Final Project

ALGORITHMIC PROBLEM SOLVING USING A SYSTEM CONSISTING OF AN ALU, A REGISTER BLOCK AND A STATE MACHINE

PROJECT DEFINITION

Students will develop an algorithmic solution to one of the problems below, using a simple processor system consisting an Arithmetic Logic Unit (ALU), a register block and an algorithmic state machine. Problems will be randomly assigned to each group (see Table 1 at page 3). **Corrections/announcements will be made if a mistake is detected in the project files on Ninova. Please check Ninova announcements regularly.**

List of Problems:

1) Multiply two 4-bit signed integers represented with two's complement and produce 8-bit signed integer as $C = A \times B$.

2) Calculate $C = A^B$, where A and B are 2-bit positive integers and C is a 6-bit positive integer.

3) Calculate $C = \frac{A}{B}$ where A is a 8-bit positive integer and B is a 4-bit positive integer. There will be two 8-bit positive integer outputs as quotient, Q, and remainder, R.

4) Calculate $C = A \ mod \ B$ where A, B and C are 8-bit positive integers.

5) Compare $2A$ and B where A and B are 8-bit positive integers. The output should give 0 when $2A < B$, 1 when $2A = B$ and, 2 when $2A > B$.

6) Compare A and $\frac{B}{2}$ where A and B are 8-bit positive integers. The output should give 0 when $A < \frac{B}{2}$, 1 when $A = \frac{B}{2}$ and 2 when $A > \frac{B}{2}$.

7) Calculate $C = abs(A - B)$ where A and B are 8-bit signed integers represented with two's complement. C is a 8-bit positive integer.

8) Calculate $C = \left\lfloor 2A + \frac{B}{3} \right\rfloor$ where A and B are 7-bit positive integers and C is a 8-bit positive integer.

9) Calculate arithmetic mean of three 8-bit positive integers, A, B and C as $D = \left\lfloor \frac{A+B+C}{3} \right\rfloor$. D is a 8-bit positive integer.

10) Calculate $C = A - 3B$, where A and B are 6-bit positive integers and C is a 8-bit sign integer.

Final Project

ALGORITHMIC PROBLEM SOLVING USING A SYSTEM CONSISTING OF AN ALU, A REGISTER BLOCK AND A STATE MACHINE

PROJECT STEPS

NOTE: Your system should **exactly** follow the design requirements given at page 2. Read them in accordance with project steps.

1) Construct an **ALU** as specified in design requirements. Place it in your top module.
2) By only using the provided operations in given ALU, design an algorithm that can solve your given problem. **The algorithm can only use the operations provided by given ALU. Also, note that you have register blocks and a possibility to generate constant values in CU block (CUconst signal) as well. Explain your algorithm steps clearly.**
3) Express your algorithm in the form of **Algorithmic State Machine (ASM)**; your design should be **Moore type.**
4) Design a **Control Unit (CU)** that realizes this ASM, via behavioral coding. Be sure to properly assign values to all of your registers.
5) Prepare a testbench and show that your design works properly.
6) You should include both your testbench code and your output waveform which shows that your algorithm works as desired. Show all different cases and corner values (overflowing, max possible value etc.), and support them with an ample number of examples. If your input set is small, consider showing results for all possible input combinations. **Do not just put a screenshot of waveforms, explain your results clearly.**
7) Include design RTL schematics in your report. You may include them as separate .pdf files if you are having a hard time fitting the visual in the report document.
8) Prepare a report that will include all requested work. You are expected to go by the experiment report format and rules given before in Ninova. Include a work package table to show specific tasks performed by each group member.

DESIGN REQUIREMENTS

1) Your TOP module should look exactly like Figure-1. **Don't use any other elements or submodules except these three given submodules.**
2) An example Register Block (RB) is given in Figure-2. **Make your design as modular as possible (Use structural design approach when viable), so your RTL schematic could look exactly like Figure-2.**
3) You may use "dont_touch" or “keep = true” constraint with your output registers, if you find them getting trimmed during synthesis.
4) CU block will contain your ASM (must be Moore type) that will control the solution steps for your assigned problem. **CU must only contain output registers and state registers, there should be no other registers within this design.**

5) CU should begin working when "start" signal is high. Then, it will keep "busy" signal high until the result is calculated.
6) Reset and clock inputs are common for CU and RB. Reset should be designed as asynchronous active-1.
7) State transition branches in your CU should only use the ALU signals Carry-Out (CO) and Zero (Z) as branching conditions. ex: if(CO), if(!Z).
8) When designing the ALU, you must use submodules just as given in Figure-3. **If you need to use combinational assignments, use them within AND, XOR, ADD or Shift blocks. Use structural design approach when viable, so your RTL schematic could look exactly like Figure-3.**
9) RTL's must be consistent with the figures given, thus you must use the exact signal names given in figures.
10) Your ALU should perform:
    - ❖ AND, when InsSel == '00',
    - ❖ XOR, when InsSel == '01',
    - ❖ ADD, when InsSel == '10',
    - ❖ Shift, when InsSel == '11'.

    For ADD, carry-out bit should be assigned to CO output. Shift should perform a circular left-shift (MSB will move to LSB), and CO will hold the old MSB. CO should be 0 for AND and XOR operations. Z signal should be 1 only when the ALU output is zero, will be 0 otherwise.

APPENDIX

Group #	Question
1	7
2	6
3	9
4	8
5	5
6	4
7	3
8	2
9	10

Table 1: Assigned Problems