

<https://www.halvorsen.blog>



LabVIEW LINX and Raspberry Pi

Using **SPI** and **I2C** Interfaces

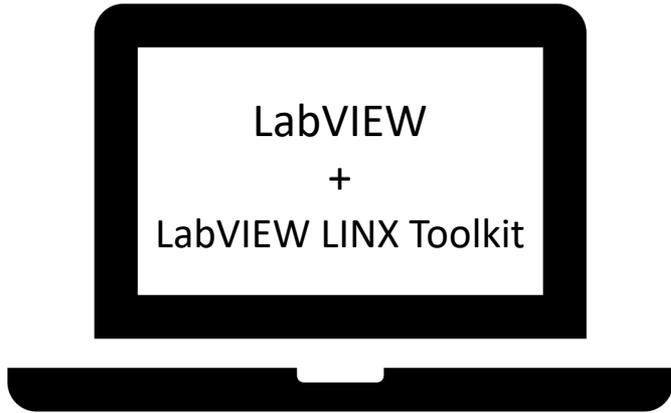
Hans-Petter Halvorsen

Table of Contents

- Raspberry Pi and LabVIEW LINX
- SPI and I2C Interfaces
- TC74 Temperature Sensor
- ..

LabVIEW + LabVIEW LINX Toolkit

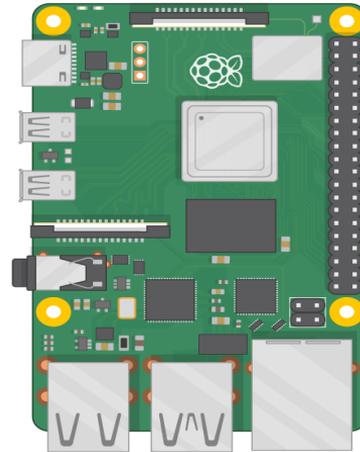
PC



Ethernet
or Wi-Fi



Raspberry Pi

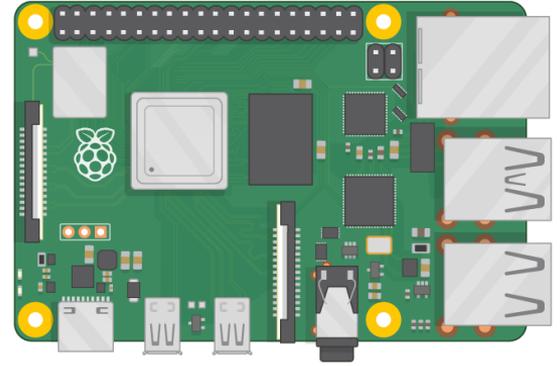
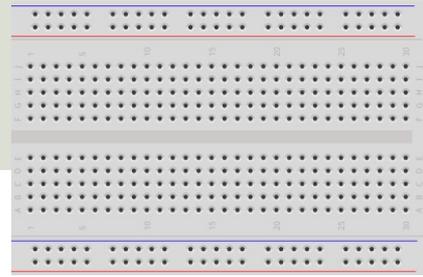
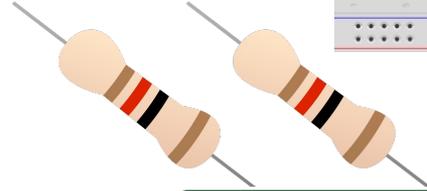
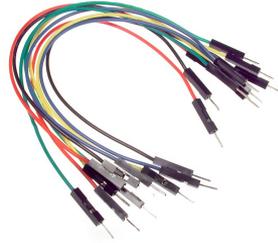


GPIO



Hardware

- Raspberry Pi
- Breadboard
- Wires (Jumper Wires)
- Resistors ($R = 270\Omega$)
- LED, Push Button
- Sensors/Components with SPI/I2C Interface



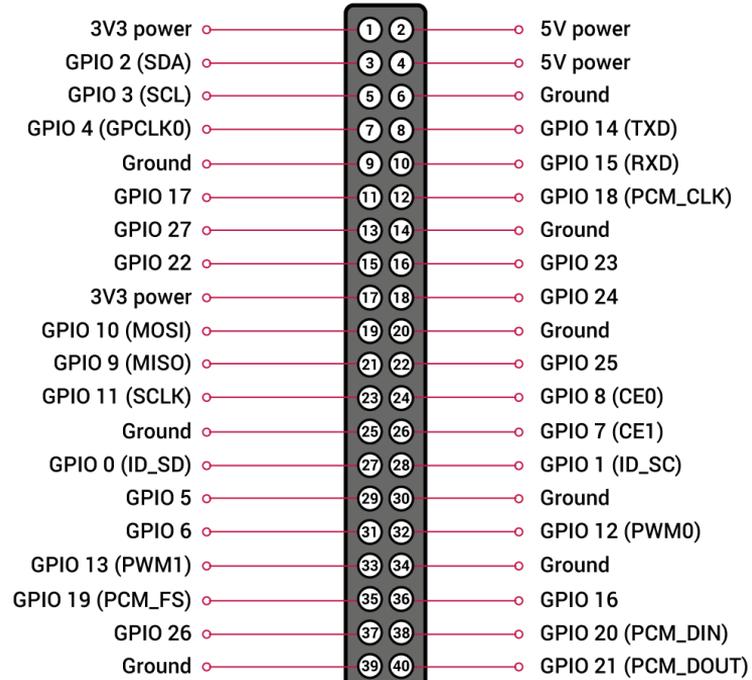
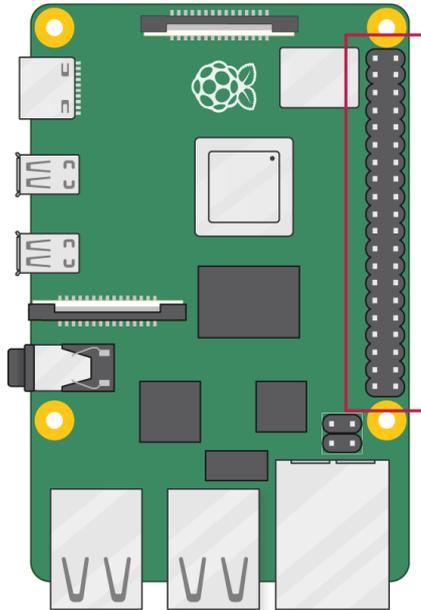
Hardware and Software

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



Raspberry Pi and LabVIEW LINUX

GPIO



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

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>

<https://www.halvorsen.blog>



LabVIEW LINUX Toolkit

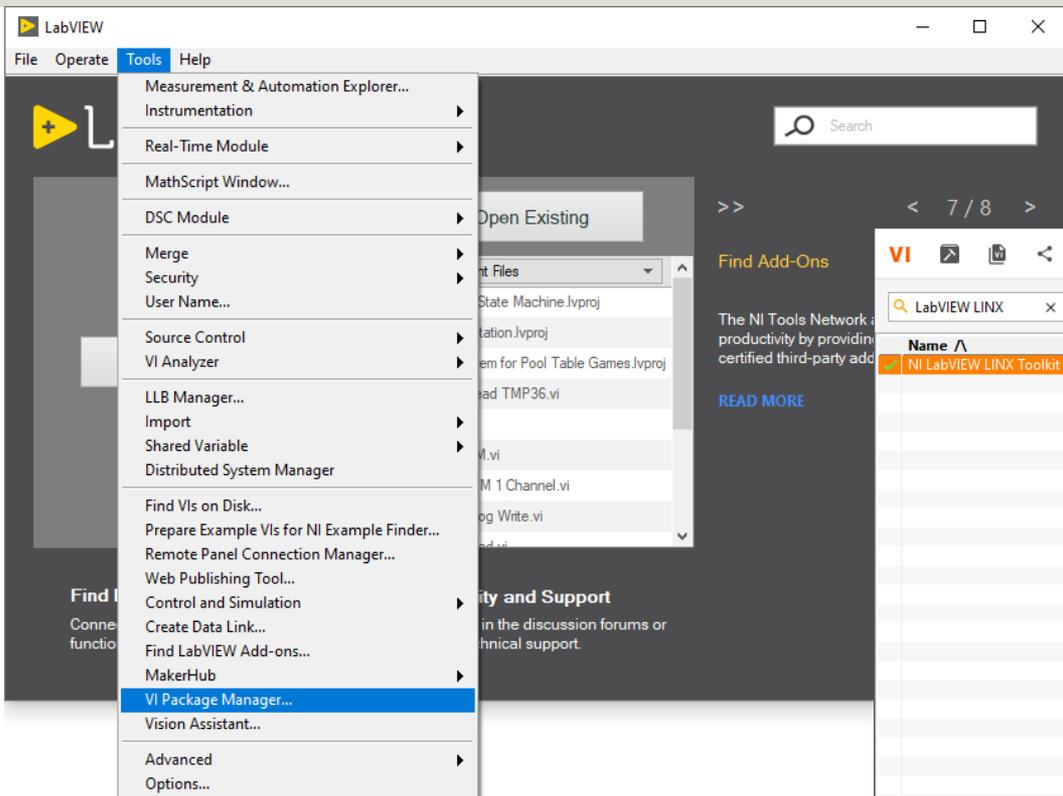
Hans-Petter Halvorsen

[Table of Contents](#)

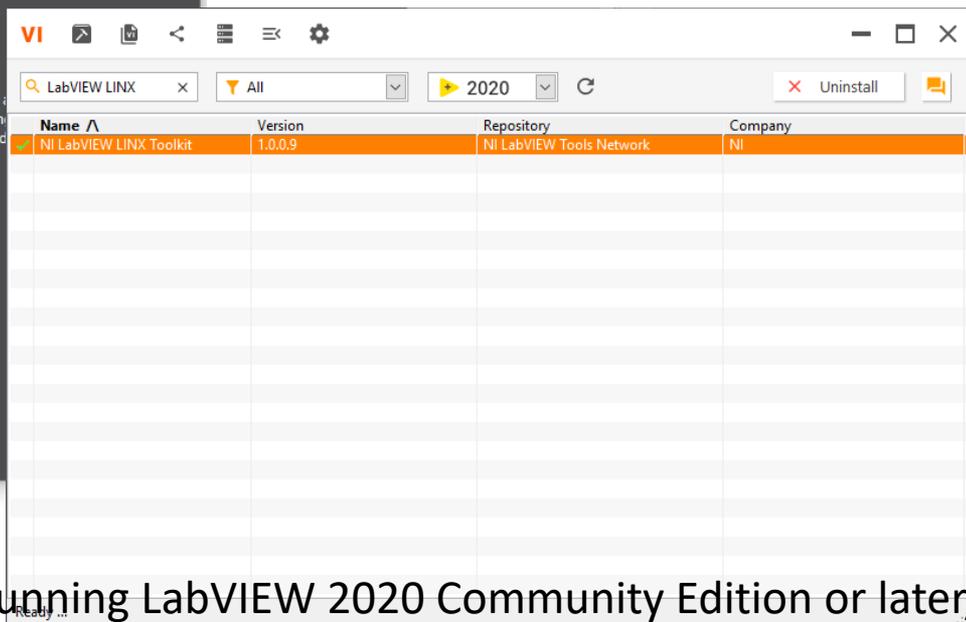
LabVIEW LINX Toolkit

- 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



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

Sensors

↑ Search Customize

Accelerometer Beta Community
Display Distance Digilent
Lights Mindstorms Misc
Motion Pmods Temp
Sig Gen

