

Tutoriel de fabrication du τετραφάρμακος

Clément Javerzac-Galy

Denis Savoie

Zubair Iftikhar

<http://cansat2012.supop.fr>

30 septembre 2012

Résumé

Le projet τετραφάρμακος¹ est un projet de Cansat réalisé par l'équipe des *Proton-thérapeutes* de l'Institut d'Optique Graduate School afin de participer au *C'Space*, une compétition internationale co-organisée par Planète-Sciences et le CNES.

Ce prototype de sonde spatiale embarque un module de mesure de l'indice de végétation fait-maison. Il enregistre en outre certains paramètres de vol à l'aide de divers capteurs (météorologiques et positionnels).

Nous espérons que toutes ces données mesurées permettront de caractériser les chances de présence de vie sur une exo-planète semblable à la Terre.

Dans ce tutoriel, vous apprendrez à utiliser les différents composants que nous avons embarqués dans notre Cansat. Vous verrez comment tester le matériel, enregistrer les données mesurées sur une carte μSD et transférer des données à votre PC via une liaison sans-fil.

1 Matériel nécessaire

Si vous souhaitez réaliser une copie exacte de notre projet (dont le coût total est d'environ 368€), il vous faudra :

1. Contrôle et calculs :

- 2 x Micro-controlleurs Arduino Mini, 2 x 17€
- 1 adaptateur Arduino-Mini – USB, 15€

2. Alimentation²

- 1 batterie 9V (NiMH), 13€
- 1 circuit d'adaptation de tension délivrant du 3,3V et du 5V, 5€

3. Stockage d'information

- 2 modules pour carte μSD, 2 x 13€
- 2 cartes μSD, 2 x 7€

4. Transfert de données sans-fil

- 2 modules XBee Pro, 2 x 36€
- 1 dongle-USB XBee, 21€

5. Capteurs

- 1 capteur d'humidité et température [RHT03], 15€
- 1 capteur de pression et température [BMP085], 18€
- 1 accéléromètre [ADXL345], 22€

1. prononcez *tetrapharmakos*

2. Pour commencer, vous pouvez utiliser l'alimentation de la liaison USB entre votre Arduino et votre ordinateur.

- 1 module GPS [EM-406A], 29€
- 2 caméras Jpeg [LinkSprite Jpeg TTL], 2 x 42€

Nous utilisons des composants en double car nous avons besoin de prendre deux photographies simultanées pour réaliser notre mesure de l'indice de végétation. Vous préferez sans doute une Arduino Nano à une Mini, puisqu'elle est plus simple à programmer et à connecter à un ordinateur. Si vous voulez réduire les coûts et les contraintes techniques, voici la liste de composants à utiliser :

1. Contrôle et calculs :

- 1 Micro-controlleurs Arduino Nano, 29€

2. Alimentation

- 1 batterie 9V (NiMH), 13€
- 1 circuit d'adaptation de tension délivrant du 3,3V et du 5V, 5€

3. Stockage d'information

- 1 modules pour carte μSD, 13€
- 1 cartes μSD, 7€

4. Transfert de données sans-fil

- 2 modules XBee, 2 x 26€
- 1 dongle-USB XBee, 21€

5. Capteurs

- 1 capteur d'humidité et température [RHT03], 15€
- 1 capteur de pression et température [BMP085], 18€
- 1 accéléromètre [ADXL345], 22€
- 1 module GPS [EM-406A], 29€
- 1 caméras Jpeg [LinkSprite Jpeg TTL], 42€

Ce qui revient alors à 266€ environ. Vous pouvez très bien adapter ce tutoriel à vos envies et choisir de faire une simple station météo (100€), ou un système de prise de photographies géolocalisées (138€).

2 Assemblage et vérification du matériel

Nous allons commencer par vérifier l'état de marche de chacun des composants. N'utilisez pas de pile au départ, l'alimentation se fera grâce à l'ordinateur. Par exemple si vous avez une Arduino Nano, il suffit de la connecter à votre ordinateur via un câble USB (si vous utilisez une Arduino Mini, vous devez utiliser l'adaptateur Arduino FTDI USB-Série pour alimenter et programmer votre système).

2.1 Le micro-contrôleur

Pour savoir comment connecter votre Arduino à votre ordinateur pour la programmer, vous pouvez vous rendre sur le site officiel du fabricant :

- Arduino Mini : <http://arduino.cc/en/Guide/ArduinoMini>
- Arduino Nano : <http://arduino.cc/en/Guide/ArduinoNano>

Attention: Veillez à NE PAS brancher votre Arduino en inversant les polarisations + et -.

