

Decode_sarsat_406 — contenu du répertoire (F4KLO / Gpredict)

6 octobre 2026 · Bernard Pidoux, F6BVP

Vue d'ensemble

~/Decode_sarsat_406 est un clone git du décodeur **dec406** de moricef (balises COSPAS-SARSAT de 1re et 2e génération), auquel vous avez ajouté une cinquantaine de fichiers personnels : scripts de lancement, journaux de réception et captures d'écran.

Vous n'écoutez pas les balises directement sur 406 MHz : vous recevez leur relais par satellite sur la liaison descendante en bande L, entre 1544,0 et 1545,1 MHz, avec un Airspy R2 ou une clé RTL-SDR.

La station est celle du radio-club F4KLO : elle utilise l'antenne du radiotélescope de la Villette, administré par votre association de radioamateurs. Elle sert à tester le décodage en écoutant les satellites, pas à assurer une veille de détresse.

Famille	Où	Origine
Programmes compilés	build/	projet amont, compilés ici
Sources, documentation	src/, include/, utils/, docs/...	projet amont
Scripts de lancement	racine, sans extension ou .sh	vous
Journaux et résultats	racine, Balises_*.txt, captures	vous

État relevé le 6 octobre 2026 :

- Taille : 125 Mo, dont 117 Mo pour l'historique .git ; le reste pèse moins de 8 Mo.
- Dépôt : branche main, commit 3a0d51d du 16 septembre, à jour avec origin (moricef) au 6 octobre. Un second dépôt distant, myfork, pointe vers votre fork GitHub f6bvp, mis à jour au même commit.
- Dernière activité : 11 août 2026 (Balises_sarsat_meo.txt).
- Aucun fichier de capture .iq ne subsiste dans le répertoire : seuls les scripts qui les produisaient sont restés.
- Les outils externes sont installés : airspy_rx et sendemail. Le service satdump est actif.

Tableau explicatif des fichiers

Les fichiers sont regroupés par usage ; le programme central est build/dec406_scan, que presque tous vos scripts appellent.

Programmes compilés (build/)

Programme	Rôle
dec406_scan	Scanner temps réel : détecte les salves, les classe 1G ou 2G, les décode. Recompilé le 6 octobre. C'est celui à utiliser.
dec406_scan_airspy	Ancien scanner réservé à l'Airspy (19 juillet). Un lien du même nom existe à la racine.
dec406_iq	Décode les balises 2G depuis un fichier de capture IQ.
dec406_fgb_iq	Décode les balises 1G depuis un fichier IQ.
dec406_hex	Décode une trame 1G ou 2G donnée en hexadécimal.

Programme	Rôle
dec406_audio	Décode le 1G depuis un fichier WAV (ancienne chaîne FM).
dec406_dsss_test	Test interne du démodulateur 2G.
reset_usb	Réinitialise un périphérique USB (clé RTL bloquée).

Vos scripts de scan en temps réel

Tous lancent `dec406_scan <début> <fin> <gain> 0`. Le gain 0 laisse le réglage par défaut ; sans mention contraire le scanner prend l'Airspy s'il est branché.

Script	Bande (MHz)	Gain	Journal
acq_sarsat	1544,0 - 1544,6	21	Balises_sarsat.txt
acq_sarsat_leo_Airspy-SDR	1544,0 - 1544,15	16	Balises_sarsat_leo.txt
acq_sarsat_leo_RTL-SDR	1544,0 - 1544,15	0	Balises_sarsat_leo2.txt
acq_sarsat_meo_Airspy-SDR	1544,4 - 1544,6	21	Balises_sarsat_meo.txt
acq_sarsat_1545_Airspy	1544,85 - 1545,1	21	Balises_sarsat_1545.txt
acq_sarsat_1545_RTL	1544,85 - 1545,1	49,6	Balises_sarsat_1545.txt
start_GPS	1544,0 - 1544,3	0	Balises_sarsat_GPS.txt
start	appelle sarsat		Balises_sarsat.txt
sarsat	1544,45 - 1544,5	50	écran seul
sarsat_Airspy	1544,0 - 1544,15	18	écran seul
sarsat_leo	1544,0 - 1544,2	50	écran seul
leosar_Airspy, leosar_Airspy2	1544,0 - 1544,1	21	écran seul
leosar_RTL	1544,0 - 1544,1	50	écran seul
sarsat_surveill.sh	1544,0 - 1544,2	0	écran seul
sarsat_RTL_surveill.sh	1544,0 - 1544,2	0	écran seul
sarsat_surveill_1544.9.sh, gps_sarsat.sh	1544,8 - 1545,0	0	écran seul
scan	1544,4 - 1544,58	21	écran seul

