LLM Transformer Decoder RTL Accelerator — Technical Report

1. Introduction

This report documents the design, implementation, and verification of a synthesizable RTL accelerator for LLM (Large Language Model) transformer decoder inference. The accelerator implements a single transformer decoder block — the fundamental repeating unit of GPT-style language models — in SystemVerilog, targeting FPGA and ASIC deployment.

The design provides two architectural variants: a **register-bridge** variant that copies BRAM weights into register arrays for single-cycle combinational access (maximum throughput), and a **streaming** variant that reads weights directly from BRAM one element per cycle (minimum area). Both share the same external interface, BRAM storage, and verification infrastructure.

Key design features include Q8.8 fixed-point arithmetic, systolic array compute, BRAM-backed weight and KV-cache storage, division-free softmax normalisation via reciprocal LUT with Newton-Raphson refinement, and packed array ports for full simulation visibility.

2. Background

2.1 Transformer Decoder Architecture

The transformer decoder processes tokens autoregressively: given a sequence of prior tokens, it predicts the next token. Each decoder layer applies two sub-layers with residual connections. First, Multi-Head Self-Attention computes attention scores between the current query and all prior key/value pairs, enabling the model to attend to relevant context. Second, a Feed-Forward Network (FFN) applies a two-layer MLP with non-linear activation independently to each position. Both sub-layers are preceded by Layer Normalization (pre-norm architecture) and followed by residual additions.

2.2 Inference vs Training

This accelerator targets inference only, which has distinct characteristics compared to training. Batch sizes are typically 1 for interactive generation. Only the forward pass is needed with no backpropagation. KV-cache eliminates redundant attention computation. Weights are fixed, enabling weight-stationary dataflows and BRAM preloading.

2.3 Fixed-Point Quantization

Modern LLM inference widely uses quantization to reduce memory bandwidth and compute cost. Our Q8.8 fixed-point format (8 integer bits, 8 fractional bits) provides a range of −128.0 to +127.996 with 1/256 resolution, which is sufficient for demonstrating the architectural concepts while keeping multiplier width at 16 bits.

3. Architecture

3.1 Top-Level Block Diagram

The design has two levels of hierarchy. The synthesis top-level wraps the compute core with BRAM-backed storage for all 49,344 weight parameters and 16,384 KV-cache entries. Two top-level variants exist: `transformer_decoder_top` (register-bridge) and `transformer_decoder_top_stream` (direct BRAM streaming).

The compute core implements the pre-norm architecture used by GPT-2, LLaMA, and most modern LLMs: LayerNorm 1, Multi-Head Attention with KV-cache, residual addition, LayerNorm 2, Feed-Forward Network with ReLU, and a second residual addition.

3.2 Module Hierarchy

The full project comprises 17 RTL source files in two parallel hierarchies:

Register-bridge variant (original — maximum throughput):

```
transformer_decoder_top          (BRAM → register arrays → decoder)
├── bram_sp ×12                  (Weight/bias/parameter storage)
├── kv_cache_bram ×2             (K and V caches)
│   └── bram_dp                  (True dual-port BRAM)
└── transformer_decoder          (Decoder compute core)
    ├── layer_norm ×2            (Pre-attention & pre-FFN normalisation)
    ├── multi_head_attention     (Causal multi-head self-attention)
    │   └── softmax_unit          (Reciprocal-LUT softmax, time-multiplexed)
    ├── feed_forward              (Two-layer FFN with ReLU)
    ├── systolic_array            (Matrix multiply engine)
    │   └── processing_element    (Single MAC unit)
    └── transformer_pkg           (Parameters, types, FP utilities)
```

Streaming variant (new — minimum area):

```
transformer_decoder_top_stream   (BRAMs → address/data → decoder)
├── bram_sp ×12                   (Same BRAMs, muxed read ports)
├── kv_cache_bram ×2              (Element-level BRAM reads)
│   └── bram_dp
└── transformer_decoder_stream    (Streaming decoder core)
    ├── layer_norm ×2             (Unchanged — small packed ports)
    ├── multi_head_attention_stream (BRAM addr/data weight reads)
    │   └── softmax_unit          (Unchanged)
    ├── feed_forward_stream       (BRAM addr/data weight reads)
    ├── systolic_array            (Shared infrastructure)
    │   └── processing_element
    └── transformer_pkg
```

Shared modules (transformer_pkg, bram_sp, bram_dp, kv_cache_bram, softmax_unit, layer_norm, systolic_array, processing_element) are identical between variants. The streaming variant adds 4 new modules: multi_head_attention_stream, feed_forward_stream, transformer_decoder_stream, and transformer_decoder_top_stream.

3.3 Module Descriptions

3.3.1 Processing Element (PE)

The PE is the atomic compute unit — a single multiply-accumulate (MAC) cell. It receives an activation from the left and a weight from the top, computes acc += a × w using full 32-bit precision, and forwards both operands to neighboring PEs. The 32-bit accumulator prevents overflow during long dot-product chains. Key properties include single-cycle MAC latency, registered inputs and outputs for timing closure, synchronous clear for accumulator reset between tiles, and parameterized data widths via transformer_pkg.

3.3.2 Systolic Array

A 4×4 grid of PEs forms the systolic matrix multiplication engine. Data flows through the array in a wave pattern: activations propagate left-to-right and weights top-to-bottom. After ROWS + COLS − 1 cycles of streaming, the 16 accumulators hold the complete output tile. This architecture achieves 16 MACs per cycle with only 8 input operands, giving a 2× data reuse factor.

