https://www.halvorsen.blog

LabVIEW LINX and Raspberry Pi

LabVIEW + LabVIEW LINX Toolkit + Raspberry Pi

Hans-Petter Halvorsen

Contents

- This Tutorial shows how we can use **Raspberry Pi** in combination with the **LabVIEW** Programming environment
- **LabVIEW LINX Toolkit** is an add-on for LabVIEW which makes it possible to program the Raspberry Pi device using LabVIEW
- In that way we can create Data Logging Applications, etc. without the need of an expensive DAQ device
- If you don’t have “LabVIEW Professional” Software, you may use the “LabVIEW Community Edition” (free for non-commercial use). You then get a very low-cost DAQ/Datalogging System!

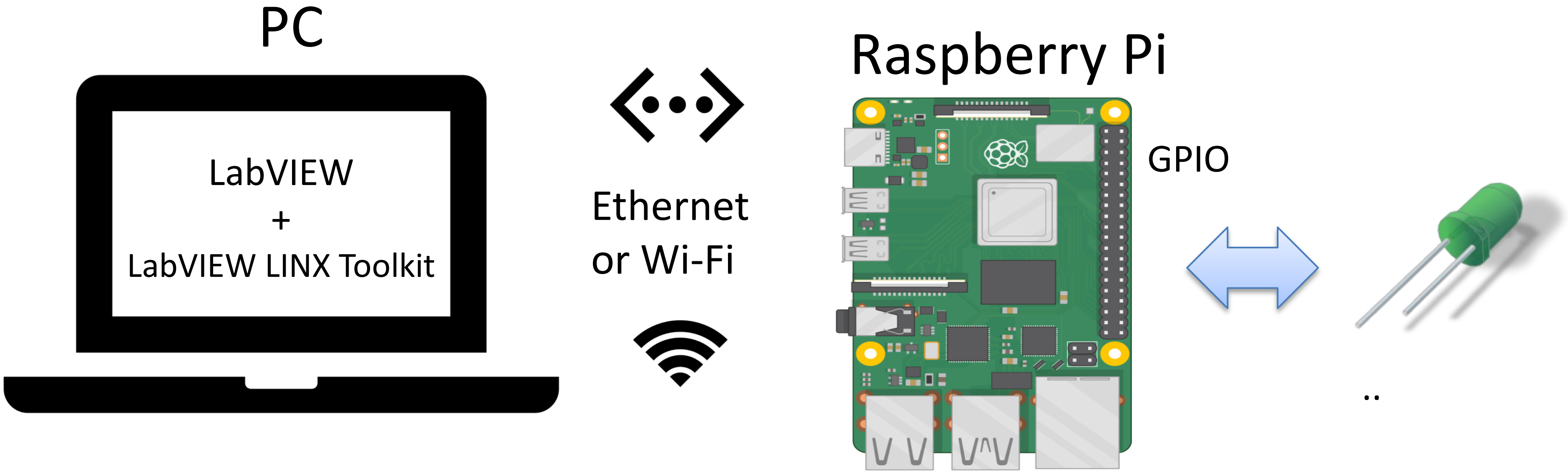
Table of Contents

- Introduction
  - LabVIEW
- **LabVIEW LINX Toolkit**
- Raspberry Pi
  - **Raspberry Pi and LabVIEW LINX Configuration**
  - DAQ System
  - Raspberry PI GPIO
- **LabVIEW Examples**
  - **Digital Out (DO)**
  - **Digital In (DI)**
  - Build and Deploy Executable LabVIEW Application running on Raspberry Pi at Startup

Introduction

Table of Contents

LabVIEW + LabVIEW LINX Toolkit



PC

LabVIEW
+
LabVIEW LINX Toolkit

Ethernet
or Wi-Fi

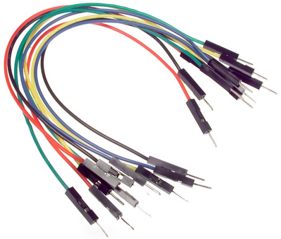
Raspberry Pi

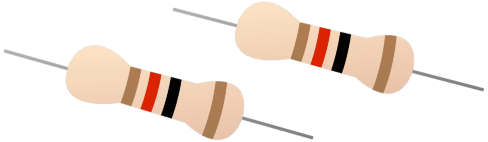
GPIO

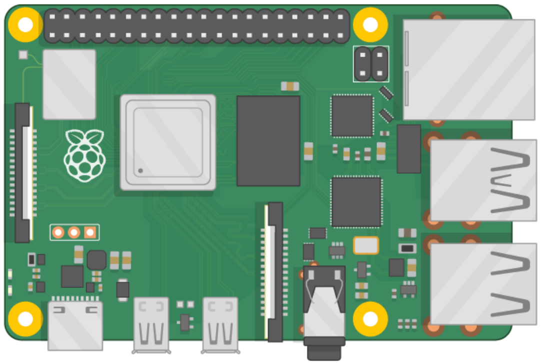
..

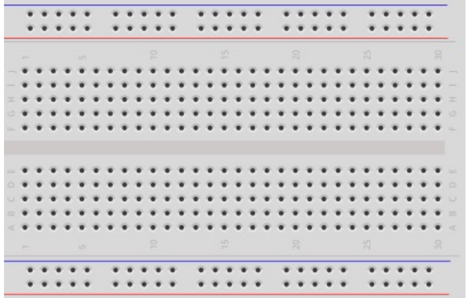
Hardware Components

- Raspberry Pi
- Breadboard
- Wires (Jumper Wires)
- Resistors ($R = 270\Omega$)
- LED









Hardware and Software

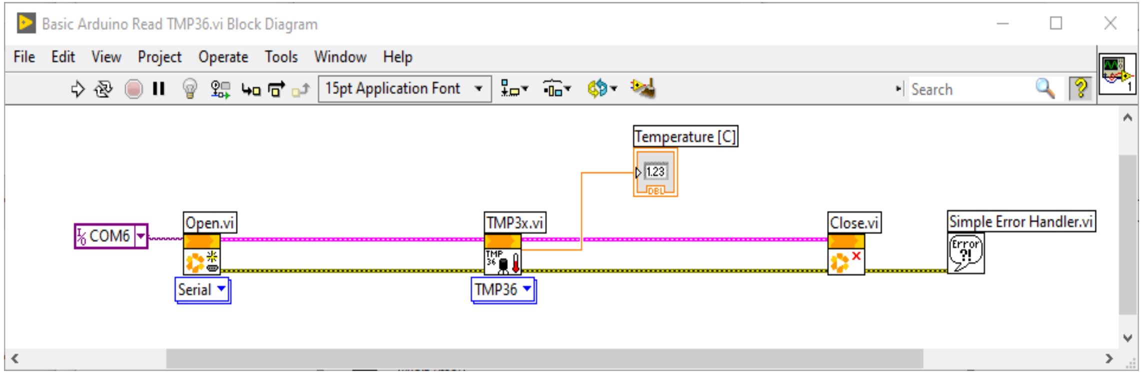
- Host PC (Windows PC)
  - LabVIEW
  - LabVIEW LINX Toolkit
  - (LabVIEW Real-Time Module)
- Raspberry Pi with Raspberry Pi OS
  - Connected to Wi-Fi
  - SSH Enabled

LabVIEW

LabVIEW

- LabVIEW is Graphical Software
- LabVIEW has powerful features for simulation, control and DAQ applications

Basic LabVIEW Example:



LabVIEW LINX Toolkit

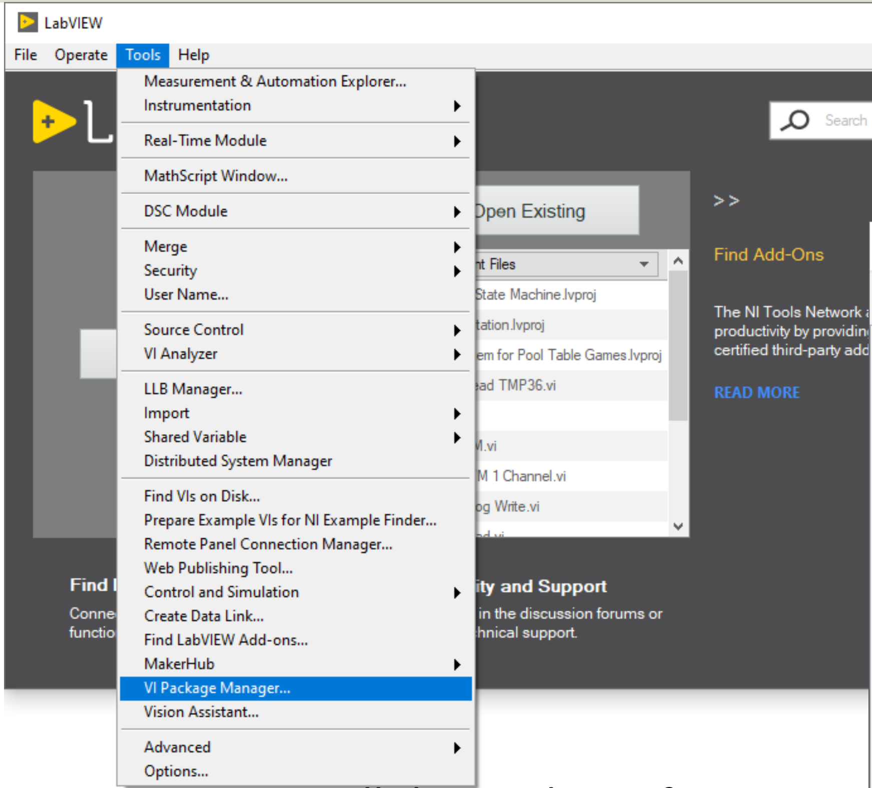
Hans-Petter Halvorsen

Table of Contents

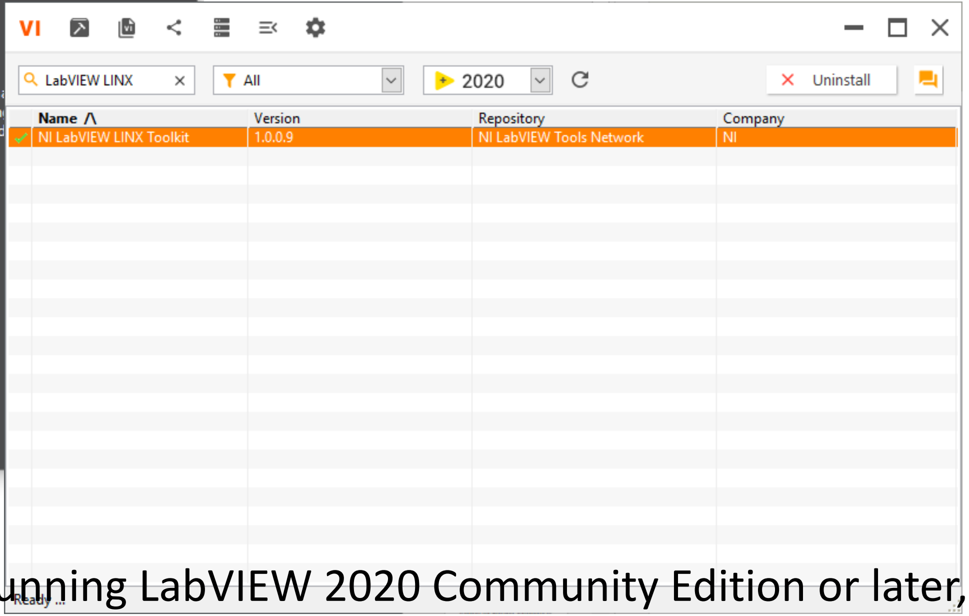
LabVIEW LINX Toolkit

- The LabVIEW LINX Toolkit adds support for Arduino, Raspberry Pi, and BeagleBone embedded platforms
- I have used LabVIEW LINX in combination with Arduino in other Tutorials
- We will use Raspberry Pi in this Tutorial

Installing LabVIEW LINX Toolkit



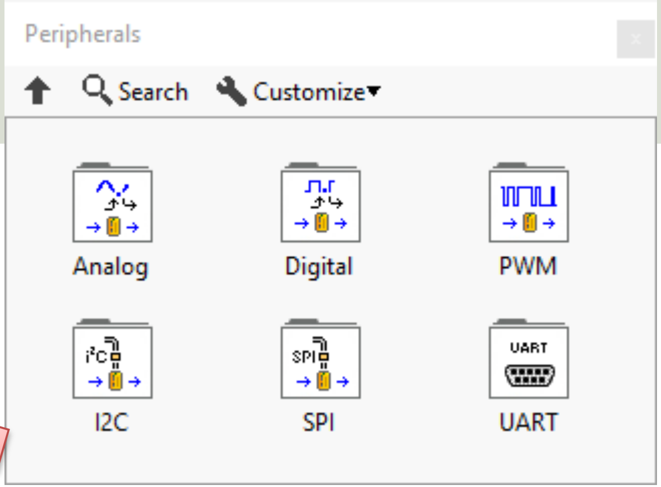
Use VI Package Manger

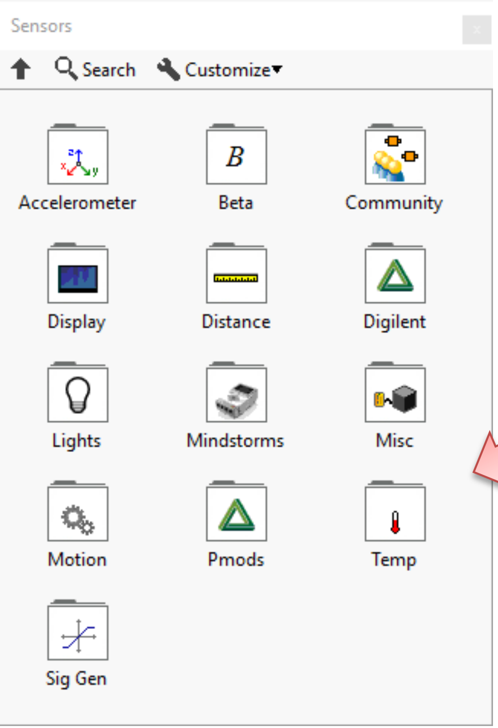


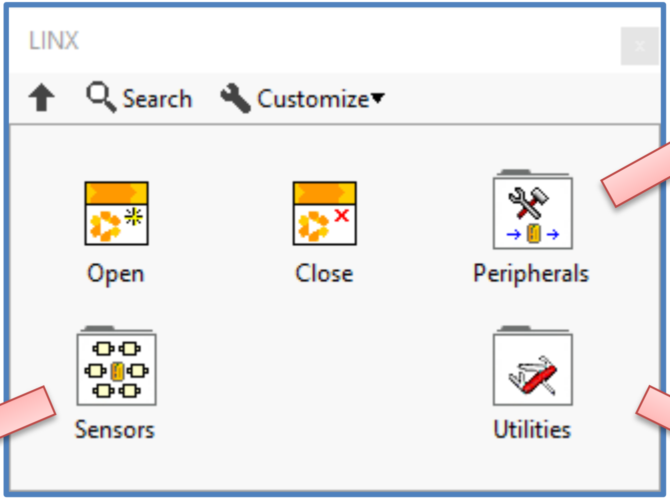
Name /\	Version	Repository	Company
NI LabVIEW LINX Toolkit	1.0.0.9	NI LabVIEW Tools Network	NI

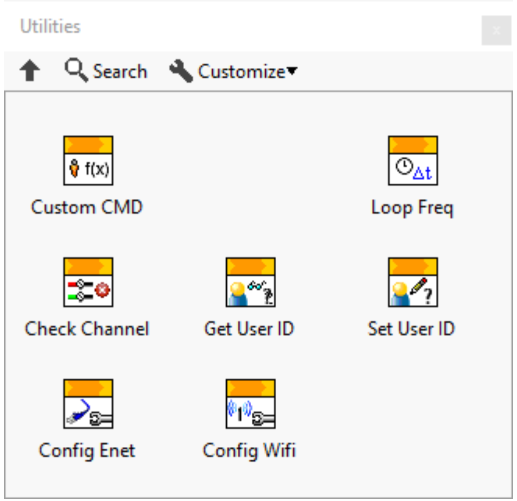
Note: Do not install this package if you are running LabVIEW 2020 Community Edition or later, as the Community Edition already includes the LabVIEW LINX Toolkit