À savoir sur ces scripts :

- `acq_sarsat_leo_Airspy-SDR` et `acq_sarsat_meo_Airspy-SDR` activent `DEC406_ALERT_TEST=1` : chaque balise décodée envoie un courriel marqué [TEST].
- Seul `sarsat_RTL_surveill.sh` impose la clé RTL (`DEC406_BACKEND=rtl`). Les autres scripts nommés « RTL » ne changent que le gain.
- `scan` est le seul à utiliser l'ancien `dec406_scan_airspy`.
- `leosar_Airspy` et `leosar_Airspy2` sont identiques, de même que `sarsat_surveill_1544.9.sh` et `gps_sarsat.sh`.
- Les journaux `_leo2`, `_1545` et `_GPS` n'existent pas dans le répertoire.

Vos scripts de capture IQ puis décodage différé

`airspy_rx` enregistre à 2,5 Méc/s en entiers 16 bits ; 100 millions d'échantillons font 40 s et environ 400 Mo par fichier.

Script	Fréquence	Ce qu'il fait
bursts_2	1544,1	1 capture de 120 s, puis décodage
bursts_3	1544,1	1 capture sans limite (Ctrl-C), puis décodage
burst_4_1544.9	1544,9	1 capture de 20 s, puis décodage
bursts_Airspy_1544.5	1544,5	6 captures (4 s puis 5 × 40 s), sans décodage
bursts_Airspy_leo	1544,05	6 captures de 40 s (n° 0 à 5), décode la n° 1
bursts_Airspy_leo2	1544,08	captures n° 0 et 5 à 9, sans décodage
bursts_Airspy_leo3	1544,08	captures n° 0 et 10 à 14, sans décodage
bursts_Airspy_leo_Hgain	1544,05	captures n° 6 à 11 en gain « sensibilité » (-h 21), décode la n° 7
decode_bursts_Airspy_1544.5		décode capture-2_1544_5.iq
decode_bursts_Airspy_leo		décode les captures leo n° 10 à 14 et 5
decode_bursts_Airspy_leo-1-5		décode les captures leo n° 6 à 9

Deux pièges : les fichiers .iq ont été supprimés, donc les scripts decode_* ne trouvent plus rien ; et decode_bursts_Airspy_1544.5 attend capture-2_1544_5.iq alors que le script de capture écrit capture-2_1544.5.iq.

Journaux et résultats

Fichier	Contenu
Balises_sarsat.txt	Journal principal, 529 Ko, jusqu'au 10 août : 696 salves détectées, 79 lignes d'identifiant de balise décodée.
Balises_sarsat_meo.txt	Journal 1544,4 - 1544,6, 72 Ko, 11 août : 7 salves, aucune balise décodée.
Balises_sarsat_leo.txt	16 lignes : seulement la bannière de démarrage du 10 août.
Second_generation_decoding.txt	Extrait du journal : une balise 2G en autotest décodée (code pays 710, Brésil), avec la commande grep qui l'a produit.
docs/articles/Balises_sarsat_18-07-26.txt	Votre journal du 18 juillet, 609 Ko, publié dans le dépôt amont.
epl_diag.csv	Mesures de corrélation bit par bit d'une salve 2G (sortie de l'outil sgb_epl_diag, 25 juillet).
Capture d'écran du 2026-07-23 ...png	9 captures du 23 juillet. Les 3 que j'ai ouvertes montrent SatDump : spectre autour de 1544,5 MHz sur clé RTL, poursuite du Soleil au rotor, courbe de puissance reçue.
country	Fragment de code C (13,7 Ko, 18 juillet) : copie de travail pour l'ajout des noms de pays. Ce n'est pas un programme.
CONTENU_DU_REPERTOIRE.md, .pdf	Le présent document (6 octobre).

