Journal of Advanced Research in Embedded System
Volume 11, Issue 1 - 2024, Pg. No. 11-17
Peer Reviewed Journal

Review Article

Developments in the Design of Microcontroller-Based Embedded Systems

Tanu Sinha

Student, JSS Science and Technology University, Mysore, Karnataka, India.

INFO

E-mail Id:
sinhatanu66@gmail.com
Orcid Id:
http://orcid.org/0001-0008-6302-4563

How to cite this article:
Sinha T. Developments in the Design of Microcontroller-Based Embedded Systems. *J Adv Res Embed Sys* 2024; 11(1): 11-17.

Date of Submission: 2024-05-01
Date of Acceptance: 2024-06-03

ABSTRACT

Microcontroller-based embedded systems are ubiquitous, underpinning a wide range of applications from consumer electronics to industrial automation. This review article explores the current trends, technological advancements, and future directions in microcontroller-based embedded system design. It covers various aspects such as hardware advancements, software development techniques, power management, connectivity, and security. Additionally, the article discusses the challenges and opportunities that lie ahead in the field. With the rapid evolution of technologies such as Internet of Things (IoT), artificial intelligence (AI), and edge computing, the design and implementation of embedded systems have become more complex yet more powerful. The article also highlights the growing importance of sustainability and the need for energy-efficient designs. By synthesizing recent research and industry developments, this review provides a comprehensive overview of the current state and future prospects of microcontroller-based embedded systems.

Keywords: Microcontroller-Based Embedded Systems, Secure Boot, IoT Integration, Firmware Over-the-Air (FOTA) Updates, Real-Time Operating Systems (RTOS)

Introduction

Embedded systems have become an integral part of modern technology, finding applications in diverse domains including automotive, healthcare, telecommunications, and home automation. At the heart of these systems are microcontrollers (MCUs), which serve as compact, low-power computing engines designed for specific tasks.

The importance of embedded systems has grown with the advent of the Internet of Things (IoT), where billions of interconnected devices communicate and cooperate to provide smarter solutions and enhance the quality of life. From smart thermostats and wearable health monitors to autonomous vehicles and industrial robots, embedded systems are the backbone of these innovations.[1,2]

Furthermore, advancements in artificial intelligence (AI) and machine learning (ML) have introduced new capabilities to embedded systems, enabling them to perform complex data analysis, make intelligent decisions, and adapt to changing environments in real-time. These capabilities are crucial in applications such as predictive maintenance, personalized healthcare, and smart cities.

In addition to AI and IoT, the push towards edge computing has also reshaped the landscape of embedded systems. Edge computing brings computation and data storage closer to the location where it is needed, reducing latency and bandwidth usage, and improving response times. This is particularly beneficial in scenarios requiring real-time processing and decision-making, such as autonomous driving and industrial automation.[3,4]

Journal of Advanced Research in Embedded System (ISSN: 2395-3802)

Hardware Advancements

The landscape of microcontroller-based embedded systems has witnessed significant hardware advancements, driving enhancements in performance, integration, power efficiency, and scalability. These advancements are pivotal in meeting the growing demands of modern applications, which require more powerful, versatile, and energy-efficient solutions. This section delves into the key hardware advancements in microcontroller technology.

Enhanced Processing Power

Recent advancements in semiconductor technology have led to the development of microcontrollers with significantly enhanced processing power. Modern MCUs, such as those based on ARM Cortex-M and RISC-V architectures, offer high clock speeds, multiple cores, and advanced instruction sets, enabling more complex and compute-intensive applications. These improvements have allowed embedded systems to handle more sophisticated tasks, such as real-time data processing, machine learning inference, and high-resolution signal processing.

The introduction of multi-core MCUs has been particularly transformative. Multi-core architectures allow parallel processing, where different cores can execute separate tasks simultaneously, improving overall system performance and responsiveness. This is essential for applications that require real-time performance, such as industrial automation, automotive systems, and robotics.[5, 6]

Integration of Peripherals

The integration of peripherals directly into the microcontroller has become more sophisticated. Contemporary MCUs come equipped with a wide array of integrated peripherals, such as analog-to-digital converters (ADCs), digital-to-analog converters (DACs), communication interfaces (SPI, I2C, UART), and timers. This integration reduces the need for external components, simplifying the design and reducing overall system cost and size.

