Implantations matérielles de réseaux de neurones à base de nanotechnologies pour le calcul événementiel

Pierre Lewden

université de BORDEAUX

THÈSE PRÉSENTÉE

POUR OBTENIR LE GRADE DE

DOCTEUR DE

L'UNIVERSITÉ DE BORDEAUX

ÉCOLE DOCTORALE DES SCIENCES PHYSIQUES ET DE L'INGÉNIEUR

SPÉCIALITÉ : ÉLECTRONIQUE

Par Pierre LEWDEN

Implantations matérielles de réseaux de neurones à base de nanotechnologies pour le calcul événementiel

Sous la direction de : Sylvain SAÏGHI

Soutenue le 04/07/2023

Membres du jury :

M. SAÏGHI, Sylvain	Professeur des universités – Université de Bordeaux	Directeur de thèse
M. PORTAL, Jean-Michel	Professeur des universités – Aix-Marseille Université	Rapporteur
M. QUERLIOZ, Damien	Chargé de recherche – Université Paris-Saclay	Rapporteur
Mme. VATAJELU, Elena-loana	Chargée de recherche – Université Grenobles Alpes	Examinatrice
M. BÉGUERET, Jean-Baptiste	Professeur des universités – Université de Bordeaux	Président du jury

Membre invité :

M.VINCENT, Adrien F.	Maître de conférences – Bordeaux INP	Co-encadrant

Thèse effectuée au sein du
laboratoire de l'**Intégration du Matériau au Système**
de l'Université de Bordeaux.
UMR 5218
Bâtiment A31,
351 Cours de la Libération,
33405 TALENCE cedex,
FRANCE.

À ma mère qui a rejoint les étoiles peu de temps avant la fin de l'écriture de ce manuscrit…
À mon père, à mes sœurs.

Résumé

Cette thèse a eu lieu au sein de l'axe thématique « IA matérielle » du groupe Bioélectronique du laboratoire IMS, dans le cadre du projet de recherche européen ULPEC. Ce projet a eu pour ambition de créer un capteur neuromorphique intégré, intelligent et à basse consommation utilisant un réseau de neurones impulsionnels matériel en combinant des technologies innovantes telles qu'une rétine artificielle (caméra événementielle), des neurones en silicium et des memristors ferroélectriques comme synapse dans une même puce. Dans le cadre de ce projet, le travail réalisé durant cette thèse a consisté en l'étude d'architectures de réseaux de neurones impulsionnels reposant sur des synapses memristives et des neurones en silicium. Les architectures étudiées par la simulation prennent en compte les contraintes physiques et électriques propres à une réalisation matérielle des neurones événementiels. En complément de ce travail exploratoire de simulations informatiques, le développement d'une plateforme dédiée pour la mise en œuvre de matrices de nanocomposants memristifs réalisées par des partenaires extérieurs a été accompli. Ces matrices de 81 × 10 nanocomposants memristifs ont pour vocation d'être utilisées soit pour réaliser des réseaux de neurones événementiels *all-to-all*, soit pour la caractérisation expérimentale de ces 810 composants afin d'affiner les simulations réalisées. Si les simulations ont permis de définir l'architecture du démonstrateur du projet européen ULPEC ainsi que de nouvelles règles d'apprentissage, la plateforme 81 × 10 a été développée et vérifiée expérimentalement et la démonstration d'apprentissage sur des matrices de composants memristifs est en phase de développement.

Mots-clé :
Réseau de neurones matériels.
Memristor.
Calcul événementiel.

Hardware implementation of event-based neural networks based on nanotechnologies

Abstract

This thesis took place within the thematic cluster "hardware AI" of the Bioelectronics Group of the IMS Laboratory during the ULPEC European Research Project. The aim of this project was to create an integrated, intelligent, low-powered neuromorphic sensor using a hardware event-based network by combining innovative technologies such as an artificial retina (event-based camera), silicon neurons and ferroelectric memristors as synapses into one chip. The work carried out in this project involved the study of architectures of event-based neural networks using memristive synapses and silicon neurons. The architectures studied by simulation means take into account the physical and electrical constraints of a hardware implantation of event-based neurons. In addition to this exploratory work with computer simulations, the development of a dedicated platform for the use of a crossbar array of memristive nanodevices made by external partners has been achieved. Those 81 × 10 crossbar arrays of memristive nanodevices are intended to be used either for an event-based all-to-all neural network or for experimentally characterizing those 810 devices to refine the simulations performed. While the simulations have provided a basis for defining the architecture of the European ULPEC project as well as new learning rules, the 81 × 10 platform has been developed and experimentally verified, but has not yet produced a learning demonstration using a memristive nanodevice crossbar array as it is still under development.

Keywords :
Hardware neural networks.
Memristor.
Event-based computation.

Notes préliminaires

Unités utilisées dans ce manuscrit :

Temps	Seconde	s
Fréquence	Hertz	Hz
Distance	Mètre	m
Tension	Volt	V
Courant	Ampère	A
Conductance électrique	Siemens	S
Capacité électrique	Farad	F
Résistance électrique	Ohm	Ω
Numération binaire	Bit	bit

Utilisation des préfixes du système international d'unités.

Utilisation du point comme séparateur décimal (ex : 0.5 V pour 500 mV).

Glossaire

ADC *Analog-to-Digital Converter*. 99, 116, 117, 123, 135, 186–188

AER *Adress Event Representation*. 33

ANN *Artificial Neural Network*. 4, 168

ARM *Advanced RISC Machines*. 7, 84, 85, 113

CCII Convoyeur de courant de seconde génération. xi, xii, xv, xvi, 11, 27, 28, 36, 41, 43, 48, 49, 74, 86, 88, 91–95, 97, 102, 124, 127, 133, 136, 138, 140–143, 147, 148, 150, 151, 153, 154, 164

CD *Compact Disc*. 22

CNN *Convolutional Neural Network*. 4, 168

CNRS Centre National de la Recherche Scientifique. 8

CPTC Croix, Plus, Triangle, Cercle. xi, xv, 11, 34

CPU *Central Processing Unit*. 17, 85, 113, 118

DAC *Digital-to-Analog Converter*. 90, 91, 98, 100–103, 109, 110, 116–118, 121, 135, 147, 174

DCB *Digital Control Block*. 38, 39, 41, 42, 44, 45, 56, 60, 62–64, 66, 72, 85, 104, 107, 113, 114, 116, 130, 134

DVD *Digital Versatile Disc*. 22

DVS *Dynamic Vision Sensor*. 8, 32, 167, 168

FLOPS *Floating Point Operations Per Second*. 17

FPGA *Field-Programmable Gate Array*. 5, 7, 18, 84, 85, 87, 96, 104, 107, 113, 134, 145

FTJ *Ferroelectric Tunnel Junction*. 23

GPU *Graphics Processing Unit*. 4–6

HH Hodgkin-Huxley. xv, 16, 17, 19, 20

IF *Integrate and Fire*. 17, 20

IMS Intégration du Matériau au Système. 8, 168

IoT *Internet of Things*. 2, 5, 167

LIF *Leaky-Integrate-and-Fire*. xi, xii, xv–xviii, 11, 19–21, 26–28, 32, 36, 38, 41, 49, 51, 81, 91–94, 96–98, 102, 119, 130, 133, 142–146, 150, 179

LTD *Long Term Depression*. 15

LTP *Long Term Potentiation*. 14

MNIST *Modified National Institute of Standards and Technology database*. 33, 34

MTJ *Magnetic Tunnel Junction*. 23

N-MNIST *Neuromorphic - Modified National Institute of Standards and Technology database*. xi, xv, xvi, 11, 33, 34, 36, 38, 39, 47–49, 54–60, 62–66, 72

PCM *Phase Change Material*. 22

PL *Programmation Logic*. 113, 116, 117

PS *Processing System*. 113, 116–119, 121, 122, 126, 130

RR *Recognition Rate*. 46, 66, 67, 79

SNN *Spiking Neural Network*. 5, 6, 8, 9, 20, 32, 34, 39, 44–46, 61, 72, 168

SPI *Serial Peripheral Interface*. 117, 118

STDP *Spike-Timing Dependent Plasticity*. xi, xv, xvi, 14, 15, 22, 23, 25, 26, 37, 42, 44, 54–57, 59, 61, 66, 81, 153

TIC Technologies de l'information et de la communication. 2

TPU *Tensor Processing Unit*. 4–6

ULPEC *Ultra Low Power Event-based Camera*. xii, xvi, 6, 8, 19, 20, 37, 38, 41, 43, 44, 52, 55, 72–75, 77, 79, 81, 85, 92, 113, 114, 138, 142, 151, 164, 167, 168

VHDL *Very (high speed integrated circuit) Hardware Description Language*. 84, 85, 113, 117–119, 128, 130, 134, 146, 147, 167

VPU *Vision Processing Unit*. 5, 6

WTA *Winner Take All*. 44

ZIF *Zero Insertion Force*. 87

Table des matières

Table des figures

Liste des tableaux

Introduction

Le cerveau permet à l'être humain de traiter, analyser et interpréter un grand nombre d'informations qui sont utilisées pour réaliser différentes actions comme reconnaître des objets, comprendre un texte ou un discours, parler ou bien se déplacer. L'acquisition de ces informations (visuelles, auditives, tactiles...) collectées par les différents organes ou capteurs du corps humain (œil, oreille, peau...), leur stockage et leur utilisation sont assimilés à la réalisation de tâches dites cognitives [1] que les scientifiques cherchent à reproduire avec des systèmes informatiques ou électroniques pour différentes applications telles que :

— La vision par ordinateur qui sert par exemple à reconnaître dans un flux vidéo provenant d'une caméra des éléments comme des objets, des formes spécifiques ou des personnes.

— Le traitement automatique du langage naturel écrit ou parlé qui regroupe le développement d'outils utilisés pour le traitement du langage naturel (reconnaissance vocale, reconnaissance optique de caractères, analyse de texte, synthèse vocale...).

— La prise de décision autonome (robots, voiture autonome...).

Le champ scientifique des neurosciences regroupe l'étude et la réalisation de ces tâches cognitives ainsi que l'étude du système nerveux et de son fonctionnement qui en sont l'inspiration principale. C'est dans ce champ d'études grandement transdisciplinaire [2] et plus précisément sur les réseaux de neurones qui relèvent des neurosciences computationnelles que se sont déroulés ces travaux de thèses.

Afin de ne pas se perdre dans les méandres des neurosciences computationnelles, un sujet si vaste qu'il résulterait en l'écriture de plusieurs ouvrages pour en aborder tous leurs aspects, cette introduction de quelques pages va, par une succession de questions et de réponses ciblées, permettre d'acquérir les notions fondamentales liées au contexte des travaux présentés.

Pourquoi s'intéresser aux réseaux de neurones et à la biologie ?

L'intérêt croissant des réseaux de neurones et du fonctionnement biologique du traitement d'informations en grande quantité s'explique par la conjoncture de plusieurs problématiques :

1. « Cognitif » : Concernent la connaissance et ses processus d'acquisition.
2. Les neurosciences peuvent faire appel entre autre à l'électronique, la biologie, la chimie, les mathématiques, la bio-informatique et la physique.

1. L'augmentation du nombre de données à traiter par les objets appartenant au domaine des technologies de l'information et de la communication (TIC) tels que les objets connectés (IoT), les téléphones mobiles ou encore les ordinateurs dont le déploiement en nombre impacte considérablement la consommation énergétique mondiale avec une part de la consommation énergétique mondiale estimée à 20.9 % en 2030 [1].
2. L'augmentation des performances des architectures de calcul de ces TIC grâce à l'application de la loi de Moore [3] [2] qui apporte de plus en plus de problématiques poussant ainsi à penser à sa fin d'usage [3]. Les dimensions de plus en plus réduites des transistors permettant d'augmenter les performances se heurtent en effet à une limitation de dimension physique minimale [4], une sensibilité plus forte à la variabilité des processus de fabrication et du bruit électronique tout en présentant des coûts de plus en plus élevés [4].
3. La limitation des fréquences de fonctionnement des circuits électroniques liée entre autres à l'échauffement en température des circuits limitant *de facto* la vitesse de traitement à cause de la capacité restreinte d'évacuation de la chaleur générée. Évacuation de la chaleur qui peut être par ailleurs une tâche très énergivore.
4. La limitation structurelle de l'architecture de VON NEUMMAN [5] [5] des ordinateurs classiques où la mémoire stockant l'information et le processeur assurant son traitement (calcul) séquentiel sont séparés et doivent ainsi fonctionner et communiquer à haute vitesse pour être performant lors de la réalisation de tâches dites cognitives.

Ces problématiques ont poussé l'intérêt pour un autre paradigme de calcul que celui des machines de VON NEUMMAN, les réseaux de neurones, où les unités de calculs (neurones) sont très fortement connectées via des points mémoires (synapses) assurant un grand parallélisme du calcul avec un lien étroit entre la mémoire et les unités de calculs. Ces réseaux de neurones prédisposés à traiter des quantités massives d'information à faible coût énergétique d'après ce qui est observable en biologie [6] se sont ainsi fortement développés pour des tâches cognitives (reconnaissances d'images, reconnaissance vocale...) ainsi que des tâches de prise de décision.

Comment la neurobiologie est arrivée à la théorie du neurone ?

Les neurosciences computationnelles et les réseaux de neurones computationnels trouvent leur origine dans l'étude et la compréhension des systèmes nerveux biologiques et notamment du cerveau humain. Si aujourd'hui l'importance primordiale du cerveau est admise comme centre de commandes des tâches cognitives réalisées par les êtres vivants qui le possèdent et

3. Doublement du nombre de transistors dans une puce électronique tous les deux ans.
4. On ne peut pas fabriquer en théorie de transistors plus petit qu'un groupement d'atomes dont le diamètre est d'environ 0.1 nm.
5. Cette limitation est courament appelé « goulot d'étranglement de VON NEUMANN »
6. Un repas et un peu de café suffisent pour qu'un être humain soit apte à réaliser toute une série de tâches complexe pendent plusieurs heures.

que son fonctionnement se fonde sur la théorie du neurone [7], cette théorie a pris du temps à s'établir. Si les premiers liens entre le cerveau et le contrôle du corps humain semblent avoir été observés de manière clinique sur des cas de blessures crâniennes [6] dès l'Égypte antique sans pour autant lui attribué le rôle central qu'on lui connaît aujourd'hui, c'est à l'époque de la Grèce antique et de l'apparition de l'atomisme au Ve siècle avant notre ère que le cerveau semble être mis en avant comme siège de la pensée d'après Démocrite, élève de Leucippe [8], même s'il n'existe pas de sources fiables sur le sujet [7][8]. Aristote à cette même époque resta fermement opposé à cette idée [7] considérant que le cœur est le centre du fonctionnement des pensées et de l'intelligence et cette théorie incorrecte d'Aristote s'imposa pendant plusieurs siècles. Ce ne fut qu'au IIe siècle que Galien démontra par ses expériences le rôle central du cerveau pour le contrôle du corps et l'activité mentale mettant un terme à l'idée que le cœur est central au fonctionnement de la pensée[9].

Si le rôle central du cerveau est établi par Galien, il faut attendre le XIXe siècle pour que le cerveau et ses fonctions commencent à être pleinement étudiés et découpés par Gall avec un modèle peu réaliste [10]. C'est Brodman qui, part une suite de travaux dès 1903 [11], met au point une carte du cerveau humain en 52 aires [9] en 1908/1909, encore utilisée à ce jour [12][13]. C'est également à cette époque que se confirme la théorie du neurone grâce aux travaux de Ramón y Cajal qui met en évidence la structure cellulaire des neurones (terme déterminé par Waldeyer [14]) connectés par des séparations que l'on appellera synapses (terme déterminé en 1897 par Sherrington [15]). Cette théorie est parachevée en 1955 par les premières images au microscope électronique de la séparation (synapse) de deux neurones, un neurone présynaptique [10] et un neurone postsynaptique [11] [16][17][18]. Neurones et synapses sont aujourd'hui les « briques » de base des réseaux de neurones computationnels. Depuis, la théorie du neurone utilisée pour étudier le fonctionnement du cerveau et comme inspiration par les neurosciences computationnelles, a été complexifiée par l'approfondissement des connaissances sur la structure complexe du cerveau, des neurones (axones, dendrites) et les divers types de synapses (électrique ou chimique).

Comment sont apparus les réseaux de neurones artificiels ?

Les réseaux de neurones artificiels se caractérisent par deux éléments : leur structure [12] et leur règle d'apprentissage [13]. C'est en 1943, quelques années avant les premières images au microscope électronique de la synapse, que McCulloch et Pitts définissent un modèle de neurone

7. Connexion d'unités indépendantes appelées neurones par l'intermédiaire de synapses.
8. Scientifique dont peu de choses sont connues, mais qui est considéré comme le père de l'atomisme.
9. Notés au format BA1 à BA52.
10. « présynaptique » : avant la synapse
11. « postsynaptique » : après la synapse
12. Agencement des neurones et synapses ainsi que leurs modèles.
13. Règle décrivant l'évolution des points mémoires ou synapses dans le réseau suivant la tâche que ce dernier doit résoudre.

pour du calcul logique [19], modèle également appelé neurone formel. Ce modèle de neurone formel sera utilisé par Rosenblatt pour mettre au point le perceptron [20], considéré en 1958 comme le premier réseau de neurones artificiels (ANN) qui associe ainsi neurones formels et règle d'apprentissage [14]. Rosenblatt réussit en effet avec ce perceptron à faire apprendre et distinguer des cartes portant des inscriptions par un ordinateur IBM 704. On peut néanmoins noter que des travaux précédents relevant de la simulation de réseau de neurones [15] inspirés de la biologie et non de neurones formels en 1954 par Farley et Clark [21][22] ont permis l'adaptation [16] d'un réseau pour classifier et reconnaître des configurations (motifs). Si les perceptrons de Rosenblatt assimilés aujourd'hui à un réseau à une couche (sans couche cachée) sont capables de résoudre (classifier) des problèmes linéairement séparables, ils sont mis en difficulté par des problèmes plus complexes non linéairement séparables.

Pour résoudre des problèmes plus complexes, les réseaux de neurones artificiels ont beaucoup évolué et se sont complexifiés en termes de structure, notamment par la mise en cascade de couche de neurones, mais aussi par la création de règles d'apprentissage plus complexes comme la *backpropagation* [17] qui résultent en des architectures nécessitant de plus en plus de calculs pour fonctionner [24]. Cet empilement de couches nécessaires pour des tâches plus complexes (classification d'images par exemple) et ces nouvelles règles d'apprentissage amènent aux réseaux de neurones dits profonds [25] puis aux réseaux de neurones convolutionnels (CNN) qui ont produit de premiers résultats performant sur la reconnaissance de chiffres manuscrits en 1989 dans les travaux de LeCun et coll. [26][27] et sont aujourd'hui parmi les réseaux de neurones les plus performants pour des tâches de reconnaissance d'images. Il existe de nos jours une multitude d'architecture de réseaux de neurones qui ont des qualités et spécificités distinctes.

Existe-t-il des puces dédiées aux réseaux de neurones artificiels ?

Si la biologie met en évidence les avantages d'architectures de calcul reposant sur des réseaux de neurones confrontées à celles reposant sur l'architecture de VON NEUMMAN, la disconvenance entre ces réseaux de neurones et les ordinateurs classiques à calcul séquentiel pose un problème de vitesse d'exécution et de mise en œuvre des réseaux de neurones artificiels. L'utilisation de processeurs graphiques (GPU [18]) présentant des architectures plus parallèles avec de nombreuses unités de calcul et des mémoires plus proches du calcul ont permis de réduire drastiquement les temps de calcul pour la phase d'apprentissage des réseaux de neurones afin de les rendre fonctionnels pour des tâches de reconnaissance. D'autres puces, appelées TPU [19],

14. La règle d'apprentissage correspondant à l'algorithme de modification des poids synaptiques du réseau de neurones.
15. Appelé par les auteurs *self-organizing systems* et non réseau de neurones.
16. Apprentissage inspiré de la règle d'apprentissage de Hebb [23].
17. Méthode de rétropropagation du gradient.
18. De l'anglais *Graphics Processor Unit.*
19. De l'anglais *Tensor Processing Unit.*

VPU [20] ou simplement accélérateur, ont également été développées pour optimiser énergétiquement et accélérer les opérations de calculs les plus utilisés par les réseaux de neurones (produit matriciel, convolution...)[28][29]. Cependant des implémentations sur GPU, TPU ou VPU pour traiter de grandes bases de données peut rapidement nécessiter un grand nombre de ces puces provoquant ainsi le besoin d'une grande quantité d'énergie pour apprendre, ce qui reste une solution coûteuse et seulement accessible à de grandes compagnies industrielles [30]. Une telle solution peut nécessiter pour des systèmes embarqués ou de l'IoT d'avoir un accès à internet stable et performent pour utiliser des modèles fonctionnant sur des serveurs. Aujourd'hui un certain nombre de systèmes reposant sur l'utilisation de réseaux de neurones fonctionnent grâce à la communication avec des serveurs distants qui ont à charge l'implantation des modèles de réseaux de neurones et cela résulte en une utilisation des réseaux de neurones comme des services. Cependant ce fonctionnement à distance pose des questions de consommation énergétique (pour les serveurs et la communication), de rapidité de traitement de l'information (traitement en ligne ou direct), mais aussi de sécurité et confidentialité des données traitées.

Pourquoi choisir les réseaux de neurones événementiels matériels ?

Parmi les différents types de réseaux de neurones qui existent, les réseaux de neurones événementiels ou SNN [21] sont ceux qui ont été retenus pour ces travaux de thèse. Ils appartiennent d'après Maass à la troisième génération de réseau de neurones après ceux utilisant les neurones de McCulloch et Pitts (première génération) et ceux utilisant des neurones avec une fonction d'activation (deuxième génération) [31]. Ces derniers, qui sont fortement inspirés du fonctionnement biologique de notre cerveau, sont considérés comme de bons candidats pour la réalisation de plateformes neuromorphiques à basse consommation puisque contrairement aux autres types de réseaux de neurones ils traitent l'information de manière événementielle et leur source d'inspiration biologique, le cerveau, est aujourd'hui considérée comme un système de calcul embarqué [22] présentant un excellent rapport entre performance et énergie consommée.

Les réseaux de neurones événementiels peuvent être réalisés de différentes manières suivant le type de neurones, le type de synapses et la méthode d'implantations choisie (numérique sur FPGA, par simulation sur processeur, par réalisation analogique ou de circuits dédiés mixtes analogique/numérique). Ces travaux de thèse portent sur l'étude de SNN dont l'implantation est réalisée avec des synapses memristives, des nanocomposants qui seront plus amplement abordées dans le CHAPITRE 1, et des neurones analogiques. Ce choix de la réalisation de ré-

20. De l'anglais *Vision Processing Unit*.
21. *Spiking Neural Network*.
22. On peut assimiler un organisme vivant possédant un cerveau biologique, pouvant se déplacer et opérer sur certaines périodes de temps sans s'alimenter en énergie tel que l'être humain ou autres mammifères comme un système embarqué où le cerveau correspond à l'architecture de calcul et de contrôle.

seaux de neurones événementiels matériels est motivé par l'idée cardinale que les SNN sont de bons candidats pour des solutions peu énergivores et que leur implantation matérielle sur des circuits analogiques sur mesure peut permettre :

— D'obtenir un système optimisé en termes d'énergie pour du calcul événementiel.

— D'avoir un calcul analogique qui se sépare des questions de valeurs finies ou échantillonnées que l'on retrouve dans un calcul en numérique.

— De quitter entièrement le paradigme de VON NEUMMAN avec un circuit qui implémente un réseau de neurones en tant que tel et non une simulation ou émulation sur processeur.

Ce choix s'est retrouvé dans le cadre du projet européen ULPEC dans lequel se sont déroulés ces travaux de thèse.

Il existe déjà des réalisations neuromorphiques matérielles pour SNN qui ne reposent pas sur l'utilisation d'accélérateurs GPU, VPU ou TPU évoqués précédemment, mais plutôt de puces ou plateformes dédiées [23] telles que celle présentées dans la TABLE 1 qui liste une série de plateformes dédiées aux réseaux de neurones événementiels (SNN).

23. Le réseau de neurones est implanté en tant que tel.

TABLE 1 – Quelques réalisations neuromorphiques matérielles.

Nom	Description
Bluehive [32]	Développée par l'université de Cambrige. Implémentation sur logique programmable (64 circuits FPGA) avec 64k neurones événementiels d'Izhikevich et 64M de synapses par FPGA.
NeuroFlow [33]	Développée par l'Imperial College de Londre. Implémentation sur logique programmable permettant l'utilisation de différents types de neurones événementiels.
SpiNNaker [34]	Développée dans le cadre du projet européen « Human Brain Project ». Implémentation sur de multiples cœurs ARM permettant l'utilisation de neurones événementiels de différent types.
Neurogrid [35]	Développée par l'université de Stanford. Combine circuits numériques et analogiques permettant de simuler entre autres 1 million de neurones évenementiels pour une dizaines de watts en temps réel.
HiER-IFAT [36]	Développée par l'université de Californie, San Diego. Puce dédiée regroupant 32 neurocœurs Integrate-and-Fire Array and Transceiver (IFAT) à signaux mixtes. Combine calcul analogique et routage synaptique numérique et événementiel.
ROLLS [37]	Développée par l'université de Zurich. Puce intégrée avec 256 neurones analogiques + synapses analogiques long termes bistables + 256×256 synapses programmables avec circuits de plasticité à court terme + 256×2 rangées d'intégrateurs utilisables comme synapses à court terme.
BrainScales [38]	Développée dans le cadre du projet européen « Human Brain Project ». Sert à émuler les dynamiques biologiques avec un facteur d'accélération de 10 000 Les neurones et les synapses sont réalisés par des circuits analogiques fonctionnant en temps continu.
TrueNorth [39]	Développée par IBM. Architecture impulsionnelle numérique en 28 nm. Permets de simuler jusqu'à 1 million de neurones avec 256 millions de synapses.
Loihi [40]	Développée par Intel. Implémente une architecture événementielle configurable en termes de paramètres des neurones, connexions entre neurones et règles d'apprentissage.

Qu'est-ce que le projet européen ULPEC ?

Les travaux présentés dans ce manuscrit se sont déroulés dans le cadre du projet européen ULPEC [24] (Ultra Low Power Event-based Camera), un projet réunissant 9 partenaires :

— Université de Bordeaux (France, porteuse du projet) via le laboratoire IMS dans lequel se sont déroulés ces travaux de thèses.

— Le CNRS (France).

— Sorbonne Université (France).

— Université de Twente (Pays-Bas).

— École polytechnique fédérale de Zurich (Suisse) aussi appelé ETH Zurich.

— IBM Research GmbH (Suisse).

— Prophesee (France).

— Robert Bosch Gmbh (Allemagne).

— Twente Solid State Technolgy B.V. (Pays-Bas) aussi appelé TSST.

Ce projet de 4.9 M€ [25] sur 4 ans financé dans le cadre du programme H2020 a eu pour ambition de créer un capteur neuromorphique intégré et intelligent à basse consommation utilisant un réseau de neurones événementiels matériel en combinant dans une même puce des technologies innovantes telles qu'une rétine artificielle (caméra événementielle aussi appelée DVS), des neurones en silicium et des memristors ferroélectriques comme synapses. Ces technologies (DVS, neurones, memristors) seront plus amplement détaillées dans le CHAPITRE 1.

Ce projet et le capteur neuromorphique qu'il vise à mettre au point se différencient des plateformes de SNN mentionnées précédemment par la combinaison de technologies visée (neurones en silicium, memristors ferroélectriques et rétine artificielle) pour mettre en place un capteur complet et intégré, ce qui n'est pas le cas des plateformes citées précédemment. Il ne s'agit pas seulement d'une plateforme neuromorphique, mais d'un capteur neuromorphique.

De quoi va parler précisément ce manuscrit ?

Dans le cadre de ce projet de recherche ULPEC, les travaux qui ont amené à ce manuscrit se sont concentrés sur l'étude de réseaux de neurones événementiels à une couche « all to all » pour implantations matérielles à base de synapses memristives [26] et se sont découpés en deux axes parallèles :

1. Définir et étudier des règles d'apprentissage pour de telles structures pour que l'implantation matérielle visée par le projet ULPEC soit fonctionnelle. Ces travaux se sont traduits par la réalisation de simulation sur ordinateur. Le simulateur en question, développé lors

24. ICT-03-2016 - SSI - Smart System Integration.
25. Somme exacte de 4 862 256,25€.
26. Memristor, composant électronique utilisé comme synapse matérielle.

de ces travaux, a été utilisé entre autres pour définir les paramètres des composants matériels du SNN du démonstrateur matériel [27] du projet européen.

2. Mettre en place un réseau de neurones événementiels réduit de 81 entrées et 10 sorties permettant de tester à petite échelle des règles d'apprentissage, mais aussi la caractérisation de 81×10 synapses memristives (memristors), composants critiques dans la réalisation des architectures SNN étudiées.

Si le CHAPITRE 1 aborde les éléments fondamentaux de connaissances nécessaires à la compréhension des travaux présentés (bases de biologie, fonctionnement des neurones événementiels, utilisation des memristors comme synapses...), le CHAPITRE 2 est consacré au premier axe des travaux réalisés, soit l'étude par la simulation de règles d'apprentissage compatibles avec une implantation matérielle exposée à des données tirées du monde réel (issues de caméra événementielle) et qui visent une faible empreinte énergétique et de surface d'implantation de par leur simplicité.

Le CHAPITRE 3 et le CHAPITRE 4 sont eux dédiés au deuxième axe de ces travaux de thèse. Le CHAPITRE 3, détaille l'architecture matérielle de la plateforme mise au point durant ces travaux et qui a nécessité le développement de cartes électroniques ainsi que le développement de l'architecture numérique de contrôle et d'utilisation de cette dernière. Le CHAPITRE 4, présente la vérification expérimentale du bon fonctionnement des circuits électroniques et numériques composant cette plateforme 81×10 ainsi que les premiers résultats de caractérisation obtenus avec cette dernière.

27. Capteur intégré comprenant un capteur neuromorphique (DVS) et un SNN de 784 entrées (image 28×28) et 100 sorties connectées par une matrice de 784×100 synapses memristives.

Chapitre 1

De l'humain au hardware

Ce chapitre expose les éléments fondamentaux de connaissance qui ont été nécessaires pour la réalisation des travaux présentés dans ce manuscrit. Dans un premier temps sera abordé le fonctionnement des cellules biologiques qui assurent le calcul et l'apprentissage au sein de notre cerveau. Ces notions de biologie nous permettront par la suite de nous focaliser sur les réseaux de neurones événementiels qui se veulent facilement intégrables et peu énergivores tout en cherchant à reproduire les fonctions de base des cellules biologiques évoquées dans ce chapitre. Une fois tous les éléments nécessaires à la compréhension de ces réseaux de neurones bio-inspirés présentés, nous parlerons des memristors ferroélectriques utilisés pour réaliser les synapses des réseaux explorés dans ce manuscrit. Le memristor ferroélectrique qui est ainsi un composant électronique essentiel à l'implantation matérielle des réseaux de neurones événementiels étudiés. Nous verrons également les différents types de données que nous pourrions chercher à traiter avec les architectures de calcul événementiel détaillées dans ce manuscrit.

1.1 Que se passe-t-il dans notre cerveau ?

1.1.1 Les bases en biologie

Avec une moyenne de 120 milliards de neurones [41] hautement interconnectés via de nombreuses synapses (environ 10^{15} synapses dans le cerveau [42]) pour seulement quelques dizaines de watts consommés [1] [43][44], le cerveau est inégalé en matière de capacité de calcul pour une telle consommation énergétique. L'architecture de calcul biologique qui est répétée à l'extrême (haut parallélisme, haute interconnectivité) et qui permet au cerveau de fonctionner se simplifie à la connexion d'unités de calcul ou traitement de l'information (neurones) au travers de points mémoire (synapses), comme illustré sur la FIGURE 1.1.

Le principe de fonctionnement de cette architecture de calcul biologique est le suivant. Un neurone présynaptique reçoit des stimulations (potentiels d'actions, événements ou impulsions) d'autres neurones depuis ses dendrites (prolongements nerveux du neurone). Ces stimulations positives (excitation) ou négatives (inhibition) impactent les concentrations de certains ions [2] au sein du neurone provoquant ainsi le changement du potentiel électrique de sa membrane.

1. Consommation globale du cerveau chez l'être humain (calcul, communication, chaleur...).
2. Il s'agit d'ions sodium Na^+ et potassium K^+. D'autres ions sont également présents comme le calcium Ca^{2+} et le chlore Cl^-, mais ne sont pas responsables de la génération du potentiel d'action.

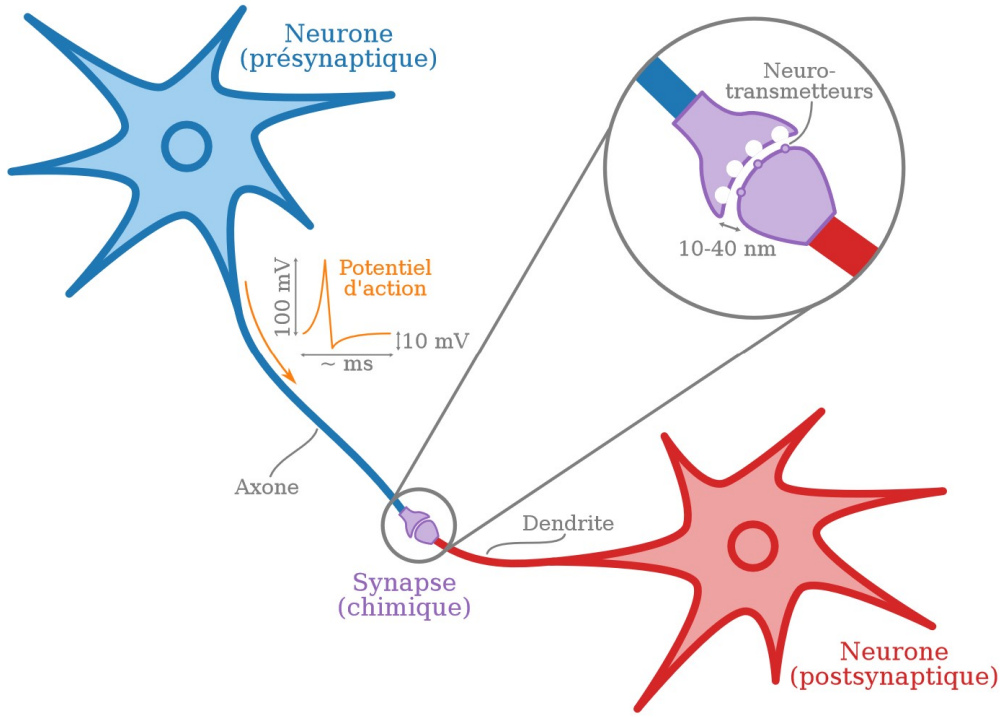


FIGURE 1.1 – Illustration de l'architecture de calcul biologique.
Image tirée de [45]. Un neurone présynaptique est connecté à un neurone postsynaptique par l'intermédiaire d'une synapse chimique.

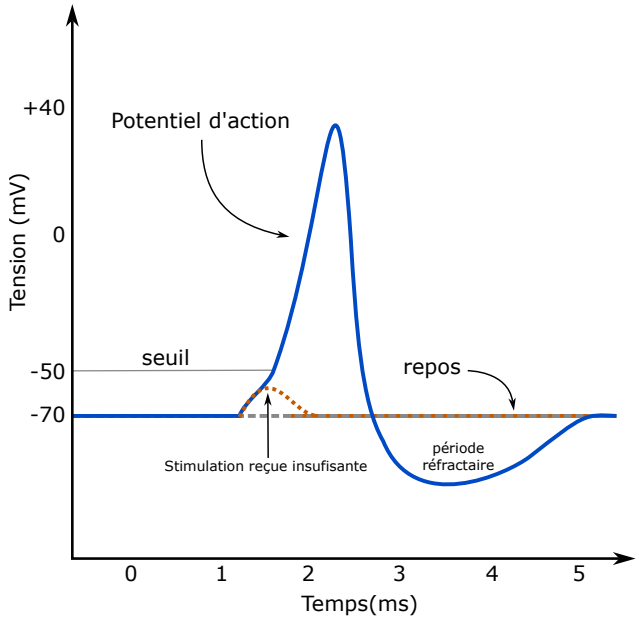


FIGURE 1.2 – Potentiel d'action biologique.
Exemple de réponse typique d'un neurone biologique. La réponse aux stimulations reçues peut être différente suivant le type de la cellule biologique (un ou plusieurs potentiels d'action, forme du potentiel d'action différente...) [46][47].

Au repos, cette membrane est polarisée négativement (−70 mV par exemple). Ce potentiel de membrane non nul est dû à une différence de concentration des ions à l'intérieur et à l'extérieur du neurone [48][49]. Si le potentiel de la membrane atteint un certain seuil (−50 mV par exemple), le neurone va à son tour émettre un potentiel d'action [50] dont la forme typique est illustrée sur la FIGURE 1.2. Cette tension de seuil est notamment atteinte grâce aux stimulations provenant d'autres neurones au travers de synapses. Le potentiel d'action émis va être transmis le long d'un axone (prolongement nerveux permettant la conduction de l'influx nerveux ou potentiel d'action) avant d'atteindre des terminaisons synaptiques qui, suivant leur type (excitatrice ou inhibitrice), vont transmettre aux dendrites d'un neurone postsynaptique une stimulation positive ou négative. La synapse, jonction chimique ou électrique [51][52], qui permet de relier l'axone du neurone présynaptique à la dendrite du neurone postsynaptique [3] transmet la stimulation sous forme de courant dont la quantité est pondérée par le poids de la synapse (sa conductance). Ce poids évolue au cours du temps lors d'une phase qu'on appelle apprentissage. L'évolution positive (potentialisation synaptique) ou négative (dépression synaptique) du poids d'une synapse va dépendre des activités des neurones présynaptiques et postsynaptiques. Cette capacité des synapses à voir leur poids évoluer au cours du temps est appelée plasticité synaptique et les conditions nécessaires à sa modification (positive ou négative) sont appelées règle d'apprentissage.

1.1.2 L'apprentissage en biologie

Si les neurones sont considérés comme les unités de calcul élémentaire du cerveau et sont comparables à des additionneurs (ils additionnent les différentes stimulations provenant d'autres neurones), c'est bien la synapse (son poids, sa fonction et sa présence ou non) qui constitue la mémoire [55]. Elle permet de stocker l'information et peut évoluer au cours du temps. Cette évolution ou plasticité synaptique peut suivre une règle d'apprentissage nommée *Spike Timing Dependent Plasticity* (STDP). Elle a été décrite dans la littérature par Hebb en 1949 [23] et mesurée expérimentalement en biologie *in vitro* [4] par Bi et Poo en 1998 [56]. Comme son nom le suggère (STDP), cette règle cause un changement de la conductance synaptique suivant la proximité temporelle des événements des neurones présynaptiques et postsynaptiques. Un exemple d'une telle règle d'apprentissage est illustré FIGURE 1.3. Lorsque les impulsions des neurones présynaptiques et postsynaptiques connectés par une synapse sont proches temporellement, la modification du poids de cette dernière se réalise de la manière suivante :

— Si l'impulsion postsynaptique intervient après l'impulsion présynaptique, il y a augmentation de la conductance synaptique pour augmenter l'importance des stimulations provenant du neurone présynaptique. Cette phase est appelée LTP (*Long-Term Potentiation*).

3. Les connexions synaptiques en biologie ne se limitent pas seulement à des connexions entre axones et dendrites [53][54].

4. À l'extérieur de l'organisme ; en milieu artificiel, en laboratoire.

— Si l'impulsion postsynaptique arrive avant l'impulsion présynaptique, il y a diminution de la conductance synaptique pour réduire l'importance des stimulations provenant du neurone présynaptique. Cette phase est appelée LTD (*Long-Term Depression*).

— Dans les deux situations, plus les impulsions sont proches dans le temps, plus la modification du poids synaptique est importante.

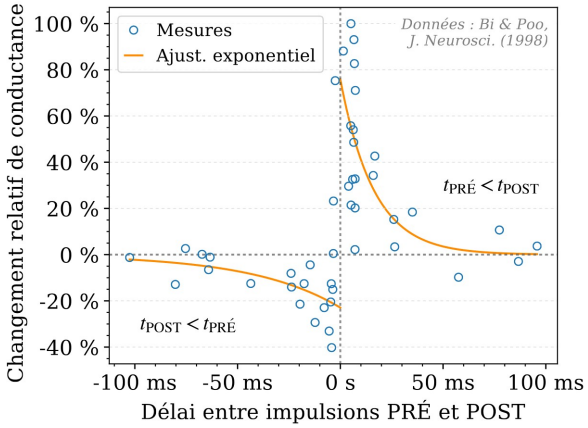


FIGURE 1.3 – Mesure *in vitro* d'une STDP sur les cellules de neurone de l'hippocampe d'un rat. Figure tirée de [45] dont les données (symboles) proviennent de [56].

La règle d'apprentissage décrite ici ne fait intervenir que la temporalité des événements locaux issus des deux neurones qui sont connectés. Il s'agit ainsi d'une règle d'apprentissage locale dite non supervisée. D'autres formes de plasticité ou de règles d'apprentissage de type STDP ont été observées en biologie, comme illustré sur la FIGURE 1.4. Il existe également en biologie des règles d'apprentissage assimilables à de la supervision où des mécanismes de récompense ou de punition interviennent [57][58] pour altérer ou moduler la règle d'apprentissage grâce à la présence de neurotransmetteurs tels que la dopamine et l'acétylcholine.

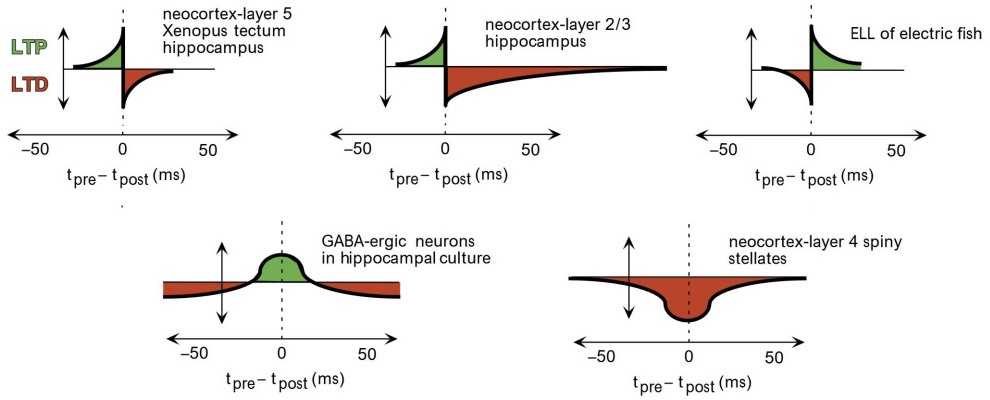
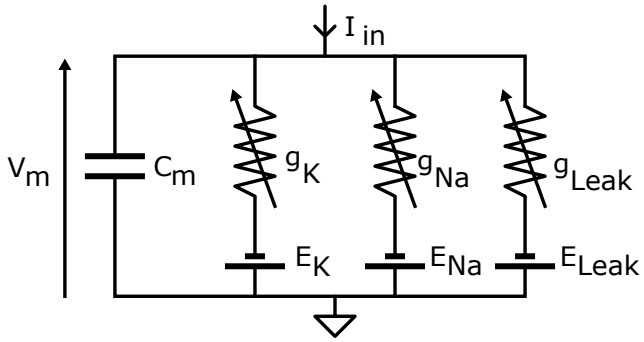


FIGURE 1.4 – Différentes formes de STDP observées en biologie. Image adaptée de [59]. La première STDP représentée montre une règle d'apprentissage qui est semblable à celle mesurée par Bi et Poo en 1998 [56].

1.1.3 Modéliser le comportement biologique

Que ce soit pour émuler fidèlement le comportement du cerveau (biomimétique) ou pour reproduire de manière synthétique son comportement (bio-inspirée) pour le développement de réseaux de neurones pour des tâches plus computationnelles (reconnaissance vocale, reconnaissance d'images...), il est nécessaire d'avoir à disposition des modèles mathématiques de neurones capables de reproduire plus ou moins fidèlement le comportement de ces cellules biologiques. Il existe aujourd'hui de nombreux modèles pour décrire le comportement des neurones impulsionnels. L'un des plus célèbres a été mis au point en 1952 par Hodgkin et Huxley [60] en s'appuyant sur des observations biologiques. Leur modèle, qui porte leurs noms (Hodgkin-Huxley ou HH), représente de manière électrique la membrane plasmique des neurones (idéalisée par une capacité électrique) en prenant en compte les différents canaux ioniques présents au niveau de cette membrane et qui sont à l'origine de l'évolution de son potentiel électrique. Ce modèle électrique est donné FIGURE 1.5.



$$C_m \frac{dV_m}{dt} = I_{in}(t) - \overline{g}_K n^4 (V_m - E_k) - \overline{g}_{Na} m^3 h (V_m - E_{Na}) - \overline{g}_{leak} (V_m - E_{leak})$$

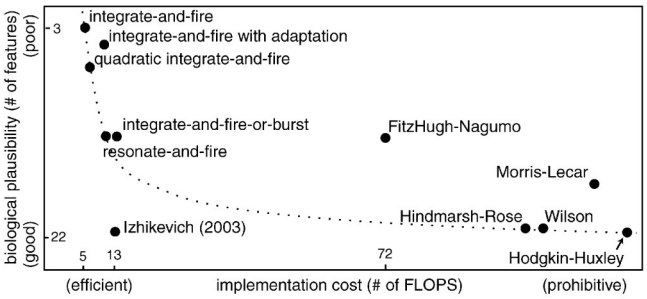
$$\frac{dn}{dt} = \alpha_n (V_m)(1-n) - \beta_n (V_m) n$$
$$\frac{dm}{dt} = \alpha_m (V_m)(1-m) - \beta_m (V_m) m$$
$$\frac{dh}{dt} = \alpha_h (V_m)(1-h) - \beta_h (V_m) h$$

FIGURE 1.5 – Schéma électrique et équations du modèle HH pour le calcul du potentiel de la membrane d'un neurone.

Les canaux ioniques à sodium et potassium permettent de faire circuler les ions à travers la membrane plasmique du neurone jouant ainsi sur le potentiel de membrane. Ces canaux ioniques sont représentés par des conductances électriques qui varient au cours du temps. Les valeurs α_i et β_i sont des facteurs d'activation et d'inactivation de ces canaux ioniques.

En plus de ce modèle de neurone qui se veut réaliste, il en existe de nombreux autres, comme illustré sur la FIGURE 1.6, qui s'éloignent plus ou moins de la représentation réaliste des différentes caractéristiques biophysiques des neurones biologiques. Cette variété de modèles per-

met de choisir le compromis que l'on peut souhaiter entre une très bonne représentation biologique et le coût d'implémentation en termes d'opérations à réaliser (représenté par le nombre de FLOPS [5] sur la FIGURE 1.6).



Models	biophysically meaningful	tonic spiking	phasic spiking	tonic bursting	phasic bursting	mixed mode	spike frequency adaptation	class 1 excitable	class 2 excitable	spike latency	subthreshold oscillations	resonator	integrator	rebound spike	rebound burst	threshold variability	bistability	DAP	accommodation	inhibition-induced spiking	inhibition-induced bursting	chaos	# of FLOPS
integrate-and-fire	-	+	-	-	-	-	-	+	-	-	-	-	+	-	-	-	-	-	-	-	-	-	5
integrate-and-fire with adapt.	-	+	-	-	-	-	+	+	-	-	-	-	+	-	-	-	-	+	-	-	-	-	10
integrate-and-fire-or-burst	-	+	+		+	-	+	+	-	-	-	-	+	+	+	-	+	+	-	-	-		13
resonate-and-fire	-	+	+	-	-	-	-	+	+	-	+	+	+	+	-	-	+	+	+	-	-	+	10
quadratic integrate-and-fire	-	+	-	-	-	-	-	+	-	+	-	-	+	-	-	+	+	-	-	-	-	-	7
Izhikevich (2003)	-	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	13
FitzHugh-Nagumo	-	+	+	-		-	-	+	-	+	+	+	-	+	-	+	+	-	+	+	-	-	72
Hindmarsh-Rose	-	+	+	+			+	+	+	+	+	+	+	+	+	+	+	+	+	+		+	120
Morris-Lecar	+	+	+	-		-	-	+	+	+	+	+	+	+		+	+	-	+	+	-	-	600
Wilson	-	+	+	+			+	+	+	+	+	+	+	+	+	+		+	+				180
Hodgkin-Huxley	+	+	+	+			+	+	+	+	+	+	+	+	+	+	+	+	+	+		+	1200

FIGURE 1.6 – Image et tableau listant différents modèles de neurones.
Image et tableau tirés de [61] Le tableau compare les caractéristiques biophysiques qu'ils implémentent ainsi que leur coût d'implémentation sur CPU en FLOPS. Si le modèle de HH est considéré comme une bonne représentation de la biologie mais est très coûteux à implémenter, le modèle *Integrate and Fire* (IF) est bien moins coûteux, mais moins représentatif.

Modéliser seulement le modèle de la membrane plasmique d'un neurone ne suffit pas à implémenter une représentation biomimétique du cerveau. Il est nécessaire de pouvoir compléter ce modèle avec des représentations des autres parties du neurone comme les axones et les dendrites qui impactent la forme du potentiel d'action perçu et émis par le neurone. Il existe ainsi des modèles dits à compartiments multiples qui permettent d'intégrer les différentes parties de la connexion entre les neurones (axone, dendrites, soma, synapses) pour mieux prendre

5. Le nombre d'opérations en virgule flottante par seconde.

en compte sa morphologie. Ces modèles peuvent être plus ou moins détaillés et faire appel à un nombre varié de compartiments. Cet ajout de compartiments permet d'améliorer la fidélité de la reproduction de la biologie d'après Davison [62] (en comparant des modèles à 2,3 et 4 compartiments) et certains modèles montent jusqu'à des milliers de compartiments [63]. Concernant la modélisation des synapses en particulier, il existe des représentations mathématiques des différentes synapses biologiques excitatrices (récepteurs $AMPA$ et $NDMA$ par exemple) ou inhibitrices (récepteurs $GABA_A$ et $GABA_B$ par exemple) qui existent dans le cerveau [64]. Aujourd'hui, l'implémentation de modèles de neurones à compartiments multiples prenant en compte la modélisation de tous ces éléments biologiques sur FPGA font l'objet de travaux et ont pour but de permettre l'étude, entre autres [6], de maladies dégénératives [65][66].

6. D'autres applications possibles sont les interfaces homme-machine au contact direct du cerveau.

1.2 Réseau de neurones événementiels matériels

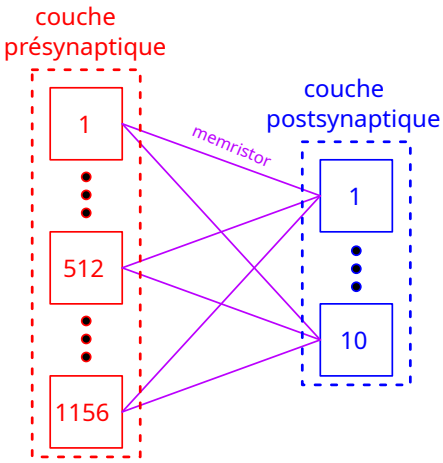


FIGURE 1.7 – Réseau de neurones événementiels *all-to-all* à une couche 1156 × 10. La couche d'entrée est constituée de 1156 générateurs de signaux (choix arbitraire pour cette image) appliquant des impulsions sur les bornes présynaptiques des memristors correspondant aux informations à traiter (image/matrice 34 × 34 vectorisée). La couche de sortie et constituée de 10 neurones postsynaptiques (choix arbitraire pour cette image) de type LIF. La couche d'entrée est connectée à tous les neurones de sortie par des synapses memristives.

Pour réaliser un réseau de neurones événementiels matériel, il est nécessaire d'avoir deux éléments : les neurones événementiels et les synapses. Le modèle de neurone HH présenté précédemment fait appel à des jeux d'équations différentielles non linéaires qui le rendent difficile à implémenter en analogique. De tels modèles de neurones ont déjà été réalisés en analogique [67][68], mais restent difficile à ajuster correctement pour obtenir le comportement souhaité [69][70] ce qui pose problème pour le passage à l'échelle d'un réseau complet. Dans le cadre de cette thèse, un modèle bien plus simple que celui de HH a été choisi (le neurone LIF) pour une implémentation plus simple et plus intégrable (peu d'équations mathématiques pour la simulation, peu de composants électroniques pour l'implémentation matérielle). Ce choix d'utiliser des neurones LIF s'est justifié par l'application uniquement computationnelle voulue pour le démonstrateur du projet ULPEC ne nécessitant donc pas un modèle biomimétique plus complexe pour la réalisation du réseau de neurones. Pour l'implémentation des synapses, le memristor, un composant électronique présentant toutes les propriétés nécessaires pour l'émulation d'une synapse a été choisi. En tant que composant analogique, le memristor permet en théorie une quantité infinie de poids synaptiques. Nous nous intéresserons en particulier aux réseaux de neurones événementiels *all-to-all* à une seule couche (un exemple est donné FIGURE 1.7) signifiant que les entrées présynaptiques du réseau sont connectées par des memristors (synapses) à tous les neurones postsynaptiques de type LIF. Ce choix restrictif

(neurones LIF, une seule couche, *all-to-all*, synapses memristives) correspond à la structure qui a été choisie [7] pour le démonstrateur du projet ULPEC, projet dans lequel s'inscrivent ces travaux.

1.2.1 Le choix des neurones : Le neurone LIF

Le modèle de neurone choisi pour ces travaux de thèse, appelé neurone "intègre et tire avec fuite" ou neurone LIF (de l'anglais *Leaky Integrate and Fire*), est issu d'une modélisation plus ancienne que le modèle HH. En 1907, Louis Lapicque modélise électriquement un premier neurone biologique avec uniquement une capacité (de membrane) en parallèle à une résistance de fuite [71], comme illustré sur la FIGURE 1.8. Il ne s'agit alors que de la partie membrane du neurone où il suffit d'injecter un courant électrique pour la stimuler.

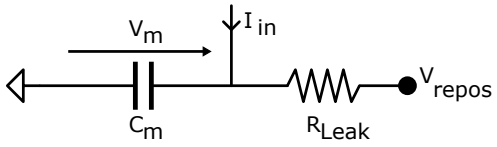


FIGURE 1.8 – Modélisation électrique du neurone par Lapicque en 1907.
La stimulation de cette membrane du neurone s'effectue par l'entrée en courant I_{in} et sa tension de repos en l'absence de stimulation est $V_m = V_{repos}$.

À partir de ce modèle de membrane est alors défini le modèle de neurone LIF qui consiste à émettre un événement lorsque la tension de membrane atteint un certain seuil. Le neurone LIF est une variante du modèle de neurone IF où une fuite sous forme de résistance a été ajoutée. Ce neurone LIF (ou IF) est aujourd'hui considéré comme la représentation la plus simplifiée d'un neurone impulsionnel, la rendant plus simple à réaliser et à implémenter ce qui a motivé le choix de ce type de neurone pour le démonstrateur ULPEC. Sur la FIGURE 1.9 sont illustrés deux versions de neurone LIF qui se différencient par le type de la décharge (fuite) de la membrane. Ces deux versions ont été simulées et/ou implémentées dans ces travaux de thèse pour des raisons différentes qui seront expliquées dans les chapitres suivants. On peut noter que lors d'implantations matérielles de neurones LIF, un système de remise à zéro ou repos de la membrane est généralement ajouté [8] afin de pouvoir (ré)initialiser aisément le neurone à sa tension de repos.

7. La structure exacte choisie du SNN pour le projet ULPEC est détaillée 2.3 Cas d'étude : Le projet ULPEC.
8. Il s'agit généralement d'un interrupteur en parallèle de la membrane reliant à la tension de repos ou une résistance faible pour décharger rapidement la membrane.

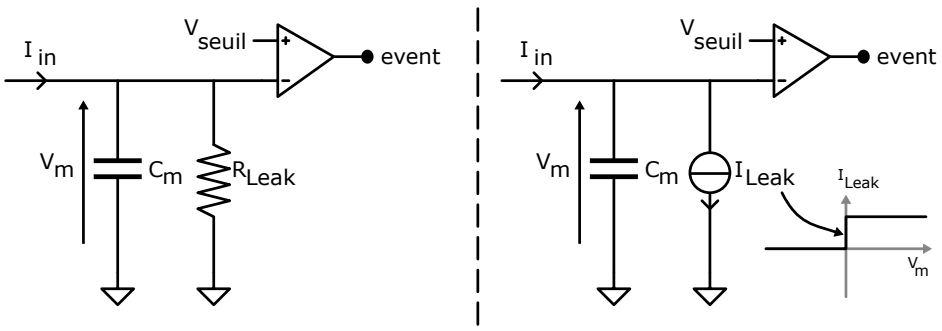


FIGURE 1.9 – Neurones LIF étudiés dans ce manuscrit.
Si le courant injecté I_{in} permet à la tension de membrane V_m d'atteindre la valeur seuil, un événement *event* est émis par le neurone. Le premier modèle à résistance de fuite peut fonctionner pour des courants I_{in} positifs (excitation) ou négatifs (inhibition). Le deuxième modèle à courant de décharge ne fonctionne que si la tension de membrane reste positive, mais permet de s'affranchir d'une décharge variable en fonction de la tension de membrane.

1.2.2 Le choix des synapses : Le memristor ferroélectrique, un nano-composant

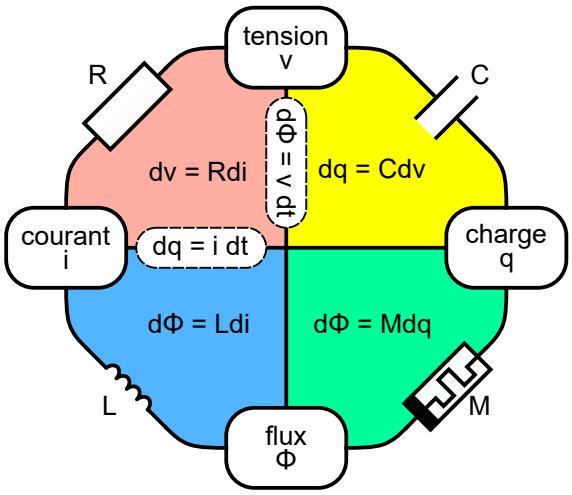


FIGURE 1.10 – Les 4 composants élémentaires passifs de l'électronique.
Composants élémentaires passifs de l'électronique et les relations entre les différentes grandeurs physiques.

Le memristor (*memory-resistor*) a été théorisé en 1971 par Léon Chua [72] comme quatrième composant élémentaire passif de l'électronique aux côtés du condensateur, de l'inductance et de la résistance, comme illustré sur la FIGURE 1.10. Ce dernier peut garder en mémoire sa valeur de résistance de manière non volatile sans avoir à consommer de courant. Il est comparable à un élément résistif dont la valeur de résistance dépend du courant qui le traverse. Pour qu'il y

ait une variation non volatile de sa résistance à une tension donnée, il faut que le courant qui le traverse soit suffisamment fort ou qu'une différence de potentiel (champ électrique) suffisante soit appliquée à ses bornes.

La première annonce de la réalisation physique de ce composant date de 2008 [73] et aujourd'hui de nombreuses nanotechnologies reposant sur des phénomènes physiques différents permettent de réaliser ce type de composant :

Les memristors fondés sur des mécanismes d'oxydoréduction

On peut distinguer deux structures de memristor utilisant sur ce type de phénomènes :

— Celles constituées d'un électrolyte solide entre deux électrodes dont l'une d'elles, « active », peut s'oxyder en ions métalliques. L'application d'un champ électrique permet la migration d'ions depuis l'électrode active vers l'autre pour former des ponts conducteurs [74]. Plusieurs expérimentations montrent la possibilité de reproduire des mécanismes de STDP avec ce type de composant afin de les utiliser comme connexion synaptique [75][76].

— Celles constituées d'oxyde de métaux de transition (HfO_x, NiO_x, TiO_x, AlO_x...) qui permettent de déplacer des vacances d'oxygène pour créer des filaments conducteurs à travers un oxyde par l'application d'un champ électrique [77][78]. Ces dispositifs sont composés d'une couche d'oxyde métallique (isolant) entre deux électrodes métalliques [9]. Les memristors annoncés en 2008 par HP appartiennent à cette famille [73]. Des travaux d'Alibart et coll. ont démontré expérimentalement en 2013 leur utilisation comme synapses artificielles pour réaliser de l'apprentissage avec un perceptron [79].

Les memristors utilisant des mécanismes de changement de phase d'un matériau

Constitué d'un verre de chacolgénure entre deux électrodes, cette technologie nommée PCM, déjà utilisée dans le stockage de données type CD ou DVD, utilise le principe de transition de phase d'un matériau entre une phase cristalline et une phase amorphe. Ce changement de phase, correspondant à un réarrangement de l'agencement des atomes du matériau, se traduit par un changement de résistance électrique [10]. Des travaux ont mis en évidence la possibilité de réaliser une STDP avec ce type de composants [80]. La démonstration expérimentale d'un apprentissage avec ce type de synapses memristives et des perceptrons a également été réalisée [81].

Les memristors organiques utilisant des nanoparticules

L'utilisation de NOMFET (*Nanoparticule Organic Memory Field-Effect Transistor*), un transistor organique, permet d'obtenir le comportement d'un memristor même si le nom privilégié

9. Structure MIM : Métal-Isolant-Métal.
10. La phase cristalline conduit mieux le courant que la phase amorphe.

pour ce type de composant est « synapstor » pour synapse-transistor. Il est en effet possible de reproduire une forme de STDP en connectant le drain et la grille de tels dispositifs [82].

Les memristors reposant sur des mécanismes purement électronique

Parmi les memristors à effet purement électronique, ce sont ceux appartenant à la famille des memristors ferroélectriques (Jonctions Tunnels Ferroélectriques FTJ) qui ont été retenus dans ces travaux de thèse [11]. Le choix de cette technologie s'explique par les motifs suivants :

— Les niveaux de résistance les plus faibles obtenus peuvent être de l'ordre de 100 kΩ [86] pour des rapports de résistance entre l'état ON (poids fort, très conducteur) et OFF (poids faible, peu conducteur) de l'ordre du millier, favorisant ainsi une consommation en courant faible lors de la phase de stimulation/lecture (quelques dizaines de micro-ampères pour des tensions de quelques volts maximum par exemple).

— Seulement des phénomènes électroniques interviennent, ce qui est favorable à la réalisation d'un réseau de neurones intégré dans une puce électronique.

— Les dimensions de ces composants sont potentiellement très petites, de l'ordre de la centaine de nanomètres [86] facilitant ainsi des intégrations à grande échelle.

Ces propriétés du memristor ferroélectrique en font ainsi un bon candidat pour la réalisation de synapses artificielles. En effet, si on lui applique :

— une tension fixe à ses bornes (stimulation), le courant qui circulera en son sein correspondra à la pondération de la tension appliquée par son poids (conductance électrique) ;

— une combinaison de tensions suffisamment importante à ses bornes, son poids changera de manière non volatile.

Plus précisément, ces memristors ferroélectriques sont constitués de 2 électrodes métalliques au contact d'une barrière ferroélectrique. Le matériau ferroélectrique est composé de dipôles électriques (ou grains de polarité) d'orientations différentes. Lorsqu'un champ électrique est appliqué aux bornes de ce composant, les différents domaines de polarisation tendent à s'orienter dans le même sens que ce champ et, suivant l'orientation totale résultante de ces domaines, la résistance du composant varie. Un exemple de stack d'un tel memristor est illustré sur la FIGURE 1.11. Sur la FIGURE 1.12 est également illustrée la mesure de résistance d'un tel composant, dont les résultats ont été obtenus avec l'application de créneaux de tension variables, ainsi que les différents domaines ferroélectriques en fonction de la résistance de ce dernier. On peut notamment y observer les seuils de tension à partir desquels la résistance change de manière non volatile (environ 1 V pour augmenter la résistance et −2.5 V pour la diminuer).

11. Il existe également des memristors à Jonctions Tunnels Magnétiques (MTJ) qui utilisent les propriétés magnétiques et spintroniques des matériaux [83][84][85].

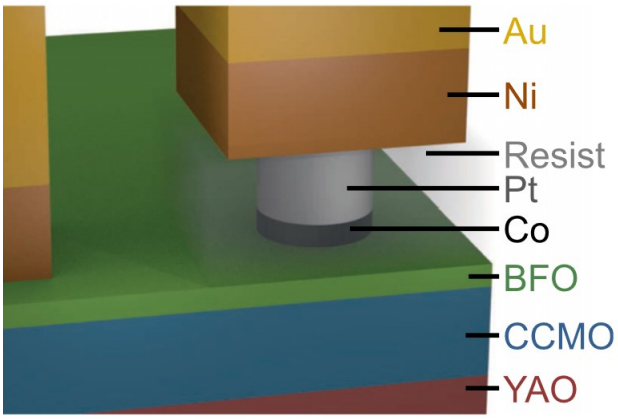


FIGURE 1.11 – Exemple de stack physique d'un memristor ferroélectrique.
Image tirée de [86]. Le matériau ferroélectrique BFO ($BiFeO_3$) est en sandwich entre les électrodes Co et CCMO ($Ca_{0.96}Ce_{0.04}MnO_3$). La structure a été réalisée sur le substrat « exotique » (non silicium) YAO ($YAlO_3$). L'électrode supérieure (Au/Ni) accède à la jonction ferroélectrique grâce à un contact en platine (Pt). L'électrode inférieure accède à la jonction en passant par le substrat YAO (électrode Au/Ni au contact direct du substrat sur le côté en passant au travers des couches BFO/CCMO).