LINX

↑ Search Customize

Open Close Peripherals
Sensors Utilities

Peripherals

↑ Search Customize

Analog Digital PWM
I2C SPI UART

Utilities

↑ Search Customize

Custom CMD Loop Freq
Check Channel Get User ID Set User ID
Config Enet Config Wifi

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

Get Support

Find Drivers and Add-ons

Community and Support

Additional Search

Keyword

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 refs 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](#)

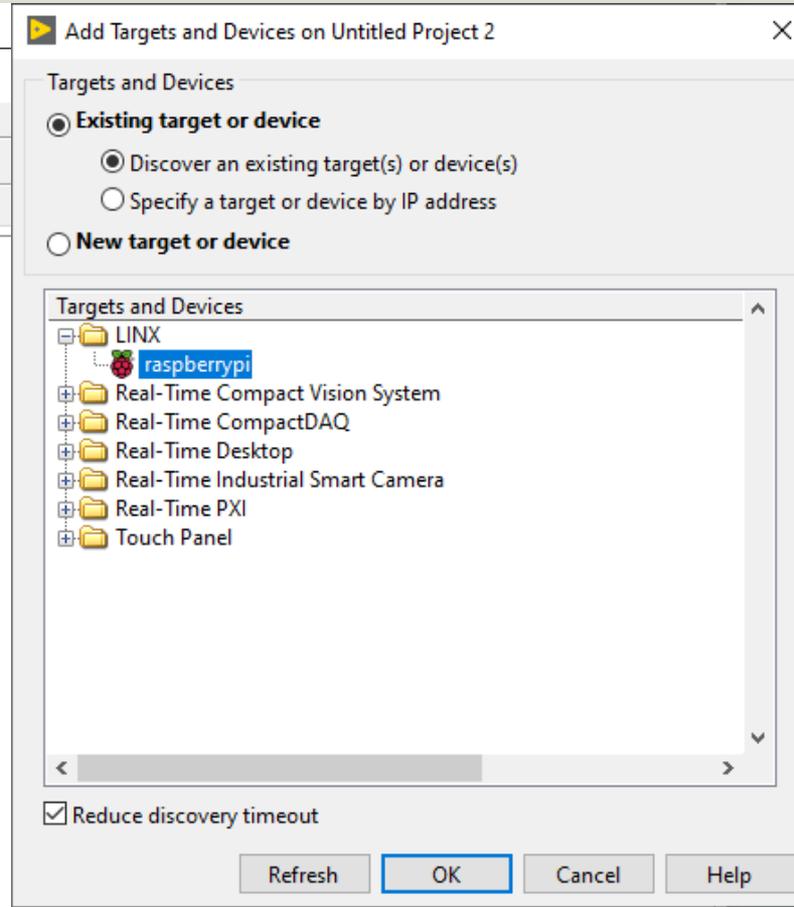
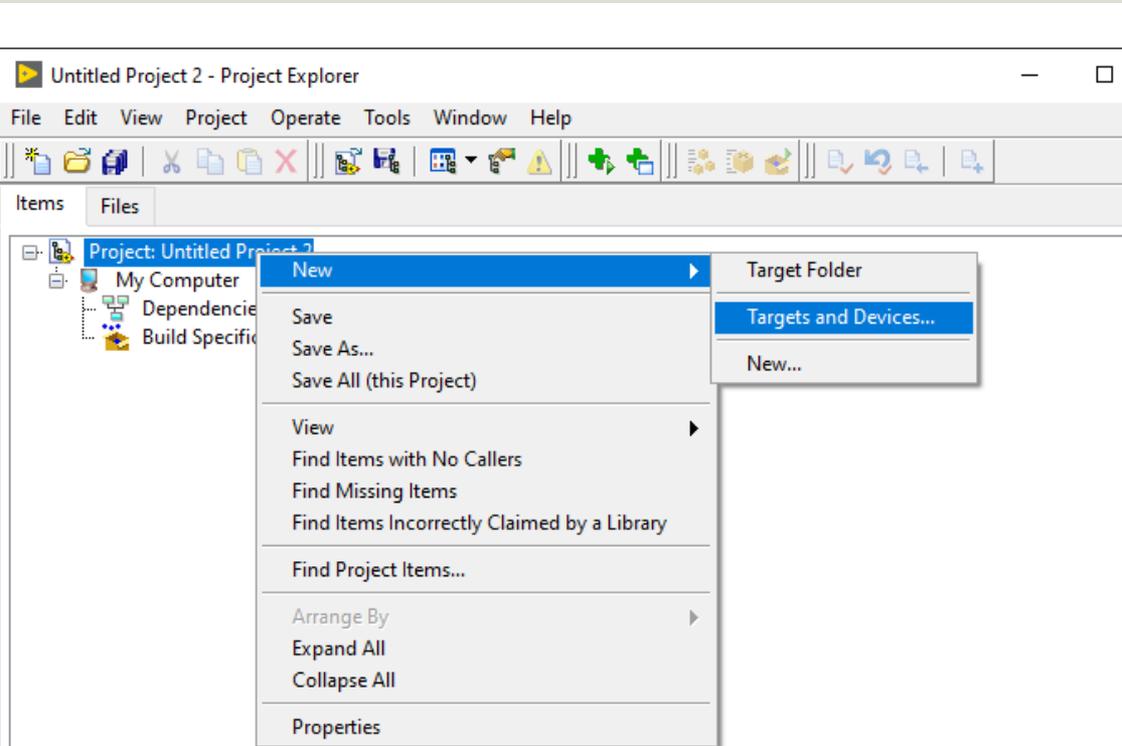
Continuous Measurement and Logging *Sample Projects*
Acquires measurements continuously and logs them to disk. This sample project is based on the Queued Message Handler template. [More Information](#)

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](#)

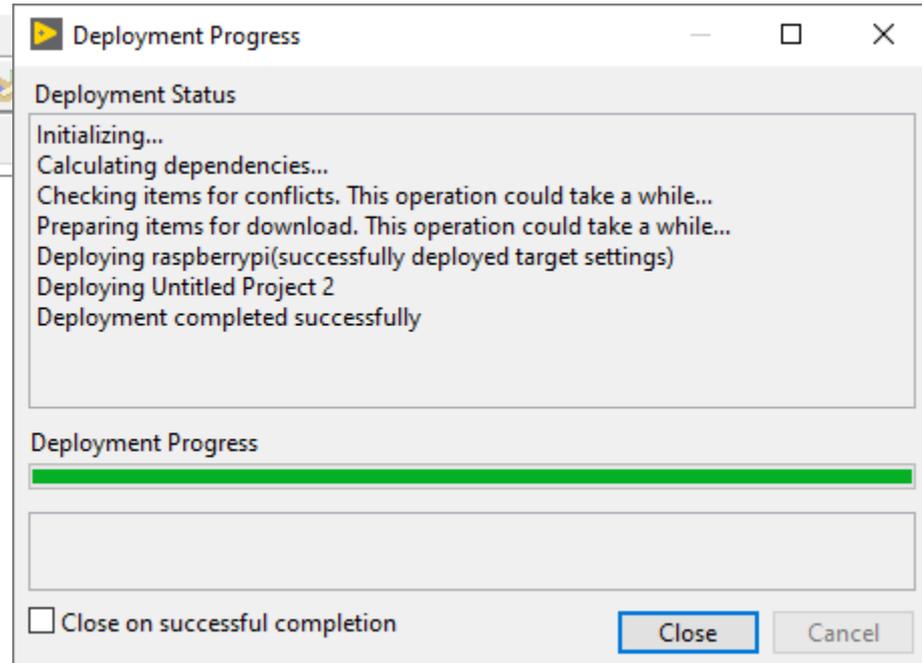
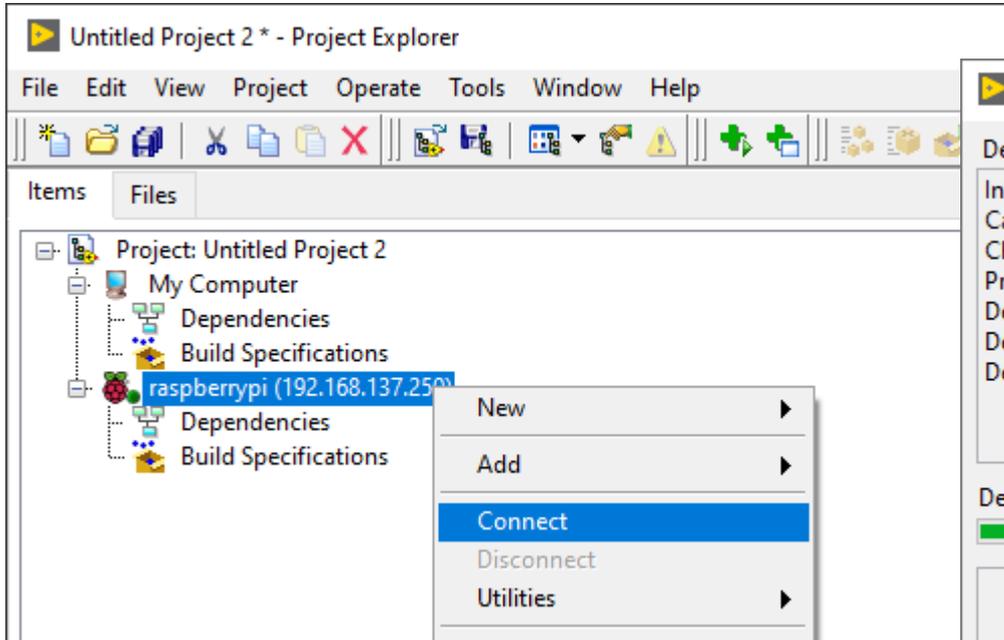
Instrument Driver Project *Templates*

