Industrial Robotic Animatronic with PLC-Based Control System

Austin Berg, Melvin Guzman, Tony Torres, Armando Diaz

Department of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

Abstract — **This paper presents an industrial animatronic system leveraging Programmable Logic Controllers (PLCs) and Human-Machine Interfaces (HMIs) to provide scalable, reliable, and cost-effective robotic motion control. Addressing the limitations of proprietary systems, the project demonstrates open-integration methodologies using industry-standard automation tools tailored for creative and educational animatronic applications. Utilizing Modbus TCP communication, real-time feedback, robust fault protection, and intuitive interfaces, the system ensures precise multi-axis motion control. The paper details component selection, integration strategies, and extensive performance evaluations, highlighting the adaptability, expandability, and industrial reliability that distinguish this approach within entertainment, education, and research settings.**

Index Terms — **Animatronics, Automation, Ethernet, Human Machine Interfaces, Programmable Logic Controllers, Robotic Control Systems.**

I. Introduction

Animatronics, combining robotics with artistic expression, have significantly enhanced storytelling and audience immersion in entertainment settings such as theme parks and museums. Despite their popularity, the widespread adoption of advanced animatronic technology remains limited due to proprietary hardware, complex integration, and high costs. Market-leading companies like Disney and Universal dominate this space with proprietary, high-performance systems leveraging advanced Programmable Logic Controllers (PLCs) and specialized Human-Machine Interfaces (HMIs). These solutions, although reliable and sophisticated, are costly and inflexible, restricting their accessibility to smaller venues, educators, and independent creators. Conversely, lower-cost alternatives typically lack the precision, reliability, and safety required for professional applications.

To bridge this gap, our project develops a modular, scalable, and cost-effective animatronic system using broadly accessible industrial automation components. By replacing proprietary controllers with standard Allen-Bradley PLCs and integrating intuitive HMIs, our design prioritizes reliability, maintainability, and affordability. Employing open industrial protocols such as Modbus TCP and Ethernet/IP, the system facilitates easy integration and future expansion, significantly lowering entry barriers.

The main objectives include delivering precise, synchronized movements through industry-standard technologies, ensuring scalability and ease of customization without substantial additional investment. Our open integration approach empowers smaller enterprises, educators, and independent creators to access professional-grade animatronics without proprietary restrictions.

Ultimately, this paper proposes an innovative solution leveraging standard PLCs and HMIs, aiming to democratize advanced animatronic capabilities and foster wider accessibility and collaborative innovation in entertainment, education, and small-scale commercial applications.

II. System Components

Understanding the animatronic system begins with examining its key components. This section offers an overview of each primary component, including the Microcontroller Unit (MCU), Programmable Logic Controller (PLC), Animatronic Motors, and Human-Machine Interface (HMI).

A. Microcontroller Unit

The MCU functions as the central processing hub, managing all lower-level hardware interactions. Specifically, the SAMD21 MCU was chosen for its efficient processing capabilities, extensive peripheral support, and suitability for embedded applications. It directly controls and coordinates motor driver signals, processes sensor inputs, and facilitates real-time communication with other system components. Additionally, the MCU handles critical timing operations, ensuring smooth and synchronized movements essential for animatronic realism.

B. Programmable Logic Controller

The PLC is employed as the primary industrial-grade control system, selected for its reliability, robust performance, and compatibility with established automation protocols. Specifically, the Micro820-AWB PLC from Allen-Bradley orchestrates the high-level motion logic, manages complex control sequences, and monitors system safety protocols. Its ability to execute precise control profiles, conduct fault detection, and support redundancy ensures consistent, safe, and repeatable operation of the animatronic movements.

C. Animatronic Motors

The animatronic system uses two types of motors for precise motion control: DYNAMIXEL XC-330-T288-T, 2XL430-W250-T, XL430-W250-T servo motors and a NEMA 17 stepper motor driven by an A4988 driver. The servo motors, with integrated feedback mechanisms, provide accurate position control, smooth movements, and reliable torque output for dynamic performance. The stepper motor is specifically utilized for movements that demand exact incremental positioning, complementing the servos and enhancing the versatility of the animatronics' mechanical actions.

