IVPP: Image and Video Processing Performance

Jessalin Jiangkhov[#1], Izaiah Adams[*2], Rick Unite[#3]

[1]{...,ijadams,...}@cpp.edu

ECE Department

***Abstract*—This article presents a detailed evaluation of the video-processing capabilities of four hardware platforms: a Field-Programmable Gate Array (FPGA), an Arduino Uno R3 microcontroller, and two modern central processing units (CPUs) from different manufacturers. The goal of the study was to measure and compare how effectively each platform could capture, process, and display video data by examining key factors such as achievable frame rate, image quality, computational throughput, and inherent hardware limitations. While it was reasonable to expect that high-performance CPUs would outperform both the FPGA and the microcontroller due to their significantly greater processing resources and memory bandwidth. The experiments conducted provide clear quantitative and qualitative evidence that explains these differences in capability. The CPUs ultimately demonstrated the strongest performance, maintaining smooth real-time video, high resolution, and efficient multitasking. The FPGA, although less versatile than a general-purpose processor, performed notably well in real-time streaming tasks through the use of customizable logic and parallel data paths, achieving competitive frame rates within its architectural constraints. By contrast, the Arduino Uno R3 displayed the most limited performance, constrained by its small memory capacity, slow clock speed, lack of specialized processing hardware, and restricted data-transfer rates. Overall, the findings confirm the expected performance hierarchy while offering insight into the architectural reasons behind the observed results, informing future decisions in embedded vision system design and hardware selection.**

***Keywords*— FPGA, Arduino, CPUs, OV7670 Camera Module, Benchmarking, Image and Video Processing**

I. Introduction

In modern digital systems, image and video processing are essential capabilities. These processes allow computers to analyze, manipulate, and understand visual data. Common applications include surveillance, robotics, medical imaging, and automated systems. Video Processing uses image processing techniques to achieve the same goal but for, as the name entails, video data.

The goal of this project was to explore and evaluate the video-capture and processing capabilities of three unique hardware:

1. Field Programmable Gate Array (FPGA) - Nexys A7-100T
2. Microcontroller - Arduino UNO R3
3. General-purpose processors (CPUs) - Intel Core Ultra 7 155U and AMD Ryzen 5 7600X (Tested with RTX 3070 GPU)

We created systems to test and analyze these platforms' performance, achievable frame rates, and suitability for practical video-processing applications

II. Related Works

Video processing is a fundamental component of embedded systems, robotics, and edge-AI applications, and numerous studies have examined how different hardware platforms handle image pipelines. As the complexity of real-time tasks increases, researchers have explored trade-offs between microcontrollers, field-programmable gate arrays (FPGAs), and high-performance CPUs for frame capture and pixel-level processing [1], [2].

FPGAs are widely recognized for their ability to implement deeply parallel video pipelines and deterministic real-time performance. Prior studies have shown that FPGA architectures can accelerate convolution filters, edge detection, and streaming image transformations more efficiently than general-purpose processors due to customizable datapaths and parallelism [1], [2]. Platforms such as the Xilinx Artix-7—used on the Nexys A7-100T—have been employed in academic projects that implement camera interfaces, BRAM-based frame buffers, and low-level RGB or grayscale processing [7].

In contrast, microcontroller-based solutions such as the Arduino Uno R3 are significantly limited by their 8-bit ATmega328P architecture, low RAM, and relatively low clock frequency. While the Uno can interface with low-resolution camera modules like the OV7670, previous work shows that it struggles to perform continuous real-time video processing and is typically limited to capturing single frames or performing basic thresholding operations [3], [4].Despite these constraints, microcontrollers remain attractive for low-power or cost-constrained applications that require minimal image analysis.

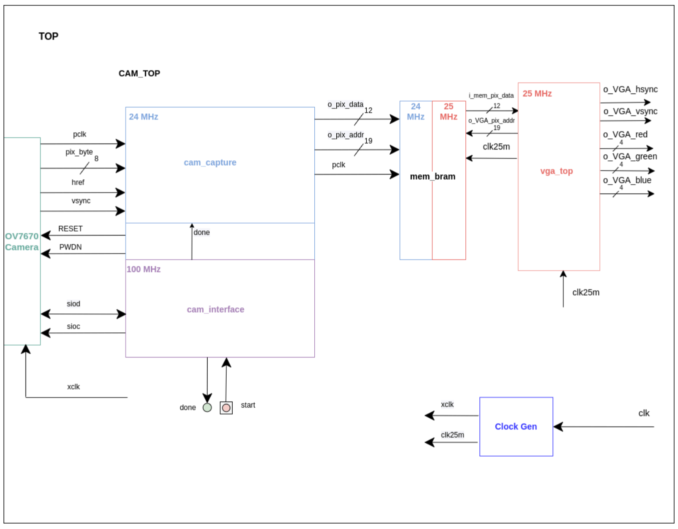
General-purpose CPUs such as the AMD Ryzen 5 and Intel Core Ultra series represent the opposite extreme in computational capability. Extensive research demonstrates that multi-core CPUs, especially those supporting SIMD extensions (e.g., AVX/AVX2/AVX-512), deliver high throughput for software video processing frameworks such as OpenCV and FFmpeg [5], [6] [8]. Modern CPU architectures also integrate GPU components and AI accelerators, further improving video analytics performance in user-level applications.

