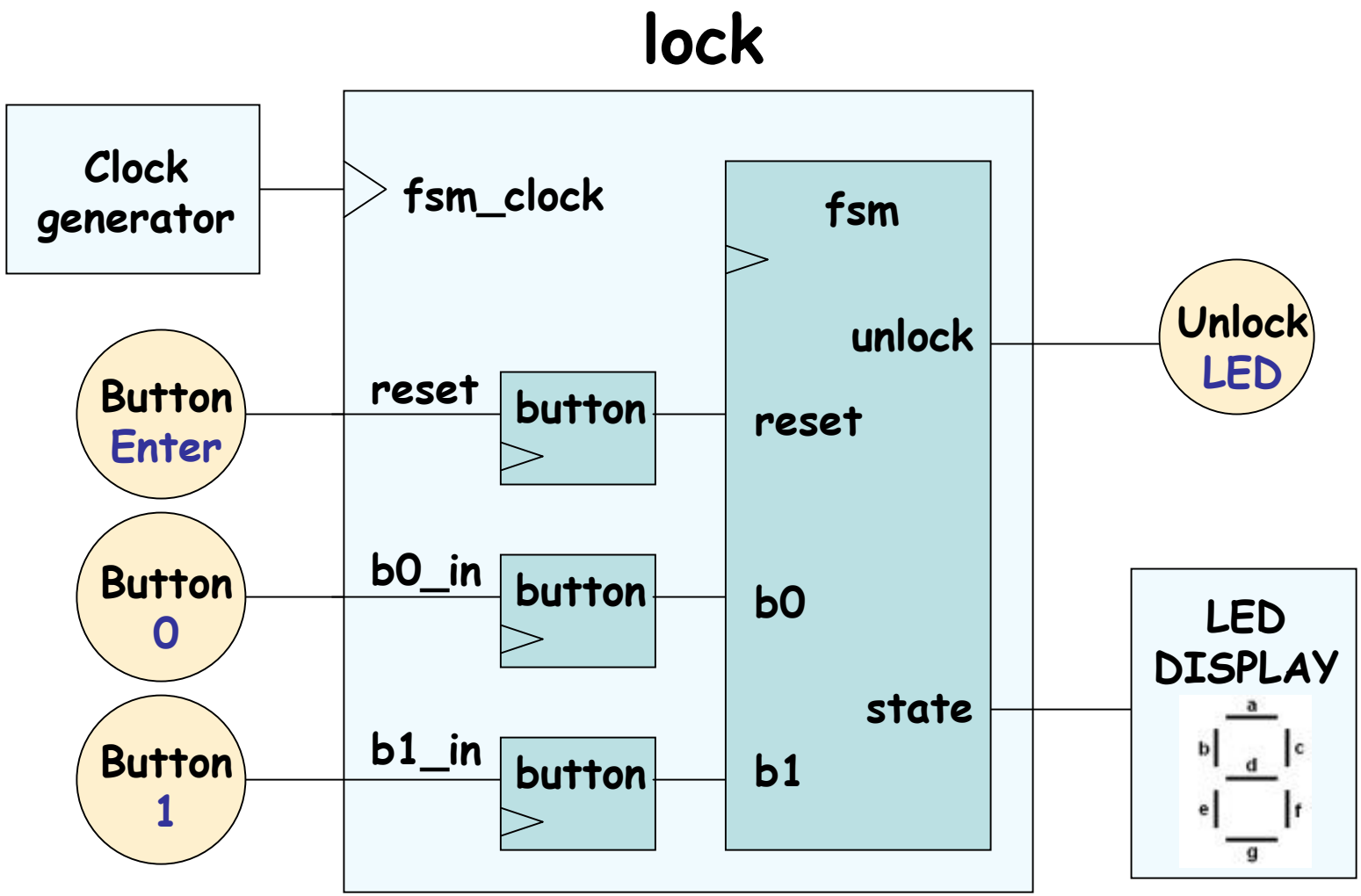
GOAL:

Build an electronic combination lock with a reset button, two number buttons (0 and 1), and an unlock output. The combination should be 01011.

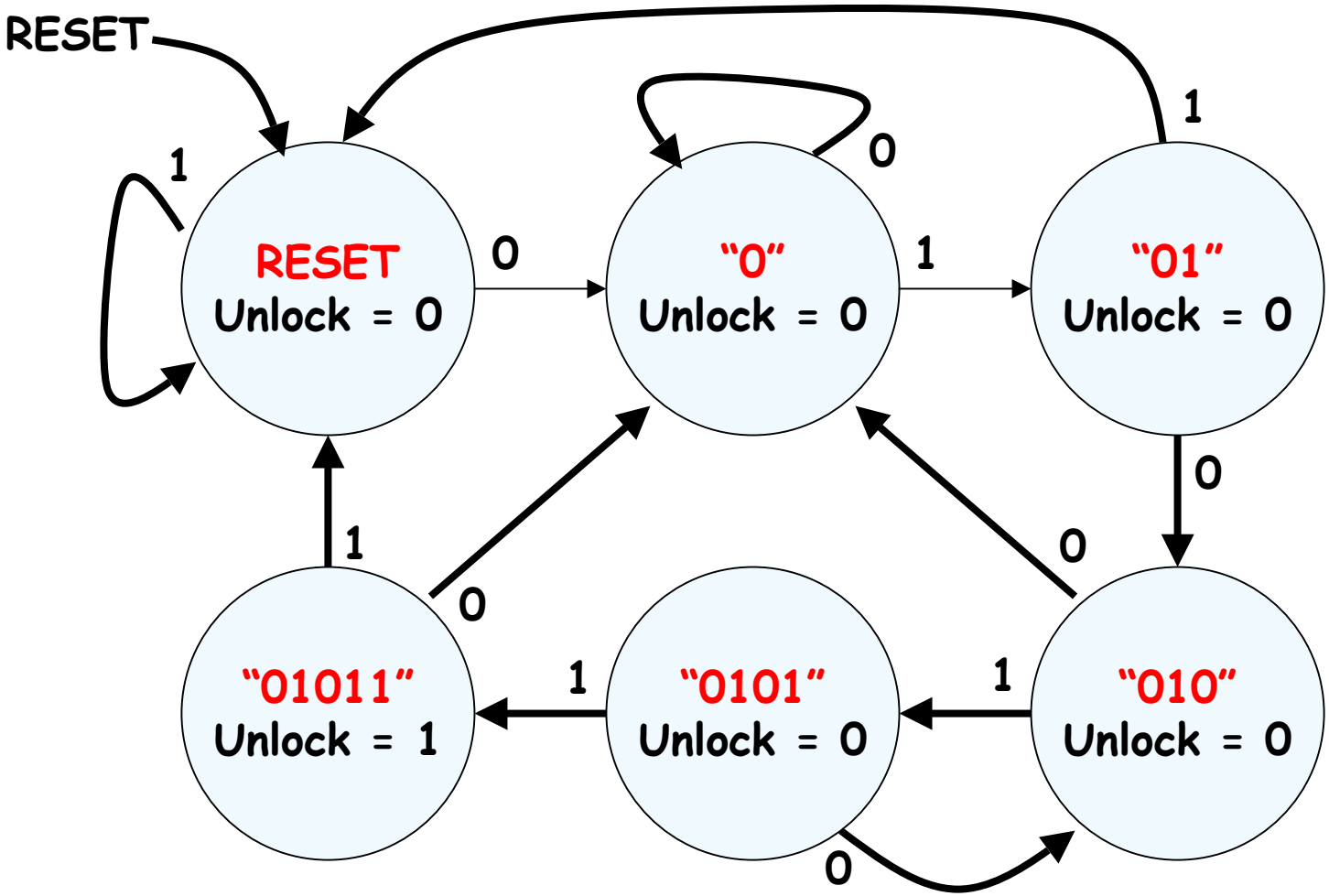
STEPS:

1. Design lock FSM (block diagram, state transitions)
2. Write Verilog module(s) for FSM
3. Use Xilinx ISE6.3 (synthesis, simulation)
4. Program FGPA, give it a whirl!

