Documentation

(courte)

Électronique

Millenium

1. Rasbperry Pi A+ / Fusée

La Raspberry Pi est un micro-ordinateur fonctionnant sous Linux. Elle dispose de nombreux ports d'entrée/sortie et permet l'utilisation d'une caméra. Ses caractéristiques sont les suivantes :

Caractéristiques techniques	
Soc/CPU GPU	Broadcom BCM2835 / 700MHz ARM Broadcom VideoCore IV
RAM	256Mo (partagé avec GPU)
Ports USB	1 (2.0)
Sorties vidéos	HDMI,Composite
Mémoire	Carte µSD
I/O	17 GPIO (compatible I²C, SPI, UART,CSI,...) +5V ou +3,3V
Consommation	200mA (max. 230mA)
Alimentation	5V via µ-USB ou GPIO
Dimensions	65mm × 53,98mm × 17mm
Masse	23g

1.1. Démarrages des programmes de vols

En l'absence d'écran, elle peut être contrôlée à distance à l'aide d'une connexion SSH. Cependant, il sera impossible de la commander via SSH le jour du lancement.

On utilise donc le programme utilitaire `cron` qui permet de planifier des taches définies dans la `crontable` . On modifie la `crontable` à l'aide de la commande `crontab -e`.

Un exemple d'entrée à ajouter à la *crontable* est :

```
@reboot ./minuterie
```

Ainsi, le programme *minuterie* sera lancé à chaque démarrage de la Raspberry Pi.

1.2. Entrées / Sorties

Les pins d'entrée/sortie de la Rpi sont numérotés de plusieurs manières : suivant la numérotation BCM (ex : GPIO 18, GPIO 24) et suivant la numérotation physique/géographique (ex : pin 4, pin 1). Il faudra donc faire attention à la numérotation utilisée.

1.3. Détection du décollage

La détection du décollage se fait à l'aide du circuit suivant :

Schéma ici →

Lorsque la prise Jack est enfoncée (circuit fermé), la tension mesurée est la tension de la masse. Cependant, lorsque la prise Jack sort, le circuit passe à l'état ouvert et la tension mesurée devient +5V. Il suffit donc de mesurer la tension sur le GPIO23/pin16 pour savoir si la fusée à décollé ou non.

La LED sert à indiquer à l'utilisateur si le circuit est bien fermé (et donc si la prise Jack est bien branchée). La résistance sert à protéger la LED et à éviter le court-circuit si la LED est absente.

1.4. Servomoteur

L'éjection de la coiffe se fait à l'aide d'un servomoteur. Celui-ci est piloté à l'aide d'un signal PWM de fréquence 50Hz (pour avoir une impulsion toutes les 20ms). Une impulsion de 1ms met le servo en positon neutre et une impulsion de 2ms le fait tourner jusqu'à son maximum.
Il suffira de changer le rapport cyclique du signal PWM pour changer la positon du servo. Le rapport cyclique correspondant à 1ms est ainsi R_{1ms}=5% et celui correspondant à 2ms est R_{2ms} = 10%. Cela donne, pour des entiers codés sur 8bits (0%=0,100%=255), les valeurs de R suivantes : R_{1ms} = 13 et R_{2ms} = 26.

Rappel : Rapport cyclique = Durée impulsion / Période

1.5. Minuterie

La minuterie est réalisée à l'aide d'un programme en C utilisant une bibliothèque de gestion de GPIO pour Raspberry Pi : *pigpio.*

Le programme est donné en annexe et fonctionne de la manière suivante :

-On initialise d'abord la bibliothèque *pigpio.*
-On définit le mode de fonctionnement des pins (numérotation BCM, ie. GPIO 18, GPIO 23)
-On positionne ensuite le servo de manière à bloquer le mécanisme d'éjection
-On lit la tension sur le GPIO 23 jusqu'à ce que la fusée décolle
-On attend 5,5s
-On fait bouger le servo pour que l'éjection de la coiffe ait lieu.

Le programme est ensuite compilé à l'aide *gcc* en n'oubliant pas d'inclure les bibliothèques . Puis on s'assure que tous les utilisateurs peuvent utiliser le programme (à l'aide de `ls -l` et de `chmod`). Il faut alors ajouter le programme à la `crontab` (en première position) à l'aide de `sudo crontab -e` puis en ajoutant `@reboot ./[dossier]/minuterie`.

Rq : La `crontab` utilisée n'est pas celle de l'utilisateur `pi` mais celle de `root` pour éviter tout problème de permission.

