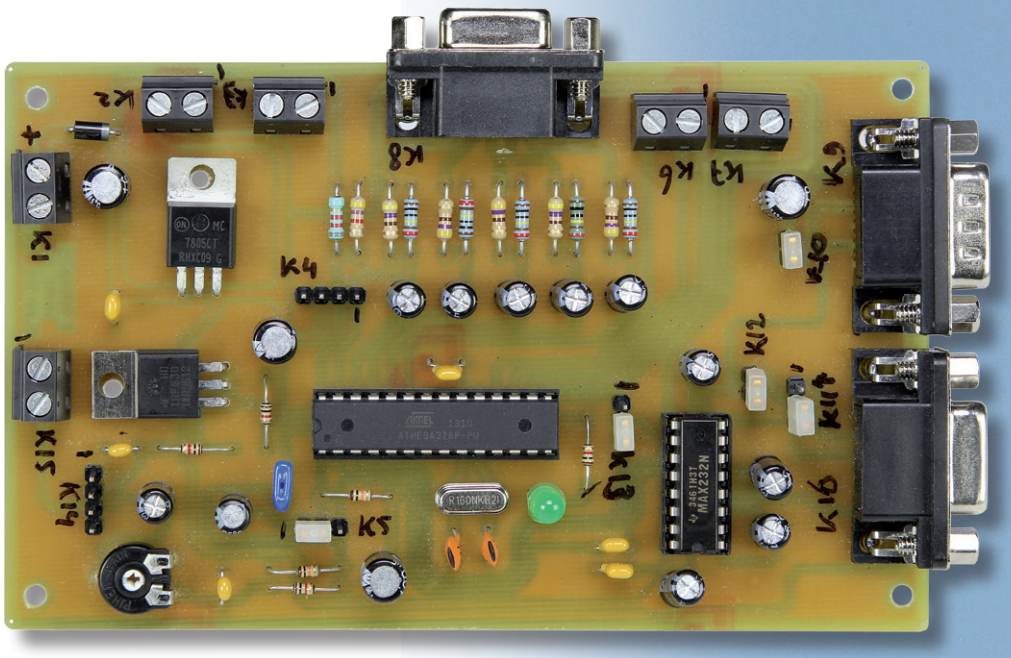
objectif ciel

carte d'acquisition de données multi-usage pour ballon-sonde

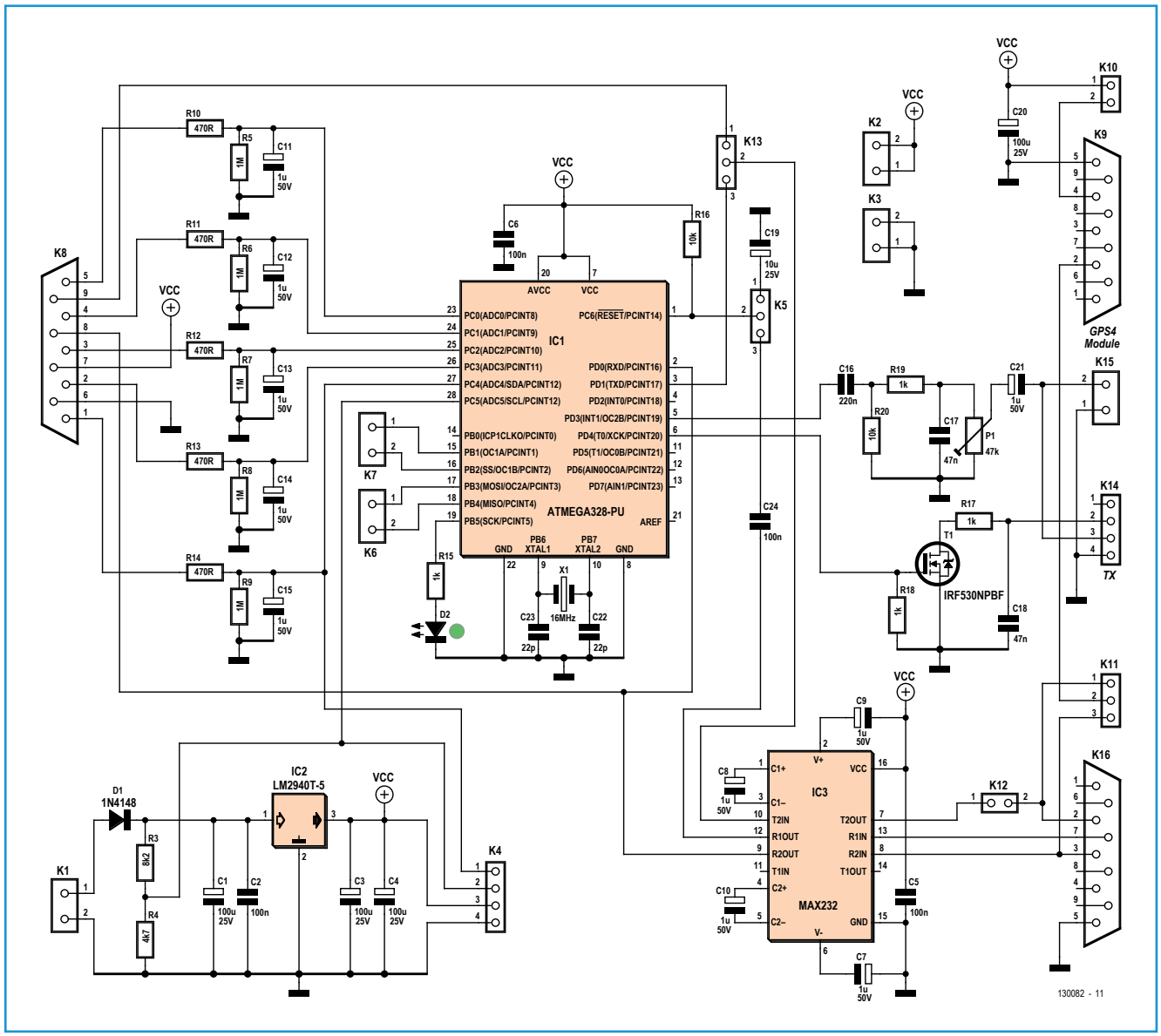
Transmettre au sol les données recueillies par un ballon-sonde n'est pas chose aisée. L'opération nécessite souvent un matériel complexe et coûteux, mais il existe des solutions bon marché, comme la carte présentée ici. Couplée à un émetteur radio standard et un récepteur GPS, elle peut relayer au sol jusqu'à six signaux analogiques et transmettre la position d'un ballon-sonde. Ce projet n'est pas la douce rêverie d'un professeur Nimbus, il a déjà vu les nuages de près par deux fois !