Additionally, modern MCUs often include specialized peripherals tailored for specific applications. For instance, some MCUs feature integrated motor control peripherals for automotive and industrial motor applications, while others include digital signal processors (DSPs) for audio and signal processing tasks. This level of integration enables designers to develop more compact and efficient systems, reducing the complexity of the overall hardware design.[7,8]

Power Efficiency

Power efficiency remains a critical focus in MCU design. Techniques such as dynamic voltage and frequency scaling (DVFS), low-power sleep modes, and energy-efficient peripherals contribute to the development of MCUs that consume minimal power. These advancements are crucial for battery-powered and energy-harvesting applications, where energy efficiency directly impacts device longevity and performance.

Low-power sleep modes allow MCUs to enter a state of minimal power consumption when idle, waking up only to perform necessary tasks. This is particularly important for applications such as remote sensors and wearable devices, where conserving battery life is essential. Additionally, advancements in semiconductor materials and fabrication techniques have resulted in lower power consumption for active operations, further enhancing overall power efficiency.

Miniaturization and Packaging

The trend towards miniaturization continues to drive innovations in MCU packaging and form factors. Advanced packaging technologies, such as System-in-Package (SiP) and Multi-Chip Module (MCM), allow multiple components to be integrated into a single package, reducing the overall footprint of the device. This is particularly beneficial for applications where space is at a premium, such as wearable devices, medical implants, and IoT sensors.

Moreover, advancements in three-dimensional (3D) packaging techniques have enabled the stacking of multiple layers of components, further reducing the size and enhancing the performance of embedded systems. These packaging innovations also improve thermal management and signal integrity, ensuring reliable operation in compact and densely populated designs.[9,10]

Scalability and Flexibility

Scalability and flexibility are becoming increasingly important in MCU design, as applications demand customizable and adaptable solutions. Modern MCUs offer a range of scalability options, including configurable peripherals, programmable logic, and software-defined functionality. This allows designers to tailor the MCU to specific application requirements, providing a high degree of flexibility and enabling rapid prototyping and development.

Reconfigurable hardware, such as field-programmable gate arrays (FPGAs) integrated with MCUs, offers another layer of flexibility. These hybrid solutions combine the programmability of FPGAs with the processing power of MCUs, enabling the development of highly customized and adaptable embedded systems. This is particularly valuable in applications that require frequent updates or customization, such as industrial automation and IoT devices.

Software Development Techniques

Advancements in microcontroller-based embedded systems are not solely driven by hardware improvements; software development techniques have also evolved significantly. These advancements have enabled developers to create more robust, efficient, and scalable embedded applications. This section explores the key software development techniques that are shaping the landscape of microcontroller-based embedded systems.

Real-Time Operating Systems (RTOS)

The use of Real-Time Operating Systems (RTOS) has become increasingly prevalent in embedded systems. RTOS such as FreeRTOS, Zephyr, and ARM Mbed OS provide deterministic task scheduling, inter-task communication, and resource management, enabling developers to create responsive and reliable applications.[11, 12]

Key Features of RTOS

- **Deterministic Scheduling:** Ensures that high-priority tasks are executed within a predictable timeframe, crucial for real-time applications.
- **Inter-task Communication:** Mechanisms such as message queues, semaphores, and mutexes facilitate communication and synchronization between tasks.
- **Resource Management:** Efficient management of CPU, memory, and other resources ensures optimal system performance.

RTOS are essential in applications where timing and reliability are critical, such as industrial automation, automotive systems, and medical devices.

Model-Based Design

Model-based design (MBD) has revolutionized the development of embedded systems by providing a higher level of abstraction. Tools like MATLAB/Simulink and LabVIEW allow designers to create and simulate models of their systems, automatically generating code that can be deployed to MCUs.

Benefits of Model-Based Design

- **Rapid Prototyping:** Allows for quick iteration and testing of designs, reducing development time.
- **Improved Reliability:** Simulation and verification of models before deployment ensure that the system behaves as expected.
- **Automatic Code Generation:** Reduces the risk of human error and ensures consistency between the model and the deployed system.

MBD is particularly useful in complex systems requiring rigorous testing and validation, such as aerospace, automotive, and industrial control systems.

Advanced Debugging and Profiling Tools

Modern integrated development environments (IDEs) and debugging tools offer advanced features that significantly enhance the software development process. Tools like SEGGER J-Link, ARM Keil, and IAR Embedded Workbench provide real-time tracing, performance profiling, and power analysis capabilities.[13,14]

Key Debugging and Profiling Features

- **Real-Time Tracing:** Allows developers to trace program execution in real-time, identifying issues such as race conditions and deadlocks.
- **Performance Profiling:** Analyzes the performance of the code, identifying bottlenecks and optimizing execution.
- **Power Analysis:** Measures the power consumption of the system, helping developers create energy-efficient applications.

