

# Contrôle des chambres RPC avec le microcontrôleur Raspberry pico et l'IoT

C.Combaret ,L.Mirabito

Institut de Physique des 2 Infinis de Lyon

June 8, 2023

# Contrôle des chambres RPC à Lyon

---

## Contrôles et mesures

L'opération des chambres RPC nécessite:

- La haute tension ( $< 10$  kV) fournie par des modules HV dans des contrôleurs CAEN (SY1527 ) ou Wiener
- La basse tension (6 V) fournie par des alimentations Lambda Genesys ou Zup
- Des débitmètres massiques Brooks pour les 3 gaz utilisés (TFE, CO<sub>2</sub>/IsoButane, SF<sub>6</sub>)
- Des capteurs environnementaux de pression et température Bosch (BMP283, BME280)
- Des capteurs d'humidité Honeywell (HIH8000)
- Des balances pour les bouteilles de gaz

# Système pré-existant

---

## Lecture

- HV: CAEN → Socket TCP/IP, Wiener → SNMP , depuis un PC
- Zup et Genesys → RS232 sur raspberry Pi
- Debitmètre Brooks → Soft windows non connecté fourni par ServInstrumentation
- BMP283,BME280 → SPI/I2C depuis raspberry PI
- HIH8000 → I2C depuis raspberry PI
- Balance ADAMS non lue

# Systeme pré-existant

---

## Software

- Service linux dédié pour chaque hardware, interfacé par des services REST (START,STOP,STATUS,COMMAND)
- Un service d'interface centrale
  - Interface utilisateur (via des scripts python)
  - Stockage MongoDB, propagation vers GRAFANA
- Un monitoring dans GRAFANA

# Système pré-existant

---

## Contraintes

- Configuration centralisée peu flexible si plusieurs tests sont menés en parallèle
- Pas de watchdog si un des service hardware est bloqué dans un accès.
- Pas de redémarrage lors d'un arrêt sauvage du PC (coupure secteur)
- Tout les contrôles dépendent d'ordinateurs avec OS (mise à jour, sécurité)

## Solution

- Utilisation d'un microcontrôleur → Raspberry pico
- Internet des Objects → MQTT

# Le Raspberry PICO

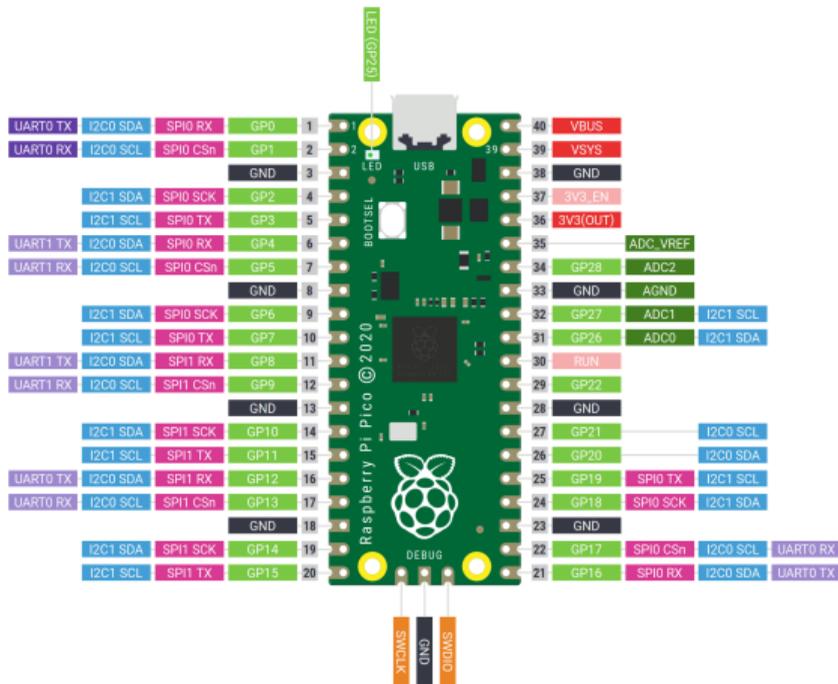
---

Le Raspberry PICO est une carte microcontrôleur développé par la fondation Raspberry:

