

# Simulation d'un émetteur / récepteur ADS-B et décodage temps réel à l'aide de radio logicielle Projet TS229 – Année 2018/2019

Guillaume Ferré, Romain Tajan et Baptiste Laporte-Fauret

## Table des matières

<b>1 Objectifs</b>	<b>2</b>
<b>2 Évaluation</b>	<b>2</b>
<b>3 ADS-B : Introduction et fonctionnement</b>	<b>4</b>
<b>4 Partie 1 – Couche physique ADS-B</b>	<b>4</b>
4.1 Présentation . . . . .	4
4.2 Questions théoriques et implémentations Matlab : . . . . .	4
4.2.1 Prise en main de la chaine de communication ADSB - Durée 4h . . . . .	5
4.2.2 Densité spectrale de puissance du signal $s_l(t)$ - Durée 2h . . . . .	6
4.2.3 Performances en terme de probabilité d'erreur binaire - Durée 2h . . . . .	7
4.2.4 Mise en oeuvre des algorithmes de codage et de décodage de canal - Durée 2h $\perp$ . . . . .	8
4.2.5 Synchronisation en temps et en fréquence - Durée 4h $\perp$ . . . . .	9
<b>5 Partie 2 – Traitement / décodage de signaux réels <math>\perp</math></b>	<b>10</b>
5.1 Structure des trames ADS-B $\perp$ . . . . .	11
5.2 Implémentation de la couche MAC - Durée 3h $\perp$ . . . . .	13
<b>6 Partie 3 – Affichage des trajectoires - Durée 3h</b>	<b>14</b>
<b>7 Contacts</b>	<b>14</b>
<b>Appendices</b>	<b>15</b>
<b>Annexe A Tableaux de Format Type Codes</b>	<b>15</b>
<b>Annexe B Tableaux de structure des messages</b>	<b>17</b>
<b>Annexe C Encodage et décodage des altitudes, latitudes et longitudes</b>	<b>18</b>
<b>Annexe D Tableaux des caractères</b>	<b>20</b>

# 1 Objectifs

L'objectif du projet TS226 est de mettre en pratique dans un cas concret les cours et les TP de communications numériques, codage de canal et traitement du signal. Pour cela, il est demandé de simuler sous *MATLAB* un émetteur/récepteur de données ADS-B (Automatic Dependent Surveillance Broadcast). Puis sur la base de vos résultats de simulation, vous adapterez votre récepteur afin d'être en mesure d'effectuer un décodage temps réel des avions survolant l'école. Ce projet peut être décomposé en 3 parties différentes :

- **Partie 1 :** La première partie de ce projet est consacrée à la simulation d'une couche physique de l'ADS-B. Vous mettrez alors en place une chaîne de simulation permettant d'émuler le fonctionnement d'un émetteur et d'un récepteur ADS-B. Après une analyse théorique des signaux émis, vous simulerez avec *MATLAB* la couche physique ADS-B. Cette dernière devra notamment être composée : d'un codeur de canal, du filtre de mise en forme de l'information binaire, d'un canal AWGN, d'un filtrage adapté, d'un bloc de décision, etc. A partir de cette chaîne vous analyserez les signaux temporels et fréquentiels ainsi que les performances du système en terme de *bit error rate* (BER) et de diagramme de l'œil.
- **Partie 2 :** La seconde partie de ce projet est dédiée à l'étude de la structure des messages envoyés pour l'ADS-B. Dans cette partie, vous décoderez les paquets démodulés afin d'en extraire des informations sur les appareils transmettant ces messages. Ces informations vont de la position de l'appareil (latitude, longitude, altitude) à son nom ou encore son adresse.
- **Partie 3 :** La troisième partie est la concaténation des résultats obtenus en première et deuxième partie. Il s'agit alors de décoder des signaux réels de bout en bout, d'en extraire les informations et d'afficher sur une carte les trajectoires à l'image de l'exemple donné en Figure 1.

# 2 Évaluation

Le projet se fait en **binôme** et l'évaluation porte sur une note de rapport et une note de travail continu (en séances).

Concernant le rapport, il ne doit pas excéder **15 pages** et il devra être remis à votre enseignant au format électronique (document pdf) au **plus tard le 28 novembre 2018**. Vous devez fournir un document scientifique et technique, qui doit présenter votre travail, vos choix techniques et dans lequel tous les résultats obtenus doivent être interprétés et commentés.

Les codes *MATLAB* doivent être transmis à votre enseignant par Email au **plus tard le 28 novembre 2018**. Ils doivent pouvoir être compris rapidement. Cela passe par l'utilisation de commentaires. Les commentaires doivent permettre de répondre au moins à la question : que fait la ligne de code ? Une attention particulière doit être portée à la lisibilité du programme : un menu *help* doit être présent dans chacune de vos fonctions et le début de chaque fonction doit être réservé à l'initialisation des variables. Il est rappelé que le plagiat est interdit, référez-vous à la charte que vous avez signée afin de mesurer les conséquences de tel acte.

Enfin, pour chaque sous-sections nous avons identifié une durée approximative de réalisation. Certaines sous-sections peuvent être traitées de manières indépendantes. Si tel est le cas nous l'indiquons avec le signe suivant  $\perp$ . Ainsi, essayer dans la mesure du possible de paralléliser le travail.



### 3 ADS-B : Introduction et fonctionnement

Afin de surveiller l'état du réseau aérien, un système de diffusion appelé *Automatic Dependant Surveillance - Broadcast* (ADS-B) a été proposé en complément des radars classiques. Dans ce système, les appareils estiment leurs positions (longitude, latitude, altitude) grâce aux techniques de positionnement par satellite (GPS - Global Positioning System, GLONASS - Global Navigation Satellite System ou encore Galiléo) et diffusent ces informations régulièrement (toutes les secondes environ). Ces informations sont ensuite récupérées :

- au sol : par des stations intermédiaires ou des tours de contrôle,
- dans les autres appareils : qui peuvent utiliser ces signaux pour leurs systèmes anti-collision.

Le principal avantage du système ADS-B par rapport au radar classique est son faible coût d'infrastructure. En effet, la station réceptrice possède seulement une antenne afin de recevoir les signaux ADS-B, le reste des traitements étant faits à bord des appareils. Le récepteur est donc entièrement passif et n'a pas besoin d'interroger l'appareil pour qu'il émette sa position.

Il existe plusieurs liaisons pour la transmission des signaux ADS-B. Les deux principales sont :

- le 1090 Extended Squitter (1090 ES),
- l'UAT (Universal Access Transponder).