Step 1A: Block Diagram



Step 1B: State transition diagram



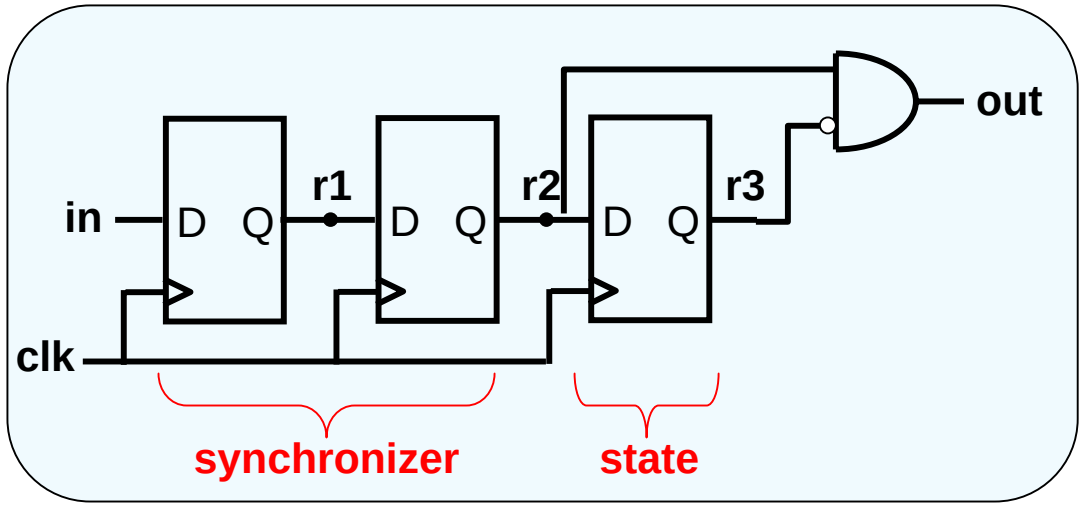
6 states → 3 bits

Step 2: Write Verilog

```
module lock(clk,reset_in,b0_in,b1_in,out);

  input clk,reset,b0_in,b1_in;
  output out;

  // synchronize push buttons, convert to pulses

  // implement state transition diagram
  reg [2:0] state;
  always @ (posedge clk)
  begin
    state <= ???;
  end

  // generate output
  assign out = ???;

  // debugging?
endmodule
```

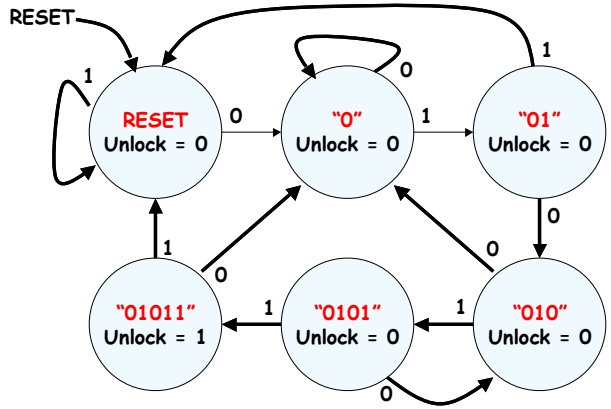
Step 2A: Synchronize buttons

```
// button -- push button synchronizer and level-to-pulse converter
// OUT goes high for one cycle of CLK whenever IN makes a
// low-to-high transition.

module button(clk,in,out);
  input clk;
  input in;
  output out;

  reg r1,r2,r3;
  always @ (posedge clk)
  begin
    r1 <= in;    // first reg in synchronizer
    r2 <= r1;    // second reg in synchronizer, output is in sync!
    r3 <= r2;    // remembers previous state of button
  end

  // rising edge = old value is 0, new value is 1
  assign out = ~r3 & r2;
endmodule
```



Step 2B: state transition diagram

```
parameter S_RESET = 0; // state assignments
parameter S_0 = 1;
parameter S_01 = 2;
parameter S_010 = 3;
parameter S_0101 = 4;
parameter S_01011 = 5;

reg [2:0] state;
always @ (posedge clk)
begin      // implement state transition diagram
  if (reset) state <= S_RESET;
  else case (state)
    S_RESET: state <= b0 ? S_0    : b1 ? S_RESET : state;
    S_0:     state <= b0 ? S_0    : b1 ? S_01    : state;
    S_01:    state <= b0 ? S_010  : b1 ? S_RESET : state;
    S_010:   state <= b0 ? S_0    : b1 ? S_0101  : state;
    S_0101:  state <= b0 ? S_010  : b1 ? S_01011 : state;
    S_01011: state <= b0 ? S_0    : b1 ? S_RESET : state;
    default: state <= S_RESET;  // handle unused states
  endcase
end
```



Step 2C: generate output

```
// it's a Moore machine!  Output only depends on current state

assign out = (state == S_01011);
```

Step 2D: debugging?

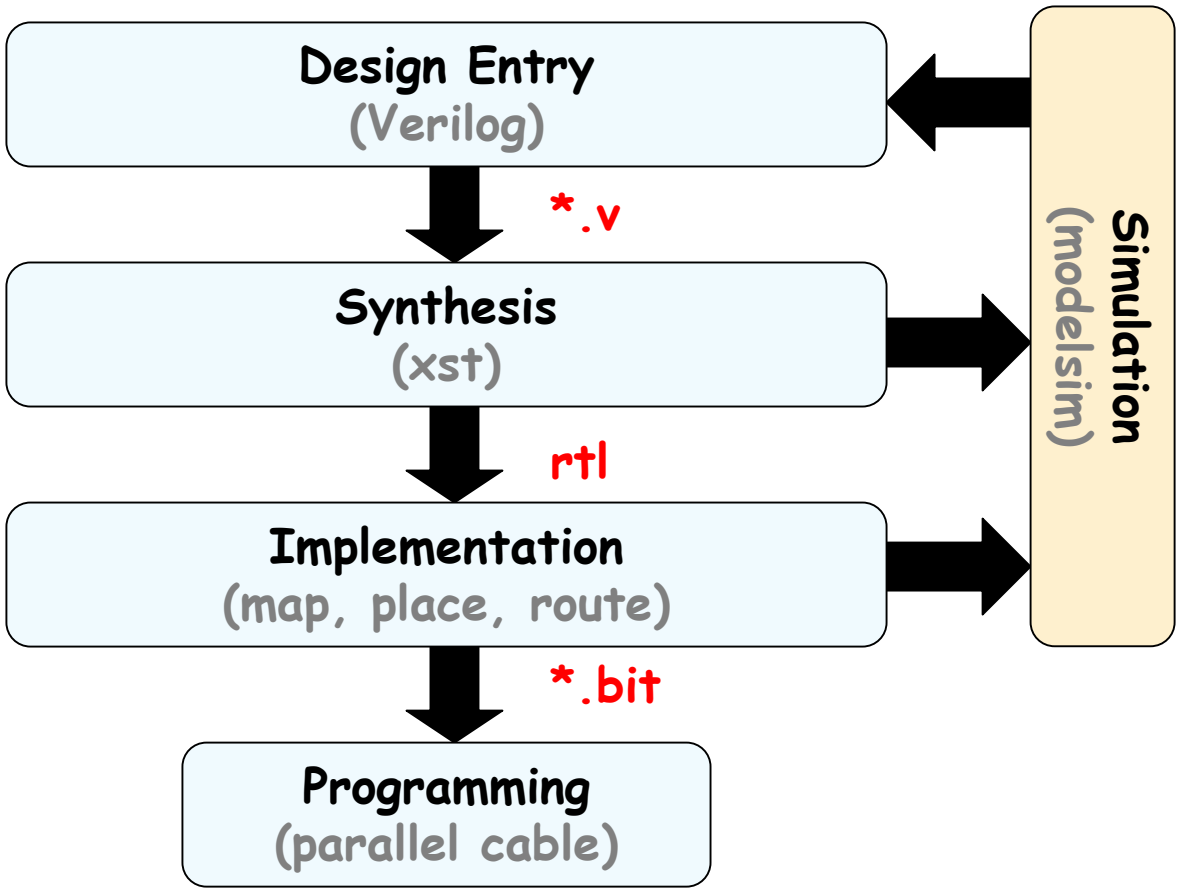
```
// hmmm.  What would be useful to know?  Current state?

assign hex_display = {1'b0,state[2:0]};
```

Step 2: final Verilog implementation

```
module lock(clk,reset_in,b0_in,b1_in,out, hex_display);

  input clk,reset,b0_in,b1_in;
  output out; output[3:0] hex_display;

  wire reset, b0, b1; // synchronize push buttons, convert to pulses
  button b_reset(clk,reset_in,reset);
  button b_0(clk,b0_in,b0);
  button b_1(clk,b1_in,b1);

  parameter S_RESET = 0; parameter S_0 = 1; // state assignments
  parameter S_01 = 2;  parameter S_010 = 3;
  parameter S_0101 = 4;  parameter S_01011 = 5;

  reg [2:0] state;
  always @ (posedge clk)
  begin                          // implement state transition diagram
    if (reset) state <= S_RESET;
    else case (state)
      S_RESET: state <= b0 ? S_0   : b1 ? S_RESET : state;
      S_0:     state <= b0 ? S_0   : b1 ? S_01    : state;
      S_01:    state <= b0 ? S_010 : b1 ? S_RESET : state;
      S_010:   state <= b0 ? S_0   : b1 ? S_0101  : state;
      S_0101:  state <= b0 ? S_010 : b1 ? S_01011 : state;
      S_01011: state <= b0 ? S_0   : b1 ? S_RESET : state;
      default: state <= S_RESET;         // handle unused states
    endcase

 assign out = (state == S_01011);       // assign output: Moore machine
 assign hex_display = {1'b0,state};      // debugging
endmodule
```

Step 3: Synthesis & Simulation

- Fall '05 6.111: This year, we will use the Xilinx toolset
- Software: ISE 6.3i (windows / linux)



Step 3A: Load source file lock.v

Xilinx - Project Navigator - C:\ike\6.111\LAB\lecture7\demo1\demo1.ise - [lock.v]

File Edit View Project Source Process Simulation Window Help

switch

Sources in Project:

- demo1.ise
- xc2v6000-4bf957
  - lock (lock.v)
    - button (lock.v)

New Source...
Add Source... Insert
Add Copy of Source... Shift+Insert
Remove Delete
Move to Library...
Open
Toggle Paths
Properties...

