ME:4145 Industrial Internet of Things

Lab #8
Analog to Digital Conversion & GUIs/HMIs (2)

Introduction

This lab will introduce you to creating a web application to act as a human machine interface (HMI). You will be replicating Lab 7 with the web application instead of a local graphical user interface (GUI). The content and directions of this lab is the same as Lab 7 (same directions and deliverables). The following is include for reference only. A basic working example can be found in the lab folder.

Getting Started

What you will need

To complete this lab, you will need the following components:

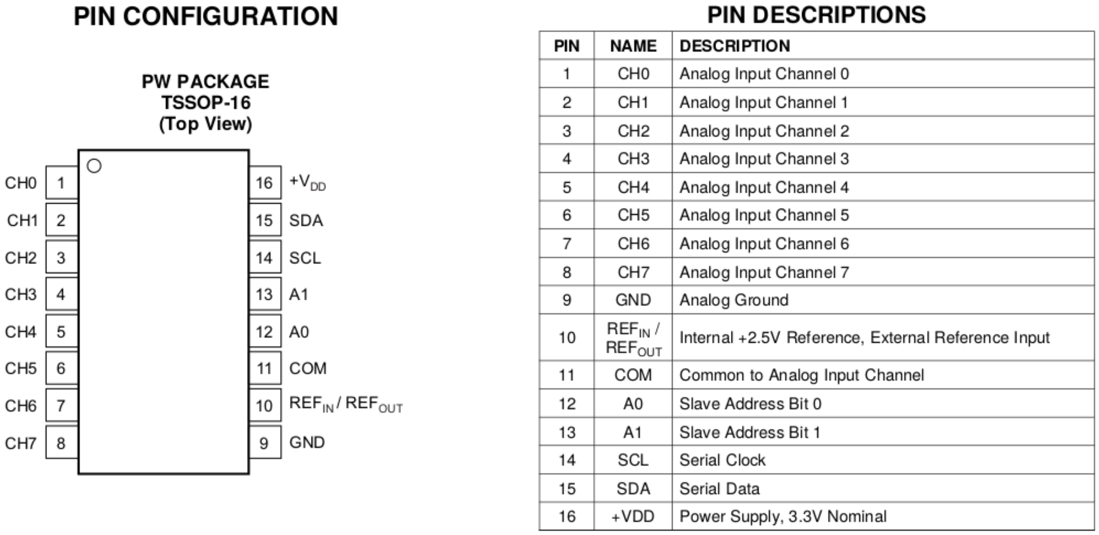
- ADS7830 or PCF8591 ADC
- 10k thermistor
- 10 kΩ resistor (3 if using the PCF8591)
- breadboard
- jumper wires

ADS7830 ADC

The ADS7830 is an 8-bit ADC that is supplied by a nominal voltage of 3.3V. The pin configuration and pin descriptions are provided below in Fig. 1. Further information is provided in the data sheet included in the lab folder.

