CANSAT BELGIUM

Hardware kit discovery

v2

CANSAT
BELGIUM

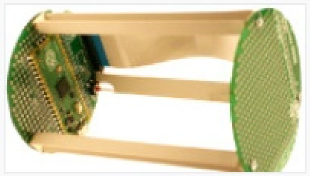
Presentation themes

- About the wiki @ cansat-pico.mchobby.be
- Arduino Uno → Raspberry-Pi Pico
- Review the kit content
- Raspberry-Pi Pico and goodies
- LiPo batteries
- Radio transmission

cansat-pico.mchobby.be

- Getting started guide

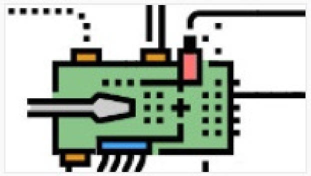
Hardware discovery



Discover the various items included within the kit.

Cliquez ici

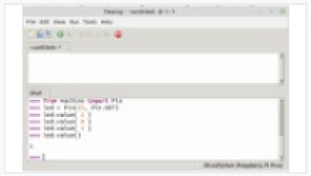
Kit Assembling



Assembling the Pico Cansat Kit

Cliquez ici

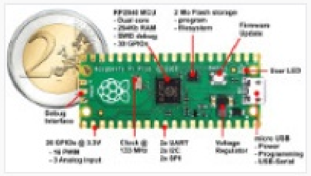
Thonny IDE



Prepare your Python IDE environment

Cliquez ici

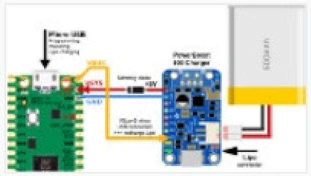
Pico User Guide



The **Raspberry-Pi Pico** user guide.

Cliquez ici

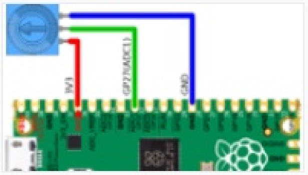
Pico Powering



How to properly power up your Pico

Cliquez ici

MicroPytho HowTo



Tips and tricks to write Python code for on MicroControler.

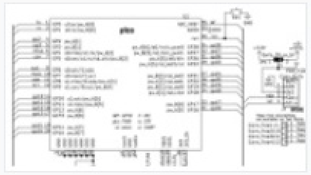
Cliquez ici

Data Logging

How to log data into a file (on MicroControler)

Cliquez ici

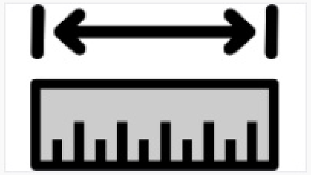
Schematics



Need to have a look on the board schematis?

Cliquez ici

Dimensions



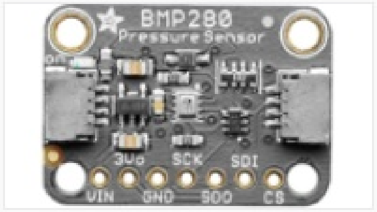
Technical size and dimensions for the cansat

Cliquez ici

cansat.mchobby.be

- Getting started guide
- Testing the devices

BMP280 sensor



Test the BMP280 pressure and elevation sensor.

Cliquez ici

TMP36 sensor

Test the TMP36 analog temperature sensor

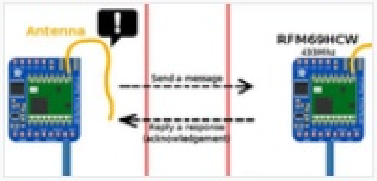
Cliquez ici

RFM69HCW radio

User guide for the **RFM69HCW** radio module.

Cliquez ici

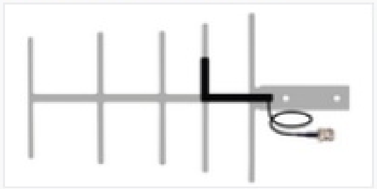
RFM69HCW Testing



Testing the communication with **RFM69HCW** and sending data through the radio module.

Cliquez ici

Radio Antenna

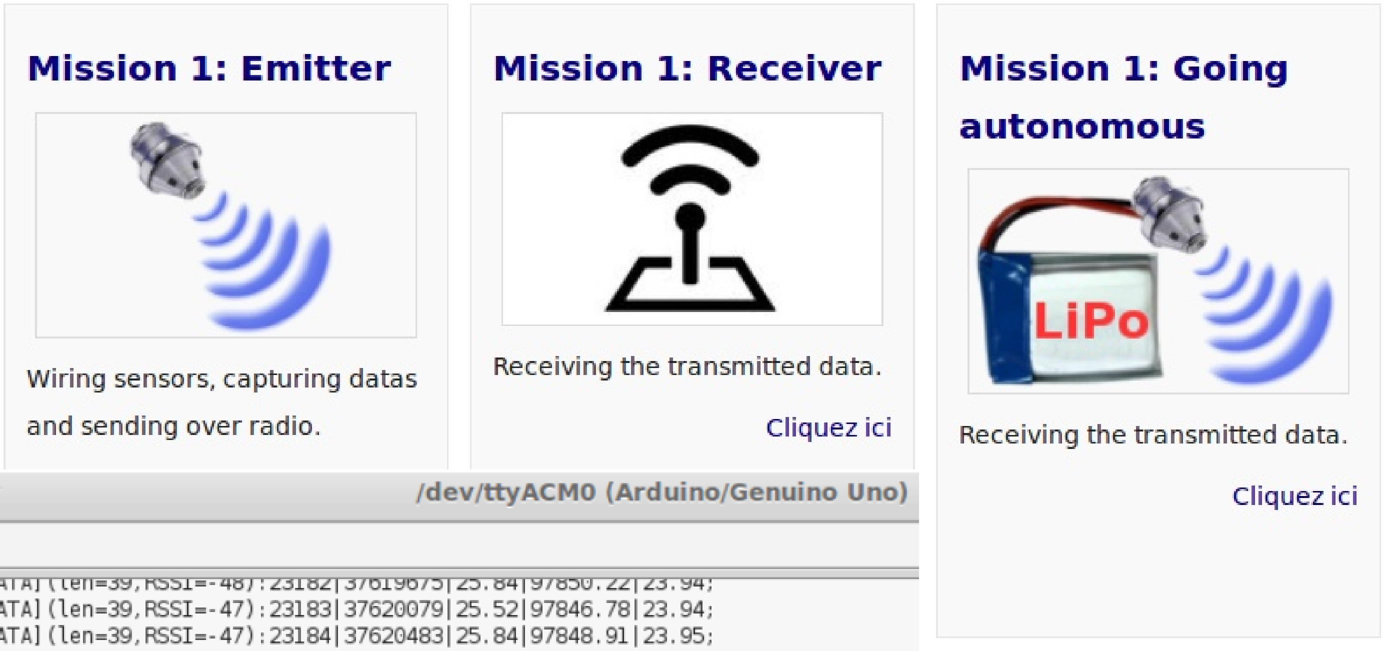


A well designed Antenna can increase the communication distance.

Cliquez ici

cansat.mchobby.be

- Getting started guide
- Testing the devices
- Mission 1
  Radio telemetry transmission.
  Frequency Plan.
  Capturing data.



Mission 1: Emitter

Wiring sensors, capturing datas and sending over radio.

Mission 1: Receiver

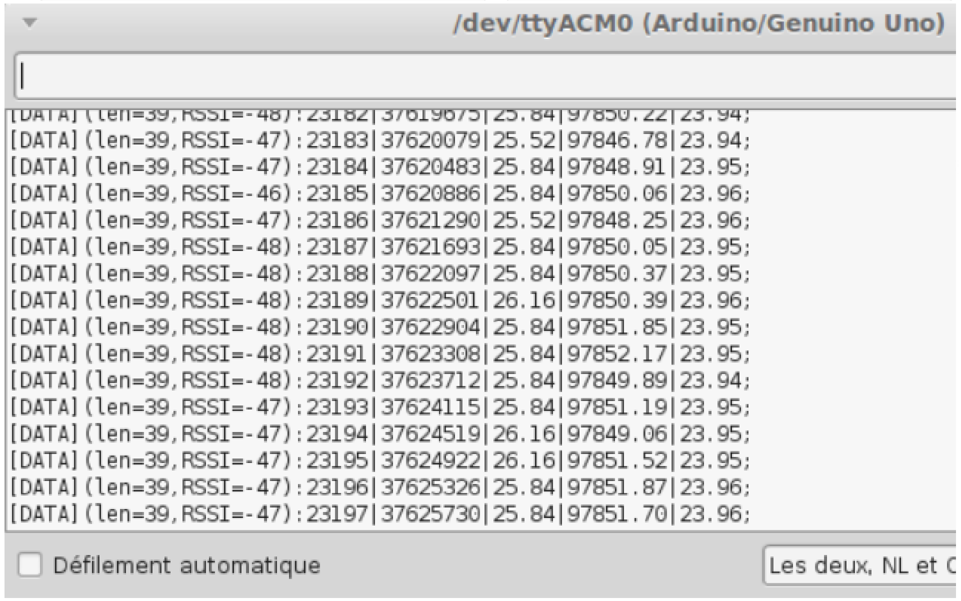
Receiving the transmitted data.

Cliquez ici

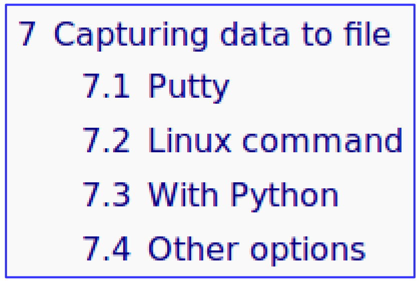
Mission 1: Going autonomous

Receiving the transmitted data.

Cliquez ici



```
[DATA] (len=39, RSSI=-48):23182|37619675|25.84|97850.22|23.94;
[DATA] (len=39, RSSI=-47):23183|37620079|25.52|97846.78|23.94;
[DATA] (len=39, RSSI=-47):23184|37620483|25.84|97848.91|23.95;
[DATA] (len=39, RSSI=-46):23185|37620886|25.84|97850.06|23.96;
[DATA] (len=39, RSSI=-47):23186|37621290|25.52|97848.25|23.96;
[DATA] (len=39, RSSI=-48):23187|37621693|25.84|97850.05|23.95;
[DATA] (len=39, RSSI=-48):23188|37622097|25.84|97850.37|23.95;
[DATA] (len=39, RSSI=-48):23189|37622501|26.16|97850.39|23.96;
[DATA] (len=39, RSSI=-48):23190|37622904|25.84|97851.85|23.95;
[DATA] (len=39, RSSI=-48):23191|37623308|25.84|97852.17|23.95;
[DATA] (len=39, RSSI=-48):23192|37623712|25.84|97849.89|23.94;
[DATA] (len=39, RSSI=-47):23193|37624115|25.84|97851.19|23.95;
[DATA] (len=39, RSSI=-47):23194|37624519|26.16|97849.06|23.95;
[DATA] (len=39, RSSI=-47):23195|37624922|26.16|97851.52|23.95;
[DATA] (len=39, RSSI=-47):23196|37625326|25.84|97851.87|23.96;
[DATA] (len=39, RSSI=-47):23197|37625730|25.84|97851.70|23.96;
```



cansat.mchobby.be

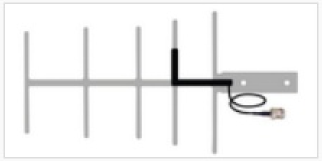
- Getting started guide
- Testing the devices
- Mission 1
  Radio telemetry transmission.
  Capturing data.
- Resources

CanSat 3D

CanSat 3D models to print your own one

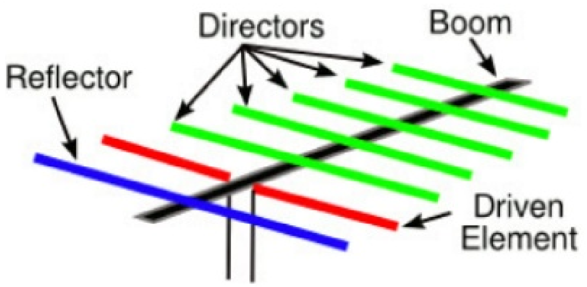
Cliquez ici

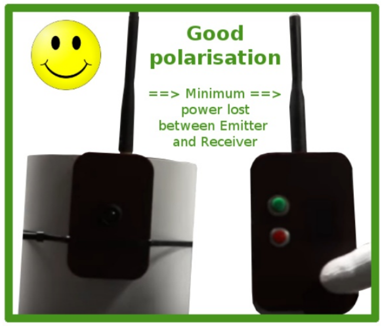
Radio Antenna



A well designed Antenna can increase the communication distance.

Cliquez ici

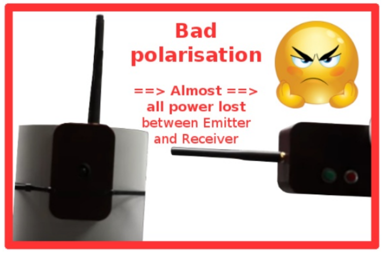




Parachute

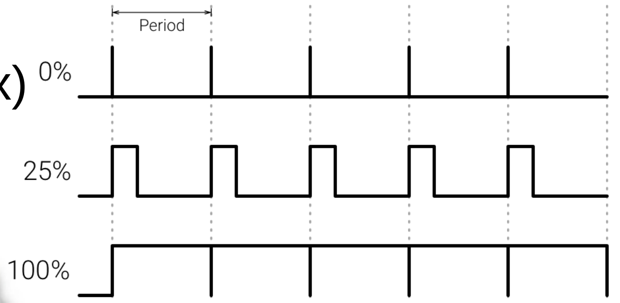
Some reference to design the parachute

Cliquez ici



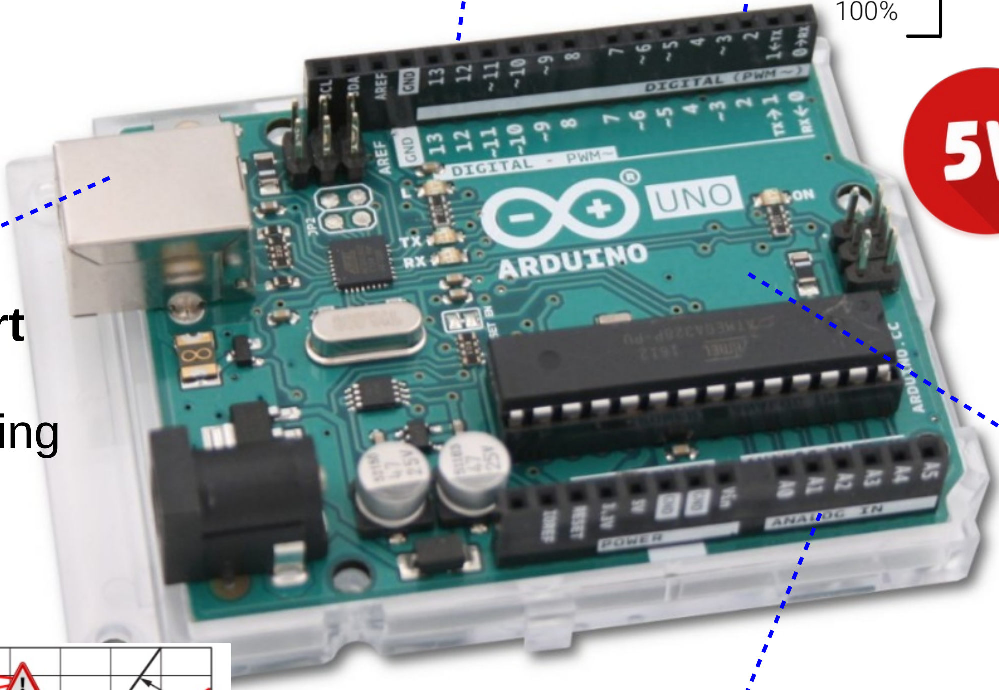
Arduino Uno → Pico

Arduino UNO



PWM (6x)

Digital I/O (13x)



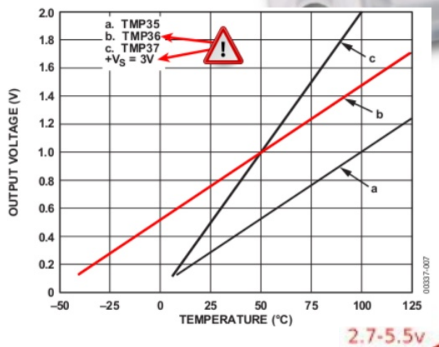
5V

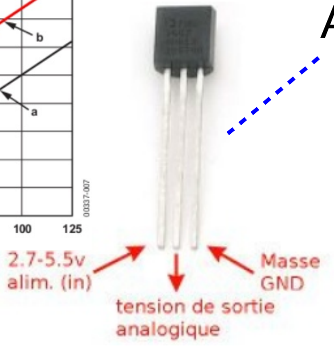
- Microcontroler ATmega328
- 32K Flash
- **2K SRam** 2048 char.

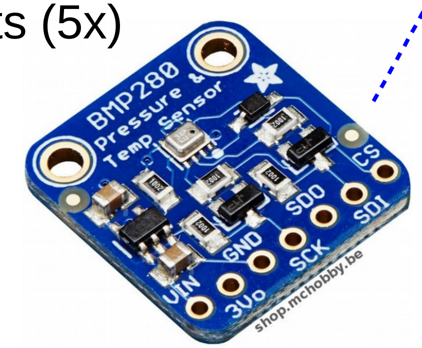
USB Port
easy
programming

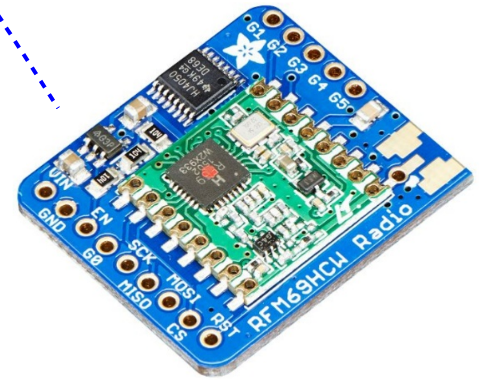
SPI & I2C buses
Easy connection
for sensors and
circuitery

Analog inputs (5x)



