International Conference on Tropical Studies and Its Applications (ICTROPS)
Volume 5, (2026), e-ISSN : xxxx-xxxx
Website: http://e-journals2.unmul.ac.id/index.php/ictrops
Resilient Tropical Futures: Digital Transformation, Green Infrastructure, and Climate Adaptation

The Effect of Random-Access Memory Capacity on the Performance of FL Studio (Fruity Loops) on Laptops

Didit Suprihanto[1], Adi Pandu Wirawan[2], Ikhsan Yusuf[3*], Happy Nugroho[4], and Arif Harjanto[5]

[1,2,3,4,5]*Department of Electrical Engineering, Faculty of Engineering, Mulawarman University, Samarinda, Indonesia*

**Corresponding author Email: enforcer1408@gmail.com*

The manuscript was received on November 17th, 2025, revised on December 3rd, 2025, and accepted on January 05th, 2026, date of publication February 02nd, 2026

Abstract

This study discusses the effect of RAM capacity on FL Studio performance on laptop devices. This study was conducted by dividing 3 different RAM configurations (32GB, 16GB, & 8GB) to be tested one by one. This study conducted an artificial experiment by creating a music project and dividing it into two variations in the form of native and third-party software plugins to see the difference in workload and the RAM requirements for each project in FL Studio. Performance data was measured from three scenarios (load, playback, and render). It was found that the third-party project (based on a large sample library) that had been created required approximately 25GB of RAM to run optimally. With a 32GB RAM configuration, the render duration was 135.0s and the load duration was 113s. In the playback scenario, the project ran optimally and CPU usage was below 50%. With a 16GB configuration, playback was possible but experienced a bottleneck as seen from the SSD read activity (average) increasing to 16.8 MB/s. The rendering process was also successful, but slower at 154.3 seconds. In the 8GB configuration, there was a performance failure in the load scenario (305 seconds), making it impossible to proceed to the playback and real-time rendering scenarios. In contrast, the native project (synthesizer) required approximately 9.4GB of RAM with rendering in 67.3 seconds. This study concluded that RAM requirements affect projects with third-party plugins (especially those based on large sample libraries).

Kata Kunci: *FL Studio, I/O Bottleneck, Plugins, RAM, System Performance.*

1. Introduction

The development of digital music technology has influenced the way music is produced, recorded, and distributed. At the centre of this transformation is the emergence of Digital Audio Workstation (DAW) technology, a computer-based system that acts as a complete virtual studio for recording, editing, and producing music [1]. This type of software has also set new standards in modern music production due to its ability to create complex musical compositions without having to rely heavily on conventional instruments, which often require a greater investment. One example of a popular DAW that is often used in the industry is FL Studio [2].

As the complexity of a music project increases, users are often faced with technical challenges that can be quite troublesome. A professional music project can involve dozens to hundreds of sound samples, virtual instruments, and audio effects (layers), all of which are processed simultaneously and in real time. This heavy workload can affect limited computing resources. One of the most burdened components is Random Access Memory (RAM), which functions as a high-speed temporary workspace that is crucial for system continuity. When its capacity is insufficient to accommodate all project assets, the system will experience a drastic decline in performance. In practice, this memory shortage can cause very disruptive problems, such as system unresponsiveness (high I/O latency), crackling audio, and even total application crashes [3-5].

Furthermore, the workload of a project is not only determined by the number of elements, but also by the type of plugins used. Third-party plugins (from external developers) often require more resources than native plugins, due to differences in software architecture. This difference is an important factor in why two seemingly similar music projects can have very different performance loads [6].

This study specifically uses laptops as a testing platform. This choice is based on fundamental technical changes in the way modern music is produced. Technology that is now increasingly affordable has enabled the creative process to shift from traditional studios to flexible working environments [7]. Although software developers provide recommended system specifications, in reality, these guidelines often do not reflect the workload of complex music projects [8]. This research is important to fill this gap by providing quantitative data on the effect of RAM capacity on FL Studio performance.

2. Method

This study uses a quantitative approach with an experimental method. The research was carried out through a series of structured stages, including problem identification, literature study, methodology design, test implementation, data analysis, and conclusion drawing. All tests were conducted on a single laptop (AMD Ryzen 5 5600H, DDR4 RAM with 8GB/16GB/32GB configuration variations, 512GB NVMe SSD, Windows 11 Home) to maintain consistency in the hardware environment.