Dans ce projet, nous allons nous concentrer sur le 1090 ES car c'est le mode privilégié en Europe. Dans l'appellation 1090 ES, 1090 signifie que la fréquence porteuse des signaux ADS-B (pour ce mode de transmission) est 1090 MHz. ES signifie Extended Squitter, ce qui pourrait être traduit par message étendu. En effet, les messages transmis en utilisant ce mode avaient une durée de 56 bits (hors préambule de synchronisation). Dans le nouveau standard, les messages peuvent contenir 112 bits, d'où la notion de "message étendu".

## 4 Partie 1 – Couche physique ADS-B

### 4.1 Présentation

Il existe de nombreuses façons de mettre en oeuvre un flux d'information binaire. Nous avons notamment vu l'année dernière en cours de communications numériques que les informations binaires étaient modulées numériquement pour créer des symboles, symboles qui étaient par la suite filtrés par un filtre dit de *mise en forme*. Dans le cadre du standard ADS-B, les signaux transmis autour de la porteuse à 1.09GHz sont obtenus par une modulation dite en position d'amplitude (PPM - Pulse Position Modulation). La modulation PPM considérée est binaire avec une période symbole  $T_s = 1\mu s$ . Cette modulation PPM encode les informations binaires 0 et 1 avec les impulsions  $p_0(t)$  et  $p_1(t)$  représentées respectivement en Figure 2(a) et Figure 2(b).

L'enveloppe complexe (i.e le signal bande de base) du signal envoyé s'écrit alors :

$$s_l(t) = \sum_{k \in \mathbb{Z}} p_{b_k}(t - kT_s) \quad (1)$$

sachant que  $T_s$  représente ici le temps de l'impulsion élémentaire ( $p_0(t)$  ou  $p_1(t)$ ) et que

$$p_{b_k}(t) = \begin{cases} p_0(t), & \text{si } b_k = 0 \\ p_1(t), & \text{si } b_k = 1 \end{cases}$$

### 4.2 Questions théoriques et implémentations Matlab :

Hypothèses :

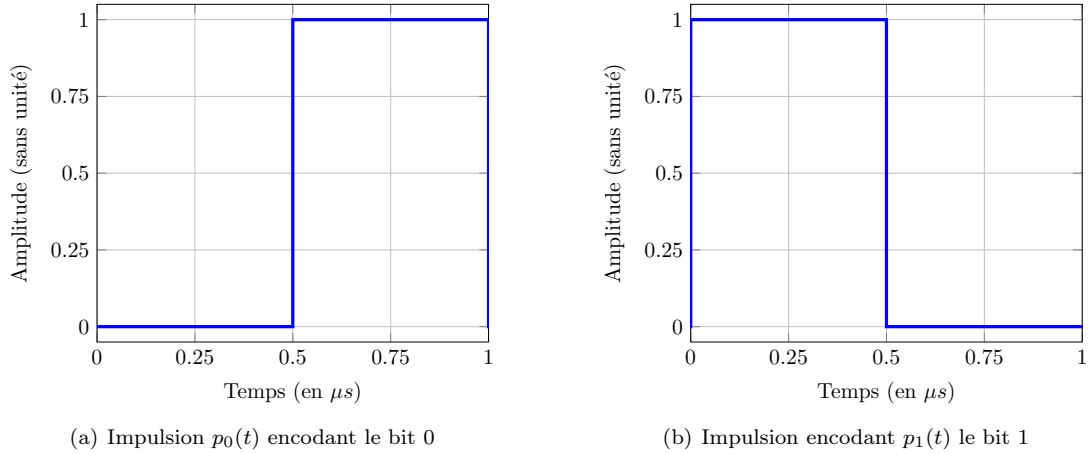


FIGURE 2 – Impulsions de base pour la modulation par position, **ces impulsions sont nulles en dehors de la partie présentée.**

- Les  $b_k$  sont indépendants et distribués uniformément,
- Le bruit bande de base  $n_l(t) \sim \mathcal{N}(0, \sigma_{n_l}^2)$  de densité spectrale de puissance (DSP) bilatérale  $\Gamma_{n_l}(f) = \frac{N_0}{2}$ ,
- Le modèle bande de base de l'architecture de communication considérée (en excluant les parties codage et décodage de canal) est présenté sur la figure 3 sachant que  $p(t)$  est donné sur la figure 4.

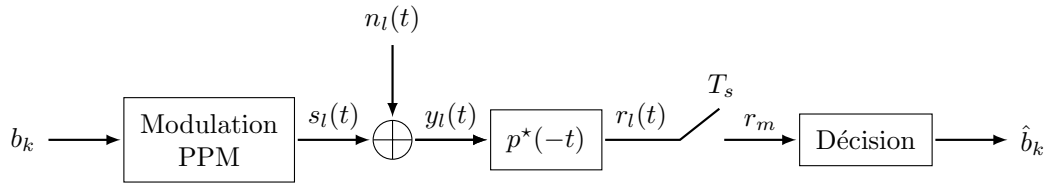


FIGURE 3 – Chaîne de communication complète,  $p(t)$  étant donné en Figure 4.

#### 4.2.1 Prise en main de la chaîne de communication ADSB - Durée 4h

**Question 1 - Théorie** - En considérant un bruit nul, que le signal binaire à émettre vaut  $[1, 0, 0, 1, 0]$  et que les signaux sont causaux, représenter graphiquement  $s_l(t)$ ,  $r_l(t)$  et  $r_m$ . Déduire de ces signaux le rôle du bloc décision.

**Question 2 - Matlab** - Sur la base de vos résultats à la question précédente, implémenter à l'aide du logiciel Matlab, la chaîne de communication présentée sur la figure 3 et corroborer les allures théoriques de  $s_l(t)$ ,  $r_l(t)$  et  $r_m$  avec celles fournies par Matlab en émettant la séquence  $[1, 0, 0, 1, 0]$ . Afin d'être en mesure d'obtenir les allures discrètes des différents filtres de mise en forme, vous supposerez que la fréquence d'échantillonnage  $f_e$  du système est fixée à 20 MHz. Les versions discrètes de vos filtres de mise en forme ( $p_0(t)$  et  $p_1(t)$ ) et du filtre de réception  $p^*(T_s - t)$  seront alors des vecteurs composés de  $F_{se} = \frac{T_s}{T_e} = 20$  échantillons. Où  $T_e = \frac{1}{f_e}$  désigne ici la période d'échantillonnage. Le code Matlab ainsi obtenu sera nommé **ADSB\_part1\_1.m**.