- RP2040 double coeur à 133 MHz, Ram 264 Ko , 2 Mo de memoire flash
- Bas coût
- Programmable en C et en python
- Large communauté de développeurs (hardware et software)

# Connectique

■	Power
■	Ground
■	UART / UART (default)
■	GPIO, PIO, and PWM
■	ADC
■	SPI
■	I2C
■	System Control
■	Debugging



- 2 SPI, 2 I2C, 2 UART, GPIO, ADC, Timer
- $\mu$  RP2040 avec eeprom embarquée (0.2 Euros)
- USB, interface wifi ou ou TCP/UDP ( wiznet W5500)

# Deux modèles

---

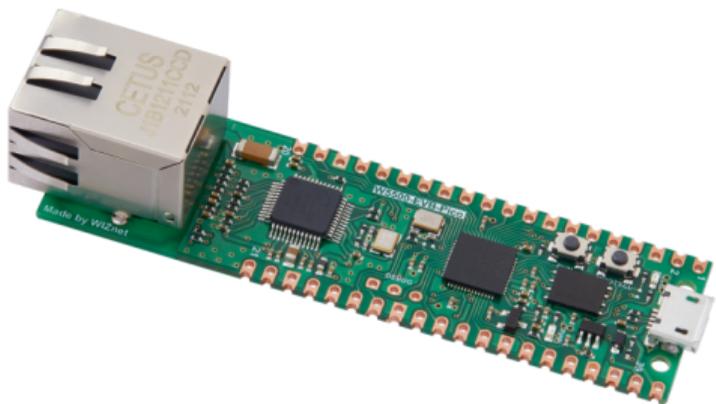


Figure: Ethernet 17 Euros

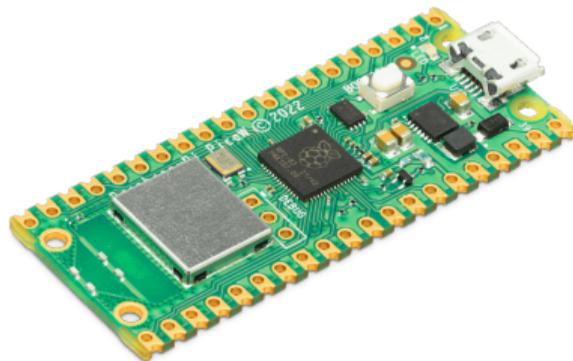


Figure: Wifi 8 Euros

# Board d'adaptation

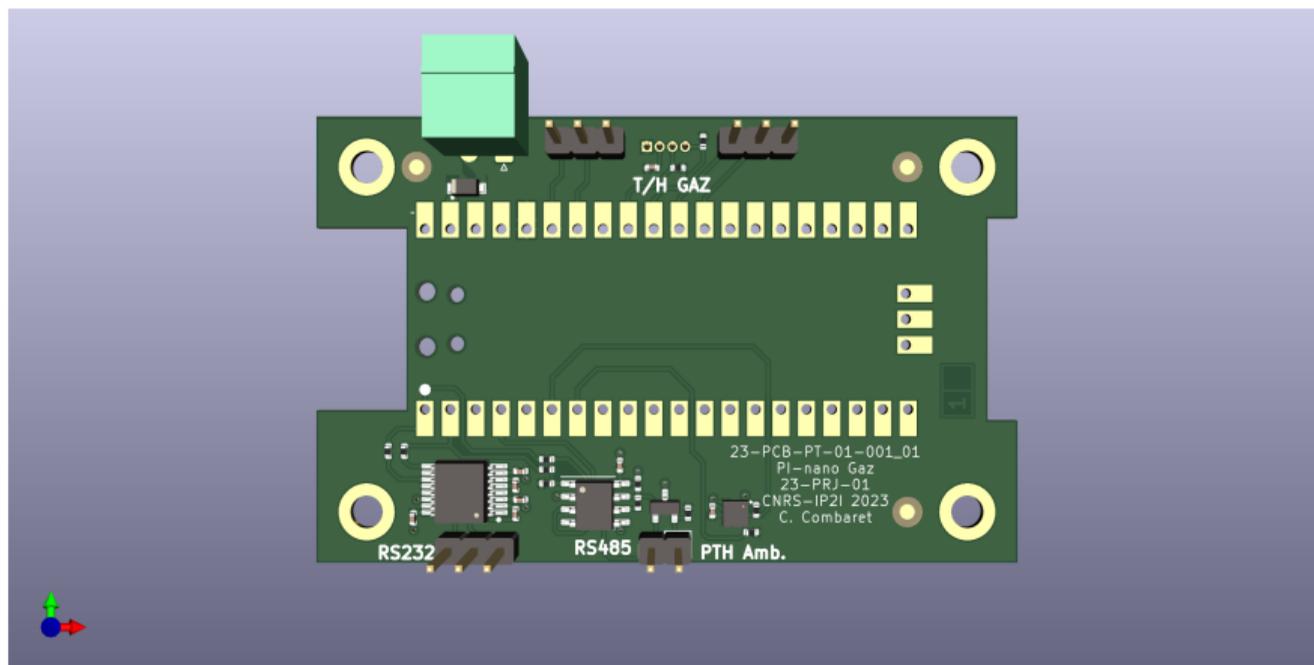