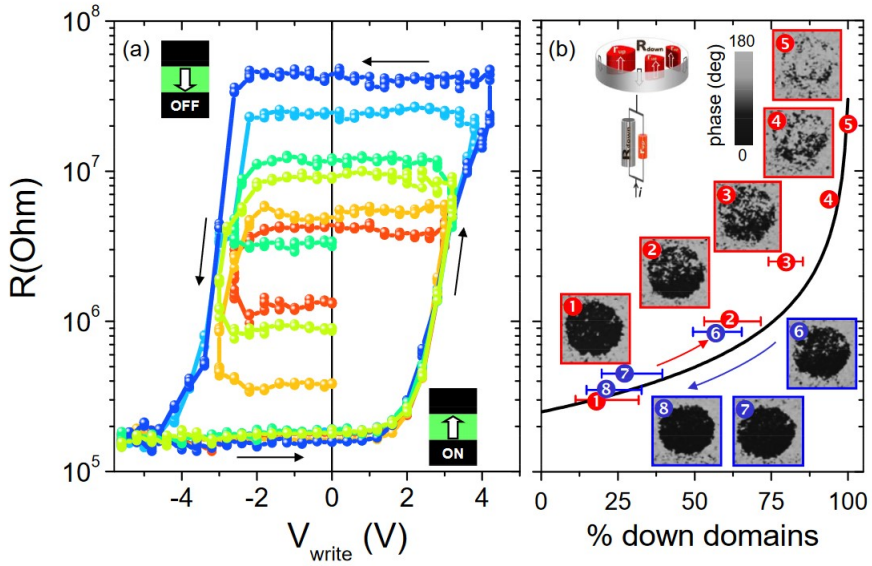


FIGURE 1.12 – Cycles d'hystérésis d'un memristor ferroélectrique et représentation des domaines en fonction de la valeur de la résistance.
Image tirée de [87]. (a) Cycles d'hystérésis obtenus par l'application successive d'impulsions de 20 ns et d'amplitudes de tension maximum différentes. (b) Images PFM (*Piezoresponse Force Microscopy* [88]) pour différents niveaux de résistance. L'application d'impulsions de plus en plus fortes et positives cause l'apparition de domaines de polarisation orientée vers le haut dans le matériau. À l'inverse, l'application d'impulsions de plus en plus négatives retourne les domaines de polarisation et augmente ainsi le nombre de domaines de polarisation orientée vers le bas. C'est le sens global d'orientation de ces domaines de polarisation dans la jonction qui impacte la valeur de résistance lue.

Le memristor ferroélectrique est capable de reproduire les fonctions nécessaires à la réalisation d'une synapse. En effet, si on sépare les tensions du memristor à ses bornes en une tension présynaptique (V_{pre}) et une tension postsynaptique (V_{post}), il est possible de réaliser une STDP similaire à celle que présentent les synapses biologiques. En appliquant à ses bornes des signaux spécifiques, comme illustré sur la FIGURE 1.13, on peut ainsi produire une STDP en jouant sur la combinaison de ces deux tensions qui sont composées de 2 parties :

— Une partie créneau de tension utilisée pour dépasser les seuils d'écriture du composant et modifier de manière non volatile sa valeur de résistance.

— Une partie rampe qui est utilisée ici à la fois pour définir la fenêtre d'apprentissage (plage temporelle sur laquelle il peut y avoir modification du poids) et pour pondérer par rapport au temps la modification du poids synaptique (plus les impulsions présynaptique et postsynaptique sont proches, plus l'amplitude de dépassement est importante et plus la modification est importante).

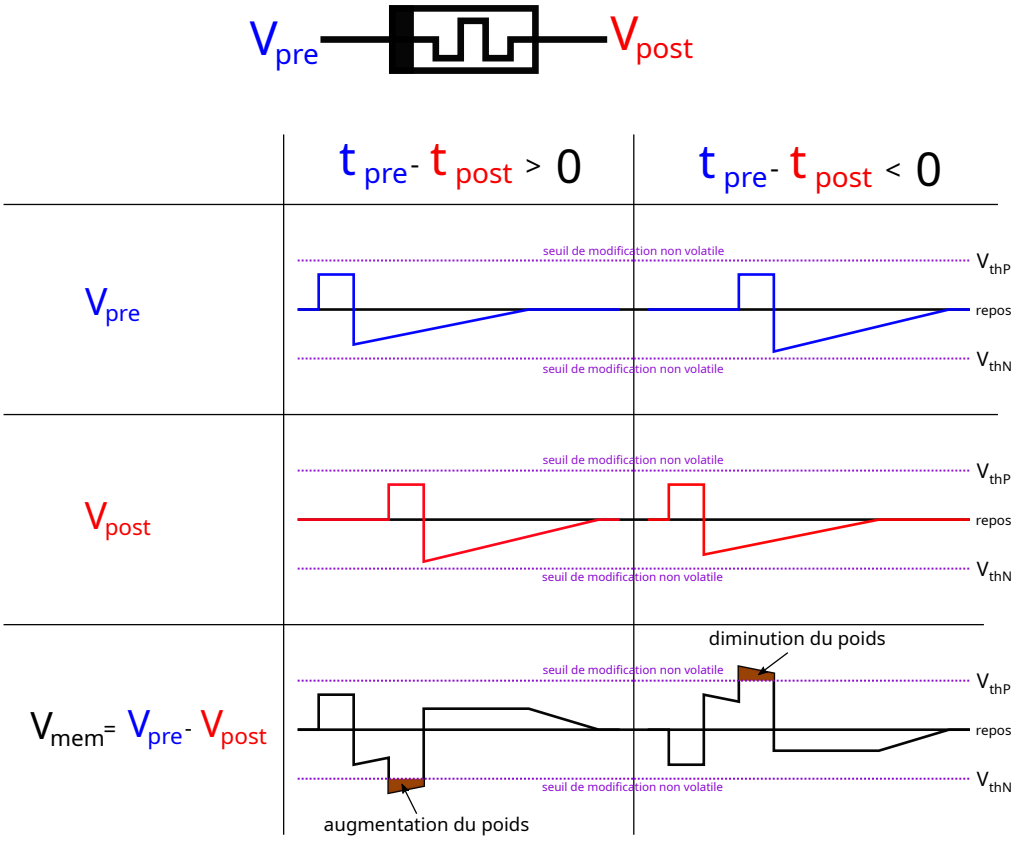


FIGURE 1.13 – Signaux V_{pre} et V_{post} utilisés pour reproduire une STDP inspirée de la biologie. Grâce à la partie rampe des signaux V_{pre} et V_{post}, plus ces derniers sont éloignés temporellement moins la tension aux bornes du memristor dépassera les seuils d'écriture (V_{thP}, V_{thN}) et plus la modification du memristor sera faible. Généralement $T_{pulse} \ll T_{rampe}$.

Cette même forme d'onde présynaptique peut également être utilisée pour faire la stimulation du neurone postsynaptique pour ne pas avoir à générer d'autres formes. Dans le cas présenté

ci-dessus, sachant que la partie impulsion est supposée être beaucoup plus courte que la partie rampe qui définit la fenêtre d'apprentissage, c'est cette dernière qui présente la contribution majeure en courant de toute la forme d'onde pour la membrane du neurone postsynaptique LIF. Ainsi c'est la partie à tension négative sur la borne présynaptique qui permet de générer le courant de simulation. C'est pour cette raison que pour tous les travaux réalisés lors de cette thèse, les tensions de simulation des synapses (memristors) excitatrices ou inhibitrices seront négatives ($V_{\text{pre}} < 0\,\text{V}$).

Plusieurs formes de tension peuvent être utilisées pour réaliser différentes formes de STDP avec des memristors, comme illustré sur la FIGURE 1.14. Ainsi, suivant les signaux appliqués aux bornes du memristor, il est possible d'obtenir un comportement très polyvalent pour des tâches d'apprentissage.

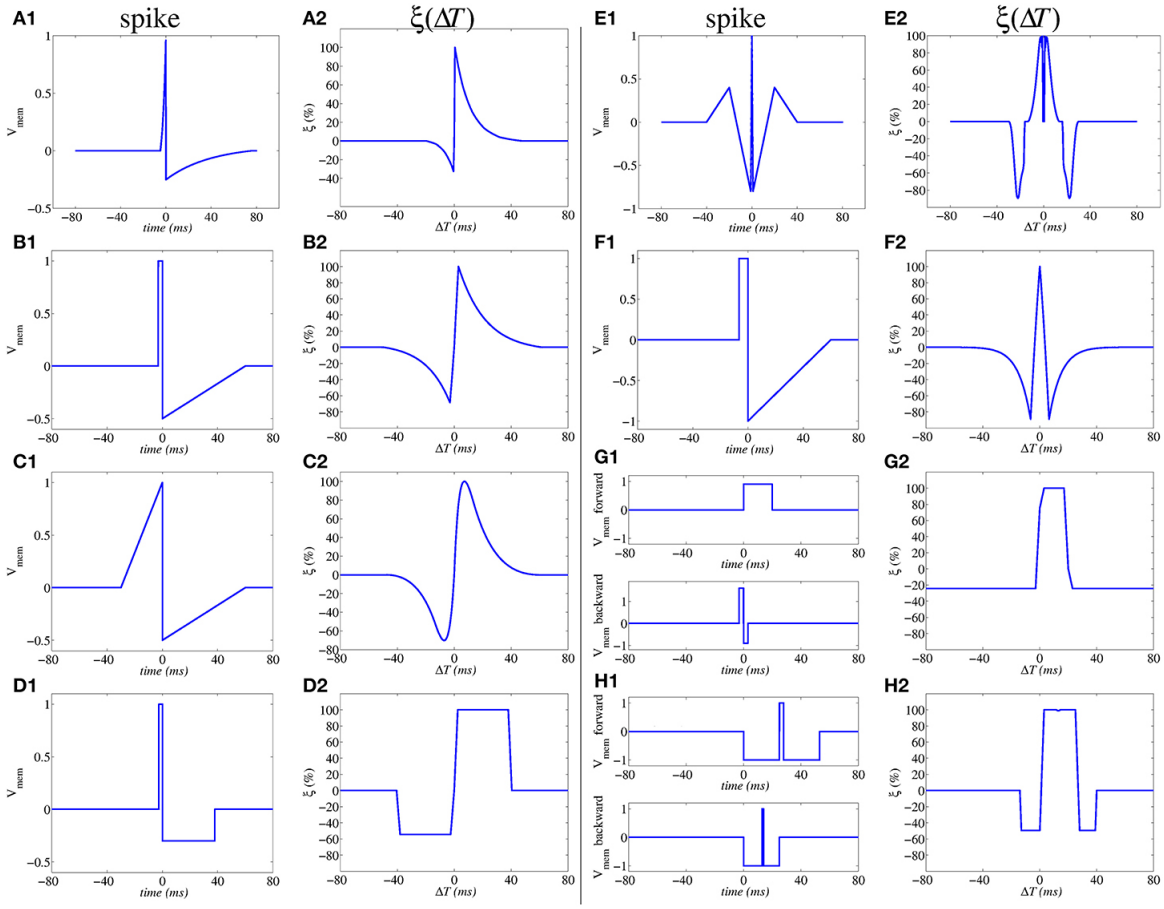


FIGURE 1.14 – Type de signaux (*spike*) aux bornes d'un memristor et STDP résultante ξ. Image tirée de [89]. Résultats de simulations.

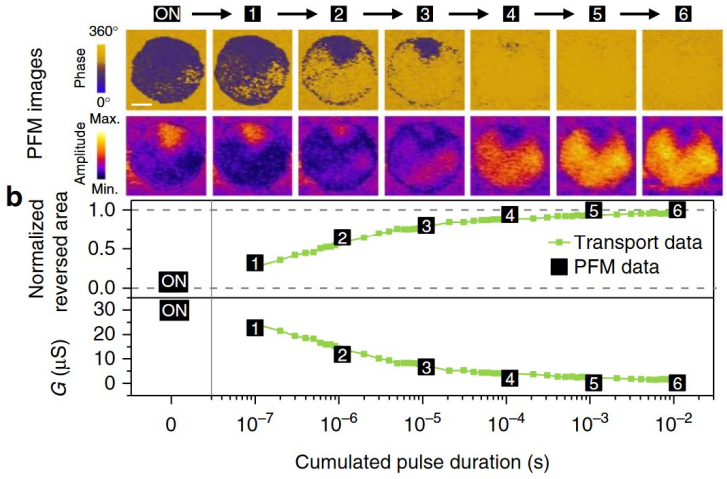


FIGURE 1.15 – Modification progressive de la résistance d'un memristor ferroélectrique par application successive d'une impulsion.
Image tirée de [90]. La partie supérieure de l'image illustre l'évolution des domaines de polarisation du memristor. La partie inférieure illustre l'évolution de de la conductance du memristor.

La méthode présentée FIGURE 1.13 utilisant une rampe de tension repose donc sur une modulation de l'amplitude d'écriture pour jouer sur la variation du poids du memristor. En plus de ce phénomène de modification pondérée par la tension, il est possible d'avoir une modification progressive du poids du memristor par application successive de créneaux d'amplitude d'écriture fixe suffisamment courts [87][90], comme illustré sur la FIGURE 1.15.

Il y a une considération pratique supplémentaire à prendre en compte concernant le comportement des memristors. Si un memristor se comporte comme une résistance pour une tension de mesure ou de stimulation donnée, son comportement réel (tant qu'il n'est pas dans sa zone de modification non volatile) est similaire à celui d'une diode. En effet, sa valeur de résistance mesurée (obtenue lors d'une hystérésis) dépend de la valeur de tension choisie pour faire cette mesure. Cette considération est à prendre en compte notamment pour choisir l'amplitude de la tension de stimulation ou de mesure du memristor lors de son utilisation.

1.2.3 Connexion entre le neurone postsynaptique LIF et la synapse memristive : le CCII

Pour utiliser le memristor comme synapse, son courant, image du poids synaptique (conductance), doit être récolté tout en assurant l'application des tensions V_{pre} et V_{post} à ses bornes afin de pouvoir modifier son poids si nécessaire. La borne postsynaptique du memristor doit donc être configurée comme entrée en tension et sortie en courant. Cette configuration est rendue possible par l'utilisation d'un convoyeur de courant de seconde génération (CCII, inventé par S. Smith en 1968 [91] et illustré figure FIGURE 1.16). L'utilisation de ce composant comme

étage d'entrée d'un neurone postsynaptique pour lui connecter des synapses excitatrices ou inhibitrices été introduit par Gwendal Lecerf en 2013 [92].

$$\begin{bmatrix} i_Y \\ v_X \\ i_Z \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & \pm 1 & 0 \end{bmatrix} \begin{bmatrix} v_Y \\ i_X \\ v_Z \end{bmatrix}$$

FIGURE 1.16 – CCII tel qu'introduit en 1968.

Le CCII utilisé dans l'étage d'entrée est ainsi crucial pour le fonctionnement du neurone postsynaptique. Il existe deux versions différentes, CCII+ et CCII– suivant le sens du courant copié. Ces deux types de CCII sont utilisés pour définir le type des synapses memristives connectées au neurone postsynaptique, comme illustré sur la FIGURE 1.17. Pour une même stimulation sur les entrées inhibitrice ou excitatrice (tension négative appliquée sur la borne présynaptique), le CCII+ injectera du courant dans la membrane pour stimuler le neurone postsynaptique alors que le CCII- en puisera pour l'inhiber. Sauf mention contraire par la suite, seules les entrées excitatrices sont considérées et il n'y a donc pas d'étude sur l'implantation de synapses inhibitrices (CCII-) sur les réseaux étudiés dans ces travaux de thèse.

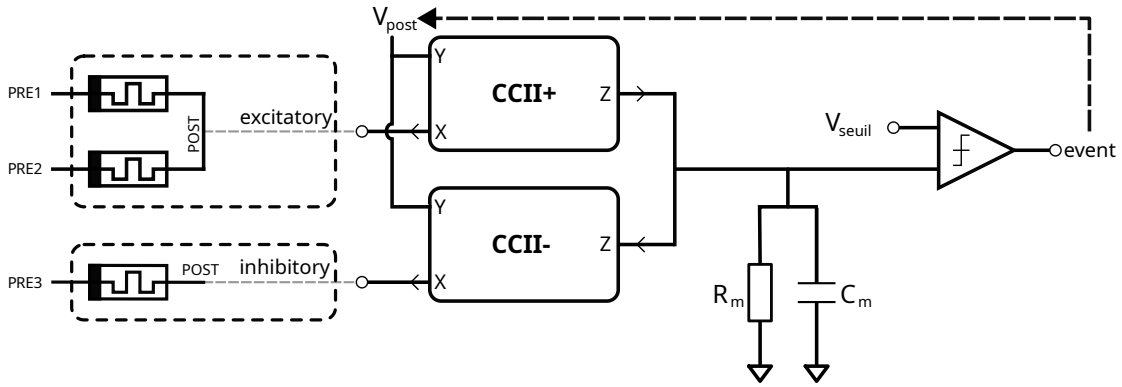


FIGURE 1.17 – Synapses excitatrices et inhibitrices connectées à un neurone postsynaptique LIF.

Cette architecture constitue la structure de base des réseaux de neurones événementiels matériels. Pour plusieurs synapses du même type connectées à un neurone postsynaptique il suffit de connecter les bornes postsynaptiques des memristors puisque tous les courants de stimulation sont collectés par le CCII suivant la loi des nœuds. L'émission d'un signal event est utilisée pour déclencher la forme d'onde postsynaptique si nécessaire.

1.2.4 Mise à l'échelle : le crossbar de memristors

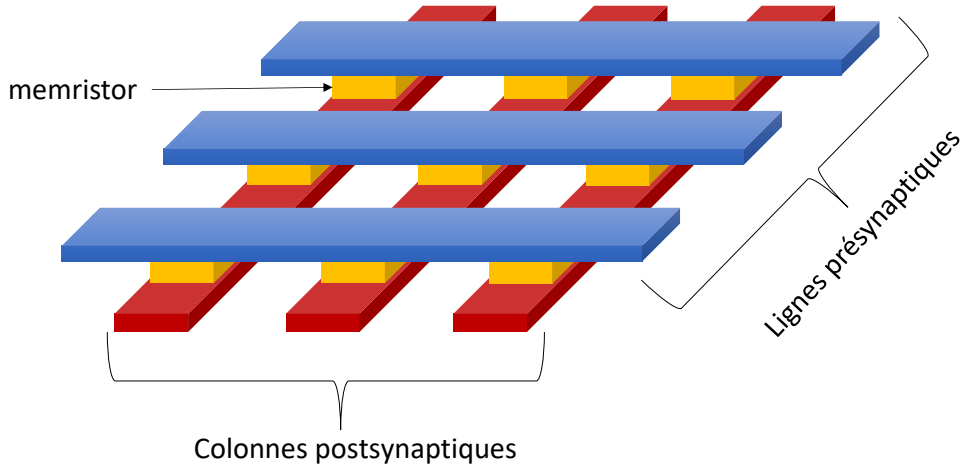


FIGURE 1.18 – Représentation schématique de la structure d'un crossbar de memristors. Il s'agit d'une représentation simplifiée. Dans les implémentations physiques, les lignes et les colonnes (pistes métalliques) se situent sur 2 couches au-dessus des memristors.

Sur la FIGURE 1.11 est illustrée la structure physique d'un memristor. Lorsque l'on s'intéresse à la réalisation matérielle d'une couche *all-to-all* d'un réseau de neurones événementiels la question de l'arrangement physique des memristors se pose lorsqu'ils sont utilisés comme synapses memristives. Les bornes du composant étant spatialement séparées (côte à côte), il est possible de déployer tous les memristors de la couche en les arrangeant physiquement sous forme de matrice. Il suffit alors de connecter électriquement sur une couche physique par des pistes métalliques les colonnes de la matrice correspondant aux mêmes connexions postsynaptiques (une piste par colonne correspond à un neurone postsynaptique) et sur une seconde couche physique les lignes de la matrice correspondant aux mêmes connexions présynaptiques (une piste par ligne correspond à un neurone présynaptique). Cet arrangement spécifique sous forme de matrice est appelé crossbar. Ce dernier se veut économe en termes de surface puisque tous les memristors de la couche sont regroupés spatialement, comme illustré sur la FIGURE 1.18. Cette structure présente néanmoins des inconvénients qui doivent être pris en compte lors de la réalisation du réseau et la conception du crossbar lui-même.

Les courants de *sneak path*

Si toute la matrice de memristors n'est pas fixée en termes de potentiels électriques, des courants dits de *sneak path*, qui varient en fonction de la valeur des memristors, apparaissent et influent sur les comportements des memristors observés à l'échelle du crossbar [93]. En effet, comme illustré sur la FIGURE 1.19, du courant peut circuler à travers les autres memristors

si leur potentiel électrique est laissé flottant. Il est donc primordial d'appliquer et de maintenir un potentiel sur toutes les connexions pour le réduire au maximum sans avoir à modifier la structure du crossbar. Il est également possible de s'en affranchir sans imposer un potentiel sur toute la matrice si des interrupteurs ou sélecteurs se trouvent en série de chaque memristor, mais cela complexifie la structure physique du crossbar [94].

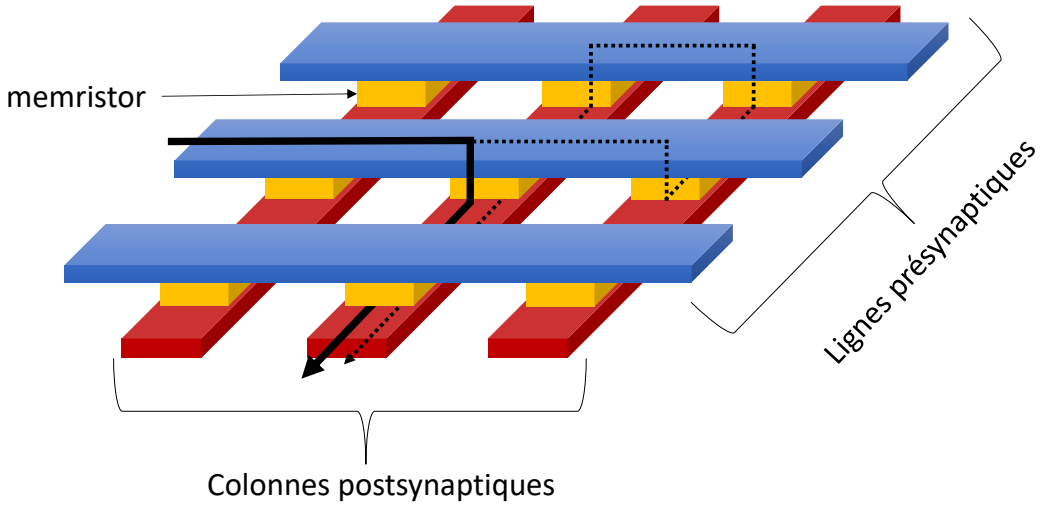


FIGURE 1.19 – Illustration d'un courant de *sneak path*.
Si on considère le courant (en noir) circulant dans le memristor lorsqu'on applique une tension sur la ligne 2 et la colonne 2, un courant de *sneak path* (en pointillé noir) peut apparaître si la ligne 2 et la colonne 2 sont connectées électriquement et que le reste du crossbar est laissé flottant.

Pour s'affranchir au maximum de ce phénomène lors de la mesure, il est donc nécessaire de maintenir un potentiel sur toutes les connexions et de faire une mesure différentielle avant et après application d'une tension de mesure pour s'affranchir des courants résiduels.

Limitation en courant

Puisque les memristors sont connectés en parallèle vu d'un neurone présynaptique (ou d'un neurone postsynaptique), la mise en parallèle d'un grand nombre de synapses memristives est limitée par la résistance équivalente qu'ils représentent. Si un trop grand nombre de composants memristifs est mis en parallèle, le courant nécessaire pour piloter toute la ligne peut être trop important. Par exemple, pour un R_{ON} des memristors (valeur la plus faible) de 100 kΩ la mise en parallèle de 1000 memristors résulte en une résistance équivalente pilotée par le neurone présynaptique de 100 Ω, ce qui à 1 V génère des courants trop importants (10 mA) pour une puce électronique intégrée pour une seule connexion synaptique.

Limitation de tenue de tension

Un dernier phénomène problématique est dû à la résistance non nulle des pistes métalliques de connexion. Plus de memristors sont connectés en parallèle, plus la chute de potentiel le long de la ligne est importante. Il est donc nécessaire lors de la fabrication du crossbar de s'assurer que la résistance entre memristors est suffisamment faible pour que le dernier memristor de la ligne ne se voie pas appliquer une tension trop atténuée.

Ainsi, si cette structure physique est très attrayante pour la conception d'un réseau de neurones événementiels, elle reste limitée en termes de dimensions à cause :

— des courants nécessaires pour la faire fonctionner ;

— des chutes de potentiel qui peuvent apparaître et ainsi provoquer la non-écriture (changement du poids) de certains memristors si les tensions de seuil ne sont pas atteintes.

1.3 Des données à traiter

Si jusqu'ici la structure des réseaux de neurones événementiels matériels étudiés a été abordée (neurones LIF et synapses memristives), la question se pose sur les données à présenter en entrée et que l'on souhaite traiter avec de telles architectures. S'inspirant de la biologie, ces réseaux de neurones événementiels (SNN) sont naturellement prédisposés pour traiter des données issues de capteurs qui se veulent bio-inspirés. Par exemple, il existe des capteurs de cochlée (banque de filtres) qui permettent de générer des événements à partir de signaux continus comme de l'audio et ainsi imiter une oreille biologique appelé *silicon cochleas* [95][96]. Cette méthode peut-être utilisée pour générer des événements à partir de divers signaux continus (audio, signaux électriques divers) afin de les rendre compatibles avec des SNN [97].

Il existe également un capteur appelé caméra événementielle, DVS (Dynamic Vision Sensor) ou encore rétine artificielle. Ce capteur se veut représentatif du comportement de l'œil biologique et des données qui sont générées à sa sortie. Une telle caméra événementielle a été développée pour la première fois par Mahowald en 1992 dans ses travaux de thèses [98][99]. Une implémentation matérielle de ce type de capteur a également été développée par Posch en 2014 [100] et dont le fonctionnement est illustré sur la FIGURE 1.20 et fonctionne de la manière suivante :

— Si aucun mouvement n'est perçu par la caméra, aucune donnée n'est émise en sortie.

— Si une variation d'intensité lumineuse positive est détectée par un pixel, il émet un événement positif (ON).

— Si une variation d'intensité lumineuse négative est détectée par un pixel, il émet un événement négatif (OFF).

— Les sorties (pixels) fonctionnent de manière asynchrone.

Ce comportement se veut similaire à celui de l'œil humain qui, de la même manière, émet des informations positives ou négatives suivant les variations de lumière qu'il détecte tout en fonctionnant de manière asynchrone. Ce comportement en biologie est rendu possible grâce aux cellules appelées ganglions. Ce type de caméra présente un avantage majeur par rapport aux caméras classiques. En effet, comme on peut le voir sur la FIGURE 1.20, la caméra effectue naturellement de la détection de contours. Elle présente également de grandes dynamiques de détection de lumière permettant, contrairement aux caméras classiques, d'être moins facilement ébloui. Selon sa configuration, elle peut accepter une dynamique de luminosité de 140dB contre 60dB pour une caméra traditionnelle [101]. Enfin, d'un point de vue consommation, elle se veut beaucoup moins énergivore qu'une caméra classique puisque seules les informations utiles [12] sont générées et transmises.

12. Là où la lumière varie.

Aujourd'hui le protocole utilisé par ce type de caméra est appelé AER (*Adress Event Representation*) [102]. Ce protocole représente de manière numérique l'adresse du pixel qui émet un événement, l'instant de l'impulsion et sa polarité (ON ou OFF).

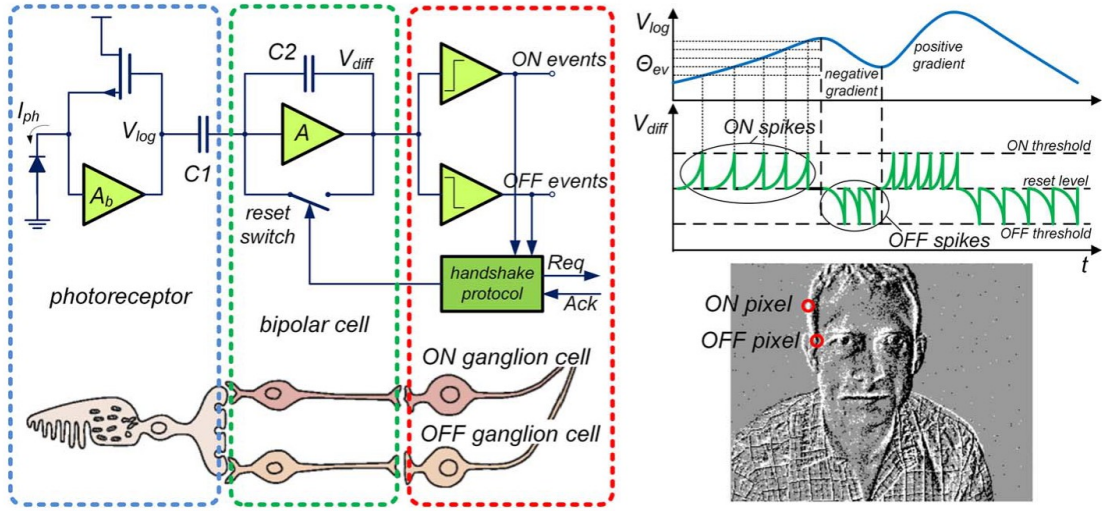


FIGURE 1.20 – Principe de fonctionnement d'une caméra événementielle.
Image tirée de [100]. À droite se trouve le schéma électrique synthétique d'un pixel d'une caméra événementielle avec les parties correspondantes en biologie. À gauche se trouve la réponse d'un pixel en fonction de la variation d'intensité lumineuse et l'image en sortie d'une caméra complète lorsqu'elle filme un être humain en mouvement.

1.3.1 N-MNIST : Des données issues d'une caméra événementielle

Une base de données générée par ce type de caméra événementielle a été créée en 2015 par Orchard et coll. [103]. Elle a été obtenue en suivant la méthode suivante. Les chiffres manuscrits de la base de données MNIST (chiffre de 0 à 9) ont été affichés devant une caméra événementielle effectuant 3 saccades successives de 100 ms chacune dans 3 directions différentes. Cette base de données comprend 60 000 échantillons dédiés à l'entraînement du réseau de neurones et 10 000 échantillons dédiés au test. Un exemple d'échantillon de cette base de données est illustré sur la FIGURE 1.21.

C'est cette base de données (N-MNIST) qui a été utilisée pour tous les travaux présentés dans le CHAPITRE 2.

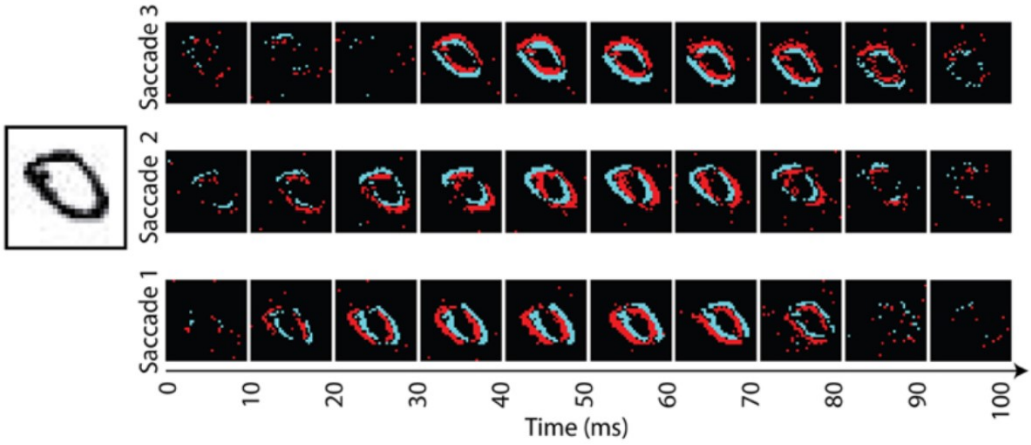


FIGURE 1.21 – Exemple de données de la N-MNIST.
Image adaptée de [103]. L'image à droite correspond au chiffre d'origine (MNIST) utilisé pour générer les données. À droite sont illustrées les données en sortie obtenues pour chaque saccade. En rouge sont représentés les événements ON et en bleu les événements OFF. Chaque image pour chaque saccade correspond à l'accumulation des événements en sortie de la caméra sur une fenêtre de 10 ms.

1.3.2 CPTC : Des données créées durant ces travaux de thèse

Les images d'origines

En plus de la base de données N-MNIST utilisée pour les travaux présentés dans le CHAPITRE 2, des données ont été créées manuellement en vue d'être utilisées par la plateforme développée dans le CHAPITRE 3. Cette base de données est constituée d'images dessinées à la main de 4 symboles différents (Croix, Plus, Triangle, Cercle) constituant la base de données nommée CPTC.

FIGURE 1.22 – Images issues de la base de données CPTC.

À partir de cette base de données CPTC, deux méthodes ont été définies pour générer des événements compatibles avec un SNN :

Génération d'événements à partir de niveaux de gris

Cette méthode génère à l'aide de scripts en Python des événements en encodant le niveau de gris de l'image d'origine ou de l'image bruitée à partir d'une conversion du niveau de gris en fréquence de tir d'événements d'après la méthode utilisée par Gwendal Lecerf en 2014 [104].

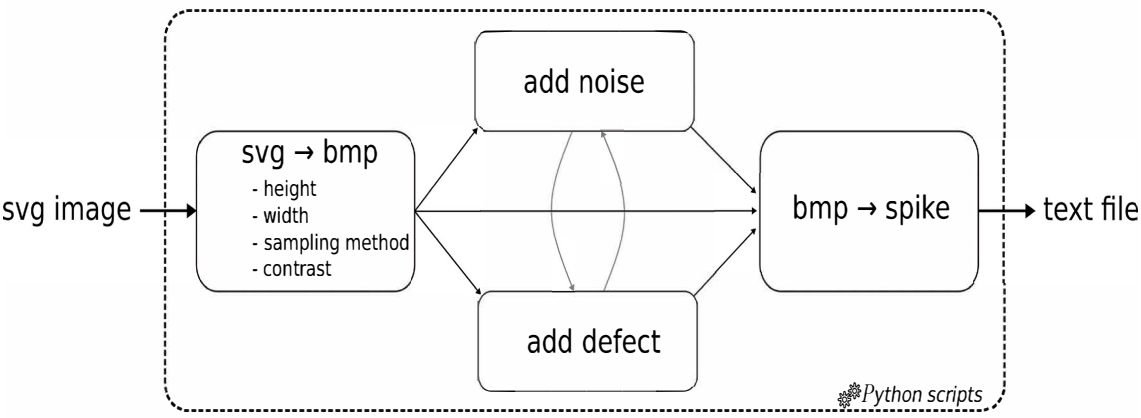


FIGURE 1.23 – Protocole de conversion d'images vers les données événementielles basé sur le niveau de gris (en anglais).

Émulation du comportement d'une caméra événementielle

Afin d'avoir des données similaires à ce que l'on peut obtenir avec une caméra événementielle un protocole basé sur l'utilisation de scripts Python et du logiciel Blender a été mis au point. Ce protocole permet via l'utilisation d'un logiciel 3D Blender d'avoir un contrôle total sur les mouvements des images ou objets, l'intensité lumineuse avec ou sans bruit. Cette méthode extrêmement flexible illustrée sur la FIGURE 1.24 nécessite néanmoins beaucoup plus de temps que la méthode précédente pour générer les données en sortie.

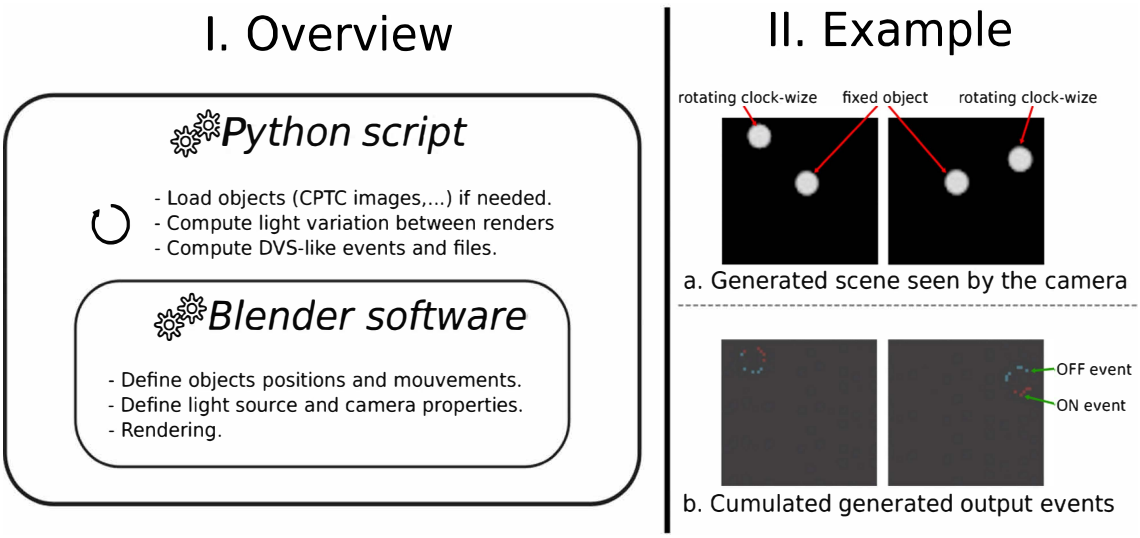


FIGURE 1.24 – Protocole de conversion d'images vers des données événementielles basées sur la variation d'intensité lumineuse (en anglais).

1.4 Conclusion

Dans ce chapitre ont été introduits les réseaux de neurones événementiels, architectures qui se veulent inspirées de la biologie pour tenter de réaliser des structures d'apprentissage consommant peu d'énergie en raison de leur caractère événementiel. Ce sont les réseaux de neurones événementiels à couche unique *all-to-all* et à synapses memristives qui sont étudiés dans ces travaux de thèse. Ces réseaux sont constitués d'une couche d'entrée émettant des événements correspondant aux informations que l'on souhaite traiter et de neurones postsynaptiques qui sont connectés aux entrées par l'intermédiaire de synapses memristives. L'intérêt des memristors ferroélectriques comme synapses artificielles s'explique par leur caractère analogique permettant une large gamme de poids synaptiques tout en assurant une faible consommation énergétique grâce à leur niveau de résistance élevée. La structure physique utilisée pour réaliser la matrice de connexion des neurones postsynaptique aux entrées (crossbar de memristor) est très avantageuse d'un point de vue de l'intégration physique, mais présente des limitations en termes de nombre et nécessite d'être complètement polarisée pour fonctionner correctement. Les neurones postsynaptiques sont une structure critique pour la réalisation de réseaux de neurones puisqu'ils constituent la partie neurones de l'architecture et utilisent le modèle LIF qui se veut simple à implémenter et à simuler. La connexion entre ces neurones postsynaptiques et les synapses memristives est assurée par un CCII. Enfin, plusieurs bases de données ont été considérées et/ou utilisées pour ces travaux de thèse : celle issue d'une caméra événementielle (N-MNIST) qui se veut similaire à ce que l'œil humain est capable de faire; celle générée par une méthode d'encodage fréquentiel de niveau de gris d'images contenant des symboles; celle générée par l'émulation logicielle d'une caméra événementielle permettant une grande flexibilité sur les données que l'on peut générer.

Chapitre 2

Simulations de réseaux de neurones événementiels en vue d'implantations matérielles

Ce chapitre s'inscrit dans le premier axe de ces travaux de thèse. Il s'agit de l'étude par la simulation de réseaux de neurones événementiels et de leurs règles d'apprentissage en vue d'une implantation matérielle. L'objectif est d'arriver à définir une architecture et un ensemble de règles d'apprentissage qui se veulent simples et faciles à implémenter pour être potentiellement peu coûteuses en énergie de par leur simplicité. La première partie de ce chapitre détaillera le fonctionnement de l'architecture choisie, architecture combinant contrôle numérique et calcul analogique, ainsi que du simulateur utilisé pour étudier son comportement et qui a été mis au point lors de ces travaux de thèse. Cette architecture mixte [1], réseau événementiel *all-to-all* à une seule couche, sera simulée pour étudier et définir différentes règles d'apprentissage qui seront abordées dans la partie 2. Dans cette deuxième partie, plusieurs règles d'apprentissage non supervisées sont étudiées pour définir quelles sont les conditions nécessaires à leur fonctionnement. Y seront également présentées des règles d'apprentissage faiblement supervisées qui se veulent simples à implémenter. Enfin, la dernière partie s'attardera sur l'étude de l'architecture mixte qui a été implémentée dans le cadre du projet ULPEC où les simulations réalisées ont permis de définir un certain nombre de paramètres des circuits électroniques qui constituent le réseau et de comportements à prendre en compte lors de son utilisation.

2.1 Architecture et simulateur

L'architecture matérielle qui a été définie pour ces travaux de thèse est illustrée sur la FIGURE 2.1. On y retrouve les éléments suivant :

— Une caméra événementielle qui génère les événements à injecter dans le réseau de neurones. Dans le simulateur cette partie est remplacée par une base de données qui, pour les simulations présentées dans ce chapitre, est issue de la N-MNIST [2].

— La partie analogique du réseau de neurones événementiels *all-to-all*. Les neurones postsynaptiques sont des neurones de type LIF à courant de décharge continu et les neurones présynaptiques de la couche d'entrée correspondent à des générateurs de tensions. Les synapses correspondent à un crossbar de memristors qui est représenté dans le simulateur par une matrice de dimension $N_{\text{pre}}N_{\text{post}}$ [3].

— Le circuit numérique de contrôle *Digital Control Block* (DCB). Ce circuit numérique de contrôle, dont les fonctionnalités ont été définies pendant ces travaux de thèse et dont les détails seront abordés par la suite, sert à piloter la partie analogique du réseau que ce soit pour le déclenchement des formes d'ondes présynaptiques ou postsynaptiques (stimulation du réseau depuis les données issues de la caméra ou modification des poids

1. Une architecture mixte combine analogique et numérique pour son fonctionnement.
2. Seule une partie des données pour chaque échantillon est utilisée.
3. Cela se base sur l'hypothèse que la résistance des pistes connectant les memristors entre eux est négligeable et ne cause donc qu'une chute de tension négligeable sur les lignes ou colones du crossbar.

synaptiques), la récolte des événements postsynaptiques ou bien la mise à zéro des tensions de membrane des neurones postsynaptiques. C'est également ce bloc DCB qui implémente les règles d'apprentissage du réseau de neurones.

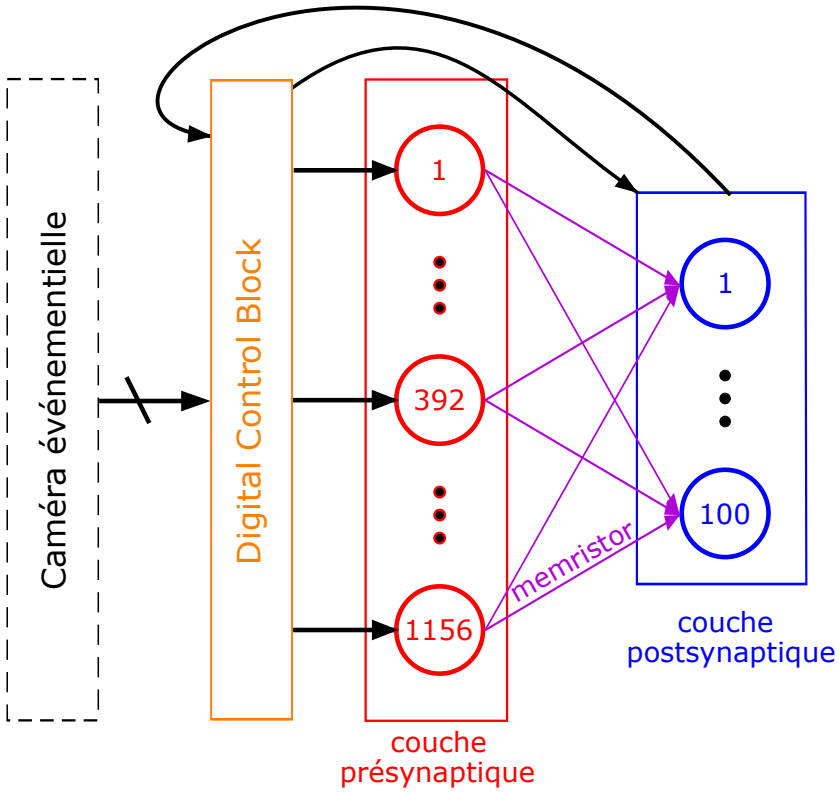


FIGURE 2.1 – Architecture matérielle du réseau de neurones événementiels étudié en simulation.
Il s'agit d'un réseau 1156 × 100. Les 1156 entrées correspondent au nombre de pixels disponible dans la N-MNIST (échantillon de 34 × 34 pixels). Le DCB a la charge de piloter la partie SNN à partir des événements issus de la base de donnée et des activitées postsynaptiques perçues.

2.1.1 Fonctionnement du réseau de neurones événementiel

Afin de comprendre le fonctionnement du simulateur et les résultats qui en découlent, il est nécessaire d'expliquer le fonctionnement de l'architecture matérielle que l'on cherche à étudier. Nous nous intéresserons surtout au fonctionnement d'un seul neurone postsynaptique puisque, pour l'architecture étudiée (*all-to-all*), tous les neurones postsynaptiques fonctionnent en parallèle et de la même manière. On s'intéresse en particulier à la situation illustrée sur la FIGURE 2.2 où 2 entrées présynaptiques (pre1 et pre2) sur les 1156 du réseau (voir FIGURE 2.1) sont actives.

Le fonctionnement d'un neurone postsynaptique du réseau se différencie en deux modes distincts [4] :

— Le mode **entraînement** : Des données sont présentées en entrée et lorsqu'un événement postsynaptique est émis il y a une phase d'apprentissage qui se déclenche et qui modifie les poids du réseau (conductances des memristors) suivant une règle d'apprentissage donnée.

— Le mode **test** : Ce mode consiste à présenter des données en entrée et observer la réponse en sortie sans phase d'apprentissage pour déterminer ce que le neurone a appris ou reconnaît. Dans ce mode, les poids qui lui sont connectés sont alors figés. C'est ce mode qui permet de déterminer les performances du réseau dans les simulations qui seront présentées dans ce chapitre.

Pour ces 2 modes, la phase où sont présentées les données afin de stimuler le neurone postsynaptique jusqu'à une réponse (émission d'un événement) est appelée **inférence**. La phase de modification des poids quant à elle est appelée **apprentissage**. Il faut noter que pour certaines données présentées, l'inférence peut résulter en une absence de réponse des neurones postsynaptiques du réseau.

4. Il est possible d'avoir un seul mode de fonctionnement : un mode entraînement permanent du neurone pour pouvoir s'adapter aux données présentées en entrée en continu.

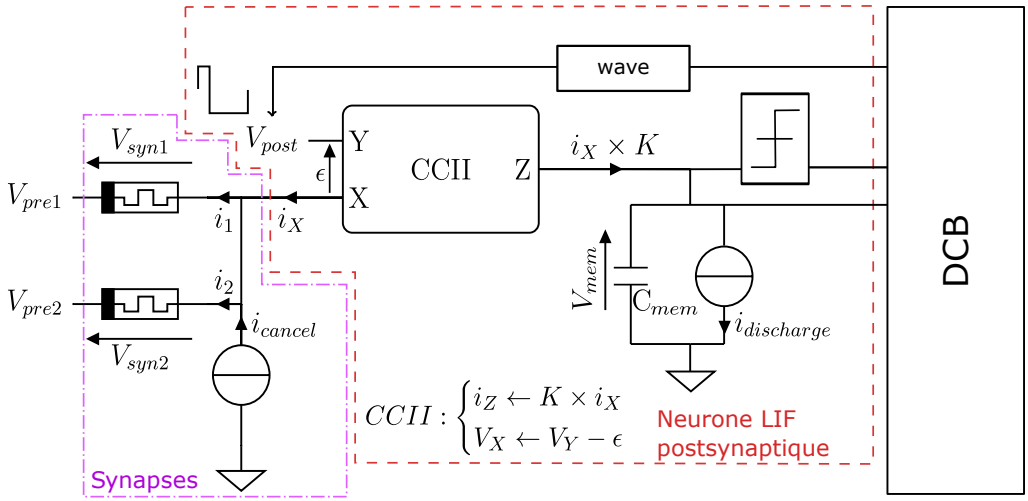


FIGURE 2.2 – Structure d'un neurone postsynaptique LIF à courant de fuite avec 2 synapses actives.
Deux memristors sont considérés comme actifs avec des tensions de stimulation V_{pre1} et V_{pre2} négatives. Le convoyeur de courant représenté ici copie en sortie le courant d'entrée avec un facteur de copie K uniquement si ce dernier est positif pour ne pas causer d'inhibition de la membrane (phénomène détaillé par la suite). Le courant i_{cancel} correspond au courant généré par les memristors inactifs connectés à ce neurone postsynaptique. Ce courant i_{cancel} est causé par l'*offset* ϵ de copie en tension du CCII+ (s'il n'est pas nul par rapport à la tension de repos des memristors). « wave » correspond au générateur de la forme d'onde analogique postsynaptique. La connexion du DCB à la membrane représente le système de mise à zéro de cette dernière.

Les simulations utilisent le modèle de neurone LIF à courant de décharge car on suppose que l'utilisation de transistors pour fixer un courant très faible de décharge prend moins de surface qu'une résistance passive lors d'une implantation matérielle dans une puce électronique. De plus le démonstrateur du projet européen ULPEC implémente ce modèle.

Pour réaliser de l'inférence sur cette architecture de calcul analogique, il faut générer des courants de stimulation pour charger les neurones postsynaptiques. Pour cela, les entrées présynaptiques doivent appliquer des formes d'ondes (V_{pre}) sur les bornes présynaptiques des synapses correspondantes lorsqu'un événement en entrée arrive depuis la caméra événementielle (ou de la base de données). Dans la partie 1.2.2 Le choix des synapses : Le memristor ferroélectrique, un nanocomposant du CHAPITRE 1, une forme d'onde composée d'une partie impulsion positive (pour l'écriture) et rampe négative en tension (stimulation postsynaptique, pondération temporelle de la modification du poids) a été présentée. Le choix d'une rampe dans la forme d'onde utilisée se justifiait alors par la volonté de moduler la modification du poids par modulation de l'amplitude de l'impulsion d'écriture.

Pour les travaux présentés dans ce manuscrit, le choix s'est porté sur le deuxième phénomène de modification du poids des memristors ferroélectriques [5] qui utilise l'application d'impulsions de même amplitude répétées dans le temps. La modification progressive du poids est alors rendue possible par la répétition de l'application de l'impulsion d'écriture et non par une variation d'amplitude. Ainsi, au lieu d'utiliser une rampe dans la forme d'onde utilisée pour stimuler le neurone postsynaptique, nous utilisons une impulsion de même signe que cette dernière (négative) et de mêmes largeurs T_{LTP} [6]. Cette nouvelle forme d'onde et l'effet qu'elle va avoir lorsque le courant qu'elle génère à travers les synapses est copié puis injecté dans la membrane du neurone sont illustrés sur la FIGURE 2.3 pour différentes versions de la copie en courant (complète, courant positif uniquement, courant négatif uniquement). On constate que la présence de l'impulsion positive (utilisé pour la programmation) lors de la stimulation provoque soit des décharges de la membrane soit des manques de charges (aucun courant injecté). Ces phénomènes qui empêchent ou réduisent la charge de la membrane ont donc un impact négatif qui peut devenir critique si un grand nombre d'impulsions présynaptiques arrivent proches dans le temps (augmentation de la décharge ou de l'absence de charge).

La solution idéale pour que la forme présynaptique soit entièrement utilisée pour la stimulation du neurone consiste à supprimer la partie "impulsion positive" lors de l'inférence. On sépare alors en deux signaux distincts la forme d'onde que peut générer le signal présynaptique et on les utilise séparément de la manière suivante :

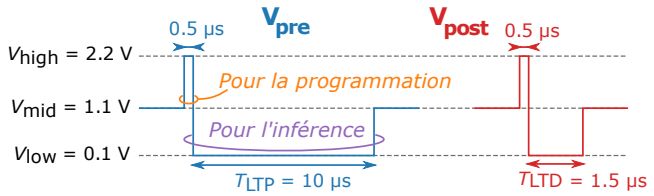
— Pour l'inférence, on utilise une impulsion d'amplitude négative qui permet de charger la membrane sans perte pour la version du convoyeur de courant choisie (copie du courant i_x positif uniquement).

— Pour renforcer le poids du memristor lors d'un apprentissage, on utilise cette même impulsion d'amplitude négative qui permet de dépasser le seuil d'écriture négatif en se combinant avec la forme d'onde postsynaptique.

— Pour déprécier le poids du memristor lors d'un apprentissage, on utilise l'impulsion d'amplitude positive que l'on déclenche lorsque l'impulsion postsynaptique applique la partie négative de sa forme d'onde pour permettre ainsi de dépasser le seuil d'écriture négatif.

Le déclenchement de la bonne forme d'onde du neurone présynaptique (impulsion positive ou négative) au bon moment (pour la combiner avec la tension postsynaptique) est assuré par le DCB et dépend de la règle d'apprentissage implémentée.

5. Présenté dans la partie 1.2.2 Le choix des synapses : Le memristor ferroélectrique, un nanocomposant.
6. Tout comme avec la rampe, il s'agit de la fenêtre d'apprentissage pour une règle de type STDP classique.



V_{pre} Pour l'inférence	bipolaire	bipolaire	bipolaire	$V_{pre} < 0$
Courant copié	i_X	$i_X < 0$	$i_X > 0$	$i_X > 0$

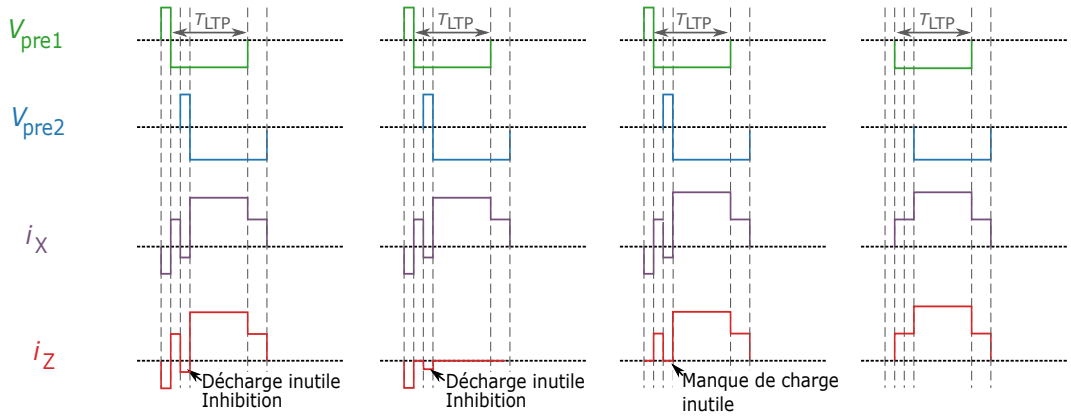


FIGURE 2.3 – Impact de la forme d'onde choisie pour l'inférence pour différentes copies du CCII.

Au-dessus du tableau sont représentées les formes présynaptiques et postsynaptiques composées d'une partie impulsion positive (pour modifier son poids qui dure ici 0.5 µs) et négative (pour les fenêtres d'apprentissage T_{LTP} et pour l'inférence dans le cas de V_{pre}). Le tableau représente l'impact de la forme d'onde d'inférence choisie sur le courant copié pour différentes versions de la copie en courant du CCII (complète, positive ou négative). V_{mid} correspond à la tension de polarisation du crossbar. La version qui copie le courant positif uniquement est celle qui permet de s'affranchir de courant d'inhibition et est donc préconisée pour l'implantation matérielle. La suppression de la partie impulsion positive permet de faire disparaître les manques de charge ou décharge de la membrane. Les durées utilisées sont liées aux contraintes matérielles du système ULPEC, mais restent néanmoins représentatives de la plupart des approches analogiques.

Phase d'apprentissage

Lorsqu'un neurone postsynaptique émet un événement, ce dernier est récolté par le DCB qui va réinitialiser les tensions de membrane selon la stratégie dite *Winner Take All* (WTA, aussi appelé inhibition latérale) utilisée dans plusieurs architectures de SNN [105][106]. Dans les SNN étudiés ici, cette règle WTA permet de ne conserver que le premier neurone qui tire et donc « gagne » lors d'une inférence (apprentissage ou test). Cette mise à zéro des membranes est maintenue jusqu'à la fin de l'apprentissage et c'est le neurone postsynaptique « gagnant » qui verra ses poids synaptiques afférents être modifiés lors de l'apprentissage. Le DCB qui détermine quel neurone postsynaptique a « gagné » est un bloc numérique qui fonctionne à une fréquence d'horloge de période T_{clk} et qui effectue toutes ses opérations sur un des deux fronts de l'horloge (montant ou descendant). Dans le cas où plusieurs neurones postsynaptiques émettent un événement au milieu de la période de l'horloge de fonctionnement du DCB (en dehors du front d'horloge de fonctionnement), un mécanisme d'arbitrage sélectionne le neurone postsynaptique avec l'index (numéro du neurone) le plus faible pour ne garder qu'un seul neurone postsynaptique pour l'apprentissage. Ce choix de garder celui avec l'index le plus faible s'est justifié par la simplicité du circuit numérique correspondant pour le déterminer afin de ne garder qu'un seul neurone postsynaptique pour la phase d'apprentissage. Une fois le neurone « gagnant » déterminé, le DCB déclenche les formes d'ondes présynaptiques et postsynaptiques nécessaires à la modification des poids synaptiques qui lui sont connectés suivant la règle d'apprentissage globale[7] qu'il implémente. La modification opérée sur les memristors qui dépend de la tension qui leur est appliquée suit une règle locale[8] qui est décrite par l'équation (2.1). Cette équation (2.1) est un modèle simplifié utilisé en simulation pour décrire l'évolution des poids synaptiques. Ce choix d'utiliser ce modèle synthétique s'est justifié par :

— Le manque d'informations sur le comportement des memristors ferroélectriques du démonstrateur ULPEC.

— La volonté d'avoir un temps de simulation réduit comparé à l'utilisation de modèles plus complexes (nécessitant plus d'opérations pour le calcul).

— L'idée que les caractéristiques précises de la règle d'apprentissage locale (STDP) des synapses memristives ne devrait avoir qu'un impact limité de par la résilience à la variabilité synaptique ou formes de STDP d'après certains travaux [107], l'intérêt de ces travaux portant sur les règles d'apprentissage globales (à l'échelle du réseau) plus que locales (à l'échelle de la synapse).

7. Ici une règle globale correspond à la règle de modification de l'ensemble des poids synaptiques connectés à un neurone postsynaptique.
8. À l'échelle du memristor. Équation de modification du poids.

$$\Delta G = \begin{cases} +A^{+} \times (G_{\max} - G_0) & \text{pour } V_{\text{syn}} \leq V_{\text{pot}} \\ -A^{-} \times (G_0 - G_{\min}) & \text{pour } V_{\text{syn}} \geq V_{\text{dep}}\,, \\ 0 & \text{sinon} \end{cases} \quad \text{avec } V_{pot} = -1.2\,\text{V et } V_{dep} = 1.2\,\text{V} \tag{2.1}$$

Ces équations font intervenir un taux d'apprentissage de potentiation (A^{+}) ou de dépression (A^{-}) pour des conductances synaptiques comprises entre $G_{\min}$ (poids faible ou OFF) et $G_{\max}$ (poids fort ou ON). V_{syn} correspond à la tension aux bornes du memristor, V_{pot} le seuil de potentiation du memristor et V_{dep} le seuil de dépression du memristor. G_0 correspondant au poids avant apprentissage et la variation ΔG obtenu avec ces équations correspond à la variation calculée issue d'une fraction de la variation maximale possible (restant à parcourir) avant d'atteindre $G_{\max}$ (si dépression) ou $G_{\min}$ (si potentiation). Cette règle d'apprentissage locale découle de la modification progressive d'un memristor par application d'impulsion répétée de même amplitude. Si un même neurone postsynaptique déclenche sa phase d'apprentissage N fois et que l'une de ses synapses afférentes se fait renforcer (ou déprécier) à chaque fois, la même impulsion de modification du poids sera appliquée (N fois), induisant ainsi une modification progressive du poids à chaque fois que le neurone postsynaptique apprend.

Une fois la modification des synapses effectuée, la phase d'apprentissage se termine. On peut alors réactiver les membranes des neurones postsynaptiques du réseau sauf celle du neurone qui vient d'effectuer un apprentissage. En effet durant ces travaux de thèse a été introduit la notion de compteur réfractaire qui maintient à zéro la membrane d'un neurone qui vient de tirer pendant N_{refrac} tirs provenant d'autres neurones postsynaptiques. Le but de ce compteur réfractaire, qui bloque pendant un certain nombre d'activités des neurones environnants le neurone qui vient d'apprendre, est de donner plus de chance aux autres neurones qui n'ont pas encore tiré d'apprendre pour éviter qu'un seul neurone postsynaptique phagocyte toute l'activité du SNN. Cette phase de repos s'inspire du phénomène de période réfractaire que l'on peut observer en biologie pendant laquelle un neurone qui vient d'émettre un événement devient insensible aux stimulations qu'il perçoit pendant une certaine période. Si les travaux précédents comme ceux de Lecerf [104] implémentent une période temporelle réfractaire (le neurone est inactif pendant une période T_{refrac}) l'utilisation d'un compteur réfractaire basé sur la réponse des autres neurones de la couche postsynaptique permet d'être moins sensible à la vitesse des réponses que peut avoir le réseau suivant les données qui lui sont présentées puisque tant que au moins N_{refrac} autres neurones postsynaptiques n'ont pas émis un évènement, celui qui est dans sa période réfractaire est insensible aux stimulations et à leur vitesse. Ce compteur réfractaire utilisé lors de l'apprentissage est implémenté dans le DCB.

Concernant les données présentées en entrée, dans le cas de l'utilisation d'échantillons pro-

venant d'une base de données, lorsqu'un neurone postsynaptique a émis une activité et que sa phase d'apprentissage est terminée, même s'il reste des informations dans l'échantillon, l'échantillon suivant est présenté sans finir de présenter les données restantes. En effet, dans les travaux présentés dans ce manuscrit, la détection/reconnaissance par le SNN des données en entrée est assurée par le premier neurone postsynaptique qui tire. Ainsi, une fois qu'un neurone postsynaptique a émis un événement, le SNN a supposément reconnu un motif et il n'y a pas besoin de continuer de présenter le reste des données de l'échantillon en cours. Dans le cas où aucun neurone n'a tiré à la fin de la présentation d'un échantillon (aucun motif reconnu par le SNN), l'échantillon suivant est présenté sans réaliser d'apprentissage après avoir remis à zéro les membranes postsynaptiques.

Étiquetage et test

Lorsqu'on définit un réseau de neurones, la question des performances est primordiale pour vérifier s'il a appris à reconnaître les données qui lui ont été présentées. La caractéristique utilisée pour déterminer les performances d'un système qui apprend est appelée **taux de reconnaissance** ou RR [9]. La méthode utilisée pour déterminer ce RR consiste à inférer des données de test (non utilisées pendant l'apprentissage [10]) et à relever les réponses du réseau de neurones. Lorsque le réseau de neurones apprend, il spécialise ses sorties pour reconnaître et différencier les différents types d'échantillons présentés en entrée que l'on appelle **classe**. La classe de réponse d'un neurone de sortie (sur laquelle il s'est spécialisé), appelée **label**, peut être prédéterminée (imposée) avant apprentissage dans le cas de l'apprentissage supervisé ou inconnue avant apprentissage dans le cas de l'apprentissage non supervisé. Le taux de reconnaissance est déterminé à partir de ces labels avec l'équation (2.2) où le nombre de réponses correctes correspond au nombre de tirs en sortie du réseau sur des neurones dont le label correspond à la classe de l'échantillon présenté en entrée. Le nombre N_{test} correspond au nombre d'échantillons total présentés lors de la phase de test. Pour savoir si un réseau a appris, on considère un niveau de reconnaissance $RR = \frac{1}{N_{\text{classes}}}$ qui correspond à une situation où les neurones de sortie répondent de manière aléatoire aux échantillons présentés en entrée en supposant que les classes sont équivalentes. Plus le RR obtenu est supérieur à ce niveau, plus le réseau de neurones est performant.

$$RR = \frac{\text{nombre de réponses correctes}}{N_{\text{test}}} \tag{2.2}$$

Afin de déterminer le taux de reconnaissance, il est donc nécessaire de déterminer à quelle classe un neurone postsynaptique est devenu sensible : son **label**. Pour déterminer le label des neurones de sortie pour ces travaux de thèse, nous utilisons une méthode qui consiste à prendre les dernières réponses d'un neurone lors de l'apprentissage (dans ces travaux nous

9. De l'anglais *recognition rate*.

10. Utiliser des données qui n'ont jamais été présentées permet de s'assurer que le réseau n'a pas surappris et qu'il peut reconnaître des échantillons qu'il n'a jamais vus.

nous intéressons aux 20 dernières réponses [11]) et de comparer la représentation de chacune des classes utilisées lors de l'apprentissage. Cette méthode cherche à savoir à quelle classe un neurone a le plus été sensible à la fin de la phase d'apprentissage au bout de laquelle il est censé avoir appris et s'être spécialisé. Pour cela, on calcule une valeur de test $L = 2 \times \frac{1}{N_{\text{classes}}}$. Si, pour la classe en question, le neurone a émis le plus d'événements sur ses dernières réponses et que le taux de tirs pour cette classe est supérieur à L, cette classe est utilisée comme label du neurone de sortie. Sinon, le neurone reste sans classe et est désactivé [12]. Dans le cas où, pour un neurone de sortie, plusieurs classes ont causé le même nombre de tirs du neurone et que ce taux de tir est supérieur à L, la classe la moins élevée est choisie. Par exemple pour la N-MNIST (classe = chiffre de 0 à 9) :

— Pour $N_{\text{classes}} = 10$ ($L = 0.2$), si un neurone sur ses 20 derniers événements a tiré 8 fois pour un 5 (→ taux de 0.4), 8 fois pour un 6 (→ 0.4) et 4 fois pour un 3 (→ 0.2), la classe choisie comme label est le 5.

— Pour $N_{\text{classes}} = 3$ ($L = 2/3$; chiffres 5, 4 et 3 uniquement), si un neurone sur ses 20 derniers événements a tiré 8 fois pour un 5 (→ 0.4), 8 fois pour un 6 (→ 0.4) et 4 fois pour un 3 (→ 0.2), le neurone n'a pas de label.

— Pour $N_{\text{classes}} = 10$ ($L = 0.2$), si un neurone sur ses 20 derniers événements a tiré 15 fois pour un 7 (→ 0.75), 2 fois pour un 2 (→ 0.1) et 1 fois pour un 3 (→ 0.05), la classe choisie comme label est le 7.

— Pour $N_{\text{classes}} = 10$ ($L = 0.2$), si un neurone a tiré moins de 20 fois, le neurone n'a pas de label.

