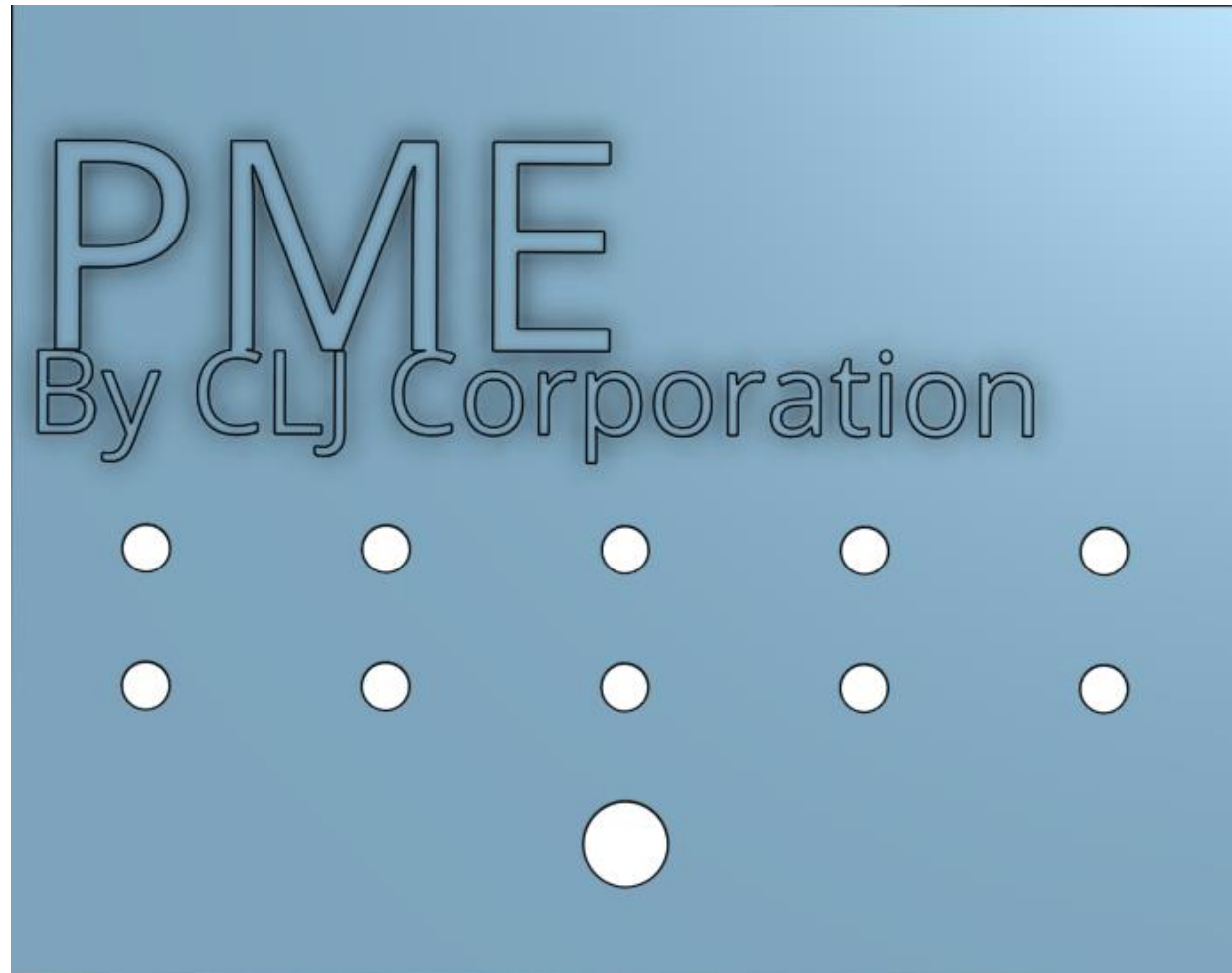


# Pédale Multi-effets (PME)

Projet 2ème année



# Objectifs atteints (12 Séances)




- Schéma PCB
- Composants
- Tests sur Guitare
  
- Prise en main du langage informatique FAUST
- Utilisation du CODEC avec restitution du signal entrée / sortie sur une carte son (STM32, CubeIDE).
- Création d'un effet simple de distorsion.