These tools are indispensable for developing high-performance and low-power embedded systems, ensuring that the software meets the stringent requirements of modern applications.

Development Frameworks and Middleware

Development frameworks and middleware play a crucial role in simplifying the development process and enhancing the capabilities of embedded systems. Frameworks like Arduino, PlatformIO, and MicroPython provide libraries and tools that streamline the development of embedded applications.

Benefits of Development Frameworks

- **Ease of Use:** Abstracts the complexity of hardware interfacing and low-level programming, allowing developers to focus on application logic.
- **Code Reusability:** Provides a wide range of pre-built libraries and components that can be reused across different projects.
- **Community Support:** Extensive communities and resources help developers troubleshoot issues and share knowledge.

Middleware solutions such as communication stacks (e.g., TCP/IP, Bluetooth, Zigbee) and real-time data processing libraries further enhance the functionality of embedded systems, enabling seamless integration and interoperability.

Version Control and Continuous Integration

Adopting version control systems (VCS) and continuous integration (CI) practices has become essential in embedded software development. Tools like Git, Jenkins, and GitLab CI enable teams to manage code changes efficiently and automate the build and testing processes.

Advantages of VCS and CI:

- **Version Control:** Tracks changes to the codebase, facilitating collaboration and reducing the risk of conflicts and errors.
- **Automated Builds and Tests:** Ensures that code

changes are automatically built and tested, identifying issues early in the development cycle.

- **Continuous Deployment:** Automates the deployment process, reducing the time and effort required to release updates.

These practices improve the overall quality and reliability of embedded software, making the development process more agile and responsive to changes.

Connectivity and IoT Integration

The proliferation of the Internet of Things (IoT) has significantly influenced the development of microcontroller-based embedded systems, driving the need for robust connectivity options. Connectivity enables embedded systems to interact with other devices, exchange data, and integrate with cloud services, facilitating remote monitoring, control, and data analytics. This section explores the key aspects of connectivity and IoT integration in modern microcontroller-based embedded systems.[15, 16]

Integrated Wireless Communication Modules

Modern microcontrollers often come with integrated wireless communication modules, providing seamless connectivity options for IoT applications. These integrated modules support various communication protocols, including Wi-Fi, Bluetooth, Zigbee, LoRa, and cellular networks. The inclusion of these modules simplifies the design process, reduces the need for external components, and ensures reliable communication.

Key Wireless Communication Protocols

- **Wi-Fi:** Suitable for high-speed data transfer and connectivity to local networks and the internet.
- **Bluetooth and Bluetooth Low Energy (BLE):** Ideal for short-range communication with low power consumption, commonly used in wearable devices and personal area networks.
- **Zigbee:** Designed for low-power, low-data-rate applications, often used in home automation and industrial control.
- **LoRa and LoRaWAN:** Long-range communication with low power consumption, suitable for remote sensing and monitoring applications.
- **Cellular (LTE, NB-IoT, and 5G):** Provides wide-area coverage and high reliability, essential for mobile and remote IoT devices.

IoT Protocols and Standards

To facilitate interoperability and communication between diverse devices, several IoT protocols and standards have been developed. These protocols ensure efficient and secure data exchange across different platforms and networks.

Prominent IoT Protocols

- **MQTT (Message Queuing Telemetry Transport):** A lightweight, publish-subscribe messaging protocol designed for constrained devices and low-bandwidth, high-latency networks. MQTT is widely used in IoT applications for real-time communication.
- **CoAP (Constrained Application Protocol):** A protocol specifically designed for resource-constrained devices, enabling efficient communication over the internet. CoAP uses a client-server model and is suitable for applications such as smart lighting and environmental monitoring.
- **HTTP/HTTPS:** Although more resource-intensive, HTTP/HTTPS is used for web-based IoT applications where security and compatibility with web technologies are paramount.
- **OPC UA (Open Platform Communications Unified Architecture):** A machine-to-machine communication protocol for industrial automation, providing reliable, secure data transfer and integration across different systems [17, 18].

Cloud Integration and Edge Computing

The integration of cloud services with embedded systems has transformed the way data is collected, processed, and analyzed. Cloud platforms such as Amazon Web Services (AWS) IoT, Microsoft Azure IoT, and Google Cloud IoT provide extensive services for data storage, analytics, machine learning, and device management.