Finish Cancel Help

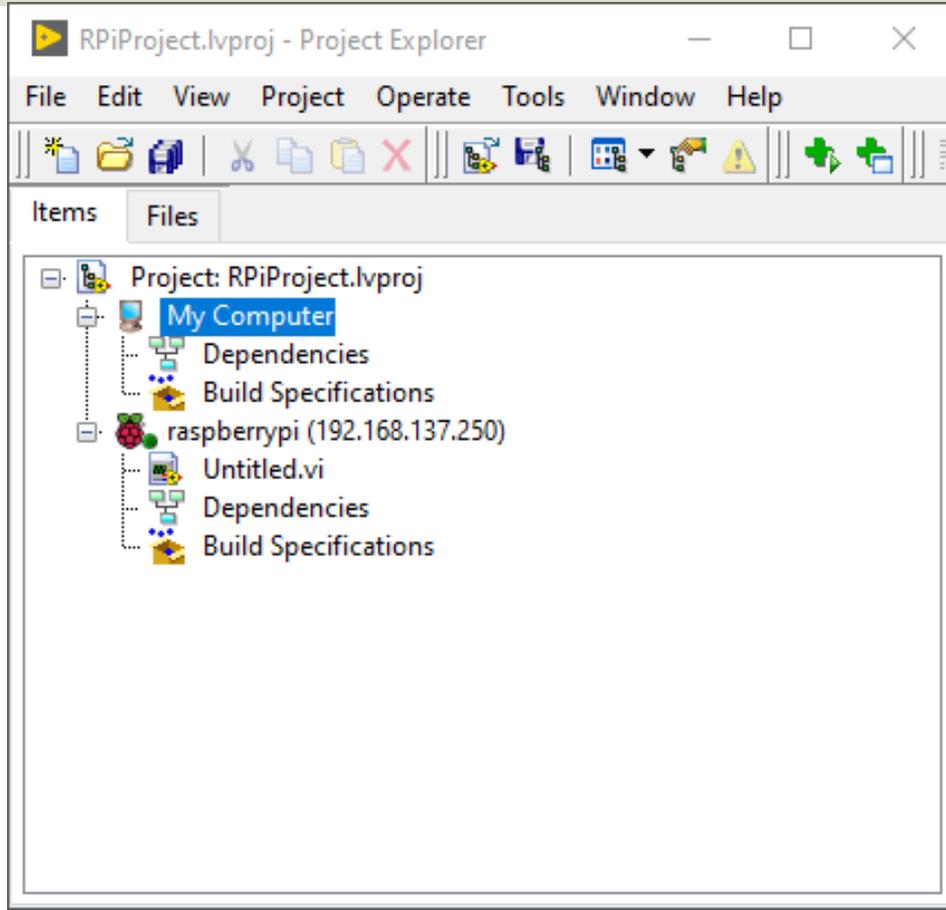
Create your Raspberry Pi Project



Create your Raspberry Pi Project



LabVIEW Project Explorer



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

Resources

Introduction to Raspberry Pi and LabVIEW LINX has been given in another Tutorial.

- LabVIEW LINX and Raspberry Pi - xxx

Blog:

<https://www.halvorsen.blog/>

YouTube Channel @Industrial IT and Automation

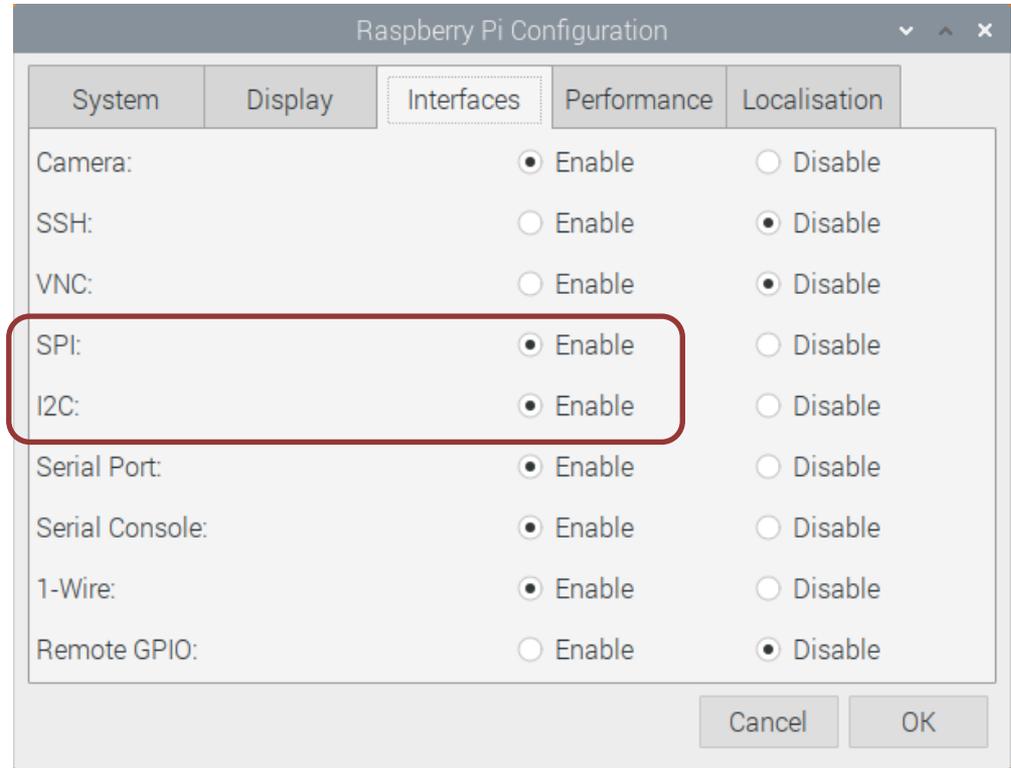
<https://www.youtube.com/IndustrialITandAutomation>



SPI and I2C Interfaces

Raspberry Pi Configuration

You need to Enable **SPI** and **I2C**

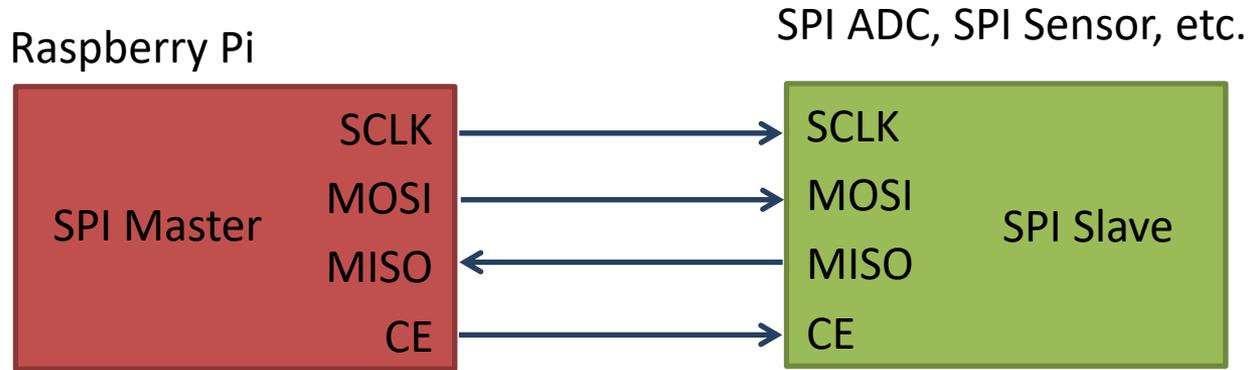


SPI

- Serial Peripheral Interface (SPI)
- 4–Wire Protocol (SCLK, CE, MOSI, MISO)
- SPI is an interface to communicate with different types of electronic components like Sensors, Analog to Digital Converts (ADC), etc. that supports the SPI interface
- Thousands of different Components and Sensors supports the SPI interface

SPI

SPI devices communicate in full duplex mode using a master-slave architecture with a single master



The SPI bus specifies four logic signals:

- **SCLK**: Serial Clock (output from master)
- **MOSI**: Master Out Slave In (data output from master)
- **MISO**: Master In Slave Out (data output from slave)
- **CE** (often also called SS - Slave Select): Chip Select (often active low, output from master)

I2C

- I2C is a multi-drop bus
- 2-Wire Protocol (SCL + SDA)
- Multiple devices can be connected to the I2C pins on the Raspberry Pi
- Each device has its own unique I2C address

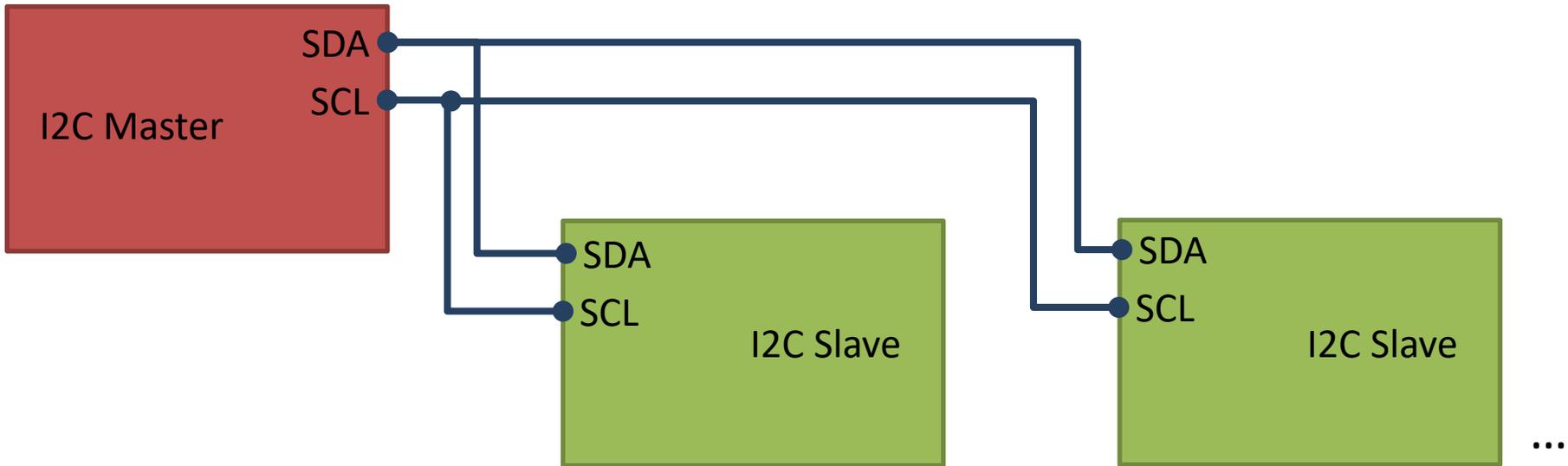
I2C

Multiple devices can be connected to the I2C pins on the Raspberry Pi

Master – Device that generates the clock and initiates communication with slaves

Slave – Device that receives the clock and responds when addressed by the master.

