CEC Continuing Education Center

DesignNews

Embedded Controls Development with OpenPLC

DAY 2 : OpenPLC Ladder Diagram Basics

Sponsored by

DigiKey

informa markets

Webinar Logistics

- Turn on your system sound to hear the streaming presentation.
- If you have technical problems, click “Help” or submit a question asking for assistance.
- Participate in ‘Attendee Chat’ by maximizing the chat widget in your dock.

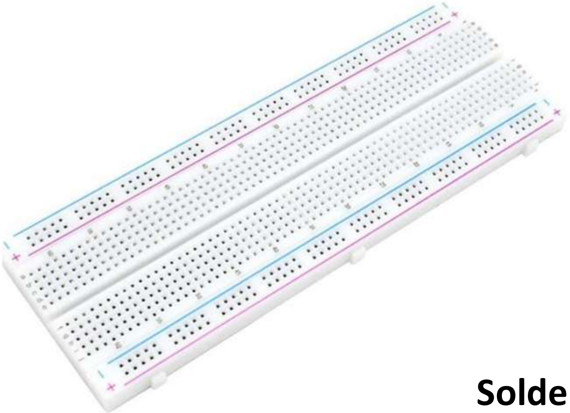
Dr. Don Wilcher

Visit ‘Lecturer Profile’ in your console for more details.

Course Kit and Materials

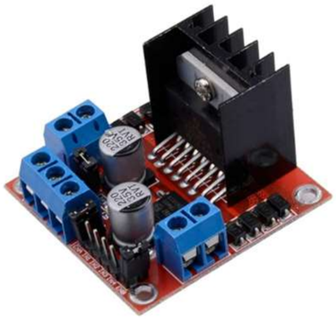
ESP32 WROOM32D DEVKITC

Solderless Breadboard x2

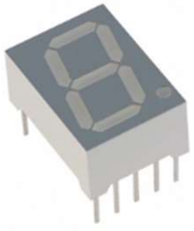


Adafruit Parts Pal Kit

L298N Motor Drive Controller



7 Segment LED Display, Common Cathode



Solderless Breadboard Power Supply Module with 9V Battery Clip Power Cable

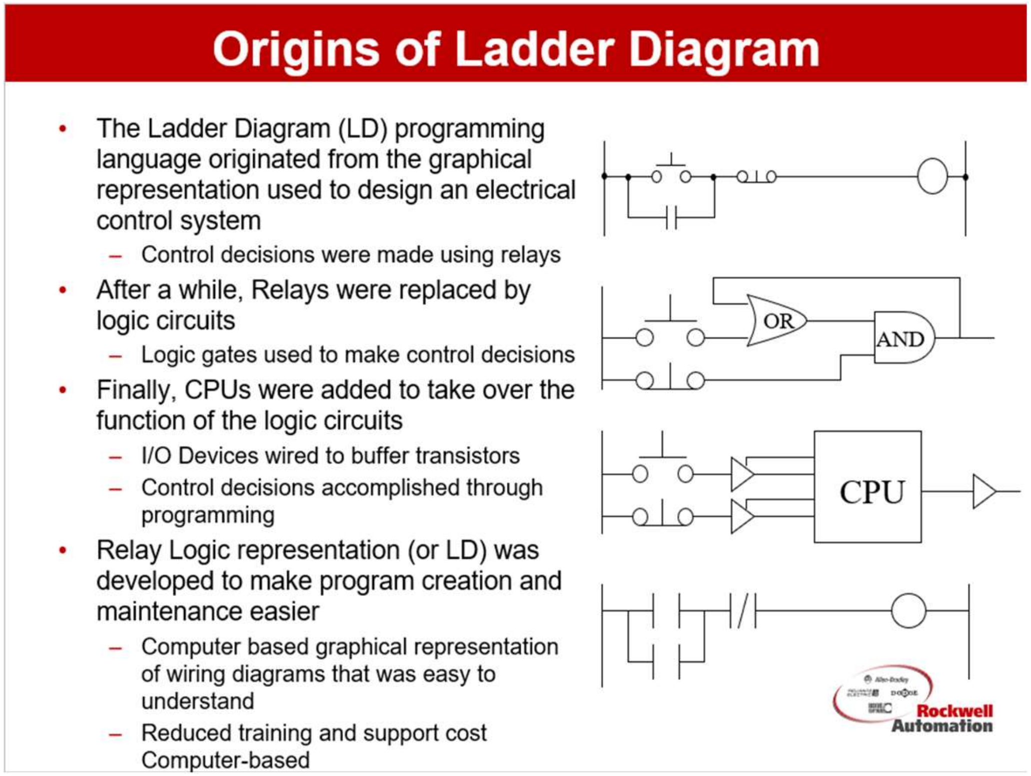
Agenda:

- OpenPLC Ladder Diagram Basics
  - a) Origins of the Ladder Diagram
  - b) IEC 61131-3 Specification Programming Model
    - i. Program Block Subclass
    - ii. Contacts, Coils, and Lines
- OpenPLC Simulator
  - a) Setup
  - b) Force ON/Force OFF tools
- Lab: Digital Logic Control Application

Research Perspective

“Embedded electronics is a subfield of electronics that can unite the power of programming with the power of electronics” (Zemmouri et al., 2023).

OpenPLC Ladder Diagram Basics



Origins of Ladder Diagram

- The Ladder Diagram (LD) programming language originated from the graphical representation used to design an electrical control system
  - Control decisions were made using relays
- After a while, Relays were replaced by logic circuits
  - Logic gates used to make control decisions
- Finally, CPUs were added to take over the function of the logic circuits
  - I/O Devices wired to buffer transistors
  - Control decisions accomplished through programming
- Relay Logic representation (or LD) was developed to make program creation and maintenance easier
  - Computer based graphical representation of wiring diagrams that was easy to understand
  - Reduced training and support cost Computer-based

Illustration courtesy of Rockwell

Question 1

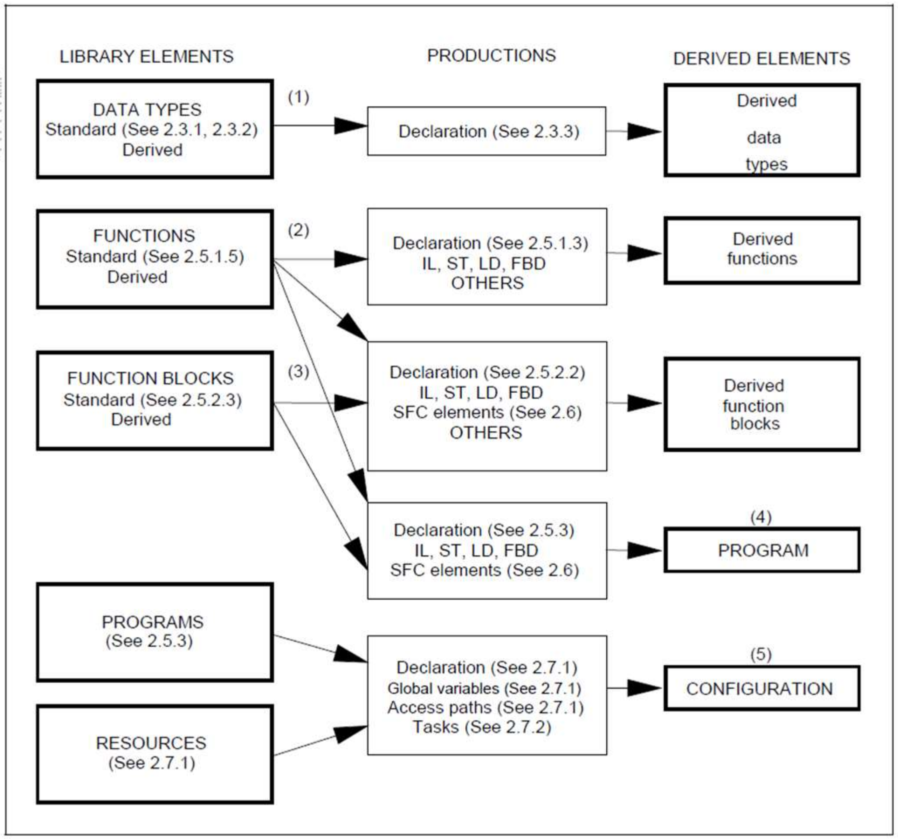
In reviewing slide 7, relays were replaced by_____.

a) CPU
b) LD
c) logic circuits
d) None of the above

OpenPLC Ladder Diagram Basics. . .

IEC 61131-3 Programming Model

Illustration courtesy of IEC 6113-3 Standard, Second Edition 2003.



OpenPLC Ladder Diagram Basics. . .

The subclass Program block is the main Declaration element of interest in programming the ESP32 microcontroller using OpenPLC.

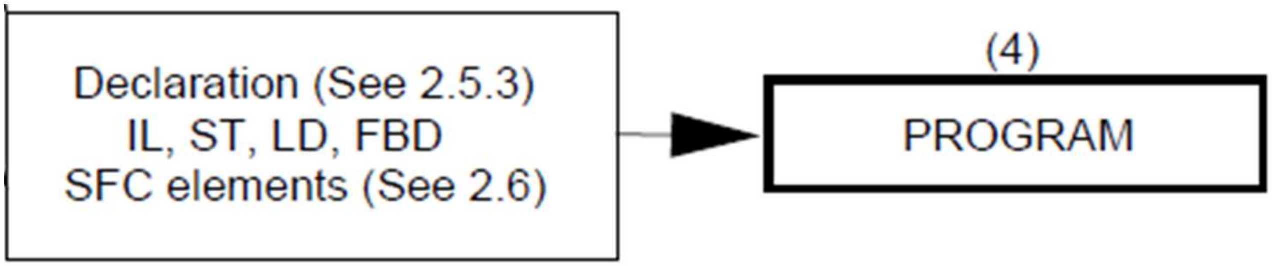


Illustration courtesy of IEC 6113-3 Standard, Second Edition 2003.

OpenPLC Ladder Diagram Basics. . .

