electronics

MDPI

Article

Mobile Video Surveillance Solution and Power Consumption Analysis for Mobile Surveillance Applications

Valer Bocan [1], Cristina-Sorina Stângaciu [1,*], Valentin Stângaciu [1], Flavius Opriţoiu [1], Bogdan-Alexandru Laziun [2] and Dan Leşeanu [3]

[1] Department of Computer and Information Technology, Politehnica University Timişoara, 2, Vasile Pârvan Bvd., 300223 Timişoara, Romania; valer.bocan@cs.upt.ro (V.B.); valentin.stangaciu@cs.upt.ro (V.S.); flavius.opritoiu@cs.upt.ro (F.O.)

[2] Faculty of Electronics, Telecommunications and Information Technologies, Politehnica University Timişoara, 2, Vasile Pârvan Bvd., 300223 Timişoara, Romania; bogdan-alexandru.laziun@student.upt.ro

[3] XTRATECRO SRL, Polonă Street, no. 23A, 300523 Timişoara, Romania

* Correspondence: cristina.stangaciu@cs.upt.ro

Abstract

This article proposes a video streaming solution together with a measurement setup for energy consumption analysis. The proposed solution, including three different types of hardware and their corresponding software configurations, has a direct application in remote mobile surveillance applications. The article presents the hardware and software architectures for each version, and an efficiency and power consumption analysis for different running scenarios and configurations. This analysis measures the energy efficiency in various scenarios, with the purpose of increasing the system's longevity and reducing operational costs.

Keywords: video surveillance; power consumption; embedded; internet of things

1. Introduction

Remote video surveillance solutions have seen important developments in the context of the Internet of Things [1–4], having extended their applications to fields such as smart cities [5,6], forestry [7] and agriculture [8], as well as being applied in the context of video super resolution applications, such as satellite surveillance [9,10].

In such remote monitoring applications, the energy efficiency of data transmission and the overall power consumption correlated with the time a system is available without human intervention are important aspects, as an energy-efficient design increases the system's longevity and reduces operational costs [11]. An energy-efficient design also leads to the increasing trend of mobile and autonomous solar-powered surveillance systems [12] and video surveillance via UAVs (unmanned aerial vehicles) [13].

A significant number of remote surveillance systems, especially those functioning in hardly accessible environments such as forests and fields, are powered by batteries or small solar cells. Moreover, even if the systems are powered from the grid, high power consumption is associated with high thermal dissipation, which has negative impacts on the system's functionality and its environment.

Furthermore, analyzing and evaluating the energy consumption of such mobile surveillance solutions in order to be able to make consumption predictions [14] has an important impact on the safety of the more complex systems they are integrated into, such as UAVs or autonomous remote surveillance systems. In these systems, anticipating the moment they

Academic Editors: Xiang Zhang and Huan Yan

Received: 19 December 2025
Revised: 10 February 2026
Accepted: 17 February 2026
Published: 26 February 2026

will run out of battery can highly increase their functionality and safety, because preventive measurements can be taken in time to recharge or replace the battery.

This article proposes a solution for remote video surveillance and a custom-made setup for the energy consumption analysis of the systems in different scenarios and configurations with the purpose of increasing systems' longevity and reducing operational costs.

The purpose of the proposed solution is, on one hand, to act as a reference point for an energy consumption analysis and, on the other hand, to help us investigate the following research questions:

- RQ1: *How can we design and evaluate a modular mobile surveillance solution?*
- RQ2: *What design choices reduce energy consumption and what are the main trade-offs of the proposed solution?*

The main contributions of this paper are as follows:

- The proposal and description of an accessible custom-made measurement setup together with a methodology for evaluating the energy efficiency of different mobile video surveillance solutions;
- The development of several reference mobile video surveillance solutions based on different versions of a Raspberry Pi;
- The collection and analysis of energy consumption data, providing reference values for energy efficiency analyses.

The paper is structured into five sections. Section 1 describes the motivation and the background of this research, Section 2 describes the proposed solution, Section 3 presents the measurement setup and methodology, and Section 4 presents and analyzes the results. The article ends with Section 5, comprising the conclusions and a discussion of future work.

2. Proposed Solution

We propose and analyze three different architectures of mobile surveillance system nodes. Two of them have a higher computation power and are based on the Raspberry Pi (RPI) 5 and 4B platforms. This choice is motivated by the high availability and popularity of Raspberry Pi platforms, supporting remote video surveillance applications [15]. The architectures act as references for performance measurements of the actual lower resource solution, which is based on Raspberry Pi zero 2W. As will be seen, the latter architecture has a lower computation power and a lower energy consumption.

The systems are composed of the following main functional blocks, as depicted in Figure 1:

- Signal acquisition;
- Processing and encoding;
- Data transmission.

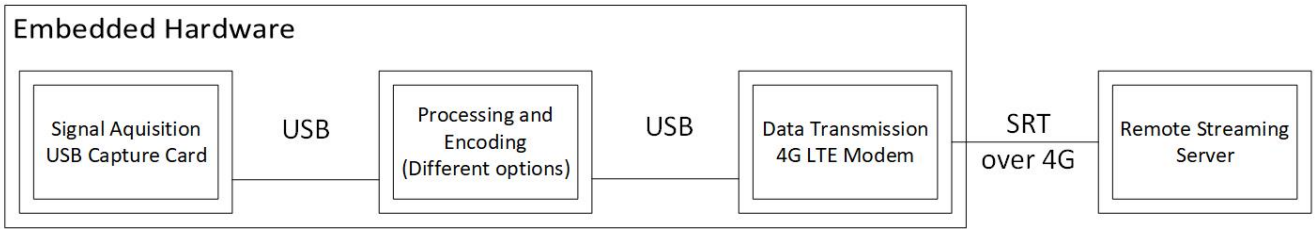


Figure 1. Main hardware components.

For conformity, the signal acquisition and data transmission are common across all the platforms.

The signal acquisition (USB Audio/Video Capture card) module converts an analog audio/video signal from an external source into a digital format that the Raspberry Pi can process, via its USB interface, and outputs raw, uncompressed or lightly compressed video frames and audio samples to the Raspberry Pi. The use of an analog audio video signal allows the camera and the microphone to be at a certain distance from the embedded hardware as presented in Figure 1. Thus, the system is able to transmit the analog signal between the camera and the capture card on coaxial cables with minimal loss, of up to several meters.

The data transmission module provides the high-speed mobile internet connection necessary to transmit the encoded video stream. The module connects to the compute platforms via a USB port and is recognized as a network interface. Two 4G LTE (long-term evolution) modems, SIM7600G-H and Sierra Wireless EM7305, were used for comparison.

This modular architecture allows for the replacement of the processing block, maintaining all the other components.

The general functional flow of the prototype is presented in Figure 2. In this figure, the dotted line represents the expansion of the Raspberry Pi processing and encoding block into its main components: capture and digitization; software or hardware encoding; RTS packaging; and the network interface.