Anthony Le Cren
(France)



Caractéristiques

- programme à code Arduino
- cinq entrées analogiques sur connecteur D-sub à 9 broches (la sixième entrée surveille la tension de la pile)
- un connecteur D-sub à 9 broches pour GPS (NMEA0183A à 4800 bauds)
- port RS-232 pour programmation in-situ du µC (avec chargeur d'amorçage Arduino)
- AFSK et sortie PTT vers l'émetteur
- utilise le protocole APRS/AX25 (paquets radio)
- 4 E/S utilisateur
- alimentation régulée de 5 V
- compatible avec le GPS4 de Byonics
- compatible avec les modèles bi-bande UV-3R et UV-5R de Baofeng



La carte

Le système est construit autour d'un microcontrôleur ATmega328P cadencé à 16 MHz **(fig. 1)**. Le programme exploite le code du projet Arduino à code source ouvert appelé Trackuino [1]. De nombreux capteurs peuvent être reliés à la carte, et la place ne manque pas pour ajouter des extensions.

Commençons par jeter un œil aux différentes parties de cette conception. Le schéma de principe est reproduit sur la figure 1 ; les **figures 2** et **3** montrent le prototype de l'auteur.

Entrées analogiques

Cinq entrées analogiques ont été prévues pour lire les données des capteurs (température, pression, etc.) Elles sont disponibles sur K8, un connecteur D-sub à 9 broches (fig. 2, en haut). Les signaux passent par un filtre passe-bas avant d'être convertis en valeurs numériques par le convertisseur CAN de l'ATmega. Elles sont ensuite accessibles sur le port C. La sixième entrée du CAN sert à mesurer la tension de la pile. Le diviseur R3/R4 abaisse les 9 V au niveau attendu par le µC.

Nous verrons plus bas comment lire les données d'un capteur à l'aide de ces entrées.

Figure 1.
Schéma de la carte d'acquisition de données pour ballon-sonde.