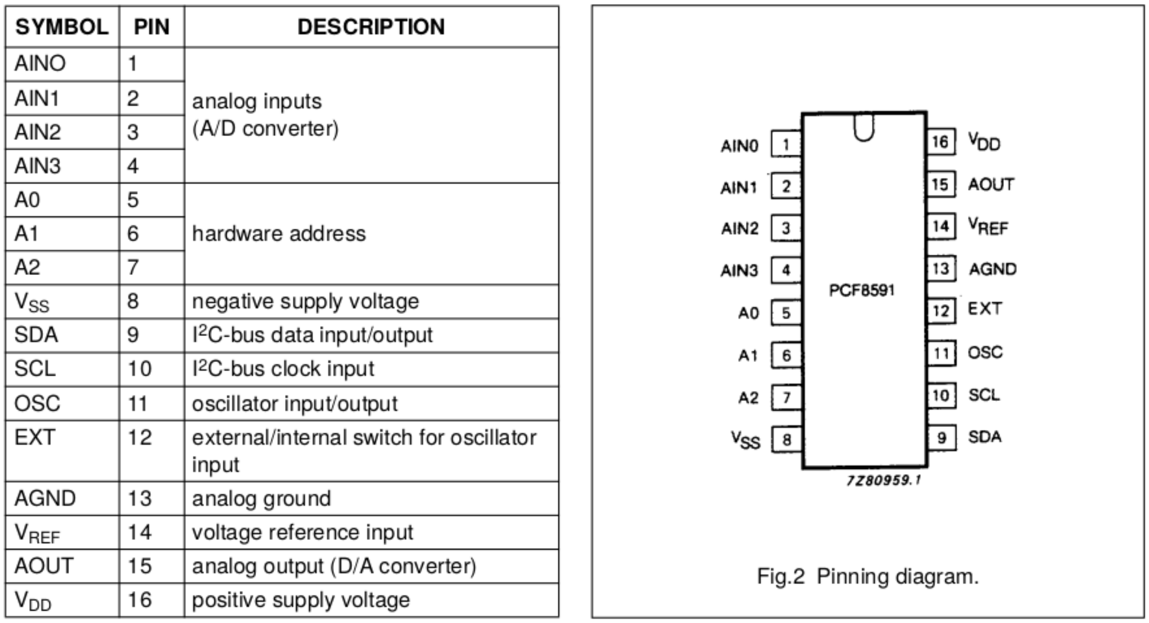
PCF8591 ADC

If your kit does not contain the ADS7830, it will likely contain the PCF8591. The PCF8591 is also an 8-bit ADC. The The pin configuration and pin descriptions are shown in Fig. 2 and the data sheet can be found in the lab folder.



PIN	NAME	DESCRIPTION
1	CH0	Analog Input Channel 0
2	CH1	Analog Input Channel 1
3	CH2	Analog Input Channel 2
4	CH3	Analog Input Channel 3
5	CH4	Analog Input Channel 4
6	CH5	Analog Input Channel 5
7	CH6	Analog Input Channel 6
8	CH7	Analog Input Channel 7
9	GND	Analog Ground
10	REF_{IN} / REF_{OUT}	Internal +2.5V Reference, External Reference Input
11	COM	Common to Analog Input Channel
12	A0	Slave Address Bit 0
13	A1	Slave Address Bit 1
14	SCL	Serial Clock
15	SDA	Serial Data
16	+VDD	Power Supply, 3.3V Nominal

Figure 1: ADS7830 pin configuration and descriptions.



SYMBOL	PIN	DESCRIPTION
AIN0	1	analog inputs (A/D converter)
AIN1	2	
AIN2	3	
AIN3	4	
A0	5	hardware address
A1	6	
A2	7	
V_{SS}	8	negative supply voltage
SDA	9	I^{2}C-bus data input/output
SCL	10	I^{2}C-bus clock input
OSC	11	oscillator input/output
EXT	12	external/internal switch for oscillator input
AGND	13	analog ground
V_{REF}	14	voltage reference input
AOUT	15	analog output (D/A converter)
V_{DD}	16	positive supply voltage

Figure 2: PCF8591 pin configuration and descriptions.

Thermistor

A thermistor is a temperature sensitive resistor. When it senses a change in temperature, the resistance of the thermistor will change. We can take advantage of this characteristic by using a thermistor to detect temperature intensity. A thermistor and its electronic symbol are shown if Fig. 3.

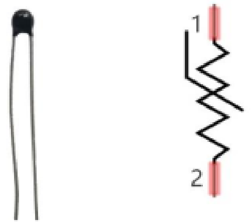


Figure 3: Thermistor and its electronic symbol.

The relationship between resistance value and temperature of a thermistor is

$$R_2 = R_1 \exp^{B(\frac{1}{T_2} - \frac{1}{T_1})}$$

where R_1 and R_2 are the thermistor resistance at temperatures T_1 and T_2 in Kelvin, respectively, and B is thermal constant. For the thermistor used in this lab, $B = 3950$ K and $R_1 = 10$ kΩ at $T_1 = 298.15$ K. To calculate the resistance R_2 we will use a voltage divider and measure the voltage using the ADC as shown in Fig. 4.

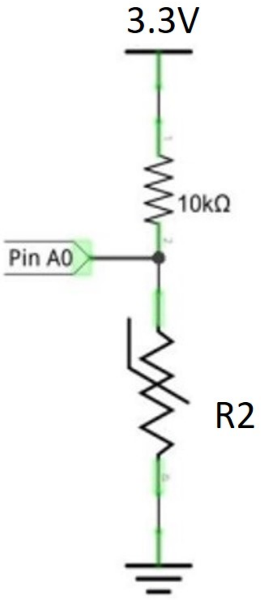


Figure 4: Thermistor voltage divider.