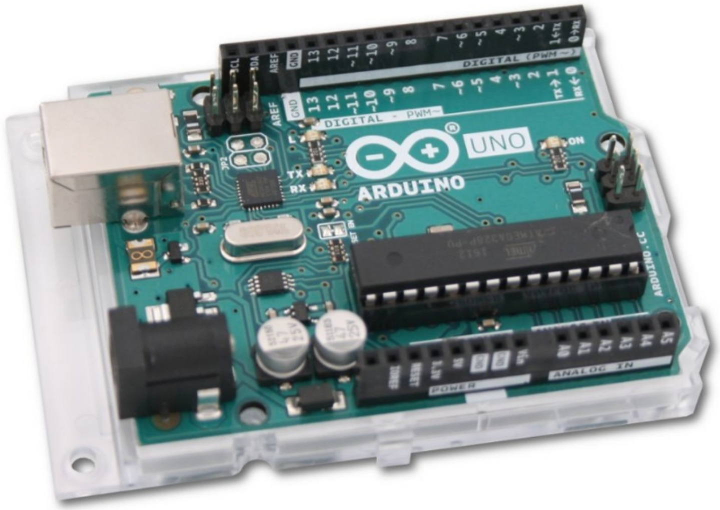


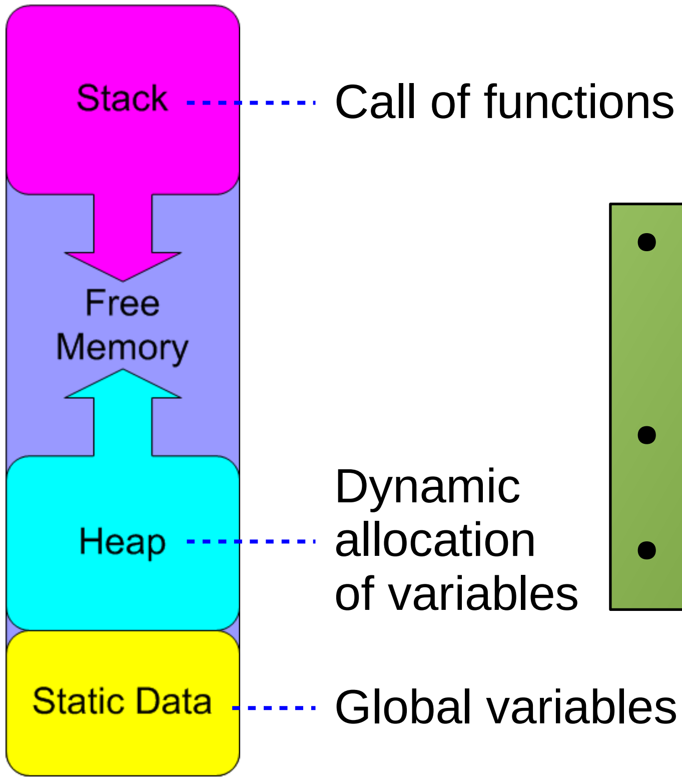




Arduino Uno → Pico

Arduino UNO





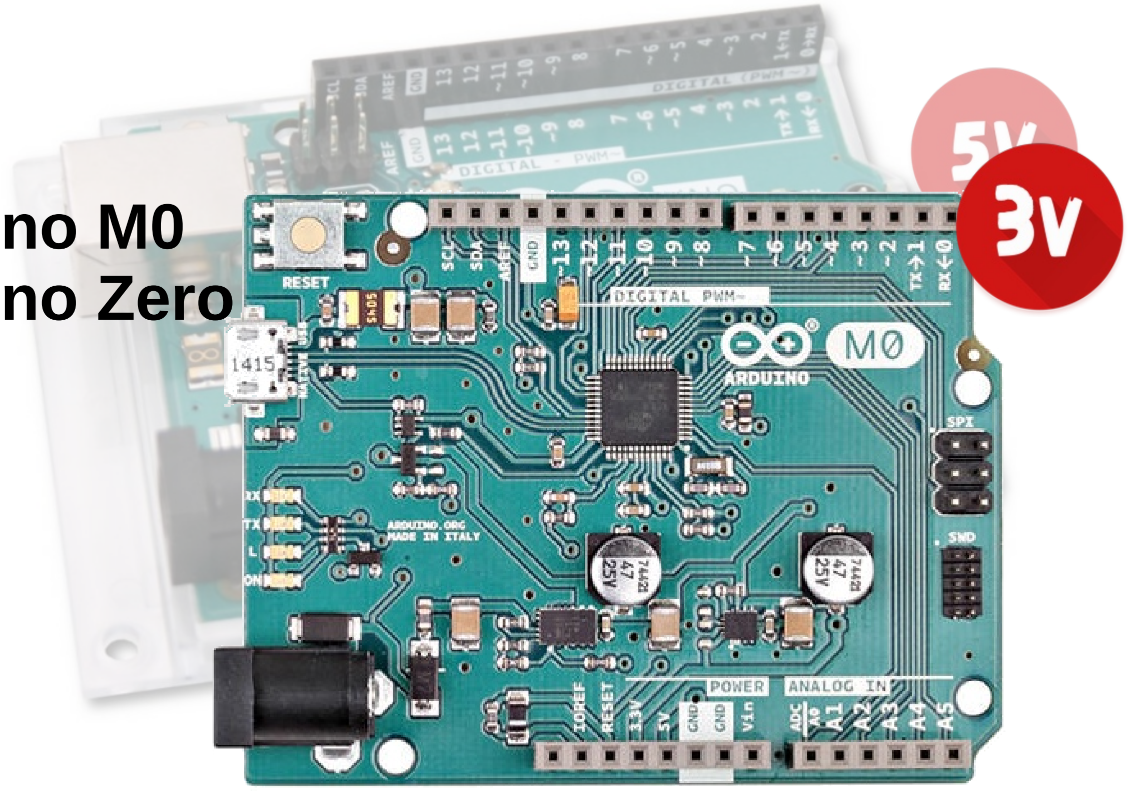
- Microcontroler ATmega328
- 32K Flash
- **2K SRam** 2048 char.

```
Sketch uses 21,316 bytes (66%) of program storage space. Maximum
is 32,256 bytes.
Global variables use 1,629 bytes (79%) of dynamic memory, leaving
419 bytes for local variables. Maximum is 2,048 bytes.
Low memory available, stability problems may occur.
```

Arduino Uno → Pico

~~Arduino UNO~~

Arduino M0
Arduino Zero



- Microcontroler ATmega328
- 32K Flash
- **2K SRam** 2048 char.

- Microcontroler ATSAMD21G18 (ARM Cortex M0+)
- 256K Flash
- **32K SRam** 32768 char.

M0

6x analog input
1x **analog output**
16 Mhz → **48 Mhz**

Arduino Uno → Pico

~~Arduino UNO~~

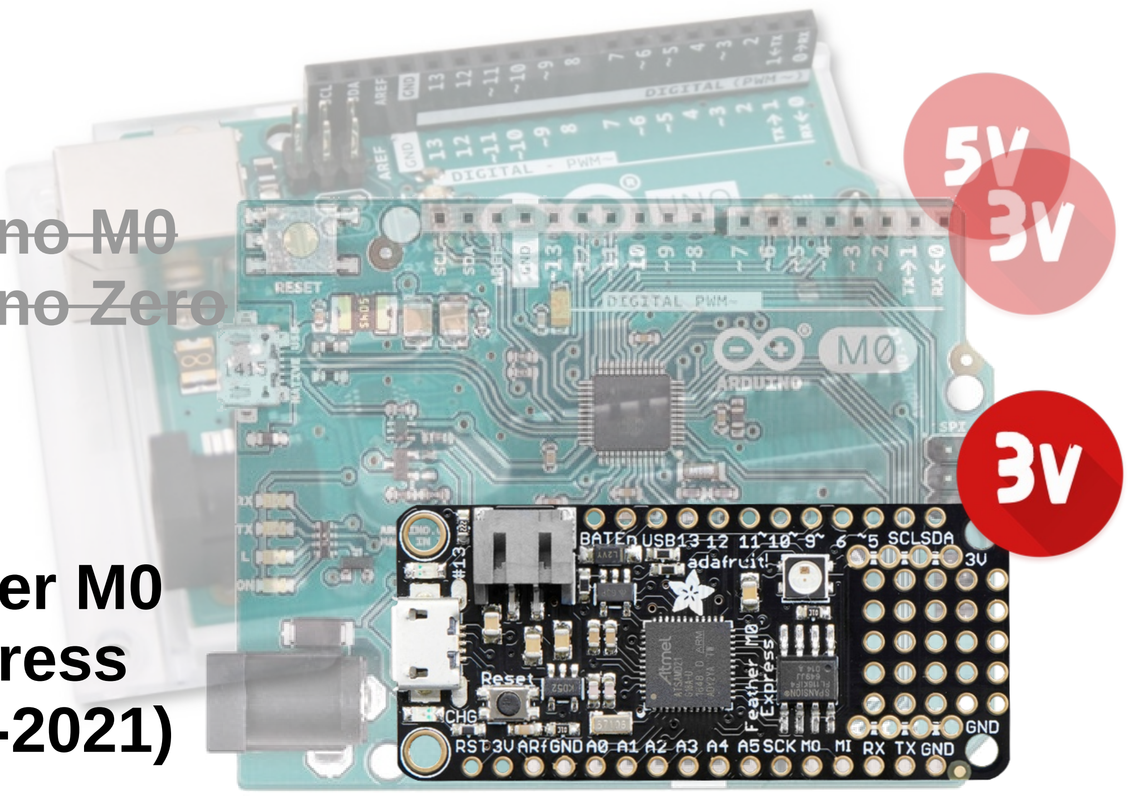
- Microcontroler ATmega328
- 32K Flash
- **2K SRam** 2048 char.

~~Arduino M0~~
~~Arduino Zero~~

- Microcontroler ATSAMD21G18 (ARM Cortex M0+)
- 256K Flash

Feather M0 Express (2019-2021)

- Microcontroler ATSAMD21 (ARM Cortex M0+)
- 256K Flash
- **32K SRam** 32768 char.



Feather

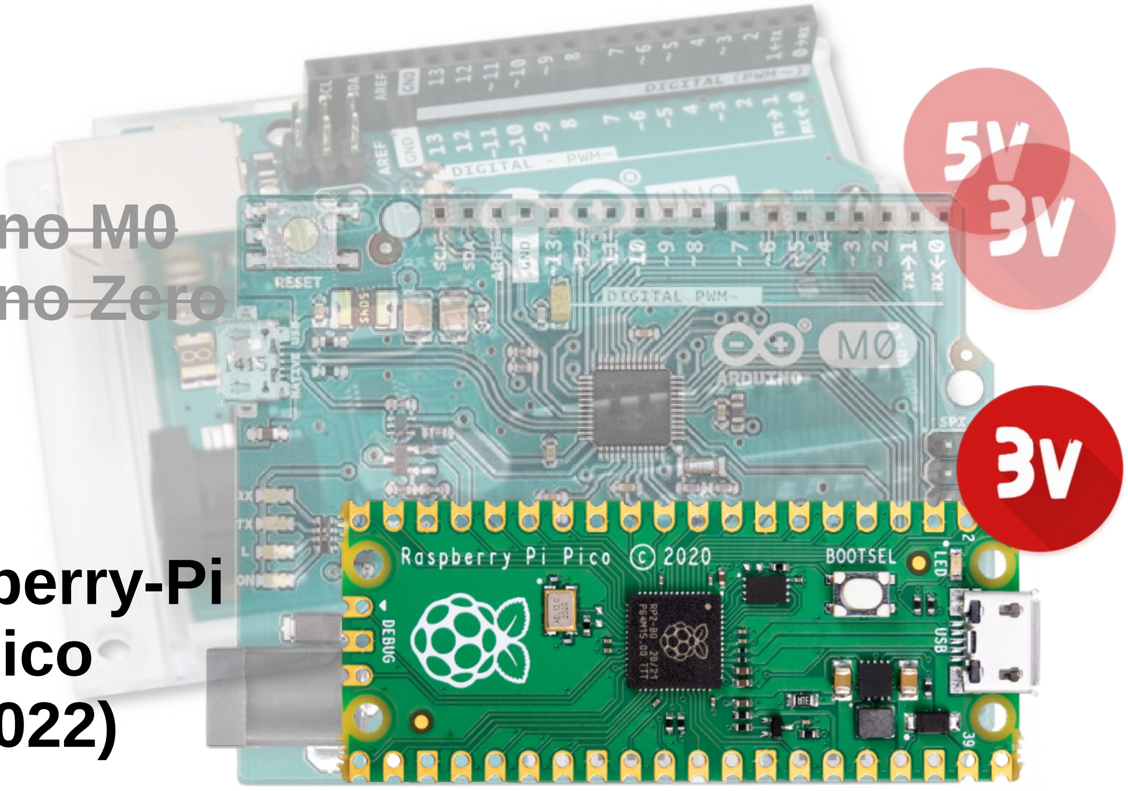
1x Lipo charger
1x **SPI Flash**
1x NeoPixel LED

Arduino Uno → Pico

~~Arduino UNO~~

~~Arduino M0~~
~~Arduino Zero~~

Raspberry-Pi Pico (2022)



- Microcontroler RP2040 @ 133 Mhz (ARM Cortex M0+)
- 2048 Kio Flash
- **265Kio SRam**.

Pico

1x **SPI Flash**
1x Dual core 133Mhz
Python ready

ARDUINO

Arduino Uno → Pico

- Microcontroler RP2040 dual code (ARM Cortex M0+)
- 133 MHz
- 2 Mio Flash
- **264K SRam**

3v

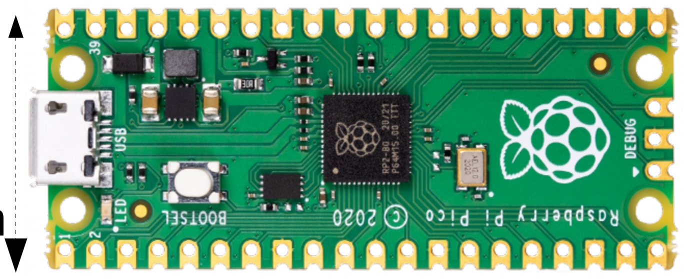
Raspberry-Pico

Pico are :

- ✓ Small (5,25 x 2,1cm)
- ✓ Light (4,0 gr)
- ✓ Powerful
- ✓ Versatile
- ✓ Polyvalent
- ✓ Multi-language
- ✓ Worldwide supported
- ✓ Widely documented

52,5mm

21 mm



Pico features :

- ➔ 2 core @ 133 Mhz
- ➔ 26 GPIOs
- ➔ 3 Analog inputs - 12 bits
- ➔ 3 Internal Analog
- ➔ 16 PWM outputs
- ➔ Hardware I2C, SPI buses
- ➔ UART
- ➔ PIO (Programmable IO)

4.0 Gr

Official C++
Pico.raspberrypi.org

Official MicroPython
micropython.org
Pico.raspberrypi.org

Rust
Rust.org

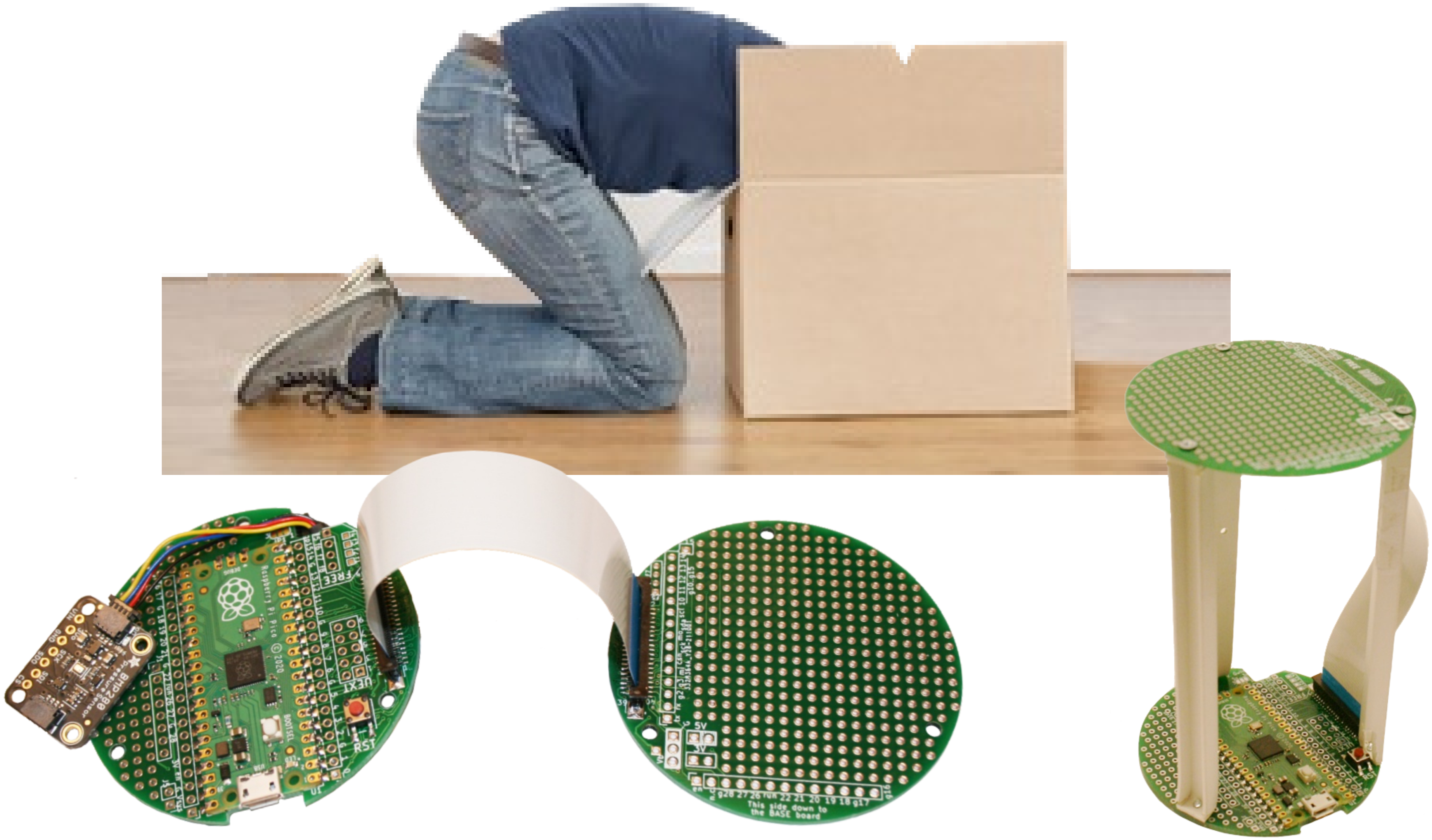
ARDUINO
arduino.cc

circuit python
Adafruit's MicroPython
learn.adafruit.com/welcome-to-circuitpython

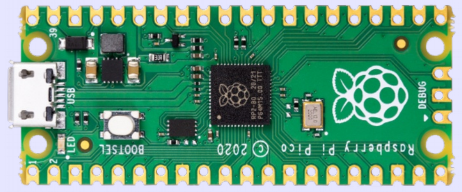
Pico

1x **SPI Flash**
1x Dual core 133Mhz
Python ready

Kit discovery



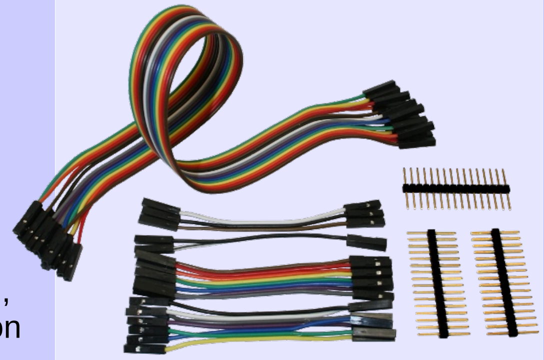
CanSat kit content



x2

Raspberry-Pi
Pico (RP2040 cores)
New Python & Arduino dual core mcu.

Compatible with MicroPython,
Arduino IDE and CircuitPython



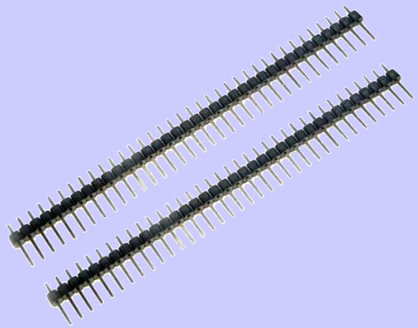
Multi-functional breadboard wires

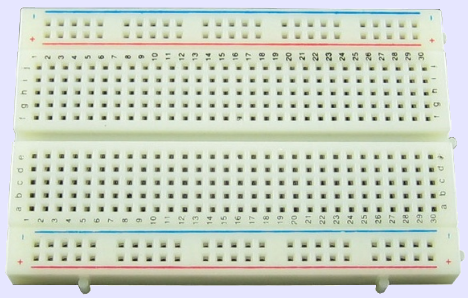
Set of wires with plug that can be modified from female to male.

USB A/microB 1m cable.

Used to plug your feather on a computer to program it or to recharge the battery.

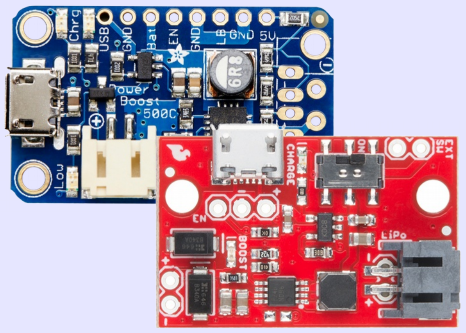
Pin Headers
Plug one Pico on breadboard and start experimenting with Pico.





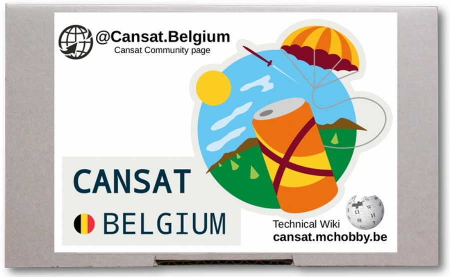
Half Size Breadboard

Solderless breadboard are used for fast prototyping.

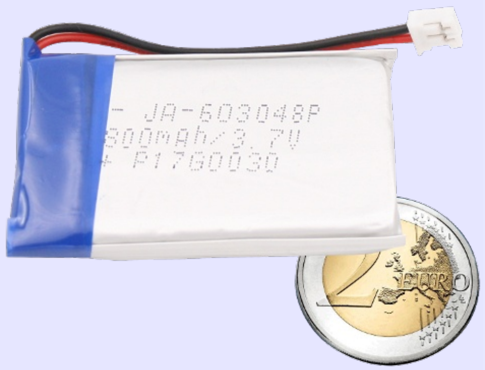


PowerBoost Charger/ Booster

Get 5V power supply from Lipo battery. Charging capability included.

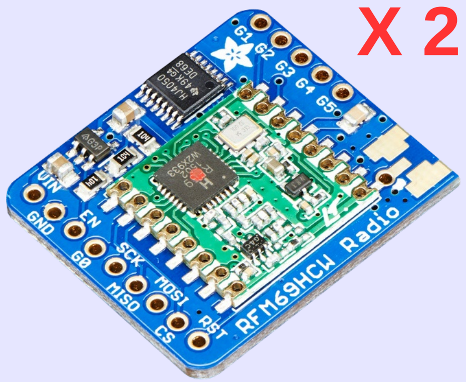


CanSat kit content



Lithium Polymer Battery

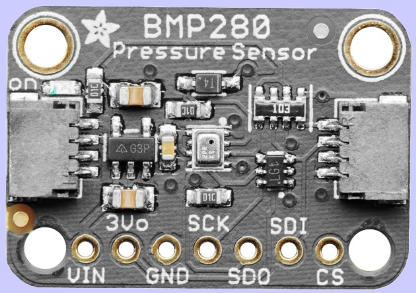
Transform the setup into an autonomous plateform with this 1400mAh Lipo.



