RÉCEPTION ET DÉCODAGE DE TRAMES GPS

Pour commencer nous avons mis en place une fonction *init*, qui nous servira à initialiser la réception des trames GPS en **4800** **8N1**, pour ce faire on a utilisé la fonction *outportb*, qui a pour but d'envoyer un octet sur le port b. Tout d’abord nous avons passez le **DLAB** à **0**, puis initialiser le registre **DDL** et enfin le registre **DLM**, avant de remettre le **DLAB** à **1**.

Une fois la fonction *init* terminée nous entrons dans une boucle *do while*, qui se terminera tant que l'utilisateur n'aura pas appuyé sur une touche.

Dans la boucle *do while* nous allons vérifier à l'aide la fonction *lireBit* si un caractère a été reçu par le port série, si oui alors nous affecterons à la variable var le caractère reçu, grâce à la fonction *inportb.*

Maintenant nous allons passer à l'analyse des caractères reçus. Ainsi si le caractère reçu est un **$,** on affecte à la variable **deb**: **1**. Puis si le second caractère reçu est un **G** et que le précèdent était un **$,** soit **deb** égale **1**, alors on affecte à **deb** la valeur **2**. Enfin si le caractère suivant est un **P** et que les deux précédents était dans l'ordres un **$** puis un **G** alors on affectera à **deb** la valeur **3**. Et enfin si **deb** est égale a trois (donc on a reçu à la suite **$,** **G** et **P**) alors on affectera au tableau **cara**, tous les caractères suivants.

Pour marquer la fin de la trame, nous allons vérifier si nous recevons le caractère retour chariot en hexadécimal : **d**,**d** et **a**. Afin de marquer la fin de la trame, nous allons:

* Finir notre tableau de caractère appelé **cara** avec un **\0**.
* Appeler la fonction *décodage* afin de décoder la trame reçue
* Afficher à l'aide de la fonction *aff* le résultat du décodage de la trame
* Et initialiser les variable **deb**, **fin** et **i** à **0**

La fonction décodage se présente de la manière suivante. Tout d'abord nous allons regarder quel est le type de trame reçu. Sachant que le premier caractère de la variable **cara** est un P, nous allons regarder à quoi correspondent les trois prochains caractères. Si ils correspondent aux caractères **RMC** alors nous faisons appel à la fonction *foncRmc*, si ils correspondent aux caractères **GGA** alors on fait appel à la fonction *foncGga*, et si ils correspondent aux caractères **VTG** alors on fait appel à la fonction *foncVtg.*

Dans la fonction *foncVtg*, nous allons étudier la chaine de caractères de la variable **cara** afin de récupérer la vitesse en km/h, pour se faire nous ferons appel à la fonction *strtok*. C'est une fonction qui retourne dans la variable **var** une chaine de caractère compris entre chaque virgule, qui aura été donnée en paramètre. Étant donné la taille fixe de chaque trame, nous pouvons ainsi prédire où seront les trames que nous recherchons. Ici, la vitesse se trouve à la 7ème position. Une fois que que **var** aura récupérer la vitesse du module GPS, nous allons stoker la valeur dans la structure **vtg.vit** initialiser au début du programme.

Dans la fonction *foncGga*, nous allons étudier la chaine de caractères de la variable **cara** afin de récupérer la le nombre de satellite, indiquant notre position et l'altitude du module GPS, comme précédemment nous ferons appel à la fonction *strtok*. Ici, le nombre de satellite se trouve à la 8éme position. Une fois que **var** aura récupérer le nombre de satellite du module GPS, nous allons stoker la valeur dans la structure **gga.sat** initialiser au début du programme. L'altitude se trouve à la 10éme position. Une fois que **var** aura récupérer l'altitude du module GPS, nous allons stoker la valeur dans la structure **gga.alt** initialiser au début du programme.

Dans la fonction *foncRmc*, nous allons étudier la chaine de caractères de la variable **cara** de la même manière que pour les trames précédentes, afin de récupérer l'heure, la date, la latitude et la longitude et les stocker dans la structure rmc initialiser au début du programme.