Implementation of a Simple CPU

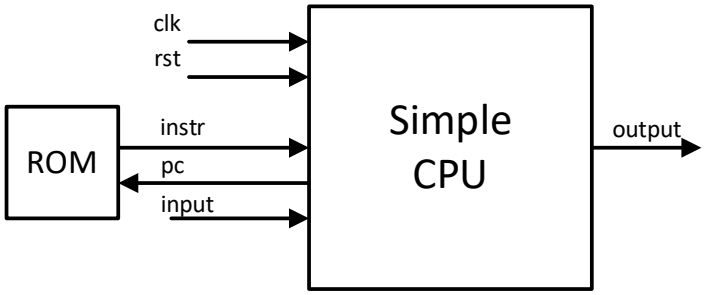
A Reduced Instruction Set Computer (RISC) Central Processing Unit (CPU) executes operations taking operands and saving results from/into a set of registers. It is required to design a digital circuit for implementing a simple RISC CPU supporting this reduced instruction set:

- RCSR Rx, CSR — Write in Rx, the content of the status register CSR
- IN Rx — Read the input into the register Rx
- OUT Rx — Write the content of register Rx to output
- MOV Ry, Rx — Move the content of register Rx into register Ry
- ADD Ry, Rx — Put into Ry the sum of Rx and Ry
- MUL Ry, Rx — Put into Ry the multiplication of Rx and Ry
- LSL Rx — Put into Rx, Rx shifted to the left of 1 position
- LSR Rx — Put into Rx, Rx shifted to the right of 1 position

Registers Rx and Ry are chosen between 4 8-bit registers (from R0 to R3). Instructions, whose format must be chosen and motivated, are taken autonomously from a 16x8-bit ROM memory. The CPU must implement three 8-bit Control Status Registers (CSR):

- Status Register (SR)
  - Bit 3: Z — Zero flag: 1 if MUL or ADD give 0 as result
  - Bit 2: OF — Overflow flag: 1 if MUL or ADD results are not representable
  - Bit 1: SH — Shift bit: contain the bit shifted by LSL and LSR operations
  - Bit 0: OP — Operation: 1 if the CPU is running
- Instruction Register (IR)
  - Contains instruction to decode
- Program Counter (PC)
  - Contains the next address of the instruction memory

The interface of the circuit to be designed is as follows:



where input and output are 8-bit wide, instr is 8-bit wide, and pc is 4-bit wide. All registers are set to 0 at reset. When the program counter reaches the last instruction, the CPU stalls, and the status register is provided as output. You are requested to deal with the various possible error situations, documenting the choices made.

The final project report must contain:

- Introduction (circuit description, possible applications, possible architectures, etc.)
- Description of the architecture designed (block diagram, inputs/outputs, etc.)
- VHDL code (with detailed comments) to be attached to the report.
- Test strategy (Test-plan) and related Testbench for verification; a detailed, though not exhaustive, verification is required, including error situations and borderline cases of functioning
- Interpretation of the results obtained in the automatic synthesis/implementation on a Xilinx FPGA platform in terms of maximum clock frequency (critical path), elements used (slice, LUT, etc.) and estimated power consumption. Comment on any warning messages.
- Conclusions