



DesignNews

Understanding Industrial Controls with an ESP32

Day 4: OpenPLC and ESP32 Industrial Controls-Part 1

Sponsored by

DigiKey



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.

ESP32-DEVKITC-V1E

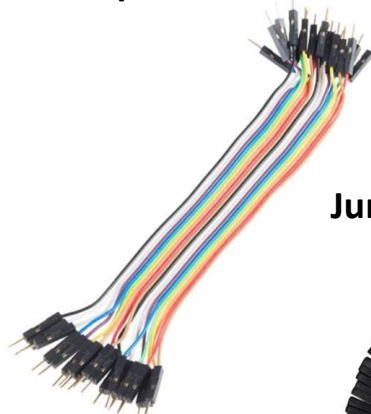


Course Kit and Materials

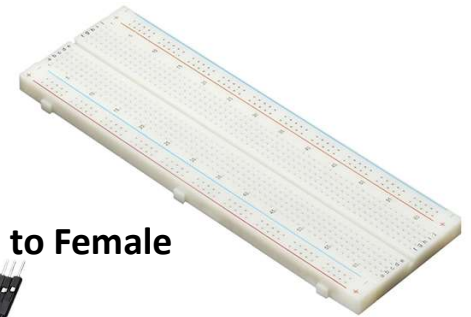
Adafruit Parts Pal Kit



Jumper Wires: Male to Male



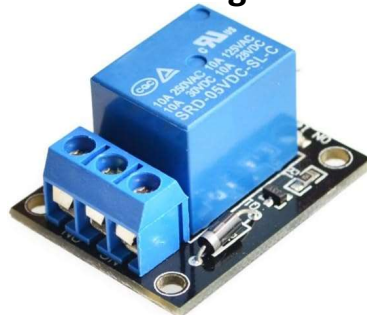
Solderless Breadboard x2



Jumper Wires: Male to Female



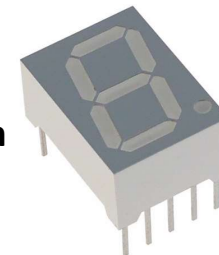
5V Relay Module, 5V Indicator Light LED



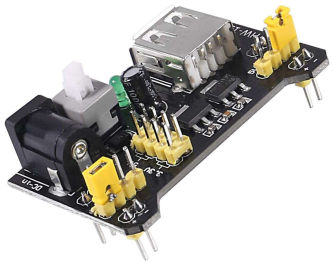
Standard Motor, 9100 RPM 6VDC



7 Segment LED Display, Common Cathode



Solderless Breadboard Power Supply



Agenda:

- OpenPLC Introduction
- Programmable Logic Controller (PLC) Architecture
- OpenPLC Software Set Up
- Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller

Research Perspective

“Programmable logic controllers (PLCs) provide an ecosystem of relatively simple software logic, robust and ruggedized hardware, networks with controllable real-time behaviors, and extensive availability of interoperable components such as sensors and actuators” (Sehr et al., 2021).

Course Question

Can an ESP32 microcontroller contribute to the Industrial Controls field?

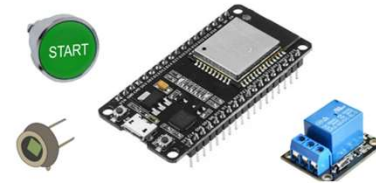


OpenPLC Introduction

- OpenPLC is an open-source industrial control platform that allows the transformation of:
 - a) popular microcontrollers into a programmable logic controller (PLC).
 - b) Raspberry Pi Single Board Computers (SBCs) into a PLC.
- OpenPLC is a fully functional open-source PLC based on the International Electrotechnical Commission (IEC standard 61131-3 on Functional Programmable Languages.

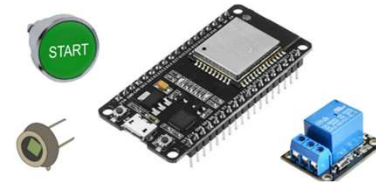


OpenPLC Introduction...



- There are five functional programming languages associated with the IEC standard.
 - a) Structured Text (ST) (Text based)
 - b) Instruction List (IL) (Text based)
 - c) Ladder Diagram (LD) (Graphical based)
 - d) Sequence Function Chart (SFC) (Graphical based)
 - e) Function Block Diagram (FBD) (Graphical based)
- The most common programming language used with the PLC is the LD.

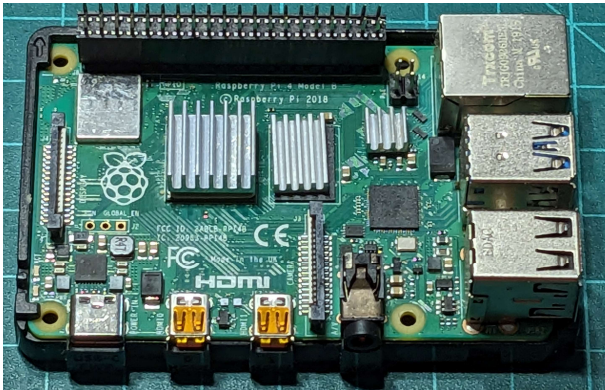
OpenPLC Introduction...



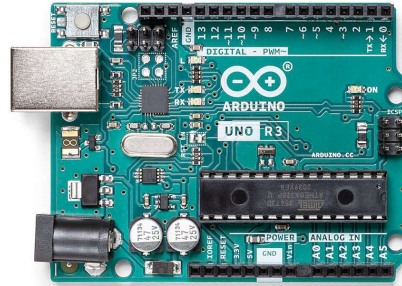
- The technical development path of the OpenPLC platform consisted of several iterative versions of the open-source programming language toolchain.
- OpenPLC Version 3 provides support for a variety of embedded devices and platforms like:
 - a) Arduino varieties
 - b) ESP32
 - c) ESP8266
 - d) Raspberry Pi

OpenPLC Introduction...

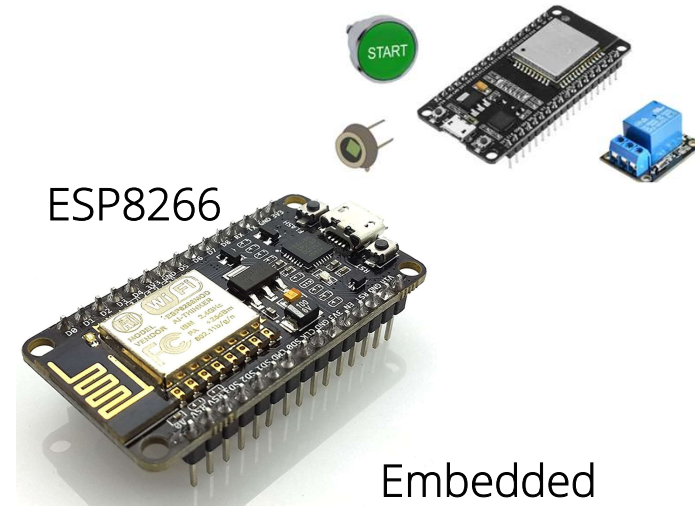
Raspberry Pi 4B



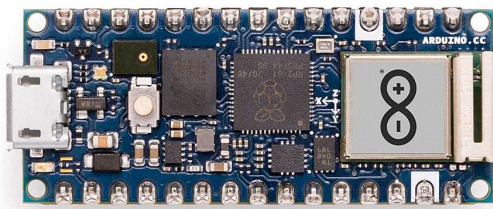
Arduino Uno



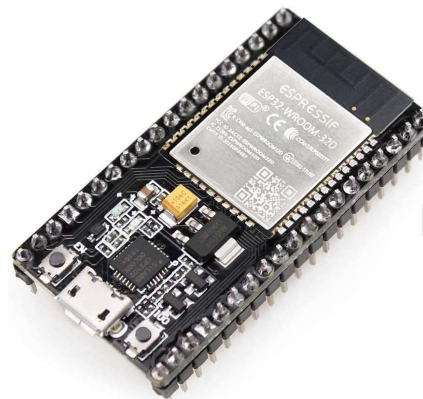
ESP8266



Arduino Nano RP2040 Connect



ESP32



Embedded
Devices and
Platforms
Supported by
OpenPLC

Question 1

Which embedded device is not supported by OpenPLC Version 3?