The research design was based on the principle of variable control. The study used two types of test projects (Native and Third-party) that were functionally comparable in terms of the number of components (tracks, instruments, effects) and identical MIDI structures. Testing was conducted based on three scenarios (loading, playing, exporting) on all RAM configurations. System performance data, including CPU, RAM, and storage, was recorded using HWiNFO64 monitoring software.

2.1. Hardware and Software Instruments

All testing was performed on a single laptop unit to ensure hardware consistency. Specifications are detailed in Table 1. The software used included Windows 11, FL Studio Signature Edition, and HWiNFO64 (v. 8.24) for hardware monitoring.

Table 1. Hardware specification

Component	***Specification***
Laptop Model	Acer Nitro AN515-45-R2NQ
CPU	AMD Ryzen 5 5600H (6 Core, 12 *Thread*)
RAM	*Config* 1: 8GB DDR4 3200Mhz (*Single Channel*) *Config* 2: 16GB DDR4 3200Mhz (*Single Channel*) *Config* 3: 32GB DDR4 3200Mhz (2x16GB, *Dual Channel*)
Storage	Kingston OM8PDP3512B-AA1 512GB M.2 PCIe Gen 3

2.2. Test Project Design

Two different FL Studio projects (FLP) were designed for this experiment: a project labelled "Native" based on the type of plugins used, and a project labelled "Third-Party." To ensure a valid comparison, both projects were made functionally equivalent, with identical MIDI compositions, number of tracks, audio samples, and mixer routing structures. The main difference is the plugins used. For example, the Native Project only uses built-in plugins (e.g., FL Keys, Fruity Reeverb 2). Meanwhile, the Third-Party Project simulates a professional workflow by using heavy external plugins (e.g., Garritan CFX Grand Piano, EastWest Spaces II Reverb) to perform equivalent functions. All other project parameters are controlled, such as a sample rate of 48000 Hz, a buffer length of 512 smp (11ms), and a total duration of 3 minutes 26 seconds.

2.3. Variables and Data Collection

The independent variables for this experiment are RAM Capacity, which was tested at three levels (8GB, 16GB, and 32GB), and Project Type (Native and Third-Party). The dependent variables measured to determine performance are Process Duration (seconds), CPU Usage (%), RAM Usage (MB), and Storage Read/Write (MB/s). Data was collected in three scenarios representing common workloads. The first scenario, Load, measured peak resource usage and the time required to open an FLP file until it was fully responsive. The second scenario, Playback, measures average resource usage during real-time playback of the entire project. The third scenario, Render, measures average resource usage and the time required to export the project to a WAV file. All tests were replicated three times to ensure data consistency.

3. Results and Discussion

Complete test results are presented in Table 2. The analysis focuses on answering the research questions. Testing was conducted on two types of projects across three RAM configurations and three workload scenarios.

Table 2. System Performance Test Results

Project Type	**Scenario**	**RAM Configuration (GB)**	**Metrics**				
			Duration (s)	**CPU (%)**	**RAM (MB)**	***Read* (MB/s)**	***Write* (MB/s)**
Third-party	*Load*	32	113,3	38,6	25.314	1096,2	38,8
		16	180,5	48,6	15.564	1269,1	572,2
		8	305,0	61,0	7.530	1283,0	855,6
	Playback	32	206,0	36,8	25.702	9,1	0,1
		16	206,0	36,6	14.865	16,8	3,5
		8	>300(Failed)	23,9	7.455	116,6	45,5
	Render	32	135,0	43,5	25.884	4,6	3,4
		16	154,3	42,6	13.561	20,9	9,4
		8	>300(Failed)	27,1	7.461	118,3	44,5
Native	*Load*	32	*35,3*	*25,2*	*9.347*	*0,3*	*3,4*
		16	*36,1*	*22,5*	*7.390*	*61,2*	*6,8*

