

République Algérienne Démocratique et populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mouloud Mammeri de Tizi Ouzou
Faculté de génie Electrique et Informatique
Département Informatique



Mémoire de fin d'étude

En vue d'obtention du diplôme Master

*Domaine : **Mathématique et Informatique***

*Filière : **Informatique***

*Spécialité : **Réseaux, Mobilité et Systèmes Embarqués***

Thème :

**« Conception et Réalisation d'un Oscilloscope Numérique Avec
le microcontrôleur STM32 »**

Soutenu le : 08/10/2020

Présenté Par :

Mlle MOKRANI Sabrina

Mlle OUMESSAOUD Damya

Proposé et dirigé par :

Mrs HEMDANI

Co-Encadreur :

Mrs DAOUI

Promotion 2019/2020

Remercîments :

Nous tenons à témoigner notre reconnaissance à Monsieur HEMDANI, pour toutes les précieuses connaissances et compétences qu'il nous a permis d'acquérir, il est non seulement un excellent instructeur mais également un modèle et une source d'inspiration.

Nous remercions chaleureusement Mrs DAOUI pour son soutien, ses orientations, le temps qu'il nous a consacré et nous lui exprimons notre reconnaissance pour son initiation à ce domaine passionnant qui est le domaine des systèmes embarqués.

Nous remercions aussi les membres du jury d'avoir accepté de juger notre travail.

Nous exprimons également notre profonde gratitude à tous les enseignants du département informatique spécialement les enseignants de la spécialité RMSE.

Nos remerciements vont enfin à nos deux familles ainsi qu'à toute personnes ayant contribué de près ou de loin à l'élaboration de notre modeste travail.

Dédicaces :

« Je dédie ce travail à mes très chers parents qui n'ont jamais cessé de me soutenir tout au long de mon parcours, qui ont toujours crus en moi, je ne les remercier jamais assez pour tous leurs sacrifices et tous leurs amours, à mes chers frères Mourad, Farid, Ahcene, Hocine et mohamed, à ma belle-sœur Radjae et à ma grand-mère Fathma.

A tous mes chers ami(e)s de la promo, à Kenza, Melissa, Nawal et Lylia. »

MOKRANI Sabrina

« Je dédie ce travail, fruit de 17 ans d'études, en premier lieu à mes parents qui m'ont toujours soutenu, qui m'ont fait confiance et qui ont toujours mis à ma disposition tous ce dont j'avais besoin dans le seul et unique but de réussir et de m'épanouir ainsi qu'à toute ma famille.

Tout comme je le dédie à mes chers et fidèles ami(e)s Samir, Nounou, Kenza et Melissa avec qui j'ai partagé les 5 plus belles années de ma vie, qui ont toujours été à mes côtés et qui ont toujours crus en moi. »

OUMESSAOUD Damya

SOMMAIRE

Liste de figures :	4
Liste de Tableaux.....	4
Introduction générale	9
Chapitre 1: Le traitement du signal	10
1. Introduction	10
2. Le signal.....	11
2.1. Définition	11
2.2. Classification	11
3. Le traitement du signal	13
3.1. Définition	13
3.2. Etapes	14
4. La conversion analogique-numérique	15
4.1. Définition	15
4.2. L'échantillonnage.....	15
4.3. La quantification	16
4.4. Le codage.....	18
4.5. Les erreurs de conversion.....	18
4.6. Les convertisseurs analogique-numérique.....	19
5. La conversion numérique-analogique	23
5.1. Définition	23
5.2. Les erreurs de conversion.....	23
5.3. Les convertisseurs numérique-analogique.....	24
6. Exemple d'utilisation.....	25
6.1. Définition de l'oscilloscope.....	25
6.2. Le principe de fonctionnement	25
6.3. Les types	26
6.4. Les caractéristiques	29
7. Conclusion.....	30
Chapitre 2: Les systèmes embarqués et le microcontrôleur STM32.....	31
1. Introduction	31

2.	Les systèmes embarqués	32
2.1.	Définition	32
2.2.	Caractéristiques	32
2.3.	Architecture générale	32
2.4.	Classification	33
2.5.	Types d'application.....	34
3.	Les microcontrôleurs	35
5.1.	Définition	35
5.2.	Avantages	36
5.3.	Les mémoires.....	36
5.4.	Le jeu d'instructions CISC et RISC	37
5.5.	Choix d'un microcontrôleur.....	38
5.6.	Processeur ARM.....	38
4.	La STM32.....	40
4.1.	Présentation	40
4.2.	Fonctionnalités	43
5.	Conclusion.....	50
Chapitre 3: Conception.....		51
1.	Introduction	51
2.	présentation du projet.....	51
3.	Problématique	51
4.	Solutions.....	52
5.	Application embarquée	52
5.1.	L'acquisition.....	52
5.2.	Le traitement	55
5.3.	La transmission	56
5.4.	Architecture de l'application embarquée	57
5.5.	Organigramme de l'application embarqué	58
6.	L'application Desktop.....	59
6.1.	Fonctionnement	59
6.2.	Architecture	60
6.3.	Organigramme.....	61

7. Architecture générale du système.....	62
8. Conclusion.....	62
Chapitre 4: Réalisation.....	63
1. Introduction	63
2. Outils et logiciels.....	63
2.1. Application embarquée	63
2.2. Application desktop.....	68
3. Schéma électronique	69
4. Maquette	70
5. Les captures D'ECRAN L'APPLICATION desktop	70
6. Conclusion.....	73
Conclusion Générale	74
Références	75

LISTE DE FIGURES :

Figure 1: Le système général de communication. [1].....	10
Figure 2 : Classification morphologique des signaux. [2]	12
Figure 3 : Classification spectrale des signaux.[3]	12
Figure 4 : Signal transitoire. [4].....	13
Figure 5 : Signal périodique. [4]	13
Figure 6 : Signal aléatoire. [4].....	13
Figure 7 : Représentation temporelle. [5]	13
Figure 8 : Représentation spectrale. [5].....	13
Figure 9 : Synthèse d'un signal. [6]	14
Figure 10 : Modulation d'un signal. [6]	14
Figure 11 : Le filtrage du bruit. [6]	14
Figure 12 : Le codage. [6]	15
Figure 13 : Les étapes de numérisation d'un signal. [7]	15
Figure 14 : L'échantillonnage. [7].....	16
Figure 15 : Erreur de quantification. [7]	17
Figure 16 : Correction de l'erreur de quantification. [7]	17
Figure 17 : Erreur de quantification symétrique. [7]	17
Figure 18 : L'erreur d'offset et de gain. [7]	18
Figure 19 : La non-linéarité. [7].....	18
Figure 20 : Erreur de code manquant et de monotonie. [7]	19
Figure 21 : Schéma de principe d'un CAN n-bits à simple rampe. [8]	19
Figure 22 : Le signal rampe. [8]	20
Figure 23 : La sortie du comparateur. [8]	20
Figure 24 : La sortie de la porte logique. [8]	20
Figure 25 : CAN flash à 3 bits. [8]	21
Figure 26 : Les tensions de seuil. [8]	21
Figure 27 : Le circuit de codage. [8]	22
Figure 28 : La fonction de transfert d'un CAN flash à 3 bits. [8]	22
Figure 29 : Convertisseur à approximations successives. [7]	23
Figure 30 : CNA à réseau de résistances pondérées. [9]	24
Figure 31 : CAN à réseau de résistances R-2R. [9]	24
Figure 32: Oscilloscope Analogique.	25
Figure 33: Oscilloscope Numérique.	25
Figure 34: Oscillogramme.	25
Figure 35: Structure d'un oscilloscope analogique. [10]	26
Figure 36: FONCTIONNEMENT DE L'OSCILLOSCOPE ANALOGIQUE. [11]	27
Figure 37: FONCTIONNEMENT de l'oscilloscope numérique. [11]	28
Figure 38: la bande Passante. [11].....	29
Figure 39: structure générale d'un système embarqué. [12]	33
Figure 40: Classification des systèmes. [13]	34
Figure 41: les éléments interne d'un microcontrôleur. [14]	35

Figure 42:types de mémoires. [14]	36
Figure 43:Modèles de microcontrôleurs.	38
Figure 44:Architecture ARM. [15]	39
Figure 45:ARM Cortex-M. [16]	40
Figure 46:ARM Cortex-M4. [16]	41
Figure 47:Les composants de la carte STM32F4.[17]	42
Figure 48:Communication de données dans l'UART. [18]	44
Figure 49:Structure d'un timer.	45
Figure 50:LES CANAUX D'ENTRES ANALOGIQUES DU MICROCONTROLEUR STM32F446RE. [19]	46
Figure 51:bloc de diagramme de l'ADC. [18]	48
Figure 52:Bloc diagramme du DMA. [18]	50
Figure 53:Etapes suivies par le système.	51
Figure 54:Diodes. [20]	52
Figure 55:Passage du signal par une diode. [21]	53
Figure 56 : Le pont de diodes. Figure 57:Redressement du signal. 53	
Figure 58:La résistance. [22]	54
Figure 59:Le circuit atténuateur de tensions.....	55
Figure 60:Architecture de l'application embarquée.....	57
Figure 61:Organigramme de l'application embarqué.....	58
Figure 62:Organigramme de la configuration des paramètres de visualisation.	59
Figure 63:Architecture de l'application desktop.	60
Figure 64:Organigramme de l'application desktop.	61
Figure 65:Architecture générale du système.....	62
Figure 66:L'INTERFACE PRINCIPALE DE STM32CUBEMX.	64
Figure 67:Onglet MCU Selector de la fenêtre New Project.....	64
Figure 68:Onglet Board Selector de la Fenêtre New Project.....	65
Figure 69:Fenêtre Principale.....	65
Figure 70: Configuration de l'ADC.....	66
Figure 71: configuration UART.....	67
Figure 72:Configuration Timer.....	67
Figure 73:circuit du projet.	69
Figure 74:Maquette du projet.	70
Figure 75:page d'accueil de l'interface.	71
Figure 76: Channel 1.	71
Figure 77 : TIME.	72
Figure 78: Informations.	72

LISTE DE TABLEAUX

Tableau1 : Architecture Cisc et Risc.....	7
Tableau 2 : Modes de configuration d'un GPIO.....	43

LISTE DES ABREVIATIONS :

ADC : Analog to Digital Converter.

DAC : Digital to Analog Converter.

CD : Compact Disk.

LSB : Least Significant Bit.

MSB : Most Significant Bit.

CRT : Cathode Ray Tube.

SAR : successive-approximation register.

CAN : Convertisseur Analogique Numérique.

CNA : Convertisseur Numérique Analogique.

INL : Integral Non Linearity.

DNL : Differential Non Linearity.

LCD : Liquid Crystal Display.

AC : Alternative Current.

DC : Direct Current.

ASIC : Application Specific Integrated Circuit.

RTOS : Real Time Operating System.

PDA : Personal Digital Assistant.

CISC : Complex Instruction Set Computer.

RISC : Reduced Instruction Set Computer.

ARM : Advanced RISC Machines.

FPU : Floating Point Unit.

DSP : Digital Signal Processor.

NVIC : Nested Vectored Interrupt Controller.

WIC : Wakeup Interrupt Controller.

MPU : Memory Protection Unit.

AHP : Advanced High Performance.

APB : Advanced Peripheral Bus.

AMBA : Advanced Microcontroller Bus Architecture.

GPIO : General Purpose Input Output.

E/S : Entrées Sorties.

UART : Universal Asynchronous Receiver Transmitter.

USART : Universal Synchronous Asynchronous Receiver Transmitter.

I2C : Inter Integrated Circuits.

SPI : Serial Peripheral Interface.

RTC : Real Time Clock.

RTS : Ready To Send.

CTS : Ready To Clear.

PWM : Pulse Width Modulated

DMA : Direct Memory Access.

FIFO : First In First Out.

VBAT : Voltage Battery.

INTRODUCTION GENERALE

Afin de mieux comprendre et apprivoiser son environnement, qui ne s'exprime que par des signaux (Ondes, Vibrations, Electricités, ...), l'homme a dû mettre en œuvre des moyens lui permettant de l'étudier.

Dans le domaine de l'électricité par exemple, il existe des dispositifs nous permettant de réaliser des mesures ponctuelles des signaux électriques, qui se traduisent par l'affichage d'une valeur numérique sur un écran ou par la déviation d'une aiguille sur un cadran (un voltmètre, un ampèremètre, un wattmètre ou un multimètre). Sauf que ces appareils ne nous permettent pas de suivre la variation des signaux en question dans le temps et c'est dans cette optique là que l'oscilloscope s'avère utile.

Un oscilloscope est un instrument qui permet de représenter sur un écran l'évolution de différentes grandeurs physiques au fil du temps. Il permet d'observer des phénomènes très rapide (mesuré en nanoseconde) et il présente également un avantage qui est de visualiser la forme des signaux. C'est pour toutes ces raisons qu'il occupe une place très importante parmi les appareils de mesures. Mais malheureusement il n'est pas mis à disposition de toute personne ayant le besoin de l'utiliser puisqu'il est non seulement couteux mais également volumineux donc pas facile à transporter.

Avec l'avènement des microcontrôleurs, il est désormais possible de créer ce genre de dispositifs et de les mettre à la portée de tout le monde. Et c'est dans ce sens que s'inscrit l'objectif de notre travail, qui est de réaliser un oscilloscope numérique avec le microcontrôleur STM32.

Notre projet s'articule sur les quatre chapitres présentés ci-dessous :

Chapitre 1 : Le traitement du signal : Dans ce chapitre nous parlerons du traitement du signal, de ses différentes étapes et des composants électroniques qui y sont dédiés. Nous présenterons également l'oscilloscope, ses types et leurs fonctionnements.

Chapitre 2 : Les systèmes embarqués et le microcontrôleur STM32 : Ce chapitre présente le système embarqué, ses caractéristiques, son architecture et les microcontrôleurs notamment le μ c STM32.

Chapitre 3 : La conception : Sera consacré à la présentation du projet, son architecture et chaque aspect qui le définit.

Chapitre 4 : La réalisation : Nous avons cité, au cours de ce chapitre, les différents outils et logiciels de développement utilisés et aussi présenté des captures du projet.

Chapitre 1: LE TRAITEMENT DU SIGNAL

1. INTRODUCTION

Pour être perçu, le monde physique s'exprime par des signaux analogiques, sous différentes formes (Son, chaleur, Image, etc...) ; et pour être compris et traités, de nombreux chercheurs et savants ont travaillé sur ceux-ci, donnant naissance à la discipline du traitement du signal.

En effet, c'est en travaillant sur la résolution de l'équation de la propagation de la chaleur que Joseph FOURIER (1768-1830) avait mis en œuvre une méthode, connue sous le nom de « L'analyse de Fourier », qui consiste en la décomposition d'une fonction mathématique difficile à décrire, en une somme infinie de fonctions cosinus et sinus. Vinrent par la suite d'autres scientifiques, tels que Harry NYQUIST et Claude SHANNON qui, en se basant sur l'analyse de Fourier, ont pu développer la science du traitement du signal, grâce au théorème de l'échantillonnage. Puis en 1948, avec l'apparition du transistor, grâce aux physiciens Shockley, Bardeen et Barratain, et la fameuse « Théorie mathématique de la communication » de Claude Shannon, où il dit que tout message peut se résumer à une suite de 0 et de 1, le traitement du signal suit le système général de communication [Figure 1], ainsi la conversion analogique numérique naquit.

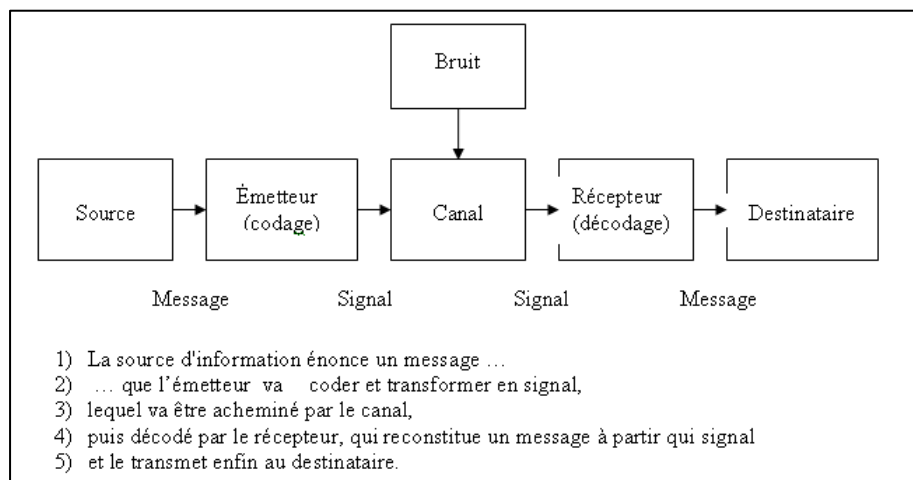


Figure 1: Le système général de communication. [1]

A présent, vu l'avancée technologique, notamment dans les circuits intégrés, le traitement du signal ainsi que sa conversion sont effectués par des systèmes de traitement de l'information efficaces et complexes, tels que les ADC et les DAC. Et c'est d'ailleurs les points traités par ce premier chapitre.

2. LE SIGNAL

2.1. Définition

Dans le langage courant, un signal est une information variable dans le temps et l'espace qui fait appel à un des 5 sens : l'odorat, l'ouïe, le goût, le toucher et la vue. On peut donc le trouver sous différentes formes : une image, un son, un texte, une odeur ... etc.

Quant au point de vu physique, un signal est une grandeur, qui dépend du temps, pouvant être représenté de façon mathématique.

On peut également le définir comme étant le support qui sert à véhiculer une information de sa source à sa destination.

2.2. Classification

Tout comme leurs formes diverses, les signaux ont différents critères de classification. Ils peuvent être classés selon le temps, la fréquence ou même l'énergie.

❖ Classification phénoménologique :

On s'intéresse à l'évolution du signal en fonction du temps. On distingue ainsi deux types :

a) *Les signaux déterministes* : Un signal est dit déterministe ou certain si, à n'importe quel moment sa valeur peut être déterminée avec certitude par un modèle mathématique.

Il en existe deux types :

- Signaux périodiques : Un signal $x(t)$ est dit périodique si, $\exists T \in \mathbb{R}, T > 0$, tel que : $x(t) = x(T + kT), \forall k \in \mathbb{Z}$ (Si T est le plus petit réel alors il est appelé Période). Il peut être composite (la répétition à l'infini d'un motif) comme le signal rectangulaire, sinusoïdal ou pseudo aléatoire (signal aléatoire qui se répète).
- Signaux non-périodiques : ce sont des signaux qui ne satisfont pas la relation précédemment citée. Ils se composent d'une part des signaux quasi- périodiques (formés d'une somme de signaux sinusoïdaux) et d'autre part des signaux transitoires (dont l'existence est limitée dans le temps).

b) *Les signaux aléatoires* : Un signal dit aléatoire, probabiliste ou stochastique est un signal dont le comportement temporel est imprévisible. On y trouve deux types :

- Les signaux stationnaires : Signaux dont les caractéristiques statiques sont invariantes dans le temps.
- Les signaux non-stationnaires : signaux pouvant être cyclo- stationnaires ou quelconques.

❖ Classification morphologique :

Dans cette classification, l'amplitude du signal est le paramètre pris en compte [Figure 2].

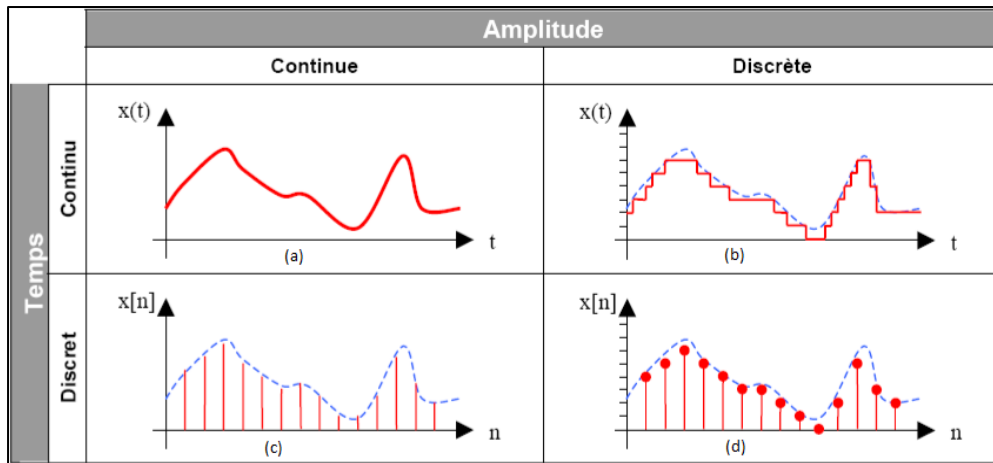


Figure 2 : Classification morphologique des signaux. [2]

- a) *Signaux analogiques* : Ce sont des signaux continus en temps et en amplitude.
- b) *Signaux échantillonnés* : Ce sont des signaux discrets en temps et continus en amplitude.
- c) *Signaux quantifiés* : Ce sont des signaux continus en temps et discrets en amplitude.
- d) *Signaux numériques* : Ce sont des signaux discrets en temps et en amplitude.

❖ Classification spectrale :

Le Spectre d'un signal est la représentation de son amplitude, de sa phase, de son énergie ou de sa puissance en fonction de sa fréquence (Hz).

La largeur de bande dite aussi largeur spectrale est le domaine des fréquences occupées par le spectre d'un signal. Elle est définie par :

$$\Delta F = F_{\max} - F_{\min}.$$

Selon ces critères, on distingue quatre types distincts [Figure 1-3] :

- a) *Les signaux à bande étroite* : avec $\Delta F / F_{\text{moy}}$ petit (soit $F_{\max} \approx F_{\min}$).
- b) *Les signaux à large bande* : avec $\Delta F / F_{\text{moy}}$ grand (soit $F_{\max} \gg F_{\min}$).
- c) *Les signaux à basses fréquences (BF)* : leur largeur de bande est concentrée sur des fréquences faibles.
- d) *Les signaux à hautes fréquences (HF)* : leur largeur de bande est concentrée sur des fréquences importantes.

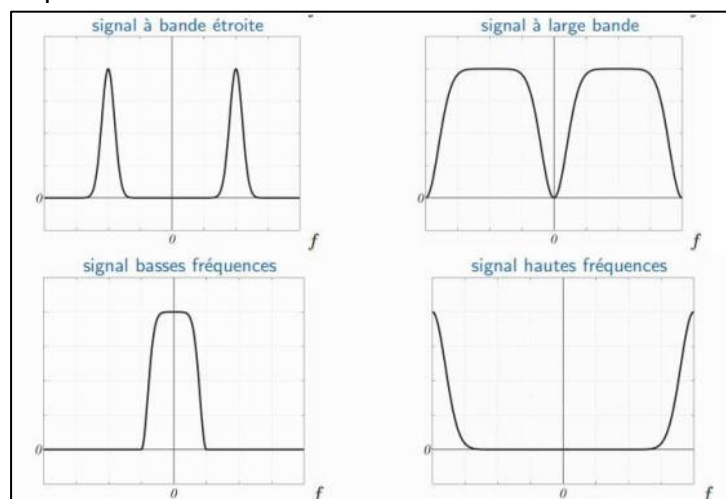


Figure 3 : Classification spectrale des signaux.[3]

❖ Classification énergétique :

En s'intéressant à l'énergie et à la puissance d'un signal, on trouve 2 catégories :

a) *Signaux à énergie finie* : ce sont des signaux à puissance nulle. On peut citer les signaux transitoires et les signaux éphémères [Figure 4].

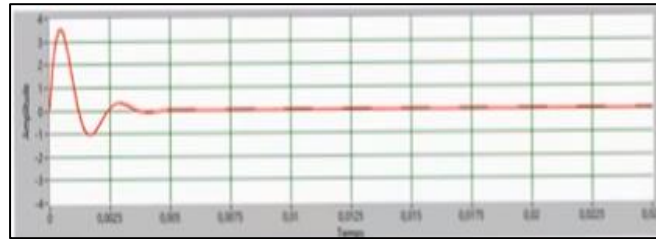


Figure 4 : Signal transitoire. [4]

b) *Signaux à énergie infinie* : ce sont des signaux à puissance finie. On peut citer les signaux périodiques [Figure 5] et les signaux aléatoires [Figure 6].

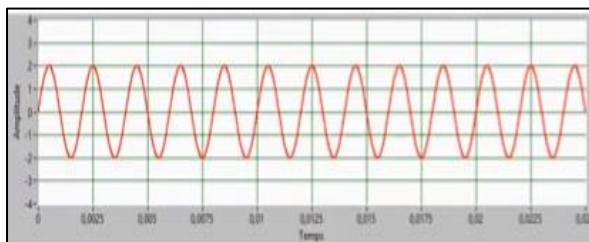


Figure 5 : Signal périodique. [4]

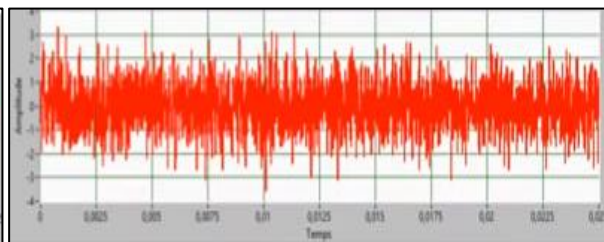


Figure 6 : Signal aléatoire. [4]

3. LE TRAITEMENT DU SIGNAL

3.1. Définition

a) *Le traitement du signal* : C'est une discipline technique qui, en se basant sur la théorie du signal, les techniques de l'électronique, de l'informatique et de la physique, vise à créer, analyser et filtrer le bruit des signaux quels qu'ils soient, pour en extraire un maximum d'informations.

b) *La théorie du signal* : C'est la science qui vise à représenter mathématiquement, dans les domaines temporels [Figure 7] et spectrale [Figure 8], un signal.

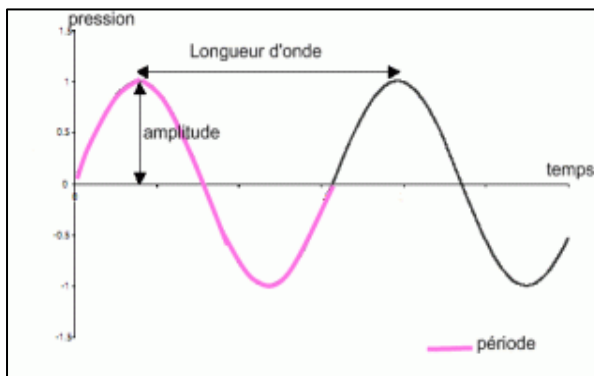


Figure 7 : Représentation temporelle. [5]

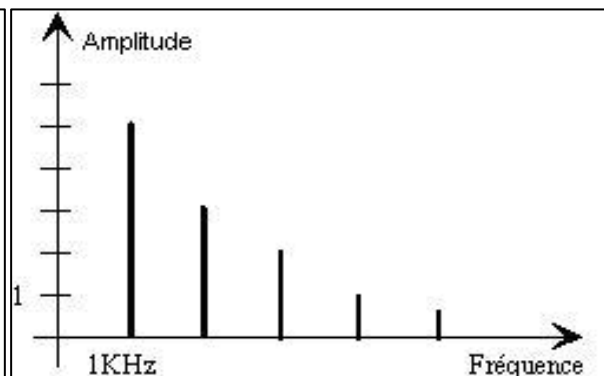


Figure 8 : Représentation spectrale. [5]

c) *Le bruit* : C'est une perturbation qui, en s'appliquant à un signal, peut gêner sa détection.