LabVIEW Palette









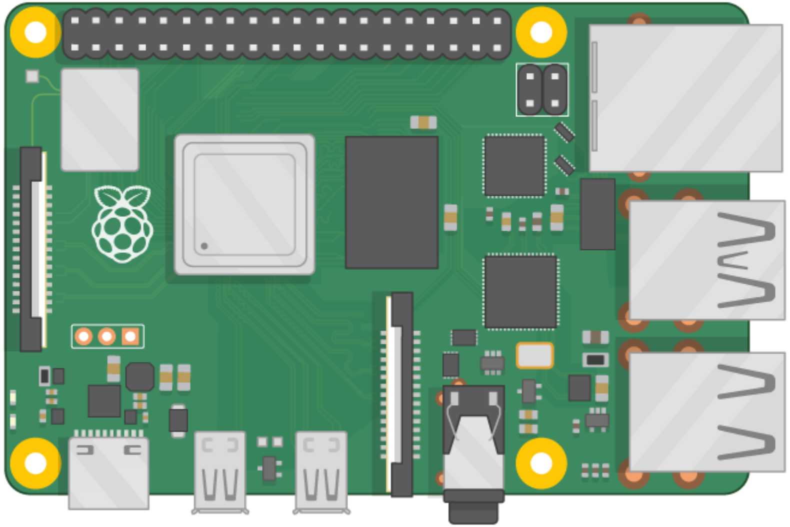
Raspberry Pi

Table of Contents

Raspberry Pi

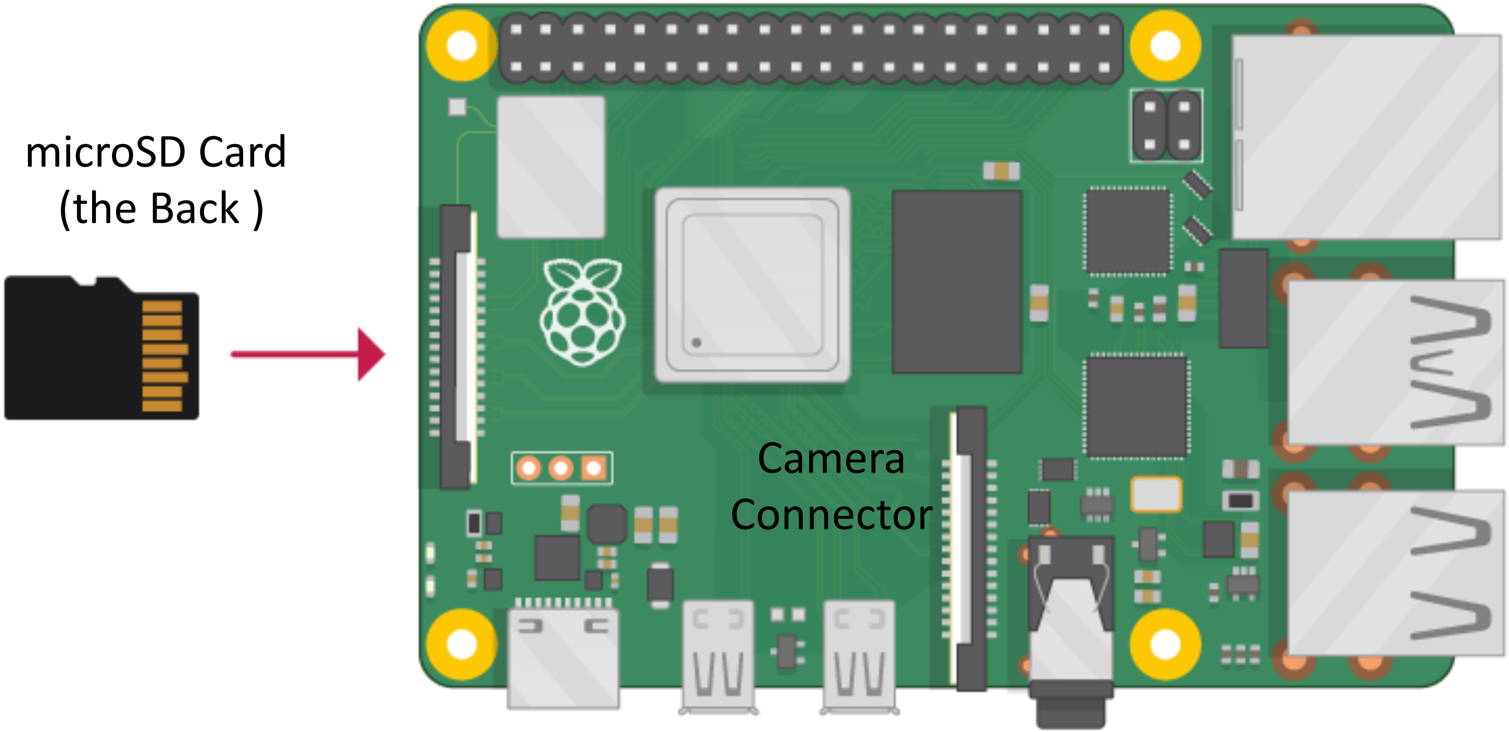
Raspberry Pi is a tiny (about 9x6cm), low-cost ($35+), single-board computer that supports embedded Linux operating systems

The recommended Operating System is called Raspberry Pi OS (Linux based)



https://www.raspberrypi.org

Raspberry Pi



GPIO Pins

microSD Card
(the Back)

Ethernet

Camera
Connector

USB A x 4

Power Supply (USB C)

microHDMI x 2

What Do you Need?

- Raspberry Pi
- microSD Card (+ Adapter)
- Power Supply
- microHDMI to HDMI Cable
- Monitor
- Mouse
- Keyboard
- Ethernet cable or use Wi-Fi

You need this when setting up your Raspberry Pi device

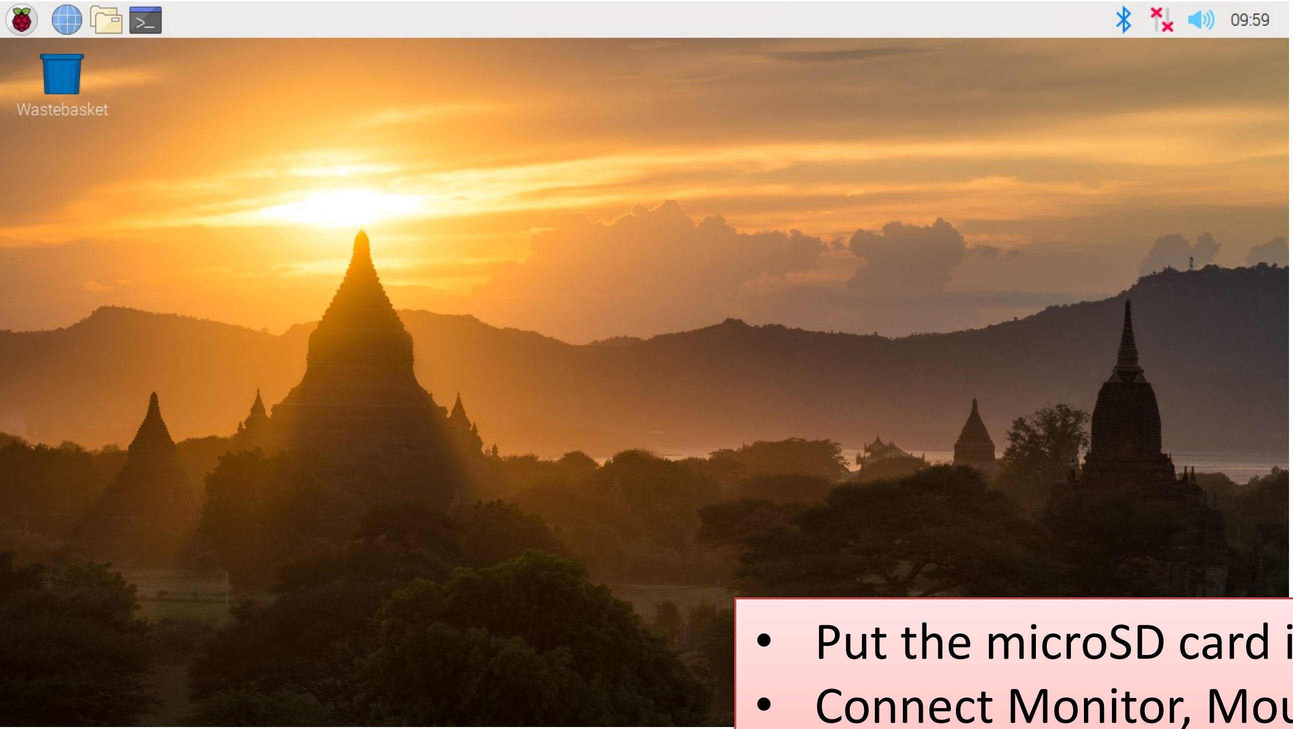
When the Raspberry Pi is configured, and you get access from your PC, you only need the Power Supply (and Ethernet cable if not using Wi-Fi)

Raspberry Pi OS

- In order make your Raspberry Pi up and running you need to install an Operating System (OS)
- The OS for Raspberry Pi is called “Raspberry Pi OS“ (previously known as Raspbian)
- Raspberry Pi runs a version of an operating system called Linux (Windows and macOS are other operating systems).
- To install the necessary OS, you need a microSD card
- Then you use the “Raspberry Pi Imager“ in order to download the OS to the microSD card.

https://www.raspberrypi.org/software/

Start using Raspberry Pi



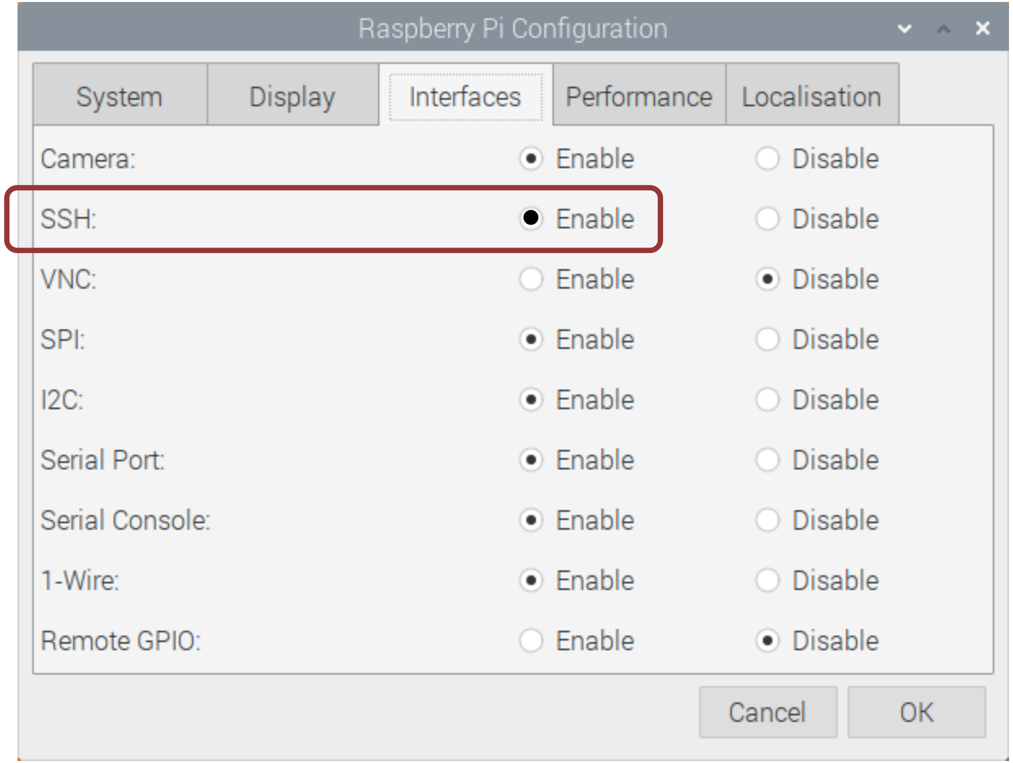
Raspberry Pi OS

- Put the microSD card into the Raspberry Pi
- Connect Monitor, Mouse and Keyboard
- Connect Power Supply
- Follow the Instructions on Screen to setup Wi-Fi

Raspberry Pi Configuration

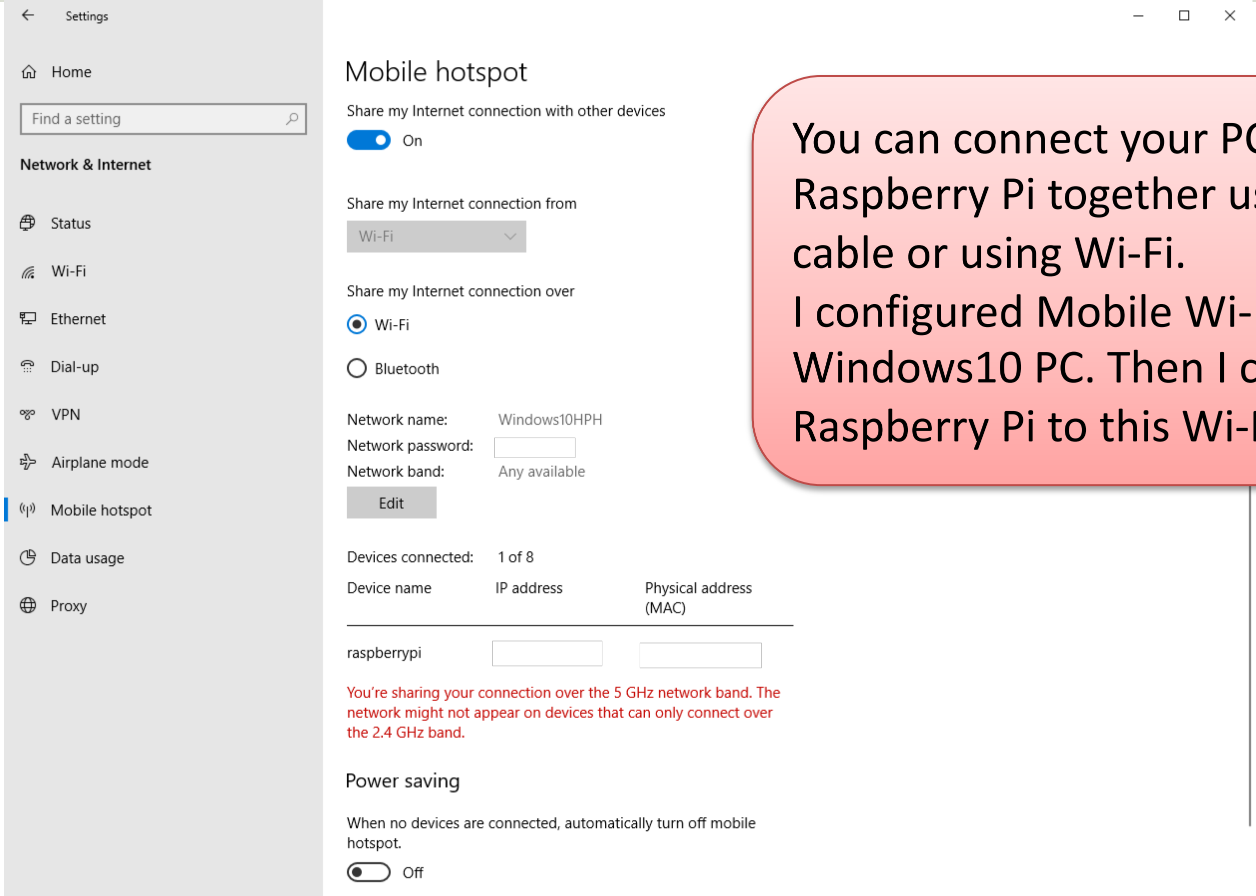
You need to Enable **SSH** so you can remotely get access to the Raspberry Pi from your Computer

SSH, also known as Secure Shell or Secure Socket Shell, is a Network Protocol that gives users, particularly system administrators, a secure way to access a computer over an unsecured network.



Camera:	● Enable	○ Disable
SSH:	● Enable	○ Disable
VNC:	○ Enable	● Disable
SPI:	● Enable	○ Disable
I2C:	● Enable	○ Disable
Serial Port:	● Enable	○ Disable
Serial Console:	● Enable	○ Disable
1-Wire:	● Enable	○ Disable
Remote GPIO:	○ Enable	● Disable

Mobile Wi-Fi hotspot on Windows10



Mobile hotspot

Share my Internet connection with other devices

● On

Share my Internet connection from

Wi-Fi

Share my Internet connection over

● Wi-Fi

○ Bluetooth

Network name: Windows10HPH
Network password:
Network band: Any available

Edit

Devices connected: 1 of 8

Device name	IP address	Physical address (MAC)
raspberrypi		

You're sharing your connection over the 5 GHz network band. The network might not appear on devices that can only connect over the 2.4 GHz band.

Power saving

When no devices are connected, automatically turn off mobile hotspot.

○ Off

You can connect your PC and the Raspberry Pi together using an Ethernet cable or using Wi-Fi.
I configured Mobile Wi-Fi hotspot on my Windows10 PC. Then I connected my Raspberry Pi to this Wi-Fi network

Resources

Raspberry Pi and Installation of Raspberry Pi OS have been covered in more detail in other available Tutorials.

