Standardized PLC to Robot Integration Using SRCI over EtherNet/IP via CIP Technologies

Tommaso Fochi
Principal Engineer
Rockwell Automation

Paul Brooks
Senior Manager, Open Architecture Management
Rockwell Automation

Joakim Wiberg
Director of Technology
ODVA, Inc.

Todd Wiese
Principal Software Systems Architect
Rockwell Automation

Presented at the ODVA
2026 Industry Conference & 24th Annual Meeting
October 21, 2026
Vilamoura, Portugal

Abstract

As industrial robotics adoption accelerates, the need for standardized, vendor neutral integration between Programmable Logic Controllers (PLCs) and robot controllers has become increasingly critical. While the Standard Robot Controller Interface (SRCI) specification [2] defines a comprehensive, PLC centric command and state model for robotic systems, it does not prescribe a Common Industrial Protocol (CIP) based transport suitable for EtherNet/IP environments. This paper proposes a new CIP profile that enables EtherNet/IP as a standardized transport for interacting with SRCI compliant robot and robot control equipment.

The paper examines the architectural and protocol considerations required to map SRCI capabilities onto CIP technologies, with emphasis on the SRCI Command formulation through EtherNet/IP connections and other applicable CIP technology capabilities. Key topics include transport mapping strategies, definition of communication and command state machines, fragmentation and reassembly of SRCI messages, priority handling for command sequencing, and execution behavior aligned with CIP requirements. The approach addresses communication patterns while preserving SRCI semantics for command execution, status reporting, and synchronization.

 ©2026 ODVA, Inc.

In addition, the paper provides categories of SRCI command support (e.g. core, extended, and optional functions) and explores how these categories can be represented via the CIP protocol for scalable adoption across ODVA compliant devices. PLC and robot controller interfacing models are discussed to illustrate how SRCI based command execution can be integrated into existing EtherNet/IP control architectures.

The expected outcome is a reusable, standards aligned framework that enables interoperable PLC to robot communication using CIP and EtherNet/IP. By providing CIP transports to support the SRCI standard and network data transmission the aim is to reduce integration complexity, promote multi-vendor compatibility, and provide a foundation for future enhancements in industrial robotics control within the ODVA ecosystem.

Keywords

Standard Robot Controller Interface (SRCI), Common Industrial Protocol (CIP), EtherNet/IP, PLC-to-Robot Communication, Industrial Robotics Integration, Transport Mapping, Robot Command State Machine, Message Fragmentation and Reassembly, Command Sequencing and Prioritization, Multi-Vendor Interoperability, Deterministic Industrial Control.

Definition of terms

ACR: Active Command Register
CIP: Common Industrial Protocol
CMD: Command
CML: Command Management Layer
PLC: Programmable Logic Controller
SRCI: Standard Robot Controller Interface

Introduction

Industrial robotics is playing an increasingly important role in modern manufacturing, where demands for flexibility and rapid deployment are driving broader adoption across both discrete and process industries. As robotic systems become more pervasive, system integrators and end users are challenged by the diversity of proprietary robot programming environments, communication interfaces, and controller integration methods. These differences often require vendor-specific engineering expertise, increase application development effort, and hinder the portability of automation solutions across robot platforms. The need for standardized, vendor-neutral mechanisms for communication between Programmable Logic Controllers (PLCs) and industrial robot controllers has therefore become a key requirement for scalable and interoperable automation architectures. The SRCI initiative was created specifically to address these challenges by providing a common PLC-centric command and state abstraction that allows robots from multiple vendors to be controlled through a consistent interface.

The Standard Robot Controller Interface (SRCI) specification defines a model for robot command execution, state management, capability access, and operational control. By standardizing command semantics and execution behavior, SRCI significantly reduces the application complexity traditionally associated with multi-vendor robotic deployments. While SRCI defines how a PLC and robot controller shall interact at the application level, it intentionally does not prescribe a communication transport aligned with widely deployed industrial Ethernet technologies. As a result, organizations adopting EtherNet/IP-based architectures must currently rely on proprietary implementations, custom gateways, or vendor-specific adaptation layers to bridge the gap between SRCI functionality and Common Industrial Protocol (CIP) communications.

 ©2026 ODVA, Inc.

This paper proposes a CIP-based transport profile that enables standardized and interoperable communication between PLCs and SRCI-compliant robotic systems using EtherNet/IP. The objective is not to alter or redefine SRCI itself, but rather to extend its applicability within ODVA-conformant architectures by defining how SRCI command semantics, execution models, state management constructs, and capabilities can be realized through existing CIP services and EtherNet/IP communication patterns. By leveraging established CIP technologies and communication mechanisms, the proposed approach provides a framework for integrating robotic systems into existing industrial control environments while preserving the behavior defined by the SRCI specification.

A fundamental design goal of this work is to achieve a non-intrusive integration model that imposes no required changes to the SRCI specification and minimizes implementation impact on robot controller vendors. Rather than introducing a new set of robot behaviors, the proposed transport profile maps existing SRCI concepts onto established CIP communication services in a manner that preserves SRCI execution semantics, sequencing behavior, and operational assumptions. This approach allows robot manufacturers to leverage existing SRCI implementations while providing the EtherNet/IP communications framework that promotes interoperability, reduces development and integration effort, and establishes a foundation for future multi-vendor robotic solutions within the ODVA ecosystem.

Background and Motivation

The Standard Robot Controller Interface (SRCI) was developed to address one of the longstanding challenges in industrial robotics: the lack of a common and vendor-neutral method for PLC-based robot control. By defining standardized command semantics, execution behavior, state management, and capability access, SRCI enables system developers to interact with robots through a consistent programming model independent of the underlying robot manufacturer. This abstraction reduces engineering effort, improves portability, and supports the industry objective of simplifying multi-vendor robotic integration. However, while SRCI successfully standardizes the application-layer behavior between PLCs and robot controllers, it intentionally remains transport-independent and does not define a communication mechanism aligned with the Common Industrial Protocol (CIP) ecosystem.

This transport independence creates a practical gap for EtherNet/IP-based automation systems, where PLCs and other industrial devices communicate using CIP services and object models. Existing SRCI implementations typically rely on communication technologies outside the CIP domain, requiring robot vendors, PLC suppliers, and system integrators to develop gateways, protocol conversion layers, or vendor-specific adaptations when deploying robotic systems. These approaches increase engineering complexity, introduce additional validation layers and maintenance requirements, and can limit the degree of interoperability that SRCI can provide. As robotic deployments continue to expand, the need for standardizing transport aligned with the dominant industrial Ethernet infrastructure becomes further important. Figure 1 illustrates the existing SRCI communication architecture and the transport gap addressed by the proposed EtherNet/IP mapping.

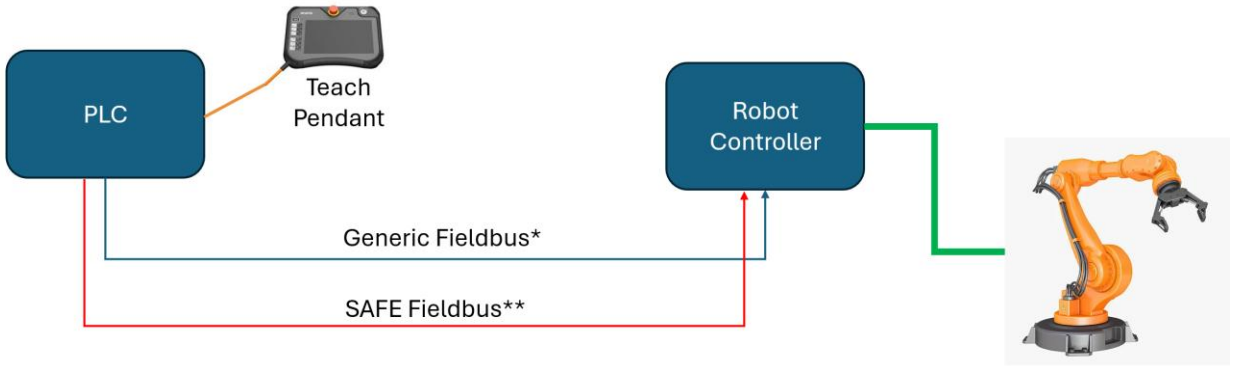


Figure 1. Existing SRCI communication architecture using generic and safety fieldbuses.