Using the voltage divider and the voltage measured by the ADC on pin A0, the resistance R_2 of the thermistor at the current temperature can be determined. Using R_2 the temperature of the can be calculated as

$$T_2 = \frac{1}{\frac{1}{T_1} + ln(\frac{R_2}{R_1})/B}$$

Raspberry Pi Connections

ADS7830 ADC wiring

Connections from the Raspberry Pi to the ADS7830 and the thermistor to the ADS7830 are shown in Fig. 5.

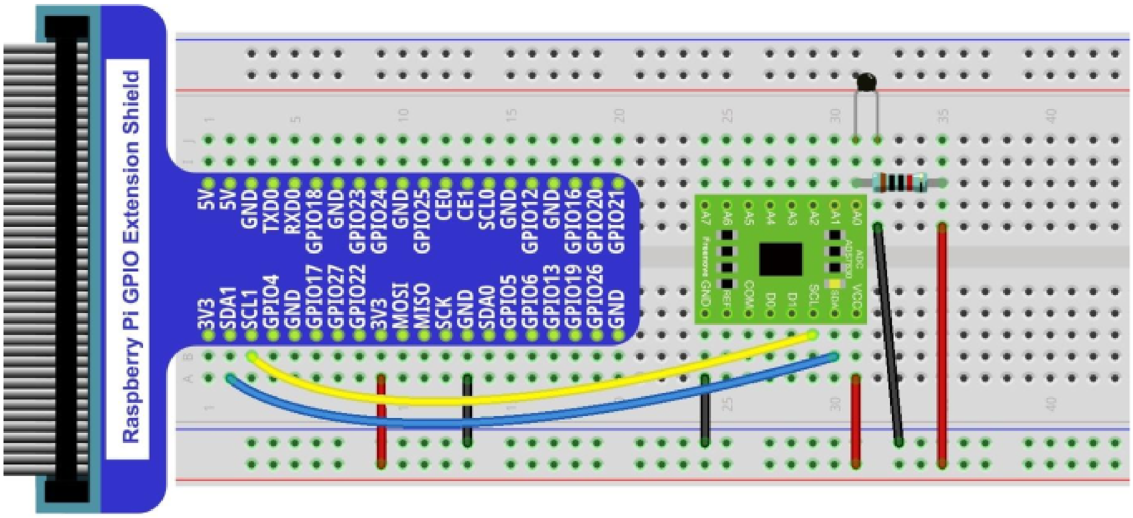


Figure 5: Raspberry Pi wiring for the ADS78360 ADC and thermistor.

PCF8591 ADC wiring

If you are using the PCF8591 the equivalent wiring is shown in Fig. 6.