Dans tous les cas, vous devez télécharger le logiciel de programmation **Arduino** sur <http://arduino.cc/en/Main/Software>. Utilisez le programme d'exemple 'Blink' pour vérifier l'état de marche de votre micro-contrôleur.

2.2 Le stockage sur carte µSD

La plupart des applications nécessitent un enregistrement sur carte SD. Le prix de ces cartes mémoire Flash a beaucoup diminué, et on peut aujourd'hui facilement trouver une carte µSD de 2Go pour moins d'une dizaine d'euros. Il est recommandé d'utiliser des cartes de marque (j'avais une carte générique qui n'était pas reconnue, bien qu'elle soit bien formattée).

La figure 1 montre comment connecter le module de carte µSD à une Arduino Mini³.

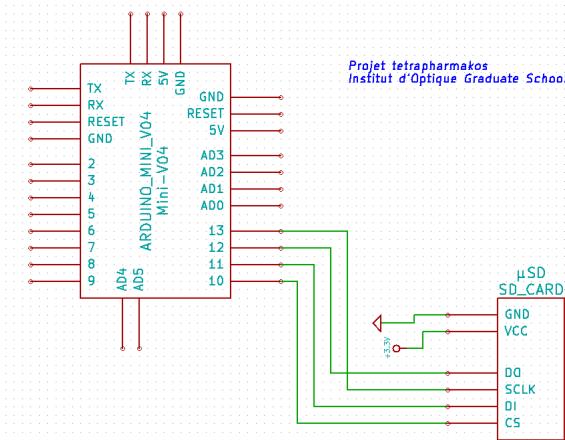


FIGURE 1 – Branchement d'une carte SD à une Arduino

Il faut alors utiliser le programme d'exemple 'CardInfo' du logiciel **Arduino** pour vérifier l'état de votre carte SD et de votre branchement.

2.3 Les capteurs

Notre Cansat devait remplir deux missions (sondage atmosphérique et mesure de l'indice de chlorophylle). Pour la première, nous avions besoin d'un capteur d'humidité et d'un capteur de pression. Pour la seconde, nous avions besoin de

3. Il faut savoir que les cartes SD doivent être alimentées en 3,3V, et que l'Arduino Mini fonctionne en 5V.

deux caméra. L'accéléromètre et le module GPS étaient utilisés en vue de préparer une troisième mission : le atterrissage maîtrisé sur une cible.

2.3.1 Capteur d'humidité

Le capteur d'humidité que nous avons utilisé est le **RHT03**. Comme le reste des capteurs que nous avons utilisés, il est très simple d'emploi. La figure 2 montre comment le brancher⁴.

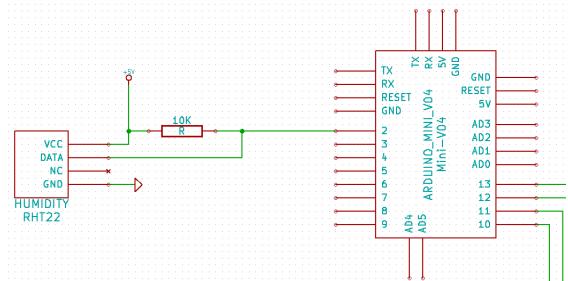


FIGURE 2 – Branchement du RHT03 à une Arduino

Il faut ensuite programmer votre micro-contrôleur pour envoyer les données du capteur sur le port Série de votre Arduino. Pour cela, téléchargez le programme 'DHTtester' à l'adresse suivante : <https://github.com/thriller91/Cansat-SupOp/tree/master/sources/Arduino/DHT22/DHTtester>. Il faut télécharger les trois fichiers (DHTtester.pde, DHT.cpp et DHT.h), et les placer dans un même dossier appelé **DHTtester**.

2.3.2 Capteur de pression

Notre capteur de pression est le **BMP085**. Il se connecte en I²C . Comme le montre la figure 3, il faut donc utiliser les broches analogiques n° 4 et 5 de l'Arduino.