Raspberry Pi



ADC, DAC, Sensor, etc. with I2C Interface

SPI/I2C

- Digital Sensors typically use either the SPI or the I2C communication protocol
- The Arduino UNO has built-in hardware support for SPI and I2C communication

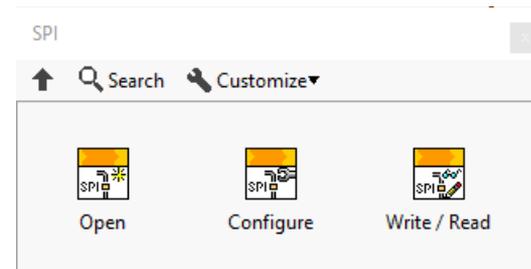
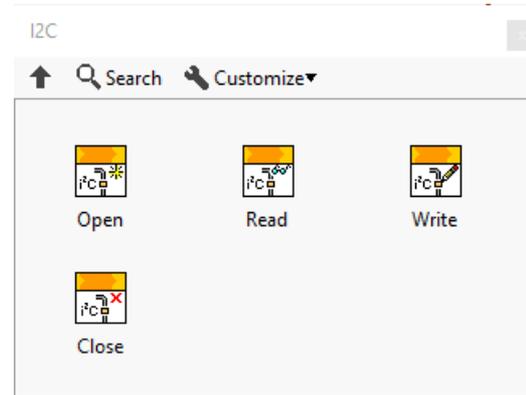
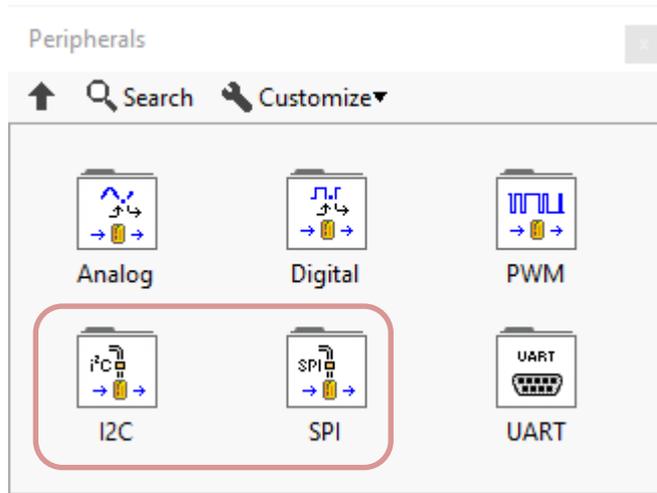
SPI

- 4-Wire Protocol
- SPI supports full-duplex. Data can be sent and received at the same time
- Higher data transfer rate than I2C
- Complex wiring if more than one Slave

I2C

- 2-Wire Protocol
- I2C supports only half-duplex. Data cannot be sent and received at the same time
- Lower data transfer rate than SPI
- Multiple Slaves are easier

SPI/I2C in LabVIEW LINX



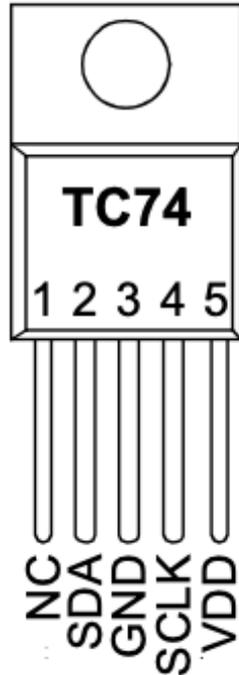
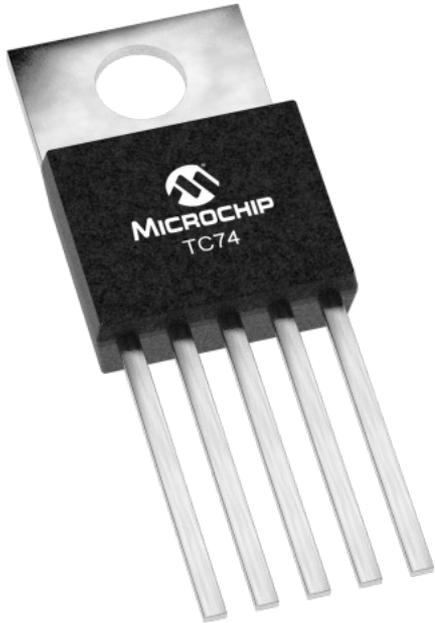


TC74 Temperature Sensor with I2C

TC74 Temperature Sensor

SMBus/I2C Interface

TC74A0-5.0VAT

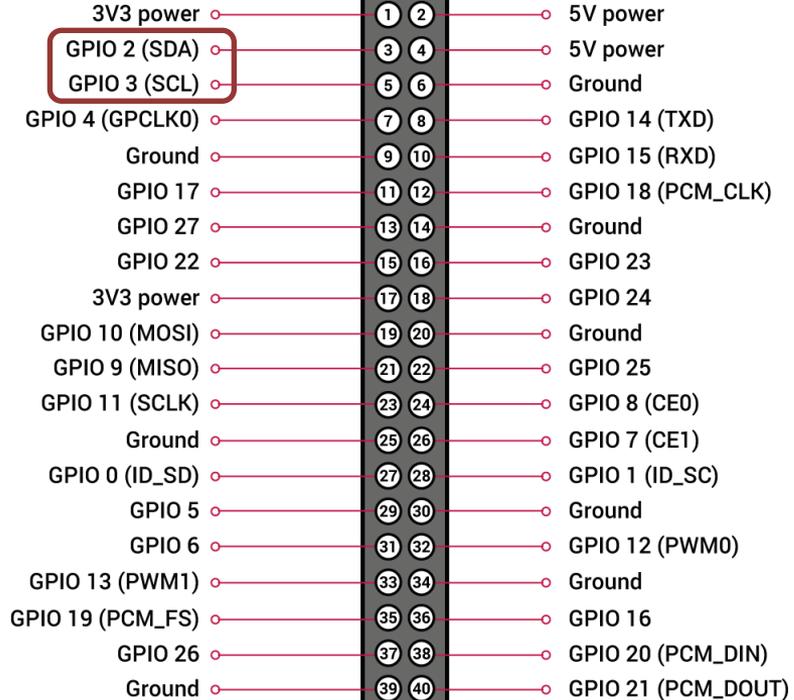
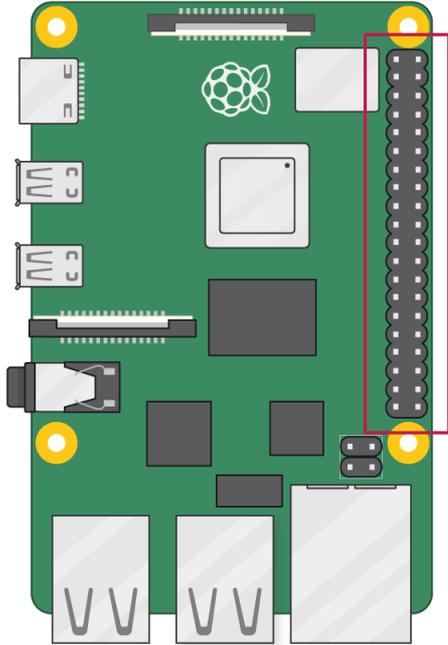


- The TC74 acquires and converts temperature information from its onboard solid-state sensor with a **resolution of $\pm 1^{\circ}\text{C}$** .
- It stores the data in an internal register which is then read through the serial port.
- The system interface is a slave SMBus/I2C port, through which temperature data can be read at any time.
- Device Address: **0x48**