Une autre méthode possible d'étiquetage des neurones de sortie a été envisagée puis abandonnée à cause de son coût en termes de temps (en simulation) et d'énergie (si implémentation matérielle). Elle consiste à présenter tous les échantillons utilisés lors de la phase d'apprentissage, mais sans modifier les poids synaptiques. Une fois l'inférence de toute la base de données d'apprentissage réalisée, il suffit de choisir pour chaque neurone postsynaptique quelle classe d'échantillon a le plus fait tirer ce neurone postsynaptique. Le problème de cette méthode et qu'il est nécessaire de refaire passer toute la base d'entraînement ce qui consommera plus d'énergie et prend plus de temps (presque autant que lors de la phase apprentissage en simulation) même s'il n'y a pas de modification des poids synaptiques.

Paramètres par défaut

Maintenant que les principes de base de l'architecture que l'on souhaite étudier ont été abordés, se pose la question des paramètres qui seront choisis pour son étude. Ces paramètres sont

11. Ce choix du chiffre 20 est arbitraire et a été fait de manière empirique, mais il est considéré comme suffisant d'après les résultats obtenus.
12. Sa tension de membrane est maintenue à 0 V.

récapitulés TABLE 2.1 et seront utilisés pour les différentes simulations présentées dans ce chapitre sauf mention contraire explicite.

Les paramètres qui dépendent de la caméra événementielle sont N_{inputs} (dimension des données en entrée), N_{classes}, N_{train} et N_{test}. Si tous les échantillons de la base de données N-MNIST sont utilisés ($N_{\text{train}} = 60\,000$ et $N_{\text{test}} = 10\,000$), seulement une sous-partie des données qu'ils contiennent est utilisée dans le simulateur. Il s'agit de la première saccade de chaque échantillon et pour cette première saccade, seulement les événements correspondant à des activités ON de la caméra événementielle sont utilisés. Cela permet de réduire drastiquement le nombre de points utilisés et ainsi le temps de calcul. De plus si on considère que chaque saccade est assimilable à une classe tout comme chaque polarité d'événements (ON ou OFF) cela réduit également le nombre de classes à apprendre (de 30×2 classes [13] à seulement 10).

Les paramètres liés aux synapses (memristors) sont A^-, A^+, G_{min}, G_{max}, V_{pot}, V_{dep} et $\Delta\nu_{\text{stim}}$. La valeur $\Delta\nu_{\text{stim}}$ a été choisie pour rester inférieure à la valeur absolue de V_{pot} (signe opposé dû aux normalisations) et ainsi ne pas causer d'écriture pendant l'inférence. Au démarrage des simulations, toutes les conductances des synapses sont tirées aléatoirement suivant une loi uniforme entre les valeurs G_{max} et G_{min}.

Les paramètres liés aux neurones postsynaptiques sont N_{refrac}, ν_{seuil}, $i_{\text{discharge}}$, C_{mem} et K. Le choix de N_{refrac} est arbitraire et a été déterminé par rapport au nombre de sorties $N_{\text{outputs}} \times 10\%$. La capacité de membrane C_{mem} a été choisie pour être de taille raisonnable dans un circuit intégré (plus sa valeur augmente, plus elle nécessite de surface pour son implantation matérielle). Le courant de décharge $i_{\text{discharge}}$ a été choisi ($100\,\text{V}\cdot\text{s}^{-1}$) pour que la durée de la décharge soit de l'ordre de la dynamique des données de la N-MNIST (une saccade prend 100 ms). La valeur du facteur de copie en courant K des CCII a été estimée par rapport aux stimulations que doivent percevoir les neurones. Dans le cas de la N-MNIST, il y a environ 100 pixels qui sont éclairés dans les données utilisées pour représenter le chiffre (contre 1156 pixels de la caméra). Si tous ces chiffres s'activent en même temps, la variation de la tension de membrane est donnée par l'équation (2.3).

$$\Delta V_{\text{mem}} = \frac{K \times i_x \times \Delta T}{C_{\text{mem}}} \qquad (2.3)$$
$$i_x = \sum G_{\text{memristor actifs}} \times \Delta\nu_{\text{stim}}$$

Dans le cas extrême où les 100 memristors sont stimulés en même temps (ou que la décharge est nulle), $\Delta T = T_{LTP} = 10\,\mu\text{s}$, on obtient $\Delta V = 10\,\text{V}$. Ce calcul (extrême) ne prend pas en compte la répartition des événements qui en réalité n'arrivent pas en même temps, ainsi que la dé-

13. Il y a 10 classes de chiffres, chacun des échantillons contenant 3 saccades (mouvements de la caméra) et 2 types de polarité d'événement.

charge qui a lieu tout le long de la stimulation. Dans les simulations, la tension de seuil de déclenchement des neurones étant de 1 V, ces 10 V ne sont jamais atteints. Le facteur K de copie choisi correspond à 1/10 du cas extrême. Il s'agit d'une détermination présimulation qui n'a pas donc pas vocation à être extrêmement précise.

TABLE 2.1 – Paramètres de référence pour les simulations.

Paramètre	Valeur	Description
N_{inputs}	1156	Nombre d'entrées présynaptiques (34×34)
N_{outputs}	100	Nombre de neurones postsynaptiques
N_{classes}	10	Nombre de classes (pour la N-MNIST)
N_{train}	60 000	Nombre d'échantillons pour l'apprentissage (c.-à-d. 1 epoch)
N_{test}	10 000	Nombre d'échantillons pour la phase de test
V_{seuil}	1 V	Tension de seuil du neurone LIF
K	0.01	Facteur de copie en courant du CCII
C_{mem}	1 pF	Capacité de membrane
N_{refrac}	10	Seuil du compteur réfractaire pour réactivation d'un neurone
$i_{\text{discharge}}$	100 pA	Courant de décharge de la membrane
Δv_{stim}	1 V	Valeur de la tension d'inférence (tension appliquée sur les synapses)
V_{pot}	−1.2 V	Tension de seuil pour l'augmentation du poids
V_{dep}	1.2 V	Tension de seuil pour la diminution du poids
T_{LTP}	10 µs	Largeur de l'impulsion d'inférence
A^+	0.05	Facteur d'augmentation du poids synaptique
A^-	0.05	Facteur de diminution du poids synaptique
G_{max}	1 µS	Conductance maximum des synapses
G_{min}	10 nS	Conductance minimum des synapses
T_{clk}	1 µs	Fréquence d'horloge de l'arbitreur d'événements

2.1.2 Méthodes de calcul du simulateur

La théorie et les paramètres de base de l'architecture que l'on souhaite étudier ayant été abordés (notions d'apprentissage et d'inférence, paramètres de référence...), se pose maintenant la question du calcul qui va être réalisée dans le simulateur développé lors de ces travaux afin d'étudier cette structure.

Représentation des événements et points de calculs

La première question à se poser est de savoir quels vont être les points de calcul dans le simulateur. Ces points doivent permettre de calculer la tension de membrane et, si elle est supérieure au seuil, déterminer le neurone qui a répondu pour effectuer le calcul des nouveaux poids synaptiques pour l'apprentissage ou le calcul des performances pour le test. La méthode de calcul choisie et qui a été implémentée dans le simulateur est illustrée sur la FIGURE 2.4. Chaque point de la base de données d'entrée est dupliqué pour obtenir deux points de calcul, le premier correspondant à l'instant t_1 de déclenchement et le deuxième à l'instant $t_{1+T_{\text{LTP}}}$ de

fin d'application de la forme d'onde. Ils sont représentés comme illustré (2.4) où « t » correspond à l'instant de l'événement, « FLAG » correspond à l'activation de la forme d'onde (ON : devient active, OFF : devient inactive) et « index neurone » indique l'entrée présynaptique qui a émis cet événement.

$$[t, \text{FLAG}, \text{index neurone}] \tag{2.4}$$

Dans le simulateur, un vecteur contenant l'information « FLAG » de toutes les entrées permet de savoir quelles entrées sont actives pour les calculs. Lorsque la tension de membrane d'un ou plusieurs neurones de sortie dépasse le seuil, l'instant exact de déclenchement est calculé pour déterminer lequel a tiré en premier et faire un arbitrage pour sélectionner le neurone « gagnant » si nécessaire.

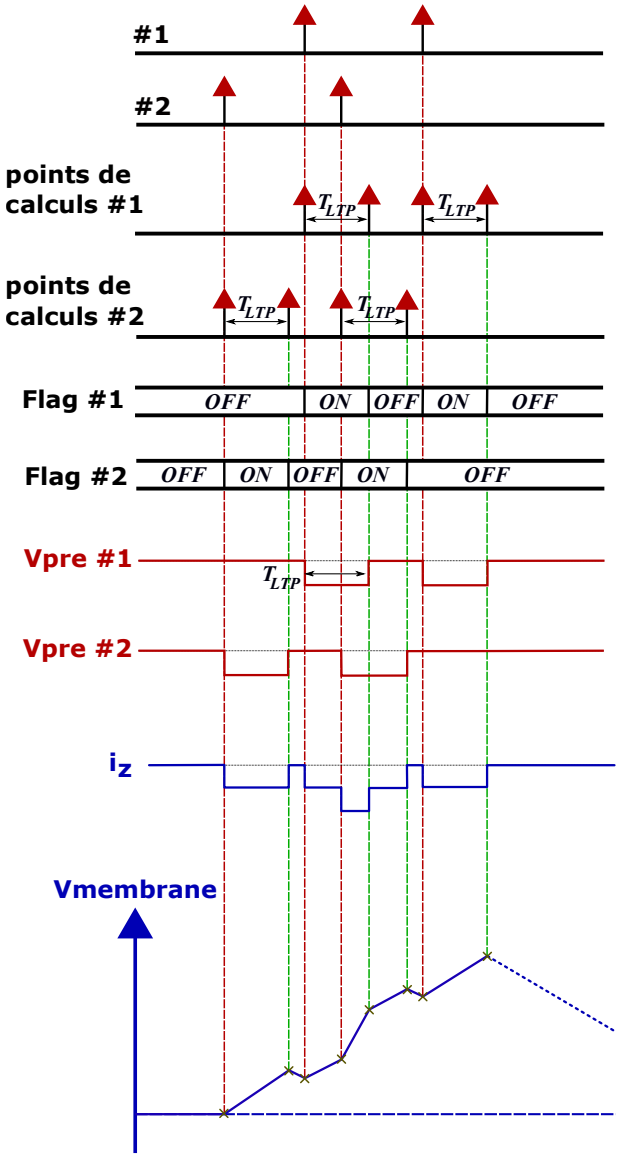


FIGURE 2.4 – Méthode de calcul choisie pour les simulations.
Pour deux entrées, on retrouve les événements d'entrée, les points de calculs correspondants ainsi que les formes d'ondes qu'ils représentent et le courant résultant. Enfin la tension de membrane que l'on peut obtenir à partir de ces points de calculs (symboles « × ») pour un neurone LIF à décharge constante. Les lignes verticales rouges représentent les points de calculs des données présentées, les lignes vertes représentent les points de calculs ajoutés par le simulateur.

Calcul des courants synaptiques et de la tension de membrane

En se basant sur la structure du neurone postsynaptique présentée FIGURE 2.2, les équations (2.6) à (2.8) permettent de déterminer la tension de membrane pour un point de calcul j à l'instant t_j depuis la valeur de tension de membrane calculée au point précédent $(j-1)$. Ces équations ne sont valables que si la décharge de la membrane est constante et les formes d'ondes de stimulation sont des impulsions.

$$i_{\text{cancel}} = \epsilon \times \sum_{\text{m} \in \text{ memristors inactifs}} G_{\text{m}} \tag{2.5}$$

$$\frac{i_z}{K} = \begin{cases} \underbrace{\sum_{\text{n} \in \text{ memristors actifs}} G_{\text{n}} \times \overbrace{\left(V_{\text{mid}} - \epsilon - V_{\text{pre n}}\right)}^{V_{\text{stim}}}}_{i_{\text{stim}}} - i_{\text{cancel}} & \text{pour } i_{\text{stim}} \geq i_{\text{cancel}} \\ 0 & \text{pour } i_{\text{stim}} \leq i_{\text{cancel}} \end{cases} \tag{2.6}$$

$$\Delta V_{\text{membrane}}|_{t_{j-1} \to t_j} = \underbrace{K \times \frac{i_z \div K}{C_{\text{mem}}} \left(t_j - t_{j-1}\right)}_{\text{charge/inférence}} - \underbrace{\frac{i_{\text{discharge}}}{C} \left(t_j - t_{j-1}\right)}_{\text{fuite}} \tag{2.7}$$

$$V_{\text{membrane}}|_{t_j} = V_{\text{membrane}}|_{t_{j-1}} + \Delta V_{\text{membrane}}|_{t_{j-1} \to t_j} \tag{2.8}$$

Dans le cas particulier où l'*offset* ϵ des convoyeurs de courants est identique (voir nul) pour tous les neurones de la couche postsynaptique, le calcul des courants synaptiques de stimulation i_{stim} se simplifie en un produit scalaire entre le vecteur d'entrée correspondant aux tensions présynaptiques et la matrice (crossbar) des conductances des memristors (synapses) donnant ainsi le vecteur des courants de stimulation des neurones postsynaptiques. Si ce n'est pas le cas (variabilité entre les neurones postsynaptiques), il est nécessaire d'effectuer un produit de Hadamard (*element-wise*) entre la matrice de tensions des synapses du réseau et la matrice des conductances des synapses du réseau. Une somme des lignes de la matrice résultante est alors nécessaire pour obtenir le vecteur des courants des neurones postsynaptiques. Cette version du calcul prend donc plus de temps. Ces deux versions du calcul ont été utilisées dans ces travaux de thèse. La version où l'*offset* ϵ est nul a été utilisé pour les simulations de la partie 2.2 Définition des règles d'apprentissage qui ont permis de définir différentes règles d'apprentissage. La version faisant intervenir des *offsets* des neurones postsynaptiques différents a été utilisée dans la partie 2.3 Cas d'étude : Le projet ULPEC pour étudier l'impact d'un tel *offset* sur l'apprentissage en vue d'une implantation matérielle.

Lorsqu'un ou plusieurs neurones postsynaptiques émettent un événement, il faut déterminer l'instant auquel il a tiré. Pour cela, l'instant précis (t_{fire}) auquel un neurone postsynaptique a dépassé le seuil est calculé à partir de l'équation (2.9) qui est adaptée de celles utilisées pour le calcul de la charge de la membrane (équations (2.6) à (2.8)).

$$t_{\text{fire}} = t_j - (V_{\text{membrane}} - V_{\text{seuil}}) \times \frac{C_{\text{mem}}}{i_z - i_{\text{discharge}}} \tag{2.9}$$

2.2 Définition des règles d'apprentissage

Si la structure du réseau de neurones et les méthodes de calcul utilisées pour simuler son comportement ont été présentées dans la partie précédente, la question de la règle d'apprentissage choisie pour le faire apprendre se pose. En effet, seul le comportement des modifications locales des poids synaptiques a été décrit par l'équation (2.1). Dans cette partie, nous verrons la limitation de la règle d'apprentissage classique inspirée de la biologie (STDP) vue dans la partie 1.1.2 L'apprentissage en biologie lorsqu'on souhaite l'implémenter sur un réseau de neurones matériels et l'utiliser pour apprendre des données originaires du monde réel telles que celles issues d'une caméra événementielle. Pour s'affranchir de cette limitation, la nouvelle classe de règles d'apprentissage nommé $i\mathrm{P}j\mathrm{D}$ introduite durant ces travaux de thèse sera présentée ainsi que les modifications possibles de ces règles pour introduire une forme de supervision lors de l'apprentissage.

2.2.1 Limitation de la règle d'apprentissage classique (STDP)

La règle classique d'apprentissage originaire de la biologie (STDP) est une candidate naturelle pour des réseaux de neurones événementiels matériels puisque ces derniers se veulent inspirés de la biologie. Cependant, lorsqu'on prend en considération les propriétés matérielles des réseaux de neurones étudiés durant ces travaux de thèse, cette règle d'apprentissage se révèle incompatible avec les données que l'on souhaite traiter (par exemple celle de la N-MNIST). La version de la STDP matérielle classique que l'on implémente sur les réseaux de neurones événementiels est illustrée sur la FIGURE 2.5 et repose sur la proximité temporelle des événements. Lorsqu'un neurone postsynaptique émet un événement :

— Si le neurone présynaptique est en train d'émettre un événement (tension d'inférence en cours d'application), la synapse est renforcée. Lorsque le neurone postsynaptique émet un événement, sa forme d'onde est immédiatement appliquée et la combinaison des tensions présynaptique et postsynaptique permet de dépasser le seuil d'écriture. La largeur de la forme d'onde d'inférence des neurones présynaptiques sert alors de fenêtre d'apprentissage pour la potentiation des synapses memristives.

— Si le neurone présynaptique était inactif, la synapse est déprimée. Pour cela on déclenche avec un certain délai un signal présynaptique de dépression qui, en se combinant avec la forme d'onde postsynaptique, permet de dépasser le seuil d'écriture (lorsque le neurone postsynaptique émet un événement, sa forme d'onde est immédiatement appliquée).

Ainsi, si un événement postsynaptique intervient durant cette fenêtre, la synapse est renforcée, sinon elle est déprimée. C'est cette fenêtre d'apprentissage qui pose problème lorsque l'on souhaite apprendre des données issues de la N-MNIST ou plus généralement d'une caméra événementielle filmant le monde réel. Lorsque ces données sont générées par un mouvement mécanique, comme c'est le cas avec la base de données N-MNIST, très peu d'événements se

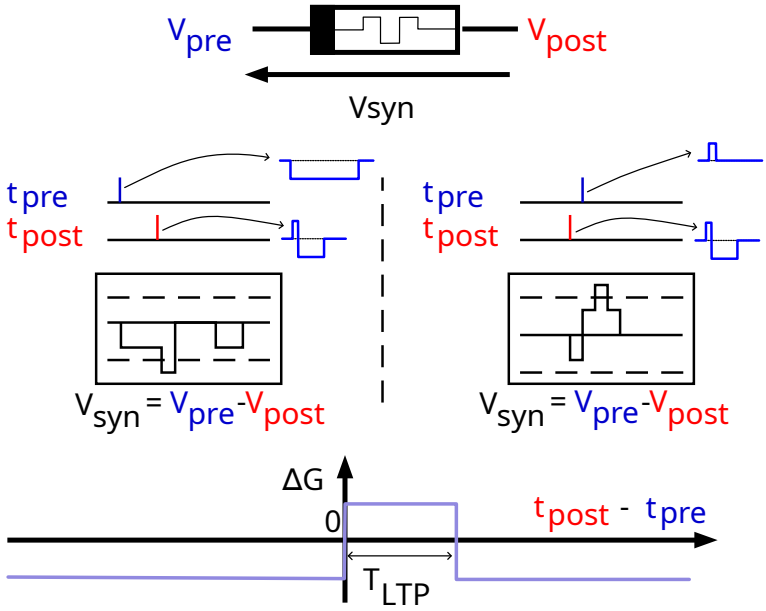


FIGURE 2.5 – Règle d'apprentissage STDP matérielle classique pour les réseaux de neurones événementiels matériels.
Seules les synapses connectées à un neurone présynaptique qui présente un évenement qui tombe dans la fenêtre T_{LTP} (largeur de l'impulsion présynaptique d'inférence) sont renforcées.

déclenchent suffisamment proches dans le temps pour tomber dans la même fenêtre d'apprentissage ($T_{LTP} = 10\,\mu s$). Cette fenêtre de temps T_{LTP} est réalisée de manière analogique [14] et l'augmenter nécessiterait des surfaces de circuit intégré plus importantes pour des implantations matérielles intégrées. Pour des réseaux de neurones de grande dimension, l'augmentation de ce temps devient donc une limitation pour l'intégration.

Pour tous les échantillons de la N-MNIST dédiés à l'apprentissage, le nombre moyen maximum d'événements présynaptique qui partage cette fenêtre d'apprentissage est illustré sur la FIGURE 2.6. Cette disparité entre la fenêtre d'apprentissage matériel et le rythme des données en entrée cause la dépression d'une majorité des synapses connectées aux neurones et seulement quelques synapses peuvent être renforcées (3.68 en moyenne par échantillon si l'évènement postsynaptique a bien lieu lorsque ces événements sont en cours d'inférence). D'après des simulations réalisées avec cette règle d'apprentissage, les neurones postsynaptiques finissent tous avec des taux de reconnaissance de 0 % car ils n'arrivent plus à déclencher. En effet, avec une majorité des poids se faisant déprimer à chaque phase d'apprentissage, les neurones postsynaptiques ne sont plus suffisamment stimulés pour pouvoir déclencher un événement.

14. C'est le cas si la partie potentiation de la fenêtre d'apprentissage correspond à la largeur du signal présynaptique (version initiale prévue pour le démonstrateur ULPEC).

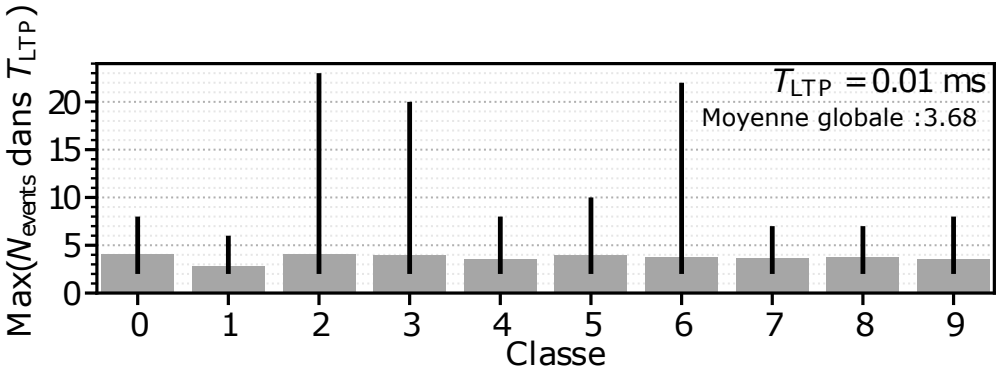


FIGURE 2.6 – Nombre maximum d'événements présynaptique N_{events} tombant dans la même fenêtre d'apprentissage T_{LTP} pour chaque chiffre (classe d'intérêt : polarité ON) de la N-MNIST.

2.2.2 Définition des règles d'apprentissage non supervisées iPjD

Pour remédier à la disparité des échelles de temps entre la fenêtre d'apprentissage et le rythme des données en entrée, nous ajoutons une information supplémentaire dans le bloc DCB. Pour chaque neurone présynaptique, une mémoire est utilisée pour stocker le nombre d'événements qu'il a émis (N_{fire}) tant qu'aucun événement postsynaptique n'a eu lieu. Ainsi en utilisant la valeur contenue dans ces mémoires (valeur indépendante de la fenêtre T_{LTP}), on définit les règles d'apprentissages illustrées TABLE 2.2 qui distinguent 3 versions différentes d'apprentissage suivant l'utilisation de cette mémoire N_{fire} et suivant les valeurs de i et j :

1. La règle 0P0D ($i = j = 0 \rightarrow$ cas particulier) qui correspond à la règle d'apprentissage de type STDP matérielle précédente. Elle renforce les synapses en cours d'utilisation et déprime le reste. C'est une règle qui se veut compatible avec des *bursts*[15] de données proches temporellement (dans la fenêtre T_{LTP}).

2. La règle 0P1D ($i = 0 \rightarrow$ cas particulier) qui évite de déprimer toutes les synapses inactives pour s'affranchir en partie de la fenêtre d'apprentissage. Elle renforce les synapses en cours d'utilisation et déprime seulement celles qui n'ont jamais émis d'événement. Elle est donc moins sensible que 0P0D à la fenêtre d'apprentissage T_{LTP}.

3. La règle générique iPjD (avec $i \geq j \geq 1$) qui s'affranchit entièrement de la fenêtre d'apprentissage T_{LTP}. On renforce les synapses qui ont émis au moins i événements ($N_{fire} \geq i$) sinon, on déprime les synapses qui ont émis moins de j événements ($N_{fire} < j$). Pour le cas particulier de 1P1D, les synapses qui ont émis au moins un événement sont renforcées et celles qui n'ont jamais émis d'événements sont déprimées. Cette règle 1P1D rappelle la règle d'apprentissage proposée par Masquelier et coll. [108] qui ne dépend que du signe de l'écart entre les événements présynaptiques et postsynaptiques.

15. Données émises sous forme de paquets proches dans le temps.

Afin d'étudier le comportement de ces différentes versions de règles d'apprentissage, des simulations ont été réalisées avec l'architecture du réseau présenté précédemment et avec les paramètres donnés TABLE 2.1 (sauf mention contraire explicite). Si la règle 0P0D ne permet pas d'apprendre avec ces paramètres du réseau de neurones, la règle 1P1D qui a été imaginée pour pallier le problème de la dynamique des données en entrée permet d'apprendre et de reconnaître à 68.1 % (moyenne de 3 simulations avec des poids initiaux différents pour 1 epoch [16]) les données issues de la N-MNIST. Un exemple des cartes de conductances [17] résultantes après apprentissage est illustré sur la FIGURE 2.7.

Les différentes règles présentées ici sont dites non supervisées puisqu'elles ne prennent pas en compte la classe de l'échantillon d'entrée pour adapter l'apprentissage en fonction de la réponse du réseau.

TABLE 2.2 – Règles d'apprentissage iPjD.

Événement présynaptique	**Règles**			
	0P0D (STDP-like)	**0P1D**	**1P1D**	i**P**j**D** avec $i \geq j \geq 1$
En cours	pot(entiation)	pot		
Inactif	dep(ression)			
$N_{\text{fire}} \geq i$			pot	pot
$j \leq N_{\text{fire}} < i$				null (pas de changement)
$N_{\text{fire}} < j$		dep	dep	dep

16. Nombre de présentation des échantillons d'entraînement.
17. Il s'agit de la représentation par neurone postsynaptique de toutes les synapses réarrangées sous forme de matrice pour correspondre aux dimensions de l'image d'entrée.

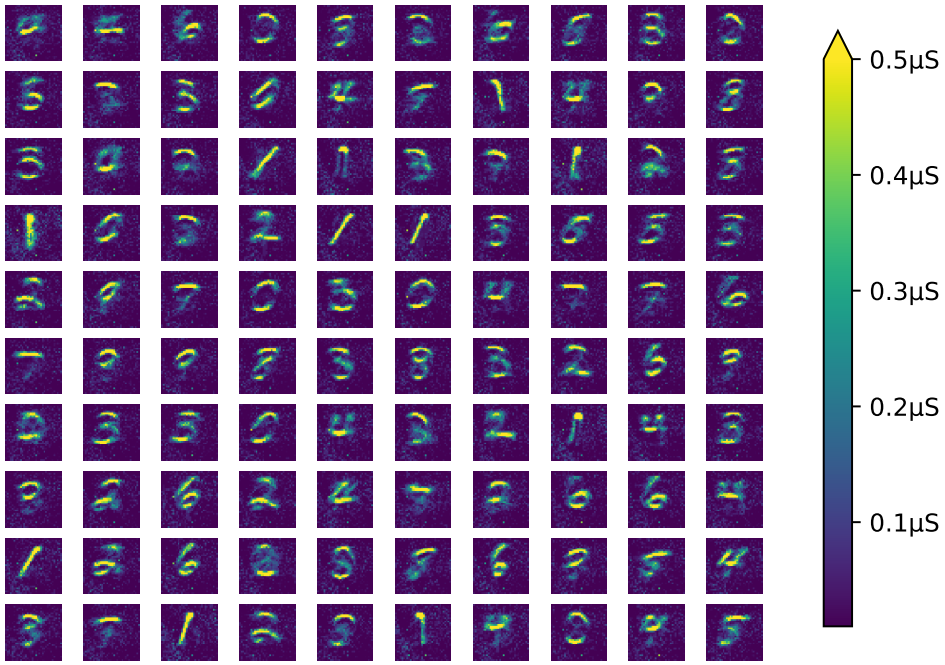


FIGURE 2.7 – Les 100 cartes de conductances obtenues après apprentissage pour 1 epoch de la N-MNIST d'un réseau 1156 × 100 (paramètres Table 2.1) et la règle 1P1D (Table 2.2).
Ces cartes de conductances donnent un taux de reconnaissance de 67.66 % après étiquetage. On y reconnaît visuellement le chiffre auquel chaque neurone est devenu sensible.

Impact de la dynamique des données en entrée

Afin d'expliquer pourquoi la règle 0P0D (STDP-like) ne fonctionne pas avec les données d'origine de la N-MNIST, l'hypothèse du manque d'événements se produisant dans la même fenêtre de temps T_{LTP} a été avancée. Pour vérifier cette hypothèse par la simulation, avec les mêmes paramètres du réseau de neurones, nous présentons les données de la N-MNIST en leur appliquant un facteur d'accélération allant de 1 (donnée originale) à 100. Ce facteur s'applique sur les événements issus de la base de données et les événements ajoutés par le simulateur sont toujours espacés de T_{LTP} par rapport aux données utilisées. Sur la FIGURE 2.8 est représenté le nombre d'événements actifs au cours du temps pour un échantillon de la base de données d'origine et accéléré par 100. On y observe l'augmentation du nombre d'événements actifs sur la même fenêtre de temps, ce qui devrait permettre à la règle 0P0D de fonctionner.

La FIGURE 2.9 illustre pour les 3 règles d'apprentissage 0P0D, 0P1D et 1P1D les taux de reconnaissance obtenus avec les mêmes paramètres de simulation pour différents facteurs d'accélération de la base de données d'entrée.

La règle 1P1D qui a été définie pour ne pas dépendre de cette fenêtre de temps d'apprentissage T_{LTP} offre, par rapport aux autres règles qui en sont plus ou moins dépendantes, des performances supérieures pour tous les facteurs d'accélération. Si pour les données d'origine son taux de reconnaissance et de 68.1 %, il diminue légèrement plus le facteur d'accélération augmente pour atteindre 60.5 % pour un facteur d'accélération de 100.

La règle 0P0D est incapable d'apprendre pour les données d'origine très peu d'événements sont superposés la même fenêtre de temps. Cependant, dès que le facteur d'accélération dépasse 10 cette règle présente un taux de reconnaissance au-dessus du niveau de réponse aléatoire (RR=10 %) jusqu'à atteindre 46.8 % pour un facteur d'accélération de 100. Ces résultats mettent en évidence l'importance primordiale pour cette règle 0P0D d'avoir des événements en entrée qui se superposent sur cette fenêtre apprentissage T_{LTP}.

La règle 0P1D a été définie pour être moins sensible que la précédente à cette fenêtre d'apprentissage en ne dépréciant que les synapses connectées aux entrées qui n'ont jamais émis d'événements. Si elle n'est pas aussi performante que la règle 1P1D, elle est capable d'apprendre et de reconnaître sur la base de données d'origine contrairement à 0P0D. Cette règle qui se place entre 1P1D et 0P0D en termes de fonctionnement (renforce les poids dans la fenêtre d'apprentissage comme 0P0D et déprécie ceux qui n'ont jamais tiré comme 1P1D) présente des performances qui se trouvent entre ces 2 règles.

Si la règle 1P1D est la meilleure du point de vue taux de reconnaissance pour ces simulations, les règles 0P0D ou 0P1D peuvent présenter un intérêt pour des données plus rapides comme

des bursts de données (symbole clignotant par exemple). Dans une telle situation, la règle 0P0D qui nécessite moins de logique (pas de compteur N_{fire}) peut présenter un intérêt si les ressources nécessaires à la mise en œuvre du DCB sont une contrainte.

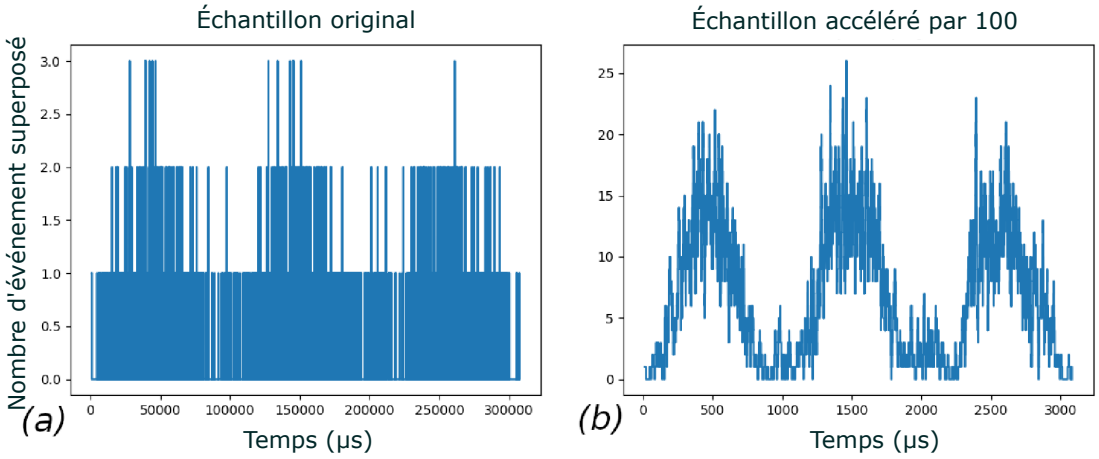


FIGURE 2.8 – Nombre d'événements actifs en même temps pour un échantillon de la N-MNIST d'origine et accéléré par 100.

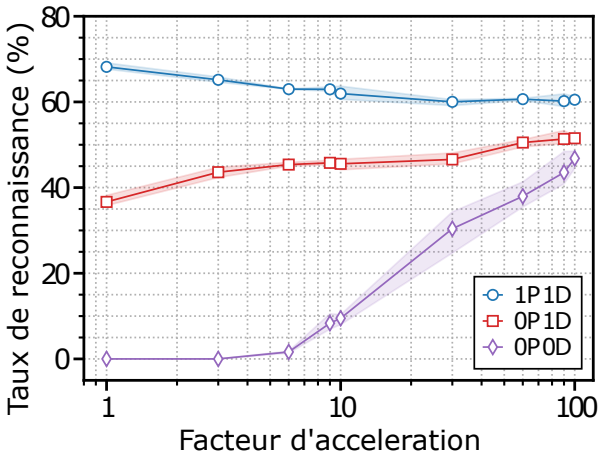


FIGURE 2.9 – Évolution du taux de reconnaissance en fonction du facteur d'accélération de la base de données N-MNIST pour 3 règles d'apprentissage différentes : 1P1D (∘), 0P1D (□), and 0P0D (◇).
Les symboles représentent la valeur moyenne sur 3 simulations et les zones colorées le maximum et minimum. Ces résultats ont été obtenus à partir des paramètres de la table pour 3 simulations utilisant 3 jeux de conductances initiales identiques.

Impact de l'asymétrie du changement des poids synaptiques

Dans les simulations qui ont été présentées précédemment, avec les paramètres par défaut du simulateur, les taux d'apprentissage A^+ et A^- qui caractérisent le taux de variation du poids

synaptique lors de l'apprentissage sont donnés tel que $A^+ = A^- = 0.05$. Ces taux d'apprentissage sont donc symétriques. Cependant dans des systèmes matériels utilisant des memristors ferroélectriques tels que celui illustré FIGURE 1.12, une asymétrie peut être présente entre ces deux taux d'apprentissage. Afin d'observer l'impact que peut avoir ce phénomène sur les différentes règles d'apprentissage définies, le réseau de neurones a été simulé pour différents rapports d'asymétrie A^+/A^- allant de 0.1 à 100. Ces résultats sont illustrés FIGURE 2.10.

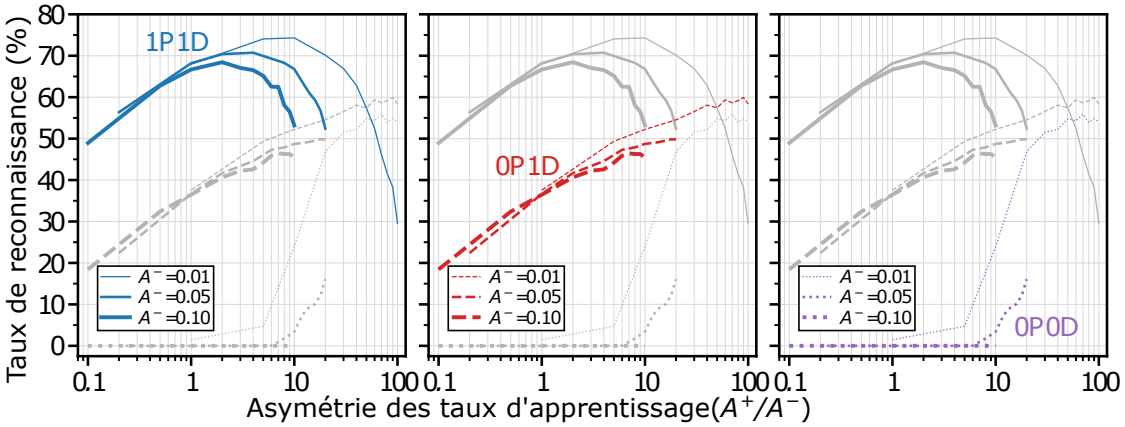


FIGURE 2.10 – Évolution du taux de reconnaissance pour les règles 0P0D, 0P1D et 1P1D en fonction de l'asymétrie des taux d'apprentissage des synapses.
Pour chaque scénario, trois valeurs de A^- sont considérées : 0.01, 0.05, et 0.1 (respectivement de la ligne la plus fine à la plus épaisse). Ces résultats sont obtenus par la moyenne de 3 simulations et chaque scénario utilise le même jeu de conductances initiales.

La règle 1P1D montre encore une fois les meilleures performances en termes de taux de reconnaissance sur la quasi-totalité des simulations réalisées. On observe néanmoins que cette règle présente un rapport de taux d'asymétrie A^+/A^- optimal local qui est situé dans une asymétrie favorisant A^+ (dans la zone $A^+/A^->1$). Cet optimum se décale à droite plus A^- est fort, ce qui signifie que si A^- augmente, A^+ doit également augmenter puisque le taux d'asymétrie où se trouve l'optimum de cette règle d'apprentissage a augmenté. Enfin, une fois cet optimum dépassé, le taux de reconnaissance tend à décroître rapidement plus le taux d'asymétrie augmente, ce qui n'est pas le cas avec les deux autres règles d'apprentissage. On peut noter que le taux de reconnaissance maximum obtenu avec $A^- = 0.01$ et $A^-/A^+ = 10$ est de 74.3 % ce qui dépasse légèrement le taux de reconnaissance de 74.03 % rapporté par Iyer et Basu [109] pour une architecture similaire (SNN à une couche *all-to-all*, utilisant une règle STDP conventionnelle avec des valeurs de paramètres temporels difficilement compatibles avec une mise en œuvre matérielle).

La règle 0P1D présente un taux de reconnaissance qui augmente plus cette asymétrie en faveur de A^+augmente. Cela s'explique par le fait qu'une telle asymétrie favorisant la potentia-

lisation avec une dépression faible permet de compenser en partie le peu d'événements renforcés en même temps. Pour les taux d'asymétrie les plus importants, lorsque 1P1D s'effondre, cette règle permet d'obtenir des taux de reconnaissances supérieures. Pour des taux d'asymétrie $A^-/A^+ \geq 50$ avec $A^- = 0.01$, cette règle devient la meilleure pour ces simulations.

La règle 0P0D montre quant à elle une capacité à apprendre et à reconnaître pour des taux d'asymétrie $A^-/A^+ \geq 10$. Cela s'explique, similairement à la règle précédente, par une potentialisation suffisamment importante par rapport à la dépression lors de l'apprentissage pour compenser la surdépression naturelle de cette règle pour la base de données N-MNIST (majorité des synapses dépréciées avec cette règle pour la N-MNIST d'origine). Tout comme 0P1D, elle parvient à dépasser les performances de 1P1D pour $A^-/A^+ \geq 50$ avec $A^- = 0.01$.

À partir de ces simulations, on peut en déduire les conclusions suivantes :

— La règle 1P1D est la candidate de choix pour une majorité de taux d'asymétrie.

— Il est possible d'optimiser l'apprentissage de la règle 1P1D en jouant sur le taux d'asymétrie pour aller chercher l'optimum du taux de reconnaissance. Pour cela on peut imaginer adapter les tensions d'écriture utilisées par le réseau de neurones pour causer cette asymétrie.

— Pour des taux d'asymétrie importants, la règle 0P1D peut permettre d'avoir un meilleur taux de reconnaissance que 1P1D, même si ce dernier risque d'être inférieur que celui qu'on peut obtenir en allant chercher l'optimum de 1P1D.

— La règle 0P0D peut être intéressante si on souhaite réaliser un système avec un DCB de taille réduite pour des taux d'asymétrie extrêmement forts.

Quel intérêt pour des valeurs de i et j supérieures à 1 ?

Les règles d'apprentissage iPjD qui ont été définies durant ces travaux de thèse offre la possibilité d'utiliser des valeurs de N_{fire} supérieures à 1. La règle 1P1D suppose que seules les données utiles (correspondant au chiffre pour la N-MNIST) sont représentées dans l'échantillon. Si des pixels filmant l'arrière-plan émettent des événements, cela peut causer des chutes de performance avec la règle 1P1D puisqu'ils se feront renforcer alors qu'ils ne correspondent pas aux données utiles que l'on souhaite apprendre (ces événements sont assimilés à du bruit). Pour pouvoir explorer ce phénomène, nous avons ajouté un bruit de grenaille à chaque pixel (pour tous les échantillons utilisés) afin de calculer des retards aléatoires Δt de déclenchement d'événements liés au bruit. La fonction de densité de probabilité utilisée est donnée (2.10) et le retard moyen entre chaque événement lié au bruit est $\langle \Delta t \rangle = \lambda^{-1}$. Les simulations faisant intervenir du bruit dans les échantillons d'entrée utilisent un paramètre d'inférence T_{LTP} =4 µs qui a été réduit pour compenser les stimulations supplémentaires provenant des événements liés à ce bruit.

$$f(\Delta t \geq 0, \lambda) = \lambda \exp(-\lambda \Delta t) \tag{2.10}$$

Afin d'observer la différence entre ce que va apprendre le réseau avec ou sans bruit pour différentes valeurs i et j pour la règle d'apprentissage, on représente sur la FIGURE 2.11 les masques d'apprentissage [18] qu'utilisera le DCB si un événement se déclenche au bout de 50 ms. On peut observer que l'augmentation de $i(=j)$ permet de filtrer les événements n'appartenant pas aux données utiles que l'on souhaite apprendre (chiffre 7 pour la FIGURE 2.11). On peut également noter que du bruit existe déjà dans les données d'origine de la N-MNIST puisque le masque d'apprentissage pour cet échantillon contient des pixels appartenant à l'arrière-plan.

Les résultats obtenus après apprentissage en utilisant les 3 règles d'apprentissage 1P1D, 2P2D et 3P3D sont donnés FIGURE 2.12 avec une illustration de quelques cartes de conductances correspondantes.

Concernant les échantillons d'origine (sans bruit ajouté), même si les cartes de conductance obtenues semblent affiner les chiffres présentés (les chiffres appris sont plus nettement visibles), augmenter i et j pour filtrer le bruit ne permet pas d'augmenter les performances. Au contraire les performances obtenues lorsqu'on augmente i et j diminuent : 68.1 % en moyenne pour 1P1D, 60.8 % en moyenne pour 2P2D et 49.0 % en moyenne pour 3P3D. Cela peut s'expliquer par le fait qu'affiner les données en entrée réduit les points communs entre les échantillons de la même classe sur lesquels le réseau va apprendre et reconnaître.

Concernant les échantillons avec du bruit ajouté, les performances obtenues avec la règle d'origine 1P1D s'effondrent pour atteindre 33.0 % en moyenne [19]. L'utilisation de 2P2D qui permet de filtrer une partie du bruit permet également de récupérer des performances en atteignant 51.1 %. La règle 3P3D présente des résultats supérieurs à 1P1D et reste moins efficace que 2P2D. On peut noter que les résultats obtenus avec et sans bruit lorsque la règle 3P 3D est utilisée sont quasi identiques (49.7 % et 49.0 % respectivement). Ce qui semble suggérer que l'on a atteint la limite du filtrage possible des événements de bruits ajoutés artificiellement par l'augmentation de i et j.

En conclusion, l'ajustement des paramètres i et j des règles de la famille iPjD permet de récupérer des performances ou les maintenir si les échantillons utilisés pendant l'apprentissage présentent un certain bruit. Il faut noter que le passage de la règle 1P1D à des valeurs supérieures à 1 pour i et j nécessite dans le DCB d'augmenter le nombre de bits nécessaire pour stocker l'information permettant de savoir si un présynaptique a tiré ou non N_{fire} fois [20]. La

18. Il s'agit de la représentation sous forme d'images des synapses qui vont se faire renforcer d'après la règle choisie.
19. Au-dessus du niveau de réponse aléatoire pour 10 classes.
20. Tandis que la règle 1P1D nécessite un seul bit par entrée présynaptique pour stocker cette information.

meilleure solution semble néanmoins être la règle 1P1D et l'utilisation de données d'entrée non bruitées pour l'apprentissage.

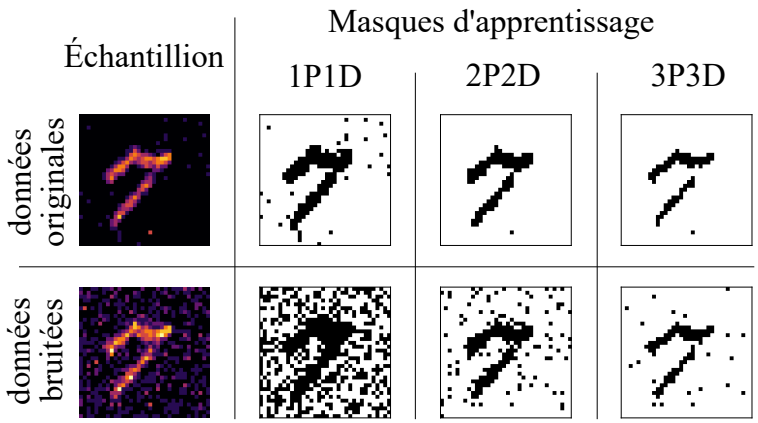
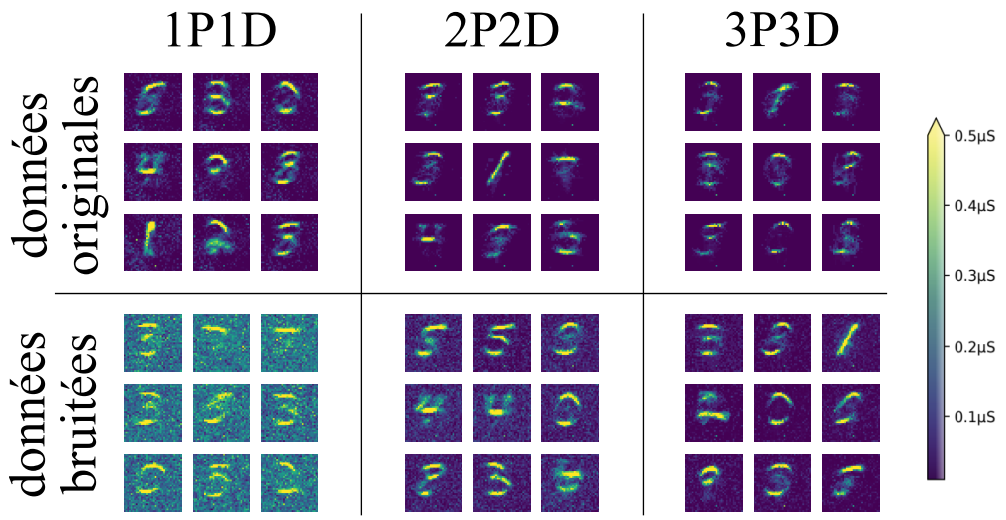


FIGURE 2.11 – Échantillon de la N-MNIST sans (au-dessus) ou avec (en dessous) du bruit supplémentaire et les masques d'apprentissage correspondants.
Échantillon : accumulation des événements de polarité ON pour les 50 ms au début de l'échantillon de la N-MNIST. Plus un point (pixel) est lumineux plus il y a eu d'événements.
Masques d'apprentissage : masque d'apprentissage dans le DCB des synapses qui seront renforcées (en noir) ou déprimées (en blanc) si un neurone postsynaptique émet un évènement au bout de 50 ms pour cet échantillon.



Règle	Données originales ($T_{LTP} = 10\mu s$)			Données bruitées ($\lambda^{-1} = 0.1\,s$, $T_{LTP} = 4\mu s$)		
	min	moyenne	max	min	moyenne	max
1P1D	67.6%	68.1%	69.1%	31.0%	33.0%	34.4%
2P2D	60.1%	60.8%	61.4%	54.1%	56.1%	59.9%
3P3D	47.0%	49.0%	51.6%	48.3%	49.7%	50.7%

FIGURE 2.12 – Taux de reconnaissance (tableau) et exemple de cartes de conductances résultantes pour de l'apprentissage non supervisé avec les règles 1P1D, 2P2D et 3P3D. Obtenu après 1 epoch de la N-MNIST avec (droite) ou sans (gauche) bruit additionnel. Résultat obtenus après 3 simulations basées sur l'utilisation des 3 mêmes sets de conductances synaptiques initiales entre les règles d'apprentissage. Le bruit poissonnien est généré pour chaque échantillon d'entraînement et de test avec le paramètre $\lambda^{-1} = 0.1\,s$. Le paramètre d'inférence T_{LTP} a été réduit pour les simulations avec bruit pour compenser la charge de la membrane provenant des évènements supplémentaires issus du bruit.

2.2.3 Règles d'apprentissage faiblement supervisé Rm-iPjD

Les règles présentées jusqu'ici appartenaient à la famille des règles d'apprentissage non supervisées puisque l'information de la classe de l'échantillon utilisé pendant l'apprentissage n'était pas utilisée pour assister les règles d'apprentissage employées. L'apprentissage supervisé, qui consiste à prédéterminer le label du neurone de sortie et à adapter l'apprentissage pour atteindre ce label, permet généralement d'obtenir avec les réseaux de neurones événementiels des taux de reconnaissance bien supérieurs au non supervisé. Avec de l'apprentissage supervisé, on peut par exemple atteindre des RR supérieurs à 98 % sur la N-MNIST grâce à l'utilisation d'algorithmes complexes [110] [21] mais cela reste difficilement intégrable pour des architectures mixtes (numérique/analogique) à faible empreinte matérielle telle que celle étudiée dans ces travaux de thèse. Ainsi, afin d'éviter l'ajout d'algorithmes complexes pour la supervision, ce sont les règles à modulation de récompense que nous avons choisi d'étudier. Ces règles R-STDP (de l'anglais *Reward-modulated* STDP) sont une famille de règles qui présentent dans la littérature des résultats prometteurs pour faire de la supervision sur des réseaux de neurones événementiels profonds [111] [112].

La FIGURE 2.13 illustre le fonctionnement de cette famille de règles dans le cadre de ces travaux de thèse lorsqu'un neurone postsynaptique émet un événement et qu'il entre dans sa phase d'apprentissage. Si le neurone tire pour un échantillon d'entrée qui correspond au label qu'on lui a attribué (classe correcte), la règle classique dite de récompense est appliquée. Si le neurone tire pour un échantillon d'entrée qui ne correspond pas au label qu'on lui a attribué (classe incorrecte), une règle dite de punition est appliquée. Ces règles d'apprentissage supervisé sont dites faibles parce que leur méthode pour "corriger" le réseau de neurones en cas d'erreur ne fait pas intervenir de correction directe des poids synaptiques et se contente de changer la règle de modification de ces derniers sans connaître leurs valeurs, contrairement aux méthodes utilisant la *backpropagation*. Cette stratégie est facilement implantable dans le DCB, puisque seul l'ajout de l'information sur le label souhaité des neurones de sortie et la classe de l'échantillon présenté pour adapter les signaux de modification des synapses déclenchées par le DCB est nécessaire. En utilisant les règles iPjD décrites précédemment comme règle de récompense, on peut définir un ensemble de règles d'apprentissage supervisé données TABLE 2.3 appelées Rm-iPjD où m indique la règle de punition choisie et iPjD la règle de récompense choisie. On choisit comme règle de récompense 1P1D pour ces simulations puisque c'est celle qui, dans les travaux présentés précédemment, présente les meilleures performances de manière générale.

Afin d'étudier le comportement de ces règles d'apprentissage supervisées, le réseau de neurones de référence a été simulé pour différentes valeurs de taux d'apprentissage identiques ($A^+ = A^-$). Pour la simulation, l'attribution des labels des neurones postsynaptiques pour l'ap-

21. Utilise la *backpropagation*.

prentissage s'est faite de la manière suivante. Le neurone de sortie d'index n prends comme label $\lfloor n/N_{\text{classes}} \rfloor$ [22]. La FIGURE 2.14 montre des cartes de conductances obtenues pour certaines de ces règles non supervisées comparées à la règle 1P1D [23]. On y observe que les cartes de conductance après apprentissage sont ordonnées contrairement à l'apprentissage non supervisé. Cette supervision permet bien de prédéterminer ce que va apprendre chaque neurone.

La FIGURE 2.15 recense les résultats de simulation pour différentes valeurs de $A^- = A^+$. Les règles R_α-1P1D et R_β-1P1D, dont la règle de punition renforce les synapses qui n'ont jamais transmis de stimulation, n'arrivent pas à apprendre puisque leur taux de reconnaissance reste au niveau de bruit RR = 10 %.

La règle $R_\varnothing$-1P1D qui est considéré comme la plus simple puisque la règle de punition consiste à ne rien faire présente un taux de reconnaissance toujours supérieur à la règle de référence 1P1D pour toutes les simulations réalisées. Elle présente néanmoins un taux de reconnaissance plus faible que R_γ-1P1D pour des taux d'apprentissage faibles $A^+ = A^- < 0.12$.

La règle R_γ-1P1D présente les meilleures performances pour de petits taux d'apprentissage $A^+ = A^- < 0.12$ alors que sa règle de punition consiste à déprécier toutes les synapses qui ont émis au moins une stimulation si l'échantillon présenté n'est pas de la bonne classe. Cette règle présente un taux de reconnaissance maximum de 80.1 % pour $A^- = A^+ = 0.02$, le plus élevé de toutes les simulations réalisées, et chute pour atteindre des valeurs sous le niveau de bruit au-delà $A^+ = A^- > 0.3$.

Si la règle R_γ-1P1D peut être utilisée pour obtenir de meilleur taux de reconnaissance avec des taux d'apprentissage faibles, ce sont les règles $R_\varnothing$-1P1D et 1P1P qui sont capable de fonctionner raisonnablement sur toute la plage et sont donc bien plus résilientes pour des taux d'apprentissage proches des 50 %. La règle $R_\varnothing$-1P1D est celle qui résiste le mieux pour des taux d'apprentissage allant jusqu'à 50 %.

TABLE 2.3 – Règles Rm-iPjD.

Règle de récompense	***i*P*j*D**			
$N_{\text{fire}} \geq i$	pot			
$N_{\text{fire}} < j$	dep			
Règles de punition *m*	α	β	γ	$\varnothing$
$N_{\text{fire}} \geq i$	dep	null	dep	null
$N_{\text{fire}} < j$	pot	pot	null	null

22. $\lfloor x \rfloor$: partie entière de x.
23. R_α-1P1D et R_β-1P1D ne sont pas illustrées, car elles ne produisent pas de résultat satisfaisant (RR = 10 % correspondant au niveau de réponse aléatoire).

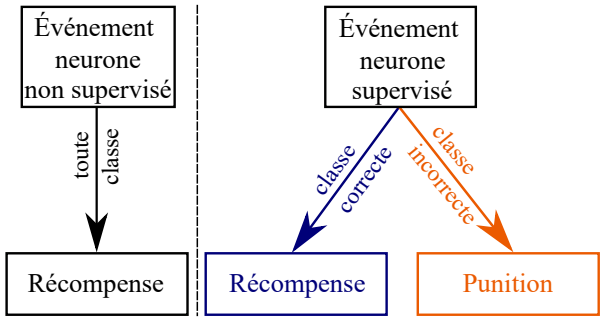


FIGURE 2.13 – Principe de fonctionnement de la règle d'apprentissage non supervisée (gauche) et faiblement supervisée (droite) lorsqu'un neurone postsynaptique tire.

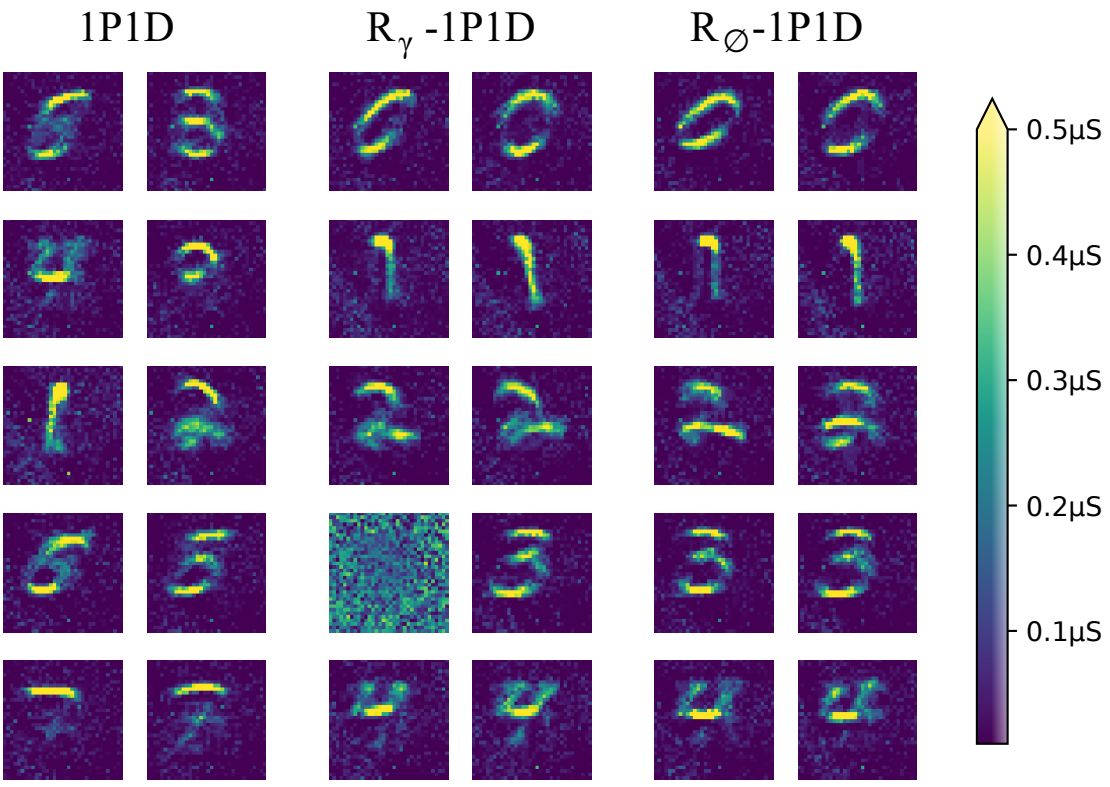


FIGURE 2.14 – Comparaison de 10 cartes de conductances pour 1P1D, R_γ-1P1D et $R_\varnothing$-1P1D. Obtenu après 1 epoch avec les paramètres du réseau de neurones de référence. La valeur maximum des conductances tracées a été écrêtée à 0.5 µS pour la visualisation et les cartes représentées correspondent aux 2 premiers neurones de sortie de chaque dizaine. Ces cartes de conductances sont issues de simulations qui ont donné des taux de reconnaissance de 67.66 % pour 1P1D, 77.68 % pour R_γ-1P1D et 74.89 % pour $R_\varnothing$-1P1D.

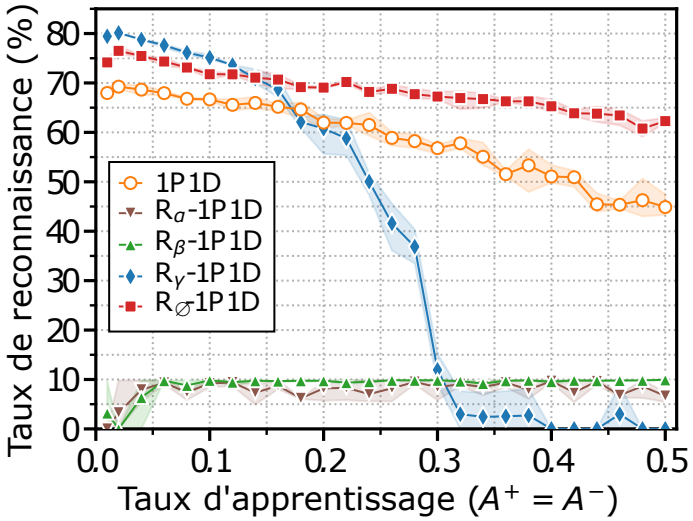


FIGURE 2.15 – Évolution du taux de reconnaissance par rapport au taux d'apprentissage (A^+=A^-) pour la règle non supervisée de référence 1P1D et ses variantes faiblement supervisées.
Résultats obtenus par la moyenne de 3 simulations avec les paramètres définis dans la TABLE 2.1 utilisant 3 jeux de conductances initiales identiques.

2.2.4 Conclusion et comparaison

Dans cette partie a été introduite une famille de règles d'apprentissage qui se veulent facilement intégrables et dédiées dans un premier temps à l'apprentissage non supervisé pour des réseaux de neurones événementiels. La simulation nous a permis d'étudier les différents comportements de ces règles non supervisées et de déterminer leurs zones de fonctionnement et d'utilisation que ce soit pour s'adapter à la dynamique des données d'entrée ou compenser des asymétries dans l'apprentissage (voire filtrer du bruit sur les données d'entrée). Nous avons ensuite adapté ces règles à de l'apprentissage faiblement supervisé en utilisant le concept de récompense modulée inspiré de la biologie. Les résultats obtenus avec supervision sont supérieurs à ceux que propose l'apprentissage non supervisé et permettent contrairement au non supervisé de préarranger les labels de réponse des neurones de sortie. Des résultats obtenus avec les meilleures règles d'apprentissage obtenu en simulation sont répertoriés dans la TABLE 2.4 et sont comparés aux travaux de Iyer et Basu sur une architecture de réseau de neurones événementiels similaire. On notera que nos simulations permettent d'atteindre un niveau de taux de reconnaissance similaire avec moins de neurones de sortie.

Les résultats de simulation présentés dans cette partie ont donné lieu à un article de conférence lors de IJCNN 2020 [113].

TABLE 2.4 – Taux de reconnaissance obtenus comparés à la littérature.

Ces travaux de thèse[†]

$N_{outputs}$	100	400	Remarques
1P1D $A^+ / A-$=0.01	74.3 %	77.6 %	Asymétrie (A^+=0.1,A^-=0.01) 4 epochs* pour $N_{outputs}$ = 400
1P1D R_γ-1P1D $R_\varnothing$-1P1D	68.1 % 78.1 % 74.6 %	75.0 % 80.8 % 78.1 %	Configuration de référence (TABLE 2.1) 4 epochs* pour $N_{outputs}$ = 400

Resultats tirés de [109]

$N_{outputs}$	400	800	Remarques
Sous-motif[+]	74.02 %	76.01 %	Reconnaissance par sous-motif 3 saccades (~ 30 classes)
Motif entier (3 saccades)	78.13 %	80.63 %	Reconnaissance motif complet avec vote majoritaire

[†] Résultats moyen obtenus par la simulation pour 3 distributions de poids synaptiques initiaux pour chaque configuration.
*Échantillons présentés dans le même ordre pour chaque epoch.
[+] Un sous-motif correspond à une saccade et le motif à l'échantillon complet (3 saccades).

2.3 Cas d'étude : Le projet ULPEC

Le projet européen ULPEC vise à réaliser un capteur bio-inspiré intégré comprenant une rétine artificielle sur silicium (caméra événementielle) et un réseau de neurones impulsionnels analogiques [24] (784 entrées, 100 sorties) *all-to-all* au travers d'un crossbar de 784×100 memristors ferroélectriques. Le but de ce capteur intégré est d'apprendre de manière non supervisée à reconnaître des chiffres manuscrits de 28×28 pixels filmés par une caméra événementielle en mouvement. Dans ce projet, mes travaux ont consisté à définir l'architecture du réseau de neurones à implanter dans le capteur. L'architecture du capteur choisie pour ce projet est celle qui a été présentée FIGURE 2.1, comprenant une caméra événementielle, un réseau de neurones événementiels analogique et le circuit numérique de contrôle DCB. C'est dans ce cadre que j'ai eu à mettre au point le simulateur en Python utilisé pour les simulations présentées dans ce chapitre. Il permet de prendre en compte les contraintes matérielles et le type des informations en entrée du réseau afin d'aider à définir une architecture et ses paramètres pour qu'il soit apte à réaliser de l'apprentissage. Ces travaux ont été réalisés en parallèle de ceux du Dr Charly Meyer [114] dont la tâche a consisté à concevoir la partie silicium du réseau de neurones et le circuit numérique DCB. Cela comprend donc le DCB, les neurones présynaptiques et les neurones postsynaptiques ainsi que les connexions vers la caméra événementielle (réalisée par un partenaire : Prophesee) et le crossbar de memristors (réalisé par un partenaire : IBM Zurich).

Le but des simulations réalisées dans le cadre de ce projet a été de déterminer les paramètres du réseau de neurones afin qu'il puisse apprendre et d'étudier l'impact de certaines variabilités dues à l'électronique sur les performances de ce réseau.

Certaines des valeurs des paramètres du réseau sont différentes de celles qui ont été établies comme les paramètres par défaut. Ces travaux, qui ont donné lieu à un article de conférence BioCAS [115], ont été réalisés avant les études présentées précédemment sur une version donc antérieure du simulateur, ce qui explique la disparité de certains paramètres et comportements du simulateur. Les différences en question sont :

— N_{inputs} qui a été fixé à 28×28. Le choix de réduire le nombre d'entrées s'explique par les contraintes en courant nécessaire pour piloter les lignes postsynaptiques par les neurones postsynaptiques. Le choix a donc été fait de réduire le nombre d'entrées par rapport à celles de la base de données de référence N-MNIST (34×34) afin de diminuer les courants requis.

— K a été fixé à 1/125 par le concepteur [25]. L'architecture implémentée possède d'autres facteurs de copie en courant pour pouvoir si besoin s'adapter à d'autres types de données en entrée, mais ces autres valeurs ne sont pas étudiées dans ces travaux.

24. Le démonstrateur entier étant une architecture mixte (SNN analogique, DCB numérique...).
25. Les simulations précédentes utilisaient $K = 1/100$.

— N_{classes} = 3. Le nombre de classes utilisées a été réduit à 3 (chiffres 5, 6 et 9) comme d'autres travaux dans la littérature [109], afin de réduire le temps de simulation. Cela résulte en 17 288 chantillons d'entraînement et 2859 échantillons de tests.