Project Type	Scenario	RAM Configuration (GB)	Metrics				
			Duration (s)	CPU (%)	RAM (MB)	*Read* (MB/s)	*Write* (MB/s)
		8	*45,8*	*26,5*	*5.307*	*52,8*	*5,5*
	Playback	32	*206,0*	*17,7*	*9.423*	*0,1*	*0,3*
		16	*206,0*	*17,1*	*7.401*	*0,3*	*0,1*
		8	*206,0*	*18,1*	*5.342*	*0,5*	*0,1*
	Render	32	*67,3*	*42,0*	*9.411*	*0,0*	*1,9*
		16	*68,1*	*42,2*	*7.442*	*0,3*	*2,0*
		8	*85,4*	*43,3*	*5.402*	*5,1*	*3,3*

3.1. The Effect of RAM Capacity on Performance

RAM capacity has a significant impact on performance, especially in third-party projects.

1. With a 32GB RAM configuration, the system operates optimally. The 25GB RAM requirement for third-party projects is fully met. Storage activity during playback is not very significant (9.1 MB/s), and the rendering process takes 135.0 seconds.
2. With a 16GB RAM configuration, the system experiences severe I/O bottlenecks. The 16GB capacity is still insufficient, causing RAM saturation. This forces the system to rely heavily on the SSD to perform activities that should be done by RAM, increasing Playback read activity to 16.8 MB/s and Render read activity to 20.9 MB/s. As a result, the render duration increases to 154.3 seconds.
3. With an 8GB RAM configuration, the system has experienced total performance failure, the operating system has become unresponsive, and FL Studio can no longer be used properly. This condition is known as memory thrashing. The third-party project takes a long time to load all the data (305 seconds) and cannot be played or rendered optimally and in real-time. This is caused by extreme I/O bottlenecks (read >116 MB/s) and CPU starvation, where CPU utilization drops (27.1%) due to waiting for data from slow storage.

3.2. Comparison of Plugin Architectures (Native vs. Third-party)

The plugin architecture determines the type of bottleneck that occurs. Third-party projects (based on large sample libraries) require sufficient RAM and storage capacity. This project requires 25GB of RAM and generates peak storage activity of more than 1096.2 MB/s during load. Meanwhile, native projects (based on synthesizers) tend to be more CPU-dependent. This project is much lighter, requiring only about 9.4GB of RAM and minimal storage activity (0.3 MB/s for load). Despite low RAM usage, the CPU load during rendering is 42.0%, comparable to the Third-party project's 43.5% when RAM is sufficient.

This study also confirms the principle of diminishing returns. For Native projects, increasing RAM from 16GB to 32GB provides almost no performance benefits, as the render duration difference is insignificant (68.1 seconds for native vs. 67.3 seconds for third-party) because the RAM requirement of 9.4GB is already met. However, even in lightweight Native projects, a slight bottleneck occurs at the 8GB RAM configuration, with render times slowing to 85.4 seconds.

3.3. Analysis of Workload Scenario Characteristics

The three scenarios impose different performance loads on the system. In the load test graph, it can be seen that CPU & RAM usage increase slowly over time when loading virtual instrument sample data. In terms of CPU usage, the relative values fluctuate, but there is an increase as all instrument data is loaded into the project. In storage, there is a spike in read & write speeds, and at the same time, RAM usage begins to increase. After the first spike in storage read & write speeds, RAM tends to rise slowly, and when there is another spike in storage read & write speeds, RAM usage rises significantly again to a peak of 25313 MB.

