Documentation technique sur les connexions de la RaspberryPi

Table des matières

MoveAndGo - GPL-3.0 License

1 - Initialisation de la connexion Bluetooth

Une fois connecté en SSH à la RaspberryPi depuis votre ordinateur, entrez la commande *bluetoothctl.*

Entrez ensuite la commande *agent on* afin d'activer le Bluetooth sur la RaspberryPi.

Pour lancer une détection de tous les appareils disponibles via Bluetooth, entrez la commande *scan on*.

Une fois que vous avez repéré votre appareil, copiez son adresse UUID, que nous appellerons désormais adress_UUID, et lancez la commande *trust adress_UUID.* Ceci permet à la RaspberryPi de repérer directement votre appareil la prochaine fois.

Si vous voulez appairer votre appareil à la carte RaspberryPI, lancez la commande paire *adress_UUID*.

Enfin si vous voulez lancer une connexion entre la RaspberryPi et votre appareil, entrez la commande *connect adress_UUID.*

2 - Lancement de la connexion pour échange de données entre l'appareil Bluetooth et la RaspberryPi

Afin d'accéder aux fichiers de paramétrisation de la connexion Bluetooth, il faut entrer la commande *cd ../../etc/repo/Armbot_C-main/models.*

```
[pi@raspberrypi:~ $ cd ../../etc/repo/ArmBot_C-main/models/
[pi@raspberrypi:/etc/repo/ArmBot_C-main/models $ ls
 bluetooth.c  bluetooth.h
 pi@raspberrypi:/etc/repo/ArmBot_C-main/models $
```

Ce dossier contient deux fichiers C : bluetooth.c et bluetooth.h. Ces fichiers ne sont pas à modifier à priori, mais vous pouvez y jeter un œil si jamais vous voulez effectuer une modification de l'initialisation de la connexion. Cependant, nous ne vous le recommandons pas. Les fonctions déclarées et implémentées dans ces fichiers sont utilisées dans le fichier main.c, afin d'initialiser la connexion Bluetooth et la RaspberryPi.

```
[pi@raspberrypi:~ $ cd ../../etc/repo/ArmBot_C-main/
[pi@raspberrypi:/etc/repo/ArmBot_C-main $ ls
 client.py  henri  LICENSE  main  main.c  models  raspberry_setup  README.md
 pi@raspberrypi:/etc/repo/ArmBot_C-main $
```


MoveAndGo - GPL-3.0 License

3 - Connexion WiFi/Ethernet

Pour établir une connexion WiFi ou Ethernet entre l'ordinateur et la RaspberryPi, on utilise le principe de client/server. Le serveur correspond à l'ordinateur et le client correspond à la RaspberryPi.

Intéressons-nous au code client.py, qui s'occupe de la connexion avec l'ordinateur. La variable *hote* doit recevoir pour valeur l'adresse IP de l'ordinateur dans le réseau local. Le port est à définir, et il faut mettre le même dans le code MasterServer.py, dont nous parlerons juste ensuite.

L'idée est de créer une socket contenant de la donnée et de l'envoyer au serveur. On crée donc une socket, et on la connecte directement au serveur. Ensuite, on encode la donnée à envoyer, et on envoie la socket. Il faut ensuite refermer la socket afin de pouvoir en utiliser une autre ensuite.

```
  GNU nano 5.4
import socket
import time
import sys

hote ="169.254.93.130" #"169.254.131.184" # "172.20.10.2" #"address ip du server"
port = 15001

def main(argv):
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect((hote, port))
    print("Connection on {}".format(port))

    # my = input("Enter command ")
    my = str(sys.argv[1])
    print("PYTHON ENVOIE :")
    print(my)
    # encode the message
    my_inp = my.encode('utf-8')

    # send request ti server
    s.sendall(my_inp)
    s.close()
    time.sleep(0.01)



if __name__ == "__main__":
   print("debut socket")
   main(sys.argv[1:])
```


MoveAndGo - GPL-3.0 License

4 - Transfert des données reçues par l'appareil Bluetooth vers l'ordinateur

Comme dit précédemment, la fonction startServerBluetooth() est appelée par le fichier main.c pour initialser une connexion Bluetooth entre la Raspberry et l'appareil mobile.

Une fois la connexion établie, la Raspberry reçoit des données lorsque l'appareil mobile lui en envoie, et les affiche dans la console.

Ensuite, nous lançons le fichier client.py, qui permet le transfert de données reçues vers l'ordinateur. Pour cela, nous créons une chaîne de caractère contenant *python3 client.py*, auquel on ajoute en paramètre la donnée reçue.

```c
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include "models/bluetooth.c"
#include <string.h>

bool startServerBluetooth() {
    // TODO modifier la taille du buf si possible
    char buf[1024] = { 0 };
    int s, bytes_read = -1;
    int client = init_server();
    int port = 3;

    do {
        // read data from the client
        memset(buf, 0, sizeof(buf));
        bytes_read = read(client, buf, sizeof(buf));

        if( bytes_read > 0 ) {
            printf("received [%s]\n", buf);
            char src[] = "python3 client.py ";
            strcat(src, buf);
            printf("envoie : [%s]", src);
            system(src);
        }
    } while (bytes_read > 0);

    // close connection
    printf("Closing connection.\n");
    close(client);
    close(s);
    sdp_close( s );

    return true;
}


// ---------------------------------------
// -              MAIN                   -
// ---------------------------------------

int main(int argc, char *argv[]){

    bool success = startServerBluetooth();

    return 1;
}
```

Avant de lancer le code, il faut d'abord le compiler. Pour cela, il faut utiliser la commande : *sudo gcc -o main main.c -lwiringPi -lrt -lpthread -lm -lrt -lcrypt -lbluetooth*.

Ensuite, pour lancer le programme, utilisez la commande : sudo *./main*.

MoveAndGo - GPL-3.0 License