N° d'ordre : 3261

THÈSE

présentée à

L'UNIVERSITÉ DE BORDEAUX I

ÉCOLE DOCTORALE DE SCIENCES PHYSIQUES ET DE L'INGENIEUR

par **Yannick Bornat**

POUR OBTENIR LE GRADE DE

DOCTEUR

SPÉCIALITÉ : ELECTRONIQUE

Réseaux de neurones sur silicium : une approche mixte, analogique / numérique, pour l'étude des phénomènes d'adaptation, d'apprentissage et de plasticité

Soutenue le : 1er décembre 2006

Après avis des rapporteurs :

M.	Jean-Marie Bilbault	Professeur	Rapporteur
M.	Pascal Nouet	Professeur	Rapporteur

Devant la commission d'examen composée de :

M.	Jean-Marie Bilbault	Professeur	Rapporteur
M.	Pascal Nouet	Professeur	Rapporteur
M.	Claude Pellet	Professeur	
Mme	Sylvie Renaud	Professeur	Directeur de thèse
Mme	Michelle Rudolph ..	Chargée de recherche cnrs .	
M.	Jean Tomas	Maître de Conférences	Co-directeur de thèse
M.	Thierry Viéville ...	Directeur de recherche inria	

– 2006 –

Merci

Je remercie le professeur André Touboul pour m'avoir accueilli au sein du laboratoire IXL pendant la réalisation de ces travaux et de mes études antérieures

Merci également au professeur Sylvie Renaud, pour m'avoir accepté dans son équipe et pour m'avoir encadré durant tout ce temps. Mais c'est surtout l'écoute, la compréhension, les discussions philosophiques, le chocolat et, plus que tout, la confiance en toutes situations qui auront laissé une trace tout ce temps.

Merci Jean Tomas, pour le soutient, la sereinité, l'humour, les discussions et la douce fragrance latino-asiatique qui ont parfumé le déroulement de ces travaux.

Je remercie bien sûr les professeurs Jean-Marie Bilbault et Pascal Nouet pour le temps qu'ils m'ont consacré en tant que rapporteurs de thèse.

Many thanks to Michelle Rudolph, you introduced me to the computational neurosciences and their concepts, your contribution influenced these works, your kindness and optimism influenced me.

Je remercie également Thierry Viéville et Claude Pellet pour l'intérêt qu'ils ont porté à mon travail en acceptant de participer au jury de thèse.

Une pensée amicale pour Sylvain Saïghi, qui a su me recevoir dans l'équipe, m'a montré la route à suivre, les pièges à éviter et a dissipé mes doutes dans les mauvais moments. Merci beaucoup.

Colin Lopez figure ici en bonne place. Notre collaboration fut très constructive et enrichissante, l'éxigence des travaux de thèse ne m'a pas permi de t'accorder l'attention que tu méritais.

Un grand merci à toute l'équipe *neurones* de l'IXL, travailler avec vous est un vrai plaisir, puissent les habitudes de l'équipe perdurer encore longtemps. Vivent les chouquettes et le chocolat.

Une pensée aussi pour tout ceux qui, directement ou non, ont contribué à ces travaux. Qu'ils soient chercheurs dans d'autres laboratoires comme Alain Destexhe, ou stagiaires de passage dans l'équipe comme Guillaume, Hicham et Katarina que j'ai encadré.

D'une façon plus générale, j'aimerais remercier l'ensemble du personnel du laboratoire IXL pour leur accueil, plus particulièrement les services administratifs et techniques pour leur aide au quotidien.

Comme pour les colonies de vacances, merci papa, merci maman : je n'aurais rien fait de tout ceci sans votre soutien.

Finalement, merci pour votre compréhension, vous pour qui j'étais si peu disponible durant ces trois dernières années : Aurélie, Aurore, Charlotte, Caroline, Caroline, Delphine, Delphine, Denis, Emeline, Eric, Estelle, Estelle, Fabien, France, Frédéric, Gaël, Guillaume, Jean, Jean-Phi, Julien, Nicolas, Olivier, Patrick, Régis, Rémi, Romain, Salif, Stéphane, Yann et Yann.

Table des matières

Introduction

La compréhension des phénomènes dont le cerveau est le siège devient possible grâce à la contribution des neurosciences. Elles nous permettent, entre autres, d'isoler les différents processus impliqués dans l'apprentissage, l'adaptation face à un environnement inconu, ou encore, la circulation d'information au sein d'un organisme. L'une des approches d'étude de cette discipline est d'étudier l'élément de base, le neurone, pour remonter peu à peu à l'échelle du groupe, de la structure puis de l'organe. Elle nécessite donc de bien identifier le comportement des neurones, voire des compartiments les composant, avant de monter dans les niveaux de description supérieurs.

Malgré une culture historiquement expérimentale, la nécessité d'utiliser des modèles mathématiques s'est imposée comme une approche complémentaire incontournable depuis une dizaine d'années. Ces modèles permettent la simulation des cellules nerveuses et évitent le recours à des cellules vivantes dont l'usage est relativement lourd, sinon parfois impossible. La formulation d'une activité permet également d'améliorer la communication et la mutualisation des efforts de recherche entre laboratoires.

Les modèles établis présentent chacun un comportement plus ou moins fidèle et un coût en conséquence au niveau de la puissance de calcul requise [Izh04]. Les descriptions à l'échelle du neurone exigent bien entendu le plus de ressources. Quant aux modèles plus pauvres, ils ne sont intéressants que pour des réseaux étendus. Dans les deux cas, le nombre d'opérations requises pour la simulation est considérable. Ce constat a abouti à la conception de circuits intégrés neuromorphiques analogiques, alternative de calcul aux simulateurs numériques présentant des propriétés bien spécifiques (calcul concurrent, variables et temps continu) [Mah91].

La conception de systèmes analogiques pour la simulation de l'activité neuronale a réellement permis de choisir l'échelle de temps utilisée pour la simulation. Les modèles les plus précis se sont orientés vers une implantation en temps réel pour s'interfacer avec des cellules vivantes selon la technique du *dynamic clamp* [Sha93] ou des réseaux hybrides [LeM95]. Il est ainsi devenu possible de réaliser des mesures de réseaux de neurones contenant des parties artificielles aisément configurables et contrôlables.

Désormais, les besoins en simulation augmentent. Il est devenu intéressant de conduire des simulations sur des réseaux de neurones, tout en gardant une fidélité très élevée par rapport aux phénomènes biophysiques. Les deux principales applications sont l'élaboration de réseaux hybrides, mélangeant neurones vivants et artificiels en grand nombre, et l'étude de phénomènes complexes, tels que la plasticité sur des réseaux de taille limitée, en lien éventuel avec des simulations de réseaux très importants utilisant des modèles plus simples.

Les travaux présentés dans cet ouvrage sont dédiés au développement de simulateurs de réseaux de modèles neuronaux très fidèles. Pour ne pas perdre l'apport de la complexité des modèles, la structure proposée se devra de n'accorder qu'un minimum de concessions aux contraintes issues de la nature électronique temps réel du système. Cela signifie, entre autres, que chaque opération doit être temporellement maîtrisée de bout en bout.

La simulation d'un réseau de neurones est plus complexe que la simulation de l'ensemble des neurones qui le constituent. Bien que cette démarche en soit la base, il est également au moins nécessaire d'assurer et de gérer la communication entre chaque cellule. Finalement, le phénomène le plus exigeant reste l'évolution même de la structure du réseau. L'incessante fluctuation des liaisons entre les neurones, la plasticité, est à la base des phénomènes d'apprentissage dont nous devons permettre l'implantation dans les systèmes de simulation, et devient donc un centre d'intérêt privilégié des neurosciences computationnelles.

Nous chercherons ici à concevoir un système capable de simuler des modèles complexes, jusqu'alors uniquement résolus par des méthodes logicielles, en gardant le temps réel comme contrainte principale. Ce système servira de support à plusieurs études scientifiques : l'aspect analogique permet d'observer l'influence du caractère non déterministe du réseau par rapport aux systèmes numériques déterministes ; le fonctionnement en temps réel sera exploité pour la conduite d'expériences hybrides ; enfin, la puissance de calcul qu'il représente sera utile pour évaluer l'influence du modèle neuronal à l'échelle du réseau sur des séries de simulations comparatives.

Cette thèse est la sixième réalisée dans l'équipe *Ingénierie des Systèmes Neuromorphiques* du laboraloire IXL depuis celle de Denis Dupeyron [DUP98] en 1998. Elle bénéficie donc de toute l'expérience nécessaire dans le cadre de la simulation analogique des neurones, ainsi que pour la réalisation de systèmes pour les réseaux hybrides. Les différents partenaires, essentiellement l'UNIC (Unité de Neurosciences Intégratives et Computationnelles) de l'institut Albert Fessard à Gif sur Yvette, disposent, eux, de la culture requise pour le développement de modèles biophysiques de neurones et réseaux de neurones, ainsi que pour la conduite de simulations de réseaux neuronaux.

Cet ouvrage s'articule en trois chapitres de description des travaux réalisés après un premier chapitre d'entrée en matière sur le sujet.

Plus particulièrement, le premier chapitre s'intéresse aux systèmes de simulation de réseaux de neurones, ainsi qu'à leurs méthodes de traitement de l'aspect temporel dans leurs simulations. Il présente également les notions fondamentales de neurobiologie, nécessaires à la compréhension des contraintes auxquelles nous serons soumis. Une grande part est consacrée aux modèles utilisés pour pouvoir appréhender la complexité visée. Bien entendu, un tour d'horizon des simulateurs et de leurs applications amènera les références nécessaires pour situer les travaux de l'équipe en général, et ceux décrits ici en particulier, au sein de la communauté scientifique internationnale.

Le deuxième chapitre présente la réalisation d'une première génération de simulateur, qui est à la fois destinée à la simulation de petits réseaux et à une démonstration de faisabilité. Les choix stratégiques sur les méthodes de résolution des modèles de

réseaux de neurones sont abordés. Le circuit neuromimétique conçu pour ce système est ensuite décrit, ce qui permet d'enchaîner sur la réalisation effectuée. Une analyse est enfin menée, tant sur les résultats de simulation eux-mêmes que sur l'adéquation du système par rapport à la conduite de simulations ainsi que sur son évolutivité.

Dans le troisième chapitre, un élargissement de la structure du système et une redistribution des charges de calcul sont étudiés pour augmenter ses performances. Un passage est consacré aux fonctions que met à l'épreuve l'augmentation du nombre de neurones. Le circuit utilisé comme base analogique de ce système est évoqué. Après une présentation de la réalisation effectuée, une étude critique est menée pour évaluer la pertinence des choix proposés.

Le dernier chapitre s'intéresse à la conception d'une troisième génération de simulateur. Une proposition est effectuée pour assurer une longévité et une polyvalence maximales à la structure, tant en terme de performances que de polyvalence. Cette proposition s'appuie sur l'expérience de simulation des deux systèmes précédents. Elle fait aussi appel à certaines considérations théoriques qui sont exposées et démontrées. Nous aborderons ensuite la conception du premier circuit intégré neuromimétique destiné à être monté sur cette dernière version de simulateur. Les choix portant sur cette réalisation utilisent eux aussi le bilan des deux premières générations. Le chapitre se termine par la phase de caractérisation du circuit et les perspectives qu'il ouvre.

La conclusion de cet ouvrage s'effectuera par une évaluation du travail restant afin d'exploiter au mieux le dernier système en chantier, ainsi que par une discussion sur les perspectives qu'ouvrent ces travaux.

Chapitre 1

Des circuits et des hommes : du vivant à la simulation

Avant de s'intéresser à la conception de simulateur de réseaux de neurones, il est nécessaire de s'attarder sur les concepts inhérents aux neurosciences computationnelles. Les travaux réalisés jusqu'ici nous apportent une aide considérable. Ils nous permettent d'établir une hiérarchie dans l'ensemble des phénomènes observés en fonction de leur impact sur le comportement global de la simulation. Il est alors possible d'effectuer des choix parmi les modèles en fonction des intérêts et des possibilités techniques et/ou technologiques.

Nous nous intéresserons d'abord aux concepts liés à toute simulation, aux compromis qui en découlent, mais aussi à son adéquation avec une problématique prédéfinie. Nous verrons ensuite une introduction sur le fonctionnement du système nerveux : les phénomènes temporels dont il est le siège, et les éléments qui le constituent. De la présentation de ces éléments à leur modélisation, il n'y a qu'un pas, que nous n'hésiterons pas à franchir, poussés par la curiosité et l'envie de répondre à toute question qui se présente. Il sera alors possible de s'intéresser à différentes approches de la simulation de neurones dans le cadre des neurosciences computationnelles. Une fois positionnés par rapport aux autres acteurs de ce sujet, nous pourons clarifier nos objectifs et arrêter le choix des modèles à implanter.

1.1 Supports et conditions de simulation

Simulation, émulation, imitation, tout le monde aura compris que nous nous positionnons par rapport à l'idée de reproduire un comportement. Ce concept est utilisé dans de nombreux domaines de la physique moderne lorsqu'une mise en œuvre immédiate est problématique. Mais au-delà de cette vague idée, interrogeons-nous sur les formes que peuvent prendre les simulations. Explorons le niveau d'intérêt qu'elles apportent par rapport à leurs exigences techniques.

1.1.1 La simulation logicielle

La simulation logicielle fut à l'origine du développement des premiers calculateurs dont l'ENIAC [GOL46] est le plus célèbre. Avec le recul, ce dernier était plutôt un ensemble d'éléments de logique booléenne. Il fallait les assembler pour effectuer un calcul. C'est bien évidemment le cablage des éléments entre eux qui déterminait

l'équation calculée au final. Avec l'augmentation de la puissance de calcul [1], il a été possible de passer du calcul numérique d'un résultat à une recherche plus abstraite des solutions. Le *résultat* n'est donc plus obligatoirement une solution, mais peut désormais prendre la forme d'une équation. De nombreuses simulations sont aujourd'hui menées sur tous types de phénomènes physiques mais aussi sociaux.

Nous avons dépassé le niveau de la simple résolution de systèmes d'équations. La simulation est l'œuvre d'un algorithme dont l'équation n'est que le cas particulier. Le calcul de la simulation est généralement exécuté par un ordinateur ou un supercalculateur, au point que l'on en oublie qu'il puisse en être autrement. L'usage d'ordinateurs apporte en effet une grande souplesse d'utilisation : ils sont facilement programmables et reprogrammables. L'usage de langages communs et de protocoles ou formats ouverts permet de plus un brassage et une mutualisation des efforts. Il est désormais possible de résoudre tout problème dont on puisse exprimer une solution, et dont le concept soit à la portée du programmeur. La seule limite est le temps que nécessitera le processus de résolution.

1.1.2 La simulation matérielle

Pour maîtriser le temps de calcul, une démarche est apparue, celle de dimensionner un système de calcul par rapport à la charge du problème à traiter. C'est ainsi qu'apparaissent les premiers systèmes basés sur des DSP (Digital Signal Processor) [Pel76] [Nis81]. Bien que la simulation soit toujours numérique, la philosophie est nouvelle : on choisit l'échelle temporelle de simulation (1/1 pour un DSP) et le reste du système est dimensionné en fonction de cette contrainte. Les systèmes utilisant des FPGAs suivent la même idée directrice et permettent de retrouver la souplesse des simulations logicielles. Le FPGA correspond finalement à l'idée d'un ENIAC dont la configuration est automatisée [2].

Cependant, pour les systèmes dont la complexité est encore supérieure, la solution numérique n'est plus envisageable. Les solutions mises en œuvre se retournent alors vers le calcul analogique, comme à l'époque où le numérique était insuffisant (voire inexistant) [Mal33]. Le principe de la simulation analogique n'est pas véritablement de programmer des équations. Il s'agit d'une méthode qui consiste à concevoir un système dont la loi d'évolution obéit aux mêmes équationis que le phénomène à simuler, tout en étant plus abordable du point de vue de l'exploitation. Ainsi aux opérateurs d'addition, de multiplication ou de dérivation en mathématiques, correspondent des montages additionneur, multiplicateur ou dérivateur en électronique analogique. La conception du simulateur consiste donc à rechercher et exploiter les propriétés des composants qui correspondent à l'opération à effectuer. De nombreux degrés de libertés sont possibles à ce niveau : il est par exemple possible de s'imposer une échelle temporelle comme nous le verrons au paragraphe 1.1.3. Pour un tel système, les performances et les capacités d'extensions sont connues dès la conception. Il devient ensuite très difficile de le faire évoluer au fil des innovations. De plus, s'il possède des avantages certains, l'investissement nécessaire à son développement est important. Le

[1] L'ENIAC était, en 1944 et 1946, le calculateur le plus puissant du monde. Compte tenu des mensurations des systèmes visant aujourd'hui ce titre, nous ne parlerons pas d'intégration. Au mieux, nous pouvons faire mention de l'usage de circuits intégrés

[2] Nous signalerons au passage que la taille, la consommation, la fiabilité et la puissance de calcul ne souffrent aucune comparaison ; il faut bien que ces quelques quarante ans qui les séparent aient servi à quelque chose.

tableau 1.1 reflète une comparaison des modes de simulation extrêmes : analogique matériel et numérique logiciel.

Simulation	Logicielle	Analogique
Temps de calcul	Moyen	Très Bon
Reconfigurabilité	Très Bon	Moyen
Coût de développement	Bon	Mauvais
Ouverture aux autres systèmes	Bon	Mauvais
Réalisme	Moyen	Très bon

TAB. 1.1 – Comparaison des systèmes de simulation

1.1.3 Aspect temporel d'une simulation

Généralement, l'aspect temporel d'un simulateur se limite uniquement à son échelle de temps. Il s'agit du rapport entre la durée nécessaire au processus de simulation pour fournir un résultat (que nous appellerons *temps de simulation*) et le temps que dure le phénomène simulé (*temps simulé*). Dans certaines conditions, il est imposé par l'environnement de simulation lui-même. Le temps de simulation était jusqu'ici une carcatéristque d'un système et servait à en évaluer les performances. Il devient désormais une contrainte à remplir pour obtenir un résultat correct. Cette situation se rencontre lorsque l'on a besoin d'une interaction entre des systèmes de natures différentes. Dans le cas qui nous intéresse (la simulation de neurones), cette contrainte temporelle apparaîtra lors de l'utilisation d'un système analogique ou vivant [SHA93],[LEM02]. Un simulateur logiciel ou numérique peut en effet gérer des attentes lorsqu'il lui manque des informations, ce dont est incapable un système évoluant de façon analogique.

L'échelle de temps n'est cependant qu'une caractéristique globale de la simulation, elle ne précise en rien l'aptitude du système à réagir en un temps satisfaisant dans toutes les situations. La figure 1.1 illustre bien la différence entre une échelle de temps globale de (1/1) et ce qu'il convient d'appeler le *temps réel*. Sur un phénomène quelconque, elle nous montre la progression hypothétique de deux simulateurs. Le simulateur (1) est un simulateur temps réel, les temps simulé et de simulation sont donc égaux en permanence. Le simulateur (2) pour sa part, fournit un résultat plus rapidement : il simule avec une échelle de temps de $0,8/1$. Cependant, on remarque une baisse de performances au temps simulé t_1. En cet instant, la tangente de la courbe est inférieure à 1. Ceci signifie que, localement, le simulateur ne peut pas fonctionner aussi vite que le phénomène simulé. Bien qu'il soit globalement plus performant, il n'est donc pas possible d'effectuer de simulation temps réel avec le simulateur (2). Pour pouvoir fonctionner en temps réel, il faut que le temps simulé soit supérieur ou égal au temps de simulation quels que soient l'échelle d'observation et l'évènement simulé. En langage plus mathématique, pour qu'un système soit qualifié de temps réel, le rapport entre le temps simulé et le temps de simulation doit appartenir à l'ensemble des fonctions 1-lipschitziennes, c'est à dire, avoir une dérivée temporelle minorée par 1.

La définition précédente n'est hélas pas compatible avec la vision séquentielle des systèmes numériques. En effet, ces derniers utilisent un système d'échantillonnage et

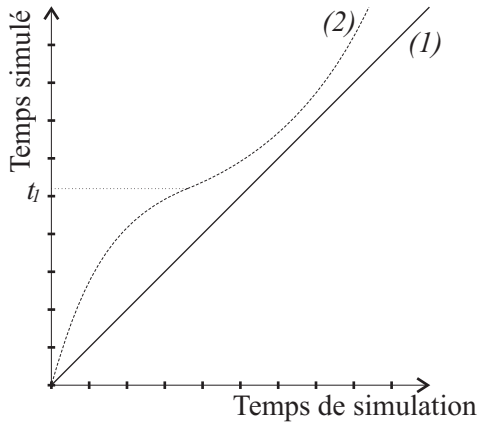

FIG. 1.1 – Évolution du temps simulé par rapport au temps de simulation

introduisent, de ce fait, des discontinuités dans les signaux. Il existera donc toujours, dans les systèmes numériques, un temps au-dessous duquel il ne sera pas possible de réagir. Ce temps est supérieur ou égal à la période d'échantillonnage. Le temps réel sera alors associé à la notion de temps de calcul. Ainsi, un simulateur considéré *temps réel à la microseconde près* aura, dans le pire cas, une microseconde de retard par rapport aux phénomènes qui l'entourent. Dans le cas de signaux analogiques, nous parlerons alors de *temps réel continu.*

Si la simulation temps réel est réalisée par logiciel sur un ordinateur muni d'un système d'exploitation, ce dernier doit, lui aussi, répondre aux contraintes du temps réel. Cela se traduit par un temps de réaction de la part du système majoré de façon absolue. Cette majoration est alors comptée comme partie du temps de calcul, et doit être ajoutée à la durée effective du calcul.

Il nous faut maintenant aborder une dernière notion sur les contraintes de temps d'un simulateur. Nous avons vu que la cohabitation d'un système numérique et analogique implique des difficultés propres au temps réel, et ce, quelle que soit l'échelle de temps choisie. Nous dirons alors, après avoir précisé cette échelle de temps, qu'il s'agit d'un système *à contraintes de temps réel.* Ce qui signifie un garantie sur le déroulement temoprel

Exemple de définition

A titre d'exemple, dans les travaux de Sylvain Saïghi [SAÏ04] ,le circuit *Pamina* est capable de simuler de façon analogique le comportement d'un neurone à différentes échelles de temps selon les réglages. Dans le cas le plus rapide, il est fait mention de *cent fois le temps réel.* Il s'agit finalement d'un système de simulation ayant une échelle de 1 : 100 à contraintes de temps réel continu. Ce circuit est contrôlé par un FPGA et contient deux neurones. Il serait alors possible de simuler un réseau avec des techniques mixtes. La fréquence de ce FPGA est de 32MHz, ce qui permet de garantir un temps de réaction inférieur à la microseconde. Nous aurions alors un simulateur mixte, avec une échelle de temps de 1 : 100, ayant une contrainte de temps réel à 100 µs près (1 µs de calcul correspond à 100 µs de temps simulé).

1.2 Bases de neurobiologie

Dans cette partie, nous ne chercherons pas à expliquer dans le détail le fonctionnement du système nerveux. Il s'agit plutôt de présenter les éléments qui le constituent, ainsi que les phénomènes qui nous intéressent pour le sujet.

1.2.1 Le système nerveux

Le système nerveux est l'ensemble des organes permettant aux animaux de percevoir leur environnement et d'y réagir de façon adaptée. Il gère le système sensoriel, analyse son environnement et construit une expérience en mémorisant et structurant ces analyses.

Plusieurs approches peuvent être utilisées pour le décrire. Une approche fonctionnelle cherche à définir dans quelles parties sont effectuées telle ou telle opération. Certaines zones ont ainsi été isolées sur différentes espèces, dont l'homme, pour décrire les zones motrices, de réflexes ou de mémorisation. Une approche anatomique tentera de définir des organes différents, par leur structure, leur position ou leur formation avant la naissance. De notre côté, et parce que nous nous situons au niveau de la cellule, le système nerveux sera considéré comme une formidable structure pouvant contenir, dans le cas de l'homme, quelques 10^{14} neurones, soit cent mille milliards, tous interconnectés en réseaux.

1.2.2 Les réseaux

Le système nerveux ne se limite pas au cerveau que l'on rencontre chez les vertébrés. De même, les opérations effectuées ne se limitent pas aux actes conscients ou à la pensée. En effet, la majeure partie des opérations effectuées par notre corps ne sont pas conscientes. Elles sont réalisées par des réseaux spécialisés.

Par exemple, quand l'acte conscient pour un individu sera de saisir un objet, un grand nombre de structures s'activent. Il faut en effet au moins :

- déduire l'ensemble des mouvements à effectuer ;
- activer les muscles nécessaires à chaque mouvement ;
- effectuer un contrôle permanent de l'avancement du geste ;
- assurer la succession cohérente des mouvements.

Cet ensemble de tâches est encore plus impressionnant si l'individu ne regarde pas l'objet à saisir puisqu'il a alors recours à ses facultés de mémorisation.

Former l'ensemble des réseaux nécessaires à la mise en œuvre de ces fonctions est le rôle de l'apprentissage. Les réseaux se créent grâce à la propriété qu'ont les neurones de toujours chercher à se connecter à d'autres. Un grand nombre de connections sont régulièrement créées. Selon qu'elles contribuent ou non à l'augmentation des performances d'un réseau, ces connexions seront renforcées, ou sont vouées à s'atrophier et à disparaître. Elles peuvent avoir une influence variable sur l'ensemble du réseau, c'est pourquoi on parle du *poids* qu'elles peuvent avoir. Les fluctuations du poids des connections sont des phénomènes qui se produisent au niveau de la *synapse* [Ito94], partie d'un neurone situé à une de ses extremités où se déroule l'ensemble des processus liés à la transmission d'information.

1.2.3 Les synapses

Les synapses peuvent être de deux types : synapses chimiques et synapses électriques. Ces dernières fonctionnent grâce à une connection électrique directe entre le neurone qui ememt l'activité ou *neurone présynaptique* (auquel appartient la synapse) et le neurone qui la reçoit ou *neurone postsynaptique*. La conséquence de ce lien direct est que la transmission de l'activité est systématique. Le comportement du neurone postsynaptique recopie en quelque sorte celui du neurone présynaptique, aux retards de propagation près.

Les synapses chimiques, si elles transmettent de l'information, ne transmettent pas une activité. Autrement dit, l'activité du neurone présynaptique influe sur le comportement du neurone postsynaptique, sans toutefois générer la même activité. Elles sont le siège de phénomènes relativement complexes, dont la principale conséquence est une connectivité sans cesse changeante au sein des réseaux [Ito94].

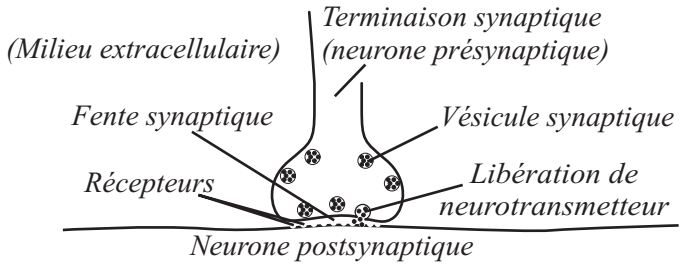

Fig. 1.2 – Schéma descriptif de la synapse

La figure 1.2 représente les principaux éléments constituant une synapse. On peut distinguer les milieux *intracellulaire présynaptique*, *intracellulaire postsynaptique* et *extracellulaire*. On distingue également des *vésicules synaptiques* contenant des *neurotransmetteurs*. Elles sont conçues sur les parois de la synapse et migrent vers la zone de transmission où elles attendent l'activité du neurone présynaptique. La transmission s'effectue par la libération dans la *fente synaptique* des neurotransmetteurs. Ils sont captés au niveau de la membrane du neurone postsynaptique par des récepteurs, lesquels forcent le passage d'ions sodium Na^+ de l'extérieur vers l'intérieur du neurone postsynaptique. Cette migration perturbe localement l'équilibre électro-chimique sur la membrane du neurone postsynaptique. Cette perturbation contribuera, ou non, à une activité de ce neurone selon sa nature, excitatrice ou inhibitrice.

Mais une synapse n'effectue pas qu'une simple transmission d'activité en seul rapport à sa force, elle possède sa propre histoire qui influera sur l'efficacité de la transmission. Par exemple, le renouvellement des vésicules de neurotransmetteurs est un processus relativement long à l'échelle du neurone. Si elle a été activée récemment, une synapse aura alors une capacité de transmission beaucoup plus faible. De plus, l'ouverture des vésicules n'est pas déterministe : seule une fraction d'entre elles réagit à une stimulation donnée. On observe également des cas où, sans raison apparente, la synapse ne se déclenche pas.

Par contre, dans le cas d'une connexion très sollicitée, on observe que la liaison entre deux neurones est assurée par plusieurs terminaisons synaptiques. On rencontre des connexions effectuées au moyen de plusieurs centaines de synapses. Cette redondance, si elle renforce la transmission, a surtout pour conséquence d'augmenter la

prédictibilité en raréfiant les non déclenchements.

Finalement, la force des synapses est sujette à fluctuation. Selon l'activité, une synapse donnée aura tendance à se renforcer ou à perdre en intensité. Ce phénomène, la *plasticité*, est à l'origine des variations observable dans un réseau de neurones. La simulation de certains algorithmes de plasticité sera le principal intérêt des systèmes dont il sera question dans cet ouvrage.

1.2.4 Les neurones

Les variations de concentrations ioniques observables lorsqu'un neurone est actif sont dues au principe de fonctionnement même des neurones. En effet, leur activité est donnée par rapport à leur *potentiel de membrane*, c'est-à-dire la différence de potentiel entre les milieux intra- et extracellulaire. Les variations de ce potentiel, et donc l'activité du neurone, sont le résultat de transferts ioniques à travers la membrane. Cette activité se présente sous la forme d'impulsions brèves qui se répètent de façon plus ou moins régulière : les *potentiels d'action*. Il fonctionnent selon le principe du tout ou rien.

Comme il a déjà été mentionné, les variations de potentiel sont générées par des déplacements d'ions. Cette propriété est rendue possible par des protéines présentes dans la membrane. Comme le montre la figure 1.3, les unes, les protéines *transmembranaires*, laissent passer sélectivement les ions sodium (Na^+) ou potassium (K^+), les autres cherchent en permanence à rétablir l'état d'origine en forçant la sortie des ions sodium, et l'entrée des ions potassium. Ce mécanisme est qualifié de *pompage actif* [Tri98].

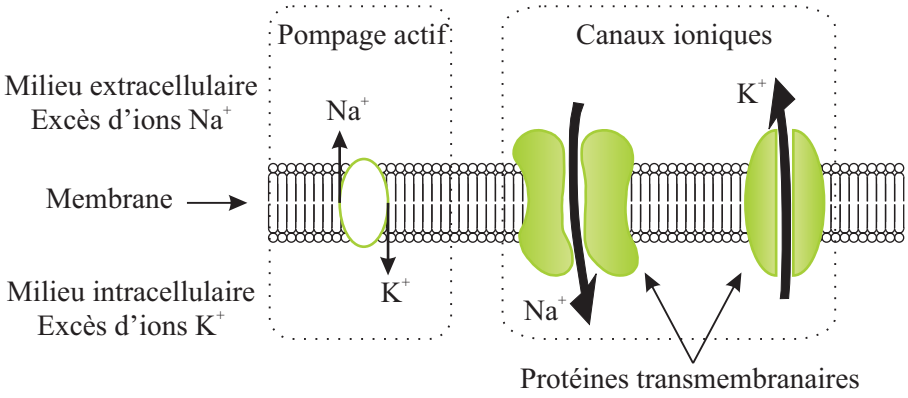

Fig. 1.3 – Représentation des canaux ioniques présents sur une membrane de neurone.

Le potentiel d'action naît de l'équilibre (ou non) entre les potentiels pour lesquels les protéines transmembranaires sont actives, ainsi que leur inertie à l'ouverture et/ou à la fermeture. L'injection d'ions sodium au niveau de la synapse afférente augmente donc la différence de potentiel entre les milieux intracellulaires et extracellulaire également appelée *potentiel de membrane*. Cette variation se propage, et est éventuellement additionnée à une autre stimulation. Au-delà d'un certain seuil, les protéines liées au sodium s'ouvrent et le potentiel de membrane devient alors positif ; on dit qu'il est *dépolarisé*. À ce stade, les protéines associées au potassium s'ouvrent également et rétablissent un potentiel de membrane négatif. Les protéines sodium

sont maintenant fermées et le potentiel de membrane est au plus bas, le neurone est *hyperpolarisé*. Les protéines potassium se ferment à leur tour ; le pompage actif rétablit finalement une situation propice au traitement de nouvelles entrées.

Ce phénomène parcourt la membrane du neurone de proche en proche. Il parviendra alors jusqu'aux synapses pour être transmis aux autres neurones. La phase d'hyperpolarisation durant plus longtemps que celle de dépolarisation, il n'y a pas de phénomènes de réflexion de l'onde aux extrémités du neurone.

Par contre, la collecte des entrées, la génération du potentiel d'action et la propagation s'effectuent chacune en des zones bien précises du neurone. La figure 1.4 montre l'anatomie d'un neurone classique. Nous pouvons distinguer l'arbre dendritique, le corps cellulaire ou *soma*, l'axone et les terminaisons synaptiques.

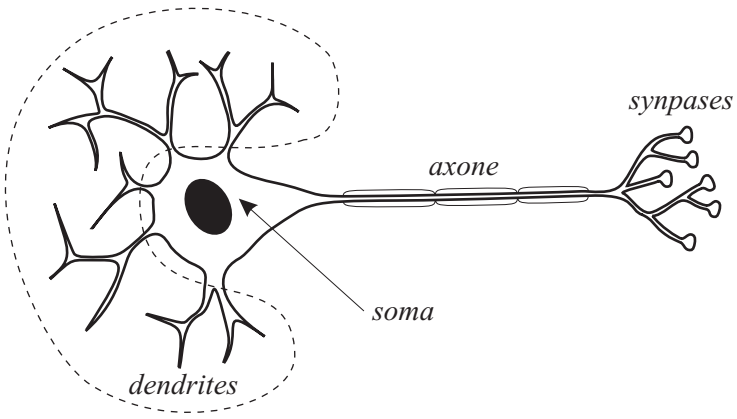

FIG. 1.4 – Anatomie du neurone

L'arbre dendritique, dont les ramifications ou *dendrites* sont très variables en nombre et en longueur, collecte les entrées sur les différentes synapses. Il ramène leurs contributions jusqu'au soma en les combinant, si nécessaire, au niveau des noeuds [TRI98]. C'est au niveau du soma qu'apparait le potentiel d'action et que s'effectuent tous les mécanismes liés à la survie et à l'entretien de la cellule. Il est muni d'une prolongation qui véhicule les potentiels d'action : l'*axone*. Ce dernier a une longueur très variable qui va de quelques micromètres jusqu'au mètre. Selon le neurone, l'axone peut être entouré d'une gaine de myéline. Celle-ci a la propriété d'accélérer la transmission par la regénération d'un potentiel d'action aux *noeuds de Ranvier* (petites portions nues de l'axone). L'axone arrive finalement aux synapses que nous avons vues précédemment. Toutes les synapses d'un neurone ont des propriétés identiques, de sorte qu'à l'intensité près, l'information délivrée est la même.

1.2.5 Qu'est-ce que l'information ?

Nous venons d'achever une description rapide du système nerveux, sans pour autant aborder la nature des informations qui y transitent. Cette question est très vaste et couvre à elle seule tout un sujet des neurosciences. Il est cependant possible de présenter certains cas connus pour illustrer ce que cette information peut ou ne peut pas être. L'essentiel est ici d'avoir une idée du niveau de complexité qu'il est possible de rencontrer.

La première approximation est de considérer le potentiel d'action lui-même en

tant que portion d'information. Cependant, l'existence d'un potentiel d'action, à l'issue d'un stimulus donné, recèle une part aléatoire non négligeable. Il semble alors que cette hypothèse ne soit pas viable.

La succession logique serait d'associer une information au niveau d'activité moyen du neurone. Ce mode de fonctionnement se rencontre pour des stimulations musculaires. Il y a, en effet, une relation entre le niveau de contraction d'un muscle et la fréquence de fonctionnement des motoneurones qui lui sont associés, c'est-à-dire les neurones chargés de fournir la stimulation au muscle[ABB94].

De même qu'il existe des neurones sensibles aux variations de stimulation [IZH04], il est possible d'imaginer des cas où l'information est codée dans la modulation de la fréquence des neurones. Cependant, l'information n'est pas uniquement constituée d'un flot de potentiels d'action. Dans le cas de la mémoire, elle est bien plus distribuée à l'échelle du réseau. Elle est alors contenue dans la présence ou non de connections, ainsi que dans les poids synaptiques associés à chaque transmission. L'intervention des potentiels d'action ne devient alors qu'un moyen de stockage (puisqu'ils sont à l'origine de la plasticité) ou de consultation de cette information (la structure du réseau se révèle lorsqu'il est parcuru par une activité). Il est également inutile de chercher dans le cerveau une structure de mémorisation répondant à une logique mathématique telle que l'Algèbre de Boole. Il s'agit d'une structure générée par une longue évolution, et non d'un système synthétisé hiérarchiquement.

1.3 Les modèles utilisés

La présentation du système nerveux que nous venons d'avoir nous permet d'aprréhender qualitativement son fonctionnement. Il nous reste désormais à réaliser une approche au travers des modèles qui, à défaut d'expliquer les phénomènes, permet de mieux les décrires. Nous ne nous intéresserons ici qu'aux modèles de description temporelle des neurones, connus dans la littérature en tant que *spiking neurons*. Contrairement à précédemment, notre approche partira du niveau cellulaire pour remonter vers les réseaux. Ce qui nous permettra de présenter un modèle de synapse qui s'intègre à celui du soma.

1.3.1 Les neurones

Proposer d'entrée un modèle correspondant au *neurone* après en avoir montré la complexité semble un peu audacieux. Il s'agit en fait d'un abus de langage couramment employé. Dans une formulation précise, les modèles que nous allons présenter décrivent le potentiel de membrane en un point du soma.

Cet abus de langage vient de l'une des principales approximations du modèle : le neurone ponctuel. L'objectif majeur n'est pas ici de simuler le comportement d'un neurone particulier en fonction de son anatomie. Le neurone simulé est l'élément de base du réseau. Seules sont donc utiles les propriétés visibles à l'échelle du réseau. Ainsi, les phénomènes de propagation le long de la membrane, les distortions dans l'arbre dendritique ou la déformation du potentiel d'action le long de l'axone ne seront pas pris en compte. Le neurone est alors considéré comme une entité ponctuelle. Cette vision, dite du *point neuron* dans les communications usant de la langue de Shakespeare, marque l'hypothèse de base de la simulation de réseaux de neurones.

Le modèle *Integrate and Fire* du neurone

Le modèle *Integrate and Fire* est construit sur le principe de l'accumulateur. Les contributions de chaque neurone afférent sont additionnées algébriquement au potentiel de membrane. Les composantes positives étant prédominantes, le potentiel de membrane augmente petit à petit. Lorsqu'il franchit un seuil prédéterminé, on considère alors qu'il émet un potentiel d'action [GER02]. Les signaux afférents sont ignorés pendant la période réfractaire, puis le potentiel de membrane est réinitialisé à sa valeur de départ.

Outre sa simplicité de mise en œuvre, le grand attrait de ce modèle est l'absence de critère temporel. Par contre, il ne fournit pas d'informations sur le potentiel de membrane à proprement parler ; la seule information fournie est l'occurence d'évènements à un temps donné, ce qui est juste suffisant pour une simulation de réseau. Lorsqu'une représentation du signal est nécessaire, il faut faire appel à une représentation échantillonnée du potentiel d'action.

Paradoxalement, un simulateur analogique utilisant ce modèle ne pourra donc pas produire de représentation analogique du potentiel de membrane. Sa conception sera, par contre, simple du point de vue analogique : les stimulations sont transformées en quanta de courant injectés sur l'armature d'un condensateur. Les circuits nécessaires à la commande d'un tel circuit seront un comparateur pour la détection d'évènements, et un système de remise à la valeur initiale.

Ce modèle est très utilisé pour la simulation de réseaux de neurones formels, c'est à dire une structure algorithmique très souple à manipuler en informatique. Il recèle cependant des limites pour les neurosciences computationnelles. En effet, son absence de dimension temporelle l'empêche de différencier deux stimulations de durées différentes dès lors que le produit $poids \times temps$ est constant.