- **Vérification** - Faites vérifier à votre encadrant la justesse de vos résultats en comparant les allures obtenues sur le papier avec celles données par Matlab. Une fois vérifié envoyer lui immédiatement par email votre code en ajoutant vos noms comme ci-après : `ADSB_part1_1_Nom1_Nom2.m`

#### 4.2.2 Densité spectrale de puissance du signal $s_l(t)$ - Durée 2h

**Question 3 - Théorie** - Montrer que le signal émis, donné en équation (1) peut se réécrire sous la forme

$$s_l(t) = 0.5 + \sum_{k \in \mathbb{Z}} A_k p(t - kT_s) \quad (2)$$

où

$$A_k = \begin{cases} 1, & \text{si } b_k = 0 \\ -1, & \text{si } b_k = 1 \end{cases}$$

et  $p(t)$  est la forme d'onde biphase donnée dans la Figure 4.

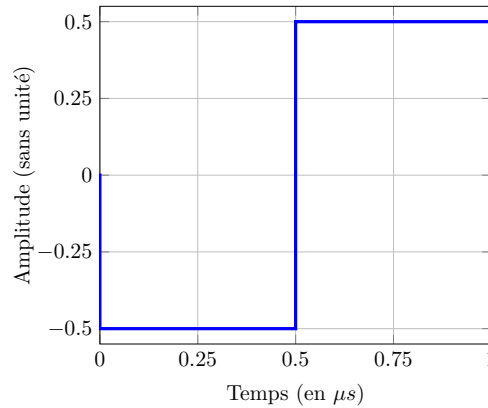


FIGURE 4 – Forme d'onde biphase  $p(t)$ , cette impulsion est nulle en dehors de la partie présentée.

**Question 4 - Théorie** - Calculer le moment d'ordre 1 du signal  $s_l(t)$ ,  $m_{s_l}(t) = E[s_l(t)]$  et montrer qu'il ne dépend pas de  $t$  (i.e.  $m_{s_l}(t) = m_{s_l}$ ).

**Question 5 - Théorie** - Calculer la fonction d'autocorrélation du signal  $s_l(t)$  :  $R_{s_l}(t, \tau) = E[s_l(t)s_l^*(t + \tau)]$ .

**Question 6 - Théorie** -  $s_l(t)$  est cyclo-stationnaire de période de cyclo-stationnarité  $T_s$ . Sa Densité Spectrale de Puissance (DSP) s'obtient alors en calculant la transformée de Fourier de l'autocorrélation moyennée du signal  $s_l(t)$  :

$$\tilde{R}_{s_l}(\tau) = \frac{1}{T_s} \int_0^{T_s} R_{s_l}(t, \tau) dt.$$

. Calculer  $\tilde{R}_{s_l}(\tau)$  et représenter graphiquement son allure.

**Question 7 - Théorie** - En déduire la DSP de  $s_l(t)$  :

$$\Gamma_{s_l}(f) = \mathcal{F} \left( \tilde{R}_{s_l}(\tau) \right).$$

(Exprimer cette DSP en fonction de  $T_s$ )

**Question 8 - Matlab** - L'objectif de cette nouvelle implémentation Matlab est d'obtenir une estimation de la DSP de  $s_l(t)$  en utilisant l'algorithme du périodogramme de Welch sans chevauchement et sans fenêtre de pondération. Cette estimation devra être comparée à l'expression analytique obtenue à la question précédente. Vous devez implémenter l'algorithme de Welch par vous-même et ne pas chercher à utiliser une fonction Matlab toute faite (exemple : `pwelch.m`).

Repartir du code `ADSB_part1_1.m` afin de l'enregistrer sous un nouveau nom : `ADSB_part1_2.m`. C'est dans ce nouveau code que vous allez implémenter le calcul de la DSP. Quelques modifications sont bien entendu nécessaires. Tout d'abord, les bits doivent désormais être générés aléatoirement suivant une loi discrète uniforme. Comme on ne s'intéresse qu'au calcul de la DSP de  $s_l(t)$ , ne conserver que les parties "ajout de bruit" et "récepteur".

Implémenter la fonction `Mon_Welch.m` dont le prototype doit être : `y=Mon_Welch(x,NFFT)` où :

- `x` est le vecteur contenant les échantillons du signal pour lequel il faut calculer la DSP,
- `NFFT` représente le nombre de points sur lequel les FFT doivent être calculées,
- `y` est l'estimation de la DSP de `x`.

Afin d'obtenir une estimation constante de la DSP elle devra être obtenue en moyennant au minimum 100 FFT de  $NFFT = 256$  points.

- **Vérification** - Afficher sur une même figure les DSP de  $s_l(t)$  obtenues analytiquement et expérimentalement. Ces dernières doivent se superposer. Si ce n'est pas le cas il vous appartient d'identifier la ou les raison(s) et de les corriger. Une fois vérifié envoyer lui immédiatement par email votre code en ajoutant vos noms comme ci-après : `ADSB_part1_2_Nom1_Nom2.m`

#### 4.2.3 Performances en terme de probabilité d'erreur binaire - Durée 2h

**Question 9 - Théorie** - Calculer la probabilité d'erreur binaire  $P_b$  pour la modulation PPM et le récepteur proposé sur la figure 3. Exprimer  $P_b$  en fonction du rapport  $\frac{E_b}{N_0}$  et de la fonction  $\text{erfc}(\cdot)$ . On vous rappelle que ce calcul s'obtient en partant de l'expression de  $r_m$  (voir cours de communications numériques).

**Question 10 - Matlab** - L'objectif de cette implémentation est d'obtenir par la simulation Matlab, la courbe de taux d'erreur binaire du système présenté sur la figure 3. Pour cela enregistrer sous `ADSB_part1_3.m` le code `ADSB_part1_1.m`. Comme pour le code `ADSB_part1_2.m`, les bits doivent être générés aléatoirement et suivant une loi discrète uniforme. Cependant cette fois-ci la taille du paquet d'information binaire est fixé et égale à  $N_b = 1000$  bits. Modifier le code en conséquence afin de d'obtenir le Taux d'Erreur Binaire (TEB) en fonction du rapport signal à bruit  $\frac{E_b}{N_0}$  pour des valeurs allant de 0dB à 10dB par pas de 1dB. On considérera qu'une valeur de TEB est légitime si le nombre d'erreur calculé est supérieur à 100. Tracer sur la même figure la valeur théorique de  $P_b$  déterminée en Question 9 ces dernières doivent se superposer.

