ESE5730 Chip Design

Computing: Digital 8-bit 6x6 Matrix Multiplier Engine

Team 8

Shun Yao

Yue Zhang

TA: Claire Kendell

DEPARTMENT OF ELECTRICAL AND SYSTEM ENGINEERING

UNIVERSITY OF PENNSYLVANIA

I. Design Flow

SIPO (**Serial In Parallel Out*****) Shift Register:*** SIPO should receive data from MCU through SPI, which requires a CS connection. So, SIPO consists of three inputs (D (Data Line), RESET (~CS for SPI), CLK) and 48 outputs (q47-q0) (refer to 6 8bit-number). To fill two input memory array, 2 SIPO are needed.

Input Memory Array: Each data word consists of 8 bits, resulting in a memory array capable of storing 36 words, or 288 bits in total. So, 48*6 SRAMs are arranged and integrated with peripheral circuitry, including sense amplifiers, write drivers, and pre-charge circuits. The memory array also incorporates row decoders that facilitate addressing and enable read and write operations.

Matrix Multiplier: Matrix Multiplier needs to do multiplication for the column and row for matrix A and matrix B, and add 6s results together to get one value stored into output memory array. Considering the overflow, finial output can reach 19bit, thus, this block consists of 6 8bit_multiplier, 3 16bit_RCA, 1 17bit_RCA, and 1 18bit_RCA.

Output Memory Array: Output Memory Array is similar to input memory Array but is 19bit, because the maximum output values from matrix multiplier can be 19bit.

PISO (Parallel In Serial Out) Shift Register: PISO should receive data from output memory array under the FSM control and sent the serial output to MCU or logic analyzer. PISO has 19bit data input, CLK input, control input and 1 serial output.

FSM (Finite State Machine): FSM consists of four different states: IDLE, LOAD, COMPUT, and OUTPUT. Each state gives 24 different output control signals that represent each state, as shown in the truth table below, where 0 stands for GND, and 1 stands for control signal generated from the state controller. The IDLE state can be hardcoded; other states are controlled by blocks: load_controller, compute_controller, and output_controller. After the OUTPUT state, the FSM will return to IDLE and wait for the next round's computation.

Table 1. Finite State Machine Truth Table

	WE [1,2,3]	SE [1,2,3]	PE [1,2,3]	EN [1,2,3]	Address [1,2,3]
IDLE (Next State Condition: CSb = 1 to 0)	000	111	000	000	000 000 000000
LOAD (Next State Condition: compute_start=0 to 1)	100	111	100	100	000 to 101 000 to 101 000000
COMPUTE (Next State Condition: output_start=0 to 1)	001	001	111	111	000 to 101 000 to 101 000000 to 100011
OUTPUT (Next State Condition: output_done =0 to 1)	000	110	001	001	000 000 000000 to 100011

Top – Cell: Two SIPO are connected to Input Memory Array, respectively. Matrix Multiplier load data from two Input Memory Array and output is stored in Output Memory Array. Last stage PISO load data from Output Memory Array to MCU/Logic Analyzer. All under the FSM's control.

II. System Block Diagram and Schematic

Top-Cell:

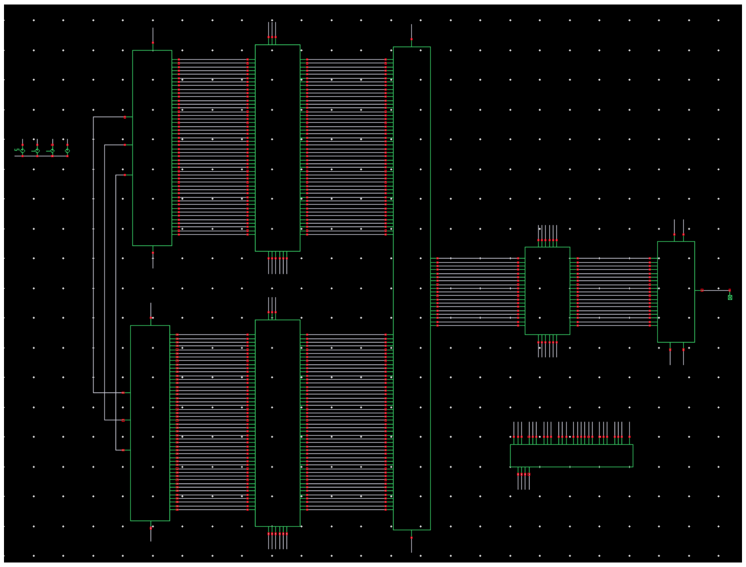


Figure 1. Top – Cell Schematic

SIPO:

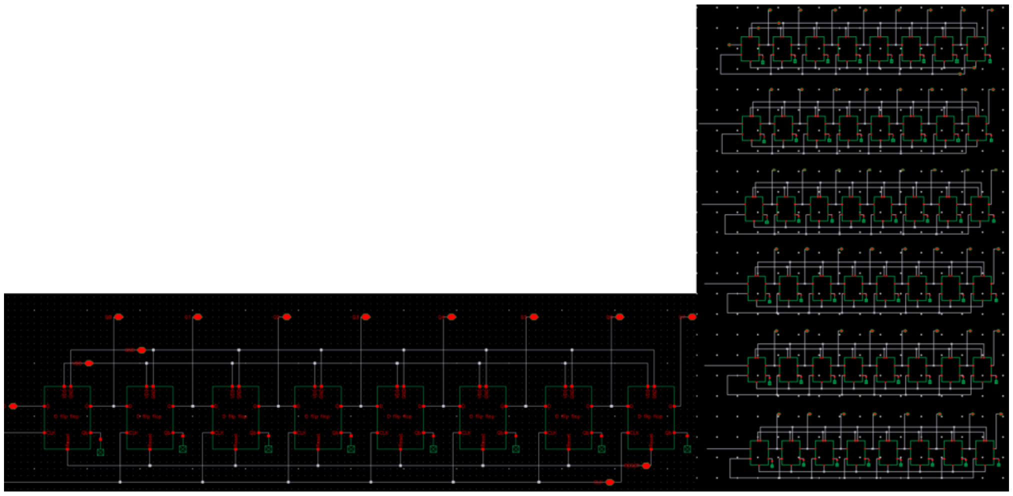


Figure 2. SIPO Shift Register (left: Detailed Schematic, right: Full Schematic)

Input Memory Array:

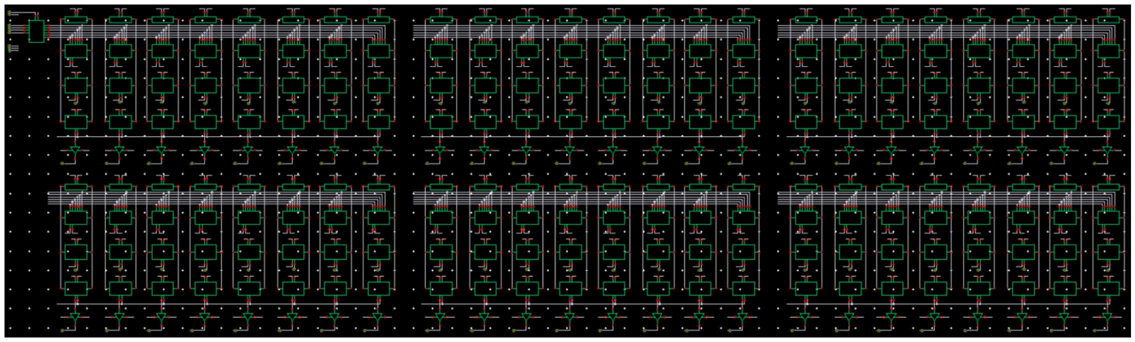


Figure 3. Input 8 bit 6x6 Memory Array

Matrix Multiplier:

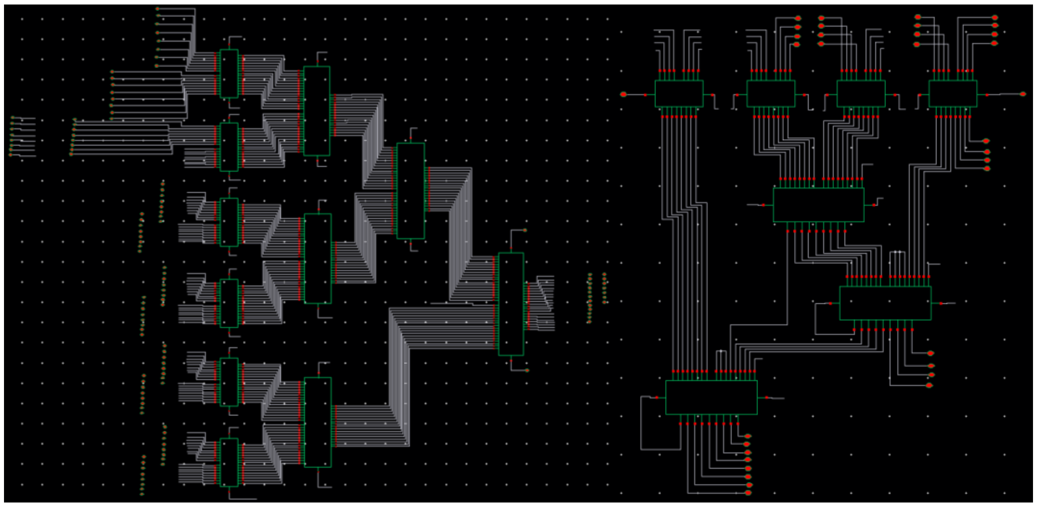


Figure 4. Matrix Multiplier (left) and 8Bit Multiplier (Right)

