Audacity

Introduction

- L'objectif de ce tutoriel est de valider la bonne acquisition du signal sonore par l'Arduino. Pour cela, nous allons utiliser le logiciel Audacity qui va nous permettre de convertir le signal audio pour pouvoir l'écouter sur un pc.

- Audacity est un logiciel libre, qui est téléchargeable sur https://audacity.fr/

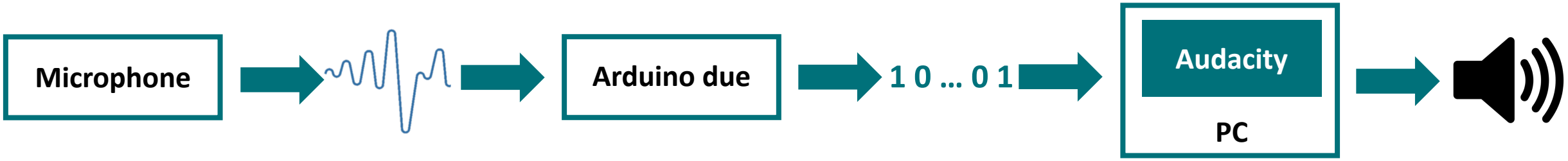


Table des matières

1 Acquisition du signal et transfert série

Acquisition du signal

- L'acquisition du signal se fait à l'aide de l'Arduino Due et de son ADC.
- Pour ce tutoriel, nous allons utiliser l'ADC avec une activation par interruption à l'aide du timer interne du microcontrôleur. Cela nous permet d'avoir une fréquence d'échantillonnage fixe, ici 44KHz.
- Il est important de noter la fréquence d'échantillonnage choisie pour l'importation dans Audacity par la suite (vérifications).

```
void setupADC() {
  PMC->PMC_PCER1 |= PMC_PCER1_PID37; // Active le périphérique ADC
  ADC->ADC_MR = ADC_MR_PRESCAL(0)  // Définit le diviseur de fréquence à 255
              | ADC_MR_STARTUP_SUT64 // Définit le temps de démarrage à 64 périodes
d'ADC_CLK
              | ADC_MR_TRACKTIM(15)  // Définit le temps de suivi à 15 périodes d'ADC_CLK
              | ADC_MR_SETTLING_AST3;// Définit le temps de stabilisation à 17 périodes
d'ADC_CLK
  ADC->ADC_CHER = 0xC0;      // Active le canal 7 (A0)
  // Configure Timer Counter 0 Channel 0 (TC0) pour samplingFrequency
  PMC->PMC_PCER0 |= PMC_PCER0_PID27; // Active le périphérique TC0
  TC0->TC_CHANNEL[0].TC_CMR = TC_CMR_TCCLKS_TIMER_CLOCK4 | TC_CMR_CPCTRG;
  // Définit la source d'horloge à TCLK4 (MCK / 128, 84 MHz / 128 = 656.25 kHz)
  // Active le déclenchement de comparaison RC
  // Définit la valeur RC pour une fréquence samplingFrequency Hz
  TC0->TC_CHANNEL[0].TC_RC = 15; // 44Khz -> ( MCK / 128 / 44000 ) -> 14,9 -> 15
  // Active l'interruption de comparaison RC
  TC0->TC_CHANNEL[0].TC_IER = TC_IER_CPCS;
  // Active l'interruption TC0_IRQn dans le NVIC
  NVIC_EnableIRQ(TC0_IRQn);
  // activation du compteur
  // activation du déclenchement logiciel
  TC0->TC_CHANNEL[0].TC_CCR = TC_CCR_CLKEN | TC_CCR_SWTRG;
}
```

```
void TC0_Handler() {
  // Lit le registre d'état pour effacer le drapeau d'interruption
  TC0->TC_CHANNEL[0].TC_SR;
  // Démarre une nouvelle conversion ADC
  ADC->ADC_CR = ADC_CR_START;
}


void setup() {
  Serial.begin(460800);

  setupADC();
}

void loop() {
  uint16_t value;
  while((ADC->ADC_ISR & 0x80)==0);
  value = ADC->ADC_CDR[7];

  Serial.println(value);
}
```

ECE
ÉCOLE D'INGÉNIEURS
ENGINEERING SCHOOL

Transfert série (1 / 2)

- Nous allons vous présenter dans ce tuto 3 formatages différente pour transférer vos données audios depuis votre Arduino jusqu'au pc.