GPS

L'unité GPS est reliée à K9 (fig. 2, connecteur D-sub en haut à droite). Son signal de sortie, disponible sur la broche 2 de K11, est compatible avec les signaux RS-232. Il est donc conditionné par le classique MAX232 (IC3) avant d'arriver au µC. K11 a pour rôle d'aiguiller le signal GPS soit vers le microcontrôleur (position 2-3), soit vers la broche TX de K16 (position 1-2) pour qui souhaite voir les « phrases » NMEA0183A sur un terminal série. Dans ce cas n'oubliez pas de retirer le cavalier K12 afin que la sortie TX2 d'IC3 n'interfère pas avec le signal GPS.
La broche 4 de K9 se charge des 5 V dont a besoin l'unité GPS. La tension d'alimentation peut être coupée par retrait du cavalier sur K10, une option utile lorsqu'on utilise un simulateur comme SIM GPS en lieu et place d'un vrai GPS. Le recours à un simulateur de GPS facilite le développement du code et permet de vérifier le bon fonctionnement du système avant son lancement. La carte et le PC se relient à l'aide d'un câble *null-modem* standard.

Radio VHF

Pour l'interface radio, l'auteur suggère l'émetteur-récepteur bi-bande UV-5R de Baofeng (fig. 3, à gauche), ou encore le UV-3R, deux modèles dotés d'un connecteur à double jack **(fig. 4)**. Seuls trois signaux sont utilisés sur K14 : PTT (*Push To Talk*, broche 2), MIC+ (broche 3) et GND (broche 4).
La transmission des données se fait par modulation par déplacement de fréquence (*Frequency-Shift Keying*, ou FSK) : le signal audio varie entre deux fréquences (1200 Hz et 2200 Hz) pour transmettre une valeur binaire. Comme l'ATmega n'a pas de convertisseur numérique-analogique (CNA), c'est ici un signal modulé en largeur d'impulsion (MLI) qui est utilisé. Ce signal MLI est transformé en signal quasi-sinusoïdal par un filtre passe-bande composé d'un passe-haut du premier ordre et d'un passe-bas en série. Le niveau de sortie peut être réglé avec le potentiomètre P1 avant d'aller alimenter l'entrée MIC de l'émetteur VHF.
Le « bouton » virtuel PTT met la radio en mode Émission. Le µC peut lui aussi « tourner » ce bouton grâce à une entrée présente sur la radio. Le signal PTT sur le port PD4 de l'ATmega commande un transistor MOS

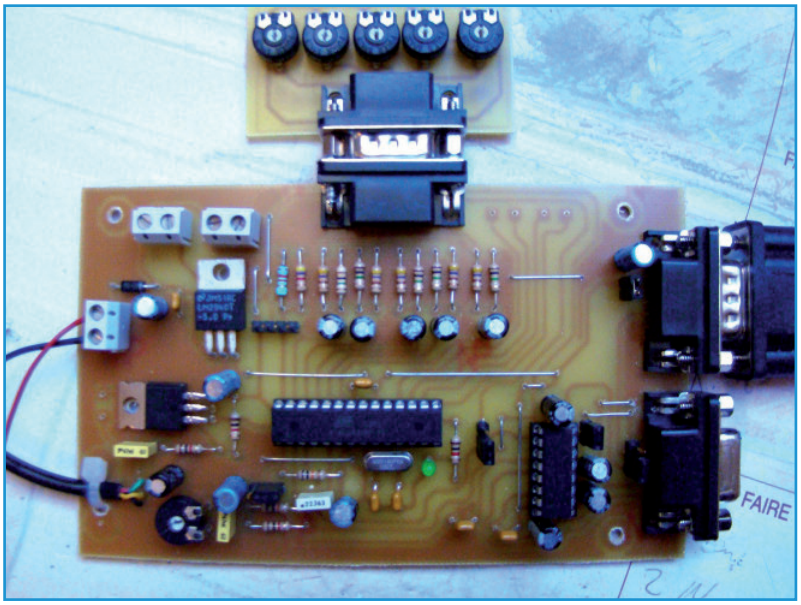


Figure 2.
La carte assemblée par l'auteur.

IRF530. La sortie à drain ouvert ainsi créée devrait être compatible avec la majorité des radios VHF. La LED D2 s'allume lorsque PTT est au niveau haut, et indique donc qu'une émission est en cours.

Alimentation

L'alimentation est traditionnelle, mais notez que pour le régulateur IC2 le classique 7805 n'est pas recommandé ici, car sa tension de déchet est élevée, environ 3 V. Avec la diode D1, il faudrait au moins 9 V sur la ligne de la