D. Human-Machine Interface

The HMI offers an intuitive, user-friendly interface for system interaction and monitoring. Equipped with an Ethernet-enabled touch display, it provides operators with real-time system status, operational controls, and diagnostic capabilities. Employing Modbus TCP and Ethernet/IP communication protocols, the HMI seamlessly interacts with the PLC and MCU, allowing operators to initiate and adjust motion sequences, monitor ongoing performance, and manage safety and system configuration settings effectively.

III. System Architecture

The system architecture defines how control logic, interface design, communication protocols, and actuation strategies are integrated to support real-time animatronic operation. Emphasis is placed on deterministic execution, modular design, and fault resilience, forming a foundation for precise motion control and safe system behavior under varying conditions.

A. PLC Logic Design

At the heart of the control system is the Allen-Bradley Micro820 Programmable Logic Controller (PLC), programmed through Connected Components Workbench (CCW). The PLC executes structured ladder logic and function block routines to manage animatronic behavior, each routine corresponding to distinct motion sequences or sensor responses. For instance, when a pre-defined motion is triggered via the HMI, the PLC evaluates positional feedback from motors, verifies safety inputs (such as emergency stop status), and initiates output signals accordingly. Each axis of motion, driven by servos or steppers, is mapped to independent PID logic that adjusts in real time, based on sensor feedback. This closed-loop control ensures synchronized motion and accurate positional execution across multiple degrees of freedom. The PLC also incorporates interlocking routines and state-based transitions to manage motion dependencies and prevent conflicting operations.

B. HMI Implementation

The Human-Machine Interface (HMI) is developed using Inductive Automation's Ignition Edge, a lightweight, modular platform designed for edge computing and industrial control applications. The system is hosted on an embedded device running a local gateway, which serves as the central node for visualization, control, and diagnostic access.

Custom-designed views display motor telemetry, system states, operational mode, and fault alerts. Bidirectional communication between the HMI and PLC is achieved using Modbus TCP, allowing user interactions, such as sequence initiation, parameter adjustment, and emergency stop acknowledgment, to propagate directly into PLC memory registers. These tags are bound to UI components in Ignition's Perspective module, ensuring a synchronized and responsive interface. Operators interact through intuitive controls, including buttons, sliders, and indicators, with real-time status updates visualized through dynamic color-coding and alert pop-ups.

C. Communication Infrastructure

A robust communication infrastructure underpins the coordination between the PLC, HMI, and motor control systems. The network is arranged in a star topology, centered around an unmanaged Ethernet switch powered at 24VDC. This switch serves as the central hub, linking each node: PLC, HMI, and SAMD21 microcontroller, using Cat 6 Ethernet cables. These cables support 1 Gbps data rates and offer superior shielding against electromagnetic interference, minimizing latency and ensuring reliable transmission in electrically noisy environments.

The system employs Modbus TCP as the primary communication protocol, enabling deterministic and structured data exchange without the constraints of proprietary ecosystems. Modbus TCP allows each device to communicate through a defined set of registers for commands, telemetry, and control status. Every node is assigned a static IP address, simplifying addressing and ensuring consistent routing behavior across power cycles. This setup allows the HMI to read and write control parameters directly to the PLC, while the SAMD21 microcontroller retrieves motion commands and reports motor telemetry over the same network.

At the hardware level, the W5500 Ethernet controller, interfaced with the SAMD21, manages low-level packet handling, translating Modbus register access into actionable motor commands. In parallel, TTL and UART serial interfaces connect to local sensors and smart servos, providing supplemental communication paths for real-time positional feedback and diagnostics. This multi-layered architecture enables distributed processing and synchronization across all components, ensuring precise timing and reliable system-wide communication.

D. Animatronic Configuration

The animatronic body integrates both rotary and linear actuation via three primary motor types. DYNAMIXEL XC-330-T288-T, 2XL430-W250-T, and XL430-W250-T smart servos are embedded at joints requiring high-resolution angular control, such as the head and arm axes. These servos communicate via TTL half-duplex and provide real-time telemetry, including position, load, voltage, and temperature. For broader-range motions, such as torso rotation, a NEMA 17 stepper motor is driven by an A4988 module under current-limited micro-stepping. The PLC issues motion initiation flags, while the SAMD21 computes position targets and translates them into appropriate command packets for each motor. A dedicated feedback loop continually monitors motor position and corrects for drift or lag, ensuring smooth transitions and realistic articulation.

