Performance Evaluation of Raspberry Pi as an IoT Edge Signal Processing Device for a Real-Time Flash Flood Forecasting System

Aslinda Hassan[1], Haniza Nahar[2], Wahidah Md Shah[3], Azlianor Abd-Aziz[4], Sarah Afiqah Sahiran[5], Nazrulazhar Bahaman[6], Mohd Riduan Ahmad[7], Isredza Rahmi A. Hamid[8], and Muhammad Abu Bakar Sidik[9]
Fakulti Teknologi Maklumat dan Komunikasi (FTMK), Universiti Teknikal Malaysia Melaka (UTeM), Jalan Hang Tuah Jaya, 76100 Durian Tunggal, Melaka, Malaysia[1,2,3,4,5,6]
Fakulti Kejuruteraan Elektronik dan Kejuruteraan Komputer (FKEKK), Universiti Teknikal Malaysia Melaka (UTeM), Jalan Hang Tuah Jaya, 76100 Durian Tunggal, Melaka, Malaysia[7]
Fakulti Sains Komputer Dan Teknologi Maklumat, Universiti Tun Hussein Onn (UTHM), 86400 Parit Raja, Batu Pahat Johor, Malaysia[8]
Department of Electrical Engineering, Faculty of Engineering, Universitas Sriwijaya (UNSRI), Palembang, Indonesia[9]

***Abstract*—The Raspberry Pi has evolved in recent years into a popular, low-cost, tiny computer for a wide range of IoT applications. Raspberry Pi is not only successful for data collection but also for data processing, including data storage and analysis. Thus, this study investigates the capability of Raspberry Pi as an edge processing device for capturing lightning strike signals in predicting flash flood locations. An electric and magnetic sensor (EMS) is connected to a Raspberry Pi in the experiment setup. The Raspberry Pi is then used to process digitised lightning signals. From the experiment, Raspberry Pi's performance is measured using the performance metrics: central processing unit (CPU) usage and temperature. The results revealed that the Raspberry Pi could handle the real-time collection and processing of lightning signals from the EMSs without affecting the hardware capability.**

Keywords*—*Raspberry Pi; IoT; edge; performance

I. Introduction

The Internet of Things (IoT) is changing how we live, work, travel and do business. It is also the cornerstone of a modern industrial revolution known as Industry 4.0 and the key to the digital transformation of businesses, communities, and society. An IoT ecosystem comprises web-enabled intelligent devices that use embedded systems, such as processors, sensors, and communication hardware. These intelligent devices collect, transmit and act on data they acquire from their environments. The devices then share the sensor data they acquire by connecting to an IoT gateway or other edge device, where the data is either transferred to the cloud for analysis or locally analysed. According to Priceconomics.com, the number of connected devices is projected to rise from 8.7 billion in 2012 to 50 billion in 2020 [1]. Huawei predicts that 100 billion connected devices will be used in every business and living area by 2025 [2]. Consequently, the data generated by the IoT is projected to reach 4.4 zettabytes by 2020 from just 0.1 zettabytes in 2013 [1].

The value of IoT goes further than data collection and real-time monitoring. Companies can gradually see the need to upload vast amounts of data to the cloud and support flexible resource management and visualised operations. They will also strive to process their data using machine learning and predictive analytics in order to introduce better technologies that will bring them success. Previously, placing all computational tasks on the cloud has proved to be an efficient way to process data since the power of cloud computing outperforms the capacity of the IoT. However, over the past few years, the significant increase of data generated by smart devices has put a strain on bandwidth utilisation [3].

Furthermore, digital traffic jams are almost anticipated, with the world estimated to generate up to 4.4 zettabytes of data by 2020. There is also the ”last mile” bottleneck problem. Essentially, the last mile defines the final networking segment, which connects an organisation's local network to the Internet. Since all network traffic destined for a particular organisation is channelled through that connection, it can be a bottleneck in networking throughput [4].

Due to the miniaturisation of processing and storage technology, current IoT devices have become more potent in collecting, storing, and processing data. This scenario has opened opportunities for organisations to optimise their networks and relocate more processing functions closer to where data is collected at the edge of the network. Gartner defines *edge computing* as a “part of a distributed computing topology where information processing is located close to the edge, where things and people produce or consume that information” [5]. In essence, edge computing brings computation and data storage closer to the smart devices rather than depending on a central location that might be thousands of kilometres away. Edge computing allows the data from the IoT devices to be analysed before being sent to the data centre. The main objective of edge computing is to prevent data, especially real-time data, from suffering latency issues that can affect the performance of an application [6].

Recent years have seen the development of Raspberry Pi as a popular, low-cost, tiny computer for several IoT applications. Raspberry Pi, referred to as a Single Board Computer (SBC), can run a complete operating system and has sufficient peripherals like memory, central processing unit (CPU), and power to initiate execution without additional hardware. In the present

TABLE I. TABLE OF ABBREVIATIONS

Abbreviation	Meaning
CAPPI	Constant Altitude Plan Position Indicator
CG	Cloud-to-Ground
CPU	Central Processing Unit
EMS	Electric and Magnetic Sensor
GPS	Global Positioning System
IoT	Internet of Things
LF	Low Frequency
LPDDR	Low-Power Double Data Rate
NBE	Narrow Bipolar Event
PaaS	Platform as a Service
RAM	Random Access Memory
RPi	Raspberry Pi
SDRAM	Synchronous Dynamic Random-Access Memory
SBC	Single Board Computer
VLF	Very Low Frequency

society, Raspberry Pi is not only an essential data-gathering device, but it can also analyse and store data in a server-like fashion.

