

UNIVERSITÉ SORBONNE PARIS NORD  
ÉCOLE D'INGÉNIEURS SUP GALILÉE  
PARCOURS INSTRUMENTATION  
2ÈME ANNÉE  
2020-2021

PROJET LINUX POUR L'EMBARQUÉ

Réalisé par :

Mr. PAEZ Edward

Encadré par :

Mr. Walid ABDAOUI

**ANNÉE UNIVERSITAIRE 2020-2021**

## Introduction

Un système embarqué est défini comme un système électronique et informatique autonome, souvent temps réel, spécialisé dans une tâche précise. Le terme désigne aussi bien le matériel informatique que le logiciel utilisé. Ses ressources sont généralement limitées spatialement (encombrement réduit) et énergétiquement (consommation restreinte).

L'un des premiers systèmes modernes embarqués reconnaissables a été le Apollo Guidance Computer en 1967, le système de guidage de la mission lunaire Apollo, développé par Charles Stark Draper du Massachusetts Institute of Technology. Chaque mission lunaire était équipée de deux systèmes (AGC), un chargé du guidage inertiel et un pour le module lunaire. Au commencement du projet, l'ordinateur AGC d'Apollo était considéré comme l'élément le moins fiable du projet. En revanche, grâce à l'utilisation de nouveaux composants qu'étaient à l'époque les circuits intégrés, des gains substantiels sur la place utile et la charge utile ont été réalisés, avec une diminution supposée des risques déjà nombreux des missions.

Les ordinateurs embarqués fonctionnant sous le système d'exploitation Linux sont massivement présents dans les technologies modernes (transports, multimédia, téléphonie mobile, appareils photos, etc.). Contrairement aux versions de Linux destinées aux ordinateurs personnels et aux serveurs, les différents systèmes Linux embarqués sont conçus pour des systèmes aux ressources limitées. Les systèmes embarqués sous Linux disposent généralement de peu de RAM et utilisent fréquemment de la mémoire flash plutôt qu'un disque dur. Comme ils sont souvent dédiés à un nombre de tâches réduites sur une cible matérielle bien définie, ils utilisent plutôt des versions du noyau Linux optimisées pour des contextes précis.

L'ordinateur Raspberry PI constitue un support d'apprentissage performant, très bon marché et disposant d'une forte communauté sur le net. Il possède des entrées/sorties puissantes permettant une connexion avec le monde physique par l'intermédiaire de capteurs et d'actionneurs.

## Objectifs pédagogiques

- Objectifs 1 : niveau 1 : Réaliser un système complet (capteur, contrôleur et actuateur) qui sera déployé sur un ordinateur Raspberry PI.
- Objectifs 2 : niveau 2 : Réaliser un script Bash pour récupérer toutes les données.
- Objectifs 3 : niveau 3 : Envoyer les données vers le Broker MQTT.
- Objectifs 4 : Niveau 4 : Réaliser l'interface graphique avec Node Red pour montrer les données.

## Description de l'ordinateur embarqué

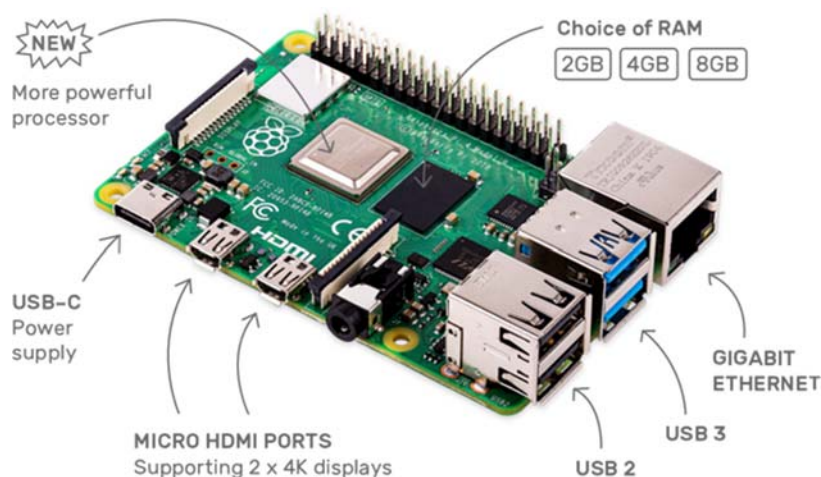


Figure 1: Carte Raspberry Pi 4 Model B

Au cœur du Raspberry Pi 4 Model B, on retrouve donc un processeur Broadcom BCM2711 dont la dénomination pourrait laisser croire à un recul par rapport au BCM2837 qui équipe la génération précédente. En réalité, le BCM2711 se distingue par la présence de quatre cœurs ARM Cortex-A72, bien plus puissants que les Cortex-A53 du Pi 3B+. La fréquence de fonctionnement est également en progrès (+ 100 MHz) à 1,5 GHz.

Raspberry Pi peut être directement connecté à une IHM classique, souris/clavier/écran HDMI ou vidéo composite, cependant comme tout ordinateur Linux, Raspberry Pi peut intégrer ses propres outils de développement et une IHM reposant sur SSH contrôlable depuis un autre ordinateur par Ethernet ou WIFI.

Le connecteur d'extension supporte les entrées/sorties parallèles ainsi que la plupart des bus de communication. C'est un support particulièrement économique et puissant qui peut être facilement mis en œuvre dans de petits systèmes nécessitant un accès au monde physique par des capteurs/actionneurs disposants d'interfaces numériques.

