J. Sustain. Wireless Syst.
Vol.01/ No. 04
Pages: 206-214
http://irojournals.com/irosws/
DOI: https://doi.org/10.36548/jsws.2019.4.001

Process Control Ladder Logic Trouble Shooting Techniques Fundamentals

Khaled Kamel
Computer Science Department,
Texas Southern University,
Houston, TX

Eman Kamel
PLC Automation,
Manvel, TX

Abstract

Programmable Logic Controller (PLC) based process control automation projects are a multi-phase task requiring months of planning, specification documentation, design, implementation, debugging, and final commissioning. Debugging real time process control implementation is a three-phase process; logic simulation debugging, field static checkout, and in plant final testing and tuning. This paper briefly discusses the fundamental techniques used in the first two stages of implemented ladder logic debugging [5, 6]. A wastewater treatment facility for the processing of high flow storm rainwater using the Allen Bradley (AB) SLC 500 PLC is used to illustrate the concepts discussed. The rainwater is channeled to two large wet wells, the east wet well and the west wet well. The water is pumped to the river from the two connected wells at constant rate using a predefined process sequence control. Pumping station with multi motors deriving constant speed immersed pumps are used to regulate the discharge water flow and the level in the two wells. Only two pumps are assumed here; one in the east wet well and the other in the west wet well. The motors provide an input discrete signal indicating if the motor is running or not. The motors can also start by activating the Push Button located on the local panel if the AUTO/MAN switch is in Manual.

Three float switches are used to provide an accurate indication of the water level at three pre-specified critical east / west wet well locations. The Low-Level Float switch triggers the stopping of the running pump. The High-level Float switch triggers the starting of the scheduled pump. If the scheduled pump fails to start within more than 5 seconds, a backup pump is selected and started (not shown in this paper). An alarm must be issued in order to alert the operator of any motor failure. The Very High-level float switch triggers the starting of both pumps. If either of the two pumps fails to start the corresponding alarm is activated by the control. Pumps are scheduled to run according to an operator pre-defined calendar. This input is expected in hours of accumulated total pump run time. The two pumps must alternate while the water level is below the Very High Level and above the Low Level. Cascaded timers are not altered during the time when the two pumps are running. This paper shows the static and simulation debugging an abbreviated version of the typical original system used in wastewater facilities.

J. Sustain. Wireless Syst.
Vol.01/ No. 04
Pages: 206-214
http://irojournals.com/irosws/
DOI: https://doi.org/10.36548/jsws.2019.4.001

I. Process Description

In this illustration a small and abbreviated part of a real implemented automation project involving a wastewater treatment pump station using Allen Bradley SLC 500 PLC control is used [1, 2]. The main goal of this paper is to demonstrate process control structured ladder logic debugging techniques. The following tables list all inputs, outputs, and internal reference addresses used in this abbreviated task; physical address, tag name, and the associated functions.

System Inputs

Tag Name	Address Number	Name Tags	Comments
Off Float	I:1/0	SS0	NO Wet Well Low Level Float Switch
On Float	I:1/1	SS1	NO Wet Well High Level Float Switch
Override Float	I:1/2	SS2	NO Wet Well Very High Level Float Switch
E Pump Input	I:1/3	SS3	East Wet Well Pump Running Input
W Pump Input	I:1/4	SS4	West Wet Well Pump Running Input
AUTO	I:1/5	SS5	System In Auto

System Outputs

Tag Name	Address Number	Name Tags	Comments
E Pump	O:2/0	PL1	East Pump Run Output
W Pump	O:2/1	PL2	West Pump Run Output
E Pump Common Alarm	O:2/2	PL3	East Pilot Light Alarm
W Pump Common Alarm	O:2/3	PL4	West Pilot Light Alarm

