

Table des matières

PARAMETRER SON ESPEasy	2
Firmware ESPEasy	2
Paramétrage des différent capteurs et sondes:	2
I2C OLED SSD1306 Display:	2
I2C OLED SSD1103 Display Framed:	3
DS18B20 Temp sensor Dallas:	3
MH-Z19 Co2 sensor Winsen:	4
BME280 Baro sensor:	5
TSL2561 Lux sensor:	6
Buzzer 3.3v	7
Buzzer Melodie	7
Utilisation des Rules	8
Structure et Opérateurs	8
Allumer ou éteindre une diode LED en fonction d'une valeur donnée	9
Requête de Rules Simple:	9
Requête de Rules Palier:	9
Exemple avec (if & else):	9
Activer UDP NETWORK	10



PARAMETRER SON ESPEasy



Firmware ESPEasy

1. [Liens de téléchargement](#)
2. [Le Wiki](#)
3. [Les Capteur Supportés](#)
4. [Les Rules](#)

Paramétrage des différent capteurs et sondes:

I2C OLED SSD1306 Display:

GND	GND
VCC	VCC
SDA	GPIO 4 (D2)
SCL	GPIO 5 (D1)

Syntaxe pour afficher sur le oled display:

```
[Dallas#Temperature] Dallas = nom du device / Temperature = nom de la valeur
```

Main Config Controllers Hardware **Devices** Notifications Tools

Task Settings

Device: Display - OLED SSD1306 ?

Name: Oled Display

Enabled:

I2C Address: 0x3c - (default)

Rotation: Rotated

Display Size: 128x64

Font Width: Normal

Line 1: IoT BSS-2

Line 2: Temp: [BME280#Temperature] °C

Line 3: Baro: [BME280#Pression] hPa

Line 4: Hum: [BME280#Humidite] %Hr

Line 5: Rsti: [DS18b20.1#Temperature] °C

Line 6: Eau: [DS18b20.2#Temperature] °C

Line 7: Lux: [TSL2561#Lux] Lux

Line 8: Ip: %ip%

Display button: - None -

Display Timeout: 0


Interval: 30 [sec]

[Documentation en ligne](#)

I2C OLED SSD1103 Display Framed:

DS18b20 Temp sensor Dallas:

GND	GND
VCC	VCC
INP	GPIO 2 (D4)


Device: Environment - DS18b20 


Name:

Enabled:


Sensor

1st GPIO: 


Device Address: 

Device Resolution:  Bit

Data Acquisition

Send to Controller 

IDX: 


Send to Controller 

IDX: 

Interval:  [sec]

MH-Z19 Co2 sensor Winsen:

GND	GND
VCC	VCC
TX	GPIO 13 (D7) Vert
RX	GPIO 15 (D8) Bleu

Device: Gases - CO2 MH-Z19 

Name:

Enabled:

Sensor


1st GPIO:

2nd GPIO:

Auto Base Calibration:

Filter:

Data Acquisition

Send to Controller 

IDX:


Interval: [sec]

[Documentation en ligne](#)

BME280 Baro sensor:

GND	GND
VCC	VCC
SDA	GPIO 4 (D2)
SCL	GPIO 5 (D1)

ICS addresss 0x76 Note: SDO Low=0x76, High=0x77

Device: Environment - BMx280 

Name:

Enabled:

I2C Address: [Detected: BME280]


Note: SDO Low=0x76, High=0x77

Altitude: [m]

Temperature offset: [x 0.1C]

Note: Offset in units of 0.1 degree Celcius (also correct humidity)

Data Acquisition

Send to Controller 

IDX:


Interval: [sec]

Penser à renseigner l'altitude pour avoir une pression hPa calibrée.

TSL2561 Lux sensor:

GND	GND
VCC	VCC
SDA	GPIO 4 (D2)
SCL	GPIO 5 (D1)

ICS addresss 0x39

Device: Light/Lux - TSL2561 

Name:

Enabled:


I2C Address:

Integration time:


Send sensor to sleep:

Enable 16x Gain:

Data Acquisition

Send to Controller 


IDX:

Send to Controller 

IDX:

Interval: [sec]

Buzzer 3.3v

 Aucun paramétrage, tout se fait dans l'onglet Rules après avoir câblé le GPIO.

