GOTRONIC
ROBOTIQUE ET COMPOSANTS ÉLECTRONIQUES



Joy-Pi : le kit complet pour Raspberry

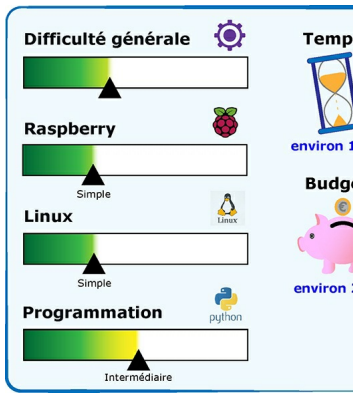
Partie 1 : Présentation

Aujourd'hui, nous allons explorer les possibilités qu'offre la valise Joy-Pi, conçue pour les cartes Raspberry 3, 3B+, pi 4.

Cette valise, facilement transportable, permet de regrouper dans un seul et même environnement un ensemble complet de prototypage sur de nombreux capteurs, modules et interfaces (servomoteurs, moteurs pas à pas).

La prise en main est facile grâce à son écran et son clavier-souris pour naviguer à travers l'OS Raspbian.

C'est un kit complet prêt à être utiliser n'importe où et n'importe quand. Il suffit d'insérer la carte Raspberry Pi et de brancher l'alimentation pour pouvoir commencer à travailler.



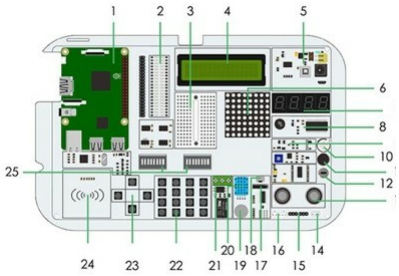
La valise met à disposition :

Capteurs :

- (9) Capteur de lumière
- (11) Capteur sonore
- (12) Détecteur de mouvement
- (13) Capteur à ultrasons
- (17) Capteur d'inclinaison
- (19) Capteur tactile
- (20) Capteur de température et d'humidité
- (24) Module RFID

Affichage :

- (2) Indicateur à LED pour le port GPIO
- (6) Matrice à LED 8 x 8
- (4) Afficheur LCD 16 x 2 caractères
- (7) Afficheur 4 digits à 7 segments
- Écran de 7 pouces + Caméra 2Mpx



Connexions :

- (3) Plaque de montage rapide
- (14/15) Contrôle de servomoteurs
- (16) Contrôle de moteurs pas-à-pas
- (18) Connecteur pour récepteur infrarouge

Modules divers :

- (8) Vibreur
- (10) Buzzer
- (21) Relais
- (22) Matrice à boutons-poussoirs
- (23) Boutons-poussoirs directionnels

Certains modules ne peuvent pas fonctionner simultanément car ceux-ci utilisent les mêmes broches du port GPIO (*General Purpose Input Output*).

D'autres modules devront être activés/désactivés grâce aux deux dip-switches selon l'utilisation.

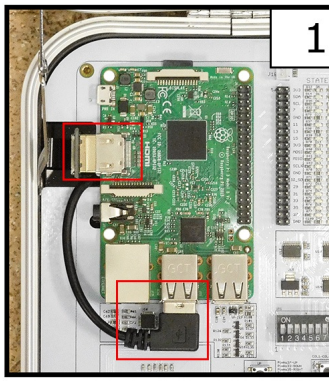
Utilisation des dip-switches :

Capteurs et modules	Dip-switch	Inverseurs
Matrice de boutons-poussoirs	Gauche	1 à 8
Boutons-poussoirs	Gauche	5 à 8
Module vibreur	Droit	1
Capteur d'inclinaison	Droit	2
Moteur pas-à-pas	Droit	3, 4, 5 et 6
Servomoteur	Droit	7 et 8

Partie 2 : Configuration

Insérez la carte micro-SD dans votre carte Raspberry.

Ouvrez la valise,
[1] raccordez votre Raspberry au cordon USB et au cordon HDMI,
[2] vissez-la,
[3] et reliez les ports GPIO avec la nappe.



Branchez l'alimentation (en haut à droite de la valise), et appuyez sur l'interrupteur.
Votre écran s'allume et l'OS Raspbian démarre.

Pour utiliser le clavier, mettez le switch sur ON (sur le bord droit) et insérez le récepteur USB sur votre carte Raspberry.

Un dossier avec des exemples de programme simples sont disponibles sur le Bureau (dossier **Joy-Pi**).

Mettre l'OS en français :

Au démarrage, l'OS est configuré en allemand.

Pour le configurer en français :

- Menu Démarrer >> ***Einstellungen*** >> ***Raspberry-Pi-Konfiguration***
- Onglet ***Lokalisierung***
- Bouton ***Sprachumgebung*** : FR – votre pays – UTF 8
- Bouton ***Zeitzone*** : votre localisation – la ville de votre fuseau horaire
- Bouton ***Tastatur*** : *[ne pas toucher au 1er paramètre]* – French – French
- Bouton ***Wifi-Land*** : votre pays
- Appuyez sur ***OK*** puis sur ***Ja***

Au redémarrage, l'OS sera en français.

Mise à jour du système :

Lancez un Terminal [wp-svg-icons icon= »console » wrap= »i »], exécutez les commandes suivantes, les unes à la suite des autres en appuyant sur la touche ***En***

sudo su
apt-get upgrade
apt-get update
apt-get install build-essential python-dev python-smbus python-pip
apt-get install python-imaging
pip install RPi.GPIO

Faites un redémarrage de votre carte Raspberry (***menu Démarrer >> Shutdown >> Reboot***).

Partie 3 : Réalisations
Projet 1 : Holorge

Nous allons concevoir un programme permettant d'afficher l'heure sur l'afficheur 7 segments.

Créer un dossier « Horloge » dans vos ***Documents***.

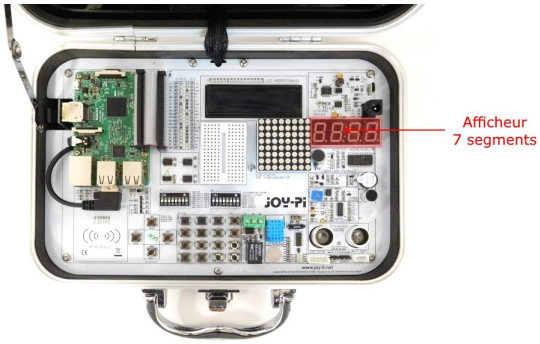
Téléchargez la librairie pour l'afficheur 7 segments [lien [wp-svg-icons icon= »github-3″ wrap= »i »] GitHub].

Placez-la dans le dossier « Horloge ». Clic-droit sur la librairie, ***Extraire ici***.

Sur le dossier extrait, clic-droit, ***Ouvrir dans un Terminal [wp-svg-icons icon= »console » wrap= »i »]***.

Exécutez la commande *sudo python setup.py install*

Dans le dossier « Horloge », clic-droit, ***Créer un fichier vide***, nommez le fichier « horloge.py ».



Ouvrez le fichier, copiez-collez le programme suivant et sauvegardez :

```
#!/usr/bin/python

import time
import datetime

from Adafruit_LED_Backpack import SevenSegment

segment = SevenSegment.SevenSegment(address=0x70)

segment.begin()
try:

  while(True):

    now = datetime.datetime.now()
    hour = now.hour
    minute = now.minute
    second = now.second

    segment.clear()
    segment.set_digit(0, int(hour / 10)) #dizaine
    segment.set_digit(1, hour % 10) #unité
    segment.set_digit(2, int(minute / 10)) #dizaine
    segment.set_digit(3, minute % 10) #unité
    segment.set_colon(second % 2) #1 Hertz
    segment.write_display()
    time.sleep(1)

except KeyboardInterrupt:

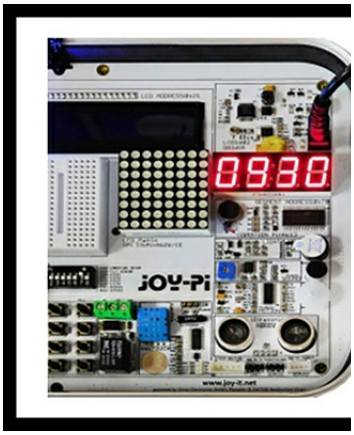
  segment.clear()
  segment.write_display()
```

Dans vos ***Documents*** faites un clic-droit sur le dossier « Horloge »,
Ouvrir dans un Terminal [wp-svg-icons icon= »console » wrap= »i »].

Tapez la commande suivante puis appuyez sur la touche ***Entrée*** :

sudo python horloge.py

L'afficheur 7 segment vous affiche l'heure.



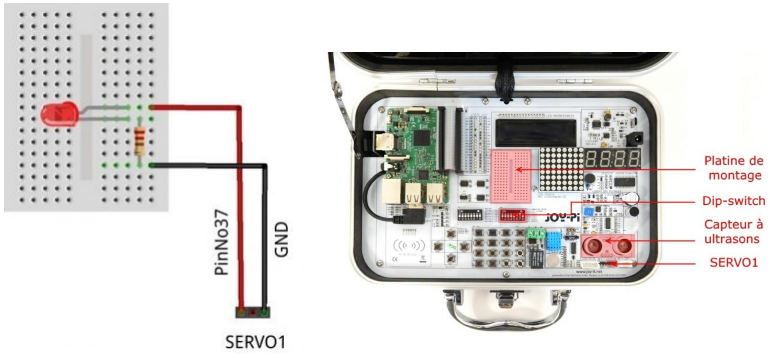
Projet 2 : Détecteur d'obstacle

Dans ce projet, nous allons faire varier l'intensité lumineuse d'une LED installée sur la platine de montage, en fonction de la distance entre le capteur à ultrasons et l'obstacle le plus proche. Plus l'obstacle est proche, plus l'intensité sera forte.

L'inverseur 7 du dip-switch de droite doit être mis sur ON.

Montez une LED en série avec une résistance de 220Ω sur la platine de montage (non inclus dans la valise).

Reliez le montage suivant le schéma ci-dessous avec des jumpers mâle-femelle (attention à ce que les jumpers ne passent pas au dessus du capteur à ultrasons) :



Dans vos ***Documents***, créez un dossier « Obstacle ».

Dans le dossier « Obstacle », clic-droit, ***Créer un fichier vide***, nommez le fichier « obstacle.py ».

Pas besoin de librairie pour ce programme.

Ouvrez le fichier, copiez-collez le programme suivant et sauvegardez :

```
#!/usr/bin/python
import RPi.GPIO as GPIO
import time

while True :

 GPIO.setmode(GPIO.BOARD)

 TRIG = 36
 ECHO = 32
 led_pin = 37

 GPIO.setup(TRIG,GPIO.OUT)
 GPIO.setup(ECHO,GPIO.IN)
 GPIO.setup(led_pin, GPIO.OUT)

 f = GPIO.PWM(led_pin, 50) #50Hz

 GPIO.output(TRIG, True)
 time.sleep(0.00001)
 GPIO.output(TRIG, False)

 while GPIO.input(ECHO)==0: #début de la mesure, capteur ultrasons

   pulse_start = time.time()

 while GPIO.input(ECHO)==1: #fin de la mesure, capteur ultrasons

   pulse_end = time.time()

 pulse_duration = pulse_end – pulse_start

 distance = pulse_duration * 17150 #valeur constructeur

 if distance <= 99:

   rp = 99-distance

 else:

   rp =1

 f.start(rp)
 f.ChangeDutyCycle(rp) #On fait varier le rapport cyclique pour changer l'intensité lumineuse
 time.sleep(0.02) #50Hz

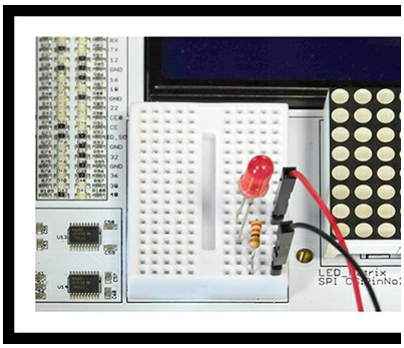
 GPIO.cleanup()
```

Dans vos ***Documents*** faites un clic-droit sur le dossier « Obstacle », ***Ouvrir dans un Terminal [wp-svg-icons icon= »console » wrap= »i »]***.

Tapez la commande suivante puis appuyez sur la touche ***Entrée*** :

sudo python obstacle.py

Le Terminal vous affiche la distance en centimètre entre l'obstacle le plus proche et le capteur, et la LED clignote de plus en plus vite si l'obstacle se rapproche.



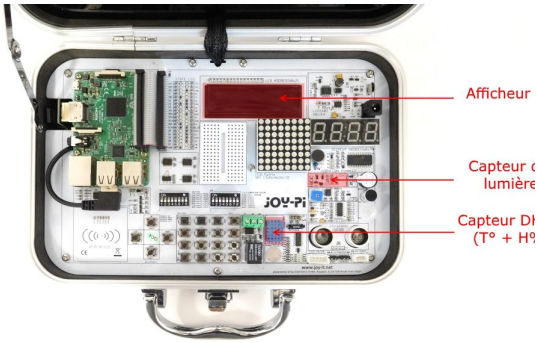
Projet 3 : Station météo

Ce dernier projet va permettre d'afficher sur l'écran LCD la température , l'humidité et l'intensité lumineuse
ambiante grâce au DHT11 et au luxmètre.

Créer un dossier « Meteo » dans vos ***Documents***.

Téléchargez les librairies (***Clone or download*** >> ***Download ZIP***) :

- Afficheur LCD [lien [wp-svg-icons icon= »github-3″ wrap= »i »] GitHub]
- Capteur de température et d'humidité [lien [wp-svg-icons icon= »github-3″ wrap= »i »] GitHub]



Placez les librairies dans le dossier « Meteo ».

Pour chaque librairie :

- Clic-droit sur la librairie, ***Extraire ici***.
- Sur le dossier extrait, clic-droit, ***Ouvrir dans un Terminal [wp-svg-icons icon= »console » wrap= »i »]***.
- Exécutez la commande *sudo python setup.py install*

Dans le dossier « Meteo », clic-droit, ***Créer un fichier vide***, nommez le fichier « meteo.py ».

Ouvrez le fichier, copiez-collez le programme suivant et sauvegardez :

```
#!/usr/bin/python
import sys
import time
import smbus
import Adafruit_DHT
import RPi.GPIO as GPIO
import Adafruit_CharLCD as LCD

if(GPIO.RPI_REVISION == 1):

  bus = smbus.SMBus(0)

else:

  bus = smbus.SMBus(1)

lcd_columns = 16

lcd_rows = 2
lcd = LCD.Adafruit_CharLCDBackpack(address=0x21)
lcd.set_backlight(0)
sensor = Adafruit_DHT.DHT11
pin = 4
humidity, temperature = Adafruit_DHT.read_retry(sensor, pin)
t=str(temperature)
h=str(humidity)

#Luxmetre
class LightSensor():

  def convertToNumber(self, data):

    return (int((data[1] + (256 * data[0])) / 1.2)) #valeur constructeur

  def readLight(self):

    data = bus.read_i2c_block_data(0x5C,0x10) #0x5C = adresse du capteur ; 0x10 = standard de mesure
    return self.convertToNumber(data)

#Temperature & Humidite
def dht():

  humidity, temperature = Adafruit_DHT.read_retry(sensor, pin)
  t=str(temperature)
  h=str(humidity)
  if humidity is not None and temperature is not None:

  lcd.message('T='+t+'*C H='+h+'%')

#Main
def main():

  print "En etat de marche... \nCTRL+C ou Z pour arreter"
  while True:

    GPIO.cleanup()
    dht()

    #Lumiere
    sensor = LightSensor()
    l=str(sensor.readLight())
    lcd.message('\n'+l+' lux')

    time.sleep(2)
    lcd.clear()

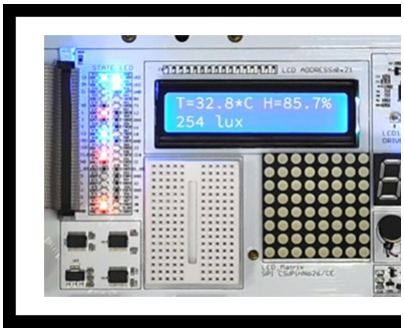
#Run
if __name__ == '__main__':
  main()
```

Dans vos ***Documents*** faites un clic-droit sur le dossier « Meteo », ***Ouvrir dans un Terminal [wp-svg-icons icon= »console » wrap= »i »]***.

Tapez la commande suivante puis appuyez sur la touche ***Entrée*** :

sudo python meteo.py

L'afficheur LCD affiche sur la 1ère ligne la température (°C) et l'humidité (%), et sur la 2ème ligne la valeur en lux de la lumière ambiante.



Démonstration

Voir la vidéo sur YouTube