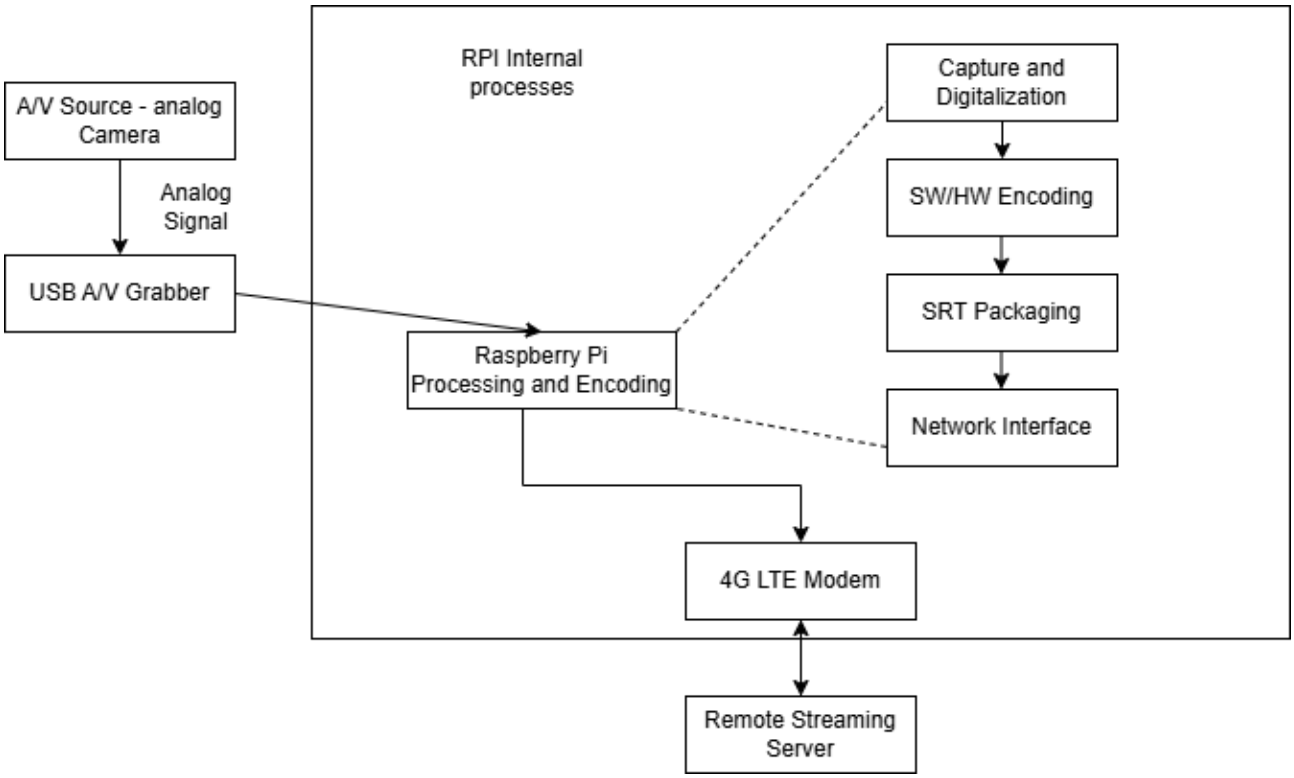


Figure 2. Functional flow.

For the Raspberry Pi 5 version, there is a software encoding, while for the other two versions there are dedicated hardware modules.

Although the three mobile platforms run the same operating system (Debian GNU/Linux v.13.3 (Trixie), Kernel: Linux 6.12.62+rpt-rpi-2712) and operate the same functional pipeline (USB audio and video capture, followed by H.264 and AAC encodings, MPEG-TS multiplexing, and publishing to a media server), the software stacks on the three platforms differ in terms of the manner in which they balance the encoding by hardware acceleration, in the way they handle audio and in how the network robustness is accounted for.

2.1. Version 1—Raspberry Pi 5 Solution

In the first proposed version, the video processing, encoding and streaming is performed by a Raspberry Pi 5 with its Broadcom BCM2712 quad-core Cortex-A76 processor. The system lacks dedicated hardware video encoding (H.264/H.265) support; therefore, the video stream must be encoded using software encoding running on the CPU. The high performance of the board's CPU is critical for maintaining an acceptable frame rate and resolution for live streaming (e.g. a resolution of 720 × 576, with a rate of 25 frames/s for PAL format). The RPI 5 processes the audio stream from the USB grabber, synchronizing it with the video stream and encoding it. The RPI 5 is responsible for packaging the encoded audio/video data into an SRT (Secure Reliable Transport) streaming protocol.

The prototype of the Raspberry Pi 5 version together with the setup for power consumption measurements is shown in Figure 3.

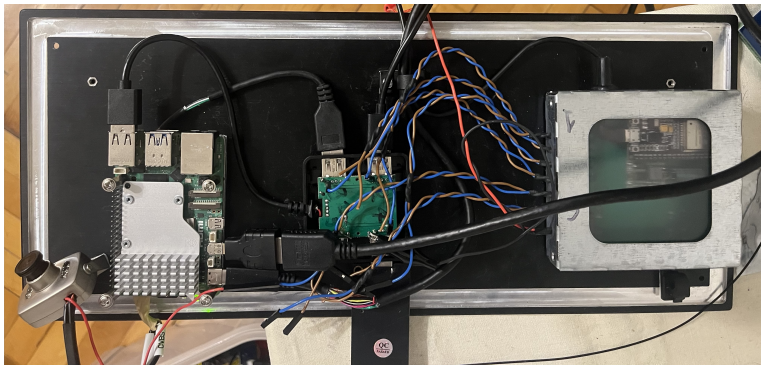


Figure 3. Proposed solution—version 1, based on Raspberry Pi 5.

The Raspberry Pi 5 software stack relies on software video encoding as the platform lacks the hardware support for it, facilitated by RPI 5's substantially higher CPU performance. This allows for greater encoding flexibility, while still maintaining a simpler, low-latency SRT setup. The overall RPI 5 exhibits a shift from resource conservation toward encoding quality and flexibility as the hardware capabilities increase.

2.2. Version 2—Raspberry Pi 4B Solution

In this version, the video processing, encoding and streaming is performed by the Raspberry Pi 4B, with Broadcom BCM2711, and a quad-core Cortex-A72 processor. The existence of a dedicated hardware encoder offloads the computationally intensive compression task from the CPU, allowing for higher quality and lower latency streaming with less power consumption.

The measurement setup together with the prototype, containing the Raspberry Pi 4B as a processing module, is depicted in Figure 4.

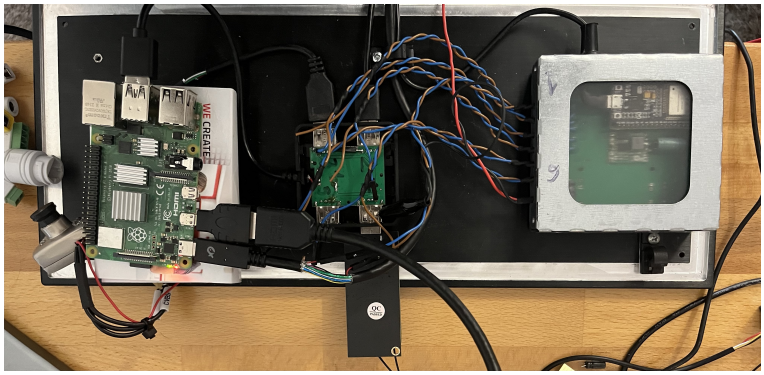


Figure 4. Proposed solution—version 2, based on Raspberry Pi 4B.