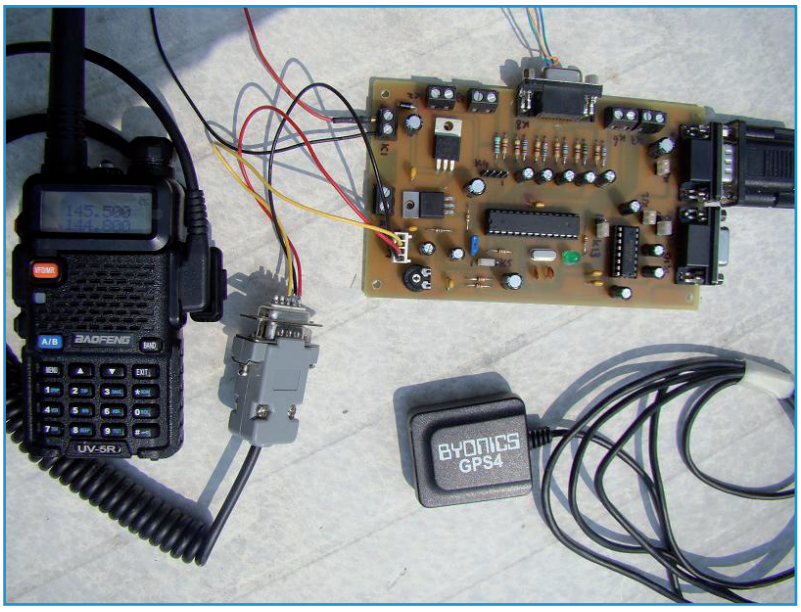


Figure 3.
Configuration matérielle complète, avec la carte, la radio VHF et l'unité GPS.

pile. Un régulateur à faible chute de tension (LDO) comme le LM2940T-5 permet d'alimenter la carte avec une tension aussi basse que 6 V (minimum). 9 – 6 = 3 V ne semble pas énorme, mais la tension de la pile chute de façon sensible avec les températures négatives des hautes altitudes. Pour ne pas rencontrer de problèmes durant le vol, le mieux est d'utiliser une alimentation d'au moins 8 à 9 V, p. ex. avec 6 piles de 1,5 V en série. La consommation totale de la carte, unité GPS comprise, est inférieure à 60 mA.

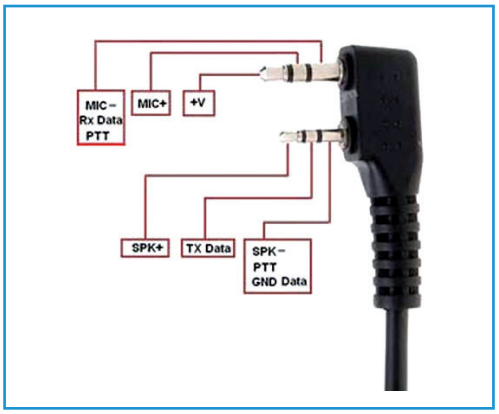


Figure 4.
Connecteur à double jack de la radio VHF UV-5R de chez Baofeng. Seuls 3 des 6 signaux sont utilisés ici.

Programmation de l'ATmega

Grâce au connecteur K16, vous pouvez programmer le microcontrôleur sans le retirer de la carte. Servez-vous d'un câble droit pour relier la carte à un PC équipé d'un vrai port RS-232 (vous en avez gardé un, n'est-ce pas ?). Le MAX232 (IC3) adapte les niveaux du signal RS-232 à ceux utilisés par l'ATmega. Les signaux TX (*transmission* en anglais, émission en français) et RX (réception) sont sur les broches 2 et 3 du µC. L'embase K5 permet de sélectionner la source du signal qui initialise le µC. Avec un cavalier sur 2 et 3, le PC peut forcer une initialisation via la broche RTS de K16, ce qui lancera le chargeur d'amorçage du µC (l'EDI Arduino le fera pour vous).
Le micrologiciel et les autres fichiers sont disponibles sur la page Elektor.LABS associée à ce projet [2]. Pour parer à tout *reset* accidentel, placez le cavalier sur les broches 1 et 2 une fois le µC programmé. L'unité GPS qui partage le même port série doit être déconnectée durant la programmation du µC : débranchez-la, ou retirez le cavalier sur K11. Le **tableau 1** résume les configurations possibles du cavalier.

Extensions

Les connecteurs K2 et K3 peuvent alimenter en 5 V d'autres circuits embarqués dans la capsule ; K6 et K7 peuvent commuter des objets durant le vol. Notez que vous aurez à adapter le micrologiciel à chaque ajout de fonction. K4 est pour ceux qui souhaitent utiliser des capteurs via un bus I^2C connecté aux ports PC4 et PC5 de l'ATmega. Cette communication I^2C vous coûtera toutefois deux entrées analogiques (canaux 4 et 5).
Placer un cavalier sur les broches 1 et 2 de K13 crée une sortie série sur la broche 9 du port analogique K8, et une entrée série est disponible sur la broche 8 de K8. Vous pouvez donc communiquer avec un PC via ce connecteur. Si vous placez le cavalier sur les broches 2 et 3 de K13, le port série sur K8 sera directement relié au µC. Ce port supplémentaire servira éventuellement à relier la ligne série à un périphérique externe, p. ex un deuxième microcontrôleur !

