Verilog HW#2: BCD ALU Module

Chun-Jen Tsai
National Chiao Tung University
4/17/2020

Verilog HW#2

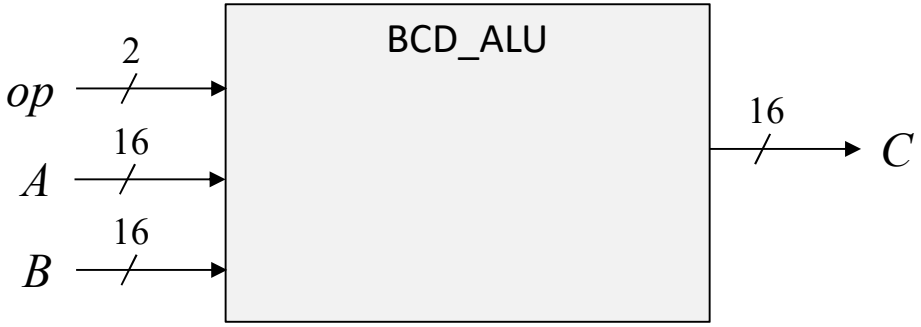
- Goal: Design a Verilog module using behavioral description that takes two 4-digit BCD number and a 2-bit operation selection signal as inputs.

  The module will perform the selected arithmetic operation and output the result as a 16-bit binary number. Negative integer output shall be represented in 2's complement format.

- Deadline: 5/4, 23:55pm. You must upload your Verilog module to the E3 website by the deadline.

Block Diagram of the Module

- Your module should be called BCD_ALU, with the following input/output ports:



  - *op* is a 2-bit input signal that specifies one of the following four BCD operations to compute: addition, subtraction, 9's complement, and comparison.
  - A and B are 4-digit BCD numbers (16 binary digits each).
  - C is the 16-bit result of the operation in two's complement binary format.

ALU Operations

- The input code "*op*" and the corresponding Arithmetic and Logic Unit (ALU) operations are as follows:

op	Operations
00	$C = A + B$
01	$C = A - B$
10	C = 9'complement of A
11	$C = \begin{cases} 16'h0001, & if\ A > B \\ 16'h0000, & if\ A = B \\ 16'hFFFF, & if\ A < B \end{cases}$

- The output C uses two's complement binary code.
  - For example, if $op = 01$, A = 16'h0007, B = 16'h0023, then C = 16'hFFF0.

Input Error Detection

- The input numbers A and B must be legal BCD numbers between $0000 \sim 9999$. If not, the output C must be set to $16'\text{hCCCC}$ regardless of the input op code.
  - For example, if $op = 10$, $A = 16'\text{h000A}$, $B = 16'\text{h0023}$, then $C = 16'\text{hCCCC}$.

Requirements for Verilog HW#2

- The module you designed must be declared as follows:

```
module BCD_ALU(input [1:0] OP, input [15:0] A, input [15:0] B,
output [15:0] C);

  /* Implement your design here. */

endmodule
```

- Do not upload your testbench module to E3, just upload the `BCD_ALU()` module and its supporting modules (if you have more than one modules/files). The TA's will use their own testbench to test your module.