3.2. Etapes

Le traitement du signal passe par trois grandes étapes et chacune a son propre déroulement.

a) *La création* : cette étape consiste en l'élaboration du signal à transmettre et cela en passant par :

- i. La synthèse : création du signal en combinant des signaux élémentaires (signaux sinusoïdaux, rectangulaires ou en dent de scie). [Figure 9]

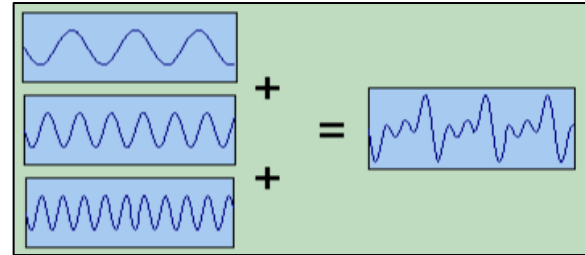


Figure 9 : Synthèse d'un signal. [6]

- ii. La modulation : changement de fréquence permettant l'adaptation du signal aux caractéristiques fréquentielles du support de transmission ou d'un filtre d'analyse. [Figure 10]

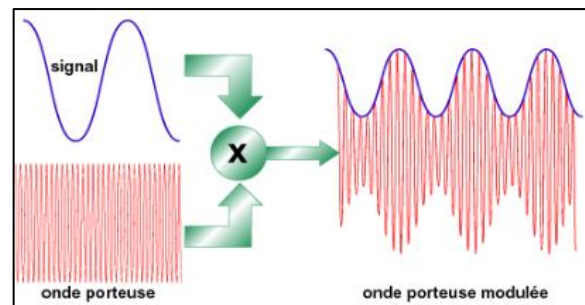


Figure 10 : Modulation d'un signal. [6]

b) *L'analyse* : à ce stade, on s'intéresse à la compréhension de la nature et des origines du signal grâce à :

- i. La détection : elle nous permet d'extraire le signal voulu d'un bruit de fond qui le perturbe.
- ii. L'identification : elle nous sert à le classer.

c) *La transformation* : elle nous sert à adapter un signal à nos besoins, en se servant du :

- i. Filtrage : pour éliminer certains effets indésirables, tels que le craquement ou l'écho (pour le son), le bruit (pour les images), ...etc. [Figure 11]

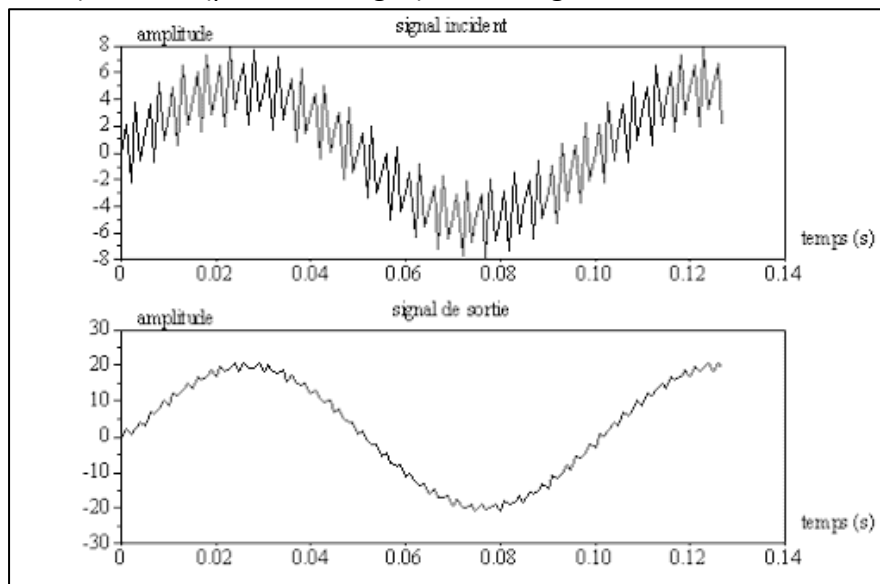


Figure 11 : Le filtrage du bruit. [6]

- ii. Codage : pour une conversion à un signal numérique qui permettra l'étude sur des appareils numériques tels qu'un ordinateur ou éviter le bruit de fond. [Figure 12]

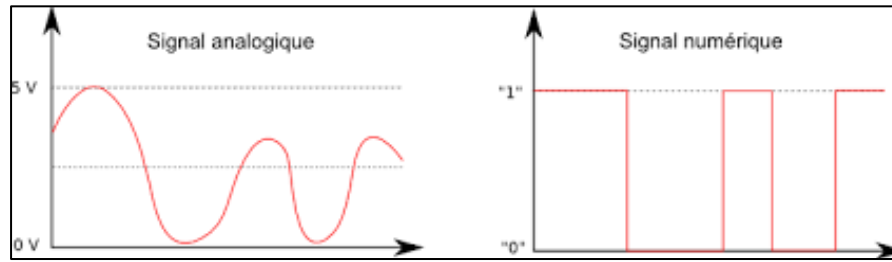


Figure 12 : Le codage. [6]

4. LA CONVERSION ANALOGIQUE-NUMERIQUE

4.1. Définition

La conversion analogique numérique, autrement dite la numérisation, est le passage d'un signal analogique (continu en temps et en amplitude) à un signal numérique (discret en temps et en amplitude) ; grâce à un dispositif électronique appelé ADC (Analog to Digital Converter).

Elle permet principalement le traitement du signal avec des appareils numériques tels que les ordinateurs, mais offre également d'autres avantages, dont :

- L'amélioration de la robustesse du signal face au bruit.
- La reproductibilité facile.
- La possibilité de traitements adaptatifs ...etc.

Cette technique passe par 3 étapes [Figure 13] : l'échantillonnage, la quantification et le codage.

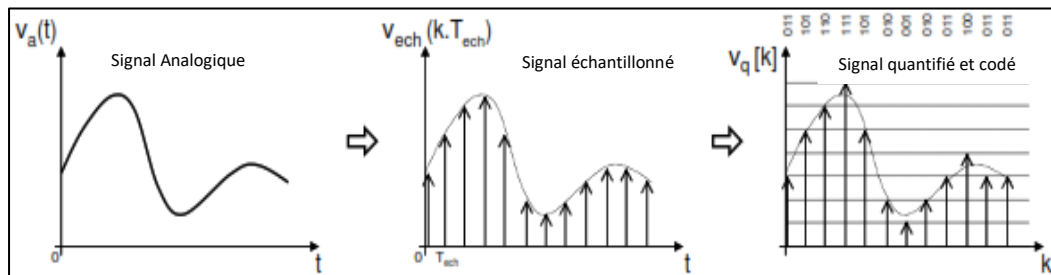


Figure 13 : Les étapes de numérisation d'un signal. [7]

4.2. L'échantillonnage

C'est l'étape qui nous permet d'extraire des morceaux du signal à des intervalles de temps constants, nommés périodes d'échantillonnage T_e .

Mathématiquement parlant [Figure 14], l'échantillonnage consiste en, la multiplication dans le domaine temporel et la convolution dans le domaine spectral/fréquentiel, de la fonction représentant le signal par la fonction représentant un peigne de Dirac.

Soient : $x(t)$ la fonction qui représente le signal à échantillonner, $\delta(t)$ la fonction du peigne de Dirac et $x_e(t)$ la fonction du signal échantillonné on aura alors :

$$x_e(k.T_e) = x(t) \cdot \sum_k \delta(t - k.T_e)$$

Mais il faut faire attention au choix de la période d'échantillonnage T_e , car plus elle est grande plus la fréquence d'échantillonnage $f_e = 1/T_e$ est petite ce qui provoque un repliement spectral (apparition de fréquences parasites). Et c'est dans le but d'éviter ces perturbations que « Le Théorème de Shannon » intervient.

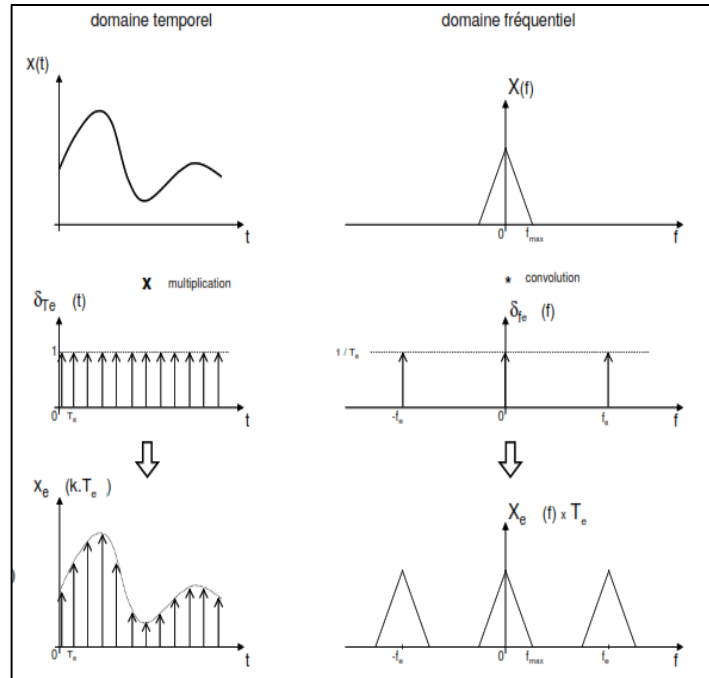


Figure 14 : L'échantillonnage. [7]

❖ Le théorème de Shannon :

Pour reconstruire un signal de sortie de manière fidèle au signal d'entrée, il faut choisir une fréquence d'échantillonnage deux fois supérieure à la fréquence maximale, contenue dans le signal d'entrée : $f_e \geq 2f_{max}$. [7]

A titre d'exemple, la fréquence d'échantillonnage des CD est de 44.1kHz, du fait que la plage de fréquence audio à laquelle l'ouïe humaine est sensible est située entre 20 Hz et 20kHz.

4.3. La quantification

Pour qu'un ordinateur, ou n'importe quel autre appareil numérique, puisse lire et traiter le signal échantillonné, il doit être représenté par des valeurs binaires et c'est là le rôle de la quantification.

Ainsi, chaque valeur échantillonnée se verra attribuer une valeur binaire sur n bits, ce qui nous donne un niveau de quantification $N=2^n$. On peut définir d'autres paramètres aux côtés du niveau de quantification, à savoir :

❖ **La pleine échelle** : notée V_{PE} , elle représente la plage de variation acceptable de la tension analogique. Autrement dit c'est la valeur maximale échantillonnée.

❖ **Le pas de quantification** : aussi appelé quantum, résolution ou LSB, il représente la dimension de division de la pleine échelle, selon le niveau de quantification. Il est donc défini comme suit :

$$q = V_{PE}/2^n$$

Si le quantum est constant alors on parle de quantification uniforme, si au contraire il est variable on dit que la quantification est non-uniforme et elle sert quand on veut apporter une meilleure précision à certains échantillons plus qu'à d'autres.

- ❖ **Le bruit de quantification** : c'est la différence entre la valeur du signal analogique en entrée et celle en sortie, due à l'arrondi aléatoire. Il est lié à la résolution d'un ADC (plus la résolution est élevée plus l'erreur est réduite). Il est généralement sous la forme décrite par la [Figure 15].

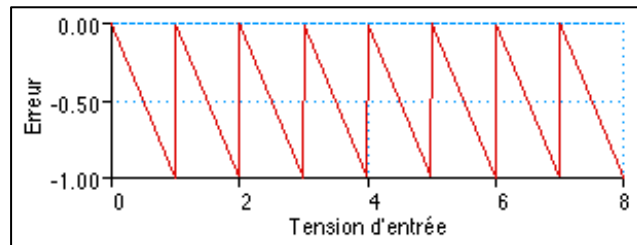


Figure 15 : Erreur de quantification. [7]

On remarque que l'erreur oscille entre 0 et -1 LSB. Sauf que pour une meilleure reconstitution il est préférable qu'elle soit centrée autour de 0, de sorte à quantifier ou par excès ou par défaut. Pour cela un décalage d' $1/2$ LSB est effectué lors de la première transition, comme il est indiqué sur la [Figure 16].

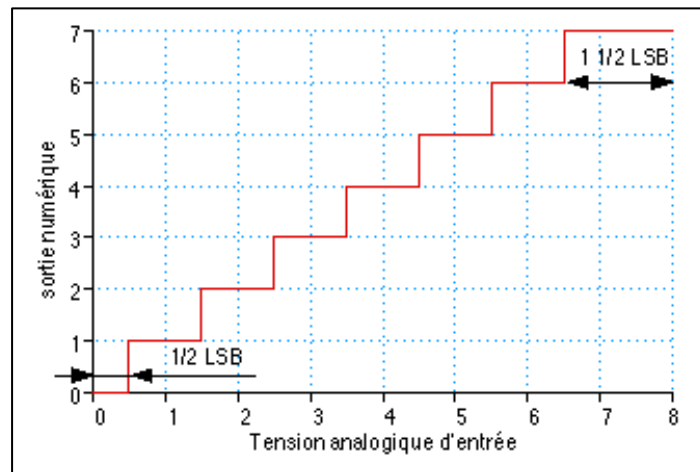


Figure 16 : Correction de l'erreur de quantification. [7]

Ainsi, l'erreur devient symétrique et vaut $\pm 1/2$ LSB. Comme le montre la [Figure 17].

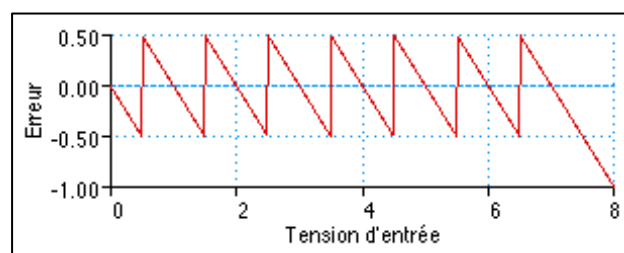


Figure 17 : Erreur de quantification symétrique. [7]

4.4. Le codage

C'est la dernière étape de la numérisation. Elle consiste en la conversion en code binaire, des valeurs quantifiées du signal échantillonné. Le nombre de bits pour le codage est spécifique au convertisseur.

4.5. Les erreurs de conversion

Comme il a été expliqué précédemment, la conversion analogique-numérique nécessite la sélection de certaines valeurs du signal en entrée à des intervalles de temps constants. Impliquant une perte de certaines informations ; celles-ci provoquent alors des erreurs, appelées également paramètres.

- ❖ **Erreur de gain** : C'est l'écart entre la valeur théorique et la valeur réelle mesurées sur la dernière transition. Elle est exprimée en LSB. [Figure 18 à gauche]
- ❖ **Erreur d'offset** : C'est le décalage entre la valeur réelle et la valeur théorique mesurées lors de la première transition. Elle est aussi exprimée en LSB. [Figure 18 à droite]

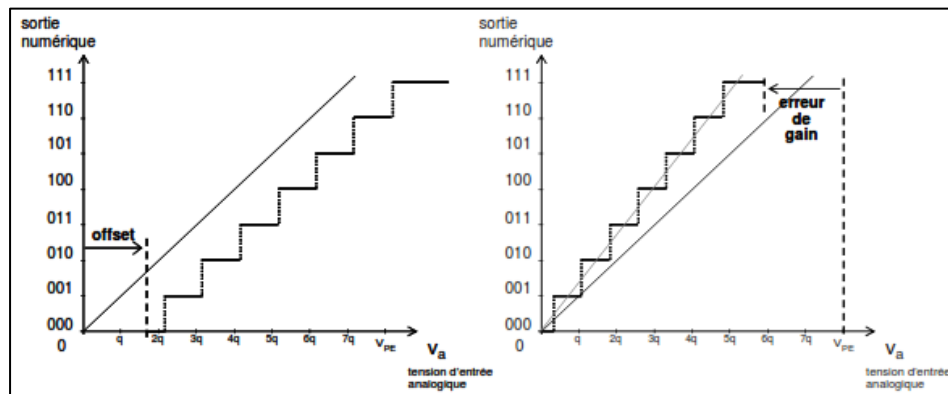


Figure 18 : L'erreur d'offset et de gain. [7]

- ❖ **Erreur de linéarité** : Elle est due à la variation de la résolution des convertisseurs, mais n'est mesurée que si les erreurs d'offset et de gain sont réglées [Figure 1-19]. Il en existe deux types :
 - La non-linéarité différentielle, notée DNL, est la différence entre l'écart mesuré et le LSB théorique qui vaut 1.
 - La non-linéarité intégrale, notée INL, représente l'écart entre les valeurs réelles et les valeurs théoriques. C'est également une représentation cumulative du DNL.

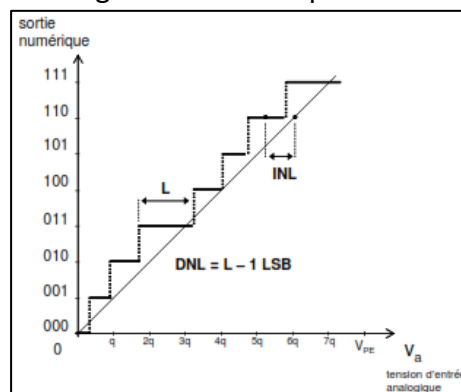


Figure 19 : La non-linéarité. [7]

- ❖ **Codes Manquants** : ce sont les codes de sortie auxquels aucune valeur d'entrée ne sera attribuée. [Figure 1-20]
- ❖ **Monotonicité** : L'erreur de monotonicité apparaît quand les codes de sortie n'évoluent pas de manière croissante pour un signal d'entrée croissant. [Figure 1-20]

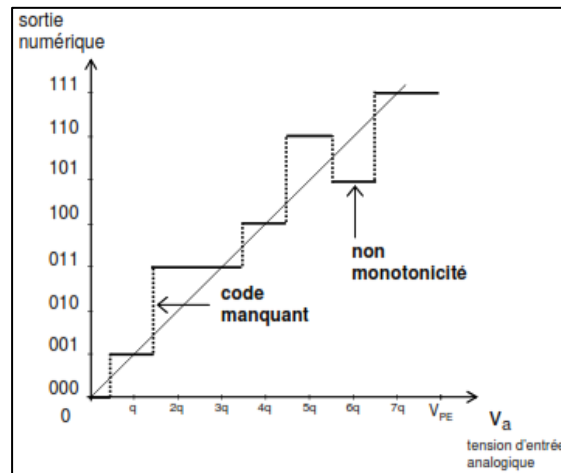


Figure 20 : Erreur de code manquant et de monotonicité. [7]

- ❖ **Temps de conversion** : C'est le temps écoulé entre l'ordre de conversion et la sortie des valeurs binaires.
- ❖ **Précision du convertisseur** : Elle est exprimée en cumulant toutes les erreurs précédentes, données en % de la pleine échelle ou en fraction du quantum.

4.6. Les convertisseurs analogique-numérique

Avec le développement inouï que vit l'électronique aujourd'hui, on trouve un vaste choix de convertisseur mais tous se basent sur 3 principaux types : les convertisseurs à simple rampe, les convertisseurs parallèles et les convertisseurs à approximations successives.

❖ Convertisseur simple rampe

Comme le montre la [Figure 21], un convertisseur à simple rampe est composé d'une entrée de tension U_e , un générateur de rampe, un comparateur, un circuit de logique de commande, une porte '&', un compteur binaire, une horloge et plusieurs sorties binaires.

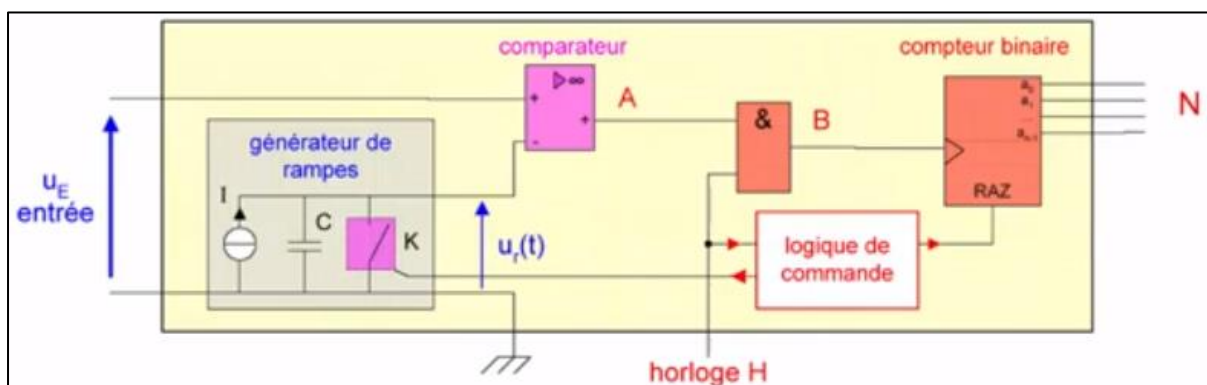


Figure 21 : Schéma de principe d'un CAN n-bits à simple rampe. [8]

Pour commencer, le circuit de la logique de commande délivre un signal rectangulaire $K(t)$ de fréquence f_0 et de rapport cyclique très faible (une impulsion) et passe en entrée du générateur de rampe, qui à son tour produit un signal rampe $U_r(t)$ de fréquence f_0 . [Figure 22]

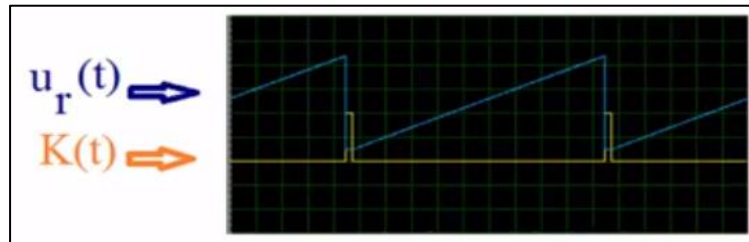


Figure 22 : Le signal rampe. [8]

Ce dernier passe par l'entrée inverseuse du comparateur et est comparé à la tension d'entrée $U_e(t)$ qui passe par l'entrée non-inverseuse. A ce stade si l'amplitude de $U_e(t)$ est supérieure à celle de $U_r(t)$ alors la sortie $A(t)$ sera à l'état haut sinon elle sera à l'état bas. Ainsi $A(t)$ sera proportionnel à $U_e(t)$. [Figure 23]

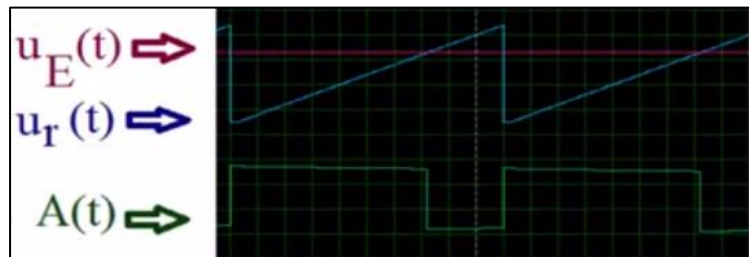


Figure 23 : La sortie du comparateur. [8]

Par la suite, $A(t)$ ainsi que le signal d'horloge $H(t)$ passent en entrée de la porte logique '&' pour avoir un signal $B(t)$ dont le nombre de période est proportionnel au temps ou $A(t)$ est à l'état haut et donc proportionnel à l'amplitude de la tension d'entrée $U_e(t)$. [Figure 24]

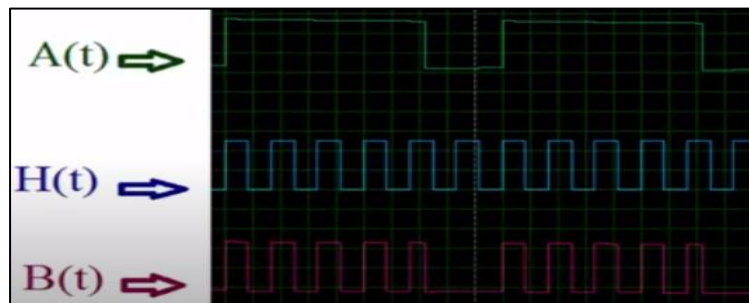


Figure 24 : La sortie de la porte logique. [8]

Pour finir, $B(t)$ passe par le compteur binaire, qui fournira à sa sortie la valeur, en binaire, N du nombre de périodes de $B(t)$ et c'est la représentation numérique de l'entrée analogique U_e . Quant à la remise à zéro, elle peut être assurée par le circuit de la logique de commande ou par le front montant d'un circuit monostable qui aura $A(t)$ comme entrée.

Ce type de convertisseurs offre un avantage de simplicité et de faible coût. Par contre, ses inconvénients sont :

- La lenteur, puisqu'une conversion nécessite 2^N cycles d'horloge.

-Une erreur d'environ 1.5 quantum, car il n'y a pas de synchronisation entre l'horloge et la remise à zéro.

C'est donc, pour palier à ces inconvénients, qu'on trouve :

- Des convertisseurs à double rampe, qui effectuent une double intégration du signal (passage par le circuit générateur de rampe), pour éliminer les erreurs dues aux composants.
- Convertisseurs à rampe numérique qui eux remplacent le circuit d'intégration analogique (le générateur de rampe) par un convertisseur numérique analogique.

❖ Convertisseur parallèle

Il est aussi appelé convertisseur à comparateurs en échelle, puisqu'il est composé de $2^N - 1$ comparateurs montés en échelle, par lesquels passent $2^N - 1$ tensions de seuil. Ces tensions de seuil sont obtenues par 2^N résistances connectées en série entre V_{ref} et la masse qu'on appelle « pont diviseur ».

Pour une quantification linéaire par défaut, toutes les résistances seront identiques. Mais pour une quantification linéaire centrée, la résistance liée à la masse sera égale à $R/2$, celle liée à V_{ref} égale à $3R/2$; quant aux autres, elles seront identiques et vaudront R .

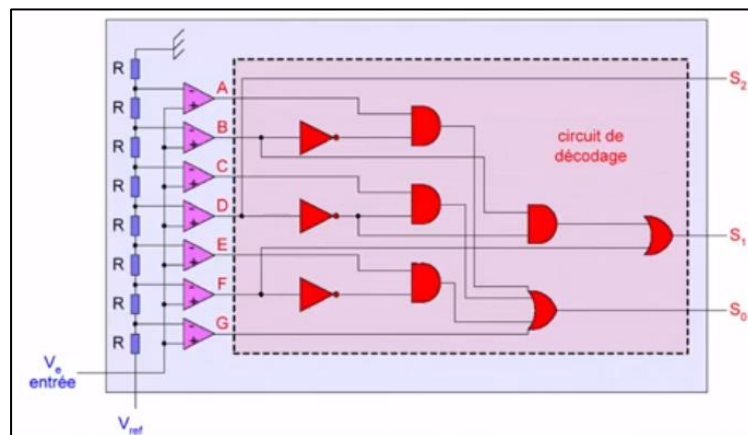


Figure 25 : CAN flash à 3 bits. [8]

En prenant exemple sur le CAN représenté par la [Figure 25], une tension d'entrée V_e est appliquée aux entrées non-inverseuses des comparateurs et une tension de seuil, obtenue par le passage de V_{ref} par le pont diviseur subissant une chute de tension à chaque borne de résistance égale au quantum $= V_{ref}/2^N = V_{ref}/8$, est appliquée aux entrées inverseuses des mêmes comparateurs. [Figure 26]

Ainsi, si la tension d'entrée est supérieure à la tension de seuil, à laquelle elle est comparée, la sortie du comparateur correspondant vaudra 1 sinon elle vaudra 0.

Par la suite, les sorties des comparateurs passeront par un circuit de codage [Figure 27] pour avoir les sorties binaires grâce aux expressions suivantes :

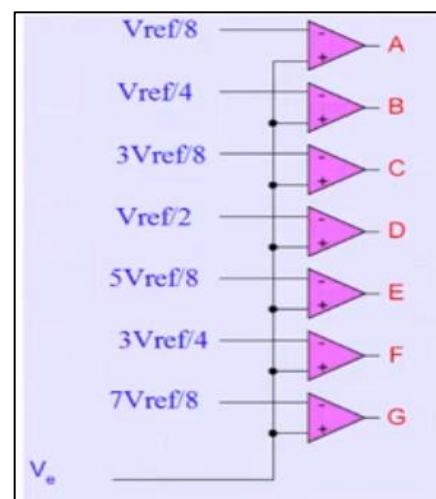


Figure 26 : Les tensions de seuil. [8]

$$\left\{ \begin{array}{l} S_2 = D \\ S_1 = B\bar{D} + F \\ S_0 = A\bar{B} + C\bar{D} + E\bar{F} + G \end{array} \right.$$

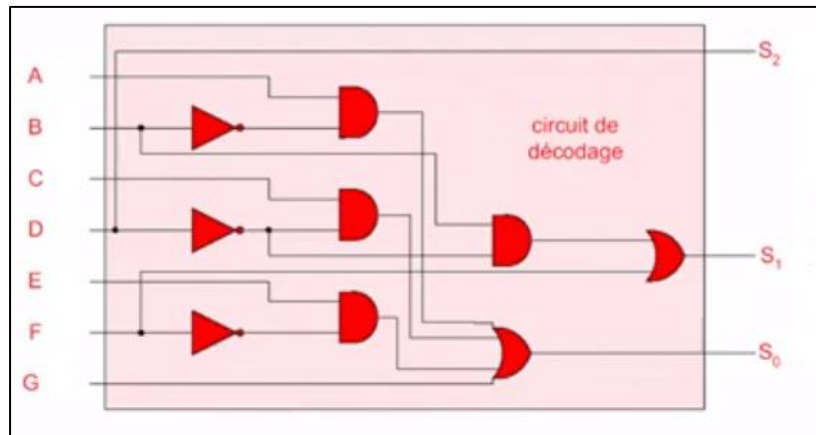


Figure 27 : Le circuit de codage. [8]

On aura alors la fonction de transfert représentée par la [Figure 28] :

V_e	S_2	S_1	S_0	N
$0 < V_e < V_{ref}/8$	0	0	0	0
$V_{ref}/8 < V_e < V_{ref}/4$	0	0	1	1
$V_{ref}/4 < V_e < 3V_{ref}/8$	0	1	0	2
$3V_{ref}/8 < V_e < V_{ref}/2$	0	1	1	3
$V_{ref}/2 < V_e < 5V_{ref}/8$	1	0	0	4
$5V_{ref}/8 < V_e < 3V_{ref}/4$	1	0	1	5
$3V_{ref}/4 < V_e < 7V_{ref}/8$	1	1	0	6
$7V_{ref}/8 < V_e < V_{ref}$	1	1	1	7

Figure 28 : La fonction de transfert d'un CAN flash à 3 bits. [8]

Ce type de convertisseurs est réputé pour sa rapidité puisque la conversion est effectuée en un seul coup d'horloge. Mais face à cela, se pose un problème de coût et de complexité qui croissent exponentiellement avec le nombre de bits. C'est donc pour cette raison que leur résolution est limitée à 12 bits.