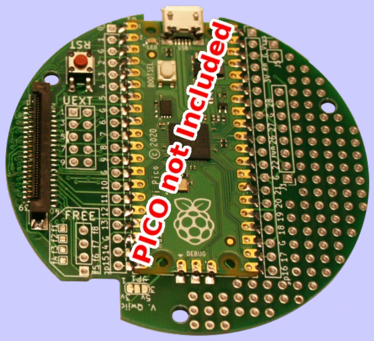
RFM69HCW Transceiver Radio
Transport data over long distance with packet radio. One breakout act as emitter, the second one as receiver.

BMP280 Barometric pressure sensor

Easily evaluate pressure, altitude and temperature.

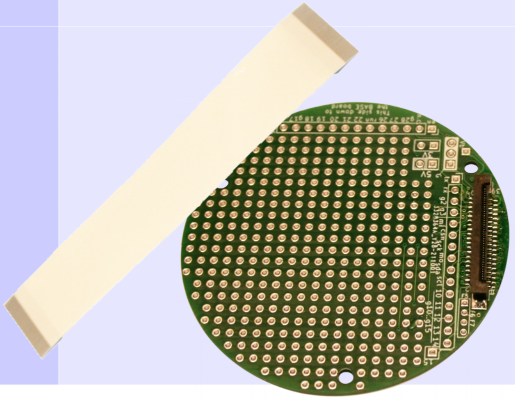


Pico Cansat BASE
Solder one of the Pico, wire a RFM69 for data transmission and use the PowerBoost to make it.

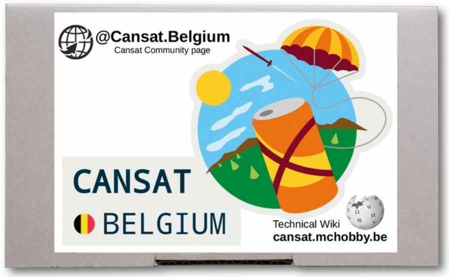


TMP36 – analog temperature sensor

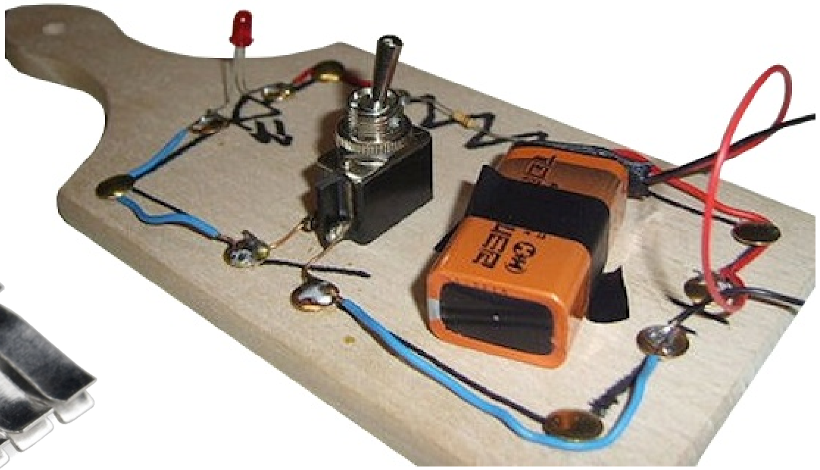
Transform the sensor voltage read to an easy-to-read temperature.

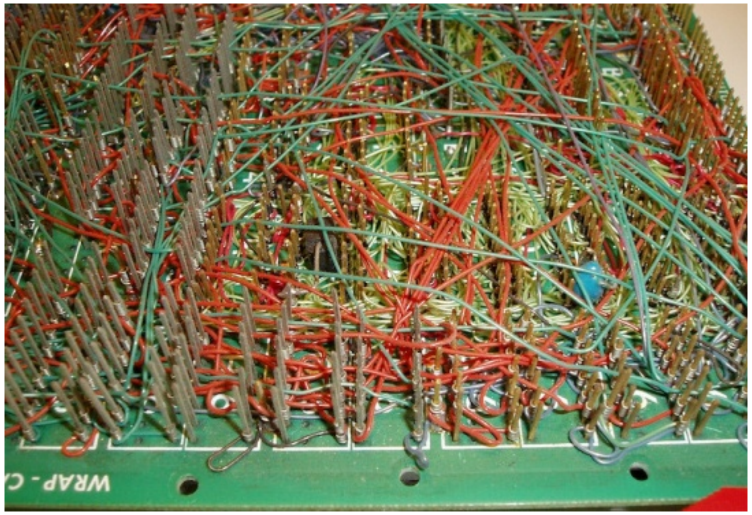


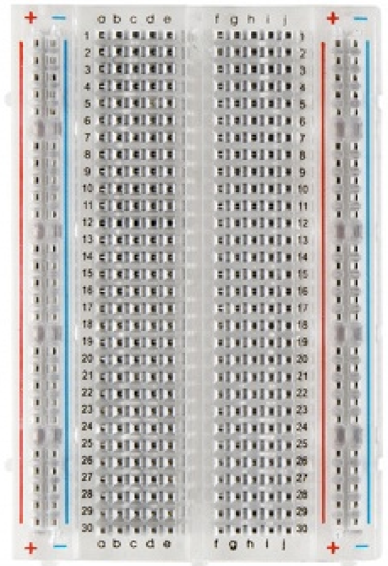
Pico Cansat PROTO
Consolidate your CAN + Prototyping area + all Microcontroler signals.

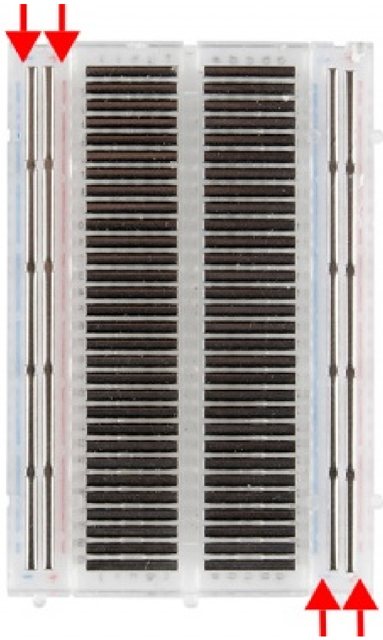


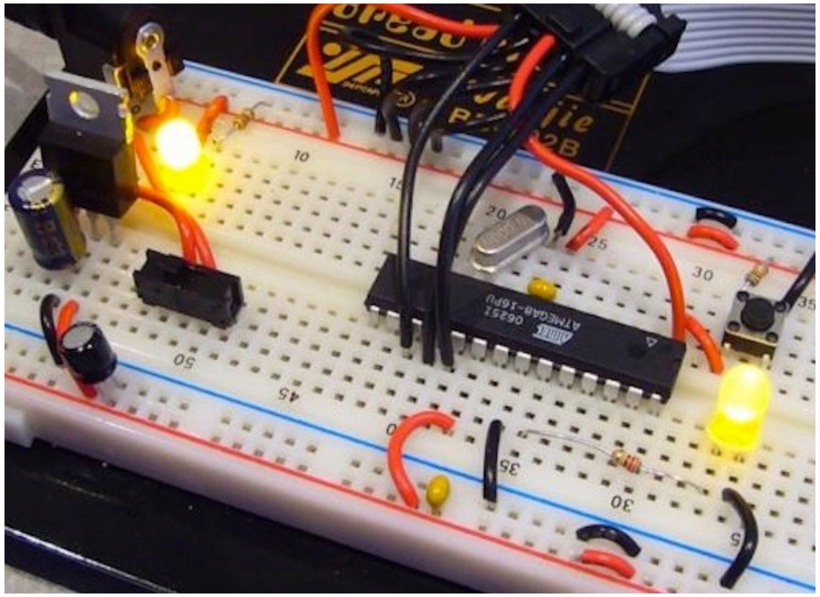
CanSat kit content - breadboard

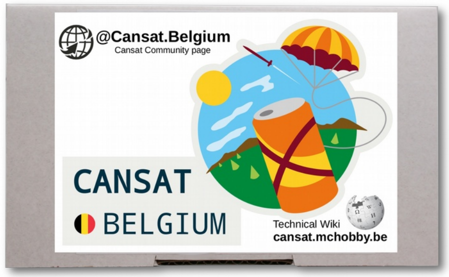










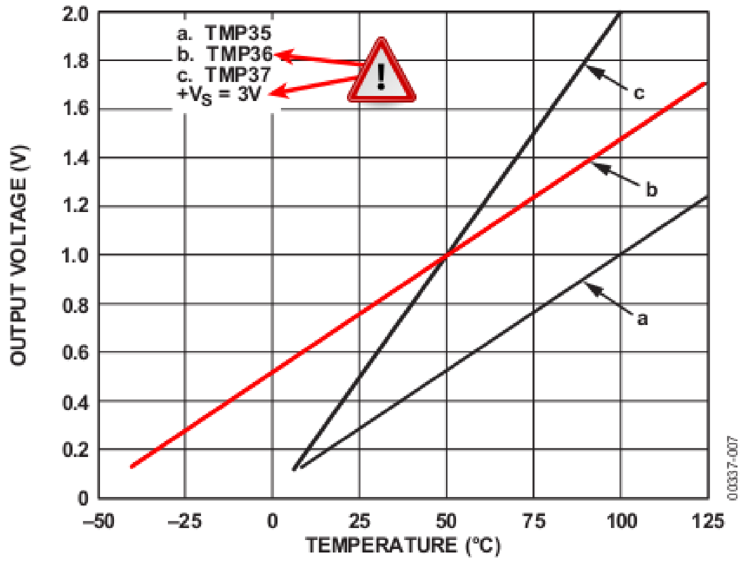


CanSat kit content - TMP36

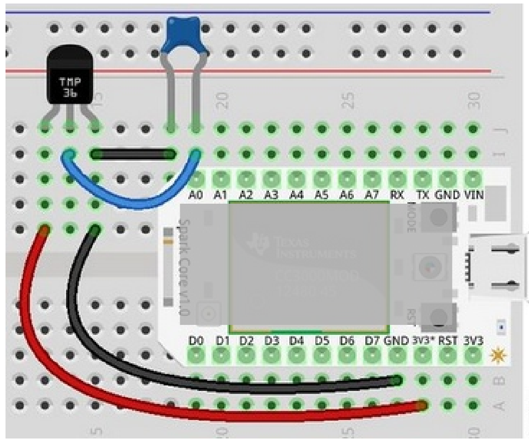
Here is the formula to use with a TMP36 powered at 3.3v:

```
Temp in °C = ( output_voltage_in_mV - 500) / 10
```

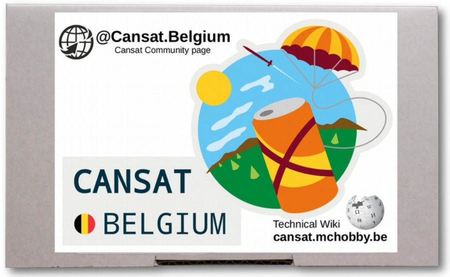
0V to 1.75V
-50°C to 125°C



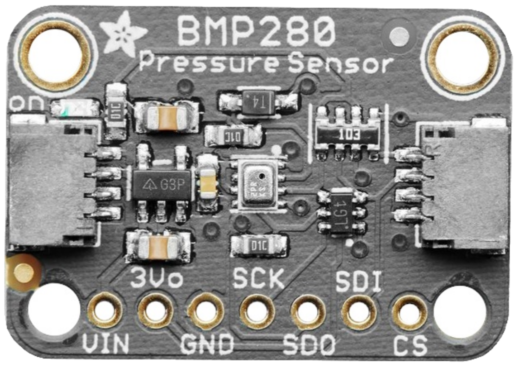
Tip & Trick – Filtering signal for greater stability



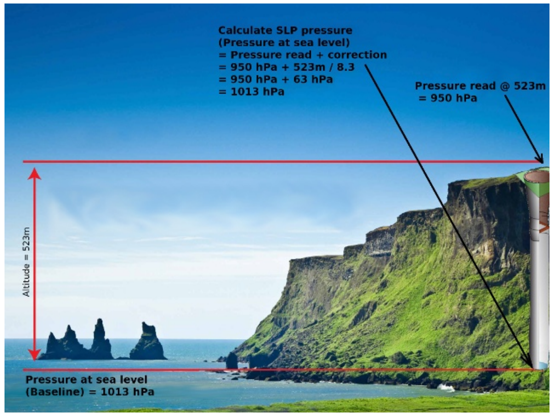
Depending on the other device you may add to your experiment, some electrical parasite may be rejected on the power supply stage.
In such case, you will notice inconsistencies when reading analog devices.
Adding a 0.01µF (10nF) ceramic capacity between ground and the microcontroler analog input can filter those parasites (seen as high frequency spikes).



CanSat kit content - BMP280



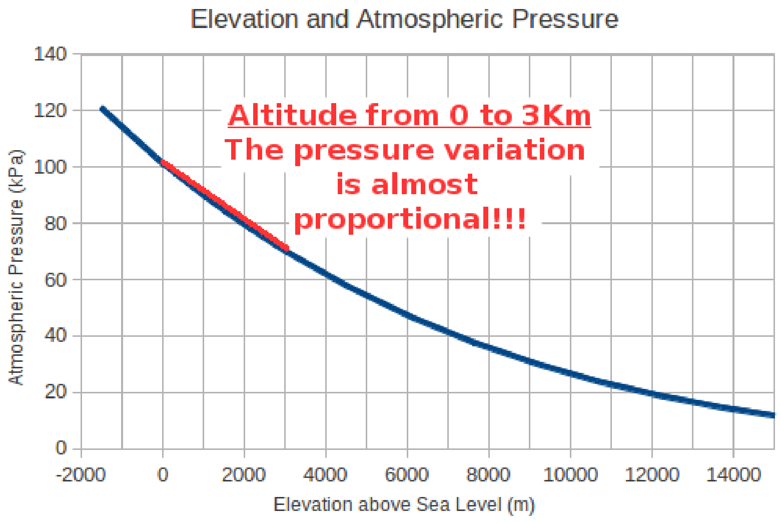
- Can use **I2C** or SPI bus
- Accuracy ± 1 hPa
  (= 100 Pa = 1 millibar)
- Pressure range:
  300...1100 hPa
- Temperature range:
  -40…85°C



Tip & Trick – Measure the altitude

As the pressure also change with the altitude, whenever the **pressure decrease from 1hPa the altitude increase of 8.3 meter**s.

The pressure sensor accuracy allows you to use the BMP280 to make an altimeter (accuracy of ±1m at worste, about 0.25m in best conditions)



Interesting Learning – Measure your absolute altitude

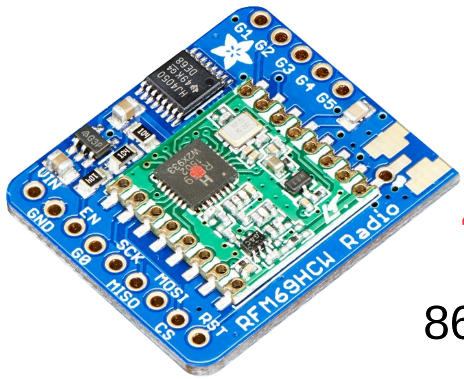
By using the today's pressure at sea-level, it is possible to calculate the absolute altitude of school / house. Compare it to one of the reference weather station near of your location.

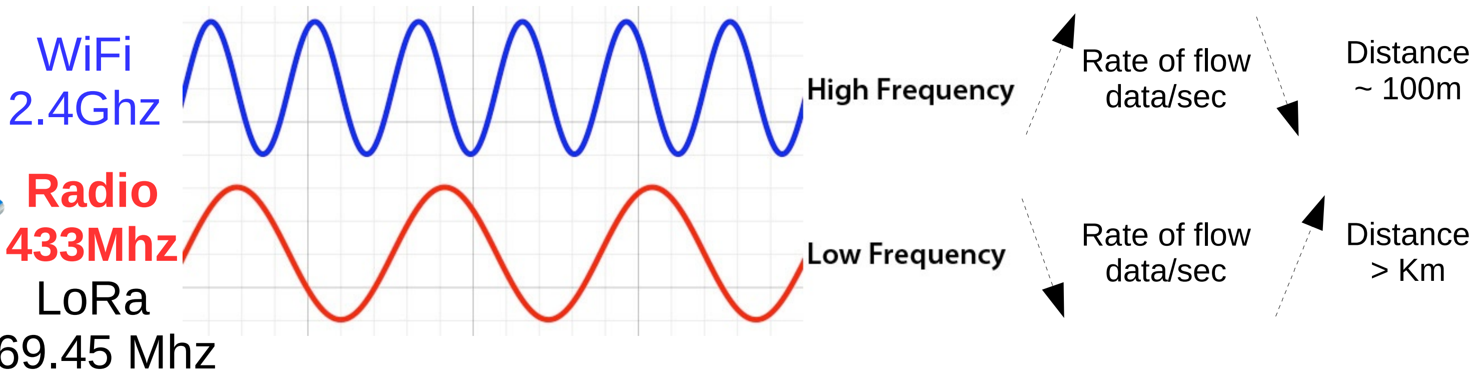
Interesting Learning – Calculate SLP (Sea Pressure Level)

Normalising your local pressure at the Sea Level (like reference wheater station does), you can compare your data with other reference station to make more accurate weather forecast.

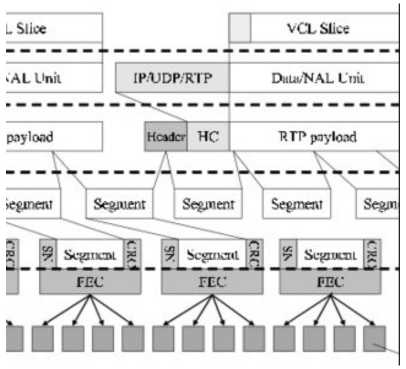
This is explained in wiki page related to BMP280 with detailed calculation.
See the picture with the well !

CanSat kit content - RFM69HCW





- Use the SPI bus
- Around 433 Mhz
- 50mA @ +13 dBm
  150mA @ +20 dBm
- Distance :
  500m to 2 Km (5 Km).

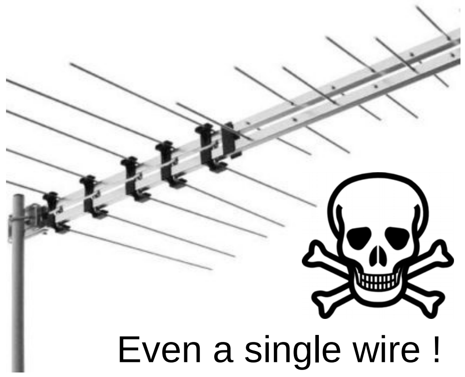


RFM69HCW - Packet Radio

This means that the module takes care of data coding, transmission, checksum, send retries, etc over the radio waves.

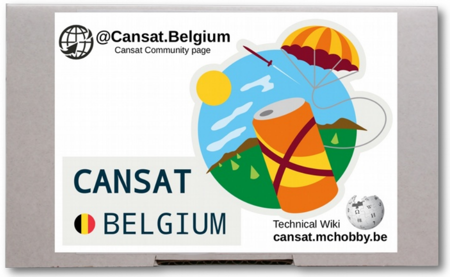
RFM69HCW – Frequency and encryption key

Antenna required !