Le modèle *Integrate and Fire* avec fuite

Pour ajouter au modèle *Integrate and Fire* la notion d'une évolution temporelle, le retour vers un état d'origine a été ajouté. Cette notion se rapporte au pompage actif qui ramène en permanence les différents ions vers leur milieu d'origine. Désormais, le potentiel de membrane subit en permanence une perte en courant qui dépend de sa propre valeur, cette perte est assimilée à une fuite des charges à travers la membrane. Nous voyons alors apparaître une équation différentielle. Les liens avec l'électronique analogique s'affirment puisque ce modèle se présente sous la forme d'un schéma électrique présenté figure 1.5 [GER02].

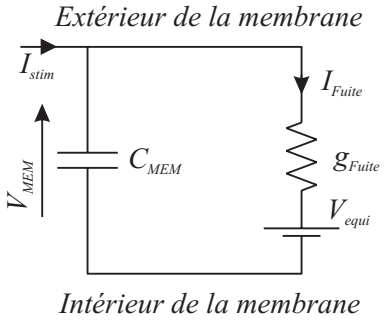

FIG. 1.5 – Le modèle *Integrate and Fire* avec fuite

Le potentiel de membrane est représenté par V_{mem}. La membrane est assimilée au diélectrique d'un condensateur. Ce modèle présente une analogie avec la réalité : en biologie, les ions s'accumulent de part et d'autre de la membrane tout comme, en électronique, les électrons s'accumulent de part et d'autre du diélectrique. La valeur de la capacité C_{mem} dépend de la surface de la membrane, et donc de la taille, du neurone modélisé. La notion de fuite est représentée par la conductance g_{fuite}. La source de tension qui produit V_{equi} définit le potentiel d'équilibre ou *de repos*. Lorsque $V_{mem} = V_{equi}$ le courant de fuite est nul. L'ajout de stimulations s'effectue alors par injection d'un courant I_{stim}.

Si ce modèle électrique décrit correctement le potentiel de membrane, il nécessite tout de même des montages périphériques identiques au modèle *Integrate and fire*. Son usage dans le cadre des neurosciences computationnelles devient néanmoins envisageable. Le compromis qu'il offre entre la fidélité du comportement et la puissance de calcul requise en fait d'ailleurs un modèle très utilisé pour la simulation de réseaux de taille significative [Liu04], [Fie04].

Le modèle *Hodgkin-Huxley*

Le modèle créé par Hodgkin et Huxley en 1950 a été développé selon une approche biophysique : il utilise une représentation des transferts ioniques à travers la membrane [Hod52]. Il est intrinsèquement plus précis puisqu'il vise à décrire les phénomènes à l'origine des fluctuations du potentiel de membrane. Il utilise également l'analogie électronique entre la membrane et un condensateur. Par rapport au modèle précédent qui simulait le pompage actif, il ajoute le fonctionnement des protéines transmembranaires. Leur contribution s'exprime sous la forme d'un canal représentant le flux régulier d'ions. Chaque canal ne se limite donc pas à simuler le comportement d'une protéine transmembranaire, mais de leur ensemble réparti sur la totalité de la membrane pour un ion donné. Il représente alors un comportement de population, avec la prise en compte de données stochastiques. Nous obtenons le modèle décrit par la figure 1.6.

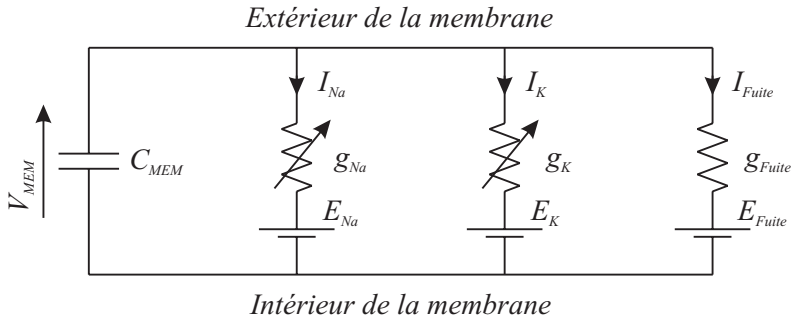

Fig. 1.6 – Le modèle *Hodgkin - Huxley*

La grande particularité de ce modèle, ainsi que sa complexité, viennent de la dépendance des conductances sodiques (g_{Na^+}) et potassiques (g_{K^+}) par rapport à V_{mem}. Elles sont régies par les équations 1.1 :

$$\begin{cases} g_{Na}(V_{mem}) = \overline{g_{Na}} \cdot m^3(V_{mem}) \cdot h(V_{mem}) \\ g_K(V_{mem}) \ = \overline{g_K} \cdot n^4(V_{mem}) \end{cases} \tag{1.1}$$

Où $\overline{g_{Na}}$ et $\overline{g_K}$ sont les conductances maximales que peut potentiellement représenter chaque canal. Les grandeurs $m(V_{mem})$, $h(V_{mem})$ et $n(V_{mem})$ représentent la réponse temporelle à l'ouverture ou à la fermeture des canaux, m et n sont les termes d'*activation* (ouverture de canal), et h est un terme d'*inactivation* (fermeture de canal). Ces grandeurs répondent à la même équation différentielle, donnée en (1.2)

$$\tau(V_{mem}) \cdot \frac{\partial x}{\partial t}(t) = x_\infty(V_{mem}) - x(t) \tag{1.2}$$

avec

$$x_\infty(V_{mem}) = \frac{1}{1 + \exp\left(s_f \frac{V_{mem} - V_{offset}}{V_{pente}}\right)} \tag{1.3}$$

Le paramètre binaire s_f représente le signe de l'expression contenue dans l'exponentielle. Il dépend de la nature de la fonction modélisée :

$$\begin{cases} s_f = 1 & \text{pour une inactivation} \\ s_f = -1 & \text{pour une activation} \end{cases} \tag{1.4}$$

Il ne nous reste qu'à définir $\tau(V_{mem})$, il est donné par l'équation 1.5.

$$\tau(V_{mem}) = \frac{x_\infty(V_{mem})}{\exp\left(a \cdot V_{mem} + b\right)} \tag{1.5}$$

Le gain en précision apporté par ce modèle peut s'illustrer à l'aide de deux cas pratiques : la temporisation au déclenchement et une réponse relative à une séquence. La temporisation au déclenchement correspond à un état évoluant lentement vers un potentiel d'action sans nécessiter de stimulation supplémentaire. La réponse relative à une séquence est la conséquence d'un plus grand nombre de variables d'état : le modèle fait ainsi une différence entre une stimulation inexistante et un ensemble de stimulations dont la somme algébrique est nulle [Izh04]. De plus, contrairement au modèle *Integrate and Fire*, le modèle de Hodgkin et Huxley donne une représentation réaliste du potentiel d'action.

Par contre, de par la nature des équations mises en oeuvre, ce modèle est plus complexe à implanter au niveau matériel. Nous avons vu dans la première section de ce chapitre que beaucoup d'opérations mathématiques sont envisageables. Cependant, les équations 1.3 et 1.5 donnant l'expression de $x_\infty(V_{mem})$ et de $\tau(V_{mem})$ sont plus délicates puisqu'elles font appel à des opérateurs mathématiques purs (inversion, exponentielle ...).

Ce modèle, dit biophysique, valut le prix Nobel de médecine 1963 à leurs créateurs. Il permet d'atteindre un très bon niveau de description du potentiel de membrane. Il y parvient en modélisant les principales fonctions ioniques qui agissent sur le comportement du neurone : le canal potassium, le canal sodium et le pompage actif. Depuis, bien d'autres espèces chimiques ont été identifiées avec, chacune, des effets négligeables à l'échelle du potentiel d'action, mais fondamentaux si l'on s'intéresse à l'évolution dite lente de l'activité du neurone (de l'ordre de quelques centaines de millisecondes).

Évolution vers un formalisme plus évolué

D'autres modèles ont été publiés en reprenant le principe des conductances ioniques. Les grandeurs caractéristiques sont alors adaptées à un autre type de neurone. Ce fut notament le cas pour le modèle de Morris-Lecar [Mor81].

Il est également possible d'ajouter des modèles de canaux ioniques supplémentaires à cette structure élémentaire. L'existence de canaux ioniques à cinétique lente ajoute une dimension supplémentaire à la simulation. En effet, désormais, le potentiel d'action ne constitue plus une remise à l'état d'origine de toutes les grandeurs. Il est alors possible de mettre en évidence des phénomènes d'accélération progressive (plateau calcique [Tri98]) ou d'adaptation et de ralentissement (modulation) alors que la stimulation est constante sur une longue période.

Cette constatation nous amène à considérer une évolution du modèle de Hodgkin et Huxley vers un *formalisme de Hodgkin et Huxley*. Ce formalisme consiste à représenter chaque contribution comme un canal ionique. On obtient alors une représentation semblable à la figure 1.7.

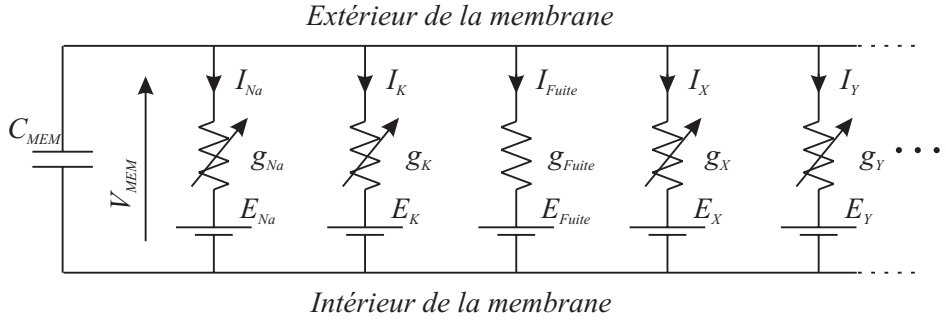

Fig. 1.7 – Le formalisme de *Hodgkin - Huxley*

La diversité offerte par ce formalisme vient de l'expression des g_{x_i} qui caractérisent les canaux ioniques. Pour les principaux d'entre eux, la conductance prend la forme donnée par l'équation 1.6.

$$g_x(V_{mem}) = \overline{g_x} \cdot m^p(V_{mem}) \cdot h^q(V_{mem}) \tag{1.6}$$

Dans cette expression générique, m représente une activation et h une inactivation. p et q sont des entiers et représentent les forces des cinétiques. Selon leurs valeurs, on retrouve les équations relatives aux canaux du modèle d'Hodgkin et Huxley :

$$\begin{cases} p = q = 0 : & \text{Canal de fuite} \\ p = 3, q = 1 : & \text{Canal sodium} \\ p = 4, q = 0 : & \text{Canal potassium} \end{cases}$$

Enfin, des phénomènes plus complexes ont été mis en évidence, avec la dépendance de certaines variables d'état aux concentrations ioniques. Nous pouvons par exemple voir un canal potassium dépendre de la concentration en ions calcium.

Le potentiel d'action : évènement ou état ?

Pour terminer cette description des modèles de neurones, il convient de souligner la différence de la nature du potentiel d'action selon le modèle utilisé. Avec l'usage de modèles *Integrate and Fire*, le potentiel d'action est un évènement généré par un circuit de contrôle, sa forme est donc totalement artificielle. Dans le cas d'un modèle suivant le formalisme d'Hodgkin et Huxley, le potentiel d'action est un état

temporaire. Par contre, il s'inclut dans le fonctionnement continu du modèle et il convient alors de le détecter.

La largeur d'un potentiel d'action est de l'ordre de la milliseconde. Il n'est donc pas possible de l'assimiler à un évènement ponctuel au sens de son traitement. La méthode consiste à choisir une forme de détection (passage d'un seuil, atteinte d'un maximum local, *et cætera*) puis à modifier le temps de propagation pour prendre en compte un éventuel retard.

Cette inconnue peut néanmoins être relativisée à l'échelle du réseau. En effet, l'erreur temporelle de détection d'un potentiel d'action est constante et uniforme [3] puisque la forme de ce dernier ne varie pas. Il n'y a donc pas de décorrélation observable entre des neurones ou des populations de neurones au sein d'un réseau.

Reste que, comme nous le verrons à la section suivante, nous ne connaissons pas non plus le temps de propagation minimal entre deux neurones au moyen d'une synapse chimique. Il ne faut donc pas oublier cette approximation. Pour le cas où l'erreur temporelle ici produite est supérieure au temps de propagation minimal observé en biologie, les résultats obtenus doivent subir un examen critique quant à la vitesse de transmission des informations.

1.3.2 Modèle de la synapse

Comme nous l'avons vu précédemment, tout phénomène agissant sur le potentiel de membrane peut être modélisé par un courant injecté sur l'armature d'un condensateur. Il en est donc de même pour les synapses. Nous arrivons cependant ici à un point subtil de la modélisation du potentiel de membrane.

Chaque neurone biologique possède des synapses qui sont des éléments lui permettant de transmettre son activité à d'autres neurones. Elles doivent donc être considérées comme des vecteurs de sortie du potentiel de membrane, et ne le modifient en rien. Par contre, l'activité simulée est hautement dépendante des synapses appartenant aux neurones afférents.

L'usage est donc d'intégrer au modèle du potentiel de membrane, celui des synapses afférentes. L'abus de langage consistant à assimiler la simulation d'un neurone avec celle de son soma ou de son potentiel de membrane, nous amène alors à une grande confusion. En effet, tant en biologie qu'en simulation, les synapses sont qualifiées de *synapses du neurone*. Seul le contexte scientifique permet de différencier les synapses appartenant au neurone biologique des synapses afférentes à un potentiel de membrane simulé (les synapses présentes dans le *modèle*).

Par la suite, nous ne représenterons les neurones que par rapport à leur modèle. Ainsi, s'il n'y a pas de précision, les synapses d'un neurone désigneront les synapses qui influent sur son comportement : celles qui lui sont afférentes. La figure 1.8 représente les éléments tels qu'ils sont regroupés au sein d'un modèle de simulation.

Ce type de rapprochement des éléments permet de limiter les échanges entre neurones à de simples évènements : l'existence ou non d'un potentiel d'action. L'axone du neurone afférent, ainsi que l'arbre dendritique du neurone simulé sont intégrés au modèle de la synapse. L'axone sera alors modélisé comme un retard dans la transmission du potentiel d'action du neurone afférent vers l'entrée synaptique relative au neurone simulé.

[3] la notion de constance se rattache ici à une invariance temporelle, alors que l'uniformité traduit une invariance d'un neurone par rapport à un autre

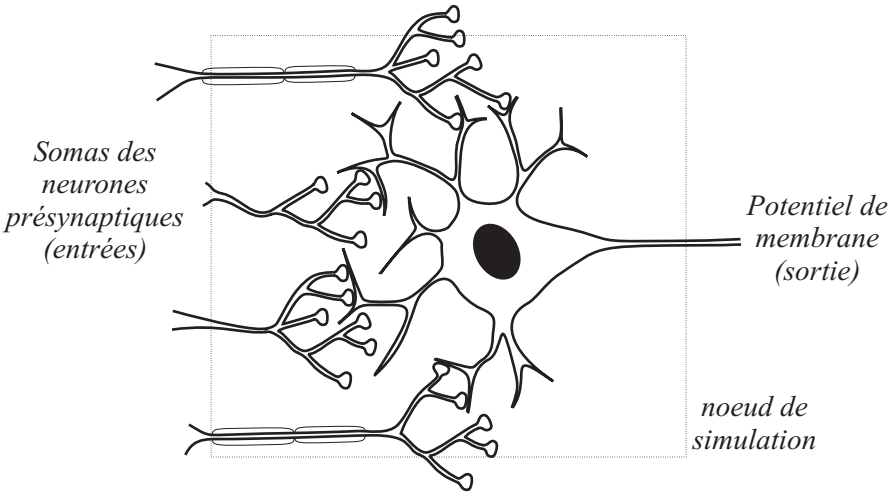

FIG. 1.8 – Répartition des éléments biologiques dans le modèle d'un neurone

L'intégration de l'arbre dendritique dans le modèle d'une synapse est légèrement plus problématique. En effet, la combinaison de plusieurs contributions synaptiques n'est pas linéaire [ELI93]. En d'autres termes, l'équivalent de deux contributions synaptiques, n'est pas le résultat de leur somme pris au sens mathématique. Il est donc possible de prendre en compte les distorsions dues à la propagation d'une contribution des dendrites jusqu'au soma. Il faut pour cela introduire un temps de retard et une altération du poids synaptique. Par contre, gérer la combinaison de deux contributions synaptiques ne peut se faire au niveau d'une synapse isolée ; nous reviendrons sur ce phénomène à la fin de cette partie. Cependant, les non-linéarités observables sont suffisamment faibles pour les négliger tout en gardant une bonne fidélité de simulation.

La contribution synaptique

Nous l'avons déjà vu, le rôle d'une synapse afférente est d'influer sur le potentiel de membrane. En gardant le même formalisme que précédemment, modifier le potentiel de membrane consiste à injecter un courant sur la capacité de membrane. Ce courant correspond au déplacement ionique observable au niveau de la fente synaptique.

Comme pour les canaux ioniques du soma, il est possible de modéliser une synapse avec une conductance variable et un potentiel de repos. Cependant, la conductance ne dépend plus ici du potentiel de membrane du neurone simulé, mais des potentiels d'action du neurone présynaptique.

La charge totale injectée représente la force de la synapse. Son signe quant à lui, indique si la synapse sera de nature excitatrice ou inhibitrice. Cependant, dès que le modèle du neurone contient un canal de fuite, le paramètre temporel prend de l'importance. Selon la précision désirée, il faut soit utiliser de brèves impulsions pour limiter les pertes, soit respecter l'évolution temporelle du courant injecté observable en biologie.

Synapse à décroissance

L'injection de courant est la conséquence indirecte de l'ouverture de vésicules de neurotransmetteur. Ces vésicules ont un temps de réponse variable mais des propriétés stochastiques définies. En considérant le comportement des vésicules comme celui d'une population, on obtient le modèle donné par l'équation 1.7 ou 1.8 basé sur une décroissance exponentielle du courant injecté. La figure 1.9 décrit l'allure de cette conductance pour une synapse isolée [Abb94].

$$\begin{cases} g_{syn}(t) & = \overline{g_{syn}} \cdot r(t) \\ r'(t) & = -\tau_{syn} \cdot r(t) + W_{syn} \cdot \delta(t - t_0) \\ lim_{t \to -\infty} r(t) = 0 \end{cases} \tag{1.7}$$

$$\begin{cases} g_{syn}(t) & = 0 & \forall t < t_0 \\ g_{syn}(t) & = \overline{g}_{syn} \cdot W_{syn} \cdot \exp(-\frac{t-t_0}{\tau_{syn}}) & \forall t \geq t_0 \end{cases} \tag{1.8}$$

Dans ces deux équations, τ_{syn} représente la vitesse de décroissance du courant, W_{syn} le poids de la synapse considérée, δ la fonction de Dirac et t_0 le moment auquel la stimulation doit être envoyée. Les équations 1.7 et 1.8 donnent la même solution tracée en figure 1.9. Chacune sera plus ou moins bien adaptée à la résolution selon le simulateur utilisé. Une simulation logicielle préfèrera l'expression explicite de l'équation(1.8). Pour un simulateur analogique, le système différentiel (1.7) sera privilégié car il est indépendant du temps absolu. Seuls interviennent les conditions initiales et l'évènement en t_0, ce qui est caractéristique de la réponse d'un système à une stimulation extérieure. Cette forme d'équation est en outre plus pratique à implanter matériellement.

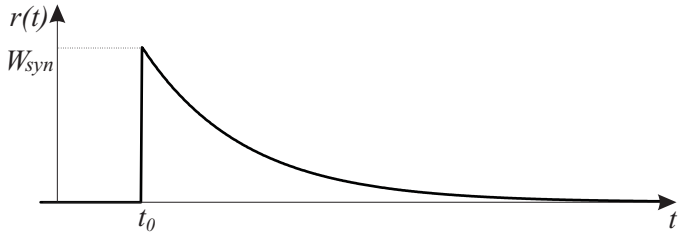

Fig. 1.9 – Évolution de la conductance du canal synaptique suite à un évènement produit en t_0 selon les équations 1.7 et 1.8

Les expressions 1.7 et 1.8 ne sont valables que pour un seul évènement en t_0. Pour n évènements se produisant aux temps $t_{i \in \{1 \cdots n\}}$, on obtient les expressions 1.9 et 1.10

$$\begin{cases} g_{syn}(t) & = \overline{g_{syn}} \cdot r(t) \\ r'(t) & = -\tau_{syn} \cdot r(t) + \sum_{i=1}^{n} w_{syn}(t_i) \cdot \delta(t - t_i) \\ lim_{t \to -\infty} r(t) = 0 \end{cases} \tag{1.9}$$

$$\begin{cases} g_{syn}(t) & = \overline{g_{syn}} \cdot \sum_{i=1}^{n} r_i(t) & \\ r_i(t) & = 0 & \forall t < t_i \\ r_i(t) & = w_{syn}(t_i) \cdot \exp(-\frac{t-t_0}{\tau_{syn}}) & \forall t \geq t_i \end{cases} \tag{1.10}$$

Le poids étant succeptible de subir des variations, il est désormais dépendant du temps, d'où la notation $w_{syn}(t)$.

Le caractère excitateur ou inhibiteur de la synapse est déterminé par le potentiel de repos. Pour une synapse inhibitrice, ce potentiel est très négatif (-80mV), pour une synapse excitatrice, il est nul.

S'il représente bien l'allure générale du courant synaptique, ce modèle souffre par contre d'un défaut majeur : une discontinuité apparaît lors de l'arrivée de chaque potentiel d'action. Il est alors possible de supprimer cette discontinuité en effectuant une croissance pendant un temps déterminé. Plusieurs formes sont envisageables, elles dépendent principalement de leur compatibilité avec le système de simulation. La forme optimale reste de nature exponentielle.

Phénomènes de saturation

Si son poids est une des caractéristiques de la synapse, cela ne signifie pas qu'une synapse ayant un poids constant transmettra tous les évènements avec une force identique. Elle est, en effet, le siège de phénomènes qui viennent moduler sa capacité de transmission. Le poids d'une synapse ne représente alors que la force maximale que l'on puisse en espérer.

Comme le courant qu'elle peut délivrer est majoré. Un phénomène de saturation intervient et limite l'augmentation du courant synaptique. Ceci signifie que $g_{syn}(t)$ ne peut excéder la valeur $\overline{g}_{syn}$. Cette saturation s'exprime par une modulation de W_{syn}, elle est exprimée dans l'équation 1.11.

$$w_{syn}(t) = W_{syn}(t)\frac{\overline{g_{syn}} - g_{syn}(t)}{\overline{g_{syn}}} \tag{1.11}$$

$W_{syn}(t)$ ne dépend pas de phénomènes synaptiques mais uniquement de la plasticité au niveau du réseau.

Une autre limite est produite par la lenteur (relative) de la recapture des neurotransmetteurs au niveau de la synapse, et la reconstitution de vésicules synaptiques. Cette inertie est à l'origine de la baisse de l'efficacité synaptique dans le cas d'un train de potentiels d'action temporellement rapprochés. Lors de cette stimulation excessive, la sollicitation soutenue des synapses limite la quantité de neurotransmetteur disponible. La contribution de chaque potentiel d'action est donc faible pendant le train, alors que les premières stimulations, d'amplitudes significatives, sont terminées. On obtient donc une transmission faible de l'information. Une expression simplifiée de ce phénomène est donnée par l'équation 1.12, où W_{plast} représente le poids synaptique déterminé par la plasticité et t_{dern} l'instant à laquelle le dernier potentiel d'action a été transmis.

$$W_{syn}(t) = W_{plast} \cdot \exp\left(t - t_{dern}\right) \tag{1.12}$$

Un dernier approfondissement peut être effectué en considérant que les vésicules synaptiques ne libèrent pas systématiquement leurs neurotransmetteurs lorsqu'elles sont stimulées. Il faut alors définir p_o : la proportion de vésicules effectivement relachées dans la ou les fentes synaptiques par rapport au nombre total de vésicules relachables. Cette grandeur est considérée constante pour chaque synapse. Elle n'est, par contre, pas uniforme à l'échelle du réseau. Sa signification est essentiellement probabilistique, il convient donc de la considérer comme un paramètre stochastique. Pour l'introduire dans le modèle, il faut modifier les équations 1.12 et 1.11 pour obtenir le système 1.13 :

$$\begin{cases} w_{syn}(t) & = W_{syn}(t)\frac{(\overline{g_{syn}}-g_{syn}(t))}{\overline{g_{syn}}} \\ W_{syn}(t) & = W_{plast} - (W_{plast} - W_{syn}(t_{dern})) \cdot \exp(t - t_{dern}) \end{cases} \tag{1.13}$$

Une orientation vers les réseaux : la multisynapse

Nous venons de voir un modèle relativement détaillé de la synapse. La complexité de ce modèle est de l'ordre de celle des canaux ioniques. Or, le nombre de canaux ioniques d'un neurone est généralement inférieur à quatre tandis qu'une synapse est nécessaire à chaque connexion venant d'un autre neurone. En biologie, ce nombre peut atteindre 10^4 synapses par neurone. Pour un réseau relativement modeste, il est juste nécessaire d'obtenir une synapse par neurone potentiellement afférent. A moins d'introduire de fortes restrictions sur les possibilités de connectivité, un réseau de n neurones nécessitera donc $(n-1)$ synapses par neurone. On obtient néanmoins très rapidement un réseau de synapses plutôt qu'un réseau de neurones, puisque ce sont elles qui nécessiteront la majeure partie des efforts de simulation.

La parade à cette exigeance croissante, est d'utiliser une propriété de la fonction exponentielle : quelles que soient les valeurs de A, B, t_a, t_b réels, il existe C, t_c réels qui vérifient la formule 1.14.

$$A \cdot \exp\frac{t-t_a}{\tau} + B \cdot \exp\frac{t-t_b}{\tau} = C \cdot \exp\frac{t-t_c}{\tau} \tag{1.14}$$

Ce qui signifie que pour toutes les synapses ayant la même constante de temps (ou *cinétique*) il est possible de n'utiliser qu'un seul canal. La figure 1.10 représente cet effet. Le canal alors constitué dit *canal postsynaptique*, reçoit l'ensemble des évènements des synapses auxquelles il se substitue. Il n'est donc nécessaire de gérer qu'un seul canal par type de synapse. Ce canal est appelé *multisynapse* [Des99a], par opposition aux *monosynapses* décrites précédemment. La différence fonctionnelle est que, au moment de la stimulation, on fournit également le poids synaptique relatif à la connexion qui est sollicitée.

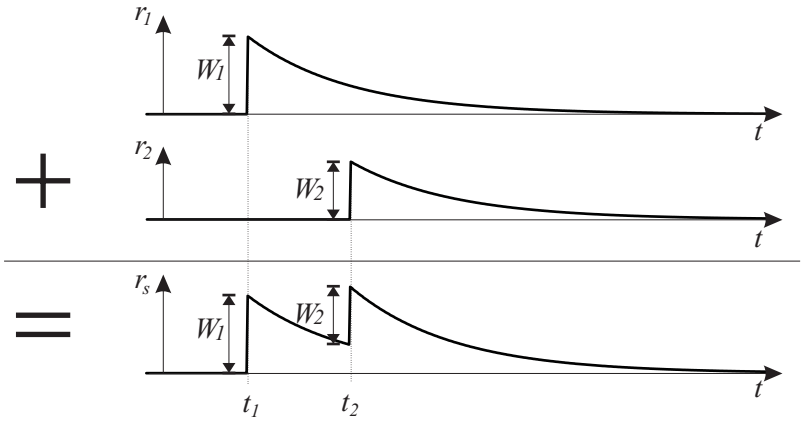

Fig. 1.10 – Sommation des conductances. Toutes les courbes ici représentées sont des exponentielles décroissantes de même constante de temps. Dans les intervalles où elles sont continues, elles ont donc pour équation $A \cdot \exp(-\frac{t-t_a}{\tau})$

L'inconvénient majeur des multisynapses réside dans le fait qu'elles ne sont pas capables de déterminer à quelle synapse correspond l'évènement qu'elles sont en train de traiter. Il sera alors impossible de traiter directement les phénomènes de modulation de poids synaptique. Ces modulations doivent être calculées en amont dans la structure du réseau, pour fournir à la synapse un poids modifié. C'est-à-dire que le poids envoyé sur l'entrée a déjà subi une modulation due aux effets d'atténuation observables dans les synapses.

Ce calcul pourra lui aussi être mutualisé pour l'ensemble des synapses d'une cellule biologique, puisqu'elles sont modélisées par les mêmes équations. Cette astuce est surtout intéressante pour les simulateurs logiciels, pour améliorer le temps total de calcul.

Retour à l'arbre dendritique

Avec l'usage de monosynapses, la seule issue pour modéliser les combinaisons de contributions au sein de l'arbre dentritique est de les interconnecter entre elles ; cette méthode est cependant très gourmande en ressources.

Maintenant que nous avons défini la multisynapse, il est possible d'imaginer une fonction qui réajuste (une dernière fois) les poids synaptiques. Une telle fonction pourrait se baser sur la structure de l'arbre et les intervalles de temps entre les stimulations.

1.3.3 À l'échelle du réseau

Cette partie pourrait s'écrire dans la continuité de la précédente, tant la frontière entre les synapses et le réseau est ténue. Les notions abordées ici du point de vue du réseau le sont par opposition aux centres d'intérêt à l'échelle de la cellule.

Le bruit synaptique

Le premier phénomène observable à l'échelle du réseau est le *bruit synaptique* : il s'agit de contributions d'origines extérieures au réseau. En effet, l'isolation électrique entre les neurones n'est pas parfaite. Une transmission synaptique perturbe les dendrites alentours, et donc le soma auquel elles sont reliées. En conséquence, les neurones baignent constamment dans un bruit issu des autres réseaux proches.

Pour prendre en compte ce phénomène, des faibles stimulations synaptiques sont envoyées selon une loi stochastique et pseudo-aléatoire.

La présence de ce bruit environnant est essentielle pour bien représenter le comportement général d'un réseau. Il est à l'origine d'un phénomène de resynchronisation des potentiels d'action des neurones proches [Des99b]. Cette activité de fond peut également être utilisée pour générer ou faire émerger une activité au sein d'un réseau qui ne reçoit pas de stimulus particulier.

La STDP (Spike Timing Dependant Plasticity)

La plasticité des réseaux de neurones a pour conséquence une fluctuation des poids synaptiques au sein de ce réseau. Dans le cadre de la STDP, les variations de poids sont associées à la causalité entre les neurones présynaptique et postsynaptique.

Si le neurone présynaptique émet un potentiel d'action avant le neurone postsynaptique, il a contribué au déclenchement du second. La connexion considérée sera

alors renforcée. Par contre, si la séquence se déroule dans l'ordre inverse, la connexion sera plus faible. L'amplitude des variations est d'autant plus faible que l'écart temporel entre les potentiels d'action est élevé. Cette caractéristique est connue sous le nom de *loi de Hebb* [HEB49]. Bien que cette loi ne se vérifie pas systématiquement, les algorithmes qui lui correspondent sont les plus utilisés.

Le modèle le plus simple considère des variations par niveau. La figure 1.11 illustre les variations appliquées au poids synaptique par rapport à $\delta_t = t_{post} - tpre$, c'est-à-dire au temps écoulé entre les potentiels d'action présynaptique et postsynaptique, considéré algébriquement.

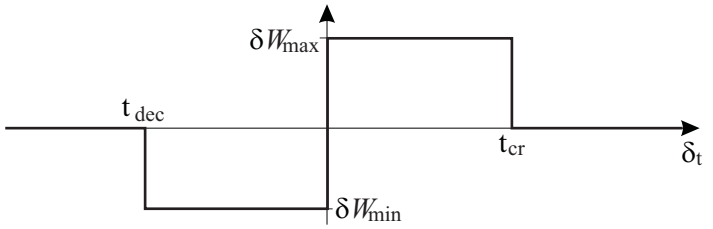

FIG. 1.11 – Loi d'évolution des variations d'un poids synaptique soumis à un modèle simple de STDP

Ce genre de comportement par paliers n'est bien sûr pas représentatif d'un comportement naturel. Un modèle plus proche décrit que l'influence subie obéit à une décroissance exponentielle [BAD06]. Elle est illustrée par la figure 1.12.

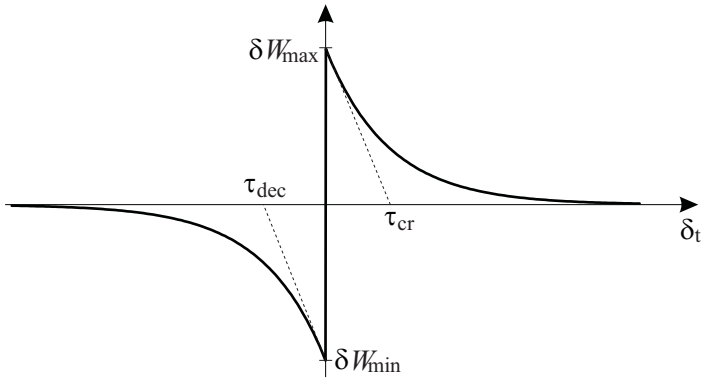

FIG. 1.12 – Loi d'évolution exponentielle des variations d'un poids synaptique soumis à un modèle fidèle de STDP

Nous remarquons cependant qu'il existe une non-linéarité très importante lorsque les deux neurones ont une activité simultanée. Les variations sont alors extrêmes, mais le signe de la contribution est aléatoire à cause de l'incertitude autour de la notion de simultanéité. Cette incohérence se résoud par l'usage d'une fonction continue. Il est possible d'implanter une portion linéaire aux faibles valeurs de δ_t, ou de remplacer l'évolution en $\exp(\delta_t)$ par une fonction $\delta_t \cdot \exp(\delta_t)$. De cette façon, la connexion de deux neurones simultanés subit des fluctuations mineures, voire négligeables. Ces fonctions sont représentées en figure 1.13.

Les dernières évolutions du modèle prennent en considération le fonctionnement

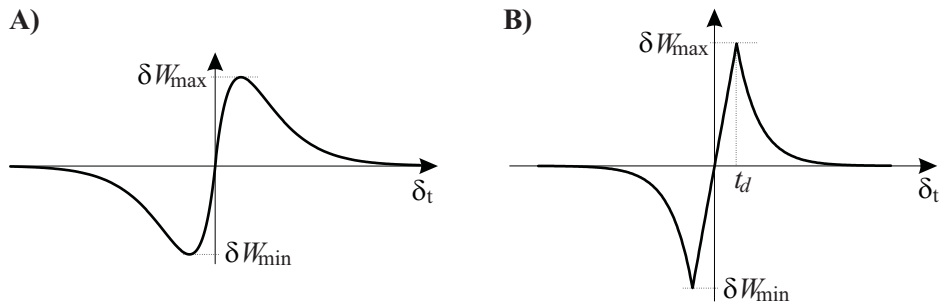

FIG. 1.13 – Lois d'évolution de variation d'un poids synaptique soumis à la STDP avec continuité en 0. A : fonctions $\delta_t \cdot \exp(\delta_t)$ B : liaison par fonction linéaire

de la synapse. Lorsque deux neurones n'ont jamais d'activité dans une même fenêtre temporelle, les équations précédentes ne proposent aucune description de l'atrophie de la synapse. Il faut pour cela utiliser une limite non nulle pour les temps infinis. De plus, si deux neurones ont des activités simultanées, les connexions qui les relient sont inutiles, puisque leurs effets n'ont pas le temps d'être intégrés comme contribution à un potentiel d'action. Un décalage temporel peut alors être introduit pour tenir compte de ce temps de propagation. La courbe donnée précédemment apparaît alors décentrée par rapport à l'origine, comme le montre la figure 1.14

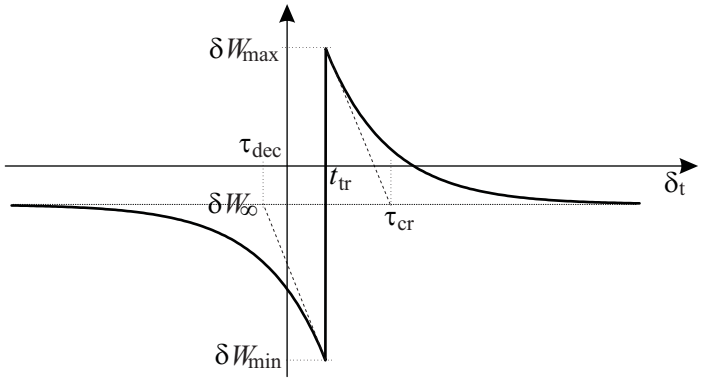

FIG. 1.14 – Prise en compte de phénomènes de second ordre en STDP

Le jeu de paramètres ainsi obtenu permet d'avoir une bonne description des phénomènes rencontrés dans une plasticité à l'échelle de la synapse.

D'autres modèles de plasticité

D'autres formes de plasticité peuvent prendre place dans les réseaux de neurones. Contrairement à la STDP, leur comportement n'est pas localisé à une synapse, mais prend en compte l'activité de son voisinage géographique et/ou structurel. Étant donné la faible utilisation de cette plasticité au sein des neurosciences computationnelles, nous ne nous attarderons pas davantage sur ce sujet. Il est cependant nécessaire de faire attention aux limites du modèle avec lequel nous appréhendons les réseaux de neurones.

1.3.4 À l'échelle de l'individu

Au-delà de la description que nous venons d'effectuer, une approche sociologique permet de mettre en évidence l'influence d'autres paramètres liés à l'individu. Le système nerveux subit des influences dont certaines sont d'origine chimique. Il convient donc de garder à l'esprit que les modèles qui viennent d'être décrits sont eux-mêmes modulés et/ou régulés sur une échelle plus grande encore.

L'influence la plus susceptible d'intervenir au niveau où nous nous trouvons est celle des hormones et des substances que nous ingérons. Si les influences comportementales sont identifiées, les modifications de modèles qui en découlent restent obscures. L'exemple des phases du sommeil est très intéressant. Les phases éveillées, de sommeil profond ou de sommeil paradoxal sont déterminées par des concentrations hormonales dans le cerveau des mammifères. Le sommeil paradoxal, montre qu'une population de neurones est insensible à certaines variations de ces concentrations, tandis qu'une autre (celle du cortex moteur), devient totalement inhibée.

La capacité d'apprentissage des jeunes individus est plus élevée que celle de leurs aînés [Buc05]. Il s'agit d'un autre exemple de phénomène à approfondir pour véritablement cerner l'ensemble des processus impliqués dans l'apprentissage. Plusieurs paramètres peuvent jouer : le nombre de neurones, le nombre de connexions existantes, mais aussi (encore une fois) les concentrations hormonales.

Le dernier exemple choisi pour illustrer l'intervention de mécanismes à l'échelle de l'individu est la sélectivité de la mémoire par rapport au contexte émotionnel. Les émotions se traduisent, elles aussi, par des concentrations hormonales. Il est remarquable que ces taux hormonaux sont contrôlés par le cortex lui-même. Nous obtenons alors une situation où un réseau (de très grande taille) détermine indirectement ses propres paramètres d'évolution. Ces études relèvent d'une grande complexité et nécessitent les efforts conjoints de plusieurs disciplines allant de la chimie à la sociologie pour être menées dans de bonnes conditions.

1.3.5 Un mot sur la variabilité

Pour clore cette section destinée à la présentation des modèles utilisés en neurosciences computationnelles, il est nécessaire de mener une discussion sur leur application. En effet, très peu de données ont été avancées pour appuyer les équations présentées. Cela vient en partie du grand nombre de familles de neurones qu'il est possible de rencontrer. En outre, même parmi plusieurs neurones de la même catégorie, il existe une variabilité des paramètres du modèle non négligeable.

Multiplier les neurones dans un réseau, en reproduisant autant de fois que nécessaire le même motif unitaire, perd ici de son sens. Le potentiel formidable des réseaux de neurones est justement de parvenir à des résultats comparables sur des réseaux non identiques. Une ressemblance stricte entre deux neurones pourrait conduire à des équilibres invraisemblables, alors que la dispersion naturelle effectue un ordonnancement des propriétés des cellules les unes par rapport aux autres. Une simulation doit donc tenir compte de ce paramètre.