- a) Arduino varieties**
- b) nRF51822**
- c) ESP32**
- d) ESP8266**



OpenPLC Introduction...

- OpenPLC Version 3 software can run ST code on a target Microchip ATMEGA32 8-bit microcontroller.
- A runtime feature allows installing C-code on the Raspberry Pi.
- The runtime feature provides a dashboard to display the Raspberry Pi-based PLC input and output devices.
- The Broadcom (BCM) microcontrollers supported by version 3 software are:
 - a) BCM 2711 (Raspberry Pi version 4B)
 - b) BCM 2937 (Raspberry Pi version 3B)



OpenPLC Introduction...

Example OpenPLC Runtime Environment



OpenPLC Running: HelloWorld OpenPLC User

Dashboard
Programs
Slave Devices
Monitoring
Hardware
Users
Settings
Logout

Status: *Running*

Stop PLC

Monitoring

Refresh Rate (ms): Update

Point Name	Type	Location	Forced	Value
HelloWorld_PB	BOOL	%IX100.0	No	<input checked="" type="checkbox"/> TRUE
HelloWorld_LED	BOOL	%QX100.0	No	<input checked="" type="checkbox"/> TRUE

Programmable Logic Controller (PLC) Architecture

- A typical PLC can be divided into five components.
 - a) Central Processing Unit (CPU)
 - b) Input/Output Devices
 - c) Optoisolators
 - d) Memory
 - e) Power Supply
- The term architecture refers to the PLC physical components or hardware.
- There are two types of architecture: Open and Closed



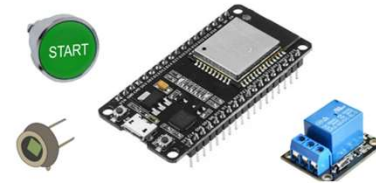
Programmable Logic Controller (PLC) Architecture...



What are Open and Closed Architecture Systems?

- An Open Architecture consists of off-the-shelf components that conform to an approved standard.
- A Closed Architecture is a proprietary system.
- Proprietary systems are difficult to connect due to using non-off-the-shelf components.

Programmable Logic Controller (PLC) Architecture...



Classical Examples of Open Architecture Systems

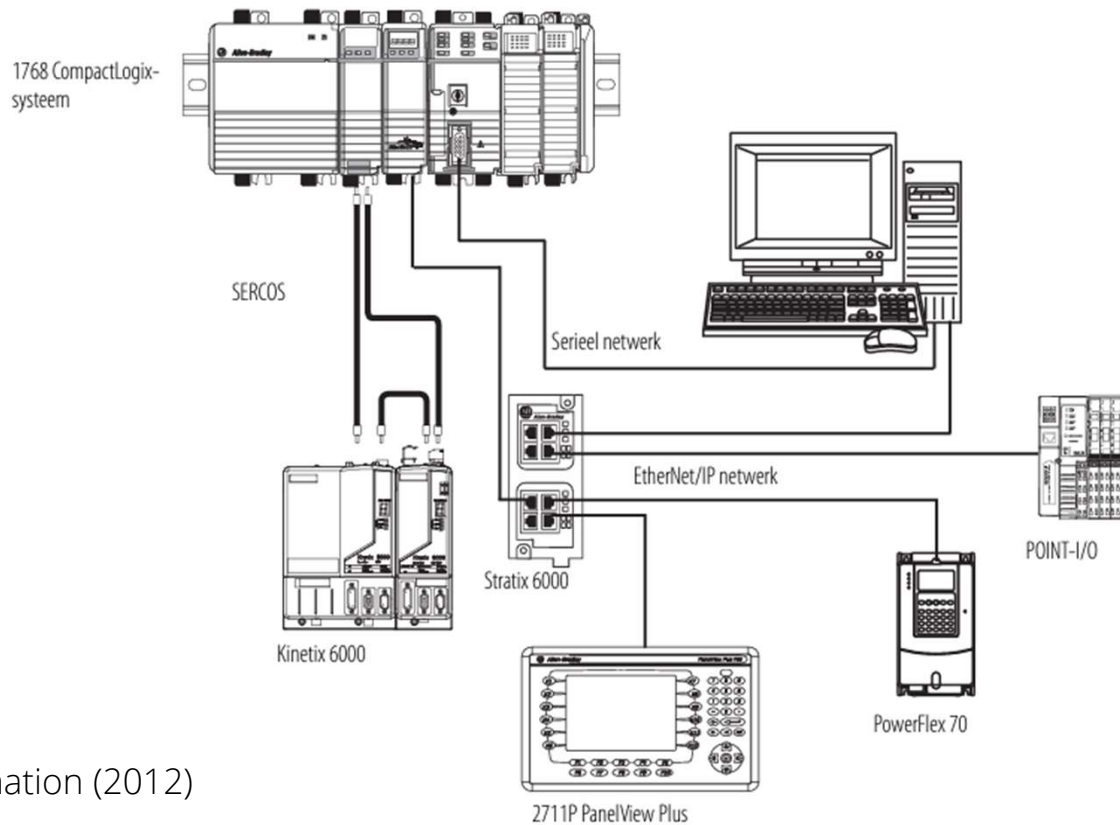
IBM PC



Apple IIc



Programmable Logic Controller (PLC) Architecture...



Example of Open
Architecture PLC
System

Programmable Logic Controller (PLC) Architecture...

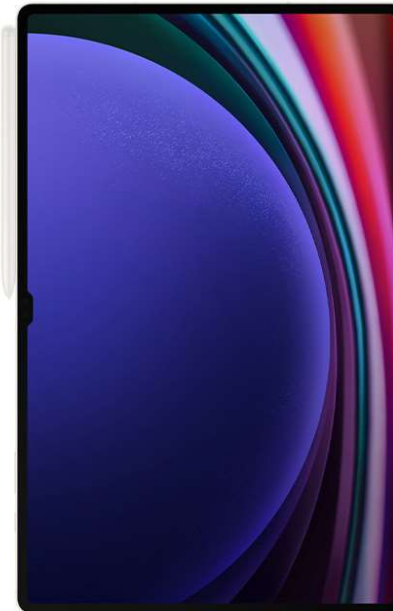
Examples of Closed Architecture Systems



Apple iPhone



Samsung Tablet



Programmable Logic Controller (PLC) Architecture...

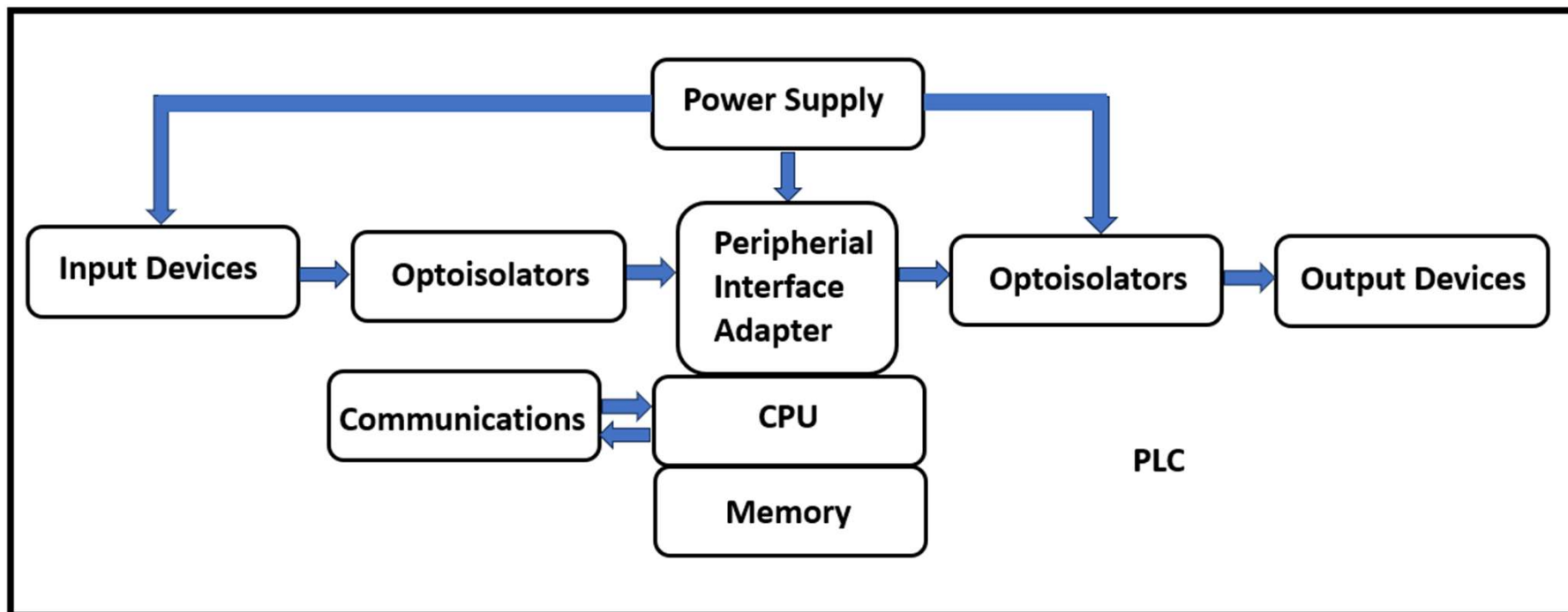
Example of A Closed PLC Architecture System