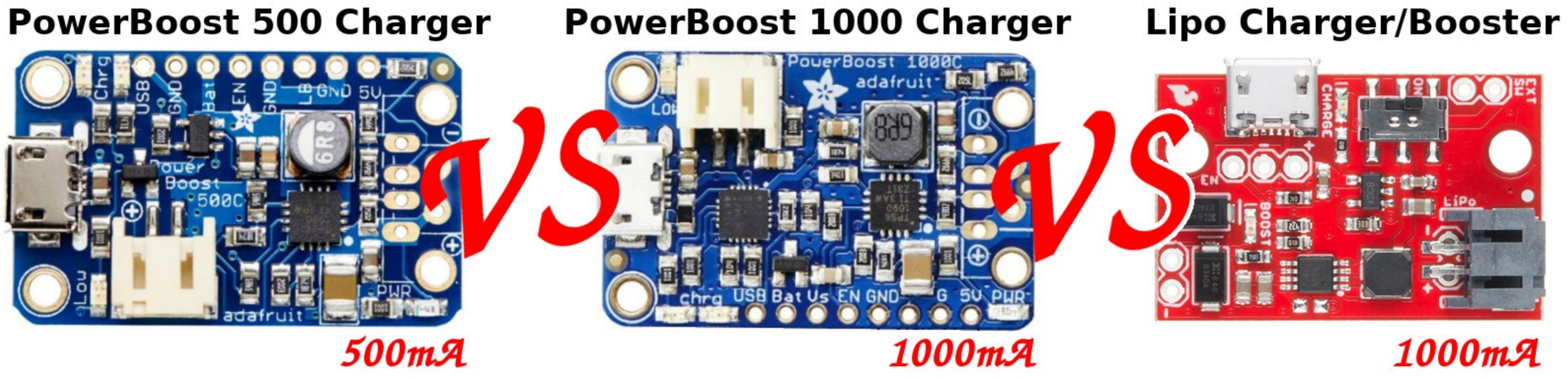


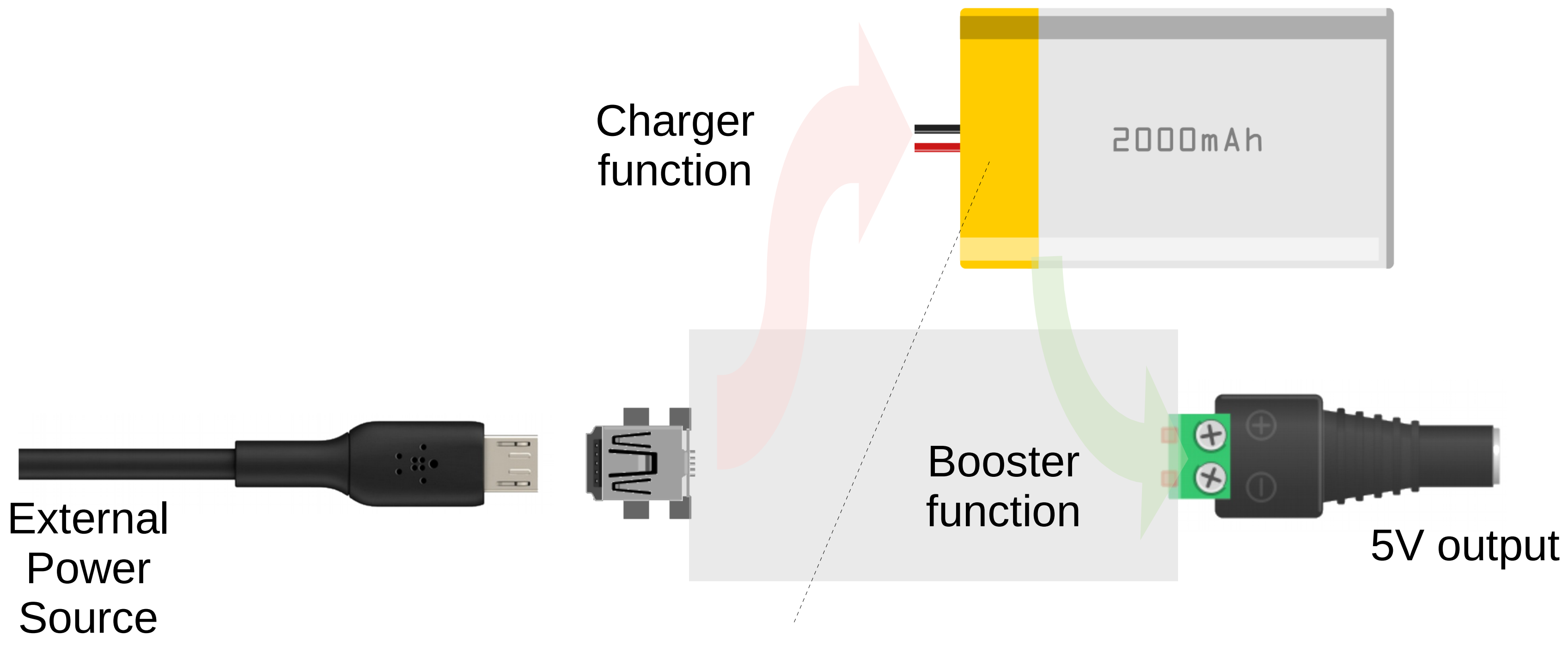
The RFM69HCW frequency can be adjuted (around 433Mhz) and data is AES encrypted with a key. Both are defined in the software and are the only parameters you really have to take care about.

The Sender and Receiver module must have :
The same frequency and
The same encryption key.

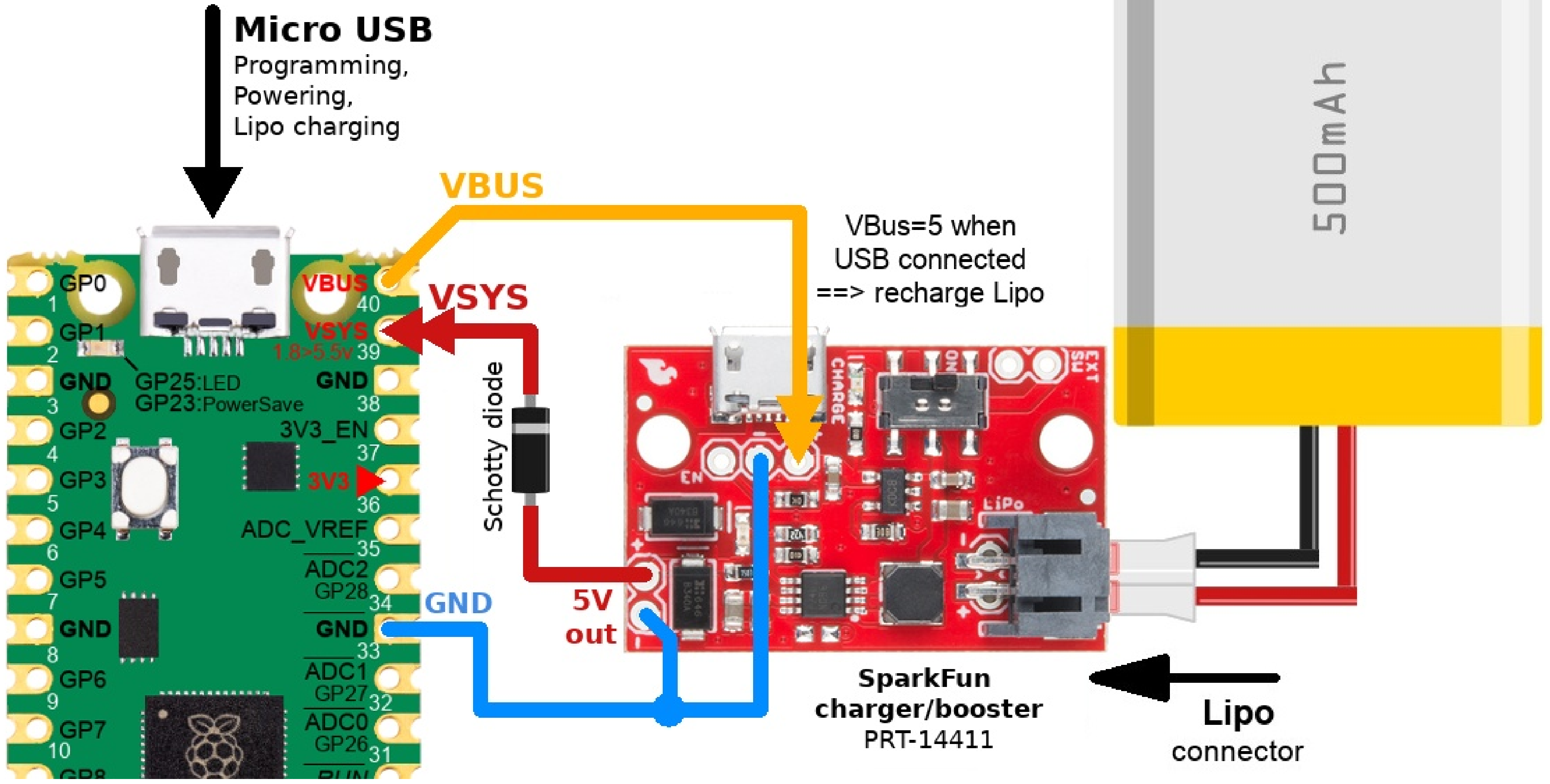


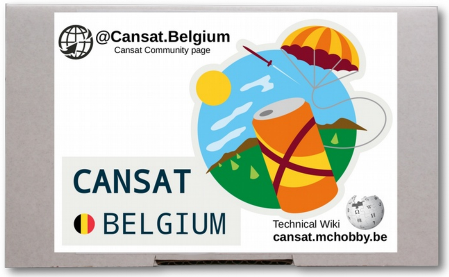
CanSat kit content – Lipo Charger/Booster





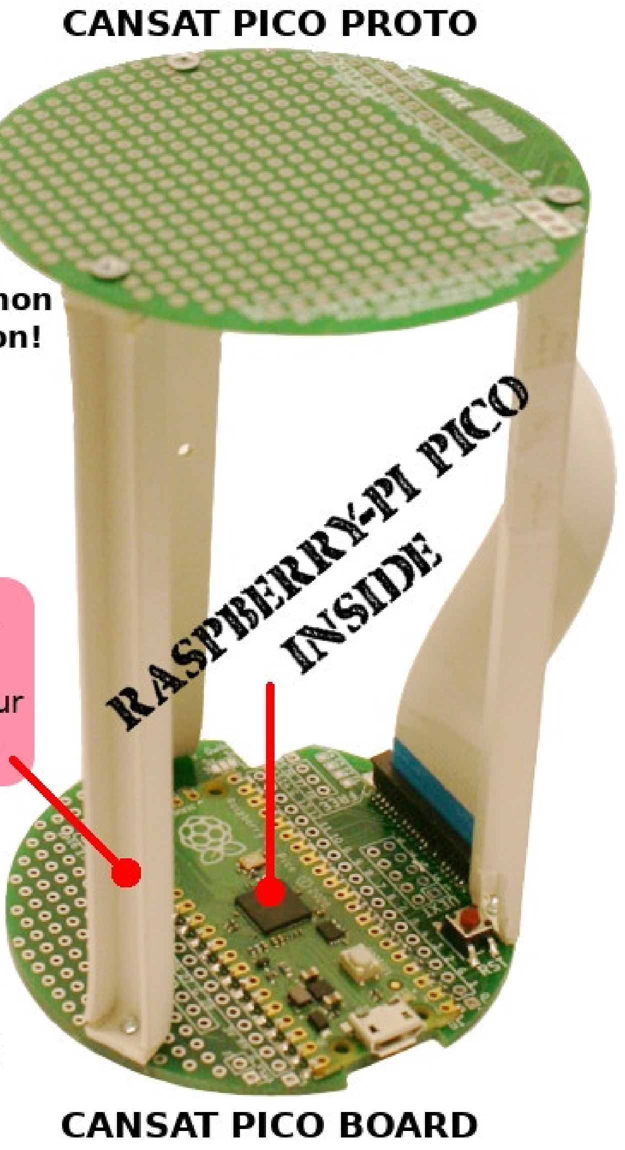
CanSat kit content – Lipo Charger/Booster





CanSat Pico kit

Not weird design but smart ideas



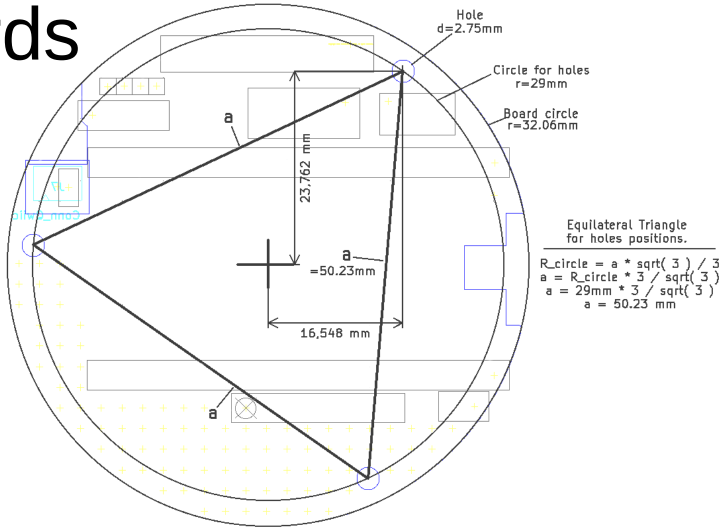
Pure MicroPython implementation!

TIPS'n' TRICKS

Design your own spacer. Customise your setup height.

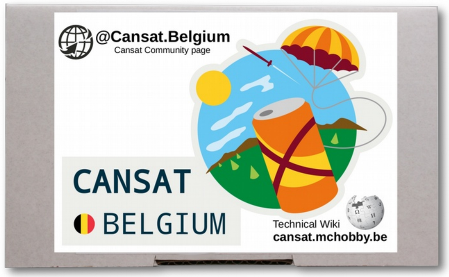
Can also be programmed with C++ & Arduino

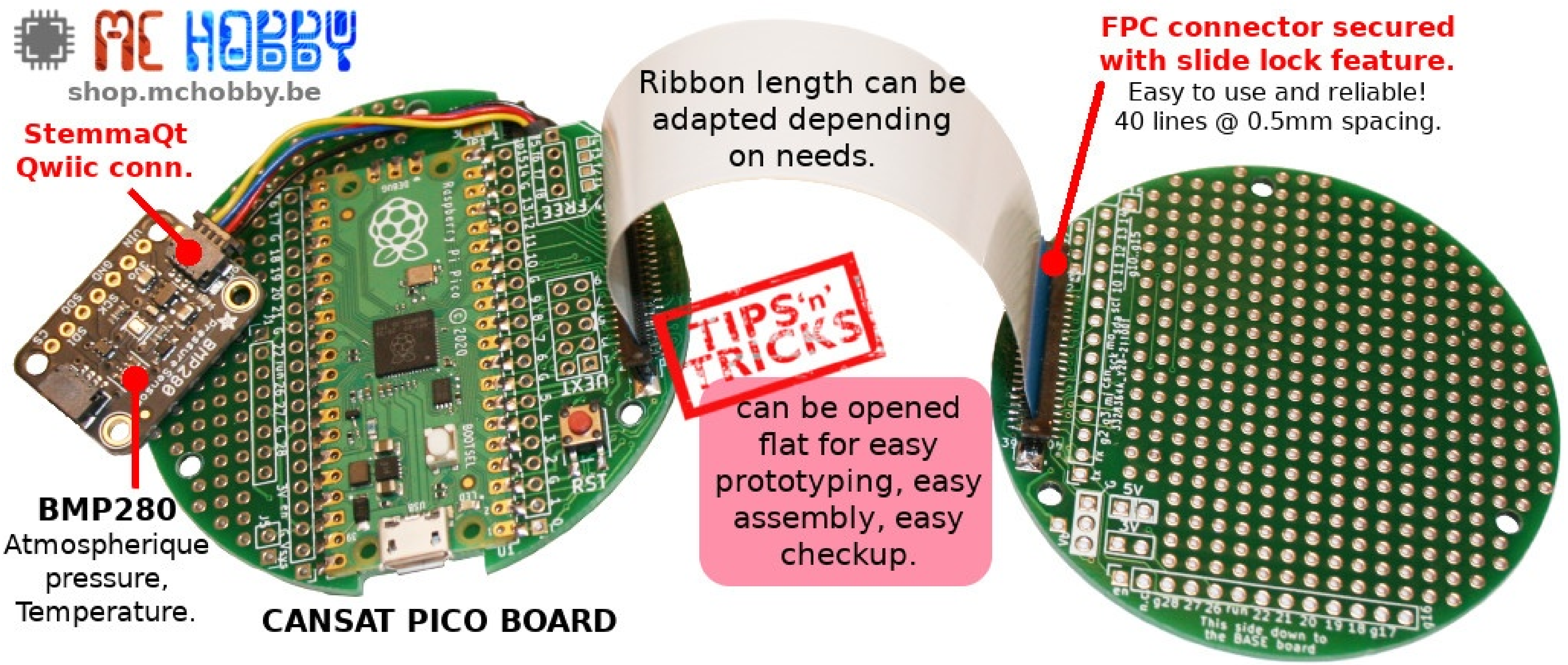
- Base CAN structure
- LOT of SPACE avail.
- Free HEIGHT design
- Add intermediate levels
- 1mm wall tickness around boards



CanSat Pico kit

Not weird design but smart ideas





BMP280 Atmospherique pressure, Temperature.

Ribbon length can be adapted depending on needs.

FPC connector secured with slide lock feature. Easy to use and reliable! 40 lines @ 0.5mm spacing.

can be opened flat for easy prototyping, easy assembly, easy checkup.

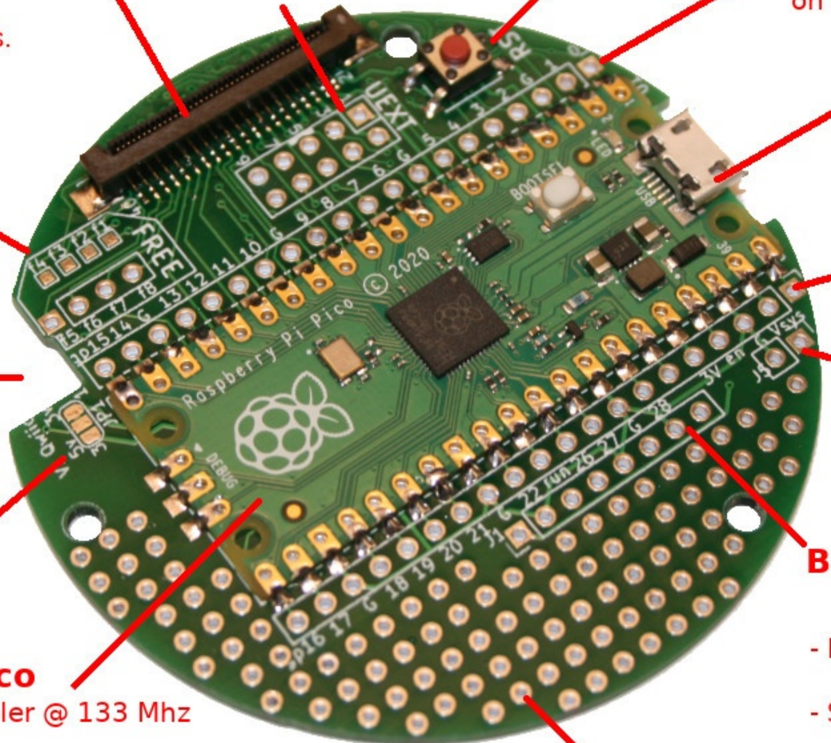
CANSAT PICO BOARD

CANSAT PICO PROTO

CanSat Pico kit

CANSAT Pico Board

(front / top)



Ribbon connector to prototyping board
transfert all pico signals + 8 user signals to prototyping boards.

UEXT connection
Can be used to connect external modules (EG:radio)
- 3.3V power
- I2C bus
- SPI bus
- UART

Reset button

Pico GPIOs
Through holes are available on the both sides.

Micro-USB
- Programming
- Powering
- Battery charging
- Debugging

Pico GPIOs
Vin pin

Power Enable
activate/deactivate the external battery management.

user signals connections
- 4 through holes (avail. on both sides)
- 4 pads (top only)

Openning
Allow to transfer cables between top and bottom.

Qwiic/StemmaQt Voltage selection
Default to 3.3V.

Raspberry-Pi Pico
Dual core microcontroler @ 133 Mhz
- 2 Mio Flash
- 264 Kio RAM
- Micropython & C++

Battery Managment Connector
- PowerBoost 500 Charger (Adafruit 1944)
- Solder on bottom of the board.

Prototyping Area
Through holes are available on the both sides.

