AIMS Mathematics, 11(2): 4220–4242.
DOI: 10.3934/math.2026169
Received: 27 October 2025
Revised: 16 January 2026
Accepted: 22 January 2026
Published: 10 February 2026

https://www.aimspress.com/journal/Math

Research article

Research on post-equalization technology of a visible light communication system based on improved LSTM model and FPGA

Ke Xiong[1,2,*]

[1] Application Management Department, Digital and Intelligent Operation Center, Guangdong Power Grid Co., Ltd., Guangzhou 510000, China
[2] School of Information Science and Technology, University of Science and Technology of China (USTC), Hefei 230026, China

* **Correspondence:** Email: xiongke@gdxx.csg.cn, engpgn1047@sina.com.

Abstract: In this study, I proposed a post-equalization method for high-speed visible light communication (VLC) systems using an improved long short-term memory (LSTM) model jointly optimized with a field-programmable gate array (FPGA) to address nonlinear distortion, multipath interference, and model deployment complexity. The LSTM structure was simplified by merging three gates into two and compressing the hidden state to 64 dimensions, while 8-bit fixed-point quantization reduced computational cost. The received signal was divided into a 32-point sliding buffer and aligned in zero phase for precise temporal-symbol matching. The computational graph was modularized into matrix operations, activation, and state update stages, each implemented as dedicated FPGA modules in a three-stage pipeline on an Artix-7 platform. Through loop unrolling and resource sharing, the design achieved a 16-cycle inference latency at 100 MHz with minimal logic utilization. Experimental results demonstrated superior performance, with symbol error rate (SER) ≤ 0.0031 and normalized mean square error (NMSE) ≤ 0.0115 across LED drive currents. Under 8-PAM modulation, total harmonic distortion (THD) and error vector magnitude (EVM) were maintained at 3.45% and 4.32%, respectively, showing enhanced nonlinear tolerance and equalization stability. As the multipath delay spread increased from 5 ns to 25 ns, the average intersymbol interference ratio (ISI-Ratio) improved from 20.7 dB to 28.4 dB, and the dynamic timing error (DTE) remained within 1.82×10^{-3}–3.14×10^{-3}. The lookup table (LUT) usage was kept within 78.2% ± 0.9%, and the inference latency was (0.160 ms ± 0.003 ms). These results indicated that the proposed method effectively mitigates multipath effects and achieves an optimal trade-off between algorithmic compactness and hardware efficiency, providing a closed-loop optimization pathway for intelligent equalization in resource-constrained optical communication systems.

Keywords: post-equalization technique; communication system; long short-term memory; field-programmable gate array; visible light communication
Mathematics Subject Classification: 68T07, 94A12

1. Introduction

Visible light communication (VLC) leverages LED light sources for high-speed data transmission. It offers inherent electromagnetic compatibility (EMC), physical-layer security, and indoor positioning capability, and is widely regarded as a promising solution for next-generation short-range wireless access [1,2]. However, device nonlinearities, environmental multipath, and a finite modulation bandwidth yield channel characteristics that require high-dimensional temporal modeling for reliable back-end signal recovery [3]. Post-equalization, situated between the physical layer and the decoder, directly affects system throughput and implementation cost (e.g., algorithmic complexity and hardware feasibility) [4–6], and is therefore a key enabler in translating VLC from laboratory prototypes to commercial products.

Most contemporary equalizers do not strike an ideal balance between algorithmic expressiveness and hardware adaptability. While deep-learning approaches can learn approximate channel inverses, conventional RNNs entail large parameter counts and complex computational graphs, hindering real-time execution on resource-constrained embedded platforms [7,8]. Some researchers introduce lightweight convolutions or attention mechanisms, yet they often neglect the strong temporal dependence and bursty nonlinearities characteristic of VLC channels, thereby limiting generalization [9,10]. At the hardware level, many implementations rely on general-purpose processors or high-power general processor units (GPUs), which cannot satisfy the low-latency, low-power, and compact-form-factor requirements of VLC terminals [11,12]. Although certain designs migrate to FPGAs for acceleration, they often retain floating-point arithmetic or suboptimal dataflow, which degrade performance and waste logic resources and memory bandwidth [13,14]. Critically, input preprocessing lacks precise timing alignment: Sliding windows are not synchronized to symbol boundaries, introducing phase noise and degrading equalization performance [15,16]. These issues collectively impede the practical realization of intelligent post-equalization for VLC systems.

Early VLC equalization research focused on linear methods such as zero-forcing and minimum mean-square-error filters, which are adequate for mild multipath but fail to compensate for nonlinear distortion [17,18]. To enhance nonlinear handling, Volterra-series equalizers with higher-order kernels were explored for modeling LED driver nonlinearities; however, parameter counts grow exponentially with order, limiting scalability to high-speed systems [19,20]. Subsequent efforts adopted neural networks, multilayer perceptrons and radial-basis function networks, for channel inversion, but their static mappings are ill-suited to time-varying multipath [21,22]. The introduction of recurrent structures was a breakthrough: Standard LSTMs capture long-range dependencies and can markedly reduce bit error rates in simulation [23,24]. However, the canonical LSTM employs three independent gates (input, forget, output), leading to parameter redundancy; hidden states are often set to 128 or 256, exceeding the complexity typically required by VLC channels. Although knowledge distillation-based compression has been attempted, the lack of concurrent hardware mapping optimization limits deployment efficiency gains [25]. In short, prior methods advanced algorithmic performance but did not establish a closed-loop pathway from model simplification to hardware deployment.

Researchers have begun to explore algorithm–hardware co-design. Some researchers deploy GRUs on Zynq SoCs, using the ARM (Advanced RISC Machines) core for preprocessing and the FPGA fabric for accelerated inference, achieving microsecond-level latency at 10 MHz. Nevertheless, the GRU's memory capacity for long sequences is weaker than that of the LSTM, and without quantization, resource usage remains high [26,27]. Other efforts emphasize fixed-point quantization, for example, compressing CNN weights to 4 bits on a Kintex-7, to reduce power consumption; however, convolutional structures struggle with the non-stationary temporal characteristics of VLC channels [28,29]. Transformer encoders, leveraging positional encoding and self-attention, can perform well offline but impose high computational complexity that challenges real-time symbol recovery [30]. Some teams decompose LSTM computations into matrix operations mapped onto DSP48E1 units, yet preserving the full three-gate structure still consumes excessive logic, precluding deployment on cost-sensitive Artix devices [31]. These attempts validate the feasibility of deep-learning-based equalization while exposing common shortcomings: Structural redundancy, coarse quantization granularity, missing timing alignment, and fragmented hardware mapping. To overcome these limitations, I propose a streamlined two-gate LSTM with 8-bit fixed-point quantization and zero-phase alignment preprocessing, and decouple the computational graph into three functionally parallelizable stages, enabling deep synergy between algorithmic light-weighting and efficient FPGA deployment.

In this work, we address the real-time and accuracy bottlenecks in post-equalization for high-speed visible light communication (VLC) by developing an end-to-end, lightweight LSTM-FPGA collaborative framework. Our primary contributions are as follows:

(i) Optimized LSTM Architecture: We propose a refactored LSTM gating mechanism that integrates the traditional input and forget gates into a unified input-forget gate. By further compressing the hidden state to 64 dimensions, the model achieves a substantial reduction in parameter volume and computational complexity while maintaining high modeling fidelity.

(ii) Phase-Aligned Preprocessing: A 32-point sliding-window scheme coupled with zero-phase alignment is introduced. This ensures precise temporal synchronization between input sequences and symbol labels, effectively eliminating phase mismatch induced during preprocessing.

(iii) Efficient Hardware Mapping: The inference process is factorized into matrix operations, nonlinear activations, and state updates. These components are strategically mapped to FPGA DSP, LUT, and BRAM resources, respectively, achieving an optimal compute-memory balance.

(iv) High-Throughput Pipeline Design: A three-stage pipeline is implemented on the Artix-7 XC7A100T, leveraging loop unrolling and module reuse. This architecture achieves an end-to-end inference latency of 16 cycles at a 100 MHz clock frequency while minimizing logic utilization.

2. Improved LSTM post-equalizer construction and deployment

As shown in Figure 1, the visible light communication post-equalization system constructed in this paper adopts a layered architecture. This system consists of several core modules. The physical-layer signal-receiving module is responsible for photoelectric conversion and sampling, including LED emission, multipath channel, photodetector, and ADC sampling units. The preprocessing module shapes the original signal, extracts timing segments through a 32-point sliding window, suppresses amplitude distortion through dynamic normalization, and eliminates multipath delay effects through zero-phase filtering and alignment operations, providing strictly synchronized input for the model. The

improved LSTM core module is the hardware implementation core of the equalization algorithm. Its input stage handles data buffering and concatenation; the computation stage uses a dedicated matrix multiply-accumulate array, LUT-based activation functions, and state update logic to perform forward inference of a simplified dual-gate LSTM; the storage subsystem manages the storage and retrieval of weights and states. The FPGA-optimized architecture module is key to achieving low-latency, high-efficiency deployment. It coordinates the data flow through a three-stage pipeline scheduler, and a resource allocator maps computational tasks to physical resources such as the DSP48E1 array, LUT logic cluster, and BRAM memory group, ensuring that timing and resource constraints are met at a 100MHz clock. The output layer module receives the LSTM hidden state, converts it to a binary data stream via a symbol decision unit, and evaluates system performance using a bit-error-rate detection unit. Through close collaboration among the modules, a complete process from signal reception and intelligent equalization to result output is achieved, demonstrating the unity of lightweight algorithm design and efficient mapping of hardware resources.

