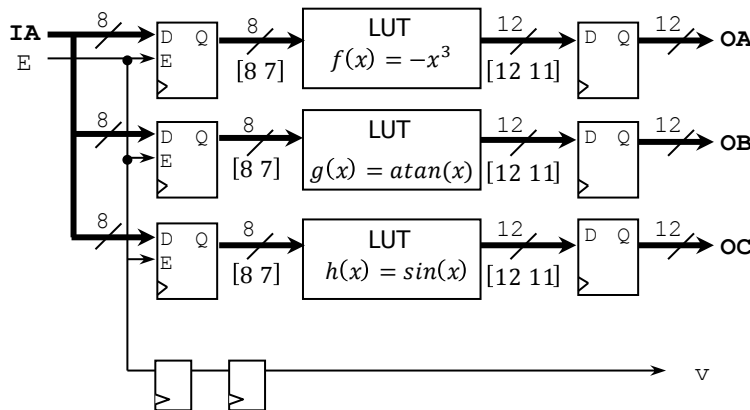# Solutions - Homework 3

(Due date: March 19th @ 7:30 pm)
Presentation and clarity are very important! Show your procedure!

PROBLEM 1 (15 PTS)
- LUT approach: We want to implement the following system.
- The purpose of this exercise is to explore how to rapidly fill up LUT values and verify the correct operation.
  Here, you are asked not to write VHDL code, but rather to setup the parameters for it, synthesize, and simulate.

- The figure shows three 3 LUT 8-to-12, where each LUT holds the pre-computed results of 3 functions:
  - ✓ Input format: [8 7] (signed). Input data range: $[-1,1)$.
  - ✓ Output format: [12 11] (signed). Output data range: $[-1,1)$.

- The VHDL code and testbench for this system can be found here.
  ```
  test.vhd
     LUT_group.vhd → File that includes all the LUTs.
         LUT_NItoNO.vhd → File that implement one LUT.
     dffe.vhd

  atb_test_sim.vhd → Testbench
  ```
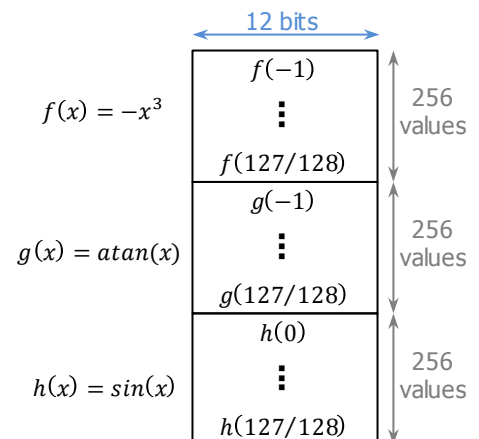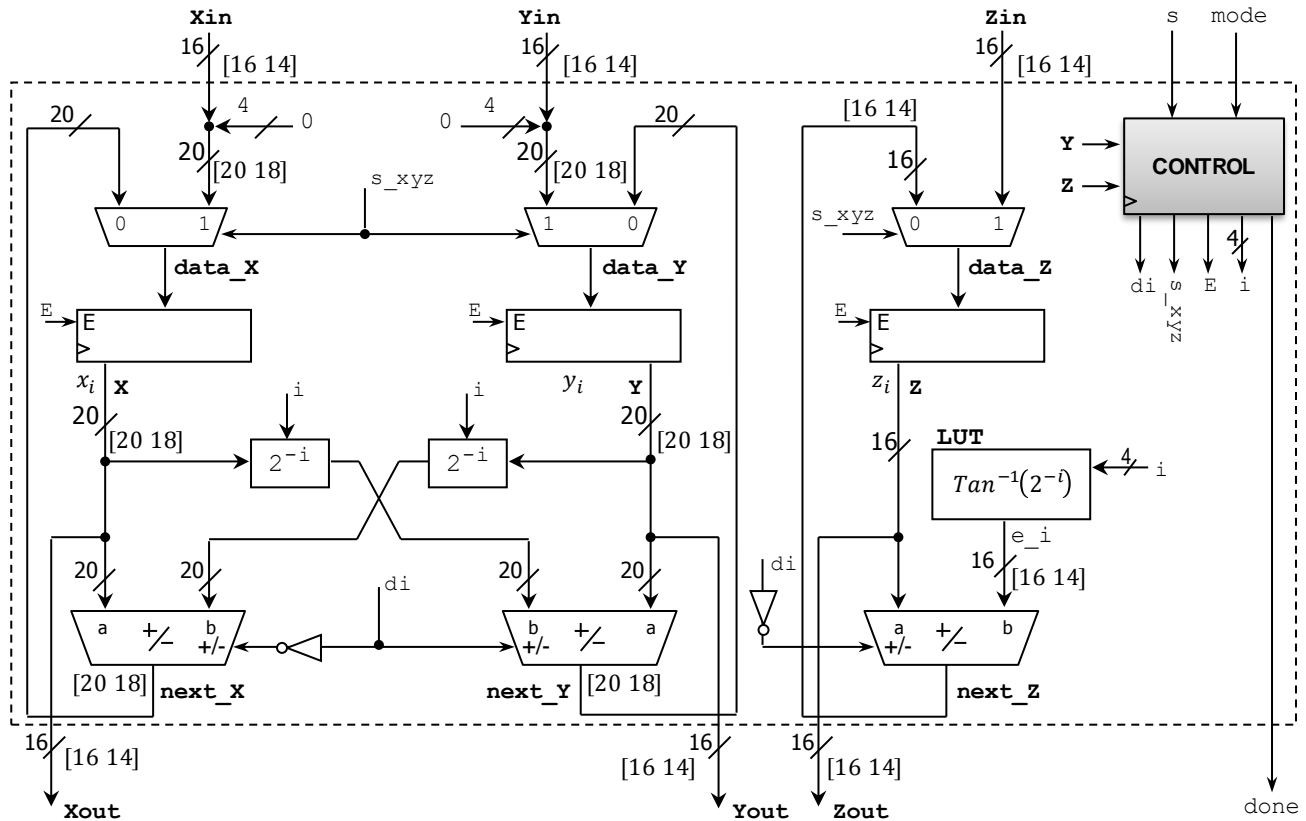