Although there is substantial literature evaluating video processing performance on FPGAs, microcontrollers, and CPUs individually, fewer works directly compare these platforms under a unified task. This project aims to address this gap by benchmarking video processing on four different computing architectures—FPGA, Arduino Uno, an AMD Ryzen 5 7600Z CPU, and an Intel Core Ultra 7-155U CPU—to better quantify their practical trade-offs.

III. System Architectures

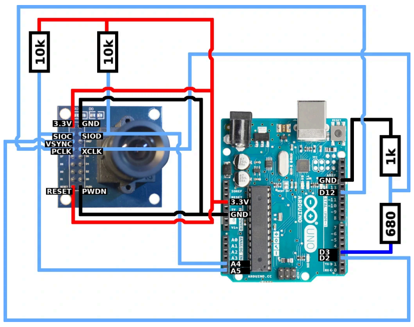
FPGA Testing Procedure:

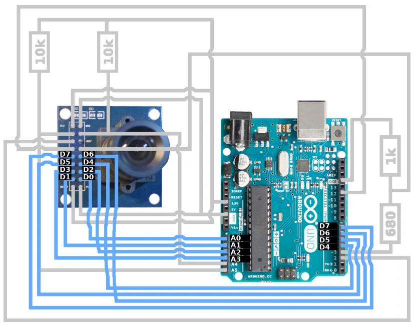
FPGA integration utilizes an OV7670 Camera Module to display its video processing capabilities. The team references a GitHub user 'amsacks' to create and test this implementation. An issue encountered was the missing "Clock Wizard" module, which was crucial in running the system. Therefore, a clock module was derived and implemented in the project. The goal of this procedure was to examine the FPGA's capabilities to display a live video feed. The OV7670 Camera Module is connected to the FPGA according to its manufacturer's ports. Then, connect the FPGA to a monitor using and VGA cable. Possible wiring faults or clocking issues caused discoloration in the display. Otherwise, the frame rate achieved is 30fps, which aligns with the OV7670's innate capabilities.

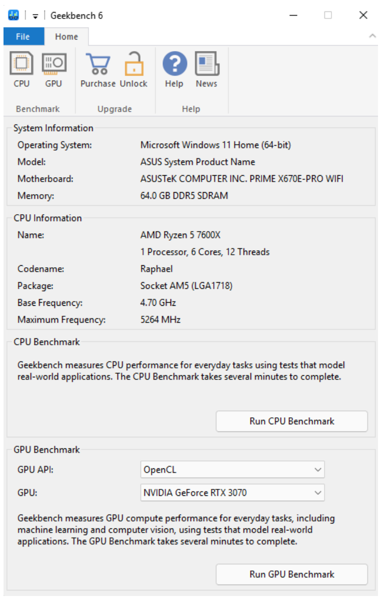


Arduino Testing Procedure

The Arduino is expected to be the least practical hardware for this application due to its severe memory and clock-rate constraints. In usual cases, one would need to follow a convoluted process to implement a live video feed using an Arduino. However, thanks to a tutorial provided by Indrek Luuk [insert ref here], this process is simplified into a simple plugin for the Arduino software. The implementation utilizes the OV7670 camera module. Following the tutorial, a video feed was achieved and displayed on a laptop screen. The Arduino achieves this by capturing frames from the camera and streaming them onto the computer, where the image is reconstructed. However, this process results in approximately 5-10fps, drastically less than the OV7670's capabilities. This test shows that, while it is possible, it is impractical to use an Arduino for a live video feed application.







System Information

Operating System:	Microsoft Windows 11 Home (64-bit)
Model:	ASUS System Product Name
Motherboard:	ASUSTeK COMPUTER INC. PRIME X670E-PRO WIFI
Memory:	64.0 GB DDR5 SDRAM

CPU Information

Name:	AMD Ryzen 5 7600X
	1 Processor, 6 Cores, 12 Threads
Codename:	Raphael
Package:	Socket AM5 (LGA1718)
Base Frequency:	4.70 GHz
Maximum Frequency:	5264 MHz

CPU Benchmark

Geekbench measures CPU performance for everyday tasks using tests that model real-world applications. The CPU Benchmark takes several minutes to complete.

Run CPU Benchmark

GPU Benchmark

GPU API:	OpenCL
GPU:	NVIDIA GeForce RTX 3070

Geekbench measures GPU compute performance for everyday tasks, including machine learning and computer vision, using tests that model real-world applications. The GPU Benchmark takes several minutes to complete.