In this study, Raspberry Pi devices are used as one of the components in the proposed architecture to illustrate its viability as an edge computing device and to evaluate its performance in terms of CPU system-wide utilisation and temperature. In this study, we believe Raspberry Pi devices may function as data processing edge devices, given their current CPU, memory, and storage capacity. If Raspberry Pi devices merely function as data collectors without additional processing and transfer all data to the cloud, then a significant amount of computer power is squandered. Consequently, in this study, our aim is not only to cut data transfer time by using Raspberry Pi's local processing capacity but also efficiently uses the otherwise underused distributed computing power. This study's contribution is to reveal the viability and potential of edge computing by doing hands-on experiments and analysing the performance in terms of CPU usage and temperature.

The rest of the paper is organised as follows. An overview of the Raspberry Pi as an edge device and similar studies on Raspberry Pi's performance as an edge device is discussed in Section II. Section III presents this study's background works. The experiment setup for the performance evaluation is presented in Section IV. Section V reports and discusses the results of the experimental evaluation. We conclude the paper in Section VI.

II. RELATED WORKS

On March 14, 2018, the Raspberry Pi Foundation introduced the Raspberry Pi 3 Model B+ as an upgrade to its predecessor [7]. It is also known as a Single Board Computer (SBC) with a Linux-based operating system, and a micro SD card is often required for boot file system storage. General Purpose Input-Output (GPIO) connectors are meant to control a vast array of electrical components in addition to being a small, low-cost computer programmatically[7]. In addition to its usage in education, it has become a popular edge processing device for other IoT applications.

The research articles in [8], [9], [10], [11] are examples of the existing literature on edge computing research utilising Raspberry Pi. The authors of [8] have created a lightweight edge computing-based distributed system that uses Raspberry Pi to directly handle the raw picture data from each camera. Consequently, the identified emotions may be easily communicated to the end user. In this study, the optimised and bespoke algorithms in the edge devices increase data processing speed, reduce network bandwidth requirements, and enhance application performance. The authors of [9] propose an automated service and resource discovery technique to effectively deploy nano services on local IoT nodes. Using a scenario involving remote healthcare monitoring, the authors offer a Raspberry Pi platform prototype implementation of the suggested method.

Several works in [12], [13], [14], [15], [16], [17] are related to performance evaluation on Raspberry Pi as edge devices. A study in [12] comparing data processing in various network setups and data stream speeds has been done in semantic data enrichment. It implemented a tiered IoT-Edge-Cloud system employing automotive sensor data at the IoT layer, Raspberry Pi at the edge layer, and Node-Red server at the cloud level in real architecture. The outcome demonstrates that data processing at the edge layer has enhanced efficiency, memory utilisation, and round-trip time.

Deep learning-based voice recognition applications have been studied in [13] to compare the performance and efficiency of Raspberry Pi with Nvidia Jetson Nano edge devices. Even if the Nvidia Jetson Nano is superior to Raspberry Pi, the word error rate for real-time inference on the Raspberry Pi CPU slightly degrades. The word error rate on the edge layer is generally more significant than the server inference, although it is not too far behind.

The objective of [14] is to conduct a comparative and experimental investigation of the performance of five distinct Raspberry Pi models (RPi Zero W, RPi Zero 2 W, RPi 3B, RPi 3B+, and RPi 4B) under a variety of situations and configurations. In conclusion, RPi 4B is significantly surpassed by competitors. In the meanwhile, the performance of the RPi Zero 2 W, RPi 3B, and RPi 3B+ is comparable, and the RPi Zero W is suggested for applications with minimal CPU and RAM capacity.

In [16], the authors demonstrate how to implement a cloud Platform as a Service (PaaS) architecture on a Raspberry Pi cluster. The main goal of the implementation is to make the cluster a suitable platform for more complex data gathering and analysis applications placed at the edge of a cloud. The findings show that while this is technically feasible, there are still some performance issues due to a relatively weak CPU, a limited network bandwidth, and problems with the file system.

Unlike the previous studies, this study aims to present the robustness and feasibility of Raspberry Pi as an edge-processing device for real-time data.

III. BACKGROUND

A. Proposed Architecture

In this paper, we are motivated to develop a real-time flash flood forecasting system utilising Raspberry Pi as an IoT edge processing device. The proposed system uses a new technique to forecast the flash flood-affected locations using the following parameters:

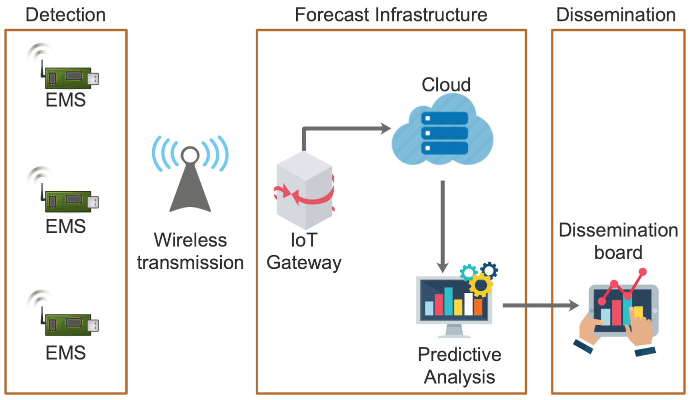


Fig. 1. The Proposed System Architecture.

