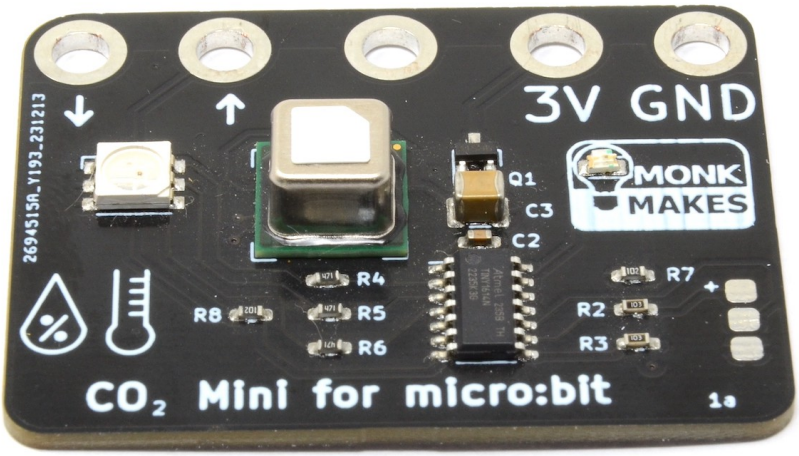
Adventures with the..

CO_2 Mini Kit

For micro:bit Hardware v1a



Please note that this product, while it has many of the same features as the MonkMakes CO2 for micro:bit is based on a completely different sensor and is not a direct replacement.

Instructions Document v1c

Table of Contents

Introduction

The CO2 Mini is a true CO2 sensor, combined with temperature and relative humidity sensors designed for use with the BBC micro:bit.

The board will work with micro:bit version 1 and 2 boards, although some of the projects included in these instructions will onlt work with micro:bit version 2 boards.

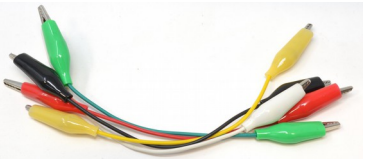
As well as communicating its readings with a micro:bit, the CO2 Mini also has an LED that changes color between green, orange and red, giving a rough indication of air quality.

This booklet includes five experiments complete with code in makecode blocks and Python.

Parts

Please note that a BBC micro:bit is NOT included in this kit.

Before you do anything else, check that your kit includes the following items:

CO2 Mini for micro:bit	
Set of alligator clip leads (5 leads)	

Operation

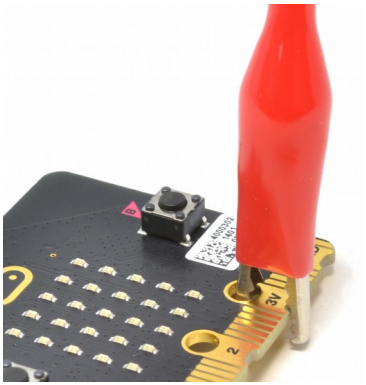
When the sensor first starts up, it goes through a self-test routine, turning the RGB LED red, green and blue in turn. The LED will then blink blue for a few seconds while the sensor starts up.

The readings are likely to be quite erratic for the first 5 minutes that the board is powered up. So, you should ignore these readings.

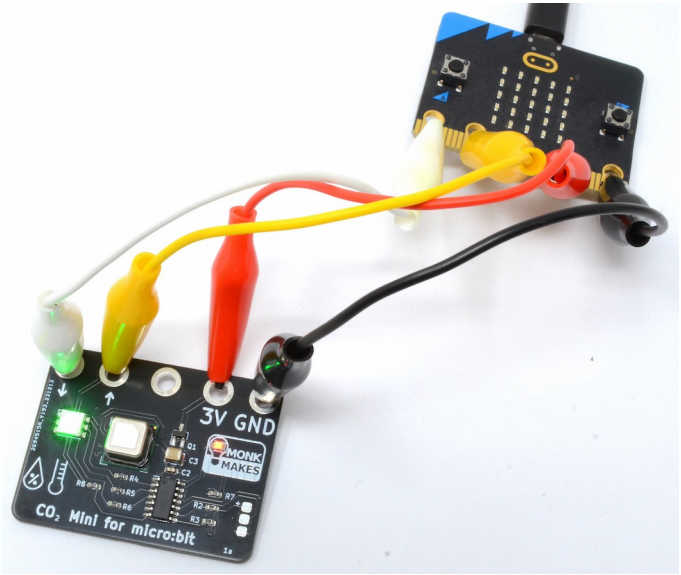
If there is a problem during the self test, the orange LED in the MonkMakes logo will blink an error message. The number of blinks indicates the problem with the board. See the Troubleshooting section for more details.

About Alligator Clips

When using the alligator clips to connect your micro:bit to the CO2 MIni for micro:bit board, you have to be a bit careful how you connect the clips at the micro:bit end. The best way is to connect the clips vertically as shown below.

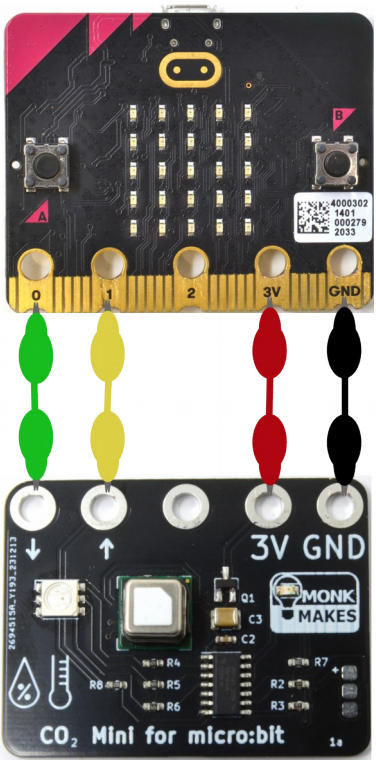


Connecting the alligator clips like this prevents any accidental connections between the large connectors with the holes in and the much smaller connectors (gold lines in the photo above)



Connect the micro:bit to the CO2 Mini for micro:bit

Use four of the alligator clip leads to connect your micro:bit and CO2 Mini for micro:bit together.