These Tutorials are available on my Blog and YouTube:

- Raspberry Pi - https://youtu.be/sPZqZDdsrkc
- Raspberry Pi Installation and Remote Access - https://youtu.be/NsxZTQysah8

Blog:

https://www.halvorsen.blog/

YouTube Channel @Industrial IT and Automation

https://www.youtube.com/IndustrialITandAutomation

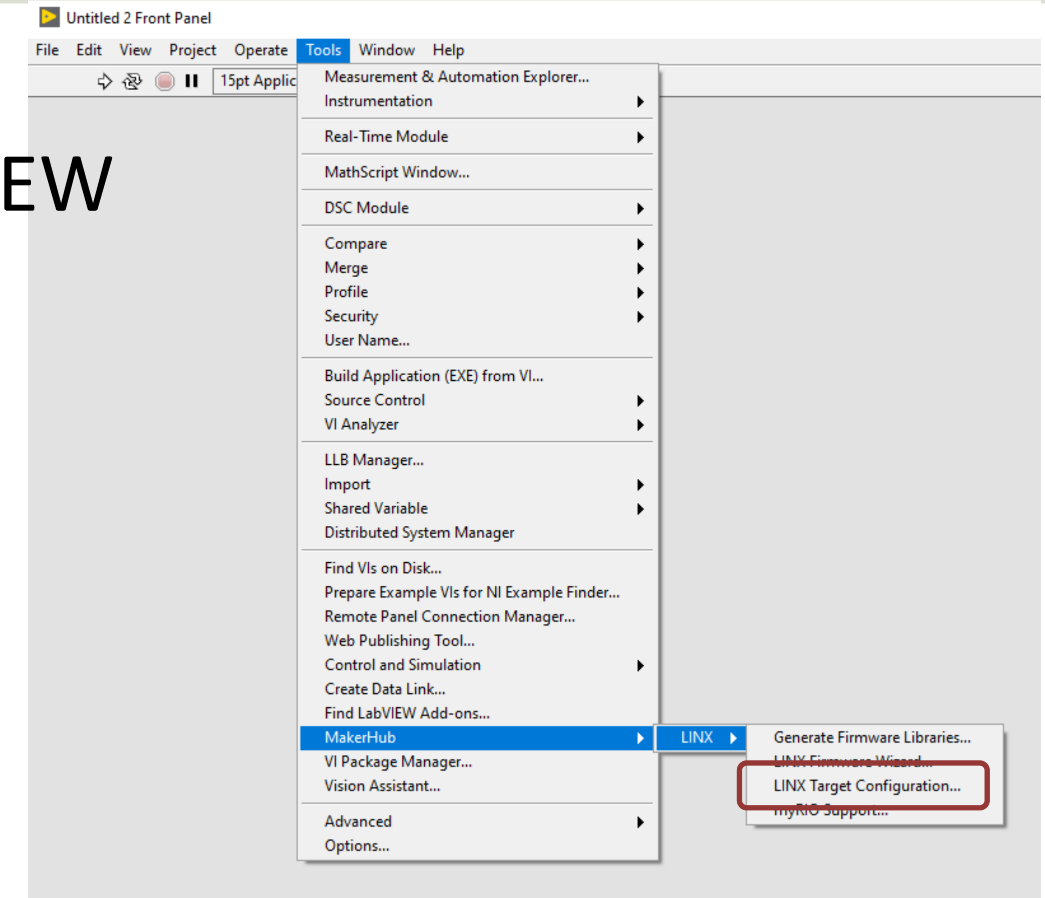
Raspberry Pi and LabVIEW LINX Configuration

Hans-Petter Halvorsen

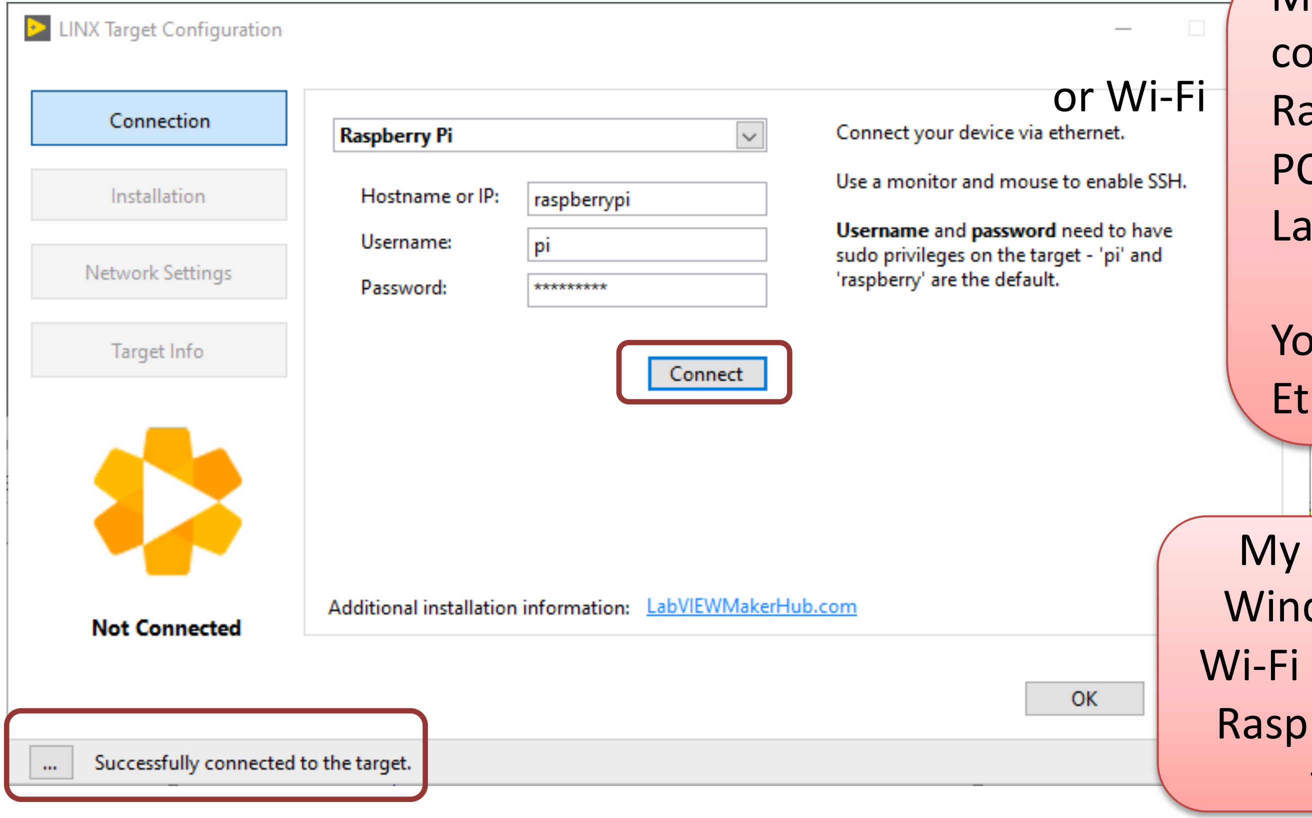
Table of Contents

Raspberry Pi LINX Configuration

LabVIEW



Raspberry Pi LINX Configuration



Raspberry Pi

Hostname or IP: raspberrypi

Username: pi

Password: *********

Connect your device via ethernet. or Wi-Fi

Use a monitor and mouse to enable SSH.

Username and **password** need to have sudo privileges on the target - 'pi' and 'raspberry' are the default.

Additional installation information: LabVIEWMakerHub.com

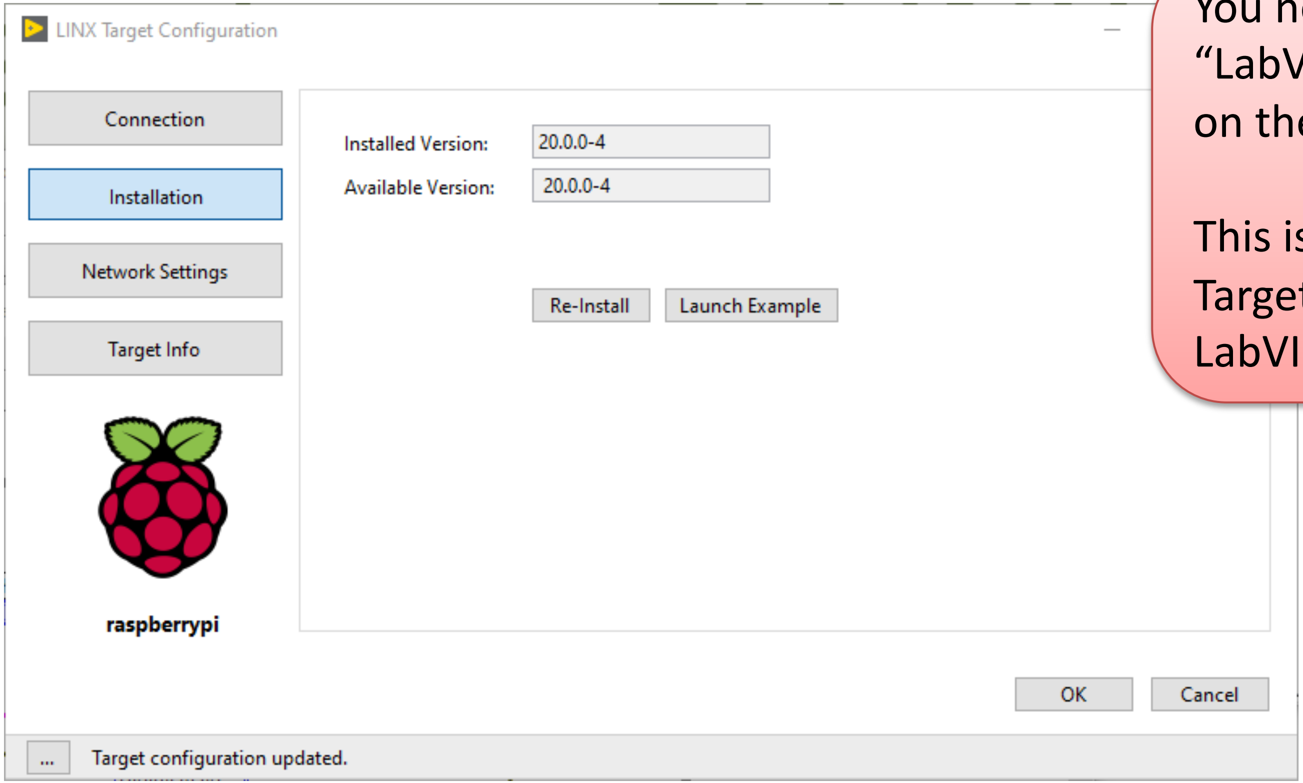
Successfully connected to the target.

Make sure you can connect to the Raspberry Pi from your PC where you have LabVIEW installed.

You can use Wi-Fi or an Ethernet cable

My Configuration: On my Windows PC I configured a Wi-Fi Mobile hotspot. On the Raspberry Pi I connected to this Wi-Fi hotspot

Raspberry Pi LINX Configuration



Installed Version: 20.0.0-4

Available Version: 20.0.0-4

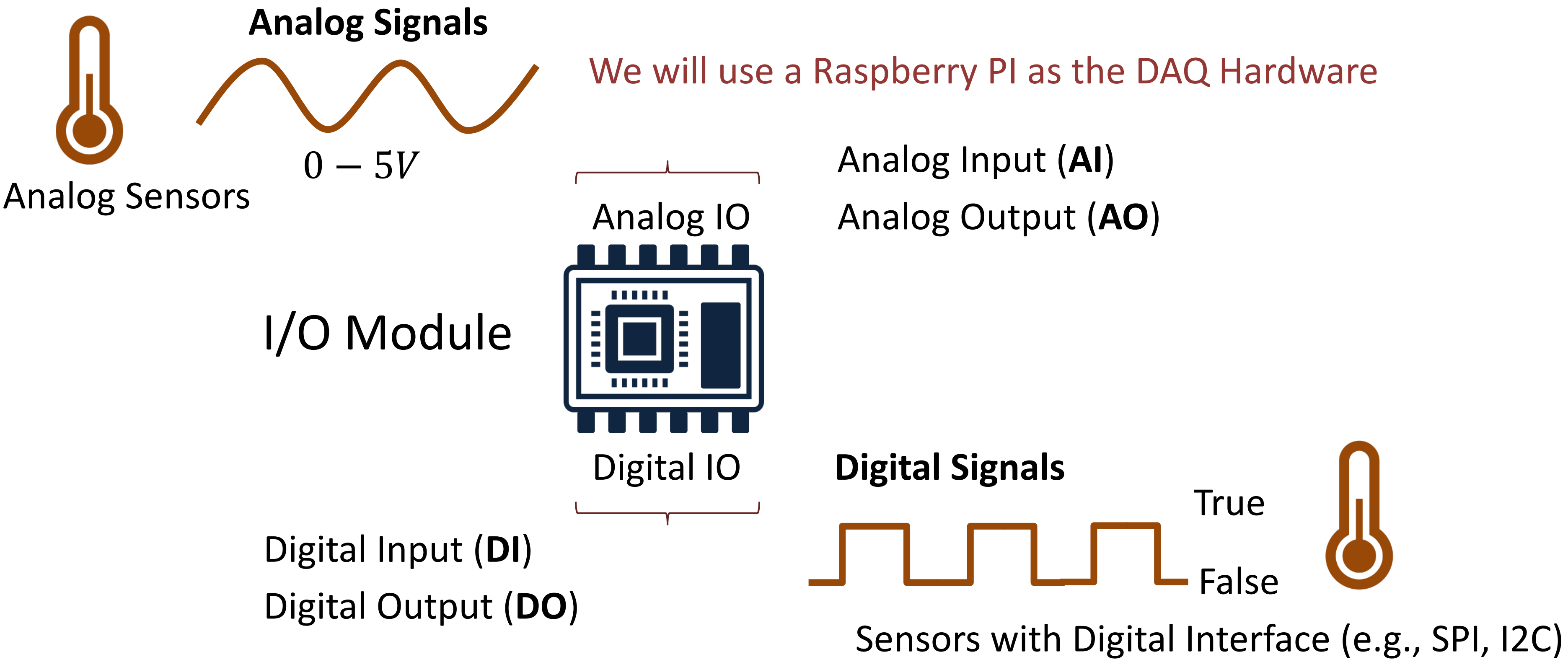
Target configuration updated.

You need to install "LabVIEW Runtime Engine" on the Raspberry Pi device.

This is done from the LINX Target Configuration in LabVIEW on your PC

DAQ System

I/O Module



Analog Signals

$0 - 5V$

Analog Sensors

We will use a Raspberry PI as the DAQ Hardware

Analog IO

Analog Input (**AI**)

Analog Output (**AO**)

I/O Module

Digital IO

Digital Signals

True

False

Digital Input (**DI**)

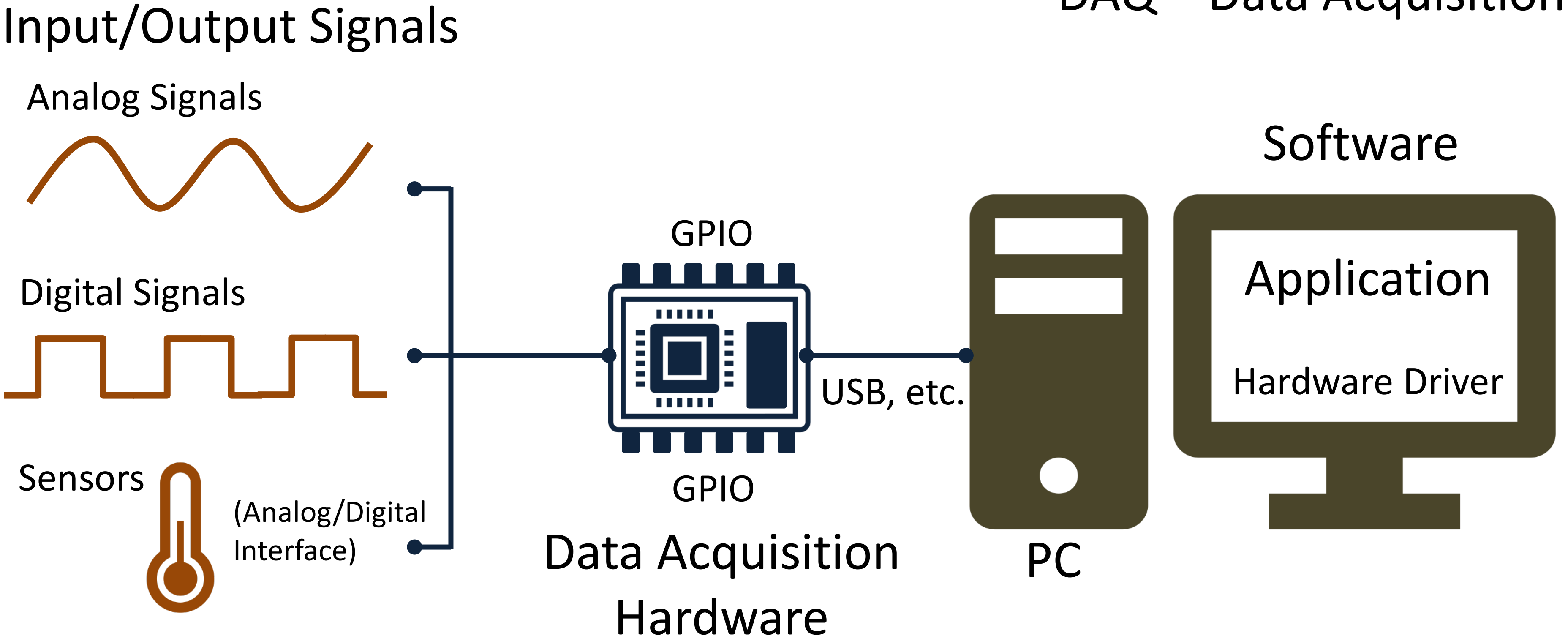
Digital Output (**DO**)

Sensors with Digital Interface (e.g., SPI, I2C)

DAQ System

Raspberry Pi has NO Analog pins!

DAQ – Data Acquisition

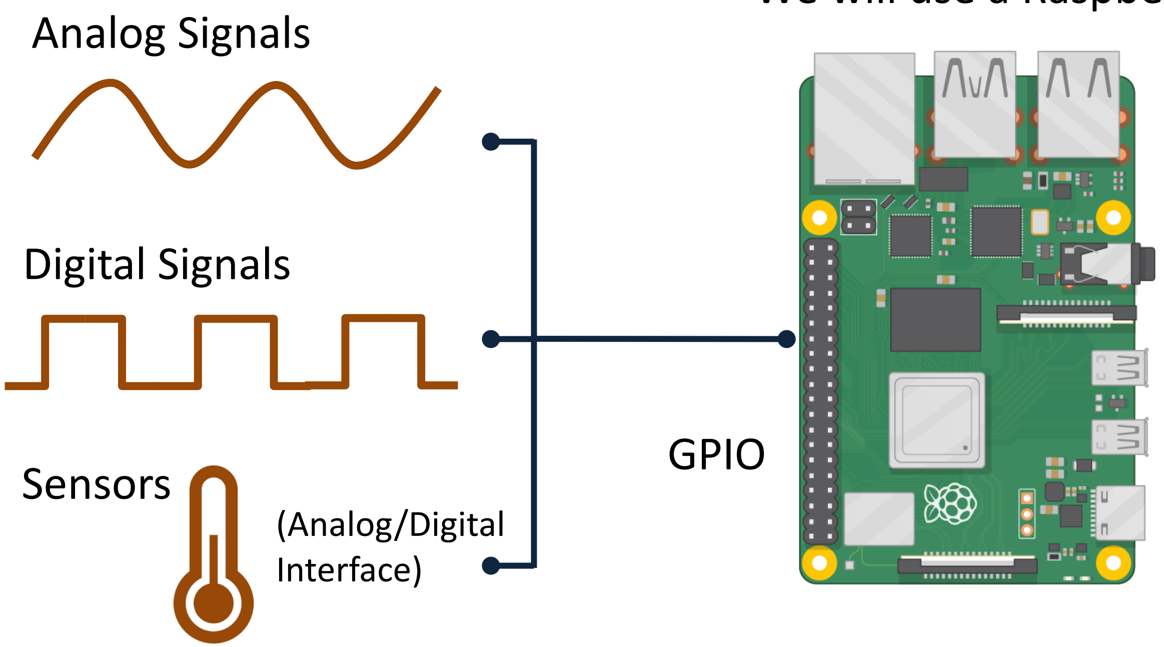


We will use a Raspberry PI as the DAQ Hardware

Final Raspberry Pi DAQ System

Input/Output Signals

We will use a Raspberry PI as the DAQ Hardware



Analog Signals

Digital Signals

Sensors

(Analog/Digital Interface)

GPIO

Raspberry PI running Raspberry PI OS and LabVIEW Run-Time System.

