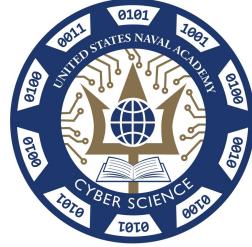




UNITED STATES  
NAVAL ACADEMY

Annapolis



# PLC and Ladder Logic Basics Workshop

Brien Croteau, USNA, Cyber Science

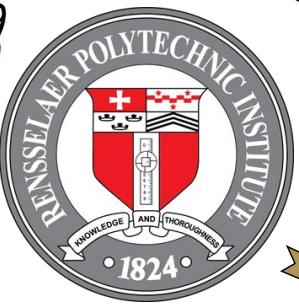
DefCon 31, ICS Village, 12 Aug 2023



Link to these slides:  
[https://github.com/brienc23/Defcon31\\_workshop\\_materials](https://github.com/brienc23/Defcon31_workshop_materials)

# My Background

24-year Active Duty Naval Officer, EE Ph.D.  
Assistant Professor, USNA Cyber Science Department  
Military Deputy for the Dean of Math and Science



EA-6B Naval Flight Officer



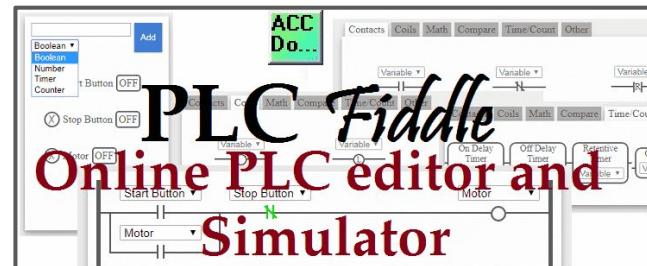
Research in Cyber-Physical Systems (CPS) Security: Detection of malicious sensors using side-channel power analysis, Alternate actuation paths, Actuation limits, Industrial Control Systems (ICS) security, Maritime Hull, Mechanical, & Electrical (HM&E) security

# Workshop Outline

A gentle introduction to programming Industrial Controllers

Three Recommended Free Tools:

1. PLC Fiddle (browser-based Ladder Logic Simulator)
2. OpenPLC (open-source PLC software & hardware)
3. Rockwell/Allen-Bradley CCW (commercial micro800 PLCs)



**Disclaimer: None of these products are endorsed by: US Gov, US Navy, or USNA**

# What is a PLC?

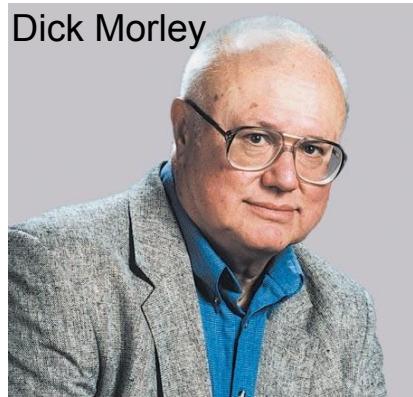
A [programmable logic controller](#) (PLC) or programmable controller is an industrial computer that has been ruggedized and adapted for the control of manufacturing processes, such as assembly lines, machines, robotic devices, or any activity that requires high reliability, ease of programming, and process fault diagnosis.



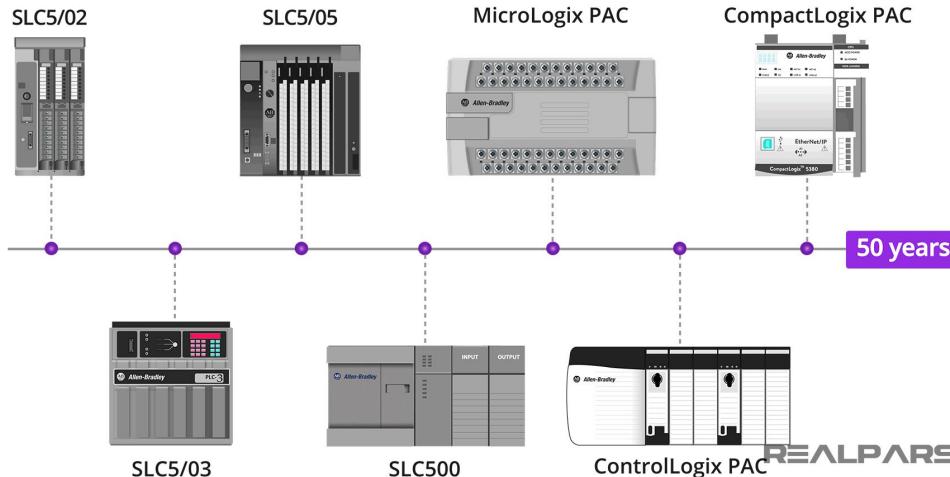
# History

There are two men credited as being the "father" of the PLC.

- Richard E. Morley (1932-2017) was an American mechanical engineer who was involved with the production of the first PLC for General Motors, Modicon, and Bedford Associates in 1968.
- Odo Josef Struger (1931-1998) was involved in the invention of the Allen-Bradley programmable logic controller (PLC) and coined that term, during 1958 to 1960 based on a concept developed in his doctoral dissertation at the Vienna University of Technology.



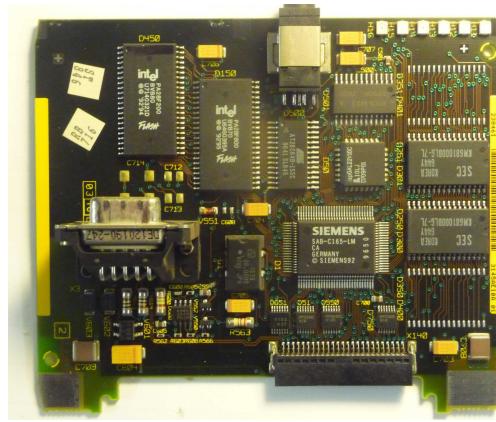
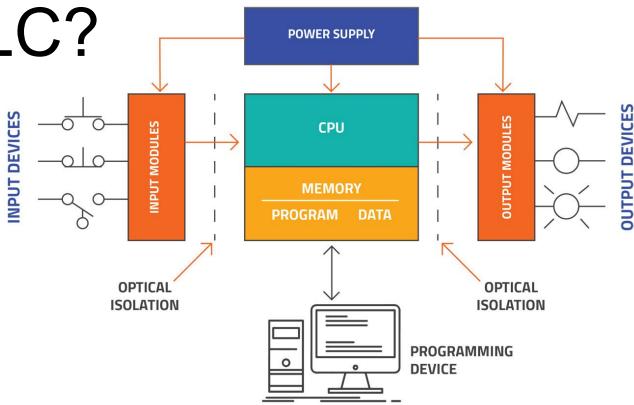
**MODICON**  
Schneider  
Electric



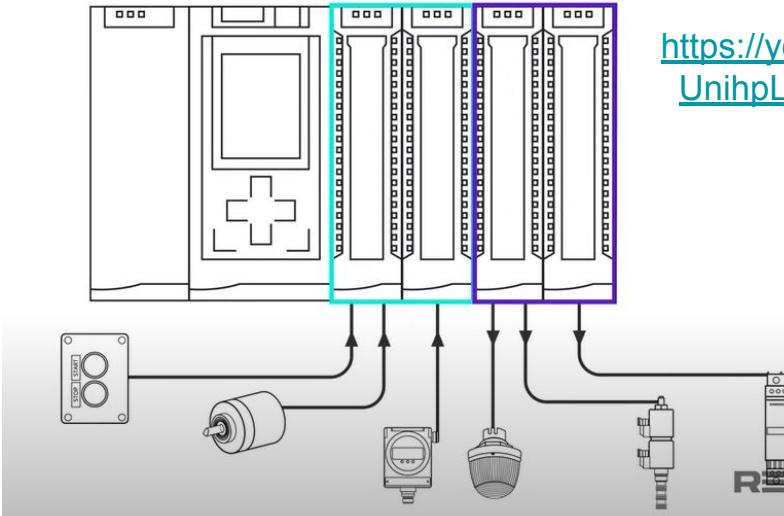
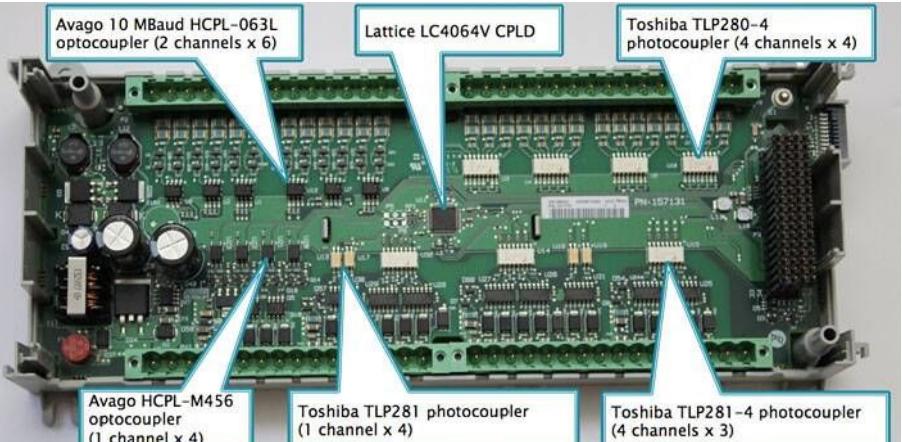
**Allen-Bradley**  
by ROCKWELL AUTOMATION

# What is inside a PLC?

- Power Supply
- Processor
- Input Modules
- Output Modules
- Interface Modules
- Programming Interface



SIEMENS CPU314IFM with  
C165 series processor



<https://youtu.be/pPUnihpL6UI?t=240>

# Simple PLC Example

"Figure 8 shows control of a manufacturing process being performed by a PLC over a fieldbus network. The PLC is accessible via a programming interface located on an engineering workstation, and data is stored in a data historian, all connected on a LAN."