Use the alligator clip leads to connect your micro:bit to the CO2 Mini.

It's a good idea to use the red lead for 3V, black for GND for the other two colors chose whatever you like.

Note that there is no connection between pin 2 of the micro:bit and the CO2 Mini board.

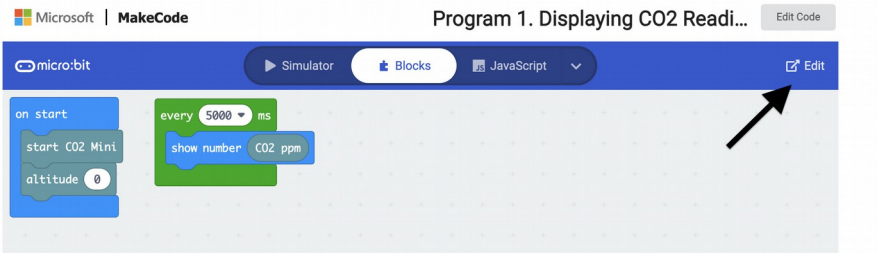
Once the CO2 Mini is receiving power from the micro:bit, the RGB LED will light up, even without any program running on the micro:bit. If it's green, the CO2 level is low, orange – medium and red high.

Experiment 1. Displaying CO2 Readings

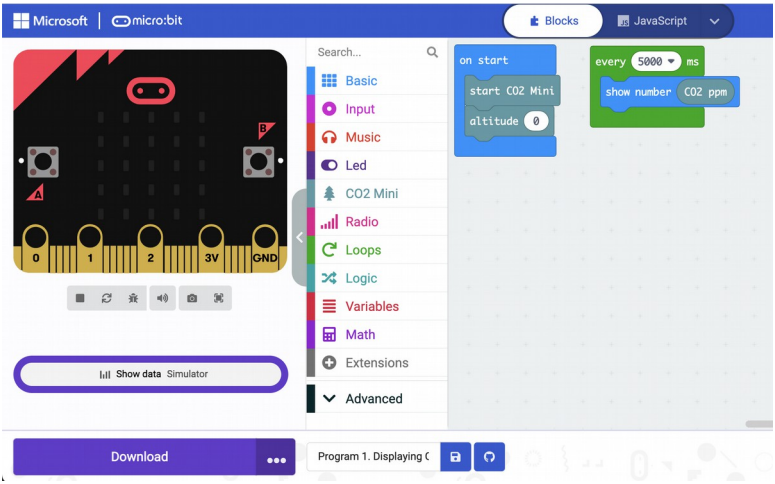
Code Link: https://makecode.microbit.org/_PkpfzTEkqAvr

This program displays the CO2 reading in parts per million, refreshing every 5 seconds.

When you click on the code link at the top of the page, the makecode system will open a window that looks like this:



Here, you can preview the program, but you can't change it or, more importantly, put it on you micro:bit, until you click on the **Edit** button indicated. This will open the normal makecode editor and you can then upload the program onto your micro:bit in the normal way.



When the program first starts, you may see readings like -1, 0 or a very high CO2 level. This is normal. The sensor used by the CO2 Mini takes a few minutes for readings to stabilise. If the readings stay stubbornly high, then you can recalibrate

you CO2 Mini by following the instructions in the section Calibration on page 22.

Once the readings have stabilised, try breathing on the CO2 Mini to increase the CO2 readings. Note that it will take some time for the CO2 readings to increase, and even longer for them to fall back down to the room's CO2 level. That's because the air that finds it's way into the sensor's chamber will take some time to mix with the air from outside the sensor.

The code is pretty simple. The **`on start`** block contains the block **`start CO2 Mini`**, that tells the sensor to start taking readings. The other block in the **`on start`** block is called **`altitude`**. This block must come after **`start CO2 Mini`**, and if you live somewhere high up (more than 500 meters) then you should change the value from 0 to your height in meters, so that the sensor can compensate for the reduced atmospheric pressure that alters the CO2 measurement.

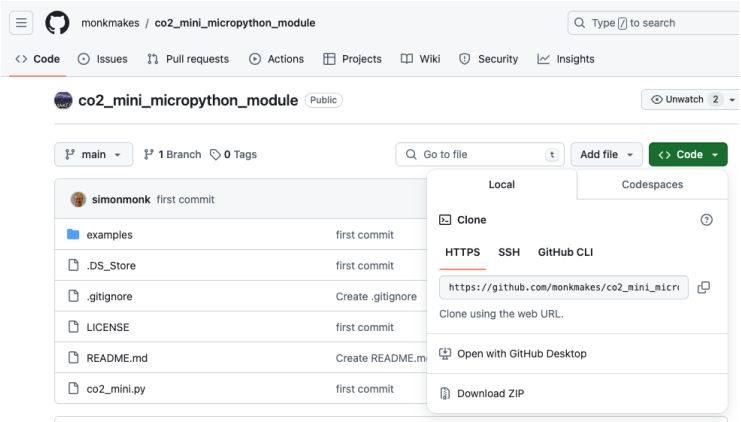
The **`every 5000ms`** block contains code that will be run every 5 seconds. You can find this useful **`every`** block in the **Loops** section of the blocks palette. This block contains the **`show number`** block that takes the **`CO2 ppm`** block as it's parameter to be scrolled across the micro:bit's display.

