



Une école de l'IMT

Ingénieur Spécialité Microélectronique et Informatique
Cours de Prototypage
2022-2023

Module Prototypage : Développement d'une Air Guitar

Élèves :

Isadora Eloah Horita
Mário da Silva Araújo

Enseignant :

Sylvain Blayac

30 mars 2023

Table des matières

Introduction	3
Développement du système	3
1 Software COMSOL Multiphysics	4
1.1 Géométrie	4
1.2 Matériaux	5
1.3 Physique	5
1.4 Étude	6
2 Circuit Conditionneur	8
2.1 Analyse de la charge à courant constant	8
2.2 Convertisseur tension courant	9
2.3 Comparateur à hystérésis	11
2.4 Fonctionnement en boucle fermée	12
2.5 Fonctionnement en alimentation unique	14
2.6 Prise en compte des parasites et simulation d'une détection de distance	15
2.7 Assemblage	17
3 Carte Microcontrôleur	21
3.1 Méthode pour identifier la fréquence	21
3.2 Microcontrôleur STM32F301K8T6	21
3.3 Réglages pour la lecture de la fréquence	22
3.4 Développement du code - Lecture de la fréquence	24
3.5 Simulation du code - Lecture de la fréquence	25
4 Carte Commande Actionneurs	27
4.1 Développement du code - Lectures des broches d'entrée	27
4.2 Buzzers	28
4.3 Réglages pour générer un signal PWM	29
4.4 Développement du code - Génération de signaux de sortie PWM	29
4.5 Simulation du code - Génération de signaux de sortie PWM	30
5 Assemblage et Résultats	32
Conclusion	34
Références bibliographiques	35
A Code pour effectuer la lecture de la fréquence	36
B Code pour effectuer la lecture de la fréquence	37
C Code pour vérifier la présence du corps	38
D Code pour effectuer la lecture des entrées	38
E Code pour effectuer le son en fonction des broches	39
F Code main	40

Table des figures

1	Vue d'ensemble du système. [1]	3
2	Composants.	4
3	Tailles e modélisation des composants.	5
4	Maillage.	6
5	Résultat de la simulation Capacité (pF) X position (m).	7
6	Résultat de la simulation Potentiel Électrique.	7
7	Résultat de la simulation du Potentiel électrique avec les lignes du champ électrique.	7
8	Circuit à courant constant.	9
9	Résultat de la simulation à courant constante.	9
10	Circuit convertisseur tension courant.	10
11	Résultat de la simulation du convertisseur tension courant.	11
12	Circuit convertisseur avec le comparateur à hystérésis.	11
13	Résultat de la simulation du convertisseur avec le comparateur à hystérésis.	12
14	Circuit en bouclé fermée.	12
15	Résultat de la simulation en bouclé fermée avec une capacité de 10pF.	13
16	Résultat de la simulation du convertisseur tension courant avec variation de la capacité.	14
17	Circuit en alimentation unique.	15
18	Résultat de la simulation en alimentation unique.	15
19	Circuit complet.	16
20	Résultat de la simulation du circuit complet.	16
21	Résultat de la simulation du circuit complet avec une variation de la distance de la main.	17
22	Carte du circuit imprimé soudé.	17
23	Fréquence de la carte imprimée et soudé.	18
24	Circuit connecté avec la capacité de 10 pF.	18
25	Fréquence de la carte avec capacité de 10 pF.	19
26	Circuit connecté avec la plaque métallique.	19
27	Fréquences obtenu pour l'approche de la main à plaque.	20
28	Exemple de forme d'onde provenant du circuit conditionneur. [5]	21
29	Microcontrôleur STM32F301K8T6 utilisé pour le système. [4]	21
30	Configuration du TIM2.	22
31	Configuration du USART2.	23
32	Test du code pour la lecture de la fréquence.	25
33	Lecture des fréquences à une distance supérieure à 10 cm.	25
34	Lecture des fréquences à une distance inférieure à 10 cm.	25
35	Lecture des fréquences à une distance supérieure à 10 cm.	26
36	Lecture des fréquences à une distance inférieure à 10 cm.	26
37	Configuration du GPIO.	27
38	Représentation d'un buzzer. [8]	28
39	Configuration du TIM1.	29
40	Fréquence de sortie Mi aiguë produite par le microcontrôleur.	31
41	Fréquence de sortie Lá produite par le microcontrôleur.	31
42	Assemblage du prototype - Air Guitar.	32
43	Test de la production de la note musicale Mi aiguë.	33
44	Test de la production de la note musicale Lá.	33

Introduction

La guitare peut être un outil précieux pour aider les élèves des écoles primaires à mieux comprendre la musique dans les cours d'art. Outre l'enseignement des compétences musicales de base, l'apprentissage de la guitare peut présenter de nombreux avantages pour le développement cognitif et social des élèves.

L'un des principaux avantages de l'apprentissage de la guitare est qu'il permet de développer la motricité fine et la coordination oeil-main. Les enfants qui apprennent à jouer de la guitare ont également tendance à avoir de meilleures capacités de concentration et de mémorisation, ainsi qu'une meilleure estime de soi et une plus grande confiance en soi.

Cependant, fournir des guitares à chaque élève peut s'avérer problématique en termes de coût. L'achat de guitares pour une classe entière peut être assez coûteux et de nombreuses écoles n'ont pas les ressources financières nécessaires pour fournir des instruments à tous les élèves. En outre, il peut être difficile de s'assurer que tous les élèves traitent leurs instruments correctement et les maintiennent en bon état.

Ainsi, un prototype de guitare électronique peut contribuer à rendre l'apprentissage de la musique plus accessible et économiquement viable pour ces écoles. En effet, les guitares électroniques peuvent être fabriquées à un coût inférieur à celui des guitares acoustiques traditionnelles, puisque leur production ne nécessite pas de matériaux aussi coûteux.

Il convient également de mentionner qu'un prototype de guitare électronique peut constituer une option plus durable en termes de ressources naturelles, puisque les matériaux utilisés pour la fabrication des guitares acoustiques peuvent être rares dans certaines régions, comme la région du nord-est du Brésil.

Développement du système

Ce rapport traite de la modélisation, de la simulation et de la fabrication d'un capteur sensitif pour développer l'Air Guitar, ce qui simule les cordes d'une guitare. Ainsi, ce capteur est basé sur la théorie des condensateurs de capacité pour capter la présence des corps qui, à leur tour, joueront le rôle d'une des électrodes du système. La Figure 1 illustre l'architecture général du système développé, où nous avons élaboré un circuit conditionneur responsable de la capture du signal en fonction de la présence de tout corps, puis nous associerons un micro-contrôleur pour traiter les données provenant du circuit sensitif afin qu'il soit alors possible de l'allumer ou de l'éteindre pour tout circuit souhaité.

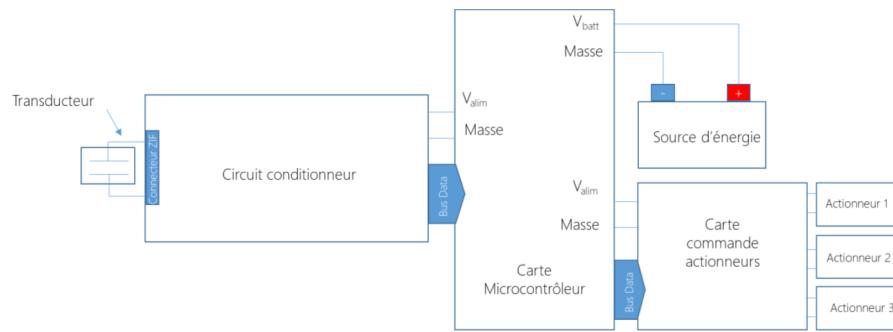


FIGURE 1 – Vue d'ensemble du système. [1]

Ce document est organisé en cinq sections : la section 1 présente l'étude et la simulation du capteur capacitif à l'aide du programme Comsol ; la section 2 présente le développement du circuit électrique pour capturer la variance de la capacitance ; la section 3 présente le traitement du signal provenant du circuit conditionneur pour indiquer la présence ou non d'un corps à proximité du capteur ; la section 4 présente la manière dont le son est produit à partir des boutons qui simulent les notes de musique ; la section 5 traite de l'assemblage et des résultats obtenus à partir du prototype avec les composants développés selon les sections précédentes.

1 Software COMSOL Multiphysics

Nous avons utilisé le logiciel pour simuler l'approche de la main vers la plaque. COMSOL Multiphysics est une plateforme de simulation qui permet de modéliser des phénomènes physiques multiphysiques et monophysiques de manière entièrement couplée. [2] En utilisant ce logiciel, nous avons pu simuler de manière satisfaisante le système comprenant la main, l'isolant et l'électrode. La simulation a permis de prendre en compte les effets de l'approximation de la main et de la variation de la taille de l'électrode.

1.1 Géométrie

La première étape du lancement de la simulation consiste à définir la simulation en 3D et à concevoir les structures. Dans notre cas, nous avons quatre composants :

01. Main ;
02. Substract ;
03. Electrode ;
04. Air.

Ces composants sont représentés sur la Figure 2. Le bloc d'air est celui qui entoure tous les composants et sert d'environnement de simulation. La main est le bloc supérieur, tandis que le substrat est le bloc inférieur dans lequel se trouve le cercle représentant l'électrode.

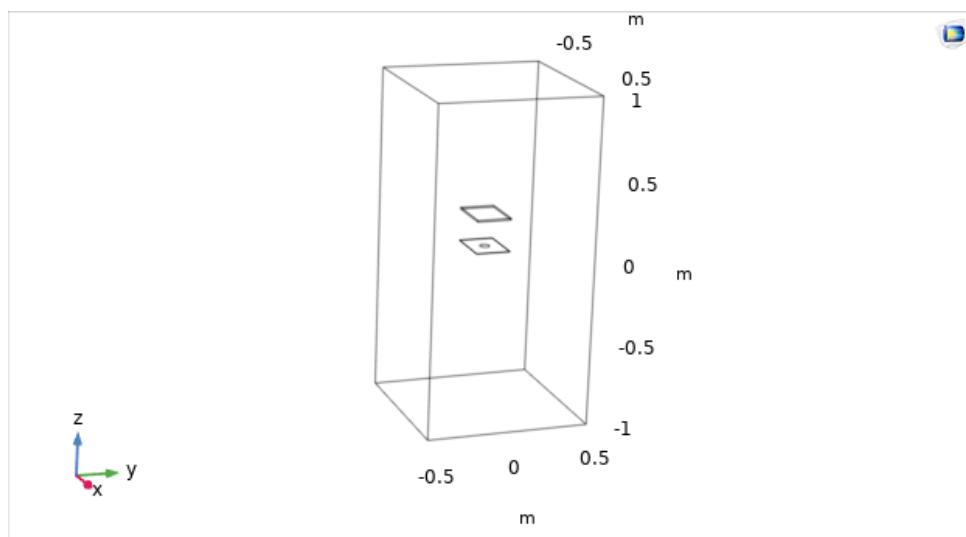


FIGURE 2 – Composants.

Leurs dimensions sont indiquées dans la Figure 3. Il est important de noter que l'électrode a été construite comme une géométrie 2D dans le plan de travail et dans la main, on a le paramètre *position* qui sera analysé dans le étude.

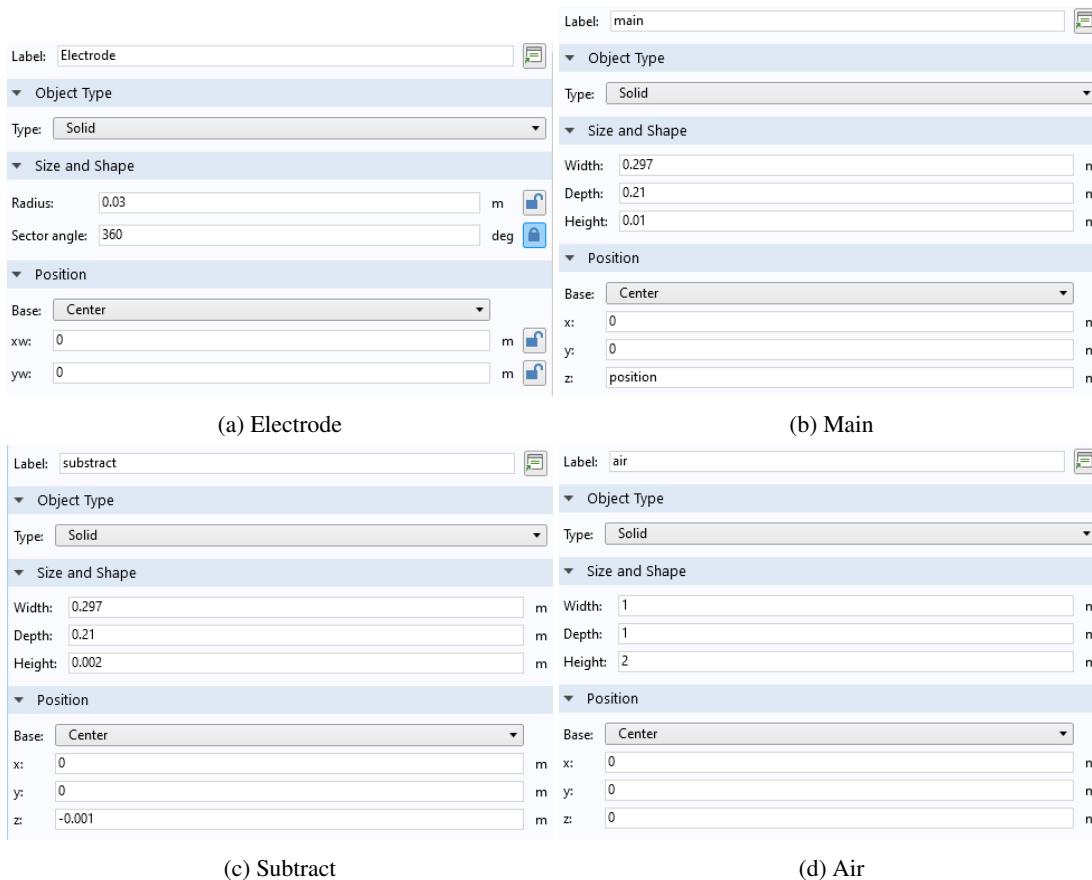


FIGURE 3 – Tailles e modélisation des composants.

1.2 Matériaux

Le étape suivante est le choix du matériau qui compose chaque composant. Pour l'électrode et le substrat, nous avons choisi du PVC (plastifiant à 10 %) [solide] dont la permittivité diélectrique a été modifiée à 3,2. De la même manière, le matériau Quartz a été choisi pour la main, mais sa permittivité diélectrique a également été modifiée, mais dans ce cas, elle a été modifiée à 1100. Enfin, pour le bloc d'air, c'est le matériau air qui a été choisi et qui n'a pas nécessité de changement puisqu'il possède déjà le 1 souhaité.