<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r3.ipd.pdf>

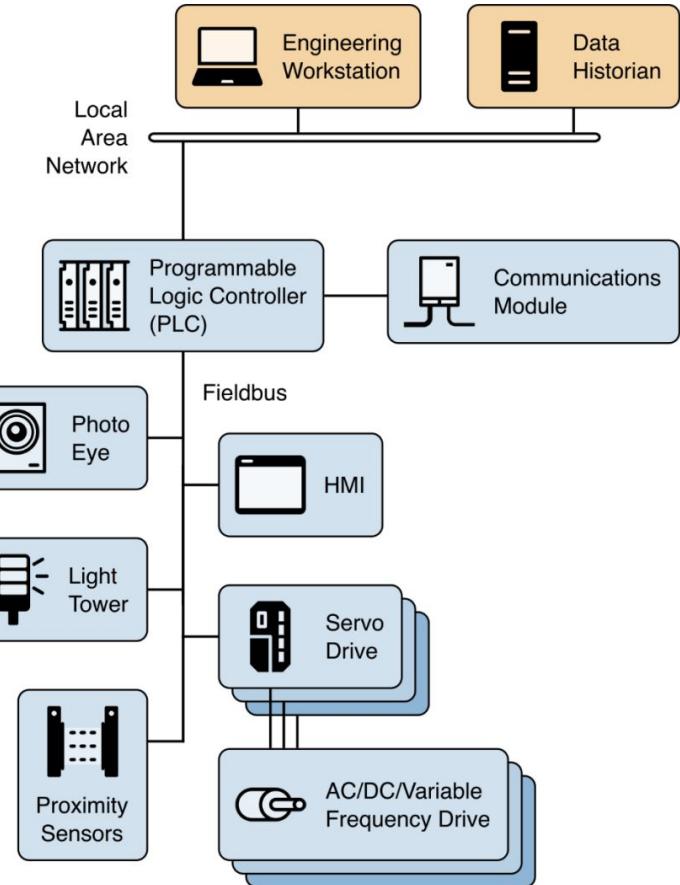
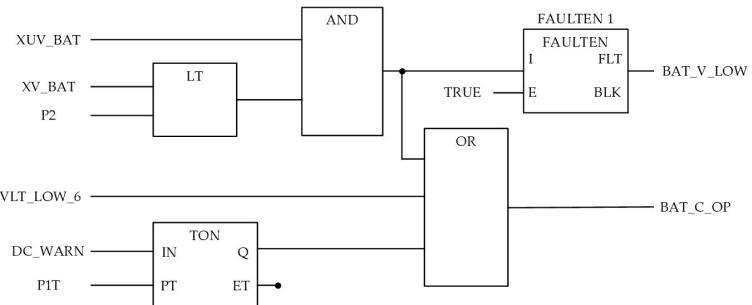


Figure 8: A PLC control system implementation example

# PLC Programming

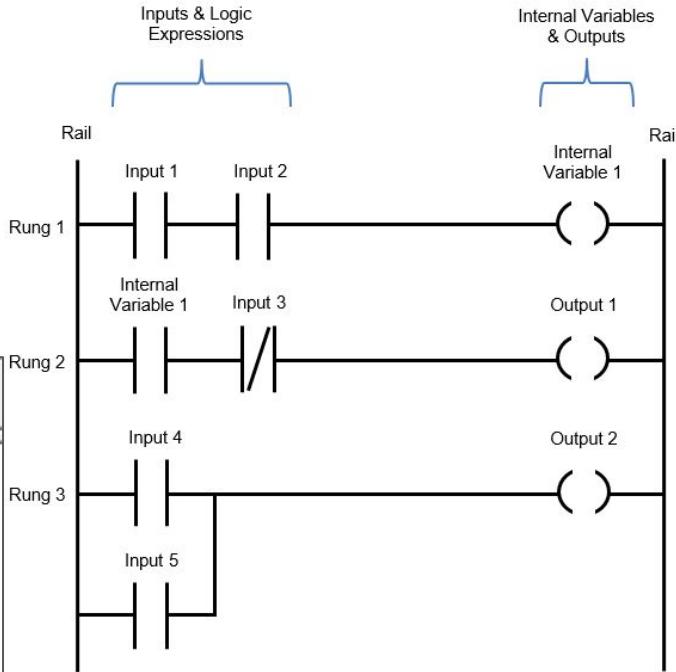
Standardized with [IEC 61131-3](#) which defines three graphical and two textual programming language standards:

- Ladder diagram (LD), graphical
- Function block diagram (FBD), graphical
- Structured text (ST), textual
- Instruction list (IL), textual (deprecated)
- Sequential function chart (SFC), graphical



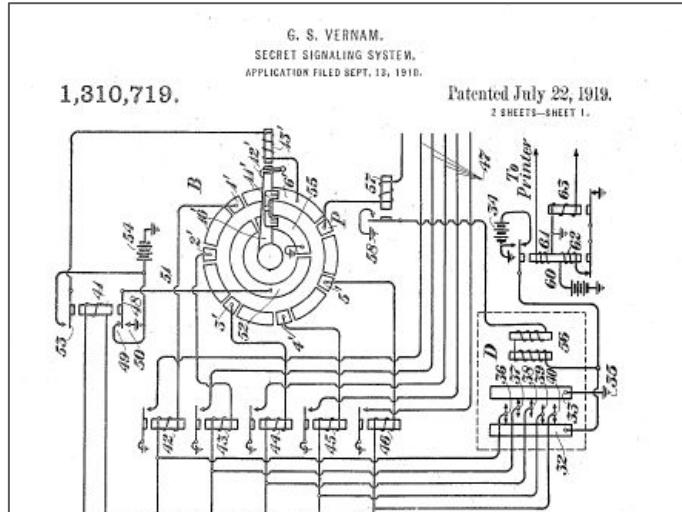
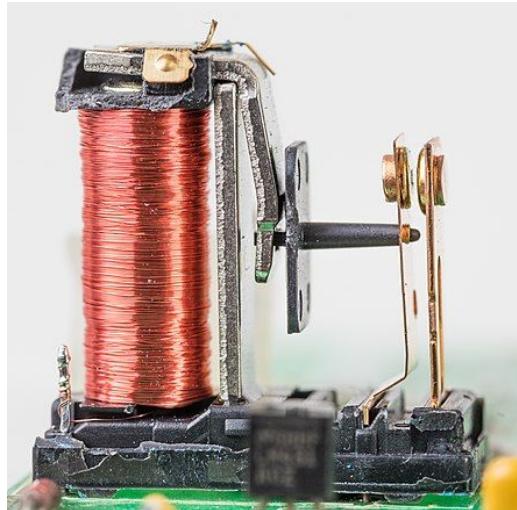
```
IF Start_PB THEN
    MOTOR_RUN_RELAY := 1;
END_IF;

IF Stop_PB THEN
    MOTOR_RUN_RELAY := 0;
END_IF;
```

