LightGBM Inference Benchmark Report

Blackcore® ACE 3100-TS+ Server with AMD Alveo U50

March 2026

Contents

1 Xelera Silva

Gradient Boosting Tree frameworks, such as XGBoost, LightGBM, and CatBoost, are widely used in financial trading systems, ransomware and DDOS detection systems, and recommender systems. Xelera Silva achieves best-in-class inference latency and throughput by leveraging commercial off-the-shelf data-centre grade PCIe-based accelerators.

Xelera Silva comes in two flavors. One is software-only optimized deployment, where inference runs on only the CPU. The other executes inference on AMD Alveo accelerator platforms to achieve even lower latency. The user application interacts with Silva via a C/C++ or Python API.

The benchmarking was conducted across the following tests:

- **Single-Model Inference** (*Single-Model Inference*): This test measures the end-to-end latency of running inference using Gradient Boosting Tree models. It provides a performance comparison between a CPU-based implementation and the Xelera Silva solution running on CPU and on the accelerator. Benchmarks were conducted using a single model to assess baseline inference performance.
- **Multi-Model Inferences** (*Multi-Model Inference*): This test evaluates the end-to-end latency of Gradient Boosting Tree model inference under concurrent execution conditions. It compares multi-model scenarios to assess Xelera Silva's impact of parallel inference workloads running on CPU and on the accelerator.
- **CPU-Only Inference** (*CPU-Only Inference*): This test measures the end-to-end latency of Xelera Silva running only on CPU. It compares multi-model inference, each pinned to a set of CPU-cores.
- **PCIe Access** (*PCIe Access*): This test measures the latency of a single 32-bit register read from the accelerator card by the host system. The results in this mode are primarily influenced by the server's PCIe architecture and the specific slot configuration used. See *Appendix* for full results.
- **PCIe Data Transfer** (*PCIe Data Transfer*): This benchmark evaluates the latency of multiple PCIe data transfer sizes to and from the accelerator card, comparing single-process and multi-process scenarios to assess the impact of concurrent access. See *Appendix* for full results.

These results offer insight into the performance characteristics of Xelera Silva software.

2 Test Setup

This document presents latency benchmark results for the Xelera Silva software executed on a Blackcore® ACE 3100-TS+ server equipped with an AMD Alveo U50 PCIe-based accelerator card. Table 1 shows the system specification.

Table 1: System under test

Server	ACE 3100-TS+ CPU: AMD Ryzen Threadripper PRO 9975WX CPU Frequency: 32 cores @ 5.1GHz overclocked all-core frequency CPU Cache: 128MB Memory: 8x 32GB DDR5 ECC UDIMM
OS	Linux Rocky 9.7
PCIe interface	Gen4 x8
AMD Alveo Card	U50 with Xelera PCIe ULL shell 2.0.0.17
Driver	Xelera PCIe ULL 2.16.8
ML Inference Software	Intel oneDAL only 2024.5.0 (**intel_oneDAL_sw**) Xelera Silva software only 8.3.0 (**xl_silva_sw**) Xelera Silva accelerator 8.3.0 (**xl_silva_acc**)

The benchmark uses three LightGBM regression model configurations, defined in Table 2 and referenced throughout the report as **GBT_S**, **GBT_M** and **GBT_X**. All models were trained on synthetic random data.

Table 2: LightGBM model configurations

Model ID	Number of Features	Number of Trees	Number of Levels
GBT_S	64	200	5
GBT_M	128	1000	8
GBT_X	1024	4000	12

The measured inference latency (batch size 1) is measured at API interface. In the case of an offload accelerator, it accounts for PCIe data transfer between the host and accelerator, as well as the computation time on the accelerator itself.

3 Single-Model Inference

The Xelera Silva software was compared against other software frameworks for the acceleration of Gradient Boosting Tree Machine Learning models. The compared ML inference software frameworks are: **intel_oneDAL_sw**, **xl_silva_sw**, **xl_silva_acc**.

The Intel oneDAL framework uses only CPU optimizations to accelerate the inference of gradient boosting models, such as the use of vector extension instructions, branch prediction and integer comparisons. Xelera Silva runs on CPU or on FPGA-based accelerator.

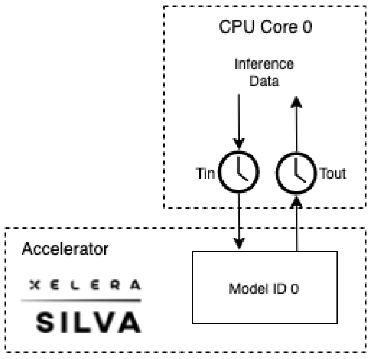

Fig. 1: Hardware system overview

The roundtrip latency at the API interface ($T_{out} - T_{in}$) is measured when running inference for the GBT_S and GBT_M model configurations defined in Table 2.

For each software framework configuration, the test involves running inference on the two models. The process is pinned to CPU core 1, which has also been isolated. The test is conducted 1,000,000 times.

3.1 Results: GBT_S

Fig. 2 shows the latency statistics of Xelera Silva in comparison to the third-party software frameworks when running the GBT_S model (Table 2). The graphs show the fraction of inference measurements (y-axis) below a specified latency (x-axis).

Table 3 compares the minimum, maximum, median (50th percentile) and the 99th percentile latency (microseconds) of the graph above.

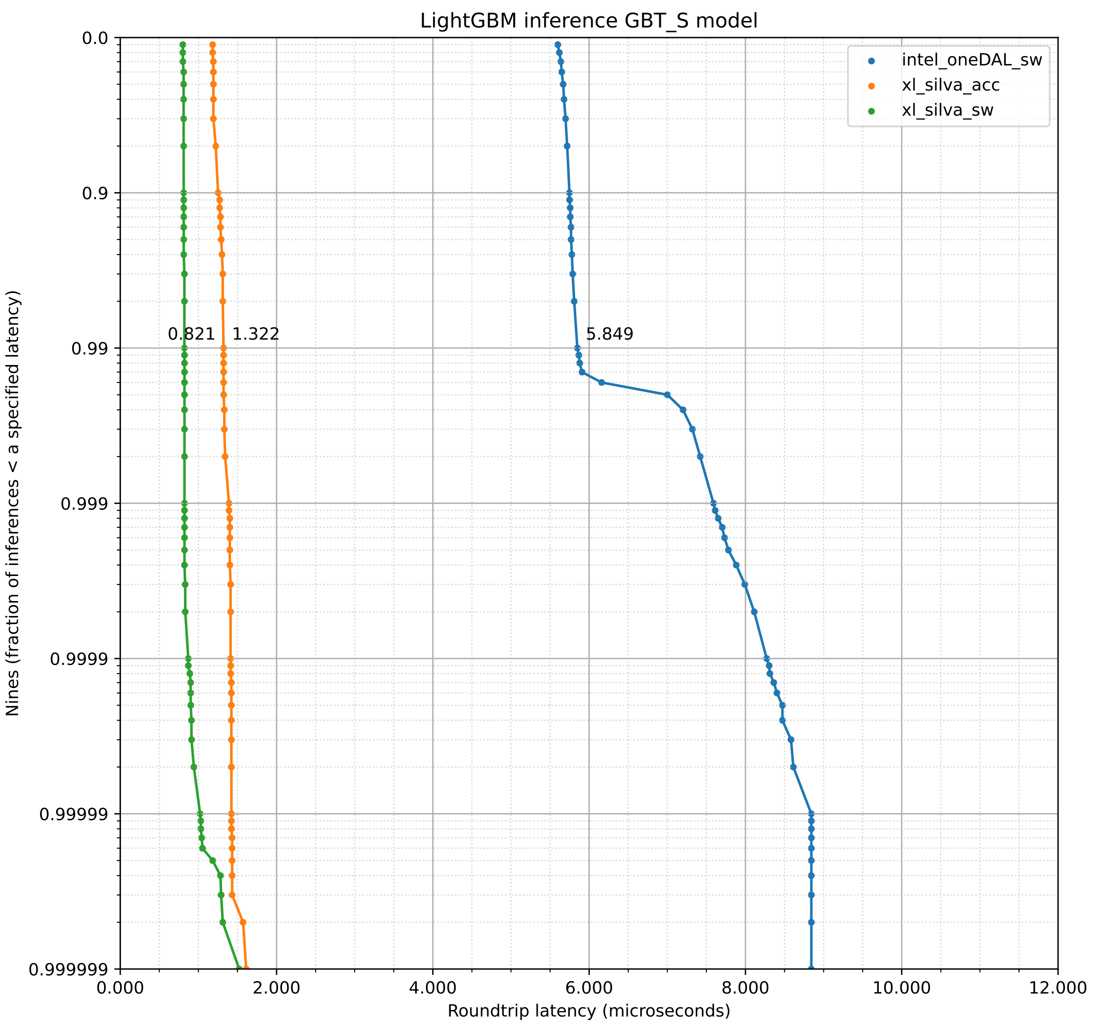

Fig. 2: Latency statistic for GBT_S model inference

Table 3: Latency statistics for GBT_S model (microseconds)