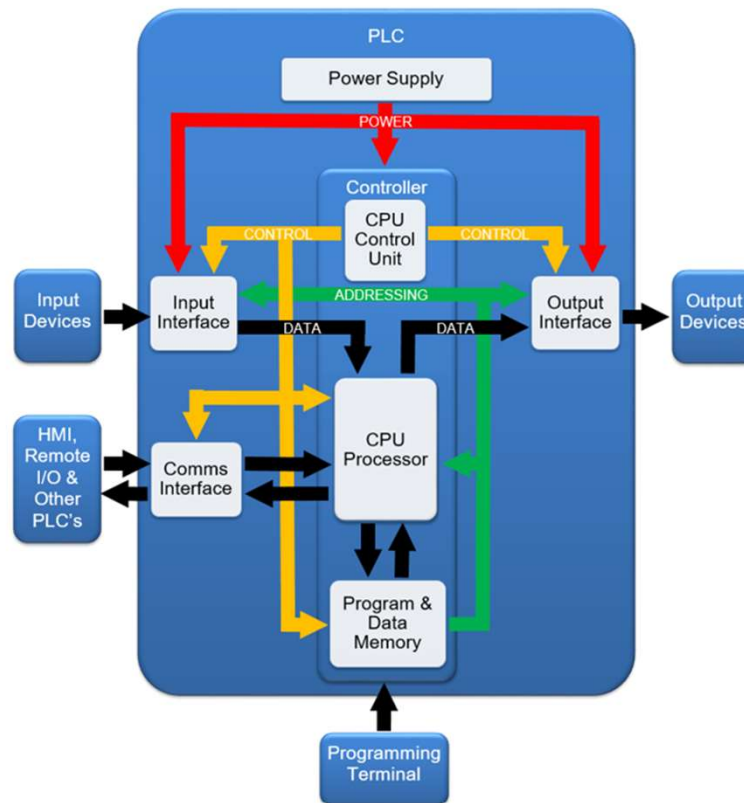
ABB (n.d.)

Programmable Logic Controller (PLC) Architecture...

Top Level Design



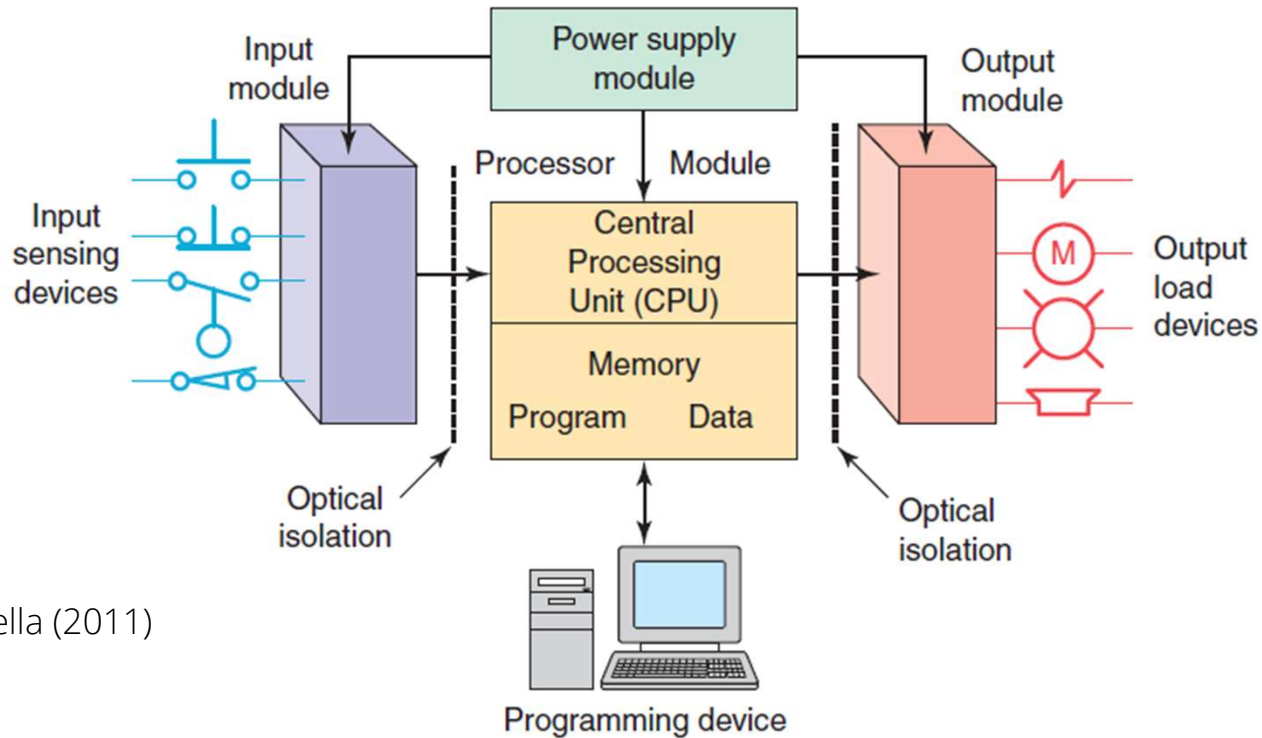
Programmable Logic Controller (PLC) Architecture...



Top Level Design

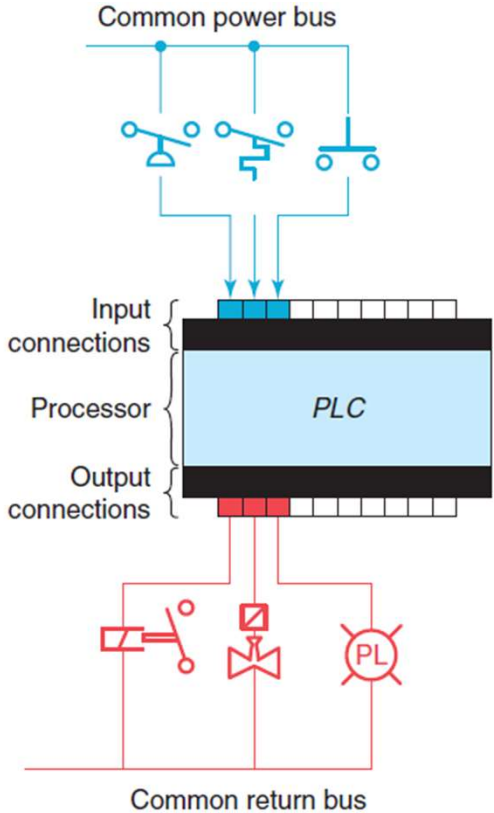
LadderLogicWorld (n.d.)