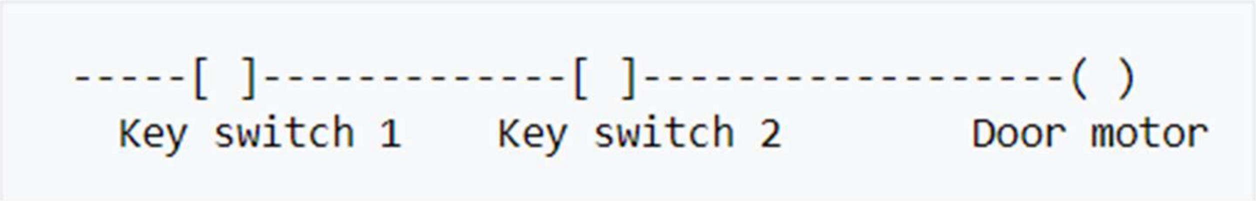
- A program is defined as a logical assembly of all the programming language elements and constructs necessary for the intended (IEC 61131-3, p.83, 2003):
  a) signal processing for processes
  b) control of a machine
- A program allows such signal processing and machine control to occur within a programmable controller system.
- A network of programming elements defines the program's construction.

OpenPLC Ladder Diagram Basics…

- A network is defined as a maximal set of interconnected elements, excluding the left and right rails (IEC 61131-3, p.135, 2003).
- Graphical languages are used to represent the flow of a conceptual quantity through one or more networks representing a control plan (IEC 6113-3, p. 135, 2003).
- There are three graphical language flows used in a control plan.
  a) Power – aligns with relay ladder diagrams (LD)
  b) Signal – aligns with function blocks (FB)
  c) Activity – aligns with sequential function charts (SFC)

OpenPLC Ladder Diagram Basics. . .

The Power flow graphical language (LD) is primarily used in building OpenPLC applications.



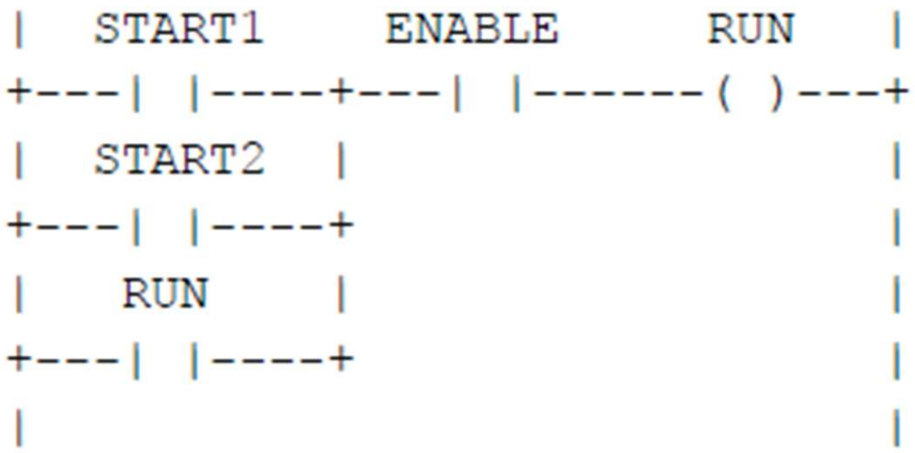
```
-----[ ]-------------[ ]------------------( )
  Key switch 1    Key switch 2        Door motor
```

An Example LD Power Flow graphical language

Illustration courtesy of https://en.wikipedia.org/wiki/Ladder_logic

OpenPLC Ladder Diagram Basics. . .

IEC 61131-3 of an LD (Power Flow) program



```
|   START1      ENABLE       RUN    |
+---| |----+---| |-------( )---+
|   START2  |                        |
+---| |----+                        |
|     RUN     |                        |
+---| |----+                        |
|                                       |
```

Illustration courtesy of IEC 6113-3 Standard, Second Edition 2003.

Question 2

In reviewing slide 11, what logic function is being performed by the LD program?

a) AND-OR
b) EXOR
c) AND
d) NAND

OpenPLC Ladder Diagram Basics. . .

A LD program enables the programmable controller to test and modify data. The data characteristics are listed below.

a) Standardized Symbols

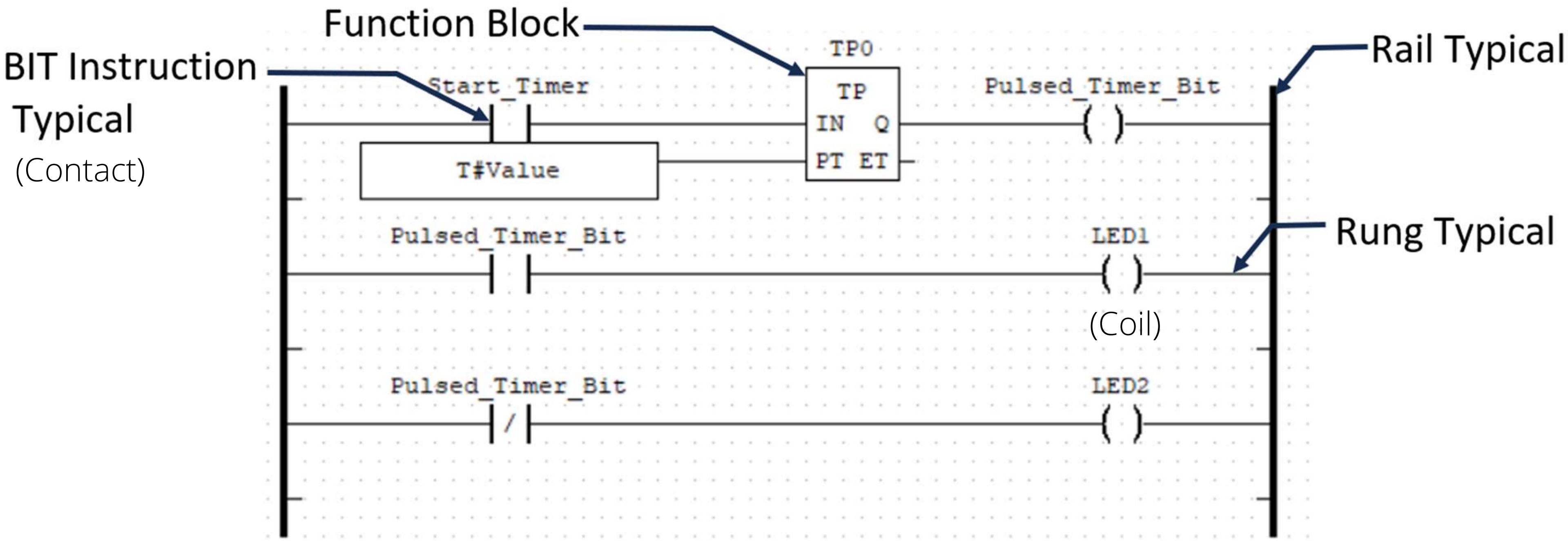
b) Standardized Symbols are the programmable elements in a network.

c) The network is arranged like a rung of a relay ladder logic diagram.

d) The networks are bounded on the left and right by power rails.

OpenPLC Ladder Diagram Basics. . .

Anatomy of a Ladder Diagram

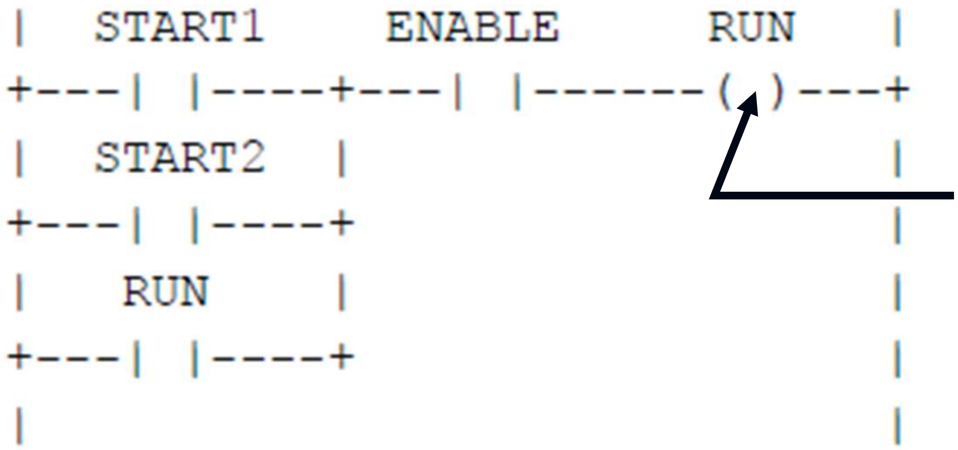


OpenPLC Ladder Diagram Basics…

- A contact is an element that imports a state to the horizontal link on its right side, which is equal to the Boolean function of the state of the horizontal link on its left side.
  - a) The right and left sides of the horizontal link are the power rail.
  - b) The horizontal link is the ladder diagram rung.
- A coil copies the state of the link on its left to the link on its right without modification and stores an appropriate function of the state or transition of the left link into the associated Boolean variables (IEC 61131-3, p.142, 2003).

OpenPLC Ladder Diagram Basics. . .

Identification of the coil on an LD.