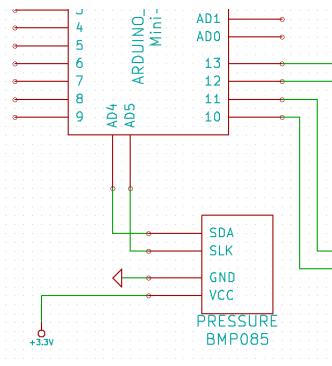


FIGURE 3 – Branchement du BMP085 à une Arduino

Le programme à télécharger pour essayer ce capteur est disponible à l'adresse suivante : <https://github.com/thriller91/Cansat-SupOp/tree/master/sources/Arduino/BMP085/BMP085tester>.

4. Il faut utiliser une résistance de tirage de 10kΩ entre la broche n° 2 et l'alimentation +5V.

2.3.3 Accéléromètre

Nous avons utilisé un accéléromètre numérique, l'*ADXL345*. Ce composant peut se connecter en I²C ou en SPI (déjà utilisé par la carte SD). On peut connecter l'accéléromètre et le capteur de pression sur le même bus I²C, mais comme nous avions 2 Arduino, nous avons utilisé une pour chaque capteur. Si vous voulez éviter d'utiliser une autre Arduino et que vous ne voulez pas placer ces deux composants sur le même bus, vous pouvez choisir d'utiliser un accéléromètre analogique (l'*ADXL335* par exemple).

La figure 4 montre comment brancher l'*ADXL345* sur une Arduino en I²C.

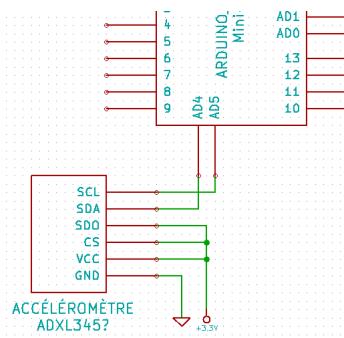


FIGURE 4 – Branchement de l'ADXL345 en I²C

Le programme de pilotage de cet accéléromètre est disponible sur : <https://github.com/thriller91/Cansat-SupOp/tree/master/sources/Arduino/ADXL345/ADXL345tester>. Si vous utilisez l'accéléromètre analogique, il suffit de le connecter comme indiqué dans la documentation technique et d'utiliser le programme fourni dans les exemples du logiciel *Arduino* 'ADXL3xx' (dans le dossier Sensors).

L'utilisation d'un accéléromètre seul (sans gyroscope) n'est pas idéale. Nous pensions pouvoir négliger les rotations du τετραφάρμακος pendant sa chute, mais nous avons rencontré un autre problème. Étant donné que la structure mécanique de notre Cansat n'était pas très rigide, elle a finie par s'incliner au fil du temps, et nous ne pouvions malheureusement plus compter sur notre capteur de pression pour déterminer l'angle de la verticale⁵ (vous pouvez retrouver nos résultats dans notre présentation <http://cansat2012.supop.fr/presentation.pdf>). Si vous voulez enregistrer le véritablement mouvement de votre système, visitez la page suivante pour choisir votre centrale inertuelle : https://www.sparkfun.com/pages/acel_gyro_guide.

2.3.4 Module GPS

Le module GPS *EM-406A* n'est pas une simple antenne GPS. Ce composant est totalement autonome, il suffit de l'alimenter pour qu'il envoie ses trames GPS (NMEA) — après s'être synchronisé avec les satellites bien évidemment. Il en-

voit ses données via un port série à 4800 bauds. Il suffit donc de le brancher comme sur le schéma de la figure 5.

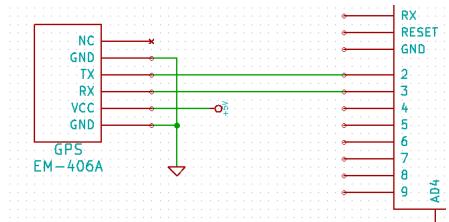


FIGURE 5 – Branchement de l'EM-406A

Des exemples de programmes d'utilisation sont disponibles dans la bibliothèque *TinyGPS*, que vous pouvez télécharger librement à l'adresse suivante : <http://arduiniana.org/TinyGPS/TinyGPS12.zip>⁶

Ces programmes utilisent la bibliothèque *SoftwareSerial* d'*Arduino*. Elle permet de simuler un port série {RX,TX} sur des broches de la carte Arduino. Il suffit de créer un objet *SoftwareSerial mySerial(2, 3); //RX, TX*, pour que la broche n°2 (resp. n°3) de votre Arduino soit l'entrée (resp. la sortie) de votre port série.