— L'arbitrage des neurones postsynaptiques en simulation ne prend pas en compte la période T_{clk} d'arbitrage contrairement aux simulations précédentes [26]. Dans le cas où plusieurs neurones postsynaptiques émettaient un événement en même temps en simulation, celui à l'index le plus faible était sélectionné. Des simulations plus récentes ont mis en évidence que cet arbitrage n'impactait pas de manière notable les performances [27].

— Le nombre de tirs finaux utilisés pour l'étiquetage est de 10 et si 60 % de ces 10 derniers tirs appartient à une même classe, elle est alors choisie comme label.

Une simulation de référence dans un cas idéal sans variabilité (C_{mem} =1 pF, ϵ =0 mV) avec la règle 1P1D a été réalisée et est illustrée sur la FIGURE 2.16 et résulte en un taux de reconnaissance de 92.4 % (moyenne de 5 simulations pour 5 distributions initiales uniformes des poids synaptiques), ce qui est proche des résultats de Iyer et Basu [109] qui obtiennent 93.68 % pour ces 3 classes avec 400 neurones de sorties contre 100 pour cette simulation.

On peut noter que les résultats des simulations dans le cas idéal sans variabilité pour le démonstrateur ULPEC présentent des taux de reconnaissance plus importants que les simulations réalisées lors de la définition des règles d'apprentissage de la partie précédente. Cela peut s'expliquer par la réduction du nombre de classes utilisées. En effet, ici, on ne cherche à reconnaître que 3 classes différentes comparées à 10 précédemment. Ainsi, pour un même nombre de neurones de sortie (100 neurones postsynaptiques), plus de neurones sont disponibles par classe, augmentant ainsi les capacités de reconnaissance.

26. Travaux réalisés sur une version antérieure du simulateur.
27. Des simulations avec et sans cet arbitrage sur la version plus récente du simulateur n'a pas mis en évidence de différence notable.

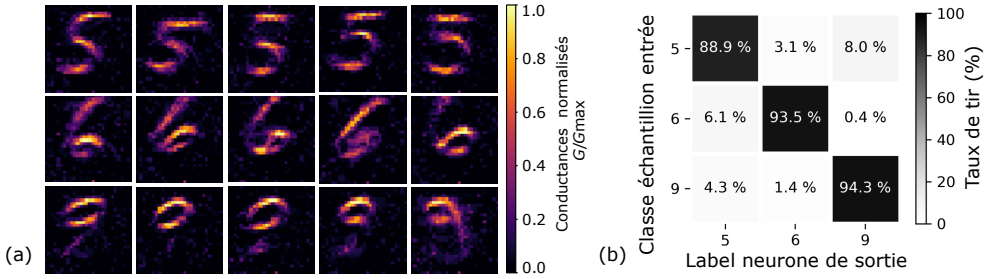


FIGURE 2.16 – Résultats de référence sans variabilité du réseau de neurones pour le projet ULPEC.
(a) Exemple de carte de conductances résultantes (images ordonnées manuellement). (b) Matrice de confusion. La matrice de confusion représente la quantité de réponses des neurones regroupés par label en fonction de la classe des échantillons présentés en entrée. Plus la diagonale est importante, plus le réseau différencie correctement les échantillons qui lui sont présentés. En effet, la diagonale correspond aux réponses pour un échantillon détecté comme un 5 pour la classe 5, comme un 6 pour la classe 6 et comme un 9 pour la classe 9.

2.3.1 Variabilité électronique des neurones postsynaptique

La variabilité électronique des neurones postsynaptiques intervient dès que l'on cherche a réaliser une implantation matérielle. Il existe plusieurs sources de variabilité électronique, mais dans le cadre de ces travaux nous nous intéressons à 2 composantes des neurones postsynaptiques que l'on considère comme primordiales pour leur fonctionnement. Il s'agit de :

— La capacité de membrane C_{mem}. La réalisation d'une capacité en électronique intégrée apporte généralement une grande variabilité. Pour la simulation nous avons choisi de considérer une capacité de membrane présentant une dispersion uniforme de C_{mem} de ±20% afin de mettre le réseau à l'épreuve d'une telle variabilité impactant la vitesse de charge des membranes des neurones postsynaptiques.

— L'*offset* de copie en tension ϵ du CCII (voir FIGURE 2.2). La valeur de cet *offset* est critique pour le bon fonctionnement du convoyeur de courant et donc du neurone postsynaptique. Si cet *offset* est négatif, il causera la génération d'un courant de stimulation permanent de la membrane ce qui, s'il est trop élevé (en valeur absolue), causera une activité spontanée des neurones postsynaptiques. Pour empêcher une telle activité spontanée des neurones postsynaptiques, les convoyeurs de courant réalisés dans ce projet ULPEC ont un circuit de correction de cet *offset* qui le place dans des valeurs positives les plus faibles possibles. De plus, pour l'architecture choisie et implémentée dans le cadre de ce projet, les convoyeurs de courant implémentés ne peuvent copier que du courant i_x positif[28] (un *offset* positif causerait ce courant i_x positif). En effet, si cet *offset* est positif, pour un convoyeur de courant ne copiant que les courants de stimulations positifs[29], il causera alors la génération d'un courant i_{cancel} qui « rognera » une partie (ou tout) du

28. Suivant la normalisation, il s'agit d'un courant sortant de la borne X du convoyeur de courant.
29. Un *offset* négatif générerait seul du courant de stimulation de la membrane.

courant de stimulation s'opposant ainsi à la charge du neurone postsynaptique et donc à son bon fonctionnement. Sans ce bridage de la copie en courant et sans activités présynaptiques, l'*offset* ϵ négatif causerait un courant d'inhibition permanent directement appliqué sur la membrane. Pour nos simulations nous choisissons de prendre des valeurs d'*offset* tirées aléatoirement suivant une loi uniforme dont la borne supérieure est prise dans l'intervalle [0 mV, 4 mV] [30]. Dans les travaux réalisés par le Dr Charly Meyer pour concevoir cette architecture en silicium, cet *offset* se trouve majoritairement entre [1 mV, 2 mV] [31].

La FIGURE 2.17 illustre ainsi les résultats obtenus avec le simulateur pour des ϵ choisis pour chaque neurone postsynaptique suivant une loi uniforme entre 0 mV [32] et une valeur comprise dans [0 mV, 4 mV] [33] dans le cas ou les capacités des membranes du réseau sont toute à 1 pF et lorsqu'elles sont choisies suivant une loi uniforme pour $C_{\text{mem}} \in [0.8\,\text{pF}, 1.2\,\text{pF}]$.

Pour C_{mem} = 1pF (pas de variabilité quant à la valeur de la capacité de membrane), lorsqu'on augmente la valeur maximale possible des *offsets* de copie en tension pour les neurones postsynaptiques, on observe que le nombre de neurones qui n'ont pas de label après apprentissage augmente (on considère donc qu'ils n'ont pas appris et sont désactivés). Quand ϵ augmente, le courant i_{cancel} augmente également, jusqu'à empêcher le neurone postsynaptique d'émettre un événement. Il se retrouve donc dans une situation où il ne peut pas apprendre et ne se fera donc pas étiqueter à la fin et on peut considérer que le neurone postsynaptique est mort (défaillant). Maintenir ϵ à des valeurs faibles est donc primordial pour ne pas perdre trop de neurones postsynaptiques. Malgré le grand nombre de neurones postsynaptiques non étiquetés pour des valeurs possibles de ϵ importantes, le taux de reconnaissance descend à seulement 87.3 % en moyenne pour les trois classes utilisées. Cela met en évidence que le réseau de neurones testé en simulation présente suffisamment de redondance en termes de neurones de sortie (100 neurones postsynaptiques) pour pouvoir continuer de reconnaître les entrées qu'on lui présente (3 classes d'échantillon).

Lorsque la variabilité sur la capacité de membrane est introduite, on constate qu'il n'y a quasiment aucune modification sur les performances du réseau à l'exception d'une légère chute du taux de reconnaissance (la courbe en pointillée pour des capacités de membrane tirées suivant la loi uniforme est légèrement en dessous de celle où toutes les capacités de membrane sont identiques). Par exemple, dans le pire scénario pour ϵ (ϵ choisi entre 0 mV et 4 mV), le taux de reconnaissance avec des capacités de membrane identiques est en moyenne de 87.3 % et lorsqu'elles sont tirées aléatoirement le taux de reconnaissance est en moyenne de 86.9 %.

30. La borne inférieure est fixée à 0 mV.
31. D'après des simulations de Monte Carlo sur Cadence® Virtuoso.
32. Borne inférieure de la distribution uniforme de ϵ fixé à 0 mV.
33. Borne supérieure de la distribution de ϵ.

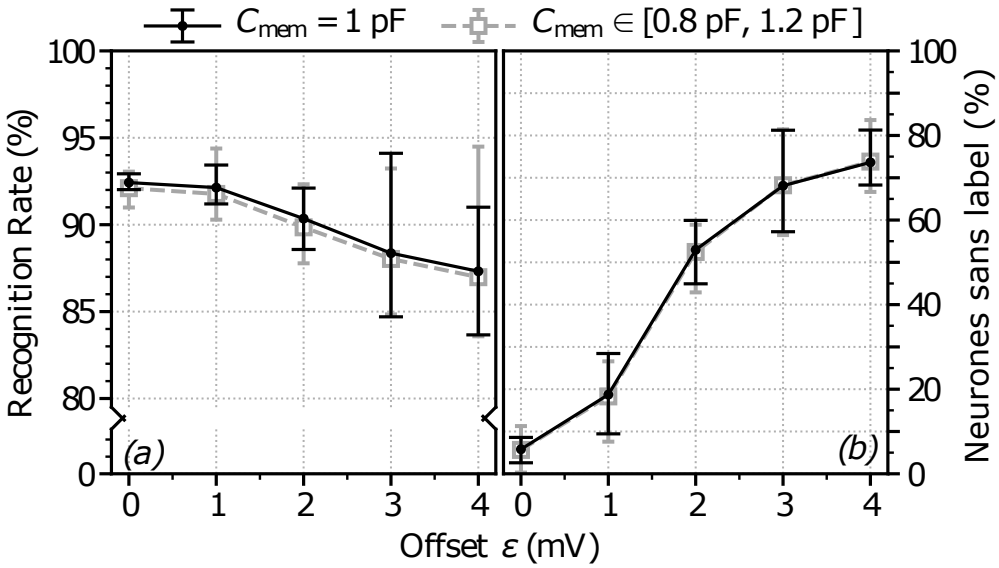


FIGURE 2.17 – Impact de la variabilité de la copie en tension avec et sans variabilité des capacités de membranes postsynaptiques.
L'axis *offset* représente la borne supérieure utilisée pour le tirage aléatoire (loi uniforme entre 0 mV et ϵ mV) de chaque neurone postsynaptique.

L'architecture de ce réseau de neurones semble donc être résiliente à la variabilité des membranes postsynaptiques. Le point critique de cette architecture reste donc l'*offset* de copie en tension qui doit être maintenu dans des valeurs positives et faibles. Il faut compter sur la redondance des neurones postsynaptiques en prévision de la présence de quelques neurones morts (défaillants) lors d'une implantation matérielle à cause (1) d'*offsets* trop importants[34] et (2) d'*offsets* dont la correction a échoué. C'est ce qu'a proposé le Dr Charly Meyer dans ses travaux de thèse avec un circuit dédié à la correction de cet *offset*.

2.3.2 Impact de l'initialisation des poids synaptiques

Dans toutes les simulations présentées jusqu'ici, la question des valeurs initiales de la matrice de conductances (crossbar de memristors) s'est résumée à une initialisation aléatoire suivant une loi uniforme continue entre G_{max} et G_{min}. Dans un système réel, il semble difficile de se retrouver dans une telle situation sans intervention particulière. En effet, dans le cas ou le crossbar de memristor :

— n'as jamais été utilisé, les memristors ont de fortes chances de se trouver dans un état particulier dit *virgin state* qui correspond généralement à un poids faible ;

— a déjà été utilisé pour une tâche précédente, les poids se trouvent alors dans des états qui dépendent du fonctionnement précédent et ne sont donc pas nécessairement répartis

34. Les limites données par les analyses de Monte Carlo correspondent à la majorité des cas, certains peuvent se trouver en dehors de ces valeurs.

de manière uniforme sur toute la matrice ;

— présente des conductances dont les valeurs sont plus ou moins élevées que prévu (pour des processus de fabrication non perfectionnés par exemple et/ou une variabilité importante des paramètres lors de la fabrication).

Ainsi, il semble difficile d'avoir naturellement un crossbar de memristors réels dont les poids sont répartis suivant une loi uniforme au démarrage de l'apprentissage. Tenter de programmer précisément chaque memristor pour atteindre une telle répartition du crossbar nécessiterait une architecture de mesure et des algorithmes de programmation complexes, ce qui s'oppose aux objectifs des architectures étudiées qui se veulent simples et facilement implantables, et qui ont été évités jusqu'ici notamment lors de l'étude des règles d'apprentissage faiblement supervisé.

L'apprentissage consistant à faire évoluer les poids synaptiques jusqu'à un état final d'équilibre[35], on s'intéresse alors à une métrique permettant de donner une information sur leur répartition et leur évolution au cours de l'apprentissage pour voir l'impact de différentes distributions initiales des poids avant apprentissage. Il s'agit de $\langle R_{network} \rangle$, la moyenne des résistances des synapses vue par les neurones postsynaptiques. Pour un neurone postsynaptique, $R_{network}$ correspond à la résistance équivalente des memristors (synapses) qui lui sont connectés.

Dans la situation de référence, la valeur de $\langle R_{network} \rangle$ de départ (conductances réparties de manière uniforme) est plus faible que celle d'arrivée[36] et donne un taux de reconnaissance moyen de 92.3 %. L'évolution de $\langle R_{network} \rangle$ pour cette situation au cours de l'apprentissage est illustrée par le tracé bleu de la figure FIGURE 2.18.b. On y observe une augmentation de $\langle R_{network} \rangle$ avant de tendre vers une valeur stationnaire. L'apprentissage du réseau est alors supposé stable. Si des les conductances du réseau sont 10 fois plus importantes que celles étudiées, il suffit d'adapter le facteur de copie en courant pour retrouver la même évolution de $\langle R_{network} \rangle$ pour un même taux de reconnaissance (92.3 %) comme illustré FIGURE 2.18.a où les tracés des $\langle R_{network} \rangle$ pour la plage de conductance d'origine et la plage multipliée par 10 évoluent de manière identique à la situation de référence. Ce comportement identique s'explique par les équations de calcul du simulateur et montre que si la plage des conductances utilisées est différente de celle prévue, il suffit d'adapter le facteur de copie en courant[37].

S'il suffit de changer le facteur de copie en courant pour s'adapter à des plages de conductances différentes lorsque la valeur de $\langle R_{network} \rangle$ initiale est plus faible que la valeur de $\langle R_{network} \rangle$ fi-

35. Il peut arriver que des neurones et les poids qui lui sont connectés ne parviennent pas à un état d'équilibre en ne parvenant pas à se spécialiser définitivement.

36. Se traduit part des conductances initiales moyenne plus fortes que les conductances cibles (après apprentissage).

37. C'est en partie pour cette raison que le circuit réalisé par le Dr Charly Meyer pour le réseau de neurones du projet ULPEC implémente 3 facteurs de copie en courant différents qui peuvent être choisis au besoin.

nale après apprentissage, la question se pose si la plage initiale des conductances se trouve dans une zone où la valeur de $\langle R_{network} \rangle$ initiale est supérieure à la valeur de $\langle R_{network} \rangle$ finale, c'est-à-dire que les poids initiaux du crossbar se situent majoritairement dans des valeurs de poids faibles (ou OFF). Si la valeur de $\langle R_{network} \rangle$ initiale est trop supérieure à la cible et que cette résistance équivalente se trouve ainsi dans une zone où elle ne stimule pas assez les neurones postsynaptiques, le réseau ne peut pas apprendre et sa valeur de $\langle R_{network} \rangle$ n'évolue donc pas, comme illustré avec la courbe pointillée rouge FIGURE 2.18.b. Cette une situation qui peut se produire avec des memristors qui n'ont jamais été utilisés et se trouvant donc dans leur *virgin state*. Pour remédier à cela, deux solutions sont envisagées ici :

— La première consiste à adapter le facteur de copie en courant au fur et à mesure qu'un neurone apprend. Ce mécanisme, à rapprocher d'une forme d'homéostasie, consiste à démarrer l'apprentissage du réseau de neurones avec des neurones dont le facteur K de copie en courant est fort (pour aider la stimulation si les poids sont faibles) et à le faire évoluer au fur et à mesure qu'un neurone émet des événements et qu'il apprend pour se rapprocher de la situation finale et stable du réseau. En simulation, le facteur K de chaque neurone démarre à $K = 1/12.5$, diminue de 0.001 à chaque tir du neurone et s'arrête à $K = 1/125$.

— L'autre solution envisagée consiste à programmer un motif simple sur les cartes de conductances pour augmenter la valeur initiale de $\langle R_{network} \rangle$. Par exemple pour chaque neurone postsynaptique, placer une synapse sur deux dans son état ON ($G_{\max}$) permet de se retrouver dans une situation de $\langle R_{network} \rangle$ similaire à celle obtenue lorsque les synapses sont réparties de manière uniforme.

Pour étudier ces 2 méthodes, le réseau de neurones a été simulé en utilisant des poids initiaux tirés aléatoirement entre $G_{\min}$ et $G_{\max}/10$, plaçant donc les simulations dans une zone où leurs poids initiaux sont faibles.

Si la première méthode, dont l'évolution de $\langle R_{network} \rangle$ est illustrée par la courbe orange FIGURE 2.18.b, démarre avec une valeur de $\langle R_{network} \rangle$ similaire à la situation où l'apprentissage n'avait pas fonctionné, l'adaptation en cours de l'apprentissage du facteur de copie en courant pour compenser un démarrage avec des poids faibles permet d'arriver à une situation finale de répartition des poids similaires à la situation de référence et donne un réseau de neurones qui a appris avec, pour ces simulations, un taux de reconnaissance moyen de 92.4 %.

La deuxième méthode qui consiste à programmer de manière simple les poids initiaux, donne une évolution quasi identique de $\langle R_{network} \rangle$ (courbe noire FIGURE 2.18) à celle où les poids sont initialisés de manière uniforme entre $G_{\max}$ et $G_{\min}$ pour un taux de reconnaissance moyen quasi identique de 92.1 %.

Les deux solutions proposées pour pallier le problème de conductances initiales trop faibles

pour démarrer l'apprentissage sont fonctionnelles d'après les résultats de simulation obtenus. Cependant, dans le cadre du projet ULPEC, c'est la deuxième solution qui a été retenue puisqu'elle demande seulement de programmer un poids sur deux à sa valeur maximale, ce qui peut par exemple se faire par l'apprentissage répété d'une image d'échiquier. La première nécessite quant à elle de pouvoir faire évoluer au cours du temps le facteur de copie en courant et que les neurones postsynaptiques disposent d'un convoyeur de courant qui peut être modifié pour proposer ces facteurs de copie en courant sur une vaste plage de valeurs et avec une granularité suffisante.

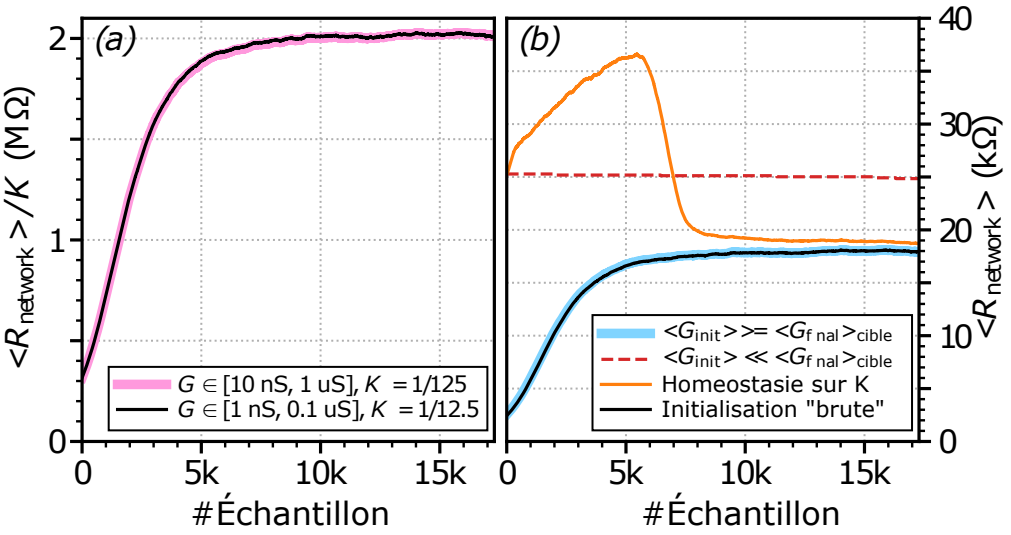


FIGURE 2.18 – Évolution de $\langle R_{network} \rangle$ au cours de l'apprentissage.
Chaque résultat est la moyenne de 5 simulations (1 epoch chacune) avec ϵ =0 mV et C_{mem} =1 pF. (a) $\langle R_{network} \rangle$ normalisée par le facteur de copie en courant K. La courbe rose représente l'évolution avec des conductances initiales tirées entre 10 nS et 1 µS avec K = 1/125 (RR moyen de 92.3 %). La courbe noire représente l'évolution avec des conductances initiales tirées entre 1 nS et 0.1 µS et K = 1/12.5 (RR moyen de 92.3 %). (b) $\langle R_{network} \rangle$ pour différente version de l'initialisation du réseau par rapport aux conductances finales cibles. La courbe bleue représente la situation de référence (même courbe que celle en rose à gauche). La courbe pointillée rouge représente le cas ou les conductances de départ sont bien supérieures aux conductances cibles (RR moyen de 0 %). Courbe orange : Les conductances de départ sont sensiblement supérieure aux conductances cibles, mais il y a un mécanisme d'homéostasie sur le facteur de copie en courant (RR moyen de 92.4 %). Courbe noire : Les conductances de départ sont initialement bien supérieures aux conductances cibles puis, avant lancement de l'apprentissage, 50 % d'entre elle ont été mise à G_{min} (RR moyen de 92.1 %).

2.3.3 Synthèse

Les simulations réalisées dans le cadre du projet ULPEC ont donc mis en évidence plusieurs points importants si l'on souhaite implémenter une architecture de ce type :

— Si les neurones postsynaptiques sont résiliants à la variabilité de leur membrane, ils sont

en revanche relativement sensibles à la variabilité de leur *offset* de copie du courant en entrée. S'assurer que ce dernier soit minime et dans des plages de valeurs positives est une manière de garantir qu'un maximum des neurones postsynaptiques implantés dans un circuit soient fonctionnels et correctement spécialisés à la fin de la phase d'apprentissage.

— La plage initiale des poids synaptiques avant apprentissage est également critique pour que le réseau apprenne. Pour éviter une situation où le réseau ne peut pas apprendre parce que les synapses initiales ne permettent pas de stimuler suffisamment les neurones postsynaptiques, il est par exemple possible d'initialiser ses poids synaptiques en lui faisant apprendre des motifs de départs tels que des échiquiers.

2.4 Conclusion

Le simulateur en Python, dont l'architecture générale modulaire est illustrée ANNEXE A, a été mis au point lors de ces travaux de thèse pour produire les résultats de ce chapitre, puis a été amélioré au sein de l'équipe de recherche pour le rendre plus flexible. Il a pour vocation :

— D'aider à étudier des réseaux de neurones événementiels en vue d'une implantation matérielle. Cela a été le cas pour le projet ULPEC. La simulation a permis de mettre en évidence certaines contraintes matérielles comme l'*offset* de copie en tension et la nécessité d'une initialisation des poids synaptiques avant l'utilisation du réseau de neurones.

— De pouvoir définir et étudier différentes règles d'apprentissage par le biais de la simulation. Cela a été détaillé dans la partie 2 de ce chapitre ou plusieurs règles d'apprentissage ont été définies pour répondre aux contraintes des données que l'on souhaite traiter. Des règles permettant de s'affranchir des fenêtres d'apprentissage issues de la STDP ont été proposées (règles iPjD) et ont été validées en simulation. Dans cette partie 2 ont également été introduites, testées et validées des versions d'apprentissage faiblement supervisé (R-1P1D) qui se veulent peu complexes et donc facilement intégrables.

Même si ces travaux de thèse se sont restreints à une architecture de réseaux de neurones événementiels à une couche *all-to-all*, ce simulateur a été mis au point de manière modulaire afin de pouvoir le modifier facilement pour construire des architectures plus complexes, changer le type de neurones de sorties ou les modèles de synapses. Concernant la modification du modèle de neurone de sortie, cette fonction a été utilisée pour la vérification expérimentale du modèle de neurone LIF à résistance de décharge dans le CHAPITRE 4. Concernant le modèle de synapses, une continuité de ces travaux de thèse consisterait à prélever des caractéristiques de composants memristifs réels et à les injecter dans le simulateur pour réaliser des études plus poussées et plus précises pour pouvoir définir par la simulation les paramètres à utiliser pour une technologie de memristors donnée. La simulation aurait donc pour objectif, dans ce cas-là, de déterminer à l'avance les capacités ou non d'apprentissage de l'implantation matérielle qui sera réalisée et d'en anticiper les performances le cas échéant.

Chapitre 3

Conception d'une plateforme 81 × 10

Ce chapitre s'inscrit dans le deuxième axe de ces travaux de thèse. Il s'agit de la mise en place d'une plateforme d'expérimentation permettant de tester une architecture de réseau de neurones événementiels *all-to-all* analogique à une seule couche pour pouvoir aisément déployer et tester différentes règles d'apprentissage (telles que celles étudiées dans le CHAPITRE 2) et rendre possible la caractérisation de crossbars de memristors ferroélectriques physiques. Cette plateforme a ainsi été développée et conçue pour deux utilisations distinctes :

— Permettre de vérifier les différentes règles d'apprentissage définies en simulation. Les simulations ont donc un rôle exploratoire et l'utilisation de la plateforme a un rôle de démonstration et de vérification expérimentale.

— Permettre d'extraire des paramètres physiques de composants memristifs pour pouvoir les intégrer et les utiliser dans le simulateur. L'intégration de caractéristiques physiques réelles dans le simulateur vise à rendre les simulations plus précises et plus fidèles aux intégrations matérielles envisagées.

Cette plateforme implémente 81 entrées présynaptiques (permettant par exemple d'utiliser des images 9 × 9) et 10 neurones postsynaptiques résultant en un réseau de neurones événementiels 81 × 10. Elle ainsi capable de piloter et caractériser un crossbar de 810 memristors.

Ce chapitre est dédié à une description de toutes les cartes électroniques constituant cette plateforme ainsi que du contrôle numérique permettant de la piloter. Ce contrôle numérique est assuré par une partie implémentée sur FPGA écrite en VHDL, et une autre compilée sur processeur ARM écrite en langage C. La première partie de ce chapitre donnera une vue d'ensemble de la plateforme et de l'agencement des différentes cartes électroniques qui la constituent. La deuxième partie abordera les deux cartes électroniques conçues pendant ces travaux de thèse (cartes MIRA et VCCS2) qui constituent principalement la partie analogique de la plateforme. La troisième partie concerne la carte ADAPT, carte développée lors de ces travaux de thèse afin de communiquer et transmettre divers signaux numériques entre la carte de contrôle et la carte MIRA. La dernière partie sera consacrée au contrôle numérique implémenté dans la carte de contrôle ZCU102.

Si ce CHAPITRE 3 aborde le rôle de chacune des cartes électroniques constituant cette plateforme ainsi que les blocs élémentaires (électroniques ou numériques) les constituants, le détail technique et le fonctionnement détaillé de chacun de ces blocs ne seront pas donnés dans ce chapitre et ce manuscrit puisque ce dernier a pour objectif d'expliquer le fonctionnement de la plateforme sans en être la documentation technique.

3.1 Architecture mixte : une plateforme à 4 cartes

La plateforme 81 × 10 a été créée pour répondre à plusieurs attentes qui dépassent la simple mise en œuvre d'un réseau de neurones événementiels. Si le démonstrateur ULPEC, étudié et défini via les simulations présentées au CHAPITRE 2, a pour objectif de prouver la faisabilité de l'intégration de neurones en silicium, d'un capteur neuromorphique et de synapses memristives dans une même puce pour la réalisation d'un capteur intelligent et intégré, cette plateforme développée entièrement durant ces travaux de thèse, de la conception de cartes électroniques à l'écriture des algorithmes de contrôle en C en passant par l'implémentation de blocs numérique en VHDL sur FPGA (DCB étendu [1]), a pour objectif d'être bien plus flexible pour :

1. Permettre d'étudier différentes règles d'apprentissage issues de la simulation et de démontrer leur fonctionnement de manière expérimentale pour un réseau de neurones événementiels (une couche, 81 × 10, *all-to-all*). C'est pour cette raison que la partie DCB étendu a été implémentée sur FPGA permettant ainsi de le modifier si nécessaire (par reconfiguration de ce dernier).
2. Permettre de piloter et de caractériser des memristors de manière automatique à grande échelle pour l'extraction de leurs paramètres. C'est pour cela que l'architecture est équipée de systèmes de mesure des courants synaptiques pour permettre de caractériser une matrice de 810 memristors (crossbar de 81 × 10).
3. Pouvoir utiliser la plateforme de manière automatisée pour diverses expérimentations. C'est pour cela que les algorithmes de contrôle et d'expérimentation ont été implémentés en langage C [2] sur processeur ARM en lien étroit avec le FPGA qui implémente le DCB.
4. Permettre la réparation/modification à moindre coût des différents blocs constituant la plateforme afin de lui assurer une longue longévité et période d'utilisation. C'est pour cette raison que la plateforme a été réalisée en électronique discrète et non par la conception d'une puce. On peut par exemple changer aisément la résistance ou capacité de membrane si nécessaire.

L'architecture de cette plateforme est illustrée sur la FIGURE 3.1 et elle est composée de 4 cartes électroniques :

— La carte ZCU102 [3]. Carte de développement Xilinx™ ZCU102 intégrant un Zynq® UltraScale+ MPSoCs [116] (puce électronique co-intégrant un CPU ARM et un FPGA) qui est utilisée ici pour piloter la plateforme au moyen d'algorithmes en C exécutés dans la partie CPU et grâce au DCB étendu implémenté dans la partie FPGA.

1. Le DCB (voir CHAPITRE 2) de cette plateforme est dit « étendu » puisqu'il présente plus de fonctionnalités que celles qui ont pu être introduites pour le démonstrateur ULPEC (qui sert de référence). Ces fonctionnalités complémentaires sont détaillées dans la partie 3.4 Carte ZCU102.
2. L'écriture des algorithmes de contrôle en langue C et non en VHDL permet d'écrire ou de modifier ces derniers plus rapidement.
3. Seule carte électronique de la plateforme 81 × 10 qui n'a pas été conçue pendant ces travaux de thèse.

— La carte MIRA. Cœur de la plateforme, elle contient le réseau de neurones événementiels 81×10 ainsi que les circuits permettant la caractérisation du crossbar de 810 memristors.

— La carte ADAPT. Carte d'interface (pont) entre la carte MIRA et la carte ZCU102 pour pallier les différents nombres d'entrées et de sorties de ces deux cartes ainsi que leurs différents niveaux de tension logique de fonctionnement.

— La carte VCCS2. Carte complémentaire (optionnelle) de la plateforme qui a été créée a posteriori pour ajuster ou compenser les courants de sortie des CCII+ des neurones post-synaptiques.

Les cartes MIRA et VCCS2 implémentent la partie analogique de la plateforme et les cartes ADAPT et ZCU102 implémentent la partie numérique de contrôle et de communication de ces dernières.

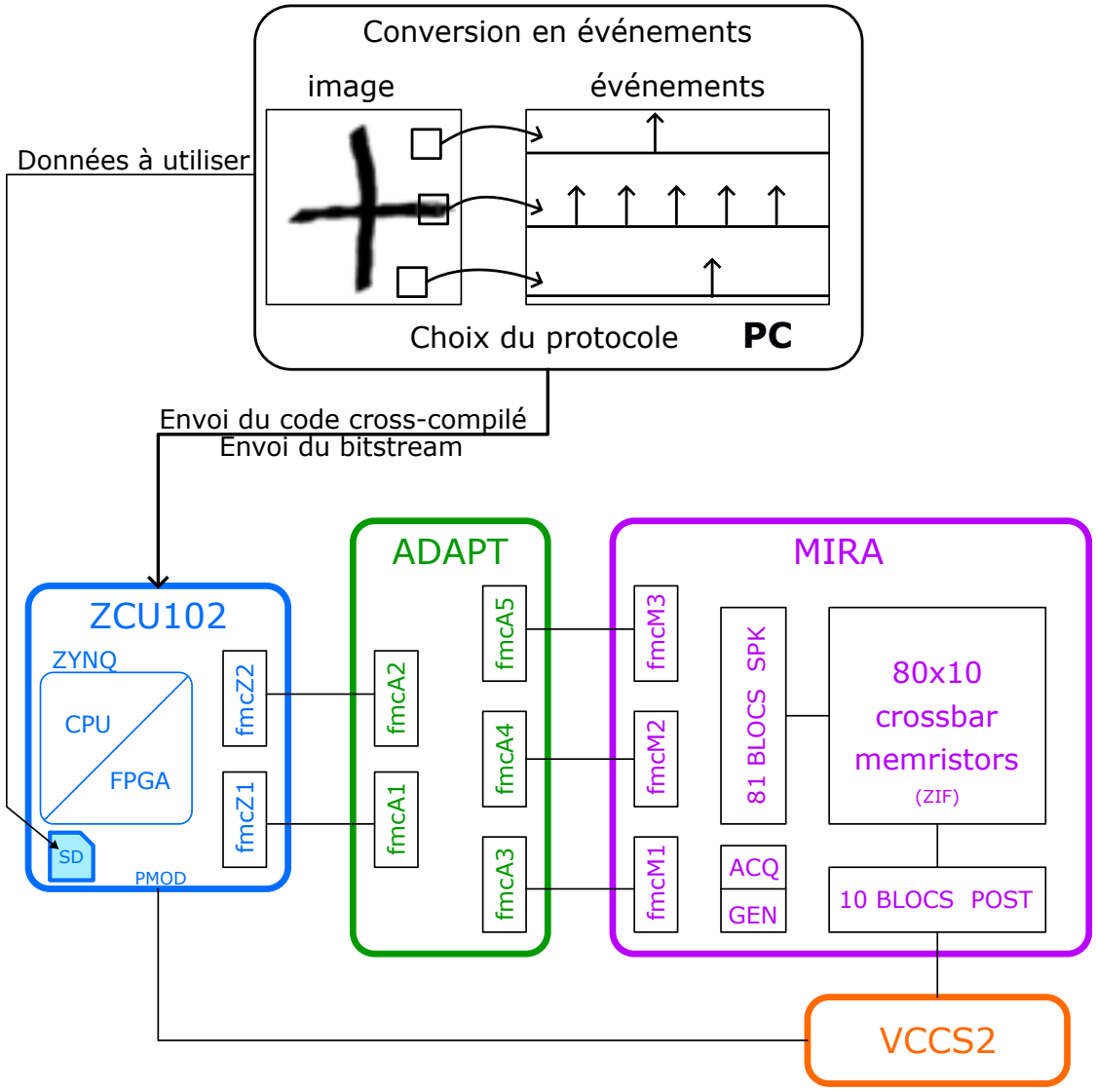


FIGURE 3.1 – Architecture de la plateforme 81 × 10.
La plateforme est constituée de 4 cartes électroniques (ZCU102, ADAPT, MIRA et VCCS2). Un ordinateur (PC) est nécessaire pour compiler le code de l'expérimentation et l'envoyer avec le *bitstream* (fichier d'implémentation numérique du FPGA) à la ZCU102 pour exécution. Si l'expérimentation nécessite des données (échantillons pour apprentissage et test), ces dernières sont générées par l'ordinateur et mises à disposition de la ZCU102 via une carte SD qui est également utilisée pour enregistrer les résultats obtenus. Les connexions entre les 3 cartes principales de la plateforme sont assurées par des connecteurs FMC. La connexion avec entre la carte VCCS2 et la ZCU102 est assurée par un connecteur PMOD. La carte MIRA qui contient le cœur de la plateforme est constituée de 4 blocs élémentaires (SPK, GEN, ACQ et POST) détaillés par la suite. Les synapses artificielles (memristors) sont connectées à la carte MIRA par l'intermédiaire d'un connecteur ZIF (*Zero Insertion Force*) permettant ainsi de changer aisément le crossbar de memristors utilisé par la plateforme.

3.2 Cartes MIRA et VCCS : Partie calcul analogique

La carte MIRA qui correspond à la partie calcul analogique (réseau de neurones) de la plateforme est constituée de 4 types de blocs élémentaires (SPK, POST, GEN et ACQ) qui assurent son fonctionnement.

La carte VCCS2 (optionnelle) est constituée d'un même bloc élémentaire, une source de courant pilotée en tension (VCCS [4]), dupliqué 10 fois pour les 10 blocs POST (blocs postsynaptiques équipés de CCII+) de la carte MIRA.

La carte MIRA nécessite 3 alimentations externes :

— Alimentation pour la partie numérique de 3.3 V.

— Alimentation symétrique pour la partie analogique de ±5 V.

— Alimentation symétrique pour la partie analogique de ±2.5 V.

La carte VCCS2 nécessite 3 alimentations externes :

— Alimentation pour la partie numérique de 3.3 V.

— Alimentation symétrique pour la partie analogique de ±5 V.

— Alimentation pour une tension de référence de 2.5 V.

On peut noter que sur la carte MIRA illustrée sur la FIGURE 3.2, les alimentations numériques et analogiques disposent de plans de masse séparés physiquement sur la carte pour s'assurer que des courants de retour ne traversent pas de la partie numérique vers la partie analogique (à cause de la proximité de parties numériques et analogiques de certains circuits sur la carte), ce qui pourrait perturber le fonctionnement de la partie analogique (connectée aux memristors ferroélectriques) de la carte MIRA.

4. De l'anglais *Voltage-Controlled Current Source.*

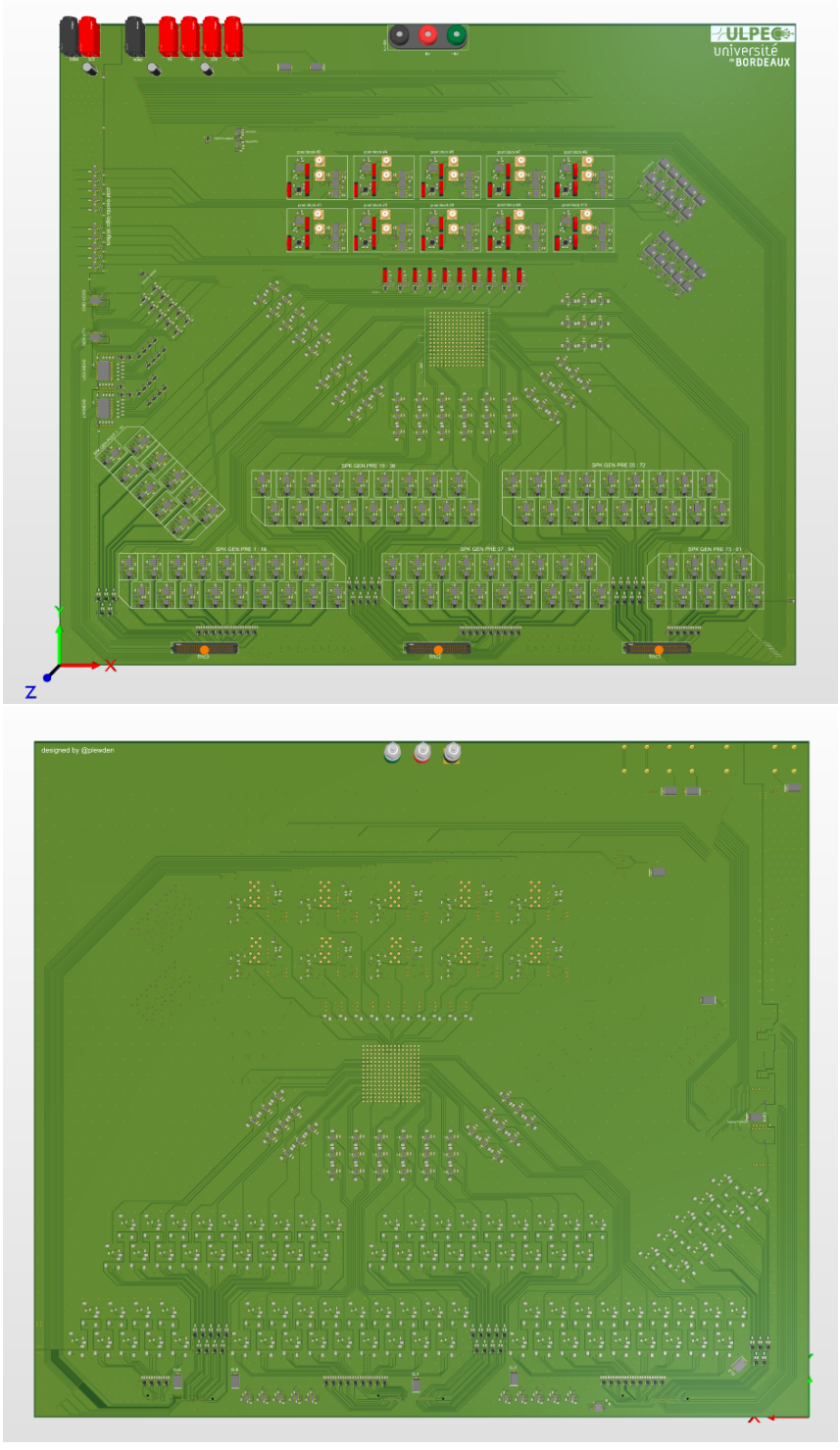


FIGURE 3.2 – Vues de dessus et de dessous de la carte MIRA.
Vues obtenues sur le logiciel de conception Altium Designer. Il s'agit d'une carte réalisée en 6 couches. Les connexions au centre correspondent à l'emplacement du ZIF (non illustré sur ces images) de connexion des memristors. Les 3 connecteurs au sud de la carte sur la vue de dessus (recouvert d'une pastille orange) sont les connecteurs FMC mâles de la carte MIRA pour la connecter à la carte ADAPT.
Cette carte fait 49.8 cm×43.4 cm.

3.2.1 Blocs SPK (neurones présynaptiques)

La plateforme est équipée de 81 neurones présynaptiques et ces derniers doivent assurer la génération des signaux à appliquer sur le crossbar de memristors. Afin de conserver la flexibilité du type de formes d'onde à appliquer sur les memristors, que ce soit pour tester différentes formes d'onde pour l'inférence et la mesure ou utiliser des formes exotiques pour modifier leur poids, le bloc SPK repose sur l'utilisation du DAC [5] AD5443 à interface série dont les signaux numériques de contrôle et de génération des formes d'onde sont générés par la carte ZCU102. La structure de ce bloc SPK est illustrée sur la FIGURE 3.3 et les entrées/sorties de ce dernier sont données TABLE 3.1.

Les 3 signaux numériques de contrôle (SCLK, DATA, SYNC) sont issus d'émetteurs numériques (triple triggers de Schmitt inverseurs [6]) situés proches des entrées physiques de la carte MIRA (FMCM1, FMCM2 et FMCM3 sur la FIGURE 3.1). Ces émetteurs réceptionnent les signaux numériques reçus par la carte MIRA avant de les envoyer aux blocs SPK. Pour ces émetteurs transmettant les signaux de contrôle des blocs SPK présynaptiques, les signaux DATA et SYNC sont uniques par bloc présynaptique pour permettre la génération de formes d'onde différentes par bloc SPK. Afin de réduire le nombre de signaux nécessaires pour piloter cette carte, de limiter la capacité de charge des circuits devant la piloter et d'avoir des chemins de routage courts, les entrées des émetteurs des signaux SCLK (en entrée de la carte MIRA) sont communes par groupe de 4 ou 5 générateurs SPK présynaptiques [7].

Ainsi, pour les 81 présynaptiques de la carte MIRA, il faut 81 signaux DATA, 81 signaux SYNC et 18 signaux SCLK. Soit 180 signaux [8] dédiés au pilotage des blocs SPK présynaptiques.

La tension de référence $V_{\text{ref pre}}$ des DAC est commune aux blocs SPK présynaptiques et est générée par une sortie du bloc GEN (décrit par la suite) afin de permettre le contrôle et la modification de la tension de pleine échelle des formes d'onde générées.

Le circuit électronique de ce bloc SPK est donné en ANNEXE B.

5. De l'anglais *Digital-to-Analog Converter* ou Convertisseur Numérique Analogique.
6. Consultez la documentation technique du composant SN74LVC3G14 pour plus d'information.
7. Ce découpage par sous-groupes de 4 et 5 générateurs permet de constituer des groupements de 2 signaux SCLK par paquet de 9 blocs présynaptiques. Ce groupement par paire est lié au design de la carte ADAPT.
8. Ne prends pas en compte les signaux nécessaires à la génération de $V_{\text{ref pre}}$ qui est issu d'un autre bloc de la carte.

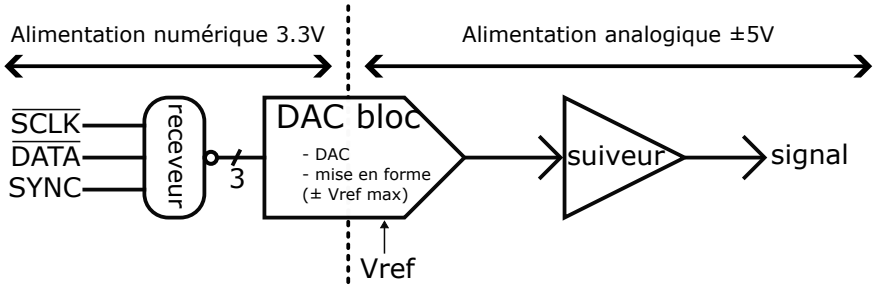


FIGURE 3.3 – Structure d'un bloc SPK.
Il comprend un receveur (triple triggers de Schmitt inverseur SN74LVC3G14), un Convertiseur Numérique Analogique (DAC AD5443), son circuit de mise en forme ainsi qu'un suiveur. Le receveur régénère les signaux numériques de contrôle (SCLK, DATA et SYNC) qui définissent la forme d'onde à appliquer. Le bloc DAC génère la forme d'onde voulue entre $\pm V_{\text{ref}}$(= $V_{\text{ref pre}}$ pour les présynaptiques) pour sa pleine échelle (écart entre la valeur minimum et maximum possible du DAC). Le suiveur assure le pilotage de la charge qui, pour les neurones présynaptique, correspond aux lignes présynaptiques du crossbar de memristors.

TABLE 3.1 – Entrées/sorties du bloc SPK.

Signal	Description	Type	Tension
SCLK	Signal d'horloge du DAC	Numérique	0 V/3.3 V
SYNC	Signal de synchronisation du DAC	Numérique	0 V/3.3 V
DATA	Signal des données séries du DAC	Numérique	0 V/3.3 V
V_{ref}	Tension de référence du DAC pour la mise en forme	Analogique	0 V/3.3 V
signal	Forme d'onde générée par le bloc SPK	Analogique	$\pm V_{\text{ref}}$

3.2.2 Blocs POST (neurones postsynaptiques multifonctions)

Les blocs POST contiennent la partie postsynaptique de l'architecture. Ils sont considérés « multifonctions » puisque, en plus d'assurer le rôle de neurones postsynaptiques LIF, ils permettent l'acquisition du courant postsynaptique pour la mesure et la caractérisation ainsi que l'acquisition de la tension de membrane grâce à des sous-blocs qui peuvent être activés ou désactivés par des interrupteurs analogiques contrôlés numériquement suivant les besoins de l'expérimentation en cours. La structure de ce bloc, ainsi que les 5 sous-blocs le constituant et les connexions qu'ils ont entre eux, est illustrée sur la FIGURE 3.4 et son schéma électrique simplifié est représenté sur la FIGURE 3.5.

Les cinq sous-blocs qui le constituent sont :

— SPK : Même circuit SPK que celui utilisé pour les neurones présynaptiques. Il est utilisé ici pour la génération des formes d'onde postsynaptiques à appliquer sur les memristors par l'intermédiaire du CCII+.

— LIF : Partie neurone du bloc POST. Il s'agit du deuxième modèle de neurone LIF, le neu-

rone LIF à résistance de décharge, présenté dans la partie 1.2.1 Le choix des neurones : Le neurone LIF. Ce modèle a été choisi pour la carte MIRA puisque, en électronique discrète, il est plus simple à réaliser (simple résistance en parallèle de la capacité de membrane) que celui nécessitant une source de courant de décharge.

— MSU LIN : Circuit permettant de récolter la tension de membrane et de l'adapter (en plage de tension de manière linéaire) en une tension V_{lin} pour une acquisition numérique.

— CCII+ : Étage d'entrée du bloc POST. Réalisé avec le composant OPA861, il assure le même rôle que le CCII+ présenté dans les simulations du CHAPITRE 2. Cependant, contrairement aux CCII+ utilisés jusqu'à présent, ce dernier permet la copie des courants positifs et négatifs en entrée. Cette copie des courants positifs et négatifs n'est pas un problème contrairement au projet ULPEC puisque le LIF utilisé pour réaliser les neurones postsynaptiques est un LIF à résistance de décharge [9].

— MSU LOG : Circuit qui permet de récolter et de mesurer le courant en sortie du CCII+, le convertissant de manière logarithmique en une tension V_{log} pour une acquisition numérique. C'est ce sous-bloc qui est utilisé pour estimer les poids synaptiques.

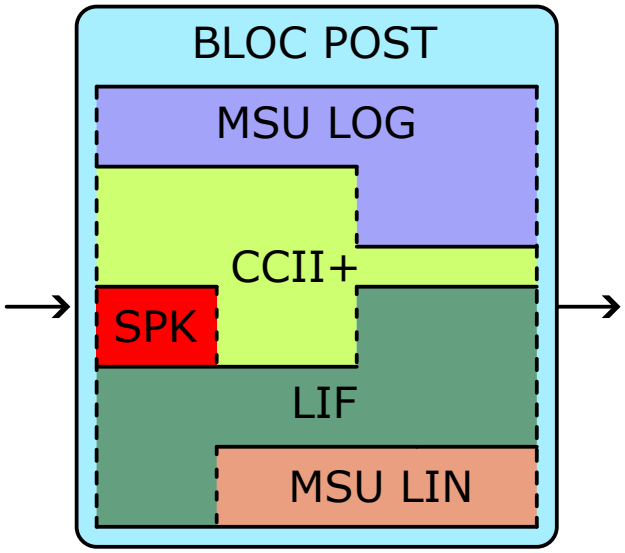


FIGURE 3.4 – Structure du bloc POST.
Il est composé de 5 sous-blocs : SPK, CCII+, MSU LOG, MSU LIN et LOG. Les sous-blocs qui communiquent entre eux sont connectés par une paroi verticale pointillée. Des 5 sous-blocs, seul SPK n'a pas de connexion en sortie vers l'extérieur du bloc POST (son signal est uniquement connecté au CCII+ (sa borne Y)) et seul MSU LIN n'a pas de connexion en entrée depuis l'extérieur du bloc POST.

9. Si un courant d'offset non nul (positif ou négatif) est copié par le CCII+ cela se traduira par une tension de repos correspondant à la résistance de décharge de la membrane multiplié par ce courant. Ce décalage peut être corrigé avec l'aide de la carte VCCS2 ou pris en compte lors du choix de la tension de seuil du neurone.

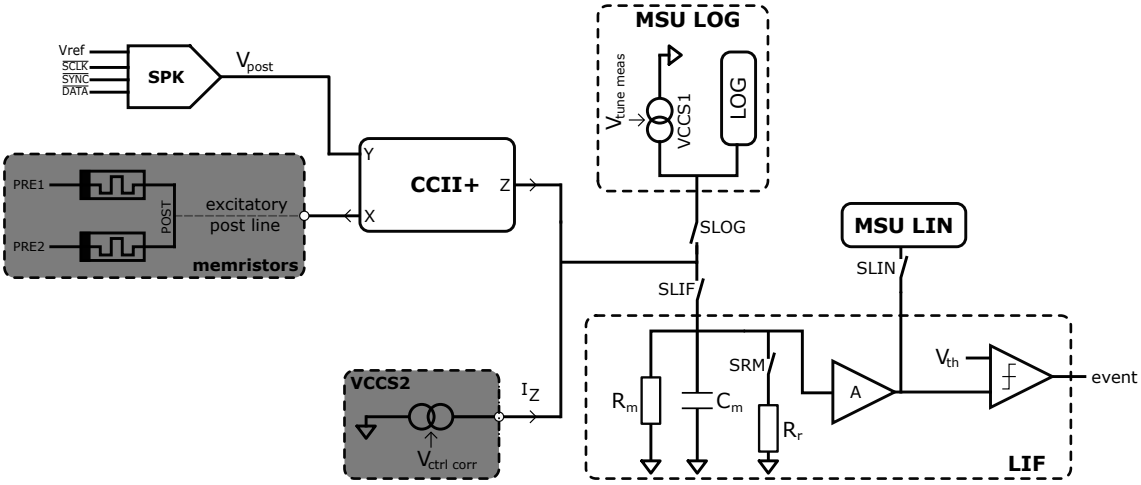


FIGURE 3.5 – Schéma électrique simplifié du bloc POST.
Les éléments grisés (**memristors** et **VCCS2**) représentent des éléments indiqués pour la compréhension mais qui n'appartiennent pas au bloc POST. On peut retrouver la présence des 4 interrupteurs (composant ADG1411 : quadruple interrupteur) qui permettent de connecter ou non certains des sous-blocs constituant le bloc POST.

Les 5 sous-blocs constituant le bloc POST permettent de l'utiliser suivant 3 modes distincts :

1. Le mode neurone LIF.

On utilise ce mode lorsque l'on souhaite utiliser la plateforme comme un réseau de neurones. Pour ce mode, l'interrupteur SLOG doit être ouvert et l'interrupteur SLIF doit être passant. L'interrupteur SLIN ne perturbe pas son fonctionnement et l'interrupteur SRM est utilisé pour remettre à zéro la membrane. Ce mode fait ainsi appel à 3 des 5 sous-blocs qui constituent le bloc POST :

- CCII+ : Convoyeur de courant utilisé comme interface avec les memristors.
- SPK : Génère la tension V_{post} à appliquer sur la borne postsynaptique des memristors.
- LIF : Module neurone LIF à résistance de décharge constitué d'une capacité de membrane C_{m} =100 nF, d'une résistance de décharge R_{m} =10 kΩ et d'un module de remise à 0 V [10] de la membrane activé par l'interrupteur SRM et constitué d'une résistance faible R_{r} =200 Ω permettant de revenir rapidement à la tension de repos de la membrane. La tension de membrane du LIF est amplifiée par le circuit d'amplification de gain A avant d'être comparée au module de tir (comparateur) pour émettre un événement si cette tension de membrane amplifiée dépasse la tension de seuil V_{th}. Le signal de tir (event) est récolté par la ZCU102 pour déclencher la phase d'apprentissage si nécessaire. Il est important de noter que les sources de courant VCCS2 présentent une impédance de 10 kΩ

10. Plus précisément, cette résistance ramène à la tension de repos correspondant à cette résistance de décharge multipliée par le courant d'offset en sortie du CCII+ (l'influence de la résistance de membrane R_{m} de 10 kΩ en parallèle est négligeable). Cette tension se veut proche de 0 V.

ce qui résulte en une résistance de la membrane du neurone de 5 kΩ si cette carte VCCS2 est utilisée.

La plateforme implémente des neurones LIF à résistance de décharge puisque seule une résistance passive, facilement remplaçable et peu coûteuse en surface pour de l'électronique discrète est nécessaire.

2. Le mode estimation de poids synaptiques.

On utilise ce mode si l'on souhaite utiliser la plateforme pour mesurer le courant traversant les memristors du crossbar et ainsi estimer leur résistance électrique. Pour ce mode, l'interrupteur SLIF doit être ouvert et l'interrupteur SLOG doit être passant. Les interrupteurs SRM et SLIN ne perturbent pas son fonctionnement. Ce mode fait ainsi appel à 3 des 5 sous-blocs qui constituent le bloc POST :

- CCII+ : Utilisé comme interface avec les memristors.
- SPK : Génère la tension V_{post} à appliquer sur la borne postsynaptique des memristors.
- MSU LOG : Sous-bloc de mesure logarithmique illustré sur la FIGURE 3.6. Ce sous-bloc collecte le courant I_Z issu du CCII+ et l'injecte dans le circuit LOG (basé sur l'utilisation du composant ADL5303 [11]) qui convertit le courant positif collecté en tension. Ce sous-bloc contient un circuit VCCS1 [12], qui permet de modifier le courant de polarisation du circuit pour le mettre dans une plage positive et l'ajuster pour améliorer la précision de la mesure. L'acquisition numérique de la tension V_{log} lors de la mesure est réalisée par un autre bloc de la carte MIRA (le bloc ACQ) piloté par la ZCU102.

11. Convertiseur de courant en tension selon un échelle logarithmique, permettant une large plage de mesure de courant en entrée.

12. Source de courant bidirectionnelle qui utilise le même montage électronique que VCCS2 mais qui est réalisée avec des valeurs de résistances et des composants différents.

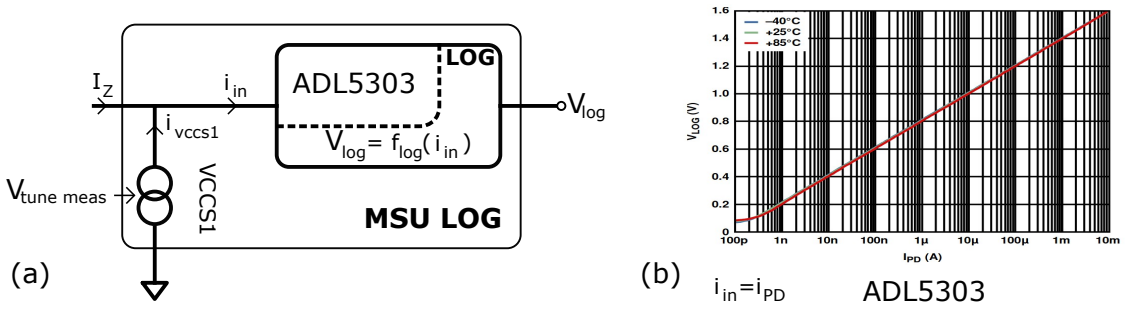


FIGURE 3.6 – Schéma électrique simplifié du sous-bloc MSU LIN.
(a) Schéma électrique simplifié du sous-bloc MSU LIN. Il est constitué d'un source de courant pilotée en tension (VCCS1) et d'un bloc de conversion logartitmique du courant en tension ne fonctionnant que pour des courant d'entrée positifs. VCCS1 est utilisée pour ajuster le point de mesure en courant. (b) Caractéristique de la conversion du courant en tension extraite de la documentation technique du composant ADL5303. Le circuit LOG implémenté sur la MIRA peut donner une pente différente de la caractéristique issue de la documentation technique qui est de 500 mV/décade au lieu des 200 mV/décade illustré ici. Cette pente peut-être choisie par un interrupteur mécanique sur la carte MIRA.

3. Le mode estimation de la tension de membrane.

On utilise ce mode si l'on souhaite utiliser la plateforme pour acquérir la valeur de la tension de membrane d'un neurone postsynaptique. Pour ce mode, l'interrupteur SLOG doit être ouvert et les interrupteurs SLIF et SLIN doivent être passants. L'interrupteur SRM est utilisé pour remettre à zéro la membrane. Ce mode fait ainsi appel à 4 des 5 sous-blocs qui constituent le bloc POST :

- CCII+ : Utilisé comme interface avec les memristors.
- SPK : Génère la tension V_{post} à appliquer sur la borne postsynaptique des memristors.
- LIF : Module neurone LIF à résistance de décharge.
- MSU LIN : Sous-bloc de mesure linéaire de la tension de membrane amplifiée. Ce sous-bloc récolte la tension de membrane amplifiée et l'adapte en une tension comprise entre 0 V et 3.3 V pour une acquisition numérique réalisée par un autre sous-bloc de la carte MIRA (le bloc ACQ) qui est piloté par la ZCU102.

Les entrées/sorties de ce bloc POST sont données TABLE 3.12. Les signaux numériques de contrôle de SPK sont issus d'émetteurs numériques situés proche des entrées physiques de la carte MIRA, de la même manière que les blocs SPK dédiés aux blocs présynaptiques. Pour les émetteurs transmettant les signaux de contrôle des sous-blocs SPK postsynaptique, les signaux DATA et SYNC sont uniques par bloc postsynaptiques. Pour les mêmes raisons que les blocs SPK présynaptiques, les entrées des émetteurs des signaux SCLK (en entrée de la carte MIRA) sont communes par groupe de 5 sous-blocs SPK postsynaptiques résultant en seulement 2 signaux SCLK pour l'ensemble des 10 neurones postsynaptiques de la carte.

La tension $V_{\text{ref post}}$ est commune aux blocs SPK postsynaptiques et est générée par une sortie du bloc GEN de génération des tensions de contrôle (décrit par la suite) afin de permettre de changer la tension de pleine échelle des formes d'onde générées.

Les tensions V_{log} et V_{lin} sont uniques par bloc POST et sont acquises numériquement par le bloc ACQ d'acquisition de tensions (décrit par la suite).

Les tensions V_{th} et $V_{\text{tune meas}}$ sont uniques par bloc POST et sont générées par le bloc GEN de génération des tensions de contrôle.

Les quatre interrupteurs (SLOG, SLIF, SRM et SLIN) des blocs POST sont pilotés numériquement. Afin de réduire le nombre de signaux numériques [13], la commande des interrupteurs SLIF est commune pour tous les blocs POST, de même que la commande des interrupteurs SLIN, résultant en 22 signaux de commande pour tous les interrupteurs de la carte MIRA.

Les 10 signaux event qui représentent les événements en sortie des neurones LIF sont compris entre 0 V et 3.3 V grâce à un circuit d'adaptation présent dans le sous-bloc LIF et sont transmis aux FMC de connexions de la carte MIRA.

Les 10 blocs POST de la carte MIRA présentent donc 54 signaux de commande en entrée de la carte qui doivent être gérés par la partie FPGA de la carte ZCU102.

13. Si ces interrupteurs étaient pilotés individuellement, 40 signaux de commande seraient alors nécessaires.

TABLE 3.2 – Entrées/sorties du bloc POST.

Signal	Description	Type
SCLK	Signal d'horloge du sous-bloc SPK pour générer V_{post}	Numérique
SYNC	Signal de synchronisation du sous-bloc SPK pour générer V_{post}	Numérique
DATA	Signal des données du sous-bloc SPK pour générer V_{post}	Numérique
event	Signal de sortie d'un événement postsynaptique	Numérique
SLIF	Connecte le LIF à la sortie du CCII+	Numérique
SLOG	Connecte MSU LOG à la sortie du CCII+	Numérique
SRM	Connecte une résistance (faible) de décharge de la membrane	Numérique
SLIN	Connecte MSU LIN à la tension de membrane amplifiée	Numérique
I_Z	Connexion au nœud de courant en sortie du CCII+	Analogique
post line	Connexion des lignes postsynaptiques du crossbar de memristors	Analogique
$V_{\text{tune meas}}$	Tension de contrôle de la source de courant (VCCS1)	Analogique
V_{log}	Tension de sortie du bloc de mesure MSU LOG	Analogique
V_{lin}	Tension de sortie du bloc de mesure MSU LIN	Analogique
V_{th}	Tension de seuil de la membrane amplifiée	Analogique
$V_{\text{ref post}}$	Tension de référence du sous-bloc SPK pour générer V_{post}	Analogique

Une partie des différents circuits électroniques constituant ce bloc POST est donnée ANNEXE C à l'exception du sous-bloc SPK déjà donné en ANNEXE B.

3.2.3 Bloc GEN de génération de tensions $V_{\text{ref pre}}$, $V_{\text{ref post}}$, V_{th} et $V_{\text{tune meas}}$

Deux DAC AD8802 à interface série de 12 canaux chacun sont utilisés pour générer les 22 tensions analogiques nécessaires aux blocs SPK présynaptiques et aux blocs POST. Le schéma électrique simplifié du bloc GEN est illustré sur la FIGURE 3.7 et résulte en 10 entrées numériques supplémentaires de la carte MIRA.

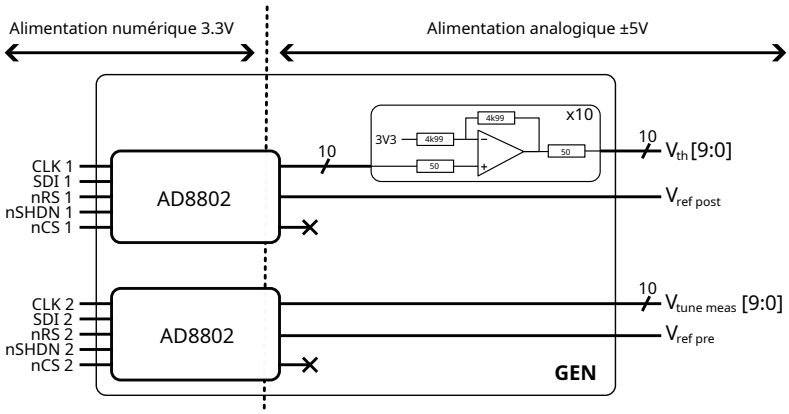


FIGURE 3.7 – Schéma électrique simplifié du bloc de génération de tensions.
Ce bloc est constitué de 2 sous-bloc VTH LIN et VTH LOG (1 par DAC) pour générer les 22 signaux analogiques nécessaires (1 $V_{\text{ref pre}}$, 1 $V_{\text{ref post}}$, 10 V_{th} et 10 $V_{\text{tune meas}}$). Les tensions $V_{\text{ref pre}}$ et $V_{\text{ref post}}$ sont comprises entre 0 V et 3.3 V permettant ainsi de générer des tensions présynaptiques et postsynaptiques de ±3.3V. On peut ainsi atteindre aux bornes des memristors des tensions maximales de ±6.6V. Les tensions de seuils des neurones LIF sont de ±3.3V. La tension de commande des sources de courant de VCCS1 sont comprises entre 0 V et 3.3 V. Pour chaque AD8802, 1 canal de tension reste inutilisé sur les 12 disponibles.

TABLE 3.3 – Entrées/sorties du bloc GEN.