Construction

L'assemblage de la carte ne devrait pas poser de problèmes sérieux. Elektor.LABS [2] en propose une version révisée sous la forme de fichiers Gerber, de PDF, et d'un projet DesignSpark PCB 5. Comme d'habitude, soudez avec soin et assemblez d'abord les petits composants. Tous sont montés sur le côté haut du circuit.

Tableau 1. Configuration des cavaliers

	programmation du µC avec l'EDI Arduino	simulation de lancement	affichage sortie GPS4	lancement du ballon
K5	2-3	1-2	1-2	1-2
K13	2-3	2-3	2-3	2-3
K12	fermé	fermé	ouvert	fermé
K11	ouvert	2-3	1-2	2-3
K10	indifférent	ouvert	fermé	fermé

La carte a été conçue pour recevoir l'unité GPS4 de chez Byonics (fig. 3, en bas à droite). Vous aurez peut-être à adapter les connexions si vous souhaitez utiliser une autre unité. Assurez-vous que le signal de sortie de votre GPS soit compatible avec RS-232 et que le débit soit de 4800 bits/s.
Idem pour l'émetteur VHF, vous pourriez devoir adapter les connexions si vous choisissez un autre modèle que les Baofeng UV-5R ou UV-3R. Il importe d'utiliser un émetteur homologué, qui possède une entrée PTT ainsi qu'une entrée audio (microphone) pour le signal modulé. Ces entrées sont en général des connecteurs jack. Vous pouvez tester la sortie radio de la carte avec un simple talkie-walkie, ou avec un module de 433 MHz doté d'une entrée analogique.

Exemples de capteurs analogiques

Les cinq entrées du connecteur K8 sont analogiques et acceptent des tensions comprises entre 0 et 5 V. Le CAN de l'ATmega est configuré en mode 8 bits et renvoie des valeurs allant de 0 à 255.

Capteur de pression

Pour relier le capteur de pression à la carte, câblez l'alimentation de 5 V, puis connectez la sortie du capteur à l'une des entrées analogiques de K8. Ce capteur est simple à utiliser, mais ne se montre pas assez sensible aux altitudes où la pression atmosphérique est inférieure à 100 millibars : sa sortie reste constante alors qu'elle devrait décroître à mesure que s'élève le ballon.

Capteur de température

Mesurer la température aussi bien à l'intérieur qu'à l'extérieur de la capsule est toujours intéressant **(fig. 5)**. Un diviseur de tension dont l'une des résistances est une thermistance CTN (Coefficient de Température Négatif) fait ici l'affaire. La valeur de la thermistance n'est pas critique, pourvu que la plage de sortie du diviseur couvre la plage de mesure intéressante. J'ai obtenu de bons résultats avec une résistance CTN de 10 kΩ reliée en série à une résistance fixe de 36 kΩ.

Logiciel

Les différences sont nombreuses, mais le logiciel embarqué dans la capsule est semblable au modèle OSI à sept couches. Ces couches sont ici :
Application → APRS → AX25 → modulation AFSK → transmission VHF.
La couche application correspond aux entrées analogiques et à l'acquisition et au décodage des données GPS NMEA0183A. Les données sont placées dans une trame selon le protocole APRS (*Automatic Packet Reporting System*) [3]. Un en-tête conforme au protocole AX25 [4] est ensuite ajouté à la trame APRS. Les bits de la trame de données doivent moduler une porteuse afin que les données soient compatibles avec l'émetteur VHF. C'est le rôle du modulateur AFSK (*Audio Frequency-Shift Keying*). Il transforme la chaîne ASCII AX25 en une suite d'impulsions sonores de fréquences 1200 Hz et 2200 Hz, à un débit de 1200 bits/s. Si la fréquence change entre deux impulsions sonores, un 0 logique est émis ; si elle est identique, c'est un 1 logique qui est transmis. La **figure 6** montre un exemple de transmission.