4Bit Multiplier:

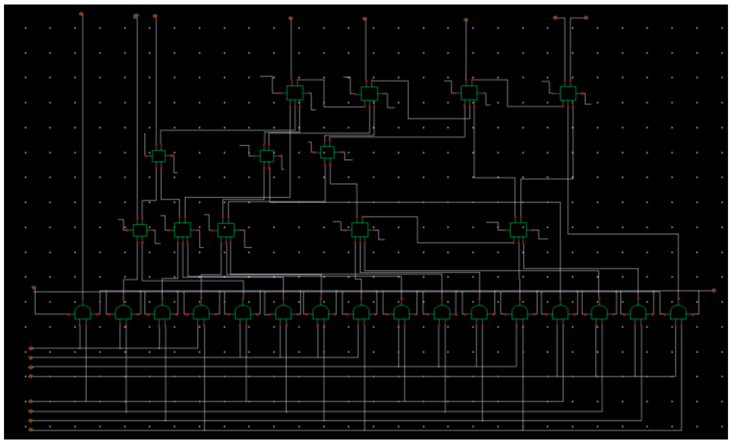


Figure 5. 4Bit Multiplier

Output Memory Array:

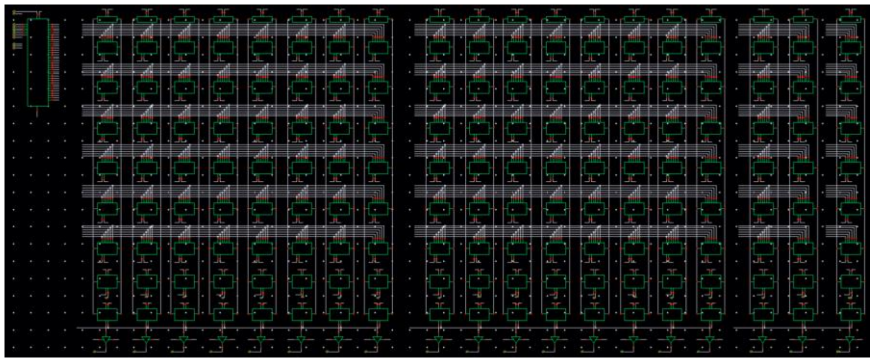


Figure 6. Output 19bit 6x6 Memory Array

PISO:

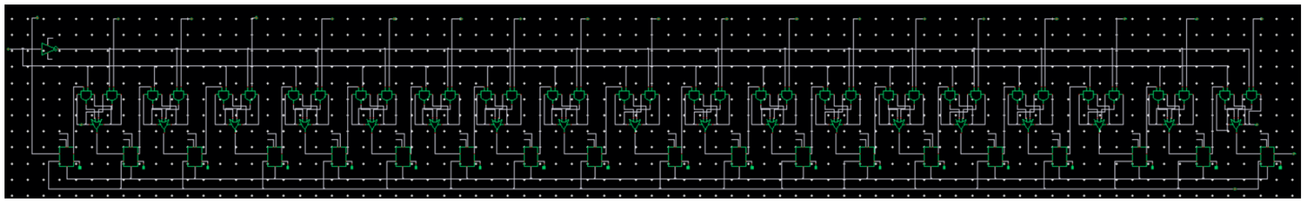


Figure 7. PISO Shift Register

State Machine:

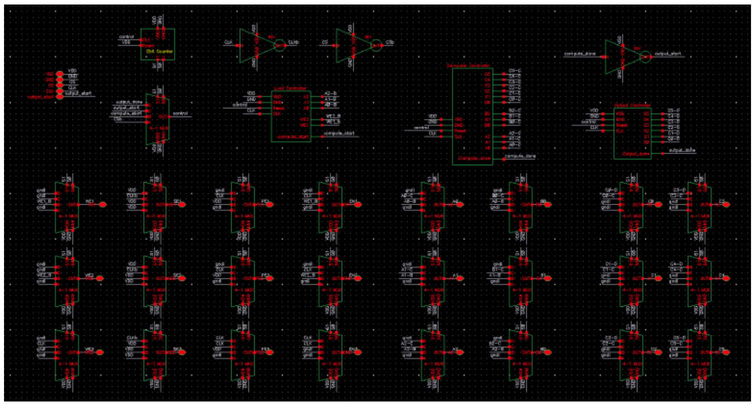


Figure 8. FSM Schematic

Load Controller:

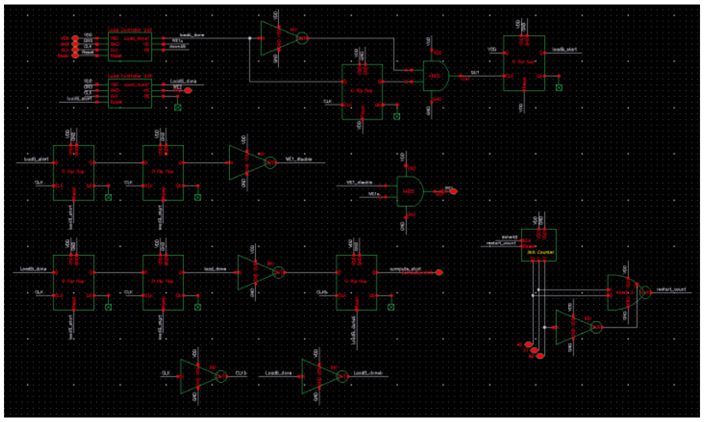


Figure 9. Load Controller Schematic

Load controller unit:

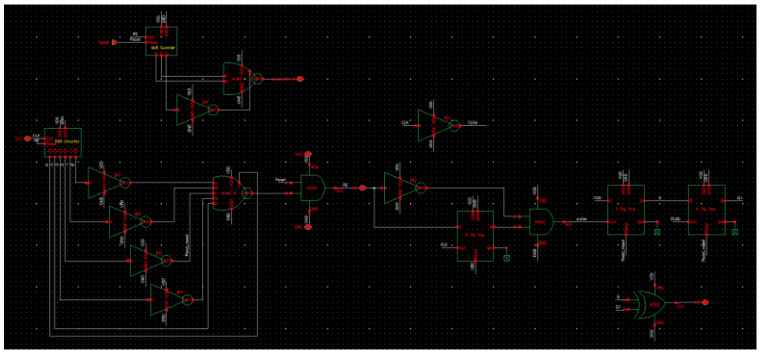


Figure 10. Load Controller Unit Schematic

Compute Controller:

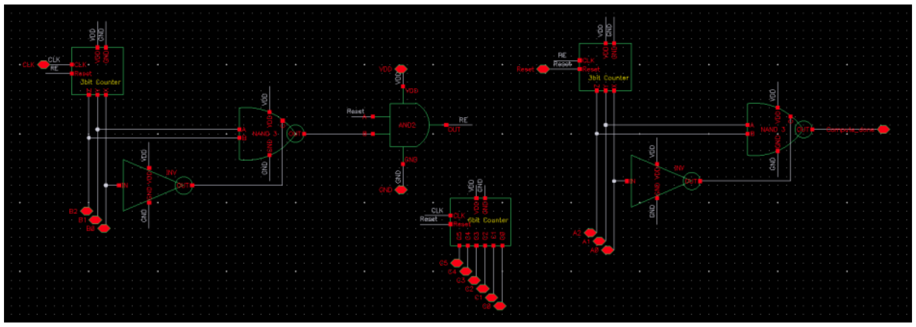


Figure 11. Compute Controller Schematic

Output Controller:

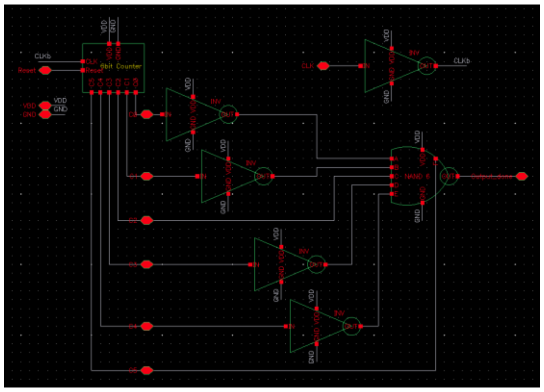


Figure 12. Output Controller Schematic

III. Cadence Simulation Setups and Results

Top Cell: Transient simulations have run with a stop time of 260ns. Take one calculation circle as an example, six 1111 1111 times six 1111 1111 correspondingly, and adding the six multiplication results together, the final result is supposed to be 19bit 101 1111 0100 0000 0110.
At time 0-10ns, the system is in IDLE waiting for the start signal ‘CS’; at time 10-20ns, the system is in LOAD loading data to MatrixA and MatrixB; at time 20-30ns, the system is in COMPUTE state, the system will read data from MatrixA and B, compute it, and write it to MatrixC in this state; at time 30-40ns, system is in OUTPUT state, it will read data from MatrixC and load it to PISO register and the 40-230ns, is for the complete result output in serial as shown in figure below:

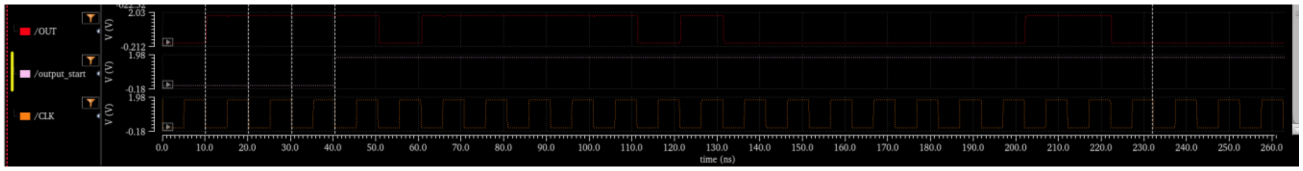


Figure 13. Top – Cell Simulation Result

SIPO: Transient simulation is run with 510ns stop time. Input data (D) (10101100) (iterate) with 10ns period and 30ns delay time; RESET is set to have 30ns low and long period high to simulate the SPI is enabled; CLK is 10ns period. Expected output is 1010110010101100…10101100 (q47-q0). Figure below shows the simulation results (q47 – q40).

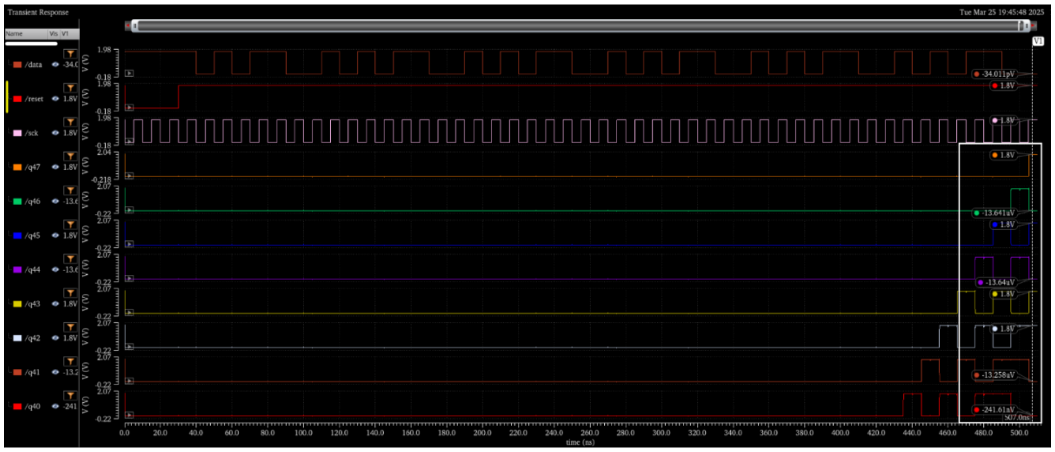


Figure 14. SIPO Simulation Result

Input Memory Array: take 8 data points for demonstration (address:111): dataIN left(0-1us), dataOUT right(2-3us)

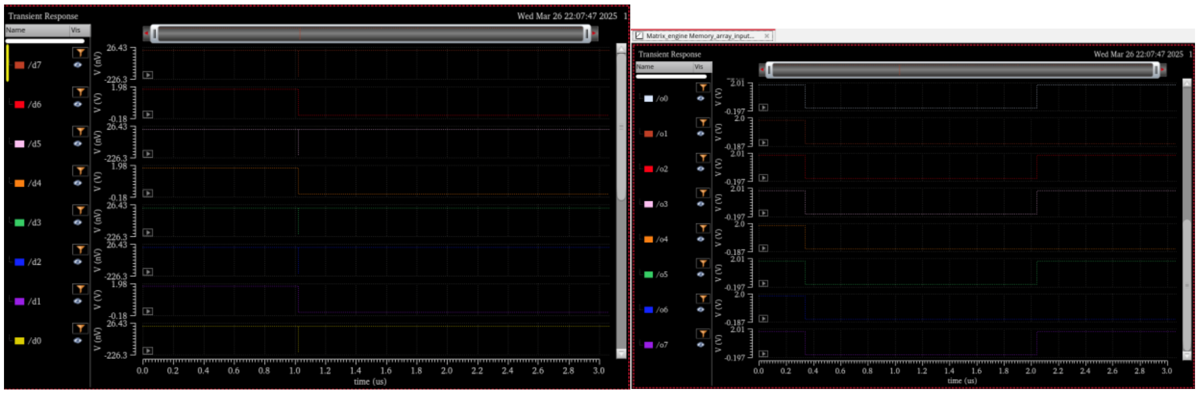


Figure 15. Input Memory Array Simulation Result

Matrix Multiplier: Transient simulation is run with 51ns stop time. Simulate the matrix multiplication of output one value. Set the column (6 8bit) and row (6 8bit) to 1, the expected result is 19bit 101 1111 0100 0000 0110. The results below meet the expectations.

