# FbBalise – Gestion de Balise

FbBalise est un gestionnaire de balise basé sur un module microcontrôleur ESP32s3.




Il permet de générer des séquences programmées par script et enregistrées en mémoire flash. Une interface WEB permet créer ou modifier les scripts et configurer le système. La taille maximale d'un script est 4K caractères.

L'accès se fait par un ordinateur ou smartphone en se connectant au réseau Wifi « FbBalise » avec le mot de passe « 12345678 » par défaut. Le nom du point d'accès et son mot de passe sont modifiables par configuration.

FbBalise peut également être connecté à un réseau WiFi local.

Caractéristiques principales :

- Interface de configuration par navigateur (chrome, opéra, firefox) via Wifi
- Alimentation 7-18V (moins de 40mA à 14v sans Wifi)
- Modes CW, OPERA, PI4, FT8, JT4, JT9, JT65, Q65, MSK144, WSPR et JTTY
- Synchronisation du temps et calcul QTH locator par GPS
- Possibilité de piloter un DDS AD9954
- Dimensions : (cm) 5.25 x 7.15

# Table des matières


1. Nouveautés de la version 0.16.x ..... 4
2. Nouveautés de la version 0.15.0 ..... 5
3. Démarrage ..... 6
4. Page accueil ..... 7
5. Page Paramètres ..... 8
   5.1. Configuration WiFi ..... 9
      Paramètes WiFi ..... 9
      Option « Délai WiFi » ..... 9
      Activation ..... 10
      Accès par WiFi internet ..... 10
      Accès à la configuration par nom d'utilisateur / mot de passe ..... 10
   5.2. Paramètres Généraux ..... 11
      Callsign ..... 11
      Language ..... 11
      Sortie du signal KEY télégraphie (actif si SPI est OFF) ..... 11
      Inversion signal KEY télégraphie ..... 12
      Sortie 1PPS (actif si SPI est OFF) ..... 12
      Script 1 ..... 12
      Géné externe ..... 12
      Sortie audio ..... 12
   5.3. Configuration SPI ..... 13
   5.4. MQTT ..... 14
   5.5. Infos ..... 15
   5.6. Mise à jour ..... 16
   5.7. Déconnexion ..... 17
6. Mots clé du script ..... 18
   6.1. BREAK ..... 18
   6.2. DEBUG / ECHO ..... 18
   6.3. DELAY ..... 18
   6.4. HEX (à implémenter) ..... 19
   6.5. IF / ELSIF / ELSE / ENDIF ..... 20
   6.6. LOOP / ENDLOOP ..... 20
   6.7. MODE ..... 21
   6.8. RESTART ..... 21

6.9. SEND....23
6.10. SET....23
6.11. TUNE....23
6.12. TIMESTAMP....24
7. Variables....25
7.1. Déclaration et utilisation de variables....25
7.2. Variables prédéfinies....25
7.3. Variables de télémesure....26
7.4. Formatage d'une variable....27
8. Expressions....29
8.1. Fonctions....29
9. Comparaisons et opérateurs logiques....31
10. Réalisation du script....32
10.1. Création du script....32
10.2. Vérification et fonctionnement autonome....32
10.3. Script simple....32
10.4. Script plus évolué....33
10.5. Script avec test conditionnel....33
10.6. Script avec synchronisation à la minute....34
10.7. Script WSPR synchronisé....34
11. Générateur externe....36
12. Schéma électronique....37
13. Connecteurs et jumpers....38
14. Interface et carte DDS AD9954....39
15. Remerciements....41

# 1. Nouveautés de la version 0.16.x

**Nouveau mode**

- Nouveau mode numérique JTTY, issu de WSJT-X 3.2.0-rc1 : MODE JT JTTY [tonalité]. À la différence des autres modes JT (indicatif/locator ou messages types), JTTY transmet un texte libre, utilisable par exemple pour de la télémesure (cf. mot-clé SEND).
- Trame de 59 symboles (13 de synchronisation + 46 de données) à 31,25 bauds, soit environ 1,9s par trame de 5 caractères ; jusqu'à 16 trames (80 caractères) par envoi SEND.
- Tonalité de base 1500Hz par défaut, comme le WSPR.

Mode limité pour l'instant au DDS AD9954 (le LMX2571 n'est pas encore supporté pour ce mode).

**Télémesure GesBal**

- La balise reçoit par le réseau local les mesures de la carte GesBal qui lui est associée : tension, intensité, puissances directe et réfléchie, ROS et températures. Aucun réglage n'est nécessaire sur la balise : l'association se fait dans GesBal (version 0.2.6 ou suivante) avec l'adresse MAC de la balise.
- Nouvelles variables prédéfinies $_GV, $_GI, $_GP, $_GR, $_GS, $_GT, $_GU (carte associée), $_GV1 à $_GU3 (carte explicite), $_GBN (carte associée) et $_GAG (âge des données), utilisables dans SEND, SET, IF et la télémétrie : voir « Variables de télémesure GesBal ».
- Onglet Infos : nouvelle ligne « GesBal: » avec la carte associée, ses mesures et l'âge des données, affichée dès la première réception.
- La réception n'interfère pas avec l'émission : elle est traitée en tâche de fond de faible priorité, sans allocation mémoire ni écriture en flash.

**Mise à jour du logiciel**

- Chaque fichier .bin contient maintenant le nom du projet et sa version. Un binaire d'un autre projet ou compilé pour une autre puce est refusé, que la mise à jour se fasse par la page Mise à jour, par MQTT ou par le réseau (OTA).
- La page Mise à jour affiche l'identité du fichier choisi (nom, version, date de compilation) avant l'envoi. Un ancien binaire non identifié n'est accepté qu'en cochant « Forcer ».
- Une mise à jour refusée affiche la raison du refus et la balise continue de fonctionner sans redémarrer.
- MQTT : la commande « update force » permet de revenir à une ancienne version non identifiée.

# 2. Nouveautés de la version 0.15.0

**Général**

- La fréquence de la balise dans l'onglet SPI est maintenant la fréquence d'émission de la balise après la multiplication externe éventuelle.

**Éditeur de script**

- Indentation automatique : deux espaces par niveau de LOOP et de IF à l'affichage, curseur placé au bon niveau après Entrée, touche Tab pour tout réindenter. Les espaces de début de ligne ne sont plus enregistrés.
- Contrôle des blocs pendant la frappe (LOOP/IF non fermé, fin de bloc ou BREAK mal placé), signalé en orange.

**Langage de script**

- DELAY et TUNE en temps absolu : UNTIL s [EVERY p], en secondes UTC (exemple : DELAY UNTIL 1 EVERY 120 pour la minute paire).
- Nouveau mot-clé BREAK pour sortir d'une boucle.
- Fonctions dans les expressions : RND, INT, ROUND, ABS, MIN, MAX.
- Nouvelles variables prédéfinies $_TOD (secondes depuis minuit UTC) et $_DOW (jour de la semaine).
- Le nombre de tours de LOOP peut être une variable ou une expression ; LOOP 0 n'exécute pas la boucle.
- Le caractère # est accepté à l'intérieur d'un texte entre guillemets.

**Fiabilité de fonctionnement autonome**

- Le script est vérifié en entier à l'enregistrement et au lancement ; un script en erreur ne démarre pas.
- La dernière version valide de chaque script est conservée et lancée à la place d'une version enregistrée avec une erreur.
- Une fois lancé, le script ne s'arrête plus sur une erreur : la ligne concernée est ignorée et signalée une seule fois.
- $_SCRIPT ne change de script que si le script demandé est utilisable.
- Les variables sont remises à zéro à chaque lancement de script.
- DELAY et TUNE sont cadencés à la microseconde, sans limite de durée pratique.
- La configuration n'est réécrite en mémoire flash que si elle a changé.

**Corrections**

- ELSIF était exécuté même lorsqu'une branche précédente l'avait déjà été.
- Après la création de 10 variables, plus aucune variable ne pouvait être modifiée.
- Deux variables consécutives dans un texte ($A$B) n'étaient pas toutes deux remplacées, et un texte long contenant des variables pouvait déborder.
- Un mode inconnu (MODE XYZ) était accepté sans erreur.
- TIMESTAMP signalait une erreur de mode même dans une branche IF non exécutée.

# 3. Démarrage

FbBalise est configuré par des pages WEB accessibles par Wifi. A la mise sous tension, quelques secondes sont nécessaires pour que le Wifi soit activé.

Depuis votre ordinateur personnel ou votre smartphone, cherchez le réseau Wifi « FbBalise » et connectez-le.

Après la connexion au Wifi « FbBalise », entrez le mot de passe si nécessaire (« 12345678 » par défaut), une fenêtre d'accueil s'affiche au bout de quelques secondes. Cette fenêtre va permettre d'accéder par un menu aux différentes pages de l'application.

Si vous êtes bien connecté au Wifi « FbBalise » mais la fenêtre ne s'affiche pas, vous pouvez la connecter à l'adresse http://192.168.4.1

FbBalise utilise le port 80 pour le protocole http et pour le protocole ws.

Dans le cas ou FbBalise est connecté à un réseau local, et si vous voulez avoir un accès public, le port externe 80 devra être ouvert. Vous pouvez bien sûr rediriger un autre port externe vers le port 80 de FbBalise.

La LED tricolore donne une information sur le fonctionnement :

- 2 clignotements rouges : Phase d'initialisation et recherche des réseaux WiFi.
- Clignotements verts : carte en mode Point d'Accès WiFi
- Clignotements bleus : carte connectée au réseau WiFi.