CIP provides a foundation for addressing this challenge through its mature, object-oriented communication architecture and large acceptance within industrial automation systems. The protocol offers well-defined mechanisms for device modeling, service invocation, Class 1 I/O connections and Class 3 explicit messaging communication, connection management, diagnostics, and capability discovery. Through these supported capabilities, CIP already supports the communication paradigms necessary to transport SRCI command execution, status reporting, configuration, and state management information. Leveraging these attributes creates an opportunity to realize SRCI semantics within a communication framework that is already familiar to EtherNet/IP device suppliers, system integrators, and end users.

The motivation for this solution is therefore to eliminate integration barriers and enable PLC-to-robot communication using standardized CIP services while preserving the behavioral consistency and execution semantics defined by SRCI. Rather than introducing new robot concepts, the proposed approach seeks to map established SRCI constructs onto existing CIP communication mechanisms in a manner that is transparent to applications and minimally disruptive to robot controller implementations. The result is an architecture that promotes interoperability, reduces development and deployment costs, leverages existing EtherNet/IP technology (Safety and Security) that extends the benefits of SRCI into ODVA-conformant automation systems without requiring changes to the SRCI specification itself.

Architectural Overview

Within the proposed architectural model, the PLC remains the initiator of robot operations, invoking SRCI-defined commands and monitoring execution status, while the robot controller is represented as a CIP device exposing standardized objects, assemblies, attributes, and services that encapsulate SRCI functionality. This approach allows SRCI applications to be integrated into EtherNet/IP architectures using established CIP communication patterns while preserving operational behavior defined in the SRCI specification. Figure 2 shows the application and command-management portions of the proposed layered architecture.

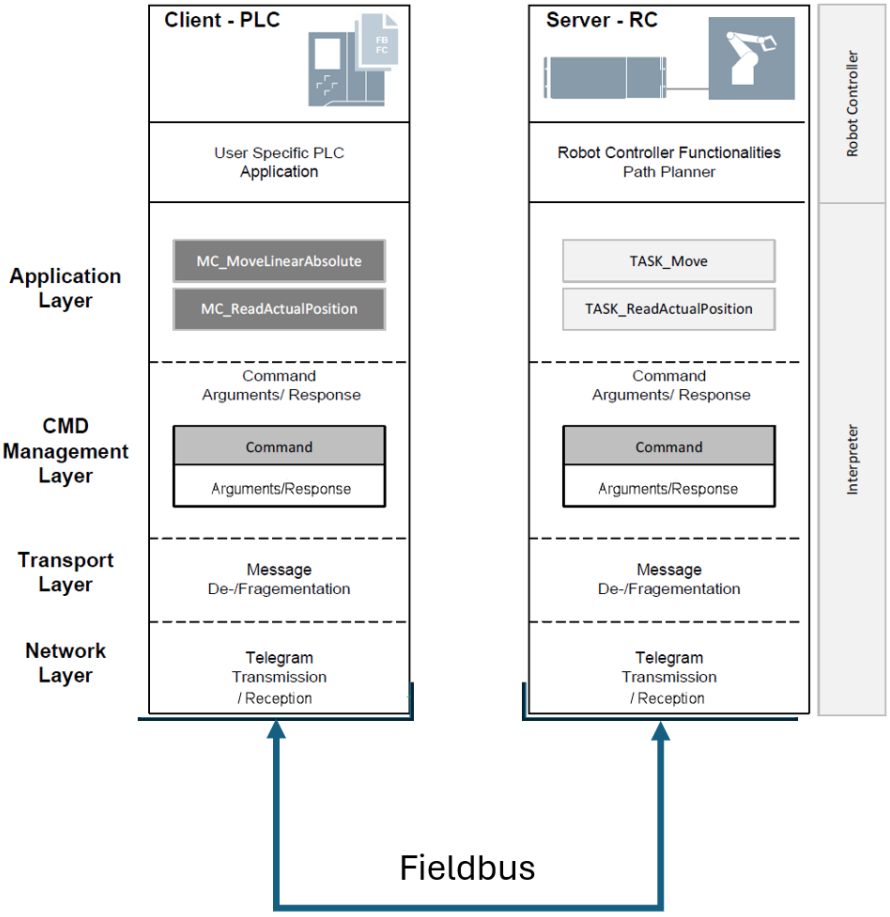


Figure 2. SRCI application and command management layers.

The architecture leverages both CIP Class 1 (implicit) and Class 3 (explicit) communications to support SRCI interactions. Class 1 cyclic connections provide deterministic exchange of command state, execution feedback, robot status, and other time-sensitive data required during active robot operation. Class 3 messaging may serve complementary functions such as capability discovery, device configuration, diagnostics, and retrieval of implementation-specific information. Together, both communication mechanisms create a framework capable of supporting SRCI command execution, state management, sequencing, and feedback reporting while aligning the communication models already deployed throughout EtherNet/IP networks. This architecture enables direct PLC-to-robot communication without custom gateways or protocol translation layers, providing multi-vendor integration and reducing deployment complexity.

Architectural Principles:

- Preservation of SRCI Semantics:
  The architecture maintains the command semantics, execution behavior, state transitions, and operational assumptions defined by the SRCI specification.

- Leverage Existing CIP Technologies:
  The solution utilizes established CIP technologies for safety and security, connection management, and messaging mechanisms rather than introducing new protocol technologies or vendor specific extensions.

- Deterministic Communication Model:
  Critical command execution and feedback are transported through CIP Class 1 cyclic communications to provide predictable and observable behavior consistent with industrial control system requirements.

- Separation of Operation and Configuration Functions:
  Real-time execution activities are isolated for configuration, diagnostics, and capability discovery functions through the appropriate use of Class 1 and Class 3 communication mechanisms.

- Vendor-Neutral Interoperability:
  Standardized models and communication behaviors enable consistent PLC-to-robot interactions across SRCI-compliant robot platforms from multiple suppliers.

- Minimal Vendor Implementation Impact:
  The architecture shall be designed to map existing SRCI functionality onto CIP technologies with minimal intended changes to robot controller implementations and without causing alteration to existing SRCI definitions.

- Alignment with ODVA-Conformant Architectures:
  The proposed profile integrates naturally within existing EtherNet/IP systems, leveraging deployed infrastructure, engineering tools, and conformance models already familiar to industrial automation users.

- Scalable Capability Representation and Discovery:
  Capability exposure, command categorization, and feature discovery can be structured to support a wide range of robot implementations, from the core, extended and optional SRCI functionality already provided.

 ©2026 ODVA, Inc.

Figure 3 summarizes how the application, command-management, Active Command Register, and network layers interact.

