Computer Aided Design Course Projects

Semester: 98-99/2

Project A1

Title: Square Root (SQRT) Operation in Floating-point Arithmetic System

Definition: Floating point operations are widely used in large set of scientific and signal processing computation. IEEE has set a standard for these operations numbered IEEE-754. In this project, part of a floating-point arithmetic operation unit needs to be developed and tested according to IEEE specifications. The operations need to be done on single and double precision floating point numbers. The concept of floating-point numbers needs to be understood well and implementation of the arithmetic unit should undergo exhaustive tests according to standards.

Specifications: According to IEEE-754-2008 for single (SP) and double precision (DP) floating point numbers

Blocks to develop (in optimized behavioral Verilog):

1. SQRT (Square Root Operation) (SP: 14 cycles, DP: 28 cycles)
    a. Find proper algorithm from web that is best suited for hardware implementation
2. Verilog Test Bench
    a. The data generated by golden model (read from a file) need to be cross-matched with what is generated in your developed hardware

Work:

1. Finding a golden model in C, Python or Java as a reference; whichever is easier for team
2. Unit test
    a. Verification against golden model
3. Synthesis and Implementation using Xilinx ISE Design Suite
    a. Target device may be chosen based on area requirements

Deliverables:

1. Verilog RTL at behavioral level
2. Testbench in Verilog according to the specifications. The standalone testbench should cover all possible cases.
3. All source codes and test-benches should be put in a compressed zip file
4. Simulation results in ModelSim simulator automatically written in a file
5. Performing synthesis on FPGA and generating meaningful reports
6. Performing Post-implementation timing simulation (Verilog Netlist + SDF);
    a. *This part as an extra mark*
7. Technical report in Persian/English

Project A2

Title: Division (DIV) Operation in Floating-point Arithmetic System

Definition: Floating point operations are widely used in large set of scientific and signal processing computation. IEEE has set a standard for these operations numbered IEEE-754. In this project, part of a floating-point arithmetic operation unit needs to be developed and tested according to IEEE specifications. The operations need to be done on single and double precision floating point numbers. The concept of floating-point numbers needs to be understood well and implementation of the arithmetic unit should undergo exhaustive tests according to standards.

Specifications: According to IEEE-754-2008 for single (SP) and double precision (DP) floating point numbers

Blocks to develop (in optimized behavioral Verilog):

1. DIV (Division Operation) (SP: 14 cycles, DP: 28 cycles)
    a. Find proper algorithm from web that is best suited for hardware implementation
2. Verilog Test Bench
    a. The data generated by golden model (read from a file) need to be cross-matched with what is generated in your developed hardware

Work:

1. Finding a golden model in C, Python or Java as a reference; whichever is easier for team
2. Unit test
    a. Verification against golden model
3. Synthesis and Implementation using Xilinx ISE Design Suite
    a. Target device may be chosen based on area requirements

Deliverables:

1. Verilog RTL at behavioral level
2. Testbench in Verilog according to the specifications. The standalone testbench should cover all possible cases.
3. All source codes and test-benches should be put in a compressed zip file
4. Simulation results in ModelSim simulator automatically written in a file
5. Performing synthesis on FPGA and generating meaningful reports
6. Performing Post-implementation timing simulation (Verilog Netlist + SDF);
    a. *This part as an extra mark*
7. Technical report in Persian/English

Project B1

Title: Pipeline CORDIC Algorithm in Vectoring Mode

Definition: CORDIC (COordinate Rotation DIgital Computer) is an iterative algorithm for the calculation of the rotation of a two-dimensional vector, in linear, circular and hyperbolic coordinate systems, using only add and shift operations. Its current application are in the field of digital signal processing, image processing, filtering, matrix algebra, etc.. CORDIC is an iterative algorithm for the calculation of the rotation of a two-dimensional vector, in linear, circular and hyperbolic coordinate systems, using only add and shift operations. It consists of two operating modes, the rotation mode (RM) and the vectoring mode (VM). In the rotation mode a vector (X, Y) is rotated by an angle θ to obtain a new vector (X' , Y'). In the vectoring mode, the length R and the angle α towards the x-axis of a vector (X, Y) are computed. For this purpose, the vector is rotated towards the x-axis so that the y-component approaches to zero. The sum of all angle rotations is equal to the value of α, while the value of the x-component corresponds to the length R of the vector (X, Y).

Specifications: According to the references provided in project folder

Blocks to develop (in optimized behavioral Verilog):

3. CORDIC in Vectoring Mode (input and outputs are 16-bit vectors)
    a. Find the pipeline algorithm from web that is best suited for hardware implementation
    b. Cover full range (360 degree)
4. Verilog Test Bench
    a. The data generated by golden model (read from a file) need to be cross-matched with what is generated in your developed hardware

Work:

4. Finding a golden model in C, Python or Java as a reference; whichever is easier for team
5. Unit test
    a. Verification against golden model
6. Synthesis and Implementation using Xilinx ISE Design Suite
    a. Target device may be chosen based on area requirements

Deliverables:

8. Verilog RTL at behavioral level
9. Testbench in Verilog according to the specifications. The standalone testbench should cover all possible cases.
10. All source codes and test-benches should be put in a compressed zip file
11. Simulation results in ModelSim simulator automatically written in a file
12. Performing synthesis on FPGA and generating meaningful reports
13. Performing Post-implementation timing simulation (Verilog Netlist + SDF);
    a. *This part as an extra mark*
14. Technical report in Persian/English

Project B2

Title: Pipeline CORDIC Algorithm in Rotation Mode

Definition: CORDIC (COordinate Rotation DIgital Computer) is an iterative algorithm for the calculation of the rotation of a two-dimensional vector, in linear, circular and hyperbolic coordinate systems, using only add and shift operations. Its current application are in the field of digital signal processing, image processing, filtering, matrix algebra, etc.. CORDIC is an iterative algorithm for the calculation of the rotation of a two-dimensional vector, in linear, circular and hyperbolic coordinate systems, using only add and shift operations. It consists of two operating modes, the rotation mode (RM) and the vectoring mode (VM). In the rotation mode a vector (X, Y) is rotated by an angle θ to obtain a new vector (X' , Y'). In the vectoring mode, the length R and the angle α towards the x-axis of a vector (X, Y) are computed. For this purpose, the vector is rotated towards the x-axis so that the y-component approaches to zero. The sum of all angle rotations is equal to the value of α, while the value of the x-component corresponds to the length R of the vector (X, Y).

Specifications: According to the references provided in project folder

Blocks to develop (in optimized behavioral Verilog):

1. CORDIC in Rotation Mode (input and outputs are 16-bit vectors)
    a. Find the pipeline algorithm from web that is best suited for hardware implementation
    b. Cover full range (360 degree)
2. Verilog Test Bench
    a. The data generated by golden model (read from a file) need to be cross-matched with what is generated in your developed hardware

Work:

1. Finding a golden model in C, Python or Java as a reference; whichever is easier for team
2. Unit test
    a. Verification against golden model
3. Synthesis and Implementation using Xilinx ISE Design Suite
    a. Target device may be chosen based on area requirements

Deliverables:

1. Verilog RTL at behavioral level
2. Testbench in Verilog according to the specifications. The standalone testbench should cover all possible cases.
3. All source codes and test-benches should be put in a compressed zip file
4. Simulation results in ModelSim simulator automatically written in a file
5. Performing synthesis on FPGA and generating meaningful reports
6. Performing Post-implementation timing simulation (Verilog Netlist + SDF);
    a. *This part as an extra mark*
7. Technical report in Persian/English