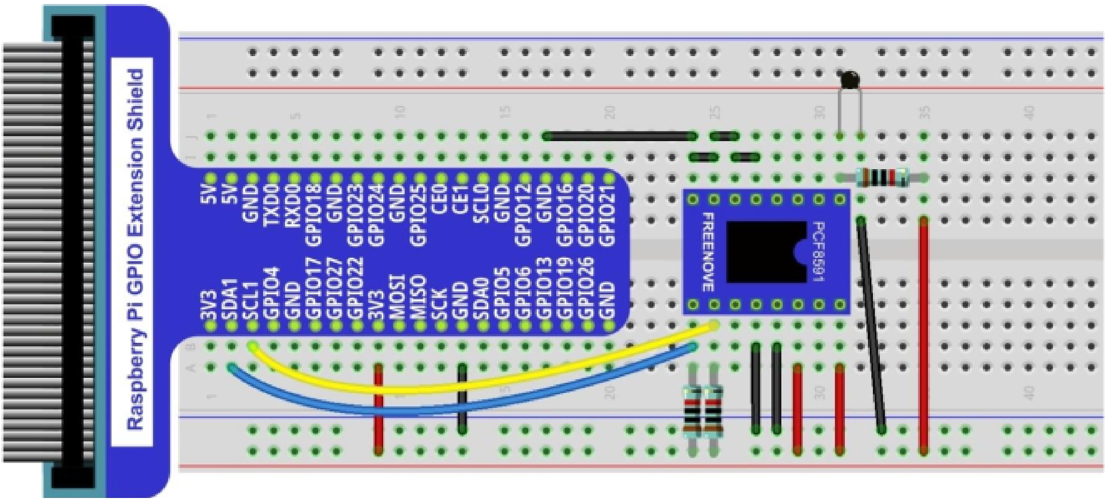
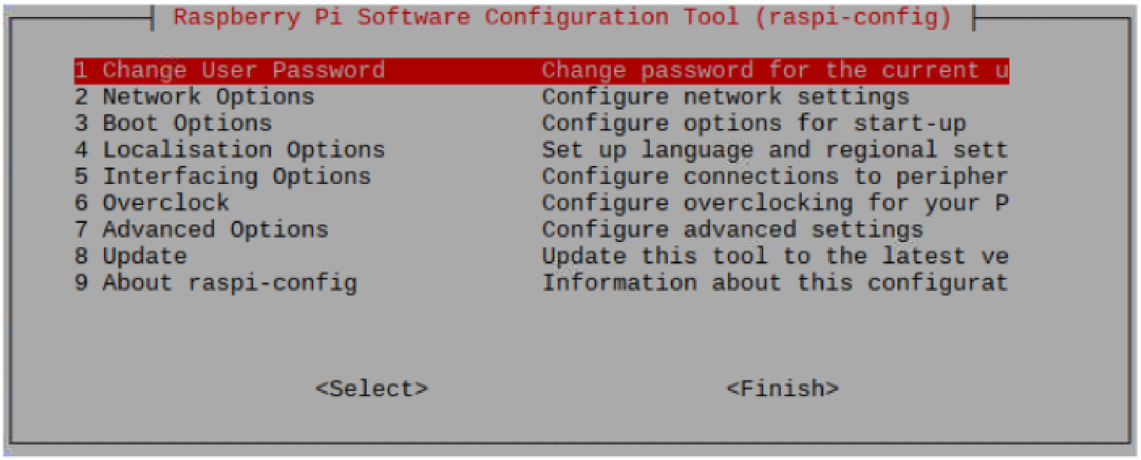


Figure 6: Raspberry Pi wiring for the PCF8591 ADC and thermistor.

Raspberry Pi Software

To use the ADC with the Raspberry Pi we need to enable the I2C interface. To do this open a terminal and type "sudo raspi-config" which will open the dialog box as seen in Fig. 7.



```
Raspberry Pi Software Configuration Tool (raspi-config)

1 Change User Password        Change password for the current u
2 Network Options             Configure network settings
3 Boot Options                Configure options for start-up
4 Localisation Options        Set up language and regional sett
5 Interfacing Options         Configure connections to peripher
6 Overclock                   Configure overclocking for your P
7 Advanced Options            Configure advanced settings
8 Update                      Update this tool to the latest ve
9 About raspi-config          Information about this configurat


              <Select>                  <Finish>
```

Figure 7: Raspi-config menu.

From this menu choose "5 Interfacing Options" then "P5 I2C" then "Yes" and then "Finish". In order for this change to take effect you will need to restart your Raspberry Pi.

Next, we need to install some software. Open a terminal and type the following

```
sudo apt install i2c-tools
```

Once the above has completed installing, install the smbus library

```
sudo apt-get install python3-smbus
```

Finally, download the archive "ADC_Device.tar.gz" from the lab assignment. Now open a terminal and change to the directory where the achive was downloaded to (most likely the Downloads folder). To do this type the following into the terminal

```
cd  Downloads
```

Extract the archive with the following command

```
tar zxvf ADC_Device.tar.gz
```

Navigate to the extracted folder using the following

```
cd ADC_Device
```

Now install the library

```
sudo python3 setup.py install
```

Task

Write a Python program that will do the following:

1. Push the current temperature to a Google Sheet database at a user-defined and modifiable rate, e.g., **once** every **5** minutes.
   - The data should include a **time** and **date** stamp and **temperature** (°F) (**averaged** over **10** readings). Format your data as follows:
     (a) date: year-mm-dd
     (b) time: hr:min:sec
     (c) round the temperature to three decimal places
2. Create a GUI/HMI that allows the user to change the update rate (frequency). This can be via text entry, slider, buttons, etc. Furthermore, the program should display the date, time, and temperature (not averaged)with a refresh rate of 1 second.

Submission Materials

Upload the following to ICON no later than the due date:

1. Python script/s

2. Screenshot of your GUI

3. Copy of your google sheet containing at least **2** hours worth of data.

   - Although not necessary, consider placing your sensor somewhere where the temperature will vary. Otherwise your data might be fairly boring.