E. Fault Detection and Protection

Robust fault handling is implemented within the PLC's safety logic layer. Each subsystem reports status through discrete I/O lines or over Modbus registers. The logic continually evaluates motor current thresholds, communication timeouts, sensor validity, and positional bounds. Upon detecting an anomaly, such as motor stall, voltage drop, or inconsistent feedback, the PLC enters a controlled fault state, halts motion outputs, and triggers both visual (HMI warnings) and physical indicators (status LEDs). Faults are categorized with specific codes, logged with timestamps, and retained in non-volatile memory for post-event analysis. Operators can clear faults manually via the HMI after system reset and validation. This strategy ensures operational continuity while minimizing damage and downtime.

F. Safety Mechanisms

Multiple layers of safety are embedded throughout the system. A hard-wired E-stop loop interfaces directly with the PLC's safety inputs, bypassing logic processing and forcing immediate coil de-energization across motor drives. This ensures physical disconnection from power regardless of software state. Lock-out/tag-out (LOTO) logic is embedded into startup routines, requiring manual acknowledgment and reset after an E-stop event. Circuit-level protections include inline fuses for each motor channel, thermally rated breakers on power rails, and buck-converter voltage regulation for 12V and 24V domains. In addition, the enclosure complies with NEMA-4 standards, ensuring environmental protection and personnel safety. Together, these measures satisfy industrial-grade safety compliance and foster reliable operation during development and deployment.

IV. HARDWARE DESIGN

The hardware framework of the animatronic system consists of a purpose-built electronics platform structured around multiple custom PCBs, each designed to fulfill a distinct role within the control and power architecture.

A. Printed Circuit Board (PCB)

The primary control PCB integrates the SAMD21 microcontroller with dedicated subsystems for programming, communication, and peripheral I/O. USB-C connectivity is routed through ESD protection circuitry to a native USB interface for firmware flashing and debugging, while a discrete reset line and status LEDs provide development-time feedback. A 12 MHz crystal oscillator ensures stable clock generation for synchronous communication tasks, particularly SPI and UART. All GPIOs are properly decoupled with ceramic capacitors and grouped by function: digital I/O, UART, SPI, and PWM, routed to keyed headers for sensor, actuator, and interface expansion.

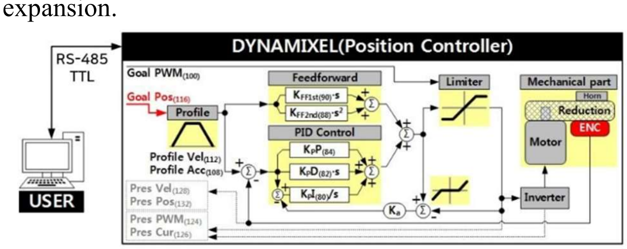

Fig 1. DYNAMIXEL TTL Position Control

The PCB responsible for TTL motor signaling includes a half-duplex transceiver for DYNAMIXEL servo communication. The transceiver toggles direction control signals via the SAMD21's digital output, ensuring proper bus arbitration and avoiding contention across the shared UART line. Pull-up resistors and termination components maintain clean signal edges, while diagnostic pins expose the RX/TX state for logic-level debugging. Stepper motor pulse generation is handled through microcontroller-controlled direction and step pins, isolated from the load stage by logic buffers and routed directly to an external A4988 driver via screw terminals.

Ethernet communication is managed on a dedicated PCB containing the W5500 Ethernet controller configured in SPI mode. A 25 MHz crystal oscillator provides the clock source for Ethernet MAC timing, while SPI signals from the SAMD21 are buffered and filtered to minimize signal integrity issues. The RJ45 jack includes integrated magnetics for signal coupling and LED indicators for link and activity status. Protection diodes across differential pairs defend against electrostatic discharge and surge events, while decoupling capacitors and power filter stages suppress switching noise from entering the communication layer.

B. Power Distribution and Protection

Power is distributed across a dual-rail system: 24VDC for industrial components such as the PLC and Ethernet switch and stepped-down 12V/5V/3.3V rails for logic and motor control. All power enters through a fused terminal block, where discrete blade fuses isolate each subsystem branch. High-side current protection is implemented using polyfuses (PTCs) on low-voltage rails, while each branch includes ceramic bypass capacitors to suppress noise and absorb transient spikes.

