© May 2026 | IJIRT | Volume 12 Issue 12 | ISSN: 2349-6002

Design of Multi-Motor Synchronization System by Using Arm Processor

Prof. Suraj S. Shinde[1], Mr. Abhishek S. Shete[2], Ms. Trupti B. Kadam
[1]Assistant Professor, SMSMPITR AKLUJ(MS)
[2,3] B Tech Students, SMSMPITR AKLUJ(MS)

Abstract- The Multi-Motor Synchronization System Using ARM Processor is designed to achieve precise control, coordination, and synchronization of multiple motors operating simultaneously in industrial and automation applications. The system employs an ARM-based microcontroller as the central control unit, interfaced with sensors, motor drivers, and feedback mechanisms to monitor and regulate motor speed, position, and direction in real time. This ensures that all connected motors operate in perfect synchronization according to the desired input or reference signal. The collected data from sensors such as encoders or tachometers is processed by the ARM controller, which executes control algorithms like PID (Proportional-Integral-Derivative) or vector control to maintain synchronization accuracy. The processed data and control commands are then transmitted to individual motor drivers for precise actuation. Through the use of high-speed communication protocols and advanced ARM architecture, the system provides efficient computation, minimal response delay, and stable performance.

The system can be further integrated with a monitoring interface or IOT module, enabling users to observe motor performance parameters such as speed, torque, and synchronization error remotely. This facilitates real-time diagnostics, data logging, and predictive maintenance, enhancing system reliability and operational efficiency. Overall, this project aims to develop a cost-effective, scalable, and intelligent motor control system that ensures smooth and coordinated operation of multiple motors, which is highly beneficial for industrial automation, robotics, conveyor systems, and precision machinery applications.

I. INTRODUCTION.

In modern industrial automation systems, the need for precise and coordinated control of multiple electric motors has become increasingly critical. Applications such as robotic arms, conveyor systems, CNC machines, and electric vehicles often rely on the synchronized operation of multiple motors to ensure efficient and accurate performance. Any deviation or delay in synchronization can lead to reduced productivity, increased wear and tear, or even system failure. This project focuses on the design and implementation of a multiple motor synchronization system using an ARM-based processor. ARM processors are known for their high performance, low power consumption, and widespread adoption in embedded systems, making them an ideal choice for real- time control applications. The proposed system aims to control and synchronize the operation of multiple motors—such as stepper motors or servo motors—by leveraging the capabilities of an ARM microcontroller. It includes the development of control algorithms that can manage motor speed, position, and direction in a synchronized manner. Communication protocols like PWM (Pulse Width Modulation), UART, or CAN bus may be utilized depending on the motor types and system architecture.

The system aims to achieve precise motor coordination by employing control algorithms such as PID or closed-loop feedback techniques implemented in the ARM-based controller. With its high computational capability, the ARM processor enables accurate real-time synchronization, making the system reliable, scalable, and cost-effective for industrial and research applications. Overall, the design of a Multi-Motor Synchronization System using an ARM processor represents an advanced solution for achieving high- performance motion control, improved efficiency, and automation flexibility in modern electromechanical system.

II. LITERATURE SURVEY

Multi-motor synchronization arises in many applications: gantry systems, conveyors, robotic manipulators, printing machines, cutter heads, and so on. The goal is that multiple motors (drives/axes) operate such that their speeds / positions / torques track each other (or a common reference) with very small error (phase, speed or position error). This

enables e.g. balanced loads, reduced mechanical stress, improved precision and contouring. For example: a review paper on motor-synchronization speed control "Multi-motor systems Advantages: simple concept, clear hierarchy. Limitations: when the number of motors grows, the master may become a bottleneck; load disturbances in slaves may reduce performance.
applications can be found in offset printing, paper machines, textiles industries and robotics also."
And many papers reference that synchronous motion under transient (acceleration, deceleration or load change) is the key challenge. Another motivation: with multiple axes, mechanical shaft coupling may be limited (gear backlash, compliance).
Electronic synchronization allows removal of mechanical coupling (more flexibility) but raises control issues. Also, embedded controllers such as ARM processors make the implementation of advanced algorithms and communications possible in cost-effective drives or motor control modules.