1) Negative narrow bipolar event (–NBE) electric and magnetic fields signal.
2) Positive cloud-to-ground flash (+CG) electric and magnetic fields signal.
3) Cloud top height data [18]
4) Constant Altitude Plan Position Indicator (CAPPI) data [18]
5) Wind speed
6) Wind direction

Briefly, the –NBE and +CG lightning signals are collected and located by several lightning sensors in real-time [19]. Our research group at Universiti Teknikal Malaysia Melaka (UTeM) built these homemade sensors from scratch. These sensors have been working since 2015, tested, and calibrated [20][21]. The signal coverage for a single sensor is within a 300 km radius. Together with the rest of the parameters, real-time tracking can be done for any storms in Malaysia that can lead to flash floods. Specifically, cloud top height and CAPPI data are used to monitor rainfall and lightning flash rate intensity. The wind speed and direction are used to predict where a particular storm is heading and at what rate. By monitoring the occurrences of +CG and –NBEs produced by a storm in real-time, together with cloud top height and CAPPI rainfall rate, the proposed system will be able to forecast the rainfall intensity and rate for the next 1-3 hours period. Facilitated by wind speed and direction, the system can forecast the location where the downpour would impact most significant. Additional details for real-time forecasting of flash floods using the above technique are given in [19], [20], [21].

Fig. 1 illustrates the architecture of the proposed Flash Flood Early Warning System. Fig. 1 provides a general view of the proposed system that includes the three components. The three components are detection, forecast architecture and dissemination. The first component falls under the detection category, which contains the lightning sensors. The lightning sensor is called *Electric and Magnetic Field Sensor* (EMS). The main purpose of this sensor is to detect and locate lightning signals, particularly +CG and –NBE, within a 300 km radius. The signal is a combination waveform of electric and magnetic fields that are used to identify different types of lightning signals [19][21]. As shown in Fig. 2, there are five major components (or subsystems) in EMS system:

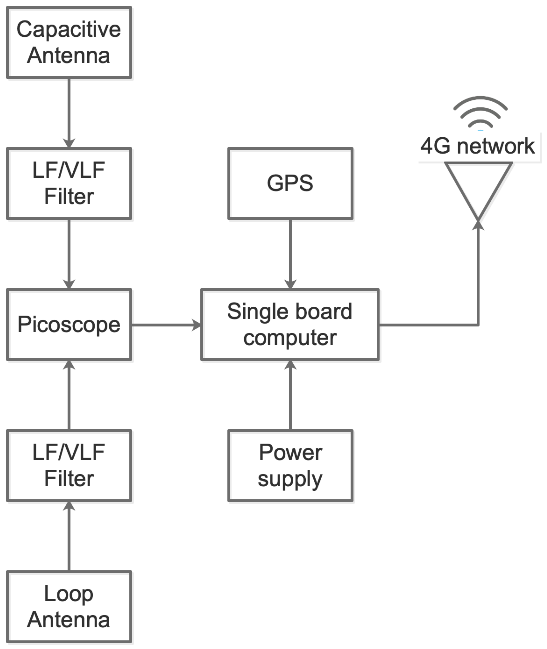


Fig. 2. Five Major Components or Subsystems of Electric and Magnetic Fields Sensor (EMS).

1) Antenna part with capacitive antenna to detect Electric field and two loop antennas to detect orthogonal components of magnetic field [20].
2) Filter circuits designed for detection of electric and magnetic fields at low frequency/very low frequency (LF/VLF) bands (or below than 1 MHz) [20].
3) Digitizer to digitize the captured analog lightning signals. Currently we are using the PicoScope and it is reliable.
4) A single board computer is used convert the digitize lightning signals to the coordinates of the lighting strikes and stream the EMS data in real time back to Forecasting Infrastructure via 4G network (or any available network infrastructure). Currently, the conversion and real time data streaming are implemented using Raspberry Pi platform.
5) Power supply from solar power, batteries, and power bank.

The focus of this paper is to evaluate the performance of the detection component, specifically the single board computer, in analysing and converting the digitised lighting signals to coordinates of the lightning strikes.