ML Inference Software	Minimum	Maximum	50th percentile	99th percentile
intel_oneDAL_sw	5.468	8.842	5.668	5.849
xl_silva_sw	0.791	1.522	0.811	0.821
xl_silva_acc	1.171	1.612	1.192	1.322

3.2 Results: GBT_M

Fig. 3 shows the latency statistics of Xelera Silva in comparison to the third-party software frameworks when running the GBT_M model (Table 2). The graphs show the fraction of inference measurements (y-axis) below a specified latency (x-axis).

Table 4 compares the minimum, maximum, median (50th percentile) and the 99th percentile latency (microseconds) of the graph above.

Table 4: Latency statistics for GBT_M model (microseconds)

ML Inference Software	Minimum	Maximum	50th percentile	99th percentile
intel_oneDAL_sw	26.759	66.790	30.885	32.933
xl_silva_sw	6.38	16.214	7.381	7.661
xl_silva_acc	1.301	1.943	1.362	1.462

3.3 Key Findings

This test measures the end-to-end inference latency of a single Gradient Boosting Tree model in isolation, providing the baseline performance profile for each software framework. In HFT systems, a single model often sits on the critical execution path — between market data arrival and order submission — where every microsecond of predictable latency directly affects fill rates and strategy profitability. The 99th percentile is the operative metric: it defines the worst-case latency the system delivers under normal operation, and any tail inflation translates directly into missed trading windows.

For the **GBT_S** model, **xl_silva_sw** achieves a 99th percentile latency of **0.821 µs** compared to **5.849 µs** for Intel oneDAL — a **7.1× reduction** in tail latency. The spread between minimum and 99th percentile is only 0.030 µs for xl_silva_sw versus 0.381 µs for oneDAL, confirming that Xelera Silva delivers significantly tighter latency distribution, which is essential for deterministic low-latency systems.

For the **GBT_M** model, the advantage is even more pronounced. **xl_silva_acc** delivers a 99th percentile of **1.462 µs**, compared to **32.933 µs** for Intel oneDAL — a **22.5× reduction**. Notably, xl_silva_acc also outperforms xl_silva_sw (99th percentile: 7.661 µs) by **5.2×** on GBT_M, demonstrating that FPGA acceleration becomes increasingly valuable as model complexity grows.

The consistency of xl_silva_acc is a key differentiator: the min-to-99th-percentile spread for xl_silva_acc is just 0.161 µs on GBT_M, compared to 6.174 µs for oneDAL. For high-frequency trading systems where deterministic latency is as important as absolute speed, this near-zero jitter profile makes Xelera Silva uniquely suited to production deployment.

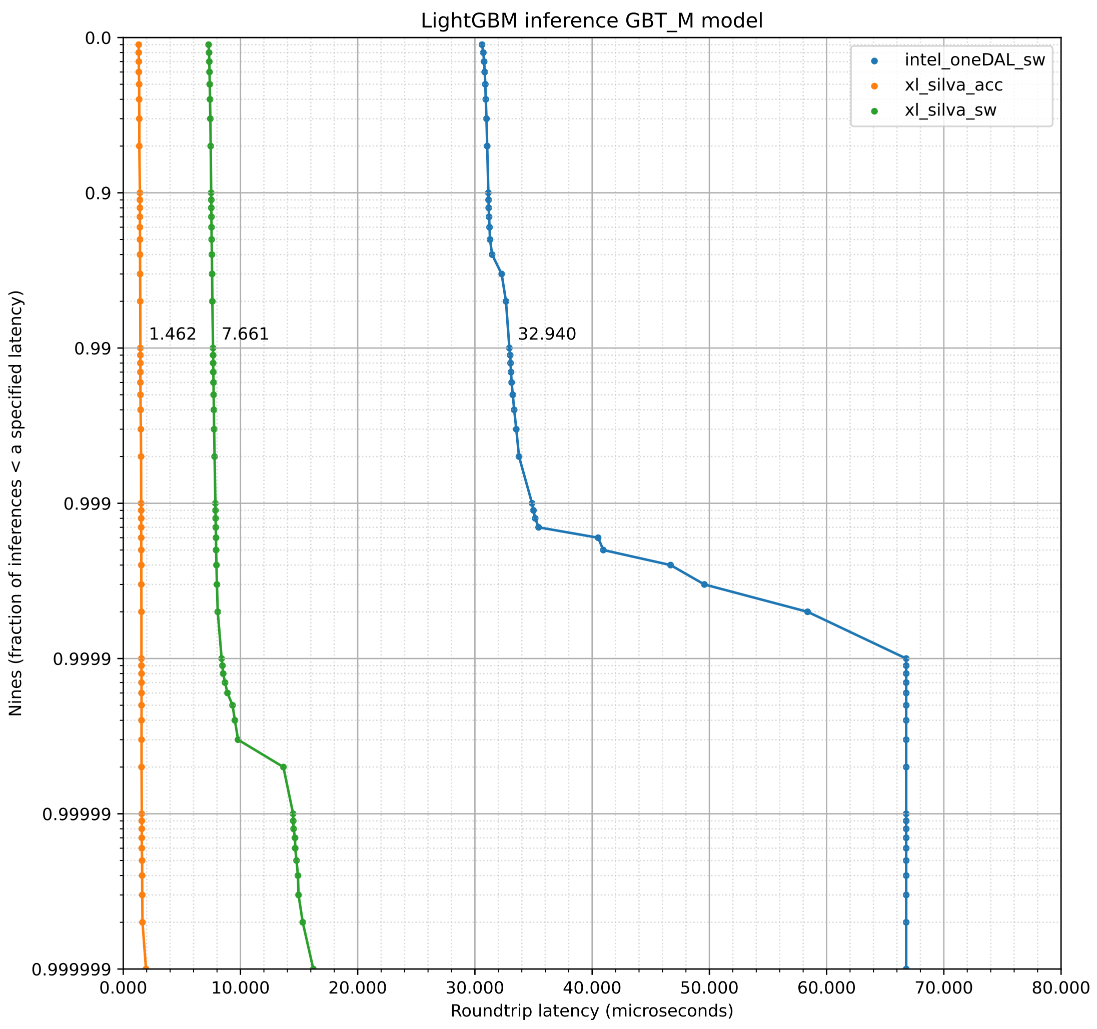

Fig. 3: Latency statistic for GBT_M model inference

4 Multi-Model Inference

In this benchmark, up to 8 models are executed simultaneously and asynchronously. Each model is accessed by the host software via an individual process. The processes are pinned to CPU cores 1 to 8 respectively. These cores have also been isolated. The test is conducted 1,000,000 times, once running CPU-only (**xl_silva_sw**) and once on the FPGA accelerator (**xl_silva_acc**).