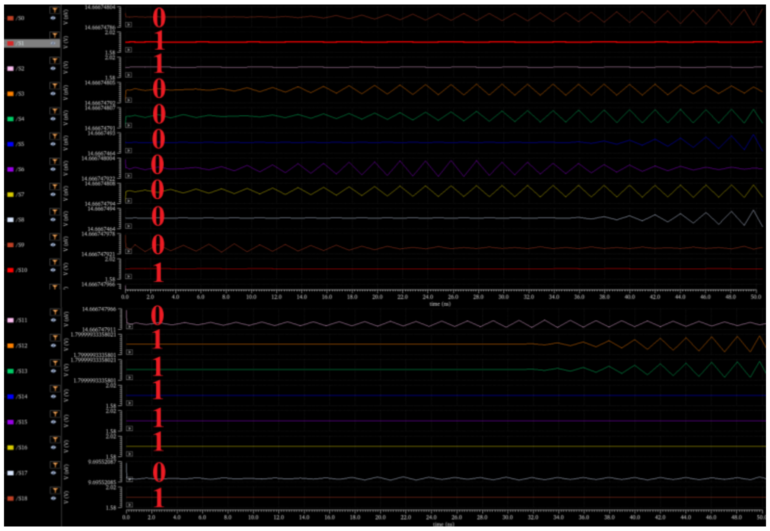


Figure 16. Matrix Multiplier Simulation Result

Output Memory Array: take 8 data points for demonstration (address:111111): dataIN left(0-1us), dataOUT right(2-3us)

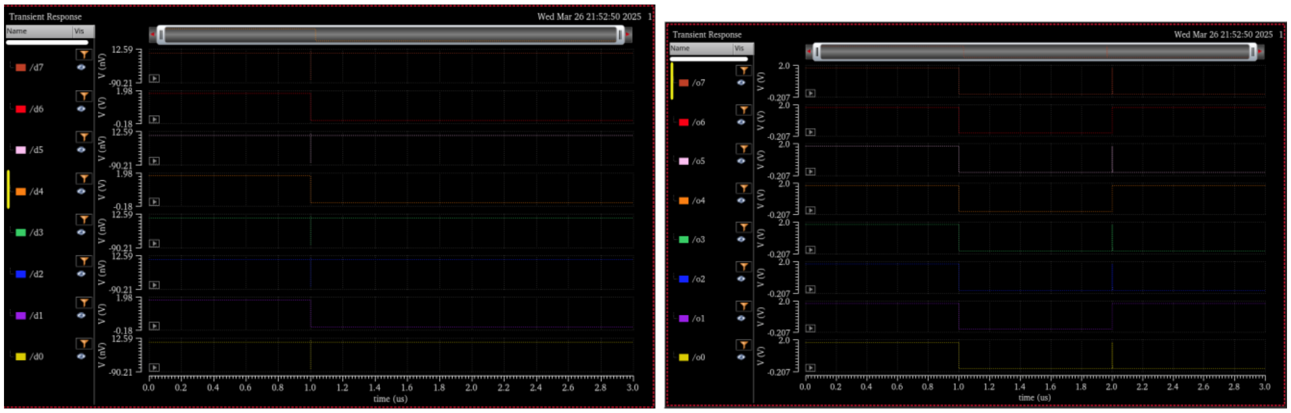


Figure 17. Output Memory Array Simulation Result

PISO: Transient simulation is run with 195ns stop time. Control input is set with 5ns low and long period high, CLK is 10ns period and parallel input from Q18 to Q0 is 0111111111000000000. Serial output is piso in the following figure.