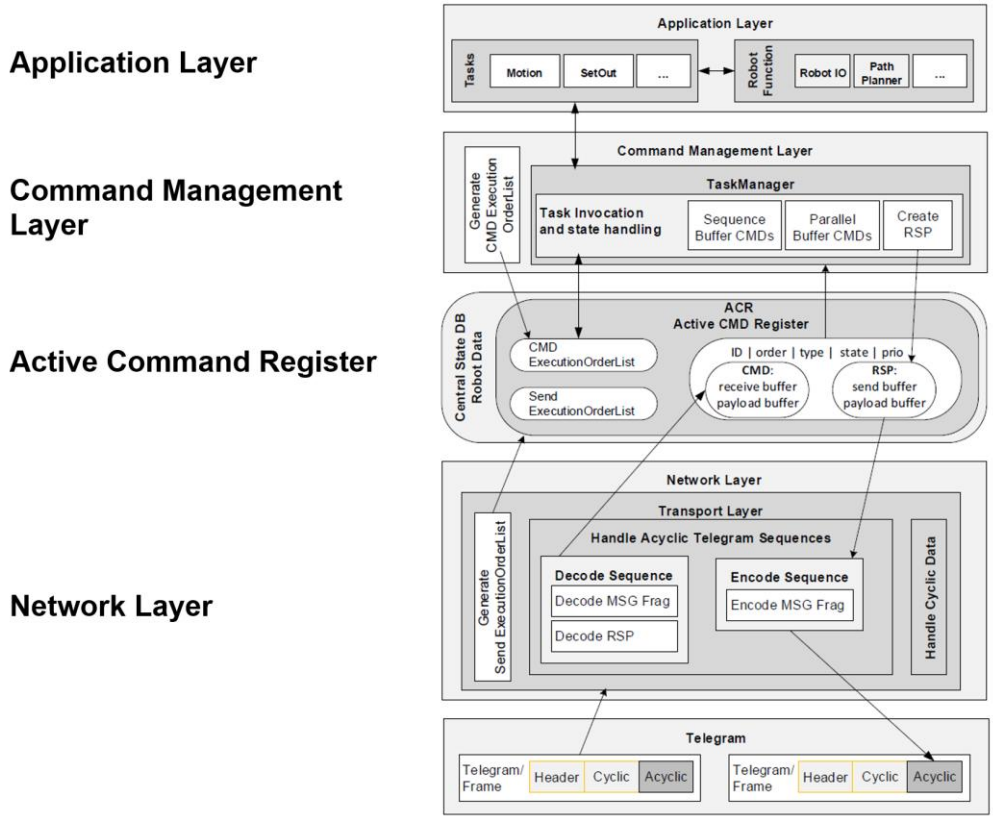


Figure 3. SRCI layered command and transport architecture.

Transport Mapping Strategy

A fundamental design principle of the mapping strategy is the separation of real-time operational exchanges from configuration and management functions. Real time control - communication associated with command execution, robot status monitoring, feedback reporting, acknowledgements, and state transitions is transported using CIP Class 1 cyclic I/O connections. This approach leverages the deterministic behavior of EtherNet/IP implicit messaging to ensure that SRCI execution state changes and operational feedback are delivered with predictable control timing characteristics. Functions that do not require cyclic updates, such as capability discovery, device configuration, diagnostics, profile identification, and implementation-specific information retrieval, may be accessed through CIP Class 3 explicit messaging services. By partitioning communication responsibilities in this manner, the architecture aligns with established CIP communication practices while supporting the SRCI interaction model.

Message Structure

The transport profile defines standardized mappings for SRCI communication elements into CIP data structures:

- SRCI Command Invocation
  - Represented within cyclic command assemblies.

    - Includes command identifier, execution trigger, command parameters, and associated execution control information.
    - Maintains one-to-one correspondence with SRCI command definitions.

- SRCI Command Responses
    - Represented through cyclic status assemblies.
    - Provides acknowledgement status, execution progress, completion state, and error indications.

- Robot Feedback Data
    - Conveyed through dedicated feedback structures.
    - Intended to include robot state information, positional data, operating mode, diagnostic status, and command-specific result information.

- Capability Information
    - Exposed through CIP Application Profiles and services.
    - Enables discovery of supported SRCI profiles, command categories, optional capabilities, and implementation limits.

Transmission Sequencing and Execution Integrity:

To preserve SRCI behavioral consistency, each command transaction shall be associated with a unique execution identifier that remains valid throughout the command lifecycle. Execution identifiers enable deterministic correlation between command requests, acknowledgements, execution state updates, completion notifications, and fault responses. The transport profile defines sequencing rules that ensure commands are processed in the order intended by SRCI execution models, including support for buffered commands, sequential execution, command interruption, and prioritization.

The cyclic communication model serves as the primary mechanism for maintaining operational synchronization between the PLC and robot controller. State transitions are communicated through continuously updated command and status assemblies, providing a deterministic view of execution progress without requiring repeated explicit service requests. This model allows SRCI command handshakes, acknowledgements, and completion semantics to be preserved while aligning with established EtherNet/IP communication patterns.

Data Representation and Formatting Rules:

To ensure interoperability, the transport mapping defines common formatting requirements for all SRCI-related CIP data:

- Standardized CIP data types shall be used for all command parameters and status information.
- Data alignment rules shall ensure consistent interpretation across controller platforms.
- Enumerated SRCI states and command results shall be represented through standardized CIP-compatible values.
- Bit-level status indicators shall be used where appropriate for execution state, acknowledgements, fault indicators, and control flags.
- Parameter structures shall preserve SRCI-defined ordering and semantic meaning.
- Synchronization and timing-related information shall be represented using CIP-compatible time and sequence constructs.

Class 1 and Class 3 Usage Model:

Table 1 identifies the preferred CIP transport for the operational, configuration, discovery, diagnostic, and maintenance functions discussed above.

Table 1. Class 1 and Class 3 usage model.

Function	Preferred Transport
Command Execution	Class 1
Command Acknowledgement	Class 1
Execution State Monitoring	Class 1
Robot Status Reporting	Class 1
Motion Feedback	Class 1
Command Completion	Class 1
Capability Discovery	Class 3
Device Configuration	Class 3
Diagnostics	Class 3
Profile Identification	Class 3
Maintenance Functions	Class 3

Class 1 data mapping format:

Client to RC Header:

Table 2 defines the proposed ordered fields in the cyclic header sent from the PLC client to the robot controller.

Table 2. Client-to-robot-controller cyclic header format.

Byte	Data Type	Name		Description	
0	BYTE	SRCIVersion		Version of SRCI Specification	
1	BYTE	FastStop	LifeSign*	Fast Stop Trigger	Connection Alive Signal
2 3	UINT	TelegramLength ClientServer		Number of Bytes of the frame to be used for the telegram of the given axis group. Direction client to server.	
4 5	UINT	TelegramLength ServerClient		Number of Bytes of the frame to be used for the telegram of the given axis group. Direction server to client.	
6	BYTE	AxesGroupID	Control	Axes Group ID	Telegram State Control
7	BYTE	Reserved		Reserved - Future	
8 9	UINT	TelegramNumber ClientServer		Configuration of the optional Cyclic Data. Direction client to server	
10 11	UINT	TelegramNumber ServerClient		Configuration of the optional Cyclic Data. Direction server to client	
12 13	IEC DATE	ClientDate		Date of the client in the format days since 1990.01.01	
14 15 16 17	IEC TIME	ClientTime		Time in the clients time zone in the format of milliseconds since start of day.	

LifeSign (*): is provided as an integrity check. The header and footer are compared to discover discrepancies in the messaging transaction. If a discrepancy is detected the processing of the received frame is skipped and an inconsistent frame counter is incremented as a part of the packet processing.

Client to RC Footer:

Table 3 defines the corresponding footer sent from the PLC client to the robot controller.

Table 3. Client-to-robot-controller cyclic footer format.

Byte	Data Type	Name		Description	
n	BYTE	Reserved	LifeSign*	Reserved	Connection Alive Signal

RC to Client Header:

Table 4 defines the proposed ordered fields in the cyclic header returned from the robot controller to the PLC client.

Table 4. Robot-controller-to-client cyclic header format.

Byte	Data Type	Name		Description	
0	BYTE	SRCIVersion		Version of SRCI specification	
1	BYTE	LifeSign*	Reserved	Connection alive signal	Reserved
2	BYTE	Reserved		Reserved	
3	USINT	TelegramState		Initialization and Telegram control state	
4	DWORD	State	Status DWord	Combination of various related states.	
5					
6					
7					
8	UINT		Override	Actual override in percentage encoding	
9					

RC to Client Footer:

Table 5 defines the corresponding footer returned from the robot controller to the PLC client.

Table 5. Robot-controller-to-client cyclic footer format.

Byte	Data Type	Name		Description	
n	BYTE	LifeSign*	Reserved	Connection Alive Signal	Reserved

The SRCI specification defines a minimum telegram size of 256 bytes for cyclic data exchange, although the exact utilization of this payload is application dependent and not strictly enforced by the profile. In existing transport implementations, a telegram may contain data associated with a single robot controller or aggregate data for multiple robot controllers within the available payload space. For EtherNet/IP Class 1 transport, a corresponding assembly size of 256 bytes is proposed to provide alignment with SRCI expectations and promote implementation consistency. The specific content contained within the assembly may vary depending on the active SRCI command and associated parameter set; however, the overall payload organization is intended to remain deterministic. Common regions of the assembly include protocol header information, variable cyclic data, sequence management information, command or status payload data, and footer elements used to support validation and processing. This structured approach provides a predictable transport format while allowing the payload content to adapt to the requirements of individual SRCI services and robot applications.