Motor power is distributed through current-limited drivers and protected by flyback diodes to handle inductive loads. The stepper motor is powered through an A4988 driver operating on a 12V line, with onboard potentiometer control for current limiting. Servo lines are protected with in-line fuses and ferrite beads to prevent conducted EMI from affecting control logic.

Power status is visually represented through LED bar indicators placed at each major output junction. A power supervisory circuit monitors voltage rails and flags under-voltage or brownout conditions back to the PLC, triggering a fault response if any supply becomes unstable.

C. Voltage Regulation

The voltage regulation scheme employs a dedicated LM2576HV buck converter circuit capable of converting the 24V industrial input to stable 12V and 5V outputs. This regulator supports up to 3A continuous current with over-temperature and over-current protection built in. Key design considerations included input/output filter capacitors with low ESR, a toroidal inductor rated for low EMI, and a Schottky catch diode for fast switching recovery.

Each regulator output feeds into a dedicated power plane on the PCB, decoupled through bulk electrolytic and ceramic capacitors. Ferrite beads are placed at the output stage of the 5V rail to prevent switching noise from entering the microcontroller domain.

Thermal management is addressed through enlarged copper areas around the LM2576HV's power pad and the use of thermal vias to transfer heat to the back layer. This passively spreads heat and prevents localized hot spots during high-current operation.

D. Interconnects and Interface Layout

All signal and power connections are routed to locking JST connectors or screw terminals, ensuring secure engagement under vibration. GPIOs from the SAMD21 are broken out for future expansion, with labeled headers for I^2C, SPI, PWM, and analog signals. Signal routing was performed with careful attention to return paths and minimal loop area, especially for high-speed communication and analog lines. PCB edge spacing and connector clearance were designed to align with DIN rail enclosure dimensions. This layout standardization allows for simple panel mounting and organized internal wiring, essential for both development and maintenance.

E. Enclosure and System Integration

All PCBs are housed within a sealed NEMA 4-rated enclosure, with cutouts for Ethernet, USB, and power connections routed through grommet-sealed knockouts. A modular mounting plate holds each PCB in fixed position, with standoffs providing mechanical isolation and airflow clearance. Internal cable routing is managed through raceways and cable ties, minimizing stress on solder joints and reducing electromagnetic cross-talk between power and logic paths.

V. Software Design

The software architecture is constructed to support deterministic control, modular expansion, and efficient coordination between system components. Development is divided across the microcontroller firmware, HMI logic, and the configuration and execution of motion profiles.

A. MCU Programming

The SAMD21 microcontroller is programmed in C using Arduino IDE with Arduino community libraries. The firmware architecture is interrupt-driven and modular, divided into subsystems for communication, motor control, sensor monitoring, and diagnostics. Peripheral initialization is handled in a structured boot sequence, configuring the SPI interface for W5500 communication, UART for DYNAMIXEL control, and GPIOs for stepper direction and pulse outputs.