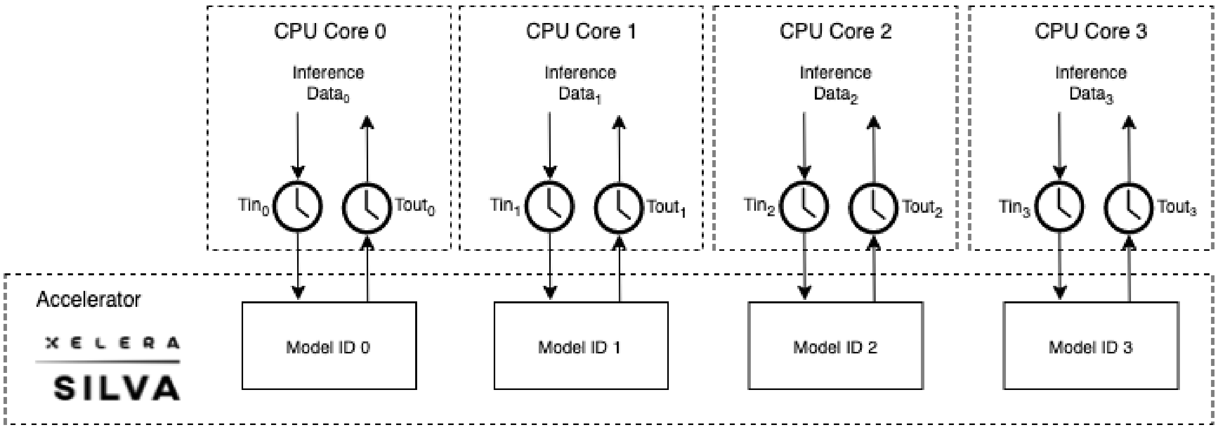

Fig. 4: Multi-model inference setup: 4 concurrent processes on FPGA

The roundtrip latency at the API interface ($T_{out,x} - T_{in,x}$) is measured when running inference for the GBT_S and GBT_M model configurations defined in Table 2.

For each model configuration, the test involves running inference with up to 8 models simultaneously in an **asynchronous** mode (independent processes accessing the models). Each process is pinned to a CPU core (1 to 8). The GBT_S configuration is tested with 8 concurrent models; the GBT_M configuration with 4 concurrent models. The test is conducted 1,000,000 times.

4.1 Results: GBT_S Model

Fig. 5 shows the latency statistics of Xelera Silva when running inference with 8 GBT_S models at the same time on FPGA accelerator (**xl_silva_acc**) and Fig. 6 on CPU-only (**xl_silva_sw**). The graphs show the fraction of inference measurements (y-axis) below a specified latency (x-axis) for each of the 8 concurrent model inferences.

Table 5 compares the minimum, maximum, median (50th percentile) and the 99th percentile latency (microseconds) of the graph above for a run with **xl_silva_acc**.

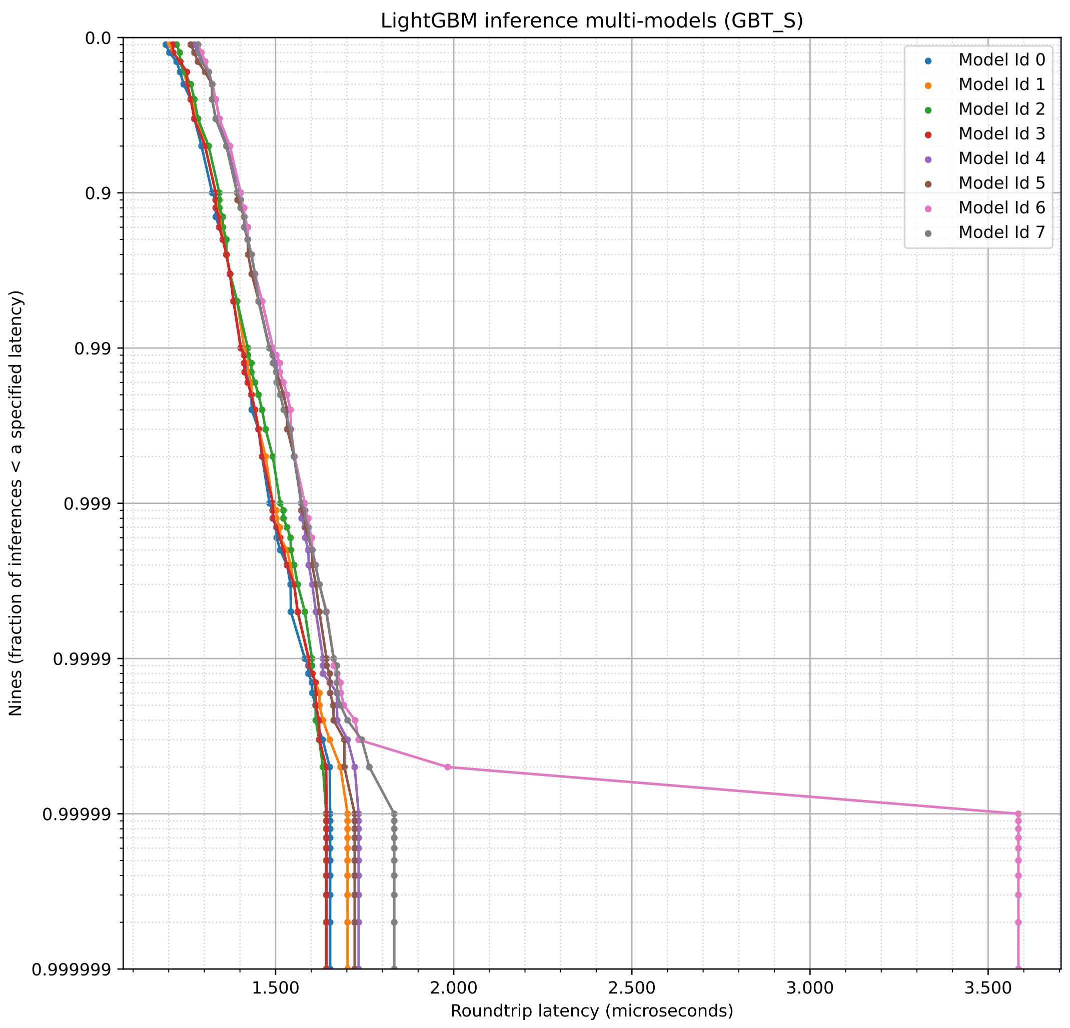

Fig. 5: Latency statistic for concurrent GBT_S model inference with **xl_silva_acc**