Transfert des données en ASCII

- Les données vont être envoyée sous forme de caractère ASCII dans un fichier texte sur le pc.

✓ Permet de vérifier « A l'œil » que les valeurs sont bonnes.

✗ Transfert lent.

✗ Conversion en binaire nécessaire par la suite avec un programme python.

```
Serial.println(value);
```

Transfert série (2 / 2)

Transfert des données en binaire

■ Le transfert binaire nécessite de connaitre la taille de la variable à transférer.

En effet, Serial.write() ne permet de transférer qu'un octet à la fois. Si la valeur est plus grande, il faut donc l'envoyer en plusieurs fois, en séparer les différents octets avec un masque. Voici un exemple pour un variable sur 16 bits :

- ✓ Transfert rapide.
- ✓ Fichier binaire directement exploitable pour Audacity.
- ✗ Lecture des données moins intuitive (possible avec un logiciel de visualisation de données binaire).

```
uint16_t value;
// Récupérer les 8 bits inférieurs
uint8_t lowByte = value & 0xFF;
// Récupérer les 8 bits supérieurs
uint8_t highByte = (value >> 8) & 0xFF;
// Envoyer les deux octets via le port série
Serial.write(lowByte);
Serial.write(highByte);
```

⚠ Il faut faire attention à toujours envoyer le lowByte avant le highByte pour que les variables sur 16 bits soient bien transférées

Transfert série (3 / 3)

Transfert des données sous forme de fichier WAV

■ L'objectif de ce formatage est de directement transférer les données sous forme d'un fichier WAV sur le pc. Pour cela, nous allons utiliser une libraire qui créer un header WAV, puis à l'aide de la commande Serial.write(), nous transférons les données dans le corps du fichier. Ce dernier pourra par la suite être écouter directement comme un fichier audio classique.

- ✓ Transfert rapide.
- ✓ Fichier .WAV écoutable sans passer par Audacity
- ✗ Les données ne sont pas utilisables pour d'autres traitements (graphique, spectre …)

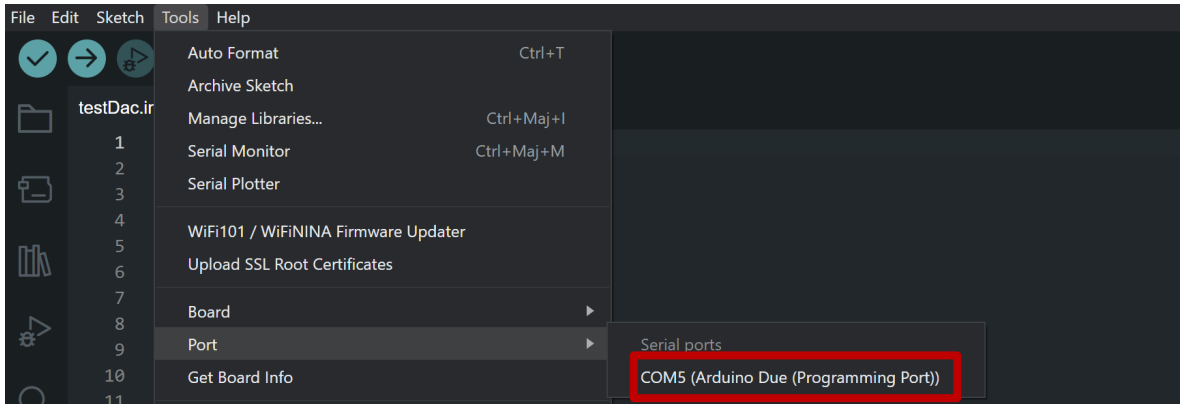
■ Le code permettant de faire ce transfert est trouvable sur GitHub : wave2audacity

Enregistrement du fichier sur le PC (1 / 3)

■ Maintenant que nous avons activé le transfert série, il faut les enregistrer dans un fichier. Pour cela nous allons utiliser le logiciel PuTTY.
si vous êtes sur Windows, choisir l'installateur « 64-bit x86 »

Il est possible de l'utiliser par interface graphique, ou ligne de commande. Pour ce tuto, nous allons utiliser les lignes de commande, plus simple et efficace.
Une fois PuTTY installé, vous pouvez ouvrir l'invite de commandes Windows (rechercher « cmd » sur Windows) puis entrer **plink .** Si l'installation s'est correctement déroulée la commande devrait être reconnue. La commande de la slide suivante est aussi à entrer sur l'invite de commande.