If you have any problems getting this to work, see the **Trouble Shooting** section at the end of these instructions.

Python

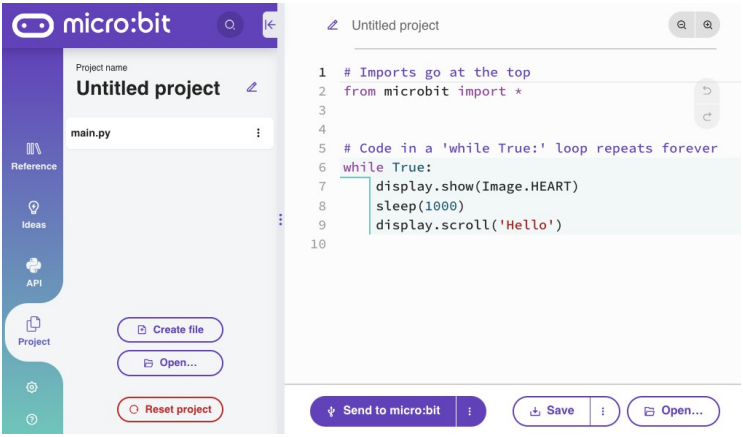
If you'd rather use Python for these projects, then Python versions of the programs are available for download, along with a Python module for the CO2 Mini. To download them open your browser on this URL:

https://github.com/monkmakes/co2_mini_micropython_module



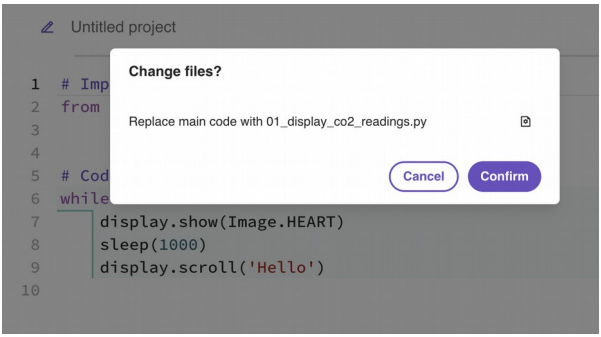
Click on the green **Code** button and then select **Download ZIP** and save the ZIP file somewhere convenient and then extract it. This will create a folder called co2_mini_micropython_module-main.

Now open a new browser tab and open the micro:bit Python editor here: https://python.microbit.org/



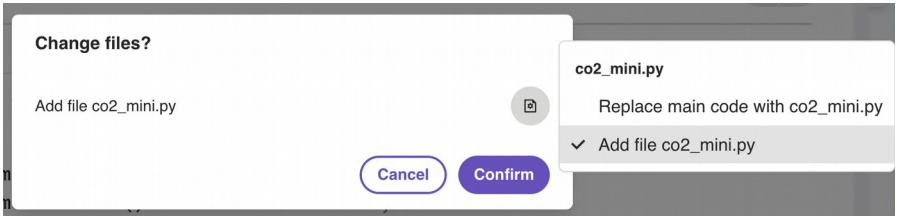
Next click on **Project** in the sidebar and then the **Open..** button and navigate to the folder you unzipped earlier. Within this folder, navigate into **examples** and then select 01_display_co2_readings.py.

This will cause a prompt to appear asking if you want to replace main.py with this example file. Click on **Confirm**.



Don't try and run the program yet, because we need to add the module file. To do this, click on the **Open..** button again and this time select the file co2_mini.py at the

root of the folder you downloaded earlier. This time, when the editor prompts you to confirm, click on the tiny cog settings icon above the confirm button and then select **Add file co2_mini.py** before clicking **Confirm**.



Now you can click on the Send to micro:bit button to install the program onto your micro:bit.

Every 5 seconds or so, the micro:bit will display the CO2 level.

Here is the Python code:

```
from microbit import *
from co2_mini import CO2Mini
from time import sleep

co2_mini = CO2Mini()
co2_mini.start() # redirects serial, so no console
co2_mini.set_altitude(0) # in meters
                         # change if you live high up

while True:
    display.scroll(str(co2_mini.co2()))
    sleep(5) # wait this many seconds
```

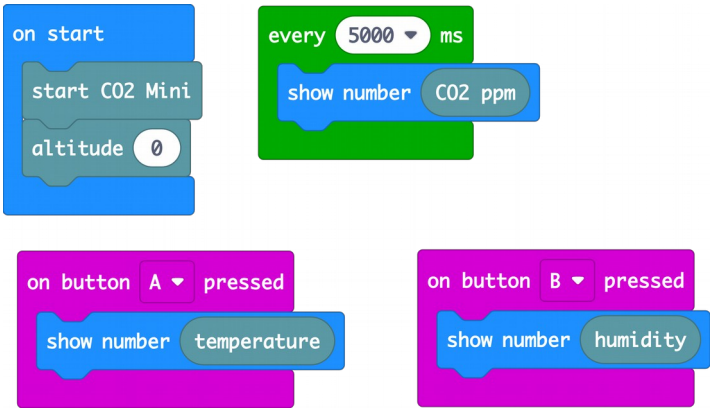
The `CO2Mini` class is imported from the module and a new instance created and assigned to the variable `co2_mini`. The CO2 Mini is started using the `start()` method, and if you live at altitude, you can change 0 to your altitude in meters in the `set_altitude` method.

The main loop uses `display.scroll` to display the CO2 readings fetched from the sensor after converting the reading from a number into a string.

Experiment 2. CO2 Meter

Code Link: https://makecode.microbit.org/_4b5iCAaDsKgP

This program builds on program 1, so that when button A is pressed, the temperature in degrees Celsius is displayed and when button B is pressed the relative humidity is displayed as a percentage.