1.4 État de L'art

Nous venons d'avoir une vue d'ensemble des modèles exploitables. La présentation effectuée est également suffisamment précise pour appréhender les compromis à effec-

tuer ainsi que l'éventail de phénomènes observables en fonction des approximations effectuées.

Voyons désormais comment ces choix ont été effectués par différentes équipes pour les concilier avec la simulation électronique matérielle. Nous ne pourrons bien sûr avoir qu'un aperçu des différentes approches. Cette présentation se limite aux plus caractéristiques. Nous verrons aussi les différentes équipes qui mettent en œuvre les systèmes hybrides, puisqu'il s'agit d'une activité cible de la simulation en temps réel.

Ces travaux étant orientés vers l'électronique associée aux neurosciences, nous nous intéresserons davantage aux techniques et à l'instrumentation mises en œuvre, plutôt qu'aux résultats dégagés par chaque groupe.

1.4.1 La conception de neurones intégrés sur silicium

Les précurseurs (Douglas *et al.*)

C'est en 1991 que fut publiée, par Misha Mahowald et Rodney Douglas, chercheurs à CalTech (CA, USA) dans l'équipe de Carver Mead, la première solution pour l'intégration analogique sur silicium d'un modèle biophysique de neurone [MAH91]. La technique utilisée pour l'implantation utilise des transistors MOS utilisés sous la tension de seuil. Le modèle utilisé suit le formalisme de Hodgkin et Huxley. Les équations propres à chaque canal ionique ont été simplifiées pour faciliter l'implantation, en revanche, cinq canaux ioniques sont présents au lieu de trois dans le modèle minimal.

Plus récemment, les travaux de cette équipe se sont orientés vers des modèles *Integrate and fire* [LIU04] pour obtenir des réseaux contenant un nombre élevé de neurones. Certains circuits possèdent quelques paramètres pour compenser la dispersion des composants, les grandeurs sont alors mémorisées par la technique des grilles flottantes. Ces réseaux de grande taille sont destinés à la conception de systèmes electroniques adaptatifs pour le traitement des informations sensorielles (visuelles ou auditives).

Un support pour l'étude de phénomènes de propagation (Bilbault *et al.*)

L'équipe de Jean-Marie Bilbault, laboratoire L2eI, Université de Bourgogne, Dijon, développe également des circuits neuromorphiques pour étudier les phénomènes de propagation au sein du neurone. Ces circuits utilisent le modèle de FitzHugh-Nagumo [FIT61]. La spécificité de leurs travaux repose sur le fait que les neurones ne sont pas considérés comme ponctuels. Les modèles sont alors dotés de circuits de retard et/ou d'atténuation pour rendre compte des propagations dans le soma et l'arbre dendritique [BIN03].

L'usage de conductances lentes (DeWeerth *et al.*)

L'équipe de Stephen De Weerth, *Laboratory for Neuroengineering* (Neurolab), à Georgia Tech (GE, USA), conçoit des circuits intégrés neuromimétiques en technologie CMOS dont le modèle est dérivé du formalisme de Hodgkin et Huxley. La simulation s'effectue en temps réel avec des conductances à la fois lentes (long terme) et rapides (potentiel d'action) [MAH89]. Chaque conductance est une approximation du modèle d'origine, ce qui permet d'intégrer plusieurs neurones par circuit.

Ces circuits sont destinés à une simulation de neurones isolés au sein de réseaux hybrides, ou en réseau pour le contrôle de robots marcheurs. Ils intègrent une interface de communication au sein du réseau de neurones de type AER, elle présentée ci après.

1.4.2 La mise en réseau

Au-delà de l'intégration de modèles de neurones sur des circuits intégrés, la simulation de réseaux nécessite une mise en relation des différents circuits. Voyons désormais différentes techniques utilisées pour interconnecter un grand nombre de neurones.

Une architecture centralisée (Meier *et al.*)

Les travaux de Karlheinz Meier, dans l'*Équipe de Vision(s) Électronique* au sein du *Kirchoff-Institut fur Physik* (Heidelberg, Allemagne), sont orientés vers l'intégration à haute densité de réseaux de neurones pour la conception de systèmes totalement intégrés [FIE04]. Les simulateurs réalisés utilisent le modèle *Integrate and fire* avec fuite sur des circuits comprenant un grand nombre de neurones et de connexions synaptiques [SCH04].

Pour répondre aux contraintes de l'intégration, les simulations sont effectuées à contraintes de temps réel avec une échelle de temps de $1 : 10^5$ (soit une échelle temporelle pour la simulation électronique égale à 10^{-5} fois le temps réel biologique). Les réseaux sont constitués de plusieurs couches logiques hiérarchisées : au niveau d'un microréseau (de l'ordre de la centaine de neurones), à l'échelle du circuit (plusieurs centaines de neurones) et au sein du système (quelques milliers). Les difficultés techniques imposent des délais d'autant plus grands que les structures traversées sont nombreuses pour relier deux neurones. La grande quantité de cellules nerveuses disponibles compense toutefois cet écueil.

Les circuits développés permettent quelques degrés de liberté sur les valeurs du modèle, ce qui permet d'introduire de la diversité au sein du réseau. Les neurones sont réglables par groupes, ce qui est cohérent avec une grande taille de réseau où les analyses s'effectuent sur des populations.

L'AER : un protocole d'entrée sortie orienté réseau (Douglas *et al.*)

Afin de relier un grand nombre de circuits neuromorphiques, l'équipe de Rodney Douglas (Institut de Neuroinformatique, ETh Zurich, Suisse) a développé un protocole d'interface générique qui ne nécessite pas d'horloge distribuée : l'AER (Address-Event Representation). Ce protocole permet la communication entre des circuits de différentes générations sans adaptation particulière [BOA00].

Étant donné qu'il ne possède pas de gestion des communications à l'échelle du système, l'AER correspond à une politique dite de *meilleur effort*. Il ne garantit donc pas de fonctionnement sans défaut. Il n'y a pas de gestion des collisions d'évènements [4], ce qui aboutit à quelques pertes d'information. Cette propriété permet par contre une circulation rapide des données, quand elles sont prises en compte.

[4] Une collision correspond à une nécessité d'envoi de plusieurs paquets d'information alors que le canal de transmission n'en permet qu'un seul

1.4.3 L'adaptation du réseau

Le modèle le plus utilisé dans les systèmes de simulation matérielle est celui de la STDP. Cela est dû à la fois au grand intérêt des neurosciences pour ce phénomène, mais aussi à sa relative simplicité de mise en œuvre. Comme nous l'avons déjà vu précédemment, il s'agit d'une propriété même de la synapse, son calcul se prête alors très bien aux architectures de réseau éxistantes.

Des synapses adaptatives analogiques (Douglas *et al.*)

Les circuits réalisés au sein de l'équipe de Rodney Douglas sont dotés de synapses plastiques analogiques. L'architecture utilisée pour la mise en réseau des circuits est l'AER, ce qui signifie que les informations temporelles des potentiels d'action ne sont pas accessibles. Les algorithmes de STDP ont alors été adaptés pour permettre une évolution des poids synaptiques par une succession de valeurs suivant l'occurence des évènements présynaptiques [IND06]. Ces dernières sont suffisamment nombreuses pour se différencier de connexions tout ou rien.

Un algorithme de plasticité a également été développé pour permettre une intégration à grande échelle [FUS02]. Il a abouti à la conception d'un circuit spécifique [MIT06].

Un calcul numérique mutualisé (Meier *et al.*)

La vision du calcul plastique chez Meier *et al.* est plus orientée vers une simulation mixte. En effet, chaque circuit est en connexion directe avec un système numérique architecturé autour d'un *Power*PC fonctionnant sous GNU/Linux en temps réel. La plasticité est alors calculée de façon logicielle au fur et à mesure que les potentiels d'action sont transmis au circuit. Cette approche permet une meilleure précision puisque l'aspect temporel des évènements peut être utilisé dans le traitement des variations qui en découlent.

1.4.4 Conception d'organes sensoriels artificiels bio-inspirés

La différence fondamentale entre les systèmes bio-inspirés et bio-réalistes se situe dans le rapport qu'ils entretiennent avec les neurosciences computationnelles. Alors que les démarches bio-réalistes cherchent à reproduire fidèlement les modèles pour permettre leur exploitation en simulation, les visions bio-inspirées utilisent ces modèles comme exemples de fonctionnement et s'en inspirent pour réaliser des fonctions électroniques.

Un seul axe de recherche est exposé ici car nous nous limitons aux systèmes qui présentent un intérêt du point de vue des neurosciences. En effet, l'usage de techniques bio-inspirées se rencontre plus souvent dans des domaines applicatifs délicats comme le contrôle de robots marcheurs [MIL94] ; par contre, peu d'efforts sont dédiés à l'analyse comportementale des réseaux établis. Cet aspect du réseau de neurones en tant qu'outil, plutôt que moyen d'étude, ne fait pas partie de nos objectifs.

Les rétines artificielles (Boahen *et al.*)

Les travaux de Kwabena Boahen sont orientés vers la conception de rétines artificielles. Les systèmes développés intégrent à la fois un récepteur électronique et un

réseau de neurones chargé de l'analyse des informations recueillies. Les récepteurs optiques s'interfacent avec les neurones en utilisant le protocole AER.

Actuellement, les recherches de cette équipe s'orientent vers l'attention et l'apprentissage en continuant à développer des réseaux de taille croissante. Les résultats sont confrontés aux modèles expérimentaux à l'échelle du réseau pour valider leur fidélité. L'objectif, à terme, est de parvenir à la conception d'un système nerveux complexe pour conduire une analyse systématique des comportements de haut niveau.

1.4.5 Les systèmes hybrides en boucle fermée

Le principal avantage de la simulation en temps réel est la possibilité d'interfacer une activité simulée avec des neurones vivants. Les deux méthodes d'interaction avec les cellules nerveuses sont :

- l'accès intracellulaire, où l'on insère un sonde dans la cellule nerveuse ;
- l'accès extracellulaire, dépot de cellules nerveuses sur un substrat au sein lequel sont reportées des microélectrodes métalliques : le MEA (Micro Electrodes Array).

MEA : controle d'un robot par un réseau vivant

Le principal intérêt de cette méthode est que les cellules déposées sur le substrat forment déjà un réseau. Ce dernier peut donc être utilisé pour réaliser des fonctions computationnelles.

Un apprentissage par la punition (Potter *et al.*) Les travaux menés à *Caltech* (California Institute of Technology) par S. Potter visent à étudier les comportements liés à l'apprentissage [DEM01].

Les cultures de neurones sont utilisées comme étant le système de contrôle d'un objet virtuel (animal ou robot) introduit dans un environnement donné. Le passage de l'objet sur une zone interdite provoque l'envoi d'une stimulation électrique sur le réseau en culture. Cette stimulation est perçue comme une punition, et, après une période d'apprentissage, l'objet évite les obstacles.

L'électronique impliquée dans ces expériences ne reproduit pas, à proprement parler, l'activité électrique de neurones, mais place le réseau vivant dans un environnement particulier qu'elle simule. Ces travaux nous concernent cependant puisqu'ils montrent l'intérêt que présente un réseau qui puisse être entièrement observé.

Culture de neurones sur MEA spécifique (Fromherz *et al.*) L'équipe de Peter Fromherz, de l'Institut Max Plank pour la biochimie, à Munich, prend l'expression *neurones sur silicium* au premier degré. Leurs travaux visent à développer des techniques de culture de neurones isolés sur substrat de silicium, pour faciliter les fonctions de mesure et de stimulation. Des expériences ont déjà été menées avec des neurones d'escargot [ZEC01].

Cette technique combine les avantages des deux méthodes : l'usage d'électrodes externes assure une grande longévité aux neurones cultivés et la séparation de chaque cellule évite les étapes de traitement de signal nécessaires avec les MEAs. La préparation de telles expériences est cependant longue, il n'est plus possible d'utiliser de réseau préformé et seuls les plus gros neurones peuvent être utilisés.

Électrodes intracellulaires

L'usage d'une électrode intracellulaire permet à la fois la mesure du potentiel de membrane et l'injection d'un courant de stimulation. Elle permet donc de faire interagir le neurone avec un système electronique. Bien que moins impressionnants que les précédents, ces travaux sont essentiels pour les neurosciences fondamentales.

Verrouillage de phase sur un réseau hybride (Canavier *et al.*) Les travaux de Sorinel A. Oprisan et Carmen C. Canavier, *Department of Psychology*, Université de New Orleans (LA, USA), s'orientent sur le fonctionnement d'un réseau hybride à deux neurones, l'un vivant, l'autre artificiel. L'intéret est l'exploration des différents modèles de neurones pour obtenir un fonctionnement du réseau en alternance de trains d'impulsions [OPR04].

Le neurone artificiel est simulé par ordinateur en temps réel, ce qui permet une grande souplesse pour le choix du modèle. La stimulation est injectée par une synapse artificielle fonctionnant par la méthode de *dynamic clamp*. Ce protocole a été élaboré pour simuler un canal ionique artificiel sur un neurone vivant.

Les recherches de cette équipe continuent sur la réduction de la complexité des modèles neuronaux pour un problème donné.

Etude de plasticité sur de petits réseaux (Nowotny *et al.*) Les travaux de T. Nowotny et V. Zhigulin, Université de Californie (CA, USA), utilisent un protocole proche du précédent. Cependant, le neurone artificiel simulé de façon logicielle utilise le modèle de Hodgkin et Huxley. Par contre, les connexions entre le neurone artificiel et le neurone vivant sont de nature plastique. Leurs travaux se focalisent alors sur l'étude de cette plasticité et ses conséquences à l'échelle de ces deux neurones [NOW03].

Un exemple de rebouclage vivant (Nicolelis *et al.*)

La présence d'interface entre un organisme vivant et un système électronique ne signifie pas nécessairement de simulation en temps réel. Les travaux de Miguel Nicolelis, Université Duke, Durham (NC, USA), ont abouti au pilotage d'un bras artificiel par l'analyse de l'activité cérébrale d'un primate [CHA99]. Un ordinateur analyse l'activité et en déduit le mouvement à effectuer.

Cependant, le calcul est effectué à la volée ; bien qu'il soit rapide, le rebouclage n'est donc pas temps réel. De plus, malgré le travail de traitement du signal effectué par l'ordinateur, la phase d'apprentissage est réalisée par le cerveau du primate. Il ne s'agit donc pas, à proprement parler, de la réalisation d'un système electronique adaptatif en temps réel. Les objectifs de l'équipe se portent d'ailleurs plus sur l'analyse de l'activité cérébrale que sur l'instrumentation.

1.4.6 Une représentation des systèmes actuels

Nous pouvons représenter l'ensemble des systèmes de réseaux de neurones artificiels dans un même espace (figure 1.15) : les différents axes représentent la complexité des neurones (x), l'échelle de temps (y) et le nombre de neurones mis en jeu (z). Le caractère plastique ou non des réseaux n'est pas représenté. L'apparition de l'adaptation dans tous les systèmes orientés vers l'étude computationnelle des réseaux est

envisageable à court ou moyen terme. Nous aurons l'occasion de voir, dans la conclusion, un instantané de l'avancement de chaque équipe valable en novembre 2006.

Les étapes de complexité sont données par les identifiants suivants : IF : *Integrate and Fire* ; FN : Fitzhug-Nagumo ; SHH : *Simplified* Hodgkin et Huxley ; HH : formalisme de Hodgkin et Huxley.

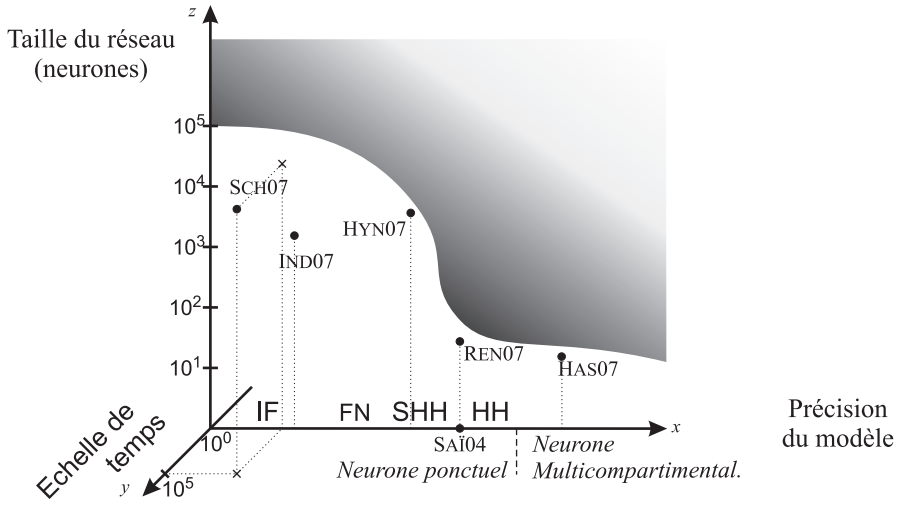

FIG. 1.15 – Positionnement relatif des réseaux de neurones existants

Les différentes équipes représentées sont les équipes de l'université de Heidelberg (UHEI) dirigée par Karlheinz Meier [SCH07], de l'ETH Zurich dirigé par Giacomo Indiveri [IND07] de GeorgiaTech dirigée par Paul Hasler [HAS07] de l'université de Stanford dirigée par Kwabena Boahen [HIN07] et l'équipe *Ingénierie des Systèmes Neuromorphiques* du laboratoire IXL dirigée par Sylvie Renaud [REN07].

Nous pouvons voir une extrapolation de la taille de réseau qu'il est possible d'espérer, par rapport à la complexité du modèle utilisé. Les modèles complexes sont plus délicats à mettre en œuvre dans de grands réseaux. Cette difficulté provient de deux phénomènes :

- les hypothèses simplificatrices effectuées sur les neurones peuvent également être reportées sur les méthodes de mise en réseau, ce qui allège la structure ;
- un faible nombre d'équations entraîne une grande densité de neurones par unité de surface de silicium, des réseaux sont alors intégrés au sein du même circuit.

L'ensemble des systèmes intégralement inclus dans le plan $(0xz)$ fonctionne en temps réel et est donc potentiellement compatible avec les réseaux hybrides. Cependant, seuls les systèmes intégrant des modèles complexes s'intéressent ouvertement à cette voie. Les réseaux basés sur des modèles multicompartimentaux sont confrontés au problème de l'instrumentation qui impose une seule mesure par soma, et donc une gestion des neurones vivants comme entités ponctuelles.

Les travaux de l'équipe *Ingéniérie des Systèmes Neuromorphiques*, décrits ci-après, y sont représentés. L'objet de cette thèse est d'effectuer la transition du point [SAÏ04] au point [REN07].

1.5 Le groupe *Ingénierie des Systèmes Neuromorphiques*

Le groupe *Ingénierie des Systèmes Neuromorphiques* a pour objectif scientifique la conception optimisée des circuits et systèmes électroniques dont l'architecture et les composantes s'inspirent des systèmes nerveux vivants. Traitée à l'IXL depuis plus de 10 ans, cette thématique est aujourd'hui en expansion forte.

Les travaux effectués mettent en œuvre des partenariats avec des laboratoires internationaux relevant d'un large éventail de disciplines allant de la neurophysiologie à l'informatique. Ces partenariats se sont concrétisés récemment sous la forme des différents projets européens *SenseMaker* (2002-2005), NEUROBIT (2002-2005), FACETS (2005-2009) et NEURO-*vers*-IT (2006-2009).

1.5.1 L'activité du groupe

Deux axes spécifiques se distinguent dans l'activité du groupe : *Conception de neurones sur silicium* et *Instrumentation des réseaux de neurones artificiels et hybrides*. Ils correspondent, d'une part, au développement de circuits intégrés à dominante analogique, et, d'autre part, à la conception de systèmes matériels et logiciels exploitant ces circuits à des fins biomédicales ou computationnelles.

Conception de neurones sur silicium

Les circuits intégrés (ASICs) analogiques et mixtes conçus dans le cadre de ces recherches modélisent l'activité électrique des neurones biologiques en temps réel et continu [REN04]. Ces circuits sont destinés à l'étude des réseaux de neurones biologiquement réalistes, et autorisent les connexions en temps réel avec des cellules vivantes.

Ces travaux sont essentiellement menés en collaboration avec le laboratoire INSERM E.358 de Physiopathologie de Réseaux Neuronaux et Médullaires (Institut François Magendie, Bordeaux), l'unité CNRS de Neurosciences Intégratives et Computationnelles (Institut Albert Fessard, UMR 2191, Gif-sur-Yvette) et le Laboratory for Computationnal Neuroscience (École Polytechnique Fédérale de Lausanne). Ces partenariats permettent de couvrir les aspects expérimentaux ainsi que le développement des modèles neurophysiologiques. Une collaboration avec l'équipe Electronic Vision Group de l'Université d'Heidelberg (UHEI) est plus orientée vers l'électronique et concerne le développement des ASICs neuromimétiques.

Les circuits réalisés modélisent l'activité électrique des neurones en résolvant, en continu et en temps réel, les équations du formalisme mathématique de Hodgkin et Huxley. Selon la taille des réseaux qu'ils sont destinés à constituer, ils intègrent un nombre variable de conductances ioniques et sont, de partiellement à totalement, reconfigurables. L'un de ces circuits correspond à une partie du travail présenté dans ce document. Ils ont été réalisés chez austria*micro*systems en technologie BiCMOS 0.8µm puis SiGe 0.35µm.

Parallèlement, des recherches sont menées pour caractériser et réaliser une bibliothèque d'IPs [SAÏ04], [LEV06]. L'objectif est ici d'optimiser le processus de conception de nouveaux ASICs en privilégiant la réutilisation et en prévoyant la migration technologique.

Instrumentation des réseaux de neurones artificiels et hybrides

L'opération de recherche *instrumentation des réseaux de neurones artificiels et hybrides* porte sur le développement de systèmes dédiés aux neurosciences computationnelles et expérimentales. Ces travaux mettent en œuvre des ASICs spécialement développés et des systèmes d'électronique numérique et analogique dans des environnements temps réel. La conception du système de simulation dont il sera question dans cet ouvrage contribue à cet axe. Des systèmes interconnectant en temps réel des cellules nerveuses *in vitro* et des réseaux de neurones artificiels ont déjà été développés dans le cadre du projet européen NEUROBIT [MAR04]. Ils se chargent de l'acquisition et de l'enregistrement de signaux issus de matrices de microélectrodes extracellulaires. Il met en œuvre une boucle de communication temps réel entre des neurones vivants et un système artificiel pour permettre l'étude des fonctions d'apprentissage dans des réseaux hybrides vivant-artificiel.

Les travaux menés pour cette activité s'appuient sur des collaborations avec le laboratoire INSERM E.358 Physiopathologie de Réseaux Neuronaux et Médullaires (Institut François Magendie, Bordeaux), le Department of Biophysical and Electronic Engineering (DIBE) de l'Université de Gênes, Italie, le Netherlands Institute for Brain Research (NIBR), le laboratoire CNRS Neurosciences Intégratives et Computationnelles (Institut Alfred Fessard, textscumr 2191, Gif-sur-Yvette) et le Laboratory for Computationnal Neuroscience (École Polytechnique Fédérale de Lausanne). Ils reçoivent et exploitent les systèmes sur des protocoles expérimentaux spécifiques.

1.5.2 Le projet européen *SenseMaker*

Les travaux présentés dans cet ouvrage ont été en grande partie réalisés lors du projet *SenseMaker* dans lequel ils s'inscrivent. Planifié sur une durée de 3 ans (2002-2005), il s'inscrit dans le cadre de l'appel d'offres 2001 *Life-Like Perception Systems* (LPS) et du programme FET (Future Emerging Technology), mis en place par la communauté européenne. Cet appel d'offre traite du développement de systèmes artificiels sensi-moteurs dont les structures seraient inspirées du vivant.

Le but du projet SENSEMAKER est de définir et implanter une architecture électronique capable de fusionner des informations sensorielles issues de modalités diverses en une représentation unique d'un environement donné. Cette architecture sera directement inspirée des principes de réception et fusion sensorielle du système nerveux. Un objectif à terme est que le système réalisé soit capable d'autoreconfiguration, créant de nouvelles connexions entre différents récépteurs sensoriels pour effectuer des traitements d'ordre supérieur. Sur ce projet sont associés des neurophysiologistes, des électroniciens, des informaticiens et des psycho-physiciens de différents laboratoires de recherche européens.

Une approche biologique et psycho-physique permet, dans un premier temps, de dégager les principes et modèles fondamentaux pour l'intégration sensorielle et multi-sensorielle. Ces travaux sont complétés par une approche de modélisation de ces phénomènes d'intégration. Des simulations permettent alors de mettre en œuvre et d'explorer les principes proposés. Des simulations de réseaux de grandes tailles sont ensuites menées. Elles intègrent les principes biologiques précédemment dégagés, ainsi que des modèles de fonctions cognitives de haut niveau.

L'équipe *Ingeniérie des Systèmes Neuromimétiques* de l'IXL, intervient au niveau de l'élaboration d'un système de simulation. Ce système s'appuie sur des modèles de

neurones oscillants développés par l'UNIC, simulés en temps réel par des ASICs. Les interactions synaptiques doivent pouvoir se contrôler numériquement pour permettre un topographie entièrement programmable, et une évolution dynamique des poids synaptiques. Pour assurer le fonctionnement en temps réel de ce système, il sera piloté informatiquement. Cet outil permettra de proposer et tester des règles optimales de connectivité et de plasticité pour un meilleur traitement des informations sensorielles, en se basant sur les phénomènes observés expérimentalement.

1.6 Résumé

Nous avons d'abord vu les aspects généraux de la simulation des réseaux de neurones, et surtout défini le cadre dans lequel ces travaux abordent les simulations. Après une présentation des éléments qui interviennent dans les systèmes simulés, nous avons vu les différents modèles utilisables selon le niveau de précision recherché, leurs avantages ainsi que leurs limites, tant du point de vue de la qualité de la simulation, que de sa mise en œuvre. Une présentation des travaux significatifs du domaine, ainsi que ceux de l'équipe d'Ingénierie des Systèmes Neuromorphiques de l'IXL a permis de positionner les travaux ici présentés par rapport à l'état de l'art. Nous pouvons désormais aborder les méthodes de mise en réseau de circuit neuromimétiques, en ayant comme principale priorité, la fidélité au biologique.

Chapitre 2

L'ordinateur synaptique

Nous allons maintenant aborder les notions propres à la constitution d'un système complet de simulation de réseaux de neurones. Ce système, conçu dans le cadre du projet européen *SenseMaker*, s'articule autour d'ASICs (Application Specific Integrated Circuit) interfacés à un ordinateur à l'aide d'une carte connectée au bus PCI. La simulation s'effectue selon la partie concernée, sur support analogique intégré, numérique materiel ou numérique logiciel.

Nous verrons d'abord une présentation du circuit *Trieste* utilisé pour simuler les neurones de façon analogique ; pour ensuite étudier dans quelle mesure le numérique peut intervenir dans un réseau analogique sans pour autant le dénaturer. Nous aborderons alors la réalisation du système en tant que tel. Dès lors, nous pourrons clore ce chapitre par une analyse critique des résultats obtenus à l'aide de cette première génération de système.

2.1 La gestion numérique du réseau

Nous nous intéressons ici à une approche mixte de la simulation de réseaux de neurones. Les neurones mis en œuvre seront simulés de façon analogique et le réseau sera géré par des méthodes relevant du domaine du numérique. Nous verrons d'abord si l'usage de methodes numériques est réellement indispensable. Nous nous intéresserons ensuite à savoir dans quelle mesure la vision analogique du système en sera perturbée. Nous chercherons finalement à en tirer le meilleur parti.

2.1.1 Nécessité d'une gestion numérique

Nous l'avons vu dans le premier chapitre (section 1.1), les simulations numérique et analogique ont des propriétés très différentes et leur interaction nécessite une structure relativement complexe. L'insertion d'une partie numérique dans notre réseau analogique ne doit donc s'effectuer que pour des raisons bien établies : une très nette amélioration des autres caractéristiques du système ou, plus simplement, s'il n'y a pas d'autre solution envisageable.

La souplesse de la conception numérique

La principale caractéristique recherchée pour le système est une entière reconfigurabilité de la part du réseau. Cela signifie, entre autres, que tout neurone du système doit pouvoir être connecté à tout autre neurone de ce même système. Par

conséquent, la croissance parabolique du nombre de connexions à envisager devient problématique. Pour associer un réseau de n neurones avec un autre réseau de p neurones, le nombre de connexions potentielles à mettre en oeuvre est $(n+p)^2$ et non n^2+p^2. Ainsi, pour un système doté d'une structure analogique, toute l'architecture du système est à recréer, alors qu'il suffit d'une reprogrammation et/ou d'une reconfiguration de FPGA pour une architecture numérique. Mieux encore, un système numérique peut aisément se concevoir pour une taille de réseau variable, auto-détectée à la mise sous tension du système.

Autre avantage d'une architecture numérique : la spécificité du système se retrouve dans la programmation. En analogique, ce sont les composants eux-mêmes qui sont à l'origine de la spécificité du système. Il est résulte un coût et un temps de développement très élevés.

Il faut de plus garder à l'esprit que, quelle que soit sa nature, le système de simulation sera géré par une interface numérique. La récupération d'informations pour étude a déjà été évoquée. Il faut en outre fournir des données de simulation comme les stimuli ou l'activité de fond du réseau. La présence de phénomènes numérisés est donc inévitable dans la structure du système. L'enjeu se situe donc plutôt sur le positionnement de la frontière numérique/analogique.

Des cinétiques synaptiques lentes

Pour optimiser l'usage de la partie numérique, il convient de la placer dans les domaines qui prennent l'analogique en défaut. Parmi ceux-ci, on trouve le stockage des paramètres correspondant aux poids synaptiques, dont il faut éviter toute dérive. Ils fluctuent à chaque potentiel d'action, soit environ toutes les 10 ms pour les activités soutenues. Ces valeurs sont des grandeurs à mémoriser de façon absolue, et non, comme dans le cas des neurones, une variable fluctuante dont il faut maîtriser l'évolution. Les erreurs dues à l'imperfection des composants ne peuvent donc pas se compenser par d'autres phénomènes.

Face à ce problème, la solution numérique est réellement supérieure puisqu'une valeur numérique ne subit pas d'altération. Elle impose par contre que l'ensemble de simulation qui lui est associé soit aussi réalisé en utilisant des méthodes numériques. La partie numérique s'introduit donc au niveau de la gestion de la connectivité du réseau.

2.1.2 Les contraintes de la simulation mixte

Le caractère mixte analogique/numérique du système de simulation que nous allons concevoir perturbe les habitudes de la simulation numérique. Par exemple, la perte de précision éventuellement due à une quantification numérique est ici négligeable puisque le calcul porte sur l'apparition d'évènements ponctuels pour le neurone analogique. Par contre, le temps de développement se trouve rallongé par la spécificité de la méthode choisie pour interfacer la partie analogique.

La méthode d'adaptation des modèles en processus de calcul numérique nécéssite avant tout de s'intéresser aux bibliothèques d'interfaces disponibles. La méthode la plus efficace en termes de mutualisation d'efforts serait d'avoir recours aux bibliothèques du logiciel NEURON (Ref), très utilisé par la communauté des neurosciences computationnelles, et dont il existe une version adaptée au temps réel qui tire parti

d'un DSP (Ref). Cette plateforme est toutefois trop lourde pour permettre un usage temps réel entièrement logiciel sur la simulation d'un réseau d'une certaine taille.

Il n'est par contre pas envisageable d'exiger des connaissances poussées en systèmes électroniques pour utiliser ce simulateur. Il convient donc de s'appuyer sur une plateforme répandue. La solution proposée ici, est l'usage d'un ordinateur utilisant un système d'exploitation. Il faut alors l'équiper de l'ensemble des strates logicielles nécessaires. Pour cela, le langage C est suffisament efficace et répandu parmi les différentes communautés scientifiques pour être le meilleur compromis.

Le dernier point, crucial pour une bonne mise en œuvre d'un système mixte, reste la contrainte temporelle. Il n'est en effet pas possible d'adapter la vitesse de fonctionnement d'un circuit analogique à la capacité de calcul disponible sur le reste du système numérique. En outre, le calcul numérique est séquentiel et génère des temps de latence, *a fortiori* dans une structure aussi complexe qu'un ordinateur muni d'un système d'exploitation, fut-il temps réel[1]. L'enjeu principal sera donc la réduction du coût temporel du traitement des données. La puissance de calcul du matériel en est une composante non négligeable, mais ne peut être considérée comme une réponse pérenne. Nous en deviendrions alors trop dépendants.

2.1.3 Les atouts de la polyvalence

Les efforts fournis pour obtenir un système polyvalent permettent de garder des possibilités d'évolution très ouvertes. Pour une architecture de première génération, il est important de ne pas trop spécialiser le système en imposant une politique de fonctionnement inadaptée à une éventuelle évolution. L'usage d'une bibliothèque d'interface en langage C permet d'envisager une évolution rapide de ce système.

L'implantation des algorithmes en langage C autorise également l'implantation d'une plasticité modifiable à souhait. Il est donc possible, selon la puissance disponible, d'utiliser un modèle plus ou moins précis, ou de choisir s'il faut privilégier les phénomènes d'adaptation au niveau du réseau ou à celui de la synapse.

En utilisant les diverses possibilités d'extensions désormais disponibles sur les ordinateurs, nous pouvons finalement envisager de relier le réseau simulé à divers objets, réels ou virtuels, dont le processeur ne serait alors qu'un relais, remplissant de surcroît les fonctions d'adaptation du signal. Encore une fois, les possibilités d'extension du système ne doivent être limitées que par la puissance de calcul de l'ordinateur, et éventuellement l'imagination de l'utilisateur.

2.2 Le circuit *Trieste*

Ce circuit a été conçu dans le cadre des travaux de thèse de Ludovic Alvado au sein de l'équipe *Ingéniérie des Systèmes Neuromorphiques* du laboratoire IXL [ALV03a], [BOR05]. Bien que n'ayant pas été réalisé par l'auteur, il est tout de même présenté puisqu'il constitue le cœur de simulation de ce système. Sa fabrication a été effectuée en technologie $0,8\mu$m chez Austria*micro*systems et il utilise une surface de silicium de$11,5\text{mm}^2$.

[1] Le caractère temps-réel d'un système d'exploitation ne fait qu'optimiser les delais, mais ne les supprime pas.

2.2.1 Le cœur de simulation

Dans la continuité des travaux de l'équipe, ce circuit utilise le formalisme étendu de Hodgkin et Huxley pour permettre une simulation fidèle au biologique, même au niveau biophysique. Le modèle choisi correspond aux populations majoritaires de neurones peuplant le néocortex. Il considère deux catégories de neurones : les neurones excitateurs (également qualifiés de *regular spiking* ou RS) et les neurones inhibiteurs (*fast spiking* ou FS). Les grandeurs des modèles qui caractérisent les deux catégories, ainsi qu'un rappel des équations régissant ce modèle, sont données en annexe A. La figure 2.1, quant à elle, nous montre le phénomène d'adaptation observable sur le modèle excitateur grâce au canal modulant. L'effet de modulation, qui apparaît ici, est le phénomène le plus précis que nous apporte le modèle par rapport à l'activité d'un neurone biologique.

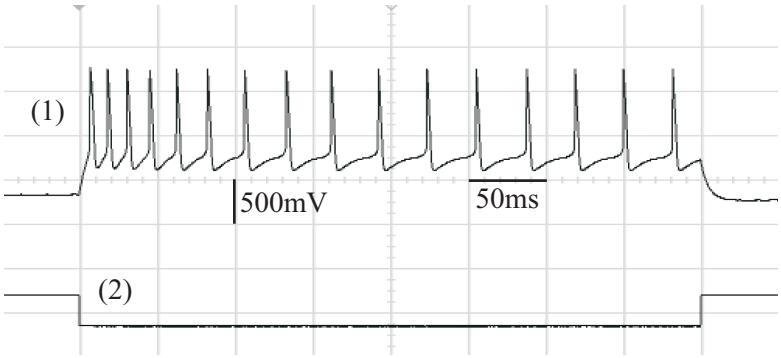

FIG. 2.1 – Phénomène d'adaptation simulé par le circuit *Trieste*

Prévue pour l'intégration de neurones en quantité, l'architecture de ce circuit se veut, avant tout, simple à mettre en oeuvre. Elle est présentée figure 2.2. Par exemple, une majorité des condensateurs nécessaires à l'implantation des différentes constantes de temps est intégrée. Vu la valeur de ces constantes de temps (au-delà de la milliseconde), cette intégration n'est pas aisée, et a nécessité l'usage de courants très faibles.

Toujours dans le souci de s'affranchir d'une mise en oeuvre complexe, les paramètres du circuit sont figés. Certains degrés de liberté sont disponibles pour l'utilisateur final, il n'est cependant pas possible de s'orienter vers un autre type de neurone, même s'il possède des canaux semblables. La dispersion, inhérente à la conception en microélectronique, va à l'encontre d'une bonne précision du modèle. Étant donné que seul intervient l'évènement du potentiel d'action, la forme qu'il peut prendre est du second ordre du point de vue du réseau. Quant aux phénomènes temporels, leur ajustement s'effectue par l'injection d'un courant supplémentaire, dit *courant de compensation* commandé en tension. Plus ce courant injecté directement sur C_{mem} est important, plus le comportement du neurone concerné est rapide. Cette qualification de *compensation* provient de la fuite inévitable des composants impliqués dans la simulation. Le seul degré de liberté que nous accorde le courant de compensation ne permet pas de correspondre très précisément au modèle. Il reste une dispersion de comportement qui sera utilisée pour établir une diversité au sein du réseau simulé.

Par rapport aux grandeurs biologiques, les échelles des signaux ont été adaptées pour mieux convenir aux contraintes électriques. Les tensions sont multipliées par 10,

les courants par 100, les conductances par 10 et les capacités par 10. Dans la suite de cet ouvrage, la référence utilisée (*électrique* ou *équivalent biologique*) sera précisée au cas par cas, selon l'intérêt (électronique ou simulatoire) de la grandeur mesurée. Étant donné que le potentiel de membrane change de signe lors d'un potentiel d'action, le zéro Volt simulé est fixé à $1,8\,\mathrm{V}$ électrique. Électriquement, ce potentiel doit se comporter en tant que masse, c'est à dire que le courant délivré peut changer de signe.

2.2.2 Un circuit numériquement contrôlable

La grande difficulté pour gérer un réseau de neurones analogiques reste la conception d'une interface pour simuler une connectivité au sein du réseau. Le circuit *Trieste* intègre les outils numériques nécessaires à cet interface. Ceux-ci vont d'un comparateur pour repérer les potentiels d'actions, à un ensemble de multiplexeurs analogiques pour gérer le choix entre les modèles implantés.

Il est ainsi possible de régler de façon numérique différents paramètres comme le potentiel de repos du canal de fuite ou la conductance maximale du canal de modulation. Deux entrées sont également destinées au choix du type de neurone (inhibiteur ou excitateur) et celui de la plage de variation du courant de compensation. Ce courant est contrôlé en tension selon différentes échelles : l'une étendue, l'autre précise. L'usage d'un convertisseur numérique/analogique permet de fournir ce paramètre statique.

La figure 2.2 présente la structure du circuit, ainsi que les différentes entrées / sorties nécessaires à son utilisation.