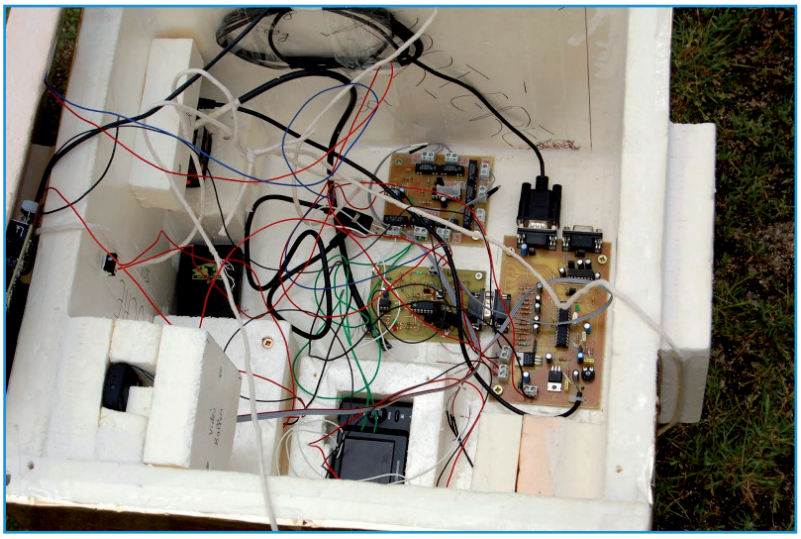


Figure 5.
L'intérieur de la capsule.
La place ne manque pas !

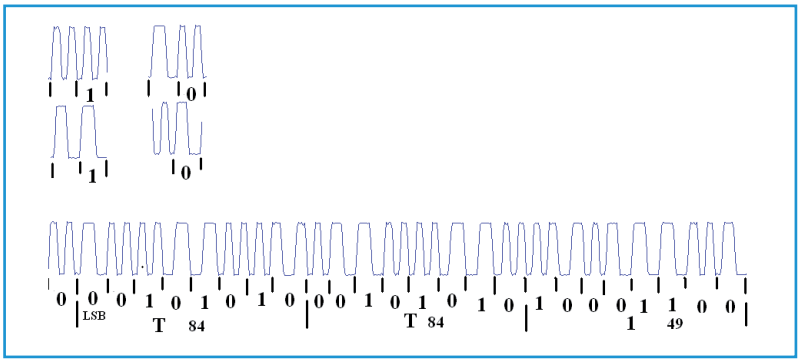


Figure 6.
Exemple d'une transmission AFSK.

Le programme est compatible avec la version actuelle d'Arduino (1.0.5). N'oubliez pas de changer l'indicatif (*call sign*) du ballon dans le fichier *config.h* :

```
#define S_CALLSIGN  "F6XXX"
```

Une façon simple d'obtenir un indicatif valide est de contacter un club radio-amateur situé près de chez soi [5].
Le code qui implante le protocole AX25 se trouve dans le fichier *aprs.cpp*. La fonction ax25_send_string permet de préparer la trame AX25 avant de l'envoyer avec la fonction ax25_flush_frame. Vous pouvez ajouter votre propre code après l'envoi de la trame, p. ex. pour commander des périphériques reliés à K6 et K7.

Configuration de la chaîne de réception

Une trame AX25 envoyée par le ballon peut être reçue avec un récepteur ou scanner VHF-UHF correctement accordé. La trame est décodée par un PC dont l'entrée « ligne » est reliée à la sortie audio du scanner.
Les programmes suivants doivent être installés :

- AGWPE : décodage de trame ;
- AGWTrackerXP : affichage de trame et position approximative du ballon sur une carte ;
- Packet Engine : affichage des données mesurées.

Vous pouvez les télécharger gratuitement depuis [6].
Redémarrez votre PC après avoir installé AGWPE, lancez-le, et configurez un port de communication avec les paramètres de la **figure 7**.

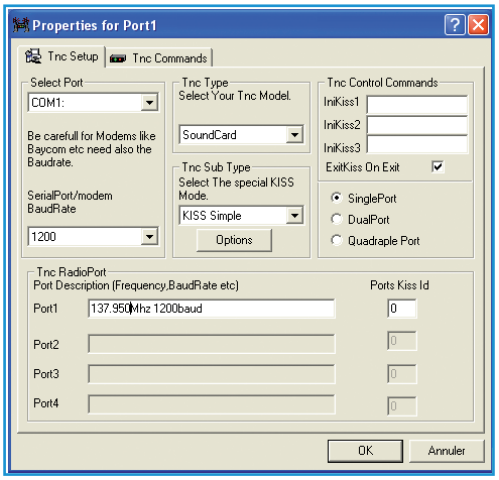


Figure 7.
Configuration du port avec AGWPE.