❖ Convertisseur à approximations successives

Ce type de CAN [Figure 29] est constitué d'une boucle de rétroaction, composée d'un registre à approximations successives (SAR), d'un convertisseur numérique analogique (CNA) et d'un comparateur. Son principe se base sur la recherche dichotomique, puisqu'à chaque coup d'horloge l'intervalle de recherche est divisé par 2.

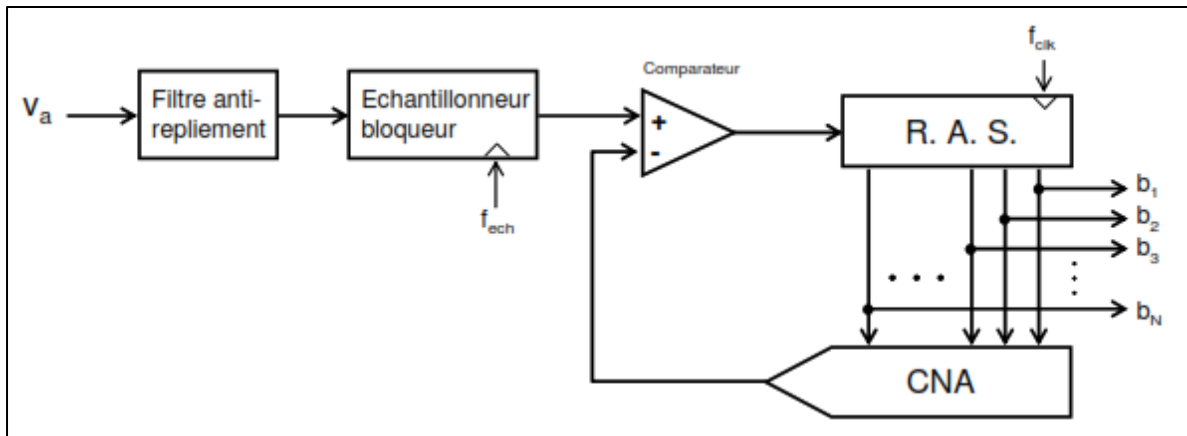


Figure 29 : Convertisseur à approximations successives. [7]

Au premier top d'horloge, un chiffre binaire, dont tous les bits sont à 0 sauf le MSB, est produit par le SAR et passe vers le CNA, pour avoir la tension équivalente. Ensuite la tension à convertir V_a est injectée, après passage par un filtre et un échantillonneur, à l'entrée non-inverseuse du comparateur et la tension résultant du CAN, qui est égale à $V_{ref}/2$, est injectée à l'entrée inverseuse. Si V_a est inférieure à $V_{ref}/2$, alors le MSB est mis à 0 sinon, il reste à 1. Et ainsi de suite, les autres bits, jusqu'au LSB, sont testés par le même principe.

Malgré l'ancienneté de leur approche, les convertisseurs SAR restent très répandus, grâce au temps de conversion fixe qui est indépendant de la taille de l'information à numériser. Et on le trouve par exemple dans les microcontrôleurs STM32.

5. LA CONVERSION NUMERIQUE-ANALOGIQUE

5.1. Définition

Les convertisseurs numérique-analogique (CNA) servent à convertir un signal numérique sur N bits (discret en temps et en amplitude) en un signal analogique (continu en temps et en amplitude).

Le principe de cette conversion se base sur la sommation sur N grandeurs multiples de 2, les unes par rapport aux autres.

$$b_{N-1}b_{N-2} \dots b_0 \rightarrow V_{out} = \frac{-V_{ref}}{2^N} \cdot \sum_{k=0}^{N-1} b_k \cdot 2^k$$

5.2. Les erreurs de conversion

Tout comme les CAN, les CNA provoquent eux aussi les erreurs de conversion, citées dans le point 4.5, à savoir : l'erreur de gain, l'erreur d'offset, l'erreur de linéarité et la monotonicité, auxquelles, on ajoute :

❖ **Le temps d'établissement** : Il représente le temps que met la tension de sortie à se stabiliser.

5.3. Les convertisseurs numérique-analogique

Il existe 2 types principaux de CNA :

❖ CNA à réseau de résistances pondérées

Comme le montre la [Figure 30], ce type de convertisseur se base sur n résistances, allant de $2R$ à $2^n R$ auxquelles on applique la tension de référence V_{ref} et qui sont contrôlées par des interrupteurs actionnés par les entrées binaires à convertir. La sortie de ce réseau de résistances passe par un amplificateur inverseur à circuit intégré qui nous donne à sa sortie la tension de sortie U_s .

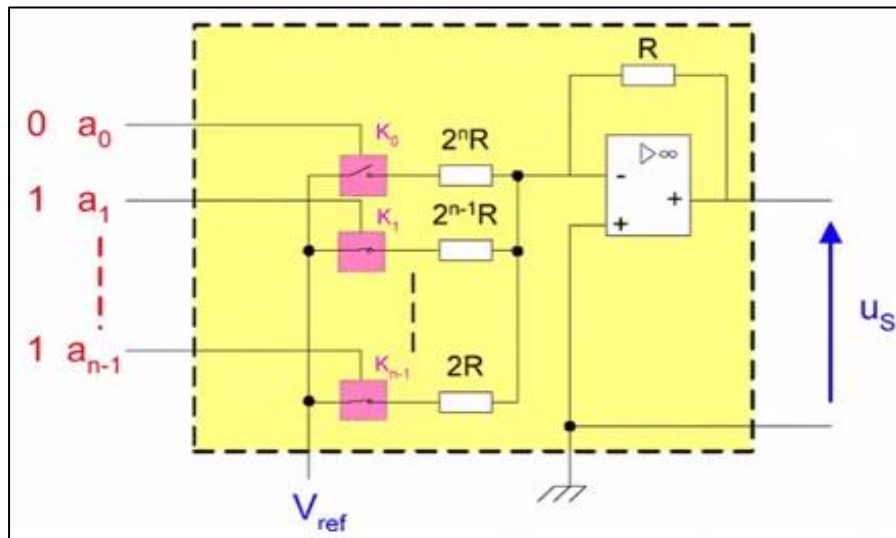


Figure 30 : CNA à réseau de résistances pondérées. [9]

❖ CNA à réseau de résistances R-2R

C'est un convertisseur qui se base sur le même principe que le précédent, mais qui est composé d'un réseau de résistances ne valant que R ou $2R$, comme le montre la [Figure 31].

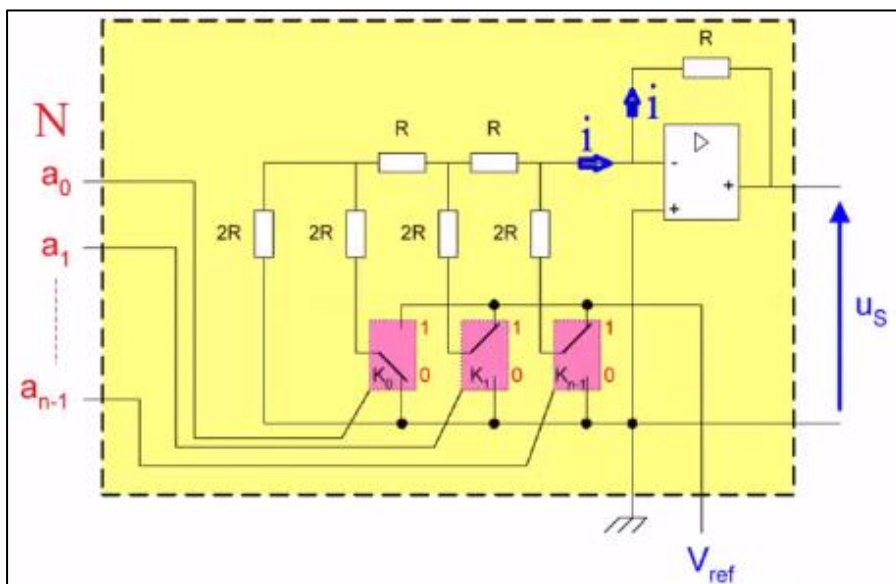


Figure 31 : CAN à réseau de résistances R-2R. [9]

6. EXEMPLE D'UTILISATION

Afin de traiter des signaux et en extraire des informations, il existe de nombreuses variétés d'applications de mesures qui permettent de mettre en œuvre les techniques de traitement de signal. Parmi ces applications on peut citer l'oscilloscope, le scanner, le radar... Dans ce qui suit nous nous baserons principalement sur l'oscilloscope puisqu'il représente l'objectif de notre travail.

6.1. Définition de l'oscilloscope

L'oscilloscope, qu'il soit analogique [Figure 32] ou numérique [Figure 33], est un appareil de mesure qui permet de visualiser la variation dans le temps de grandeurs physiques préalablement transformées en tension grâce à un convertisseur ou des capteurs.



Figure 32: Oscilloscope Analogique.



Figure 33: Oscilloscope Numérique.

Les signaux captés sont tracés sur un graphique XY, appelé oscillogramme [Figure 34], à partir de deux entrées différentes, où l'axe horizontal X représente le temps et l'axe vertical Y représente la mesure de tension.

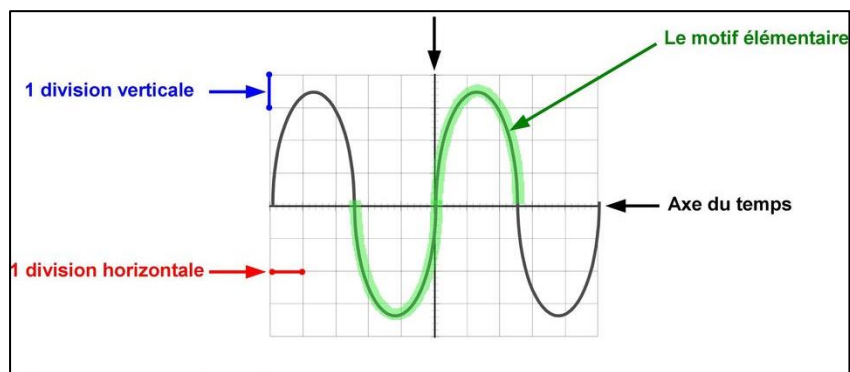


Figure 34: Oscillogramme.

6.2. Le principe de fonctionnement

L'usage le plus commun est avec l'axe Y, contrôlé par une sonde connectée au système testé ; et l'axe X contrôlé par une base de temps interne pouvant fonctionner à différentes fréquences.

Si l'entrée X est laissée non connectée et l'entrée Y est attachée à un signal, qui oscille rapidement ; la trace à l'écran clignote rapidement de haut en bas et trace simplement une

ligne verticale. En appliquant la base de temps sur l'axe X, le signal d'entrée variable est réparti horizontalement, de sorte qu'il est possible de voir comment il varie dans le temps.

Pour lire la courbe sur l'écran de l'oscilloscope :

- Sur L'axe vertical Y (axe des tensions) : on multiplie le nombre de divisions lues par la sensibilité verticale choisie (tension/div).
- Sur l'axe horizontale X (axe des temps) : on multiplie le nombre de divisions lues par la base de temps (temps/div).

6.3. Les types

❖ *L'oscilloscope analogique*

L'oscilloscope analogique est le plus ancien et est en voie de disparition, puisqu'il ne permet que l'observation de tensions périodiques. Il fonctionne par déflexion électrique d'un faisceau d'électrons en fonction de la tension mesurée. En effet, la tension dévie le faisceau verticalement (de haut en bas) et trace ainsi une courbe sur l'écran, qui représente le signal en temps réel.

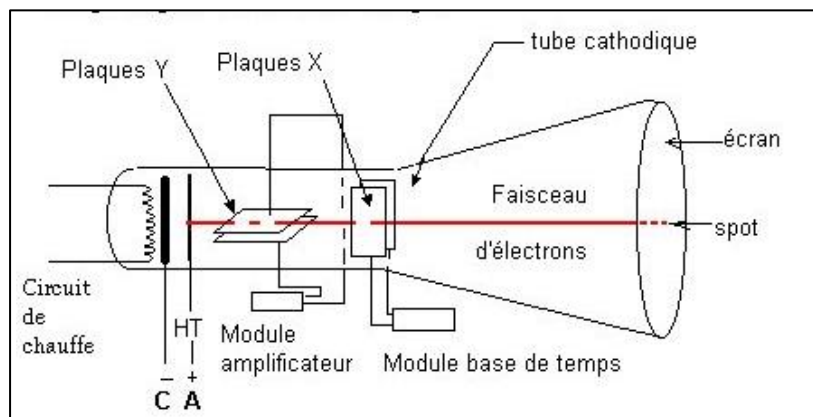


Figure 35: Structure d'un oscilloscope analogique. [10]

Il utilise un amplificateur vertical, un amplificateur horizontal (une base de temps), une alimentation électrique et un tube à rayons cathodiques (CRT) [Figure 35].

Quand une sonde d'oscilloscope est connectée à un circuit [Figure 36], le signal de tension détecté se déplace vers le système vertical de l'oscilloscope analogique, où il sera soit amplifié par l'amplificateur ou réduit par l'atténuateur, selon la configuration de l'échelle verticale. Après le signal transite vers la plaque de déviation verticale du CRT.

La tension appliquée à ces plaques provoque un spot lumineux sur l'écran qui est un faisceau d'électrons frappant le luminophore du CRT. Quand la tension est négative, le point se déplace vers le bas et quand elle est positive, il se déplace vers le haut. Le signal se déplace ensuite vers le système de déclenchement pour déclencher un balayage horizontal, qui consiste en le déplacement du point de la gauche vers la droite sur l'écran. Le déclenchement du système horizontal permet ainsi de démarrer la base de temps, qui va effectuer un balayage dans un intervalles de temps spécifique. Le point lumineux peut également balayer jusqu'à 500 000 fois par seconde.

La répétition du balayage et la déviation verticale permettent la visualisation de la courbe sur l'écran, en vue d'un affichage clair ; le déclencheur stabilise le signal en garantissant que le balayage commence toujours au même point. On déduit ainsi, que les trois paramètres de base d'un oscilloscope analogiques sont :

- *La Base de temps du système horizontal* : Elle est représentée par l'échelle horizontale sur l'écran (temps/div), elle permet d'ajuster le temps par division du signal.
- *L'amplificateur ou l'atténuateur du système vertical* : il est représenté par l'échelle verticale sur l'écran (tension/div), il permet d'ajuster la tension par division du signal avant qu'il ne soit appliqué au système de déflexion vertical.
- *Le déclenchement* : il permet de déclencher la base de temps et aussi stabiliser un signal répétitif.

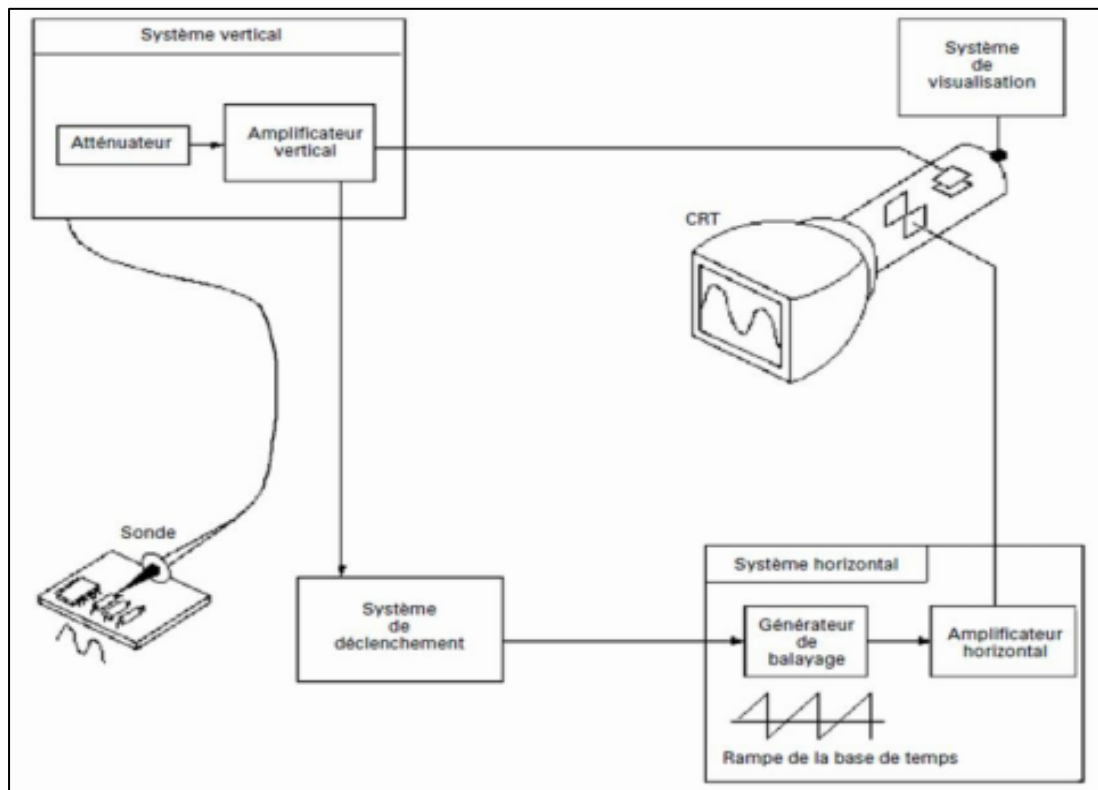


Figure 36: FONCTIONNEMENT DE L'OSCILLOSCOPE ANALOGIQUE. [11]

❖ L'oscilloscope numérique

Ce type d'oscilloscope est le plus répandu actuellement. Il n'utilise pas un tube à rayons cathodique mais plutôt un CAN, pour échantillonner le signal mesuré et le convertir en une information numérique ; qui sera ensuite utilisée pour reconstruire une courbe sur l'écran.

Un oscilloscope numérique est composé d'un amplificateur vertical d'entrée analogique, d'un système d'acquisition formé d'une mémoire, d'un CAN et d'un microprocesseur, un système horizontal qui comprend une horloge et une base de temps, des circuits d'affichage et de reconstruction de la forme d'onde, un écran LCD et une alimentation. [Figure 37]

Quand l'oscilloscope est connecté à un circuit, le système vertical ajuste l'amplitude du signal de la même manière que dans l'oscilloscope analogique ; ensuite le CAN du système d'acquisition échantillonne périodiquement le signal mesuré avec une fréquence

d'échantillonnage déterminée par le déclenchement de l'horloge du système horizontal et quantifie les échantillons prélevés en valeurs numériques qui seront stockées dans la mémoire comme points de la courbe. L'écran permet ainsi de visualiser ces points.

Les fonctions de base d'un oscilloscope numériques sont :

- *La Base de temps du système horizontal* : elle est représentée par l'échelle horizontale sur l'écran (temps/div), elle permet d'ajuster le temps par division du signal.
- *L'amplificateur ou l'atténuateur du système vertical* : il est représenté par l'échelle verticale sur l'écran (tension/div), il permet d'ajuster la tension par division du signal avant qu'il ne soit appliqué au système de déflexion vertical.
- *Le déclenchement de l'oscilloscope* : il permet de déclencher l'horloge d'échantillonnage, qui va permettre de déterminer l'instant de prélèvement des échantillons du signal, il permet également de déterminer le début et l'arrêt d'un enregistrement (longueur d'un enregistrement est le nombre de points nécessaire pour représenter une courbe).
- *Le mode d'acquisition et le mode d'échantillonnage* : ces réglages dépendent principalement de la fréquence et de la complexité du signal, et sont déterminés en ajustant les échelles verticale et horizontale.

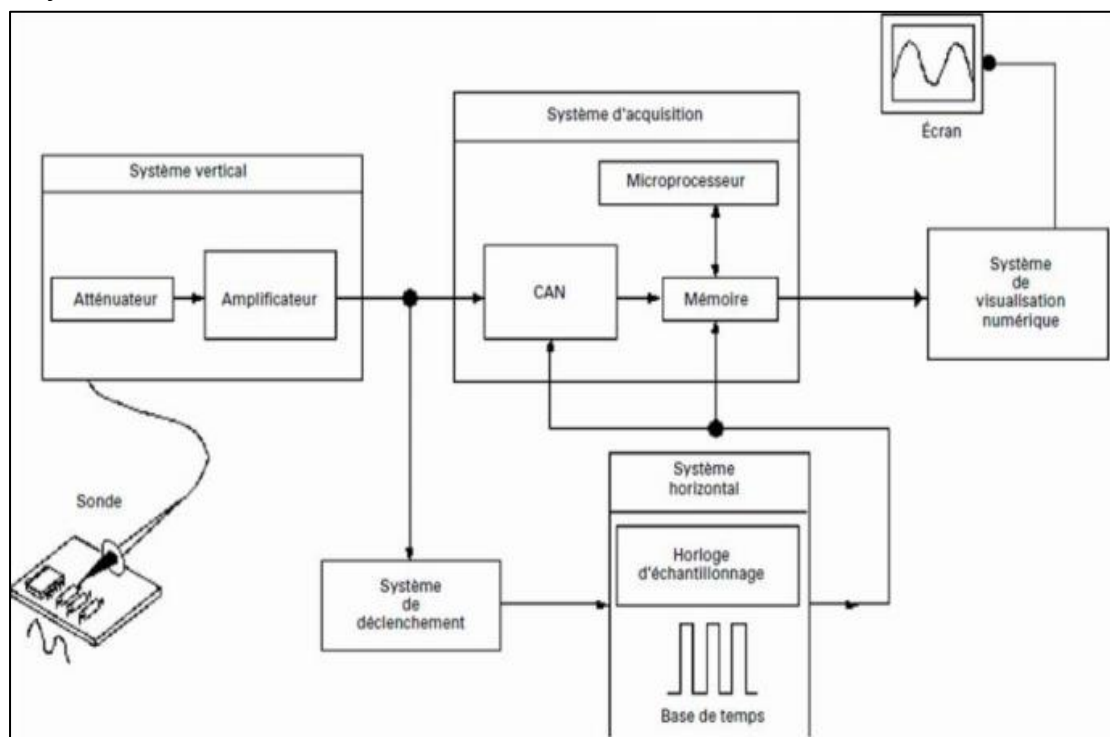


Figure 37:FONCTIONNEMENT de l'oscilloscope numérique. [11]

L'oscilloscope numérique est un dispositif moderne qui implémente les techniques de traitement numérique du signal. Il présente ainsi plusieurs avantages qui ont causé la disparition petit à petit de l'oscilloscope analogique ; on peut citer ces quelques avantages :

- L'analyse du signal en temps réel.
- Le stockage des données numériques, acquises dans des dispositifs externe (mémoire) pour une visualisation ultérieure.

- Il dispose de systèmes d'analyses et de traitements, qui permettent de mesurer les caractéristiques des signaux.
- Il possède plusieurs filtres performants pour améliorer la visibilité des détails, plus un écran à cristaux liquides pour l'économie d'énergie.

6.4. Les caractéristiques

❖ *La bande passante :*

Elle permet d'indiquer la fréquence maximale que l'oscilloscope peut capturer et analyser ; quand la fréquence se rapproche de la fréquence maximale la précision de l'oscilloscope diminue. Elle permet également d'indiquer la fréquence pour laquelle le signal affiché est réduit à 70,7% de l'amplitude à l'entrée (cette atténuation de 70,7% est appelée l'atténuation à -3dB). [Figure 38]

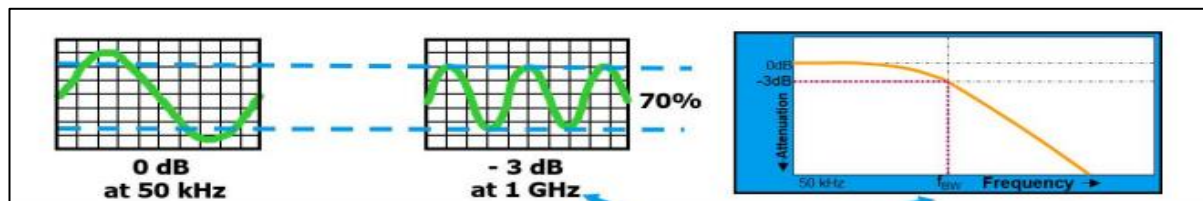


Figure 38: la bande Passante. [11]

❖ *Le temps de montée :*

Le temps de montée de la forme d'onde est la durée que met le signal pour passer de 10% à 90% de sa valeur ; c'est un autre moyen de décrire la fréquence utile d'un oscilloscope. La relation entre le temps de montée et la bande passante est présentée par la formule :
 Temps de montée (en secondes) = 0,35/ La bande passante (HZ).

Un oscilloscope ne peut pas afficher correctement un temps de montée plus rapide que celui spécifié, comme le temps de montée de l'oscilloscope.

❖ *La base de temps (ou balayage ou sensibilité horizontale) :*

Elle correspond à l'échelle horizontale de l'oscillogramme. Un oscilloscope a, en général, deux bases de temps : la principale (P) et la retardée (R). Ces bases permettent de sélectionner un détail dans le signal qui à une vitesse P, pour ensuite zoomer sur tout l'écran avec une vitesse R>P.

❖ *La sensibilité verticale :*

Elle correspond à l'échelle verticale de l'oscillogramme, elle indique de combien l'amplificateur vertical peut amplifier un signal faible à l'entrée. La plus petite tension qu'un oscilloscope d'usage général peut détecter est d'environ 2 mV par division. Il faut régler la sensibilité verticale de telle sorte que la courbe occupe tout l'écran.

❖ *La résolution verticale :*

Résolution d'un instrument en termes de mesure, est le plus petit incrément que l'instrument indique ou affiche. La résolution verticale d'un oscilloscope numérique est liée au nombre de bits du CAN.

❖ *Le couplage :*

L'utilisateur a le choix de filtrer ou non le signal entrant. Il existe trois types de couplage d'entrée (modes d'entrées) : AC, DC, GND.

- En GND (masse) : l'entrée est reliée à la masse de l'instrument, pour servir de référence aux mesures d'amplitude.
- En AC (alternatif) : la composante continue du signal, est bloquée par une capacité et ne sera pas transmise.
- En DC (continu) : toutes les composantes du signal sont transmises, le signal sera transmis tel qu'il est.

❖ *Sonde :*

Une sonde d'oscilloscope est une composante essentielle, pour effectuer la mesure des tensions à laquelle elle est destinée. Elle permet d'augmenter l'impédance de l'oscilloscope.

Elle est composée de deux bouts de configuration différents ayant différentes fonctions :

- Le premier bout est formé d'une pointe conductrice qui permet par contact, de mesurer les signaux électriques.
- L'autre bout est équipé d'un connecteur de type BNC mâle qu'on connecte sur l'oscilloscope.

7. CONCLUSION

Pour conclure ce premier chapitre, nous rappelons brièvement ce qui a été vu, à savoir : Le traitement du signal, ces différentes étapes, les différents composants électroniques qui y sont dédiés ainsi qu'un exemple concret de son utilisation qui est l'oscilloscope et qui représente le principal objet de notre projet de fin d'étude.