Signal	Description	Type	Plage de tension
CLK 1,2	Signal d'horloge du DAC	Numérique	0 V/3.3 V
nCS 1,2	Signal de synchronisation du DAC	Numérique	0 V/3.3 V
SDI 1,2	Signal des données du DAC	Numérique	0 V/3.3 V
nRS 1,2	Signal de remise à zero du DAC	Numérique	0 V/3.3 V
mSHDN 1,2	Signal de *shutdown* du DAC	Numérique	0 V/3.3 V
$V_{\text{ref pre}}$	Tension de référence des blocs SPK	Analogique	0 V/3.3 V
$V_{\text{ref pre}}$	Tension de référence des blocs POST	Analogique	0 V/3.3 V
$V_{\text{th}}[9:0]$	Tension de seuil des neurones LIF	Analogique	±3.3V
$V_{\text{tune meas}}[9:0]$	Tension de contrôle de VCCS1	Analogique	0 V/3.3 V

3.2.4 Bloc ACQ d'acquisition de tensions (V_{log} et V_{lin})

Deux ADC [14] TLV2553 12 bits à interface série de 11 canaux sont utilisés pour collecter les 20 tensions analogiques issues des blocs POST pour mesure. Le schéma électrique simplifié est illustré sur la FIGURE 3.8 et résulte en 10 entrées/sorties numériques supplémentaires de la carte MIRA.

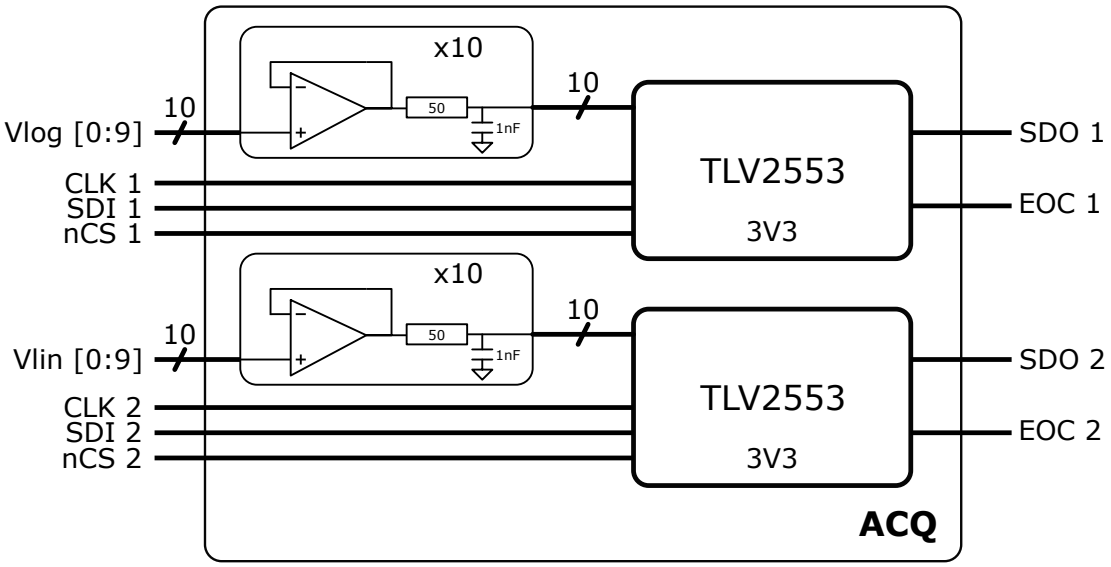


FIGURE 3.8 – Schéma électrique simplifié du bloc d'acquisition de tensions.
Ce bloc est constitué de 2 ADC pour mesurer les 20 signaux analogiques d'intérêt (10 V_{lin} et 10 V_{log}).

TABLE 3.4 – Entrées/sorties du bloc ACQ.

Signal	Description	Type	Plage de tension
CLK 1,2	Signal d'horloge de l'ADC	Numérique	0 V/3.3 V
nCS 1,2	Signal de synchronisation de l'ADC	Numérique	0 V/3.3 V
SDI 1,2	Données de contrôle de l'ADC	Numérique	0 V/3.3 V
SDO 1,2	Données de mesure de l'ADC	Numérique	0 V/3.3 V
EOC 1,2	Indication de fin de conversion de l'ADC	Numérique	0 V/3.3 V
V_{log}	Tensions de mesure des blocs MSU LOG	Analogique	0 V/3.3 V
V_{lin}	Tensions de mesure des blocs MSU LIN	Analogique	0 V/3.3 V

14. De l'anglais *Analog-to-Digital Converter* ou convertisseur analogique numérique.

3.2.5 Agencement des blocs sur la carte MIRA

La carte MIRA est réalisée en électronique discrète avec un circuit imprimé à 6 couches. Les nombreux circuits SPK nécessitent des signaux numériques de 25 MHz [15] (pour les signaux d'horloge SCLK des 91 SPK). Afin de s'assurer que ces signaux numériques soient correctement reçus et qu'ils n'impactent pas la partie analogique, un agencement spécifique de ces blocs a été défini :

— Mise en quinconce des blocs SPK pour économiser de la surface.

— Séparation des plans de masse numérique et analogique pour éviter la perturbation des signaux générés par certains bloc SPK physiquement proches des DAC (et donc de signaux numériques) de blocs SPK voisins à cause du positionnement en quinconce de ces derniers.

Selon cet agencement, la partie analogique et la partie numérique sont séparé en deux zones correspondant à la séparation des plans de masse analogique et numérique de la carte MIRA. Il existe sur la carte des points de connexion entre les deux plans au travers de résistance de 0 Ω et de diode en tête-bêche le long de la séparation.

De plus, ces 91 blocs SPK (neurones présynaptiques et postsynaptiques) agencés en quinconce ont été placés au plus proche des émetteurs en entrée de la carte pour respecter une distance maximale de routage. Cette distance est obtenue par une « règle de table » contraignante pour s'assurer de ne pas avoir à considérer de propagation des premières harmoniques des signaux numériques de commande, ce qui entraînerait le cas échéant des distorsions de ces derniers. La « règle de table » utilisée fonctionne de la manière suivante :

1. Détermination de la 3e harmonique non nulle du signal numérique de commande. Ici, il s'agit de f_5 =125 MHz pour un signal d'horloge (carré) de f_0 =25 MHz.
2. Estimation de la vitesse de propagation dans la piste. Environ [16] $v = \frac{2}{3}c$ (avec c la célérité) pour la propagation d'un signal électrique dans la piste d'une carte électronique.
3. Calcul de la longueur d'onde correspondante. Ici, $\lambda = \frac{v}{f_5}$ =160 cm.
4. Routage avec des pistes dont la longueur maximale est d'environ $\frac{\lambda}{10}$ =16 cm.

La carte MIRA respecte majoritairement [17] cette règle pour le placement des blocs SPK et des émetteurs en entrée.

Ce sont ces deux contraintes, séparation des parties numérique et analogique (pour éviter une perturbation des signaux analogiques par les signaux numériques) et contrainte de longueur

15. Même si le fonctionnement théorique à 50 MHz a été vérifié, en pratique la carte MIRA fonctionne avec des signaux jusqu'à 25 MHz.
16. On cherche à **estimer** des longueurs maximales de routage en utilisant la 3e harmonique, la valeur précise de la vitesse de propagation n'est pas primordiale.
17. Certaines longueurs de pistes avoisinent les 20 cm.

des pistes de certains signaux numériques (signaux de contrôle des DAC des blocs SPK) qui ont résultées en l'agencement choisi des différents blocs de la carte MIRA.

3.2.6 Synthèse des entrées/sorties de la carte MIRA

La carte MIRA possède donc 256 signaux numériques qui sont récapitulés TABLE 3.5. Ces 256 signaux sont répartis sur 3 connecteurs FMC alignés sur le long du bord sud de la carte MIRA.

TABLE 3.5 – Entrées/sorties numériques de la carte MIRA.

Bloc	Signal	Nombre
x	en post*	1
x	en pre**	1
SPK (pré)	SCLK	81
SPK (pré)	DATA	81
SPK (pré)	SYNC	18
POST	SCLK	2
POST	DATA	10
POST	SYNC	10
POST	SLOG	10
POST	SLIN	1
POST	SLIF	1
POST	SRM	10
POST	event	10
ACQ	CLK	2
ACQ	DATA IN	2
ACQ	DATA OUT	2
ACQ	nCS	2
ACQ	EOC	2
GEN	CLK	2
GEN	nRS	2
GEN	nSHDN	2
GEN	SDIN	2
GEN	nRS	2
MIRA	Tous signaux	**256**

* Signal complémentaire d'activation d'un circuit appliquant aux blocs SPK présynaptiques le signal $V_{\text{ref pre}}$ généré par GEN.

** Signal complémentaire d'activation d'un circuit appliquant aux blocs POST le signal $V_{\text{ref post}}$ généré par GEN.

3.2.7 VCCS : carte optionnelle d'ajustement des courants de sortie du CCII+

Lors des premières expérimentations avec la plateforme 81 × 10, certains des neurones postsynaptiques ou blocs POST ont présenté des *offsets* de courant en sortie des CCII+ plus importants que prévu et qui étaient difficilement corrigeables avec l'élément VCCS1 du sous-bloc MSU LOG lors de la mesure ou le changement de V_{th} (pour compenser l'offset de la tension de membrane issue de cet offset en courant). Afin de pallier ce problème, la carte complémentaire VCCS2 illustrée sur la FIGURE 3.9 a été mise au point. Cette carte contient 10 sources de courant dont le circuit électronique est donné en ANNEXE D et correspond au même montage électronique que VCCS1, mais avec des valeurs de composants et de références différentes pour changer la plage de courant qu'elle peut générer. Ces 10 sources de courants pilotées en tension se connectent à l'entrée/sortie I_Z des blocs POST [18]. La commande de ces sources de courant peut s'effectuer de 2 manières différentes sur cette carte :

1. L'ajustement manuel pour chaque source de courant de la carte d'une résistance variable pilotant la tension de contrôle.
2. Le pilotage d'un DAC AD8802 [19] par la carte ZCU102 par l'intermédiaire d'un connecteur PMOD pour générer les 10 tensions de contrôle des sources de courant de la carte VCCS2.

Si seulement la première méthode a été utilisée dans une partie des expérimentations qui sont présentées par la suite (ajustement manuel avant chaque expérimentation), c'est la deuxième méthode qui est envisagée pour le long terme [20] puisque, grâce au circuit d'acquisition de la carte pour le courant ou pour la tension de membrane et le contrôle numérique de cette carte, il est possible de mettre en place un système d'ajustement automatique de ces courants de correction [21] générés par cette carte.

Il faut noter que les sources de courant de cette carte VCCS2 présentent une impédance de sortie de 10 kΩ, ce qui résulte en une résistance de la membrane de 5 kΩ si elle est connectée lors de l'utilisation de la plateforme.

18. Nœud en sortie Z du convoyeur de courant dont la tension est pilotée par ce dernier. Peut être utilisé pour compenser différents *offsets* ou récolter le courant pour de la mesure avec les appareils de « paillasse ».
19. Même référence que les DAC utilisés dans le bloc GEN.
20. Après ces travaux de thèse.
21. Ces courants en sortie du CCII+ dépendent de la polarisation du crossbar de memristors et des circuits qui sont connectés en sortie du convoyeur de courant. Cela signifie qu'il faut adapter la correction de manière différente lorsque l'on fait de la mesure en courant ou lorsque l'on utilise le neurone LIF.

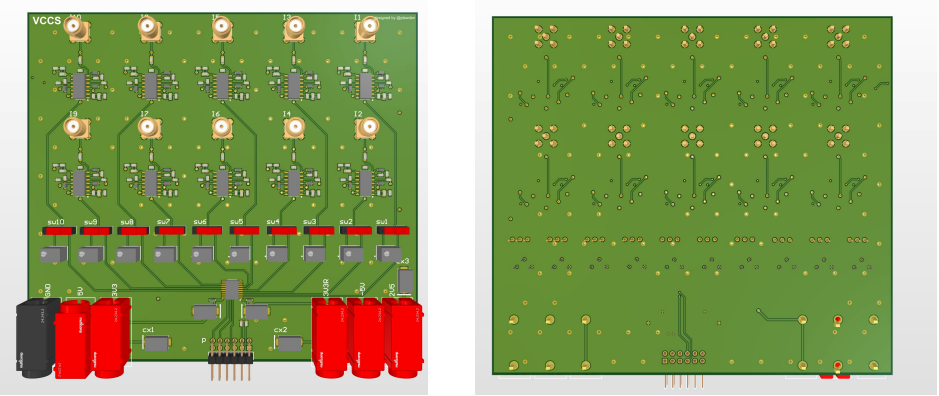


FIGURE 3.9 – Vues de dessus et de dessous de la carte VCCS2.
Vues obtenues sur le logiciel de conception Altium Designer. Il s'agit d'une carte réalisée en 4 couches. On y distingue les 10 sorties (connecteurs SMA) des 10 sources de courant pilotées en tension. Les interrupteurs mécaniques sw[1 : 10] permettent de choisir si le pilotage de la source de courant est réalisé par des *trimmers* ou par le DAC AD8803. Le PMOD au sud de la carte correspond au connecteur donnant accès aux entrées de commande du DAC.
Cette carte fait 13.6 cm×12.2 cm.

3.3 Carte ADAPT : Partie communication numérique

La partie de la plateforme 81 × 10 qui implémente le réseau de neurones analogiques qui se trouve sur la carte MIRA nécessite la gestion de 256 entrées/sorties numériques en logique 3.3 V. La carte ZCU102, qui a été choisie pour implémenter toute la partie DCB étendu (contrôle et supervision des expérimentations), présente la majorité de ses connexions programmables (partie FPGA du Zynq) d'entrées/sorties numériques disponibles sur deux FMC[22]. Ces deux connecteurs mettent à disposition un total de 128 signaux programmables *single-ended*[23] qui ne suffisent donc pas pour piloter directement les 256 signaux de la carte MIRA. De plus la tension logique de fonctionnement des signaux numériques sur ces deux connecteurs FMC de la carte ZCU102 est au maximum de 1.8 V.

Le rôle de la carte ADAPT illustrée sur la FIGURE 3.10 est donc de permettre l'adaptation des niveaux de tension entre les deux cartes (pour l'émission ou la réception de signaux depuis la carte ZCU102) et le dé-multiplexage de certains signaux issus de la carte ZCU102 pour pouvoir gérer les 256 entrées/sorties de la carte MIRA.

La carte ADAPT nécessite 3 alimentations externes :

— Une alimentation correspondant au niveau de tension logique de la carte ZCU102. (1.8 V)

— Une alimentation pour la partie dé-multiplexage temporel fixée à 3.3 V mais qui peut être modifiée pour augmenter les performances en fréquence des composants si nécessaire.

— Une alimentation correspondant au niveau de tension logique de la carte MIRA (3.3 V).

Les différents canaux de communications qui constituent cette carte ADAPT et qui permettent le lien entre les cartes MIRA et ZCU102 sont listés TABLE 3.6.

22. FMC de format différent que ceux choisis pour la carte MIRA.
23. La ZCU102 permet de choisir entre 128 signaux *single-ended* ou 64 signaux différentiels.

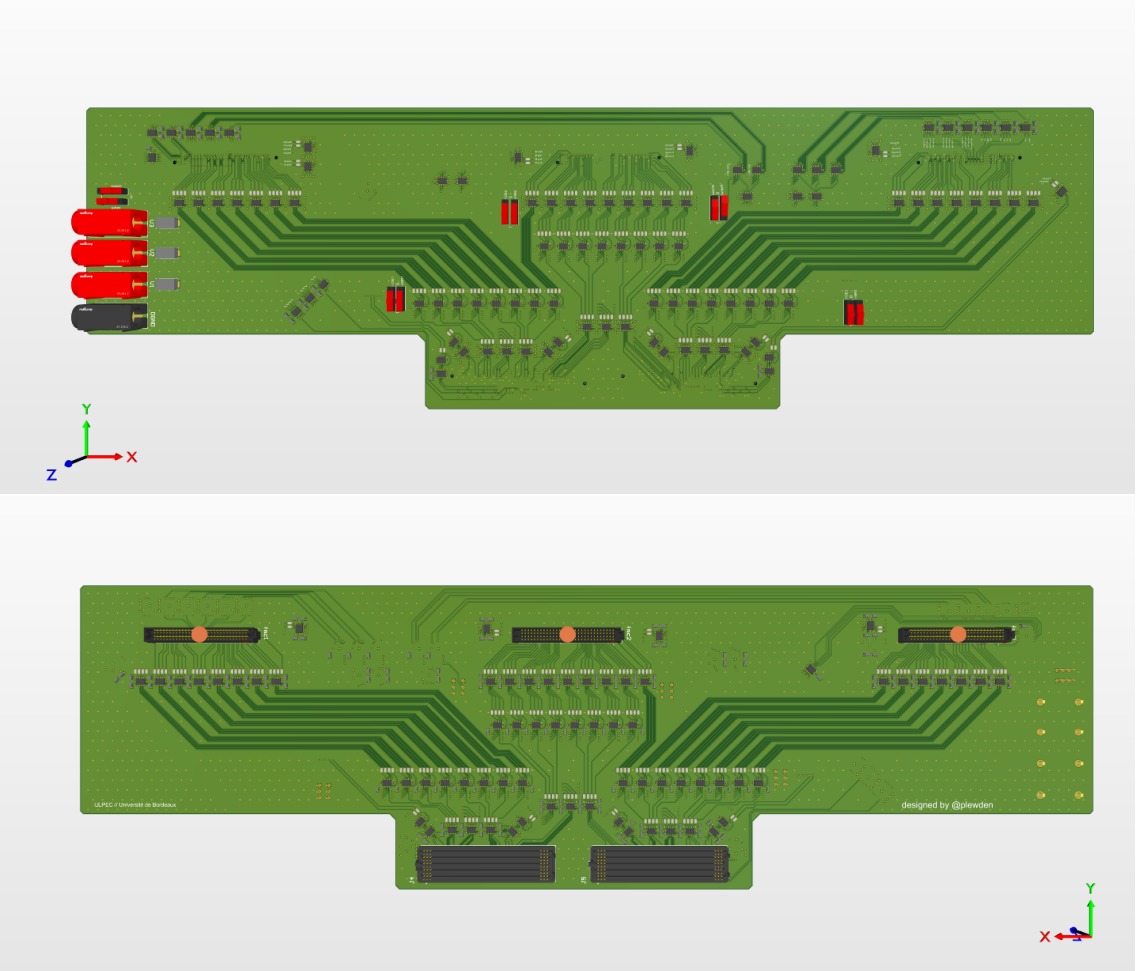


FIGURE 3.10 – Vues de dessus et de dessous de la carte ADAPT.
Vues obtenues sur le logiciel de conception Altium Designer. Il s'agit d'une carte réalisée en 6 couches. Les 2 connecteurs FMC mâles au sud de la carte correspondent aux connexions avec la carte ZCU102 et les 3 connecteurs FMC femelles au nord de la carte correspondent aux connexions avec la carte MIRA.

TABLE 3.6 – Canaux de communication de la carte ADAPT.

Canal	Blocs (MIRA)	Utilisation	Principe de communication	Direction
SPK controls 1	SPK et POST	Transmettre SYNC et DATA	Registre à décalage *Level shifters**	ZCU102→MIRA
SPK controls 2	SPK et POST	Transmettre SCLK	Registre à décalage *Level shifters**	ZCU102→MIRA
ACQ controls	ACQ	Transmettre et recevoir les signaux de ACQ	*Level shifters**	ZCU102↔MIRA
GEN controls	GEN	Transmettre les signaux de GEN	Registre à décalage *Level shifters**	ZCU102→MIRA
POST config	POST	Transmettre les signaux des interrupteurs analogiques	Registre à décalage *Level shifters**	ZCU102→MIRA
Post Event	POST	Transmettre les événements postsynaptiques	*Level shifters**	ZCU102←MIRA

* Composant 74AVC4TD245 assurant la conversion des niveaux de tension logique d'une tension A vers une tension B.

Les canaux qui font intervenir des registres à décalage utilisent du multiplexage temporel pour pouvoir envoyer à la carte MIRA depuis la ZCU102 tous les signaux numériques nécessaires. Seuls les canaux Post Event et ACQ controls transmettent les signaux numériques uniquement au travers de *level shifters*. Ces canaux dont les signaux ne sont pas multiplexés temporellement sont ceux qui transmettent des informations de la carte MIRA vers la carte ZCU102. En effet, seuls les canaux communiquant uniquement de la ZCU102 vers MIRA présentent du multiplexage temporel.

Les registres à décalage 74LVC595A choisis présentent une particularité qui permet de générer les signaux de commande pour MIRA de manière synchronisés. En effet, ils fonctionnent avec deux horloges distinctes :

— La première (SHCP) permet d'enregistrer à chaque front montant une valeur dans le registre à décalage.

— La deuxième (STCP) permet d'appliquer simultanément en sortie à chaque front montant les données contenues dans le registre.

On peut ainsi remplir un registre des signaux que l'on souhaite appliquer avec SHCP, et les générer simultanément en sortie pour la carte MIRA avec STCP. Ces registres à décalage prennent donc en entrée des signaux multiplexés contenant plusieurs signaux numériques de commande pour la carte MIRA. Ce multiplexage des signaux numériques est assuré par le DCB étendu de la plateforme qui est implémenté sur la partie FPGA du Zynq de la ZCU102.

3.3.1 SPK controls 1

La carte MIRA nécessite en entrée 91 signaux DATA et 91 signaux SYNC pour les 81 SPK pré-synaptiques et les 10 blocs POST, soit un total de 182 signaux DATA et SYNC. Pour réduire ce nombre de 182 à 72 signaux de commande issus de la ZCU102, le canal de communication illustré sur la FIGURE 3.11 est dupliqué 6 fois [24].

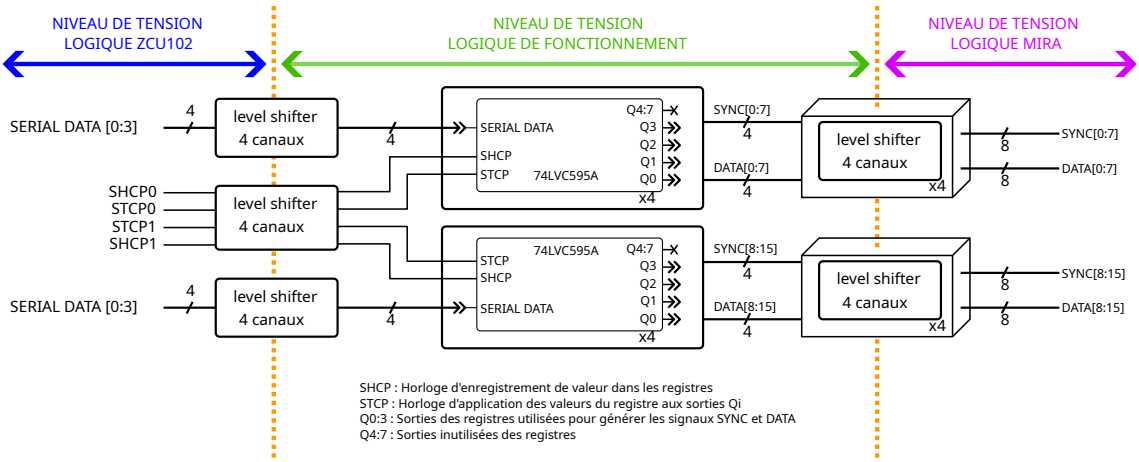


FIGURE 3.11 – Schéma du canal SPK controls 1 de la carte ADAPT.
Permet la génération de 16 signaux SYNC et 16 signaux DATA pour 12 signaux en entrée.

Les signaux SERIAL DATA contiennent 4 signaux (DATA ou SYNC) multiplexés temporellement et sont dé-multiplexés par les registres à décalage 8 bits 74LVC595A où seulement 4 sorties sont utilisées [25]. Pour pouvoir utiliser plus de sorties de ces registres 8 bits les signaux SERIAL DATA, STCP et SHCP devraient présenter des fréquences plus élevées.

24. Sur la sixième duplication certaines des sorties ne sont pas utilisées car elles dépassent les 182 signaux nécessaires.

25. L'utilisation des registres à décalage pour ces canaux de communication est limitée à 4 sorties à cause de leur fréquence d'utilisation maximale.

3.3.2 SPK controls 2

La carte MIRA nécessite en entrée 20 signaux SCLK pour les 81 blocs SPK présynaptiques et les 10 blocs POST [26]. La synchronicité des signaux de données (DATA, SYNC) et d'horloges (SCLK) des DAC de la carte MIRA est critique pour leur bon fonctionnement. Ainsi, afin de leur faire subir la même transformation et délai de transmission, le canal de communication SPK control 2 illustré sur la FIGURE 3.12 est utilisé et fonctionne avec les mêmes références de composants électroniques. Ce canal de transmission, contrairement aux autres qui visent à réduire le nombre de signaux de commande issues de la ZCU102, fait passer de 20 signaux nécessaires à 22. Cette augmentation de 2 signaux est requise afin de s'assurer que la transmission des signaux SCLK suive des délais de transmission à travers les circuits électroniques utilisés similaires à ceux des signaux DATA et SYNC qui passent au travers du canal de transmission SPK controls 1.

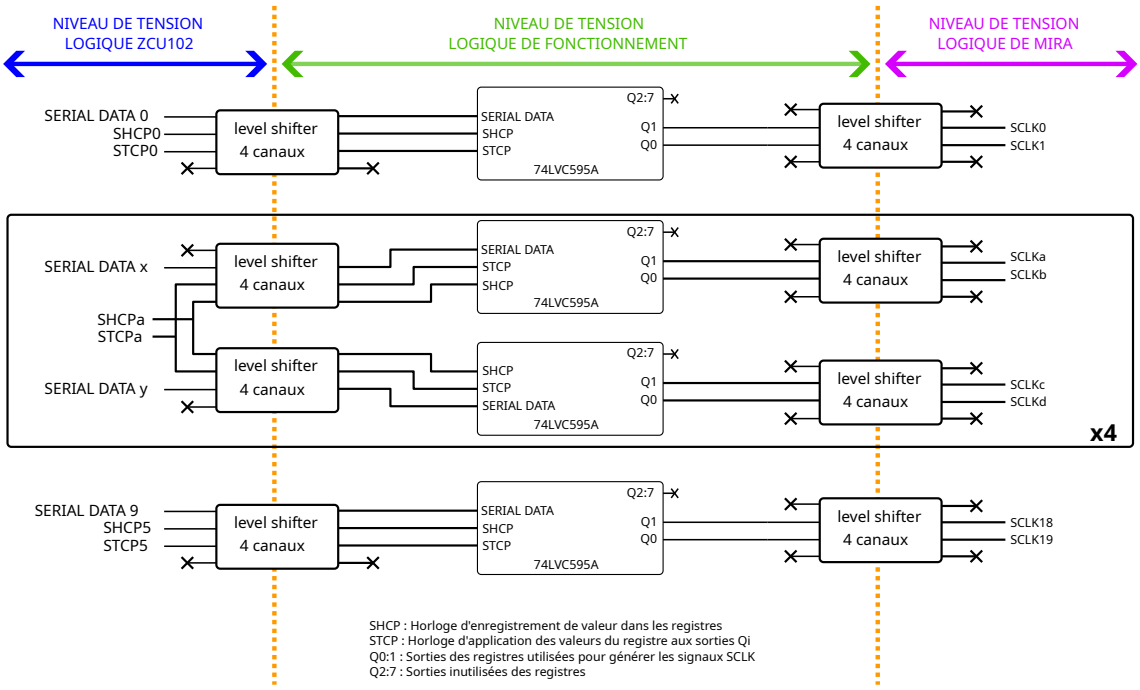


FIGURE 3.12 – Schéma du canal SPK controls 2 de la carte ADAPT.
Permet la génération de 20 signaux SCLK (18 pour les présynaptiques, 2 pour les postsynaptiques) à partir de 22 signaux de commande.

Les signaux SERIAL DATA contiennent, contrairement aux canaux SPK controls 1, seulement deux signaux SCLK multiplexés temporellement et n'utilisent que deux sorties des registres à décalage. En effet, les signaux SCLK fonctionnant à des fréquences deux fois plus élevées que les signaux de données DATA et SYNC, le nombre de sorties utilisables des registres à décalage est divisé par 2.

26. Les signaux SCLK sont partagés par groupe de 4 ou 5 blocs SPK sur la carte MIRA.

3.3.3 GEN controls

Pour piloter le bloc GEN, 10 signaux numériques sont nécessaires. Le canal de transmission GEN controls illustré sur la FIGURE 3.13 permet de réduire ce nombre à 4. Le bloc GEN fonctionnant à une fréquence moins élevée que les blocs SPK, plus de sorties des registres peuvent être utilisées.

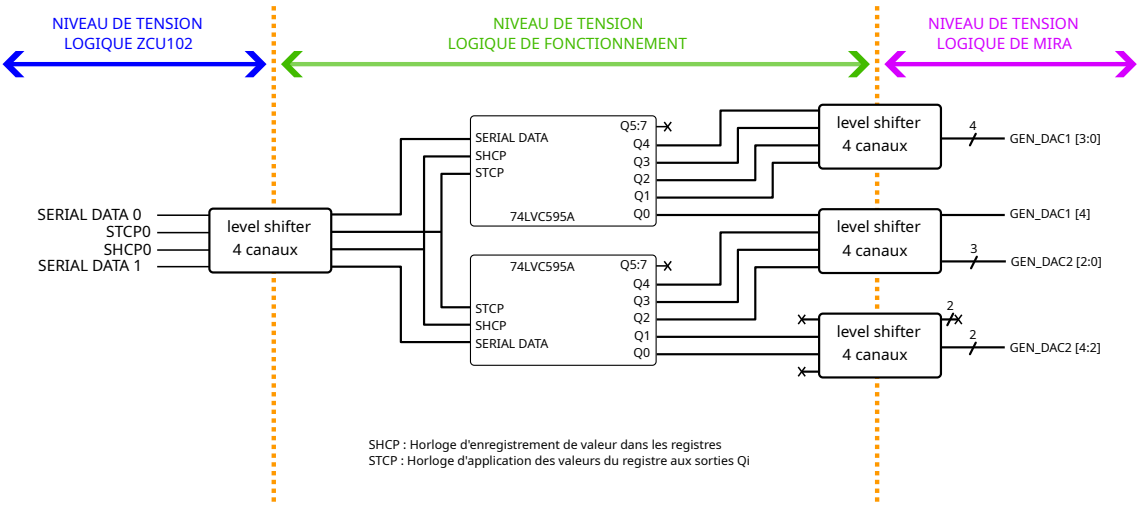


FIGURE 3.13 – Schéma du canal GEN controls de la carte ADAPT.
Permet la génération des 10 signaux de commande des DAC 1 et 2 du bloc GEN de la carte MIRA à partir de 4 signaux de la carte ZCU102.

3.3.4 POST config

Pour piloter les interrupteurs SLIF, SLOG, SRM et SLIN des blocs POST, 22 signaux sont nécessaires. Le canal de transmission POST config illustré sur la FIGURE 3.14 permet de réduire ce nombre à 5 en utilisant 3 registres à décalage.

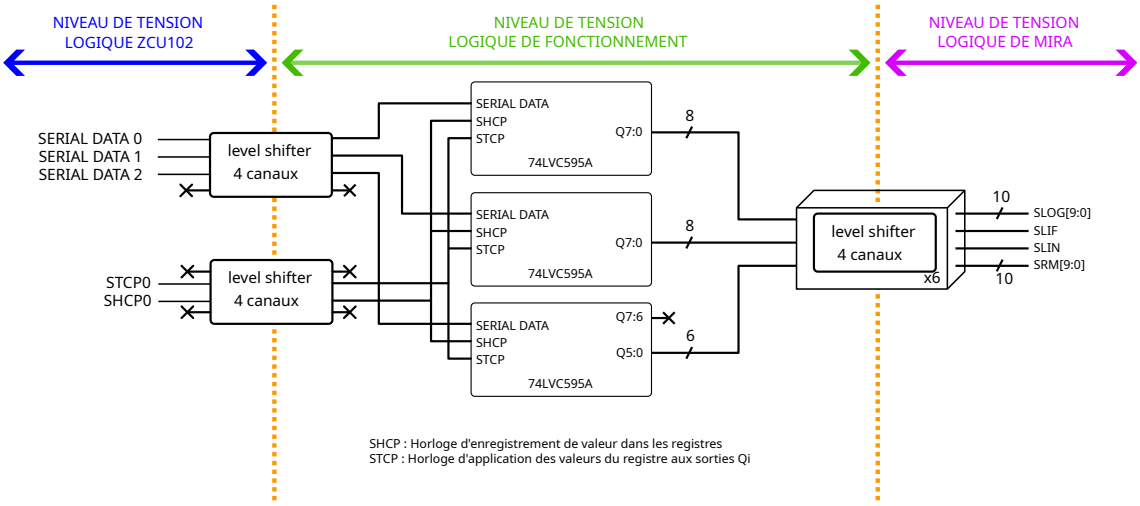


FIGURE 3.14 – Schéma du canal POST config de la carte ADAPT.
Permet la génération des 22 signaux de commande des interrupteurs analogiques des blocs POST de la carte MIRA à partir de 5 signaux de la carte ZCU102.

3.3.5 Post event, ACQ controls, « en post » et « en pre »

Les signaux « en pre » et « en post » sont gérés par 2 interrupteurs mécaniques sur la carte ADAPT et ne nécessitent donc pas de signaux issus de la ZCU102.

Les 20 signaux pour les canaux Post event (10 signaux event) et ACQ controls (10 signaux pour le bloc ACQ), n'utilisent que des *levels shifters* pour adapter les niveaux de tensions et nécessitent donc 20 connexions de la ZCU102.

3.3.6 Synthèse

Les canaux de transmission implémentés sur la carte ADAPT permettent de passer de 256 signaux de contrôle numériques à 123 tout en adaptant les niveaux de tension logique entre le 1.8 V de la carte ZCU102 et le 3.3 V de la carte MIRA. Il est donc possible de piloter la carte MIRA avec les signaux disponibles sur les deux connecteurs FMC de la carte ZCU102 (128 signaux configurable disponibles) au travers de la carte ADAPT. Le nombre de connexions de chacun de ces canaux de transmission constituant la carte ADAPT est donné TABLE 3.7.

TABLE 3.7 – Nombres de connexions de la carte ADAPT.

Canal	Connexions avec la ZCU102	Connexions avec MIRA	Direction
SPK controls 1	72	182	ZCU102→MIRA
SPK controls 2	22	20	ZCU102→MIRA
GEN controls	4	10	ZCU102→MIRA
POST config	5	22	ZCU102→MIRA
ACQ controls	10	10	ZCU102↔MIRA
Post event	10	10	ZCU102←MIRA
« en pre » et « en post »	0	2	ADAPT→MIRA
TOTAL	**123**	**256**	**ZCU102↔MIRA**

3.4 Carte ZCU102

Le pilotage de la plateforme 81 × 10 est assuré par la carte ZCU102. Cette carte comporte un Zynq de la famille Xilinx™ qui co-intègre une partie processeur ARM (PS ou CPU) et de la logique programmable (FPGA) dans une même puce, simplifiant ainsi la communication entre le CPU et le FPGA. L'utilisation du Zynq pour cette plateforme a été déterminée par la volonté d'utiliser un FPGA pour implémenter le DCB étendu sur de la logique programmable (similaire au démonstrateur ULPEC 784 × 100) et l'utilisation du CPU pour écrire et exécuter des algorithmes en C haut niveau pour les différentes expérimentations que l'on souhaite réaliser avec cette plateforme.

L'architecture de contrôle se répartit donc sur la partie PS (processeurs, code en C) et la partie PL (Logique Programmable/FPGA, VHDL). Une vue générale de cette architecture de contrôle est illustrée sur la FIGURE 3.15. On y distingue notamment 4 zones :

1. Zone ARM++ représentant la plateforme hors logique programmable et sa configuration (les processeurs ARM, périphériques physiques, ...) et représente donc la zone où les algorithmes d'expérimentations s'exécuteront [27]. Cette zone correspond à la partie PS du Zynq.
2. Zone A du DCB étendu dédiée aux interfaces AXI entre la PL et la PS. Il s'agit des modules VHDL qui assurent la communication entre la partie PS et PL par l'intermédiaire de registres en mémoire de 32 bits suivant le protocole AXI4-Lite [28]. Les valeurs contenues dans ces registres 32 bits sont utilisées pour échanger les informations entre la PL et la PS. Cette zone se trouve dans la partie PL du Zynq.
3. Zone B du DCB étendu. Cette zone contient toutes les machines d'états et modules VHDL assurant le fonctionnement de la plateforme et la génération des signaux nécessaires au pilotage de la carte MIRA. C'est cette zone qui implémente notamment les règles d'apprentissage qui ont pu être définies dans le CHAPITRE 2. Cette zone se trouve dans la partie PL du Zynq.
4. Zone C du DCB étendu. Cette zone est dédiée au pilotage de la carte ADAPT. Elle contient les machines d'états et modules VHDL qui multiplexent temporellement les signaux avant de les envoyer aux canaux de transmission utilisant des registres à décalage. Si la zone B du DCB étendu génère les signaux naturellement compatibles avec MIRA, la zone C les modifie pour pouvoir les transmettre par l'intermédiaire de la carte ADAPT. Cette zone se trouve dans la partie PL du Zynq.

La raison pour laquelle le DCB est qualifié d'étendu pour cette plateforme est parce qu'il per-

27. Le code s'exécute au plus proche du matériel (bas niveau) sans passer par l'intermédiaire d'un système d'exploitation. La communication avec la partie PL se fait par lecture et écriture à des adresses mémoires spécifique.
28. Il existe 3 protocoles de communication pour l'AXI4 : AXI4-Stream, l'AXI4-full et l'AXI4-Lite. C'est ce dernier qui a été choisi et qui permet de communiquer via des adresses mémoires 32 bits.

met de réaliser plus que le pilotage du réseau de neurones de la carte MIRA (zone B du DCB) comme cela avait pu être décrit lors de la définition du DCB pour les simulations et le démonstrateur ULPEC.

Pour s'assurer de la synchronicité des signaux envoyés à la carte ADAPT (en sortie de la zone C du DCB étendu), un fichier de contraintes est défini pour que l'implémentation se réalise en respectant des contraintes de temps sur les signaux émis.

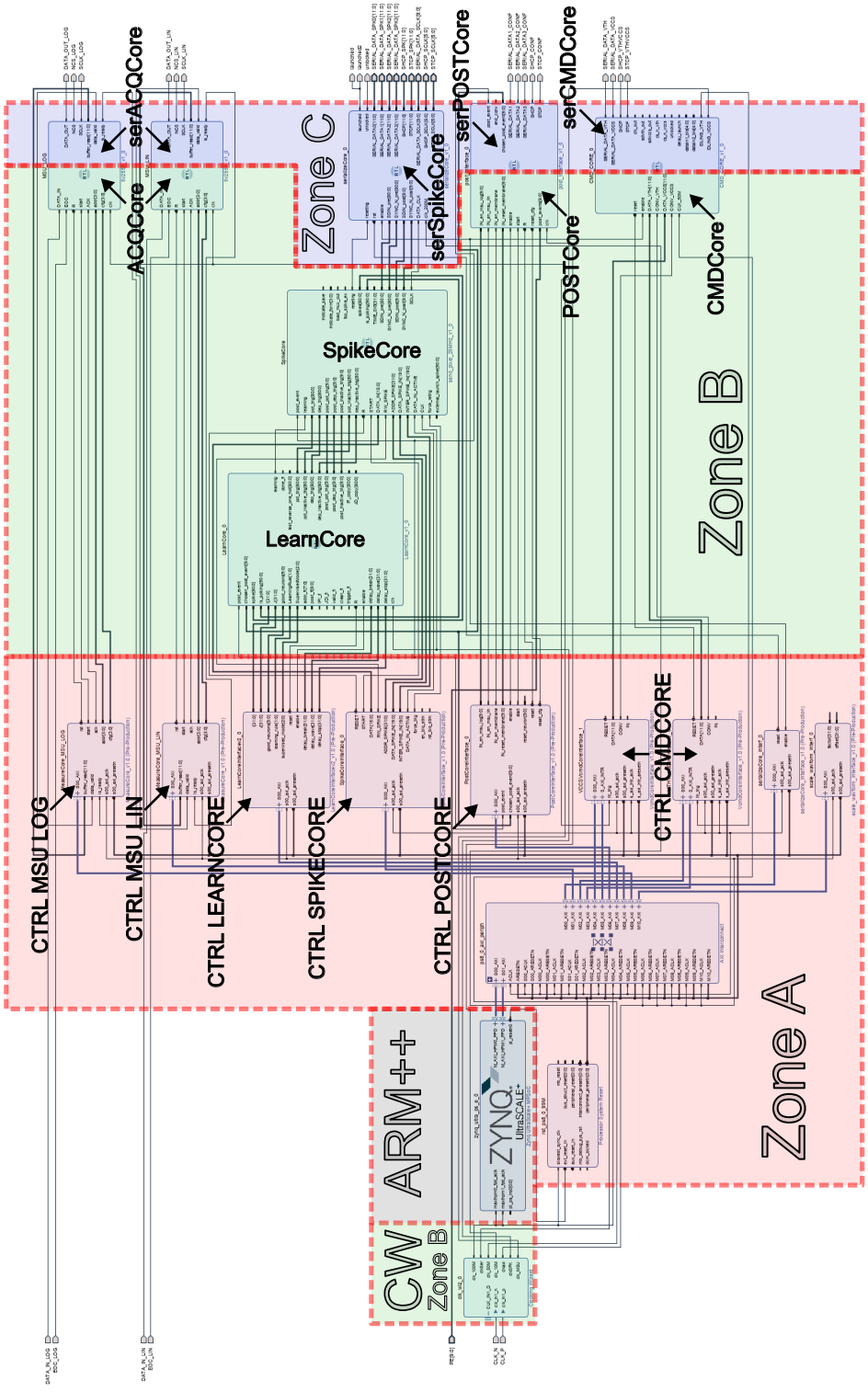


FIGURE 3.15 – Vue générale de cette architecture de contrôle de la plateforme 81 × 10 sur le logiciel Vivado.

Le bloc CW (appartenant à la zone B) correspond au module clock wizard de la puce (boucles à verrouillage de phase configurables) permettant de générer différents signaux d'horloge de fonctionnement des modules.

3.4.1 Zone B du DCB étendu

La zone A du DCB est dédiée à la lecture/écriture de valeurs 32 bits en mémoire pour communiquer des informations et signaux de commande entre la partie PL et PS. La zone C, elle, est dédiée à la génération et sérialisation (multiplexage temporel) des différents signaux (SERIAL DATA, STCP, SHCP) de pilotage avant de les transmettre à la carte ADAPT qui les dé-sérialise pour les transmettre à la carte MIRA.

La zone B quant à elle contient le cœur du DCB puisque c'est elle qui :

— Implémente les règles d'apprentissage dans le module LearnCore.

— Gère la génération des formes d'onde à envoyer aux DAC des blocs SPK et POST dans le module SpikeCore.

— Assure la configuration des blocs POST et la récolte de leurs événements et leur arbitrage (choix du neurone postsynaptique qui a émis un événement) dans le module PostCore.

— Pilote les ADC du bloc ACQ pour l'acquisition de valeurs dans ACQCore (modules MSU LOG et MSU LIN).

— Pilote les tensions de seuils, de commandes VCCS et de référence $V_{\text{ref pre}}$ et $V_{\text{ref post}}$ dans le module CMDCore.

1. ACQCore.

Les deux modules MSU LOG et MSU LIN qui constituent ACQCore sont illustrés sur la FIGURE 3.16 et leurs entrées/sorties sont données TABLE 3.16. Ils utilisent un protocole série pour communiquer avec le DAC et transmettent à des adresses en mémoire (module CTRL MSU LIN ou LOG en zone A) les résultats obtenus ainsi que des signaux pour indiquer si le module est prêt à exécuter une mesure [29] ou si la valeur écrite en mémoire est valide [30]. Après le lancement d'une mesure, tant que la valeur lue n'a pas été acceptée par la PS (grâce au signal ACK) les modules MSU LIN et MSU LOG ne peuvent pas faire d'autre lecture (sauf remise à zéro avec R). Ces modules fonctionnent avec une horloge CLK de 10 MHz générée par le bloc CW et émettent les données à transmettre aux DAC de la carte MIRA avec un protocole série à 1 MHz. Les signaux numériques à transmettre à la carte MIRA passent par un module de sérialisation en zone C nommé serACQCore qui permet de les multiplexer temporellement et de les transmettre au canal de transmission ACQ control de la carte ADAPT tout en générant les signaux de contrôle supplémentaires nécessaires (STCP, SHCP).

29. Le module est prêt s'il est en attente du signal start (après une réinitialisation ou après une mesure précédente reçue par la PS).

30. La valeur est valide si la machine d'état a fini et la mesure en sortie (buffer read) correspond bien à la mesure.

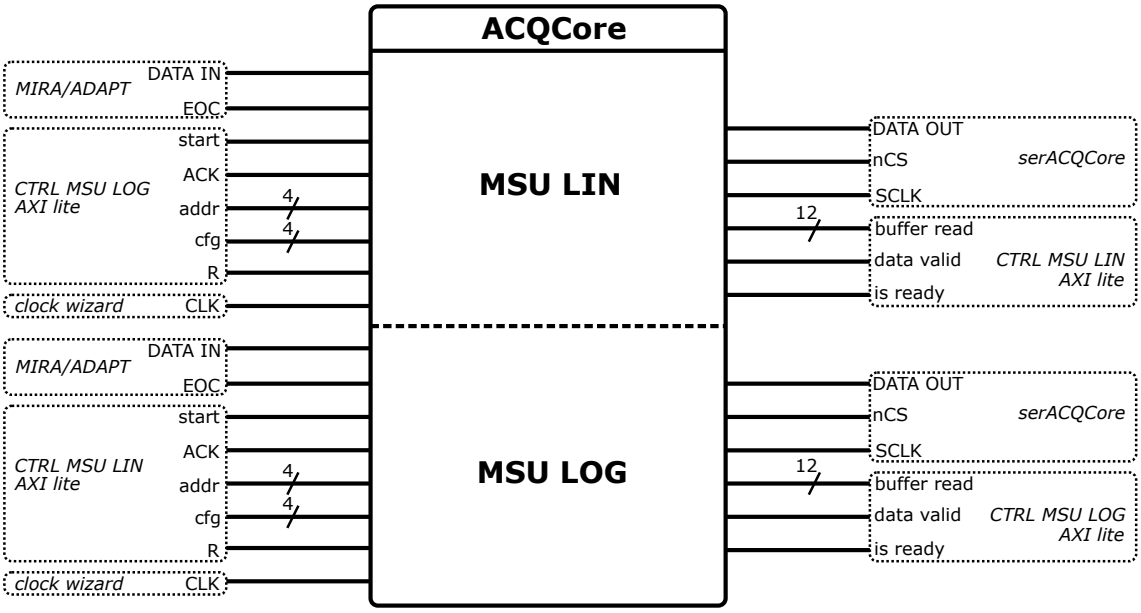


FIGURE 3.16 – Module ACQCore.

TABLE 3.8 – Entrées/sorties d'un module MSU (LIN ou LOG) de ACQcore.

Signal	Description	Entrée/Sortie	Connexion
DATA IN	Données émises par l'ADC (mesure)	E	externe
EOC	Donnée émise par l'ADC (Fin de conversion)	E	externe
start	Signal pour le lancement de la machine d'état pour la lecture	E	interne (PS)
ACK	Signal pour notifier que la PS a lu en mémoire la valeur	E	interne (PS)
addr[3 :0]	Adresse du canal de l'ADC choisi pour la lecture	E	interne (PS)
cfg[3 :0]	Configuration du mode de lecture de l'ADC	E	interne (PS)
clk	Horloge de fonctionnement du module VHDL	E	interne (PL)
R	Remise à zéro (réinitialisation) du module VHDL	E	interne (PS)
DATA OUT	Données de commande à transmettre à l'ADC	S	externe
nCS	Données de commande à transmettre à l'ADC	S	externe
SCLK	Signal d'horloge à transmettre à l'ADC	S	externe
buffer read[11 :0]	Mesure reçue de l'ADC mise en parallèle et écrite en mémoire pour la PS	S	interne (PS)
data valid	Valeur (bit) transmise à la PS avec buffer read pour indiquer que la valeur en mémoire est valide	S	interne (PS)
is ready	Valeur indiquant que le module VHDL est prêt à réaliser une mesure	S	interne (PS)

2. CMDCore.

Le module CMDCore est illustré sur la FIGURE 3.17 et ses entrées/sorties sont données TABLE 3.9. Il utilise un protocole série[31] pour communiquer avec les DAC du bloc GEN de la carte MIRA. Après le lancement de ce module, tant que la communication SPI transmettant les valeurs aux DAC n'a pas fini, ce module ne peut pas transmettre d'autres valeurs (sauf remise à zéro avec R). Ce module fonctionne avec une horloge CLK de 50 MHz et transmet les données aux DAC du bloc GEN avec un protocole série à 5 MHz. Les signaux numériques à transmettre à la carte MIRA passent par le module serCMDCore qui permet de les multiplexer temporellement et de les transmettre au canal de transmission GEN controls de la carte ADAPT tout en générant les signaux de contrôle supplémentaires nécessaires (STCP, SHCP).

31. Voir la documentation technique du composant AD8803.

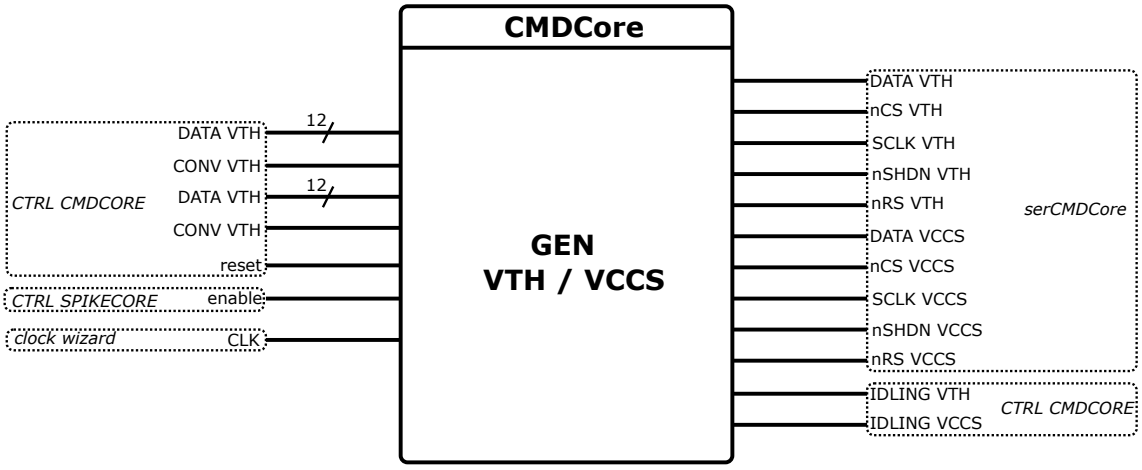


FIGURE 3.17 – Module CMDCore.

TABLE 3.9 – Entrées/sorties du module CMDCore.

Signal	Description	Entrée/Sortie	Connexion
reset	Remise à zéro (réinitialisation) du module VHDL	E	interne (PS)
enable	Activation du module VHDL	E	interne (PS)
DATA_VTH[11 :0]	Données à transmettre au DAC1*	E	interne (PS)
DATA_VCCS[11 :0]	Données à transmettre au DAC2**	E	interne (PS)
CONV_VTH	Signal pour lancer la transmission SPI au DAC1	E	interne (PS)
CONV_VCCS	Signal pour lancer la transmission SPI au DAC2	E	interne (PS)
DATA_VTH	Signal DATA du DAC1	S	vers modules de multiplexage
nCS_VTH	Signal nCS du DAC1	S	vers modules de multiplexage
nSHDN_VTH	Signal nSHDN du DAC1 (fixé à zéro)	S	vers module de multiplexage
nRS_VTH	Signal nRS du DAC1 (fixé à zéro)	S	vers module de multiplexage
DATA_VCCS	Signal DATA du DAC2	S	vers modules de multiplexage
nCS_VCCS	Signal nCS du DAC2	S	vers modules de multiplexage
nSHDN_VCCS	Signal nSHDN du DAC2 (fixé à zéro)	S	vers module de multiplexage
nRS_VCCS	Signal nRS du DAC2 (fixé à zéro)	S	vers module de multiplexage
SCLK_VTH	Horloge de fonctionnement commune à DAC1	S	vers module de multiplexage
SCLK_VCCS	Horloge de fonctionnement commune à DAC2	S	vers module de multiplexage
IDLING_VTH	Signal indiquant que module est prêt pour envoi au DAC1	S	interne (PS)
IDLING_VCCS	Signal indiquant que module est prêt pour envoi au DAC2	S	interne (PS)

* DAC (12 bits) générant V_{vth} et $V_{\text{ref post}}$.
** DAC (12 bits) générant $V_{\text{tune meas}}$ et $V_{\text{ref pre}}$.

Les signaux DATA_VTH et DATA_VCCS contiennent l'adresse et le code de la tension à générer par le DAC. C'est le code en C exécuté dans la partie CPU qui détermine ces valeurs et les écrits aux adresses mémoires de communication de l'interface AXI correspondante (module CTRL CMDCORE en zone A).

3. PostCore

Le module PostCore est illustré sur la FIGURE 3.18 et ses entrées/sorties sont données TABLE 3.12. Ce module transmet les signaux de contrôle des interrupteurs SLIF, SLOG, SLIN et SRM au module de multiplexage dédié et a également à charge d'arbitrer les événements postsy-

naptiques reçus [32]. Cet arbitrage des événements postsynaptiques évoqués dans le CHAPITRE 2 permet de s'assurer qu'un seul événement postsynaptique est choisi lors de la phase d'inférence si plus d'un événement est reçu entre 2 fronts d'horloge de fonctionnement de ce module. Dans une telle situation, ce module choisit le bloc POST dont l'indice est le plus faible. Une fois l'arbitrage effectué, le neurone choisi est transmis au module LearnCore et à la PS sur une adresse mémoire dédiée au format *one hot* [33] par le signal chosen_post_event. Ce module fonctionne à 1 MHz.

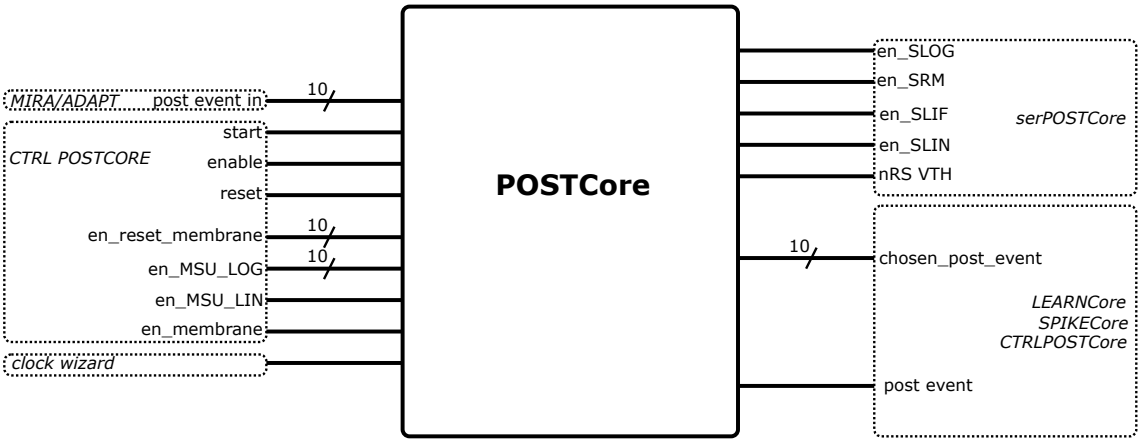


FIGURE 3.18 – Module POSTCore.

TABLE 3.10 – Entrées/sorties du module PostCore.

Signal	Description	Entrée/Sortie	Connexion
start	Signal de démarrage du module pour la détection d'événements*	E	interne (PS)
R	Signal de remise à zéro (réinitialisation) du module	E	intere (PS)
enable	Signal d'activation du module*	E	interne (PS)
post_event_in[9 :0]	Événements postsynaptiques des blocs POST	E	externe (MIRA/ADAPT)
chosen_post_event[9 :0]	Représentation *one hot* du postsynaptique choisi après arbitrage**	S	interne (PS/ LearnCore/SpikeCore)
post_event	Signal indiquant si un événement postsynaptique a été détecté	S	Interne (PS/ LearnCore/SpikeCore)
en_membrane	Signaux pour activer SLIF	E	interne (PS)
en_reset_membrane[9 :0]	Signaux pour activer SRM	E	interne (PS)
en_MSU_LIN	Signaux pour activer SLIN	E	interne (PS)
en_MSU_LOG[9 :0]	Signaux pour activer SLOG	E	interne (PS)
en_SLOG[9 :0]	Signaux de commande des interrupteurs SLOG	S	vers module de multiplexage
en_SLIF	Signal de commande des interrupteurs SLIF	S	vers module de multiplexage
en_SLIN	Signal de commande des interrupteurs SLIN	S	vers module de multiplexage
en_SRM[9 :0]	Signaux de commande des interrupteurs SRM	S	vers module de multiplexage

* Les signaux enable et start sont utilisés pour actionner le module de détection lorsque l'expérimentation en a besoin (pour de l'apprentissage/reconnaissance). Pour des expérimentations type tracé d'hystérésis, la détection d'événements n'est pas nécessaire.

** Si aucun événement postsynaptique a été détecté, ce registre est rempli de 0.

4. SpikeCore

Le module SpikeCore est illustré sur la FIGURE 3.19 et ses entrées/sorties données TABLE 3.11.

32. Les événements reçus sont envoyés à des bascules RS asynchrones en entrée pour s'assurer que la partie VHDL conserve l'information sur un événement postsynaptique même si un neurone LIF repasse en dessous de la tension de membrane de déclenchement (à cause du phénomène de décharge).

33. Ce format *one hot* consiste à transmettre un registre contenant un seul bit à 1 qui se trouve l'indice correspondant au numéro du neurone qui a tiré.

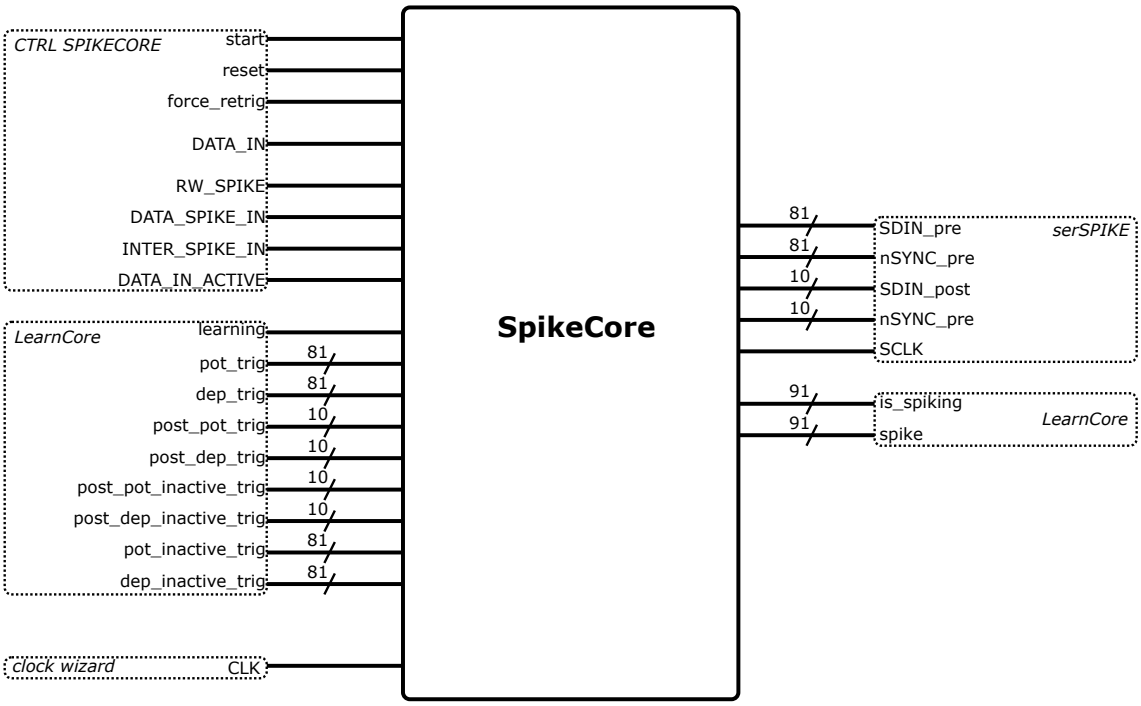


FIGURE 3.19 – Module SpikeCore.

TABLE 3.11 – Entrées/sorties du module SpikeCore.

Signal	Description	Entrée/Sortie	Connexion
R	Remise à zéro (réinitialisation) du module	E	interne (PS)
start	Démarrage du module	E	interne (PS)
DATA_IN[15 :0]	Données en entrée d'un module de configuration	E	interne (PS)
RW_SPKIE	Configure les points mémoire des formes d'onde en lecture ou écriture	E	interne (PS)
DATA_SPKIE_IN	Code du niveau de tension du DAC à enregistrer en mémoire	E	interne (PS)
INTER_SPIKE_IN	Temps de maintien du niveau de tension à écrire en mémoire	E	interne (PS)
DATA_IN_ACTIVE	Indique si de la forme d'onde est active pour ce point, à écrire en mémoire	E	interne (PS)
CLK	Horloge de fonctionnement du module	E	interne
force_retrig	Autorise le re-déclenchement (ou non) si une forme d'onde est en cours	E	interne (PS)
learning	Indique si le réseau entre en phase d'apprentissage (modification des poids)	E	Interne (SpikeCore)
pot_trig[80 :0]	Signal de déclenchement de la forme présynaptique de potentiation	E	interne (SpikeCore)
post_pot_trig[9 :0]	Signal de déclenchement de la forme postsynaptique de potentiation	E	interne (SpikeCore)
dep_trig[80 :0]	Signal de déclenchement de la forme présynaptique de dépression	E	interne (SpikeCore)
post_pot_trig[9 :0]	Signal de déclenchement de la forme postsynaptique de dépression	E	interne (SpikeCore)
pot_inactive_trig[80 :10]	Signal de déclenchement de la forme présynaptique inactive si potentiation	E	interne (SpikeCore)
post_pot_inactive_trig[80 :10]	Signal de déclenchement de la forme postsynaptique inactive si potentiation	E	interne (SpikeCore)
dep_inactive_trig[80 :10]	Signal de déclenchement de la forme présynaptique inactive si dépression	E	interne (SpikeCore)
post_dep_inactive_trig[80 :10]	Signal de déclenchement de la forme postsynaptique inactive si dépression	E	interne (SpikeCore)
is_spiking[91 :0]	Indique par bloc SPK et POST si une forme d'onde est en cours d'application	S	interne (LearnCore)
spike[81 :0]	Indique le déclenchement d'un événement présynaptique	S	interne (LearnCore)
SDIN_pre[80 :0]	Signal de données sérialisées pour les DAC des blocs SPK	S	vers module de multiplexage
nSYNC_pre[80 :0]	Signal nSYNC pour les DAC des blocs SPK	S	vers module de multiplexage
SDIN_post[9 :0]	Signal nSYNC pour les DAC des blocs POST	S	vers module de multiplexage
nSYNC_post[9 :0]	Signal nSYNC pour les DAC des blocs POST	S	vers module de multiplexage
SCLK	Horloge de fonctionnement de tout les blocs et sous-blocs SPK (pré et post)	S	vers module de multiplexage

* Les signaux enable et start sont utilisés pour actionner le module de détection lorsque l'expérimentation en a besoin (pour de l'apprentissage/reconnaissance). Pour des expérimentations de type tracé d'hystérésis, la détection d'événements n'est pas nécessaire.

** Si aucun événement postsynaptique n'a été détecté, ce registre est rempli de 0.

Ce module spikeCore sert à envoyer les données aux blocs SPK de la carte MIRA pour la génération des formes d'onde présynaptiques et postsynaptiques. Ces formes d'onde sont enregistrées dans des blocs mémoires dont les valeurs sont modifiables par le code en C s'exécutant dans la partie PS du Zynq. Ce module spikeCore contient 91 sous-modules « spike » illustrés sur la FIGURE 3.20 permettant de générer et transmettre les points contenus dans ces mémoires et qui constituent une forme d'onde. Ces formes d'onde sont représentées dans la partie logique programmable par un groupement de 3 vecteurs de données (mémoires) de dimension X égal à :

→ 1 pour la transmission d'un niveau de tension continu tel que $V_{max} = V_{ref}$ (code numérique hexadécimal 0xFFF à transmettre au DAC du bloc SPK utilisé), $V_{min} = -V_{ref}$ (code 0x000) ou encore V_{mid} =0 V (code 0x7FF).

→ 100 pour contenir les points de données des niveaux de tensions constituant une forme d'onde.

Ces trois vecteurs de données constituant la représentation d'une forme d'onde sont :

- data_spike[0 : X-1] : Vecteur de données contenant les codes 12 bits des points de tension à appliquer avec les DAC AD5443 des blocs SPK sur la carte MIRA.
- inter_spike[0 : X-1] : Vecteur de données contenant le temps de maintien du point de tension. Le code minimal utilisable correspond à 2 µs pour un bloc SPK fonctionnant à 25 MHz voire 1 µs pour un bloc SPK fonctionnant à 50 MHz [34]. Le code inter_spike[i]

34. Le fonctionnement à 50 MHz a été utilisé sur une carte de test réduite détaillée par la suite.

= 0 au point mémoire i est réservé pour signifier la fin de la forme d'onde. Les points data_spike[j] suivants (j>i) dans les vecteurs de données ne seront pas transmis aux blocs SPK.

- data_active[0 : X-1] : Ce vecteur de données est utilisé pour indiquer si le point de tension en cours (maintenu jusqu'à la fin de la valeur de inter_spike[i]) correspond à une forme d'onde en cours d'application (data_active[i]=1) ou non (data_active[i]=0). Cette valeur est transmise au module LearnCore et est notamment utilisée pour appliquer les règles d'apprentissage 0P0D ou 0P1D qui ont besoin de cette information.

Chaque sous-module « spike » contient 5 mémoires de taille X = 100 permettant d'utiliser en cours d'expérimentation jusqu'à 5 formes d'onde différentes par neurone présynaptique ou postsynaptique. Ces sous-modules disposent également de 4 mémoires de taille X = 1 pour 4 niveaux de tension continue. Les valeurs contenues dans ces mémoires sont calculées et remplies par le code s'exécutant dans la partie PS du Zynq donnant ainsi un grand degré de liberté sur les formes d'onde que l'on peut utiliser [35].

