Matrix Multiplication 1-Design

Digital Design & Logic Synthesis

Project: Matrix Multiplication

Block: matmul

Digital High Level Design

Version 0.3

Revision Log

Rev	Change	Description	Reason for change	Done By	Date
0.1	Initial document		Initial document	Roee & Noam	07/01/2024
0.2	Upload Flow Chart		Upload Flow Chart	Roee & Noam	8/2/2024
0.3	Update after Verification		Update after Verification	Roee & Noam	5/03/2024

Table of Content

LIST OF FIGURES *4*

1. BLOCKS FUNCTIONAL DESCRIPTIONS *5*

1.1.1 Component Top #1 Matmul 5
1.1.2 Component #2 Matmul Calculator 7
1.1.3 Component #3 Operand Registers A and B 10
1.1.4 Component #4 Control Register 13
1.1.5 Component #5 Scratchpad 15
1.1.6 Component #6 Adress Decoder 19
1.1.7 Component #7 PE (Processing Element) 20
1.1.8 Component #8 APB Slave 23
1.1.9 Component #9 Register Flags 25

2. FLOW CHART *26*

3. APPENDIX *27*

3.1 Terminology **27**

LIST OF FIGURES

Figure 1: view of the block matmul. 6
Figure 2: view of the block matmul. 7
Figure 3: view of the block matmul_calculator. 9
Figure 4: view of the block matmul_calculator. 9
Figure 5: view of the block matmul_calculator. 9
Figure 6: view of the block Register_A. 11
Figure 7: view of the block Register_A 11
Figure 8: view of the block Regiser_A 11
Figure 9: view of the block Register_B 12
Figure 10: view of the block Register_B 12
Figure 11: view of the block Registe_B 13
Figure 12: view of the block Control Register 14
Figure 13: view of the block Control Register 15
Figure 14: view of the block Control Register 15
Figure 15: view of the block ScratchPad 16
Figure 16: view of the block ScratchPad 17
Figure 17: view of the block ScratchPad 18
Figure 18: view of the block Address Decoder 19
Figure 19: view of the block Address Decoder 20
Figure 20: view of the block Address Decoder 20
Figure 21: view of the block PE 22
Figure 22: view of the block PE 22
Figure 23: view of the block PE 23
Figure 24: view of the block APB 24
Figure 25: view of the block Register Flags 25

1. BLOCKS FUNCTIONAL DESCRIPTIONS

1.1.1 Component Top #1 Matmul

Functional Description:

The Matmul module serves as a hardware accelerator for matrix multiplication operations. It is designed to efficiently perform matrix multiplication using a systolic array architecture, which enables a highly parallel and pipelined computational structure, to achieve high throughput and reduced latency compared to traditional sequential methods.

Input Ports:

- clk_i: Clock input.
- rst_ni: Reset Negative input.
- psel_i: Peripheral select input.
- penable_i: Peripheral enable input.
- pwrite_i: Write enable input for peripheral.
- pstrb_i: Valid data input for peripheral.
- pwdata_i: Data write input for peripheral.
- paddr_i: Address input for peripheral.

Output Ports:

- pready_o: Peripheral ready output.
- pslverr_o: Peripheral slave error output.
- prdata_o: Data output for peripheral.
- busy_o: Busy signal output.
- done_o: Done signal output.

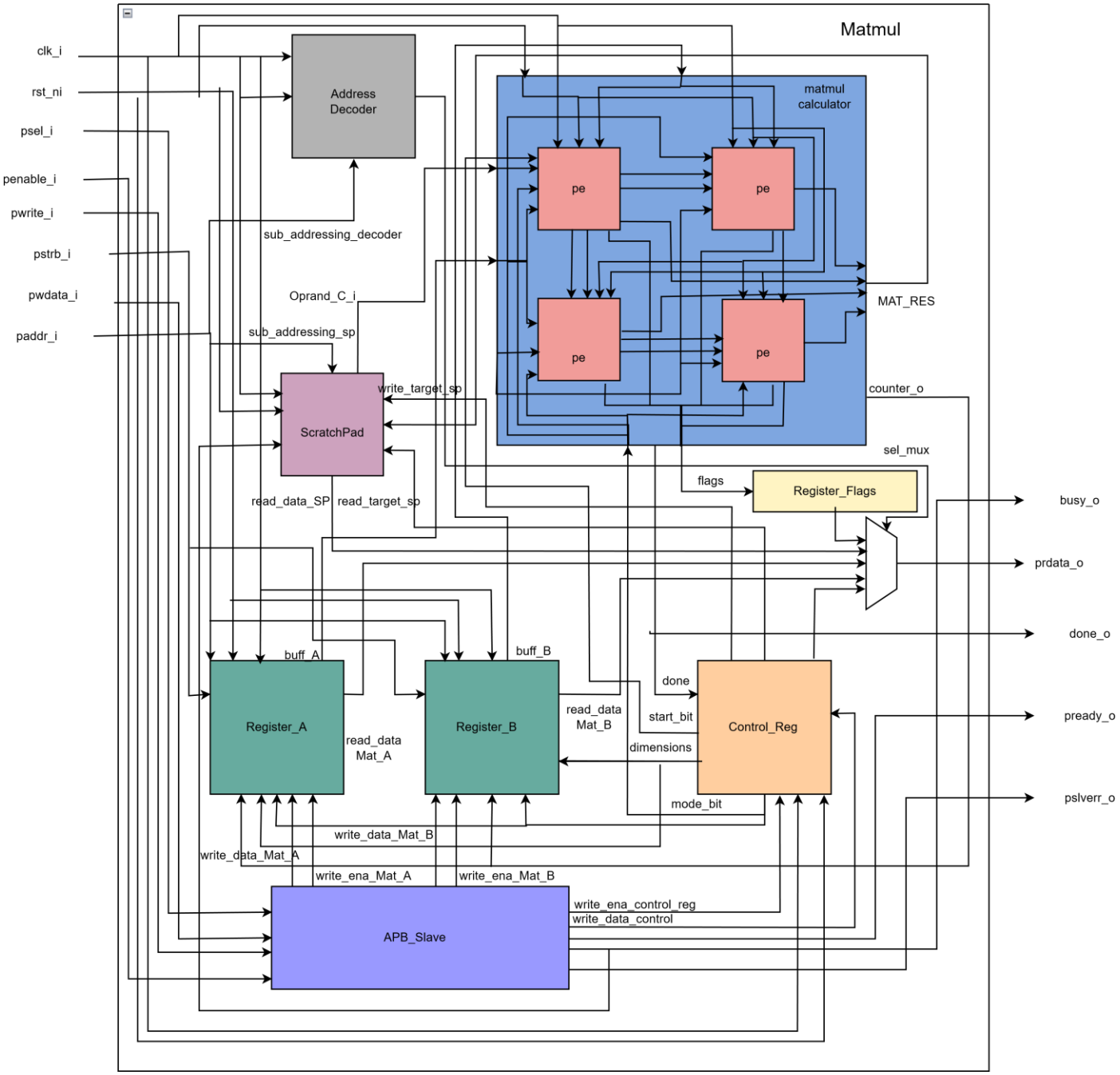


Figure 1: view of the block matmul.