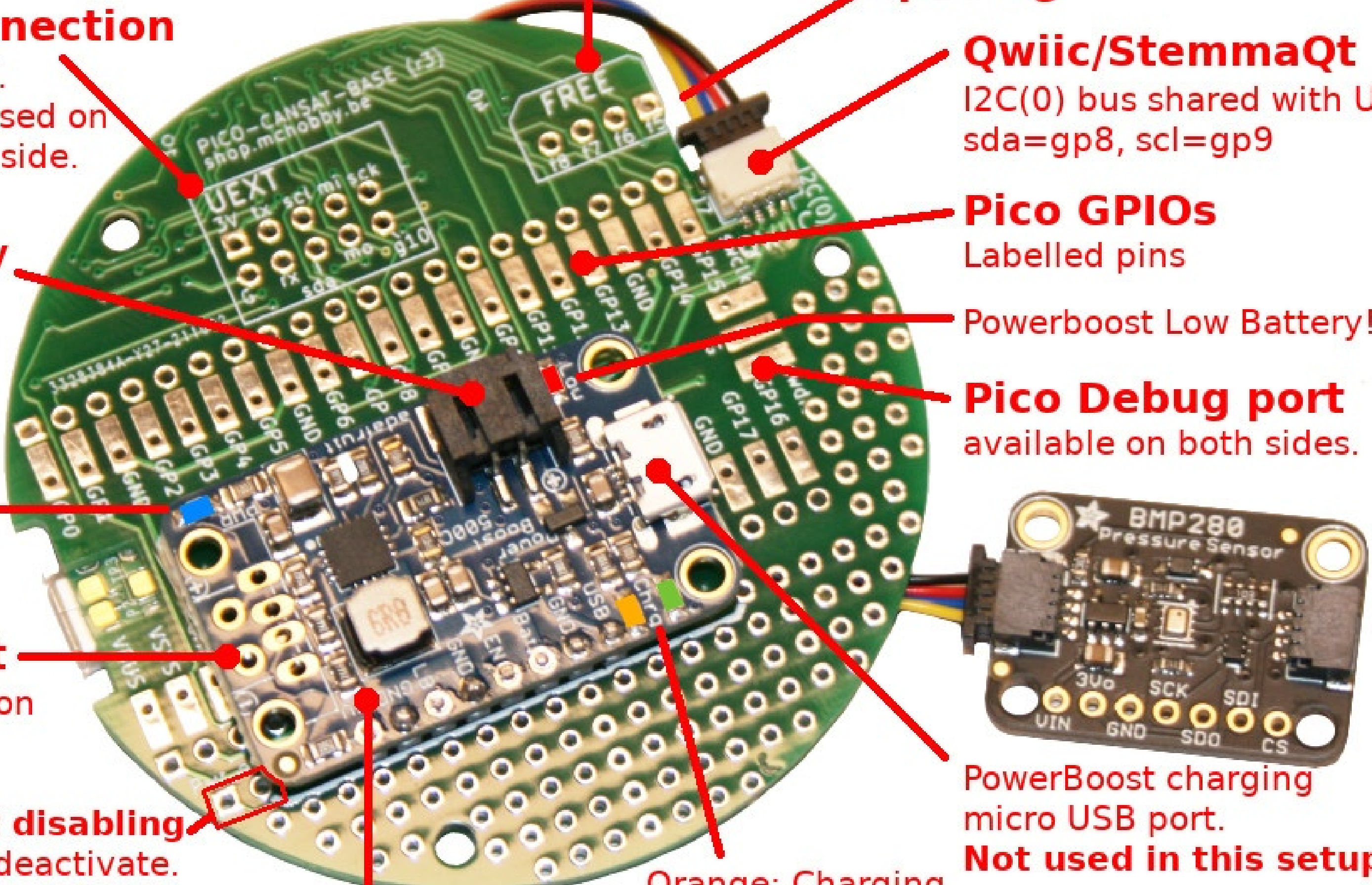
UEXT

	+3.3V	1	2	GND
UART(0)	gp0=tx	3	4	rx=gp1
I2C(1)	gp9=scl	5	6	sda=gp8
SPI(0)	gp4=miso	7	8	mosi=gp7
	gp6=sck	9	10	csn=gp10

shop.mchobby.be

CanSat Pico kit

CANSAT Pico Board
(rear / back)



user signals connections 4 through holes (avail. on both sides)

Opening

Qwiic/StemmaQt
I2C(0) bus shared with UEXT.
sda=gp8, scl=gp9

Pico GPIOs
Labelled pins

Powerboost Low Battery!

Pico Debug port
available on both sides.

UEXT Connection
Fully labelled.
Can also be used on the this back side.

Lipo Battery connector

PowerBoost under power.

5V output
Extra output on back side.

PowerBoost disabling
short pins to deactivate.

PowerBoost 500 Charger
5V @ 500mA capability from Lipo battery.
Battery recharge capability when Pico connected to power source.
Feeds the Pico and project with 5V.

Orange: Charging.
Green: Fully charged.

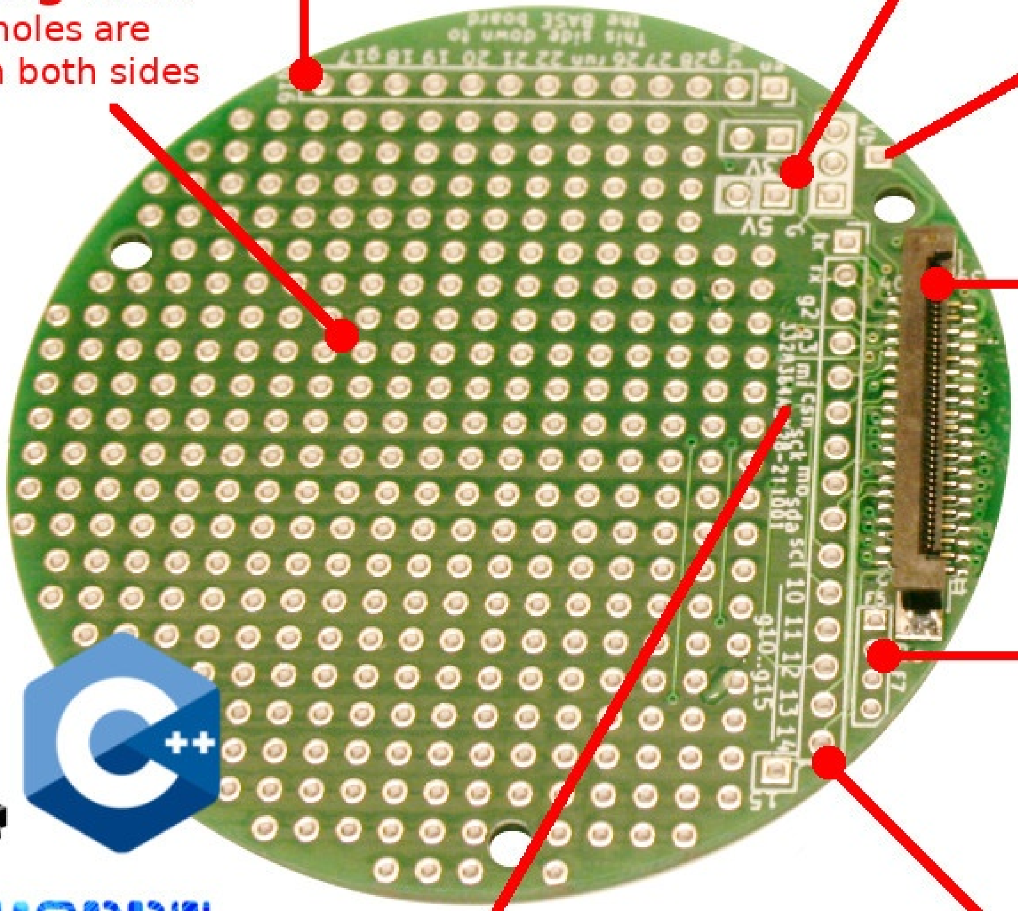
PowerBoost charging micro USB port.
Not used in this setup.

shop.mchobby.be

CanSat Pico kit

CANSAT Pico Proto

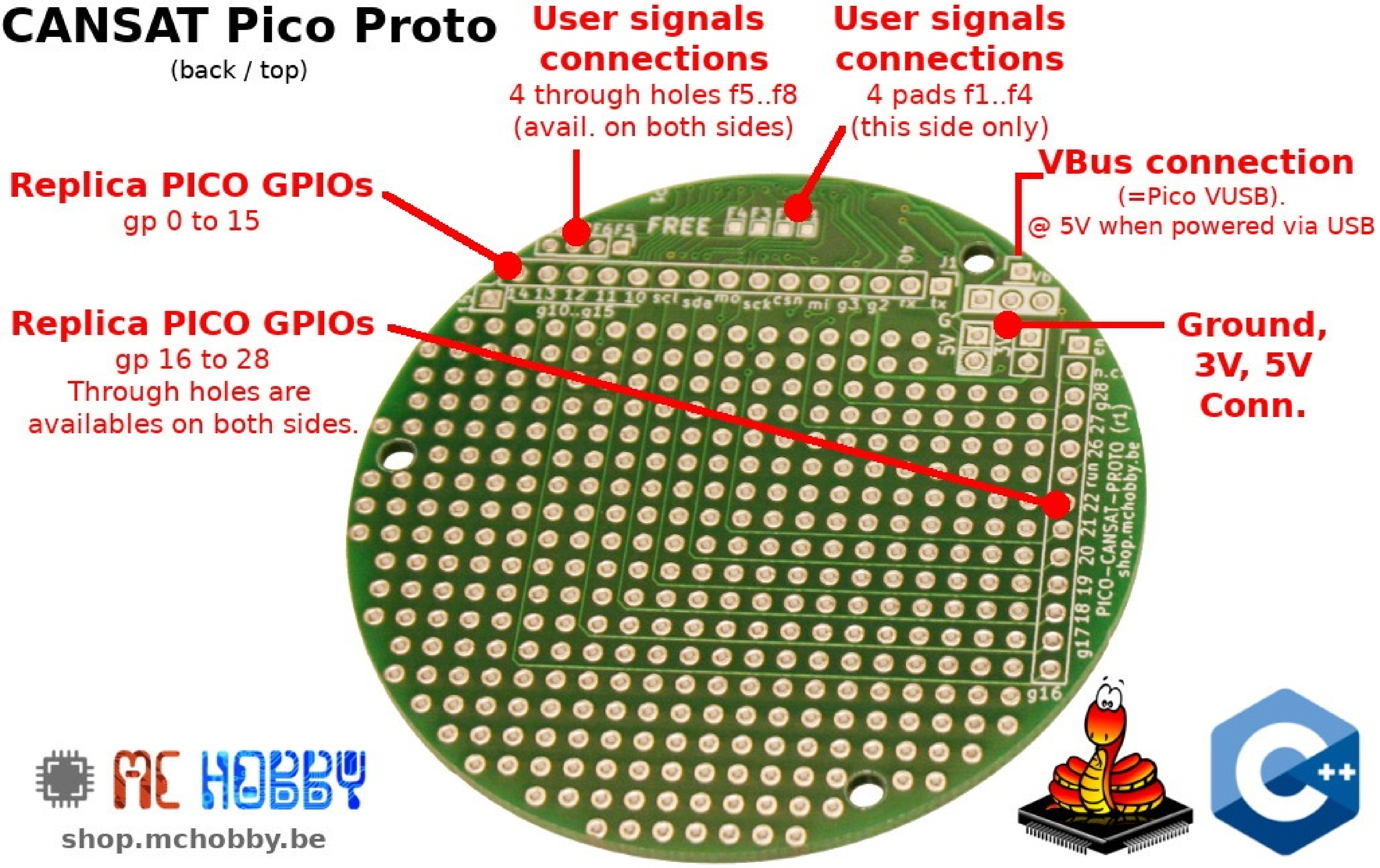
(front / bottom)



Replica PICO GPIOs
gp 16 to 28
Through holes are availables on both sides.
Protoyping Area
Through holes are available on both sides
Ground, 3V, 5V connections
VBus connection
(=Pico VUSB).
@ 5V when powered via USB
Ribbon connector from CANSAT Pico Board
receive all pico signals + 8 user signals.
User signals connections
4 through holes f5..f8
(avail. on both sides)
Using UART, I2C, SPI pin names
(pins shared with UEXT connector).
Replica PICO GPIOs
gp 0 to 15