Chapitre 2: LES SYSTEMES EMBARQUES ET LE MICROCONTROLEUR STM32

1. INTRODUCTION

Depuis la nuit des temps, l'homme n'a cessé de chercher, d'innover et d'inventer, afin de faciliter sa vie quotidienne. A chaque siècle, voire même chaque année, qui passe, les tabous sont brisés et les limites surpassées. Prenons le simple exemple de l'évolution du domaine de l'informatique et de l'électronique, qui, de nos jours sont intrinsèquement liés.

Vers la fin du 19^{ème} siècle, les savants ont marqués le début de la technologie, grâce à la découverte des rayons cathodiques et les propriétés de l'électron. Quelques années plus tard, en 1904, un nouveau bond fut effectué avec l'invention du tube à vide (Diode) ainsi que la transmission sans fil (manipulation des signaux radio et audio...). Vient par la suite, la grande révolution du transistor, en 1948 durant la seconde guerre, qui propulse le développement de l'informatique et de l'électronique ; ce qui a induit à l'apparition des circuits intégrés. Depuis, ces circuits intégrés subissent une impressionnante évolution. D'ailleurs, de nos jours, on les retrouve enfouis dans différents systèmes de différents domaines tels que : l'automobiles, la téléphonie, la médecine, ... etc. Et c'est ces systèmes-là qui intègrent ces circuits qu'on appelle systèmes embarqués.

Dans ce deuxième chapitre, nous vous présentons ce qu'est un système embarqué, ses caractéristiques et son architecture. Puis nous allons aborder la notion de microcontrôleurs ainsi qu'un exemple, le μ C STM32, qui représente la pièce maitresse de notre travail.

2. LES SYSTEMES EMBARQUES

2.1. Définition

Un système embarqué est un système informatique intégré dans le monde réel dont le but est d'effectuer une tâche bien spécifique, comme par exemple les circuits électroniques qui se trouvent dans les dispositifs mobiles, appareils ménagers, automobiles...

Du point de vue technique, un système embarqué est un système électronique complexe avec du logiciel conçu pour répondre à des fonctionnalités données.

Il dispose d'une architecture semblable à celle des ordinateurs, puisqu'il est composé d'une partie matérielle comportant un processeur, des mémoires, des interfaces d'entrées sorties et un ensemble d'ASIC, et d'une autre partie logicielle représentant des programmes.

2.2. Caractéristiques

Les principales caractéristiques d'un système embarqué sont :

- ❖ *Répondre aux fonctionnalités* : Un système embarqué doit être conçu pour une application cible spécifique.
- ❖ *Autonome* : Il doit pouvoir fonctionner sans une intervention externe.
- ❖ *Fonctionnement en temps réel* : La conception de ces systèmes est souvent basée sur un fonctionnement en temps réel. Ils doivent être réactifs aux événements. Ainsi, le résultat des opérations et traitements effectués doit être délivré dans des délais. En effet, le manque d'échéances entraîne souvent des fautes et des dégradations de performance.
- ❖ *Sûreté et sécurité* : En cas de panne, un système embarqué doit toujours pouvoir fonctionner correctement.
- ❖ *Cout réduit* : Les systèmes embarqués sont sensibles au coût de production. Mais lorsqu'ils sont fabriqués en grande quantité, leurs prix reviennent, ainsi, moins chers.
- ❖ *Poids et volume restreint* : La taille de ces systèmes ne doit pas présenter une contrainte au niveau de leurs portabilités.
- ❖ *Consommation d'énergie* : Une consommation importante d'énergie augmente considérablement le prix, car il faudra des batteries avec une plus grande capacité.

2.3. Architecture générale

Les systèmes embarqués sont conçus autour d'un microcontrôleur. Ils comportent deux parties, une partie programmable, entourés principalement d'ASICs et une autre partie regroupant des capteurs et des actionneurs qui interagissent continuellement avec l'environnement et d'autres systèmes externes.

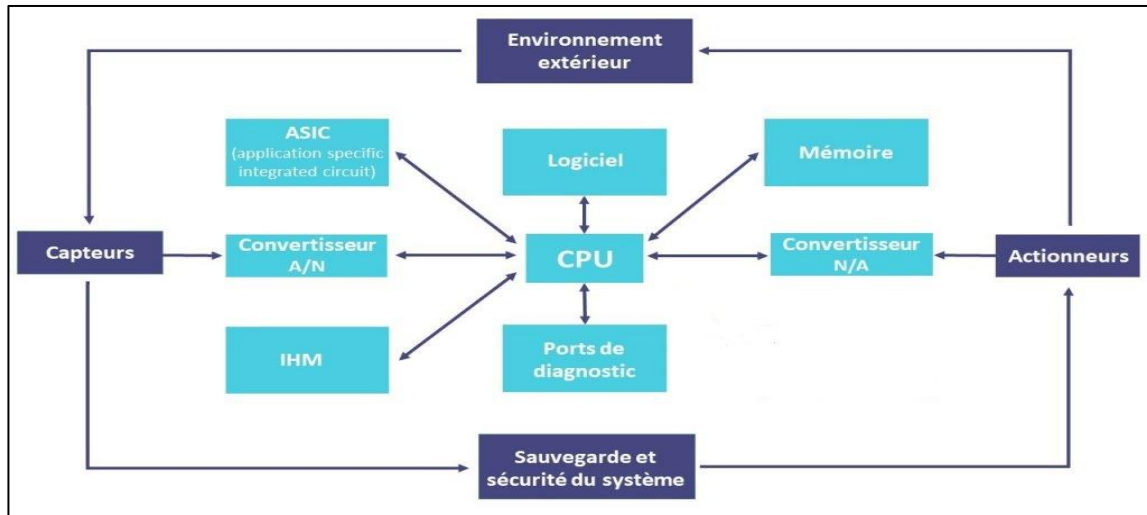


Figure 39:structure générale d'un système embarqué. [12]

L'architecture, représentée dans la [Figure 39] peut varier selon les systèmes. Mais l'architecture de base est la plupart du temps composée de 3 unités principales, qui sont définies comme suit :

- ❖ **L'Unité de captages (d'acquisition)** : Elle capte les informations de l'environnement extérieur avec des capteurs.
- ❖ **L'unité de traitement** : Elle est principalement composée de :
 - CAN : convertisseur analogique numérique, qui convertit les données analogiques captées en données numériques afin que le processeur puisse les traiter.
 - Processeur : Il s'occupe principalement du traitement des données.
 - Mémoire : Elle permet le stockage de données.
 - Logiciel : Il a souvent la fonctionnalité ou la tâche à exécuter qui est spécifique à l'application.
 - ASICs : Ce sont des circuits dédiés à des applications spécifiques.
 - CNA : convertisseur numérique analogique.
- ❖ **L'unité de sortie** : Elle comporte :
 - Les actionneurs : c'est des dispositifs chargés d'agir sur l'environnement, en fonction des informations reçues (ex : led, écran, moteur...). Ils sont couplés au CNA.
 - L'IHM : c'est l'interface homme-machine ; permet la communication entre les deux, comme par exemple : l'écran.

2.4. Classification

Les systèmes embarqués sont classés selon leurs relations avec l'environnement, et sont présentés comme suit [Figure 40] :

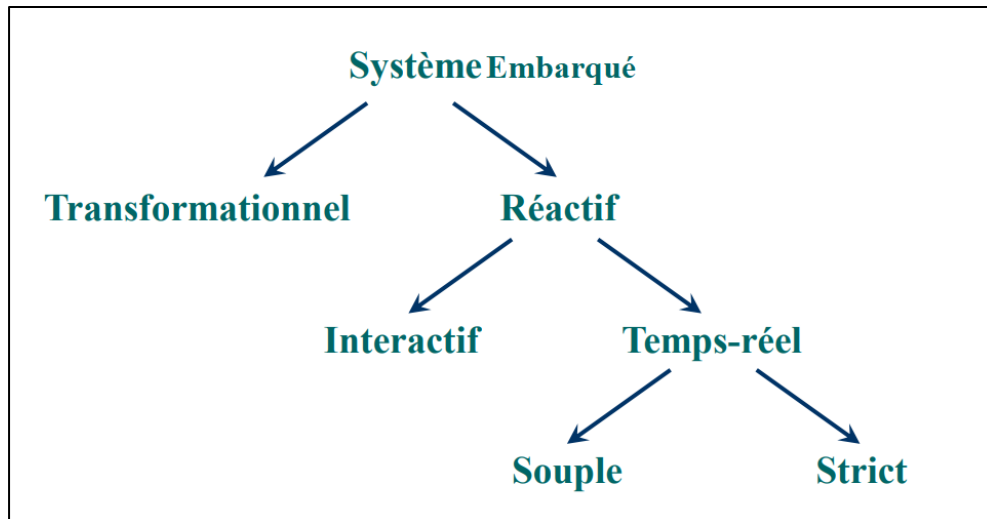


Figure 40: Classification des systèmes. [13]

- ❖ **Systèmes transformationnels** : Leurs fonctions se résument à la lecture de données lors du démarrage, de son calcul indépendant de l'environnement, puis il meurt après avoir fourni le résultat en sortie.
- ❖ **Systèmes Réactifs** : Ils sont attentifs aux événements extérieurs, par conséquent le résultat produit dépend de sa réaction aux événements. On distingue principalement deux types de ces systèmes :
- ❖ **Système Interactif** : son interaction est quasi permanente avec son environnement, et sa vitesse de réaction est propre à lui (défini par le système).
- ❖ **Système temps réel** : son interaction avec son environnement est permanente, et sa vitesse de réaction est déterminée par la nature de l'événement, et est aussi soumis à des contraintes temporelles strictes ou souples.

2.5. Types d'application

Les systèmes embarqués sont conçus pour satisfaire et répondre aux fonctionnalités d'une application spécifique. On peut donc distinguer 4 types d'applications visés :

- ❖ **Calcul général** : Ce sont des systèmes à usage général, avec une exécution similaire à celle des ordinateurs.
Exemple : PDA, jeux vidéo.
- ❖ **Contrôle de systèmes en temps réel** : Ils contiennent généralement un système d'exploitation temps réel (RTOS).
Exemple : système de navigation aérien.
- ❖ **Traitement du signal** : Ils permettent de réaliser des calculs sur de gros flux de données.
Exemple : Radar.
- ❖ **Réseaux et communication** : Ils servent à la transmission de données.
Exemple : téléphone portable.

3. LES MICROCONTROLEURS

5.1. Définition

Un microcontrôleur est un circuit intégrant, à lui tout seul, les éléments indispensables à un ordinateur, tel qu'un processeur, de la mémoire, des unités périphériques et des interfaces d'entrées-sorties destinés aux systèmes embarqués.

Comparés aux microprocesseurs qu'on trouve dans les ordinateurs, les microcontrôleurs permettent de diminuer la taille, la consommation et également le coût des systèmes.

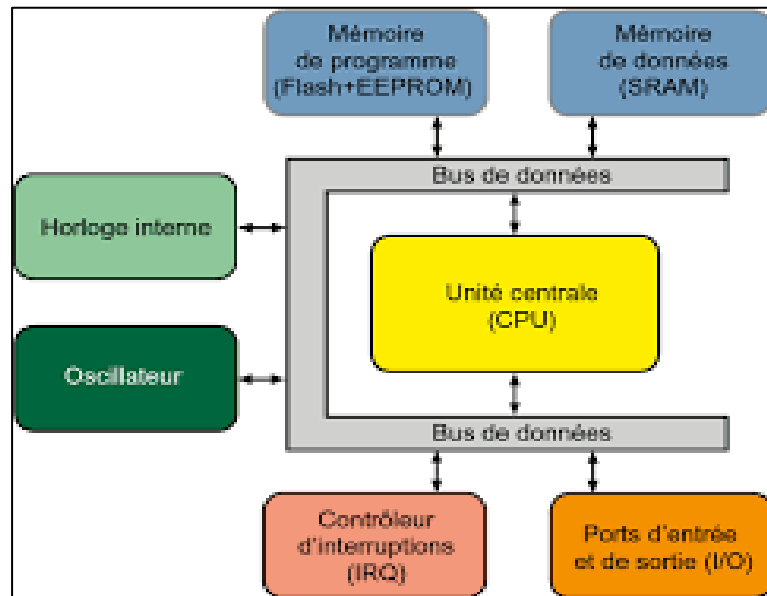


Figure 41: les éléments interne d'un microcontrôleur. [14]

Donc les éléments qu'on peut trouver, généralement dans un microcontrôleur, sont présentés par la [Figure 41] et cités ci-dessous :

- ❖ **Un processeur** : sa capacité peut aller de 4 à 64 bits, selon le modèle choisis. Il effectue principalement le traitement des informations et de données au rythme définie par la fréquence d'horloge interne.
- ❖ **Une mémoire volatile** : dite RAM, est une mémoire à accès aléatoire en lecture et écriture. Elle permet de stocker temporairement les données et variables utilisées au cours d'exécution.
- ❖ **Une mémoire morte** : dite ROM c'est une mémoire, qui contient le programme de démarrage qui permet de charger l'application depuis la mémoire flash. Il existe plusieurs types de ROM : EPROM, EEPROM...
- ❖ **Un oscillateur interne** : qui permet de cadencer le rythme d'exécutions des instructions du processeur.
- ❖ **Les entrées sorties** : ce sont des périphériques d'entrées-sorties, qui offrent la possibilité au processeur de communiquer avec d'autre périphériques externes.
- ❖ **Un watchdog ou chien de garde** : qui va s'assurer que le système ne reste pas bloquer à une étape particulière.

❖ **Autres modules** : dont l'ADC, DAC, Timers, Contrôleurs IRQ...

5.2. Avantages

Les microcontrôleurs offrent plusieurs avantages, dont :

- Diminution de l'encombrement du matériel, grâce à l'intégration de plusieurs éléments.
- Augmentation de la fiabilité du système.
- Consommation réduite d'énergie.
- Réduction des coûts au niveau des composants qu'il intègre et aussi au niveau de la conception.
- Evolution de différents environnements de programmation.

5.3. Les mémoires

Il y'a plusieurs types de mémoires [Figure 42] dans les systèmes embarqués et elles sont classifiées par différent critères.

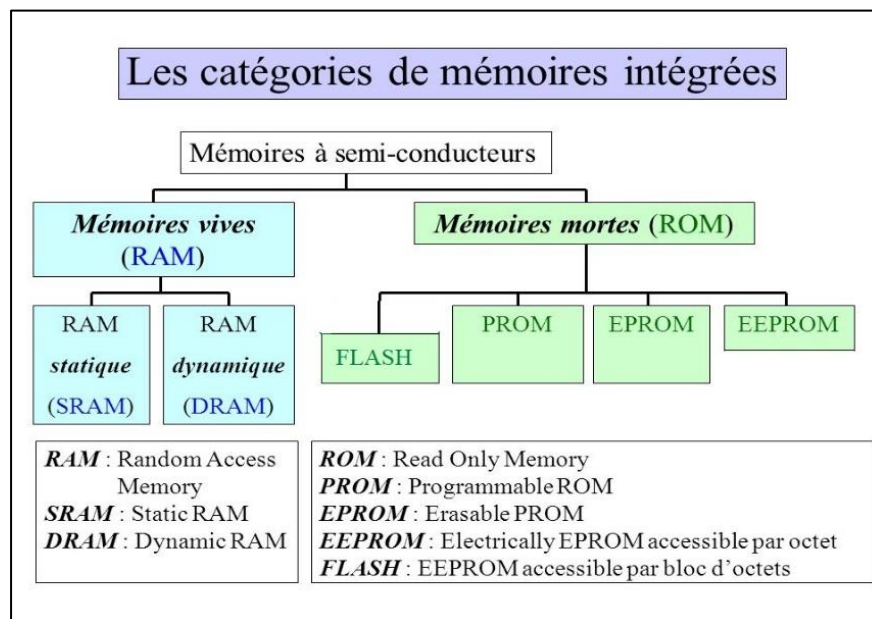


Figure 42:types de mémoires. [14]

❖ **Les mémoires vives ou volatiles (RAM)** : Sont des mémoires à accès aléatoire en lecture et écriture. Elles permettent de stocker temporairement des données. On distingue deux types :

- **SRAM** : C'est une mémoire statique à accès aléatoire, utilisée souvent dans les mémoires caches pour exécuter un bout de code critique (piles...).
 Avantage : très rapide, simple d'utilisation.
 Inconvénient : petite densité de sauvegarde, chère.
- **DRAM** : C'est une mémoire dynamique à accès aléatoire, elle contient le programme en exécution.
 Avantage : grande densité de sauvegarde.
 Inconvénient : Moins rapide, Nécessite un rafraichissement périodique.

❖ **Les mémoires mortes ou non- volatiles (ROM)** : Sont des mémoires à lecture seule avec un temps d'accès lent par rapport à la RAM. Elles contiennent le programme de démarrage, qui permet de charger l'application depuis la mémoire flash. On distingue 4 types :

- **PROM** : C'est une ROM vierge programmable une seule et unique fois. Inconvénient : pas reprogrammable

- **EPROM** : C'est une mémoire qu'on peut programmer électriquement (en lui appliquant une tension de + 12V), et qu'on peut effacer (en l'exposant à des rayons ultraviolets).

Avantage : Reprogrammable.

Inconvénient : peut-être endommagée avec un effacement répété.

- **EEPROM** : C'est une mémoire qu'on peut programmer et effacer en lui appliquant des tensions électriques.

Avantages : L'écriture et l'effacement sans dispositifs, l'effacement répété ne l'endommage pas.

- **FLASH** : Cette mémoire est utilisée pour sauvegarder le programme du système embarqué, qui sera chargé de la mémoire flash à la RAM pour être exécuter. Elle est connue pour sa grande densité de sauvegarde. On peut trouver deux types de mémoire flash :

➤ LA NAND-FLASH : lente mais a une grande densité de sauvegarde.

➤ LA NOR-FLASH : rapide mais a une petite densité de sauvegarde.

Inconvénient : endommagement de la mémoire avec l'écriture répétée.

5.4. Le jeu d'instructions CISC et RISC

Chaque microprocesseur a un jeu d'instruction, qui représente l'ensemble des opérations possible à exécuter. On distingue 4 grandes classes d'instructions :

- Transfert de données : chargement de la mémoire, sauvegarde en mémoire, transfert de registre en registre...
- Opérations arithmétiques : addition, soustraction, division, multiplication.
- Opérations logiques : ET, OU, NON, NAND...
- Contrôle de séquence : branchement, tests...

Il existe deux types de jeu d'instruction [Tableau 1] :

Architecture CISC	Architecture RISC
Jeu d'instruction étendu : le nombre d'instruction varie entre 75 et 150.	Le nombre d'instructions est très réduit, il est entre 10 et 30.
Son jeu d'instruction est complexe : une seule instruction peut désigner plusieurs opérations (LOAD...)	Petite instruction simple, toutes de même taille, ayant toute presque le même temps d'exécution.
L'accès mémoire est plus élevé.	A deux instructions pour la mémoire, cela permet de limiter l'accès mémoire : LOAD, STORE

Latence dans le décodage et l'exécution des instructions.	Accélération en pipeline, en effet le décodage et l'exécution des instruction est relativement rapide.
Résulte un code compact.	Code Moins compacte.
Exemple : Intel x86/pentium, Vax.	Tous les microprocesseurs moderne utilisent ce paradigme : MIPS, ARM, Power Pc...

Tableau 1 : Architecture CISC et RISC.

5.5.Choix d'un microcontrôleur

Il y'a un nombre impressionnant de modèles de microcontrôleurs, qui sont à la portée des utilisateurs. En effet, il existe plusieurs familles proposées par des fabricants. Chaque famille compte des centaines de modèles différents. On peut trouver des microcontrôleurs avec 6 pattes et plus. [Figure 43]

Afin de faire le bon choix du microcontrôleur, quelques critères sur lesquels va se baser notre choix sont proposés :

- Le Nombre de pattes d'entrées-sorties.
- La Taille de la mémoire : RAM, ROM.
- La Consommation d'énergie électrique : tension de fonctionnement, courant consommé.
- L'adaptation de l'architecture interne aux besoins de l'application.
- L'environnement de développement (disponibilité du matériel et du logiciel).

On peut notamment citer quelques familles de microcontrôleurs actuels :

- Microcontrôleurs à 8 Bits :

PIC : de la société américaine Microship qui a popularisé les microcontrôleurs, AVR : de ATMEL utilisé pour l'Arduino, Dérivés du 80C51, C52.

- Microcontrôleurs à 16 BITS : dsPIC de Microship, MSP430 de Texas instrument.

- **Microcontrôleurs à 32 BITS** : AVR32, PIC 32, MIPS, PowerPC ..., ARM : processeur développé par une société anglaise, mais qui a donné la licence à de nombreux fabricants pour développer d'autres microcontrôleurs à base de ce processeur. Ces fabricants sont N/XP, STMicro, Texas, Samsung, Analog Device, Infineon, Freescale... C'est pour cela que les processeurs ARM sont les plus répandus dans le marché, actuellement.

5.6. Processeur ARM

Les processeurs ARM [Figure 44] sont des processeurs à architecture RISC de 32 Bits, qui sont toujours en cours de développement. Il y'a une large gamme de processeurs ARM qui se

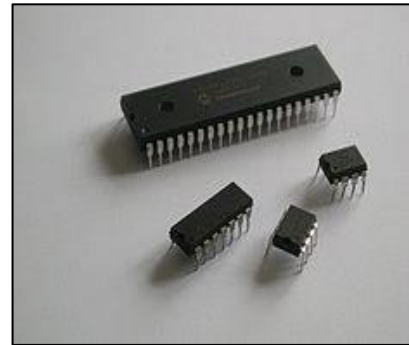


Figure 43: Modèles de microcontrôleurs.

démarquent par leurs architectures et leurs cœurs (Core). Ce sont les processeurs ARM-Cortex.

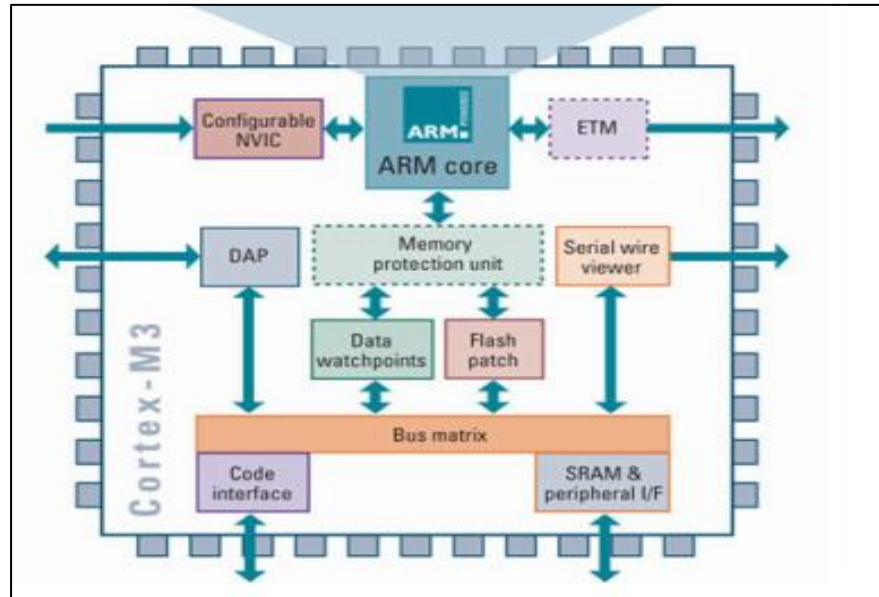


Figure 44:Architecture ARM. [15]

On distingue 3 familles de ces gammes :

- ❖ **Cortex-A (Application)** : Cette gamme est principalement destinée aux applications qui nécessitent d'effectuer des calculs complexes.
- ❖ **Cortex-R (Real-time)** : Cette gamme est destinée aux systèmes temps réel, ses processeurs sont caractérisés par leurs hautes performances.
- ❖ **Cortex-M (eMbedded)** : Les processeurs de cette gamme sont développés pour des microcontrôleurs avec une consommation d'énergie et un coût réduit.

Il y'a différents modèles dans la gamme cortex-M [Figure 45], on distingue :

- **Le cortex-M0/M0+/M1** : Ils permettent de traiter des données générales et des tâches d'entrée-sortie standards.
- **Le cortex-M3** : Il permet de traiter des données avancées, et s'occupe de la manipulation des champs de bits.
- **Le cortex-M4** : C'est un M3 mais avec une extension d'un module DSP (Digital Signal Processor), pour faire du traitement de signal.
- **Le cortex-M4F** : C'est un M4 avec une FPU (Floating Point Unit/ Unité à virgule flottante) en plus.
- **Le Cortex-M7** : C'est le plus récent, son niveau de performance est supérieur.



Figure 45:ARM Cortex-M. [16]

4. LA STM32

4.1. Présentation

STMicroelectronics est une société de développement et de réalisation de solutions destinées à un grand nombre d'applications microélectroniques. Elle fournit une vaste gamme de cartes électroniques, comme la famille microprocesseurs STM32FXXXX. Parmi ces cartes, nous citons la STM32F446RE NUCLEO qui représente l'élément clé de notre projet.

La carte STM32F446RE NUCLEO [Figure 47] est développée autour du microcontrôleur ARM Cortex-M4F 32Bits, qui a une architecture RISC. Elle s'appuie sur les deux écosystèmes Mbed et Arduino ce qu'il lui permet une compatibilité avec les Shields Arduino. Elle utilise également un pipeline 3 étages (chargement, décodage, exécution), ce qui permet l'exécution de plusieurs instructions en un cycle CPU.

Le kit de découverte STM32F4 est fournis avec une bibliothèque logicielle complète (firmware) qui est STM32Cube, la plateforme matérielle du kit permet ainsi d'accéder aux fonctionnalités du microcontrôleur, et de développer de nombreuses applications facilement, en exploitant ses capacités.

Les caractéristiques de la carte STM32F446RE NUCLEO sont les suivantes :

❖ **Un microcontrôleur STM32F446RE** : Avec :

- Un processeur ARM Cortex-M4F 32 bits fonctionnant à une fréquence de 180 MHz [Figure 46], Doté de :
 - Une FPU : Elle permet d'effectuer des calculs en virgule flottante, en quelques cycles.

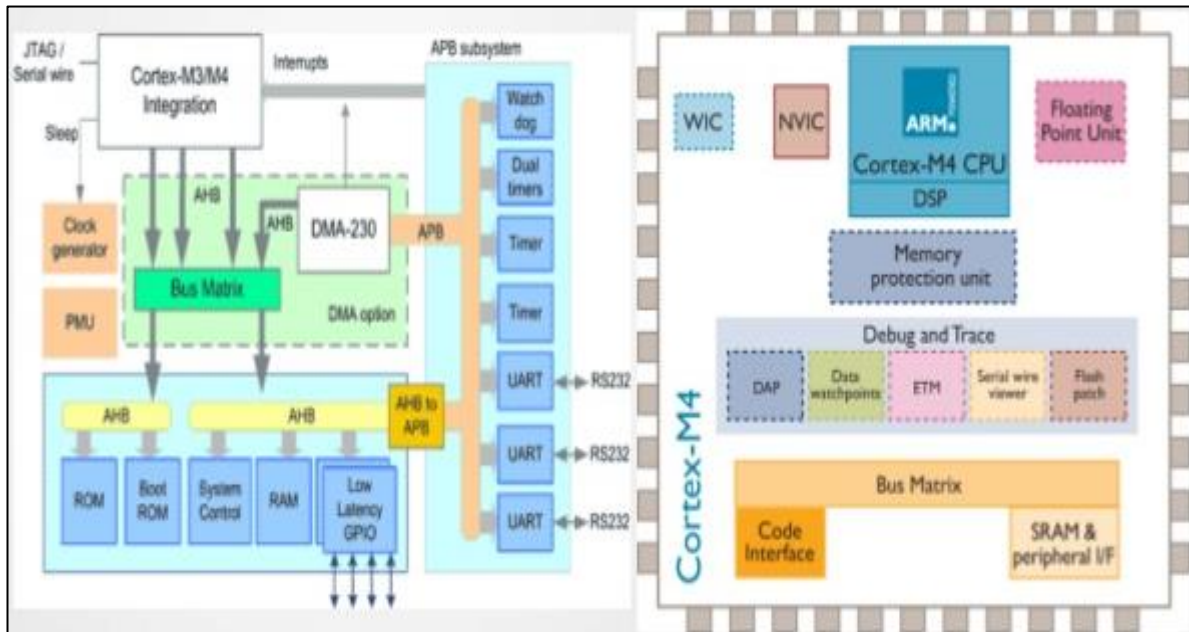


Figure 46: ARM Cortex-M4. [16]

- Un DSP : Il permet de faire du traitement de signal car il comporte un ensemble de fonctions particulières qui lui permettent d'augmenter ses performances dans ce domaine
- NVIC : Unité de contrôle d'interruption qui peut gérer 240 sources d'interruption
- WIC : Cette unité permet de détecter les interruptions et aussi de réveiller le processeur.
- Debug and trace : Il contient des unités pour le débogage et le traçage de données (ETM, DAP...)
- MPU : Unité de protection de la mémoire, elle permet de contrôler l'accès mémoire.
- SRAM.
- Bus Matrix de 32bits : La société ARM propose le bus AMBA qui repose sur la norme multi-bus multi-maitres, il est composé de 2 bus :
 - AHB : c'est un bus système multi-maitres et rapide, il permet de connecter des périphériques rapides comme un processeur à la mémoire ou à d'autres maitres.
 - APB : c'est un bus lent connecté au système (AHB) via un bridge, il permet d'interfacer des périphériques lents (d'E/S...) au système.
- Une mémoire flash de 512 Ko, une SRAM de 128 Ko et une SRAM de sauvegarde de 4 K0, trois ADC de 12 bits avec une fréquence de 2.4 MSPS, deux DAC de 12 bits, SPI, deux UARTs, quatre USARTs, quatre interfaces I2C de 11,25 Mbits/s, quatre interfaces SPI de 45Mbits/s, DMA à usage général, RTC, 17 TIMERS : 1 SysTick timer, 2 watchdog de timers, 12 timers de 16bits et 2 timers de 32bits chacun avec une fréquence de 180MHZ.
- ❖ Un ST-LINK/V2 intégré.
- ❖ Alimentation de la carte.
 - Par bus USB.

