Paul KLAK

François MONTEIL

Compte rendu

–

Raspberry

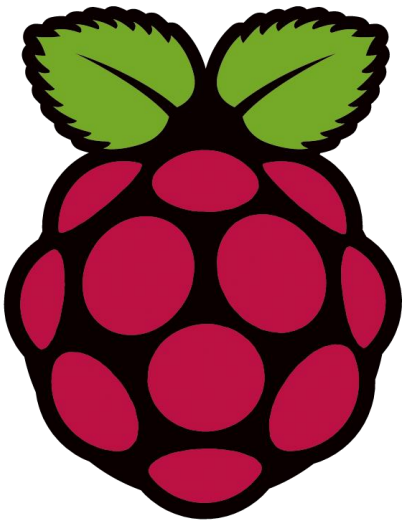

Table des matières

Introduction

I – Préparation de la micro carte SD

Dans un premier temps nous devons préparer la micro carte SD nous allons alors la partitionner en deux :

- Une partie en Fat32 dans laquelle nous retrouverons la configuration du RPI, le noyau…
  - Partitionner Fdisk /dev/sdb
  - Rendre cette partition bootable (t)
  - 128 Mo
- Une partie Ext4 dans laquelle nous retrouverons le système de fichier.

Pour la suite nous avons récupérer la partition boot d'un Raspbian, nous nous occuperont d'installer un noyau linux sur la partition root.

Il nous faut ensuite modifier le fichier cmdline.txt et remplacer le champ : PARTUUID par /dev/mmcdlk0p2

II – Installation de busybox

Pour pouvoir installer busybox sur la RPI il faut utiliser un compilateur croisé car la Raspberry Pi possède un processeur ARM alors que les ordinateurs possèdent des processeurs x86.

Pour cela nous avons réalisé un script de compilation :

```
#!/bin/sh

set –e


CROSS_COMPILE="/home/francois/Desktop/RPI/tools-master/arm-bcm2708/gcc-linaro-arm-linux-gnueabihf-raspbian-x64/bin/arm-linux-gnueabihf-"


cd busybox-1.29.3

make CROSS_COMPILE="$CROSS_COMPILE" CONFIG_PREFIX="/mnt/root" install
```

Une fois busybox installer, il faut regarder ses dépendances, pour cela nous devons utiliser la commande ldd, mais celle fournit par le compilateur croisé :

```
./arm-linux-gnueabihf-ldd --root /mnt/root/bin /mnt/root/bin/busybox
```

La commande nous renseigne alors sur les librairies manquantes, librairie que nous devons récupérer dans un sous dossier du compilateur croisé et les copier dans le dossier lib à la racine de la partition root.

III – Installation librairie wiringPi

La librairie wiringPi met à notre disposition un programme nommé gpio qui permet de contrôler et voir le statut des port GPIO de la Raspberry.

Pour l'installer dans notre busybox depuis notre environnement de travail sur ordinateur, il va falloir utilisé de nouveau le compilateur croisé.

Tout d'abord nous téléchargeons le dossier wiringPi qui est à notre disposition sur http://wiringpi.com/ .

```
$ git clone git://git.drogon.net/wiringPi
```

Pour obtenir les librairies wiringPi (.so) ainsi que l'exécutable gpio, il va nous falloir compiler les éléments dans le bon ordre.

Compilation wiringPI

Tout d'abord nous compilons les librairies wiringPI, pour cela nous allons modifier le Makefile afin qu'il utilise notre compilateur croisé :

Tout d'abord on redirige l'installation dans un dossier à la base du bureau, on aurait très bien pu la rediriger directement sur la partition root de la raspberry :

```
DESTDIR?=/home/francois/Desktop
PREFIX?=/wiringPi
```

Ensuite on lui fournit le chemin vers le compilateur croisé, pour enfin lui définir quel gcc utilisé :

```
PATH_CROISE = /home/francois/Desktop/RPI/tools-master/arm-bcm2708/gcc-linaro-
arm-linux-gnueabihf-raspbian-x64/bin
GCC_CROISE = $(PATH_CROISE)/arm-linux-gnueabihf-gcc
CC      = $(GCC_CROISE)
```

Une fois ces modifications faites, on peut compiler :

```
$ make all
$ make install
```

Une fois ces commandes effectuées, on retrouve dans le dossier wiringPi sur le bureau deux dossier lib et include dans lesquelles nous retrouverons nos librairies compilées avec le compilateur croisé.

Compilation devLib

Nous pouvons passer à la compilation des devLib, cette fois si en plus de lui indiquer ou s'installer et quel compilateur choisir, il va falloir lui indiquer où allez chercher les includes :

```
DESTDIR?=/home/francois/Desktop
PREFIX?=/wiringPi

INCLUDE = -I$(DESTDIR)$(PREFIX)/include
```

Une fois chose fait on compile de la même manière que wiringPi et on obtient de nouveau fichier dans les dossier include et lib.

