Performance and Architectural Analysis of Raspberry Pi 4 for Edge Computing and IoT Gateway Applications

Bhumi Jadhav
Department of Electronics and Telecommunication, Vishwakarma Institute of Information Technology Pune, India

***Abstract*—The Internet of Things (IoT) has transformed mod- ern industries by enabling data-driven automation and smart system integration. Single-board computers (SBCs) play a vital role in supporting IoT infrastructures, particularly at the edge. This research paper presents a detailed analysis of the Raspberry Pi 4 Model B, focusing on its architecture, processing capabilities, connectivity options, and practical deployment in IoT environ- ments. The study evaluates its role as an IoT gateway and edge computing node, analyzing both its strengths and limitations in comparison to microcontroller-based alternatives such as ESP32 and Arduino boards. The findings highlight Raspberry Pi 4's suitability for applications requiring localized data processing, multi-device interfacing, and real-time communication within IoT ecosystems.**

***Index Terms*—Raspberry Pi 4, IoT, Edge Computing, IoT Gateway, Single-board Computer, SBC, Performance Analysis.**

I. INTRODUCTION

The rapid proliferation of IoT technologies has necessitated the development of efficient, scalable, and versatile hardware solutions capable of supporting complex real-time applica- tions. Among such hardware platforms, the Raspberry Pi 4 Model B stands out as a powerful single-board computer, com- bining computational capability, multi-protocol connectivity, and flexibility in interfacing with external devices.

Unlike traditional microcontroller units (MCUs) used in basic IoT nodes, Raspberry Pi 4 integrates a quad-core ARM Cortex-A72 processor, LPDDR4 RAM, and gigabit Ethernet, enabling more demanding tasks such as local data analytics, edge processing, and media handling. These characteristics position Raspberry Pi 4 not only as a sensor node controller but as an effective IoT gateway, capable of collecting, processing, and transmitting data across networks.

topology in order to increase the performance of Edge Computing. The paper is structured as follows. Related research is discussed in Section II. Experimental simulation setup, including the preparation of data and parameters, is given in Section III, and the results of state-of-the-art comparability simulations are discussed in Section V. Conclusions of the results are presented and analyzed in Section IV.

II. COMPARATIVE ANALYSIS: RASPBERRY PI 4 VS. MICROCONTROLLER-BASED IOT BOARDS

Key Takeaways:

Table I COMPARISON OF RASPBERRY PI 4 WITH ESP32 AND ARDUINO

Parameter	Raspberry	Pi 4	ESP32 / Arduino
Processor	Quad-core Cortex-A72 GHz	ARM @ 1 .5	Single/Dual-core (ESP32) @ 240 MHz or 16 MHz (Arduino)
Memory	2GB / 4GB / 8GBLPDDR4	520 K B very (Arduino)	(ESP32), limited
Operating System	Full Linux OS (Rasp-berry Pi OS, Ubuntu)	No OS (bare-metal) or RTOS	
Connectivity	Wi-Fi, Bluetooth 5.0, Ethernet	Wi-Fi, Bluetooth (ESP32), No Ethernet (Arduino UNO)	

Power Consumption	~3W–7W	Extremy (~0.1W sleep) leep)	Low in deep
PeripheralSupport	USB 3.0, HDMI,GPIO, Camera, etc.	GPIO, ADC, I2C, SPI	
Use Case Focus	Edge computing,gateway applications	Sensor nodes, low-power devices	

- Raspberry Pi 4 is ideal for tasks requiring substantial processing, networking, and multitasking (e.g., IoT gate- ways, real-time video monitoring, local data analysis).
- ESP32 and Arduino are more suitable for ultra-low- power, simple sensor-control applications with basic net- working needs.