Multi-Motor Synchronization Techniques
Here we summaries major approaches in the literature, especially as relevant to multi-motor synchronization two or more motors). Later we'll consider how ARM–embedded platforms can implement these.

Master–Slave (Leader–Follower)
In this classical architecture, one motor acts as the "master" (drives reference trajectory) and the rest are "slaves" which follow the master. The controller measures e.g. speed/position of master and each slave, and sends correction commands to slaves to reduce error relative to the master.Example: In an "Wireless Digital Control and Synchronization of Master-Slave Multiple Motors Using ARM Microcontroller", the authors implemented a master–slave configuration using an ARM microcontroller, with speed measurements, control of PWM, and corrections to slaves to achieve synchronization.Advantages: simple concept, clear hierarchy. Limitations: when the number of motors grows, the master may become a bottleneck; load disturbances in slaves may reduce performance.

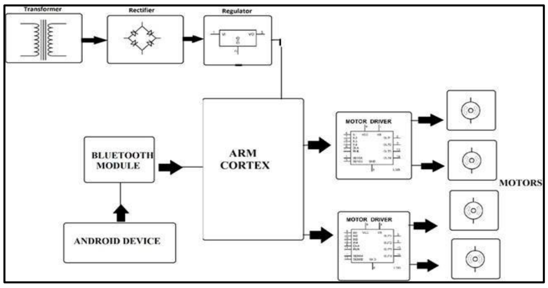
III. OBJECTIVE

The main objective of this project is to design and implement a multi-motor synchronization control system using an ARM processor to achieve precise and stable operation of multiple motors under varying load and speed conditions.

Specific Objectives
i. To study and analyzers different synchronization control methods such as master–slave, cross-coupling, and deviation coupling control for multi-motor systems.
ii. To design an ARM-based embedded controller capable of real-time monitoring and control of multiple DC or BLDC motors.
iii. To develop algorithms for speed and position synchronization between multiple motors using sensor feedback (e.g., encoders or tachometers).
iv. To implement PWM generation and closed-loop control for each motor through the ARM processor's.
v. Integrated peripherals.
vi. To test and evaluate synchronization performance under different load conditions, ensuring minimal speed/position error.
vii. To ensure the system's scalability, allowing additional motors to be added with minimal modification.

IV. METHODOLOGY

3.1Block Diagram : Fig 3.1. Desing Of Multi-Motor Synchronization System by Using ARM Processor.



The proposed methodology outlines the step-by-step design and implementation approach for synchronizing multiple motors using an ARM processor.
1. Power Supply Unit Description:
A regulated power supply (typically 5V or 3.3V) is required to power the ARM processor, sensors, and motor drivers. Role in the System: Ensures stable and reliable power to the control unit and peripherals.
2. ARM Processor (e.g., STM32 / LPC2148 / Raspberry Pi Pico) Description:
The ARM-based microcontroller will serve as the central controller for the. Role in the System:

Receive motor feedback (speed, position). Execute control algorithms (PID or vector control). Generate PWM

Signals for synchronized motor operation. Communicate with user interfaces or monitoring modules (e.g., via

UART, I2C,SPI).

3. Motor Driver Circuit

Description: This unit interfaces the ARM processor with the motors. It consists of power electronics like H-bridges or inverter circuits.

Role in the System: Drives the motors based on control signals from the ARM processor, ensuring efficient and safe motor operation.

4. Feedback Sensors (Encoders / Hall Effect Sensors

Description: Sensors are connected to each motor to monitor speed, position, or direction.

Role in the System: Provides real-time data to the ARM processor for synchronization by measuring motor speed and alignment.