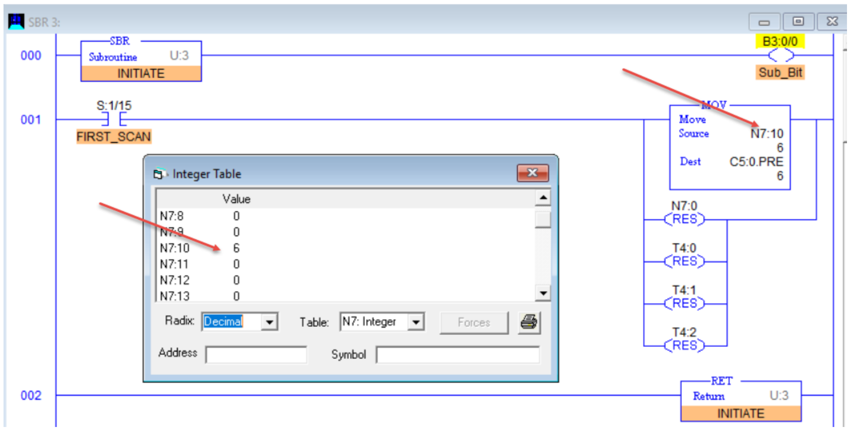
Internal References

T4:0	East Pump One Hour Accumulation Timer
T4:1	West Pump One Hour Accumulation Timer
C5:0	East Pump Cascaded Timer
C5:1	West Pump Cascaded Timer
N7:0/0	East/West Pump Alteration Bit

II. Ladder Logic Implementation

J. Sustain. Wireless Syst.
Vol.01/ No. 04
Pages: 206-214
http://irojournals.com/irosws/
DOI: https://doi.org/10.36548/jsws.2019.4.001

Four subroutines were implemented to realize the process control specifications; INITIATE (U3), PUMP ALTERNATION (U4), PUMP START/STOP (U5), and PUMP ALARMS (U6) [3, 4]. The U3 subroutine function shown in Figure 1 is to reset all timers / counters / registers used in the program and initialize the schedule calendar for the two pumps. The U4 subroutine shown in Figure 2 implements the logic to alternate between the East / West pumps based on the status of the least significant bit of N7:0 (even/odd) at the end of each calendar expiration. The U5 subroutine in Figure 3 starts /stops the pumps based on the Schmitt trigger logic; below the low threshold of water level both pumps are not running, above the very high water level both pumps are running, and in between the pump will latch the previous status to run or not to run. Finally, Figure 4 subroutine U6 shows the pumps alarms. The alarm will be triggered every time the output is sent to the pump and the running input from the pump motor is not received within 5 seconds.



	Value
N7:8	0
N7:9	0
N7:10	6
N7:11	0
N7:12	0
N7:13	0

Figure 1 Initialization Subroutine U3

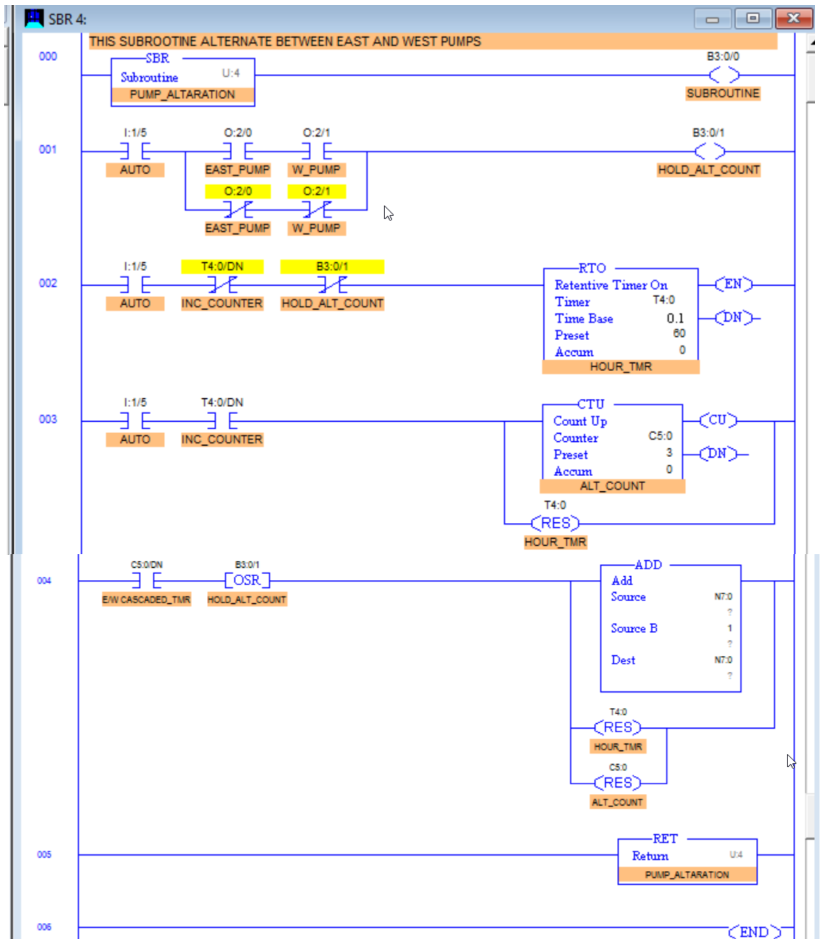
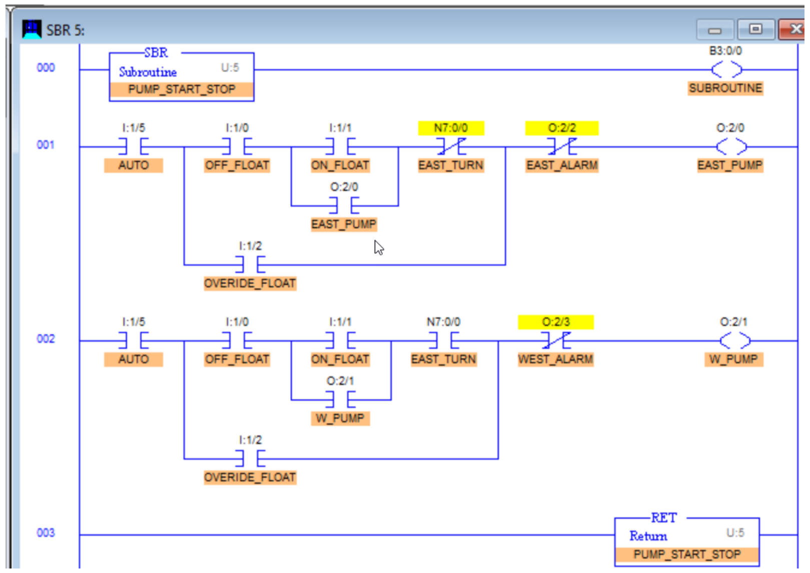