PROCEDURE
- Select the proper parameters to implement the circuit shown. These need to be indicated in test.vhd and in atb_test_sim.vhd.
- Generate the text file (LUT_values8to12.txt) that contains the pre-computed values. The filename is a parameter of LUT_group.vhd. The text file is divided as follows: the first 256 entries for the first function, the second 256 entries for the second function, and the third 256 entries for the third function. An 'L' separator is included between groups of 256 entries.
  You can use the provided MATLAB script (LUTvalGen8to12.m) to generate this file. This script requires the FX converter.

- Create a Vivado project and synthesize your circuit.
- Perform Functional Simulation:
  - ✓ The provided testbench atb_test_sim.vhd will generate all possible input cases (from 00000000 to 11111111) and write the output results in a text file. The testbench writes three 12-bit words per line (256 lines), where each 12-bit word represents the output of a different function.
  - ✓ Simulate the circuit until all the 256 input cases are processed. To verify the correct operation of your circuit, compare the text file generated by the Simulation with the input text file you created for Synthesis.

- Upload (as a .zip file) the following files to Moodle (an assignment will be created). DO NOT submit the whole Vivado project.
  - ✓ VHDL code, VHDL testbench: These files will be modified with the proper VHDL parameters.
  - ✓ Input text file (pre-computed function values) and output text file (results).
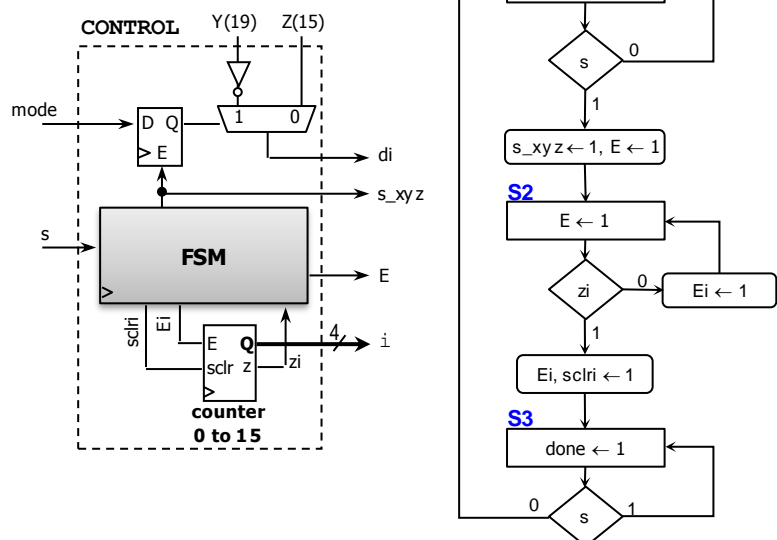
▪▪▪

See attached .zip file: SolutionsHW3p1.zip.