- Par alimentation externe : 3 V ou 5 V.
- ❖ **3 LEDs :**
 - LD1 (rouge/verte) pour la communication USB.
 - LD2 (verte) LED utilisateur.
 - LD3 (rouge) pour l'alimentation.
- ❖ **Deux boutons-poussoirs (USER : utilisateur et RESET : réinitialisation).**
- ❖ **Interface USB OTG avec connecteur micro-AB.**
- ❖ **Oscillateur à Cristal de 4 à 26 MHz.**
- ❖ **Oscillateur 32 kHz pour RTC avec étalonnage**
- ❖ **Deux types d'extension :** En-têtes de broches d'extension pour les E/S (appelé ST Morpho headers) pour une connexion rapide à la carte, Connecteurs ARDUINO UNO V3.
- ❖ **Logiciel gratuit :** complet et comprenant une variété d'exemples, constituant le package logiciel STM32CubeF4.

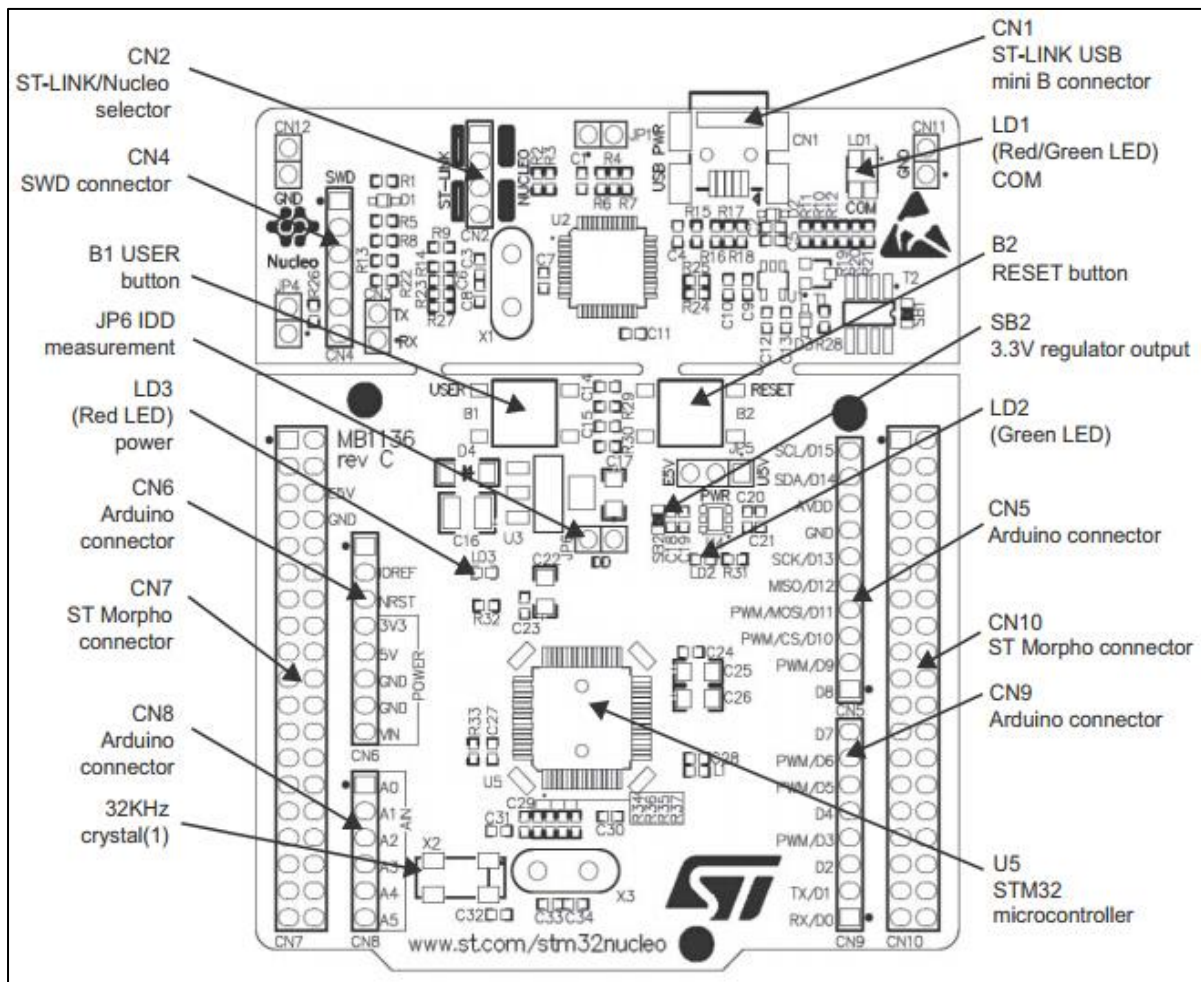


Figure 47: Les composants de la carte STM32F4.[17]

4.2. Fonctionnalités

La STM32F446RE NUCLEO présente plusieurs fonctions d'acquisitions, de traitement et de communication. Nous allons donc présenter uniquement les fonctionnalités en relation avec notre projet, dont :

❖ Les GPIOs

Un GPIO est un moyen de communication entre le microcontrôleur et les périphériques externes.

L'µc STM32F446RE est suffisamment munie de broches d'entrées-sorties à usage général ; ayant typiquement 50 broches bidirectionnels, programmable à un niveau de tension 3.3V. Ces broches sont disposées en plusieurs ports avec différentes lignes d'E/S.

Ces ports sont nommés A, B, C, D, E et H et sont tolérant à 5V (PA0-15, PB0-15, PC0_15, PD2, PH0-1).

Donc, pour configurer un périphérique relié à une broche, il suffit de configurer les GPIOs Correspondant en utilisant le module HAL_GPIO.

Les caractéristiques principales de la GPIO sont :

- Un GPIO a différents modes de configuration [Tableau 2-2], ils sont présentés dans le tableau qui suit :

Modes	Description
D'entrée : GPIO_MODE_INPUT	Mode d'entrée : c'est le mode de base pour une entrée (elle n'a aucune source de signal). On peut fixer l'état des résistances : à 0 Pull-up, à 1 pull down.
De sortie : GPIO_MODE_OUTPUT_	_OD : Mode Open Drain en sortie.
GPIO_MODE_ANALOG	_PP : Mode Push-Pull en sortie.
Alternative : GPIO_MODE_AF_	Mode Analogique
Interruption : GPIO_MODE_IT_	_OD : Mode Open Drain en Alternative.
	_PP : Mode Push-Pull en Alternative.
	_RISING : détection d'interruption au front montant.
	_FALLING : détection d'interruption au front descendant.
	_RISING_FALLING : au front montant et descendant.
Evènement : GPIO_MODE_EVT_	_RISING : détection de déclenchement d'évènement au front montant.
	_FALLING : au front descendant.
	_RISING_FALLING : au front montant et descendant.

Tableau 2 : Modes de configuration d'un GPIO. [18]

- Un multiplexage très flexible des broches : permet l'utilisation des broches d'E/S comme GPIO avec une simple configuration sans, pour autant connaître les registres, leurs mappages en mémoires ou la manière de configuration des périphériques.
- Attribution séparée des vitesses pour les E/S.

❖ *Les UARTs :*

La communication avec l'UART se fait par l'envoi d'une série de bits (trame) sur un seul fil. [Figure 48]

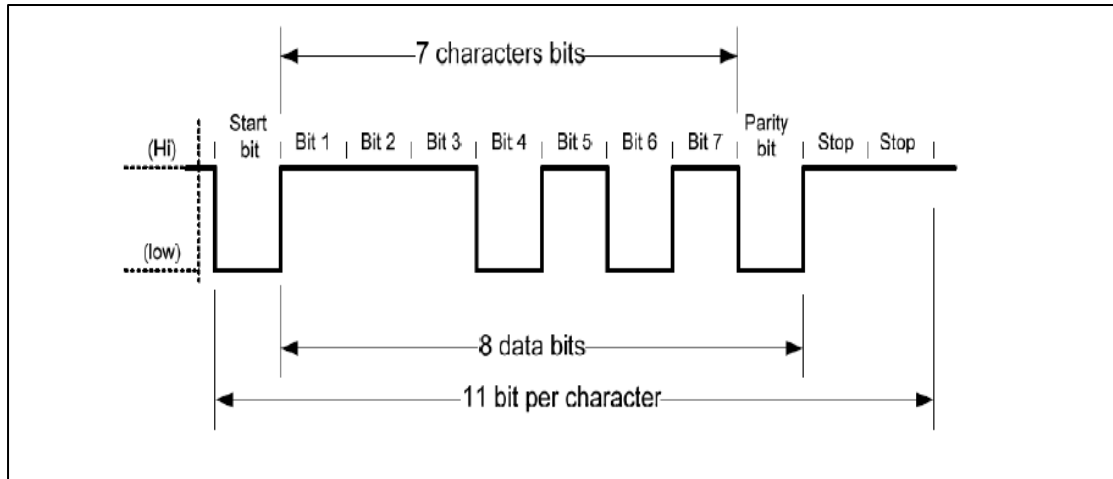


Figure 48: Communication de données dans l'UART. [18]

Dans la transmission asynchrone, la trame est constituée de :

- Un bit de démarrage (START).
- Une Séquence de 8 bits de données.
- Un bit de parité (PARITY) qui peut être utilisé pour détecter des erreurs.
- Un bit (ou deux) d'arrêt (STOP).

Les caractéristiques de l'UART sont :

- BaudRate : Il représente la vitesse de transmission des bits (Bits/sec), ces valeurs : 2400, 9600, 19200, 38400, 57600, 115200, ... sont les valeurs communes de BaudRate.
- WordLength : C'est le nombre de bits que peut contenir une trame (8 ou 9 bits sans inclure les bits généraux, tels que les bits START et STOP).
- StopBits : c'est le nombre de bits STOP transmis (1 ou 2 bits), ils permettent de signaler la fin d'une trame.
- Parity : il indique le mode de parité (Aucune, paire ou impaire), il permet aussi de vérifier s'il y'a eu des erreurs lors de la transmission.
- Mode : il indique le mode de l'UART, s'il est en Emission Tx, ou en Réception Rx, ou à la fois en émission et en réception.
- HwFlowCtl : il indique le mode de contrôle de flux matériel RS232 ; on distingue 4 modes :
 - Désactivé.
 - RTS est activé.
 - CTS est activé.
 - RTS et CTS sont activés.
- OverSampling : c'est une technique qui permet d'échantillonner un signal, avec une fréquence d'échantillonnage supérieure à la fréquence du signal. Elle peut prendre 8 ou 16 échantillons pour chaque bit de trame.

❖ **Timer :**

En général, un Timer est un compteur électronique, avec une fréquence de comptage qui représente une fraction de son horloge source. Il a une résolution qui dépend de son architecture et contient donc l'amplitude des valeurs sur lesquelles il peut compter. Par exemple un timer de résolution de 8 bits peut compter de 0 à 255 et génère un événement lorsqu'il atteint 255.

La fréquence de comptage d'un timer est déterminée par son horloge. Pour ralentir celle-ci, le prescaler (diviseur) intervient en effectuant une division sur la fréquence de l'horloge. Quant à l'Autoreload (chargement automatique), il contient la capacité maximale du compteur. Par conséquent, il est associé au compteur, afin de contrôler le débordement. [Figure 49]

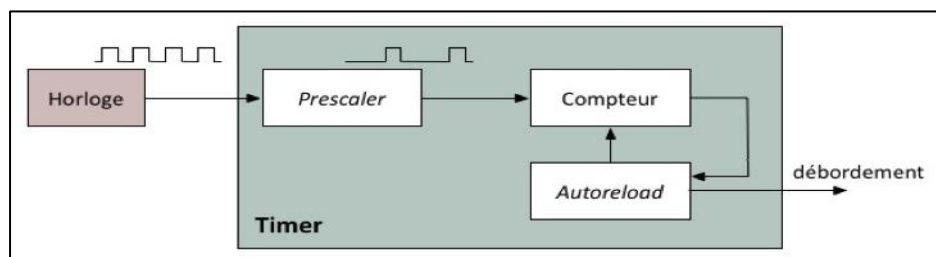


Figure 49: Structure d'un timer.

Les Timers du μ C STM32F446RE se disposent en les catégories suivantes :

- **Les timers de base** : Ce sont des timers à 16 bits sans broches d'entrées-sorties, utilisés comme générateurs de base de temps, on trouve deux timers dans l' μ C STM32F446RE : TIM6 et TIM7.

Les timers de base sont utilisés principalement pour le déclenchement de l'ADC/DAC et peuvent également être utilisés comme "maîtres" pour les autres timers (utilisés comme esclaves).

- **Les timers à usage général** : Ce sont des timers à 16/32 bits. Ils sont en tout dix (TIM2-TIM5 sont de 32 bits, TIM9-TIM14 sont de 16 bits), utilisés aussi comme générateur de base de temps. Ils fournissent également des fonctionnalités classiques d'un timer d' μ C moderne.

Ces timers sont généralement utilisés dans les applications pour :

- Une comparaison de sortie (synchronisation et génération de délai).
- Une capture d'entrée (pour mesurer la fréquence d'un signal externe).
- La génération d'un signal PWM.

Les timers de cette catégorie fournissent quatre canaux d'E/S programmables exceptés pour :

- Les timers à 1 canal.
- Les timers à 2 canaux.

- **Les timers avancés** : TIM1 et TIM8 sont les timers avancés du μ C STM32F446RE. Ils sont à 16 bits et ils représentent les timers les plus complets de ce microcontrôleur. En plus des fonctionnalités d'un timer à usage général, ils comportent plusieurs fonctionnalités liées à la commande de moteurs et aux applications de conversion de puissance numérique...

- **Le timer SysTick** : Ce timer est dédié principalement aux RTOS, mais peut également être utilisée comme un compteur descendant (downcounter) standard.
- **Watchdogs timer** : On distingue deux watchdogs :
 - Un watchdog (chien de garde) indépendant : Il fonctionne indépendamment de l'horloge principale en effet il est cadencé par un RC interne de 32 KHZ. Il peut remplir deux fonctions : Celle de chien de garde pour réinitialiser l'appareil en cas de problème, ou celle d'un timer d'exécution libre pour la gestion des délais d'expiration des applications.
 - Un watchdog de fenêtre : Il est cadencé à partir de l'horloge principale, il peut être utilisé comme chien de garde pour réinitialiser l'appareil si un problème survient, avec une capacité d'interruption d'alerte précoce.

❖ ADC

Le microcontrôleur STM32F446RE a trois modules ADC (ADC0, ADC1, ADC2) [Figure 51] identiques de 12 bits qui sont basés sur la méthode d'approximations successives. Chaque module possède jusqu'à 19 canaux multiplexés :

- 16 entrées externes.
- 3 entrées internes : V_{REFINT} , V_{BAT} , temp-sensor.
- Une entrée de référence analogique positive V_{REF+} : ($1.8v \leq V_{REF+} \leq V_{DDA}$).
- Une entrée d'alimentation analogique V_{DDA} ($1.8v \leq V_{DDA} \leq 3,6V$) : (pour une vitesse réduite) ou ($2,4v \leq V_{DDA} \leq 3,6V$) : (à pleine vitesse).
- Une entrée de référence analogique négative V_{REF-} : ($V_{REF-} = V_{SSA}$).
- Une entrée de masse analogique V_{SSA} .
- Un bloc de conversion analogique numérique.
- Un bloc de registre de données de 16 bits.
- Un bloc de commande d'interruption.
- Un bloc de déclenchement de la conversion.
- Un watchdog analogique qui permet de détecter si la tension d'entrée est au-delà des seuils définis par l'utilisateur.

Pin Number	Pin Name	Channel Name
23	PA0	ADC{1, 2, 3}_IN0
24	PA1	ADC{1, 2, 3}_IN1
25	PA2	ADC{1, 2, 3}_IN2
26	PA3	ADC{1, 2, 3}_IN3
29	PA4	ADC{1, 2}_IN4
30	PA5	ADC{1, 2}_IN5
31	PA6	ADC{1, 2}_IN6
32	PA7	ADC{1, 2}_IN7
35	PB0	ADC{1, 2}_IN8
36	PB1	ADC{1, 2}_IN9
15	PC0	ADC{1, 2, 3}_IN10
16	PC1	ADC{1, 2, 3}_IN11
17	PC2	ADC{1, 2, 3}_IN12
18	PC3	ADC{1, 2, 3}_IN13
33	PC4	ADC{1, 2}_IN14
34	PC5	ADC{1, 2}_IN15

Figure 50: LES CANAUX D'ENTRES ANALOGIQUES DU MICROCONTROLEUR STM32F446RE. [19]

Les modules ADC partagent 16 canaux d'entrées analogiques, la relation entre les pins du μC et les canaux est présentée par la [Figure 50].

L'ADC de la STM32F446RE NUCLEO présente plusieurs caractéristiques :

- Résolution de l'ADC est configurable à 12, 10, 8 ou même 6 bits.
- Deux principaux modes de fonctionnement :
 - ❖ Mode indépendant (independent) : Chaque module ADC fonctionne de manière autonome et sans aucune dépendance mutuelle.
 - ❖ Mode double (dual) : Deux unités ADC travaillant mutuellement ensemble comme s'il s'agissait d'une seule unité.

- Mode de Conversion :
 - ❖ Single : conversion simple, qui peut être une seule conversion à partir d'un canal.
 - ❖ Scan : conversion multicanale, qui peut être une seule conversion à partir de plusieurs canaux.
 - ❖ Single continuous : conversion simple et continue, qui peut être plusieurs conversions à partir d'un canal.
 - ❖ Scan continuous : conversion multicanale continues, qui peut être plusieurs conversions à partir de plusieurs canaux.
 - ❖ Discontinuous : conversion séquentielle à partir de quelques canaux dans un groupe.
- Le déclencheur de l'ADC peut être déclenché par logiciel (HAL_ADC), par matériel interne (timers) ou externe (les broches EXTI-11, EXTI-15), afin d'effectuer l'opération de conversion analogique-numérique.
- Le stockage de données est configurable avec DMA (Direct Memory Access).
- L'horloge du bloc ADC (ADCCLK) est générée à partir de l'horloge APB2, divisée par un prescaler programmable ; cela permet donc à l'ADC de fonctionner avec des vitesses d'horloges allant jusqu'à 42 MHz.
- Le temps de conversion total est le temps d'échantillonnage + 12 cycles d'horloge, cela permet un taux d'échantillonnage et de conversion allant jusqu'à 2,4 Msps (millions of samples per second).
- Un groupe de conversion se compose d'une séquence de conversions qui peut être effectué sur n'importe quel canal et dans n'importe quel ordre. La conversion est organisée en deux groupe :
 - ❖ Groupe régulier : il est composé de jusqu'à 16 conversions, c'est un groupe de canaux fixe régulièrement converti.
 - ❖ Groupe injecté : il est composé de jusqu'à 4 conversions, il peut interrompre temporairement une conversion d'un groupe régulier car il est plus prioritaire,

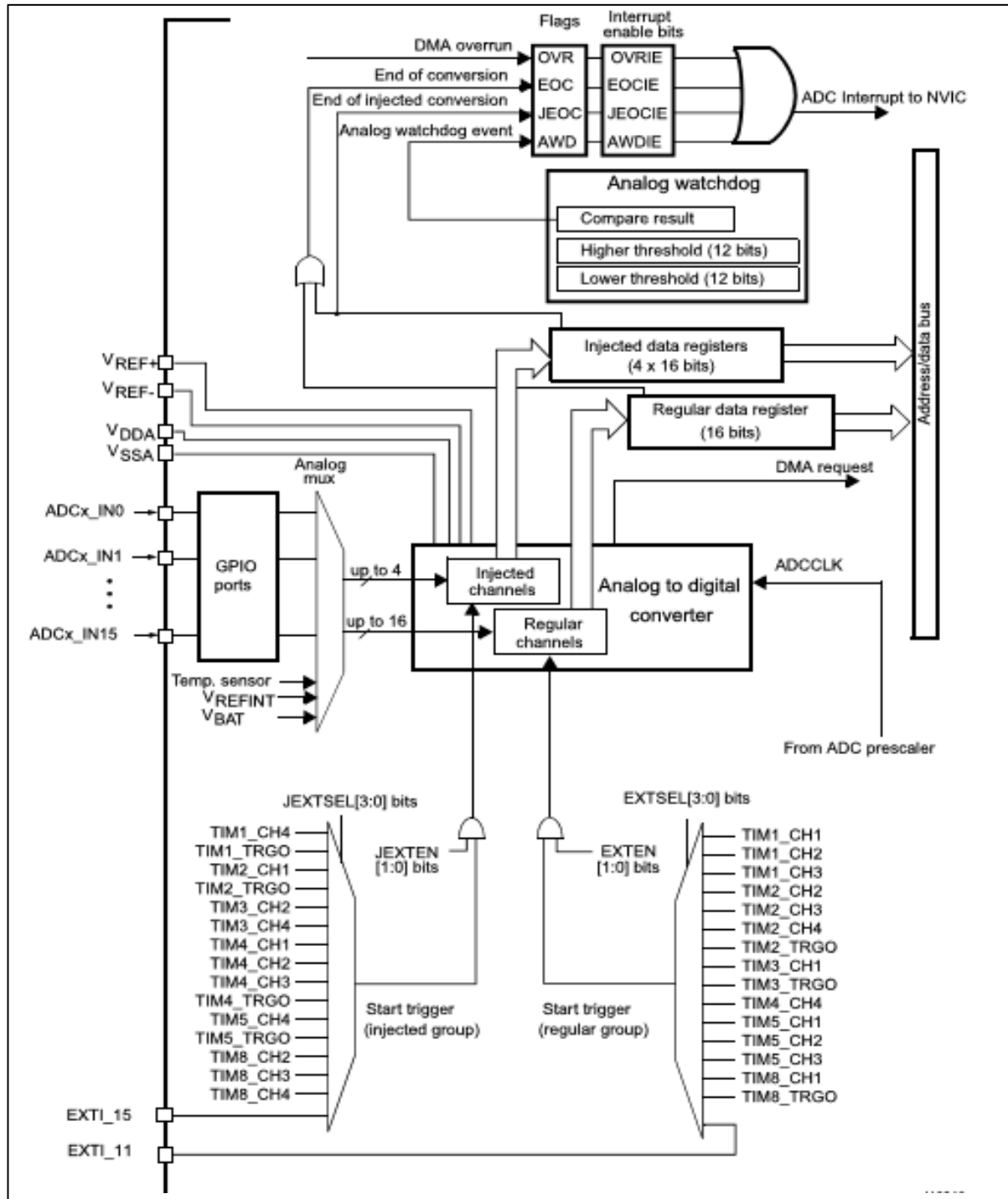


Figure 51: bloc de diagramme de l'ADC. [18]

❖ DMA

Le DMA est un module important du microcontrôleur [Figure 52]. En effet, étant donné que le stockage des valeurs converties d'un canal est effectué dans un seul registre de données, il est important d'utiliser le module DMA pour une conversions de plusieurs canaux réguliers, ce qui permet d'éviter la perte de données.

Il est également utilisé pour effectuer un transfert de données à haute vitesse en arrière-plan, sans avoir recours aux ressources du processeur. Ce qui signifie que le processeur peut effectuer d'autres tâches en même temps.

Le microcontrôleur STM32F446RE possède deux modules DMA, chaque module est caractérisé comme suit :

- Le DMA est basé sur une architecture complexe de bus matrix, il combine deux architectures du bus maître AHB avec FIFO pour optimiser la bande passante du système.
- Il a deux Architectures du bus AHB master (maitre), une dédiée aux accès mémoire et une dédiée aux accès périphériques.
- Il a une interface de programmation AHB slave (esclave) prenant en charge uniquement les accès 32 bits.
- Le DMA a un total de 8 flux (16 flux pour les 2 DMA).
- Chaque flux du DMA a jusqu'à 8 canaux sélectionnables, selon les bits CHSEL [2 :0] dans les DMA_SxCR.
- Chaque canal est dédié à un périphérique spécifique et peut être sélectionner par logiciel. Ce qui permet à plusieurs périphériques d'utiliser le DMA au même temps.
- Chaque canal a un tampon FIFO, d'une longueur de 4 mots (4 à 32 bits), ces tampons :
 - ❖ Peuvent être utilisés pour stocker, temporairement, des données entrantes ; ensuite quand le seuil sélectionnée ($\frac{1}{4}$, $\frac{2}{4}$, $\frac{3}{4}$, ou plein) est atteint, le transfert pourra commencer (mode FIFO).
 - ❖ Peuvent également être utilisés en mode de transfert direct (mode direct).
- Mode de transfert : normal, circulaire (cas de tampons circulaires).
- Types de transfert :
 - ❖ De mémoire en mémoire (seul le 2-ème module DMA pourra être utilisé)
 - ❖ Transfert de mémoire en périphérique, et de périphérique en mémoire.
- Le module DMA est configuré pour incrémenter automatiquement l'adresse source et l'adresse destination, après chaque transfert de données.
- Les priorités entre les demandes de flux DMA sont programmables par logiciel (4 niveaux, composé de : très haut, haut, moyen, bas) ou matériel en cas d'égalité (demande 0 a priorité sur la demande 1, etc.). Ces priorités sont gérées par l'arbitre (Arbiter).
- Les tampons FIFO peuvent être configuré comme double tampon (transfert en Ping-Pong).
- Lorsque le mode FIFO est sélectionné, le transfert de donnée en rafale peut être utilisé avec une taille de rafale de 4, 8 ou 16 unités de données.
- Le DMA à 5 flags d'événement (DMA Half Transfer, DMA Transfer complete, DMA Transfer Error, DMA FIFO Error, Direct Mode Error) rassemblé dans une seule requête.