shop.mchobby.be

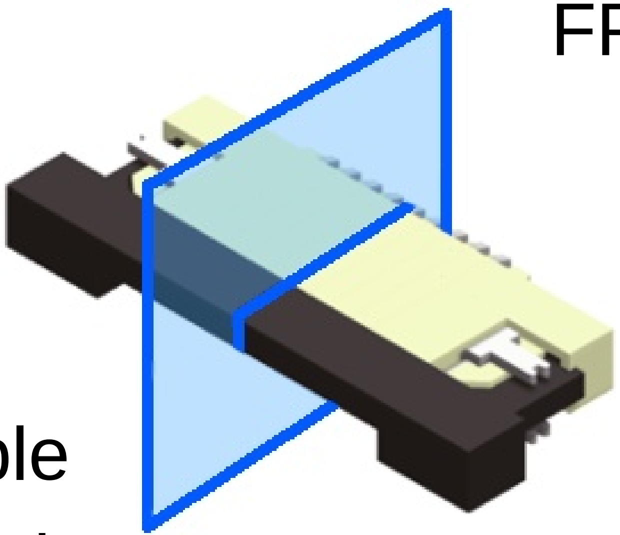
CanSat Pico kit

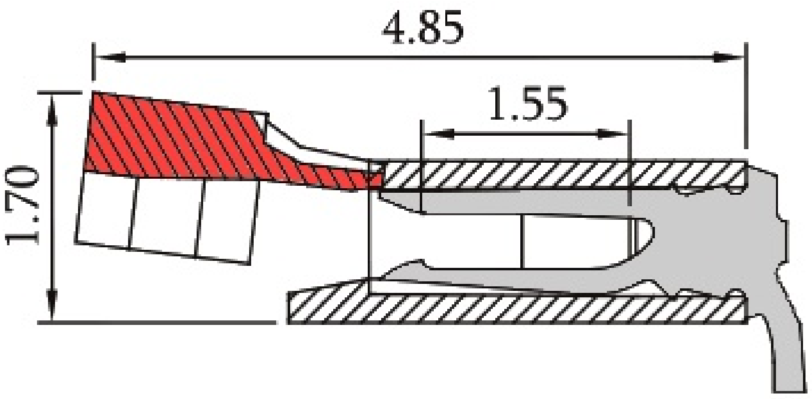


shop.mchobby.be

CanSat Pico kit

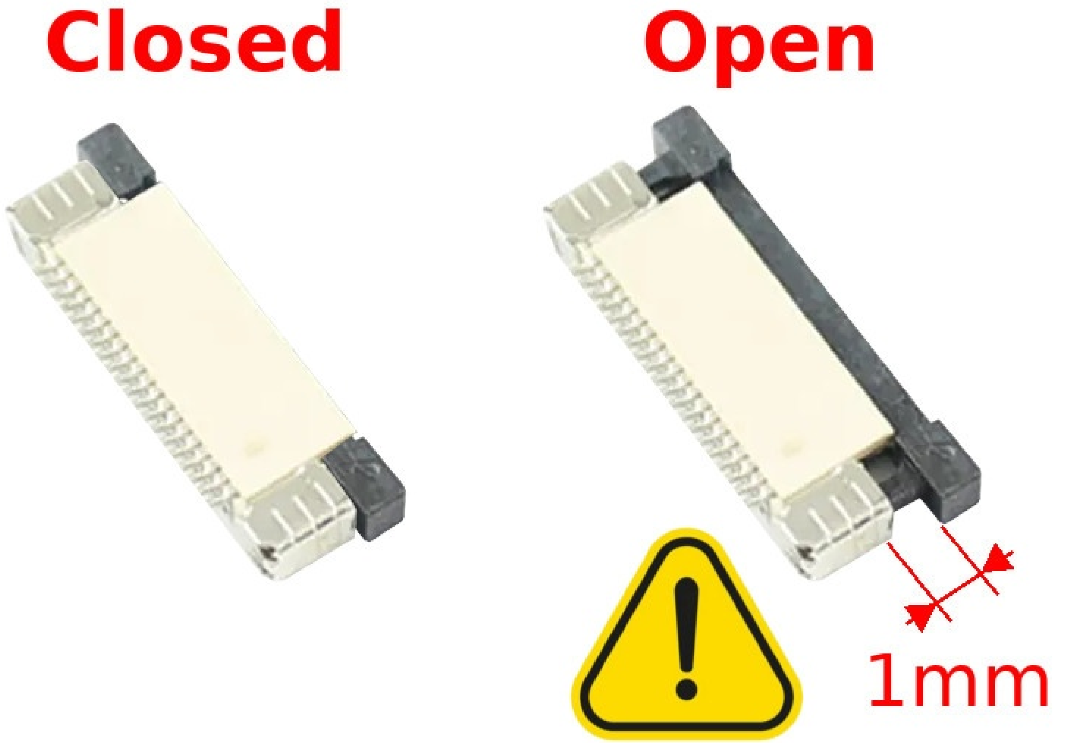
FPC & FFC Connector



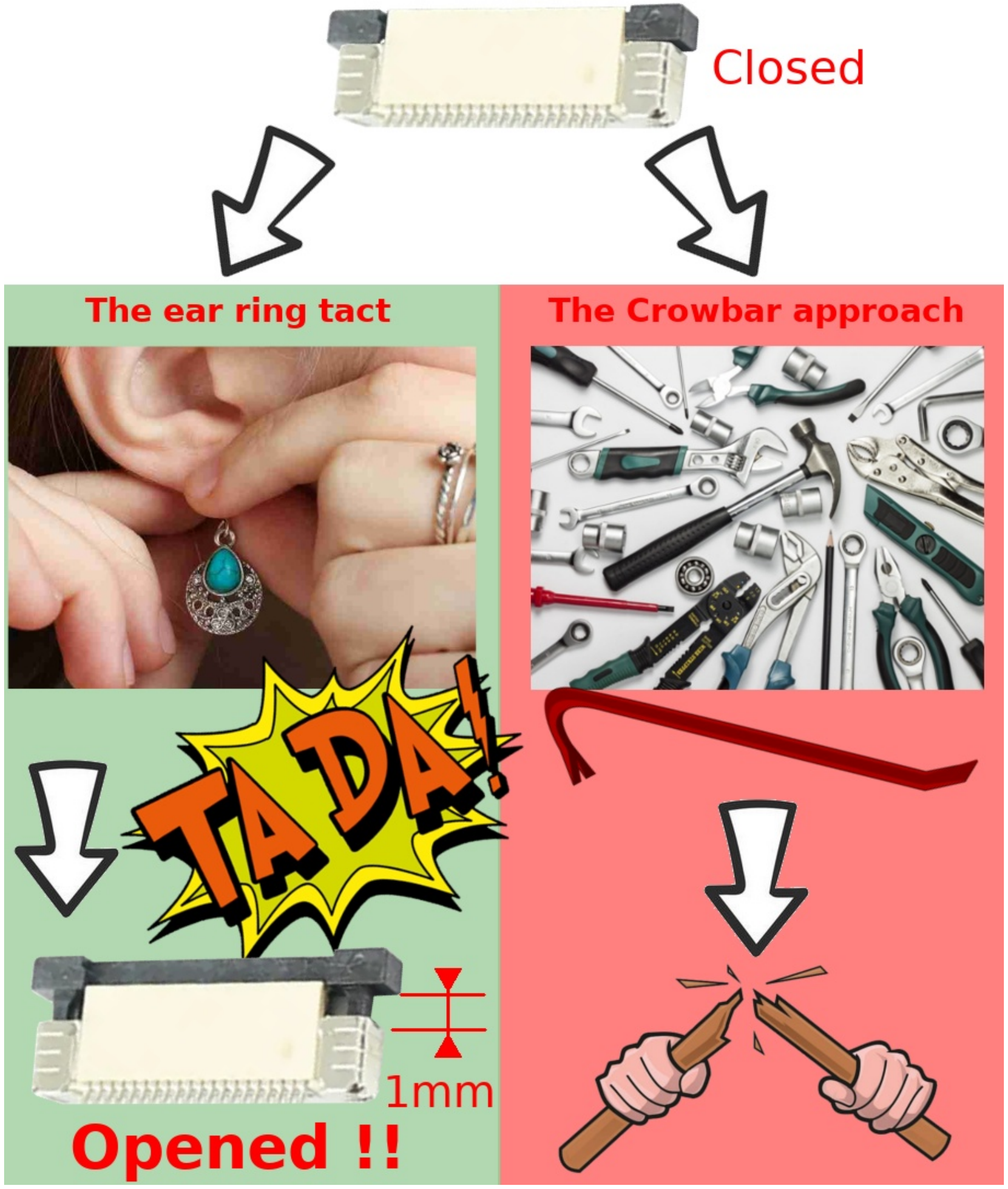


Flexible
Printed
Circuit

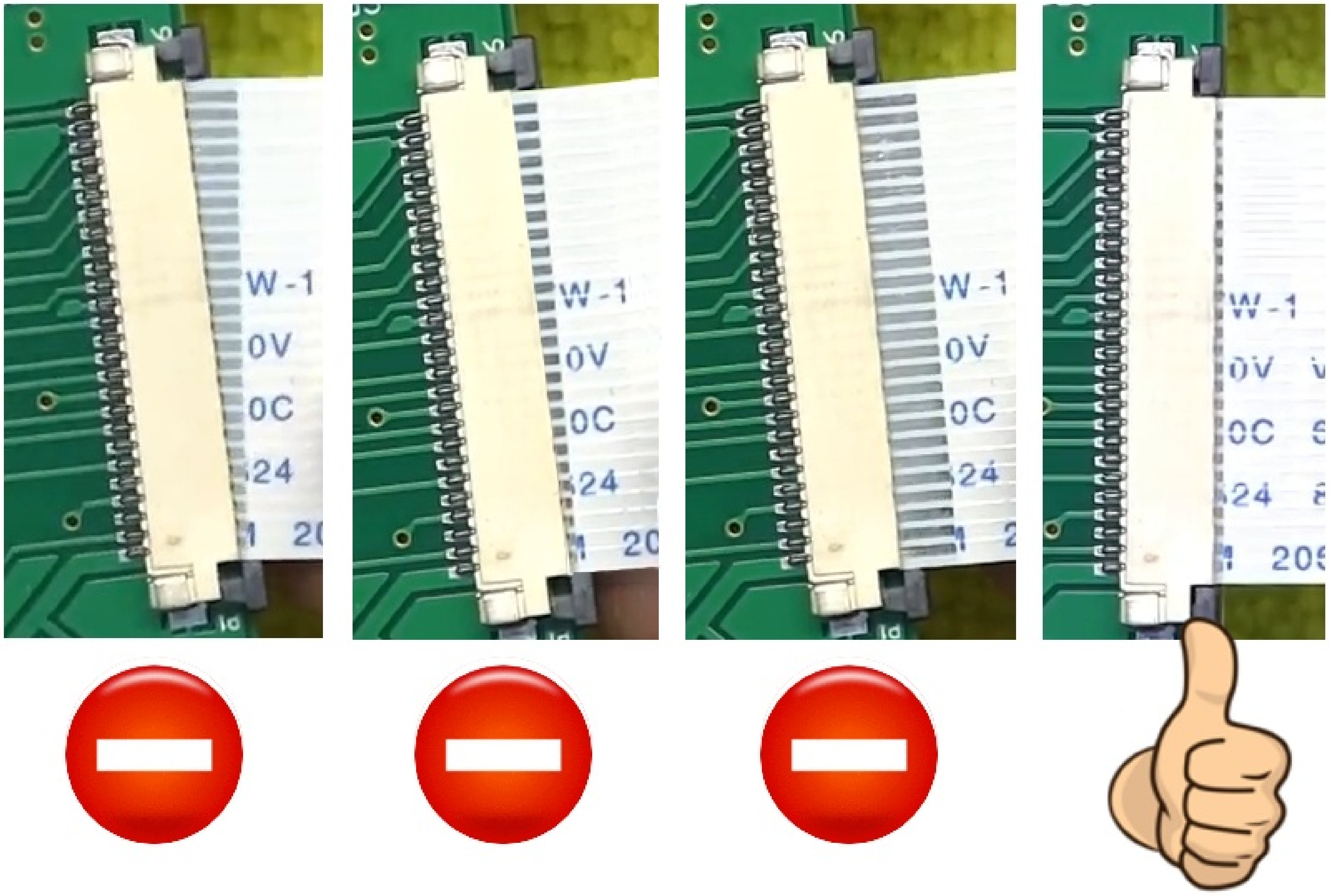
Flexible
Flat
Cable



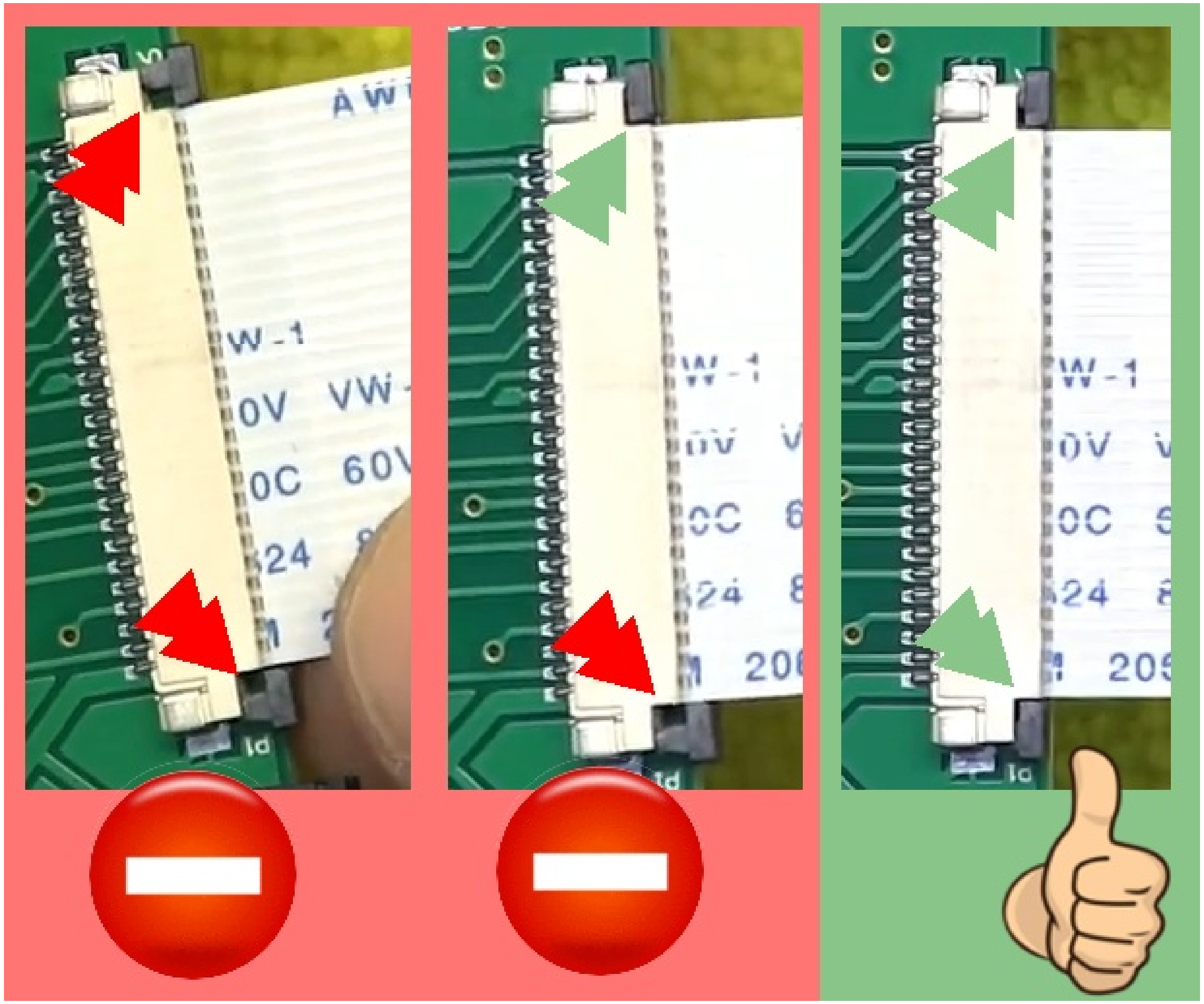
CanSat Pico kit



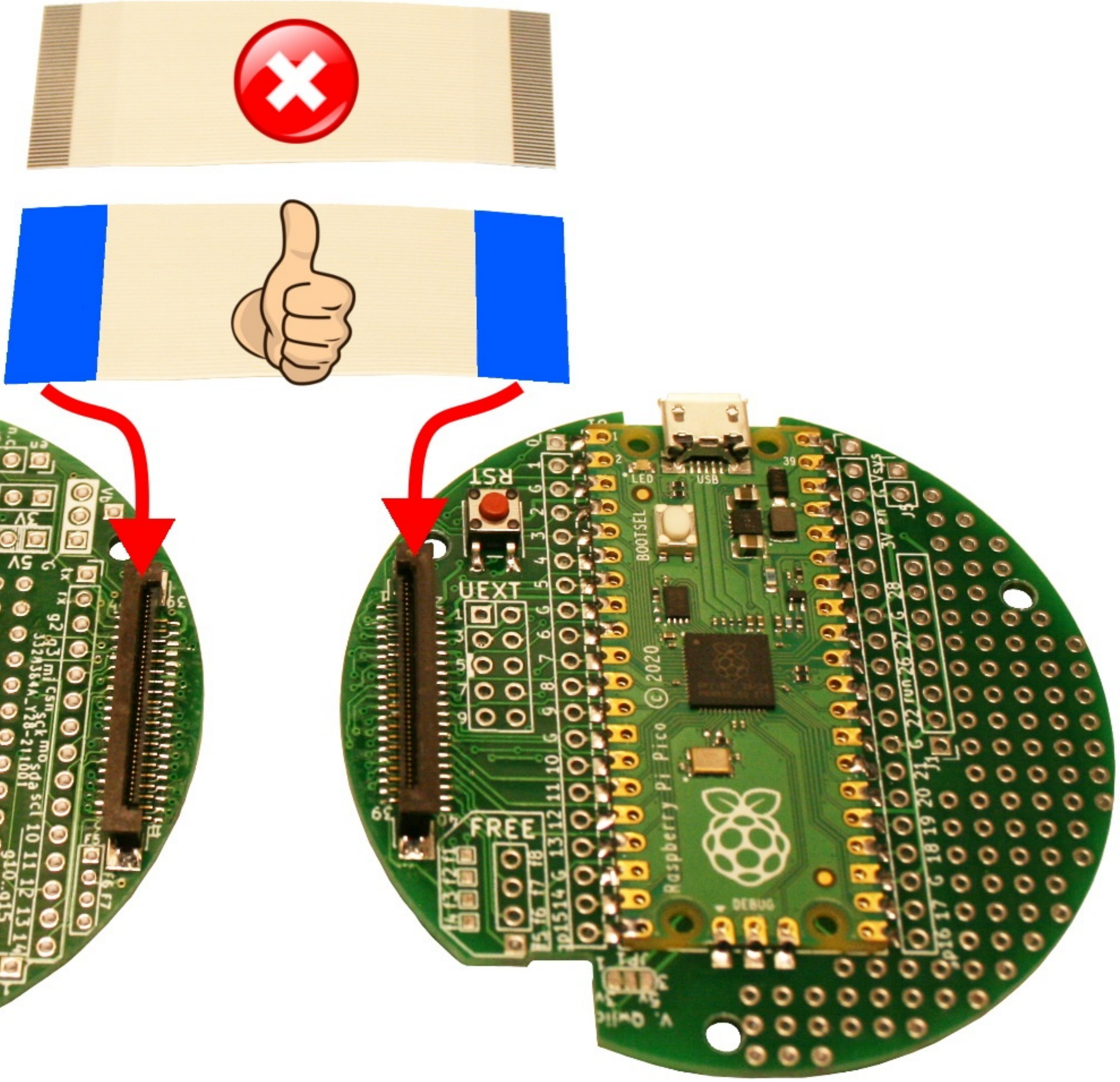
CanSat Pico kit

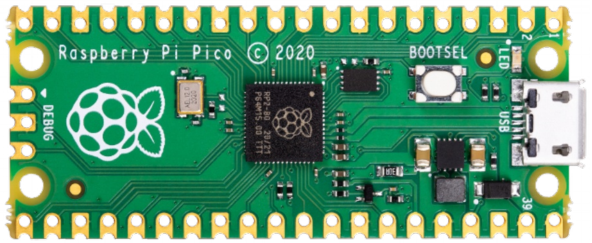


CanSat Pico kit

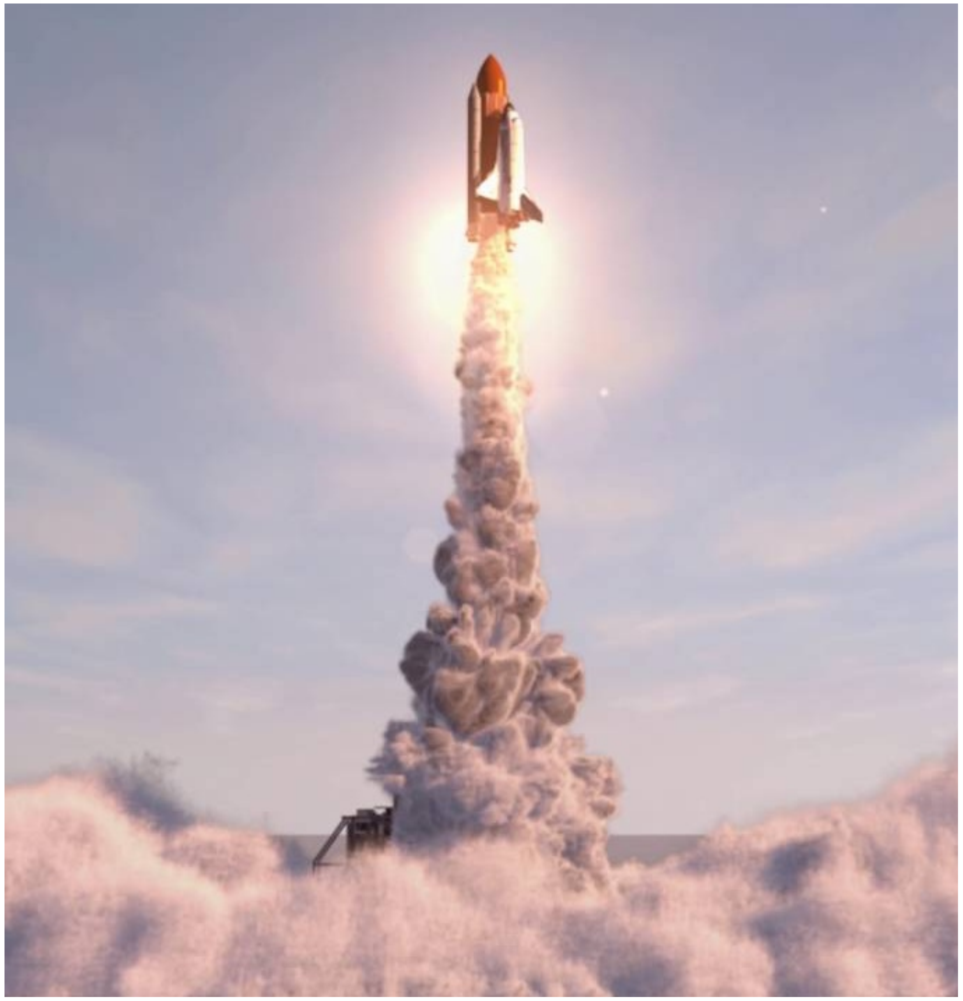


CanSat Pico kit



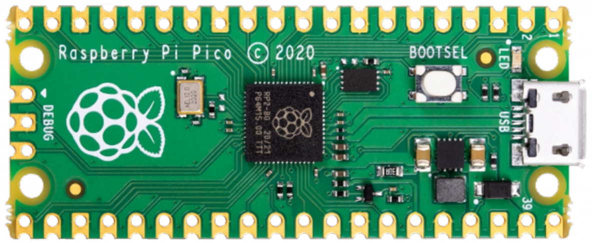


Raspberry-Pi Pico



A bunch of power
to launch your project

Raspberry-Pi Pico



3V

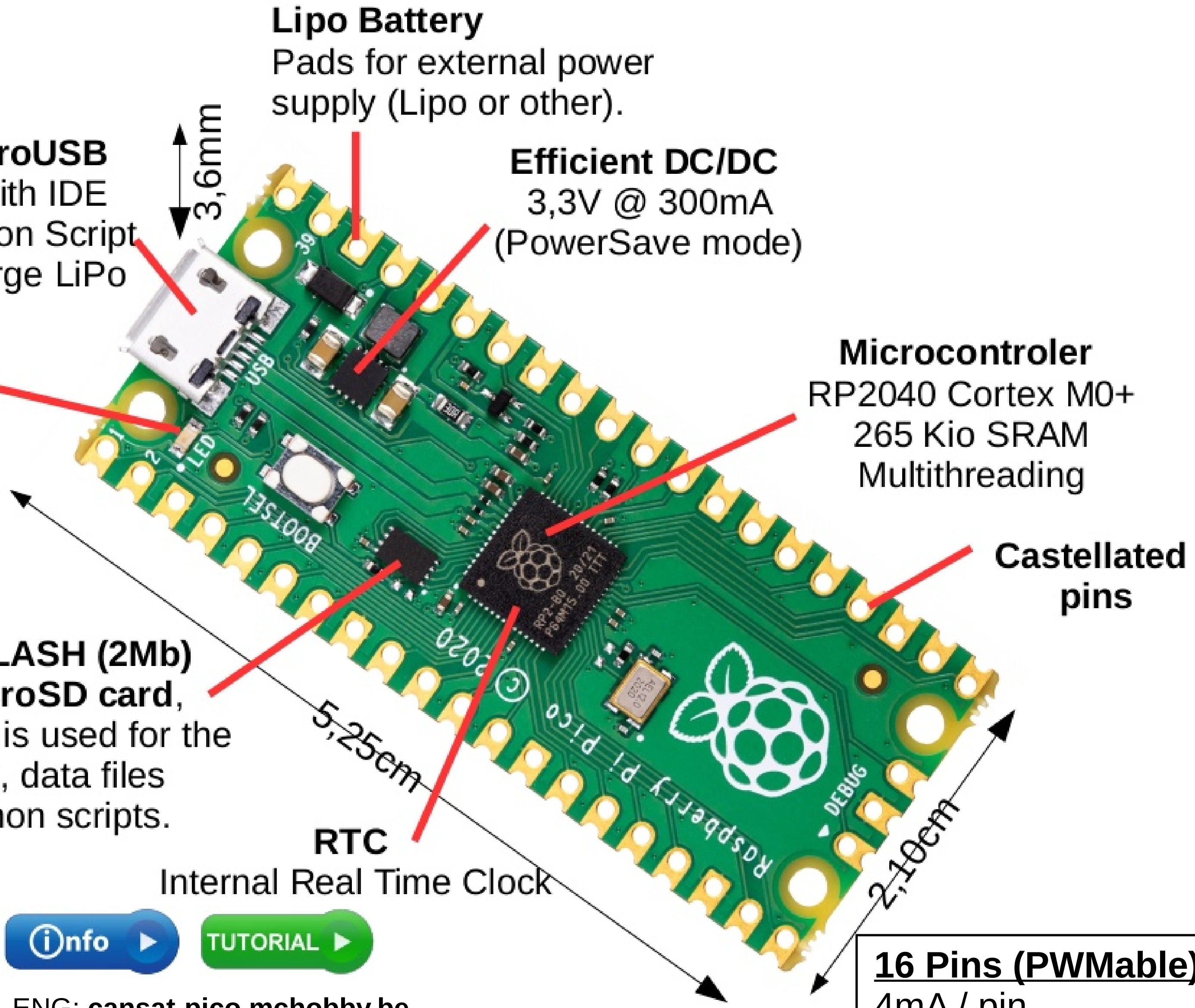
300mA max

3 Analog inputs
12 bits resolution.
Value 0 – 4096

Note :
Arduino IDE use 10 bit resolution by default (0 – 1024) but this could be changed with analogReadResolution(12)

2 cores
real multithreading under MicroPython

Serial still available !