PROBLEM 2 (65 PTS)

- Design (write the VHDL code) for the iterative Circular CORDIC FX architecture with 16 iterations. $i = 0, 1, 2, 3, \dots 15$. x_0, y_0, z_0: initial conditions. $mode = $ '0' $\rightarrow$ Rotation Mode. $mode = $ '1' $\rightarrow$ Vectoring Mode. (35 pts)
- **Operation**: When $s = 1$, x_{in}, y_{in}, z_{in} and $mode$ are captured. Data will then be processed iteratively. When data is ready ($done = $ '1'), output results appear in $x_{out}, y_{out}, z_{out}$.
- **Input/Intermediate/Output FX Format**:
  ✓ Input values: x_{in}, y_{in}, z_{in}: [16 14].   Output values: $x_{out}, y_{out}, z_{out}$: [16 14]
  ✓ Intermediate values: z_i: [16 14]. x_i, y_i: [20 18]. Here, we use 4 extra bits (add four 0's to the LSB) for extra precision.
  ✓ We restrict the inputs $x_0 = x_{in}, y_0 = y_{in}$ to $[-1,1)$. Then, CORDIC operations need up to 2 integer bits (determined via MATLAB simulation). For consistency, we use 2 integer bits for all input/intermediate/output data.
- **Angles**: They are represented in the format [16 14]. Units: radians. Pre-compute the values and store them in an LUT.
- **Barrel shifters**: Use the file `mybarrelshift_gen.vhd` with `SHIFTTYPE="ARITHMETIC"` (signed data), `N=20`, `SW=4`, `dir='1'`.