1.3 Physique

Le choix du matériau étant déjà fait, il est alors possible de créer le maillage, Figure 4 qui permettra d'effectuer la simulation. Mais avant cela, il faut préciser quel est le phénomène physique que l'on souhaite analyser. Dans notre cas, nous avons choisi l'électrostatique, ce qui nous permet de choisir la main comme masse et l'électrode comme terminal, qui, dans cette simulation, a reçu une charge de 5 Coulomb.

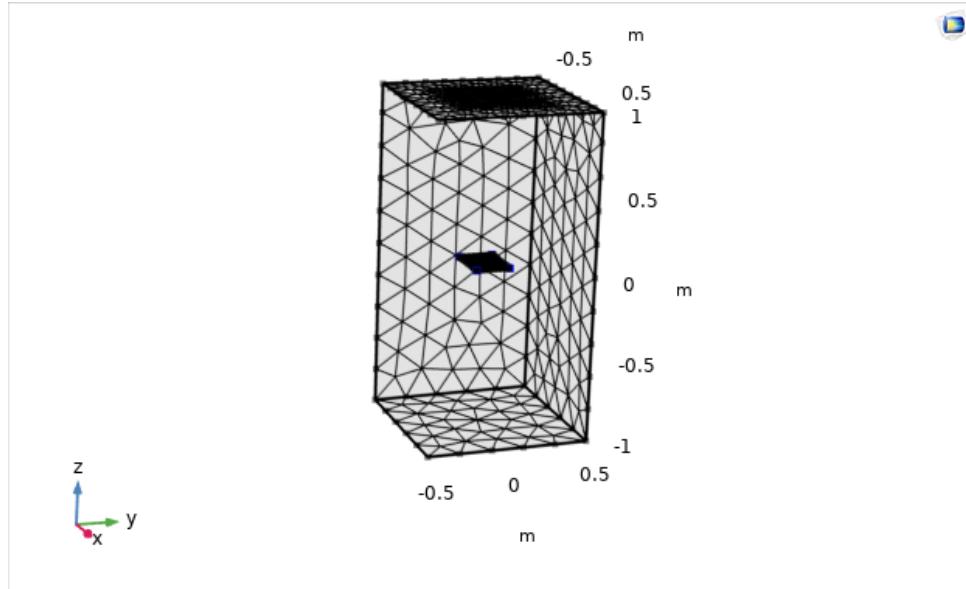


FIGURE 4 – Maillage.

1.4 Étude

Après avoir construit la étape physique, nous pouvons effectuer l'étude. Dans cette section, il est important d'effectuer le sweep paramétrique qui nous permet de réaliser la simulation pour plusieurs variations de taille, de distance et autres. En utilisant cette fonction, la première simulation a été réalisée pour évaluer l'effet que le changement du rayon de la circonférence de l'électrode a sur la capacité résultante lors de l'approche de la main. Pour illustrer cette étude, le Tableau 1 a été élaboré, dans lequel on peut noter que dans tous les cas, l'approche de la main entraîne une augmentation de la capacité du système. Mais on peut aussi observer que lorsque le rayon est de 3cm, on a la capacité souhaitée de 10pF.

position (m)	capacité (pF)				
5,00E-03	844.94	1,45E+07	2,24E+07	3,06E+07	4,18E+07
1,00E-02	2.6896	4.2851	5.8516	7.8027	10.511
1,50E-02	1.9740	2.9710	3.8769	5.0828	6.7026
2,00E-02	1.7214	2.4932	3.2142	4.1616	5.3753
2,50E-02	1.5897	2.2951	2.9246	3.7016	4.7072
3,00E-02	1.5381	2.1612	2.7025	3.4076	4.2852
3,50E-02	1.5174	2.0709	2.5656	3.2184	4.0165
4,00E-02	1.4248	1.9720	2.4645	3.0812	3.8010
4,50E-02	1.3911	1.9460	2.3764	2.9504	3.6274
5,00E-02	1.3928	1.8816	2.3030	2.8474	3.5058
rayon (cm)	1	1.5	2	2.5	3

TABLE 1 – Tableau de analyse du rayon, position et capacité.

Pour approfondir l'étude, avec un rayon de 3cm, nous avons encore effectué un sweep plus complet des paramètres dans lequel la position de la main est comprise entre 0,005m et 0,2m avec un pas de 0,002m. Le résultat est illustré dans la figure suivante. Le résultat est présenté à la Figure 5.

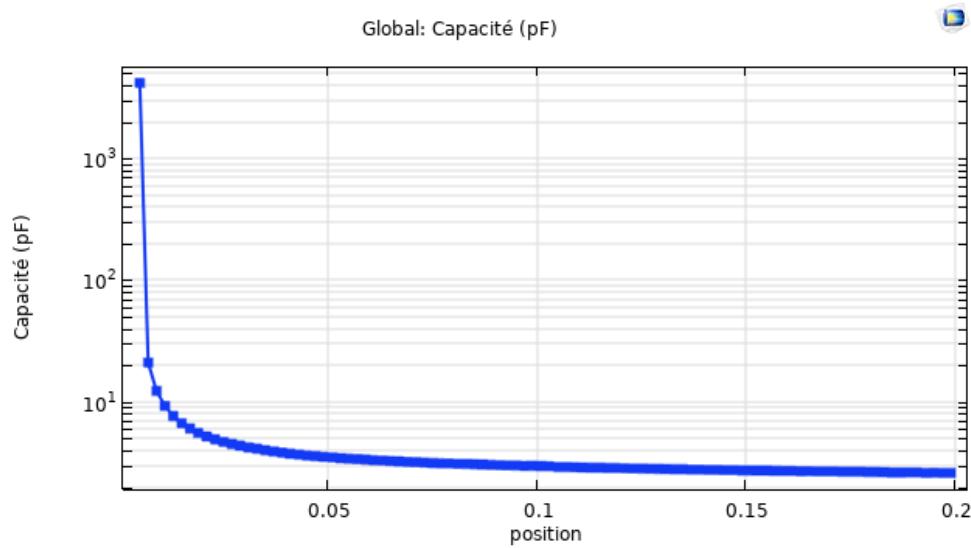


FIGURE 5 – Résultat de la simulation Capacité (pF) X position (m).

L'étude du potentiel électrique par ligne de courant du même système a également été réalisée et les résultats sont présentés dans les Figures 6 et 7.

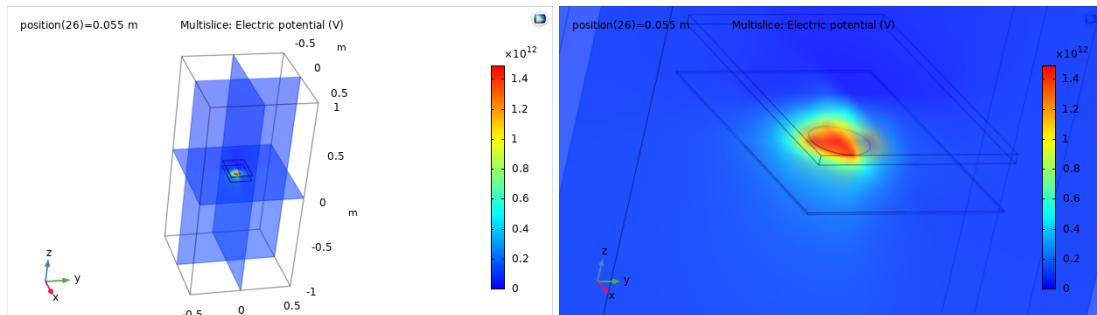


FIGURE 6 – Résultat de la simulation Potentiel Électrique.

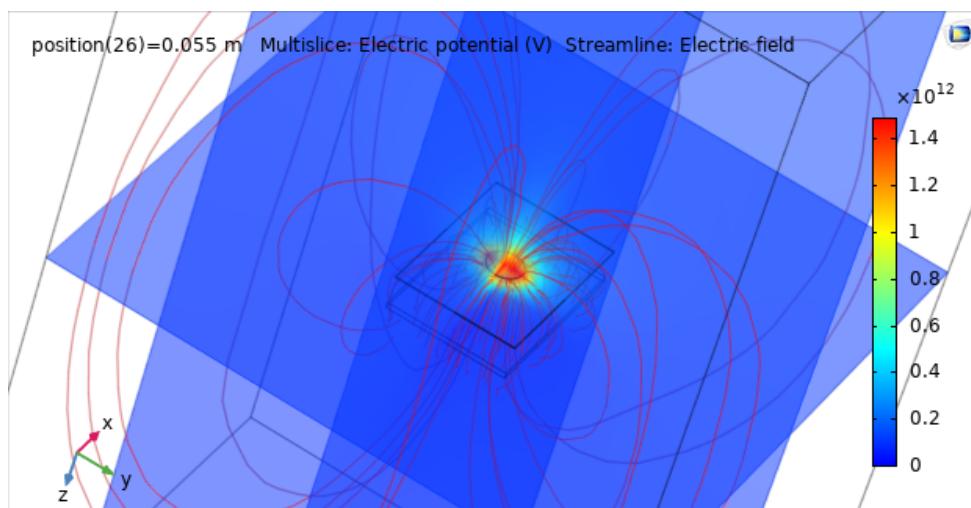


FIGURE 7 – Résultat de la simulation du Potentiel électrique avec les lignes du champ électrique.

2 Circuit Conditionneur

Le circuit de conditionnement d'acquisition capacitive fonctionne de manière à traduire l'ap- proche de la main vers la plaque en un changement de fréquence du système. De cette façon, le circuit de conditionnement reçoit un signal électrique d'entrée de la plaque de cuivre qui est transmis via un connecteur ZIF [3] qui permet une connexion facile aux électrodes. L'alimentation du circuit est effectuée à l'aide d'une carte microcontrôleur, STM32 Nucleo L432 KC [4], qui est capable d'interpréter le signal à travers des ports logiques. Le circuit doit avoir les caractéristiques suivantes :

01. Les dimension de la carte besoin de être identique à la carte microcontrôleur pour le système shield ;
02. La tension d'alimentation qui viens de le microcontrolleur varie entre 0V et 5V ;
03. La conection du élément flexible est fait pour le connecteur ZIF ;
04. La capacité nominale du élément flexible est de $10pF$;
05. Le signal de sortie est logique et varie entre 0V e 5V

Pour mesurer la capacité du circuit, nous avons utilisé un conditionneur basé sur le temps de charge d'une capacité à courant constante, qui résulte en une variation linéaire de la tension. En mesurant la capacité en utilisant a mesure de temps de charge à courant constante nous pouvons limiter l'impact des parasites et bruits captés par l'élément capacitif. Cette méthode consiste à utiliser le temps de charge de la capacité pour cadencer un oscillateur, dont on mesure ensuite la fréquence d'oscillation. Alors, elle permet de avoir un relation direct entre le temps de charge et la valeur de la capacité. Dans ce cas, la capacité à mesurer est reliée en parallèle avec une capacité de référence, et elle module l'état de charge de cette dernière.

Pour réaliser ce circuit, nous avons divisé le projet en étapes pour faciliter la compréhension. Il est également important de souligner que le circuit et la simulation sont réalisés avec le software 'PartQuest'. Tous le modèles étaient développer utilisent les ressources de le site y compris l'utilisation d'un bloc VHDL qui nous a permis de créer nos propres composants.

2.1 Analyse de la charge à courant constant

La première étape a consisté à étudier le fonctionnement de la capacité de $10pF$ à une source de courant pulsé constante. L'équation suivante règle le système :

$$I = C \frac{dV_c}{dt} \Rightarrow V_c = \frac{1}{C} \int_{t_1}^{t_2} Idt = \frac{1}{C} I (t_2 - t_1) = \frac{I}{C} \times \Delta T$$

$$V_{charge} = \frac{I}{C} \times T_{pulse}$$
(1)

Pour la simulation, nous avons :

$$V_{charge} = 2V, I = 10e6, C = 10pF$$

$$\text{Alors, } T_{pulse} = 10\mu s$$
(2)

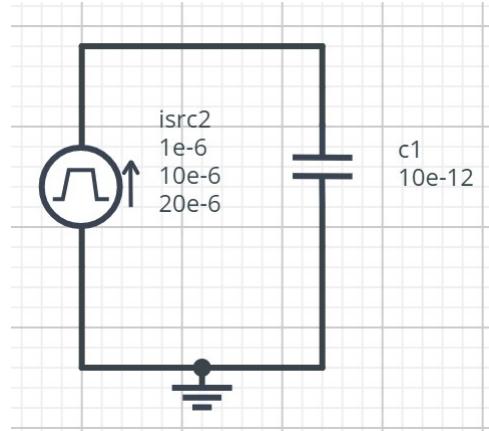


FIGURE 8 – Circuit à courant constant.

Dans la Figure 8 on peut observer le circuit et sa simulation représentée dans la Figure 9. Dans le résultat de la simulation, il est possible de remarquer le bon fonctionnement de la capacité, puisqu'il est chargé et déchargé lorsque le courant est respectivement maximum et minimum.

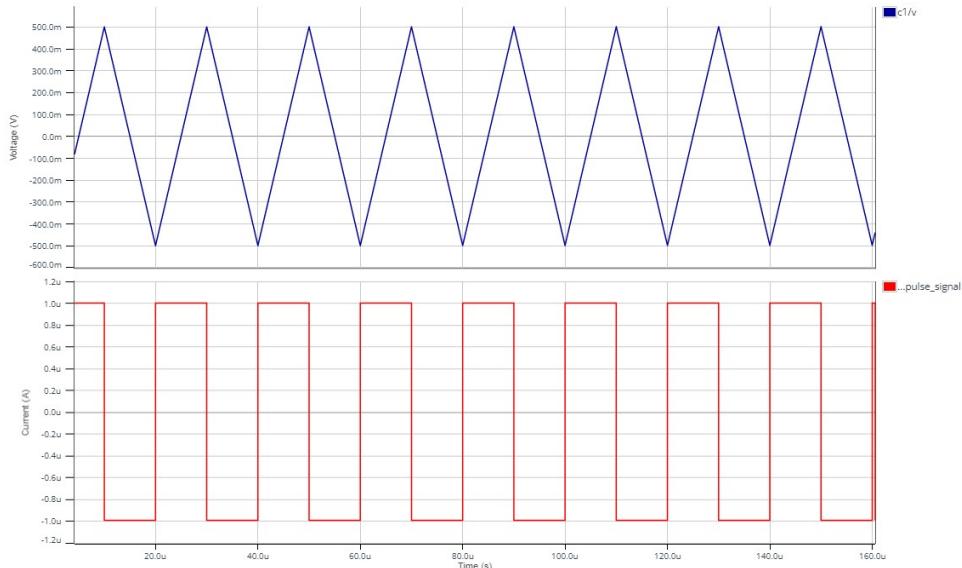


FIGURE 9 – Résultat de la simulation à courant constante.