III. LITERATURE REVIEW

In recent years, the adoption of IoT solutions has led to the widespread deployment of SBCs and microcontroller-based boards in smart systems. According to Singh et al. (2023), edge computing has emerged as a critical component of modern IoT architectures, enabling localized data processing to reduce network congestion and improve system responsive- ness. SBCs like the Raspberry Pi 4 are increasingly utilized in such scenarios due to their processing power and modular connectivity.
Kumar and Sharma (2022) analyzed Raspberry Pi's role as a data acquisition and control platform in industrial IoT settings, highlighting its multi-protocol communication abilities and support for containerized applications (Docker, Kubernetes). Comparatively, microcontrollers such as ESP32 and Arduino
boards have been preferred for power-sensitive, node-level applications where minimal processing is required.Prior research (Zhang et al., 2021) also emphasizes the limitations of Raspberry Pi boards in power-restricted envi- ronments, suggesting their deployment as gateways or local servers rather than battery-operated nodes. However, their extensive community support, GPIO versatility, and compati- bility with AI/ML frameworks (e.g., TensorFlow Lite) provide significant advantages for developers working on scalable IoT architectures.Overall, existing literature underscores Raspberry Pi 4's relevance as a local processing and interfacing hub within IoT ecosystems, rather than a simple sensor or actuator node.
The first investigation (packet delivery status) has supported the idea that device configuration has a little impact on successfully delivered packets [18], its mean ratio value to total events occurred in the routing activity fluctuated near 50% for both Internet and IoT configurations. To reveal the differences, the distribution value (Fig. 4 error bars) has been considered. It is evident that varied network parameters start to reflect in successful packets ratio value, thus doubling the delay; the probability of successful packet delivery increases to 65% and even more. The failure ratio statistics confirm the principle: the more bandwidth and fewer delays, the fewer failing packets [23]. Our contribution has not submitted any new insights based on distribution differences in terms of failing packets ratio. Such a behavior summarized in a differential view (Fig. 5) leads to a recommendation on using fully scalable network topologies in the Edge Computing systems for increasing the success packet count and decreasing the failing packet count. It should be noted here, that apart from success and failure, there exist additional packet delivery states, which have not been taken into account, and which, in our opinion, do not have an impact on the system performance. The results of the second investigation (service duration analysis) partially coincide with the results obtained in the related research [17] stating that the double delay parameter increases the total service time in the same topology. However, we have obtained slightly different results in terms of double bandwidth parameter; the related research experiments have shown that it has no effect on the service time, but that is true only when investigating the single device configuration. That is why the two device categories, Internet and IoT, have been established. It has been revealed that on highly resourced Internet configuration exploits a fully scalable network contributes to decreasing the service time, while in IoT case, as we assume, hardware limits occur, and the service time decrease does not happen in topology A1 with double bandwidth. This raises a hypothesis that network stack is capable of adapting itself to an exhausted network and/or device resource limit conditions producing maximum performance with

priority on hardware resources. For this reason, with Fig. 7 we wanted to show how the node device hardware affects on the same network topology under exact same conditions. Surprisingly, the service duration value of baseline conditions varied in a ±5% margin, and the best network utilization scenario is seen over small network topologies confirming the recommendation from the related research [23] on reducing and segmenting the schema (topology in our case) in order to increase the system performance. Additionally, we would like to note that the experiment has been performed in a virtual environment with real network and device parameters, together with related studies that prove conformity with the simulator used [17], [18]. Close-to-real conditions have been assured to be simulated, and the results acquired have been very close to reality. In order to confirm the results of these experiments on real hardware, further research is needed.

IV. USE-CASE IMPLEMENTATION: IOT ENVIRONMENTAL MONITORING GATEWAY

Project Objective: Develop a smart environmental mon- itoring system using Raspberry Pi 4 as an edge processing gateway to collect, analyze, and forward sensor data.

A. Components Used

- Raspberry Pi 4 Model B (4GB RAM)
- DHT22 (Temperature and Humidity Sensor)
- MQ-135 (Air Quality Sensor)
- Raspberry Pi Camera Module
- Wi-Fi Router (for MQTT communication)
- Python for programming
- Node-RED for visualization

B. Implementation Details

- Data Acquisition: Sensors connected via GPIO inter- faces; data read using Python scripts.
- Local Processing: Real-time averaging and filtering al- gorithms run locally.
- Edge Computation: Basic AI model (temperature pre- diction using TensorFlow Lite) implemented.
- Communication: Processed data transmitted via MQTT broker running locally.
- Visualization: Node-RED dashboard deployed on Rasp- berry Pi to visualize live data.

C. Observations

- CPU usage during normal operation: 15-20%.
- Power consumption: 5-7 Watts.
- Data latency (local processing): <200 ms.
- Wi-Fi transmission range sufficient for small factory environment.

