

Student name: Zakarya Guerinat

Student ID: 217090531

SIT225: Data Capture Technologies

Activity 2.1: Working with sensor - DHT22

DHT22 is a temperature and humidity sensor.

Hardware Required

Arduino Board

DHT22 sensor

USB cable

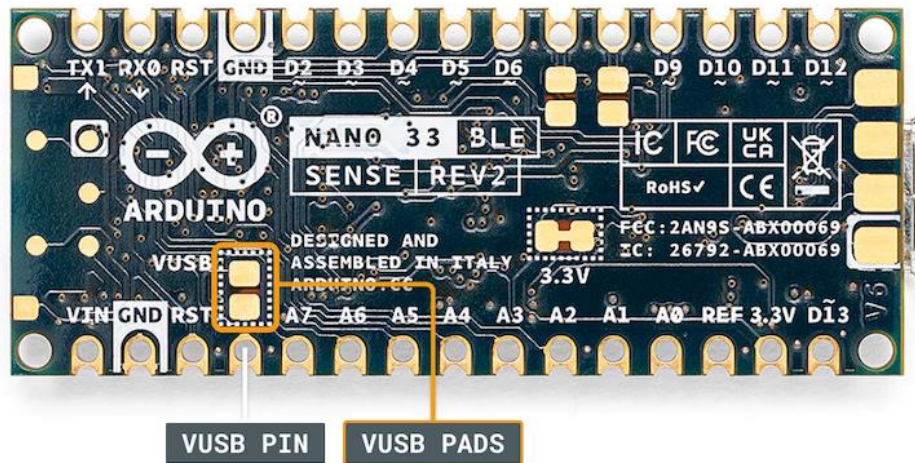
Software Required

Arduino programming environment

Known issue, action required

Arduino Nano 33 IoT board operates on 3.3 V, it needs to be arranged to make it 5 V. The Arduino board has a pin called VUSB or VBUS and there are 2 pads next to the pin. **These two pads must be shorted to enable the pin** (see detail here

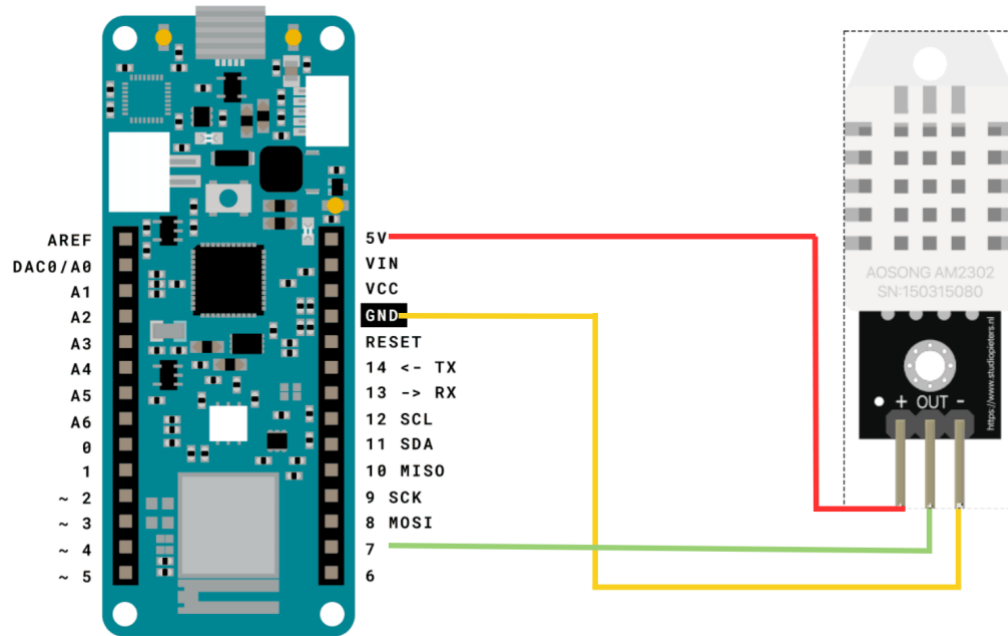
<https://support.arduino.cc/hc/en-us/articles/360014779679-Enable-5-V-power-on-the-VUSB-or-VBUS-pin-on-Nano-boards>).