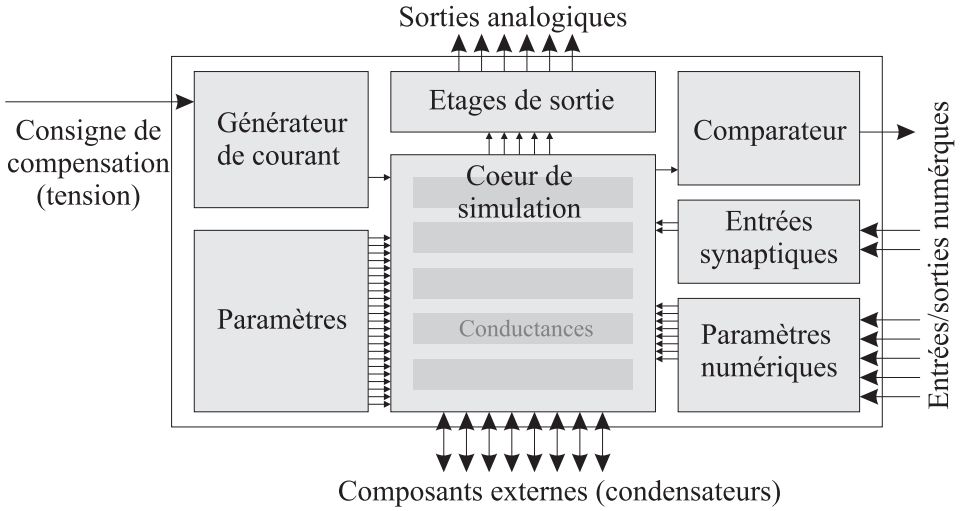

FIG. 2.2 – Schema-bloc du circuit *Trieste*

2.2.3 L'implantation des multisynapses

Le dernier élément d'entrée/sortie du circuit, mais cependant un des plus importants, est l'entrée synaptique. Pour faire face aux spécificités du réseau, ce circuit utilise le modèle de la multisynapse [DES99a]. On trouve donc deux multisynapses

pour chaque neurone, une excitatrice et une inhibitrice, qui correspondent aux deux types de neurones qui seront présents dans les réseaux construits avec ce circuit.

Le signal de commande de la synapse étant numérique, il reste tout de même le problème de l'introduction du poids synaptique lors de la stimulation. Le choix effectué est de coder la valeur du poids par la longueur de l'impulsion numérique chargée de signifier la stimulation. La broche correspondant à l'entrée synaptique possède donc un statut mixte : elle est destinée à être pilotée par un circuit numérique, elle a un fonctionnement binaire, mais elle commande directement des composants analogiques.

Comme illustré par la figure 2.3, l'entrée synaptique commande une tension en rampe lorsqu'elle est à l'état haut (phase A), et autorise ensuite une évolution en exponentielle décroissante lorsqu'elle est à l'état bas (phase B). Cette forme sera toujours continue, de sorte que si le signal est non nul à l'arrivée d'une stimulation, la valeur analogique introduite par la rampe est ajoutée à la valeur du signal.

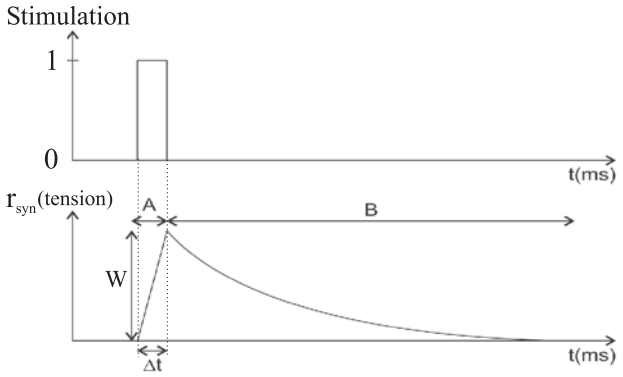

FIG. 2.3 – Fonctionnement d'une entrée synaptique de *Trieste*

Cette implantation n'est qu'une approximation du modèle original de la multisynapse. Lors d'une phase de montée, par exemple, le phénomène de saturation n'est pas pris en compte. Par conséquent, la valeur du signal à la retombée du signal numérique de commande est légèrement surestimée. La différence observable est cependant faible par rapport au modèle théorique.

Un point plus litigieux provient du temps de montée du signal. En effet, la stimulation ne peut être considérée comme envoyée qu'à la descente de la commande, c'est-à-dire au passage de la phase A à la phase B. Ceci introduit un retard égal à Δt. Ce décalage temporel est ici problématique puisqu'il dépend directement du poids synaptique envoyé. Autrement dit, en revenant à une représentation de monosynapses, plus une synapse est forte, plus elle subit de retard. Cette corrélation pourrait faire apparaître des effets indésirables. La rampe de croissance a été réglée pour que la croissance maximale n'excède pas la milliseconde. Cette valeur permet de rester dans l'ordre d'incertitude acceptable sur les temps de propagation.

2.3 Peace, peace, Mercutio, peace !

Nous allons désormais aborder le système de simulation, que nous identifierons désormais par PAX pour *Plasticity Algorithm Computing System*, sous un aspect tech-

nique. Nous l'avons vu précédemment, l'ensemble du système de simulation se compose d'un ordinateur équipé d'une extension contenant les neurones artificiels. PAX était à l'origine le nom de l'extension. Cette appellation fut reprise par ses utilisateurs pour désigner l'ensemble formé par l'extension, l'ordinateur et les logiciels fournis pour leur mise en œuvre. Nous reprendrons ici cet usage pour rester cohérent avec les publications réalisées. Afin d'éviter tout risque de confusion, l'extension matérielle développée spécifiquement sera mentionnée explicitement.

2.3.1 Cahier des charges

Nous avons établi que la partie cruciale de la simulation, le réseau et sa plasticité, doit être réalisée par un ordinateur alors que les éléments de ce réseau, les neurones, seront interfacés sur une extension.

Le point essentiel, nous l'avons vu, est de permettre un accès total aux circuits pour permettre d'interfacer le réseau avec toutes formes d'éléments. Pour permettre une bonne maîtrise des aspects temporels, la méthode la plus adaptée est, hélas, la scrutation. Une perte de performances est donc à prévoir puisque le logiciel de simulation doit en permanence accéder à l'état des neurones. Ce mode de fonctionnement permet néanmoins de laisser l'utilisateur au centre des choix stratégiques par rapport à la simulation souhaitée.

Le nombre de neurones à rassembler dans ce système doit être significatif pour permettre la simulation de petits réseaux. L'objectif n'est cependant pas de réaliser une intégration à grande échelle, mais de permettre d'appréhender les besoins en communication et en puissance de calcul pour un tel réseau. Le nombre de 8 neurones a été choisi.

Parmi les contraintes temporelles, la plus caractéristique est la fréquence de scrutation de l'état des neurones par le logiciel. Le meilleur compromis a été fixé à 10kHz. Cette fréquence, utilisée pour l'échantillonnage des potentiels de membrane lors des mesures *in vitro*, est suffisamment élevée par rapport aux constantes de temps observées au niveau des modèles des neurones. Elle est également techniquement envisageable pour réaliser une boucle temps réel incluant un ordinateur. Les algorithmes de plasticité devront cependant être adaptés pour permettre un calcul rapide et temporellement peu pénalisant.

2.3.2 Le choix du bus de communication

En effectuant une traduction du cahier des charges, nous pouvons calculer le débit nécessaire pour interfacer le système analogique et l'ordinateur chargé de le contrôler. Chaque circuit neuromimétique sort l'état d'un potentiel de membrane sur 1 bit. Pour prélever l'état des 8 neurones, il faut donc lire un octet à chaque cycle. La fréquence de la boucle temps réel étant fixée à 10kHz, le débit nécessaire à la lecture est donc de 10 kilo-octets par seconde. Pour ce qui est du débit nécessaire en écriture, il faut alimenter en données l'entrée des deux multisynapses de chaque neurone. On obtient alors un débit en écriture 20 ko/s. L'interface sera finalement sollicitée à 30 ko/s, ce qui est faible et à la portée de tout port d'entrées/sorties de l'ordinateur.

La contrainte principale n'est donc pas issue des données techniques descriptives, mais de la nécessité de maintenir un faible temps de réponse sur l'ensemble du système. Les faibles temps d'accès sont caractéristiques des interfaces dites rapides, qui

possèdent de hauts débits. Sur un ordinateur, ces interfaces sont le bus PCI (Peripheral Component Interconnect), le bus USB (Universal Serial Bus) lorsqu'il fonctionne dans le mode *High-speed* défini par la version 2.0, ou encore les bus dédiés aux unités de stockage comme les bus SCSI (ANSI X3.131-1994), *Firewire* (IEEE 1394) ou ATA (ANSI NCITS 361-2002). Cependant, de par l'architecture des ordinateurs compatibles PC, l'ensemble des contrôleurs de ces bus de communnication dialogue avec le système central *via* le bus PCI. Ils ne peuvent donc amener que des délais supplémentaires de traitement. Le bus PCI est donc le meilleur choix envisageable.

Notons que le bus USB est malgré tout intéressant pour son faible coût de mise en œuvre tant en production qu'en temps de développement. Cependant, à l'époque de la conception du système PAX, ce bus, ainsi que les composants qui permettaient de le mettre en œuvre, manquaient de maturité. En effet, les bus PCI et USB ne sont ni des normes, ni des standards, mais des ensembles de spécifications gérées par des associations de fabricants.

La pierre d'angle du système sera donc une carte équipée de huit circuits *Trieste* et destinée à être insérée dans un emplacement PCI de l'ordinateur. Sur un ordinateur équipé d'une carte mère de qualité, le bus destiné aux extension est disjoint du bus destiné à l'interfaçage des composants internes. Dans ce cas, la carte de simulation reste seule sur son bus, ce qui garantit une accessibilité et des débits optimaux. Voyons maintenant les caractéristiques techniques du bus PCI pour mieux appréhender la structure de la carte PAX

Description du bus PCI

Le bus PCI a été développé par la société *Intel* pour permettre une augmentation des performances de la communication au sein des systèmes informatiques. Très polyvalent dès l'origine, il a réussi à s'imposer comme un standard incontournable dans de nombreuses applications. Son implantation a été adaptée à une grande quantité de plateformes et, malgré les besoins croissants en performances, il est toujours d'actualité, notamment à travers sa dernière évolution : le PCI-*express*.

Il est essentiellement défini comme un protocole d'échange associé à quelques recommandations. Chaque implantation ajoute ensuite ses spécifications matérielles et précise les fonctions indispensables, optionnelles ou inutiles. Il existe, par exemple, des versions industrielles ou embarquées de cette norme.

Dans les ordinateurs compatibles PC, il est mis en œuvre dans sa version de base : un bus de données de 32 bits cadencé à 33 MHz, ce qui offre un débit théorique de 132 Mo/s. Il n'est cependant pas possible d'obtenir un tel débit dans les faits puisque les envois d'adresses et de données sont multiplexés temporellement sur les mêmes signaux. Cela signifie que chaque accès s'effectue sur au moins deux cycles : un pour l'adresse, un pour les données. Pour améliorer les débits, des envois en trains sont possibles : on utilise un cycle pour envoyer une adresse de début, les cycles suivants contiennent des données. Cette technique nécessite un compromis entre le débit réel et la latence d'accès au bus. En effet, des longs trains de données augmentent considérablement le débit, mais, comme ils ne peuvent être interrompus, la latence d'accès croît dans les mêmes proportions. Par exemple : un train unique ayant une durée d'une seconde, permet d'approcher de très près le débit théorique, cependant, aucun autre périphérique ne peut demander l'accès au bus durant cette même seconde.

Les contraintes d'universalité amènent ce bus à gérer des tailles de mot variables.

Il est possible de gérer les données sur 8, 16 ou 32 bits. Les adresses sont codées sur 32 bits pour des mots de 8 bits, ce qui permet d'adresser 4 Go de données. En fonctionnement sur des données de 32 bits, l'adresse est réduite sur les 30 bits de poids fort pour garder une cohérence.

Pour faire cohabiter les différents types de données, le bus fonctionne comme quatre canaux de 8 bits fonctionnant en parallèle. Les signaux de contrôle sont communsi, à l'exception de ceux de validation des données. Pour des données codées sur 16 bits, l'adresse est codée sur 31 bits afin de couvrir la même quantité de mémoire. Cette adresse est cependant décomposée. Les 30 bits de poids fort sont envoyés comme pour un accès en 32 bits, le bit de poids faible est, quant à lui, codé dans les canaux utilisés pour écrire. Pour communiquer sur une adresse 31 bits paire, on utilise les 16 bits de poids faible, et pour une adresse impaire, on utilise les 16 bits de poids fort sur cette même adresse de 30 bits. De même, pour des données 8 bits, on utilise l'adresse de base d'un transfert 32 bits en n'activant que le canal correspondant.

La grande nouveauté apportée par le bus PCI a été une procédure de reconnaissance et de configuration automatisée, le *Plug'n Play*. Chaque périphérique est relié de façon exclusive au maître du bus. La négociation qui s'en suit lui permet alors de déterminer les besoins de chaque périphérique et de leur assigner jusqu'à six plages d'adresses pour les transferts. Cette étape de configuration permet également de collecter et d'envoyer au système d'exploitation les fonctions disponibles à travers le bus. Un système de références codées sur 64 bits permet d'identifier chaque périphérique de manière unique : 4 mots de 16 bits chacun identifient le constructeur, le développeur, la référence produit et son niveau de version. Le système peut alors, selon ses ressources, utiliser un pilote générique pour fonctionner dans un mode de compatibilité ou utiliser un pilote dédié pour optimiser les performances globales.

Si la communication à travers le bus est relativement accessible, la gestion de la séquence de reconnaissance est, elle, complexe. Elle nécessite de gérer l'ensemble des capacités de transfert ainsi qu'un protocole spécifique d'accès puisque la répartition des adresses d'accès n'est pas encore effectuée. Cette phase ne permet que peu de diagnostic en cas de problème puisque le système reste bloqué, et ne permet pas l'envoi de trames de test. Le recours à un composant dédié, réalisant un pont vers le bus PCI, constitue donc le choix le plus judicieux.

2.3.3 L'ordinateur d'accueil

Bien que secondaire, l'ordinateur d'accueil mérite tout de même quelques lignes. Lorsqu'il a été assemblé, il regroupait l'ensemble des techniques innovantes susceptibles d'optimiser les performances de la carte PAX en termes de communication. Le choix s'est porté sur une carte mère dont les composants nécessaires avaient les meilleures performances possibles pour ne pas générer de délais de traitement, mais, surtout, étaient intégrés. En effet, le bus PCI qui accueille les cartes d'extension est différent de celui sur lequel sont interfacés les composants intégrés. Ainsi, les contrôleurs des interfaces Ethernet et du disque dur n'entrent pas en concurrence avec la carte dédiée pour la communication. Les autres périphériques, quant à eux, ont été désactivés pour éviter tout usage superflu du gestionnaire d'interruptions.

Unité de calcul Une grande quantité de mémoire permet de minimiser les accès au disque, et le choix du processeur s'est porté sur le Pentium 4 équipé de la fonction

dite d'*hyperthreading*. Contrairement aux jeux d'instruction MMX, SSE ou encore *3D-now* qui ne sont utiles qu'avec un développement spécifique et lourd, cette astuce accélère le système général en optimisant les unités de calcul. En effet, les processeurs conçus chez *intel* utilis(ai)ent[2] des fils d'éxécution ou *pipelines* à grand nombre de niveaux pour mieux monter en fréquence. Lors d'un branchement conditionnel, le réamorçage éventuel du pipeline est donc très coûteux en temps. De nombreux efforts ont été investis pour la conception de circuits de prédiction de tests. Cela permet, lorsque la prédiction est juste, d'économiser le temps de réamorçage du pipeline. Une autre technique suivie a été de précalculer en parallèle les deux éventualités. Mais l'apport de telles structures n'est semble-t-il pas suffisant. La solution apportée par l'*hyperthreading* consiste à offrir deux *pipelines* pour une unité de calcul, chacun des deux étant déclaré comme un processeur à part entière. Le système d'exploitation répartit alors les processus sur les deux processeurs qu'il détecte. Lorsqu'une file d'éxécution est en amorçage, l'autre utilise l'unité de calcul qui, sans ce double usage, serait vacante.

Dans notre cas, ou il n'y a qu'un seul processus de simulation, mais où le temps de transition entre le système d'exploitation et la simulation doit être optimal, cette technique offre beaucoup de gain.

2.3.4 La carte PCI

La carte PAX s'architecture autour d'un FPGA : un *Spartan IIe* de marque *Xilinx*. Étant donné le modeste nombre d'ASICs embarqués, ils sont directement interfacés avec le circuit programmable grâce à leurs entrées/sorties numériques. Le FPGA est également relié au pont vers le bus PCI. L'ensemble du système est cadencé à 64 MHz. La figure 2.4 présente la structure de la carte PAX ainsi que les principaux composants présents, notamment une mémoire SRAM pour un éventuel stockage de données (bruit synaptique, historique du réseau *etc*), des diodes d'état pour un déverminage direct, ainsi que des connecteurs regroupant 60 entrées/sorties pour permettre une évolutivité du système. La carte réalisée est présentée en figure 2.5

La présence sur cette carte de composants analogiques implique également de grandes exigences sur les alimentations, qui, destinées à une implantation dans un système entièrement numérique, ont nécessité un grand effort de filtrage et de régulation. Une séparation franche a de plus été effectuée entre les deux domaines pour éviter tout couplage avec les signaux numériques.

Le pont dédié à la connection au bus PCI est assuré par le composant *PCI9056* de la société *PLX*. Il permet d'utiliser un bus local à configuration figée. L'ensemble des paramètres nécessaires à la séquence de démarrage et d'auto-configuration est mémorisé dans une mémoire non volatile externe. Ce composant peut gérer trois normes de bus internes pour s'adapter à différents microcontrôleurs. Il est par exemple possible de choisir si les adresses sont multiplexées ou non. La différence entre la fréquence locale et celle du bus PCI est également gérée. Enfin, le pont se charge de la translation[3] d'adresses, ce qui permet au bus local d'être figé.

[2] La stratégie commerciale de la société a été entièrement revisitée depuis les deux dernières années. Face aux difficultés rencontrées pour franchir les 4GHz et les bonnes performances de l'architecture *Pentium* II/III/M, la communication s'effectue désormais sur la capacité de calcul fournie par Watt consommé

[3] La traduction vers le français du mot anglais *translation* est ici problématique. Le mot littéraire officiel est *traduction*. Mais dans notre contexte, les adresses sont décalées d'une valeur fixe. Elles

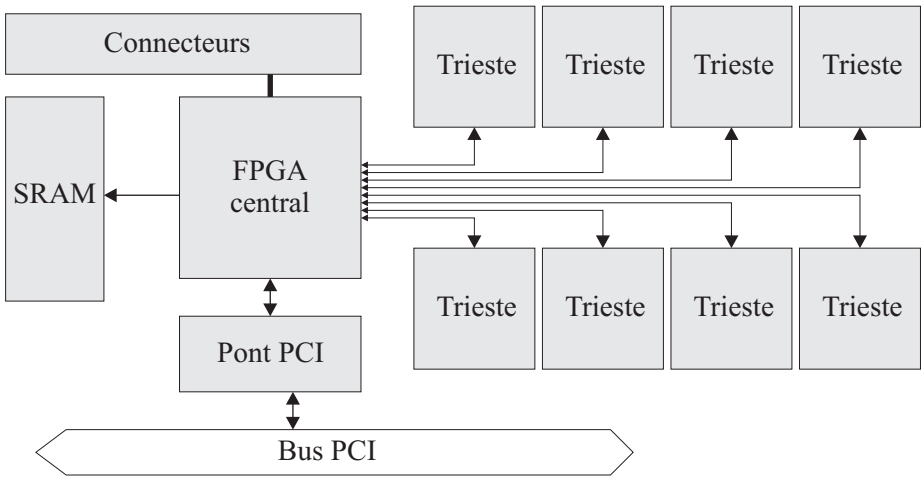

FIG. 2.4 – Schéma de principe de la carte PAX

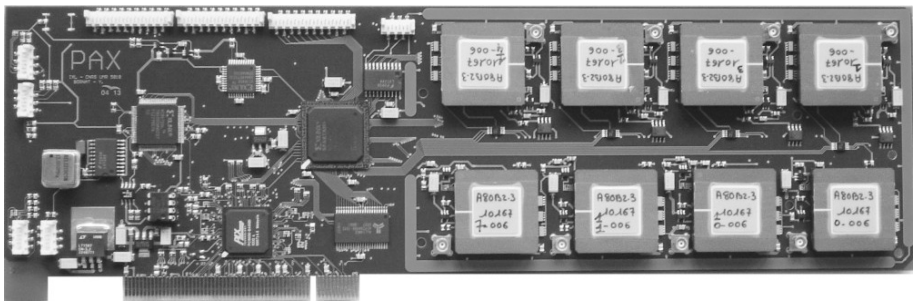

FIG. 2.5 – Photographie de la carte PAX

2.3.5 Programmation embarquée

La programmation du FPGA se décline en deux catégories. La première relève du relais des informations et des interfaces spécifiques de chaque composant présent sur la carte. La seconde revêt un aspect plus orienté vers l'optimisation des ressources disponibles. Le FPGA est mis à profit pour intégrer au plus bas niveau des fonctions dont la programmation logicielle aurait coûté du temps de calcul.

La figure 2.6 résume l'architecture de la configuration du FPGA embarqué sur la carte PAX, la répartition des fonctions par adresse est donnée en annexe B.

Une interface souple et reconfigurable

En tant que relais, le FPGA stocke et met en forme les données pour les rendre disponibles sur la carte. Il peut s'agir de convertir des données parallèles en série pour les convertisseurs numérique/analogique, de multiplexage avec une fonction de maintien des données pour les paramètres des circuits ou encore de la conversion d'un entier en largeur d'impulsion, pour permettre une gestion du poids sur les entrées synaptiques. La communication s'effectue en émulant le comportement d'une mémoire.

sont donc *translatées* au sens algébrique selon l'unique axe possible, celui de leurs valeurs

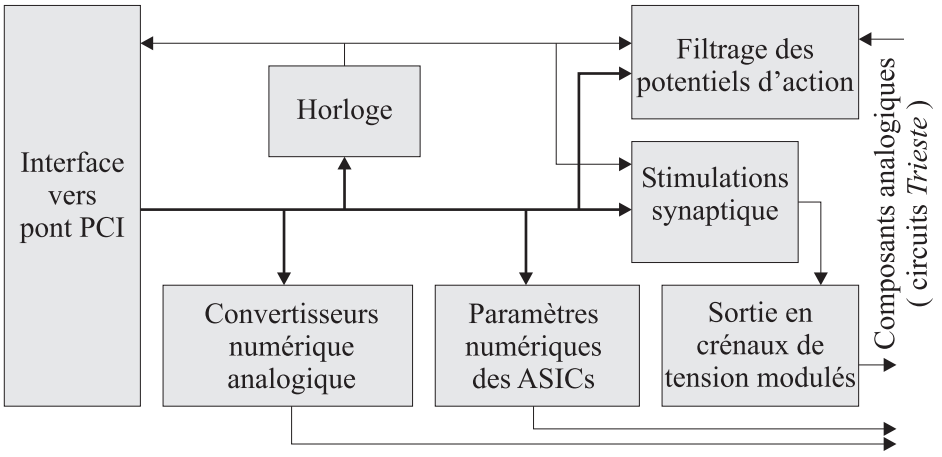

FIG. 2.6 – Structure numérique implantée sur le FPGA de la carte PAX

Lorsque qu'un entier est reçu par le FPGA, il est à la fois mémorisé pour permettre une relecture et transmis au composant concerné. La sélection de la fonction s'effectue par l'adresse d'écriture. L'ensemble des données accessibles en lecture est mémorisé dans des registres qu'il suffit de renvoyer sur le bus lorsqu'ils sont consultés.

La partie gérant le relais d'interface consomme une part non négligeable des ressources de programmation. En effet, du point de vue logiciel, le système est assimilé à une zone mémoire. Il se comporte donc comme tel, et n'utilise aucune optimisation au niveau matériel, ce qui permet à quiconque habitué à la programmation en C, de contrôler le système sans avoir à remettre en cause les nombreux automatismes liés aux zones mémoire. Cela signifie que les données écrites peuvent être relues à la même adresse. En outre, l'écriture et la lecture sont indépendantes du mode d'accès aux données. En d'autres termes, pour écrire une donnée codée sur 32 bits, l'usage d'un transfert sur 32 bits doit être équivalent à deux transferts sur 16 bits, ou encore quatre sur 8 bits. Cette mixité des politiques d'accès ainsi que la mémorisation systématique de la valeur de chaque paramètre sont gourmands en blocs configurables sur le FPGA.

Optimisation de la puissance de calcul disponible

Les fonctions dites secondaires, si elles sont indispensables au fonctionnement du système, ne nécessitent pas une implantation matérielle. Le qualificatif de secondaire doit donc se percevoir du point de vue implantation matérielle, et non du point de vue de la nécessité fonctionnelle. Par contre, dans un système temps réel, une capacité de calcul non exploitée s'assimile à une perte de performances, il convient donc de prétraiter les informations au maximum pour éviter que cette charge ne soit imputée au processeur central.

Parmi ces fonctionnalités, la gestion de l'horloge de synchronisation du système est prioritaire. Non seulement la génération logicielle d'une telle fonction est très gourmande en ressources du fait de la présence de nombreux services systèmes (gestionnaire de tâches, générateur d'exceptions, *etc*), mais elle est relativement imprécise à cause des temps de réponse du système. Une horloge programmable est donc configurée à partir de la fréquence de fonctionnement de la carte PAX. Dans le principe, un registre est décrémenté à chaque front d'horloge, lorsque ce dernier s'annule, une

interruption est envoyée au système. Dans le même temps, le registre est rétabli à une valeur définie en paramètre. La période de ce système d'interruption est donc réglable en fonction des besoins du protocole de simulation. Le registre-compteur est également accessible en lecture, ce qui permet de réaliser une mesure précise des performances. L'intégration de cette fonction permet en outre de synchroniser la validation de certains paramètres avec l'horloge générale du système au niveau du FPGA. Il s'agit principalement des entrées synaptiques.

L'usage d'un FPGA au plus bas niveau permet également un filtrage des données. En effet, il est nécessaire de transformer l'état du potentiel de membrane en évènement de potentiel d'action. Cette détection fait appel à des combinaisons logiques simples. Un processeur devra cependant, pour les traiter, faire appel à des mémoires externes, qui génèrent des temps d'attente pour les accès. La perte d'information sur la durée du potentiel d'action que l'on subit au bas niveau est donc compensée par un allègement significatif de la charge du processeur. De plus, ayant accès aux signaux de l'interface PCI, il est possible d'attendre que le logiciel ait réellement eu accès à l'évènement avant de le remettre dans son état de repos.

2.3.6 L'environnement logiciel

L'environnement logiciel de l'ordinateur hôte du système est basé sur le noyau *Linux* dans sa version la plus récente à l'époque du développement (Référence 2.4.22). Le noyau Linux n'ayant pas les propriétés de temps réel nécessaires, la patch *adeos* du projet RTAI y a été appliqué. Le système final obtenu est basé sur la distribution *Debian*. Le grand intérêt d'une base logicielle libre, comme *Debian*, est la possibilité de supprimer les services inutiles pour alléger la charge du processeur. Le système final gère moins de dix processus. En théorie, le temps réel évite le besoin de libérer les ressources. Il faut cependant garder à l'esprit qu'un système temps réel ne peut, au mieux, que répondre rapidement à une requète ; si la puissance de calcul disponible est insuffisante, il ne sera d'aucun secours. Bien que les processus de simulation soient prioritaires, il faut pouvoir les contrôler, et l'interface utilisateur n'est pas temps réel.

Une description de la répartition des calculs sur les différents éléments de PAX est donnée par la figure 2.7

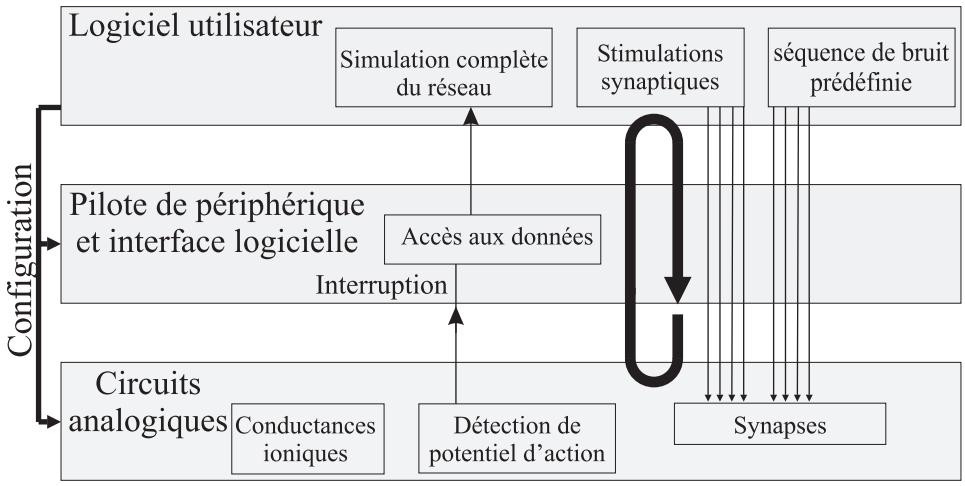

FIG. 2.7 – Répartition des calculs dans les strates de PAX

Vers une validation de la structure matérielle

Pour limiter le temps consacré à la recherche et à la correction d'erreurs, une validation au niveau matériel des fonctionnalités de la carte doit être effectuée. Le pilote logiciel réalisé dans ce but permet des accès à de bas niveaux de description. Il ne dispose que de quatre instructions : une pour la lecture sur 32 bits, et trois pour l'écriture dans chacun des modes 8, 16 ou 32 bits. Une description a été réalisée en langage C pour l'usage à partir de programmes élaborés. Mais surtout, ces fonctions de lecture et écriture ont été réalisées en tant que programmes isolés. Il est alors possible de les utiliser directement ou à travers des *scripts* dont l'écriture est rapide et facile.

Ces outils ont permis dans un premier temps de vérifier le bon fonctionnement de la carte. Ils ont surtout contribué au test des différents blocs implantés en VHDL. Aucune mesure réaliste sur des réseaux de neurones n'a pu en découler, puisque le fonctionnement ne s'effectue pas dans les conditions de simulation. L'ensemble de fonctions réalisé a néanmoins permi de démarrer sereinement la constitution d'une structure logicielle plus évoluée.

Le fonctionnement temps réel

La gestion du temps réel n'étant pas native dans le noyau *Linux*, le recours à *adeos* (fonctionnalité du projet RTAI) modifie les hiérarchies de fonctionnement. Au lieu de modifier en profondeur l'ordonnanceur des tâches du système principal, c'est un autre système d'exploitation temps réel qui prend place, et qui, en l'absence de requête temps réel, éxécute le noyau *Linux*. *Adeos* n'a alors aucune connaissance des processus utilisateurs puisque ces derniers sont gérés par Linux, de même, dès qu'une fonction est chargée dans *adeos*, Linux en perd toute trace. Les communications s'effectuent alors au travers d'un module spécifique qui met en œuvre l'API (*Application Programming Interface* [4]) *ad hoc*.

Ce mode de fonctionnement pose tout de même le problème de la fiabilité du système. Un logiciel en cours de conception recèle de nombreuses erreurs. Le noyau est (plus ou moins) capable de les gérer et d'interrompre l'exécution en les signalant à l'utilisateur, le tout en garantissant l'immunité des autres programmes. Mais dès que le programme est, lui aussi, éxécuté au niveau du noyau, en tant que module, ou dans *adeos* en tant que tâche temps réel, il n'existe plus de possibilité de rattrapage d'une erreur. Le développement est donc long et réservé aux programmeurs systèmes avertis.

La structure de fonctionnement

Le fonctionnement choisi est orienté vers un processus qui, du point de vue de l'expérimentateur, ne quitte jamais le mode dit *utilisateur*, c'est-à-dire le mode courant où sont valables les règles de sécurité. L'ensemble des accès et des opérations délicates est géré par une bibliothèque fournie, ce qui donne le découpage logiciel décrit ci-dessous.

[4] Une interface de programmation est la description de l'ensemble des fonctions qui permettent l'usage d'un composant. Elle constitue un cahier des charges portable et elle est indépendante des bibliothèques qui la mettent en œuvre.

- Un pilote de périphérique : il s'agit d'un ensemble de fonctions à charger dans le noyau et qui permet des accès binaires à la carte PCI.
- Une bibliothèque de fonction : elle fait appel au pilote de périphérique. Elle contient des fonctions dont les noms sont explicites, et dont les paramètres ont une signification du point de vue de la simulation. Elle effectue principalement une traduction des actions en adresses d'accès et une conversion des paramètres en mots binaires. Elle autorise néanmoins des accès bas niveau, ce qui laisse à l'utilisateur la liberté de changer de politique d'accès.
- Le programme de simulation : écrit par l'utilisateur final selon un schéma prédéfini, il permet la mise en œuvre du réseau et contient le calcul de plasticité.

Le principe est ici de dissimuler l'ensemble des considérations temporelles dans la bibliothèque de fonctions. L'utilisateur réalise alors le programme de simulation en suivant une structure prédéfinie, et peut le tester en toute sécurité. Lors de la simulation, il peut ensuite faire appel à un utilitaire qui se charge d'éxécuter ce même programme avec une priorité extrême.[5]

Le programme de simulation doit suivre la structure suivante : une première partie se charge de configurer le réseau, c'est-à-dire de fournir à chaque neurone l'ensemble des paramètres dont il a besoin pour fonctionner dans les conditions voulues. La seconde partie se répète jusqu'à la fin de la simulation et réalise dans l'ordre :

- la lecture des informations de potentiels d'action ;
- la préparation des stimulations synaptiques qui en découlent ;
- l'ajout éventuel de bruit de fond synaptique ;
- l'envoi des stimulations sur les synapses ;
- le calcul de plasticité et la mise à jour des poids synaptiques.

La figure 2.8 représente la succession des étapes dans un cycle de simulation.

La gestion de l'attente des interruptions n'est pas accessible pour l'utilisateur, elle est effectuée lors de la lecture des potentiels d'action. Pour ce faire, la fonction de lecture correspondante place le processus en état de sommeil si besoin est. C'est alors le pilote qui le réveillera à la prochaine requête de la carte PCI. Avec ce principe, le processus reçoit les états des neurones dans un temps optimal après le déclenchement de l'interruption par l'horloge.

Des optimisations sont envisageables pour augmenter la capacité de ce système. En effet, la contrainte de calculer la plasticité pendant une seule période d'échantillonnage est très forte et ne permet pas d'envisager de dépasser des réseaux de 10 ou 15 neurones. Cette technique limite à la fois l'algorithme utilisé et la taille du réseau simulé puisque pour le potentiel d'action d'un neurone donné, il faut mener le calcul sur l'ensemble des connections dans lesquelles il est impliqué. Une méthode plus souple serait de mener deux processus en parallèle : le premier, rapide, se charge d'observer le réseau et d'envoyer les stimulations, tandis que le second se charge du calcul plastique, avec des contraintes temporelles plus faibles. La mise en place d'une telle structure nécessite cependant des compétences poussées en ingéniérie informatique, et chacun, l'auteur comme les autres, doit savoir situer ses limites.

[5] Le mot *extrême* n'est pas usurpé, il s'agit d'une priorité dite *real-time* qui surpasse l'ordonnanceur traditionnel des processus et offre le maximum de ressources. À titre d'exemple, cette priorité surpasse la gestion de l'affichage graphique et du réseau. Ces interfaces sont traditionnellement utilisées pour contrôler la machine, d'où la difficulté de déverminer.

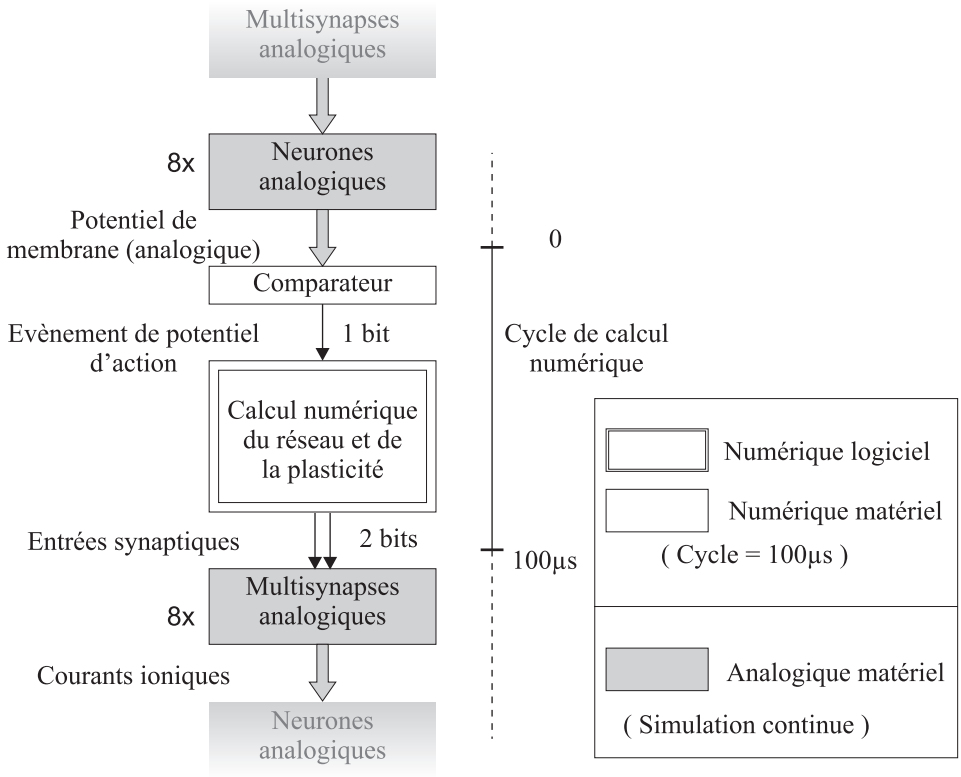

FIG. 2.8 – Succession des différentes étapes de calcul lors d'une simulation

2.4 Résultats

Nous venons de parcourir les différentes étapes qui nous mènent à la réalisation d'un système complet. Voyons maintenant son comportement. Il s'agit dans un premier temps de vérifier que ce dernier est conforme aux spécifications de départ, pour ensuite mettre à l'épreuve le principe même du simulateur. Il sera alors possible de réaliser des simulations ayant une signification biologique.

2.4.1 Représentation des grandeurs d'un neurone

La première validation nécessaire est celle de l'environnement de fonctionnement des circuits analogiques. Il faut effectivement s'assurer que les circuits *Trieste* sont bien isolés du bruit des circuits numériques et de l'alimentation de l'ordinateur.

La figure 2.9 présente les différentes grandeurs produites par un circuit en fonctionnement sur le système. Il s'agit du potentiel de membrane du neurone (A) ainsi que des courants de chacune des conductances. On remarque d'abord que le circuit fonctionne dans de bonnes conditions. Les courants de fuite et modulant sont sujets à un niveau relatif de bruit plus important. Il s'agit d'un effet d'échelle : ces signaux sont très faibles, ce qui déteriore le rapport signal à bruit.

Nous pouvons cependant remarquer un léger décrochage sur chaque courant lors du pic du canal sodium (B) et donc de la montée du potentiel de membrane. Ce décrochage n'a pas été observé sur le dispositif de test des circuits, ce dernier est directement alimenté par des générateurs d'instrumentation. Nous pourrions donc

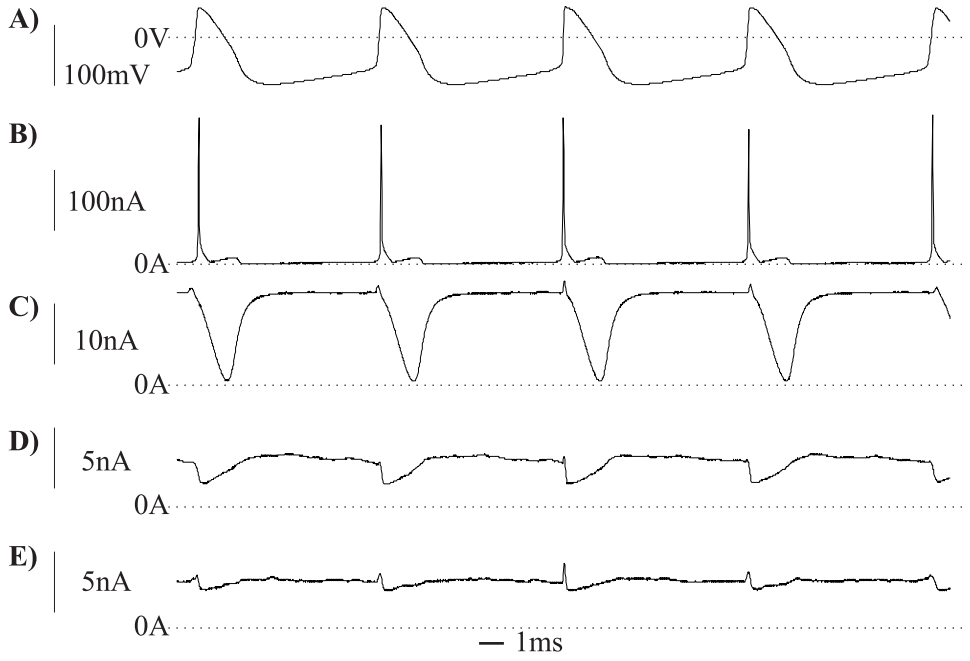

FIG. 2.9 – Le potentiel de membrane (A), ainsi que les différents courants ioniques pour le circuit *Trieste* : ions sodium (B), potassium (C), de fuite (D) et modulant (E).