- **Control**: This circuit controls the iteration index i, as well as the internal signals:
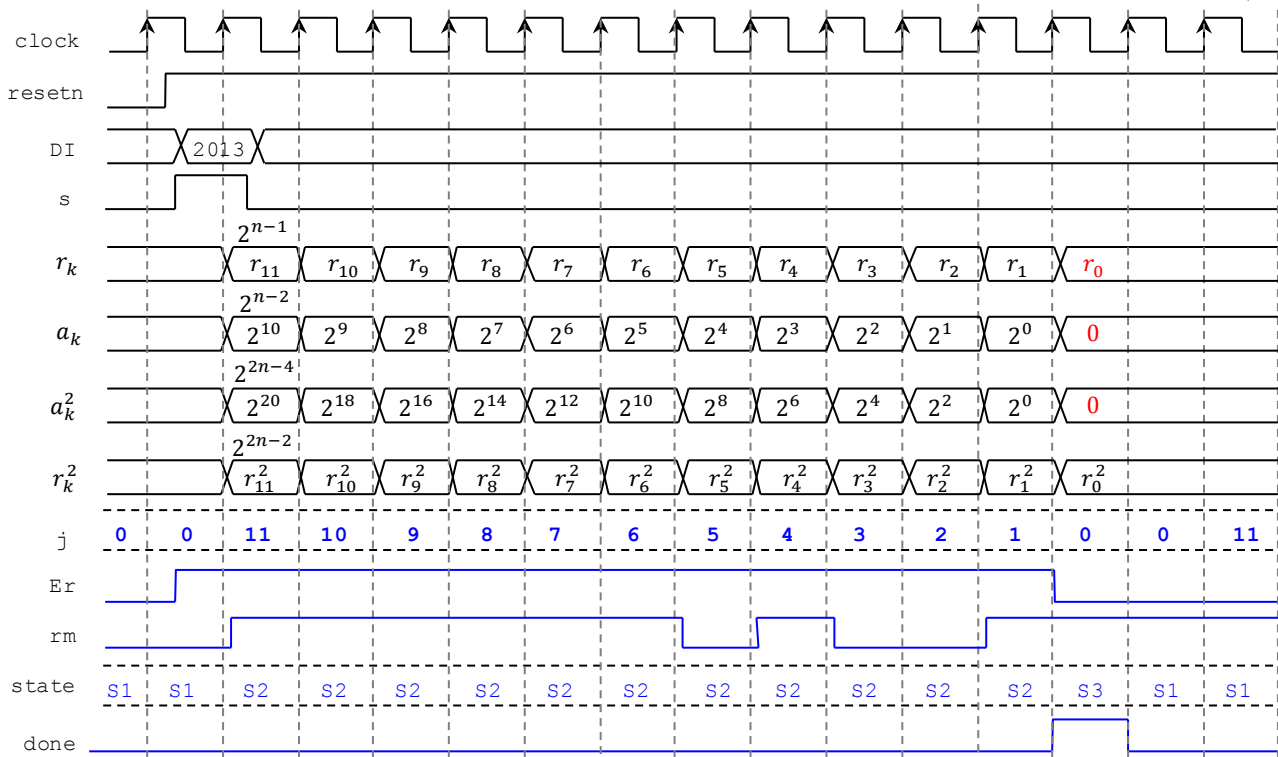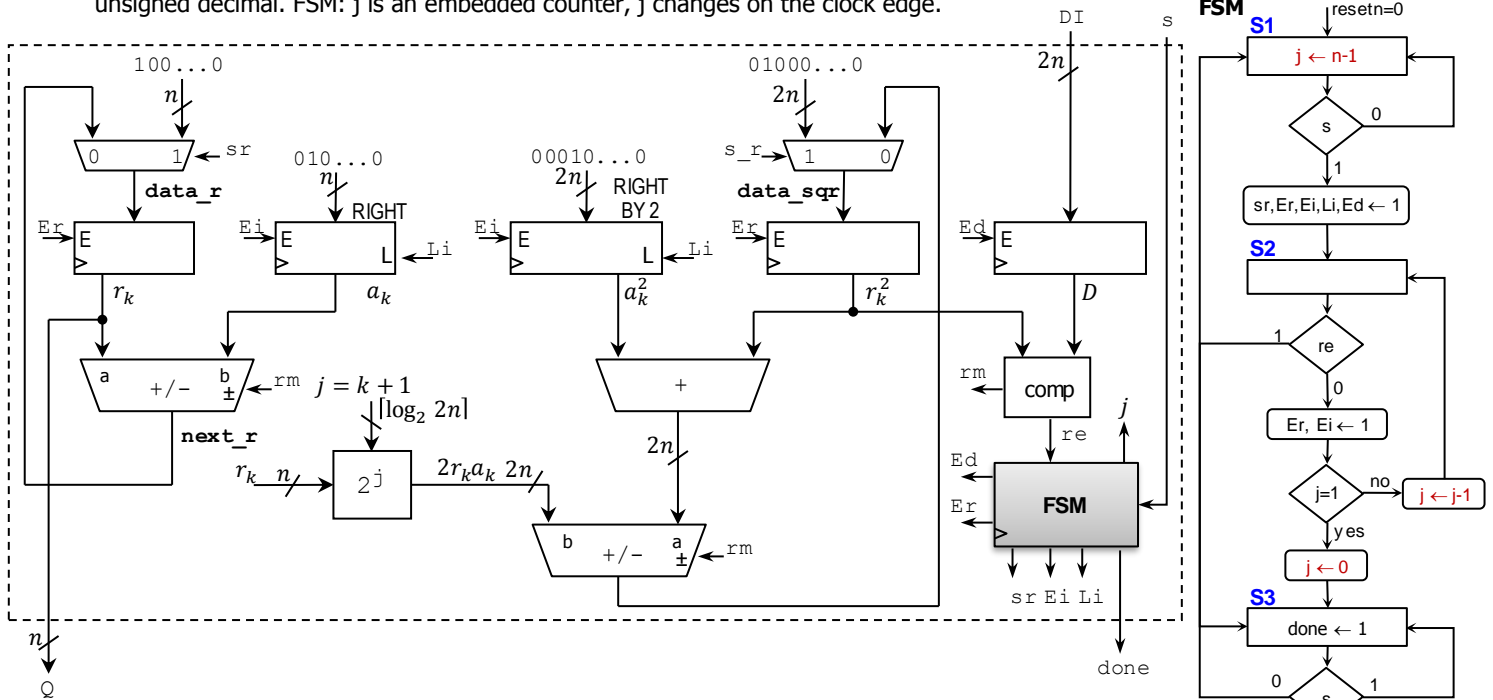
SIMULATION (Behavioral)

- A Circular CORDIC MATLAB/Octave model is available. Make sure to use the 'Basic CORDIC'. This model can be useful to verify the hardware output data as well as to generate input data and LUT angles represented in Fixed-Point arithmetic. The .zip file contains the following files:
  - ✓ run_examples_cordic.m (this file only works in MATLAB), cordic_circular.m, get_scalefactor.m.
  - ✓ my_dec2fx.m, my_fx2dec.m, my_bitcmp.m: This Fixed-Point to decimal converter is helpful to convert the LUT angles and input data to their Fixed-Point representation (binary).