Deux boutons :

- SW1 : reset.
- SW2 : boot. Un appui prolongé pendant la phase de fonctionnement (supérieur à 10s) efface le login mot de passe de la carte, en cas d'oubli du mot de passe. La prise en compte fait clignoter la LED rapidement en couleur blanche).

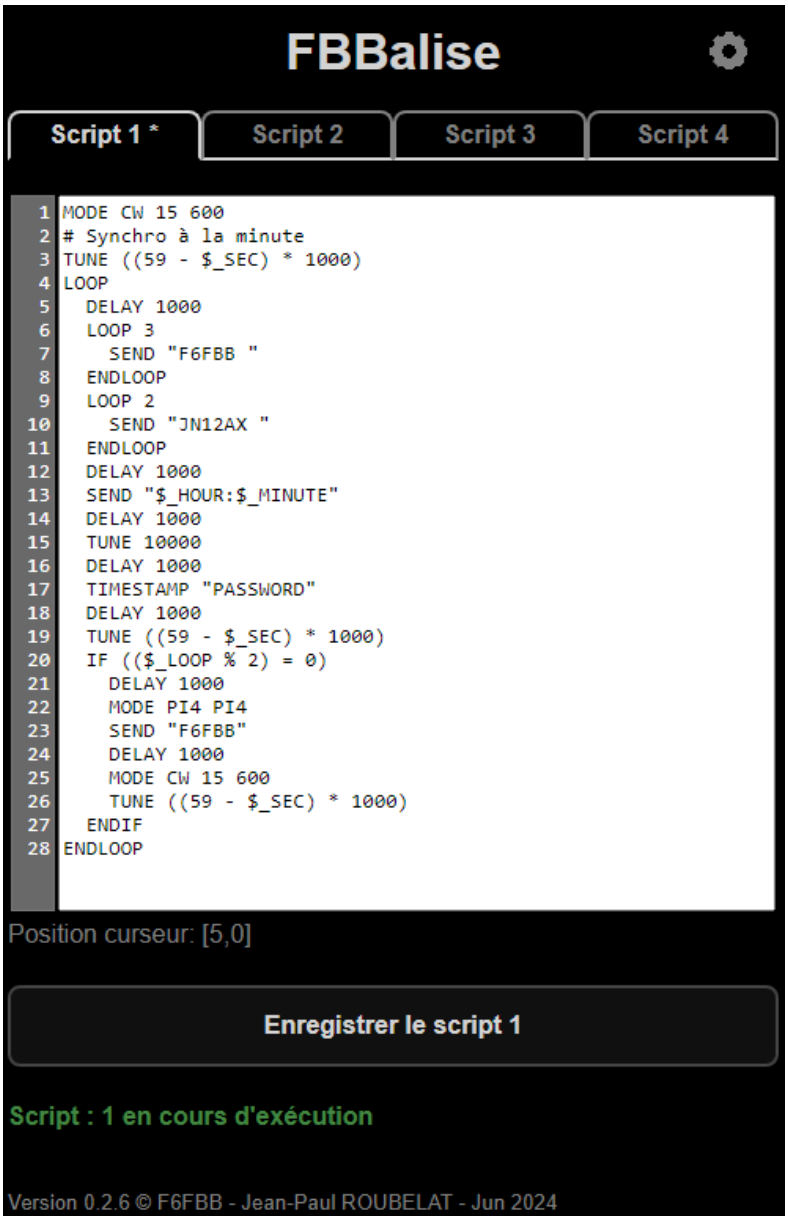
Deux prises antenne IPX/UFL :

- J1 : GPS.
- J3 : WiFi.

# 4. Page accueil

La page d'accueil propose un accès à :

- La configuration par la roue dentée en haut à droite de la fenêtre,
- Quatre onglets permettant de gérer et exécuter quatre scripts différents.
    - Si un nom d'onglet est suivi d'un astérisque, celui-ci a été modifié.
- Une fenêtre texte éditable qui permet de créer ou modifier le script, avec la position du curseur sous la fenêtre.
- Une ligne affichant un éventuel message d'erreur ou une ligne de débug.
- Un bouton permettant de démarrer, redémarrer ou sauvegarder un script suivant le contexte
- Une ligne de status donnant l'état d'exécution du script.



# 5. Page Paramètres

Un appui sur la roue dentée en haut à droite permet d'aller sur la page des paramètres.

Cette page permet de modifier la configuration par défaut et a quatre onglets :

- Configuration WiFi,
- Paramètres généraux,
- SPI
- mise à jour.



**Toute modification devra être enregistrée par le bouton « Validation »**

3 boutons en bas de page vous permettent de choisir un type de redémarrage :

- « Redémarrer » : le FbBalise est redémarré, les paramètres sont conservés, aucune modification n'est faite au PWS.
- « Effacer la flash » : les scripts contenus dans la mémoire flash sont effacés . Le FbBalise redémarre et la configuration est conservée.
- « Paramètres usine » : tous les paramètres et scripts sont effacés, le FbBalise redémarre en configuration par défaut.

## 5.1. Configuration WiFi




### Paramètes WiFi

- Nom du réseau WiFi (SSID) pour le point d'accès (AP).
- Mot de passe pour l'accès à ce réseau WiFi. Par défaut « 12345678 ».

### Option « Délai WiFi »

- Cette option n'est active qu'en mode point d'accès (AP). Si le FbBalise est connecté à un réseau WiFi, il ne sera pas déconnecté.
- Le Wifi s'active à la mise sous tension de FbBalise.
- Si l'option est active, le Wifi s'arrêtera automatiquement au bout du temps d'inactivité spécifié en minutes, uniquement en mode point d'accès (AP).
- Le délai minimal est de 5 minutes. Toute action de changement de page ou de sauvegarde redonne le délai spécifié.
- Si l'option est validée, le point d'accès WiFi ne sera de nouveau accessible qu'après coupure et remise sous tension ou dévalidation/validation par « activation ».

### Activation

- Il est possible de désactiver ou activer le point d'accès WiFi à l'aide d'un niveau 0 (désactivé) ou 3.3v (activé) sur l'entrée spare spécifiée. Si l'entrée spare n'est pas connectée, le point d'accès est activé par défaut.

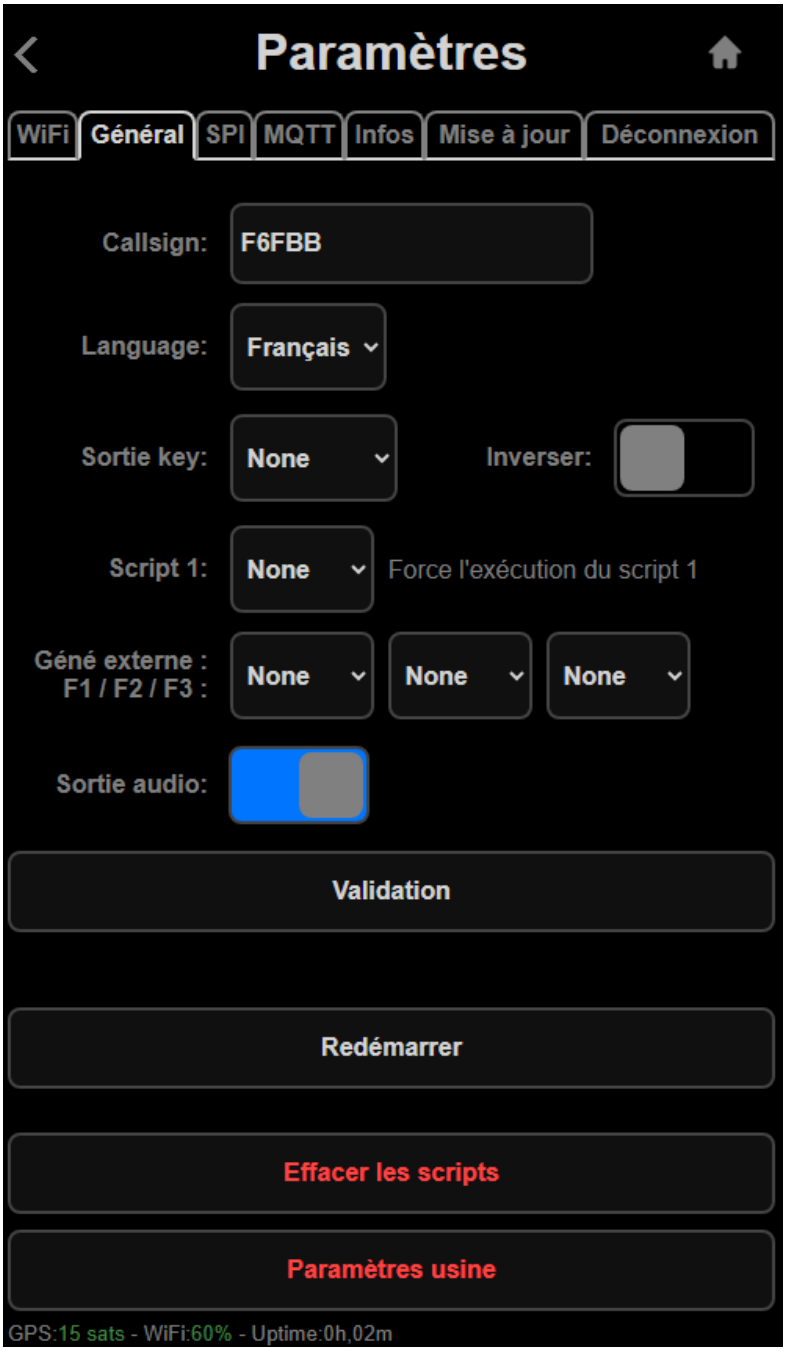
### Accès par WiFi internet

- Si un réseau WiFi et son mot de passe sont spécifiés, FbBalise tentera de se connecter à ce réseau. S'il échoue, il lancera son propre accès WiFi.