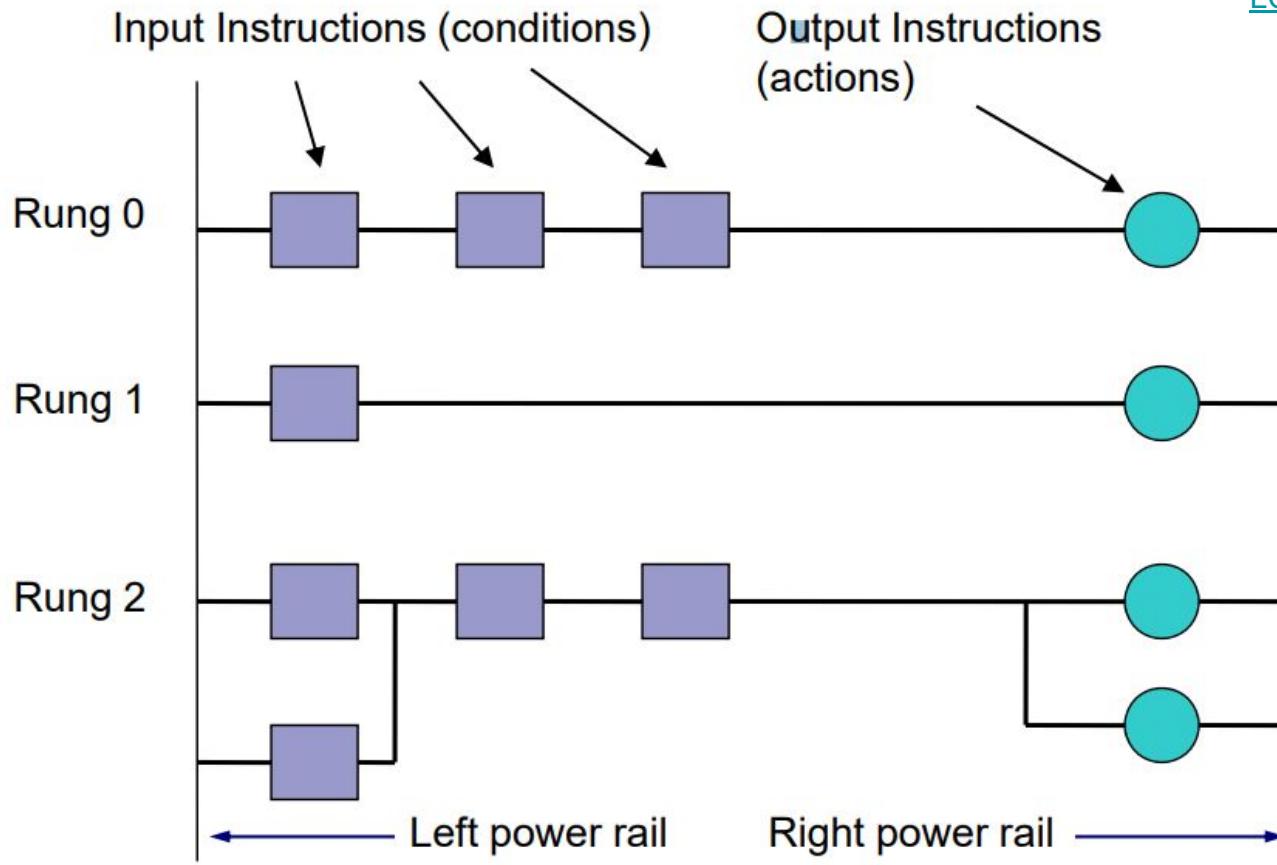


# Where does LL come from?

Ladder logic was originally a written method to document the design and construction of [relay racks](#) as used in manufacturing and [process control](#).<sup>[1]</sup> Each device in the [relay](#) rack would be represented by a symbol on the ladder diagram with connections between those devices shown. In addition, other items external to the relay rack such as pumps, heaters, and so forth would also be shown on the ladder diagram.



# Anatomy of a Ladder Program



# Typical Symbols

NO Contact



Positive Transition-Sensing Contact



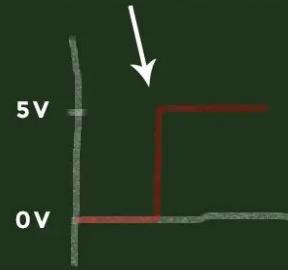
NC Contact



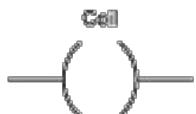
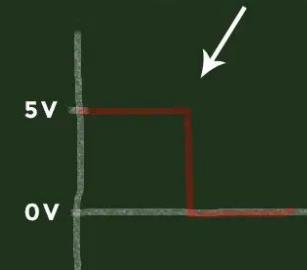
Negative Transition-Sensing Contact



POSITIVE EDGE



NEGATIVE EDGE



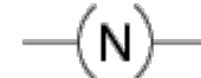
Positive Transition-Sensing Coil



[https://www.plcacade  
my.com/ladder-logic-  
symbols/](https://www.plcacade my.com/ladder-logic-symbols/)

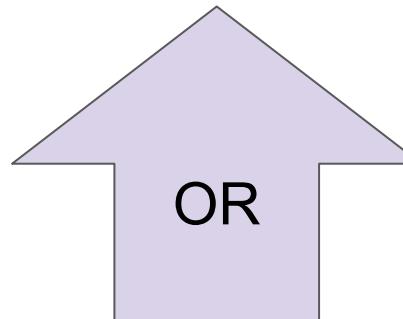
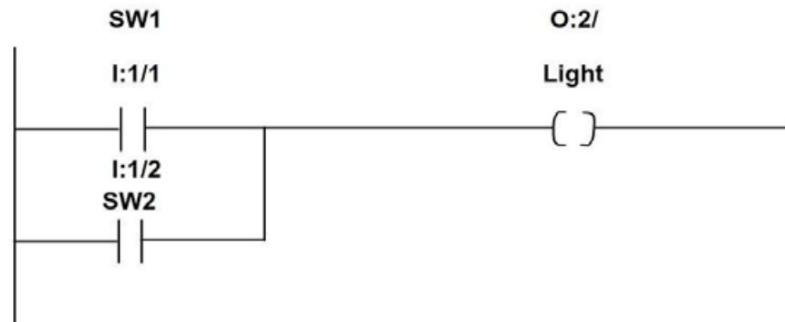
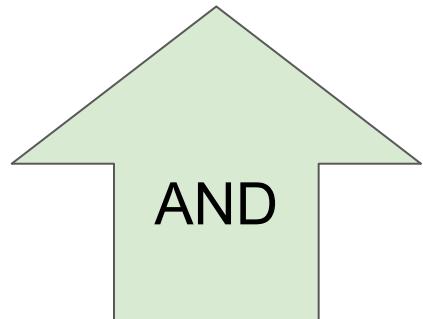
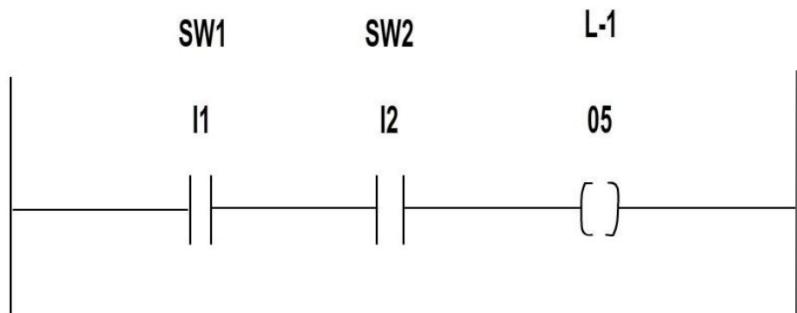


Negative Transition-Sensing Coil



# Basic Logic

<https://www.philadelphia.edu.jo/academics/waraydah/uploads/Introduction%20to%20PLC%20and%20Ladder%20Logic%20Programming.pdf>

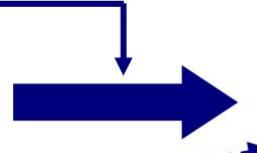


# The Scanning Process

[https://my.ece.utah.edu/~ece3510/  
Ladder%20Logic%20Fundamentals%20PLC%20tutorial.pdf](https://my.ece.utah.edu/~ece3510/Ladder%20Logic%20Fundamentals%20PLC%20tutorial.pdf)

Read inputs

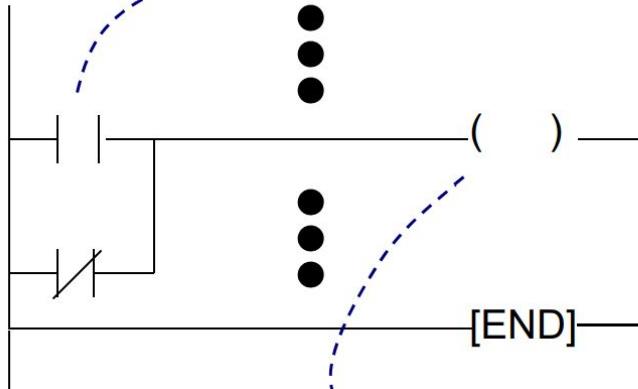
All input  
Terminals



Input Image  
Table

Solve the ladder program

(update output image table as  
necessary)



Left to Right  
Top to Bottom

Update outputs

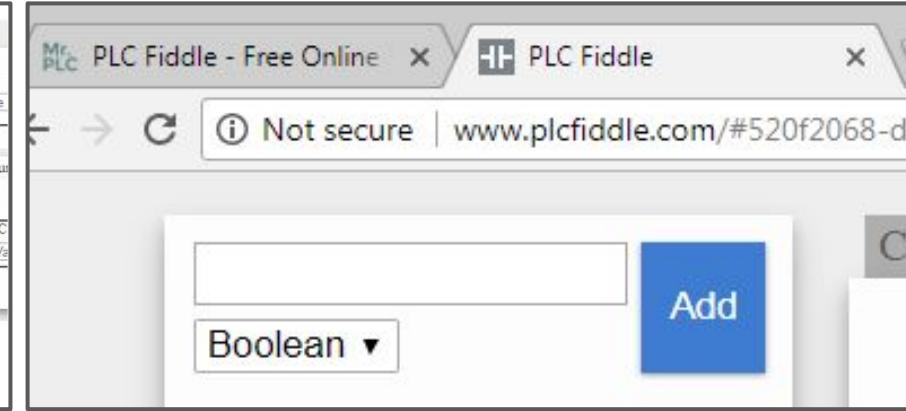
All output  
Terminals



Output Image  
Table



PLC Fiddle  
Online PLC editor and  
Simulator



# 1. PLC Fiddle

# About PLC Fiddle

<https://www.plcfiddle.com/>

PLC Fiddle is a free online PLC Ladder Logic Editor and Simulator that works in your browser.