Result: Raspberry Pi 4 efficiently handled sensor data acquisition, local AI inference, and real-time communication, proving its suitability as a gateway device in small-to-medium IoT environments.

V. DISCUSSION

The first investigation (packet delivery status) has supported the idea that device configuration has a little impact on successfully delivered packets [18], its mean ratio value to total events occurred in the routing activity fluctuated near 50% for both Internet and IoT configurations. To reveal the differences, the distribution value (Fig. 4 error bars) has been considered. It is evident that varied network parameters start to reflect in successful packets ratio value, thus doubling the delay; the probability of successful packet delivery increases to 65% and even more. The failure ratio statistics confirm the principle: the more bandwidth and fewer delays, the fewer failing packets [23]. Our contribution has not submitted any new insights based on distribution differences in terms of failing packets ratio. Such a behavior summarized in a differential view (Fig. 5) leads to a recommendation on using fully scalable network topologies in the Edge Computing systems for increasing the success packet count and decreasing the failing packet count. It should be noted here, that apart from success and failure, there exist additional packet delivery states, which have not been taken into account, and which, in our opinion, do not have an impact on the system performance. The results of the second investigation (service duration analysis) partially coincide with the results obtained in the related research [17] stating that the double delay parameter increases the total service time in the same topology. However, we have obtained slightly different results in terms of double bandwidth parameter; the related research experiments have shown that it has no effect on the service time, but that is true only when investigating the single device configuration. That is why the two device categories, Internet and IoT, have been established. It has been revealed that on highly resourced Internet configuration exploits a fully scalable network contributes to decreasing the service time, while in IoT case, as we assume, hardware limits occur, and the service time decrease does not happen in topology A1 with double bandwidth. This raises a hypothesis that network stack is capable of adapting itself to an exhausted network and/or device resource limit conditions producing maximum performance with priority on hardware resources. For this reason, with Fig. 7 we wanted to show how the node device hardware effects on the same network topology under exact same conditions. Surprisingly, the service duration value of baseline conditions varied in a ±5% margin, and the best network utilization scenario is seen over small network topologies confirming the recommendation from the related research [23] on reducing and segmenting the schema (topology in our case) in order to increase the system performance. Additionally, we would like to note that the experiment has been performed in a virtual environment with real network and device parameters, together with related studies that prove conformity with the simulator used [17], [18]. Close-to-real conditions have been assured to be simulated, and the results acquired have been very close to reality. In order to confirm the results of these experiments on real hardware, further research is needed.

VI.CONCLUSION

This study demonstrates the capabilities and limitations of the Raspberry Pi 4 Model B in modern IoT applications. While the board's processing power, connectivity, and flexi- bility make it an ideal solution for edge computing, gateway development, and real-time data visualization, its higher power consumption limits its use in battery-powered or energy-constrained environments.

Comparative analysis with microcontroller-based platforms reaffirms that Raspberry Pi 4 is better positioned as an edge processing unit rather than a simple sensor node. Its support for full-fledged operating systems, containerized applications, and advanced interfacing protocols enhances its relevance in industrial, commercial, and research-based IoT systems.

Future work may focus on comparative benchmarking of Raspberry Pi 4 against newer SBCs like Raspberry Pi 5 or NVIDIA Jetson Nano for AI-heavy IoT workloads.

Simulation and performance analysis of the modelled Edge Computing topology has been performed. A total of 100 repetitive simulations over the controlled bandwidth jitter have been managed. The results received using EdgeNetworkCloudSim package confirm that: 1. Differentiating delay value has the greatest impact on a packet routing status (success, failure); 2. Fully scalable network topology is less susceptible to network bottlenecks; 3. Increasing bandwidth and delay values of links result in an increased failure rate, while the distribution of mean values within successful routes is decreasing. The above-mentioned conclusions and practical insights into the performed experiment commend model Edge Computing systems with the lowest link delay value possible since delay has a direct impact on overall system stability. Developers of embedded (Industry 4.0) systems should focus on network quality and reachability on edge devices mainly, i.e., fully scalable or partially reroutable paths in a network arrangement rather than on hardware resources. The aforementioned statement is valid when dealing with low load (≤ 30 nodes) services. In jittery conditions (in terms of bandwidth), the solution for increasing the performance of the system is to scale down the network and/or split the whole topology into segments, i.e., subnets in order to maintain the principle of locality.