Module ...

Processes for Source: "lock"

- Add Existing Source
- Create New Source
- View Design Summary
- Design Utilities
- User Constraints
- Synthesize - XST
- Implement Design
- Generate Programming File
  - Programming File Generation Repor
  - Generate PROM, ACE, or JTAG File
  - Configure Device (iMPACT)

Process View

```
1
2  module lock(clk, reset_in, b0_in, b1_in, out, hex_display);
3      input clk, reset_in, b0_in,b1_in;
4      output out; output [3:0] hex_display;
5
6      // synchronize push buttons, convert to pulses
7
8      wire reset, b0, b1;
9      button b_reset(clk,reset_in,reset);
0      button b_0(clk,b0_in,b0);
1      button b_1(clk,b1_in,b1);
2
3      // implement state transition diagram
4
5     parameter S_RESET = 0;
6     parameter S_0 = 1;
7     parameter S_01 = 2;
18    parameter S_010 =3;
19    parameter S_0101 = 4;
20    parameter S_01011 = 5;
21
22    reg [2:0] state;
23
24    always @ (posedge clk)
25    begin
26      if (reset) state <= S_RESET;
27      else case (state)
28        S_RESET: state <= b0 ? S_0   : ( b1 ? S_RESET : state
29        S_0:     state <= b0 ? S_0   : ( b1 ? S_01 : state );
30        S_01:    state <= b0 ? S_010 : ( b1 ? S_RESET : state
```

lock.v

Step 3B: Compile/Synthesize

Xilinx - Project Navigator - C:\ike\6.111\LAB\lecture7\demo1\demo1.ise - [lock.v]

File Edit View Project Source Process Simulation Window Help

switch

Sources in Project:

- demo1.ise
- xc2v6000-4bf957
  - lock (lock.v)
    - button (lock.v)

Module ... | Snapshot ... | Library View

Processes for Source: "lock"

- Add Existing Source
- Create New Source
- View Design Summary
- Design Utilities
- User Constraints
- Synthesize - XST
- Implement Design
- Generate Programming File

Process View

```
1
2  module lock(clk, reset_in, b0_in, b1_in, out, hex_displa
3      input clk, reset_in, b0_in,b1_in;
4      output out; output [3:0] hex_display;
5
6      // synchronize push buttons, convert to pulses
7
8      wire reset, b0, b1;
9      button b_reset(clk,reset_in,reset);
10     button b_0(clk,b0_in,b0);
11     button b_1(clk,b1_in,b1);
12
13     // implement state transition diagram
14
15    parameter S_RESET = 0;
16    parameter S_0 = 1;
17    parameter S_01 = 2;
18    parameter S_010 =3;
19    parameter S_0101 = 4;
20    parameter S_01011 = 5;
21
22    reg [2:0] state;
```

lock.v

```
Minimum period: 3.479ns (Maximum Frequency: 287.439MHz)
Minimum input arrival time before clock: 1.712ns
Maximum output required time after clock: 6.987ns
Maximum combinational path delay: No path found

=========================================================================
```

Console | Find in Files | Errors | Warnings

Step 3B: Create testbench

Xilinx - Project Navigator - C:\ike\6.111\LAB\lecture7\demo1\demo1.ise - [lock.v]

File Edit View Project Source Process Simulation Window Help

switch

Sources in Project:

- demo1.ise
- xc2v6000-4bf957
  - lock (lock.v)
    - button (lock.v)

Module ... Snapshot ...

```
1
2  module lock(clk, reset_in, b0_in, b1_in, out, hex_displ
3      input clk, reset_in, b0_in,b1_in;
                                                  play;

                                                  vert to pulses

                                                  );

                                                  agram
```

Processes for Source: "lock"

- Add Existing Source
- Create New Source
- View Design Summary
- Design Utilities
- User Constraints
- Synthesize - XST
- Implement Design
- Generate Programming

Process View

New Source

- BMM File
- Implementation Constraints File
- IP (CoreGen & Architecture Wizard)
- MEM File
- Schematic
- State Diagram
- Test Bench Waveform
- User Document
- Verilog Module
- Verilog Test Fixture
- VHDL Library
- VHDL Module
- VHDL Package
- VHDL Test Bench

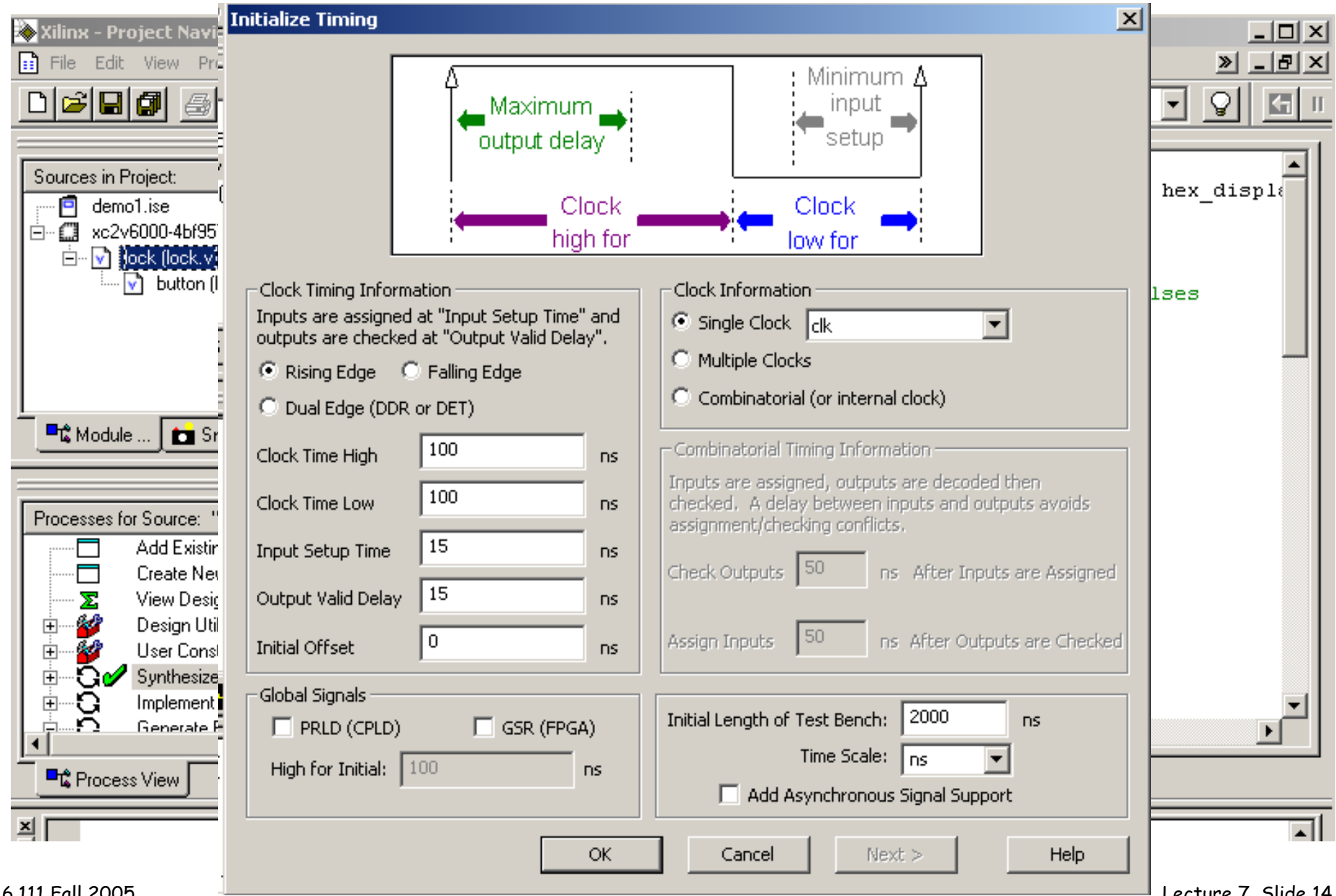
File Name: lock_tb

Location: C:\ike\6.111\LAB\lecture7\der ...

