Design and FPGA Implementation for Determining the Maximum Eigen value of a Complex Square Matrix

***Abstract*—Abstract: This experiment focuses on designing and implementing a digital VLSI architecture for determining the maximum eigenvalue of a complex matrix $\mathbb{R}_{n\times n}$. The process involves understanding and translating pseudocode into MATLAB code, designing an isomorphic VLSI architecture for n = 4, writing error-free Verilog HDL code, testing functionality, and determining the maximum operating clock frequency using the Vivado tool set. The hardware utilization of the design is also evaluated. Finally, the Verilog HDL code is dumped on the Zybo FPGA board, and the output (maximum eigenvalue) is displayed on the computer screen via Integrated Logic Analyzer (ILA). This experiment provides hands-on experience in VLSI design, Verilog programming, FPGA implementation, and performance analysis.**

***Index Terms*—Eigenvalue,verilog, architecture,MATLAB**

I. Introduction

Introduction: Digital Very Large Scale Integration (VLSI) plays a crucial role in modern electronic systems, enabling the implementation of complex algorithms and computations in hardware. In this experiment, the focus is on designing and implementing a digital VLSI architecture for determining the maximum eigenvalue of a complex matrix. Eigenvalue computation is a fundamental operation in various fields such as signal processing, control systems, and machine learning. By translating pseudocode into MATLAB code and subsequently into Verilog HDL for FPGA implementation, this experiment aims to provide a hands-on experience in VLSI design, hardware description language programming, and FPGA utilization. The experiment involves understanding the theoretical concepts behind eigenvalue computation, designing an efficient VLSI architecture, testing functionality, and analyzing hardware performance metrics. Through this experiment, we will gain practical insights into the intersection of algorithm design, hardware implementation, and performance optimization in the realm of digital VLSI systems.

II. Methodology:

The methodology adopted for solving the experiment involves a systematic approach to designing and implementing a digital VLSI architecture for determining the maximum eigenvalue of a complex matrix using the power method. The power method, an iterative algorithm for estimating dominant eigenvalues and eigenvectors, forms the basis of the computational process.

A. Understanding the Pseudocode and High-Level Code Development:

The initial step involves comprehending the provided pseudocode, which outlines the process of calculating the maximum eigenvalue using the power method. Subsequently, a high-level code in Python is developed to implement the power method and validate its functionality. Various test vectors are employed to verify the accuracy of the Python implementation against expected outcomes.

B. The pseudocode

The pseudocode used to find maximum eigenvalue is as follow:

```
1. Input: All elements of $\mathbb{R}_{n\times n}$ matrix, where n = 4 for this experiment.
2. Initialize a vector: $\mathbb{C}^{t=0}_{n\times 1}$ =$[\mathbf{1}]_{n\times 1}$.
3. Initialize: t = 0.
4.Repeat:
(a) t = t+1.
(b) $\mathbb{Z}^t = R \times C^{t-1}$.
(c) $\mathbb{M}_t$ = max(R($\mathbb{Z}^t$)) where R(.) indicates the real part of each element.
(d) $\mathbb{C}^t = \mathbb{Z}^t/\mathbb{M}_t$ .
(e) Until N iterations.
5. Return: $\mathbb{M}_t$.
6. Output:$\mathbb{E}_{max} = \mathbb{M}_t$.
```

C. Verilog HDL Code Programming:

Transitioning from the high-level Python code, the next phase entails programming the functionality in Verilog HDL. The methodology includes creating distinct modules such as a division block, a comparator module for determining the maximum value, and a module for calculating absolute values. These modules are then instantiated within the top-level module to form the complete VLSI architecture. Simulations are conducted to observe the behavior and outcomes of the Verilog HDL implementation, ensuring alignment with expected results.

By following this structured methodology encompassing code development, simulation, and verification at both high-level and hardware description language levels, the experiment aims to demonstrate the successful implementation of the power method for calculating the maximum eigenvalue in a digital VLSI architecture.