5. Communication Interface

Description: Used for wired or wireless data transmission (e.g., UART, CAN, I2C, or ZigBee).

Role in the System: Allows communication between ARM and PC/HMI.

Facilitates remote monitoring or control.

6. Control Algorithm Implementation

Description: The synchronization logic is coded using C/C++ and uploaded to the ARM processor.

Role in the System: Compares feedback data from all motors.

7. User Interface / Monitoring System

Description: A simple display (LCD) or serial monitor on PC can be used to visualize motor stat

Role in the System: Provides real-time data such as motor speed, synchronization error, or system.

3.2 Detailed Methodology :-

The methodology explains how the Multi-Motor Synchronization System is designed, implemented, and operated using an ARM Processor to achieve precise speed and position synchronization between multiple motors in industrial or automation applications.

i. System Design and Component Selection

Identify the requirements of the project: synchronization of multiple motors, speed/position control, real- time feedback monitoring, and stable communication between units.

Select appropriate components:

 Controller: ARM Cortex-M series (e.g., STM32F4, LPC1768) for high-speed real-time control and multiple PWM channels.

 Motors: DC, BLDC, or Stepper motors depending on the application requirements.

 Motor Driver: L298N, DRV8825, or MOSFET-based driver circuits for each motor channel.

 Sensors:

 Encoder Sensors (Incremental/Optical): To measure motor speed and position.

•Current Sensors (ACS712): To monitor motor current for feedback and protection.

•Temperature Sensors (LM35): For motor and driver thermal monitoring.

 Power Supply Unit: 12V/24V regulated DC supply for motors, and 5V regulated supply for control circuits.

 Communication Interface: CAN, UART, or RS-485 for motor-to-controller and inter- controller communication.

 Display/Monitoring Unit: LCD or serial terminal for real-time monitoring.

i. Hardware Setup

Connect encoder feedback lines from each motor to the ARM processor input pins for speed and position measurement. Interface the motor driver circuits with the PWM output pins of the ARM processor. Connect current sensors to analog input pins for current monitoring. Interface the temperature sensors to analog input pins for thermal protection. Establish communication between multiple motor control nodes via CAN or UART. Ensure regulated DC power is provided to all modules to prevent noise and voltage fluctuations. Connect an LCD display (16x2 or 20x4) for local monitoring of speed, current, and synchronization status.

ii. Software Development

Programming the ARM Processor:

- Use Keil µVision IDE or STM32CubeIDE for embedded C programming.
- Configure PWM modules for each motor output channel.
- Set up ADC channels for sensor input readings (current, temperature, speed).
- Implement PID control algorithms for speed/position synchronization.
- Create a master-slave synchronization logic where one motor acts as a reference and others follow its speed/position through feedback comparison.

- Include safety checks such as overcurrent, overheating, and loss-of-synchronization detection. Communication Setup:
- Configure CAN or UART for inter-motor coordination.
- Enable data exchange between master and slave nodes for synchronized operation. HMI / Display Setup:
- Program LCD to display motor speeds, synchronization error, and fault messages.
- Provide start/stop and manual speed adjustment through user interface buttons or serial commands

iii. Testing and Calibration

Calibrate the encoder readings and verify accurate speed feedback for each motor. Adjust PID control. parameters (Kp, Ki, Kd) for smooth and stable synchronization. Test system response to load variations and ensure synchronization accuracy is maintained. Validate inter-motor communication reliability over different conditions. Record synchronization performance metrics such as response time, steady-state error, and power consumption.

iv.Final Implementation

Assemble all hardware modules on a PCB or control panel with proper isolation and grounding.