Add to project

< Back | Next > | Cancel | Help

Step 3B: Create testbench



Step 3B: Generate Simulation Results

Xilinx - Project Navigator - C:\ike\6.111\LAB\lecture7\demo1\demo1.ise - [lock_tb.tbw]

File Edit View Project Source Process Simulation Test Bench Window Help

switch

Sources in Project:
- demo1.ise
- xc2v6000-4bf957
  - lock (lock.v)
    - lock_tb (lock_tb.tbw)
    - button (lock.v)

Modu... Snap... Librar...

Processes for Source: "lock_tb"
- Add Test Bench To Projec
- Xilinx ISE Simulator
  - Generate Expected Simulation Results
  - Simulate Behaviora
  - Simulate Post-Place

Process View Hierarchy - lo...

End Time: 4000 ns

100 ns 500 ns 900 ns 1300 ns 1700 ns 2100 ns 2500 ns 2900 ns 3300 ns 3700 ns

clk	0	
b0_in	0	
b1_in	0	
reset_in	0	
out	1	x
hex_display[...	5	0 4... 0 1 2 3 4 5

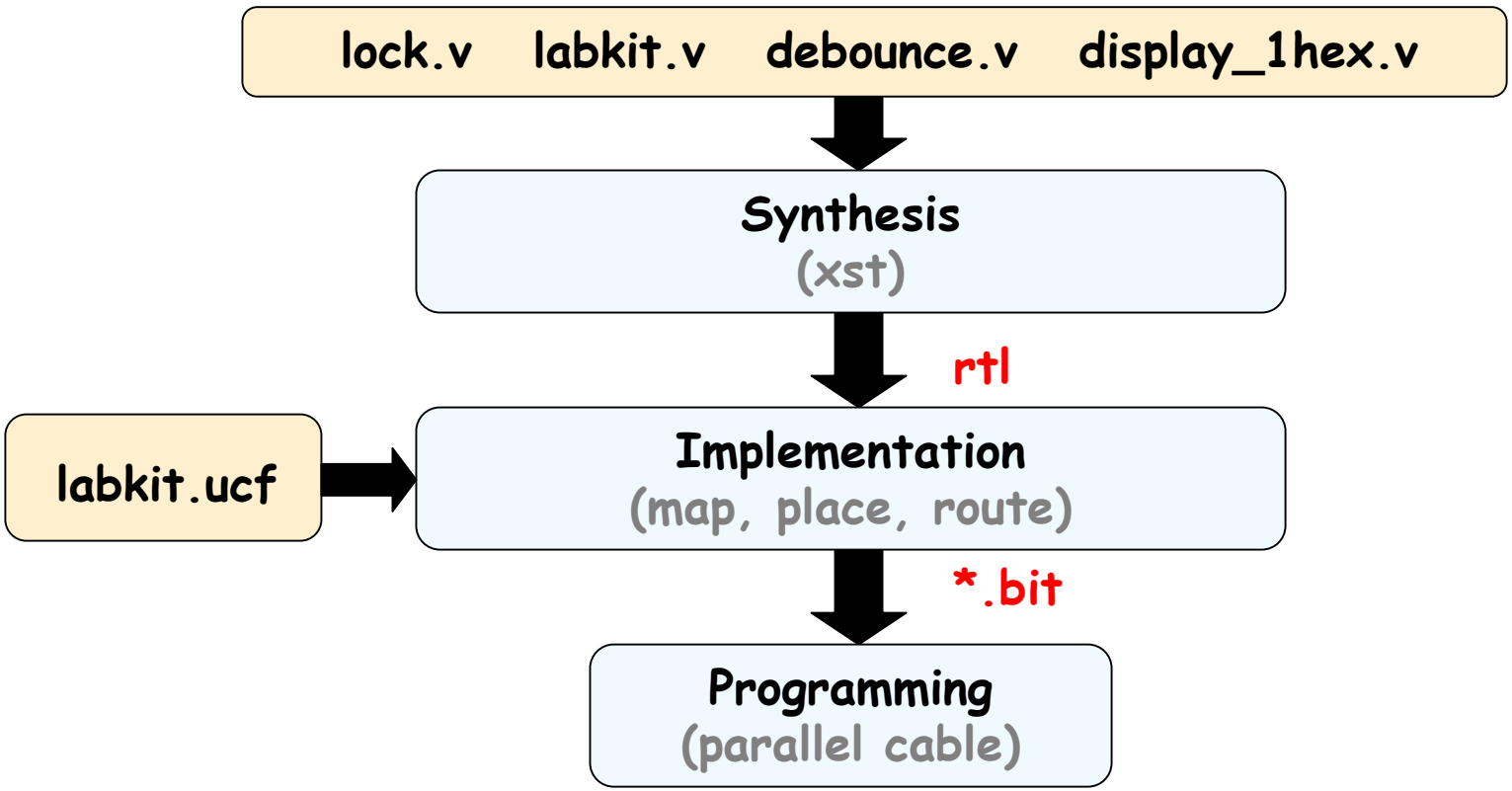
lock.v lock_tb.tbw

```
Finished circuit initialization process.
Success! Annotation Simulation Complete.
Stopped at time : 4.200 us : File "C:/ike/6.111/LAB/lecture7/demo1/lock_tb.ant" Line 74

Compiling verilog file "c:\ike\6.111\lab\lecture7\demo1\lock.v"
```

Console Find in Files Errors Warnings

Step 4: Implementation – Program FPGA

- Pin assignments: User constraints file
- Labkit: peripherals definitions
- Optimization: Placing and Routing



Step 4: Implementation – Program FPGA

- Pin assignments: User constraints file

labkit.ucf

Xilinx PACE - C:\ike\6.111\LAB\lecture7\demo1\labkit.ucf

File Edit View IOBs Areas Tools Window Help

Design Browser

- I/O Pins
- Global Logic
- Logic

Design Object List - I/O Pins

I/O Name	I/O Direction	Loc	Bank	
ac97_bit_clock	Input	ah24	BANK5	
ac97_sdata_in	Input	aj24	BANK5	
ac97_sdata_out	Output	ac18	BANK5	LVD
ac97_synch	Output	ac17	BANK5	LVD
analyzer1_clock	Output	G2	BANK2	LVT
analyzer1_data<0>	Output	R2	BANK2	LVT
analyzer1_data<1>	Output	R1	BANK2	LVT

#	Group	Loc	I/O Std.
8	vga_out_red		LVTTL
8	vga_out_green		LVTTL
8	vga_out_blue		LVTTL
10	tv_out_ycrcb		LVDCI_33
20	tv_in_ycrcb		
36	ram0_data		LVDCI_33

Package Pins for xc2v6000-4-bf957 (Flight Time)

Top View

Package View | Architecture View

The 6.111 Labkit: Subsystems

Nathan Ickes

Clocks

27 MHz
2 user inputs

Xilinx Virtex-II FPGA

XC2V6000-6BF957

6M gates
1M RAM
684 I/O pads

0.15μm 8 layer CMOS

Memory

4Mx36 ZBT SRAM
128 mbit Flash ROM

General I/O

9 buttons
8 switches
8 LED's
ps/2 kbd & mouse
RS-232 serial

Audio

LM4550 AC'97
Stereo In/Out
Headphone/mic

Video Output

AD7194
NTSC encoder

VGA Video

AD7125
Tripple-out DAC
1024 x 768 RGB

Display

16 char
5x7 dots

Video Input

ADV7185
NTSC decoder

Step 4A: FPGA Device Assignment

Xilinx - Project Navigator - C:\ike\6.111\LAB\lecture7\demo1\demo1.ise - [labkit.v *]

File Edit View Project Source Process Simulation Window Help

switch

Sources in Project:

- demo1.ise
- xc2v6000-4bf957
  - labkit (labkit.v)
    - labkit.ucf
    - debounce (c
    - display_1he:

Modu... Snap...