2.2 Convertisseur tension courant

Nous avons ensuite travaillé sur le développement d'un convertisseur de tension en courant utilisant une capacité de $10pF$ à capacité constante. En appliquant une tension continue de $-5V$ à $5V$, le courant continu généré circule à travers la capacité. L'équation qui régit son fonctionnement est la suivante :

$$V^- = \frac{R_2}{R_1 + R_2} \times V_{\text{out}} = \frac{V_{\text{out}}}{2}, \text{ comme } R_1 = R_2 \text{ et } i^- \approx 0$$

Alors, $i_4 = i_3 + i_c$ (ayant $i^+ \approx 0$)

$$\frac{V_{\text{in}} - V^+}{R_4} = \frac{V^+ - V_{\text{out}}}{R_3} + i_c \text{ avec } R_3 = R_4 = n \cdot R_1$$

$$\text{Donc, } \frac{V_{\text{in}} - 2V^+ + V_{\text{out}}}{R_4} = i_c \Rightarrow V_{\text{in}} - 2V^+ + V_{\text{out}} = R_4 \cdot i_c \quad (3)$$

Or $V^+ = V^- = \frac{V_{\text{out}}}{2}$ alors,

$$V_{\text{in}} - V_{\text{out}} + V_{\text{out}} = R_4 \cdot i_c$$

$$i_c = \frac{V_{\text{in}}}{R_4}$$

Pour notre cas, nous avons que : $R4 = R3 = 2,5e6\Omega$

La Figure 10 montre le circuit et la Figure 11 montre le résultat de sa simulation. Nous pouvons y voir que le régle $V_{\text{out}} = 2 * V_C$ est respectée.

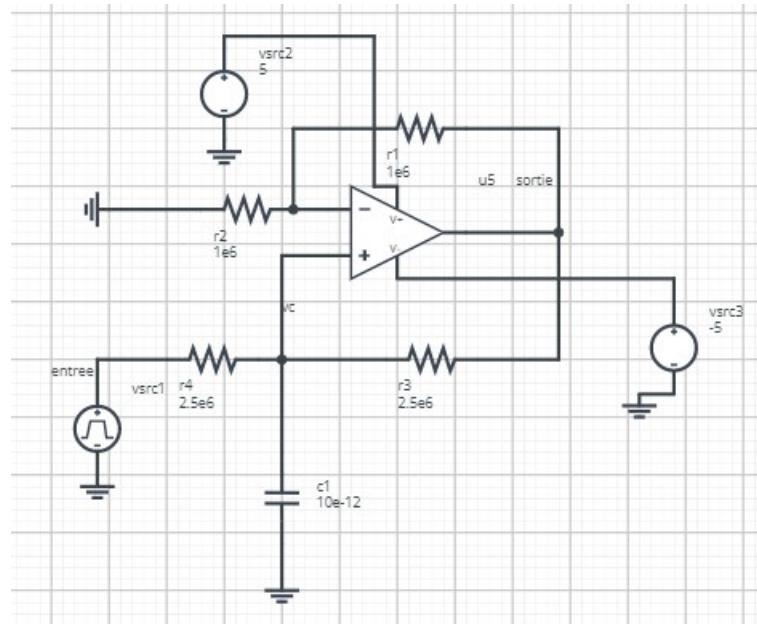


FIGURE 10 – Circuit convertisseur tension courant.

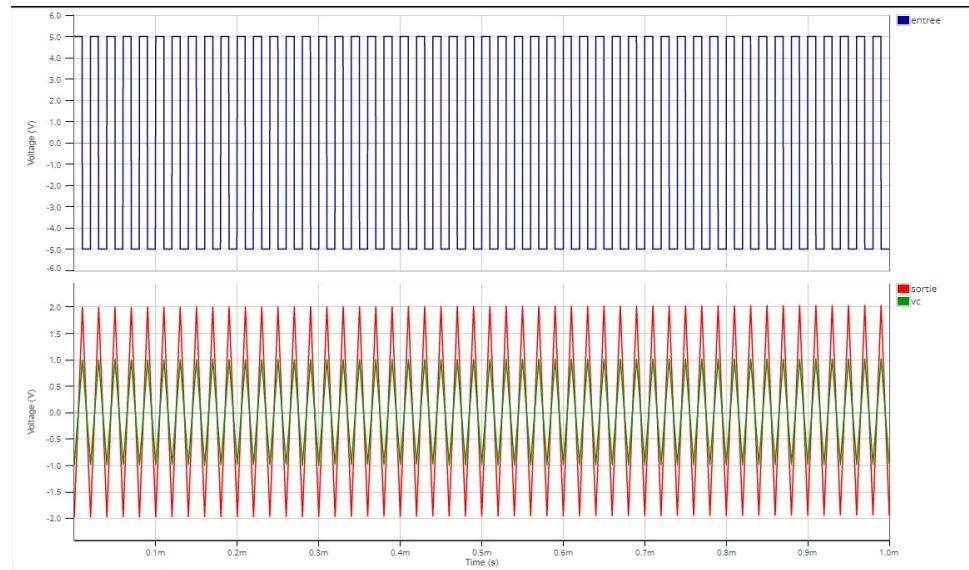


FIGURE 11 – Résultat de la simulation du convertisseur tension courant.

2.3 Comparateur à hystérésis

Pour que la sortie du système soit une onde carrée, nous utilisons le comparateur à hystérésis à la fin du circuit précédent. Pour dimensionner les résistances $R5$ et $R6$ pour qu'elles aient une tension de seuil égale au quart de la tension d'alimentation nous avons utilisé la formule suivante :

$$V_{\text{seuil}}^+ = \frac{R6}{R5 + R6} \times V_{\text{sortie}}^+ \text{ et } V_{\text{seuil}}^- = \frac{R6}{R5 + R6} \times V_{\text{sortie}}^-$$

$$V_{\text{seuil}} = \frac{V_{\text{sortie}}}{4}$$
(4)

Alors, nous avons que $R5 = 3 * R6$ et nous avons choisi $R5 = 3e5\Omega$ et $R6 = 1e5\Omega$. Dans la Figure 12 nous pouvons observer le circuit et sa simulation est montre dans la Figure 13. Le résultat obtenu est celui attendu, à savoir une onde carrée qui varie entre 5V et -5V quand la capacité charge et décharge, respectivement.

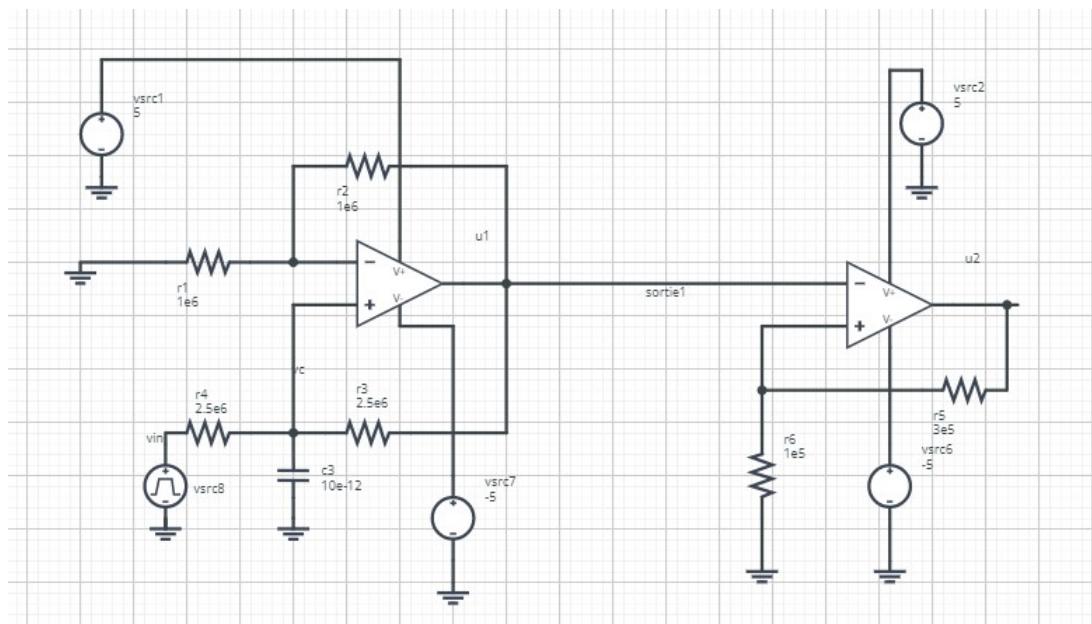


FIGURE 12 – Circuit convertisseur avec le comparateur à hystérésis.

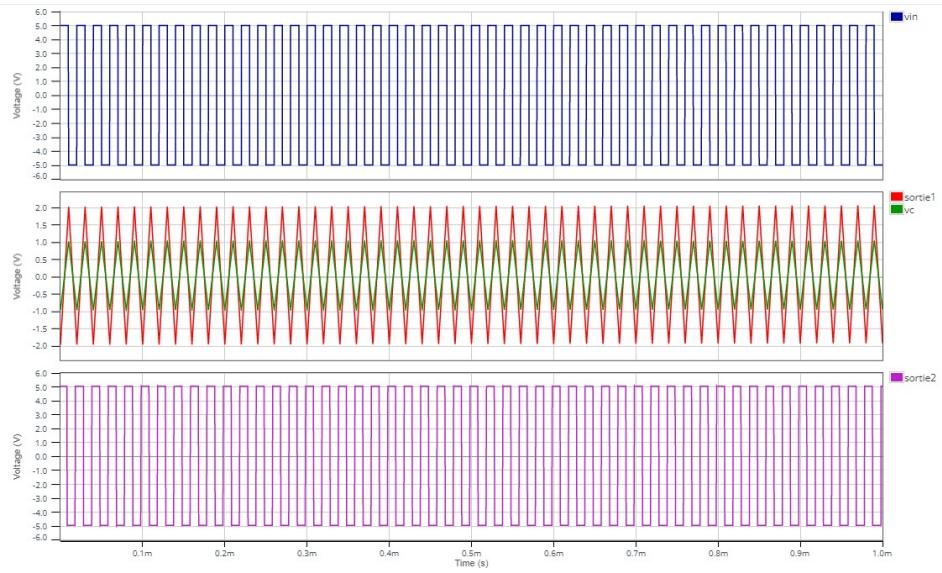


FIGURE 13 – Résultat de la simulation du convertisseur avec le comparateur à hystérésis.

2.4 Fonctionnement en boucle fermée

En actionnant le circuit de fermeture, le convertisseur fonctionnera de la même manière qu'un convertisseur à hystérésis. Cependant, la tension d'entrée et la capacité seront modifiées à chaque variation de tension de sortie, car celle-ci est bouclée dans le convertisseur de courant. Dans la Figure 14, nous pouvons voir le circuit et dans la Figure 15 sa simulation avec une capacité de $10pF$.

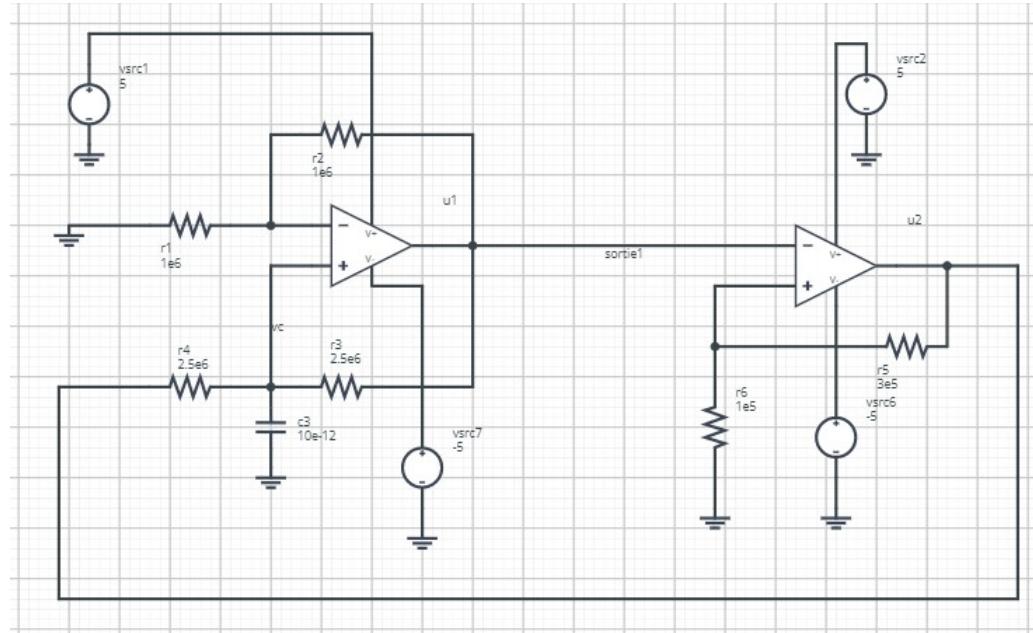


FIGURE 14 – Circuit en bouclé fermé.

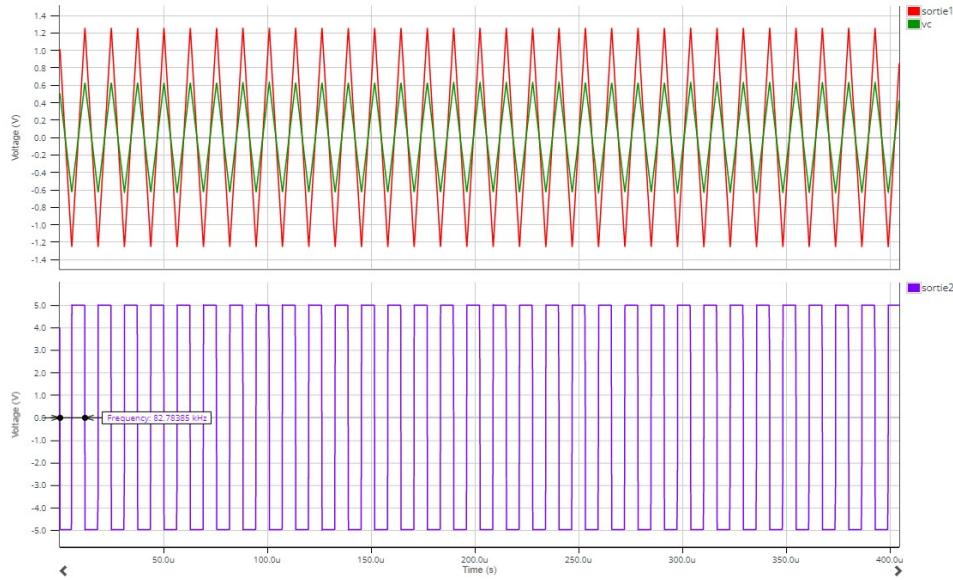


FIGURE 15 – Résultat de la simulation en bouclé fermée avec une capacité de 10pF.