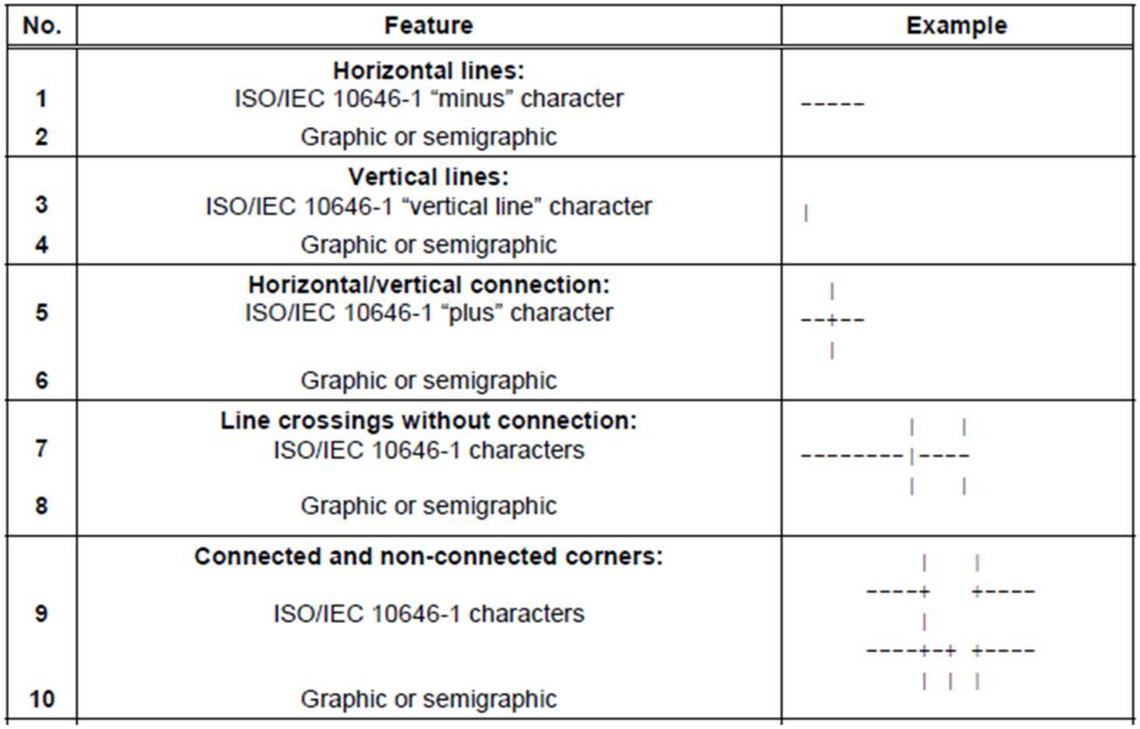
```
|   START1     ENABLE       RUN    |
+---| |----+---| |-------( )---+
|   START2   |                     |
+---| |----+                       |
|    RUN     |                     |
+---| |----+                       |
|                                  |
```

Coil

Illustration courtesy of IEC 6113-3 Standard, Second Edition 2003.

OpenPLC Ladder Diagram Basics. . .

Representation of Lines:
Lines can be extended by the use of connectors (IEC 61131-3, p.135, 2003)

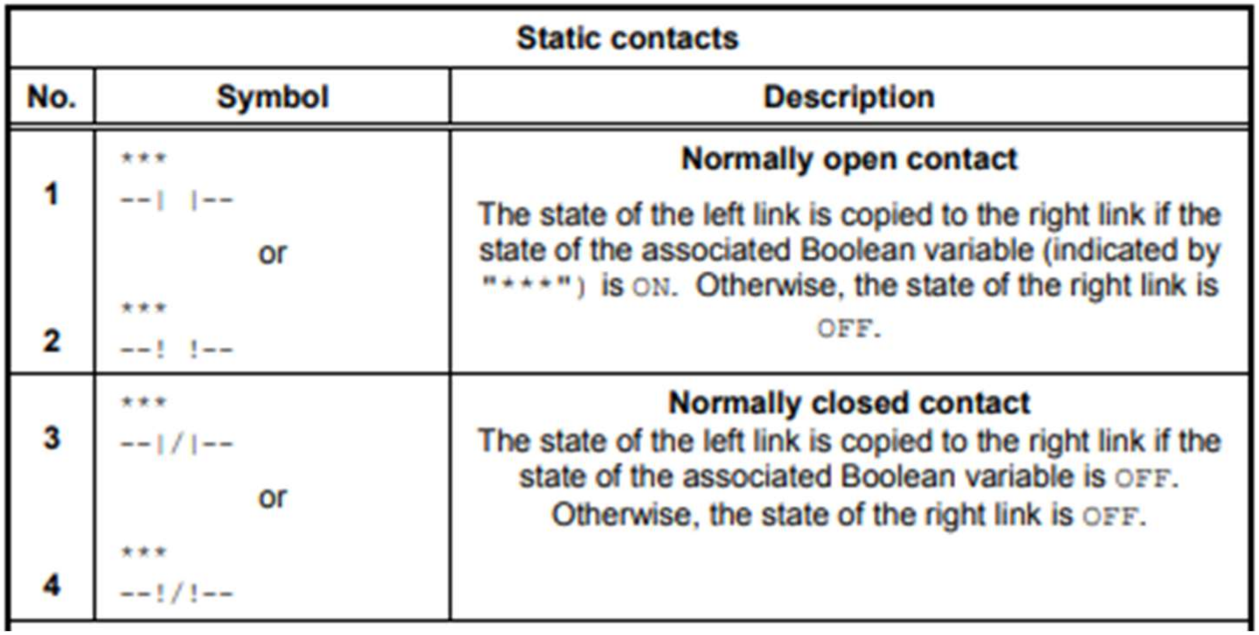


<table>
<tr><th>No.</th><th>Feature</th><th>Example</th></tr>
<tr><td>
1
2</td><td>Horizontal lines:
ISO/IEC 10646-1 “minus” character
Graphic or semigraphic</td><td>-----</td></tr>
<tr><td>
3
4</td><td>Vertical lines:
ISO/IEC 10646-1 “vertical line” character
Graphic or semigraphic</td><td>|</td></tr>
<tr><td>
5
6</td><td>Horizontal/vertical connection:
ISO/IEC 10646-1 “plus” character
Graphic or semigraphic</td><td>|
--+--
|</td></tr>
<tr><td>
7
8</td><td>Line crossings without connection:
ISO/IEC 10646-1 characters
Graphic or semigraphic</td><td>| |
--------|----
| |</td></tr>
<tr><td>
9
10</td><td>Connected and non-connected corners:
ISO/IEC 10646-1 characters
Graphic or semigraphic</td><td>| |
----+ +----
|
----+-+ +-----
| | |</td></tr>
</table>

Illustration courtesy of IEC 6113-3 Standard, Second Edition 2003.

OpenPLC Ladder Diagram Basics. . .

Representation of Contacts:
A contact is an element which imparts a state to the horizontal link on its right side (IEC 61131-3, p.140, 2003)



Static contacts		
No.	**Symbol**	**Description**

1	`***`
	`--| |--`
	or
		Normally open contact
		The state of the left link is copied to the right link if the state of the associated Boolean variable (indicated by "***") is ON. Otherwise, the state of the right link is OFF.

2	`***`
	`--! !--`	

3	`***`
	`--|/|--`
	or
		Normally closed contact
		The state of the left link is copied to the right link if the state of the associated Boolean variable is OFF. Otherwise, the state of the right link is OFF.

4	`***` `--!/!--`	

Illustration courtesy of IEC 6113-3 Standard, Second Edition 2003.

Question 3

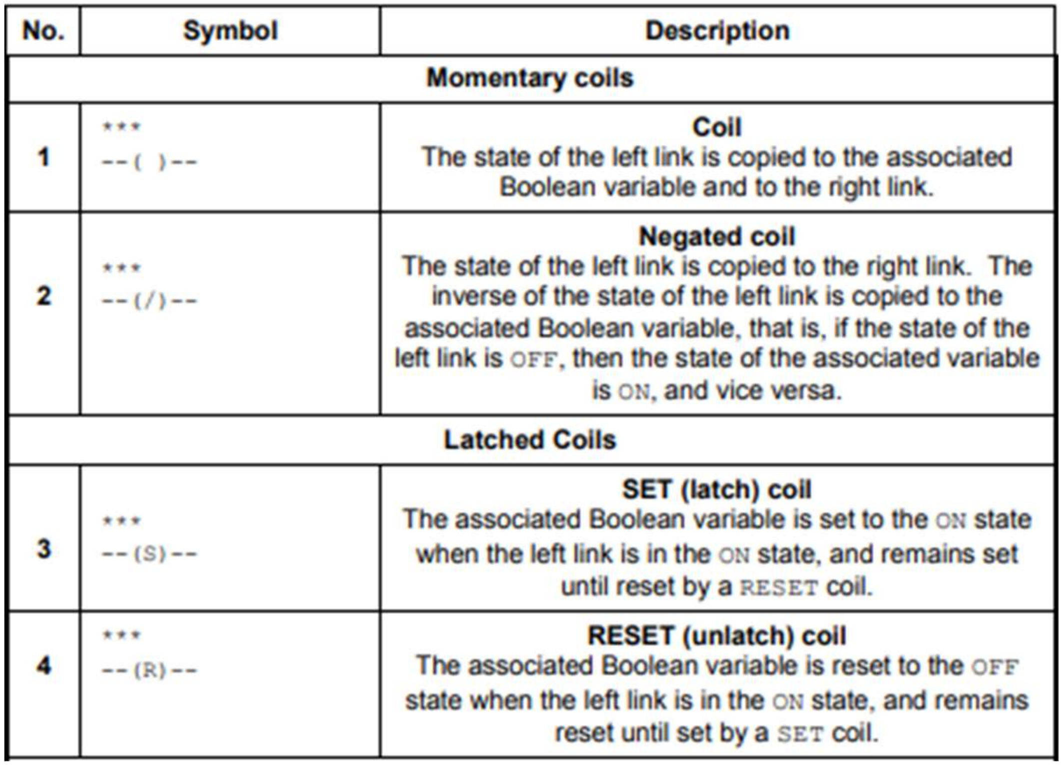
A contact is an element that imparts a state to the horizontal link on its left side.

a) True

b) False

OpenPLC Ladder Diagram Basics. . .

Representation of Coils:
A coil copies the state of the link on its left side (IEC 61131-3, p.140, 2003)



No.	Symbol	Description
Momentary coils		
1	*** --( )--	**Coil** The state of the left link is copied to the associated Boolean variable and to the right link.
2	*** --(/)--	**Negated coil** The state of the left link is copied to the right link. The inverse of the state of the left link is copied to the associated Boolean variable, that is, if the state of the left link is OFF, then the state of the associated variable is ON, and vice versa.
Latched Coils		
3	*** --(S)--	**SET (latch) coil** The associated Boolean variable is set to the ON state when the left link is in the ON state, and remains set until reset by a RESET coil.
4	*** --(R)--	**RESET (unlatch) coil** The associated Boolean variable is reset to the OFF state when the left link is in the ON state, and remains reset until set by a SET coil.