### Fiche technique du Raspberry Pi 4 Model B

- Processeur : Broadcom BCM2711, quad-core Cortex-A72 64-bit à 1,5 GHz

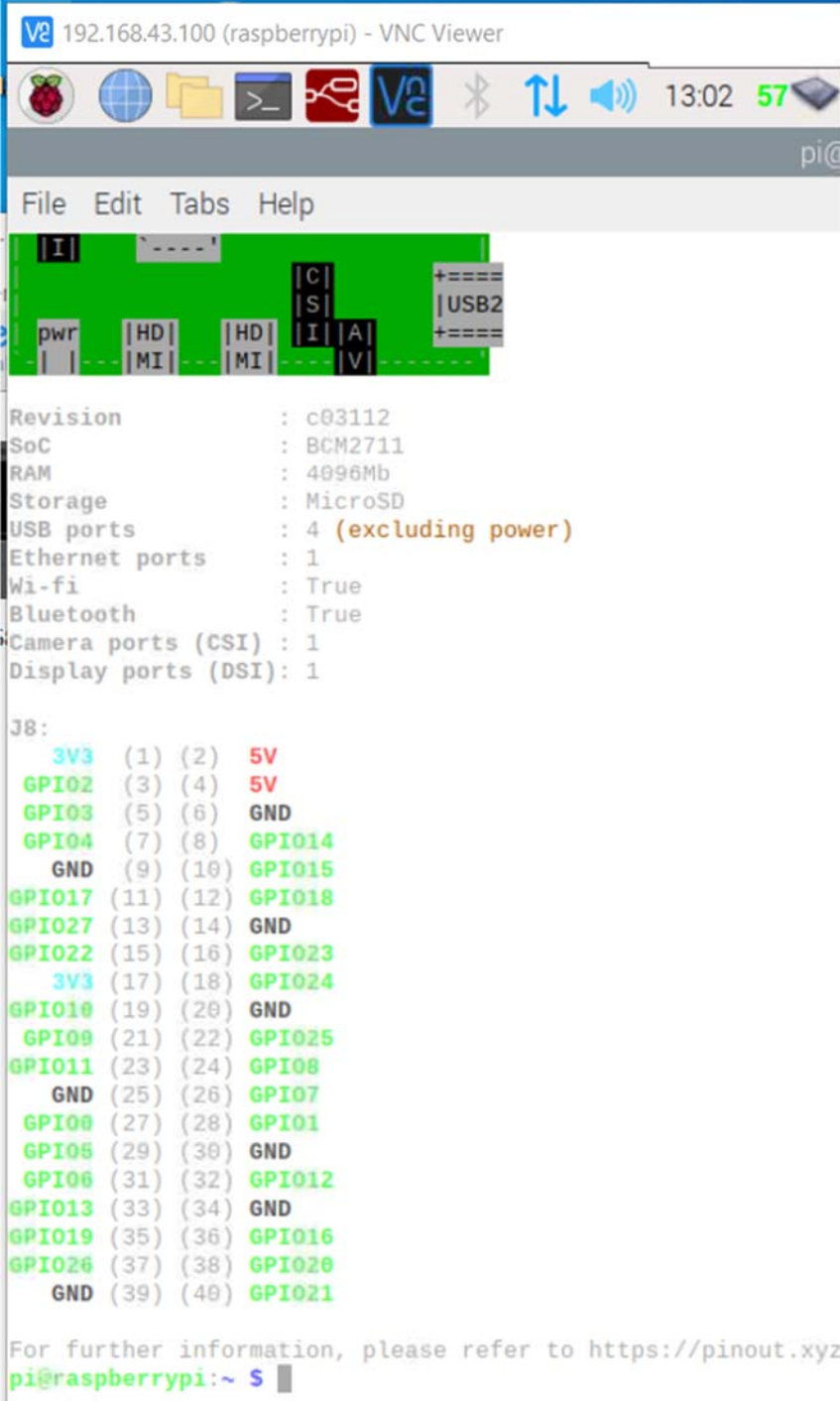
- GPU : Broadcom VideoCore VI à 500 MHz
- Mémoire vive : 1 Go, 2 Go ou 4 Go de LPDDR4-3200 SDRAM (selon le modèle)
- Réseaux : Gigabit Ethernet et Wi-Fi 802.11b/g/n/ac 2,4 / 5 GHz ; Bluetooth 5.0, Bluetooth Low Energy
- Stockage : Lecteur de cartes microSD
- Connectique : Ports USB 2.0 (x2), USB 3.0 (x2), Ethernet (RJ45), micro-HDMI (x2), jack audio 3,5 mm, Camera Serial Interface (CSI), Display Serial Interface (DSI) et USB-C (alimentation), General Purpose Input/Output (GPIO) 40 broches
- Dimensions : Format « carte de crédit » : 88 x 58 x 19 mm, 46 grammes

## Les connecteurs d'extension

Le connecteur d'extension de la carte Raspberry PI est utilisé pour raccorder des périphériques de communication (UART, I2C, SPI, ...) ou TOR (Tout Ou Rien). Les broches peuvent avoir des fonctions différentes suivant qu'elles sont activées en tant que GPIO (Global Purpose Input Output), périphérique de communication ou sorties PWM (Pulse Width Modulation).



Figure 2: Pinout Raspberry Pi 4



The screenshot shows a terminal window titled "V2 192.168.43.100 (raspberrypi) - VNC Viewer". The terminal output displays the following hardware information:

```

Revision      : c03112
SoC           : BCM2711
RAM           : 4096Mb
Storage       : MicroSD
USB ports     : 4 (excluding power)
Ethernet ports : 1
Wi-fi        : True
Bluetooth    : True
Camera ports (CSI) : 1
Display ports (DSI): 1

```

Below the hardware information, the pinout for the J8 header is listed:

Pin	Signal	Pin	Signal
1	3V3	2	5V
3	GPI02	4	5V
5	GPI03	6	GND
7	GPI04	8	GPI014
9	GND	10	GPI015
11	GPI017	12	GPI018
13	GPI027	14	GND
15	GPI022	16	GPI023
17	3V3	18	GPI024
19	GPI010	20	GND
21	GPI009	22	GPI025
23	GPI011	24	GPI008
25	GND	26	GPI007
27	GPI006	28	GPI001
29	GPI005	30	GND
31	GPI006	32	GPI012
33	GPI013	34	GND
35	GPI019	36	GPI016
37	GPI026	38	GPI020
39	GND	40	GPI021

For further information, please refer to <https://pinout.xyz>

The terminal prompt is `pi@raspberrypi:~$`.

Figure 3: Pinout Raspberry Pi 4

## Raspbian

Raspbian est un système d'exploitation libre basé sur la distribution GNU/Linux Debian, et optimisé pour le plus petit ordinateur du monde, la Raspberry Pi.

Raspbian ne fournit pas simplement un système d'exploitation basique, il est aussi livré avec plus de 35 000 paquets, c'est-à-dire des logiciels précompiles livrés dans un format optimisé, pour une installation facile sur votre Raspberry Pi via les gestionnaires de paquets.

La Raspberry Pi est une framboise merveilleuse, mais elle reste néanmoins dotée d'une puissance inférieure à celle d'un ordinateur moderne. Par conséquent, il est préférable d'installer un système optimisé pour la Raspberry.

Raspbian a été créé dans cette optique, et il est donc tout particulièrement adapté à la Raspberry.

Par ailleurs, en tant que distribution dérivée de Debian, il répond à la majeure partie de la très vaste documentation de Debian.

<https://www.raspbian.org/>

## Qu'est-ce que le protocole MQTT

Le protocole MQTT (Message Queuing Telemetry Transport) est un protocole de messagerie de type publication/souscription basé sur le protocole TCP/IP et développé en 1999 par Andy Stanford-Clark de IBM et Arlen Nipper d'EuroTech. Ce protocole spécialement dédié au monde du M2M (machine to machine) et aux objets connectés est maintenant devenu un standard.

Le MQTT permet à deux équipements distants de communiquer via des messages de manière asynchrone avec une faible bande passante. Il est de plus en plus utilisé pour faire communiquer des objets connectés : les objets connectés collectent les différentes informations issues de capteurs intégrés et ces données sont envoyées via MQTT.

Le MQTT fonctionne sur les périphériques embarqués comme l'Arduino ou le Raspberry Pi mais aussi avec des automates programmables industriels. Pour mettre en place une communication via MQTT, on aura d'une part le Broker ou serveur MQTT et d'autre part le client MQTT. Il existe des projets open source comme Mosquitto qui permet de mettre en œuvre le protocole MQTT. Des bibliothèques MQTT sont aussi disponibles pour la conception de clients MQTT dans les langages comme Arduino, C, C++, Java, C#, Python etc.