Installez ensuite *AGWTrackerXP*. Entrez l'indicatif qui désignera la station réceptrice. Une connexion TCP/IP devrait être créée automatiquement entre *AGWPE* et *AGWTrackerXP*.
Le dernier programme à installer, *Packet Engine*, enregistre les données reçues dans un fichier texte. Un tableur permet ensuite de les analyser. La **figure 8** montre le format des chaînes enregistrées. L'en-tête AX25 et la trame APRS (qui comprend l'heure, la latitude, la longitude, etc.) sont faciles à identifier : l'en-tête AX25 se termine par le caractère « | », la trame APRS par « Balloon || ».

Garder le contact

Suivre le ballon durant la majorité du vol ne devrait pas poser trop de problèmes. Il n'est toutefois pas rare de perdre le signal radio lorsque le ballon descend sous 2000 m.
Pour être certain de retrouver la capsule, il est souhaitable d'y ajouter un module de suivi (*tracker*) GSM/GPRS/GPS. Le surpoids ne sera que faible, et s'il est équipé d'une carte SIM ce module pourra envoyer à un téléphone tactile

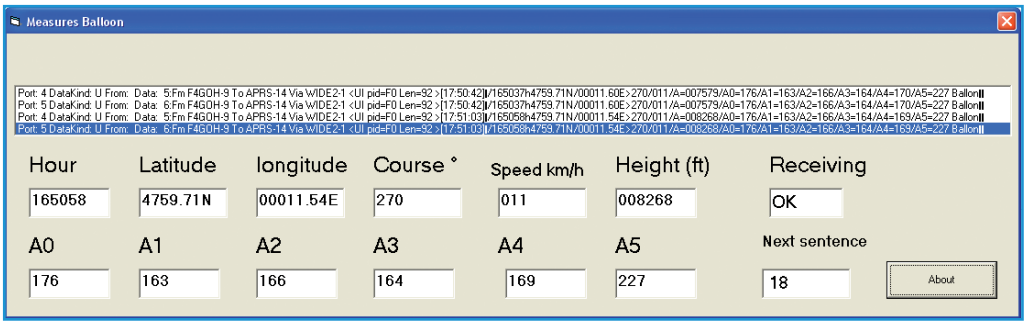


Figure 8.
Données reçues affichées sous *Packet Engine.*

des SMS indiquant la latitude et la longitude. Les antennes des répéteurs cellulaires (GSM) sont légèrement orientées vers le sol, avec pour conséquence que le module de suivi perd son réseau à partir d'environ 700 m. Il se réactive toutefois dès qu'il détecte un réseau, c'est-à-dire au moment où la capsule est proche du sol.

Tout le monde aime jouer au ballon

La réglementation relative aux ballons-sondes diffère selon les pays. En France, si vous avez établi un partenariat avec *Planète Sciences*, votre capsule doit mesurer au moins 30 cm et la fréquence de transmission doit être de 137,950 MHz.

L'hélium est cher, les amateurs préfèrent donc utiliser l'air chaud ou le solaire ; ces ballons n'explosent pas et peuvent parcourir des centaines de kilomètres.

La prise de photos ou de vidéos peut se faire de différentes façons selon le budget disponible. Je me suis servi de deux caméras : une HD, qui a donné des images d'excellente qualité, et une mini-caméra « porte-clés », dont les prises de vue étaient correctes pour le prix payé.

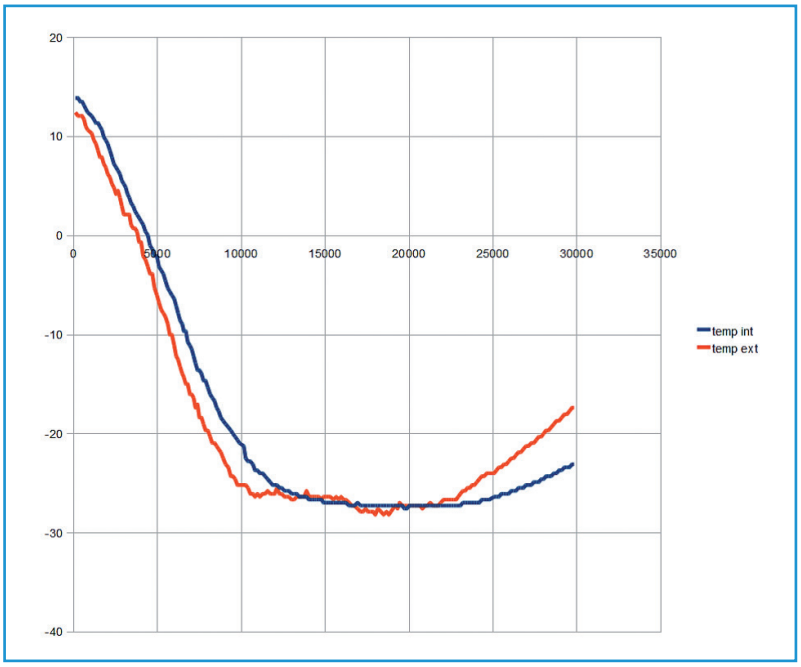


Figure 9. Température enregistrée durant le vol test. Nous attendions 30 °C de moins. L'écart est sans doute dû à la chaleur dégagée par la capsule.

