



Co-funded by the  
Erasmus+ Programme  
of the European Union



# EUPANTEC

Physics and New Technologies

2019 - 2022



- Escola Secundária de Rocha Peixoto - Póvoa de Varzim, Portugal
- Istituto di Istruzione Superiore Pietro Scalcerle - Padova, Italy
- Lycee Pilote Innovant International – Jaunay-Marigny, France
- Viscardi-Gymnasium - Fürstenfeldbruck, Germany
- Zespół Szkół Nr 1 im. Legionów Polskich - Kozienice, Poland



## Table of Contents

I.	Introduction by Paweł Boryczka.....	4
II.	Padova meeting – Motion.....	6
A.	France.....	7
1.	Two wheel robot's movement	
B.	Germany.....	13
1.	Data transfer with Phyphox	
2.	Damped spring pendulum with Phyphox	
3.	(Simple) Pendulum	
4.	Spring pendulum	
C.	Italy.....	19
1.	Motion basic	
2.	Motion	
3.	Parabolic Motion	
4.	Pressure	
5.	Motion with constant acceleration	
D.	Poland.....	39
1.	Acceleration measurement	
2.	Optical photo gate – movement and speed measurement	
E.	Portugal.....	45
1.	Free fall	
III.	Kozienice meeting – Electricity and Environment.....	53
A.	France.....	54
1.	Drive a rover with Python - Regular polygon	
2.	Drive a rover with Python – Emergency stop	
3.	Programming an online temperature sensor	
4.	Programming a microcontroller ESP8266 for measures of temperature and light	
B.	Germany.....	59
1.	Centripetal acceleration	
2.	Distance measurement	
C.	Italy.....	62
1.	Wireless weather sensor	
D.	Poland.....	63
1.	Data logger	
2.	DS18B20-energy efficiency measurement	
3.	Energy changes on an inclined plane	

4.	Energy changes in the harmonic motion of mass on a spring	
5.	Introduction to the use of the Pico microcontroller	
6.	Resistance and voltage measurement	
E.	Portugal.....	102
1.	Joule experiment revisited	
IV.	Poitiers meeting – Physics and Environment.....	103
A.	Poland.....	104
1.	Measurement of light intensity	
2.	Measurement of humidity and temperature with DHT11	
3.	Measurement of pressure	
4.	Measurement of soil moisture	
V.	Fürstenfeldbruck meeting – Electricity.....	128
A.	Germany.....	129
1.	Comparison of the efficiency of halogen and LED lights	
2.	Magnetic field coil	
3.	Study of RC circuit	
B.	Poland.....	135
1.	Electric conductivity- Keyboard of fruits and vegetables	
2.	Skin resistance measurement- GSR	
3.	Measurement electric power consumption, current measurement	
C.	Portugal.....	155
1.	Earth magnetic field	
2.	Eddy currents	
3.	Gauss cannon	
VI.	Povoa de Varzim meeting – Waves.....	168
A.	Germany.....	169
1.	Dopplereffect with smartphones	
2.	Interference with two smartphones	
3.	Speed of sound	
B.	Italy.....	173
1.	Standing wave on string	
C.	Poland.....	181
1.	Math and Music	
2.	Principle of operation of the RFID system	
3.	Sound level meter	
4.	Simple harmonic motion with ultrasonic sensor	
D.	Portugal.....	202
1.	Estimating the thickness of a hair by light interference	

## Introduction

We present to you a manual containing instructions for the preparation of experiments in physics that use new technologies. They are the result of the work of many students and teachers during the Erasmus+ project "Physics and new Technologies" Eupantec 2019. These materials are the result of cooperation between five European countries from the east, west, south and north. They are the result of exchanging experiences and learning together. In addition to the materials made available, the project also produced other materials that were used locally in each country in preparation for foreign mobilities. We had to make a choice due to the limited time allocated for training during the mobilities, we also verified the degree of difficulty of the developed experiment. It is worth mentioning here the project of building a thermal imaging camera, smog measurement system, water quality measurement, or experiments allowing the cooperation of the microcontroller and the Phyphox application. The construction of measuring systems also encouraged us to use 3D printing to develop housing for sensors and the Pico measuring interface. We will make the designs of 3D elements available on our website [eupantec2019.eu](http://eupantec2019.eu), as well as MicroPython scripts.

We hope that our proposals will inspire teachers of physics, computer science and other subjects. Thanks to our study, lessons in physics or computer science can be more interesting. We know this because we have feedback not only from project participants, but also from students who used these materials during regular lessons. The published materials should be useful for conducting optional classes, developing skills in the field of science and computer science. We have such plans.

Personally, I have to say that before starting the project, I didn't think I would learn so much during the project. I think colleagues from other countries have similar feelings. New technologies force the updating of knowledge. While working on the project, we verified the usefulness of the tools we were supposed to use. I notice the great potential of the free Phyphox mobile application. A great discovery was the ease and effects of using MicroPython with the Raspberry Pi Pico microcontroller for € 5.-, which in many situations allows to successfully replace the laboratory equipment offered by world tycoons. Pico offers opportunities for cheap and effective



equipment for physical laboratories. This is not without significance for many European schools. Additionally, we can achieve better results, because the students have access to the code, understand the process of performing the measurement, they can adapt it to their own needs and create their own "firmware" of the measuring systems being built. Thanks to this, at the same time, along with physical knowledge, students improve their programming techniques.

Despite the very difficult conditions for the implementation of the project caused by the COVID-19 pandemic, most of the objectives have been achieved. For the participants, apart from building knowledge during the workshops, it was important to get to know colleagues from other countries and work together. The opportunity to get to know other countries and their culture was certainly of great importance. The practical use of foreign language skills was also very important. I am convinced that the positive experiences gained by students during the project may have an impact on the choice of fields of study related to physics and new technologies. We can already observe it.

We wish you pleasant reading and satisfaction with the results of the experiments described in this publication.

Paweł Boryczka  
project coordinator



Padova

Istituto di Istruzione Superiore Pietro Scalcerle

# MOTION

10.12. – 13.12.2019



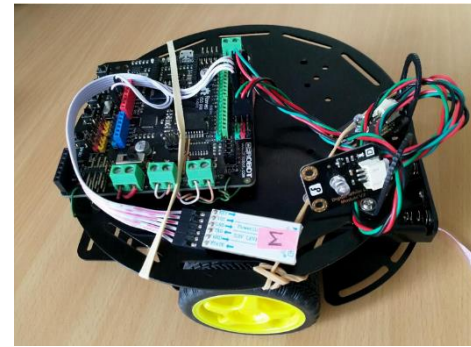


Topic	Age	Country	Date
2-wheel robot's movement	>14	 France	Padova Dec 2019

Objective:

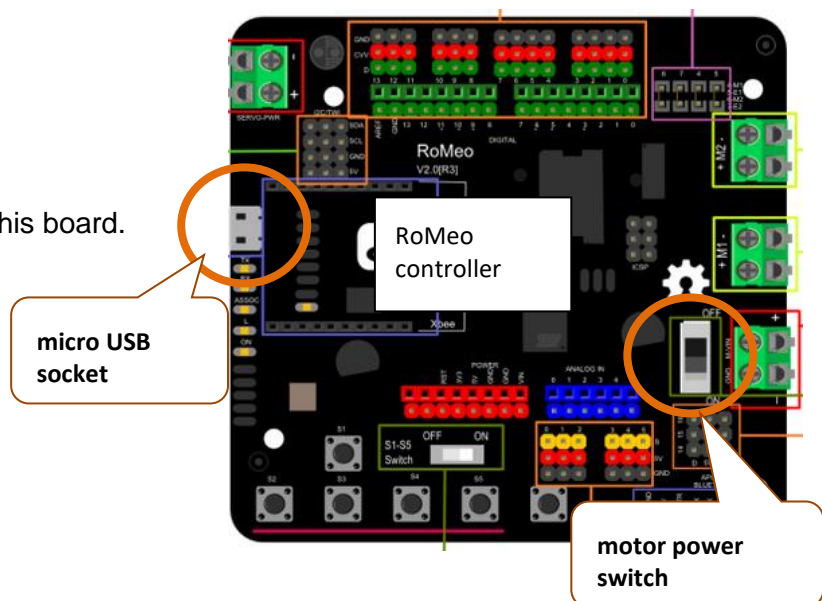
During this activity students will modify a C code to move a 2-wheel robot.

This robot can simulate the behavior of an autonomous vacuum cleaner like the one shown in the picture beneath.



The robot's brain is the RoMeo controller.

You will use the Arduino IDE to program this board.





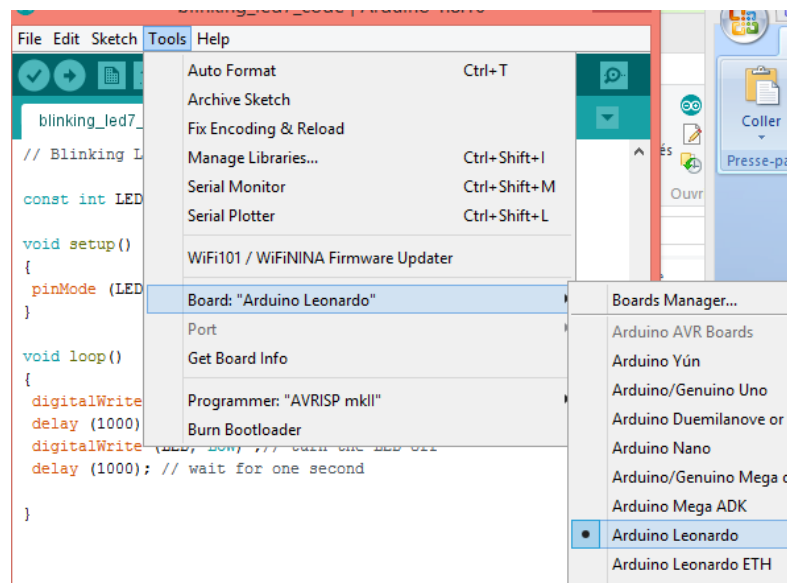
Please **Turn OFF the Motor Power Switch** when Romeo is connected to the computer through USB cable. Or the external power supply(>12V) will destroy your Romeo.

## STEP 1 Blinking Led code - first step with Arduino IDE

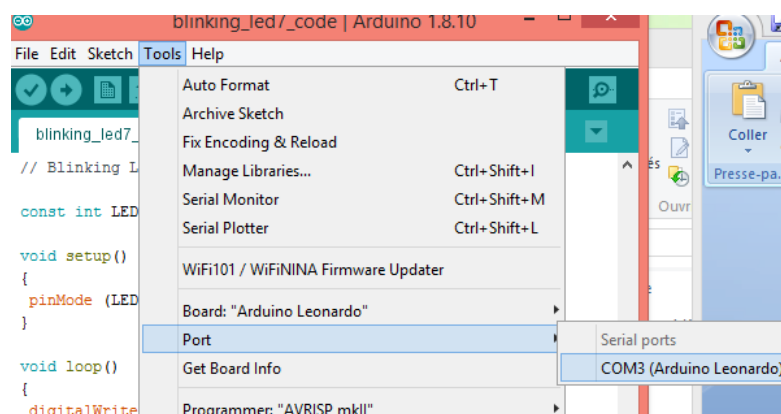
Using the USB wire, **connect** your RoMéo board to the computer. The micro USB socket is on the RoMéo's left side (see the picture on the first page).

Open the code "blinking\_led3\_code":

Click **Tools**, then Board and select **Arduino Leonardo** in the list.



To choose the serial port, click **Tools>Port** and **select the COM** where Arduino Leonardo is connected to.





Click on the **arrow Upload** to upload the sketch (code) to Romeo:



If your sketch has uploaded successfully, **you will now see this LED happily flashing on and off slowly on your board.**

If so, **congratulations**, you have just successfully uploaded and ran your first sketch.

Have a look at the "blinking\_led3\_code" and find the command that sets the blink's time.  
Then, **change the time** to **100** milliseconds and upload your new sketch.

**Find**, in the code, the statement that switches the LED on.

You will use a similar statement to rotate motors.

```
blinking_led3_code
// Blinking Led code

const int LED=3; // LED connected to digital PIN 3

void setup()
{
  pinMode (LED, OUTPUT) ; // LED is an OUTPUT
}

void loop()
{
  digitalWrite (LED, HIGH) ;// turn the LED on
  delay (1000); // wait for one second
  digitalWrite (LED, LOW) ;// turn the LED off
  delay (1000); // wait for one second
}
```



## **STEP 2 Testing robot's movement - first step in coding robots**

Insert the batteries in your robot.

Connect your robot to the PC. **Return** your robot (so it will not move on the table)..

Open the code "**Romeo\_motor\_forward**". Click **Tools>Port** and **select the right COM**.  
**Upload** the code.

**Remove** the USB wire, put the robot on the floor and slide the power switch to **ON**.  
After some seconds your robot will move forward and stop.

## **STEP 3 Moving the robot about 50 cm forward.**

**Slide the power switch to OFF.**

Question *How much duration (delay statement) do you need to go forward about 50cm?*

Hint:

If the wheel makes two revolutions, how far the robot will go?

Explain your answer.

Now, you have to modify the previous code (the part shown in the picture) so the robot moves straight and its wheels make about two rotations.

```
// move forward during 1 second:  
analogWrite (E1,120); //speed 120  
digitalWrite (M1,LOW);  
analogWrite (E2,120); //speed 120  
digitalWrite (M2,LOW);  
delay(1000); //wait for 1000 ms, so
```

**Change** the parameters in **analogWrite (E2, 120)** and in **delay (1000)** to have a straight movement and two revolutions of the wheels. The marks on the wheels can help you.

**Save** your new code as "**Romeo\_motor\_forward\_1**".



Upload your code, remove the USB wire, put the power switch to ON and TEST the robot's movement.

## **STEP 4 Moving the robot about 25 cm backward.**

Slide the power switch to OFF.

To reverse the direction of rotation, you have to change the polarity of the supply voltage.

For example, if you want the M1 motor to move backward, you have to change the statement : **digitalWrite** (M1, **LOW**) to **digitalWrite** (M1, **HIGH**).

Add some statements to the previous code so your robot drives forward 50cm, then stops for 3 seconds, moves backward about **25cm** and stops.

Save your code as "Romeo\_forward\_backward".

## **STEP 5 Turn the robot 180°**

Slide the power switch to OFF.

There are several methods to turn the robot. In this activity you have to program the one shown in the picture.

Question:

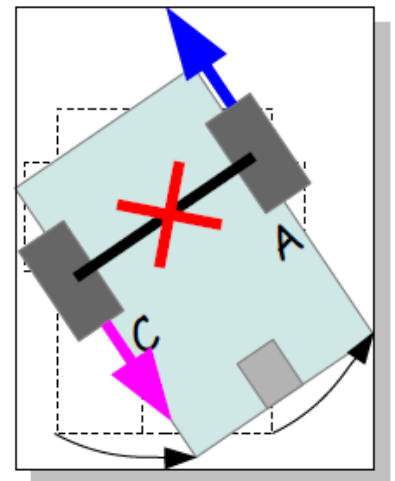
*How much duration does the wheel need for the robot to turn 180°?*

Explain your answer.

**Add** to the previous code some statements to turn your robot.

Keep experimenting until it is almost perfect!

**Save** your code as "Romeo\_move\_turn".



**Extra tasks:**

### **Task 1 Movement and blinking Leds**

Connect a green Led to the Pin 3 and a red Led to the Pin 2.



**Modify** your code "Romeo\_move\_turn" to have the green Led switched on when the robot moves forward, the red led switched on when the robot moves backward and the both Leds switched on when the robot turns.

## Task 2 Moving around a square

Make your robot drive around a square.

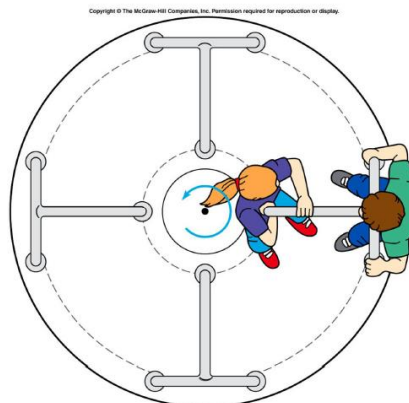
## Task 3 Calculating speeds

Calculate the robot's velocity:  $v$  (m/s)

Calculate the robots rotational speed:  $N$  (rotation/minute) and  $\omega$  (radian/second).

Check if the calculated velocities satisfy the relation shown in the picture:

- Relationship between linear and rotational velocity



- On a merry-go-round, a rider near the edge travels a greater distance in 1 revolution than one near the center.
- The outside rider is therefore traveling with a greater linear speed.

$$v = r\omega$$

## Task 4 Edge detection

Connect an edge detection sensor to your robot and write a code that will make the robot move on a table (not black surface). The sensor value is **LOW** or **0** when an object is detected (<10cm) and **HIGH** or **1** when an edge is detected.

See the "edge sensor hint code", if you need some hints.

*A black surface must be 4-5 cm from the sensor (if less, the sensor reads 1).*

[https://wiki.dfrobot.com/DFRobot\\_Infrared\\_sensor\\_breakout\\_SKU\\_SEN0042](https://wiki.dfrobot.com/DFRobot_Infrared_sensor_breakout_SKU_SEN0042)

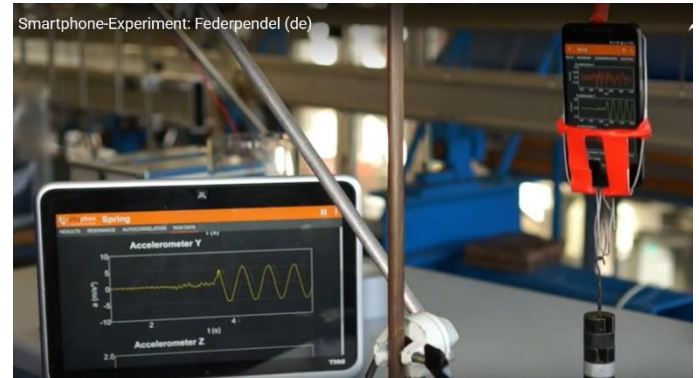


Topic	Age	Country	Date
Data transfer (phyphox)	>14	Germany	Oct. 2019

## Remote control and data transfer

The picture on the right shows the swinging mobile phone with the tablet showing the elongation over time. That's a great advantage since you can read the data immediately during the experiment without looking at the smartphone.

Instead of a tablet one can also use a laptop or a second smartphone.



## There are two possibilities to connect your smartphone with your tablet

### 1. WIFI

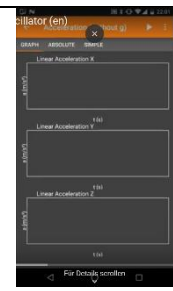
Both devices are logged onto the same WIFI!



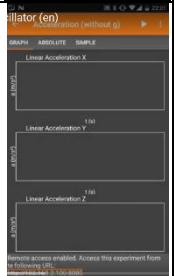

(Unfortunately schools and hotels use different suberversers, sometimes it is the same.)

1.) Open the Phyphox App und click to the application “Spring”.



2.) Open the menu bar by clicking on the three dots on the top right on the screen.



<p>3.) Activate “Allow remote access”.</p>		<p>4.) Click OK.</p>	
<p>5.) A URL - address will be shown at the bottom of the screen.</p>		<p>6.) Enter this address to the web-Browser on your tablet. If there are more address numbers, try them all.</p>	

## 2) Mobile Hotspot

Activate a hotspot on your smartphone.

<p>1. Deactivate your WIFI on your smartphone. Open a Mobile Hotspot on your smartphone.</p>	<p>2. If you have an android smartphone deactivate <i>mobile data</i> (this option is not available on iPhones).</p>
<p>3. Connect your tablet with your smartphone: To do so you have to click the WIFI symbol on your tablet and find the name of your smartphone.</p>	<p>4. Now both devices use the same network.</p>
<p>5. Open the Phyphox App und click to the application “Spring”.</p>	<p>6. Open the menu bar by clicking on the three dots on the top right on the screen.</p>
<p>7. Activate “Allow remote access”</p>	<p>8. Click <b>OK</b>.</p>



9. The URL - address  
will be shown at the  
bottom of the screen.



10. Enter this  
address to the  
web-Browser on  
your tablet. If  
there are more  
address numbers, try them all.



Topic	Age	Country	Date
(Simple) Pendulum with Phyphox	>14	Germany	Oct 2019

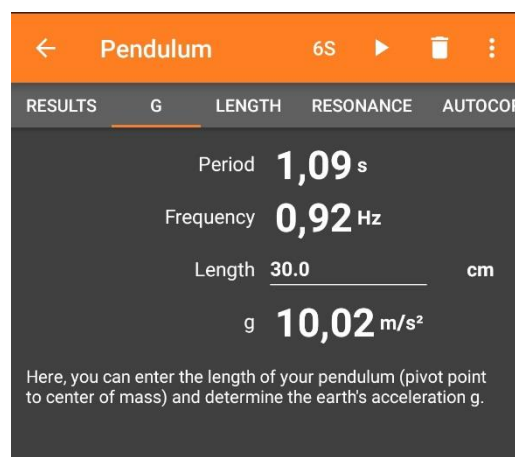
- toilet roll
- some string
- a construction to fix the pendulum (-> use information in the app)

1) You can e.g. determine the gravity constant by using your smartphone as pendulum. You have to enter the length  $L$  of the pendulum. With the including gyroscope it's possible to measure the period  $T$ . ( $g = \frac{4\pi^2 \cdot L}{T^2}$ )

2) You can plot the amplitude against the detected frequency.

3) You can discover the formula  $f = \frac{1}{2\pi} \cdot \sqrt{\frac{g}{l}}$

Important: You can use the possibility of a timed run (enter *start delay* and *duration*) or use a second smartphone/ tablet/ laptop/ PC for starting and stopping the measurement.





Topic	Age	Country	Date
Spring pendulum with Phyphox	>14	Germany	Oct. 2019

### Experimental setup materials:

1 smartphone and a smartphone case  
1 tablet/laptop  
1 pencil/ mounting device  
scotch tape  
2 springs  
weights

### Experimental setup and procedure:

Start **Phyphox** App on your smartphone and setup the experiment as shown in the picture. You can either use the mounting device or a pencil with some tape. Be aware, the smartphone position in the case must be **vertical**.

You can use the possibility of a timed run (enter *start delay* and *duration*) or use another smartphone/ tablet/ laptop/ PC for starting and stopping the measurement.

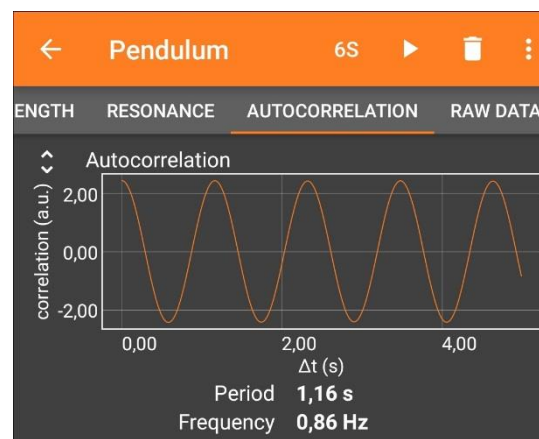
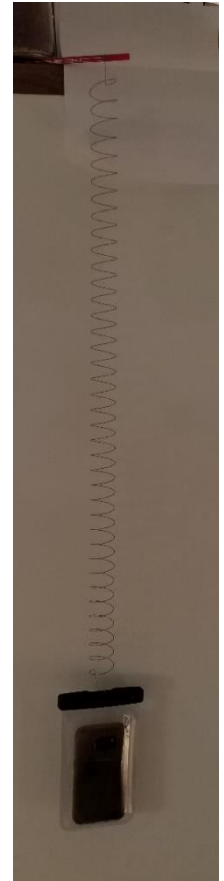
### Evaluation of the experiment:

#### By quality:

- 1) Describe the t-y-diagram. Discuss/explain the correlation between oscillation and the period T by using the t-y-diagram. Sometimes it is useful to click the Pause button.
- 2) Modify the elongation of the oscillation (not varying weight and spring) and observe the variation of period T.
- 3) Modify the mass of your smartphone, you could tape weights to the case. Observe the variation of period T.
- 4) Modify the spring constant D (be aware, use the same mass and elongation as in the earlier experiments) and observe the period T.

#### By quantity: (influence on T)

- 5) Modify the mass and enter the measured data of the period in an m-T-diagram.
- 6) Modify the spring constant and enter the measured data of the period in a D-T-diagram.
- 7) Do you find a correlation between T, D and m?





Topic	Age	Country	Date
Damped spring pendulum with Phyphox	>14	Germany	Oct 2019

## Damped spring pendulum

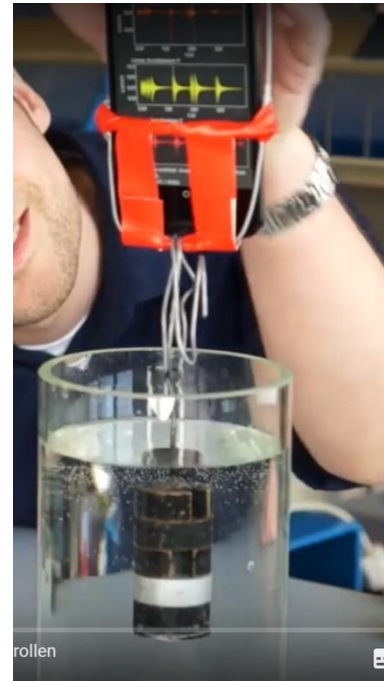
### Experimental setup materials:

- 1 smartphone
- 1 tablet/laptop
- 1 pencil/ mounting device
- Scotch tape
- 1 spring
- weights
- 1 glass of water

### Experimental setup and procedure:

Start the **Phyphox App** on your smartphone and setup the experiment as shown in the picture. Fix the weights to the smartphone case with Scotch tape. Be aware, the smartphone position in the case must be **vertical**.

Elongate the smartphone and start the experiment on the **tablet/laptop screen**. In order to do so you must click the **play button** (on the top right hand side of the screen). Select the button **acceleration**. For this experiment limit the time of the measurement to 5 seconds.



### Experiment evaluation:

- 8) Describe the t-y-diagram.
- 9) Explain the differences between the t-y-diagram of the harmonic oscillator and the damped oscillator!
- 10) Guess what will happen if you exchange the water for jelly? Draw a rough t-y-diagram for this case.

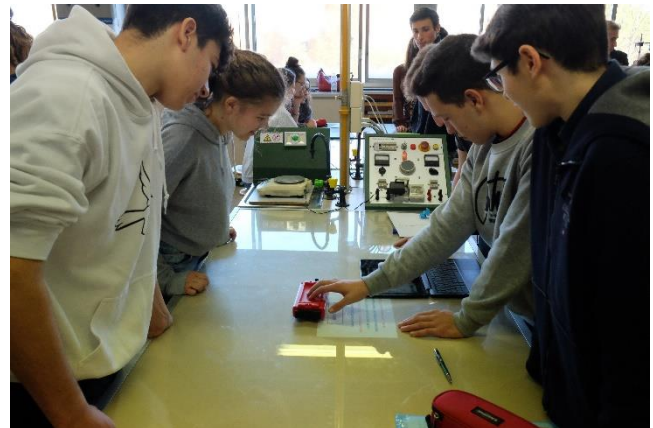




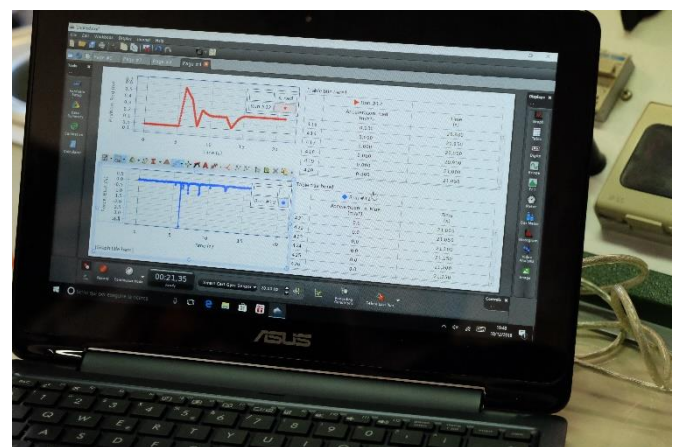
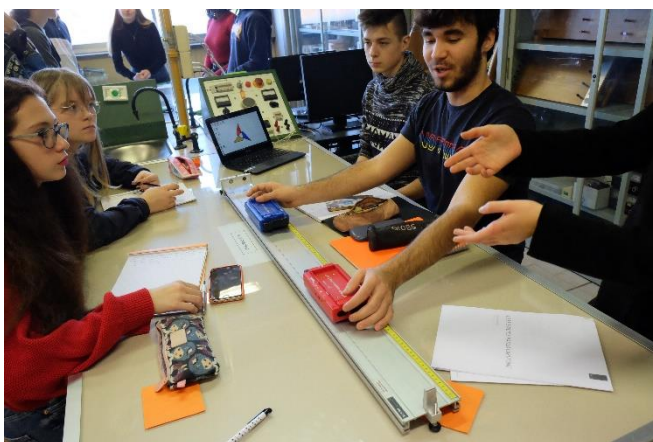
Topic	Age	Country	Date
Motion (Capstone)	>14	Italy	Dec 2019

A smart cart is moved about the desk. It sends data directly to the computer.

Position, speed and acceleration are visualized by the program.



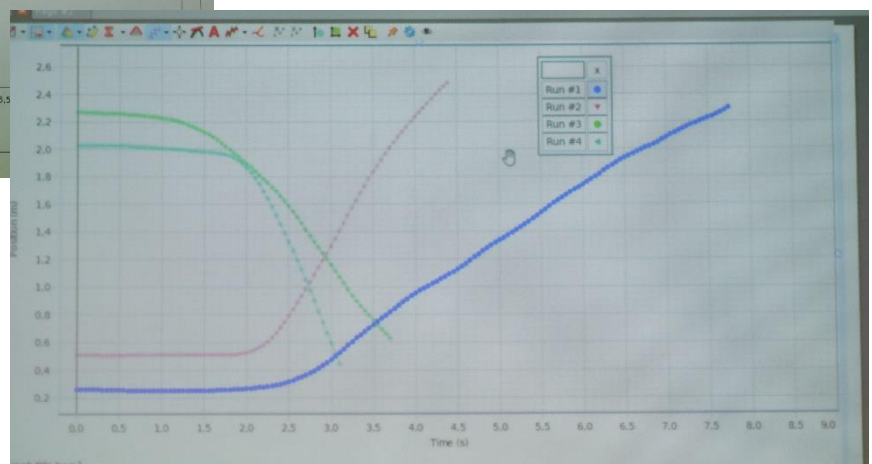
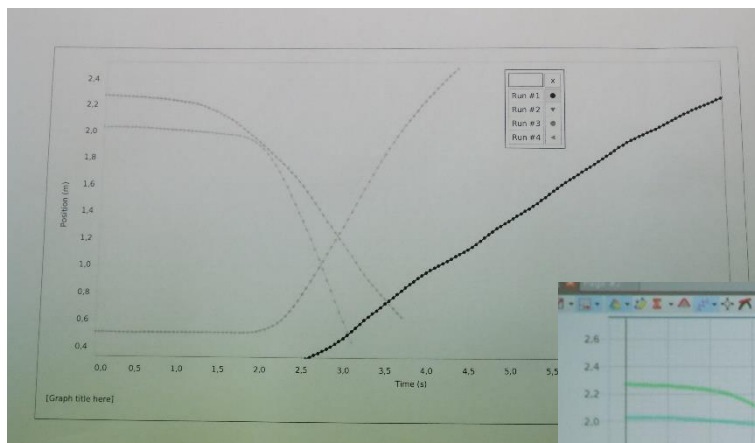
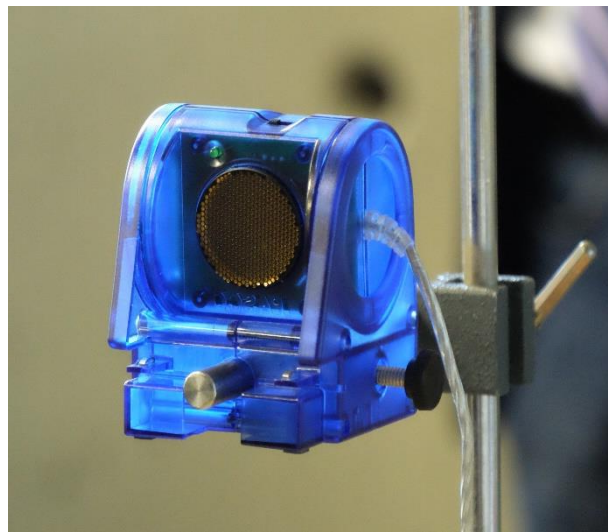
The movements of two different carts can be compared.





Topic	Age	Country	Date
Motion (Pasco)	>14	Italy	Dec 2019

Position-time graphs are created by linear motion of students.  
Different graphs can be compared and shaped to match one another.  
The students are asked to create a given graph.



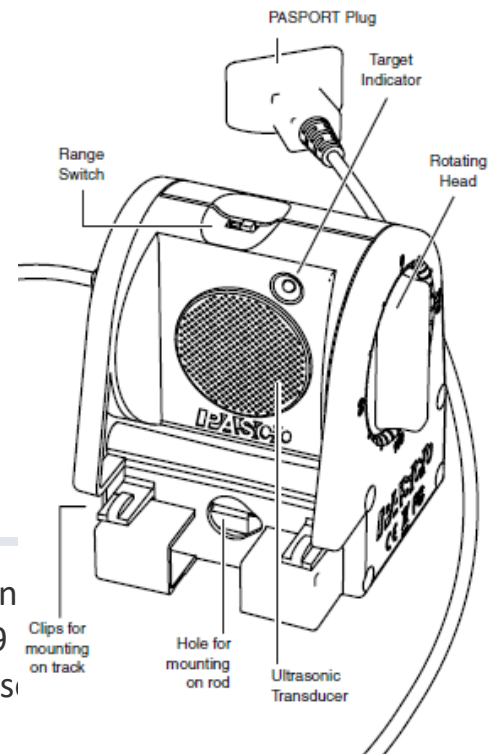


## PASPORT Motion Sensor - PS-2103A



### How It Works

An electrostatic transducer in the face of the Motion 16 **ultrasonic pulses** with a frequency of about 49 reflect off a target and return to the face of the sensor. The sensor flashes when the transducer detects an echo.



The sensor measures the time between the trigger rising edge and the echo rising edge. It uses this time and the speed of sound to calculate the distance to the object.



### Specifications

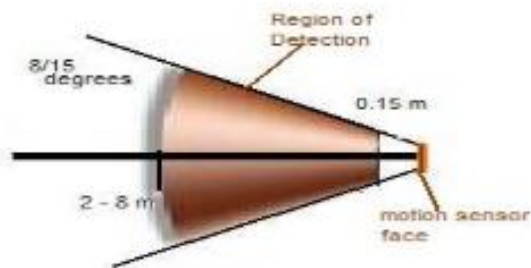
<b>Range</b>	<ul style="list-style-type: none"> <li>0.15 to 8 m</li> </ul>
<b>Resolution</b>	<ul style="list-style-type: none"> <li>1.0 mm</li> </ul>
<b>Maximum Sample Rate</b>	<ul style="list-style-type: none"> <li>50 Hz</li> </ul>
<b>Transducer Rotation Range</b>	<ul style="list-style-type: none"> <li>360°</li> </ul>



The normal range of sampling rates is between 1 Hz and 50 Hz. At the default rate, the Motion Sensor can measure distance up to 8 m. The maximum distance decreases with increasing sample rate. At very high sample rates (between 50 Hz and 250 Hz), the maximum distance is less than 2 m.

### To Aim the Motion Sensor at an Object

1. Set the range switch to the short range ( ) or long range ( ) setting.

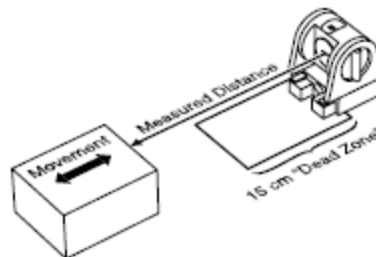
- Select  for measuring a cart on a track.
- Select  for measuring most other objects.



 15°  8° small cone angular opening

2. Arrange the Motion Sensor and object so that the Motion Sensor's transducer faces the object.

- The object should be at least 15 cm from the transducer.
- If the object will move, it should move directly toward or away from the Motion Sensor.
- Aim the motion sensor slightly up to avoid detecting the tabletop.



3. Remove objects that may interfere with the measurement. These include objects between the sensor and target object, either directly in front of the sensor or to the sides.



## Experiment: Understanding Motion - Distance and Time (Motion Sensor)

Concept: linear motion

### EQUIPMENT NEEDED

- PASCO Capstone Software
- PASCO USB Link Interface
- PASPORT Motion Sensor
- Base and support rod
- Reflector board (optional)

### PURPOSE

The purpose of this activity is to introduce the relationships between the motion of an object - YOU - and a Graph of position and time for the moving object.

NOTE: This activity is easier to do if you have a partner to run the computer while you move.

### THEORY

When describing the motion of an object, knowing where it is relative to a reference point, how fast and in what direction it is moving, and how it is accelerating (changing its rate of motion) is essential. A sonar ranging device such as the Motion Sensor uses pulses of ultrasound that reflect from an object to determine the position of the object. As the object moves, the change in its position is measured many times each second. The change in position from moment to moment is expressed as a velocity (meters per second). The change in velocity from moment to moment is expressed as an acceleration (meters per second per second). The position of an object at a particular time can be plotted on a graph. You can also graph the velocity and acceleration of the object versus time. A graph is a mathematical picture of the motion of an object. For this reason, it is important to understand how to interpret a graph of position, velocity, or acceleration versus time. In this activity you will plot a graph in real-time, that is, as the motion is happening.

### PROCEDURE

For this activity, you will be the object in motion. The Motion Sensor will measure your position as you move in a straight line at different speeds. The Capstone program will plot your motion on a graph of position and time.

### Computer Setup

1. Plug the *USB Link* into the computer's USB port.
2. Plug the *PASPORT Motion Sensor* into the USB Link. This will automatically launch the Capstone window.

The motion detector is presented here as a "black box". There is no need to explain its working to students. It is enough that students are convinced that the motion detector does what it is expected to do — that is, that it specifies the position of an object at different times. This is goal of the first approach.

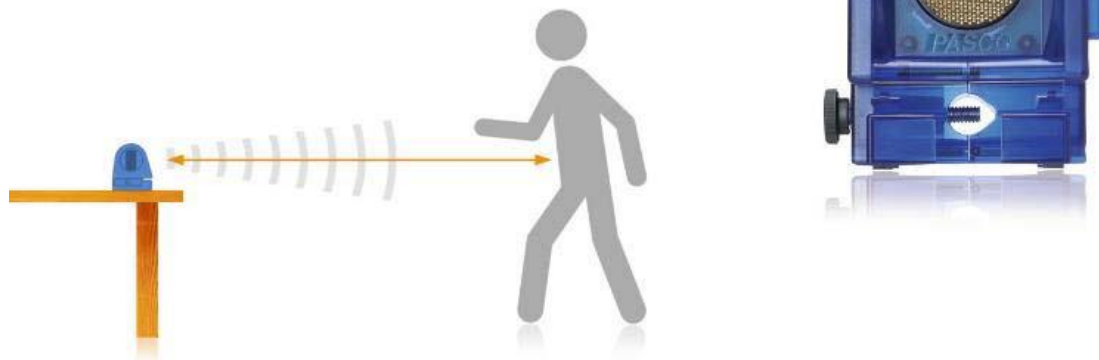
Warning: You will be moving backward, so be certain that the area behind you is free of obstacle because the distance is calculated from the first object that reflects the ultrasounds.

## Equipment Setup

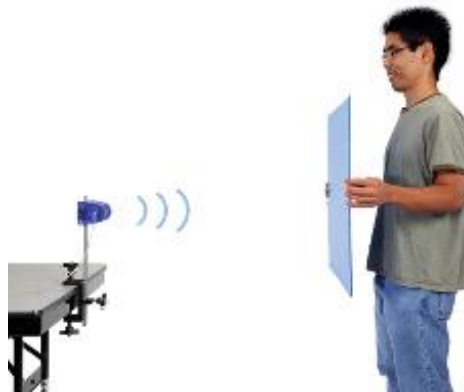
- Mount the Motion Sensor on a support rod so that it is aimed at your midsection when you are standing in front of the sensor. Make sure that you can move at least 2 meters away from the Motion Sensor.
- Position the computer monitor so you can see the screen while you move away from the Motion Sensor

### Motion Sensor

Make sure to clear the area of obstacles where you will be walking.



So the sensor will detect the closest object. If you swing your arms or take large steps, the detector will at one moment see an arm, at another a leg. This motion causes an erratic position-time graph. The best position-time graphs are obtained by clapping a reflector board in front of you and moving with short, shuffling steps of about 5 cm each.



### What are we going to measure?

All the points that are on a circumference that has the sensor as its center are at the same distance from the sensor itself, although it is possible to collect measurements only of those points that are located within the cone "visible" to the sensor. Furthermore, it should be noted that what is detected is the distance and not the position, ie the sensor does not say exactly where the object is but only how far it is. To verify this statement, let's do a simple test.

We place your partner in front of the sensor about 2 meters away and make a position-time graph; obviously, the result is a line parallel to the time axis, since, if your partner has been reasonably still, its distance from the sensor has not changed.

Small variations are attributable to the fact that your partner, despite his best effort, cannot be perfectly still because he is an animated being and therefore he breathes.

Don't forget to stop recording, otherwise your computer's memory will be filled unnecessarily.



Now let's try to make a similar graph, but making our lab rat move along the line drawn on the ground on which before it had been stopped, that is, to make it clear, the line that highlights the 2 meters. Obviously, the movement has to take place inside the detection cone of the sensor.

We can see that this second graph is extremely similar to the first one despite, in this case, your partner has changed his position several times; this is because he did not change his distance from the sensor since he walked along the line whose points were equidistant from the sensor itself. We can therefore conclude that what is measured is the distance of the object and not its position. But the motions that will be observed and studied will almost all be in one direction only and therefore, in this hypothesis, the two concepts of **distance and position are the same**, from this point on we will simply speak of position.

#### **Investigation A**

Observe, while recording, the motion of your partner while, after standing still for the first two seconds, he moves away from the sensor slowly and with the most regular gait as possible. Don't forget to stop recording. As your partner moves, the computer creates a graph that represents how its position from the sensor changes as time passes.

#### **Investigation B**

Now look at what happens if a person repeats experience A increasing the gait. Let's make the computer superimpose the two graphs; what do you notice?

#### **Prediction**

What's your prediction about the position-time graph when your partner moves towards the sensor with a regular and slow gait? Draw with a pencil in the apposite space below the graph you predict:

GRAPH C

#### **Investigation C**

Let's make now a verification of your prediction by carrying out the motion. As in the previous cases, a person will stand still for a couple of seconds and then leave. Bring the newly computer-generated graph to the same space where you just sketched your prediction (graph C). Was your prediction correct?

#### **Prediction**

How do you think the position-time graph will be, repeating experience C but with a more sustained gait? Draw your prediction below.

GRAPH D

#### **Investigation D**

We create the graph using the computer. Place it together with your prediction and don't forget to note the characteristics of the motion. Let's ask now the computer to superimpose the two graphs C and D. What do you notice?

#### **THEORY**

Remember that in the previous graphs the **position** and the **instant** are to be understood in the definition we give to them in physics: the *position* of a body is a point in space, ie we *model* the body as a point, the *instant* is a point on the axis of time and it is NOT an interval, as in everyday life! The position and the corresponding



reading of the instant on a clock (eg the one in the computer) are closely connected and we call this combination "**event**".

The change of position from a point  $s_1$  to another point  $s_2$  is called **displacement**  $\Delta s$ , where  $\Delta s = s_2 - s_1$  (the symbol  $\Delta$  represents the change in value of a quantity: final value minus initial value).

When numbers are inserted as position values, the movement in the positive direction (away from the sensor) is always a positive sign, and a movement in the opposite direction (approaching the sensor) will always be represented by a negative sign number.

The displacement is an example of a **vectorial quantity**, that is a quantity which is characterized by a *direction*(angle), with its *sense*(orientation), and by a *magnitude*. In the displacement, its *magnitude* (ie the number of meters) is the *distance* between the *initial* and *final position*, the sense of its direction, from the initial position to the final position, *is represented by a positive or negative sign*.

In previous experiences you have been invited to move with a slow and sustained gait. Analyzing the respective position-time graphs, it has been observed that different speeds are linked to different slopes of the straight sections in the position-time graph. The greater the speed, the greater the slope of the path in the graph.

Another clear information that can be deduced from the graphs is the negative slope obtained in approaching motions. Remembering what we have studied with the straight line in the Cartesian plane, let's think about what calculation we can perform with the numbers **s** and **t** to obtain direct information on how quickly the person (a body in general) is moving.

By **average velocity in a given time interval we mean the displacement divided by the time interval in which the movement took place**:

$$v_m = \frac{\Delta s}{\Delta t} = \frac{s_2 - s_1}{t_2 - t_1}$$

Apply this formula with the data in your possession from the previous activities recorded in the file and calculate the average velocity in some time intervals of the different motions. Think about how these values vary.