## Cahier des charges

Créer une Pédale de Guitare :

- Carte électronique 
- Boitier 

Coder des Effets Numérique :

- IN/OUT Codec 
- Effet simple 
- Effet complexe 

# Objectifs restants (10 séances)⌚

- Création d'un boîtier pour la pédale
- Finalisation du PCB.
- Prendre en compte les potentiomètres lors de la réalisation des effets.
- Effet numérique complexe
- Management (impact environnemental, etc...)

## Cahier des charges

Créer une Pédale de Guitare :

- Carte électronique 🤖
- Boîtier ⌚

Coder des Effets Numérique :

- IN/OUT Codec ✓
- Effet simple ✓
- Effet complexe ⌚

# Bilan SOFTWARE ⌚

- Réalisation d'un effet de Reverb en langage Faust

DSP view (FaustIDE) :



Reverb is controlled by the Dry/Wet Mix ( Dry/Wet Mix=-1 → [Large Reverb](#) ; Dry/Wet Mix=+1 → [Small Reverb](#))

# Bilan SOFTWARE ⌚

- Restitution entrée/sortie par le biais d'un CODEC audio avec STM32

```
static int rx_half_cplt_flag = 0;
static int rx_half_cplt_counter = 0;
void HAL_SAI_RxHalfCpltCallback(SAI_HandleTypeDef *hsai)
// half reception SAI is complete
{
    if (SAI2_Block_B == hsai->Instance)
    {
        rx_half_cplt_flag = 1;

//        distort(sai_rx_buffer, SAI_RX_BUFFER_LENGTH / 2, maxsignal);
//        granular(sai_rx_buffer, SAI_RX_BUFFER_LENGTH / 2);
        compute( SAI_RX_BUFFER_LENGTH / 2, sai_rx_buffer, sai_tx_buffer);

        for (int i = 0; i < SAI_RX_BUFFER_LENGTH / 2; i++) {
            sai_tx_buffer[i] = sai_rx_buffer[i];
        }

        rx_half_cplt_counter++;
    }
}
/* USER CODE END 0 */
```

Question

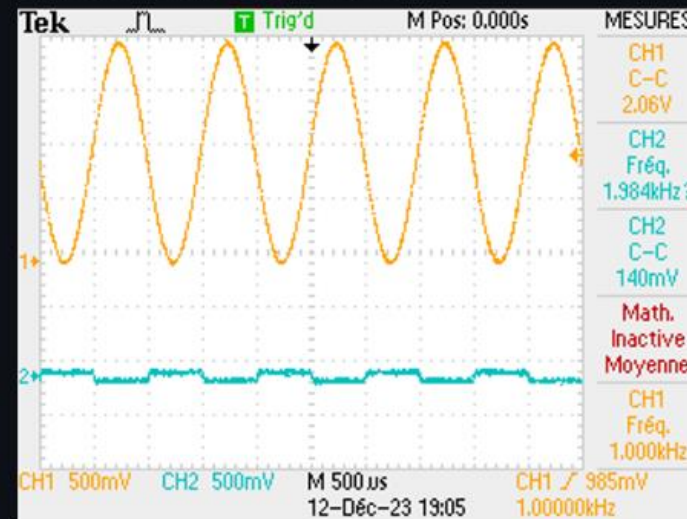
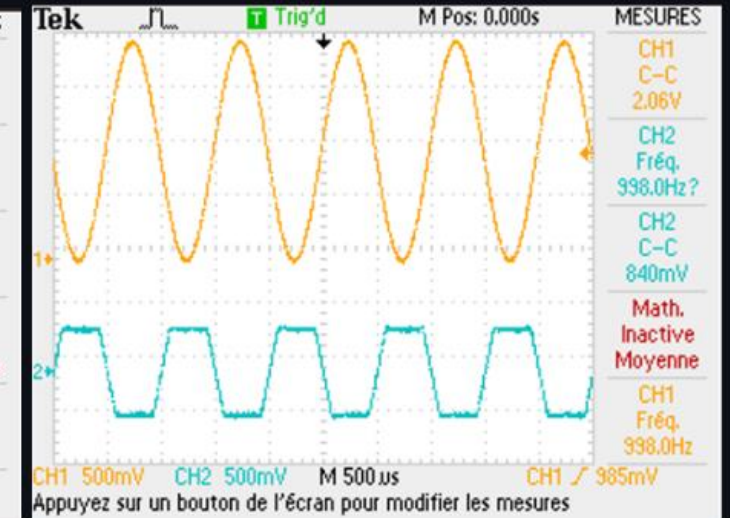
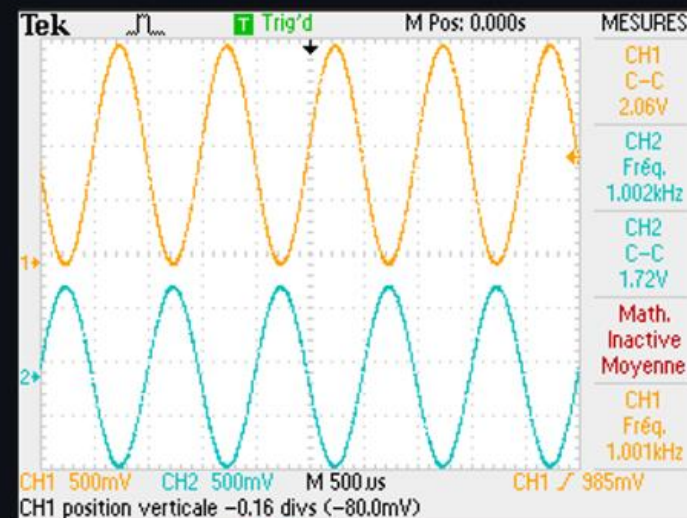
**Int16\_t ou Float**  
**?**

# Bilan SOFTWARE ⌚

- Effet Simple Distorsion (distorsion.c, distorsion.h)

Ajustement de la valeur  
Max à attribuer.

The results are in :



# Bilan SOFTWARE ⌚

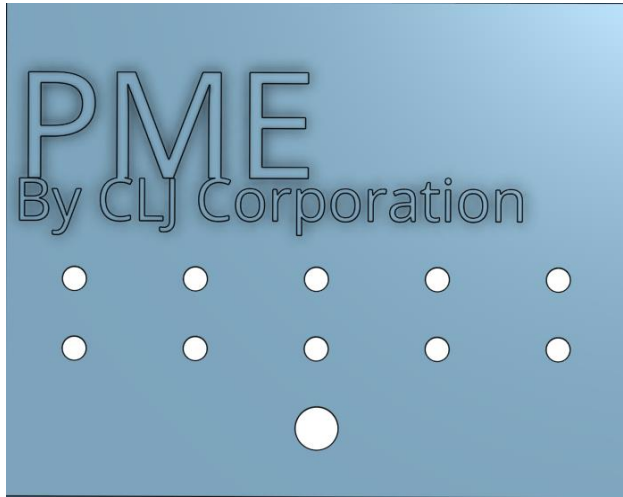
- Effet Reverb by Faust (convert in Cpp)