IV. Experiment Environment

As previously mentioned, this study aimed to evaluate the performance of the single board computer in the EMS system of the detection component (refer to Fig. 2). Fig. 3 presents the flowchart for the EMS system processes. As shown in Fig. 3, the antennas in the EMS will capture the lightning signals whenever the lightning event occurs [20][21]. The analogue signals are then digitised using the PicoScope. Next, the digitised lightning signals using a custom-written Python program that synchronises the signals with the Global Positioning System (GPS) clock and produces the coordinates

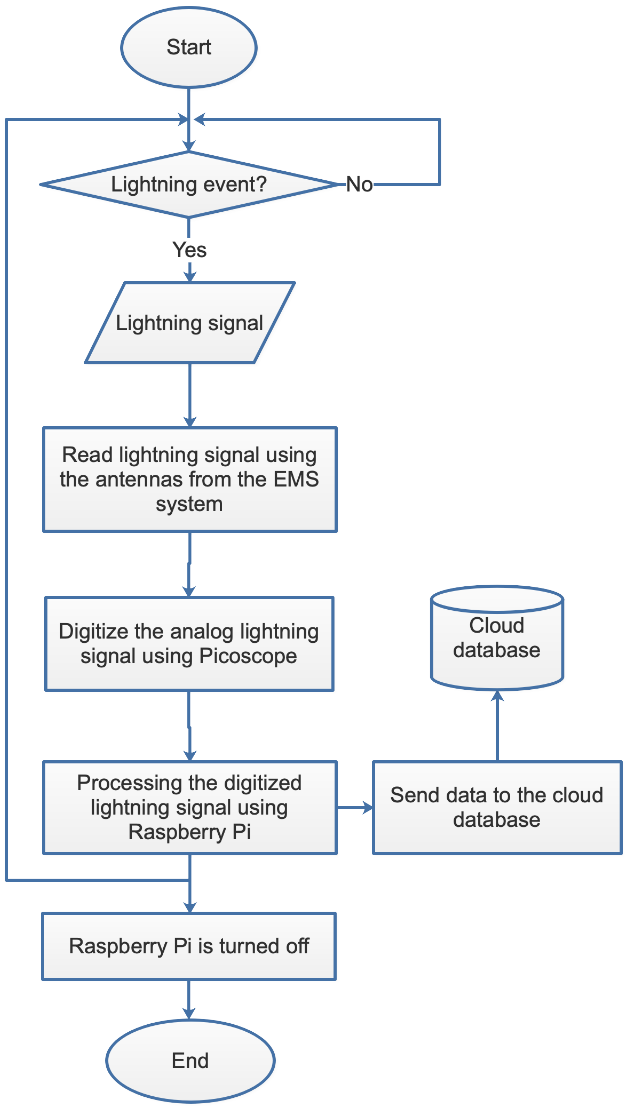

Start
Lightning event?
No
Yes
Lightning signal
Read lightning signal using the antennas from the EMS system
Digitize the analog lightning signal using Picoscope
Cloud database
Processing the digitized lightning signal using Raspberry Pi
Send data to the cloud database
Raspberry Pi is turned off
End

Fig. 3. The Flowchart for the EMS System.

of the lightning strikes [20][21]. Subsequently, the resulting coordinates are sent to the cloud database via IoT Gateway using any available wireless transmission (refer to Fig. 1).

A. Experiment Setup

In our experiment, as shown in Fig. 4, the components were integrated around the Raspberry Pi 3 as the main computer. The integration is done by connecting the EMS to the Raspberry Pi with the specification listed below in Table II. Fig. 3 shows that the digitised lightning signal is being fed into the Raspberry Pi to be processed by the custom-written Python program (refer to Algorithm 1) and produce the lightning strike locations. The resulting lightning strike locations are sent to the cloud database via an IoT gateway.

In the experiment, the process of converting the lightning signals into the lightning strike locations is repeated for 50 iterations in one cycle. The cycle is then repeated five times for statistical validity. However, in the actual scenario, the process in Fig. 3 only happens during a lightning storm. The main goal of this experiment is to observe the Raspberry Pi's performance if the lightning signal processing in Fig. 3 is executed repeatedly by measuring the Raspberry Pi's CPU

Algorithm 1 Signal Processing Program for the Digitised Lightning Signals

```
for each sequence until 100 do
    SET current time
    print sequence attempt
    INIT empty arrays: az, az2, d, ftype, pola, timeLDloc, ranges, typeflash, polar
    $result \leftarrow CPUTemperature$
    if ranges array is empty then
        print NO FLASHES WITHIN SELECTED RANGE
    else
        INIT empty arrays: flash, flashS
        for each element of typeflash array do
            if element of typeflash array is empty then
                if element of polar array is empty then
                    flash array ← 1
                    flashS array $\leftarrow' CG+'$
                else
                    flash array ← 2
                    flashS array $\leftarrow' CG-'$
                end if
            else
                if element of polar array is empty then
                    flash array ← 3
                    flashS array $\leftarrow' IC+'$
                else
                    flash array ← 4
                    flashS array $\leftarrow' IC-'$
                end if
            end if
        end for
        CALL math.radians(2.3139) RETURNING latitude radians INTO lat1
        CALL math.radians(102.3185) RETURNING longitude radians INTO lon1
        INIT empty arrays: lat2, lon2, rangedeg
        for each element of ranges array do
            CALL math.radians(¡element of rangedeg array¿/40075.01*360)
            return value radians INTO rangedeg array
        end for
        for each element of ranges array do
            $dlon2 \leftarrow math.asin(math.sin(lat1) \times math.cos(\frac{\text{element of rangedeg array}}{6378.1}) + math.cos(lat1) \times math.sin(\text{element of rangedeg array}) \times math.cos(\text{element of az2 array}))$
            $a \leftarrow math.sin(\text{element of az2 array}) \times math.sin(\frac{\text{element of rangedeg array}}{6378.1}) \times math.cos(lat1)$
            $b \leftarrow math.cos(\frac{\text{element of rangedeg array}}{6378.1}) - math.sin(lat1) \times math.sin(dlon2)$
            $dlat2 \leftarrow lon1 + math.atan2(a, b)$
            CALL math.degrees(dlat2) RETURNING value degrees INTO lat2 array
            CALL math.degrees(dlon2) RETURNING value degrees INTO lon2 array
        end for
    end if
end for
```