You can see that the value is great when the body moves quickly; the value is small when the body moves slowly; the algebraic sign indicates the direction of motion, and so on.

Warning: often in our everyday language we mean by velocity a quantity that is not the one previously defined (*velocity vector*), but rather the *average scalar speed* (the gait).

While the average vector velocity concerns the displacement of a body, this other type of average velocity considers the total length actually traveled (for example the number of meters traveled), regardless of direction.

That is  $V_s = (\text{total path length}) / \Delta t$ .

The average scalar velocity differs from the average vector velocity also because it does NOT include the sense(orientation) and therefore lacks an algebraic sign.

These differences lead to different results depending on the concept of speed considered. Eg in the forward and backward motion in front of the sensor, when you return to the same starting position, you have a zero vector velocity because  $\Delta s = 0$  !!

In Anglo-Saxon countries the different use of the concept of speed is solved with two different words: "*speed*" for the scalar speed (the gait of our motions) and "*velocity*" for vector velocity.

## Match the Graphs

A competition among groups

Let's form now small groups with your companions. Analyze, together with your group mates, the graph below:



### Prediction

How should your partner move to get a graph that overlaps the one above? Do a group prediction.

In the first four seconds the person must \_\_\_\_\_ between the fourth and sixth second must ...

Let's verify empirically the group predictions. Each group will now designate a representative who will:

1. Move according to the indications developed by its group.
2. Try to obtain, in any possible way, the previous graph, that will now appear on the computer screen.

Were certain parts of the plots easier to match than other parts? Why or why not?

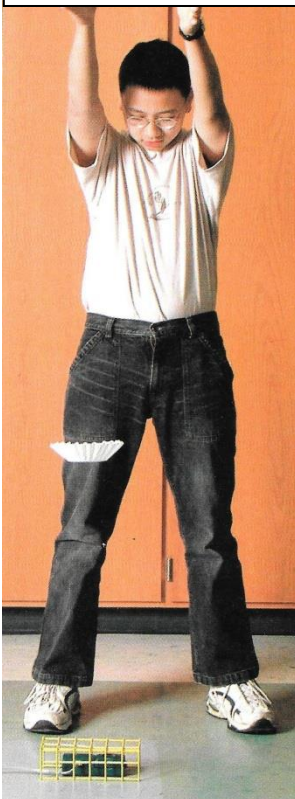


**EXPERIMENT - TERMINAL SPEED** (Uri Haber-Schaim et al, "Force, Motion, and Energy", Science Curriculum Inc. USA, 2002)

When you ride a bicycle starting from rest, you very quickly reach a speed that you want to keep constant for most of the ride. For cars, trains, airplanes, and so on, this speed is called the *cruising speed*. Does a "cruising speed" occur only through human control, or does it also happen in nature?

To find out, you will investigate how a coffee filter falls to the floor after being released. In deciding from which position to release the filter, take two considerations into account. You want the filter to fall over the longest possible distance, but you also want the detector to "see" only filter, not you as well.

A coffee filter shortly after being released. Note the position of the student's arms and the position of the motion detector.



- How should you hold the filter to achieve a reasonable compromise between these two considerations?
- After releasing the filter, why should you keep your arms up until the filter reaches the floor?

With your partner, try a run or two to see how the graph look. The motion detector only registers the location of objects at a distance greater than 15 cm.

- How does the height-versus-time graph show that the motion detector loses sight of the filter when it is closer than 15 cm?

Change the time axis so that only the part where the motion of the filter has been recorded is visible.

- In what time interval did the coffee filter fall at constant speed? How did you reach your conclusion?

The constant speed reached by a falling body is called the terminal speed.

- What was the terminal speed of the coffee filter?

### THEORY

There are only two forces acting on the filter once you release it — the gravitational force and the friction with the air. When you released the filter, the force of friction exerted by the air must have been less than the gravitational force. If the force of friction had been as strong as the gravitational force, the filter would have stayed suspended in midair. It did not. The filter fell and reached a terminal speed in a very short time. This tells us that the frictional

force exerted by the air increased as the speed of the filter increased until it balanced the gravitational force. Generally, the force of friction increases with speed as a solid object moves through a gas or a liquid.

Figure is a graph of height versus time for a falling coffee filter. The motion detector registered the height every tenth of a second. From the graph, find the time at which the filter reached terminal speed.



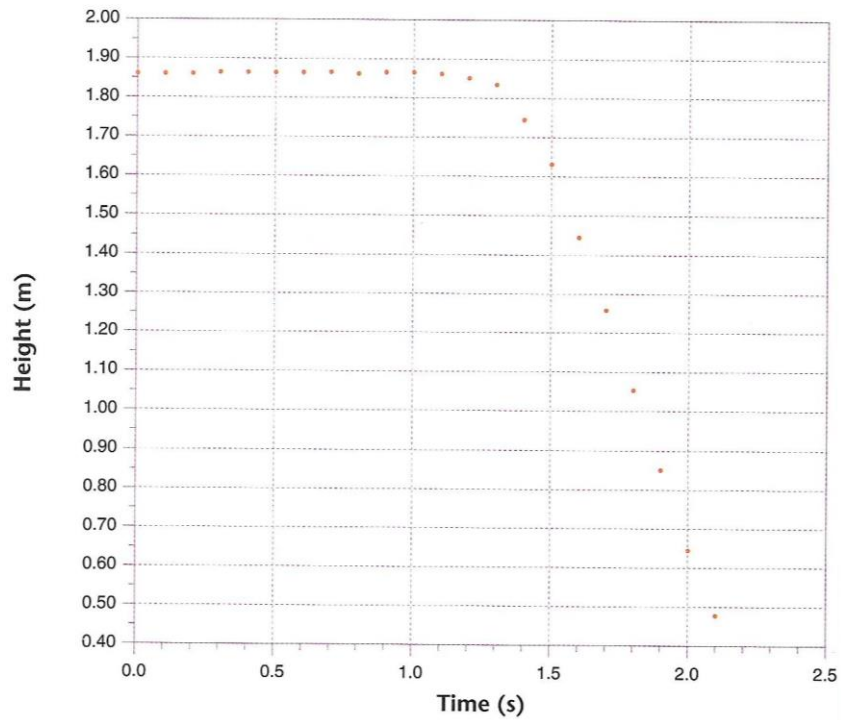


Figure is a graph of height versus time for a falling coffee filter. The motion detector registered the height every tenth of a second. From the graph, find the time at which the filter reached terminal speed.



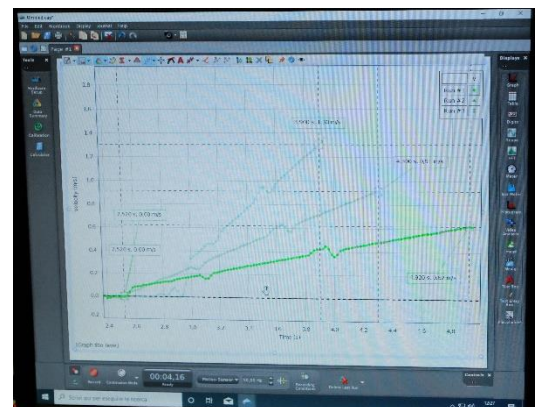
Topic	Age	Country	Date
Parabolic Motion (Pasport)	>14	Italy	Dec 2019

## Parabolic motion

TEAM n° \_\_\_\_ Students: \_\_\_\_\_

**Aim:** to determine the range and the maximum height that a body fired by a cannon reaches along its parabolic trajectory.

**Equipment:** Ballistic cart, smartphone, Pasco Capstone software.



### Instructions:

- 1) Prepare equipment as in the picture:
- 2) Launch the ballistic cart and shoot a video of what happens, if possible, in high definition
- 3) Send the video to the PC to analyze the motion



### Theoretical outline

Parabolic motion is a motion in 2 dimensions, the results from the composition of two simultaneous and independent one-dimensional motions, along perpendicular directions:

- uniform rectilinear motion
- uniform accelerated rectilinear motion

It can be described by means of the kinematic relationships between vector quantities position, velocity and acceleration.

### Enter Analysis Mode

To analyze a movie clip, you will need to define the horizontal and vertical axis. The axis is movable by click-dragging the yellow circle at the origin. Grabbing the yellow circles on the axis can also rotate the coordinate system. You will also need something in the movie frame of known length. The calibration tool can be moved and scaled to define the movie scale. The setting of these two tools will give us the proper values in our analysis.

The default movie controls for video are for play and data sync mode.

Video analysis mode is already enabled. You access it by clicking on the button. (Note that the toolbar changes to a new set of tools).



1. Using the playback controls (Experiment Control bar at the bottom of the page) advance the movie to the frame where the ball is coming out of the ballistic cart launcher.

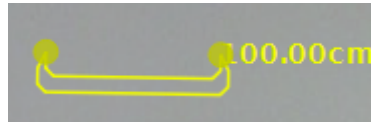


2. Click in the movie frame to make it active.
3. Move the origin of the XY axis tool to the middle of the PASCO cart (where the white marker is – it is probably partially hidden by the bracket that launches the projectile out of the cart). Once positioned, you can rotate the axis a little to align with the track.

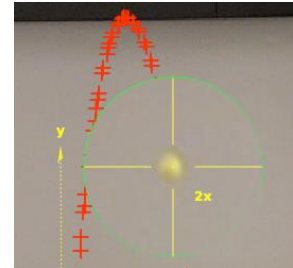




4. Click on the Calibration Tool and move its ends to end-stops on the track. The distance between the outside of both end stops is 120 cm (this distance has been pre-set for you in properties and under "Calibration Tool" - "Real World Length").



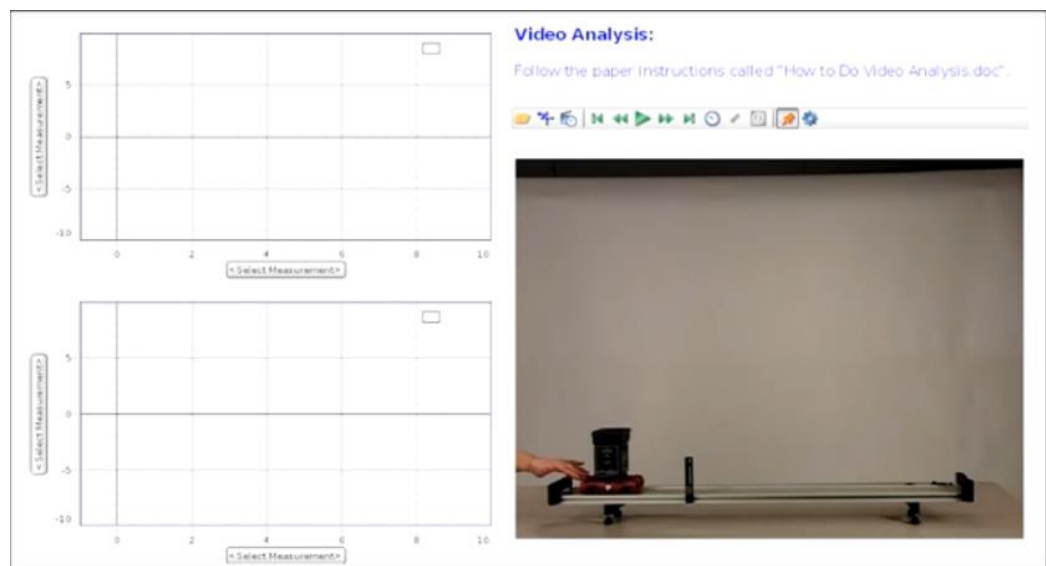
5. You can start clicking the ball to track it – but using the magnifier tool will yield more accurate data. Activate the magnifier tool (in the movie toolbar). Now you can see the projectile positions better and target a point on the ball consistently with each frame.



### Graphs:

Obtain the following graphs:

- a.  $v_x$  (x-component of velocity) vs. time graph. Find the average horizontal velocity of the cart
- b.  $v_y$  (y-component of velocity) vs. time graph.
- c. Superimpose the two graphs and print the result.



### Answer the following questions:

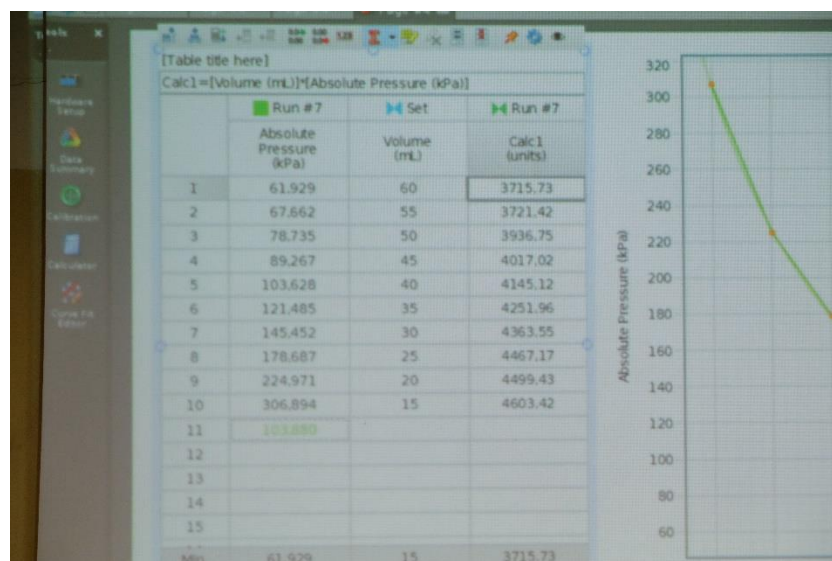
- 1) How does the x-component (horizontal) of the velocity of the ball vary?
- 2) How does the y-component (vertical) of the velocity of the ball vary?
- 3) Get the software to display the graph of y-component of the acceleration vs. time and calculate its average value. Is this value close to  $a$ ?
- 4) Can you infer more information from other graphs?



Topic Pressure (Capstone)	Age >14	Country Italy	Date Dec 2019
---------------------------------	------------	------------------	------------------

Rising pressure is created in an empty syringe and registered by a pressure sensor.

The rising rates of pressure are visualized by the program.





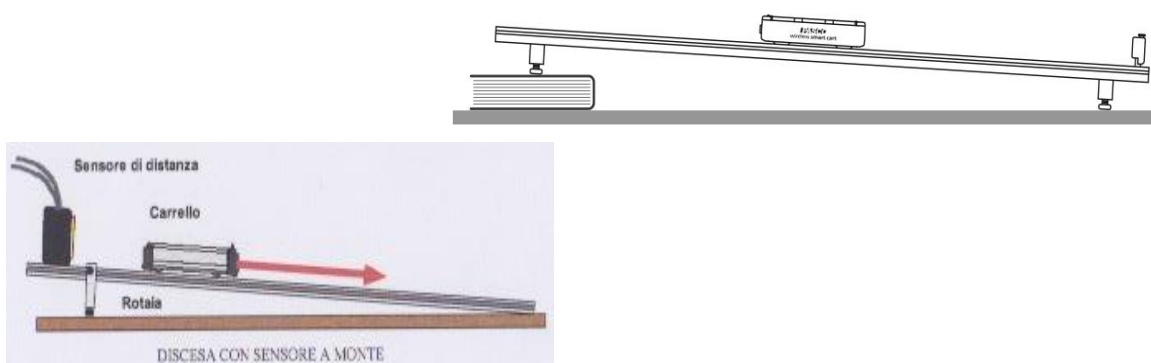
Topic	Age	Country	Date
Motion with constant acceleration (Pasco)	>14	Italy	Dec 2019

## EXPERIMENT: Motion with constant acceleration

Team n° \_\_\_\_ Students: \_\_\_\_\_

**Aims:** to define acceleration and discuss its sign. Relate speed vs. time graphs with acceleration vs. time graphs. Identify accelerated and decelerated motion in a speed vs. time graph.

**Equipment configuration:** on the left configuration with the sonar sensor; on the right with the Bluetooth cart.



### Instructions for sonar sensor, data acquisition software and cart operation

1) Launch the CAPSTONE software (icon on the PC desktop)



2) You will need aluminium rail, Pasco ultrasound sonar “motion sensor”

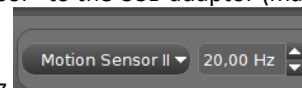


, Pasco USB adaptor



3) Connect the USB adaptor to the PC (via USB); connect the sonar “motion sensor” to the USB adaptor (make

sure that the connector faces the right way). Set a sampling rate of at least 50Hz



4) Set the cart at its start position, making sure that it is, at least, 10 cm away from the “motion sensor” (below such distance, position measurements are inaccurate).

5) Set the starting point (origin of position axis) using the position reset button





6) Start data acquisition



7) Stop the cart (by hand) before it reaches the end of the rail. Select and delete position data that are acquired after the cart has stopped.

#### Instructions for Bluetooth cart:

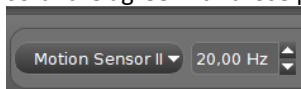
1) Launch the CAPSTONE software (icon on the desktop of laptop PC)



2) You will need aluminium rail, Pasco Bluetooth cart.



3) Turn the Bluetooth cart on, using the switch on its side, and connect the cart to the data acquisition software. To achieve this, click on the cart icon, making sure that characteristics and codes shown in the software agree with those printed on the cart. Set a sampling rate of at least 50Hz (for position measurements)

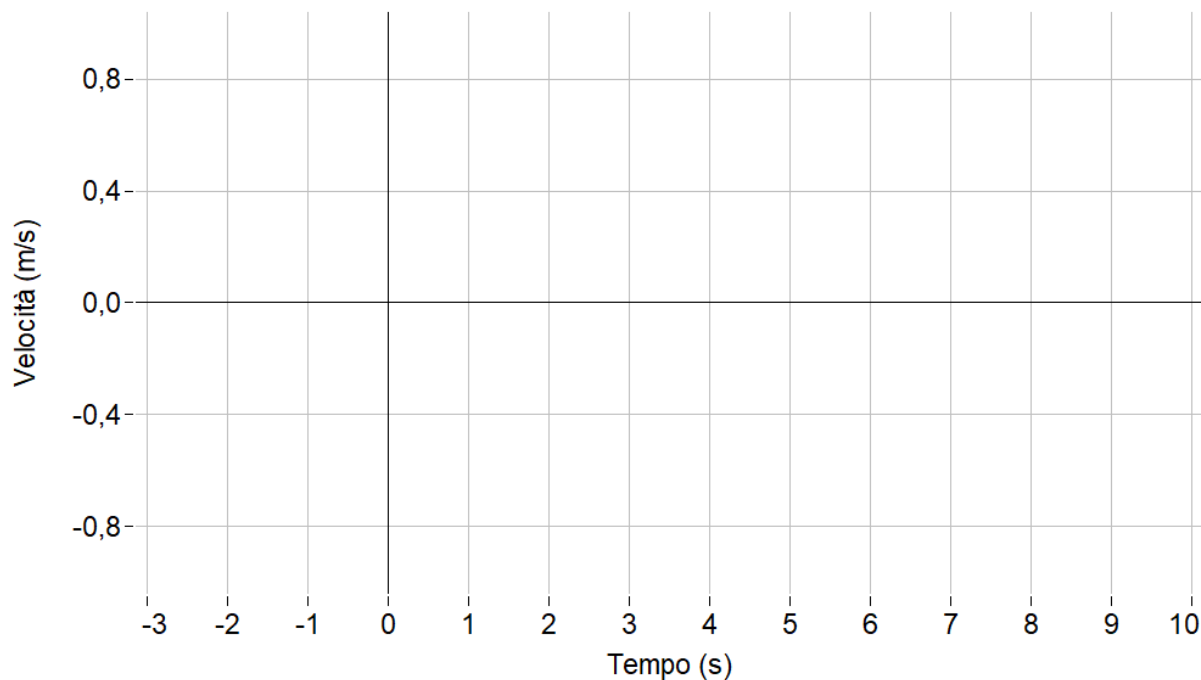


4) Set the cart on its start position (top of the rail)

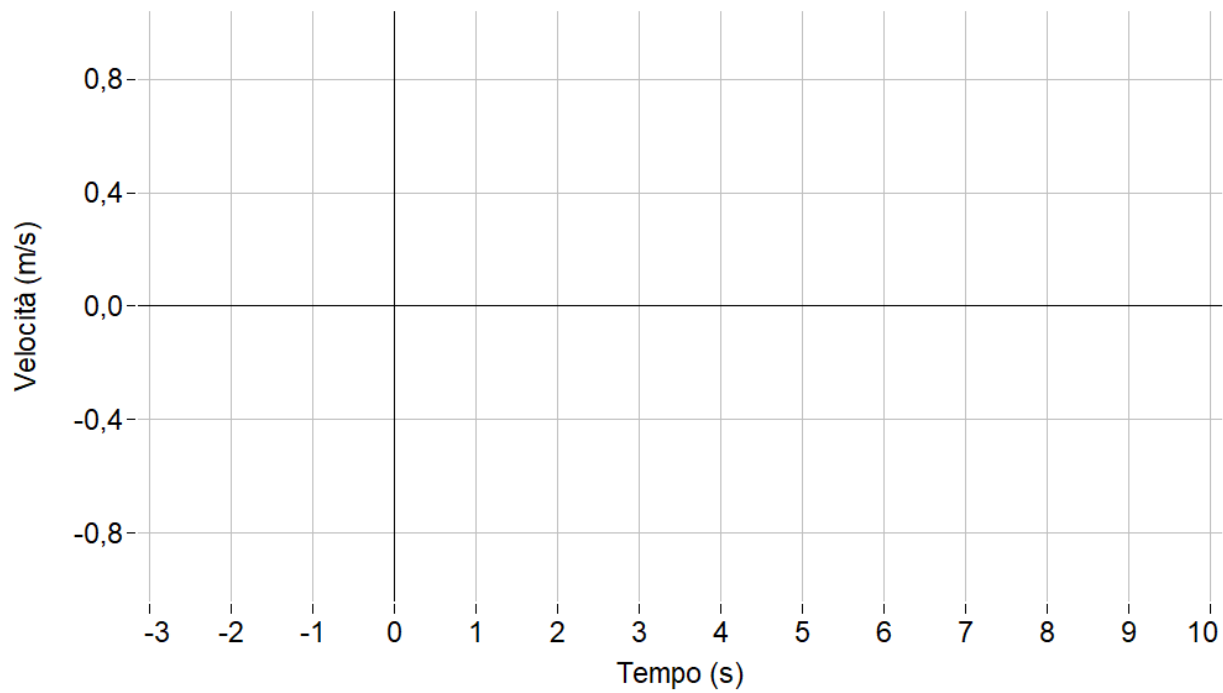
5) Check the direction: the cart should move towards the positive direction of the x-axis printed on it.

6) Stop the cart (by hand) before it reaches the end of the rail. Select and delete position data that are acquired after the cart has stopped.

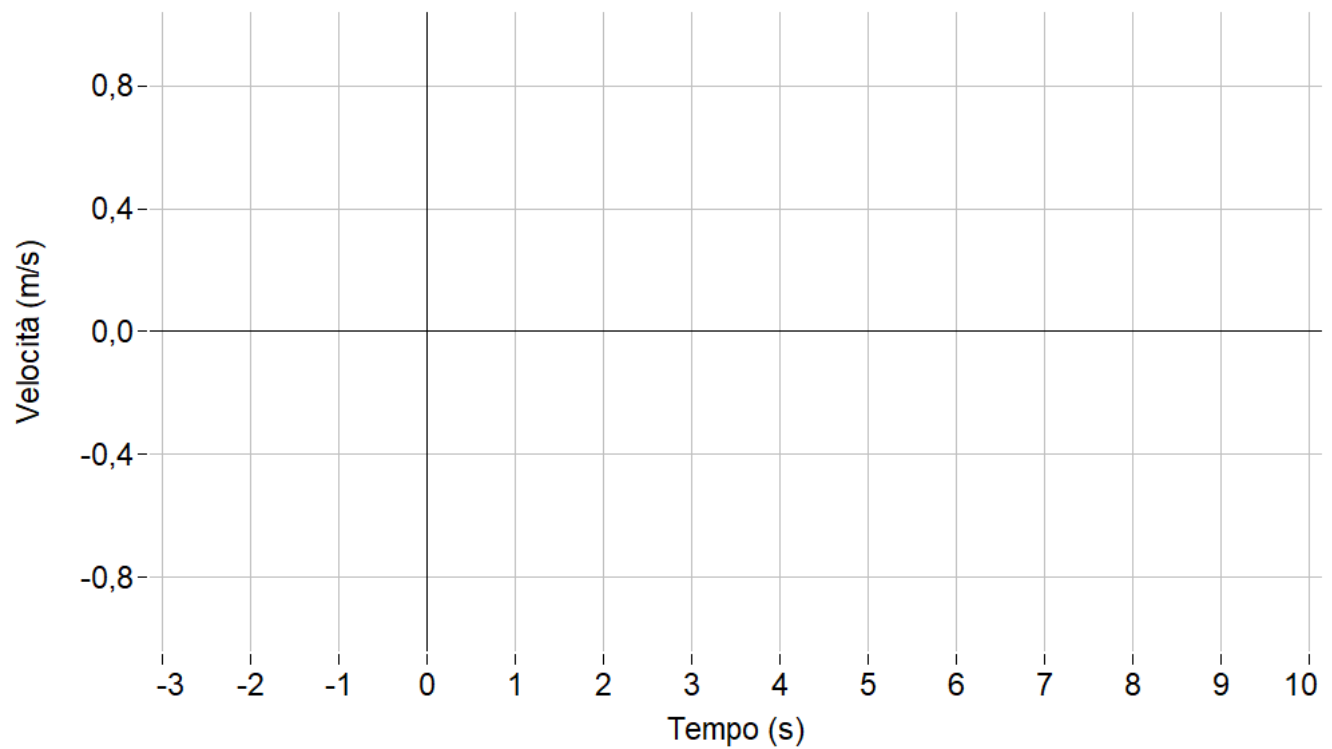
**Experiment (a): what will the velocity vs. time graph look like, if the cart is held still for a couple of seconds and then released?** Draw below, in different colours, both your expectation and the experimental results.



**Experiment (b): the same as above, but now setting the rail at a steeper slope.** Draw below, in different colours, both your expectation and the experimental results.



**DRAW BOTH GRAPHS FROM EXPERIMENTS 1 AND 2 (use different colours)**



Answer the following questions





- 1) Can you tell, by just looking at the graphs, in which case the rate of change of velocity is higher?

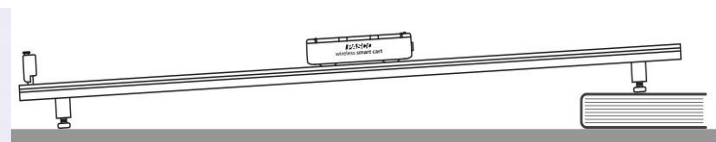
- 2) Calculate the slopes of the straight lines in the two graphs, after filling in the two tables below. To find velocity data you can use the “smart tool”<sup>1</sup> in CapStone.

TEST 1	$v_1(\text{m/s})$	$v_2(\text{m/s})$	$t_1(\text{s})$	$t_2(\text{s})$	$a(\text{m/s}^2)$

TEST 2	$v_1(\text{m/s})$	$v_2(\text{m/s})$	$t_1(\text{s})$	$t_2(\text{s})$	$a(\text{m/s}^2)$

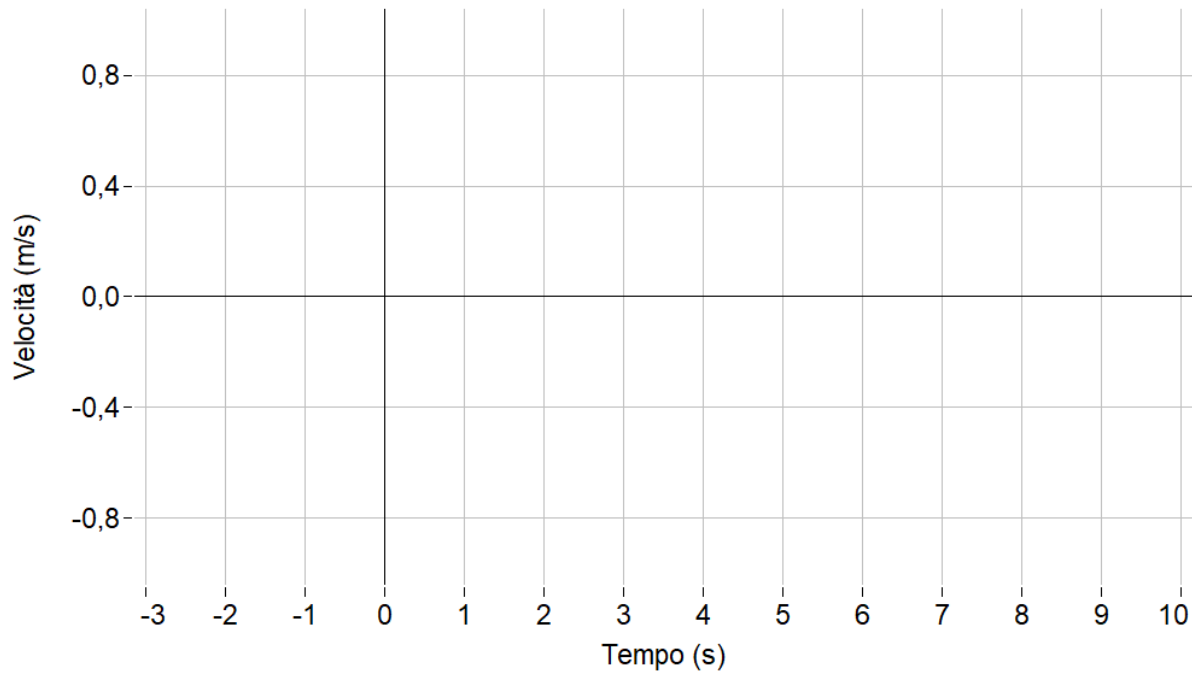
EXPERIMENT (c): Now we will change the “motion sensor” position, as in the picture below.

Using the Bluetooth cart, set the book below the rail, in such a way as to have an opposite slope (with respect to Experiments 3a and 3b).



What will the velocity vs. time graph look like now, if the cart is held still for a couple of seconds, and then released?

<sup>1</sup> One of the buttons in CapStone software, just above the graph.



Perform the experiment and compute the acceleration:

TEST 3	$v_1(\text{m/s})$	$v_2(\text{m/s})$	$t_1(\text{s})$	$t_2(\text{s})$	$a(\text{m/s}^2)$

Compare now the graph from this last activity to the former ones. You can see that the same falling motion of the cart along the rail (with the same slope) has two different graphical representations. The reason is that a new frame of reference was chosen, by moving the sonar sensor to a new position.

Conclusions:

---

---

---

---

---

---

---



Topic Motion  Acceleration measurement	Age  >14	Country  Poland	Date v.1 December 2019 / v.2 September 2021
--	----------------	-----------------------	---

*\*The original version of the experiment (v.1) was developed using the Arduino IDE. With the advent of the Raspberry Pi Pico microcontroller and its great possibilities, the description of the experiment has been adapted to MicroPython (v.2), which will make it much easier for students to perform it.*

## Function, realisation

Construction of a measuring instrument based on microcontroller and Python that measures the light intensity.

## Hardware required

- Microcontroller Raspberry Pi Pico or Raspberry Pi Pico W with Micropython 1.19 or above
- Cytron Maker Pico docking station for Pico
- ADXL345 module (accelerometer)
- 3V3 LCD 1602 I2C display from Seedstudio with Grove socket,
- PC or Mac computer.

## Materials required

- 1 standard Grove wire, 1xGrove female wire
- micro usb 2.0 high speed cable(30 cm or longer)
- powerbank (for standalone version)
- paper for note

## Software required

- Thonny App (thonny.org)
- library for ADXL sensor: [https://github.com/DFRobot/micropython-dflib/blob/master/ADXL345/user\\_lib/ADXL345.py](https://github.com/DFRobot/micropython-dflib/blob/master/ADXL345/user_lib/ADXL345.py)
- library for LCD:  
[https://files.seeedstudio.com/wiki/Grove\\_Shield\\_for\\_Pi\\_Pico\\_V1.0/Libraries.rar](https://files.seeedstudio.com/wiki/Grove_Shield_for_Pi_Pico_V1.0/Libraries.rar)

Using Thonny App You can save libraries inside lib folder in Pico memory.



Topic-Motion	Age	Country	Date
Optical photo gate – movement and speed measurement	>14	Poland	v1. December 2019 v.2 September 2021

\*The original version of the experiment (v.1) was developed using the Arduino IDE. With the advent of the Raspberry Pi Pico microcontroller and its great possibilities, the description of the experiment has been adapted to MicroPython (v.2), which will make it much easier for students to perform it.

## Function, realisation

Build experimental system based which can measures speed and movement.

## Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18 or above
- Cytron Maker Pico docking station for Pico (or different docking station for Pico: Waveshare, Seeedstudio),
- 1x Seeed studio Screw Terminal (or available on the market simple screw terminal for wire connection),
- IR beam interruption sensor - LED 5mm (or 3mm LED)- 0-25 cm, illumination angle of **20°** (In our case, the sensor is embedded in a self-made wooden housing or in a housing printed on a 3D printer)
- 3V3 LCD 1602 I2C display Seeedstudio with Grove Socket,
- PC or Mac computer.

## Materials required

- 2x Grove wires
- micro usb 2.0 high speed (15cm or 30 cm)
- paper for note

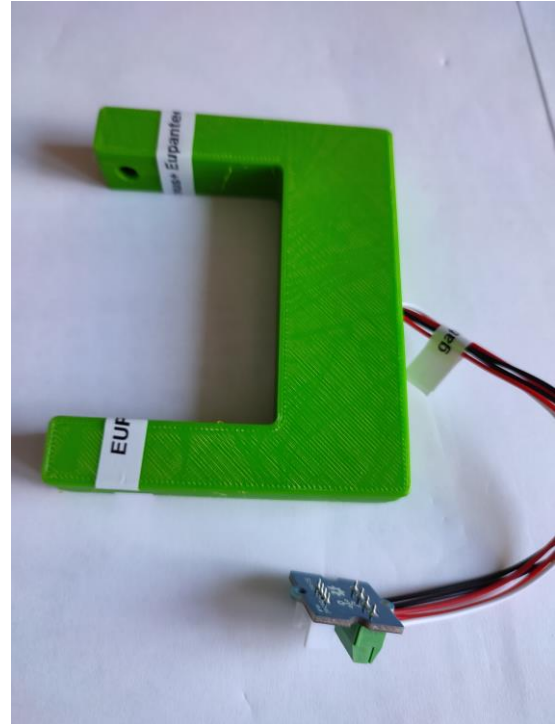
## Software required

- Thonny (thonny.org)
- llibrary for LCD1602:  
[https://files.seeedstudio.com/wiki/Grove\\_Shield\\_for\\_Pi\\_Pico\\_V1.0/Libraries.rar](https://files.seeedstudio.com/wiki/Grove_Shield_for_Pi_Pico_V1.0/Libraries.rar)
- script: gate\_time\_pullup1lcd



## Setup Hardware

Connections have been simplified thanks to the docking station and Grove cables. It is much harder to make a mistake or damage.



In above illustration: on the left IR beam sensor, on the right sensor inside 3D printed housing and Grove screw terminal



IR Beam sensor (photo gate) connection with Raspberry Pi Pico:

Gate (IR Beam)	Raspberry Pi Pico pins	Grove
Signal	GP9	yellow
not connected	GP8	white
power (+)	3V3	red



ground (-)	GND	black
------------	-----	-------

## LCD I2C screen connection with Raspberry Pi Pico:

LCD Pins	Raspberry Pi Pico pins	Grove wire
SCL	GP7	yellow
SDA	GP6	white
VCC	3V3	red
GND	GND	black

## Setup software

### Script:

MicroPython script measures how long the beam was interrupted and gives the result in milliseconds. After understanding the code and making small modifications, you can use the motion measurement system that you have built in other applications.

Please note in the script the **pull\_up** option for the pin where the gate is connected.

Timing is performed by instructions:

```
timer_start = utime.ticks_ms()  
period_time = utime.ticks_diff (utime.ticks_ms(), timer_start)
```

where

**utime.ticks\_ms()** them measures the number of milliseconds since the script was run.

The second instruction computes the difference between the two timestamps.

Low state (0) of the beam pin means "gate broken"

High state (1) means "gate open"



```
30 print("state:",val)
31 while True:
32     val=gate.value()
1 from lcd1602 import LCD1602
2 from machine import I2C,Pin
3 import utime
4 # setting I2C bus
5 i2c = I2C(1,scl=Pin(7), sda=Pin(6), freq=400000)
6 #LCD connected to GP6, GP7
7 d = LCD1602(i2c, 2, 16) #initialisation of LCD
8 d.display()
9 utime.sleep(1)
10 d.clear() timer_start)
11 d.print('Erasmus+ ')
12 utime.sleep(1)
13 d.setCursor(0, 1) #second row(1) selected for print, we count from 0
14 d.print('Eupantec2019.eu')
15 utime.sleep(1)
16 d.clear()
17 d.setCursor(0, 0)
18 d.print('Optical Gate')
19 d.setCursor(0, 1)
20 d.print('time')
21 utime.sleep(1)
22 d.clear()
23 d.setCursor(0, 0)
24 #Gate connected with Grove wire to GP8, Gp9
25 BEAM_PIN=9
26 print("Beam Ready")
27 #Important: PULL_UP option must be set
28 gate=machine.Pin(BEAM_PIN, machine.Pin.IN, machine.Pin.PULL_UP)
29 val=gate.value()
30 print("state:",val)
```

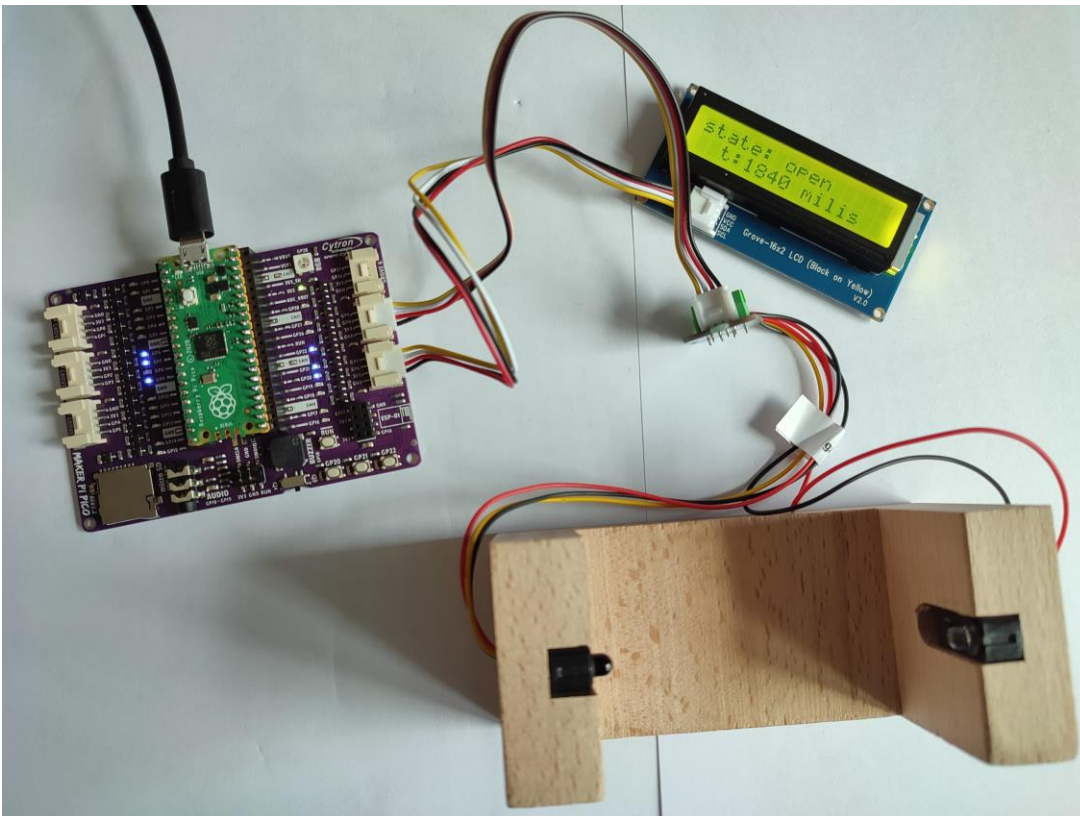
```
30 print("state:",val)
31 while True:
32     val=gate.value()
33     if val==0:
34         timer_start=utime.ticks_ms()
35         while val==0:
36             print("gate broken")
37             d.setCursor(0,0)
38             d.print('state: broken')
39             val=gate.value()
40             if val==1:
41                 #calculating the time
42                 period_time=utime.ticks_diff(utime.ticks_ms(), timer_start)
43                 d.clear()
44                 d.setCursor(2,1)
45                 d.print("t:"+str(period_time))
46                 d.print(" milis")
47                 utime.sleep(0.1)
48     else:
49         print("state: open")
50         d.setCursor(0,0)
51         d.print('state: open')
52         utime.sleep(0.1)
```



## Demonstration of the system operation:

### Go further- ideas, testing:

-place an item in the gate for a while, then remove it and read the time



- You can check how the sensor reacts to a glass filled with currant juice
- You can change the code for measure pendulum period
- You can apply 2 photo gates to count speed
- design a system that sends the gate state to a website or via bluetooth to a mobile application on your phone.



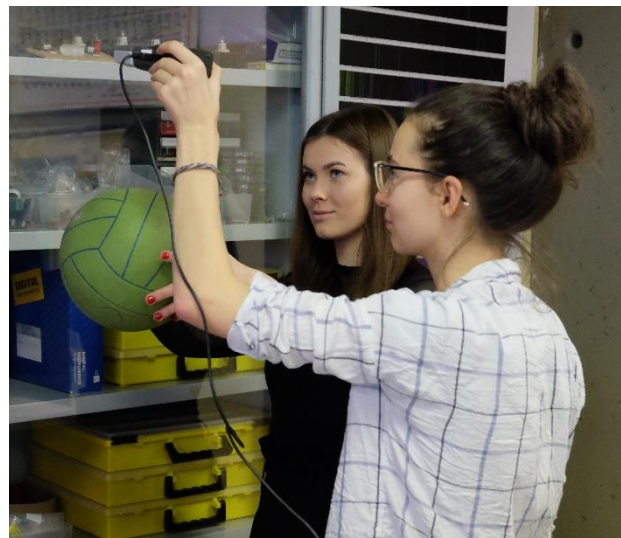


Topic	Age	Country	Date
Free fall Measuring $g$ Terminal velocity	>14	Portugal	Dec 2019

## Does the heaviest object fall first?

In this experiment, we'll drop several objects with different masses and follow their movements through a position sensor. We'll do it in two parts:

**Part I** - drop different **sports balls** with significantly different masses;



**Part II** - drop different numbers of **cup cake** paper forms.



## PART I - Sport balls

Drop different sports balls under the sensor and follow their movements with a position sensor.

Check the graph of the distance of the ball to the sensor in function of time, in the graphic calculator, and fill in the table of registration on the next page.

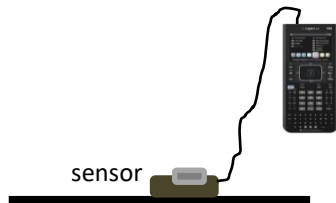
Consult the document in the attachments in order to better understand how to use the equipment, if necessary.



Ball details			Initial instant (when starts falling)	Final instant (when the floor is reached)	Duration of fall	Acceleration
sport	mass/g	Diameter (aprox.)/cm	$t_i/s$	$t_f/s$	$t/s$	$g/ms^{-2}$



## PART II – paper forms



Drop a different number off paper cup cake forms directly above the position sensor, from about your shoulder height.

Check the graph in the graphic calculator and fill in the table of registration.

Consult the document in attachments in order to better understand how to use the equipment, if necessary.

cup cakes			Initial instant (when starts falling)	Final instant (when the floor is reached)	Duration of fall	Terminal velocity
Number of cup cakes	Mass/g	Diameter (aprox.)/cm	$t/s$	$t_f/s$	$t/s$	$v_t/m \cdot s^{-1}$

Questions:



### Part I

**1.1-** In the sports balls' falling movement, has the time of fall significantly changed when the mass of the balls was varied?

(further exploration: percentaged, how much as the masses varied ? And the duration of fall?)