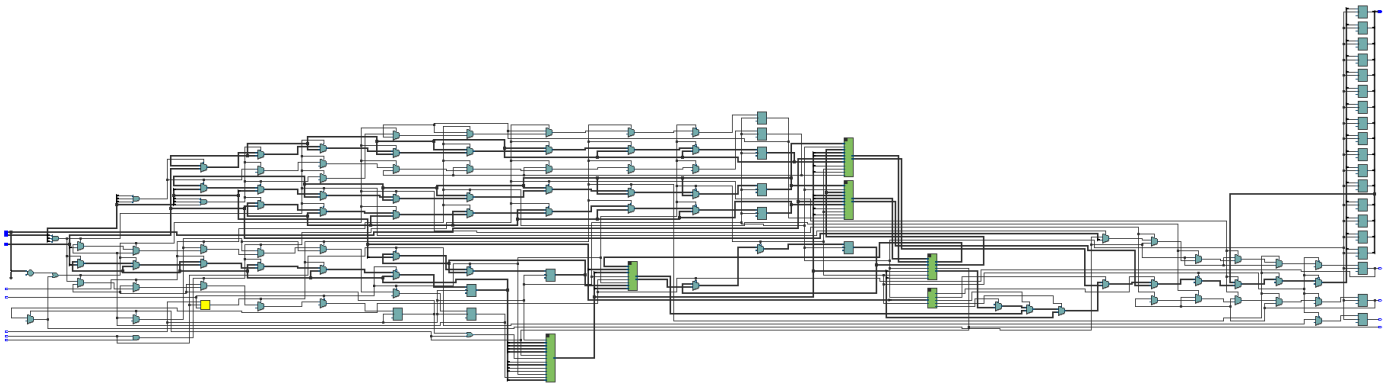


Figure 2: view of the block matmul.

1.1.2 Component #2 Matmul Calculator

The Matmul Calculator is designed to perform matrix multiplication of two input matrices A and B and produce the result matrix in res_o.

It utilizes a parallel processing approach where each element of the result matrix is computed independently using Processing Elements (PEs).

Inputs:

clk_i: Clock input.

rst_ni: Reset negative input.

A: Input matrix A from buffer

B: Input matrix B from buffer

start_bit: Start bit signal to initiate the computation.

mode_bit: Mode bit t is set to biased operation add or not add matrix C.

Outputs:

Met_Res: Result matrix.

flags: Flags indicating the status of overflow, underflow and carry.

done: Indicates when the computation is completed.

counter: Counter signal.

Internal Signals/Interfaces:

A_internal: Internal representation of elements in matrix A.

B_internal: Internal representation of elements matrix B.

start_bit_internal: Internal start bit signal for the next PE.

counter_i: Internal counter signal to count till the computation is done.

done_internal: Internal signal indicating completion of computations.

Model Hierarchy:

Inside the module, there's a matrix of Processing Elements (PEs) arranged in rows and columns to perform parallel computation.

PEs communicate with each other to exchange data and control signals.

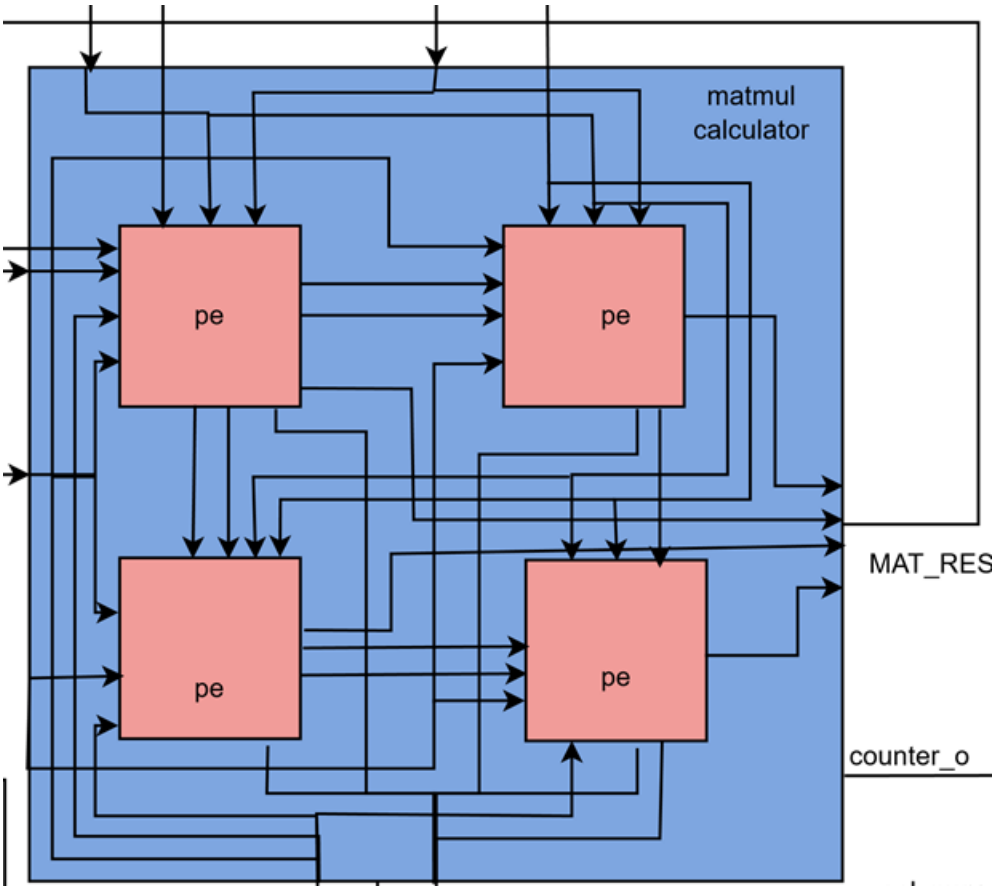


Figure 3: view of the block matmul_calculator.

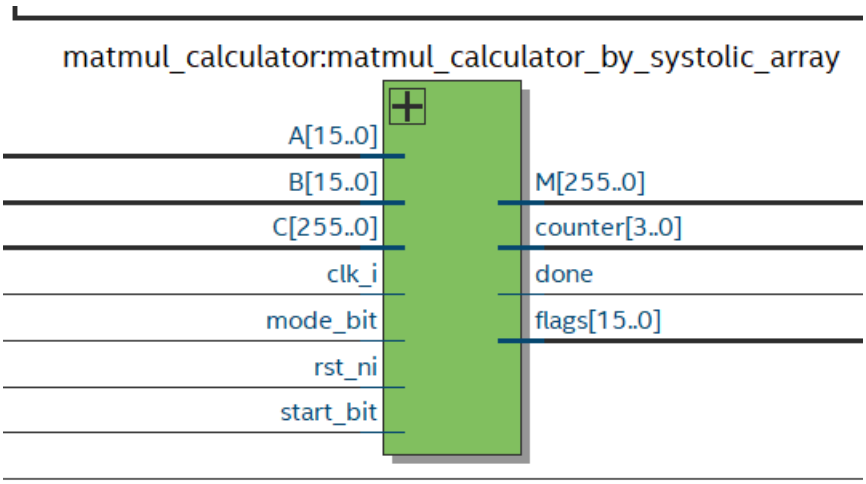


Figure 4: view of the block matmul_calculator.

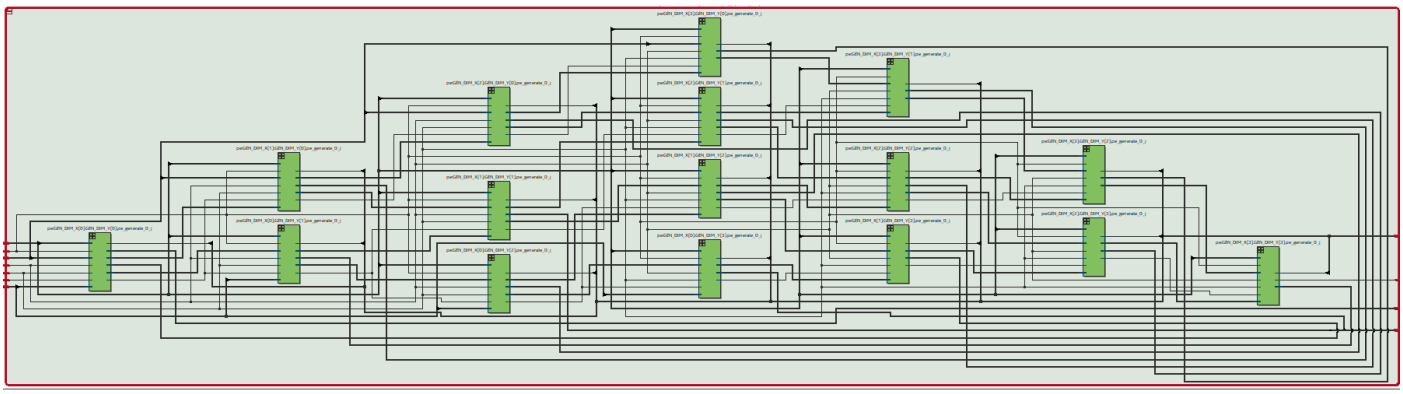


Figure 5: view of the block matmul_calculator.