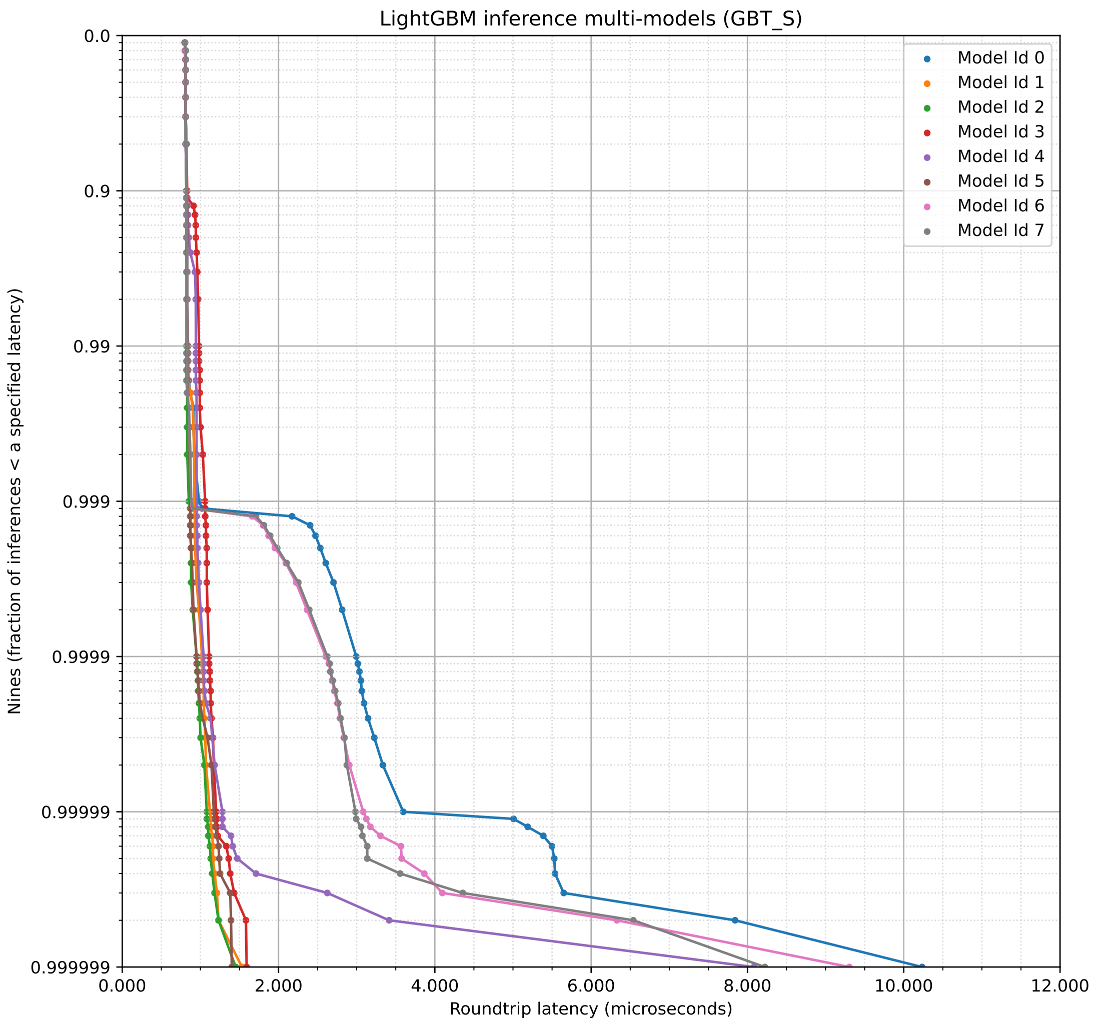

Fig. 6: Latency statistic for concurrent GBT_S model inference with **xl_silva_sw**

Table 5: Latency statistics for concurrent GBT_S model inference (microseconds) with **xl_silva_acc**.

Model ID	Minimum	Maximum	50th percentile	99th percentile
0	1.151	1.653	1.242	1.403
1	1.161	1.702	1.252	1.412
2	1.181	1.642	1.262	1.422
3	1.171	1.643	1.252	1.402
4	1.231	1.733	1.322	1.492
5	1.222	1.722	1.322	1.483
6	1.251	3.585	1.322	1.493
7	1.241	1.833	1.322	1.482

While Table 6 for a run with **xl_silva_sw**.

Table 6: Latency statistics for concurrent GBT_S model inference (microseconds) with **xl_silva_sw**.

Model ID	Minimum	Maximum	50th percentile	99th percentile
0	0.791	10.235	0.811	0.841
1	0.791	1.532	0.811	0.841
2	0.791	1.452	0.811	0.822
3	0.791	1.592	0.811	0.982
4	0.791	1.502	0.811	0.942
5	0.791	1.122	0.811	0.841
6	0.791	1.312	0.811	0.832
7	0.791	1.322	0.811	0.832

4.2 Results: GBT_M Model

Fig. 7 shows the latency statistics of Xelera Silva when running inference with 4 GBT_M models at the same time. The graphs show the fraction of inference measurements (y-axis) below a specified latency (x-axis) for each of the 4 concurrent model inferences.

Table 7 compares the minimum, maximum, median (50th percentile) and the 99th percentile latency (microseconds) of the graph above for a run with **xl_silva_acc**.

