International Journal of Innovative Research in Engineering
Volume 5, Issue 2 (March-April 2024), PP: 99-102.
www.theijire.com **ISSN No: 2582-8746**

Design and Implementation of Clusters of Microcontroller for various sensors data on real time using Embedded System

Priyanka Gaherwar[1], Rahul Dhuture[2], Amol Dhenge[3]
[1]PG Scholar,Electronics and communication Engineering, TulsiramjiGaikwad-Patil College of Engineering & Technology, Nagpur, Maharashtra, India.
[2,3] Professor, Electronics and communication Engineering, TulsiramjiGaikwad-Patil College of Engineering & Technology, Nagpur, Maharashtra, India.

How to cite this paper:
Priyanka Gaherwar[1], Rahul Dhuture[2], Amol Dhenge[3], "Design and Implementation of Clusters of Microcontroller for various sensors data on real time using Embedded System", IJIRE-V5I02-99-102.

Abstract: *The need for an efficient and scalable system for real-time collection, processing, and management of data from various sensors using embedded systems. The proposed solution involves the design and implementation of clusters of microcontrollers, providing a distributed architecture capable of handling diverse sensor types in a seamless and energy-efficient manner. The Key parameters for development of embedded systems optimized for low power consumption, the establishment of real-time communication protocols for data exchange within clusters, and the integration of fault-tolerant mechanisms to ensure the robustness of the system. Clusters are dynamically formed, allowing for the addition of new sensors and microcontrollers to accommodate evolving requirements. The microcontrollers within the clusters are tasked with real-time processing of sensor data, including computations, filtering, and aggregation. Customizable sensor data fusion algorithms are implemented to enhance the accuracy and reliability of the collected data. Energy efficiency is a focal point, ensuring that the system is well-suited for deployment in resource-constrained environments, such as remote or inaccessible locations.*

Key Word: *Arduino, Clustering, Master Slave Real time.*

I.INTRODUCTION

Distributed systems are a suitable alternative to optimize the current embedded technologies, since this kind of systems increase processing power to execute tasks, which otherwise, would take plenty of time to finish with a single processing core, likewise, a larger amount of electronic sensing devices can be added to the system, improving its capacity to collect and analyze data what's over. In this work, we propose to use distributed systems theory,in8 bit CMOS (Complementary Metal-Oxide Semiconductor) microcontrollers, more in particular, devices from Microchip, which have are diced and limited architecture. By doing this, a low-power embedded system focused on performing multiple operations and tasks is proposed, in which, time is a vital element, such as real time operating systems, using hard- wired embedded communication protocols to accomplish the correct distribution of data. Using cluster system of the microcontrollers, instead of a microprocessor-based distributed system or a single core embedded system, implies important advantages, such as considerable cost reduction, size flexibility to install this kind of systems in several applications, and a great possibility to modify the initial design and functionality to couple to all the needs of the user. Also, one of the objectives of this development is to create an easily scalable distributed system, depending on the needs of the use given. The combination of the features mentioned, makes this system suitable for educational purposes, since its simplicity is ideal for students of topics regarding distributed systems.

Security measures, including encryption, authentication, and authorization, are incorporated to safeguard the integrity and confidentiality of sensor data. The system is designed to seamlessly integrate with cloud or edge computing platforms, facilitating centralized data storage, analytics, and visualization for further analysis and decision-making. A user interface is developed to provide real-time monitoring and visualization of the sensor data and cluster status, enhancing system management and troubleshooting capabilities. The scalability of the system allows for flexibility in adapting to changing sensor deployments and evolving application requirements. This project contributes to the advancement of embedded systems for sensor data management, offering a robust and efficient solution that can be applied to various domains, including environmental monitoring, industrial automation, and smart infrastructure. The successful implementation of clusters of microcontrollers demonstrates the feasibility and effectiveness of the proposed architecture in real-world applications.

II.LITERATURE REVIEW

n modern military operations, effective communication plays a pivotal role in achieving mission success and ensuring the safety of personnel. Long-Range (LoRa) wireless technology has emerged as a valuable asset in addressing the communication challenges faced in military environments. This paper introduces the concept of employing LoRa wireless communication using two Arduino boards for military purposes, highlighting its potential to revolutionize information exchange in the field[1].