- **Vérification** - Faites vérifier à votre encadrant le résultat de superposition et une fois vérifié envoyer lui immédiatement par email votre code en ajoutant vos noms comme ci-après : `ADSB_part1_3_Nom1_Nom2.m`

#### 4.2.4 Mise en oeuvre des algorithmes de codage et de décodage de canal - Durée 2h ±

L'objectif de cette partie est de vous faire implémenter le codeur et le décodeur de canal utilisés dans le cadre de la transmission des signaux ADS-B. La structure des signaux ADS-B émis par les avions est présentée sur la figure 8. Il ne s'agit pas ici de comprendre précisément la signification des différentes parties de la trame ADS-B. Ce travail se fera dans un second temps, lorsque nous aborderons la couche MAC. Ici nous allons simplement remarquer qu'une trame ADS-B dure  $120\mu s$  soit l'équivalent de 120 bits et la présence d'un contrôle de parités de 24 bits en fin de trame (voir 8).

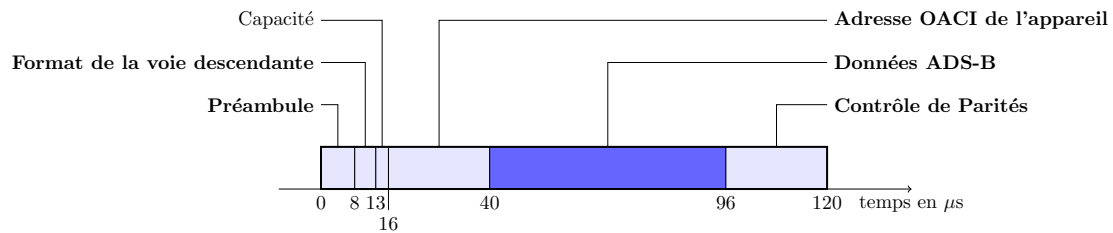


FIGURE 5 – Format d'une trame ADS-B

Les bits de parités sont issus d'un Code à Redondance Cyclique (CRC) de 24 bits dont le polynôme générateur est donné ci-dessous

$$p(x) = x^{24} + x^{23} + x^{22} + x^{21} + x^{20} + x^{19} + x^{18} + x^{17} + x^{16} + x^{15} + x^{14} + x^{13} + x^{12} + x^{10} + x^3 + 1.$$

Le codage CRC rajoute 24 bits au message initial ce dernier étant composé de 88 bits d'informations utiles et de d'un entête (dit préambule) dont la durée est équivalente à celle de 8 bits. Les 24 bits de CRC servent à détecter d'éventuelles erreurs (parmi les 88 bits utiles) en réception. Le but principal de ce projet n'étant pas de programmer un détecteur d'erreur basé sur les CRC, vous utiliserez les fonctions MATLAB dédiées à cet effet. Deux fonctions pourront en particulier être utiles :

- `crc.detector` qui sert à construire un détecteur CRC à partir du polynôme générateur,
- `detect` méthode d'un objet de type `crc.detector` afin de détecter la présence d'erreurs.

**Question 11 - Matlab** - L'objectif de cette implémentation est de mettre en œuvre les algorithmes de codages et de décodage de canal. Pour cela enregistrer sous `ADSB_part1_4.m` le code `ADSB_part1_1.m`. Comme pour le code `ADSB_part1_2.m`, les bits doivent être générés aléatoirement et suivant une loi discrète uniforme. Cependant cette fois-ci la taille du paquet d'information binaire est fixée et égale à  $Nb = 88$  bits, soit la taille du paquet de bits utiles dans le paquet ADS-B. Rajouter les parties de codage et de décodage de canal comme présenté sur la Figure 3.

Implémenter cette nouvelle chaîne de communication en y insérant le codeur et le décodeur de canal. Tester le bon fonctionnement du système en vérifiant que le décodeur vous indique :

- que le message est intègre en l'absence d'erreur,
- que le message n'est pas intègre en présence d'erreur parmi les 88 bits de données utiles,

- **Vérification** - Faites vérifier à votre encadrant les résultats et une fois vérifié envoyer lui immédiatement par email votre code en ajoutant vos noms comme ci-après : `ADSB_part1_4_Nom1_Nom2.m`.



#### 4.2.5 Synchronisation en temps et en fréquence - Durée 4h ⊥

Deux défauts n'ont pas été pris en compte dans les questions précédentes :

- la synchronisation temporelle : le signal subit un délai de propagation  $\delta_t$ , il faut le compenser,
- la synchronisation fréquentielle : l'effet Doppler introduit par le mouvement de l'avion ainsi que les défauts d'oscillateurs locaux introduisent un décalage en fréquence  $\delta_f$ .

Ces deux effets sont pris en compte avec le modèle en bande de base suivant

$$y_l(t) = s_l(t - \delta_t)e^{-j2\pi\delta_f t} + n_l(t)$$

où  $\delta_t$  et  $\delta_f$  représentent respectivement les désynchronisations temporelle et fréquentielle du signal.

**Question 12** - Théorie - Quelle est l'ordre de grandeur du décalage de fréquence Doppler d'un avion se déplaçant à 900km/h ?

**Question 13** - Théorie - On va désormais considérer l'architecture de communication présentée sur le figure 6.

1. Montrer que  $|y_l(t)|^2 = s_l(t - \delta_t) + z_l(t)$ .  $z_l(t)$  est-il un bruit blanc gaussien ?  $z_l(t)$  est-il indépendant de  $s_l(t)$  ?
2. Quel est l'avantage de prendre le carré du module de l'enveloppe complexe du signal reçu ?

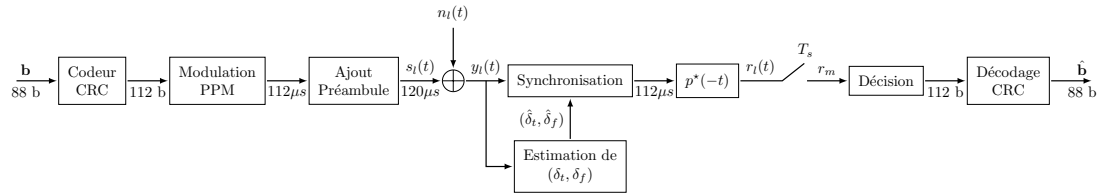


FIGURE 6 – Chaîne de communication complète,  $p(t)$  étant donné en Figure 4.

La synchronisation est réalisée à la réception en utilisant un signal  $s_p(t)$  de durée  $T_p = 8 \mu s$  appelé préambule et envoyé en entête des trames ADS-B (voir figure 8). Le préambule est le signal donné en Figure 7. Une telle forme d'onde ne peut pas être présente dans le signal  $s_l(t)$ . En effet si vous essayez d'obtenir  $s_p(t)$  à partir d'une combinaison binaire de 8 bits en sortie du modulateur PPM vous n'y parviendrez pas. Cette unicité permet de mettre en œuvre une méthode simple et très répandue pour effectuer la synchronisation temps/fréquence d'un signal : **l'intercorrélation**.