Figure: Board versatile d'interface vers nos capteurs. Un écran LCD peut également être connecté

# Organisation des contrôles

---

hardware	Bus	Mesures	device
Genesys / Zup	RS232	V,I	pico board
Brooks SLA5800	RS485	flow	pico board
BME280	SPI	P,T,H	pico board
HIH8000	I2C	H,T (Gaz)	pico board
ADAMS CW+	RS232	Masse	pico board
Wiener HV	SNMP	V,I	Service linux sur PC
SY1527 HV	TCP socket	V,I	Service linux sur PC

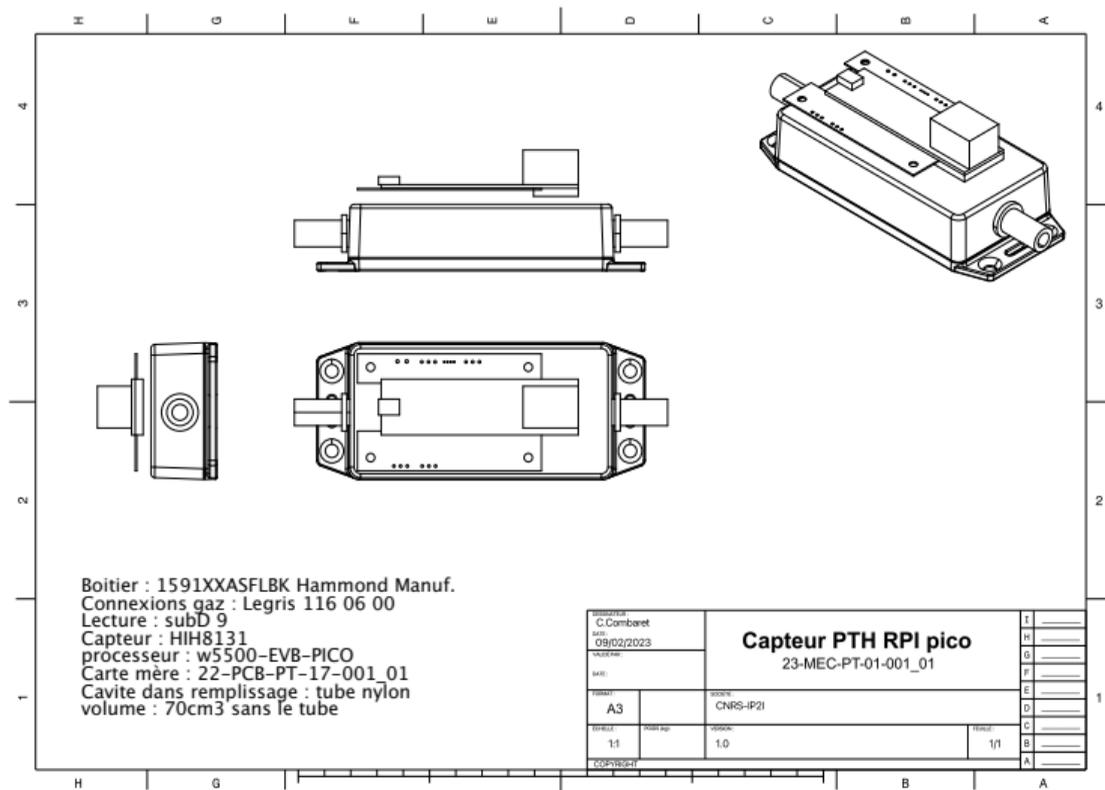
# Exemple

---



Figure: Boitier d'adaptation vers une interface RS232 (LV ou balance). Un capteur P/T/H ambient BME280 est rajouté pour monitorer les variations de pression.