D. Block diagram of Hardware architecture

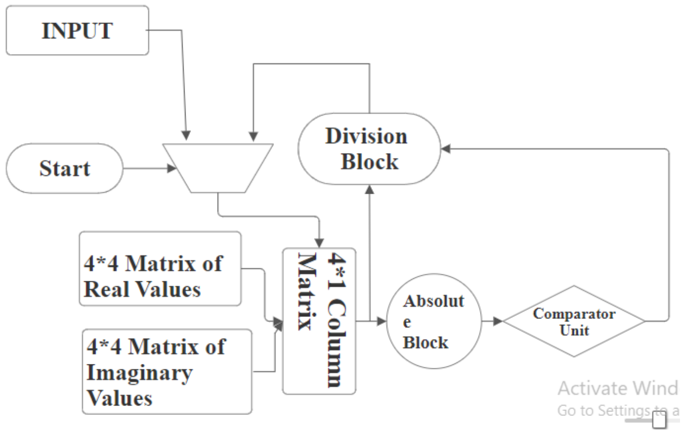
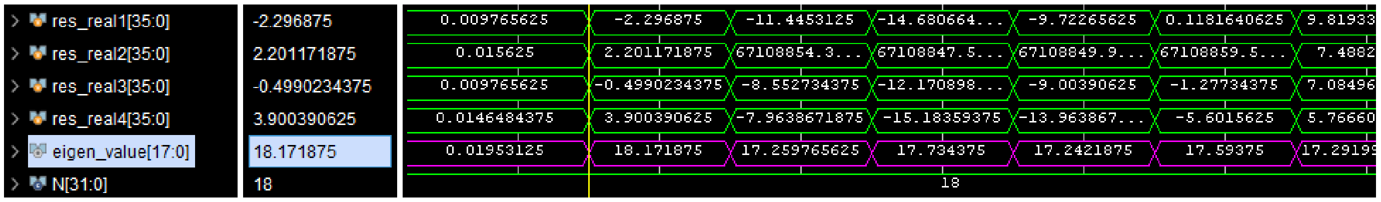


Fig. 1. Block diagram of Hardware

III. Simulation, Resource Utilization, Power consumption, ILA Results

A. Simulation ,Results

The simulation results, depicted in Figure 2.



```
res_real1[35:0]     -2.296875      0.009765625  -2.296875      -11.4453125    -14.680664...  -9.72265625   0.1181640625  9.81933
res_real2[35:0]     2.201171875    0.015625     2.201171875    67108854.3...  67108847.5...  67108849.9... 67108859.5... 7.4882
res_real3[35:0]     -0.4990234375  0.009765625  -0.4990234375  -8.552734375   -12.170898...  -9.00390625   -1.27734375   7.08496
res_real4[35:0]     3.900390625    0.0146484375 3.900390625    -7.9638671875  -15.18359375   -13.963867...  -5.6015625    5.76660
eigen_value[17:0]   18.171875      0.01953125   18.171875      17.259765625   17.734375      17.2421875     17.59375      17.29199
N[31:0]             18                                                        18
```

Fig. 2. Simulation results of Eigen values

B. Resource Utilization

The FPGA Board I choose for this experiment is Nexys 4 DDR and the presentation of resource utilization for the design to find the Eigen value is delineated in Table 1 and resource utilization for the design of customizable Integrated Logic Analyzer (ILA) IP core is shown in Table 2.

TABLE I
Resource Utilization for Eigen design block

Site Type	Used	Fixed	available	Utils%
Slice LUTs	6541	0	63400	10.32
LUT as Logic	6541	0	63400	10.32
Slice Registers	80	0	126800	0.06
Register as Flip-Flop	80	0	126800	0.06

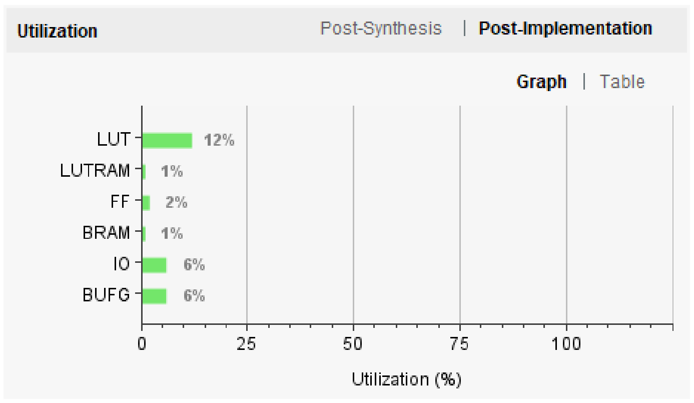


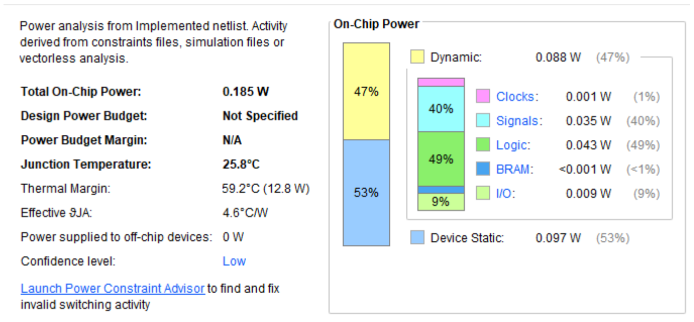
Fig. 3. Resource Utilization

TABLE II
Resource Utilization for ILA

Site Type	Used	Fixed	available	Utils%
Slice LUTs	679	0	63400	1.07
LUT as Logic	599	0	63400	0.94
LUT as Memory	80	0	19000	0.42
LUT as Shift Register	80			
Slice Registers	1158	0	126800	0.91
Register as Flip-Flop	1158	0	126800	0.91
F7 Muxes	4	0	31700	0.01

