Assignment 3
Simulink and Model Composer Models and Report

Course Code: ENEL 864
Course Title: FPGA Design Applications

Submitted to:
Dr. Lei Zhang
Associate Professor
Electronic Systems Engineering

Submitted by:
MARUF AHMAD
ID: **200460690**
Email: mah370@uregina.ca
MASc in Electronic Systems Engineering
The University of Regina.

Date of Submission: **22 November 2022**

In the phase plane, an oscillating neuron E(t) can be written by the complex exponential form as (1),

$$E(t) = e^{i\omega t+\theta} = e^{\theta}e^{i\omega t} = e^{\theta}(\cos\omega t + i\sin\omega t)$$
(1)[1]

Where ω is the angular frequency, the real component θ is used to make the oscillation amplitude.

The synaptic weight can also be represented by the complex exponential form as $W = e^{i\phi}$. Where ∅ represents the phase delay of the neural connection.

Therefore a weighted input can be represented as E(t).W = $e^{i\omega t+\theta}e^{i\phi} \;=\; e^{\theta}e^{i(\omega t+\phi)}$.

In the neural network, pre-synaptic neurons are multiplied by their corresponding weight and then summed up to the post-synaptic neuron. This summation (S) in the post-synaptic neuron can also be represented by a complex exponential form as in (2).

$$\begin{aligned} S &= E_1W_1 + E_2W_2 \\ &= e^{i\omega_1 t+\theta_1}e^{i\phi_1} + e^{i\omega_2 t+\theta_2}e^{i\phi_2} \\ &= e^{\theta_1}e^{i(\omega_1 t+\phi_1)} + e^{\theta_2}e^{i(\omega_2 t+\phi_2)} \end{aligned}$$
(2)

Three parameters ω, Ɵ and ∅ are used to calculate the weighted sum.
In this report the equation of the complex exponential function is represented by

$$e^{i\phi} = cos\phi + isin\phi \quad (3)$$

In this report, I have presented three ways of implementing the weighted sum of two neurons.

In the first approach, the complex exponential function of the weighted sum of two neurons are implemented using the lookup table method where Block-RAM HDL Blocks from the Simulink Xilinx toolbox are used. Three separate designs are implemented using 8, 16, and 32-bit fixed-point data format configurations respectively and then generate IP core using Xilinx System Generator.

In the second approach, a CORDIC 6.0 HDL block from the Xilinx toolbox in model composer in Simulink is used to get the value of the exponential function (sinɸ and cosɸ). In this case, also, three separate designs are implemented using 8, 16, and 32-bit fixed-point data formate configurations respectively.

In the third approach, the weighted sum of two neurons is directly implemented in VHDL using CORDIC 6.0 IP core from the Xilinx Vivado design suite. In this approach also the IP core is configured with 8, 16, and 32-bit fixed-point data format and three separate projects are created respectively.

Finally, the above outputs are compared with the model design implemented in MATLAB Simulink.

In all approaches below configuration are used.

Inputs	$E_1 ✖ W_1 = e^{iw_1 t+\theta_1} e^{i\varnothing_1}$			$E_1 ✖ W_1 = e^{iw_2 t+\theta_2} e^{i\varnothing_2}$		
	w_1	θ_1	$\varnothing_1$	w_2	θ_2	$\varnothing_2$
	1	log(2)	pi/2	2	log(2)	pi/2