Benefits of Cloud Integration

- **Scalability:** Cloud platforms offer virtually unlimited resources, enabling the scaling of IoT applications to accommodate growing numbers of devices and data volumes.
- **Data Analytics and Machine Learning:** Cloud services provide powerful tools for analyzing data and applying machine learning algorithms, deriving insights and enabling predictive maintenance and optimization.
- **Remote Management:** Cloud integration allows for remote monitoring, updates, and control of IoT devices, enhancing maintenance and reducing downtime.

In contrast, edge computing brings computation and data storage closer to the source of data, reducing latency and bandwidth usage. Edge devices process data locally, making real-time decisions and only sending relevant data to the cloud. This approach is particularly beneficial for applications requiring immediate response times, such as autonomous vehicles, industrial automation, and smart healthcare.

Security in IoT Integration

Security is a paramount concern in IoT integration, as connected devices are often targets for cyber-attacks. Ensuring the security of embedded systems involves implementing robust hardware and software measures.

Security Measures

- **Secure Boot and Firmware Updates:** Ensures that devices boot using only trusted firmware and receive authenticated updates, preventing unauthorized access and tampering.
- **Encryption:** Data encryption during transmission and storage protects against eavesdropping and data breaches. Protocols such as TLS/SSL are commonly used to secure data communication.
- **Authentication and Authorization:** Ensures that only authorized devices and users can access the system. Techniques include multi-factor authentication (MFA) and Public Key Infrastructure (PKI).
- **Hardware Security Modules (HSM):** Dedicated hardware components that provide secure key storage and cryptographic operations, enhancing overall system security.

Interoperability and Standards Compliance

Ensuring interoperability between different devices and systems is crucial for the widespread adoption of IoT solutions. Adhering to industry standards and protocols facilitates seamless communication and integration, reducing development complexity and enhancing compatibility.

Key Standards

- **IEEE 802.11 (Wi-Fi):** Ensures compatibility across different Wi-Fi devices and networks.
- **Bluetooth SIG (Special Interest Group):** Defines standards for Bluetooth communication, ensuring interoperability between devices.
- **Zigbee Alliance:** Develops and maintains standards for Zigbee communication, promoting interoperability in home automation and industrial applications.
- **3GPP (3rd Generation Partnership Project):** Defines standards for cellular communication, including LTE and 5G, ensuring compatibility and reliability in wide-area networks.

Security Considerations

As microcontroller-based embedded systems become increasingly integrated into critical infrastructure and consumer applications, security has emerged as a paramount concern. The interconnected nature of these systems makes them vulnerable to a wide range of cyber threats. Ensuring robust security in embedded systems involves addressing challenges at both the hardware and software levels. This section explores key security considerations and techniques used to safeguard embedded systems.

Secure Boot

Secure boot is a fundamental security measure that ensures a device boots using only trusted and authenticated firmware. This process involves verifying the digital signature of the firmware before execution, preventing the system from running unauthorized or malicious code.

Key Aspects of Secure Boot

- **Cryptographic Signatures:** Firmware is signed using a cryptographic algorithm, and the signature is verified during the boot process.
- **Bootloaders:** Implement secure boot mechanisms to validate the integrity and authenticity of the firmware.
- **Root of Trust:** A secure and immutable part of the hardware that stores cryptographic keys and performs the initial verification steps.

Firmware Over-the-Air (FOTA) Updates

Firmware updates are critical for maintaining security by patching vulnerabilities and adding new features. FOTA updates allow devices to receive firmware updates remotely, ensuring that devices remain secure and up-to-date without physical intervention.

Security Considerations for FOTA

- **Authentication:** Ensures that firmware updates come from a trusted source, preventing unauthorized updates.
- **Integrity Checks:** Verifies the integrity of the firmware package to ensure it has not been tampered with during transmission.
- **Rollback Protection:** Prevents the installation of older, potentially vulnerable firmware versions.

Data Encryption

Encryption is essential for protecting sensitive data both at rest and in transit. Embedded systems often handle critical data, such as personal information, financial transactions, and control signals, which must be secured against unauthorized access and eavesdropping.

Types of Encryption

- **Symmetric Encryption:** Uses the same key for encryption and decryption. Suitable for fast, real-time data protection.
- **Asymmetric Encryption:** Uses a pair of keys (public and private) for encryption and decryption. Commonly used for secure communications and key exchange.
- **TLS/SSL:** Protocols that provide end-to-end encryption for data transmitted over networks, commonly used in IoT communications.