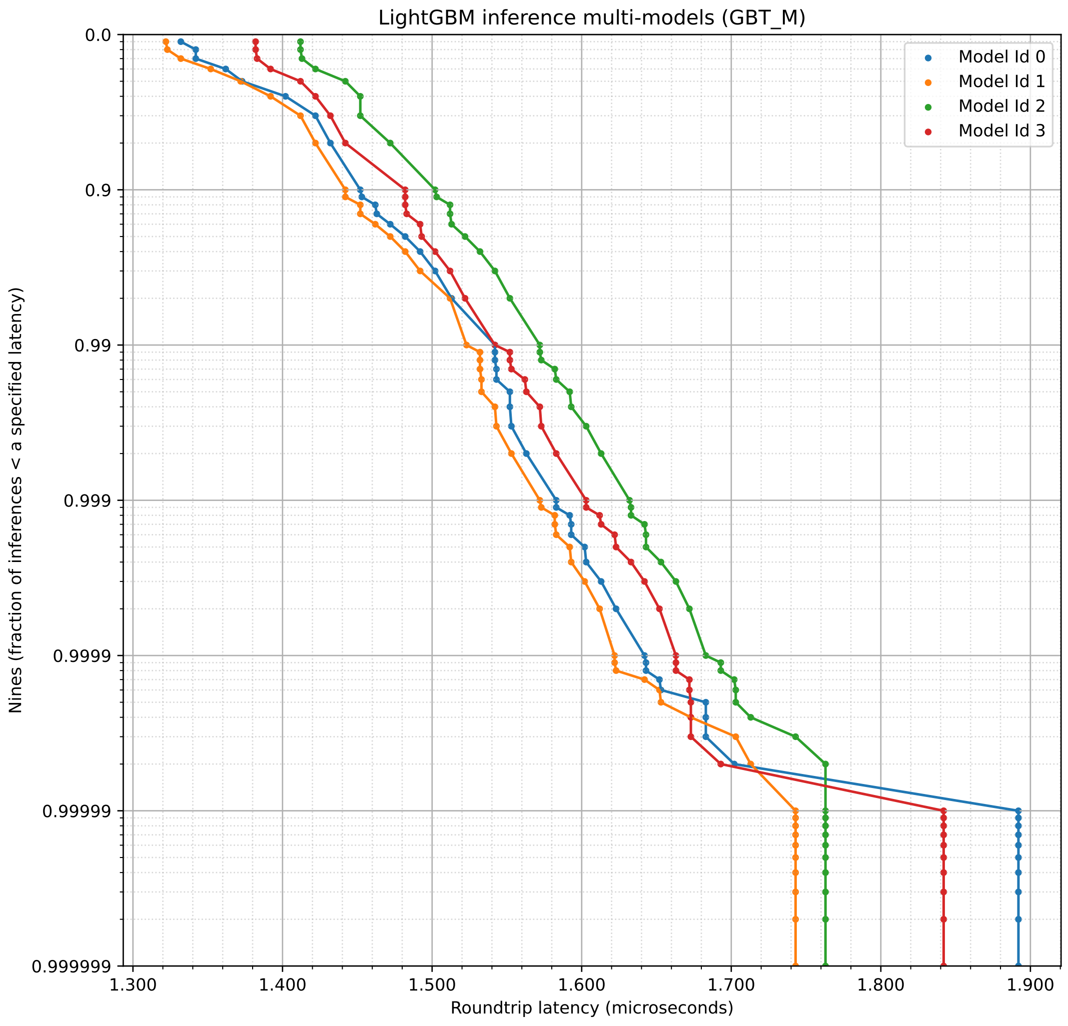

Fig. 7: Latency statistic for concurrent GBT_M model inference

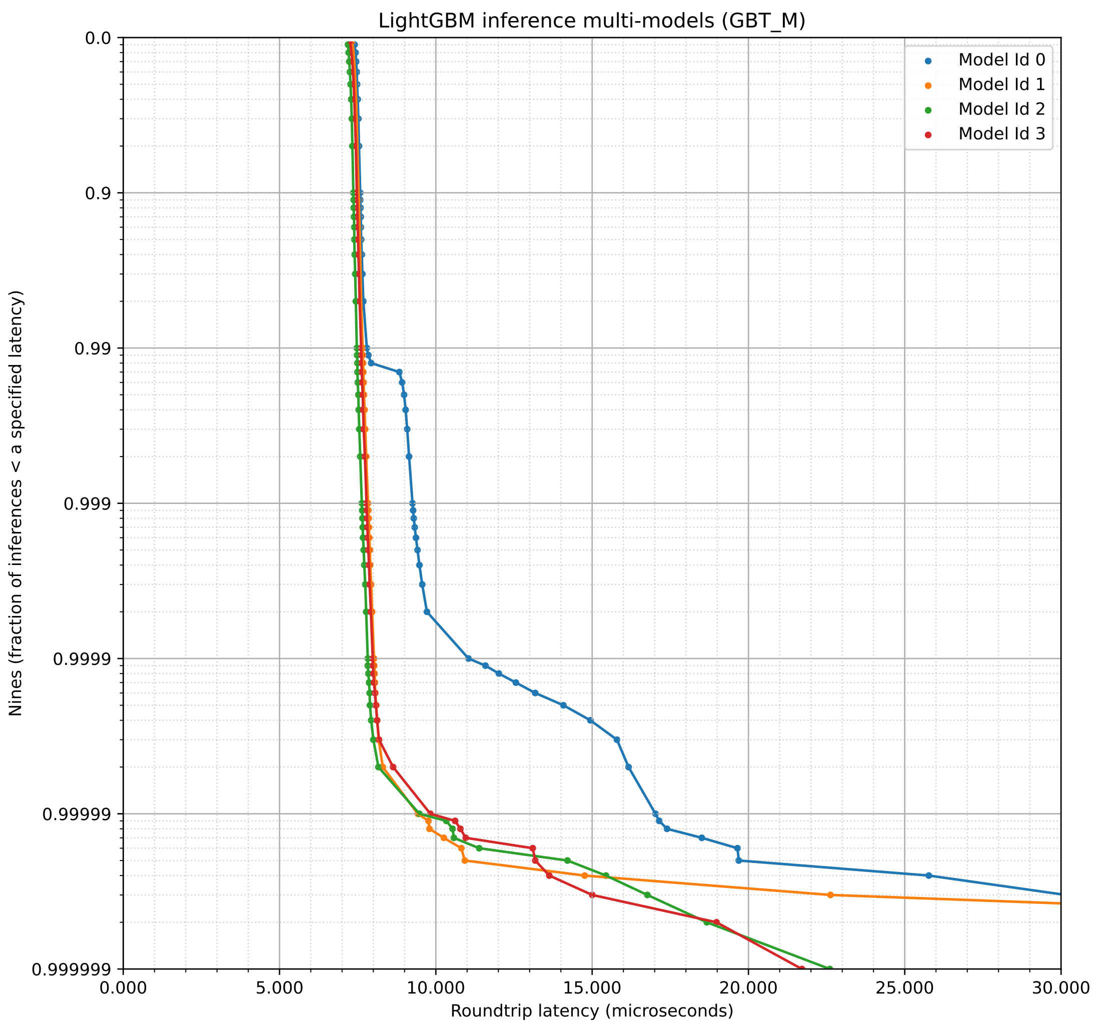

Fig. 8: Latency statistic for concurrent GBT_M model inference with **xl_silva_sw**

Table 7: Latency statistics for concurrent GBT_M model inference (microseconds) with **xl_silva_acc**.

Model ID	Minimum	Maximum	50th percentile	99th percentile
0	1.331	2.853	1.462	1.542
1	1.322	3.175	1.462	1.523
2	1.332	1.843	1.462	1.572
3	1.331	7.586	1.452	1.542

While Table 8 for a run with **xl_silva_sw**.

Table 8: Latency statistics for concurrent GBT_M model inference (microseconds) with **xl_silva_sw**.

Model ID	Minimum	Maximum	50th percentile	99th percentile
0	6.681	389.819	7.481	7.792
1	6.480	110.066	7.431	7.661
2	6.420	22.604	7.271	7.472
3	6.440	21.712	7.371	7.611

4.3 Key Findings

This test evaluates inference latency when multiple independent models execute simultaneously on the same hardware, each served by a dedicated process. This scenario is directly representative of production HFT deployments, where a trading system typically runs several strategy models in parallel — each monitoring different instruments or signals — and must guarantee that adding a new strategy does not degrade the latency SLA of models already in production. The ability to scale horizontally without tail latency growth is therefore as important as raw single-model speed.

For **GBT_S with 8 concurrent models**, **xl_silva_acc** holds a 99th percentile latency between **1.403 µs and 1.493 µs** across all 8 model instances — a spread of only 0.090 µs between the fastest and slowest model. In contrast, **xl_silva_sw** shows a wider spread (0.841 µs to 0.982 µs at the 99th percentile) and significantly higher maximum latency spikes (up to 10.235 µs vs 3.585 µs for xl_silva_acc). For trading systems requiring uniform latency across all running strategies, the FPGA path provides substantially better tail behaviour.

For **GBT_M with 4 concurrent models**, the scaling advantage of xl_silva_acc is decisive. The 99th percentile latency remains within **1.523 µs to 1.572 µs** — less than 0.050 µs of inter-model variation — while xl_silva_sw shows 99th percentile latencies of 7.472 µs to 7.792 µs with maximum spikes reaching **389.819 µs**. Such spikes would be catastrophic in a live trading context; xl_silva_acc eliminates them entirely.

The scaling behaviour of xl_silva_acc is also noteworthy: moving from 1 to 4 or 8 concurrent models produces virtually no increase in 99th percentile latency compared to the single-model baseline in *Single-Model Inference*. This near-linear horizontal scaling means trading firms can add model instances to the same accelerator card without degrading the latency SLA of existing strategies.

5 CPU-Only Inference

This chapter presents the software-only inference results for Xelera Silva when inference is executed on CPU cores only. The objective is to characterize how latency evolves under three complementary execution modes: running a single model on one core, scaling the number of models executed in parallel up to the CPU capacity of the system, and distributing a larger model across multiple CPU cores.

The measurements in this chapter focus on the API roundtrip latency ($T_{out} - T_{in}$) observed for the model configurations defined in Table 2 in *Test Setup*. For the GBT_S and GBT_M configurations, the intent is to quantify the latency of the worst core as the number of concurrently running models increases. For the GBT_X configuration, the objective is to evaluate both multi-core execution of a single model and concurrent execution of multiple multi-core model instances.

Unless stated otherwise, each model instance is pinned to dedicated CPU cores so that the reported results reflect the scaling behavior of Xelera Silva under controlled software-only conditions.

5.1 GBT_S

This section reports the latency of the GBT_S model when one model instance is mapped to one CPU core and when multiple GBT_S instances are executed in parallel. The table reports the minimum, 50th, 99th, and maximum latency measured on the worst-performing core for each level of parallelism.

Table 9: Xelera Silva software-only latency for GBT_S worst core (microseconds)

Models in parallel	Minimum	50th percentile	99th percentile	Maximum
1	0.791	0.811	0.821	1.522
2	0.791	0.811	0.831	6.980
4	0.791	0.811	0.842	9.204
8	0.791	0.811	0.982	10.235
16	0.791	0.812	0.981	189.166
32	0.791	0.811	0.951	220.393

The GBT_S results are intended to show how the smallest model scales from a single active core to full parallel loading of the platform. By reporting the worst-core statistics, the table highlights the latency bound that should be used when dimensioning the system for deterministic multi-model operation.

5.2 GBT_M

This section reports the latency of the GBT_M model when one model instance is mapped to one CPU core and when multiple GBT_M instances are executed in parallel. As for GBT_S, the reported values correspond to the minimum, 50th, 99th, and maximum latency observed on the worst core at each level of concurrency.

Table 10: Xelera Silva software-only latency for GBT_M worst core (microseconds)

Models in parallel	Minimum	50th percentile	99th percentile	Maximum
1	6.439	7.311	7.461	1240.545
2	6.690	7.462	7.661	543.922
4	6.681	7.481	7.792	389.819
8	6.670	7.471	7.962	444.391
16	6.620	7.562	7.902	770.944
32	6.670	8.002	8.733	1675.462

The GBT_M results quantify the behavior of the medium model under increasing parallel software load. Together with the GBT_S measurements, they provide a direct comparison of how model size influences latency while scaling the number of simultaneously active model instances up to the number of available CPU cores.

5.3 GBT_X

GBT_X represents the largest software-only model configuration considered in this benchmark. In this case, the evaluation is split into two parts. First, the latency of a single GBT_X instance is measured while spreading inference across multiple CPU cores. Second, the latency is measured when multiple GBT_X instances run concurrently, each instance being distributed over 8 CPU cores.

5.3.1 Single GBT_X model spread across multiple cores

The following table reports the latency when a single GBT_X model is executed using 1, 2, 4, or 8 CPU cores. The results capture the minimum, 50th, 99th, and maximum roundtrip latency for each core allocation.

Table 11: Xelera Silva software-only latency for GBT_X single-model inference across multiple CPU cores (microseconds)

CPU cores	Minimum	50th percentile	99th percentile	Maximum
1	57.697	80.161	115.704	1361.977
2	31.268	53.301	80.122	6846.109
4	14.382	27.390	36.818	6597.390
8	7.071	13.264	16.535	2030.551

These results are intended to show how a single large model benefits from additional CPU resources and to identify the point at which adding cores no longer produces a meaningful latency reduction.