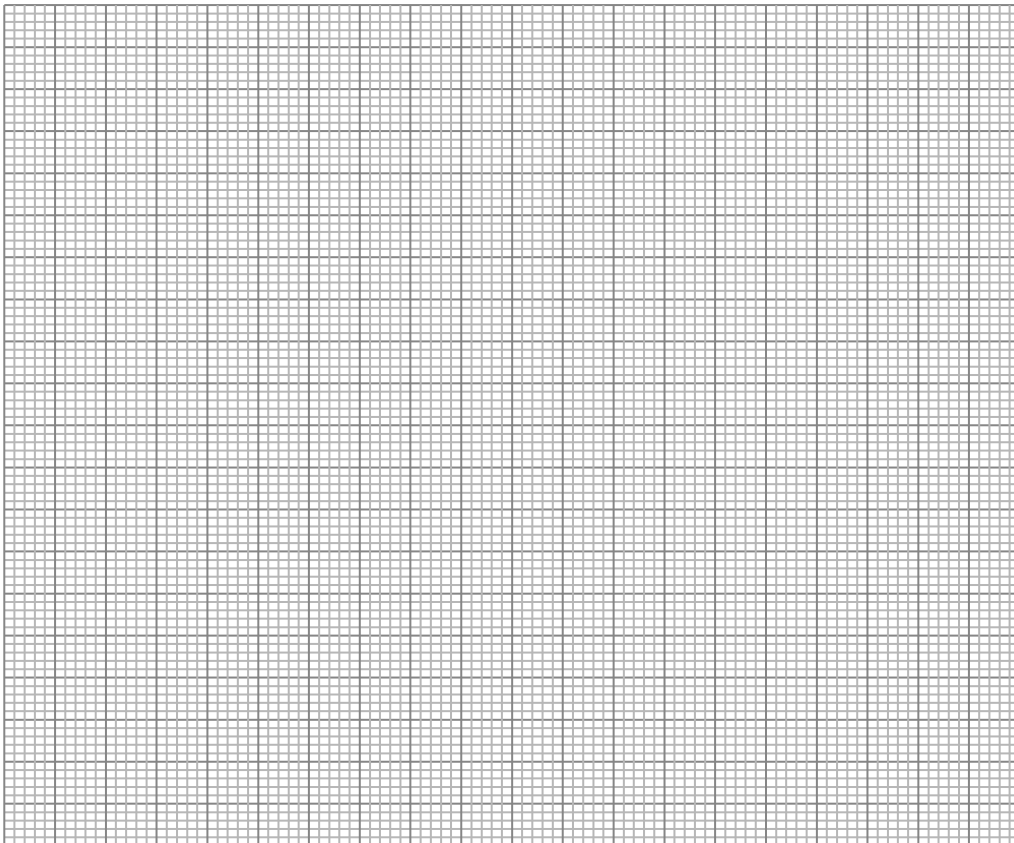
**1.2-** Has the mass off the falling ball affected the acceleration of the motion?

(further exploration: percentaged, how much as the masses varied? And the acceleration?)

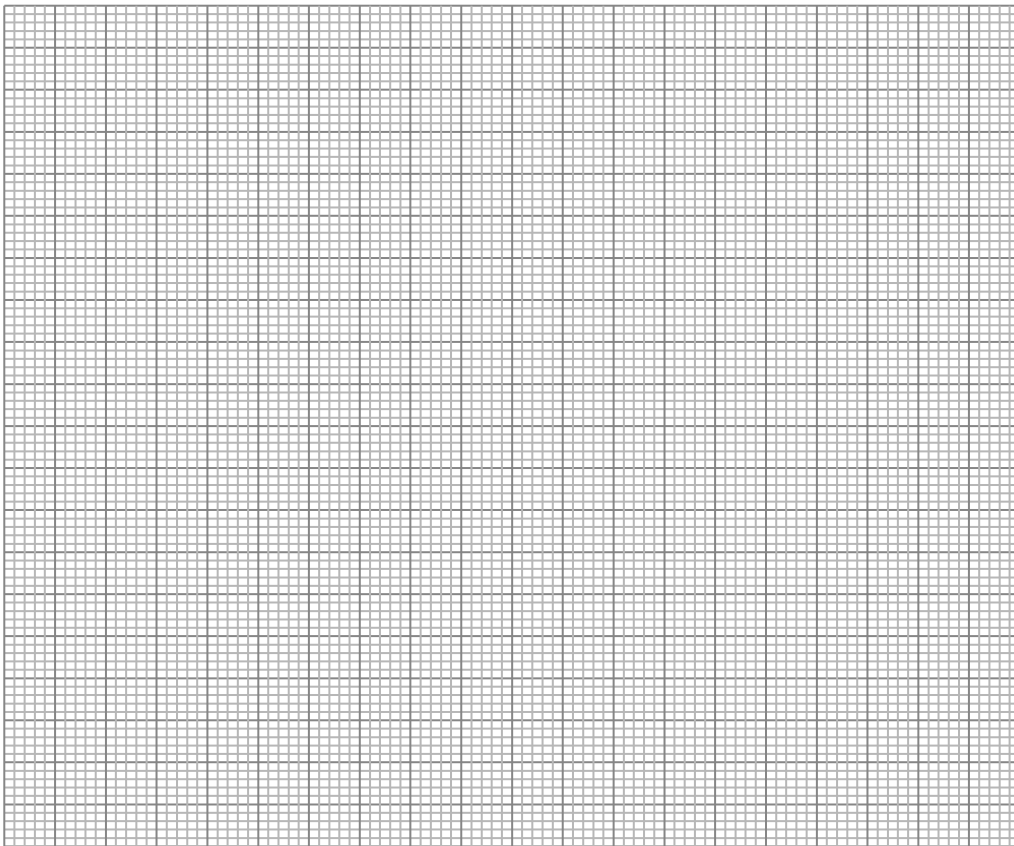
**1.3-** "An object falls with a constant acceleration of  $10 \text{ m}\cdot\text{s}^{-2}$ ". What does this mean?

### Part II

**2.1-** Draw an outline of the duration of fall vs number of forms graph.



**2.2-** Draw an outline of terminal velocity vs number of forms graph.



**2.3-** In the cupcake falling movement, has the duration of fall significantly changed has the number and mass varied?

**2.4-** What is the relationship between the mass of the objects dropped (number of cup cake forms) and the terminal velocity acquired?

**2.5-** What would it be the final velocity of fall if f there were 10 filters falling?

### Global Questions

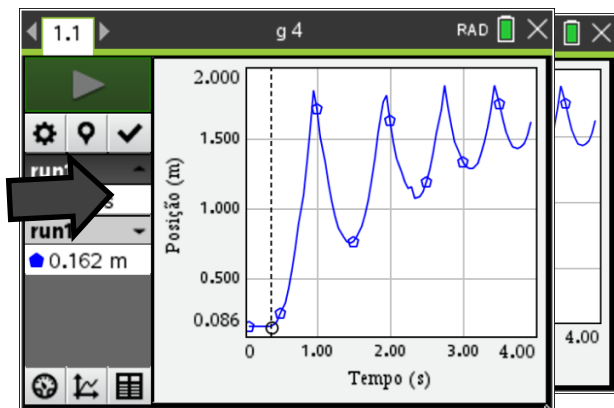
**3.1 –** In which situation - sports balls or in the cup cake forms – has the air represented a significant obstacle in the motion?

**3.2 – Does the heaviest object fall the first?**

## Attachment 1 - Consulting the graphs for obtaining the times

1. Connect a CBR 2™ to the TI-Nspire™ handheld.
2. Place the ball under the CBR 2.
3. Click the Start button in the DataQuest app to begin sampling.
4. When you hear the CBR 2 begin clicking, drop the ball.
5. When the ball lands on the floor, click the Stop button .
6. On the main screen, select graph view and then select point you need, by moving your cursor.

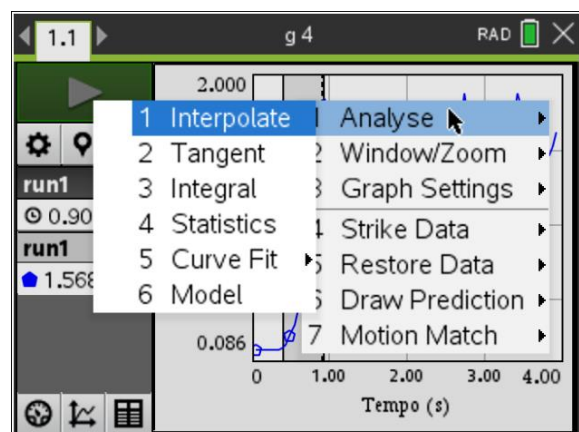
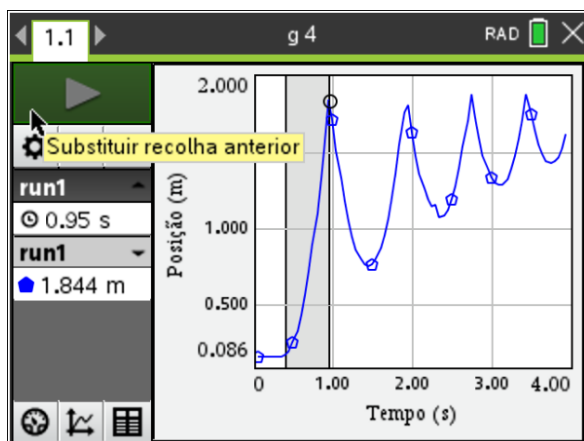
Instant when the  
ball starts to fall



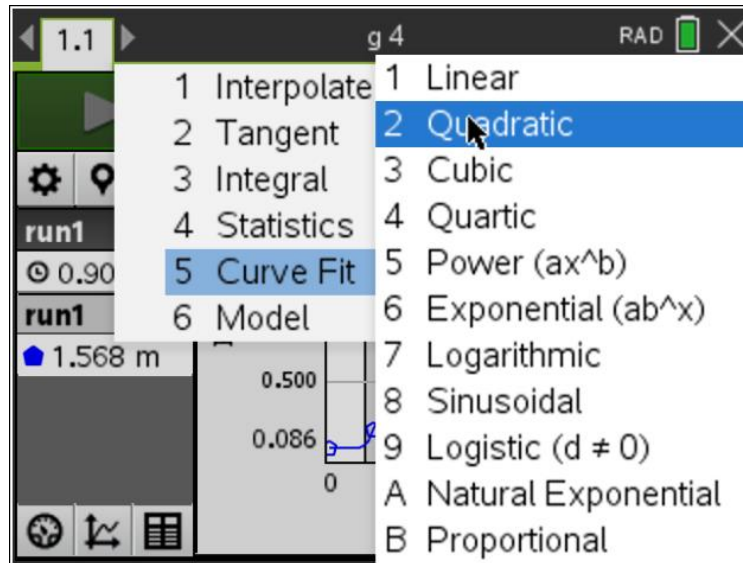
Instant when the  
ball reaches the floor

## Attachment 2 - Consulting the graphs for obtaining the acceleration

1. On the main screen, select graph view and then select the point you need, and then select the region that represents the ball falling by moving your cursor to the starting point of the drop and clicking to set the left bound. Move the cursor to the right to expand the selection region, and click to select the right bound
2. Analyse the parabolic segment by selecting **Click Menu > 4.analyse > 6.curve fit > 2.quadratic**

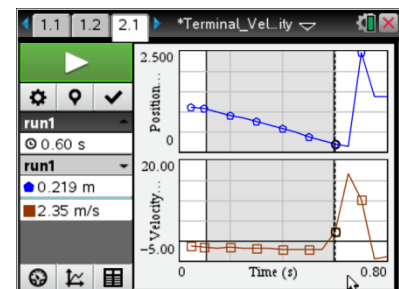






### Attachment 3 - Consulting the graphs for obtaining the terminal velocity

1. Connect a CBR 2™ to the TI-Nspire™ handheld.
2. Place the CBR 2 on the floor facing upward.
3. Hold a cupcake paper form directly above the probe at your shoulder height.
4. Click the Start button in the DataQuest app to begin sampling.
5. When you hear the CBR 2 begin clicking, drop the ball.
6. When the ball lands on the floor, click the Stop button .
7. On the main screen, select graph view and then select point you need, and then select the region that represents the coffee filter falling by moving your cursor to the start point of the drop and clicking to set the left bound.
8. Move the cursor to the right to expand the selection region, and click to select the right bound.
9. Analyse the linear segment by selecting **MENU > 4.Analyze > 6.Curve Fit > 1.Linear** and finding the velocity of the single cupcake paper form.





Kozienice

Zespół Szkół Nr 1 im. Legionów Polskich

# ENERGY AND ENVIRONMENT

12.10. – 15.10.2021





Topic	Age	Country	Date
Drive a rover with Python Regular polygon	>15	France	Oct 2019

## Drive a programmed rover in Python

### Experimental setup materials :

- 1 calculator TI
- 1 rover TI
- 1 hub (microcontroller)



### Experimental setup and procedure :

Start the **TI-Innovator Rover**. Choose **prgm** on the calculator.

Run this program :

```
ÉDITEUR : PLYGONE
LIGNE DU SCRIPT 0009
import ti_rover as rv
n=int(input("number sides "))
d=input("lenght ")
def polyg(n,d):
    for i in range(n):
        rv.forward(d)
        rv.left(360/n)
polyg(n,d)
```

### Experiment evaluation:

- 1) Describe and explain the passage the rover takes. You can use a marker in the hole so that the rover can trace his passage.
- 2) Create a new program, using a loop and a function Python.



Topic	Age	Country	Date
Drive a rover with Python Emergency stop	>15	France	Oct 2019

## Drive a programmed rover in Python

### Experimental setup materials :

- 1 calculator TI
- 1 rover TI
- 1 hub (microcontroller)

### Experimental setup and procedure :

Start the **TI-Innovator Rover**. Choose **prgm** on the calculator.

Run this program :

```
ÉDITEUR : LUMIERE  
LIGNE DU SCRIPT 0005  
import ti_rover as rv  
d=rv.ranger_measurement()  
print(d)  
if d>0.2:  
    rv.forward(10*d-1)_
```



### Experiment evaluation:

- 1) Describe and explain the passage the rover takes.
- 2) Create a program, using a loop that the rover travels a regular polygon.

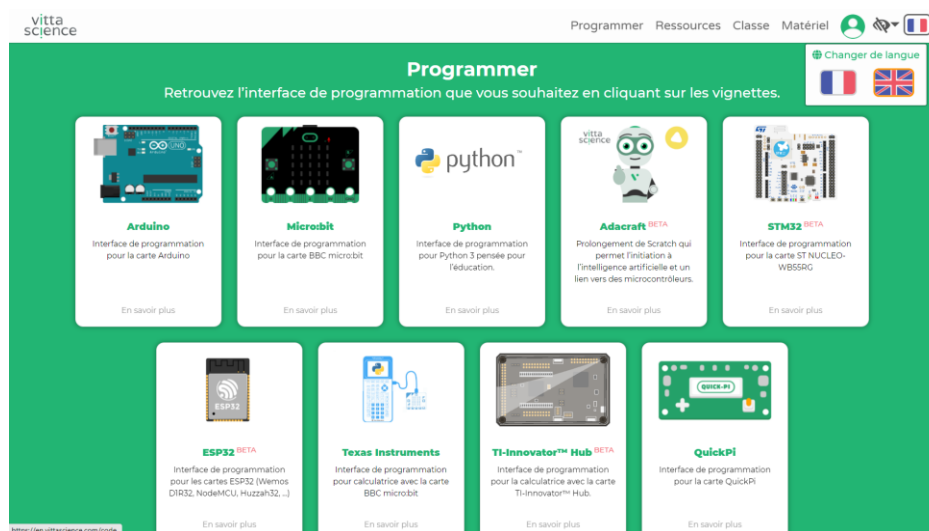


Topic	Age	Country	Date
Programming an online temperature sensor	>15	France	Oct 2019

## Simulate a Python sensor program

### Experimental setup materials :

1 laptop or a computer with internet connection



### Experimental setup and procedure :

Go to the website Vittascience [en.vittascience.com/python](https://en.vittascience.com/python)

### Experiment evaluation:

- 1) Reproduce this program and run it. Observe the Python code.
- 2) Create another program of your choice, using another sensor.





Topic	Age	Country	Date
Programming a microcontroller ESP8266 for measures of temperature and light	>15	France	Oct 2021

## Python sensor program

### Experimental setup materials :

- 1 laptop or a computer with internet connection
- 1 microcontroller ESP8266
- 1 numeric sensor of temperature
- 1 analogic sensor of light

### Experimental setup and procedure :

With Thonny, use the program : *copy and paste the Python code*

```
from machine import Pin, I2C, ADC
```

```
import ssd1306
```

```
import time
```

```
i2c = I2C(-1, Pin(5), Pin(4)) # liaison serie : Pin4 = Serial Data, Pin5 = Serial Clock
```

```
display = ssd1306.SSD1306_I2C(128, 64, i2c) # resolution de 128x64
```

```
buffer = bytearray(2)
```

```
CaptLum = ADC(0)
```

```
while True:
```

```
    M = CaptLum.read()
```

```
    buffer = i2c.readfrom(0x49, 2)
```

```
    N = (buffer[0] << 3) | (buffer[1] >> 5)
```

```
    Temp = N * 0.125 # resolution LM75A is 0,125 °C
```

```
    display.fill(0) # Efface l'écran
```

```
    display.text("Temp= " + str(Temp) + " C", 0, 20)# + concatenation operator
```

```
    #function str() convertes number on string
```

```
    if M >= 1023 : M = 1023
```

```
    display.text("light = " + str(M), 0, 10)
```

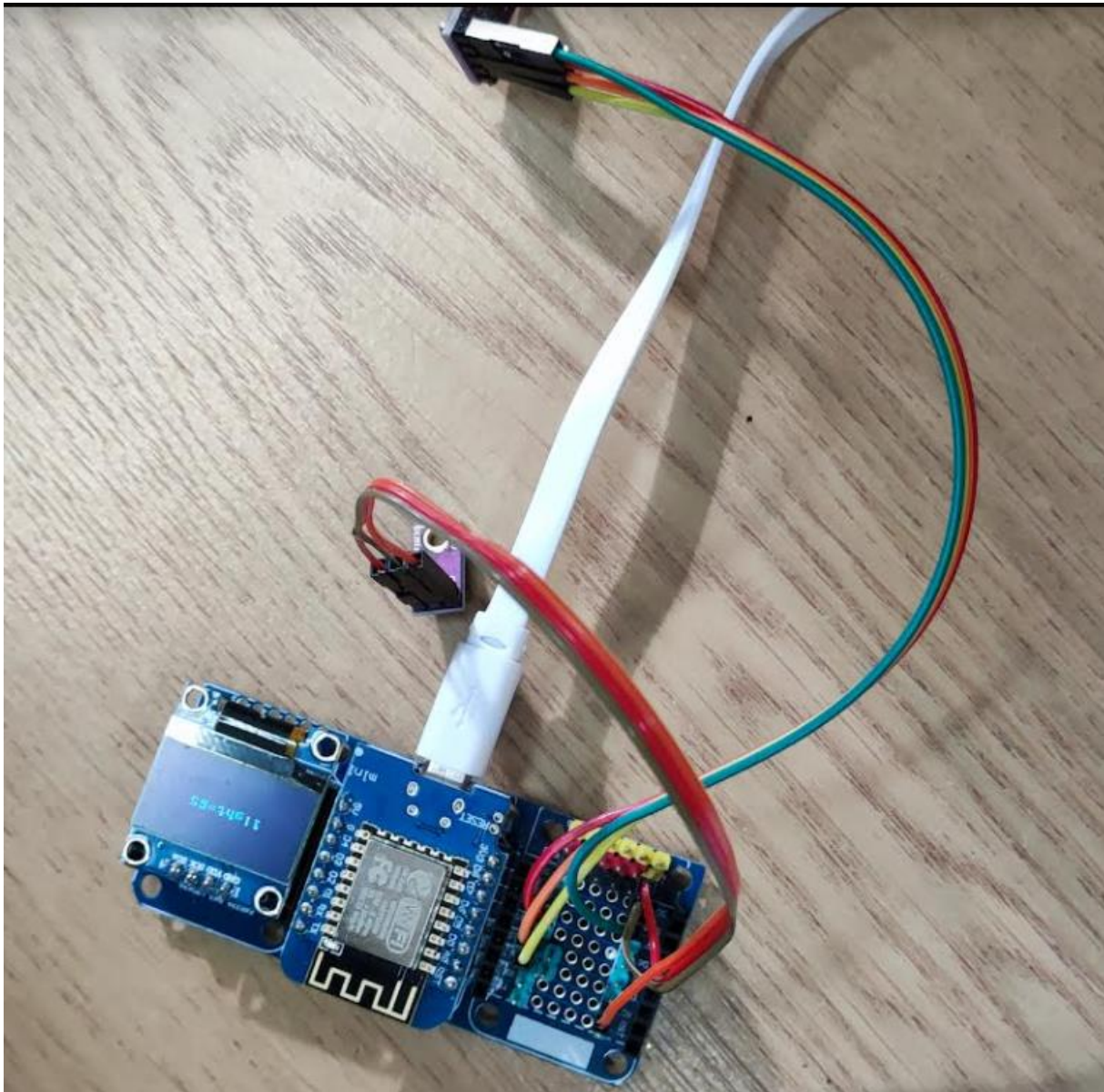




```
display.show()  
time.sleep_ms(1000) # each one seconde
```

### Experiment evaluation:

- 1) Connect the two sensor on the microcontroller ESP8266, like of the photo :



- 2) Run this program and look at the results displayed on the OLED screen.





Topic	Age	Country	Date
Centripetal acceleration (Phyphox)	>14	Germany	Oct 2021

- Rotating “machine” (salad spinner, special chair, wheel etc.)

This program visualizes centripetal acceleration as a function of angular velocity.

To see this, you need to measure at different angular velocities but a fixed radius.

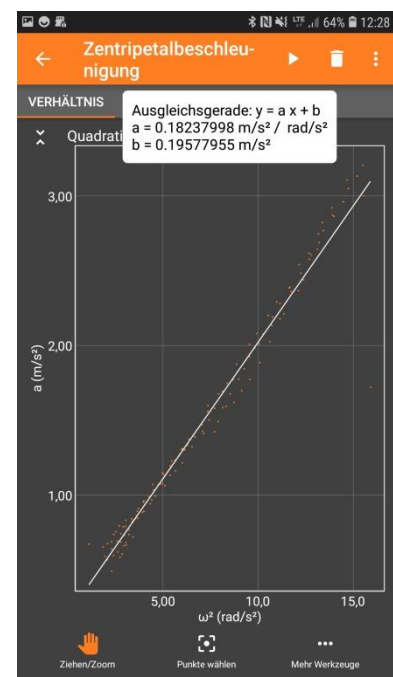
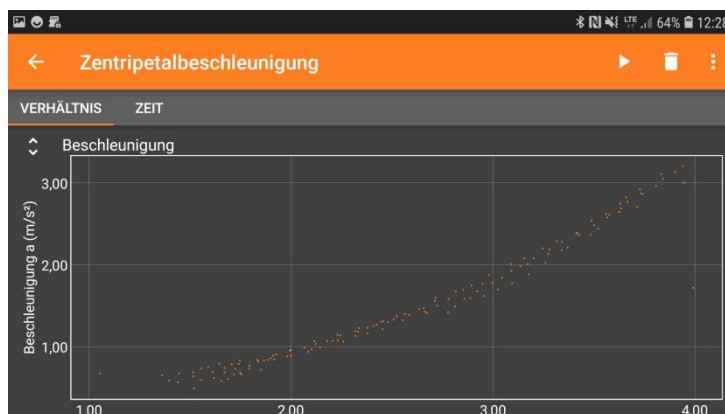


Smartphone fixed on a rotating wheel



Smartphone fixed inside a salad spinner:

After the experiment you can analyse the graphs in Phyphox by tapping on a diagram:



The relation  $a_z \propto \omega^2$  is a logical consequence from both diagrams.



Topic	Age	Country	Date
Distance measurement with phyESPx	>14	Germany	Oct 2021

## Rolling bottle

### Experimental setup materials:

- 1 phyESPx server with attached HC-SR04 ultrasonic measurement hat
- 1 tablet/laptop for evaluation
- 1 bottle (filled with water completely)

### Experimental setup and procedure:

Turn on the **phyEXPx** server and access the web interface by scanning the QR code on top of the server or by entering the IP address (you will find it right beneath the QR code) in a browser tab. Put the sensor kit down on a flat surface (eg. a table) and make sure the distance sensor is **facing parallel to the surface**. **Remove all obstacles in the range of the sensor** (about 30° - 1,5 meters) and place the bottle right in front of it.

Now you can start the measurement by pressing the **start** button (or the **play** button on the remote device). If you can see the first data values displayed, you can **roll the bottle away from the sensor** by giving it a **gentle push**.

Once the bottle has stopped rolling or is out of sensor range, you can stop the measurement and evaluate your data.



### Experiment evaluation:

- 1) Describe the t-x-diagram.
- 2) Try to imagine the fitting t-a-diagram.
- 3) Guess why the distance might only be accurate at 20°C

Topic	Age	Country	Date
Wireless weather sensor	>14	Italy	Oct 2021

The Wireless Weather Sensor is an all-in-one instrument for monitoring complex environmental conditions. It houses several sensing elements within a single unit to provide 19 different measurements.



We use the sensor as a handheld instrument to study microclimates and record ambient conditions relevant to environmental phenomena, such as weather and light measurements:

- Ambient Temperature
- Barometric Pressure
- Relative Humidity
- Wind Speed
- Ambient Light (lux)

We transmit data wirelessly to our device (computer, tablet and mobile) for classroom analysis when group activities are constrained by time, within SPARKvue software.



### Procedure

1. Select a location for your experiment according to your teacher's instructions.
2. Start a new experiment on the data collection system (Sparkvue)
3. Attach the weather sensor to the data collection system.
4. Display barometric pressure, temperature, relative humidity and wind speed in a table.
5. Change the sample rate to once every 10 minutes.
6. Keep the data collection system off the ground.
7. Keep the direct sun from shining on your system by placing a weather shield over it.
8. Start data recording.
9. After the data has been collected for a few minutes, stop recording data.
- 12 Save your experiment.
- 13 Return the equipment and the data collection system to the classroom.



Topic: Energy Field experiments. Data logger	Age  >15	Country  Poland	Date  October 2021
---	----------------	-----------------------	-----------------------------

## Function, realisation

In this manual, we will show you how to record measurement results during field experiments. We will describe an easier way - saving the results on the internal Pico memory. We will use the internal thermometer of the Raspberry Pi Pico as the sensor. This script can be easily adapted to record measurements from other sensors, e.g. smog, pressure, water purity or many others, including the time of measurement or coordinates describing the place of measurement taken from a GPS connected to Pico. We will show you how to save the measurement results in the Pico's internal memory as a CSV file, which can be easily processed later by a spreadsheet or other scripts using the **Matplotlib** library for plotting graphs. We will also present a more difficult solution - saving the results on a MicroSD card.

## Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18
- Cytron Maker Pico docking station for Pico
- Optional 3V3 LCD 1602 I2C display from Seeedstudio with Grove Socket,

## Materials required

- micro usb 2.0 high speed (15cm or 30 cm),
- powerbank,



- MicroSD card (max 16 GB),
- optional 1 standard Grove wire.

## Software required

- Thonny (thonny.org)
- library for LCD (script with LCD only):  
[https://files.seeedstudio.com/wiki/Grove\\_Shield\\_for\\_Pi\\_Pico\\_V1.0/Libraries.rar](https://files.seeedstudio.com/wiki/Grove_Shield_for_Pi_Pico_V1.0/Libraries.rar)
- sdcard library (script with MicroSD reader support only):  
<https://github.com/micropython/micropython/blob/master/drivers/sdcard/sdcard.py>

## Setup software

Below Python code for saving in microcontroller using Thonny App. Write the script in Thonny, then select "File/Save" and select Raspberry Pi Pico as destination. Remember please to add .py at the end of filename.

```
1 from utime import sleep
2 from machine import ADC
3 #thermometer initialization (on the Pico board)
4 sensor_temp = ADC(ADC.CORE_TEMP)
5 #Factor for converting ADC indications to voltage
6 conversion_factor = 3.3 / (65535)
7 #filename assignment and opening for writing
8 file = open("measurements.csv", "w")
9 while True:
10     reading = sensor_temp.read_u16() * conversion_factor
11     #Temperature calculations based on the sensor characteristics
12     #rounding to one decimal place
13     temperature = round(27 - (reading - 0.706)/0.001721,1)
14     print(temperature)
15     file.write(str(temperature) + "\r\n")
16     #flush()-works like the "Close" statement, but the file remains open
17     file.flush()
18     sleep(5)
```



## Code analysis

The **machine** module contains specific functions related to the hardware on a particular board. We import methods related to ADC- Analog Digital Converter (line 2).

The **utime** module (line 1) provides time functions. In this case we use **sleep** method for delay.

In lines number 4,6 we make thermometer initialisation (on the Pico board) and set factor for converting ADC indications to voltage.

You can also initialise another sensor here. In the eighth line we make filename assignment and opening for writing Any data obtained from the sensor are saved in the file: **measurements.csv**. The file is opened for writing (w), its previous contents are erased.

On lines 9-18 there is an infinite loop.

Lines 10 and 13 are used for read the data from the sensor and convert them using the appropriate conversion factor. The "**round**" method rounds the result to one decimal place

Lines 15 and 17 are essential for writing to the file.

The flush() method in Python file handling clears the internal buffer of the file. In Python, files are automatically flushed while closing them. However, a programmer can flush a file before closing it by using the flush() method and finally save results in a file.

If we want to save the values of several variables in a file, it is worth using tabs to separate columns:

```
file.write(str(press)+"\t"+str(temp)+"\t"+ str(humid)+"\r\n")
```

In above example we write values of 3 variables: press. temp and humid as strings separated by tabs.

Notice: \t means tab, \r means return, \n means "new line".





Stored file has the the name with CSV

extension, then this text file may be interpreted as database of results by many apps, easy to analyse and process.

## Data recording in time

The last line “**sleep(5)**” makes the temperature readable every four seconds.

If you want to measure the value every minute, you have to change 5 to 60, one per hour: 3600. This value can be freely changed, but not more often than the maximum fixed for a given sensor. In practice, there is no need to write frequently, because the temperature, for example, does not change quickly. If we do not save the results often, Pico memory will be enough for a longer time

## Optional version with lcd screen support

```
1 from utime import sleep
2 from machine import ADC, Pin, I2C
3 from lcd1602 import LCD1602
4 #thermometer initialisation (on the Pico board)
5 sensor_temp = ADC(ADC.CORE_TEMP)
6 #Factor for converting ADC indications to voltage
7 conversion_factor = 3.3 / (65535)
8 #initialisation: I2C bus for LCD connected to GP6, GP7
9 i2c = I2C(1, scl=Pin(7), sda=Pin(6), freq=400000)
10 #LCD initialisation
11 d = LCD1602(i2c, 2, 16)
12 d.display()
13 #Clear screen
14 d.clear()
15 #filename assignment and opening for writing
16 file = open("measurements.csv", "w")
17 while True:
18     reading = sensor_temp.read_u16() * conversion_factor
19     #Temperature calculations based on the sensor characteristics
20     #rounding to one decimal place
21     temperature = round(27 - (reading - 0.706)/0.001721, 1)
22     print(temperature)
23     d.setCursor(0, 0)
24     d.print("int. temperature:")
25     d.setCursor(4, 1)
26     d.print(str(temperature)+"C")
27     file.write(str(temperature) + "\r\n")
28     #flush()-works like the "Close" statement, but the file remains open
29     file.flush()
30     sleep(5)
```



We must add libraries required to handle the display. In above script lines 9-14 define the display and lines 23-26 are for printing results on LCD display. **Note that according to the code, the lcd should connect to GP6, GP7.**

**Sample content of the file with measurement results:**

1	16.3
2	19.6
3	19.6
4	19.6
5	19.6
6	19.6
7	19.1
8	19.6
9	19.1
10	20.0
11	19.6
12	19.1
13	19.6
14	19.6
15	

## Performance of the experiment

If you want have autonomous system then save the script on the Pico with special name: **main.py**. This script will start automatically (No need to connect to a computer). In this case you can power the Pico from powerbank. Firstly you have to connect board to powerbank (or other type of electric source) through the micro USB wire.

**Warning!** If you use the name **main.py** (with autostart) and you want save the file with results, detach the LCD screen. This will allow you to terminate the script before erasing the file.



## Evaluation, testing:

- you must remember, that every time you start the script, then you erase data in file
- you can adjust information stored in file or use with different sensor
- estimate the file size assuming that the script runs for 1 day and the temperature is recorded once every 60 seconds
- open the CSV file with a spreadsheet and use it to chart (from Pico)

```
1 from utime import sleep
2 from machine import Pin, ADC, SPI
3 import sdcard
4 import uos
5 """ SPI mode Cytron Maker Pico shield
6 micro SD socket (description from the board)
7 GP10-SCK (SCLK)
8 GP11-SDI (SDI<---mosi)
9 GP12-SDO (SDO<---miso)
10 GP15-CSN (chip select)
11 All internal connections, no wires needed,
12 you don't have to connect anything"""
13 # Assign chip select (CS) pin (and start it high)
14 cs = Pin(15, Pin.OUT)
15 # Initialize SPI peripheral (start with 1 MHz)
16 spi = SPI(1,
17          baudrate=1000000,
18          polarity=0,
19          phase=0,
20          bits=8,
21          firstbit=SPI.MSB,
22          sck=Pin(10),
23          mosi=Pin(11),
24          miso=Pin(12))
25 # Initialize SD card
26 sd = sdcard.SDCard(spi, cs)
27 # Mount filesystem
28 vfs = uos.VfsFat(sd)
29 uos.mount(vfs, "/sd")
30 #filename assignment and opening for writing
31 file = open("/sd/measurements.csv", "w")
32 #termometer initialisation
33 sensor_temp = ADC(ADC.CORE_TEMP)
34 conversion_factor = 3.3 / (65535)
35 while True:
36     reading = sensor_temp.read_u16() * conversion_factor
37     temperature = round(27 - (reading - 0.706)/0.001721,1)
38     print(temperature)
39     file.write(str(temperature) + "\r\n")
40     #flush()-works like the "Close" statement, but the file remains open
41     file.flush()
42     sleep(5)
```

internal memory or MicroSD card.

- consider what sensors you could use for field measurements

**Go further\*\*:**

- write a Python script that will automatically display the chart based on the CSV file using the **Matplotlib** library
- prepare a project that reads the coordinates from the GPS module, adjust it to work with the logger.
- write a Python script that will read the **measurements.csv** file from the MicroSD card and display its contents in the console

**Below data logger inside Pico with our 3D printed housing**





Topic: Energy  DS18B20-energy efficiency measurement.	Age  >14	Country  Poland	Date  October 2021
---	----------------	-----------------------	--------------------------

## Function, realisation

Construction of a measuring instrument based on microcontroller with Micropython and digital waterproof thermometer, which can be used for measurement of energy efficiency, equipment efficiency.

## Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18 or above
- Cytron Maker Pico docking station for Pico
- DS18B20 waterproof probe -digital thermometer (3.3V for Pico) with **onewire** support
- 3V3 LCD 1602 I2C display from Seeedstudio with Grove socket,
- PC or Mac computer.

## Materials required

- minibreadboard
- 4,7 kOhm resistor
- 1xstandard Grove wire, 1xGrove male wire
- ARK4 screw terminal for breadboard (minimum screw 3 inputs)
- micro usb 2.0 high speed cable(15cm or 30 cm)
- LCD library:  
[https://files.seeedstudio.com/wiki/Grove\\_Shield\\_for\\_Pi\\_Pico\\_V1.0/Libraries.rar](https://files.seeedstudio.com/wiki/Grove_Shield_for_Pi_Pico_V1.0/Libraries.rar)
- powerbank (for standalone version)
- paper for note

## Software required

- ThonnyApp (thonny.org),
- ds18b20 use library included in MicroPython (onewire, ds18x20),
- library for LCD.

Using Thonny App You can save library inside lib folder in Pico memory.

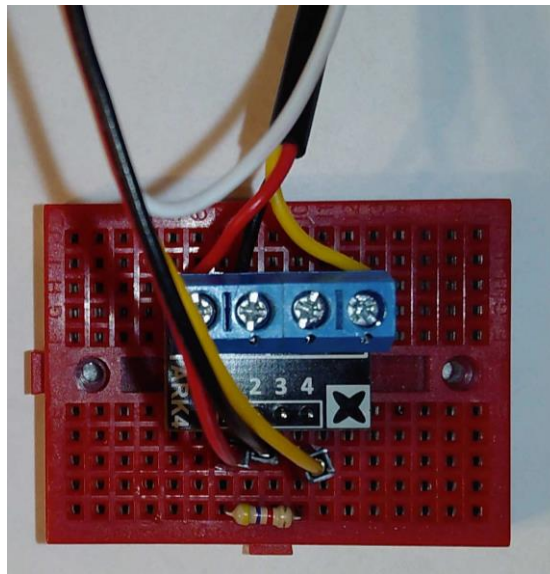
## Technical specifications of DS18B20 digital thermometer:

- Supply voltage: 3 V to 5 V

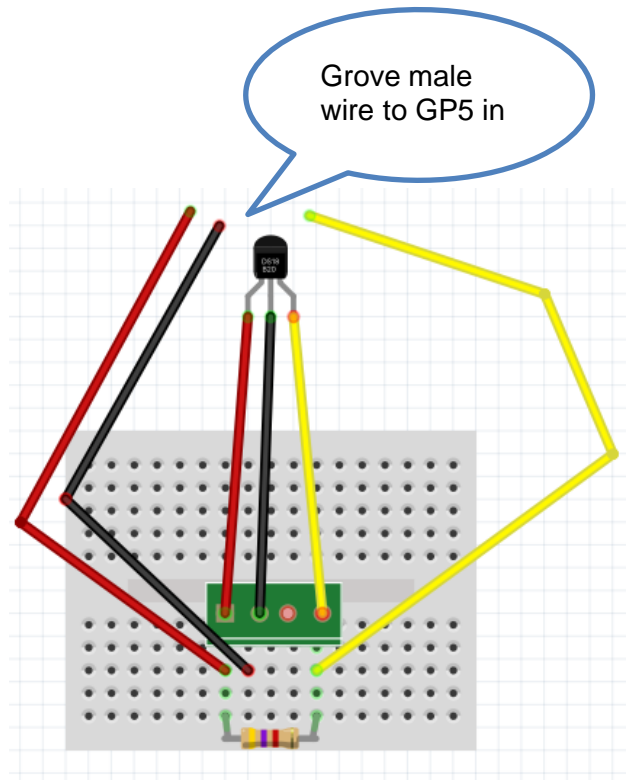
- Operating temperature range: -55°C to 125°C
- Storage temperature: from -55°C to 125°C
- Accuracy:  $\pm 0.5^\circ\text{C}$  in temperature range from -10°C to 85°C

### Setup Hardware

**DS18B20** digital thermometer support 1 wire. This allows multiple thermometers to be connected to one signal line. They are recognised by a unique address in each thermometer. This is very convenient if you need multiple thermometers. We don't even need to modify the program, it will work fine.



DS18B20 thermometer pins	Raspberry Pi Pico pins	Grove male wire
VCC (red)	3V3	red
GND (black)	GND	black
Signal (yellow)	GP5	yellow
not connected	GP4	white



**Important! Use mini breadboard and a 4.7k Ohm resistor as a pull-up resistor to the signal pin of the thermometer. Best of all, the circuit will not work.**

Resistor connection:

Resistor 4.7kOhm	Grove male wire	DS18B20 ARK 4 screw terminal
1. leg line	red	VCC (1)
-----	black	GND (2)
-----	white	not connected (3)
2. leg line	yellow	signal (4)

First resistor leg at the same line like red wire, second resistor leg at the same line like yellow - signal wire.





## Note the offset of the terminal pins relative to the screw connections!

The Pico use 3.3V power (not 5!!!). We usually connect the red wire to pin 3V3 and black wire to the ground (GND). SDA and SCL are the pins used for I2C communication.

We suggest using the LCD screen to display the results. In this case, we can also build an autonomous, mobile system that can work without a computer. The use of grove cables makes the connection as easy as possible.

LCD 1602 pins	Raspberry Pi Pico pins	Grove standard wire
GND	GND	black
VCC	3V3	red
SDA	GP6	white
SCL	GP7	yellow

Raspberry Pi Pico in our 3D printed housing with Grove LCD and BME280 pressure sensor

### Setup software

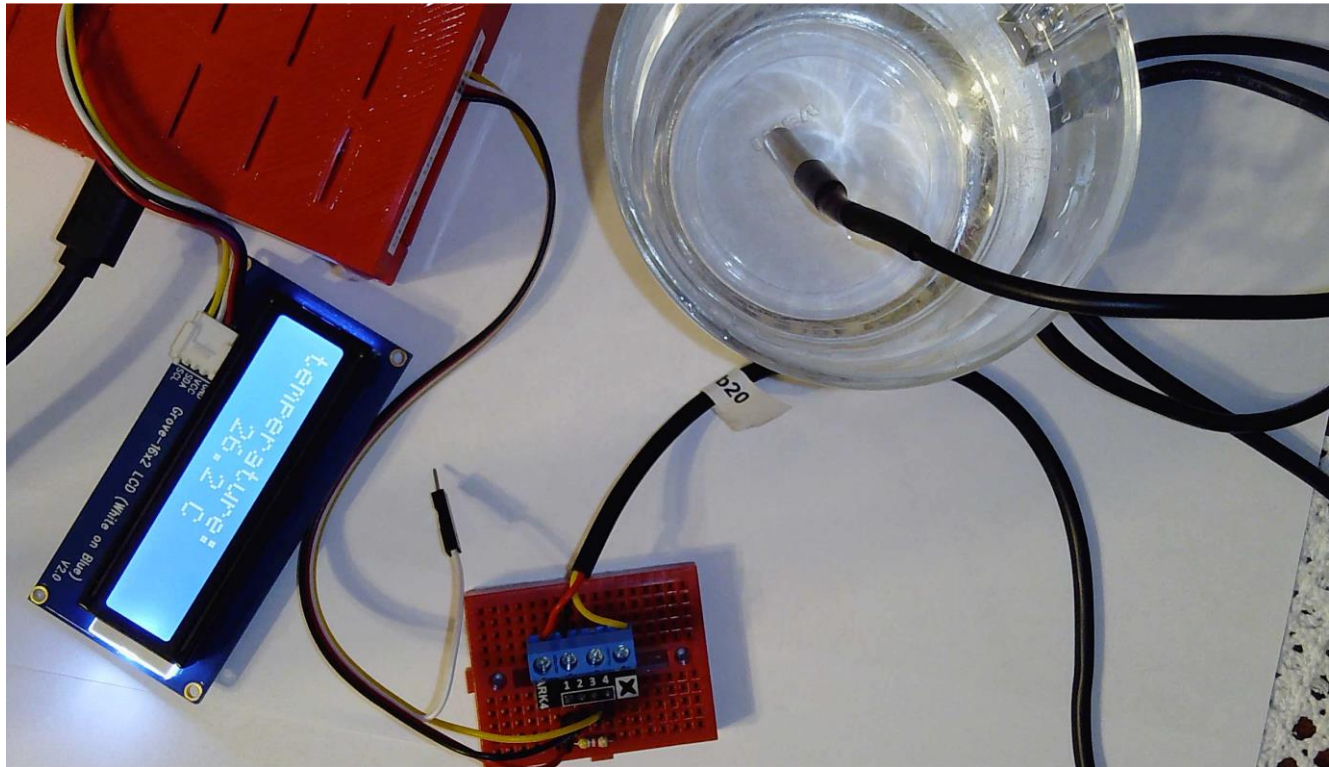
Below Python code for saving in microcontroller using Thonny App. Write the script in Thonny, then select "File/Save" and select Raspberry Pi Pico as destination. Remember please to add .py at the end of filename. Our proposal for filename: **ds18b20termdigLCD.py**

```
1 from machine import Pin, I2C
2 import onewire, ds18x20
3 from utime import sleep, sleep_ms
4 from lcd1602 import LCD1602
5
6 #signal pin connected to GP5
7 ds_pin = Pin(5,Pin.PULL_UP)
8
9 ds_sensor = ds18x20.DS18X20(owewire.OneWire(ds_pin))
10 roms = ds_sensor.scan()
11 print('Found DS devices: ', roms)
12 #I2C Bus initialisation
13 i2cbus = I2C(1,scl=Pin(7), sda=Pin(6), freq=400000)
14 #LCD initialisation
15 d = LCD1602(i2cbus, 2, 16)
16 d.display()
17 d.setCursor(0,0)
18 d.print("Initialisation")
19
20 while True:
21     ds_sensor.convert_temp()
22     sleep_ms(750)
23     for rom in roms:
24         d.clear()
25         d.setCursor(0,0)
26         d.print("temperature:")
27         #print rom temperature
28         print(round(ds_sensor.read_temp(rom),1))
29         d.setCursor(5,1)
30         d.print(str(round(ds_sensor.read_temp(rom),1))+ " C")
31         sleep(5)
```

Below You can find simple version of the code, which can be used with Thonny Plotter. Will be good to start with this simple version to check if everything works well.

```
1 from machine import Pin
2 import onewire, ds18x20
3 from utime import sleep, sleep_ms
4 #signal pin connected to GP5
5 ds_pin = Pin(5,Pin.PULL_UP)
6
7 ds_sensor = ds18x20.DS18X20(owewire.OneWire(ds_pin))
8 roms = ds_sensor.scan()
9 print('Found DS devices: ', roms)
10 while True:
11     ds_sensor.convert_temp()
12     sleep_ms(750)
13     for rom in roms:
14         #print rom temperature
15         print(round(ds_sensor.read_temp(rom),1))
16         sleep(5)
```

If you want have autonomous system then save the script on the Pico with special name: **main.py**. This script will start automatically (No need to connect to a computer). In this case you can power the Pico from powerbank.