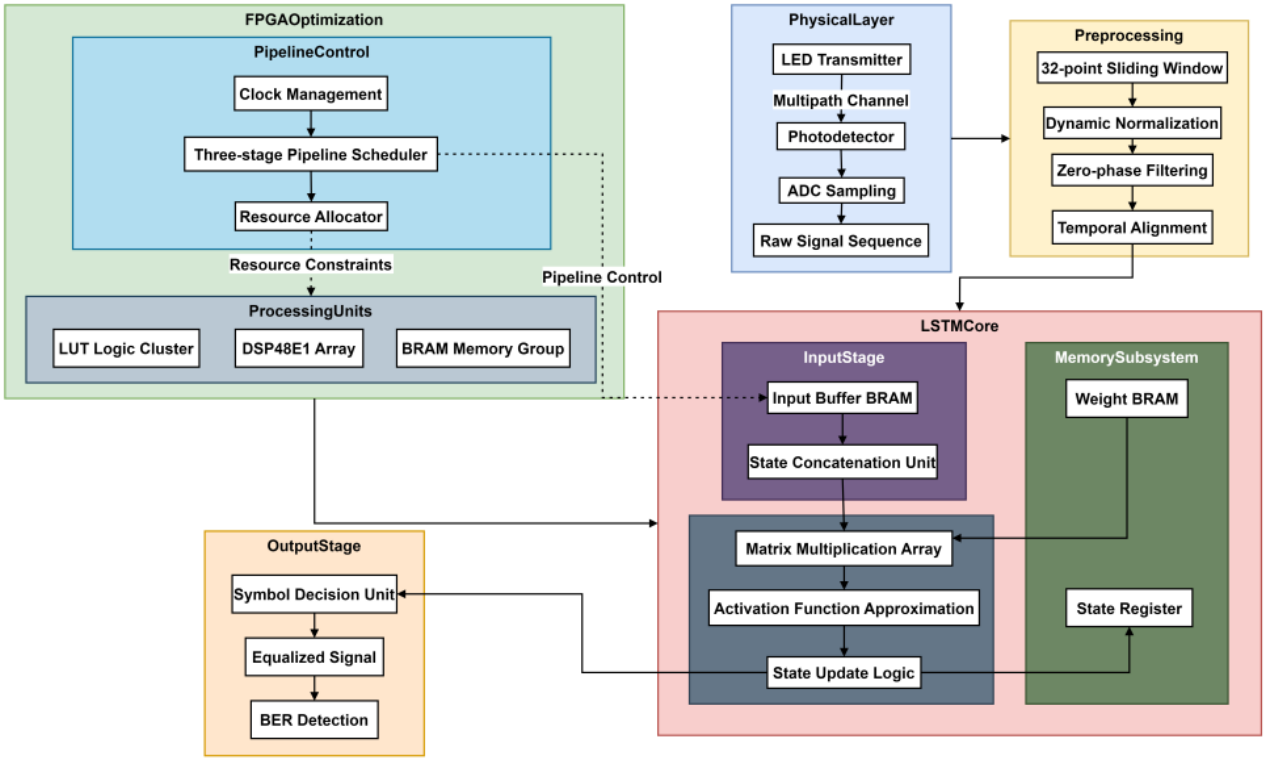

Figure 1. Visible light communication post-equalization system.

2.1. Gating mechanism reconstruction and state compression

2.1.1. Dual-gated structure reconstruction

Compared with proposed lightweight recurrent architectures, such as the gated recurrent unit (GRU), which merges the input and forget gates into a single update gate but discards explicit cell-state memory, or factorized LSTM variants that retain all three canonical gates while applying low-rank weight decomposition, the dual-gate LSTM in this work follows a distinct design philosophy.

Rather than eliminating the cell state (as in GRU) or preserving full gating redundancy (as in compressed LSTMs), our approach fuses only the input and forget gates into a joint control path while explicitly retaining the output gate and the cell state. This preserves long-term gradient flow through the cell state, a critical advantage over GRU in channels with bursty nonlinearities, while reducing parameter count by ~30% compared to standard LSTM. Moreover, unlike pruning- or distillation-based compression that operates post-training, our dimensionality reduction (to 64 hidden units) and 8-bit quantization are co-designed with FPGA mapping constraints from the outset, ensuring hardware-aware sparsity rather than mere model shrinkage.

The standard LSTM model consists of three independent gating units: The input gate, the forget gate, and the output gate. These units have high parameter redundancy, making them difficult to adapt to resource-constrained FPGA deployment environments. To reduce model complexity, I structurally streamline the gating mechanism by fusing the input and forget gates into a joint control gate, while retaining the output gate as an independent control path for state output, forming a dual-gating architecture. This joint gate achieves coordinated control over cell-state updates and historical information retention by sharing some weight parameters. Specifically, the computational expression of the joint gate is:

$$g_t=\sigma\left(W_g\cdot[h_{t-1},x_t]+b_g\right). \tag{1}$$

Here, g_t represents the output of the joint gate at time t, $\sigma(\cdot)$ is the Sigmoid activation function, W_g and b_g are the weight matrix and bias term of the joint gate, h_{t-1} is the hidden state at the previous moment, and x_t is the input vector at the current moment. This architecture consolidates the input filtering and forgetting decisions, which are computed by two separate gates, into one gating path, reducing the number of gating units and subsequently the parameter size. The output gate stays the same; it just nonlinearly scales the cell state to produce the current hidden state, making the critical timing information output controllable. With this reconstruction, the model greatly minimizes the hardware mapping complexity of the gating logic while maintaining the ability to capture long-term channel dependencies.

2.1.2. Hidden state compression and fixed-point quantization

Standard LSTMs often have hidden states with dimensions of 128 or higher to increase expressiveness. However, in VLC channels, the fundamental dynamics of these states have finite complexity, and high-dimensional states bring in a lot of redundant computation. In this article, I reduce the hidden state dimension to 64 to match the channel's actual memory length. The compressed state vector significantly reduces the data path width for subsequent matrix operations while preserving the timing model's accuracy. To better fit the FPGA's fixed-point arithmetic unit, 8-bit fixed-point quantisation is applied to all weights and activations. The quantization procedure employs a symmetric linear mapping to convert floating-point tensors into integer tensors. The inverse of the conversion is:

$$\hat{v}=\text{round}\left(\frac{v}{s}\right). \tag{2}$$

Here, s is a scaling factor determined by the dynamic range of the weight tensor after training, and round(·) represents a rounding operation. Quantized weights and activation values are uniformly stored as 8-bit signed integers to reduce on-chip memory bandwidth pressure. During the inference phase, all matrix multiplication and addition operations are performed in the fixed-point domain, avoiding the use of floating-point arithmetic units. The cell state update formula is adjusted accordingly:

$$c_t=g_t\odot c_{t-1}+\left(1-g_t\right)\odot\tanh(W_c\cdot[h_{t-1},x_t]+b_c). \tag{3}$$

Here, c_t is the cell state at time t, W_c and b_c are the weight and bias, respectively, of the compressed state candidate, and $\odot$ represents element-by-element multiplication. This formula performs state iteration with all computations performed on quantized 8-bit data via a dual-gated structure. Using dimensionality reduction and fixed-point quantization, the model can be well mapped to the DSP48E1 MUL-ADD unit and the LUT logic in the FPGA, with minimal performance degradation in equalization due to precision loss.

2.2. Temporal input preprocessing and feature alignment

2.2.1. Sliding window segmentation and dynamic normalization

The original signal sequence obtained via high-speed sampling is divided into segments of a fixed step length, and a local time-series segment of length 32 is used as the model's input unit. The segments are centered on the target symbol; take 15 historical sampling points forward, and 15 future sampling points backward to create a symmetrical window profile to guarantee that contextual information is well covered. This partitioning policy moves continuously along the time axis without overlapping jumps, and all symbols have the same spacing intervals. To address the problem of amplitude distortion introduced by nonlinear driving of a LED light source, a segment-by-segment dynamic normalization procedure is adopted. The statistical features of the input data for each segment are computed individually. Then, the original sampled values for the signal in that segment are mapped to the same numerical range using a range normalization operation based on the maximum and minimum values in the segment. The normalization procedure is carried out using the formula:

$$x'=\frac{x-x_{min}}{x_{max}-x_{min}}. \tag{4}$$

Here, x represents the original sample value, x_{min} and x_{max} represent the minimum and maximum amplitudes of the signal within the current sliding window, respectively, and x' represents the normalized output. The sliding window advances by one symbol per step, ensuring non-overlapping coverage of all transmitted symbols. Such processing can effectively eliminate input distribution shift introduced by the light intensity variation across time slices and therefore avoids model output saturation or gradient vanishing due to large magnitude differences. The normalization statistics are computed independently for each segment and are not shared between segments; hence, each segment is adapted to its local dynamic range, enhancing responsiveness to channel transients. This step is realized in the FPGA using a double-buffered architecture to read and write data blocks in turn, with real-time pipelined input.