Illustration courtesy of IEC 6113-3 Standard, Second Edition 2003.

OpenPLC Simulator

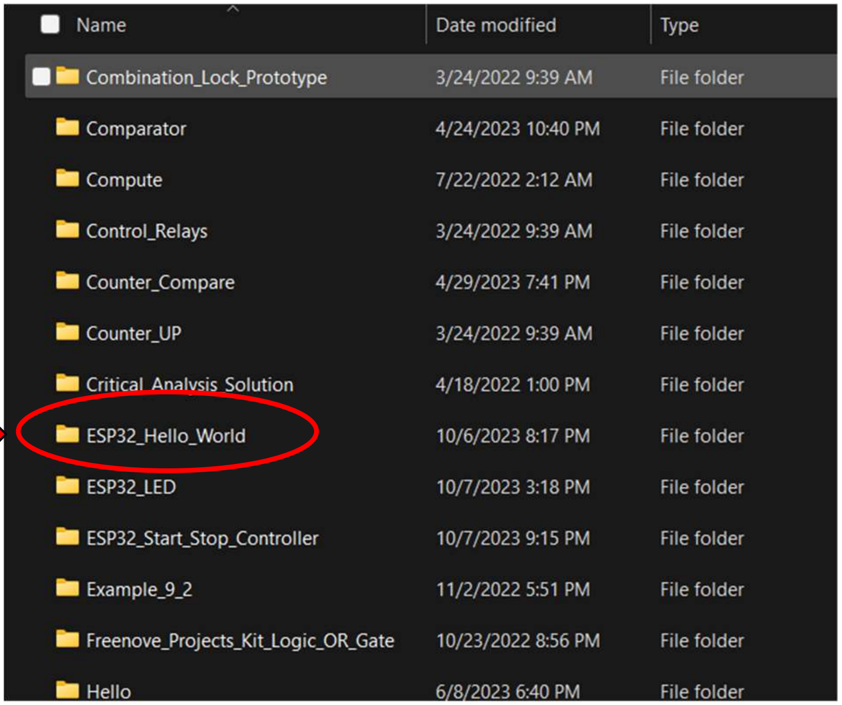
- An embedded application using the ESP32 microcontroller can be tested using a simulator.
- The OpenPLC simulator allows ESP32 microcontroller embedded applications to be tested without the actual hardware.
- Features and functions of the ESP32 microcontroller can easily be tested using the OpenPLC simulator.
- Once the simulated event is completed, the LD program can be downloaded to the ESP32 microcontroller.

OpenPLC Simulator Setup

Before setting up the OpenPLC Simulator, a project folder needs to be created.

Create a Main Projects Folder

Name of Project Folder

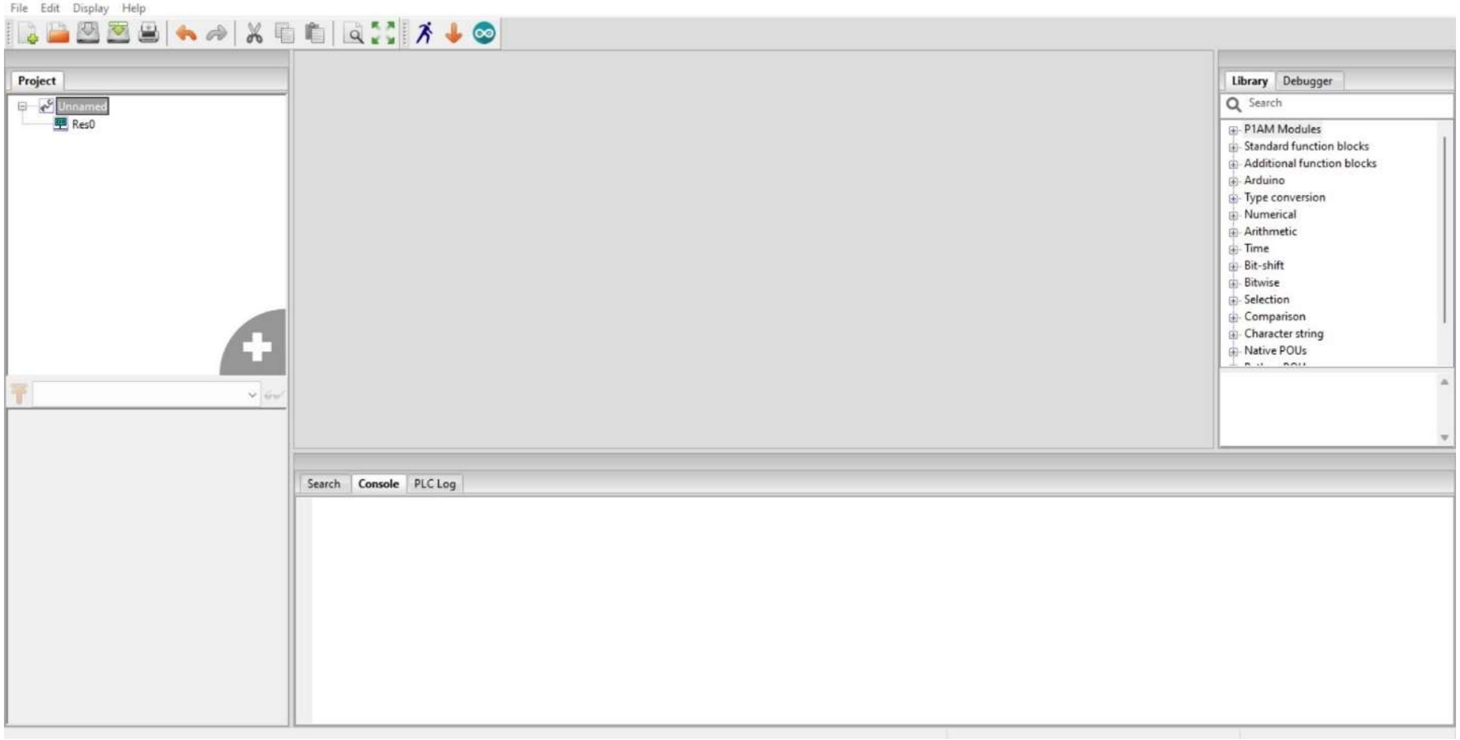


Name	Date modified	Type
Combination_Lock_Prototype	3/24/2022 9:39 AM	File folder
Comparator	4/24/2023 10:40 PM	File folder
Compute	7/22/2022 2:12 AM	File folder
Control_Relays	3/24/2022 9:39 AM	File folder
Counter_Compare	4/29/2023 7:41 PM	File folder
Counter_UP	3/24/2022 9:39 AM	File folder
Critical_Analysis_Solution	4/18/2022 1:00 PM	File folder
ESP32_Hello_World	10/6/2023 8:17 PM	File folder
ESP32_LED	10/7/2023 3:18 PM	File folder
ESP32_Start_Stop_Controller	10/7/2023 9:15 PM	File folder
Example_9_2	11/2/2022 5:51 PM	File folder
Freenove_Projects_Kit_Logic_OR_Gate	10/23/2022 8:56 PM	File folder
Hello	6/8/2023 6:40 PM	File folder

OpenPLC Simulator Setup. . .

Open the OpenPLC editor to start a new LD program.

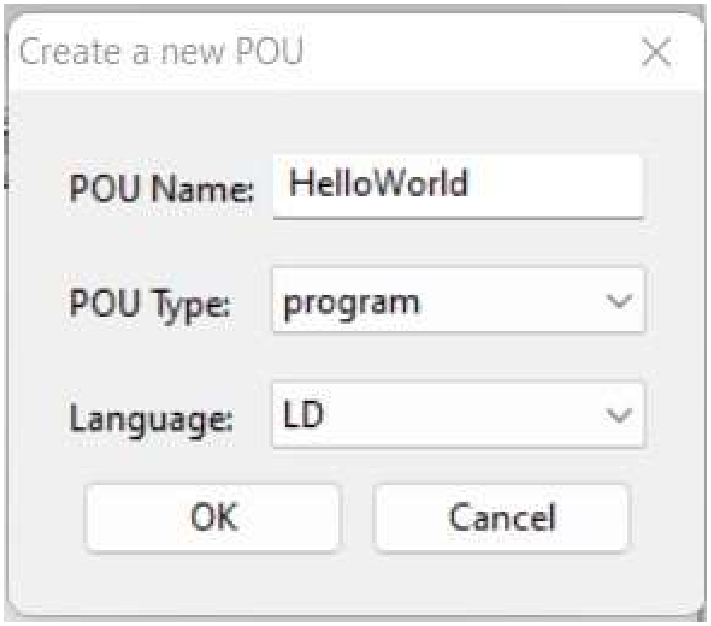
An open OpenPLC editor



OpenPLC Simulator Setup. . .

Create a Program Organizational Unit (POU) (Name/Type). Select LD for programming language.

Creating a New POU



Create a new POU

POU Name: HelloWorld

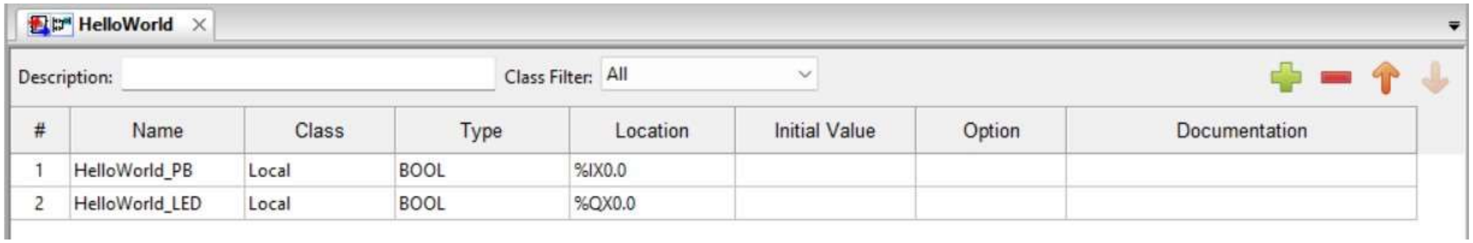
POU Type: program

Language: LD

OpenPLC Simulator Setup. . .