LabVIEW Application running on Raspberry Pi at Startup

Raspberry Pi GPIO

Hans-Petter Halvorsen

Table of Contents

GPIO Features

Raspberry Pi has NO Analog pins!

The GPIO pins are **Digital Pins** which are either True (+3.3V) or False (0V). These can be used to turn on/off LEDs, etc.

The Digital Pins can be either Output or Input.

In addition, some of the pins also offer some other Features:

- PWM (Pulse Width Modulation)

Digital Buses (for reading data from Sensors, etc.):

- SPI
- I2C

Analog In?

Raspberry Pi has NO Analog pins!

What if we want to connect Analog Sensors like the TMP36 Temperature Sensor?

- You then need to use an external **ADC**. These ADC chips have either **SPI** or **I2C** interface
- Or: You can use a **Digital Sensor** that has either SPI or I2C interface built-in

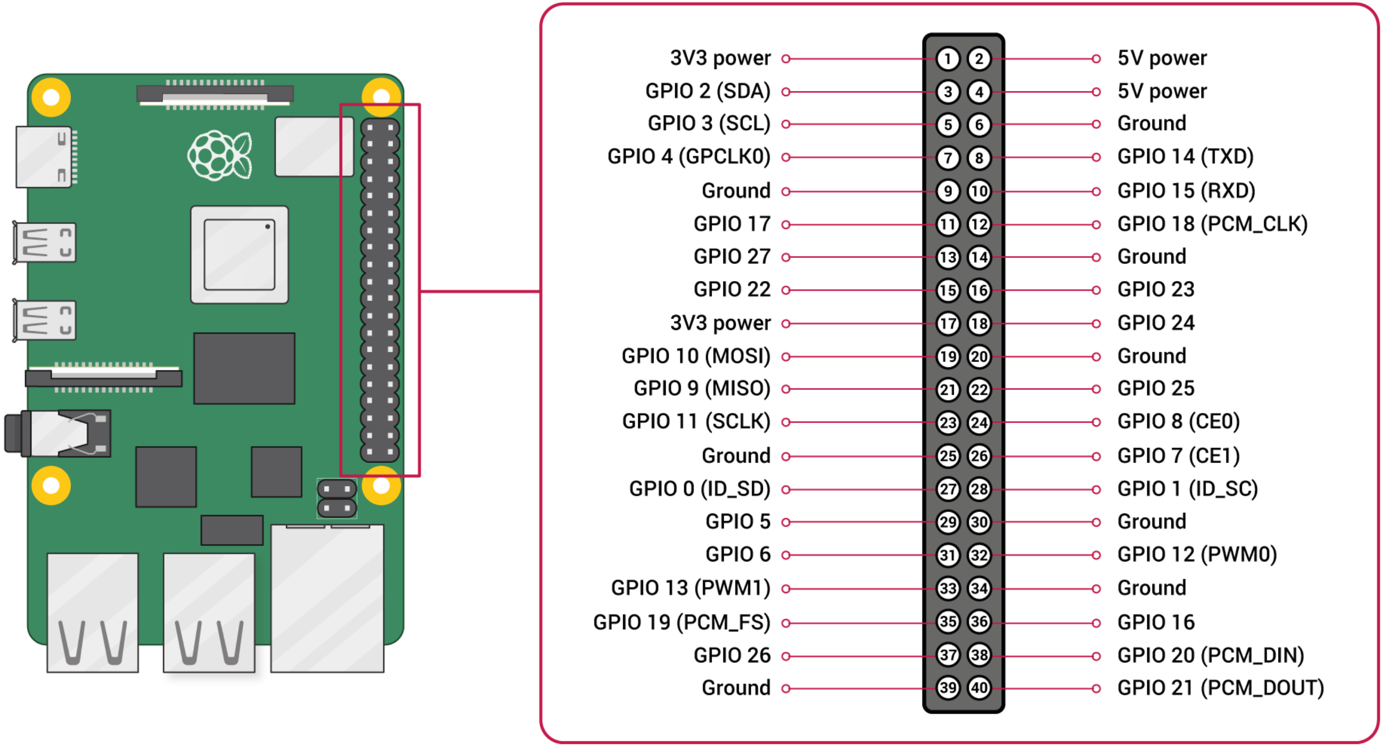
Analog Out?

Raspberry Pi has NO Analog pins!

What if we want to control an external device using an Analog Signal between 0-5V?

- You then need to use an external **DAC**. These DAC chips have either SPI or I2C interface
- Or: Raspberry Pi supports **PWM** (Pulse Width Modulation)
  - PWM can be used to control brightness of a LED, control the speed of a Fan, control a DC Motor, etc.

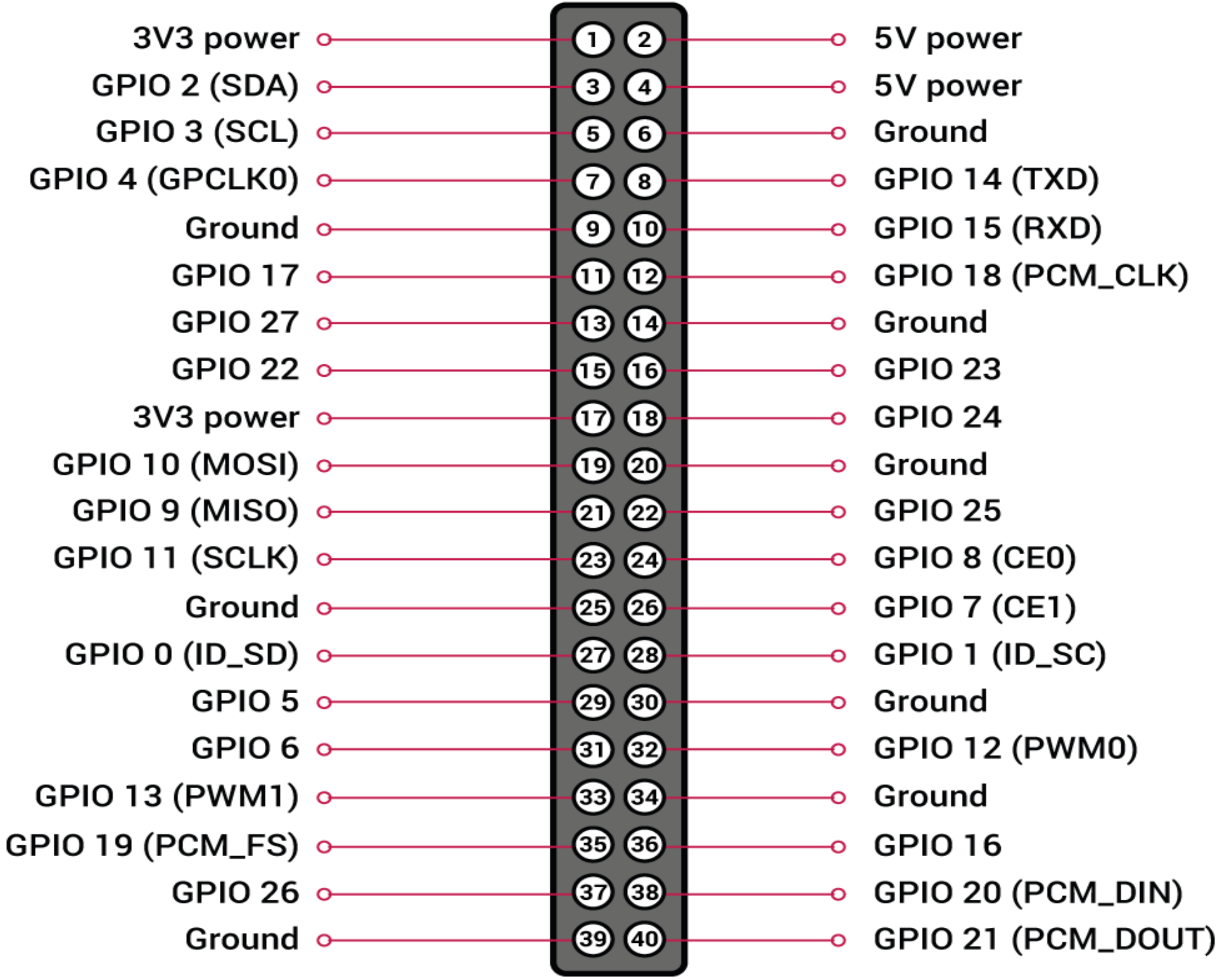
GPIO



3V3 power	1	2	5V power
GPIO 2 (SDA)	3	4	5V power
GPIO 3 (SCL)	5	6	Ground
GPIO 4 (GPCLK0)	7	8	GPIO 14 (TXD)
Ground	9	10	GPIO 15 (RXD)
GPIO 17	11	12	GPIO 18 (PCM_CLK)
GPIO 27	13	14	Ground
GPIO 22	15	16	GPIO 23
3V3 power	17	18	GPIO 24
GPIO 10 (MOSI)	19	20	Ground
GPIO 9 (MISO)	21	22	GPIO 25
GPIO 11 (SCLK)	23	24	GPIO 8 (CE0)
Ground	25	26	GPIO 7 (CE1)
GPIO 0 (ID_SD)	27	28	GPIO 1 (ID_SC)
GPIO 5	29	30	Ground
GPIO 6	31	32	GPIO 12 (PWM0)
GPIO 13 (PWM1)	33	34	Ground
GPIO 19 (PCM_FS)	35	36	GPIO 16
GPIO 26	37	38	GPIO 20 (PCM_DIN)
Ground	39	40	GPIO 21 (PCM_DOUT)

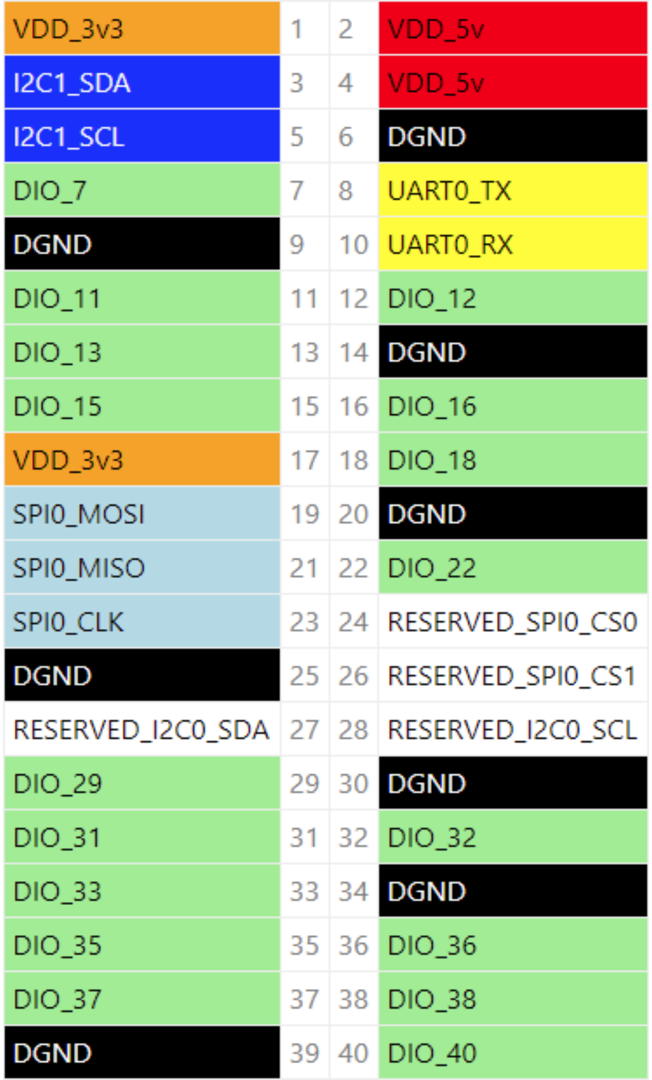
A powerful feature of the Raspberry Pi is the GPIO (general-purpose input/output) pins. The Raspberry Pi has a 40-pin GPIO header as seen in the image

GPIO



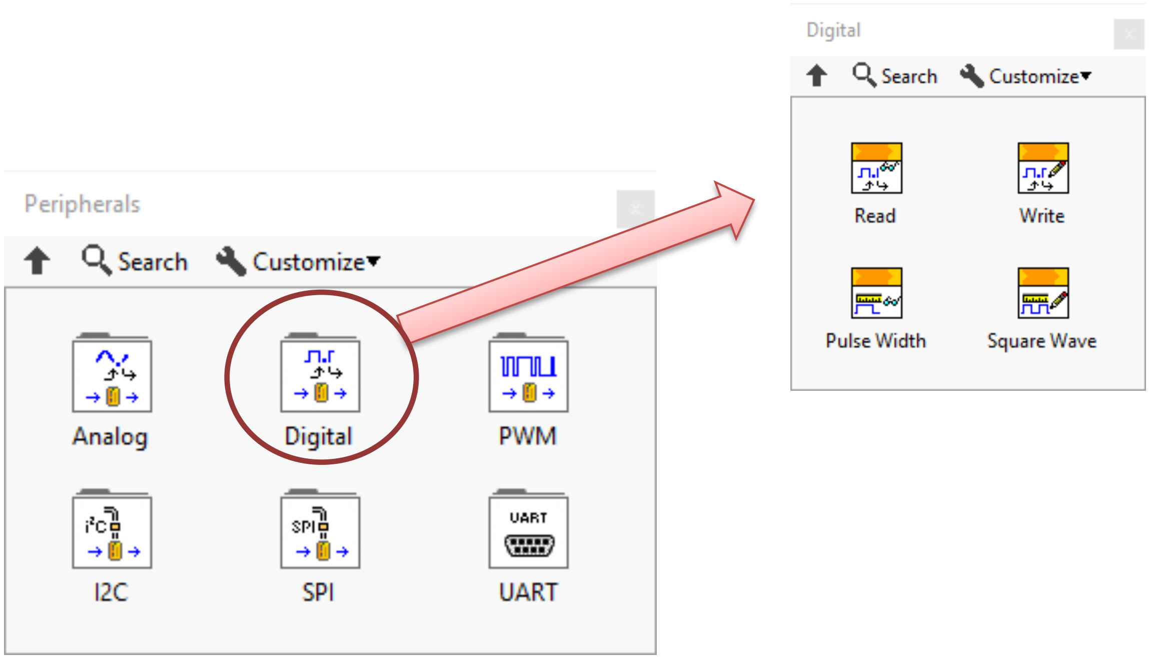
3V3 power	1	2	5V power
GPIO 2 (SDA)	3	4	5V power
GPIO 3 (SCL)	5	6	Ground
GPIO 4 (GPCLK0)	7	8	GPIO 14 (TXD)
Ground	9	10	GPIO 15 (RXD)
GPIO 17	11	12	GPIO 18 (PCM_CLK)
GPIO 27	13	14	Ground
GPIO 22	15	16	GPIO 23
3V3 power	17	18	GPIO 24
GPIO 10 (MOSI)	19	20	Ground
GPIO 9 (MISO)	21	22	GPIO 25
GPIO 11 (SCLK)	23	24	GPIO 8 (CE0)
Ground	25	26	GPIO 7 (CE1)
GPIO 0 (ID_SD)	27	28	GPIO 1 (ID_SC)
GPIO 5	29	30	Ground
GPIO 6	31	32	GPIO 12 (PWM0)
GPIO 13 (PWM1)	33	34	Ground
GPIO 19 (PCM_FS)	35	36	GPIO 16
GPIO 26	37	38	GPIO 20 (PCM_DIN)
Ground	39	40	GPIO 21 (PCM_DOUT)

GPIO



VDD_3v3	1	2	VDD_5v
I2C1_SDA	3	4	VDD_5v
I2C1_SCL	5	6	DGND
DIO_7	7	8	UART0_TX
DGND	9	10	UART0_RX
DIO_11	11	12	DIO_12
DIO_13	13	14	DGND
DIO_15	15	16	DIO_16
VDD_3v3	17	18	DIO_18
SPI0_MOSI	19	20	DGND
SPI0_MISO	21	22	DIO_22
SPI0_CLK	23	24	RESERVED_SPI0_CS0
DGND	25	26	RESERVED_SPI0_CS1
RESERVED_I2C0_SDA	27	28	RESERVED_I2C0_SCL
DIO_29	29	30	DGND
DIO_31	31	32	DIO_32
DIO_33	33	34	DGND
DIO_35	35	36	DIO_36
DIO_37	37	38	DIO_38
DGND	39	40	DIO_40