Processes for Source: "labk

- Add Existing S
- Create New Sc
- User Constrain

```
285    assign user2 = 32'hZ;
286    assign user3 = 32'hZ;
287    assign user4 = 32'hZ;
```

dy are inputs

Project Properties

Project Properties

Property Name	Value
Device Family	Virtex2
Device	xc2v6000
Package	bf957
Speed Grade	-4
Top-Level Module Type	HDL
Synthesis Tool	XST (VHDL/Verilog)
Simulator	ISE Simulator
Generated Simulation Language	Verilog

OK Cancel Default Help

Speed: -4

Virtex 2: xc2v6000

Package: bf957

Step 4B: Add labkit files

Xilinx - Project Navigator - C:\ike\6.111\LAB\lecture7\demo1\demo1.ise - [labkit.v *]

File Edit View Project Source Process Simulation Window Help

Sources in Project:

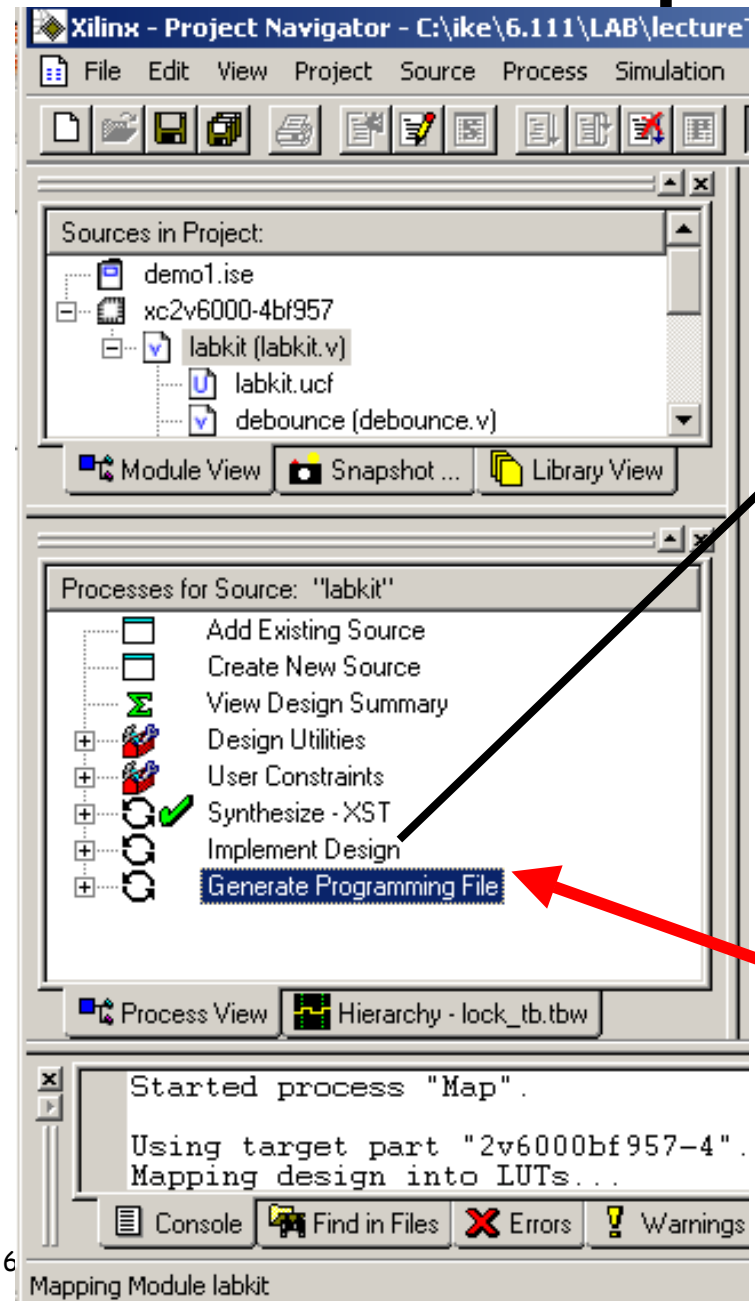
- demo1.ise
- xc2v6000-4bf957
  - labkit (labkit.v)
    - labkit.ucf
    - debounce (debounce.v)
    - display_1hex (display_1hex_v2.v)
    - lock (lock.v)
      - lock_tb (lock_tb.tbw)
      - button (lock.v)

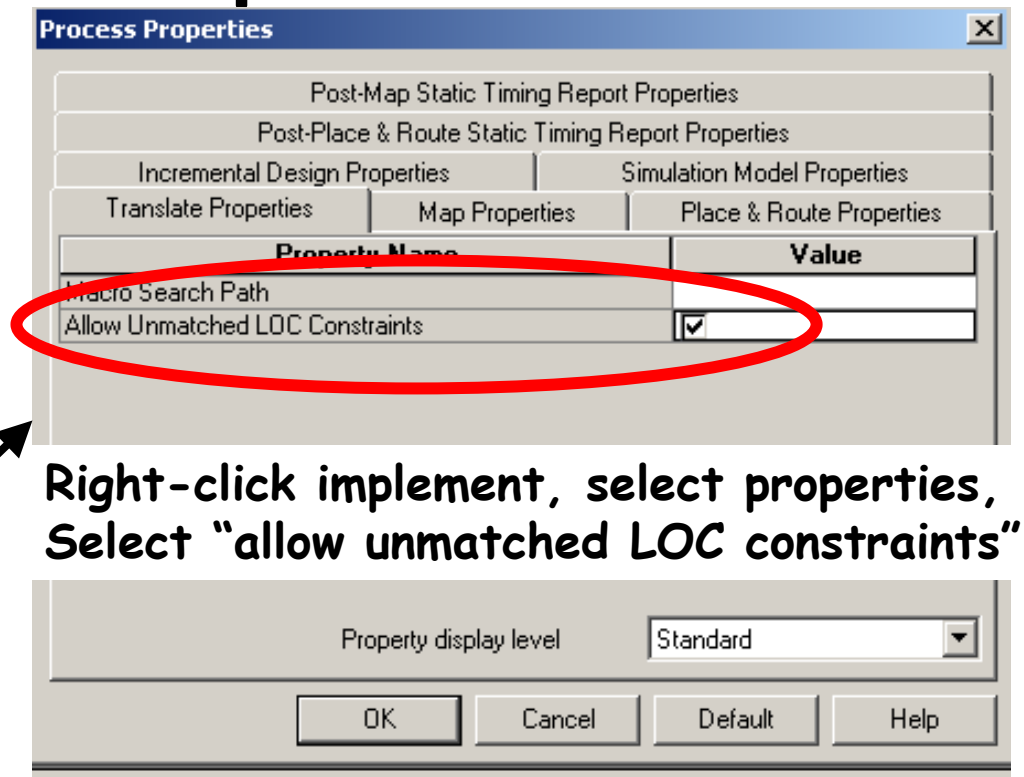
New Source...
Add Source... Insert
Add Copy of Source... Shift+Insert
Remove Delete
Move to Library...
Open
Toggle Paths
Properties...

Module View | Snapshot ... | Library View

```
285    assign user2 =
286    assign user3 =
287    assign user4 =
299
300    // Logic Analyz
```

Step 4C: Implement





Right-click implement, select properties, Select "allow unmatched LOC constraints"

Double-click here to implement design, and create the <u>labkit.bit</u> file

Step 4C: Implement

- Useful reports: Resource Utilization, Timing, RTL diagram

Xilinx - Project Navigator - C:\ike\6.111\LAB\lecture7\demo1\demo1.ise - [Design Summary]

File Edit View Project Source Process Simulation Window Help

switch

Sources in Project:

- demo1.ise
- xc2v6000-4bf957
  - labkit (labkit.v)
    - labkit.ucf

Module ... | Snapshot... | Library ...

Processes for Source: "labkit"

- Add Existing Source
- Create New Source
- View Design Summary
- Design Utilities
- User Constraints
- Synthesize - XST
- Implement Design
  - Translate
  - Map
  - Place & Route

Process View | Hierarchy - lock_tb.tbw

Design Overview for labkit

Property	Value
Project Name:	c:\ike\6.111\lab\lecture7\demo1
Target Device:	xc2v6000
Report Generated:	Sunday 09/25/05 at 16:22
Printable Summary (View as HTML)	labkit_summary.html

Device Utilization Summary

Logic Utilization	Used	Available	Utilization	Note(
Number of Slice Flip Flops:	127	67,584	1%	
Number of 4 input LUTs:	161	67,584	1%	
Logic Distribution:				
Number of occupied Slices:	142	33,792	1%	
Number of Slices containing only related logic:	142	142	100%	
Number of Slices containing unrelated logic:	0	142	0%	
Total Number 4 input LUTs:	198	67,584	1%	
Number used as logic:	161			
Number used as a route-thru:	37			
Number of bonded IOBs:	576	684	84%	

Performance Summary

lock.v | lock... | labk... | Desig... | labki... | labki... | labki... | labki...