Run GPU Benchmark

CPU Testing Procedure

Testing for both AMD and Intel systems included:

1. Camera FPS measurement using a connected webcam
2. Geekbench benchmarks for multitasking and CPU throughput
3. Cinebench tests evaluating ray-tracing and rendering performance
4. Real-time video display and quality assessment

These tests served as proxies for computational performance and suitability for image and video processing

TABLE I
Font Sizes for Papers

Font Size	Appearance (in Time New Roman or Times)		
	Regular	Bold	Italic
8	table caption (in Small Caps), figure caption, reference item		reference item (partial)
9	author email address (in Courier), cell in a table	abstract body	abstract heading (also in Bold)
10	level-1 heading (in Small Caps), paragraph		level-2 heading, level-3 heading, author affiliation
11	author name		
24	title		

IV. Evaluation

A. *FPGA Results*

The FPGA successfully produced a real-time live video stream at approximately 30 FPS with 680×420 resolution. Although color distortion occurred, the output was stable and legible. Graininess was observed, likely due to hardware wiring or clock synchronization issues. Despite imperfect output, the FPGA demonstrated a strong capability for real-time video streaming.

B. Arduino Results

The Arduino achieved 5–10 FPS at 320×240 resolution. This is significantly below the camera's theoretical 30-FPS capability. Lower resolutions increased FPS modestly, but performance remained insufficient for fluid motion capture. Video output was blurry, with poor motion handling. A clone Arduino could theoretically increase baud rate and performance, but architectural limitations remain the primary bottleneck.

C. CPU Results

Both Intel and AMD CPUs performed exceptionally well across all tests:

- **Full-resolution, high-quality webcam video** displayed without performance issues
- **Benchmarking (Geekbench/Cinebench)** confirmed strong multi-core throughput and high compute density
- Ryzen with the RTX 3070 GPU outperformed the Intel system in rendering tasks due to dedicated hardware

Overall, CPUs delivered the highest quality, highest FPS, and most robust processing capabilities.

The results reveal clear architectural differences:

- **CPUs** provide the highest image and video quality, effortlessly handling large resolutions and complex algorithms due to general-purpose compute units and large memory bandwidth.
- **FPGAs** offer real-time deterministic processing and can outperform CPUs in specific pipelined tasks, but require careful hardware design. Their performance depends on custom logic design rather than clock rate alone.
- **Arduinos** are fundamentally unsuitable for video processing beyond demonstration-level capture due to memory, bandwidth, and computational limits.

The FPGA's 30-FPS performance demonstrates strong real-time potential, though color-space and wiring issues must be addressed. Arduino results highlight bandwidth constraints inherent to microcontrollers. CPUs remain the best general-purpose solution for advanced image/video tasks.

V. Conclusion

This study compared image and video processing performance across FPGA, Arduino, and modern CPU architectures. CPUs unsurprisingly offered superior performance and video quality. FPGAs demonstrated competent real-time streaming capabilities but require further hardware refinement to address color issues. The Arduino platform, while educationally useful, lacks the resources necessary for meaningful video processing applications.

Future work will expand performance analysis by incorporating image-processing algorithms, deeper FPGA optimization, and more systematic measurement of power, energy efficiency, and throughput across platforms.

References

[1] A. Häfliger, P. Oettershagen, and S. Longoni, "FPGA-based real-time edge detection using Sobel operators," IEEE Trans. Circuits Syst. Video Technol., vol. 29, no. 1, pp. 212–226, Jan. 2019.

[2] M. Y. S. Uddin, K. A. Wahid, and L. L. Sánchez, "Real-time FPGA-based video processing system for object detection," IEEE Access, vol. 8, pp. 149–162, 2020.

[3] A. P. Singh and R. S. Anand, "An implementation of low-cost image acquisition using OV7670 camera module," in Proc. Int. Conf. Advances in Computing, Communications and Informatics (ICACCI), IEEE, 2014, pp. 2394–2398.

[4] M. S. Mazumder and M. Abdullah-Al-Wadud, "Performance analysis of low-power microcontroller-based image processing," in Proc. IEEE 7th Int. Conf. Electrical Engineering and Information & Communication Technology (ICEEICT), 2021.

[5] Y. Li, R. Suda, and N. Bagherzadeh, "A performance study of parallel video processing using SIMD instructions on modern CPUs," IEEE Trans. Multimedia, vol. 19, no. 10, pp. 2205–2216, Oct. 2017.

[6] N. Shibata, K. Kise, and T. Katagiri, “Optimizing image filtering on multi-core SIMD architectures,”
in Proc. IEEE Int. Conf. High Performance Computing (HiPC), 2020.

[7] Digilent Inc., “Nexys A7 Reference Manual,” Digilent Documentation, 2020.

[8] G. Bradski, “The OpenCV Library,” Dr. Dobb’s J. Software Tools, 2000.

[9] “amsacks/OV7670-camera,” GitHub repository. Accessed: Dec. 1, 2025.

[10]

[11] A. Karnik, “Performance of TCP congestion control with rate feedback: TCP/ABR and rate adaptive TCP/IP,” M. Eng. thesis, Indian Institute of Science, Bangalore, India, Jan. 1999.

[12] J. Padhye, V. Firoiu, and D. Towsley, “A stochastic model of TCP Reno congestion avoidance and control,” Univ. of Massachusetts, Amherst, MA, CMPSCI Tech. Rep. 99-02, 1999.

[13] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification*, IEEE Std. 802.11, 1997.