2.3. Version 3—Raspberry Pi Zero 2W Solution

In the last version, the video processing, encoding and streaming is performed by the Raspberry Pi zero 2W with the Broadcom BCM2710A1 quad-core Cortex-A53 processor. As in the previous case, the existence of a dedicated hardware encoder offloads the computationally intensive compression task from the CPU, allowing for higher quality and lower latency streaming with less power consumption.

The last version of the prototype is shown in Figure 5.

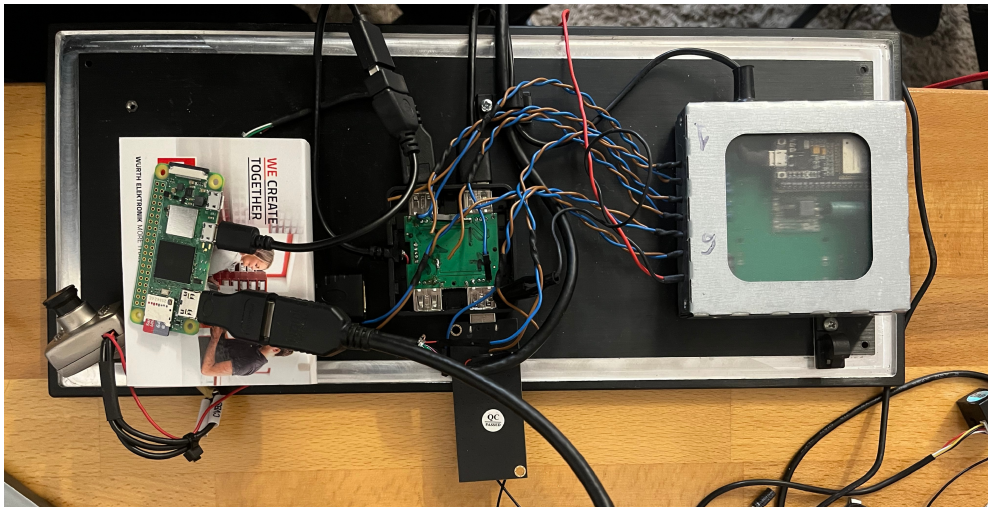


Figure 5. Proposed solution—version 3, based on Raspberry zero 2W.

On the top of the operating system runs a multimedia stack aimed at reliable, low-latency audio/video streaming over cellular networks. Media acquisition is handled by the Linux kernel through the Video4Linux2 (V4L2) and the Advanced Linux Sound Architecture (ALSA) subsystems, capturing uncompressed video from a USB UVC-compliant device and PCM audio from a USB audio interface. Video encoding is offloaded to the platform's internal hardware H.264 encoder via the V4L2 memory-to-memory (M2M) interface, significantly reducing the CPU load, while audio is encoded in AAC, using the platform's software stack, after a simple down-mixing stage. The encoded video and audio streams are multiplexed into an MPEG-TS container that best suits the continuous, real-time streaming requirements of the overall architecture.

On this board, the software components are tuned for a more constrained environment by relying on hardware H.264 encoding, offloading the CPU as much as possible, adding input buffering and audio channel remapping to compensate against USB timing jitters. On the same platform handling the link to the SRT media server involves encryption, forward error correction and a higher latency to adapt for 3G/4G connections. Overall, the software stack aims to optimize for low-power, constrained compute resources and unstable network conditions.

For all versions of the proposed solution, FFmpeg plays the role of the central orchestration component, interfacing with kernel drivers for capture. FFmpeg performs minimal signal processing and manages both the encoding and multiplexing. The resulting MPEG-TS stream is transmitted using the Secure Reliable Transport (SRT) protocol, which operates over UDP and exhibits encryption, packet loss recovery, and latency control. In particular, loss recovery and latency control are important features when operating over 3G/4G connections. Network connectivity is supplied by a USB cellular modem using standard Linux networking drivers, allowing the system to remain fully headless and self-contained. This software stack combines hardware-accelerated media processing with robust transport

mechanisms, making it well suited for embedded, unattended audio/video streaming in variable network conditions.

On the server side, the software architecture was designed to be modular and platform independent. It includes the following modules:

- A streaming server;
- Video processing;
- DVR (Digital Video Recorder) and archive storage system.

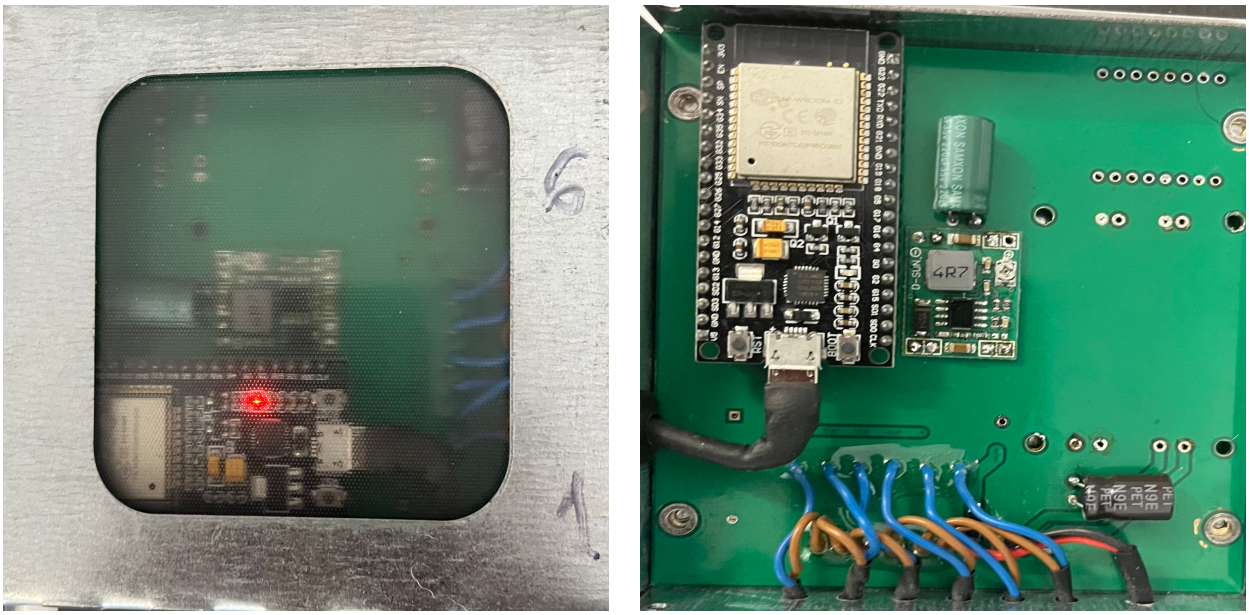
3. Measurement Setup and Methodology

In order to compare and analyze the previously described reference architectures, the three versions were considered using, in turn, each of the two communication modems and connection services from two different providers. A measurement-based energy profiling method was used [14].

3.1. Measurement Setup

The measurement setup consisted of the following main hardware modules:

- The measurement board (Figure 6): a custom board based on the ESP Development Module used for measuring currents from three different sources, Raspberry Pi, modem and capture card, by using three of six signal acquisition channels that were available;
- The target board: the Raspberry Pi-based boards, including the communication module and the capture card, each module being powered and measured on a different line.



(**a**) With cover (**b**) Without cover

Figure 6. Measurement board.

The measurement board is represented by a multichannel acquisition board, designed for measuring electric parameters (voltage and electric current) on each power line corresponding to each module of the target board. The architecture is based on an ESP32 microcontroller and six INA219-integrated circuits communicating on I2C.