Step 4C: Implement

- Useful reports: Resource Utilization, Timing, RTL diagram

Xilinx - Project Navigator - C:\ike\6.111\LAB\lecture7\demo1\demo1.ise - [labkit.dly (READ ONLY)]

File Edit View Project Source Process Simulation Window Help

switch

Sources in Project:
- demo1.ise
- xc2v6000-4bf957
  - labkit (labkit.v)
    - labkit.ucf

Module ... | Snapshot... | Library ...

Processes for Source: "labkit"
- Place & Route
  - Place & Route Repor
  - Asynchronous Delay Report
  - Pad Report
  - Guide Results Report
  - Generate Post-Place
  - View/Edit Placed De
  - View/Edit Routed De
  - Analyze Power (XPow
  - Generate Power Data

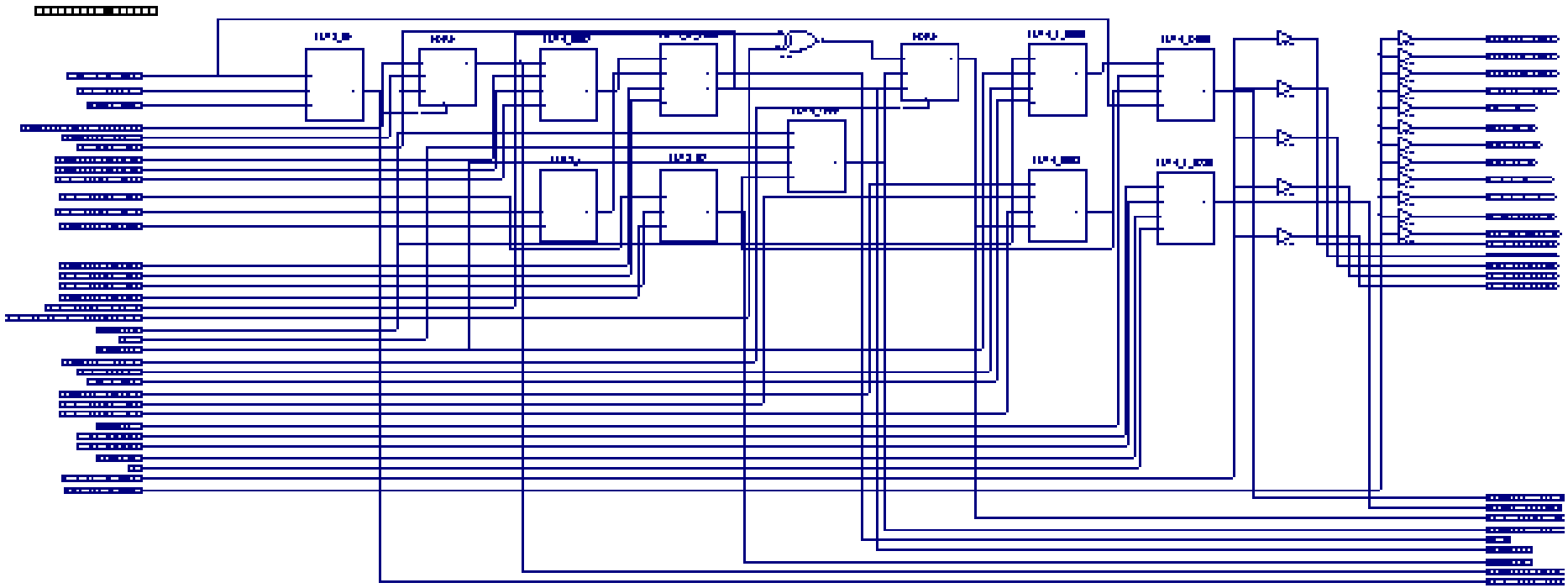
Process View | Hierarchy - lock_tb.tbw

```
1   Release 7.1.03i - reportgen H.41
2   Copyright (c) 1995-2005 Xilinx, Inc.  All right
3
4   Sun Sep 25 16:39:00 2005
5
6   File: labkit.dly
7
8    The 20 worst nets by delay are:
9   +-----------------+-----------+
10  | Max Delay       | Netname   |
11  +-----------------+-----------+
12     5.778            button_enter_IBUF
13     5.467            debounce0/clean
14     5.184            button1_IBUF
15     4.807            debounce1/clean
16     4.462            hexdisp1/clock
17     4.300            button0_IBUF
18     3.827            hexdisp1/disp_ce_b
19     3.760            hexdisp1/state_FFd5
20     3.583            hexdisp1/dreset
21     3.340            hexdisp1/disp_rs
```

lock.v | lock... | labk... | Desi... | labki... | labki... | labki... | labkit....

