

1 ACD - Analog to Digital Conversion

An ACD converter is needed to translate analog data (temperature) from the thermistor to digital data (voltage) that the RaspberryPi can understand. If I were using an Arduino I would not need such a converter as an Arduino has an ADC converter built in where as the RaspberryPi is a digital only microcontroller. Using the MCP3008 for ADC, its pin out:

- VDD - power
- DGND - Digital ground

used to power the converter

- DOUT - data out of the coverter (the covnerted data we desire)
- CLK - clock pin
- DIN - data in from raspberry pi
- CS - Chip select

These 4 pins are used for the SPI (Serial Peripheral Interface) of the raspberry pi. Section 2 gives more on this.

- AGND - analog ground (used for precision which is optional and so can merely be connected to GND)
- Vref - analog reference voltage (used as a scale for the ADC)

Everything above comes from Analog Inputs for Raspberry Pi Using the MCP3008. Discussed below is how we can use this digital data from the ADC converter to translate the data into what we expect, see Analog to Digital Conversion by Adapfruit for more.

Consider a microcontroller is powered with 5v, it understands 0v as a binary 0 and 5v as a binary 1. Now we have a problem when the voltage is anything between 0 and 5 volts (what will be binary 0 and what is binary 1) - this is analog data and is why micocontrollers have difficult time translating what analog data means, this is why we need an ADC converter, to understand analog data in a form it understands (digital data). The MCP3008 ADC converter is a 10-bit converter, meaning it can detect $2^{10} = 1024$ discrete analog levels. The ADC is a ratio value given as $\frac{\text{Resolution of the ADC converter}}{\text{System Voltage}} = \frac{\text{ADC Reading}}{\text{Analog Voltage Measured}}$. This means the converter will use the value 1023 to depict the system voltage and value between 0 and 1024 will be a ratio between the 5V and the ADC value 1023. The system voltage of the raspberry pi is either 3.3V or 5V depending on the voltage pin used to power the converter. Thus we can get the *analog voltage* from this equation that uses ADC values. As an example, imagine we are measuring a voltage with our Pi and the system voltage we are using is 5V, lets assume the ADC value measured on the Pi is 434 and we are using a 10-bit ADC converter. Now to get the analog voltage from this we do the following

$$\frac{1023}{5} = \frac{434}{\text{Analog Voltage Measured}} \quad (1)$$
$$\text{Analog Voltage Measured} = \frac{2.12V}{1}$$

2 SPI - Serial Peripheral Interface

SPI is a communication protocol. SPI is used for synchronous communication between devices (a master and a slave). The 4 connections of SPI:

- SCLK - clock
- MOSI - Master Out Slave In which is a data line from the master device to the slave device
- MISO - Master In Slave Out which is another data line for sending data from a slave to a master
- SS - Slave Select used to tell the slave that it must be on the 'look out' for data (either sending or receiving)

As the pins suggested, devices operating with SPI can act either as a slave or as a master. The SS pin also hints at the ability to connect multiple slaves to a master device. We can have 3 slaves and we can select which one we want to communicate with (either to receive data from them or to send data to) by applying or not applying voltage to the slave's SS connection of interest. To start communication with a slave device we set the SS pin to low.

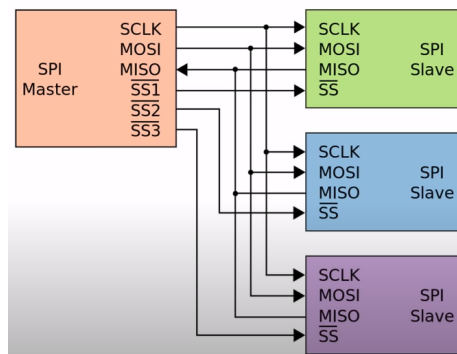


Figure 1: Image from nanland

SPI is full duplex meaning you can send and receive data at the same time. It is not suited for long distance communication (nanland goes as far as saying that even more than a meter is not suited for SPI).

Now when it comes to programming the microcontroller for SPI communication, there are 4 modes of SPI we can set that correspond to when data must be read based on the clock signal send from master to slave via the CLK pin. These 4 modes can be

1. Read/write when clock is low and on rising edge
2. Read/write when clock is high and on rising edge

3. Read/write when clock is low and on falling edge
4. Read/write when clock is high and on rising edge

Great Scott explains the modes rather well. The mode to use depends on the device being used, see the datasheet of the device for more.

SPI transfers data at a CRAZY rate and is part in part the reason for its existence

3 Thermistor - Reading Temperature From Voltage

An article titled Thermistors/Temperature Measurement with NTC Thermistors was referenced for this section of the research.