être, dans ce cas, confrontés à une baisse des performances de l'alimentation lors de cette demande en courant. Encore qu'aucune autre observation ne permette de confirmer cette hypothèse.

Un léger bruit de commutation est également observable lorsque le potentiel de membrane traverse le 0 V biologique (soit 1, 8 V électrique). Il peut résulter des commutations du comparateur sur le potentiel de membrane, il est cependant plus probable que cette commutation provienne du changement de cinétique du canal modulant (Cf annexe A : un paramètre change de valeur selon le signe du potentiel de membrane)

2.4.2 Fréquence par rapport à la stimulation

La première utilisation fonctionnelle du système consiste à tracer précisément la fréquence d'activité des neurones par rapport à leur stimulation. Les modélisateurs parlent de *courbe f/i*. Cette courbe est, avec la forme des potentiels d'action, révélatrice de la bonne implantation du modèle dans le circuit. Bien qu'insuffisante pour prouver que le neurone simulé est représentatif du modèle, elle permet une sélection significative des circuits. Un fonctionnement représentatif se traduit par une évolution croissante de la fréquence du neurone avec sa stimulation. La figure 2.10 représente ces courbes tracées par un programme dédié sur les différents modèles implantés par un circuit.

Nous pouvons voir que la famille de courbes correspond à l'évolution attendue du modèle : une croissance avec un ralentissement pour les valeurs élevées. La valeur quantitative de la stimulation n'est pas significative. En effet, elle permet à la fois

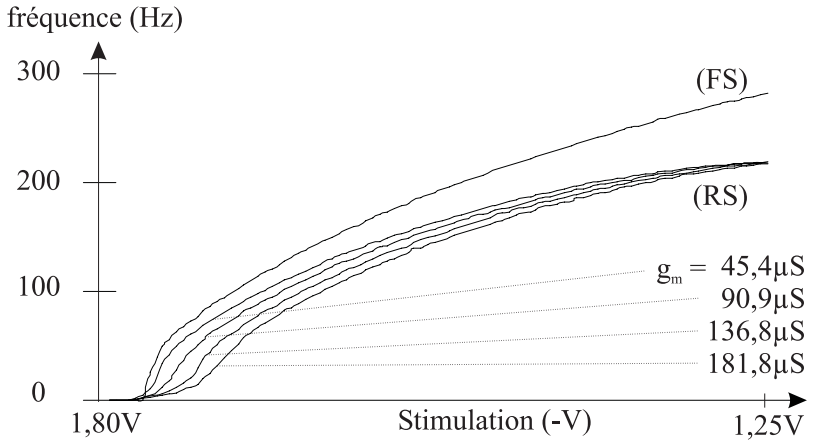

FIG. 2.10 – fréquence d'oscillation du neurone simulé par un circuit *Trieste* en fonction de la consigne de compensation.

de stimuler le neurone et de compenser les dérives dues à l'imperfection des composants intégrés. Il n'est donc pas possible de faire la part des choses entre le courant injecté pour stimuler et celui qui ne fait que compenser des pertes. Il existe bien une valeur nominale définie par conception, mais les mesures la montrent peu significative [ALV03a]. Par contre, il reste vrai que le courant réellement injecté est linéaire par rapport à la tension de consigne, avec une constante négative.

Pour être complète, une procédure de validation de circuits devrait intégrer une méthode de caractérisation dynamique. Une méthodologie fiable, permettant de trier les circuits fonctionnels et défectueux selon ce principe, est, hélas, difficile à élaborer. Il reste encore beaucoup de chemin à parcourir pour y parvenir.

2.4.3 Petits réseaux entre amis : Démonstration de fonctionnement

Le but de cette section n'est pas de fournir des résultats sur les usages du système dans le cadre des neurosciences, mais de mettre en évidence les fonctionnalités du système. Les mesures mettent en œuvre quatre neurones pour des raisons techniques d'acquisition des potentiels de membrane analogiques.

Sommation spatiale - Validation des multisynapses

La sommation spatiale se conçoit par opposition à la sommation temporelle. On ne cherche pas à accumuler des stimulations au cours du temps, mais à les combiner à un moment donné. Il est important de rappeler ici que le principe même du modèle de la multisynapse repose sur la transformation de toute somme spatiale en somme temporelle. Ce réseau est présenté en figure 2.11.

Dans ce réseau, les connexions et les neurones sont excitateurs. Les neurones A,B et C (présynaptiques) subissent une faible stimulation, ce qui génère une activité lente. Le neurone D n'est pas stimulé. Les connections sont instantanées et excitatrices. Leur poids est modéré ; ainsi, pour générer une activité sur D, il est nécessaire que les trois neurones présynaptiques soient actifs dans une fenêtre temporelle très limitée.

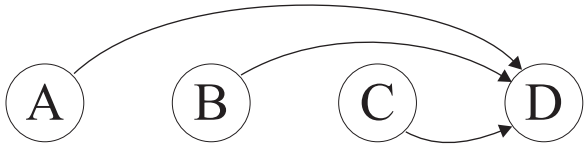

FIG. 2.11 – Configuration du réseau mettant en œuvre le principe de sommation spatiale.

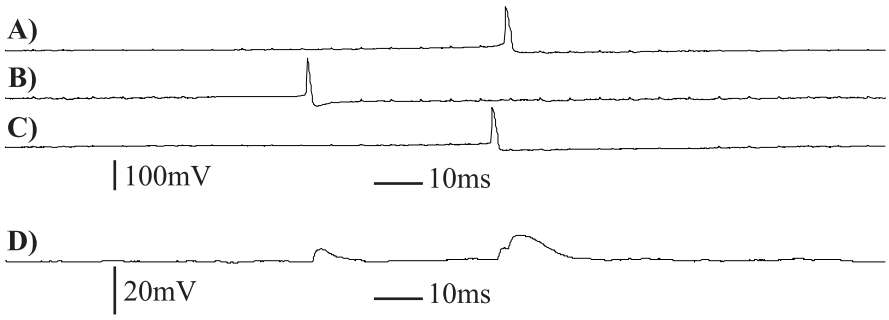

FIG. 2.12 – Évolution des quatre neurones du réseau de combinaison spatiale simulé par PAX.

La figure 2.12 montre les potentiels de membrane de chaque neurone. Aucune activité n'est constatée sur le neurone D. L'échelle a été dilatée pour ce dernier afin de mettre en évidence l'influence des stimulations consécutives au fonctionnement du réseau. Il est, par exemple, possible d'observer l'effet d'une connection simple (celle du neurone B), ainsi que celui de deux connexions proches (neurones A et C). La figure 2.13 nous montre la réaction du neurone D lorsque A, B et C sont actifs dans une fenêtre temporelle suffisament réduite.

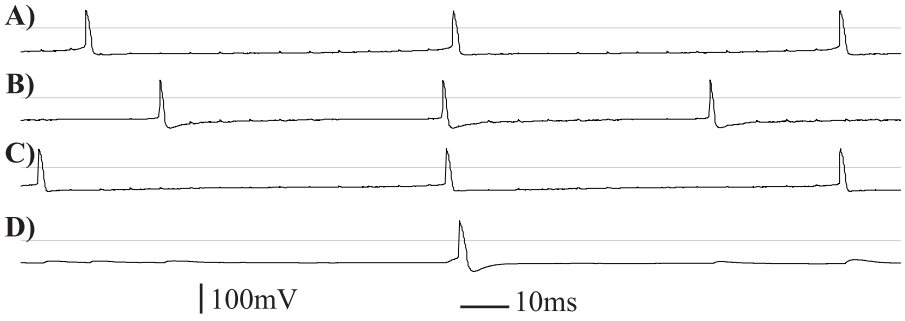

FIG. 2.13 – Comportement des quatre neurones du réseau avec génération de potentiel d'action due à la concordance de l'activité des trois neurones présynaptiques.

Comme prévu, on observe la génération d'un potentiel d'action au niveau du neurone D lorsque les trois neurones présynaptiques sont actifs dans la même fenêtre temporelle.

On observe dans ces deux tracés, ainsi que dans certains suivants, des petits pics de bruit sur les neurones A et B. L'origine de ce résidu d'une fréquence de 100 Hz est

d'autant plus inconnue qu'aucun élément du système ne génère de telle fréquence. En outre, ce bruit n'a pas pu être reproduit lors de séances de mesures successives. Le plus vraisemblable est une perturbation sur les appareils de mesure par un phénomène externe. La pollution du système par le 50 Hz du secteur n'est, de toute évidence, pas à craindre puisque l'alimentation est à découpage ; les sources de bruit majeures se situent donc dans des domaines de fréquences bien plus élevées.

Un chenillard de neurones

Ce deuxième réseau, présenté en figure 2.14, est le résultat de la déformation électronicienne qu'a subie l'auteur. Le chenillard de neurones est la transcription d'une structure numérique, il est inadapté pour représenter des phénomènes biologiques. Ce réseau permet simplement d'illustrer le fonctionnement de connexions associées à un délai de propagation.

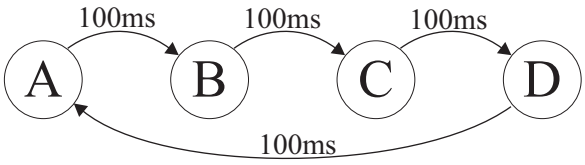

FIG. 2.14 – Configuration du réseau de chenillard. Les connexions et les neurones sont excitateurs. Les connexions subissent un délai de propagation arbitrairement fixé à 100 ms.

Ici aussi, nous utilisons quatre neurones excitateurs. Cette fois, les connexions sont trop fortes pour prétendre être biologiquement réalistes. Ainsi, un potentiel d'action génèrera nécessairement un autre potentiel d'action sur chaque neurone vers lequel il est connecté. Des neurones subissent un faible courant de stimulation (A, C), la lente activité ainsi produite amorce la chaîne, les autres sont au repos. Désormais, chaque connexion synaptique subit un délai de propagation de 100 ms. Ce délai est généré par le logiciel de gestion du réseau qui attend le temps indiqué avant d'envoyer la stimulation correspondante à l'activité mesurée. Il est nécessaire que la période de l'activité du réseau soit plus longue que la période réfractaire de chaque neurone, autrement, la propagation du potentiel d'action n'est pas garantie. La figure 2.15 nous montre alors le résultat produit sur les potentiels de membrane.

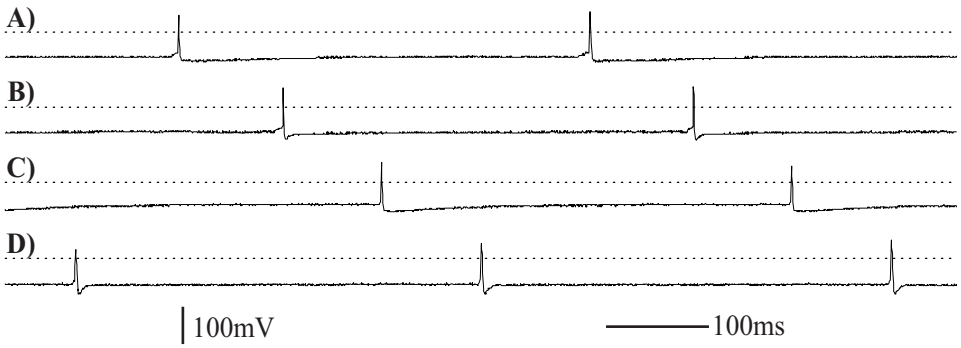

FIG. 2.15 – Activité des quatre neurones A,B,C et D du réseau du chenillard

Nous pouvons voir que le réseau fonctionne comme attendu. Il est cependant important, pour observer un chenillard correct, qu'aucun neurone n'ait de fonctionnement plus rapide que la fréquence du chenillard. Si un tel cas se produisait, le neurone le plus rapide se déclencherait avant d'être stimulé, et le réseau entier serait synchronisé sur lui.

Il n'est par contre pas nécessaire que chaque neurone présente une activité isolée ; un seul potentiel d'action est suffisant pour amorcer le réseau. L'origine de ce potentiel d'action peut être spontanée ou provoquée, l'activité finale restera inchangée. L'arrêt de cette activité auto-entretenue se provoque par une inhibition suffisante sur l'un des neurones.

Reconnaissance temporelle

Le dernier réseau de démonstration réalisé dans cette partie regroupe l'intérêt des deux précédents. La figure 2.16 qui le décrit nous montre en effet à la fois une somme spatiale et des connexions à retard, synonymes d'un aspect temporel.

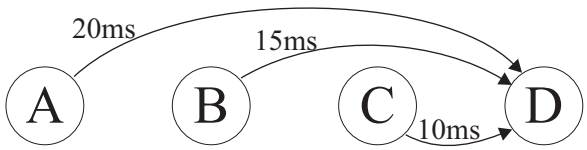

FIG. 2.16 – Réseau de reconnaissance de stimuli temporel.

Comme précédemment, les neurones A, B et C sont excitateurs et faiblement stimulés. Quant au neurone D, il a besoin de l'apport simultané des trois connexions entrantes pour s'activer. Par contre, vu les délais mis en jeu, ce n'est pas une activité simultanée des trois neurones présynaptiques qui provoquera cette situation, mais une séquence d'activation bien définie : A, puis B et enfin C, à environ 5 ms d'intervalle. Le résultat de ce réseau est présenté en figure 2.17

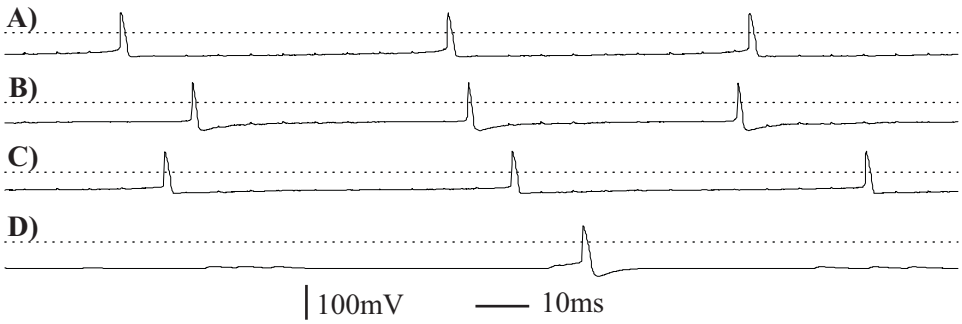

FIG. 2.17 – Comportement des quatre neurones impliqués dans le réseau de reconnaissance de stimuli temporel

Cette acquisition nous montre que le réseau fonctionne effectivement comme nous nous y attendions, mais là n'est pas l'essentiel. Il est relativement immédiat de remarquer que la succession des activités des neurones présynaptiques ne suit pas rigoureusement le schéma attendu : le neurone C est légèrement en retard. La réponse du

neurone D est tout de même un potentiel d'action, mais anormalement tardif. Nous nous trouvons à la limite de déclenchement de la fonction réalisée. Ce phénomène très intéressant nous montre à quel point nous sommes éloignés de l'électronique numérique ou des réseaux de neurones formels, et ce, malgré de grandes similitudes apparentes. Bien que les phénomènes obéissent à la règle du tout ou rien, il n'est pas possible de dessiner des frontières claires entre les cas positif ou négatif. Selon les niveaux de bruit analogique inévitable, une séquence rigoureusement identique pourrait n'avoir produit aucun résultat. Il n'est, hélas, pas possible de montrer un tel exemple puisque l'acquisition est déclenchée par l'activité du neurone D.

2.4.4 Utilisation dans le cadre de simulations plastiques

Nous allons désormais nous attarder sur des réseaux simulés dans des configurations biologiquement réalistes, mettant en œuvre du bruit synaptique et des connexions soumises à l'algorithme de la STDP.

Le bruit synaptique

Dans un réseau de neurones biologiques, les synapses sont sensibles à toutes sortes de phénomènes. Il leur arrive par conséquent de provoquer une stimulation mineure dont l'origine n'est pas l'activité d'un neurone présynaptique. De telles stimulations amènent des perturbations dont les conséquences sont variables. Elles sont appelées *bruit synaptique* ou *bruit d'activité environnante*.

Dans les simulations sur un réseau de taille conséquente, il est créé par un ensemble de connexions faibles et non plastiques. Ces connexions sont choisies entre des neurones ayant une très faible corrélation. Mais lorsque le réseau simulé est trop réduit pour trouver ce genre de connexions, il est nécessaire d'introduire ce bruit différemment.

La simulation de cette activité s'effectue alors par l'envoi de faibles stimulations sur chaque synapse. Elle est d'origine aléatoire ou pseudo-aléatoire, mais obéit à des lois stochastiques (loi de Poisson... [DES99b]). Selon les besoins de la simulation, il peut également être nécessaire de corréler les séquences envoyées sur certaines synapses.

La génération de l'ensemble des séquences de bruit pour un réseau est donc relativement lourde en puissance de calcul. C'est pourquoi elles sont précalculées pour ne pas accaparer de temps de calcul lors de la simulation.

Validation algorithmique

Le premier réseau est présenté figure 2.18, il utilise deux neurones excitateurs qui se stimulent entre eux. Chacun est soumis à l'influence de bruit synaptique. Les deux connexions sont plastiques et obéissent à l'algorithme de STDP [BAD06]. L'activité est générée par le bruit synaptique.

Ce réseau, bien que réaliste, n'a pas de fonction biologique particulière. Il sert essentiellement à valider le principe de calcul de plasticité. Cette simulation a été réalisée par Quan Zou de l'*Unité de Neurosciences Intégratives et Computationnelles* (UNIC, CNRS Gif sur Yvette) [ZOU06a] et ces travaux ont été l'objet de différentes communications [ZOU06b]. Le poids de départ des connexions est quelconque, mais

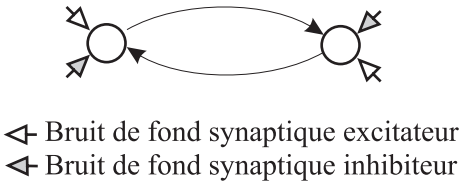

FIG. 2.18 – Réseau mettant en œuvre le calcul de plasticité

toujours trop faible pour engendrer à lui seul un potentiel d'action. Le comportement du réseau a été simulé pour 6 valeurs initiales différentes.

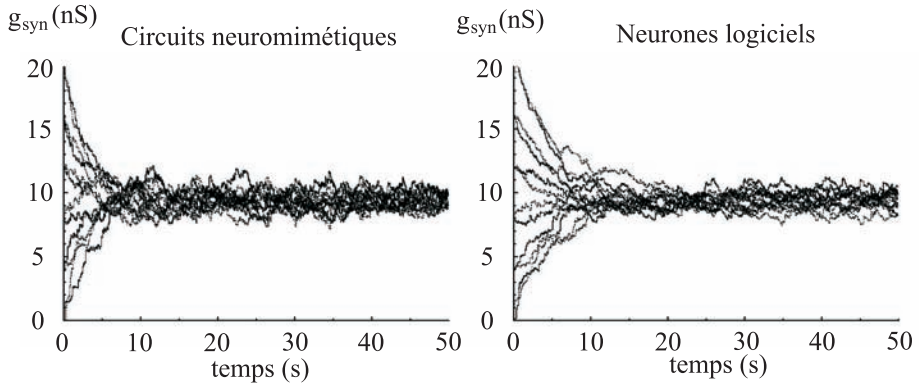

FIG. 2.19 – Évolution du poids des connections plastiques (excitatrices) en fonction du temps

L'effet attendu est un équilibrage des poids synaptiques pour rétablir la symétrie du réseau. La même simulation a été réalisée à l'aide du simulateur logiciel NEURON. Le tracé comparatif des poids synaptiques en fonction du temps pour les simulations logicielles et utilisant PAX est donné par la figure 2.19.

L'observation de ces courbes nous montre qu'effectivement, le réseau s'équilibre. La comparaison avec le résultat de simulation logicielle permet également de montrer que les fluctuations obtenues ont une amplitude conforme au modèle. Les poids synaptiques finaux sont égaux pour les simulations logicielle et matérielle. La seule différence observable se situe au niveau du temps de convergence des poids synaptiques.

Usage du système au complet

Une simulation plus étendue a été réalisée avec un réseau de 8 neurones. Ce réseau est présenté figure 2.20. Il a recours à des neurones inhibiteurs pour stabiliser l'activité. Les connexions représentées sont toutes à double sens : inhibitrices des neurones inhibiteurs vers les excitateurs, et excitatrices plastiques des neurones excitateurs vers les neurones inhibiteurs.

Cette simulation a également été menée sur les deux simulateurs analogique et logiciel. Les résultats sont équivalents à ceux de la figure 2.19. Ils montrent bien une convergence des poids synaptiques dans les deux cas. Cependant, pour les poids les plus élevés, un léger décalage apparaît.

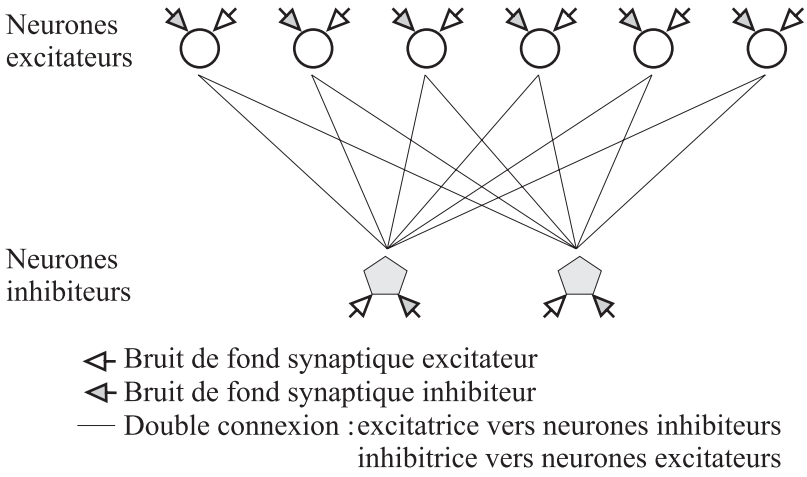

FIG. 2.20 – Réseau de plasticité avec neurones inhibiteurs.

Ce phénomène, qui n'intervient que lorsque les synapses sont fortement sollicitées, pourrait être associé à une saturation dans le circuit de simulation des synapses. Dans ce cas, le courant réellement injecté par la synapse est inférieur à ce que demande le logiciel. Étant donné que la plasticité fonctionne comme une contre-réaction contrôlée par l'activité du réseau, la valeur finale se stabilise de telle sorte que ce soit le courant réellement injecté par la synapse qui soit ajusté. Le poids demandé par le logiciel est donc naturellement supérieur.

Le réseau peut donc fonctionner correctement grâce à l'action de la plasticité qui corrige les poids réels ; l'activité des neurones et du réseau n'est donc pas perturbée. Les valeurs des connexions doivent cependant être corrigées si leur exploitation en externe est prévue.

2.5 Analyse

La description de l'*ordinateur synaptique*, machine qui gère les synapses, mais aussi vision du réseau de neurones comme un calculateur fonctionnant *par* les synapses, est désormais achevée. Revenons d'abord aux points essentiels de ce système, pour pouvoir mener une analyse critique, nécessaire sur cette première version de simulateur mixte de réseaux de neurones.

2.5.1 Points essentiels

Pour simuler les neurones, le système réalisé utilise, comme élément de base, des circuits intégrés analogiques. Ces circuits simulent un nombre réduit de modèles figés basés sur le formalisme d'Hodgkin et Huxley. La commande du circuit, et des convertisseurs de signaux numérique vers analogique, est entièrement réalisée par un FPGA. Ce dernier est contrôlé *via* le bus PCI par un ordinateur fonctionnant avec un système GNU/Linux.

Le processeur peut ainsi accéder directement à chaque circuit. Il simule le comportement du réseau en lisant l'état des neurones, puis en envoyant les stimulations correspondantes sur les entrées synaptiques appropriées. Le contrôle de la simula-

tion du réseau est donc pris en charge par le processeur qui scrute les neurones en permanence à la fréquence de 10kHz.

Des simulations ont été effectuées pour valider le principe. Elles montrent que les approximations effectuées (multisynapses, incertitudes sur la date précise des potentiels d'action..) n'ont pas d'influence observable.

2.5.2 Conclusion

Ce système prouve la faisabilité et la pertinence du principe d'un simulateur mixte. Les simulations effectuées montrent que le système réalisé ne souffre pas de la comparaison avec un simulateur numérique en termes de résultats, tout en étant supérieur en équivalence de capacité de calcul.

Il est cependant nécessaire de concevoir une structure plus adaptée pour des systèmes capables de gérer un nombre de neurones plus important. Le principe de scrutation impose que tout calcul de plasticité s'effectue dans un temps limité à 100µs sur l'ensemble des connexions impliquant un neurone donné. Associée à un processeur traditionnel, même puissant, cette structure ne permet pas d'envisager de réseau d'une taille supérieure à 20 neurones.

Passée l'étape de validation de principe, il nous reste donc à créer une architecture d'un niveau supérieur, permettant la gestion d'un nombre significatif de neurones. Une meilleure gestion de la puissance de calcul disponible en est un élément indispensable.

Chapitre 3

Vers le réseau de neurones matériel

Maintenant que PAX a permis de valider le principe du réseau de neurones analogiques géré par un système numérique, il reste à produire un système capable d'évoluer pour gérer des réseaux de plus grande taille. De la capacité à gérer de tels réseaux dépend la mise en œuvre de simulations plus proches du biologique. L'objectif, par rapport au système de première génération, est de gagner un ordre de grandeur sur le nombre de neurones.

Nous commencerons d'abord par effectuer un bilan des contraintes que nous apporte l'agrandissement du réseau, confrontées avec l'expérience de l'usage de PAX. Nous verrons ensuite la conception de ce nouveau système du point de vue matériel, et l'évaluation des capacités que donne cette nouvelle structure. Nous pourrons alors observer une nouvelle série de résultats, ainsi qu'effectuer un nouveau retour d'expérience.

3.1 Évolutions requises par la croissance du réseau

Le système PAX n'était destiné qu'à une validation de principe et, à ce titre, n'a pas bénéficié d'une architecture permettant l'usage d'un grand nombre de neurones. Il a néanmoins permis de localiser les points cruciaux pour la conception d'un système de simulation plus ambitieux. La conduite de simulations, dans des conditions d'étude pour les neurosciences, apporte également une vision pragmatique de l'usage du système. Il est alors possible de l'optimiser pour cet usage.

3.1.1 Faire appel à un spécialiste

Maintenir la polyvalence d'une architecture, lorsqu'elle est étendue à plus grande échelle, est la tâche qui monopolise le plus d'efforts. Avant de construire les bases de la prochaine génération, il est donc crucial de connaître les fonctions dont l'abandon simplifiera l'augmentation du nombre de neurones.

Le premier point fait référence au bruit de fond synaptique. En effet, dans les simulations qui le mettent en œuvre, la génération de ce bruit utilise une base entièrement pseudo-aléatoire décorrélée de l'activité des neurones. Pour alléger les besoins de calculs, les séquences de bruit sont précalculées puis envoyées au fur et à mesure sur les synapses. Il est donc envisageable de prévoir le nouveau système de telle sorte que l'envoi du bruit soit optimisé, voire que ce dernier soit directement mémorisé localement, sans solliciter le processeur central.

La liberté de déterminer un temps de propagation s'est également révélée peu intéressante. De par l'architecture de PAX, le temps de propagation d'une information était compris entre 100 µs et 200 µs. Ce temps pouvait en outre être allongé par quanta de 100 µs. Or, les réseaux simulés étant de nature locale, ces temps de propagation sont considérés nuls dans les modèles. Plutôt que de chercher à les étendre par grandes valeurs, il semblerait donc qu'il soit préférable de les raccourcir le plus possible.

Par rapport au modèle utilisé, le choix se porte sur une sommation linéaire des stimulations synaptiques. Cette absence de phénomène de second ordre permet de nombreuses optimisations, tant matérielles que logicielles.

Il convient d'ajouter à tout cela que les possibilités de dialogue avec le bus PCI ont été prévues bien plus ouvertes que nécessaire. En effet, l'ensemble des accès matériels ont été réalisés par la bibliothèque de fonctions fournie. L'auteur seul ayant écrit les fonctions accédant à la carte, il est possible de supprimer les possibilités censées faciliter les méthodes d'accès. L'interface sera alors limitée aux accès 32 bits, et l'inutile relecture des paramètres supprimée.

3.1.2 Répartition des fonctions de calcul

Au niveau matériel, il ne sera plus possible d'envisager une interface directe de l'ensemble des circuits neuromimétiques sur un seul FPGA. Il convient donc de réaliser une structure numérique capable de contrôler l'ensemble des neurones.

Une telle architecture a déjà été proposée sur d'autres systèmes de simulation de réseaux de neurones, il s'agit de l'AER (Adress-Event Representation) [BOA00]. Décrit rapidement dans l'état de l'art (page 34), l'AER n'est par contre que très peu adaptée à un circuit contenant moins de 16 neurones. Pour le mettre en œuvre, il faudrait grouper plusieurs circuits entre eux, et leur fournir une interface à l'aide d'un composant supplémentaire. Ceci dit, grouper plusieurs circuits dans une même interface requiert l'usage d'un composant numérique externe. L'usage de FPGA s'impose alors. Compte tenu des possibilités qu'un tel composant apporte en terme d'interfaces de communication, nous pouvons aisément choisir un structure plus élaborée qui permet, entre autres, d'associer à chaque évènement l'instant auquel il a été produit. L'architecture pressentie est présentée figure 3.1. Nous retrouvons de la capacité de calcul à chaque niveau hiérarchique.

L'interface entre le système et la machine reste le bus PCI. Sa mise en œuvre n'a pas révélé de limite particulière. Les débits nécessaires estimés sont encore très en deçà de la bande passante disponible. L'interface destinée à la structure reliant les FPGA au sein du système reste par contre à définir. Cependant, elle dépend essentiellement du nombre de circuits à interconnecter. Cette donnée ne sera disponible qu'une fois la nouvelle génération de circuits neuromimétiques fabriquée et testée.

3.1.3 Exigences de la fonctionnalité de connectivité

Les deux parties précédentes laissent apparaître une meilleure répartition des tâches, ainsi que certains allègements algorithmiques au prix d'une spécialisation du système. Toutefois, malgré les allègements prévus ci-dessus, la puissance d'un ordinateur n'est toujours pas suffisante pour gérer un réseau de plusieurs dizaines de neurones. Au lieu d'utiliser un plus petit FPGA pour l'interface vers l'ordinateur, il est préférable d'intégrer la fonction la plus gourmande dont le processeur avait la charge. Il s'agit de la scrutation logicielle de l'état des neurones. Pour un concepteur

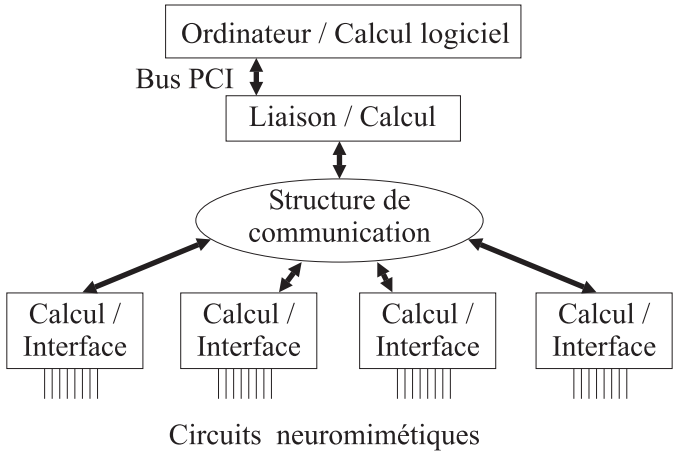

FIG. 3.1 – PAX2 : un calcul plus réparti pour plus de capacité de traitement

de systèmes, le simple mot *scrutation* est synonyme de gaspillage. Nous ne pouvions cependant pas nous permettre d'éviter ce principe sans avoir éprouvé les méthodes d'accès au réseau. L'expérience acquise avec PAX nous y autorise désormais.

La contrainte d'offrir la possibilité de connecter tout neurone à chaque autre neurone du réseau est toujours présente. Or, la complexité algorithmique de la gestion des connexions du réseau est proportionnelle au nombre de connexions possibles. Elle évolue donc par rapport au carré du nombre de neurones gérés. La puissance disponible pour la plasticité serait alors moindre que sur le système précédent, elle ne permettait cependant pas d'envisager plus de 15 ou 20 neurones.

Pour terminer, la limite du calcul en 100 µs était due à la scrutation. En réalisant les interruptions du système par rapport au déclenchement de potentiels d'action, plutôt qu'en se synchronisant sur une horloge fixe, nous laissons à l'utilisateur le choix entre un grand réseau ou un modèle de plasticité plus complexe.

Un tel système engendre cependant deux limites :

- sans repère temporel fixe, l'envoi de bruit synaptique prééchantillonné devient problématique ;
- le système logiciel n'a plus les moyens de gérer les potentiels d'action simultanés de façon simple en mode utilisateur.

En ce qui concerne le bruit synaptique, il est parfaitement possible de le prendre en charge au niveau matériel. Pour assurer une haute priorité à la connectivité, la solution est moins évidente. Plusieurs solutions sont envisageables :

- il est possible, comme mentionné dans le chapitre 2, de générer deux processus, l'un prioritaire pour les interruptions et la connectivité, et l'autre plus lent pour la plasticité ;
- il serait techniquement plus simple de calculer la connectivité du réseau au niveau du gestionnaire d'interruption, dans l'espace noyau, mais cette méthode est bien moins *propre* ;
- la dernière solution est une intégration matérielle de cette fonction, au niveau du FPGA.

La solution retenue est l'implantation matérielle du bruit, tout en utilisant les capacités de l'ordinateur pour mémoriser les séquences prédéfinies sur une longue durée.

Il est possible de mémoriser les poids synaptiques dans une RAM interne. Pour chaque potentiel d'action, un bloc se charge d'envoyer la stimulation correspondante si besoin est. Outre l'allègement calculatoire du processeur central, cette méthode offre l'avantage d'être très rapide du point de vue du réseau puisqu'un cycle d'horloge suffit à établir une connection. Par exemple, pour un réseau de 100 neurones et un FPGA cadencé à 50 MHz, l'ensemble des stimulations synaptiques consécutives à un potentiel d'action est établi en environ $2\,\mu s$[1].

3.1.4 Maîtrise de l'aspect temporel

Finalement, l'augmentation du nombre de neurones modifie en profondeur la circulation de l'information et son traitement au sein du système. Nous venons de voir une technique capable de réduire fortement la communication au sein des neurones, alors qu'un nombre de connections en évolution parabolique annonce un processeur central surchargé pour le calcul de plasticité. Il n'est donc plus possible de considérer que l'instant de traitement est celui de l'évènement (le temps réel est cependant conservé puisqu'il est possible de garantir que la nouvelle connection plastique sera établie avant d'être sollicitée).

L'usage d'une référence temporelle fiable devient essentielle pour éviter l'influence de la discontinuité du modèle de STDP. Une optimisation de la circulation des données, associée à une évaluation précise, en est un aspect. Pour garantir une précision optimale, chaque évènement doit être accompagné d'une référence temporelle. Cette dernière est à attribuer dès la première étape. Il est alors possible de compenser des délais variables susceptibles de survenir au cours de la progression de chaque évènement jusqu'à l'ordinateur de simulation

3.2 Le système PAX2

Nous abordons désormais la partie traitant de la réalisation physique du simulateur de deuxième génération. Nous verrons d'abord le circuit utilisé, l'architecture générale, puis la gestion de la communication et finalement la répartition des tâches au sein de ce nouveau système.

3.2.1 Le circuit *Claudia*

Conçu spécifiquement pour ce système, le circuit *Claudia* répond aux principales contraintes rencontrées dans ce cadre : l'intégration et l'interfaçage. Ce circuit de $19\,mm^2$ ($4,4\,mm \times 4,4\,mm$) a été réalisé en technologie BiCMOS $0,8\,\mu m$ de austria*micro*system. Comme *Trieste*, il a été routé automatiquement à l'échelle des blocs de calcul.

Vers une plus grande densité

La principale caractéristique de ce circuit est l'intégration de quatre nœuds de calcul. Il est donc capable de simuler le comportement de quatre neurones analogiques simultanément et en temps réel. Pour la réalisation, ce circuit se base sur la même bibliothèque de fonctions mathématiques que le circuit *Trieste*. Il se situe donc dans

[1] environ car nous ne comptabilisons ici que les 100 accès mémoire, il convient d'ajouter l'amorçage de la chaîne de traitement

la même philosophie de conception, à savoir : une implantation de paramètres figés et un nombre minimal de composants externes.

Une interface minimalisée

Le caractère figé observable dans le circuit *Trieste* a ici été renforcé puisque les noyaux de calcul sont spécialisés. Le circuit peut simuler trois neurones excitateurs (RS) et un neurone inhibiteur (FS). Cette proportion correspond aux statistiques observées en biologie. Le seul paramètre réglable est la force du canal modulant pour le modèle excitateur. D'un jeu de quatre valeurs, il passe à huit valeurs possibles. Cette valeur est commune à l'ensemble des neurones du circuit.

Alors que huit signaux numériques par neurone étaient nécessaires pour *Trieste*, quatre signaux le sont en moyenne pour les neurones excitateurs, et trois pour les neurones inhibiteurs.

Une mise en œuvre délicate

Bien que la conception du circuit *Claudia* se soit effectuée en reprenant des éléments de bibliothèque éprouvés, son usage s'est révélé plus problématique que prévu. Aucun circuit ne présente l'activité d'un potentiel de membrane. L'environnement de fonctionnement du circuit a été modifié pour imposer un fort courant de compensation sur les composants externes. Seul 21 noyaux de calcul sur 100 ont alors fourni un résultat exploitable.

Cependant, ce dysfonctionnement ne semble pas lié à un défaut de conception ou de *design*. En effet, quelle que soit la fonction choisie, il existe au moins trois circuits au sein desquels elle est opérationnelle. L'observation microphotograhique a révélé une fabrication peu soignée, mais rien qui ne soit susceptible d'altérer le fonctionnement du circuit. Il semble que nous ayons ici une illustration des limites de l'architecture de circuits à paramètres figés : si les processus amènent un résultat trop éloigné du cas typique, les dérives des différents composants se combinent, et les paramètres ne sont plus significatifs.

Toujours est-il que le nombre de neurones réellement fonctionnels est en deça des prévisions pour la réalisation du système de seconde génération. La décision fut alors prise de ne pas concevoir un système complet, mais d'ajouter les neurones fonctionnels sur PAX au travers des connecteurs de tests disponibles sur la carte PCI. Ce système sera néanmoins qualifié de seconde génération étant donnés les changements structurels considérables qu'il aura subi, il sera désormais dénommé PAX2

3.2.2 Un système éclaté

De par le nombre de composants mis en jeu, il était évident que PAX2 serait constitué de plusieurs cartes interconnectées. Le fait de connecter ce système au moyen de la carte PCI développée pour PAX ajoute une difficulté supplémentaire due à l'hétérogénéité du système, elle est cependant acceptable compte tenu du fait qu'il reste. possible de continuer à utiliser les huit circuits *Trieste*.

La carte PCI possède trois connecteurs de vingt signaux numériques chacun, ce qui permet une interface à haut débit vers le système. Les possibilités du FPGA permettent un transfert synchrone à 32 MHz pour des mots de grande taille. Le choix s'est cependant porté sur un système asynchrone plus modeste en terme de débits,

mais capable de s'adapter quelles que soient les fréquences de synchronisation de chaque côté. Cela permet une plus grande marge de manœuvre pour la réalisation de l'extension.