Programmable Logic Controller (PLC) Architecture... (Component Level Design)



Petruzella (2011)

Programmable Logic Controller (PLC) Architecture...



Input/Output Configuration

Petruzella (2011)

Question 2

What is an Open Architecture System?

- a) A system that uses off-the-shelf components that conform to an approved standard.**
- b) A system that uses special components that conform to an approved standard.**
- c) A system that is difficult to connect due to using non-off-the-shelf components.**
- d) none of the above**



OpenPLC Software Set Up



Steps:

1. The first step in setting up the OpenPLC platform is downloading the software.
2. Go to the Autonomy website to obtain the OpenPLC software.
<https://autonomylogic.com/>
3. Download and Install the OpenPLC software on your development machine.
4. Run the OpenPLC software on your development machine.
5. Build your ladder diagram (LD) (ladder logic program).
6. Download the LD to the ESP32.
7. Test the LD on the ESP32.

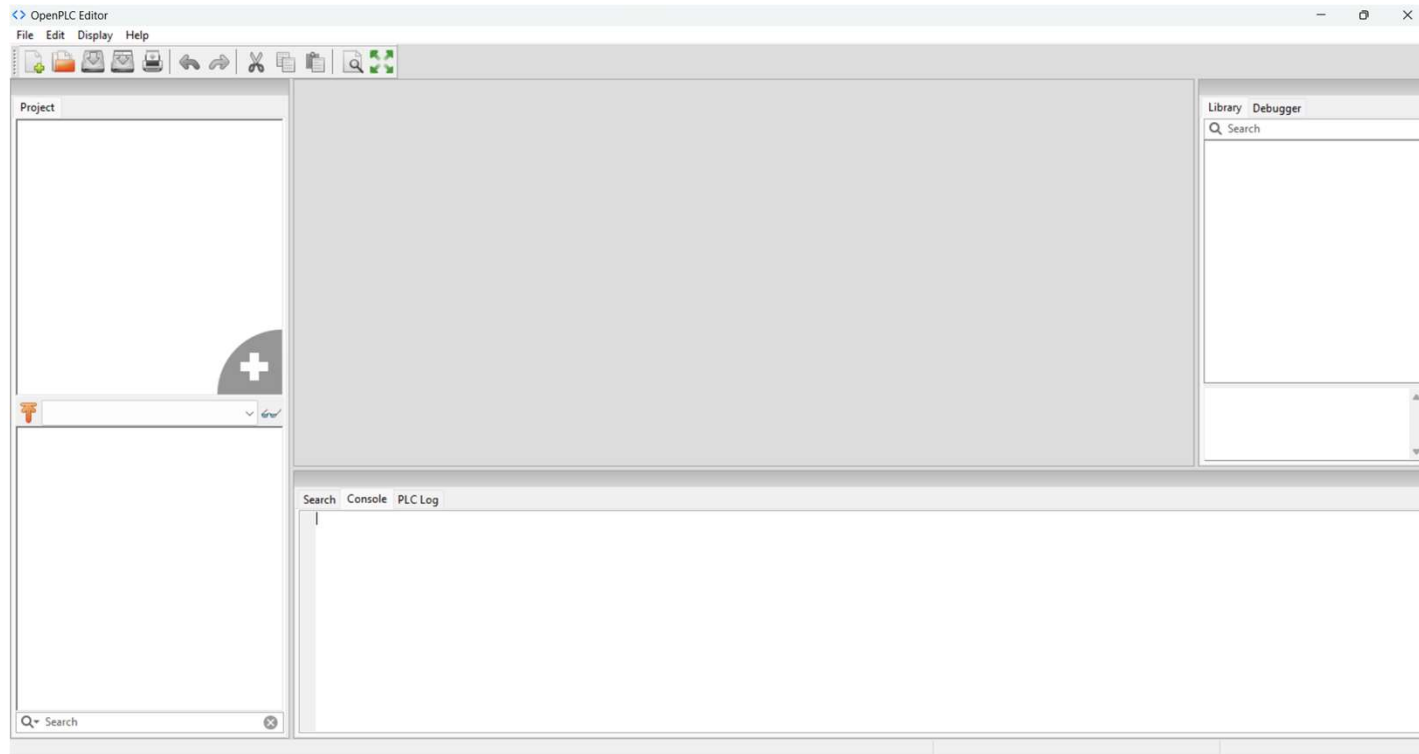
OpenPLC Software Set Up...



Autonomy-OpenPLC website

OpenPLC Software Set Up ...

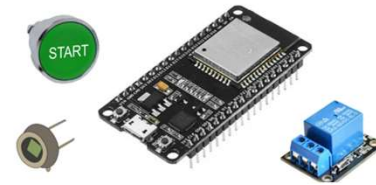
Step 4: Run the OpenPLC software on your development machine.



OpenPLC Software Set Up ...

Step 4: Run the OpenPLC software on your development machine.

ESP32 GPIO-Physical Addressing



esp32

Digital In	17, 18, 19, 21, 22, 23, 27, 32 33	%IX0.0 - %IX0.7 %IX1.0 - %IX1.0
Digital Out	01, 02, 03, 04, 05, 12, 13, 14 15, 16	%QX0.0 - %QX0.7 %QX1.0 - %QX1.1
Analog In	34, 35, 36, 39	%IW0 - %IW3
Analog Out	25, 26	%QW0 - %QW1

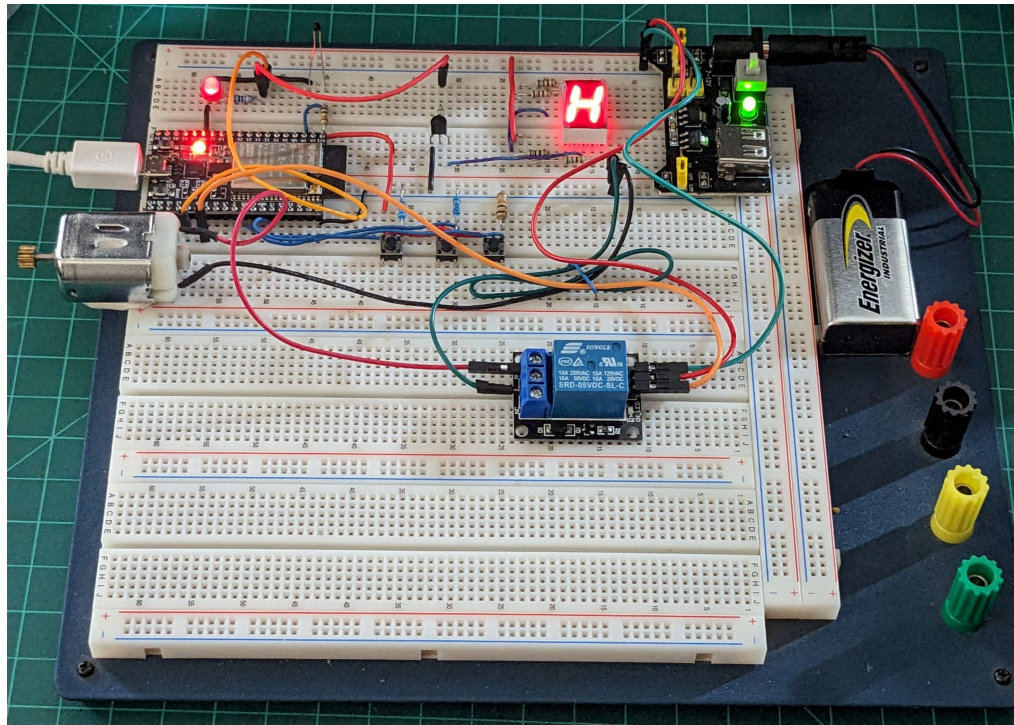
Question 3

On slide 29, what is the physical address for GPIO pin 5?