It includes:

1. Sandbox-like simulator = "Playground"
2. Interactive Tutorial/Lessons = "Code School"

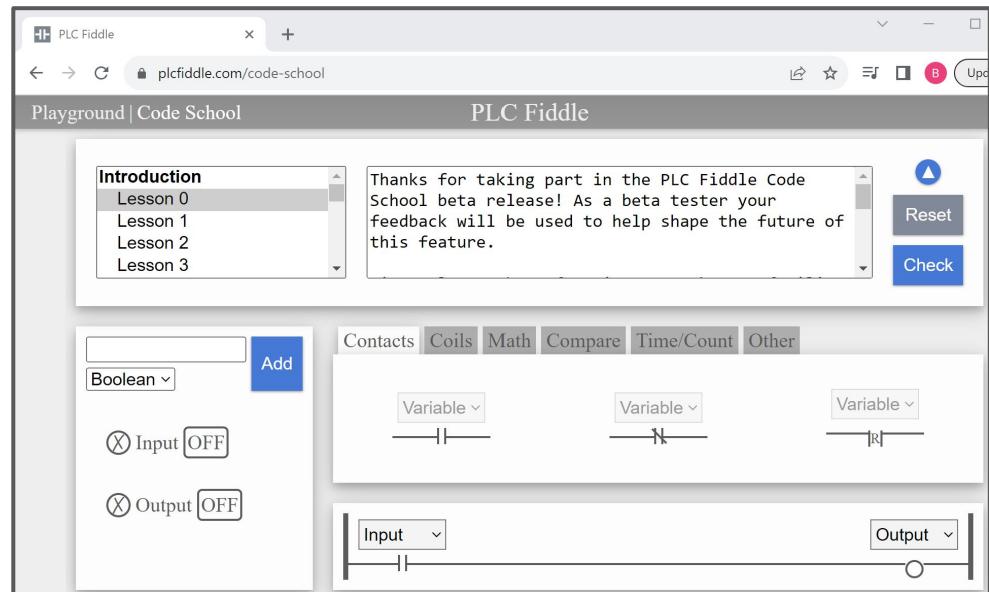
The screenshot shows the 'Playground' section of the PLC Fiddle interface. It features a ladder logic editor with three parallel rungs. The top rung has two normally open contacts (labeled 'Variable') in series. The middle rung has one normally closed contact (labeled 'N') in series. The bottom rung has one normally open contact (labeled 'NR'). To the left of the ladder, there are input and output terminals labeled 'Input' and 'Output'. Below the ladder, there are dropdown menus for 'Boolean', 'Contacts', 'Coils', 'Math', 'Compare', 'Time/Count', and 'Other'. A blue 'Add' button is located at the top right of the editor area. At the bottom, there is a message: 'We love hearing from you! [Contact us](#) to share how you're using PLC Fiddle!' and a note about transitioning to paid features.

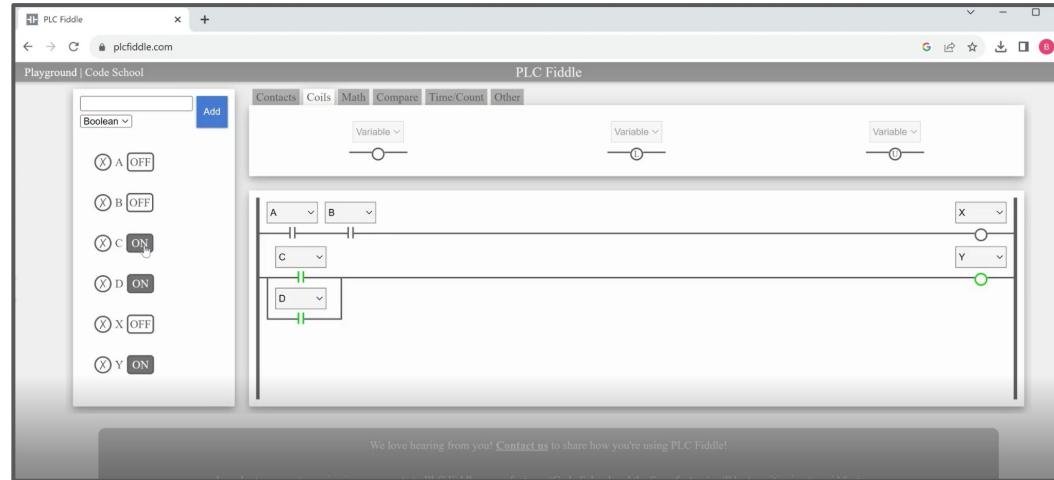
The screenshot shows the 'Code School' section of the PLC Fiddle interface. It displays an 'Introduction' panel with a list of lessons: 'Lesson 0', 'Lesson 1', 'Lesson 2', and 'Lesson 3'. To the right of the introduction, there is a message: 'Thanks for taking part in the PLC Fiddle Code School beta release! As a beta tester your feedback will be used to help shape the future of this feature.' Below the introduction, there is a ladder logic editor with three parallel rungs, identical to the one in the playground. The interface includes the same navigation tabs ('Boolean', 'Contacts', etc.) and an 'Add' button. On the right side, there are 'Reset' and 'Check' buttons. The number '5' is visible in the bottom right corner.

# PLC Fiddle "Code School" Topics

Here is a list of the topics the Code School covers:

1. Introduction (to the Interface)
2. Boolean Logic Basics
3. Intro to Counters
4. Intro to Timers
5. Working with Numbers





# PLC Fiddle Demo

[https://github.com/brienc23/Defcon31\\_workshop\\_materials/blob/main/PLC\\_Fiddle\\_demo\\_v1\\_09Aug23\\_default.mp4](https://github.com/brienc23/Defcon31_workshop_materials/blob/main/PLC_Fiddle_demo_v1_09Aug23_default.mp4)

# thiagoralfes/ **OpenPLC\_v3**

OpenPLC Runtime version 3



19  
Contributors

30  
Issues

795  
Stars

353  
Forks



# thiagoralfes/ **OpenPLC\_Editor**

OpenPLC Editor - IDE capable of creating programs  
for the OpenPLC Runtime



14  
Contributors

29  
Issues

275  
Stars

147  
Forks



## 2. OpenPLC

# OpenPLC Project Overview

[https://github.com/thiagoralves/OpenPLC\\_v3](https://github.com/thiagoralves/OpenPLC_v3)

"OpenPLC is an open-source **Programmable Logic Controller** that is based on easy to use software. Our focus is to provide a low cost industrial solution for automation and research. OpenPLC has been used in **many research papers** as a framework for industrial cyber security research, given that it is the only controller to provide the entire source code."

The OpenPLC Project consists of two sub-projects:

1. **Runtime**
2. **Programming editor**

<https://openplcproject.com/docs/openplc-overview/>

"It is the first fully-functional standardized open-source PLC, both in software and in hardware. The OpenPLC project was created in accordance with the IEC 61131-3 standard, which defines the basic software architecture and programming languages for PLCs."

# History and Documentation

Developed by a group of Brazilian ICS security researchers in 2014

Undergone significant updates

Open Source = Free (as in speech [and beer])

**OpenPLC: An Open Source Alternative to Automation**

Thiago Rodrigues Alves, Mario Buratto, Flávio Maurício de Souza, Thelma Virginia Rodrigues  
Departamento de Engenharia Elétrica e de Telecomunicações  
PUC Minas  
Belo Horizonte, Brazil  
thiagorales@gmail.com

**Abstract**  
Companies are always looking for ways to increase production. The elevated consumerism pushes factories to produce more in less time. Industry automation came as the solution to increase quality, production and decrease costs. Since the early 70s, PLC (Programmable Logic Controller) has dominated industrial automation by replacing the relay logic circuits. However, due to its high costs, there are many places in the world where automation is still inaccessible. This paper describes the creation of a low-cost open source PLC, comparable to those already used in industry automation, with the same functionality and sufficient expansion capabilities. Our goal with this project is to create the first fully functional standardized open source PLC. We believe that, with enough help from the open source community, it will become a low cost solution to speed up development and industrial production in less developed countries.

**Keywords**—PLC, OpenPLC, Automation, MODBUS, Open source

**I. INTRODUCTION**  
In early 60s, industrial automation was usually composed of electromechanical parts like relays, cam timers and drum

is made with inexpensive components to lower its costs, opening doors to automation where it wasn't ever possible before.

**II. THE PLC ARCHITECTURE**  
The PLC, being a digital controller, shares common terms with typical PCs, like CPU, memory, bus and expansion. But there are two aspects of the PLC that differentiate them from standard computers. The first one is that its hardware must be sturdy enough to survive a rugged industrial atmosphere. The second is that its software must be real time.