D'équations qui régissent le circuit sont :

$$\begin{aligned}
 I &= C \frac{dV}{dt} \text{ donc } dV = \frac{I}{C} dt \Rightarrow \text{en intégrant, on aura } \Delta V = \frac{I}{C} * T \\
 &\text{Comme } I = \frac{V_{in}}{R_4} \\
 &\text{et } \Delta V = \frac{V_{out}}{2} \\
 &\frac{V_{out}}{2} = \frac{V_{in} * T}{R_4 * C} \text{ avec } V_{in} = V_{sortie} \\
 &\text{Alors, } \frac{1}{2} = \frac{T}{R_4 * C} \rightarrow \frac{1}{T} = \frac{2}{R_4 * C} \\
 &\text{Donc, } f = \frac{2}{R_4 * C}
 \end{aligned} \tag{5}$$

Pour une capacité de 10pF , nous avons $f = \frac{2}{2.5e6 * 10e-12} = 80\text{kHz}$ que est très proche de le résultat obtenu. Quand nous varions la capacité de 5pF à 10pF , nous avons obtenu le résultat de la Figure 16, où le premier point de chaque graphique est le condensateur de 5pF et sa valeur augmente de 1pF en 1pF .

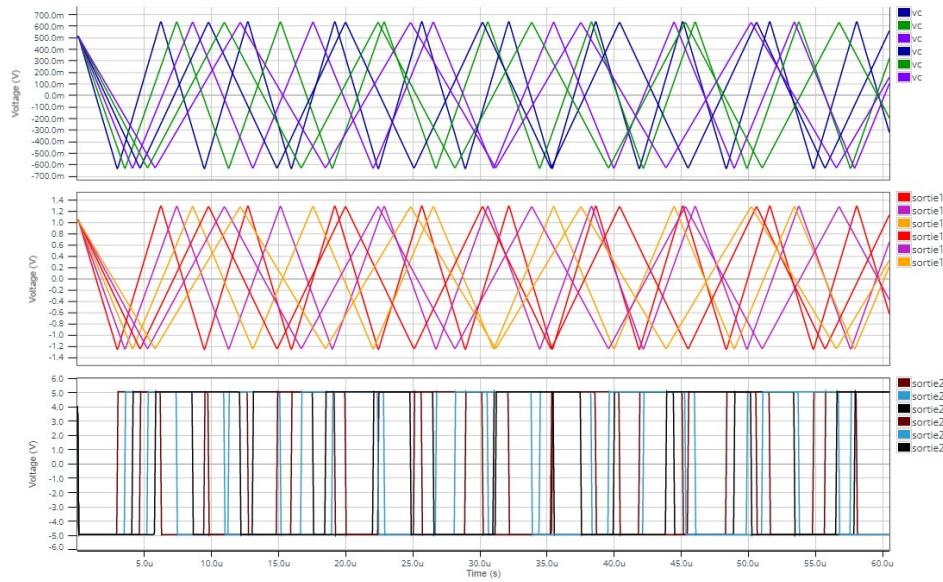


FIGURE 16 – Résultat de la simulation du convertisseur tension courant avec variation de la capacité.

Ainsi, à mesure que la capacité augmente, le temps de charge augmente également, ce qui entraîne une atteinte plus lente de la tension seuil. Par conséquent, la période à la sortie du comparateur à hystérésis est plus longue et la fréquence est plus faible. Cette relation inverse entre la capacité et la fréquence s'explique ainsi.

2.5 Fonctionnement en alimentation unique

Étant donné que notre prototype fonctionne avec un microcontrôleur STM32, sa plage de fonctionnement est de 0V à 5V. Pour ce faire, nous utilisons un pont diviseur. Afin que le pont diviseur se comporte de manière appropriée, nous allons utiliser un amplificateur opérationnel en tant que suiveur de tension, avec une impédance d'entrée suffisamment élevée pour empêcher tout courant de sortir du pont diviseur, et une impédance de sortie suffisamment faible pour ne pas perturber les composants déjà présents dans notre système. Le circuit est représenté pour la Figure 17 e sa résultat dans la Figure 18.

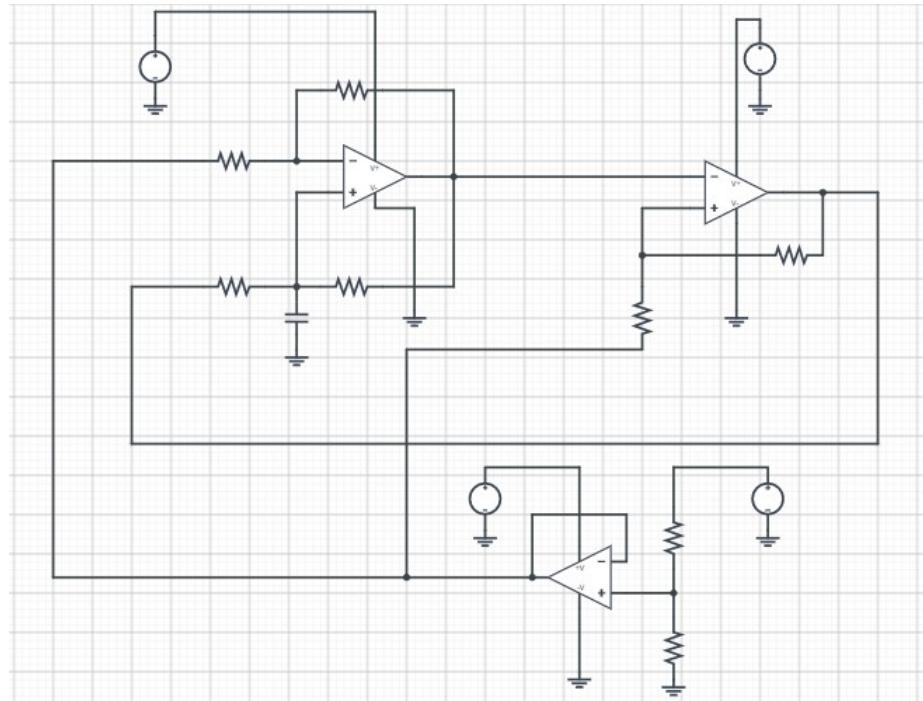


FIGURE 17 – Circuit en alimentation unique.

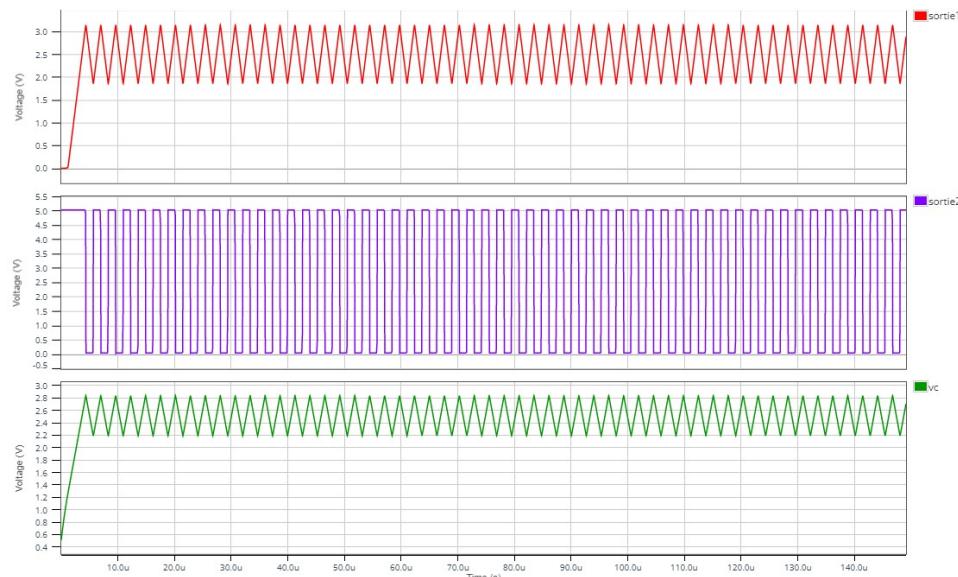


FIGURE 18 – Résultat de la simulation en alimentation unique.

2.6 Prise en compte des parasites et simulation d'une détection de distance

Pour représenter l'approche d'une main vers une plaque métallique, il est nécessaire de travailler avec un bloc en VHDL à l'intérieur du logiciel *PartQuest*, le bloc HBM (Human Body Model). Le code VHDL être consulté en Annexe A. Pour une utilisation pratique, il est essentiel de prendre en compte certaines considérations pour garantir le bon fonctionnement de l'élément sensible. Tout d'abord, il est important de prendre en compte la présence d'une capacité parasite en parallèle avec l'élément sensible, qui peut être estimée à environ $2pF$. Cette prise en compte de la capacité parasite et de l'équivalent du corps humain permettra d'obtenir un circuit de simulation plus réaliste, comme illustré dans la Figure 19 e sa résultat est dans la Figure 20.

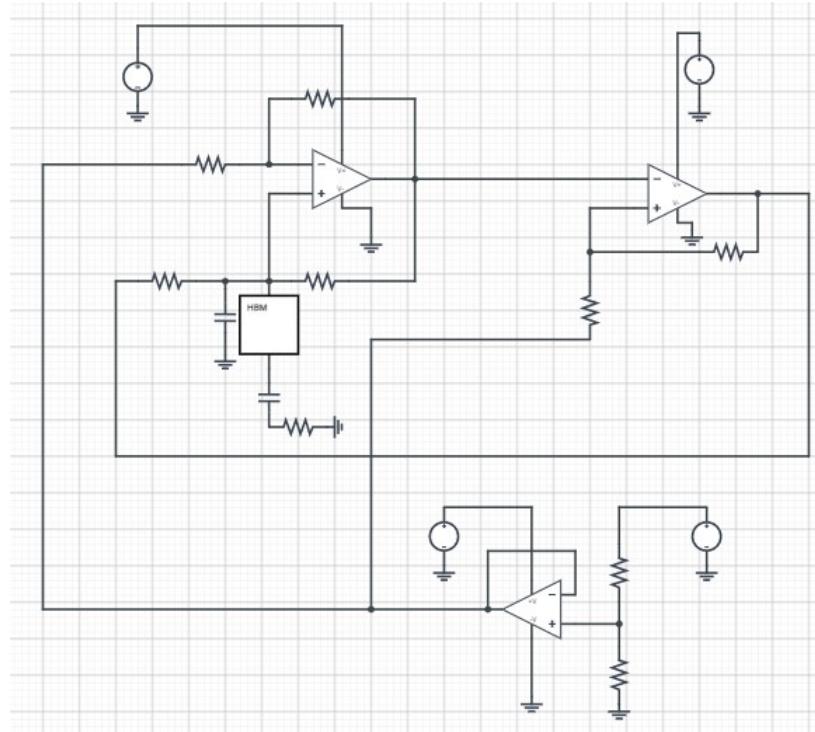


FIGURE 19 – Circuit complet.

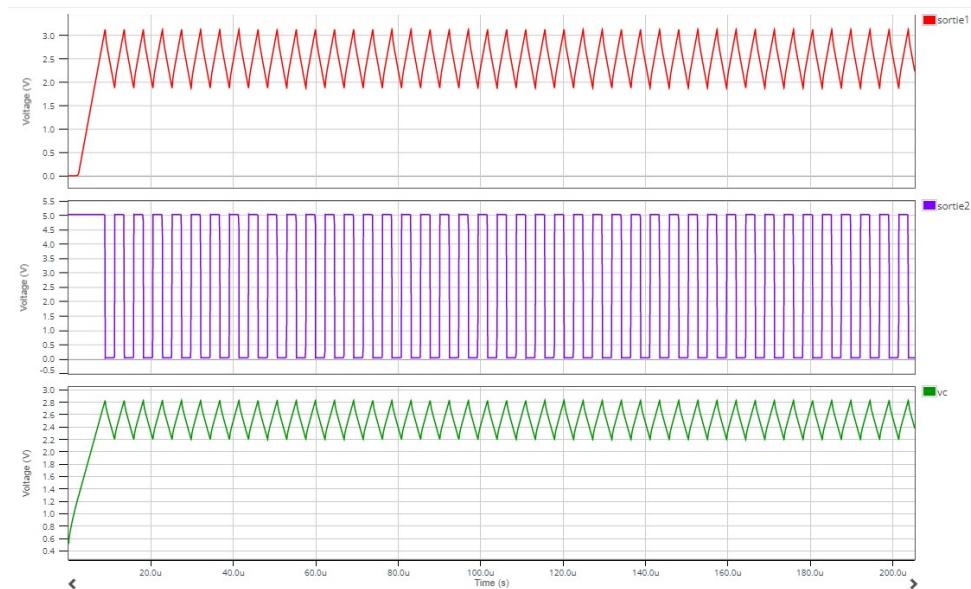


FIGURE 20 – Résultat de la simulation du circuit complet.

La variation de la distance de la main entraîne une variation de la fréquence du système. Plusieurs simulations ont été réalisées pour obtenir le graphique dans la Figure 21, qui montre la variation de la fréquence en fonction de la distance allant de 1cm à 15cm par incrément de 0,25cm. Cependant, nous pouvons constater que la variation de fréquence devient moins perceptible au-delà de 7cm, avec une diminution de la pente. Cela peut rendre la détection de la variation de fréquence plus difficile pour notre microcontrôleur.

