### **Reconfigurable Architecture (5)**

<u>osana@eee.u-ryukyu.ac.jp</u>

### Review: Testbench

- `timescale, initial, always #
  - System tasks begins with "\$"
  - Generate input signals to UUT

instance.signal to refer signals in UUT and its submodules

### **Review: Vivado**

- Create project with FPGA order #

  - \* Add RTL files as "Design Source"
  - \* Add testbenches as "Simulation Source"

### \* Keep source files in separate folder to protect (but not mandatory)



## **Review: Vivado Simulator**

- \* Basic usage:
  - Set runtime to 0 in simulation settings
  - \* Advance simulation for  $\Delta t$
  - Changes in signal list:
- Example source codes on the web
  - http://mux.eee.u-ryukyu.ac.jp/lecture.html.en



Changes in source code: 🗔





# Today's goal

- \* To make the board work
  - \* LED flashing, then 7 segment LED display
  - Write RTLs, simulate, then synthesize + place & route
- Using "parameter" in simulation / implementation
- Then, first assignment in this course



## Implementation Flow

- Logic synthesis
- Technology mapping
- Place and route
- Bitstream generation
  - Details in hands-on



# Today's hands-on

- Synthesize, place & route, bitstream generation and FPGA programming
  - All Source files available on the web
    - Including the code on the slide



# First goal of assignment

Making a stopwatch

- \* Push buttons + 7 segment LED display controller
- Decimal counter
- Can be achieved using most of today's source code



### Push buttons

- Chattering filter

  - Optional task in the assignment

### Last week's example emits 5Hz pulse while keep pushed down





# 7 segment LED display

- Common anode
  - Column to illuminate: Anode=L
  - Segment to illuminate: Cathode=L
  - \* (dynamic drive)



*Figure 18. Common anode circuit node* 

### Quickly scanning all columns, show all figures with spectrum

## **Timescale: Human and Circuit**

- \* FPGA runs at 100MHz, but we don't
  - \* LED at 100MHz is invisible, counters to make slower signals
  - 24bits counter is about 6Hz: 2<sup>24</sup>=16M
  - But simulating up to 16M is not realistic
    - Better if different # bits can be used to simulate / implement

# Led flashing, again

- Red figure is to be changed
  - \* 23 to implement
  - 2 to simulate: enough fast to trace on waveform

```
module led_kurukuru
    ( input wire CLK, RST,
      output reg [15:0] LED );
  reg [23:0] CNT;
  wire STROBE = &CNT;
  always @ (posedge CLK) begin
     if (RST) begin
        CNT <= 0;
        LED <= 16'b1000_0000_0000;
     end else begin
        CNT <= CNT+1;
        if (STROBE)
          LED <= {LED[0], LED[15:1]}
    end
  end
endmodule
```

# New syntax: Parameter

- 2 ways to declare:
  - in module
  - or before port definitions
- Both requires default value
- The latter can change port width

```
Style 1:
module led_kurukuru
    ( input wire CLK, RST,
      output reg [15:0] LED );
  parameter CounterBits = 24;
  reg [(CounterBits-1):0] CNT;
  (no changes below)
Style 2:
module led_kurukuru #
      parameter CounterBits = 24 )
     input CLK, RST,
      output reg [15:0] LED );
  reg [(CounterBits-1):0] CNT;
  (no changes below)
```

# Changing parameters

- Do nothing for 24bits
  - Parameters can be overwritten on instantiation
  - \* kuru2 has a 2 bit counter
- Handy for faster simulation

led\_kurukuru
 kuru1(.CLK(CLK), .RST(RST), .LED());

led\_kurukuru # ( .CounterBits(2) )
kuru2(.CLK(CLK), .RST(RST), .LED());





# 7-seg LED

0~9 figures and their pattern

- 7bit (without dot) for
   cathode, 0 to illuminate
- Anode (column) later



Individual cathodes

# Single-column driver

### \* 4bit in $\rightarrow$ 7bit out



```
module segment_driver
  (input wire [3:0] VAL,
    output wire [6:0] CATHODE );
```

```
assign CATHODE =
    (VAL==4'h0) ? 7'b000 0001 :
    (VAL==4'h1) ? 7'b100 1111 :
    (VAL==4'h2) ? 7'b001 0010 :
    (VAL==4'h3) ? 7'b000_0110 :
    (VAL==4'h4) ? 7'b100 1100 :
    (VAL==4'h5) ? 7'b010 0100 :
    (VAL==4'h6) ? 7'b010_0000 :
    (VAL==4'h7) ? 7'b000_1111 :
    (VAL==4'h8) ? 7'b000_0000 :
    (VAL==4'h9) ? 7'b000_1100 : 7'h0;
```

endmodule

Logic compression done by Vivado

### **Overall structure**

- \* 8 columns = 32 bits in
  - \* 4bits for each column

 Hexadecimal is also possible with modified segment\_driver module







# Today's Assignment

- Make a stopwatch
  - \* Minimum 1/10000s
  - \* Maximum 9999.9999s
- Suppose that the board's 100MHz clock is 100% reliable