LabVIEW Palette – Digital I/O

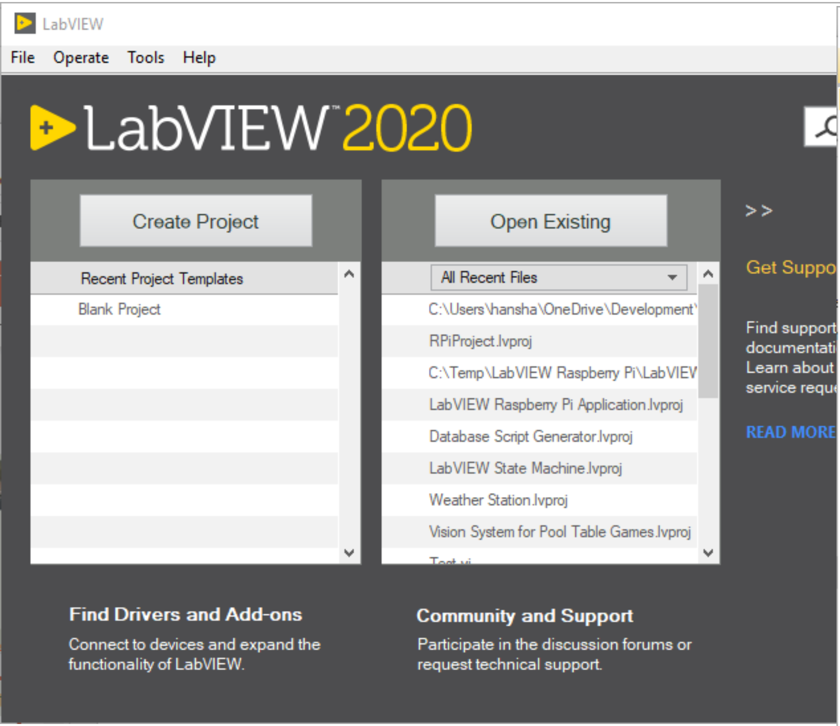


LabVIEW Examples

Hans-Petter Halvorsen

Table of Contents

Create your Raspberry Pi Project



LabVIEW

File Operate Tools Help

LabVIEW 2020

Create Project

Open Existing

Recent Project Templates

- Blank Project

All Recent Files

- C:\Users\hansha\OneDrive\Development\
- RPiProject.lvproj
- C:\Temp\LabVIEW Raspberry Pi\LabVIEW
- LabVIEW Raspberry Pi Application.lvproj
- Database Script Generator.lvproj
- LabVIEW State Machine.lvproj
- Weather Station.lvproj
- Vision System for Pool Table Games.lvproj
- Test.vi

>>

Get Suppo

Find support documentati Learn about service reque

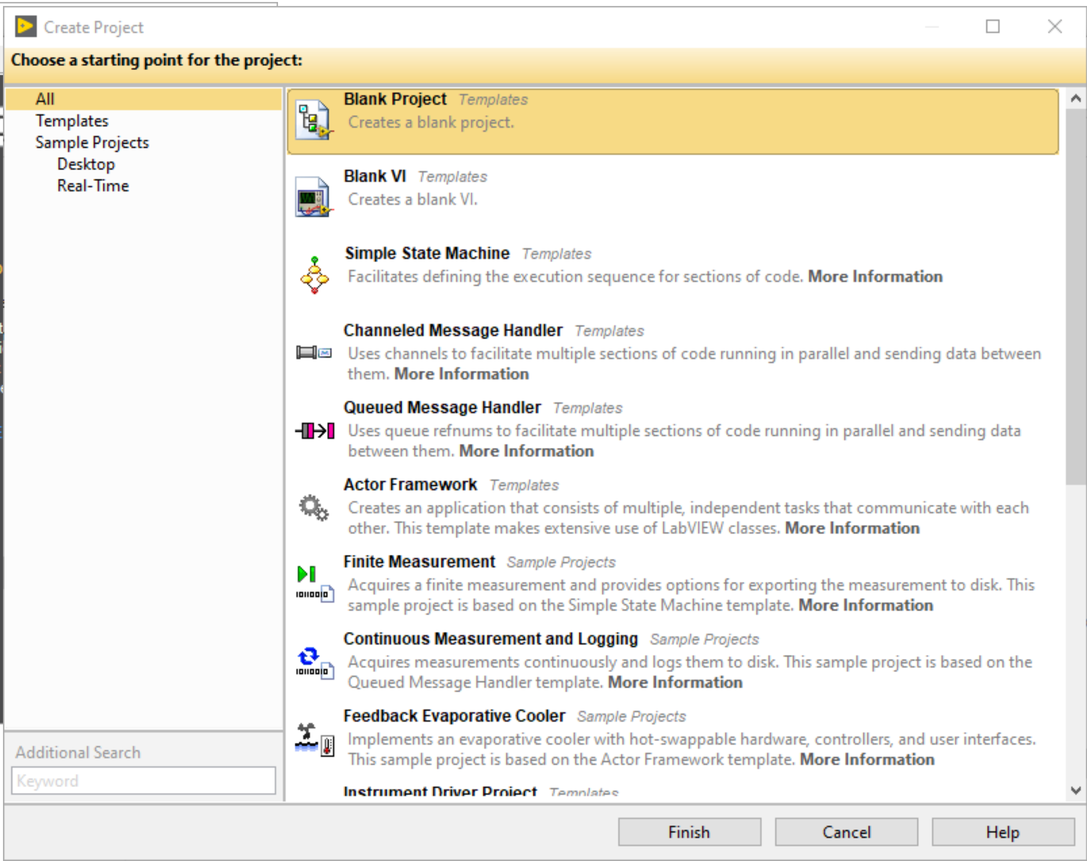
READ MORE

Find Drivers and Add-ons

Connect to devices and expand the functionality of LabVIEW.

Community and Support

Participate in the discussion forums or request technical support.



Create Project

Choose a starting point for the project:

- All
- Templates
- Sample Projects
  - Desktop
  - Real-Time

Blank Project *Templates*
Creates a blank project.

Blank VI *Templates*
Creates a blank VI.

Simple State Machine *Templates*
Facilitates defining the execution sequence for sections of code. **More Information**

Channeled Message Handler *Templates*
Uses channels to facilitate multiple sections of code running in parallel and sending data between them. **More Information**

Queued Message Handler *Templates*
Uses queue refnums to facilitate multiple sections of code running in parallel and sending data between them. **More Information**

Actor Framework *Templates*
Creates an application that consists of multiple, independent tasks that communicate with each other. This template makes extensive use of LabVIEW classes. **More Information**

Finite Measurement *Sample Projects*
Acquires a finite measurement and provides options for exporting the measurement to disk. This sample project is based on the Simple State Machine template. **More Information**

Continuous Measurement and Logging *Sample Projects*
Acquires measurements continuously and logs them to disk. This sample project is based on the Queued Message Handler template. **More Information**

Feedback Evaporative Cooler *Sample Projects*
Implements an evaporative cooler with hot-swappable hardware, controllers, and user interfaces. This sample project is based on the Actor Framework template. **More Information**

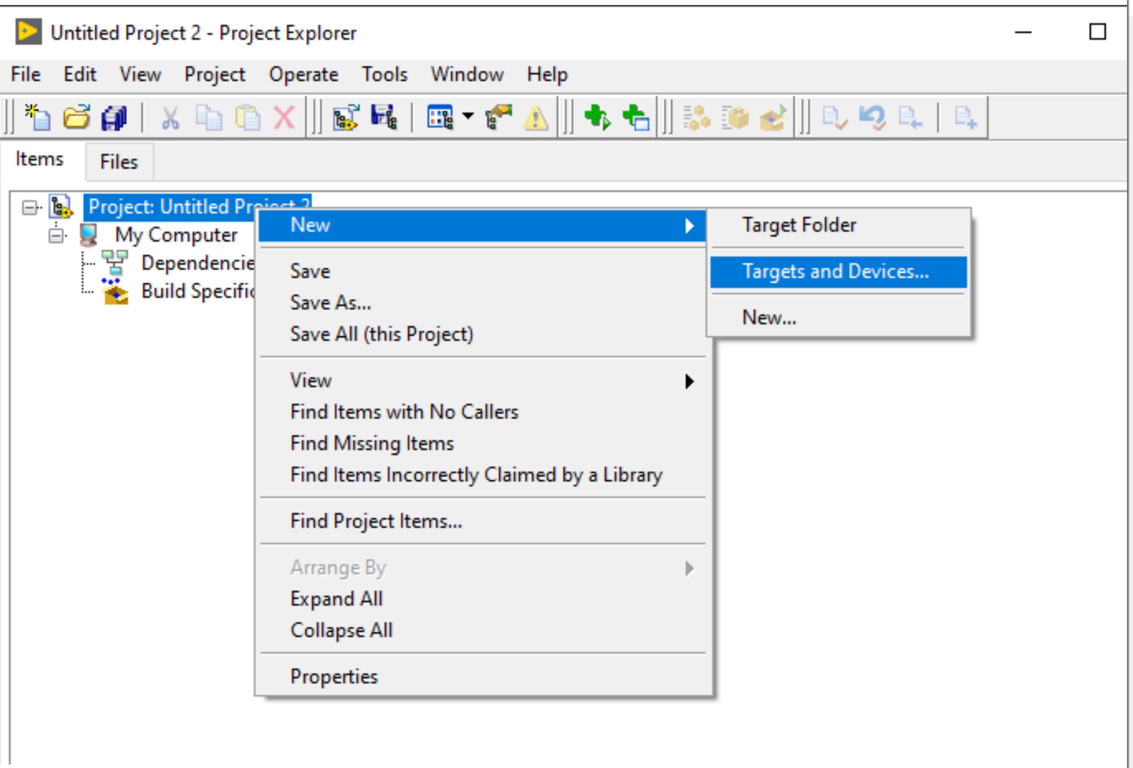
Instrument Driver Project *Templates*

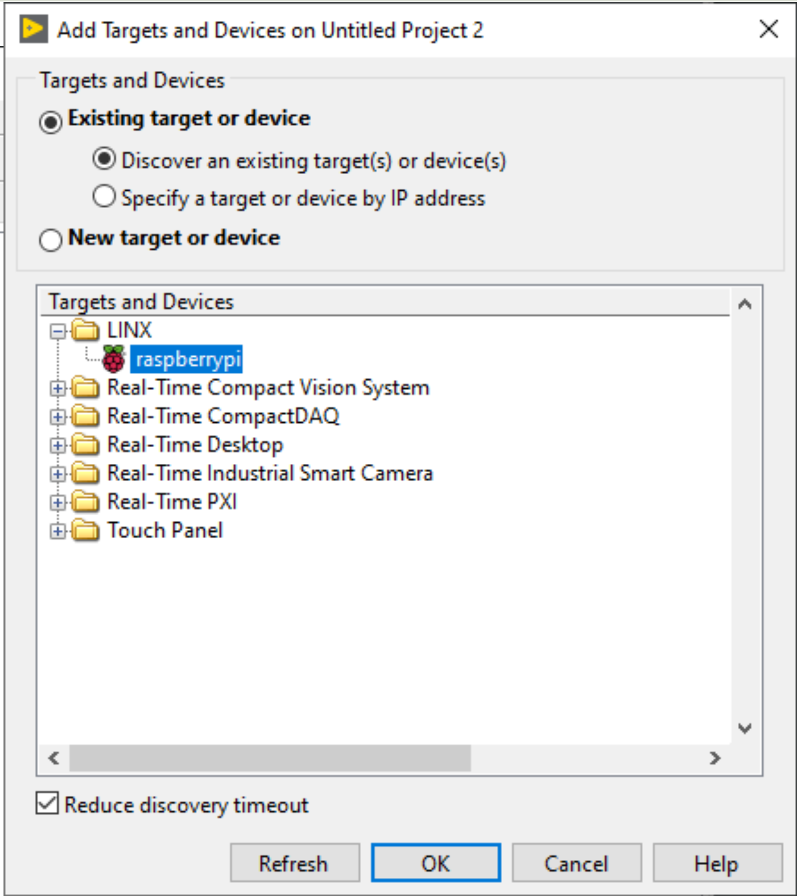
Additional Search

Keyword

Finish Cancel Help

Create your Raspberry Pi Project




Add Targets and Devices on Untitled Project 2

Targets and Devices

● **Existing target or device**

● Discover an existing target(s) or device(s)

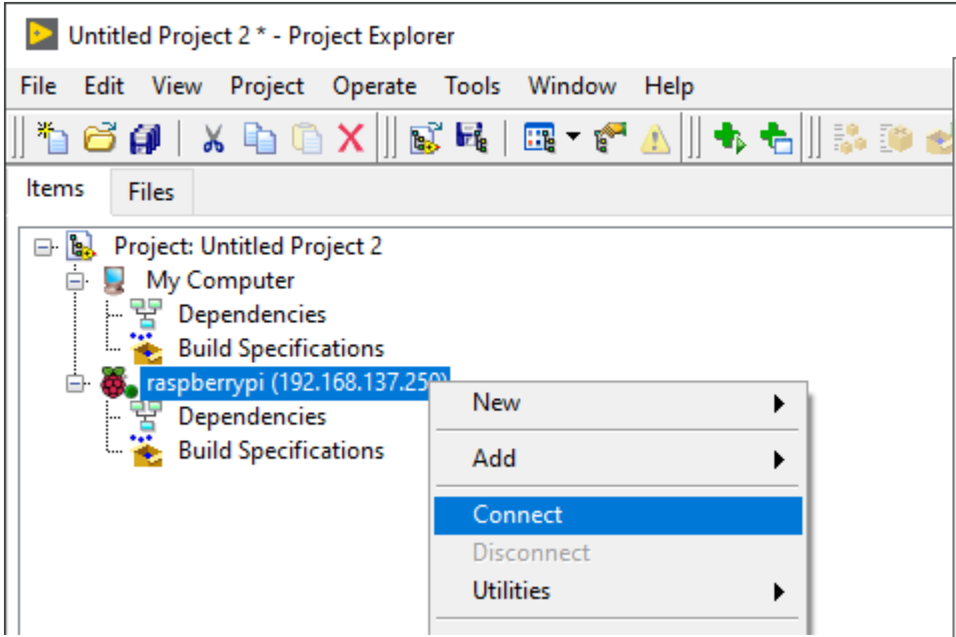
○ Specify a target or device by IP address

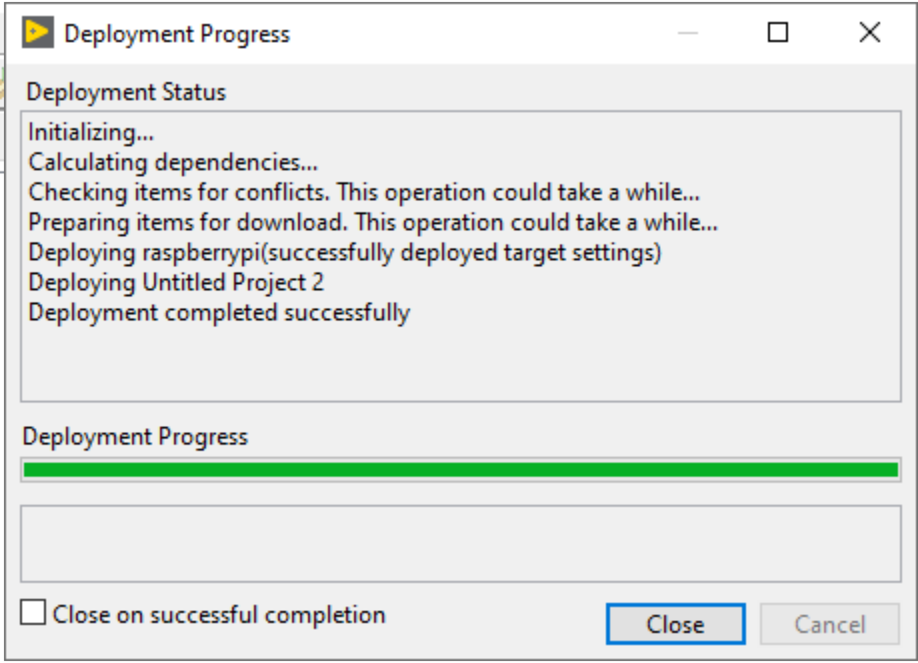
○ **New target or device**

Targets and Devices
- LINX
  - raspberrypi
- Real-Time Compact Vision System
- Real-Time CompactDAQ
- Real-Time Desktop
- Real-Time Industrial Smart Camera
- Real-Time PXI
- Touch Panel

☑ Reduce discovery timeout

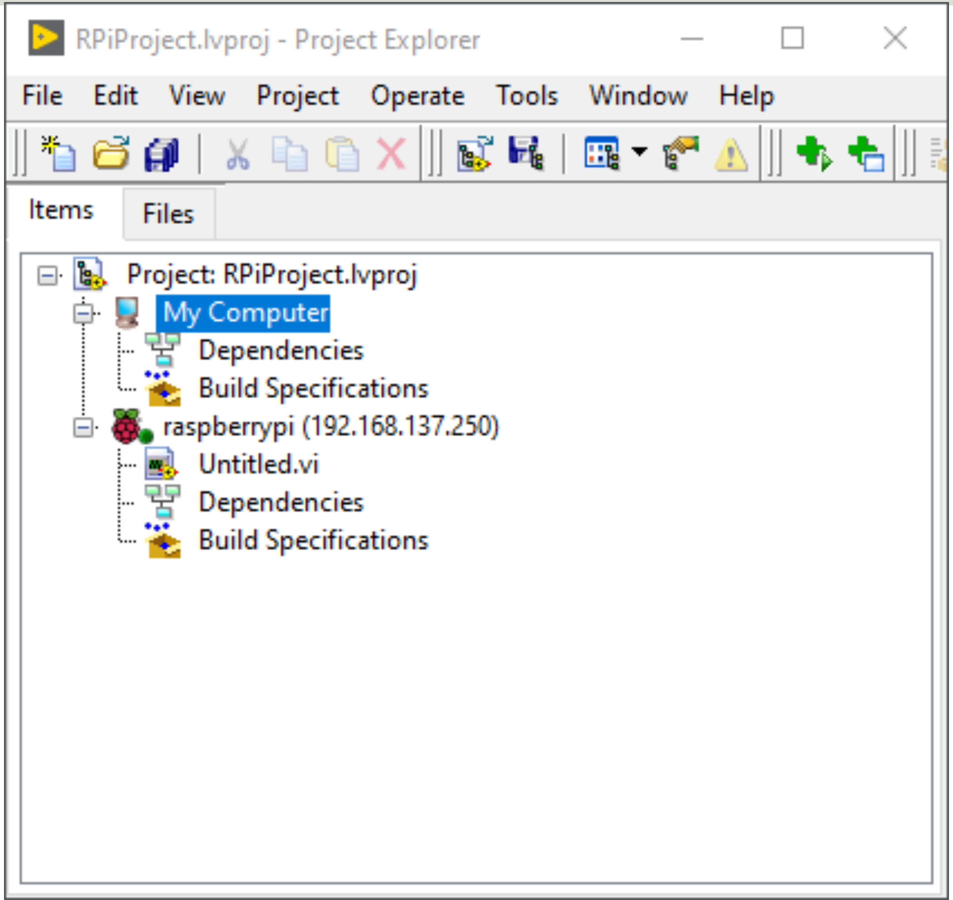
Create your Raspberry Pi Project





```
Initializing...
Calculating dependencies...
Checking items for conflicts. This operation could take a while...
Preparing items for download. This operation could take a while...
Deploying raspberrypi(successfully deployed target settings)
Deploying Untitled Project 2
Deployment completed successfully
```