The microcontroller is powered by an external source, but the ground is common to all modules. Each INA219 module measures the currents by using shunt resistors, while also monitoring the voltage on each channel.

An overview of the measurement board's architecture is presented in Figure 7.

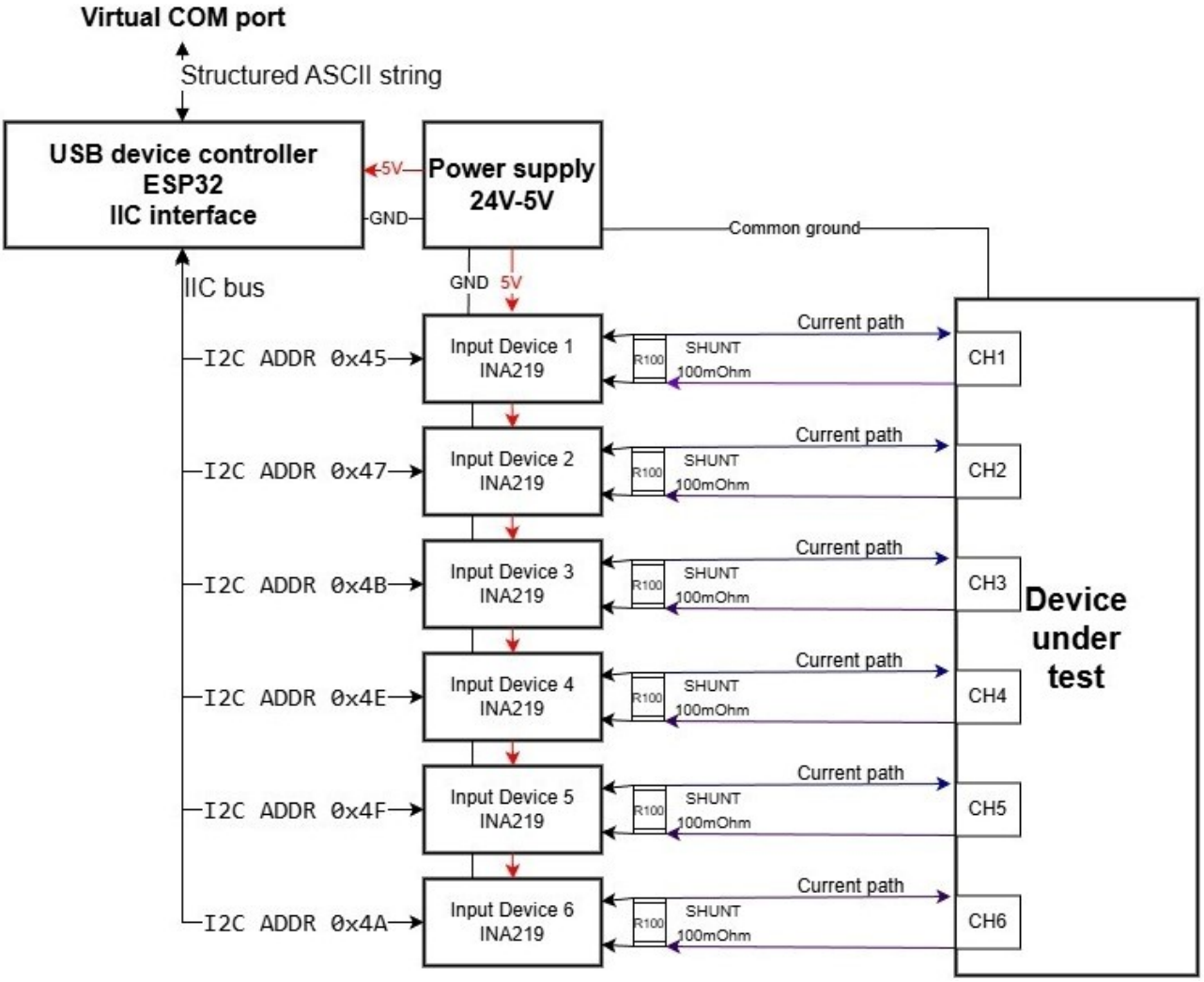


Figure 7. Measurement board architecture.

The sampling parameters are identical for all channels. Each INA219 is configured with a 16 V full-scale bus voltage range, Programmable Gain Amplifier (PGA) gain of eight (corresponding to a ±320 mV shunt voltage range), 12-bit analog to digital converter (ADC) resolution for both bus and shunt measurements, and 128-sample internal averaging enabled for both ADCs. The devices operate in continuous bus and shunt measurement mode, resulting in an effective integration time of approximately 68.1 ms per measurement.

The electrical current measurement is calibrated by comparison with a reference bench multimeter (Siglent SDM3065X-SC). After calibration, the expected measurement error is +/−0.5% of the reading (or +/−2 mA, whichever is greater).

After collecting the data on all the channels, the microcontroller performs some processing and sends the information on an USB port, configured as a Virtual COM Port, to a host PC, where it is stored and analyzed.

The communication flow of the analyzer is depicted in Figure 8.

In summary, the system allows for simultaneous measurement, with a relatively small error of the currents and voltage of a testing board, on different channels.