Réalisation de l'extension

Comme décrit précédemment, les circuits sont interfacés directement à un FPGA. Ces derniers sont associés au sein d'une même structure de communication pour assurer le fonctionnement du réseau. Le FPGA gérant la carte PCI y est assimilé étant donné qu'il est, lui aussi, en contact direct avec des circuits neuromimétiques.

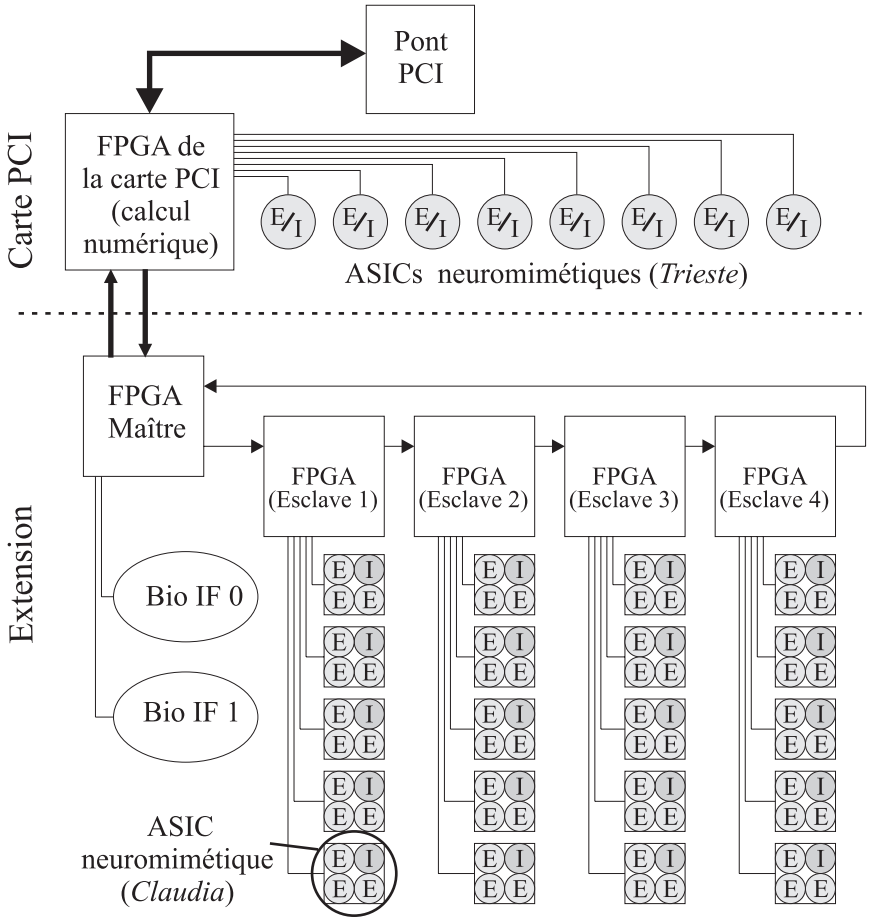

FIG. 3.2 – Description de la structure du système PAX2

Le schéma de la figure 3.2 présente la structure du système PAX2. Nous pouvons voir qu'il contient six FPGA :

- Quatre FPGAs dits *esclaves*, chargés de l'interface vers les circuits *Claudia*. Ils communiquent de façon chaînée par un protocole en série.
- Un FPGA dit *maître*. Ce nom ne signifie pas qu'il contrôle l'ensemble du système. Il s'occupe essentiellement de gérer la chaîne série et le relai des informations

depuis et vers la carte PCI. Les fonctionnalités *Bio-IF* seront commentées plus loin.

- Le dernier FPGA dit PCI est celui dans lequel était configuré le système PAX. Il s'agira désormais du noyau de calcul gérant l'ensemble du réseau. Il est relié directement aux huit circuits *Trieste*, au pont PCI et au FPGA *maître*.

L'ensemble de l'extension se répartit sur trois cartes dont deux sont identiques. L'une, unique, se charge de la gestion de l'ensemble de l'extension. Elle héberge donc la connection vers la carte PCI, le FPGA maître, l'oscillateur commun au système ainsi que les alimentations. Son nom de référence est *Blanche*, en référence à la matière blanche du cerveau qui rassemble les prolongements axonaux, et permet ainsi la connection des neurones.

L'autre carte, présente en deux exemplaires, héberge les neurones ainsi que les FPGAs esclaves associés. Chaque exemplaire contient deux FPGA et dix circuits *Claudia*. Son nom de fabrication est *Grise*, par rapport à la matière grise du cerveau où se situent les neurones. Les différentes cartes sont présentées figure 3.3 avec une photographie de l'extension complète montée.

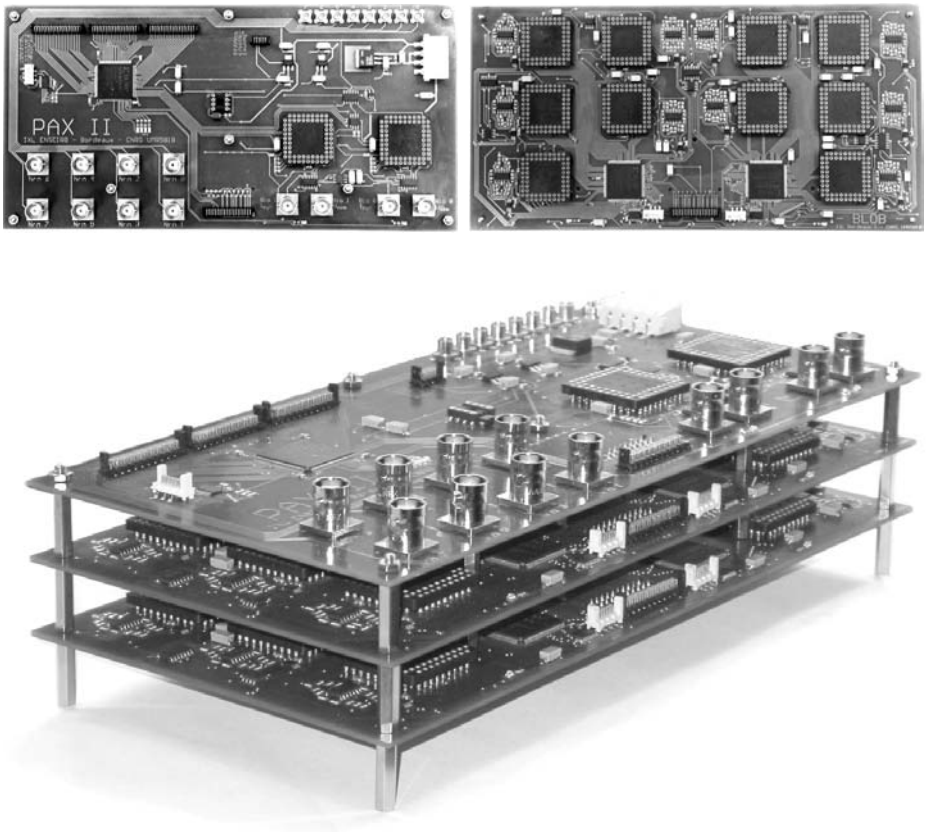

FIG. 3.3 – Les différentes cartes du système PAX2 : *Blanche* (en haut à gauche), *Grise* (en haut à droite) et le système complet monté. Il prend place au fond du boitier de l'ordinateur d'accueil, sous la carte PCI.

Les différents signaux et alimentations nécessaires aux cartes *Grise* sont fourni par *Blanche* au moyen d'un connecteur central. Chaque carte possède un connecteur d'entrée et de sortie des signaux. En fin de boucle, il est nécessaire de renvoyer les

signaux vers *Blanche* pour fermer la chaîne de communication. Cette extension a été conçue pour être alimentée par l'ordinateur comme un périphérique quelconque. Sa fréquence de fonctionnement, prévue à 32 MHz, peut être élevée à 40 MHz. Chaque FPGA de l'extension n'effectue que de la traduction d'interface vers les ASICs neuromimétiques et la communication interne qui s'y rapporte.

Vers les réseaux hybrides

La carte *Blanche* accueille une dernière fonctionnalité expérimentalement très intéressante. Il s'agit des deux éléments référencés *Bio-IF* sur le schéma fonctionnel (figure 3.2), et qui permettent de connecter des neurones biologiques au réseau de PAX2. Pour ce faire, ils utilisent les synapses de circuits *Trieste*, la simulation du soma étant désactivée physiquement. Un ensemble d'amplificateurs permet de les connecter aux instruments de mesure et de stimulation des neurones *in vitro* comme lors des expériences de réseaux de neurones hybrides déjà réalisées au sein de l'équipe [LEM02]

Grâce aux interfaces numériques des circuits contenant les synapses, ces neurones biologiques sont accessibles depuis le système par les mêmes méthodes que les neurones artificiels.

3.2.3 La communication

Nous allons désormais nous attarder sur les capacités de communication des différentes structures de la carte. L'ensemble des calculs a été effectué par rapport à 32 neurones : 8 simulés par les circuits *Trieste* sur la carte PCI et 24 par les circuits *Claudia* sur l'extension. Les chiffres fournis sont donc surestimés puisque seuls 21 noyaux de calcul seront exploitables sur l'extension.

Les communications peuvent prendre trois formes au sein du système.

- Les données de configuration : ce sont les moins pénalisantes, elles ne circulent pas pendant la simulation, et ne nécessitent donc pas de gestion en temps réel (il est simplement important de garantir leur arrivée).
- La remontée d'informations concernant chaque potentiel d'action : le numéro de neurone ainsi qu'une référence temporelle précise doivent être envoyés vers le FPGA PCI dans les conditions de temps réel.
- L'envoi des stimulations synaptiques : l'ensemble des stimulations synaptiques calculées par le FPGA PCI doivent être acheminées au neurone concerné en temps réel. Ce sont les données les plus exigeantes car les plus nombreuses. Elles sont générées par paquets puisque plusieurs stimulations résultent d'un seul potentiel d'action. Il faut également leur ajouter celles qui sont issues du bruit de fond synaptique.

Le dernier point à souligner est la présence d'un signal électrique dédié à la remise à zéro du système et à la synchronisation des registres d'horloge sur les deux systèmes.

La communication vers les circuits de la carte PCI ne pose pas de problème particulier puisqu'ils sont accessibles directement. Nous allons par contre nous attarder sur les méthodes mises en œuvre pour véhiculer les informations sur l'extension.

Les pires cas

Nous allons définir les pires cas de fonctionnement, c'est à dire, l'ensemble des conditions qui, réunies, sollicitent le plus fortement le système. Il y en a deux : le

premier génère une communication très intense de façon ponctuelle et engorge les structures de communication, le second met le système à l'épreuve sur une plus grande période et exploite essentiellement les capacités de calcul.

Pour tester la fluidité de la communcation au sein du système, nous considérons un réseau de 32 neurones, chacun étant connecté aux 31 autres. Ce cas d'école peu réaliste nous permet de chiffrer la caractérisation du système. Les neurones sont supposés générer un potentiel d'action au même instant précisément. Pour bien s'assurer que la situation soit la plus contraignante, une stimulation de bruit synaptique est supposée être envoyée à chaque synapse. Considérant 24 neurones sur l'extension, ceci aboutit à 48 stimulations de bruit, 24×23 stimulations générées par les neurones de l'extension, et 8×24 stimulations provenant de l'activité des circuits *Trieste*. Le total est de 792 stimulations.

Pour vérifier la tenue du système en terme de puissance de calcul, nous considérons toujours un réseau de 32 neurones, chacun connecté à tous les autres. Cette fois, les neurones ne présentent pas de synchronisation particulière, cependant, ils sont considérés avoir une activité très rapide (150 Hz). Cette fréquence est trop élevée pour être rencontrée sur un réseau biologique entier. Nous considérons également que chaque synapse reçoit des stimulations de bruit de fréquence moyenne 100 Hz. Nous obtenons alors, pour chaque seconde, $(24 \times 23 + 8 \times 24) \times 150$ stimulations pour l'activité du réseau, et 48×100 stimulations pour le bruit synaptique. Le total est de 116 400 stimulations par seconde.

Ces pires cas seront étudiés qualitativement car ils ne sont pas probables en simulation. La définition d'un pire cas probable est néanmoins très délicate, nous en tenterons une approche vers la fin de ce chapitre.

Les deux configurations présentées ne sont pas compatibles entre elles. En effet, si les neurones sont actifs simultanément, un engorgement et des délais apparaissent. Par contre, la longue inactivité qui s'ensuit permet au système de traiter des tâches de fond avec une meilleure efficacité. La montée en charge des calculs liés au réseau doit donc s'effectuer par une succession d'activités morcelées.

La chaîne bouclée de transmission (extension)

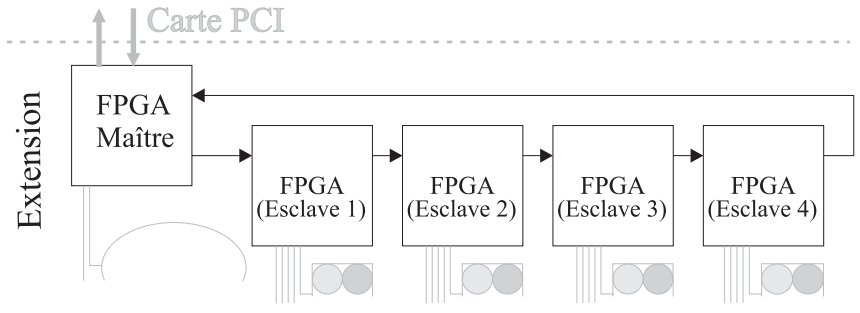

FIG. 3.4 – La chaîne bouclée de communication au sein de PAX2

La transmission d'informations au sein de l'extension (Fig. 3.4) utilise un protocole série synchrone avec un signal de données et un signal de début de mot. Elle est unidirectionnelle. Les données sont codées sur des mots de 16 bits. Leur codage est donné en annexe C. L'envoi des mots est synchronisé sur l'ensemble du système,

ce qui signifie qu'en un instant donné, tous les FPGAs émettent et reçoivent un mot. Le FPGA maître envoie à l'ensemble du système l'horloge et le signal de début de transaction.

À la fin de chaque mot, chaque FPGA détermine s'il est, ou non, le destinataire. Dans le cas positif, l'instruction est executée et rien n'est envoyé envoyé au FPGA suivant. Dans le cas contraire, l'instruction est envoyée au suivant de la chaîne. Lorsqu'une information doit être transmise d'un esclave vers le maître, il suffit d'attendre une situation dans laquelle un mot vide aurait été transmis (mot vide reçu ou information destinée à un circuit local). Le mot à expédier comblera alors l'espace vide.

Le FPGA situé en bout de chaîne dépend, pour retourner des données, des mots vides que lui envoient les autres. Il pourrait alors passer pour moins prioritaire, risquant d'être réduit au silence. Cependant, il y a plus de mots qui descendent vers les esclaves qu'il n'en remonte. Cette proportion garanti que chaque composant de la chaîne sera toujours en mesure de renvoyer ses données.

En ce qui concerne les performances de la communication, la mise en chaîne des composants par un protocole série apparaît comme un goulet d'étranglement. Voyons comment il se comporte dans les pires cas. Chaque mot étant codé sur 16 bits, 16 cycles sont nécessaires à leur transmission. Les derniers bits sont destinés à contenir des paramètres. Le traitement du mot peut donc être commencé avant sa réception totale. Il n'est donc pas nécessaire de perdre de cycle de traitement entre deux envois de mots. Pour un fonctionnement à 40 MHz, il est donc possible d'envoyer 2,5 mots par microseconde. Dans le pire cas avec des neurones synchrones, les 792 stimulations synaptiques nécessiteront $316,8\,\mu s$ pour être acheminées à travers le système.

Si plus de $300\,\mu s$ sont rédhibitoires pour une réponse temporelle correcte, il est possible d'envisager une communication utilisant les deux fronts de chaque période d'horloge. Les FPGAs ne peuvent alors tolérer qu'une fréquence de 32 MHz, mais il ne faut plus que 8 cycles pour transmettre chaque mot. Il est alors possible de transférer 4 mots par microseconde. Le pire cas descend alors à un temps de propagation maximal de $198\,\mu s$, ce qui est équivalent aux possibilités de PAX.

La connexion asynchrone (extension – carte PCI)

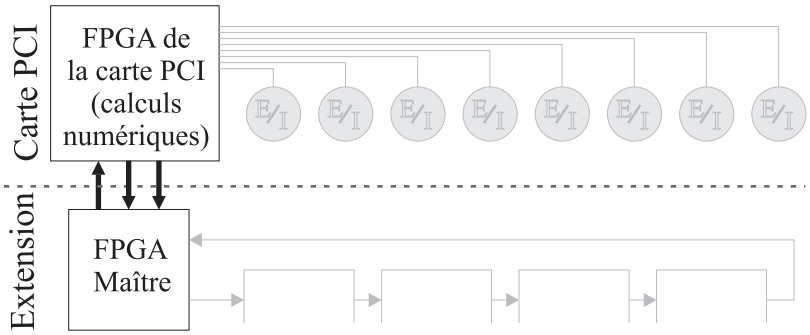

FIG. 3.5 – Les liaisons asynchrones entre la carte PCI et l'extension

Les connexions asynchrones utilisées entre la carte PCI et l'extension (représentées figure 3.5) sont unidirectionnelles et nécessitent chacune deux signaux pour le

protocole. Les données sont transférées en parallèle. Il y a trois connexions mises en œuvre, chacune pour un type de données précis :

- les mots de configuration : de la carte PCI vers l'extension
- les stimulations synaptiques : de la carte PCI vers l'extension
- les évènements de potentiel d'action : de l'extension vers la carte PCI

Les deux signaux de contrôle sont *données-valides* et *confirmation*. Un diagramme représente une transaction en figure 3.6.

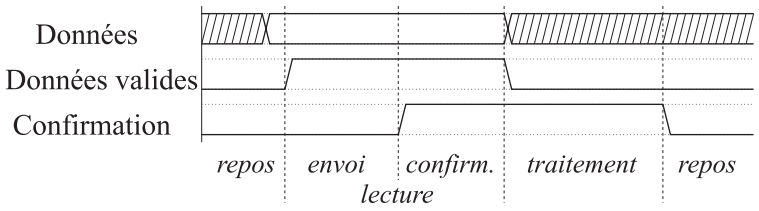

FIG. 3.6 – Déroulement temporel d'un transfert asynchrone

Pour effectuer une transmission, l'émetteur attend que le récepteur ait remis à zéro le signal de confirmation. Il envoie alors les données et lève au niveau logique 1 le signal *données-valides*. Dès détection de l'envoi, le récepteur confirme par un niveau 1 sur la ligne de confirmation. L'émetteur peut alors remettre à zéro la validation, le transfert est effectué. Aucune nouvelle transaction ne sera initiée tant que le récepteur n'aura pas remis à zéro la de confirmation.

Lorsqu'il est saturé, le récepteur peut mettre en attente l'émetteur. Soit il ne libère pas le canal (confirmation = 1, même après le transfert), soit il attend le temps nécessaire pour confirmer l'arrivée d'un nouveau mot.

Ce principe de communication fonctionne en quatre étapes. Le coût temporel de ce protocole peut donc être majoré par 4 périodes d'horloge du composant le plus lent. Soit, pour une extension à 32 MHz, 8 mots par microseconde. Dans tous les cas, ce canal est donc plus performant que la chaîne série. Il offre aussi l'avantage d'autoriser le FPGA maître à forcer une attente du FPGA PCI.

Considérations globales

Outre l'ensemble des délais dûs à la saturation des connexions, il est également nécessaire de s'intéresser au temps de transit, ainsi qu'au temps de calcul. Suivons pour cela le parcours équivalent à une transmission synaptique d'un potentiel d'action. Ce temps de parcours sera alors à ajouter au temps de congestion observable dans les pires cas. Il est maximal lorsqu'un neurone du premier esclave est connecté à un neurone du dernier. Il faut alors (les temps sont majorés, une illustration du parcours est disponible en figure 3.7) :

- parcourir la chaîne ($2\,\mu s$) ;
- envoyer l'évènement sur la carte PCI ($0,125\,\mu s$) ;
- traiter l'évènement : ($0,5\,\mu s$: un cycle à 64 MHzpour chaque connexion vers un autre neurone) ;
- envoyer la stimulation vers l'extension ($0,125\,\mu s$) ;
- parcourir la chaîne jusqu'au dernier esclave ($2\,\mu s$).

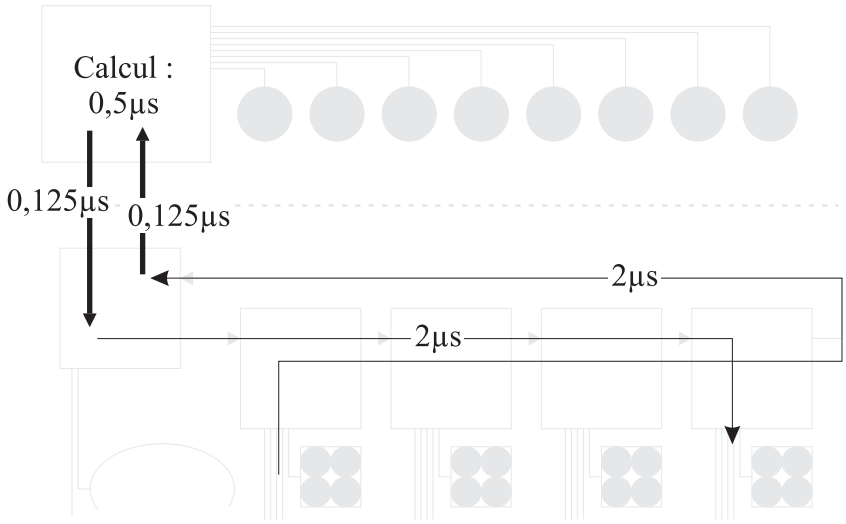

FIG. 3.7 – Illustration du parcours de l'information

Le total approche donc 5 µs. Nous n'avons pas tenu compte du délai éventuel pour l'attente d'un mot vide qui permet de faire remonter l'information du potentiel d'action. Dans le pire cas où les neurones sont synchronisés, la chaîne est libre pour les premiers envois. Arrive ensuite une situation où l'arrivée des stimulations synaptiques occupe la chaîne et ralentit la remontée des évènements pour les neurones moins prioritaires. Cependant, chaque FPGA recevant une donnée a la possibilité d'en émettre une autre. La latence est alors créée par l'envoi des stimulations synaptiques dans la chaîne. Dans le pire cas où les neurones sont synchronisés, la propagation d'un évènement sera donc de 321 µs au plus (203 µs en utilisant deux fronts par cycle).

Intéressons-nous désormais au temps de propagation dans le pire cas relatif au fonctionnement global. Bien que l'activité des neurones soit corrélée du fait des connexions, l'activité consécutive à une stimulation ne peut intervenir avant l'arrivée de cette dernière. De plus, l'activité résultante n'est pas instantanée et temporellement entachée d'erreur. Il est alors possible de considérer qu'à l'échelle temporelle du système (de 10 à 100 µs), l'occurence des évènements est aléatoire. Nous pouvons ainsi déterminer les temps de propagation maximaux en fonction du nombre de neurones actifs au même instant et les associer à une probabilité.

Lorsqu'un neurone emet une activité, le système a besoin de 18 µs pour réagir. Si plusieurs neurones émettent un potentiel d'action à moins de 18 µs d'intervalle, les informations vont se gêner, et le temps de réaction (ou de propagation) sera plus long. Ce phénomène sera appelé *collision*. Le tableau 3.1 nous représente des données concernant les collisions, en fonction du nombre de neurones impliqués. Nous pouvons voir le temps de propagation maximal qui en découle ainsi qu'une évaluation de leur taux d'apparition sur une période à 150 Hz. Ce taux de présence est ensuite ramené à une estimation d'apparition d'une collision sur une simulation de cinq minutes.

Ce tableau a été établi suite à 80 000 séries de 2 000 000 tirages pseudo-aléatoires concernant la répartition temporelle des 32 neurones du système. Bien que ces séries fournissent des résultats similaires à 0,1 % près, elles ne sont utilisées que comme illustrations. Un usage en tant que probabilités nécessite une démonstration à la fois pour l'établisement des valeurs, mais aussi pour les hypothèses simplificatrices.

Neurones	Temps de propagation	Apparition	Sur 300 secondes
2	29 µs	92,86 %	100 %
3	41 µs	21,73 %	100 %
4	53 µs	2,67 %	100 %
5	65 µs	0,310 %	100 %
6	77 µs	355 ppm	99,99998 %
7	89 µs	40,1 ppm	83,55 %
8	101 µs	4,42 ppm	18,04 %
9	113 µs	0,474 ppm	2,11 %
10	125 µs	0,049 6 ppm	0,22 %

TAB. 3.1 – Propriétés associées aux collisions en fonction du nombre de neurones mis en jeu.

Nous pouvons voir que les collisions à plus de 8 neurones sont très rares (l'estimation prévoie que moins de 3% des simulations de 5 minutes en contiennent), les délais de transmissions sont donc très majoritairement inférieurs à 100 µs, ce qui est au moins mieux que sur PAX.

Il est essentiel de rappeler qu'une collision de n neurones correspond à p neurones générant une activité en moins de $(p-n+1)\times 12+5$ µs. Par exemple, une activité de neurone suivie d'une autre après 10 µs, puis d'une troisième 10 µs plus tard est une double collision à deux neurones, mais pas une unique collision à trois neurones. En effet, il se déroule 20 µs entre les activités du premier et du troisième neurone, ceci signifie que le traitement du premier potentiel d'action est terminé lorsque le dernier survient. Ils ne se gênent donc pas mutuellement.

L'influence des collisions sur le système est donc relativement limitée : il n'y a pas de perte d'information comme avec l'AER, seul un délai supplémentaire est observable.

Et la plasticité ?

Nous n'avons pas parlé des temps de calculs associés à la plasticité. En effet, ils dépendent trop du modèle choisi et des diverses optimisations logicielles. Il faut cependant signaler qu'il ne sera pas possible d'effectuer un calcul de plasticité sur l'ensemble des 992 connexions sollicitées à 150 Hz décrites dans les pires cas. La capacité de calcul de l'ordinateur se mesure désormais en nombre de connexions plastiques simulées par seconde. Selon la complexité du modèle utilisé, il est possible de calculer de 8 000 à 50 000 modifications de connexion plastique par seconde. Bien que la limite inférieure aboutisse à un temps de calcul supérieur à 100 µs par connexion plastique, elle n'est pas contradictoire avec les possibilités de PAX. En effet, le calcul de plasticité ne dépend pas que de l'apparition d'un seul potentiel d'action, les calculs étaient alors répartis sur plusieurs périodes de 100 µs.

3.2.4 La répartition des tâches

Après avoir présenté la structure de l'extension, et avant de s'intéresser à la section suivante, il est intéressant de revenir sur l'organisation globale du système. La répartition des tâches et la communication entre les différents niveaux de calcul sont

présentés en figure 3.8. Dans ce schéma, nous pouvons voir apparaître les couches matérielle analogique (circuits), matérielle numérique (FPGAs), logicielle système (pilote de périphérique et bibliothèque d'interface) et logicielle utilisateur.

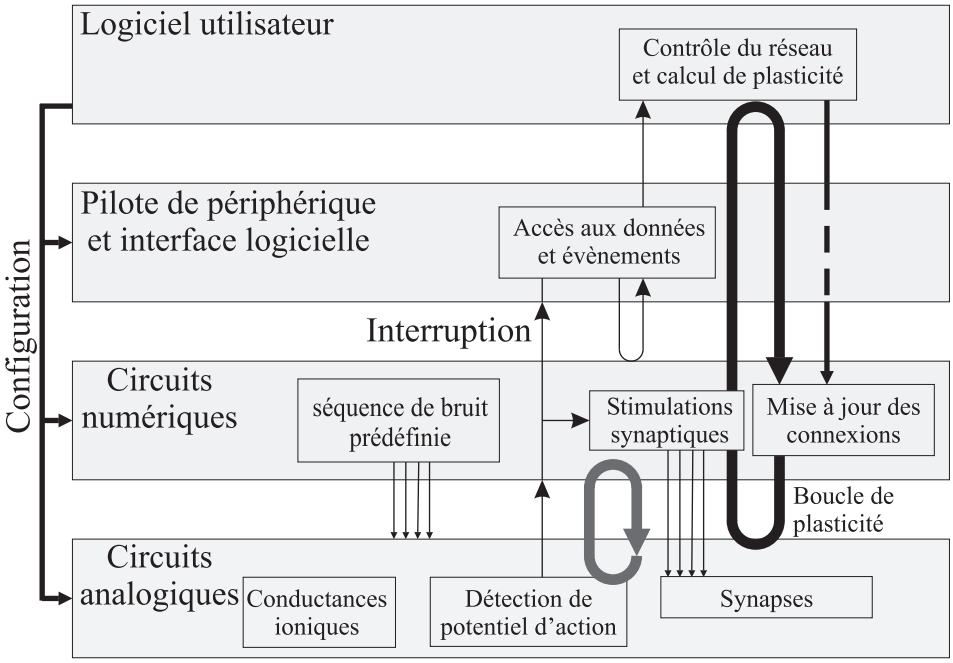

FIG. 3.8 – Répartition des calculs au sein du système PAX2

Nous pouvons voir dans un premier temps que le bruit synaptique est généré directement par les couches systèmes. Sa gestion n'est donc pas à la charge de l'utilisateur pendant la simulation.

Une première boucle est effectuée en bas niveau, entre les deux couches matérielles. Il s'agit de la gestion des connexions entre les neurones : l'information transite des circuits aux FPGA (potentiels d'action) et revient sous forme de stimulations synaptiques en fonction des poids synaptiques mémorisés.

Une seconde boucle, temporellement plus longue, se situe entre la couche matérielle numérique et les couches logicielles. L'information des potentiels d'action est propagée jusqu'aux couches logicielles. Le logiciel écrit par l'utilisateur fournit alors une mise à jour des poids synaptiques à la couche métérielle numérique.

3.3 Modifications de PAX

L'évolution structurelle est telle entre les deux générations de systèmes que, bien que le matériel de PAX soit toujours utilisé, il a fallu reconfigurer et reprogrammer l'ensemble des couches de simulation pour passer à PAX2.

La grande différence vient du fait que, désormais, un réseau peut être simulé sans logiciel programmé par l'utilisateur, à condition qu'il ne contienne pas de connexion plastique.

Les réalisations techniques évoquées dans cette partie ont très largement bénéficié du concours de Colin Lopez, ingénieur d'études travaillant dans notre équipe. Il

s'est chargé de la migration du système d'exploitation de l'ordinateur hôte et de la configuration de la majeure partie des blocs du FPGA de la carte PCI.

3.3.1 Le système d'exploitation

Désormais, l'ordinateur d'accueil fonctionne avec le système d'exploitation Linux *Ubuntu*, dérivé de la base *Debian*. Le principal intérêt est le changement de version majeure du noyau *Linux* (version 2.6.x). Cette version apporte une optimisation de la gestion de l'*hyperthreading* au niveau du noyau, ce qui devrait encore réduire les latences dues au système lui même. Il est également possible de gérer la préemption, ce qui signifie qu'une tâche en priorité utilisateur peut temporairement prendre une priorité supérieure à celle du noyau lui-même.

La migration du système signifie surtout la rédaction d'un nouveau pilote de périphérique. L'usage d'échanges à faible niveau de description permet alors une migration rapide de systèmes.

3.3.2 Le FPGA de la carte PCI

Sur la carte PCI, le FPGA gère, en plus des tâches de PAX, les connexions vers l'extension, la gestion du bruit synaptique et surtout le calcul des stimulations synaptiques en fonction des potentiels d'action.

Le bruit synaptique

La partie gérant le bruit de fond synaptique n'est pas très complexe. Il suffit d'expédier des stimulations synaptiques d'un poids déterminé selon des temps prédéfinis. La grande difficulté est de pouvoir mémoriser les séquences pour l'ensemble du réseau et sur toute la durée de la simulation.

Bien qu'il soit possible d'utiliser un codage optimisé pour minimiser les besoins de mémoire, l'ensemble des données nécessaires peut dépasser la dizaine de kilooctets par seconde. La solution choisie consiste alors à utiliser la mémoire de l'ordinateur et une faible mémoire tampon embarquée. Lorsque cette dernière ne contient plus assez de données, la carte effectue une requète pour se mettre à jour. La gestion est entièrement prise en charge par la bibliothèque de fonctions de sorte qu'il n'est pas nécessaire de s'en préoccuper pendant la simulation.

Les connexions du réseau

L'envoi des stimulations synaptiques, consécutives à l'émission d'un potentiel d'action, s'effectue par une consultation exhaustive des connexions possibles. L'ensemble des poids, appelé *matrice de connexion*, est mémorisé dans une RAM. Pour chaque poids synaptique, l'adresse d'accès est directement constituée à partir des numéros des neurones présynaptiques et postsynaptiques. Les potentiels d'action arrivants sont placés dans une file d'attente. Pour chacun d'entre eux, l'ensemble des connexions possibles est consulté. La lecture d'un poids non nul provoque la génération d'une stimulation synaptique. Cette dernière est placée dans une file d'attente jusqu'à son envoi vers le système. La file d'attente de sortie est inutile pour les stimulations à destination des circuits de la carte PCI, elle est cependant utilisée pour synchroniser les envois locaux avec ceux destinés à l'extension.

Schéma-bloc

Le schéma-bloc du FPGA de la carte PCI est présenté figure 3.9.

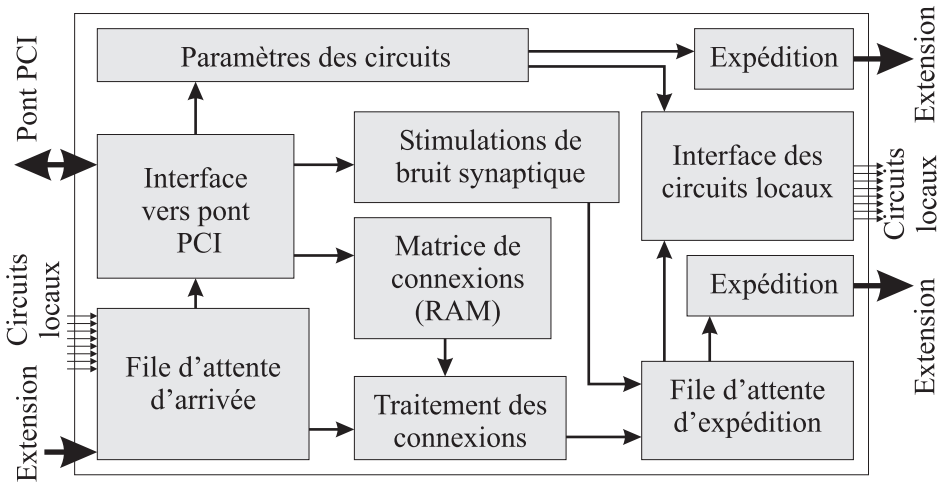

FIG. 3.9 – Schéma-bloc de la structure du FPGA de la carte PCI

Ce schéma-bloc nous montre que les fonctions qui constiuaient PAX sont devenues minoritaires. Au niveau de l'interface PCI, seuls les transferts sur 32 bits sont maintenant utilisables. Étant donné que le logiciel ne peut pas lire plus d'un évènement à la fois, les potentiels d'action sont placés dans une file d'attente.

Les nouvelles capacités utilisent quant à elles bien plus de ressources. Les blocs de mémoire sont particulièrement sollicités pour les files d'attente et le stockage des poids synaptiques. Le dialogue est également plus élaboré avec le pilote logiciel puisque désormais, plusieurs types d'évènements peuvent provoquer l'envoi d'une interruption.

3.3.3 Un système réactif

Désormais, le système ne suit plus une horloge fixe. Il s'agit d'une structure réactive mise en activité dès qu'un noyau de calcul analogique émet un potentiel d'action. Comme pour PAX, lorsqu'il n'y a aucune donnée à traiter, le processus de simulation se met en sommeil et rend la main au système d'exploitation. Du point de vue du programme de la couche utilisateur, tout se déroule donc comme s'il y avait en permanence de l'activité à récupérer depuis le réseau.

L'ensemble de calcul formé par le FPGA de la carte PCI et la bibliothèque d'instructions prend en charge une grande partie des fonctions temporellement critiques. Il constitue une couche d'abstraction supplémentaire sur le système. Alors que PAX interfaçait un logiciel avec des neurones, PAX2 interface un logiciel à un réseau fonctionnel. La figure 3.10 présente le déroulement temporel d'une simulation.

Le temps de 100 µs indiqué pour le calcul est un ordre de grandeur incluant l'ensemble des phénomènes de connexion et le plasticité. Nous pouvons voir deux files d'éxécutions en parallèle : le réseau sur la partie matérielle, et la plasticité en logiciel. La gestion du bruit exige très peu de ressources.

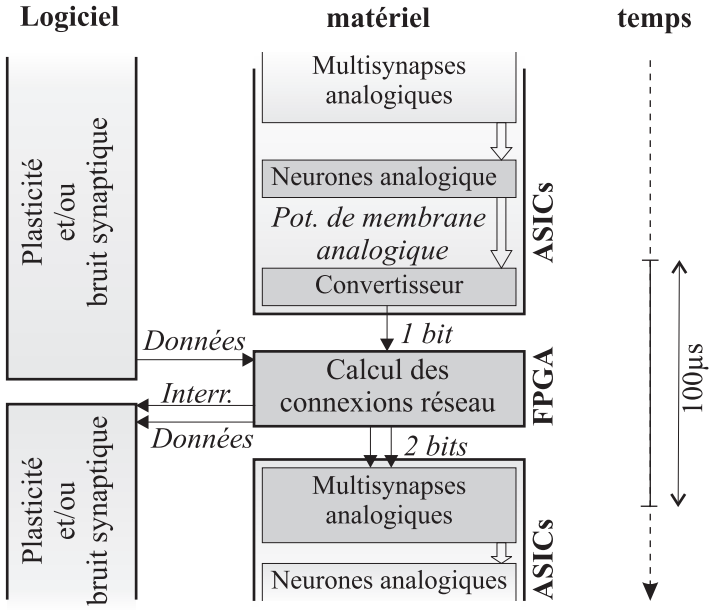

FIG. 3.10 – Déroulement d'une boucle de simulation sur PAX2

3.4 Résultats

Nous abordons désormais une série de simulations réalisées sur le système PAX2. La mise en œuvre du système a révélé un principal écueil : les cœurs de simulation fonctionnels, sur le circuit *Claudia*, ne correspondent pas au modèle recherché. Bien que les caractérisations réalisées (courbes f/i) donnent des résultats intéressants, leur usage en simulation montre que leur activité prend la forme de trains de potentiels d'action uniquement. L'ensemble des simulations prévues ne pourra, hélas, pas être réalisé puisque seuls les circuits *Trieste* génèrent une activité correspondant au modèle. Les simulations proposées ici ne peuvent toujours pas prétendre décrire un réseau biologique.

3.4.1 Une interface de sortie

Cette première expérience ne constitue pas, à proprement parler, une simulation de réseau de neurones. Il s'agit de fournir une couche de sortie analogique pour des réseaux dont les modèles n'offrent pas de valeur exploitable du potentiel de membrane (typiquement, le modèle *integrate and fire* et ses dérivés), ou dont l'échelle de temps est différente de 1 (et interdit donc le dialogue avec le vivant). Cette mesure a été effectuée dans le cadre du projet européen *SenseMaker* pour le contrôle d'électronique d'interface sensorimotrice.

Dans cette configuration, les circuits neuromimétiques n'interviennent pas dans le réseau de simulation. Il n'est donc pas nécessaire d'adapter les deux systèmes pour qu'ils fonctionnent à la même échelle de temps. Il suffit d'échantillonner le fonctionnement du système de simulation. Le comportement obtenu doit alors être reproduit sur PAX2.