- a) %QX0.4**
- b) %QX0.0**
- c) %QX0.5**
- d) none of the above**



Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller



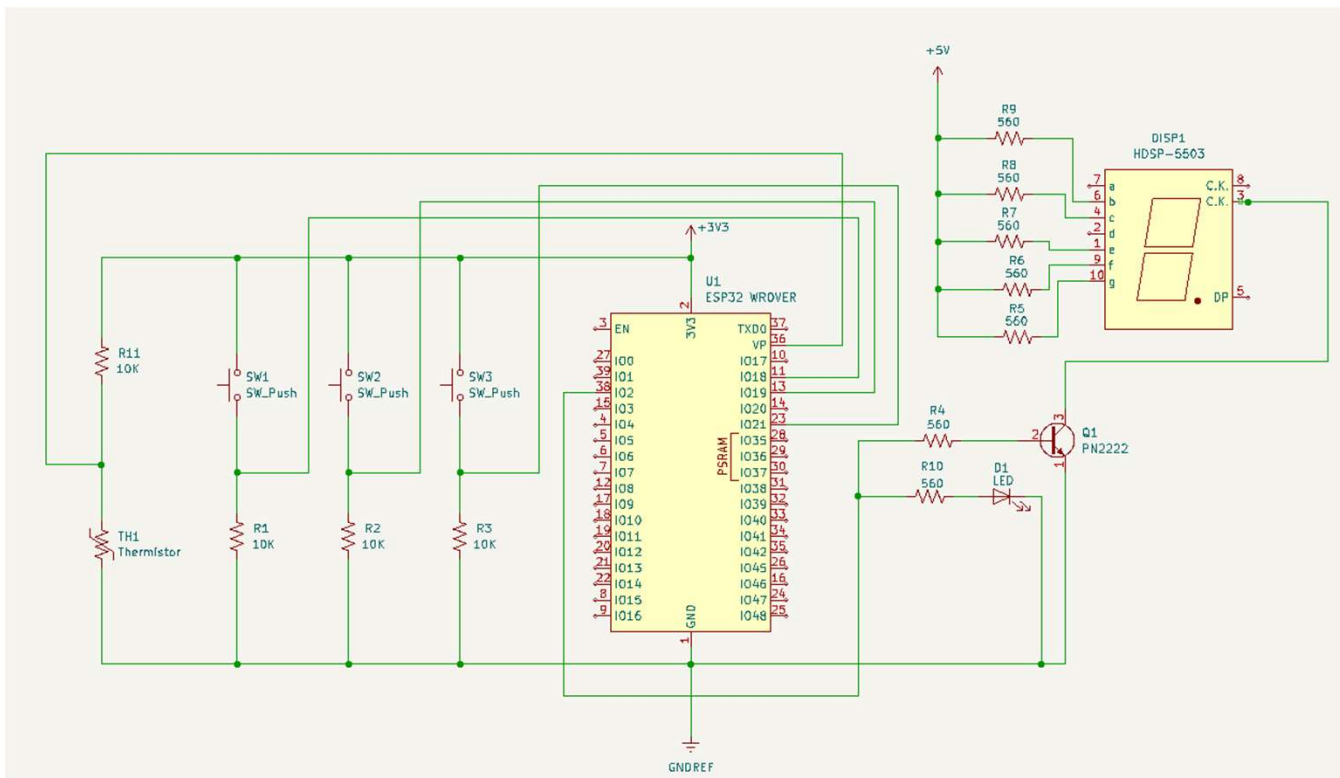
Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...



Participant Learning Objectives:

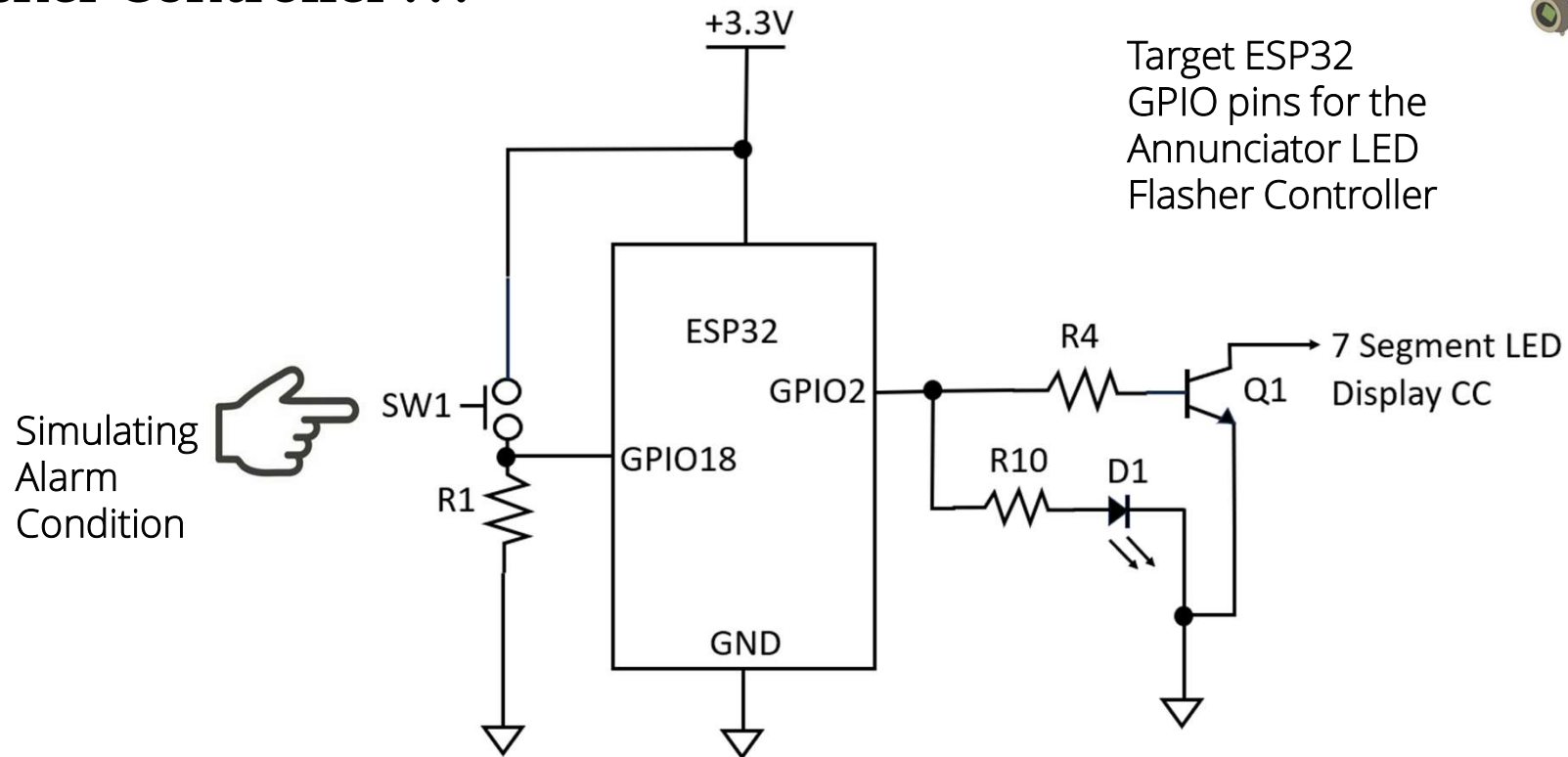
- Participants will learn to modify the OpenPLC Blink LD to detect a pushbutton switch actuation.
- Participants will learn to assign Physical I/O addresses to an ESP32 microcontroller GPIO pins.
- Participants will learn to transfer the modified Blink LD to the ESP32 microcontroller.
- Participants will learn to test the Annunciator LED Flasher Controller on the ESP32 Micro Trainer.

Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...