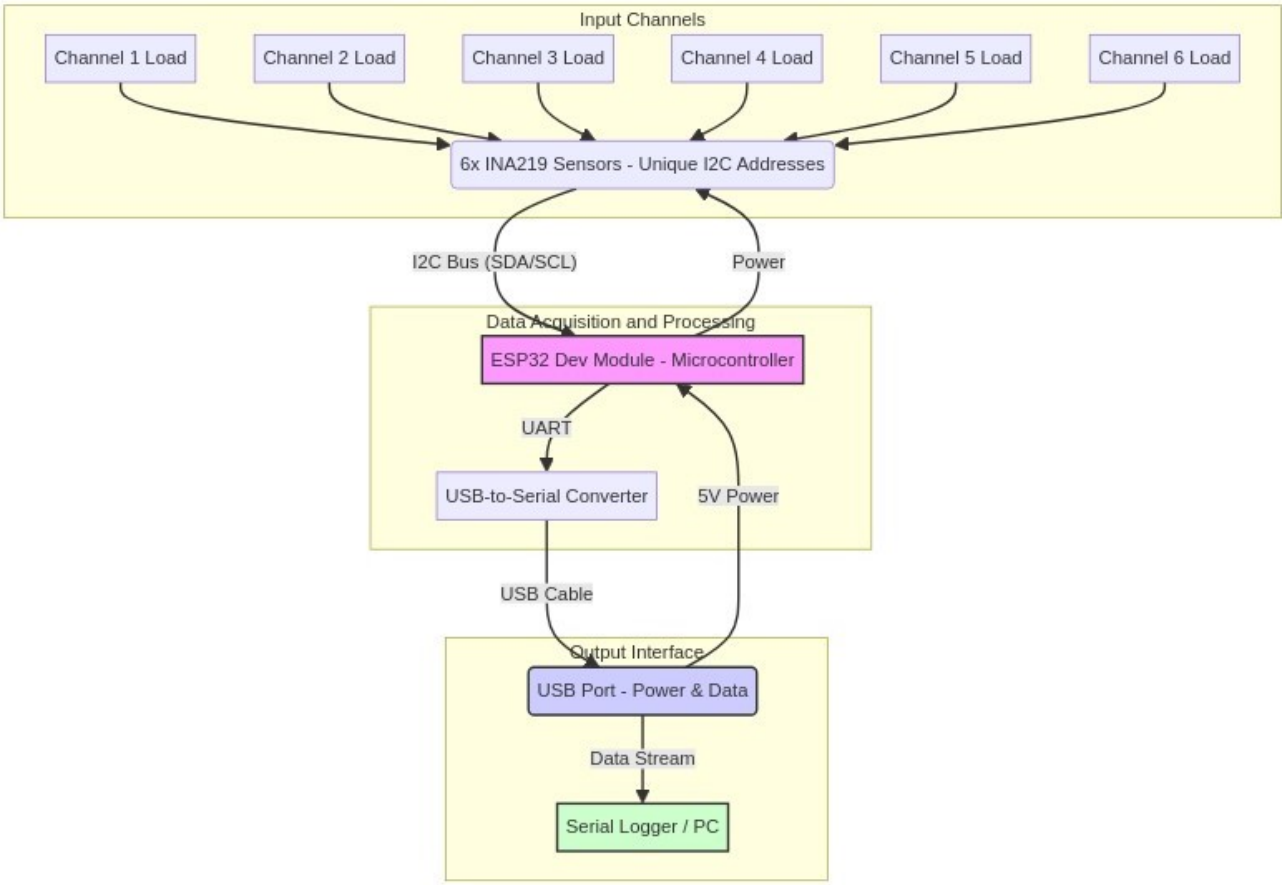


Figure 8. Analyzer communication flow.

3.2. Methodology

Measurements were performed regarding the performance, in terms of energy consumption and communication stability for uploading and downloading.

The specifications and measurement methods for each parameter were the following:

- Power measurements (idle/streaming) for each component, measured with the USB measurement board;
- CPU utilization, measured with top/htop;
- End-to-end latency—the difference between the display and capture time;
- Temperature, measured by the on-chip sensors.

The testing conditions consisted of a 4 G signal of exactly three lines (medium conditions) and an indoor temperature of 22–24 °C.

The measurement board was recalibrated for each target board, even if the conditions were similar (same temperature range; same location).

All INA219 devices were read sequentially, and the measured values were logged via the Universal Asynchronous Receiver/Transmitter (UART) protocol, together with an internally generated timestamp. The acquisition loop is repeated with a fixed period of 100 ms.

The streaming configuration for all scenarios is presented in Table 1.

Table 1. Streaming configurations.

Stream	Configurations
0	Codec: H264-MPEG-4 AVC
	Type: Video
	Video resolution: 720 × 576
	Buffer dimensions: 720 × 576
	Frame rate: 25
	Orientation: Top left
	Chroma location: Left
1	Codec: ADTS (AAC encoder)
	Type: Audio
	Channels: Mono
	Sample rate: 48,000 Hz
	Bits per sample: 32

In order to evaluate and compare the overall performance and power consumption of the modules involved in the proposed solutions we have defined several test cases, as follows:

1. Raspberry Pi 5 board architecture:
   (a) Sim7600G-H communication modem, configured as Remote Network Driver Interface Specification (RNDIS), on ORANGE provider.
   (b) Sim7600G-H communication modem (RNDIS) on DIGI provider.
   (c) Sierra Wireless EM7305 communication modem, configured as Mobile Broadband Interface Model (MBIM), on ORANGE provider.
   (d) Sierra Wireless EM7305 communication modem (MBIM) on DIGI provider.
2. Raspberry Pi 4B board architecture with Sierra Wireless EM7305 communication modem (MBIM), and ORANGE provider.
3. Raspberry Pi zero 2W board architecture with Sierra Wireless EM7305 communication modem (MBIM), and ORANGE provider.

For a more thorough analysis of the current consumption and comparison of the three boards, the following scenario was considered:

- Streaming: 720 × 576 pixels at 25 fps;
- Sierra Wireless EM7305;
- SRT streaming;
- H.264 encoding;
- Debian GNU/Linux v.13.3, Kernel: Linux 6.12.62 + rpt-rpi-2712;
- ffmpeg version 7.1.3-0 + deb13u1 + rpt1.

The video streaming bitrate is 2000 kbit/s, while the audio stream is encoded at 128 kbit/s (mono).

During transmission, additional bandwidth is required due to the SRT protocol overhead and forward error correction (FEC). As a result, the effective transmitted bitrates are approximately 1.84 Mb/s for video and approximately 137 kb/s for audio.

The output bitrate refers to the encoded media streams, while the transmitted bitrate includes the transport overhead and error correction data required for reliable streaming.

The configuration, for the previously mentioned scenario, is reflected in Listing 1:

Listing 1. Linux bash file.

```
#!/bin/bash
ffmpeg \
-f v4l2 \
-input_format yuyv422 \
-framerate 25 \
-video_size 720 × 576 \
-i /dev/video0 \
-f alsa -ac 1 -i hw:MS210x, 0 \
-c:a aac -b:a 128k -ac 1 \
-f mpegts ''srt://git.x-tec.ro:9000?mode=caller&latency=200
&passphrase=xtecxtecxtec&fct=12 × 12&overhead=25''
```

This listing was used for the actual testing of all the platforms. The only difference regarding the bash files used for each platform is that, for the Raspberry Pi 5, the following line was added (Listing 2):

Listing 2. Encoding library bash command.

```
-c:v libx264 -preset veryfast -b:v 4000k -pix_fmt yuv420p
```

The above line reflects the usage of software encoding. In the other two platforms the encoding is performed by dedicated hardware.

4. Results

4.1. Power Consumption and Temperature Dissipation Results

One of the most important criteria for mobile applications is the energy efficiency. The power consumption captures for the Raspberry Pi 5 for SRT streaming in each configuration are depicted in Figure 9 for scenario 1 (a), Figure 10 for scenario 1 (b), Figure 11 for scenario 1 (c) and Figure 12 scenario 1 (d), respectively. The captures show a variation not only between the modems, where the least consumption is obtained by the Sierra Wireless EM7305, but also between the communication networks.

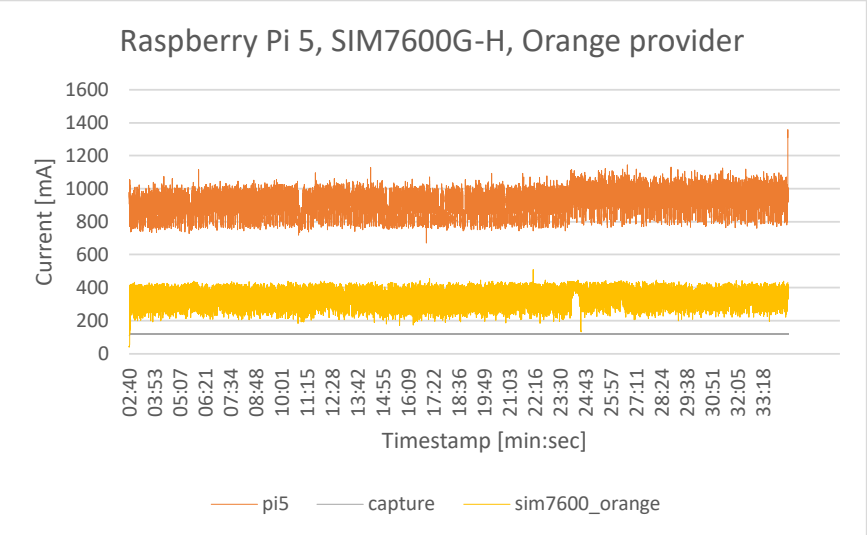


Figure 9. Power consumption for RPI 5 with Sim7600G-H on ORANGE.