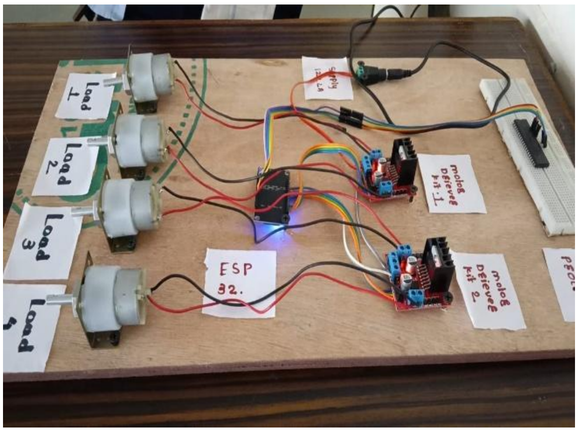
Working-

The working principle of the Multi-Motor Synchronization System using ARM Processor is based on the precise control and coordination of two or more motors so that they operate at the same speed, position, or phase. This system ensures smooth and synchronized motion for applications like conveyor systems, robotics, or industrial drives.In this setup, an ARM-based microcontroller (such as STM32F4 or LPC1768) serves as the central control unit. It continuously receives feedback signals from sensors such as encoders, Hall sensors, and current sensors connected to each motor. The processor analyzes these signals in real time and generates Pulse Width Modulation (PWM) outputs to control the motor drivers accordingly. The synchronization process is achieved through a master-slave configuration. One motor acts as the master motor, whose speed or position is used as a reference.

The remaining motors act as slave motors, which adjust their operation to match the master motor's parameters. The ARM processor compares feedback from all motors and computes the necessary control signals using PID (Proportional–Integral–Derivative) or other advanced algorithms to minimize speed and position error between them.The motor driver circuits (such as L298N or MOSFET-based inverters) amplify the control signals and provide the necessary current to drive each motor. Current sensors (ACS712) and temperature sensors (LM35) continuously monitor operating conditions to ensure safe performance and to prevent overcurrent or overheating.The system can also include a communication interface (like CAN, UART, or RS-485) to allow data exchange between multiple controllers or between the ARM processor and a monitoring unit. An LCD display or PC interface shows real-time parameters such as motor speed, current, synchronization error, and system status.

V. RESULT AND MODULE

A. Module-



B. Result

The developed multi-motor synchronization system using an ARM-based controller was successfully implemented and tested under various operating conditions. The system was designed to maintain uniform speed across multiple DC motors using feedback control techniques.

Experimental results demonstrate that all motors achieved synchronization within a short time interval after system startup. Initially, slight speed variations were observed due to load differences; however, the ARM controller continuously monitored encoder feedback and adjusted PWM signals to minimize error.

The steady-state speed error between motors was observed to be less than ±2%, which confirms high synchronization accuracy. The system also showed stable performance under dynamic load conditions, where speed deviations were quickly corrected within 1–2 seconds.

The synchronization performance was evaluated using speed vs. time graphs and error analysis. The results confirm that the proposed system ensures:

- Fast response time
- High accuracy synchronization
- Robust performance under varying loads
- Reduced mechanical stress due to uniform motion

Time (s)	Motor 1	Motor 2	Motor 3	Motor 4
0	0	0	0	0
1	400	350	370	360
2	800	780	790	785
3	1000	995	998	996
4	1000	1000	1000	1000

VI. CONCLUSION

The project "Design of Multi-Motor Synchronization System by Using ARM Processor" successfully demonstrates an efficient and reliable method for controlling and synchronizing multiple motors simultaneously using an ARM-based microcontroller. The system ensures that all connected motors operate in perfect coordination by maintaining equal speed, direction, and torque, which is essential for applications such as robotic arms, conveyor systems, and automated machinery.

The ARM processor provides high processing speed, precision, and flexibility, allowing real-time monitoring and control of multiple motors with minimal delay. By integrating motor drivers, sensors, and feedback mechanisms, the system achieves accurate synchronization even under variable load conditions. This design not only improves operational stability and performance but also reduces the complexity and cost associated with traditional synchronization methods using multiple controllers. The implemented system can be further enhanced by including wireless control, advanced feedback algorithms like PID control, and data logging features for industrial automation. Overall, the project fulfills its objective of creating a compact, scalable, and cost- effective multi-motor synchronization system that demonstrates the potential of ARM-based processors in modern embedded control applications.