ESP		Buzzer
GPIO	<-->	I/O
Power		
3.3V	<-->	VCC
GND	<-->	GND

Buzzer Melodie

Comment utiliser un buzzer avec de simple rules

Exemple avec ne sonde de température:

```
//LED BLEU
on DS18B20#Temperature<20,4 do
Pulse,12,1,800
rtttl,14:d=2,o=4,b=450:8f,8a
timerSet,1,1
endon
```

Dans cet exemple, quand la température est inférieure à 20.4°C la LED Bleu du GPIO 12 s'allume 800 ms puis le buzzer du GPIO 14 joue une mélodie.

Exemple avec l'heure du système (FONCTION HORLOGE REVEIL)

```
//Looney Toon
On Clock#Time=All,07:05 do
rtttl,14:d=4,o=5,b=140:c6,8f6,8e6,8d6,8c6,a.,8c6,8f6,8e6,8d6,8d#6,e.6,8e6,8e6,8c6,8d6,8c6,8e6,8c6,8d6,8a,8c6,8g,8a#,8a,8f
Pulse,13,1,30000
timerSet,1,10
endon
```

Dans cet exemple, tous les jours à 7H05 le système joue une mélodie et allume une LED sur le GPIO 13 pendant 30 sec.

Utilisation des Rules

Structure et Opérateurs

```
DeviceName#ValueName<<value>
DeviceName#ValueName=<value>
DeviceName#ValueName><value>
DeviceName#ValueName>=<value>
DeviceName#ValueName<=<value>
DeviceName#ValueName!<value>
DeviceName#ValueName<><value>
```

```
equal (=) to
less (<) than
greater (>) than
less or equal (<=) to
greater or equal (>=) to
not equal (!= or <>) to
```

Allumer ou éteindre une diode LED en fonction d'une valeur donnée

Requête de Rules Simple:

```
on MH-Z19#PPM<=700 do
GPIO,14,1,250 //1=ok, 0=no / 250=ms
timerSet,1,30 // toute les 30sec
endon
```

Requête de Rules Palier:

```
//LED BLEU
on MH-Z19#PPM>1 do
Pulse,2,1,200
timerSet,1,1
endon
on MH-Z19#PPM>=749 do
GPIO,2,0
timerSet,1,10
endon
```

```
//LED VERT
on MH-Z19#PPM>750 do
Pulse,16,1,250
timerSet,1,1
endon
on MH-Z19#PPM>=1000 do
GPIO,16,0
timerSet,1,10
endon
```

Exemple avec (if & else):

```
on sw1#state do
if [dummy#var1]=0
TaskValueSet 12,1,1
else
TaskValueSet 12,1,0
endif
gpio,16,[dummy#var1]
gpio,13,[dummy#var1]
endon
```

Activer UDP NETWORK

L'activation du réseau UDP entre vos modules peu s'avérer utile pour une gestion centralisé de ceux-ci.

Une fois ce réseaux activé, vous avez accès a tout vos modules depuis n'importe lequel.

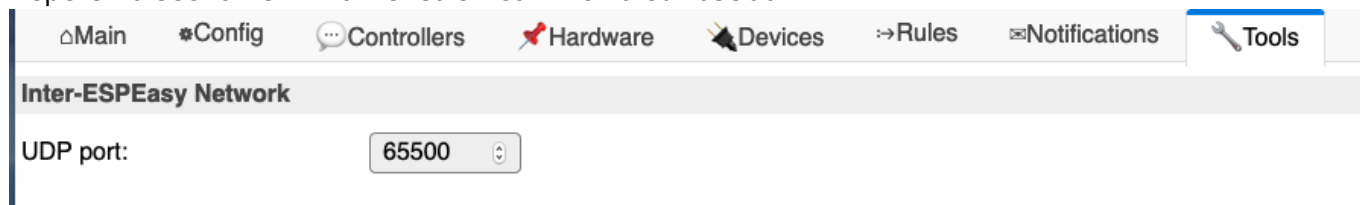
Node List	Name	Build	Type	IP
Unit 0	Iron.Guard	20107	ESP Easy Mega	
Unit 1	Home.Guard	20107	ESP Easy Mega	
Unit 2	Pool.Guard	20103	ESP Easy Mega	

Pour activer cet option:

Depuis l'interface WEB de votre ESP, aller dans **Tools** puis **Advanced**



Repérer la section UDP Port et saisir comme valeur 65500.



IMPORTANT: Il est impératif de le faire sur tout vos modules afin qu'ils puisse dialoguer entre eux.

— [sylvain](#) 2020/06/21 17:37

From:
<https://wiki.mazinger.fr/wiki/> - **My Personal Wiki**

Permanent link:
https://wiki.mazinger.fr/wiki/doku.php?id=arduino:esp8266:esp_easy

Last update: **2024/03/03 12:56**