Côté mesures, j'ai enregistré une température extérieure minimum de –28 °C à 15000 m, alors que la température attendue était d'environ –56 °C à 10000 m. Je pense que le capteur avait été affecté par la chaleur émise

Liste de vérification

Préparation

- Sur le PC, lancer les programmes dans cet ordre :
  - AGWPE
  - AGWTrackerXP
  - Packet Engine
- allumer le récepteur VHF (accordé avec l'émetteur)
- ne pas connecter tout de suite le récepteur VHF au PC.

Simulation GPS avec SimGPS

- Vérifier les cavaliers sur la carte
- relier le connecteur K9 au PC via un câble croisé doté à chaque extrémité d'un connecteur D-sub femelle à 9 broches
- lancer le simulateur NMEA, envoyer des trames
- allumer l'émetteur VHF
- alimenter la carte en 8 à 9 V
- la LED 2 s'allume avec l'envoi d'une trame.

Réception

- Les trames entrantes devraient être audibles côté récepteur
- si c'est le cas, relier le récepteur au PC
- le programme AGWTrackerXP devrait afficher une trame
- pour voir le ballon se déplacer sur la carte, faire un clic droit sur son indicatif, cliquer sur Locate, puis sur Show on map.

par la capsule. Lorsque le ballon entre dans la stratosphère, la température augmente à nouveau. La température mesurée à l'intérieur de la capsule suit la même courbe **(fig. 9)**.
La **figure 10** montre l'altitude mesurée par l'unité GPS. La phase de montée est très linéaire ; elle correspond, en fonction du volume de l'hélium, à une vitesse comprise entre 4 m/s et 6 m/s. La descente rapide s'explique par l'éclatement du ballon vers 30.000 m.
Le système décrit ici a été développé en milieu scolaire, avec des étudiants qui ont pris autant de plaisir que moi à le construire. D'expérience, je puis vous dire que l'excitation et la nervosité augmentent à mesure qu'approche la date du lancement, donc prenez le temps d'établir une liste de vérification et d'en contrôler les points plusieurs fois : il n'y a qu'un seul ballon ! Le voir s'envoler et récolter les données est vraiment plaisant. Étudier les prévisions météo pour estimer l'endroit où atterrira la capsule est également amusant. La chasse à la capsule commence dès l'instant où le ballon éclate. Même si le module de suivi permet de la retrouver facilement, la présence d'arbres ou de cours d'eau peut réserver quelques surprises. Autre récompense de vos efforts, le visionnage des photos ou vidéos prises durant le vol. Si vous avez raté les clichés saisissants présentés dans l'Elektor.POST n° 20, vous pouvez les retrouver sur la page du lien [7]. La dernière image montre le trajet suivi par le ballon depuis Le Mans. Impressionnant !

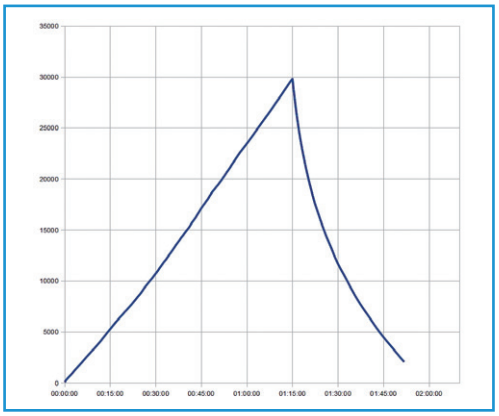


Figure 10.
L'altitude enregistrée par le GPS. On voit que le ballon a éclaté à 30000 m.

(130082 – version française : Hervé Moreau)

Liens

[1] http://www.trackuino.org

[2] http://www.elektor-labs.com/130082

[3] http://fr.wikipedia.org/wiki/Automatic_Packet_Reporting_System

[4] http://www.tapr.org/pub_ax25.html

[5] http://en.wikipedia.org/wiki/List_of_amateur_radio_organizations

[6] http://www.sv2agw.com/downloads/default.htm

[7] http://bit.ly/17x8dlm