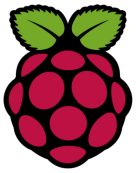
CHARLES CARNUS
LYCÉE TECHNIQUE
ENSEIGNEMENT SUPÉRIEUR



Raspberry Pi

TP (suite)

Prof : **Kamal Boudjelaba**

19 mars 2022

B T S S N

LTP Charles Carnus
Avenue de Saint Pierre - Bourran - 12000 RODEZ, France

IR–EC

1. Introduction : commandes Linux Raspberry Pi

Commande	Signification
`$ raspi-config`	Outil de configuration pour Raspberry Pi
`$ raspistill`	Prendre une photo avec le module camera du Raspberry Pi
`$ wget`	Téléchargement d'un fichier sur Raspberry Pi
`$ apt-get install <paquet>`	Installer le ou les paquet(s) spécifié(s) `$ sudo apt-get install conda pip`
`$ apt-get remove <paquet>`	Désinstaller un paquet `$ sudo apt-get remove conda`
`$ cd <chemin>`	Change de dossier (va au dossier indiqué entre dans <...>) `$ cd /home/pi`
`$ ls`	Lister les fichiers et dossiers présents dans le dossier actuel, ou celui spécifié `$ ls` ou `$ ls /home/pi`
`$ mkdir <dossier>`	Créer un dossier à l'emplacement actuel ou spécifié `$ mkdir MonDossier` ou `$ mkdir /home/pi/MonDossier`
`$ pwd`	Permet de savoir dans quel dossier vous êtes (à ne pas confondre avec `passwd`) `$ pwd`
`$ tree`	Analyser l'emplacement actuel dans l'arborescence des fichiers. Il montrera toute l'arborescence existante dans le dossier actuel ou spécifié `$ tree`

Commande	Signification
`$ tar -c`	Utiliser `tar` pour regrouper plusieurs fichiers dans une même archive (généralement avec gzip afin de compresser des fichiers) `$ tar -cvfz archive.tar.gz /home/pi/Documents/MonDossier` `-c` : création d'une archive `-v` : mode verbeux `-f` : on spécifie le nom du fichier juste après `-z` : on utilise gzip pour la compression
`$ tar -x`	Même commande, mais pour extraire les fichiers `$ tar -xvfz archive.tar.gz`

2. Exécuter des programmes sur le Raspberry Pi (Linux ARM)

2.1 Programme en C

1. Compiler le programme `ProgC.c` :
   ```
   $ gcc -o TestC ProgC.c
   ```
2. Exécuter le programme :
   ```
   $ ./TestC
   ```

2.2 Programme en C++

1. Compiler le programme `ProgCPP.cpp` :
   ```
   $ g++ -o TestCPP ProgCPP.cpp
   ```
2. Exécuter le programme :
   ```
   $ ./TestCPP
   ```

2.3 Programme en Python

1. Exécuter le programme `ProgPython.py` : ce n'est pas utile de compiler le programme
   ```
   $ python ProgPython.py
   ```

3. Les broches GPIO (entrées/sorties)

3.1 Programme en C avec la bibliothèque Wiring Pi

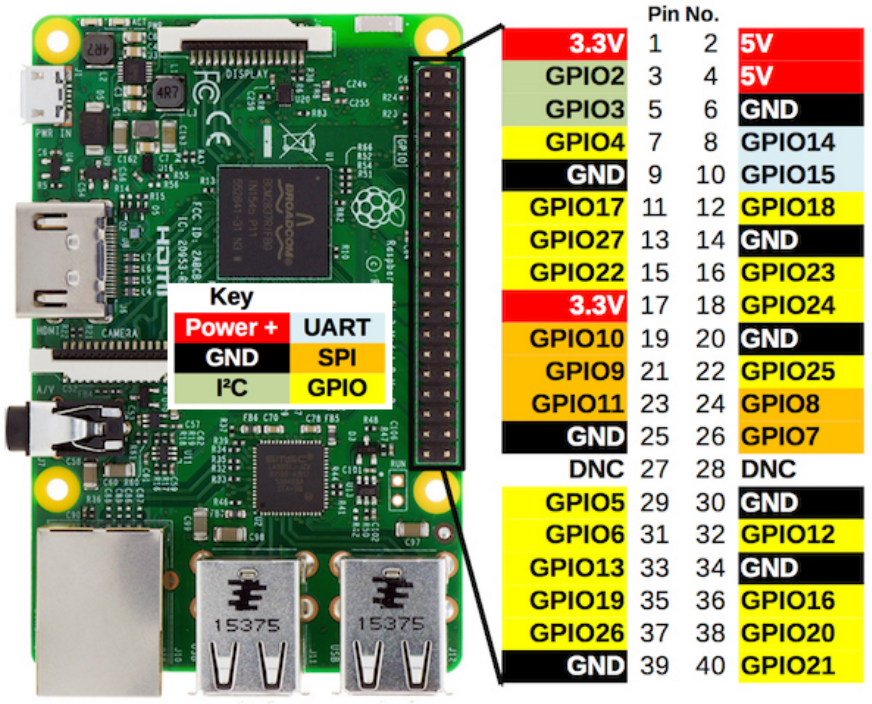
1. Compilation : `$ gcc -o Test_C Prog.c -l wiringPi`
2. Exécution : `$ sudo ./Test_C` (sudo pour permettre l'accès aux ports d'entrées/sorties)

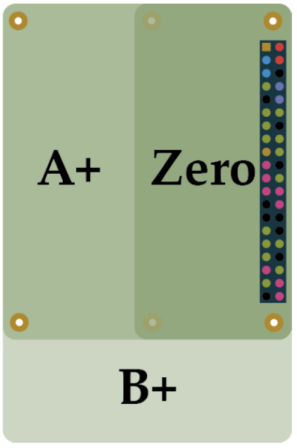
3.2 Programme en C++ avec la bibliothèque Wiring Pi

