Flash Attention on FPGA

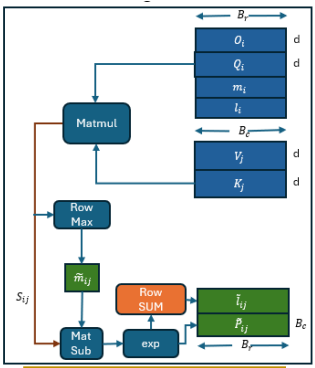
Aakarsh A
Department of Computer Science and Automation
Indian Institute of Science

May 23, 2025

Overview

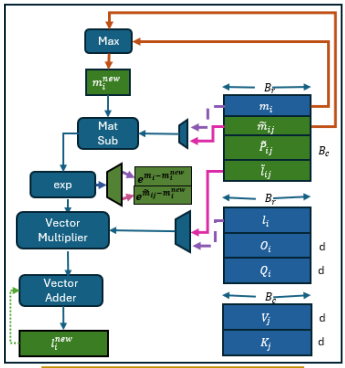
1. Flash Attention Implementation in brief
2. e^{-x} Implementation
3. Future Work

Block Level Design



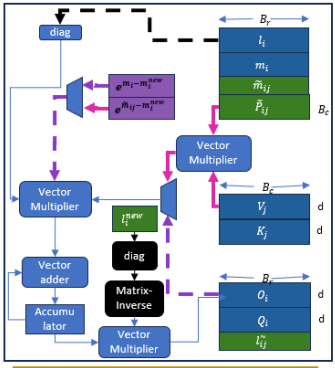
Phase 1 : Calculation of $\tilde{m}_{ij}$, $\tilde{P}_{ij}$ and $\tilde{l}_{ij}$

- $S_{ij} = Q_i K_j^T$
- $\tilde{m}_{i,j} = rowmax(S_{i,j})$
- $\tilde{P}_{ij} = \exp(S_{ij} - \tilde{m}_{ij})$
- $\tilde{l}_{ij} = rowsum(\tilde{P}_{ij})$



Phase 2 : Calculation of m_i^{new} and l_i^{new}

- $m_i^{new} = \max(m_i, \tilde{m}_{ij})$,
- $l_i^{new} = e^{m_i - m_i^{new}} l_i + e^{\tilde{m}_{ij} - m_i^{new}} \tilde{l}_{ij}$



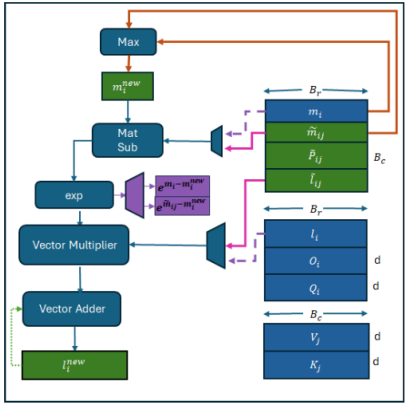
Phase 3 : Calculation of O_i

- $O_i = diag(l_i^{new})^{-1} \times \left(diag(l_i) e^{m_i - m_i^{new}} O_i^{old} + e^{\tilde{m}_{ij} - m_i^{new}} \tilde{P_{ij}} V_j \right)$

Focus on Phase - 2

Phase 2 : Calculation of m_i^{new} and l_i^{new}

- $m_i^{new} = \max(m_i, \tilde{m}_{ij})$,
- $l_i^{new} = e^{m_i - m_i^{new}} l_i + e^{\tilde{m}_{ij} - m_i^{new}} \tilde{l}_{ij}$