1.1.3 Component #3 Operand Registers A and B

Functional Description of Operand Registers A and B Module:

These modules serve as registers for storing matrix operands A, B. Handle data loading and retrieval for matrix multiplication.

Input Signals:

- clk_i: Clock input signal.
- rst_ni: Reset signal (active low).
- start_bit: Start bit signal for initiating operations.
- reload_op: Signal for reloading operations.
- pwdata_i: Write data input for Matrix.
- counter: Counter for the buffer.
- Mat_i: Address input for row i Matrix (sub-addressing).
- write_en_Mat_i: Enable write signal for Matrix.
- pstrb_i: Valid data input for peripheral.
- n, k: Dimensions for operand A.
- k, m: Dimensions for operand B.
- Output Signals:
- read_data_Mat_o: Read data output for Matrix.
- buff: Read data output for multiplication operand.
- Internal Signals:
- zeros: Signal representing all zeros.
- matrix_Mat: 2D array representing the Matrix.

- write_data_Mat: Array storing write data for the Matrix.

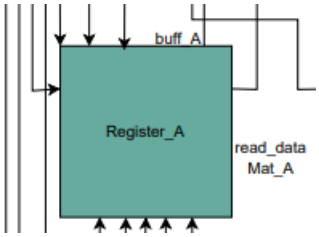


Figure 6: view of the block Register_A.