Communication is the backbone of any military operation, enabling coordination, information dissemination, and real-time decision-making. In dynamic and complex scenarios, such as combat zones or remote deployments, traditional communication methods may encounter limitations due to terrain obstacles, signal interference, and power constraints. To overcome these challenges, innovative technologies like LoRa are being explored to establish robust and resilient communication networks [2]

LoRa stands out as a wireless modulation technique designed to provide long-range communication with minimal power consumption LoRa's ability to transmit data over extended distances, sometimes exceeding several kilometres, while consuming very little energy, makes it an appealing solution for military applications[3].

Arduino, an open-source hardware and software platform. Its potential within military communication lies in its capability to serve as a foundation for building customized communication systems. By integrating LoRa modules into Arduino boards, military personnel can swiftly create communication setups tailored to specific operational requirements [4]

III.SYSTEM ARCHITECTURE

The system architecture employs a hierarchical structure with clusters of Arduino microcontrollers responsible for collecting, processing, and aggregating sensor data. The architecture is designed to be scalable, energy-efficient, and capable of real-time communication.

2.1 Components:

Sensor Nodes: Arduino microcontrollers equipped with various sensors such as temperature, humidity, and motion sensors. Each sensor node is responsible for collecting data from its sensors and transmitting it to the cluster head.

Cluster Heads: Selected Arduino microcontrollers act as cluster heads. They are responsible for aggregating data from multiple sensor nodes within their cluster, performing initial processing, and transmitting the aggregated data to a central unit or gateway.

Central Unit/Gateway: At the highest level, a central unit or gateway is responsible for receiving data from multiple cluster heads, performing further processing if necessary, and transmitting the final data to external systems or storage platforms.

2.2 Communication:

Wireless Communication: Arduinos communicate wirelessly using protocols such as Zig bee or Bluetooth, allowing seamless communication between sensor nodes and cluster heads. This wireless communication ensures flexibility and ease of deployment in various environments. Serial Communication: Arduino microcontrollers within a cluster communicate with each other using serial communication. This enables real-time data exchange, allowing efficient aggregation and processing of sensor data at the cluster head.

3. Components of the Implementation:

3.1 Hardware Implementation:

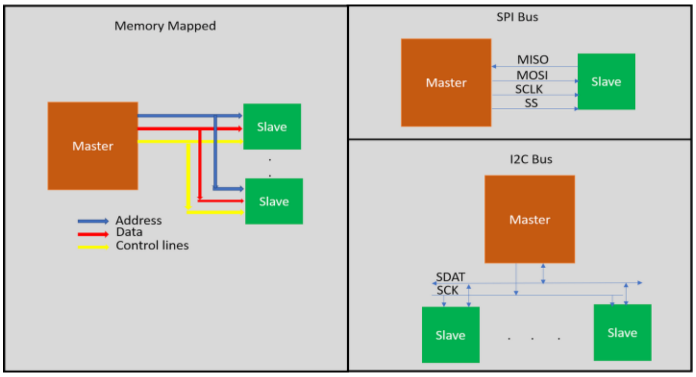
Arduino Microcontrollers: Deploy low-cost, energy-efficient Arduino boards (e.g., Arduino Uno, Arduino Nano) as the primary microcontrollers for both sensor nodes and cluster heads. Sensors: Integrate various sensors compatible with Arduino boards. For instance, use DHT sensors for temperature and humidity, PIR motion sensors, or other sensors based on the application requirements.

3.2 Software Implementation:

Arduino IDE: Develop the firmware using the Arduino IDE, incorporating Arduino-specific programming language (based on C/C++). Write code for sensor data collection, real-time communication, and aggregation algorithms.

Real-Time Processing Algorithms: Implement algorithms for real-time data processing directly on Arduino boards. For example, perform basic calculations, filtering, or aggregation of sensor data to reduce the amount of transmitted information.

IV.MASTER SLAVE CONCEPT



The master-slave concept is a communication paradigm commonly used in embedded systems and microcontroller-based applications. It involves one primary device (master) controlling and interacting with one or more secondary devices (slaves). The Arduino platform, with its various models and capabilities, is often employed in master-slave configurations for diverse applications. Let's explore the master-slave concept in the context of Arduino:

Components of a Master-Slave System using Arduino:

1. Arduino Boards:
Master Arduino: This board assumes the role of the master device, typically responsible for initiating communication and controlling one or more slave devices.

Slave Arduino(s): These boards act as subordinate devices, responding to commands from the master and possibly performing specific tasks or providing data.