The Class 1 connection data mapping format defines the ordered layout of the cyclic data exchanged between the PLC client and the robot controller over EtherNet/IP. The Client-to-RC and RC-to-Client mapping tables establish the logical sequence, data types, and sizes of the SRCI elements that shall be included in each connection direction, while the PLC-to-Robot Command Assembly and Robot-to-PLC Status Assembly examples shown in the following example, illustrates the realization of SRCI commands within EtherNet/IP I/O assemblies. Together, these definitions create a deterministic payload structure in which SRCI command, parameter, status, and feedback information is packed sequentially into the Class 1 connection data buffer according to specified offsets and lengths. Both transmitting and receiving endpoints are expected to interpret the payload using a defined assembly definition, ensuring consistent decoding of individual SRCI elements and preserving the ordering, structure, and semantic meaning of the underlying SRCI data throughout cyclic communication.**Example**: SRCI MC_MoveLinearAbsolute Command Mapping

> The following example illustrates how SRCI motion commands may be transported using a CIP Class 1 cyclic connection. The PLC initiates the command by populating a command assembly with the required SRCI command identifier, execution control information, and motion parameters. The robot controller continuously consumes the cyclic data, detects the command request, validates the parameters, and updates corresponding status assemblies to reflect command acknowledgement, execution progress, and completion state.

PLC-to-Robot Command Assembly:

Table 6 provides an example PLC-to-robot command assembly for the MC_MoveLinearAbsolute command.

Table 6. Example PLC-to-robot command assembly for MC_MoveLinearAbsolute.

Field	Description	Example Value
Command Sequence ID	Unique execution identifier	1001
Command ID	SRCI Command = MC_MoveLinearAbsolute	0x0021
Execute Trigger	Rising edge initiates command	TRUE
Target X	Cartesian target position	1250.0 mm
Target Y	Cartesian target position	450.0 mm
Target Z	Cartesian target position	800.0 mm
Orientation A	Tool orientation	180°
Orientation B	Tool orientation	0°
Orientation C	Tool orientation	90°
Velocity	Motion velocity	1000 mm/s
Acceleration	Motion acceleration	500 mm/s²

Robot-to-PLC Status Assembly:

Table 7 provides the corresponding example robot-to-PLC status assembly.

Table 7. Example robot-to-PLC status assembly for MC_MoveLinearAbsolute.

Field	Description	Example Value
Command Sequence ID	Echoes active command	1001
Command Accepted	Command validation successful	TRUE
Command Active	Motion currently executing	TRUE
Command Complete	Motion finished	FALSE
Error Present	Fault condition	FALSE
Current State	SRCI execution state	EXECUTING
Progress Indicator	Optional execution progress	65%

Example Command Flow:

Figure 4 illustrates the cyclic command and status exchange for the MC_MoveLinearAbsolute example.

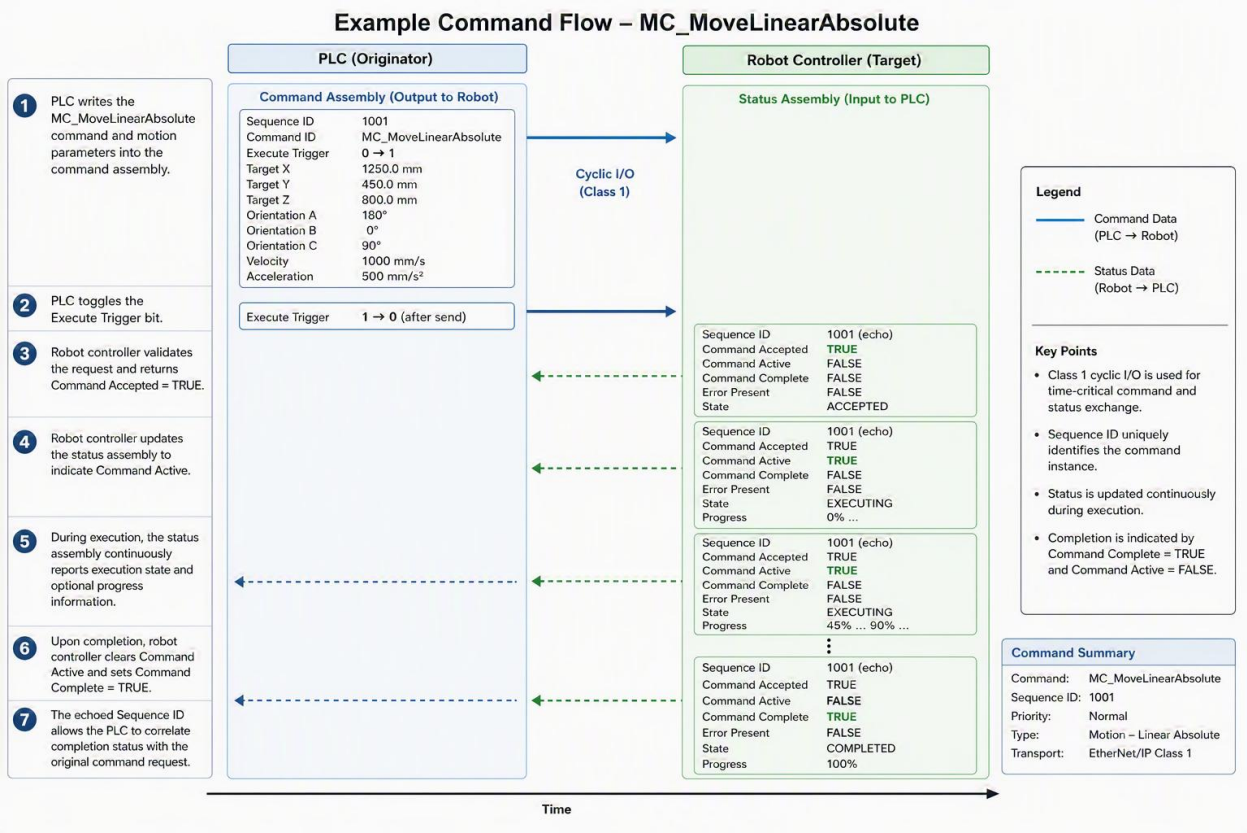


Figure 4. Example command flow for MC_MoveLinearAbsolute.

Communication and Command State Machines

The interaction model between PLC and robot controller through formalized communication and execution state machines proposed is defined in the following section. The SRCI command lifecycle, includes issuing, acknowledgment, execution, completion, and error handling, and how it is mapped into CIP-compliant state transitions.

The communication and command state machine architecture provides the framework through which SRCI commands are created, prioritized, transported, executed, and monitored throughout their lifecycle. Within the architecture, the Application Layer remains responsible for invoking SRCI function blocks, while the Command Management Layer serves as the intermediary responsible for converting these function invocations into transportable command instances (CMDs). Each CMD encapsulates the information required to execute a robot operation, including command-specific parameters, execution control information, and status tracking data. The Command Management Layer also receives responses and execution updates from the Transport Layer and routes them back to an originating function block, providing a clear separation between application logic and communication mechanisms.

To support reliable command management and deterministic execution behavior, every CMD is assigned a unique Command Sequence Identifier (Command Sequence ID) when it is created. The Command Sequence ID allows command instances to be tracked throughout their complete lifecycle and ensures that responses, acknowledgements, execution state changes, and completion notifications can be correlated with the originating request. This mechanism is particularly important in scenarios where multiple instances of the same SRCI command are active simultaneously or when commands are buffered for future execution.