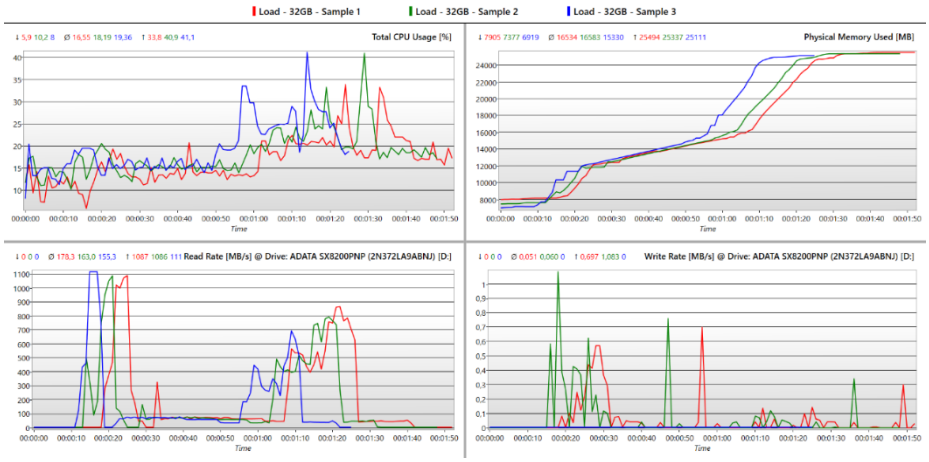

Fig 1. Load on 32GB RAM

In the playback test graph, CPU and RAM usage increased slowly over time and correlated with the increasing complexity of the instrument composition when played simultaneously. When playback on the track was complete or there was no longer any significant simultaneous

instrument activity, there was a decrease in CPU and RAM usage, similar to the beginning before the track was played. Storage shows an average read activity of 9.1 MB/s, which is relatively constant but fluctuates, indicating minor sample streaming. Meanwhile, there is no activity in the storage write speed.

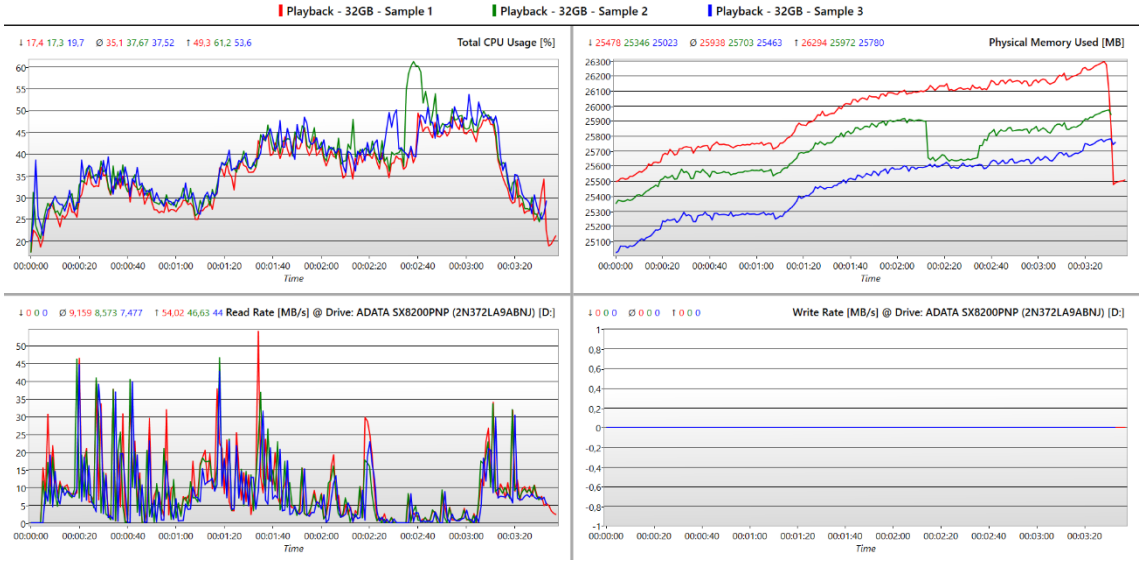

Fig *2.* Playback on 32GB RAM

On the render test graph, CPU and RAM usage are relatively more constant than in the playback test, with CPU usage tending to stabilize at 43.5% and RAM usage at 25884. In terms of storage, there is still an increase in read speed that is not as significant as in the playback test.