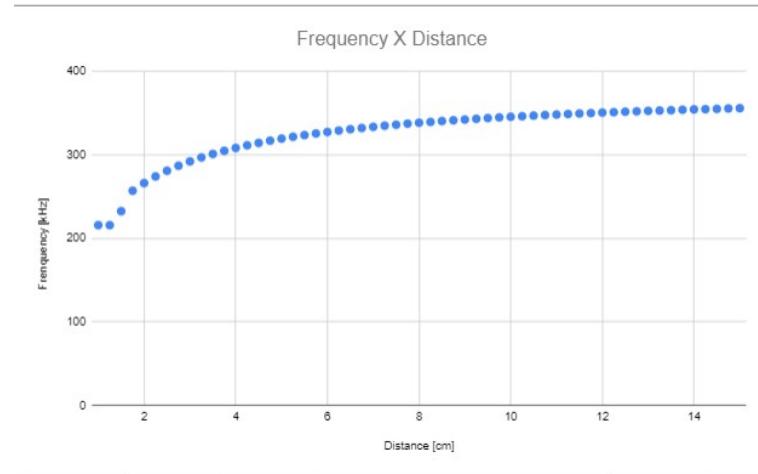


FIGURE 21 – Résultat de la simulation du circuit complet avec une variation de la distance de la main.

2.7 Assemblage

Après avoir développé le circuit dans le logiciel PartQuest, nous pouvons le développer sous sa forme physique. Pour ce faire, nous avons obtenu des résistances planaires de $1k\Omega$, $10k\Omega$ et $1M\Omega$ qui ont été soudées sur le circuit imprimé fourni par l'école. La Figure 22 montre la carte soudée.

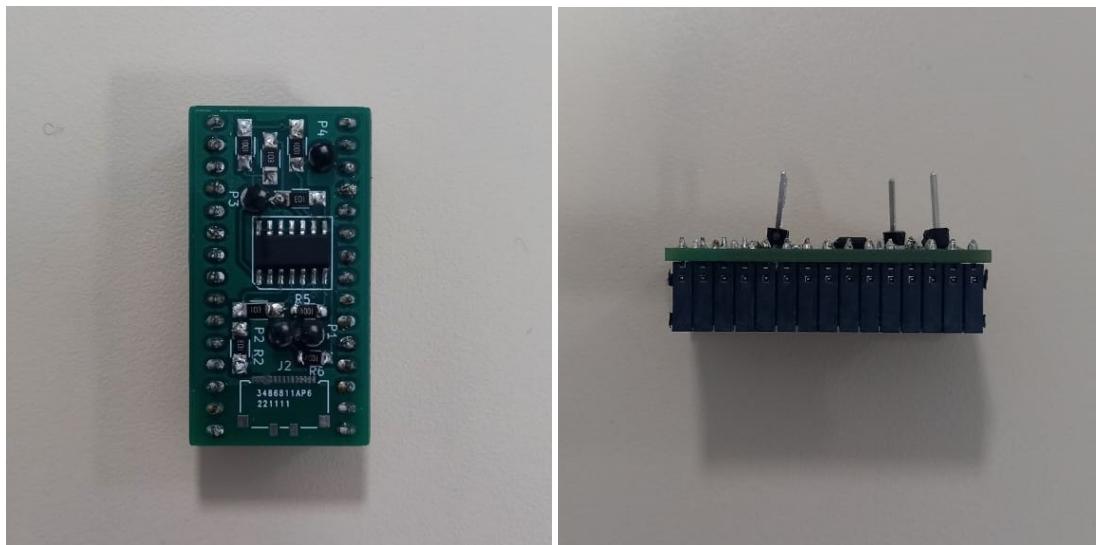


FIGURE 22 – Carte du circuit imprimé soudé.

En mesurant sa fréquence à l'oscilloscope, on obtient le résultat de la figure 23.

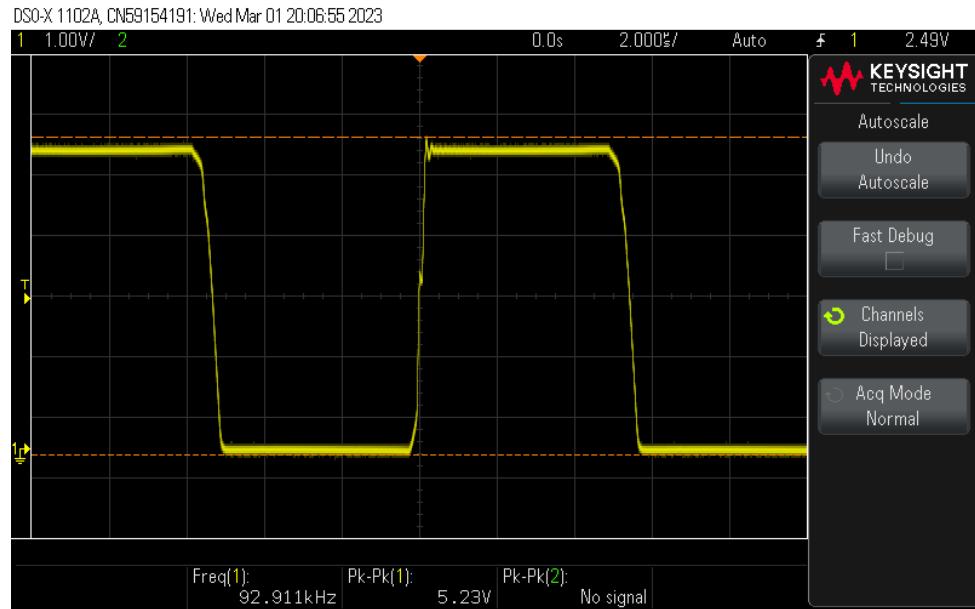


FIGURE 23 – Fréquence de la carte imprimée et soudé.

Ensuite, nous avons connecté le circuit à une capacité de 10pF , comme la Figure 24 nous montre et sa résultat dans la Figure 25

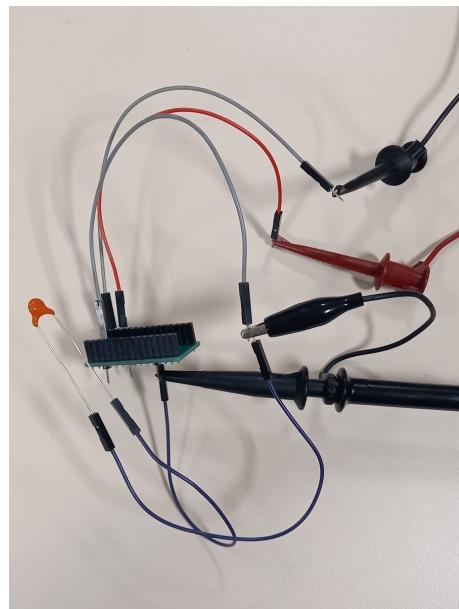


FIGURE 24 – Circuit connecté avec la capacité de 10 pF .

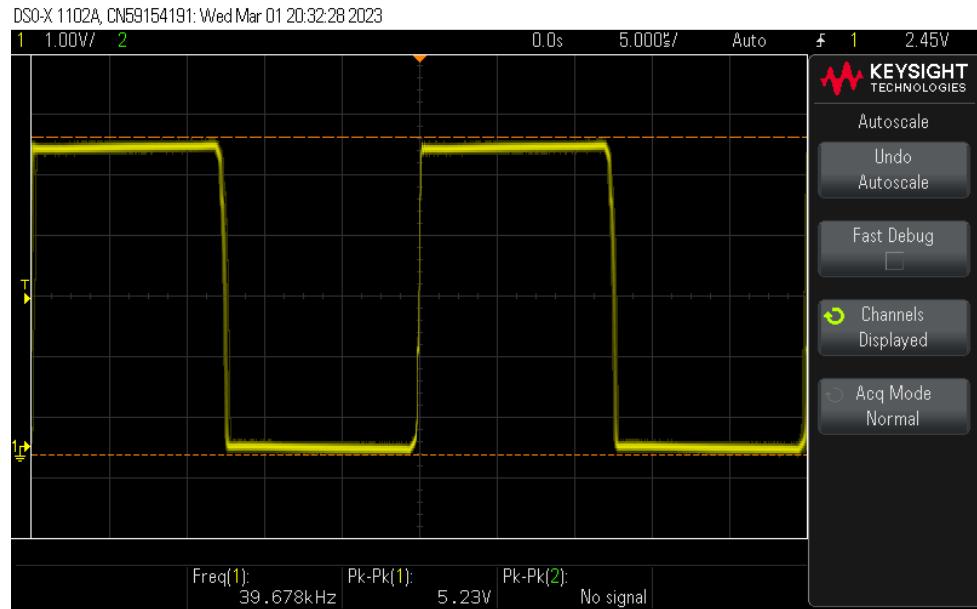


FIGURE 25 – Fréquence de la carte avec capacité de 10 pF.

Nous remplaçons donc le condensateur par la plaque métallique, comme le montre la Figure 26, de sorte que possible de mettre au point des tests pour le moment où la main s'approche de la plaque sensible.

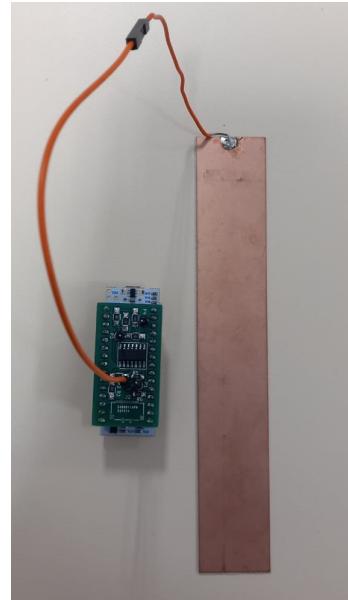


FIGURE 26 – Circuit connecté avec la plaque métallique.

Les fréquences qui en résultent sont indiquées dans la Figure 27.

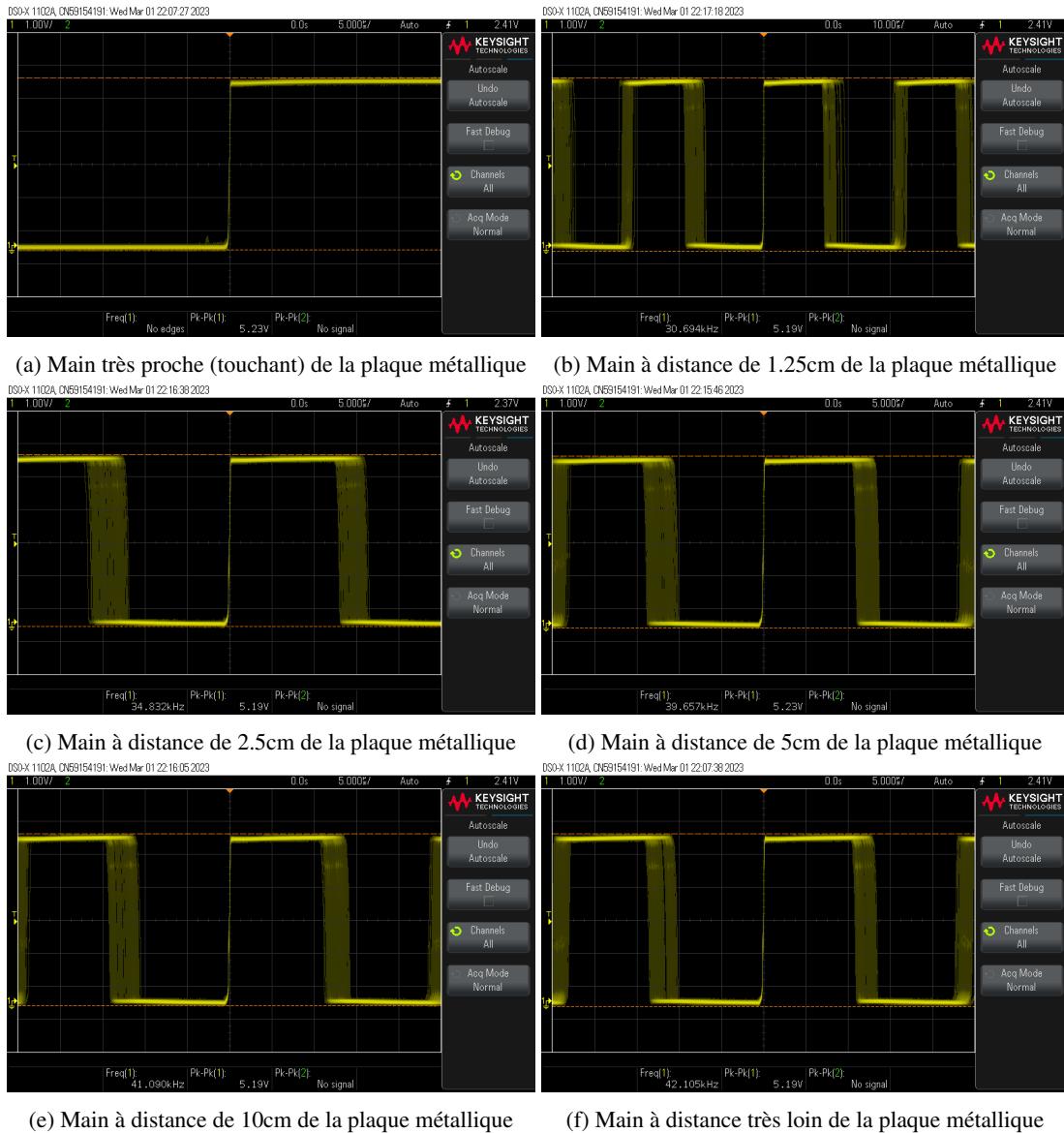


FIGURE 27 – Fréquences obtenu pour l'approche de la main à plaque.

Comme on l'a déjà vu dans les études réalisées en simulation, on constate que la fréquence diminue à mesure que la main s'approche de la plaque et que, par conséquent, la capacité augmente.

3 Carte Microcontrôleur

Après avoir fabriqué le circuit conditionneur, le microcontrôleur STM32F301K8T6 sera utilisé pour capturer le signal de sortie de ce circuit, puisque ce signal a une fréquence qui est liée à la distance entre le corps et la plaque métallique. Ainsi, avec cette fréquence, il sera possible de déterminer la capacité générée dans le système.

3.1 Méthode pour identifier la fréquence

Le signal de sortie du circuit conditionneur sera capturé par le microcontrôleur. Ce signal est un signal PWM qui consiste d'une série d'impulsions électriques de largeur variable. De cette manière, quand nous regardons la Figure 28, nous pouvons identifier sa fréquence à partir de deux points consécutifs. Par exemple, nous pouvons considérer ces deux points lorsque le signal est sur des fronts ascendants ou descendants. Ainsi, en effectuant la différence, nous obtenons une période de cette onde et ensuite, en calculant son inverse, nous obtenons la fréquence correspondant à l'onde.