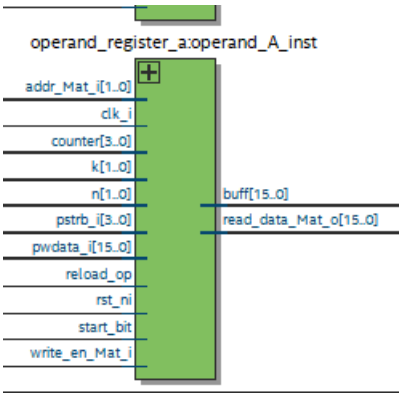


Figure 7: view of the block Register_A

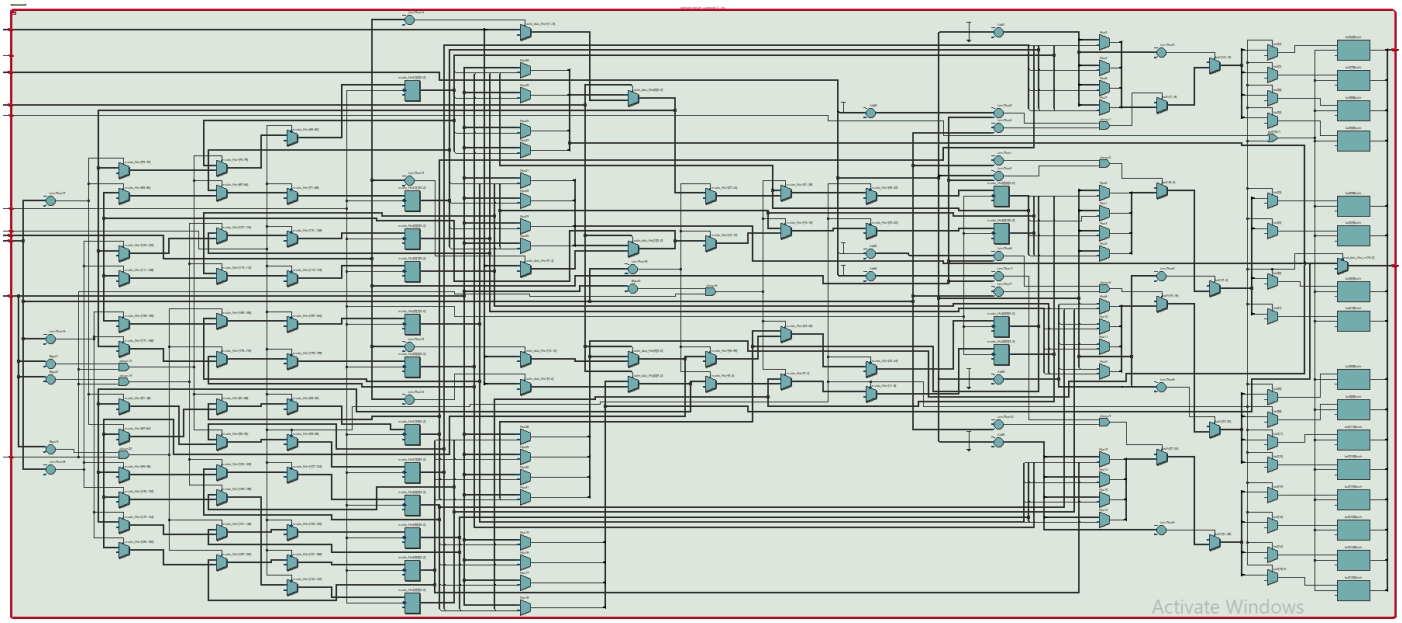


Figure 8: view of the block Regiser_A

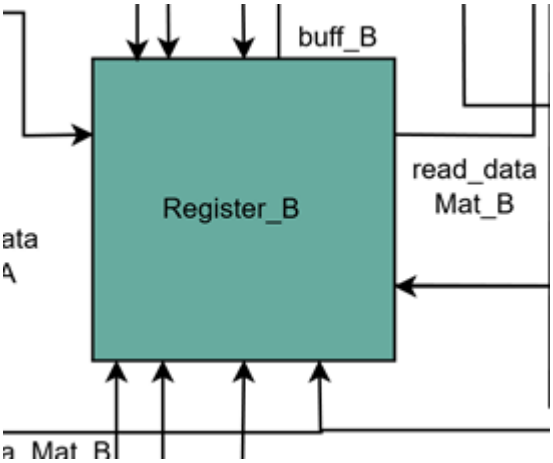


Figure 9: view of the block Register_B

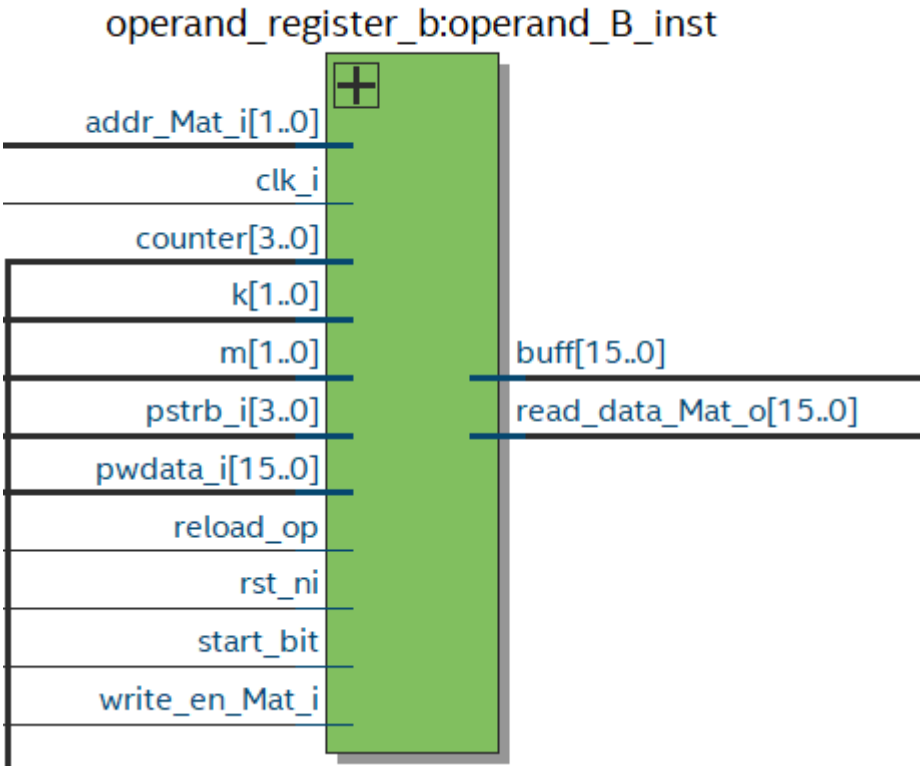


Figure 10: view of the block Register_B

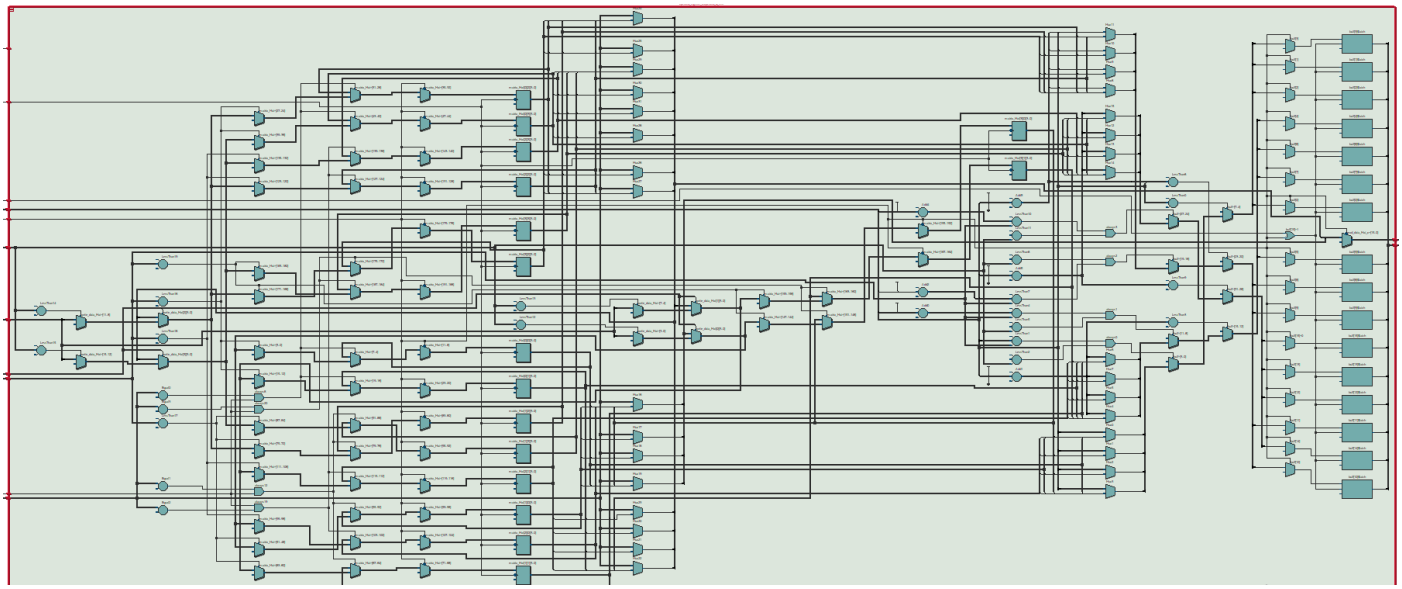


Figure 11: view of the block Registe_B

1.1.4 Component #4 Control Register

Functional Description of Control Register Module:

The Control Register module plays a vital role in managing control signals and operation parameters within the matrix multiplication accelerator design. By providing a configuration setting, it enables flexible and efficient operation of the accelerator, allowing users to customize its behavior according to their requirements.

Inputs:

- clk_i: Clock signal.
- rst_ni: Reset negative signal.
- done: Signal indicating completion of an operation.
- ena_write_control_reg: Enable signal for writing to the control register from APB.
- start_bit: Signal indicating the start of an operation.
- mode_bit: Signal representing the operation mode add or not add matrix C from SP.
- write_target: Signal specifying the target for write to ScratchPad.

- read_target: Signal specifying the target for read from ScratchPad.
- dataflow_type: not in use.
- dimension_n, dimension_k, dimension_m: Signals representing the dimensions of matrices A, B, and C, respectively.
- reload_operand_a, reload_operand_b: Signals indicating whether to reload operands A and B, respectively (not in use).

Outputs:

- control_register: Output signal containing the configuration settings stored in the control register.