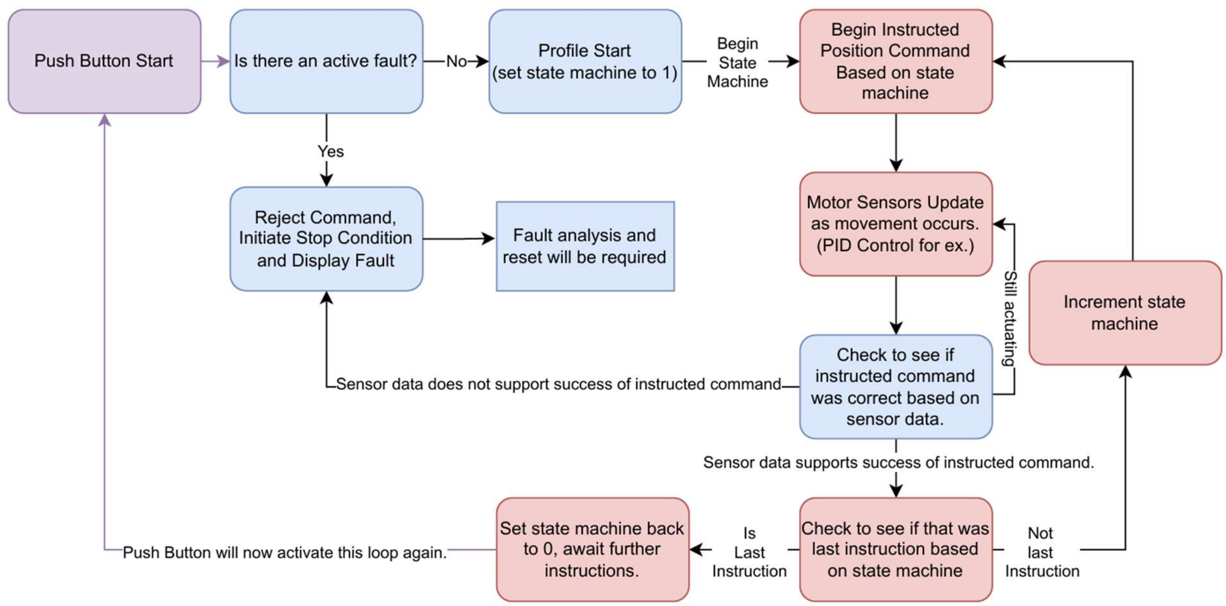

Fig 2. Animatronic State Machine Logic Diagram

Real-time behavior is achieved through a combination of hardware timers and software state machines. One timer is configured for millisecond tick generation, used for task scheduling and motion timing, while additional timers generate PWM-based pulse trains for the stepper motor. UART transactions to the DYNAMIXEL servos are managed through a half-duplex protocol driver, which toggles transmission direction and waits for servo acknowledgments. A command queue system processes motion commands received via Modbus TCP, parsing register data and mapping it to discrete motor actions.

Data integrity and system robustness are enforced through watchdog timers, CRC-checked communication buffers, and timeout logic for all external transactions. System status, including motor positions, error flags, and sensor feedback, is serialized into a compact Modbus register map and published cyclically to the PLC and HMI. Debug-level output is enabled through conditional compilation, allowing serial terminal access for live diagnostics during development.

B. HMI Software

The HMI is developed in Inductive Automation’s Ignition Edge platform using the Perspective module. The interface is constructed as a responsive, tag-driven web application, rendered locally on an edge device and accessible through touchscreen or networked clients. Each visual component: buttons, indicators, status fields, is dynamically bound to live PLC tags over Modbus TCP.

The HMI design consists of distinct navigation views: system overview, manual/maintenance mode, data analysis, and fault/alarm diagnostics. Input actions trigger tag writes, which propagate to the PLC and subsequently to the MCU via mapped Modbus registers. For example, selecting a motion routine sets a designated register, prompting the PLC to initiate a command dispatch to the SAMD21. Feedback such as current motor positions, system uptime, and active fault codes are pulled and updated on the interface at one-second intervals.

To facilitate usability, the interface integrates touch-friendly controls, color-coded status indicators, and confirm prompts for critical actions like emergency reset or reinitialization. System faults are displayed with descriptive codes and timestamps, and maintenance features allow operators to clear fault states or initiate homing sequences directly from the panel. Authentication layers can be configured to restrict control access, ensuring safe operation in multi-user environments.

C. Motion Profiles

Motion profiles define the position trajectories, timing, and coordination between each axis of the animatronic system. Profiles are stored as pre-programmed sequences in the MCU firmware, each represented as a discrete table of position-time pairs or step instructions. Upon receiving a motion profile ID, the MCU executes a sequenced routine, updating motor positions at fixed intervals using timer interrupts to maintain temporal precision.

DYNAMIXEL servos are controlled through position write commands targeting specific register addresses. The MCU performs interpolation between profile keyframes to smooth acceleration and deceleration, minimizing mechanical stress and enhancing motion realism. Each servo's onboard PID controller is configured with tuned gain parameters to balance responsiveness with overshoot suppression.

For the stepper motor, the MCU generates variable-frequency pulse trains based on motion timing and target displacement. Ramping functions are implemented to reduce torque loss at startup and improve step accuracy. The motion logic accounts for direction changes and ensures coordinated execution when multiple axes are activated simultaneously.

Profiles are written to be modular and reusable, allowing the same sequence to be executed in different contexts or chained together for complex animations. The software structure allows new profiles to be added through firmware updates, and future revisions may support runtime-loaded sequences from external memory or the HMI.