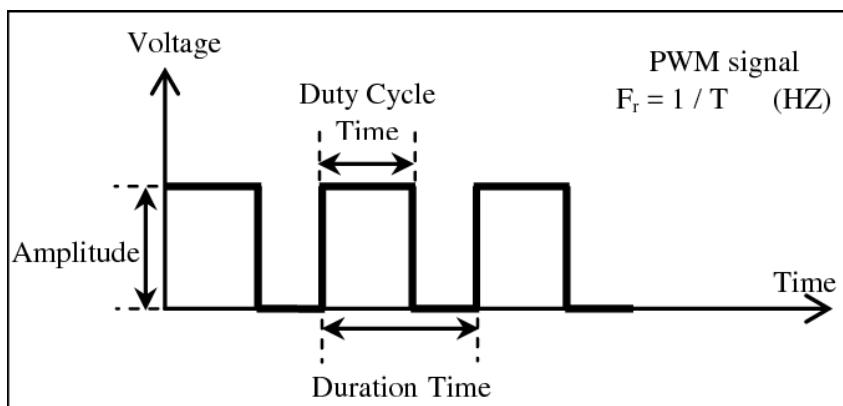


FIGURE 28 – Exemple de forme d'onde provenant du circuit conditionneur. [5]

3.2 Microcontrôleur STM32F301K8T6



FIGURE 29 – Microcontrôleur STM32F301K8T6 utilisé pour le système. [4]

Le STM32F301K8T6 est un microcontrôleur 32 bits basé sur l'architecture ARM Cortex-M4F. Il est fabriqué par STMicroelectronics et est largement utilisé dans les applications industrielles, automobiles, médicales et de grand public.

Certaines des principales caractéristiques du STM32F301K8T6 incluent :

- Architecture : ARM Cortex-M4F;
- Fréquence de fonctionnement : jusqu'à 72 MHz;

- Mémoire Flash : 64 Ko ;
- SRAM : 8 Ko ;
- Connectivité USB : USB 2.0 Full Speed ; et
- Ports d'entrée/sortie : 37 ports GPIO (General Purpose Input/Output) avec capacité d'entrée analogique et interface pour communication série.

Le STM32F301K8T6 est capable d'exécuter des tâches complexes de traitement de signal en temps réel, ce qui en fait un choix idéal pour les applications nécessitant des performances élevées et une faible consommation d'énergie dans un espace réduit.

Nous l'utiliserons donc pour détecter la présence d'un corps à partir de la fréquence provenant du circuit conditionneur, transmettre les données obtenues à partir de la fréquence lue par la communication série de l'UART, identifier les boutons sur lesquels on appuie pour émettre une fréquence à l'aide du module PWM afin de produire un son dans le buzzer.

3.3 Réglages pour la lecture de la fréquence

Pour effectuer la lecture de la fréquence, le microcontrôleur STM32F301K8T6 a été configuré à partir du logiciel STM32CubeIDE, qui est un environnement de développement intégré (IDE) gratuit qui inclut le compilateur, le débogueur et d'autres outils nécessaires pour développer et configurer des projets pour les microcontrôleurs ST.

Ainsi, le microcontrôleur a été configuré avec les paramètres suivants :

- Le TIMER 2 sera utilisé pour détecter les fronts montants, de sorte qu'il est possible de mesurer le temps auquel une période se produit. De cette manière, il sera possible de calculer la fréquence générée par le circuit conditionneur.

(a) Configuration des paramètres.

(b) Configuration de NVIC.

FIGURE 30 – Configuration du TIM2.

- L'USART 2 sera utilisé pour exporter les données de fréquence et de capacitance vers un terminal série. Il sera configuré pour fonctionner en mode asynchrone.

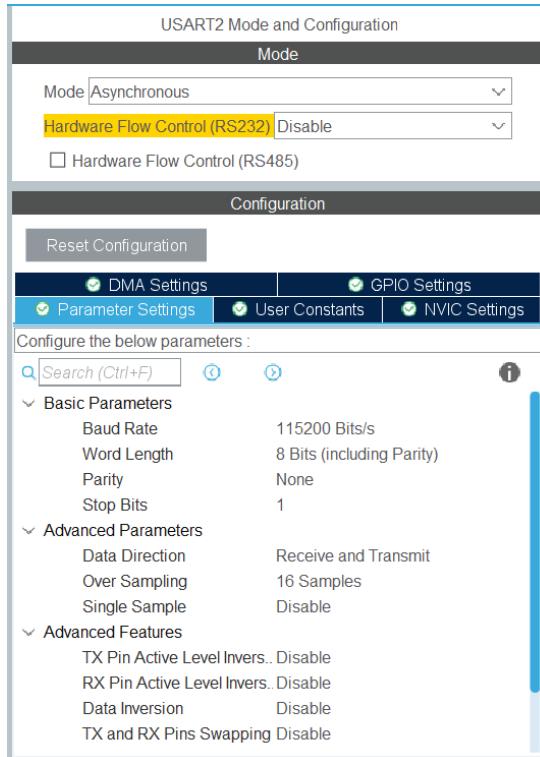


FIGURE 31 – Configuration du USART2.

Ensuite, les fichiers de configuration du microcontrôleur sont générés et, parmi eux, le fichier *main.c* permet de lire la fréquence. Alors, les fonctions suivantes seront utilisées pour atteindre cet objectif :

- *HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim)* : Cette fonction est appelée lorsqu'un signal d'entrée est détecté sur la broche de capture associée au timer qui, dans ce cas, sera activé lorsqu'il y a un front montant. Ainsi, la fréquence générée par le circuit conditionneur sera calculée.
- *HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_1)* : Cette fonction permet de lire la valeur capturée par le timer sur le canal 1.
- *HAL_UART_Transmit (UART_HandleTypeDef *huart, uint8_t *pData, uint16_t Size, uint32_t Timeout)* : Cette fonction est utilisée pour envoyer des données via un port série UART.
- *HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef* htim)* : Cette fonction est appelée lorsqu'un événement de saturation du timer se produit, dans ce cas, nous utilisons le timer 2.

3.4 Développement du code - Lecture de la fréquence

La fréquence sera calculée en utilisant la différence entre le temps des fronts montants, la fréquence de fonctionnement du microcontrôleur et les paramètres du Timer2. Ainsi, l'équation 6 calcule la différence entre les fronts et l'équation 7 modélise le comportement de la fréquence provenant du circuit conditionneur.

$$\Delta t = \text{sample}(NbCycle) - \text{sample}(0) + OF \cdot ARR \quad (6)$$

$$f_{OUT} = \frac{NbCycle \cdot f_{APB1}}{(PSC + 1) \cdot \Delta t} \quad (7)$$

Où :

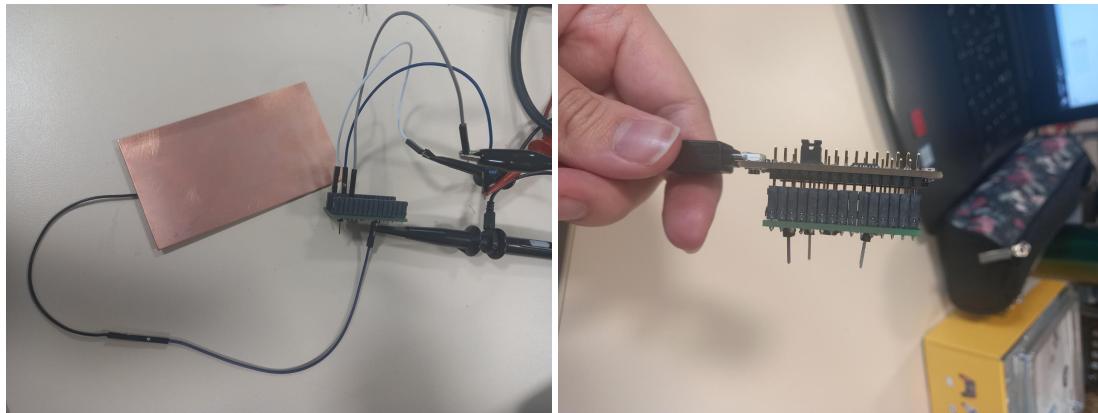
- f_{OUT} : Fréquence provenant du circuit conditionneur ;
- f_{APB1} : Fréquence de fonctionnement du APB1 ;
- Δt : Différence entre les fronts montants ;
- OF : En cas de dépassement, la variable OverFlow sera égale à 1 et corrigera le calcul de la fréquence ;
- NbCycle : Nombre de cycles ;
- PSC : Prescale est définie dans les paramètres du TIMER2, il sert à régler la fréquence de base du Timer2 ;
- ARR : AutoReload est définie dans les paramètres du TIMER2, il sert à contrôler la génération d'interruptions périodiques sur une timer ;
- $\text{sample}(NbCycle)$: Temps capturé sur la dernière bord ; et
- $\text{sample}(0)$: Temps capturé sur la première bord.

À partir des configurations susmentionnées, les valeurs des variables seront les suivantes :

- $f_{APB1} = 64\text{MHz}$;
- $\text{NbCycle} = 4$;
- $\text{PSC} = 0$; et
- $\text{ARR} = 4294967295$.

Ainsi, le code utilisé pour effectuer la lecture de la fréquence est disponible dans l'annexe B de ce document.

3.5 Simulation du code - Lecture de la fréquence



(a) Circuit conditionneur connecté à une plaque métal- (b) Microcontrôleur connecté à l'ordinateur pour réaliser
lique. la mesure de la fréquence.

FIGURE 32 – Test du code pour la lecture de la fréquence.

Nous avons programmé le microcontrôleur avec le code avec les codes développés dans le sujet précédent. Ainsi, pour vérifier que le microcontrôleur lit correctement la fréquence provenant du circuit conditionneur, un oscilloscope a été utilisé pour mesurer la fréquence. De cette façon, nous pouvons comparer les lectures faites par le microcontrôleur - dont nous pouvons voir la valeur de la variable *Frequency* en déboguant - et la fréquence mesurée avec l'oscilloscope.

Voici les résultats obtenus par chaque moyen d'acquisition.

- Logiciel STM32CubeIDE :

Variables Breakpoi... Expressi... Registers Live Exp... SFRs		
Expression	Type	Value
Frequency	volatile uint32_t	44467
Add new expression		

FIGURE 33 – Lecture des fréquences à une distance supérieure à 10 cm.

Variables Breakpoi... Expressi... Registers Live Exp... SFRs		
Expression	Type	Value
Frequency	volatile uint32_t	35565
Add new expression		

FIGURE 34 – Lecture des fréquences à une distance inférieure à 10 cm.

- Oscilloscope :

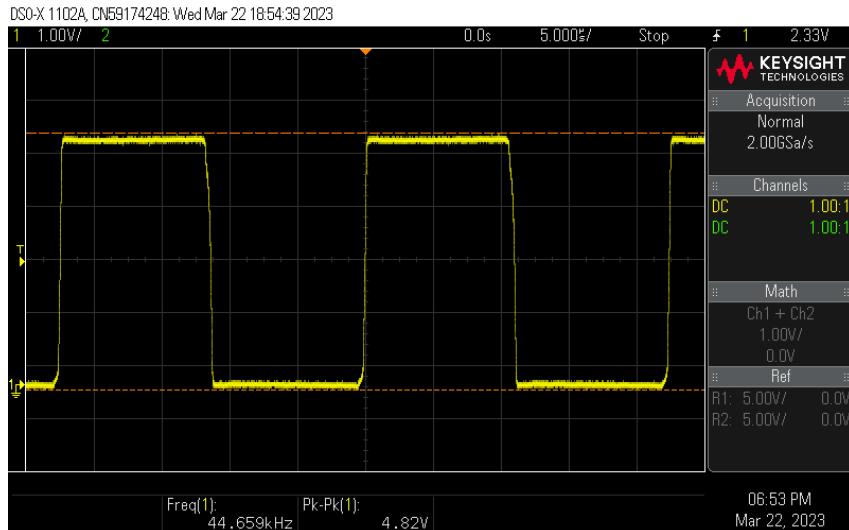


FIGURE 35 – Lecture des fréquences à une distance supérieure à 10 cm.

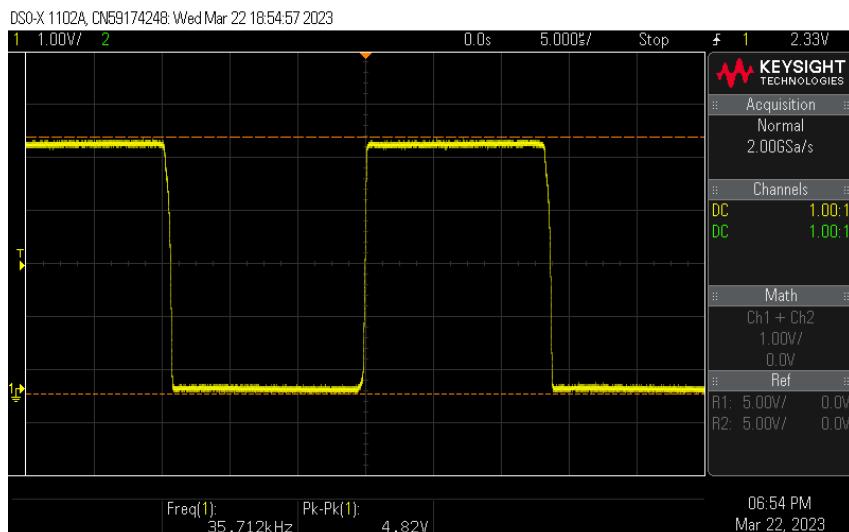


FIGURE 36 – Lecture des fréquences à une distance inférieure à 10 cm.

Nous pouvons donc voir sur les images ci-dessus les valeurs de fréquence enregistrées dans le tableau 2. Ce tableau montre encore la moyenne relative entre la valeur obtenue à l'oscilloscope et la valeur enregistrée dans le microcontrôleur.

	Oscilloscope	Logiciel STM32CubeIDE	Moyenne Relative
Inférieur à 10 cm	35,712 kHz	35,565 kHz	0,41%
Supérieur à 10 cm	44,659 kHz	44,467 kHz	0,43%

TABLE 2 – Résumé des fréquences obtenues.

De cette manière, nous pouvons prouver que le microcontrôleur a une bonne lecture de la fréquence provenant du circuit conditionneur puisque la moyenne relative présente une valeur très faible, qui peut être négligée. Alors, la fonction *check_frequency()* a été créée pour vérifier la présence ou non de la main à partir de la lecture de la fréquence, qui a été fixée à 42 kHz (voir l'annexe C).

4 Carte Commande Actionneurs

L'air guitar sera développé de manière à simuler une vraie guitare. Pour ce faire, pour que l'air guitar produise une note de musique, le microcontrôleur devra détecter la présence de la main et les boutons sur lesquels on appuie. Ainsi, cette action simule la même manipulation qu'un élève ferait s'il était en possession d'une guitare.