# Buttons: RST and START-STOP, latter requires chattering removal

### FPGA boards for lab/assignment

- Digilent Nexys4 (discontinued)
- Digilent Nexys4 DDR and Nexys A7-100T (they're same)
  - Pin assignments differ between Nexys4 <-> DDR / A7
  - \* Choose the right XDC file, or you'll break the FPGA



### Lab 1: Flashing LEDs



## Goal of the lab

- Different # bits to simulate and implement
  - Change the parameter in simulation
  - Source files are available on the web
- Understand Vivado usage in its implementation flow

## Source codes

- Available from http://mux.eee.u-ryukyu.ac.jp/lecture.html.en
  - Download and unzip lab-05.zip
  - Constraint: nexys4.xdc or nexys4ddr.xdc
  - Design Source: led\_kurukuru.v
  - Simulation Source: led kurukuru test.v



# Make a project

- \* Extract the zip file in working directory: 05-flash-led + 05-btn-cnt
- Make a Vivado project in 05-flash-led/vivado
  - \* Device: XC7A100T-1CSG324 (xc7a324tcsg324-1 in vivado)
  - \* 3 source files: RTL, TB and constraint
- Important: Choose the right constraint file for your board!



### Run simulation



# Flow Navigator

- Project Manager
- Simulation
- **RTL** Analysis
- Synthesis
- Implementation: technology mapping + P&R
  - Program and Debug: Bitstream generation and programming

### Implementation flow (up to down)

(IP Integrator is not in today's scope)



## Project Manager

- \* is the initial screen
  - Target device and project summary
- On right bottom:
  - Messages / Log: check if you've got errors
  - Design Runs: progress of Implementation flow tasks



# **RTL Analysis**

- \* "Open Elaborated Design" to initiate
- Schematic is linked to RTL, jump on right click
  - Not very important, just for check \*



Launched automatically on synthesis and later commands RTL Analysis



RTL Analysis



### Synthesis

- \* "Run Synthesis" to launch
  - Open Synthesized Design to see reports/schematics \*
    - \* Only I/Os shown in device view (because the circuit) isn't placed yet)
  - Resource estimate in Project Summary
    - Fixed after implementation, just an estimation at this point



*
75 100
ation (%)
Synthesis
Synthesis Settings
No Synthesis Settings
<ul> <li>Open Synthesized Design</li> </ul>
Sunthasis
Synunesis
No Synthesis Settings
Synthesis
Synthesized Design
🚠 Constraints Wizard
🚵 Edit Timing Constrain
💐 Set Up Debug
🧭 Report Timing Summ:
™> Report Clock Network
🛐 Report Clock Interacti
🥝 Report DRC
掘 Report Noise
📓 Report Utilization
🗊 Report Power
🕍 Schematic

29

Synthesis

Synthesis

### Implementation

- \* "Run Implementation" to launch
  - Whole circuit is placed and routed
  - Resource usage in "Project summary" is fixed
  - Timings and I/O results are very important





# Implementation: Timing

- Available after implementation
  - \* Open Implemented Design  $\rightarrow$  Window  $\rightarrow$  Timing

Timing - Timing Summary - impl_1	
옥 🖀 🖨 ᆃ 🍃 🚽	De
i This is a <u>saved report</u> ×	Set
General Information	
—Timer Settings	
— Design Timing Summary	
Clock Summary (1)	
-Check Timing (17)	
-Intra-Clock Paths	
—Inter-Clock Paths 📃	All
—Other Path Groups 👘	
User Ignored Paths	◀
Timing Summary - impl_1 ×	
🔲 Tcl Console 🔎 Messages 🛛 🖾	Log

### \* "All user specified timing constraints are met" is what we want

				_ 🗆 🖻 ×						
sign Timing Summary										
up		Hold		Pulse Width						
Worst Negative Slack (WNS):	<u>6.037 ns</u>	Worst Hold Slack (WHS):	<u>0.236 ns</u>	Worst Pulse Width Slack (WPWS):						
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):						
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:						
Total Number of Endpoints:	56	Total Number of Endpoints:	56	Total Number of Endpoints:						
user specified timing constraints are met.										
				4 ▷ 🗉						
🖹 Reports 🚯 Design Runs 🍯 Timing 🕞 I/O Ports										



## Implementation: I/O Ports

- \* Open Implemented Design  $\rightarrow$  Window  $\rightarrow$  I/O Ports
  - \* All ports must be "Fixed" and their I/O Standard must be correct
  - If not, the board/FPGA may be broken