LabVIEW Project Explorer


RPiProject.lvproj - Project Explorer
File Edit View Project Operate Tools Window Help
Items
Files

Project: RPiProject.lvproj
- My Computer
  - Dependencies
  - Build Specifications
- raspberrypi (192.168.137.250)
  - Untitled.vi
  - Dependencies
  - Build Specifications

You are now ready to start creating LabVIEW Code that control the GPIO pins on the Raspberry Pi device

Digital Out (DO)

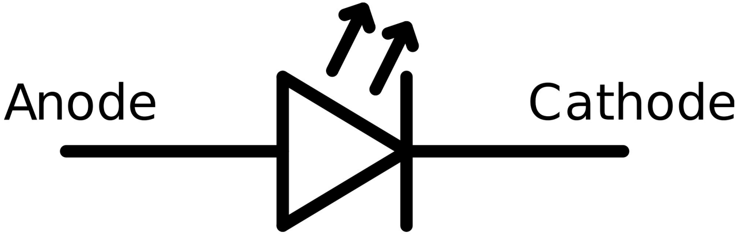
Table of Contents

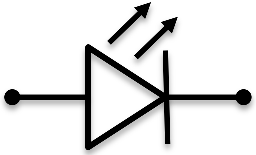
Digital Write/Out (DO)

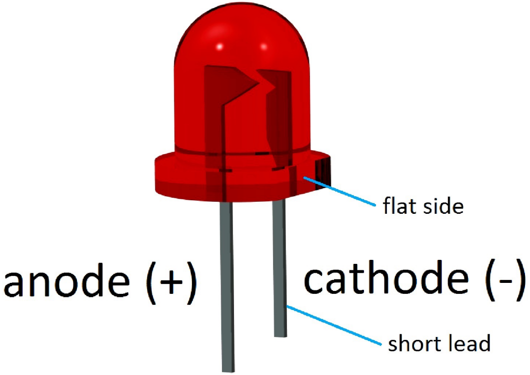
- We will use one of the GPIO (Digital Out/Write pins to turn on/off a LED

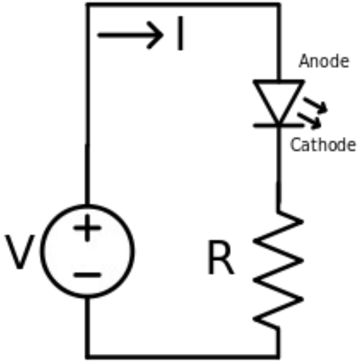
Light-emitting diode - LED

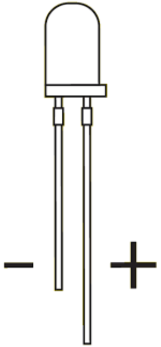
A light-emitting diode (LED) is a semiconductor light source that emits light when current flows through it



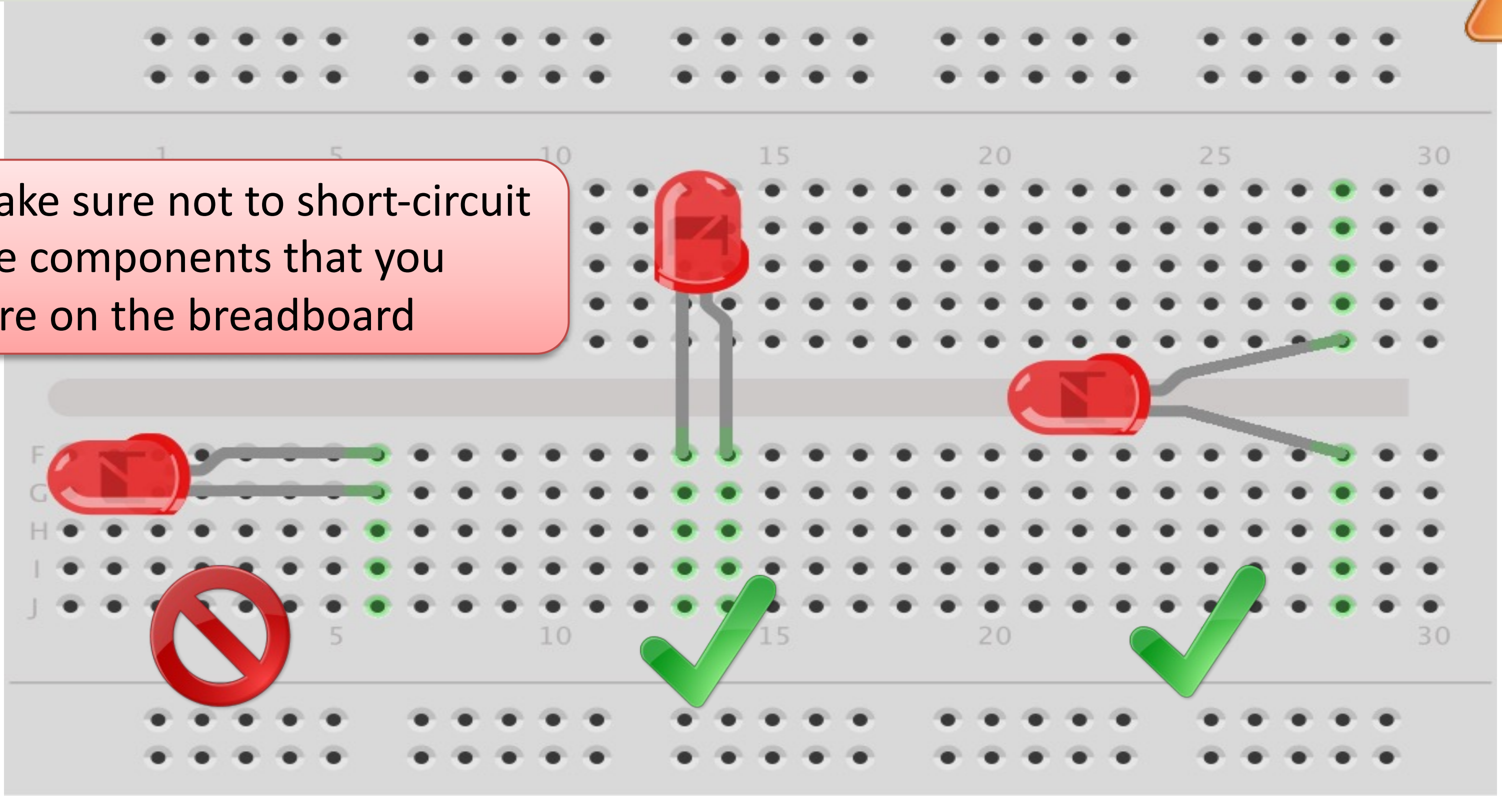








Breadboard Wiring

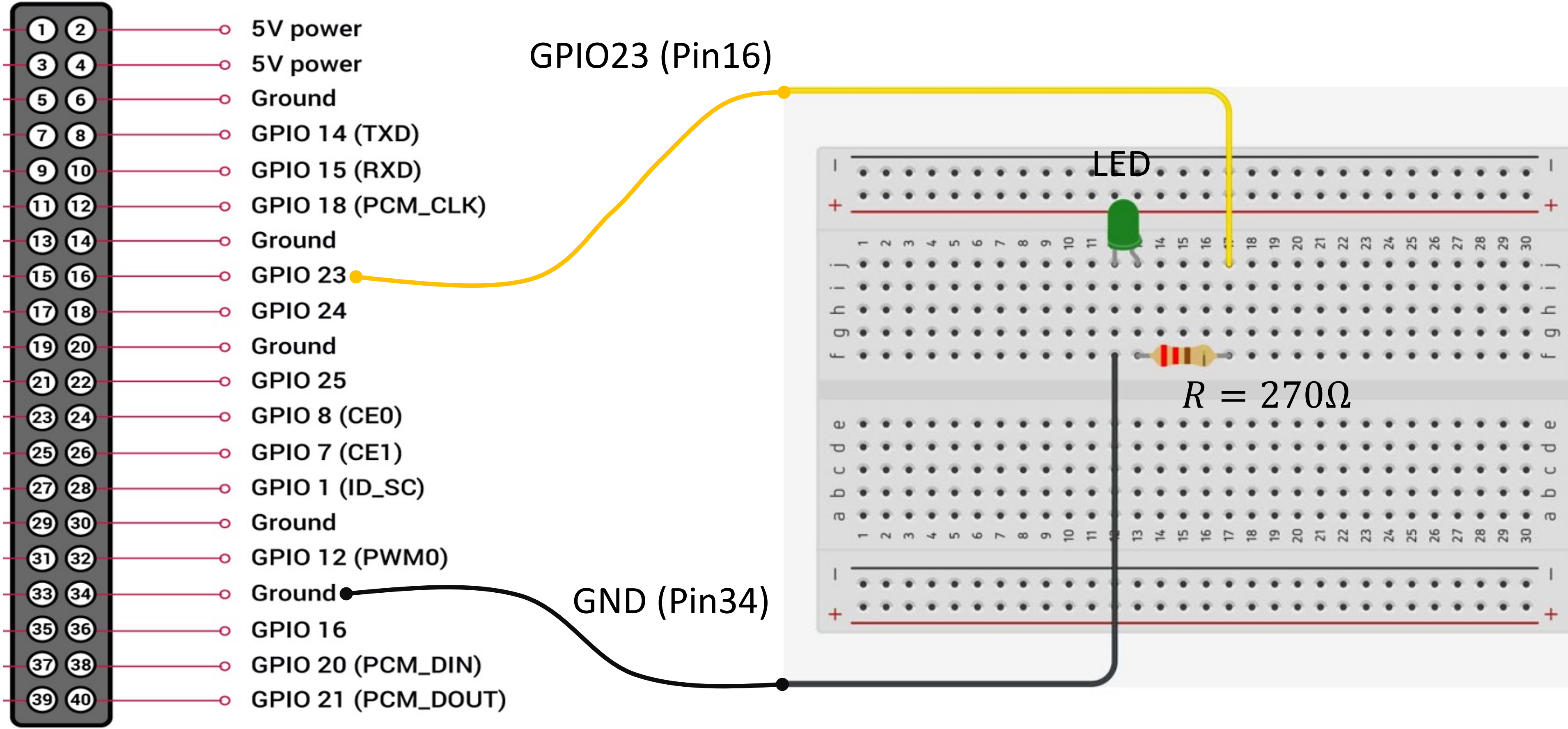


Make sure not to short-circuit the components that you wire on the breadboard

The Breadboard is used to connect components and electrical circuits

fritzing

LED Wiring



1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
5V power
5V power
Ground
GPIO 14 (TXD)
GPIO 15 (RXD)
GPIO 18 (PCM_CLK)
Ground
GPIO 23
GPIO 24
Ground
GPIO 25
GPIO 8 (CE0)
GPIO 7 (CE1)
GPIO 1 (ID_SC)
Ground
GPIO 12 (PWM0)
Ground
GPIO 16
GPIO 20 (PCM_DIN)
GPIO 21 (PCM_DOUT)
GPIO23 (Pin16)
LED
$R = 270\Omega$
GND (Pin34)

Why do you need a Resistor?

If the current becomes too large, the LED will be destroyed. To prevent this to happen, we will use a Resistor to limit the amount of current in the circuit.

What should be the size of the Resistor?

A LED typically need a current like 20mA (can be found in the LED Datasheet).
We use Ohm's Law:

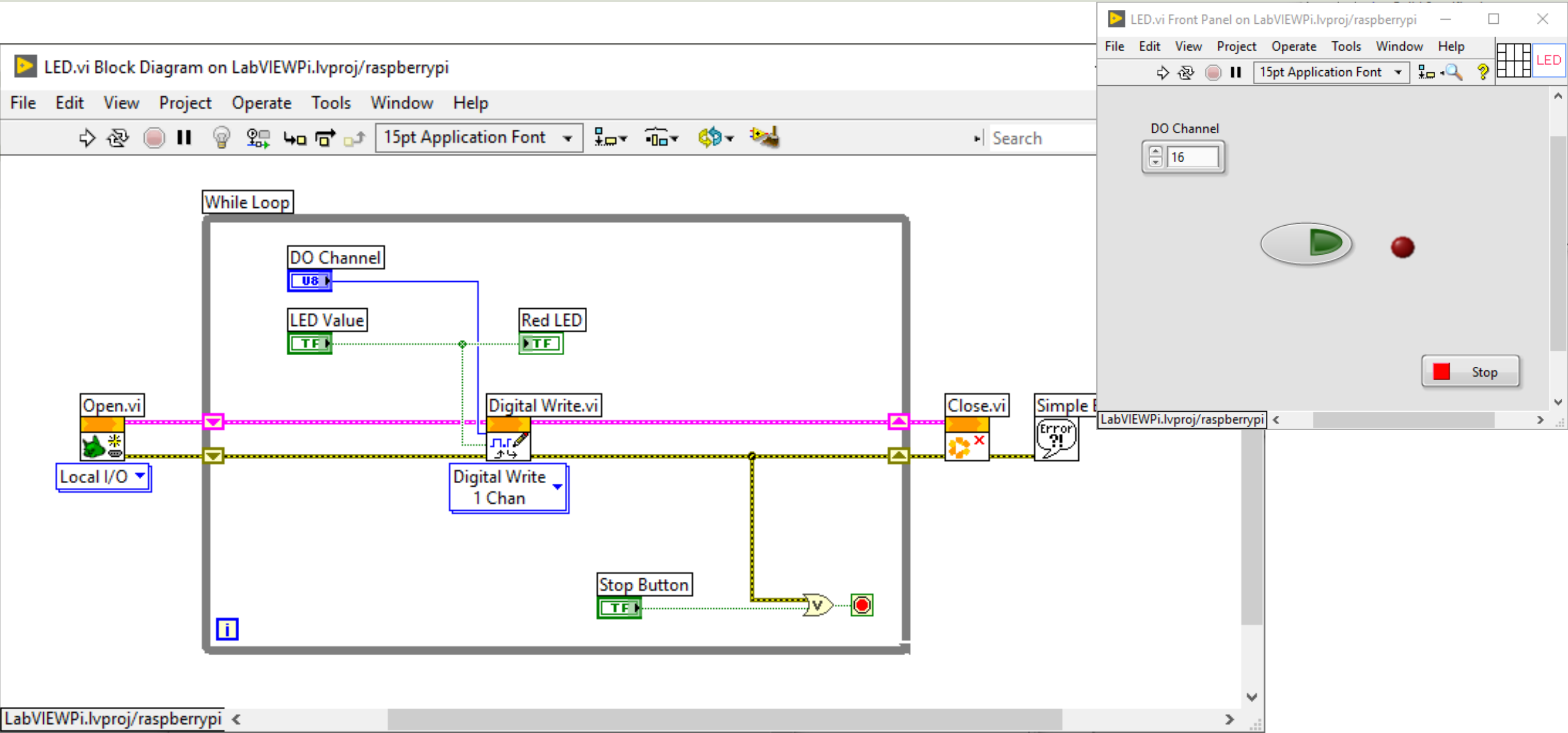
$$U = RI$$

Raspberry Pi gives U=3.3/5V and I=20mA. We then get:

$$R = \frac{U}{I}$$

The Resistor needed will be $R = \frac{5V}{0.02A} = 250\Omega$. Resistors with R=250Ω is not so common, so we can use the closest Resistors we have, e.g., 270Ω

LED ON/OFF - LabVIEW Example



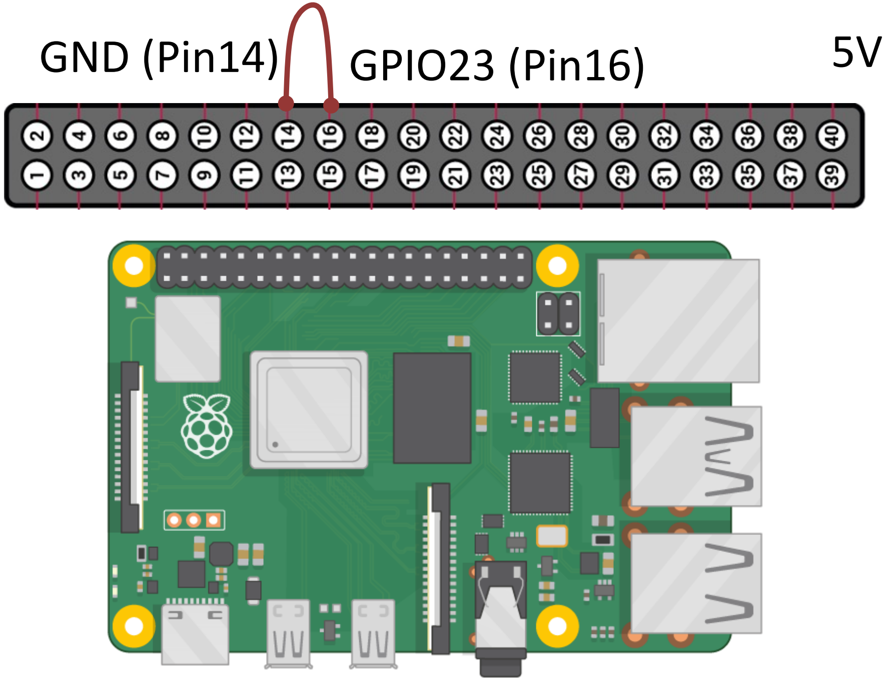
Digital In (DI)