Install this program onto your micro:bit in the same way as you did program 1, but using the code link at the top of this page.

Python

The Python version of this program can be found in the examples folder that you downloaded for Program 1. The file is called 02_co2_meter.py. Since you already have the co2_mini.py module installed from Program 1, you only need to click on the **Open..** button and navigate to 02_co2_meter.py in examples and Confirm replacement of the main.py file when prompted.

Here is the Python version of the code.

```
from microbit import *
from co2_mini import CO2Mini
from time import sleep, ticks_ms, ticks_diff

co2_mini = CO2Mini()
co2_mini.start() # redirects serial, so no console
co2_mini.set_altitude(0) # in meters change if
                         # you live high up

last_time = 0
```

```
while True:
    now = ticks_ms()
    if ticks_diff(now, last_time) > 5000:
        display.scroll(str(co2_mini.co2()))
        last_time = now
    if button_a.was_pressed():
        display.scroll(str(co2_mini.temp()))
    if button_b.was_pressed():
        display.scroll(str(co2_mini.humidity()))
```

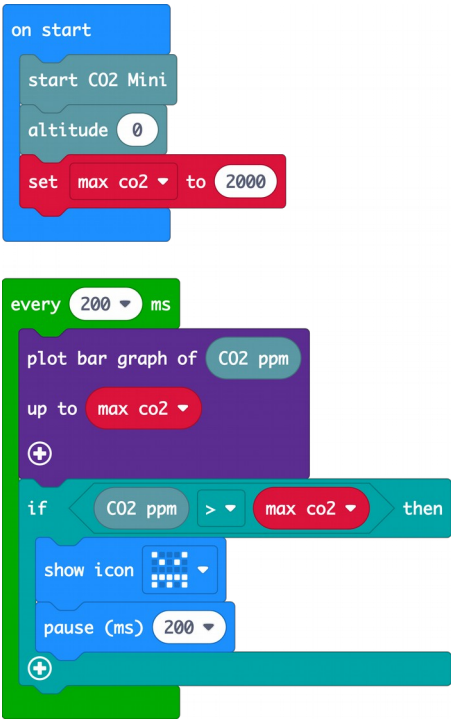
The micro:bit will scroll out the message every 5 seconds, but the rest of the time, it needs to monitor the buttons to detect presses. This means that, unlike Program 1, we can't just `sleep` for 5 seconds. Instead we use a variable called `last_time` and two functions from the time module (`ticks_ms` and `ticks_diff`). The variable `now` is assigned a value by calling `ticks_ms`, which returns the number of milliseconds since the micro:bit was powered up. This is compared in an `if` statement using `ticks_diff` to determine how many milliseconds have elapsed. When this reaches 5000 (5 seconds) the CO2 reading is scrolled on the display and then `last_time` is set to `now` so that this code does not get called for another 5 seconds.

This means that except when the display is scrolling a reading, the A and B buttons can be checked to see if they have been pressed, and if they have, the necessary messages for temperature and humidity are displayed.

Experiment 3. CO2 Alarm

Code Link: https://makecode.microbit.org/_CfDCJ4Vipf75

This program displays the CO2 level as a bar graph on the micro:bit's display rather than as a number. Also, when the CO2 level exceeds a preset value, the display shows a warning symbol.



Python

Using the same procedure as Programs 1 and 2, you can find the Python version of this program in the file 03_co2_alarm.py.

```
from microbit import *
from co2_mini import CO2Mini

max_co2 = 2000
```

```
co2_mini = CO2Mini()
co2_mini.start() # redirects serial, so no console
co2_mini.set_altitude(0) # in meters change
                         # if you live high up

def bargraph(reading): # 0..max_co2
    a = int(reading / (max_co2 / 5)) # 0..4
    if a > 4:
        a = 4
    display.clear()
    display.set_pixel(2, 4, 9) # turn on middle pixel
                               # of bottom row
    for y in range(0, 5):
        if a > y:
            for x in range(0, 5): # turn row on
                display.set_pixel(x, 4-y, 9)

while True:
    co2 = co2_mini.co2()
    bargraph(co2)
    if co2 > max_co2:
        display.show(Image.ANGRY)
        sleep(400)
    sleep(1000)
```

Python for the micro:bit does not have an equivalent to the nice built-in `bar graph` block of makecode. So we have to implement this ourselves as the function `bargraph`.

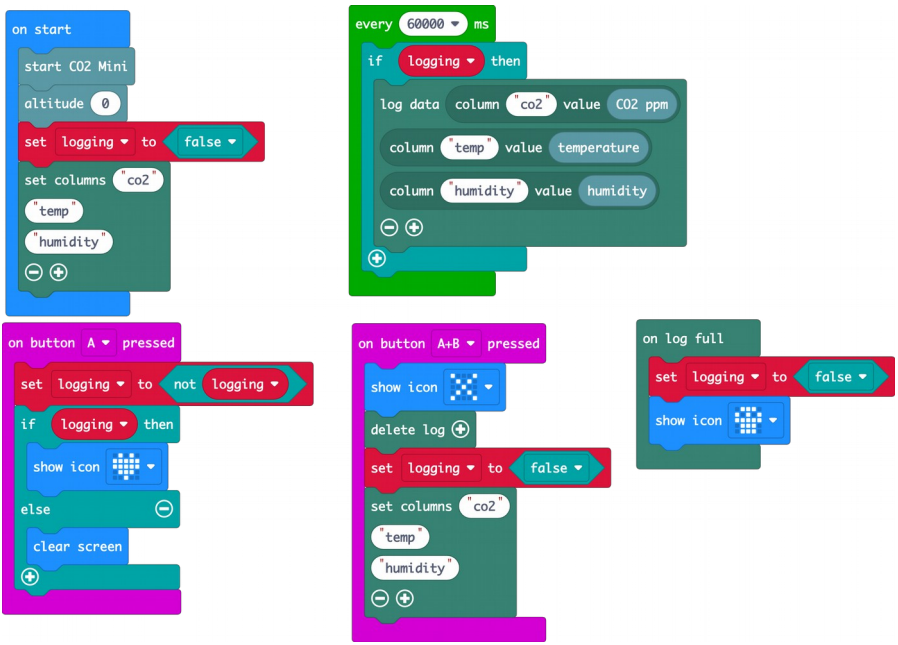
This function first clears the display and turns on a single pixel in the middle of the bottom row (so we can see the micro:bit is working. It then scales the CO2 reading from a number between 0 and `max_co2` to a number of rows to display (`num_rows`). If `num_rows` is greater than the current row number, then the inner for loop turns on all the LEDs on that row.

