Latchline Controls

Legacy program in. Verified documentation out.

Functional Design Specification & Migration Review

PROJECT	Rover Pallet Shuttle (MicroLogix 1400)
CONTROLLER / PROGRAM	CALD26
DATE	03 September 2026

SAMPLE DELIVERABLE

This sample was generated from a publicly available RSLogix 500 program (a MicroLogix 1400 pallet-shuttle controller published on GitHub). It contains no client data.

hello@latchlinecontrols.com

About this document

What this is

A complete engineering documentation package produced from a single export of a running legacy program (Studio 5000 .L5X, or RSLogix 500 .SLC for SLC-500 and MicroLogix). Deterministic analysis recovers the structure — every tag, every reference, every call, every piece of dead or disabled logic — and an AI documentation pass, verified by a controls engineer, recovers the intent. The result is what a migration, audit, or FAT actually needs: a functional specification, a tag dictionary, a full cross-reference, an engineering review findings register, and an executable test plan.

How to read it

Sections 1-3 describe what the program does and where every signal comes from. Section 4 lists every item that needs a human ruling before migration — disabled logic, uncalled routines, suspect tags. Section 5 is a factory acceptance test plan derived directly from the logic: every boolean output the program writes, with the conditions that drive it and drafted test steps. Statements the analysis could not prove from the export alone are marked VERIFY and collected for review — nothing is silently guessed.

The offer

Fixed price, fixed deadline, per program: legacy program in, verified documentation out. Conversion packages (PLC-5 / SLC-500 to ControlLogix with restructured code and FAT support) are quoted per system after a free structural analysis of your export.

Contents

This is a summary excerpt. The full deliverable adds the per-routine functional narratives, every section above, and the complete rung listings.

Functional Design Specification

Scope: 1 program(s), 13 routine(s), 250 rung(s), 1783 instruction(s), 0 tag(s).

Program: Processor

Program overview — CALD26 (MicroLogix 1400, Bul.1766)

What the program controls

The program runs a battery-powered pallet-transport rover. The rover has two wheel drives (X and Y) selected by a motor-driven cam that changes wheel orientation, a three-position lift with analog position feedback, wheel encoders on high-speed counters, down- and up-looking photo eyes plus two Ethernet Keyence sensor units for floor-flag and pallet-presence detection, an Ethernet barcode reader for position ID, and a dock/charger. It receives motion commands from a System PLC ("TC") over a radio/Ethernet link and reports status back. A MobileView pendant provides manual control and can inject the same command frames the radio delivers. Tag descriptions are absent from the export; all names in this specification are inferred (see VERIFY below).

How the routines divide the work

- **LAD_2_MAIN** — loads constants, calls the twelve subroutines in a fixed non-sequential order (3, 4, 12, 5, 7, 8, 6, 9, 11, 10, 13, 14), skips LAD_5_SEQ_MOV in Mobile View mode (B3:10/2), and clears the math-overflow flag S:5/0 every scan.
- **LAD_3_POWER** — power-up detection, battery charging on the dock, low-battery alarm and 5-minute power shutdown, and the two safety-relay feed outputs driven by the network E-stop latch.
- **LAD_4_IO_MAP** — conditions discrete/analog/Keyence inputs into status bits (E-stop status B3:4/0, pallet-present, full-bin, battery %), drives indicator lights and buzzer.
- **LAD_5_SEQ_MOV** — decodes the accepted command, computes distance/speed/accel-decel profile, and sequences cam, lift and wheel requests to the three motor routines.
- **LAD_6_MOTOR_WHL / LAD_7_MOTOR_CAM / LAD_8_MOTOR_LIFT** — axis controllers: wheel velocity profile and analog speed outputs, cam window positioning with inching approach, lift positioning to three analog windows. Each also has a Mobile View manual path.
- **LAD_9_RADIO_IN / LAD_10_RADIO_OUT** — inbound command validation (address, transaction ID, data checks) and outbound 20-word status block plus network stop-command handling.
- **LAD_11_BARCODE** — four-message CIP handshake with the reader; produces position ID N11:39.
- **LAD_12_MBLVIEW** — pendant mode selection and HMI-button command injection into the radio buffer.
- **LAD_13_FAULTS** — thirteen timed fault detectors, single master fault bit and code, radio fault report, post-move encoder verification.
- **LAD_14_VELOCITY** — diagnostic velocity calculation and phase-duration FIFO logs; only F39:28 is consumed elsewhere.

Main operating sequence

1. A command arrives in N11:0–9 (radio, or pendant via LAD_12). LAD_9 validates it, checks rover ID (N10:2) and new transaction ID, and latches command data. Any accepted motion command clears the B30 fault file.
2. LAD_5 sets "sequence active" N7:10/1 (blocked while a fault B30:0/0 is set), computes the profile, and issues requests in command-specific order: cam index (N7:10/11) -> cam settle -> wheel move (N7:10/12, requires charger-clear permissive B9:0/2) -> lift (N7:10/13), or lift first for code 20; codes 5/8/22 are single-step variants.
3. LAD_7 positions the cam into the target window; LAD_6 drives the wheels through accel/cruise/decel/creep and stops on floor flags; LAD_8 moves the lift to the coded position. Completion bits (N7:11/1, N7:11/6, N7:10/15) return to LAD_5.

4. LAD_5 latches move-complete N7:12/0 and requests a radio report; LAD_10 sends the status block and retries every 5 s until acknowledged. LAD_5 then clears the N7:10–N7:13 words. LAD_6 zeroes the encoders and holds position while idle.
5. In parallel: LAD_11 reports new barcodes seen inside a position window, LAD_10 sends a periodic status every ~50 s while stationary, and LAD_3 charges when docked and idle.

Interlock and alarm philosophy

- **Master permissive B3:4/0** (E-stop status from LAD_4) gates every automatic motion path, the sequencer, the barcode cycle and the analog speed outputs. It is standard PLC logic; I:0.0/8 alone satisfies it. VERIFY: existence and coverage of a hardwired safety circuit independent of the PLC, and the polarity/meaning of I:0.0/8–11.
- **Network stops** (99 E-stop, 152 stop, 252 kill) are software commands. Code 99 drops the safety-relay feed outputs, which LAD_3 re-arms automatically after 3 s. VERIFY (safety): acceptability of auto re-arm and of communications-mediated stops that wait for any in-flight message.
- **Gaps to review:** wheel drive enables (O:0.0/2, /3) are never unlatched on fault or E-stop in the PLC; lift jog and cam jog bypass B3:4/0 and have only analog/software travel limits; in Mobile View, analog outputs and enables hold last values when a button is released. VERIFY each with the plant safety assessment.
- **Fault handling** is centralised in LAD_13: the first fault to trip latches B30:0/0, writes one code to N11:41, sets a code bit in B30:30–37, latches drive inhibit N7:11/10, and reports by radio up to three times. A fault blocks new sequences (LAD_5) and zeroes encoders (LAD_6) but does not directly stop cam or lift motors. Reset is remote only — any accepted motion command or code 152 clears the whole B30 file (LAD_9). No communication loss to the Keyence units, barcode reader or System PLC is detected as a fault.
- **Overflow masking:** LAD_2 clears S:5/0 every scan, so divide-by-zero and overflow conditions program-wide are silent.

VERIFY: Many gating and configuration bits (B3:8/0–4, B3:9/2, B3:10/5–15, B3:11/x, B3:13/0, B3:16/x, N7:3, all setpoints) have no PLC writer and are assumed HMI- or data-table-sourced; their values and sources must be captured before migration.