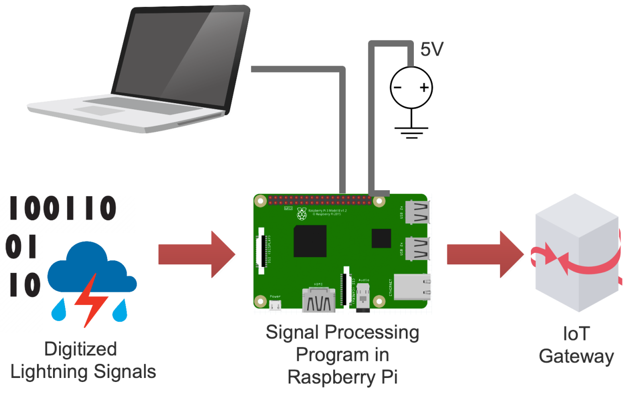

5V
100110
01
10
Digitized Lightning Signals
Signal Processing Program in Raspberry Pi
IoT Gateway

Fig. 4. The Experiment Setup.

TABLE II. HARDWARE SPECIFICATION FOR RASPBERRY PI

Hardware Specification	Raspberry Pi 3B+
CPU	ARMv8 Cortex-A5, 1.4GHz
CPU Cores	4
Memory	1GB LPDDR2 SDRAM
Integrated Wi-Fi	2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN

usage and temperature.

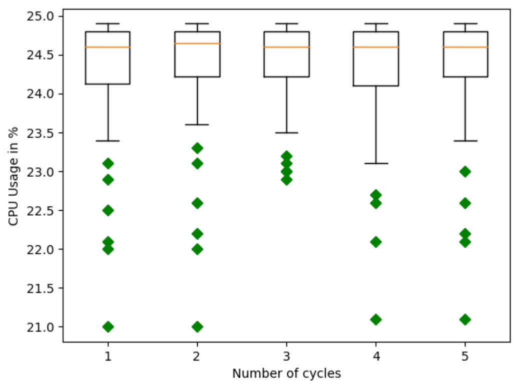
In each iteration, the CPU usage and the Raspberry Pi's temperature are measured using Python cross-platform library for retrieving information on running processes and system utilisation called psutil[22] library. In the experiment, the "cpu_percent()" function from the psutil library is used to measure the Raspberry Pi's system-wide CPU utilisation [22]. Another function called "CPUTemperature()" is used to monitor the Raspberry Pi's current temperature. This function is called from an application programming interface library for Raspberry Pi devices' general-purpose input/output (GPIO) devices [23]. In essence, the CPUTemperature function returns the Raspberry Pi's current temperature in degrees Celcius.

V. RESULTS AND DISCUSSION

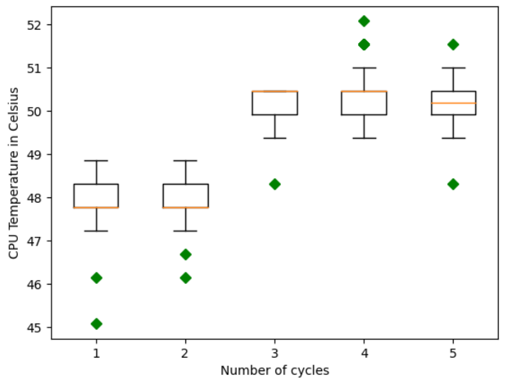
This section presents the performance evaluation results on the Raspberry Pi for digitised lightning signal processing. As mentioned in the previous section, the processing is repeated in 50 iterations in each cycle. Box plots in Fig. 5 show the statistic of the CPU usage and temperature measured in the experiment. For each data set, the first and third quartiles are represented by a box, with the median indicated in the box's centre. Whiskers, the lines extending from the box's edges, display the minimum and highest CPU utilisation and temperature.

Figs. 5a and 5b show the boxplots for the Raspberry Pi's system-wide CPU utilisation percentage and the Celcius's CPU temperature, respectively. In Fig. 5a, the interquartile range for Raspberry Pi's CPU usage is consistently between 23% and 25%, with a median of approximately 24.6% for each cycle. Each of the box plots for the CPU usage in Fig. 5a skews toward the value of 25%, which indicates that the processing of the lightning signals uses only 25% of the Raspberry Pi's CPU.