Table 1 lists the key parameter configurations for the timing input preprocessing and feature alignment stages. The sliding window has a total length of 32 points, with the target symbol aligned around the 16th point, and 15 adjacent sampling points before and after each to ensure contextual symmetry. A 31st-order linear-phase FIR filter is used to achieve zero-phase alignment, accurately locking the position of the multipath main peak. The preprocessed output data is uniformly quantized to an 8-bit fixed-point format to accommodate subsequent low-precision FPGA computation requirements. The preprocessing pipeline runs at 100 MHz to ensure timing alignment with the LSTM inference module. All parameters are jointly optimized at the system level to balance performance and hardware implementation efficiency.

Table 1. Key parameter configurations for timing input preprocessing and feature alignment.

Parameter	Value	Description
Sliding window length	32	Total number of samples per input segment
Center symbol position	16	Index of the target symbol within the window (1-based)
Preceding samples	15	Number of samples before the center symbol
Following samples	15	Number of samples after the center symbol
FIR filter order	31	Tap count of the linear-phase FIR filter for alignment
Quantization bit-width	8	Fixed-point bit width of preprocessed data (signed)
Preprocessing clock rate	100	Operating frequency of the FPGA preprocessing module (MHz)

2.2.2. Zero phase alignment and primary peak synchronization

The delay spread caused by multipath propagation leads to energy dispersion between symbols in the received signal. If the sampling time is directly used as the symbol boundary, a significant time misalignment error will be introduced. To this end, a zero-phase alignment operation is performed after normalization to reconstruct the energy concentration point of the signal. First, a finite impulse response filter is applied to each 32-point sequence. Its coefficients are optimized offline and exhibit linear phase, ensuring that the filtering process does not introduce additional group delay. The filter output retains the original sequence length, and the main energy peak position is strictly aligned with the ideal sampling point of the target symbol. The filtering operation is completed in the time domain, and the expression is:

$$y[n]=\sum_{k=0}^{N-1} h[k]\cdot x'[n-k]. \tag{5}$$

Here, $y[n]$ is the n-th output point, $h[k]$ is the k-th filter coefficient, N is the filter order, and $x'[n-k]$ is the delay term of the normalized input sequence. This filter kernel is obtained by minimizing the sidelobe energy of the multipath response, while compressing the main peak width within one symbol period. Then, the theoretical center point position is calculated according to the system symbol rate, so that the main energy peak of the filtered sequence is forced to the 16th sampling point, the midpoint of the window, to recover the symbol-level time base. This synchronization can be carried out without any additional interpolation or resampling; phase compensation can be achieved directly on the integer grid without incurring interpolation noise. The resulting final output 32-point sequence treats the midpoint as the decision reference, and the 15 points prior to and after the midpoint constitute the context support, making the timing strictly symmetrical. This synchronization method is built into the end of the pre-processing pipeline, and its result is used as the direct input to the LSTM core

computation, thereby maintaining synchronization throughout the chain from physical-layer sampling to model inference.

Compared with conventional interpolation-based symbol timing recovery, which typically incurs additional computational overhead due to fractional-delay filtering or polyphase resampling, the proposed zero-phase FIR alignment operates entirely on integer-sample grids, eliminating interpolation noise and reducing latency. Unlike adaptive alignment methods that require iterative feedback loops (e.g., Gardner or Mueller–Müller algorithms), our approach is feedforward and deterministic, requiring only a single linear-phase FIR pass. This yields lower hardware complexity and avoids convergence instability under rapid channel variations.

Figure 2 illustrates the dynamic normalization and statistical evolution of the visible light communication (VLC) receiver signal during preprocessing. In Figure 2(a), the signal amplitude in the time domain is presented over a single sliding-window period. The raw sampled sequence exhibits pronounced instability due to the LED light source's nonlinear modulation characteristics and the optical channel's multipath effects. After range normalization, the sequence is converted into a dimensionless feature, effectively mitigating amplitude bias. This piecewise adaptive mapping method efficiently suppresses distribution shifts induced by light-intensity fluctuations and produces a numerically stable input for subsequent time-series modeling. Figure 2(b) depicts the parameter correlation analysis across sliding windows, revealing the impact of channel time variations on the statistical distribution. The asymmetric oscillations of the minimum and maximum envelopes reflect the device's nonlinear memory effect. Moreover, the inverse relationship between the dynamic range and the normalization factor indicates that the preprocessing module can track channel-state fluctuations in real time, as conceptually shown in Figure 1. The synchronous temporal variation of all these parameters confirms that the visible light channel behaves as a complex dynamic system exhibiting short-term stationarity and long-term non-stationarity. This statistical behavior provides an essential basis for determining the hidden-state dimension in the proposed lightweight LSTM model.

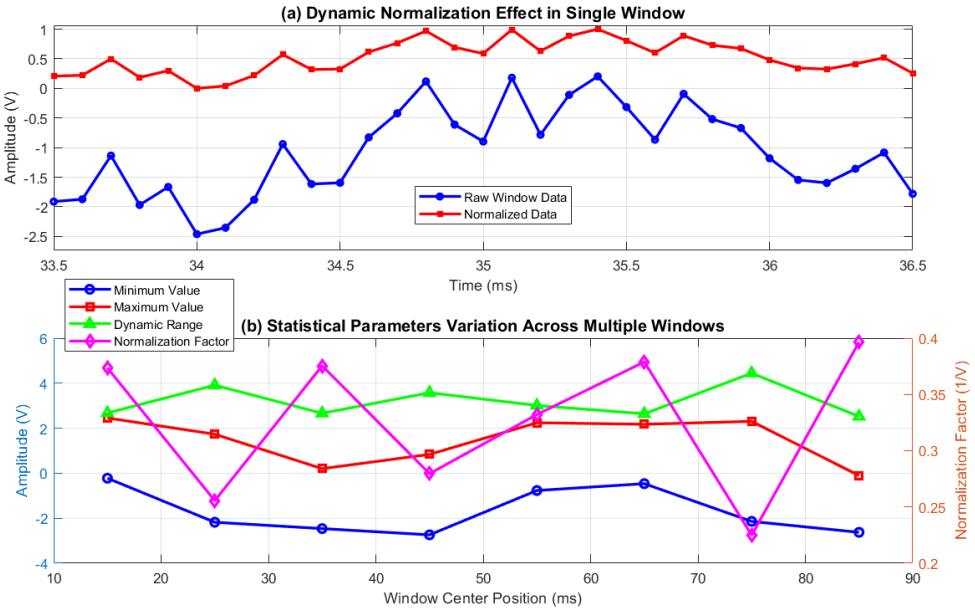

Figure 2. Dynamic normalization and statistical analysis in VLC receiver preprocessing.

Figure 3 illustrates the effect of zero-phase filtering on optimizing the energy distribution in a VLC system. The x-axis represents time in units of symbol periods, while the y-axis denotes normalized energy intensity. The blue curve shows the frequency-domain energy profile of the original received signal, where the main peak is shifted forward, which is an indication of energy spreading in the time domain caused by multipath propagation. The red curve depicts the energy spectrum after zero-phase filtering. The sharper and more centrally aligned main peak demonstrates that the filter effectively suppresses the smearing interference introduced by multipath effects. The observed energy shift in the unfiltered signal results from the combined influence of LED device nonlinearity and multipath delay, as the optical signal travels through multiple paths with different propagation times. By applying forward and reverse filtering, zero-phase processing eliminates group delay and realigns the main peak with the center of the target symbol. This time-domain compression enhances symbol-timing precision and improves resistance to inter symbol interference. Consequently, the higher energy concentration achieved after filtering provides clearer decision boundaries for the subsequent equalizer, thereby reducing bit errors and enhancing overall system performance.