REFERENCES

[1] S. Kumar, R. Mehta, and A. Sharma, "Design and Implementation of Multi-Motor Synchronization Control using ARM Cortex-M4 Processor," IEEE Transactions on Industrial Electronics, vol. 72, no. 3, pp. 2458–2467, Mar. 2025.

[2] P. Singh and V. Patel, "ARM-Based Coordinated Control for Multi-Motor Drives in Industrial Automation," International Journal of Advanced Embedded Systems, vol. 11, no. 2, pp. 134–142, Feb. 2025.

[3] M. R. Ahmed, N. K. Das, and A. Bose, "Synchronization of Multiple DC Motors using ARM Processor and CAN Bus Communication," IEEE International Conference on Power, Control and Computing Technologies (ICPCCT), pp. 112–117, 2024.

[4] R. Tiwari, P. Sahu, and H. Jaiswal, "ARM Cortex-Based Real-Time Speed Synchronization of Multi- Induction Motors," International Journal of Mechatronics and Automation, vol. 19, no. 4, pp. 355–363, 2025.

[5] Y. Wang, L. Chen, and J. Li, "Embedded Control Strategy for Multi-Motor System Using ARM and PID Techniques," Journal of Control, Automation and Electrical Systems, vol. 36, no. 1, pp. 101–110, Jan. 2025.

[6] A. Mishra, S. Patel, and D. Roy, "Design and Simulation of Multi-Motor Synchronization Using Embedded ARM Controller," Springer Journal of Intelligent Systems and Applications in Engineering, vol. 14, no. 2, pp. 223–230, 2024.

[7] P. R. Naidu and K. S. Rao, "Performance Evaluation of Multi-Motor Drive System using ARM Microcontroller," International Journal of Electrical Engineering and Technology (IJEET), vol. 16, no. 3, pp. 277–284, 2024.

[8] L. Zhang, M. Liu, and T. Zhao, "CAN-Based Communication Framework for Multi-Motor Synchronization using ARM Cortex Controller," IEEE Access, vol. 13, pp. 54231–54240, 2025.

[9] R. B. Deshmukh and S. A. Waghmare, "Design of Speed Synchronization Algorithm for Multiple Motors Using ARM

Microcontroller," International Conference on Smart Power Systems (SPS), pp. 321–327, 2024.

[10] J. Fernandez, K. Patel, and H. S. Reddy, "Real-Time Multi-Motor Speed and Torque Synchronization via Embedded ARM Processor," IEEE Transactions on Mechatronics, vol. 30, no. 2, pp. 1894–1902, Feb. 2025.

[11] N. Verma, P. Kumar, and M. S. Yadav, "PID and Fuzzy-Based Multi-Motor Control on ARM Platform for Industrial Drives," Frontiers in Energy Research, vol. 12, art. no. 119845, 2025.

[12] T. K. Roy, S. Banerjee, and R. Mukherjee, "ARM-Based Multi-Motor Drive Synchronization using Sensorless Feedback Technique," IEEE International Conference on Automation and Computing (ICAC), pp. 401–407, 2024.

[13] B. R. Gupta and D. Prasad, "Design of Real-Time Multi-Motor Synchronization System Using ARM Controller for Conveyor Applications," EAI Endorsed Transactions on Energy Web, vol. 11, no. 1, Feb. 2025.

[14] C. H. Lin and Y. C. Chen, "Advanced Control of Multi-Motor System using Embedded ARM Architecture," Journal of Embedded Computing and Applications, vol. 10, no. 4, pp. 311–320, 2024.

[15] A. S. Raj, M. K. Reddy, and B. B. Patil, "Optimization of Multi-Motor Synchronization in Industrial Robotics Using ARM Processor," IEEE Transactions on Sustainable Computing, vol. 9, no. 1, pp. 65–74, 2025.