■ Il va aussi être important de connaître le port COM utilisée par votre carte Arduino.



ECE ÉCOLE D'INGÉNIEURS ENGINEERING SCHOOL

Enregistrement du fichier sur le PC (2 / 3)

- La commande suivante permet d'enregistrer le flux de données série dans un fichier sur le pc.

```
plink -serial COM5 -sercfg 460800 > "C:\Users\(username)\audio.bin"
```

Chemin vers le fichier de sortie, en binaire vous pouvez mettre l'extension que vous voulez, avec du texte préférez un .txt

opérateur pipe, il permet d'enregistrer la sortie d'une commande dans un fichier

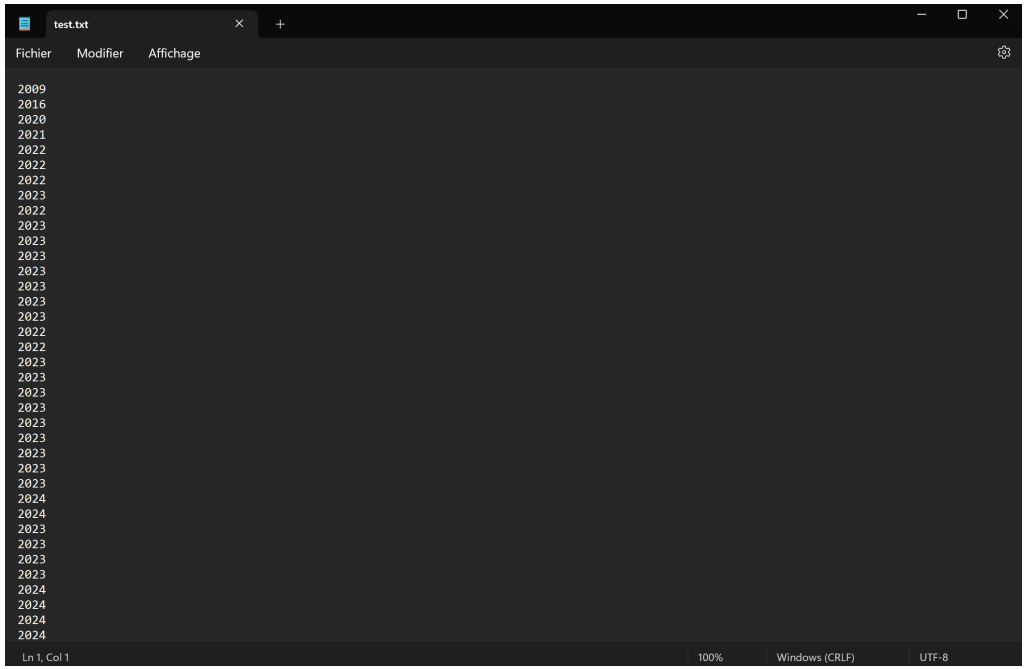
Configuration série, c'est ici que vous indiquez le baud rate (valeur du Serial.begin() de votre code)