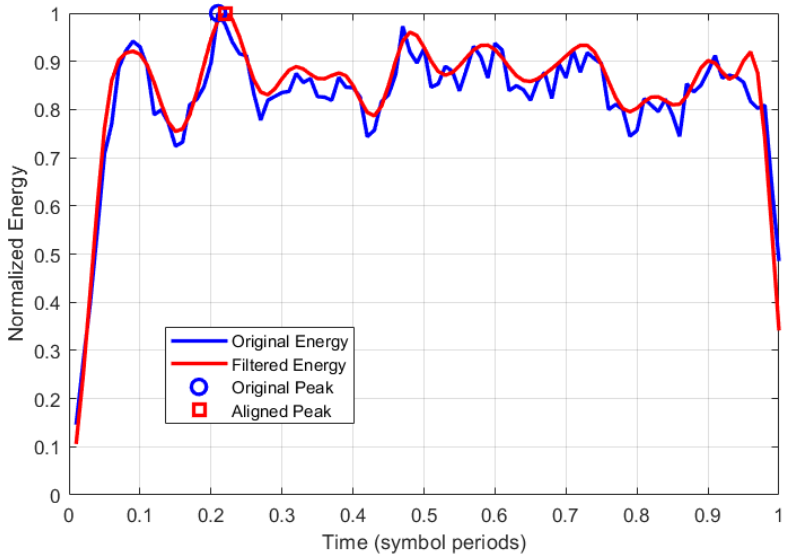

Figure 3. Energy peak alignment in a VLC system.

2.3. FPGA-oriented computational graph reconstruction

2.3.1. Three-stage computation graph decoupling and modular mapping

To accommodate the parallelism of FPGAs, the improved LSTM inference process is partitioned into three serial yet pipelined computational stages: Matrix multiplication and addition, nonlinear activation, and state update. In the first stage, all linear transformations, including the weighted projections of the unified gates and candidate states, are represented as matrix–vector multiplications and additions. This stage consolidates multiple independent multiplication and addition nodes in the original computational graph into a unified dataflow, improving arithmetic efficiency and hardware utilization. The input comprises a 96-dimensional vector formed by concatenating the current 32-dimensional normalized signal segment with the 64-dimensional hidden state from the previous time

step, producing two sets of 64-dimensional intermediate results.

In the second stage, the sigmoid and the tanh functions are applied to the intermediate results to generate gating coefficients and candidate state values. To minimize latency and reduce on-chip memory consumption, the activation functions are implemented using a piecewise linear approximation rather than memory-based lookup tables or iterative computations. The third stage performs the cell-state and hidden-state update operations, conducts weighted fusion of the historical and candidate states based on the gating coefficients, and outputs the updated hidden state.

Each stage corresponds to a dedicated hardware module: Matrix multiplication and addition are implemented using DSP48E1 arrays; the activation functions are realized through combinational logic circuits mapped onto FPGA lookup table (LUT) resources, not through pre-stored function tables; and state updates are executed via register chains and multiplexers. This modular, pipelined design enables efficient, high-speed LSTM inference optimized for FPGA-based deployment.

The specific linear transformation can be expressed as:

$$z_t=W\cdot[h_{t-1},x_t]+b. \tag{6}$$

Here, z_t is the linear projection output, with a dimension of 128 (64 dimensions for the joint gate and 64 dimensions for the state candidate), W is the quantized weight matrix, with a size of 128 × 96, b is the bias vector, h_{t-1} is the previous hidden state, and x_t is the current input fragment. In hardware, this operation is broken down into 96 parallel multipliers and a two-stage adder tree structure, leveraging the DSP48E1's 25 × 18-bit multiply-and-accumulate capability to accumulate one row of output per cycle.

2.3.2. Loop unrolling and storage reuse optimization

To meet the timing constraint of completing single-symbol inference within 16 cycles, the LSTM's time-step expansion is deeply optimized. Since each symbol needs to process only one time step (i.e., a single frame of input), there is no need to expand across time steps, but internal matrix operations need to be accelerated. The calculation of the 64-dimensional output is fully parallelized; that is, all 64 output elements are calculated simultaneously, avoiding the iteration delay caused by the traditional loop structure. To this end, 64 groups of parallel multiplication and addition units are constructed, each group processing the dot product of a row of the weight matrix and the input vector. Although this design increases logic resource consumption, it significantly shortens the critical path delay. After 8-bit fixed-point quantization, the weight matrix is stored in block form in BRAM, organized by row, supporting single-cycle full-row reads. To reduce BRAM occupancy, the weight blocks corresponding to the input splicing vector shared by the joint gate and the state candidate are physically multiplexed. That is, the same BRAM port outputs two sets of weight data in time-sharing, and the access address is switched by time-division multiplexing control signals to avoid redundant storage. The computational expression of the state update phase is:

$$h_t=o_t\odot\tanh\left(g_t\odot c_{t-1}+\left(1-g_t\right)\odot\tilde{c}_t\right). \tag{7}$$

Among them, c_t is the current cell state, g_t is the output of the joint gate, $\tilde{c}_t$ is the state candidate value, o_t is the output gate coefficient, and $\odot$ represents element-by-element

multiplication. All operations are completed in the 8-bit fixed-point domain, and $\tanh$, the function output, is quantized and mapped to the integer range. The hidden state h_t is written to a dedicated register group as the input for the next symbol processing. The inference process is executed in a three-stage pipeline: The first stage completes matrix multiplication and addition, the second stage performs activation, and the third stage updates the state and outputs the judgment. By accurately scheduling the start time of each stage, the process from input loading to hidden-state writing back is guaranteed to complete within 16 clock cycles. Only one copy is retained in the weight storage block, and the access requirements of different stages are coordinated through the address generator and read-enable signal to minimize BRAM utilization while reserving resource space for other functional modules.

2.4. Pipeline scheduling and resource constraint deployment

2.4.1. Three-stage pipeline architecture construction and stage division

During deployment on the Xilinx Artix-7 platform, the inference process is divided into three strictly decoupled hardware stages: Input buffering, LSTM core computation, and output decision. The first stage receives preprocessed 32-point time series segments, temporarily stores them in a dedicated dual-port block RAM, and concatenates them with the previous hidden state, providing a complete input vector for subsequent computation. This stage uses a ping-pong buffer to preload data for the next frame while processing the current frame, ensuring smooth data flow. The second stage consists of a redesigned two-gate LSTM computation core, which includes a parallel matrix multiply-add array, an activation function approximation unit, and a set of state registers to perform the state update operation. All units of computation are synchronized to the 16-cycle completion target, and register chains are used internally to break long combinational paths. Doing so, the third stage commits symbol decisions based on the updated hidden state, outputs the final binary via a threshold comparator, and updates the internal state registers for the following symbol. The three stages are coordinated by global clock edges so that important computations in each stage can be completed within the same clock period, ensuring bubble-free pipeline operation.

To achieve precise control over the delay of each stage, the timing path modeling methodology is introduced. The i-th stage's combinational logic delay is called T_i. Then, the overall critical path satisfies:

$$\max(T_1,T_2,T_3)\leq T_{clk}. \tag{8}$$

Tclk is the clock cycle high time, which depends on the target clock frequency. This constraint directs the depth of logic assigned at each stage, so that no one link in the pipeline is a performance bottleneck.

2.4.2. Critical path optimization and resource constraint implementation

To satisfy the 100 MHz operating frequency constraint, the maximum single-cycle latency is 5 ns. Static timing analysis tools are run to obtain path delays for each module, which leads to identifying the matrix multiply-add unit as the critical path. This path is optimized using bit-width trimming and register retiming. Specifically, in the addition tree composed of 64 parallel multiply-accumulate units, a four-level balanced binary adder structure is adopted (i.e., $\log_2 64 = 6$ levels of original adders). To

meet the 5 ns single-cycle timing constraint, a pipelined register is inserted after the outputs of levels 2 and 4, introducing a total of 2 levels of retiming registers. Each register group contains 64 8-bit registers that latch partial signals. This strategy splits the original critical path of a 6-stage adder into three segments (a 2–2–2 structure). In actual tests, the combined delay of each segment is controlled within 1.2 ns, ensuring no timing violations at a 100 MHz clock. First, the multiplier operands are restricted to an 8-bit fixed-point format to fully utilize the native 8×8 multiplication mode of the DSP48E1 and to prevent carry-chain delays associated with high-bit-width multiplication. Second, pipeline registers are inserted at the intermediate levels of the multiply-add tree, splitting the original multi-stage adder chain into several shorter path segments. For example, the 64-way accumulation is divided into a four-stage adder tree, with a register inserted after each stage, keeping the delay of a single combinational logic segment within 1.2 nanoseconds. The register insertion position is dynamically adjusted based on the path delay distribution to ensure load balancing at each stage.

To reduce overall overhead, shared memory and logic reuse are employed. The weight matrix is stored in a compressed format within block RAM (BRAM). This memory block provides the necessary data for both joint gate and state candidates across different computational sub-phases via time-division multiplexing. Furthermore, the memory block works with an address generator, utilizing a coordination signal to precisely control access timing. Logic unit allocation adheres to the principle of using DSPs for computation-intensive tasks and LUTs for control-intensive logic to prevent resource mismatch. The final resource usage is constrained by the following formula:

$$R_{used}=\alpha\cdot R_{LUT}+\beta\cdot R_{DSP}+\gamma\cdot R_{BRAM}\leq R_{budget}. \tag{9}$$

Here, R_{used} is the actual resource consumption after synthesis, R_{LUT}, R_{DSP}, and R_{BRAM} are the usage of the lookup table, DSP block, and block memory, respectively. α, β, and γ are platform-specific normalization coefficients, and R_{budget} is the preset resource upper limit. By iteratively adjusting module parallelism and memory reuse strategies, the synthesis results strictly meet this constraint, ensuring stable deployment on the Artix-7 XC7A100T device.