D. Communication and Synchronization

Synchronization across components is achieved through the Modbus TCP protocol, with the PLC acting as the supervisory controller. The MCU continuously polls designated holding registers to receive commands and updates, while periodically writing back telemetry data. Each read/write cycle is aligned with the PLC scan rate to prevent data collision and ensure real-time consistency.

The software includes a heartbeat mechanism to monitor connectivity between the MCU and PLC. If the heartbeat signal is interrupted beyond a configured timeout threshold, the MCU halts all motion routines and enters a fault state, requiring manual recovery from the HMI. This mechanism ensures failsafe operation and prevents unintended motion in the event of communication loss.

E. Diagnostics and Maintenance Utilities

Built-in diagnostics support both real-time monitoring and post-event analysis. The MCU logs runtime status, servo error flags, voltage anomalies, and watchdog resets, which can be accessed through a debug UART or read indirectly via PLC tag mapping. These metrics enable targeted maintenance and assist in identifying hardware or configuration issues.

VI. System Testing and Validation

System validation followed a structured testing methodology focused on verifying operational integrity, fault tolerance, and interactive responsiveness across all subsystems. Testing was performed in incremental phases: unit-level, subsystem-level, and full system integration, allowing faults to be isolated and corrected prior to final validation.

A. Functional Testing

Functional testing verified that each system module performed per design specifications and inter-component communication was synchronized across the control hierarchy. Initial tests were conducted on individual PCBs using logic analyzers and multimeters to confirm correct voltage regulation, signal timing, and I/O mapping. Microcontroller firmware was validated through a series of test routines executed via UART terminal, including GPIO toggling, UART packet loopback, SPI frame integrity, and stepper/servo control confirmation.

Following component-level verification, integration testing began with communication pathways. Modbus TCP exchanges between the PLC, HMI, and MCU were validated by cycling through predefined register write/read sequences and confirming mirrored state changes on all nodes. Each motion profile was then tested in isolation: the PLC issued profile execution commands, the MCU initiated motion routines, and resultant position data was logged and compared against expected trajectories. Real-time feedback from the DYNAMIXEL servos was monitored for accuracy, and stepper pulses were captured with an oscilloscope to confirm precise timing and acceleration ramping.

Tests were also conducted under varying loads and supply conditions to assess system robustness. Voltage rails were monitored under full motion load to detect ripple or brownout behavior. The LM2576HV regulator and A4988 driver were stress-tested at near-maximum rated current.

B. Fault Scenario Simulation

To evaluate system resilience, controlled fault scenarios were introduced at both hardware and software levels. These simulations targeted critical failure modes, including power interruption, communication timeout, invalid command injection, and actuator stall conditions.

Power loss tests were conducted by disconnecting individual voltage rails mid-operation. The system was expected to shut down gracefully and resume a known-safe state on reboot. Watchdog behavior on the MCU was verified by intentionally deadlocking the main execution loop, ensuring a system reset and fallback to initialization routines.

Communication loss was simulated by severing Ethernet links and injecting malformed Modbus frames into the network. The MCU properly identified timeout events, ceased motion activity, and raised a fault flag to the PLC.

Error responses were logged in the HMI and persisted until manually acknowledged.

Actuator-specific faults were tested by inducing mechanical stalls in both servo and stepper outputs. DYNAMIXEL servos reported overcurrent and position error flags, which were transmitted back to the MCU and displayed in the HMI fault log. For the stepper motor, fault detection relied on missed pulse compensation; stall conditions resulted in motion tracking errors, prompting the PLC to suspend the sequence and lock the system into a manual reset state.

All fault simulations were repeated under varying operating conditions and system loads to confirm consistent recovery behavior. Each detected fault was time-stamped and logged for traceability, ensuring that the fault-handling logic remained deterministic and met safety criteria across repeated trials.

C. Idle Mode and Interactivity

Testing of idle behavior and user interactivity focused on system responsiveness, safe default states, and rapid transition between modes. When not executing motion sequences, the system enters a low-activity state where all motion outputs enter a waiting state, but monitoring subsystems remain active. During idle mode, periodic polling maintains PLC and HMI connectivity, and the system continues to log environmental status such as voltage levels and uptime.

Interactive behavior was tested by issuing commands from the HMI in various sequences, including rapid motion switching, parameter adjustment mid-sequence, and emergency stop interrupts. The system correctly queued or rejected commands based on current state and safety rules, ensuring that only valid transitions occurred. Manual override functionality was also tested, allowing operators to reposition actuators directly and verify that feedback reporting remained accurate.