## Experiment Evaluation

After checking the correct operation of the constructed system, you can proceed to the next tests. Testing electric kettles. **Perform the experiment under the supervision of the teacher. You have to be sure that it is safe to use your thermometer kettle.**

Write down your results:

parameters	pressure	temperature	humidity
initial temperature	.....	initial temperature	.....
final temperature	100 C	final temperature	100 C
temperature difference	.....	temperature difference	.....



time to boil	.....	time to boil	.....
mass of water in the kettle	500 g	mass of water in the kettle	1000 g
water boiling time	.....	water boiling time	.....
kettle power (W)	.....	kettle power (W)	.....
efficiency (%)	.....	efficiency (%)	.....

Calculation of the heat energy needed to boil water based on the temperature and weight of the water (use the specific heat of water)

.....  
.....

Calculating the real energy needed to boil water based on the boiling time and electric power of the kettle

.....  
.....

## Go further

This sensor can be used in many situations, where waterproof is required.

- Measure temperature air, ice water, tea
- Test the rate of changes in the temperature of the liquid, apply thermal insulation and observe the difference in cooling time
- Use a glass of 0.25 liter water placed outside the house to record changes in temperature throughout the day. You can calculate the energy changes between morning (7:00 am) and 2:00 pm. You can use our datalogger example for this
- Measure the temperature of the tea in the glass. Verify the result using a thermal camera
- The use of the presented measuring system can be found in the experiment prepared by the Portuguese school: "Joule experiment"

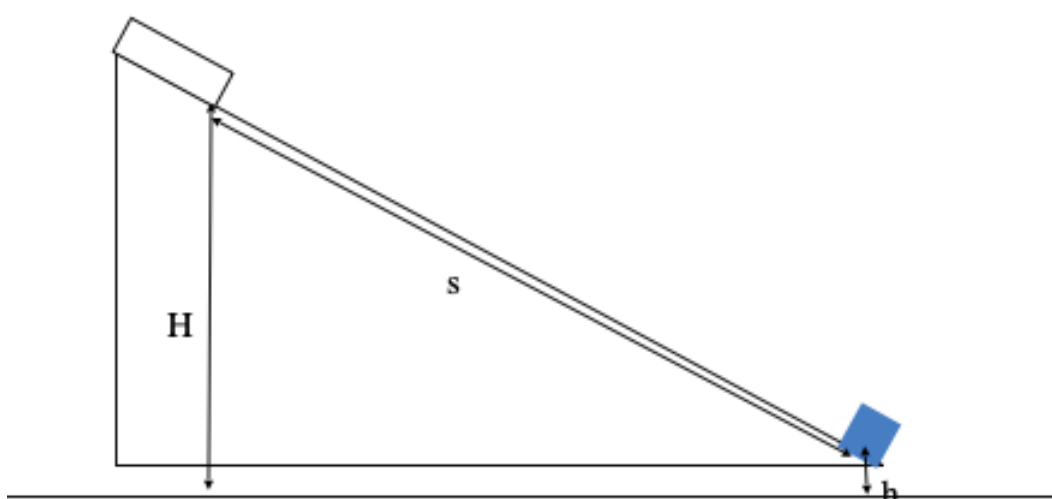


Topic: Energy	Age	Country	Date
Energy changes on an inclined plane	>14	Poland	October 2021

## Function, realisation

Observations of energy changes during the block's movement on an inclined plane

Follow the instructions as described below



1. Place the inclined plane at an angle to the horizontal (as in the picture above)
2. place the photo gate at the end of the slope
3. Make measurements of the height of H, h and the length of S and write them down in the attached table
4. Activate the speed measurement on photo gates and release the



cuboid from the top, equal to the height H

5. Record the speed measurement results in table 1
6. Perform the action in point 4 four more times for the same height and table the results
7. Calculate the average speed from the obtained average speed  $V_{avg}$
8. Using the appropriate formulas and dependencies, calculate the potential energy  $E_p$  at the height H and the kinetic energy  $E_k$  at the bottom of the inclined plane.

	mass	H	h	s		
$V_1$						
$V_2$						
$V_3$	$V_s$					
$V_4$						
$V_5$						

Calculate the value of the potential energy at the top of the slope:

$$E_p = mg(H-h)$$

Take the value  $g = 9,81 \text{ m/s}^2$

Calculate the value of the kinetic energy of the block at the bottom of the inclined plane:

$$E_k = \frac{mv^2}{2}$$

Find the value of the friction force T.



Ideally, when friction does not exist, the law of conservation of mechanical energy in motion can be applied. This means that the total energy  $E_t$ , i.e. the sum of the kinetic energy  $E_k$  and potential  $E_p$ , does not change during the movement. During the movement of the block, the potential energy  $E_p$  turns into kinetic energy  $E_k$ .

At the time of the start, the total energy is equal to the potential energy  $E_t = E_p = mgh$ ,

At the end of the tier, all potential energy is converted into kinetic energy:

$$E_t = E_k = 0.5 mV^2$$

Ideally, we can write:

$$mgh = 0.5 mV^2$$

In the case of a frictional motion, the frictional force does the work and the energy is lost, therefore the factor  $W = Ts$  must appear in the energy balance, where  $s$  - the path on which the constant friction force  $T$  acts.

At the top of the inclined plane we have:  $E_t = E_p = mgh$

However, at the end of the equation  $W + E_k = Ts + 0.5mV^2$ .

Therefore, in this case, the energy balance has the form:  $Ts + 0.5mV^2 = mgh$ .

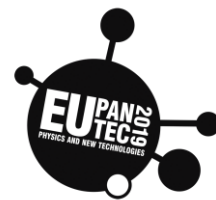
### Hardware required

- 2x Vernier Photogates





- Labquest mini interface  
(microcontroller)
- (instead of above you can use 2 photo gates with IR sensor and Pico microcontroller with MicroPython)
- Pc or Mac



## **Materials required**

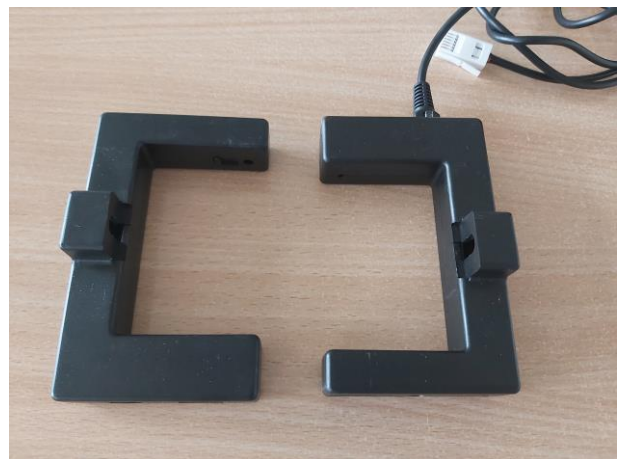
- inclined plane
- block
- cart

## **Software required**

- Vernier Logger Pro or Graphical Analysis (or Thony App if you use Pico)

## **Experiment Evaluation**

System testing:





## Go further

- determination of the friction coefficient
- performing an experiment with a different slope angle



Topic: Energy			
Energy changes in the harmonic motion of mass on a spring (determination of the spring constant)	Age  >14	Country  Poland	Date  October 2021

### Function, realisation

To calculate the potential spring energy, it is necessary to measure the spring constant  $k$ . Hooke's law states that the force of the spring is proportional to its elongation from equilibrium, or  $F = -k\Delta x$ . A known force may be applied to balance the spring force by suspending a series of weights from the spring.

For example: 100g, 150g, 200g, 250g and 300g determination of the force of gravity in N and comparison with the



elastic force by registering the elongation of the spring  $\Delta x$

**Follow the instructions as described below**

1. Install the spring and mark the end of the spring as  $x = 0$ , hang a known weight (Figure 1). and measure the elongation  $\Delta x$ . Record the results in **table 1**

2. Determine the spring constant  $k$  comparing the force of gravity with the force of elasticity:

$$mg = k\Delta x$$

$$k = mg / \Delta x$$

3. Connect the motion detector and place it directly under the hanging weight. Protect the motion detector by placing e.g. a wire basket over the detector. The mass should be approximately 30 cm above the detector when it is at rest.



4. Start by stretching the spring so that the weight moves up and down, vibrating about 10 - 15 cm between the minimum and maximum positions,

Be careful that the weight does not swing sideways

5. Record the data on the weight's position and speed.

**Table 1**

mass [kg]	weight Q [N]	elongation $\Delta x$ [m]	spring constant k [N/m]

The mean value of the constant  $k = \dots\dots\dots$





## Hardware required

- Vernier Go motion- motion detector
- Vernier Labquest mininterface (microcontroller) (instead of above you can Pico microcontroller with MicroPython and HCSR04P ultrasonic sensor)
- Pc or Mac

## Materials required

- springs
- mass

## Software required

- Vernier Logger Pro or Graphical Analysis (or Thony App if you use Pico)

## Theory:

Energy is present in three forms for the mass of the sinker and the spring. A mass  $m$ , moving at a speed  $v$ , may have the kinetic energy  $E_k$

$$E_k = 1 / 2mv^2$$

The spring can store elastic potential energy  $E_{ps}$  or gravity potential energy  $E_p$ . The elastic potential energy  $E_{ps}$  can be calculated from the formula:

$E_{ps} = 1 / 2k \Delta x^2$ , where  $k$  is the spring constant and  $\Delta x$  is the elongation or compression of the spring measured from the equilibrium position.



The weight-spring system also has gravitational potential energy ( $E_p = mg\Delta x$ ), but we do not need to account for this energy if we measure the length of the spring from an equilibrium position. Then we can focus on the energy exchange between the kinetic energy and the elastic potential energy

If the system does not experience other forces, the principle of conservation of energy tells us that sum

$E_k + E_p = 0$ , which we can test experimentally.

## Experiment Evaluation

System testing:





Topic: Energy			
Introduction to the use of the Pico microcontroller in experiments. LED control. Thonny App.	Age >12	Country Poland	Date October 2021

## Function, realisation

Introducing the use of a microcontroller to perform experiments. Getting to know the Thonny environment, controlling the pin states of the microcontroller on the example of LED

## Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18 or above
- Cytron Maker Pico docking station for Pico
- PC or Mac computer.

## Materials required

- 3xLED module (Dfrobot) with different colour
- 3xGrove male wire
- paper for note

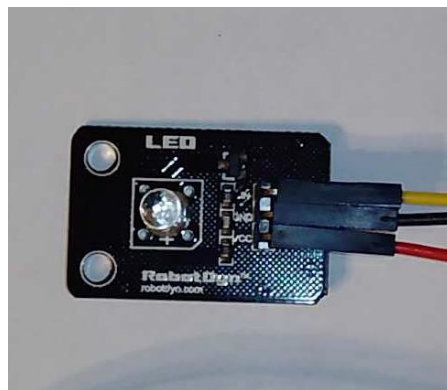
## Software required

- ThonnyApp (thonny.org),
- this example don't need custom libraries

## Setup Hardware

LED modules have 3 Pins: VCC, GND and OUT. Connect each LED according to the table below.

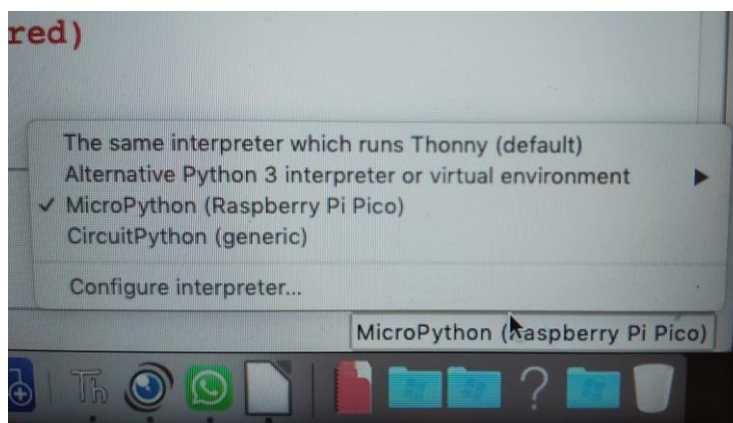
LED module	Raspberry Pi Pico pins	Grove male wire
VCC	3V3	red
GND	GND	black
Out (Signal)	GP3/GP5/GP9 (for 3 led modules)	yellow
not connected		white



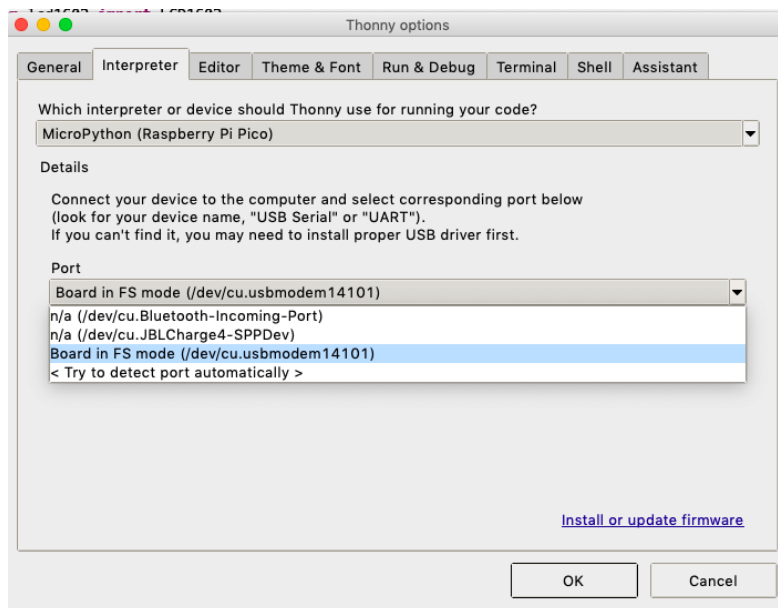
## Setup software

In the lower right corner of the application window, select the Micropython Raspberry Pi Pico interpreter. In this case the code may be run by Pico.

Sometimes the Pico board port will not be recognized automatically and you will have to manually point the interpreter and USB port. Select Menu "Tools/Options"



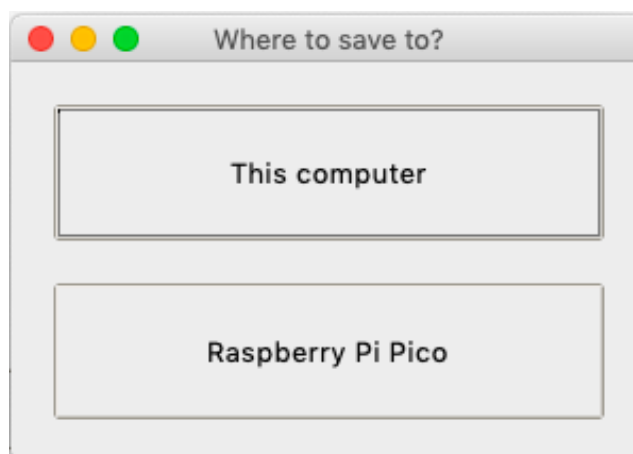
You can also use this screen to install the



### MicroPython on the Pico board

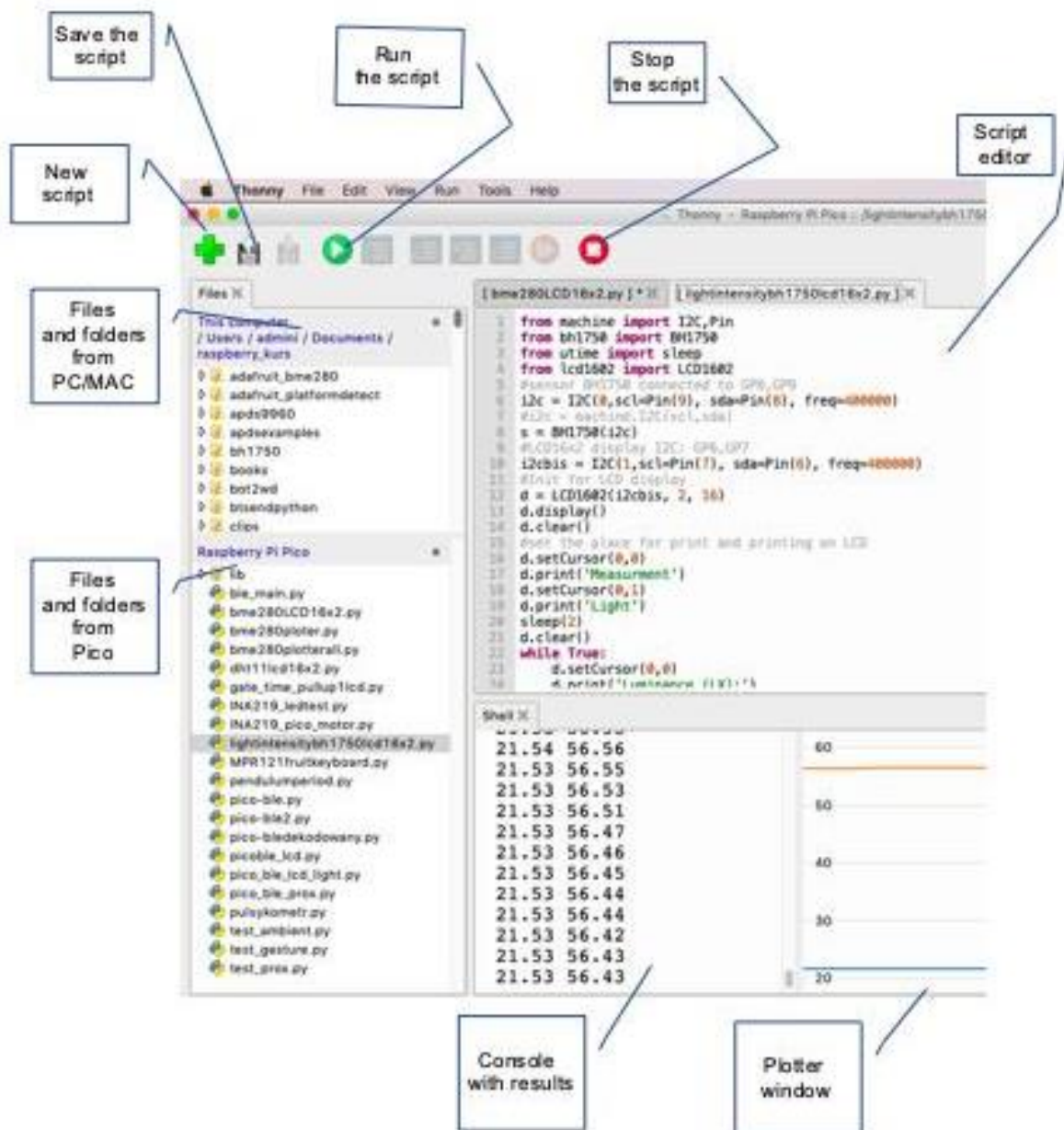
In this manual you can find simple Python code for saving in microcontroller using Thonny App. Write the script in Thonny editor, then select "File/Save" and select Raspberry Pi Pico as destination. Remember please to add .py at the end of filename. Our proposal for filename: LEDon.py.

After selecting the save option from the menu, you will be asked whether to save the script to local disk or Pico memory.



It is recommended to select "View / files" from the menu to see the scripts saved on the local disk and in the Pico memory.

An explanation of the Thonny application interface elements:



The digital pins of the microcontroller can be high (1) or low (0).  
The led1.value (1) instruction sets the state of the pin to which the LED is connected to high.

Python code example to control 4 LEDs



```
1 from machine import Pin
2 from utime import sleep
3 # Pin numbers depend on connection with Cytron docking station
4 # for Cytron may be GP3,GP5,GP9
5 #onboard led internal connected to GP25
6 #grove wire connect to LED module (yellow to signal)
7 LED1= Pin(3,Pin.OUT)
8 LED2= Pin(5,Pin.OUT)
9 LED3= Pin(9,Pin.OUT)
10 LED0= Pin(25,Pin.OUT)
11 while True:
12     #switch off all leds for 1 second
13     LED1.value(0)
14     LED2.value(0)
15     LED3.value(0)
16     LED0.value(0)
17     sleep(1)
18     #switch on all leds for 3 seconds
19     LED1.value(1)
20     LED2.value(1)
21     LED3.value(1)
22     LED0.value(1)
23     sleep(3)
24     #Switch off LED3 and internal led (GP25)
25     LED3.value(0)
26     LED0.value(0)
27     sleep(4)
28     #Switch off: LED1, LED2 and switch on: LED3 for 3 seconds
29     LED1.value(0)
30     LED2.value(0)
31     LED3.value(1)
32     LED0.value(1)
33     sleep(3)
```

## Example Code explanation

First two lines of the script:

```
from machine import Pin
from utime import sleep
```

mean the use of modules and functions defined in them





The **machine** module contains specific functions related to the hardware on a particular board.

The **utime** module (line 2) provides time functions. In this case we use **sleep** method for delay.

The following lines define the way of connecting the LED to the corresponding pins of the microcontroller (GP3, GP5, GP9 respectively).

GP25 is internally connected with Pico onboard LED.

```
LED1= Pin(3,Pin.OUT)
```

```
LED2= Pin(5,Pin.OUT)
```

```
LED3= Pin(9,Pin.OUT)
```

```
LED0= Pin(25,Pin.OUT)
```

The following line:

```
while True:
```

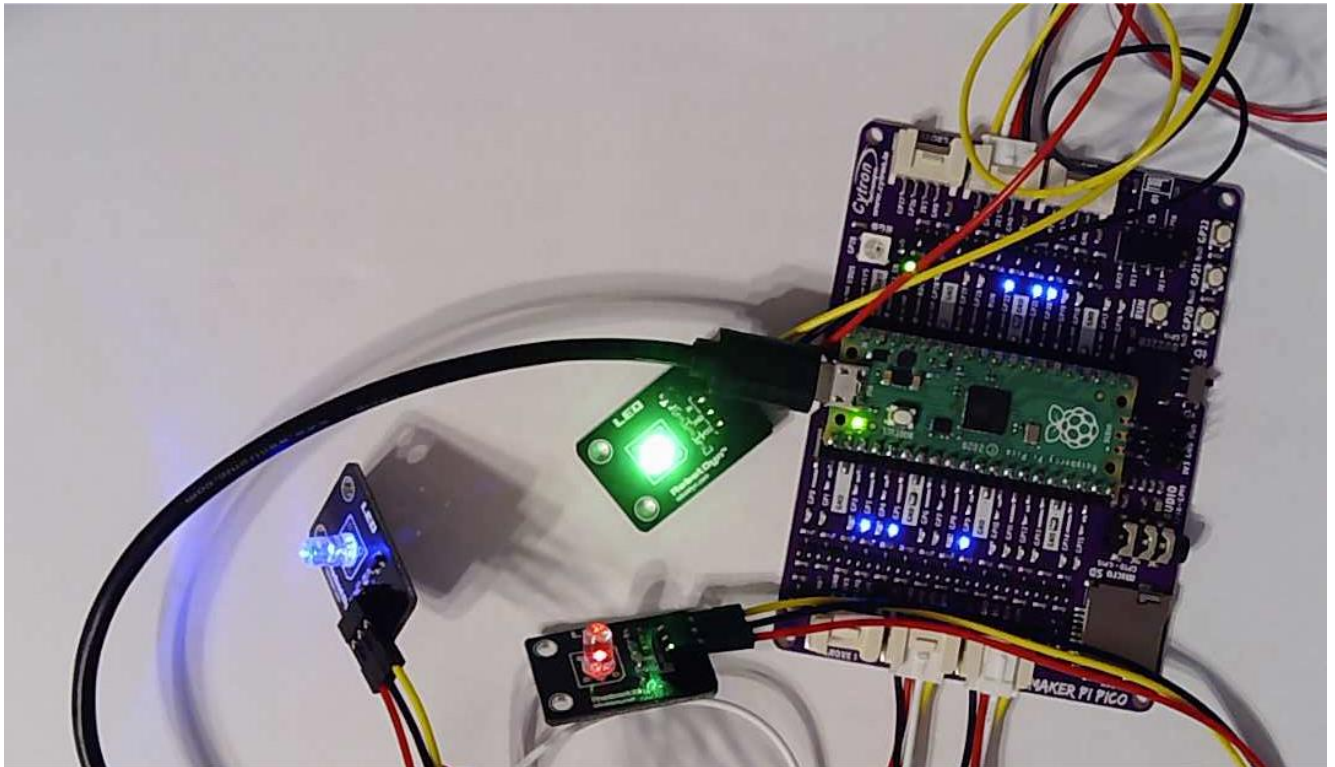
means infinite loop. The instructions in the loop will be repeated indefinitely. Pay attention to the indents. In the case of Python, they are very important.

Lines starting with # are comments.

Likewise, """remark""" text is a multi-line comment.

After writing the code inside editor window, we can run the code using the appropriate icon:





## System operation effect

### Go further

- Change the code and design a traffic light system at the crossroads.
- Use Pin 25 and onboard LED to have the microcontroller inform you about its condition in your projects.



Topic: All			
Resistance and voltage measurement (rotary analog resistor). Reading from analog sensor.	Age  >14	Country  Poland	Date  October 2021

## Function, realisation

Voltage and resistance measurement using a rotary potentiometer. Interpreting the measurement results from the analog sensor (ADC-Analog to Digital Converter).

## Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18 or above
- Cytron Maker Pico docking station for Pico
- LCD16x2 Grove I2C 3V3
- PC or Mac computer.

## Materials required

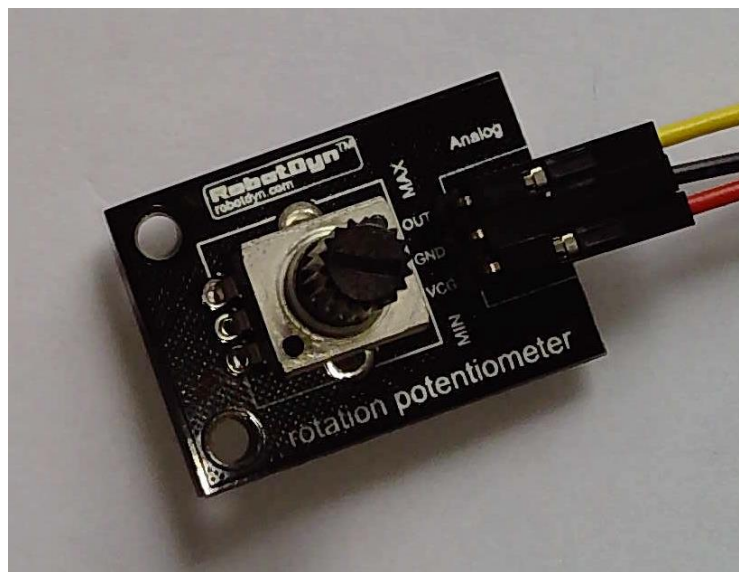
- 1xGrove standard wire (2xGrove grove plug)
- 1xGrove male wire (1xGrove plug, 1xDuPont )
- 10 kΩ rotary potentiometer from Robotdyn
- paper for note

## Software required

- ThonnyApp (thonny.org),
- simple version script does not require custom libraries
- LCD version require lcd1602 library  
([https://files.seeedstudio.com/wiki/Grove\\_Shield\\_for\\_Pi\\_Pico\\_V1.0/Libraries.rar](https://files.seeedstudio.com/wiki/Grove_Shield_for_Pi_Pico_V1.0/Libraries.rar))

## Setup Hardware

10 kΩ rotary potentiometer have 3 Pins: VCC, GND and OUT. Connect potentiometer pins according to the table below:



Potentiometer	Raspberry Pi Pico pins	Grove male wire
VCC	3V3	red
GND	GND	black
Out (Signal)	GP27	yellow
not connected	GP26	white

### LCD screen connection

LCD 1602 pins	Raspberry Pi Pico pins	Grove standard wire
GND	GND	black
VCC	3V3	red

SDA	GP6	white
SCL	GP7	yellow

## Explanation of how the potentiometer works

When you rotate the knob, the sliding wiper connected to the knob slides on the resistive material with the rotation. With the sliding, the length of resistive material between  $U_2$ –  $U_{wyj}$  and  $U_{wyj}$ –  $U_1$  changes, which causes the resistance value and thus the output voltage to change accordingly.

Reading an analogue input is almost identical to reading a digital input, except for one thing: when reading a digital input you use `read()`, but this analogue input is read with `read_u16()`. That last part, `u16`, simply warns you that rather than receiving a binary 0 or 1 result, you'll receive an unsigned 16-bit integer – a whole number between 0 and 65,535. If you can't get your Pico's analogue input to read exactly zero or exactly 65,535, that is correct. All electronic elements are built with a tolerance. In the case of the potentiometer, it will likely never reach exactly 0 or 100 percent of its input – but it will get you very close.

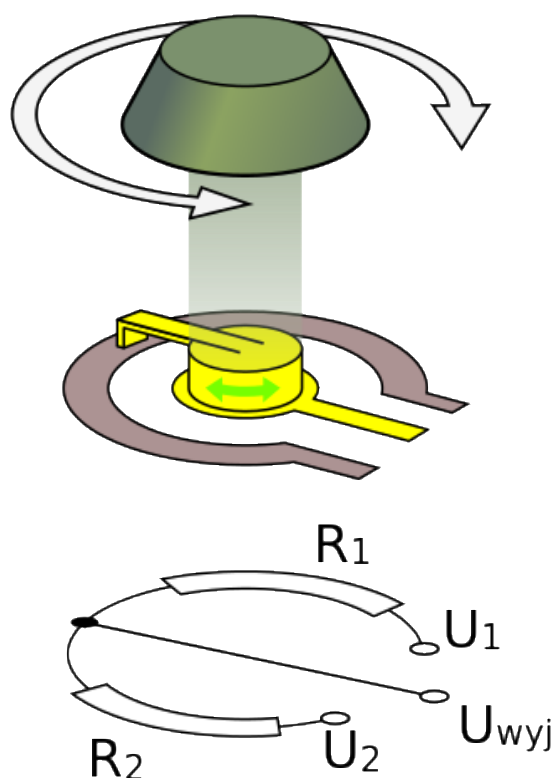


image source: pl.wikipedia.org



## Setup software

Simple version (printing in console) of the script. Write the code inside Thonny App editor, save and run.

```
1 from machine import ADC
2 from utime import sleep
3 #ADC0, GP26
4 #ADC1, signal from rotary resistor connected to GP27
5 potentiometer = ADC(27)
6 conversion_factor = 3.3 / (65535)
7 #Micropython ADC: 65536 steps from 0 to 65535, support 16 bit ADC
8 #Real ADC steps scale in Pico: 0-4095. Pico has only 12-bit ADC (levels: 0-4095)
9 #4095 correspond to 65535
10 print("RAW ADC reading, voltage: ")
11 while True:
12     raw=potentiometer.read_u16()
13     voltage =raw * conversion_factor
14     print(raw,"\t",voltage)
15     sleep(2)
```

## Version with LCD support

```
1 from lcd1602 import LCD1602
2 from machine import I2C,Pin,ADC
3 from utime import sleep
4 #ADC0, GP26
5 #ADC1, signal from rotary resistor connected to GP27
6 potentiometer = machine.ADC(27)
7 conversion_factor = 3.3 / (65535)
8 #Micropython ADC: 65536 steps from 0 to 65535, support 16 bit ADC
9 #Real ADC steps scale in Pico: 0-4095. Pico has only 12-bit ADC (levels: 0-4095)
10 #4095 correspond to 65535
11
12 #I2Cbus and LCD initialisation. LCD connected to GP6,GP7
13 i2c = I2C(1,scl=Pin(7),sda=Pin(6),freq=400000)
14 d = LCD1602(i2c,2,16)
15 d.display()
16 print("RAW ADC reading, voltage: ")
17 while True:
18     raw=potentiometer.read_u16()
19     voltage =raw * conversion_factor
20     print(raw,"\t",voltage)
21     d.setCursor(0,0)
22     d.print('Raw ADC:'+str(raw))
23     d.setCursor(0,1)
24     d.print("Volts:"+str(round(voltage,3)))
25     sleep(2)
26     d.clear()
```

## Example Code explanation

Reading voltage from analog Pin:



```
raw=potentiometer.read_u16()
```

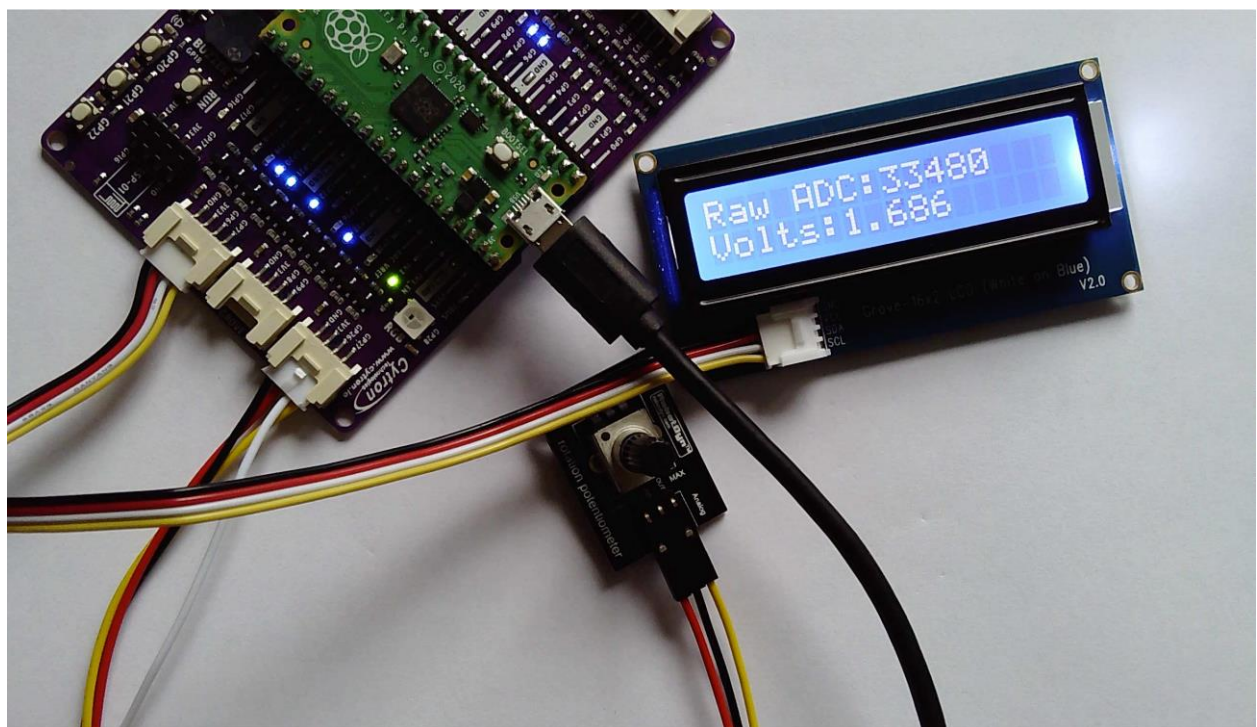
The number we see corresponds to the ADC level, in the range 0..65535. 65,535 means 'full voltage-3.3V', this is raw reading from ADC. We can easily change this to print the voltage corresponding to this level. We use conversion factor inside the script:

```
conversion_factor = 3.3 / (65535)
```

Then we can make calculations for voltage:

```
voltage =raw * conversion_factor
```

## System operation effect



## Testing

Slowly turn the potentiometer knob. Observe the measurement results.

Note the results for the position of the knob:

- turned all the way to the right,
- turned all the way to the left,





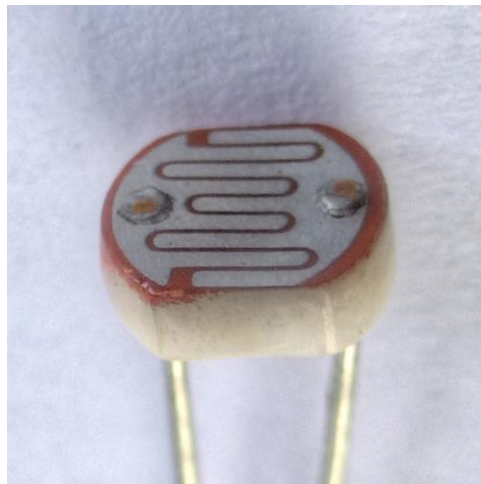
Try to reach voltage of 1.65V. What ADC level corresponds to 1.65V?

### Go further

- instead of a potentiometer, use a photoresistor that changes the resistance depending on the light
- use a rain sensor instead of a potentiometer, the resistance of which changes as rain increases
- use an analog water quality sensor (TDS) instead of a resistor
- use the experience gained for other analog sensors. Compare the characteristics of the sensors and, on this basis, adjust the formula that calculates the sensor indications in the appropriate units

In the picture  
photoresistor:

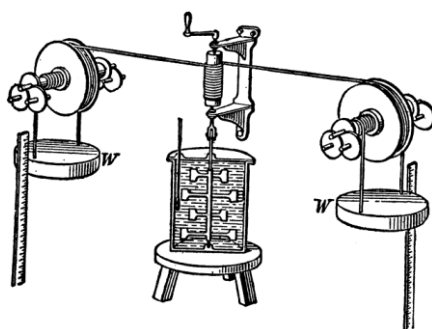
below you will find a





Topic	Age	Country	Date
Joule experiment revisited in modern ways	>14	Portugal	Oct 2021

*Does agitating the water increase or decrease its temperature?*



Joule's experiments, that you have probably heard of, proves that temperature is related to the average of the kinetic energy of the particles composing the body. Let's check!

----

First, we must get the **temperature sensor working** (this is the technological part of the experiment). Use instructions in attachment if necessary.

-----

Then, use the hand blender to agitate the water (very little water, just a bit in the bottom) and measure the temperature during agitation. **Be sure the sensor is next to and not under the fans of the hand blender.**

Wait until water temperature is stable and the scale adjusts to a small interval (maybe from 19°C to 20°C). Turn on the hand blender. Observe and register the results.

#### Exploration:

- 1- Does agitating the water increase or decrease temperature?
- 2- If we can warm up water both with heat and work, what is the connection between them?
- 3- If agitating the water increases the temperature, why does a very hot cup of tea cool off when we stir it?

-----

#### Extra Task:

*Can you measure the efficiency of this heating process?*

(Tip: use a chronometer to measure the time used and the power of the apparatus to establish the energy used in the process and  $Q = m \cdot c \cdot \Delta T$  to measure the energy the water has received.  $c_{\text{water}} = 4,18 \times 10^3 \text{ J} \cdot \text{kg}^{-1} \cdot ^\circ\text{C}^{-1}$ )



Poitiers

Lycee Pilote  
Innovant International

# PHYSICS AND ENVIRONMENT



Topic Environment and society	Age	Country	Date
Measurment of light intensity.	>14	Poland	v.1 March 2020 v.2 October 2021

*\*The original version of the experiment (v.1) was developed using the Arduino IDE. With the advent of the Raspberry Pi Pico microcontroller and its great possibilities, the description of the experiment has been adapted to MicroPython (v.2), which will make it much easier for students to perform it.*

## Function, realisation

Construction of a measuring instrument based on microcontroller and Python that measures the light intensity.

## Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18
- Cytron Maker Pico docking station for Pico (or different docking station for Pico: Waveshare, Seeedstudio)
- BH1750 module (3.3V for Pico)
  - 3V3 LCD 1602 I2C display from Seeedstudio with Grove Socket,
- PC or Mac computer.

## Materials required

- 1 standard Grove wire, 1xGrove female wire
- micro usb 2.0 high speed cable(15cm or 30 cm)
- powerbank (for standalone version)
- paper for note

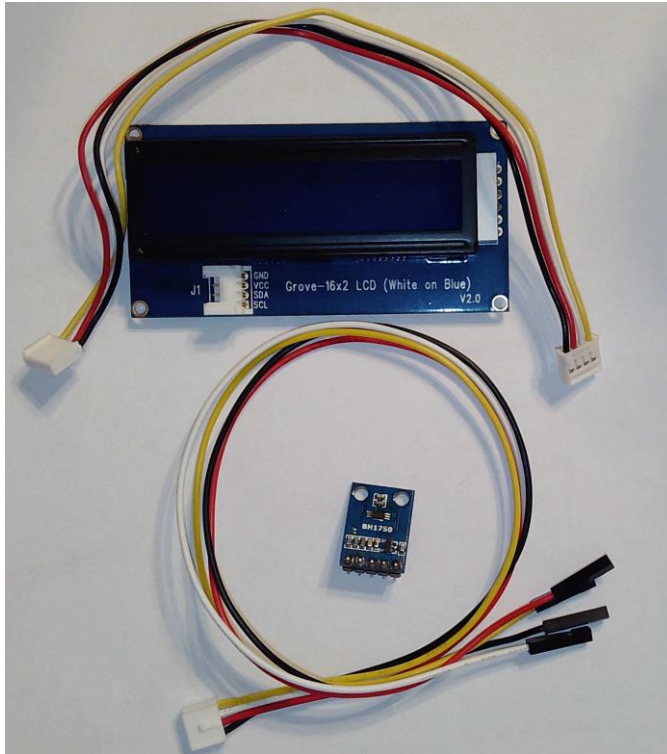
## Software required

- Thonny App (thonny.org)
- library for BH1750 sensor:  
<https://github.com/PinkInk/upylib/tree/master/bh1750/bh1750>
- library for LCD:  
[https://files.seeedstudio.com/wiki/Grove\\_Shield\\_for\\_Pi\\_Pico\\_V1.0/Libraries.rar](https://files.seeedstudio.com/wiki/Grove_Shield_for_Pi_Pico_V1.0/Libraries.rar)

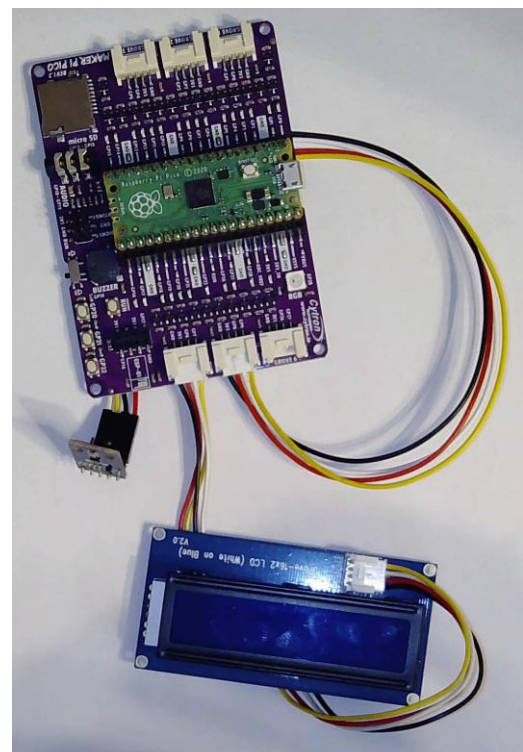
Using Thonny App You can save libraries inside lib folder in Pico memory.

## Setup Hardware

BH1750 module is a digital light sensor which uses the I2C interface. This allows it to be connected to the Raspberry Pi with only four wires. BH1750 is a Digital Ambient light sensor. It is easy to interface with a microcontroller, as it uses the I2C communication protocol. It consumes a



very low amount of current.



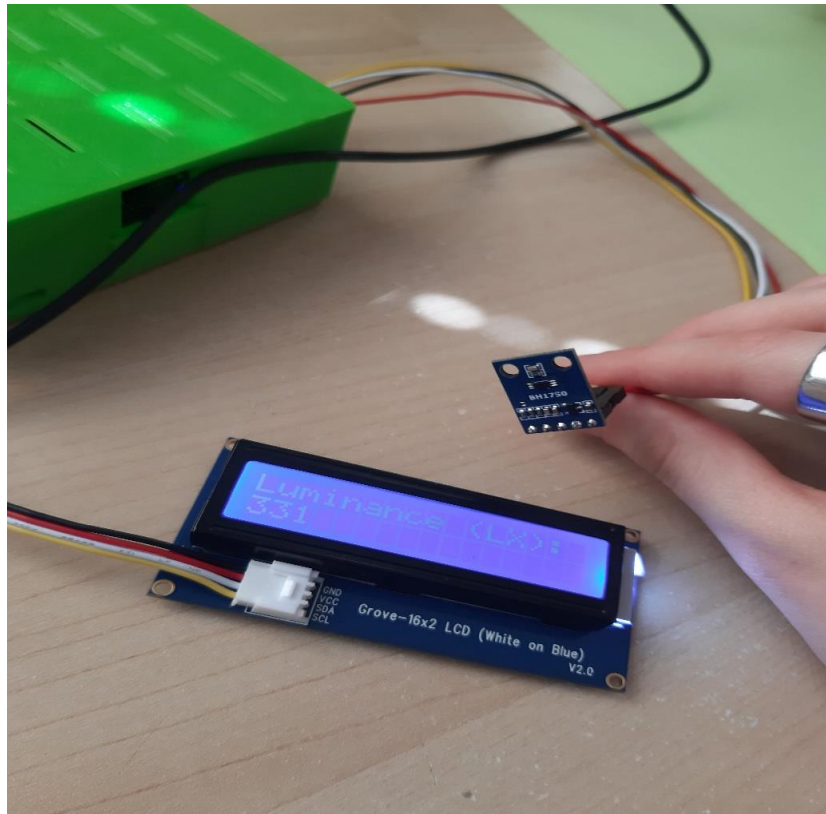


BH1750 sensor pins	Raspberry Pi Pico pins	Grove female wire
VCC	3V3	red
GND	GND	black
SCL	GP9	yellow
SDA	GP8	white

The Pico use 3.3V power (not 5!!!). We usually connect the red wire to pin 3V3 and black wire to the ground (GND). SDA and SCL are the pins used for I2C communication.