# Mesure du taux d'humidité du gaz (HIH8000)



# Software

---

## MQTT

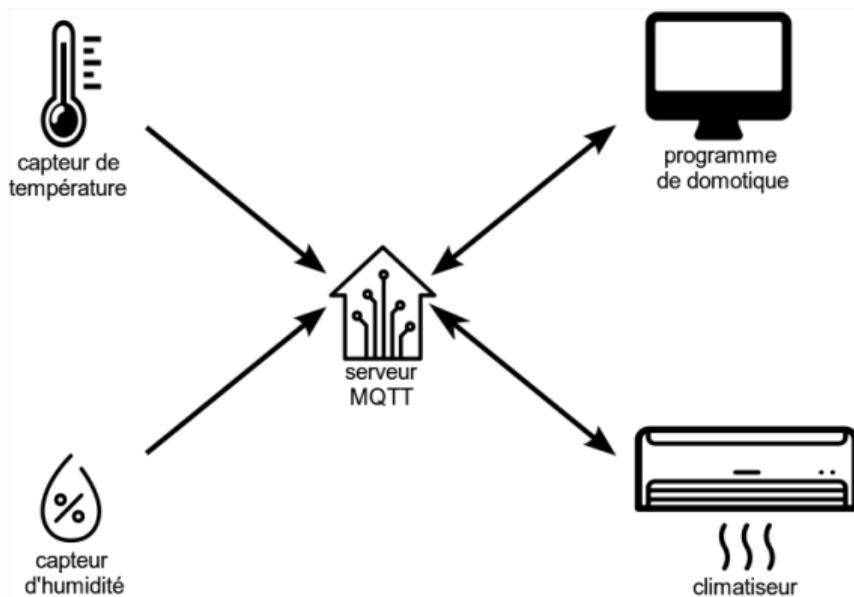
- Protocole développé pour l'IoT
- Aiguillage centralisé (BROKER) pour des messages PUBLISH/SUBSCRIBE
- Bibliothèques disponibles C,C++, python
- Broker local sous linux → mosquitto

## Code dédié

- C++ pour les alimentations HV (TCPIP), python sur tous les microcontrolleurs
- Chaque client publie périodiquement ses mesures et le cas échéant souscrit à des commandes (LVON,HVSET,FLOWSET....)
- On conserve un service linux unique pour le stockage dans une base de données

# MQTT

Protocole pour l'internet des objects:



- Le **broker** centralise les messages et les redistribue
- Les capteurs publient leurs données
- Le climatiseur publie son status et souscrit à des commandes
- Le programme de domotique monitore les capteurs et publie des commandes au climatiseur