Idle timeout and wake-from-idle routines were verified by simulating user inactivity and observing whether the system safely holds its state and reset motion registers. Upon user input, the system resumed polling and re-established full readiness without requiring a hard reboot.

D. Verification Metrics and Compliance

System verification adhered to a pass/fail criteria matrix based on functionality, responsiveness, and safety. Each major feature: motion profile execution, HMI interaction, sensor feedback, and fault handling, was tested under both nominal and off-nominal conditions. Test cases were executed multiple times to assess repeatability and failure recovery.

Quantitative metrics such as servo accuracy ($\pm 0.5°$), stepper timing resolution (1 µs), network packet latency (≤ 10 ms), and voltage ripple (<100 mV under load) were all confirmed to remain within acceptable operational thresholds. Functional requirements were mapped to test outcomes in a traceability table to confirm full design coverage.

The system passed all critical validation points and demonstrated reliable, deterministic behavior during extended runtime tests. Results indicate the architecture is suitable for continuous interactive operation and capable of gracefully handling fault events, confirming its readiness for real-world deployment.

VII. Conclusion

This project successfully demonstrates the feasibility and effectiveness of using industrial automation technologies to drive a modular, scalable animatronic system. While keeping a tight budget in mind, we have created a system that maintains a high level of confidence and reliability, whilst cutting costs in unique ways through resourcefulness and streamlined design. Utilizing tightly integrated architecture combining a PLC-based logic core, Ignition Edge HMI, and embedded microcontroller coordination, the system achieves real-time motion control, robust fault handling, and user-friendly interactivity. Custom-designed PCBs support reliable signal routing and efficient power distribution, with voltage regulation engineered to handle dynamic actuation loads. Firmware development on the SAMD21 enables deterministic communication, closed-loop servo control, and configurable motion profiling, while the HMI facilitates live diagnostics and control in an accessible web-based interface. Extensive testing, including functional verification, fault scenario simulation, and idle mode interactivity, confirmed the system's precision, safety, and resilience. Together, these elements validate the animatronic platform as a reliable, extensible solution for creative and educational automation applications, bridging the gap between industrial standards and interactive robotic performance.

Acknowledgement

The authors wish to acknowledge Dr. Lei Wei, Dr. Mike Borowczak, Dr. Saleem Sahawneh, Professor Mark Maddox, and sponsor Richard Kim from ROBOTIS for their guidance and support.

REFERENCES

[1] Microchip Technology Inc. SAM D21E / SAM D21G / SAM D21J Datasheet. 2017. www.microchip.com/content/dam/mchp/documents/OTH/ProductDocuments/DataSheets/40001882A.pdf
[2] WIZnet Co., Ltd. W5500 Datasheet. Version 1.0.2, November 2013. https://cdn.sparkfun.com/datasheets/Dev/Arduino/Shields/W5500_datasheet_v1.0.2_1.pdf
[3] Rockwell Automation. Micro800 Programmable Controllers Technical Data. Publication 2080-TD001D-EN-P, 2023. https://literature.rockwellautomation.com/idc/groups/literature/documents/td/2080-td001_-en-p.pdf
[4] Texas Instruments. LM2576 SIMPLE SWITCHER® 3A Step-Down Voltage Regulator Datasheet. SNVS107C, Revised April 2013. https://www.ti.com/lit/ds/symlink/lm2576.pdf
[5] ROBOTIS. DYNAMIXEL XC330-T288-T Manual. https://emanual.robotis.com/docs/en/dxl/x/xc330-t288/
[6] ROBOTIS. DYNAMIXEL 2XL430-W250-T Manual. https://emanual.robotis.com/docs/en/dxl/x/2xl430-w250/
[7] ROBOTIS. DYNAMIXEL XL430-W250-T Manual. https://emanual.robotis.com/docs/en/dxl/x/xl430-w250/
[8] EDATEC. ED-HMI3010-101C Datasheet. Mouser Electronics, 2024. https://www.mouser.com/datasheet/2/1563/HMI3010_10datasheet_en-3446798.pdf
[9] MakerBot. T-Rex Skeleton. Thingiverse Thing:275091, 2013. https://www.thingiverse.com/thing:275091