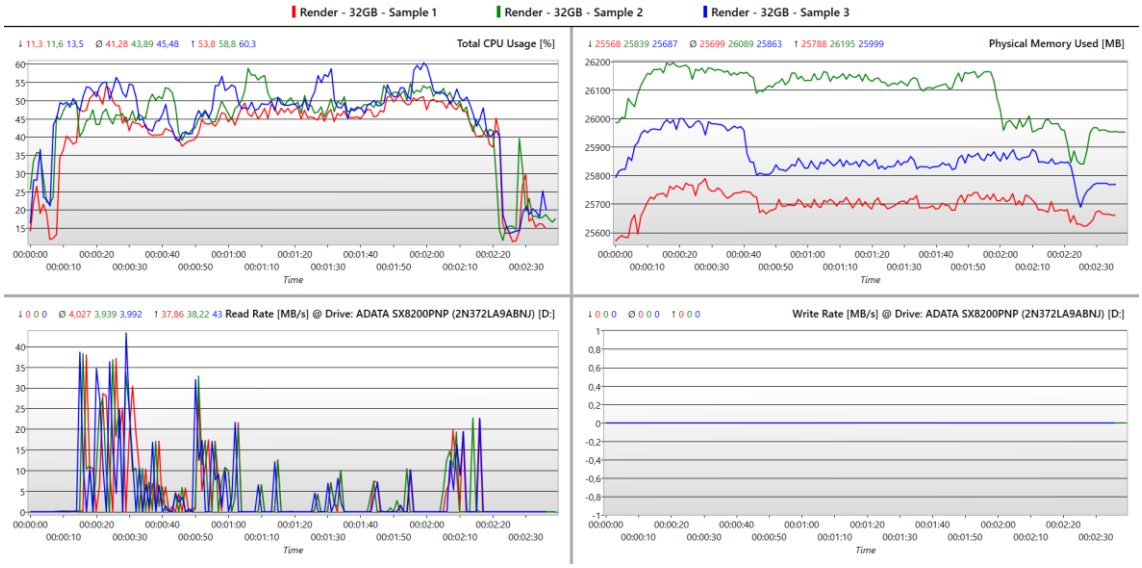

Fig 3. Render on 32GB RAM

4. Conclusion

RAM capacity has a significant impact on FL Studio performance. Insufficient RAM capacity (16GB and 8GB) when running large projects with heavy third-party plugins requiring 25GB of RAM can cause bottlenecks in storage I/O and CPU. With a 16GB RAM configuration, the bottleneck causes a longer render time of approximately 154.3 seconds. With 8GB RAM, the bottleneck is so extreme that it causes total performance failure in all test scenarios. The plugin architecture determines the type of bottleneck. Third-party projects (library-sample-based) are RAM and storage-intensive, requiring 25GB of RAM. Native projects (synthesizer-based) are more CPU-intensive, requiring only 9.4GB of RAM, but resulting in nearly identical CPU render loads. Adding RAM capacity shows diminishing returns. In this case, 16GB of RAM is optimal for running lightweight Native projects because increasing the RAM capacity to 32GB will not provide significant benefits, such as in rendering time, with a negligible difference (68.1 seconds at 16GB vs. 67.3 seconds at 32GB). However, 32GB of RAM is required as a baseline specification for heavy third-party projects to avoid significant bottlenecks.

Acknowledgments

The author would like to express gratitude to the Electrical Engineering Undergraduate Program and the Faculty of Engineering, Mulawarman University, for the support and guidance provided throughout this research.

References

[1] H. Wyatt and T. Amyes, *Audio Post Production for Television and Film*. Focal Press, 2005.
[2] Y. Yang, *Analysis Of Different Types of Digital Audio Workstations. Highlights in Science, Engineering and Technology* CSIC, 85, 2024.
[3] W. Stallings, *Computer Organization and Architecture: Designing for Performance (11th ed.).* Prentice Hall, 2022.
[4] Hennessy, J. L., & Patterson, D. A. (2019). *Computer Architecture: A Quantitative Approach. Morgan Kaufmann*
[5] Farah Humainah, R., Firmansyah, A. F. T., Ramadhani, S. I., & Didik Aribowo. (2023). *Pengaruh Kapasitas Memori RAM (Random Access Memory) Terhadap Kecepatan Memori Pada Laptop*. Jurnal Elektronika Dan Teknik Informatika Terapan (JENTIK), 1(4), 178–186. https://doi.org/10.59061/jentik.v1i4.458
[6] U. Zölzer, M. Holters, E. Gerat, P. Nowak, P. Bhattacharya, L. Köper, and D. Ahlers, *Digital Audio Signal Processing* (3rd ed.). Wiley, 2022.
[7] A. P. Bell, *Dawn of the DAW: The studio as musical instrument.* Oxford University Press, 2018.
[8] M. Richards and N. Ford, *Fundamentals of Software Architecture An Engineering Approach.* O'Reilly Media, Inc., 2020.