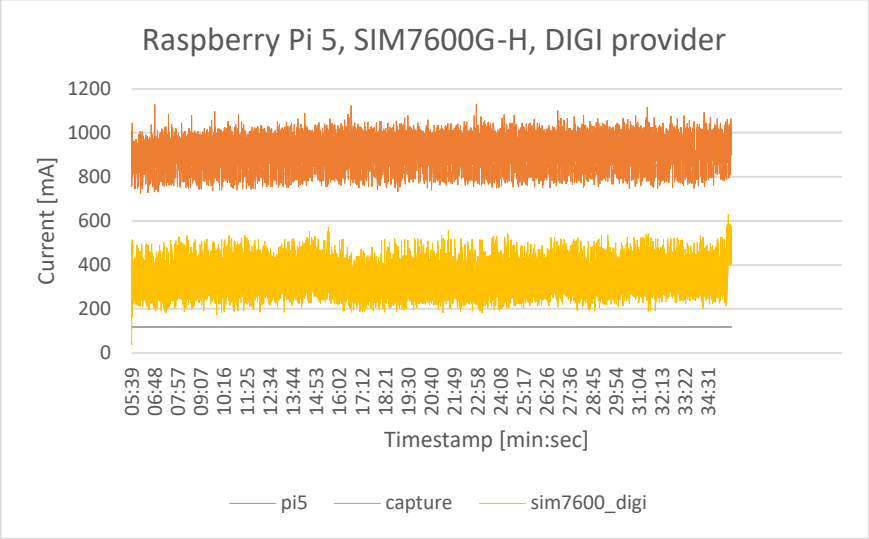


Figure 10. Power consumption for RPI 5 with Sim7600G-H on DIGI.

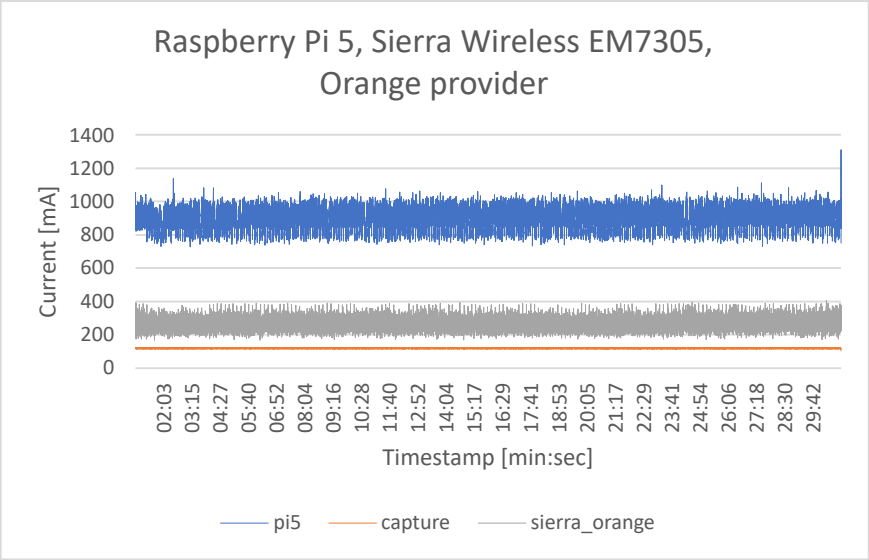


Figure 11. Power consumption for RPI 5 with Sierra Wireless EM7305 on ORANGE.

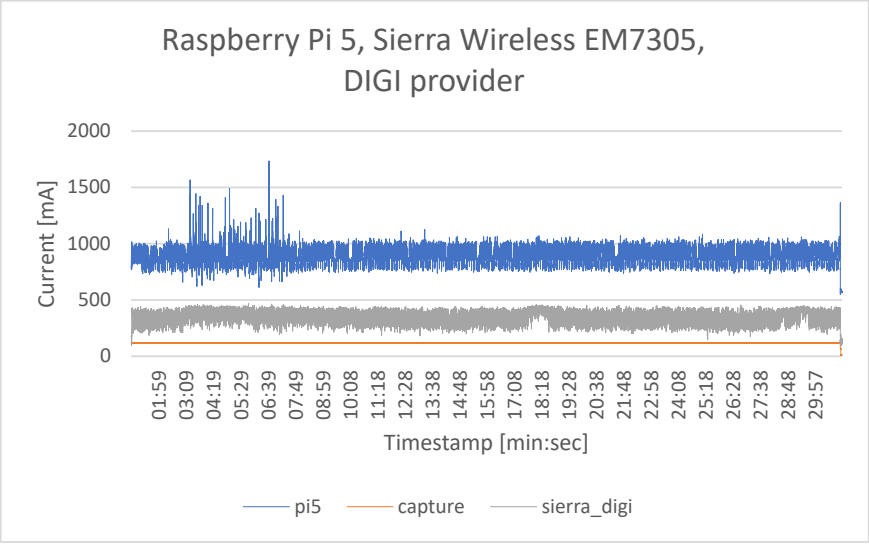


Figure 12. Power consumption for RPI 5 with Sierra Wireless EM7305 on DIGI.

Since we are mostly uploading, the upload speed of the two tested modems (Sierra Wireless EM7305-4G LTE CAT3 and SIM7600G-H-4 G LTE CAT4) is identical, but the power consumption differs, the CAT4 modem being more energy consuming compared to a CAT3 modem.

With regard to the communication modems, because less energy consumption is required using the Sierra Wireless EM7305 on the ORANGE network, the other scenarios (2 and 3) are based on this hardware configuration.

A comparison between the currents, measured in mA, is depicted in Figure 13. As it can be seen, the smallest values are obtained by Raspberry Pi Zero 2W, while the least variable consumption is obtained by Raspberry Pi 4B, an aspect reflected also in Table 2, where for each of the boards, the average, min, max and standard deviation of the current consumption are presented. The Raspberry Pi's current consumption varies according to the encoding volume, and the modem has consumption spikes due to the transmission burst; thus, its values fluctuate the most.

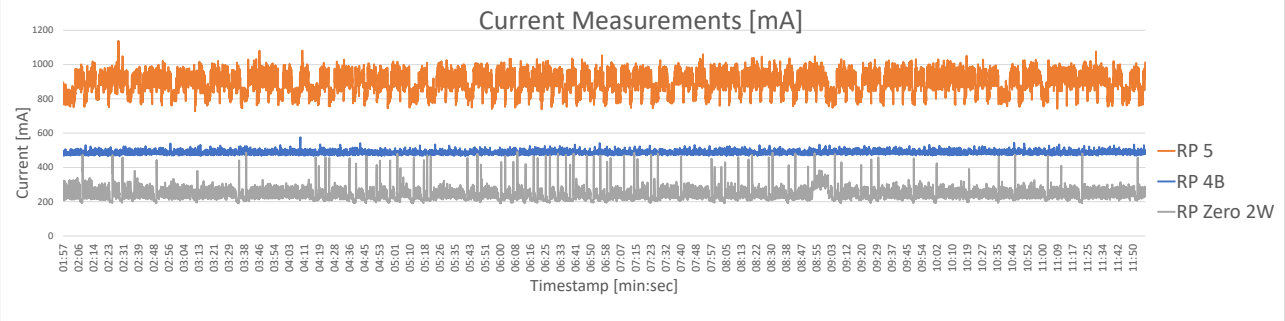


Figure 13. Current measurements in mA.