Figure 2 Pump Alternation Subroutine U4



J. Sustain. Wireless Syst.
Vol.01/ No. 04
Pages: 206-214
http://irojournals.com/irosws/
DOI: https://doi.org/10.36548/jsws.2019.4.001

Figure 3 Pump Start / Stop Subroutine U5

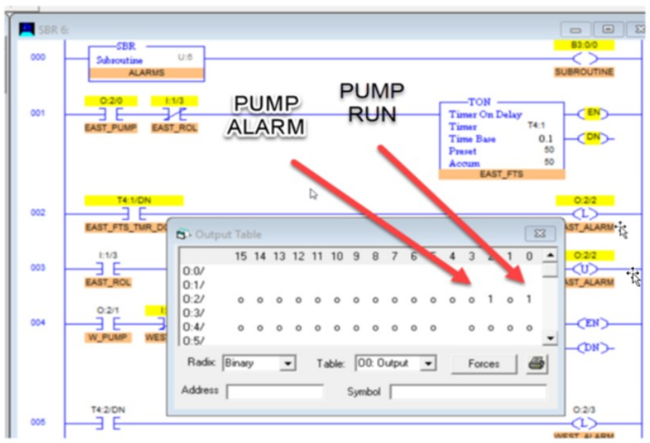


Figure 4 Alarms Subroutine U6

III. Ladder Logic Simulation Debugging

During the simulation logic debugging, each subroutine is enabled / disabled using four inputs; I: 3/1, I: 3/2, I: 3/3, and I: 3/4 in the main program U2 (Lad 2 :) as shown in Figure 5. The following are the main steps to check out the implemented U3 logic using LogixPro 500 simulator before the final check out on the real system:

1. Go to the data table integer file N7:10. Select decimal then inter the calendar time duration value you choose to employ for alternating between the two pumps (East and West). The value used in this illustration is 6 simulating a 6 month alternation schedule as shown in Figure 1.
2. Enable U3 by activating I: 3/1 to check out the INITIATE logic while disabling the inputs to the other three subroutines
3. Verify that the new calendar schedule value of 6 exists in counter 5 preset value tag name C5:0.PRE, which was stored in assigned register N7:10 prior to running the program.
4. The system preconfigured first scan pulse S:1/15 is used to execute the initialization / configuration code only once after power on.
5. N7:0/0 is the bit used for the two pumps alteration. N7:0 register is incremented at the end of every calendar expiration. The east pump runs on even N7:0 values and the west pump on the odd values.
6. Verify that N7:0, T4:0, T4:1, and T4:2 are reset to zero.