We suggest using the LCD screen to display the results. In this case, we can also build an autonomous, mobile system that can work without a computer. The use of grove cables makes the connection as easy as possible..

LCD 1602 pins	Raspberry Pi Pico pins	Grove standard wire
GND	GND	black
VCC	3V3	red
SDA	GP6	white
SCL	GP7	yellow



Raspberry Pi Pico in our 3D printed housing with Grove LCD and BH1750  
Light sensor

### Setup software

Below Python code for saving in microcontroller using Thonny App. Write the script in Thonny, then select "File/Save" and select Raspberry Pi Pico as destination. Remember please to add .py at the end of filename. Our proposal for filename: **bh1750lcd16x2.py**



```
1 from machine import I2C,Pin
2 from bh1750 import BH1750
3 from utime import sleep
4 from lcd1602 import LCD1602
5 #sensor BH1750 connected to GP8,GP9
6 i2c = I2C(0,scl=Pin(9), sda=Pin(8), freq=400000)
7 #i2c = machine.I2C(scl,sda)
8 s = BH1750(i2c)
9 #LCD16x2 display I2C: GP6,GP7
10 i2cbis = I2C(1,scl=Pin(7), sda=Pin(6), freq=400000)
11 #Init for LCD display
12 d = LCD1602(i2cbis, 2, 16)
13 d.display()
14 d.clear()
15 #set the place for print and printing on LCD
16 d.setCursor(0,0)
17 d.print('Measurment')
18 d.setCursor(0,1)
19 d.print('Light')
20 sleep(2)
21 d.clear()
22 while True:
23     d.setCursor(0,0)
24     d.print('Luminance (LX):')
25     d.setCursor(0,1)
26     d.print(str(int(s.luminance(BH1750.ONCE_HIRES_1))))
27     sleep(2)
28     d.clear()
```

Below You can find simple version of the code, which can be used with Thonny Plotter. Will be good to start with this simple version to check if everything works well.

```
1 from machine import I2C,Pin
2 from bh1750 import BH1750
3
4 #sensor BH1750 connected to GP8,GP9
5 i2c = I2C(0,scl=Pin(9), sda=Pin(8), freq=400000)
6
7 s = BH1750(i2c)
8
9 while True:
10     print(int(s.luminance(BH1750.ONCE_HIRES_1)))
11
```

If you want have autonomous system then save the script on the Pico as with special name: **main.py**. This script will start automatically (No need to connect to a computer). In this case you can power the Pico from Powerbank.

## The principle of operation of Digital Light Sensor

BH1750 Sensor board. When an ambient light is incident on the sensor the photo diode with a latency nearly equals to human eye absorbs the



light and produces an analog voltage signal which is further connected to Analog to Digital converter (ADC unit). Result signal is sent to Pico through I2C bus.

### Technical specifications of BH1750:

- Operating voltage: 3V to 5V
- Measures Lux range with a high resolution from 1 to 65535lx.
- Illuminance to Digital Converter
- The influence of infrared is very small
- Small measurement variation (+/- 20%)
- Feature to select 2 different I2C addresses.
- Low power consumption

### Definition of Luminous ,Lumen

**Illuminance** is a measure of how much luminous flux is spread over a given area. One can think of luminous flux (measured in lumens) as a measure of the total "amount" of visible light present, and the illuminance as a measure of the intensity of illumination on a surface

**Lumen** : The unit for the quantity of light flowing from a source in any one second (the luminous power, or luminous flux) is called the lumen. In our sensor we will take a reading from it in Lux which is equal to one lumen per square metre:

**Lux = 1 Lm/m<sup>2</sup> .**



## Experiment Evaluation

System testing:

- measure the light intensity in various situations: sunlight, torch, normal lightning, sensor being covered with something,
- find dependency between luminance and distance from light source,
- compare the obtained results with the results of the mobile application,
- change the delay (sleep) and observe the effect on the results,
- change the Python script to plot graph based on results (using internal ploter from Thonny App),
- you can use the built-in system to check if the lighting in the room is good enough according to the standards,
- you can check if the lighting in the desk you are working on is sufficient.

Write down your results:

situation	result (lux)	situation	result (lux)

This sensor is mainly used in LCD displays, TVs, Monitors and mainly in Mobile displays to automatically adjust the screen brightness according to the outdoor light conditions. This sensor can also used to Turn on/Off the lights.



Topic: Environment and society  Measurement of humidity and temperature with DHT11	Age  >14	Country  Poland	Date  v1 March 2020 v.2 October 2021
--	----------------	-----------------------	---

*\*The original version of the experiment (v.1) was developed using the Arduino IDE. With the advent of the Raspberry Pi Pico microcontroller and its great possibilities, the description of the experiment has been adapted to MicroPython (v.2), which will make it much easier for students to perform it.*

## Function, realisation

Build experimental system based on DHT11 sensor, which can measures air parameters: temperature and humidity.

The temperature and humidity DHT11 sensor is the basic device for measuring temperature and humidity. This sensor is used in many devices to measure their temperature. It is widely known and used all over the world.

## Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18 or above,
- Cytron Maker Pico docking station for Pico
- DHT11 sensor **with module** or DHT11 sensor from Seeedstudio with Grove socket. (Notice: on the market you can buy also sensor without module, but in this case you need resistor, breadboard and connection to Pico is not so easy.),
- PC or Mac computer,
- 3V3 LCD 1602 I2C display Seeedstudio with Grove socket.

## Materials required

2xGrove standard wire (or 1 Grove standard Wire and 1 Grove-Female wire, depend on sensor selection),  
micro usb 2.0 high speed (15cm or 30 cm)  
paper for note

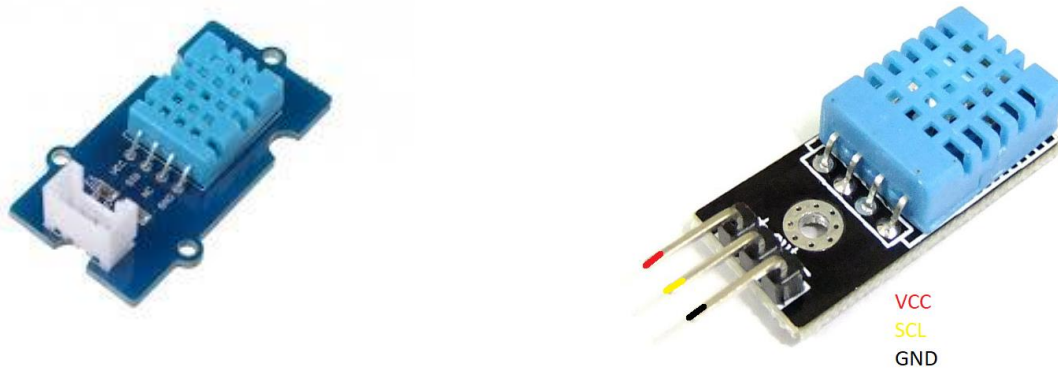
## Software required:

- Thonny App (thonny.org)
- library for DHT11 sensor is a standard Micropython library

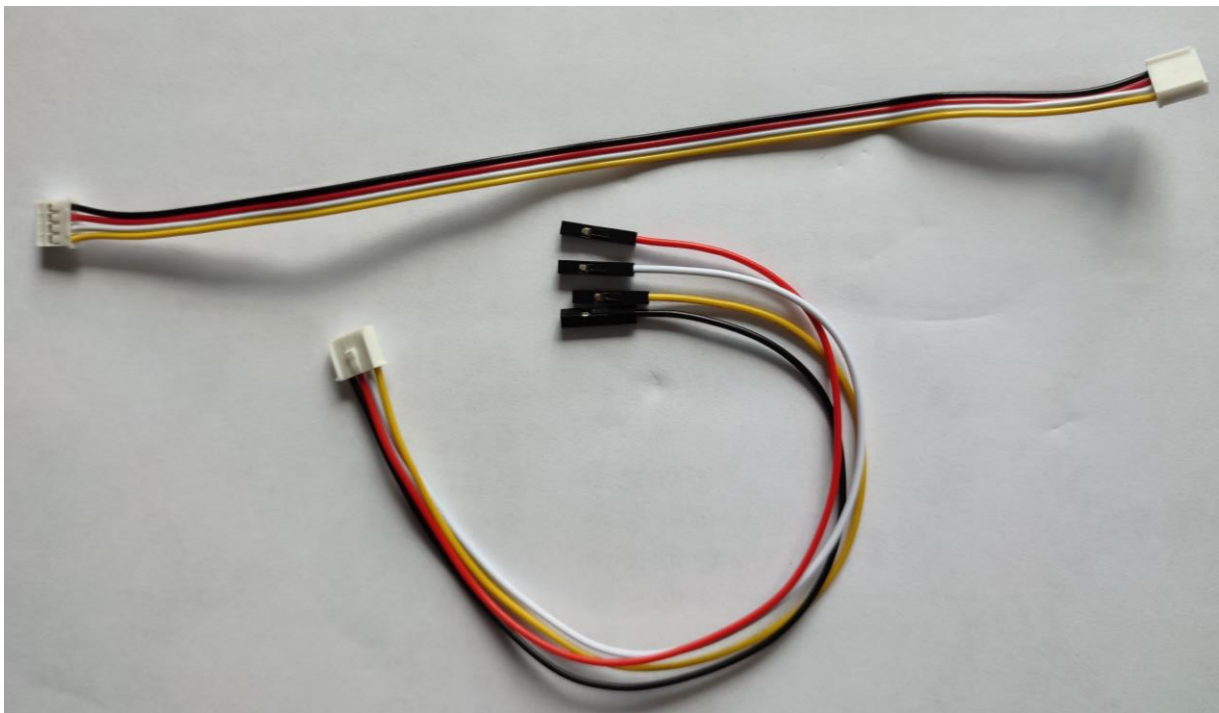


- library for LCD:

[https://files.seeedstudio.com/wiki/Grove\\_Shield\\_for\\_Pi\\_Pico\\_V1.0/Libraries.rar](https://files.seeedstudio.com/wiki/Grove_Shield_for_Pi_Pico_V1.0/Libraries.rar)



DHT11 sensor with Grove socket (on the left) and cheaper DHT11 with module on the right.



On above illustration on the top standard Grove Wire, on the bottom female Grove wire for sensor connection.



## Setup Hardware

Connections have been simplified thanks to the docking station and Grove cables. It is much harder to make a mistake or damage.

### DHT 11 Sensor connection:

DHT 11 pins	Raspberry Pi Pico pins	Grove wire
Signal	GP9	yellow
not connected	GP8	white
power (+)	3V3	red
ground (-)	GND	black

The pins can be changed according to your needs, but in this case you must change the code.

All functions of Raspberry Pi Pins, you can find on the website: [pico.pinout.xyz](https://pico.pinout.xyz).

### LCD I2C screen connection:

LCD Pins	Raspberry Pi Pico pins	Grove wire
SCL	GP7	yellow
SDA	GP6	white
VCC	3V3	red
GND	GND	black

The Pico uses a 3.3V power supply (not 5V !!!). Usually we connect the red wire to the 3V3 pin and the black wire to the ground (GND).

## Setup software



The script use special libraries, but only standard Micropython libraries, we are not allowed to install libraries.

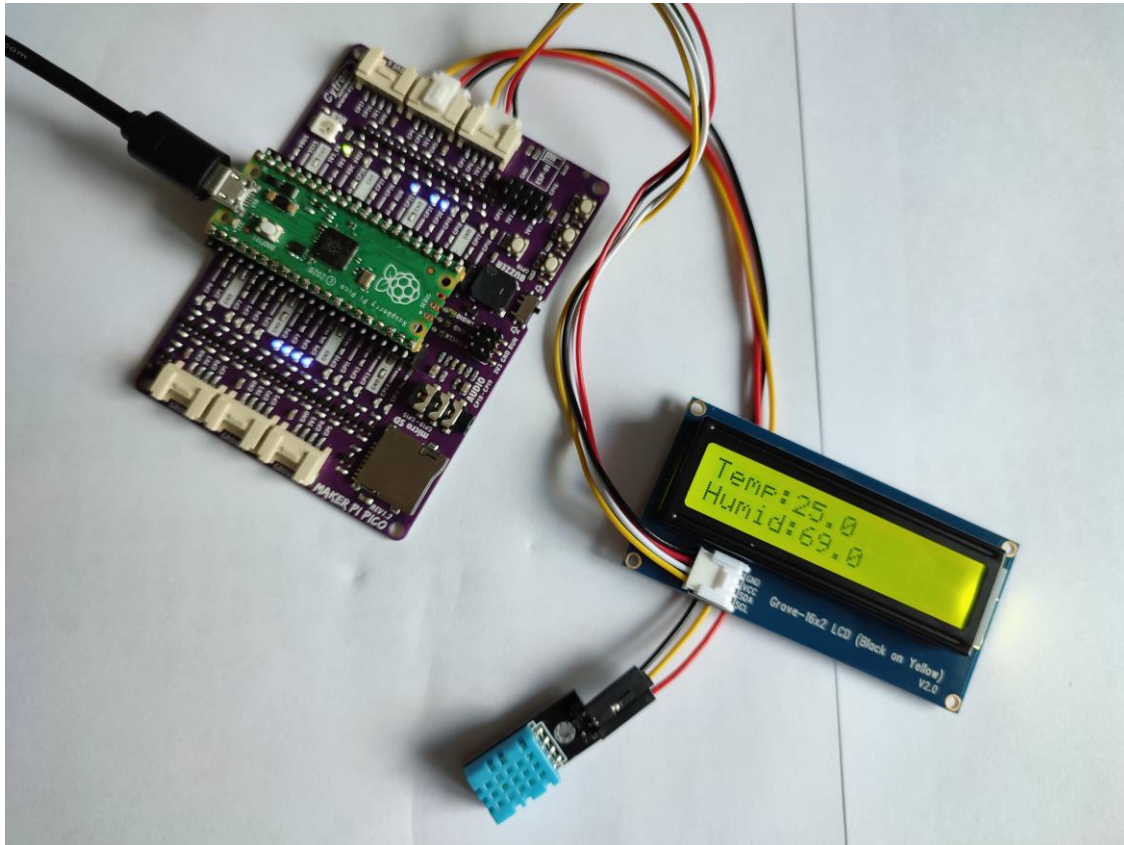
The script is very simple, very easy to understand, as are the scripts that read data from analog sensors.

```
1 from lcd1602 import LCD1602
2 from dht11 import *
3 from machine import I2C, Pin
4 from utime import sleep
5 # 1 LCD 16x2 display initialisation
6 i2c = I2C(1, scl=Pin(7), sda=Pin(6), freq=400000)
7 d = LCD1602(i2c, 2, 16)
8 d.display()
9 # 1 LCD display
10 # sensor connected to PIN 9
11 dht = DHT(9)
12
13 while True:
14     #reading from the sensor
15     temp, humid = dht.readTempHumid()#temp:  humid:
16     sleep(1)
17     d.clear()
18     # set the place on LCD for printing results
19     d.setCursor(0,0)
20     d.print("Temp:"+str(temp))
21     #the line below allows us to observe the results in the console and plot with Plotter
22     print(temp, humid)
23     d.setCursor(0,1)
24     d.print("Humid:"+str(humid))
25     sleep(1)
```

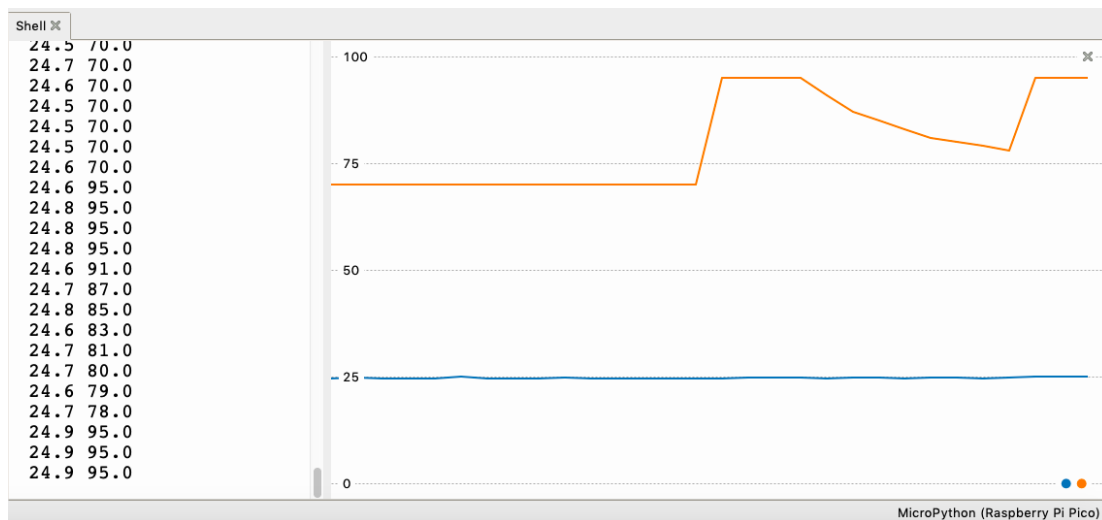
This script allows you to plot the measurement results with a plotter. To do this, select "View / Plotter" from the Thonny menu.

**Demonstration of the system operation:**





Terminal window and Plotter window with results of reading:



**Go Go further- ideas, testing:**

- test the system by finger touching
- place a heat source nearby.



## **The sensor is not waterproof !!!**

- You can use this simple example to build weather station
- You can build the system with logger and use outside
- design a system that sends the sensor result to a website or via bluetooth to a mobile application on your phone.



Topic: Environment and society	Age	Country	Date
Measurement of pressure	>14	Poland	v1 March 2020 v.2 October 2021

*\*The original version of the experiment (v.1) was developed using the Arduino IDE. With the advent of the Raspberry Pi Pico microcontroller and its great possibilities, the description of the experiment has been adapted to MicroPython (v.2), which will make it much easier for students to perform it.*

## Function, realisation

Construction of a measuring instrument based on microcontroller and Python that measures pressure (also temperature and humidity).

## Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18
- Cytron Maker Pico docking station for Pico
- BME280 module (3.3V for Pico)
  - 3V3 LCD 1602 I2C display from Seeedstudio with Grove Socket,
- PC or Mac computer.

## Materials required

- 1 standard Grove wire, 1xGrove female wire
- micro usb 2.0 high speed cable(15cm or 30 cm)
- powerbank (for standalone version)
- paper for note

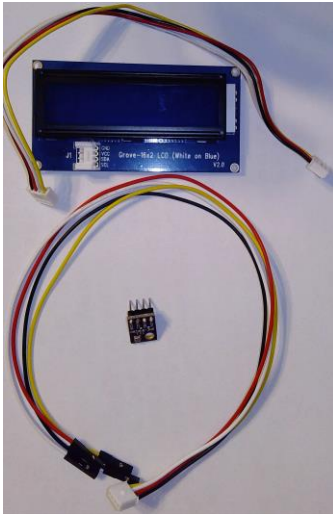
## Software required

- ThonnyApp (thonny.org)
- library for the BME280 sensor:  
[https://github.com/SebastianRoll/mpy\\_bme280\\_esp8266/blob/master/bme280.py](https://github.com/SebastianRoll/mpy_bme280_esp8266/blob/master/bme280.py), display of units has been removed from the library
- library for LCD:  
[https://files.seeedstudio.com/wiki/Grove\\_Shield\\_for\\_Pi\\_Pico\\_V1.0/Libraries.rar](https://files.seeedstudio.com/wiki/Grove_Shield_for_Pi_Pico_V1.0/Libraries.rar)

Use Thonny app to save libraries inside lib folder in Pico memory.

## Technical specifications of BME280:

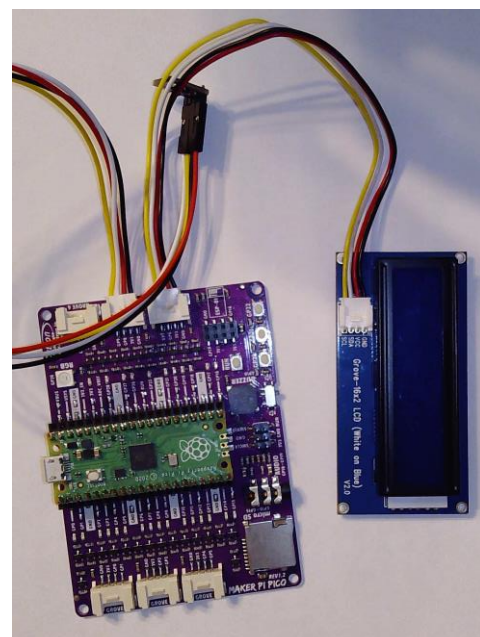
- Supply voltage: 3.3 V



- Temperature measurement range: -40 °C to 85 °C
- Accuracy:  $\pm 1$  °C
- Humidity measuring range: 10 to 100% RH
- Accuracy:  $\pm 3\%$  RH
- Pressure measuring range: 300 to 1100 hPa
- Accuracy:  $\pm 1$  hPa

### Setup Hardware:

Humidity, temperature and pressure sensor powered with the voltage of 3.3 V with I2C. It is easy to interface with a microcontroller. It consumes a very low amount of current.





BME280 sensor pins	Raspberry Pi Pico pins	Grove female wire
VCC	3V3	red
GND	GND	black
SCL	GP7	yellow
SDA	GP6	white

The Pico use 3.3V power (not 5!!!). We usually connect the red wire to pin 3V3 and black wire to the ground (GND). SDA and SCL are the pins used for I2C communication.

We suggest using the LCD screen to display the results. In this case, we can also build an autonomous, mobile system that can work without a computer. The use of grove cables makes the connection as easy as possible.

LCD 1602 pins	Raspberry Pi Pico pins	Grove standard wire
GND	GND	black
VCC	3V3	red
SDA	GP6	white
SCL	GP7	yellow

Raspberry Pi Pico in our 3D printed housing with Grove LCD and BME280 pressure sensor

## Setup software

Below Python code for saving in microcontroller using Thonny App. Write the script in Thonny, then select "File/Save" and select Raspberry Pi Pico as destination. Remember please to add .py at the end of filename. Our proposal for filename: **bme280lcd16x2.py**



```
1 from machine import Pin, I2C
2 from bme280 import BME280
3 import utime
4 from lcd1602 import LCD1602
5 #I2C Bus for bme280
6 i2c = I2C(0, scl=Pin(9), sda=Pin(8), freq=400000)
7 #sensor initialisation
8 bme = BME280(i2c=i2c, address=0x76)
9 #I2C Bus for LCD
10 i2cbis = I2C(1, scl=Pin(7), sda=Pin(6))
11 #LCD display initialisation
12 d = LCD1602(i2cbis, 2, 16)
13 d.display()
14
15 print("temp. C, pressure hPa,humidity %")
16 while True:
17     # clear LCD from previous results
18     d.clear()
19     #set the place for print and printing on LCD
20     d.setCursor(0,0)
21     d.print("T:"+str(bme.values[0]))
22     d.setCursor(0,1)
23     d.print("P:"+str(bme.values[1]) + " hPa")
24     d.setCursor(8,0)
25     d.print(str(bme.values[2]) + "%")
26     print(bme.values[0],bme.values[1],bme.values[2])
27     utime.sleep(5)
```

Below You can find simple version of the code, which can be used with Thonny Plotter. Will be good to start with this simple version to check if everything works well.

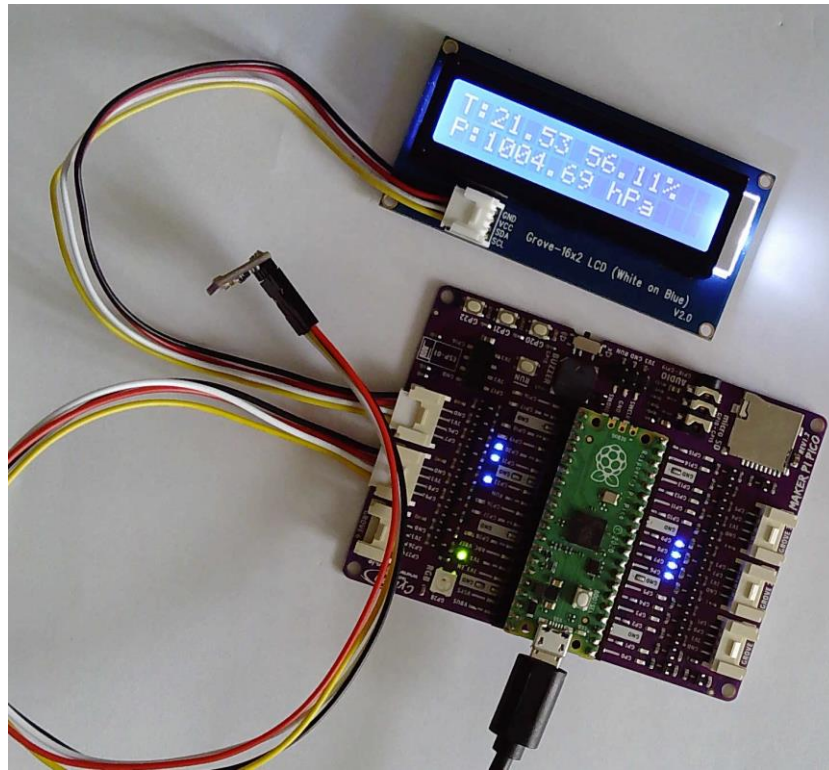
```
1 from machine import Pin, I2C
2 from bme280 import BME280
3 import utime
4 #I2C Bus initialisation
5 i2c = machine.I2C(0, scl=Pin(9), sda=Pin(8))
6 #sensor initialisation
7 bme = BME280(i2c=i2c, address=0x76)
8 print("temp. C, pressure hPa,humidity %")
9 while True:
10     print(bme.values[0],bme.values[1],bme.values[2])
11     utime.sleep(2)
```

If you want have autonomous system then save the script on the Pico with special name: **main.py**. This script will start automatically (No need to





connect to a computer). In this case you can



power the Pico from powerbank.

## Experiment Evaluation

System testing:

- measure pressure, humidity and temperature,
- test the system in vacuum,
- test the system in a mine or in a cave during the tour
- observe pressure and temperature changes as you hike to the mountains
- use a balloon to observe changes in temperature, pressure and humidity  
record changes in air parameters throughout the day (find our "data logger" instructions)
- change the scripts, to measure and display/plot only one parameter from the list: temperature, humidity, pressure

Write down your results:





situation	pressure	temperature	humidity

This sensor can be used in many situations. May be also the base for built weather station.



Topic: Environment and society <b>Measurement of soil moisture</b>	Age <b>&gt;12</b>	Country <b>Poland</b>	Date <b>v1 March 2020 v.2 October 2021</b>
---	----------------------	--------------------------	---

*\*The original version of the experiment (v.1) was developed using the Arduino IDE. With the advent of the Raspberry Pi Pico microcontroller and its great possibilities, the description of the experiment has been adapted to MicroPython (v.2), which will make it much easier for students to perform it.*

## Function, realisation

Build experimental system which measures humidity of soil and other things.

This project is of great importance due to the lack of water and the need for its rational use. The wide application of similar solutions in agriculture can reduce losses in plant production and lower costs.

## Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18 or above
- Cytron Maker Pico docking station for Pico
- Grove moisture sensor v1.4
- PC or Mac computer
- 

## Materials required

- 1 long grove wire
- micro usb 2.0 high speed (15cm or 30 cm)
- paper for note
- plant in a pot

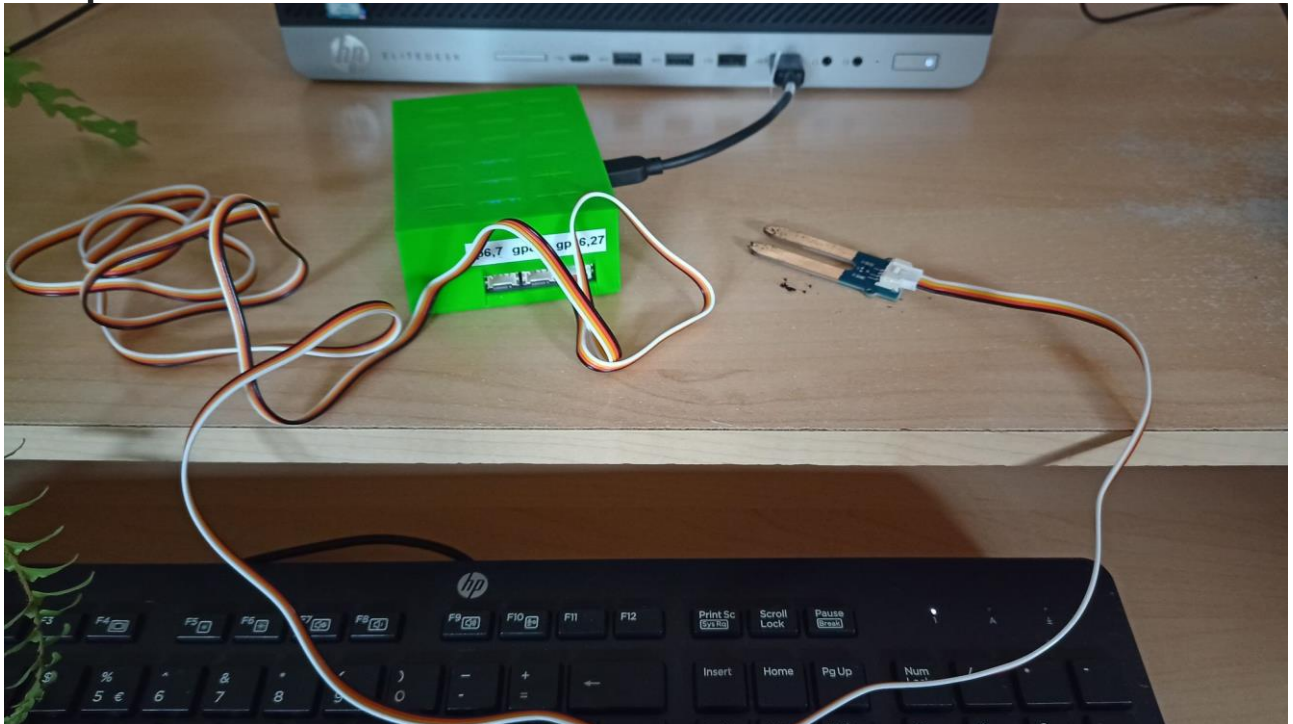


- water

## Software required

- Thonny App (thonny.org)
- custom library not required

## Setup Hardware



Sensor pin	Pico Pin	Grove wire
VCC	3V3	red
GND	GND	black
SIG	GP27	yellow
not connected	GP26	white

The reader use 3V3 power (not 5V!!!). Usually we connect red wire to pin 3V3 and black wire to the ground (GND)



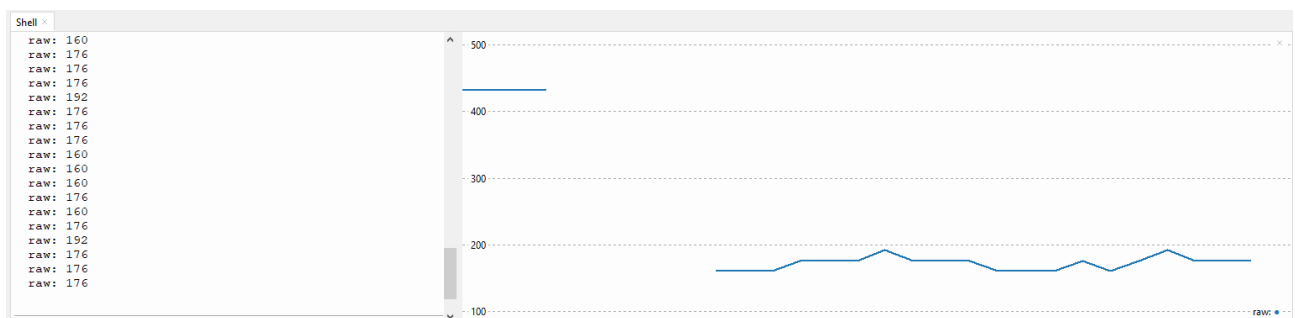
## Setup software

The file `analogwatersensor.py` which contain script for soil testing we must run on our Pico microcontroller. We can do this with Thonny software.

The script for testing

```
1 import utime
2 import machine
3 #Grove Water sensor connected to Cytron Maker ADC1
4 analog_value = machine.ADC(1)
5
6 while True:
7     sensor_raw = analog_value.read_u16()
8     print("raw:", sensor_raw)
9     utime.sleep(2)
```

## Terminal window with results of reading



Possible results are in the scope: 0-65535



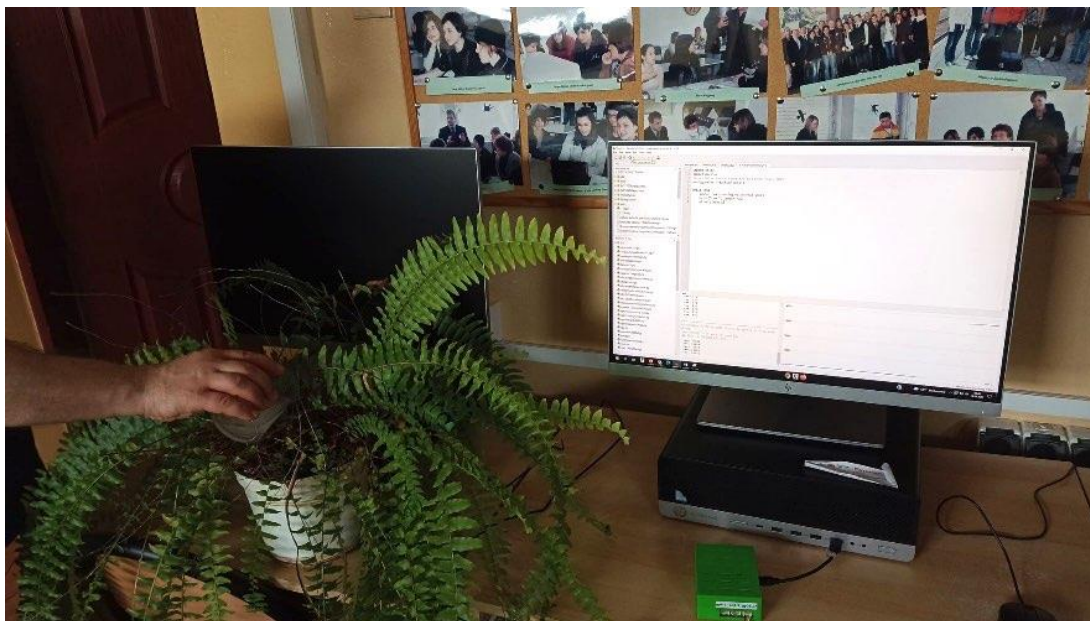
## Description of the sensor

Grove - Soil Moisture Sensor can measure soil moisture for plants. The soil moisture sensor consists of two probes that allow the current to pass through the soil and then obtain resistance values to measure soil moisture content. It can be used to decide if the plants in a garden need watering. You can also use soil moisture sensors in gardens to automate watering plants. It can be used very easily by just inserting the sensor into the soil and reading the output using ADC. You can define the level at which the alarm should be sent or the watering can be started by a relay.

## Experiment Evaluation

### System testing:

- put sensor into different soils
- run the script
- read the results from Thonny's terminal





Write down Your results

plant	average result	description of the state from observation	result after adding to the pot small glass of the water

### Go further:

The script can be modified to give information on a scale of very dry, dry, normal, wet, very wet.

You can also make sure that this information is passed over the network (SMS, website, LoRa WAN).



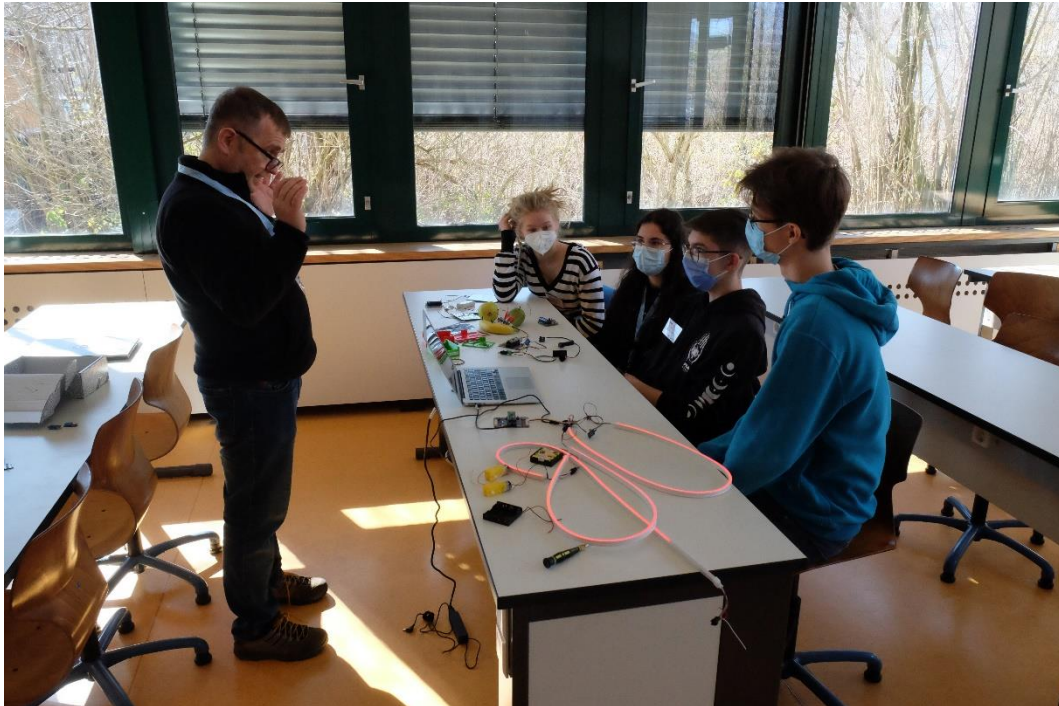


Fürstenfeldbruck

Viscardi Gymnasium

# ELECTRICITY

08.03. – 11.03.2022



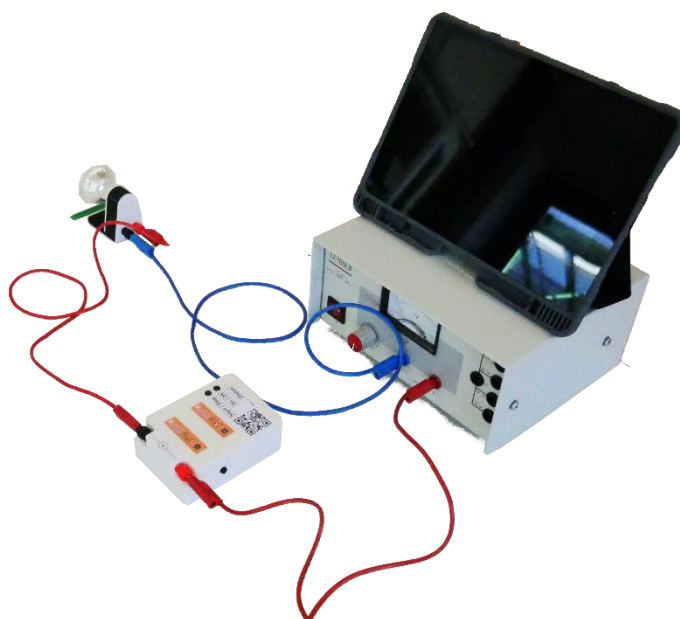


Topic	Age	Country	Date
Comparison of the efficiency of halogen and LED lights (using phyESPx)	>14	Germany	Mar 2022

## Halogen vs. LED

### Experimental setup materials:

- 1 Lab power supply
- 1 Halogen light bulb
- 1 LED light bulb
- 2 PhyESPx server units
- 1 PhyEXPx illuminance sensor
- 1 PhyEXPx current sensor
- 1 iPad
- Some banana wires to connect everything



### Experimental setup and procedure:

**Attach** the **current sensor** to one of the **servers** and connect it to the power supply and the LED bulb **according to the circuit diagram**. Snap the **luminance hat** on the remaining **server unit**, and place it **directly in front of the LED** at around 0.5m distance (the sensor must point **directly towards the light source**). **Power up** the PhyESPx servers, **open** the web interface and put a cardboard box over the lamp and the sensor (in order to avoid light from other sources from being detected by the sensor). **Turn on the PSU** and **start the experiment** using the play button. Crank up the output voltage to **12 Volts** and note down the flowing current and the measured intensity.

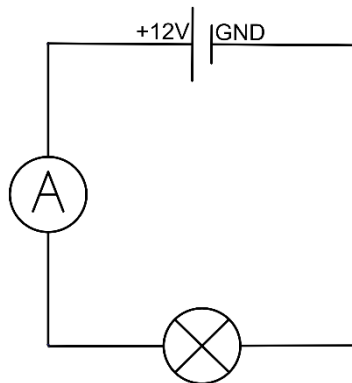
**Repeat** the whole process with the **halogen bulb**. **Make sure** to keep the **distance** between the lamp and the illuminance sensor **remains the same**.

### Experiment evaluation:

- 1) Try to find out which of the lights is more efficient!
- 2) Calculate by how much the LED is more efficient than the halogen lamp.
- 3) Verify your results with the help of the internet



Circuit diagram:



Useful formula:

$$P_{EL} = U \times I$$

$$E_v = \frac{\Phi_v}{A}$$

Experiment results:

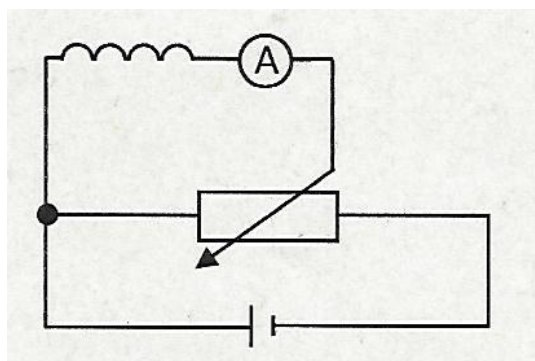
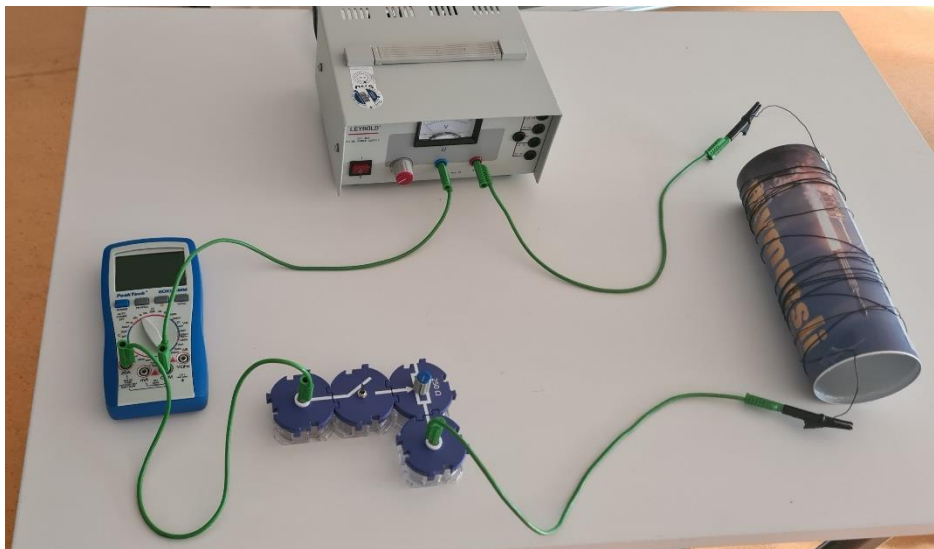
	LED BULB	HALOGEN BULB
CURRENT (IN AMPS)		
VOLTAGE (IN VOLTS)		
POWER (IN WATTS)		
ILLUMINANCE (IN LUX)		
EFFICIENCY (IN LUX PER WATT)		

Evaluation:



Topic	Age	Country	Date
Magnetic field coil	>14	Germany	Mar 2022

- Power Supply (0 - 5V)
- Cylindrical box (e.g. for crisps)
- Ampmeter (0 – 3A)
- Flexible resistance (250 Ohm)
- 2 banana plugs
- Flex ( $A = 0,14\text{mm}^2$ ; length 2 m)
- Cables



1. The bottom and the top of the box have to be removed. Then you can regularly wrap the cable around the box and fix it at the end (by holes or with a tape).
2. Choose the function “Magnetometer” from “Phyphox”. It is advisable to use the possibility of a timed run (fix start delay and duration – e.g. 120s). Even better is to allow the remote access and to use another smartphone or a tablet PC.