Ainsi, comme nous avons déjà développé les fonctions de lecture de la fréquence et d'identification de la présence de la main, nous devons développer deux autres fonctions : une fonction de détection des boutons pressés et une fonction de production du son à partir de la combinaison des boutons pressés.

L'air guitar a été développé de manière à produire 6 notes de musique qui sont : Mi grave, Lá, Ré, Sol, Si et Mi aiguë (basée sur la guitare [6]).

4.1 Développement du code - Lectures des broches d'entrée

Pour sélectionner la note qui sera déclenchée, nous devons d'abord activer les broches d'entrée digitale. Les broches sélectionnées sont les suivantes : PB1, PB6 et PB7. Ainsi, le GPIO a été configuré pour satisfaire à notre demande.

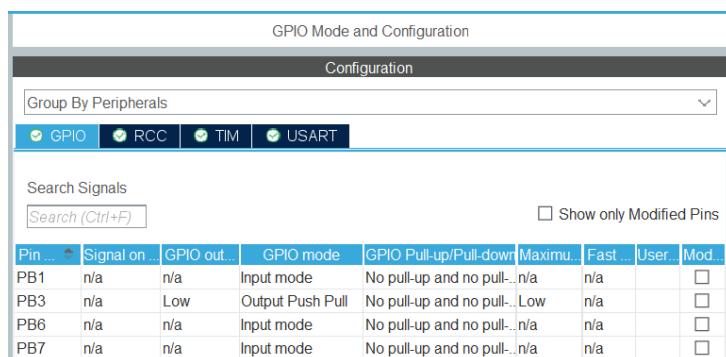


FIGURE 37 – Configuration du GPIO.

Ainsi, nous utiliserons la fonction `HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_X)` pour lire l'entrée de la broche, où X indique le numéro de la broche. Si l'entrée a un état logique élevé, une valeur de 1 sera enregistrée pour cette broche, sinon une valeur de 0 sera enregistrée, et à la fin, une somme des valeurs obtenues est faite pour décider quel son sera produit. De plus, les boutons sont montés en mode *Pull-Down* [7].

Donc, le code utilisé pour effectuer la lecture des broches d'entrée est disponible dans l'annexe D de ce document.

Le son produit par le buzzer sera émis en fonction de la combinaison des lectures des broches d'entrée. Nous utiliserons 3 entrées digitales pour réaliser cette combinaison. Le tableau 3 montre tous les cas possibles de combinaison des 3 boutons.

Note Musicale	Broches		
	Broche 1	Broche 2	Broche 3
Mi (grave) - E	1	0	0
Lá - A	0	1	0
Ré - D	0	0	1
Sol - G	1	1	0
Si - B	0	1	1
Mi (aiguë) - E	1	0	1

TABLE 3 – Table de vérité pour la génération de notes musicales, où 0 indique l'état logique bas et 1 l'état logique haut.

4.2 Buzzers



FIGURE 38 – Représentation d'un buzzer. [8]

Le buzzer est un dispositif qui convertit l'énergie électrique en énergie sonore à travers un élément vibratoire, tel qu'un diaphragme ou un cône. Nous savons que la fréquence du son émis par le buzzer est directement liée à la fréquence du courant électrique qui le traverse. Lorsque le courant électrique est appliqué au buzzer, il fait osciller l'élément vibratoire, produisant des ondes sonores qui correspondent à la fréquence du courant électrique.

La fréquence de fonctionnement du buzzer peut être contrôlée par la fréquence du courant électrique qui lui est appliquée. Par exemple, si le courant électrique oscille à une fréquence de 1 kHz, le buzzer produira un son avec cette même fréquence. Si la fréquence du courant électrique est modifiée pour atteindre 2 kHz, le buzzer produira un son avec cette nouvelle fréquence. Cette relation directe entre la fréquence du courant électrique et la fréquence du son produit rend le buzzer un outil utile dans les applications qui nécessitent la génération de signaux sonores à des fréquences spécifiques, comme dans les alarmes, les signalisations et les dispositifs de communication sonore.

4.3 Réglages pour générer un signal PWM

Pour que le buzzer émette différentes notes musicales, le Timer1 sera utilisé pour générer un signal de sortie PWM. Le Timer1 a la configuration suivante :

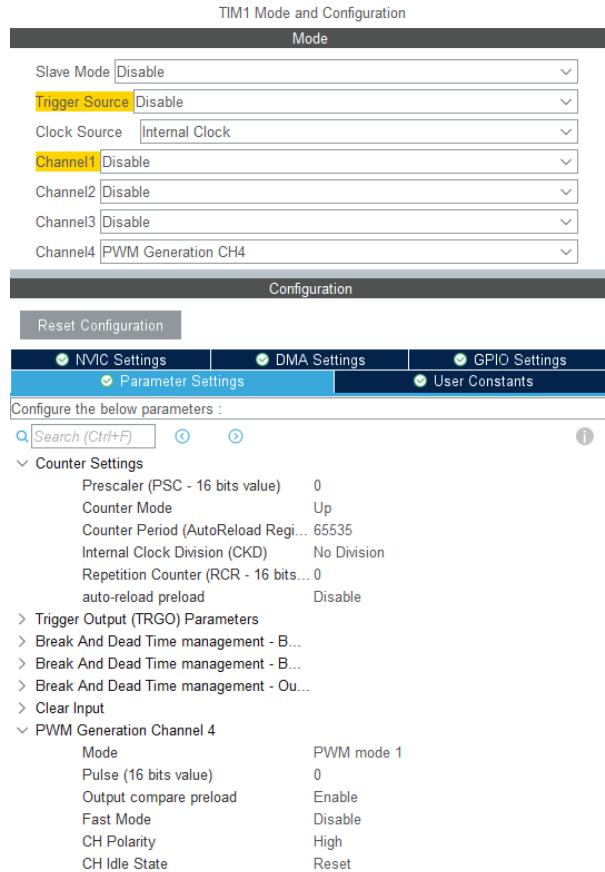


FIGURE 39 – Configuration du TIM1.

La broche PA11 est responsable de la sortie du signal PWM (TIMER1/Channel4).

4.4 Développement du code - Génération de signaux de sortie PWM

Pour déterminer la fréquence de sortie, nous devons en premier lieu déterminer la fréquence de base du Timer1. L'équation 8 permet de définir la fréquence de base.

$$f_{\text{BASE}} = \frac{f_{\text{APB}1}}{(PSC + 1)} \quad (8)$$

Où :

- f_{BASE} : Fréquence base ;
- $f_{\text{APB}1}$: Fréquence de fonctionnement du APB1 ; et
- PSC : Prescale est définie dans les paramètres du Timer1, il sert à régler la fréquence de base du Timer1.

Ainsi, sachant que la fréquence du APB1 est de 64MHz, nous définissons que la fréquence de base sera de 20kHz. Par conséquent, selon l'équation 8, la valeur Prescale doit être égale à 3199.

Après avoir déterminé la fréquence base de fonctionnement du Timer1, nous devons utiliser les équations suivantes pour régler la sortie PWM :

- Période du signal PWM :

$$T_{PWM} = \frac{1}{f_{PWM}} \quad (9)$$

Où :

- T_{PWM} : Période du signal PWM ; et
- f_{PWM} : Fréquence du signal PWM.

- Période du PWM :

$$ARR = T_{PWM} \cdot f_{BASE} \quad (10)$$

Où :

- ARR : AutoReload est définie dans les paramètres du Timer1, il sert à contrôler la génération d'interruptions périodiques sur une timer ;
- T_{PWM} : Période du signal PWM ; et
- f_{BASE} : Fréquence base.

- Valeur du rapport cyclique PWM en termes de cycles de timer pour agir à 50 % :

$$CCR4 = \frac{ARR}{2} \quad (11)$$

Où :

- CCR4 : CCR4 se réfère à le Pulse, c'est-à-dire le duty cycle, qui sera 50% ; et
- ARR : AutoReload est définie dans les paramètres du Timer1, il sert à contrôler la génération d'interruptions périodiques sur une timer.

Alors, le tableau 4 montre quelles sont les valeurs de ARR et CCR4 qui permettent d'obtenir les fréquences de sortie pour chaque note musicale (nous utilisons les fréquences basées sur la guitare [9]).

Note Musicale	f_{PWM} (Hz)	T_{PWM} (s)	ARR	CCR4
Mi (grave) - E	82.40	0.01213592	243	121
Lá - A	110.00	0.00909091	182	91
Ré - D	146.85	0.00680967	136	68
Sol - G	196.00	0.00510204	102	51
Si - B	246.95	0.00404940	81	40
Mi (aiguë) - E	329.65	0.00303352	61	30

TABLE 4 – Les valeurs de ARR et de CCR4 pour générer chaque fréquence de sortie PWM souhaitée.

Ainsi, le code utilisé pour produire la fréquence de sortie PWM est disponible dans l'annexe E de ce document.

4.5 Simulation du code - Génération de signaux de sortie PWM

Nous avons programmé le microcontrôleur avec les paramètres décrits ci-dessus. Ainsi, pour vérifier que le microcontrôleur émet correctement la fréquence de sortie, un oscilloscope a été utilisé pour mesurer cette fréquence et nous pouvons ainsi visualiser l'onde générée.

Voici les résultats obtenus par les fréquences de Mi aiguë et de Lá.

- Fréquence de Mi aiguë :

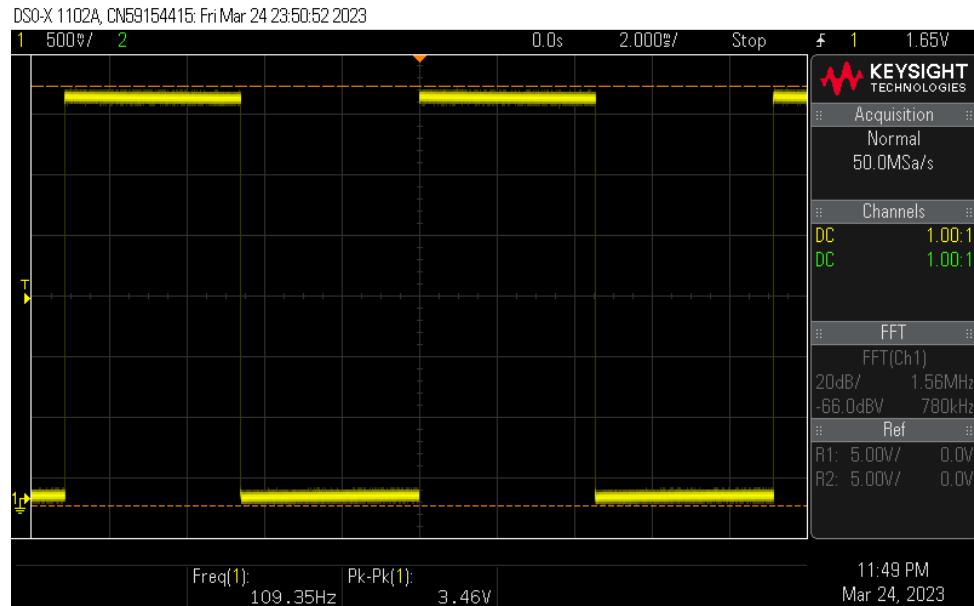


FIGURE 40 – Fréquence de sortie Mi aiguë produite par le microcontrôleur.

- Fréquence de Lá :

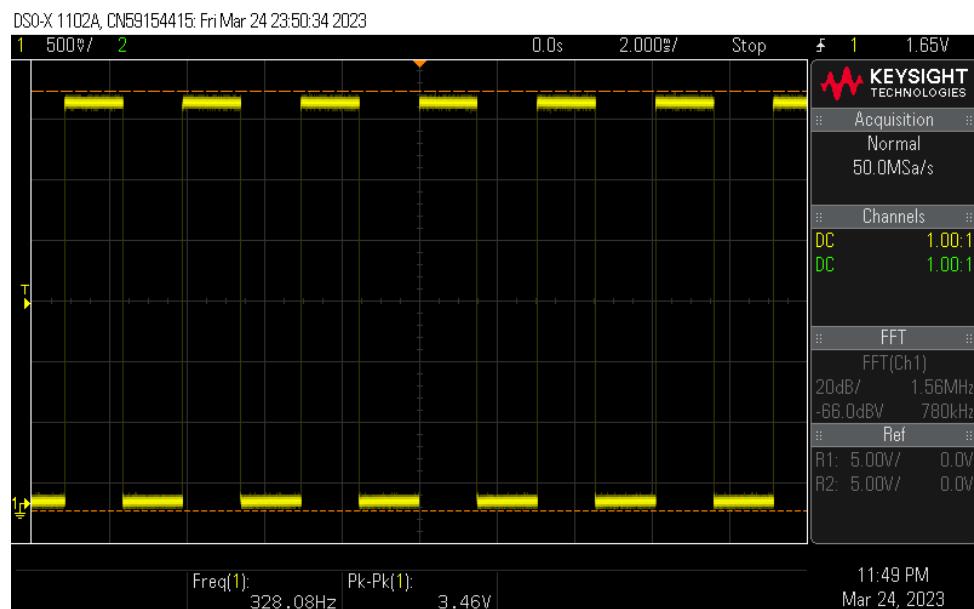


FIGURE 41 – Fréquence de sortie Lá produite par le microcontrôleur.

Ainsi, les figures 40 et 41 illustrent respectivement les fréquences de sortie PWM de 109Hz et 328Hz. La programmation du Timer1 est donc correcte, parce qu'elle présente des valeurs très proches de celles souhaitées, puisque les valeurs des registres doivent être des entiers.

À la fin, il reste à joindre chaque code développé à la fonction principale. Tous les codes sont placés à l'intérieur du loop afin que différentes notes de musique puissent être jouées (voir l'annexe F).

5 Assemblage et Résultats

Après avoir développé tous les codes nécessaires à la conception de l'Air Guitar, nous pouvons assembler le prototype. La figure 42 montre le prototype final.