### Accès à la configuration par nom d'utilisateur / mot de passe

- Si ces champs sont renseignés, un nom d'utilisateur et un mot de passe seront demandés à la connexion.

## 5.2. Paramètres Généraux




### Callsign

- Indicatif par défaut de la balise

### Language

- Langue de l'interface

### Sortie du signal KEY télégraphie (actif si SPI est OFF)

- Sélection du port de sortie CW Key (spare 1 à 6).

### Inversion signal KEY télégraphie

- Inverse le signal de sortie CW.

### Sortie 1PPS (actif si SPI est OFF)

- Pulse de début de seconde généré par le GPS (spare 1 à 6).

### Script 1

- Permet de sélectionner un spare qui verrouillera la balise sur le script 1.

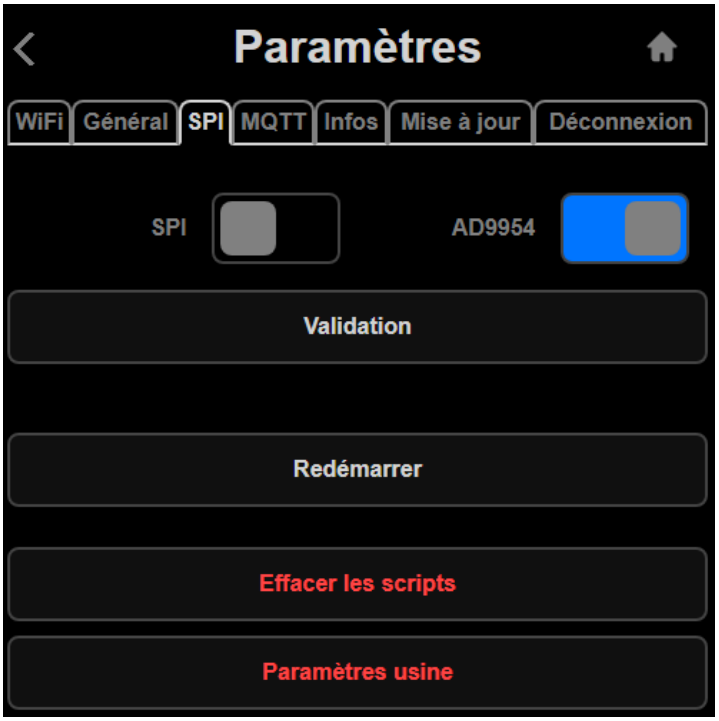
### Géné externe

- spécifie trois spares qui donneront la phase utilisée pour la configuration d'un générateur externe. Voir paragraphe Générateur externe.

### Sortie audio

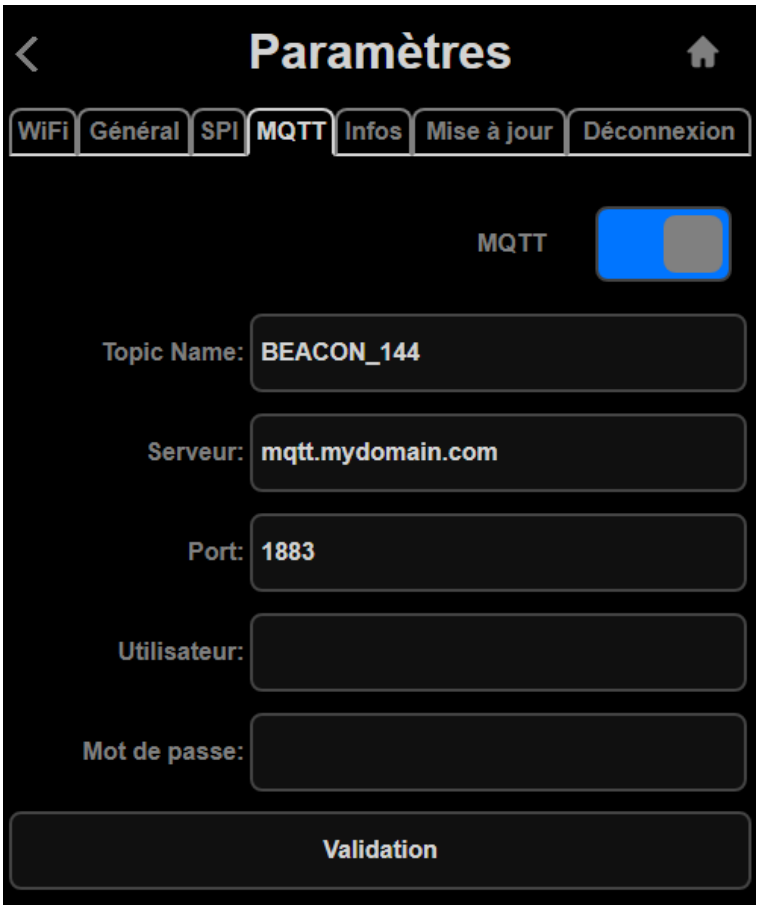
- Valide ou dévalide la génération du signal audio. Un redémarrage est nécessaire pour prendre en compte la modification.

## 5.3. Configuration SPI

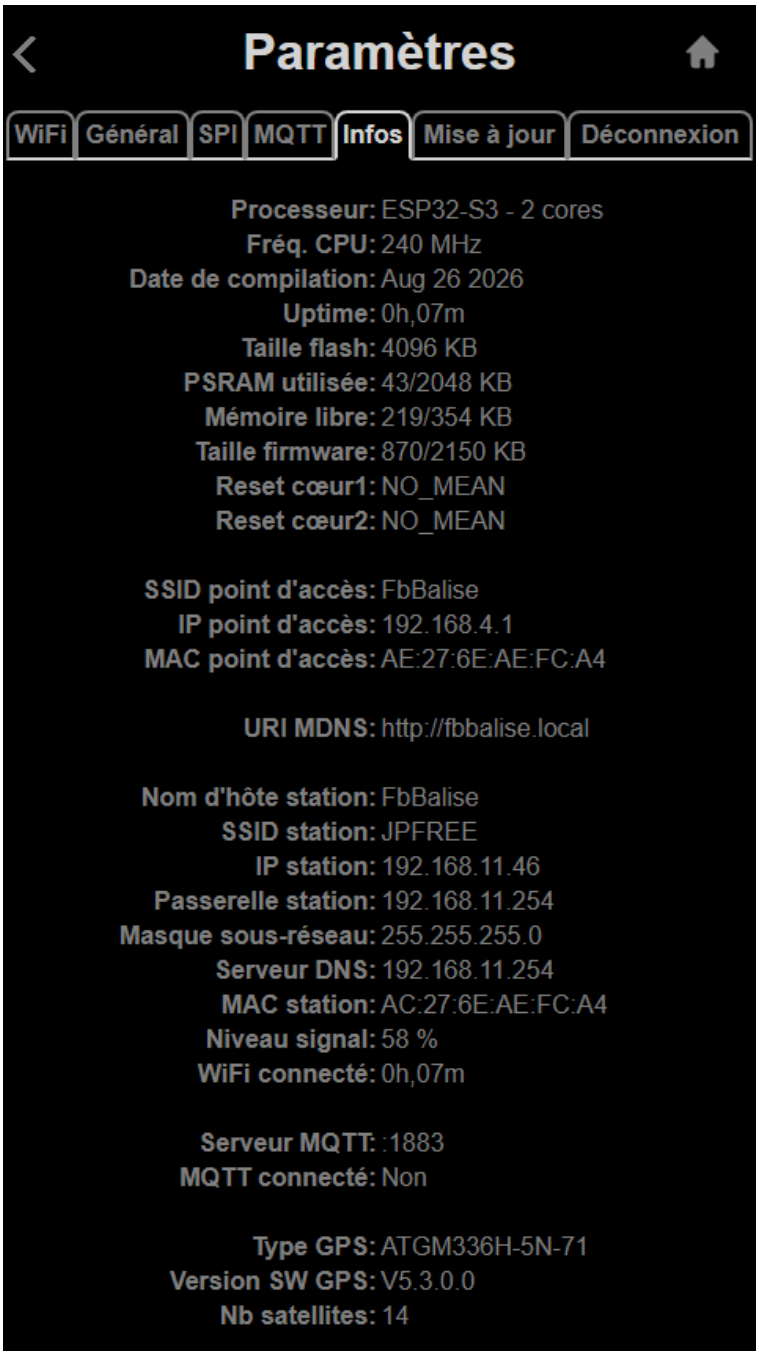



Si les deux switches SPI et AD9954 sont validés, La page de configuration SPI permet de programmer un DDS de type AD9954. Cf §12.

## 5.4. MQTT


Paramètres
WiFi
Général
SPI
MQTT
Infos
Mise à jour
Déconnexion
MQTT
Topic Name:
BEACON_144
Serveur:
mqtt.mydomain.com
Port:
1883
Utilisateur:
Mot de passe:
Validation

## 5.5. Infos




Donne les informations générales sur le système.

## 5.6. Mise à jour




Cette page permet de mettre à jour le logiciel avec une nouvelle version de logiciel. Le fichier est de la forme « nom_de_fichier.bin ».