When no SRCI command is pending, the command-processing mechanism remains in an “Empty” state which does not require any additional NO OP command. The EtherNet/IP Class 1 connection continues its cyclic exchange of communication control, robot status, and LifeSign information, thereby maintaining the connection independently of command activity. Upon invocation of an SRCI function block, the Command Management Layer creates a command instance and transitions it from “Empty” to “Created”. The command is then selected for transmission and progresses to “Sent” after its complete transfer to the robot controller. Following receipt and validation, the robot controller reports the command as “Buffered”, indicating that it has been accepted for processing. When execution begins, the command transitions to “Active”, and upon successful completion it transitions to “Done” before the associated command-processing resources are released and returned to the “Empty” state. The Command Sequence ID remains associated with the command throughout this lifecycle to correlate the request with its acknowledgements, execution-state updates, and completion status.

The architecture further defines a command prioritization model that governs transmission and execution ordering. Each command is assigned a priority level that determines its relative importance within the command processing system. Higher-priority commands are transmitted and processed before lower-priority commands and may interrupt the transmission or execution sequencing of pending lower-priority operations when permitted by SRCI execution rules. This capability enables real time critical functions, such as stop, abort, or fault recovery operations, to be processed with predictable responsiveness while maintaining deterministic command handling behavior.

In addition to identification and prioritization, each Command ID is assigned a command type that defines the structure and interpretation of its payload. The command type informs both the PLC client and the robot controller how the associated parameter data shall be decoded and processed. By combining command identification, priority assignment, and command typing within a standardized command management framework, the proposed architecture ensures consistent interpretation of SRCI commands while enabling predictable transport, execution tracking, and state synchronization across the EtherNet/IP communication channels.

Diagram of Command ID:

Figure 5 shows the command identifier structure used to associate command priority, type, and execution state with the command payload and response.

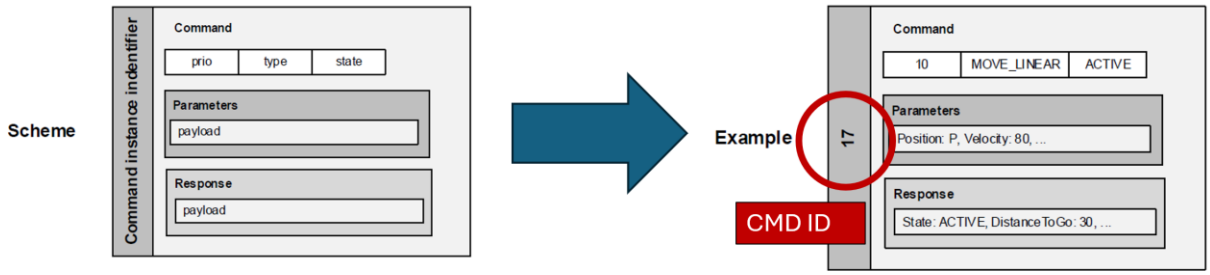


Figure 5 SRCI command identifier scheme and example.

The SRCI architecture does allow multiple commands to exist simultaneously in different states, including:

- Commands waiting in sequence buffers.
- Commands executing in parallel buffers.
- Continuous commands.
- Triggered commands.
- Buffered commands awaiting execution.
- Commands that have completed but whose final state has not yet been acknowledged.

An Active Command Register (ACR) identified in the SRCI specification is effectively designed to maintain the state of multiple active command instances concurrently

Although nothing is finalized for the transport mapping to EtherNet/IP, the possibility of supporting multiple SRCI commands may exist simultaneously within the ACR depending on implementation, including commands that are buffered, executing, interrupted, completed, or awaiting acknowledgment. To preserve SRCI execution semantics, the Robot Controller may report status information for multiple command instances within the cyclic EtherNet/IP status “stacked command data” assemblies. Where each status entry contains the associated Command Sequence ID, allowing a PLC to correlate acknowledgments, state transitions, execution progress, completion indications, and fault information with the originating command request. When the number of active command instances exceeds the capacity of a single cyclic transfer, status information may be distributed across successive communication cycles while maintaining deterministic command correlation and execution. This approach would enable concurrent command execution and reporting without requiring separate status assemblies for each command instance and maintain consistency with the SRCI Active Command Register model.

Fragmentation, Reassembly, and Data Handling

While the majority of SRCI command exchanges can be transported efficiently using standard cyclic EtherNet/IP communication, certain SRCI functions may produce payloads that exceed the practical size limits of a single CIP transfer. Examples include commands containing large parameter sets, extensive robot configuration information, complex motion definitions, diagnostic datasets, program-related data

exchanges, or future SRCI extensions. To ensure support for the full range of SRCI functionality without imposing artificial restrictions on message size, the proposed transport profile intends to define a standardized fragmentation and reassembly mechanism that allows large SRCI payloads to be transmitted across multiple CIP Class 1 communication cycles while preserving the original command semantics and execution behavior. An alternative option is to utilize opening a Class 1 connection with a Large Forward Open command to increase the payload capability for large SRCI payload transport requirements and leverage the existing EtherNet/IP solution space.

SRCI parameter structures are represented within a PLC and robot controller as ordered collections of typed data elements whose sequence, size, and semantic interpretation are defined by the SRCI profile. Prior to transmission, these structures shall be packed in a deterministic manner that preserves the SRCI-defined field alignment, ordering, and data representation requirements, ensuring that parameter sets can be reconstructed without ambiguity by the receiving endpoint. During EtherNet/IP transport, the packed structures will be serialized into a contiguous byte stream while preserving the SRCI-defined parameter ordering and data representation rules. No reordering or semantic transformation of individual parameters will be performed by the transport layer. Instead, the encoded byte stream is carried transparently within the EtherNet/IP connection payload. When a complete SRCI structure exceeds the capacity of a single transfer, the data is segmented into multiple fragments for transmission and subsequently reassembled by the receiving endpoint to reconstruct the original SRCI structure exactly as defined by the SRCI specification. This approach ensures that parameter relationships, ordering, structure packing, and semantic meaning remain consistent across heterogeneous PLC and robot controller implementations.

The fragmentation mechanism divides a large SRCI message into an ordered series of transport fragments, each containing protocol metadata necessary to support reconstruction at the receiving device. Fragment headers include a unique command or transaction identifier, fragment sequence number, total fragment count, payload length information, and integrity validation data. The receiving endpoint uses this information to reassemble the original SRCI message in the correct sequence before processing the command or associated data. Fragment ordering rules ensure that payload segments are reconstructed deterministically, regardless of transport timing variations, while maintaining full correlation with the originating SRCI command instance. The transport layer remains responsible for presenting a complete and validated SRCI message to the command processing layer, thereby isolating application logic from transport-specific fragmentation details.

To ensure reliable operation, the intended transport profile shall incorporate mechanisms for integrity verification, missing-fragment detection, timeout supervision, and recovery handling. Each fragmented transfer undergoes consistency validation prior to message acceptance, ensuring that incomplete, corrupted, duplicated, or out-of-sequence fragments cannot result in unintended command execution. When transmission errors or communication interruptions are detected, the transport layer may initiate retransmission or recovery procedures consistent with CIP communication behavior and implementation requirements. By providing standardized fragmentation, reassembly, and validation mechanisms, the proposed architecture enables the transport of large SRCI payloads while preserving command integrity, execution determinism, and interoperability across multi-vendor EtherNet/IP implementations.

Priority Handling and Execution Semantics

This section addresses how command prioritization and execution ordering are managed within the CIP-based transport. SRCI defines logical expectations for command sequencing and responsiveness, which must be preserved within the EtherNet/IP communication model.

The proposed architecture (shown in the diagram below) defines mechanisms for assigning command priority, and sequencing that determine transmission order, and preserve SRCI-defined execution behavior to ensure deterministic processing of concurrent robot operations.

Figure 6 summarizes the proposed encoding, integrity checking, fragmentation, reassembly, and error-handling workflow.