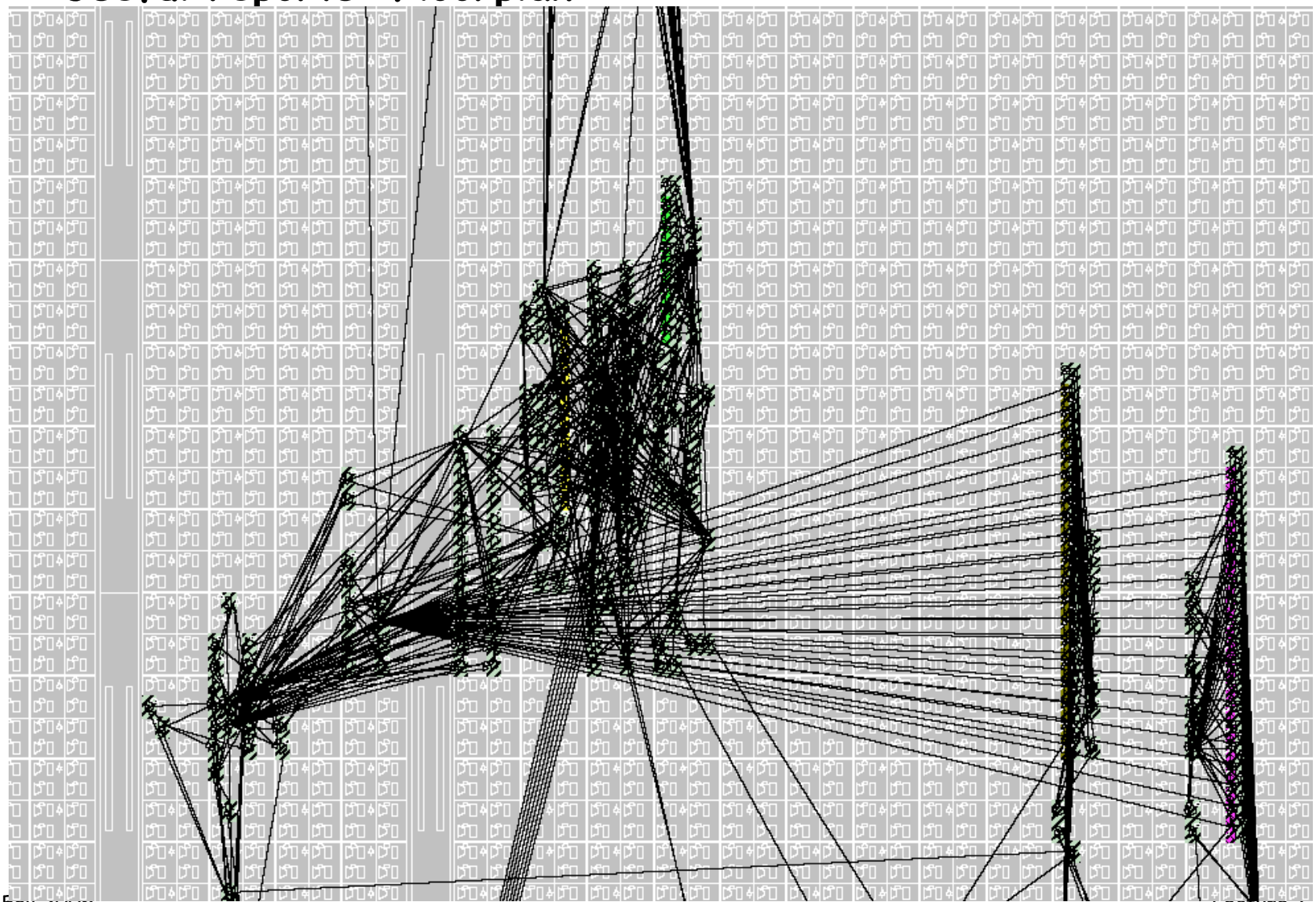
Step 4C: Implement

- Useful reports: Resource Utilization, Timing, RTL diagram



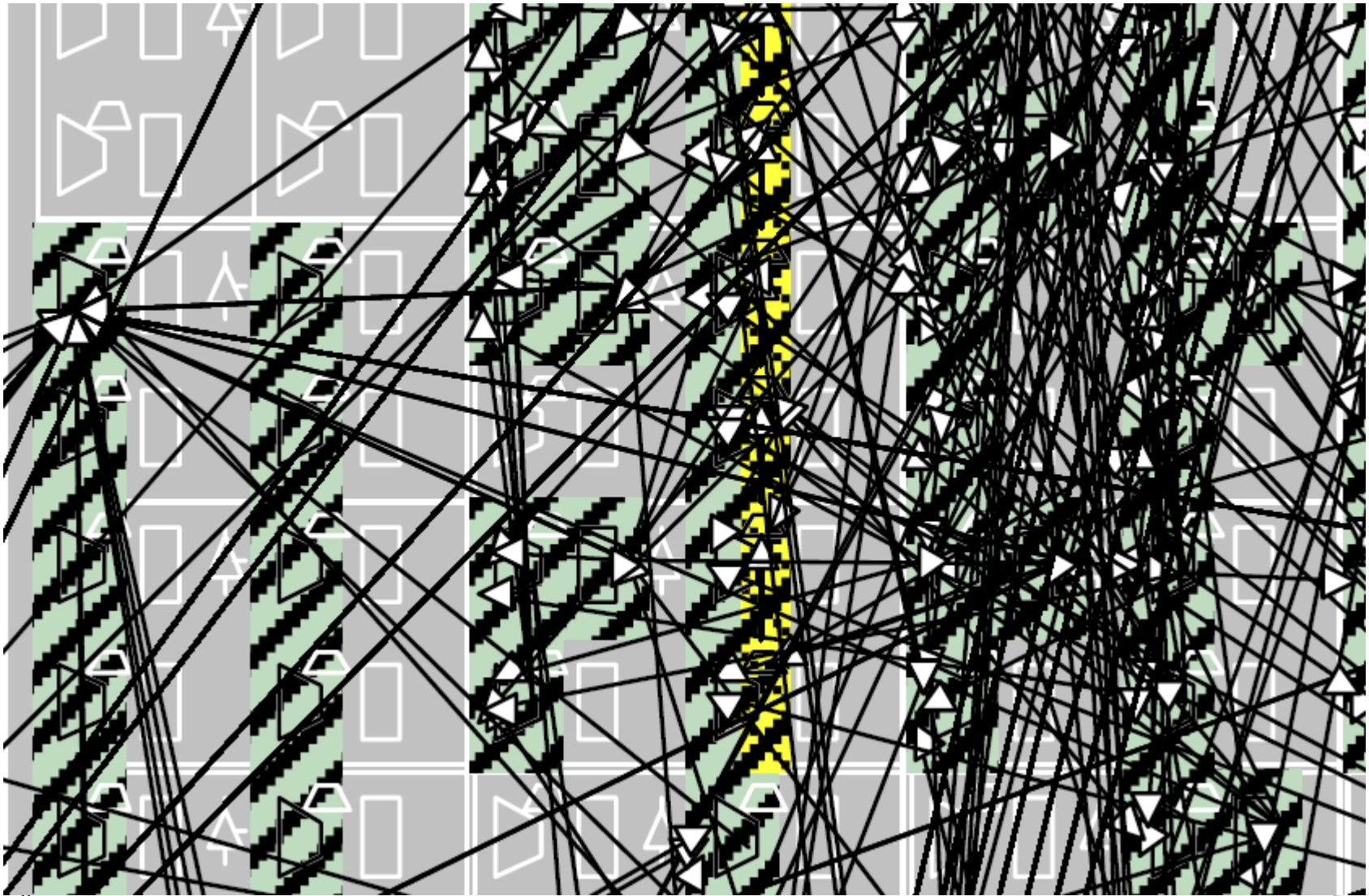
Step 4C: Implement

- **Useful reports: Floorplan**



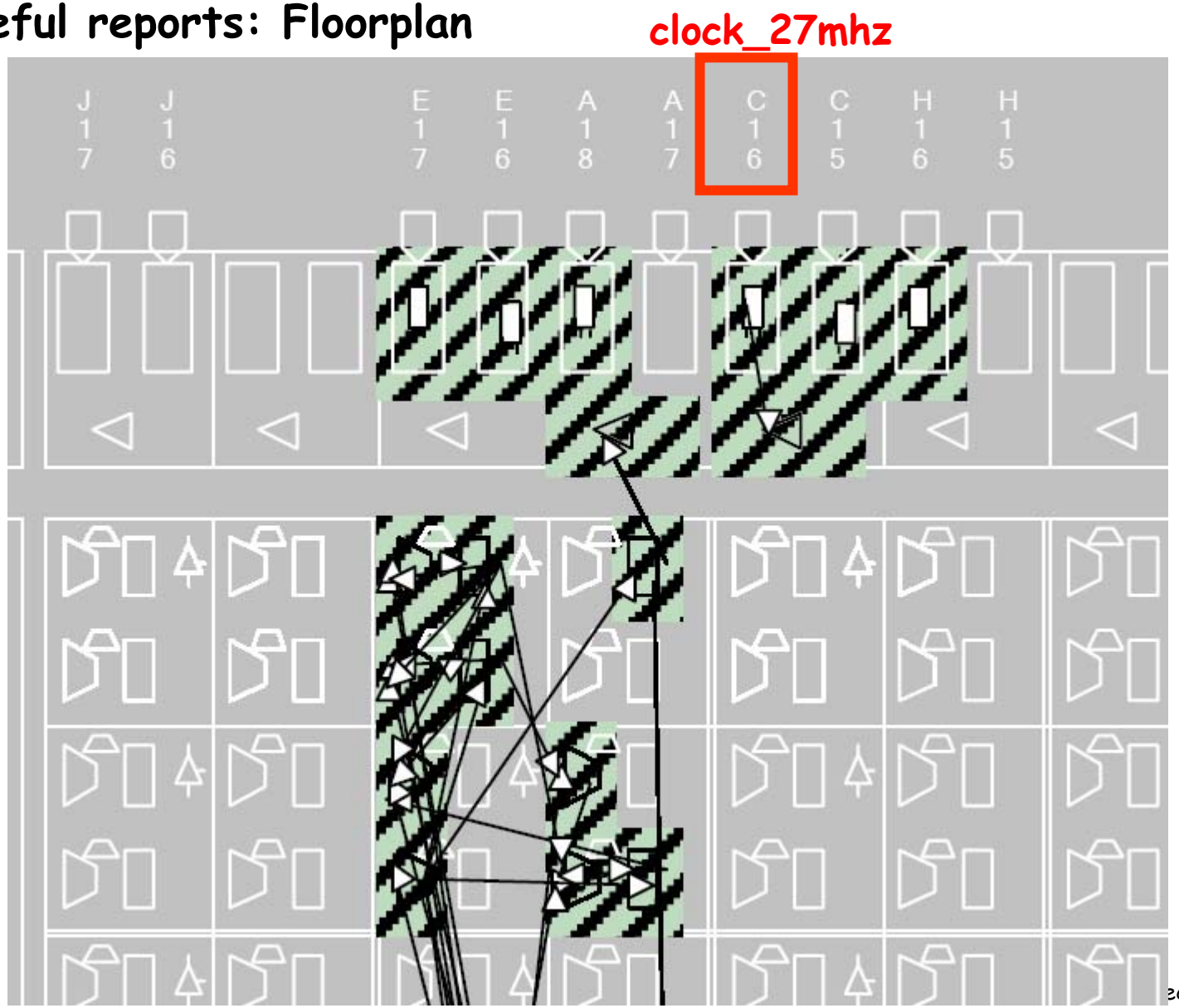
Step 4C: Implement

- Useful reports: Floorplan



Step 4C: Implement

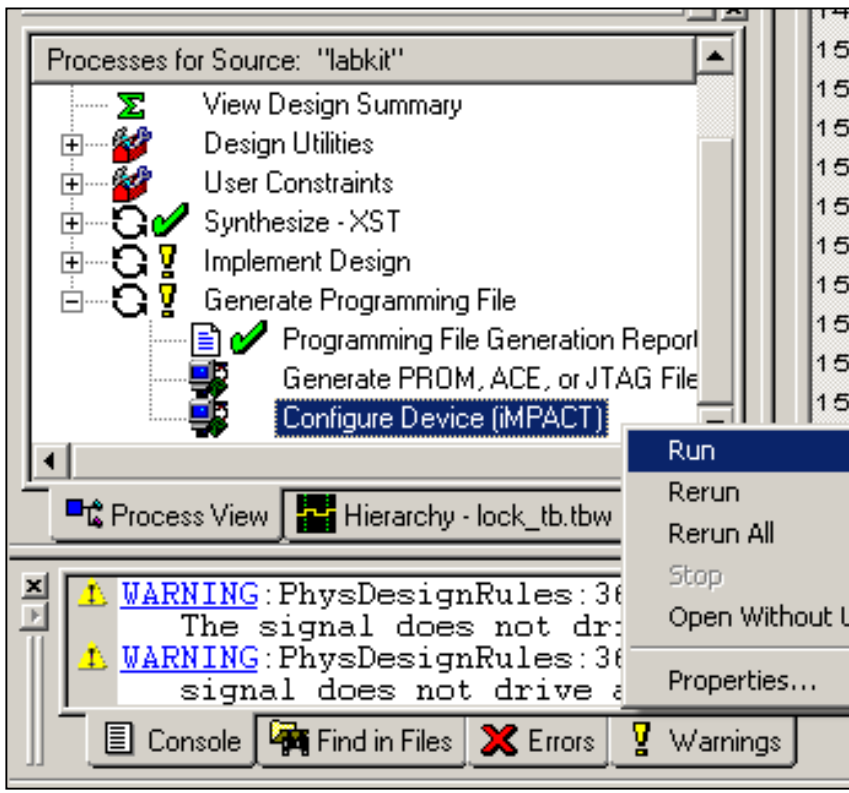
- Useful reports: Floorplan



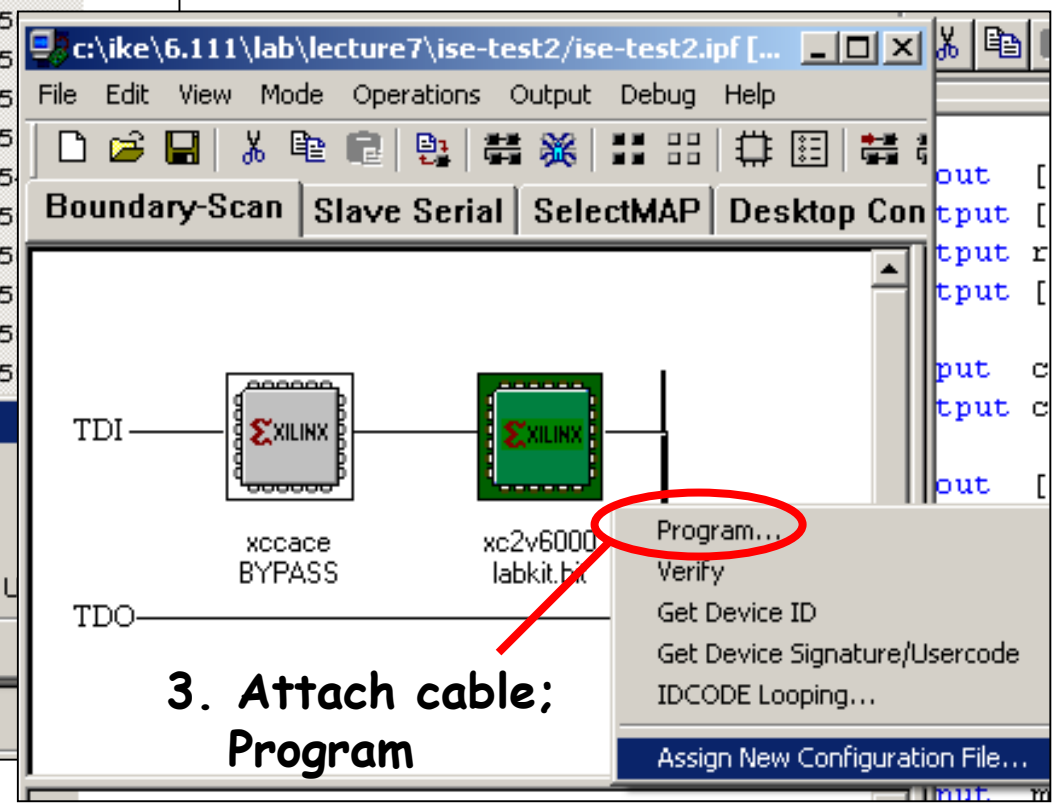
Step 4D: Program FPGA

- Fall '2005 6.111: We'll use the parallel port IV cable
- Transfer program: "IMPACT" – uses JTAG serial chain