Le déclenchement des formes d'onde (envoi des données aux blocs SPK de la carte MIRA) et leur choix parmi les 9 disponibles (5 formes à 100 points maximum et 4 niveaux de tension continue) suivant l'expérimentation sont également régis par ce module spikeCore. On distingue 2 mode de déclenchement :

- **Mode inférence** : Ce module spikeCore est équipé par sous-module « spike » présynaptique ou postsynaptique d'un vecteur de données de dimension 10 dont les valeurs sont modifiables par le code en partie PS. Ces 10 points correspondent à un instant de déclenchement allant de l'instant de départ à maximum 3 ms avec un décalage minimum entre ces 10 points de 1 µs [36]. Chaque neurone (présynaptique ou postsynaptique) peut ainsi émettre entre 0 et 10 événements. La forme d'onde choisie (parmi les 9 disponibles) à appliquer sur les synapses suivant ce vecteur de 10 instants de déclenchement est sélectionnée de manière indépendante pour chaque neurone (présynaptique ou postsynaptique) depuis le code en C dans la partie PS.
- **Mode apprentissage** : Déclenchement d'une forme d'onde choisie parmi les 9 disponibles par des signaux de déclenchements externes qui sont uniques pour tous les sous-modules « spike » présynaptiques ou postsynaptiques. Ces signaux sont générés par le module LearnCore et la forme d'onde à utiliser, suivant le signal de déclenchement utilisé, est configurée par le code en C et peut être différente selon le sous-module et le signal de déclenchement.

35. Jusqu'à 9 formes d'onde (5 possédant jusqu'à 100 points de tensions et 4 niveaux de tension continue) qui peuvent être différentes pour chaque neurone présynaptique ou postsynaptique puisque les mémoires sont définies par neurone.

36. Le code 0 dans ces mémoires correspond à aucun événement et le code 1 correspond au premier instant de déclenchement.

En mode apprentissage, il existe 4 signaux de déclenchement différents qui sont différenciés entre neurones présynaptiques et neurones postsynaptiques. Pour les présynaptiques il s'agit de pot_trig, dep_trig, pot_inactive_trig et dep_inactive_trig. Pour les postsynaptiques il s'agit de post_pot_trig, post_dep_trig, post_pot_inactive_trig et post_dep_inactive_trig. Ces 4 signaux pour les neurones présynaptiques ou postsynaptiques assurent le déclenchement des formes d'onde à utiliser pour modifier les poids synaptiques lors d'une phase d'apprentissage comme cela est décrit dans le détail du module LearnCore par la suite.

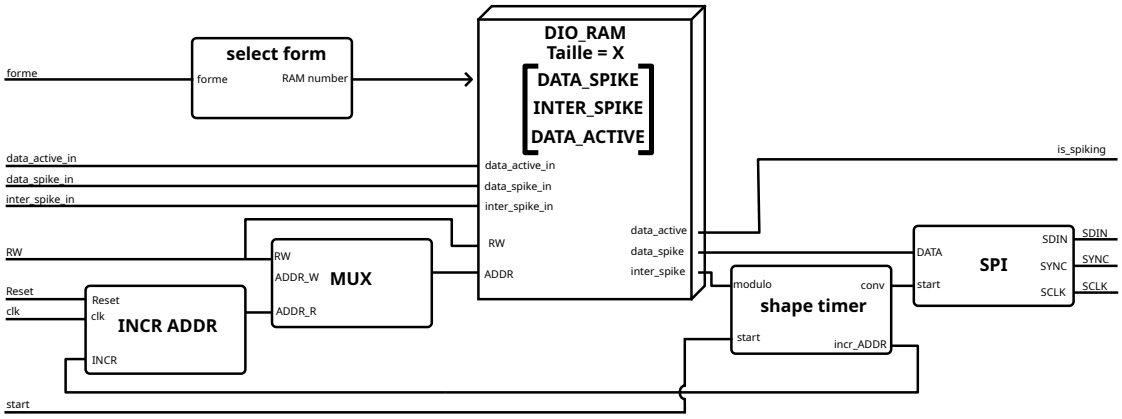


FIGURE 3.20 – Sous-module « spike » simplifié de transmission des points de la forme d'onde synaptique à générer.
Le signal forme permet de choisir quelle mémoire (forme d'onde) est utilisée parmi les 8 disponibles par ce sous-module « spike ». Le bloc select form permet de sélectionner la mémoire ou forme d'onde a utiliser. Le bloc INCR ADDR permet d'incrémenter l'indice des valeurs en mémoire à chaque déclenchement du signal interne incr_ADDR qui est issu du bloc shape timer qui sert à maintenir pendant la durée inter_spike le point en cours de la forme d'onde contenue en mémoire. Concernant les signaux, start permet de déclencher la lecture des points d'une forme et d'envoyer ces données à la carte MIRA. SDIN, SYNC et SCLK correspondent aux données à transmettre aux ADC des blocs SPK. Le signal is_spiking (contenant la valeur de data_active) est transmis au module LearnCore pour savoir si une forme est en cours d'application. RW est utilisé pour choisir si les mémoire sont en mode lecture (génération des formes d'onde) ou écriture (enregistrement des données data_active_in, data_spike_in et inter_spike_in à l'addresse ADDR_W). Reset permet de réinitialiser si besoin le module permettant de lire successivement les points en mémoire des formes d'onde. Clk correspond à l'horloge interne de fonctionement.

5. LearnCore

Ce module, en cours de développement lors de l'écriture de ce manuscrit, sert à déclencher les formes d'onde présynaptiques et postsynaptiques lors de la phase d'apprentissage pour modifier les poids synaptiques du crossbar.

Pour pouvoir fonctionner et implémenter les différentes règles d'apprentissage que l'on souhaite utiliser, ce module a besoin de 2 informations provenant du module PostCore :

- post_event qui permet de savoir si un événement postsynaptique a eu lieu.
- chosen_post_event[9 : 0] pour identifier quel neurone postsynaptique doit subir une phase d'apprentissage.

Ce module nécessite également 2 informations provenant du module SpikeCore :

- is_spiking[90 : 0] permettant de savoir si un neurone est en cours d'application pour les règles 0P0D et 0P1D.
- spike[90 : 0] permettant de savoir si un neurone vient d'être déclenché pour incrémenter les compteurs N_{fire} [37] des neurones présynaptiques pour les règles d'apprentissage qui en ont besoin.

Une version initiale de ce module permettait de déclencher 4 types de formes d'onde sur le crossbar pour réaliser l'apprentissage comme illustré sur la FIGURE 3.22.A. Lorsqu'un neurone postsynaptique est détecté, cette version du module déclenchait toutes les formes d'onde pour la potentialisation et la dépression sur les entrées présynaptiques ainsi que la forme postsynaptique permettant de modifier tous les poids. Cette méthode pour réaliser l'apprentissage en une seule passe a posé problème lors des premiers tests sur la plateforme. En effet la modification simultanée des poids synaptiques pour la potentialisation puis la dépression cause la génération de courants issus des blocs postsynaptiques importants [38] à cause du comportement non linéaire des memristors ferroélectriques (similaire à une diode [39]), phénomène auquel s'est combiné l'utilisation de memristors avec des résistances plus faibles que prévues lors des tests. Des courants d'autant plus importants que le nombre de memristors que l'on modifie simultanément est important. Ces courants, trop importants pour les CCII+ utilisés sur la carte MIRA, provoquaient le dysfonctionnement de la copie en tension [40] de ces derniers.

Pour pallier ce problème, le nouveau module en cours de développement lors de l'écriture de ce manuscrit réduit les courants nécessaires lors de la phase d'écriture du crossbar pour l'apprentissage en réalisant une modification séquentielle des memristors comme illustré sur la FIGURE 3.22.B. Avec cette méthode, lorsqu'un neurone postsynaptique émet un événement, les courants que les CCII+ des blocs POST ont à gérer sont réduits par rapport à ceux de la méthode précédente. En effet, en ne modifiant qu'un seul memristor à la fois, seul ce dernier se voit appliquer des tensions importantes de modification de son poids synaptique limitant ainsi les courants importants liés à la non-linéarité de ces composants. Lors des premiers tests de ce

37. Les compteurs N_{fire} utilisés pour savoir combien de fois un neurone présynaptique a émis d'événements avant le déclenchement d'un événement postsynaptique sont implémentés dans ce module LearnCore.
38. Le postsynaptique doit gérer le courant de toutes les synapses qui lui sont connectées. Ce courant atteint des valeurs importants lors des phases d'apprentissage.
39. Comme évoqué partie 1.2.2 Le choix des synapses : Le memristor ferroélectrique, un nanocomposant.
40. Pour des courants trop élevés, la copie en tension de la borne Y vers la borne X n'est plus fonctionnelle : la contre-réaction permettant cette copie sature en courant

module, les neurones postsynaptiques n'ont pas présenté de défaillance de copie en tension.

Cette méthode de modification successive des poids synaptiques lors de l'apprentissage nécessite que ce module LearnCore transmette au module SpikeCore des signaux de déclenchement de formes d'onde différentes à tous les neurones présynaptiques et postsynaptiques pour pouvoir réaliser les combinaisons de tensions souhaitées sur le crossbar. Ces signaux (FIGURE 3.22.B) permettent de déclencher 4 types de formes d'onde différentes pour les présynaptiques[41] et 4 types de formes d'onde différentes pour les postsynaptiques[42] qui sont enregistrées dans les formes d'onde mémorisées des neurones présynaptiques ou postsynaptiques du module SpikeCore. Ainsi, lorsque l'on souhaite faire de l'apprentissage, 4 mémoires de formes d'onde des neurones dans SpikeCore doivent être réservées et configurées pour les formes d'apprentissage en plus de celles utilisées pour l'inférence amenant ainsi aux 5 formes d'onde disponibles par neurone dans le module SpikeCore.

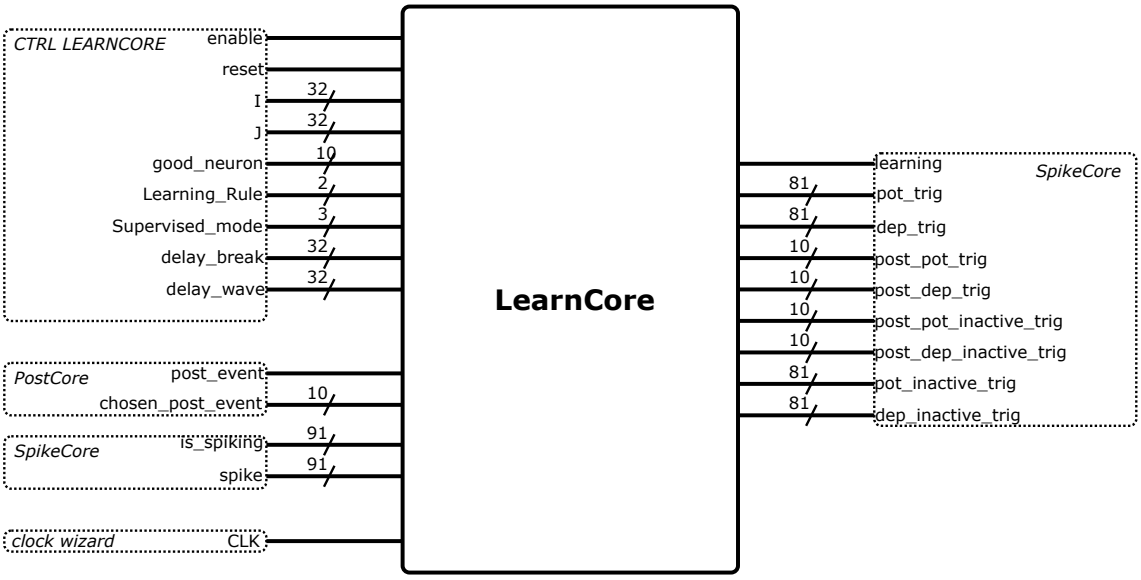


FIGURE 3.21 – Module LearnCore.

41. Il s'agit des signaux de déclenchement pot_trig, dep_trig, pot_inactive_trig et dep_inactive_trig.
42. Il s'agit des signaux post_pot_trig, post_dep_trig, post_dep_incative_trig et post_dep_inactive_trig.

TABLE 3.12 – Entrées/sorties du module LearnCore.

Signal	Description	Entrée/Sortie	Connexion
enable	Active le module pour réaliser l'apprentissage	E	interne (PS)
R	Remise à zéro (réinitialisation) du module	E	interne (PS)
post_event	Indique si un postsynaptique a été détecté et arbitré	E	interne (PostCore)
chosen_post_event	Représentation *one hot* du postsynaptique choisi après arbitrage	E	interne (PostCore)
is_spiking[91 :0]	Indique si une forme d'onde est en cours d'application	E	interne (SpkieCore)
spike	Signale le déclenchement d'un événement pré-synaptique	E	interne (SpikeCore)
I[31 :0]	Valeur binaire de i pour les règles $iPjD$ (sur 32 bits)	E	interne (PS)
J[31 :0]	Valeur binaire de j pour les règles $iPjD$ (sur 32 bits)	E	interne (PS)
good_neuron[9 :0]	Indique quels neurones sont corrects pour l'échantillon en cours si supervision.	E	interne (PS)
LearningRule[1 :0]	Choix de la règle d'apprentissage (0P0D, 0P1D ou 1P1D)	E	interne (PS)
SupervisedMode[2 :0]	Choix du mode de supervision	E	interne (PS)
delay_break[31 :0]	Valeur du délai avant déclenchement de l'apprentissage	E	interne (PS)
delay_wave[31 :0]	Valeur du délai entre application des tensions d'apprentissage	E	interne (PS)
learning	Indique si le réseau entre en phase d'apprentissage (modification des poids)	S	Interne (SpikeCore)
pot_trig[80 :0]	Signal de déclenchement de la forme présynaptique de potentiation	S	interne (SpikeCore)
post_pot_trig[9 :0]	Signal de déclenchement de la forme postsynaptique de potentiation	S	interne (SpikeCore)
dep_trig[80 :0]	Signal de déclenchement de la forme présynaptique de dépression	S	interne (SpikeCore)
post_pot_trig[9 :0]	Signal de déclenchement de la forme postsynaptique de dépression	S	interne (SpikeCore)
pot_inactive_trig[80 :10]	Signal de déclenchement de la forme présynaptique inactive si potentiation	S	interne (SpikeCore)
post_pot_inactive_trig[80 :10]	Signal de déclenchement de la forme postsynaptique inactive si potentiation	S	interne (SpikeCore)
dep_inactive_trig[80 :10]	Signal de déclenchement de la forme présynaptique inactive si dépression	S	interne (SpikeCore)
post_dep_inactive_trig[80 :10]	Signal de déclenchement de la forme postsynaptique inactive si dépression	S	interne (SpikeCore)

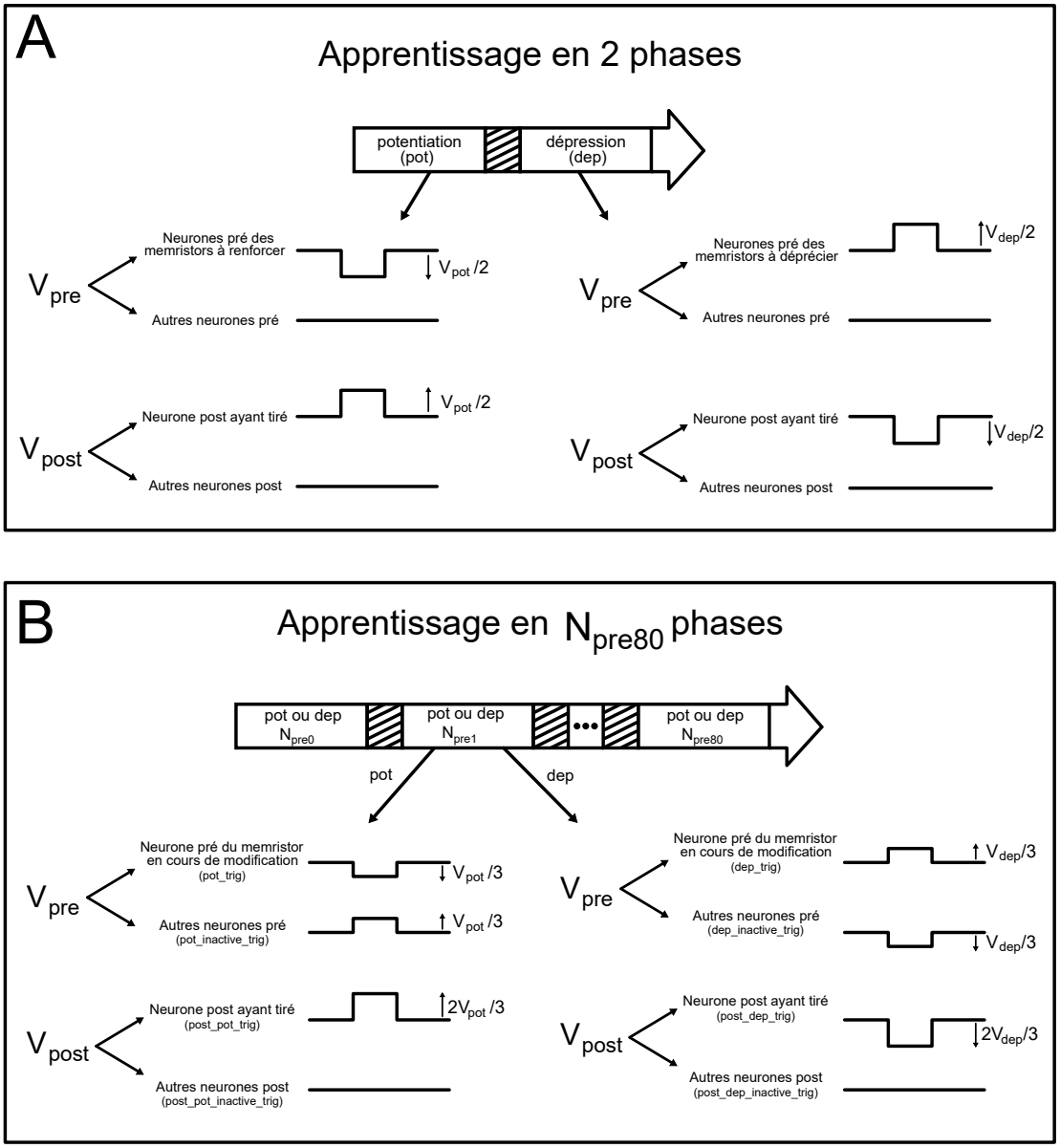


FIGURE 3.22 – Combinaison des formes d'onde déclenchées pour réaliser un apprentissage.
A. Version initiale des combinaisons de tension utilisées pour l'apprentissage résultant en des courants circulant dans le crossbar trop importants pour les CCII+ postsynaptiques.
On suppose que $\pm V_{\text{dep}}/2$ et $\pm V_{\text{pot}}/2$ ne suffisent pas à modifier les poids synaptiques puisqu'il s'agit des tensions que l'on retrouve sur les bornes des memristors du crossbar que l'on ne cherche pas à modifier.
B. Version des combinaisons de tensions pour apprentissages successifs permettant de réduire les courants que les CCII+ ont à gérer.
On suppose que $\pm V_{\text{pot}}/3$ et $\pm V_{\text{dep}}/3$ ne suffisent pas à modifier les poids synaptiques puisqu'il s'agit des tensions que l'on retrouve sur les bornes des memristors du crossbar que l'on ne cherche pas à modifier.

3.4.2 Zone code en C : Algorithmes de contrôle et de configuration

Afin de piloter tous les modules VHDL qui permettent de contrôler la carte MIRA pour réaliser de l'inférence, de l'apprentissage ou faire l'acquisition des courants synaptiques pour la caractérisation du crossbar, un ensemble de bibliothèques écrites en langage C ont été développées pour :

— Piloter les interfaces AXI4-Lite et ainsi communiquer avec les modules VHDL.

— Exécuter des fonctions de plus haut niveau décrivant des expérimentations à réaliser (mesure, application de tensions sur le crossbar de memristors, ...).

L'architecture générale de ce code en C est illustrée sur la FIGURE 3.23. La stratégie choisie pour l'écriture du code des expérimentations considère l'écriture d'un fichier app.c différent pour chaque expérimentation à réaliser. Cette architecture est constituée de 7 bibliothèques et 1 fichier de définitions écrits durant ces travaux de thèse et qui ont pour rôle de gérer la communication avec les modules VHDL par l'échange d'informations par des registres 32 bits à des adresses mémoires fixes ainsi que la réalisation de certains algorithmes de base nécessaires au bon fonctionnement de certaines expérimentations. Ces bibliothèques sont :

→ **ad8802.[c/h]** : Permet de contrôler le module CMDCore (VHDL) pour définir et appliquer les tensions V_{th}, $V_{\text{tune meas}}$, $V_{\text{ref pre}}$ et $V_{\text{ref post}}$ par bloc GEN sur la carte MIRA.

→ **LearnCore.[c/h]** : Permet de contrôler et configurer le module LearnCore (VHDL) en charge de l'apprentissage du réseau de neurones.

→ **MeasureCore.[c/h]** : Permet de contrôler et configurer le module ACQCore (VHDL) pour récupérer les valeurs numériques correspondant aux tensions V_{log} et V_{lin} des blocs POST acquises par le bloc ACQ sur la carte MIRA.

→ **protocols.[c/h]** : Utilisée pour la définition de routines plus complexes comme par exemple l'acquisition des poids d'une ligne présynaptique suivant le protocole de mesure des poids synaptiques donné ANNEXE E.

→ **SD.[c/h]** : Permet la définition des fonctions nécessaires à la lecture et l'écriture des données sur la carte SD utilisée pour stocker des informations lors d'une expérimentation.

→ **SpikeCore.[c/h]** : Permet de contrôler/piloter le module SpikeCore (VHDL). Elle permet notamment d'enregistrer les différentes formes d'onde des neurones présynaptiques et postsynaptiques, l'enregistrement des instants de déclenchement, ainsi que le choix des formes d'onde à utiliser lors d'une phase d'apprentissage.

→ **waveform_compute.[c/h]** : Contient l'ensemble des fonctions permettant de calculer les points des formes d'onde que l'on souhaite utiliser dans le module SpikeCore. Cette bibliothèque repose sur la définition de formes d'onde élémentaires (impulsion, rampe, exponentielle) que l'on peut combiner successivement grâce à des fonctions appelées « constructeurs » permettant de calculer les points d'une forme d'onde utilisant une à six formes d'onde élémentaires dans sa définition. Des exemples de formes d'onde générées

par les blocs SPK définis par ces « constructeurs » sont illustrés dans la partie 4.1 Vérification des blocs électroniques, modules et algorithmes sur carte réduite du CHAPITRE 4.

MIRAaddr.h est un fichier de définition des adresses mémoires de communication des interfaces AXI pour la plateforme MIRA.

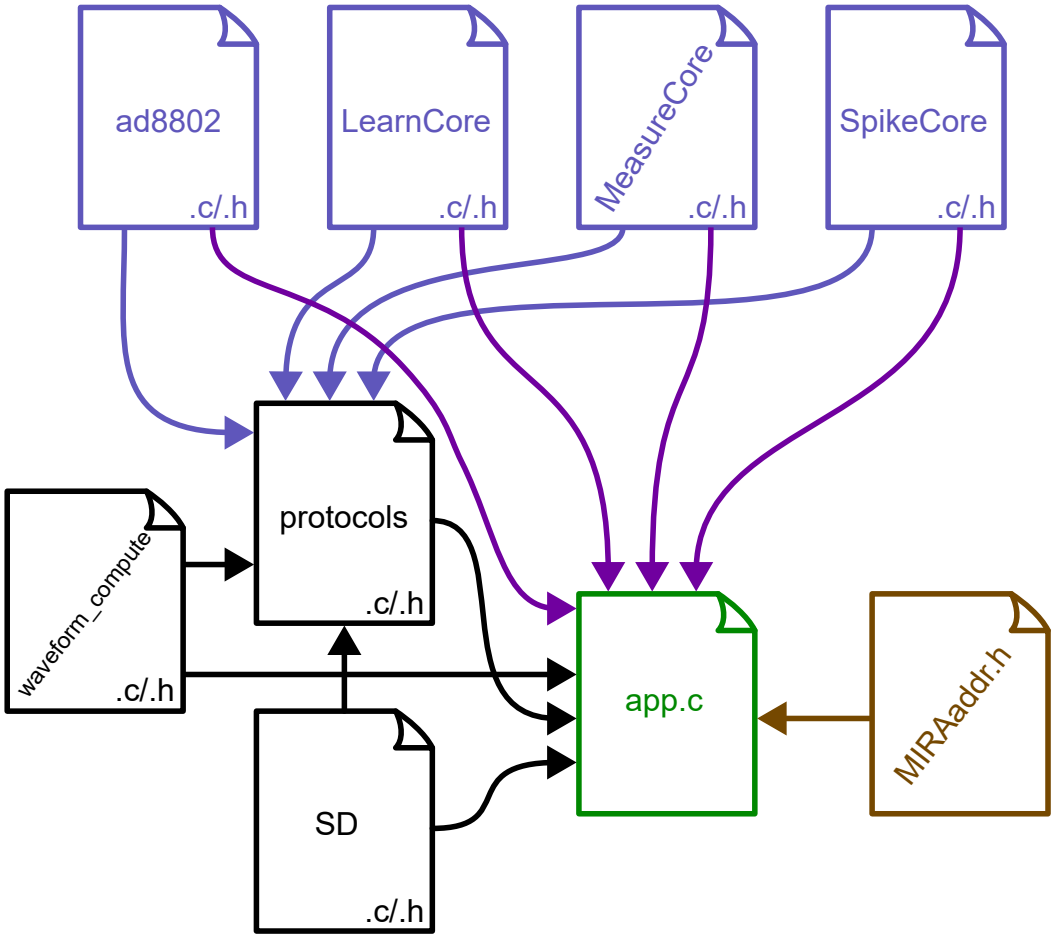


FIGURE 3.23 – Structure simplifiée du code en C permettant le contrôle de la plateforme. Cette structure ne représente que les algorithmes écrits lors de ces travaux de thèse et non toutes les sources nécessaires pour réussir à configurer et compiler le code s'exécutant sur le Zynq de la ZCU102. Le fichier app.c représente le code de l'expérimentation à réaliser.

3.5 Conclusion

Ce CHAPITRE 3 qui s'inscrit dans le deuxième axe de ces travaux de thèse a abordé les différentes cartes électroniques qui constituent cette plateforme qui ont été réalisées (à l'exception de la carte ZCU102) lors de ces travaux de thèse, ainsi que toute la partie contrôle numérique écrite en VHDL et en C également développé lors de ces travaux de thèse. La plateforme 81 × 10 qui a été mise au point intègre un réseau de neurones événementiels analogiques de 81 neurones présynaptiques et 10 neurones postsynaptiques LIF à résistance de décharge connectés par des synapses memristives sous forme de crossbar. Les neurones postsynaptiques sont équipés en plus de leur partie neurone, d'unités de mesure permettant, au besoin, de déterminer la valeur des memristors constituant cette matrice de 810 synapses.

Ces travaux retranscrit dans ce CHAPITRE 3 ont ainsi permis de faire émerger cette plateforme grâce à un travail qui s'est effectué sur 3 fronts différents :

1. Définition et conception de toute la partie électronique constituant la plateforme.
Cette étape de conception a abouti à la réalisation de 3 cartes électroniques qui implémentent chacune une section importante de la plateforme :

— La carte MIRA qui implémente la partie réseau de neurones événementiels analogiques et circuits de mesure.

— La carte ADAPT qui implémente la partie communication numérique entre la carte MIRA et une carte de contrôle ZCU102 (cette dernière prise sur étagère).

— La carte VCCSS2 qui implémente des circuits complémentaires à la carte MIRA pour améliorer et corriger son fonctionnement.

2. Définition et écriture du DCB de contrôle en VHDL.
Pour pouvoir piloter la plateforme comme un réseau de neurones, le bloc numérique DCB introduit dans le CHAPITRE 2 est nécessaire. Ce bloc numérique implémenté sur la partie logique programmable du Zynq de la ZCU102 contient notamment de nombreux modules dédiés au pilotage des circuits numériques de la plateforme.

2. Définition et écriture des algorithmes de contrôle et d'expérimentation en C.
La carte ZCU102 utilisée pour piloter cette plateforme nécessite l'écriture de code en C à exécuter sur la partie PS du Zynq de cette carte pour piloter le DCB implémenté dans la partie logique programmable du Zynq. Les algorithmes de contrôle qui ont été écrits jusqu'à présent permettent la réalisation d'hystérésis.

Si le travail sur le premier front est achevé (les cartes électroniques ont été réalisées), le travail sur les deux autres (écriture de modules en VHDL et d'algorithme d'expérimentation en C) est

en cours, mais a commencé à produire des résultats concluants.

Le CHAPITRE 4, qui s'inscrit également dans le deuxième axe de ces travaux de thèse, détaillera la vérification par l'expérimentation de certains blocs électroniques et modules de contrôle ainsi que les premières expérimentations réalisées avec cette plateforme.

Chapitre 4

Vérification et utilisation de la plateforme 81 × 10

Ce chapitre, tout comme le CHAPITRE 3 qui était consacré à une description technique de la plateforme 81 × 10, s'inscrit dans le deuxième axe de ces travaux de thèse. Il s'agit de la vérification expérimentale d'un certain nombre de blocs électroniques et numériques de cette plateforme 81 × 10 (afin de s'assurer de leur bon fonctionnement), mais aussi de premières utilisations de cette dernière pour la caractérisation automatisée d'un crossbar de memristors. La première partie de ce chapitre est ainsi consacrée à la vérification expérimentale des neurones présynaptiques et postsynaptiques, des circuits de mesures et des modules de contrôle numérique sur une plateforme réduite (2 neurones présynaptiques et 2 neurones postsynaptiques) permettant d'effectuer des tests et des déploiements de l'architecture de contrôle plus rapidement que sur la plateforme 81 × 10. La deuxième et dernière partie est consacrée à la mise en route de la plateforme ainsi qu'aux premières expérimentations de caractérisation sur memristors qui ont été réalisées avec cette dernière pour extraire automatiquement certains des paramètres des memristors.

4.1 Vérification des blocs électroniques, modules et algorithmes sur carte réduite

L'architecture implémentée sur le Zynq de la ZCU102 est conséquente et nécessite à chaque modification plus d'une heure pour générer le *bitstream* de la partie FPGA. Afin de pouvoir développer plus rapidement la plateforme pour tester les différents modules VHDL et les algorithmes de contrôle ainsi que de vérifier le bon fonctionnement de la partie électronique (blocs SPK, POST, ACQ et GEN), une carte plus petite, illustrée sur la FIGURE 4.1, contenant 2 SPK présynaptiques et 2 blocs POST (au lieu des 81 SPK présynaptiques et 10 POST de MIRA) a été réalisée et est pilotée avec une autre carte (la Zybo-Z7020) utilisant un Zynq plus petit et avec moins de fonctionnalités [1]. La génération du bitstream pour cette version réduite de la plateforme prend moins de temps (une dizaine de minutes) puisque, même si elle reprend les blocs électroniques de MIRA, elle présente des différences qui permettent de réduire drastiquement l'architecture de contrôle implémentée sur la Zybo :

— La commande des blocs électroniques de la carte réduite ne passe pas par des canaux de transmission reposant sur du multiplexage temporel. Cela permet de ne pas avoir à implémenter toute la zone C du DCB étendu.

— La présence de seulement 2 blocs SPK présynaptique et 2 blocs POST permet de réduire de manière significative le nombre de blocs mémoires implémentés sur le FPGA utilisés pour la définition et l'enregistrement des formes du bloc SpikeCore (de 91 pour MIRA à 4 pour la carte réduite) [2].

1. La famille Zynq-7000 possède notamment moins de ressources dans la partie logique programmable.
2. De manière générale, plus l'architecture numérique à implémenter sur la partie logique programmable est petite, moins le temps passé à réaliser la synthèse, l'optimisation et l'implémentation du code VHDL est long.

— Seulement 1 ADC est utilisé pour le bloc ACQ pour acquérir les 4 tensions d'intérêt $V_{\text{log 0}}$, $V_{\text{lin 0}}$, $V_{\text{log 1}}$ et $V_{\text{lin 1}}$ pour les 2 blocs POST de cette carte.

— Seulement 1 DAC est utilisé pour le bloc GEN pour générer les 4 tensions d'intérêt $V_{\text{tune meas 0}}$, $V_{\text{th 0}}$, $V_{\text{tune meas 1}}$ et $V_{\text{th 1}}$ pour les 2 blocs POST de cette carte.

— Les ressources disponibles sur le Zynq de la carte Zybo étant plus restreintes que celle de la ZCU102, la génération du *bitstream* requiert moins de temps.

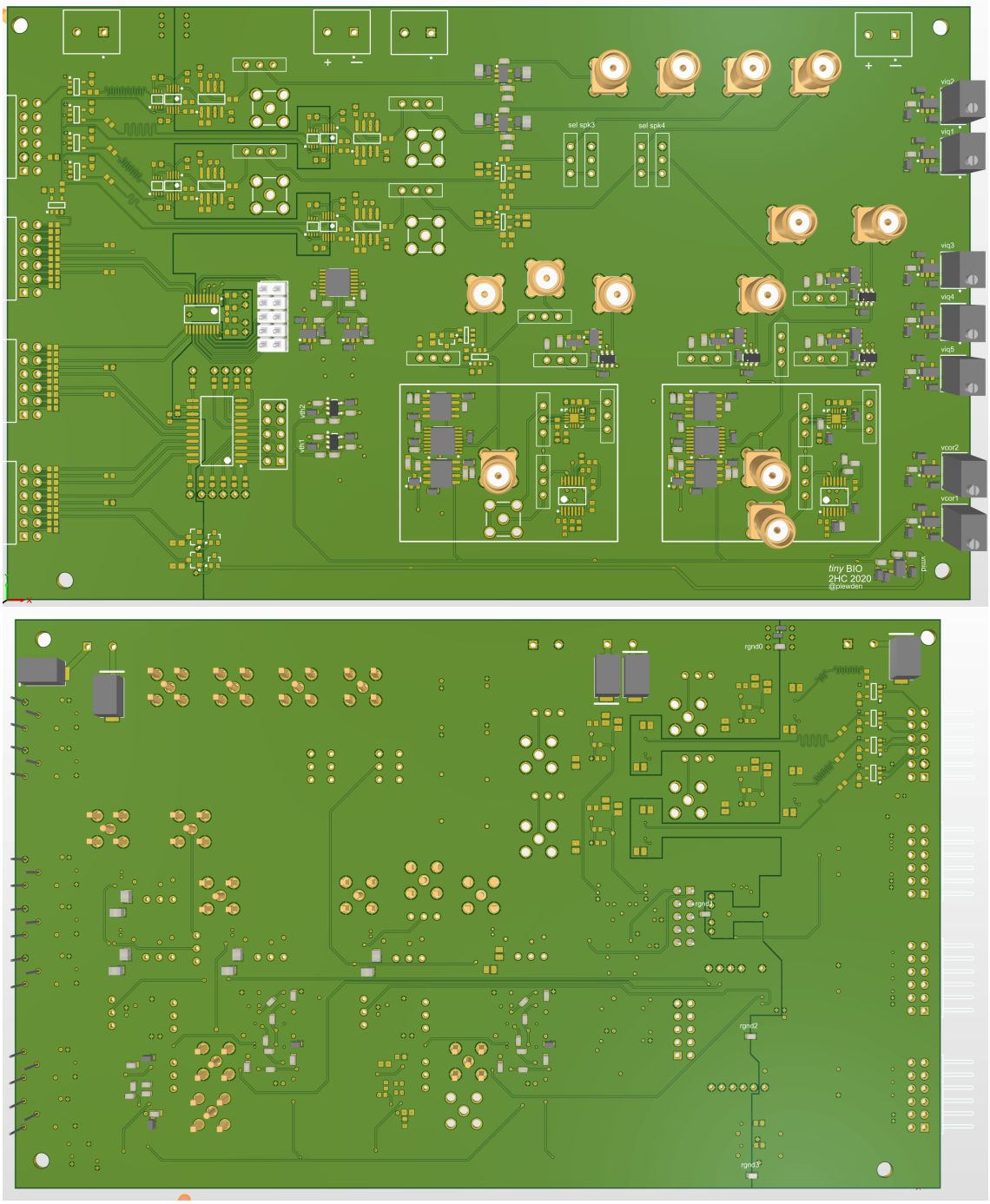


FIGURE 4.1 – Vues de dessus et de dessous de la carte réduite.
Vues obtenues sur le logiciel de conception Altium Designer. Carte 6 couches permettant de tester les circuits électroniques et vérifier à plus petite échelle les modules de la plateforme 81 × 10. Elle présente quelques variantes des circuits électroniques par rapport à la carte MIRA, notamment la présence de CCII- (mise en cascade de deux CCII+) en parallèle des CCII+ pour la réalisation de synapse inhibitrices mais qui ne sont pas l'objet d'étude dans les travaux présentés dans ce manuscrit.

4.1.1 Génération des formes d'ondes

Afin de vérifier la génération des différentes formes d'ondes et leur déclenchement, un oscilloscope a été placé en sortie des modules SPK accessibles via des connecteurs SMA sur cette carte réduite. Les formes d'ondes qui sont illustrées sur la FIGURE 4.2 correspondent à des formes d'ondes dont les points ont : (1) été calculés dans la partie PS du Zynq, puis (2) qui ont été enregistrés dans le module SpikeCore avant que ces formes d'onde soient (3) sélectionnées puis (4) déclenchées par le lancement du module SpikeCore.

Ces formes d'ondes qui sont construites à partir de motifs élémentaires (impulsion, rampe, exponentielle) ont été mesurées à l'oscilloscope sur la carte réduite et ont également été reproduites sur la plateforme 81 × 10 et mettent en évidence la capacité pour la plateforme de générer des formes d'ondes diverses qui peuvent être appliquées sur des memristors.

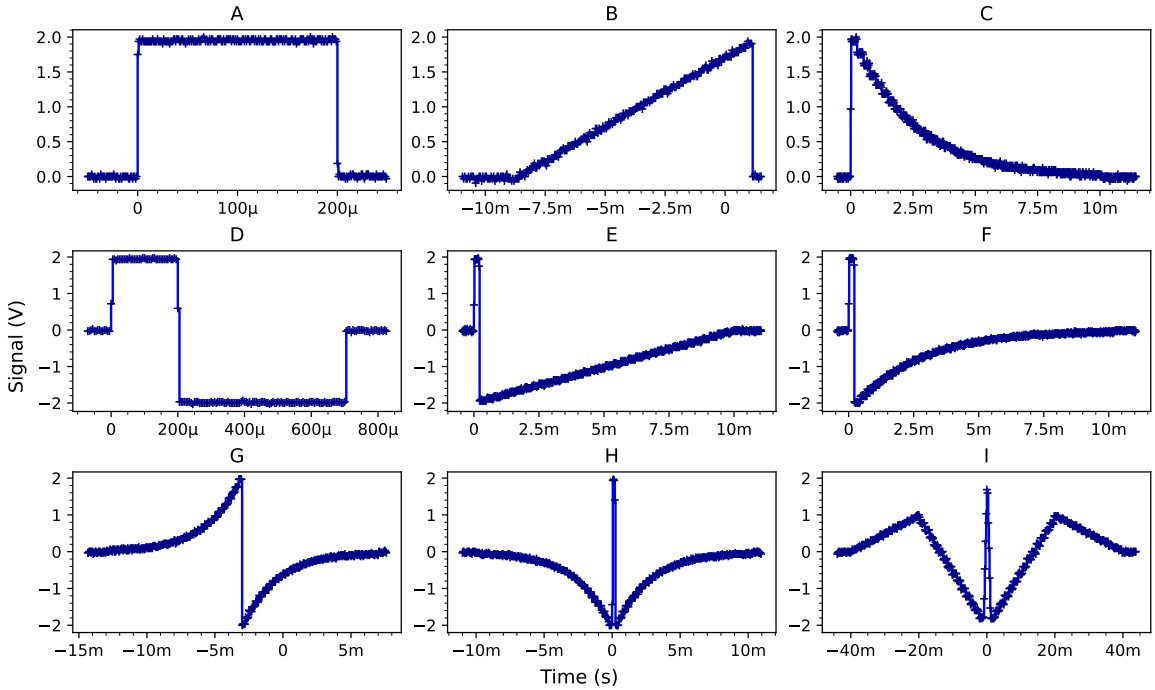


FIGURE 4.2 – Exemples de formes d'ondes générées par le bloc SPK.
Signaux mesurés avec un oscilloscope placé en sortie du suiveur d'un bloc SPK avec $V_{\text{ref pre}} = 2\,\text{V}$.
Les signaux A (pulse), B (rampe) et C (exponentielle) ont été calculés par le « constructeur » à une forme élémentaire de la librairie **waveform_compute.[c/h]**.
Les signaux D (pulse+pulse), E(pulse+rampe) et F(pulse+exponentielle) ont été calculés par le « constructeur » à deux formes élémentaires.
Le signal H (exponentielle+rampe+rampe+exponentielle) a été calculé par le « constructeur » à quatre formes élémentaires.
Le signal I (6 rampes successives) a été calculé par le « constructeur » à six formes élémentaires.

4.1.2 Preuve de fonctionnement de l'étage d'entrée postsynaptique (CCII+)

L'utilisation d'un CCII+ comme étage d'entrée de synapses excitatrices d'un neurone postsynaptique connecté à des synapses memristives a été introduite dans les travaux de Gwendall Lecerf [92]. Ce composant électronique, utilisé pour les simulations du CHAPITRE 2 et choisi pour la réalisation des neurones postsynaptiques du projet ULPEC, n'a pas été validé expérimentalement avec des memristors dans le cadre de ces travaux de thèse. Ainsi, pour démontrer le bon fonctionnement du CCII+ comme étage d'entrée avec des memristors ferroélectriques, deux expérimentations (REF et VCT) ont été réalisées pour reproduire le tracé d'hystérésis d'un memristor avec et sans CCII+, permettant ainsi de vérifier les copies en courant et en tension de ce dernier lorsqu'il pilote un memristor.

Vérification expérimentale avec un CCII+

Le premier montage (REF) illustré sur la FIGURE 4.3 sert à établir un résultat (hystérésis) de référence. Il fait usage d'un banc de mesure constitué d'un générateur de tension arbitraire W1 [3] pour l'application d'impulsions de programmation d'amplitude V_{prog}, d'un picoampèremètre [4] pour la mesure du courant et d'un Bias-T [5] (BT ou T de polarisation) permettant la connexion de ces deux appareils au memristor utilisé. Le pilotage de ces appareils est assuré par un algorithme écrit en Python 3.

Le protocole utilisé pour la réalisation de l'hystérésis du memristor testé est le suivant :

1. Application de N_{pulses} impulsions de programmation d'amplitude V_{min} (tension de programmation minimale utilisée lors de l'hystérésis) afin de rapprocher le memristor de son état de résistance faible ou LRS [6].
2. Mesure initiale du memristor :
   (a) Mesure du courant moyen de référence I_0 avec le picoampèremètre (moyenne de N_{meas} mesures) en appliquant 0 V sur le memristor.
   (b) Mesure du courant moyen I_{meas} avec le picoampèremètre (moyenne de N_{meas} mesures) en appliquant V_{meas} (tension appliquée par le générateur de tension interne du picoampèremètre).
   (c) Calcul de la résistance du memristor $R_{mem} = \dfrac{V_{meas}}{I_{meas} - I_0}$.
3. Pour chaque tension de programmation de l'hystérésis :
   (a) Application de N_{pulse} impulsions de programmation d'amplitude V_{prog} choisie et de largeur T_{pulse}.

3. Appareil Keysight 81160A.
4. Appareil Keithley 6847.
5. Référence ZFBT-6GW+.
6. Low Resistance State.

(b) Mesure du memristor et calcul de sa résistance (même protocole que lors de la mesure initiale du memristor).

4. Répétition de l'étape 3 pour atteindre le nombre de cycles (N_{cycle}) souhaité.

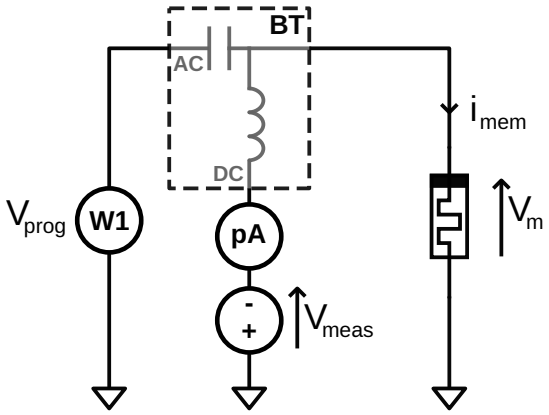


FIGURE 4.3 – Montage REF.

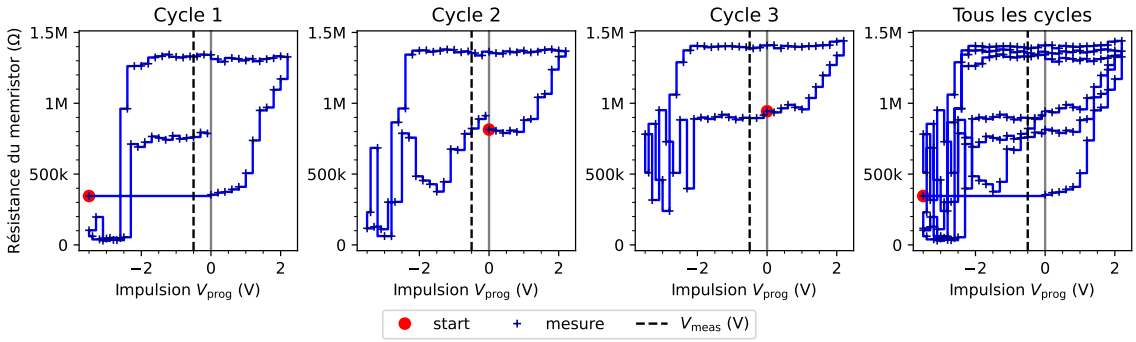


FIGURE 4.4 – Hystérésis obtenues avec le montage REF.
Résultats obtenus avec l'échantillon L5C5 (memristor BFO[86]) avec $N_{meas} = 5$, $V_{meas} = 0.5\,V$, $N_{cycles} = 3$, $N_{pulses} = 10$, $V_{max} = 2.2\,V$, $V_{min} = -3.5\,V$, $V_{step} = 0.2\,V$, $T_{pulse} = 1\,\mu s$.

Les résultats (FIGURE 4.4) obtenus avec le montage REF mettent en évidence le comportement attendu du memristor. Au-delà de 1 V la résistance de ce dernier change (augmente) de manière non volatile avant d'atteindre à 2.2 V un palier de 1.35 MΩ environ. Pour changer de nouveau l'état du memristor de manière non volatile et baisser sa résistance, il faut appliquer des impulsions inférieures à −2.5 V. Les tensions de seuil de ce memristor sont donc 1 V et −2.5 V. Malgré un comportement similaire cycle à cycle (plage de valeur, tension de seuil), ce memristor ne présente pas de répétabilité importante cycle à cycle, notamment pour les niveaux de résistance minimum et maximum qui sont visiblement distincts sur le tracé « Tous les cycles » FIGURE 4.4.

Le deuxième montage (VCT : Vérification Courant Tension) illustré FIGURE 4.5 permet de vérifier la copie en courant et en tension. La copie en courant est vérifiée en comparant les courants obtenus par les deux picoampèremètres (celui placé en entrée DC du Bias-T et celui ajouté en sortie du CCII+). La copie en tension est vérifiée en répartissant la tension de programmation entre la borne PRE du memristor et la borne Y du CCII+ et en comparant l'hystérésis obtenue à celle du montage REF. Le CCII+ est réalisé avec le composant discret OPA861.

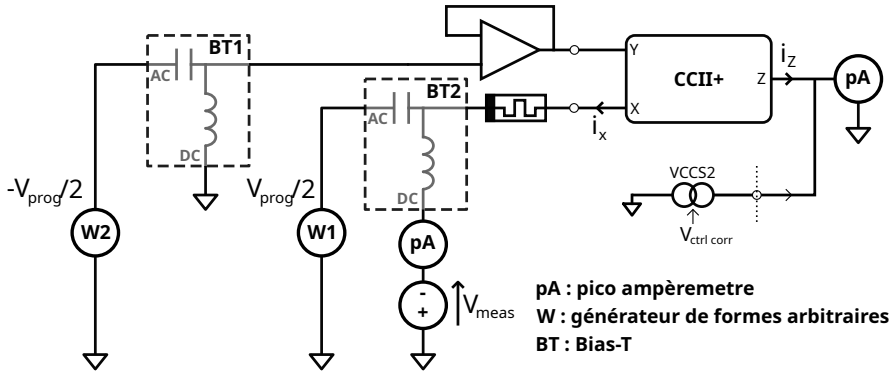


FIGURE 4.5 – Montage VCT de validation d'utilisation d'un CCII+
La source de courant VCCS2 en parallèle du deuxième picoampèremètre en sortie Z du CCII+ est utilisée pour ajuster la sortie en courant du CCII+ qui peut présenter plusieurs microampères de courant de décalage au repos. Le suiveur à la borne Y du CCII+ fait partie du bloc POST.

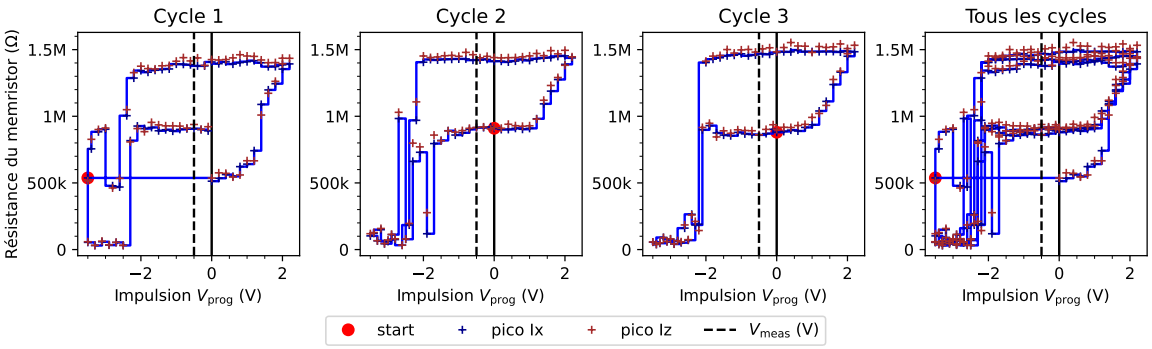


FIGURE 4.6 – Hystérésis obtenues avec le montage VCT.
Résultats obtenus avec l'échantillon L5C5 (memristor BFO[86]) avec $N_{meas} = 5$, $V_{meas} = 0.5\,V$, $N_{cycles} = 3$, $N_{pulses} = 10$, $V_{max} = 2.2\,V$, $V_{min} = -3.5\,V$, $V_{step} = 0.2\,V$, $T_{pulse} = 1\,\mu s$.

Les résultats (FIGURE 4.6) obtenus avec le montage VCT mettent en évidence un comportement du memristor similaire au montage REF. Les tensions de seuil du memristor L5C5 sont 1 V et −2.5 V comme avec le montage REF alors que la tension de programmation est répartie entre la borne PRE du memristor et la borne Y du CCII+. Cela met en évidence le bon fonctionnement de la copie en tension assuré par le CCII+ qui permet d'avoir V_{prog} appliqué aux bornes du memristor. Concernant la copie en courant du CCII+, lorsque l'on compare les valeurs obte-

nues avec le picoampèremètre placé en amont mesurant i_X et celui en aval mesurant $i_Z - i_{VCCS2}$ (courant copié ajusté) on constate que le comportement mesuré est quasiment identique malgré un petit décalage des mesures notamment dû au facteur de copie en courant [7].

Ainsi, à la vue des résultats présentés FIGURE 4.4 et FIGURE 4.6 obtenus avec les montages REF et VCT, le CCII+ utilisé pour la réalisation de la plateforme 81 × 10 permet effectivement de piloter un memristor tout en récupérant son courant.

7. Le gain de copie en courant du composant réel OPA861 peut être légèrement différent de 1 et varier de composant à composant.

4.1.3 Mode neurone LIF à résistance de décharge

Si les circuits électroniques nécessaires pour piloter des memristors (convoyeur de courant pour piloter la tension postsynaptique et copier le courant synaptique, blocs SPK pour générer des formes d'onde variées) ont été vérifiés, la question se pose sur le bon fonctionnement de l'ensemble de la chaîne constituant un neurone postsynaptique et qui sert de brique de base pour la mise en place du réseau de neurones de la plateforme 81 × 10. Pour vérifier le bon fonctionnement de cette chaîne, une expérimentation reposant sur le montage illustré sur la FIGURE 4.7 a été réalisée. Cette expérimentation utilise le mode neurone LIF du bloc POST décrit à la partie 3.2 Cartes MIRA et VCCS : Partie calcul analogique pour réaliser une inférence en utilisant une résistance à la place d'un memristor afin de s'assurer que le neurone LIF du bloc POST est fonctionnel. Pour cela on applique une impulsion négative répétée sur la borne présynaptique de la résistance tout en observant l'évolution de la tension de membrane amplifiée ainsi que la sortie *event* du module LIF qui indique si un événement postsynaptique a eu lieu. L'inférence réalisée pour tester le neurone LIF consiste en 7 impulsions de 3.3 V large de 200 µs à une fréquence de 2 kHz.

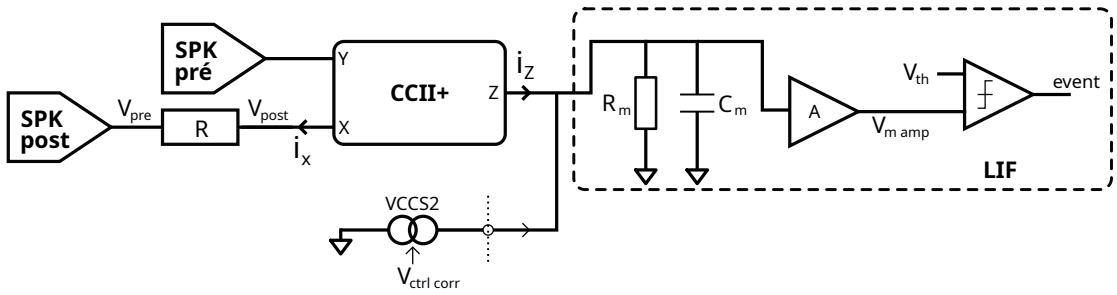


FIGURE 4.7 – Schéma simplifié de vérification du module LIF du bloc POST.
Le bloc SPK pré est utilisé pour fixer la tension V_{post} à 0 V. Le sous-bloc SPK post est utilisé pour appliquer des impulsions négatives successives pour réaliser une inférence. Une résistance $R = 100\,k\Omega$ est utilisée pour émuler une synapse memristive puisque l'on s'intéresse ici seulement à la charge du neurone LIF et non à la plasticité des synapses. Les paramètres choisis pour le module LIF d'un bloc POST sont $K = 1$(gain de copie en courant du CCII+), C_m =100 pF, R_m =10 kΩ et $A = 20$ et V_{th} =1 V (V_{th} est généré par le module GEN).

La FIGURE 4.8 présente le tracé théorique (de référence) de l'évolution de la tension de membrane amplifiée obtenue avec le simulateur décrit CHAPITRE 2 utilisant une autre méthode de calcul de la tension de membrane des neurones LIF, celle du modèle à résistance de décharge évoqué 1.2.1 Le choix des neurones : Le neurone LIF, différente de celle à courant de décharge utilisée dans le cadre du démonstrateur ULPEC et dans les simulations du CHAPITRE 2.

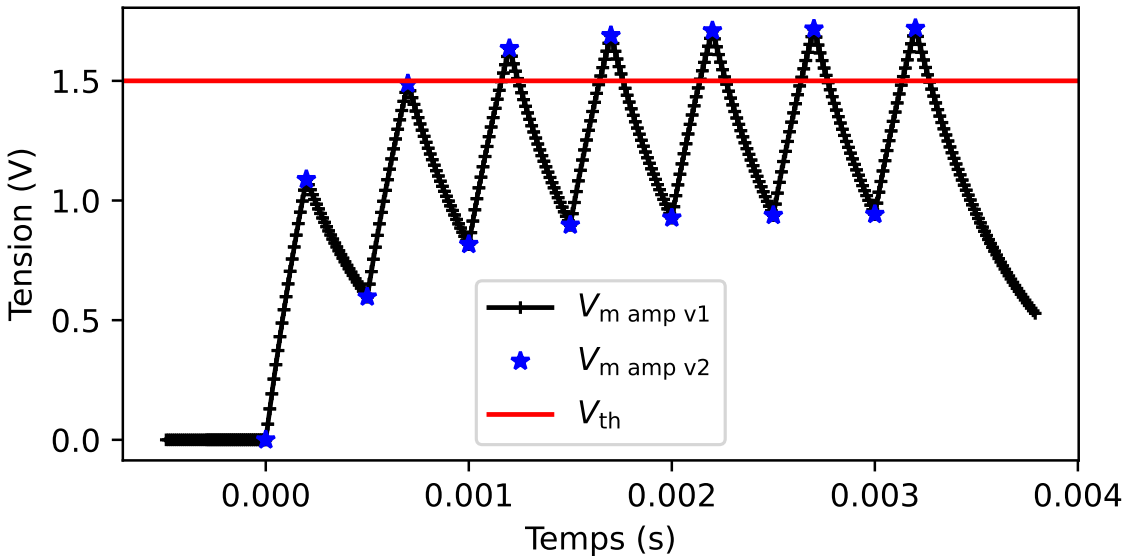


FIGURE 4.8 – Comportement théorique du mode LIF du bloc POST.
La simulation a été réalisée avec les paramètres $C_{\text{m}} = 100\,\text{pF}$, $R_{\text{m}} = 5\,\text{k}\Omega$, $K = 1$ (gain copie en courant du CCII+) et $A = 20$. L'utilisation de $R_{\text{m}} = 5\,\text{k}\Omega$ à la place de $R_{\text{m}} = 10\,\text{k}\Omega$ se justifie par la présence de la résistance de sortie de la source de courant VCCS2 ($R_{\text{output}} = 10\,\text{k}\Omega$) mise en parallèle de la membrane pour ajuster la tension de repos lors de l'expérimentation la ramenant ainsi à $R_{\text{m}} = 5\,\text{k}\Omega$. Le tracé de $V_{\text{m amp v2}}$ correspond aux points de calcul du simulateur lorsqu'on lui injecte les mêmes événements que ceux utilisés lors de l'expérimentation (7 impulsions de 3.3 V large de 200 µs à une fréquence de 2 kHz). Le tracé de $V_{\text{m amp v1}}$ correspond au calcul du simulateur lorsqu'on lui injecte des points de calcul supplémentaires à plus haute fréquence pour obtenir un tracé plus précis de l'évolution de la membrane.

Une image du montage expérimental réalisé est donnée FIGURE 4.9 et les mesures des différentes tensions d'intérêt acquises avec un oscilloscope sont illustrées sur la FIGURE 4.10. On y observe :

— La décharge de la membrane lorsque aucune activité présynaptique (impulsion négative) n'a lieu et que la tension de membrane n'est pas à son niveau de repos (0 V pour cette expérimentation). Phénomène caractéristique de la partie *Leaky* (ou L) d'un neurone LIF.

— La charge de la membrane lorsqu'une activité présynaptique (impulsion négative) a lieu. Phénomène caractéristique de la partie *Integrate* (ou I) d'un neurone LIF.

— L'émission d'un événement (passage à 0 V de la tension V_{event}) lorsque la tension de membrane amplifiée $V_{\text{m amp}}$ dépasse la tension de seuil $V_{\text{th}} = 1.5\,\text{V}$. Phénomène caractéristique de la partie *Fire* (ou F) d'un neurone LIF.

Cette expérimentation met donc en évidence que le circuit électronique réalisé pour l'implémentation d'un neurone LIF pour la plateforme 81 × 10 est valide puisque les 3 fonctions caractéristiques de ce dernier (*Leaky*, *Integrate* et *Fire*) sont assurées.

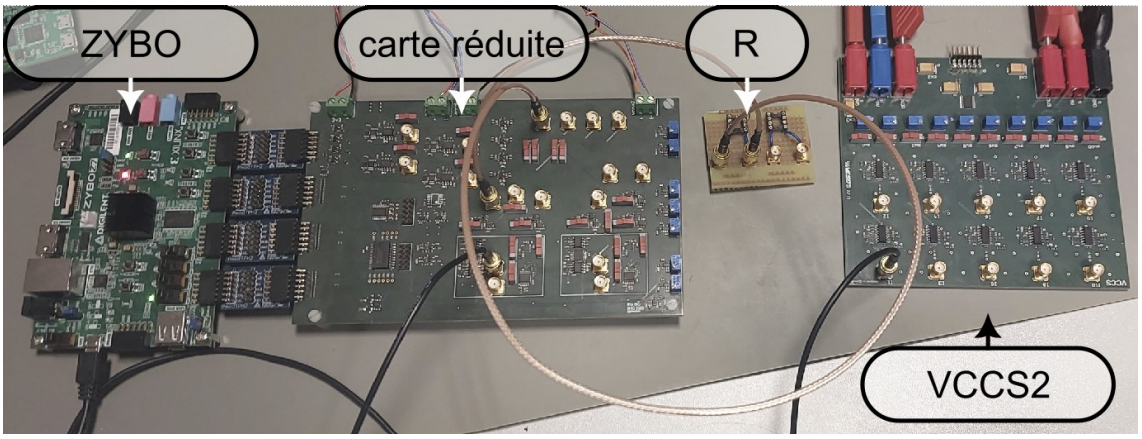


FIGURE 4.9 – Montage expérimental de test du module LIF sur carte réduite.

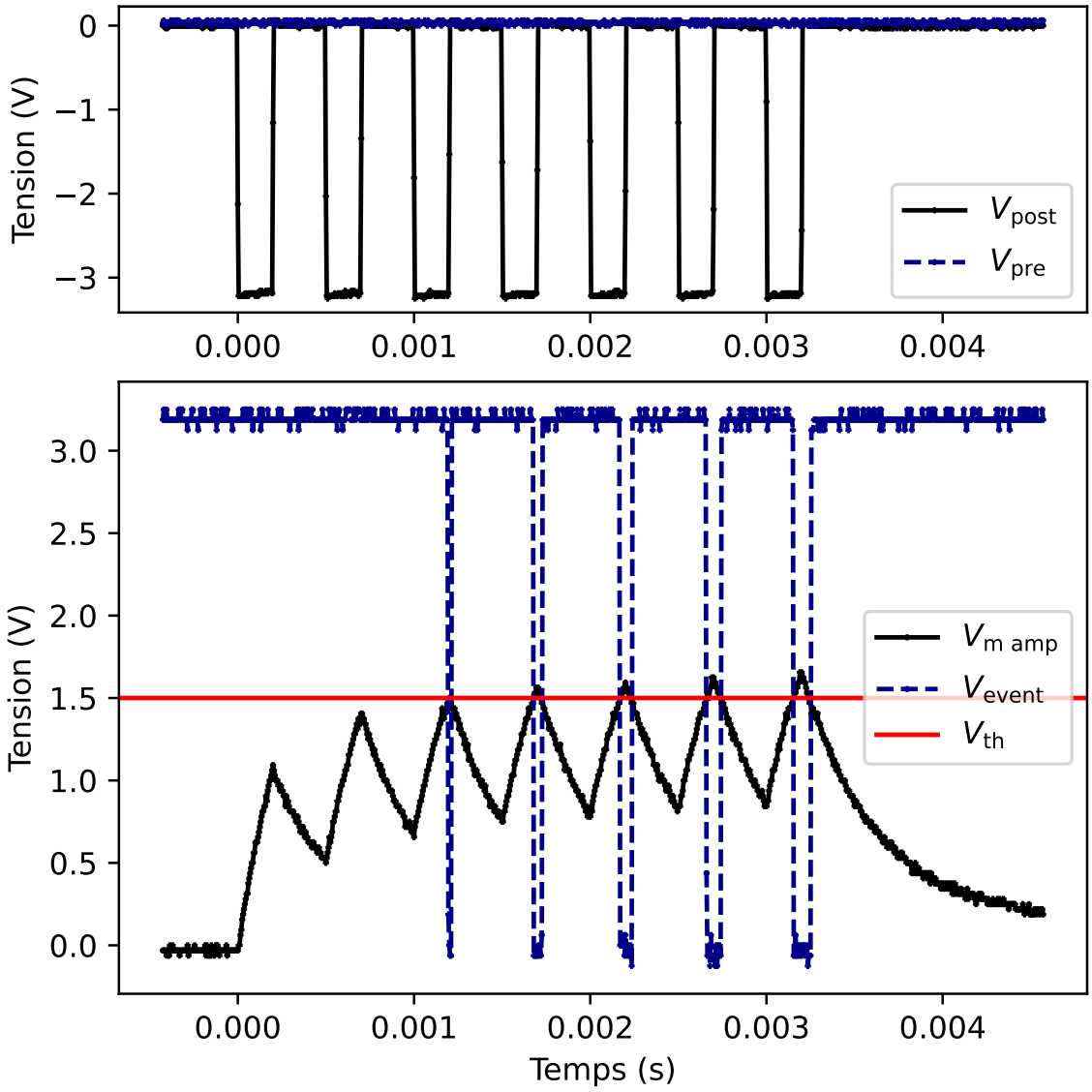


FIGURE 4.10 – Comportement expérimental du mode LIF du bloc POST.
Comportement expérimental du neurone LIF lorsqu'une résistance de 100 kΩ est stimulée par 7 impulsions de 3.3 V large de 200 µs à une fréquence de 2 kHz. Les tensions V_{pre}, V_{post}, $V_{m\ amp}$ et V_{event} ont été acquises avec un oscilloscope. À chaque fois que la tension amplifiée de la membrane dépasse la tension de seuil, un événement est émis. Le FPGA du Zynq de la carte ZCU102 a la charge de détecter les événement postsynaptiques pour stopper l'inférence (application d'impulsion) et déclencher l'apprentissage si nécessaire.