#### A) Variation of current

After the start of the measure you choose 0 mA for the first 15s to get the value of the magnetic field of the earth in y-direction. Then you can increase the electric current in steps of e.g. 500mA each 15 s.

Perhaps you have to deactivate the recalibration of magnetic sensor and calculate the real values.



I (A)						
B(μT)						

#### B) Variation of length

Now you can cut the cable in half and keep the current at a fixed value.

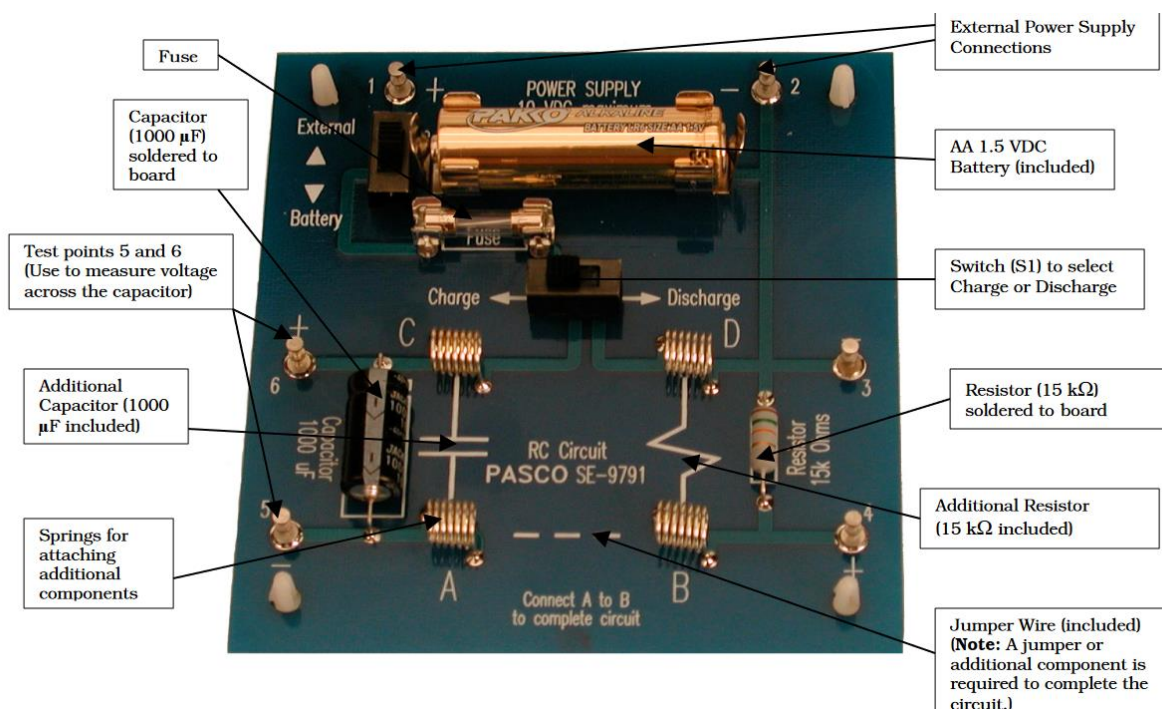
#### C) Conclusion

Topic	Age	Country	Date
Study of RC circuit	>14	Italy	March 2021

## Study of RC circuit

### Experimental setup materials:

- 1 smartphone with SPARKvue APP
- 1 RC circuit
- 1 voltmeter
- 1 Chronometer



### Experimental setup and procedure:

When a capacitor is placed in series with a battery and a resistor, the capacitor charges up to the voltage of the battery.

This kind of circuit is called a RC Circuit because the only two components besides a power supply are a resistor and a capacitor.

The resistor limits the electrical current so that the charging takes place over an extended time.

This allows students time to think about what must be happening as the circuit charges up to the applied voltage of the battery. After the simple voltage-time data for the charging is collected, fundamental electrical quantities of charge, current, and capacitance can be calculated via a spreadsheet.



At the end of the experiment, you will be able to understand how the charge, voltage, and current change as the capacitor charges

How long does it take for the capacitor to charge?

The charging time depends on the capacitance and resistance values of the capacitor.

RC product is called time constant.

When a fully charged capacitor discharges through a resistor, the amount of charge which flows in the first instant is large. However, as more and more charge is transferred, coulombic repulsion starts to slow down the flow.

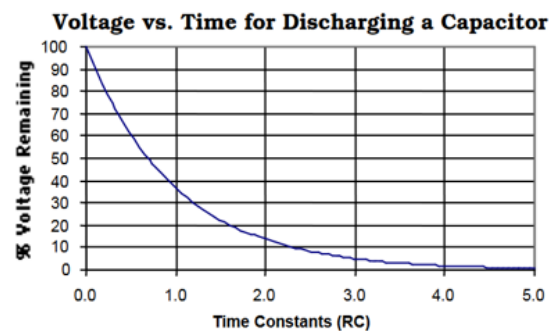
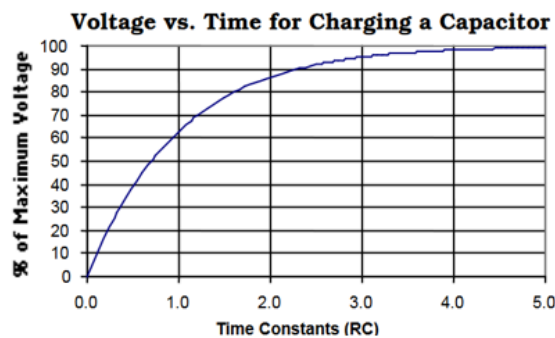
Finally, when the capacitor is fully discharged, each plate of the capacitor is electrically neutral.

The shape of the graph for this discharge is a classical definition for exponential decay.

The time constant for exponential growth or decay, (the Greek letter tau), is the time in seconds for the capacitor to charge or discharge to a certain percent of its final voltage.

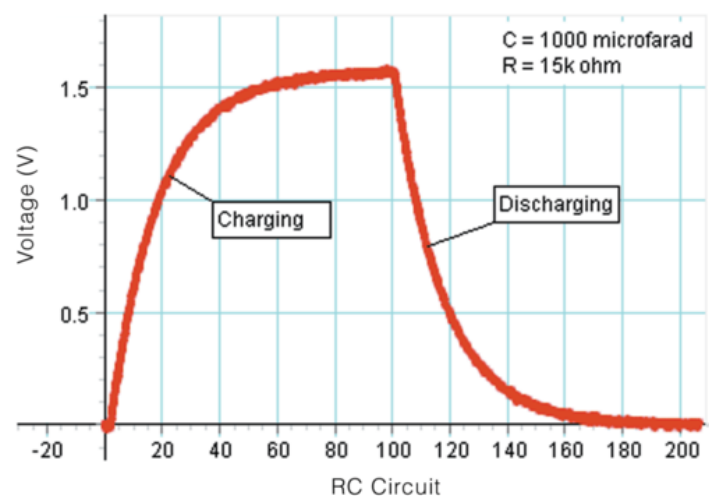
For the charging cycle, when the voltage across the capacitor gets to 63.2 % of its maximum value, one time constant,  $t$  seconds, has passed.

For the discharge cycle, when the voltage across the capacitor gets to 36.8 % of its starting maximum value, one time constant,  $t$  seconds, has passed.



### Experiment evaluation:

- 1) build an RC circuit,
- 2) find the time constant,
- 3) Build the charge and discharge curve by recording voltage value every 5 s with MS-EXCEL







Topic: Electricity	Age	Country	Date
Electric conductivity- Keyboard of fruits and vegetables	>12	Poland	March 2022

## Function, realisation

This project will allow students to understand the phenomenon of electrical conductivity in an attractive way. It is also an opportunity to understand how a conditional statement "IF" works in Python. Conductive material responds when contacted by another electrical conductor, like finger. Students will experimentally confirm that vegetables and fruits, and even the human finger, are electrical conductors. They can also come to the conclusion that the reason for the good operation of the created keyboard is a high water content in fruits and vegetables. They can also replace vegetables with, for example, aluminum foil packaging. Students will learn that the same phenomenon is used in touch screens

## Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18 or above
- Cytron Maker Pico docking station for Pico (or different docking station for Pico: Waveshare, Seeedstudio)
- Grove - 12 Key Capacitive I2C Touch Sensor V3(MPR121) from Seeedstudio
- PC or Mac computer

## Materials required

- 12 Key Capacitive I2C Touch Sensor V3 (MPR121) from Seeedstudio



- 1xGrove wire for connection between MPR121 sensor and Raspberry Pi Pico
- Crocodile --> male DuPont wire (maximum 12)
- up to 12 fruits, vegetables
- micro usb 2.0 high speed (15cm or 30 cm)
- paper for note

### Software required

- Thonny App (thonny.org)
- library: <https://github.com/mcauser/micropython-mpr121/blob/master/mpr121.py>

### Setup Hardware

Connect MPR121 sensor with socket GP6, GP7 on docking station for Raspberry Pi Pico using standard Grove Wire.

Detailed information about this connection

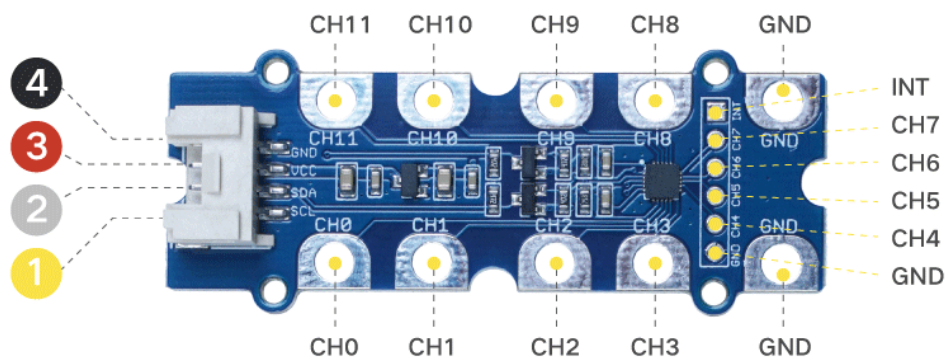
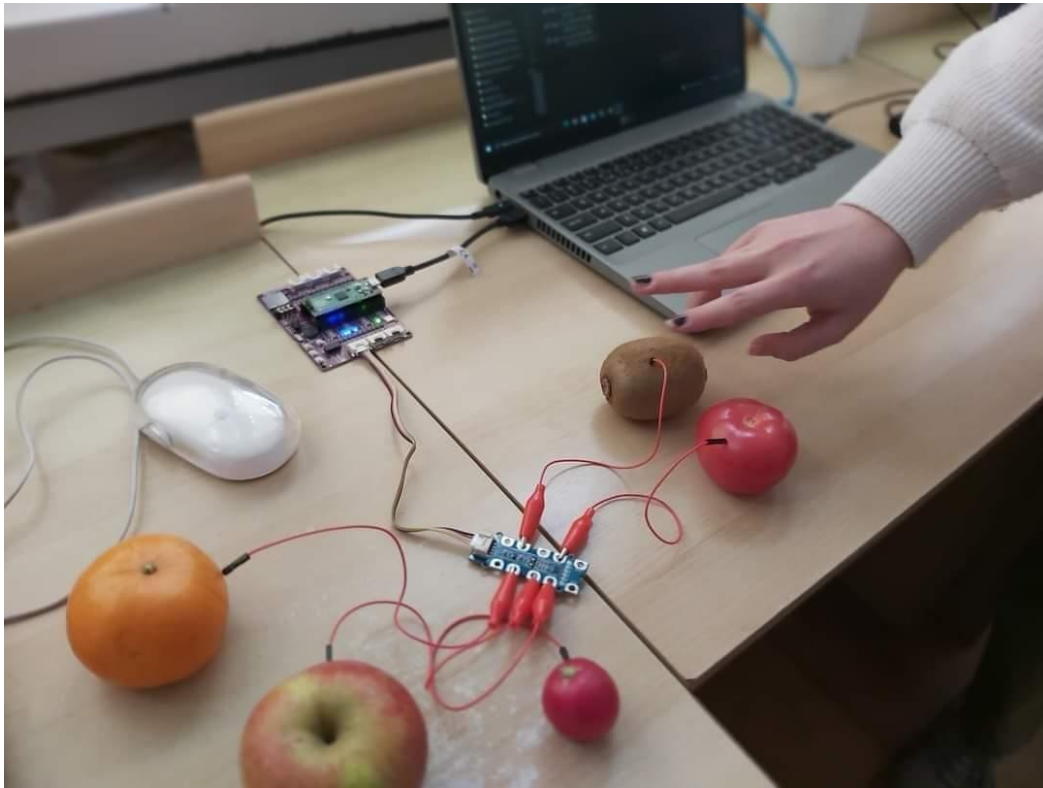
MPR121 Grove socket	Raspberry Pi Pico pins	Grove wire
SCL	GP7	yellow
SDA	GP6	white
VCC	3V3	red
GND	GND	black

Connections between fruits, vegetables and MPR121 sensor is very easy.



Use crocodile wires like on the photo below

(Ch0-Ch11 pads)



- 4 GND: connect this module to the system GND
- 3 VCC: you can use 5V or 3.3V for this module
- 2 SDA: I<sup>2</sup>C serial data
- 1 SCL: I<sup>2</sup>C serial clock

Setup software



Library for MPR121 must be saved inside  
folder "lib" on Raspberry Pi Pico.

We must modify the script according to connection made and to the  
name of fruits/ vegetables.

```
1 import mpr121
2 from machine import Pin, I2C
3 import utime
4 i2c = I2C(1, scl=Pin(7), sda=Pin(6), freq=400000)
5 utime.sleep_ms(100)
6 #It is worth checking if your sensor also has the address 0x5B
7 mpr = mpr121.MPR121(i2c, 0x5B)
8 # check keys one by one
9 while True:
10
11     if mpr.is_touched(0):
12         print("banana")
13     if mpr.is_touched(1):
14         print("mango")
15     if mpr.is_touched(2):
16         print("kiwi")
17     if mpr.is_touched(3):
18         print("orange")
19     if mpr.is_touched(8):
20         print("beetroot")
21     if mpr.is_touched(9):
22         print("apple")
23     if mpr.is_touched(10):
24         print("potato")
25     if mpr.is_touched(11):
26         print("pear")
27
28     utime.sleep_ms(100)
```

MPR121 sensor- main features

- I2C interface Hardware configurable I2C address



- 12 electrodes/capacitance sensing inputs in which 8 are multifunctional for LED driving and GPIO

### Applications

- PC Peripherals
- MP3 Players
- Remote Controls
- Lighting Controls

### Experiment testing, evaluation and application ideas.

#### Go further.

- comparison of the operation of the keyboard depending on the percentage of water in the fruit
- Replacing fruit with aluminum foil packaging
- building a system that generates different sounds depending on the choice of fruit
- Multiple **Grove - 12 Key Capacitive I2C Touch Sensor V3 (MPR121)** to be used together for channel expansions in a single system, you can build a touch system that contains max. 36 electrodes.



Topic: Electricity	Age	Country	Date
Skin resistance measurement-GSR	>14	Poland	March 2022

## Function, realisation

Grove - GSR Sensor stands for galvanic skin response and it is a method of measuring the electrical conductance of the skin. It can be used to reflect human emotional activity. When we are emotionally stressed or have strong expressions on the face, sympathetic activity increases and promotes the secretion of sweat glands, which increases the skin's electrical conductivity.

Grove - GSR allows you to spot such strong emotions by simply attaching two electrodes to two fingers on one hand. It is an interesting gear to create emotion related projects like sleep quality monitor. In some galvanic skin response devices such as lie detectors, this scientific principle is also applied.

In the experiment, we inserted our fingers from the same hand into the two electrode fingertips of the sensor

## Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18 or above





- Cytron Maker Pico docking station for Pico (or different docking station for Pico: Waveshare, Seeedstudio)
- PC or Mac computer

## Materials required

- Grove male wire
- micro usb 2.0 high speed (15cm or 30 cm)

## Software required

-Thonny (thonny.org)

Custom micropython library is not required, because it is analog sensor.

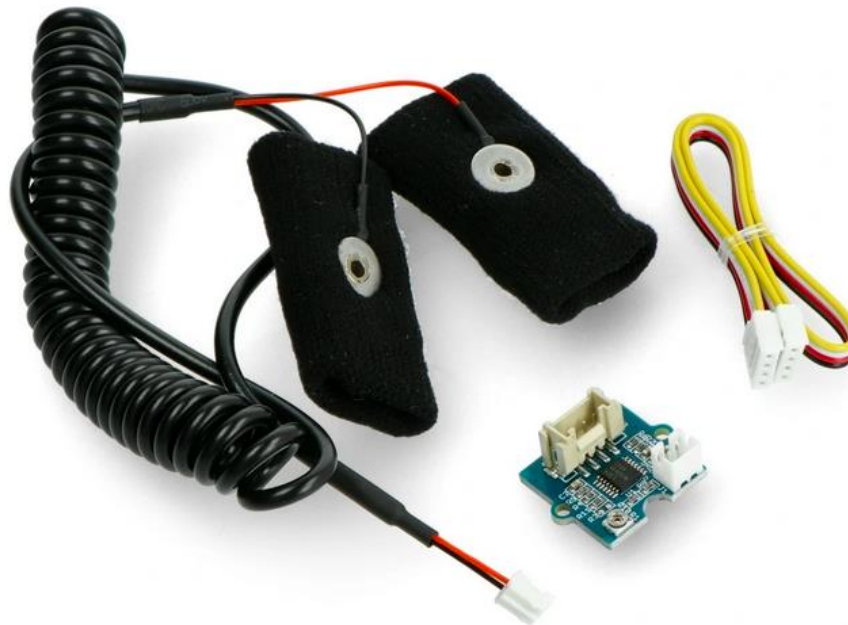
We read raw data in the form of voltage on the pin

## Setup Hardware

```
1 import utime
2 import machine
3 #GSR connected to pin28- ADC2
4 analog_value = machine.ADC(27)
5 #conversion_factor = 3.3/ 65535
6 while True:
7     gsr_raw = analog_value.read_u16()
8     print("raw:", gsr_raw)
9     skin_resistance=(65535+2*gsr_raw)*10000/(32768-gsr_raw)
10    print("Skin resistance", skin_resistance)
11    utime.sleep(2)
```

In our Python script, we assume that the

GSR  
two  
pins:



analog  
sensor  
will use  
signal  
GP26,  
GP27

Sensor	Raspberry Pi Pico pins	Grove
analog signal	GP27	yellow
not connected	GP26	white
power (+)	3V3	red
ground (-)	GND	black

The Pico use 3.3V power (not 5!!!). Usually we connect red wire to pin 3V3 and black wire to the ground (GND)

GSR sensor can show us how person who wore electrodes feels. On the graph, on the top of the page, we can see the moment when someone takes a deep breath.

## Experiment Evaluation



System testing:

- electrical conductance of the skin
- influence of changing feelings on skin resistance

Write down Your results:

.....

.....

.....

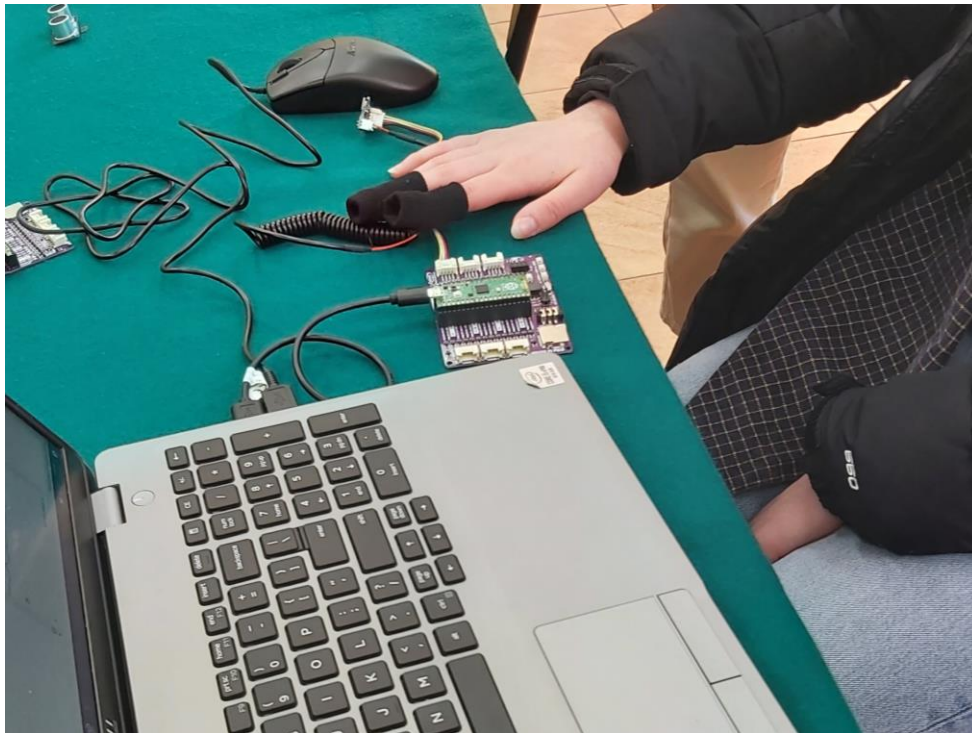
.....

.....



## Go further

- Supplementing the code with the operation of the LCD display
- Building a mobile version (with UPS) adding results logging to the microcontroller memory
- script modification to provide results in Ohm
- logging results during a sleep, math test in classroom



- truthfulness test



Topic: Electricity			
Measurement electric power consumption, current measurement	Age >14	Country Poland	Date March 2022

## Function, realisation

In this experiment we propose energy consumption testing using Gravity Wattmeter module with INA219 Texas with Raspberry Pi Pico. This experiment describes how to measure the energy consumption of the motor, LEDs and sensors. Measurement precision very high: 1%. INA219 is present inside many professional measuring devices.

## Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18 or above
- Cytron Maker Pico docking station for Pico (or different docking station for Pico from Waveshare, Seeedstudio),
- Dfrobot Gravity Wattmeter module with INA219 chip (analog 3.3V for Pico, 5V for Arduino),
  - 3V3 LCD 1602 I2C display Seeedstudio with Grove socket,
  - DHT11 sensor (temperature and humidity) or different 3V3 sensor,
  - 5 led modules with different colour led (red, green, blue, yellow, white),
  - 3V DC motor or different (compatible with additional power source, e. g. 9V battery),
- PC or Mac computer

## Materials required

- Grove male wire (or 3x Dupont male-male wire )
- 1 Grove standard Wire and 1 Grove-Female wire
- Dupont wires (male-female, male-male)
- micro usb 2.0 high speed (15cm, 30 cm or longer)
- paper for note
- optional 9V battery

## Software required

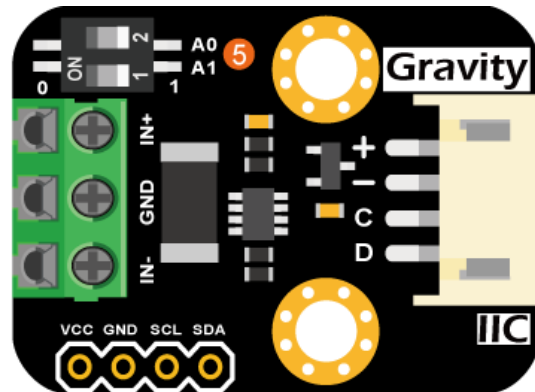
- Thonny App (thonny.org)
- library for INA219 sensor:  
[https://github.com/chrisb2/pyb\\_ina219/blob/master/ina219.py](https://github.com/chrisb2/pyb_ina219/blob/master/ina219.py)

-library for LCD1602:

[https://files.seeedstudio.com/wiki/Grove\\_Shield\\_for\\_Pi\\_Pico\\_V1.0/Libraries.rar](https://files.seeedstudio.com/wiki/Grove_Shield_for_Pi_Pico_V1.0/Libraries.rar)

## Setup Hardware

The Pico **USES** 3.3V power (not 5!!!). Usually we connect red wire to pin 3V3 and black wire to the ground (GND)



## Specification of Gravity Wattmeter module with INA21

- Version: 2.0
- Supply voltage: from 3.3 V to 5 V
- Voltage measurement range: 0 V to 26 V
- Resolution voltage: 4 mV
- Error of voltage measurement: up to 0,2% (typical)
- Measuring range of the current: from 0 to 8 A (bi-directional)
- Resolution of the current: 1 mA
- Error of current measurement: up to 2% (typical, it requires manual calibration)
- The range of power measurement: from 0 to 206 W
- Resolution power: 20 mW (hardware) / 4 MW (software)
- Quiescent current: 07 mA
- Interface: I2C Gravity
- I2C address: 0x40, 0x41, 0x44, 0x45
- Dimensions: 30 x 22 mm

## Power Pins

The sensor on the breakout requires between a 2.7V and 5.5V, and can be easily used with most microcontrollers

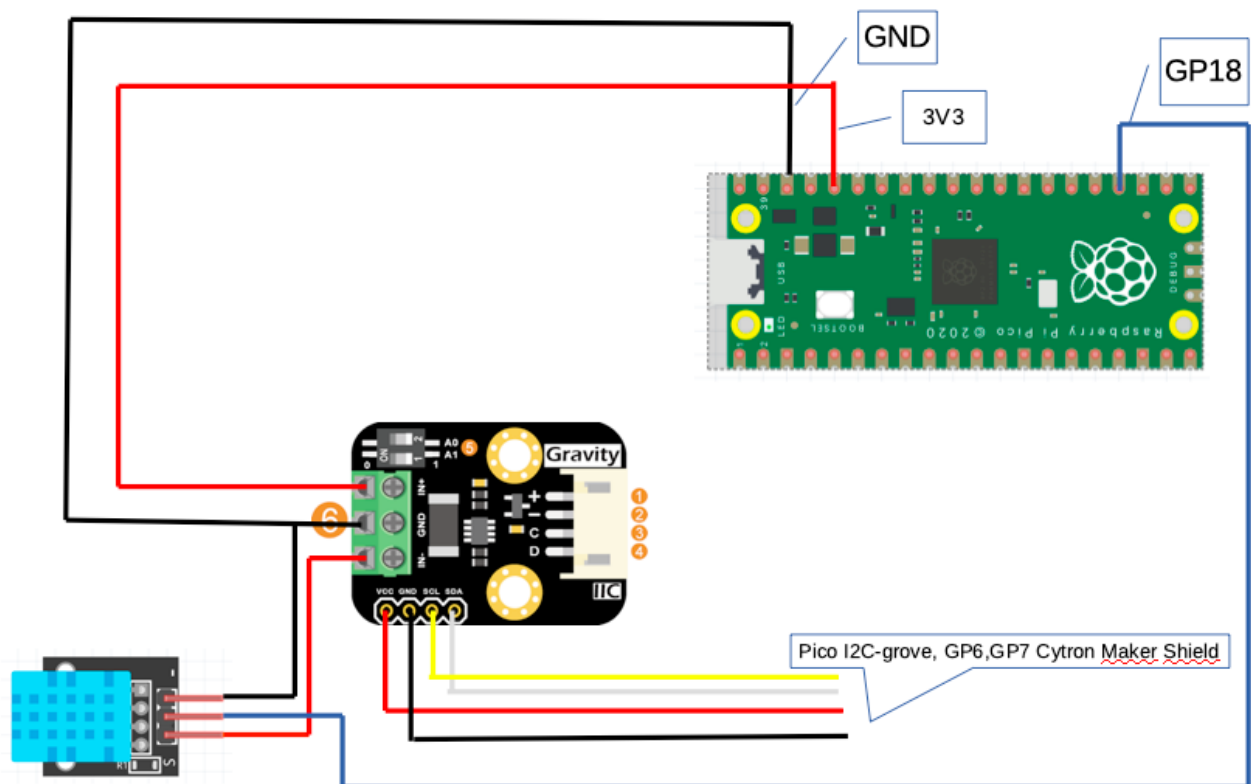
- **Vin+** is the positive input pin.

Connect to supply for high side current sensing or to load ground for low side sensing.

- **Vin-** is the negative input pin. Connect to load for high side current sensing or to board ground for low side sensing

**GND** - This is common ground for power and logic.

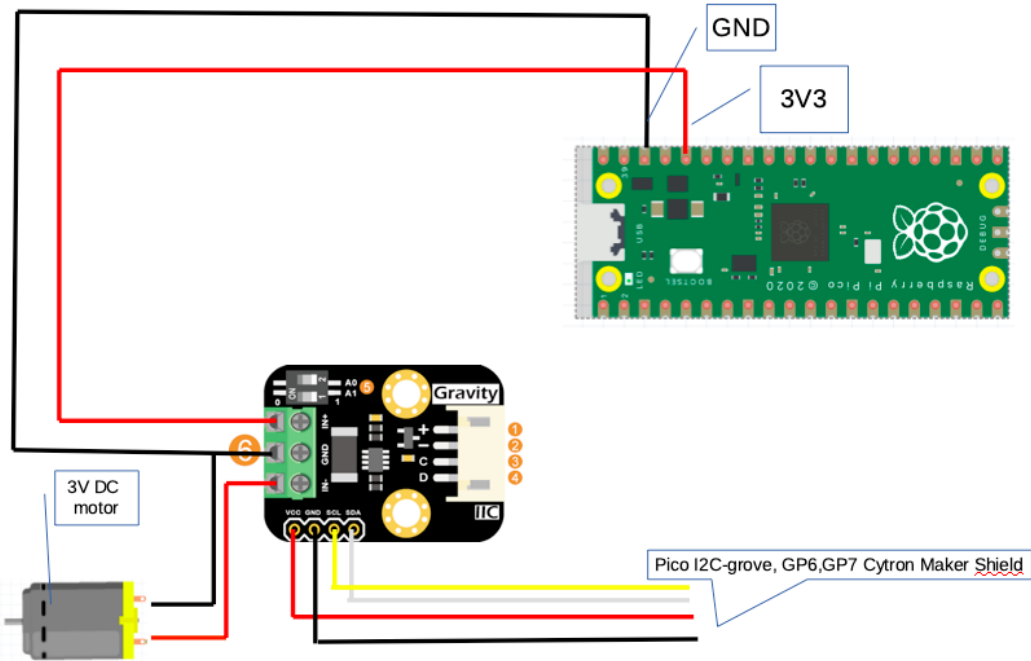
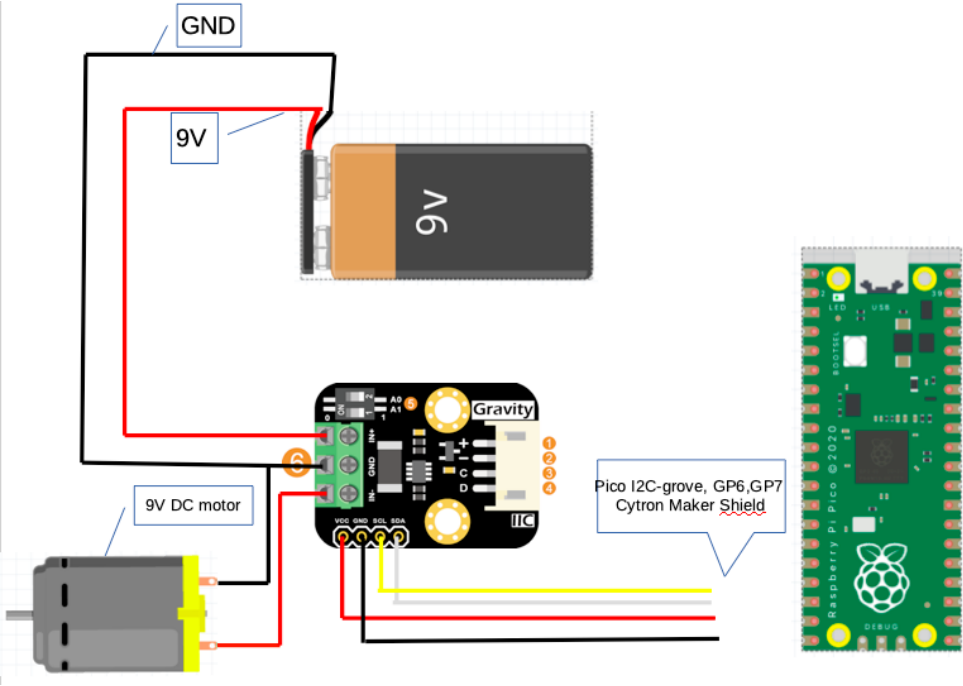
This script requires an LCD screen with GROVE socket which is really



easy to connect.

**A0 and A1 solder jumpers** -You can use it, if you want change the I2C address (in the case of many Wattmeters used at the same I2C bus).



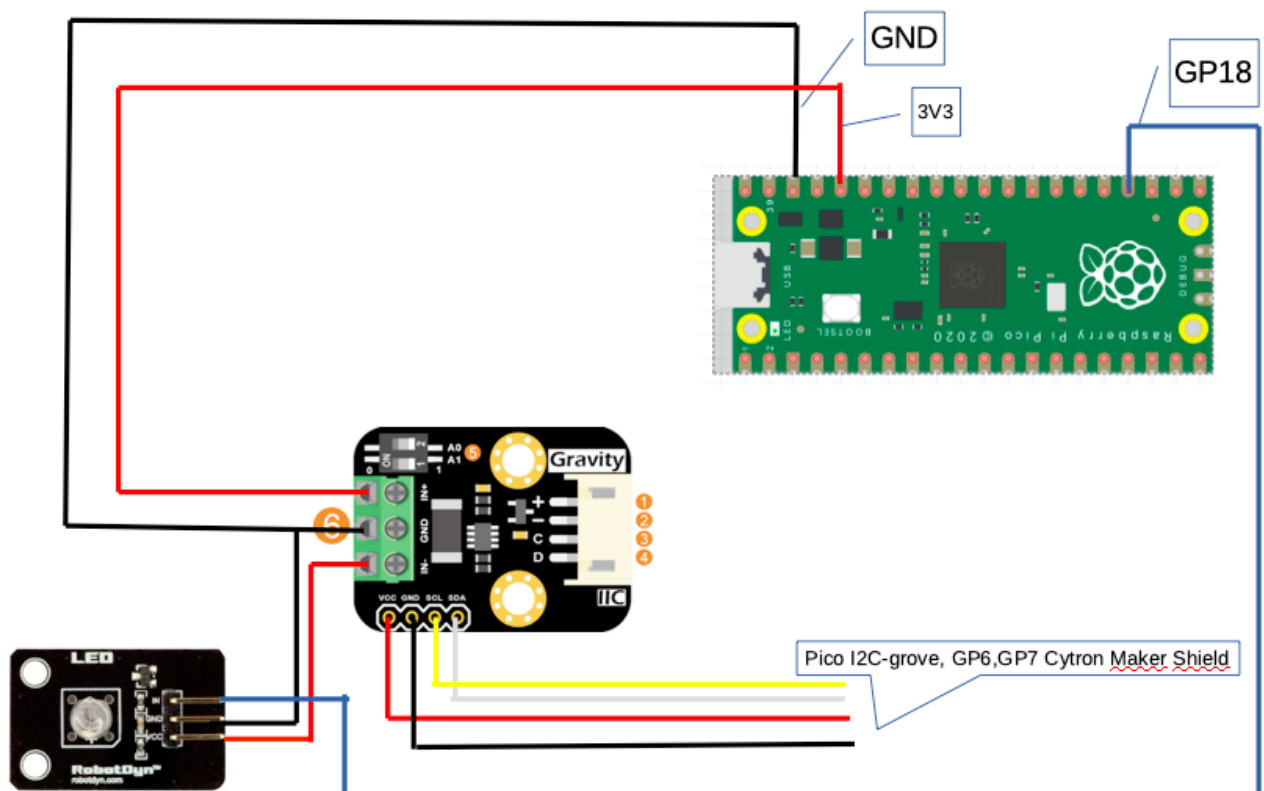


LCD 1602 pins	Raspberry Pi Pico pins	Grove wire
GND	GND	black
VCC	3V3	red

SDA	GP8	white
SCL	GP9	yellow

## Wattmeter-Pins

- **Vin+** is the positive input pin. Connect to supply for high side current sensing or to load ground for low side sensing.
- **Vin-** is the negative input pin. Connect to load for high side current sensing or to board ground for low side sensing
- according to the list below.



## Setup software

### Libraries installation

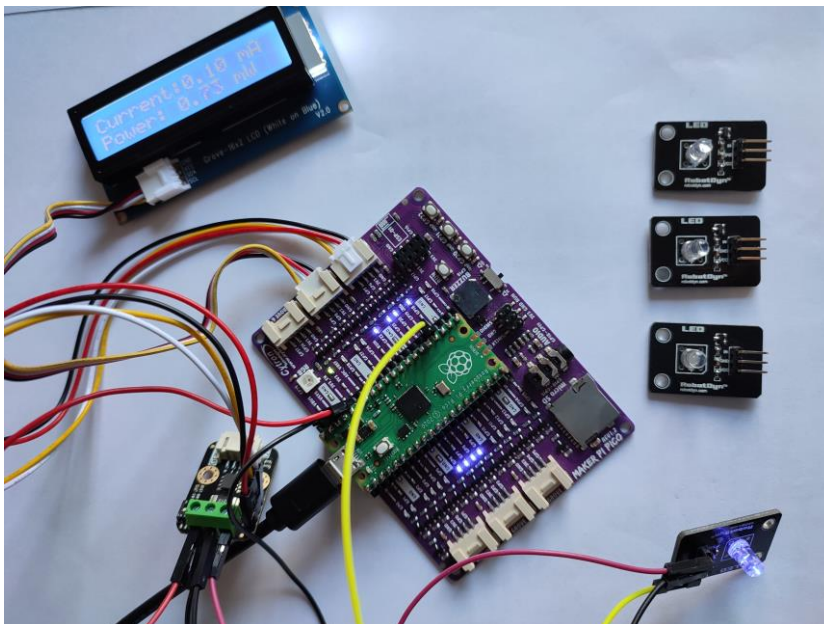


In Thonny Application we can open Python file with library downloaded from github: and save in the lib folder on the Pico

```
1 from lcd1602 import LCD1602
2 from machine import Pin, I2C
3 from ina219 import INA219
4 from logging import INFO
5 from utime import sleep
6 from dht11 import DHT
7 LEDtest = machine.Pin(15,machine.Pin.OUT)
8 LED = machine.Pin(25,machine.Pin.OUT)
9 LED.value(0)
10 LEDtest.value(1)
11 SHUNT_OHMS = 0.1
12 i2cbis = I2C(0, scl=Pin(9), sda=Pin(8), freq=400_000)
13 i2c = I2C(1,scl=Pin(7), sda=Pin(6), freq=400000)
14
15 d = LCD1602(i2cbis, 2, 16)
16 d.display()
17
18 dht = DHT(18)
19
20 ina = INA219(SHUNT_OHMS, i2c, log_level=INFO)
21 ina.configure()
22
23 for i in range(60):
24     LED.value(1)
```

```
25     sleep(1)
26     d.clear()
27     LED.value(0)
28     d.setCursor(0,0)
29     #print("Bus Voltage: %.2f V" % ina.voltage())
30     #d.print("B:%.3f V" % ina.voltage())
31     #d.setCursor(8,0)
32     print("Current: %.3f mA" % ina.current())
33     temp,humid = dht.readTempHumid()
34     d.print("Current:%.2f mA" % ina.current())
35     print("Power: %.3f mW" % ina.power())
36     d.setCursor(0,1)
37     d.print("Power: %.2f mW" % ina.power())
38     sleep(1)|
```

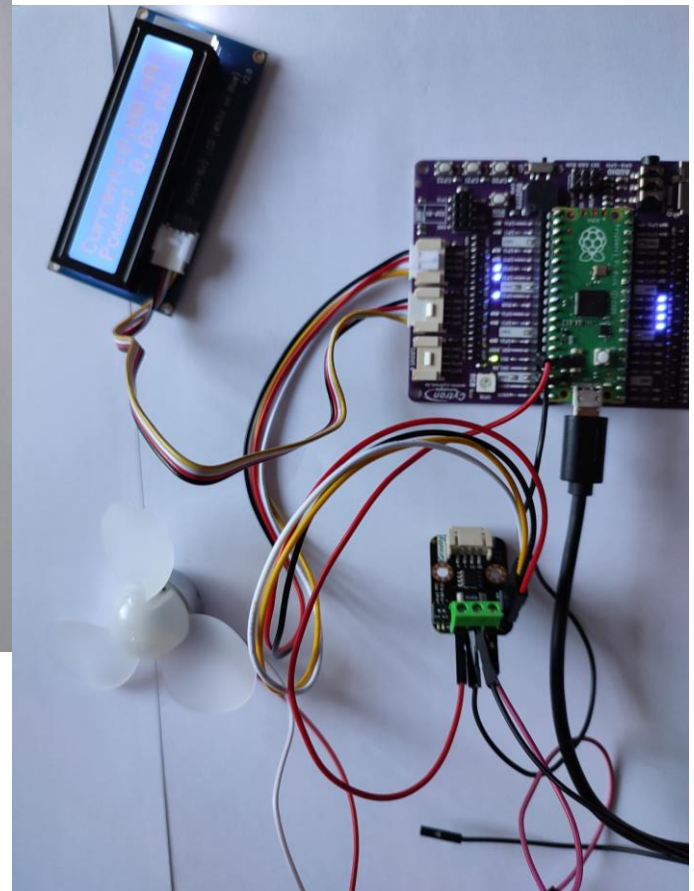
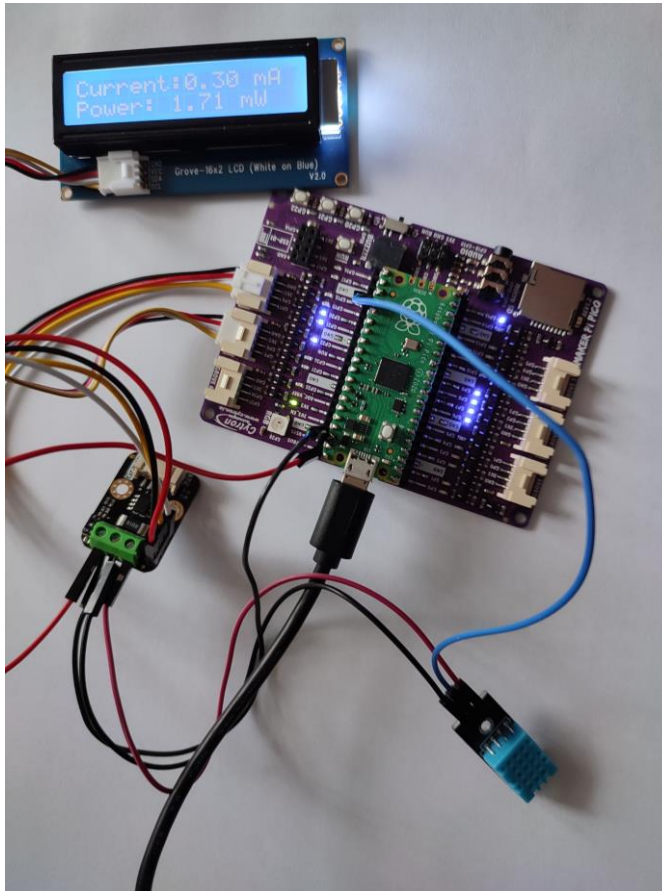
## Demonstration of the system operation:



Testing power led's consumption depend of the colour

Testing DHT11 power consumption and DC motor  
power consumption:

**Experiment testing,  
evaluation and application  
ideas:**



System testing:

- measurement the power consumptions of various sensors connected to microcontroller,
  - measurement power consumptions by leds depending of the colour of the led (the same size of the led from the same manufacturer),
  - measurement the power consumption of DC motors,
  - measurement power generated by mini solar panels,
  - measure discharge of battery,
  - find dependencies between power consumption of DC motor under the weight and the mass,
- change the Python script to plot graph based on results (using internal ploter from Thonny App).

The code for testing Led's power consumption:





```
1 from lcd1602 import LCD1602
2 from machine import Pin, I2C
3 from ina219 import INA219
4 from logging import INFO
5 from utime import sleep
6 #external LED
7 LED_EX = machine.Pin(18,machine.Pin.OUT)
8 LED_EX.value(0)
9 SHUNT_OHMS = 0.1
10 #I2C bus intialisation
11 i2cbis = I2C(0, scl=Pin(9), sda=Pin(8), freq=400_000)
12 i2c = I2C(1,scl=Pin(7), sda=Pin(6), freq=400000)
13 d = LCD1602(i2cbis, 2, 16)
14 d.display()
15
16 ina = INA219(SHUNT_OHMS, i2c, log_level=INFO)
17 ina.configure()
18
19 for i in range(60):
20     d.clear()
21     LED_EX.value(1)
22     d.setCursor(0,0)
23     print("Current: %.3f mA" % ina.current())
24     d.print("Current: %.2f mA" % ina.current())
25     print("Power: %.3f mW" % ina.power())
26     print("Bus Voltage: %.2f V" % ina.voltage())
27     d.setCursor(0,1)
28     d.print("Power: %.2f mW" % ina.power())
29     sleep(2)
```



Write down your results

Sensor/led/motor	power consumption	min	max
red led			
green led			
blue led			
yellow led			
white led			
DC motor 3V			
DHT11			
DC motor under mass			





Topic	Age	Country	Date
Earth magnetic field	$\geq 14$	Portugal	March 2022

## Earth magnetic field - Basic information

(from <https://ngdc.noaa.gov/geomag/declination.shtml>)

The Earth acts like a large spherical magnet: it is surrounded by a magnetic field that changes with time and location. The field is generated by a dipole magnet (i.e., a straight magnet with a north and south pole) located at the centre of the Earth.