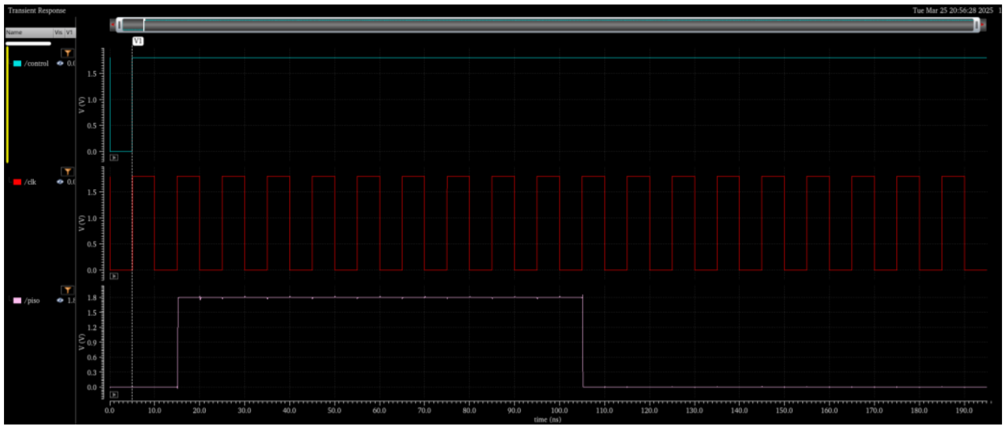
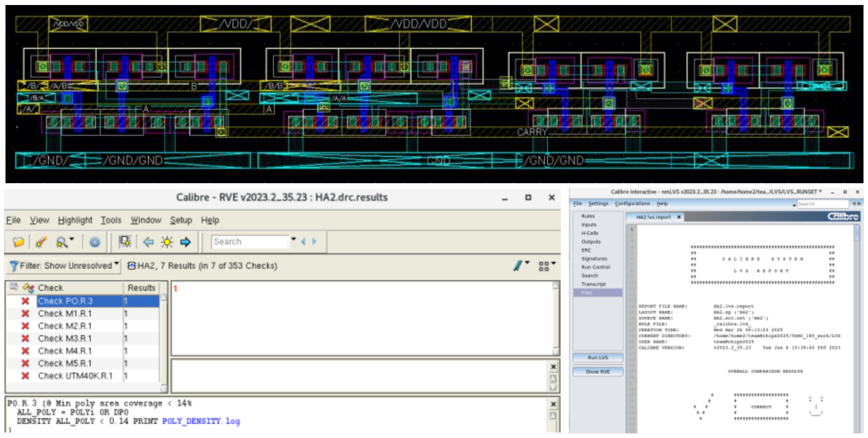


Figure 18. PISO Simulation Result

Layout:


/VDD/VDD
/GND/GND
CARRY
Calibre - RVE v2023.2_35.23 : HA2.drc.results
File View Highlight Tools Window Setup Help
Filter: Show Unresolved
HA2, 7 Results (in 7 of 353 Checks)

Check	Results
Check PO.R.3	1
Check M1.R.1	1
Check M2.R.1	1
Check M3.R.1	1
Check M4.R.1	1
Check M5.R.1	1
Check UTM40K.R.1	1

```
PO.R.3 (@ Min poly area coverage < 14%
  ALL_POLY = POLYi OR DPO
  DENSITY ALL_POLY < 0.14 PRINT POLY_DENSITY.log
}
```

Figure 19. Half Adder DRC/LVS Clean Layout

IV. Table of Performance

Table 2. Performance and Comparison

Block	VCC	Proposed Speed	Design Speed	Proposed Power	Design Power
Matrix Multiplier	1.8	720ns (for 36 times)	7ns*36	18mW	93.632n*36*1.8
SIPO		4ns (delay)	<1ns	0.1mW	3.2n*1.8
PISO		4ns (delay)	<1ns	0.1mW	2.8n*1.8
Input Memory		15ns	4ns	2mW	23n*1.8
Output Memory		15ns	6ns	2mW	26.3n*1.8

V. Project Timeline and Task Assignment

Table 3. Project Timeline with Current Status, Task Explaination and Task Assignment

Task	Start Date	End Date	Task Description
Design Confirmation	2.26	3.5	Determine overall architecture, design specifications, and performance goals before starting (Both);

Block Schematic Design (Pre-layout Simulation)	3.6	3.26	Shun: Input/Output Memory Array (with decoder), FSM;
			Yue: SIPO, Matrix Multiplier, PISO;

Top-Cell Schematic Design (Pre-layout Simulation)	3.25	3.26	Schematic design and pre layout simulation to check the correctness(Both);
Block Layout (DRC, LVS, RC extraction, Post-Layout Simulation)	3.10	4.17	Shun: Already finished some block's layout, continuing routing the corresponding's layout; Yue: Already finished some block's layout, continuing routing the corresponding's layout;
Top-Cell Layout (DRC, LVS, RC extraction, Post-Layout Simulation)	4.18	4.23	Shun: do Top Cell layout; Yue: do Top Cell's post layout simulation;
Check and Ready for Tape-Out	4.24	4.30	Check each block (schematic, layout, performace), DRC/LVS clean, performance and ready for tape-out (Both).