2.3.5 Caméra Série Jpeg

Notre mission principale était la mesure de l'indice de chlorophylle, cette mesure passe par la capture deux photographies d'une même scène, chacune derrière un filtre optique. Notre problème était qu'il est difficile de trouver une caméra facile à interfaçer avec notre micro-contrôleur. L'idéal est alors de choir une caméra avec une sortie série. Le prix de ce type de caméra peut sembler élevé — environ le quadruple — à côté d'une simple webcam, mais il est quasiment impossible de démonter le capteur d'une webcam pour s'en autrement qu'avec les pilotes logiciel fourni par le constructeur (et qui ne sont pas destinés aux micro-contrôleurs).

Nous avions commencé par utiliser une caméra Jpeg de 4DSystems, la *µCAM-TTL*. Nous avions réussi à prendre des photographies en la pilotant sur ordinateur avec notre propre programme écrit en C. Mais la traduction de ce code en programme pour *Arduino*⁷ ne permettait malheureusement pas de prendre des images avec le micro-contrôleur — bien que nous arrivions à savoir que la caméra était bien synchronisée avec ce dernier.

Il fallait alors changer de caméra, nous avons donc choisi la caméra Jpeg de LinkSprite, la *LinkSprite JPEG Color Camera TTL Interface*. Nous avons rapidement maîtrisé cette caméra, et nous l'avons connectée au port série natif de notre micro-contrôleur. Le branchement que nous avons fait est expliqué sur la figure 6.

6. Dézippez ce fichier dans le dossier *libraries* d'*Arduino*, puisque ses exemples commencent par `#include <TinyGPS.h>`. Redémarrer ensuite *Arduino* pour pouvoir voir les programmes que vous venez de télécharger dans les exemples du logiciel.

7. Les programmes sont disponibles dans notre répertoire GitHub, dans la branche *µCam-Z* sur <https://github.com/thriller91/Cansat-SupOp>.

5. En utilisant une formule du nivellement barométrique

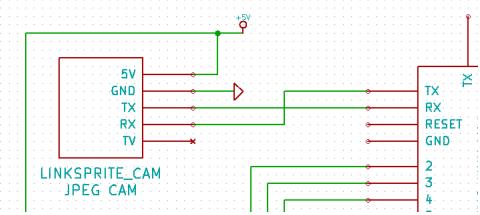


FIGURE 6 – Branchement de la caméra Jpeg LinkSprite

Attention: Lorsque la caméra est branchée de cette façon, la carte Arduino ne peut pas être programmée, il faut débrancher la caméra pour pouvoir programmer votre micro-contrôleur.

Si vous êtes gêné par ce branchement, il vous est aussi possible de brancher la caméra sur deux autres broches de votre Arduino, et utiliser la bibliothèque SoftwareSerial d'*Arduino*.

Nous utilisons la carte SD pour stocker la photographie, puis nous attendons l'atterrissement du *τετραράμυχος* avant de transférer les informations à une station au sol. Le programme spark (que vous pouvez télécharger à l'adresse : <https://raw.github.com/thriller91/Cansat-SupOp/master/sources/Arduino/LinkSprite/exemples/spark/spark.ino>) est très largement inspiré d'un exemple de code donné par un utilisateur de SparkFun.com, il prend une photographie et l'enregistre sur la carte mémoire. On peut relancer ce programme plusieurs fois, les images ne sont pas écrasées, elles sont renommée IMGxx.JPG sur la carte SD.

Avec ce programme, la capture d'une photo prend environ 5 secondes, et son enregistrement 20 secondes. Vous pouvez changer la résolution et taux de compression de l'image selon la qualité d'image que vous voulez, mais ces changements modifiront le temps d'exécution du programme. Il ne faut évidemment ni débrancher la carte mémoire, ni mettre la caméra ou le micro-contrôleur hors-tension avant la fin du programme ; sinon, vous obtiendrez un fichier image de 0 octet de mémoire sur votre carte SD.

Pour récupérer l'image en utilisant ce programme, il faut glisser votre carte mémoire dans un adaptateur µSD-SD ou votre téléphone portable pour ouvrir directement le fichier image.

2.4 Transfert de données sans-fil

Une liaison sans-fil avec le Cansat était essentielle pour le projet *τετραράμυχος*, nous simulons en effet l'*atterrissement* d'une sonde sur une planète extra-terrestre. Cette sonde doit pouvoir communiquer ses résultats à une station au sol. Nous utilisons pour ce faire un module appellé *XBee*. Ce module est basé un protocol similaire au Wifi. Il existe toute une gamme d'*XBee*, nous avons choisi d'utiliser des *XBee-Pro S1* longue portée. L'une était branchée au micro-contrôleur, et l'autre à la station au sol *via* un *dongle* USB.

