CS220A Lab#2
Adders and Comparators

Mainak Chaudhuri
Indian Institute of Technology Kanpur

Sketch

- Assignment#1: two-bit adder
- Assignment#2: seven-bit adder
- Assignment#3: eight-bit comparator

Lab#2

- Make new folders Lab2_1, Lab2_2, Lab2_3 under CS220Labs to do the assignments
- Refer to lab#1 slides for Xilinx ISE instructions
- Finish pending assignments from lab#1 first
- Assignment marks:
  - Assignment#1: 2 marks
  - Assignment#2: 4 marks
  - Assignment#3: 5 marks

Assignment#1

- Assignment#1
  - Design a two-bit adder with cin of the least significant bit adder assumed to be zero
    - Total number of inputs is four: x[0], x[1], y[0], y[1]
    - Three outputs: z[0], z[1], and carry from the most significant bit adder
    - z = x + y
    - Use SW0 and SW1 to feed x[0] and x[1]
    - Use SW2 and SW3 to feed y[0] and y[1]
    - Observe z[1] and z[0] in LED1 and LED0
    - Observe carry in LED2

Assignment#1

- Write a Verilog module for a full adder
- Write a Verilog module that instantiates two full adder modules and connects them appropriately to design the two-bit adder
- Write a top-level Verilog Test Fixture for simulation
- Simulate and synthesize your design

Assignment#1

- Verilog module for two-bit adder

```
module two_bit_adder (x, y, z, carry);
   input [1:0] x;
   input [1:0] y;
   output [1:0] z;
   wire [1:0] z;
   output carry;
   wire carry;
   wire carry0;
   full_adder FA0 (x[0], y[0], 1'b0, z[0], carry0);
   full_adder FA1 (x[1], y[1], carry0, z[1], carry);
endmodule
```

Assignment#2

- Seven-bit adder
  - Only four slide switches on board, but need to input two seven-bit numbers
  - Idea is to divide each input (say, A and B) into two halves and use the four push buttons
    - Place the slide switches in position for the lower four bits of A and push one of the push buttons PB1 (say, BTN_NORTH)
    - Place the slide switches in position for the higher three bits of A and push another push button PB2 (say, BTN_SOUTH)
    - The Verilog module should be written such that when PB1's posedge comes, A[3:0] is stored and when PB2's posedge comes, A[6:4] is stored
    - Input B using the remaining two push buttons

Assignment#2

- Push buttons
  - Read pages 16 and 17 from https://www.xilinx.com/support/documentation/boards_and_kits/ug230.pdf
  - We will use BTN_EAST, BTN_WEST, BTN_SOUTH, and BTN_NORTH along with the slide switches for feeding the inputs in this assignment
  - The seven-bit sum and carry-out should be shown on the eight LEDs
    - LED0 should be sum[0], …, LED6 should be sum[6], and carry-out should be LED7

Assignment#2

- Plan the Verilog modules
  - 1st level: a full adder module that can add two bits with a carry-in
  - 2nd level: an array of seven full adders in ripple-carry mode (carry-in of LSB should be set to 0)
    - Inputs: PB1, PB2, PB3, PB4, Y where Y is a four-bit number; four bits of Y are the slide switch inputs
    - Outputs: seven-bit sum and carry-out
    - Structure:
      - Four always blocks, one each for posedge PB1/2/3/4; sequential assignment for A[3:0], A[6:4], B[3:0], B[6:4] in the always blocks (one assignment per always block); for example, A[3:0] <= Y; A[6:4] <= Y[2:0];
      - Array of seven full adders (outside always blocks)

Assignment#2

- Write a suitable Verilog Test Fixture for ISim simulation
- Run PlanAhead to assign pins to inputs and outputs
  - Make sure to use exactly same IOSTANDARD, PULLUP/PULLDOWN, SLEW, DRIVE for the pins as shown in Chapter 2 of https://www.xilinx.com/support/documentation/boards_and_kits/ug230.pdf
  - Manually add the following in the .ucf file
    - NET "PB1" CLOCK_DEDICATED_ROUTE = FALSE;
    - Same for PB2, PB3, PB4
- Synthesize the seven-bit adder on FPGA

Assignment#3

- Eight-bit comparator
  - Compares two eight-bit inputs A and B
  - Outputs whether A<B, A>B, A==B
  - Accept inputs same way as the previous assignment
  - LED0 should glow if A<B, LED1 should glow if A>B, LED2 should glow if A==B

Assignment#3

- Verilog modules
  - Decompose the eight-bit comparator as follows
    - Number the bit positions 0 to 7 from LSB to MSB
    - Focus on bit position k comparing a_k and b_k
    - Start comparing from MSB and the first bit position where $a_k \neq b_k$ decides whether A<B or A>B; this observation leads to the first level module
    - 1st level: inputs are two bits (a, b) to be compared and less, greater, equal indication from the next MSB position
      - If the less input is 1, the output is less irrespective of a, b
      - If the greater input is 1, the output is greater irrespective of a, b
      - If equal input is 1, output is a<b, a>b, or a==b

Assignment#3

- Verilog modules
  - 2nd level
    - Array of eight 1st level comparators
    - The equal input of MSB comparator should be 1 and the less, greater inputs should be 0
    - The outputs of the LSB comparator are the final outputs
- Write a Verilog Test Fixture with appropriate inputs for ISim
- Use PlanAhead to assign pins and synthesize your hardware on FPGA
  - Manually add the following in the .ucf file
    - NET "PB1" CLOCK_DEDICATED_ROUTE = FALSE;
    - Same for PB2, PB3, PB4