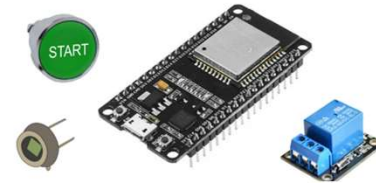


Partial ESP32
Micro Trainer
Electronic Circuit
Schematic
Diagram

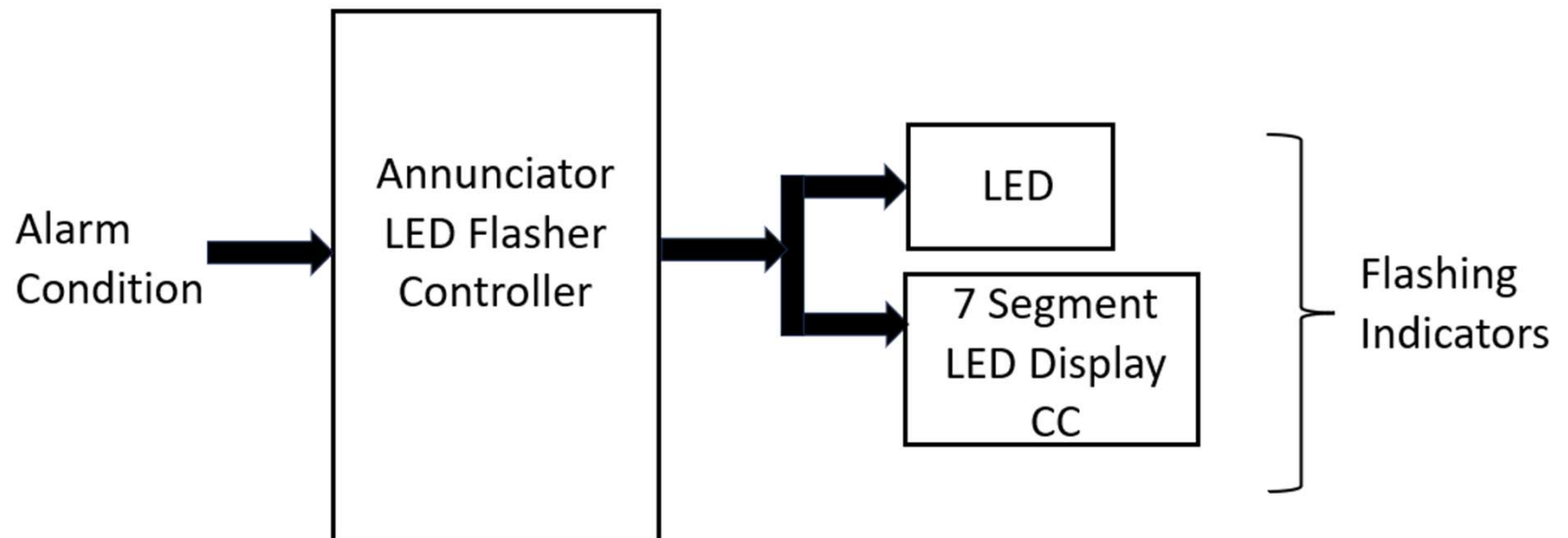
Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller ...



Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...



Concept Block Diagram

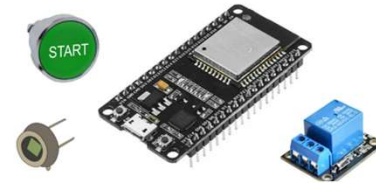


Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...

Building the Annunciator LED Controller

Steps:

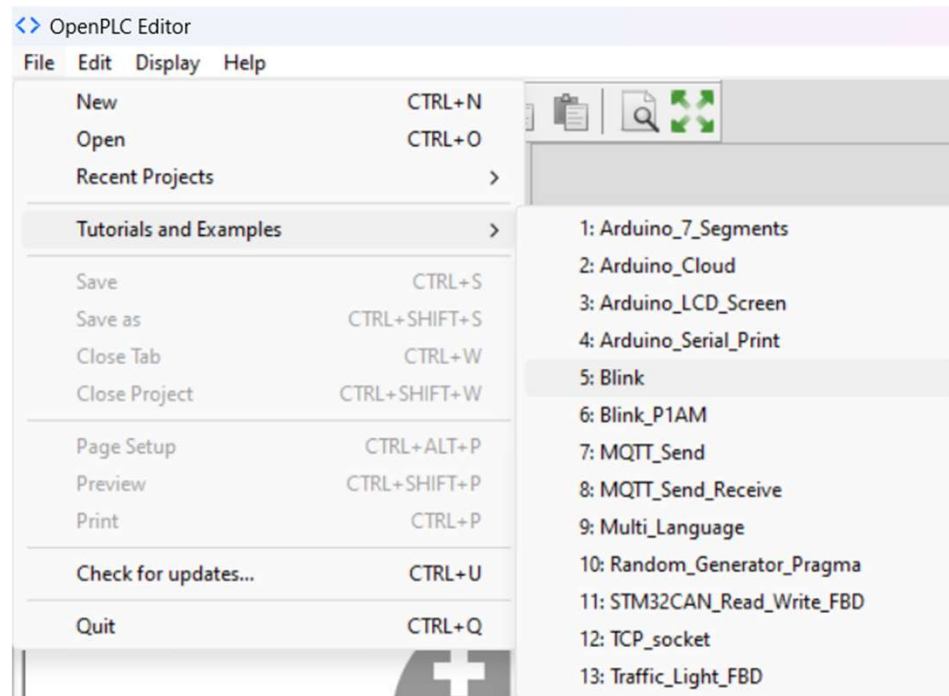
1. Create a project folder.
2. Name the folder "Annunciator_LED_Controller"
3. Open OpenPLC software
4. Click on Files>Tutorial and Examples>Blink



Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...

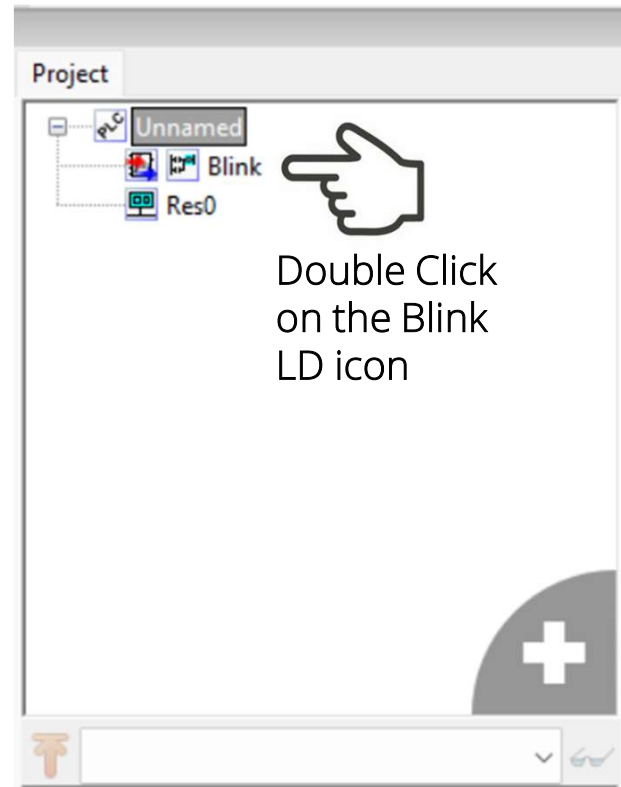


Step 4:



Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...

This panel will appear on the Screen:



Question 4

What is the name of the LD used to create the Annunciator LED Flasher Controller?

- a) Flasher**
- b) Blinker**
- c) Switcher**
- d) Blink**



Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...

The OpenPLC Editor with the Blink LD will appear on the Screen:



The screenshot displays the OpenPLC Editor interface. The main workspace shows a ladder logic diagram for a square wave generator. The diagram consists of a normally open contact labeled 'blink_led' connected to the EN (Enable) input of a TON (Timer On Delay) block. The TON block has a PT (Preset Time) of T#500ms and its Q (Output) is connected to the EN input of a TOF (Timer Off Delay) block. The TOF block also has a PT of T#500ms and its Q output is connected back to the 'blink_led' output. A text box above the diagram reads: "This example cascades two timers (TON and TOF) to generate a square wave. The width of the wave is determined by the size of the PT variable on both timers." The left sidebar shows a project tree with 'Unnamed' and 'Res0' folders, and a variable declaration table for 'Config0.Res0.instance0'.