The axis of the dipole is offset from the axis of the Earth's rotation by approximately 9 degrees. This means that the north and south geographic poles and the north and south magnetic poles are not located in the same place.

At any point and time, the Earth's magnetic field is characterized by a direction and intensity which can be measured.

Often the parameters measured are the magnetic declination,  $D$ , the horizontal intensity,  $H$ , and the vertical intensity,  $Z$ . From these elements, all other parameters of the magnetic field can be calculated.

The Earth's magnetic field intensity is roughly between 25 – 65  $\mu\text{T}$

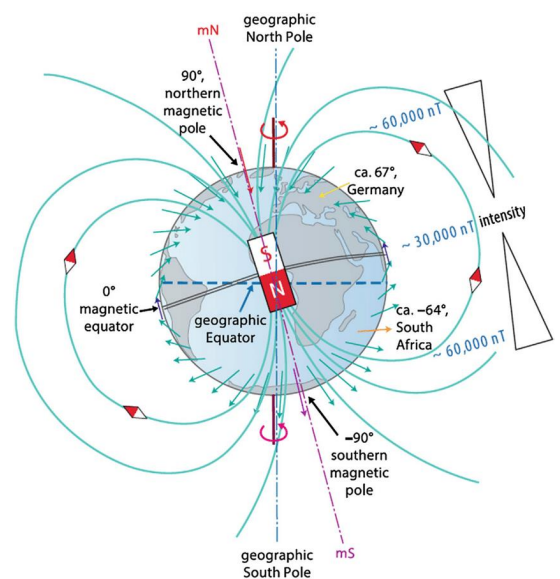
Magnetic declination is the angle between magnetic north and true north.  $D$  is considered positive when the angle measured is east of true north and negative when west.

Magnetic inclination is the angle between the horizontal plane and the total field vector, measured positive into Earth.

The magnetic field is different in different places. In fact, the magnetic field changes with both location and time

The magnetic poles are defined as the area where declination is vertical. Based on the NOAA models, the geomagnetic north pole is at 72.68°W longitude and 80.65°N latitude. The axis of the dipole is currently inclined at 9.41° to the Earth's rotation axis.

The Earth's magnetic field is slowly changing and appears to have been changing throughout its existence.



### Does the compass needle point toward the magnetic pole?

**No.** The compass points in the directions of the horizontal component of the magnetic field where the compass is located, and not to any single point.

### What influences the magnetic field measured by my compass?

The Earth's magnetic field is actually a composite of several magnetic fields generated by a variety of sources. These fields are superimposed on each other and through inductive processes interact with each other. The most important of these geomagnetic sources are:

- the Earth's conducting, fluid outer core (~90%)
- magnetized rocks in Earth's crust
- fields generated outside Earth by electric currents flowing in the ionosphere and magnetosphere
- electric currents flowing in the Earth's crust (usually induced by varying external magnetic fields)



e. ocean current effects

## Determination of the $\vec{B}_{\text{earth}}$

### Horizontal direction of the field.

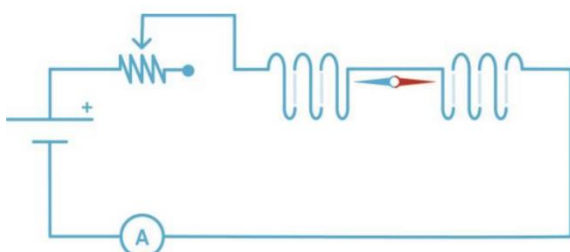
Fix a paper sheet in a plane surface, like a table, use a compass to get the horizontal direction of the field. Certify that the measurement **is** done far from other magnetic fields or ferromagnetic materials and in a horizontal surface.

Trace the horizontal direction of **the** magnetic field component of the field in the paper.

### Horizontal magnitude of the field

### Building a simple apparatus based on electromagnetic phenomena

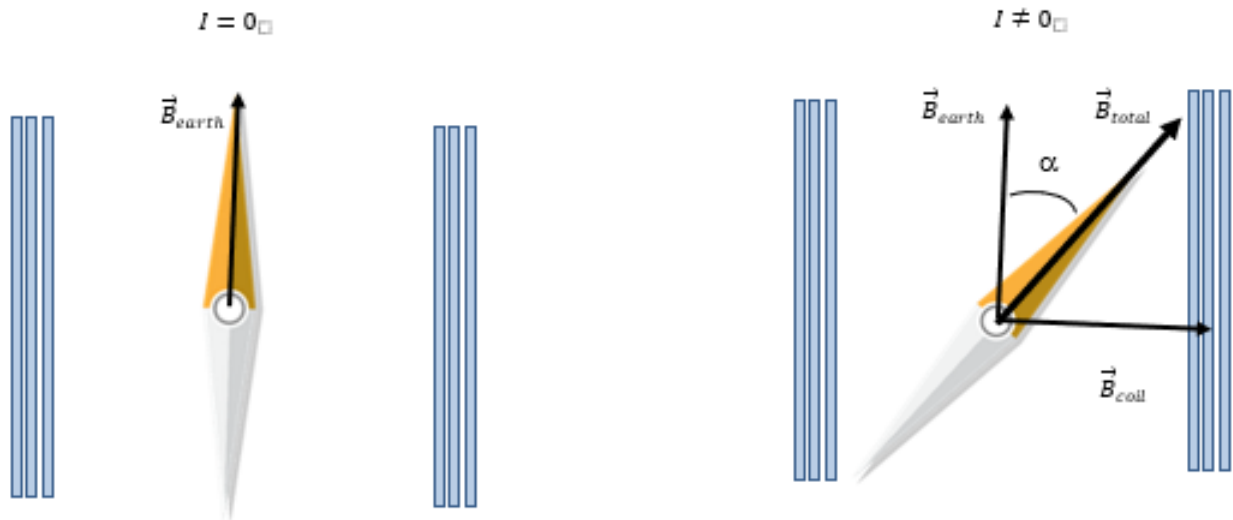
The experiment requires the construction of a Helmholtz coil and the determination of **an** angle displaced by the needle of a compass placed inside it, due to an electrical current.



#### Material:

- 0.5 l empty plastic water bottle with parallel grooves;
- 1 ammeter (0-200mA);
- Insulated copper wire with varnish;
- Support 1.5V batteries
- Compass
- Rheostat from 100Ω to 1kΩ
- Electric wire qb
- Caliper
- connectors

When an electric current  $I$  is flowing through the coil, a uniform magnetic field  $B_{\text{coil}}$  is created in the region between the coils of radius  $R$  and  $n$  spires; if the distance between the coil equal to their radius, this field is determined by  $B = (4/5)^{\frac{3}{2}} \frac{\mu_0 n I}{R}$ , where  $\mu_0$  is the permeability of free space ( $\mu_0 = 4\pi \times 10^{-7} \text{ (SI)}$ )



Setting the angle  $\alpha=45^\circ \rightarrow |B_{\text{earth}}| = |B_{\text{coil}}|$

Setting the coil in order to be simple to measure  $B_{\text{coil}}$ .

- A- Measure the diameter of the bottle;
- B- Determine the  $n$  spires needed on each coil in order to get  $B_{\text{coil}} = 1 \mu\text{T}$  when a  $I = 1 \text{ mA}$ ; this will depend on the radius of the bottle.

$$n = \left(\frac{4}{5}\right)^{-3/2} \frac{r}{\mu_0 I} B \cong 26$$

(17, no nostro caso)

- C- A value of **a** few dozen off turns is expected (i.e., 15 to 30)

Place the compass inside the bottle and between the coils according to their medium plane, align the bottle so that the coils are parallel to the compass needle. In this way we ensure that when the electric current begins, the created field will be orthogonal to the earth field;

The circuit is connected and the resistance of the rheostat is reduced in order to increase the current circuit, which is easily observed when measuring the mA value in the ammeter. **Because we choose the number of turns properly, this measure indicates the value of the field magnetic directly in microtesla.**

**When the compass needle deviates  $45^\circ$  the field created is of equal value to the local field that in the beginning is only that of the Earth.** It is advisable to verify that there are no objects with significant masses of **ferromagnetic material** nearby.

Due to the sensitivity of the compass, it is also advisable to perform the experiment a dozen **d** times *either approaching by excess or by default* the  $45^\circ$  measurement. In this way, the systematic errors due to internal friction **within the** needle are minimized.



$B_{\text{earth}} / \mu\text{T}$ (=Bcoil= "miliamperimeter")	$B_{\text{earth}} / \mu\text{T}$ (mean value)	

## Vertical magnitude off the field

Using high technologies – your mobile phone!

Open the "magnetometer" on the Phyphox.

Place the telephone in a horizontal plane, far from objects with significant masses of material ferromagnetic nearby.

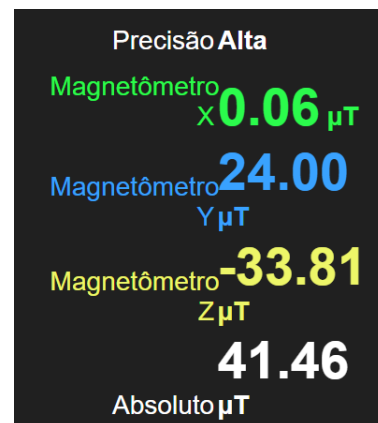
Aline the phone in order to have the xx axis with zero field and the zz axis vertically.

Compare the  $B_y$  value off the field with the horizontal component obtain previously with the electromagnetic experiment. Comment on the results.

Get the missing  $B_z$  component of earth magnetic field in the day and location where the experiment took place! Thanks to Viscardi-Gymnasium Fürstenfeldbruck for this opportunity!

Final results:

GPS coordinates		$B / \mu\text{T}$			
Latitude/°	Longitude/°	$B_x$	$B_y$	$B_z$	$B_{\text{value}}$





Co-funded by the  
Erasmus+ Programme  
of the European Union

## Physics and New Technologies



You can compare the results with the ones derived from the American Model from NOAA (National Oceanic and Atmospheric Administration) in:  
<https://www.ngdc.noaa.gov/geomag/calculators/magcalc.shtml#igrfwmm>



Topic	Age	Country	Date
Eddy currents	$\geq 14$	Portugal	March 2022

## Experiments with neodymium magnets #1

### 1- Introduction: How electromagnetic brakes work?

Over their lifetime running, they can cost half as much as traditional friction brakes. What are they and how do they work? Let's take a closer look!



### 2-Information on neodymium magnets (from Wikipedia<sup>1</sup>)

"A neodymium magnet (also known as NdFeB, NIB or Neo magnet) is the most widely used type of rare-earth magnet. It is a permanent magnet made from an alloy of neodymium, iron, and boron to form the Nd<sub>2</sub>Fe<sub>14</sub>B tetragonal crystalline structure. Developed independently in 1984 by *General Motors* and *Sumitomo Special Metals*, neodymium magnets are the strongest type of permanent magnet available commercially."<sup>1</sup>

### 3- Experiment:

- A- Test the material available with the magnet. Divide them in 1- *metals that are attracted to the magnet* (ferromagnetic), 2- *metals that are NOT attracted to the magnet* (diamagnetic) and 3- *non-metals* (diamagnetic)
  
- B- Let the neodymium magnet slip from a non-metal surface of an inclining plate. Observe and pay attention to the time the magnet takes to slip across the plane.

What to expect if we do a similar experiment with a ferromagnetic plate (e.g. iron)?

What to expect if we do a similar experiment with a non-ferromagnetic plate (like aluminium or copper)?

<sup>1</sup> [https://en.wikipedia.org/wiki/Neodymium\\_magnet](https://en.wikipedia.org/wiki/Neodymium_magnet)



- C- Let the neodymium magnet slip from a non-ferromagnetic plate (like aluminium or **copper**).

Now use the similar plate but make the magnet slip above the cut.  
Compare it with the uncut nom(?) -metal experiment.



How to understand these results?





Topic	Age	Country	Date
Gauss cannon	$\geq 14$	Portugal	March 2022

## Experiments with neodymium magnets #2

### The magnetic accelerator

Start with a simple neodymium magnet. If you roll a steel ball toward the magnet, the magnet induces a magnetic dipole within the ball. A strong force of attraction pulls the two objects together.

If you roll a second steel ball toward the first ball, an attraction occurs again, but this time **it's** weaker. The third ball feels a weaker attraction still.

Remove one ball at a time, noting the increasing force that you need to apply as you remove each one.

-----



Attach the three balls again and place the system in the groove of the track.

Take a fourth ball and place it in the groove on the opposite side of the magnet, but do not **allow it to** get pulled in by the magnet.

Give the ball a small push, **just** enough that it rolls slowly toward the magnet-ball system. The incoming ball should strike the magnet and knock loose a ball on the opposite side. Repeat this scenario a few times. [Feel free to try out different scenarios – e.g., 4 balls on one side, 2 on the other, etc. – to familiarize yourself with the basic interaction.]

An exact model of the magnetic accelerator it's not basic physics ... but we can make a first approach from some basic **laws** of physics, like Energy Conservation, and maybe get to know interactions better.

## Experiments with neodymium magnets #2



### The magnetic accelerator

initial – 1



final -2



1. Would you say that **the spheres and the magnet are more strongly attached in** the initial system 1 () or **in** the final system 2 ()?

(It may be helpful to actually feel the magnet and the balls in both configurations to get a tactile sense of how tightly they are held together.)

2. Think in a more familiar situation, for example, the gravitational force. When a body approximates the earth, **does** the force **during the** interaction increase or decrease? And the potential energy between them, **does it** increase or decrease?

3. Based on your previous two answers, would you say that the magnet-ball system has a lower potential energy in the final (2) or in the initial (1) configuration? (When it is 'weakly bonded' or 'strongly bonded')?

4. Diagram this scenario with an energy diagram you like (energy pies, energy bars, etc). Consider four moments:

- 1) the incoming ball rolling very slowly toward the magnet-ball system,



- 2) the moment before the incoming ball strikes the magnet,
- 3) the moment the outgoing ball releases from the right side, and
- 4) the outgoing ball farther away.

5. If a magnet and ball are far apart, then no discernible attraction is felt. We say that the magnet-ball system has zero magnetic potential energy. When the ball and magnet come together, does the magnet-ball system have positive, zero, or negative energy? Explain your reasoning.

7. In this demonstration, it seems as though “we get something for nothing.” The incoming **ball** **is** given a small push and the outgoing ball exits with a large amount of kinetic energy. To explore what is going on in more detail, take a glass marble and roll it slowly toward the initial setup. What happens? What do you need to do in order to actually remove the outgoing ball?

8. Based on the above example, which phrase is more appropriate: “making a bond releases energy” or “making a bond requires energy”?

## Appendix 1 - Magnetic properties of matter

(from <https://www.britannica.com/science/magnetism/Induced-and-permanent-atomic-magnetic-dipoles>)

All matter exhibits magnetic properties when placed in an external magnetic field. Even substances like copper and aluminium that are not normally thought of as having magnetic properties are affected by the presence of a magnetic field such as that produced by either pole of a bar magnet. Depending on whether there is an attraction or repulsion by the pole of a magnet, matter is classified as being either paramagnetic or diamagnetic, respectively. A few materials, notably iron, show a very large attraction toward the pole of a permanent bar magnet; materials of this kind are called ferromagnetic.

In 1845 Faraday became the first to classify substances as either diamagnetic or paramagnetic. He based this classification on his observation of the force exerted on substances in an inhomogeneous magnetic field. At moderate field strengths, the magnetization  $M$  of a substance is linearly proportional to the strength of the applied field  $H$ . The magnetization is specified by the magnetic susceptibility  $\chi$ , defined by the relation  $M = \chi H$ .

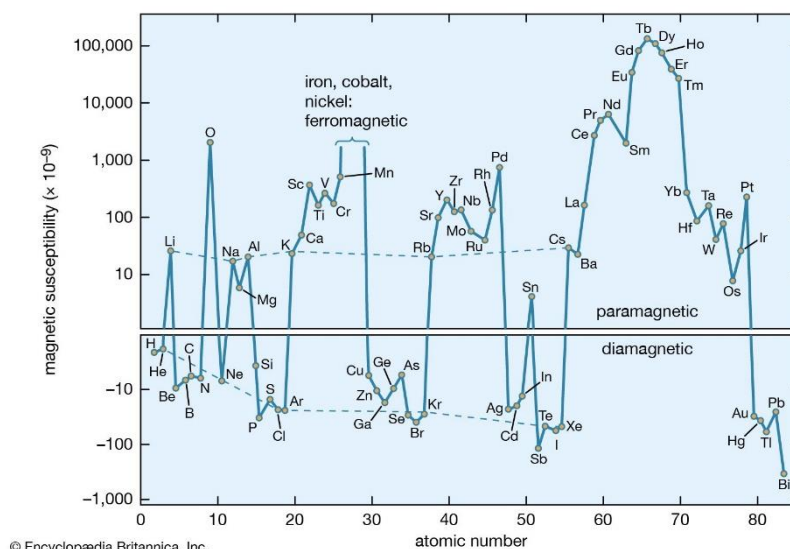
Substances for which the magnetic susceptibility is negative (e.g., copper and silver) are classified as diamagnetic.

Substances for which the magnetic susceptibility is positive are classed as paramagnetic.

### Induced and permanent atomic magnetic dipoles

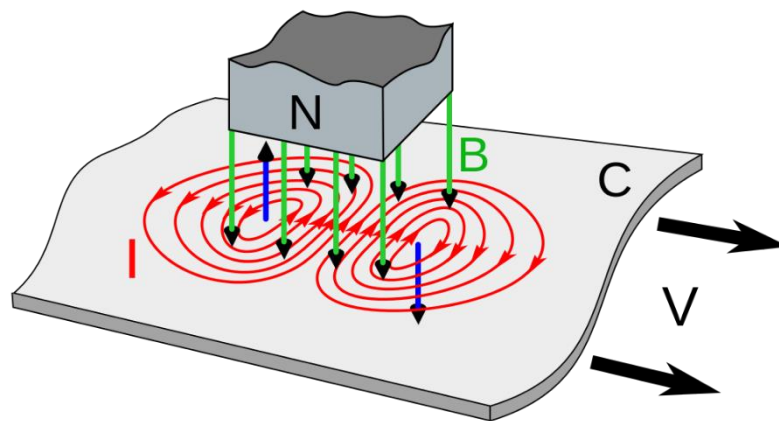
Whether a substance is paramagnetic or diamagnetic is determined primarily by the presence or absence of free magnetic dipole moments (i.e., those free to rotate) in its constituent atoms. When there are no free moments, the magnetization is produced by currents of the electrons in their atomic orbits. The substance is then diamagnetic, with a negative susceptibility independent of both field strength and temperature.

In matter with free magnetic dipole moments, the orientation of the moments is normally random and, as a result, the substance has no net magnetization. When a magnetic field is applied, the dipoles are no longer completely randomly oriented; more dipoles point with the field than against the field. When this results in a net positive magnetization in the direction of the field, the substance has a positive susceptibility and is classified as paramagnetic.



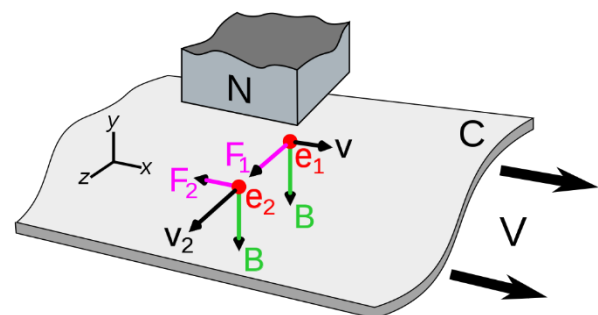
## Appendix 2 – Eddy currents

Eddy currents (also called Foucault's currents) are loops of electrical current induced within conductors by a changing magnetic field in the conductor according to Faraday's law of induction. Eddy currents flow in closed loops within conductors, in planes perpendicular to the magnetic field. They can be induced within nearby stationary conductors by a time-varying magnetic field created relative motion between a magnet and a nearby conductor. The magnitude of the current in a given loop is proportional to the strength of the magnetic field, the area of the loop, and the rate of change of flux, and inversely proportional to the resistivity of the material. When graphed, these circular currents within a piece of metal look vaguely like eddies or whirlpools in a liquid.



Eddy currents ( $I$ , red) induced in a conductive metal plate ( $C$ ) as it moves to the right under a magnet ( $N$ ). The magnetic field ( $B$ , green) is directed down through the plate. The Lorentz force of the magnetic field on the electrons in the metal induces a sideways current under the magnet. The magnetic field, acting on the sideways moving electrons, creates a Lorentz force opposite to the velocity of the sheet, which acts as a drag force on the sheet. The (blue arrows) are counter magnetic fields generated by the circular motion of the charges.

Forces on an electron in the metal sheet under the magnet, explaining where the drag force on the sheet comes from. The red dot shows a conduction electron in the sheet right after it has undergone a collision with an atom, and shows the same electron after it has been accelerated by the magnetic field. On average the electron has the same velocity as the sheet (black arrow) in the direction. The magnetic field (green arrow) of the magnet's North pole  $N$  is directed down in the direction. The magnetic field exerts a Lorentz force on the electron (pink arrow) of  $\vec{F}_1$ , where  $e$  is the electron's charge. Since the electron has a negative charge, from the right-hand rule this is directed in the direction. At this force gives the electron a component of velocity in the sideways direction (black arrow)  $\vec{v}_2$ . The magnetic field acting on this sideways velocity, then exerts a Lorentz force on the particle of  $\vec{F}_2$ . From the right-hand rule, this is directed in the direction, opposite to the velocity of the metal sheet. This force accelerates the electron giving it a component of velocity opposite to the sheet. Collisions of these electrons with the atoms of the sheet exert a drag force on the sheet.





Co-funded by the  
Erasmus+ Programme  
of the European Union

## Physics and New Technologies



---



Póvoa de Varzim

Escola Secundária de Rocha Peixoto

# W A V E S

17.05. – 20.05.2022







Topic	Age	Country	Date
Dopplereffect with Smartphones (Phyphox)	>14	Germany	Mai 2022

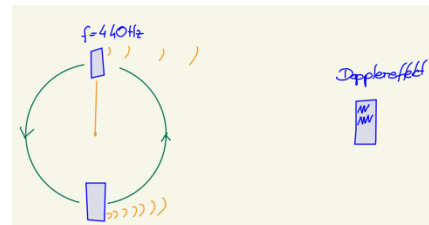
## Dopplereffect with a Smartphone

### Experimental setup materials:

2 Smartphones

1 case

1 rope



### Experimental setup and procedure:

Start the **Phyphox App** on your first smartphone and setup the experiment „Tone generator“. Mount the smartphone in the case and fix the rope to your case. Play the frequency 440Hz with the phyphox App. Do movements of circles with the rope and smartphone over your head. Try to circle in a constant angular velocity.

**Pay attention, nobody should stand next to you!**

The other students have to stand 3 m away from you, while they are listening to the tone !

### Experiment evaluation:

- 1) Describe, what you hear in a distance of 3 m!
- 2) Explain what you hear, at the part of the circle, where the smartphone moves in your direction. Explain the connection of the wavelength you hear  $\lambda_E$  and the velocity of the smartphone  $v_{Smart}$ !

- 3) Measure the frequency, you hear  $f_E$  with a second smartphone and the program „frequency“ in the Phyphoxapp . The frequency of the smartphone is called  $f_s$  .

Calculate the speed of the Smartphones  $v_{Smart}$ , with the help of Doppler-formula:

$$f_E = f_s \cdot \frac{1}{1 \pm \frac{v_{Smart}}{v_{Schall}}}$$

( $v_{schall}$ = 342m/s (Speed of Sound))

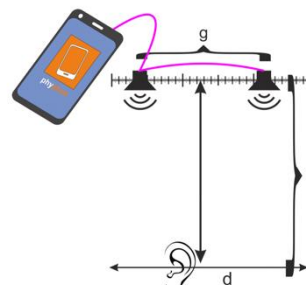
- 4) Open the program „Dopplereffect“ on your second Smartphone. Write 440Hz in the settings and click to *results*. Now you can measure the shift in wavelength directly; you can also compare the speed you have calculated with the speed which shown in the phyphox App.



Topic	Age	Country	Date
Interference with two smartphones (Phyphox)	>14	Germany	Mai 2022

### Experimental materials:

- 2 smartphones/ tablets
- 1 tape measure
- 1 duct tape



### Experimental setup and procedure:

Start the **Phyphox App** on both smartphones and open the experiment „Tongenerator“. Fix them in a horizontal position, so that the speaker points towards you. The distance between both speakers has to be 0,80 m.

Start the frequency of 440Hz on both smartphones. Close one of your ears with your hand and go slowly to a distance of  $a=3,00$  m along line d.

### Experiment evaluation:

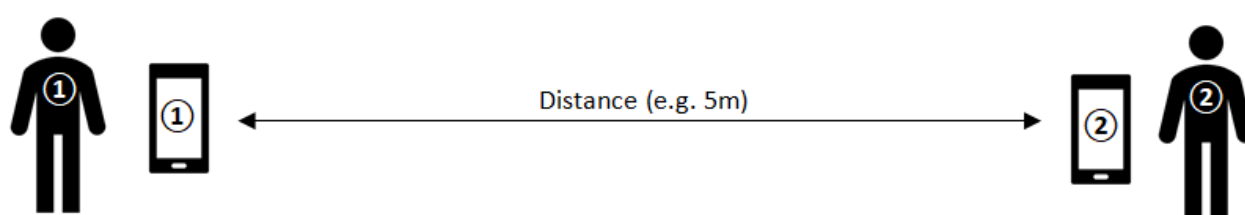
- 1) Describe, what you hear while you are walking line d.
- 2) How do the positions of the accoustic maxima change, if you use different frequencies?
- 3) Find out a connection for the wavelength  $\lambda$  and the distances of the positions of the maxima  $d_k$ .
- 4) The frequency is defined as:

$$f = \frac{c_L}{\lambda} = \frac{343 \frac{m}{s}}{\lambda}$$

- 5) Find a connection between the distance of the speakers b and the positions of the accoustic maxima  $d_k$ .
- 6) Experiment with the distances of the speakers b, what do you observe especially for short distances?
- 7) Can you find a set-up with positions of total silence?

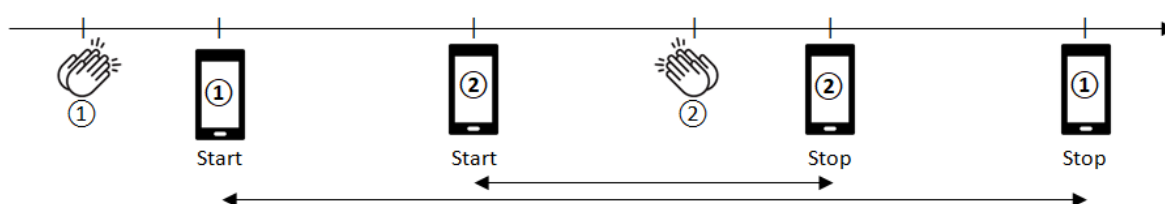
Topic	Age	Country	Date
Speed of sound (Phyphox)	>14	Germany	May 2022

- 2 Smartphones
- 2 persons
- Tape measure



In this experiment you can approximate the speed of sound. 2 persons have a certain distance (tape measure!) from each other (e.g. 5m).

For this experiment you need the acoustic stopwatch of “Phyphox” to get the time between two acoustic events. The first signal causes from the clapping of the first person. The second signal by clapping of the second person stops the clock running.

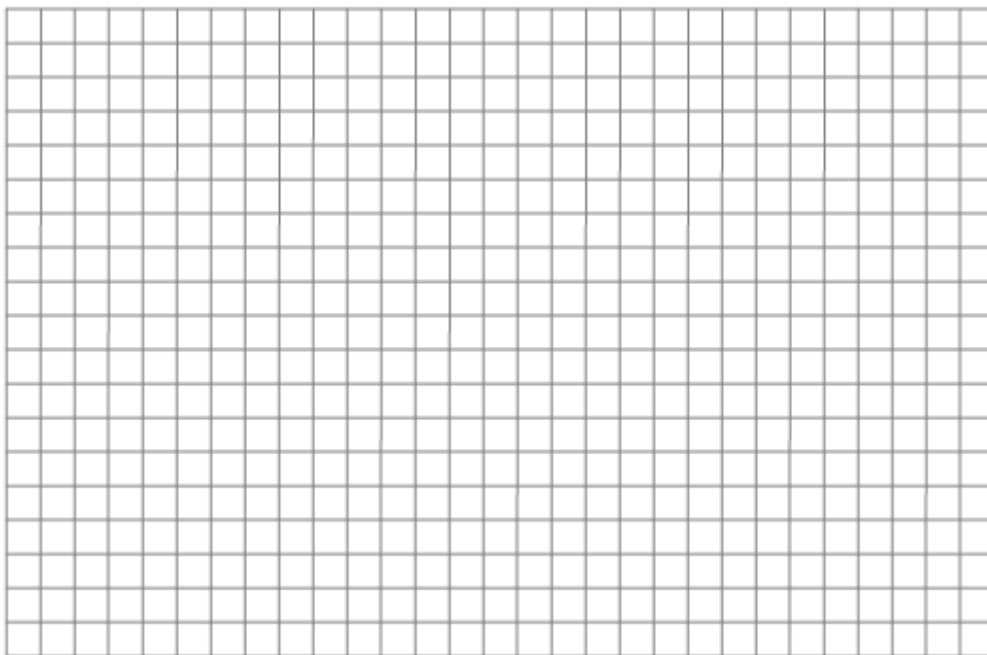
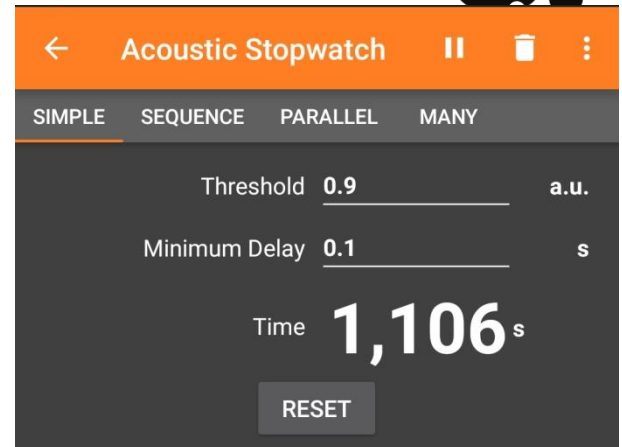


At the end you have to subtract the time of smartphone 2 from the time of smartphone 1. So you get the time for the double distance of the sound.



It is very important to put the threshold to such a level, that the noise from the area doesn't disturb your measure.

Now we can calculate the speed of sound by using the distance  $d$  and the time difference  $\Delta t$  :



Calculated value:  $c =$

Theoretical value:  $c =$



Topic	Age	Country	Date
Standing waves on a string	>14	Italy	May 2022

## Equipment

String Vibrator
Physics String
Force Sensor
Tape Measure
PASCO Capstone

**The general appearance of waves can be shown by means of standing waves in a string.**

This type of wave is very important because most of the vibrations of extended bodies, such as the prongs of a tuning fork or the strings of a piano, are standing waves.

**The purpose of this experiment is to examine how the tension required to produce a standing wave in a vibrating string of fixed length and mass density is affected by the wavelength and the frequency of the wave.**

## General Theory

**Standing waves (stationary waves) are produced by the addition of two traveling waves, both of which have the same wavelength and speed, but travel in opposite directions through the same medium.**

Figure 1 shows such a system, where a mechanical vibrator produces a wave on a string which moves to the right and reflection from a fixed end produces a left moving wave. Where the two waves are always 180° out of phase, very little motion occurs (none if the amplitudes are the same). Such places are called **nodes** (see Figure 1).

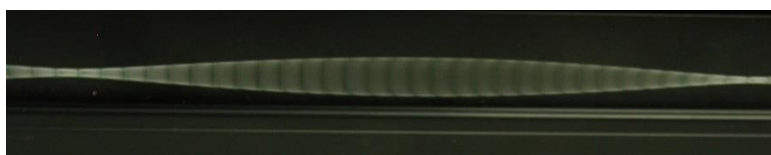


Figure 1: Standing Wave Photo

Where the two waves are in phase, the motion is maximum. These positions are called anti-nodes. Figure 2 shows a representation of a standing wave.

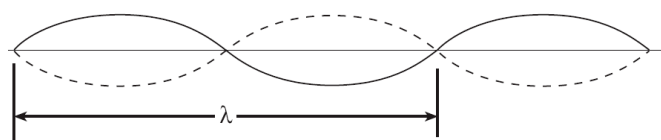


Figure 2: Standing Wave Representation



This representation shows the two extreme positions of the string.

Note that as shown in Figure 2, **the node-to-node distance is one-half of the wavelength.**

**The necessary conditions for the production of standing waves on a stretched string fixed at both ends is the length of the string be equal to an integer number of half wavelengths so that there can be a node at each fixed end.**

For this experiment, one fixed end is where the string attaches to the force sensor and the other is where the string attaches to the mechanical vibrator. The end attached to the force sensor is a true node, but the end attached to the metal wand on the vibrator is not exactly a node since the wand vibrates up and down a little. Close examination (see Figure 3) shows that the true node would be a little to the left of the knot so the effective string length will be a bit longer than we measure. However, the difference does not appear to be more than a few millimeters, so is only a fraction of a percent.

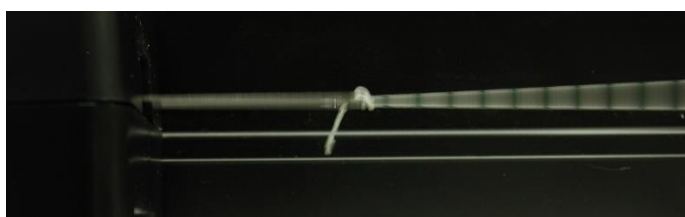


Figure 3: Vibrator ~ Node

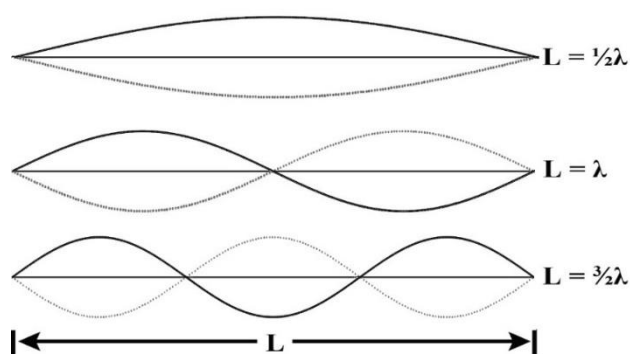


Figure 4: Modes of Vibration

## Theory: Standing Waves in Strings

A stretched string has many natural modes of vibration (three examples are shown in Figure 4 above).

If the string is fixed at both ends then there must be a node at each end.

It may vibrate as a single segment, in which case the length ( $L$ ) of the string is equal to  $1/2$  the wavelength ( $\lambda$ ) of the wave.

It may also vibrate in two segments with a node at each end and one node in the middle; then the wavelength is equal to the length of the string.

It may also vibrate with a larger integer number of segments.

**In every case, the length of the string equals some integer number of half wavelengths.**

For any wave with wavelength  $\lambda$  and frequency  $f$ , **the speed,  $v$** , is

$$v = \lambda f \quad (1)$$

The speed of a wave on a string is also related to the tension in the string,  $F$ , and the linear **density** (=mass/length),  $\mu$ , by

$$v^2 = F/\mu \quad = \lambda^2 f^2 \quad (2)$$

**$L$  is the length of the string and  $n$  is the number of segments.** (Note that  $n$  is *not* the number of nodes). Since a segment is  $1/2$  wavelength then



$$\lambda = 2L/n \quad \text{where } n = 1, 2, 3, \dots \quad (3)$$

Solving Equation 2 for the **tension** yields:

$$F = \mu \lambda^2 f^2 \quad (4)$$

**In the first part of the experiment, we will hold  $\lambda$  constant** by always choosing a two segment pattern so that  $\lambda = L$  since  $n = 2$  and vary the frequency while measuring the tension at which a two segment standing wave appears. By plotting  $F$  versus  $f^2$ , we should see a straight line with a slope of  $\mu \lambda^2$ .

**In the second part, we hold the frequency constant** and vary the tension (Figure 4) to get standing waves with different numbers of segments, so different values of  $\lambda$ . Plotting  $F$  versus  $\lambda^2$  should give a straight line with slope  $\mu f^2$ .

## String Mass Density

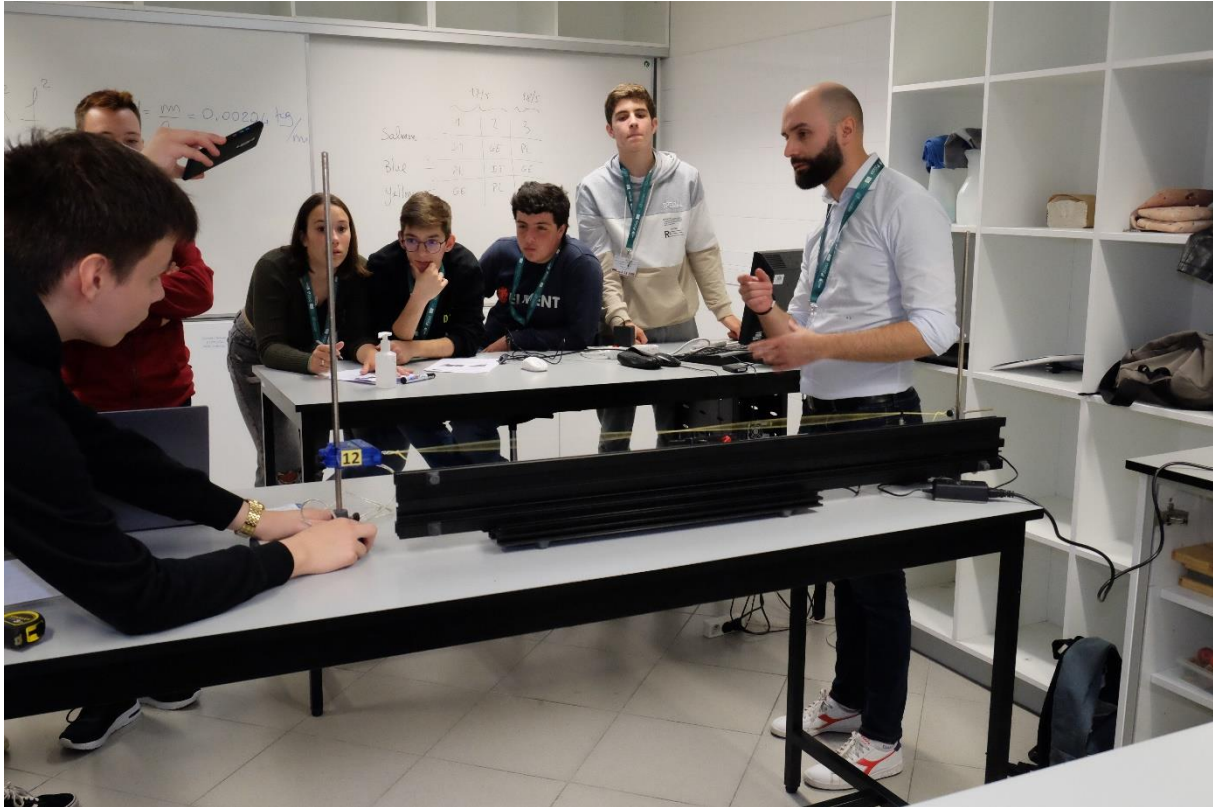
Each lab group will use a piece of string about one meter long.

1. Use the tape measure to determine the length of the string to a precision of  $\pm 1$  mm.
2. Determine the mass of the string using a single pan balance. The balance precision is 0.1 g. Record the mass and calculate the mass/length.

## Setup

1. Setup as shown in Figure 5. Clamp the String Vibrator firmly to the table. Attach two banana cables from Universal Interface to the Inputs on the String Vibrator. Polarity does not matter.
2. Attach the Force Sensor
3. Tie the 1 m string to the hook on the Force Sensor and to the vibrating blade on the String Vibrator.
4. The oscillation of the string is all in a vertical plane. It is nice to provide a dark backdrop so the student can see the pattern better.





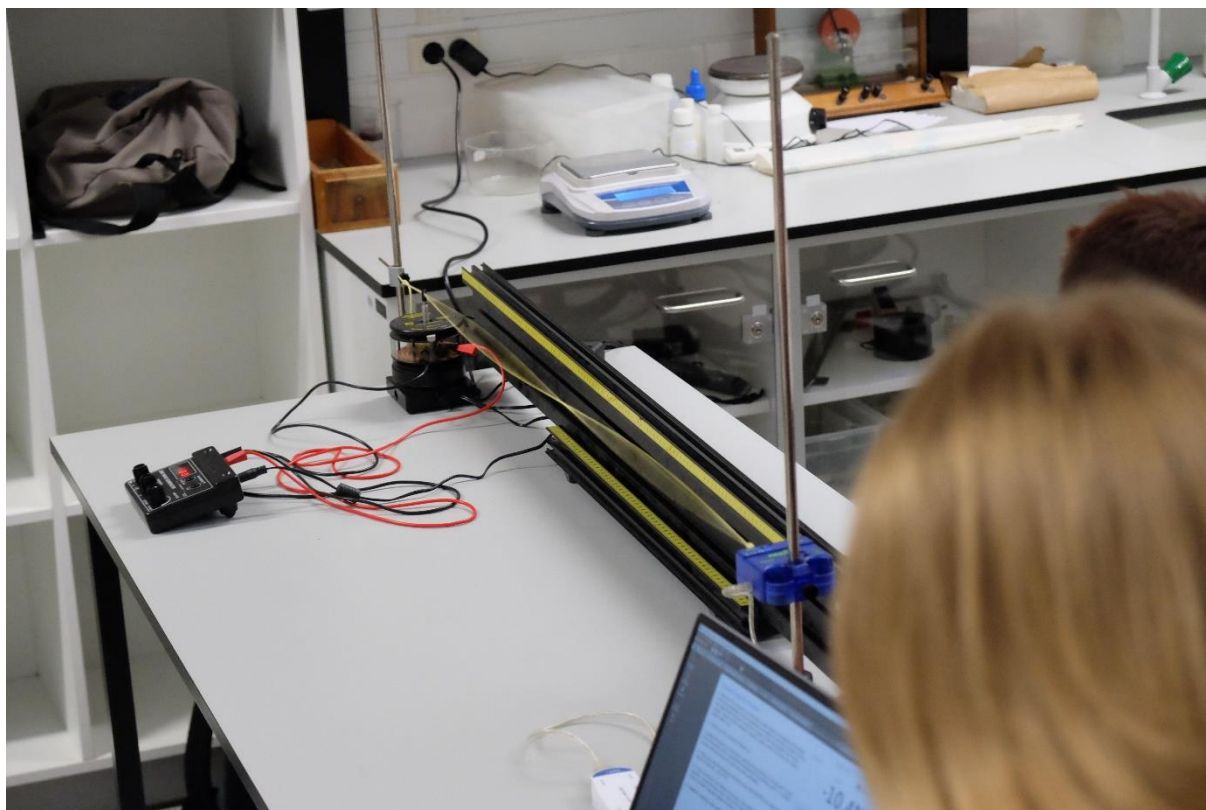


Figure 5: Standard Setup

5. In PASCO Capstone, create a Digits Display and select the Force. Then turn on the Statistics with the Mean selected.
6. Create a table as shown. Create User-Entered Data sets called "Frequency" with units of Hz and "Tension" with units of N.
7. Set the sample rate to 20 Hz.

Frequency (Hz)	Tension (N)
60	0,72
85	1,49
104	2,16
120	2,86
134	3,62

Table 1

## CONSTANT WAVELENGTH PROCEDURE

1. Move the Force Sensor **so the string is under tension and determine the length of the string** between the knot attached to the blade on the String Vibrator and the knot attached to the hook on the Force Sensor. **We will adjust for two segments (as in Figure 5)** for each case, so the knot to knot distance will be the wavelength. Record your value.
2. Click on the Signal Generator at the left of the screen.
3. Click On to turn the String Vibrator on.
4. Adjust the tension in the string to get the string vibrating in two segments (as in Figure 5) and to maximize the amplitude of the standing wave. It is easiest to start with the tension too high and slowly decrease it.
5. When you have the maximum standing wave, click RECORD and collect data for about 10 seconds. Click STOP.



6. The force shown in the Force digits display is the average value over the 10 seconds. Ignore the minus sign. Record this value in the “Tension” column of the table in the 60 Hz row.
7. Repeat for frequencies of 85 Hz, 104 Hz, 120 Hz, 134 Hz
8. Question: Why do we have this rather strange looking choice of frequencies? Hint: See the graph in the Analysis section.