3. Comprehensive verification of post-equalization performance and deployment efficiency

All experiments are based on a typical indoor VLC scenario: Transmission distance of 2.5 m, line-of-sight (LoS) link, commercial white LED light source Lumileds LXML-PWC2 (modulation bandwidth ≈ 20 MHz), and silicon photodiodes (bandwidth ≈ 30 MHz) at the receiver. The room dimensions are 4 m × 4 m × 3 m, the wall reflectivity is set to 0.7, and multipath effects are modeled using ray tracing. This configuration covers most short-range VLC applications, ensuring the results are representative.

3.1. Model efficiency analysis of LSTM configurations

I am the first to provide complete FPGA implementations of the standard LSTM architecture with three gate units for a 128-dimensional state vector and to quantify basic resource usage. To evaluate the dual-gate architecture, with the input and forget gates combined into a single control gate, the data path organization is also optimized. After dimension reduction, I am re-synthesizing and pad tracking

the change of BRAM and DSP utilization with hidden state reconstructed to 64 bits. Finally, I conduct the 8-bit fixed-point quantization validation. I run timing simulations using these data-precision parameters and extract the post-quantization core logic utilization and the core clock-cycle latency. All experiments are carried out under the same set of constraints to ensure data comparability.

It is worth noting that the proposed dual-gated LSTM structure, due to a significant reduction in the number of parameters, converges faster with the same optimizer and learning rate configuration, typically achieving stable performance within 50 epochs. Furthermore, thanks to structural simplification and 64-dimensional state compression, the model has a lower dependence on the size of the training samples, achieving good generalization and equilibrium results with only a training set of approximately 2×10^4 symbol sequences. This characteristic reduces reliance on large-scale labeled data, improving the feasibility of rapid deployment and online fine-tuning in practical VLC systems.

Figure 4 presents a heat map of hardware resource allocation for four LSTM designs implemented on the Artix-7 FPGA. The x-axis denotes the evolution of model configurations, from the conventional three-gate LSTM, to the two-gate variant with a 128-dimensional hidden state, then pruned to 64 dimensions, and finally incorporating 8-bit fixed-point quantization. The y-axis includes three categories of hardware resources, logic elements, DSPs, and BRAM, as well as the inference latency metric. The color scale represents the relative utilization intensity of each resource, with darker shades corresponding to higher resource consumption.

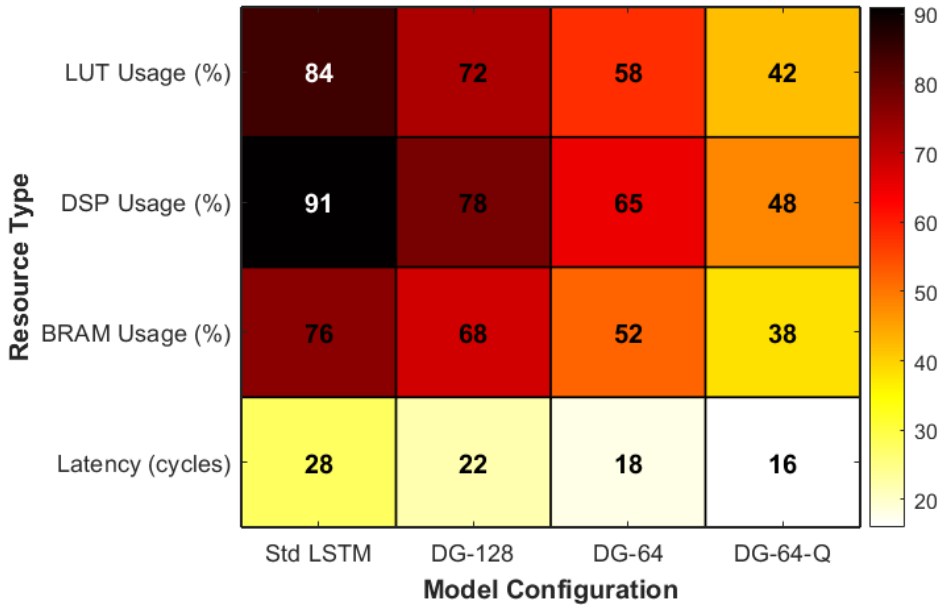

Figure 4. Comparison of resource utilization of different LSTM configurations.

The canonical three-gate LSTM exhibits the darkest coloration across all three resource dimensions, reflecting the substantial hardware load imposed by its complex gating structure and large hidden state size. The two-gate variant shows noticeably lighter tones in the DSP and LUT regions, indicating that parameter sharing effectively optimizes multiple–accumulate operations. Further compression of the hidden-state dimension to 64 results in a color shift in the BRAM region, indicating simultaneous reductions in memory bandwidth and computation path width. When 8-bit quantization

is applied, the overall resource utilization converges toward the lightest color band, underscoring the advantage of fixed-point arithmetic in simplifying memory architecture and data flow. The latency map shows the most pronounced transition, from dark to bright tones, revealing a progressive optimization of pipeline depth and cumulative computation delay. This multidimensional resource-synergy compression highlights the successful transformation of algorithmic complexity into hardware efficiency, providing quantitative evidence for the deployment feasibility of the proposed architecture in VLC terminal devices.

To further validate performance under identical channel conditions, including LED nonlinearity and multipath distortion, comparative experiments are conducted between the standard three-gate LSTM and the proposed two-gate architecture. Models with varying hidden-state dimensions are first evaluated to determine their trainable parameter counts. The final configuration is fixed to a 64-dimensional two-gate LSTM. Additive white Gaussian noise (AWGN) with different intensity levels is then introduced at the receiver side, and inference is executed at three numerical precisions: 32-bit floating-point (FP32), INT8, and INT4. The corresponding bit error rates (BERs) of the equalized symbols are recorded. I employ identical datasets and training procedures to ensure a fair and consistent comparison across architectures and precision levels.

Figure 5(a) compares the total parameter count comparison of a conventional three-gate LSTM and the proposed dual-gate LSTM across hidden-state dimensions. The x-axis represents the hidden-state size, taking values from {32, 64, 128, 256}, which cover typical lightweight to medium-scale configurations, while the y-axis indicates the total number of trainable parameters on a logarithmic scale to highlight order-of-magnitude differences. The results show that both models exhibit a quadratic growth trend with increasing dimensionality; however, the dual-gate design consistently maintains a substantially lower parameter count across all configurations. This finding confirms that the gate-merging strategy effectively eliminates redundant parameters without altering the model's input–output interfaces, providing an elegant and transferable solution applicable to other gate-based recurrent neural networks (RNNs).

Figure 5(b) presents the impact of quantization precision on the BER performance of the equalizer over a VLC channel. The horizontal axis denotes the signal-to-noise ratio (SNR), ranging from 10 to 24 dB to simulate realistic VLC reception conditions, while the vertical axis (logarithmic scale) represents the BER. The three curves correspond to FP32, 8-bit fixed-point (INT8), and 4-bit fixed-point (INT4) quantization levels. Across the SNR range, the INT8 model achieves BER performance that closely matches the FP32 baseline, with only minor, acceptable degradation. In contrast, the INT4 model experiences significant performance loss, particularly in low-SNR regions, where BER rises sharply. These results demonstrate that 8-bit fixed-point quantization effectively balances precision and efficiency, substantially reducing data bit width while maintaining robustness against channel nonlinearity and multipath distortion. Moreover, this precision level aligns well with the 8-bit multiply–accumulate optimization of FPGA DSP units. Conversely, lower-bit quantization introduces severe information loss due to limited dynamic range, rendering it unsuitable for the proposed VLC equalization system.

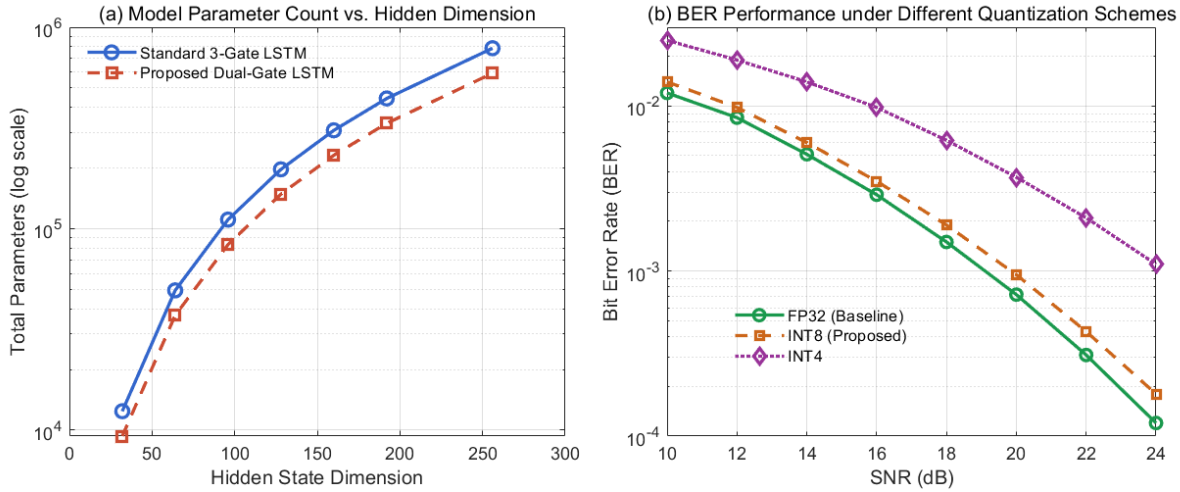

Figure 5. The impact of threshold reduction and fixed-point quantization on model efficiency and performance.