Pointage d'antenne et SatDump

Fichier	Rôle
MSG-4	Pointe la parabole à l'azimut 170° et l'élévation 37° (Meteosat) via <code>poursuit_sat.py</code> .
<code>poursuit_sat.py</code>	Lien vers <code>~/MEOSAR-pointing/poursuit_sat.py</code> .
<code>gps.sh</code>	Lance <code>meosar_pointing_old.py</code> (élévation minimale 84°). Ne fonctionne pas depuis ce répertoire : le script Python est dans <code>~/MEOSAR-pointing</code> et le journal vise un dossier qui n'existe plus.
<code>tools/meosar_visible.py</code>	Liste les satellites MEOSAR visibles depuis votre station (projet amont, nécessite <code>skyfield</code>).
<code>restart_satdump</code> , <code>status_satdump</code>	Redémarre ou affiche l'état du service <code>satdump</code> .

Projet amont : sources et documentation

Élément	Contenu
README_FR.md, README.md	Mode d'emploi complet, en français et en anglais. À relire en premier.
CHANGELOG_FR.md, CHANGELOG.md	Historique des versions.
Makefile	Compilation : <code>make</code> reconstruit tout dans <code>build/</code> .
LICENSE, .gitignore	Licence MIT ; fichiers ignorés par git.
src/	29 sources C : décodeurs <code>dec406_v1g.c</code> (1G) et <code>dec406_v2g.c</code> (2G), démodulateurs, scanner, alertes courriel, pilotes Airspy, RTL-SDR, Pluto et HackRF.
src/dec406_v2g_ex.c	Votre variante du décodeur 2G (25 juillet), hors dépôt et non compilée par le Makefile.
include/	En-têtes C, dont <code>country_codes.h</code> (noms de pays, votre contribution intégrée en amont).
include/country_pseudo_MID_codes.h	Votre version de travail de la table des pays (19 juillet), hors dépôt.
utils/	13 outils de diagnostic en C : générateurs de signaux de test, comparaison de codes, rééchantillonnage IQ.
scripts/	Deux scripts Python de tracé de figures.
tests/	Un test unitaire (<code>test_twc_rf4.c</code>).
docs/	Notes techniques : protocoles couverts, état et journal d'enquête du décodage 2G, procédures de validation.
systemd/	Modèles de services <code>scan406</code> et <code>scan1544</code> , avec les chemins de l'auteur (<code>/home/firmin/...</code>) à adapter.
data/config_mail.txt	Vos identifiants SMTP Gmail pour les alertes (lecture réservée à vous). Le modèle est <code>config_mail.txt.example</code> .
.git/	Historique du dépôt, 117 Mo. Ne pas y toucher.

Chaîne de décodage

Deux voies mènent du satellite à la balise décodée : le scan en temps réel, qui fait tout seul, et la capture IQ suivie d'un décodage différé.

1. **Pointer l'antenne.** `tools/meosar_visible.py` indique les satellites visibles ; `MSG-4` ou `poursuit_sat.py` orientent la parabole.
2. **Libérer le récepteur.** Un SDR ne sert qu'un programme à la fois. Le service `satdump` est actif : s'il tient le récepteur voulu, l'arrêter d'abord.
3. **Voie temps réel.** Un script `acq_*` lance `dec406_scan` sur une bande. Le scanner repère chaque salve, la classe 1G (moins de 20 kHz de large) ou 2G, la décode et l'affiche. `tee` recopie tout dans un journal `Balises_*.txt`.
4. **Voie différée.** Un script `bursts_*` enregistre des fichiers `.iq` avec `airspy_rx`, puis `dec406_iq <fichier> -i -s 2500000` les décode. Utile pour rejouer un même signal après une mise à jour du décodeur.
5. **Alerte courriel.** Si `data/config_mail.txt` existe, une vraie balise de détresse entendue deux fois déclenche un courriel. Avec `DEC406_ALERT_TEST=1`, toute balise décodée envoie un.
6. **Contrôle d'une trame.** `dec406_hex <trame hexadécimale>` redécodé une trame relevée dans un journal.