Port série utilisé par la carte Arduino

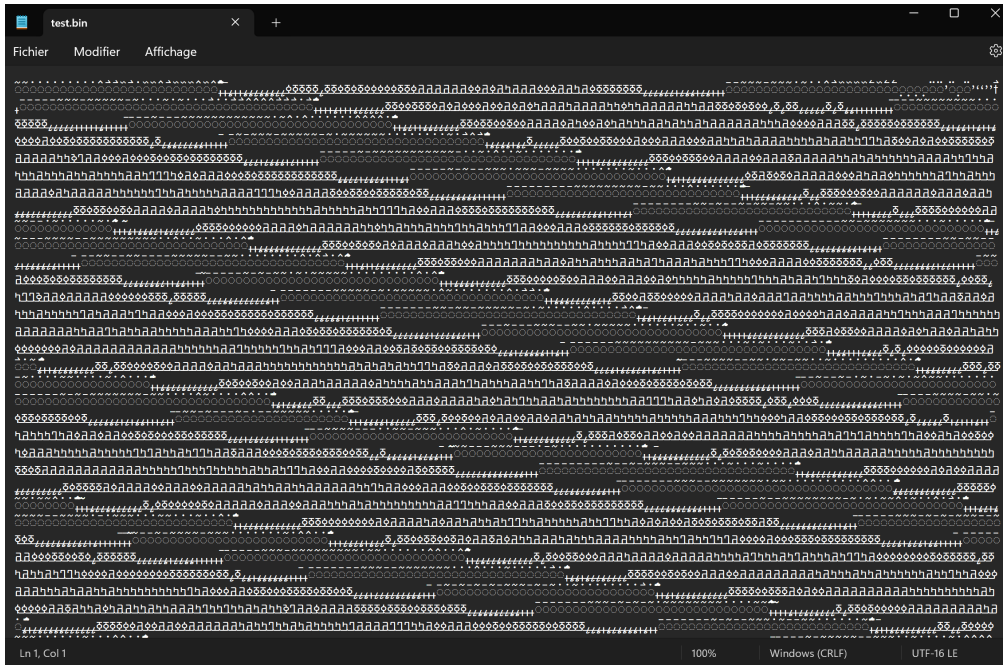
Enregistrement du fichier sur le PC (3 / 3)

■Une fois la commande exécutée, elle va enregistrer indéfiniment le flux de données envoyé en série dans le fichier de sortie. Pour **stopper la commande** à la fin de l'enregistrement, vous pouvez utiliser le raccourci **CTRL+C**.
Vous devriez alors avoir votre fichier enregistré avec les valeurs.

Transfert des données en ASCII



Transfert des données en binaire / WAV



Conversion ASCII en binaire

■ Si vous avez choisi de transférer vos données sous forme de caractères, il va maintenant falloir les convertir en binaire pour pouvoir les importer sur Audacity.