Les premiers potentiels d'actions générés sont présentés figure 3.11. Les données d'activité ont été produites par le laboratoire *Intelligent Systems Engineering Labo-*

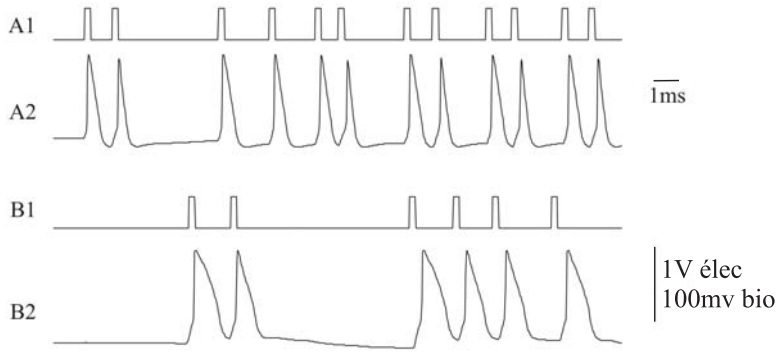

FIG. 3.11 – Début de l'activité reproduite par PAX2

ratory de L'*University of Ulster, McGee campus*. La simulation met en œuvre des modèles qui suivent le formalisme d'Hodgkin et Huxley simulés numériquement par des FPGAs.

Du point de vue du système, les neurones sont indépendants, il n'y a aucune connexion entre eux. L'activité leur est transmise par l'entrée de bruit synaptique. La contribution du bruit est réglée pour que la génération d'un potentiel d'action soit systématique à chaque stimulation. Elle n'est cependant pas maximale pour influer le moins possible sur la forme même des potentiels d'action.

3.4.2 Réseau bistable

Le premier exemple de réseau simulé à l'aide de PAX2 n'est pas plastique. La capacité du système à gérer les connexions de façon autonome est tout de même exploitée. Il s'agit du réseau présenté figure 3.12. Les neurones excitateurs (1e et 2e) sont stimulés pour osciller spontanément en l'absence d'inhibition. Les neurones inhibiteurs (1i et 2i) doivent, eux, recevoir une excitation pour émettre une activité. Le poids de chaque connexion du réseau est élevé. Un tel réseau, bien que peu complexe du point de vue du nombre d'unités de calcul, permet trois comportements bien distincts.

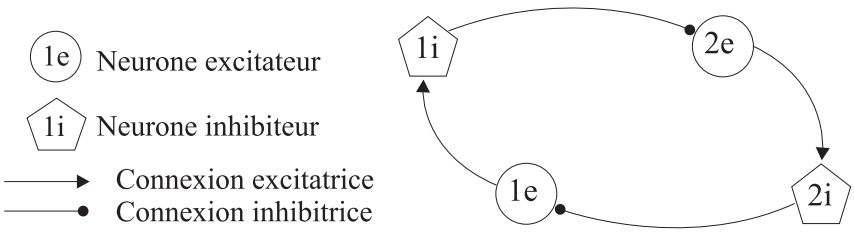

FIG. 3.12 – Quatre neurones pour un réseau aux multiples comportements

Un réseau à effet mémoire

Le comportement principal de ce réseau constitue un effet mémoire. Si l'un des neurones excitateurs est actif, il provoque une inhibition de l'autre. Nous nous retrouvons alors dans le cas où seule la moitié des neurones du réseau fournit une activité (couple 1x ou 2x). Pour générer un basculement de l'activité, d'un couple de neurones vers l'autre, il est nécessaire d'apporter des stimuli. Ce réseau est donc capable de garder, sans aucune plasticité, une trace de ses stimulations passées.

Une stabilisation mutuelle

Si les poids sont suffisamment bien équilibrés, le comportement est radicalement différent. En effet, les deux parties en concurrence s'autorégulent et l'ensemble du réseau fonctionne à la même fréquence. Les stimulations et le bruit peuvent cependant influer sur la synchronisation entre les deux parties du réseau (1x et 2x). Selon les modes, elles s'activent soit en phase (simultanément) soit en opposition de phase (alternativement).

Le passage d'un mode à l'autre est cependant très sensible, et le bruit électrique est suffisant pour provoquer une transition. Ce comportement est relativement difficile à produire, mais reste cependant très stable une fois initié. Il a été obtenu par des méthodes assimilables à celles des réseaux supervisés. Ceci signifie que les poids synaptiques n'ont pas été réglés en fonction d'un modèle lié à l'activité des neurones, mais par rapport à un objectif de comportement à l'échelle du réseau.

Des oscillations stables

Le nom donné à cette activité est une contradiction du point de vue électronique. Cependant, selon l'échelle comportementale observée, elle peut prendre tout son sens. Nous pourrions en effet parler de la stabilité d'un mode dans le cas de la stabilisation mutuelle. Ici la contradiction est poussée un peu plus loin : la stabilité de l'activité signifie une alternance régulière du caractère oscillant, comme présenté en figure 3.13.

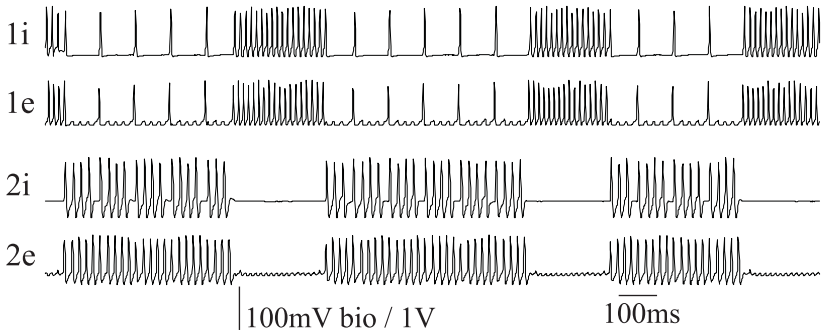

FIG. 3.13 – Le caractère bistable du réseau, ou la constance dans l'alternance des oscillations

Un tel comportement est très intéressant puisqu'il met en évidence le rôle de la modulation présent dans les modèles excitateurs. La longueur du phénomène reproduit illustre l'influence de canaux à cinétique lente. La mise au point de ce réseau est délicate et a nécessité, elle aussi, des méthodes de réseaux supervisés.

Le comportement obtenu peut être qualifié de stable puisque, malgré les variations observables, il ne dérive pas vers un autre comportement.

Réalisme du réseau

Selon les circonstances, le réseau et les comportements qui lui sont associés peuvent se retrouver dans le vivant. Il faut, pour cela, considérer que chaque moitié comportant deux neurones (1x ou 2x) reflète en pratique une population de neurones.

Le réglage des poids synaptiques ne serait plus effectué par un superviseur, mais par la plasticité intrinsèque des synapses. Le modèle biologique de neurones, et donc le modèle de plasticité correspondant, ferait naturellement évoluer le réseau vers l'un des trois comportements précédents.

3.4.3 Un chenillard plastique

Le réseau de départ

Nous allons présenter ici les transformations apportées par la plasticité sur un réseau de six neurones. Toutes les connexions de ce réseau sont actives et plastiques. Chaque neurone est donc potentiellement relié à tous les autres. Cependant, lors de l'étape initiale, seuls cinq poids synaptiques sont non nuls. Le réseau se présente alors comme un chenillard non rebouclé (Fig. 3.14 :A). L'activité du réseau se synchronise sur celle du premier neurone, faiblement stimulé. Les autres neurones n'oscillent pas sans stimulation du réseau.

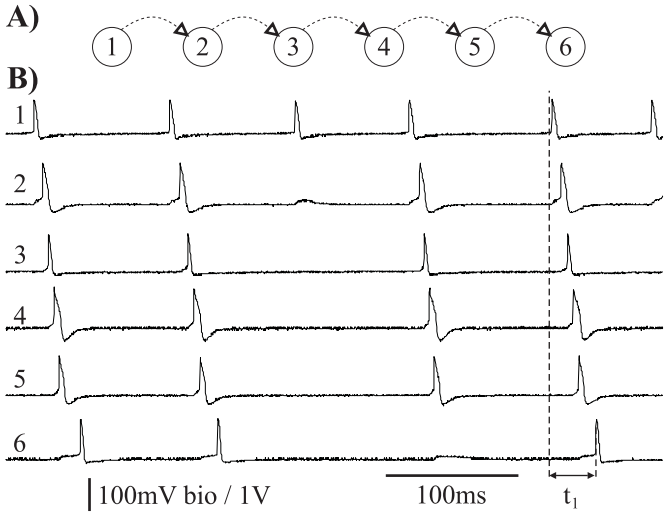

FIG. 3.14 – Le réseau et son activité associée avant application de plasticité

L'activité du réseau (Fig. 3.14 :B) met en évidence l'insuffisance des poids synaptiques pour que la transmission de l'information soit fiable. Lorsque l'activité est intégralement transmise, celle-ci subit un délai important (t_1 sur la figure 3.14). En effet, t_1 fluctue beaucoup (entre 20 ms et 40 ms).

La plasticité est une STDP prenant en compte la loi de variation illustrée figure 1.12 (exponentielles, limites infinies nulles et discontinuité en zéro). Elle est

associée aux phénomènes de saturation du poids synaptique ainsi que d'efficacité de chaque potentiel d'action.

La simulation

Lorsqu'elle est appliquée, la plasticité modifie le réseau en suivant les étapes présentées ci-dessous.

- Renforcement des connexions synaptiques : les connexions synaptiques non nulles sont renforcées, certaines autres sont naissantes (1 vers 3, 2 vers 4, *etc*). La transmission des évènements est maintenant systématique, elle est également plus rapide.
- Création de nouvelles connexions : les connexions d'un neurone n vers un neurone m ($m > n$) sont désormais non nulles. La propagation s'accélère, elle devient quasiment instantanée pour certains neurones (différences de temps de l'ordre de la demi milliseconde). Il est désormais possible de retirer un neurone du réseau sans en modifier la fonctionnalité. Il s'agit de l'effet de robustesse apporté par la plasticité.
- Optimisation des délais de transmission : le réseau se stabilise dans une configuration où le temps de transfert de l'information est optimal. La figure 3.15 nous montre le réseau final (A) et son comportement (B). L'ordre de propagation d'origine n'est plus vérifié. Pour analyser fonctionnellement le réseau final, nous allons assimiler le neurone 1 à une couche d'entrée et le neurone 6 à une couche de sortie. Les neurones 2, 4 et 5 sont désormais inutiles : ils sont actifs après le neurone 6. Le neurone 3, quant à lui, n'est pas indispensable puisqu'une connexion directe relie le neurone 1 au neurone 6. Cependant, comme il est plus réactif que le neurone 6, sa présence augmente la stimulation envoyée au neurone 6, et raccourcit donc la réponse de l'ensemble du réseau.

Le réseau final stabilisé

Dans la figure 3.15 :A, une dimension verticale a été ajoutée pour représenter la réactivité des différents neurones : neurone 1 à l'origine de l'activité, neurone 3 le plus réactif, neurones 2 et 5 les plus lents.

La durée de la simulation n'est que de quelques secondes. Le temps total de propagation de l'information (t_2 sur la figure 3.15) est maintenant inférieur à 5 ms. La suppression d'un neurone n'influe que très peu sur le réseau final, contrairement au réseau initial. Ces résultats sont conformes aux conséquences théoriques de la STDP présentée, à savoir :

- les connexions allant d'un neurone précoce vers un neurone tardif voient leur poids augmenter (renforcement des connexions et du réseau) ;
- les connexions allant d'un neurone tardif vers un neurone précoce ont un poids qui diminue (exclusion des neurones les plus lents).

Ce réseau n'aboutit pas à des connexions modérées mais toujours extrémales. Cela est dû au fait qu'il n'existe qu'une seule source d'activité indépendante : le neurone 1.

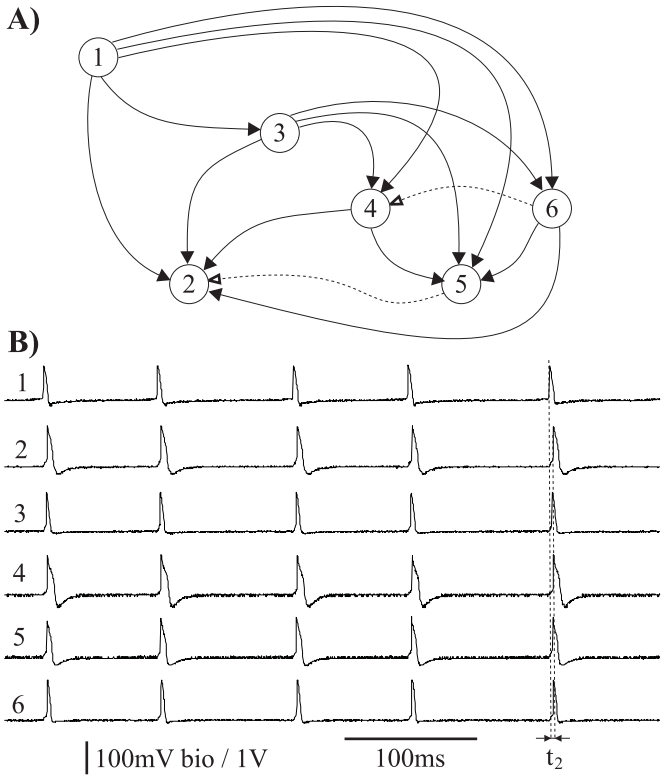

FIG. 3.15 – Le réseau et son activité associée après convergence des effets de plasticité

Influence des paramètres des neurones

La vitesse *temps réel* de la simulation permet de réaliser toute une série de mesures par rapport aux disparités observables d'un neurone à l'autre. Les phénomènes, présentés ici, ne sont évidemment valables que pour le modèle de plasticité choisi.

Ordre initial dans le réseau Plusieurs simulations ont été effectuées en utilisant différentes configurations de départ. Par exemple, l'ordre du chenillard d'origine a été modifié, mais le réseau final est resté identique. La hiérarchie observable dans le réseau final parmi les neurones est donc uniquement due aux propriétés de chacun d'entre eux. Cette mesure met en avant l'influence de la dispersion.

Changement d'échelle pour certaines propriétes Une série de mesures a été réalisée en modifiant la réactivité des neurones (modification du courant de compensation). S'il est mieux stimulé (et donc plus réactif) le neurone 6 (de réactivité moyenne) est d'autant plus favorisé dans le réseau final. Mais l'observation principale est que l'ensemble de réseau lui-même est plus réactif dans sa configuration finale. De même, en diminuant la réactivité du neurone 3, t_2 atteint les valeurs de l'ordre de 7 ms. Les mêmes phénomènes ont été observés avec le neurone 4.

La réactivité de l'ensemble du réseau dépend donc de la combinaison des réactivités de chaque neurone, et non, comme nous aurions pu l'espérer, de celle du plus rapide ou du plus lent. Nous voyons aussi que les neurones privilégiés sont les plus réactifs.

3.5 Analyse

Au terme de ce chapître, il nous reste à en dresser le bilan. Bien que le nombre de neurones exploitables n'ait pas pu être accru dans le système PAX2, les nouvelles fonctionnalités restent néanmoins bien présentes.

3.5.1 Points essentiels

Le système PAX2 est constitué du système PAX sur lequel une extension contenant les circuits *Claudia* a été greffée. Cependant, ces derniers mettent en évidence les lacunes de l'architecture à paramètres figés. Le modèle réellement implanté est peu fonctionnel et ne peut pas être utilisé dans le cadre du projet *SenseMaker*.

Une partie du calcul a été intégrée aux FPGA. Le système est désormais entièrement réactif et permet d'optimiser les performances du processeur central (système *event-based*). Bien que non encore exploitée, une interface vers les neurones biologiques a été réalisée. Les couches matérielles sont capables de simuler un réseau complet. L'ordinateur d'accueil n'intervient que pour les séquences de bruit à envoyer et pour le calcul de plasticité.

3.5.2 Conclusion

Les expériences réalisées montrent un système opérationnel et réactif. Bien que le pire cas théorique soit défavorable par rapport aux performances de PAX, la perte est relativement limitée face au gain obtenu en terme de taille de réseau simulé. Le fonctionnement en situation réelle affiche cependant des performances bien meilleures.

La simulation mettant en œuvre la plasticité (cf. 3.4.3) montre que l'apparition d'une erreur temporelle, même faible, lors de l'accès aux neurones peut jouer sur l'évolution du réseau. La communication devient donc l'élément crucial des prochains systèmes : les temps de propagation doivent être identiques quelle que soit la connexion considérée. Le mode de communication choisi, s'il est plus performant que précédemment, devra donc être repensé sur les systèmes suivants pour permettre des délais de propagation synaptique suffisamment homogènes.

Un autre point soulevé par les différents utilisateurs du système concerne l'interface logicielle fournie pour contrôler le réseau. Bien qu'elle soit pleinement fonctionnelle pour réaliser une simulation, elle se révèle par contre trop sommaire à bas niveau pour réaliser des tests. Des solutions ont été mises en place, mais il convient à l'avenir de considérer cette éventualité dès la conception du système. Il est des situations où l'usage des neurones en accès direct est préférable, malgré la perte de puissance de calcul qui en découle. Il faut donc laisser à l'utilisateur le choix entre la puissance de calcul offerte et la flexibilité de l'accès aux neurones.

3.5.3 Bilan par rapport au projet *SenseMaker*

PAX2 fut la dernière version du système de simulation de réseaux de neurones conçu dans le cadre du projet *SenseMaker*. Les objectifs de calcul de réseau adaptatif en temps réel ont été atteints. Cependant, les difficultés évoquées sur le circuit *Claudia* limitent la capacité du système final en terme de taille de réseau.

Le système est maintenant destiné à être utilisé dans le cadre des neurosciences computationnelles. Des simulations hybrides, exploitant la capacité de PAX2 à se connecter aux instruments de contrôle des neurones, sont également planifiées.

Chapitre 4

La Voix de l'intégration

Ce dernier chapitre présente un système de simulation de troisième génération dont la réalisation est en cours. Il s'appuie sur les retours d'expérience des deux systèmes précédents, ainsi que sur les besoins et objectifs du nouveau contrat européen par lequel il est financé, le projet FACETS. Bien que nous cherchions à concevoir un système capable de contenir des réseaux de l'ordre de la centaine de neurones, la priorité reste une grande fidélité au comportement biologique des neurones.

Cependant, un grand nombre de neurones, signifie également un grand nombre de circuits, et un grand encombrement spatial. Comment, alors, parvenir aux hauts débits nécessaires sur l'ensemble du système pour que ce dernier fonctionne avec des délais acceptables ? Seule une intégration de l'ensemble du système permet d'entrevoir une solution. Pour suivre cette voie, nous sommes alors contraints d'être à l'écoute de ses exigences.

Après une présentation du projet FACETS, nous évaluerons les besoins d'un simulateur pour des réseaux à grande échelle ainsi que les fondations du nouveau système, actuellement en chantier. Nous verrons également le nouveau circuit conçu pour en être la brique de base, ainsi que les premiers résultats de ses tests. Nous pourrons alors, dans le bilan, esquisser les applications envisageables pour ce simulateur.

4.1 Le projet FACETS

Le projet européen FACETS (FP6-2004-IST-FETPI) se situe dans la lignée du projet *SenseMaker*, tout en élargissant son champ d'étude.

4.1.1 Le projet dans son ensemble

FACETS (Fast Analog Computing with Emergent Transient States in neural architectures) a pour objectif de créer une base théorique et expérimentale (données et outils) pour permettre l'émergence de nouveaux paradigmes sur l'étude des réseaux de neurones biologiques. L'approche expérimentale s'effectue dans le cadre de la vision, fonction computationnelle massivement parallèle confrontée à des stimuli variant rapidement.

Cette approche se base sur des simulateurs analogiques conçus à partir de circuits neuromorphiques et/ou neuromimétiques. Ils bénéficient d'une caractérisation à la fois structurelle et fonctionnelle de réseaux biologiques *in-vivo* et *in-vitro*.

Les trois lignes de recherche principales sont :

- la caractérisation expérimentale de cellules et réseaux corticaux (*in-vivo* et *in-vitro*) ;
- l'étude des modèles théoriques informatiques de cellules et réseaux ;
- la conception, la réalisation et la mise en œuvre de circuits intégrés et systèmes électroniques selon les modèles biologiques.

Les systèmes conçus doivent permettre d'accéder à de nouveaux paradigmes de traitement de l'information. Le cortex visuel est choisi comme exemple de référence pour chacun des trois points.

Ce projet est un défi scientifique, technique mais surtout organisationnel puisqu'il regroupe quinze équipes partenaires.

4.1.2 Au niveau de l'IXL

L'équipe *Ingénierie des Sytèmes Neuromorphiques* de l'IXL se charge de la réalisation d'un système de simulation de réseaux de neurones fidèle à l'échelle de la cellule. Les modèles, proposés par l'UNIC (*Unité de Neurosciences Intégratives et Computationnelles*, CNRS, Gif sur Yvette), reflètent essentiellement la diversité observable dans les réseaux biologiques. Cette diversité est rendue par la dispersion des composants, mais surtout, un plus grand nombre de jeux de paramètres qu'il n'a été utilisé pour le projet *SenseMaker*.

Une validation de l'implantation des modèles doit ensuite être réalisée. S'ensuit une série de simulations pour extraire les propriétés intéressantes au niveau des modèles de cellules et des algorithmes de plasticité. Ces propriétés seront alors intégrées dans des réseaux de plus grande échelle implantant des modèles moins complexes.

La capacité du système à fonctionner en temps réel, ainsi que sa grande fidélité aux comportements biologiques seront alors utilisées. Les modèles seront mis à l'épreuve, pour en analyser les résultats, et éventuellement les rectifier. Il est à noter que, pendant la réalisation du premier système financé par FACETS, PAX2 servira de plateforme pour les opérations d'analyse.

4.2 Vers un système à grande échelle

Pour simuler des réseaux de taille plus importante, les capacités du système doivent être augmentées dans des proportions au moins équivalentes. PAX2 nous a cependant montré que la puissance de calcul d'un ordinateur ne pourrait plus suffire au-delà d'un réseau de 50 neurones, et que la communication au sein du réseau lui-même devenait problématique.

Nous allons voir ici en détail les impasses où mène une architecture centralisée comme celle de PAX et PAX2. Une halte théorique nous permettra d'entrevoir une solution élégante et optimale. Nous nous attarderons finalement sur certaines caractéristiques qui, bien que relevant d'une architecture centralisée, ne doivent pas être remises en cause.

4.2.1 Répartition de la puissance de calcul

La communication au cœur du problème

Nous avons déjà évoqué l'engorgement que provoque la transmission de l'ensemble des stimulations synaptiques. Les besoins en communication croissent avec le carré

du nombre de neurones, alors que la taille physique du système diminue ses capacités de communication.

La taille du réseau est également problématique lorsqu'il s'agit de fournir une puissance de calcul suffisante pour calculer la plasticité de l'ensemble des connexions. Un ordinateur seul ne suffit plus. Cependant, l'usage d'une ferme de calcul est trop aléatoire. En effet, les nœuds de calcul sont reliés entre eux par une interface *ethernet*, or cette liaison ne peut pas assurer de communications en temps réel. De même, concevoir un système architecturé autour d'ordinateurs accueillant chacun une partie des neurones du réseau nécessite une plateforme de communication temps réel entre eux.

Augmenter la taille du réseau nécessite donc une architecture capable de véhiculer un grand volume d'informations. Il faut également ajouter que le modèle de la STDP nécessite des informations relatives entre les neurones présynaptique et postsynaptique (lorsque l'un est actif, la variation du poids dépend de l'activité de l'autre).

Le compromis entre les débits et la puissance de calcul

L'architecture la plus adaptée est celle qui répartit plusieurs sous-unités de calcul sur des groupes de neurones, tout en les reliant au sein d'une structure temps réel (Fig 4.1). Sur un tel système, il n'y a plus de calcul centralisé, ainsi les informations de connexion sont réparties au sein des sous-unités où elles seront exploitées.

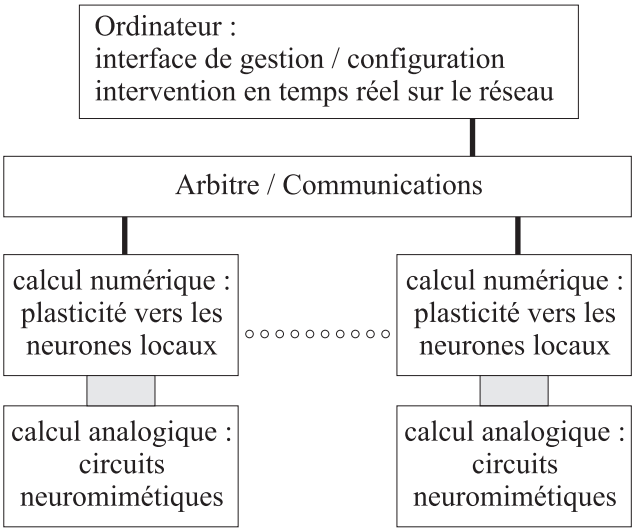

FIG. 4.1 – Principe de répartition de calcul

Si le nombre de neurones à contrôler est suffisament réduit, il n'est pas nécessaire d'utiliser un ordinateur pour assurer le rôle d'une sous-unité. Il est possible d'imaginer un microcontrôleur chargé à la fois du calcul de plasticité et de l'accès aux circuits. Ce composant peut être soit un composant discret, soit configuré dans un FPGA.

La communication minimale au sein de l'interface temps réel est uniquement constituée par l'activité des différents neurones du réseau. *A contrario*, le calcul minimal consiste à générer les grandeurs relatives aux neurones gérés localement et qui sont impliquées dans le calcul de plasticité. Étant donné que ce calcul nécessite des

informations sur l'activité d'un neurone au moment ou l'autre émet un potentiel d'action, la mise en œuvre ne peut réunir ces deux cas. Une optimisation de la puissance de calcul nécessitera bien entendu une surcharge en communications pour véhiculer les grandeurs calculées ; l'inverse est également vrai.

Illustration des performances par un cas concret

Sur un tel système, il est alors possible d'envisager un envoi de potentiel d'action par cycle d'horloge. Pour une structure synchronisée à 10 MHz et contenant 1 000 neurones, il est donc possible de garantir que les potentiels d'action soient transmis en 100 µs dans le pire cas. Si ce système est constitué de 25 sous-unités de 40 neurones, il leur faudra, dans chacune d'entre elles, assurer le calcul relatif à 40 neurones postsynaptiques et 1 000 neurones potentiellement présynaptiques.

4.2.2 Technique d'optimisation du calcul de plasticité

Bien que nous venions de voir que la propagation d'informations pouvait être résolue sur un système capable de simuler un grand réseau, il nous reste néanmoins à dessiner l'architecture d'une sous-unité de calcul capable de prendre en charge les grandeurs de centaines de neurones. Bien que la charge ait été réduite par rapport au problème initial, elle reste plus importante que celle gérée par PAX2, malgré l'abandon du système d'exploitation. Il est donc inévitable de procéder à une série d'optimisation.

Analyse du calcul de STDP traditionnel

La figure 1.12 représente le modèle le plus simple de plasticité qui soit cohérent avec la fidélité des modèles de neurones. Il est basé sur l'usage de la fonction exponentielle dans ses valeurs négatives. La mise en œuvre du calcul s'effectue en deux parties. En effet, dans une connexion, chaque potentiel d'action d'un neurone intervient à la fois avant et après un potentiel d'action de l'autre neurone. Pour chacun d'entre eux, il faut prendre en compte la contribution correspondante par rapport aux potentiels d'action passés. Deux fonctions distinctes sont alors présentées selon la fonction (présynaptique ou postsynaptique) qu'occupe le neurone dans la connexion considérée. Il s'agit des fonctions dites de *potentiation* (neurone postsynaptique) et de *dépression* (neurone présynaptique).

Les deux fonctions nommées respectivement $ltp(t)$ et $ltd(t)$ (*long term potentiation* et *long term depression*) obéissent chacune à deux règles énoncées par les expressions 4.1.

$$\begin{cases} ltp(t<0) = ltd(t<0) = 0 \\ \delta w(t) = ltp(t) - ltd(-t) \end{cases} \tag{4.1}$$

Nous avons vu que, dans l'algorithme de STDP le plus commun, les deux fonctions suivent l'évolution d'une exponentielle décroissante. De même, dans le reste des équations, nous retrouvons essentiellement des combinaisons linéaires de ce phénomène. Il est donc essentiel de mener à bien le calcul de ces fonctions exponentielles au plus bas niveau pour réaliser le calcul de la plasticité. Une fois le principe d'un tel calcul validé pour le temps réel, l'ensemble des phénomènes s'implante suivant le même principe.

L'enjeu de parvenir à un calcul matériel permet de mieux maîtriser l'aspect temporel de la simulation. Nous pourrons alors rendre compte de l'ensemble des phénomènes de saturation ou d'efficacité sans solliciter de traitement en file d'attente comme avec un processeur ou un contrôleur.

Le calcul de décroissances exponentielles

Nous allons d'abord voir brièvement l'ensemble des méthodes qui permettent le calcul d'une évolution en exponentielle décroissante.

- L'usage de tables précalculées : il s'agit de la méthode la plus couramment employée dans les applications embarquées. Elle souffre cependant de défauts suffisament importants qui rendent cette méthode peu intéressante. En effet, pour obtenir une implantation suffisamment configurable, la fonction réalisée doit être mise à l'échelle en amplitude ainsi qu'en évolution temporelle. Ceci implique des circuits de mise à l'échelle non précalculés.

  Les accès à une table s'effectuent séquentiellement. La mise en œuvre de calcul hautement parallèle à l'aide de cette technique nécessite alors plusieurs tables, ce qui est très gourmand en circuits.

- Le recours aux méthodes de convergence : cette méthode se repose sur l'usage de suites mathématiques dépendant de x et convergeant vers e^x. Il s'agit de la définition même de la fonction exponentielle que l'on retrouve en équation 4.2. Il reste bien sur possible d'utiliser des fonctions optimisées pour le calcul numérique qui convergent plus vite ou qui nécessitent moins de calcul par itération.

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} \tag{4.2}$$

  Bien optimisée, cette méthode nécessite peu de circuits, par contre, le nombre de cycles nécessaires à son calcul est difficilement prévisible. Il est possible de le majorer puisque l'on travaille sur un compact (segment de valeurs fermé et borné), mais cela devient rédhibitoire pour un usage massivement parallèle en temps réel.

- L'approximation polynômiale : il s'agit de réaliser une approximation à l'aide d'un polynôme de degré relativement faible. Cette technique est un compromis entre les deux précédentes. En effet, il est possible d'optimiser la fonction pour le temps de calcul ou pour l'espace utilisé selon le nombre de niveaux de *pipeline*. De plus la définition du polynôme permet de pré-évaluer l'erreur réalisée sur chaque calcul.

- Le (co)processeur mathématique. L'usage d'un processeur évolué ou d'un coprocesseur mathématique revient en quelque sorte à déplacer le problème plutôt qu'à le résoudre. La méthode réellement employée est l'une des trois précédentes, voire une combinaison d'entre elles. L'intérêt par contre, est une optimisation poussée du calcul, puisque ce genre de circuits est le fruit de longues années de recherche et développement spécifiques. L'inconvénient reste que, sur chaque composant, le calcul est séquentiel.

- L'évolution analogique de la tension aux bornes d'un condensateur : cette technique est irréaliste à cause des contraintes de calcul évoquées en 2.1.1 (page 45), du dialogue mixte qui serait nécessaire, ainsi que du nombre de condensateurs

qu'elle implique. Elle amène pourtant des concepts très intéressants, notamment par l'usage d'un système de calcul peu performant mais qui fonctionne en permanence et en continu. Nous voyons aussi une description d'un système réactif : en augmentant le nombre de variables d'état du système (par les valeurs des tensions sur les condensateurs), il n'est plus nécessaire de recourir à une mémorisation des instants auxquels se sont produits les évènements.

Aucune des méthodes précédentes n'est réellement satisfaisante, cependant, les concepts présentés sur la simulation analogique peuvent être appliqués sur des techniques numériques. En effet, les méthodes numériques présentées permettent de calculer des exponentielles de données aléatoires, ce qui ne sera jamais notre cas. Nous pouvons alors simplifier grandement ce calcul. L'astuce est ici de résoudre en temps réel l'équation différentielle dont l'exponentielle est solution (4.3). Si cette résolution est suffisament optimisée elle peut être parallélisée à grande échelle.

$$f(t) = e^{-t} \Rightarrow f'(t) = -f(t) \tag{4.3}$$

Nous obtenons alors :

$$\frac{df}{dt}(t) = -f(t) \tag{4.4}$$

$$\frac{f(t+dt) - f(t)}{dt} = -f(t) \tag{4.5}$$

$$f(t+dt) = f(t) - f(t) \cdot dt \tag{4.6}$$

Étant donné que nous travaillons en numérique, nous tiendrons compte des quanta de simulation en écrivant :

$$f(t+\delta t) = f(t) - f(t) \cdot \delta t \tag{4.7}$$

Cette formule est très intéressante pour du calcul réalisé par FPGA. En effet, si $f(t)$ est mémorisé dans un registre sous forme binaire et que nous choisissons $\delta t = 2^{-n}$, la muliplication $f(t) \cdot \delta t$ s'effectue par un décalage de bits, ce qui signifie que son implantation est réalisée par un jeu de cablage sans utiliser le moindre composant. L'implantation de cette fonction en circuits logiques est présentée figure 4.2. Elle ne necessite qu'un registre et un soustracteur sur $2n$ bits. L'entrée *init* du registre permet d'initialiser la valeur de l'exponentielle pour $t = 0$.

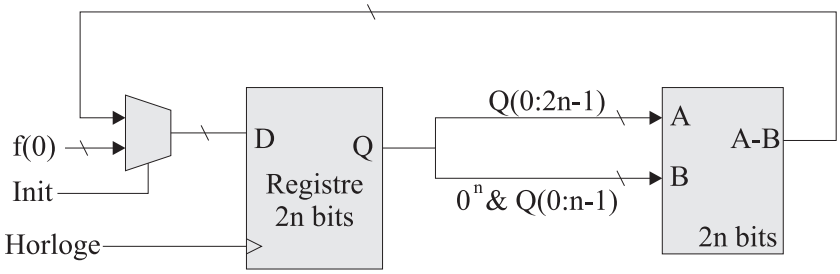

FIG. 4.2 – Circuit de calcul d'un signal de la forme $e^{-t/\tau}$

La constante de temps τ du signal se calcule par la méthode de l'intersection de la dérivée en 0 avec l'axe des abscisses. Elle dépend de la fréquence de fonctionnement

F_0 ou de sa période P_0. Nous obtenons :

$$\tau \cdot f'(0) = -f(0) \tag{4.8}$$

$$\tau = -\frac{f(0)}{f'(0)} = \frac{f(0)}{\frac{f(\delta t)-f(0)}{P_0}} \tag{4.9}$$

$$\tau = -\frac{f(0)}{f(0)\cdot(1-\delta t)-f(0)} \cdot P_0 \tag{4.10}$$

$$\tau = \frac{P_0}{\delta t} = \frac{P_0}{2^{-n}} = P_0 \cdot 2^n \tag{4.11}$$

Ainsi, si $n = 10$ et $F_0 = 1\,\text{MHz}$, nous obtenons $\tau = 1,024\,\text{ms}$

Évaluation de la méthode

Cette méthode a été configurée dans un FPGA pour être évaluée. Elle utilise 20 bascules et 30 LUTs ($n = 10$). Configurée sur un *Spartan2e 600* de marque *Xilinx* (XC2S600-6), la fréquence maximale d'utilisation est supérieure à 130 MHz. Elle permet de réaliser des fonctions possédant une constante de temps inférieure à $8\,\mu\text{s}$ ($n = 10$).

Le calcul étant réalisé en utilisant des entiers binaires, il rentre dans le cadre des techniques dites de *virgule fixe*. Le résultat n'est donc pas aussi précis qu'une valeur calculée en *virgule flottante*. Le tableau 4.1 nous montre l'écart maximal entre la valeur calculée selon cette technique, et celle calculée par *Maple*.

n	Erreur relative à f(0)	en valeur LSB
6	1,6%	1 LSB (0,977)
7	0,8%	1 LSB (0,989)
8	0,4%	1 LSB (0,994)
9	0,2%	1 LSB (0,997)
10	0,1%	1 LSB (0,998)
12	0,03%	1 LSB (0,999)
14	60,9 ppm	1 LSB (0,998)

TAB. 4.1 – Erreur de la méthode de calcul selon le nombre de bits utilisés

Les erreurs de calcul observables sont particulièrement raisonnables compte tenu du faible besoin en ressources numériques. Leur maximum se rencontre aux premiers instants, lorsque la dérivée est la plus élevée.

La constante de temps calculée ici dépend de la fréquence de fonctionnement du circuit et de la précision souhaitée (en nombre de bits). Étant données les valeurs rencontrées dans le monde biologique (plus élevées d'au moins deux ordres de grandeur), il existe deux techniques pour paralléliser ce calcul sur plusieurs grandeurs : l'usage de plusieurs unités de calcul, et le multiplexage temporel sur l'usage d'une unité de calcul. Vu le grand nombre de calculs à réaliser, la solution retenue utilisera très certainement une combinaison de ces deux techniques.

Définition des paramètres de la méthode de calcul

Nous venons de voir une méthode très intéressante pour réaliser le calcul d'évolutions en exponentielles décroissantes. Il nous reste cependant à s'assurer qu'il est possible de calculer celle qui nous intéresse en termes d'amplitude et de constante de temps.

Pour ce qui est de la constante de temps, il n'est pas possible de la réduire autrement qu'en augmentant la fréquence du circuit, ou en baissant la précision du calcul. Ce qui signifie qu'il existe, pour une configuration donnée, une constante de temps minimale. Il est par contre possible de l'augmenter indéfiniment en ne fonctionnant pas à chaque cycle, mais de façon alternative. Un circuit est alors nécessaire pour arbitrer les cycles de calcul et les cycles d'attente, de sa capacité maximale dépendra le taux d'évolution le plus lent. En ne fonctionnant qu'un cycle sur deux, trois ou même dix, nous obtiendrons des constantes de temps respectivement deux, trois et dix fois supérieures. Pour un diviseur de fréquence de n bits, τ pourra ainsi être multiplié jusqu'à 2^n fois. Ici aussi, l'usage d'un compteur en virgule fixe permet des coefficients rationnels. Ainsi, un coefficient codé sur 10 bits avec 3 bits en puissances négatives permettra un coefficient entre 1 et $2^7 = 128$ réglable par pas de $2^{-3} = 0,125$. En reprenant $n = 10$ et $F_0 = 1\,\text{MHz}$, cette configuration permet de régler τ de $1,024\,\text{ms}$ à $131,2\,\text{ms}$ par pas de $0,128\,\text{ms}$.

Pour régler l'amplitude, l'usage d'un multiplieur n'est pas judicieux. En effet, la linéarité de l'équation différentielle utilisée permet de calculer directement $A \cdot e^t$ au lieu de e^t. Il suffit pour cela d'utiliser $f(0) = A$ au lieu de $f(0) = 1$ comme condition initiale.

Cette propriété de linéarité permet également de réaliser très simplement la sommation de deux évènements, même s'ils ne sont pas simultanés. Il suffit pour cela d'ajouter une impulsion à la valeur en cours du registre, autrement dit, il faut effectuer l'opération $f \leftarrow f + A$ pour obtenir un échelon d'amplitude A.

Mise en œuvre

Ces techniques de calcul ont été exploitées pour effectuer la simulation du poids d'une synapse plastique. Ce travail a été effectué par Guillaume Raigné dans le cadre de son stage de deuxième année d'élève-ingénieur ENSEIRB. Le calcul d'un poids synaptique occupe 788 bascules et 1 307 LUTs, soit l'équivalanet de 21 074 portes logiques. Sur un *Spartan*IIe 600 (XC2S600-6), il peut fonctionner à 68 MHz. La structure de calcul peut donc être multiplexée temporellement pour calculer plusieurs centaines de connexions par bloc. Dans un tel cas, les registres sont régulièrement échangés avec une mémoire.

Cette réalisation démontre la faisabilité de l'implantation matérielle du calcul.

Un élargissement de la technique pour un modèle plus complexe

La technique peut être transposée à toute fonction définie par une équation différentielle, et si cette dernière est linéaire, peu de circuits seront nécessaires à sa mise en œuvre. deux restrictions doivent tout de même être posées : les fonctions divergentes ne sont pas compatibles avec les calculs en virgule fixe, et les phénomènes oscillatoires voient les erreurs de calcul s'additionner et font diverger le signal.