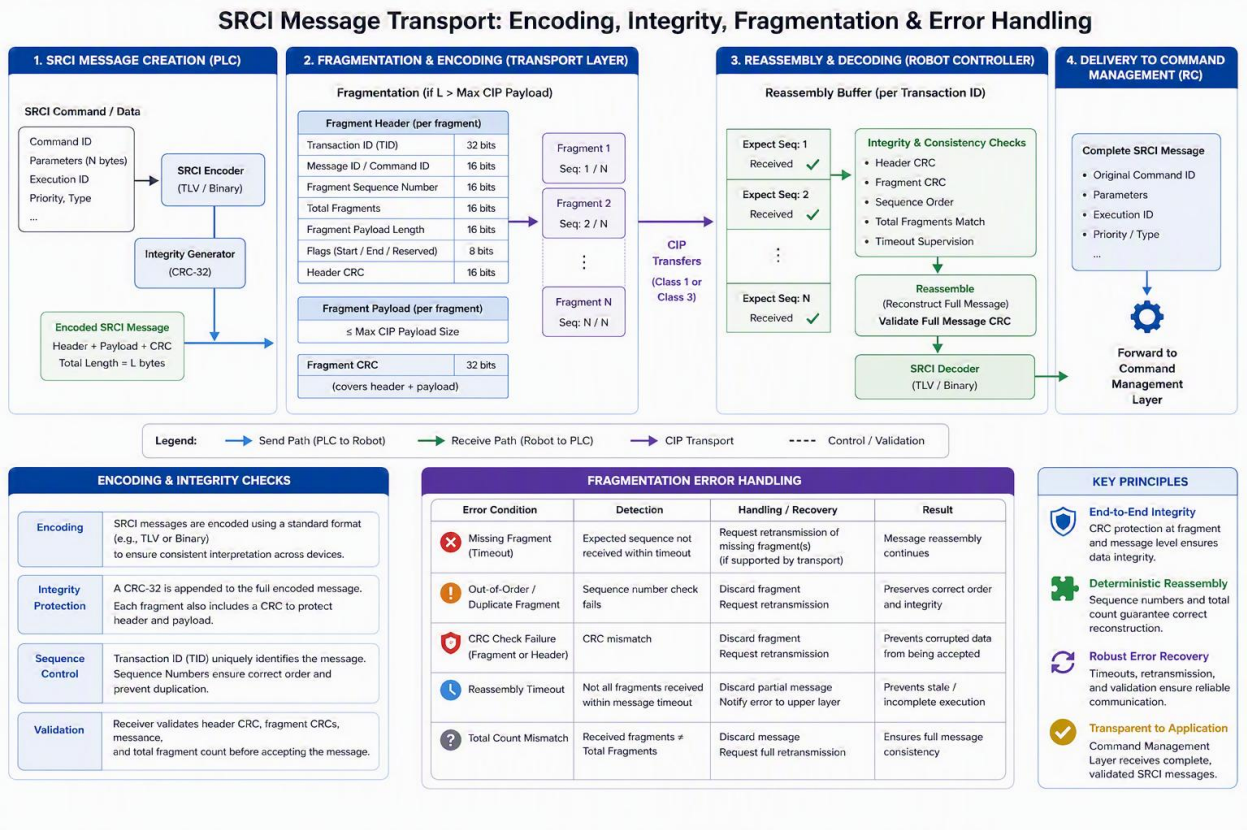


Figure 6. SRCI message transport encoding, integrity, fragmentation, and error handling.

SRCI Command Categorization and Capability Representation

The SRCI specification defines a broad set of robot control functions organized into capability categories that support a wide range of robotic applications and controller implementations. These categories typically include a mandatory Core Profile, which contains the fundamental command set required for basic robot operation, as well as Extended and Optional capabilities that provide additional functionality for advanced motion control, process integration, diagnostics, and specialized application requirements. By organizing functions into defined capability groups, SRCI enables robot vendors to implement a level of functionality appropriate for their products while maintaining a standardized programming model for PLC applications. This approach provides a scalable framework that accommodates both simple and highly capable robotic systems without requiring unique interfaces for each implementation.

 ©2026 ODVA, Inc.

Figure 7 groups representative SRCI commands into Core, Extended, and Optional capability categories.

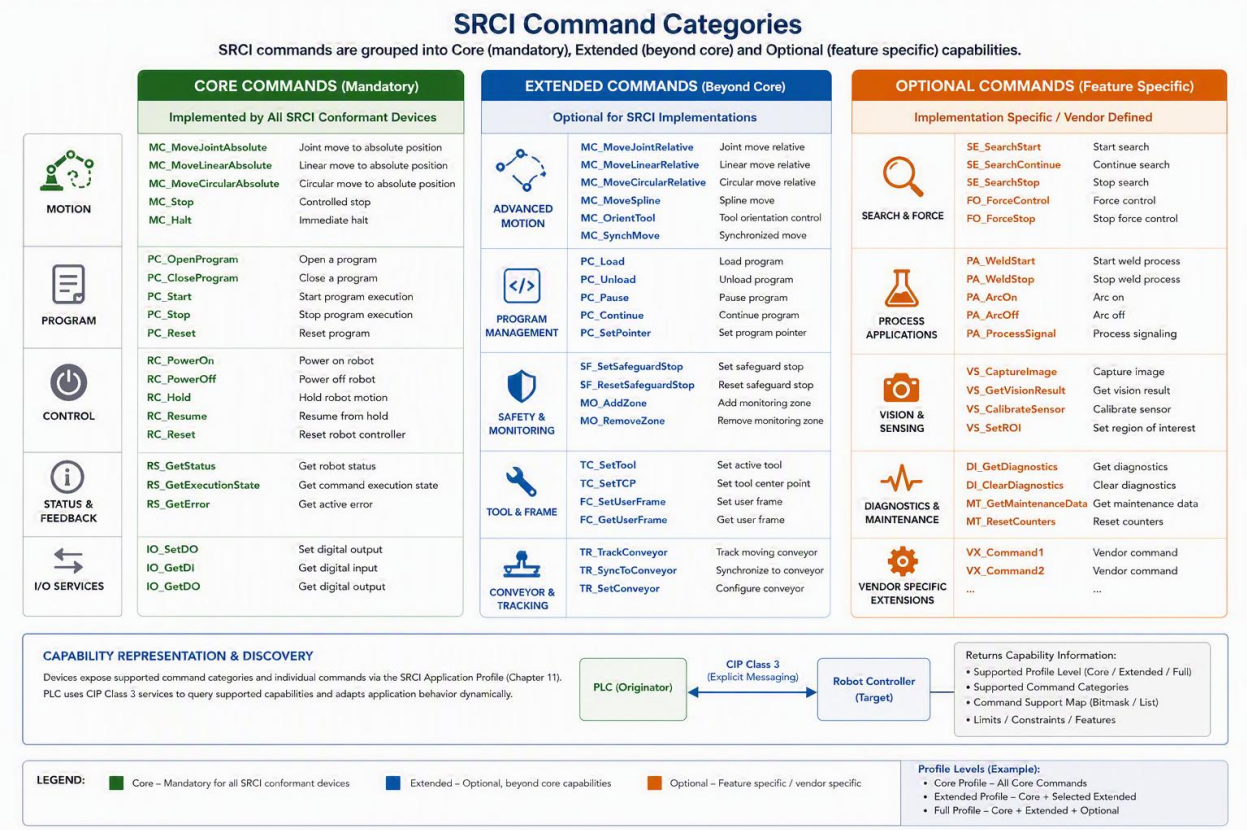


Figure 7. SRCI command categories and example capabilities.

To support this flexibility within an EtherNet/IP environment, it is suggested that the proposed architecture defines a CIP-based SRCI Application Profile that can provide a standardized method for representing supported command categories, functional capabilities, and implementation-specific limits. Rather than requiring PLC applications to rely on static device configuration or vendor-specific documentation. This allows a robot controller to advertise its conformance level, supported and implemented command groups using a common discovery mechanism. The resulting model enables a robot device to communicate not only what commands it supports, but also how those functions can be utilized within the context of the SRCI execution model.

A key element of the proposed design is the use of CIP explicit messaging services for capability discovery and profile interrogation. During system commissioning or connection establishment, a PLC could query the robot controller to retrieve information describing its supported SRCI profile information and command categories. This capability information can include support for Core functions, Extended functions, Optional functions, vendor-defined enhancements, payload limitations, and transport-related capabilities. Once discovered, the PLC application may dynamically adapt its behavior to the available feature set of the connected robot controller. This reduces engineering effort, minimizes configuration errors, and supports multi-vendor interoperability by allowing the same application logic to operate across robot platforms with differing functionality levels. As an alternative approach an EDS file may be enhanced to identify the supported SRCI command set in an offline tool.