Datasheet: <https://ww1.microchip.com/downloads/en/DeviceDoc/21462D.pdf>

I2C Wiring on Raspberry Pi

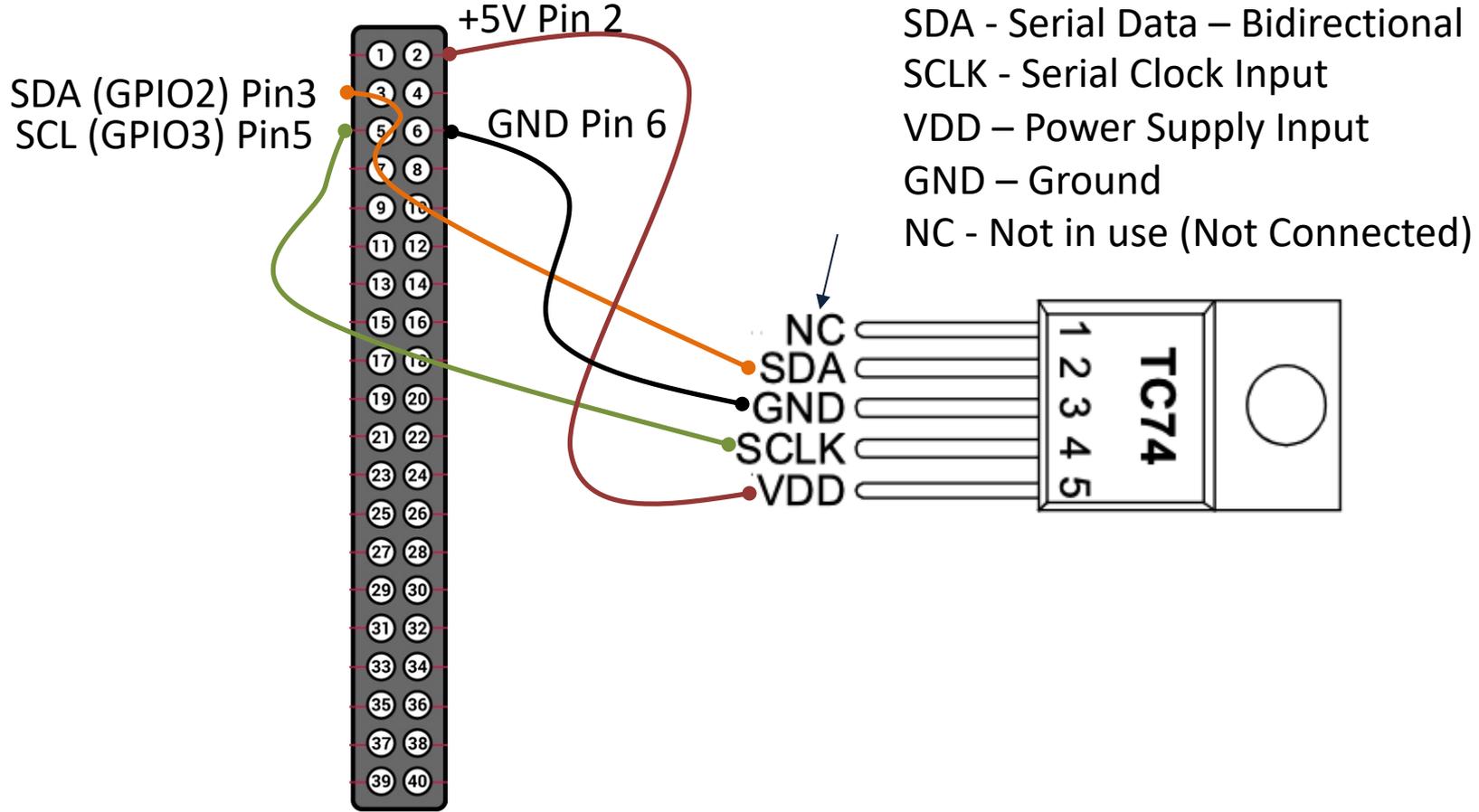
GPIO 40 pins Connector



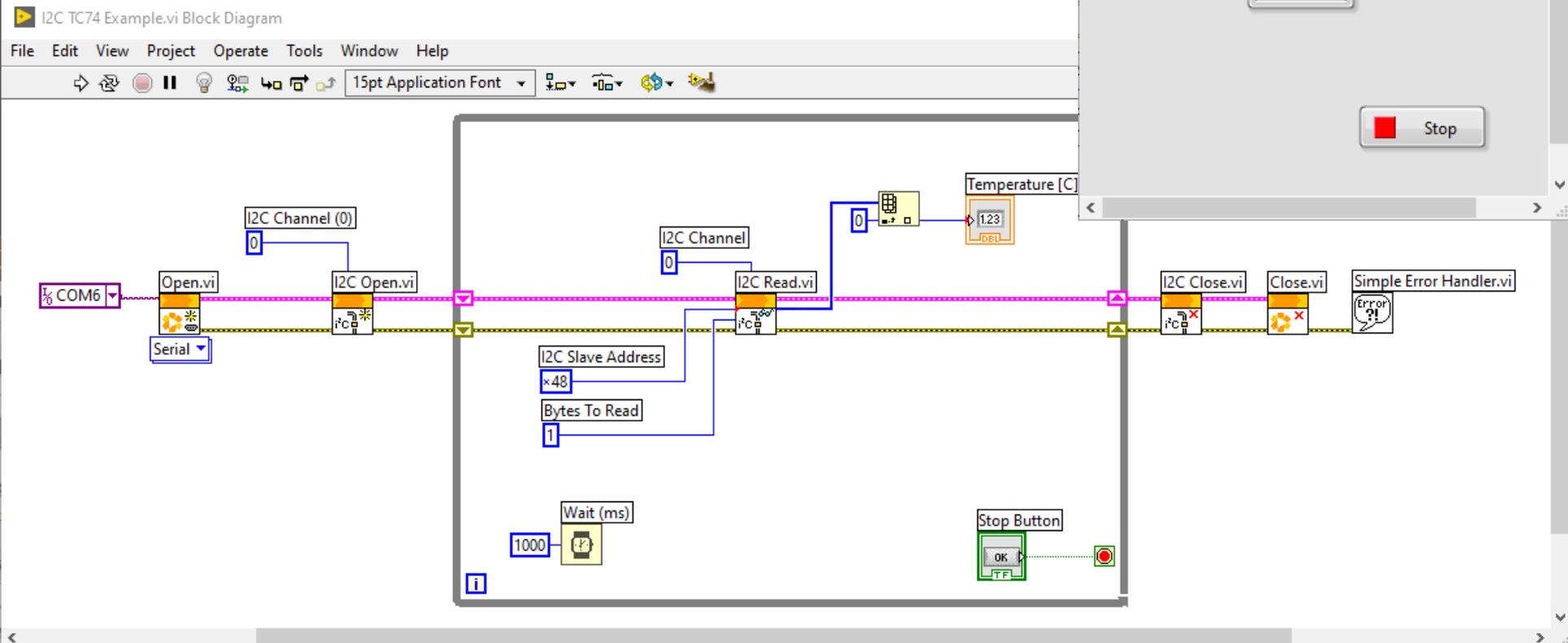
Note! The I2C pins include a fixed 1.8 kΩ pull-up resistor to 3.3v.

TC74 Wiring

Raspberry Pi GPIO Pins



LabVIEW



Slave Address

The image displays a LabVIEW block diagram titled "Basic I2C TC74 Example.vi Block Diagram". The diagram includes a "Serial" block connected to a "COM6" port, followed by an "Open.vi" block, an "I2C Open.vi" block, and an "I2C Read.vi" block. The "I2C Open.vi" block has an "I2C Channel" input set to 0 and an "I2C Slave Address" input set to 48. The "I2C Read.vi" block has an "I2C Channel" input set to 0 and a "Bytes To Read" input set to 1. A context menu is open over the "I2C Slave Address" input, with "Hex" selected. A red callout box points to the "I2C Slave Address" input with the text "The TC74 Slave address is a Hexadecimal Number".

The "Numeric Constant Properties: I2C Slave Address" dialog box is open, showing the "Appearance" tab. The "Label" is "I2C Slave Address", "Visible" is checked, and "Show radix" is checked. The "Position" is set to Left: 396 and Top: 187. A red callout box points to the "Show radix" checkbox with the text "Right-click and Select Properties".

Buttons for "OK", "Cancel", and "Help" are visible at the bottom of the dialog box.

<https://www.halvorsen.blog>

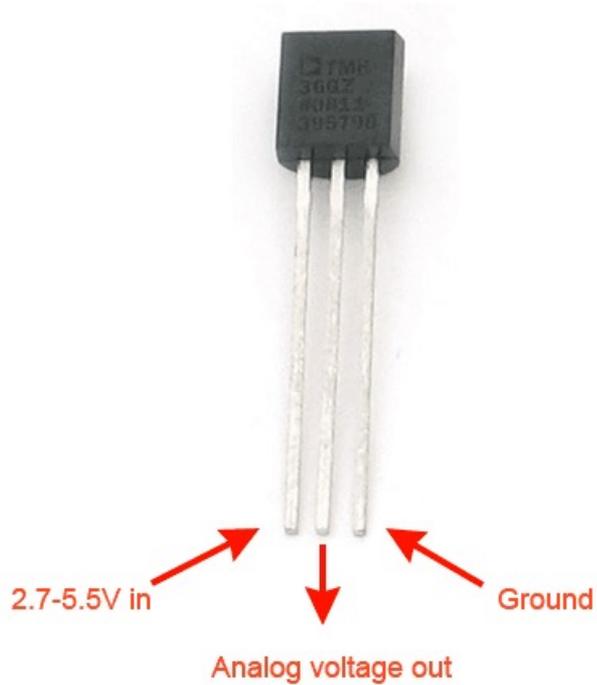


TMP36

Temperature Sensor

Hans-Petter Halvorsen

TMP36 Temperature Sensor

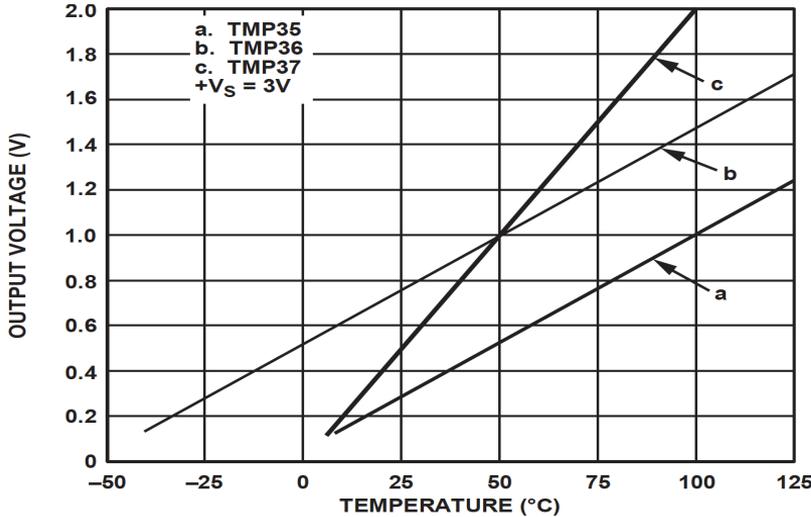


A Temperature sensor like TM36 use a solid-state technique to determine the temperature.

They use the fact as temperature increases, the voltage across a diode increases at a known rate.

<https://learn.adafruit.com/tmp36-temperature-sensor>

TMP36 Temperature Sensor



Convert from Voltage (V) to degrees Celsius

From the Datasheet we have:

$$(x_1, y_1) = (0.75V, 25^\circ C)$$

$$(x_2, y_2) = (1V, 50^\circ C)$$

There is a linear relationship between Voltage and degrees Celsius:

$$y = ax + b$$

This gives:

$$y - 25 = \frac{50 - 25}{1 - 0.75} (x - 0.75)$$

Then we get the following formula:

$$y = 100x - 50$$

We can find a and b using the following known formula:

$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1} (x - x_1)$$

<https://www.halvorsen.blog>



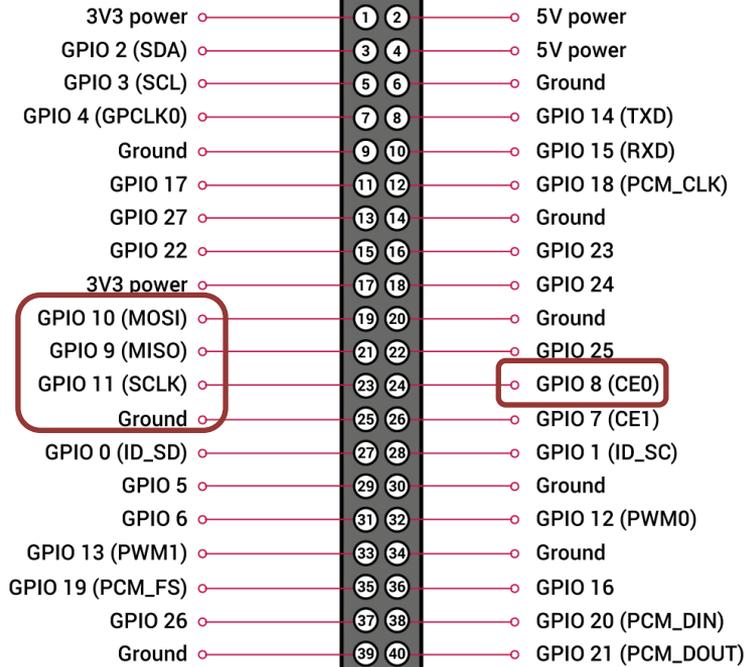
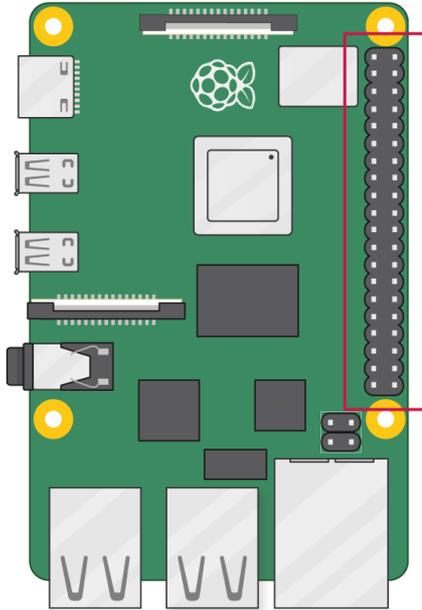
ADC – MCP3002