Authentication and Authorization

Authentication and authorization mechanisms ensure that only authorized users and devices can access the embedded system and perform specific actions. Robust authentication processes are crucial for preventing unauthorized access and ensuring that entities interacting with the system are legitimate.

Authentication Methods

- **Password-Based Authentication:** Simple but less secure method relying on passwords.
- **Token-Based Authentication:** Uses tokens, such as JWT (JSON Web Tokens), for secure access.
- **Multi-Factor Authentication (MFA):** Combines multiple authentication factors (e.g., something you know, something you have, something you are) to enhance security.

Authorization

- **Role-Based Access Control (RBAC):** Assigns permissions based on the user's role within the system, limiting access to sensitive functions and data.
- **Access Control Lists (ACLs):** Define which users or devices have access to specific resources and operations.

Hardware Security Modules (HSM)

Hardware Security Modules (HSM) are dedicated hardware components that provide secure key storage and cryptographic operations. HSMs enhance the security of embedded systems by offloading cryptographic functions to tamper-resistant hardware, ensuring that sensitive keys are protected from unauthorized access.

Functions of HSM

- **Key Generation and Management:** Securely generate, store, and manage cryptographic keys.
- **Cryptographic Operations:** Perform encryption, decryption, digital signatures, and other cryptographic functions.
- **Tamper Resistance:** Designed to resist physical tampering and unauthorized extraction of keys.

Intrusion Detection and Prevention Systems (IDPS)

Intrusion Detection and Prevention Systems (IDPS) monitor embedded systems for signs of malicious activity and respond to detected threats. IDPS can be implemented in both software and hardware to provide real-time protection.

IDPS Features

- Anomaly Detection: Identifies deviations from normal behavior, indicating potential security threats.
- Signature-Based Detection: Uses known signatures of malware and attack patterns to identify threats.
- Response Mechanisms: Includes automated actions such as blocking suspicious activities, alerting administrators, and logging security events.

Secure Communication Protocols

Using secure communication protocols ensures the confidentiality, integrity, and authenticity of data exchanged between devices. Protocols such as HTTPS, MQTT with TLS, and CoAP with DTLS provide encrypted communication channels, protecting data from interception and tampering.

Key Features of Secure Communication Protocols

- **End-to-End Encryption:** Ensures that data is encrypted from the sender to the receiver.
- **Mutual Authentication:** Both parties authenticate each other to prevent man-in-the-middle attacks.
- **Data Integrity Checks:** Verifies that data has not been altered during transmission.

Regular Security Audits and Penetration Testing

Regular security audits and penetration testing are essential for identifying and mitigating vulnerabilities in embedded systems. These practices involve systematically examining the system's security posture and testing its defenses against simulated attacks.

Security Audit Activities

- **Code Review:** Analyzing source code for security flaws and vulnerabilities.
- **Configuration Review:** Ensuring that security settings and configurations follow best practices.
- **Compliance Checks:** Verifying adherence to security standards and regulations.

Penetration Testing

Vulnerability Scanning: Automated tools scan the system for known vulnerabilities.
Exploitation: Ethical hackers attempt to exploit identified vulnerabilities to assess their impact.
Reporting and Mitigation: Detailed reports on findings, with recommendations for remediation.[19, 20]

Challenges and Future Directions

Scalability and Flexibility

Designing scalable and flexible systems that can adapt to changing requirements remains a challenge. Future developments in reconfigurable hardware and software-defined peripherals are expected to address these issues.

Artificial Intelligence and Machine Learning

The integration of AI and machine learning capabilities into embedded systems is an emerging trend. MCUs with specialized hardware accelerators for neural networks are being developed to enable on-device AI processing, reducing the need for constant cloud connectivity.

Sustainability

As the demand for embedded systems grows, so does the need for sustainable design practices. Future developments will likely focus on creating more eco-friendly MCUs with longer lifespans, reduced environmental impact, and better recyclability.

Conclusion

The field of microcontroller-based embedded systems design is rapidly evolving, driven by advances in hardware, software, connectivity, and security. These innovations are enabling the development of more powerful, efficient, and secure embedded systems that are integral to modern technology. As the field progresses, it will continue to address the challenges of scalability, AI integration, and sustainability, paving the way for the next generation of embedded applications.