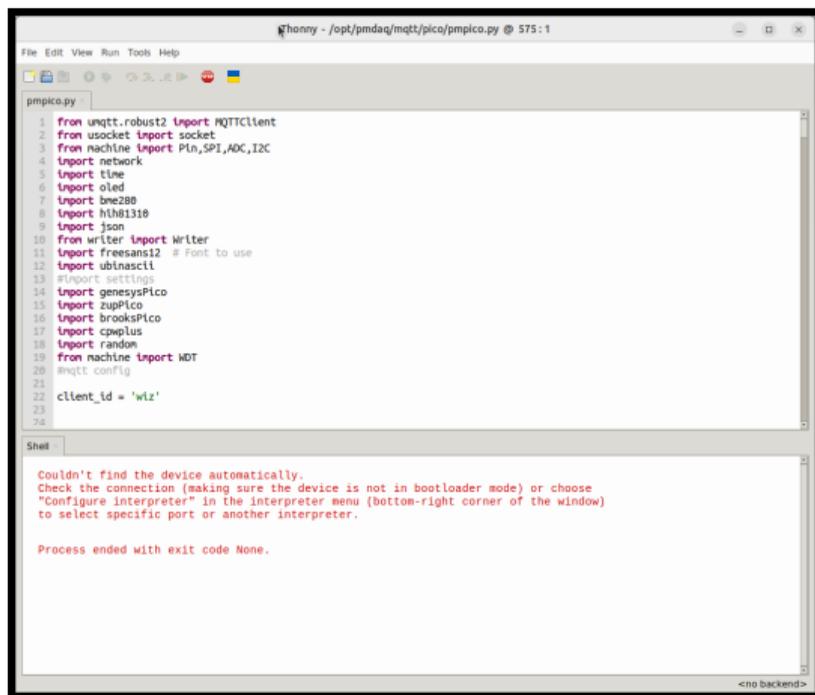
# Environnement de développement

---

## $\mu$ python

- Implémentation python adaptée aux microcontrôleurs
- Bibliothèques pour chaque type de processeurs (Ethernet, MQTT, SPI, I2C, UART, ADC...)
- Déploiement et maintenance aisée (copie du code en Flash)

# Thonny IDE



```
thonny - /opt/pmdaq/mqtt/pico/pmpico.py @ 575:1
File Edit View Run Tools Help
pmpico.py
1 from umqtt.robust import MQTTClient
2 from usocket import socket
3 from machine import Pin,SPI,ADC,I2C
4 import network
5 import time
6 import oled
7 import lme289
8 import hih81318
9 import json
10 from writer import Writer
11 import freesans12 # Font to use
12 import ubtncastil
13 #import settings
14 import genesysPico
15 import zupPico
16 import brooksPico
17 import cplusplus
18 import random
19 from machine import I2C
20 #mqtt config
21
22 client_id = 'wiz'
23
24
Shell
Couldn't find the device automatically.
Check the connection (making sure the device is not in bootloader mode) or choose
"Configure interpreter" in the interpreter menu (bottom-right corner of the window)
to select specific port or another interpreter.

Process ended with exit code None.
<no backend>
```

- IDE Python
- Bibliothèques PICO et MQTT
- Debugueur du microcontrôleur via USB

Figure: Editeur Python interfacé au raspberry

# Interface utilisateur

---

## Commandes

- Commande en ligne : *mosquito\_pub* *mosquitto\_sub*
- Scripts python (bibliothèque Eclipse Paho)
- Page web avec bibliothèque MQTT en javascript