3.3.3 Softmax Unit (Division-Free)

Softmax normalisation computes probs[i] = exp(scores[i] − max) / Σexp. The implementation proceeds through six FSM stages: find maximum score for numerical stability, subtract maximum and apply PWL exponential approximation, accumulate the sum of exponentials, compute the reciprocal of the sum via LUT + Newton-Raphson, and normalise by multiplying each exponential by the reciprocal.

The PWL exponential uses four linear segments covering [−8, 0] with non-negative clamping at segment boundaries. The normalisation step uses a 32-entry reciprocal LUT (indexed by the top 5 mantissa bits of the CLZ-normalised sum) providing an initial estimate in Q2.14. One Newton-Raphson iteration doubles precision to approximately 12 bits. The entire reciprocal computation is combinational (single-cycle) using a 512-bit ROM, two 16×16 multipliers, and a barrel shifter. Maximum error versus exact division is ±1 LSB.

3.3.4 Layer Normalization

LayerNorm computes the mean and variance of the input vector, then normalizes each element through three FSM stages: mean computation via sequential accumulation, variance computation using centered differences, and element-wise normalization with learnable gamma and beta parameters. The reciprocal square root uses a 4-entry LUT.

3.3.5 Multi-Head Attention

The attention module implements multi-head scaled dot-product attention with integrated softmax and KV-cache support. A single softmax_unit (VEC_LEN = 128) is time-multiplexed across N_HEADS = 4 heads. The register-bridge variant receives full weight matrices on array ports. The streaming variant replaces these with BRAM address/data interfaces, reading Wq/Wk/Wv in lockstep with three parallel accumulators.

3.3.6 Feed-Forward Network

The FFN implements FFN(x) = ReLU(x × W1 + b1) × W2 + b2. The register-bridge variant computes full dot products per cycle. The streaming variant reads W1, b1, and W2 from BRAM one element per cycle. Both share identical ReLU logic.

3.3.7 BRAM Modules

Three BRAM types: bram_sp (single-port, read-first, optional hex init), bram_dp (true dual-port), and kv_cache_bram (vector-write FSM + element-level read).

4. Streaming Architecture

4.1 Motivation

The register-bridge architecture copies all 49,344 BRAM words into register arrays, consuming approximately 789,504 flip-flops. The streaming architecture eliminates these by having compute modules generate BRAM read addresses directly from their FSM state.

4.2 Weight Read Protocol

Each streaming module exposes address/data/enable ports for its weight BRAMs. The top-level muxes each BRAM's read port between the weight-loading bus and the compute address. BRAM latency is one cycle: address issued in cycle N, data arrives in cycle N+1.

For QKV projection, Wq, Wk, and Wv share the same read address. Three accumulators run simultaneously, producing Q, K, V for one dimension in D_MODEL cycles.

4.3 Performance Comparison

Metric	Register-Bridge	Streaming	Change
Total pipeline (seq_pos=0)	~3,200 cycles	~45,000 cycles	~14× slower
Weight registers	789K FFs	5K FFs	**99.4% reduction**
Weight multipliers	~192 combinational	6 sequential	**97% reduction**

Metric	Register-Bridge	Streaming	Change
DSP48 usage	~150	~10	**93% reduction**

5. Verification

5.1 Summary

Total: **83/83 tests pass** (54 behavioural + 29 RTL simulation across 6 testbenches).

5.2 Behavioural Model (54 tests)

Fixed-Point Utilities (16/16), Processing Element (9/9), Systolic Array (8/8), Softmax Unit (8/8), Layer Normalization (5/5), Feed-Forward Network (4/4), Decoder Integration (12/12).

5.3 RTL Simulation (29 tests)

Processing Element (6/6), Systolic Array (4/4), Softmax (4/4), BRAM (8/8), Decoder — Register-Bridge (4/4), Decoder — Streaming (3/3).

The streaming decoder testbench loads weights via the wl_en bus, then verifies single-token and sequential two-token inference. Output out_emb[0] = 0x00A8 (0.656) matches the register-bridge variant's first dimension.

5.4 RTL Lint

17 RTL files: 0 errors, 5 warnings (BRAM blocks intentionally omit reset).

6. Implementation Metrics (Estimated)

Block	DSP48	FFs	LUTs	BRAM18K
Processing Element	1	~30	~20	—
4×4 Systolic Array	16	~500	~350	—
Softmax Unit	2	~2K	~3K	—
Layer Normalisation	2–4	~300	~500	—
Multi-Head Attention	~64	~5K	~8K	—

Block	DSP48	FFs	LUTs	BRAM18K
Feed-Forward Network	~64	~3K	~5K	—
Weight BRAMs (12)	—	—	—	~110
KV-Cache BRAMs (2)	—	—	—	~16
Full Decoder (register-bridge)	**~150**	**~11K**	**~17K**	**~126**
Full Decoder (streaming)	**~10**	**~5K**	**~12K**	**~126**

Clock frequency estimate: 100–200 MHz.

7. Limitations and Future Work

7.1 Resolved

BRAM-backed weights, division-free softmax, packed array ports, streaming weight architecture (99.4% FF reduction).

7.2 Remaining

LayerNorm still uses `/` operator and coarse rsqrt LUT. Only ReLU activation. Systolic array not connected to datapath. No batching.

7.3 Planned

LayerNorm right-shift + reciprocal-LUT, systolic-tiled projections, parallel softmax, GELU/SiLU PWL, multi-layer stacking, multi-device distribution, AXI-Lite interface.

8. Conclusion

This project demonstrates a complete, synthesizable transformer decoder block in SystemVerilog with two architectural variants: a high-throughput register-bridge design and a minimum-area streaming design. The 17-module hierarchy mirrors the transformer's conceptual structure while addressing practical synthesis concerns. All 83 verification tests pass across both variants. The streaming architecture achieves 99.4% register reduction while maintaining functional equivalence, making the design feasible on the smallest FPGA targets.