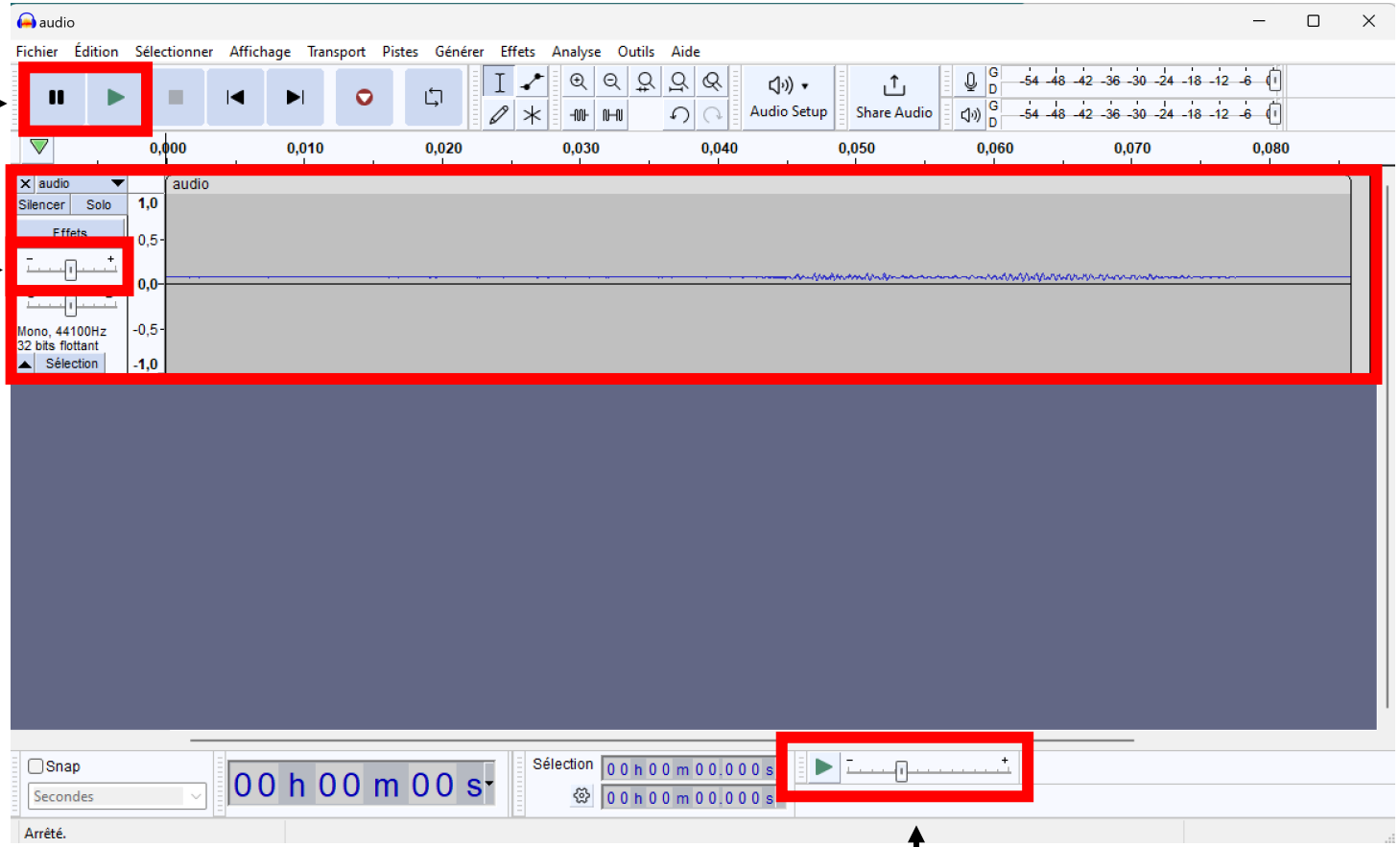
■ Il y a plusieurs options pour effectuer la conversion, nous vous proposons ici un script python permettant de faire la conversion de chaque ligne du fichier texte sous forme de int16.

```
import struct
with open('test.txt', 'r') as f:
    values = [int(x.strip()) for x in f.readlines()]

with open('test.bin', 'wb') as f:
    for value in values:
        binary = struct.pack('<h', value)
        f.write(binary)
```

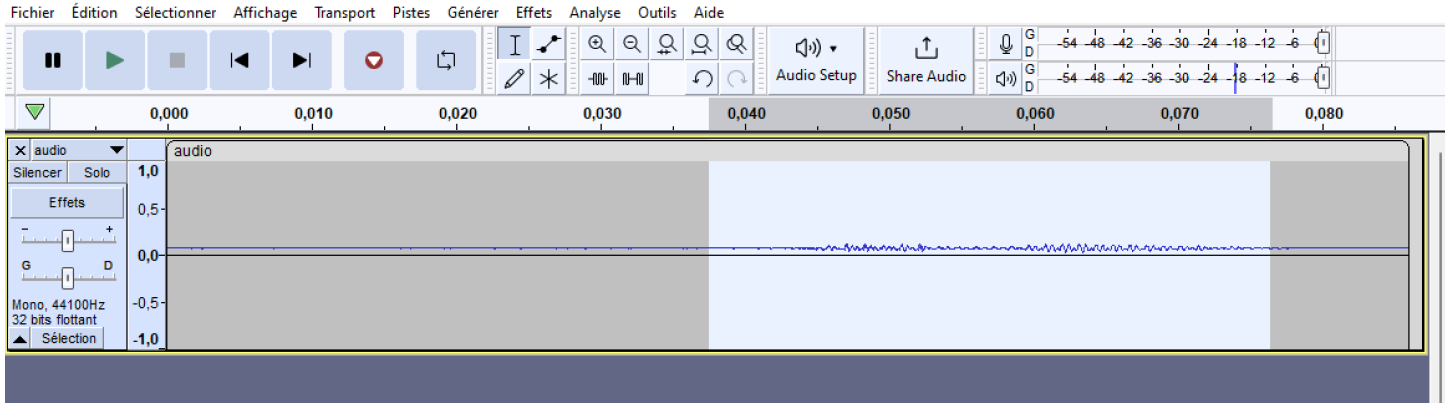
2 Audacity

Prise en main (1 / 2)

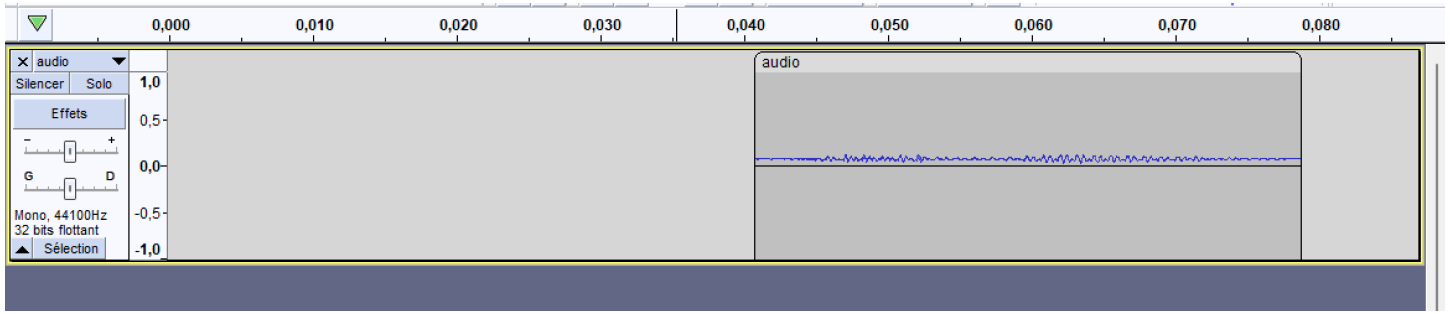


Prise en main (2 / 2)

- Il est possible de ne lire qu'une partie d'un audio en sélectionnant la zone à écouter.



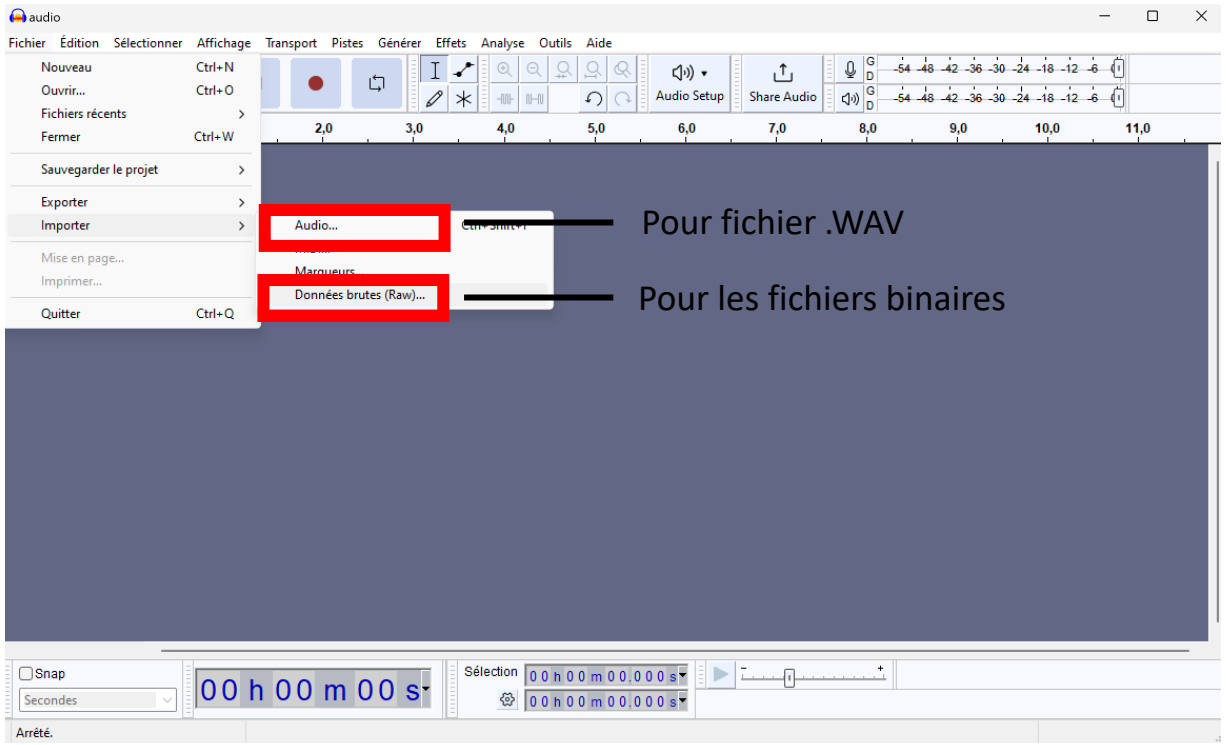
- Vous pouvez aussi retirer des parties des audios, notamment le début et la fin pour ne garder que le son enregistré. Il suffit de drag and drop les bordures de la piste audio.