5.3.2 Multiple GBT_X models spread over 8 cores each

The following table reports the latency when each GBT_X model instance is distributed over 8 CPU cores and 1, 2, or 4 model instances are executed at the same time. The table captures the impact of concurrent multi-core execution on latency.

Table 12: Xelera Silva software-only latency for GBT_X with 8 CPU cores per model and multiple concurrent models (microseconds)

Models in parallel	Minimum	50th percentile	99th percentile	Maximum
1	7.071	13.264	16.535	2030.551
2	6.880	12.950	16.175	4466.137
4	7.170	14.782	27.902	38000.277

5.4 Key Findings

The software-only results show two main behaviors.

First, GBT_S and GBT_M exhibit the same scaling trend under increasing parallelism. For both models, the 99th percentile latency remains nearly constant from 1 to 32 parallel model instances, showing that replication across CPU cores has little impact on the typical inference latency. The difference is the absolute latency level: GBT_S remains below 1 microsecond at the 99th percentile, whereas GBT_M remains within 7.461–8.733 microseconds at the 99th percentile. In both cases, the main impact of higher load appears in the observed maximum latency rather than in the typical latency.

Second, GBT_X behaves differently because it benefits strongly from multi-core execution. When a single GBT_X model is distributed across more CPU cores, the 99th percentile latency decreases from 115.704 microseconds on 1 core to 16.535 microseconds on 8 cores. This shows that the largest model requires multiple CPU cores to reach low software-only latency.

For concurrent GBT_X execution with 8 CPU cores per model, the latency remains close to the single-model case for up to 2 parallel models. At 4 parallel models, the 99th percentile increases to 27.902 microseconds, showing the onset of contention in the tail latency.

Overall, the results show that Xelera Silva software-only inference scales efficiently on CPU cores. GBT_S and GBT_M maintain stable 99th percentile latency across the full parallel loading range, while GBT_X achieves substantial latency reduction when distributed across multiple cores. The main dimensioning constraint is the growth of tail and maximum latency at high load.

6 Summary and Conclusions

This report benchmarks the Xelera Silva machine learning inference software on the Blackcore® ACE 3100-TS+ server equipped with an AMD Alveo U50 FPGA accelerator, with a focus on the requirements of high-frequency trading (HFT) and other latency-critical real-time systems. In these environments, the decisive metrics are not average-case performance but **99th percentile latency** and **tail reliability**: a trading system must meet its latency budget on every inference, not merely on most of them.

The benchmark suite covers the full signal path from PCIe host access through data transfer to single-model and multi-model inference, using three LightGBM model configurations (GBT_S, GBT_M, GBT_X) representative of the model sizes deployed in real trading pipelines.

6.1 Platform Reliability

The foundation of the results is the stability of the underlying platform. PCIe register access latency spans just **30 nanoseconds** between minimum and maximum across one million measurements, with identical 50th and 99th percentile values of 0.547 µs. Single PCIe data transfers remain at or below a **99th percentile of 1.152 µs** for all payload sizes up to 1024 bytes, and hold within **1.362 µs** even under 10 fully concurrent parallel transfer streams. These figures confirm that the ACE 3100-TS+ PCIe topology and the Xelera ULL driver introduce no measurable variability into the communication path — a prerequisite for any system claiming deterministic latency.

6.2 Single-Model Inference

For the GBT_S model, Xelera Silva software-only (**xl_silva_sw**) achieves a **99th percentile of 0.821 µs**, compared to **5.849 µs** for Intel oneDAL — a **7.1× reduction** in tail latency. The latency distribution is also significantly tighter: the min-to-99th spread is 0.030 µs for xl_silva_sw versus 0.381 µs for oneDAL.

For the more computationally intensive GBT_M model, **xl_silva_acc** (FPGA-accelerated) delivers a **99th percentile of 1.462 µs** against **32.933 µs** for Intel oneDAL — a **22.5× improvement** — while the min-to-99th spread is just 0.161 µs. The xl_silva_acc result also outperforms xl_silva_sw by **5.2×** on GBT_M, demonstrating that FPGA offload becomes progressively more valuable as model depth and tree count increase.

6.3 Multi-Model Scalability

A key requirement for production HFT deployments is the ability to run many independent strategy models simultaneously without degrading the latency of any individual model. The concurrent inference results show that Xelera Silva meets this requirement with exceptional consistency.

With **8 GBT_S models running simultaneously**, xl_silva_acc delivers 99th percentile latencies between **1.403 µs and**

1.493 µs across all model instances — an inter-model spread of only 0.090 µs. With **4 concurrent GBT_M models**, the 99th percentile remains within **1.523 µs to 1.572 µs**, compared to 7.472–7.792 µs for xl_silva_sw. Critically, xl_silva_sw maximum latency spikes reach **389.819 µs** under four concurrent GBT_M streams — more than 200× the xl_silva_acc maximum of 7.586 µs. Such spikes are incompatible with live trading operation; xl_silva_acc eliminates them.

The 99th percentile latency of xl_silva_acc is essentially flat from single-model to full concurrent load. This near-ideal horizontal scaling means that trading firms can add model instances to the same FPGA card to increase strategy coverage without renegotiating the latency SLA of strategies already in production.

6.4 Software-Only Scaling (CPU Path)

For deployments that do not require FPGA acceleration, the CPU-only results in *CPU-Only Inference* show that xl_silva_sw scales efficiently across all tested configurations. For GBT_S and GBT_M, the 99th percentile remains stable from 1 to 32 parallel model instances, confirming that CPU-based replication is a viable path for less latency-sensitive workloads. For the largest model (GBT_X), distributing inference across 8 CPU cores reduces the 99th percentile from 115.704 µs to 16.535 µs — a 7× gain — with further scaling to multiple concurrent 8-core instances remaining manageable up to 2 parallel models before tail latency begins to grow.

6.5 Conclusions

Xelera Silva on the Blackcore® ACE 3100-TS+ platform delivers the combination of low absolute latency, tight tail distribution, and stable multi-model scalability required for production-grade high-frequency trading infrastructure:

- **Sub-1.5 µs 99th percentile** inference latency for all tested model sizes when using FPGA acceleration, across both single-model and full concurrent multi-model loads.
- **Up to 22.5× lower tail latency** compared to Intel oneDAL on the GBT_M model.
- **Near-zero inter-model latency variation** under concurrent execution — a critical property for fair and deterministic multi-strategy deployment.
- **Elimination of maximum-latency spikes** that would cause missed trading windows or risk SLA violations in a live environment.
- **Flexible deployment options**: the software-only CPU path provides a cost-effective baseline for less demanding workloads, while the FPGA path delivers a step-change in tail latency and consistency for the most demanding strategies.

These results demonstrate that Xelera Silva is well-suited for real-time financial trading systems, cyber-defense pipelines, and any application where consistent low-latency inference at scale is a hard system requirement.

7 Appendix

7.1 PCIe Access

The PCIe access latency has been measured with the open-source tool `pcie-lat`.

The latencies are measured by calculating the time taken to read a 32-bit word from a PCIe device using a Linux kernel module.

The process is pinned to CPU core 1, which has also been isolated. The test is conducted 1,000,000 times.

7.1.1 Results

Fig. 9 displays the latency statistics for reading a 32-bit word from a PCIe device. The y-axis represents the fraction of inference measurements that fall below a specified latency on the x-axis.

Table 13 compares the minimum, maximum, median (50th percentile) and the 99th percentile latency (in microseconds) of the graph above.

Table 13: Latency statistics for 32-bit word PCIe read (microseconds)

Minimum	Maximum	50th percentile	99th percentile
0.537	0.567	0.547	0.547

7.1.2 Key Findings

The PCIe access latency measurements confirm that the Blackcore® ACE 3100-TS+ server provides an exceptionally stable host-to-accelerator communication path. The 32-bit register read latency ranges from a minimum of **0.537 µs** to a maximum of **0.567 µs** — a total jitter envelope of only **30 nanoseconds** across 1,000,000 measurements.

The 50th and 99th percentile are identical at **0.547 µs**, indicating that the PCIe access latency distribution is essentially a step function with no measurable tail. For high-frequency trading systems, this means the raw communication overhead introduced by using an FPGA accelerator is both predictable and negligible — it contributes a fixed, known cost to inference latency without adding any statistical uncertainty.