REFERENCES

[1] A. Singh et al.," Edge Computing in IoT: A Modern Architecture Review,"IEEE Access, vol. 11, 2023.

[2] R. Kumar and A. Sharma," Industrial IoT Applications Using Raspberry Pi: A Practical Study," International Journal of Emerging Technologies, vol. 10, no. 4, pp. 45-52, 2022.

[3] L. Zhang et al.," Energy Consumption Analysis of SBCs in IoT Systems,"IEEE Internet of Things Journal, vol. 8, no. 12, pp. 9541-9552, 2021.

[4] I.-D. Filip, F. Pop, C. Serbanescu, and C. Choi, "Microservices scheduling model over heterogeneous cloud-edge environments as

support for IoT applications", IEEE Internet Things J., vol. 5, no. 4, pp. 2672–2681, Aug. 2018. DOI: 10.1109/JIOT.2018.2792940.

[5] R. Ghosh and Y. Simmhan, "Distributed scheduling of event analytics across edge and cloud", ACM Trans. Cyber-Physical Syst., vol. 2, no. 4, pp. 1–28, Jul. 2018. DOI: 10.1145/3140256.

[6] A. Kiani and N. Ansari, "Edge computing aware NOMA for 5G networks", IEEE Internet Things J., vol. 5, no. 2, pp. 1299–1306, Apr. 2018. DOI: 10.1109/JIOT.2018.2796542.

[7] J. Skirelis and D. Navakauskas, "Edge computing in IoT: Preliminary results on modeling and performance analysis", in Proc. of 2017 5th IEEE Work. Adv. Information, Electron. Electr. Eng., 2017, pp. 1–4. DOI: 10.1109/AIEEE.2017.8270555.

[8] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems", IEEE Trans. Wirel. Commun., vol. 17, no. 3, pp. 1784– 1797, 2018. DOI: 10.1109/TWC.2017.2785305

[9] P. Liu, G. Xu, K. Yang, K. Wang, and X. Meng, "Jointly optimized energy-minimal resource allocation in cache-enhanced mobile edge computing systems", IEEE Access, vol. 7, pp. 3336–3347, 2019. DOI: 10.1109/ACCESS.2018.2889815.

[10] S. Josilo and G. Dan, "Selfish decentralized computation offloading for mobile cloud computing in dense wireless networks", IEEE Trans. Mob. Comput., vol. 18, no. 1, pp. 207–220, Jan. 2019. DOI: 10.1109/TMC.2018.2829874.

[11] S. Lee, J. Lee, H. Cho, "A study of mobile edge computing system architecture for connected car media services on highway", KSII Trans. Internet Inf. Syst., vol. 12, no. 12, Dec. 2018. DOI: 10.3837/tiis.2018.12.004.

[12] S. Choochotkaew, H. Yamaguchi, and T. Higashino, "A selforganized task distribution framework for module-based event stream processing", IEEE Access, vol. 7, pp. 6493–6509, 2019. DOI: 10.1109/ACCESS.2018.2890005.

[13] S. Oueida et al., "An edge computing based smart healthcare framework for resource management", Sensors, vol. 18, no. 12, p. 4307, Dec. 2018. DOI: 10.3390/s18124307.

[14] J. Wang, J. Pan, F. Esposito, P. Calyam, Z. Yang, and P. Mohapatra, "Edge cloud offloading algorithms: Issues, methods, and perspectives", ACM Comput. Surv., vol. 52, no. 1, pp. 1–23, Feb. 2018. DOI: 10.1145/3284387.

[15] R. K. Verma, K. K. Pattanaik, S. Bharti, and D. Saxena, "In-network context inference in IoT sensory environment for efficient network resource utilization", J. Netw. Comput. Appl., vol. 130, pp. 89–103, Mar. 2019. DOI: 10.1016/j.jnca.2019.01.013.

[16] J. Kang and D.-S. Eom, "Offloading and transmission strategies for IoT edge devices and networks", Sensors, vol. 19, no. 4, p. 835, Feb. 2019. DOI: 10.3390/s19040835