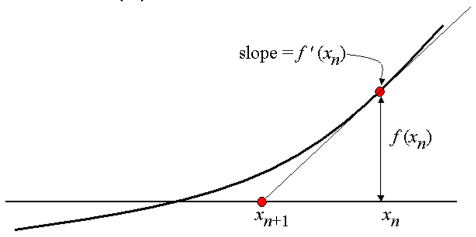
Design of generic square root computing architectures on FPGA using VHDL

The aim of this project is to design on FPGA different architectures implementing two different algorithms for integer square root computation with different design constraints. These architectures will be designed and tested using VHDL. They will compute the square root of an unsigned input A whose size is 2n bits.
The first considered algorithm is based on the Newton method, whose principle is shown on Figure 1, to compute the root of the f(x) = x^2 - A function.



$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Figure 1 : Newton’s method principle

Starting from an arbitrary initial guess x_0, a new result closer to the solution is obtained at each iteration using Newton’s formula. The algorithm stops when two consecutive iterations give the same result, which is the integer square root of A. Few iterations are required to get the result.

The second algorithm is an iterative algorithm computing sequentially the bits of the result, starting from the MSB, based on a modified non-restoring integer square root computing algorithm [1]. It is detailed on Figure 2.

```
D = A ;
R = 0 ;
Z = 0 ;
For i = n-1 downto 0 do
    If R>=0 then
        R = R * 4 + D / $2^{2n-2}$ – ( 4 * Z + 1 ) ;
    Else
        R = R * 4 + D / $2^{2n-2}$ +( 4 * Z + 3 ) ;
    End If ;
    If R>=0 then
        Z = 2 * Z + 1 ;
    Else
        Z = 2 * Z ;
    End If ;
    D = D*4 ;
End For ;
Result = Z;
```

Figure 2 : Considered iterative integer square root algorithm

The circuit must be useable as a coprocessor by an associated microprocessor. The A and Result ports must be declared using the std_logic_vector type even if they represent numbers. It will ease the association of the block to an automatically generated embedded processor. If the designed circuit is sequential, in addition to the *A* input, it shall then feature a *start* input that launches the computations when its value is 1. It will remain at 1 during the whole computation. Once the computation is over, a *finished* output must be set to 1 to indicate to the microprocessor that it can get the correct result. Once the microprocessor has read the result, the *start* signal is reset and a new A value may be sent for a new computation. Figure 3 summarizes the top-level view of the sqrt circuit.

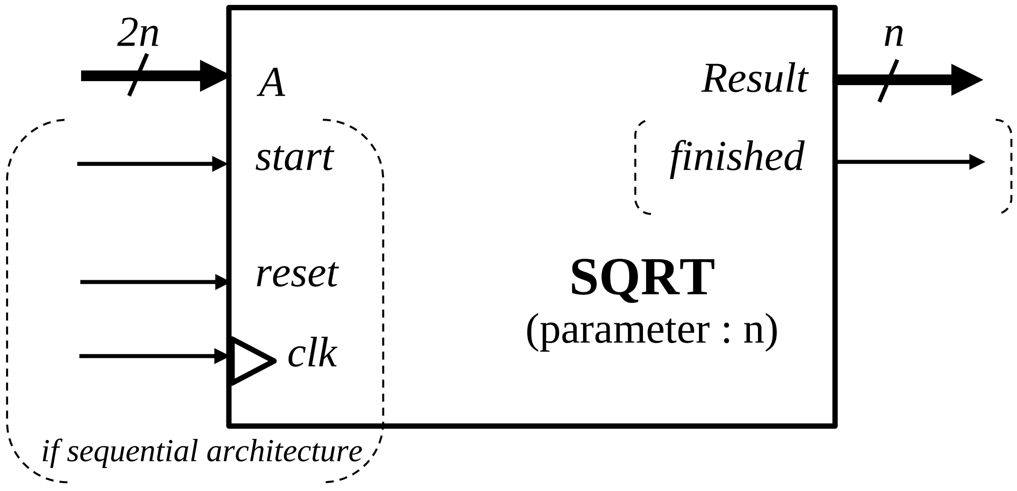


Figure 3 : Top level view of the sqrt circuit

You will have to design several architectures for this circuit and compare their performance in terms of computation duration and circuit size in the case n=32.
In order to do so, you will have, **for each architecture**, to determine the maximum working frequency, the worst-case required number of clock cycles to perform the computations and the resources used in the FPGA.
Of course, each architecture will have to be thoroughly tested beforehand, using testbenches written in VHDL. The test must perform at least the successive computations of the following A values : 0, 1, 512, 5499030, 1194877489. A specific test will be written for the pipeline architecture where the A value is modified at each clock cycle.
The project report must feature the simulation waveforms validating the working of each architecture together with the performance comparison.

1) Behavioral approach :

In this part, all the architectures will be designed in a behavioral way in VHDL **based on only one process**.

Architecture 1 : this architecture is sequential and aims at implementing the algorithm based on Newton's method by minimizing the number of states necessary to perform the computation. Each necessary iteration should be performed in one clock cycle.

Architecture 2 : this architecture is sequential and aims at implementing the second algorithm trying to minimize the number of clock cycles required to perform the computation. Rewrite the algorithm and implement the architecture so that each iteration of the for loop is performed in one clock cycle.

<u>Architecture 3</u> : Based on architecture 2, implement a combinatorial architecture for sqrt.

<u>Architecture 4</u> : Modify architecture 3 to get a pipelined architecture.

2) Structural approach :

The circuits obtained using high-level approach are dependent on the synthesis tools used. If you want to better control the architecture of the obtained circuit, it may be necessary to create a more detailed description of the circuit by explicitly instantiating the desired blocks. This approach will be used to implement the last architecture.

<u>Architecture 5</u> : This architecture is sequential, based on the second algorithm, and composed of a datapath featuring elementary blocks (like registers, multiplexers …) but only one arithmetic block (an adder/subtracter) to limit the circuit size, which is one of the main objectives of this architecture. You can refer to the article for the structure of the datapath or create your own. Associated to this datapath is a control unit (Finite State Machine) configuring the datapath properly at each computation step. Design all the required elementary blocks for the datapath using separate components for each of them (if time is too short, some could be borrowed from your teacher's catalog) and the corresponding control unit and instantiate all these blocks to obtain the desired architecture.

[1] "A New Non-Restoring Square Root Algorithm and Its VLSI Implementations", Yamin Li and Wanming Chu, ICCD'96, International Conference on Computer Design