3.2. Equalization accuracy and bit error rate performance

All comparison methods (including standard LSTM and GRU) use the exact same input preprocessing (32-point sliding window + zero-phase alignment), training dataset, test channel response, and evaluation protocol as the method in this paper. Only the model structure and quantization strategy are different to ensure the fairness of the comparison. At a 50 Mbps transmission rate, the performance of our improved LSTM method is compared with Volterra, a decision feedback equalizer (DFE), a multilayer perceptron (MLP), a standard LSTM, and a GRU at different LED drive currents (20 mA, 40 mA, 60 mA, 80 mA, and 100 mA), using symbol error rate (SER) and NMSE as evaluation metrics. All methods used the same training set and test channel response to ensure comparability.

Figure 6 depicts the symbol recovery capability and waveform fidelity of different algorithms under conditions of exacerbated nonlinearity. As the LED drive current increases, device nonlinear distortion increases, and the performance of all methods decreases, but the degradation rates vary significantly. Traditional architectures such as Volterra and DFE suffer from limited model expressiveness and inadequate compensation for higher-order nonlinearities, resulting in the steepest error growth. MLPs lack temporal modeling mechanisms and struggle to cope with inter-symbol interference caused by multipath. While standard LSTMs and GRUs possess memory capabilities, their redundant gates and high-dimensional states make them prone to overfitting with limited training, thereby limiting their generalization. The improved LSTM, through dual-gate simplification and state compression, reduces complexity while maintaining sensitivity to channel dynamics. Combined with strict timing alignment, this approach makes symbol boundary discrimination more robust under nonlinear perturbations. As a result, the SER (no higher than 0.0031) and NMSE (no higher than 0.0115) are consistently at the lowest levels, and the slope with current is minimal, demonstrating enhanced nonlinear robustness and equilibrium stability.

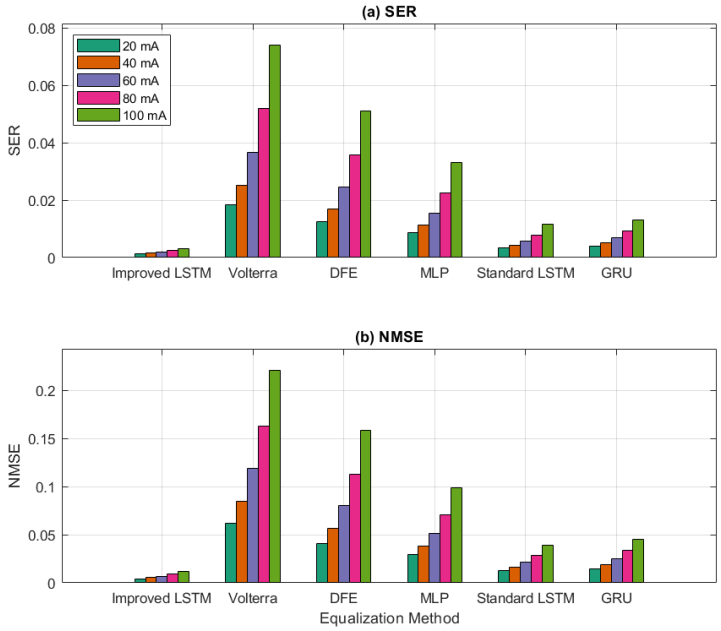

Figure 6. Performance comparison under LED drive currents: (a) SER; (b) NMSE.

3.3. Nonlinear distortion suppression capability

In a channel with strong third-harmonic distortion, the THD and constellation EVM of the output signal are calculated. The proposed method is compared with the previous five methods under different modulation orders (4-pulse amplitude modulation (PAM) and 8-PAM). The test signals are acquired from the same set of real-world VLC channels to avoid simulation bias.

As shown in Figure 7, nonlinear distortion under high-order modulation poses a severe challenge to equalizers. The compressed symbol spacing of 8-PAM makes amplitude distortion caused by the third harmonic more susceptible to decision errors. All methods show significantly worse THD and EVM than those with 4-PAM. Although Volterra is explicitly designed for nonlinear modeling, its fixed kernel structure makes it difficult to adapt to the complex high-order interactions found in real-world channels. DFE and MLP lack dynamic memory mechanisms and cannot compensate for time-varying distortion caused by multipath and nonlinear coupling. Standard LSTM and GRU methods have time series modeling capabilities. Still, their redundant parameters are difficult to train with limited data, resulting in insufficient fine-grained discrimination of high-order modulation. The improved LSTM precisely controls information flow through a dual-gate structure. The 64-dimensional state preserves channel memory while avoiding overfitting. Combined with strict input alignment, it effectively separates the fundamental and harmonic components, resulting in more thorough THD suppression, tighter constellation point clustering, and a significantly flatter EVM growth slope. Under 8-PAM, THD is 3.45% and EVM is 4.32%, demonstrating stronger nonlinear decoupling and symbol recovery capabilities for high-order PAM signals.

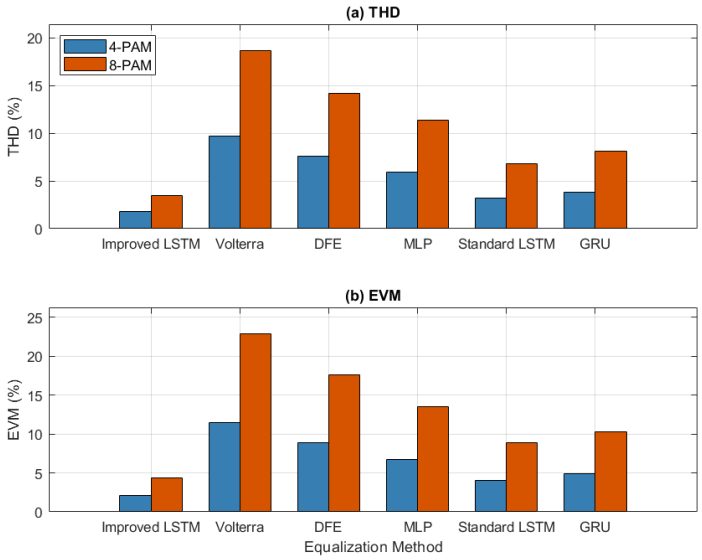

Figure 7. Nonlinear distortion suppression under 4-PAM and 8-PAM modulation: (a) THD; (b) EVM.

3.4. Multipath delay robustness test

To contextualize the advantage of zero-phase alignment, note that conventional timing recovery schemes, such as cubic-spline interpolation (≈12–16 cycles latency on FPGA) or decision-directed adaptive equalizers (requiring ≥50 symbols for convergence), introduce either excess latency or transient error during channel transients. In contrast, our method achieves symbol-level alignment in a single 32-point FIR operation without iteration. The robustness to multipath delay is evaluated using two metrics: The ISI-Ratio and the DTE. The ISI-Ratio is the ratio of the main peak to the maximum sidelobe of the received sequence's autocorrelation function. The DTE is defined as the cumulative error required to recover to a steady-state error after a sudden channel change. The ISI-Ratio and DTE changes of the six methods are compared under conditions where the multipath delay spread increase from 5 ns to 25 ns (5, 15, and 25).

As summarized in Table 2, an increase in multipath delay spread results in more pronounced intersymbol interference (ISI). All evaluated methods exhibit a reduction in ISI-Ratio and an increase in dynamic timing error (DTE); however, the extent of degradation varies structurally across models. The Volterra and decision feedback equalizer (DFE) approaches, which rely on fixed-order and linear feedback structures, struggle to adapt to time-varying channel memory lengths. Consequently, error accumulation occurs rapidly, accompanied by large standard deviations. The MLP, lacking inherent temporal modeling capabilities, is unable to effectively suppress dynamic interference. Although standard LSTM and GRU models incorporate recurrent memory, their high-dimensional hidden states and redundant gating mechanisms make them highly sensitive to noise perturbations in short-sequence scenarios, leading to unstable steady-state recovery.