#	Name	Class	Type	Location	Initial Value	Option	Documentation
1	blink_led	Local	BOOL				
2	TON0	Local	TON				
3	TOF0	Local	TOF				

Library / Debugger

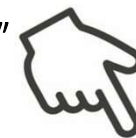
- Standard function blocks
- Additional function blocks
- Arduino
- Microver
- Communication
- P1AM Modules
- MQTT
- Sequent Microsystems Modules
- Jaguar
- SL-RP4
- Type conversion
- Numerical
- Arithmetic
- Time
- Bit-shift
- Bitwise
- Selection
- Comparison
- Character string
- Native POU's
- User-defined POU's

Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...

Creating a Tag – “Alarm_Condition”:

A “blink_led0” tag will be created and placed on the spreadsheet listing.

Click the “+” sign

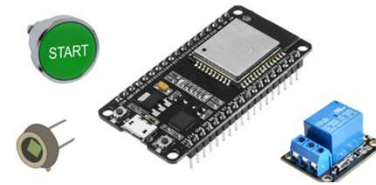


#	Name	Class	Type	Location	Initial Value	Option	Documentation
1	blink_led	Local	BOOL				
2	blink_led0	Local	BOOL				
3	TON0	Local	TON				
4	TOF0	Local	TOF				

Double-click in the blink_led0 cell. Rename the tag as “Alarm_Condition”.

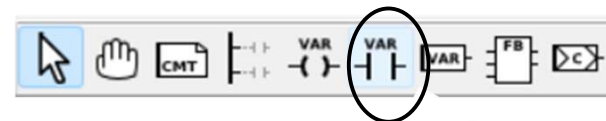
Click on the Save As and rename the LD as “Annunciator_LED_Controller”. Save the rename LD in the Annunciator_LED_Controller folder.

Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...



Alarm_Condition Tag is created!

Description:		Class Filter:	
#	Name	Class	Type
1	blink_led	Local	BOOL
2	Alarm_Condition	Local	BOOL
3	TON0	Local	TON
4	TOF0	Local	TOF



Click this
Variable
(VAR) Contact
symbol

The OpenPLC Editor with the Blink LD will appear on the Screen:

Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...



Click anywhere within the Blink LD editor, the following pane will be displayed on the screen.

The screenshot shows the 'Blink' editor window. On the left, the 'Edit Contact Values' dialog is open, showing 'Normal' selected under 'Modifier' and a 'Variable' dropdown. The main editor area displays a ladder logic diagram with a table of variables and a descriptive text box.

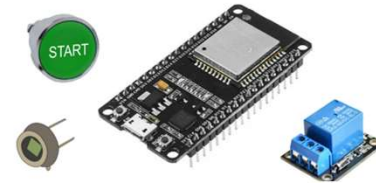
#	Name	Class	Type	Location	Initial Value	Option	Documentation
1	blink_led	Local	BOOL				
2	TON0	Local	TON				
3	TOF0	Local	TOF				

This example cascades two timers (TON and TOF) to generate a square wave. The width of the wave is determined by the size of the PT variable on both timers.

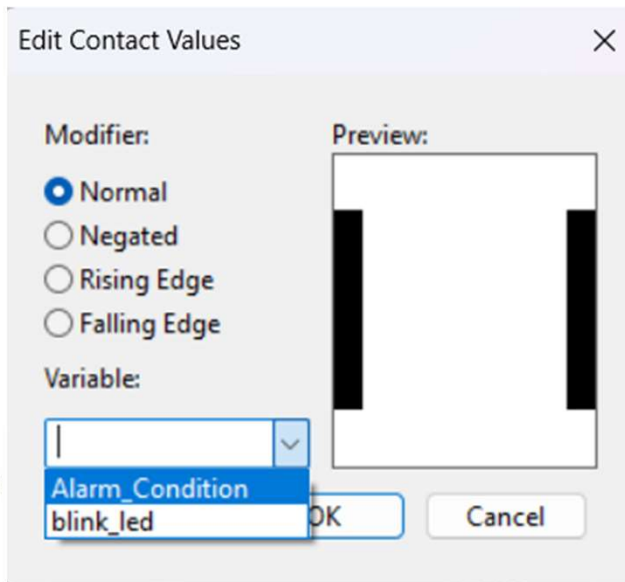
```
graph LR
    blink_led[blink_led] --> TON0[TON0]
    TON0 --> TOF0[TOF0]
    TOF0 --> blink_led
```

Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...

Assigning Alarm_Condition tag to the VAR Contact:



Click the Down arrow button and select Alarm_Condition. Click the OK button.

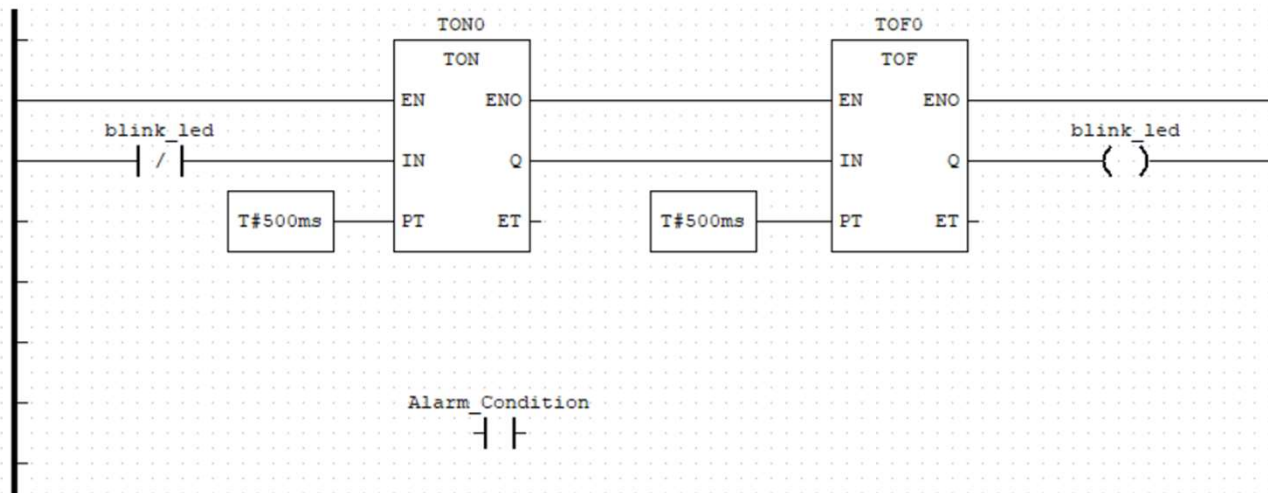


Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...

Alarm_Condition VAR Contact will be placed on the LD:



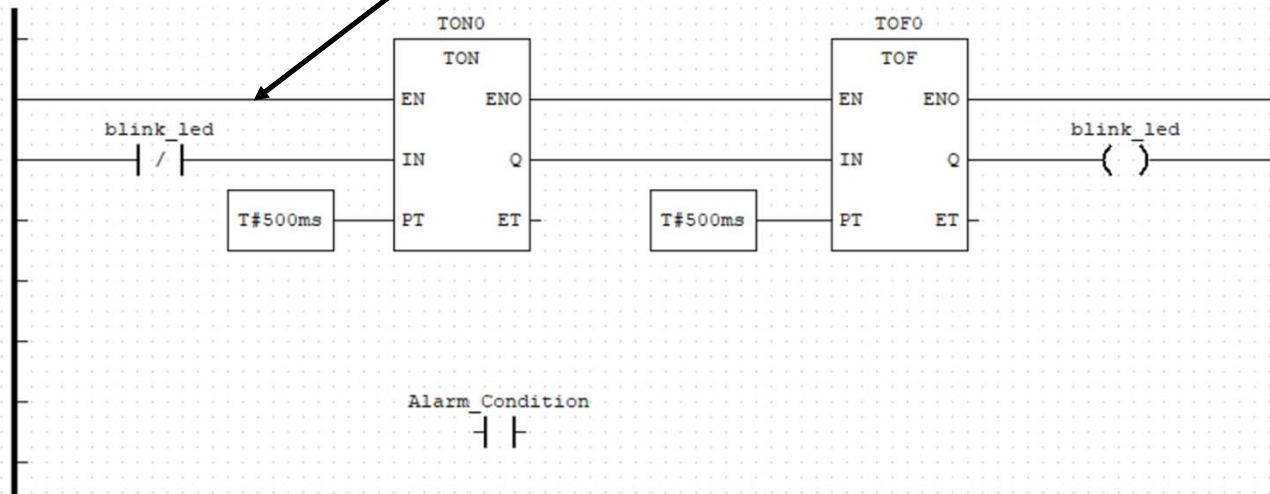
This example cascades two timers (TON and TOF) to generate a square wave. The width of the wave is determined by the size of the PT variable on both timers.



Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...

Delete the "EN" line.

This example cascades two timers (TON and TOF) to generate a square wave. The width of the wave is determined by the size of the PT variable on both timers.

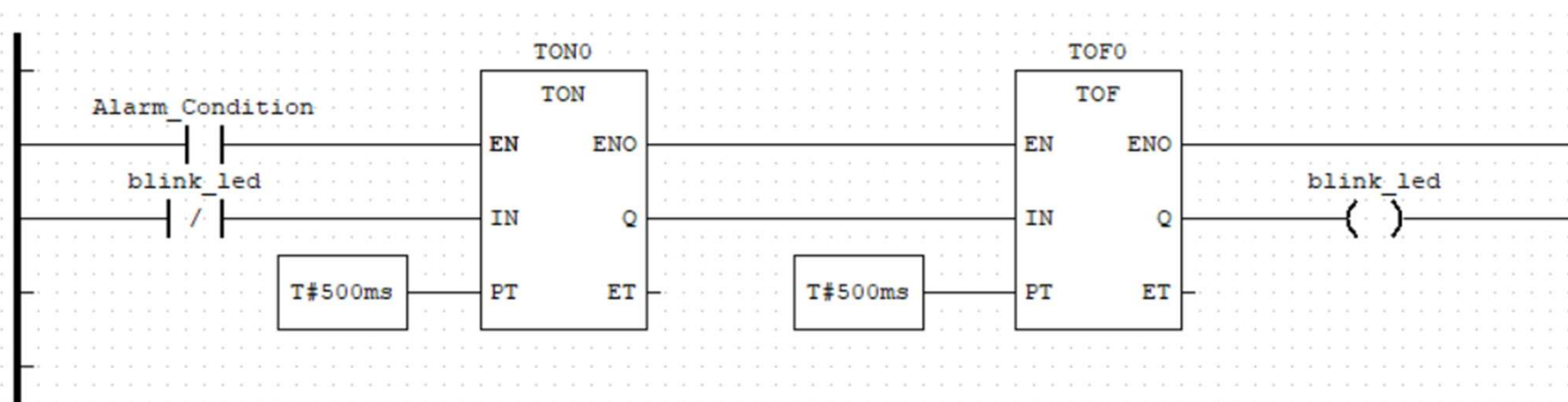


Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...



Wiring the Alarm_Condition VAR Contact to the TON timer:

Click on the left side of the VAR contact and drag a line to the left power rail. Click on the right side of the VAR contact and drag a line to the TON EN input.



Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...



1 Click the Transfer program to PLC icon



Click the Transfer icon to select the Board type and the COM port to connect with the ESP32 microcontroller.

2

Transfer Program to PLC

Board Type: ESP32 Generic (3.0.7)

COM Port: Silicon Labs CP210x USB to UART Bridge (COM3) (COM3)

Compile Only

Transfer

I/O Config

Communications

Compilation output

```

Writing at 0x0009aab0... (60 %)
Writing at 0x0009fcaa... (63 %)
Writing at 0x000a51f8... (65 %)
Writing at 0x000aa4b2... (68 %)
Writing at 0x000afb5b... (71 %)
Writing at 0x000b50f3... (73 %)
Writing at 0x000ba889... (76 %)
Writing at 0x000c0542... (78 %)
Writing at 0x000c5f18... (81 %)
Writing at 0x000ce995... (84 %)
Writing at 0x000d6b8c... (86 %)
Writing at 0x000dbc67... (89 %)
Writing at 0x000e12b7... (92 %)
Writing at 0x000e6c23... (94 %)
Writing at 0x000ec16f... (97 %)
Writing at 0x000f1c03... (100 %)
Wrote 925984 bytes (607039 compressed) at 0x00010000 in 27.2 seconds (effective 271.9 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
New upload port: COM3 (serial)
$? = 0

Done!

```

Transfer to PLC

Note:
The first time transferring the LD to the ESP32 will take several minutes, depending on the various software packages that are installed.

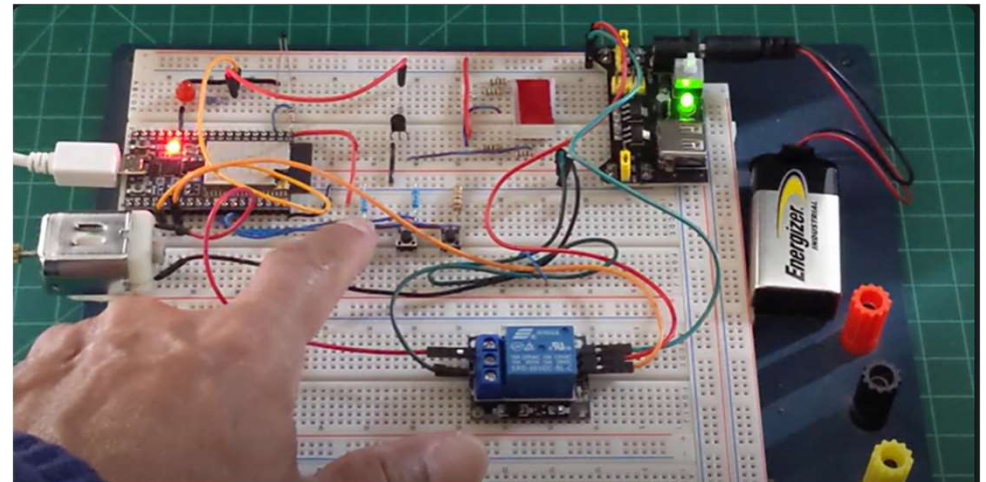
3 Click the Transfer to PLC button to load the Annunciator LD to the ESP32 microcontroller.

Lab: Build An ESP32 OpenPLC Annunciator LED Flasher Controller...

Press the SW1 pushbutton switch The LED and the 7-Segment LED display (letter H) will flash. Releasing the pushbutton switch, both visual devices will stop flashing.



Functional ESP32 OpenPLC
Annunciator LED Flasher Controller



Watch the Video Clip!

https://www.youtube.com/watch?v=PF_OoM2oZvQ

Question 5

What symbol was used to create the Alarm_Condition tag?

- a) VAR Coil**
- b) FB**
- c) VAR Contact**
- d) CMT**



Thank you for attending

Please consider the resources below:

ABB. (n.d.). *AC500*. <https://new.abb.com/plc/programmable-logic-controllers-plcs/ac500>

LadderLogicWorld. (n.d.). *The basics of how plc architectures work*. <https://ladderlogicworld.com/plc-architecture/>
<https://ladderlogicworld.com/plc-architecture/>

Petruzella, F. D. (2011). *Programmable logic controllers* (4th ed.) McGraw-Hill.

Rockwell Automation. (2012). *Compactlogix-systeem* [PDF]. https://literature.rockwellautomation.com/idc/groups/literature/documents/sg/1769-sg001_-en-p.pdf

Sehr, M.A, Lohstroh, M., Weber, M., Ugaide, I., Witte, M., Neidig, J., Hoeme, S., Niknami, M., & Lee, E.A. (2021). Programmable logic controllers in the context of industry 4.0. *IEEE Transactions On Industrial Informatics* 17(5), 3523 – 3535.
<https://ieeexplore.ieee.org/document/9134804>

Wilcher, D. (2024). *Understanding industrial controls with an esp32*. GitHub. https://github.com/DWilcher/DesignNews-WebinarCode/blob/main/December_24_Webinar_Code.zip



DesignNews

Thank You

Sponsored by

DigiKey

