Smart Conveyor System with Automated Inspection & Reject Handling

Studio 5000 Emulator-Based Automation Project

---

1. Project Objective

This project implements a **multi-zone conveyor line** that detects, inspects, and rejects defective parts automatically.
It was built entirely in **Studio 5000 Logix Designer** and tested with the **ControlLogix 5570 Emulator**, allowing a full industrial-style design without physical hardware.

The system mimics a real manufacturing inspection station where products move through several conveyor zones, undergo quality checks (label, seal, weight), and are either passed downstream or automatically diverted if they fail.

---

2. Scope of Work

- Implement **Auto/Manual mode selection** with safe start/stop sequencing.
- Develop **zone conveyor control** using reusable AOI_ConveyorMotor blocks.
- Build a **FIFO queue** (FFL/FFU) to track every part from entry to exit.
- Implement **inspection logic** (label, seal, weight) and pass/fail result assignment.
- Actuate a **reject mechanism** automatically for failed parts.
- Design a **scalable alarm/notification system** using AOI_Alarm.
- Organize ladder logic into **modular routines** for clarity and reuse.
- Simulate and test the entire system in the **Studio 5000 Emulator** — no chassis or real I/O needed.

---

3. Modes of Operation

Manual Mode

- Operator can jog conveyor zones, trigger the reject actuator, and test sensors individually.
- Useful during debugging and simulation.

Auto Mode

- Start command initiates full conveyor sequence.
- Each detected part enters the **FIFO queue**, moves zone-by-zone, is inspected, and either passes or is rejected automatically.
- Alarms trigger on jams, inspection timeouts, or motor feedback faults.

---

4. Software Architecture

The controller is organized into **clear, purpose-built routines**:

Routine	Purpose
MainRoutine	Entry point; calls all modular routines
IO_Map	Logical tag mapping and virtual I/O stubs
Mode_Handling	Auto/Manual selection and safety interlocks
Conveyor_Control	Zone motor logic (start/stop, feedback proving, fault detection)
Part_Inspection	Simulated quality checks and result assignment
FIFO	FFL/FFU-based queue for part tracking
Alarm_Notification	Centralized alarm logic built on AOI_Alarm
ONS_RE	One-shot rising edge triggers for sensors
Sim_Routine	Allows forcing and testing without hardware
PowerUp_Reset	Clears queues and resets system safely at startup

Reusable Building Blocks

- **AOI_ConveyorMotor** – Standard motor start/stop with feedback proving and fault detection.
- **AOI_Alarm** – Modular latching alarm with trip, acknowledge, and reset signals.
- **UDT_Alarms** – Alarm metadata and state bits.
- **UDT_PartInfo** – Per-part inspection data (label/seal/weight/result).

5. UDTs & AOIs

5.1 User Defined Data Types

UDT Name	Purpose
UDT_ConveyorMTR	Encapsulates motor run command, feedback, and fault bits.
UDT_Alarms	Holds alarm trigger, latch, acknowledge, and active status bits.
UDT_PartInfo	Tracks each part's LabelOK, SealOK, Weight, and Result (PASS/FAIL).

5.2 Add-On Instructions

AOI_ConveyorMotor

- Inputs: Cmd_Run, Fb_Running
- Outputs: Fault, Cmd_Out
- Handles motor start/stop and feedback proving.

AOI_Alarm

- Inputs: TripCond, Ack
- Outputs: Latched, Active
- Provides latching alarm logic with trip, acknowledge, and reset signals.

---

6. Functional Test Criteria

Test Case	Expected Behavior
Start in Auto & toggle RunCmd	Zone 1–3 motors start sequentially; conveyor runs
Simulate PE_Entry pulse	Part added to FIFO queue
Set Label_OK=0 or Seal_OK=0	Part flagged as FAIL
Simulate PE_Inspect	FIFO updates; inspection routine runs
FAIL part reaches reject	RejectCylinder actuates; part removed

Jam condition (JamDetect=1)	Alarm trips; AlarmHorn=ON; alarm visible
Trigger ResetAlarms	Clears all active/latched alarms
Manual Mode ON	Individual zones can be jogged; reject works manually

---

7. Summary & Highlights

This project demonstrates:

- **Data-driven PLC programming** with UDTs & AOIs.
- **FIFO-based part tracking** — common in production/inspection lines.
- **Reusable motor & alarm logic** ready for scaling.
- **Auto/Manual mode handling** for safe commissioning.
- **Fully emulated testing** — no hardware required.

The design mirrors **OEM/integrator standards** for modularity, scalability, and maintainability