1. Compilation : `$ g++ -o Test_CPP Prog.cpp -l wiringPi`
2. Exécution : `$ sudo ./Test_CPP`

3.3 Programme en Python avec la bibliothèque GPIO

1. Exécution : `$ python Prog.py`





GPIO (General Purpose IO)
SPI (Serial Peripheral Interface)
I²C (Inter-integrated Circuit)
UART (Universal Asyncronous Receiver/Transmitter)
PCM (Pulse Code Modulation)
Ground
5v (Power)
3.3v (Power)

Figure 1. GPIO et Wiring Pi

4. Travaux Pratiques

4.1 TP 1 : Création d'une application avec Qt Creator

Voir le tutoriel disponible à partir de ce lien vidéo
Le montage à réaliser est donné dans la figure 2.

Figure 2. Circuit

4.2 TP 2 : Envoyer des e-mails à partir du Raspberry Pi

But : Envoyer des e-mails contenant des images en utilisant un Raspberry Pi, un détecteur de mouvement PIR et Python. A chaque fois qu'un mouvement est détecté, la caméra Raspberry Pi prendra une photo et enverra un e-mail avec la photo en pièce jointe.

Configuration du compte Gmail :

- Se connecter à votre compte Gmail en saisissant les identifiants de connexion.
- Cliquer sur la photo de profil puis cliquer sur "Compte Google".
- Sous "Connexion et sécurité", cliquer sur "Applications et sites connectés"
- Cliquer sur "Autoriser les applications moins sécurisées" pour l'activer. (Par défaut, il est désactivé)
- Remarque : à la fin du TP, vous pouvez remettre les paramètres initiaux.

Composants et matériel :

- Capteur PIR
  - Connecter la broche VCC du capteur de mouvement PIR à la broche + 5V du Raspberry Pi.
  - Connecter la broche GND du capteur de mouvement PIR à la broche GND du Raspberry Pi.
  - Connecter la broche DATA du capteur PIR au GPIO24 du Raspberry Pi.
- LED rouge et une résistance de 220 Ω
  - Brancher la résistance en série avec la LED entre le GPIO20 et la broche GND.
- Caméra
  - Activer le module caméra dans Raspi-Config
  - La caméra sera brancher par le prof.

Code Python : Ne pas oublier de saisir dans le code

- l'adresse e-mail de l'expéditeur
- le mot de passe de l'expéditeur
- l'adresse e-mail du destinataire

Le code lira la sortie du capteur et, lors de la détection de mouvement, capturera une image et l'enregistrera dans le dossier de la base de données.
Lorsqu'on exécute le code pour la première fois, il créera automatiquement le dossier. Une fois que l'appareil photo capture une image, l'image sera jointe à l'e-mail et envoyée à l'adresse de l'expéditeur.

```
import RPi.GPIO as GPIO
import time
import datetime
import picamera
import os
import smtplib
from email import encoders
from email.mime.base import MIMEBase
from email.mime.multipart import MIMEMultipart

camera = picamera.PiCamera()
GPIO.setmode(GPIO.BCM)

GPIO.setup(23, GPIO.IN) #PIR
GPIO.setup(24, GPIO.OUT) #LED

"""
ts = time.time()
st = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d %H:%M:%S')
"""

COMMASPACE = ', '

def Send_Email(image):
    sender = 'Votre Adresse Mail'
    gmail_password = 'Votre Mot De Passe'
    recipients = ['Adresse Mail Du Destinataire']

    # Creation du message
    outer = MIMEMultipart()
    outer['Subject'] = 'Test Fichier Joint'
    outer['To'] = COMMASPACE.join(recipients)
    outer['From'] = sender
    outer.preamble = 'You will not see this in a MIME-aware mail reader.\n'

    # Liste des pièces jointes
    attachments = [image]

    # Ajout de la pièce jointe au message
    for file in attachments:
        try:
            with open(file, 'rb') as fp:
                msg = MIMEBase('application', "octet-stream")
                msg.set_payload(fp.read())
            encoders.encode_base64(msg)
            msg.add_header('Content-Disposition', 'attachment', filename=os.path.basename(file))
            outer.attach(msg)
        except:
            print("Ouverture impossible de la pièce jointe. Erreur: ", sys.exc_info()[0])
            raise

    composed = outer.as_string()

    # Envoi du message (mail)
    try:
        with smtplib.SMTP('smtp.gmail.com', 587) as s:
            s.ehlo()
            s.starttls()
            s.ehlo()
            s.login(sender, gmail_password)
            s.sendmail(sender, recipients, composed)
            s.close()
        print("Email envoyé!")
    except:
        print("Impossible d-envoyer le mail. Erreur: ", sys.exc_info()[0])
        raise

try:
    time.sleep(2) # to stabilize sensor
    while True:
```

```
        ts = time.time()
        st = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d %H:%M:%S')
        if GPIO.input(23):
            GPIO.output(24, True)
            time.sleep(0.5) #LED allumée pendant 0.5 sec
            print("Mouvement Detecté at {}".format(st))
            ##Ajout timestamp pour image
            camera.capture('image_Time_{}.jpg'.format(st))
            image = ('image_Time_{}.jpg'.format(st))
            Send_Email(image)
            time.sleep(2)
            GPIO.output(24, False)
            time.sleep(5) #tPour éviter la détection multiple

        time.sleep(0.1) #délai de boucle, doit être inférieur au délai de détection

except:
    GPIO.cleanup()
```