*A. Hardware*  
With the exception of "Brick PLCs" that are not modular, the hardware of a usual PLC can be divided into five basic components:

- Rack
- Power Supply
- CPU [Central Processing Unit]
- Inputs
- Outputs

Like a human spine, the rack has a backplane at the rear allowing communication between every PLC module. The

<https://ieeexplore.ieee.org/document/6970342>

Version of Record: <https://www.sciencedirect.com/science/article/pii/S0167404818305388>  
Manuscript\_3ae7d3c7e934c1529c54693eb17d6bd1

## OpenPLC: An IEC 61131-3 Compliant Open Source Industrial Controller for Cyber Security Research

Thiago Alves  
Electrical and Computer Engineering  
The University of Alabama in Huntsville  
USA  
thiago.alves@uah.edu

Thomas Morris  
Electrical and Computer Engineering  
The University of Alabama in Huntsville  
USA  
tommy.morris@uah.edu

### ABSTRACT

The last years have seen multiple instances of cyber attacks that have been successful in advancing the normal operation of SCADA systems and PLCs. To counter these attacks, researchers have put their efforts in finding defense mechanisms that can protect the network and the PLC. However, since vendors don't make available information about the hardware and firmware of their devices, it becomes challenging to perform cyber security research for PLCs. This work proposes the development of an open source PLC, compliant with the IEC 61131-3 international standard. A description of the hardware architecture, development environment, supported SCADA protocols and an additional HMI editor package is presented. Additionally, this work presents a methodology for validating PLC logic execution, performance, and SCADA connectivity, and also compares the behavior of OpenPLC with four other popular commercial PLCs when under a Modbus injection attack, to support the claim that OpenPLC is a valid platform for PLC cyber security research.

**Keywords**  
PLC, SCADA, ICS, Cybersecurity, Vulnerability Analysis, OpenPLC, MODBUS, HMI.

**I. INTRODUCTION**  
The closed industrial environment hides key information related to the development of technology behind patents, copyrights, and

<https://www.sciencedirect.com/science/article/abs/pii/S0167404818305388>

# OpenPLC Runtime v3 Target Platforms

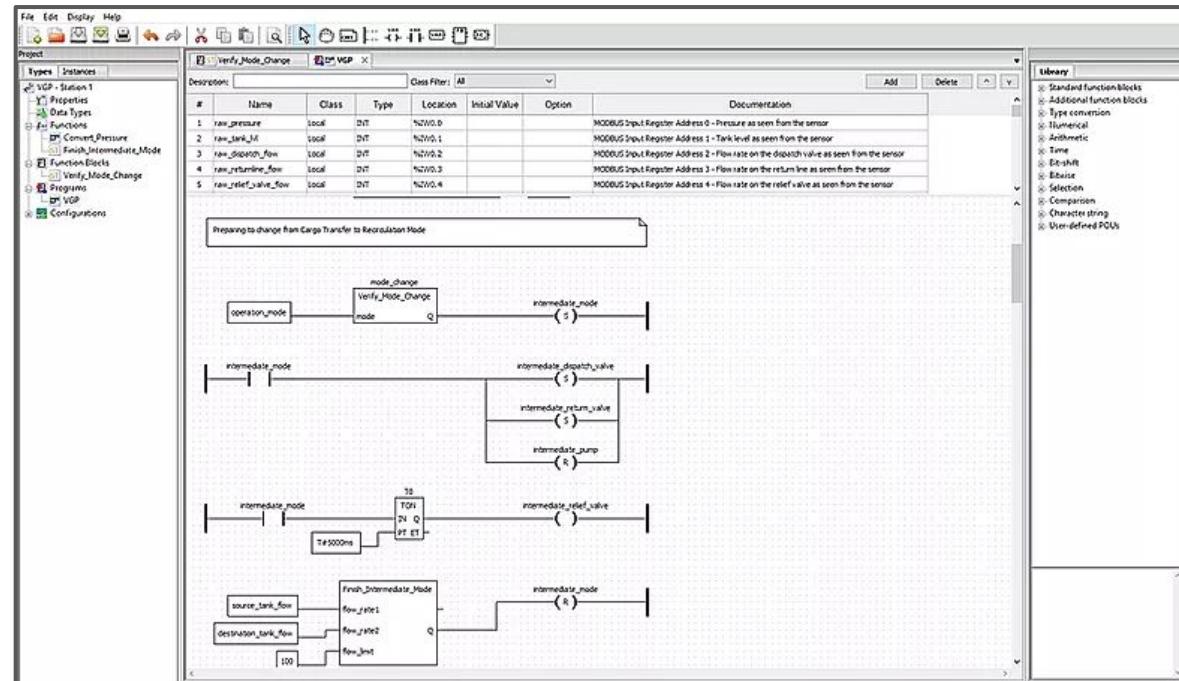
- Arduino Uno / Nano / Leonardo / Micro
- Arduino Mega / Due
- Arduino Nano Every / IoT / BLE
- Arduino RB2040 Connect
- Arduino Mkr / Zero / WiFi
- Arduino Pro (Machine Control and EDGE)
- Controllino Maxi / Automation / Mega / Mini
- Productivity Open P1AM
- ESP8266 (nodemcu)
- ESP32
- **Raspberry Pi 2 / 3 / 4**
- PiXtend
- UniPi Industrial Platform
- Neuron PLC
- FreeWave Zumlink
- FreeWave ZumIQ
- Windows (generic target as a soft-PLC)
- Linux (generic target as a soft-PLC)

# OpenPLC Editor

<https://openplcproject.com/docs/3-1-openplc-editor-overview/>

v3.0+ uses the  
python-based Beremiz IDE

Writes programs according  
to IEC-61131-3 and  
conforms to PLCopen XML

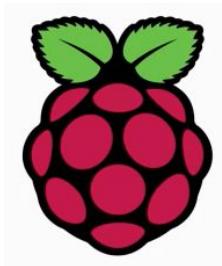
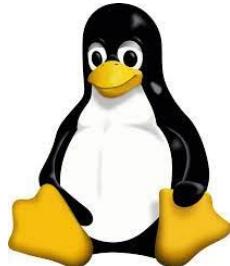


# OpenPLC Runtime

Runs PLC programs created on the OpenPLC Editor.

The main runtime has a built-in webserver that allows configuration & control.

A soft PLC can be run from Windows (via Cygwin) or Linux



<https://openplcproject.com/docs/2-1-openplc-runtime-overview/>

Dashboard

Status: **Stopped**

Program: Blank Program

Description: Dummy empty program

File: blank\_program.st

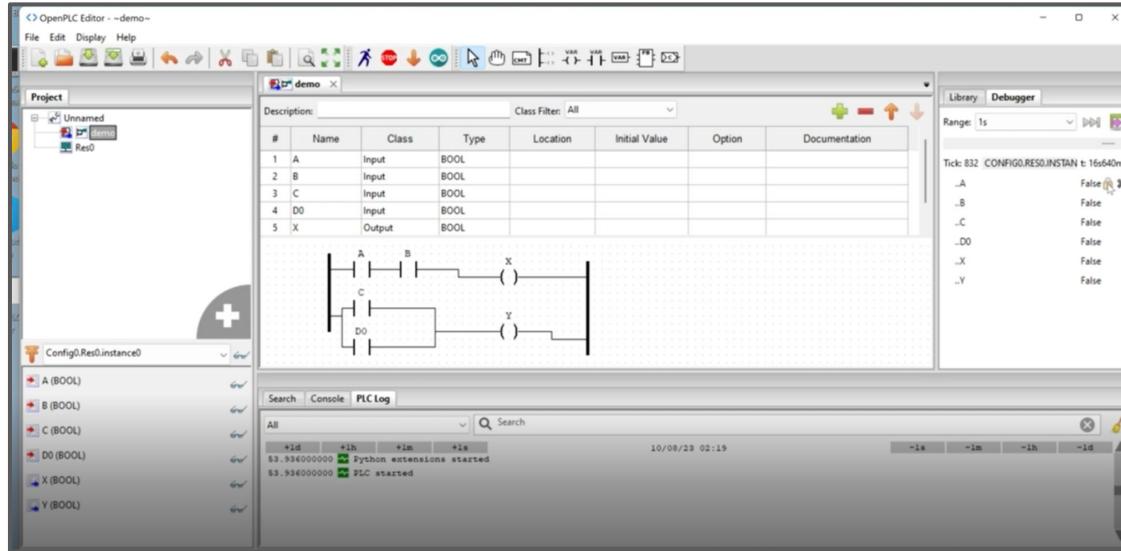
Runtime: N/A

Start PLC

Monitoring

Refresh Rate (ms): 100

Point Name	Type	Location	Forced	Value

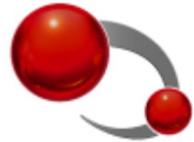


# OpenPLC Demo

[https://github.com/brienc23/Defcon31\\_workshop\\_materials/blob/main/OpenPLC\\_demo\\_v1\\_09Aug23\\_default\\_570p.mp4](https://github.com/brienc23/Defcon31_workshop_materials/blob/main/OpenPLC_demo_v1_09Aug23_default_570p.mp4)

**Connected Components Workbench**

release 12.00



**Rockwell  
Automation**

© Allen-Bradley • Rockwell Software



**Rockwell  
Automation**



**Allen-Bradley**

by ROCKWELL AUTOMATION

### 3. Rockwell/Allen-Bradley CCW

# Connected Components Workbench (CCW) Overview

Standard Edition is Free (as in beer)

Program and run their newest low-cost line of PLC [micro8X0](#)