To test, you can use a wire and connect these 2 pads manually by hand to see if data is coming through DHT22 sensor. For long data collection, you will need to solder the wire permanently to connect the pads. Seeking help from tutors there is an on-campus facility called Maker Space where you can do it.

Steps:

Step	Action
1	<p>Connect your DHT22 Temperature and Humidity Sensor to the Arduino board. Note that the pin layout in the image below may look different than the board you may have.</p>



- Pick a red male-female jumper wire and attach the female end to pin 1 (VCC pin) on the sensor. Plug the male end into the Arduino board's 5V power pin.
- Pick a blue male-female jumper wire and attach the female end to pin 2 (DATA pin) on the sensor. Plug the male end into the Arduino board's digital data pin 2.
- Pick a black male-female jumper wire and attach the female end to pin 4 (GND) on the sensor. Plug the male end into the Arduino board's GND pin.
- Sensor's pin 3 is not used.

2 Connect your Arduino board to your computer using the USB cable.

3 Write an Arduino sketch (or download it from https://github.com/deakin-deep-dreamer/sit225/blob/main/week_2/sketch_dht22.ino) which looks like below. Compile the code in Arduino IDE, deploy to the board and observe output in the Arduino IDE serial monitor.

	<pre> 6 7 // Add "DHT sensor library" 8 #include <DHT.h> 9 10 #define DHTPIN 2 // digital pin number 11 #define DHTTYPE DHT22 // DHT type 11 or 22 12 DHT dht(DHTPIN, DHTTYPE); 13 14 // variables to store data. 15 float hum, temp; 16 17 void setup() { 18 // Set baud rate for serial communication 19 Serial.begin(9600); 20 21 // initialise DHT library 22 dht.begin(); 23 } 24 25 void loop() { 26 // read data 27 hum = dht.readHumidity(); 28 temp = dht.readTemperature(); 29 30 // Print data to serial port - a long way 31 // 32 // Serial.print("Humid: "); 33 // Serial.print(hum); 34 // Serial.print(" %, Temp: "); 35 // Serial.print(temp); 36 // Serial.println(" Celsius"); 37 38 // Print data to serial port - a compact way 39 Serial.println(String(hum) + "," + String(temp)); 40 41 // wait a while 42 delay(15*1000); 43 } </pre>	
4	<p>Question: A spec of the DHT22 sensor is given in the link below. It mentions that the sampling rate is 0.5 Hz.</p> <p>https://lastminuteengineers.com/dht11-dht22-arduino-tutorial</p> <ul style="list-style-type: none"> i) What does the sampling rate mean? ii) Where is this used in the Arduino code? <p>Answer: The sampling rate means how often the Arduino reads data from the sensor, its shown in the code by <code>delay(2000);</code> 2000ms = 2 secs</p>	
5	<p>Question: Take a screenshot of your Serial Monitor displaying temperature & humidity sensor data logs. Add the image here.</p>	

Answer:



The screenshot shows the Serial Monitor window in an IDE, connected to an Arduino NANO 33 IoT. The window displays a stream of data consisting of humidity and temperature readings. The humidity values range from 59.40% to 59.80%, and the temperature values range from 20.70 °C to 21.30 °C. The data is presented in two columns, with humidity on the left and temperature on the right. The window has a title bar with '41' and 'Serial Monitor'. The status bar at the bottom indicates 'Message (Enter to send message to 'Arduino NANO 33 IoT' on '/dev/cu.usb...')', 'New Line', and '9600 baud'.

Humidity	Temperature
Humidity: 59.40 %	Temperature: 21.30 °C
Humidity: 59.40 %	Temperature: 21.20 °C
Humidity: 59.40 %	Temperature: 21.20 °C
Humidity: 59.50 %	Temperature: 21.10 °C
Humidity: 59.60 %	Temperature: 21.00 °C
Humidity: 59.70 %	Temperature: 21.00 °C
Humidity: 59.70 %	Temperature: 20.90 °C
Humidity: 59.80 %	Temperature: 20.90 °C
Humidity: 59.80 %	Temperature: 20.80 °C
Humidity: 59.80 %	Temperature: 20.70 °C
Humidity: 59.80 %	Temperature: 20.70 °C

Activity 2.2: Working with sensor - HC-SR04

HC-SR04 is an Ultrasonic sensor.