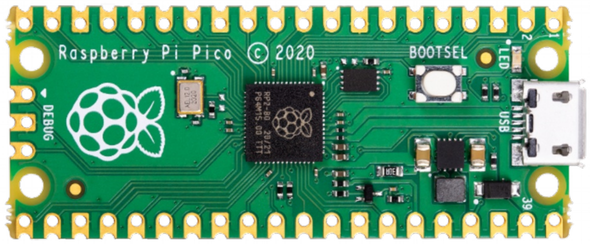


16 Pins (PWMable)
4mA / pin.
50MA max on chip

Pico Power vs *Arduino Uno*
Clock : **133 MHz** vs *16 MHz for Uno*
RAM : **265 Kb** vs *2Kb for Uno*
Flash : **2048 Kb** vs *32Kb for Uno*
Real Time Clock : **Yes** vs *none for Uno*

Info ▶ TUTORIAL ▶

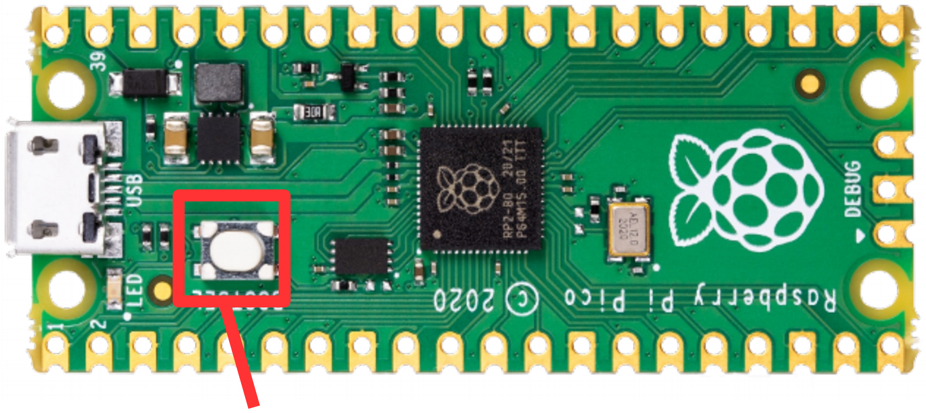
ENG: **cansat-pico.mchobby.be**
ENG: www.raspberrypi.com/documentation/microcontrollers/
FR : shop.mchobby.be/product.php?id_product=2271



Raspberry-Pi Pico

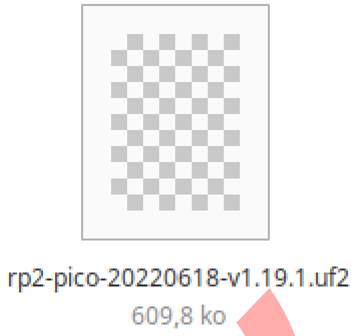
Putting MicroPython on the board

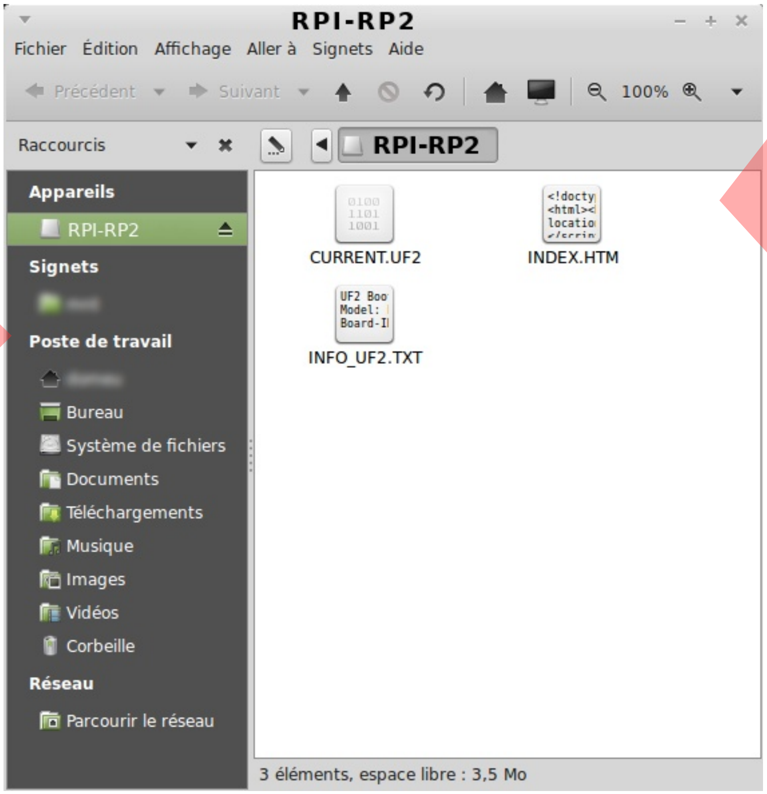
Manual activation of the bootloader

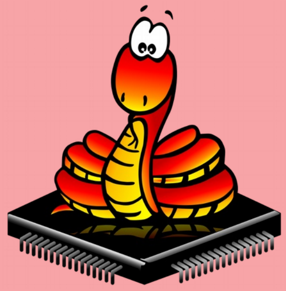


Press button while power on

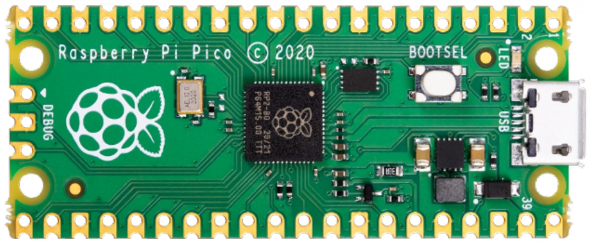
MicroPython firmware can be downloaded from **micropython.org**







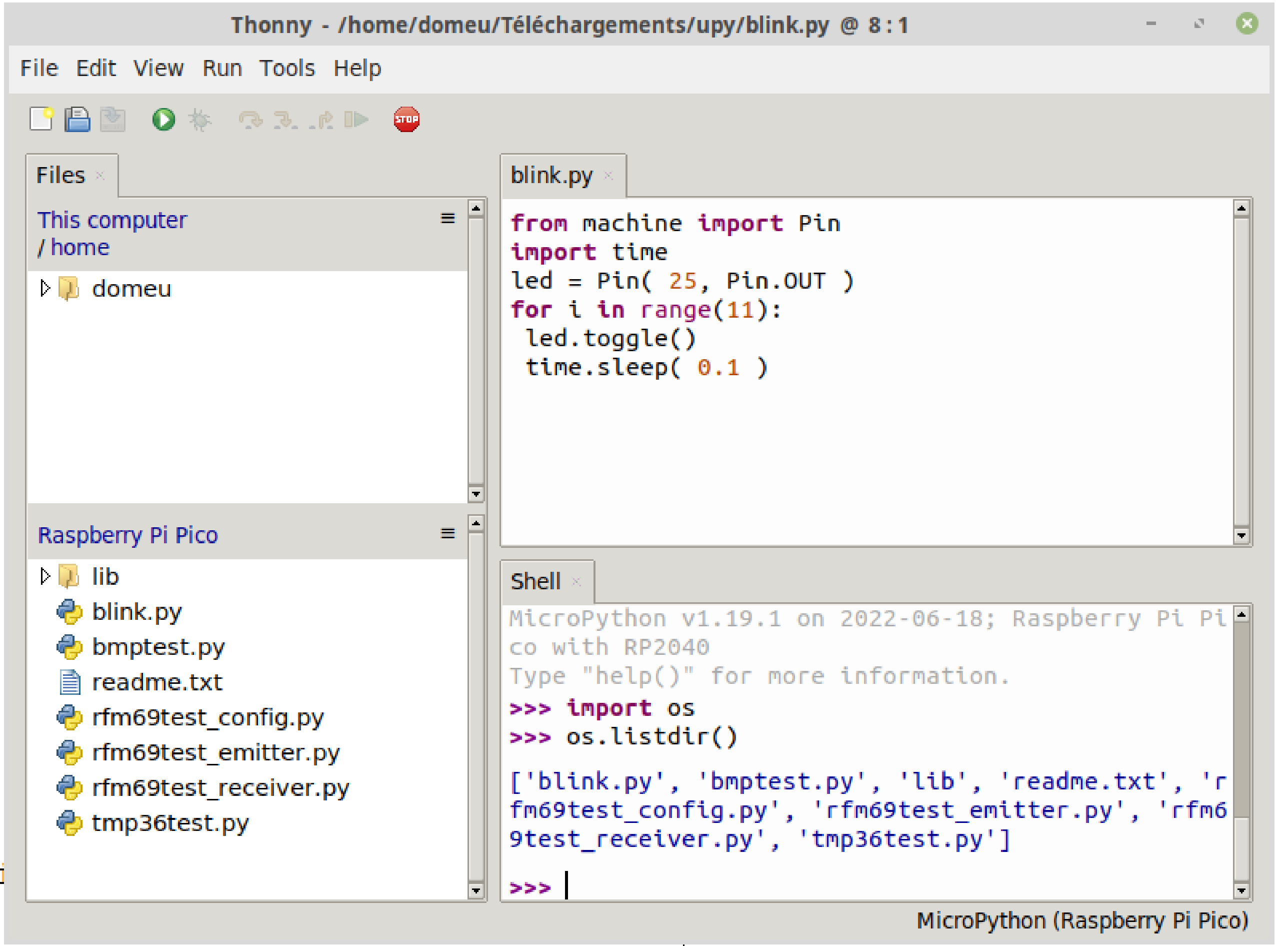
Once installed…. MicroPython doesn't requires anymore the bootloader activation.



Raspberry-Pi Pico

Pre-initialized MicroPython filesystem

Thonny IDE
(graphical)



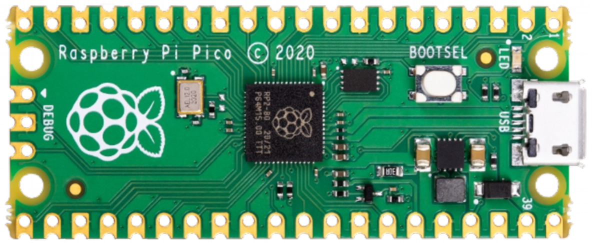
```python
from machine import Pin
import time
led = Pin( 25, Pin.OUT )
for i in range(11):
 led.toggle()
 time.sleep( 0.1 )
```

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> import os
>>> os.listdir()

['blink.py', 'bmptest.py', 'lib', 'readme.txt', 'rfm69test_config.py', 'rfm69test_emitter.py', 'rfm69test_receiver.py', 'tmp36test.py']

>>> 
```

MPRemote
(command line)

```
MPY: soft reboot
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pi
Type "help()" for more information.
>>> import os
>>> os.listdir()
['blink.py', 'bmptest.py', 'lib', 'readme.txt', 'rfm69test_config.py', 'rfm69test_emitter.py',
 'rfm69test_receiver.py', 'tmp36test.py']
>>> os.listdir('lib')
['__ap.py', '__append.py', '__df.py', '__hexdump.py', '__ifconfig.py', '__ptest.py', '__touch.
py', '__uname.py', '__wifi.py', 'bme280.py', 'mshell.py', 'mshell.txt', 'pye.py', 'rfm69.py']
>>> 
```

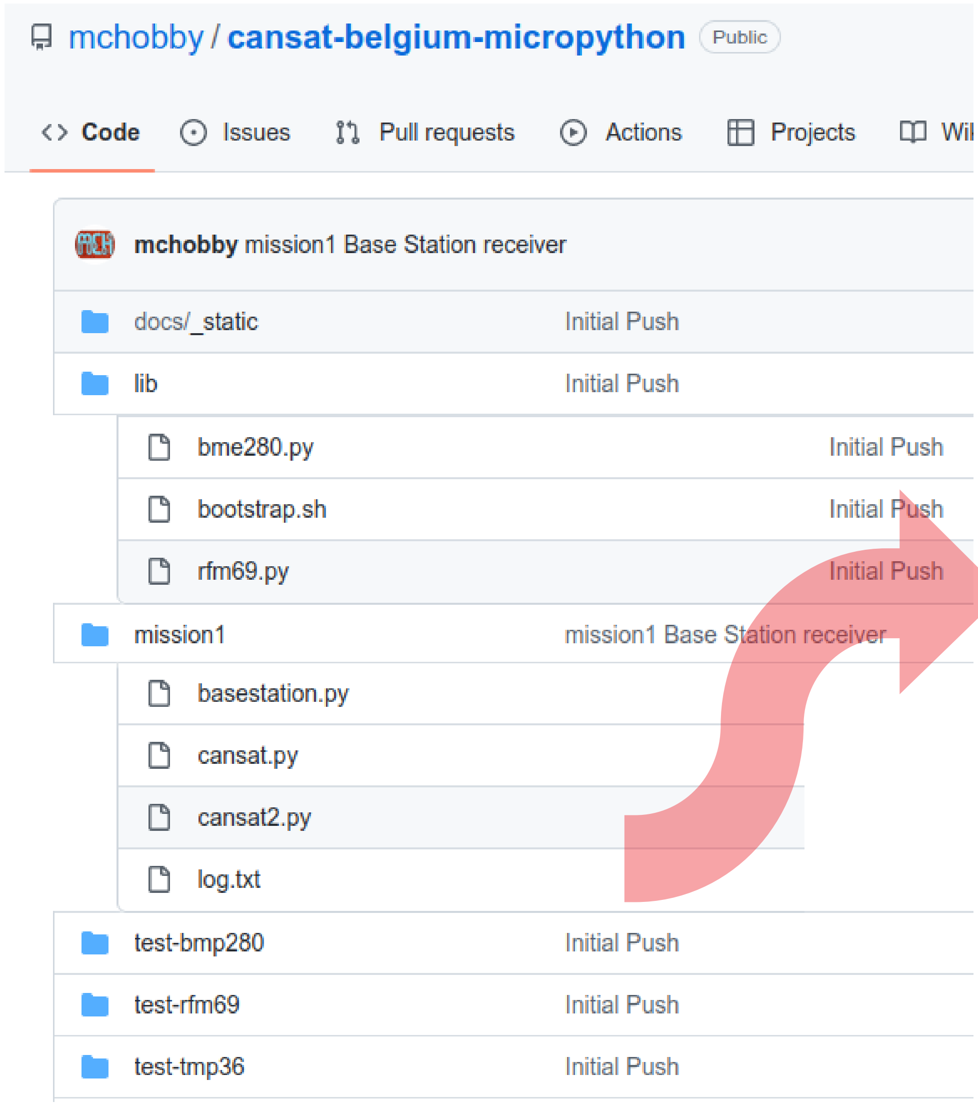
Execution on the fly :
Try this from REPL prompt!
`>>> import blink`



Raspberry-Pi Pico

The source code

https://github.com/mchobby/cansat-belgium-micropython



mchobby / cansat-belgium-micropython Public

Code Issues Pull requests Actions Projects Wi

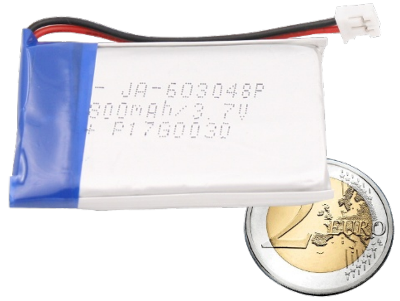
mchobby mission1 Base Station receiver

docs/_static	Initial Push
lib	Initial Push
bme280.py	Initial Push
bootstrap.sh	Initial Push
rfm69.py	Initial Push
mission1	mission1 Base Station receiver
basestation.py	
cansat.py	
cansat2.py	
log.txt	
test-bmp280	Initial Push
test-rfm69	Initial Push
test-tmp36	Initial Push

In cansat.py

```
...
print( 'Frequency     :', rfm.frequency_mhz )
print( 'encryption    :', rfm.encryption_key )
print( 'NODE_ID       :', NODE_ID )
print( 'BASESTATION_ID:', BASESTATION_ID )
print( '***HEADER***' )
print( ":iteration_count,time_sec,pressure_hpa,
               tmp36_temp,bmp280_temp;" )
print( '***DATA***' )
...
```