2. Communication Protocols:
Serial Communication: Arduino boards often communicate using UART serial communication. The master and slave devices can exchange data through the serial ports of their respective Arduino boards.

I2C (Inter-Integrated Circuit): In scenarios where multiple slaves need to communicate with a master, I2C can be employed. The master initiates communication and addresses specific slaves on the bus.

SPI (Serial Peripheral Interface): Similar to I2C, SPI is another protocol suitable for communication between a master and multiple slaves. It involves a master device controlling the communication with one slave at a time.

3. Implementation Steps:
Master Initialization: The master Arduino initializes the communication interface and sets parameters such as baud rate (for serial communication) or I2C/SPI configurations.

Slave Configuration:
Each slave Arduino is configured to listen for commands or data from the master. The slave may be programmed to execute specific tasks upon receiving instructions.

Communication Protocol Handling:
The master sends commands or queries to the slave devices using the chosen communication protocol. This could involve sending data, requesting information, or instructing the slaves to perform specific actions.

Slave Response:
Upon receiving a command or request, each slave processes the information and responds accordingly. Responses may include sending data back to the master or acknowledging the successful execution of a command.

Error Handling:
Both master and slave devices implement error-handling mechanisms to manage communication failures, ensuring the reliability of the system.

V.CONCLUSION

The master-slave concept in the context of Arduino provides a versatile and scalable framework for communication and coordination between devices in embedded systems. This communication paradigm is particularly useful for scenarios where one central device (the master) needs to control and interact with one or more peripheral devices (the slaves). The simplicity and accessibility of Arduino make it an ideal platform for implementing master-slave configurations in various applications.

References

[1] *Chithrashree G.S., 1Divya N.M., 1Shalmali S. Mankikar, 2Dr. Sindhu Sree M., 3Dr. Pavithra G., 4Dr. T.C.Manjunath , "Wireless LoRa Communication Between Two Arduino Uno for Military Application inSoldier Tracking1Apeksha U"., Tuijin Jishu/Journal of Propulsion Technology ISSN: 1001-4055 Vol. 44 No. 3(2023) 768*

[2] *Haiping Zhou, Yuxiang Yang, Zhekang Dong, Zhiwei He ,"An intelligent master–slave collaborative robot system for cafeteria service",F L 2015 Sistemas Distribuidos (UAM colecci´on una decada) ISBN 9786072804760 URL http://dccd.cua.uam.mx/libros/texto/texto_pdf.php?url=LopezFuentes-SistemasDistribuidos*

[3] *White E 2011 Making Embedded Systems: Design Patterns for Great Software Oreilly and Associate Series (O'Reilly Media, Incorporated) ISBN 9781449302146 URL https://books.google.com.mx/books?id= bI8w17SyNdYC*

[4] *Sastry J, Ganesh J V and Bhanu J S 2015 Indian Journal of Science and Technology 8 cited by: 19; All Open Access, Bronze Open Access*

[5] *Phillips Semiconductors 2003 AN10216-01 I2C Bus Phillips Semiconductors Eindhoven, Netherlands an 10216 ed Frankowiak M R, Grosvenor R I and Prickett P W 2005 Microprocessors and Microsystems 29 189 – 196.*

[6] *Ibrahim D 2014 PIC Microcontroller Projects in C: Basic to Advanced (Elsevier Science) ISBN 9780080999678 URLhttps://books.google.com.mx/books?id=xQajAgAAQBAJ%7D*

[7] *L.T. Handoko,, "Public Cluster : web-based open parallel machine", Indonesian Copyright No. B 268487 (2006).*

[8] *Z. Akbar et.al., "LIPI Public Cluster", http://www.cluster.lipi.go.id.*

[9] *Z. Akbar, Slamet, B.I. Ajinagoro, G.J. Ohara, I. Firmansyah, B. Hermanto and L.T. Handoko, Open and Free Cluster for Public, Proceeding of the International Conference on Rural Information and Communication Technology, 6–7 August 2007.*

[10] *10. Z. Akbar, Slamet, B.I. Ajinagoro, G.J. Ohara, I. Firmansyah, B. Her- manto and L.T. Handoko, Public Cluster : parallel machine with multi-block approach, Proceeding of the International Conference on Electrical Engineering and Informatics, 17 June 2007.*

[11] *Z. Akbar and L.T. Handoko, "Web-based Interface in Public Cluster", Proceeding of the 9th International Conference on Information Integration and Web-based Applications and Services, 3–5 December 2007.*