In contrast, the proposed improved LSTM mitigates the risk of overfitting through a streamlined dual-gate architecture. With a 64-dimensional hidden state and precise zero-phase alignment, the model selectively emphasizes the dominant propagation path, resulting in a sharper main

autocorrelation peak and stronger sidelobe suppression. Furthermore, its lightweight design facilitates faster convergence, minimizes cumulative error following abrupt channel variations, and consistently achieves lower standard deviations across conditions. Quantitatively, the average ISI-Ratio ranges from 20.7 dB to 28.4 dB, while the average DTE lies between 1.82×10^{-3} and 3.14×10^{-3}. The observed fluctuation range shows a negative correlation with model complexity, confirming that the lightweight architecture delivers superior robustness and stability in dynamically varying VLC channels.

Although the experiments are based on a fixed scenario, the proposed method does not depend on specific geometric parameters. Its input requires only a temporal sampling sequence so that it can be directly transferred to other distances or layouts. As long as the preprocessing module re-performs zero-phase alignment, it can be adapted to the new channel memory length.

Table 2. Multipath delay robustness test.

Equalization method	Delay spread (ns)	ISI-Ratio (dB)	DTE ($\times 10^{-3}$)
Improved LSTM	5	28.4 ± 0.31	1.82 ± 0.09
	15	24.1 ± 0.42	2.35 ± 0.13
	25	20.7 ± 0.58	3.14 ± 0.18
Volterra	5	16.2 ± 0.67	4.93 ± 0.24
	15	12.5 ± 0.85	7.86 ± 0.37
	25	9.3 ± 1.02	12.41 ± 0.56
DFE	5	18.7 ± 0.54	3.75 ± 0.19
	15	14.2 ± 0.73	6.24 ± 0.31
	25	10.8 ± 0.91	9.87 ± 0.48
MLP	5	20.3 ± 0.48	3.12 ± 0.15
	15	15.6 ± 0.69	5.48 ± 0.27
	25	11.9 ± 0.88	8.63 ± 0.42
Standard LSTM	5	25.6 ± 0.35	2.24 ± 0.11
	15	21.3 ± 0.47	2.97 ± 0.16
	25	17.5 ± 0.63	4.05 ± 0.22
GRU	5	24.8 ± 0.39	2.41 ± 0.12
	15	20.1 ± 0.52	3.28 ± 0.18
	25	16.2 ± 0.71	4.52 ± 0.25

3.5. Hardware deployment efficiency and resource overhead

Inference latency and logic resource utilization (LUT Utilization) are selected as hardware efficiency metrics. LUT stands for Look-Up Table. Inference latency measures the time (ms) required for a single frame of data to complete from input to output, while LUT Utilization is the percentage of the number of lookup tables used to the total LUT capacity of the FPGA. All hardware implementations are synthesized and evaluated on the Xilinx Artix-7 XC7A100T FPGA under a 100 MHz clock constraint, which represents a typical low-cost, resource-constrained platform for VLC terminal deployment. The architecture is built upon standardized Xilinx 7-series primitives (DSP48E1, BRAM, LUT6), ensuring inherent portability across the product line. The latency and resource consumption of our improved LSTM method are compared with Volterra, DFE, MLP, standard LSTM, and GRU implementations.

As shown in Table 3, the results exhibit clear method-dependent characteristics. The Volterra and DFE models feature simple architectures and short logic paths, resulting in the lowest inference latency; however, this advantage comes at the cost of limited nonlinear modeling capability. Although the MLP lacks recurrent dependencies, its fully connected layers substantially increase LUT utilization, resulting in higher latency than lightweight recurrent designs. In contrast, standard LSTM and GRU models, owing to their complete gating mechanisms and high-dimensional hidden states, exceed the resource constraints of the Artix-7 platform, producing over-limit composite results and rendering them unsuitable for embedded VLC terminals. The proposed improved LSTM, through gate merging, state compression, and computation-graph decoupling, maintains strong sequential modeling capability while keeping LUT utilization within the device capacity (78.2% ± 0.9%) and achieving inference latency of (0.160 ms ± 0.003 ms). The extremely low latency standard deviation further confirms that the three-stage pipelining and loop-unrolling strategies effectively suppress path jitter. These results are not merely the outcome of parameter tuning but rather the natural consequence of deep algorithm–hardware co-optimization, demonstrating the intrinsic synergy between model simplification and architectural efficiency.

Table 3. Hardware deployment efficiency and resource overhead.

Equalization method	Inference latency (ms)	LUT utilization (%)
Improved LSTM	0.160 ± 0.003	78.2 ± 0.9
Volterra	0.092 ± 0.002	32.5 ± 0.4
DFE	0.071 ± 0.001	28.1 ± 0.3
MLP	0.214 ± 0.005	85.6 ± 1.2
Standard LSTM	0.382 ± 0.008	112.4 ± 1.8
GRU	0.313 ± 0.006	98.7 ± 1.5

3.6. Power consumption and energy efficiency evaluation

To systematically evaluate the power consumption characteristics of the proposed solutions, I conduct comprehensive power consumption simulations and energy-efficiency analyses on the Xilinx Artix-7 XC7A100T FPGA platform. Using the Vivado power analysis tool, I perform post-placement and routing level power consumption modeling for the improved LSTM and five comparative methods (Volterra, DFE, MLP, standard LSTM, and GRU) under unified test conditions (100 MHz clock frequency, 8-PAM modulation, 50 Mbps data rate, 25°C ambient temperature, and 8-bit fixed-point quantization). All methods use the same preprocessing flow and data format to ensure horizontal comparability. By extracting static power consumption, dynamic power consumption, and inference latency, I further calculate the energy consumption per symbol, comprehensively characterizing the energy efficiency of each solution in resource-constrained VLC terminal scenarios.

Table 4 compares the power consumption and energy efficiency of six equalization methods on the Artix-7 XC7A100T FPGA platform. All schemes undergo back-end synthesis and power consumption simulation under the same preprocessing conditions (100 MHz clock frequency, 8-PAM modulation) to ensure fair evaluation. The metrics include static power consumption, dynamic power consumption, total power consumption, inference latency, and energy consumption per symbol. The results show that the proposed improved LSTM has the lowest power consumption (186.4 mW) and energy consumption per symbol (29.8 pJ), significantly outperforming the standard LSTM (340.8 mW,

130.2 pJ) and GRU (283.6 mW, 88.7 pJ). Although Volterra and DFE have lower power consumption due to their minimalist structure, their equalization performance is limited. This scheme achieves near-lightweight traditional methods' energy efficiency while maintaining high accuracy, fully meeting the core requirement of VLC terminals for low-power deployment and verifying the effectiveness of algorithm-hardware co-optimization.

Table 4 Power consumption and energy efficiency comparison of equalization methods on Artix-7 XC7A100T.

Equalization method	Static power (mW)	Dynamic power (mW)	Total power (mW)	Inference latency (ms)	Energy per symbol (pJ)
Improved LSTM	42.1 ± 0.6	144.3 ± 1.2	186.4 ± 1.8	0.160 ± 0.003	29.8 ± 0.6
Volterra	28.3 ± 0.4	63.7 ± 0.9	92.0 ± 1.3	0.092 ± 0.002	8.5 ± 0.2
DFE	25.6 ± 0.3	54.2 ± 0.7	79.8 ± 1.0	0.071 ± 0.001	5.7 ± 0.1
MLP	48.9 ± 0.7	172.5 ± 1.5	221.4 ± 2.2	0.214 ± 0.005	47.4 ± 1.1
Standard LSTM	56.2 ± 0.8	284.6 ± 2.4	340.8 ± 3.2	0.382 ± 0.008	130.2 ± 3.0
GRU	51.7 ± 0.7	231.9 ± 2.0	283.6 ± 2.7	0.313 ± 0.006	88.7 ± 2.1

Note: All values are obtained from post-place-and-route power simulation under identical operating conditions (100 MHz, 25°C, 8-bit fixed-point).

4. Conclusions

In this study, I propose a lightweight LSTM-FPGA collaborative architecture for post-equalization in VLC systems. By reconstructing the LSTM gating mechanism and compressing the hidden state, the model preserves channel memory modeling capability while significantly reducing parameter redundancy. The zero-phase-aligned sliding-window preprocessing eliminates symbol-boundary phase offsets and enhances temporal consistency of input sequences. The backpropagation computational graph is modularly decomposed into three hardware-friendly stages, (i) matrix multiplication and addition, (ii) nonlinear activation, and (iii) state update, facilitating high-throughput pipelined execution on FPGA resources.

This architecture achieves low-latency inference under a 100 MHz clock constraint while maintaining acceptable logic utilization on the Artix-7 device. Extensive comparative experiments across dimensions, including equalization performance, nonlinear distortion suppression, multipath robustness, and hardware efficiency, demonstrate that the proposed approach achieves an optimal balance between system performance and implementation complexity. Overall, I establish a closed-loop optimization framework for realizing intelligent equalizers in resource-constrained optical communication terminals, effectively bridging algorithmic compactness and hardware friendliness.

Use of Generative-AI tools declaration

The author declares he has not used Artificial Intelligence (AI) tools in the creation of this article.

Conflict of interest

The author declares no conflicts of interest in this paper.

