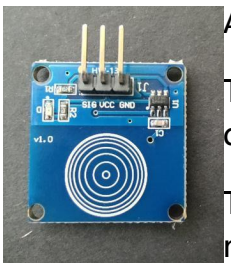
Analog and Digital Sensors

Analog sensor data

Analog sensors produce a continuously varying output voltage signal that can be measured by an analog pin on your microcontroller board.



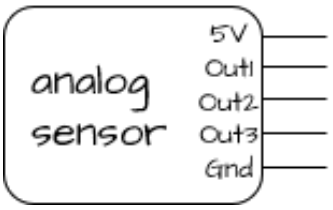
An analog sensor

The measurement passes the sensor's output voltage through an analog to digital converter (ADC).

This value can then be read and processed by program code with a special read command specific to the microcontroller used by your microcontroller board. ADCs in microcontrollers typically have 8-16 bits of resolution (256-65536 possible values), so storing ADC output data as an integer will work, although you may wish to consider unsigned byte or similar data types to reduce memory overhead.



Many sensors have 3 pins - one each for power, ground, and the analog output voltage.



The pins can be positioned in any order and can also have slightly different names.

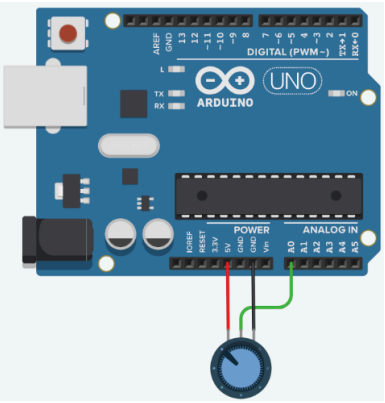
The diagrams above show a standard 3-pin sensor configuration as well as another analog sensor that produces 3 separate output signals (for example a 3-axis accelerometer).

Common pin names for each purpose are:

- Power - Vcc, Vss, Vin, 5V, 3.3V, 3-5V, +
- Ground - Gnd, -
- Analog output signal - Vo, Vout, Out, Sig, D.

Task:

Modify the analog sensor data simulation on Wokwi to determine the range of available values. Add this code to the run routine:



```
None
analogValue=potentiometer.read_u16()    #Read value from pin
print(analogValue)                      #Show value in monitor
sleep(0.1)                              #Wait 100ms
```

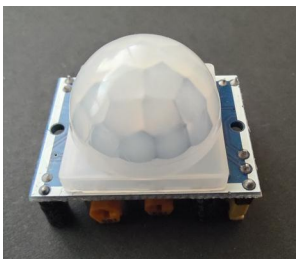
Output:

Max value:

Min value:

What is the relationship between the max output value and the variable potentiometer.read_u16?

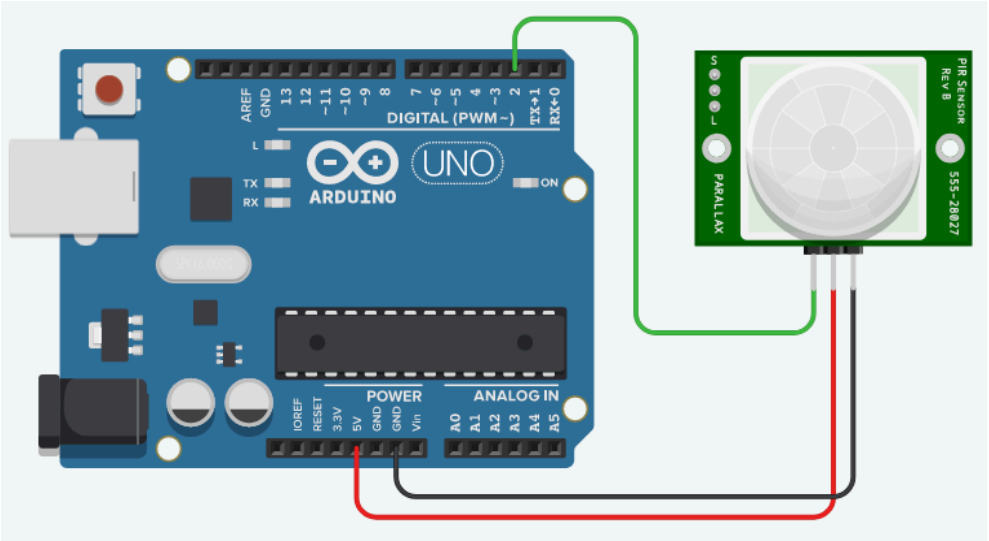
On/off digital sensor data



On/off digital sensors come with similar pinouts to analog sensors, but the output is either a high or a low voltage.

They are useful when the presence (or absence) of something needs to be indicated rather than a range of values describing something about the detection.

On/off digital data is most efficiently stored using a Boolean data type and can also be used directly within control structure conditions.



This is an example with an Arduino and sensor. Your example in Wokwi is with a pico. It's the same sort of setup, and this will help you work.

Task:

Modify the digital sensor data simulation on Wokwi to connect a PIR motion sensor to the microcontroller board.

The sensor needs power and should be connected to a digital pin used by the code.

```python
from machine import Pin
from time import sleep

PIR=Pin(28,Pin.IN)              #Configure pin 28 for digital input

while True:
    digitalValue=PIR.value()    #Get the value at the pin
    print(digitalValue)
    sleep(0.1)
```

Modify the code and circuit to use a different microcontroller pin.

Use the Serial monitor to verify that the circuit still works.

Then copy your code and an image of your circuit into the table below.

Code to read sensor value and produce output	Image of circuit

Then: connect the PIR motion sensor to an analog input and adjust the code to check whether the value being read from the analog to digital converter is above or below a threshold value.

Include your completed code here:

```python
```