This result also validates the server platform itself: the PCIe Gen4 x8 topology and the isolated core configuration together eliminate the latency variability that is common in shared-access server environments, establishing a reliable foundation for the inference benchmarks reported in the main chapters.

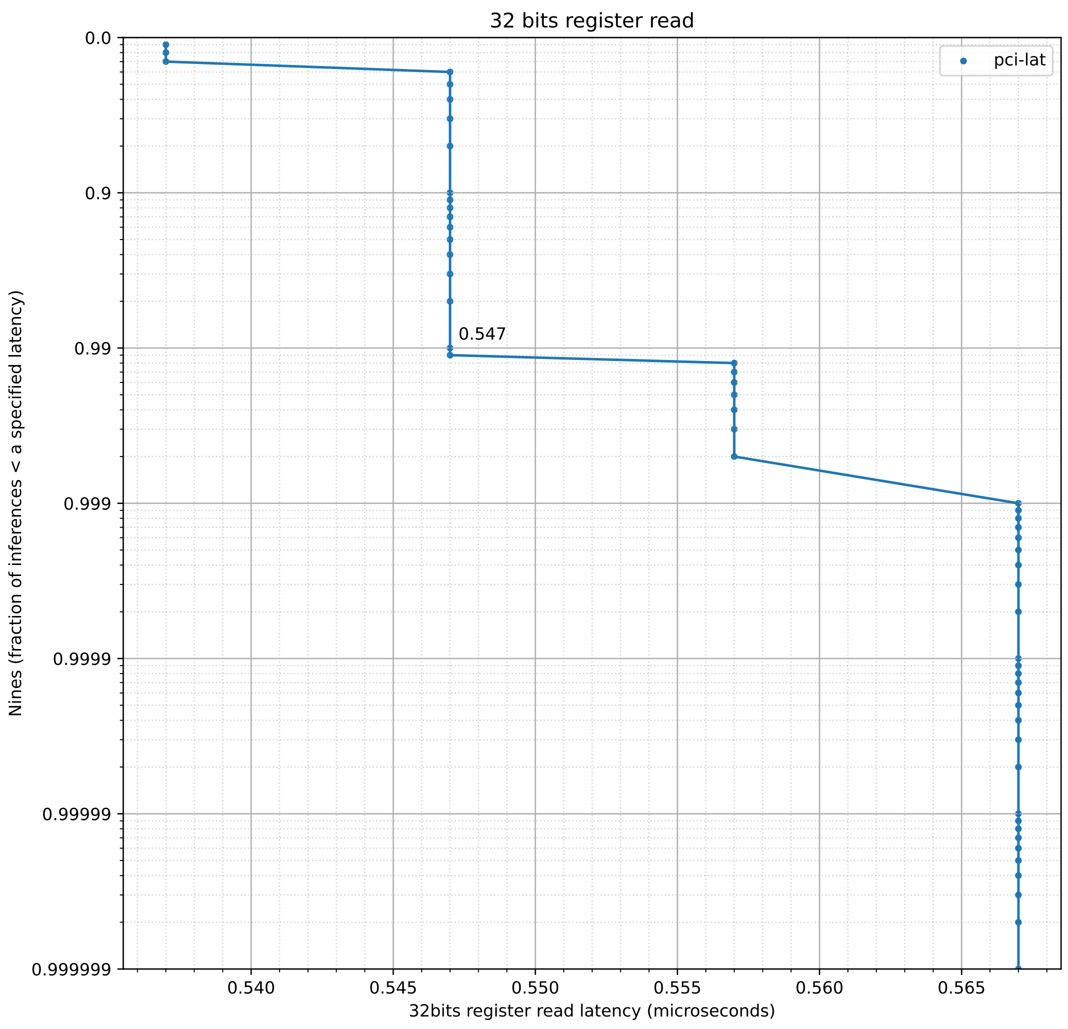

Fig. 9: Latency statistic for 32-bit word read from PCIe device

7.2 PCIe Data Transfer

This benchmark evaluates the latency of PCIe data transfers of various sizes between the host system and the accelerator card.

Each data packet is transmitted to the accelerator, internally looped back, and returned to the host, with roundtrip latency measured at the API interface ($T_{out} - T_{in}$) to capture the full transfer cycle. Packet sizes tested include 16, 32, 64, 128, 256, 512, and 1024 bytes.

Two test configurations are employed (see Fig. 10): the first involves single-core execution to assess latency across multiple packet sizes; the second performs four concurrent, asynchronous transfers using 1024-byte packets.

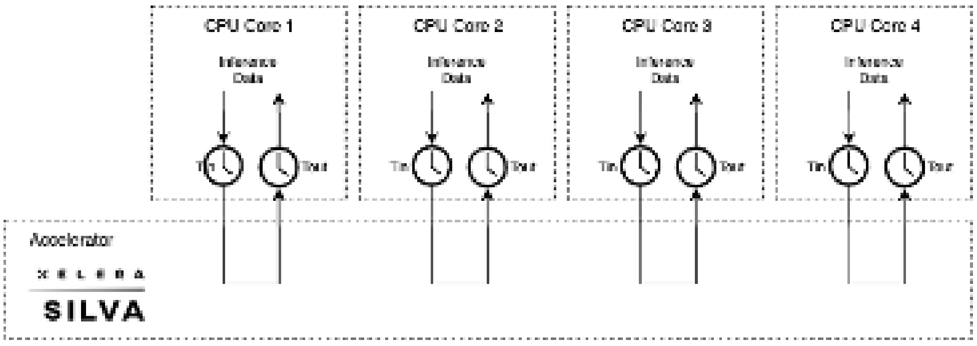

Fig. 10: PCIe data transfer test configuration

To ensure consistent and reliable measurements, each process is pinned to an isolated CPU core (cores 1 through 4). All tests are repeated 1,000,000 times to ensure statistical robustness.

7.2.1 Results: Single Transfer

Fig. 11 shows the latency statistics of single PCIe data transfer for multiple packet sizes using only CPU core 1. The graphs show the fraction of inference measurements (y-axis) below a specified latency (x-axis).

Table 14 compares the minimum, maximum, median (50^{th} percentile) and the 99^{th} percentile latency (microseconds) of the graph above.