joint test access group



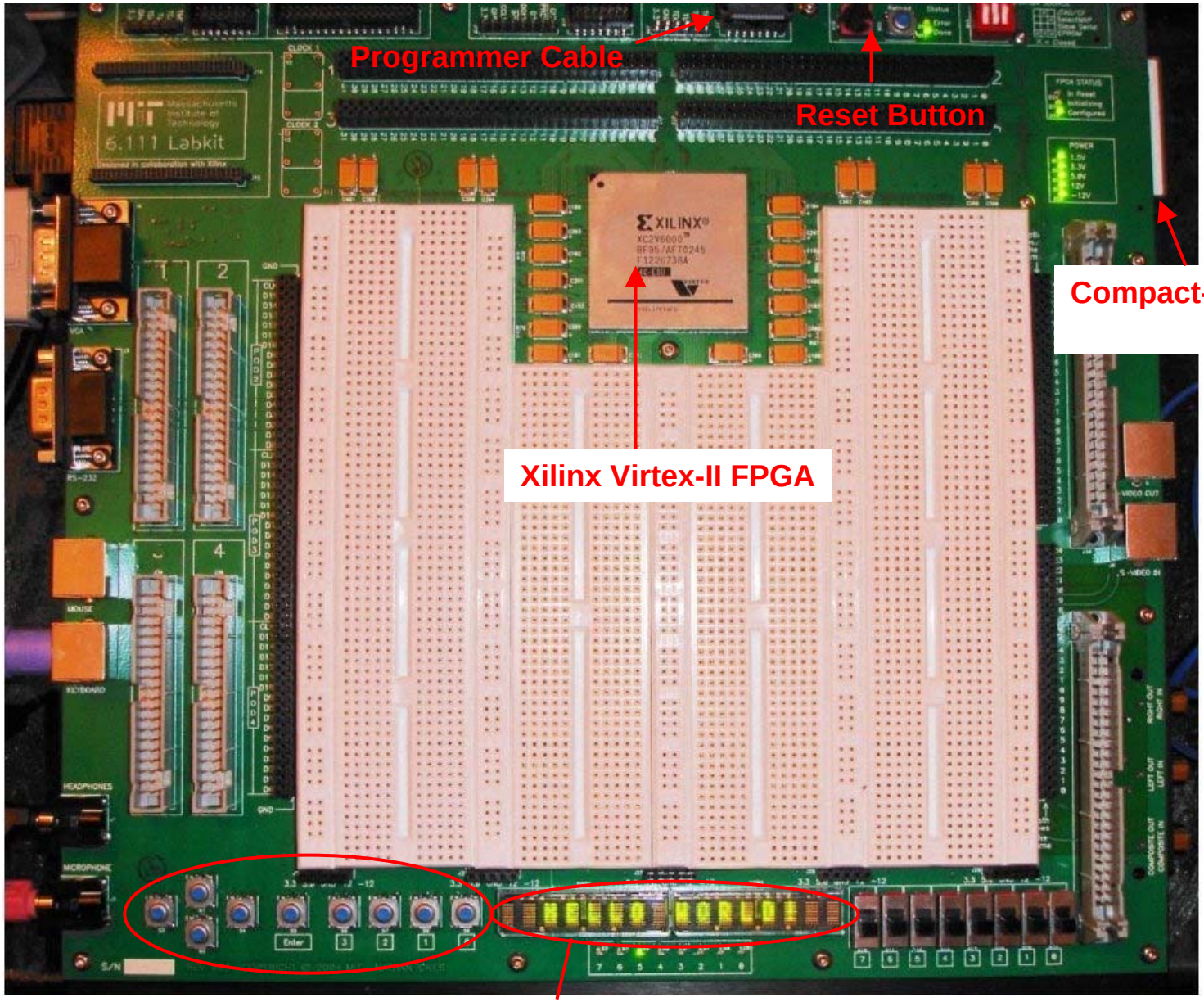
1. Select "Configure Device"



3. Attach cable; Program

2. Bypass first device
assign labkit.bit to second device

Step 4D: Program FPGA



Summary

- Modern digital system design:
  - Hardware description language ➡ FPGA / ASIC
- Toolchain:
  - Design Entry ➡ Synthesis ➡ Implementation
  - Simulate
- New Labkit:
  - Black-box peripherals
  - Almost all functionality is programmed in!
  - How to generate video? Synchronize systems? Create/Digitize Audio? Serial & communications?