- Identification partie du code  
Intéressante

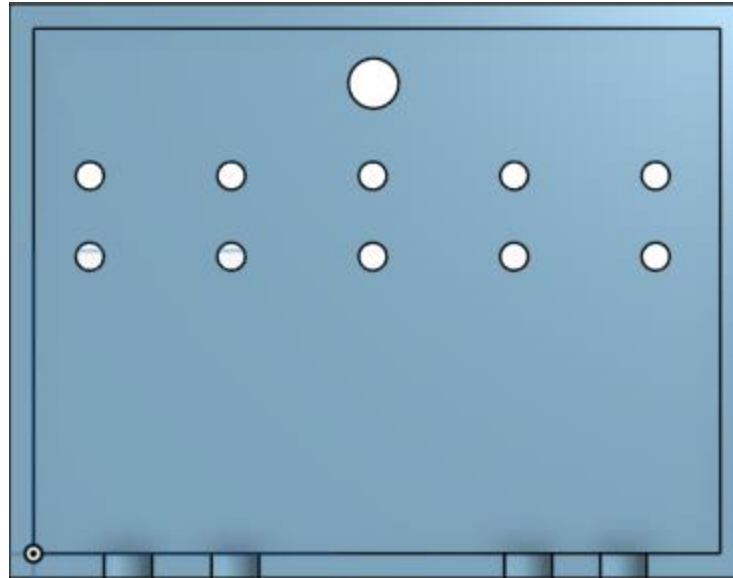
```
virtual void compute(int count, FAUSTFLOAT** RESTRICT inputs, FAUSTFLOAT** RESTRICT outputs) {  
    FAUSTFLOAT* input0 = inputs[0];  
    FAUSTFLOAT* input1 = inputs[1];  
    FAUSTFLOAT* output0 = outputs[0];  
    FAUSTFLOAT* output1 = outputs[1];  
    float fSlow0 = fConst126 * float(fVslider0);  
    float fSlow1 = fConst126 * std::pow(1e+01f, 0.05f * float(fVslider1));  
    for (int i0 = 0; i0 < count; i0 = i0 + 1) {  
        float fTemp0 = fConst6 * fRec1[1];  
        fRec13[0] = -(fConst19 * (fConst18 * fRec13[1] - (fRec6[1] + fRec6[2])));  
        fRec12[0] = fConst21 * (fRec6[1] + fConst20 * fRec13[0]) + fConst16 * fRec12[1];  
        fVec0[IOTA0 & 16383] = 0.35355338f * fRec12[0] + 1e-20f;  
        float fTemp1 = float(input0[i0]);  
        fVec1[IOTA0 & 16383] = fTemp1;  
        float fTemp2 = 0.3f * fVec1[(IOTA0 - iConst24) & 16383];  
        float fTemp3 = fTemp2 + fVec0[(IOTA0 - iConst23) & 16383] - 0.6f * fRec10[1];  
        fVec2[IOTA0 & 4095] = fTemp3;  
        fRec10[0] = fVec2[(IOTA0 - iConst25) & 4095];  
        float fRec11 = 0.6f * fTemp3;
```

- Utilisation du code dans STM32

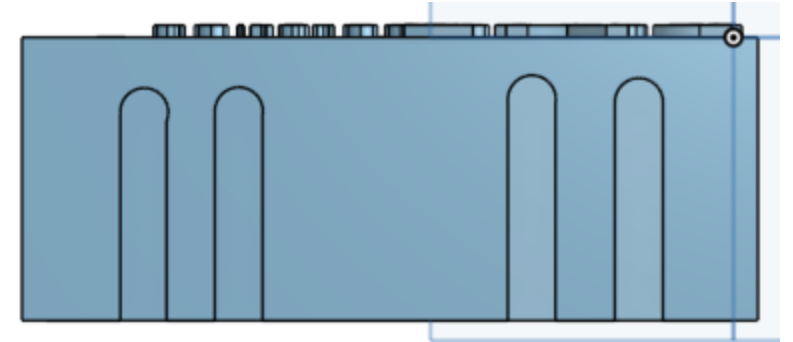
# Modélisation 3D du boîtier de la pédale



Vue de dessus



Vue de dessous



Vue de derrière



# Bilan HARDWARE ⌚