An if statement in the main loop checks to see if the threshold is exceeded and if it is, displays the angry face icon.

Experiment 4. Data Logging to a File

Code Link: https://makecode.microbit.org/_5svhx86ue0Ky

The makecode version of this experiment will only work on a micro:bit version 2 and there is no Python version for this project.



To use the program, press button A to start data logging. Sampling is set to 60000 milliseconds (1 minute) – ideal for running the experiment overnight. But if you want to speed things up, change this value in the `every` block.

When you want to finish logging, press button A again. You can delete all the data by pressing buttons A and B at the same time.

If the micro:bit runs out of flash memory in which to store the data, it will stop logging and show the 'skull' icon.

The data is written into a file called MY_DATA.HTM. If you go to the MICROBIT drive on your file system, you will see this file. The file is actually more than just the data, it also contains mechanisms for viewing the data. If you double-click on MY_DATA.HTM, it will open in your browser and look something like this:

micro:bit

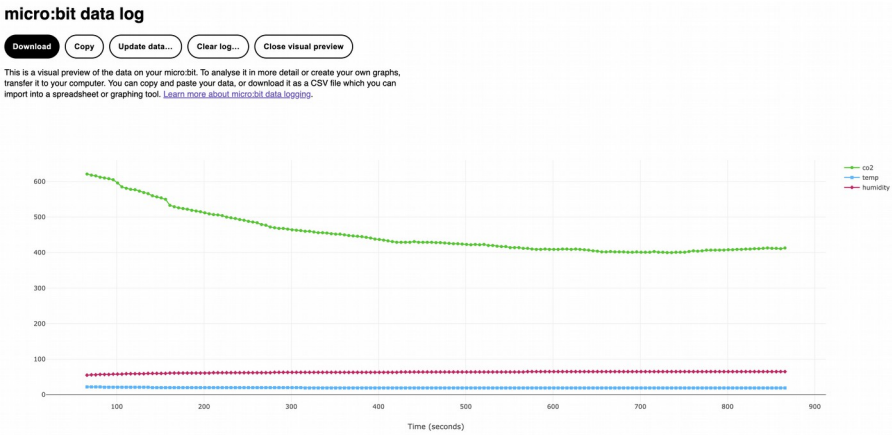
micro:bit data log

Download | Copy | Update data... | Clear log... | Visual preview

This is the data on your micro:bit. To analyse it and create your own graphs, transfer it to your computer. You can copy and paste your data, or download it as a CSV file which you can import into a spreadsheet or graphing tool. Learn more about micro:bit data logging.

Time (seconds)	co2	temp	humidity
65.82	621	22	55
70.82	618	22	56
75.82	616	22	56
80.82	612	22	57
85.82	610	21	57
90.82	608	21	57
95.82	605	21	58
100.82	596	21	58
105.82	585	21	58
110.82	581	21	59
115.82	578	21	59
120.82	577	21	59
125.82	573	21	59
130.82	569	21	59

If you click on the Visual preview button, a simple plot of the data will be displayed.

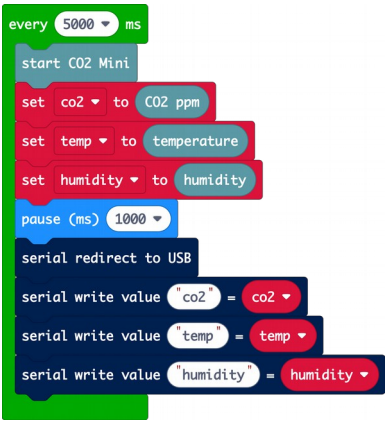


Experiment 5. Data Logging over USB

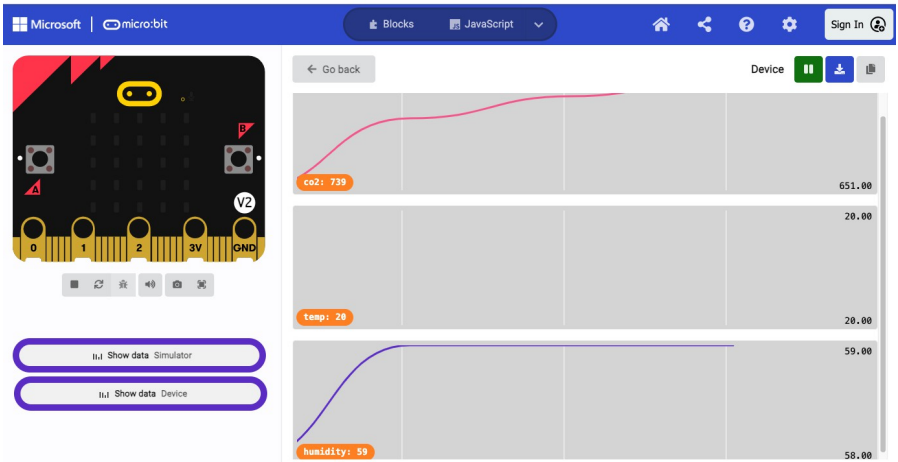
Code Link: https://makecode.microbit.org/_C3w0vJhY6cMc