Désormais, pour  $t \in [0, T_p]$   $s_l(t) = s_p(t)$  de sorte que

$$y_l(t) = s_p(t - \delta_t)e^{-j2\pi\delta_f t} + n_l(t), \text{ pour } t \in [\delta_t, \delta_t + T_p]$$

où  $s_p(t)$  est le signal de préambule connu de l'émetteur et du récepteur. On effectue la synchronisation temporelle en cherchant le maximum de la corrélation suivante :

$$\rho(\delta'_t) = \frac{\int_{\delta'_t}^{\delta'_t + T_p} r_l(t) s_p^*(t - \delta'_t) dt}{\sqrt{\int_0^{T_p} |s_p(t)|^2 dt} \cdot \sqrt{\int_{\delta'_t}^{\delta'_t + T_p} |r_l(t)|^2 dt}}.$$

L'estimation est alors réalisée en prenant la valeur de  $\hat{\delta}_t$  telle que

$$\hat{\delta}_t = \arg \max_{\delta'_t} |\rho(\delta'_t)| \quad (3)$$

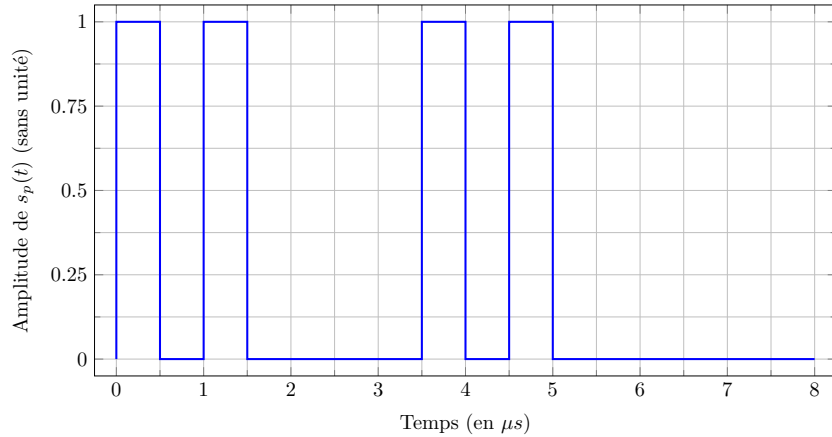


FIGURE 7 – Préambule  $s_p(t)$  de  $T_p = 8 \mu s$  débutant les trames ADS-B

**Question 14 - Théorie -** Montrer que  $|\rho(\delta'_t)| \leq 1$  pour tout  $\delta'_t$ . Donner le cas d'égalité.

**Question 15 - Matlab -** L'objectif de cette implémentation est de mettre en oeuvre l'algorithme de synchronisation temporelle du signal. Pour cela enregistrer sous `ADSB_part1_5.m` le code `ADSB_part1_4.m`. Modifier le code en conséquence afin que ce dernier soit cohérent avec la figure 6. Les distorsions seront modélisées sachant que :

- Le délai de propagation  $\delta_t$  est aléatoirement et uniformément réparti entre 0 et  $100T_e$ ,
- Le décalage en fréquence  $\delta_f$  est aléatoirement et uniformément réparti entre  $-1kHz$  et  $1kHz$ .

Il est conseillé d'implémenter l'algorithme de synchronisation dans une fonction Matlab dédiée à cet effet.

**- Vérification -** Faites vérifier à votre encadrant que sans bruit dans la chaîne de communication vous estimez parfaitement le décalage temporel  $\delta_t$ . Une fois vérifié envoyer lui immédiatement par email votre code en ajoutant vos noms comme ci-après : `ADSB_part1_5_Nom1_Nom2.m`

**Question 16 - Matlab -** la bonne réponse à cette question vous offrira un bonus de 3 points. On cherche maintenant à observer les performances du récepteur proposé malgré les imperfections en temps et en fréquence. Adapter votre précédent code `ADSB_part1_5.m` en le renommant au préalable `ADSB_part1_6.m` pour désynchroniser aléatoirement chaque nouvelle trame émise et calculer le TEB en fonction de  $\frac{E_b}{N_0}$  pour des valeurs allant de 0 à 10dB par pas de 1dB. Superposer cette courbe de résultats avec la courbe de probabilité d'erreur binaire théorique et identifier le nombre de dB perdu pour un  $TEB=10^{-3}$ .

**- Vérification -** Montrer à votre encadrant le résultat obtenu et envoyer lui immédiatement par email votre code en ajoutant vos noms comme ci-après : `ADSB_part1_6_Nom1_Nom2.m`

## 5 Partie 2 – Traitement / décodage de signaux réels ⊥

Le but de cette partie est de se familiariser avec la structure des signaux ADS-B réels afin de pouvoir tester le décodeur implémenté dans la section précédente sur des signaux réels.

## 5.1 Structure des trames ADS-B

Les signaux émis par les appareils pour l'ADS-B ont une durée de  $120\ \mu s$ . Ils sont constitués des parties suivantes :

- le préambule (identique à celui de la section précédente),
- le format de la voie descendante,
- la capacité,
- l'adresse OACI (Organisation de l'Aviation Civile Internationale) de l'appareil,
- les données ADS-B,
- les bits de contrôle de parité.

La durée de chacune des parties ainsi que leur position dans une trame ADS-B sont représentées en Figure 8. Nous détaillons maintenant la fonction des différentes parties de la trame.

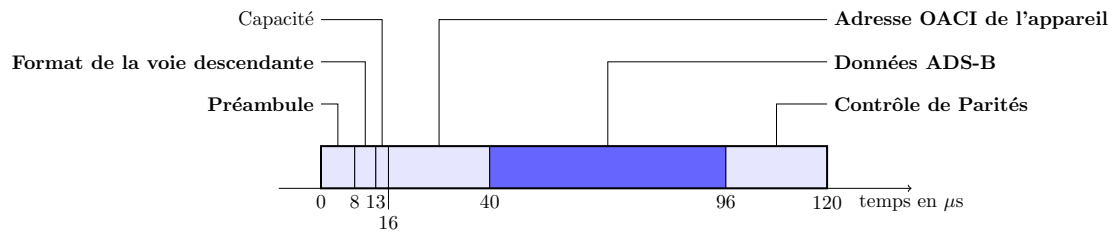


FIGURE 8 – Format d'une trame ADS-B

### Le préambule

Il sert pour la synchronisation temporelle et fréquentielle des signaux. Le préambule est identique quelque soit les types de signaux ADS-B transmis, il dure  $8\ \mu s$  et est identique à celui utilisé dans la section précédente.