Après avoir saisi le nom de fichier, appuyer sur le bouton « mettre à jour ». Le fichier sera transmis au module balise pour sa mise à jour. En fin de mise à jour, le module sera redémarré.

⚠ **Ne pas débrancher le module pendant la phase de mise à jour.**

Il est également possible de sauvegarder et restaurer la configuration dans un fichier JSON.

## 5.7. Déconnexion

Ferme la session. L’entrée du login et mot de passe sera nécessaire pour accéder de nouveau à l’interface WEB.

# 6. Mots clé du script

- Un paramètre entre crochets est optionnel
- Une expression peut être
  - une valeur numérique
  - une chaine de caractères entre guillemets
  - une expression entre parenthèses, contenant des expressions, variables, valeurs numériques, parenthèses et signes +, -, /, *, %
  - une condition entre parenthèses, contenant des opérations, variables, valeurs numériques, parenthèses et signes >, >=, <, <=, =
  - une fonction : RND, INT, ROUND, ABS, MIN, MAX (cf. paragraphe « Fonctions »)

## 6.1. BREAK

- BREAK n'a pas de paramètre.
- BREAK termine immédiatement la boucle LOOP la plus proche : l'exécution reprend après son ENDLOOP.
- BREAK est généralement placé dans un IF. En dehors d'une boucle, il est signalé comme une erreur à l'enregistrement du script.

```
LOOP  # attend que le GPS reçoive au moins 4 satellites
      IF ($_NBSATS > 3)
            BREAK
      ENDIF
      DELAY 1000
ENDLOOP
```

## 6.2. DEBUG / ECHO

- DEBUG a un paramètre texte affiché sur la page d'accueil, sous la fenêtre d'édition.
- ECHO est identique au mot-clé DEBUG mais n'affiche pas l'information DEBUG + N° de ligne.
- Utiliser des guillemets pour définir l'argument car il peut contenir des espaces.
- Si l'argument contient une variable, son contenu sera transformé et affiché en texte.
- Une variable peut être formatée cf Formatage d'une variable.
- Dans cet exemple suivant, on suppose un pont diviseur externe d'un rapport de 5.12. La variable sera toujours affichée avec une décimale.

```
DEBUG "TENSION: $($_A2 * 5.12):1 VOLTS UPTIME : $_UPTIME"
```

## 6.3. DELAY

- DELAY a un paramètre expression spécifiant le délai de pause en millisecondes
- DELAY peut aussi attendre un instant absolu, exprimé en secondes de l'heure UTC (synchronisée par le GPS), avec le mot UNTIL :
  - DELAY UNTIL s : attend la prochaine seconde s de la minute ($0 \leq s < 60$).
  - DELAY UNTIL s EVERY p : attend le prochain instant t (en secondes UTC) tel que t modulo p = s ($0 \leq s < p$). Par exemple EVERY 120 pour une minute paire (WSPR), EVERY 3600 pour le début de chaque heure.

    - s et p peuvent être des valeurs, des variables ou des expressions.
- Attention : sans UNTIL, la valeur est en millisecondes ; avec UNTIL, elle est en secondes.
- Le comportement est strict : si l'instant visé vient d'être dépassé, même de quelques millisecondes, l'attente se prolonge jusqu'à la période suivante. Il appartient au script de garder la marge nécessaire.
- Si l'heure n'est pas encore synchronisée par le GPS, un avertissement est affiché et l'attente se fait sur l'horloge interne.

```
DELAY (EXPRESSION)
DELAY UNTIL (EXPRESSION) [EVERY (EXPRESSION)]
DELAY UNTIL 10.5  # attend la seconde 10,5 de la minute
DELAY UNTIL 1 EVERY 120# attend la seconde 1 d'une minute paire
```

## 6.4. HEX (à implémenter)

- Fonction qui transforme un nombre ou expression en une chaine hexadécimale. Elle attend deux paramètres sous forme d'expression:
    - Un nombre dont seule la partie entière sera prise en compte
    - Le nombre de bits du nombre hexadécimal. Si le nombre est plus grand que cette longueur, il sera limité à cette longueur.

```
HEX((4.52*10), 12)          RETOURNE LA CHAINE « 02D »
```

- Cette fonction ne réserve pas le signe, c'est de la responsabilité de l'intégration. Donc en 12 bits, 4095 et -1 retourneront tous deux « FFF »

## 6.5. IF / ELSIF / ELSE / ENDIF

- IF ou ELSIF ont un paramètre entre parenthèses définissant la condition.
    - Si la condition est vraie, seule la partie IF/ELSIF sera exécutée, Un seul IF/ELSIF sera exécuté, si aucun IF ou ELSIF n'est vrai, le ELSE sera exécuté.
    - Il peut y avoir plusieurs blocs ELSIF.
    - Si la condition est égale à 0, seule la partie ELSE sera exécutée.
- Un ENDIF doit obligatoirement spécifier la fin du test conditionnel.
- Si la condition ne peut pas être évaluée à l'exécution (variable inconnue par exemple), elle est considérée comme fausse et le script continue.

```
IF (CONDITION)
      ...
ELSIF (CONDITION)
      ...
ELSE
      ...
ENDIF
```

## 6.6. LOOP / ENDLOOP

- LOOP peut avoir un paramètre optionnel spécifiant le nombre de boucles
- Si le nombre de boucles n'est pas spécifié, la boucle sera infinie
- Un ENDLOOP doit obligatoirement spécifier la fin de boucle.
- Le nombre de boucles peut être une valeur, une variable ou une expression. LOOP 0 n'exécute pas le contenu de la boucle.
- Si le nombre de boucles ne peut pas être évalué à l'exécution (variable inconnue, valeur négative), la boucle n'est pas exécutée et le script continue.
- Le mot-clé BREAK permet de sortir d'une boucle avant la fin.
- Les blocs LOOP et IF peuvent être imbriqués sur 10 niveaux au maximum.

```
LOOP [(EXPRESSION)]
      ...
ENDLOOP
```

## 6.7. MODE

- Mode attend un ou plusieurs paramètres spécifiant les options du mode
  - CW (vitesse en mots / minute) [tonalité]
    - Tonalite 800Hz par défaut
  - OPERA PERIOD [tonalité]
    - PERIOD parmi OPERA05, OPERA1, OPERA2, OPERA4, OPERA8, OPERA32, OPERA65, OPERA2H
    - Tonalite 1500Hz par défaut
  - OOK48 [tonalité]
    - Tonalite 800Hz par défaut
    - Nécessite FbBalise V3 (avec le signal GPS 1pps)
  - PI4 Type [tonalité]
    - TYPE parmi PI4, PI4-80, PI4-96, PI4-120
    - Tonalite de base 800Hz par défaut
  - JT Type [tonalité]
    - TYPE parmi
      - FT8, JT4A->JT4G, JT65A->JT65C, JT9A->JT9H
      - Q65-15A-> Q65-15C, Q65-30A-> Q65-30D, Q65-60A-> Q65->60E, Q65-120A-> Q65-120E, Q65-300A-> Q65-300E,
      - WSPR
      - JTTY
    - Tonalite de base 800Hz par défaut pour JT/Q65xx et 1500Hz pour WSPR et JTTY
    - JTTY est pour l'instant limité au DDS AD9954 (non supporté avec un LMX2571)
  - JT MSK144 [nb_de_trames]
    - Pas de configuation de tonalité. La fréquence centrale est de 1500Hz avec un décalage de +/- 500Hz.
    - Une trame dure 72ms, une transmission de 30 secondes demande environ 400 trames.



**Si le paramètre tonalité est spécifié, il sera pris en compte pour le décalage de fréquence (FSK) du mode sélectionné, au lieu du paramètre par défaut.**

```
MODE CW (EXPRESSION) [(EXPRESSION)]
MODE OPERA OPERA05 [(EXPRESSION)]
MODE OOK48 [(EXPRESSION)]
MODE PI4 PI4 [(EXPRESSION)]
MODE JT JT4G [(EXPRESSION)]
```

## 6.8. RESTART

- Ne nécessite pas de paramètre,
- Redémarre FbBalise. La configuration et les scripts sont conservés.

```
IF ($_UPTIME >= 24)          # Redémarre FbBalise toutes les 24 heures
  RESTART
```

```
ENDIF
```

## 6.9. SEND

- SEND permet d'envoyer un texte passé en argument en utilisant le mode précédemment défini par le mot-clé MODE.
- Utiliser des guillemets pour définir l'argument car il peut contenir des espaces.
- Si l'argument contient une variable, son contenu sera transformé en texte
- Une variable peut être formatée cf Formatage d'une variable.
- Le contenu peut être limité par le mode utilisé.
- Dans le cas de OPERA, seul un indicatif valide sera accepté en argument.
- Dans le cas du JT, l'indicatif et le qth locator peuvent être envoyés, les messages type 1, 2 ou 3 pour le WSPR, et un texte libre (jusqu'à 80 caractères) pour le JTTY.

```
SEND  "F6FBB JN12BT"
SEND  "TENSION $($_A2 * 4.1):1"      #  1 chiffre après la virgule
```

## 6.10. SET

- SET permet de définir ou redéfinir la valeur d'une variable.
- Une variable doit commencer par le caractère $.
- Une variable est exclusivement numérique.
- Une variable peut être utilisée dans une expression.
- SET a deux paramètres :
    - Un paramètre spécifiant le nom de la variable. Ce nom doit obligatoirement commencer par une consonne et être exclusivement alphanumérique.
    - un paramètre expression spécifiant la valeur numérique affecté à la variable

```
SET $NOMDEVAR ($AUTREVAR / 12.5)
SET $NOMDEVAR 12.5
SET $NOMDEVAR ($NOMDEVAR + 1)        # incrémente $NOMDEVAR
```