This project only works on micro:bit version 2 and using the Google Chrome browser. Even so, you may find some problems with this experiment, which should definitely deserves the label of 'experimental'

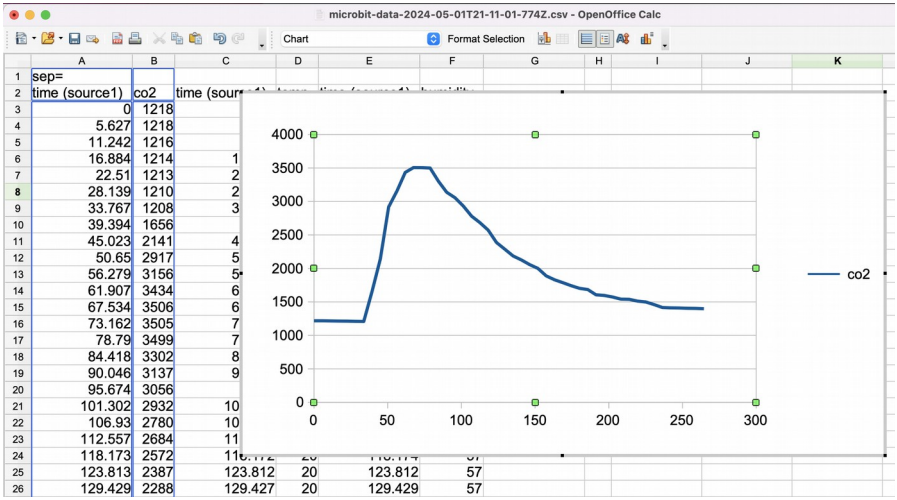
We can extend the previous example to also log temperature and humidity, with this code:



Once the program is uploaded, using a paired micro:bit, click on the **Show data Device** button and you will see something like this.



Having captured the data, you can then click on the blue download icon to save it as a CSV file that can be imported into a spreadsheet, where you can plot charts.



	A	B	C	D	E	F
1	sep=					
2	time (source1)	co2	time (sour			
3	0	1218				
4	5.627	1218				
5	11.242	1216				
6	16.884	1214	1			
7	22.51	1213	2			
8	28.139	1210	2			
9	33.767	1208	3			
10	39.394	1656				
11	45.023	2141	4			
12	50.65	2917	5			
13	56.279	3156	5			
14	61.907	3434	6			
15	67.534	3506	6			
16	73.162	3505	7			
17	78.79	3499	7			
18	84.418	3302	8			
19	90.046	3137	9			
20	95.674	3056				
21	101.302	2932	10			
22	106.93	2780	10			
23	112.557	2684	11			
24	118.173	2572	11			
25	123.813	2387	123.812	20	123.812	57
26	129.429	2288	129.427	20	129.429	57

Makecode Extension

The CO2 Mini uses a Makecode extension to provide a set of blocks for the CO2 Mini. The previous example programs already have the extension installed, but if you are starting a new project, you will need to install the extension.

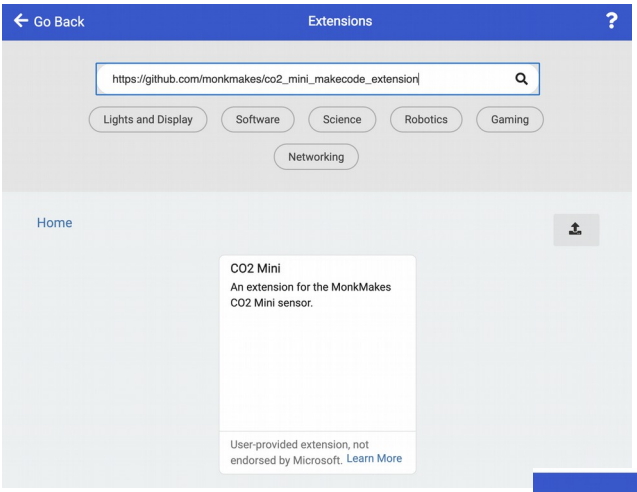
To do this:

* Go to the makecode for micro:bit website here: https://makecode.microbit.org/

* Click on **+ New Project** to create a new makecode project – give it whatever name you like

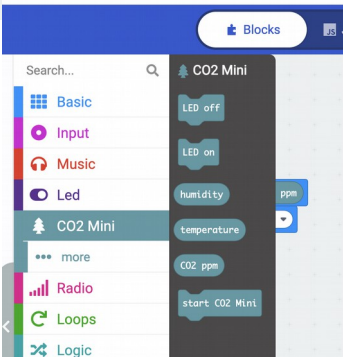
* Click on the **+ Extension** and in the Search area paste the following web address: https://github.com/monkmakes/co2_mini_makecode_extension

This should bring up a single search result.

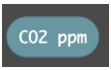
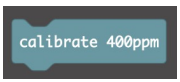


* Click on the CO2 Mini extension and it will be installed.

* Click on ← **Go Back** and you will find that some new blocks have been added to your list of blocks under the category CO2 Mini.