## Monitoring

- Debug: MQTT explorer
- Visualisation: plugin MQTT dans GRAFANA

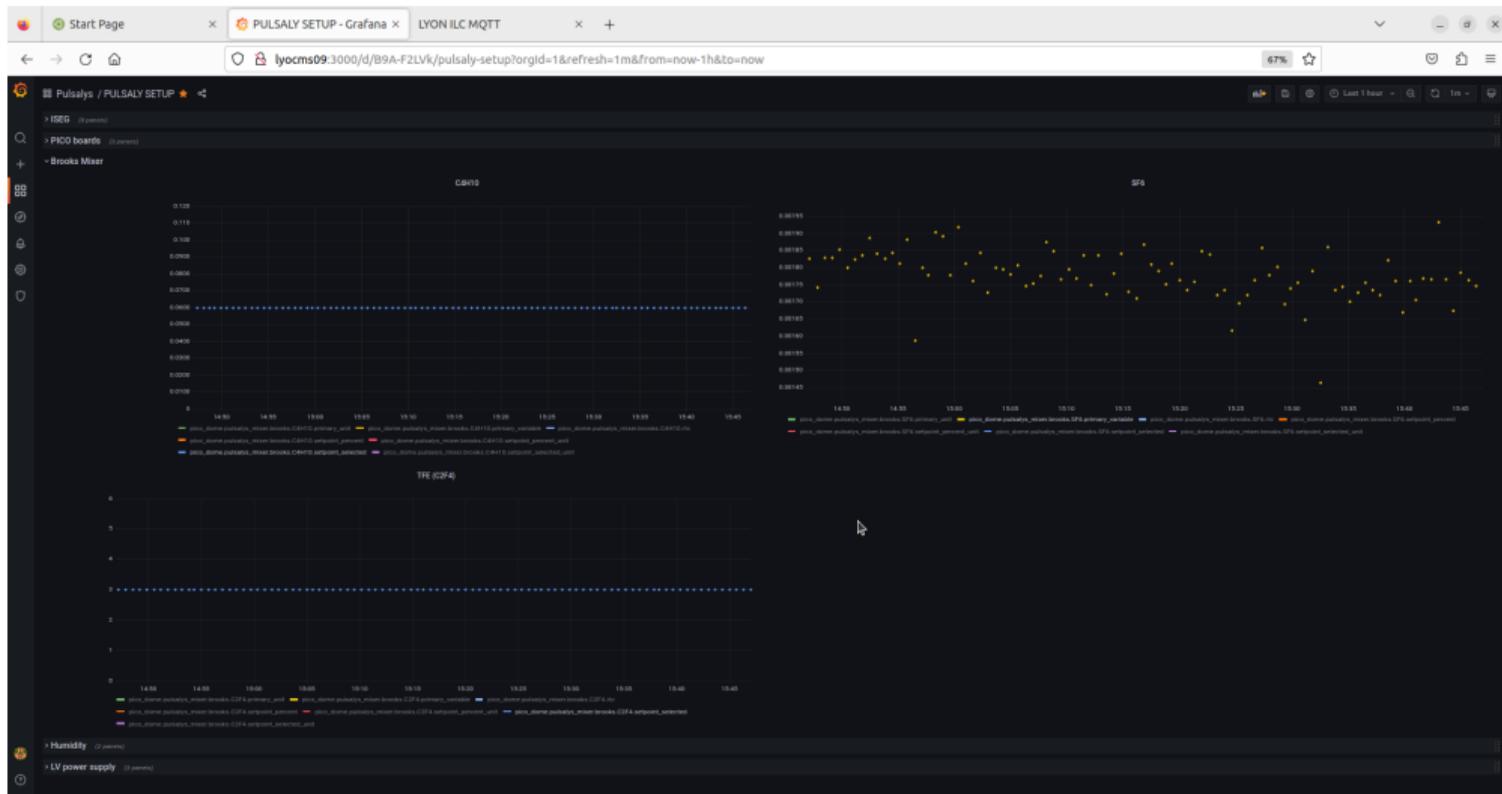
# MQTT explorer

The screenshot displays the MQTT Explorer application interface. At the top, there is a title bar with "MQTT Explorer" and standard window controls. Below the title bar is a navigation bar with a hamburger menu, the text "MQTT Explorer", a search bar, a status indicator, and a "DISCONNECT" button. The main interface is divided into several sections:

- Left Panel (Tree View):** A hierarchical tree view of MQTT topics. The selected topic is `pico_dome/pulsalys_mixer/brooks/SF6`. The tree shows various topics like `tyolico07/test`, `pico_test`, `pico_dome`, `pulsalys_inlet`, `pulsalys_mixer`, `rp2040`, `brooks`, `INFORMS`, `GAS`, `C2F4`, `SF6`, `C4H10`, `SF6`, `C4H10`, `C2F4`, `pulsalys_generys`, `dome_inlet`, `pulsalys_outlet`, `cms_inlet`, `cms_outlet`, and `telescope_inlet`.
- Right Panel (Message List):** A list of messages received on the selected topic. The selected message is: `{ "primary_unit": 138, "primary_variable": 0.002069886, "rtc": 1610416772, "setpoint_selected": 0.06 }`. The message list also shows "QoS: 0", "06/07/2023", and "1:32:26 PM".
- Bottom Panel (Graphs):** Two line graphs showing data over time. The left graph is titled "P pico\_test/Bureau/bme" and the right graph is titled "T pico\_test/Bureau/bme". Both graphs show a fluctuating orange line representing data points.