## 6.11. TUNE

- TUNE a un paramètre expression spécifiant le temps du signal en millisecondes
- TUNE accepte la même forme absolue que DELAY (UNTIL / EVERY, en secondes) : la porteuse est émise jusqu'à l'instant indiqué. Voir DELAY.

```
TUNE (EXPRESSION)
TUNE UNTIL (EXPRESSION) [EVERY (EXPRESSION)]
TUNE UNTIL 58      # porteuse jusqu'à la seconde 58
```

## 6.12. TIMESTAMP

- TIMESTAMP a un paramètre texte spécifiant le mot de passe utilisé pour le calcul de la clé. Ce mot de passe est **OBLIGATOIREMENT** en majuscules. Il suit la spécification de G4JNT.
- Le mode en cours doit être CW.
- Cette commande ne fonctionne que si le GPS est connecté.
- L'heure utilisée pour le calcul est l'heure GMT fournie par le GPS.
- Si l'heure n'est pas valide (i.e. absence de GPS) le message « TIME ? » sera transmis. Un avertissement est alors affiché et le script continue.

```
TIMESTAMP MOTDEPASSE
```

# 7. Variables

## 7.1. Déclaration et utilisation de variables

Les variables sont obligatoirement numériques. Elles sont définies ou redéfines par le mot-clé « SET ». Une variable spéciale **$(EXPRESSION)** n'a pas besoin de déclaration, contient l'expression à évaluer et représente son contenu.

Si la variable est utilisée dans une ligne « SEND », sa valeur sera éventuellement formatée et transformée en texte. Le nombre de variables est limité à 10. Les variables sont propres à chaque script : elles sont remises à zéro à chaque lancement du script.

## 7.2. Variables prédéfinies

Plusieurs variables relatives au système sont prédéfinies. Il s'agit toujours de variables numériques. Leur nom commence toujours par un underscore (tiret bas) excepté pour $(expression). Certaines sont en lecture seule (pas d'initialisation nécessaire), d'autres en écriture seule.

| Variable | Valeur | Commentaire | R/W |
|---|---|---|---|
| **$_I1 A $_I6** | 0 ou 1 | Etat du spare en entrée 0=LOW 1=HIGH | R |
| **$_O1 A $_O6** | 0 ou <> 0 | Positionne le spare 0=LOW sinon HIGH | W |
| **$_A1 A $_A3** | 0 à 3,3 | Tension en entrée sur l'analog en volts | R |
| **$_T0 A $_T9** | Float ou Texte | Télémétrie pour l'application windows | W |
| **$_PTT** | 0 ou 1(PTT on) | Positionne la sortie PTT | W |
| **$_HOUR** | 0 à 23 ou -1 | Donnera -1 si le GPS n'est pas actif | R |
| **$_MINUTE** | 0 à 59 ou -1 | Donnera -1 si le GPS n'est pas actif | R |
| **$_SECOND** | 0.000 à 59.999 ou -1 | Donnera -1 si le GPS n'est pas actif. Avec ms | R |
| **$_DAY** | 1 à 31 ou -1 | Donnera -1 si le GPS n'est pas actif | R |
| **$_MONTH** | 1 à 12 ou -1 | Donnera -1 si le GPS n'est pas actif | R |
| **$_YEAR** | yyyy ou -1 | Donnera -1 si le GPS n'est pas actif | R |
| **$_UPTIME** | Nombre d'heures | Nombre d'heures depuis le démarrage (flottant) | R |
| **$_UPSCRIPT** | Nombre d'heures | Nombre d'heures depuis le démarrage du script | R |
| **$_LOOP** | Nombre de boucles | Nombre d'itérations dans la boucle active | R |
| **$_POWER** | Tension | Donne la tension d'alimentation du module | R |
| **$_TEMPE** | Température | Ou -1 si le BME280 non connecté | R |
| **$_HUMID** | Humidité | Ou -1 si le BME280 non connecté | R |
| **$_BAROM** | Pression barom. | Ou -1 si le BME280 non connecté | R |
| **$_ALTITUDE** | Altitude (mètres) | Ou -1 si le BME280 non connecté | R |
| **$_BALTI** | Corr. altitude mètres | Correction de pression barométrique. | RW |
| **$_CALLSIGN** | Indicatif | Texte uniquement tel que défini dans params | R |
| **$_LOCATOR** | Qth Locator | Texte uniquement, calculé à partir du GPS | R |
| **$_TIMEST[.]** | Time stamp | Texte uniquement, Password entre crochets. | R |
| **$_NBSATS** | >= 0 | Nb de satellites GPS | R |
| **$_SCRIPT** | 1 à 4 | Retourne le script ou change le script courant. Le script demandé est vérifié avant le changement : s'il est inutilisable, le script courant continue. | RW |
| **$_TOD** | 0 à 86399.999 | Secondes depuis minuit UTC | R |
| **$_DOW** | 1 à 7 | Jour de la semaine UTC : 1=lundi, 7=dimanche | R |

⚠ **La variable $_TIMEST[PASSWORD] contient le timestamp au format spécifié par G4JNT, I.E. la minute en 2 chiffres, un espace et 3 lettres. Le password entre crochets est OBLIGATOIREMENT en caractères majuscules.**

⚠ **La variables $_BALTI a la valeur 0 par défaut, donc pas de correction de la pression. La valeur devra être initialisée avec l'altitude pour la correction.**

⚠ **Les variables $_TEMPE, $_HUMID et $_BAROM nécessitent la présence d'un BME280 connecté sur le port I2C. La détection d'adresse est automatique.**

⚠ **Les variables $_I1 à $_I6 et $_A1 à $_A3 positionnent le port spare en entrée. Les variables $_O1 à $_O 6 positionnent le port spare en sortie. Si le port spare concerné est utilisé pour un autre usage, cela peut affecter son fonctionnement.**

⚠ **La tension en entrée sur un port spare ne doit pas dépasser 3,3v sous peine de détruire le microcontroleur. Les mesures de tension supérieures peuvent se faire en utilisant un pont diviseur externe. La valeur peut alors être multipliée en utilisant une expression.**

Exemples :

```
SET $_O2 1   # Positionne le port spare 2 au niveau haut

SET $_O2 0   # Positionne le port spare 2 au niveau bas

IF ($_I1)    # Vrai si le port spare 1 est à niveau haut

IF ($_A3 > 2.1)    # Vrai si le port analog 3 est à une tension supérieure à 2,1v

SET $_T0 "TENSION BATTERIE: $_POWER:2,V"    # Envoie la tension d'alimentation vers
```
l'application Windows sur la télémétrie 0

## 7.3. Variables de télémesure

Lorsque la balise est installée avec un GesBal, celui-ci lui transmet périodiquement (toutes les 10 secondes par défaut) les mesures de ses cartes par le réseau local (UDP, port 5250). Ces trames ne sortent pas du réseau local.

La carte de mesure correspondant à la balise est choisie dans GesBal (onglet Station / cartes, champ « balise » de chaque carte) en indiquant l'adresse MAC de la balise, visible dans l'onglet Infos de FbBalise à la ligne « MAC station ». Aucun réglage n'est nécessaire sur la balise. Dès qu'une première trame est reçue, l'onglet Infos affiche une ligne « GesBal: » avec la carte associée, ses mesures et l'âge des données.

| Variable | Valeur | Commentaire | R/W |
|---|---|---|---|
| $_GV | Volts | Tension d'alimentation mesurée par la carte GesBal associée à la balise | R |
| $_GI | Ampères | Intensité consommée | R |
| $_GP | Watts | Puissance directe crête sur la période de mesure GesBal | R |
| $_GR | Watts | Puissance réfléchie crête sur la période de mesure GesBal | R |
| $_GS | 1 à 999 ou 0 | ROS calculé sur les valeurs crête. 0 si pas | R |

| Variable | Valeur | Commentaire | R/W |
|---|---|---|---|
| | | d'émission | |
| $_GT | Température | Sonde NTC1 de la carte (°C) | R |
| $_GU | Température | Sonde NTC2 de la carte (°C) | R |
| $_GT1 à $_GU3 | Température | Mêmes valeurs pour la carte 1, 2 ou 3 de GesBal, quelle que soit la carte associée à la balise | R |
| $_GBN | 0 à 3 | Numéro de la carte GesBal associée à la balise, 0 si aucune | R |
| $_GAG | Secondes | Âge des dernières données reçues de GesBal, 999999 si aucune donnée reçue | R |



**Si une valeur n'est pas disponible (GesBal absent, aucune carte associée à la balise, sonde absente), la variable vaut 0 et un avertissement est affiché une seule fois. Le script continue.**



**La variable $_GAG permet de vérifier que les données sont récentes avant de les utiliser, par exemple IF ($_GAG < 60). Les puissances et le ROS décrivent la dernière période de mesure de GesBal, donc l'émission précédente.**

Exemples :

**`SET $_T1 "U=$_GV:1V I=$_GI:2A P=$_GP:1W"`**# Tension, intensité et puissance vers l'application Windows sur la télémétrie 1

**`SEND "F6FBB $_GT:0C"`** # Insère la température NTC1 dans un message

**`IF (($_GAG < 60) & ($_GS > 3))`** # Vrai si les données sont récentes et le ROS supérieur à 3

## 7.4. Formatage d'une variable

Lors de l'utilisation d'une variable sous forme textuelle (dans DEBUG ou SEND par exemple), il est possible de formater la variable en utilisant le format suivant :