Test of Digital Read

We can test the Digital In (Read) by wiring to GND (False/Low) or 5V (True/High)

GPIO23 (Pin16) is used in this example, but you can of course use another GPIO pin

False/Low

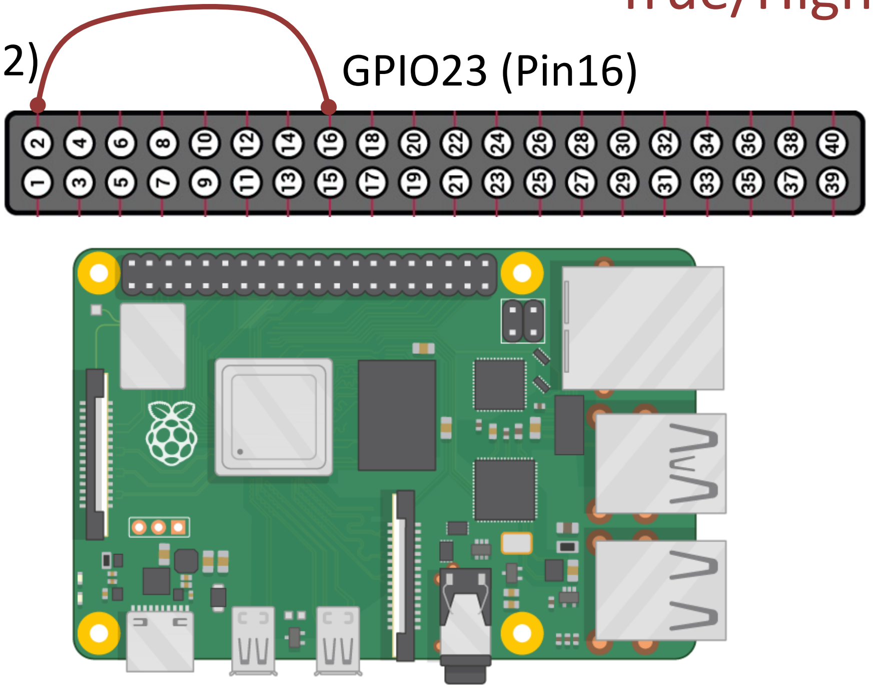


GND (Pin14)

GPIO23 (Pin16)

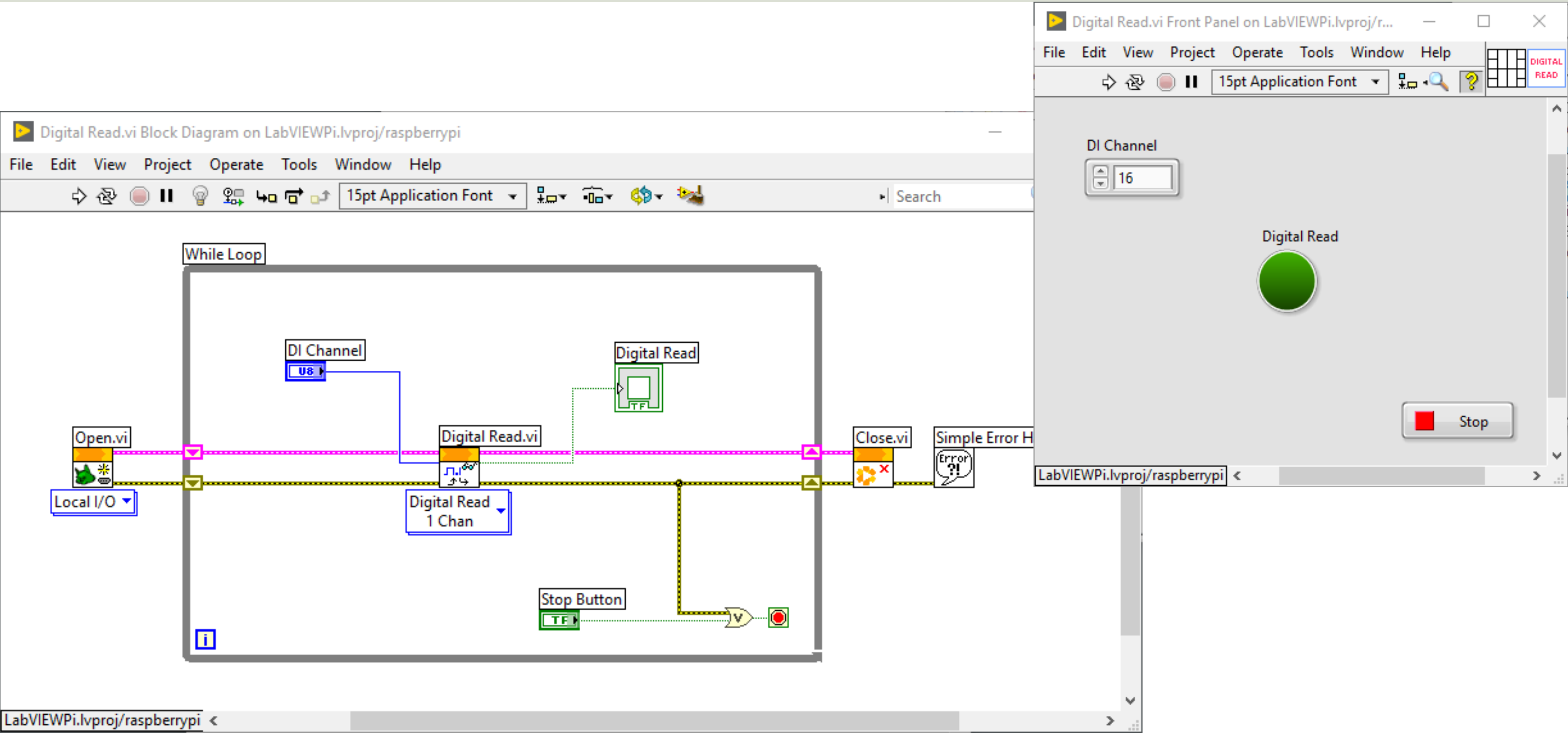
True/High

5V (Pin2)



GPIO23 (Pin16)

LabVIEW - Digital Read

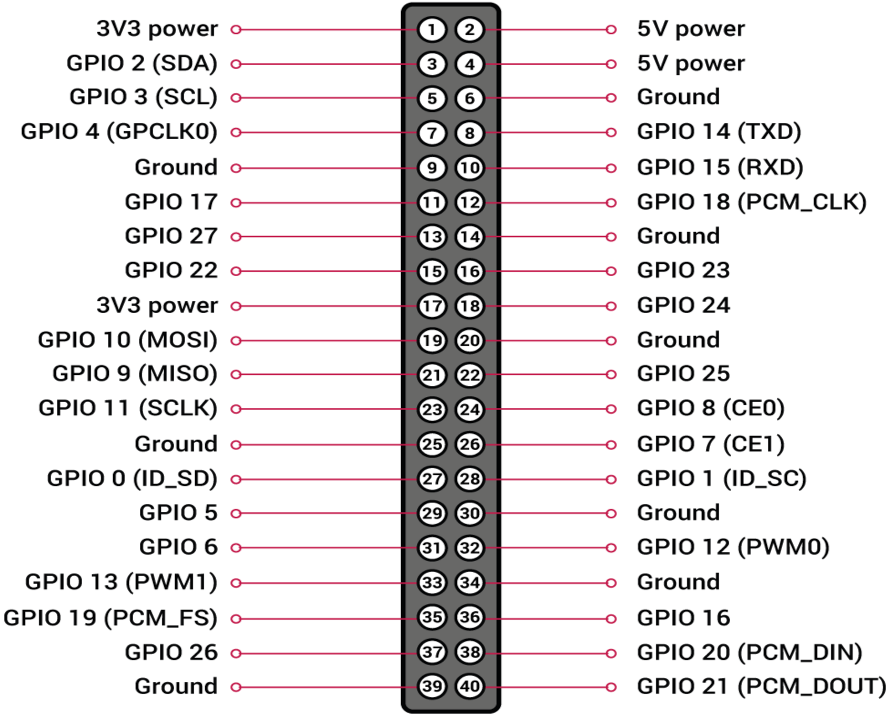


DI Channel: 16

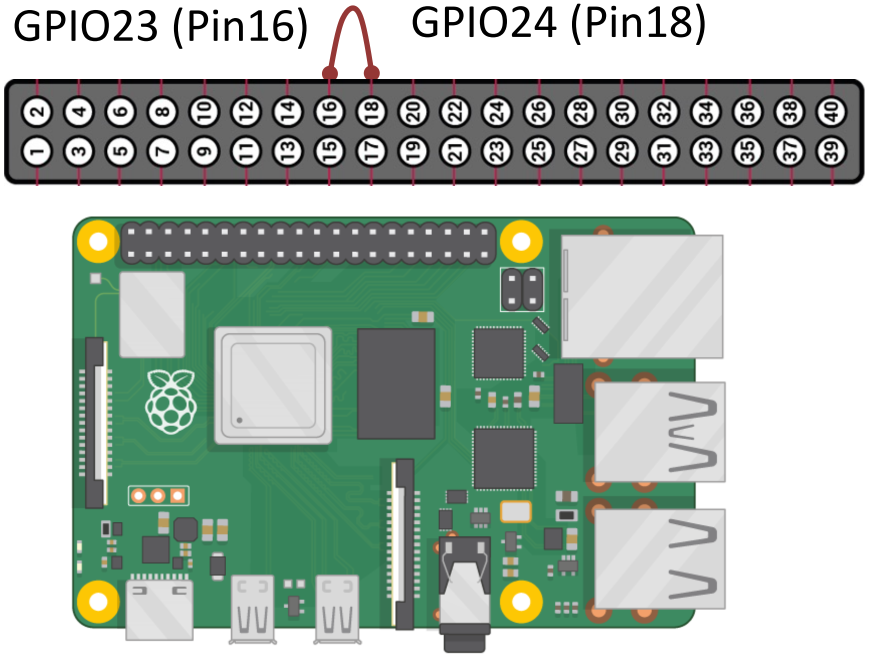
Digital Read.vi Block Diagram on LabVIEWPi.lvproj/raspberrypi
File Edit View Project Operate Tools Window Help
15pt Application Font
Search
While Loop
DI Channel
U8
Digital Read
TF
Open.vi
Local I/O
Digital Read.vi
Digital Read 1 Chan
Close.vi
Simple Error H
Error ?!
Stop Button
TF
i
LabVIEWPi.lvproj/raspberrypi
Digital Read.vi Front Panel on LabVIEWPi.lvproj/r...
File Edit View Project Operate Tools Window Help
15pt Application Font
DIGITAL READ
Digital Read
Stop
LabVIEWPi.lvproj/raspberrypi

LabVIEW Digital Write - Read

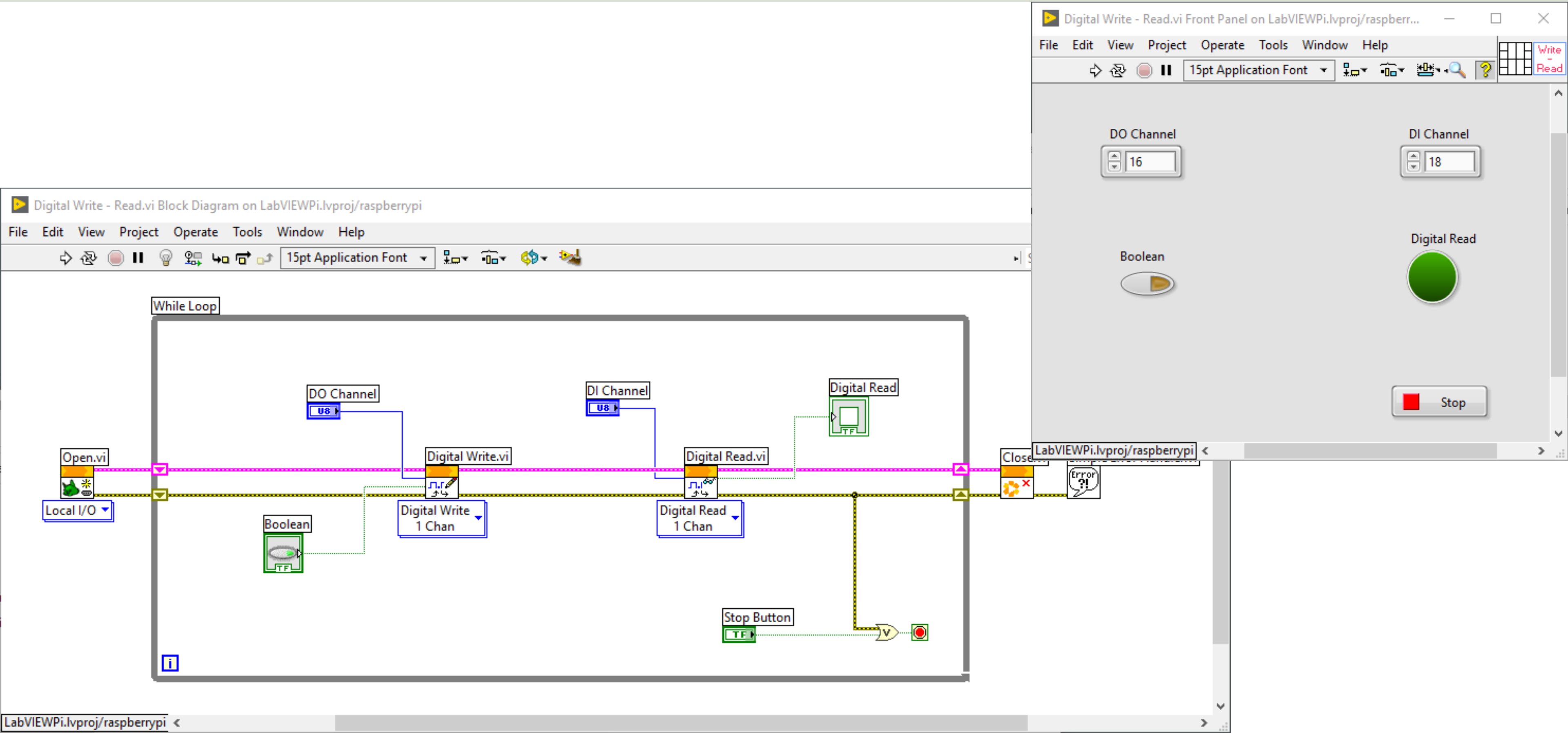
We can test the Digital Read by wiring a “Digital Out” (Write) Channel to the “Digital In” (Read) Channel



3V3 power	1	2	5V power
GPIO 2 (SDA)	3	4	5V power
GPIO 3 (SCL)	5	6	Ground
GPIO 4 (GPCLK0)	7	8	GPIO 14 (TXD)
Ground	9	10	GPIO 15 (RXD)
GPIO 17	11	12	GPIO 18 (PCM_CLK)
GPIO 27	13	14	Ground
GPIO 22	15	16	GPIO 23
3V3 power	17	18	GPIO 24
GPIO 10 (MOSI)	19	20	Ground
GPIO 9 (MISO)	21	22	GPIO 25
GPIO 11 (SCLK)	23	24	GPIO 8 (CE0)
Ground	25	26	GPIO 7 (CE1)
GPIO 0 (ID_SD)	27	28	GPIO 1 (ID_SC)
GPIO 5	29	30	Ground
GPIO 6	31	32	GPIO 12 (PWM0)
GPIO 13 (PWM1)	33	34	Ground
GPIO 19 (PCM_FS)	35	36	GPIO 16
GPIO 26	37	38	GPIO 20 (PCM_DIN)
Ground	39	40	GPIO 21 (PCM_DOUT)



LabVIEW Digital Write - Read



Digital Write - Read.vi Front Panel on LabVIEWPi.lvproj/raspberr...