RTL Components

- Negative Exponential Engine
- BRAM/URAM Access and Storage
- Timing and Control Unit
- Vector Multiplier
- Vector Adder
- Subtractor
- Comparator

e^{-x} Implementation

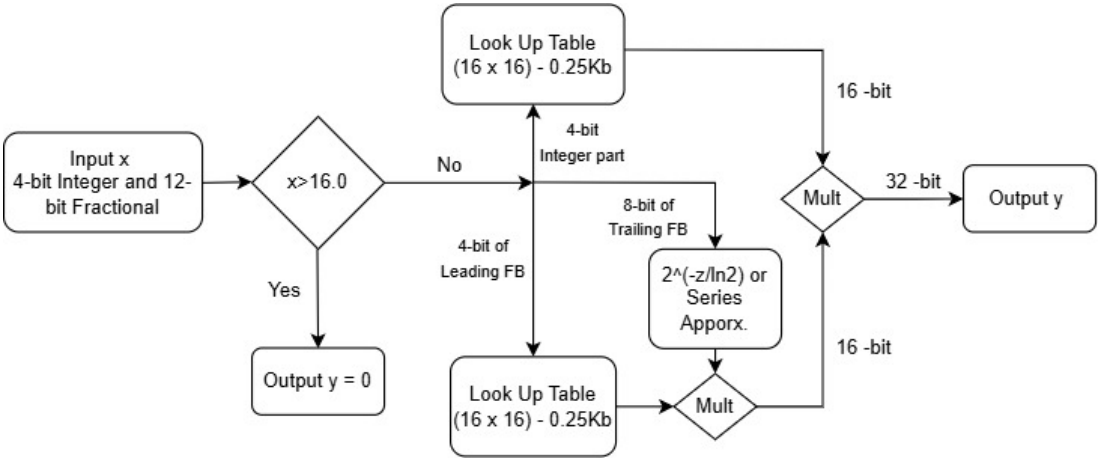
Different Approaches for RTL Implementation

	Latency - CC	**Memory Usage**	**Compute Resources**
CORDIC	High - O(n)	Low - LUT	Shift registers & adders
Piece-Wise Approximation	2 - 3	Medium - Accuracy parameter	Few DSPs & adders
Look Up Table Approach	1	High - Accuracy parameter	Null
Hybrid (LUT & $2^{\frac{x}{\ln 2}}$)	1 - 2	Low - Store in LUTs	Few DSPs & adders
Hybrid (LUT & Series Approx.)	1 - 2	Low - Store in LUTs	Few DSPs & adders

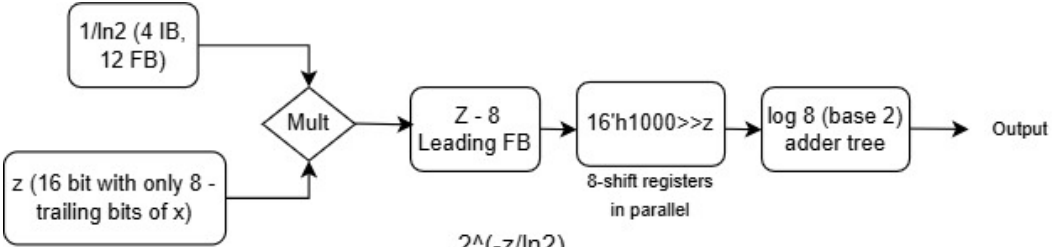
Table: Brief Comparative Study on a similar accuracy scale

Every approach in the above table can be pipelined, and the maximum operating frequency is dependent on the hardware.

Hybrid Approach: Part 1



Hybrid Approach: Part 2



Condition for operation in 1 clock cycle:

3 Multipliers + 1 Comparator + 1 shift operation + 3 adders less than T_c

The above condition is chosen based on the slowest path. Based on our frequency of operation, we can make it for 2 clock cycles as well.

Why Hybrid Approach?

Our goal: Hardware acceleration.

- Faster Memory Access (Infer LUTs) at the expense of hardware (DSPs).
- Minimize Latency and maximize operation frequency.

Future Work

1. Accuracy comparison between different implementation techniques.
   Target error: Less than 0.005
2. RTL implementation of the Timing and Control Unit, along with other subunits,
3. Verification of Phase-2 on the Alveo V80 FPGA.

The End