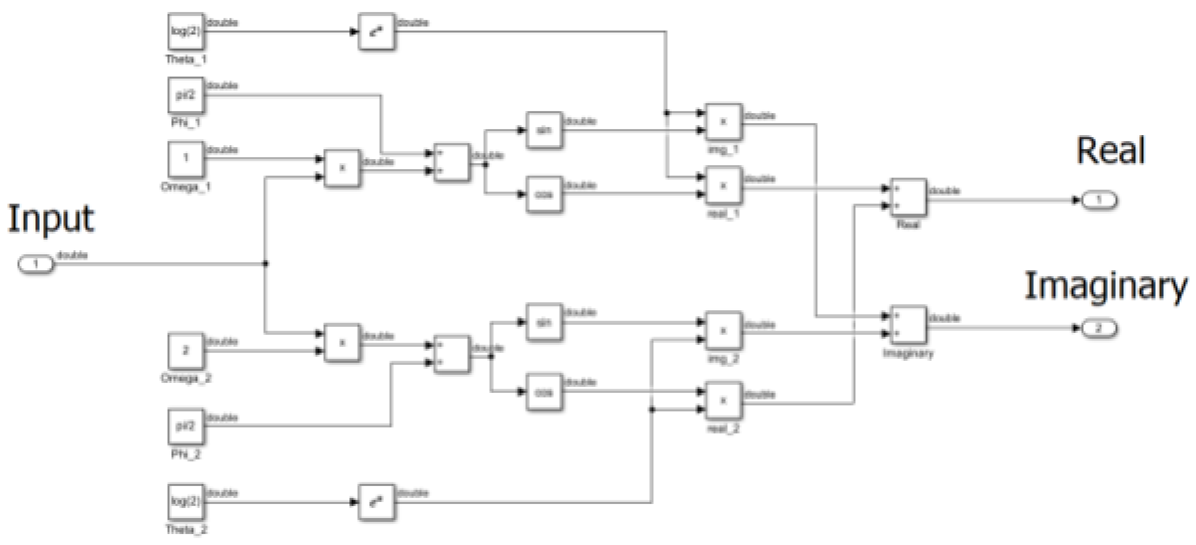
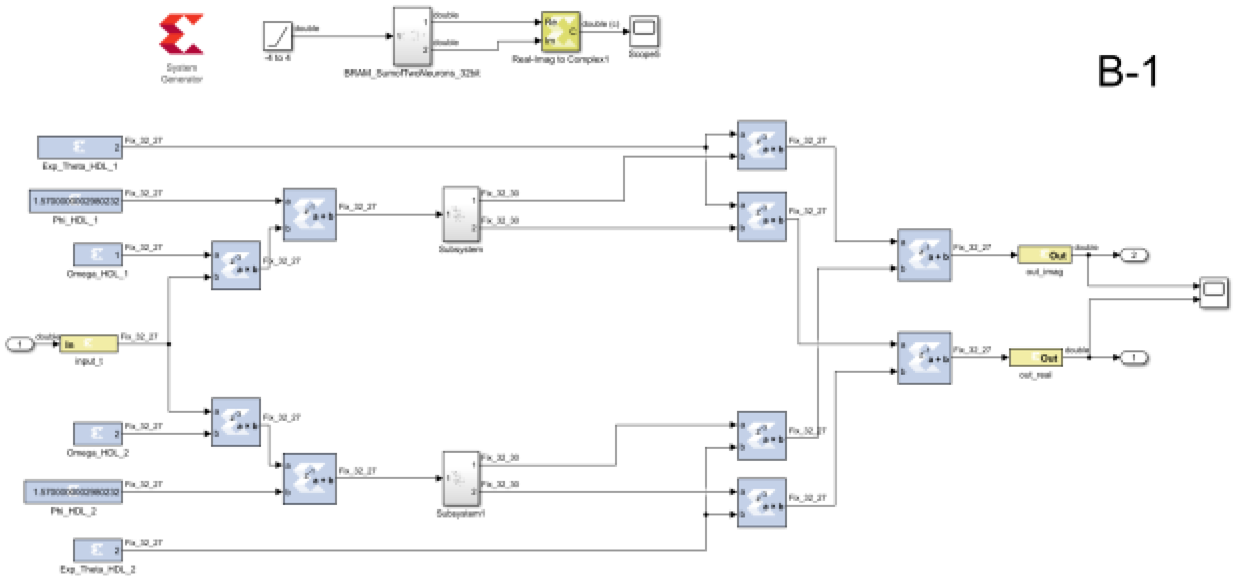
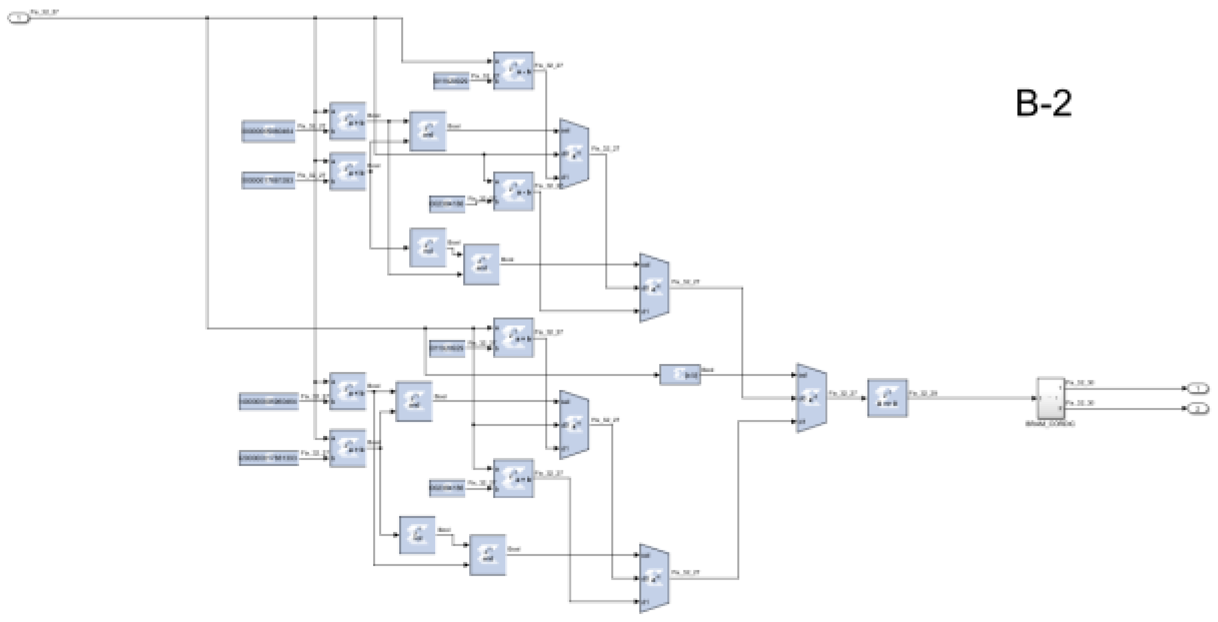


Figure 1: MATLAB simulink design of the weighted sum of two neurons

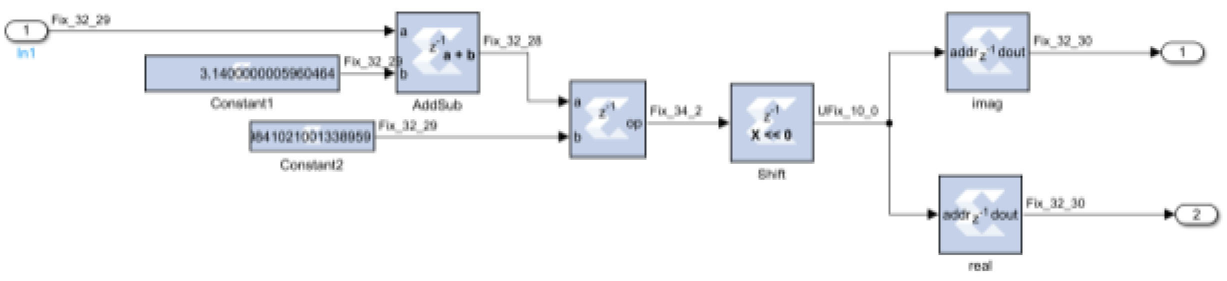
1. BRAM based Design in Model composer:


System Generator
double
-4 to 4
double
double
BRAM_SumofTwoNeurons_32bit
Re
Im
double (c)
Real-Imag to Complex1
Scope6
Fix_32_27
Exp_Theta_HDL_1
1.5700000002980232
Phi_HDL_1
Omega_HDL_1
a + b
Fix_32_30
Subsystem
double
In
input_1
Omega_HDL_2
Phi_HDL_2
Subsystem1
Exp_Theta_HDL_2
Out
out_imag
out_real
1
2

B-1



B-2


1
In1
Fix_32_29
3.1400000005960464
Constant1
841021001338959
Constant2
a
b
z^{-1}
a + b
AddSub
Fix_32_28
op
Fix_34_2
X << 0
Shift
UFix_10_0
addr
dout
imag
Fix_32_30
real
1
2

B-3

Figure 2: Lookup table method-based design using Block-RAM of the Xilinx System generator Toolbox. (B-1: weighted sum of two neurons, B-2: normalization block, B-3: Lookup table block)

2. CORDIC IP-based design in model composer:

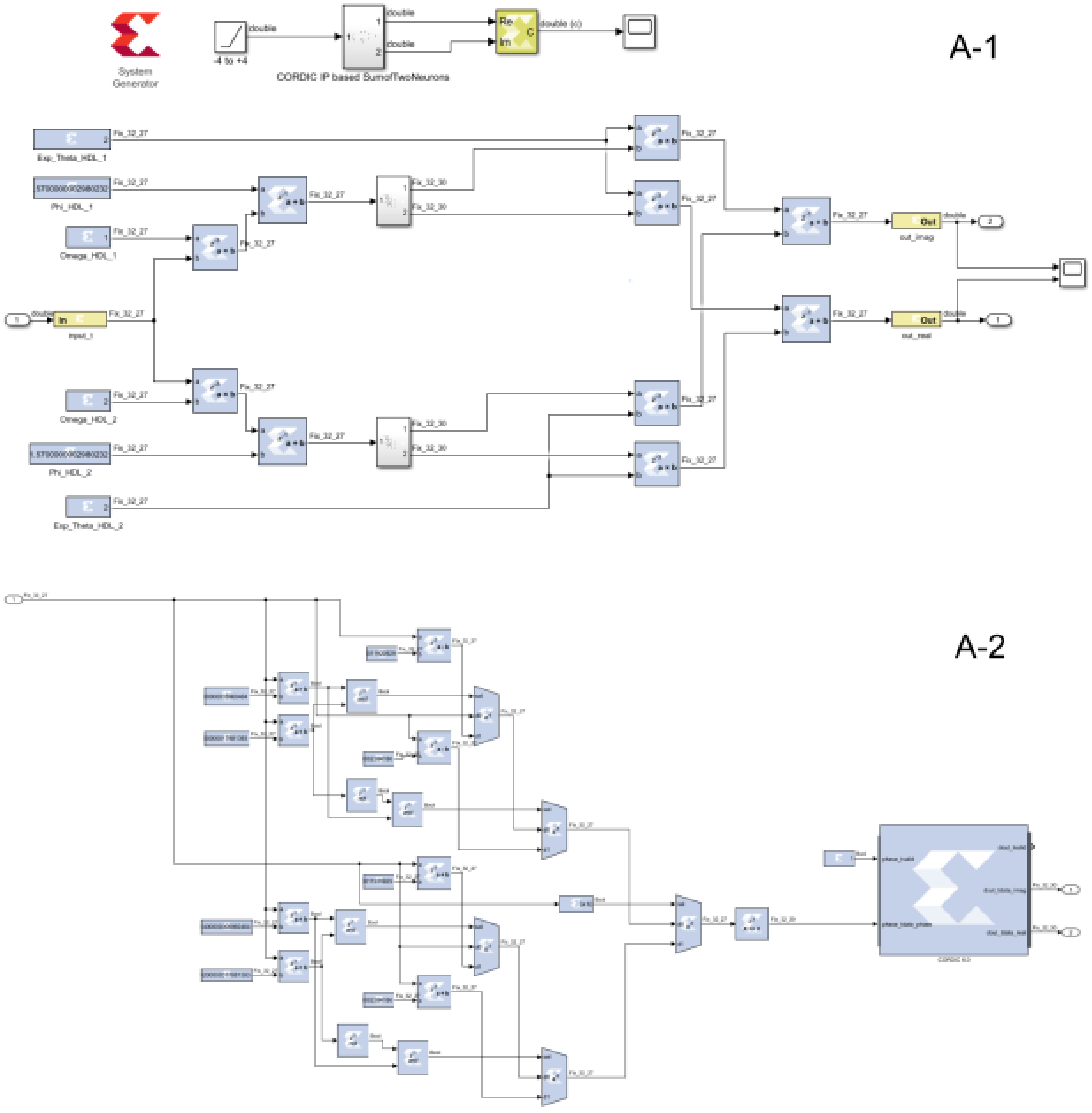


Figure 3: CORDIC IP-based design in model composer

3. Direct VHDL implementation of the weighted sum of two neurons:

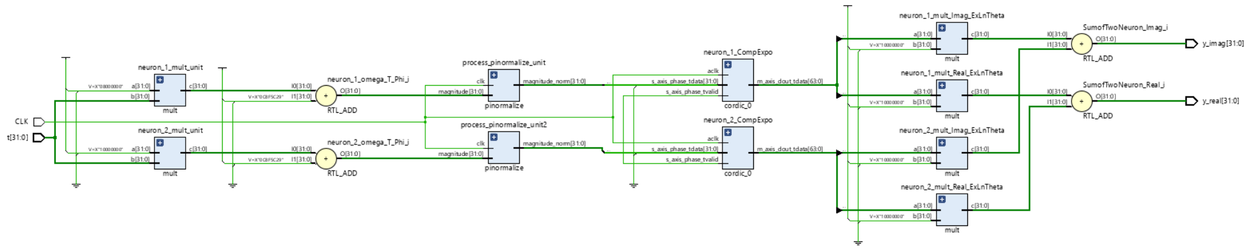
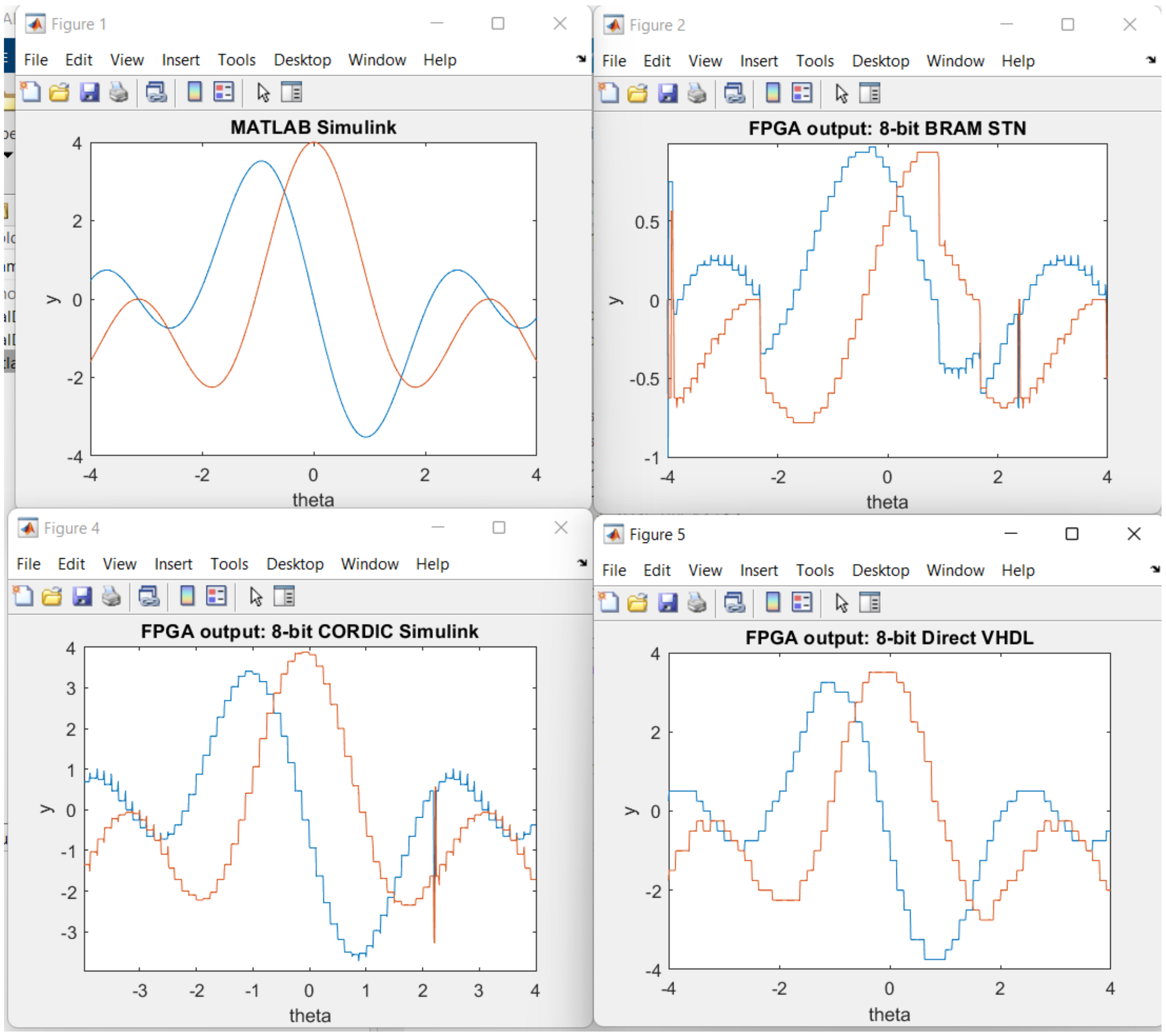
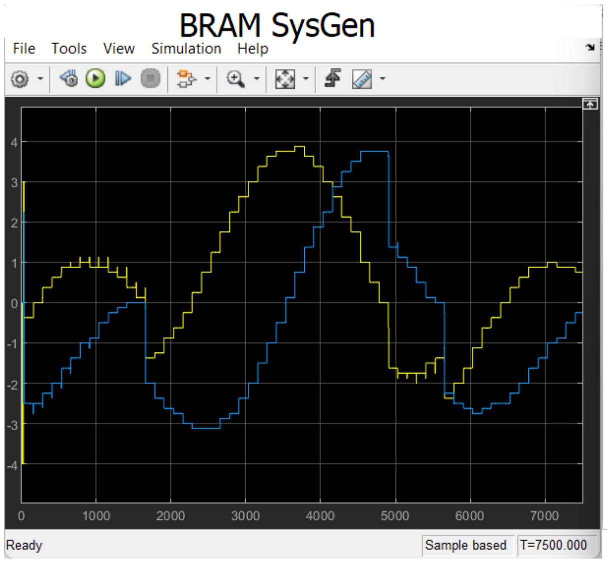
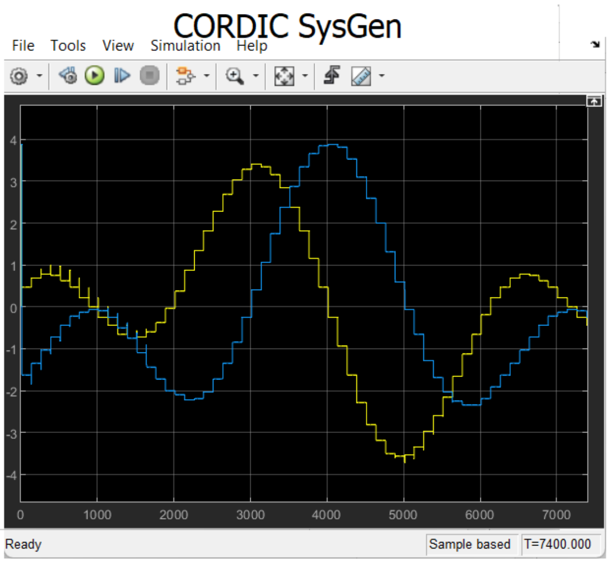