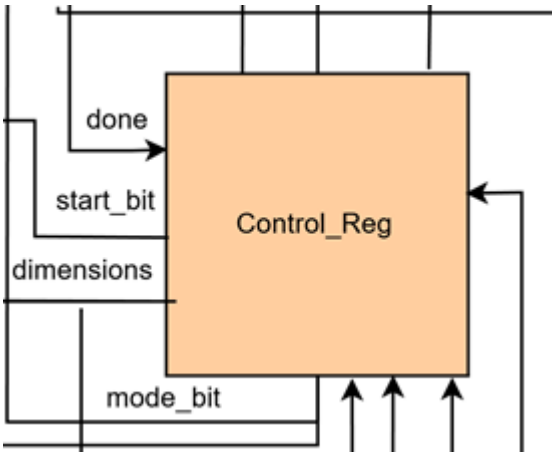


Figure 12: view of the block Control Register

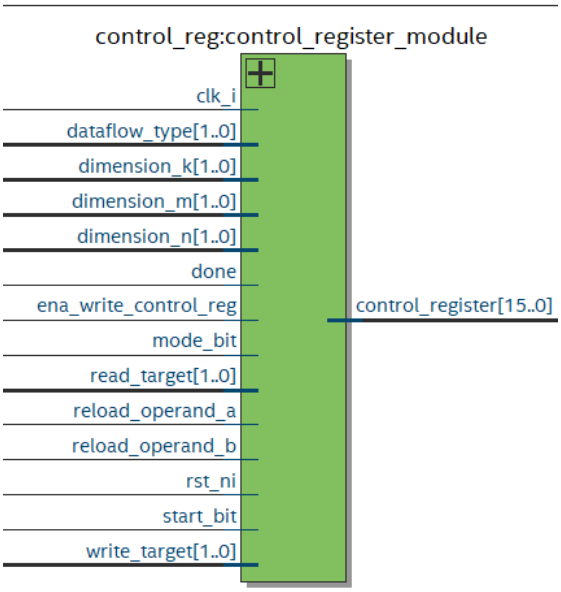


Figure 13: view of the block Control Register

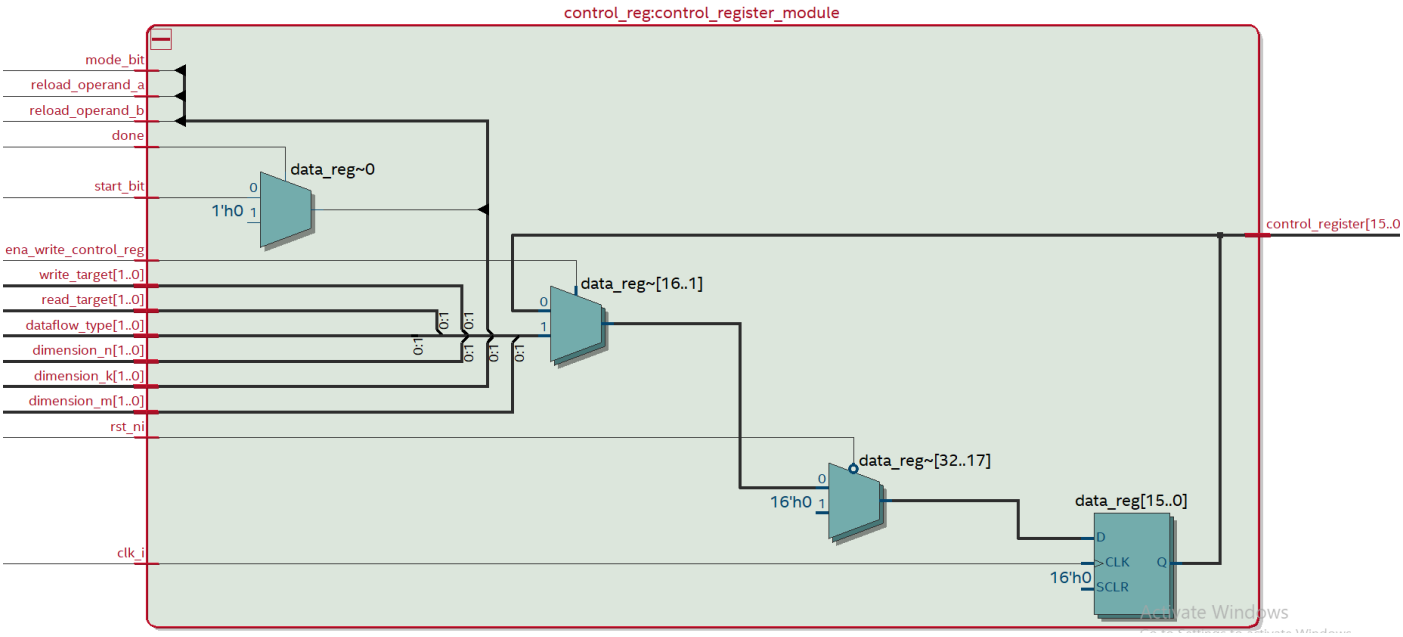


Figure 14: view of the block Control Register

1.1.5 Component #5 Scratchpad

Functional Description:

The Scratchpad module serves as a memory structure with configurable parameters for storing and accessing data. It operates with multiple targets, allowing simultaneous read and write operations to different sections of memory.

Inputs:

clk_i: Clock input.

rst_ni: Reset negative input.

write_target_sp: Address input for the destination inside SP

read_target_sp: Address input for the source in SP to matrix c.

ena_write: Enable signal from busy signal APB.

sub_address_i: Sub-address input for read element from matrix

Data_i: Data input for write to ScratchPad.

Outputs:

Mat_o: Output matrix data static to oprand C.

Data_o: Output data for read from the design.

Internal Signals/Interfaces:

- scratchpad_reg: Register array for storing data of matrix c.
- zeros: Constant value of zero for initialization.