### Le format de la voie descendante

Le format de la voie descendante (parfois noté DF pour Downlink Format) est codé sur 5 bits et indique le type de la trame envoyée. Les valeurs particulières de ce format sont les suivantes

- 11 : message d'acquiescement
- **17 : message de type ADS-B**
- 18 : message de type TIS-B (Traffic information service – broadcast)
- 19 : message de type ADS-B militaire (crypté)

Dans ce projet, nous nous intéresserons seulement aux trames avec **DF = 17**.

### La capacité

La capacité est notée CA (pour CApacity). Elle est codée sur 3 bits et représente un sous-type de trame envoyée. Nous ne nous préoccupons pas de sa signification.

### L'adresse OACI de l'appareil

L'OACI est l'Organisation de l'Aviation Civile Internationale, elle définit entre autre une immatriculation pour les aéroports, aérodromes et appareils volants. Dans les trames ADS-B, l'adresse OACI de l'appareil (parfois notée AA pour Aircraft Address) est codée sur 24 bits. Chaque appareil possède une adresse unique tout au long de sa vie, quelque soit sa compagnie ou son plan de vol.

Dans ce projet, il est primordial de pouvoir décoder correctement cette adresse car elle servira à connaître de quel appareil proviennent les données décodées. Cette adresse sert aussi pour trouver un avion dans les bases de données des tours de contrôles, ou des radars virtuel (tels que <http://planefinder.net> ou <http://www.flightradar24.com>). On pourra donc se servir des radars virtuels afin d'avoir plus d'informations sur les appareils ayant transmis les signaux reçus.

### Les données ADS-B

Dans le cadre de données envoyée pour ADS-B (DF = 17), les messages sont composés de 56 bits et durent  $56\mu s$ . Les messages pouvant être transmis sont répertoriés dans des registres. Le contenu de ces messages dépend du type de registre. On trouve entre autres des registres pour les informations de position au sol, de **position en vol**, de **signes d'identification** et de vitesse. Ceux qui vont nous intéresser sont ceux écrits en gras, les derniers étant des bonus.

Dans tous les cas, les registres sont constitués de 56 bits. Les 5 premiers bits du message constituent le code du format des données (FTC-Format Type Code en anglais). Ces bits indiquent si le message correspond à un message de position au sol, un message de position en vol ou un message d'identification. Les tables de correspondances entre les 5 bits et le type de message sont données dans les Tables 3 à 5 en Annexe A.

**Question 17** *Quelles valeurs de FTC correspondent à des trames de position en vol ? Même question pour les messages d'identifications.*

La structure des trames de position en vol est donnée dans la Table 1 ci-dessous. Pour la suite du projet, nous ne considérerons ni les 2 bits de *surveillance* ni le bit de *type d'antenne* ni le bit *indicateur de temps UTC*. L'encodage de l'altitude ainsi que ceux des latitude et longitude sont donnés en Annexe C.

TABLE 1 – Composition du message de position en vol

Index binaire	Champs
1	MSB
⋮	Format Type Code
5	LSB
6	MSB
7	Surveillance Status
8	LSB
8	Indicateur de type d'antenne
9	MSB
⋮	Altitude
20	LSB
21	Indicateur de temps UTC
22	Indicateur de format CPR
23	MSB
⋮	Latitude encodée avec CPR
39	LSB
40	MSB
⋮	Longitude encodée avec CPR
56	LSB

La structure des trames d'identification est donnée Table 2. L'identification de l'appareil est composée de 8 caractères, chacun étant encodé sur 6-bits. La table de correspondance entre les 6 bits et les caractères est donnée en Table 7 dans l'Annexe D.

TABLE 2 – Composition du message d'identification

Index binaire	Champs
1	MSB
⋮	Format Type Code
5	LSB
6	MSB
⋮	Catégorie de l'appareil
8	LSB
9	MSB
⋮	Caractère 1 de l'identifiant
14	LSB
15	MSB
⋮	Caractère 2 de l'identifiant
20	LSB
21	MSB
⋮	Caractère 3 de l'identifiant
26	LSB
27	MSB
⋮	Caractère 4 de l'identifiant
32	LSB
33	MSB
⋮	Caractère 5 de l'identifiant
38	LSB
39	MSB
⋮	Caractère 6 de l'identifiant
44	LSB
45	MSB
⋮	Caractère 7 de l'identifiant
50	LSB
51	MSB
⋮	Caractère 8 de l'identifiant
56	LSB

## 5.2 Implémentation de la couche MAC - Durée 3h $\perp$

Le but de cette partie est de développer une fonction MATLAB qui convertit un message binaire en registre afin de pouvoir en extraire les données sur les appareils concernés. Afin de rendre synthétique votre code MATLAB, un registre sera représenté par une structure possédant les champs **format** (contenant DF), **adresse** (contenant AA), **type** (contenant le Format Type Code-FTC), **nom** (contenant le nom de l'appareil), **altitude** (contenant l'altitude), **timeFlag** (contenant l'indicateur de temps UTC), **cprFlag** (contenant l'indicateur CPR), **latitude** (contenant la latitude) et **longitude** (contenant la longitude). La façon la plus simple d'initialiser cette structure est de de la forme suivante

```
registre = struct('adresse',[],'format',[],'type',[],'nom',[], ...
                 'altitude',[],'timeFlag',[],'cprFlag',[], ...
                 'latitude',[],'longitude',[], 'trajectoire', []);
```

Pour initialiser un des champs de cette structure vous utiliserez la syntaxe suivante

```
registre.nom = 'AF1234';
registre.altitude = '34000';
```

Tous les messages envoyés pour l'ADS-B ne comportent pas systématiquement toutes les informations. En effet, les informations transmises dépendent du type de registre en train d'être envoyé et donc de la partie "message" d'une trame ADS-B. La structure de cette partie "message" est donnée en Annexe B pour des messages de position en vol, position au sol et d'identification de l'appareil.

Le code principal MATLAB nécessaire pour répondre aux questions suivantes devra être nommé `ADSB_part2.m`.

**Question 18** - Matlab - Écrire la fonction `MATLAB bit2registre` qui prend en argument un vecteur de 112 bits et un registre à mettre à jour, qui extrait les informations du vecteur binaire et qui renvoie le registre mis à jour seulement si le CRC ne détecte pas d'erreur.