**`$VAR:N,S`**

- N est le nombre de décimales. La valeur sera arrondie en fonction de ce nombre.
- S est le caractère de séparation qui remplacera le point décimal.
- Le formatage est optionnel, mais devra être utilisé dans cet ordre

Exemples :

**`SET $PI 3.1415926536`** # Initialise la variable avec la valeur 3.1415926536

**`SEND "$PI:3,V"`** # Enverra le texte 3V142

**`SEND "$PI:,V"`** # Enverra le texte 3V14159

Variables tension d'alimentation et météo pour PI4:

```
SEND "/_PS$_POWER:1,V" # Envoie la tension 12.4v : '/_PS12V4'

SEND "/_HUMI$_HUMI:0"  # Envoie 'hygrométrie 56% : '/_HUMI56'

IF ($_TEMPE >= 0)          # Cas pour +26° et -12°
  SEND "/_TMPO$_TEMPE:0"       # Envoie la température 26° : '/_TMPO26'
ELSE
  SEND "/_TMNE$(-$_TEMPE):0" # Envoie la température -12° : '/_TMNE12'
ENDIF

SET $_BALTI 400            # Correction de la pression barom avec l'altitude (400m)
SEND "/_BARO$($_BAROM%100):0" # Envoie la pression 1016hPa : '/_BARO16'
```

# 8. Expressions

Une expression est de la forme « A opérateur B » où :

- A est une valeur, une variable ou une expression.
- Un opérateur du tableau ci-dessous,
- B est une valeur, une variable ou une expression.

. Les opérations sont réalisées en virgule flottante.

Les opérateurs sont :

| Opérateur | Action | Commentaire |
|---|---|---|
| **+** | Addition | |
| **-** | Soustraction | |
| ***** | Multiplication | |
| **/** | Division | Quotient de la division flottante |
| **%** | Modulo | Reste de la division flottante |

Les multiplications, divisions et modulo sont effectuées en priorité, de gauche à droite.

Les additions et soustractions sont effectuées en deuxième priorité, également de gauche à droite.

Une expression peut être entourée de parenthèses, auquel cas les expressions entre parenthèses sont effectuées en priorité.

Exemples :

**(3 + 5 * 2)** => 3 + 10 => 13

**((3 + 5) * 2)** => 8 * 2 => 16

**(3 * 5 + 2)** => 15 + 2 => 17

## 8.1. Fonctions

Une expression peut utiliser les fonctions suivantes. Leur nom peut être écrit en majuscules ou en minuscules.

- **RND(N)** : nombre aléatoire décimal, compris entre 0 (inclus) et n (exclu).
- **INT(X)** : partie entière de x (arrondi vers zéro).
- **ROUND(X)** : x arrondi à l'entier le plus proche.
- **ABS(X)** : valeur absolue de x.
- **MIN(A; B)** et **MAX(A; B)** : la plus petite / la plus grande des deux valeurs.

Les deux arguments de MIN et MAX sont séparés par un point-virgule, car la virgule est acceptée comme séparateur décimal (1,5 = 1.5). Un appel de fonction peut contenir des espaces sans être entouré de parenthèses.

Exemples :

```
SET $F (1400 + INT(RND(200)))          # fréquence audio aléatoire entre 1400 et 1599 Hz

SET $D MAX(0; $D - 1)    # décrémente $D sans descendre sous 0
```

# 9. Comparaisons et opérateurs logiques

Une comparaison est de la forme « A Comparateur B » où :

- A est une valeur, une variable ou une expression.
- B est une valeur, une variable ou une expression.

Les calculs sont faits en virgule flottante. L'égalité peut donc être difficile à trouver. Le résultat d'une comparaison est égal à 1.0 si le résultat est vrai sinon 0.0 si le résultat est faux.

Les opérateurs logiques fonctionnent également sur des valeurs égales à 0 (0 logique) ou différentes de 0 (1 logique)

Les opérateurs sont :

| Comparateur | Action | Commentaire |
|---|---|---|
| `<` | Inférieur | |
| `>` | Supérieur | |
| `<=` | Inférieur ou égal | |
| `>=` | Supérieur ou égal | |
| `!= ou <>` | Différent | |
| `== ou =` | Egalité | |
| `&` | Et logique | |
| `\|` | Ou logique | |
| `^` | Ou exclusif logique | |

Exemples :

```
IF ($_A3 > 2.1)    # Vrai si le port analog 3 est à une tension supérieure à 2,1v
```

Exécute si analog3 supérieur à 2.1v

```
ELSE
```

Exécute si le le port analog 3 est à une tension inférieure ou égale à à 2,1v

```
ENDIF
```

```
IF ($_T & 2.1)     # Vrai si la valeur de $_T est différente de 0
```

# 10. Réalisation du script

## 10.1. Création du script

Le script est entré au clavier (ou collé) dans la fenêtre d'édition. Lorsque l'édition est terminée, le bouton de sauvegarde va envoyer et sauvegarder le script dans la flash de l'ESP32.

- Le script sera interprété en majuscules et ne fait pas de différences entre majuscules et minuscules.
- Toute ligne devra commencer par un mot clé, suivi de ses arguments éventuels.
- Toute ligne vide est ignorée.
- Toute fin de ligne sera ignorée après le caractère # (sauf à l'intérieur d'un texte entre guillemets).
- L'indentation est automatique : il n'est pas nécessaire de saisir des espaces ou des tabulations en début de ligne. L'éditeur affiche deux espaces par niveau de LOOP et de IF, et place le curseur au bon niveau après la touche Entrée.
- La touche Tab réindente tout le script ; un texte collé est réindenté automatiquement. Les espaces et tabulations de début de ligne ne sont pas enregistrés.
- Pendant la frappe, l'éditeur signale en orange un bloc LOOP/IF non fermé, une fin de bloc sans LOOP/IF ou un BREAK hors d'une boucle.

## 10.2. Vérification et fonctionnement autonome

Le script est vérifié en entier, sans être exécuté, à chaque enregistrement et à chaque lancement : syntaxe, mots-clés, nombre de paramètres, modes, structure des blocs LOOP/IF, expressions, fonctions et variables prédéfinies.

- À l'enregistrement, le script est toujours sauvegardé ; la première erreur trouvée est affichée sous la fenêtre d'édition.
- Au lancement, un script en erreur ne démarre pas. Si une version correcte de ce script a déjà été enregistrée ou lancée, c'est cette dernière version valide qui est exécutée, avec un avertissement indiquant l'erreur. Ainsi, après une coupure d'alimentation, la balise émet toujours, même si un script en cours d'écriture a été enregistré avec une erreur.
- Une fois lancé, le script ne s'arrête plus sur une erreur : une ligne qui ne peut pas être exécutée (variable inconnue, mode non défini…) est ignorée, son message n'est affiché qu'une seule fois, et le script continue.
- Les conditions extérieures au script (heure GPS pas encore valide, port spare occupé) sont signalées par un avertissement (« Warning ») et n'interrompent pas l'exécution.
- Le changement de script par $_SCRIPT n'a lieu que si le script demandé (ou sa dernière version valide) est correct ; sinon le script courant continue.
- Les variables sont propres à chaque script et sont remises à zéro à chaque lancement.

## 10.3. Script simple

Ce script va permettre de :

- Sélectionner le mode CW à 15 mots/minute, son à 800Hz
- Envoyer l'indicatif et le qth locator suivi d'une pause de 1s, tune de 5s, pause de 1s
- Le tout en boucle infinie.

```
MODE CW 15 800
LOOP
      SEND  "F6FBB JN12BT"
      DELAY 1000
      TUNE  5000
      DELAY 1000
ENDLOOP
```

## 10.4. Script plus évolué

Ce script va générer :

- Un mode CW 15 mots minute à 800Hz,
- trois fois l'indicatif, le qth locator et une pause de 500ms,
- ensuite, pause de 500ms puis tune de 5 secondes puis pause de 1 seconde
- il bouclera indéfiniment…

```
MODE CW 15 800
LOOP
      LOOP 3
            SEND  "F6FBB JN12BT"
            DELAY 500
      ENDLOOP
      DELAY 500
      TUNE  5000
      DELAY 1000
ENDLOOP
```

## 10.5. Script avec test conditionnel

Ce script va générer :

- Un mode CW 15 mots minute à 800Hz,
- trois fois l'indicatif, une pause de 1s puis deux fois le qth locator,
- ensuite, pause de 500ms puis tune de 5 secondes puis pause de 1 seconde
- Toutes les 4 boucles il enverra l'indicatif en mode OPERA puis retour en mode CW
- il bouclera indéfiniment…

```
MODE CW 15 800
LOOP
      LOOP 3
            SEND "F6FBB "
      ENDLOOP
      DELAY 1000
      LOOP 2
```

```
            SEND "JN12BT "
      ENDLOOP
      DELAY 1000
      TUNE  5000
      IF (($_LOOP % 4) = 0)
            DELAY 1000
            MODE OPERA OPERA05 1500
            SEND "F6FBB"
            MODE CW 15 800
      ENDIF
      DELAY 1000
ENDLOOP
```