- **First testbench**: Simulate the circuit for the following cases. You can use $A_n = 1.6468$. Convert the real numbers to the signed FX format [16 14]. For each case, verify that x_{16}, y_{16}, z_{16} reach the proper values. (10 pts)
  - ✓ Rotation Mode: $x_0 = 0, y_0 = 1/A_n, z_0 = \pi/6$.
  - ✓ Rotation Mode: $x_0 = 0, y_0 = 1/A_n, z_0 = -\pi/3$.
  - ✓ Vectoring Mode: $x_0 = y_0 = 0.8, z_0 = 0$
  - ✓ Vectoring Mode: $x_0 = 0.5, y_0 = 1, z_0 = 0$

- **Second Testbench**: Simulate the circuit reading input values (x_0, y_0, z_0) from input text files and writing output values (x_{16}, y_{16}, z_{16}) on an output text file. (20 pts). Your testbench must:

  - ✓ Read input values (x_0, y_0, z_0) from two input text files (provided):

    - ▫ in_benchR.txt: Data for Rotation Mode testing.
      20 data points (x_0, y_0, z_0). Data format: [16 14]. Each line per data point written as hexadecimals: |x0|y0|z0|.
      Data set: $x_0 = 0, y_0 = 1/A_n, z_0 = -\pi/2 \ to \ \pi/2$. z_0: 20 equally-spaced values between $-\pi/2 \ to \ \pi/2$.
      With this data set in the rotation mode, note that $x_{16} \to -sin(z_0)$, $y_{16} \to cos(z_0)$.

    - ▫ in_benchV.txt: Data for Vectoring Mode testing.
      20 data points (x_0, y_0, z_0). Data format: [16 14]. Each line per data point written as hexadecimals: |x0|y0|z0|.
      Data set: $x_0 = 0.0 \ to \ 0.5, y_0 = 1, z_0 = 0$. x_0: 20 equally-spaced values between $0.0 \ to \ 0.5$.
      With this data set in the vectoring mode, note that $x_{16} \to A_n\sqrt{x_0^2 + y_0^2}, z_{16} \to atan(y_0/x_0)$.

  - ✓ Write output results (x_{16}, y_{16}, z_{16}) on out_bench.txt. Data format: [16 14], each line per data point written as hexadecimals: |x16|y16|z16|. The output text file should have 40 data points (20 from the rotation mode and 20 from the vectoring mode).

  - ✓ Vivado tips:
    - ▫ Make sure that the input text files are loaded as simulation sources.
    - ▫ The output text file should appear in sim/sim_1/behav.
    - ▫ To verify that the output results are correct, you need to represent data as fixed-point numbers. Use Radix → Real Settings in the Vivado simulator window.

  - ✓ Submit (as a .zip file) the generated files: VHDL code, VHDL testbenches, and output text file to Moodle (an assignment will be created). DO NOT submit the whole Vivado Project.

• •

Output text file (out_bench.txt): They were obtained with a very precise A_n value. To verify your values, just make sure that they approximate the real values of the functions that the CORDIC algorithm approximates to.

	Rotation	Vectoring		Rotation	Vectoring
First	DFFD376A0001	773CFFFF3241	11	FAB93FC70002	6CFB0000540F
testbench	376C2000FFFF	75D5FFFF46DB	12	F04A3E0AFFFD	6DB80000527D
1	3FFFFFFCFFFD	6964FFFF6483	13	E64A3A9B0000	6E86FFFF50F7
2	3F1F0A8B0000	696E000062D7	14	DCFF3594FFFF	6F6300004F6D
3	3C8814C60001	698A00006129	15	D4A52F13FFFE	7051FFFF4DED
4	38471E780000	69B800005F75	16	CD7E274EFFFF	714E00004C7B
5	3281274E0001	69F9FFFF5DD1	17	C7B71E750000	725AFFFF4B03
6	2B5A2F130002	6A4DFFFF5C2B	18	C37714C6FFFF	73750000499B
7	22FF3594FFFF	6AB200005A7F	19	C0DF0A880000	749E0000482F
8	19B23A9C0000	6B2A000058DB	20	C000FFFAFFFD	75D5FFFF46DB
9	0FB53E0A0003	6BB40000573B			
10	05463FC7FFFE	6C4EFFFF55A1			

PROBLEM 3 (10 PTS)

- Complete the timing diagram of the following circuit, which computes integer square root using a binary search approach. $n = 12$. Note that $rm = 1$ if $r_k^2 > D$, $else$ 0, $re = 1$ if $r_k^2 = D$, $else$ 0. Shift registers: serial input is '0'. The value of D is an unsigned decimal. FSM: j is an embedded counter, j changes on the clock edge.

r_{11}	r_{10}	r_9	r_8	r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0
2048	1024	512	256	128	64	32	48	40	44	46	45

PROBLEM 4 (10 PTS)

- Attach a printout of your Initial Project Report (no more than a page). This report should contain the project title, a brief project description, and the current status of the project, including a block diagram of your system. Use the provided template (`Final Project – Report Template.docx`).