Fig. 5b shows the Raspberry Pi's CPU temperature distribution. The box plots in Fig. 5b demonstrate an increase in



(a) CPU Usage

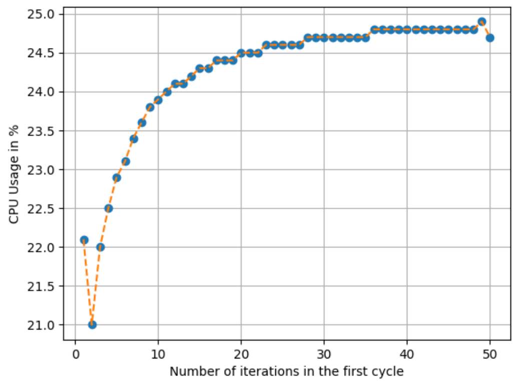


(b) Temperature

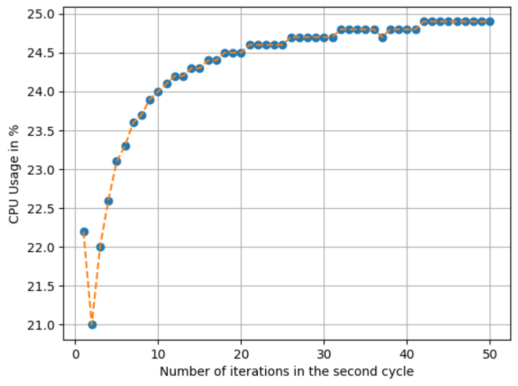
Fig. 5. The Raspberry Pi CPU Usage and Temperature Distributions.

the CPU temperature from 47 degrees Celcius to a maximum of 51 degrees Celcius. The median for cycles 1 and 2 is approximately 48 degrees Celcius with an increment to 50 degrees Celcius for cycles 3, 4 and 5. According to the Raspberry Pi official documentation [24], the Raspberry Pi is constructed with commercial-grade chips that are qualified for varied temperature ranges to keep prices down; the manufacturers indicate the USB and Ethernet controller of the Pi 3+ (Microchip LAN7515) as being qualified from 0°C to 70°C. The operating temperature range for the SoC (System on Chip - the integrated circuit that performs the Pi's processing, a Broadcom BCM2837B0) is -40 to 85 degrees Celsius [24], [25], [26]. Based on the official specification [24], the current workload in the experiment that is subjected to the Raspberry Pi is not enough to spike the temperature to a level that can damage the hardware.

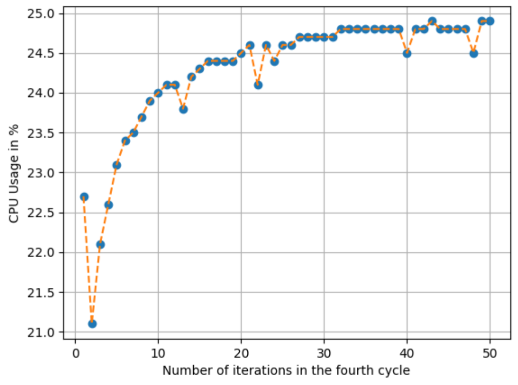
Figs. 6 and 7 shows the distribution for system-wide CPU utilisation and temperature for each cycle in the experiment. Figs. 6 show that the system-wide CPU utilisation is consistently at a maximum of 25% in each cycle, whereas Figs. 7 show an increment in the CPU temperature from Cycle 1 until Cycle 5. The distributions of CPU utilisation and temperature for 50 iterations in each cycle shown in Figs. 6 and 7 are consistent with the boxplot statistical distribution shown in



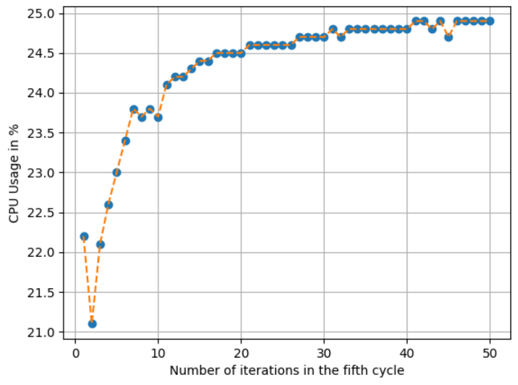
(a) First Cycle



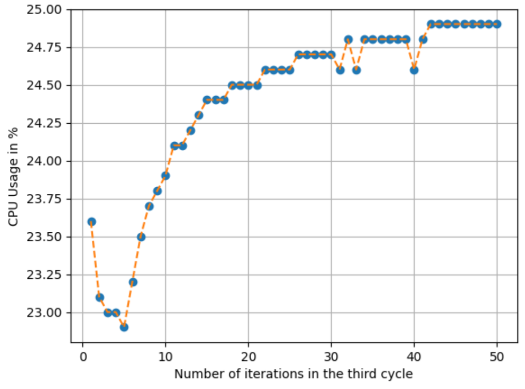
(b) Second Cycle



(c) Third Cycle



(d) Fourth Cycle

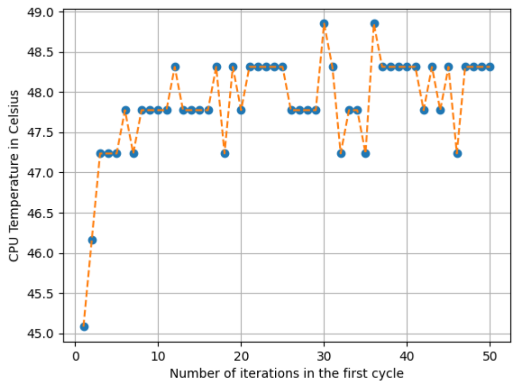


(e) Fifth Cycle

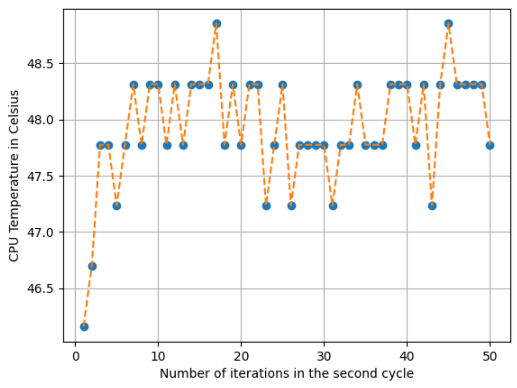
Fig. 6. The Raspberry Pi System-Wide CPU Utilisation for Five Cycles.