En comparant le tracé théorique FIGURE 4.8 et la mesure expérimentale FIGURE 4.10 de la tension de membrane amplifiée, on observe que le comportement est quasiment le même malgré une charge de la membrane légèrement plus rapide en simulation. La différence est due à la variabilité électronique des composants et circuits électroniques utilisés (variabilité électronique des résistances, de la capacité de membrane, des gains de copie en courant et d'amplification de la membrane). Cette expérience permet également de démontrer le bon fonctionnement du mode de déclenchement d'événement pour l'inférence qui utilise un vecteur de 10 points d'instants de déclenchement dont les valeurs sont déterminées par le code exécuté sur la partie PS. Ici, seulement 7 points mémoires de ce vecteur d'instants de déclenchement sont utilisés (sur les 10 disponibles) correspondant aux 7 impulsions utilisées pour charger la membrane. Le succès de cette inférence avec le mode neurone LIF démontre donc le fonctionnement de toute une chaîne de contrôle de la plateforme allant de la partie logicielle en C aux circuits numériques en passant par les modules VHDL (SpikeCore, CMDCore,...).

4.1.4 Mode estimation de poids synaptiques

Le mode estimation de poids synaptiques décrit dans la partie 3.2 Cartes MIRA et VCCS : Partie calcul analogique est une autre fonction possible des blocs POST qui a également été vérifiée pour s'assurer que les circuits électroniques réalisés ainsi que les modules VHDL implémentés et les algorithmes de contrôle écrits en code C fonctionnent correctement et permettent d'obtenir une estimation des poids synaptiques connectés à un bloc POST.

Pour cela on définit 2 expérimentations :

1. La première expérimentation consiste à réaliser la mesure d'une simple résistance en appliquant différentes tensions sur sa borne présynaptique et en relevant le courant obtenu avec le sous-bloc MSU LOG du bloc POST. Les valeurs obtenues numériquement sont ensuite utilisées pour estimer cette résistance connectée entre le bloc SPK présynaptique appliquant la tension de mesure et le bloc POST. L'algorithme de mesure et d'estimation de résistance décrit ANNEXE E est utilisé pour déterminer la résistance du composant testé. Cette première expérimentation a pour objectif de montrer la capacité de la plateforme à réaliser une mesure de manière embarquée, c'est à dire sans appareil de mesure externe comme cela a pu être le cas lors de la vérification du comportement du CCII+.
2. La deuxième expérimentation consiste à réaliser l'hystérésis d'un memristor de manière embarquée pour prouver que les circuits électroniques et l'architecture de contrôle définie permettent de caractériser automatiquement un memristor (réalisation autonome de l'hystérésis d'un memristor et acquisition des points de mesures).

La FIGURE 4.12 donne les résultats obtenus pour la première expérimentation lors de l'utilisation de deux résistances différentes de 99.7 kΩ et 1.012 MΩ dont les valeurs ont été mesurées avec un multimètre (Fluke 179). On remarque que pour les 2 résistances choisies, les résultats bruts obtenus sont supérieurs à la valeur de référence, mais se retrouve dans le bon ordre de grandeur malgré la présence d'une certaine variabilité suivant la tension utilisée pour réaliser la mesure. Variabilité qui est plus importante pour une résistance plus élevée c'est-à-dire des courants de mesure plus faibles. Ces différences entre les valeurs obtenues sur la carte réduite de test et la valeur de référence ont plusieurs origines :

— Les formules de calculs choisies pour extraire des valeurs de résistances dites brutes supposent des CCII+ idéaux dont le gain en courant est de 1. En pratique, ce gain en courant n'est pas parfaitement unitaire et peut varier d'un composant à l'autre. On peut en théorie s'en affranchir en appliquant un facteur multiplicatif sur les résultats obtenus.

— Les formules de calcul utilisées reposent sur l'utilisation de paramètres théoriques liés à la conversion du courant en tension du composant ADL5303.

— Les tensions utilisées pour le calcul de la résistance correspondent au code de tension choisi transmis aux DAC des blocs SPK de la carte MIRA. La tension réelle appliquée sur

la résistance de test peut être différente de quelques dizaines de millivolts à cause de la variabilité électronique des composants constituant ces blocs SPK.

Les résultats obtenus sont donc des résultats bruts sur une chaîne de mesure dont les différents éléments qui la constituent (CCII, blocs SPK, formule de conversion courant/tension) n'ont pas été caractérisés. Malgré cette absence de caractérisation, les résultats obtenus sont satisfaisants puisqu'on est capable de retrouver des valeurs du même ordre de grandeur et ils correspondent ainsi à une estimation. L'utilisation brute (sans caractérisation) de la plateforme permet donc d'estimer les valeurs de résistance et par extension des poids de memristors. Si le but *in fine* de la plateforme est de permettre de réaliser de la mesure de composants memristifs, les travaux présentés dans ce manuscrit se sont arrêtés à l'estimation puisque l'étape de caractérisation de la chaîne de mesure n'a pas encore été réalisée. En effet, même si la caractérisation est une étape importante pour pouvoir utiliser de manière pérenne la plateforme, elle requiert un temps non négligeable pour la réaliser puisque sur la carte MIRA il y a 91 blocs SPK générant des tensions qu'il faut caractériser, ainsi que 10 blocs POST (CCII+, composant ADL5303, ...). Le choix a donc été fait de mettre en suspens cette caractérisation afin de commencer et tester les premières expérimentations sur la plateforme 81 × 10.

La FIGURE 4.13 illustre le tracé d'une hystérésis obtenu avec la plateforme réduite avec le montage expérimental illustré sur la FIGURE 4.11. La tension V_{prog} de l'impulsion de programmation est répartie entre les deux bornes du composant ($V_{\text{pre}} = V_{\text{prog}}/2$ et $V_{\text{post}} = -V_{\text{prog}}/2$). On y retrouve le comportement attendu d'un memristor ferroélectrique. Au-delà de $V_{\text{prog}} = 1\text{V}$, la résistance augmente de manière non volatile avant de diminuer si elle passe sous $-1.5\,\text{V}$.

Les blocs électroniques, modules numériques et algorithmes de contrôle en C qui constituent le mode estimation de poids synaptiques de plateforme sont donc capables de relever et estimer le comportement de memristors ferroélectriques.

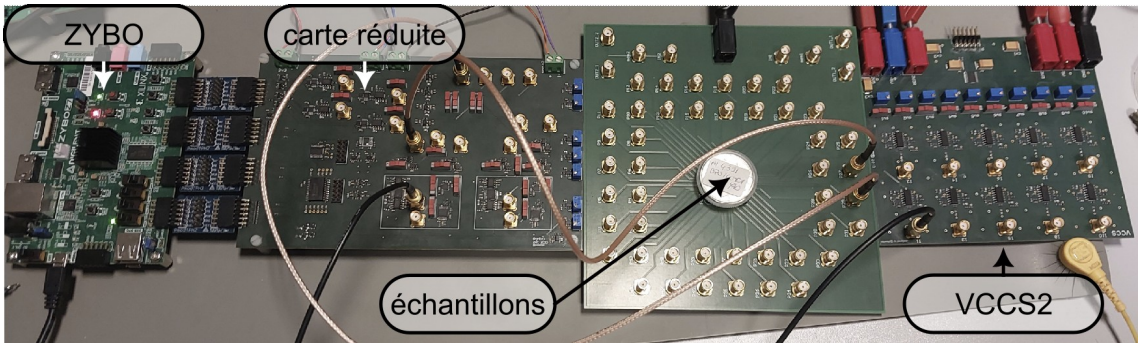


FIGURE 4.11 – Montage expérimental de test d'hystérésis d'un memristor avec carte réduite.

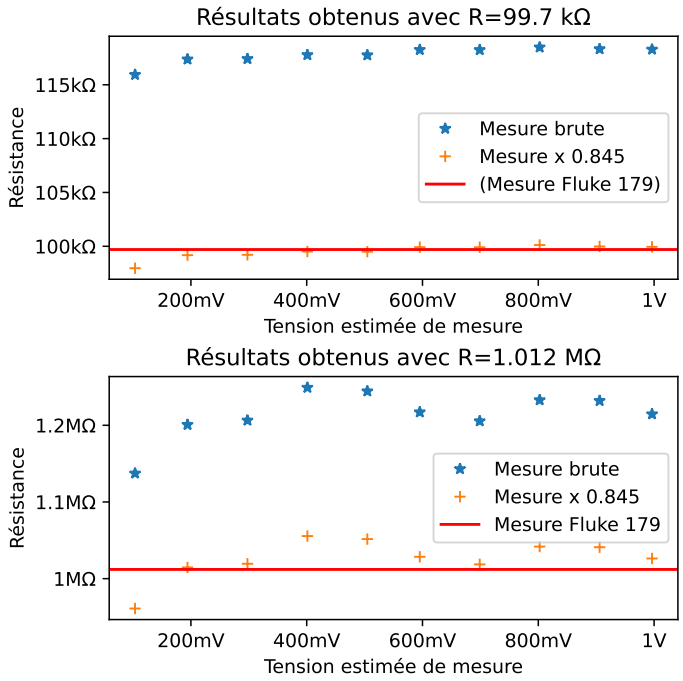


FIGURE 4.12 – Estimations de résistances de test avec la plateforme réduite tinyBIO. Les tensions estimées de mesure correspondent à la valeur sélectionnée dans le code en C de l'algorithme de mesure. Elles sont estimées puisqu'il n'y a pas de phase d'étalonnage. Les « Mesure brute » et « Mesure × 0.845 » correspondent à des valeurs obtenues par l'algorithme de mesure de la plateforme.

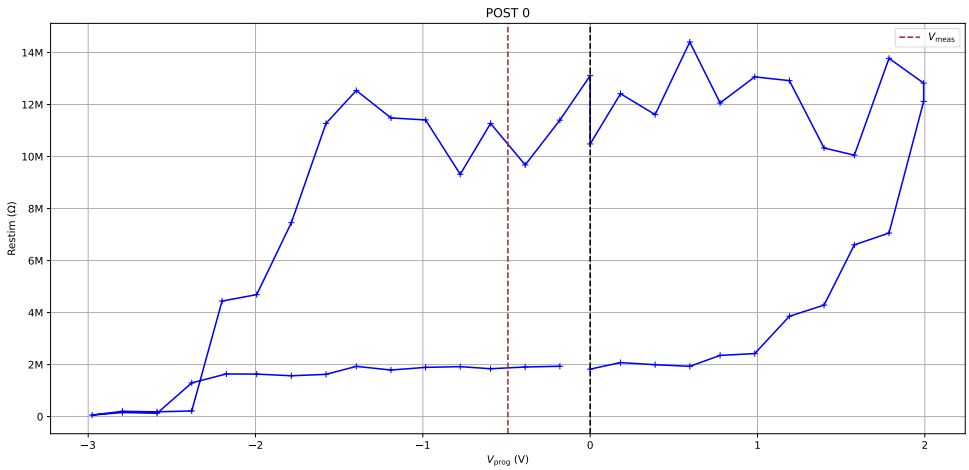


FIGURE 4.13 – Exemple d'hystérésis d'un memristor obtenue avec la plateforme réduite.

4.1.5 Synthèse

La version réduite de la plateforme a été mise en place afin de pouvoir vérifier les différents blocs et fonctions de la plateforme finale 80 × 10. L'utilisation d'un CCII+ a été vérifiée expérimentalement pour s'assurer que la plateforme est à même de piloter ces derniers. La génération de formes d'ondes grâce aux blocs SPK qui ont à charge de générer les tensions présynaptiques et postsynaptiques via des formes définies numériquement ont également été validées. Enfin, 2 des 3 modes d'utilisation des blocs POST (neurones postsynaptiques multifonctions) ont été vérifiés expérimentalement jusqu'ici :

— Le mode neurone LIF qui permet la mise en place d'un réseau de neurones événementiels a été vérifié.

— Le mode estimation de poids synaptique a également été vérifié. Il permet d'obtenir une image du comportement d'un memristor.

Le dernier mode, le mode estimation de la tension de membrane qui vise à faire l'acquisition de la tension de membrane, a été testé manuellement sur des tensions continues pour des points de tension stable de la tension de membrane. S'il n'a pas été conçu pour réaliser une acquisition en continue des tensions de membrane pour en obtenir leur tracé, il peut toutefois en théorie faire l'acquisition de la tension de membrane d'un neurone postsynaptique à la fois [8] à une fréquence d'échantillonnage de 150 kHz.

Les briques électroniques et de contrôle numérique de la plateforme ont été vérifiées expérimentalement sur carte réduite. Cette vérification a été répétée sur la plateforme 81 × 10 et une première utilisation à l'échelle d'un crossbar de memristors a été réalisée et est présentée dans la partie 4.2 Mise en œuvre de la plafeforme.

8. L'acquisition des 10 tensions de membranes est réalisée par 1 seul composant multi-canaux. On ne peut donc faire l'acquisition que d'une seule tension de membrane à la fois.

4.2 Mise en œuvre de la plafeforme

Afin d'utiliser la plateforme 81 × 10, des crossbars de 810 memristors sont nécessaires. Dans le cadre du projet européen ULPEC, les collaborateurs d'IBM Zurich en charge de la fabrication des memristors ont fourni un crossbar de test nommé LBE93 constitué de memristors ferroélectriques HZO[93]. Cette technologie de memristors qui utilise des matériaux différents de ceux des memristors choisis pour le démonstrateur du projet ULPEC (memristor BFO) présente des propriétés différentes. Différences telles que des rapports de résistance R_{ON} et R_{OFF} plus faibles, de l'ordre de 1.75 environ, d'après les mesures réalisées sur le crossbar de test fourni avec la plateforme MIRA au lieu de quelques centaines pour des memristors BFO ou encore des tensions de seuil de modification non volatile plus faibles.

La plateforme, mise au point pour piloter ce type de crossbar, a été développée pour 2 utilisations distinctes :

1. Tester l'apprentissage sur un réseau 81 × 10 et vérifier expérimentalement le comportement de règles d'apprentissage définies en simulation.
2. Extraire les paramètres physiques de composants memristifs (tension de seuils, plage de valeurs des poids).

La première utilisation (apprentissage d'un réseau 81 × 10) s'est heurtée à plusieurs difficultés pendant ces travaux de thèse. Notamment la méthode de modification des poids dans le crossbar pendant l'apprentissage. La version initiale retenue en 2 phases [9], non fonctionnelle à cause du courant trop important [10] pour les CCII+ de la plateforme, a dû être remplacée par une modification en N_{pre} phases [11]. C'est pour cela notamment que les expérimentations réalisées au moment de la rédaction de ce manuscrit n'ont pas permis d'obtenir d'apprentissage.

La deuxième utilisation (extraction de paramètres) a quant à elle permis de produire des résultats sur ce crossbar de test LBE93. Ce sont ces premiers résultats qui sont décrits dans cette partie. Il faut noter que le crossbar LBE93 qui a été fourni pour tester la plateforme présente un certain nombre de memristors déconnectés des lignes présynaptiques [12] puisqu'un certain nombre d'entre eux sont apparus comme dysfonctionnels [13] lors du procédé de fabrication.

La FIGURE 4.14 montre le montage expérimental de la plateforme avec les différentes cartes (ADAPT, MIRA, VCCS2 et ZCU102) la constituant.

9. Voir FIGURE 3.22.A.
10. La modification simultanée de memristors génère des courants bien plus importants que lorsqu'ils sont modifié 1 à 1 à cause de leur non linéarité.
11. Voir FIGURE 3.22.B.
12. Cela se traduit sur le masque de fabrication du crossbar par la suppression des connexions de ces memristors aux lignes métaliques présynaptiques.
13. Laissant passer beaucoup de courant et ne présentant pas de comportement memristif

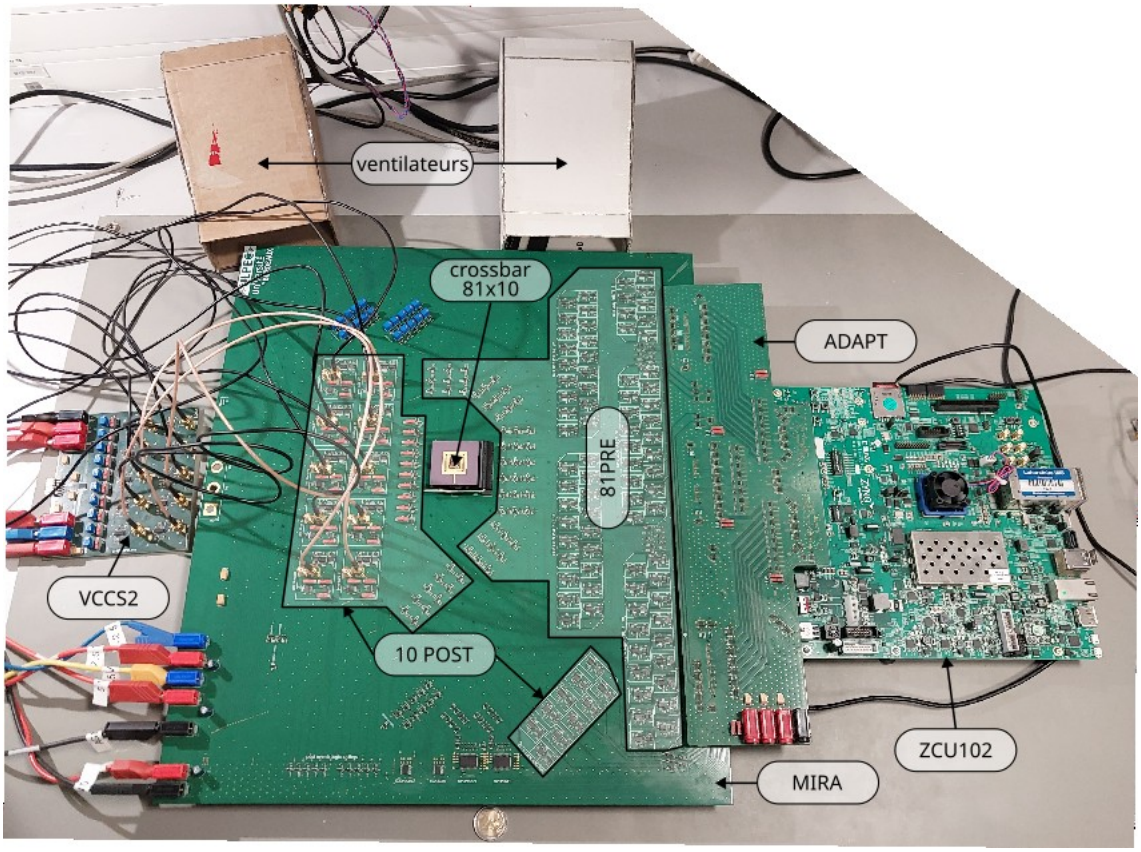


FIGURE 4.14 – Plateforme 81 × 10.
Les deux ventilateurs sont utilisés pour refroidir la carte MIRA.

4.2.1 Hystérésis à l'échelle d'un crossbar

Afin d'extraire des informations sur les propriétés des memristors, il est nécessaire de définir une expérimentation à réaliser. Il existe plusieurs types d'expérimentations possibles pour obtenir des informations sur le comportement des composants memristifs que l'on souhaite utiliser. Il y a par exemple la réalisation de courbes I-V, le tracé de courbes d'hystérésis ou encore la réalisation de STDP. Si cette plateforme est en théorie capable de réaliser toutes ces expérimentations, celle qui a été choisie en premier et réalisée avec succès durant ces travaux de thèse (et qui a pu être vérifié sur la plateforme réduite en premier lieu) est le tracé de courbes d'hystérésis.

Lorsque l'on réalise une hystérésis sur un memristor comme cela a été le cas sur la plateforme réduite, la question des tensions à appliquer sur toutes les lignes présynaptiques et postsynaptiques ne se pose pas. Dans le cas de la réalisation d'hystérésis sur crossbar 81 × 10 avec la plateforme, le choix a été fait de réaliser ces hystérésis ligne présynaptique par ligne présynaptique, soit 10 hystérésis de memristors simultanément afin de limiter les courants gérés par les blocs POST. Le protocole suivant a été utilisé pour l'obtention d'un point de l'hystérésis :

1. Mise à 0 V de tous les SPK présynaptiques et SPK postsynaptiques.
2. Configurations d'impulsions de programmation d'amplitude $V_{\text{prog}}/2$ et de largeur T_{pulse} sur la ligne présynaptique d'intérêt et d'amplitude $-V_{\text{prog}}/2$ et de largeur T_{pulse} sur toutes les autres lignes présynaptiques ainsi que les blocs POST.
3. Configuration du même instant de déclenchement pour toutes les impulsions présynaptiques et postsynaptiques.
4. Déclenchement de N_{pulses} impulsions de programmation successives.
5. Mesure des memristors de la ligne présynaptique d'intérêt suivant le protocole donné ANNEXE E.

Ce protocole permet de réduire au maximum les courants issus des blocs SPK ou POST puisque seuls les memristors de la ligne d'intérêt verront une tension non nulle. Ainsi, en théorie, seul 1 memristor fait circuler du courant dans le CCII+ d'un bloc POST.

En pratique, il y a sur la carte ADAPT 4 registres à décalage utilisés pour les canaux SPK controls 1 qui sont non fonctionnels. Ainsi, 8 blocs SPK présynaptiques ne reçoivent pas correctement leurs signaux de contrôle SYNC et DATA. Ces blocs 8 SPK présynaptiques ont été modifiés temporairement pour appliquer en permanence 0 V.

Des cycles d'hystérésis ont été réalisés sur le crossbar de test LBE93 en utilisant ce protocole. La FIGURE 4.15 illustre les hystérésis obtenues pour la ligne 7 du crossbar où tous les memristors sont connectés. La figure FIGURE 4.16 illustre celle de la ligne 11 où 2 memristors ne sont pas physiquement connectés (représenté par le fond gris des tracés correspondants). L'absence de

points dans les tracés des memristors non connectés est causée par l'algorithme de mesure et d'extraction des valeurs de résistances qui ne trace pas de points si ces points correspondent à des mesures de courants rejetés [14] lors du traitement des données obtenues par la plateforme. On remarque que les memristors non connectés du crossbar ont donc bien été « repérés » par la plateforme puisque tous les points de mesures ont été rejetés par l'algorithme de mesure, ce qui est cohérent puisque ces derniers ne sont pas connectés. Pour ces mesures, le circuit VCCS2 présente une résistance de sortie plus faible (1 kΩ au lieu de 10 kΩ précédemment). Elle a été ajustée lors de cette expérimentation pour permettre une plus large gamme de correction des courants en sortie des CCII+.

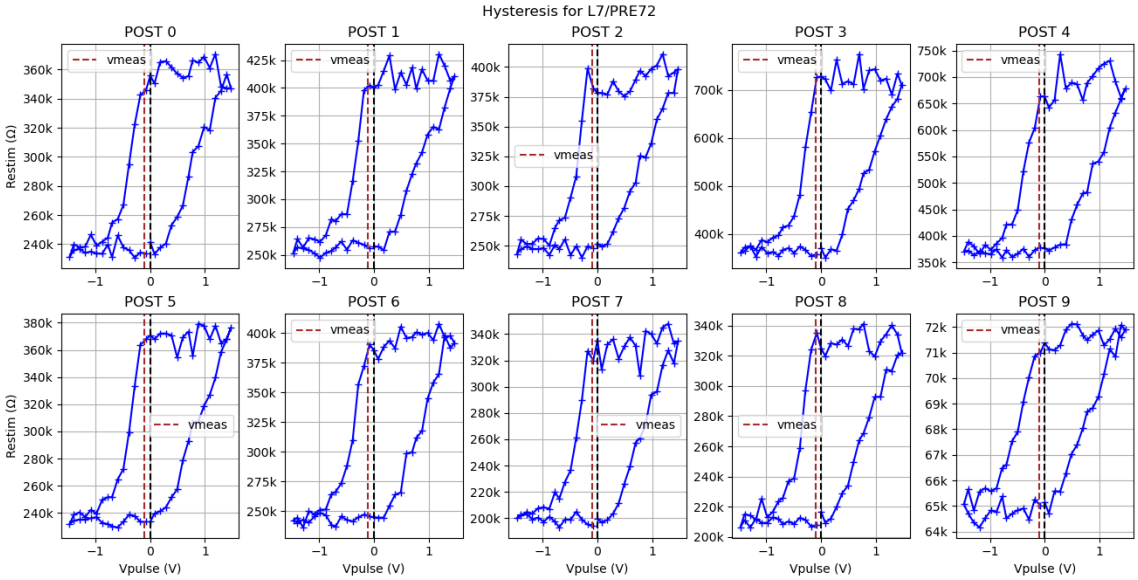


FIGURE 4.15 – Hystérésis des memristors de la ligne 7 du crossbar LBE93.
La tension de mesure sélectionnée par la plateforme est de 0.1 V. La largeur des impulsions T_{pulse} est de 200 µs. Le nombre d'impulsions de programmation pour 1 point est $N_{\text{pulse}} = 1$.

14. Décrit ANNEXE E.

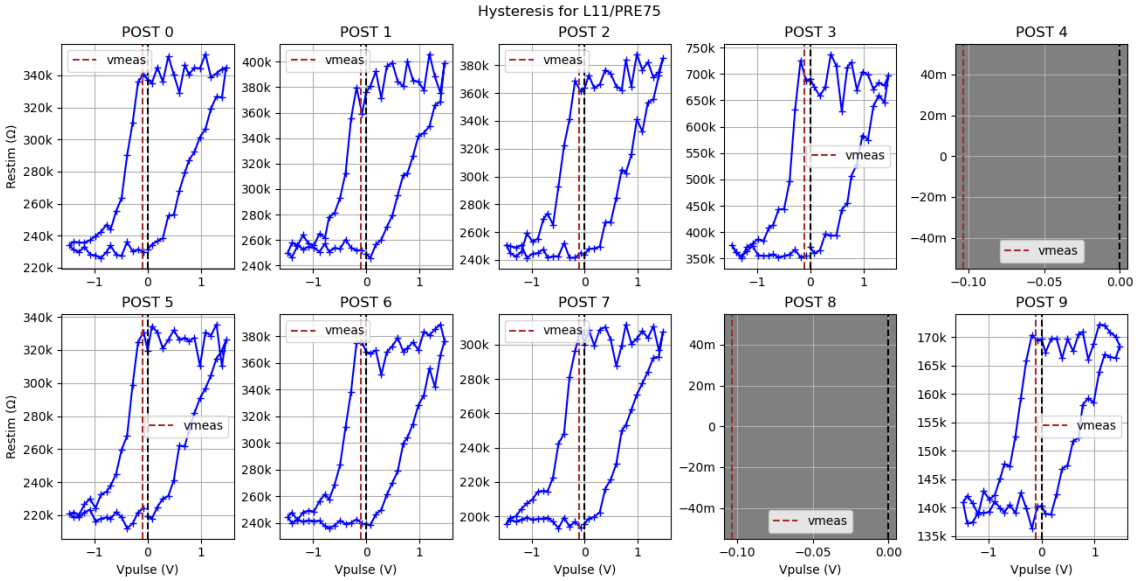


FIGURE 4.16 – Hystérésis des memristors de la ligne 11 du crossbar LBE93.
La tension de mesure sélectionnée par la plateforme est de 0.1 V. La largeur des impulsions T_{pulse} est de 200 µs. Le nombre d'impulsions de programmation pour 1 point est $N_{\text{pulse}} = 1$.

4.2.2 Extraction automatisée de paramètres

La plateforme permet d'obtenir automatiquement les tracés d'hystérésis des memristors du crossbar qui lui est connecté même si cela peut prendre jusqu'à plusieurs jours suivant le nombre de points et de cycles souhaités. À partir de ces hystérésis, il est possible d'extraire automatiquement 4 paramètres pour chaque memristor :

— R_{ON} : Le niveau le plus bas de résistance obtenu avec la plateforme. Il correspond au LRS (*Low Resistance State*) s'il s'agit du niveau le plus bas que peut présenter le memristor.

— R_{OFF} : Le niveau le plus haut de résistance obtenu avec la plateforme. Il correspond au HRS (*High Resistance State*) s'il s'agit du niveau le plus haut que peut présenter le memristor.

— $V_{\text{th n}}$: Le seuil de modification négatif du memristor permettant de diminuer de manière non volatile sa résistance.

— $V_{\text{th p}}$: Le seuil de modification positif du memristor permettant d'augmenter de manière non volatile sa résistance.

Il faut noter que sur ces 4 paramètres 2 d'entre eux (R_{ON} et R_{OFF}) dépendent de la tension de mesure choisie à cause du comportement non linéaire des memristors ferroélectriques. Ainsi si ces derniers ont vocation à être utilisés pour définir des paramètres pour de l'apprentissage d'un réseau, il faut que la tension de mesure utilisée pour le tracé des hystérésis corresponde à l'amplitude des impulsions choisie pour l'inférence.

Afin d'extraire automatiquement ces paramètres, une méthode en 6 étapes est suivie pour toutes les hystérésis obtenues :

Étape 1 : Déterminer si les données sont traitables.
Les hystérésis obtenues après expérimentation et extraction des résistances mesurées peuvent résulter en une absence partielle [15] ou totale de points dans le tracé si ces derniers sont considérés comme invalides. Le choix a été fait de ne traiter que les données d'hystérésis comprenant au moins 90 % de valeurs valides (sur le nombre total de points de l'hystérésis réalisée pour un cycle donné).

Étape 2 : Normalisation des données.
Afin de simplifier le traitement des données, toute hystérésis traitée est normalisée par rapport au premier point de son tracé correspondant au point de départ de l'hystérésis. Toutes les opérations de calcul sont ensuite réalisées sur le tracé normalisé.

15. Par exemple si certaines valeurs de résistance sont trop fortes pour être mesurées correctement. Cela peut être le cas pour les valeurs R_{OFF} de memristors si elles s'approchent de plusieurs centaines de MΩ.

Étape 3 : Filtrage des tracés d'hystérésis.

Les tracés d'hystérésis obtenus peuvent présenter des variations brutales. Afin de simplifier les opérations suivantes, une première étape de filtrage est réalisée pour lisser le tracé. La FIGURE 4.17 illustre pour les hystérésis d'une ligne du crossbar le résultat de cette étape de filtrage.

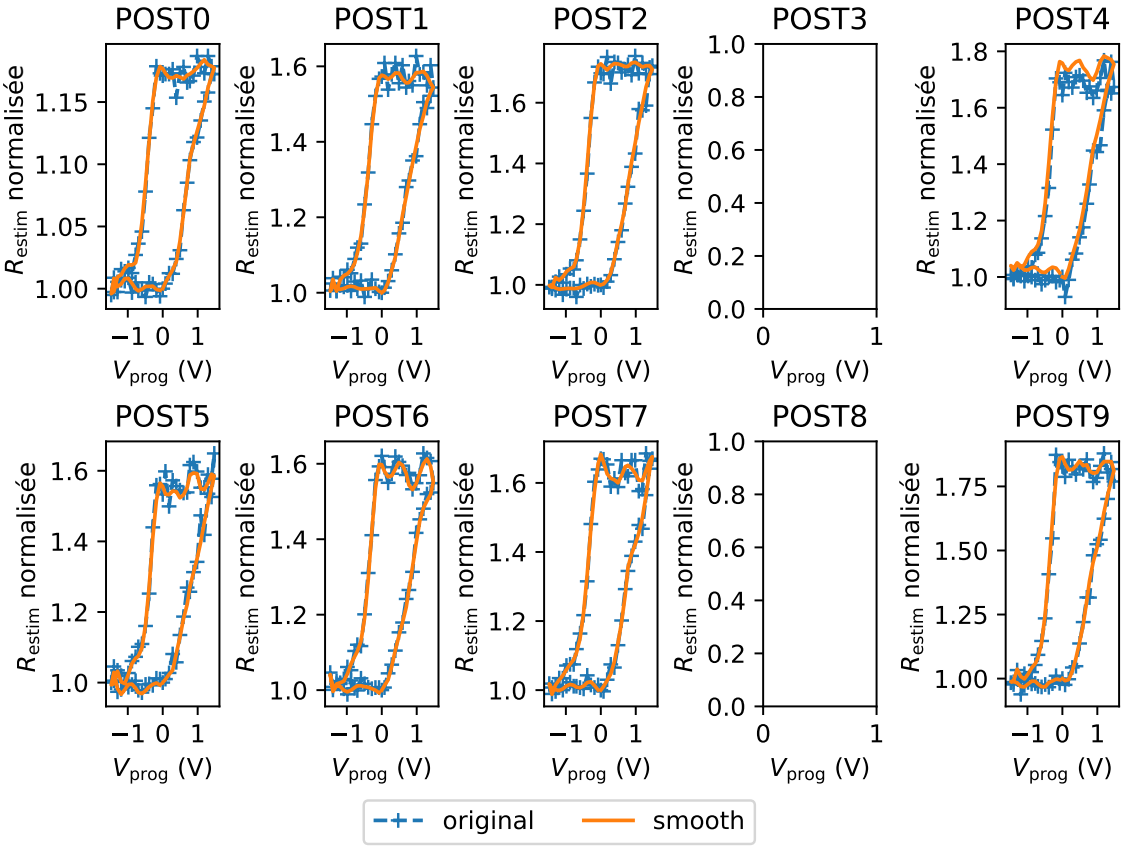


FIGURE 4.17 – Tracés d'hystérésis d'origine et tracés lissés.
Un filtre de Savitzky-Golay (choix arbitraire) a été appliqué en Python sur les données. Il s'agit de la fonction *signal.savgol_filter(data, 21, 10)* de scipy. Les paramètres du filtre ont été choisis empiriquement afin de donner un résultat satisfaisant lors des étapes suivantes.

Étape 4 : Isolation des deux parties de l'hystérésis par découpage.

Afin de traiter séparément la partie montée et descente de l'hystérésis, on découpe le tracé en 2 parties. Ce découpage d'une hystérésis se réalise suivant la méthode suivante :

1. Calcul de la dérivée du tracé lissé de l'hystérésis.
2. Détermination de *pos_max*, la position où la dérivée est maximale (doit correspondre au milieu de la pente de montée de l'hystérésis en théorie).
3. Détermination de *pos_min*, où la dérivée est minimale (doit correspondre au milieu de la pente de descente de l'hystérésis en théorie).

4. Détermination de la position $pos\_mid = \frac{pos\_max + pos\_min}{2}$ pour déterminer une position de découpage de l'hystérésis au niveau de R_{OFF}.
5. Détermination d'une position $pos\_start = \frac{pos\_max + N_{pt}}{2}$ (N_{pt} : nombre de points du tracé d'hystérésis) de découpage de l'hystérésis au niveau de R_{ON}.

On peut ainsi créer deux tracés qui isolent les deux parties de l'hystérésis :

— *data_partie1* : Contiens les points allant de *pos_start* jusqu'à *pos_mid*.

— *data_partie2* : Contiens les points allant de *pos_mid* jusqu'à la fin du tracé.

Le résultat de cette étape de découpage est illustré sur les figure FIGURE 4.18 et FIGURE 4.19.

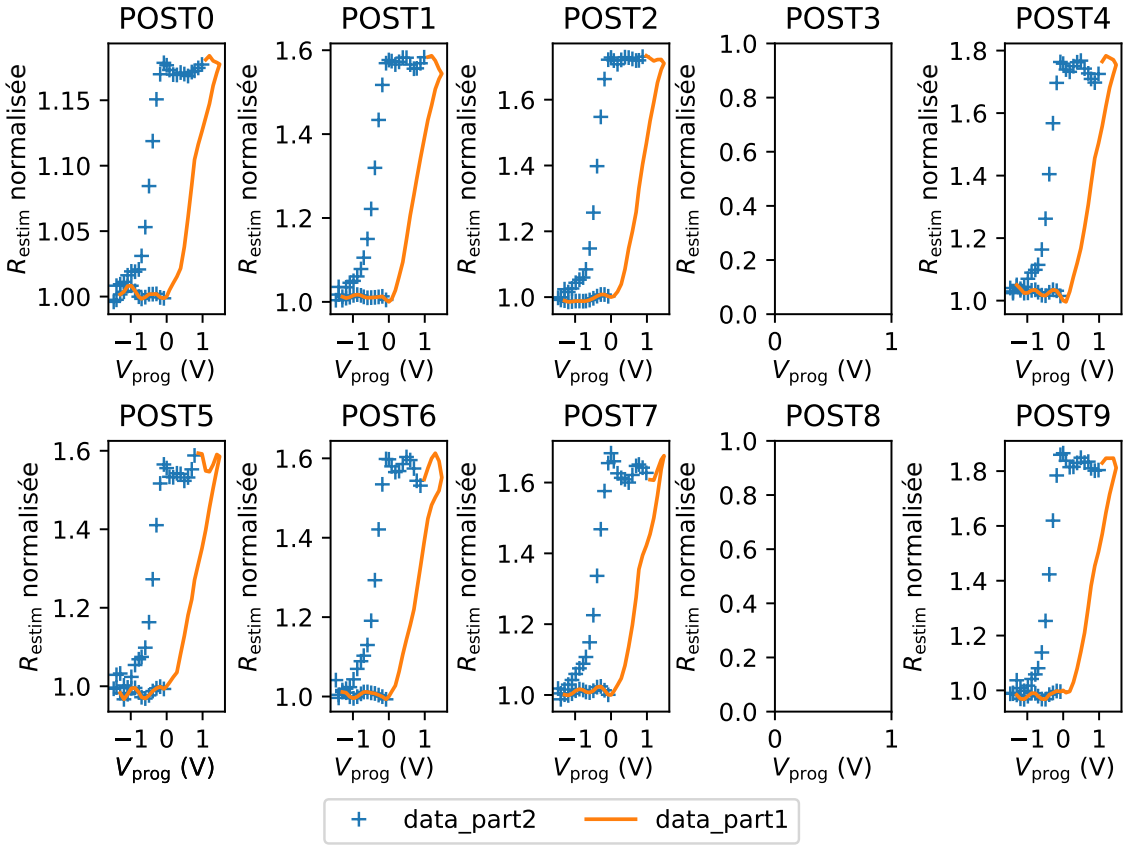


FIGURE 4.18 – Tracés d'hystérésis découpées en deux sections.
Les données traitées sont les tracés lissés de la FIGURE 4.17 (étape 3).

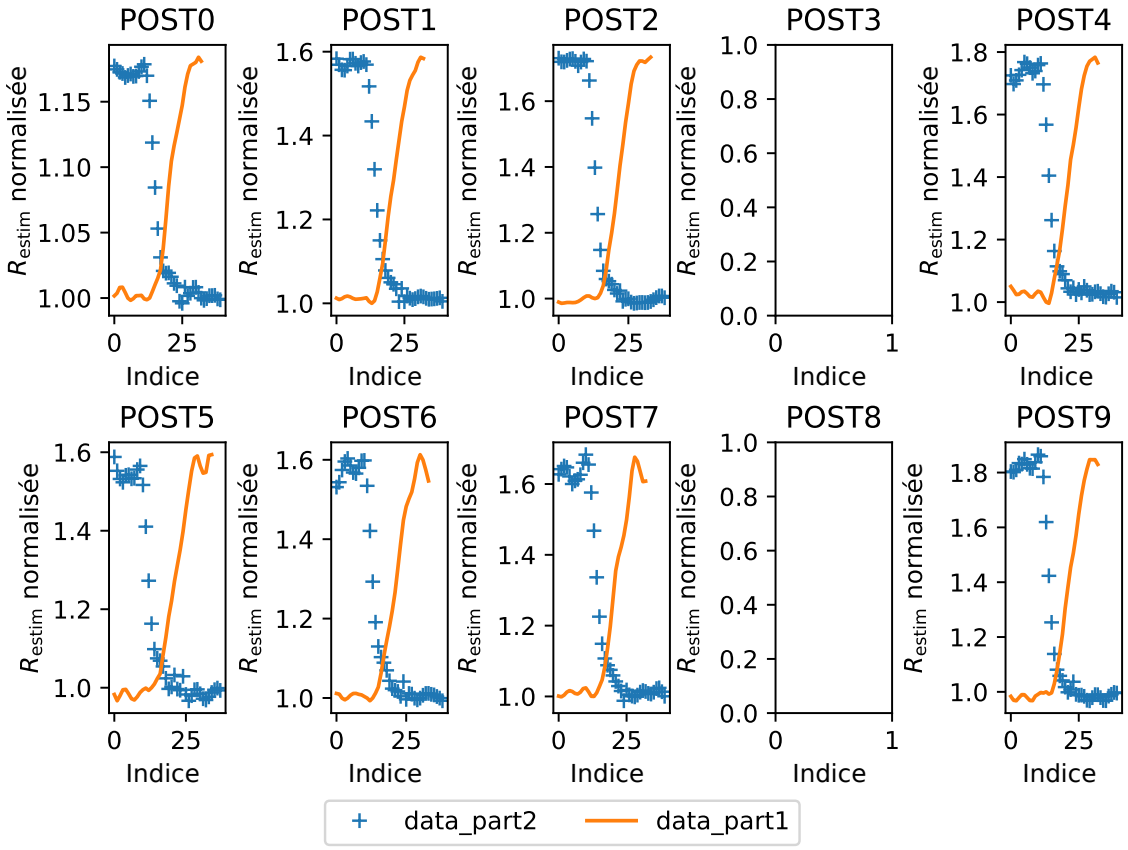


FIGURE 4.19 – Tracés d'hystérésis ordonnées et découpées en deux sections.

Étape 5 : *Curve fitting* de sigmoïde sur les deux parties de l'hystérésis.

À partir des tracés découpés, on réalise du *curve fitting*[16] sur des fonctions de type sigmoïde. Ce seront ces fonctions sigmoïdes ajustées qui seront utilisées pour déterminer les paramètres des memristors. On utilise deux fonctions données équation (4.1) pour *data_part1* et équation (4.2) pour *data_part2* avec a, b, G et E les paramètres à ajuster. Les sigmoïdes obtenues sont illustrées FIGURE 4.20.

$$Sp1(x,(a,b,G,E)) = \frac{G}{1+e^{-a(x-b)}} + E \tag{4.1}$$

$$Sp2(x,(a,b,G,E)) = \frac{G}{1+e^{a(x-b)}} + E \tag{4.2}$$

16. Ajustement de courbe.

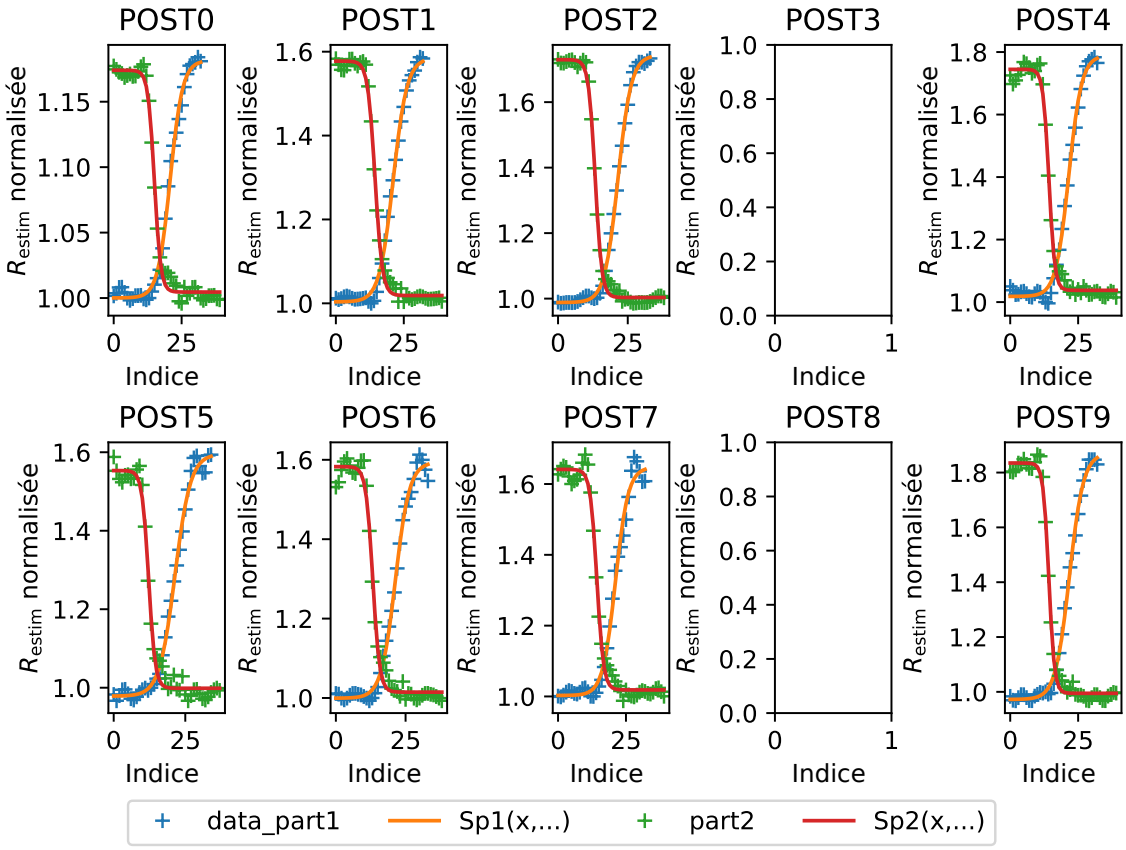


FIGURE 4.20 – Tracés d'hystérésis découpées et sigmoïdes ajustées.

Étape 6 : Détermination des paramètres R_{ON}, R_{OFF}, $V_{th\ n}$ et $V_{th\ p}$.
Pour déterminer R_{ON}, on calcule la valeur de *Sp1(0)* multipliée par la première valeur de l'hystérésis pour s'affranchir de la normalisation.

Pour déterminer R_{OFF}, on calcule la valeur de *Sp2(0)* multipliée par la première valeur de l'hystérésis pour s'affranchir de la normalisation.

Pour déterminer $V_{th\ p}$, on détermine le moment où *Sp1(x)* augmente de 10 % (choix arbitraire). Cela correspond à l'instant ou le memristor commence à varier de manière non volatile.

Pour déterminer $V_{th\ n}$, on détermine le moment où *Sp2(x)* diminue de 10 % (choix arbitraire). Cela correspond à l'instant ou le memristor commence à varier de manière non volatile.

Étape 7 (finale) : Extraction de paramètres à l'échelle du crossbar.
En suivant ces 6 étapes pour toutes les hystérésis obtenues avec le crossbar, on obtient les caractéristiques des memristors dont les données sont considérées comme traitables. La FIGURE

4.21 illustre le résultat de l'étape 5 pour tous les memristors du crossbar par neurone postsynaptique. Le neurone POST8 semble présenter un nombre faible de memristors fonctionnels d'après les expérimentations. La cause peut être due soit à une colonne du crossbar présentant des défaillances en nombre, soit une défaillance du bloc POST8 sur la carte MIRA. Pour le reste, on note que les étapes 1 à 6 résultent en des sigmoïdes qui suivent correctement les données filtrées de l'hystérésis et permettent donc d'extraire efficacement et en masse les paramètres des memristors du crossbar.

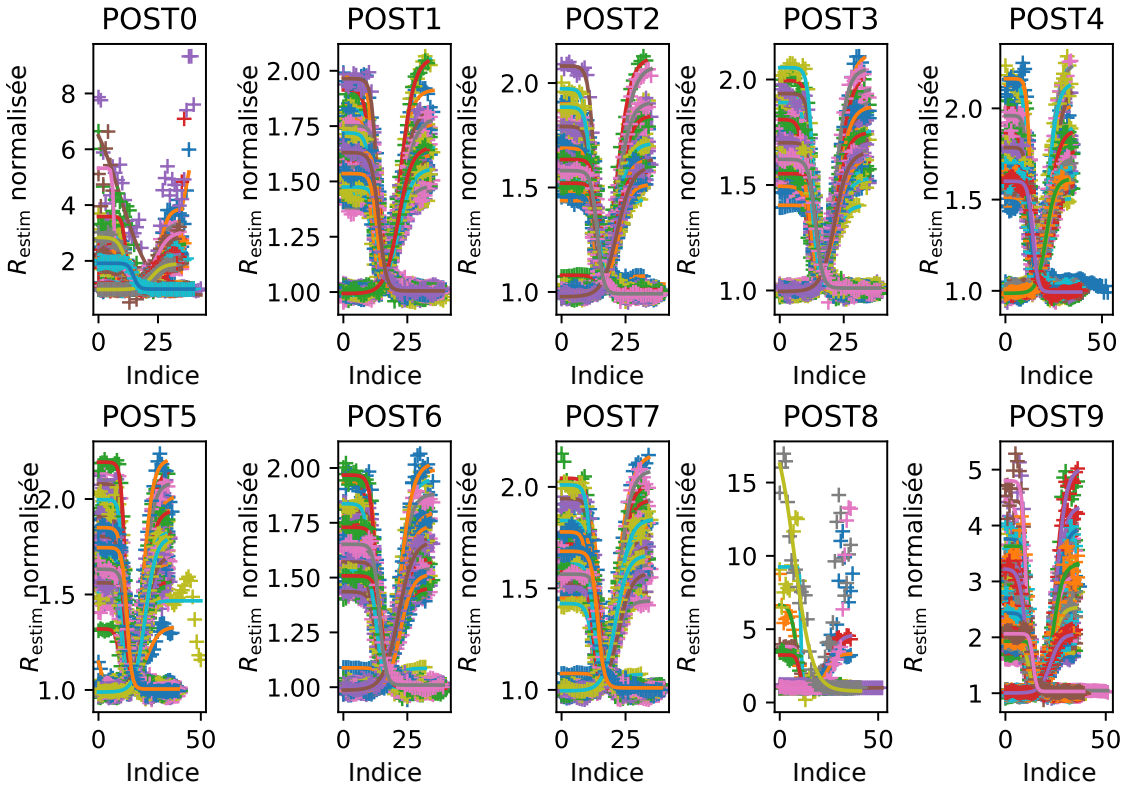


FIGURE 4.21 – Tracés d'hystérésis découpées et sigmoïdes ajustées pour l'ensemble des memristors du crossbar 81 × 10 LBE93.

En appliquant ensuite l'étape 7 on peut par exemple tracer un histogramme des tensions de seuils, comme illustré sur la FIGURE 4.22, ainsi qu'un histogramme des R_{ON} et R_{OFF}, comme illustré sur la FIGURE 4.23. Ces deux histogrammes représentent les paramètres extraits grâce à la plateforme à l'échelle du crossbar 81 × 10 LBE93. On est ainsi capable d'extraire à l'échelle d'un crossbar les paramètres de memristors automatiquement en (1) lançant l'expérimentation de tracé d'hystérésis sur le crossbar puis (2) en lançant les routines d'extraction en Python 3 de ces paramètres sur les résultats obtenus par la plateforme.

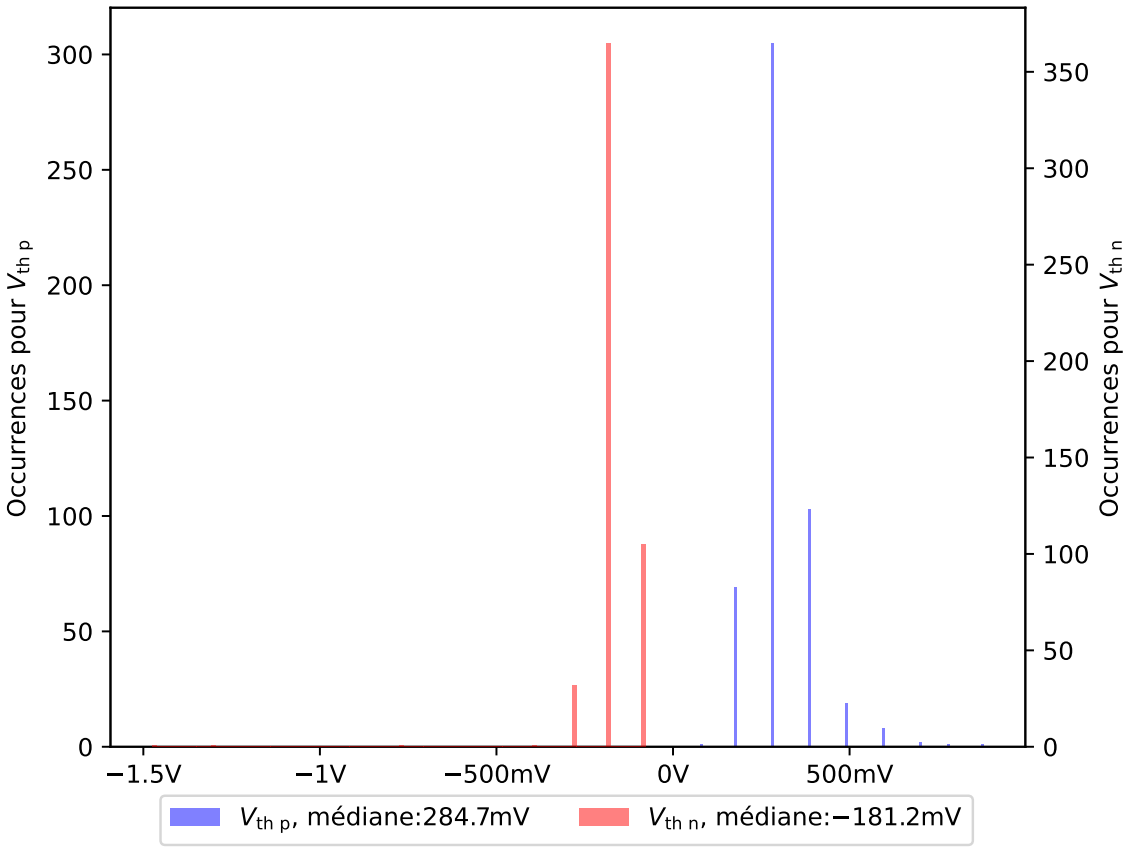


FIGURE 4.22 – Histogramme des tensions de seuil des memristors du crossbar LBE93.

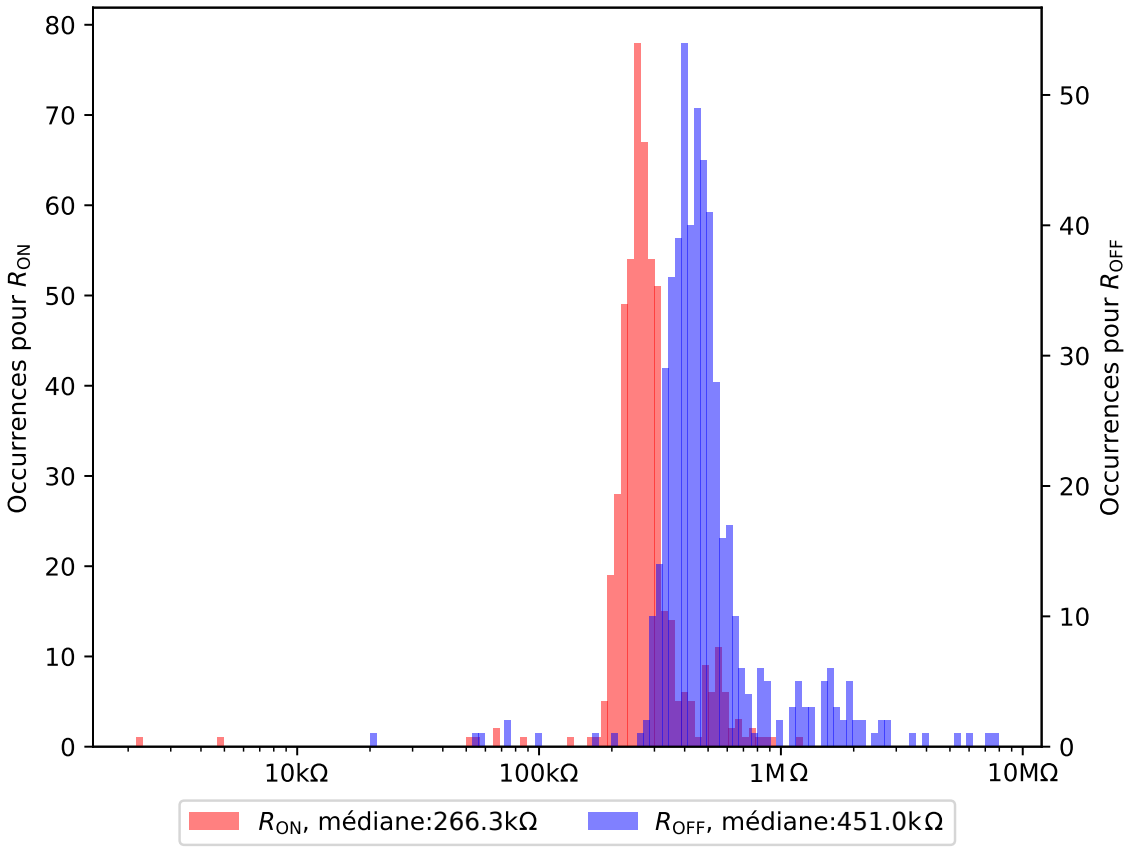


FIGURE 4.23 – Histogramme des niveaux de résistance R_{ON} et R_{OFF} des memristors du crossbar LBE93.

4.3 Conclusion

Les vérifications expérimentales décrites dans ce CHAPITRE 4 mettent en évidence le bon fonctionnement des blocs électroniques et de toute une chaîne de contrôle et pilotage de la plateforme.

La réalisation d'hystérésis embarquée vérifiée sur plateforme réduite partie 4.1.2 Preuve de fonctionnement de l'étage d'entrée postsynaptique (CCII+) puis déployée sur la plateforme 81 × 10 partie 4.2 Mise en œuvre de la plafeforme a permis de concrétiser l'une des deux raisons d'être de cette plateforme : la caractérisation et l'extraction de paramètres à l'échelle de crossbar de 810 memristors ferroélectriques.

En effet, les premiers résultats de cette plateforme ont permis d'estimer le comportement des composants d'un crossbar de memristors HZO et d'en extraire certains paramètres par analyse des résultats. S'il reste à caractériser entièrement la chaîne de mesure pour passer d'une estimation de la valeur des memristors à une caractérisation de ces derniers, ces premières expérimentations prouvent la capacité de la plateforme à caractériser en masse et automatiquement des memristors (810 composants memristifs sous forme de crossbar).

La deuxième raison d'être de cette plateforme quant à elle n'a pas encore été accomplie. Il s'agit de réaliser de l'apprentissage sur un réseau de neurones événementiels 81 × 10. Des tentatives ont été réalisées lors de ces travaux de thèse, mais elles n'ont pas été fructueuses. En effet ces essais se sont heurtés à 2 problèmes majeurs :

1. Le premier qui a été évoqué dans le CHAPITRE 3 concerne la méthode initialement envisagée de modification des poids synaptiques en 2 phases, méthode similaire à celle qui a pu être implémentée dans le démonstrateur ULPEC. À cause du comportement non linéaire des memristors ferroélectriques une modification simultanée d'un grand nombre de poids synaptiques comme c'est le cas avec une modification en 2 phases cause pour la plateforme des défaillances de copie en tension des CCII+ des neurones postsynaptiques en raison de courants trop importants. Pour pallier ce problème, une nouvelle méthode en $N_{\text{pre}} = 81$ phases est en préparation et consiste à modifier successivement les poids synaptiques pour limiter la modification à 1 seul composant par colonne du crossbar. Si cette méthode est plus lente, elle permet de réduire les courants nécessaires qui étaient la cause de défaillance pour la méthode de modification en 2 phases.
2. Le deuxième problème qui n'a pas été évoqué dans ce chapitre est lié aux propriétés des composants du crossbar de test LBE93 utilisé à ce jour. Tout d'abord il présente un certain nombre de memristors non connectés. Ensuite les memristors qui ont été testés semblent présenter des plages de fonctionnement et des propriétés largement différentes de celle utilisée dans les simulations réalisées jusqu'à présent. Pour rappel , les si-

mulations présentées dans le cadre du CHAPITRE 2 considéraient par exemple l'usage de memristors présentant des rapports de l'ordre de 100. La réalisation de simulations avec des synapses memristives présentant des propriétés similaires à celles des memristors caractérisés expérimentalement dans ce chapitre serait donc judicieuse pour évaluer si la plateforme expérimentale a des chances raisonnables de pouvoir réaliser un apprentissage avec ces derniers.

Conclusion et perspectives

Ce manuscrit a exposé les travaux réalisés lors de cette thèse qui ont eu lieu dans le cadre du projet de recherche européen ULPEC. Les difficultés majeures associées à ce travail de thèse sont la pluridisciplinarité des tâches réalisées et la nécessité de faire appel à des compétences variées (programmation en C et Python, numérique en VHDL, analogique pour la conception de cartes) nécessaires à leur accomplissement et qui sont liées au déroulement sur deux axes de travail distincts :

— Le développement d'un simulateur pour réseaux de neurones événementiels et prospection d'architectures et de règles d'apprentissages compatibles avec une implantation matérielle.

— Le développement d'une plateforme expérimentale pour : (1) la caractérisation de memristors ferroélectriques contenus dans des crossbars et utilisés comme synapses artificielles et (2) la démonstration expérimentale des règles d'apprentissage définies en simulation.

Aujourd'hui, le déploiement en nombre d'objets connectés (IoT, systèmes connectés, capteurs en tout genre) pour réaliser des tâches de plus en plus complexes rend de plus en plus critiques les questions de consommation énergétique et d'optimisation des architectures de calcul. L'une des solutions envisagées, les réseaux de neurones et plus généralement l'intelligence artificielle, est ainsi de plus en plus attrayante. Ainsi, ces travaux et ce projet ULPEC, projet qui a eu pour finalité la mise en place d'un capteur neuromorphique réalisé par la combinaison de diverses technologies (DVS, neurones événementiels en silicium, memristors ferroélectriques), s'inscrivent dans cette volonté de mettre au point de nouveaux types de capteurs à la fois intelligents et optimisés en termes de consommation énergétique. Une volonté qui, d'un point de vue global, s'inscrit dans cette tendance qui amène de l'industrie 4.0 à l'industrie de demain : l'industrie 5.0 [117]. L'industrie 4.0 depuis 2011 a amené ce déploiement des IoT en masse générant de plus en plus de données avec la volonté d'automatiser les processus au maximum avec un traitement par réseaux de neurones pour traiter ces quantités massives d'informations [118]. L'industrie 5.0, l'étape suivante dont l'Union européenne commence à esquisser les contours [119][120][121], se veut plus efficace, plus verte énergétiquement et avec une collaboration accrue de l'humain avec des machines de plus en plus intelligentes.

Dans cette tendance globale amenant à cette nouvelle étape dans les interactions homme-machine, le développement de capteurs intelligents à basse consommation est donc primordial. On peut ainsi comprendre la volonté maintenue sans relâche à travers cette thèse de rester simple pour trouver des architectures de réseaux de neurones basse consommation qui fonctionnent certes à petite échelle dans ces travaux (une couche *all-to-all*), mais qui combinent des technologies innovantes (SNN, DVS, synapses memristives) considérées comme d'excellents candidats pour ces capteurs intelligents à basse consommation puisque soit leur fonctionnement (pour les memristors ferroélectriques) se veut peu énergivore, soit leur source d'inspiration, le vivant (pour les SNN et la DVS), est une démonstration d'un système performant tout en consommant peu d'énergie.

Si des travaux s'intéressent à la conversion de réseaux de neurones plus conventionnels existants (CNN par exemple) à des versions événementielles [122], les travaux décrits dans ce manuscrit se sont inscrits à contre-courant en se penchant sur le développement d'outils pour l'étude d'architectures événementielles conçues pour un capteur donné afin de le rendre intelligent et non converties d'une architecture existante (CNN, ANN,...) en une architecture événementielle.

Si à l'instant où ces pages sont écrites, le projet de recherche européen ULPEC est terminé, le démonstrateur de ce projet (le capteur neuromorphique complet) devrait être testé dans un futur proche au sein de l'équipe de recherche du laboratoire IMS où s'est déroulée cette thèse. L'architecture du réseau de neurones qui intègre ce capteur a été définie à partir des travaux détaillés dans le CHAPITRE 2. Chapitre qui décrit le premier axe de travail de cette thèse durant laquelle de nouvelles règles d'apprentissage pour des réseaux de neurones événementiels matériels ont été établies en prenant en compte les diverses contraintes et problèmes afférents tels que la fréquence des données à traiter ou les dimensions de certains paramètres en circuit intégré (capacité de membrane). Ces nouvelles règles et cette architecture du démonstrateur ont été définies en respectant l'idée cardinale que l'utilisation de structures simples (règles d'apprentissage les moins complexes possibles) est à priori une des solutions pour obtenir des capteurs intelligents à basse consommation énergétique.

La question se pose néanmoins sur l'architecture en elle-même des réseaux de neurones à implémenter dans un capteur pour le rendre intelligent et efficace énergétiquement. Si une grande partie des réseaux de neurones utilisés pour faire du traitement visuel utilisent généralement un empilement de couches de neurones, voire une simple couche *all-to-all* comme cela a été le cas pour ces travaux, peut-on affirmer que ce sont ces architectures qui sont les plus optimales pour des implantations basse consommation ?

Certes, ces types de réseaux de neurones fonctionnent et peuvent permettre d'atteindre de bon

taux de reconnaissance, mais rien ne prouve aujourd'hui que ce mécanisme d'empilement de couches, résultant en des réseaux de neurones dits profonds, est LA solution présentant le meilleur rapport entre le taux de reconnaissance et la consommation énergétique. S'intéresser à la construction d'architectures différentes telles que les réseaux de neurones récurrents ou les réseaux de neurones à réservoir, où la notion de couches semble s'effacer, peut être intéressant puisque dans le cas du cerveau biologique l'agencement complexe des neurones entre eux ne relève pas d'un simple empilement de couches de neurones.

Les outils qui ont pu être développés lors de ces travaux de thèse, un simulateur flexible pour des réseaux de neurones événementiels comme décrit dans le CHAPITRE 2, mais aussi d'une plateforme de test et de caractérisation de memristors ferroélectriques comme décrits dans les CHAPITRE 3 et CHAPITRE 4 pourraient être utiles pour aider à répondre à cette question. Pour cela, le simulateur aurait un rôle de prospection pour tester des architectures et règles d'apprentissage variées et peut-être plus exotiques comme cela a été le cas dans le CHAPITRE 2, et la plateforme 81×10 servirait soit à l'extraction des propriétés de synapses matérielles artificielles telles que des memristors ferroélectriques pour les intégrer dans les simulations, soit à démontrer expérimentalement de nouvelles règles d'apprentissages.