## 10.6. Script avec synchronisation à la minute

Ce script va permettre d'envoyer 3 fois l'indicatif suivis d'une pause d'une seconde, puis un tune jusqu'à la seconde 59, faire une pause de 1 seconde puis d'envoyer l'indicatif en mode OPERA . Il nécessite la présence du module GPS pour avoir l'information horaire.

```
MODE CW 15 800
LOOP
      LOOP 3
            SEND "F6FBB "
      ENDLOOP
      DELAY 1000
      TUNE UNTIL 59
      DELAY 1000
      MODE OPERA OPERA05 1500
      SEND "F6FBB"
      DELAY 1000
      MODE CW 15 800
ENDLOOP
```

## 10.7. Script WSPR synchronisé

Ce script attend que le GPS reçoive au moins 4 satellites, puis, à chaque minute paire, envoie une trame WSPR sur une fréquence audio tirée au hasard, l'indicatif en CW, et une porteuse jusqu'à 2 secondes avant la minute paire suivante.

```
MODE CW 15 800
LOOP  # attente du GPS
      IF ($_NBSATS > 3)
            BREAK
      ENDIF
      DELAY 1000
ENDLOOP
LOOP
      DELAY UNTIL 1 EVERY 120# seconde 1 d'une minute paire
      MODE JT WSPR (1400 + INT(RND(200)))
      SEND "F6FBB JN12 23"
```

```
        MODE CW 15 800
        SEND "F6FBB"
        TUNE UNTIL 118 EVERY 120        # porteuse jusqu'à 2 s avant la minute
paire
ENDLOOP
```

# 11. Générateur externe

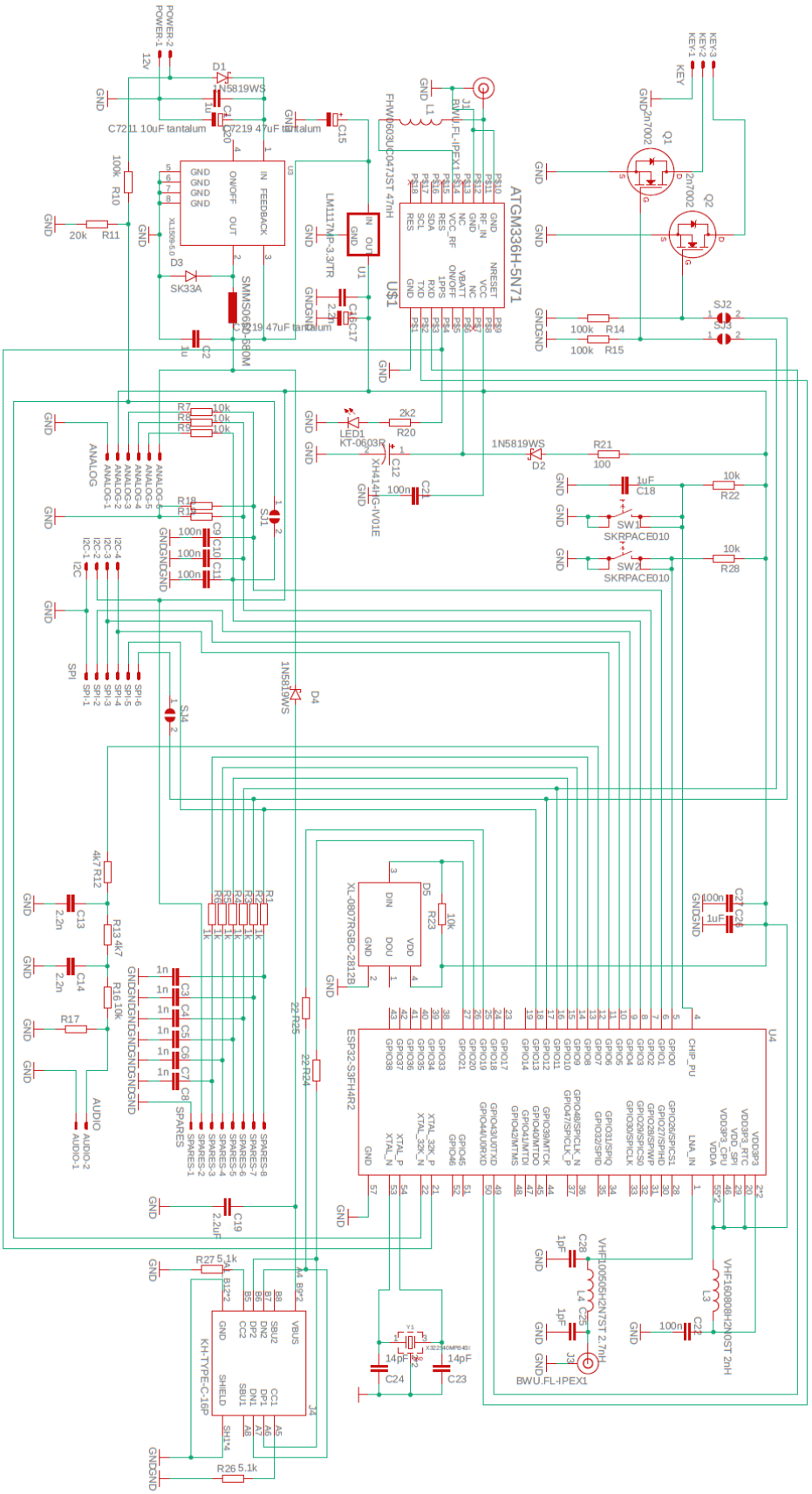
Dans l'onglet « Fréquences » du menu configuration, il est possible de sélectionner 3 spares pour activer un générateur externe OOK ou de fréquences.

Le niveau des spares attribués à F1, F2 et F3 suivra la logique suivante :

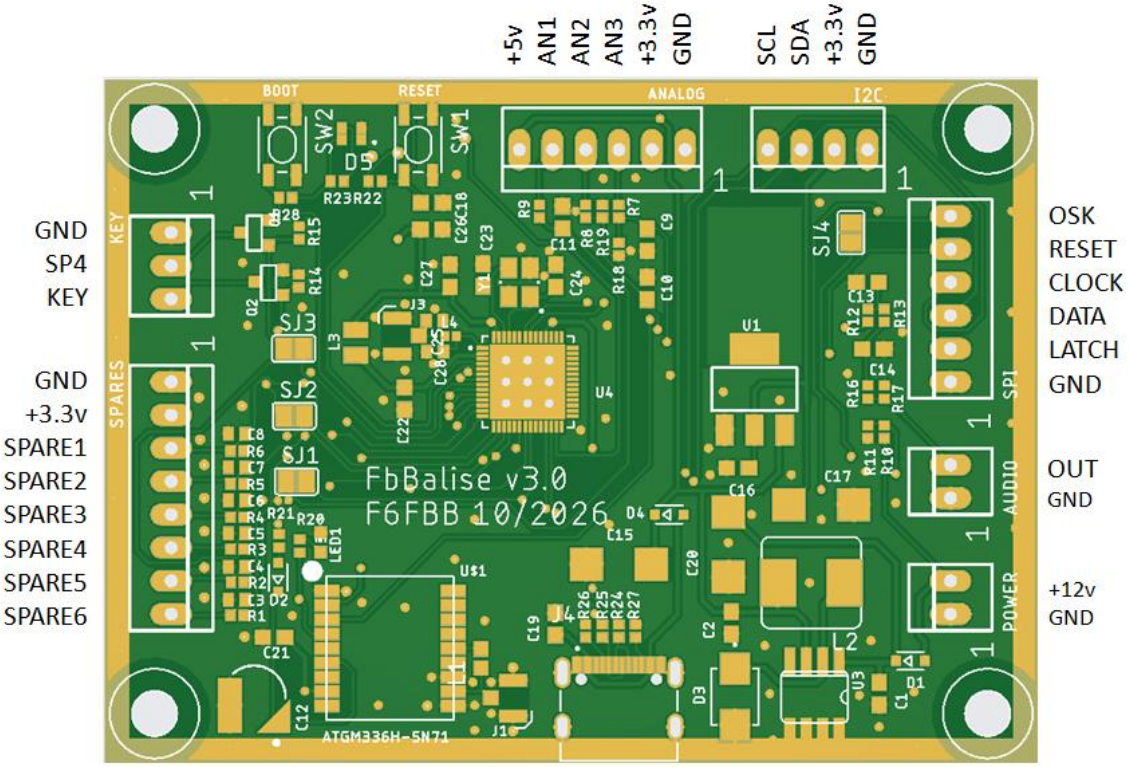
| F3 | F2 | F1 | Fréquence à générer | |
|---|---|---|---|---|
| 0 | 0 | 0 | CW_OFF / SPACE | Génération OOK ou shift suivant hardware |
| 0 | 0 | 1 | CW_ON / MARK | Génération OOK ou shift suivant hardware |
| 0 | 1 | 0 | OPERA_OFF / SPACE | Génération OOK ou shift suivant hardware |
| 0 | 1 | 1 | OPERA_ON / MARK | Génération OOK ou shift suivant hardware |
| 1 | 0 | 0 | PI4/0 | PI4 fréquence 0 |
| 1 | 0 | 1 | PI4/1 | PI4 fréquence 1 |
| 1 | 1 | 0 | PI4/2 | PI4 fréquence 2 |
| 1 | 1 | 1 | PI4/3 | PI4 fréquence 3 |

- La sélection des 3 spares n'est pas obligatoire, Il est possible de sélectionner tout ou partie de ces spares.,
    - F1 => « KeyUp / KeyDown. » dans le cas d'une balise CW uniquement
    - F2 => CW /OPERA si PI4 n'est pas utilisé
    - F3 => mode PI4

