ECA-2: Homework assignment 2

Assignment: define in Haskell/CλaSH the variants described below of the core as introduced in the previous slide[1], i.e., a core with a *heap* and a *stack*.

1. Define a function *value* that yields the numerical value of elements of type *Value*.

   Note: this function needs the state (consisting of the *heap* as extra parameter, in order to calculate the value that is stored at a specific address.

2. Define the functions *alu* and *core* such that *program0* given on the previous slide[2]is executed correctly. The function *core* should get *prog* for a list of instructions as its first rgument. You have to use a program counter to choose an instruction from the program *prog*.

3. Add an instruction **Send** that sends a value to the output (assume: non-negative). Thus, your definition of *core* should be in the form of a Mealy Machine (as input you may assume that there will only a clock tick as input, e.g., represented by the number 1). For all other instructions, the output should be -1.

4. Add a value **Top** to the type *Value* that *yields* the value on the top of the stack, and an instruction **Pop** that *removes* the top value of the stack. Extend the definition of *core* accordingly.

5. On slide 2[3]the functions *take* and *drop* are used to yield and remove (respectively) the top two values of the stack in a single clock cycle. Rewrite the definition of *core*, such that *take* and *drop* are replaced by using *Top* and *Pop*.

   Note: by using *Top* and *Pop* only *one* value at a time can be yielded or removed, so in order to replace the usage of *take* and *drop*, two clock cycles are needed. Hence, you will need one or more additional registers.

6. Assume that both the heap and the stack are lists of *fixed length*, i.e., their lengths may not be changed during the execution of a program. Hence, you will need a *stack pointer* that indicates the top of the stack, and that is changed when a value is pushed to or popped from the stack. That is to say, you will need a register for the stack pointer.

7. *Optional*: reformulate your code in term sof CλaSH, generate VHDL, and compile it with Quartus.

Deadline: Friday, February 3, 2017.
Submit your solution by email to Marco Gerards, m.e.t.gerards@utwente.nl.

[1] Refers to slide 4 from *HeaphAndStack.pdf*, presented during the lecture of Jauary 18, 2017. Available through *Blackboard*, as usual.

[2] Again refers to slide 4.

[3] Of the same slide set as above