Create Tags for the Hello World application.

Creating Tags for the Hello World Application



#	Name	Class	Type	Location	Initial Value	Option	Documentation
1	HelloWorld_PB	Local	BOOL	%IX0.0			
2	HelloWorld_LED	Local	BOOL	%QX0.0			

The Location on the Tag Listing table is where the addresses for the GPIO pins are included.

Question 4

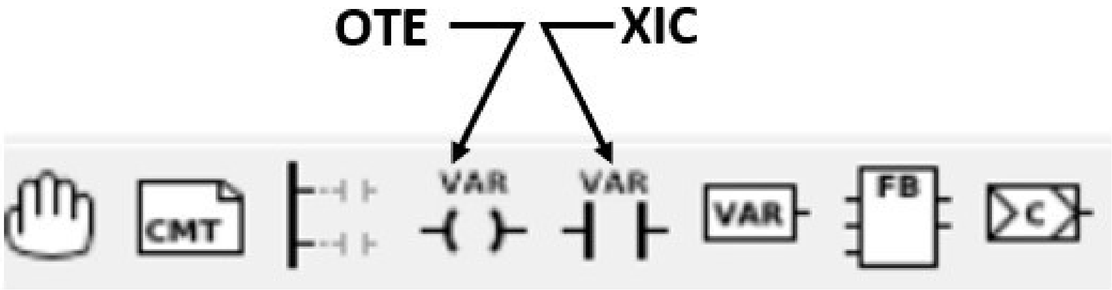
Which BIT instruction examines the state of a specified bit?

a) Programming Organizational Unit
b) Programming Organized Unit
c) Program Organizational Unit
d) None of the above

OpenPLC Simulator Setup. . .

A contact and coil will be used to build the Hello World Simulator application.

Selecting Contact and Coil BIT Instructions

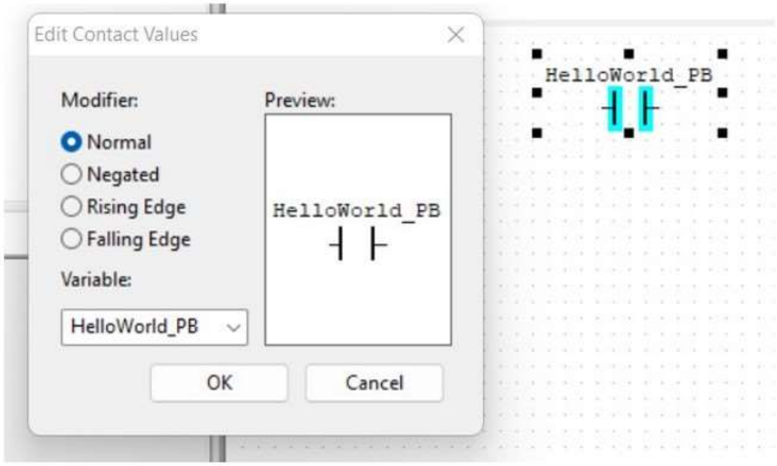


Notes:
OTE – Output Energized
XIC – Examine If Closed

OpenPLC Simulator Setup. . .

A contact and coil are aligned with defined Tags.

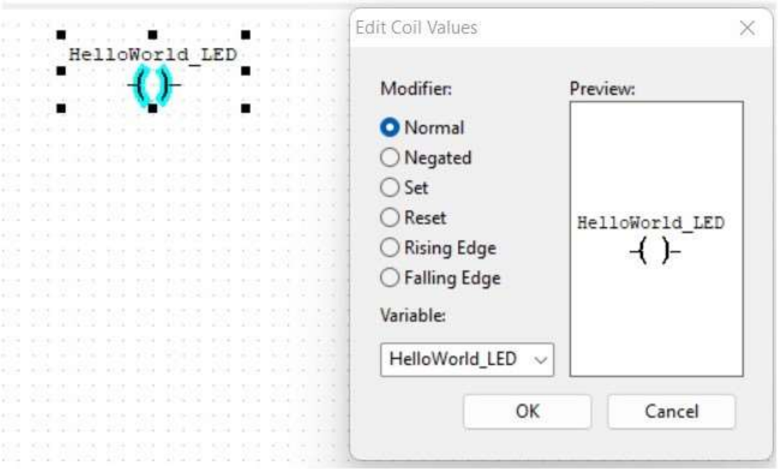
Aligning Contact and Coil With Tags



Edit Contact Values

Modifier:
- ● Normal
- ○ Negated
- ○ Rising Edge
- ○ Falling Edge

Variable: HelloWorld_PB



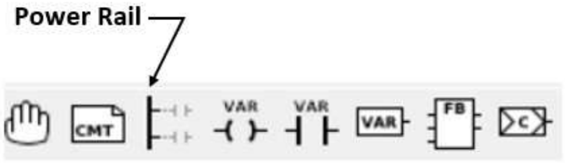
Edit Coil Values

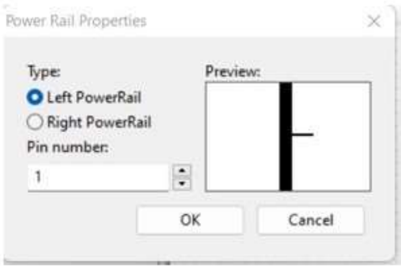
Modifier:
- ● Normal
- ○ Negated
- ○ Set
- ○ Reset
- ○ Rising Edge
- ○ Falling Edge

Variable: HelloWorld_LED

OpenPLC Simulator Setup. . .

Selection and Placement of Power Rails.



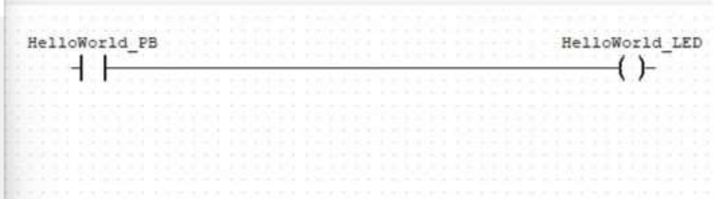

Power Rail Properties

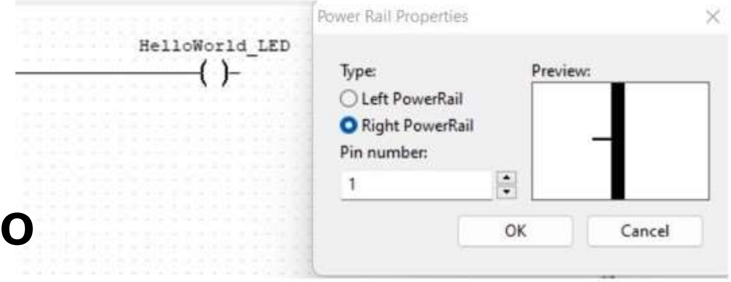
Type:

● Left PowerRail

○ Right PowerRail

Pin number: 1





Power Rail Properties

Type:

○ Left PowerRail

● Right PowerRail

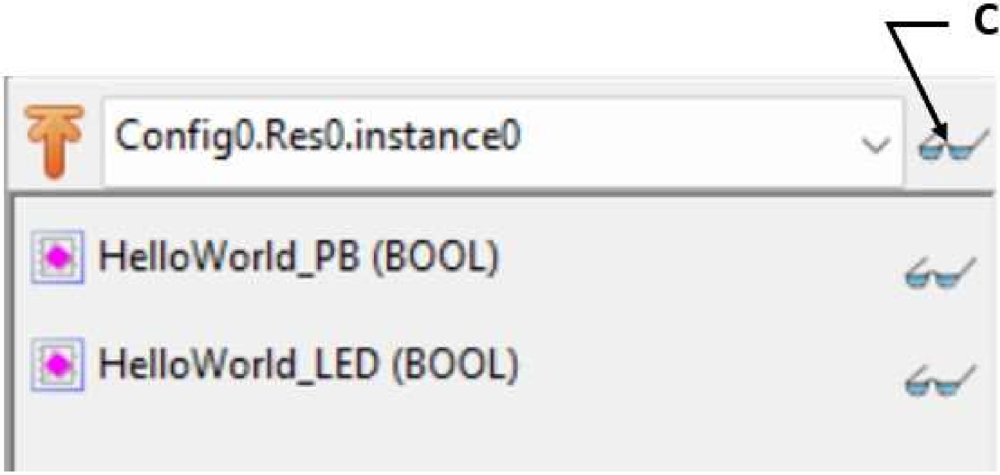
Pin number: 1

Adding power rails to contact and coil components (BIT instructions)

OpenPLC Simulator Setup. . .

Starting a Simulation Initiated Event.

Click here to initiate a simulation event



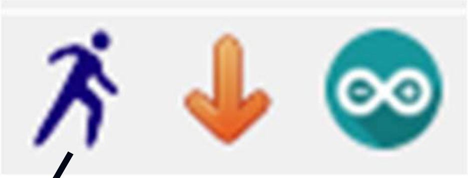
Config0.Res0.instance0

HelloWorld_PB (BOOL)

HelloWorld_LED (BOOL)

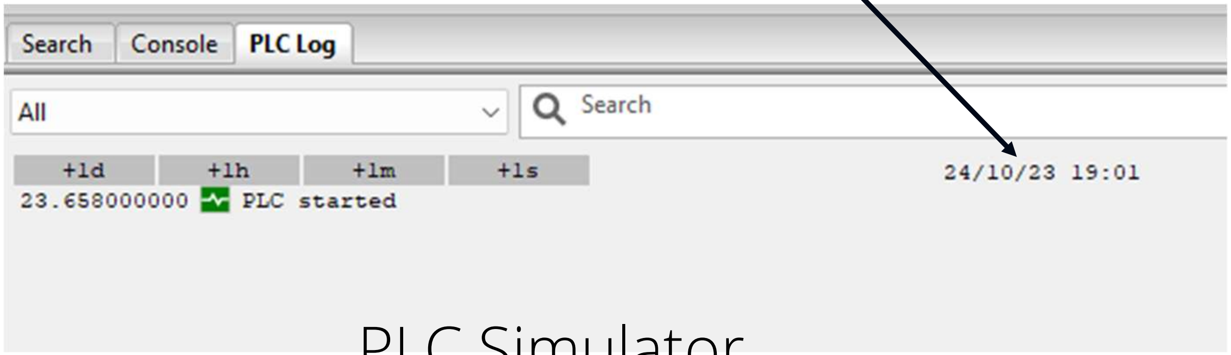
OpenPLC Simulator Setup. . .