U4 and U5 are two interlinked subroutines logic and thus must be checked together. The following are the main steps to check out the implemented U4 and U5 logic using LogixPro 500 simulator before the final check out on the real system:

1. Activate I: 3/2 and I: 3/3 to enable the PUMP_ALTERNATION and the PUMP_START_STOP subroutines.
2. From subroutine 4, activate I: 1/5, the AUTO switch, and start East pump to enable HOLD_ALT_COUNT bit B3:1/0. Monitor the alternating Counter C5:0 (ALTERNATING_COUNTER) and the incrimination of register N7:0 indicating that the 6 month passed and it is time to start the west pump.
3. This ping pong between the east and west pump will run till the stop / emergency switch is activated.
4. While Subroutine 4 and 5 are active and the program is in AUTO, start both pumps by activating I: 1/0 and I: 1/1. Confirm that East pump is running. Switch to Subroutine 4 and monitor N7:0 incrimination, which cause the alternation between both pumps.

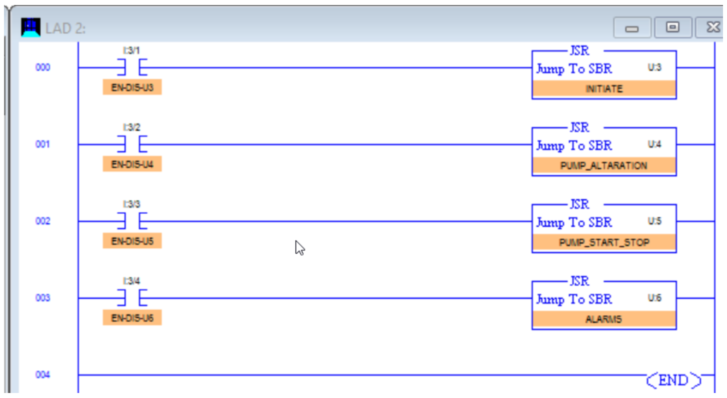


Figure 5 Main Ladder Program

Disable all subroutines except Subroutine 6. Enable Subroutine 6 by activating I: 3/4. While in subroutine 6 click right, go to data table and switch radix to binary and enter 1 to enable O: 2/0 the east pump output since forcing output is not permitted in the LogixPro simulation software. This will trigger the east pump alarm and start the east pump timer, after 5 seconds the east pump alarm will latch. Once east pump is fixed the running input will unlatch the east alarm output. Repeat same procedure to check the west pump. Clear all entries from the data table before exiting. Forcing output is available in the real PLC systems by placing the PLC in Test mode. If the PLC does not support the Test mode, you can perform the output forcing after removing the output modules fuses.

IV. Operating Modes of the CPU

The CPU has three modes of operation; PROGRAM, REMOTE, and RUN as shown in Figure 6. The following are the characteristic of each of the three CPU modes and two additional states:

J. Sustain. Wireless Syst.
Vol.01/ No. 04
Pages: 206-214
http://irojournals.com/irosws/
DOI: https://doi.org/10.36548/jsws.2019.4.001

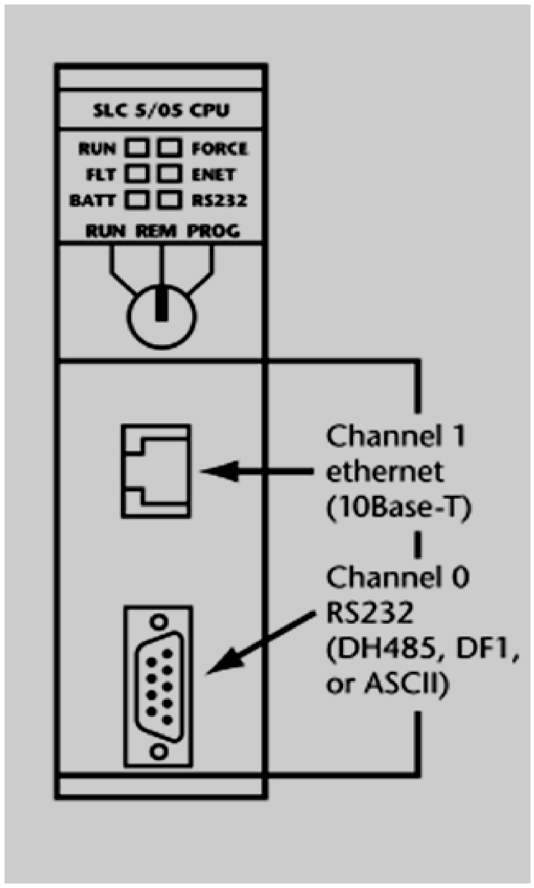

SLC 5/05 CPU
RUN
FORCE
FLT
ENET
BATT
RS232
RUN REM PROG
Channel 1 ethernet (10Base-T)
Channel 0 RS232 (DH485, DF1, or ASCII)

Figure 6 PLC Processor Modes

1. In the PROGRAM mode, the processor is accepting and compiling new instructions; either as a new program or as changes to an existing program.
2. In the TEST state, which is available from the remote mode, the processor reads inputs and solves ladder program, but does not allow field devices to be energized. Test mode is used to test a program during installation, maintenance, or troubleshooting.
3. In RUN mode, the scan cycle is executed repeatedly in the processor memory and outputs is activated according to the implemented program logics. Once the creation or editing of a program is complete, the processor is put into the run mode to execute the stored code. Program cannot be downloaded in this mode.