The proposed categorization framework further supports future extensibility by allowing new SRCI command groups and profile definitions to be introduced without disrupting existing implementations. Because capabilities are exposed through standardized profile mechanisms rather than being hardcoded within applications, new functions may be adopted incrementally while preserving compatibility with

existing controllers and engineering tools. The result is a scalable ecosystem in which robot suppliers can differentiate products through supported capability levels while end users and system integrators benefit from automatic capability discovery, simplified device commissioning, and a consistent PLC programming experience across a full spectrum of SRCI-compliant robotic systems.

Example Application Profile Capability Structure

For illustration, an SRCI Application Profile may expose capability information using a hierarchy similar to the following:

Table 8 summarizes an illustrative capability structure that could be exposed by an SRCI Application Profile.

Table 8. Example SRCI Application Profile capability structure.

Capability Category	Example Representation
Profile Level	Core, Extended, Full
Supported Commands	Bitmask or Command List
Optional Functions	Individual Capability Flags
Motion Features	Linear, Circular, Coordinated Motion
Robot Services	Tool Control, Conveyor Functions, Search Functions
Transport Capabilities	Maximum Payload Size, Fragmentation Support
Controller Limits	Maximum Queued Commands, Maximum Concurrent Transactions
Profile Revision	Supported SRCI Version and Application Profile Revision

Using this model, a PLC or engineering tool may automatically determine whether commands such as the following examples: MC_MoveLinearAbsolute, MC_CircularRelative, MC_SearchHardStop, or other optional SRCI functions are supported before attempting execution, enabling robust and interoperable operation across diverse robotic platforms.

Although “Application Profile Revisions” are not currently defined in CIP Application Profiles, it may be possible to support incremental enhancements to the SRCI specification while maintaining backward compatibility. A proposed solution for the SRCI Application Profile may be to reserve a portion of the 32-bit profile descriptor for an Application Profile Revision field. This is similar to the approach used by the EtherNet/IP Conformance Level Profile. In this case the profile identifier would continue to represent the SRCI capability category, while the allocated revision field bits could indicate the specific version of that profile definition. This mechanism would allow the introduction of new commands, capabilities, or profile attributes through successive profile revisions without requiring the creation of a new Application Profile identifier. Engineering tools and controllers could use the revision information during capability discovery to determine the exact set of functionality supported by a device and adapt behavior accordingly.

Alternatively, since revisions of CIP Application Profiles are not formally defined, an additional option to support identifying SRCI version could be through the declaration of a SRCI device type. The device type in addition to describing the information model of an SRCI conformant robot controller interface could also provide an array of Supported Device Type Revisions – as defined in the “Conditional” implementation attribute 38 in the Identity Object [1]. In this case the SRCI specification revision(s) that the device interface conforms to could be specified.

Figure 8 illustrates a proposed SRCI Application Profile capability-discovery sequence using CIP explicit messaging.

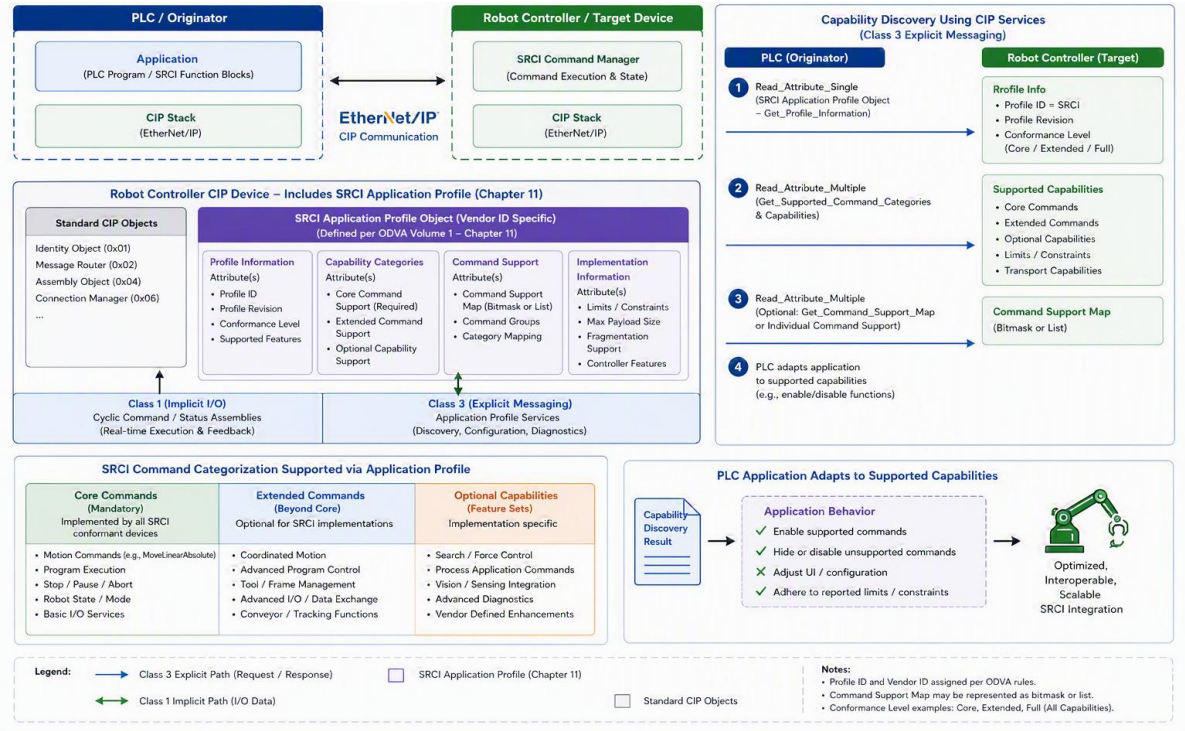


Figure 8. SRCI Application Profile capability discovery using CIP services.

PLC and Robot Controller Integration Models

The proposed SRCI-over-EtherNet/IP architecture is designed to integrate into existing industrial automation systems by leveraging standard CIP communication mechanisms already used by PLCs, drives, I/O devices, and other networked equipment. In a deployment, the PLC serves as the SRCI client and command originator, issuing robot commands through standardized SRCI function blocks while the robot controller acts as an EtherNet/IP target device exposing SRCI services and status information. Command execution is performed through cyclic Class 1 connections that transport command requests, acknowledgements, execution state information, and completion status in a deterministic manner. This model enables PLC applications to execute robot operations using the same engineering workflows and communication paradigms already employed for other EtherNet/IP devices, minimizing integration effort while preserving SRCI command semantics.

A common implementation scenario involves a PLC coordinating robot operations as part of a larger machine or production process. The PLC issues SRCI commands such as motion execution, tool control, or program management while continuously monitoring execution feedback and robot status through cyclic I/O exchanges. Robot activities may be synchronized with other CIP-connected devices, including motion axes, vision systems, conveyors, safety controllers, and distributed I/O, allowing robot operations to be incorporated into broader machine-level control sequences. Because the communication model leverages standard EtherNet/IP infrastructure, SRCI-enabled robots can participate in coordinated control strategies without requiring protocol gateways, custom interfaces, or vendor-specific integration

mechanisms. This supports a unified control architecture where robotic and non-robotic devices coexist within the same network and supervisory control environment.

The architecture's intent is to allow scaling effectively to more complex deployments involving multiple robot controllers and heterogeneous device populations. In a multi-robot workcell, a PLC may manage independent SRCI command channels to several robots while using capability discovery services to identify supported command sets and operational constraints for each device. The same EtherNet/IP infrastructure can support coordination between robots, machine subsystems, and higher-level production control functions while maintaining a consistent programming model. This flexibility allows implementations to range from a single robot integrated into a standalone machine to large-scale manufacturing systems incorporating numerous robots and other CIP-based devices. By preserving established EtherNet/IP communication patterns and enabling dynamic adaptation to device capabilities, the proposed transport approach provides a scalable and interoperable foundation for deploying SRCI-based robotic control across a broad range of industrial automation applications.