DO Channel: 16

DI Channel: 18

Digital Write - Read.vi Block Diagram on LabVIEWPi.lvproj/raspberrypi

Build and Deploy Executable LabVIEW Application running on Raspberry Pi at Startup

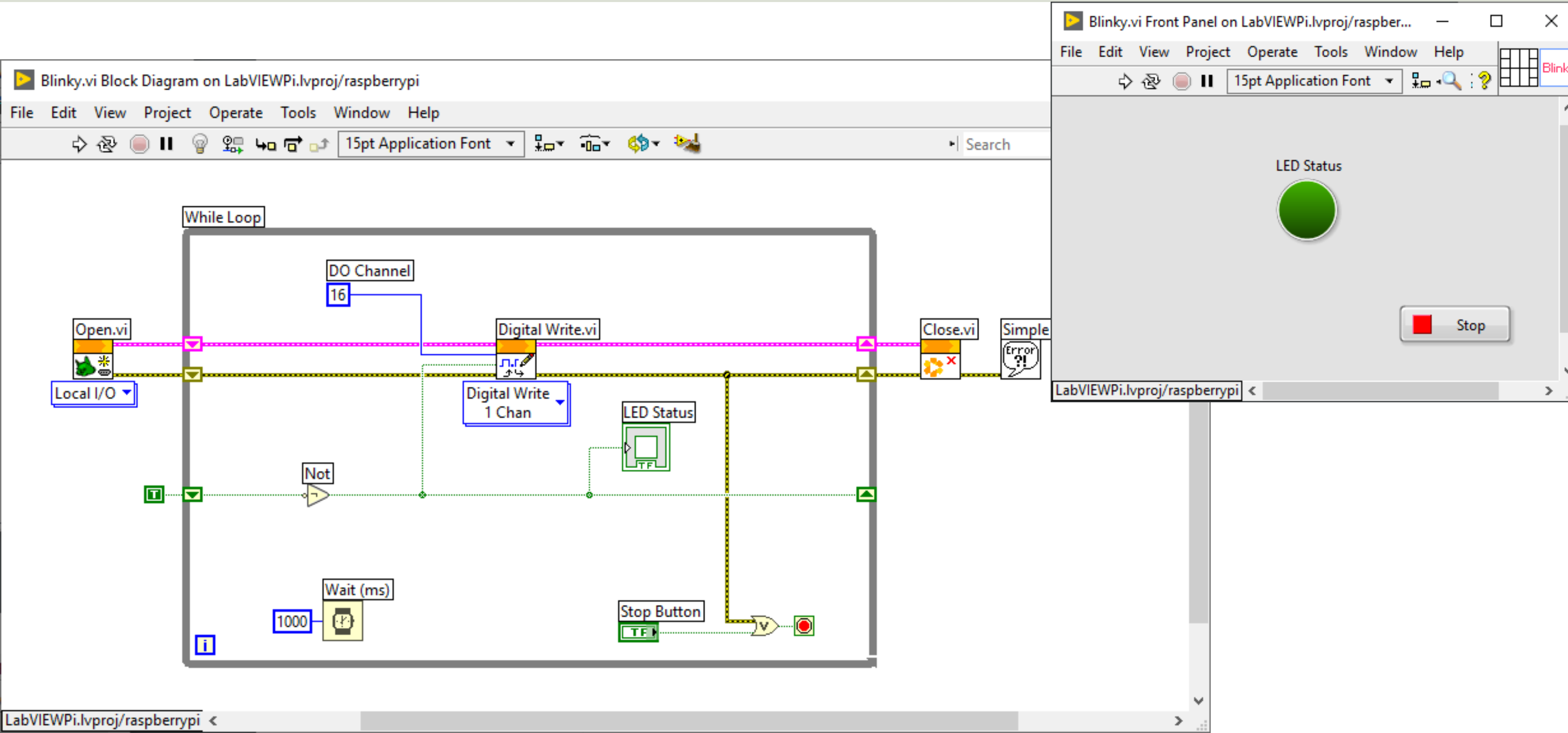
Hans-Petter Halvorsen

Table of Contents

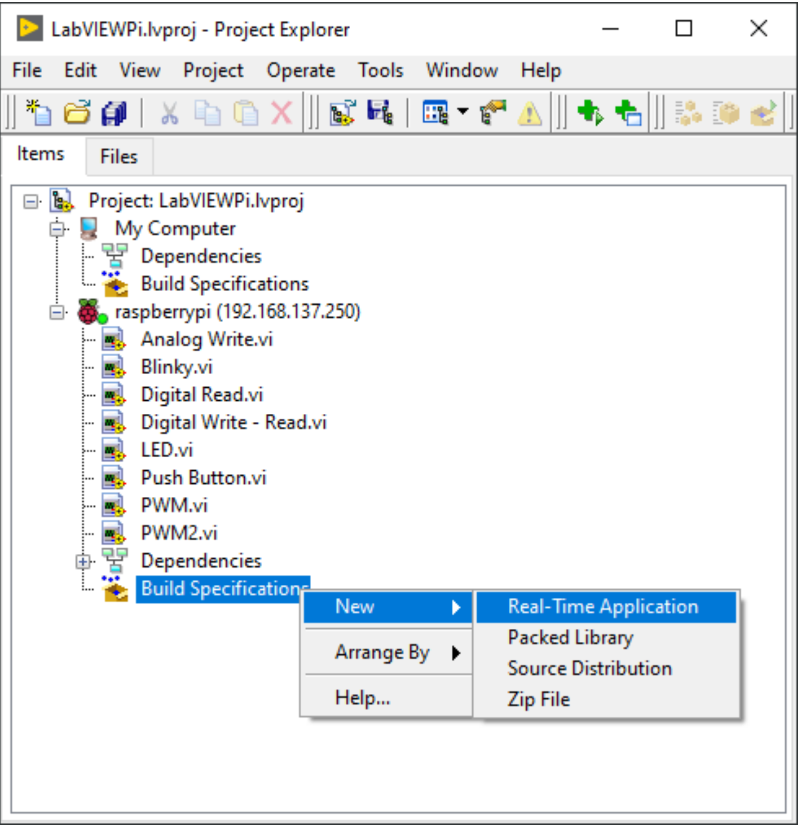
Building Executable Applications

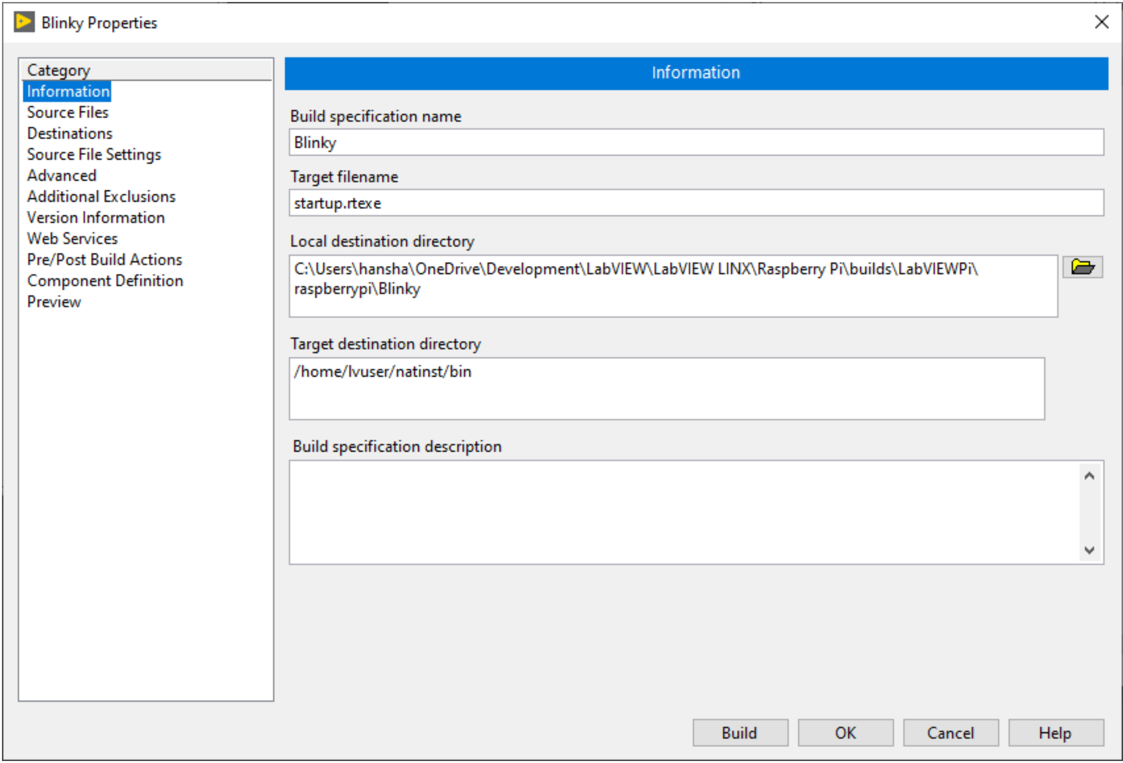
- We will deploy an Executable LabVIEW Application, so it runs on startup of the Raspberry Pi without having a connection to the Host PC
- In order to create and build executable Application you need the Application Builder package
- From LabVIEW 2022 Q3 and newer the Application Builder is included with LabVIEW Professional Development System

Blinky Application



Build Application





Build specification name: Blinky

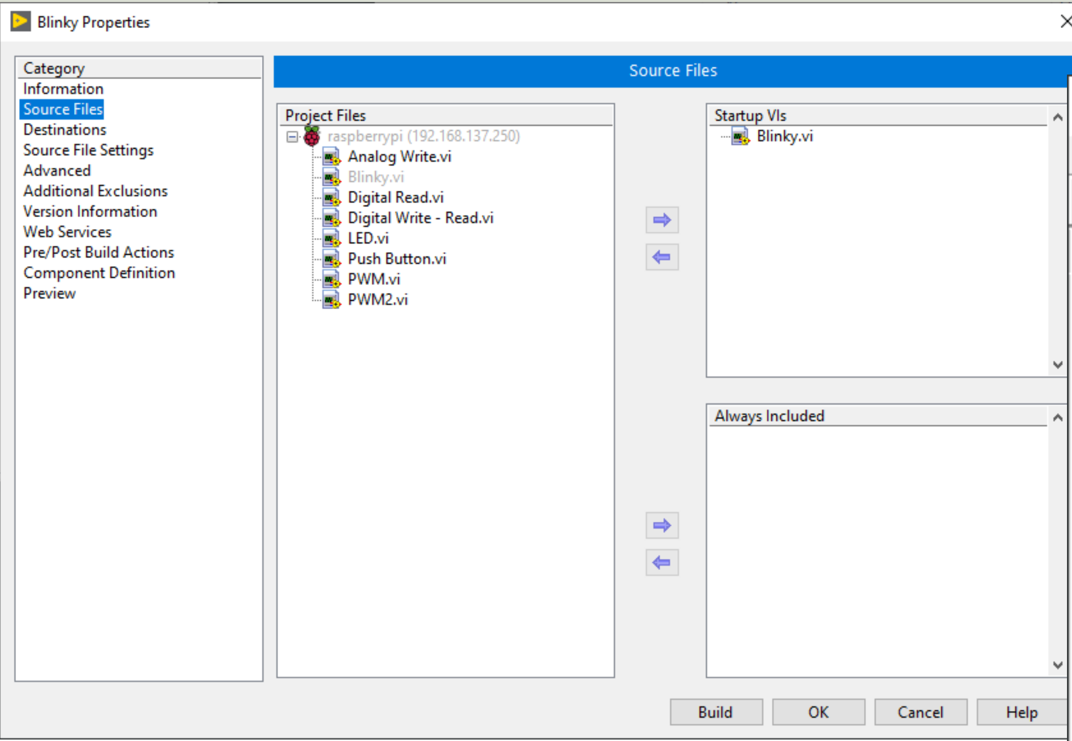
Target filename: startup.rtexe

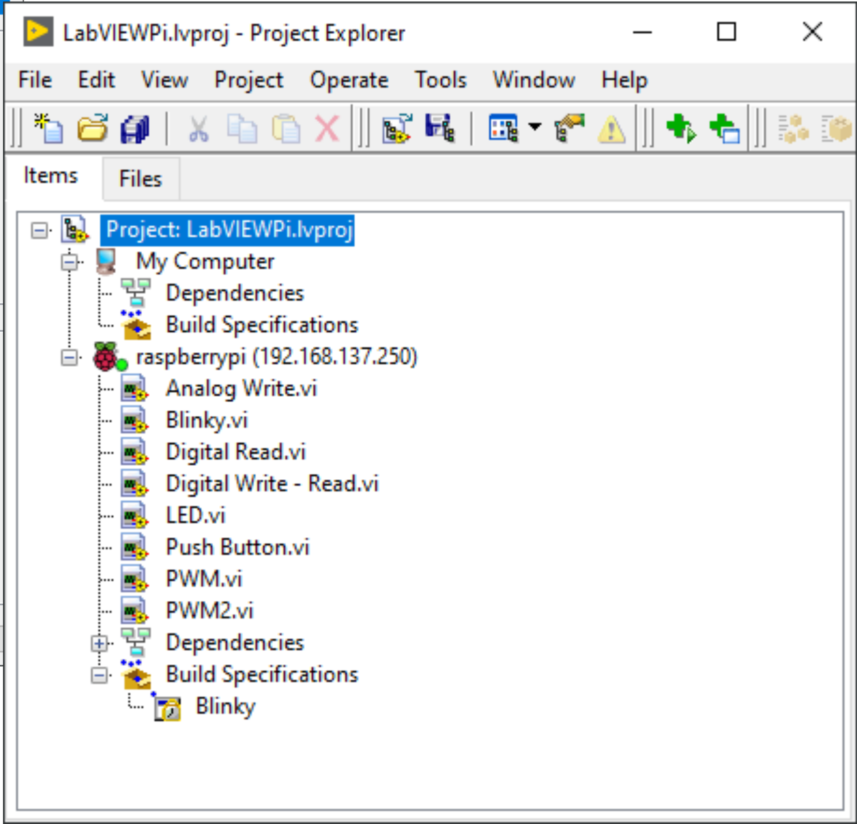
Local destination directory: C:\Users\hansha\OneDrive\Development\LabVIEW\LabVIEW LINX\Raspberry Pi\builds\LabVIEWPi\raspberrypi\Blinky

Target destination directory: /home/lvuser/natinst/bin

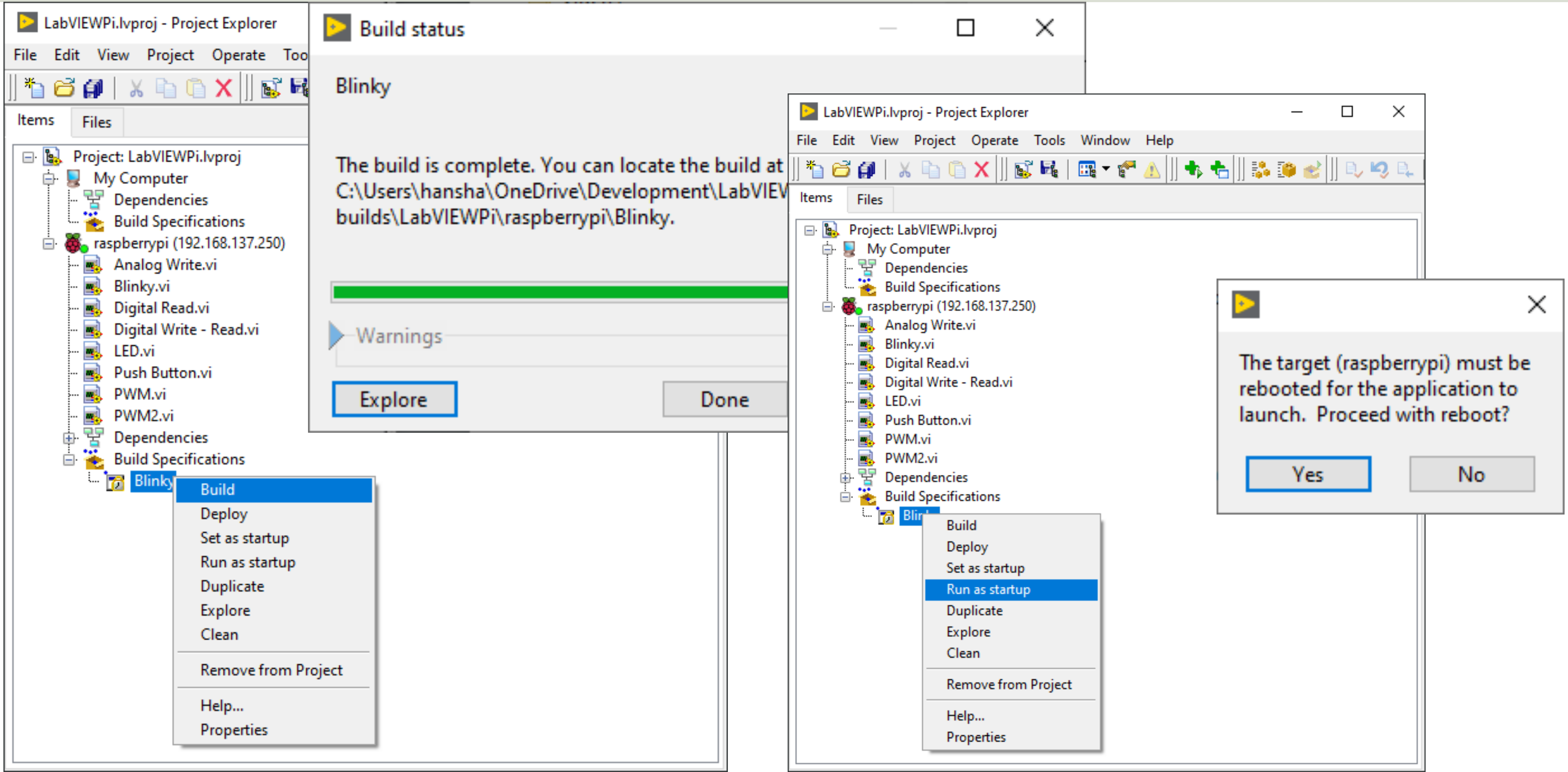
Build specification description:

Build Application





Build Application



Build status

Blinky

The build is complete. You can locate the build at C:\Users\hansha\OneDrive\Development\LabVIEW

builds\LabVIEWPi\raspberrypi\Blinky.

The target (raspberrypi) must be rebooted for the application to launch. Proceed with reboot?

Summary

- This Tutorial has shown how we can use Raspberry Pi in combination with the LabVIEW Programming environment
- “LabVIEW LINX Toolkit” is an add-on for LabVIEW which makes it possible to program the Raspberry Pi device using LabVIEW
- In that way we can create Data Logging Applications, etc. without the need of an expensive DAQ device
- If you in addition use the “LabVIEW Community Edition” (free for non-commercial use) you get a very low-cost DAQ/Datalogging System!
- You can also easily add features for logging data to Files or a Database System like SQL Server, or an OPC Server, etc.
- In later Tutorials, I will show how you can use Pulse Width Modulation (PWM), Push Buttons, I2C and SPI Interfaces, etc.

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog