

# Documentation - ADRF5720 Control Board

---

## Table des matières

Documentation - ADRF5720 Control Board.....	1
1. Présentation générale .....	1
2. Description de la carte Shield .....	1
3. Fonctionnement de l'ADRF5720 .....	4
4. Code embarqué C.....	4
4.1 Téléversement du code embarqué.....	4
5. Logiciel de contrôle Python .....	4
5.1 Lancement du script Python.....	5
6. Fichier de configuration JSON.....	5
7. Utilisation.....	6
8. Avertissement .....	6

## 1. Présentation générale

Cette documentation présente la carte shield développée pour contrôler le module d'atténuation RF numérique ADRF5720-EvalZ via une carte Arduino Uno ou un microcontrôleur STM32 équivalent (STM32 NUCLEOF103RB). Elle permet d'envoyer des commandes d'atténuation à l'ADRF5720 par une interface série grâce à un logiciel Python et à un code embarqué sur l'Arduino/STM32. Le but de ce module de contrôle est d'ajuster le gain d'un signal de bruit Gaussien à l'entrée de l'ADC dans le cadre du projet ALMA.

## 2. Description de la carte Shield

La carte Shield permet l'alimentation et le contrôle de l'ADRF5720 Eval-Z.

Voici ses principales caractéristiques matérielles :

- Connecteur CON2 : permet le choix de l'alimentation à l'aide d'un jumper (Arduino : 1-2 / Externe : 2-3)
- Bornier d'alimentation : source extérieure d'alimentation

- Connecteur CON6 : se branche sur le connecteur 19 broches de l'ADRF5720 Eval-Z pour piloter l'atténuation
- Connecteur CON3 : fournit les tensions nécessaires au fonctionnement de l'ADRF5720 (VSS = -3.3V, GND, VDD = +3.3V)
- Points de test : permettent de vérifier l'ordre correct des séquences d'allumage et d'extinction (VDD avant VSS à l'allumage, VSS avant VDD à l'arrêt)
- LEDs : 6 LEDs affichent l'état des 6 bits envoyés au module  
LED éteinte = 0, LED allumée = 1  
**Exemple** : 000001 = 0.5 dB, 001100 = 6 dB, 111111 = 31.5 dB

## Schéma électronique

Vous trouverez aussi le schéma électronique disponible ci-dessous dans le dossier d'utilisation en format pdf : Schéma\_électronique\_ADRF5720

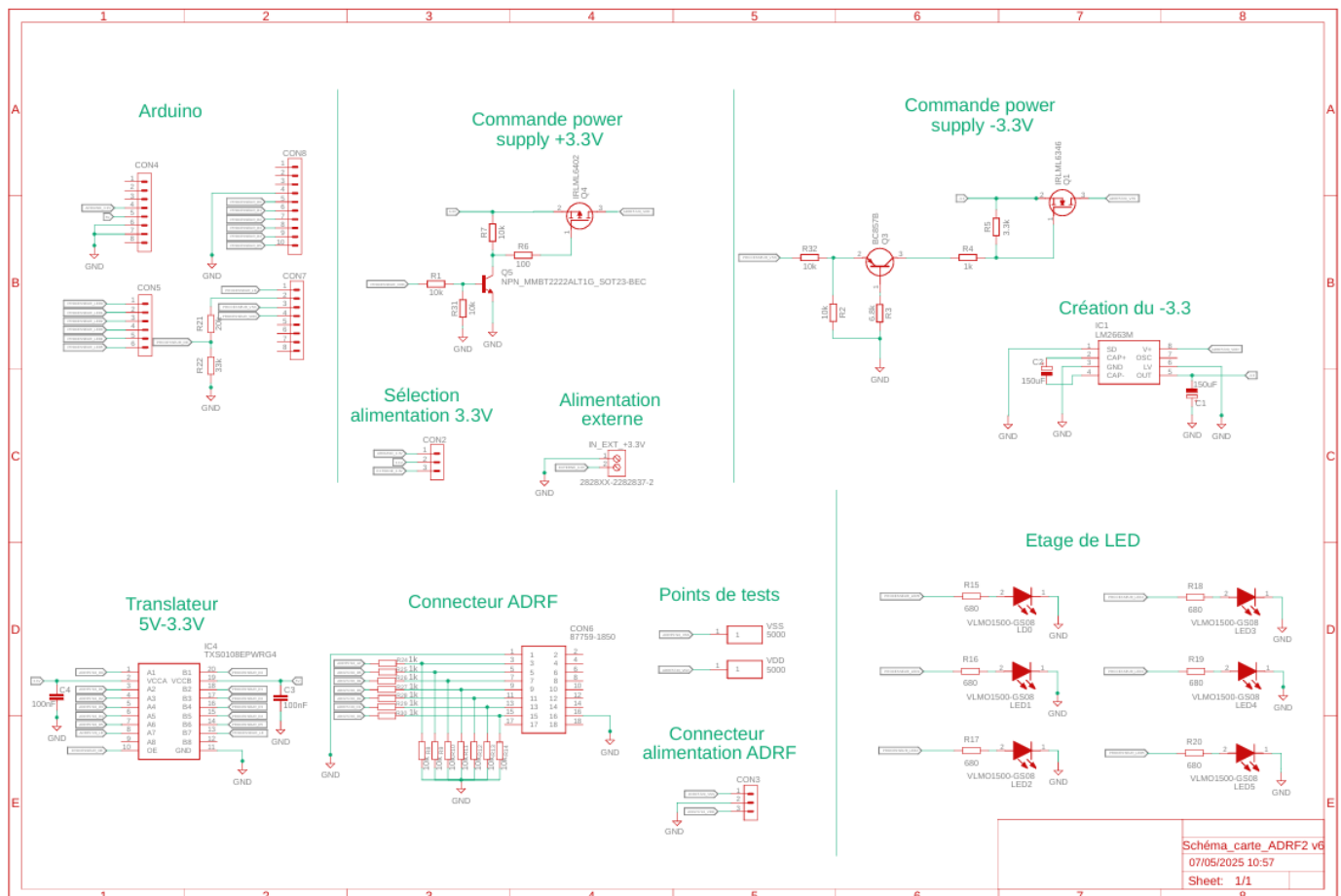
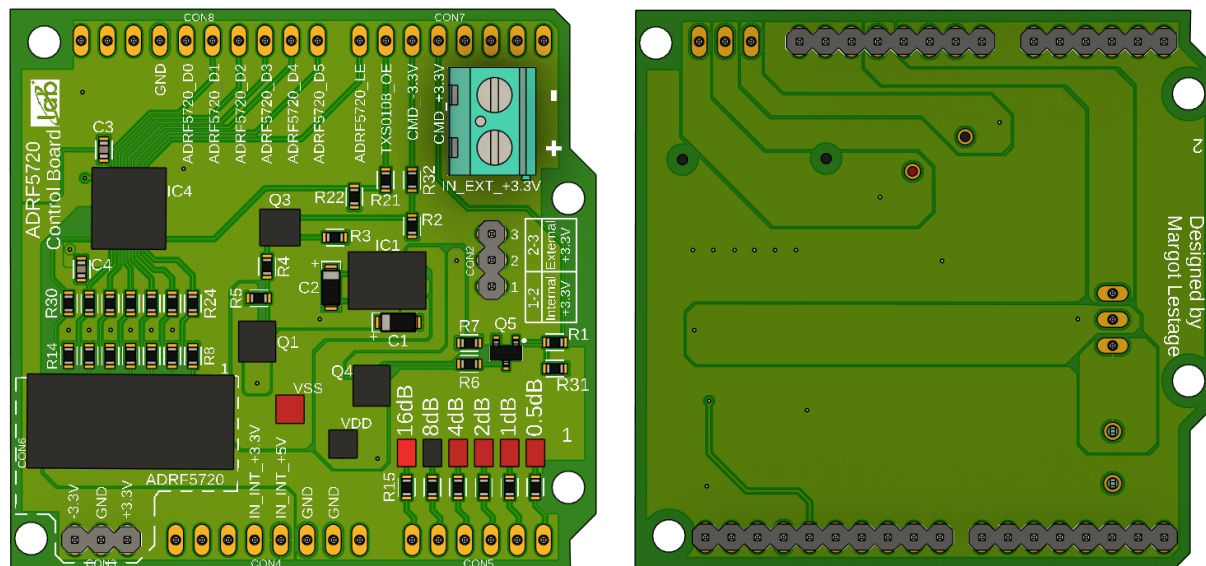


Figure 1 : Schéma électronique de la carte shield de contrôle de l'ADRF5720

### *Vue 3D de la carte*

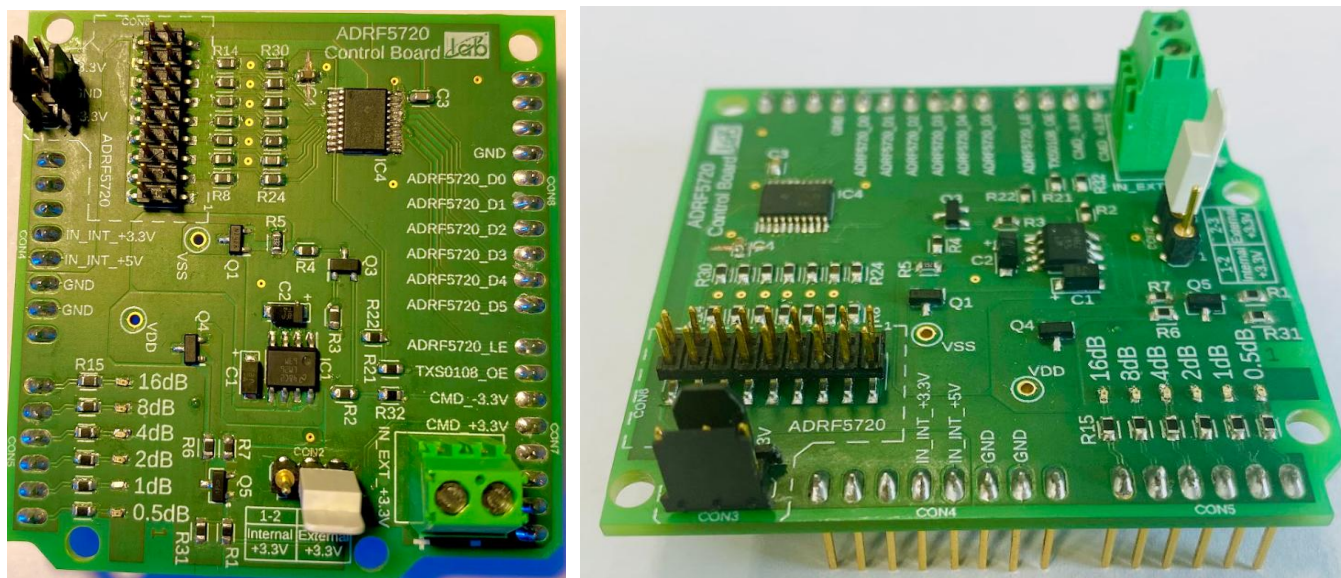
Sur cette vue 3D certains composant n'ont pas de design 3D rattaché à la bibliothèque.



**Figure 2 : Vue 3D de la carte shield de contrôle de l'ADRF5720**

### *Photo de la carte*

Voici des photos de la carte de contrôle.



**Figure 3 : Photos de la carte shield de contrôle de l'ADRF5720**



### 3. Fonctionnement de l'ADRF5720

L'ADRF5720 est un atténuateur ajustable numérique RF. Il permet de contrôler l'atténuation d'un signal RF sur une plage de 0 à 31.5 dB par pas de 0.5 dB. La bande passante de l'ADRF5720 est de 9 kHz à 40 GHz donc adapté au projet ALMA. Sur la carte de contrôle, le module est piloté via une interface parallèle 6 bits.

### 4. Code embarqué C

Le firmware embarqué sur l'Arduino ou la carte STM32 est responsable de la séquence d'alimentation correcte de l'ADRF5720, de la réception des commandes série et de l'envoi des bits d'atténuation. Il contient des protections contre les erreurs de syntaxe ou d'utilisation.

Le code gère :

- L'état de l'alimentation (power on/off)
- L'envoi de l'atténuation à l'ADRF5720
- La synchronisation via le LE (Latch Enable)
- La visualisation via les LEDs

#### 4.1 Téléversement du code embarqué

##### ➤ Cas 1 – Arduino Uno :

1. Ouvrir le fichier **Code\_C\_ADRF5720.ino** via le logiciel Arduino IDE.
2. Dans le menu Outils → Type de carte, sélectionner "Arduino Uno".
3. Dans Outils → Port, choisir le port COM correspondant à l'Arduino (visible dans le gestionnaire de périphériques).
4. Cliquer sur Téléverser (icône flèche).

##### ➤ Cas 2 – STM32 Nucleo (STM32F103RB) :

1. Ouvrir le fichier **ADRF5720\_code\_STM320.ioc** dans STM32CubeMX.
2. Générer le code via Project → Generate Code, puis ouvrir le projet dans STM32CubeIDE.

Le branchement du shield sur la carte est automatiquement orienté correctement grâce aux connecteurs asymétriques. Aucun risque d'inversion possible.

### 5. Logiciel de contrôle Python

Un script Python permet à l'utilisateur de communiquer avec l'Arduino via le port série. Il peut envoyer des commandes individuelles ou des séquences automatisées d'atténuation.



Le script inclut les fonctionnalités suivantes :

- La lecture du port et paramètres depuis un fichier JSON
- L'envoi de commande simple (exemple : 'att 6.0')
- L'envoi de séquences d'atténuation (exemple : de 0 à 31.5 dB par pas de 1 avec temps de 10s entre chaque)
- L'affichage des réponses ou erreurs

## 5.1 Lancement du script Python

1. Ouvrir le fichier `Code_Python_ADRF5720_V5.py` avec un éditeur Python (VSCode, etc...).
2. Installer la bibliothèque nécessaire à la communication série via le terminal :

```
pip install pyserial
```

3. Vérifier que le fichier `Parameter.json` est présent dans le même dossier que le script.
4. Cliquer sur « Run Python File in Terminal » en haut à droite si vous utilisez VS Code.

→ Si vous utilisez un autre éditeur que VS Code :

5. Vous pouvez aussi lancer le script depuis un terminal avec :

```
python Code_Python_ADRF5720_V5.py
```

## 6. Fichier de configuration JSON

Le fichier `Parameter.json` contient les paramètres nécessaires au fonctionnement du script Python : port de communication, débit en bauds et messages d'erreur personnalisables.

Ce fichier doit impérativement être adapté à la configuration de votre poste de travail, pour définir le port COM auquel est connectée la carte Arduino. Pour ce faire regarder dans le gestionnaire de périphérique (device manager) quel port l'Arduino ou la STM32 occupe puis changer le fichier .json en fonction (pensez à sauvegarder).

Exemple :

```
{
  "port" : "COM3",
  "baudrate" : 9600,
  "errors" : {
    "ERR0" : "Commande inconnue. Utilisez 'power' ou 'att'.",
    "ERR1" : "Commande mal formée. Vérifiez la syntaxe.",
    "ERR2" : "Valeur de power invalide.",
    "ERR3" : "Erreur de format : utilisez '.' et non ',' ",
    "ERR4" : "Valeur hors limite (0 - 31.5 dB par pas de 0.5).",
    "ERR5" : "Alimentation éteinte : allumez-la d'abord."
  }
}
```



## 7. Utilisation

1. Branchez la carte shield sur la carte Arduino Uno ou STM32 Nucleo-F103RB.
2. Connectez la carte au PC via USB.
3. Chargez le firmware C embarqué selon la méthode indiqué précédemment.
4. Vérifiez le port COM du module dans le gestionnaire de périphériques Windows (device manager)
5. Ouvrez et modifiez le fichier JSON si nécessaire
6. Lancez le script Python
7. Choisissez la source d'alimentation à l'aide du jumper
8. Utilisez les commandes pour gérer l'atténuation

Nom de fonction	Description
att	Renvoie l'atténuation actuelle
att 'valeur'	Change l'atténuation selon la valeur donnée avec 'valeur' compris entre 0 et 31.5dB
power	Renvoie l'état de l'alimentation
power on/off (1/0)	Change l'état de l'alimentation
serie	Commence une série d'atténuation avec la durée entre les changements d'alimentation, le pas d'atténuation et la valeur de départ changeable par l'utilisateur
exit	Sort du programme Python et éteint l'ADRF5720

9. Surveillez les LED pour confirmation visuelle de l'atténuation

## 8. Avertissement

Le code C ne doit pas être modifié sans avoir compris les protections mises en place, en particulier celles relatives à l'alimentation (séquence VDD/VSS) et à la validité des commandes série.