Scaled Dot-Product Attention for Transformer Models

Name: Nirmal Kumar Sedhumadhavan
Unityid: nsedhum@ncsu.edu
StudentID: 200483323

Delay (ns to run provided provided example). Clock period: 5.6 ns	Logic Area: 5973.828 (um^2) Memory: 205 MBytes	1/(delay.area) $2.9892 * 10^{-5}$ ($ns^{-1}.um^{-2}$)

Introduction

This project focuses on the hardware design of a Scaled Dot-Product Attention mechanism, a core component of Transformer models. The design performs a series of matrix operations to emulate the attention mechanism in hardware. Initially, input matrix I is multiplied with weight matrix W to generate the Query (Q), Key (K), and Value (V) matrices. These matrices are stored in the output SRAM.

Subsequently, the Score (S) matrix is computed by performing a matrix multiplication between the Q and K matrices. The resulting S matrix is stored in the output SRAM as well.

Finally, the Scaled Dot-Product Attention (Z) is computed by multiplying the S matrix with the V matrix. The resulting Z matrix is stored in the output SRAM. This final output is then validated through a series of correctness checks to ensure proper functionality of the design.

Implementation

The matrix multiplication process begins by computing the product of the input matrix and the weight matrix. For each operation, an element from a row of the input matrix and the corresponding element from a column of the weight matrix are fetched and multiplied. The result is then accumulated using a Multiply-Accumulate (MAC) unit.

This process continues element by element across the row of the input matrix and the corresponding column of the weight matrix. Once the entire row-column pair has been processed, the accumulated result is written to a designated location in the scratchpad SRAM.

The system then continues with the same row of the input matrix and proceeds to the next column of the weight matrix. This step is repeated until all columns of the weight matrix have been processed for the current row. After completing one row of the input matrix, the system moves to the next row and restarts the process with the first column of the weight matrix.

This sequence is repeated until the entire input and weight matrices have been multiplied, and the full result matrix is computed and stored in the scratchpad SRAM.

After the computation of the Q, K, and V matrices, the Q matrix is stored in the output SRAM, while both the K and V matrices are saved in the output SRAM as well as the scratchpad SRAM. To compute the Score (S) matrix, the system reads the Q matrix from the output SRAM and the K matrix from the scratchpad SRAM, performs the necessary matrix multiplication, and stores the resulting S matrix in both the output and scratchpad SRAMs.

Next, the Scaled Dot-Product Attention (Z) matrix is calculated by multiplying the S matrix, retrieved from the output SRAM, with the V matrix, accessed from the scratchpad SRAM. The resulting Z matrix is stored in the output SRAM. Once the Z matrix is available, the output SRAM, which now contains all intermediate and final results, is used for verification and correctness testing of the entire attention computation process.

2. Interface Specification

Signal Name	Width	Function/Description
current_state	4 bits	Current state of the state machine.
next_state	4 bits	Next state of the state machine.
set_dut_ready	1 bit	Control signal to set the DUT ready status.
compute_complete	1 bit	Set high when computation is complete.
get_array_size	1 bit	Control signal to get the size of the array.
save_array_size	1 bit	Control signal to save the size of the array.
mac	32 bit	Multiply-accumulate result register
col_a	3 bit	Column index for matrix A
Scratch_counter	3 bit	Counter for scratchpad-related iterations
row_a	4 bit	Row index for matrix A

col_b	4 bit	Column index for matrix B
row_counter	4 bit	General row counter
col_counter	4 bit	General column counter
counter	6 bit	General element counter
W_size	6 bit	Size of the W matrix
QKV_size	6 bit	Size of the Q, K, V matrix
Scratch_size	6 bit	Size of scratchpad matrix
S_size	7 bit	Size of S matrix
dut__tb__sram_result_write_enable_r	1 bit	Enable signal for writing to result SRAM
dut__tb__sram_scratchpad_write_enable_r	1 bit	Enable signal for writing to scratchpad SRAM
dut__tb__sram_scratchpad_read_address_r	5 bit	Read address for scratchpad SRAM
dut__tb__sram_scratchpad_write_address_r	5 bit	Write address for scratchpad SRAM
dut__tb__sram_input_read_address_r	6 bit	Read address for input SRAM
dut__tb__sram_weight_read_address_r	7 bit	Read address for weight SRAM
dut__tb__sram_result_read_address_r	7 bit	Read address for result SRAM
dut__tb__sram_result_write_address_r	8 bit	Write address for result SRAM
dut__tb__sram_result_write_data_r	32 bit	Data to write to result SRAM
dut__tb__sram_scratchpad_write_data_r	32 bit	Data to write to scratchpad SRAM

counter_sel	2 bit	Selects which counter to use/control
row_counter_sel	2 bit	Selector for row counter operations
col_counter_sel	2 bit	Selector for column counter operations
Scratch_counter_sel	2 bit	Selector for scratchpad counter operations
sram_write_enable_sel	2 bit	Selects which SRAM write enable is active
result_write_addr_sel	2 bit	Selects which result write address input to use
scratch_write_addr_sel	2 bit	Selects which scratchpad write address input to use
weight_read_addr_sel	3 bit	Selects source for weight read address
input_read_addr_sel	3 bit	Selects source for input read address
result_read_addr_sel	3 bit	Selects source for result read address
scratch_read_addr_sel	3 bit	Selects source for scratchpad read address
compute_mac	2 bit	Control for MAC (Multiply-Accumulate) operation
KVS_Flag	2 bit	Flag for QKV-type dataflow control

3. Technical Implementation
FSM Attached as JPEG file

4. Results Achieved

Clock Period: 5.6 ns
Area: 5973.828 um^2
Performance: $2.9892 * 10^{-5}$ ($ns^{-1}.um^{-2}$)