Starting PLC Simulation event.



Start simulation

Time PLC Simulation Started

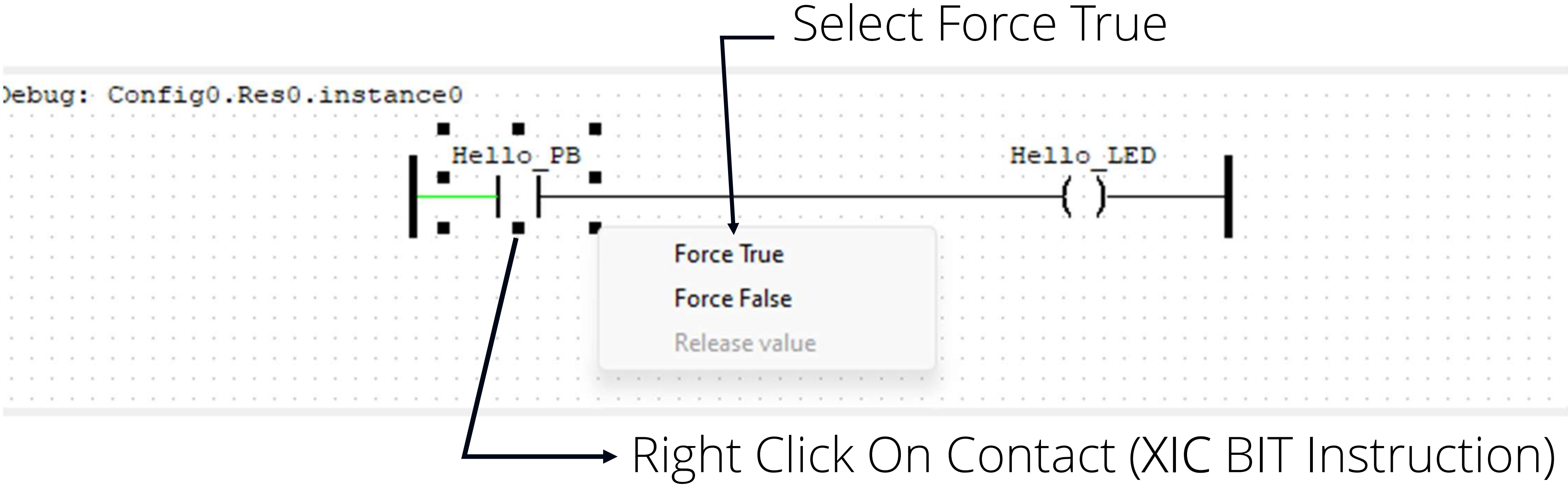


```
24/10/23 19:01
23.658000000 PLC started
```

PLC Simulator Started

OpenPLC Simulator Setup. . .

How to Force ON/Force OFF events.



Select Force True

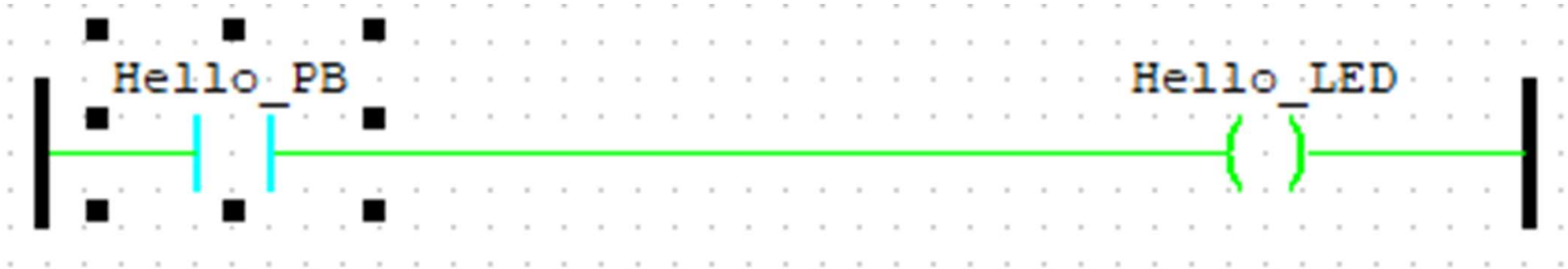
```
)ebug: Config0.Res0.instance0
```

Right Click On Contact (XIC BIT Instruction)

OpenPLC Simulator Setup. . .

How to Force ON/Force OFF events. . .

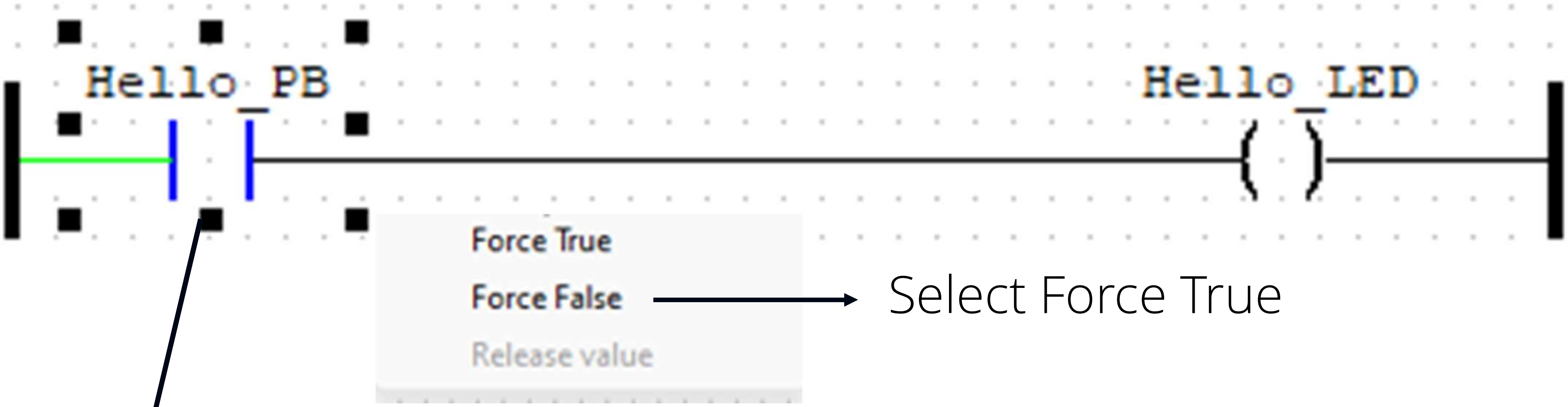
Rung link and Coil (OTE) turns Green:
Output and Rung link are energized



OpenPLC Simulator Setup. . .

How to Force ON/Force OFF events. . .

Rung link and Coil (OTE) turns Black:
Output and Rung link are de-energized

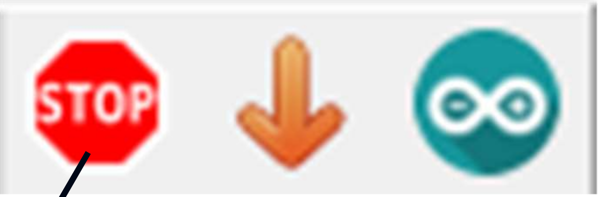


Select Force True

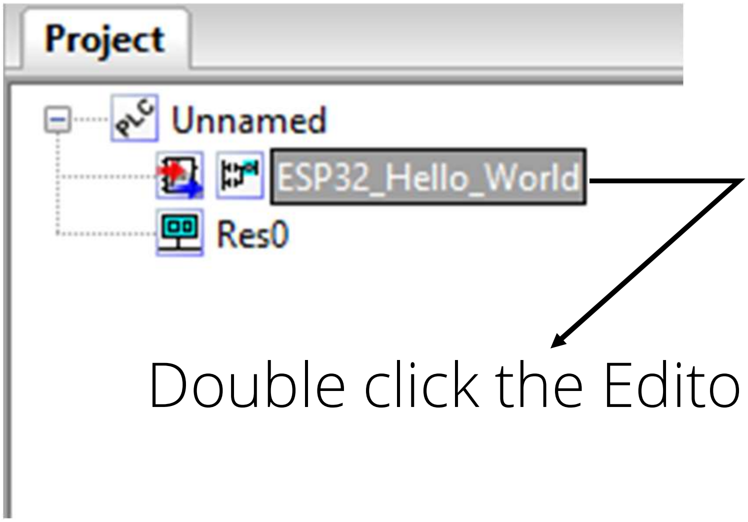
Right Click On Contact (XIC BIT Instruction)

OpenPLC Simulator Setup. . .

Stopping Simulation Event and Returning to Programming Editor mode.