Hardware Required

Arduino Board

HC-SR04 Ultrasonic sensor

USB cable

Software Required

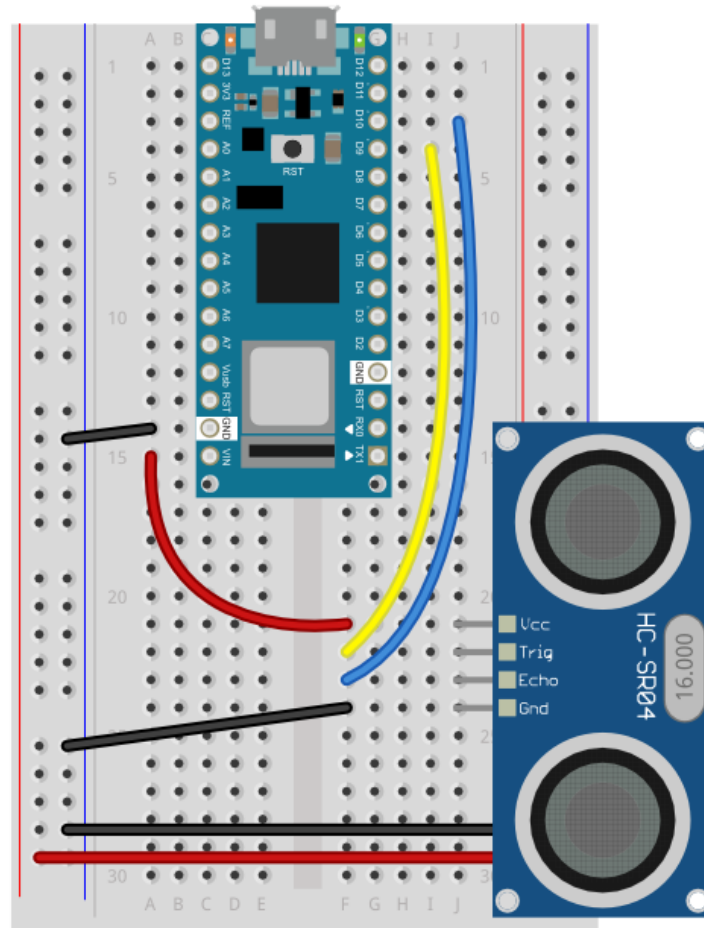
Arduino programming environment

Known issue, action required

The same known issue applies to SR04 which operates at 5 V source while the Arduino Nano 33 IoT supplies 3.3 V. It requires 2 VUSB (or VBUS) pads to be shorted to enable the pin.

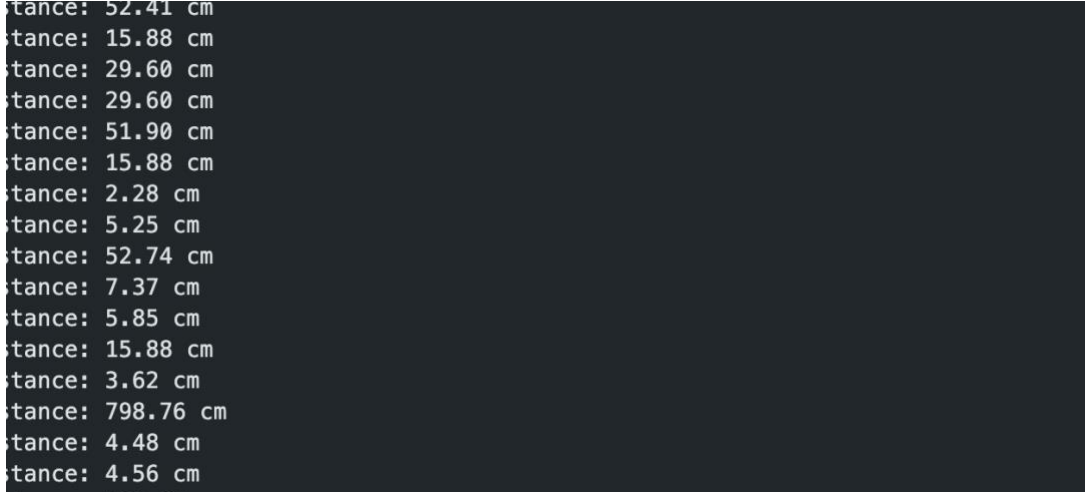
Steps:

Step	Action
1	Connect your HC-SR04 sensor to the Arduino board. Note that the pin layout in the image below may look different than the board you may have.



