Hierarchical Modeling & Synthesis with access via Boundary Scan

ELEC 4200

Auburn University

October 24th, 2011
• Write a new top-level VHDL to combine your models for
  • Hierarchical model from Lab 7
  • Multiplexers (need to write model)
  • 6-bit shift register that connects to internal BSCAN_SPARTAN3
    • Use your universal counter/shift register from Lab 5
• Simulate and verify (debugging where needed) design using ModelSim
• Synthesize, download, & verify design in Spartan 3 FPGA using Impact Boundary Scan interface
**Pre-lab Assignment**

1. Write VHDL models for
   - Multiplexer bank
   - Top level model that connects
     - Lab 6 circuit with MUXs
     - Shift register with instantiation and connection to BSCAN_SPARTAN3 module (referred to BSCAN module from here) and to multiplexers

2. Determine the FPGA pin number for the mode select pin

3. Determine FPGA pin numbers to connect BSCAN module signals to LEDs
   - This will help you watch BSCAN in action and is useful for debugging
   - Depending on which BS port you choose, probe either
     1. DRCK1, SEL1, Shift, Update, TDI, TDO1
     2. DRCK2, SEL2, Shift, Update, TDI, TDO2

4. Read
   - Spartan 3 Configuration (pages 169-183) from class webpage
   - Gefu’s tutorial on “Communicating with BS” on class webpage
Boundary Scan Interface

- The FPGA has two internal Boundary Scan ports to access the programmable logic and routing resources
  - Boundary Scan is a serial interface and will require a serial to parallel conversion to communicate with the register file and display controller
  - Use a mode switch to select between the switch & push button interface from Lab 7 and the new Boundary Scan interface circuitry
- Specifications for design:
  - A bank of seven 2-to-1 MUXs controlled by a common select input (Choose from remaining switches on Spartan3 PCB)
  - For 6-bit serial-to-parallel shift register (Lab 5) use
    - Active high enable (connects to SHIFT signal from BSCAN module)
    - Serial data in (connects to TDI signal from BSCAN module)
    - Serial data out (connects to TDO1 or TDO2 signal to BSCAN module)
    - Parallel data out (connects to MUXs)
Instantiating BSCAN module

- Include the following library and package to access the BSCAN module

```vhdl
library UNISIM;
use UNISIM.vcomponents.all;
```

- Include the following instantiation of BSCAN module
  - NOTE: you do not declare the component
  - NOTE: you do not need to include unused connections
  - NOTE: you should bring BS signals to LEDs to watch the operation of the BS interface in conjunction with movement through the TAP state diagram
  - Choose your connections to either BS User Port 1 (Red), or BS User Port 2 (Green) from ONE of the next two slides.
BS User Port 1

BSCAN_SPARTAN3_inst:BSCAN_SPARTAN3
portmap(
    "- Data register output for USER1 functions (clock for shift register - bring to LED)
    DRCK1 => DRCK1,
    "- USER1 active output (not used in your design but bring it out to an LED)
    SEL1 => SEL1,
    "- SHIFT output from TAP controller (enable for shift register - bring it out to an LED)
    SHIFT => SHIFT,
    "- TDI output from TAP controller (input data to shift register - bring it out to an LED)
    TDI => TDI,
    "- UPDATE output from TAP controller (write enable to Lab 6 circuit)
    UPDATE => UPDATE,
    "- Data input for USER1 function (output data from shift register - bring it an LED)
    TDO1 => TDO1,
);

### Instantiating BSCAN module

#### BS User Port 2

```vhdl
BSCAN_SPARTAN3_inst:BSCAN_SPARTAN3
portmap (    
- - Data register output for USER2 functions (clock for shift register - bring to LED)
DRCK2 => DRCK2
- - USER2 active output (not used in your design but bring it out to an LED)
SEL2 => SEL2,
- - SHIFT output from TAP controller (enable for shift register - bring it out to an LED)
SHIFT => SHIFT,
- - TDI output from TAP controller (input data to shift register - bring it out to an LED)
TDI => TDI,
- - UPDATE output from TAP controller (write enable to Lab 6 circuit)
UPDATE => UPDATE,
- - Data input for USER2 function (output data from shift register - bring it to an LED)
TDO2 => TDO2,
);
```
Instructions for BSCAN Access

- 6-bit instruction
  - User port 1: 0x02
  - User port 2: 0x03
    - note that this is what is shifted into the IR in the time example below
  - Shifted into IR LSB first
  - Exit IR on last bit of instruction
  - You are now in Shift-DR and ready to access BSCAN port
- See visual example on the next page
Instructions for BSCAN Access

TMS=01100 000001 1100 time →
TDI=xxxxxx 110000 xxxx

moving from Test-Logic-Reset to Shift-IR

shifting in instruction 0x03 LSB first to Shift-DR

moving from Exit-IR to Shift-DR

Note: transitions on rising edge of TCK based on TMS value
Boundary Scan Communications

- As you serially shift in the address and data on TDI via the USER2 port of the BSCAN module, the previous values you sent to the register file will shift out on TDO.
- As a sanity check, you can continue to shift in data and see it come out 6 clock cycles later in TDO in the Impact GUI.
- As you shift in the last bit of your address and data bring TMS high to Exit-DR and the next clock with TMS high will activate Update and load your data into the appropriate register in your register file.
- You can go back to Shift-DR to shift in a new set of address and data values for another register and repeat the process as long as you as you like without changing IR.

Note: the connections shown are for User Port #2.

Signals are active only when instruction for user boundary scan access is in IR.
Lab Exercise

- Simulate your VHDL models and verify your design using ModelSim, debug as necessary
  - Note that you will “force” the outputs of the BSCAN module during your simulation since you will not be able to simulate the module in ModelSim
- Synthesize your design for the Spartan 3S200 FPGA using ISE
  - Open the synthesis report file and record the number of Slices, LUTs, and FF/latches
- Test and debug your circuit using the Impact Boundary Scan interface to move through the TAP state diagram and communicate with your internal circuitry
- Demonstrate your working circuit with the ability to communicate with BS to the GTA
1. Did you parametrize your multiplexer bank? Why or why not?
2. Did you need to use probes? How did you accomplish this?

Report must contain

1. Verified Parametrized VHDL models
2. Indicate which BS User Port you chose
3. Simulation Results
4. A logic diagram showing interconnection and hierarchy of your VHDL models with internal signal names
5. Synthesis Report results including number of slices, LUTs, and FFs/latches
6. All prelab work including FPGA pin numbers