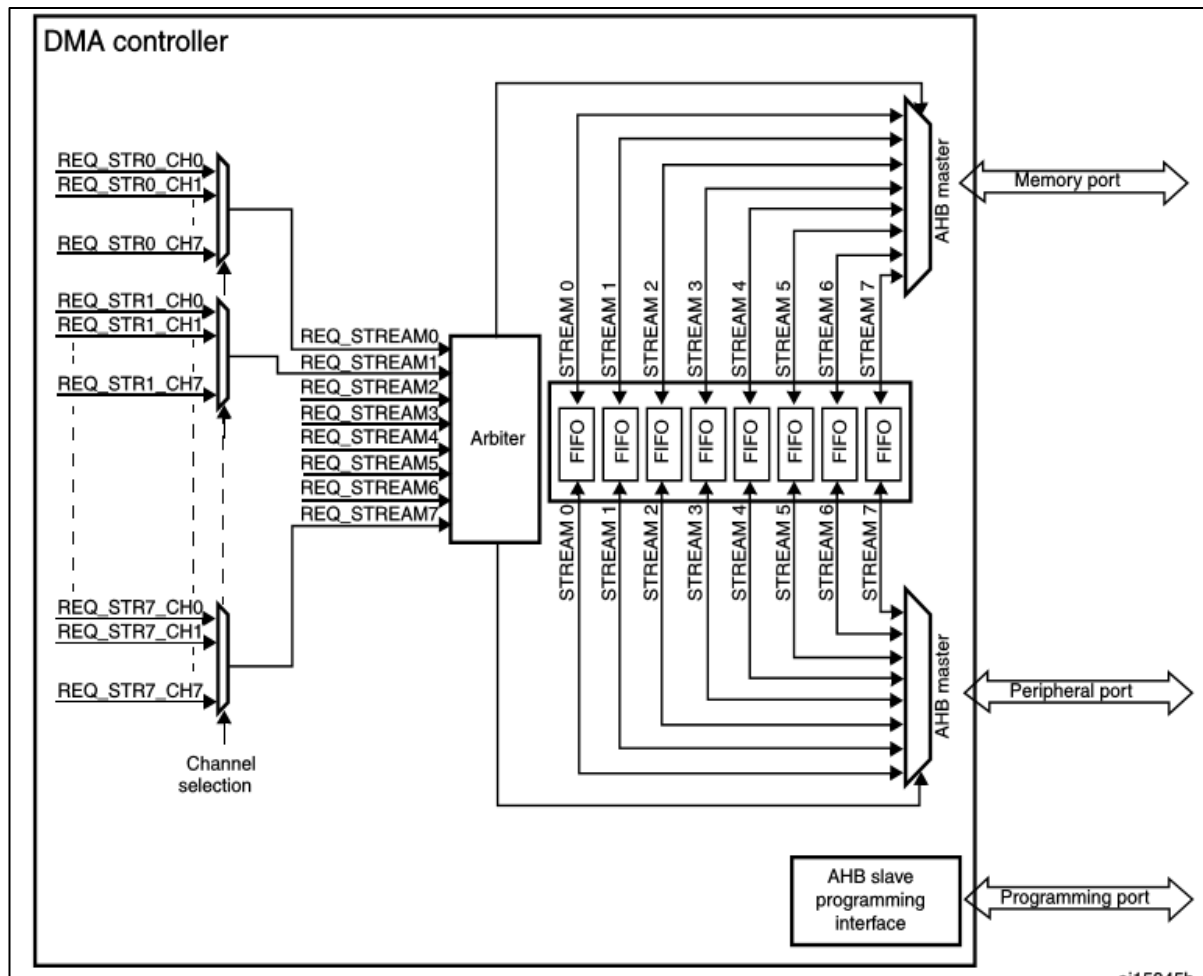


Figure 52: Bloc diagramme du DMA. [18]

5. CONCLUSION

A l'issue de ce chapitre, nous avons expliqué en premier lieu, quelques notions sur les systèmes embarqués en général et les microcontrôleurs en particulier. Puis nous sommes passés à la présentation de microcontrôleur de la carte STM32F446RE NUCLEO, ses caractéristiques et les principales fonctionnalités sur lesquelles nous allons travailler.

Dans le chapitre suivant, nous allons entamer notre conception où nous avons présenté l'architecture de notre projet et les différents aspects qu'il constitue.

Chapitre 3: CONCEPTION

1. INTRODUCTION

Comme tout projet informatique, la conception est une étape primordiale puisqu'elle nous permet de structurer les idées et de définir les besoins et les objectifs visés afin de mieux s'organiser et de préparer le terrain pour la réalisation. C'est donc dans cette optique que nous allons présenter dans ce chapitre notre conception du système en question.

Nous allons commencer par la présentation du projet dans sa globalité et détailler par la suite chacune des étapes suivies pour l'accomplissement de la tâche principale.

2. PRESENTATION DU PROJET

Notre travail consiste en la conception et la réalisation d'un oscilloscope numérique à base d'un microcontrôleur STM32.

Pour y parvenir nous avons conçu une application embarquée dans la carte de développement STM32-Nucleo et une application desktop.

En ce qui concerne l'application embarquée, elle a pour but de capturer des signaux analogiques, de les traiter en utilisant les techniques de traitement du signal adéquates, pour pouvoir les transmettre [figure 53]. Quant à l'application desktop elle nous permettra de visualiser les signaux transmis par l'application embarquée sous forme de graphe appelé oscillogramme.

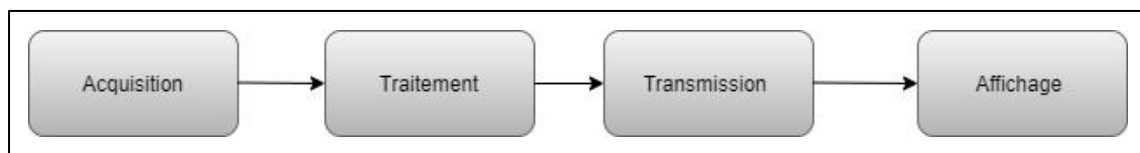


Figure 53: Etapes suivies par le système.

3. PROBLEMATIQUE

Un oscilloscope numérique se doit d'assurer un bon traitement du signal puisqu'il faut avoir en sortie un signal similaire au signal analogique en entrée. Le critère essentiel pour une conversion analogique-numérique optimale est la fréquence d'échantillonnage puisque selon le théorème de Shannon elle doit être au minimum égale à deux fois la fréquence du signal à convertir. C'est donc là que se pose notre première problématique.

Concernant la visualisation, une communication doit être établie pour faire le pont entre l'application embarquée qui traite le signal et l'application desktop qui l'affichera.

Toutefois, pour une meilleure performance, nous avons pensé à élargir la plage de tensions acceptées par la carte de développement STM32F446RE NUCLEO, afin de lui permettre l'acquisition de signaux allant de -12V à 12V.

4. SOLUTIONS

Pour assurer une bonne conversion du signal analogique nous avons eu recours au périphérique ADC de la carte de développement STM32F446RE NUCLEO, que nous déclencherons grâce à un timer qui fixera la fréquence d'échantillonnage. Puis nous transmettrons les données résultantes grâce au périphérique UART qui assurera une communication série.

Quant à l'élargissement de la plage de tensions, nous avons opté pour un circuit d'entrée composé d'un pont de diodes permettant de différencier le signal positif du signal négatif et de deux circuits atténuateurs de tensions.

5. APPLICATION EMBARQUEE

Cette partie du projet se base sur trois étapes principales présentées ci-dessous :

5.1. L'acquisition

Pour la capture de signaux analogiques allant de -12V à 12V, on a combiné trois circuits :

❖ *Un pont de diodes :*

Une diode est un composant électronique dipôle dont la polarité est déterminée par une anode (fil positif) et une cathode (fil négatif). Elle est polarisée dans le sens direct si elle permet au courant de circuler sinon elle agit comme isolant [Figure 54].

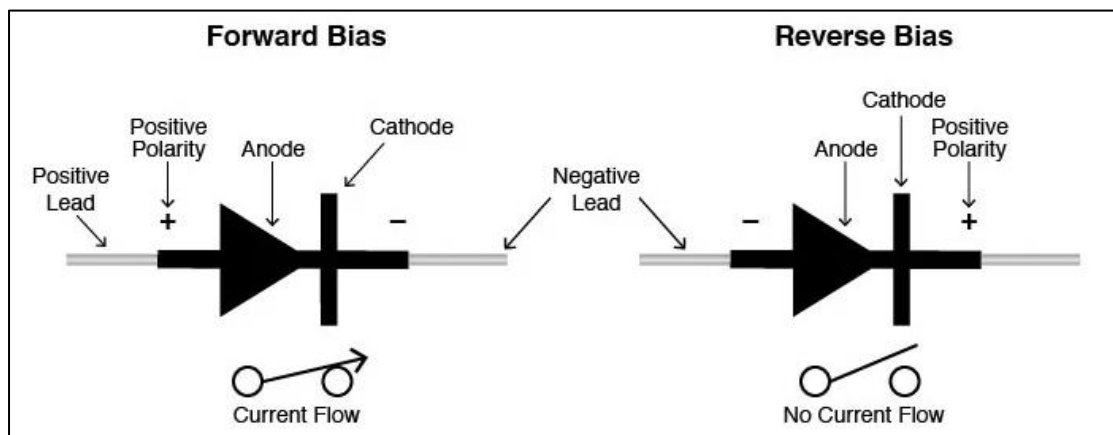


Figure 54:Diodes. [20]

Elle permet entre autres le passage du courant en un sens unique si une tension positive est appliquée à l'anode et le redressement du courant puisqu'elle ne laisse passer que le courant positif [Figure 55].

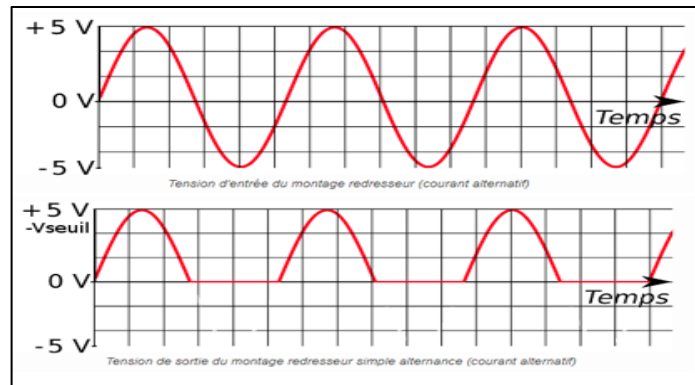


Figure 55: Passage du signal par une diode. [21]

Le pont de diode appelé également pont de Graetz est un circuit redresseur à double alternance composé de quatre diodes montées en pont. Sa structure astucieuse permet au courant en entrée négatif ou positif de passer que dans un seul sens [Figure 56]. Lors de l'alternance positive de la tension d'entrée, seules les deux diodes ayant une tension d'anode supérieure à la tension de cathode sont conductrices. Les deux autres diodes seront bloquées et ne laisseront donc pas passer le courant. Pour l'alternance négative, ce sont les deux autres diodes qui conduisent [Figure 57].

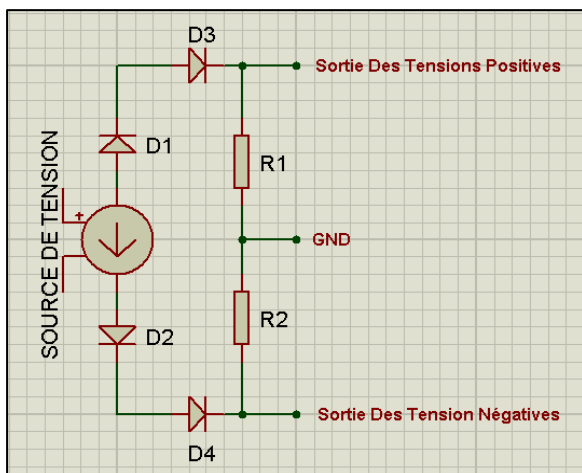


Figure 56 : Le pont de diodes.

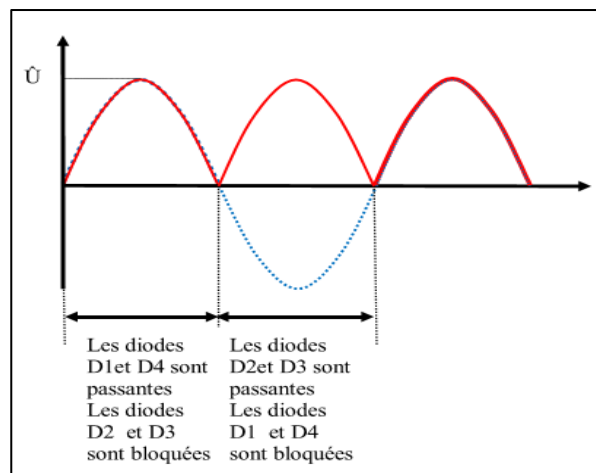


Figure 57: Redressement du signal.

❖ *Deux circuits atténuateurs du signal :*

Une résistance est un conducteur ohmique, qui permet principalement d'opposer une résistance mesurée en ohms (Ω) à la circulation du courant électrique pour ainsi provoquer la diminution de l'intensité du courant. Plus la résistance dans un circuit augmente et plus l'intensité du courant diminue.

La mesure de la valeur d'une résistance peut s'effectuer avec :

- *Un ohmmètre* : En branchant l'ohmmètre directement aux bornes du dipôle entre Ω et COM, en dehors d'un circuit.

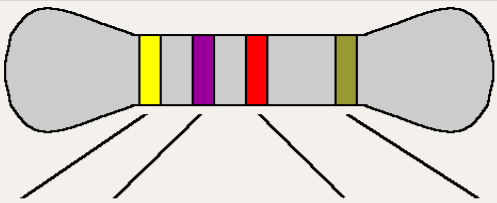
- *Un code de couleurs* : Il est nécessaire d'identifier les couleurs présentes sur la résistance puis l'associer au code universel pour connaître la valeur ohmique de ce composant.

Les deux premières couleurs à gauche représentent les deux premiers chiffres du nombre, la troisième couleur représente le nombre de zéros à ajouter aux 2 chiffres. Par exemple la résistance de la [Figure 58] est de 4700Ω à 0.5% près.

L'association de résistances peut se faire soit :

- En série : Lorsque l'on met bout à bout des résistances, la valeur ohmique de la résistance équivalente, est alors égale à l'addition de toutes les valeurs des résistances. $R_{eq} = R_1 + R_2 + R_3 + \dots + R_n$.
- En parallèle : Lorsque l'on associe plusieurs résistances en parallèle, alors la résistance équivalente se trouve à l'aide de la formule : $1/R_{eq} = (1/R_1) + (1/R_2) + (1/R_3) + \dots + (1/R_n)$

Pour permettre l'abaissement de la tension d'entrée et ainsi protéger le système d'éventuelles surtensions, nous allons utiliser deux circuits montés aux deux sorties du courant du pont de diodes. Chaque circuit est constitué de deux résistances montées en série [Figure 59].



	1 ^{er} anneau gauche 1 ^{er} chiffre	2 ^e anneau gauche 2 ^e chiffre		Dernier anneau gauche Multiplieur	Anneau droite Tolérance
noir	0	0		1	-
marron	1	1		10	1%
rouge	2	2		10^2	2%
orange	3	3		10^3	-
jaune	4	4		10^4	-
vert	5	5		10^5	0.5%
bleu	6	6		10^6	0.25%
violet	7	7		10^7	0.1%
gris	8	8		10^8	0.005%
blanc	9	9		10^9	-
or	-	-		0.1	5%
argent	-	-		0.01	10%

Figure 58: La résistance. [22]

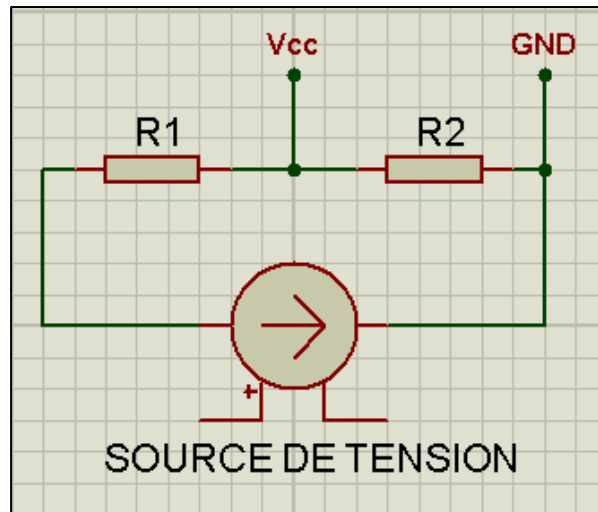


Figure 59: Le circuit atténuateur de tensions.

Dans notre cas de figure, il faut atténuer une tension de 12V à 3.6V puisque c'est la tension maximale supportée par la carte STM32F446RE NUCLEO. Notre circuit assure l'atténuation selon la formule suivante :

$$V_{cc} = \frac{V_{in} \times R_2}{R_1 + R_2}$$

Avec :

- V_{cc} = La tension en sortie du circuit.
- V_{in} = La tension en entrée.
- R_1 et R_2 : des résistances.

En fixant la valeur de R_2 et en sachant que $V_{in} = 12V$ et que $V_{cc} = 3.6V$, on trouvera la valeur de R_1 selon la formule suivante :

$$R_1 = \frac{(V_{in} \times R_2) - (V_{cc} \times R_2)}{V_{cc}}$$

5.2. Le traitement

En l'appliquant à un pont de diodes, un signal peut passer par deux sorties selon son signe. Pour cela il a fallu relier ces deux sorties à deux ADC. L'ADC1 se chargera de la conversion des tensions positives, l'ADC2 de celles des tensions négatives. Et les deux seront déclenchés par le Timer à usage général TIM2.

- Les paramètres de configuration des ADC sont :
 - Clock Prescaler : pour définir le facteur de division de la fréquence de l'horloge. Il sera initialisé à « PCLK2 divided by 4 ».
 - Resolution : pour définir le nombre de bits de la valeur numérique en sortie. Il sera initialisé à 12 bits.
 - External Trigger Conversion Source : il permet de préciser la source du déclenchement qui est « Timer 2 Trigger Out Event »

- External Trigger Conversion Edge : pour préciser le front de la source de déclenchement pour lequel l'ADC réagira. Il sera à « Trigger detection on the rising edge ».
- Sampling Time : pour définir la période d'échantillonnage. Son initialisation est à « 3 cycles ».

Ces paramètres nous permettent de calculer le temps de conversion avec la formule suivante

$$T_{conversion} = \frac{Sampling\ Time + Resolution}{\frac{Clock}{Clock\ Prescaler}}$$

En appliquant cette formule on aura :

$$T_{conversion} = \frac{3+12}{5000000/4} = 1.2 \times 10^{-6} \text{ secondes}$$

Avec ce résultat on aura 8×10^5 échantillons/s

➤ Les paramètres de configuration du Timer sont :

- Prescaler : pour définir le facteur de division de l'horloge du Timer. Son initialisation est à 100.
- Period : pour définir la valeur maximale du compteur du Timer. Il est initialisé à 16.

Ces paramètres permettent de fixer la fréquence du timer qui se calcule avec la formule suivante :

$$Frequency = \frac{Clock}{(period+1)(prescaler+1)}$$

5.3. La transmission

La communication série représente l'une des méthodes les plus utilisées pour la transmission de données binaires numériques entre un ordinateur et un autre périphérique par le biais de connecteurs (ports série). Il existe deux types de transmission série :

- La transmission série synchrone : Un signal d'horloge commun est partagé entre l'émetteur et le récepteur.
- La transmission série asynchrone : L'émetteur et le récepteur disposent chacun de leur propre horloge, le début et la fin de la transmission sont déterminés respectivement par le bit START et le bit STOP.

Cet échange est soumis à différents protocoles assurant ainsi la sécurité et la fiabilité de la transmission série de données. On peut trouver plusieurs interfaces de communications séries, les plus populaires sont : USB, UART, I2C, SPI.

Pour satisfaire notre besoin, nous allons nous servir du périphérique UART de la STM32F446RE NUCLEO. Ses paramètres de configurations sont :

- Baudrate : qui permet de définir la vitesse de transmission qui est mis à 115200bits/s.
- Word length : qui permet de définir la taille maximale du mot à transmettre. Il est mis à 8.

5.4. Architecture de l'application embarquée :

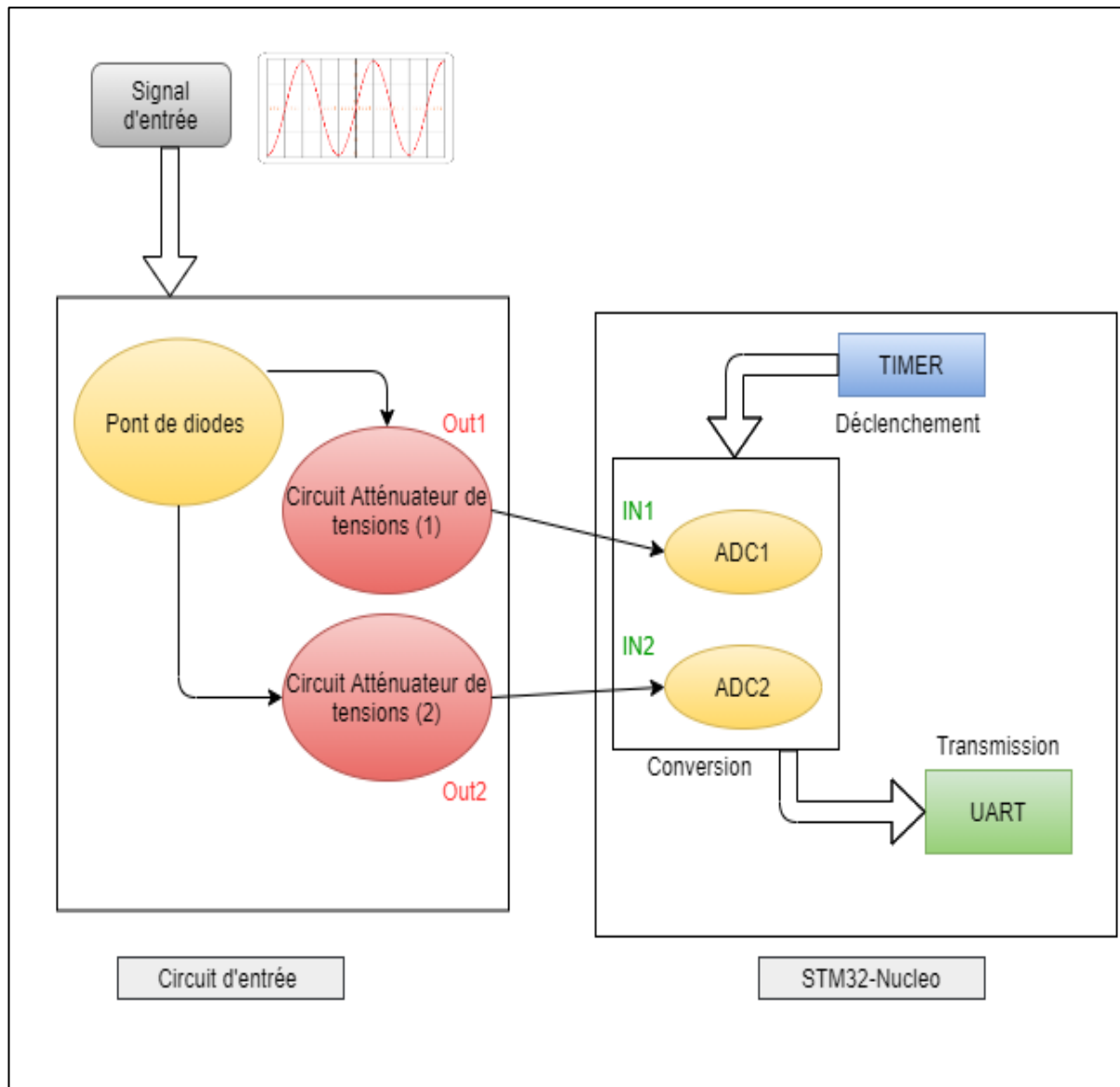


Figure 60: Architecture de l'application embarquée.

5.5. Organigramme de l'application embarqué :

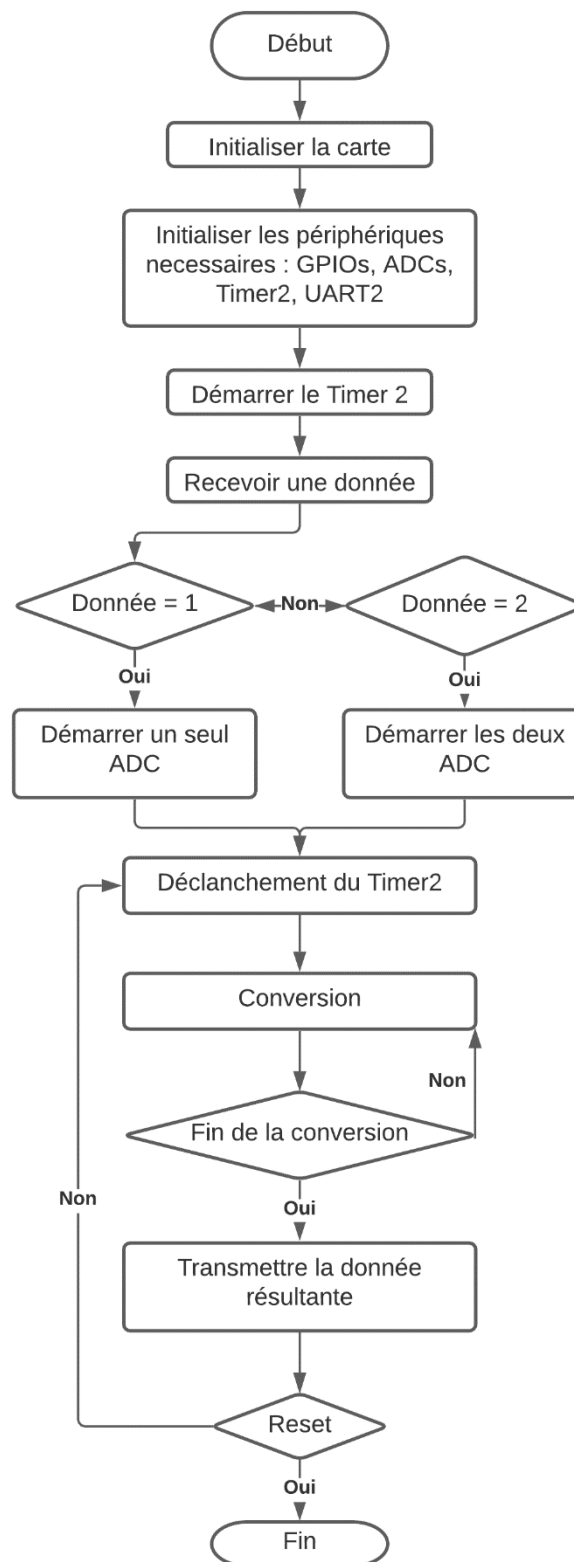


Figure 61: Organigramme de l'application embarqué.

6. L'APPLICATION DESKTOP

6.1. Fonctionnement

Pour pouvoir visualiser les tensions captées, converties et transmises via le port série de la carte STM32F446RE NUCLEO, nous avons conçu une application desktop qui sera développée en Java. Elle représentera l'interface d'interaction avec l'application embarquée.

Cette application nous permettra de configurer selon le besoin les paramètres d'affichage de l'oscillogramme [figure 62] comme dans un oscilloscope ordinaire, à savoir la division du temps et celle de la tension. Elle assurera également la récupération des données transmises à travers le port série de l'ordinateur auquel est connectée le microcontrôleur, pour ensuite les afficher sur un graphe qu'on pourra manipuler avec les paramètres d'affichages cités précédemment pour une meilleure visualisation.

Elle nous permettra aussi d'afficher certains détails des tensions reçus tel que U_{min} qui est la valeur minimale de la tension, U_{max} qui est la valeur maximale de la tension et U_{eff} qui représente la valeur moyenne de la tension alternative calculé avec $U_{eff} = \frac{U_{max}}{\sqrt{2}}$.

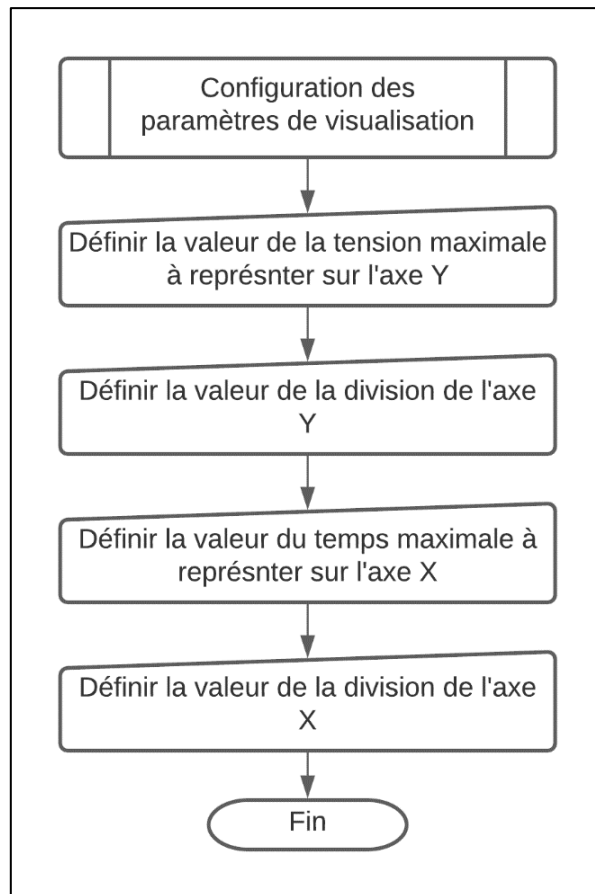


Figure 62: Organigramme de la configuration des paramètres de visualisation.