Figs. 5.

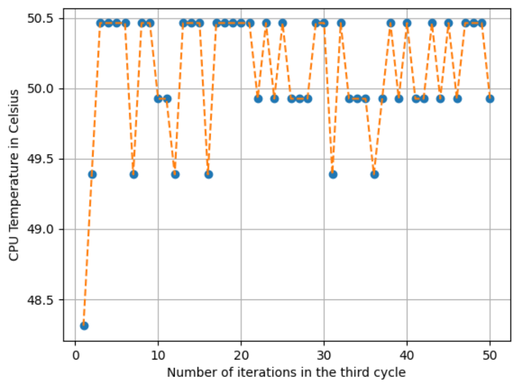
As mentioned in the previous section, the experiment conducted in this study involved the processing of converting lightning signals to lightning strike locations are run for fifty iterations throughout a single cycle where the cycle is repeated five times. Based on the workload introduced to Raspberry Pi in this experiment and the results obtained (Refer to Figs. 5 - 7), it can be ascertained that the Raspberry Pi can function as an IoT edge processing device. Figs. 5 - 6 demonstrate that Raspberry Pi maintains only 25% system-wide CPU utilisation with a maximum temperature of 51 degrees Celcius. As stated in the preceding section, the Raspberry Pi will not be subjected to a persistent heavy workload in a practical scenario since the collection and conversion processes are done during a



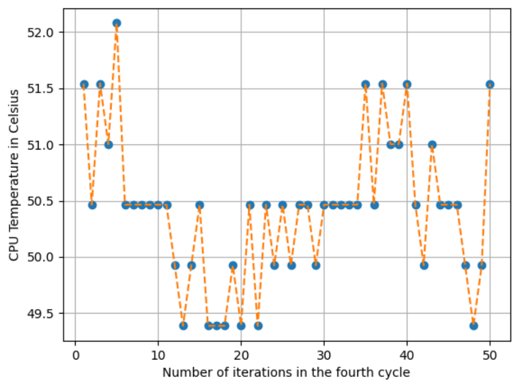
(a) First Cycle



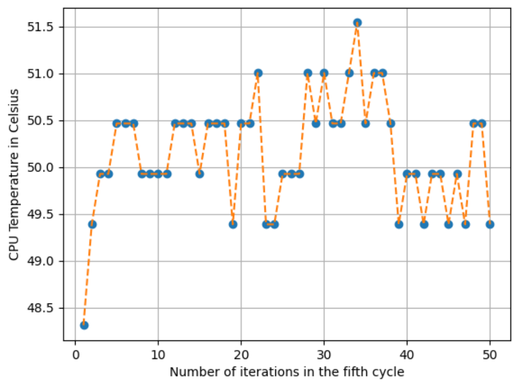
(b) Second Cycle



(c) Third Cycle



(d) Fourth Cycle



(e) Fifth Cycle

Fig. 7. The Raspberry Pi CPU Temperature for Five Cycles.

thunderstorm only.

VI. Conclusion

This research analysed the CPU utilisation and temperature of Raspberry Pi as an edge device that processes lightning strike signals to anticipate the location of flash floods. The experiment aimed to transmit continuous digital signals for processing to Raspberry Pi. The data indicate that the CPU is only utilised to a maximum of 25%, with a maximum temperature of 51 degrees Celsius. In a real-world scenario, data collection and processing are only performed during a thunderstorm. Therefore, these findings demonstrate Raspberry Pi's capability to handle a persistently substantial workload. As less processing is needed in a real scenario because the processing occurs only when a thunderstorm strikes, the results prove the Raspberry Pi's capability to process real-time signals.

Currently, this study only focuses on the performance evaluation of Raspberry Pi as an edge processing device in the proposed architecture. The proposed architecture will be implemented and evaluated in a real scenario for future work.

ACKNOWLEDGEMENTS

Authors are grateful to Universiti Teknikal Malaysia Melaka for the financial support through Collaborative Reserch Grant (CRG) S01707 and Short Term Grant (PJP/2020/FTMK/PP/S01764).