Repère de fréquences, donné de mémoire et à vérifier : les satellites Galileo relaient vers 1544,1 MHz, les satellites défilants et géostationnaires (dont Meteosat) vers 1544,5 MHz, GPS et Glonass vers 1544,9 MHz. Vos scripts appellent « leo » la bande 1544,0 – 1544,15 et « meo » la bande 1544,4 – 1544,6.

Alertes par courriel

Le scanner peut envoyer un courriel quand une balise est décodée. Sur cette station, qui ne fait que tester le décodage, seul le mode essai est utile.

Réglage	Effet
<code>data/config_mail.txt</code> présent	Active les alertes ; la bannière affiche <code>alerts : enabled</code> . C'est le cas ici.
Aucune variable (mode direct)	Alerte seulement pour une balise en détresse réelle, sur un canal de détresse 406 MHz, entendue deux fois en moins de 3 minutes.
<code>DEC406_ALERT_TEST=1</code>	Envoie un courriel marqué [TEST] pour chaque balise décodée, tous filtres ignorés.
<code>DEC406_ALERT_TEST=2</code>	Même essai, limité aux balises 2G (depuis la mise à jour du 6 octobre).
<code>DEC406_ALERT_MODE=downlink</code>	Mode prévu pour 1544 MHz : alerte si la position décodée est à moins de <code>DEC406_ALERT_RADIUS_KM</code> de <code>DEC406_ALERT_CENTER=lat,lon</code> . Non utilisé ici.

Conséquences pour vos scripts :

- En mode direct, l'écoute sur 1544 MHz n'envoie aucun courriel : les fréquences reçues ne sont pas des canaux 406 MHz. C'est ce que disent la documentation et le code ; cela n'a pas été vérifié par un essai.
- Seuls `acq_sarsat_leo_Airspy-SDR` et `acq_sarsat_meo_Airspy-SDR` envoient des courriels, grâce à `DEC406_ALERT_TEST=1`. Dans `acq_sarsat`, la ligne qui l'active est en commentaire.
- Pour essayer l'envoi sur une autre bande, placer `DEC406_ALERT_TEST=1` devant la commande `./build/dec406_scan`.
- L'envoi passe par le programme `sendemail`, installé sur `f4klo`.

La référence est README_FR.md, sections « Alertes courriel sur les canaux de détresse T.012 » et « Service de liaison descendante 1544 MHz ».

Pour reprendre en main

Commencez par README_FR.md, puis relancez ./acq_sarsat, le script le plus récent et le plus large (1544,0 – 1544,6 MHz, journal Balises_sarsat.txt).

```
cd ~/Decode_sarsat_406
less README_FR.md
./status_satdump
./acq_sarsat
```

Pour relire ce qui a déjà été reçu :

```
grep -c BURST Balises_sarsat.txt
grep "Hex ID" Balises_sarsat.txt | sort | uniq -c
grep SECOND -B25 -A35 Balises_sarsat.txt | less
```

Points à garder en tête :

- La bannière de démarrage affiche le récepteur trouvé, le gain et l'état des alertes : vérifiez-la à chaque lancement.
- Les lignes monitoring toutes les 15 s sont normales ; une réception se signale par BURST, puis par un bloc 406 MHz ... BEACON.
- Le dépôt et le fork ont été mis à jour le 6 octobre, et les programmes recompilés sauf l'ancien scanner Airspy et reset_usb. git fetch origin puis git log main..origin/main montreront ce que l'auteur publiera ensuite.
- Depuis le relevé, seuls le dépôt, les programmes compilés et ce document ont changé sur f4klo.