Table 2. Current consumption statistics.

Scenario	RPI	Average Modem	Capture Card	RPI	Min Modem	Capture Card
1.a	915.04	336.38	118.61	729.33	192.43	116.83
1.b	914.40	345.71	118.37	731.43	180.53	116.76
1.c	900.99	269.39	118.39	727.72	174.3	116.76
1.d	895.00	347.61	118.50	620.76	207.83	116.90
2	486.96	307.52	117.25	464.31	114.66	115.57
3	249.21	365.10	116.93	190.89	117.11	115.22
	RPI	**Max Modem**	**Capture Card**	**RPI**	**STD Modem**	**Capture Card**
1.a	1113.49	438.06	119.84	64.75	48.25	0.58
1.b	1128.68	532.56	119.56	58.57	63.09	0.54
1.c	1137.64	387.52	119.77	60.02	36.73	0.58
1.d	1564.78	456.68	119.91	67.38	50.62	0.60
2	576.38	457.38	118.79	10.95	50.87	0.51
3	503.30	476.98	118.3	38.09	73.32	0.52

A mean current consumption approximation for each component, powered at 5 V, is presented in Table 3. As it can be seen, in the most cases, the processor has the highest impact, followed by the modem and then the capture card.

Table 3. Mean current consumption approximation for each component at 5 V.

Platform	Processor	Modem	Capture Card
RPI 5	900 mA	270–347 mA	118 mA
RPI 4B	486 mA	307 mA	117 mA
RPI Zero 2W	249 mA	363 mA	117 mA

The average values of the power consumption of each solution and the main specifications are summarized in Table 4.

Table 4. Performance comparison table.

Characteristics	RPI 5	RPI 4B	RPI Zero 2W
CPU	A76–2.4 GHz	A72–1.5 GHz	A53–1 GHz
RAM	8 GB	4 GB	512 MB
Video Encoder	Software	H.264	H.264
Idle Power (W)	2.9	2	0.9
Peak Power (W)	7.8	2.8	2.5
Temperature (°C)	49	34	41

Regarding the CPU, a temperature dissipation analysis in correlation with the CPU usage is depicted in Figures 14–16, where measurements were taken for approximately a 30 min period, and where the Raspberry Pi Zero 2W received no cooling, while the other two had passive coolers.

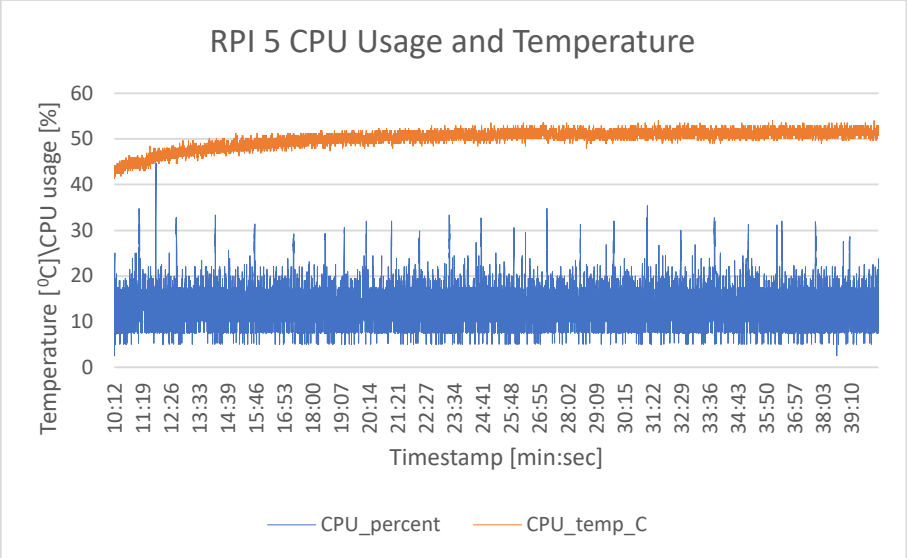


Figure 14. CPU temperature and CPU usage on Raspberry Pi 5.

A summary of the average temperature for each board's CPU is presented in Table 5.

Table 5. Temperature for each Raspberry Pi version.

Platform	Idle °C	Load Streaming °C	Stress Test °C
RPI 5	41	49	53
RPI 4B	29	34	37
RPI Zero 2W	34	41	43

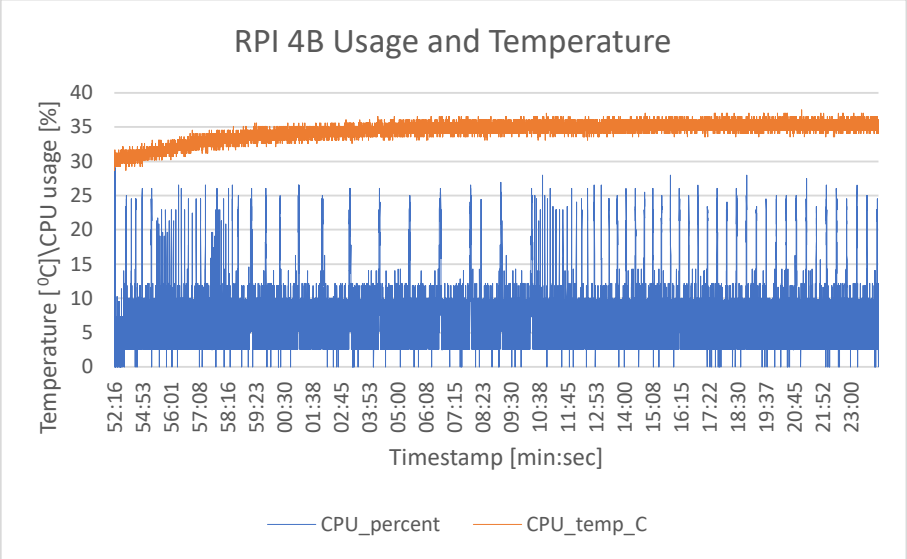


Figure 15. CPU temperature and CPU usage on Raspberry Pi 4B.

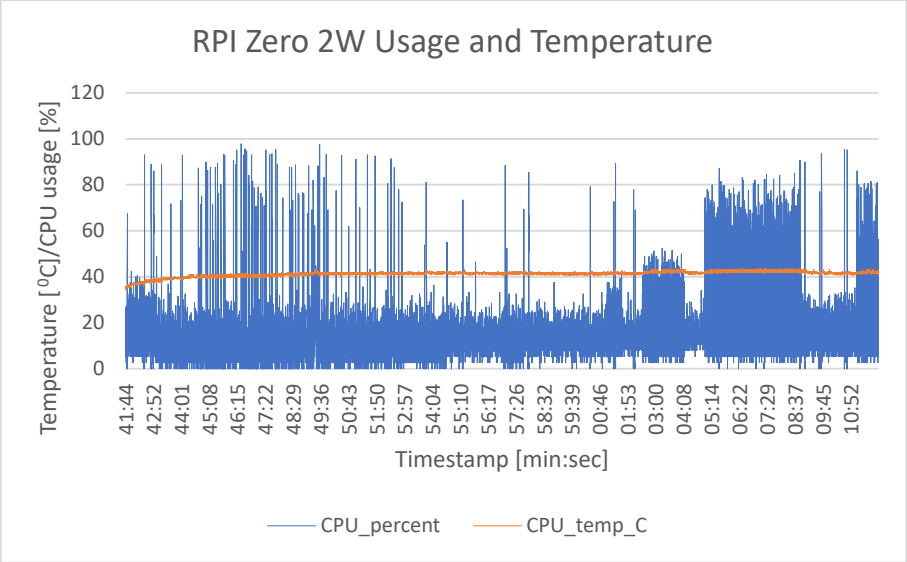


Figure 16. CPU temperature and CPU usage on Raspberry Pi Zero 2W.