Figure 9 illustrates the PLC and robot controller integration model and the complementary use of Class 1 and Class 3 communications.

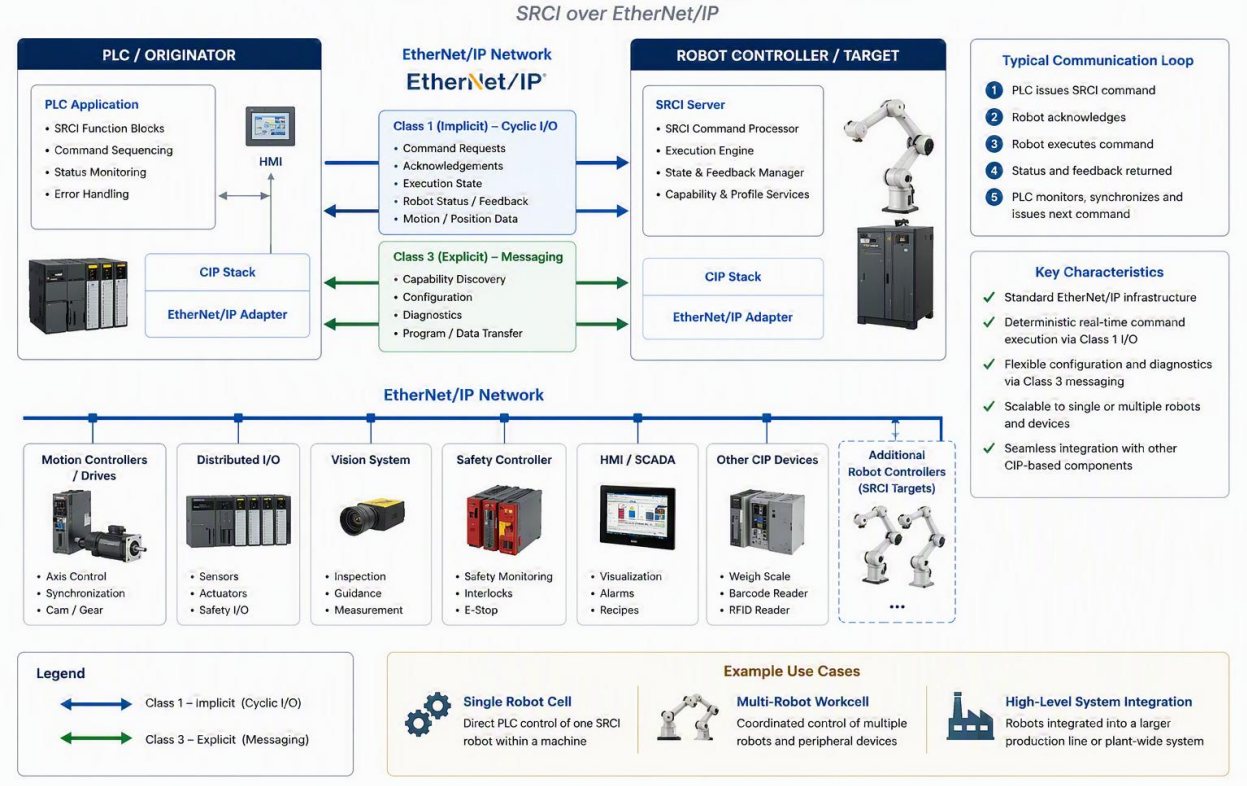


Figure 9. SRCI-over-EtherNet/IP PLC and robot controller integration model.

Interoperability, Safety and Compliance Considerations

Successful adoption of an SRCI-over-EtherNet/IP transport profile depends on establishing a consistent set of interoperability requirements that enable devices from multiple vendors to communicate and behave predictably within a common automation architecture. To achieve this objective, implementations should support standardized object definitions, service interfaces, command representations, execution state reporting, capability discovery mechanisms, and communication behaviors defined by the standard and profile. Particular emphasis should be placed on preserving SRCI command semantics, execution

sequencing, acknowledgement mechanisms, fault reporting, and capability categorization across all compliant implementations. Consistent interpretation of command structures, execution identifiers, status indications, and capability information is essential to ensure that PLC applications can interact with SRCI-compliant robot controllers without requiring vendor-specific adaptations. In addition, profile definitions should clearly identify mandatory and optional behaviors to promote interoperability while allowing flexibility for product differentiation and feature expansion.

The proposed profile should further align with existing ODVA conformance, interoperability, security, and device certification frameworks, leveraging established CIP technologies wherever possible rather than introducing new protocol-specific behaviors. Areas requiring additional standardization may include capability representation, command support categorization, fragmentation behavior, execution state mapping, profile discovery services, and transport-specific error handling to ensure consistent implementation across suppliers. Safety-related functions should continue to remain separate from non-safety SRCI communications and rely upon existing CIP Safety™ technologies or vendor-specific functional safety architectures as appropriate. Likewise, cybersecurity considerations should follow established ODVA security recommendations and CIP Security™ mechanisms where applicable to protect configuration, diagnostics, and device-level communications. To promote broad industry adoption, the profile intends to be accompanied by defined validation procedures, interoperability testing guidelines, and conformance certification criteria that verify consistent implementation of mandatory behaviors and service interactions across all compliant devices. This helps by providing confidence to system integrators, end users, robot manufacturers, and PLC suppliers from an interoperable system standpoint.

Conclusion and Future Work

This paper has presented a proposed approach for extending the applicability of the Standard Robot Controller Interface (SRCI) within EtherNet/IP environments through the definition of a Common Industrial Protocol (CIP)-based transport. By mapping SRCI command semantics, state models, capability discovery mechanisms, and communication behaviors onto existing CIP technologies, the approach aims to simplify PLC-to-robot integration while preserving the vendor-neutral programming model established by SRCI. The proposed architecture leverages widely deployed EtherNet/IP infrastructure and established CIP communication services to reduce integration complexity, improve interoperability among multi-vendor robotic systems, and enable robotic devices to be integrated into existing industrial control architectures without disrupting established engineering practices. The result is intended to be a standards-aligned framework that combines the benefits of SRCI application portability with the scalability, determinism, and ecosystem support provided by CIP technologies.

It is important to recognize that this effort is currently in its early stages, and much of the material presented in this paper represents a proposed architectural direction rather than a finalized specification. Significant work remains to define and standardize the required CIP interfaces/objects or services, application profile structures, capability discovery models, communication behaviors, and conformance requirements necessary to support broad industry adoption. Future activities are expected to include the continued development of an SRCI CIP Application mapping, interoperability validation, and collaboration within the ODVA CIP System Architecture SIG subcommittee which will oversee the SRCI-to-CIP specification integration. Through this collaborative standardization effort, the industry has the opportunity to establish a common foundation for interoperable PLC-to-robot communication that can support advances in robotics, coordinated motion systems and next-generation industrial control architectures.

References

[1] ODVA, The CIP Networks Library, Volume 1: Common Industrial Protocol, Ann Arbor: ODVA, Inc., 2001-2026.
[2] Profile Standard Robot Command Interface SRCI – Version 1.5 January 2025

****

The ideas, opinions, and recommendations expressed herein are intended to describe concepts of the author(s) for the possible use of ODVA technologies and do not reflect the ideas, opinions, and recommendation of ODVA per se. Because ODVA technologies may be applied in many diverse situations and in conjunction with products and systems from multiple vendors, the reader and those responsible for specifying ODVA networks must determine for themselves the suitability and the suitability of ideas, opinions, and recommendations expressed herein for intended use. Copyright ©2026 ODVA, Inc. All rights reserved. For permission to reproduce excerpts of this material, with appropriate attribution to the author(s), please contact ODVA on: TEL +1 734-975-8840 FAX +1 734-922-0027 EMAIL odva@odva.org WEB www.odva.org. CIP, Common Industrial Protocol, CIP Energy, CIP Motion, CIP Safety, CIP Sync, CIP Security, CompoNet, ControlNet, DeviceNet, and EtherNet/IP are trademarks of ODVA, Inc. All other trademarks are property of their respective owners.