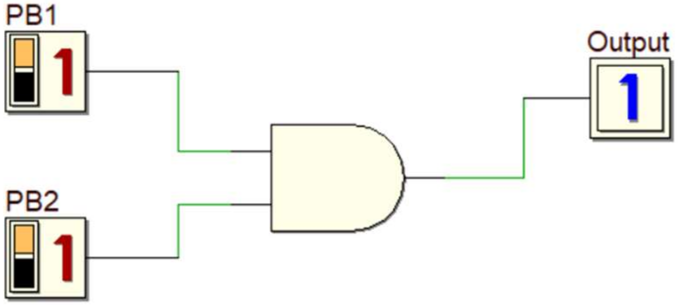


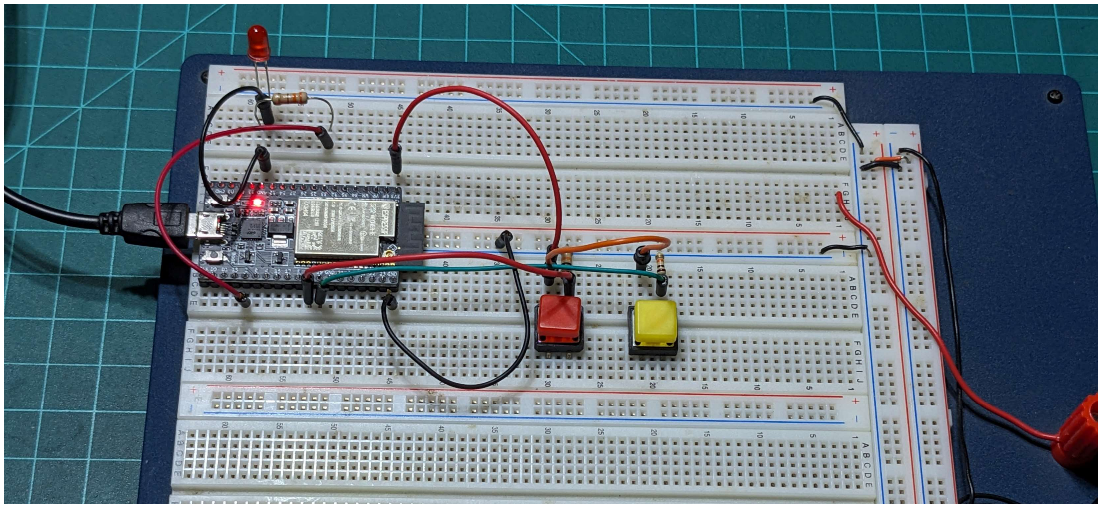
Stop simulation



Double click the Editor Icon

Lab: Digital Logic Control Application



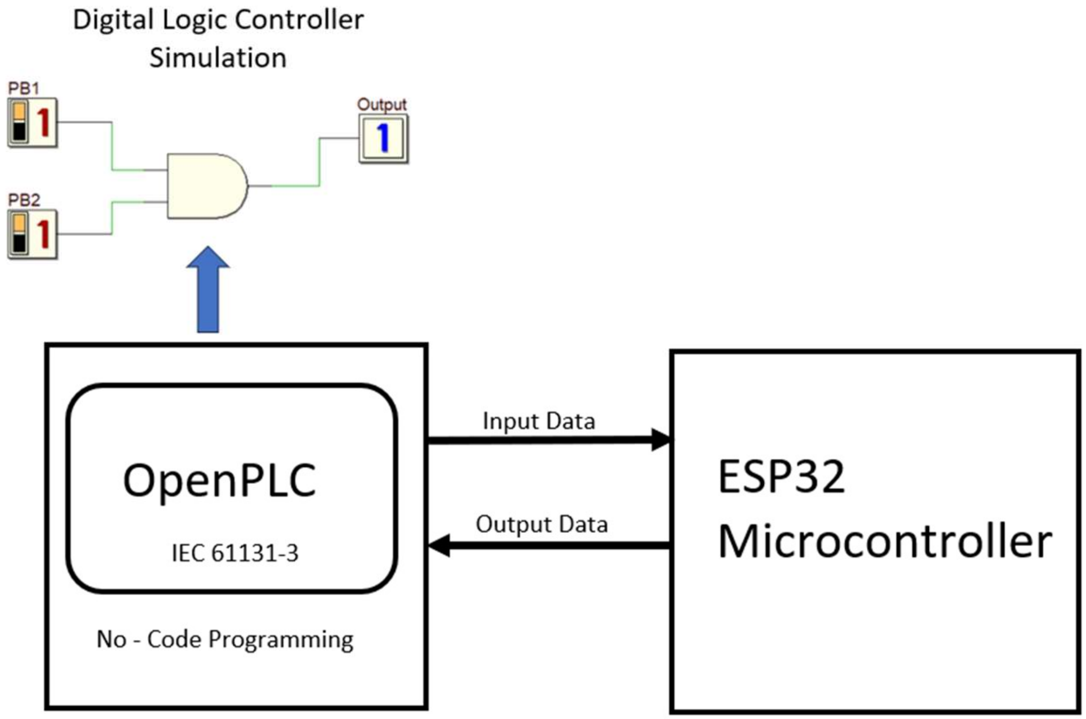


Lab: Digital Logic Control Application . . .

Lab Objectives:

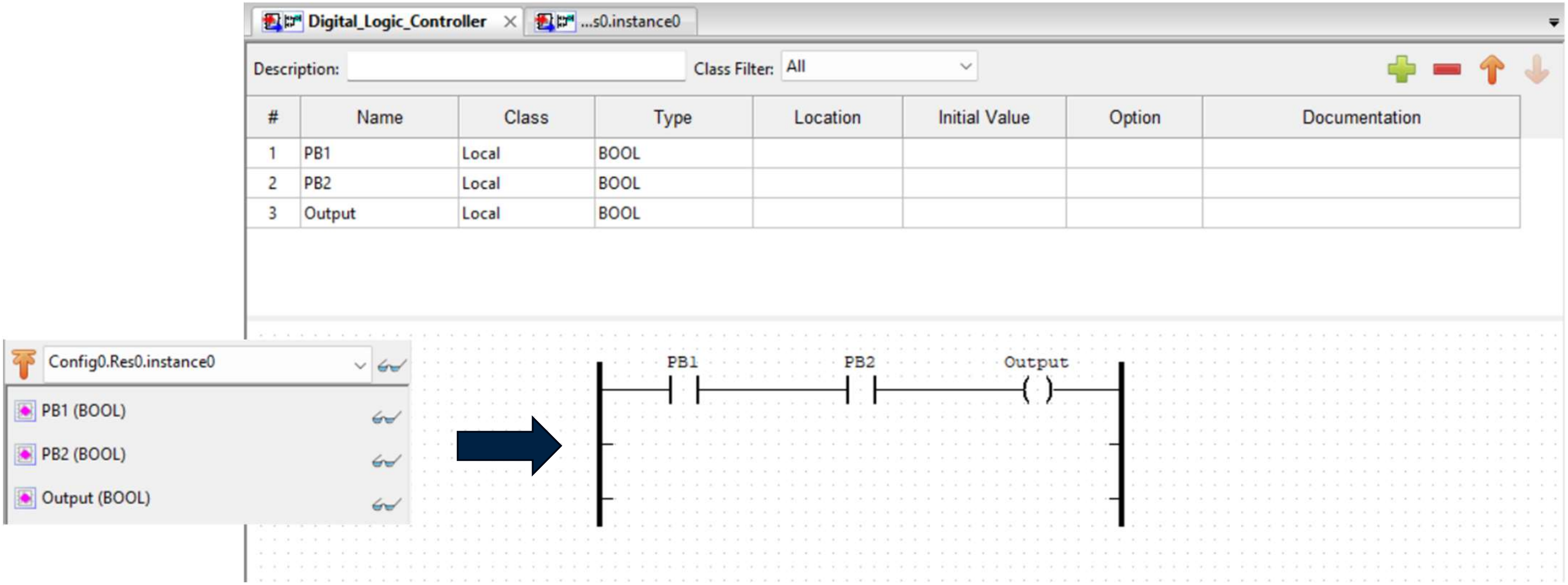
- Participants will learn to set up a Basic Digital Logic Controller simulation.
- Participants will learn to execute a Basic Digital Logic Controller Simulation.
- Participants will learn to program the ESP32 microcontroller using OpenPLC.
- Participants will learn how to run the Basic Digital Logic Controller on an ESP32 microcontroller

Lab: Digital Logic Control Application . . . Concept System Block Diagram



Lab: Digital Logic Control Application . . .

OpenPLC Digital Logic Controller Model



Digital_Logic_Controller × ...s0.instance0

Description: Class Filter: All

#	Name	Class	Type	Location	Initial Value	Option	Documentation
1	PB1	Local	BOOL				
2	PB2	Local	BOOL				
3	Output	Local	BOOL				

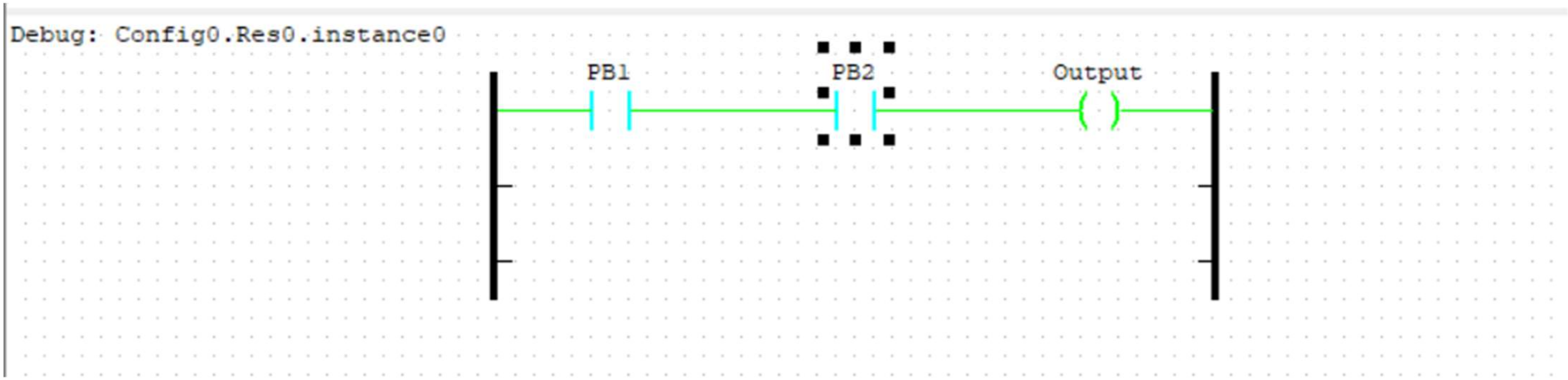
Config0.Res0.instance0

- PB1 (BOOL)
- PB2 (BOOL)
- Output (BOOL)

Lab: Digital Logic Control Application . . .