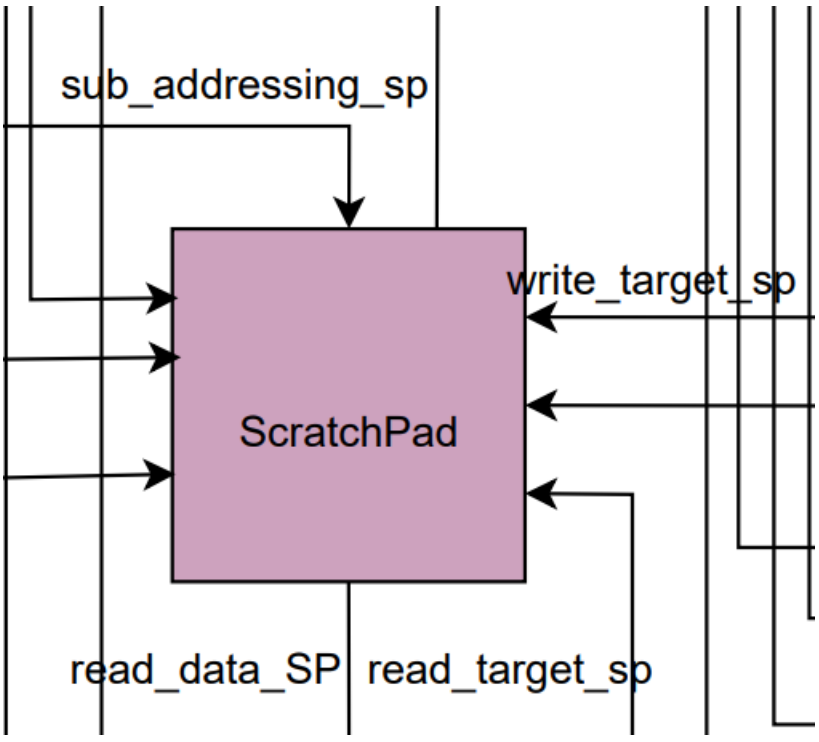


Figure 15: view of the block ScratchPad

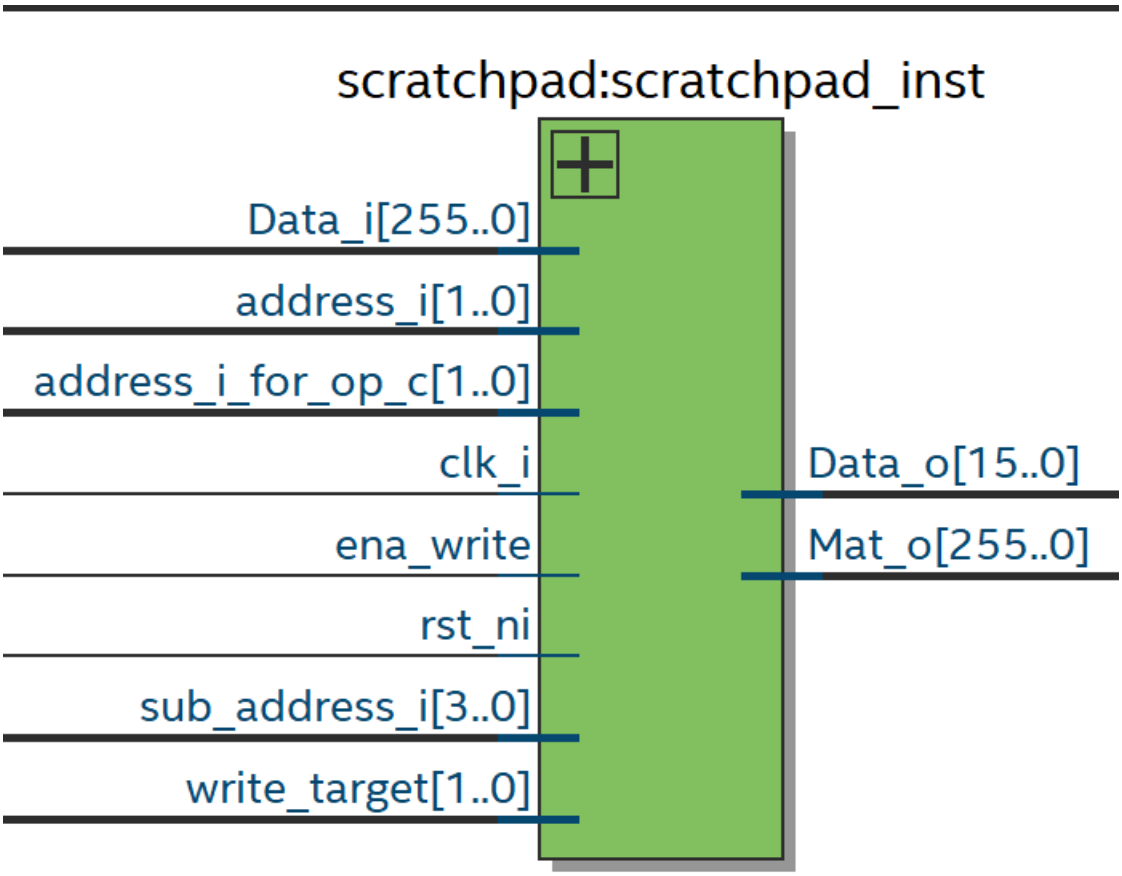


Figure 16: view of the block ScratchPad

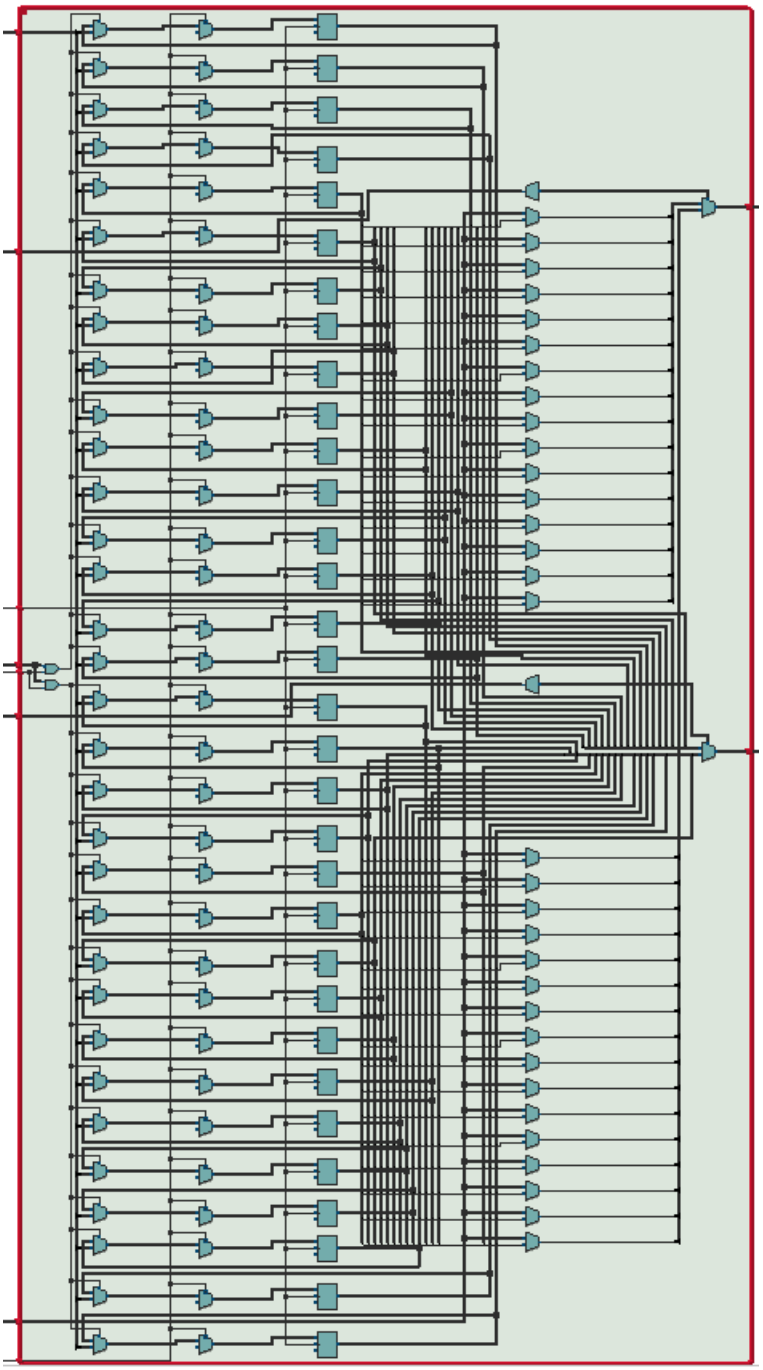


Figure 17: view of the block ScratchPad

1.1.6 Component #6 Adress Decoder

The Address Decoder module is responsible for decoding the address inputs and generating select signals for various memory and control registers within a system.

Inputs:

- clk_i: Clock input.
- rst_ni: Reset input.
- paddr_i: Address input.

Outputs:

- Address_sel_Mat_A: Select signal for memory matrix A.
- Address_sel_Mat_B: Select signal for memory matrix B.
- Address_sel_control_reg: Select signal for control registers.
- Address_sel_SP: Select signal for scratchpad memory.
- Address_sel_flags_reg: Select signal for flags registers.

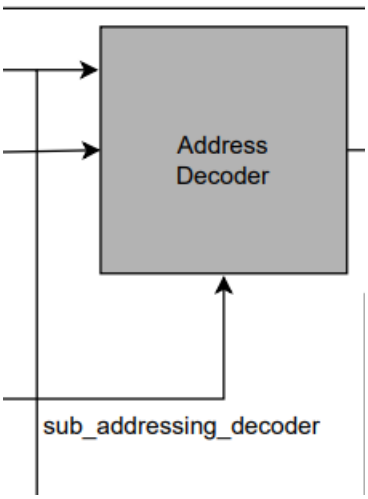


Figure 18: view of the block Address Decoder

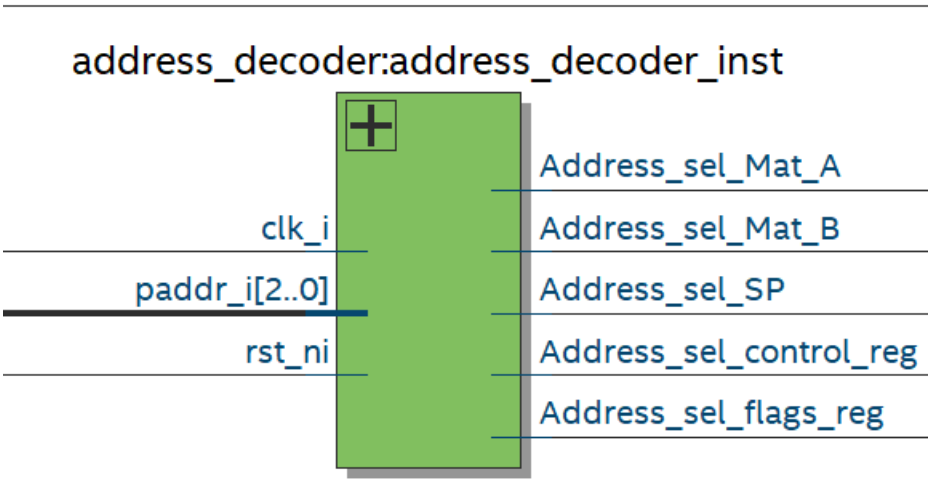


Figure 19: view of the block Address Decoder

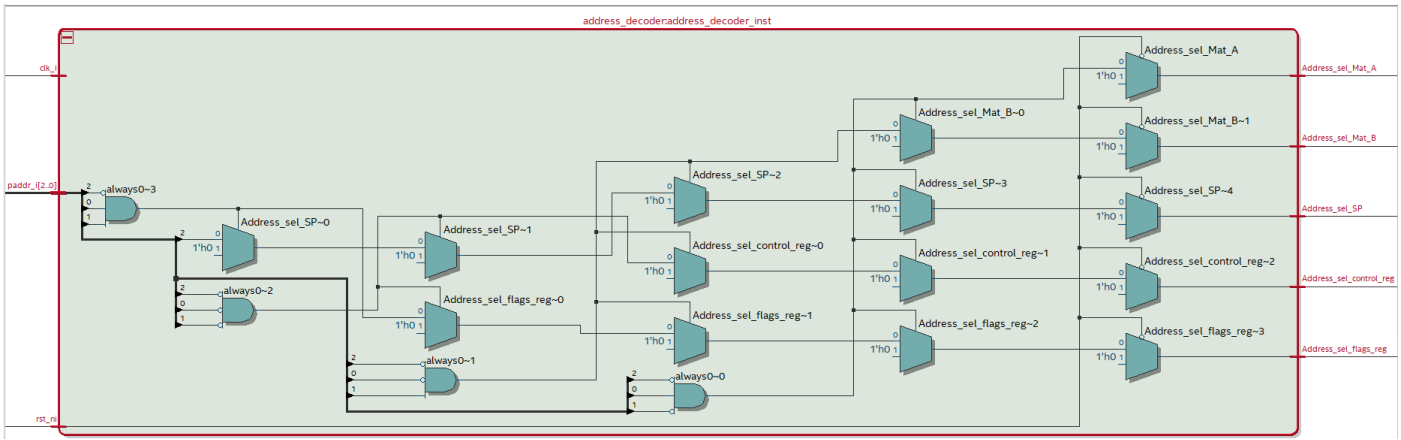


Figure 20: view of the block Address Decoder

1.1.7 Component #7 PE (Processing Element)

Functional Description:

The PE (Processing Element) module is a computational unit designed to perform multiplication and addition operations on input data streams. This module provides a fundamental building block for parallel processing systems, allowing efficient computation of matrix.

Inputs:

- up_i: Input data from the upper neighbor PE or from design to first row PE
- left_i: Input data from the left neighbor PE from design to first column.
- c_i: Input data from matrix c
- clk_i: Clock input.
- rst_ni: Reset negative input.
- start_bit: Start bit signal to initiate the computation.
- mode_bit: Mode bit signal indicating whether to use the matrix c data in computation.

Outputs:

- down_o: Output data to the lower neighbour PE.
- right_o: Output data to the right neighbour PE.
- res_o: Output result of the computation.
- carry_o: Carry output signal.
- start_bit_o: Start bit output signal for synchronization.
- done: Done signal indicating completion of computation.
- counter: Counter signal for tracking computation progress.

Internal Signals/Interfaces:

- res_temp: Temporary result variable.
- res_prev: Previous result variable.
- A_mul_B: Product of input up_i and left_i.
- A_prev: Previous value of input left_i.

- B_prev: Previous value of input up_i.
- zeros: Constant value of zero for comparison.

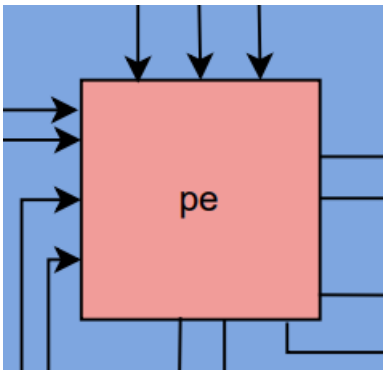


Figure 21: view of the block PE

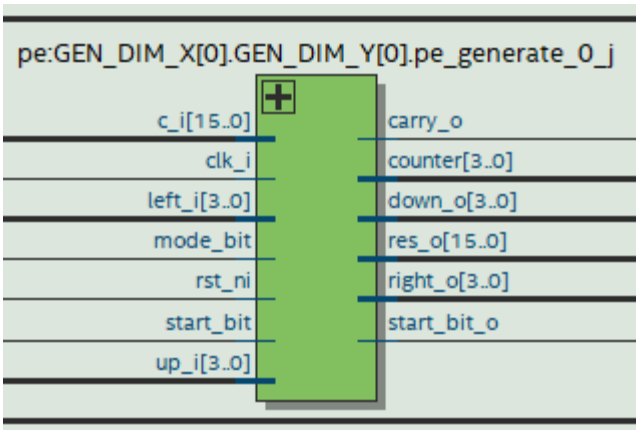


Figure 22: view of the block PE

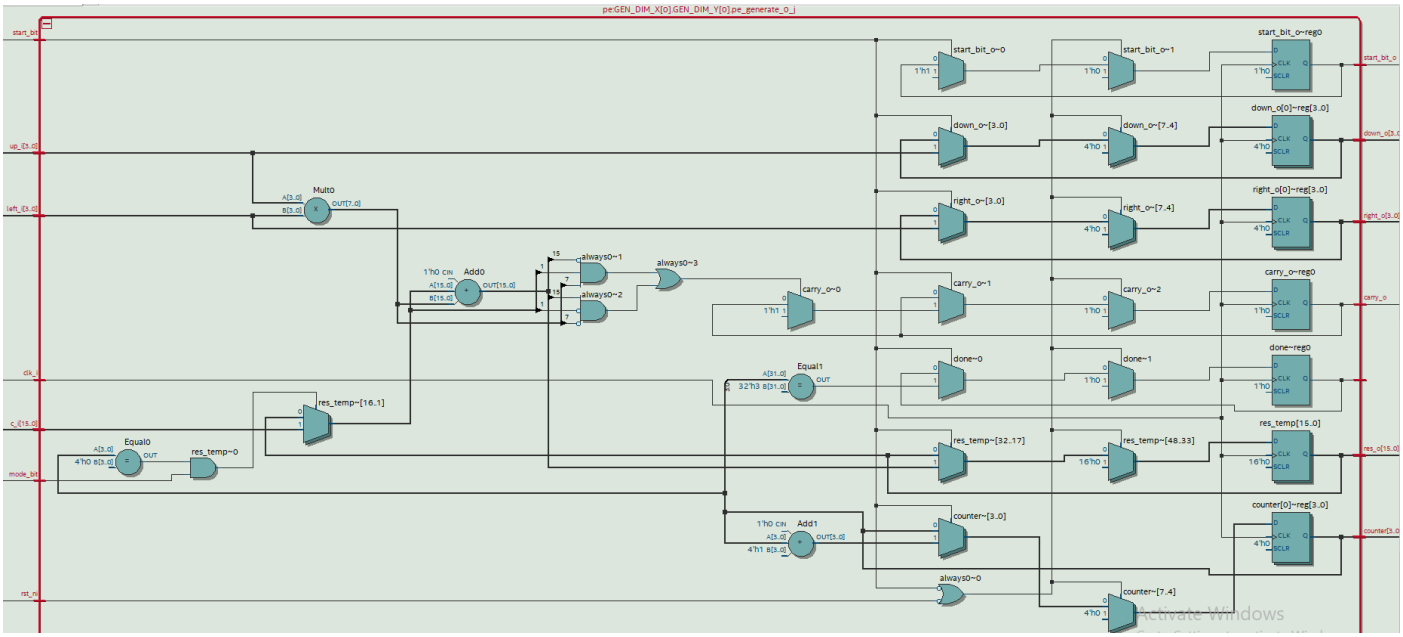


Figure 23: view of the block PE

1.1.8 Component #8 APB Slave

Functional Description:

The APB (Advanced Peripheral Bus) Slave module acts as a slave interface within a system-on-chip (SoC) design, providing connectivity to external peripherals or memory-mapped registers. It follows the Advanced Peripheral Bus protocol, which is commonly used for low-power and low-latency communication in embedded systems.

Inputs:

- clk_i: Clock input.
- rst_ni: Reset negtive input.
- psel_i: Peripheral select input signal.
- penable_i: Peripheral enable input signal.
- pwrite_i: Peripheral write enable input signal when low read enable.
- pwdata_i: Peripheral write data input.

- paddr_i: Peripheral address input.
- pstrb_i: Valid data input for peripheral.
- **Outputs:**
- prdata_o: Read data output.
- pready_o: Ready signal indicating readiness for data transfer.
- pslverr_o: Slave error output signal.
- busy_signal: Busy signal indicating ongoing operation.
- write_ena_control_reg: Write enable signal for control registers.
- write_ena_Mat_A: Write enable signal for matrix A.
- write_ena_Mat_B: Write enable signal for matrix B.
- start_bit: Start bit signal.