C. Power Consumption

The On-chip power consumption reports after implementation is shown below:



Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	**0.185 W**
Design Power Budget:	**Not Specified**
Power Budget Margin:	**N/A**
Junction Temperature:	**25.8°C**
Thermal Margin:	59.2°C (12.8 W)
Effective ϑJA:	4.6°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

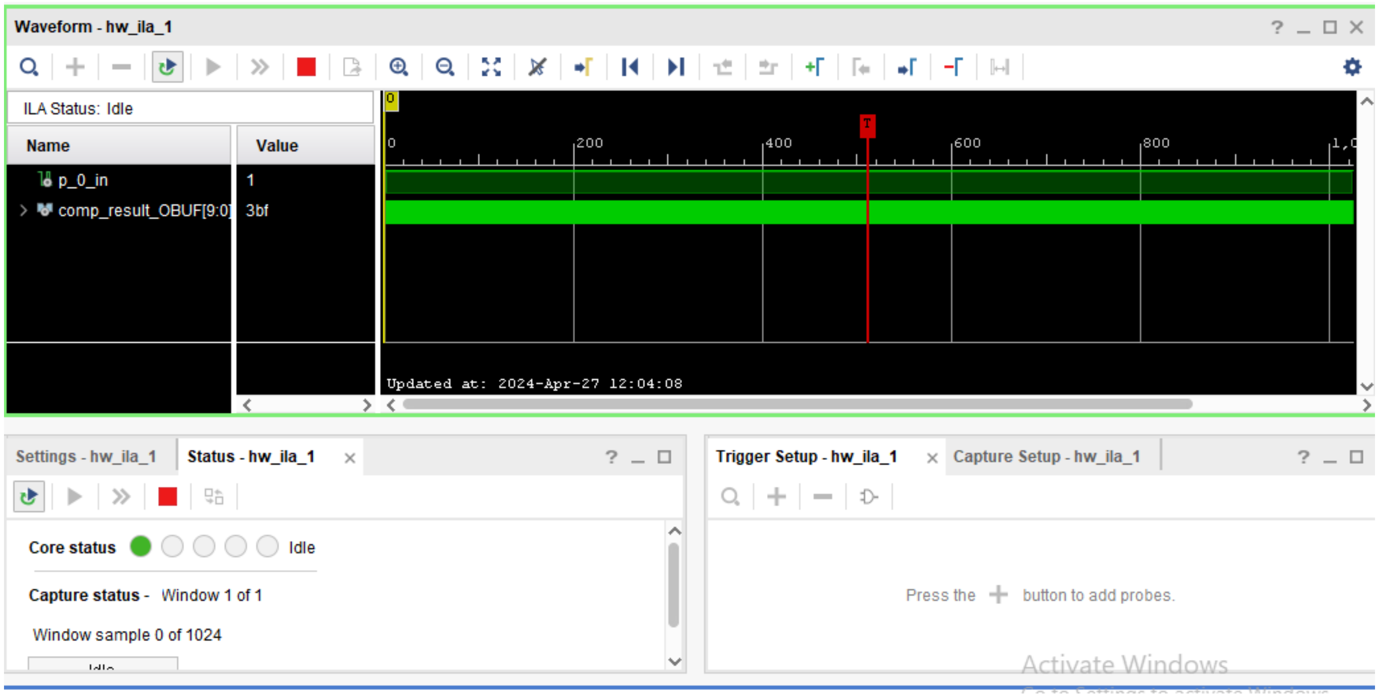
Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

Dynamic:	0.088 W	(47%)
Clocks:	0.001 W	(1%)
Signals:	0.035 W	(40%)
Logic:	0.043 W	(49%)
BRAM:	<0.001 W	(<1%)
I/O:	0.009 W	(9%)
Device Static:	0.097 W	(53%)

Fig. 4. Power consumption reports

D. ILA Outputs



Waveform - hw_ila_1

ILA Status: Idle

Name	Value
p_0_in	1
comp_result_OBUF[9:0]	3bf

```
Updated at: 2024-Apr-27 12:04:08
```

Settings - hw_ila_1 | Status - hw_ila_1

Core status Idle

Capture status - Window 1 of 1

Window sample 0 of 1024

Trigger Setup - hw_ila_1 | Capture Setup - hw_ila_1

Press the + button to add probes.

Fig. 5. Output results of ILA

IV. Conclusion

Conclusion: The experiment on designing a digital VLSI architecture for calculating the maximum eigenvalue of a complex matrix using the power method provided a hands-on learning experience in algorithm translation and hardware implementation. By converting pseudocode into Python and Verilog HDL, I gained insights into VLSI design and verification. This experiment underscored the significance of algorithm-hardware co-design for efficient digital VLSI systems, enhancing skills in Verilog programming and performance analysis.

References

[1] *S. Palnitkar, Verilog HDL: a guide to digital design and synthesis.Prentice Hall Professional, 2003, vol. 1.*