OpenPLC Digital Logic Controller Model

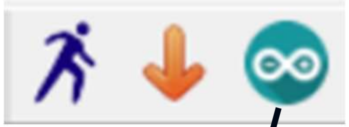
Executed Simulation session



Debug: Config0.Res0.instance0

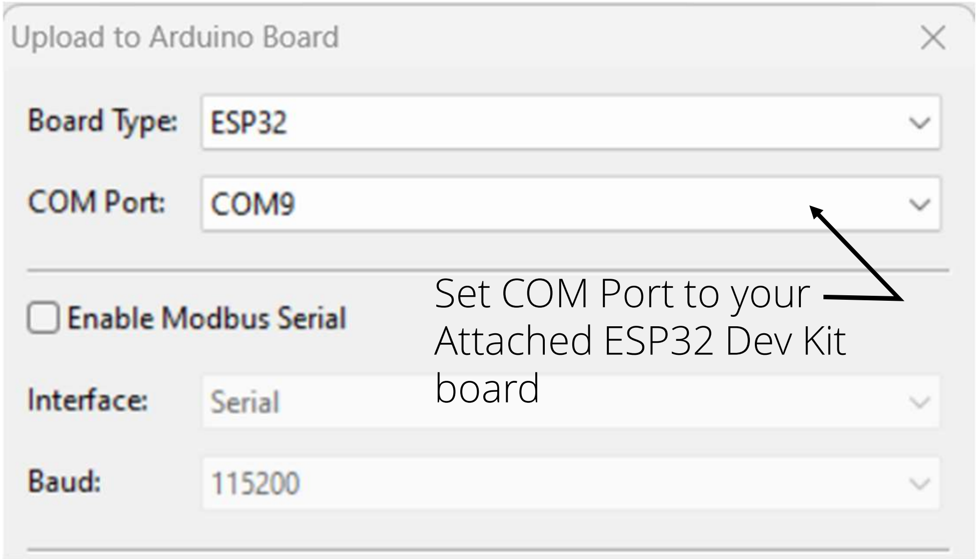
Lab: Digital Logic Control Application . . .

OpenPLC Digital Logic Controller Hardware Setup



Opens

Click Arduino Icon



Upload to Arduino Board

Board Type: ESP32

COM Port: COM9

□ Enable Modbus Serial

Interface: Serial

Baud: 115200

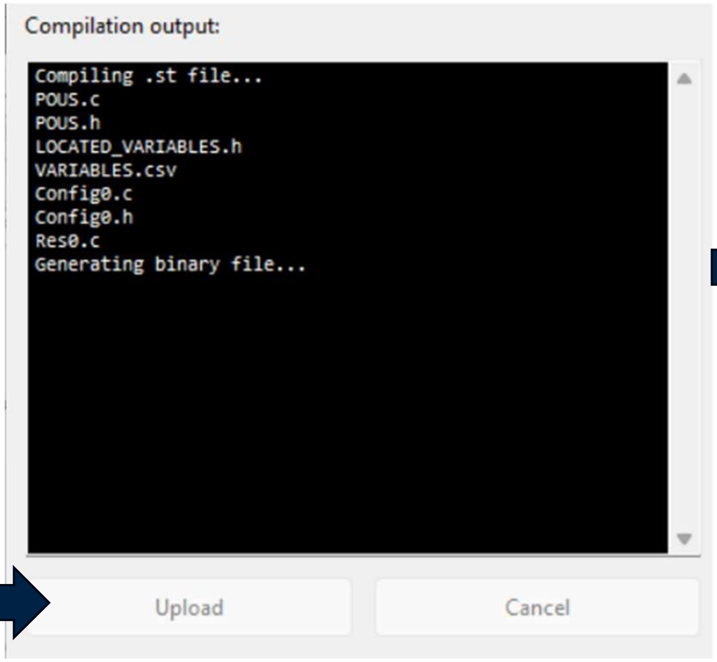
Set COM Port to your Attached ESP32 Dev Kit board

Lab: Digital Logic Control Application . . .

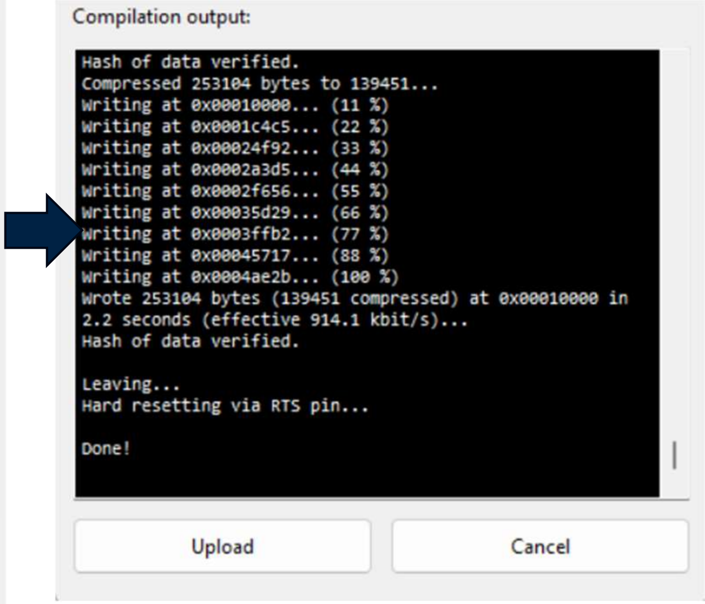
OpenPLC Digital Logic Controller Hardware Setup

Upload LD program to the ESP32 Dev Kit Board

Click Arduino Icon


Compilation output:

```
Compiling .st file...
POUS.c
POUS.h
LOCATED_VARIABLES.h
VARIABLES.csv
Config0.c
Config0.h
Res0.c
Generating binary file...
```


Compilation output:

```
Hash of data verified.
Compressed 253104 bytes to 139451...
Writing at 0x00010000... (11 %)
Writing at 0x0001c4c5... (22 %)
Writing at 0x00024f92... (33 %)
Writing at 0x0002a3d5... (44 %)
Writing at 0x0002f656... (55 %)
Writing at 0x00035d29... (66 %)
Writing at 0x0003ffb2... (77 %)
Writing at 0x00045717... (88 %)
Writing at 0x0004ae2b... (100 %)
Wrote 253104 bytes (139451 compressed) at 0x00010000 in
2.2 seconds (effective 914.1 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...

Done!
```

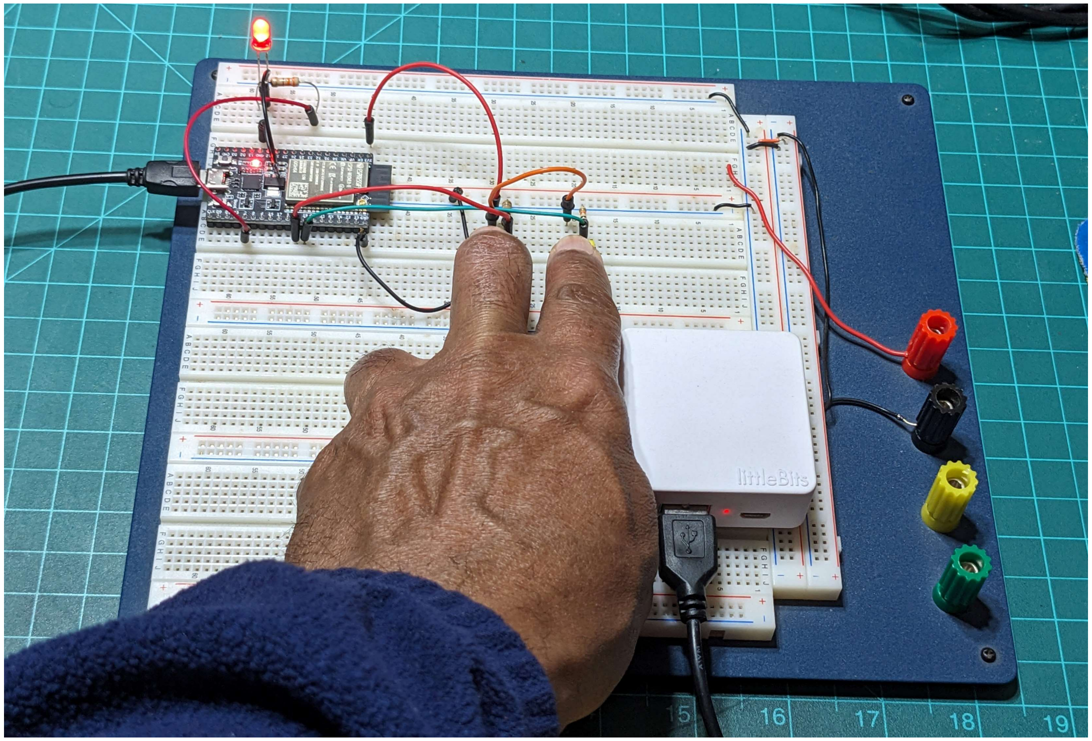
Lab: Digital Logic Control Application . . .

OpenPLC Digital Logic Controller Hardware Setup

Functional OpenPLC Digital Logic Controller

Digital Logic Controller LD program can be found in the Github Embedded_Controls_ DevelopmentCode folder here:

https://github.com/DWilcher/HCI_Electronics



Watch Video Clip Below

https://youtu.be/lEFBvDeUa9A

Question 5

What file is generated when uploading an LD program to an ESP32 microcontroller?

a) ST file
b) IL file
c) hex file
d) binary file

Thank you for attending

Please consider the resources below:

International Electrotechnical Commission. (2003). *International standard* (IEC61131-3). https://d1.amobbs.com/bbs_upload782111/files_31/ourdev_569653.pdf

Zemmouri, A., Barodt, A., Dahou, H., Alarequi, M., Eigouri, R., Htou, L., & Benbrahim, M. (2023). A microsystem design for controlling a dc motor by pulse width modulation using microblaze soft-core. *International Journal of Electrical and Computer Engineering*, *13*(2), 1337-1448. https://www.researchgate.net/publication/365994306_A_microsystem_design_for_controlling_a_DC_motor_by_pulse_width_modulation_using_MicroBlaze_soft-core

OpenPLC.(2023). *Openplc overview*. https://autonomylogic.com/docs/openplc-overview/

CEC Continuing Education Center

DesignNews

Thank You

Sponsored by

DigiKey

informa markets