Analog to Digital Converter

Hans-Petter Halvorsen

SPI Wiring on Raspberry Pi

GPIO 40 pins Connector



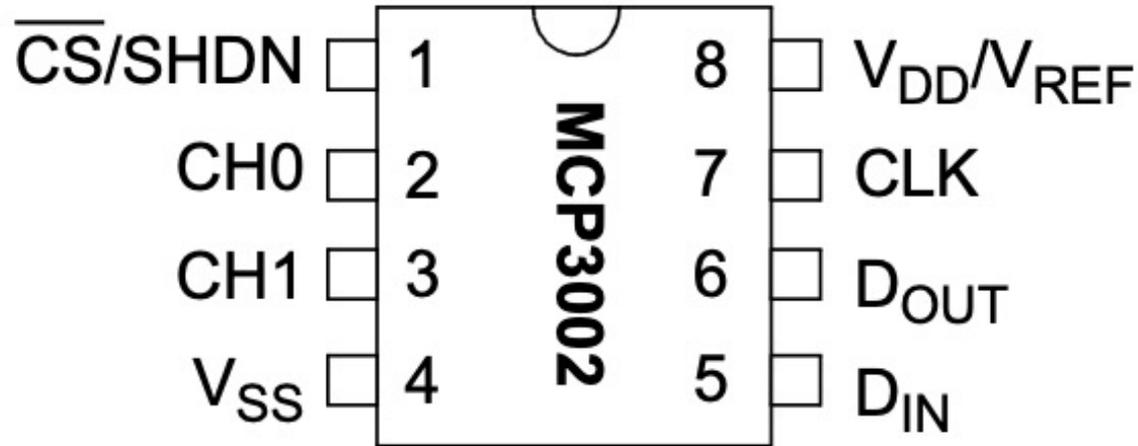
ADC

- The Raspberry Pi has only Digital pins on the GPIO connector
- If you want to use an Analog electric component or an Analog Sensor together with Raspberry Pi, you need to connect it through an external ADC chip
- ADC – Analog to Digital Converter

MCP3002 ADC chip

The MCP3002 is a 10-bit analog to digital converter with 2 channels (0-1).

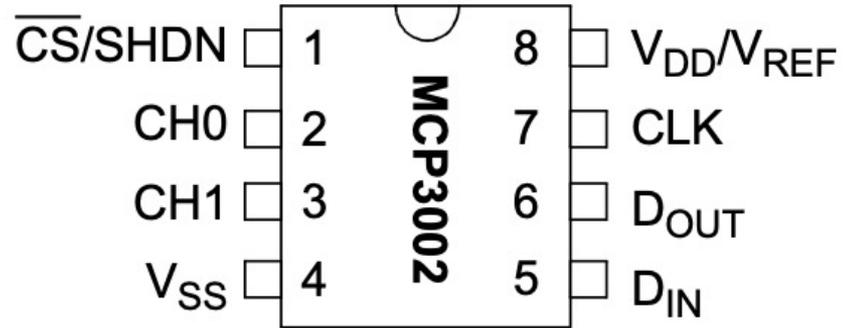
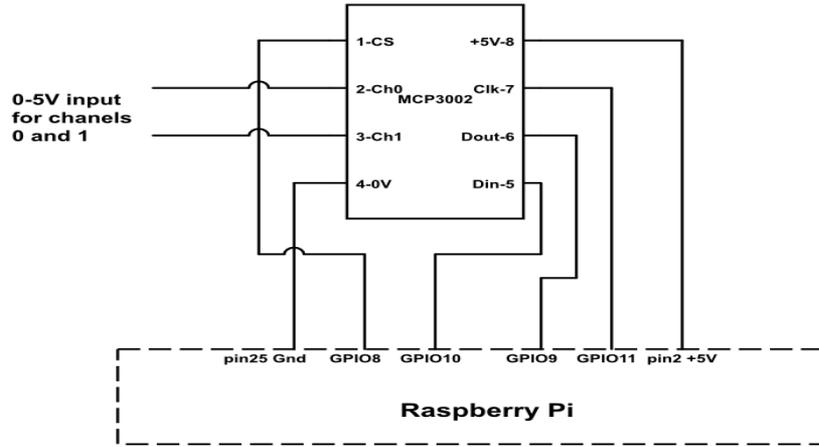
The MCP3002 uses a SPI Interface



<http://ww1.microchip.com/downloads/en/DeviceDoc/21294E.pdf>

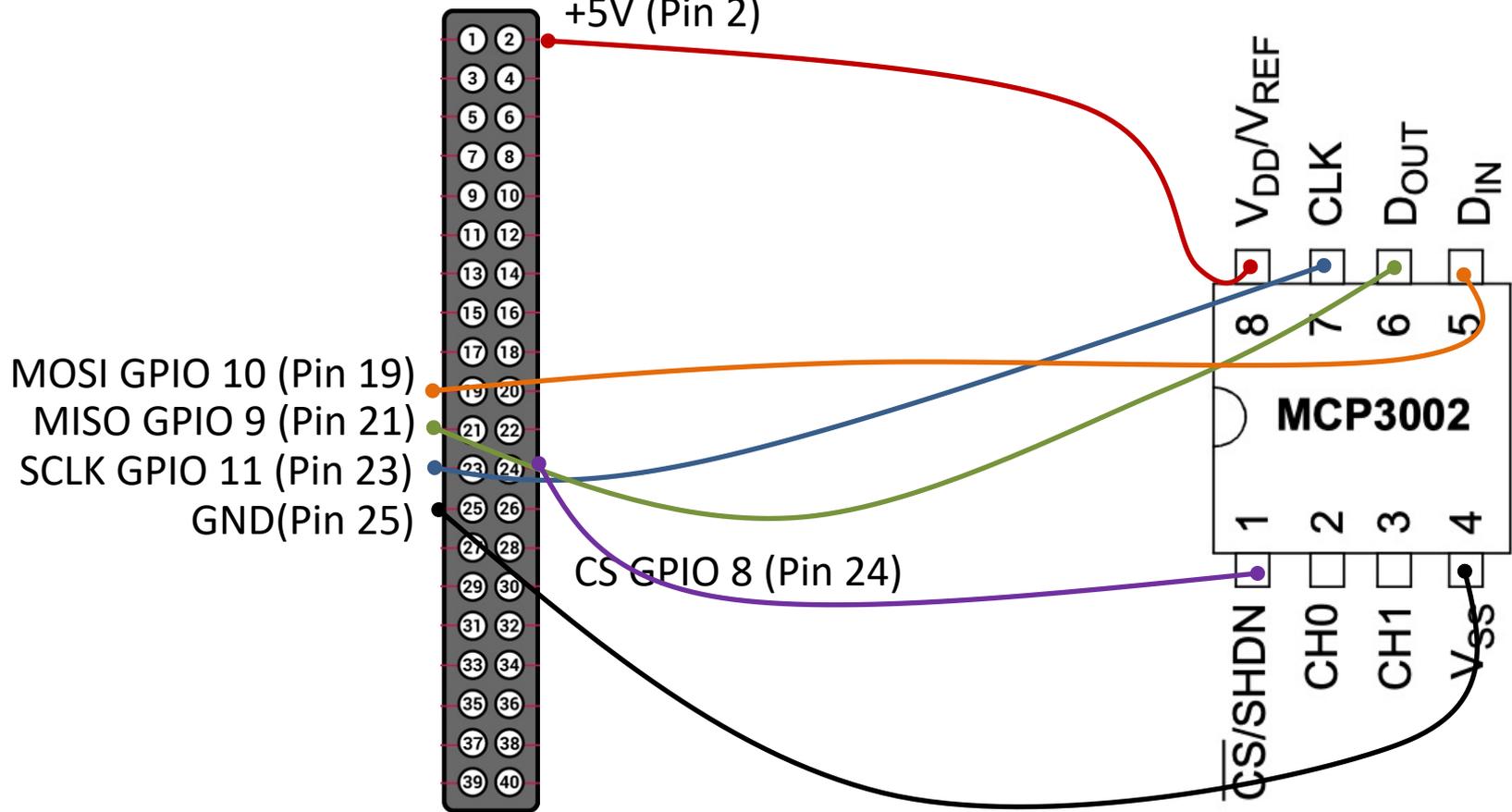
<https://learn.sparkfun.com/tutorials/python-programming-tutorial-getting-started-with-the-raspberry-pi/experiment-3-spi-and-analog-input>

Wiring



Wiring

Raspberry Pi GPIO Pins



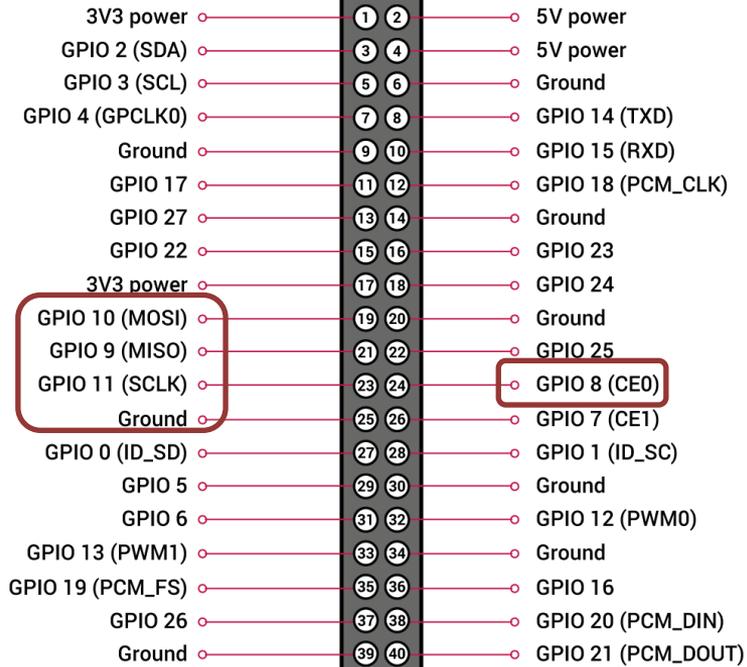
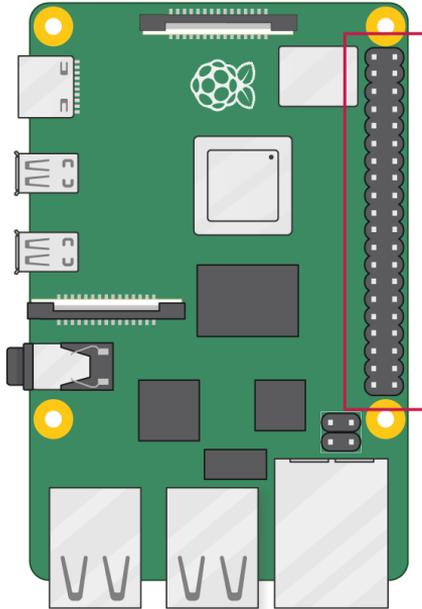


