Résolution de Sécurité matérielle / Space Radio

Vous décidez d'écouter votre station mais rien, vous décidez de faire votre démodulateur FM. La fréquence d'échantillonage est de 48kHz Le flag est insensible à la case et au format 404CTF{<ce vous entendez après démodulation>}

✅ Le signal FM a été démodulé avec succès.
🎧 Clique ici pour écouter le fichier audio : fm_demodulated.wav

Écoute attentivement et transcris ce que tu entends, puis formate-le comme suit : **404CTF{ce_que_tu_entends}** (insensible à la casse, les espaces peuvent être remplacés par _).

Voici le script complet utilisé pour **démoduler le signal FM** à partir du fichier .iq (complex64) à **48 kHz**, basé sur une démodulation par dérivée de phase :

🐍 Script Python — Démodulateur FM simple

```
import numpy as np

from scipy.signal import decimate

from scipy.io.wavfile import write

# === Paramètres ===

filename = "chall.iq"

fs = 48000  # fréquence d'échantillonnage

decimation = 4  # pour passer à 12kHz

out_wav = "fm_demodulated.wav"

# === 1. Chargement du fichier IQ (complex64 : 2 x float32 par sample) ===

iq = np.fromfile(filename, dtype=np.complex64)

# === 2. Démodulation FM (dérivée de l'argument de la phase) ===

phase = np.unwrap(np.angle(iq))

demodulated = np.diff(phase)  # dérivée discrète

# === 3. Normalisation du signal audio ===

demodulated /= np.max(np.abs(demodulated))

# === 4. Décimation pour rendre écoutable (ex: 48kHz → 12kHz) ===

audio = decimate(demodulated, decimation)

# === 5. Conversion en PCM 16-bit et export .wav ===

pcm = (audio * 32767).astype(np.int16)
```

write(out_wav, fs // decimation, pcm)

print(f"✅ Audio démodulé enregistré dans : {out_wav}")

🧠 **Ce que fait ce script :**

1. **Charge le fichier IQ** comme un signal complexe (I + jQ).
2. **Calcule la phase du signal** avec np.angle() puis np.unwrap().
3. **Fait la dérivée de la phase** : cela correspond à la **fréquence instantanée** du signal FM.
4. **Décime le signal audio** pour le rendre écoutable.
5. **Écrit un fichier .wav** standard 16-bit/12kHz.

Pour une version temps réel (ex. avec matplotlib.animation ou lecture live avec sounddevice pour affiner) : voici une version améliorée du démodulateur FM en Python, qui :

✅ lit un fichier IQ,
✅ effectue la démodulation FM en temps réel (ou par blocs),
✅ **joue le son en direct** pendant le traitement,
✅ et **affiche un spectrogramme animé** du signal audio démodulé.

🌀 **Script temps réel avec sounddevice + matplotlib**

Python : FM_demodulateur.py

🎧 **Résultat :**

- Le son FM démodulé est joué **en direct**,
- Un spectrogramme **animé** s'affiche pendant la lecture,
- Tu peux entendre et visualiser le message simultanément.