6.2. Architecture

L'architecture de notre application desktop est présentée comme suit [figure 63] :

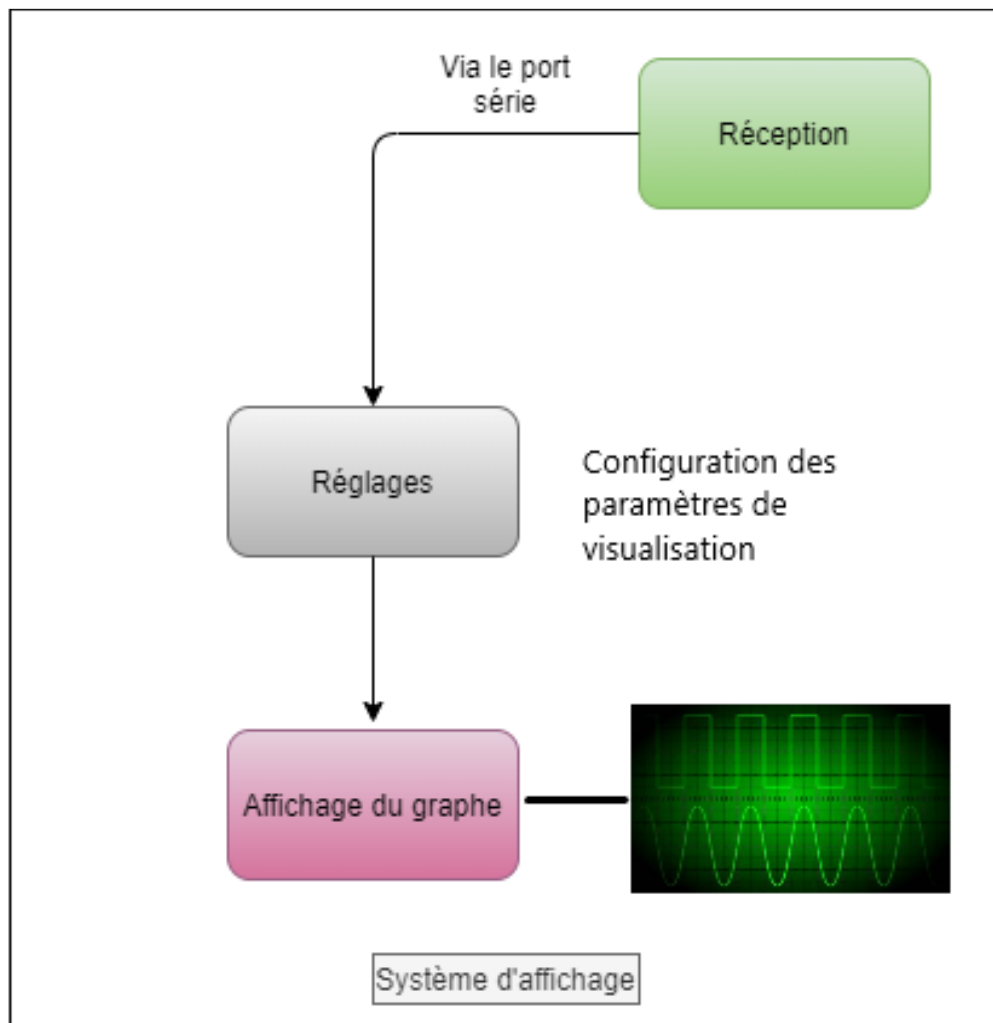


Figure 63: Architecture de l'application desktop.

6.3.Organigramme

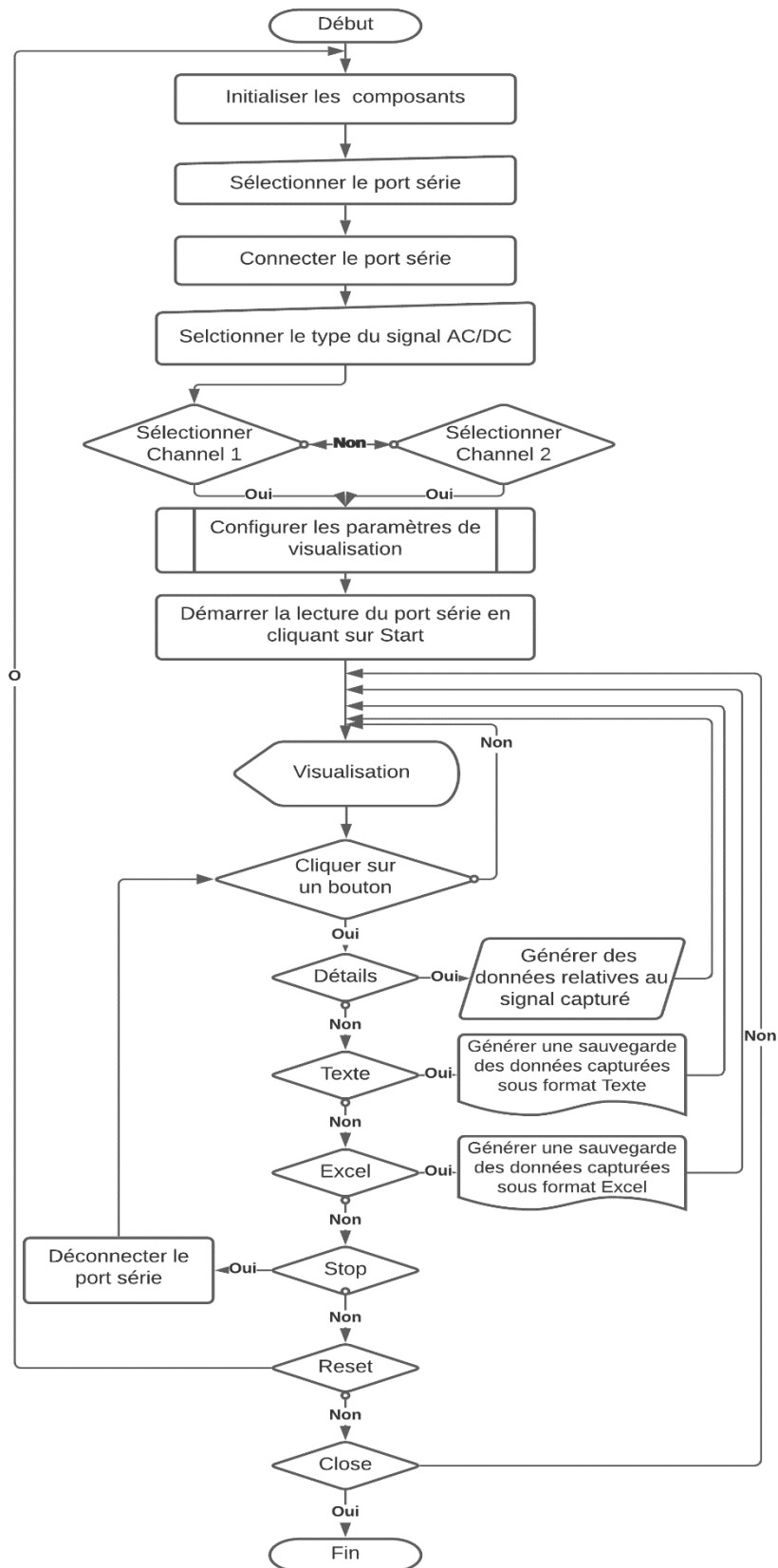


Figure 64:Organigramme de l'application desktop.

7. ARCHITECTURE GENERALE DU SYSTEME

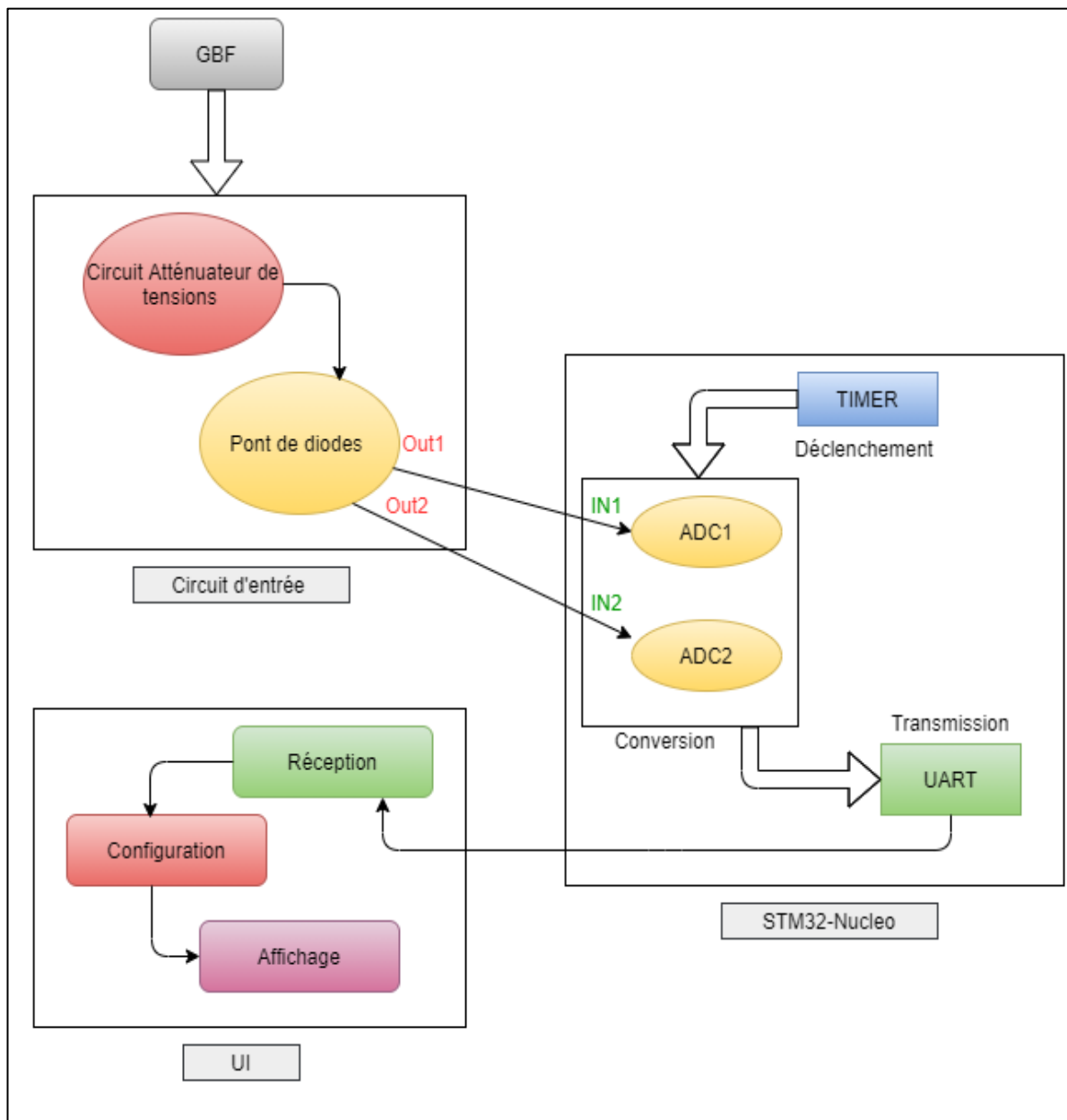


Figure 65: Architecture générale du système.

8. CONCLUSION

Après avoir expliqué en quoi consiste notre travail et après avoir vu chacun de ses aspects en détails nous allons passer dans le chapitre suivant à sa réalisation.

Chapitre 4: REALISATION

1. INTRODUCTION

Dans le chapitre précédent, nous avons exposé la problématique et les contraintes imposées ce qui nous a conduit à proposer des solutions qui y répondent. La prochaine étape consiste en la réalisation, qui permettra de mettre en œuvre tout ce qui a été dit précédemment et développer le projet. Il va de soi que les solutions ne sont admises et considérées comme réalisable que si les résultats des tests confirment leur validité et efficacité.

Nous allons donc au fil de ce chapitre parler des environnements logiciels qui nous ont servis pour le développement, réaliser le circuit du montage de l'oscilloscope et présenter l'interface de l'application desktop avec des captures.

2. OUTILS ET LOGICIELS

2.1. Application embarquée

Le développement d'applications basées sur la carte STM32-Nucleo ne peut s'effectuer sans passer par les deux étapes de configuration et de programmation. Pour la phase de configuration, le software STM32CubeMx est mis à notre disposition afin de générer des fichiers de configuration en langage de programmation C. Cependant, pour la phase de programmation des tâches à exécuter par le microcontrôleur, plusieurs softwares sont disponibles dans le marché comme SW4STM32, True Studio, MDK-ARM, EWARM... Mais dans le cadre notre projet, nous avons opté pour le Système Workbench pour STM32 (SW4STM32), qui est un IDE Eclipse.

2.1.1. STM32CubeMx

STM32Cube est un outil logiciel de conception de microcontrôleurs STM32, disponible gratuitement, cette puissante plate-forme de développement simplifie et accélère les projets des clients.

La plate-forme de développement STM32Cube comprend le configurateur graphique STM32CubeMX, utilisé pour générer des fichiers de configuration pour un microcontrôleur STM32 conformément à la configuration matérielle de la carte choisie, un générateur de code d'initialisation avec le langage C qui constitue un code de départ pour un projet quelconque.

Les principales caractéristiques du STM32CubeMX sont :

- La sélection du μ C STM32.
- La sélection de la carte cible à partir d'une liste des cartes de STMicroelectronics.
- La facilité de la configuration du microcontrôleur (broches, arbre d'horloge, périphériques, middleware).
- La génération du code d'initialisation en langage C.

- La génération de rapports de configuration.
- La génération de projet complet pour l'environnement de développement choisi (SW4STM32).
- Mise à jour automatique de STM32CubeMX et les bibliothèques STM32Cube.

Les principales pages de configuration dans STM32CubeMX sont :

○ Page d'accueil de STM32CubeMX :

La page d'accueil est la première fenêtre qui s'ouvre au lancement du programme STM32CubeMX. Elle reste ouverte aussi longtemps que la demande est en cours d'exécution.

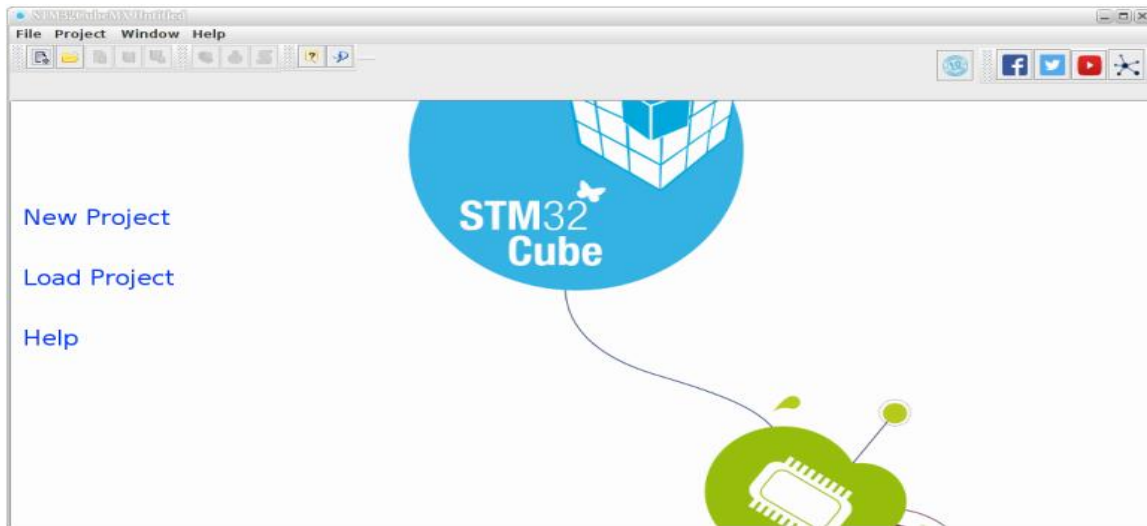


Figure 66: L'INTERFACE PRINCIPALE DE STM32CUBEMX.

○ La fenêtre de nouveau projet :

Cette fenêtre affiche deux onglets au choix :

- L'onglet MCU Selector proposant une liste de processeurs cible, utilisé dans le cas où la carte n'a pas été développée par ST.

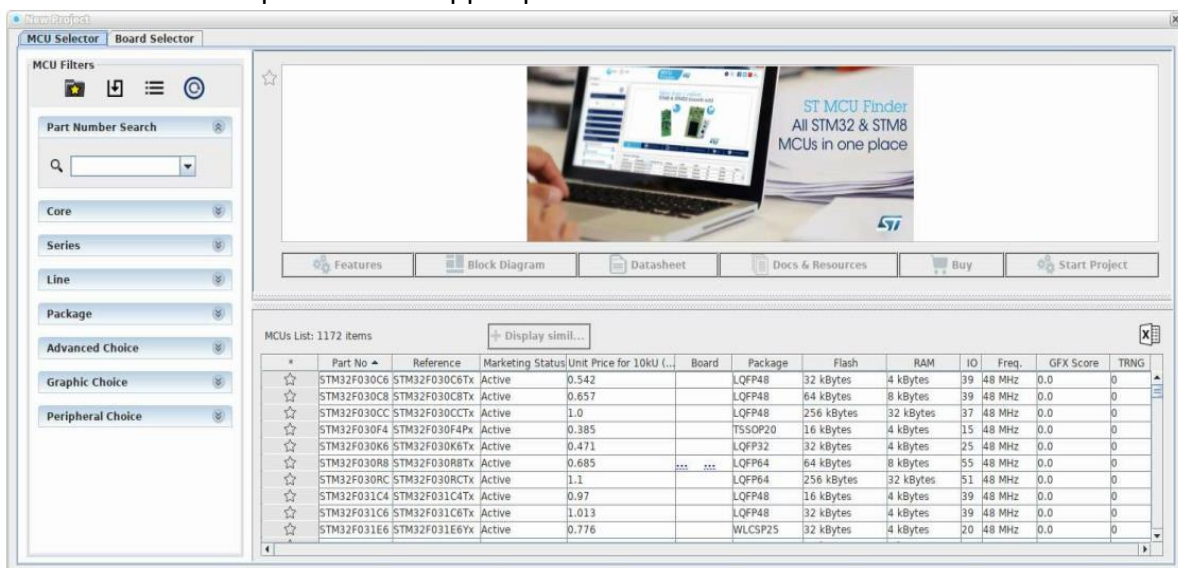


Figure 67: Onglet MCU Selector de la fenêtre New Project.

- L'onglet Board Selector est utilisé dans le cas où il s'agit d'une carte développée par ST, il permet de filtrer suivant le type de carte. L'outil propose pour chaque carte sa configuration par défaut.

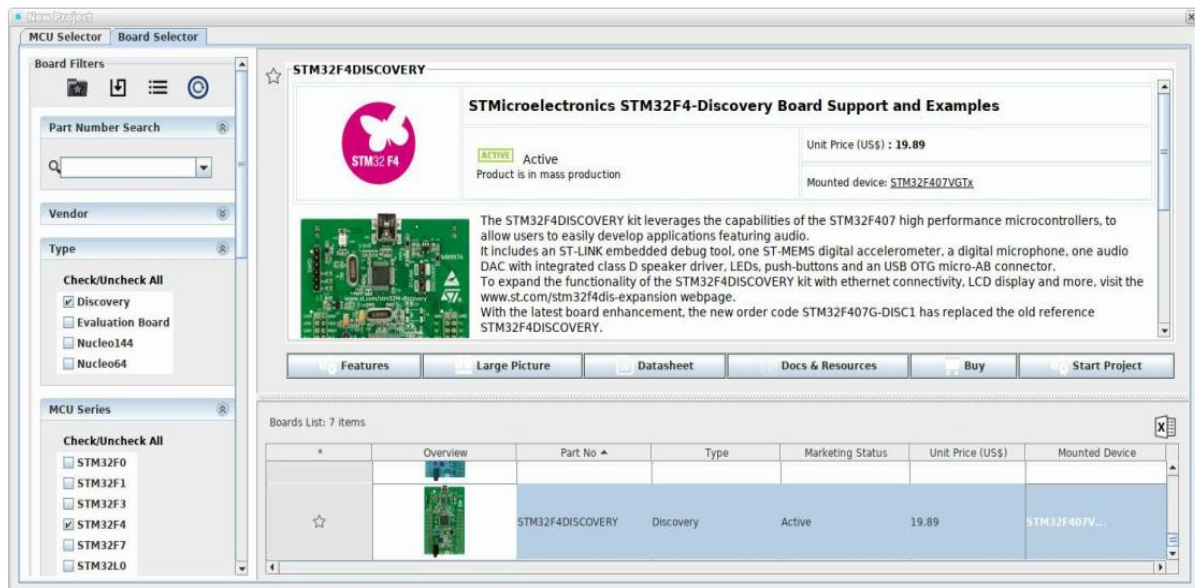


Figure 68: Onglet Board Selector de la Fenêtre New Project.

Après la sélection du μC ou d'une carte, une nouvelle fenêtre s'ouvre. Comme montré dans la figure 69.

○ La fenêtre principale :

La fenêtre principale affiche tous les composants et les menus de la STM32CubeMX (Pinout, Clock Configuration, Configuration, Power Consumption Calculator).

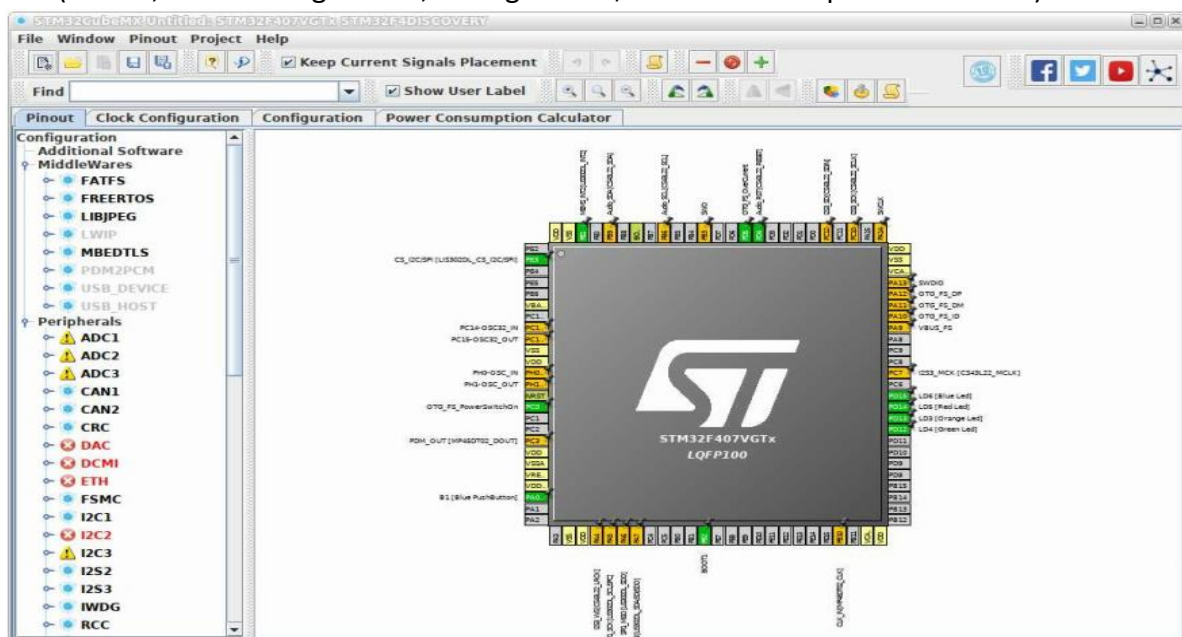


Figure 69: Fenêtre Principale.

Cette plateforme de développement nous permet de configurer selon nos besoins les paramètres de configuration de l'ADC, du TIMER et de l'UART.

En ce qui concerne l'ADC nous pouvons spécifier [figure 70] :

- La résolution de l'ADC : 12, 10, 8 ou 6bits.
- Le diviseur d'horloge : 4, 6 ou 8.
- L'alignement de données : gauche ou droite.
- Le mode de conversion : continue, multiple, discontinue...
- Conversion avec ou sans DMA.
- Nombre de conversion à la fois.
- La source de déclenchement.

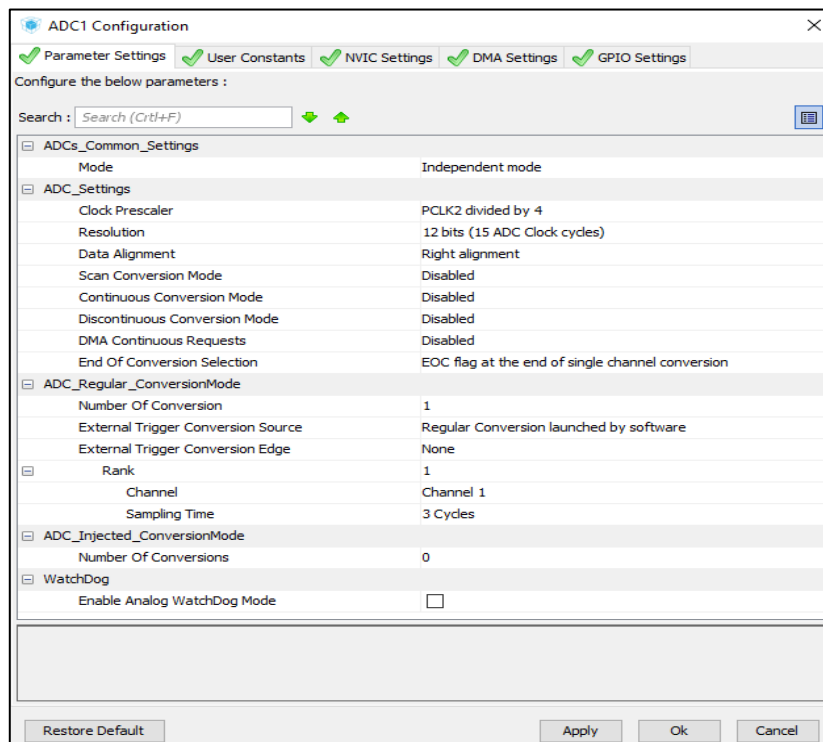


Figure 70: Configuration de l'ADC.

Pour les configurations du TIMER il suffit de préciser [figure 72] :

- Le prescaler et la période.

Et enfin pour l'UART il faut préciser [figure 71] :

- Le baudrate : 9600, 19600, 115200...
- La taille du mot à transmettre.
- La parité.
- Le Bits STOP.

Quand les configurations nécessaires ont été effectuées, il ne reste plus qu'à générer le code du projet créé dans l'IDE sélectionné (par exemple SW4STM32). [16]

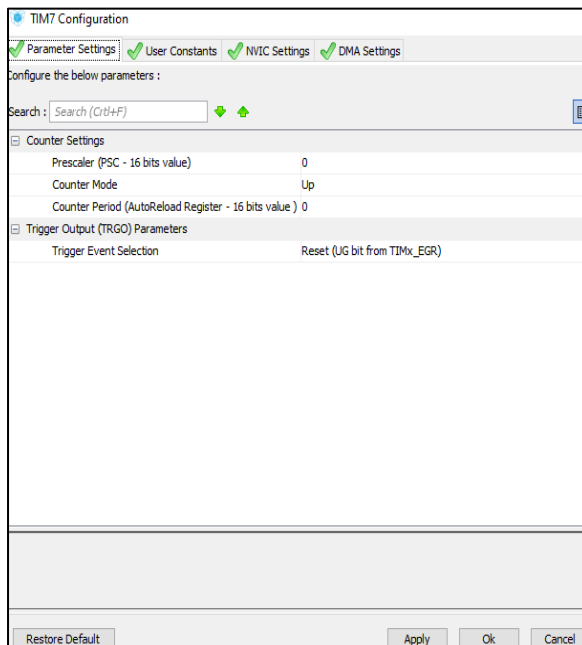


Figure 72: Configuration Timer.