Figure: Outil de debug MQTT

# Monitoring GRAFANA



# Interface JavaScript

**PICO boards MQTT interface**

Data display is available in [GRAFANA](#)

Broker   Brooks   Low Voltage   P/T/H Status   HV control

**Connection to broker**

Hostname or IP Address and Port Number:

lyoilc07

8080

Location and system:

pico\_dome

Connection: **Connect**   Disconnect

```
Connecting to lyoilc07 on port 8080
Using the client id clientID-22
Subscribing to topic pico_dome/RUNNING
Message to topic pico_dome/LIST is sent
Topic:pico_dome/RUNNING] Message : [{"devices":
["wiener"],"location":"pico_dome","subsystem":"dome_iseg"}]
Topic:pico_dome/dome_iseg/wiener/INFORM] Message : [{"commands":
["CLEARALARM","ISET","OFF","ON","RAMPUP","RESET","STATUS","VSET"]}
Topic:pico_dome/RUNNING] Message : [{"location":"pico_dome","subsystem":
"pulsalys_genesis","devices":["tp2040","genesys"]}
Topic:pico_dome/pulsalys_genesis/genesys/INFORM] Message : [{"id":"genesys","cmds":
["SETON","SETADDRESS","SETOFF","SETVOLTAGE","SETCURRENT",
"SETREMOTE","CLEAR","VIEW","RESET","STATUS"],"tfc":1609459218}
Topic:pico_dome/pulsalys_genesis/tp2040/INFORM] Message : [{"id":"tp2040","tfc":
```

**PICO boards MQTT interface**

Data display is available in [GRAFANA](#)

Broker   **Brooks**   Low Voltage   P/T/H Status   HV control

**Brooks control**

**Brooks system pulsalys\_mixer**

Gas	Id	Max l/h	Flow set (l/h)	Flow read	New value (l/h)		view
C2F4	9057812	10	3.00000	0.09067	<input type="text"/>	Set	VIEW
SF6	9057806	0.3	0.06000	0.00251	<input type="text"/>	Set	VIEW
C4H10	9057816	1	0.06000	0.00062	<input type="text"/>	Set	VIEW

# Interface JavaScript

PICO boards MQTT interface

Data display is available in [GRAFANA](#)

Broker Brooks Low Voltage P/T/H Status HV control

### Zup and Genesys Control

#### Lambda system pulsalsys\_genesys

V Set	I Set	V read	I read	Status	V Req.	I Req.	Set Max Current	ON	OFF	VIEW
5.042	39.800	0.002	0.000	0	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="button" value="VIEW"/>

PICO boards MQTT interface

Data display is available in [GRAFANA](#)

Broker Brooks Low Voltage P/T/H Status HV control

### HV Monitoring

#### Wiener system dome\_lseg

Channel	V Set (V)	I Set (uA)	Ramp Up	V read (V)	I read (uA)	Status	V Req. (V)	I Req. (uA)	Ramp req.	Set Max Current	Set Ramp	ON	OFF	VIEW
0	6800.0	46.000	-47.0	-6800.1	-4.788	BITS: 80 00 80 outputOn(0) outputConstantVoltage(16)	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="button" value="VIEW"/>				
1	6800.0	50.000	-47.0	-6800.2	-14.364	BITS: 80 00 80 outputOn(0) outputConstantVoltage(16)	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="button" value="VIEW"/>				
2	6800.0	50.000	-47.0	-6800.1	-3.220	BITS: 80 00 80 outputOn(0) outputConstantVoltage(16)	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="button" value="VIEW"/>				
3	6800.0	50.000	-47.0	-6800.2	-22.232	BITS: 80 00 80 outputOn(0) outputConstantVoltage(16)	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="button" value="VIEW"/>				
4	6800.0	50.000	-47.0	-6800.1	-1.903	BITS: 80 00 80 outputOn(0) outputConstantVoltage(16)	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="button" value="VIEW"/>				
5	6800.0	50.000	-47.0	-6800.1	-1.164	BITS: 80 00 80 outputOn(0) outputConstantVoltage(16)	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="button" value="VIEW"/>				
6	6800.0	50.000	-47.0	-6800.2	-17.656	BITS: 80 00 80 outputOn(0) outputConstantVoltage(16)	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="button" value="VIEW"/>				

# Conclusion

---

## Hardware

L'utilisation du raspberry pico et le développement du PCB d'interface nous a permis d'avoir un véritable couteau suisse pour contrôler l'ensemble de nos systèmes: gaz, LV et environnement. Le coût de chaque boîtier n'excède pas quelques dizaines d'Euros.

## Software

MQTT est une solution légère pour gérer ces processus de slow control. Enormément d'outils de développement et d'analyse sont disponibles, sur une grande variété de plateformes. L'ensemble de notre code est disponible sous github.