DAC - MCP4911

DAC – Digital to Analog Converter

SPI Wiring on Raspberry Pi

GPIO 40 pins Connector



DAC – MCP4911

- DAC – Digital to Analog Converter
- Arduino UNO has no real Analog Out Channel – only Digital PWM channels
- We can use an external DAC in order to provide a real Analog Out
- MCP4911 is a single channel, 10-bit DAC with an external voltage reference and SPI interface

MCP49xx

MCP49xx is a family of DAC ICs:

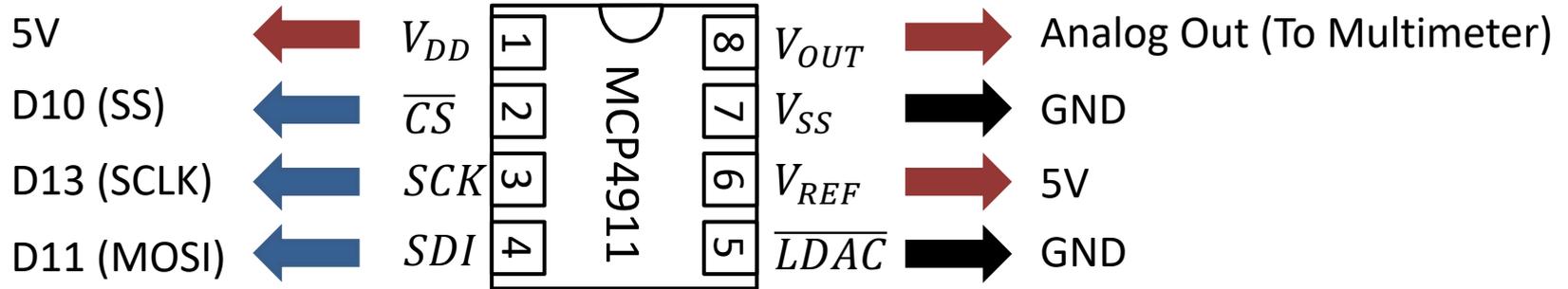
- MCP4901: 8-Bit Voltage Output DAC
- **MCP4911: 10-Bit Voltage Output DAC**
- MCP4921: 12-Bit Voltage Output DAC



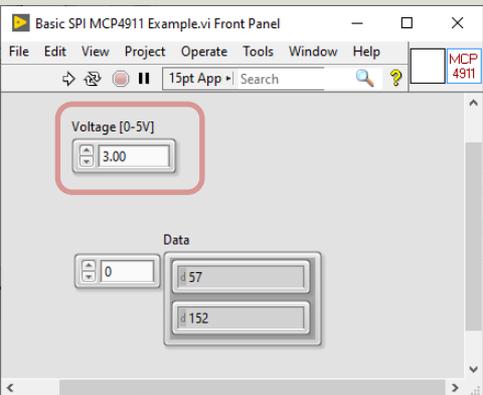
The different MCP49xx DACs work in the same manner, the only difference is the resolution (8, 10, or 12 resolution)

Datasheet: <https://www.microchip.com/en-us/product/MCP4911>

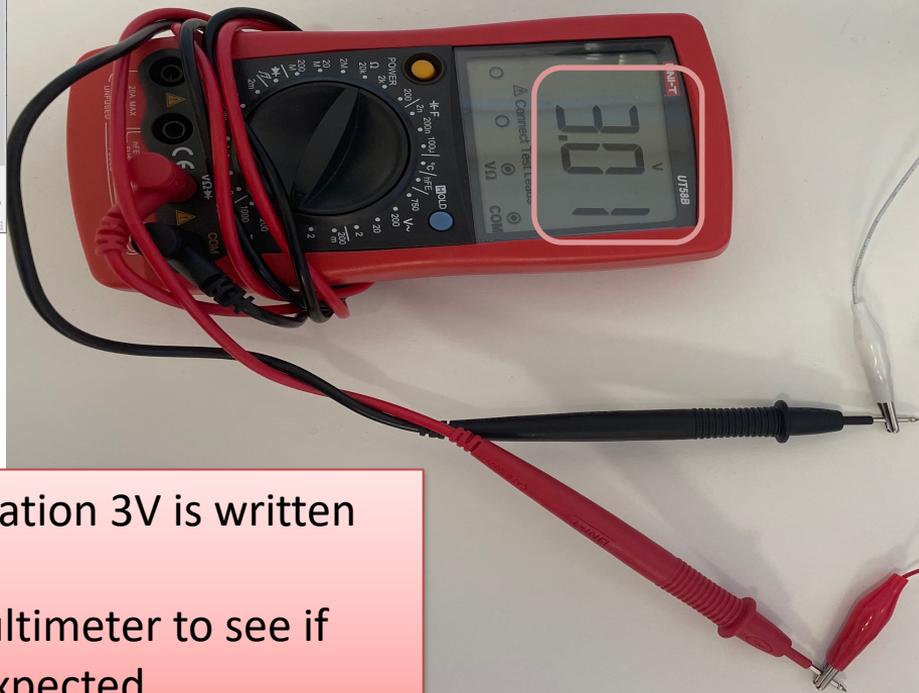
MCP4911 - Arduino Wiring



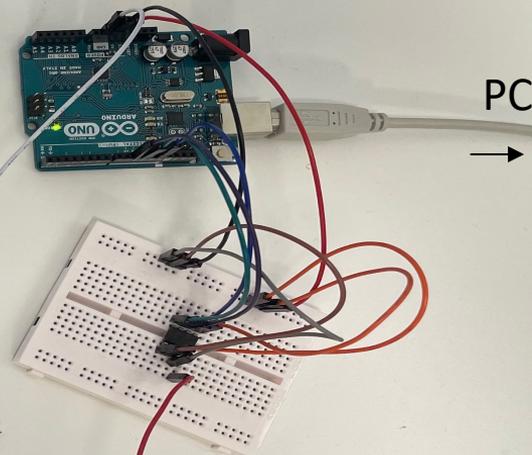
Test Setup



Multimeter

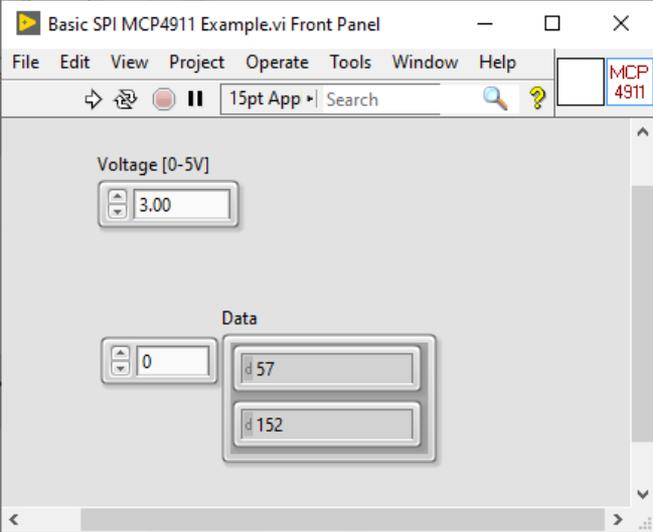


Arduino UNO

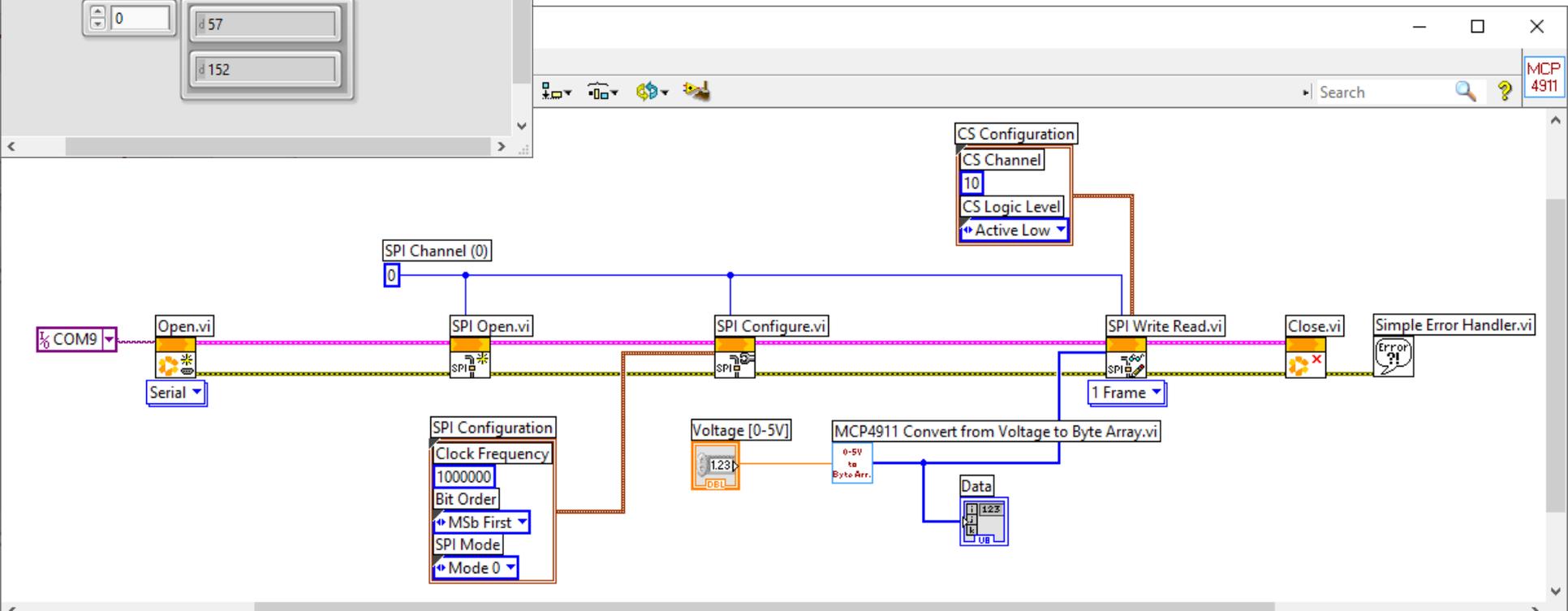


Breadboard with
MCP4911 DA and some
wires to the Arduino

In the LabVIEW Application 3V is written to the MCP4911 DAC.
Then we can use a Multimeter to see if everything works as expected