2	Connect your Arduino board to your computer using the USB cable.
3	Write an Arduino sketch (or download it from https://github.com/deakin-deep-dreamer/sit225/blob/main/week_2/sketch_hcsr04_distance.ino) which looks like below. Compile the code in Arduino IDE, deploy to the board and observe output in the Arduino IDE serial monitor.

	<pre> 7 const int trigger = 2; 8 const int echo = 3; 9 10 int getUltrasonicDistance(){ 11 // Function to retrieve the distance reading of the ultrasonic 12 long duration; 13 int distance; 14 15 // Assure the trigger pin is LOW: 16 digitalWrite(trigger, LOW); 17 // Brief pause: 18 delayMicroseconds(5); 19 20 // Trigger the sensor by setting the trigger to HIGH: 21 digitalWrite(trigger, HIGH); 22 // Wait a moment before turning off the trigger: 23 delayMicroseconds(10); 24 // Turn off the trigger: 25 digitalWrite(trigger, LOW); 26 27 // Read the echo pin: 28 duration = pulseIn(echo, HIGH); 29 // Calculate the distance in centimeter (CM): 30 distance = duration * 0.034 / 2; 31 32 // Uncomment this line to return value in IN instead of CM: 33 //distance = distance * 0.3937008 34 35 // Return the distance read from the sensor: 36 return distance; 37 } 38 39 void setup() { 40 // Define inputs and outputs: 41 pinMode(trigger, OUTPUT); 42 pinMode(echo, INPUT); 43 44 // Start the serial monitor: 45 Serial.begin(9600); 46 } 47 48 void loop() { 49 // Print the distance to the serial monitor: 50 Serial.print("Distance: "); 51 Serial.println(getUltrasonicDistance()); 52 53 // Wait one second before continuing: 54 delay(1000); 55 } </pre>
4	<p>Question: Spec of SR04 is available here (https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf). Identify 2 critical aspects you should be careful about this sensor operation.</p> <p>Answer: 1 always connect the GND wire first before powering it on 2 make sure the object is at least 0.5 square metres in size on a surface that is flat and smooth</p>

5	<p>Question: Take a screenshot of your Serial Monitor displaying distance values while you try to generate a periodic motion by moving your hand gradually back and forth towards the sensor. Add the image here.</p> <p>Answer:</p>  <pre>Distance: 52.41 cm Distance: 15.88 cm Distance: 29.60 cm Distance: 29.60 cm Distance: 51.90 cm Distance: 15.88 cm Distance: 2.28 cm Distance: 5.25 cm Distance: 52.74 cm Distance: 7.37 cm Distance: 5.85 cm Distance: 15.88 cm Distance: 3.62 cm Distance: 798.76 cm Distance: 4.48 cm Distance: 4.56 cm</pre>

Activity 2.3: Working with sensor - Accelerometer

LSM6DS3 module on the Arduino Nano 33 IoT is an accelerometer and gyroscope sensor.

Hardware Required

Arduino Nano 33 IoT Board (has inbuilt LSM6DS3 module)

USB cable

Software Required

Arduino programming environment

Steps:

Step	Action
1	Write Arduino sketch (or download code from https://github.com/deakin-deep-dreamer/sit225/blob/main/week_2/sketch_accelero.ino) which looks like below. Compile the code in Arduino IDE, deploy to the board and observe output in the Arduino IDE serial monitor.

	<pre> 6 7 // Add Arduino_LSM6DS3 library 8 // from Arduino IDE Library Manager. 9 #include <Arduino_LSM6DS3.h> 10 11 float x, y, z; 12 13 void setup() { 14 Serial.begin(9600); // set baud rate 15 while (!Serial); // wait for port to init 16 Serial.println("Started"); 17 18 if (!IMU.begin()) { 19 Serial.println("Failed to initialize IMU!"); 20 while (1); 21 } 22 23 Serial.println(24 "Accelerometer sample rate = " 25 + String(IMU.accelerationSampleRate()) + " Hz"); 26 } 27 28 void loop() { 29 // read accelero data 30 if (IMU.accelerationAvailable()) { 31 IMU.readAcceleration(x, y, z); 32 } 33 34 Serial.println(35 String(x) + ", " + String(y) + ", " + String(z)); 36 37 delay(1000); 38 } </pre>
2	<p>Question: Spec of LSM6DS3 is available here (https://content.arduino.cc/assets/st_imu_lsm6ds3_datasheet.pdf). Identify at least 3 attributes of this sensor you think important to work with.</p> <p>Answer:</p> <ul style="list-style-type: none"> a) Measurement range b) Sampling rate c) Interrupt capabilities
3	<p>Question: Take a screenshot of your Serial Monitor displaying sensor readings. Add the image here.</p>