References

1. Yiu J. The Definitive Guide to ARM® Cortex®-M3 and Cortex®-M4 Processors. Newnes; 2013 Oct 6.
2. Khan WZ, Rehman MH, Zangoti HM, Afzal MK, Armi N, Salah K. Industrial internet of things: Recent advances, enabling technologies and open challenges. Computers & electrical engineering. 2020 Jan 1;81:106522.
3. Hayyolalam V, Aloqaily M, Özkasap Ö, Guizani M. Edge-assisted solutions for IoT-based connected healthcare systems: A literature review. IEEE Internet of Things Journal. 2021 Dec 14;9(12):9419-43.
4. Yun J, Rustamov F, Kim J, Shin Y. Fuzzing of embedded systems: A survey. ACM Computing Surveys. 2022 Dec 15;55(7):1-33.
5. Gilmore PG, Rivest RL, Schiller JI, Schneier B. pp 6–15; also as DEC SRC Research Report no 125 (June 1 1994) [3] A Abbasi, HC Chen,“Visualizing Authorship for Identification”, in ISI 2006, LNCS 3975 pp 60–71 [4] H Abelson, RJ Anderson, SM Bellovin, J Benaloh, M Blaze, W Diffie, J. IBM Journal of Research & Development. 1984;28(1):2-14.
6. Cooper J, De la Vega A, Paige R, Kolovos D, Bennett M, Brown C, Pina BS, Rodriguez HH. Model-based development of engine control systems: Experiences and lessons learnt. In2021 ACM/IEEE 24th International Conference on Model Driven Engineering Languages and Systems (MODELS) 2021 Oct 10 (pp. 308-319). IEEE.
7. Cooper J, De la Vega A, Paige R, Kolovos D, Bennett M, Brown C, Pina BS, Rodriguez HH. Model-based development of engine control systems: Experiences and lessons learnt. In2021 ACM/IEEE 24th International Conference on Model Driven Engineering Languages and Systems (MODELS) 2021 Oct 10 (pp. 308-319). IEEE.
8. Eibeck A, Shaocong Z, Mei Qi L, Kraft M. Research data supporting” A Simple and Efficient Approach to Unsupervised Instance Matching and its Application to Linked Data of Power Plants”.
9. Zhang J, Tao D. Empowering things with intelligence: a survey of the progress, challenges, and opportunities in artificial intelligence of things. IEEE Internet of Things Journal. 2020 Nov 19;8(10):7789-817.
10. Anand P, Singh Y, Selwal A, Alazab M, Tanwar S, Kumar N. IoT vulnerability assessment for sustainable computing: threats, current solutions, and open challenges. IEEE Access. 2020 Sep 9;8:168825-53.
11. Chakraborty C, Rajendran SR, Rehman MH. SECURITY OF INTERNET OF THINGS NODES.
12. Cho J. Efficient Autonomous Defense System Using Machine Learning on Edge Device. Computers, Materials & Continua. 2022 Feb 1;70(2).
13. Wei L, Yang Y, Wu J, Long C, Li B. Trust management for internet of things: A comprehensive study. IEEE Internet of Things Journal. 2022 Jan 3;9(10):7664-79.
14. Mahapatra SN, Singh BK, Kumar V. A survey on secure transmission in internet of things: taxonomy, recent techniques, research requirements, and challenges. Arabian Journal for Science and Engineering. 2020 Aug;45(8):6211-40.
15. Attarian R, Mohammadi E, Wang T, Beni EH. Mixflow: Assessing mixnets anonymity with contrastive architectures and semantic network information. Cryptology ePrint Archive. 2023.
16. Chen Y, Zheng B, Zhang Z, Wang Q, Shen C, Zhang Q. Deep learning on mobile and embedded devices: State-of-the-art, challenges, and future directions. ACM Computing Surveys (CSUR). 2020 Aug 20;53(4):1-37.
17. Sarjan H, Ameli A, Ghafouri M. Cyber-security of industrial internet of things in electric power systems. IEEE Access. 2022 Aug 29;10:92390-409.
18. De Micco L, Vargas FL, Fierens PI. A literature review on embedded systems. IEEE Latin America Transactions. 2019 Feb;18(02):188-205.
19. Arfaoui G, Gharout S, Traoré J. Trusted execution environments: A look under the hood. In2014 2nd IEEE international conference on mobile cloud computing, services, and Engineering 2014 Apr 8 (pp. 259-266). IEEE.
20. Xenofontos C, Zografopoulos I, Konstantinou C, Jolfaei A, Khan MK, Choo KK. Consumer, commercial, and industrial iot (in) security: Attack taxonomy and case studies. IEEE Internet of Things Journal. 2021 May 13;9(1):199-221.