## Constant Wavelength Analysis

1. Create a graph of Tension vs. frequency with the data of Table 1. Use the QuickCalcs on the frequency axis to linearize the data.
2. Based on Equation 4, what should be the slope and y-axis intercept of a Tension vs. frequency<sup>2</sup> plot?
3. Select a Linear fit.
4. How well do the experimental slope and intercept from step 4 agree with the value you calculated for slope and intercept in step2? Discuss briefly. What does this show?

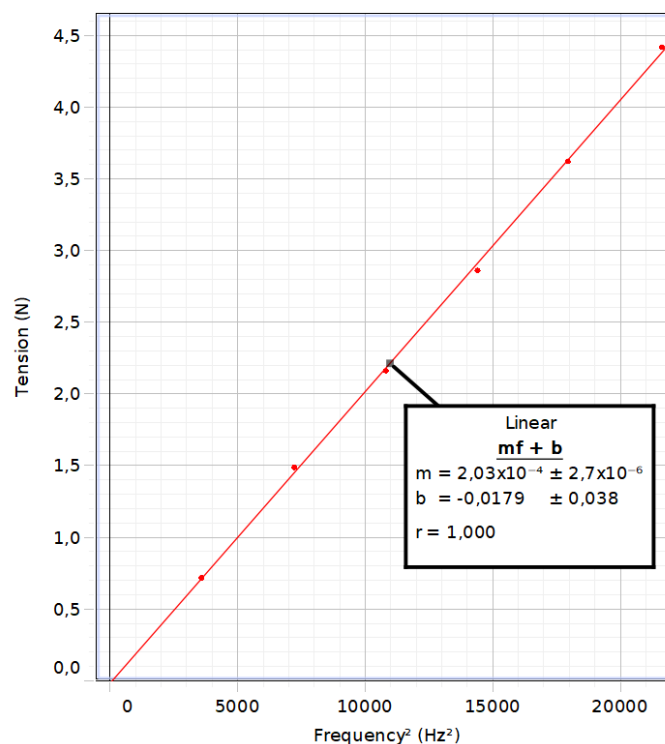


Figure 6: Constant wavelength

*Agreement is perfect! Data fits a straight line. Slope and intercept agree with prediction within the uncertainties. Shows the equations from Theory (particularly Equation 1 and 2) are valid.*



## CONSTANT FREQUENCY PROCEDURE

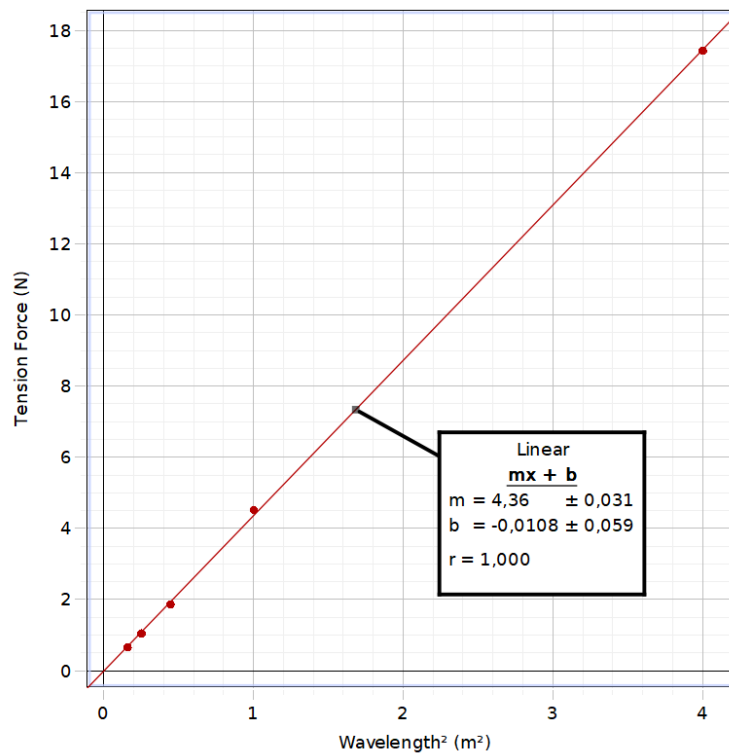
1. Create a table as shown. Create User-Entered Data sets called “# of Segments”, “String Length” with units of m, and “Tension Force” with units of N. Then open the Capstone calculator and make the calculation with units of m:  $\text{Wavelength} = 2 * [\text{String Length (m)}] / [\# \text{ of segments}]$

N. of Segments	String Length (m)	Wavelength (m)	Tension Force (N)
1	1	2,00	17,41
2	1	1,00	4,51
3	1	0,67	1,87
4	1	0,50	1,04
5	1	0,40	0,66

2. Move the Force Sensor so the string is under tension and determine the length of the string between the knot attached to the blade on the String Vibrator and the knot attached to the hook on the Force Sensor. Enter this length in the “String Length” column of the table in each of the five rows (same value in each row).
3. Click on the Signal Generator at the left of the screen.
4. Click On to turn the String Vibrator on.
5. Adjust the tension in the string to get the string vibrating in one segment (nodes only at the ends) and to maximize the amplitude of the standing wave. It is easiest to start with the tension too high and slowly decrease it.
6. When you have the maximum standing wave, click RECORD and collect data for about 10 seconds. Click STOP.
7. The force shown in the Tension Force box is the average value over the 10 seconds. Ignore the minus sign. Record this value in the “Tension Force” column of the table in the first row.
8. Repeat for standing wave patterns with 2, 3, 4, and 5 segments.

## Constant Frequency Analysis

1. Create a graph of Tension Force vs. Wavelength. Use the QuickCalcs on the frequency axis to linearize the data.



2. Based on Equation 4 from Standing Wave Theory, what should be the slope and y-axis intercept of the Tension vs. wavelength<sup>2</sup> plot?.
3. Click on the Scale-to-Fit icon from the graph toolbar and select a Linear Fit.
4. How well do the experimental slope and intercept from step 3 agree with the value you calculated for slope and intercept in step 2? Discuss briefly. What does this show?

*Agreement is perfect! Data fits a straight line. Slope and intercept agree with prediction within the uncertainties. Shows the equations from Theory (particularly Equation 1 and 2) are valid.*





Topic: Waves	Age	Country	Date
Math and Music	>14	Poland	May 2022

## Function, realisation

Some sound sources generate very pure tones (e.g. sound fork). The sounds generated by musical instruments are composed of many tones of different frequencies. The task of the experiment is to find the peak frequency (using Fourier transform FFT). To also get the timbre of the instrument would require extraction of the overtones as well.

The aim of the experiment is to show that the frequencies of the octave sound form a geometric sequence with the  $q=2^{(1/12)}$ .

## Hardware required

- Vernier Analogue Microphone sensor MCA-BTA
- Labquest mini interface (microcontroller).
- PC or Mac computer
- sound sources: sound fork, guitar, flute, piano
- calculator
- tone generator mobile app
- 

## Materials required

- paper for note

## Software required

- Logger Pro software from Vernier



## Setup Hardware



We must connect microphone plug to selected 1 of 3 available analog sockets in Labquest mini microcontroller. Then we must connect Labquest mini with USB cable to PC or Mac.

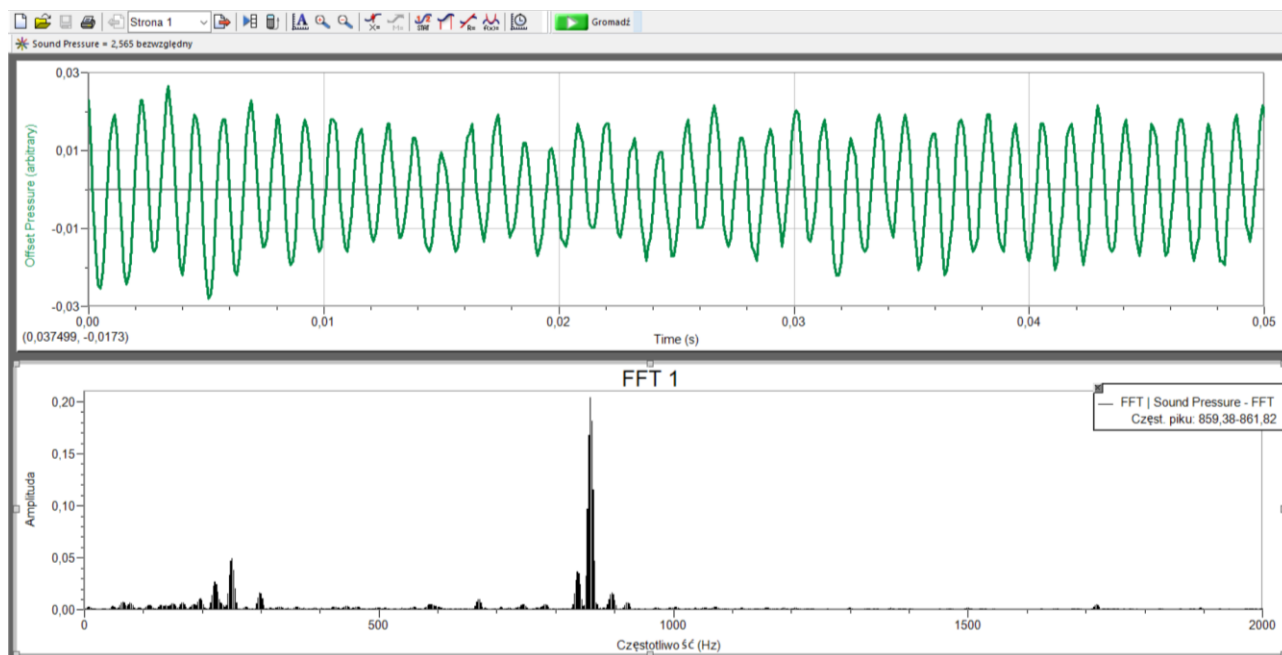




## Setup software

For starting experiment we must find and open experiment file  
Mathemattics and Musics (from the set Physics with Vernier)

For start measure frequency of the sound we must click green button  
(see illustration below)



## Tasks- testing steps

- measure tone frequency generated by sound fork
- finding find the peak frequency of the sound generated by different musical instruments: guitar, flute, piano (the same note)
- testing frequency of the sound generated by mobile app.

After testing the operation of the system on various instruments and sound sources, we move on to finding the relationship between frequency and note pitch.



## Experiment Evaluation

System testing:

-measure tone frequency generated by sound fork

Write down Your results

Nr	key	Frequency Hz	ratio to previous note	ratio to A4
1	A4	440	-----	1
2	A4#			
3	H4			
4	C4			
5	C4#			
6	D4			
7	D4#			
8	E4			
9	F4			
10	F4#			
11	G4			
12	G4#			
13	A5			

### Go further

We can do the same using Python script (example given <https://pythonnumericalmethods.berkeley.edu/notebooks/chapter24.04-FFT-in-Python.html>), but the code is a bit long, and not all is easy to understand by all students. This code use Numpy, Pandas and Matplotlib libraries.



Topi: Waves	Age	Country	Date
Principle of operation of the RFID system	>14	Poland	May 2022

## Function, realisation

Build experimental system which provide testing tags, cards and reading information stored inside RFID tags, which use radio waves

## Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18 or above
- Cytron Maker Pico docking station for Pico (or different docking station for Pico: Waveshare, Seeedstudio)
- RFID-RC522 card reader
- Mifare 13,56kHz cards, tags
- PC or Mac computer

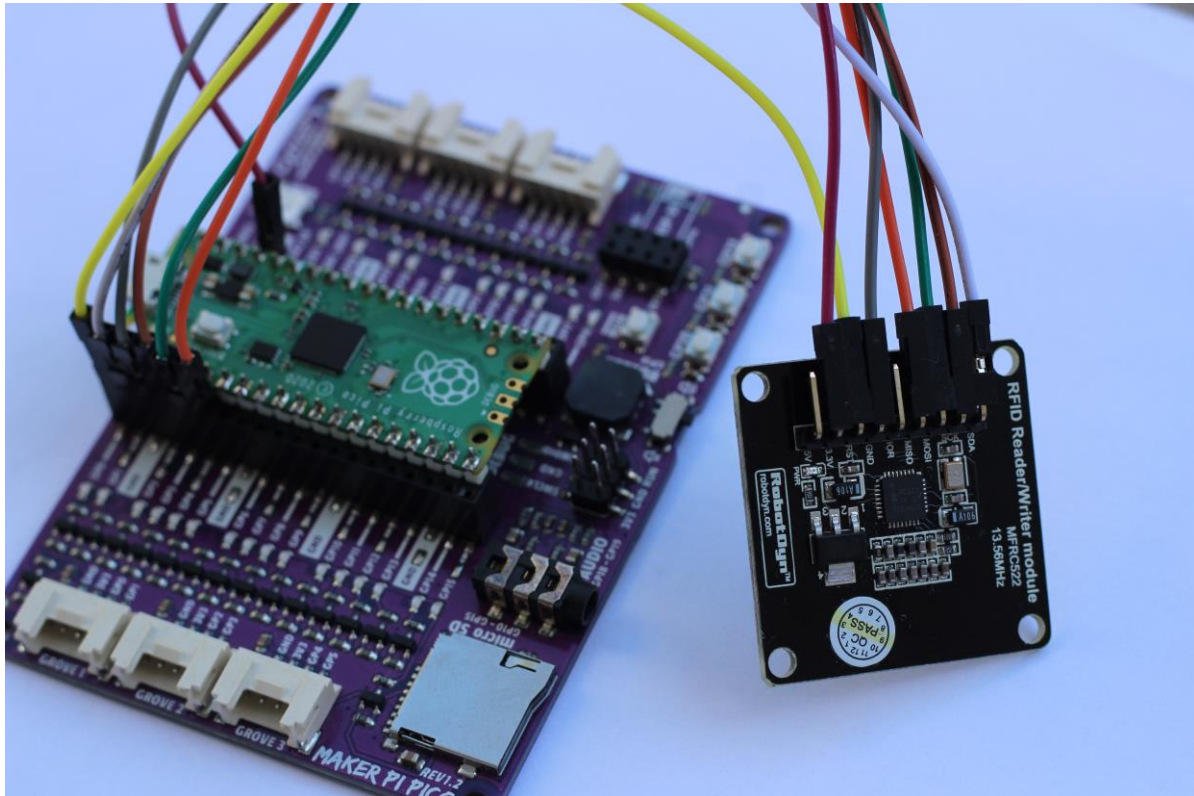
## Materials required

- 7 wires in different colours (Male-Female DuPont wire)
- micro usb 2.0 high speed (15cm or 30 cm)
- paper for note

## Software required

- Thonny (thonny.org)
- library for RFID <https://github.com/danjperron/micropython-mfrc522/blob/master/mfrc522.py>
- script [https://github.com/danjperron/micropython-mfrc522/blob/master/Pico\\_example/Pico\\_read.py](https://github.com/danjperron/micropython-mfrc522/blob/master/Pico_example/Pico_read.py)

## Setup Hardware



RFID reader use serial communication in **SPI standard**, so we need a bit more wires then in I2C case.

Reader Pin	Raspberry Pi Pico	GPIO ESP32	GPIO ESP8266
sck	<b>GP2</b>	18	0
mosi	<b>GP3</b>	23	2
miso	<b>GP4</b>	19	4
rst	<b>GP0</b>	22	5
cs (SDA)	<b>GP1</b>	21	14

The table also contains information on how to connect to other popular microcontrollers (ESP32, Esp8266)

The reader use 3.3V power (not 5!!!). Usually we connect red wire to pin 3V3 and black wire to the ground (GND)



## Setup software

The file `mfrc522.py` which contain library for the reader we must put to `lib` folder on our Pico microcontroller. We can do this with Thonny software.

The script for testing

```
1 from mfrc522 import MFRC522
2 import utime
3 def uidToString(uid):
4     mystring = ""
5     for i in uid:
6         mystring = "%02X" % i + mystring
7     return mystring
8 reader = MFRC522(spi_id=0,sck=2,miso=4,mosi=3,cs=1,rst=0)
9 print("")
10 print("Please place card on reader")
11 print("")
12 PreviousCard = [0]
13 try:
14     while True:
15         reader.init()
16         (stat, tag_type) = reader.request(reader.REQIDL)
17         #print('request stat:',stat,' tag_type:',tag_type)
18         if stat == reader.OK:
19             (stat, uid) = reader.SelectTagSN()
20             if uid == PreviousCard:
21                 continue
22             if stat == reader.OK:
23                 print("Card detected {} uid={}".format(hex(int.from_bytes(bytes(uid),"little",False)).upper(),reader.tohexstring(uid)))
24                 defaultKey = [255,255,255,255,255,255]
25                 reader.MFRC522_DumpClassic1K(uid, Start=0, End=64, keyA=defaultKey)
26                 print("Done")
27                 PreviousCard = uid
28             else:
29                 pass
30         else:
31             PreviousCard=[0]
32             utime.sleep_ms(50)
33 except KeyboardInterrupt:
34     pass
35
```

```
MicroPython v1.18 on 2022-01-17; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT
```

Please place card on reader

```
Card detected 0XFB862396 uid=[0x96, 0x23, 0x86, 0xFB]
00 S00 B0: Authentication error
Done
Card detected 0XFB862396 uid=[0x96, 0x23, 0x86, 0xFB]
00 S00 B0: Authentication error
Done
Card detected 0X59C01E0D uid=[0x0D, 0x1E, 0xC0, 0x59]
00 S00 B0: 0D 1E C0 59 8A 08 04 00 62 63 64 65 66 67 68 69 ...Y....bcdefg
hi
01 S00 B1: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
..
02 S00 B2: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
..
```

## Terminal window with results of reading



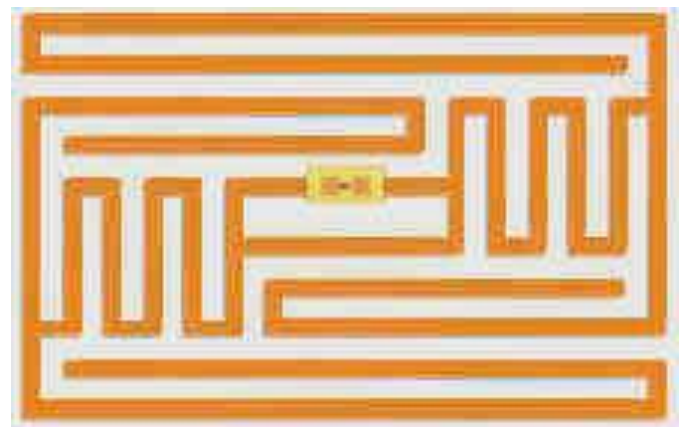
## The principle of operation of RFID

### RFID (Radio-frequency identification)- introduction

A technology that uses radio waves to transmit data and power the electronic circuit (RFID tag) of an object tag by a reader to identify the object. The technique makes it possible to read and sometimes also write an RFID chip. Depending on the design, it allows you to read labels from a distance of up to several dozen centimetres or several meters from the reader antenna.

The system operation is as follows:

the reader uses the transmitter antenna to generate an electromagnetic wave, the same or the second antenna receives electromagnetic waves, which are then filtered and decoded so as to read the responses of the



tags.

Passive tags do not have their own power supply, when they are in the electromagnetic field with the resonant frequency of the receiving system, they collect the received energy in a capacitor contained in the tag's structure. Upon receipt of sufficient energy, a reply containing the marker code is sent.

Popular standards (in our case 13.56 kHz compatible with the Mifare standard)

125kHz, 13.56kHz and others

image source: wikipedia.org

(Mifare also supports ATM cards, tickets)



## Experiment Evaluation

System testing:

- trying to read different receivers: identification cards, tags;
- practical check of the range of tags and reader
- learning about the information sent to the reader
- checking the effectiveness of security measures against card cloning

Write down Your results

card/ tag	data sentt from card	card/tag	effective distance





Topic: Waves	Age	Country	Date
Sound level meter	>14	Poland	May 2022

## Function, realisation

Construction of a circuit based on microcontroller and Python that measures the sound level

## Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18 or above
- Cytron Maker Pico docking station for Pico
- Dfrobot Level meter module (analog 3.3V for Pico, 5V for Arduino)
- PC or Mac computer

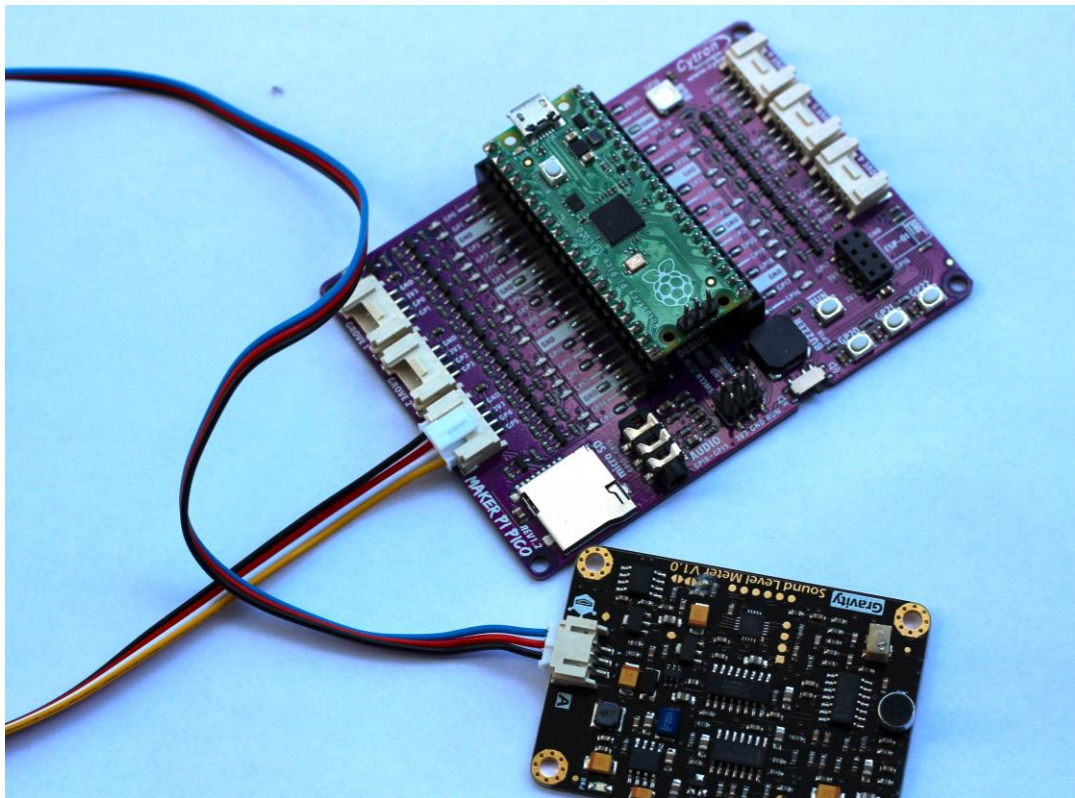
## Materials required

- Grove male wire (or 3x Dupont male-male wire )
- micro usb 2.0 high speed (15cm or 30 cm)
- paper for note

## Software required

- Thonny (thonny.org)
- Sound level meter doesn't require custom library for Micropython
- LCD 1602 display library:  
[https://files.seeedstudio.com/wiki/Grove\\_Shield\\_for\\_Pi\\_Pico\\_V1.0/Libraries.rar](https://files.seeedstudio.com/wiki/Grove_Shield_for_Pi_Pico_V1.0/Libraries.rar)

## Setup Hardware



Because **Sound Level Meter** sensor is **analog**, then we must connect them to GP26, GP27 or GP28- analog ports of Raspberry Pi Pico. In our case the best way is select GP27 or GP26 (we have grove socket on the shield Maker Pico connected to these pins). Below

Sensor pins	Raspberry Pi Pico pins	Dupont	Grove wire
signal (A)	<b>GP27</b>	blue	<b>yellow</b>
power (+)	<b>3V3</b>	red	<b>red</b>
ground (-)	<b>GND</b>	black	<b>black</b>

The Pico use 3.3V power (not 5!!!). Usually we connect red wire to pin 3V3 and black wire to the ground (GND)

## Setup software



The script doesn't use special libraries, but only standard libraries from Micropython, custom libraries not required.. The script is very simple, very easy to understand, like scripts, which read data from analog sensors.

Python script for measure sound level which use Analog Digital Converter (12-bit ADC)

```
import utime
import machine
# Dfrobot Sensor SoundLevelmeter connected to ADC1 (GP27)
# other options: pin26- ADC0, pin28- ADC2
analog_value = machine.ADC(1)
#conversion_factor = 3.3/ 65535
# in Micropython ADC can give us 65535 levels (2^16-1)
while True:
    sound_lev = analog_value.read_u16()
    print("raw:", sound_lev)
    #print(" minivolt:", minivolt)
    voltageValue = sound_lev / 65535.0
    print(voltageValue,"V")
    dbValue = voltageValue * 50.0 * 3.3
    #if we use other microcontroller, which use 5V, then we must change 3.3 to 5.
    #Pico: only 3.3V
    print(dbValue,"dB")
    utime.sleep(0.5)
#You can change sleep time to get better results
```

**You can use Shall window, to observe results or change the script and plot results in Plotter window (View/Plotter).**

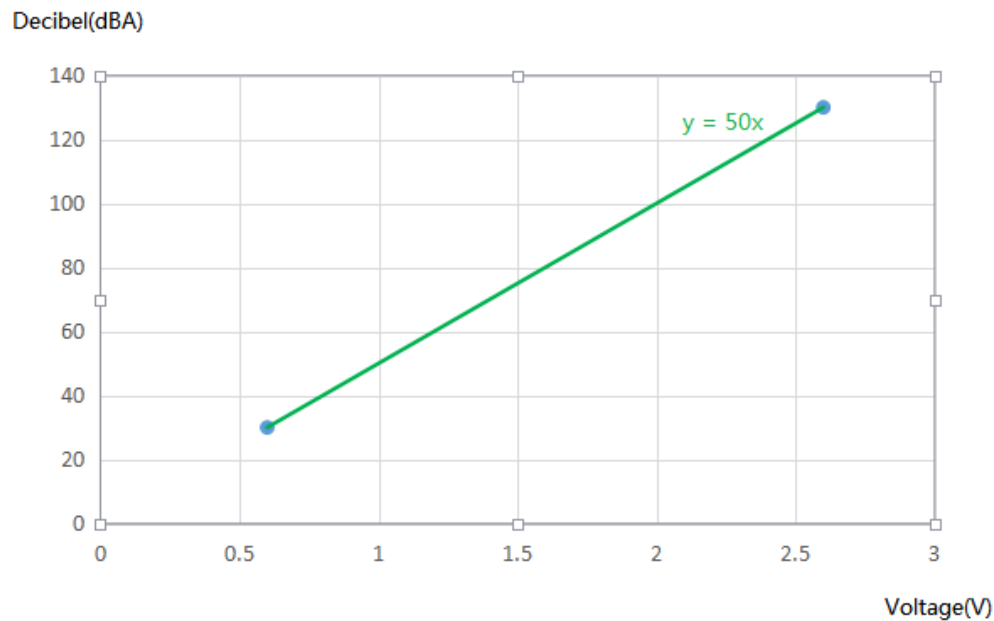
## The principle of operation of Sound Level Meter

Sound level meter (also known as the decibel meter, noise meter) is a basic noise measurement instrument. We have launched a sound level meter, which is compatible with Arduino, and is plug-and-play. It can accurately measure the sound level of its surrounding environment. This product uses an instrument circuit and a low noise microphone, which makes it highly precise. It supports 3.3-5.0V wide input voltage, 0.6-2.6V voltage output. The decibel value is linear with the output voltage, which leads to a simple conversion, and therefore does not need a complex algorithm. The connector is plug-and-play so this product can be easily used in your application.

## Relation between Decibel Value and Voltage Output



For this product, the decibel value is linear with the output voltage. When the output voltage is 0.6V, the decibel value should be 30dB. When the output voltage is 2.6V, the decibel value should be 130dB. The calibration is done before leaving the factory, so there is no need to calibrate it. So we can get this relation: Decibel Value(dB) = Output



Voltage(V)  $\times$  50, as shown below.



## Experiment Evaluation

System testing:

- Measure the loudness level in various situations: street noise, library, school corridor, scream, musical instruments
- Compare the obtained results with the results of the mobile application
- Change the delay (sleep) and observe the effect on the results
- Change the Python script to plot graph based on results (using internal ploter from Thonny App)

Write down Your results

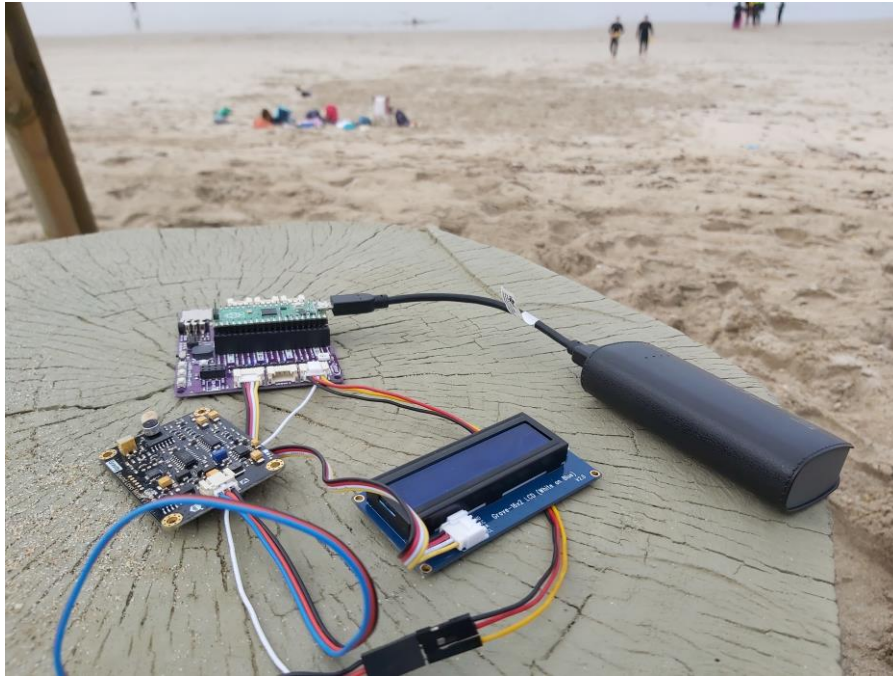
situation	result (dba)	situation	result (dba)

### Go further

- We can add to the script conditional instruction **IF**, which can display additional information depend on sound level,: level like on the street, level like in library, etc.
- To compare results, we can use free of charge mobile App from Google Play store: **Sound meter**.
- Supplementing the code with the operation of the LCD display
- Building a mobile version (with UPS) adding results logging to the microcontroller memory.



In the photo below, an autonomous system made by a Polish team that measures the sound level on the beach in Vila do Conde (Portugal)  
The script was saved in Pico's memory under the name **main.py**.



Measure the sound level using a mobile application:





Topic: Waves  Simple harmonic motion with ultrasonic sensor	Age  >14	Country  Poland	Date  May 2022
---	----------------	-----------------------	----------------------

## Function, realisation

Construction of a circuit based on microcontroller and Python that measures distance from the sensor to the mass of the spring. Plotting the position in harmonic motion on the basis of indications of the distance from the ultrasonic sensor.

## Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18
- Cytron Maker Pico docking station for Pico
- ultrasonic ranger HC-SR04P (special version for 3.3 V microcontrollers, we don't need to use resistor like in the case of standard HC-SR04 )
- PC or Mac computer

## Materials required

- Grove male wire (or 4x Dupont male-male wire )
- micro usb 2.0 high speed (15cm or 30 cm)
- mass
- stand
- spring
- paper for note

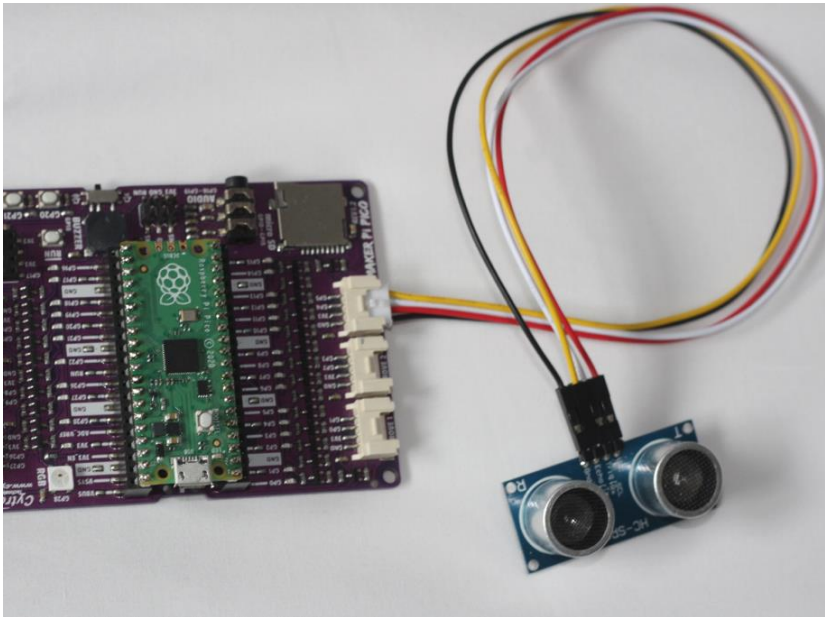
## Software required

- Thonny (thonny.org)
- micropython library <https://github.com/rsc1975/micropython-hcsr04>

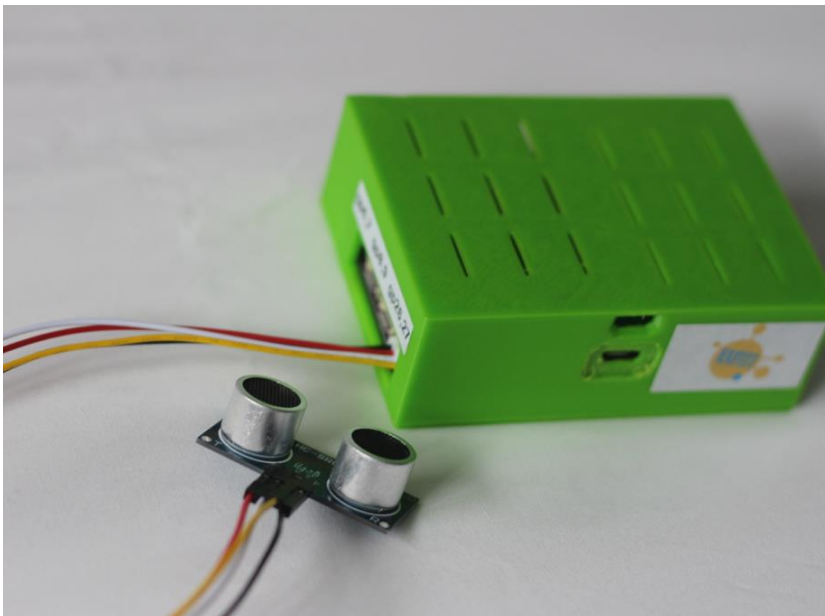




## Setup Hardware



Below the same system, but with our 3D printed housing



In our Python script, we assume that the ultrasonic sensor HC-SR04P will use two signal pins: GP26, GP27



Sensor pins	Raspberry Pi Pico pins	Grove
Echo	GP27	yellow
Trig	GP26	white
power (+)	3V3	red
ground (-)	GND	black

The Pico uses 3.3V power (not 5!!!). Usually we connect red wire to pin 3V3 and black wire to the ground (GND)

## Setup software

Before use of the system we must install libraries from the site:

<https://github.com/rsc1975/micropython-hcsr04>

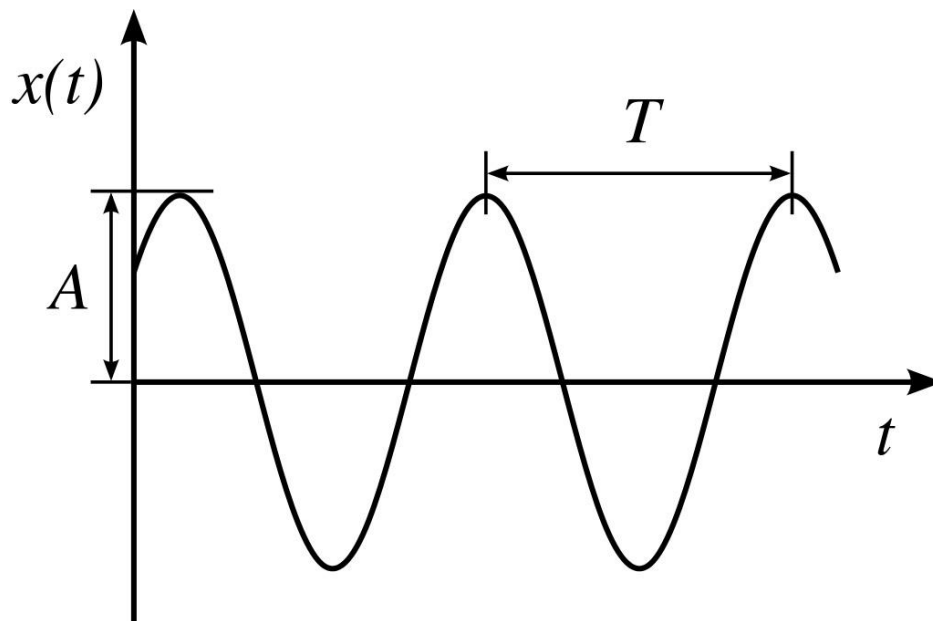
We must put the file hcsr04.py to folder with the name lib on our Raspberry Pi Pico. We can do this with Thonny software.

The script is very simple, very easy to understand and we can use 2 methods from library: **distance\_cm** or **distance\_mm**.

```
1 # Harmonic motion with HC-SR04P and Raspberry Pi Pico
2 # based on the methods from library
3 # https://github.com/rsc1975/micropython-hcsr04
4 from hcsr04 import HCSR04
5 from utime import sleep
6
7 sensor = HCSR04(trigger_pin=26, echo_pin=27, echo_timeout_us=1000000)
8 while True:
9     distance = sensor.distance_cm()
10    #print('Distance:', distance, 'cm')
11    print(distance)
12    sleep(0.05)
```

To look for the graph of distance over time, we must select from Thonny's menu **View/plotter**.

**We can compare our experimental plot with the picture below**



## Displacement in Simple harmonic motion

Simple harmonic motion- type of periodic motion where the restoring force on the moving object is directly proportional to the magnitude of the object's displacement and acts towards the object's equilibrium (medium) position. It results in an oscillation which continues indefinitely, if

$$x(t) = A \cos(\omega t - \varphi),$$

uninhibited by friction or any other dissipation of energy.  
 $x$  is its displacement from the equilibrium (or mean) position  
 $A$ -amplitude.  
 $\omega$ = .....  
 $f$ = .....

## Experiment Evaluation

System testing:

- measure displacement the mass from mean position
- comparison results of different mass and different springs
- change the delay (sleep) and observe the effect on the results



- change the Python script to plot graph based on results (using internal ploter from Thonny App)

Write down Your results:

.....

.....

.....

.....

.....

.....

.....

.....

.....

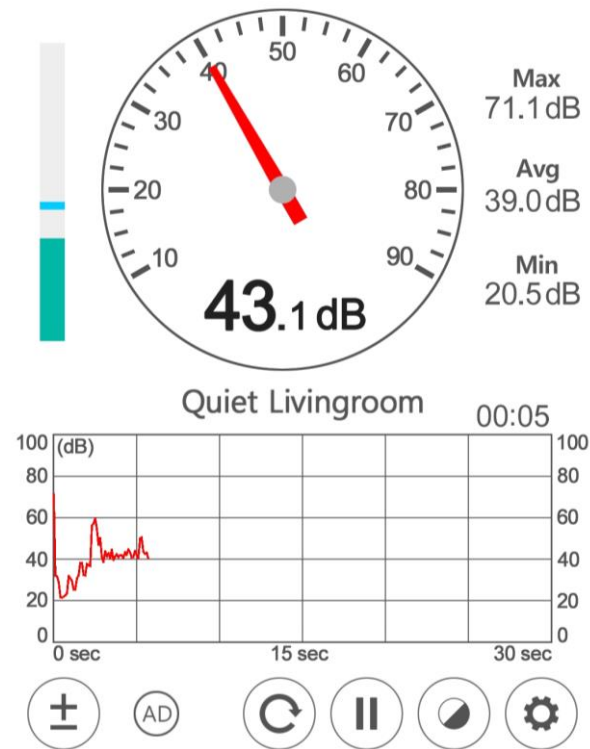
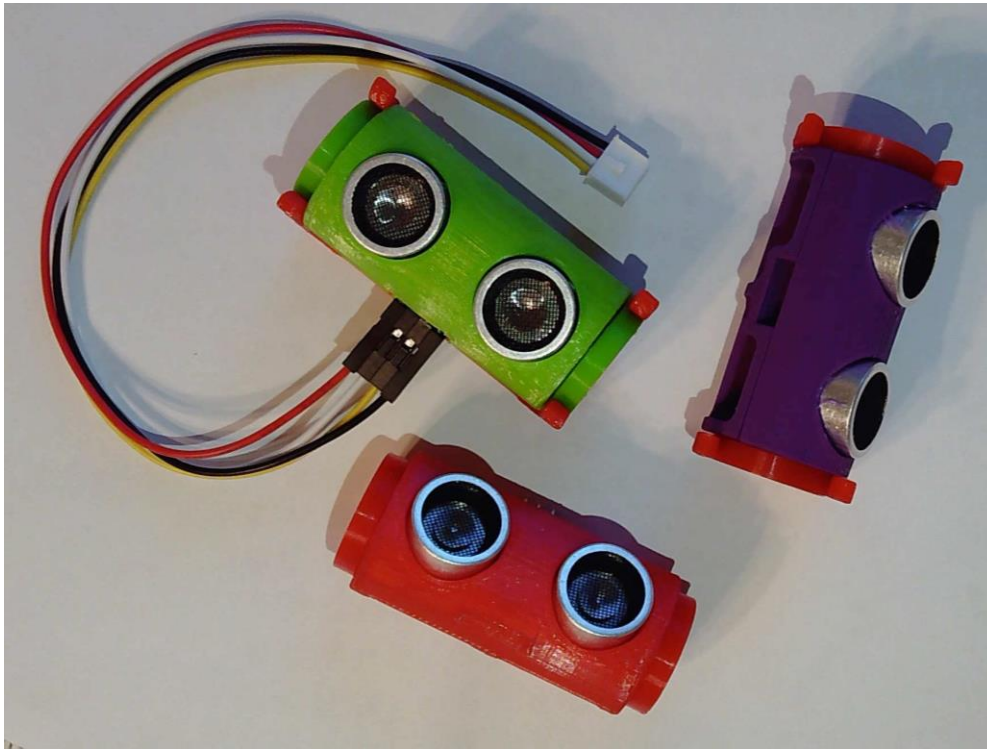
.....

### Go further

We can use this system in many other experiment and applications, e.g measure sound speed, measure distance and many other cases.  
Use the circuit made to determine the spring constant elasticity

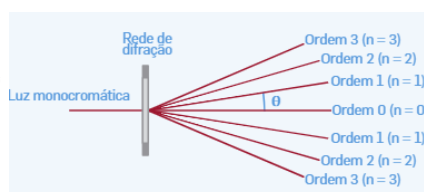
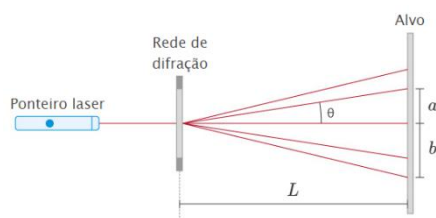
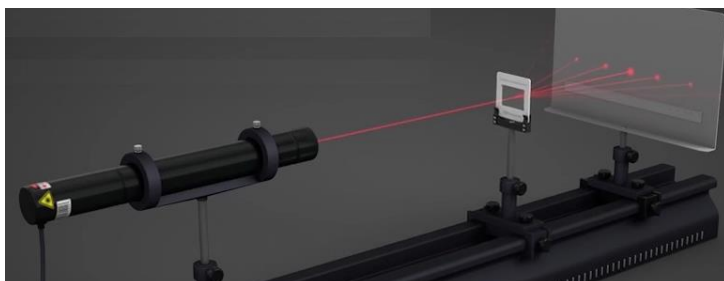
Supplementing the code with the operation of the LCD display  
Building a mobile version (with UPS) adding results logging to the microcontroller memory.

In the photo below - housings for ultrasonic distance sensors HCSR04P, printed by a Polish team on a 3D printer. The housings allow easy mounting and convenient rotation of the sensor.





Topic	Age	Country	Date
Estimating the thickness of a hair by light interference	$\geq 14$	Portugal	May 2022



$$n\lambda = d \sin \theta$$



### Part I – Test it!

$L =$  \_\_\_\_\_  $n =$  \_\_\_\_\_  $a =$  \_\_\_\_\_  $d = ?$  \_\_\_\_\_ (300 lines/mm)

$\lambda_{\text{measured}}$	$\lambda_{\text{fabricant}}$	% accuracy

### Part II – Use it!

$L =$  \_\_\_\_\_  $n =$  \_\_\_\_\_  $a =$  \_\_\_\_\_  $\lambda =$  \_\_\_\_\_

$d = ?$

(estimate the  
thickness of  
the hair)