Pour commencer, il faut configurer les *XBee* pour qu'elles se connaissent entre elles. Il faut télécharger le logiciel *X-*

CTU sur le site de **Digi**, il est uniquement disponible sous Windows. La documentation fournie permet de configurer rapidement son matériel. Notez bien la vitesse de la communication série (en baud).

Laissez une *XBee* branchée à votre ordinateur, et connectez l'autre à votre micro-contrôleur comme sur la figure 7⁸.

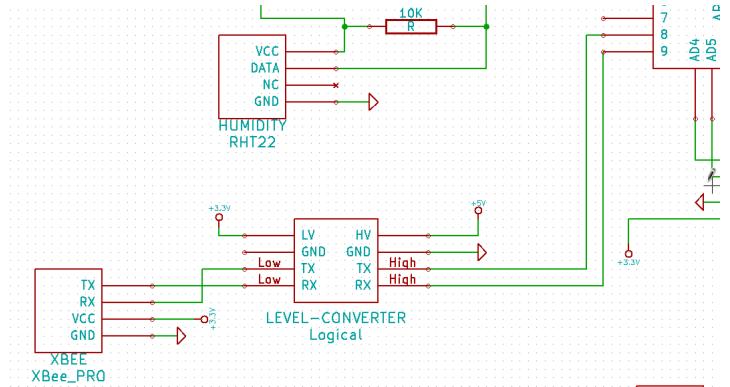


FIGURE 7 – Branchement du module XBee

Attention: La chose à ne surtout pas faire avec votre *XBee* est de l'alimenter en inverse. Vous risquer de la tuer en quelques secondes — croyez nous. Les *XBee* fonctionnent en 3.3V, et comme leur prix est assez élevé, il est sage d'utiliser un convertisseur de niveaux logiques pour l'interfacer avec une *Arduino* à 5V.

Il faut ensuite programmer votre micro-contrôleur pour qu'il envoie et reçoit des messages de la part de la station au sol. Vous pouvez commencer par charger un programme du dossier d'exemples SoftwareSerial du logiciel *Arduino*. Les deux *XBee* agissent comme des fils entre les RX et TX du convertisseur de niveau logique et le *dongle* USB de la station au sol⁹.

Vous savez maintenant communiquer sans-fil avec votre système. Vous pouvez donc recevoir les données qu'il vous envoie. Cependant, vous êtes dépendant du moniteur série d'*Arduino*. Vous aurez donc du mal à recevoir des données binaires — comme des images par exemple. Pour vous affranchir du logiciel *Arduino*, vous pouvez choisir de rediriger le flux d'information vers un fichier, sous Linux, vous pouvez entrer `cat /dev/ttyUSB0 > reception.bin` dans un terminal¹⁰. Ce genre de solution est très simple, mais on ne voit le programme *Arduino* s'exécuter ; en particulier, on ne sait pas quand il se termine. Si vous n'avez pas Linux, ou que vous voulez utiliser une méthode plus évoluée que la redirection de flux, vous pouvez télécharger

8. La distance entre deux broches de la *XBee* est de 2mm et non 2.54mm comme pour la plupart des composants électriques, vous pouvez acheter un *socket* pour éviter de souder directement sur la *XBee*.

9. Nous vous conseillons de commencer par utiliser des fils croisés entre les RX et TX de ces deux cartes, remplacez les fils par des *XBee* lorsque ça fonctionne.

10. Saisir `Ctrl+C` pour arrêter la réception. Il faut remplacer `/dev/ttyUSB0` par le bon port. Vous pouvez aussi utiliser `screen` pour remplacer le moniteur série d'*Arduino*, il faudra alors entrer `screen /dev/ttyUSB0 9600` dans le terminal.

notre programme Python à l'adresse suivante : <https://raw.githubusercontent.com/thriller91/Cansat-SupOp/master/sources/prgm%20sol/communication/pycrop/pycom.py>. Ce programme n'est qu'une ébauche, mais c'est celui que nous avons utilisé le jour de la compétition, et nous avons réussi à récupérer notre image intacte¹¹.

¹¹. Pour l'anecdote, le *τετραφύρμαχος* avait atterri dans un arbre et nous n'avions pu le récupérer qu'à la fin de la journée, mais cela ne nous a pas importuné.