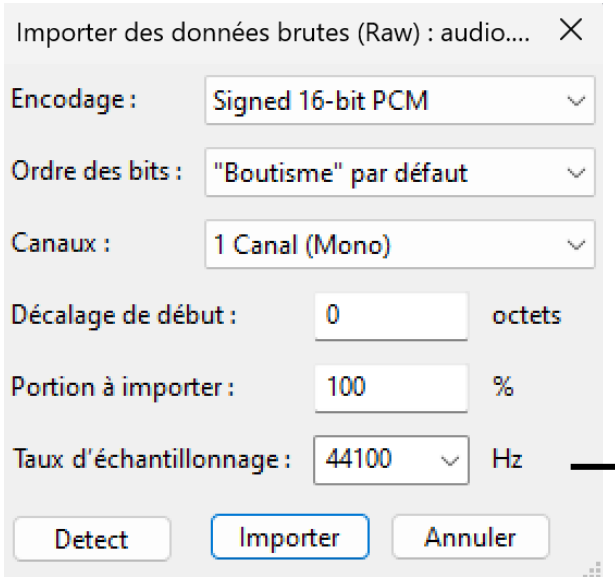
ECE
ÉCOLE D'INGÉNIEURS
ENGINEERING SCHOOL

Importation d'un fichier

- Nous pouvons maintenant importer le fichier binaire créer précédemment dans Audacity



Pour les fichiers binaires



Type de variable : nous avions dans le code :

```
uint16_t value;
```

Or value en sortie de l'ADC est sur 12 bits, donc nous pouvons faire Une correspondance entre Unsigned et signed sans risque d' overflow

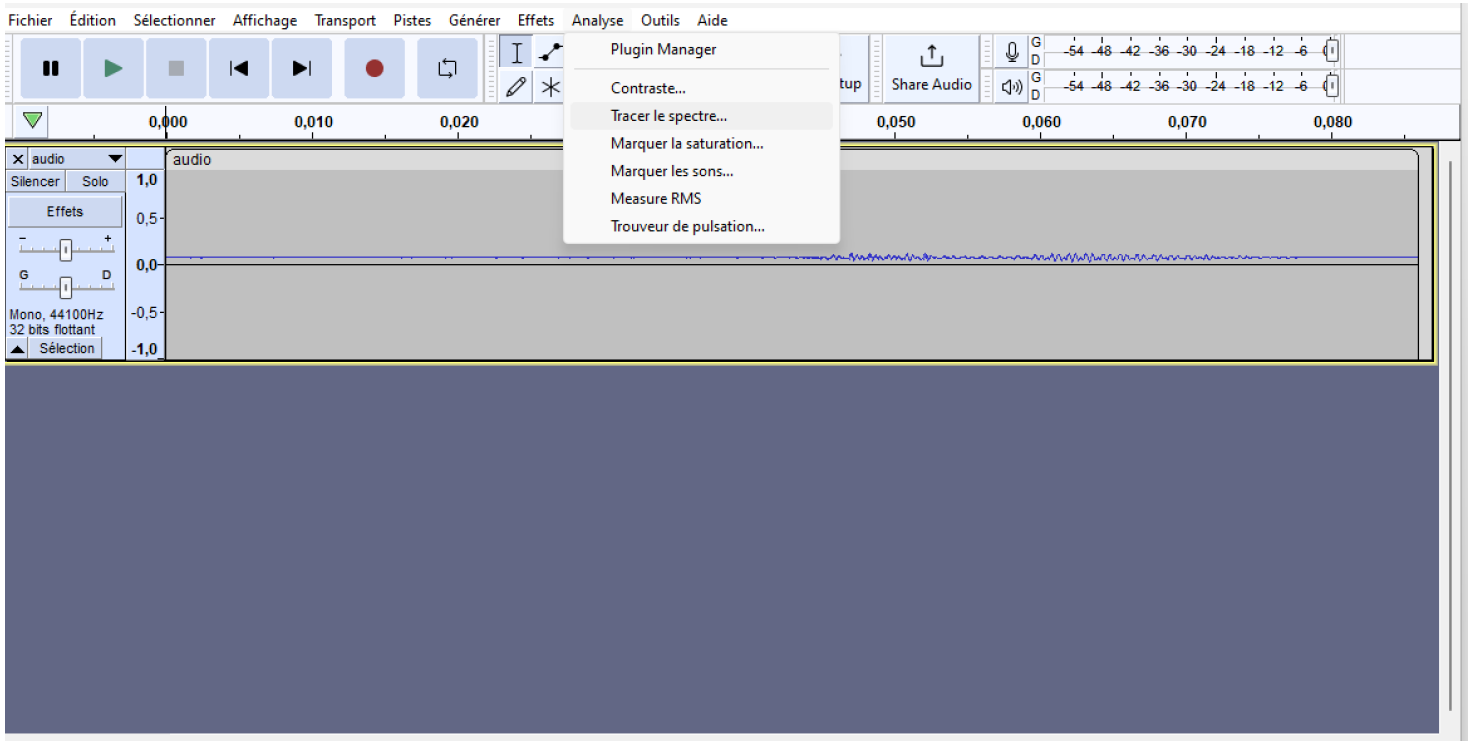
A modifier en fonction de la Fréquence d'échantillonnage De l'ADC.

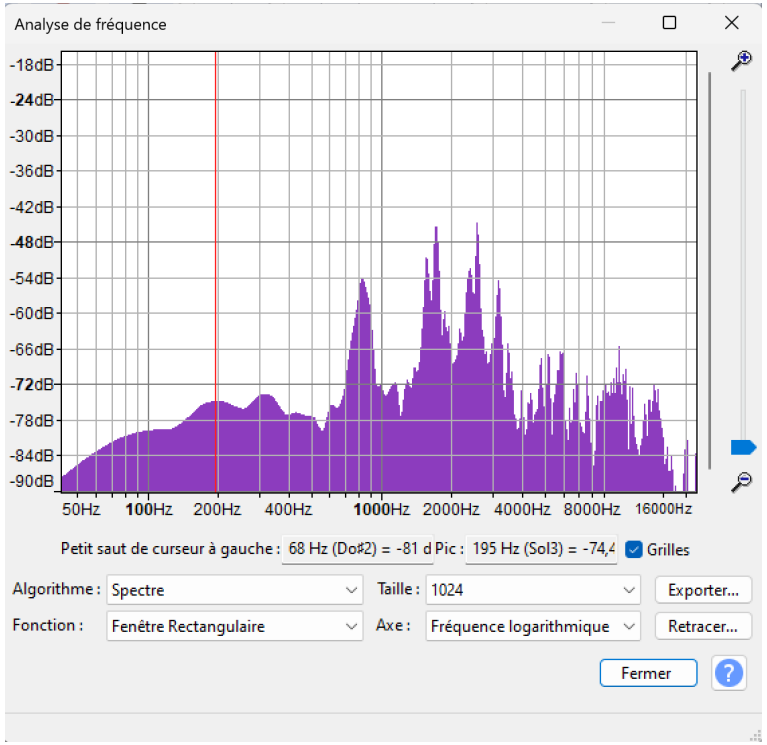
Cliquez sur importer pour confirmer

Analyse de l’audio

- Vous devriez maintenant pouvoir écouter votre audio en appuyant sur PLAY

Il est aussi possible d’en obtenir le spectre à l’aide des outils d’analyse du logiciel :





ECE ÉCOLE D'INGÉNIEURS ENGINEERING SCHOOL

Exportation de l’audio

- Afin de pouvoir écouter l’audio en dehors du logiciel, il est possible d’exporter le fichier dans le format voulu.