1/0	) Ports											— 🗆 L	<u> -</u> ×
0	Name	Direction	Neg Diff Pair	Site		Fixed	Bank	I/O Std		/cco   Vi	ref   Drive	Strength Slew Ty	/pe
-	P- 🐼 All ports (18)												
<b>1</b>	∲- 🔞 LED (16)	OUT					3	4 LVCMOS33*	Ŧ	3.300	12	👻 SLOW	- <b>v</b>
		OUT		P2	~	<b>V</b>	3	4 LVCMOS33*	Ŧ	.300	12	👻 SLOW	-
⊡ <sub>≩</sub>		OUT		R2	~		3	4 LVCMOS33*	Ŧ	.300	12	👻 SLOW	-
	LED[13]	OUT		U1	~	<b>V</b>	3	4 LVCMOS33*	Ŧ	.300	12	👻 SLOW	- L_I
-		OUT		P5		<b>V</b>	3	4 LVCMOS33*	w	.300	12	👻 SLOW	▼ 1
2		OUT		R1		<b>V</b>	3	4 LVCMOS33*	Ŧ	.300	12	👻 SLOW	I
æ		OUT		V1	~	<b>V</b>	3	4 LVCMOS33*	w	300	12	👻 SLOW	-
		OUT		U3		<b>V</b>	3	4 LVCMOS33*	Ŧ	.300	12	👻 SLOW	-
$\checkmark$		OUT		V4	~	<b>V</b>	3	4 LVCMOS33*	w	300	12	👻 SLOW	*
		OUT		U6		<b>V</b>	3	4 LVCMOS33*	w	.300	12	👻 SLOW	* I
		OUT		U7		<b>V</b>	3	4 LVCMOS33*	w	.300	12	👻 SLOW	- T
		OUT		Τ4		<b>V</b>	3	4 LVCMOS33*	Ŧ	.300	12	👻 SLOW	- L
		OUT		Τ5			2	1 I VCMOS22*		200	17	₩OUZ ₩	
				-									
	🔚 Tcl Console 🗋 🔎 Messages 🔄 🖳 Log 📄 Reports 🛛 🦈 Design Runs 🛛 🧭 Timing 🕞 🖬 / O Ports												



### Bitstream generation

- \* Place & Route result to be written on an FPGA
  - and generate bitstream
  - This is handy, but will be sometimes too time consuming



### Just click on "Generate Bitstream" to synthesize, implement



# Hardware Manager

- Connect the board and turn power on
- \* "Open Target" to find the FPGA
- \* "Program device" to program
  - Default filename is good in typical use case



Hardware Manager - macpro-linux/xilinx_tcf/Digilent/210274593047A									
There are no debug cores. Program device Refresh device									
Hardware	🔷 xc7a100t_0	Debug Probes (							





### Check it:

Also try to change parameter for simulation

\* LED flashes from left to right, in different speed with simulation



# Lab 2: Hexadecimal LED counter

# Make a project

- \* File  $\rightarrow$  New Project
- Sources in 05-btn-cnt/src
  - \* push\_counter\_test.v: TB, nexys4.xdc or nexys4ddr.xdc: constraints
  - All others as design sources



# Organization





### Button assignments

- \* Left (BTNL) : ADV
  - Increment the counter
- \* Center (BTNC): RST
  - Reset



### Parameters

- LED column switching speed (ColumnCounterBits)
  - \* 10bit counter = 1/100 ms (may be too fast...)
- Chattering filter counter (ButtonFilterCount)
  - \*  $20x10^6 = 1/5sec$



# Try it

- Keep pressing down left button
  - Counter++ at 5Hz
- ColumnCounterBits = 24 in push\_counter.v
  - Slower LED column scanning, same to exercise 1



## Assignment

- Stopwatch
  - You can use any of today's source codes
- Deadline Nov.20
  - Send me zipped all source codes (\*.v and \*.xdc), pages in A4)

### with a description of the design's organization in PDF (1 or 2)



# Note for remote programming

- \* Sharing an FPGA card in the la
  - Connect the card to a host
  - Launch "hw\_server" on the host
     (no full version of Vivado is required, included in Vivado Lab edition)
- Choose "Remote server" on launching HW manager, "Auto connect" is not available for remote use

ab	Hardware Sen Select local or ren is attached to the	Hardware Server Settings Select local or remote hardware server, then configure the host name and port settings. Use Local server if the t is attached to the local machine; otherwise, use Remote server.						
	<u>Connect to:</u>	emote server	(target is on remot	e machine) 🖌 🗸				
	<u>H</u> ost name: Port:	nucl 3121	[default is 312	Ø ✓				
nost								