Blocks Description

Block	Description
	This block is usually placed in your on start block and it redirects the micro:bit's serial port to talk to the CO2 Mini.
	This block returns the CO2 concentration detected by the sensor in ppm (parts per million). Fresh air is usually about 400ppm. This sensor can measure concentrations as high as 5000ppm.
	This block returns the temperature in degrees Celsius.
	This block returns the relative humidity as a percentage.
	If this block is used the CO2 Mini's built-in LED will display an indication of the CO2 level by changing color. Green if under 1000ppm, Orange if under 3000ppm and Red if over 3000ppm.
	This block turns off the CO2 Mini's RGB LED.
Advanced Blocks	
	The NDIR (nondispersive infrared) sensor used in the CO2 Mini will give inaccurate results if you live at any significant altitude (more than a few hundred metres). For more accurate results use this block in your on start block. The default altitude is 0m.
	When this block is invoked, the CO2 sensor recalibrates itself assuming that the CO2 level it is currently detecting is 400ppm.
	This block will remove any altitude or manual calibration settings, to return the sensor to it's original condition.

Python Module

The Python module for the CO2 Mini can be found here:
https://github.com/monkmakes/co2_mini_micropython_module

It comprises a class called CO2Mini that has the following methods, that correspond to the blocks in the makecode extension.

Method	Description
start()	This method redirects the micro:bit's serial port to talk to the CO2 Mini.
co2()	This method returns the CO2 concentration detected by the sensor in ppm (parts per million). Fresh air is about 400ppm. This sensor can measure concentrations as high as 5000ppm.
temp()	This block returns the temperature in degrees Celsius.
humidity()	This block returns the relative humidity as a percentage.
led_on()	If this block is used the CO2 Mini's built-in LED will display an indication of the CO2 level by changing color. Green if under 1000ppm, Orange if under 3000ppm and Red if over 3000ppm.
led_off()	This block turns off the CO2 Mini's RGB LED.
set_altitude(m)	The NDIR (nondispersive infrared) sensor used in the CO2 Mini will give inaccurate results if you live at any significant altitude (more than a few hundred meters). For more accurate results use this function in your on start block. The default altitude is 0m.
calibrate400()	When this block is invoked, the CO2 sensor recalibrates itself assuming that the CO2 level it is currently detecting is 400ppm.
factory_reset()	This block will remove any altitude or manual calibration settings, to return the sensor to it's original condition.
firmware_version()	Returns the CO2 Mini's firmware version.

Altitude Compensation

If you are using the sensor at under 500 meters of altitude then you do NOT need to worry about this. However, if you happen to live in a high-altitude place, then you need to tell the CO2 Mini the altitude by including an **`altitude`** block from the CO2 Mini extension (in the **...more** section) or the `set_altitude` function in the Python module.

In both cases, the parameter is your altitude in meters.

You can change the altitude compensation value anywhere in your programs using this block, but normally it makes sense to just do this in the **on start** block or before the main loop in a Python program.

Calibration

Given 5 or 10 minutes to settle, the CO2 Mini should produce reasonably accurate CO2 readings without calibration. However, when you need the most accurate results, the CO2 Mini should be calibrated. As time goes on, its readings will drift away from the true CO2 level. If your CO2 Mini starts reporting readings of 0 (or well below the atmospheric average of about 400) for CO2, then it probably needs recalibrating.

To do this:

1. Choose a room where you can leave a window open for a while.
2. Load this makecode program onto your micro:bit connected to the CO2 Mini. https://makecode.microbit.org/_8i38w9AyvXM8 This will repeatedly scroll the CO2 level across this micro:bit's display.
3. Take a look at the readings for a little while, to make sure everything is working.
4. Open the window and leave the room so that the CO2 level can fall to that of the air outside. I would leave the room empty for at least 10 minutes.
5. While trying not to breathe on the CO2 Mini, press button A on the micro:bit. This will calibrate it to 400ppm.
6. Watch the readings, and after half a minute or so, the readings should start to be showing something close to 400.
7. Sometimes, the readings appear to stick at 0. If this happens Press button A again and wait for the readings to come back to near 400.

The makecode extension block, that causes the calibration to take place, is in the **... more** section of the CO2 Mini extension. You could use it like this:

Note that if left powered up and running for over 44 hours, the CO2 Mini will auto-calibrate, by assuming that the lowest reading in that 44 hour period is 400ppm. This is a reasonable assumption, as the CO2 level will naturally fall to about 400ppm in an unoccupied room overnight.

CO2 and Health

The level of CO2 in the air we breathe has a direct influence on our well-being. CO2 levels are of particular interest from a public health point of view as, to put it simply, they are a measure of how much we are breathing other people's air. We humans breathe out CO2 and so, if several people are in a poorly ventilated room, the level of CO2 will gradually increase. As will the viral aerosols that spread disease.

Another important impact of CO2 levels is in cognitive function – how well you can think. The following quote is from the National Centre for Biotechnology Information in the USA:

"at 1,000 ppm CO2, moderate and statistically significant decrements occurred in six of nine scales of decision-making performance. At 2,500 ppm, large and statistically significant reductions occurred in seven scales of decision-making performance" *Source: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3548274/*

The table below is based on information from https://www.kane.co.uk/knowledge-centre/what-are-safe-levels-of-co-and-co2-in-rooms and shows the levels at which CO2 can become unhealthy.

Level of CO2 (ppm)	**Notes**
250-400	Normal concentration in ambient air.
400-1000	Concentrations typical of occupied indoor spaces with good air exchange.
1000-2000	Complaints of drowsiness and poor air.
2000-5000	Headaches, sleepiness and stagnant, stale, stuffy air. Poor concentration, loss of attention, increased heart rate and slight nausea may also be present.
5000	Workplace exposure limit in most countries.
>40000	Exposure may lead to serious oxygen deprivation resulting in permanent brain damage, coma, even death.

Troubleshooting

Problem: The amber power LED on the CO2 Mini for micro:bit is not lit.

Solution: Make sure that your micro:bit itself is receiving power and that the GND and 3V connections of the micro:bit are connected to the GND and 3V connectors of the CO2 Mini.

Problem: When a program is running, the orange LED in the MonkMakes logo of the CO2 Mini blinks repeatedly.