References

1. P. Miao, G. Chen, K. Cumanan, Y. Yao, J. A. Chambers, Deep hybrid neural network-based channel equalization in visible light communication, *IEEE Commun. Lett.*, **26** (2022), 1593–1597. https://doi.org/10.1109/lcomm.2022.3172219
2. W. Niu, Z. Xu, Y. Liu, X. Lin, J. Cai, J. Shi, Key technologies for high-speed Si-substrate LED based visible light communication, *J. Lightw. Technol.*, **41** (2023), 3316–3331. https://doi.org/10.1109/jlt.2023.3252005
3. Y. Zhao, P. Zou, Z. He, Z. Li, N. Chi, Low spatial complexity adaptive artificial neural network post-equalization algorithms in MIMO visible light communication systems, *Opt. Express*, **29** (2021), 32728–32738. https://doi.org/10.1364/oe.440155
4. Z. Jin, L. Yan, S. Zhu, X. Cui, P. Tian, 10-Gbps visible light communication in a 10-m free space based on violet series-biased micro-LED array and distance adaptive pre-equalization, *Opt. Lett.*, **48** (2023), 2026–2029. https://doi.org/10.1364/ol.487747
5. R. Kisacik, M.Y. Yagan, M. Uysal, A. E. Pusane, A. D. Yalcinkaya, A new LED response model and its application to pre-equalization in VLC systems, *IEEE Photon. Technol. Lett.*, **33** (2021), 955–958. https://doi.org/10.1109/lpt.2021.3100924
6. Z. Du, M. Fu, B. Wu, M. Sun, B. Zheng, Equalization method for bandwidth-constrained underwater visible light communication systems, *Opt. Eng.*, **64** (2025), 018101. https://doi.org/10.1117/1.oe.64.1.018101
7. S. H. Ab Aziz, N. A. M. Nor, S. A. Zabidi, Comprehensive analysis of a bridged-T pre-equalizer circuit for high-speed visible light communications, *IIUM Eng. J.*, **26** (2025), 205–216. https://doi.org/10.31436/iiumej.v26i1.3255
8. J. Shi, W. Niu, Z. Li, C. Shen, J. Zhang, S. Yu, Optimal adaptive waveform design utilizing an end-to-end learning-based pre-equalization neural network in an UVLC system, *J. Lightw. Technol.*, **41** (2022), 1626–1636. https://doi.org/10.1109/jlt.2022.3225335
9. X. Chen, Y. Wang, X. Liu, Z. Wang, X. Zhang, F. Zhou, A hybrid active-passive single-order equalizer for visible light communication systems, *IEEE Photon. Technol. Lett.*, **35** (2023), 1395–1398. https://doi.org/10.1109/lpt.2023.3327268
10. R. Zhang, J. Xiong, M. Li, L. Lu, Design and implementation of low-complexity pre-equalizer for 1.5 GHz VLC system, *IEEE Photon. J.*, **16** (2024), 7300410. https://doi.org/10.1109/jphot.2024.3351192
11. A. Thao, X. Kang, G. H. Lee, Implementation of an optical wireless communication system deploying a novel post-equalization circuit for indoor application, *Microw. Opt. Technol. Lett.*, **63** (2021), 13–18. https://doi.org/10.1002/mop.32556
12. C. Chen, Y. Nie, M. Liu, Y. Du, R. Liu, Z. Wei, Digital pre-equalization for OFDM-based VLC systems: Centralized or distributed? *IEEE Photon. Technol. Lett.*, **33** (2021), 1081–1084. https://doi.org/10.1109/lpt.2021.3104618

13. J. Shi, W. Xiao, Y. Ha, W. Niu, Z. Xu, O. Huang, et al., 3.76-Gbps yellow-light visible light communication system over 1.2 m free space transmission utilizing a Si-substrate LED and a cascaded pre-equalizer network, *Opt. Express*, **30** (2022), 33337–33352. https://doi.org/10.1364/oe.463989
14. J. Jin, J. Wang, H. Lu, D. Chen, Separate least mean square based equalizer with joint optimization for multi-CAP visible light communication, *China Commun.*, **19** (2022), 264–273. https://doi.org/10.23919/jcc.2022.01.019
15. Z. Lu, J. Cai, Z. Xu, Y. Zhou, J. Zhang, C. Shen, et al., 11.2 Gbps 100-meter free-space visible light laser communication utilizing bidirectional reservoir computing equalizer, *Opt. Expres*s, **31** (2023), 44315–44327. https://doi.org/10.1364/oe.506056
16. H. Yang, S. Zhang, A. Alphones, C. Chen, K. Y. Lam, Z. Xiong, et al., An advanced integrated visible light communication and localization system, *IEEE Trans. Commun.*, **71** (2023), 7149–7162. https://doi.org/10.1109/tcomm.2023.3309823
17. X. Huang, F. Yang, C. Pan, J. Song, Flexible NOMA-based NOHO-OFDM scheme for visible light communication with iterative interference cancellation, *Opt. Express*, **29** (2021), 5645–5657. https://doi.org/10.1364/oe.420848
18. P. Kanjilal, A. Kumar, S. Bhowmick, J. P. Maroor, A. Nanthaamornphong, Implementation of companding scheme for performance enhancement of optical OFDM structure, *J. Opt. Commun.*, **46** (2025), 733–740. https://doi.org/10.1515/joc-2024-0095
19. Y. Xu, L. Huang, W. Jiang, W. Hu, L. Yi, Automatic optimization of Volterra equalizer with deep reinforcement learning for intensity-modulated direct-detection optical communications, *J. Lightw. Technol.*, **40** (2022), 5395–5406. https://doi.org/10.1109/jlt.2022.3177446
20. S. Li, Y. Zou, F. Liu, J. Song, Pre-equalization scheme for visible light communications with trial-and-error learning, *Opt. Lett.*, **49** (2024), 1636–1639. https://doi.org/10.1364/ol.516235
21. C. He, S. Collins, Signal demodulation using a radial basis function neural network (RBFNN) in a silicon photomultiplier-based visible light communication system, *IEEE Photon. J.*, **14** (2022), 7337814. https://doi.org/10.1109/jphot.2022.3184672
22. H. Chen, W. Niu, Y. Zhao, J. Zhang, N. Chi, Z. Li, Adaptive deep-learning equalizer based on constellation partitioning scheme with reduced computational complexity in UVLC system, *Opt. Express*, **29** (2021), 21773–21782. https://doi.org/10.1364/oe.432351
23. P. Miao, G. Chen, K. Cumanan, Y. Yao, J. A. Chambers, Deep hybrid neural network-based channel equalization in visible light communication, *IEEE Commun. Lett.*, **26** (2022), 1593–1597. https://doi.org/10.1109/lcomm.2022.3172219
24. Y. Yu, D. Zhao, J. Chen, K. Fu, S. Yu, L. Gao, LSTM-KAN: Revolutionizing indoor visible light localization with robust sequence learning, *Big Data Min. Anal.*, **8** (2025), 1245–1260. https://doi.org/10.26599/bdma.2025.9020021
25. T. Zhang, Q. He, M. Luo, W. Liu, D. Wang, D. Zhang, et al., Efficient nonlinear equalization across S+C+L bands using online knowledge distillation in high-capacity DWDM systems, *Opt. Express*, **33** (2025), 20433–20448. https://doi.org/10.1364/oe.559025
26. R. Kısacık, M. Y. Yagan, M. Uysal, A. E. Pusane, T. Baykas, G. Dundar, et al., A 130 nm CMOS receiver for visible light communication, *J. Lightw. Technol.*, **40** (2022), 3681–3687. https://doi.org/10.1109/jlt.2022.3150250

27. Z. Yang, T. Wang, Y. Lin, Y. Chen, H. Zeng, J. Pei, et al., A vision chip with complementary pathways for open-world sensing, *Nature*, **629** (2024), 1027–1033. https://doi.org/10.1038/s41586-024-07358-4
28. X. Ke, Q. Zhang, H. Qin, CNN neural network temporal feature storage structure fusion for the visible channel equalization algorithm, *Appl. Opt.*, **62** (2023), 9238–9252. https://doi.org/10.1364/ao.502683
29. L. S. Hsu, D. C. Tsai, C. W. Chow, Y. Liu; Y. H. Chang, Y. Z. Lin, et al., Using data pre-processing and convolutional neural network (CNN) to mitigate light deficient regions in visible light positioning (VLP) systems, *J. Lightw. Technol.*, **40** (2022), 5894–5900. https://doi.org/10.1109/jlt.2022.3184931
30. Z. Zhao, F. N. Khan, Z. A. H. Qasem, B. Deng, Q. Li, Z. Liu, et al., Convolutional-neural-network-based versus vision-transformer-based SNR estimation for visible light communication networks, *Opt. Lett.*, **48** (2023), 1419–1422. https://doi.org/10.1364/ol.485321
31. O. Huang, J. Shi, N. Chi, Performance and complexity study of a neural network post-equalizer in a 638-nm laser transmission system through over 100-m plastic optical fiber, *Opt. Eng.*, **61** (2022), 126108. https://doi.org/10.1117/1.oe.61.12.126108

AIMS Press