Annexe A

Architecture modulaire du simulateur

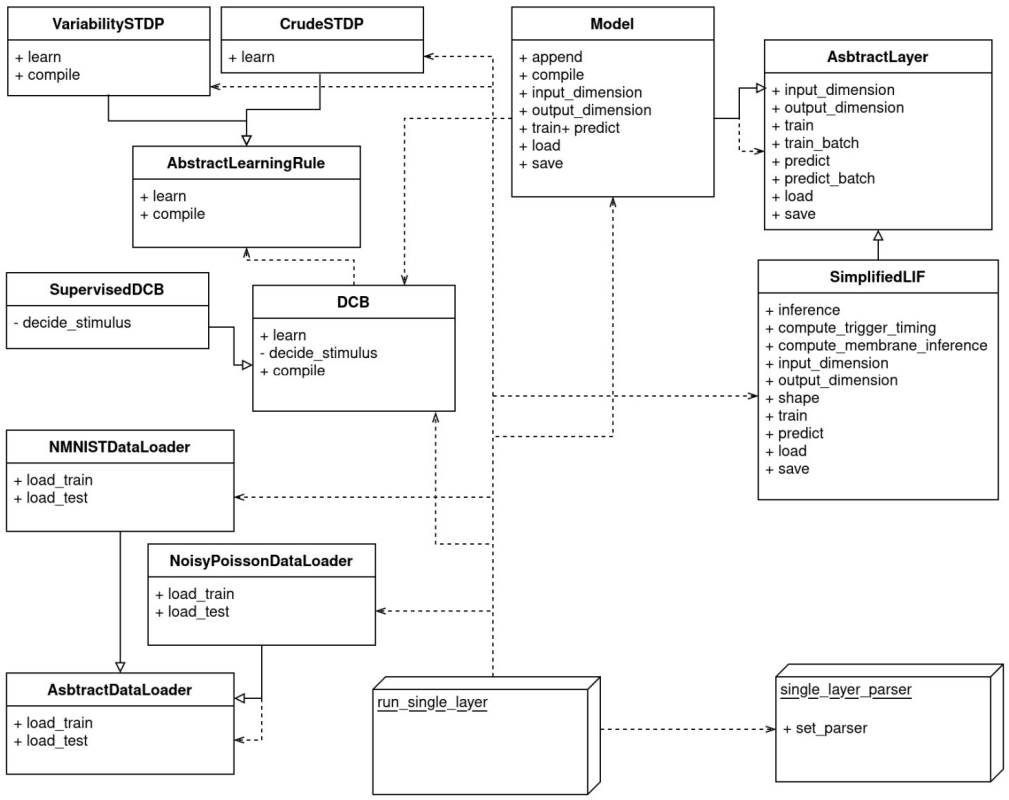


FIGURE A.1 – Architecture (pseudo diagramme) simplifiée du simulateur.

Le simulateur a été conçu au cours de ces travaux de thèse il y a considérablement évoluer. La version représentée ici est le résultat de la contribution de plusieurs personnes, mais ne représente pas la dernière version disponible du simulateur où de nouveaux type de couche sont utilisables en supplément comme LeakyLIF (modèle de neurones à résistance de décharge) en plus de SimplifiedLIF par exemple.

Annexe B

Circuits électroniques du bloc SPK

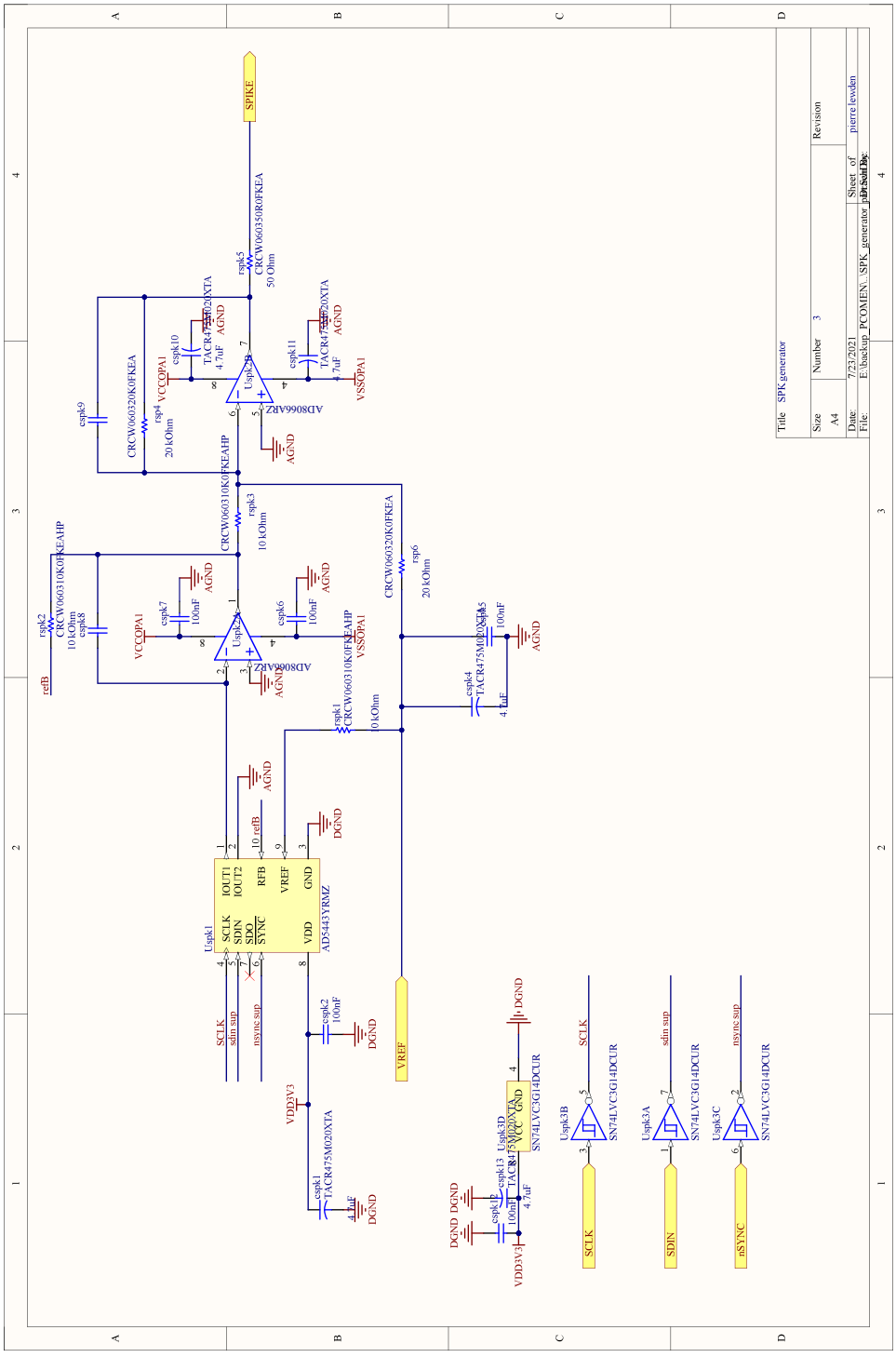


FIGURE B.1 – Partie 1 du ircuit électronique du bloc SPK.
Il s'agit de la partie émetteur et DAC bloc du circuit SPK. On peut noter la séparation des masses numériques (DGND) et analogiques (AGND) du circuit.

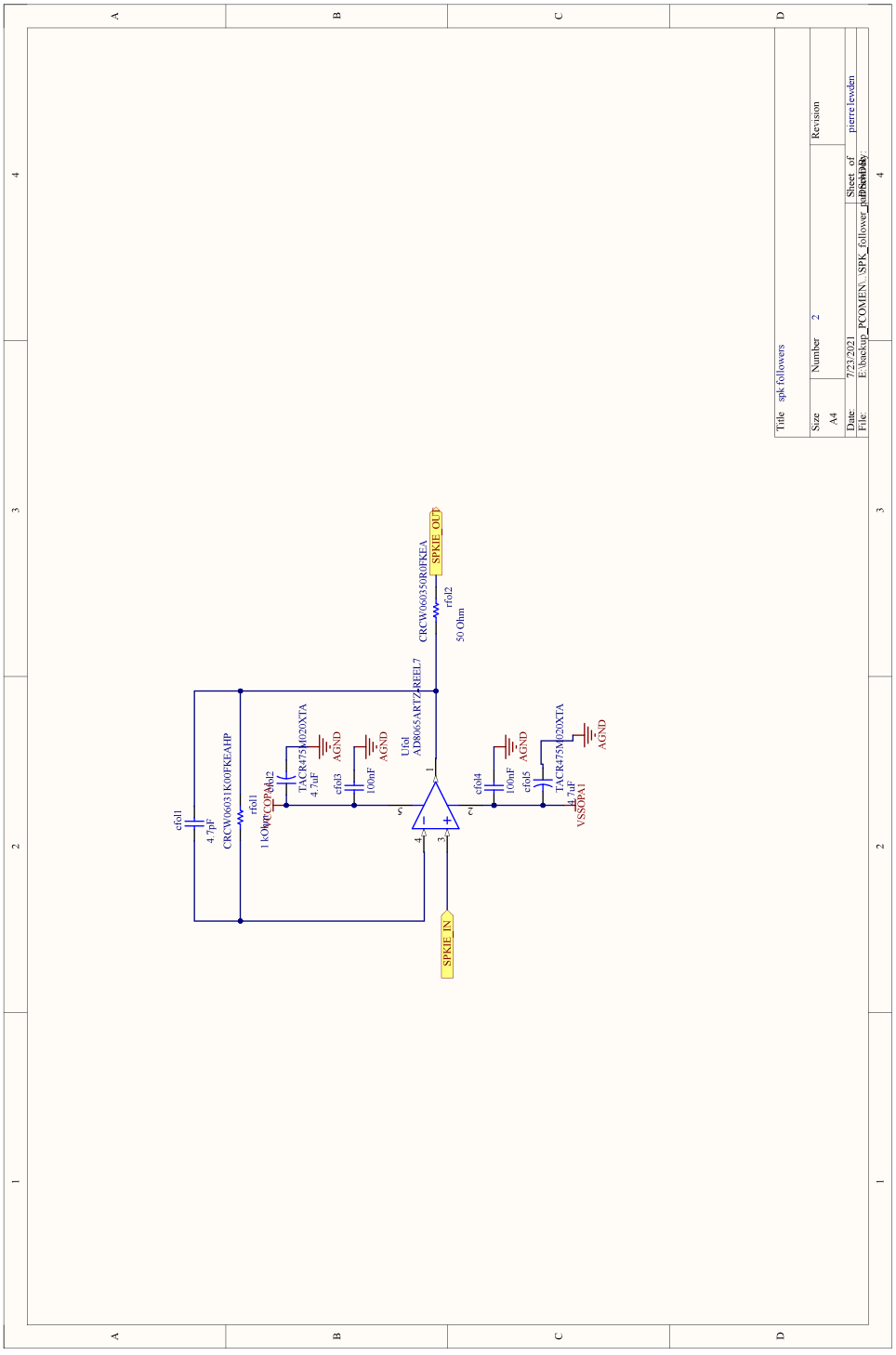


FIGURE B.2 – Partie 2 du circuit électronique du bloc SPK.
Il s'agit de la partie suiveur du circuit SPK.

Annexe C

Circuits électroniques du bloc POST

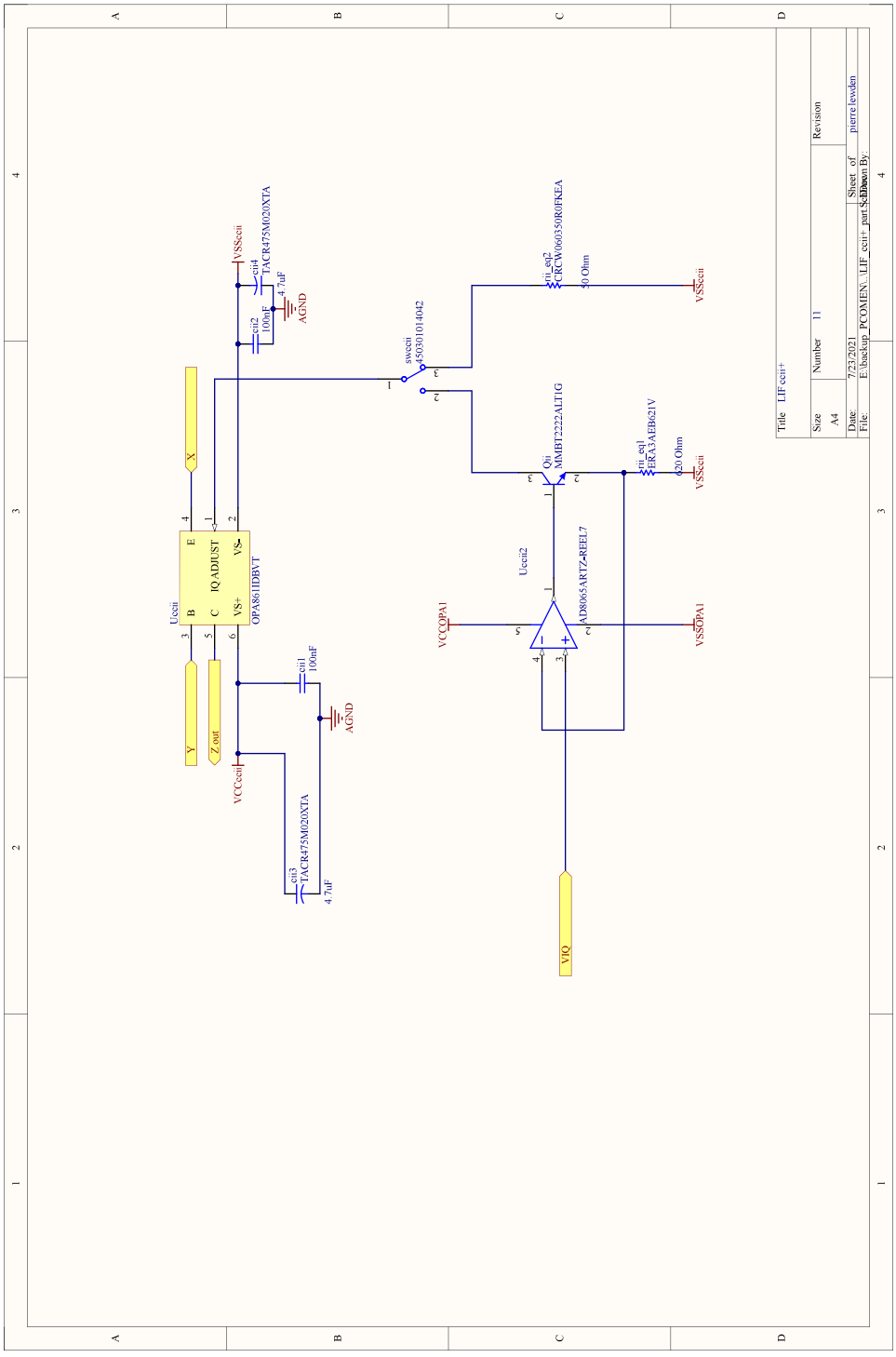


FIGURE C.1 – Circuit de l'étage d'entrée à convoyeur de courant.

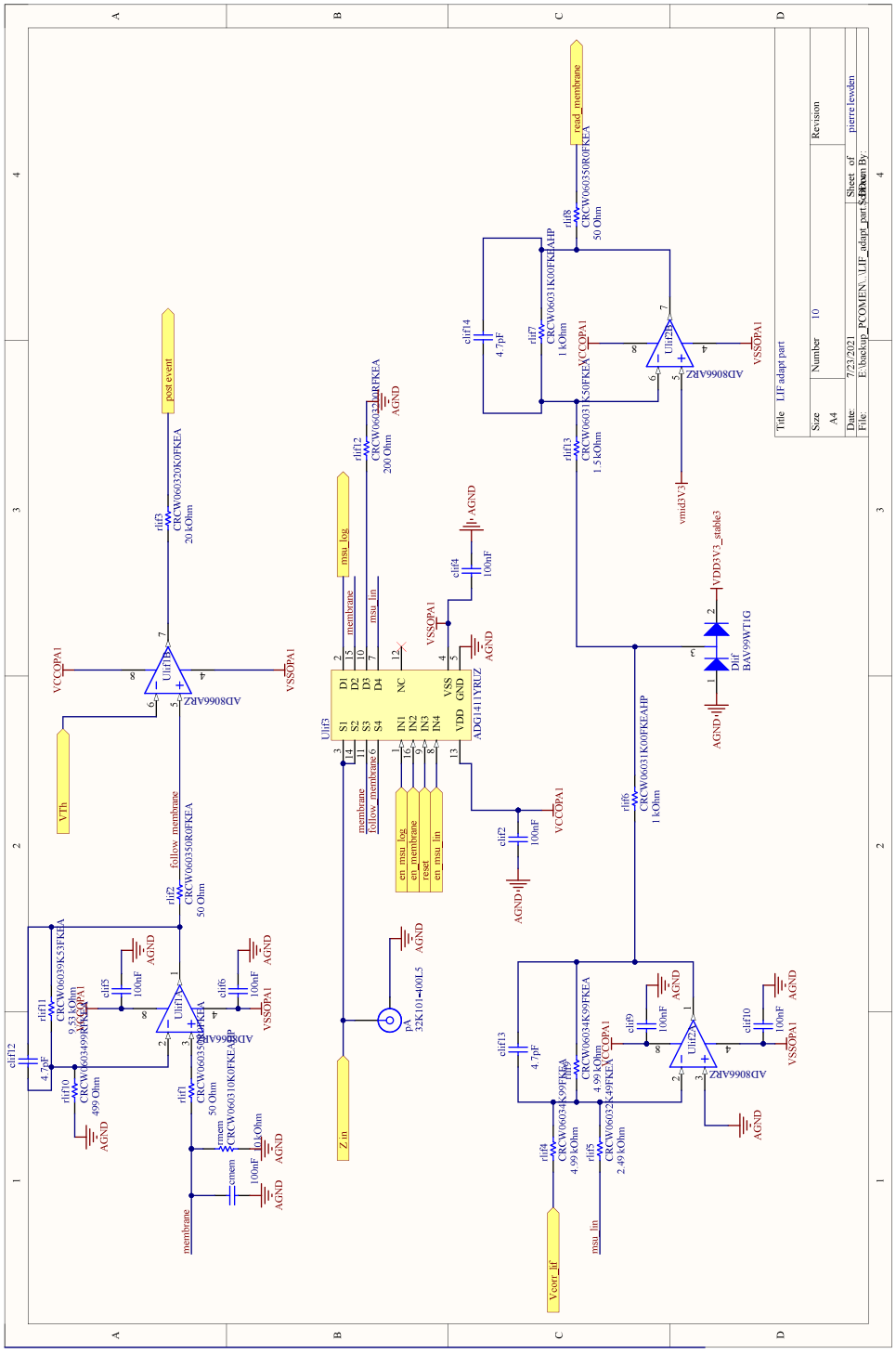


FIGURE C.2 – Circuit du module LIF, de MSU LIN et des interrupteurs analogiques. Le composant utilisé pour Ulif1B présente une limitation pour une utilisation comme comparateur (voir documentation technique) et à vocation à être remplacé.

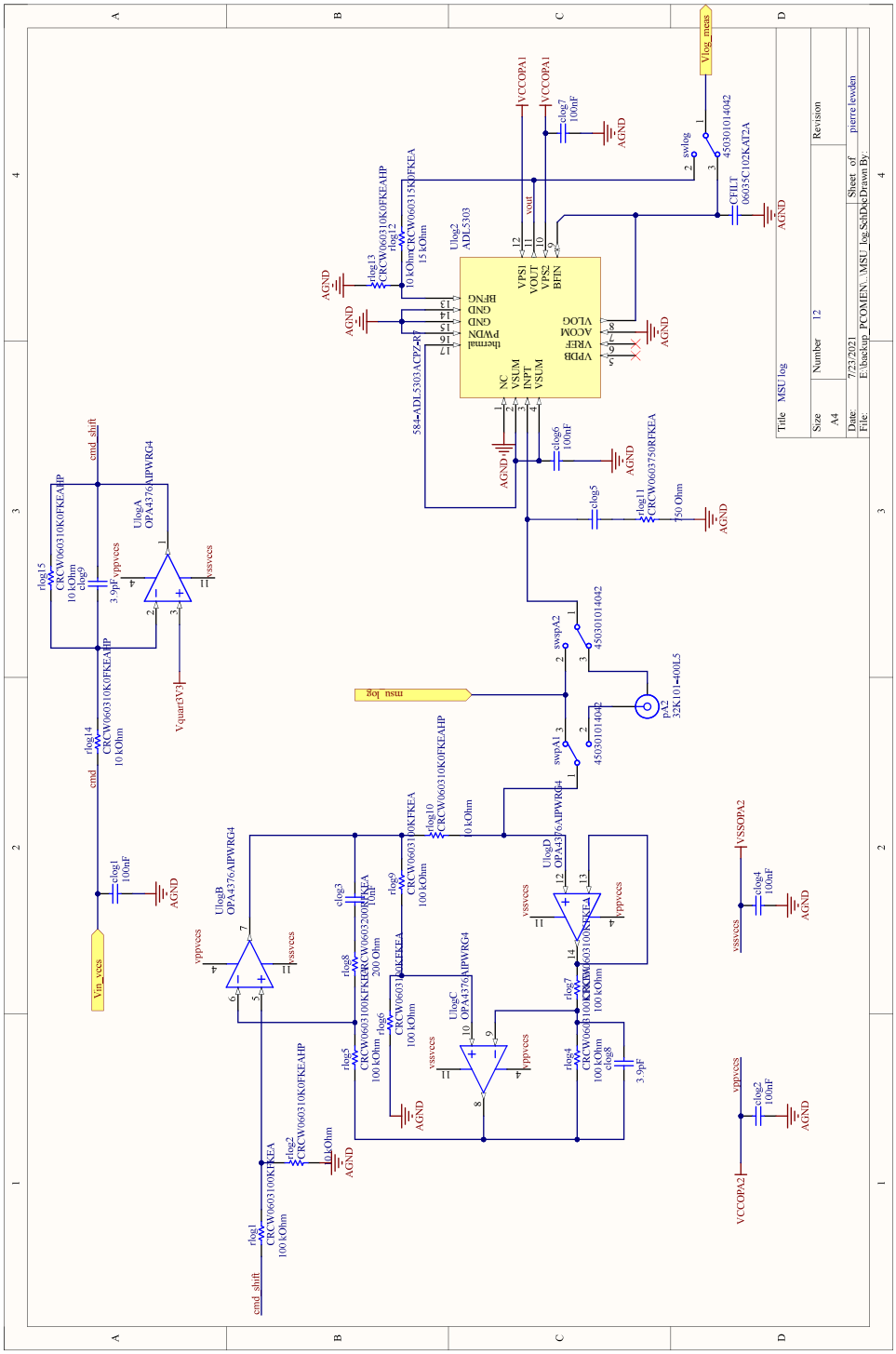


FIGURE C.3 – Circuit de MSU LOG

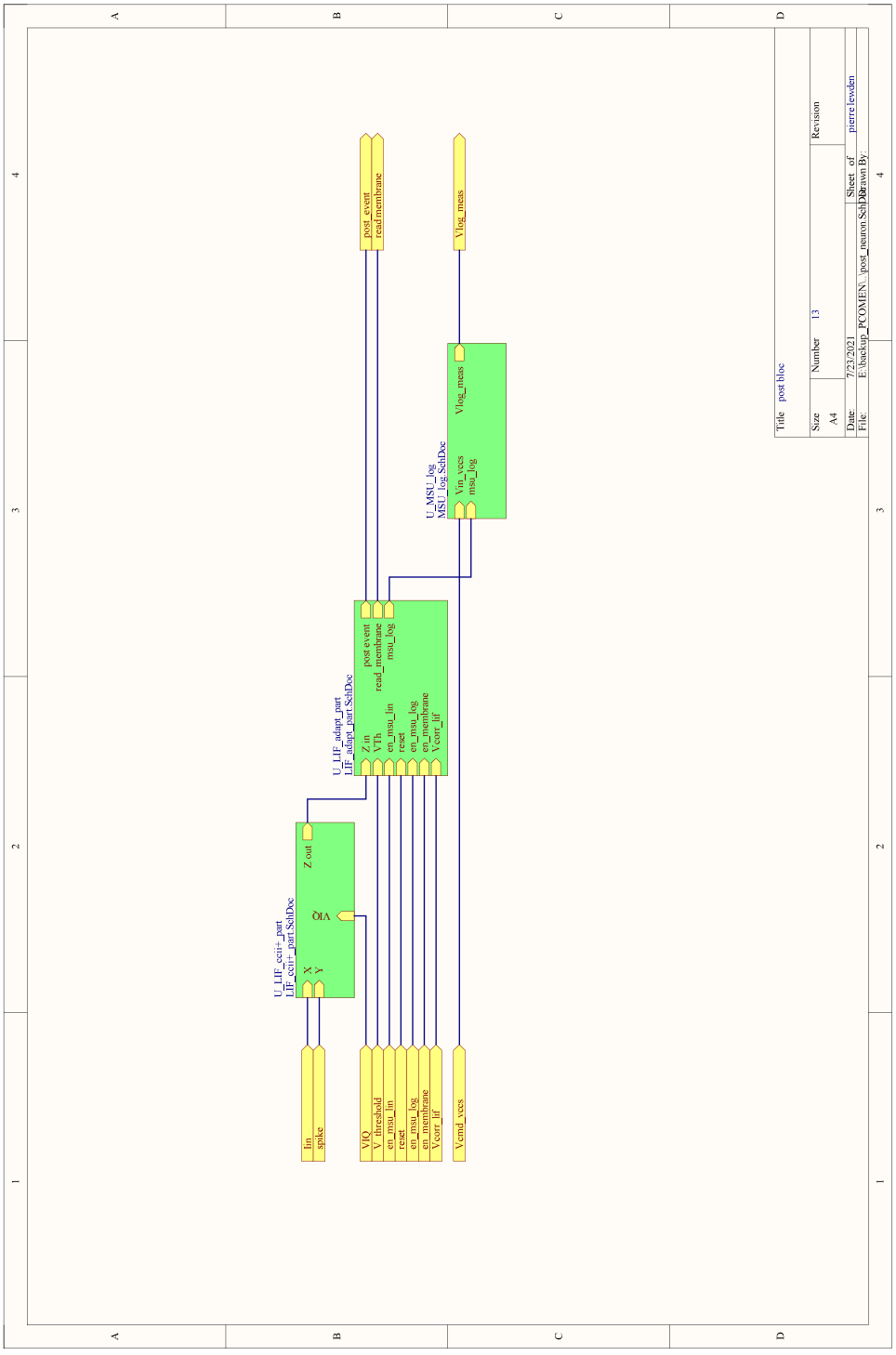


FIGURE C.4 – Connexion des circuits constituant POST

Annexe D

Circuits électroniques de la carte VCCS2

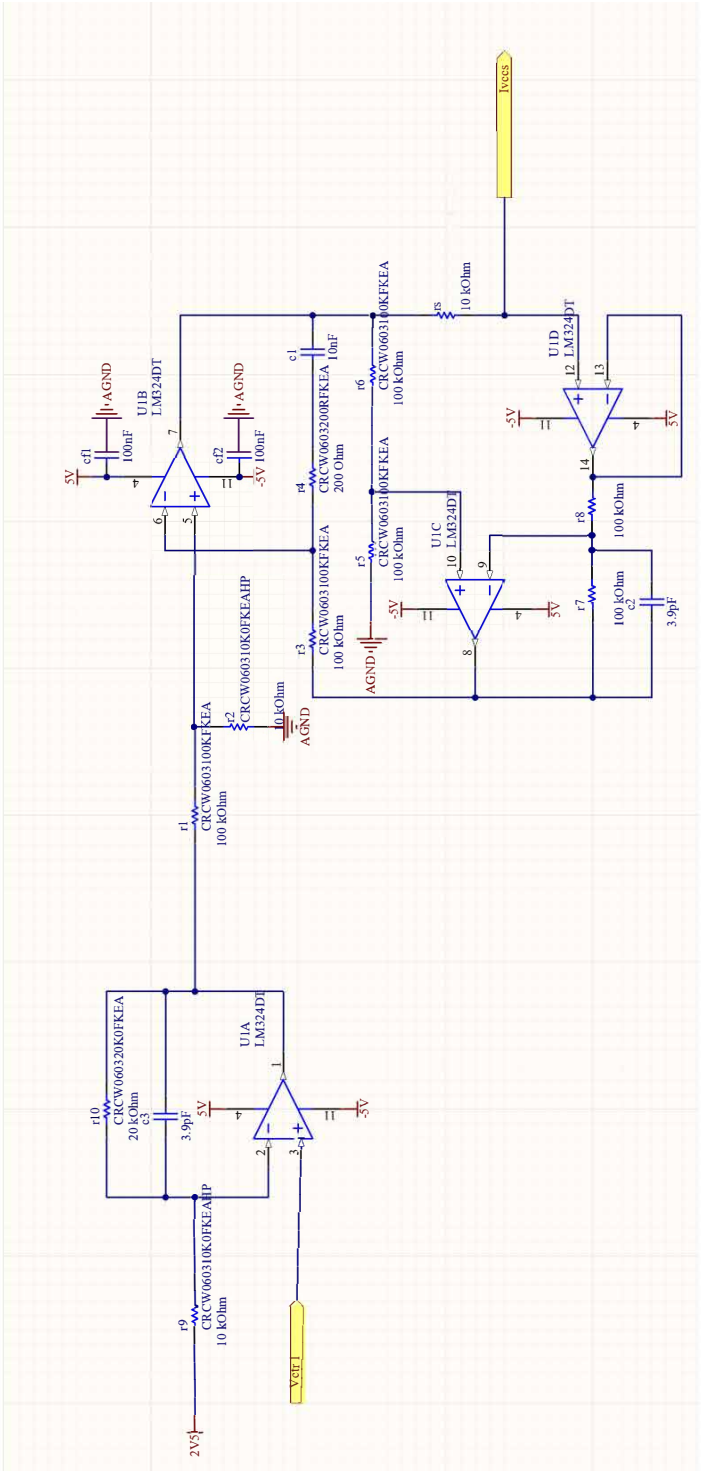


FIGURE D.1 – Circuit de la VCCS répétée 10 fois sur la carte VCCS2.

Annexe E

Acquisition et extraction de valeur de résistance.

Cette annexe décrit les protocoles allant de l'acquisition d'un point de mesure sur la plateforme au calcul de la valeur d'un memristor par le traitement des données expérimentales en Python (post-traitement). Il s'agit des protocoles utilisés pour commencer les premiers tests de la plateforme et ils ont donc pour vocation d'être ajustés (voire corrigés) dans le futur si nécessaire.

Pour la compréhension de cette annexe, il est fortement conseillé de relire la partie 3.2.2 Blocs POST (neurones postsynaptiques multifonctions).

Le terme « code » est utilisé ici pour dénommer une valeur sur 12 bits issue de l'ADC de mesure.

Le terme « point » (faux ami) est utilisé ici pour dénommer un ensemble de codes pour une mesure souhaitée. Il s'agit donc de l'ensemble des données brutes (codes) obtenues par la plateforme pour une mesure souhaitée (un point dans le tracé d'une hystérésis par exemple) avant post-traitement.

Acquisition d'un code stable de mesure (sur la plateforme) :

Avant de rentrer dans le détail des protocoles permettant l'acquisition des points de mesure qui seront ensuite utilisés et traités pour en extraire la valeur de résistance d'un composant memristif sur la plateforme, il faut aborder la méthode utilisée pour l'acquisition d'un code unique de mesure. L'acquisition sur la plateforme est assurée par un ADC 12 bits qui mesure la tension V_{log} du neurone postsynaptique d'intérêt. Pour rappel, cette tension est une image logarithmique du courant mesuré en sortie du convoyeur de courant (image du courant synaptique) comme détaillé 3.2.2 Blocs POST (neurones postsynaptiques multifonctions). Lorsqu'une acquisition est lancée [1], le protocole suivant est suivi pour s'assurer que la mesure est considérée stable :

1. Acquisition de N_{window} codes (valeur sur 12 bits émise par l'ADC lors d'une acquisition) de mesure de l'ADC d'acquisition du bloc ACQ de la carte MIRA. Pour les différents résultats présentés de ce manuscrit $N_{window} = 500$.
2. Calcul du code moyen à partir des N_{window} codes acquis.
3. Acquisition de nouveaux codes de mesure et calcul d'un nouveau code moyen suivant la méthode de moyenne glissante avec toujours le même nombre de points dans la fenêtre N_{window}.
   (a) Si que le code moyen varie de plus de 0.5 %, l'acquisition continue.
   (b) Si que le code moyen varie de moins de 0.5 %, le code moyen correspond au point de mesure.
4. Le code moyen obtenu est celui qui est conservé et enregistré.

1. On considère ici que la configuration d'un bloc POST pour pouvoir effectuer cette mesure a été effectuée et que les tensions voulues aux bornes des composants memristifs ont été appliquées.

Protocole de mesure des points (sur la plateforme) :

La méthode de mesure de poids synaptiques est basée sur une acquisition différentielle (sans et avec tension de mesure) suivant le protocole suivant :

1. Mise à V_{mid} de tout les memristors.
2. Acquisition de codes stable de référence par balayage de VCCS1 (source de courant pilotée en tension [2]) :
   (a) Configuration de VCCS1 à un courant de départ (vccs_start).
   (b) Acquisition d'un code stable.
   (c) Jusqu'à atteindre une configuration finale de VCCS1 (vccs_stop) :
      i. Augmentation du courant généré par VCCS1.
      ii. Acquisition d'un code stable pour cette nouvelle configuration de VCCS1.
3. Application sur le memristor (échantillon unique) ou la ligne présynaptique d'intérêt (crossbar) de la tension de mesure.
4. Acquisition des codes de mesure pour tous les postsynaptiques avec le balayage de VCCS1 (méthode donnée étape 2.).

Ce balayage de VCCS1 pour l'acquisition d'un point de référence et de mesure permet par la suite de sélectionner la meilleure configuration de VCCS1 lors du calcul de la résistance du memristor en plaçant $V_{\log}$ dans un niveau de tension correspondant à un courant qui n'est pas trop élevé pour que la différence entre la référence et la mesure soit non nulle ou négative (observé expérimentalement lorsque la résistance du composant est trop élevée ou que ce dernier n'est pas connecté dans le crossbar). En effet, $V_{\log}$ étant une représentation logarithmique du courant, si ce dernier est trop élevé lorsque la tension de mesure est appliquée sur un memristor, il peut ne pas être « visible » si le courant de mesure traversant le memristor est trop faible par rapport à ce courant de référence. Le courant en sortie de VCCS1 permet ainsi d'ajuster ce point de référence.

Extraction des meilleurs codes d'un point de mesure (Python) :

Après expérimentation, les résultats bruts obtenus contiennent pour chaque lancement d'une mesure sur un memristor (point de référence et point de mesure) plusieurs codes correspondant au balayage de VCCS1. Lorsque l'on souhaite extraire les valeurs de résistance des résultats expérimentaux bruts, on doit sélectionner pour chaque balayage une configuration de VCCS1 pour ne garder qu'un code de référence et de mesure qui seront utilisés pour le calcul de la résistance du composant. Pour cela, on considère :

— Que la tension de mesure cause une augmentation du courant acquis par le bloc MSU LOG d'acquisition du courant. Il y a donc une augmentation de la tension $V_{\log}$ et du code acquis par l'ADC entre la référence et la mesure.

2. Pour rappel, la tension de contrôle est contrôlée numériquement.

— Un code minimal de l'ADC en dessous duquel les tensions $V_{\log}$ correspondent à des courants trop faibles pour la référence. Lors des diverses expérimentations réalisées, on ne s'intéresse qu'aux codes de l'ADC 12 bits supérieurs ou égals à 2000, soit une tension $V_{\log}$ de 1.611 V correspondant en théorie à I_{in} =167 nA (courant acquis par MSU LOG) d'après la formule théorique du composant ADL5303 (E.1) et (E.2).

$$V_{\log} = V_{\text{slope}} \times \log_{10}\left(\frac{I_{in}}{I_Z}\right) \tag{E.1}$$

$$I_{in} = I_Z \times 10^{\left(\frac{V_{\log}}{V_{\text{slope}}}\right)} \tag{E.2}$$

Pour les blocs POST, les valeurs théoriques (non caractérisées) sont V_{slope} =500 mV et I_Z =100 pA.

On sélectionne alors dans la plage d'acquisition pour la référence la configuration de VCCS1 où le code est le minimum tout en étant supérieur à 2000. Si pour cette configuration de VCCS1, l'acquisition pour la mesure présente un code supérieur à celui pour la référence, le point de mesure est valide et ce sont ces valeurs qui seront utilisées pour le calcul de résistance. Si l'acquisition pour la mesure présente un code inférieur ou égal à celui pour la référence (soit par ce que la résistance du composant est trop faible soit parce qu'elle n'est pas connectée dans la structure), le calcul de résistance donnerait une résistance nulle ou négative. Dans une telle situation, le point de mesure est invalide et ne sera donc pas utilisé.

Calcul de résistance (Python) :

Si on a un code de référence et de mesure valide, on peut calculer la valeur de résistance suivant les formules théoriques données (E.3) à (E.5) . Théorique puisque la caractérisation de la plateforme n'a pas été réalisée. Les paramètres de ces équations sont :

— $V_{\log\ \text{ref}}$ et $V_{\log\ \text{mes}}$, les deux tensions acquises par l'ADC à la configuration de VCCS1 choisie.

— I_Z =100 pA.

— V_{slope} =500 mV.

— V_{mes} correspond à la tension choisie pour la mesure.

$$I_{\text{ref}} = I_Z \times 10^{\left(\frac{V_{\log\ \text{ref}}}{V_{\text{slope}}}\right)} \tag{E.3}$$

$$I_{\text{mes}} = I_Z \times 10^{\left(\frac{V_{\log\ \text{mes}}}{V_{\text{slope}}}\right)} \tag{E.4}$$

$$R = \frac{V_{\text{mes}}}{I_{\text{mes}} - I_{\text{ref}}} \tag{E.5}$$

Bibliographie

[1] Nicola Jones et al. How to stop data centres from gobbling up the world's electricity. *Nature*, 561(7722) :163–6, 2018.

[2] Gordon E. Moore. Cramming more components onto integrated circuits, reprinted from electronics, volume 38, number 8, april 19, 1965, pp.114 ff. *IEEE Solid-State Circuits Society Newsletter*, 11(3) :33–35, 2006.

[3] John Shalf. The future of computing beyond Moore's law. *Philosophical Transactions of the Royal Society A*, 378(2166) :20190061, 2020.

[4] Arindam Mallik, Julien Ryckaert, RH Kim, P Debacker, S Decoster, F Lazzarino, R Ritzenthaler, N Horiguchi, D Verkest, and A Mocuta. Economics of semiconductor scaling-a cost analysis for advanced technology node. In *2019 Symposium on VLSI Technology*, pages T202–T203. IEEE, 2019.

[5] II Arikpo, FU Ogban, and IE Eteng. Von Neumann architecture and modern computers. *Global Journal of Mathematical Sciences*, 6(2) :97–103, 2007.

[6] Jean-Pierre Changeux. L'homme neuronal. 1983.

[7] Léa Derome. Le cerveau selon Aristote. 2017.

[8] Michele Angelo Murgia. *Qu'est-ce que le temps ?* Atramenta, 2015.

[9] Laeticia Loviconi. Cœur, cerveau et nerfs dans les théories explicatives antiques et médiévales des mouvements volontaires. *Société des Neurosciences*, 63(1), 2022.

[10] Stuart Zola-Morgan. Localization of brain function : The legacy of Franz Joseph Gall (1758-1828). *Annual review of neuroscience*, 18(1) :359–383, 1995.

[11] Goran Šimic and Patrick R Hof. In search of the definitive Brodmann's map of cortical areas in human. *J. Comp. Neurol*, 523 :5–14, 2015.

[12] Alfredo Ardila, Byron Bernal, and Monica Rosselli. How Localized are Language Brain Areas ? A Review of Brodmann Areas Involvement in Oral Language. *Archives of Clinical Neuropsychology*, 31(1) :112–122, 12 2015.

[13] Seyed Yahya Shirazi and Helen J Huang. More reliable EEG electrode digitizing methods can reduce source estimation uncertainty, but current methods already accurately identify brodmann areas. *Frontiers in neuroscience*, 13 :1159, 2019.

[14] Andreas Winkelmann. Wilhelm von Waldeyer-Hartz (1836–1921) : An anatomist who left his mark. *Clinical Anatomy*, 20(3) :231–234, 2007.

[15] Max R Bennett. The early history of the synapse : from Plato to Sherrington. *Brain research bulletin*, 50(2) :95–118, 1999.

[16] Rainer W Guillery. Observations of synaptic structures : origins of the neuron doctrine and its current status. *Philosophical Transactions of the Royal Society B : Biological Sciences*, 360(1458) :1281–1307, 2005.

[17] Sanford L Palay and George E Palade. The fine structure of neurons. *The Journal of biophysical and biochemical cytology*, 1(1) :69, 1955.

[18] Eduardo DP De Robertis and H Stanley Bennett. Some features of the submicroscopic morphology of synapses in frog and earthworm. *The Journal of Cell Biology*, 1(1) :47–58, 1955.

[19] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5 :115–133, 1943.

[20] Frank Rosenblatt. The perceptron : a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6) :386, 1958.

[21] W. Clark and B. Farley. Generalization of pattern recognition in a self-organizing system. In *Proceedings of the March 1-3, 1955, western joint computer conference*, pages 86–91, 1955.

[22] B. Farley and W. Clark. Simulation of self-organizing systems by digital computer. *Transactions of the IRE Professional Group on Information Theory*, 4(4) :76–84, 1954.

[23] Donald O Hebb. The first stage of perception : growth of the assembly. *The Organization of Behavior*, 4 :60–78, 1949.

[24] Bernard Widrow and Michael A Lehr. 30 years of adaptive neural networks : perceptron, madaline, and backpropagation. *Proceedings of the IEEE*, 78(9) :1415–1442, 1990.

[25] Jürgen Schmidhuber. Deep learning in neural networks : An overview. *Neural Networks*, 61 :85–117, 2015.

[26] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4) :541–551, 1989.

[27] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11) :2278–2324, 1998.

[28] Sergio Rivas-Gomez, Antonio J Pena, David Moloney, Erwin Laure, and Stefano Markidis. Exploring the vision processing unit as co-processor for inference. In *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 589–598. IEEE, 2018.

[29] Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th annual international symposium on computer architecture*, pages 1–12, 2017.

[30] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in NLP. *arXiv preprint arXiv :1906.02243*, 2019.

[31] Wolfgang Maass. Networks of spiking neurons : the third generation of neural network models. *Neural networks*, 10(9) :1659–1671, 1997.

[32] Simon W Moore, Paul J Fox, Steven JT Marsh, A Theodore Markettos, and Alan Mujumdar. Bluehive-a field-programable custom computing machine for extreme-scale real-time neural network simulation. In *2012 IEEE 20th International Symposium on Field-Programmable Custom Computing Machines*, pages 133–140. IEEE, 2012.

[33] K Cheung, SR Schultz, and W Luk. Neuroflow : a general purpose spiking neural network simulation platform using customizable processors. front. neurosci. 9, 1–15 (2016), 2015.

[34] Steve B Furber, David R Lester, Luis A Plana, Jim D Garside, Eustace Painkras, Steve Temple, and Andrew D Brown. Overview of the spinnaker system architecture. *IEEE transactions on computers*, 62(12) :2454–2467, 2012.

[35] Ben Varkey Benjamin, Peiran Gao, Emmett McQuinn, Swadesh Choudhary, Anand R Chandrasekaran, Jean-Marie Bussat, Rodrigo Alvarez-Icaza, John V Arthur, Paul A Merolla, and Kwabena Boahen. Neurogrid : A mixed-analog-digital multichip system for large-scale neural simulations. *Proceedings of the IEEE*, 102(5) :699–716, 2014.

[36] Jongkil Park, Theodore Yu, Siddharth Joshi, Christoph Maier, and Gert Cauwenberghs. Hierarchical address event routing for reconfigurable large-scale neuromorphic systems. *IEEE transactions on neural networks and learning systems*, 28(10) :2408–2422, 2016.

[37] Ning Qiao, Hesham Mostafa, Federico Corradi, Marc Osswald, Fabio Stefanini, Dora Sumislawska, and Giacomo Indiveri. A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128k synapses. *Frontiers in neuroscience*, 9 :141, 2015.

[38] Sebastian Schmitt, Johann Klähn, Guillaume Bellec, Andreas Grübl, Maurice Guettler, Andreas Hartel, Stephan Hartmann, Dan Husmann, Kai Husmann, Sebastian Jeltsch, et al. Neuromorphic hardware in the loop : Training a deep spiking network on the brainscales wafer-scale system. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 2227–2234. IEEE, 2017.

[39] Paul A Merolla, John V Arthur, Rodrigo Alvarez-Icaza, Andrew S Cassidy, Jun Sawada, Filipp Akopyan, Bryan L Jackson, Nabil Imam, Chen Guo, Yutaka Nakamura, et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197) :668–673, 2014.

[40] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. Loihi : A neuromorphic manycore processor with on-chip learning. *Ieee Micro*, 38(1) :82–99, 2018.

[41] Suzana Herculano-Houzel. The human brain in numbers : a linearly scaled-up primate brain. *Frontiers in Human Neuroscience*, 3, 2009.

[42] Peter Lennie. The cost of cortical computation. *Current Biology*, 2003.

[43] Karlheinz Meier. Special report : Can we copy the brain ? - the brain as computer. *IEEE Spectrum*, 54(6) :28–33, 2017.

[44] William B Levy and Victoria G Calvert. Communication consumes 35 times more energy than computation in the human cortex, but both costs are needed to predict synapse number. *Proceedings of the National Academy of Sciences*, 118(18) :e2008173118, 2021.

[45] Adrien F. Vincent. *Vers une utilisation synaptique de composants mémoires innovants pour l'électronique neuro-inspirée*. PhD thesis, 2017. Thèse de doctorat dirigée par Galdin-Retailleau, Sylvie Electronique et optoélectronique, nano- et microtechnologies Université Paris-Saclay (ComUE) 2017.

[46] Barry W Connors and Michael J Gutnick. Intrinsic firing patterns of diverse neocortical neurons. *Trends in neurosciences*, 13(3) :99–104, 1990.

[47] Bruce P Bean. The action potential in mammalian central neurons. *Nature Reviews Neuroscience*, 8(6) :451–465, 2007.

[48] Stephen H Wright. Generation of resting membrane potential. *Advances in physiology education*, 28(4) :139–142, 2004.

[49] Steven M. Chrysafides, Stephen J. Bordes, and Sandeep Sharma. *Physiology, Resting Potential*. StatPearls Publishing, Treasure Island (FL), 2022.

[50] Jonathan Platkiewicz and Romain Brette. A threshold equation for action potential initiation. *PLoS computational biology*, 6(7) :e1000850, 2010.

[51] Alan Peters and Sanford L Palay. The morphology of synapses. *Journal of neurocytology*, 25 :687–700, 1996.

[52] Alberto E Pereda. Electrical synapses and their functional interactions with chemical synapses. *Nature Reviews Neuroscience*, 15(4) :250–263, 2014.

[53] Donald Duncan and Ricardo Morales. Relative numbers of several types of synaptic connections in the substantia gelatinosa of the cat spinal cord. *Journal of Comparative Neurology*, 182(4) :601–609, 1978.

[54] R Hassler and JW Chung. The discrimination of nine different types of synaptic boutons in the fundus striati (nucleus accumbens septi). *Cell and Tissue Research*, 168 :489–505, 1976.

[55] Mark Mayford, Steven A Siegelbaum, and Eric R Kandel. Synapses and memory storage. *Cold Spring Harbor perspectives in biology*, 4(6) :a005751, 2012.

[56] Guo-qiang Bi and Mu-ming Poo. Synaptic modifications in cultured hippocampal neurons : dependence on spike timing, synaptic strength, and postsynaptic cell type. *Journal of neuroscience*, 18(24) :10464–10472, 1998.

[57] Zuzanna Brzosko, Sara Zannone, Wolfram Schultz, Claudia Clopath, and Ole Paulsen. Sequential neuromodulation of hebbian plasticity offers mechanism for effective reward-based navigation. *Elife*, 6 :e27756, 2017.

[58] Nicolas Frémaux and Wulfram Gerstner. Neuromodulated spike-timing-dependent plasticity, and theory of three-factor learning rules. *Frontiers in neural circuits*, 9 :85, 2016.

[59] Larry F Abbott and Sacha B Nelson. Synaptic plasticity : taming the beast. *Nature neuroscience*, 3(11) :1178–1183, 2000.

[60] Alan L Hodgkin and Andrew F Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4) :500, 1952.

[61] Eugene M Izhikevich. Which model to use for cortical spiking neurons? *IEEE Transactions on Neural Networks*, 15(5) :1063–1070, 2004.

[62] Andrew P Davison, Jianfeng Feng, and David Brown. A reduced compartmental model of the mitral cell for use in network models of the olfactory bulb. *Brain research bulletin*, 51(5) :393–399, 2000.

[63] Quentin JM Huys, Misha B Ahrens, and Liam Paninski. Efficient estimation of detailed single-neuron models. *Journal of neurophysiology*, 96(2) :872–890, 2006.

[64] Alain Destexhe, Zachary F Mainen, Terrence J Sejnowski, et al. Kinetic models of synaptic transmission. *Methods in neuronal modeling*, 2 :1–25, 1998.

[65] Timothée Levi, Farad Khoyratee, Sylvain Saïghi, and Yoshiho Ikeuchi. Digital implementation of hodgkin–huxley neuron model for neurological diseases studies. *Artificial Life and Robotics*, 23(1) :10–14, 2018.

[66] Farad Khoyratee. *Conception d'une plateforme modulable de réseau de neurones biomimétiques pour l'étude des maladies neurodégénératives*. Theses, Université de Bordeaux, December 2019.

[67] Sylvie Renaud, Jean Tomas, Yannick Bornat, Adel Daouzli, and Sylvain Saïghi. Neuromimetic ics with analog cores : an alternative for simulating spiking neural networks. In *2007 IEEE international symposium on circuits and systems*, pages 3355–3358. IEEE, 2007.

[68] Sylvain Saighi, Yannick Bornat, Jean Tomas, Gwendal Le Masson, and Sylvie Renaud. A library of analog operators based on the hodgkin-huxley formalism for the design of tunable, real-time, silicon neurons. *IEEE transactions on biomedical circuits and systems*, 5(1) :3–19, 2010.

[69] Laure Buhry, Filippo Grassia, Audrey Giremus, Eric Grivel, Sylvie Renaud, and Sylvain Saïghi. Automated parameter estimation of the hodgkin-huxley model using the differential evolution algorithm : application to neuromimetic analog integrated circuits. *Neural computation*, 23(10) :2599–2625, 2011.

[70] Filippo Grassia, Laure Buhry, Timothée Lévi, Jean Tomas, Alain Destexhe, and Sylvain Saïghi. Tunable neuromimetic integrated system for emulating cortical neuron models. *Frontiers in NEUROSCIENCE*, 5 :134, 2011.

[71] Larry F Abbott. Lapicque's introduction of the integrate-and-fire model neuron (1907). *Brain research bulletin*, 50(5-6) :303–304, 1999.

[72] Leon Chua. Memristor-the missing circuit element. *IEEE Transactions on circuit theory*, 18(5) :507–519, 1971.

[73] Dmitri B Strukov, Gregory S Snider, Duncan R Stewart, and R Stanley Williams. The missing memristor found. *nature*, 453(7191) :80–83, 2008.

[74] Rainer Waser, Regina Dittmann, Georgi Staikov, and Kristof Szot. Redox-based resistive switching memories–nanoionic mechanisms, prospects, and challenges. *Advanced materials*, 21(25-26) :2632–2663, 2009.

[75] Sung Hyun Jo, Ting Chang, Idongesit Ebong, Bhavitavya B Bhadviya, Pinaki Mazumder, and Wei Lu. Nanoscale memristor device as synapse in neuromorphic systems. *Nano letters*, 10(4) :1297–1301, 2010.

[76] Chao Du, Wen Ma, Ting Chang, Patrick Sheridan, and Wei D Lu. Biorealistic implementation of synaptic functions with oxide memristors through internal ionic dynamics. *Advanced Functional Materials*, 25(27) :4290–4299, 2015.

[77] Rainer Waser and Masakazu Aono. Nanoionics-based resistive switching memories. *Nature materials*, 6(11) :833–840, 2007.

[78] H-S Philip Wong, Heng-Yuan Lee, Shimeng Yu, Yu-Sheng Chen, Yi Wu, Pang-Shiu Chen, Byoungil Lee, Frederick T Chen, and Ming-Jinn Tsai. Metal–oxide rram. *Proceedings of the IEEE*, 100(6) :1951–1970, 2012.

[79] Fabien Alibart, Elham Zamanidoost, and Dmitri B Strukov. Pattern classification by memristive crossbar circuits using ex situ and in situ training. *Nature communications*, 4(1) :2072, 2013.

[80] Duygu Kuzum, Rakesh GD Jeyasingh, Byoungil Lee, and H-S Philip Wong. Nanoelectronic programmable synapses based on phase change materials for brain-inspired computing. *Nano letters*, 12(5) :2179–2186, 2012.

[81] Geoffrey W Burr, Robert M Shelby, Severin Sidler, Carmelo Di Nolfo, Junwoo Jang, Irem Boybat, Rohit S Shenoy, Pritish Narayanan, Kumar Virwani, Emanuele U Giacometti, et al. Experimental demonstration and tolerancing of a large-scale neural network (165 000 synapses) using phase-change memory as the synaptic weight element. *IEEE Transactions on Electron Devices*, 62(11) :3498–3507, 2015.

[82] Fabien Alibart, Stéphane Pleutin, Olivier Bichler, Christian Gamrat, Teresa Serrano-Gotarredona, Bernabe Linares-Barranco, and Dominique Vuillaume. A memristive nanoparticle/organic hybrid synapstor for neuroinspired computing. *Advanced Functional Materials*, 22(3) :609–616, 2012.

[83] Nicolas Locatelli, Vincent Cros, and Julie Grollier. Spin-torque building blocks. *Nature materials*, 13(1) :11–20, 2014.

[84] Patryk Krzysteczko, Jana Münchenberger, Markus Schäfers, Günter Reiss, and Andy Thomas. The memristive magnetic tunnel junction as a nanoscopic synapse-neuron system. *Advanced Materials*, 24(6) :762–766, 2012.

[85] Vaibhav Ostwal, Punyashloka Debashis, Rafatul Faria, Zhihong Chen, and Joerg Appenzeller. Spin-torque devices with hard axis initialization as stochastic binary neurons. *Scientific reports*, 8(1) :1–8, 2018.

[86] S Boyn, S Girod, Vincent Garcia, S Fusil, S Xavier, C Deranlot, H Yamada, C Carrétéro, E Jacquet, M Bibes, et al. High-performance ferroelectric memory based on fully patterned tunnel junctions. *Applied Physics Letters*, 104(5) :052909, 2014.

[87] André Chanthbouala, Vincent Garcia, Ryan O Cherifi, Karim Bouzehouane, Stéphane Fusil, Xavier Moya, Stéphane Xavier, Hiroyuki Yamada, Cyrile Deranlot, Neil D Mathur, et al. A ferroelectric memristor. *Nature materials*, 11(10) :860–864, 2012.

[88] Elisabeth Soergel. Piezoresponse force microscopy (pfm). *Journal of Physics D : Applied Physics*, 44(46) :464003, 2011.

[89] Sylvain Saïghi, Christian G. Mayr, Teresa Serrano-Gotarredona, Heidemarie Schmidt, Gwendal Lecerf, Jean Tomas, Julie Grollier, Sören Boyn, Adrien F. Vincent, Damien Querlioz, Selina La Barbera, Fabien Alibart, Dominique Vuillaume, Olivier Bichler, Christian Gamrat, and Bernabé Linares-Barranco. Plasticity in memristive devices for spiking neural networks. *Frontiers in Neuroscience*, 9, 2015.

[90] Sören Boyn, Julie Grollier, Gwendal Lecerf, Bin Xu, Nicolas Locatelli, Stéphane Fusil, Stéphanie Girod, Cécile Carrétéro, Karin Garcia, Stéphane Xavier, et al. Learning through ferroelectric domain dynamics in solid-state synapses. *Nature communications*, 8(1) :1–7, 2017.

[91] Adel S Sedra. The current conveyor : History and progress. In *IEEE International Symposium on Circuits and Systems,*, pages 1567–1571. IEEE, 1989.

[92] Gwendal Lecerf, Jean Tomas, and Sylvain Saïghi. Excitatory and inhibitory memristive synapses for spiking neural networks. In *2013 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1616–1619. IEEE, 2013.

[93] Laura Bégon-Lours, Mattia Halter, Francesco Maria Puglisi, Lorenzo Benatti, Donato Francesco Falcone, Youri Popoff, Diana Dávila Pineda, Marilyne Sousa, and Bert Jan Offrein. Scaled, ferroelectric memristive synapse for back-end-of-line integration with neuromorphic hardware. *Advanced Electronic Materials*, page 2101395, 2022.

[94] Kuk-Hwan Kim, Siddharth Gaba, Dana Wheeler, Jose M Cruz-Albrecht, Tahir Hussain, Narayan Srinivasa, and Wei Lu. A functional hybrid memristor crossbar-array/cmos system for data storage and neuromorphic applications. *Nano letters*, 12(1) :389–395, 2012.

[95] Vincent Chan, Shih-Chii Liu, and Andr van Schaik. Aer ear : A matched silicon cochlea pair with address event representation interface. *IEEE Transactions on Circuits and Systems I : Regular Papers*, 54(1) :48–59, 2007.

[96] Lloyd Watts, Douglas A Kerns, Richard F Lyon, and Carver A Mead. Improved implementation of the silicon cochlea. *IEEE Journal of Solid-state circuits*, 27(5) :692–700, 1992.

[97] Clémence Gillet, Adrien F Vincent, Bertrand Le Gal, and Sylvain Saïghi. Flexible design methodology for spike encoding implementation on fpga. In *2022 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pages 379–383. IEEE, 2022.

[98] Misha Mahowald. Vlsi analogs of neuronal visual processing : a synthesis of form and function. 1992.

[99] Tobi Delbrück, Bernabe Linares-Barranco, Eugenio Culurciello, and Christoph Posch. Activity-driven, event-based vision sensors. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pages 2426–2429. IEEE, 2010.

[100] Christoph Posch, Teresa Serrano-Gotarredona, Bernabe Linares-Barranco, and Tobi Delbruck. Retinomorphic event-based vision sensors : bioinspired cameras with spiking output. *Proceedings of the IEEE*, 102(10) :1470–1484, 2014.

[101] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. High speed and high dynamic range video with an event camera. *IEEE transactions on pattern analysis and machine intelligence*, 43(6) :1964–1980, 2019.

[102] K.A. Boahen. Point-to-point connectivity between neuromorphic chips using address events. *IEEE Transactions on Circuits and Systems II : Analog and Digital Signal Processing*, 47(5) :416–434, 2000.

[103] Garrick Orchard, Ajinkya Jayawant, Gregory K Cohen, and Nitish Thakor. Converting static image datasets to spiking neuromorphic datasets using saccades. *Frontiers in neuroscience*, 9 :437, 2015.

[104] Gwendal Lecerf. *Développement d'un réseau de neurones impulsionnels sur silicium à synapses memristives*. PhD thesis, Bordeaux, 2014.

[105] Olivier Bichler, Damien Querlioz, Simon J Thorpe, Jean-Philippe Bourgoin, and Christian Gamrat. Extraction of temporally correlated features from dynamic vision sensors with spike-timing-dependent plasticity. *Neural networks*, 32 :339–348, 2012.

[106] Paul Ferré, Franck Mamalet, and Simon J Thorpe. Unsupervised feature learning with winner-takes-all based stdp. *Frontiers in computational neuroscience*, 12 :24, 2018.

[107] Damien Querlioz, Olivier Bichler, Philippe Dollfus, and Christian Gamrat. Immunity to device variations in a spiking neural network with memristive nanodevices. *IEEE transactions on nanotechnology*, 12(3) :288–295, 2013.

[108] Timothée Masquelier and Simon J Thorpe. Unsupervised learning of visual features through spike timing dependent plasticity. *PLoS computational biology*, 3(2) :e31, 2007.

[109] L. R. Iyer and A. Basu. Unsupervised learning of event-based image recordings using spike-timing-dependent plasticity. *International Joint Conference on Neural Networks (IJCNN)*, pages 1840–1846, 2017.

[110] Jun Haeng Lee, Tobi Delbruck, and Michael Pfeiffer. Training deep spiking neural networks using backpropagation. *Frontiers in neuroscience*, 10 :508, 2016.

[111] Milad Mozafari, Saeed Reza Kheradpisheh, Timothée Masquelier, Abbas Nowzari-Dalini, and Mohammad Ganjtabesh. First-spike-based visual categorization using reward-modulated stdp. *IEEE transactions on neural networks and learning systems*, 29(12) :6178–6190, 2018.

[112] Milad Mozafari, Mohammad Ganjtabesh, Abbas Nowzari-Dalini, Simon J Thorpe, and Timothée Masquelier. Bio-inspired digit recognition using reward-modulated spike-timing-dependent plasticity in deep convolutional networks. *Pattern recognition*, 94 :87–95, 2019.

[113] Pierre Lewden, Adrien F Vincent, Charly Meyer, Jean Tomas, and Sylvain Sïghi. Toward hardware spiking neural networks with mixed-signal event-based learning rules. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

[114] Charly Meyer. *Conception de réseaux de neurones sur silicium à l'aide de synapses memristives : application au traitement d'image*. PhD thesis, 2021. Thèse de doctorat dirigée par Saighi, Sylvain Electronique Bordeaux 2021.

[115] Pierre Lewden, Adrien F Vincent, Charly Meyer, Jean Tomas, Shidoush Siami, and Sylvain Saïghi. Hardware spiking neural networks : Slow tasks resilient learning with longer term-memory bits. In *2019 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pages 1–4. IEEE, 2019.

[116] Lee Hansen. White paper : Zynq ultrascale+ mpsocs, unleash the unparalleled power and flexibility of zynq ultrascale+ mpsocs. Technical Report WP470 (v1.1), Xillinx, 2016.

[117] Praveen Kumar Reddy Maddikunta, Quoc-Viet Pham, Prabadevi B, N Deepa, Kapal Dev, Thippa Reddy Gadekallu, Rukhsana Ruby, and Madhusanka Liyanage. Industry 5.0 : A survey on enabling technologies and potential applications. *Journal of Industrial Information Integration*, 26 :100257, 2022.

[118] Michele Compare, Piero Baraldi, and Enrico Zio. Challenges to IoT-enabled predictive maintenance for industry 4.0. *IEEE Internet of Things Journal*, 7(5) :4585–4597, 2019.

[119] European Commission, Directorate-General for Research, and Innovation. *Industry 5.0 : human-centric, sustainable and resilient*. Publications Office, 2021.

[120] European Commission, Directorate-General for Research, Innovation, M Breque, L De Nul, and A Petridis. *Industry 5.0 : towards a sustainable, human-centric and resilient European industry*. Publications Office of the European Union, 2021.

[121] European Commission, Directorate-General for Research, Innovation, A Renda, S Schwaag Serger, D Tataj, A Morlet, D Isaksson, F Martins, M Mir Roca, C Hidalgo, A Huang, S Dixson-Declève, P Balland, F Bria, C Charveriat, K Dunlop, and E Giovannini. *Industry 5.0, a transformative vision for Europe : governing systemic transformations towards a sustainable industry*. Publications Office of the European Union, 2022.

[122] Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, Michael Pfeiffer, and Shih-Chii Liu. Conversion of Continuous-Valued Deep Networks to Efficient Event-Driven Networks for Image Classification. *Frontiers in Neuroscience*, 11, 2017.

Titre : Implantations matérielles de réseaux de neurones à base de nanotechnologies pour le calcul événementiel.

Résumé : Cette thèse a eu lieu au sein de l'axe thématique « IA matérielle » du groupe Bioélectronique du laboratoire IMS, dans le cadre du projet de recherche européen ULPEC. Ce projet a eu pour ambition de créer un capteur neuromorphique intégré, intelligent et à basse consommation utilisant un réseau de neurones impulsionnels matériel en combinant des technologies innovantes telles qu'une rétine artificielle (caméra événementielle), des neurones en silicium et des memristors ferroélectriques comme synapse dans une même puce. Dans le cadre de ce projet, le travail réalisé durant cette thèse a consisté en l'étude d'architectures de réseaux de neurones impulsionnels reposant sur des synapses memristives et des neurones en silicium. Les architectures étudiées par la simulation prennent en compte les contraintes physiques et électriques propres à une réalisation matérielle des neurones événementiels. En complément de ce travail exploratoire de simulations informatiques, le développement d'une plateforme dédiée pour la mise en œuvre de matrices de nanocomposants memristifs réalisées par des partenaires extérieurs a été accompli. Ces matrices de 81 × 10 nanocomposants memristifs ont pour vocation d'être utilisées soit pour réaliser des réseaux de neurones événementiels *all-to-all*, soit pour la caractérisation expérimentale de ces 810 composants afin d'affiner les simulations réalisées. Si les simulations ont permis de définir l'architecture du démonstrateur du projet européen ULPEC ainsi que de nouvelles règles d'apprentissage, la plateforme 81 × 10 a été développée et vérifiée expérimentalement et la démonstration d'apprentissage sur des matrices de composants memristifs est en phase de développement.

Mots-clé : Réseau de neurones matériels ; Memristor ; Calcul événementiel.

Title : Hardware implementation of event-based neural networks based on nanotechnologies.

Abstract : This thesis took place within the thematic cluster "hardware AI" of the Bioelectronics Group of the IMS Laboratory during the ULPEC European Research Project. The aim of this project was to create an integrated, intelligent, low-powered neuromorphic sensor using a hardware event-based network by combining innovative technologies such as an artificial retina (event-based camera), silicon neurons and ferroelectric memristors as synapses into one chip. The work carried out in this project involved the study of architectures of event-based neural networks using memristive synapses and silicon neurons. The architectures studied by simulation means take into account the physical and electrical constraints of a hardware implantation of event-based neurons. In addition to this exploratory work with computer simulations, the development of a dedicated platform for the use of a crossbar array of memristive nanodevices made by external partners has been achieved. Those 81 × 10 crossbar arrays of memristive nanodevices are intended to be used either for an event-based all-to-all neural network or for experimentally characterizing those 810 devices to refine the simulations performed. While the simulations have provided a basis for defining the architecture of the European ULPEC project as well as new learning rules, the 81 × 10 platform has been developed and experimentally verified, but has not yet produced a learning demonstration using a memristive nanodevice crossbar array as it is still under development.

Keywords : Hardware neural networks ; Memristor ; Event-based computation.

Unité de recherche

Laboratoire de l'Intégration du Matériau au Système, UMR 5218

Bâtiment A31, 351 Cours de la Libération 33405 TALENCE cedex, FRANCE.