**Question 19** - Matlab - Afin de tester votre code, téléchargez le fichier <http://rtajan.vvv.enseirb-matmeca.fr/Cours/Telecom/T2/>. Ce fichier contient les 112 bits suivant le préambule de 21 trames issues d'un même appareil (ces trames sont rangées en colonnes). Extrayez toutes les informations de ces trames et affichez la trajectoire de l'avion considéré.

Pour effectuer l'affichage, vous pourrez vous servir de la commande `plot_google_map`. Afin de vous aider à comprendre le fonctionnement de `plot_google_map`, un code exemple vous est fourni dans le fichier `affiche_pos_avion.m`.

## 6 Partie 3 – Affichage des trajectoires - Durée 3h

Sur <http://rtajan.vvv.enseirb-matmeca.fr/Cours/Telecom/T2/> vous pourrez télécharger des enregistrements durant 1 seconde utiliser ces enregistrements pour tester vos codes en situation réelle. Le code MATLAB nécessaire pour effectuer le travail de cette partie devra être nommé `ADSB_part3.m`.

Lorsque votre code fonctionne, appeler votre encadrant afin de mettre en place la chaîne en temps réel.

## 7 Contacts

- Guillaume Ferré - [guillaume.ferre@ims-bordeaux.fr](mailto:guillaume.ferre@ims-bordeaux.fr)
- Romain Tajan - [romain.tajan@ims-bordeaux.fr](mailto:romain.tajan@ims-bordeaux.fr)
- Baptiste Laport-Fauret - [baptiste.laporte-fauret@ims-bordeaux.fr](mailto:baptiste.laporte-fauret@ims-bordeaux.fr)

## Annexe A Tableaux de Format Type Codes

TABLE 3 – Tableaux des valeurs du FTC pour  $FTC \in [0, 8]$

TYPE Code	Format	Horizontal protection limit (HPL)	95% Containment radius, $\mu$ and $v$ , on horizontal and vertical position error	Altitude type (see §A.2.3.2.4)	NUC <sub>P</sub>
0	No position information			Barometric altitude or no altitude information	0
1	Identification (Category Set D)			Not applicable	
2	Identification (Category Set C)			Not applicable	
3	Identification (Category Set B)			Not applicable	
4	Identification (Category Set A)			Not applicable	
5	Surface position	HPL < 7.5 m	$\mu < 3$ m	No altitude information	9
6	Surface position	HPL < 25 m	$3 \text{ m} \leq \mu < 10$ m	No altitude information	8
7	Surface position	HPL < 185.2 m (0.1 NM)	$10 \text{ m} \leq \mu < 92.6$ m (0.05 NM)	No altitude information	7
8	Surface position	HPL > 185.2 m (0.1 NM)	(0.05 NM) $92.6 \text{ m} \leq \mu$	No altitude information	6

TABLE 4 – Tableaux des valeurs du FTC pour  $FTC \in [9, 18]$

9	Airborne position	HPL < 7.5 m	$\mu < 3$ m	Barometric altitude	9
10	Airborne position	$7.5 \text{ m} \leq \text{HPL} < 25$ m	$3 \text{ m} \leq \mu < 10$ m	Barometric altitude	8
11	Airborne position	$25 \text{ m} \leq \text{HPL} < 185.2$ m (0.1 NM)	$10 \text{ m} \leq \mu < 92.6$ m (0.05 NM)	Barometric altitude	7
12	Airborne position	$185.2 \text{ m (0.1 NM)} \leq \text{HPL} < 370.4$ m (0.2 NM)	$92.6 \text{ m (0.05 NM)} \leq \mu < 185.2$ m (0.1 NM)	Barometric altitude	6
13	Airborne position	$370.4 \text{ m (0.2 NM)} \leq \text{HPL} < 926$ m (0.5 NM)	$185.2 \text{ m (0.1 NM)} \leq \mu < 463$ m (0.25 NM)	Barometric altitude	5
14	Airborne position	$926 \text{ m (0.5 NM)} \leq \text{HPL} < 1\,852$ m (1.0 NM)	$463 \text{ m (0.25 NM)} \leq \mu < 926$ m (0.5 NM)	Barometric altitude	4
15	Airborne position	$1\,852 \text{ m (1.0 NM)} \leq \text{HPL} < 3\,704$ m (2.0 NM)	$926 \text{ m (0.5 NM)} \leq \mu < 1\,852$ m (1.0 NM)	Barometric altitude	3
16	Airborne position	$3.704 \text{ km (2.0 NM)} \leq \text{HPL} < 18.52$ km (10 NM)	$1.852 \text{ km (1.0 NM)} \leq \mu < 9.26$ km (5.0 NM)	Barometric altitude	2
17	Airborne position	$18.52 \text{ km (10 NM)} \leq \text{HPL} < 37.04$ km (20 NM)	$9.26 \text{ km (5.0 NM)} \leq \mu < 18.52$ km (10.0 NM)	Barometric altitude	1
18	Airborne position	HPL $\geq 37.04$ km (20 NM)	$18.52 \text{ km (10.0 NM)} \leq \mu$	Barometric altitude	0

TABLE 5 – Tableaux des valeurs du FTC pour  $FTC \in [19, 31]$

19	Airborne velocity	Not applicable	Not applicable	Difference between "Barometric altitude" and "GNSS height (HAE) or GNSS altitude (MSL)" (2.3.5.7)	N/A
20	Airborne position	HPL < 7.5 m	$\mu < 3$ m and $v < 4$ m	GNSS height (HAE)	9
21	Airborne position	HPL < 25 m	$\mu < 10$ m and $v < 15$ m	GNSS height (HAE)	8
22	Airborne position	HPL $\geq 25$ m	$\mu > 10$ m or $v \geq 15$ m	GNSS height (HAE)	0
23	Reserved for test purposes				
24	Reserved for surface system status				
25 – 27	Reserved				
28	Extended squitter aircraft emergency priority status				
29	Reserved				
30	Reserved				
31	Aircraft operational status				



## Annexe B Tableaux de structure des messages

TABLE 6 – Composition du message de position au sol

Index binaire	Champs
1	MSB
⋮	Format Type Code
5	LSB
6	MSB
⋮	Indicateur de mouvement
12	LSB
13	Statut
14	MSB
⋮	Latitude encodée avec CPR
20	LSB
21	Indicateur de temps UTC
22	Indicateur de format CPR
23	MSB
⋮	Latitude encodée avec CPR
39	LSB
40	MSB
⋮	Longitude encodée avec CPR
56	LSB