Figure 4: Direct VHDL implementation of the weighted sum of two neurons using CORDIC IP core

8-bit Fixed-point implementation result: Fix_8_3







16-bit Fixed-point implementation result: Fix_16_11:

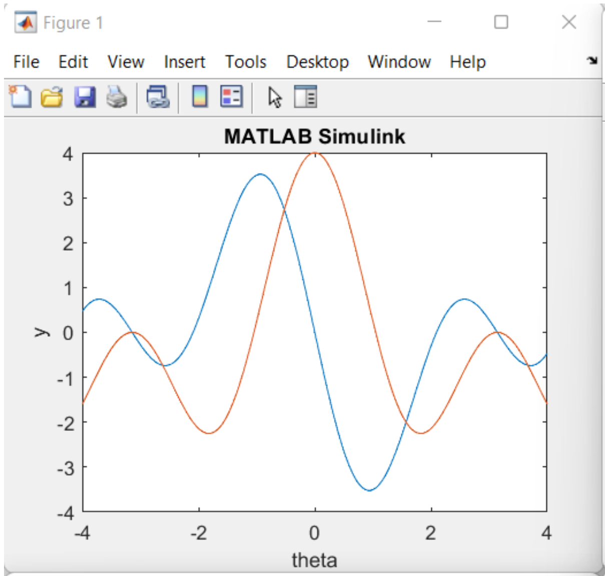


Figure 1
File Edit View Insert Tools Desktop Window Help
MATLAB Simulink
y
theta

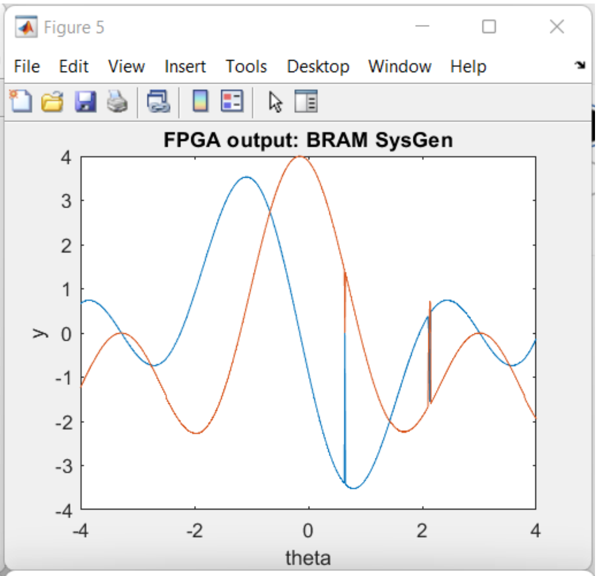


Figure 5
File Edit View Insert Tools Desktop Window Help
FPGA output: BRAM SysGen
y
theta

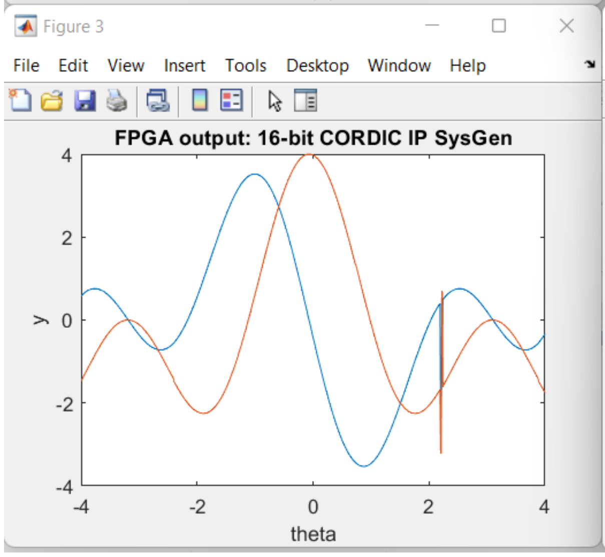


Figure 3
File Edit View Insert Tools Desktop Window Help
FPGA output: 16-bit CORDIC IP SysGen
y
theta