You need to sign up for an account on the [Rockwell Automation](#) website

**VERSIONS** ?

CHANGE PRODUCT ■ LEGEND

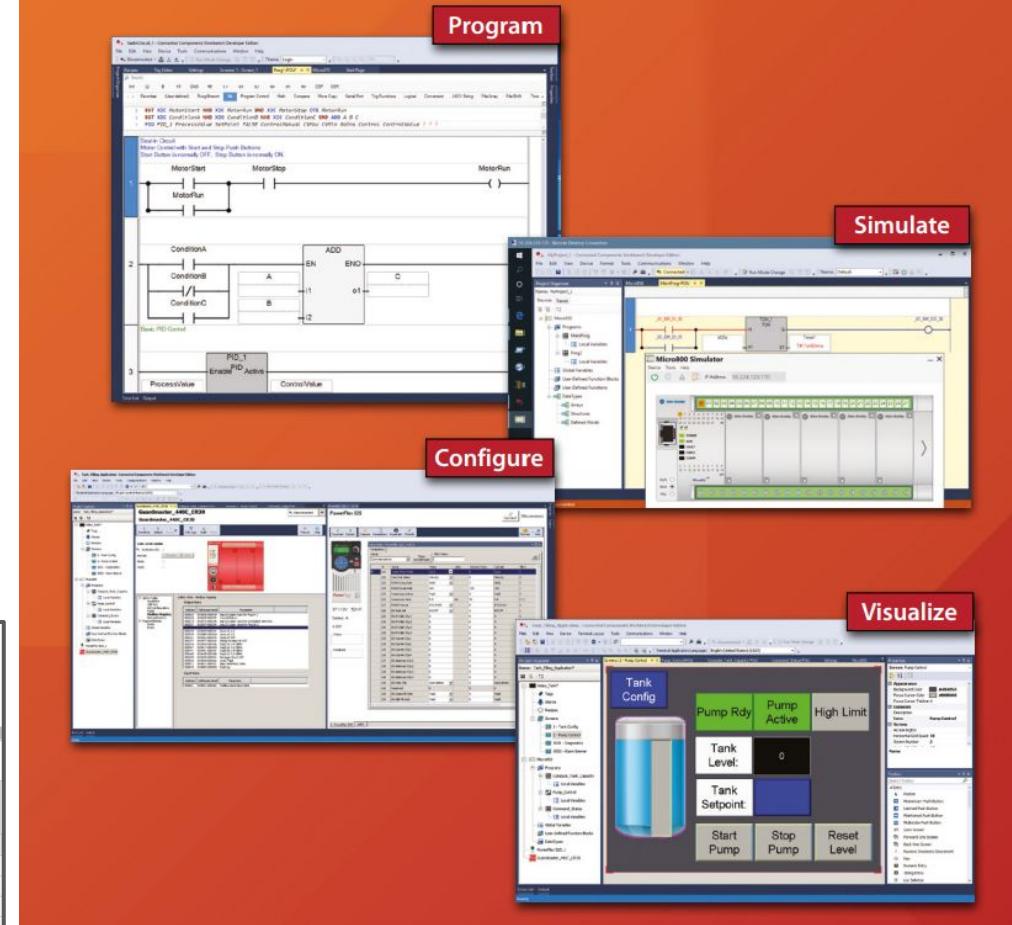
Connected Components Workbench  
Connected Components Workbench (CCW) Standard Edition (free) single software to program, simulate, configure, and visualize.

Version	21.01.00	21.00.00	20.01.00	13.00.00	12.00.00	11.00.00	10.01.00
Downloads Information							

**Operating Systems**

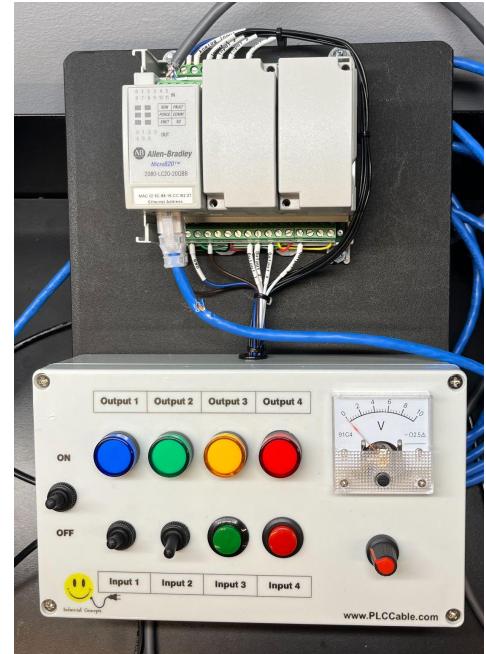
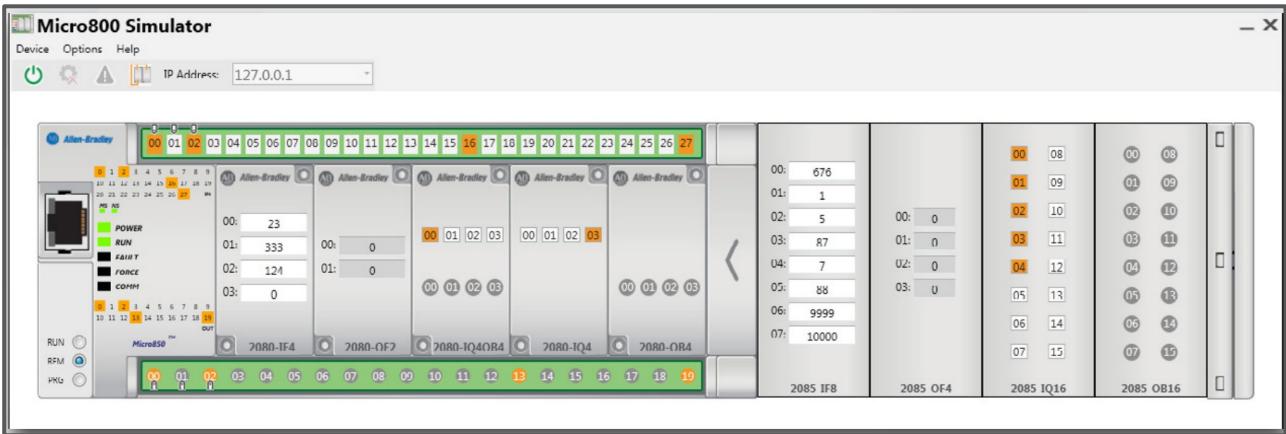
- Windows 10
- Windows 11
- Windows 2003
- Windows 2003 R2
- Windows 2008

<https://compatibility.rockwellautomation.com/Pages/ProductReplacement.aspx?crumb=101&restore=1&vid=62094>



[https://literature.rockwellautomation.com/idc/groups/literature/documents/pp/9328-pp001\\_-en-p.pdf](https://literature.rockwellautomation.com/idc/groups/literature/documents/pp/9328-pp001_-en-p.pdf)

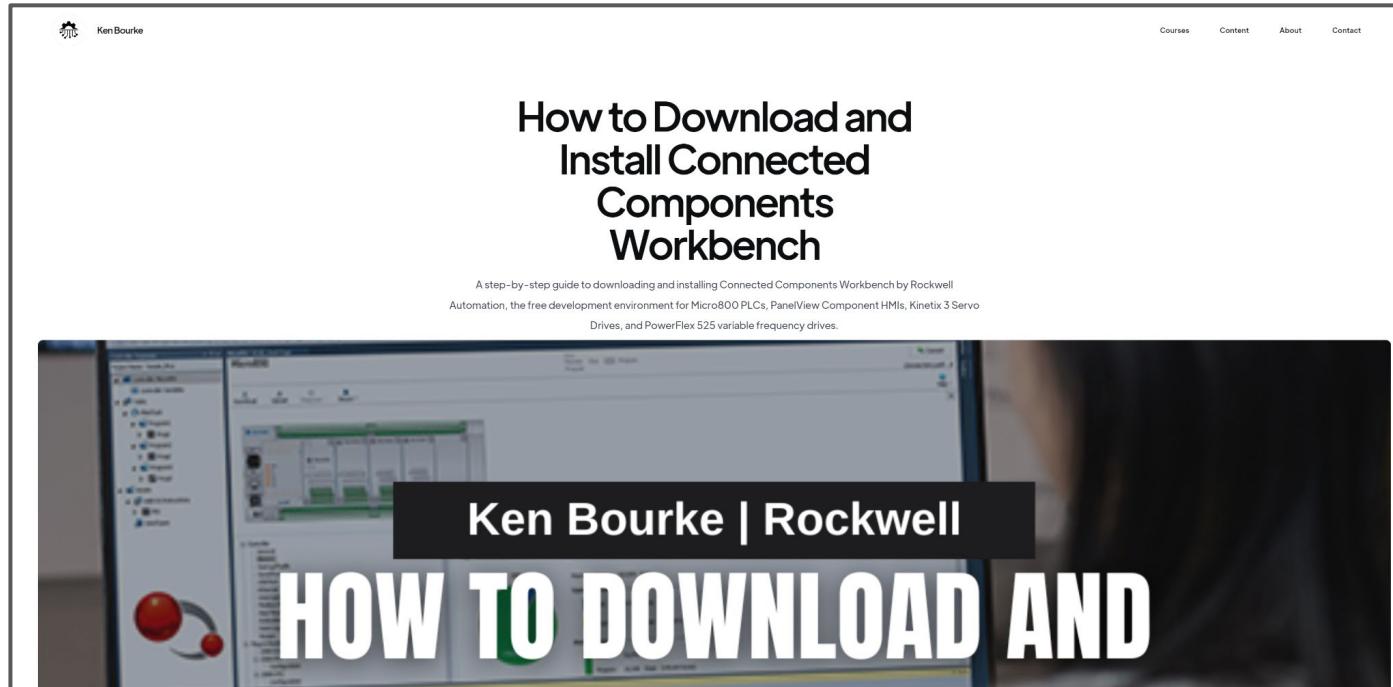
# Built-in Simulator -or- Real-World Hardware