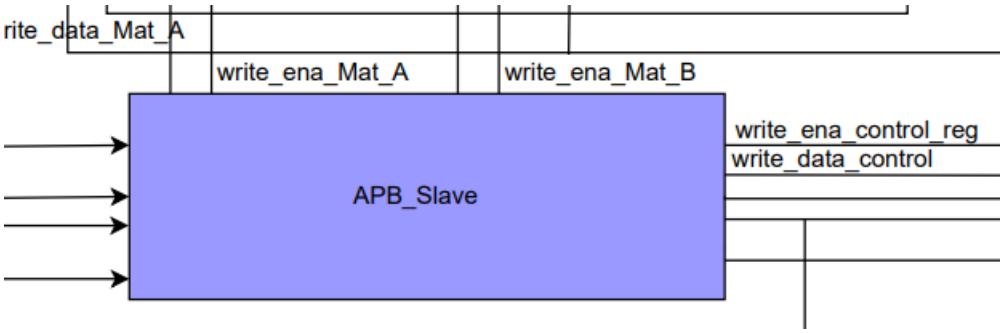


Figure 24: view of the block APB

1.1.9 Component #9 Register Flags

Functional Description:

Register flags in computer hardware are bits stored in a special register to indicate specific conditions or states of the Matmul Calculator like overflow, underflow and carry flags are particularly relevant.

Inputs:

- clk_i: Clock input.
- rst_ni: Reset negtive input.
- flags: carry/ overflow/ underflow input signal

Outputs:

- prdata_o: Read data Register flags output.

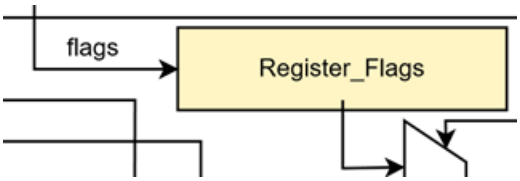
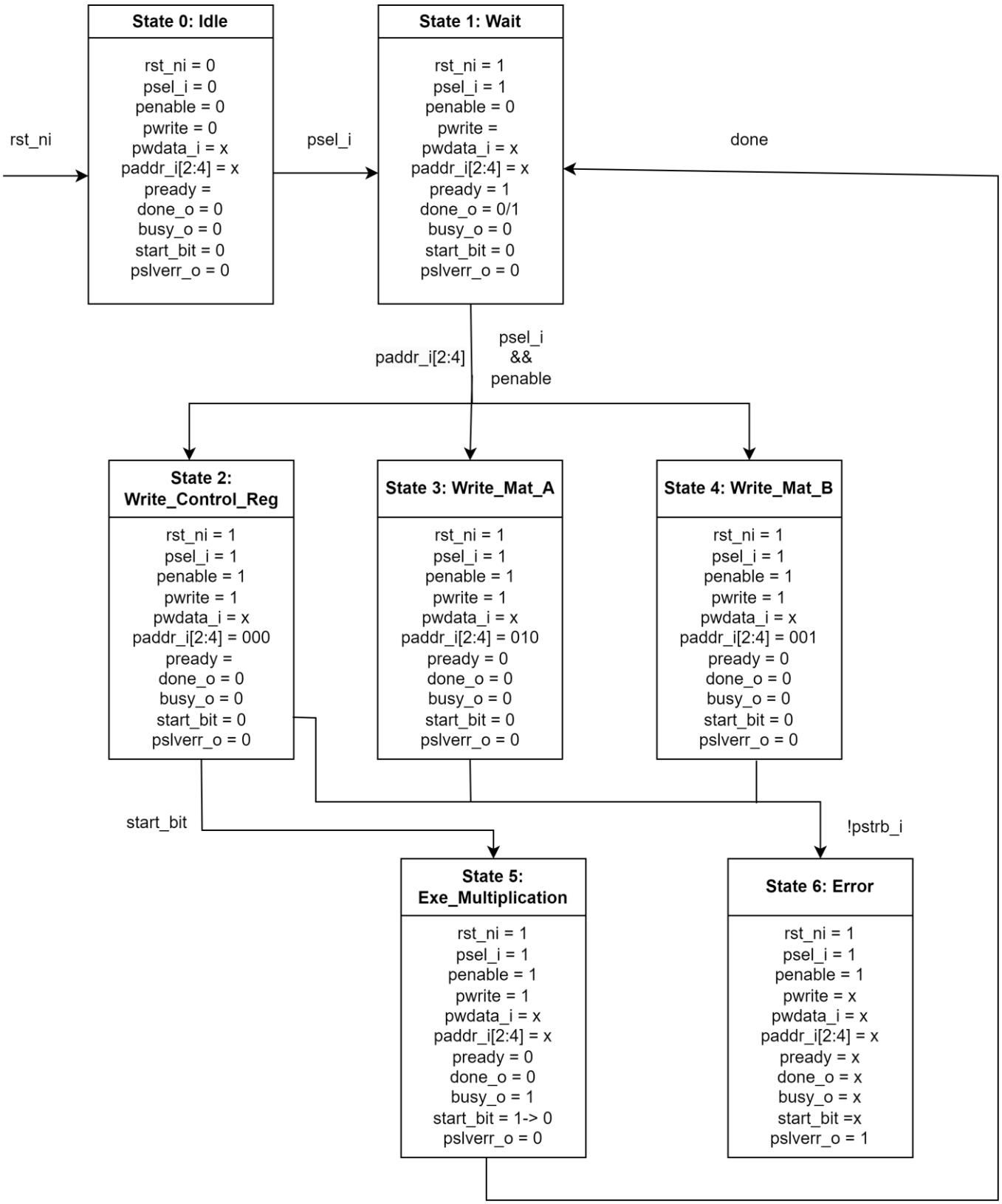


Figure 25: view of the block Register Flags

2. FLOW CHART


State 0: Idle
rst_ni = 0
psel_i = 0
penable = 0
pwrite = 0
pwdata_i = x
paddr_i[2:4] = x
pready =
done_o = 0
busy_o = 0
start_bit = 0
pslverr_o = 0
rst_ni
psel_i
State 1: Wait
rst_ni = 1
psel_i = 1
penable = 0
pwrite =
pwdata_i = x
paddr_i[2:4] = x
pready = 1
done_o = 0/1
busy_o = 0
start_bit = 0
pslverr_o = 0
done
paddr_i[2:4]
psel_i && penable
State 2: Write_Control_Reg
rst_ni = 1
psel_i = 1
penable = 1
pwrite = 1
pwdata_i = x
paddr_i[2:4] = 000
pready =
done_o = 0
busy_o = 0
start_bit = 0
pslverr_o = 0
State 3: Write_Mat_A
rst_ni = 1
psel_i = 1
penable = 1
pwrite = 1
pwdata_i = x
paddr_i[2:4] = 010
pready = 0
done_o = 0
busy_o = 0
start_bit = 0
pslverr_o = 0
State 4: Write_Mat_B
rst_ni = 1
psel_i = 1
penable = 1
pwrite = 1
pwdata_i = x
paddr_i[2:4] = 001
pready = 0
done_o = 0
busy_o = 0
start_bit = 0
pslverr_o = 0
start_bit
!pstrb_i
State 5: Exe_Multiplication
rst_ni = 1
psel_i = 1
penable = 1
pwrite = 1
pwdata_i = x
paddr_i[2:4] = x
pready = 0
done_o = 0
busy_o = 1
start_bit = 1-> 0
pslverr_o = 0
State 6: Error
rst_ni = 1
psel_i = 1
penable = 1
pwrite = x
pwdata_i = x
paddr_i[2:4] = x
pready = x
done_o = x
busy_o = x
start_bit =x
pslverr_o = 1

3. APPENDIX

3.1 Terminology

PE - Processing Element

SP - Scratchpad

APB - Advanced Peripheral Bus.

ENA - Enable

MAT - Matrix

MUL - Multiplication

RST - Reset

CLK - Clock

Addr - Address