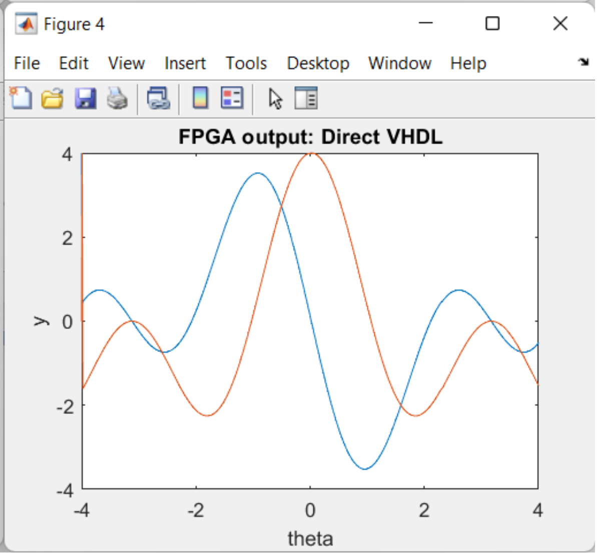
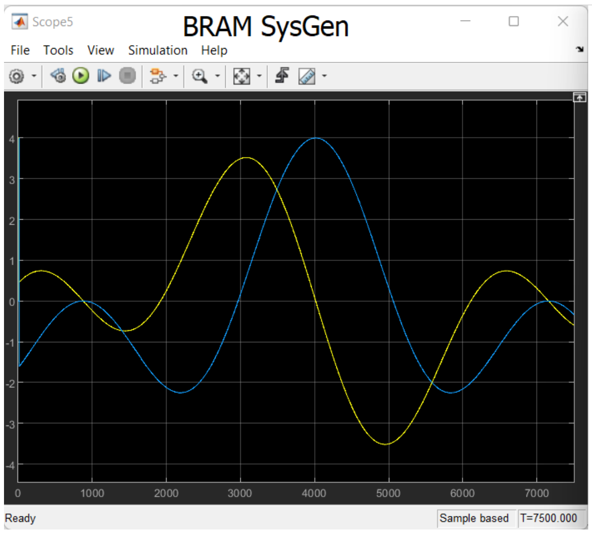
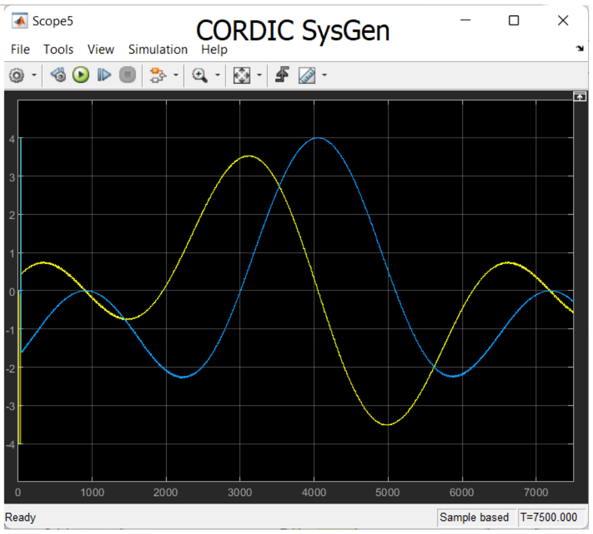
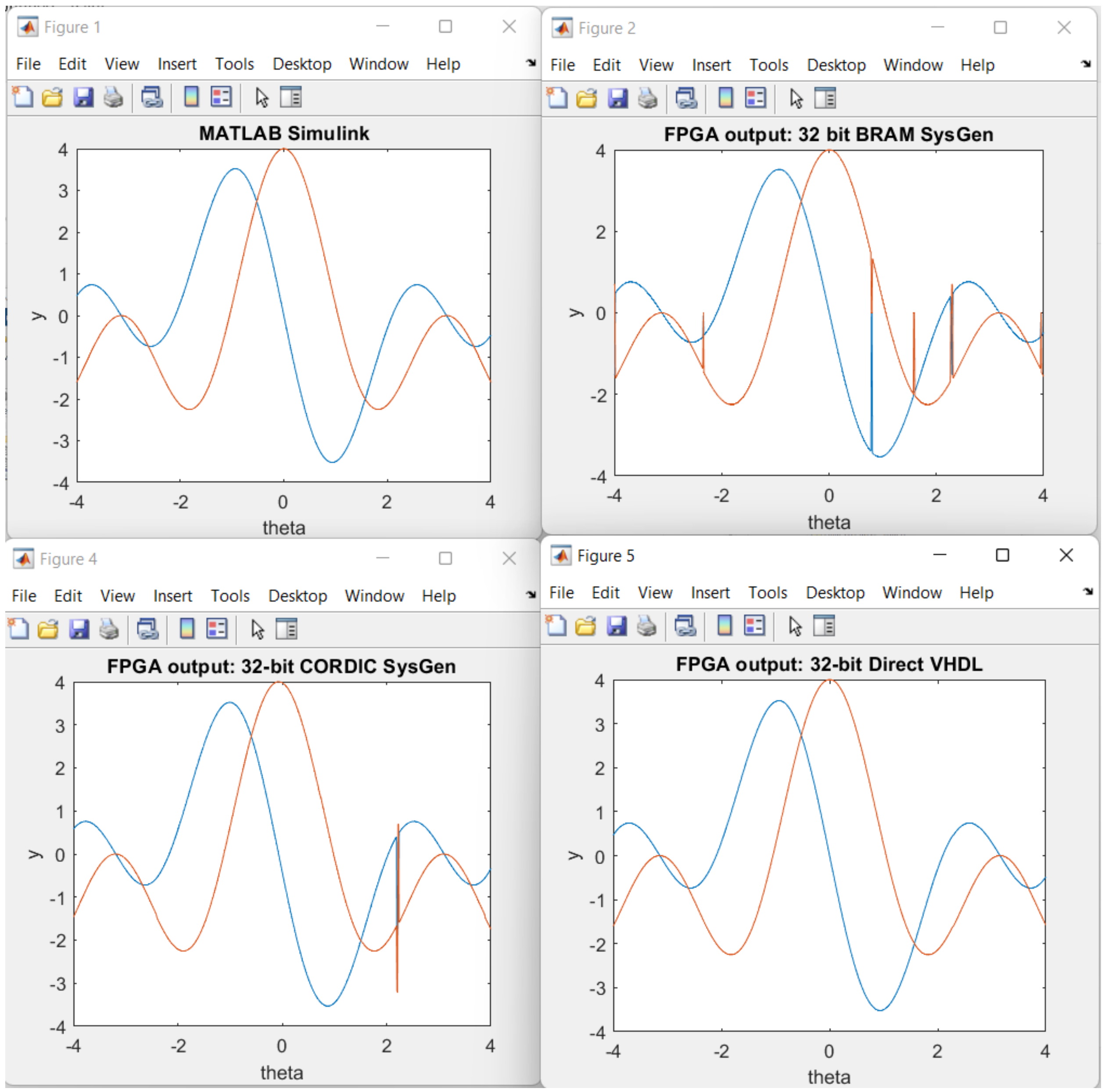