<https://www.plccable.com/allen-bradley-micro820-analog-ccw-plc-trainer-micro800-training-kit/>

# How to Install

<https://www.kenbourke.me/posts/how-to-download-and-install-connected-components-workbench>



# Plenty of other Online Resources



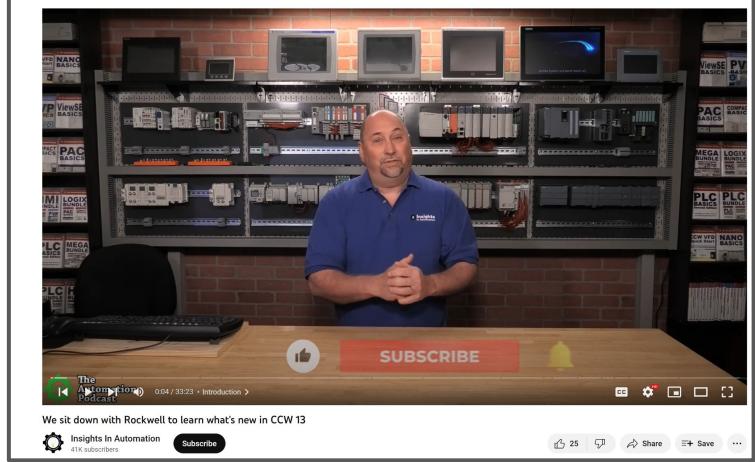
## Allen Bradley Micro800 Connected Components Workbench CCW

Tim Wilborne

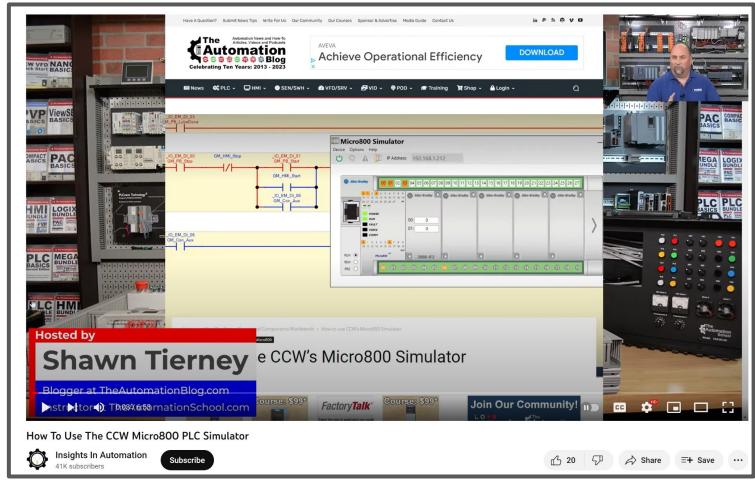
53 videos 54,980 views Last updated on Feb 2...

In this lesson series we show you how to download your software, go through the PLC programming instru ...More

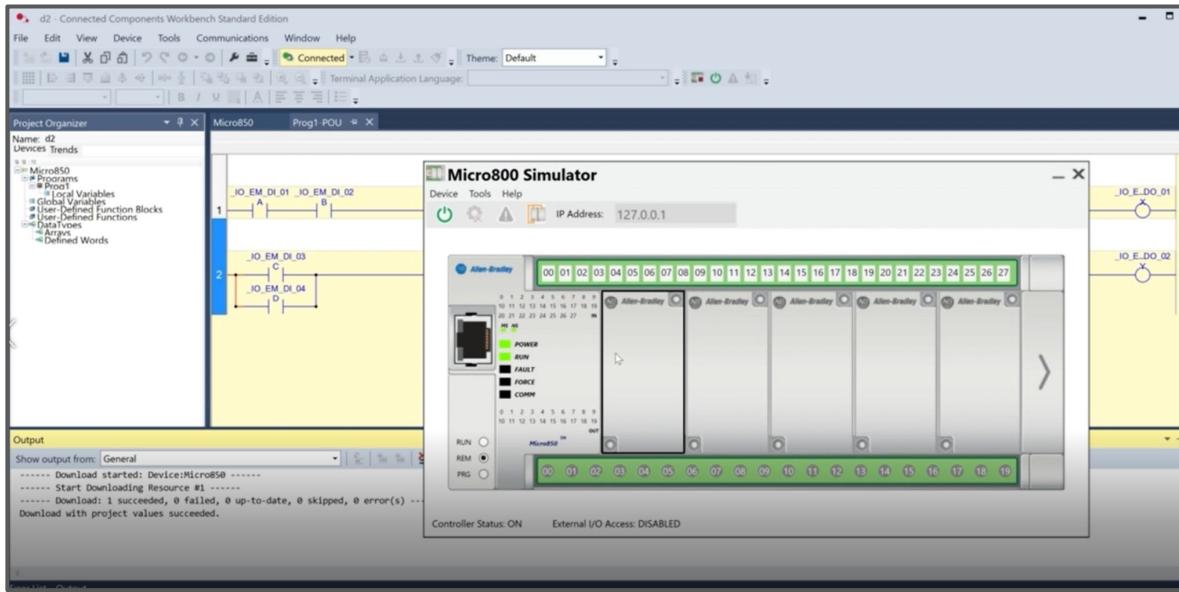
<https://www.youtube.com/playlist?list=PLUi5cdVq3wTCjGF-QqwyVLDejTLhHrvAT>



<https://www.youtube.com/watch?v=j0uWjQD0PIY>



<https://www.youtube.com/watch?v=s6dRL-8meFs> 29

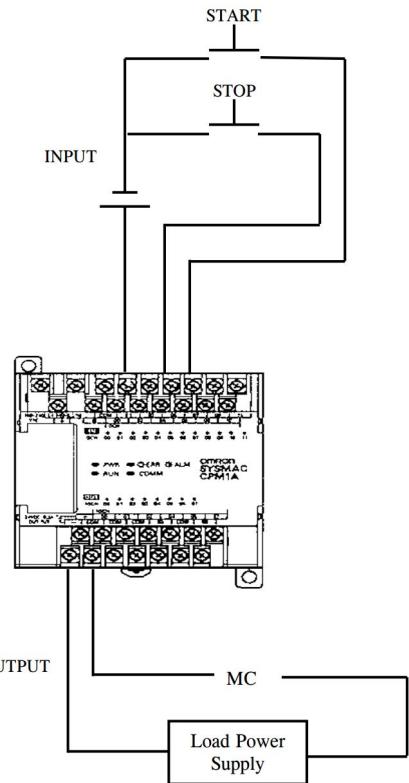


# CCW Demo

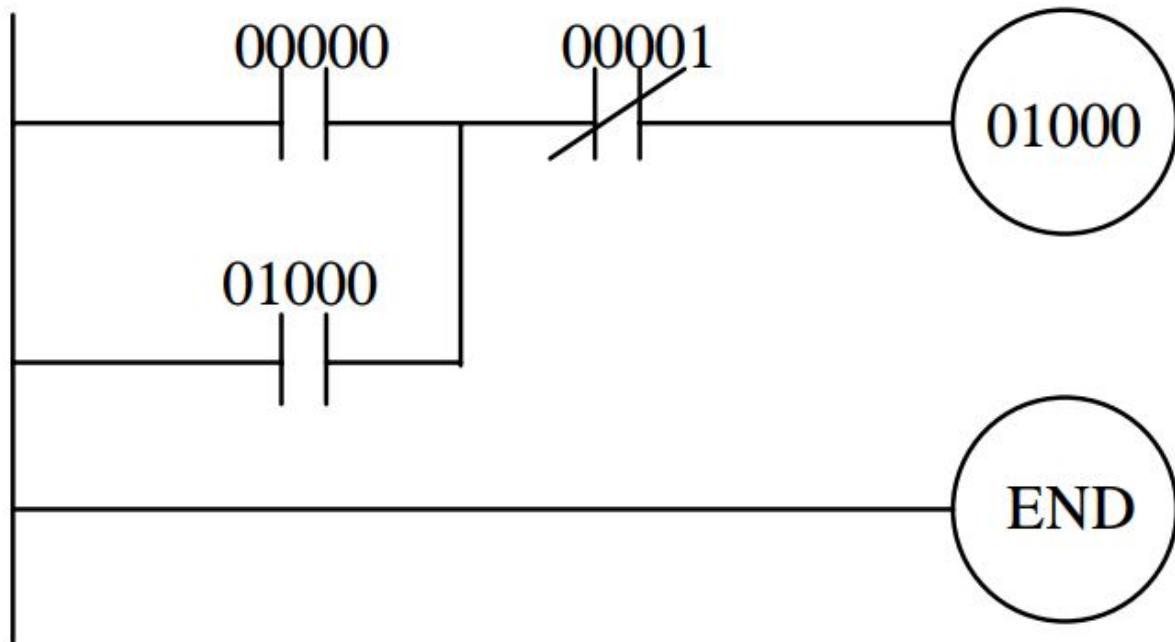
[https://github.com/brienc23/Defcon31\\_workshop\\_materials/blob/main/CCW\\_Demo\\_v1\\_09Aug23\\_570p.mp4](https://github.com/brienc23/Defcon31_workshop_materials/blob/main/CCW_Demo_v1_09Aug23_570p.mp4)