Solution: This indicates an error with the board and the number flashes indicates the error. Try disconnecting and reconnecting the power to your micro:bit, but if the problem persists, make a note of the number of LED blinks in each cycle and contact support@monkmakes.com

Problem: The amber power LED on the CO2 Mini for micro:bit is lit, but my program is only seeing CO2, temperature and humidity readings that are all -1.

Solution: This indicates that your micro:bit is not communicating with the CO2 Mini. There can be several reasons for this.
* The wiring may be misconnected. Take all the alligator clip leads off and reattach them.
* There may be a faulty alligator lead. Try swapping each lead out in turn for a spare.
* You may have forgotten to put the **start CO2 Mini** block in the **on start** block of your program.

Problem: Sometimes the readings freeze and communication seems unreliable.

Solution: Try and use short alligator leads like the ones supplied and keep them away from potential sources of electrical interference, such as power leads or leads to electric motors.

Problem: When I first run my program, the CO2 readings seem wrong, sometime 0 or -1.

Solution: This is normal. The sensor takes some time to settle. Disregard any readings for the first minute after the sensor starts up. Also, see the section Calibration on page 14.

Problem: I'm using Python but I cant use the Python Shell.

Solution: The CO2 Mini uses the micro:bit's serial interface, making the Python shell unavailable.
If you are trying to use trace to debug your program, you can scroll messages onto the micro:bit screen using **display.scroll('my message')**.

Problem: I'm using Python and the CO2 Mini RGB LED has gone off and no readings are coming back, but the Orange power LED in the MonkMakes logo is lit.

Solution: A side-effect of sharing the serial link between USB and the CO2 Mini, is that if an error occurs in your code, the error message will be sent to CO3 Mini, causing unexpected problems. For example, if the error message contains the letter 'l' the LED will be turned off. Worse, if it contains a 'k' the CO2 Mini will be recalibrated.

The solution is to fix the problem in your code and then disconnect the GND lead to the CO2 Mini to cycle the power to it.

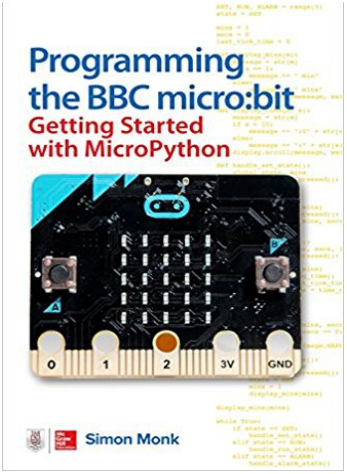
Learning

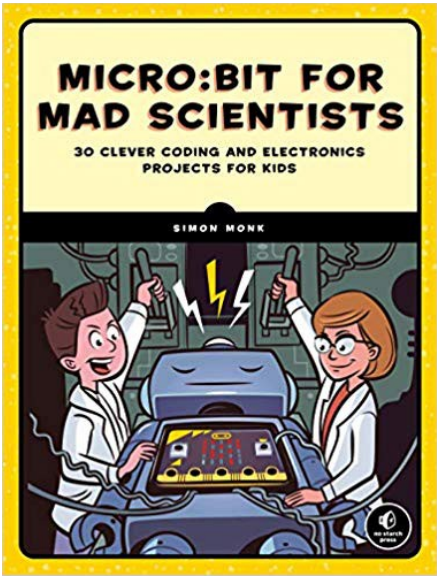
micro:bit Programming

If you want to learn more about programming the micro:bit in MicroPython, then you should consider buying Simon Monk's book 'Programming micro:bit: Getting Started with MicroPython', which is available from all major book sellers.

For some interesting project ideas, you might also like micro:bit for the Mad Scientist from NoStarch Press.

You can find out more about books by Simon Monk (the designer of this kit) at: http://simonmonk.org or follow him on Twitter where he is @simonmonk2

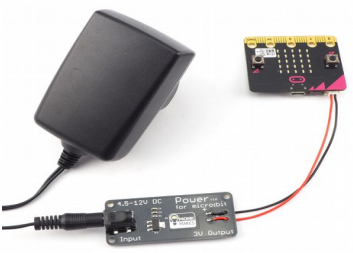


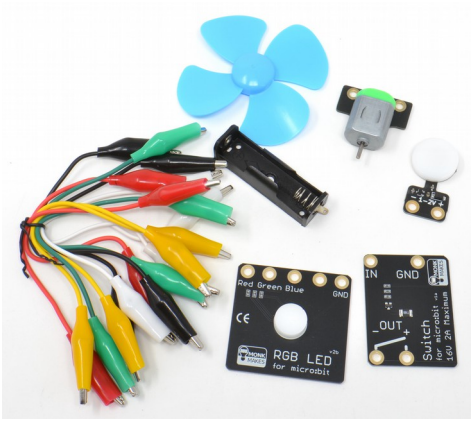


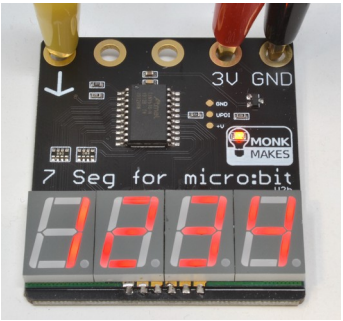
MonkMakes

For more information on this kit, the product's home page is here:
https://monkmakes.com/mb_slider

As well as this kit, MonkMakes makes all sorts of kits and gadgets to help with your maker projects. Find out more, as well as where to buy here: https://monkmakes.com you can also follow MonkMakes on Twitter @monkmakes.







From left to right: Solar Experimenters Kit for micro:bit, Power for micro:bit (AC adapter not included), Electronics Kit 2 for micro:bit and 7 Segment for micro:bit.