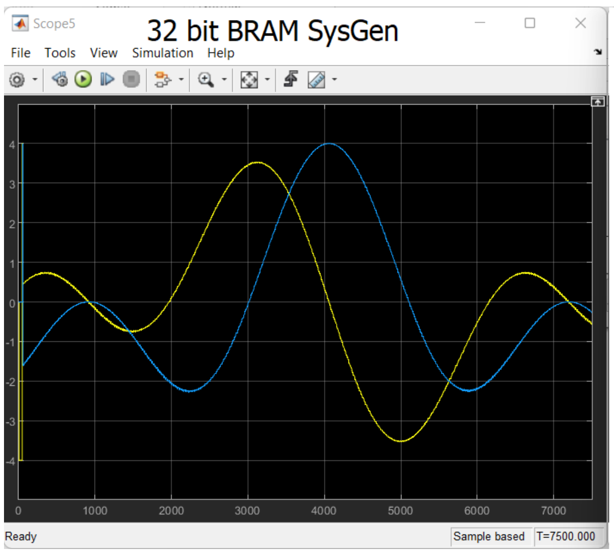
Figure 4
File Edit View Insert Tools Desktop Window Help
FPGA output: Direct VHDL
y
theta

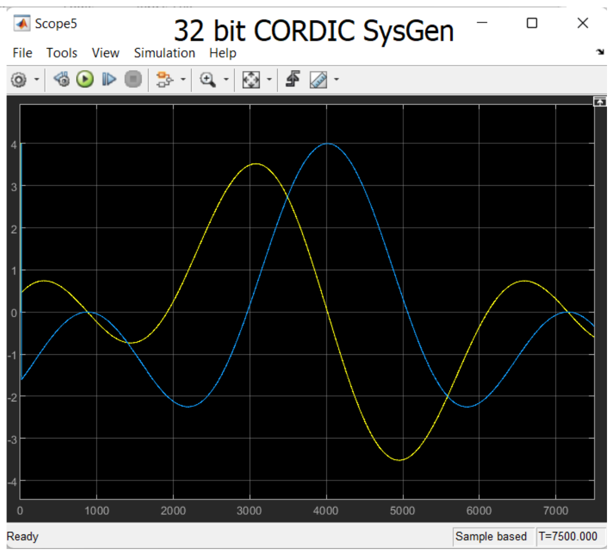

Scope5
BRAM SysGen
File Tools View Simulation Help
Ready
Sample based
T=7500.000


Scope5
CORDIC SysGen
File Tools View Simulation Help
Ready
Sample based
T=7500.000

32-bit Fixed-point implementation result: Fix_32_27:







Mean squared error in direct VHDL design with 8, 16, and 32 bits:

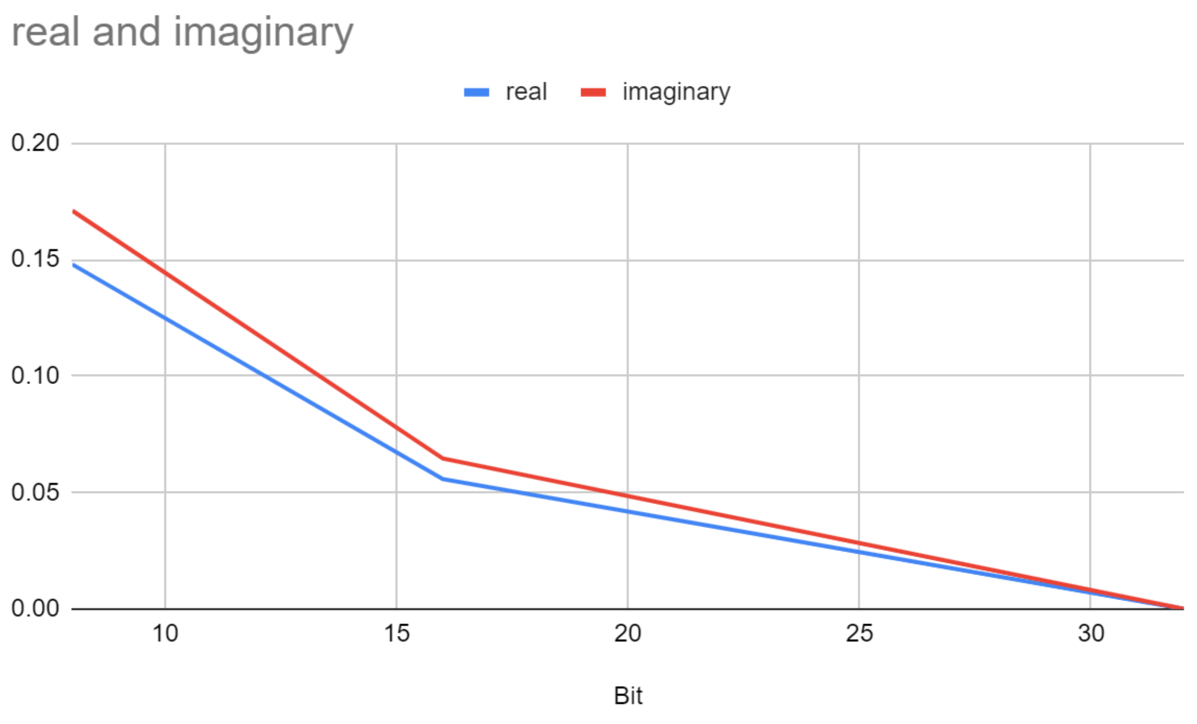
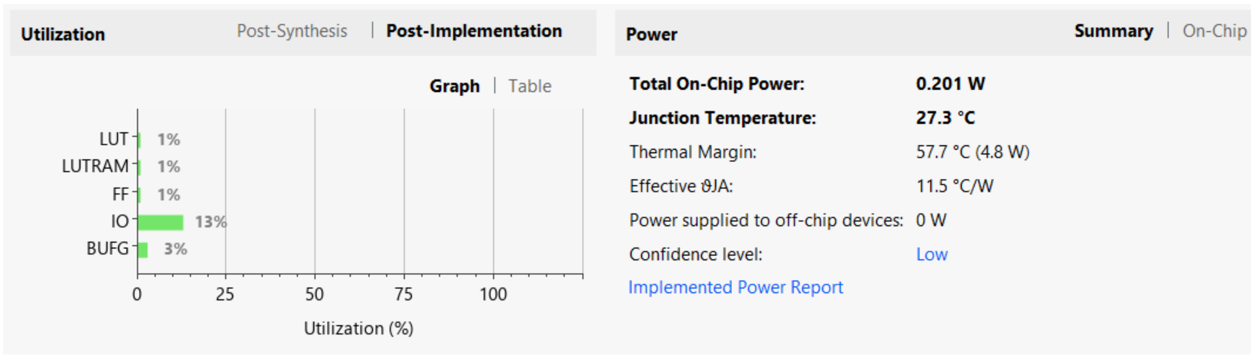