1.6. Caméra

La caméra utilisée est une PiCam v1. Celle-ci est spécialement conçue pour la Rpi et se connecte via un connecteur propriétaire : le CSI.

Voici quelques caractéristiques techniques :

Caractéristiques	
Dimensions	25 x 24 x 9 mm
Masse	3g
Résolution max (capture de photo)	2592 x 1944 pixels

Pour plus d'info, voir :
https://www.raspberrypi.org/documentation/hardware/camera/README.md

La caméra peut enregistrer des vidéos à l'aide de la commande suivante :
`raspivid -o [nom de video].h264 -t [durée]`

→ Il faudra donc ajouter cette commande dans la `crontab` pour que la carte commence l'enregistrement vidéo dès sa mise en route terminée (la vidéo contiendra des minutes d'enregistrement inutiles : installation sur le pas de tirs, attente du décollage,... mais ce n'est pas grave).

1.7. Capteurs

Les capteurs utilisés par la Rpi sont :

-un accéléromètre LIS331HH fabriqué par Sparkfun/STMicroelectronics. Ce capteur a été choisi car sa plage de mesure, allant de -24g à +24g, devrait permettre de mesurer l'accélération de la fusée au décollage.

-un altimètre/baromètre BMP280 fabriqué par Adafruit/Bosch.

Les capteurs communiquent tous les deux avec la Rpi à l'aide du protocole I²C et sont branchés comme indiqué sur le schéma.

Les programmes Python 3 permettant de recevoir les données sont fournis en annexe. (Rq : Les programmes fournis ne font qu'afficher les données à l'écran, il faudra changer les `print(...)` par des `write(...)`).

a) LIS331HH

Ce capteur fonctionne simplement : il suffit de le régler et de lire les données. Il serait toutefois intéressant de connaître à l'avance les fréquences de vibration de la fusée lors du vol (ex : vibrations dues au moteur) pour pouvoir régler correctement le filtre interne de l'accéléromètre (@Anthony :cf le cours de TdS, Shannon, repliement et tout le reste)

b) BMP280

Là aussi il suffit de régler le baromètre et de lire les registres du baromètre pour pouvoir obtenir des données. Cependant, il faut utiliser une formule de correction des données pour pouvoir obtenir des mesures cohérentes. Il faut donc penser à lire les valeurs des registres de compensation avant d'utiliser pleinement le capteur (voir programme).

→ Les programmes pourront être réécrits en C si nécessaires.

→ Il faut là aussi ajouter les programmes à la `crontable`.

c) Schéma électrique

Voir Annexes.

2. Arduino Nano / Coiffe

La carte Arduino Nano est une carte de la famille Arduino. Ses caractéristiques principales sont les suivantes :

Caractéristiques	
Microcontrôleur	Atmel ATmega328
Mémoire (pour les programme)	16Ko
Tension d'alimentation	7-12V
Tension de fonctionnement (interner et I/O)	5V
Pins	8 Analogues et 14 Digitaux
Courant max : par pin / total	40mA /
Dimensions	45 x 18 mm
Masse	5g

2.1. Alimentation

La carte Arduino Nano est branchée à son alimentation via les pins Vin et GND. LA tension d'alimentation doit être comprise entre 7 et 12V.

2.2. Capteurs

Les capteurs utilisés sont regroupés dans le module 10-DOF IMU de chez Adafruit (IMU = *Inertial navigation system*). Ce module contient : le baromètre BMP180, l'accéléromètre / boussole LSM303DLHC et le gyroscope L3DG20H.

Il est intéressant de noter que l'accéléromètre a une plage de mesure allant de -16g à +16g (au maximum) et devrait donc être saturé au moment du décollage. C'est pourquoi un deuxième accéléromètre, le LIS331HH, est utilisé.

Le module 10-DOF IMU est exploité à l'aide d'un programme pour carte Arduino (codé en un dérivé du C ou C++ ?) et de la bibliothèque prévue à cet effet par Adafruit.

2.3. Lecteur de carte uSD

La carte Arduino Nano ne dispose pas d'une mémoire suffisante pour stocker toutes les données enregistrée pendant le vol. Nous avons donc besoin d'utiliser un stockage externe.

Le support de stockage utilisé ici est une carte uSD. Celle-ci est montée dans un adaptateur pour une utilisation et un branchement plus facile. Les données sont ensuite enregistrées sur la carte SD à l'aide des programmes disponibles dans l'IDE Arduino.

2.4. Schéma électrique

Voir annexe