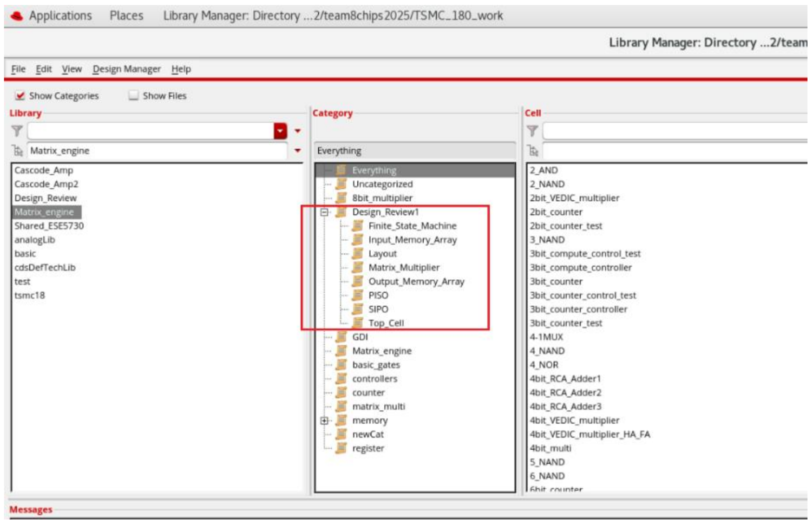
Reflection:

During the proposal, I did not realize that we needed to complete the schematic design of all blocks and perform the pre-layout simulation for the top cell by March 26th design review.

In my initial project proposal, I outlined a design flow that involved completing each block sequentially. This approach meant finishing the schematic, pre-layout simulation, layout, and post-layout simulation of one block before moving on to the next. However, this strategy proved problematic because it left insufficient time to complete all block schematic designs before the design review on March 26th.

Moreover, the original timeline had an inherent flaw. Completing one block at a time before progressing to the next means that the top cell's pre-layout simulation would not take place until the final stages of the project. If issues were to arise at that point, we would have limited time to revise sub-block designs.

VI. Schematic and Layout File Path



Library: Matrix_engine

Cascode_Amp
Cascode_Amp2
Design_Review
Matrix_engine
Shared_ESE5730
analogLib
basic
cdsDefTechLib
test
tsmc18

Category: Everything

Everything
Uncategorized
8bit_multiplier
Design_Review1
- Finite_State_Machine
- Input_Memory_Array
- Layout
- Matrix_Multiplier
- Output_Memory_Array
- PISO
- SIPO
- Top_Cell

GDI
Matrix_engine
basic_gates
controllers
counter
matrix_multi
memory
newCat
register

Cell

2_AND
2_NAND
2bit_VEDIC_multiplier
2bit_counter
2bit_counter_test
3_NAND
3bit_compute_control_test
3bit_compute_controller
3bit_counter
3bit_counter_control_test
3bit_counter_controller
3bit_counter_test
4-1MUX
4_NAND
4_NOR
4bit_RCA_Adder1
4bit_RCA_Adder2
4bit_RCA_Adder3
4bit_VEDIC_multiplier
4bit_VEDIC_multiplier_HA_FA
4bit_multi
5_NAND
6_NAND
6bit_counter

Figure 20. File Path

VII. Updated Measurement Plan

Serial Communication:
ESP32 (80MHz) (MCU) communicate with our chip through SPI with MOSI, SCLK and CS, corresponding to D, CLK and RESET in our chip. When CS is switched from high to low, the SPI communication channel is opened. To load the data from MCU to SIPO (in chip), an enable input from MCU to trigger the FSM change state from idle to load data. The calculation output from SIPO can be captured by logic analyzer for following verification.

Functional verification:

1. *Basic Arithmetic Correctness Test:* Send matrix data to the input of the chip through the MCU and compare the calculation results with theoretical calculation values to verify whether the matrix multiplication is correct using logic analyzer.

Performance Evaluation:

1. *Maximum Operating Frequency Test:* Gradually increase the clock frequency to test the stability of the chip at different frequencies and determine its maximum reliable operating frequency.

2. *Power Consumption Measurement:* Use a power measurement tool to measure operating power, compared to simulated power consumption.

Typical Application Scenarios

1. *Artificial Intelligence & Machine Learning:* Acceleration of small matrix operations in Convolutional Neural Networks (CNNs) to improve AI computational efficiency.

2. *Digital Signal Processing (DSP):* For image processing, edge detection, filtering, and other scenarios to increase processing speed.

3. *Embedded Computing & Low Power Applications:* Fast matrix computation in IoT devices to improve energy efficiency and reduce computing resources.

Changes:
Removed: Computation Latency Measurement.
Added: More clear serial communication explanation.