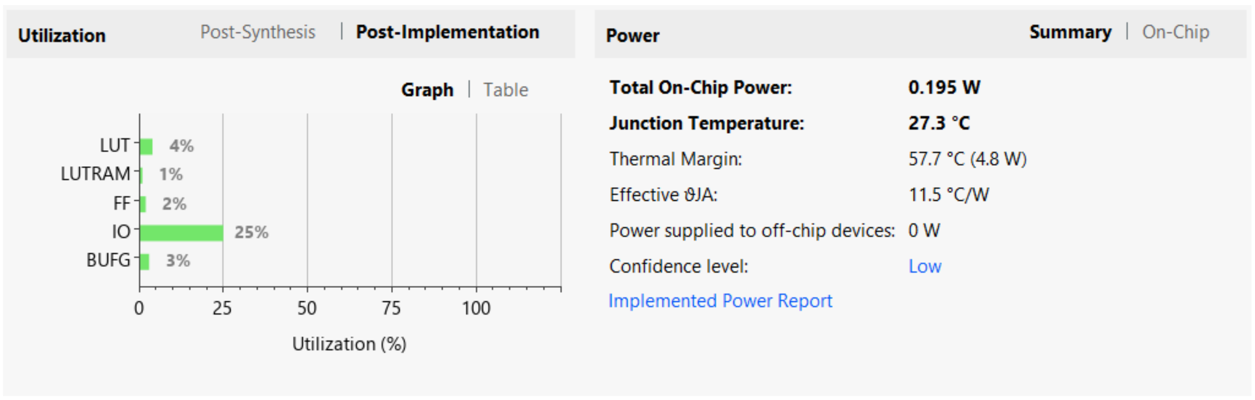
Figure 9: mean squared error in direct VHDL design with 8, 16, and 32 bits.

Utilization Report for direct VHDL design:

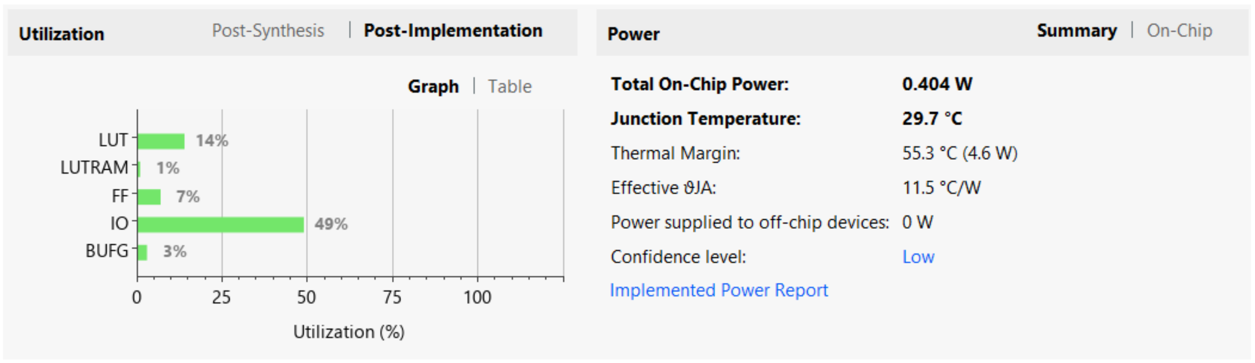
The utilization report for 8, 16, and 32 bits are shown below:



Total On-Chip Power: **0.201 W**
Junction Temperature: **27.3 °C**
Thermal Margin: 57.7 °C (4.8 W)
Effective ϑJA: 11.5 °C/W
Power supplied to off-chip devices: 0 W
Confidence level: Low
Implemented Power Report



Total On-Chip Power: **0.195 W**
Junction Temperature: **27.3 °C**
Thermal Margin: 57.7 °C (4.8 W)
Effective ϑJA: 11.5 °C/W
Power supplied to off-chip devices: 0 W
Confidence level: Low
Implemented Power Report



Total On-Chip Power: **0.404 W**
Junction Temperature: **29.7 °C**
Thermal Margin: 55.3 °C (4.6 W)
Effective ϑJA: 11.5 °C/W
Power supplied to off-chip devices: 0 W
Confidence level: Low
Implemented Power Report