## Annexe C Encodage et décodage des altitudes, latitudes et longitudes

### Décodage de l'altitude

L'altitude est encodée dans un mot  $\mathbf{b}_a$  de 12 bits. Le 8<sup>me</sup> bits de  $\mathbf{b}_a$  étant inutile dans notre cas, il ne doit pas être considéré. Le registre obtenu finalement est le suivant

$$\mathbf{r}_a = [b_a^1, b_a^2, b_a^3, b_a^4, b_a^5, b_a^6, b_a^7, b_a^9, b_a^{10}, b_a^{11}, b_a^{12}].$$

On notera  $r_a$  la valeur entière non signée contenue de ce registre en considérant  $b_a^1$  comme bit de poids fort. La valeur de l'altitude *alt* exprimée *en pieds* est obtenue comme

$$\text{alt} = 25r_a - 1000. \quad (4)$$

### Décodage de la longitude et de la latitude

L'encodage des latitudes et longitudes dans les trames ADS-B est effectué suivant un format appelé Compact Position Reporting (CPR). Ce format permet d'encoder des positions précises sur peu de bits (34 bits seulement par trames de position en vol). L'encodage CPR admet intrinsèquement une imprécision de 180 nœuds nautiques par trames. Afin de décoder avec une plus grande précision les longitudes et latitudes, il est nécessaire d'avoir

- soit deux trames consécutives ayant des bits *indicateurs de format CPR* différents,
- soit une position de référence de l'appareil à moins de 180 nœuds nautiques.

Nous ne considérerons dans un premier temps que le second cas. Ce cas est justifié si on fait l'hypothèse que l'appareil détecté était proche de la position de l'antenne de réception.

Dans les messages de position en vol, la latitude et la longitude de l'appareil sont encodées sur des mots de 17 bits. Ces registres sont notés  $\mathbf{r}_{lon} = [r_{lon}^1, \dots, r_{lon}^{17}]$  et  $\mathbf{r}_{lat} = [r_{lat}^1, \dots, r_{lat}^{17}]$ . Aussi, nous noterons LAT et LON les valeurs entières contenues respectivement dans  $\mathbf{r}_{lon}$  et  $\mathbf{r}_{lat}$  avec  $r_{lon}^1$  et  $r_{lat}^1$  comme bits de poids fort.

Nous commencerons par le calcul de la **latitude**. Ce calcul comporte trois étapes.

1) calcul de la grandeur  $D_{lat_i}$  :

$$D_{lat_i} = \frac{360^\circ}{4N_Z - i} \quad (5)$$

où  $N_Z = 15$  est le nombre de latitudes géographiques considérées entre l'équateur et un pôle et  $i$  est le bit *indicateur de format CPR*.

2) Calcul de  $j$  :

$$j = \left\lfloor \frac{lat_{ref}}{D_{lat_i}} \right\rfloor + \left\lfloor \frac{1}{2} + \frac{\text{MOD}(lat_{ref}, D_{lat_i})}{D_{lat_i}} - \frac{\text{LAT}}{2^{N_b}} \right\rfloor \quad (6)$$

où  $N_b = 17$  est le nombre de bits constituant le registre de latitude  $\lfloor x \rfloor$  est la fonction renvoyant le plus petit entier  $k$  tel que  $k \leq x$  et  $\text{MOD}(x, y) = x - y \left\lfloor \frac{x}{y} \right\rfloor$ .

3) Calcul de la latitude  $lat$  :

$$lat = D_{lat_i} \left( j + \frac{\text{LAT}}{2^{N_b}} \right) \quad (7)$$

Nous finissons par le calcul de la **longitude**. Ce calcul comporte aussi trois étapes.

1) calcul de la grandeur  $D_{lon_i}$  :

$$D_{lon_i} = \begin{cases} \frac{360^\circ}{N_L(lat)-i} & \text{si } N_L(lat) - i > 0 \\ 360^\circ & \text{si } N_L(lat) - i = 0 \end{cases} \quad (8)$$

où  $i$  est le bit *indicateur de format CPR*,  $lat$  est la latitude calculée précédemment et  $N_L(x)$  est la fonction suivante

$$N_L(x) = \left\lfloor 2\pi \left[ \arccos \left( 1 - \frac{1 - \cos \left( \frac{\pi}{2N_Z} \right)}{\cos^2 \left( \frac{\pi}{180^\circ} |x| \right)} \right) \right]^{-1} \right\rfloor \quad (9)$$

pour la plupart des  $x$  à l'exception des points suivants :

$$\begin{aligned} N_L(0) &= 59 \\ N_L(87) &= 2 \\ N_L(87) &= 2 \\ N_L(x) &= 1 \text{ si } |x| > 87. \end{aligned}$$

Afin de garantir une implémentation efficace pour le temps réel, la fonction  $N_L(x)$  est en général tabulée. Pour vous permettre de gagner du temps, cette fonction est fournie dans l'archive `projet_adsb.zip` et se nomme `cprNL`.

2) Calcul de  $m$  :

$$m = \left\lfloor \frac{lon_{ref}}{D_{lon_i}} \right\rfloor + \left\lfloor \frac{1}{2} + \frac{\text{MOD}(lon_{ref}, D_{lon_i})}{D_{lon_i}} - \frac{\text{LON}}{2^{N_b}} \right\rfloor \quad (10)$$

où  $N_b = 17$  est le nombre de bits constituant le registre de latitude  $\lfloor x \rfloor$  est la fonction renvoyant le plus petit entier  $k$  tel que  $k \leq x$  et  $\text{MOD}(x, y) = x - y \left\lfloor \frac{x}{y} \right\rfloor$ .

3) Calcul de la longitude  $lon$  :

$$lon = D_{lon_i} \left( m + \frac{\text{LON}}{2^{N_b}} \right) \quad (11)$$

## Annexe D Tableaux des caractères

TABLE 7 – Tableaux des caractères

				$b_6$	0	0	1	1
				$b_5$	0	1	0	1
$b_4$	$b_3$	$b_2$	$b_1$					
0	0	0	0			P	SP	0
0	0	0	1		A	Q		1
0	0	1	0		B	R		2
0	0	1	1		C	S		3
0	1	0	0		D	T		4
0	1	0	1		E	U		5
0	1	1	0		F	V		6
0	1	1	1		G	W		7
1	0	0	0		H	X		8
1	0	0	1		I	Y		9
1	0	1	0		J	Z		
1	0	1	1		K			
1	1	0	0		L			
1	1	0	1		M			
1	1	1	0		N			
1	1	1	1		O			