V. Static Input / Output Wiring Checkout

A static input wiring checkout should be performed with power applied to the controller and input devices. This check will verify that each input device is connected to the correct input

J. Sustain. Wireless Syst.
Vol.01/ No. 04
Pages: 206-214
http://irojournals.com/irosws/
DOI: https://doi.org/10.36548/jsws.2019.4.001

terminal and that the input modules are functioning properly. Proper input wiring can be verified using the following procedures:

1. Place the controller in the TEST mode this mode will inhibit the PLC from any automatic operation.
2. Apply power to the system power supply and input devices. Verify that all system diagnostic indicators show proper operation. Typical indicators are AC OK, DC OK, processor OK, memory OK, and I/O communication OK.
3. Verify that the emergency stop circuit will de-energize power to the I/O devices.
4. Manually activate each input device. Monitor the corresponding LED status indicator on the input module and/or monitor the same address on the programming device, if used. Also verify the associated bits status in the input image table. If properly wired, the indicator will turn ON. If an indicator other than the expected one turns ON when the input device is activated, the input device may be wired to the wrong input terminal. If no indicator turns ON, then a fault may exist in either the input device, field wiring, or input module.
5. Take precautions to avoid injury or damage when activating input devices that are connected in series with loads that are external to the PLC.

A static output wiring check should also be performed with power applied to the controller and the output devices. A safe practice is to first locally disconnect all output devices that involve mechanical motion (e.g., motors, solenoids, etc.) or place the PLC in a TEST Mode. Proper output wiring can be verified using the following procedures:

1. Verify that each output device is connected to the correct terminal address and that the PLC is in TEST mode.
2. While the PLC is in TEST mode, force every discrete output ON/OFF and verify that it is responding correctly.

Fully simulated and debugged program must be downloaded to the PLC processor memory before switching to the RUN mode. For final check out, place the PLC in RUN Mode, and avoid changing any parameters while the Program is running online unless is done by expert programmer and the change is minor like Timer / Counter preset value. Critical changes can be made while the PLC in program mode and must be fully simulated and debugged as stated in Section IV before switching to the run mode [7, 8].

Conclusion

This paper briefly discussed the fundamental PLC debugging techniques including ladder logic simulation and testing, static input / output wiring checkout, and the final online system commissioning in the RUN mode. An abbreviated waste water treatment pump stations process control was used to demonstrate the debugging concepts. Structured programming methodology was used with emphasis on simplifying the overall debugging task. Great care *and safety precaution must be taken during the debugging / testing and thus this task cannot be delegated to non-experts. In the vast majority of projects, the entire debugging task including the time intensive static input / output wiring / instrumentation calibration must be run / supervised by the expert PLC programmer with adequate technician support as needed.*

References

1. Eman Kamel and Khaled Kamel, "Hands-On PLC Programming with RSLogix 500 and LogixPro", McGrew-Hill professional, (ISBN-13: 978-1259644344), August 2016.
2. Khaled Kamel and Eman Kamel, "Programmable Logic Controllers: Industrial Control", First Edition, McGraw-Hill Professional (**ISBN:** 0071810455 / 9780071810456), August 2013.
3. Khaled Kamel and Eman Kamel, "Waste Water Treatment Wet wells Pump Station PLC Control", International Journal of Science and Engineering Technology, Vol. 1, October 2015.
4. Khaled Kamel and Eman Kamel, "PLC Motor Control Fundamentals and Safety Features: A Case Study", Journal of Computer Science Applications and Information Technology, December 2018.
5. Drives & Systems, "Troubleshooting Industrial Control Systems – Tips and Pointers", December 6, 2017.
6. Dave Perkon, Technical Editor Control Design, "How to document a PLC program", May 20, 2019.
7. Edvards Csanyi, EEP Electrical Engineering Portal, "PLC start up and checking procedures before applying the power", September 23, 2016.
8. "Diagnosis and debugging of programmable logic controller control programs by neural networks", proceedings of the 2005 IEEE Conference on Automation Science and Engineering.