## À quoi sert le protocole MQTT ?

Pour ce qui est de notre domaine qu'est l'automatisme industriel, le protocole MQTT peut servir par exemple à connecter un automate programmable doté d'un serveur OPC UA au Cloud (Azure, IBM Bluemix ou AWS). Le modèle pub/sub de l'OPC UA pourra utiliser le MQTT comme moyen de transport afin d'envoyer des données industrielles provenant des capteurs vers le Cloud. On pourra par exemple exploiter ces données via des tableaux de bords interactifs qui pourront être consultés sur tablette tactile ou sur un simple smartphone.

Comparé au système de "polling", le système pub/sub utilisé par le MQTT permet de gagner en bande passante vu que les équipements ne sont plus scrutés périodiquement. En effet, si l'on prend l'exemple d'un système SCADA qui grâce à un driver de communication scrute périodiquement des



capteurs pour afficher leurs valeurs, en utilisant le protocole MQTT, avec le système pub/sub, aucune scrutation n'est effectuée, les capteurs vont de leurs propres initiatives publier leurs valeurs sur un "topic" lorsqu'il y'a un changement. Ainsi, un broker englobera tous les "topics" et les équipements (clients) qui seront intéressés par un topic en particulier n'auront qu'à s'abonner à celui-ci. À chaque fois qu'il y'a un changement sur un topic en particulier, les abonnés à ce topic seront notifiés par le broker qui se chargera d'acheminer les nouvelles valeurs vers les abonnés.

Aujourd'hui, il est possible de contrôler à distance un automate programmable grâce au protocole MQTT. Pour cela, vous aurez besoin d'un broker. Sur TIA Portal par exemple, il existe un bloc de fonction MQTT qui permet d'assurer les communications entre votre automate et votre broker MQTT. Ce bloc de fonction nommé "LMQTT\_Client" est un bloc FB qui permet à un automate S7-1200 ou S7-1500 d'établir une connexion au Cloud par l'intermédiaire d'un broker.

## Node-RED

Node-RED est un outil de programmation permettant de câbler des dispositifs matériels, des API et des services en ligne de manière nouvelle et intéressante. Il fournit un éditeur basé sur un navigateur qui facilite le câblage des flux en utilisant le large éventail de nœuds de la palette qui peut être déployé sur son temps d'exécution en un seul clic.



Dans de nombreux projets, l'IHM est constituée d'une application web accessible depuis un navigateur. Il n'est pas toujours aisé de programmer le lien entre le matériel (capteurs et/ou actionneurs) et la page web fournie à l'utilisateur. Une solution consiste à utiliser un script Python dont l'exécution peut être planifiée avec Cron, pour interagir avec le matériel, lire les données des capteurs et les stocker dans une base de données comme MySQL. Un serveur web comme Apache2, via une page web php, met à disposition des utilisateurs les informations. Cette solution décrite dans un précédent article nécessite la mobilisation de nombreuses technologies et plusieurs langages de programmation, contraignant les développeurs du projet à retarder la mise œuvre d'un prototype pour se former.

Combiné avec une solution matérielle constituée d'une Raspberry et éventuellement une carte Arduino, Node Red se révèle être une alternative très intéressante :

Node-RED est un outil puissant pour construire des applications de l'Internet des Objets (IoT) en mettant l'accent sur la simplification de la programmation qui se fait grâce à des blocs de code prédéfinis, appelés « nodes » pour effectuer des tâches. Il utilise une approche de programmation visuelle qui permet aux développeurs de connecter les blocs de code ensemble. Les nœuds connectés, généralement une combinaison de nœuds d'entrée, de nœuds de traitement et de nœuds de sortie, lorsqu'ils sont câblés ensemble, constituent un « flow ».

## Conceptualisation

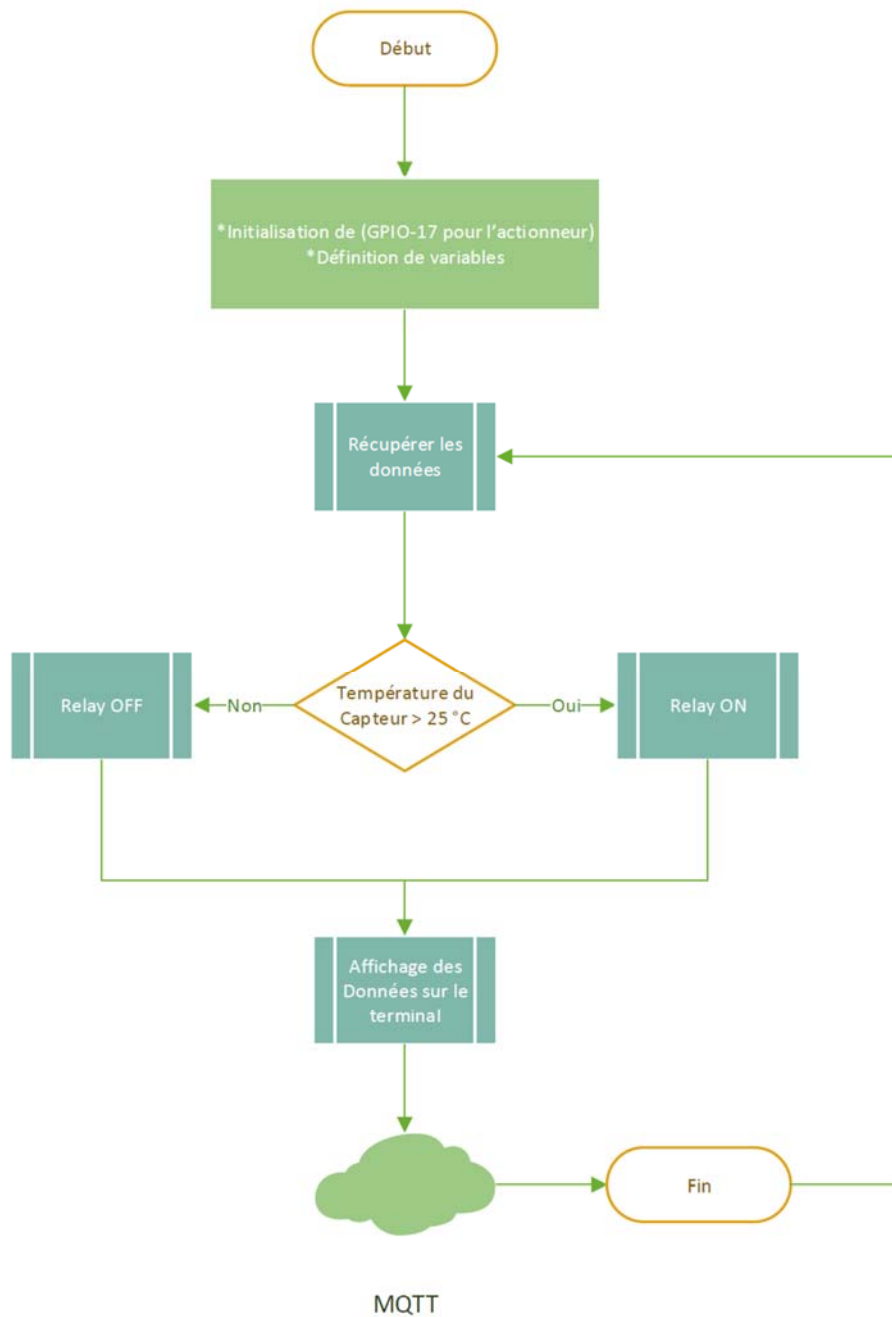
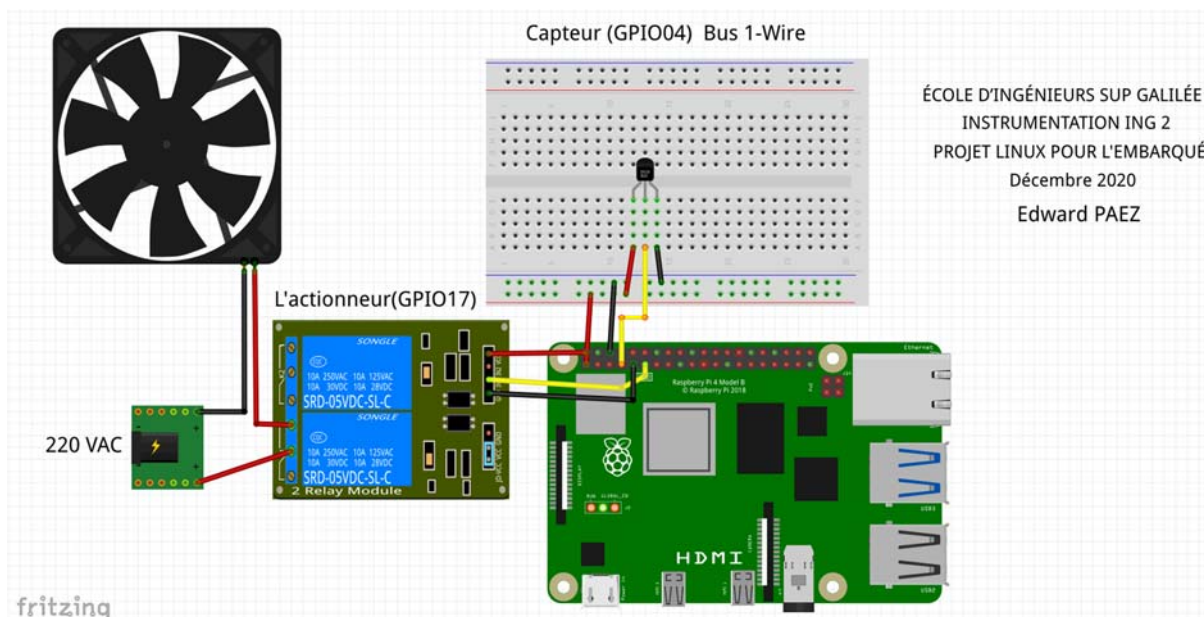


Figure 4: Diagramme de flux du programme en Bash



On a relié pour les capteurs et l'actionneur :

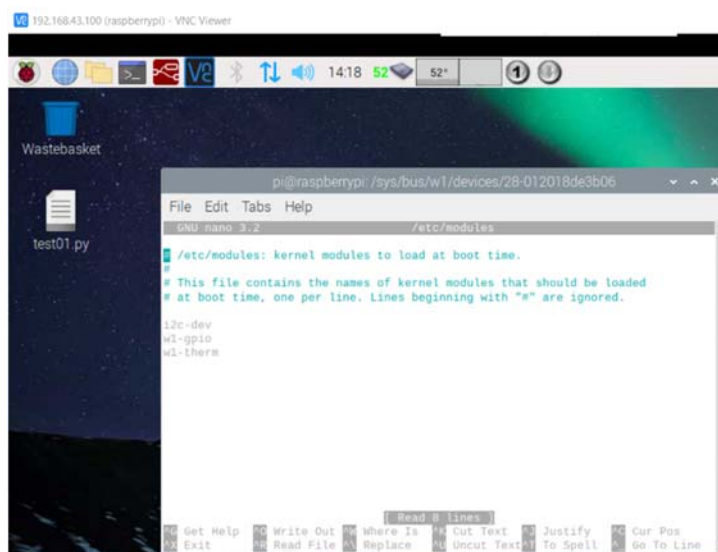
- Fil rouge de la sonde sur le pin #1 (3.3V)
- Fil noir de la sonde sur le pin #6 (Ground / masse)
- Fil jaune de la sonde sur le pin #7 (appelé GPIO4)
- Fil jaune du Relay sur le pin #11 (appelé GPIO17)
- Fil rouge du Relay sur le pin #2 (5V)

Pour la configuration de la Raspberry (1-Wire). Il faut ajouter ces deux lignes à la fin du fichier /etc/modules :

1. w1-gpio
2. w1-therm

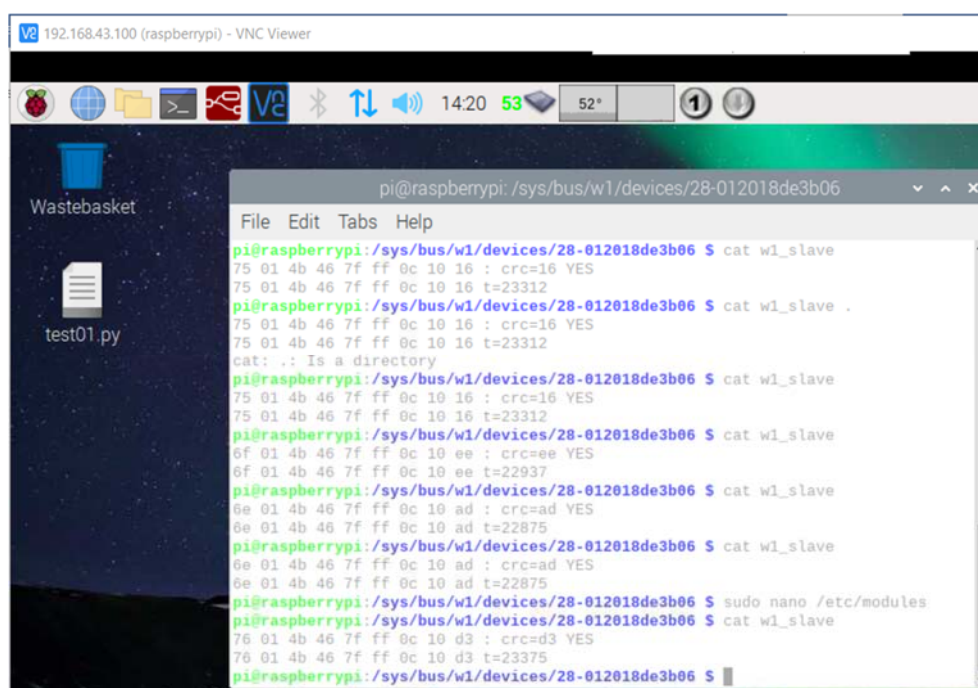
Exécutez ensuite ces deux commandes (une par une) pour activer ces deux modules :

- modprobe w1-gpio
- modprobe w1-therm



Dans le fichier /boot/config.txt, rajoutez cette ligne qui permet de corriger un problème lié aux mises à jour du Pi : dtoverlay=w1-gpio. Et puis on doit redémarrer ensuite la Raspberry Pi.

Pour lire les valeurs de la sonde, il suffit de se rendre dans le dossier /sys/bus/w1/devices/ et de rentrer ensuite dans le dossier commençant par 28-. Enfin, lisez simplement le fichier w1\_slave avec la commande cat.



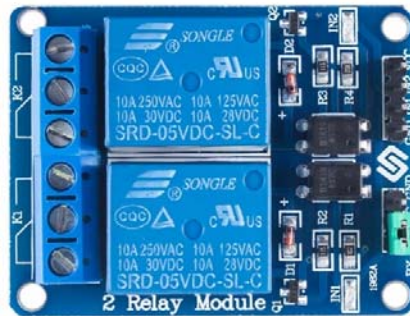
```

pi@raspberrypi: /sys/bus/w1/devices/28-012018de3b06
File Edit Tabs Help
pi@raspberrypi:/sys/bus/w1/devices/28-012018de3b06 $ cat w1_slave
75 01 4b 46 7f ff 0c 10 16 : crc=16 YES
75 01 4b 46 7f ff 0c 10 16 t=23312
pi@raspberrypi:/sys/bus/w1/devices/28-012018de3b06 $ cat w1_slave .
75 01 4b 46 7f ff 0c 10 16 : crc=16 YES
75 01 4b 46 7f ff 0c 10 16 t=23312
cat: .: Is a directory
pi@raspberrypi:/sys/bus/w1/devices/28-012018de3b06 $ cat w1_slave
75 01 4b 46 7f ff 0c 10 16 : crc=16 YES
75 01 4b 46 7f ff 0c 10 16 t=23312
pi@raspberrypi:/sys/bus/w1/devices/28-012018de3b06 $ cat w1_slave
6f 01 4b 46 7f ff 0c 10 ee : crc=ee YES
6f 01 4b 46 7f ff 0c 10 ee t=22937
pi@raspberrypi:/sys/bus/w1/devices/28-012018de3b06 $ cat w1_slave
6e 01 4b 46 7f ff 0c 10 ad : crc=ad YES
6e 01 4b 46 7f ff 0c 10 ad t=22875
pi@raspberrypi:/sys/bus/w1/devices/28-012018de3b06 $ cat w1_slave
6e 01 4b 46 7f ff 0c 10 ad : crc=ad YES
6e 01 4b 46 7f ff 0c 10 ad t=22875
pi@raspberrypi:/sys/bus/w1/devices/28-012018de3b06 $ sudo nano /etc/modules
pi@raspberrypi:/sys/bus/w1/devices/28-012018de3b06 $ cat w1_slave
76 01 4b 46 7f ff 0c 10 d3 : crc=d3 YES
76 01 4b 46 7f ff 0c 10 d3 t=23375
pi@raspberrypi:/sys/bus/w1/devices/28-012018de3b06 $
  
```

On voit qu'on a bien YES de présent. Si non, la Raspberry Pi n'arrive pas à communiquer avec la sonde DS18B20 identifier par 28-012018de3b06.

La température est exprimée en degrés Celsius, multipliée par 1000. Ici, la température est de 23.375°C. Elle est représentée sous la forme t=23375.

Pour commander le Relay avec la Raspberry :



Pour exporter le pin vers l'espace utilisateur

```
echo "17" > /sys/class/gpio/export
```

Pour définir la broche 17 comme sortie

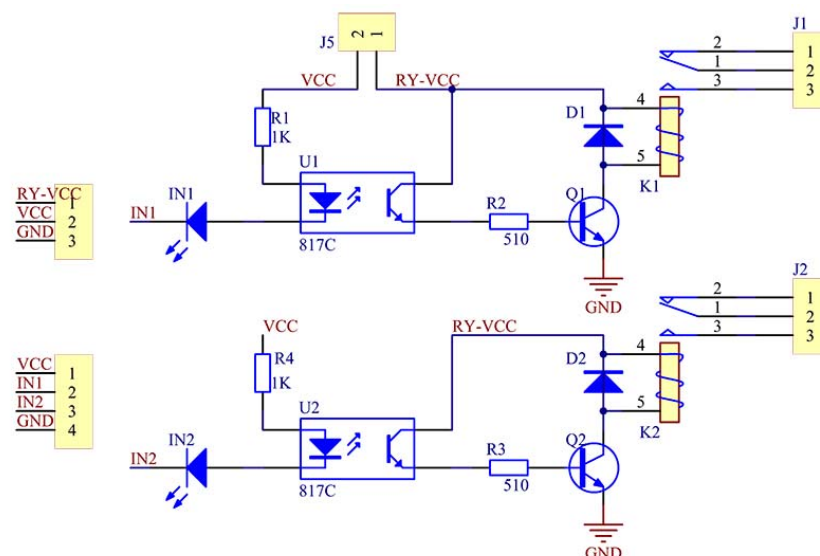
```
echo "out" > /sys/class/gpio/gpio17/direction
```

On met la broche 17 en position haute

```
echo "1" > /sys/class/gpio/gpio17/value
```

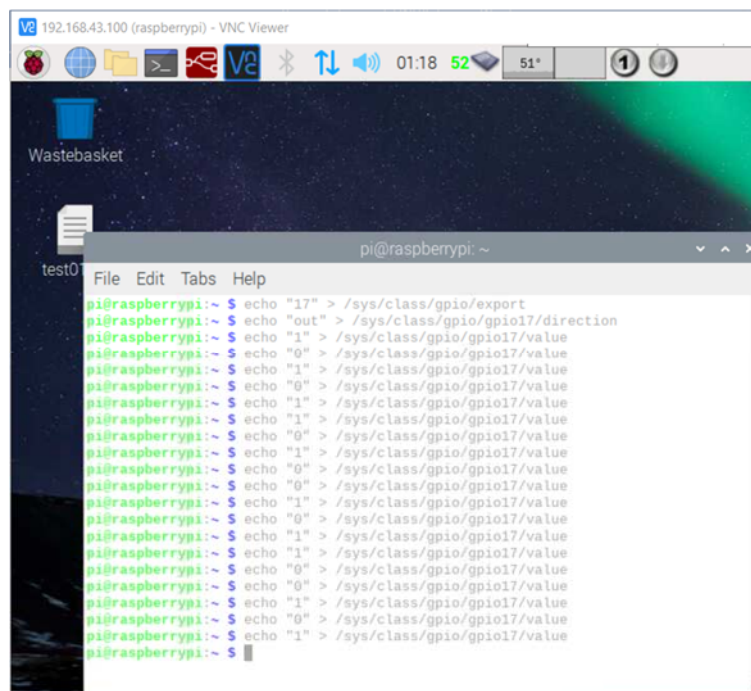
On met la broche 17 à un niveau bas

```
echo "0" > /sys/class/gpio/gpio17/value
```



On voit que le Relay travail avec une logique inverser, c'est-à-dire que si on met la broche 17 à « 1 » la bobine K1 n'est pas alimenter avec VCC. Par contre si on met la bronche 17 à « 0 » la bobine K1 est bien alimenter avec VCC, et par conséquence on fait la fermeture du NO (contacts normalement ouverts) avec le COM(Common).





```

pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberrypi:~$ echo "17" > /sys/class/gpio/export
pi@raspberrypi:~$ echo "out" > /sys/class/gpio/gpio17/direction
pi@raspberrypi:~$ echo "1" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "0" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "1" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "0" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "1" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "0" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "1" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "0" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "0" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "1" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "0" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "1" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "0" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "1" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "0" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "1" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "0" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$ echo "1" > /sys/class/gpio/gpio17/value
pi@raspberrypi:~$

```

Figure 5: Test pour commander le Relay

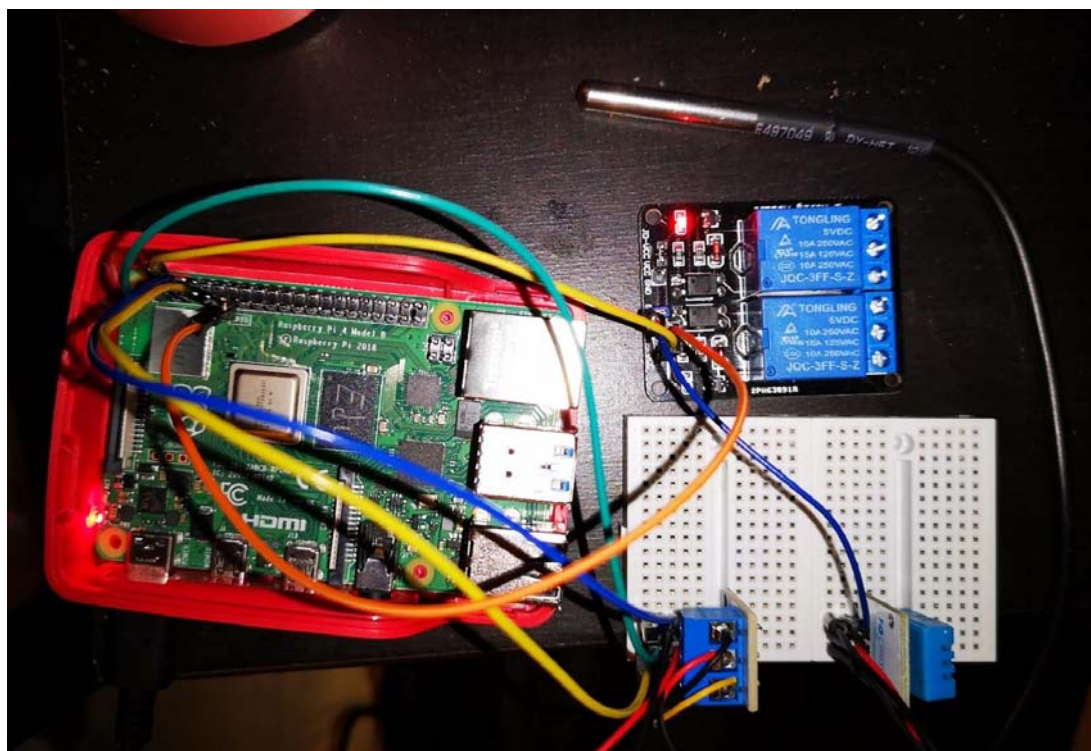


Figure 6: L'état du Relay en « ON » avec GPIO-17 = 0

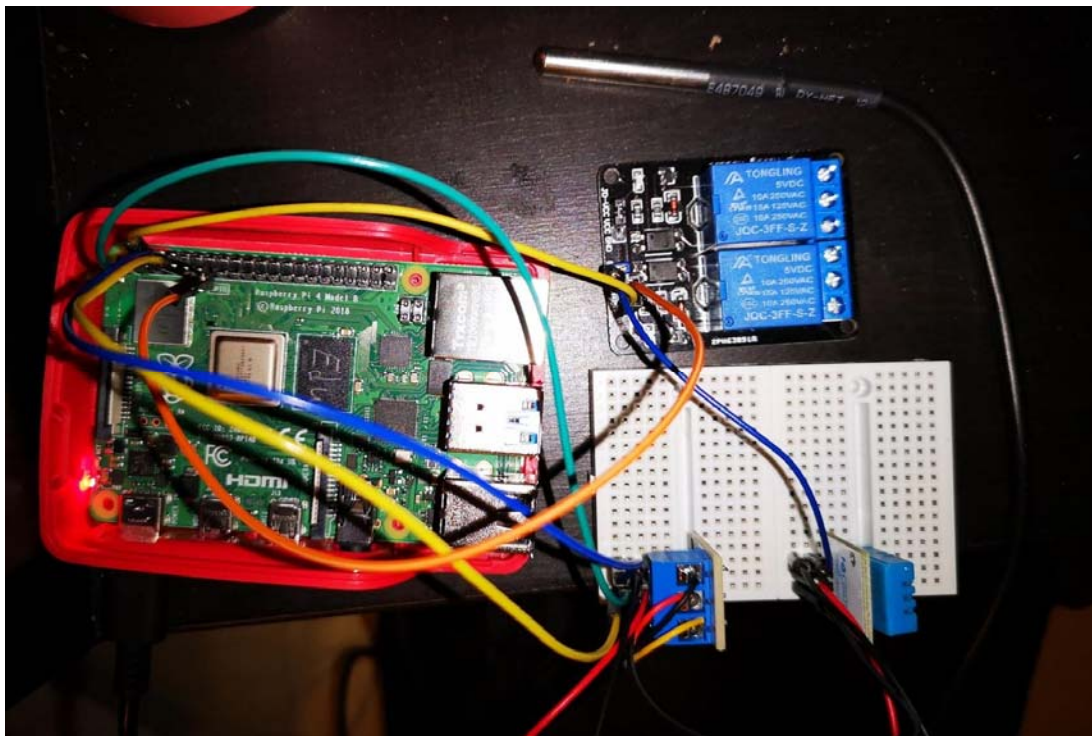


Figure 7: L'état du Relay en « OFF » avec GPIO-17 = 1



Pour l'interface graphique avec Node-Red :

On a installé dans la palette les Nodes suivantes :

- node-red-dashboard
- node-red-contrib-ui-led

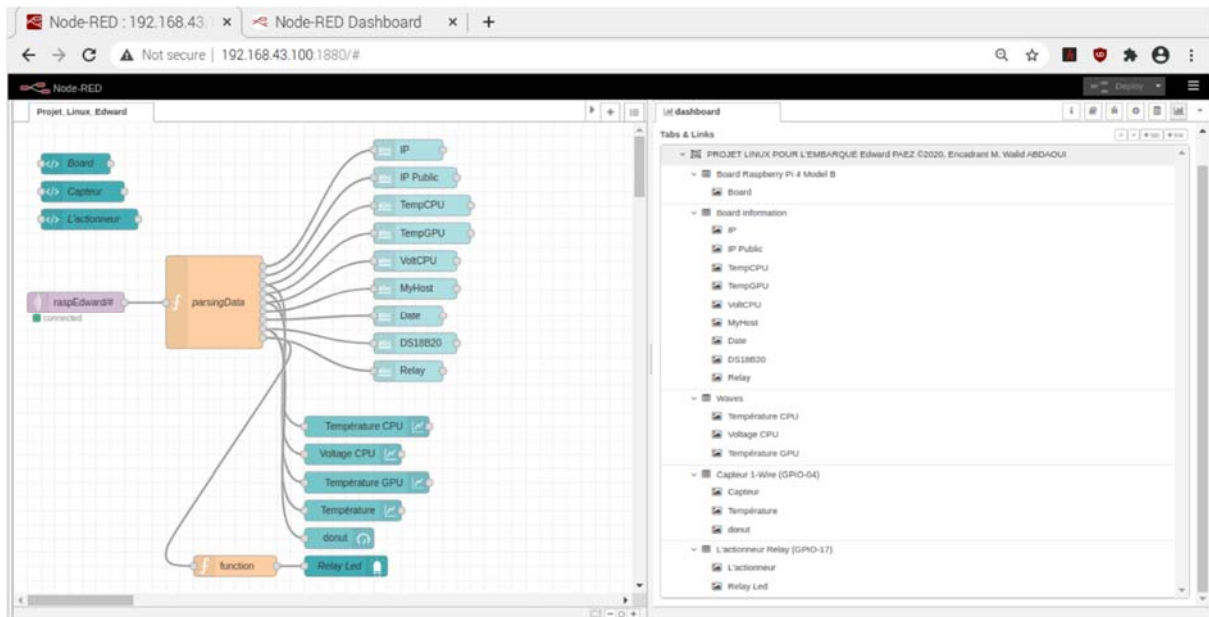


Figure 8: La version final de mon flows Node-RED

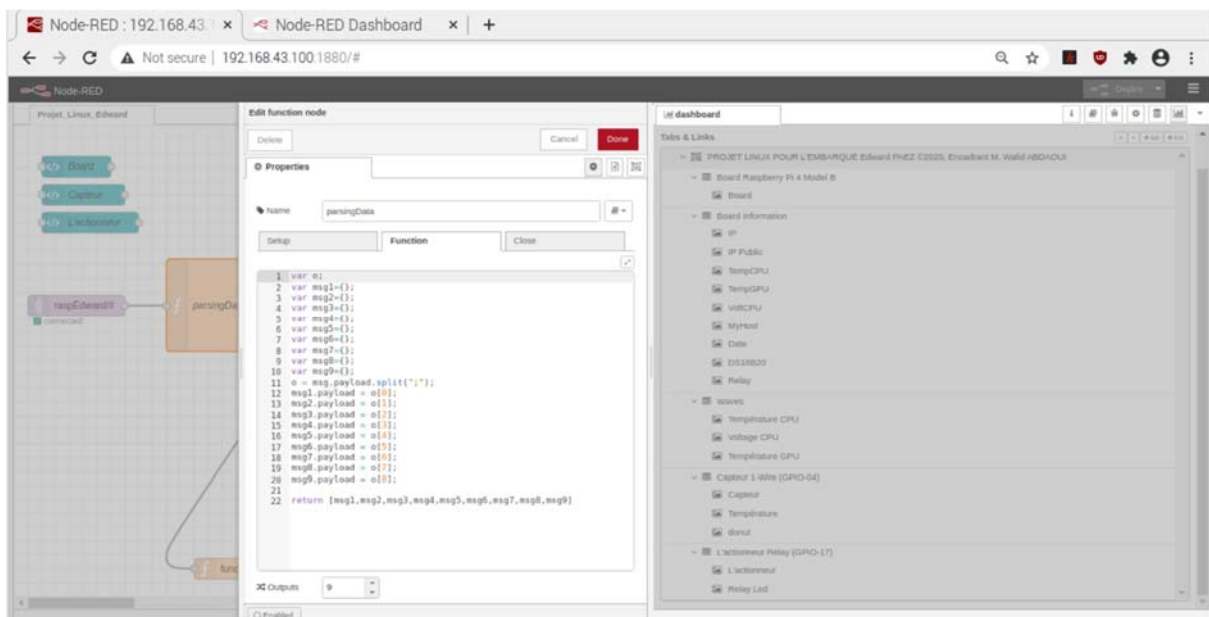


Figure 9: La fonction Parsing utilisé en Javascript pour faire la séparation de variables

```

pi@raspberrypi: ~
File Edit Tabs Help

ÉCOLE D'INGÉNIEURS SUP GALILÉE
INSTRUMENTATION ING 2
PROJET LINUX POUR L'EMBARQUÉ
Edward PAEZ ©2020, Encadrant M. Walid ABDAOUI

♦ Adresse IP de la RPI: 192.168.43.100
♦ Adresse IP publique de la RPI: 80.215.78.6
♦ Température du CPU et GPU
  - Temp.CPU ▶ 63.783
  - Temp.GPU ▶ 64.2°C
  - Volt.CPU ▶ 0.8500V
♦ Nom de la RPI : raspberrypi
♦ Date et heure : Nous sommes le Thu 24 Dec 18:49:14 UTC 2020
♦ Capteur DS18B20 branché sur les GPIO-04 : 23.562 °C
♦ L'actionneur Relais branché sur les GPIO-17 : 1; Relay OFF
  
```

Figure 10: L'exécution du programme Projet\_Edward.sh

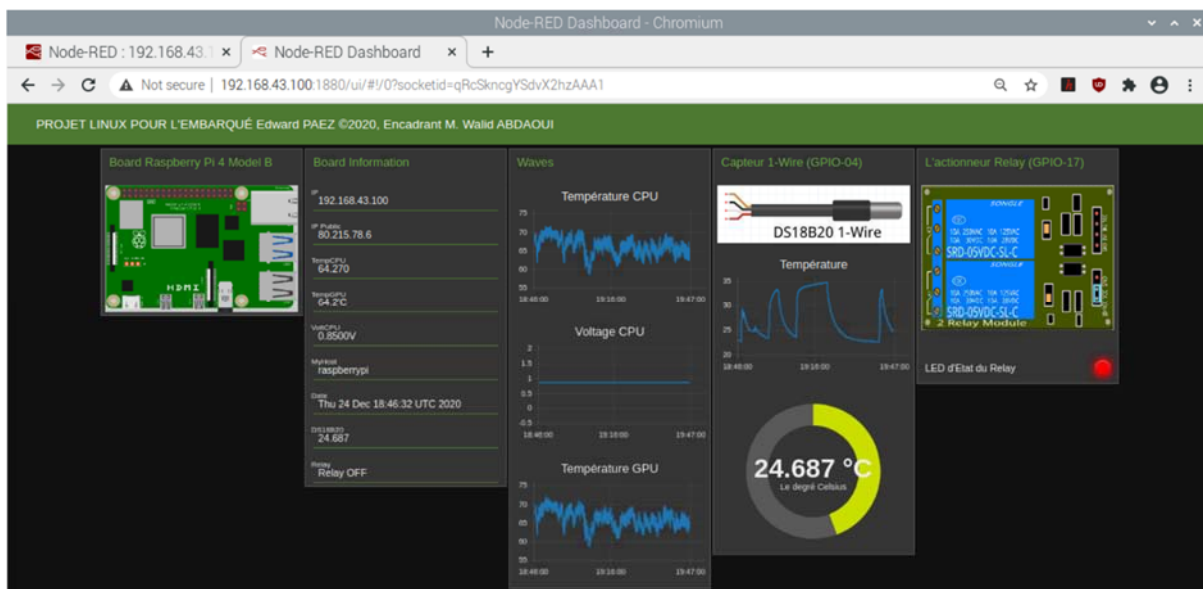


Figure 11: Mon live data Dashboard

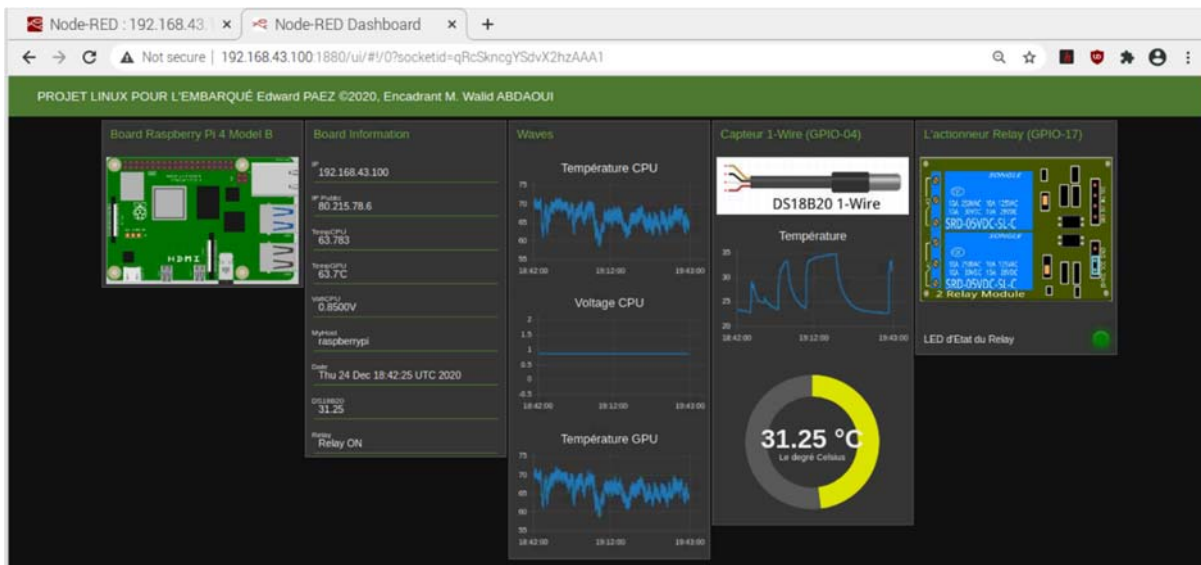


Figure 12: Mon live data Dashboard

On peut constater que notre système suivre en temps réel la consigne de température fixé dans notre programme (Projet\_Edward.sh), c'est-à-dire, qu'un fois la température mesurer par le capteur DS18B20 est bien supérieur à 25°C l'état de l'actionneur change. L'état OFF du Relay "LED ROUGE" vers l'état ON "LED VERT".

## Conclusion

On a tout d'abord pris en main, La carte Raspberry Pi 4 Model B, le capteur température DS18B20 avec le bus de communication 1-Wire, l'actionneur 2 Module Relay, Node-RED, la programmation en langage Bash pour faire fonctionner le système et aussi un peu de Javascript.

En effet, ce projet m'a permis de mettre en pratique les connaissances que nous avons acquises pendant le cours des Travaux Pratiques et aussi pendant le cours de Linux pour l'embarque.

En plus, ce qui m'a particulièrement intéressé est d'avoir effectué un travail de conception dans son intégralité, j'ai remarque aussi que le développement de systèmes embarqués nécessite des connaissances à la fois en électronique et en informatique, la documentation (datasheet) sur les composants utilisés. C'est essentiel

## Références Bibliographique

- Raspberrypi-France. Quelles évolutions de l'OS pour Raspberry Pi 4 [en ligne] (page consultée le 16/12/2020) <https://www.raspberrypi-france.fr/category/actualites/>
- Wikipédia, l'encyclopédie libre. Système embarqué, [en ligne] (page consultée le 17/12/2020) [https://fr.wikipedia.org/wiki/Syst%C3%A8me\\_embarqu%C3%A9](https://fr.wikipedia.org/wiki/Syst%C3%A8me_embarqu%C3%A9)
- L'embarque. La Raspberry Pi se muscle pour l'embarquer avec le module Compute Module, [en ligne] (page consultée le 21/12/2020) [http://www.lembarque.com/la-raspberry-pi-se-muscle-pour-lembarque-avec-le-module-compute-module-4\\_010467](http://www.lembarque.com/la-raspberry-pi-se-muscle-pour-lembarque-avec-le-module-compute-module-4_010467)
- Tutorialspoint. Unix, Linux Commands, [en ligne] (page consultée le 21/12/2020) [https://www.tutorialspoint.com/unix\\_commands/](https://www.tutorialspoint.com/unix_commands/)
- Yavin4. Afficher la température CPU du Raspberry Pi, [en ligne] (page consultée le 21/12/2020) <https://yavin4.ovh/index.php/2014/12/31/raspberry-pi-recuperer-la-temperature-cpu-dans-un-fichier/#:~:text=Ce%20script%20r%C3%A9cup%C3%A8re%20d'abord,dans%20plusieurs%20conditions%20%C2%AB%20if%20%C2%BB>
- Le blog de Claude. Contrôle de la température et fréquence du CPU, [en ligne] (page consultée le 21/12/2020) <http://emery.claude.free.fr/temperature-frequence.html>
- Automation-sense. Qu'est-ce que le protocole MQTT, [en ligne] (page consultée le 21/12/2020) <https://www.automation-sense.com/blog/automatisme/qu-est-ce-que-le-protocole-mqtt.html>