LabVIEW

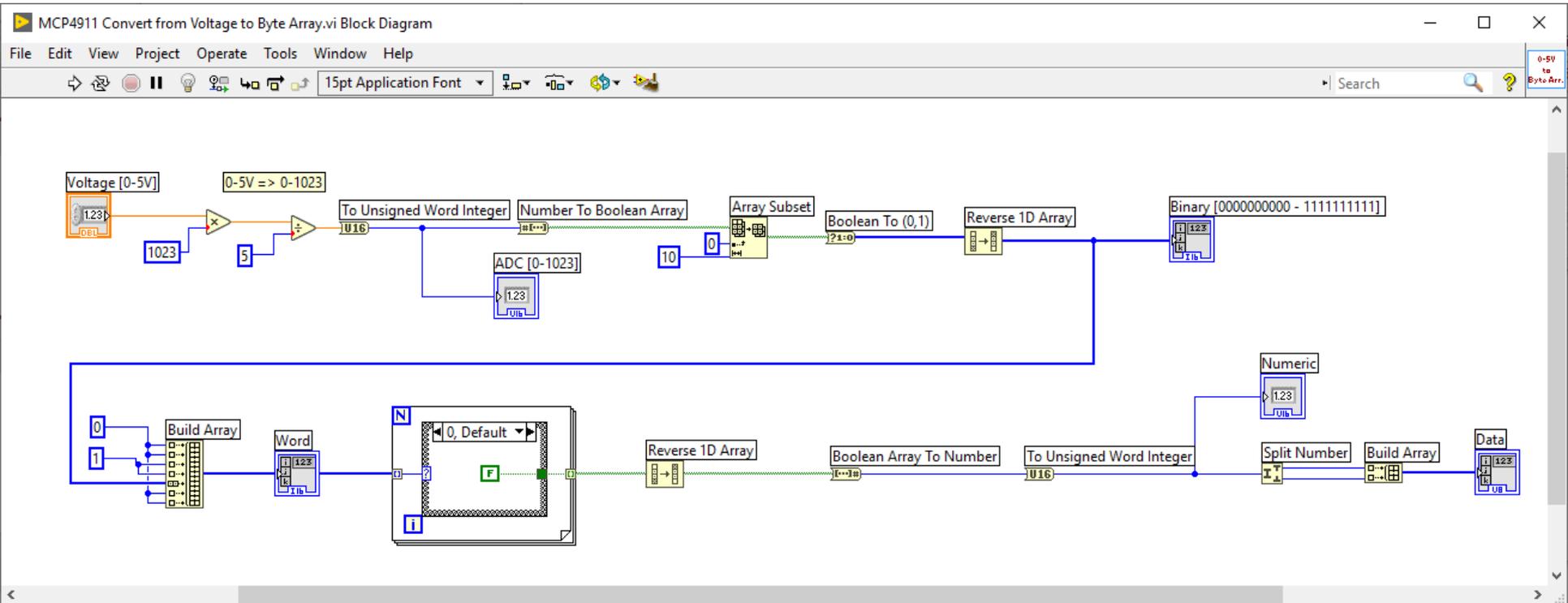


Convert from Voltage to Byte Array

The screenshot displays the front panel of a LabVIEW virtual instrument (VI) titled "MCP4911 Convert from Voltage to Byte Array.vi Front Panel". The interface includes a menu bar (File, Edit, View, Project, Operate, Tools, Window, Help) and a toolbar with various icons. The main panel features several controls:

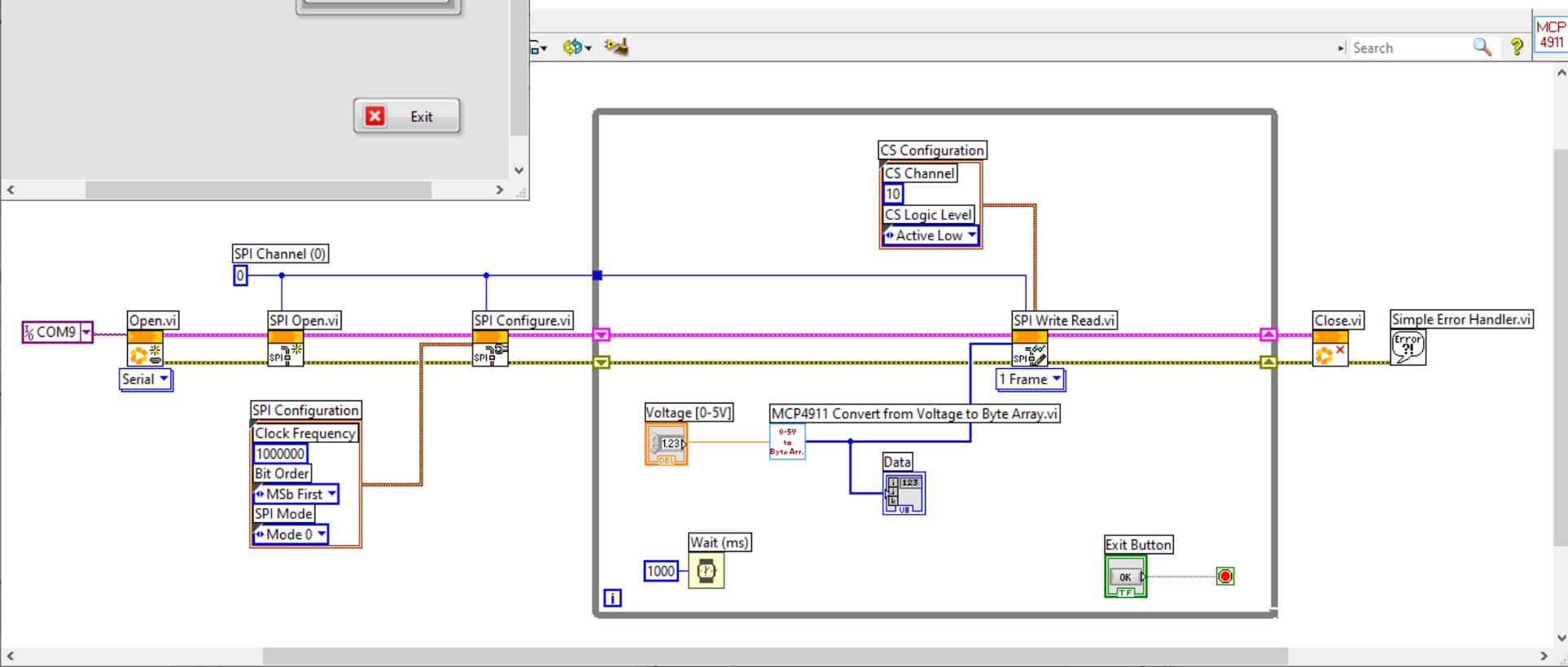
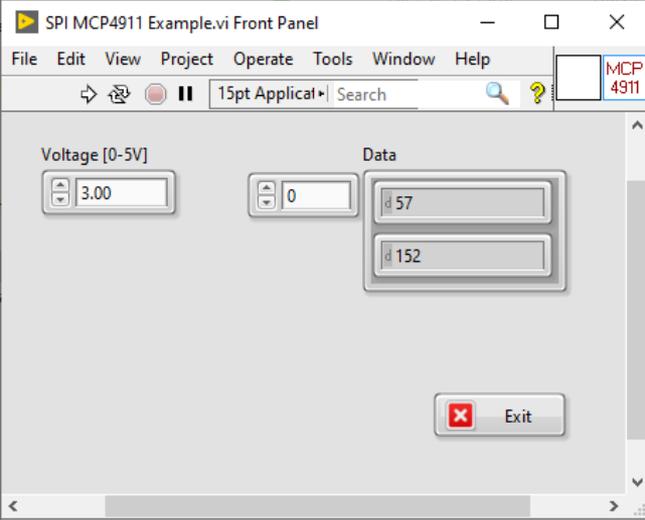
- Voltage [0-5V]:** A numeric control with a value of 0.000.
- ADC [0-1023]:** A numeric control with a value of 0.
- Numeric:** A numeric control with a value of 14336.
- Binary [0000000000 - 1111111111]:** A binary display showing 16 bits, with the first three bits (001) highlighted in bold. The bits are: 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0.
- Word:** A word display showing 16 bits, all of which are 0.
- Data:** A data control with a value of 0 and two indicators labeled "d0".

Convert from Voltage to Byte Array



LabVIEW

Continuous Writing Example:





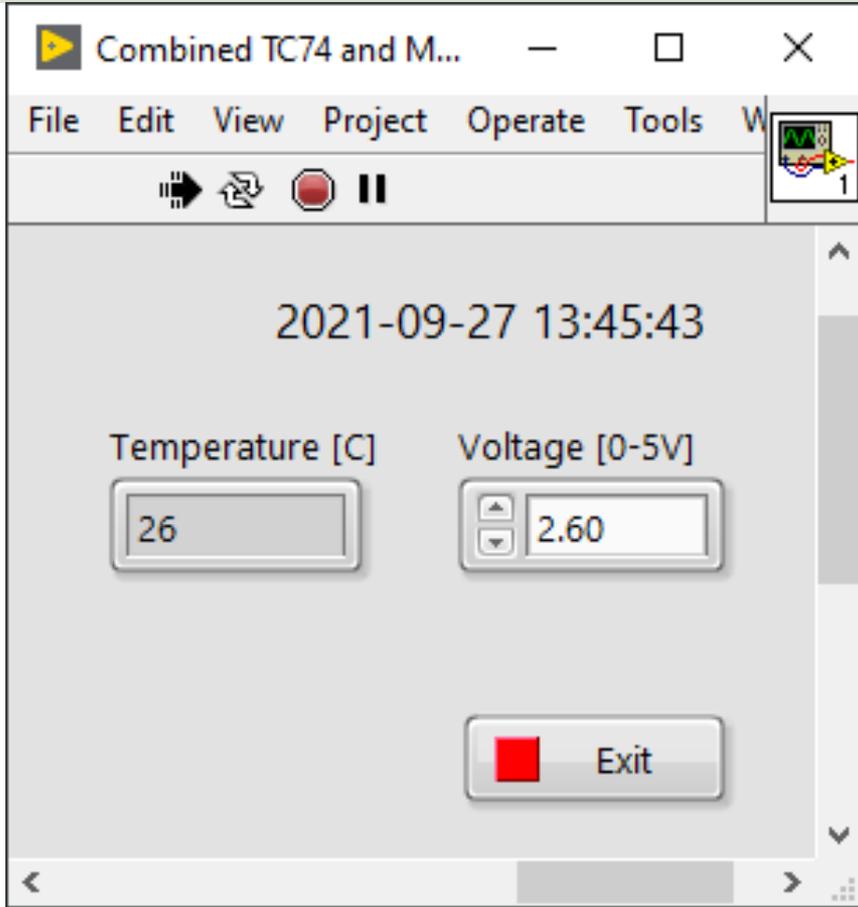
Combined System

TC74 + MCP4911

I2C

SPI

TC74 (I2C) + MCP4911 (SPI)



Here is a basic example presented where reading TC74 Temperature Data is combined with writing values to the MCP4911 DAC.

It can easily be extended with, e.g., a PID Control System

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

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

Web: <https://www.halvorsen.blog>