Compilation de gpio

Pour compiler le programme gpio, il faut comme précédemment indiquer :

1. Le dossier où l'installer
2. Le chemin du compilateur croisé
3. Quel fichier include utiliser

Et il faut de plus lui indiquer le chemin vers les librairies :

```
DESTDIR?=/home/francois/Desktop
PREFIX?=/wiringPi

LDFLAGS = -L$(DESTDIR)$(PREFIX)/lib
```

On compile et dans notre dossier wiringPi nous retrouvons un dossier bin avec notre programme gpio.

On copie include, lib et gpio dans la partition root, on remet la micro SD dans la raspberry et on la démarre.

Utilisation de gpio sur la RPI

Une fois la raspberry lancer on utilise une commande tel que gpio readall.
La raspberry nous renvois un message d'erreur, on répertorie alors le dossier lib :

```
# ls -al
```

On remarque alors que libwiringPi.so et libwiringPiDev.so sont des liens symboliques qui pointe vers des fichiers se trouvant dans un répertoire de l'ordinateur de préparation.

On supprime alors ces fichiers et rééditons les liens :

```
# rm libwiringPi.so
# rm libwiringPiDev.so
# ln -s /lib/libwiringPi.so.2.46 libwinringPi.so
# ln -s /lib/libwiringPiDev.so.2.46 libwinringPiDev.so
```

On réutilise la commande et on obtient alors :

```
/ # gpio readall
 +-----+-----+----------+------+---+---Pi 3---+---+------+----------+-----+-----+
 | BCM | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | BCM |
 +-----+-----+----------+------+---+----++----+---+------+----------+-----+-----+
 |     |     |    3.3v  |      |   |  1 || 2  |   |      | 5v       |     |     |
 |   2 |   8 |   SDA.1  |   IN | 1 |  3 || 4  |   |      | 5v       |     |     |
 |   3 |   9 |   SCL.1  |   IN | 1 |  5 || 6  |   |      | 0v       |     |     |
 |   4 |   7 | GPIO. 7  |   IN | 1 |  7 || 8  | 0 | IN   | TxD      | 15  | 14  |
 |     |     |      0v  |      |   |  9 || 10 | 1 | IN   | RxD      | 16  | 15  |
 |  17 |   0 | GPIO. 0  |   IN | 0 | 11 || 12 | 0 | IN   | GPIO. 1  | 1   | 18  |
 |  27 |   2 | GPIO. 2  |   IN | 0 | 13 || 14 |   |      | 0v       |     |     |
 |  22 |   3 | GPIO. 3  |   IN | 0 | 15 || 16 | 0 | IN   | GPIO. 4  | 4   | 23  |
 |     |     |    3.3v  |      |   | 17 || 18 | 0 | IN   | GPIO. 5  | 5   | 24  |
 |  10 |  12 |    MOSI  |   IN | 0 | 19 || 20 |   |      | 0v       |     |     |
 |   9 |  13 |    MISO  |   IN | 0 | 21 || 22 | 0 | IN   | GPIO. 6  | 6   | 25  |
 |  11 |  14 |    SCLK  |   IN | 0 | 23 || 24 | 1 | IN   | CE0      | 10  | 8   |
 |     |     |      0v  |      |   | 25 || 26 | 1 | IN   | CE1      | 11  | 7   |
 |   0 |  30 |   SDA.0  |   IN | 1 | 27 || 28 | 1 | IN   | SCL.0    | 31  | 1   |
 |   5 |  21 | GPIO.21  |   IN | 1 | 29 || 30 |   |      | 0v       |     |     |
 |   6 |  22 | GPIO.22  |   IN | 1 | 31 || 32 | 0 | IN   | GPIO.26  | 26  | 12  |
 |  13 |  23 | GPIO.23  |   IN | 0 | 33 || 34 |   |      | 0v       |     |     |
 |  19 |  24 | GPIO.24  |   IN | 0 | 35 || 36 | 0 | IN   | GPIO.27  | 27  | 16  |
 |  26 |  25 | GPIO.25  |   IN | 0 | 37 || 38 | 0 | IN   | GPIO.28  | 28  | 20  |
 |     |     |      0v  |      |   | 39 || 40 | 0 | IN   | GPIO.29  | 29  | 21  |
 +-----+-----+----------+------+---+----++----+---+------+----------+-----+-----+
 | BCM | wPi |   Name   | Mode | V | Physical | V | Mode | Name     | wPi | BCM |
 +-----+-----+----------+------+---+---Pi 3---+---+------+----------+-----+-----+
/ # _
```

Le programme gpio fonctionne.