	<p>Answer:</p> <pre> Accel (m/s²) -> X: -1.31 Y: 0.68 Z: 1.12 Gyro (°/s) -> X: 179.32 Y: -426.70 Z: -28.56 Accel (m/s²) -> X: -1.73 Y: 0.71 Z: 1.79 Gyro (°/s) -> X: 172.24 Y: -403.02 Z: -15.93 Accel (m/s²) -> X: -2.77 Y: -0.46 Z: 2.13 Gyro (°/s) -> X: 133.24 Y: -256.35 Z: -67.14 Accel (m/s²) -> X: -0.82 Y: 0.80 Z: 0.77 Gyro (°/s) -> X: 271.06 Y: -399.23 Z: -53.89 Accel (m/s²) -> X: 0.60 Y: 1.58 Z: -0.17 Gyro (°/s) -> X: -279.30 Y: 360.47 Z: -51.09 Accel (m/s²) -> X: 0.61 Y: 1.05 Z: -1.29 Gyro (°/s) -> X: -3.54 Y: 50.11 Z: -87.10 Accel (m/s²) -> X: -1.31 Y: 0.16 Z: 0.86 Gyro (°/s) -> X: -61.22 Y: 482.24 Z: -101.38 Accel (m/s²) -> X: 0.71 Y: 0.94 Z: -1.15 Gyro (°/s) -> X: 77.27 Y: -231.63 Z: -65.49 Accel (m/s²) -> X: -0.15 Y: 0.57 Z: -0.97 Gyro (°/s) -> X: -19.84 Y: 38.45 Z: 8.97 Accel (m/s²) -> X: -0.79 Y: 0.27 Z: -0.58 Gyro (°/s) -> X: -0.37 Y: 2.69 Z: -0.98 Accel (m/s²) -> X: -0.58 Y: 1.39 Z: 0.89 Gyro (°/s) -> X: 331.18 Y: -486.94 Z: 37.11 Accel (m/s²) -> X: -0.88 Y: -0.02 Z: -0.59 Gyro (°/s) -> X: 15.93 Y: -3.60 Z: -11.72 Accel (m/s²) -> X: -0.72 Y: 0.14 Z: -0.72 Gyro (°/s) -> X: 1.28 Y: 1.65 Z: 0.73 Accel (m/s²) -> X: -0.60 Y: 0.06 Z: -0.21 Gyro (°/s) -> X: 125.37 Y: -269.59 Z: 106.57 Accel (m/s²) -> X: -0.53 Y: 0.29 Z: -0.80 Gyro (°/s) -> X: -4.88 Y: 9.16 Z: -3.11 Accel (m/s²) -> X: -0.66 Y: 0.19 Z: -0.73 Gyro (°/s) -> X: 5.07 Y: -7.26 Z: 0.37 Accel (m/s²) -> X: -0.69 Y: 0.14 Z: -0.71 Gyro (°/s) -> X: 2.08 Y: -0.18 Z: -2.62 Accel (m/s²) -> X: 0.21 Y: 2.16 Z: 0.67 Gyro (°/s) -> X: 328.43 Y: -512.57 Z: 278.38 Accel (m/s²) -> X: 1.89 Y: 1.02 Z: -1.43 Gyro (°/s) -> X: 8.30 Y: 27.53 Z: -111.76 </pre>
4	<p>Question: Identify the max sampling rate and consider reducing the delay (line 37 in the sketch) to increase the number of samples. Summarise your findings here.</p> <p>Answer: reducing the delay will increase the frequency of data sampling which will improve responsiveness and resolution.</p>

Activity 2.4: Plot data using Python Notebook

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. You can find detail in official website (<https://matplotlib.org>).

Hardware Required

No hardware required

Software Required

Python Jupyter notebook

Steps:

Step	Action
1	Download the Jupyter notebook week2_notebook.ipynb from here (https://github.com/deakin-deep-dreamer/sit225/blob/main/week_2/week2_notebook.ipynb). Follow the instructions in the notebook to carry out instructions and finally convert the notebook to PDF so you can combine it with this activity sheet PDF.