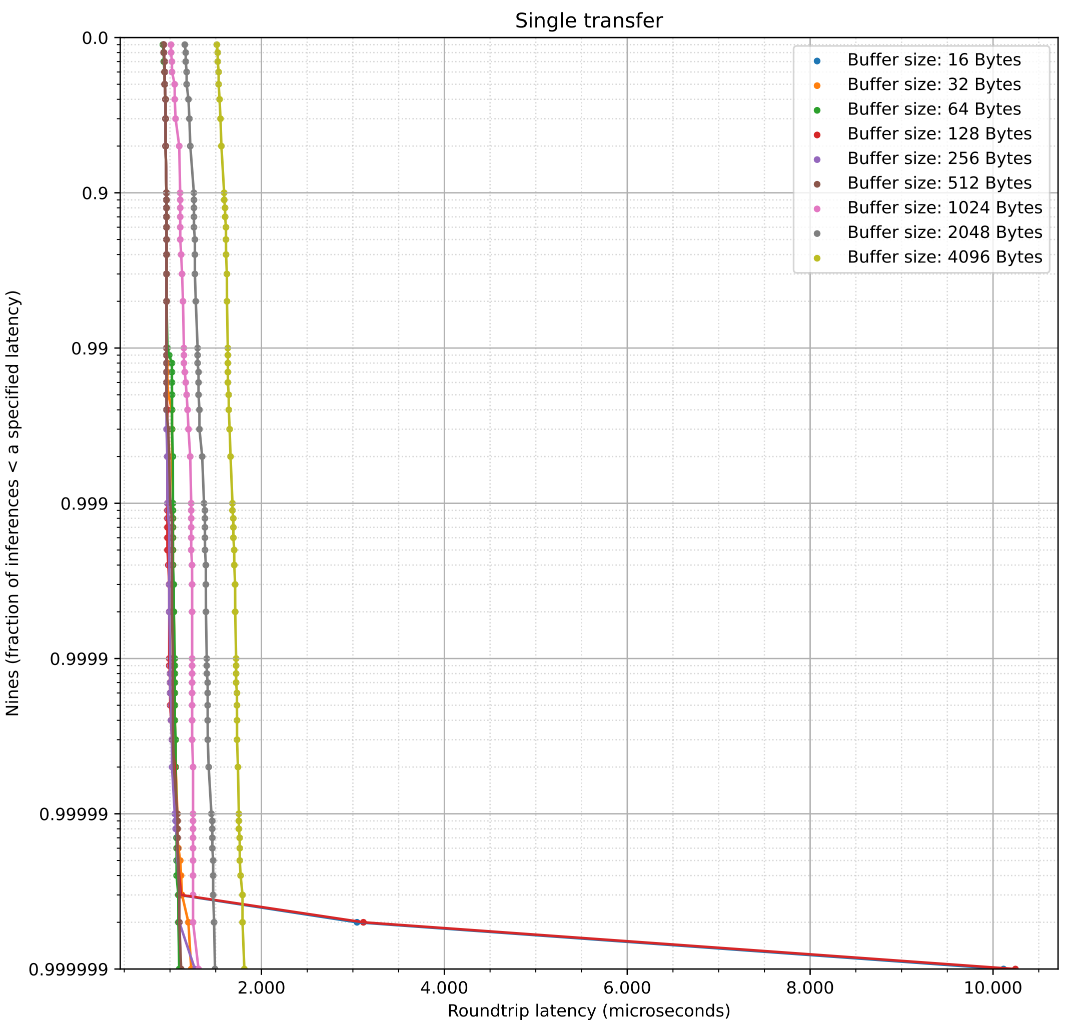

Fig. 11: Latency statistic for single PCIe data transfer

Table 14: Single transfer loopback latency (microseconds)

Transfer size [Bytes]	Minimum	50th percentile	99th percentile	Maximum
16	0.821	0.942	0.962	10.115
32	0.821	0.942	0.962	1.232
64	0.821	0.942	0.972	1.101
128	0.831	0.942	0.962	10.245
256	0.851	0.942	0.962	1.272
512	0.891	0.942	0.962	1.122
1024	1.001	1.051	1.152	1.312
2048	1.151	1.182	1.302	1.492
4096	1.472	1.532	1.632	1.813

7.2.2 Results: Parallel Transfers

Fig. 12 shows the latency statistics of 10 concurrent, asynchronous PCIe data transfers using 512-byte packets on CPU cores 1–4. The graphs show the fraction of inference measurements (y-axis) below a specified latency (x-axis).

Table 15 compares the minimum, maximum, median (50th percentile) and the 99th percentile latency (microseconds) of the graph above.

Table 15: Parallel transfer loopback latency for 10 concurrent 512-byte transfers (microseconds)

Transfer ID	Minimum	50th percentile	99th percentile	Maximum
0	0.881	1.012	1.232	1.543
1	0.881	1.012	1.232	1.533
2	0.891	1.032	1.242	1.572
3	0.891	1.032	1.252	1.553
4	0.901	1.022	1.232	1.543
5	0.941	1.042	1.252	3.705
6	0.971	1.081	1.332	3.997
7	0.951	1.132	1.362	2.214
8	0.951	1.051	1.232	2.124
9	0.951	1.042	1.232	2.413

7.2.3 Key Findings

The PCIe data transfer results establish that the host-to-accelerator data path imposes minimal and highly predictable latency overhead across the full range of payload sizes relevant to Gradient Boosting Tree inference.

For **single transfers**, the 99th percentile latency increases from **0.962 µs** at 16 bytes to **1.632 µs** at 4096 bytes — a growth of only 0.670 µs across a 256× increase in payload size. Below 1024 bytes, the 99th percentile remains at

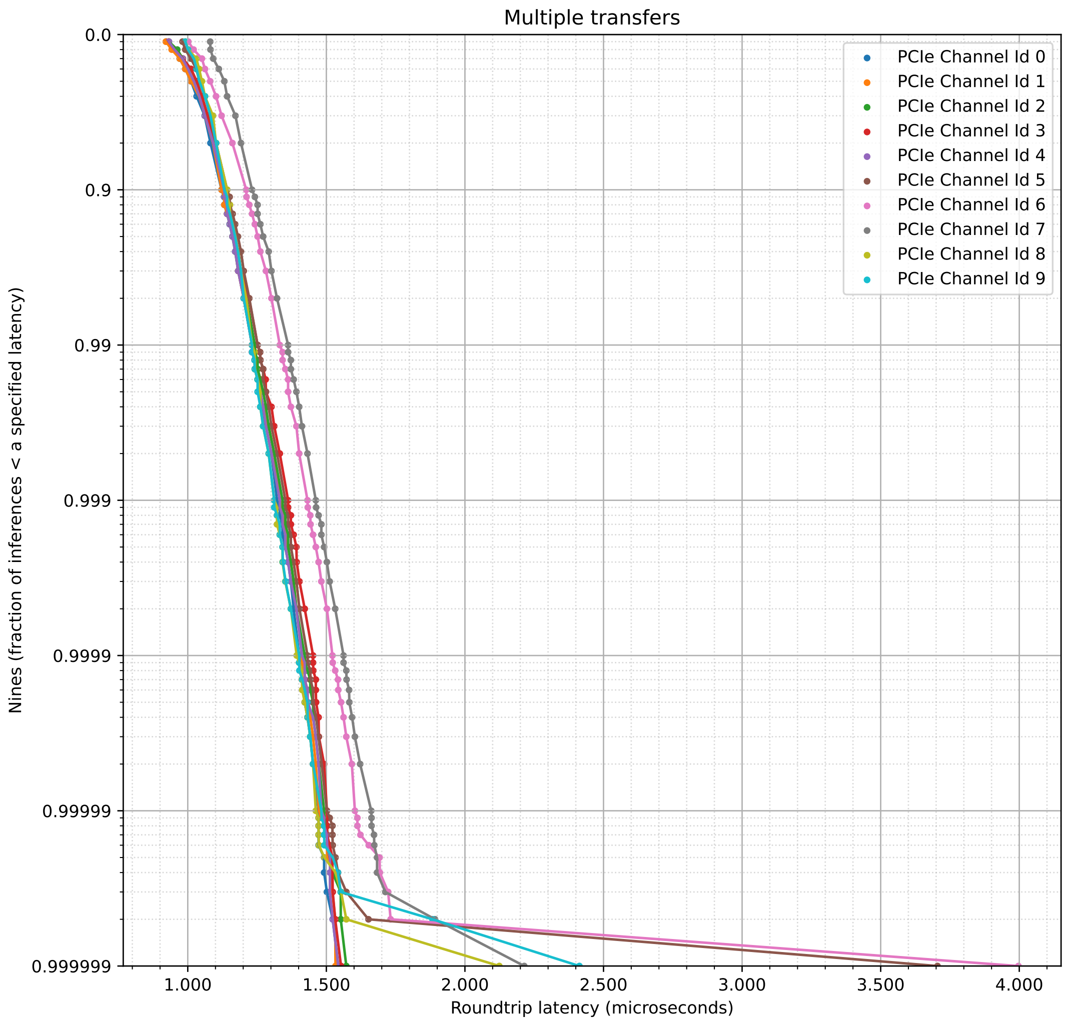

Fig. 12: Latency statistic for multiple parallel transfers

or below **1.152 µs**, confirming that for the feature vector sizes typical in high-frequency trading models (64–1024 features × 4 bytes), the transfer overhead is effectively constant and sub-microsecond at the 99th percentile.

For **10 concurrent parallel transfers** using 512-byte packets, the 99th percentile remains within **1.232 µs to 1.362 µs** across all 10 simultaneous transfer streams — a spread of only 0.130 µs. This tight clustering under full concurrency demonstrates that the Xelera PCIe ULL shell arbitrates multi-process access without introducing latency imbalance between competing processes, a critical property for trading systems where multiple strategy threads share the same accelerator card.

Maximum latency spikes (up to 3.997 µs on transfers 6–7) are isolated outliers and do not affect the 99th percentile, reinforcing that the tail behaviour of the data path is well-controlled under all tested load conditions.

8 References

[1]	Xelera Technologies, “Xelera Silva,” [Online]. Available: https://www.xelera.io/products/silva
[2]	Blackcore Technologies, “ACE 3100-TS+,” [Online]. Available: https://www.blackcoretech.com/product/ace-3100-ts-plus-overclocked-server