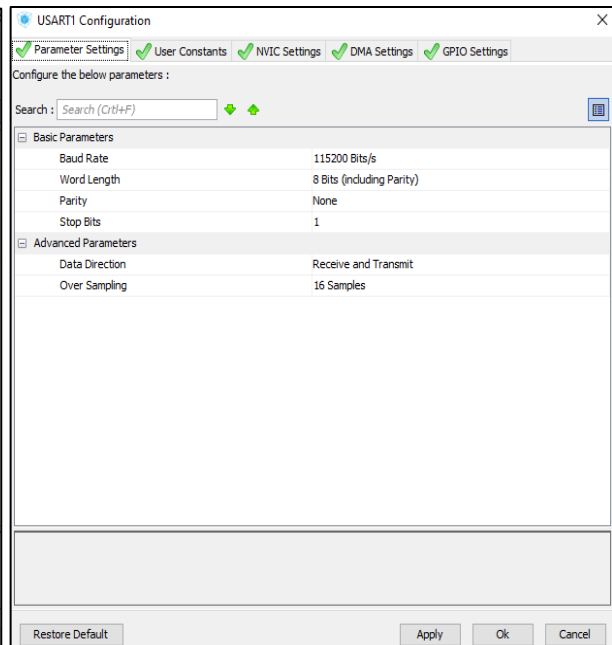


Figure 71: configuration UART.

2.1.2. SW4STM32

System Workbench for STM32 (SW4STM32), est un IDE Eclipse, qui fournit une plateforme de développement logiciel pour les cartes STM32.

L'IDE aide à créer rapidement un projet C embarqué pour une carte cible donnée. Il intègre un éditeur de code complet, des outils de compilation (compilateur, assembleur, éditeur de liens ...) et des outils de débogage à distance.

Fonctionnalités de SW4STM32 :

- Base de données et bibliothèques pour les cartes et les μ C STM32.
- Éditeur de code source.
- Générateur de scripts pour l'éditeur des liens.
- Outils de construction (compilateur croisé, assembleur et éditeur de liens basés sur GCC) pour générer le fichier binaire.
- Outils de débogage (OpenOCD, GDB).
- Outils de programmation (flashage). [16]

2.1.3. STM32 ST-LINK

L'utilitaire STM32 ST-Link est une interface logicielle complète pour la programmation des microcontrôleurs STM32. Il fournit un environnement facile à utiliser et efficace pour la lecture, écriture et vérification des périphériques mémoires.

Caractéristiques :

- Logiciel gratuit.
- Chargement, modification, enregistrement des fichiers exécutables et de données générées par l'assembleur, éditeurs de liens ou les compilateurs C.
- Effacement, programmation, affichage et vérification des mémoires.
- Programmation de la mémoire effectué une seule fois.
- Offre une interface de ligne de commande.

- Comparaison de fichier avec la mémoire cible.
- Prend en charge la vue de l'état de la mémoire dans la mode de mise à jour direct.
- Mise à jour du firmware ST-Link. [16]

2.1.4. Générateur de fonction

Pour des raisons de tests nous avons eu recours au générateur de basse fréquence (GBF) appelé aussi générateur de fonction, qui est une source de tension dont la fréquence et l'amplitude sont réglable. Il permet principalement de générer un signal sous forme de sinusoïdes, de créneaux, ou bien de triangles qui sera ainsi observé dans notre l'oscilloscope.

2.2. Application desktop

L'interface graphique, qui nous sert de dispositif d'affichage pour notre oscilloscope, est réalisée avec JavaFX Scene builder pour ensuite être généré avec java dans Eclipse Quant à sa communication avec l'application embarquée elle a été rendue possible grâce à la bibliothèque Java Simple Serial Connector.

2.2.1. Eclipse

Eclipse est une plate-forme de développement d'application Java développé par IBM, il est créé sous forme d'un IDE et est constitué d'un ensemble de Framework d'applications, de logiciels, d'outils open-sources.

Il est principalement connu pour ses plugins qui permettent aux développeurs de tester du code écrit dans d'autres langages de programmation comme C, C++, PHP, etc.

Il représente plusieurs caractéristiques :

- Gratuit et open source.
- Disponible pour la plupart des systèmes d'exploitation.
- Ergonomique.
- Facile à prendre en main.
- Ouverture de son noyau, ce qui permet l'ajout de plugins comme des éditeurs XML, HTML, JSP...
- Prend en charge plusieurs outils tel que : Ant, SVN, JUnit...
- Sa distribution est sous forme de bundles, qui contiennent des plugins préconfigurés comme eclipse jee pour le développement Java EE ou encore eclipse -sdk pour le développement de plugins.
- Disponibilité de plusieurs bibliothèques comme JavaFX, JSSC... [22]

2.2.2. JavaFx

Pour concevoir des interfaces graphiques ergonomiques nous avons fait appel à JavaFX, qui est une bibliothèque Java utilisée pour créer des applications enrichies. Les applications écrites en utilisant cette bibliothèque peuvent fonctionner de manière cohérente sur plusieurs plates-formes, et sur divers périphériques.

Pour développer des interfaces graphiques en utilisant le langage Java, les programmeurs s'appuient sur des bibliothèques telles qu'Advanced Windowing Toolkit et Swings. Mais après l'avènement de JavaFX, ils peuvent maintenant développer des applications graphiques

sophistiquées avec un contenu riche, cela grâce à l'ensemble d'API que cette bibliothèque fournit pour la gestion des fenêtres graphiques. [24]

2.2.3. Scene Builder

JavaFX Scene builder est un outil de mise en page visuelle qui permet aux utilisateurs de concevoir facilement et rapidement des interfaces utilisateurs.

Il permet tout simplement de faire glisser et déposer des composants dans un layout, modifier leurs propriétés, appliquer des feuilles de styles CSS et ainsi générer automatiquement le code FXML correspondant dans un fichier FXML qui sera ensuite combiné avec un projet java.

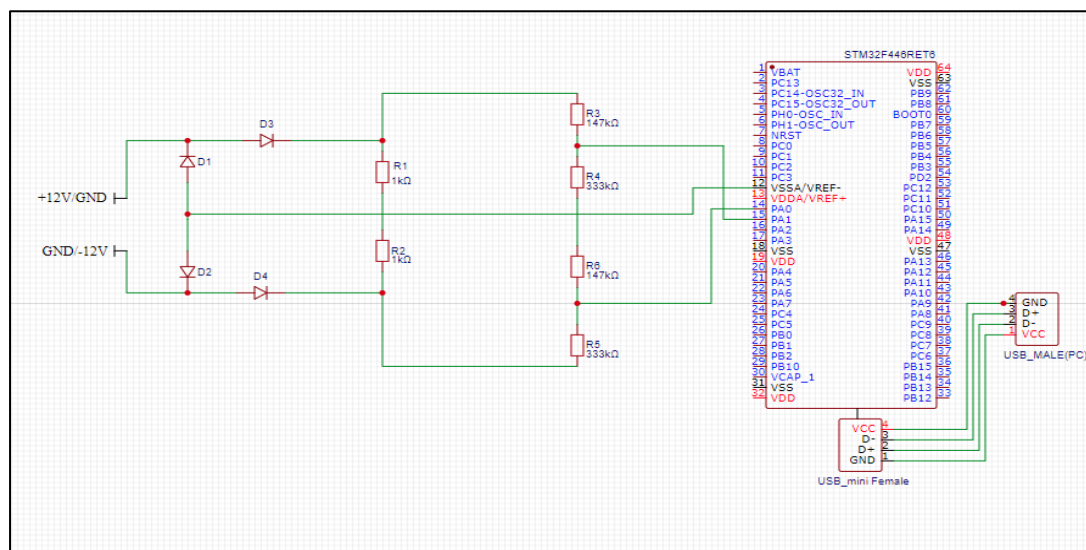
Fonctionnalités :

- Outil de mise en page de l'interface utilisateur (UI Layout Tool) : met à la disposition de l'utilisateur des charts, contrôleurs, Containers, Shapes...
- FXML Visual Edition : génération d'un fichier FXML écrit avec le langage de balise XML, permet de développer l'interface Utilisateur UI indépendamment de l'application.
- Il peut être associé à n'importe quel IDE Java, également à NetBeans IDE.
- Disponible dans les systèmes d'exploitation : Windows, Linux, Mac et OS X.
- Possibilité de lui appliquer des feuilles de styles CSS ce qui rend les interfaces plus modernes et sophistiqués. [25]

2.2.4. JSSC

Java-Simple-Serial-Connector est une bibliothèque Java autonome qui permet de communiquer avec les ports série et par conséquent avec les périphériques reliés à ces ports. Elle est considérée comme le remplacement de la bibliothèque RxTx qui n'est plus en développement actif. JSSC est installable dans les IDE JAVA comme eclipse, Netbeans... Et est également supportée dans les systèmes d'exploitation Windows, Linux, Mac, Solaris...

3. SCHEMA ELECTRONIQUE :



4. MAQUETTE :

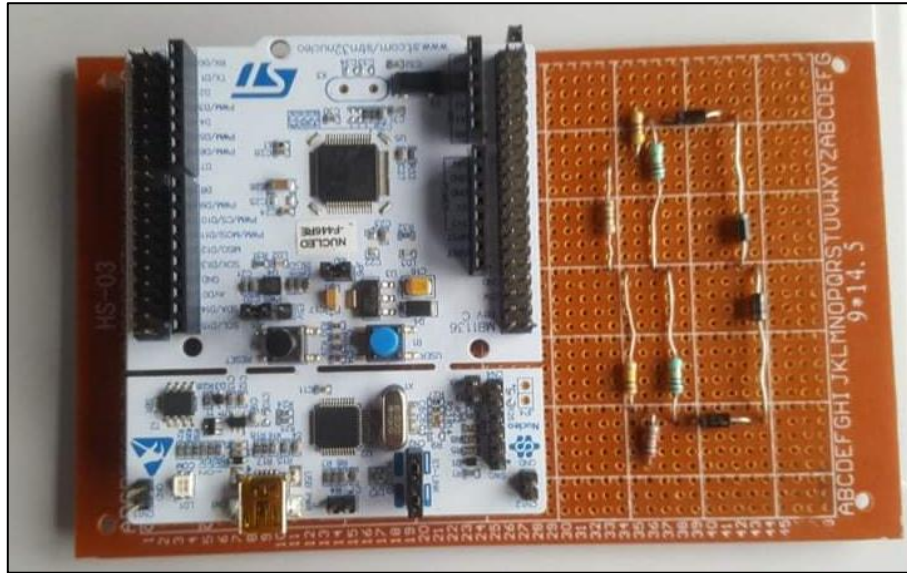


Figure 74:Maquette du projet.

5. LES CAPTURES D'ECRAN L'APPLICATION DESKTOP

L'interface de l'application desktop se constitue de deux parties [Figure 75] :

- 1) Une partie réglages : qui permet d'effectuer les réglages nécessaires sur le graphe.
- 2) Une partie affichage : qui représente l'oscillogramme de l'oscilloscope.

Channel 1 [Figure 76] nous permet d'effectuer des réglages sur la sensibilité verticale du graphe 2) de la [Figure 75], on y trouve les éléments suivants :

- 3) Menu déroulant Port : sélectionner le port auquel la carte est connectée.
- 4) Radio Bouton AC/DC : choisir de filtrer ou non le signal d'entrée selon ses composantes.
- 5) Spinner Max Voltage : préciser la valeur maximale de la tension.
- 6) Spinner Division : manipuler les divisions verticales de l'oscillogramme.
- 7) Spinner YPOS : déplacer le graphe verticalement.
- 8) Bouton Details : permet d'afficher dans [figure 78] les détails concernant la tension.
- 9) Bouton Excel : fait une sauvegarde des données reçues dans un fichier Excel.
- 10) Bouton Word : fait une sauvegarde des données reçues dans un fichier Word.
- 11) Bouton Start : permet de lancer la réception des tensions via port série.
- 12) Bouton Stop : permet de déconnecter le port série et ainsi arrêter la réception.
- 13) Bouton Reset : remise à l'état initiale de tous les composants de l'interface.

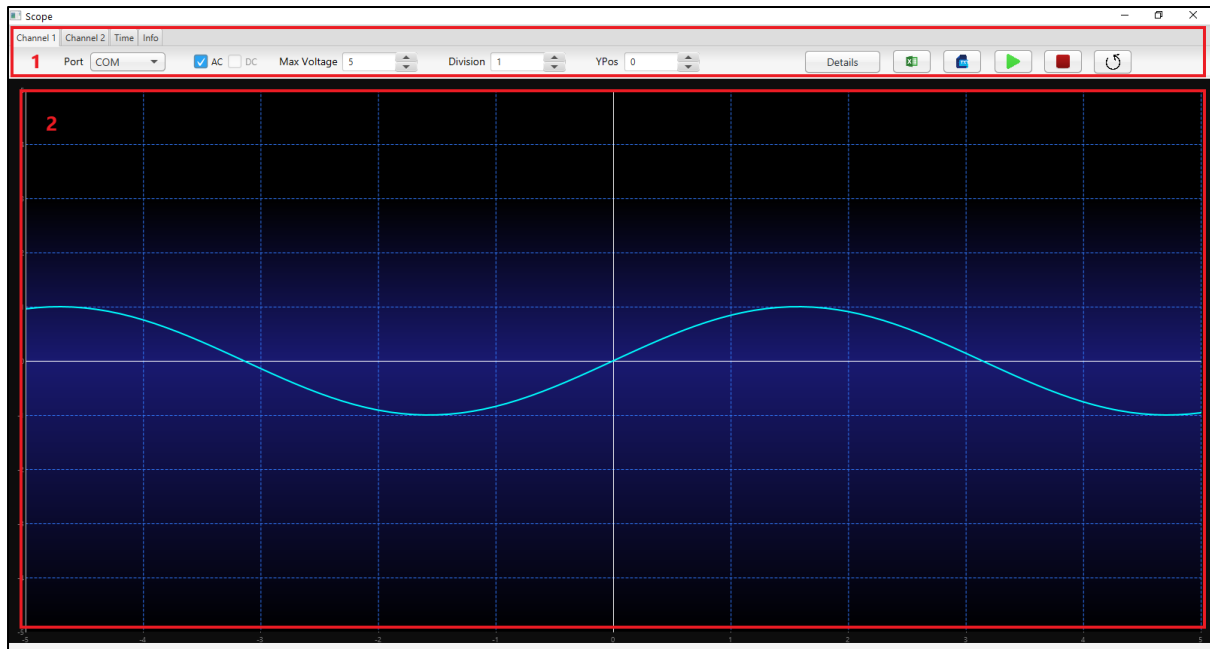


Figure 75:page d'accueil de l'interface.

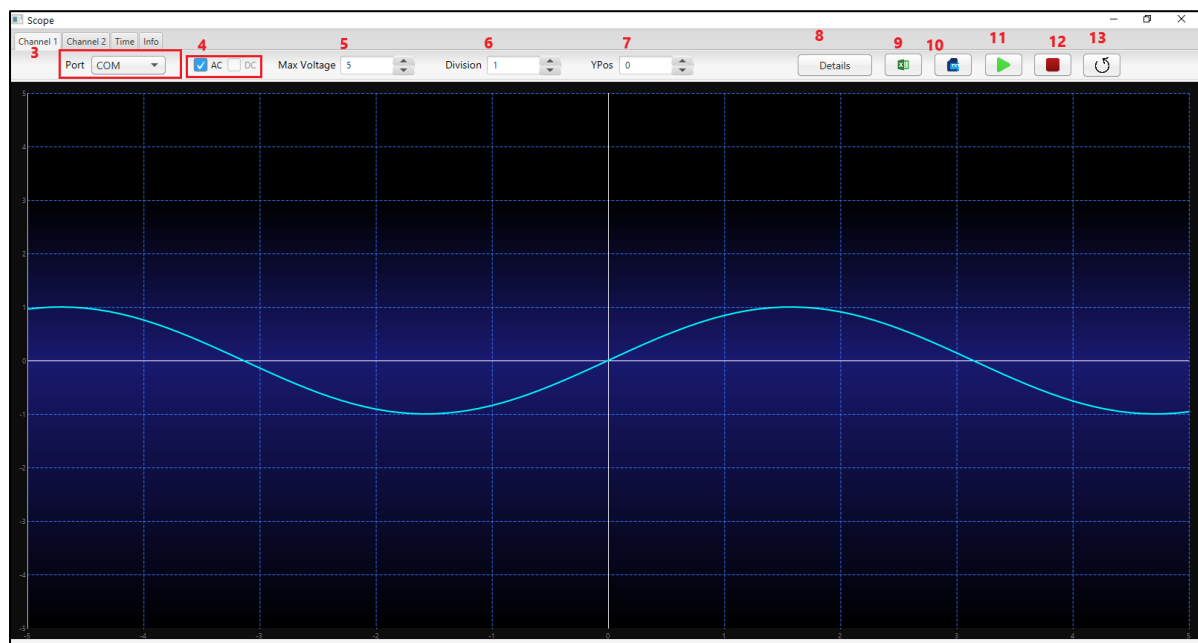


Figure 76: Channel 1.

Après avoir terminé avec Channel 1 on passe à Time [Figure 77] où on pourra effectuer les réglages concernant la sensibilité horizontale de l'oscillogramme 2) [Figure 75] :

- 14) Spinner Max Time : préciser le temps maximum.
- 15) Spinner Division : manipuler les divisions horizontales de l'oscillogramme.
- 16) Spinner Xpos : Déplacer le graphe horizontalement.

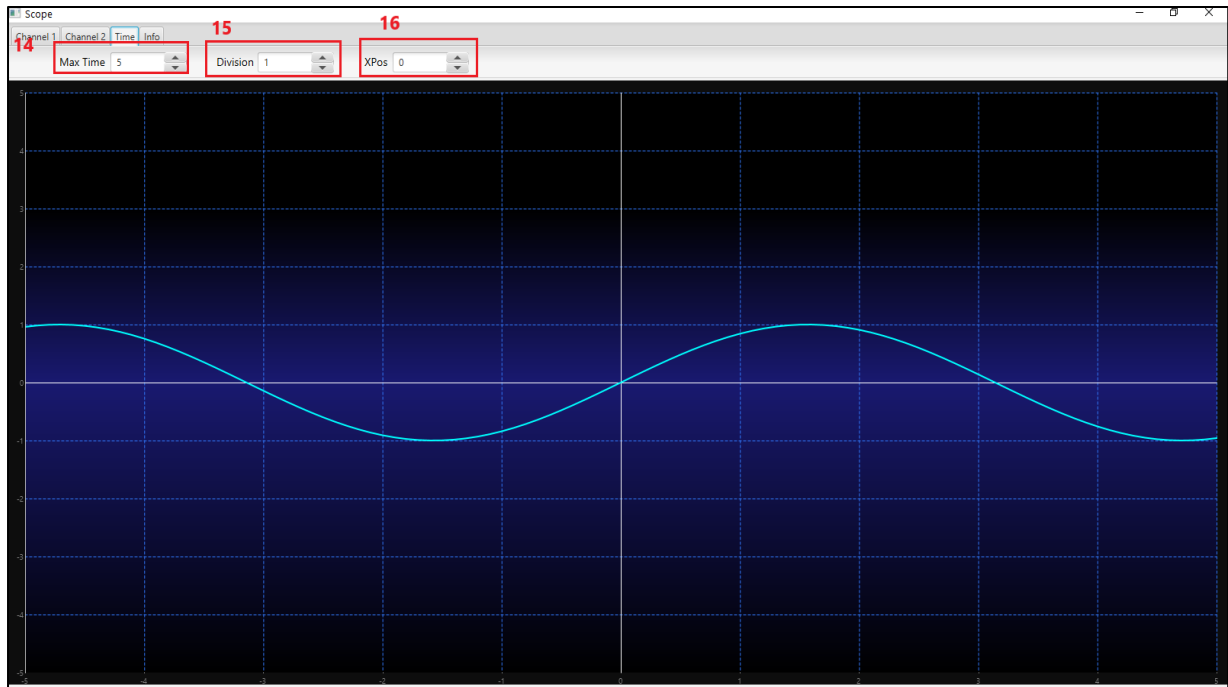


Figure 77 : TIME.

Si on veut afficher des détails sur les tensions du signal reçues, on clique sur le bouton Details (8) [figure 76], ainsi on aura dans [figure 78] les informations nécessaires tel que :

- Umax : la valeur de la tension maximale
- Umin : la valeur de la tension minimale.
- Ueff : la valeur efficace de la tension alternative.
- Umoy : la valeur moyenne de la tension.

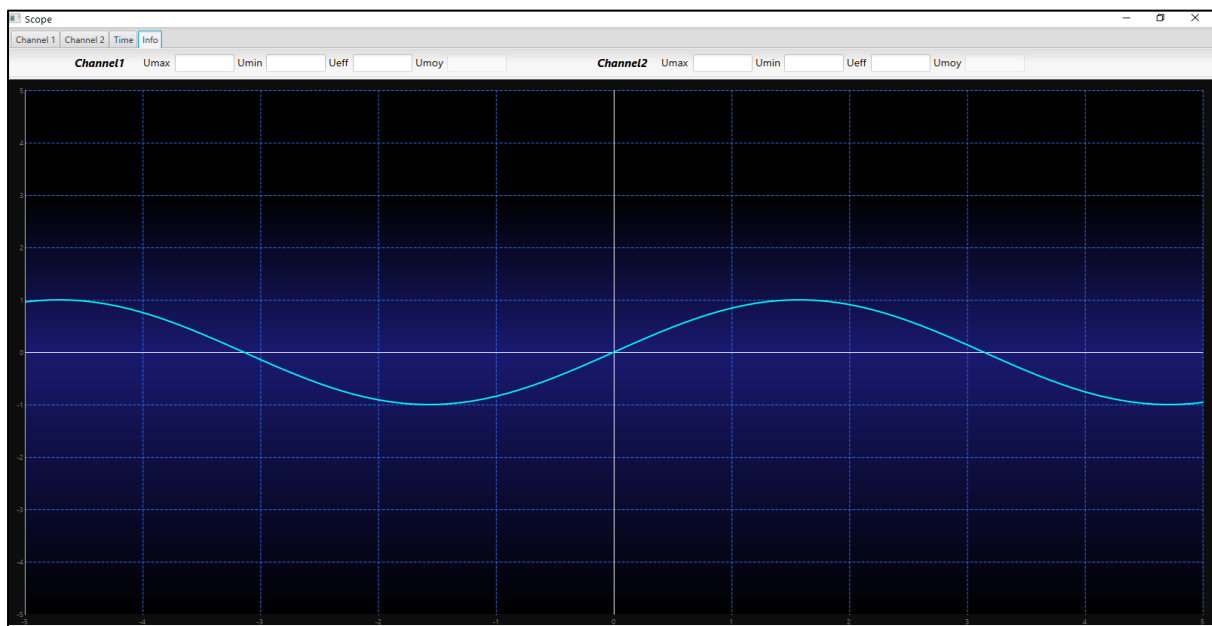


Figure 78: Informations.

6. CONCLUSION

Dans ce chapitre on a présenté les différents outils et logiciel de développement, nous avons réalisé notre système tout en respectant les objectifs fixé au départ. Et nous avons également présenté l'interface graphique de l'oscilloscope avec des captures.

CONCLUSION GENERALE

En guise de projet de fin du cycle Master spécialité Réseaux, Mobilité et Systèmes Embarqués, nous avons travaillé sur la conception et la réalisation d'un oscilloscope numérique à base d'un microcontrôleur STM32.

Pour assurer au mieux les fonctionnalités d'un oscilloscope nous avons configuré deux ADC de la carte STM32F446RE NUCLEO, ADC1 ET ADC2, de sorte à ce qu'ils soient déclenchés par un timer. Nous avons eu recours à deux ADC pour pouvoir convertir à la fois des signaux à courant continu et des signaux à courant alternatif. En effet si nous choisissons de visualiser un signal DC on n'activera que l'ADC1 puisqu'il n'a que des composantes positives. Quant au signal alternatif, il nécessitera les deux ADC puisqu'il passera par un pont de diodes qui séparera la composante positive de la composante négatives qui prendront donc deux sorties différentes. Après conversion, les données résultantes seront stockées dans un tampon en attendant d'être transmises via le port série USART2. Une fois la transmission effectuée, l'application desktop intervient pour recevoir les données via le port série de l'ordinateur auquel est connectée la carte de développement pour ensuite les représenter sur un graphe qu'on nomme « Oscillogramme ».

En élaborant ce travail nous avons eu l'opportunité de mettre en œuvre les connaissances acquises durant nos 5 années d'études concernant les systèmes embarqués, la programmation orientée objet (Java, C), Les IHMs, ... etc. Et comme à tout achèvement de projet nous avons pu développer nos compétences, à la fois pédagogiques et personnelles, puisque :

- Nous avons découvert ce qu'est le traitement du signal et toutes les techniques qu'il englobe.
- Nous nous sommes mieux familiarisés avec les microcontrôleurs STM32.
- Nous avons acquis et mis en pratique certaines notions de base de l'électricité et de l'électrotechnique.
- Nous avons eu une première idée du monde professionnel, puisque nous avons pris en charge un projet et nous avons mené sa conception et sa réalisation à bout.
- Nous avons su bien gérer le travail de groupe.
- Nous avons su réagir aux conditions de travail inhabituelles imposées par la situation sanitaire due au COVID19.

Pour conclure, nous espérons que notre travail soit à la hauteur des attentes, et qu'il sera utile à toute personne ayant besoin d'un oscilloscope numérique.

Toutefois des perspectives d'améliorations de notre travail restent toujours envisageable, nous évoquons ci-dessous des points de perspectives futures de l'application :

- Rajouter un autre canal (Channel 2), pour ainsi utiliser les 2 canaux à la fois (dual).
- Utiliser un écran tactile comme interface d'affichage.
- Augmenter les performances tel que la fréquence d'échantillonnage de l'ADC.
- Sauvegarder l'oscillogramme tel qu'il est.

REFERENCES

- [1] <http://nalya.canalblog.com/archives/2008/01/09/7499662.html>
- [2] https://sti.discip.ac-caen.fr/IMG/pdf/traitement_signal.pdf
- [3] http://www.gipsa-lab.grenoble-inp.fr/~christian.jutten/mescours/Cours_Theorie_Signal_2018.pdf
- [4] https://www.youtube.com/watch?v=5ppzxUuh_rQ
- [5] Francis Cottet, Aide-Mémoire Traitement du signal, Dunod 2000.
- [6] https://moodle.insa-rouen.fr/pluginfile.php/6090/mod_resource/content/0/cours1.pdf
- [7] https://www.emse.fr/~dutertre/documents/cours_convertisseurs.pdf
- [8] <https://www.youtube.com/watch?v=wmAgUqIPvXY&t=435s>
<https://www.youtube.com/watch?v=547Tjxiio20&t=6s>
- [9] <https://www.youtube.com/watch?v=y7EcXIhNMEM&list=PLnFWWhiyw5bCu5ndPAIYZjSwh5KIO6OlzZ&index=3>
<https://www.youtube.com/watch?v=ugdmpmj8Bpl&list=PLnFWWhiyw5bCu5ndPAIYZjSwh5KIO6OlzZ&index=4>
- [10] http://exam2ham.free.fr/donnees/oscillo/emission_spot.html
- [11] https://cdn.rohdeschwarz.com/fr/general_37/local_webpages/Fondamentaux_dun_oscilloscopell.pdf
- [12] http://perso.citi.insa-lyon.fr/trisset/cours/rts12/slides/cours_RTS12_intro.pdf
- [13] <https://mouridsmida.files.wordpress.com/2014/03/chapitre-1-2p.pdf>
- [14] <https://www.les-electroniciens.com/sites/default/files/cours/hcs12.pdf>
- [15] <https://www.esen.tn/portail/medias/documents/enseignement/1488739733495.pdf>
- [16] <https://www.st.com/>
- [17] <https://community.st.com/s/>
- [18] STM32 Reference Manual, RM0090.
- [19] <https://www.amazon.fr/Digital-Signal-Processing-Cortex-M-Microcontrollers/dp/1911531166>
- [20] <https://www.fluke.com/fr/apprendre/meilleures-pratiques/bases-des-mesures/electricite/qu-est-ce-qu-une-diode>
- [21] <http://www.elektronique.fr/cours/composant-diode.php>
- [22] <http://www.elektronique.fr/cours/resistance/resistance.php>

- [23] <https://www.techno-science.net/definition/517.html>
- [24] <https://openjfx.io/>
- [25] <https://www.oracle.com/java/technologies/javase/javafxscenebuilder-info.html>