# 12. Schéma électronique



# 13. Connecteurs et jumpers




SJ1 : mesure de la tension d'alimentation sur AN1

SJ2 : sortie OOK open drain

SJ3 : sortie PTT open drain

SJ4 : sortie OOK SPI

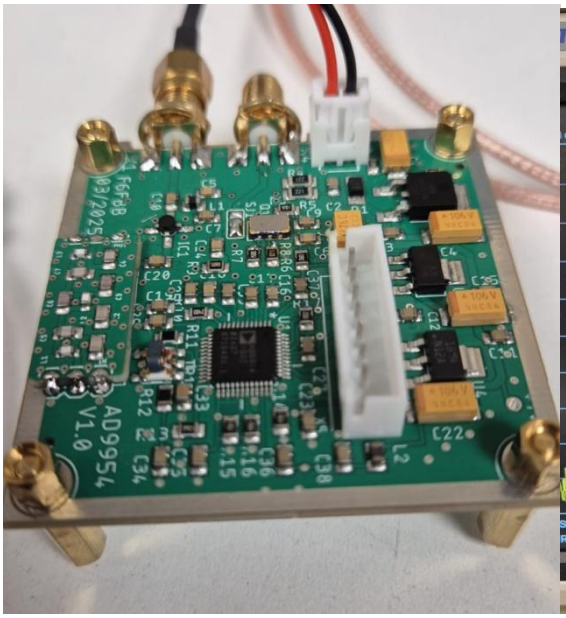
J1 : antenne GPS

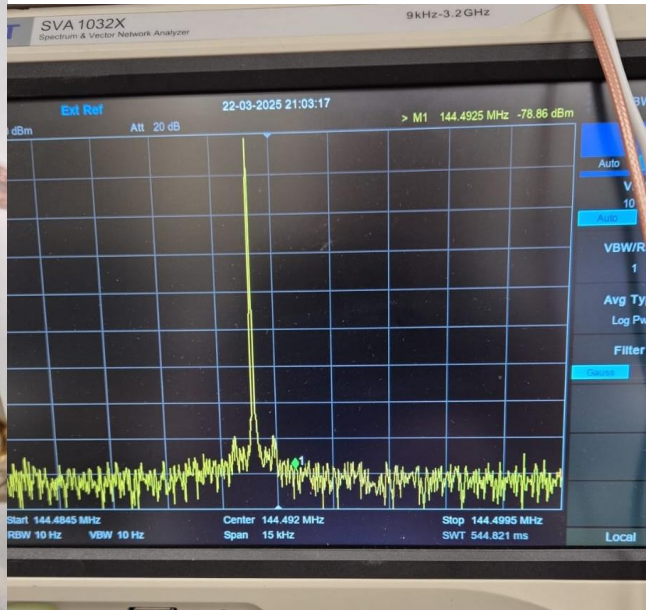
J3 : Antenne WiFi

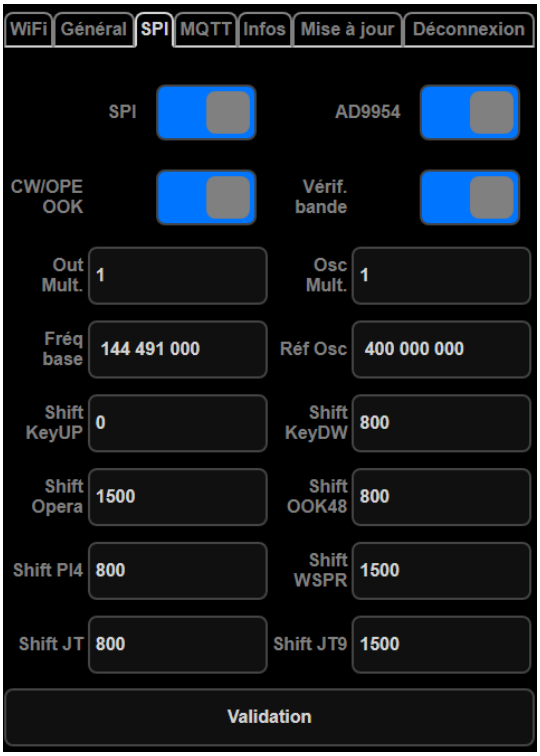
# 14. Interface et carte DDS AD9954

Une carte basée sur le circuit DDS AD9954 de Analog Devices peut être directement pilotée via le bus SPI.

Si le switch SPI et le switch AD9954 sont activés, une fenêtre spécifique permet de configurer le circuit et de générer le signal HF en FSK ou OOK suivant les modes et configurations.



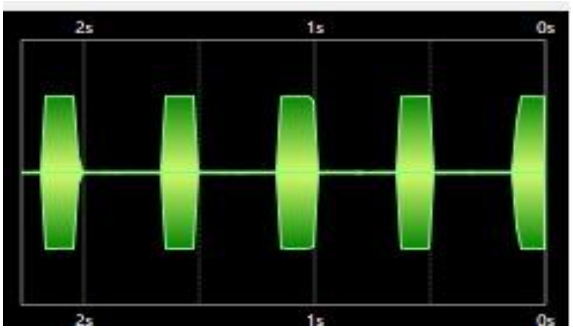




La valeur de la fréquence réelle de la balise doit être renseignée. C'est la fréquence DDS multipliée par Out Mult.

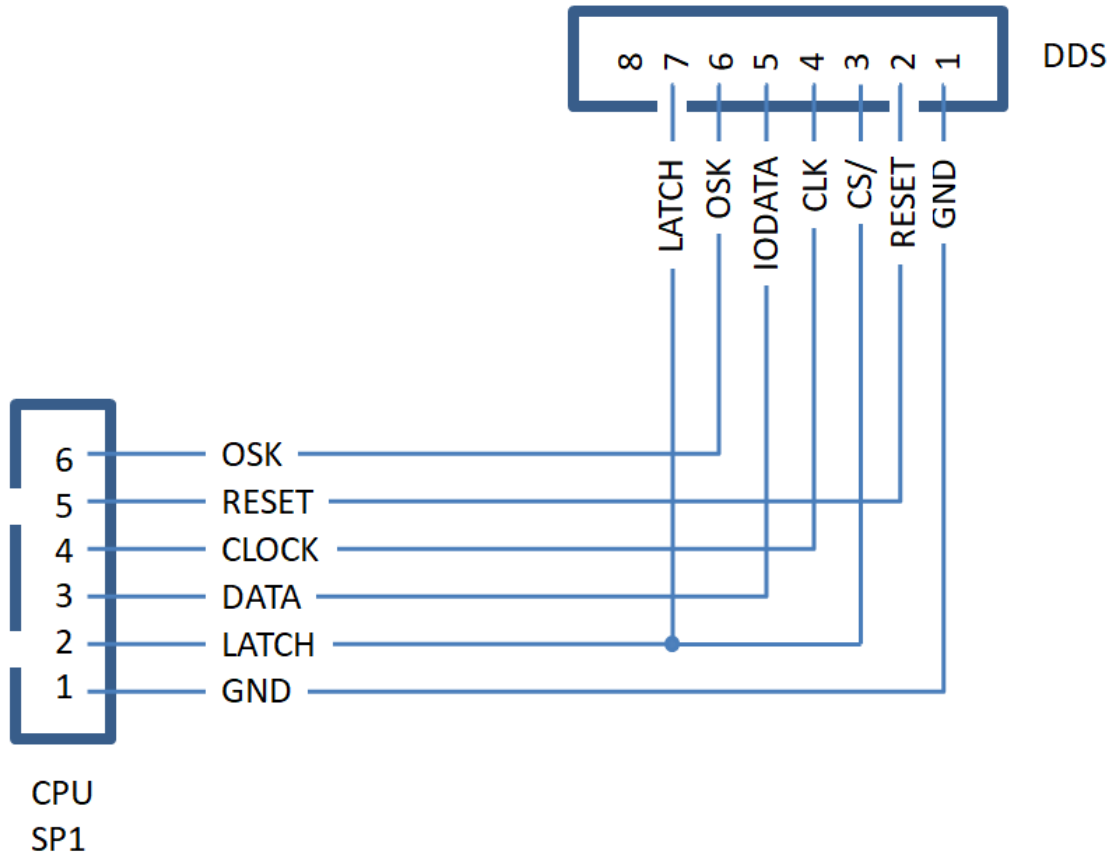
Toutes les valeurs de shift et de fréquences sont exprimées en Hz.

Le « Osc Mult. » (1 ou 4 à 10) permet de définir la multiplication de la fréquence de référence pour approcher « Ref Osc » (400MHz par défaut).

Le « Out Mult. » permet de calculer la fréquence du DDS en fonction de la fréquence balise et d'adapter les valeurs de décalage des modes de modulation en fonction d'une multiplication de la fréquence générée. I.E. dans le cas d'une génération 144MHz multipliée par 3 pour générer du 432MHz, le décalage de 315Hz en JT4G sera ramené à 105Hz, de façon à retrouver la valeur nominale après les étages de multiplication de l'émetteur. Les autres modes FSK auront le même fonctionnement.




Génération de la modulation OOK avec fronts « anti-claquements »




Câble FbBalise vers connecteur DDS

# 15. Remerciements

La réalisation logicielle est de ma conception. Ce logiciel fonctionne sur une carte FbBalise et pourra utiliser une carte fille GPS (pour la mise à l'heure et la localisation)

Le logiciel est écrit en C/C++ sous l'environnement « platformio / vscode » et inclut plusieurs librairies tierces ESP32 .

Merci à F5DJL et F1HDI pour leurs idées, remarques et soutien !

Jean-Paul ROUBELAT, F6FBB.