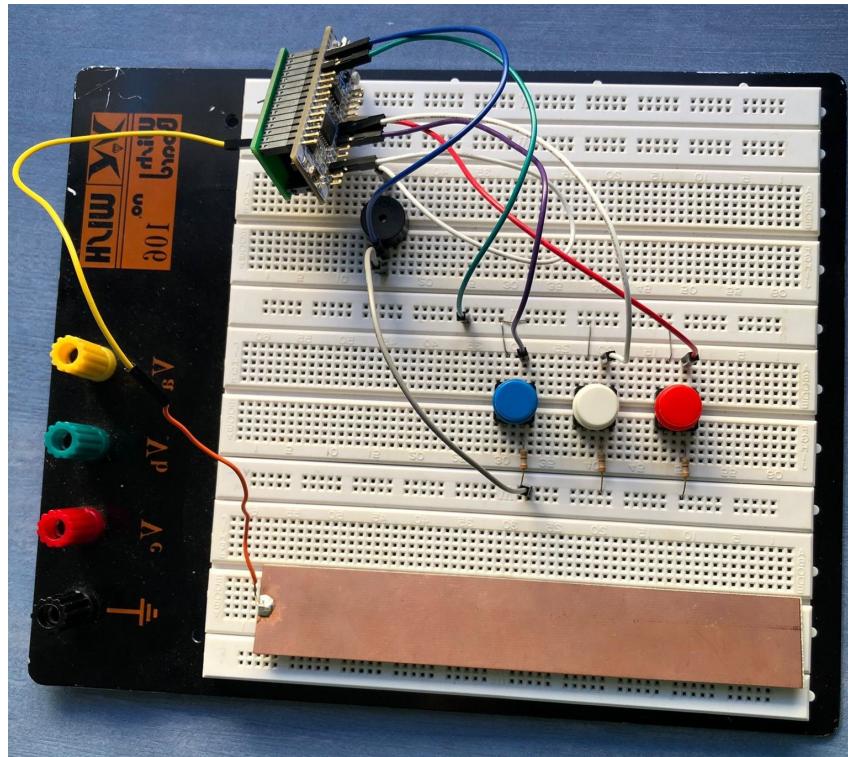


FIGURE 42 – Assemblage du prototype - Air Guitar.

Les composants utilisés sont les suivants :

- 1 Protoboard;
- 1 Carte Microcontrôleur STM32F301K8T6;
- 3 Résistances de $390\ \Omega$;
- 3 Boutons ;
- 1 Buzzer ;
- 1 Plaque Métallique ; et
- 15 fils de connexion ;

Nous avons ensuite effectué des tests pour vérifier le fonctionnement du prototype. Pour cela, nous avons testé le fonctionnement des boutons et du détecteur de présence pour la production du son.

A partir de la communication UART, nous envoyons un message informant de la fréquence lue par le microcontrôleur et de la note musicale jouée. Nous avons donc obtenu les résultats suivants :

- Note musicale Mi aiguë :

The screenshot shows the Arduino IDE interface with the sketch `sketch_mar29a.ino`. The serial monitor window displays a series of messages indicating the generation of the Mi aiguë note. Each message consists of a frequency value followed by the text "Frequency = <value> Hz - Note Musical = Mi-a". The frequencies listed are 40093, 40119, 40043, 40050, 40175, 40050, 40156, 40006, 40194, 40056, 40093, 40106, 40093, 40000, 40213, 40050, 40391, and 40100 Hz.

```
frequency = 40093 Hz - Note Musical = Mi-a
Frequency = 40119 Hz - Note Musical = Mi-a
Frequency = 40043 Hz - Note Musical = Mi-a
Frequency = 40050 Hz - Note Musical = Mi-a
Frequency = 40175 Hz - Note Musical = Mi-a
Frequency = 40050 Hz - Note Musical = Mi-a
Frequency = 40156 Hz - Note Musical = Mi-a
Frequency = 40006 Hz - Note Musical = Mi-a
Frequency = 40194 Hz - Note Musical = Mi-a
Frequency = 40056 Hz - Note Musical = Mi-a
Frequency = 40093 Hz - Note Musical = Mi-a
Frequency = 40106 Hz - Note Musical = Mi-a
Frequency = 40093 Hz - Note Musical = Mi-a
Frequency = 40000 Hz - Note Musical = Mi-a
Frequency = 40213 Hz - Note Musical = Mi-a
Frequency = 40050 Hz - Note Musical = Mi-a
Frequency = 40391 Hz - Note Musical = Mi-a
Frequency = 40100 Hz - Note Musical = Mi-a
```

FIGURE 43 – Test de la production de la note musicale Mi aiguë.

- Note musicale Lá :

The screenshot shows the Arduino IDE interface with the sketch `sketch_mar29a.ino`. The serial monitor window displays a series of messages indicating the generation of the Lá note. Each message consists of a frequency value followed by the text "Frequency = <value> Hz - Note Musical = La". The frequencies listed are 39987, 40031, 39912, 40025, 39881, 40119, 39875, 40257, and 40201 Hz.

```
Frequency = 39987 Hz - Note Musical = La
Frequency = 40031 Hz - Note Musical = La
Frequency = 39912 Hz - Note Musical = La
Frequency = 40025 Hz - Note Musical = La
Frequency = 39881 Hz - Note Musical = La
Frequency = 40119 Hz - Note Musical = La
Frequency = 39875 Hz - Note Musical = La
Frequency = 40257 Hz - Note Musical = La
Frequency = 40201 Hz - Note Musical = La
```

FIGURE 44 – Test de la production de la note musicale Lá.

Ainsi, nous avons observé que lorsque nous appuyons sur les boutons 1 et 3, la valeur 61 est attribuée à la variable `Freq_Timer1` et la variable `note musical` reçoit M aiguë ; et lorsque nous appuyons sur le bouton 2, la valeur 182 est attribuée à la variable `Freq_Timer1` et la variable `note musical` reçoit Lá, comme nous pouvons le voir sur l'UART. Cette variable modifie donc les paramètres du Timer1, qui modifie à son tour la fréquence de sortie du PWM.

Conclusion

Dans ce rapport, il a été décrit comment construire un capteur capacitif pour détecter la présence de la main afin de simuler les cordes d'une guitare et ainsi créer l'Air Guitar. Pour ce faire, les études suivantes ont été réalisées :

- Dans la section 1, l'étude et la simulation du comportement de la plaque métallique en tant que capteur ont été réalisées dans le programme Comsol. Nous avons ainsi appris à manipuler la plaque métallique pour différentes valeurs de charge, mais aussi le moyen dans lequel la plaque est insérée et d'autres phénomènes ;
- Dans la section 2, nous avons étudié le circuit électronique afin de pouvoir capturer la variation de capacitance en fonction de la présence de la main. Ensuite, la fabrication de ce circuit a été réalisée, en ajoutant les composants sur le circuit imprimé. Ainsi, nous avons appris à utiliser des amplificateurs opérationnels pour générer une fréquence en mode PWM ;
- Dans la section 3, nous programmons le microcontrôleur pour capturer la fréquence PWM et ainsi traiter le signal obtenu pour classer deux cas : existence d'un corps très proche (inférieur à 10cm) du capteur ou non ;
- Dans la section 4, nous programmons le microcontrôleur pour qu'il émette un son en fonction des boutons pressés. Nous avons donc développé un code qui identifie les boutons pressés et qui, à partir de la combinaison des boutons pressés, permet au microcontrôleur de générer une fréquence PWM qui, à son tour, passe par un buzzer afin d'émettre une note musicale ;
- Et finalement, dans la section 5, nous procédons à l'assemblage du prototype pour valider son fonctionnement.

Ainsi, nous avons pu développer un prototype capable de simuler une vraie guitare avec peu de ressources. Nous espérons donc que ce prototype pourra résoudre le problème du manque de guitares dans les écoles publiques pour enseigner l'art et la musique aux élèves. Par ailleurs, nous souhaitons améliorer notre prototype afin qu'il se rapproche de plus en plus de la version réelle. Les principaux points sont les suivants :

- Mettre en place une communication entre le microcontrôleur et le téléphone portable de l'élève au moyen d'une application afin que l'élève puisse modifier les effets et d'autres aspects liés à la reproduction du son ; et
- Créer un moule à l'aide de l'imprimante 3D afin de transformer le prototype en un objet prêt à l'emploi.

Références bibliographiques

- [1] S. BLAYAC, *Conception d'un conditionneur d'acquisition capacitive*. 2023.
- [2] C. MULTIPHYSICS®, *Logiciel de simulation COMSOL Multiphysics®*. adresse : <<https://www.comsol.fr/comsol-multiphysics>>, (consulté le : 22/03/2023).
- [3] L. HIROSE ELECTRIC CO., *0.5mm and 1mm Pitch, 2mm Height, FPC/FFC ZIF Connectors*. adresse : <<https://www.farnell.com/datasheets/2239115.pdf>>, (consulté le : 25/03/2023).
- [4] STMICROELECTRONICS, *STM32 Nucleo-32 development board with STM32F301K8 MCU, supports Arduino nano connectivity*. adresse : <<https://www.st.com/en/evaluation-tools/nucleo-f301k8.html#documentation>>, (consulté le : 10/03/2023).
- [5] A. L. FERREIRA, *Interrupção por timer no Arduino - gerando sinal de saída PWM*. adresse : <<https://www.squids.com.br/arduino/index.php/projetos-arduino/projetos-squids/basico/264-projeto-80-interrupcao-por-timer-no-arduino-disparo-de-alarme-com-sensor-de-toque>>, (consulté le : 16/03/2023).
- [6] R. SUPERMUNDO, *Afinar violão : aprenda e entenda os conceitos básicos*. adresse : <<https://www.supermundo.com.br/musica/afinar-violao-aprenda-e-entenda-os-conceitos-basicos/>>, (consulté le : 24/03/2023).
- [7] S. G. LARA, *Entendendo o pull-up e pull-down no Arduino*. adresse : <<https://www.makerhero.com/blog/entendendo-o-pull-up-e-pull-down-no-arduino>>, (consulté le : 24/03/2023).
- [8] XUKYO, *Utilisation d'un Buzzer avec Arduino*. adresse : <<https://www.aranacorp.com/fr/utilisation-d-un-buzzer-avec-arduino>>, (consulté le : 22/03/2023).
- [9] K. K. SHIMA, *Onda + Amplitude + Frequênciac + Harmônicos = Timbre ? – Parte 2*. adresse : <<https://blog.santoangelo.com.br/onda-amplitude-frequencia-harmonicos-timbre-parte-2/>>, (consulté le : 24/03/2023).

A Code pour effectuer la lecture de la fréquence

```

library IEEE;
use IEEE.electrical_systems.all;
use IEEE.energy_systems.all;
use IEEE.math_real.all;

entity human_body_model is

generic (
    L : real; -- largeur [M]
    d : real; -- hauteur [M]

    -- Forces voltage = initial_voltage during DC solution if TRUE.
    -- Unforced if False.
    use_initial_voltage : boolean := False;
    -- Initial capacitor voltage [V]
    initial_voltage      : voltage := 0.0);

port (terminal p1, p2 : electrical);

end entity human_body_model;

architecture human_body_model_arch of human_body_model is
    quantity v across i through p1 to p2;
    quantity power_input      : power;
    quantity energy_stored     : energy;
    constant capacitance_value : capacitance:= (8.85e-12*L*L)/d;

begin

    if domain = quiescent_domain use
        if use_initial_voltage use
            v == initial_voltage;
        else
            i == 0.0;
        end use;
    else
        i == capacitance_value*v'dot;
    end use;

    --- For information only
    power_input == v*i;
    energy_stored == 0.5*capacitance_value*v**2;

end architecture human_body_model_arch;

```

B Code pour effectuer la lecture de la fréquence

```
void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim) {
    // Time capture when there is a Rising Edge
    sample[Ind]=(uint32_t) HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_1);
    Ind++;

    // Processing the data to calculate frequency
    if (Ind == 5){
        Ind = 0;
        DeltaT = sample[NbCycle] - sample[0] + Overflow*ARR;
        Overflow = 0;
        Frequency = NbCycle*F_APB1/((htim->Init.Prescaler+1)*DeltaT);

        // Formatting of the display spaces
        sprintf(Text, "Frequency = %d Hz - Note Musical = %s\n", Frequency, Note_Musical);

        // Display with a speed 30 times less for readability
        if(Show++ == 30){
            Show = 0;
            HAL_UART_Transmit(&huart2, Text, 60, 1000);
        }
    }
}

void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef* htim) {
    Overflow++;
}
```

C Code pour vérifier la présence du corps

```
void Check_Frequency(void) {
    /* Checks if the frequency is lower than 42kHz, that is, ...
     * if there is the presence of the hand. In addition, it checks ...
     * that Freq_Timer1 is non-zero to emit a sound */
    if (Frequency < 42000 && Sum_Current != 0 && Freq_Timer1 != 0) {

        /* Checks the previous state to see if the same musical note is ...
         * being requested */
        if (Sum_Current != Sum_Before) {
            Sum_Before = Sum_Current;

            /* PWM frequency change */
            MX_TIM1_Init();
        }
        else {
            /* Null frequency */
            Freq_Timer1 = 0;
            Sum_Before = 0;

            /* PWM frequency change */
            MX_TIM1_Init();
        }

        /* Waits a time (in milliseconds) to execute the next cycle */
        HAL_Delay(100);
    }
}
```

D Code pour effectuer la lecture des entrées

```
void Check_Pin(void) {
    /* Reading the pins and assigning values to variables */
    Input_Pin_1 = HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_1) == GPIO_PIN_SET ? 1 : 0;
    Input_Pin_2 = HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_6) == GPIO_PIN_SET ? 1 : 0;
    Input_Pin_3 = HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_7) == GPIO_PIN_SET ? 1 : 0;

    /* Calculates the combination present on the pins */
    Sum_Current = (Input_Pin_1 << 2) | (Input_Pin_2 << 1) | Input_Pin_3;
}
```

E Code pour effectuer le son en fonction des broches

```
void Config_PWM(void) {
    /* From the value added in combination, a frequency for ...
     * the musical note will be generated */
    switch (Sum_Current) {
        case 0:
            Freq_Timer1 = 0;      // Null
            strcpy(Note_Musical, "Null");
            break;
        case 1:
            Freq_Timer1 = 136;   // Ré
            strcpy(Note_Musical, "Re");
            break;
        case 2:
            Freq_Timer1 = 182;   // Lá
            strcpy(Note_Musical, "La");
            break;
        case 3:
            Freq_Timer1 = 81;    // Si
            strcpy(Note_Musical, "Si");
            break;
        case 4:
            Freq_Timer1 = 243;   // Mi-g
            strcpy(Note_Musical, "Mi-g");
            break;
        case 5:
            Freq_Timer1 = 61;    // Mi-a
            strcpy(Note_Musical, "Mi-a");
            break;
        case 6:
            Freq_Timer1 = 102;   // Sol
            strcpy(Note_Musical, "Sol");
            break;
        case 7:
            Freq_Timer1 = 0;      // Null
            strcpy(Note_Musical, "Null");
    }
}
```

F Code main

```
/* Infinite Loop */
while (1)
{
    /* Check the input values of pins PB 1, 6 and 7 */
    Check_Pin();

    /* Checks the combination of pins PB 1, 6 and 7 */
    Config_PWM();

    /* Checks the value of the variable Frequency_Local */
    Check_Frequency();
}
```