Birla Institute of Technology and Science Pilani
CS F342 Computer Architecture
First Semester 2021-22
Lab Test (28/11/2021)

Max. Marks: 30 **Max. Time: 60 minutes**

Division of Unsigned Binary numbers

Strictly follow the naming conventions for the signals and modules. Additional ***parameters***, ***wires*** *and* ***regs*** *if needed can be added only internal to the module.*

The objective here is to implement the division of two unsigned numbers. The Dividend is a 16-bit number (***inA***); divisor is an 8-bit number (***inB***). The Quotient (***qnt***) and remainder (***rem***) are 16-bit, 8-bit each. The operation has to be performed sequentially by the method of repetitive subtraction. The module has a **CLOCK** (positive edge triggered) signal, **RESET** signal (active high, asynchronous). The **FLAG** signal indicates the status of the division (look at the table). The module has an additional input **LOAD**, the new operands are loaded into the divider when this goes high for one clock cycle.

```
module division (FLAG, qnt, rem, inA, inB, CLOCK, RESET, LOAD);

input CLOCK, RESET, LOAD ;
input [15:0]inA ;
input [7:0]inB;
output [2:0] FLAG;
output [15:0] qnt; output  [7:0]rem;

// Your code here
endmodule
```

Algorithm

1. The system is in IDLE state until the **LOAD** input goes high.
2. *If(LOAD)* the operands are loaded and the process begins.
3. Check for the special cases (see page-2)
4. If no special case occurs the proceed for the subtraction.
5. Repeatedly subtract the *divisor* from *dividend* incrementing the *quotient.*
6. Repeat *step-5* till the result of the subtraction is less than or equal to 0.
7. Process completed and valid output available *qnt* and *rem*.

Special Cases

The process described here is a very slow division process and hence to speed up the process, some special conditions are treated separately.

1. The divisor (***inB***) is 0. qnt= rem= all 1's;
2. Divisor is unity, ***qnt = dividend*** and ***rem=0***;
3. Divisor is a power of two (2^n), output the ***qnt*** and ***rem*** based on n.
4. Dividend is a left shifted version of divisor, ***qnt= shift*** and ***rem=0.***
5. Dividend is less than divisor, ***qnt= 0*** and ***rem=dividend.***

Flag conditions

S.No.	FLAG	Comment
1.	XXX	The system is in unknown state
2.	000	The system is in reset mode and idle
3.	001	No Special case detected, processing inputs.
4.	010	Operation completed; output available in ***qnt, rem***
5.	011	Operation completed; Special case-1 detected;
6.	100	Operation completed; Special case-2 detected;
7.	101	Operation completed; Special case-3 detected;
8.	110	Operation completed; Special case-4 detected;
9.	111	Operation completed; Special case-5 detected;

a. Using *dataflow* statements only, identify the special cases and raise the FLAG. Provide comments before each statement. Write all these statements at one place in the Verilog code and in sequence.

b. Complete the module (repetitive subtraction) ***division*** to implement the division algorithm.

c. Write a test bench module *tb_division* to test the module *division* the first five test cases should invoke each of the special cases in sequence, a test case each for the minimum and maximum delay case and any 2 random test cases has to be included. Include comments to explain the test cases.

******End of paper******