4.2. Discussions

Figures 9–12 show how the current consumption, for the same Raspberry Pi 5 architecture, varies between modems, but also between networks. ORANGE performs slightly better in both cases. The Sim7600G-H modem shows a higher variation compared to Sierra Wireless EM7305, which has a lower energy consumption.

The research questions were addressed as follows:

- RQ1: *How can we design and evaluate a modular mobile surveillance solution?* A detailed hardware and software architecture was provided for both the solution and the measurement setup. The highlights are on the modularity of the proposed solution and on the separate power lines for each module, which make the measurement and evaluation easier.
- RQ2: *What design choices reduce the energy consumption and what are the main trade-offs of the proposed solution?* The component accounting for the largest share of total energy consumption is the processing module, followed by the modem and then the capture card (see Table 3). As can be seen from Figures 9–12, there is an obvious trade-off, on

one hand, between the power consumption measured and, on the other hand, the processing performance.

5. Conclusions

This article proposes a mobile surveillance solution and evaluates three different hardware and software architectures, providing detailed measurements for each relevant aspect in terms of power consumption.

Moreover, the proposed measurement setup together with the evaluation methodology can provide valuable tools and references for anyone interested in developing similar solutions and evaluating them.

Author Contributions: Conceptualization, V.B. and D.L.; methodology, D.L. and B.-A.L.; software, F.O. and V.B.; validation, C.-S.S., V.B., V.S. and B.-A.L.; investigation, B.-A.L.; resources, V.B.; data curation, C.-S.S. and V.S.; writing—original draft preparation, V.B.; writing—review and editing, C.-S.S.; project administration, V.B.; and funding acquisition, V.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by UEFISCDI, through project 6SoL(T6)/2024 (project code PN-IV-P6-6.3-SOL-2024-2-0265).

Data Availability Statement: Data available at https://data.upt.ro/handle/123456789/34.3 (accessed on 16 February 2026).

Acknowledgments: The authors gratefully acknowledge the support of the national funding authority, UEFISCDI, whose financial assistance through project 6SoL(T6)/2024 (project code PN-IV-P6-6.3-SOL-2024-2-0265) made this work possible.

Conflicts of Interest: Author Dan Leseanu was employed by the company XTRATECRO SRL, Timisoara. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

4G	Fourth Generation of mobile telecommunication
CPU	Central Processing Unit
LTE	Long-Term Evolution
MBIM	Mobile Broadband Interface Model
RPI	Raspberry Pi
RNDIS	Remote Network Driver Interface Specification
SRT	Secure Reliable Transport
UAV	Unmanned Aerial Vehicle

References

1. Muhammad, K.; Hussain, T.; Rodrigues, J.J.; Bellavista, P.; De Macedo, A.R.L.; De Albuquerque, V.H.C. Efficient and privacy preserving video transmission in 5G-enabled IoT surveillance networks: Current challenges and future directions. *IEEE Netw.* **2020**, *35*, 26–33. [CrossRef]
2. Kumar, P.P.; Pal, A.; Kant, K. Resource efficient edge computing infrastructure for video surveillance. *IEEE Trans. Sustain. Comput.* **2021**, *7*, 774–785. [CrossRef]
3. Matheen, M.; Sundar, S. IoT multimedia sensors for energy efficiency and security: A review of QoS aware and methods in wireless multimedia sensor networks. *Int. J. Wirel. Inf. Netw.* **2022**, *29*, 407–418. [CrossRef]
4. Ravindran, A.A. Internet-of-things edge computing systems for streaming video analytics: Trails behind and the paths ahead. *IoT* **2023**, *4*, 486–513. [CrossRef]

5. Jain, R.; Nagrath, P.; Thakur, N.; Saini, D.; Sharma, N.; Hemanth, D.J. Towards a smarter surveillance solution: The convergence of smart city and energy efficient unmanned aerial vehicle technologies. In *Development and Future of Internet of Drones (IoD): Insights, Trends and Road Ahead*; Springer: Cham, Switzerland, 2021; pp. 109–140.
6. Myagmar-Ochir, Y.; Kim, W. A Survey of Video Surveillance Systems in Smart City. *Electronics* **2023**, *12*, 3567. [CrossRef]
7. Kizilkaya, B.; Ever, E.; Yatbaz, H.Y.; Yazici, A. An effective forest fire detection framework using heterogeneous wireless multimedia sensor networks. *ACM Trans. Multimed. Comput. Commun. Appl. (TOMM)* **2022**, *18*, 47. [CrossRef]
8. Usmanova, N.; Mirzayev, D.; Ergashev, F.; Yunusova, D. Field monitoring application based on video surveillance: Evaluation of system performance. In *Proceedings of the E3S Web of Conferences*; EDP Sciences: Les Ulis, France, 2023; Volume 443, p. 06016.
9. Xiao, Y.; Yuan, Q.; Jiang, K.; Jin, X.; He, J.; Zhang, L.; Lin, C.W. Local-global temporal difference learning for satellite video super-resolution. *IEEE Trans. Circuits Syst. Video Technol.* **2023**, *34*, 2789–2802. [CrossRef]
10. Xiao, Y.; Yuan, Q.; Jiang, K.; Chen, Y.; Wang, S.; Lin, C.W. Multi-Axis Feature Diversity Enhancement for Remote Sensing Video Super-Resolution. *IEEE Trans. Image Process.* **2025**, *34*, 1766–1778. [CrossRef] [PubMed]
11. Ahmed, S.F.; Alam, M.S.B.; Afrin, S.; Rafa, S.J.; Taher, S.B.; Kabir, M.; Muyeen, S.; Gandomi, A.H. Toward a secure 5G-enabled internet of things: A survey on requirements, privacy, security, challenges, and opportunities. *IEEE Access* **2024**, *12*, 13125–13145. [CrossRef]
12. Izuka, U.; Bakare, A.; Olurin, J.; Ojo, G.; Lottu, O. Unlocking solar power for surveillance: A review of solar-powered CCTV and surveillance technologies. *Acta Electron. Malays.* **2023**, *7*, 54–61. [CrossRef]
13. Dilshad, N.; Hwang, J.; Song, J.; Sung, N. Applications and challenges in video surveillance via drone: A brief survey. In Proceedings of the IEEE 2020 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Republic of Korea, 21–23 October 2020; pp. 728–732.
14. Guo, C.; Ci, S.; Zhou, Y.; Yang, Y. A survey of energy consumption measurement in embedded systems. *IEEE Access* **2021**, *9*, 60516–60530. [CrossRef]
15. Mohamed, M.A.M.; Oniz, Y. Real-Time Long-Range Control of an Autonomous UAV Using 4G LTE Network. *Drones* **2025**, *9*, 812. [CrossRef]