# Example Circuits

# Latch / Self Holding Circuit



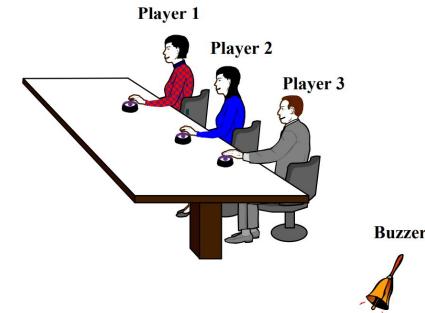
## Ladder Diagram



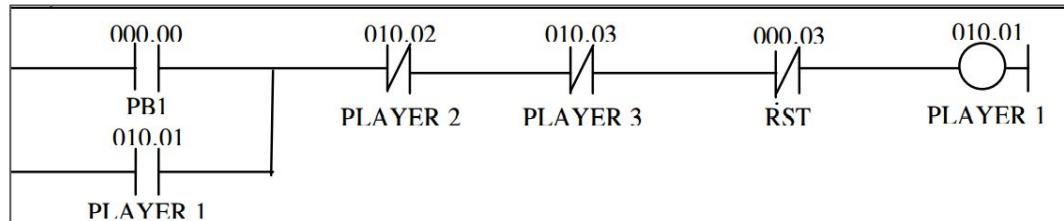
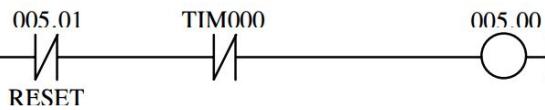
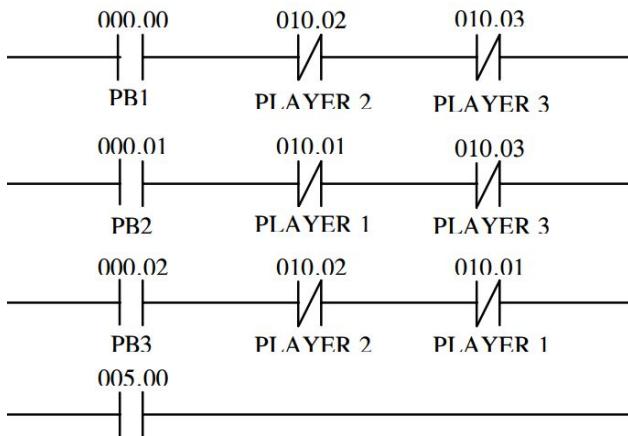
# Example - Quiz Show

The game buzzer control requirement:

1. After the Host has finished with question.
2. The 3 players will press the switch in front of them to fight to be first to answer the question.
3. The buzzer will sound for 10 sec after any one of the players has touched the switch.
4. The light indicator in front of each player will light-up and only reset by the Host switch.



Input	Device	Output	Device
00000	PB1	01000	Buzzer
00001	PB2	01001	Player 1 light
00002	PB3	01002	Player 2 light
00003	RST (reset)	01003	Player 3 light



# Backup Slides

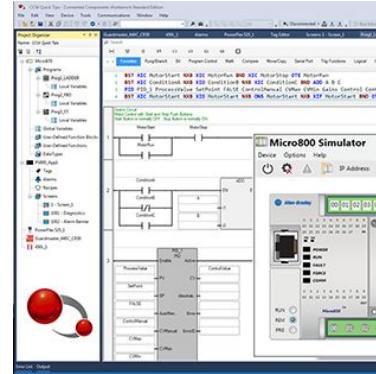
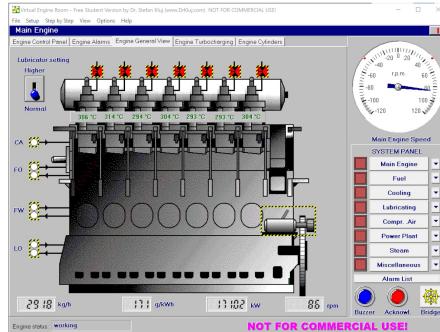
# MICS Course Offering

Spring 2023, SY486K Maritime Industrial Control Systems Cybersecurity

Three-credit technical elective

Lectures, Labs, Student Presentations, and YP Project

Topics included: Maritime Systems, PLC, Ladder Logic, Modbus, CIP, Attacking ICS



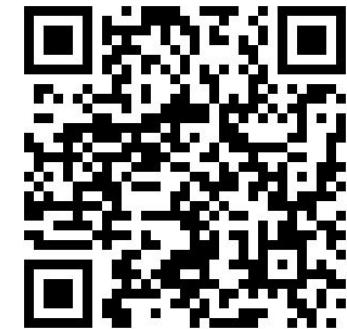
[https://github.com/brienc23/MICS\\_Course\\_Materials](https://github.com/brienc23/MICS_Course_Materials)



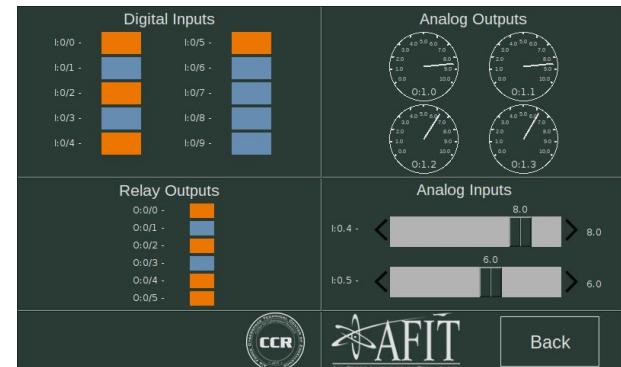
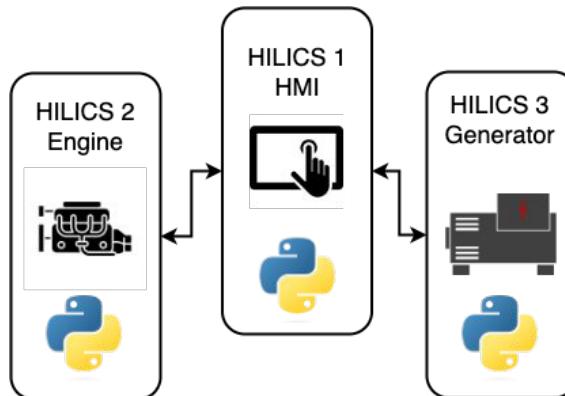
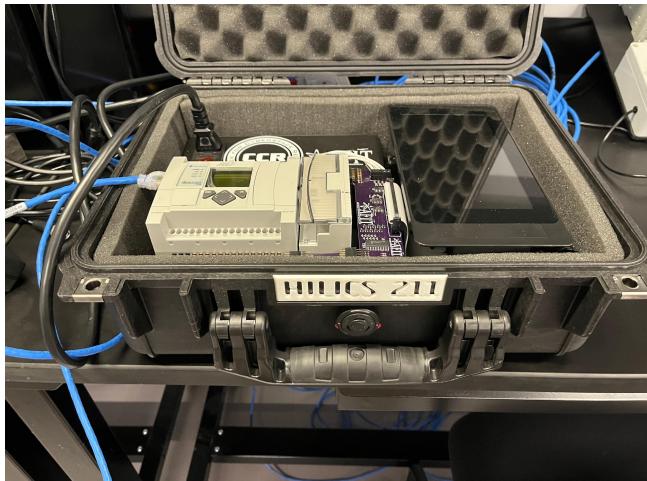
# Hardware-in-the-Loop ICS (HILICS)



Produced by Air Force Institute of Technology (AFIT)  
AB microLogix 1100+RasPi (with touchscreen HMI) in a Pelican case



[https://github.com/  
sdunlap-afit/hilics](https://github.com/sdunlap-afit/hilics)



# The Scanning Process

[https://my.ece.utah.edu/~ece3510/  
Ladder%20Logic%20Fundamentals%20PLC%20tutorial.pdf](https://my.ece.utah.edu/~ece3510/Ladder%20Logic%20Fundamentals%20PLC%20tutorial.pdf)

- The scan sequence can be broken into two functional parts:
  - The **Program Scan**
    - Scan the ladder program
  - The **I/O Update Scan**
    - Write outputs, Read inputs

## ■ The Program Scan:

- For each rung executed, the PLC processor will:
  - Examine the status of the input image table bits,
  - Solve the ladder logic in order to determine logical continuity (is the rung true?),
  - Update the appropriate output image table bits, if necessary.

**Note:** The output will not actually be energized until the I/O update part of the scan.

## ■ The I/O Update Scan:

- Copy the output image table status to the ALL of the output terminals (discrete output circuits)
  - Power is applied to the output device if it's output image table bit has been previously set to a 1.
- Copy the status of ALL of the input terminals to the input image table
  - If an input is active (i.e., there is electrical continuity), the corresponding bit in the input image table will be set to a 1.