REFERENCES

[1] Priceonomics Data Studio, "The IoT Data Explosion: How Big Is the IoT Data Market?" 2019. [Online]. Available: https://priceonomics.com/the-iot-data-explosion-how-big-is-the-iot-data/

[2] Huawei.com, "Internet of Things Sensing Our Way into the Future," p. 1, 2017. [Online]. Available: https://www.huawei.com/en/industry-insights/technology/digital-transformation/iot http://www.huawei.com/en/industry-insights/digital-transformation/iot

[3] Cbinsights.com, "What Is Edge Computing? — CB Insights Research," 2020. [Online]. Available: https://www.cbinsights.com/research/what-is-edge-computing/

[4] K. Gyarmathy, "The Benefits and Potential of Edge Computing," 2019. [Online]. Available: https://www.vxchnge.com/blog/the-5-best-benefits-of-edge-computing

[5] Gartner, "Information Technology (IT) Glossary - Essential Information Technology (IT) Terms & Def initions — Gartner," n.d. [Online]. Available: https://www.gartner.com/en/information-technology/glossary/edge-computing

[6] K. Shaw, "What is edge computing and why it matters — Network World," 2019. [Online]. Available: https://www.networkworld.com/article/3224893/what-is-edge-computing-and-how-it-s-changing-the-network.html

[7] "Buy a Raspberry Pi 3 Model B+ – Raspberry Pi," n.d. [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/

[8] J. Yang, T. Qian, F. Zhang, and S. U. Khan, "Real-Time Facial Expression Recognition Based on Edge Computing," *IEEE Access*, vol. 9, pp. 76 178–76 190, 2021.

[9] J. Islam, T. Kumar, I. Kovacevic, and E. Harjula, "Resource-aware Dynamic Service Deployment for Local IoT Edge Computing: Healthcare Use Case," *IEEE Access*, 2021.

[10] T. Gizinski and X. Cao, "Design, Implementation and Performance of an Edge Computing Prototype Using Raspberry Pis," in *2022 IEEE 12th Annual Computing and Communication Workshop and Conference, CCWC 2022*. Institute of Electrical and Electronics Engineers Inc., 2022, pp. 592–601.

[11] C. Hegde, Z. Jiang, P. B. Suresha, J. Zelko, S. Seyedi, M. A. Smith, D. W. Wright, R. Kamaleswaran, M. A. Reyna, and G. D. Clifford, "AutoTriage - An open source edge computing Raspberry Pi-based clinical screening system," *medRxiv*, pp. 1–13, April 2020.

[12] F. Xhafa, B. Kilic, and P. Krause, "Evaluation of IoT stream processing at edge computing layer for semantic data enrichment," *Future Generation Computer Systems*, vol. 105, pp. 730–736, April 2020.

[13] S. Gondi and V. Pratap, "Performance evaluation of offline speech recognition on edge devices," *Electronics (Switzerland)*, vol. 10, no. 21, p. 2697, Nov 2021.

[14] E. Gamess and S. Hernandez, "Performance Evaluation of Different Raspberry Pi Models for a Broad Spectrum of Interests," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 2, pp. 819–829, Aug 2022.

[15] T. J. Freeborn, "Performance evaluation of raspberry Pi platform for bioimpedance analysis using least squares optimization," *Personal and Ubiquitous Computing*, vol. 23, no. 2, pp. 279–285, Feb 2019.

[16] A. Komninos, I. Simou, N. Gkorgkolis, and J. Garofalakis, "Performance of raspberry pi microclusters for edge machine learning in tourism," *Network (Mbps)*, vol. 100, no. 100, p. 100, 2019.

[17] L. Miori, J. Sanin, and S. Helmer, "A platform for edge computing based on raspberry pi clusters," in *British International Conference on Databases*. Springer, 2017, pp. 153–159.

[18] "MetMalaysia: Radar Semenanjung," n.d. [Online]. Available: http://www.met.gov.my/pencerapan/radar/radarsemenanjung

[19] M. R. Ahmad, M. R. Esa, V. Cooray, Z. A. Baharudin, and P. Hettiarachchi, "Latitude dependence of narrow bipolar pulse emissions," *Journal of Atmospheric and Solar-Terrestrial Physics*, 2015.

[20] D. Periannan, M. R. Ahmad, M. H. M. Sabri, M. R. M. Esa, S. A. Mohammad, G. Lu, and S. B. York, "Environmental study of tropical hailstorm and its relationship with negative narrow bipolar event and positive ground flashes," *Ekoloji*, 2019.

[21] M. H. Sabri, M. R. Ahmad, M. R. Esa, D. Periannan, G. Lu, H. Zhang, V. Cooray, E. Williams, M. Z. Aziz, Z. Abdul-Malek, A. A. Alkahtani, and M. Z. Kadir, "Initial electric field changes of lightning flashes in tropical thunderstorms and their relationship to the lightning initiation mechanism," *Atmospheric Research*, 2019.

[22] PSUtil, "psutil documentation — psutil 5.8.1 documentation," 2021. [Online]. Available: https://psutil.readthedocs.io/en/latest/

[23] B. Nuttall and D. Jones, "gpiozero — GPIO Zero 1.6.2 Documentation," 2021. [Online]. Available: https://gpiozero.readthedocs.io/en/stable/

[24] R. pi Foundation, "Raspberry Pi Documentation," p. 1, 2014. [Online]. Available: https://www.raspberrypi.com/documentation/

[25] H. Leadbeater and L. Walsh, "Prototyping on a Pi," Brainboxes, Tech. Rep., 2020. [Online]. Available: https://offer.brainboxes.com/prototyping-on-a-pi

[26] Brainboxes Buff, "BB-440 Industrial Edge Controller powered by Raspberry Pi," 2020. [Online]. Available: https://www.rs-online.com/designspark/how-does-raspberry-pi-deal-with-overheating