```
[DATA](len=32,RSSI=-31)bytearray(b':10949,4785,1012.90,22.45,22.19;')
[MSG] :10949,4785,1012.90,22.45,22.19;
[DATA](len=32,RSSI=-31)bytearray(b':10950,4786,1012.83,17.45,22.18;')
[MSG] :10950,4786,1012.83,17.45,22.18;
[DATA](len=32,RSSI=-31)bytearray(b':10951,4786,1012.84,23.25,22.19;')
[MSG] :10951,4786,1012.84,23.25,22.19;
[DATA](len=32,RSSI=-31)bytearray(b':10952,4786,1012.84,23.33,22.19;')
[MSG] :10952,4786,1012.84,23.33,22.19;
[DATA](len=32,RSSI=-25)bytearray(b':10953,4787,1012.83,14.07,22.18;')
[MSG] :10953,4787,1012.83,14.07,22.18;
[DATA](len=32,RSSI=-33)bytearray(b':10954,4787,1012.84,-1.00,22.18;')
[MSG] :10954,4787,1012.84,-1.00,22.18;
```



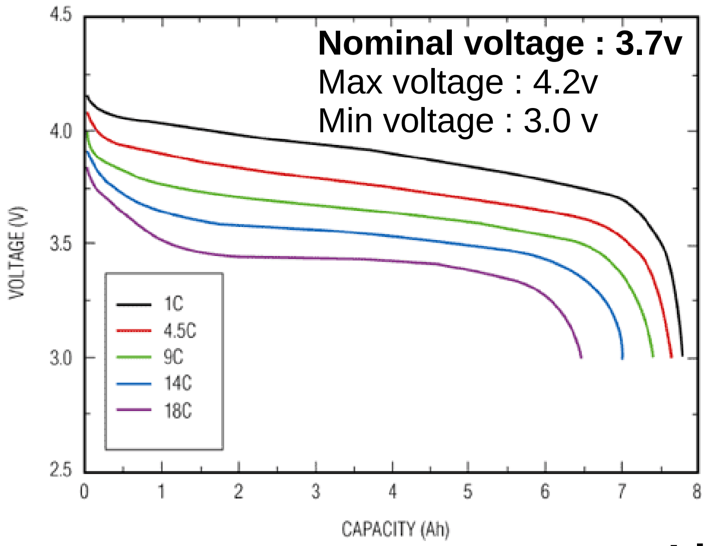
LiPo batteries

LiPo offers the best ratio Power/Weight.

Q (mAh) is the quantity of power.

C is discharge rate.
Charging is usually limited to half of discharge rate.

For battery with Q = 800mAh :
- 1C means that it can be discharged continuously at 800mA.
- 1/2C means that it can be discharged at 400mA.
- 3C means that it can be discharged at 2400mA.



Nominal voltage : 3.7v
Max voltage : 4.2v
Min voltage : 3.0 v

Lipo for electronic
- Usually rated for 1C max.
- Use protection circuitry (over-current or under-voltage).
- Finer wires.
- Lighter.
- Power cycle the Lipo when security get activated.

Lipo for portable radio and electric vehicles.
- Can deliver several C (40C or more).
- No protection circuitry.
- Thick wires.
- More heavy (more cells).
- Must be charged with special device & under surveillance.

LiPo batteries

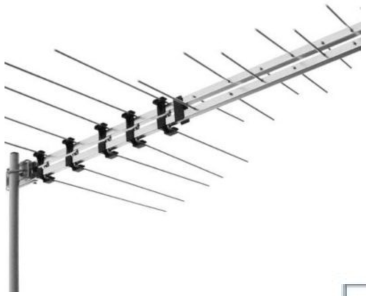
<u>Estimate discharge time</u>

With a 800 mAh Lipo battery :

IF the project sink a current of 150mA from the battery THEN
The lipo will last after 800 mAh / 150 mA = 5.3 Hours

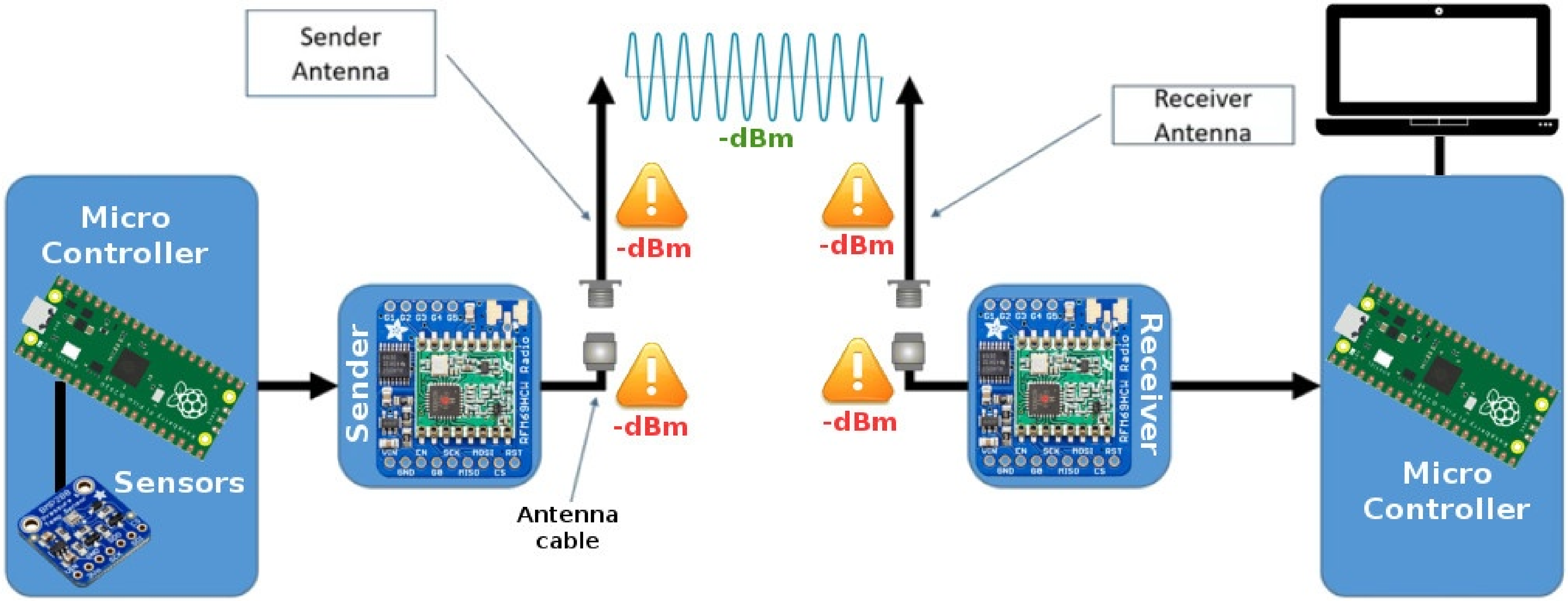
<u>Do not abuse LiPo :</u>

- Do not twist, bend.
- Do not drill.
- Do not fire.
- Do not over-charge/
  over-discharge.
- Do not use when deformed or inflated.
- ALWAYS CHARGE UNDER SURVEILLANCE

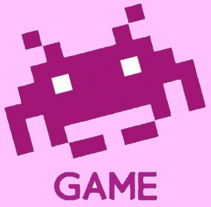


About Radio

The antenna is the key

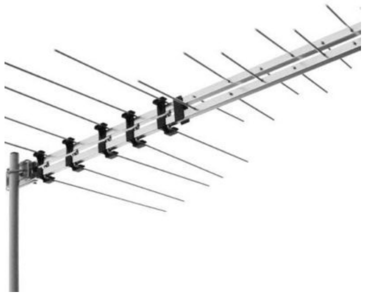


dBm = dB / measured mW = efficient way to measure absolute power.



Radio Transmission is a game where the goal is to lose as little power as possible !

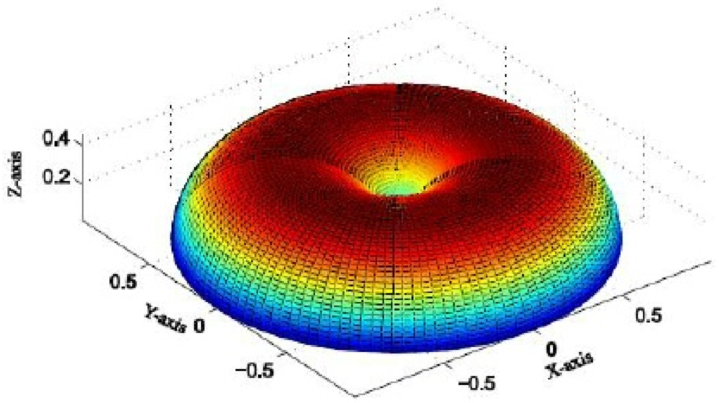
It is even possible to win power.



About Radio

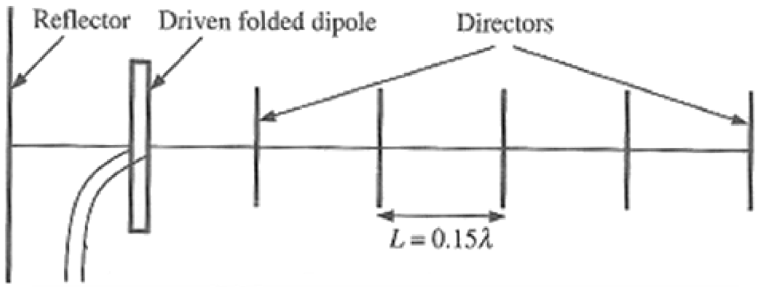
The antenna is the key

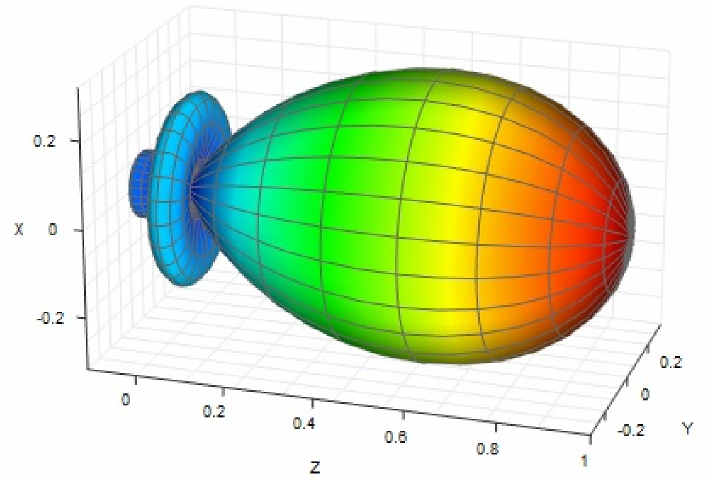
Unipole Antenna



Yagi Antenna

direction of maximum radiation ---->

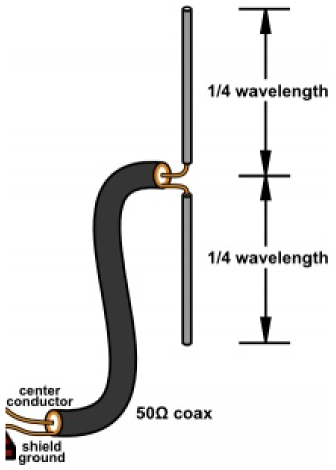


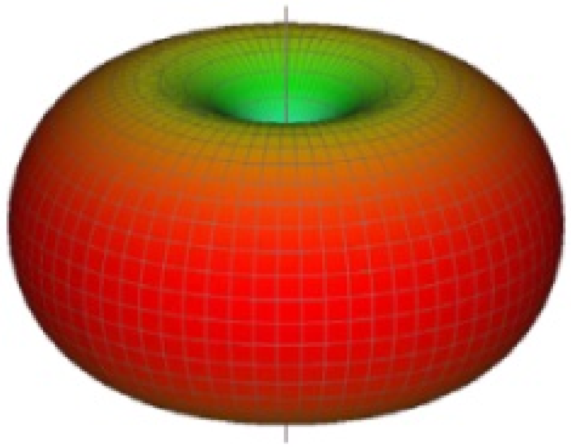


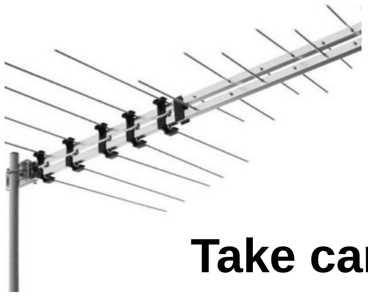
A 6 elements Yagi can offer a gain up to 11.2 dBi.
A 11 elements Yagi can double that gain !!!

See the wiki for a Yagi Antenna with build dimensions for 433Mhz antenna.

Dipole Antenna

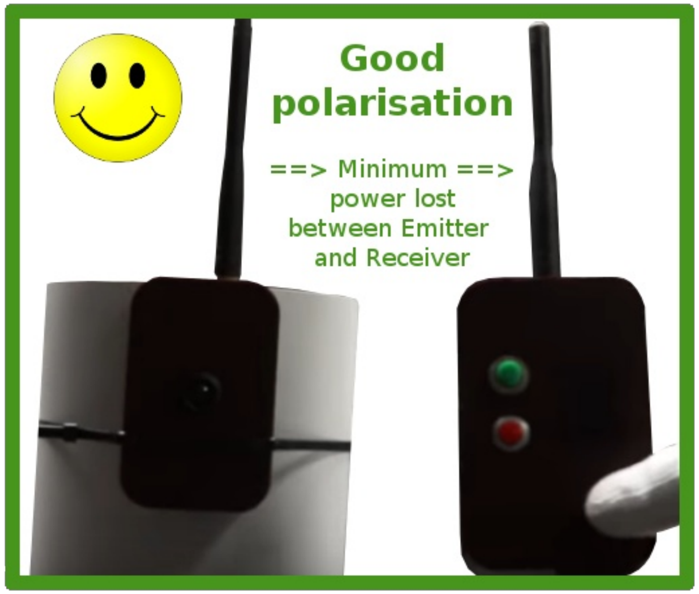


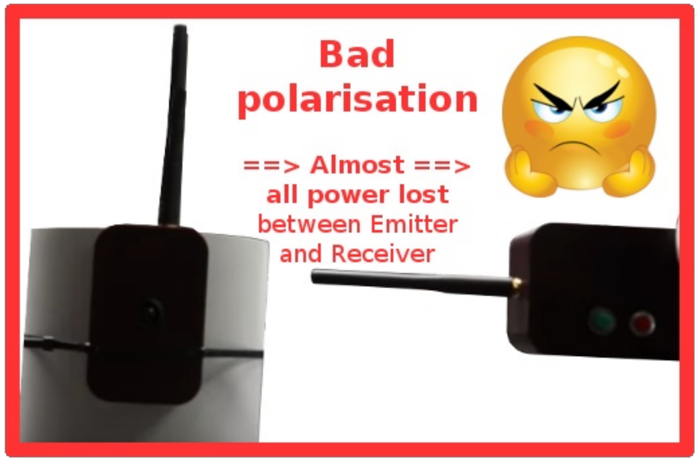




About Radio

Take care about the polarisation





Take care about antenna length

The frequency that the antenna resonates at (operates at) is determined by the length of the antenna.

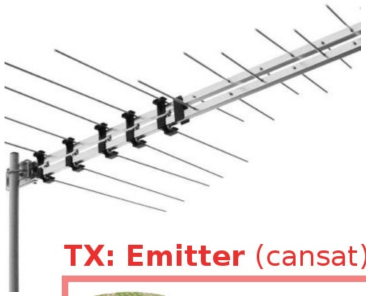
For unipole (and dipole) antenna, the maximum gain of the antenna is fixed and dependent on the operating frequency (the frequency the antenna should resonnate).

Quarter wavelength (1/4 λ) antenna length

```
      c
L = -------
    4 x f
```

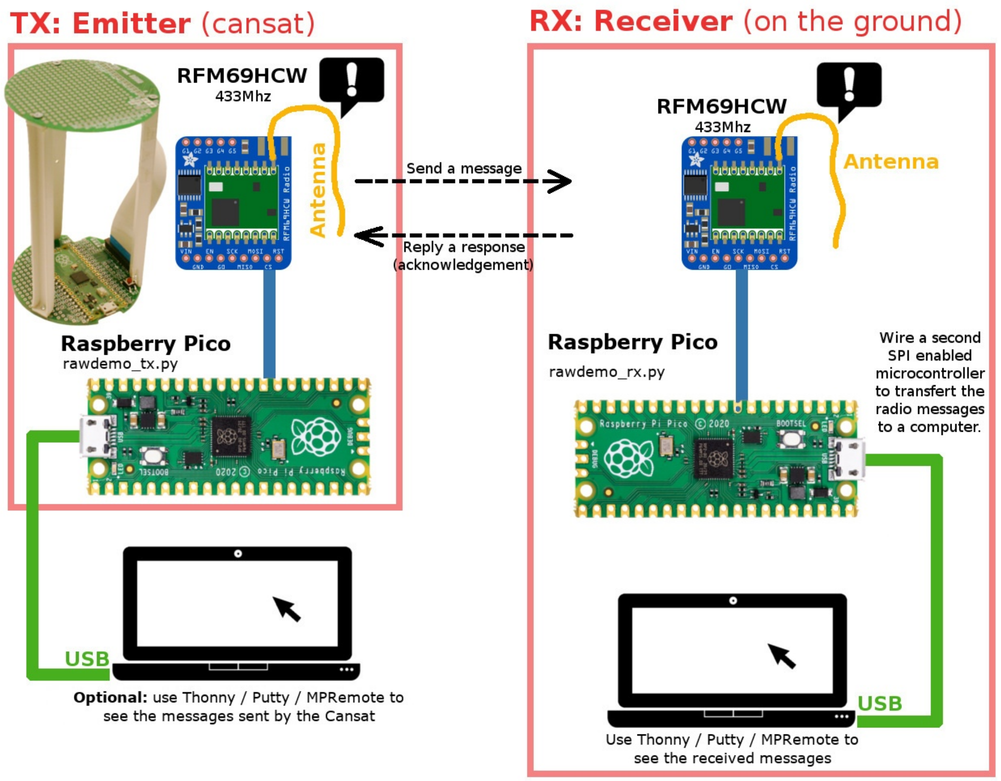
What should be the length of 1/4 λ antenna for the frequency of 433Mhz ?

```
        3x10E8
L = -------------- = 0.1732m
    4 x 433*10E6
```

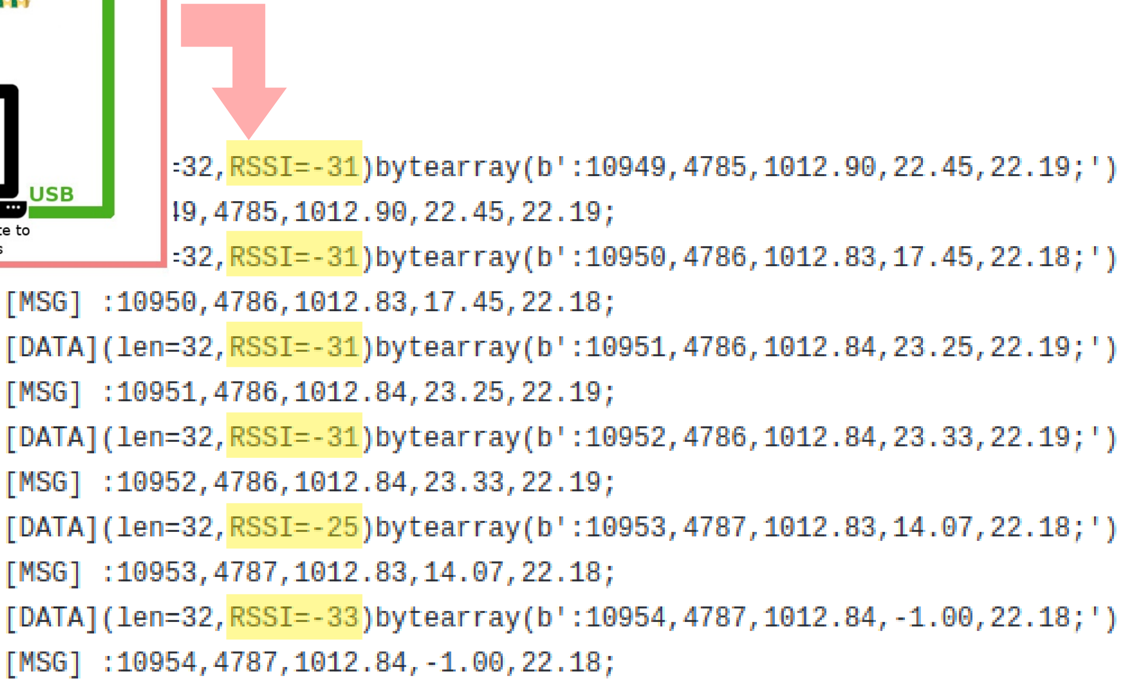


About Radio

RSSI : Evaluate quality of radio setup !


TX: Emitter (cansat)
RFM69HCW
433Mhz
Antenna
Raspberry Pico
rawdemo_tx.py
Send a message
Reply a response
(acknowledgement)
USB
Optional: use Thonny / Putty / MPRemote to
see the messages sent by the Cansat
RX: Receiver (on the ground)
RFM69HCW
433Mhz
Antenna
Raspberry Pico
rawdemo_rx.py
Wire a second SPI enabled microcontroller to transfert the radio messages to a computer.
USB
Use Thonny / Putty / MPRemote to
see the received messages

When testing the « Mission 1 » setup available on the wiki :
The receiver display the telemetric data with an additional information named RSSI.



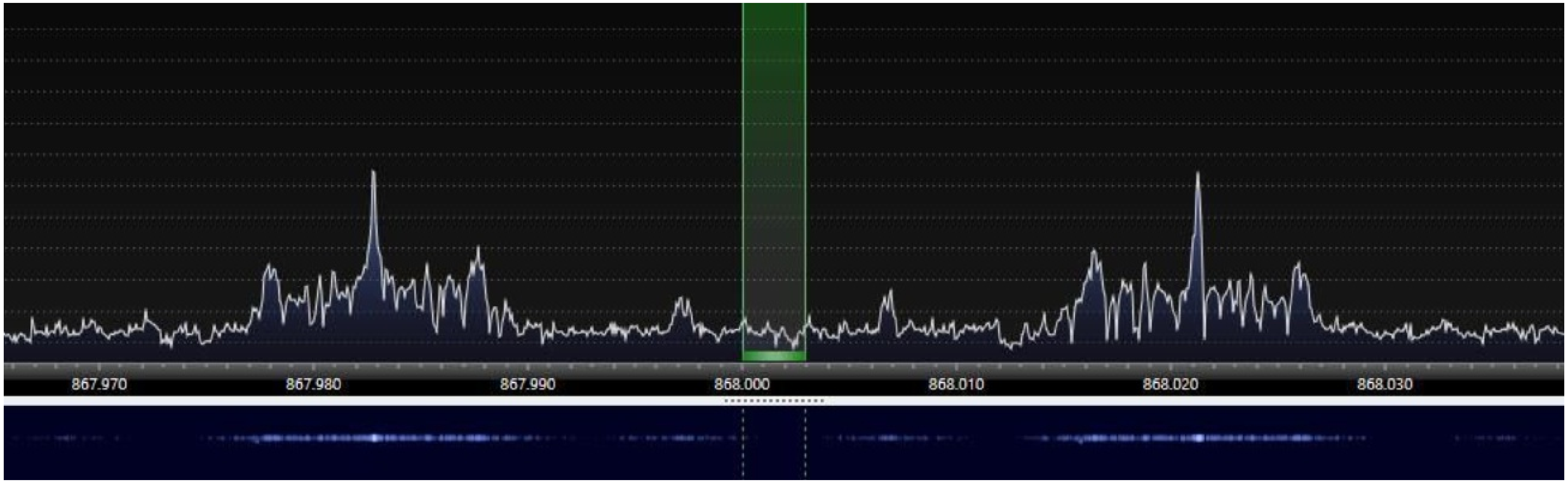
```
:32,RSSI=-31)bytearray(b':10949,4785,1012.90,22.45,22.19;')
!9,4785,1012.90,22.45,22.19;
:32,RSSI=-31)bytearray(b':10950,4786,1012.83,17.45,22.18;')
[MSG] :10950,4786,1012.83,17.45,22.18;
[DATA](len=32,RSSI=-31)bytearray(b':10951,4786,1012.84,23.25,22.19;')
[MSG] :10951,4786,1012.84,23.25,22.19;
[DATA](len=32,RSSI=-31)bytearray(b':10952,4786,1012.84,23.33,22.19;')
[MSG] :10952,4786,1012.84,23.33,22.19;
[DATA](len=32,RSSI=-25)bytearray(b':10953,4787,1012.83,14.07,22.18;')
[MSG] :10953,4787,1012.83,14.07,22.18;
[DATA](len=32,RSSI=-33)bytearray(b':10954,4787,1012.84,-1.00,22.18;')
[MSG] :10954,4787,1012.84,-1.00,22.18;
```

RSSI: Received Signal Strength Indication - indicated the strength of the radio signal received on the transceiver. (-15 at best, -90 at worst).

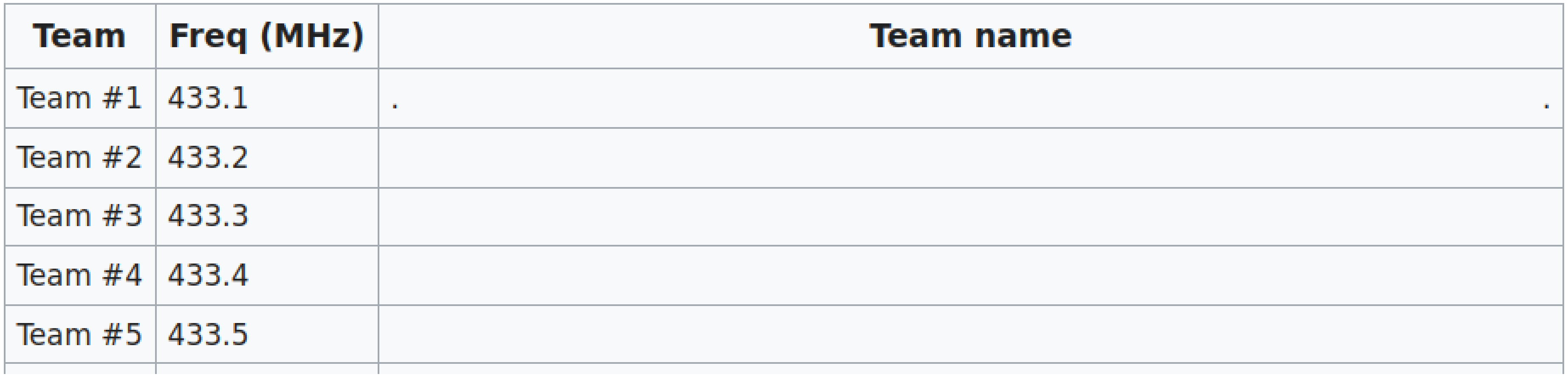
About Radio

Frequency Plan : share the radio bandwidth.

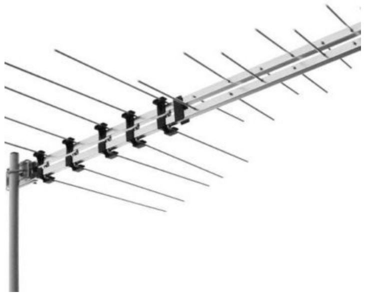
The following SDR capture shows the spectrum view of a RFM69 emiting on the 868.0 MHz frequency. (from USA)



Carrier WaveLength - 30 KHz - Carrier WaveLength + 30 KHz.



Team	Freq (MHz)	Team name
Team #1	433.1	. .
Team #2	433.2	
Team #3	433.3	
Team #4	433.4	
Team #5	433.5	



About Radio

Getting Help with radio stuff !

- **Rule #1:** Use short, high quality and thick antenna cables.
- **Rule #2:** An SWR below 2 is acceptable (less than 11% of power is reflected so we have much of the power available for transmission).
- **Rule #3:** Always connect an antenna to the sender (otherwise 100% of signal is reflected, which may kill the sender)
- **Rule #4:** Keep the polarization of your antennas the same way.
- **Rule #5:** The more dBi, the more power in one direction.
- **Rule #6:** With a proper antenna setup, the distance in air is not an issue if we have a line of sight.
- **Rule #7:** Longer is not always better for antennas. Smarter is better.

Andreas Spiess – Video tutorial

https://youtu.be/J3PBL9oLPX8

Finding Radio Amateur Club

http://map.mchobby.be