Nous avons cependant vu un modèle plus précis pour la STDP utilisant la fonction $g(t) = t \cdot e^{-t}$. Cette fonction peut s'exprimer comme le résultat de l'équation 4.12.

$$g'(t) = -g(t) + e^{-t} \tag{4.12}$$

Après un calcul analogue à celui qui aboutit à l'expression 4.7, nous retrouvons que la fonction $g(t)$ est résolue par l'équation 4.13.

$$g(t + \delta t) = g(t) - g(t) \cdot \delta t + e^{-t} \cdot \delta t \tag{4.13}$$

Bien que plus gourmand en ressources (deux registres de 2n bits, un additionneur n bits et deux additionneurs 2n bits), le calcul de cette fonction reste valable pour une implantation en temps réel et permet un modèle bien plus précis. En effet, il résoud le problème de discontinuité de la STDP lorsque les potentiels d'action présynaptiques et postsynaptiques sont simultanés.

La convergence des fonctions simulées ne pose pas de problème étant donné que la nature exponentielle garantit une attraction forte vers la valeur finale nulle. La complexification augmente l'erreur par rapport au modèle, cependant, le résultat est plus fidèle aux phénomènes modélisés, il s'agit donc quand même d'un gain en précision sur le plan de la simulation.

L'extension de la méthode permet de garantir la faisabilité (sous réserve toutefois de la puissance de calcul disponible) de toute expression de la forme $P(t) \cdot e^{-t}$ où $P(t)$ est un polynôme de degré fini. Ces expressions sont utiles pour faire apparaître des phénomènes plus complexes. Par exemple, la figure 4.3 nous montre une forme de *ltp* avec dépression au-delà d'un certain temps. Cette forme pourrait révéler de nouveaux phénomènes car elle est qualitativement différente. Elle s'obtient en utilisant un polynôme de degré 2 pour $P(t)$: $P(t) = 3 \cdot t - t^2$.

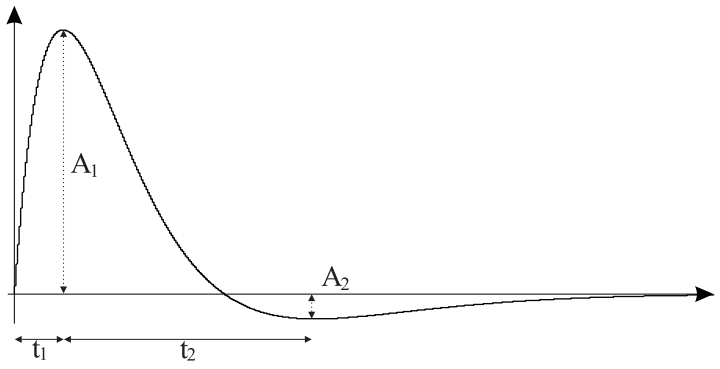

FIG. 4.3 – Une *ltp* plus complexe

Cette forme est calculée comme étant la solution g du système d'équations 4.14 :

$$\begin{cases} f'(t) = -f(t) & \text{et } f(0) = 1 \\ g'(t) = h(t) - g(t) & \text{et } g(0) = 0 \\ h'(t) = -2 \cdot f(t) - h(t) & \text{et } h(0) = 3 \end{cases} \tag{4.14}$$

Les relations différentielles étant linéaires, la solution pour les temps négatifs est $(f, g, h) = (0, 0, 0)$. L'amorçage de la fonction s'effectue donc en ajoutant le triplet $P_{ltp} \times (1, 0, 3)$ aux fonctions (f, g, h). Étant donné que la pondération P_{ltp} est un paramètre fixe de la plasticité, le triplet d'amorçage n'est calculé qu'en phase de configuration. Comme la multiplication par deux est en réalité un décalage d'un bit

vers les poids forts, la fonction de la figure 4.3 s'établit elle aussi par un jeu simple [1] d'additions/soustractions et de décalages de bits.

Pour choisir des relations liant à la fois t_1 à t_2 et A_1 à A_1, le recours à un polynôme de degré 3 est nécessaire. Dans ce cas de figure, les dérivations successives du terme en t^3 requièrent l'usage d'une multiplication par 6, cette opération ne nécessite qu'une addition supplémentaire associée aux décalages de bits *ad hoc*. Il s'agit cependant de l'apparition d'une réelle multiplication dans la méthode de résolution ; le fait qu'un terme soit constant limite néanmoins l'impact sur les performances.

4.2.3 Un mot sur la polyvalence

Nous venons d'établir des techniques théoriques qui permettent de repousser la limite que constitue la taille du réseau. Il faut cependant garder à l'esprit que ce gain s'effectue au prix d'une spécialisation poussée. Or, nous avons vu les intérêts d'un système entièrement souple.

L'ensemble des améliorations vues ici doit ainsi être considéré comme une aide facultative et donc contournable pour la gestion du réseau. En rétablissant un accès direct de l'ordinateur de contrôle/configuration vers les neurones, de nombreuses voies peuvent encore être explorées malgré une chute des performances.

Cette considération impose une double structure pour permettre un fonctionnement du même type que PAX.

4.3 La réalisation du système

Le simulateur de troisième génération est constitué par une unité indépendante configurée par un ordinateur qui garde une possibilité d'intervention en temps réel. Les éléments théoriques que nous venons de voir y sont implantés. La structure présentée ici a été conçue en concertation avec Colin Lopez qui s'est également occupé de la réalisation des cartes utilisées dans ce chapitre.

Bien que la structure de ce système soit très différente du premier, l'appellation PAX a été gardée pour clarifier la visibilité des travaux. Nous abordons ainsi l'élaboration de PAX3.

4.3.1 Structure générale

L'ensemble du système se loge dans un boitier de 3 unités de haut pour baies 19 pouces. Ce boitier permet de relier 21 cartes sur un fond de panier commun, lequel distribue alimentations et signaux. Parmi les 80 broches dédiées aux signaux, 10 sont réservées à la constitution de 5 structures chaînées reliant exclusivement chaque carte à la précédente ou la suivante. Le boitier est également disponible pour des hauteurs de 6 unités ou 9 unités. Ces versions permettent de réaliser deux ou trois fois la même structure en gardant une cohésion mécanique, et de les relier entre elles.

La figure 4.4 représente l'architecture du système. Sur les 21 cartes présentes figurent le maître du bus (carte *Thalamos*) et jusqu'à 20 sous-unités de calcul. Le maître prend en charge l'arbitrage des signaux du bus, la configuration de la structure à la mise sous tension et le relais des instructions provenant de l'ordinateur de

[1] Dans le contexte de ce calcul, la simplicité vient du faible nombre d'opérations et d'une fonction réalisable en une seule période d'horloge. Cette précision est nécessaire pour éliminer tout recours à une multiplication *cachée*.

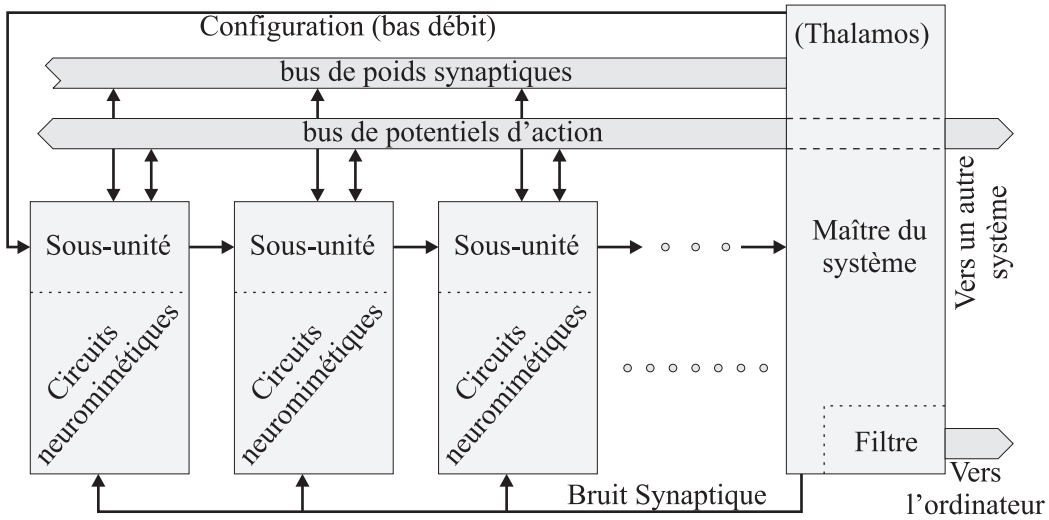

FIG. 4.4 – Structure logique du système de PAX3

contrôle. Les sous-unités de calcul peuvent contenir un nombre et des types de neurones différents entre elles ; elles sont également interchangeables. À la demande de *Thalamos*, chaque carte fournit ses statistiques, et reçoit ses identifiants pour interagir avec le reste du système (numéros attribués à la carte et aux neurones qu'elle contient). La sélection individuelle des cartes s'effectue à l'aide de la structure chaînée.

Une fois le système configuré, la simulation commence. Les signaux sont répartis en trois systèmes de communication :

- le bus de potentiels d'actions où chaque carte écrit les évènements générés par les neurones qu'elle contôle et lit l'activité de l'ensemble du réseau ;
- le bus de remontée d'informations permet de faire parvenir les valeurs de la matrice de connexion des sous-unités de calcul vers les étages supérieurs de la structure (maître, ordinateur) ;
- le bus de supervision où circulent l'ensemble des informations en provenance des couches supérieures à destination des cartes (par ordre de priorité décroissante : stimuli, bruit synaptique, paramètres réseaux, paramètres neuronaux).

Le bus de supervision permet également des accès bas niveau en écriture vers les neurones. L'ensemble fonctionne à 20 MHz et la limite maximale du nombre de neurones est fixée à 511, la transmission d'un potentiel d'action est alors garantie en 26 µs quelle que soit l'activité du réseau. Pour assurer un temps de propagation homogène, il est possible d'attendre que ce delai soit terminé avant de prendre en compte l'information reçue.

4.3.2 Répartition des capacités de calcul

Chaque sous-unité ayant accès à l'activité de l'ensemble du réseau, elle calcule les grandeurs dont elle a besoin pour éviter tout usage superflu de la structure de communication. Cela signifie qu'elle mémorise les poids de l'ensemble des connexions afférentes aux neurones qu'elle gère, et uniquement ceux-là. L'envoi des stimulations

synaptiques vers le circuit ne pose pas de problème puisqu'au sein d'une sous-unité, le composant gérant le calcul numérique est en contact direct avec les circuits.

La carte maîtresse, outre les fonctions déjà évoquées, est également chargée d'effectuer la liaison entre deux structures si le système utilise plus de 20 sous-unités de calcul (la structure doit alors se répartir sur deux systèmes physiques). Elle joue également un rôle majeur pour le calcul d'un modèle de plasticité alternatif. Elle a accès à l'ensemble de l'activité du réseau, et sa situation privilégiée lui permet de modifier l'état des connexions à la volée. Selon la volonté de l'utilisateur et les exigences du modèle, le calcul est mené, au choix, dans un processeur embarqué sur la carte (*powerpc*) ou par l'ordinateur de contrôle. Plus le calcul s'effectue à haut niveau, plus la puissance disponible sera importante, mais plus les temps d'accès seront grands.

4.3.3 Récupération des résultats

Contrairement aux systèmes précédents, des structures différentes sont envisagées pour le fonctionnement de la simulation et la récupération des résultats. Si le calcul embarqué apporte un atout considérable en terme de puissance disponible ou de temps de réaction, il supprime l'accessibilité des grandeurs simulées.

La carte maîtresse *Thalamos* étant connectée à l'ordinateur tout en ayant accès à l'activité du réseau, la récupération de cette dernière s'effectue sans grande difficulté. Le transfert est d'autant plus aisé qu'il n'est pas soumis au temps réel et peut donc faire intervenir des mémoires tampons intermédiaires (le pire cas requiert $150\,\mathrm{Hz} \times 511\,\text{neurones} = 76800$ potentiels d'action par seconde). Il en est autrement pour accéder aux poids synaptiques soumis à la plasticité. Ces grandeurs ne sont théoriquement pas nécessaires, puisqu'il est possible de les recréer depuis l'activité du réseau. Cependant, l'usage d'une telle technique ne permet pas de contrôler que le simulateur se comporte comme le prédisent les équations. Aucune politique n'est encore décidée pour la relecture des poids synaptiques. Il est possible d'effectuer un rafraîchissement permanent ou de faire remonter individuellement les poids modifiés pendant la simulation. La seconde solution permet une copie conforme en temps réel, mais pose des problèmes de gestion de l'écriture sur le bus.

Selon la nature de la simulation, l'ordinateur a besoin de relire l'ensemble de ces données ou juste une partie d'entre elles (pour une plasticité particulière par exemple). Le FPGA de la carte maîtresse permet alors de filtrer les données pour ne pas surcharger le canal de transmission, ni surtout l'occupation du processeur.

Finalement, la carte *Thalamos* possède de nombreuses extension de connexion (USB, RS232...). Ces extensions lui permettent de s'adresser à deux ordinateurs simultanément : l'un en temps réel pour la simulation, l'autre pour archiver les données.

4.3.4 Des éléments diversifiés

Le protocole de communication, permet de raccorder des sous-unités de types variés. La seule contrainte est d'embarquer un système de calcul numérique capable de gérer la plasticité pour les éléments locaux. Bien qu'il soit techniquement possible d'intégrer des plasticités différentes selon les sous-unités, il faut en vérifier la pertinence par rapport au modèle : une sous-unité ne gère la plasticité que pour les connexions ayant ses neurones locaux comme neurones postsynaptiques.

L'axe de recherche de FACETS étant basé sur la diversité dans les réseaux, la cohabitation de circuits différents est planifiée. La réalisation d'une carte d'interface vers

des neurones biologiques est également prévue. Dans un premier temps, un nouveau circuit neuromimétique a été réalisé pour former le cœur de calcul du système : le circuit *Galway*.

4.4 Le circuit Galway

À la différence des circuits *Trieste* et *Claudia*, le circuit *Galway* est conçu pour être flexible et configurable. Il s'appuie sur les travaux de thèse de Sylvain Saïghi [Saï04] qui ont abouti à la réalisation d'une bibliothèque de fonctions analogiques. Cette dernière a été validée par un premier circuit configurable : *Pamina* [Alv04].

La bibliothèque a été réalisée en technologie *Austria*micro*systems* SiGe $0,35\,\mu m$ et le circuit *Galway* occupe $10,5\,mm^2$ pour 50 000 composants (transistors, résistances et condensateurs).

4.4.1 Cahier des charges

La conception d'un système contenant un grand réseau nécessite d'intégrer un maximum de neurones par circuit. Sur un système complet comprenant 20 sous-unités, le nombre moyen de neurones par carte doit être de l'ordre de 25 pour atteindre la capacité maximale. Le format de la carte constituant chaque sous-unité étant limité par la structure mécanique qui les accueille, le nombre de circuits par sous-unité est lui aussi limité.

L'autre caractéristique recherchée est une polyvalence accrue par rapport aux différents modèles envisageables. Cela permet de raccourcir le temps de développement de chaque circuit. En effet, sur un circuit reconfigurable, il n'est pas nécessaire d'attendre que tous les paramètres des modèles de neurones soient définis pour effectuer la conception et la fabrication. En outre, une dérive importante du processus de fabrication peut être compensée au niveau de la configuration du circuit.

Le caractère configurable permet, s'il laisse suffisamment de degrés de liberté, un emploi des circuits dans d'autres applications que celles prévues à l'origine. Cependant, les accès aux grandeurs internes du circuit dans *Galway* sont prévus pour des applications expérimentales , et non pour une étude du modèle au niveau des canaux ioniques. Le circuit *Pamina* ayant été conçu en ce sens, il est par conséquent inutile de consacrer davantage de ressources à cette exploration.

4.4.2 La bibliothèque analogique

La bibliothèque utilisée ne constitue pas uniquement une migration technologique vers une finesse de gravure de $0,35\,\mu m$. Elle rassemble les opérations nécessaires à l'intégration d'un modèle qui suit le formalisme d'Hodgkin et Huxley. Contrairement à la bibliothèque utilisée pour le circuit *Trieste*, l'ensemble des fonctions est paramétrable. Cette solution permet de retrouver un modèle exploitable malgré les dérives qui surviennent au cours du processus de fabrication. Par contre, le réglage des quelques dizaines de paramètres pour chaque neurone devient une étape importante lors de la mise en œuvre du circuit.

Le circuit *Pamina* a servi de démonstrateur pour cette bibliothèque. Il simule deux neurones constitués de cinq conductances : fuite, sodium, potassium, calcium et potassium-dépendant-calcium. La conductance représentant les ions calcium peut également être utilisée pour représenter une conductance modulante, c'est-à-dire un

canal regroupant plusieurs ions amenant une modulation de l'activité du neurone. Dédié avant tout aux réseaux de petite taille, ce circuit intègre 8 monosynapses. Ce choix limite la taille du réseau maximal implantable, mais l'application visée est essentiellement sur la simulation paramétrée du soma.

Chacun de ces canaux peut, au choix, être rattaché au noyau de simulation. Chaque courant ionique est observable individuellement. Cette topographie du neurone est introduite dans le circuit par une interface numérique en série. L'ensemble des paramètres de fonctions est mémorisé de façon analogique dans des cellules mémoire. Le fonctionnement de ces cellules est basé sur le principe d'un échantillonneur/bloqueur. Les paramètres sont introduits et rafraîchis en permanence à l'aide d'une interface analogique série.

Le rapport entre les grandeurs simulées et celles utilisées par le circuit est de 5 pour les tensions, 50 pour les courants et donc 10 pour les conductances et capacités. Le circuit *Pamina* est fonctionnel, il ne sera cependant pas réutilisé. La conductance potassium-dépendant-calcium, dont il est pourvu, n'est pas utile car d'un ordre secondaire aux besoins en précision ; mais surtout, l'objectif en taille de réseau impose l'usage de multisysnapses. Ces dernières ont été développées spécifiquement pour *Galway* après une étude de migration technologique depuis la bibliothèque de *Trieste* effectuée par Katarina Pak dans le cadre de son stage de DEA en électronique [PAK05].

L'architecture du circuit *Galway* étant différente, le développement d'une nouvelle interface numérique a été nécessaire.

4.4.3 Des synapses mixtes ?

Comme il a été exprimé ci-dessus, il est nécessaire de réaliser des multisynapses. La partie centrale de cette fonction est la réalisation de $r(t)$ dans l'équation 1.7, c'est-à-dire un signal ayant une évolution en exponentielle décroissante. La question sur la nature de ce circuit se pose alors. Nous pouvons soit utiliser un montage comparable à celui utilisé dans *Trieste* [ALV03b], soit implanter un calcul numérique utilisant la technique présentée en 4.2.2.

Une proposition de gestion centralisée des synapses

Une architecture pour la gestion numérique des décroissances synaptiques est proposée en figure 4.5. Nous pouvons distinguer :

- une partie numérique pour le calcul en tant que tel, la réception des stimulations ou encore la gestion des étages suivants ;
- un convertisseur numérique/analogique en sortie du bloc numérique ;
- un ensemble de mémoires analogiques pour permettre l'usage d'une unique source analogique, quel que soit le nombre de synapses à gérer ;
- un ensemble de fonctions analogiques pour le reste des synapses

Le principe de cette structure est équivalent à l'ensemble des mémoires analogiques. Cependant, le nombre de sorties est moindre, ce qui permet l'usage d'une cellule mémoire moins robuste. Il faut également ajouter que la nature fluctuante des signaux de sortie impose un balayage plus rapide de l'ensemble, cependant, le caractère entièrement intégré de l'ensemble compense cette contrainte de vitesse sur le convertisseur.

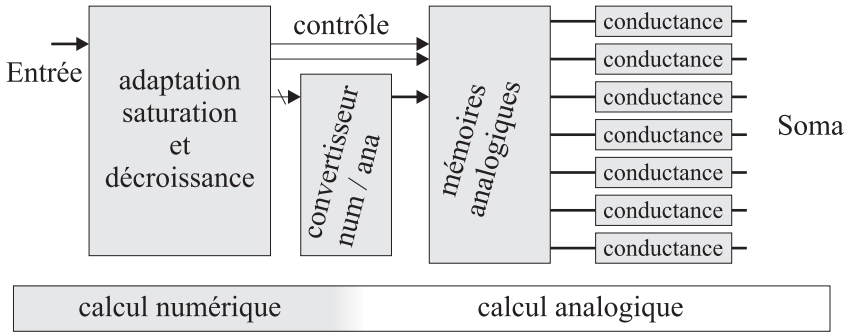

FIG. 4.5 – Une architecture pour un calcul numérique et centralisé des synapses

Analyse

Ce principe de synapse mixte possède de sérieux atouts :

- les paramètres définis numériquement ne subissent pas de dispersion et n'ont pas besoin d'être rafraîchis ;
- il est possible de proposer plusieurs modèles pour chaque multisynapse, voire pour chaque synapse ;
- une gestion centralisée des entrées et l'absence de condensateur externe diminue le nombre de broches nécessaires au circuit final ;
- il est possible d'intégrer sur silicium les fonctions de saturation synaptique, voire de plasticité.

Cette technique possède également d'autres caractéristiques, comme la nature du bruit amené sur le circuit. Une synapse entièrement analogique sera soumise à de nombreuses sources de bruit, fluctuant notamment en fonction de l'environnement du circuit. La synapse mixte laissera passer un bruit de nature numérique sur les courants synaptiques. Si les commutations peuvent être filtrées, il n'est pas possible de supprimer le bruit de quantification.

La mise au point d'une interface commune reste à effectuer pour ne pas retrouver les problèmes de débit et d'engorgement des interfaces rencontrés avec PAX2. Finalement, l'ensemble de la structure repose sur la qualité du convertisseur numérique/analogique. Ce dernier devant être soit acheté, soit développé.

Bien que prometteuse, une telle structure ne peut être déployée sans caractérisation préalable. Un convertisseur 8 bits a été développé, et implanté comme module séparé, pour évaluer les performances d'une synapse isolée en simulation. Il met en cascade deux convertisseurs 4 bits basés sur un réseau de résitances en pont diviseur comme le montre la figure 4.6. Le premier, pour les poids forts, délivre deux tensions, celles-ci correspondent aux potentiels minimal et maximal du second, pour les poids faibles.

Ce module occupe une surface de $225 \times 230\,\mu m^2$. Sa consommation est de $20\,\mu A$ pour une plage de sortie de $1\,V$ et son impédance de sortie atteint environ $500\,k\Omega$. Par simulation, le temps de conversion est de $300\,ns$ sur un condensateur de $100\,fF$.

Un dernier point à souligner est que, malgré l'usage de circuits numériques, cette architecture occupe plus de surface que l'équivalent entièrement analogique. Il faudra donc attendre une finesse de gravure supérieure pour que l'intégration de fonctions par des méthodes numériques minimise le coût de fabrication.

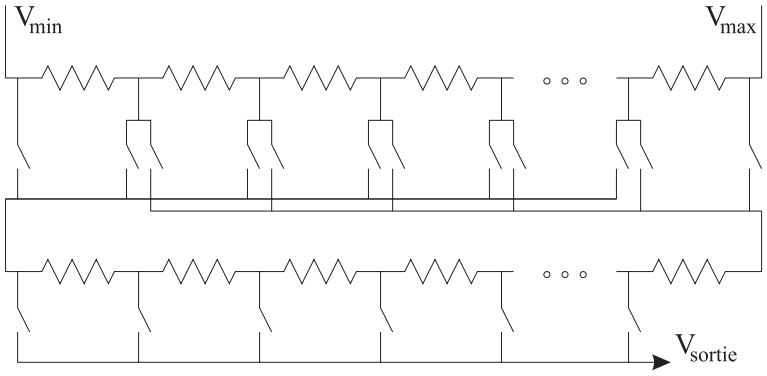

FIG. 4.6 – Structure du convertiseur numérique/analogique

4.4.4 Contenu du circuit *Galway*

La structure du circuit *Galway* est présentée figure 4.7. Il contient 5 neurones artificiels. Le premier neurone contient 3 conductances (sodium, potasssium et fuite), il peut simuler un neurone inhibiteur (FS). Les suivants ont une quatrième conductance pour l'ajout d'un courant calcium ou modulant. Le cinquième neurone a sa quatrième conductance doublée pour avoir à la fois un courant calcium et un courant modulant. Pour vérifier un comportement du circuit sous forme de *matrice de conductances programmable*, une conductance flottante a été ajoutée. Il s'agit d'une conductance de type modulante. Elle peut être ajoutée, au choix, au dernier neurone, pour obtenir un soma constitué de six conductances, ou à l'avant-dernier, pour obtenir deux neurones ayant cinq conductances chacun.

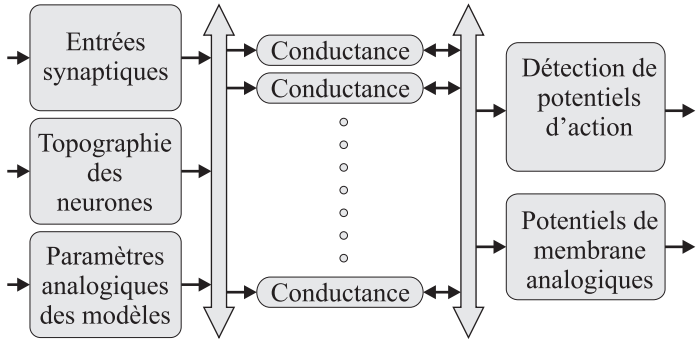

FIG. 4.7 – Structure du circuit *Galway*

Chaque neurone possède également trois multisynapses. Il est ainsi possible de modéliser trois types différents de connexions. Il est également possible de fonctionner selon le schéma à deux types (excitatrice et inhibitrice) auquel s'ajoute une entrée spécifique pour les stimuli.

Une sortie en courant est dédiée à la visualisation des courants ioniques. Cette visualisation permet une caractérisation des modèles réellement simulés ainsi qu'un réglage plus fin de chaque paramètre analogique. Le choix des courants envoyés vers cette sortie s'effectue par l'intermédiaire des paramètres numériques du circuit.

Finalement, le convertisseur numérique/analogique 8 bits est intégré pour caractérisation.

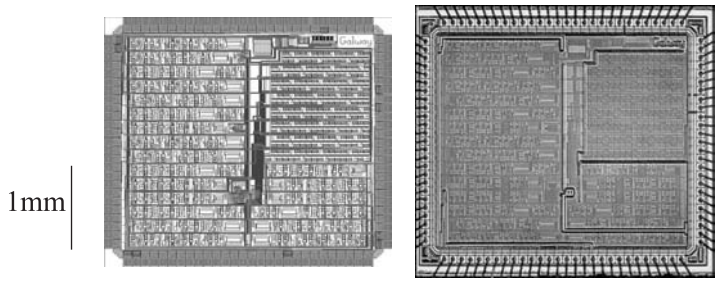

FIG. 4.8 – Routage final (gauche) et photographie (droite) du circuit *Galway*

L'ensemble nécessite 96 paramètres numériques d'un bit répartis sur 16 mots de commande, ainsi que 204 paramètres analogiques. Étant donné que ce circuit sollicite les mémoires analogiques au-delà de leurs performances nominales, une mémoire supplémentaire a été ajoutée ; après isolation, la valeur mémorisée est envoyée à l'extérieur du circuit pour être caractérisée. Le routage final (à gauche) et une photographie (à droite) du circuit sont présentés figure 4.8. Ce composant nécessite 105 broches, dont 47 vers des composants passifs et 16 pour les alimentations.

Interfaces

Nous allons étudier ici les interfaces qui permettent l'usage de *Galway*. Seule la méthode d'introduction des paramètres est présentée, la correspondance entre les paramètres et le modèle est décrite en Annexe E.

Paramètres numériques L'introduction des paramètres numériques s'effectue par mots de 14 bits combinant identifiant et valeur. La transmission se fait en série par une interface de 3 signaux parallèles : horloge, données et validation. La validation s'effectue pendant le dernier bit du mot, ce qui évite l'usage d'un séquenceur. Le diagramme des signaux est donné en figure 4.9.

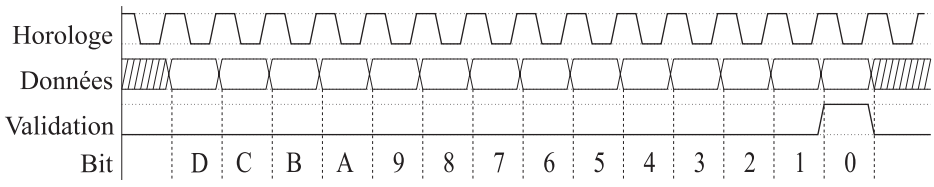

FIG. 4.9 – Chronogramme de l'entrée série des paramètres numériques de *Galway*

Bien que la bibliothèque numérique utilisée puisse fonctionner à plusieurs dizaines de mégahertz, l'interface n'a été simulée qu'à trois fréquences : 100 kHz, 1 MHz et 10 MHz. L'envoi de signaux ne s'effectue que pendant l'étape de configuration, le débit a donc peu d'importance.

Paramètres analogiques L'interface d'entrée des paramètres analogiques requiert quatre signaux, dont trois numériques : horloge, initialisation du rafraîchissement et chargement de la valeur (Validation). Le quatrième signal, analogique, prend les valeurs successives des paramètres du circuit. Les différents signaux sont représentés sur le chronogramme de la figure 4.10.

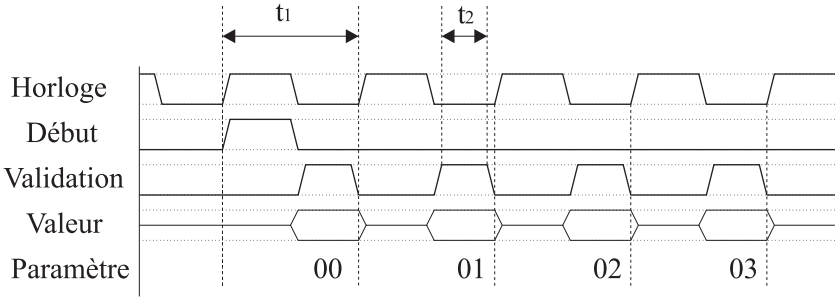

FIG. 4.10 – Chronogramme de l'entrée série des paramètres analogiques de *Galway*

Initialement conçu pour fonctionner à 100 kHz, cette entrée est modulable sous certaines conditions.

- Bien que le front descendant de l'horloge ne soit pas utilisé, il est préférable de conserver un rapport cyclique de 50 %. La fréquence nominale de fonctionnement est de 100 kHz ($t_1 = 10$ ms).
- Le signal d'initialisation est asynchrone, il doit cependant être activé au moins 100 ns avant le chargement.
- La valeur à charger doit être présentée sur une durée minimale t_2 de 3 µs

Les cellules mémoires ont été conçues pour maintenir les valeurs à 1 mV près pendant 2 ms ce qui autorise la gestion de 200 paramètres à 100 kHz. Cependant, les performances obtenues en simulation vont au-delà et nous pourrons effectuer une estimation du nombre maximal de paramètres avec l'étude des mesures obtenues sur le circuit.

Synapses Les entrées synaptiques fonctionnent selon le même principe que les multisynapses utilisées dans les systèmes des deux premières générations. Le poids synaptique est toujours codé dans la durée de l'impulsion de déclenchement ; cependant, la montée a été accélérée pour réduire les décalages que cette technique génère : elle dure au maximum 60 µs. Au-delà de ce temps, des effets de saturation peuvent apparaître. Cette durée permet une précision de 9 bits pour un circuit de contrôle fonctionnant à 10 MHz. Il est possible de monter à 12 bits de précision pour un FPGA fonctionnant à 70 MHz.

Si une telle précision du poids synaptique n'est pas nécessaire, il est possible soit d'utiliser de très faibles temps de montée, soit de configurer la synapse avec une faible contribution pour utiliser un circuit de contrôle à faible fréquence. A titre de comparaison, PAX2 fonctionne à 32 MHz et utilise des poids codés sur 8 bits, ce qui permet un temps de montée inférieur à 10 µs.

4.4.5 Calibrage automatique

Pour mettre en œuvre un grand nombre de circuits, il est nécessaire de déterminer un protocole de réglage automatisé des paramètres. L'usage des valeurs nominales aboutit à des comportements erronés relatifs aux diverses erreurs de recopies de courant ou de traitement au sein du circuit intégré. Il faut alors une méthode qui détermine des valeurs corrigées pour compenser ces dérives.

Ce réglage s'effectue canal par canal, en observant le courant généré par une tension de membrane imposée.

Le cas de la conductance de fuite

Le réglage de la conductance de fuite est le plus simple puisque, seules, deux grandeurs sont à régler : le potentiel de repos et la conductance. Les valeurs sont alors déterminées par le tracé de la caractéristique $I_{fuite} = f(V_{mem})$. Une méthode systématique peut résoudre le réglage en deux étapes :

- pour un potentiel de membrane imposé égal au potentiel de repos et une conductance arbitrairement faible, l'algorithme détermine la valeur du paramètre correspondant au potentiel de repos qui annule le courant de sortie (si $g_{fuite} \neq 0$, $I = 0 \Rightarrow V_{mem} = V_{equi}$).
- pour un potentiel de membrane différent, il possible de régler la conductance donnée par l'équation même du canal : $g_{max} = I_{fuite}/(V_{mem} - V_{equi})$

Avec un aspect temporel : la synapse

Pour les synapses, la technique est similaire : le réglage s'effectue pour un poids synaptique donné, la valeur de $\bar{g}_{syn}$ est déterminée par le maximum de courant observable.

La détermination de la constante de temps de $r(t)$ s'effectue ensuite, par exemple, par la recherche d'une valeur égale à g_{syn}/e au bout du temps τ. Cette mesure peut également s'effectuer en réglant le maximum en valeur absolue de la dérivée de I_{syn}, lui aussi égal à τ.

Le cas général

Pour le réglage de conductances plus complexes, il reste possible de scinder les équations à déterminer. Pour les canaux d'ions sodium, par exemple, le potentiel de repos se mesure par la technique décrite précédemment. Il est ensuite possible d'annuler l'effet de l'inactivation en réglant temporairement V_{offset_h} très élevé et V_{pente_h} très faible. De même pour l'activation avec V_{offset_m} fortement négatif. Cette situation permet la détermination de la conductance.

Finalement, les paramètres se déduisent un à un, par exemple, $V_{mem} = V_{offset}$ si et seulement si le courant a été divisé par 2 pour l'inactivation, ou par 8 pour l'activation. V_{pente} devient ensuite la seule inconnue dans la valeur du courant en régime permanent. Lorsque le régime permanent est réglé, il ne reste alors plus qu'à déterminer là valeur de τ.

Les paramètres, évalués séparément pour l'activation et l'inactivation, peuvent ensuite être configurés simultanément pour aboutir à un canal fonctionnel. Bien que laborieuse, cette méthode a l'avantage d'être transposable en algorithmes, ce qui

permet le développement d'une plateforme automatisée de calibration pour le circuit. Le dernier problème à résoudre est une dépendance de V_{offset} par rapport à V_{pente} observable avec la bibliothèque analogique et qui n'apparaît pas dans le modèle. La solution doit alors être obtenue par une suite d'itérations.

4.5 Premiers résultats sur le circuit *Galway*

La plateforme de test du circuit a été réalisée par Colin Lopez. Elle interface un ASIC *Galway* avec un FPGA contrôlé par ordinateur. Ce banc de test a été configuré par Hicham Lahbil dans le cadre d'un stage de deuxième année d'élève-ingénieur ENSEIRB.

4.5.1 Oscillation des neurones

La première fonction testée sur le circuit concerne la capacité des neurones à osciller. Les paramètres des modèles envoyés au circuit sont ceux définis lors des simulations logicielles. Comme dans le cas du circuit *Trieste*, un réglage (grossier) a été effectué en jouant sur le courant de stimulation. Nous obtenons alors les courbes données en figure 4.11 qui représentent l'activité observée sur les potentiels de membrane.

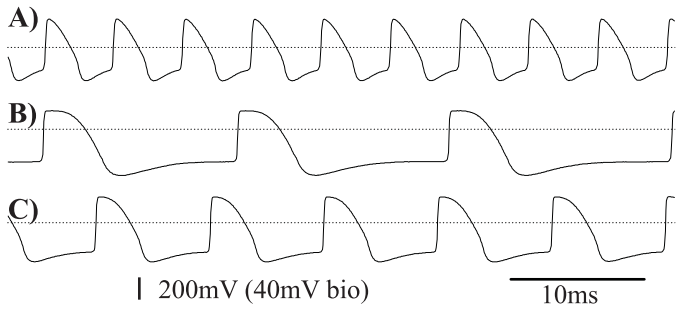

FIG. 4.11 – Activités observables sans réglage fin sur le circuit *Galway*. Les potentiels de membrane représentés sont ceux des neurones 1 (courbe A), 2 (courbe B) et 4 (courbe C).

Bien que les formes observables rappellent celles des potentiels d'action, elles restent largement perfectibles. Ce travail de réglage fin est obligatoire pour un usage du circuit dans de bonnes conditions. Il n'est pas encore réalisé à l'heure où sont écrites ces lignes. Cependant, la fonctionnalité complète du circuit a été démontrée puisqu'il a été possible d'obtenir, en même temps, des oscillations sur chacun des 5 neurones du circuit testé.

Test fonctionnel de l'ensemble de la série

La série de tests effectuée ici ne cherche pas à reproduire un comportement oscillatoire de neurone, très gourmande en temps de manipulation. Elle vise à vérifier si chaque circuit produit un comportement cohérent à celui attendu pour permettre son réglage. Pour chaque circuit, 86 tests sont effectués. Ils vérifient, entre autres, l'apti-

tude de chaque conductance à agir sur le potentiel de membrane et à être visualisée. les résultats sont présentés dena le tableau 4.2.

Echelle de description	Total	Fonctionnel	%
Test isolé	2150	2135	99,3
Conductance	1050	1040	99,0
Neurone	125	118	94,4
Circuits - partiellement	25	25	100
Circuits - totalement		16	64

TAB. 4.2 – Rendement observé sur le circuit *Galway*

Un circuit est considéré partiellement fonctionnel lorsqu'au moins un test s'est avéré négatif. Dix défauts ont été constatés sur l'ensemble de la série : 7 sur les neurones et 3 sur la conductance libre. À ce niveau de test, seuls quatre d'entre eux rendent un neurone inutilisable selon le formalisme de Hodgkin et Huxley. Ces neurones peuvent être utilisés selon le modèle *Integrate and fire* avec fuite, mais ne sont pas considérés fonctionnels car cette approche ne fait pas partie des problématiques de l'équipe.

Les chiffres affichés ne considèrent qu'un aspect du comportement attendu du circuit, et non la capacité des conductances à agir selon le modèle. Les tests sur le comportement analogique temporel ne sont pas encore effectués.

La multisynapse

Pour mettre en évidence le caractère programmable du circuit, nous utiliserons la fonction qui simule la multisynapse. Il s'agit d'un nouvel élément de la bibliothèque analogique, les mesures de sa caractérisation serviront d'illustration. La mesure des courants ioniques s'effectue par une broche dédiée du circuit. Les acquisitions sont réalisées aux bornes d'une résistance de 100 kΩ

Les premières mesures mettent en évidence la différence de notion entre la force du canal synaptique et le poids synaptique des stimulations transmises. Bien que ces deux grandeurs soient paramétrables, leurs applications sont différentes : le poids synaptique fluctue sans cesse d'une stimulation à l'autre, tandis que la force du canal associé est définie pour la durée de la simulation.

Les mesures suivantes sont effectuées dans un mode de fonctionnement ou le potentiel de membrane est fixé arbitrairement. Cette technique, dite de *voltage-clamp*, permet de supprimer les variations du courant dues aux fluctuations du potentiel de membrane dont chaque canal ionique dépend.

La figure 4.12 nous montre que la pente constante, à la montée, permet de lier la durée et l'amplitude de la stimulation. Le poids synaptique est donc associé à chaque stimulation. Dans les conditions de test utilisées ici, le FPGA qui gère le circuit est cadencé à 64 MHz, la durée maximale de montée permet alors une définition de 12 bits pour le codage des poids synaptiques. Seules les valeurs les plus élevées risquent de faire apparaître une saturation, puisqu'elles excèdent le temps de 60 µs au delà duquel un comportement correct n'est plus garanti. Le circuit ayant servi aux mesures permet visiblement des stimulations plus longues sans déformation apparente.

La figure 4.13 représente le même courant synaptique par rapport à la force

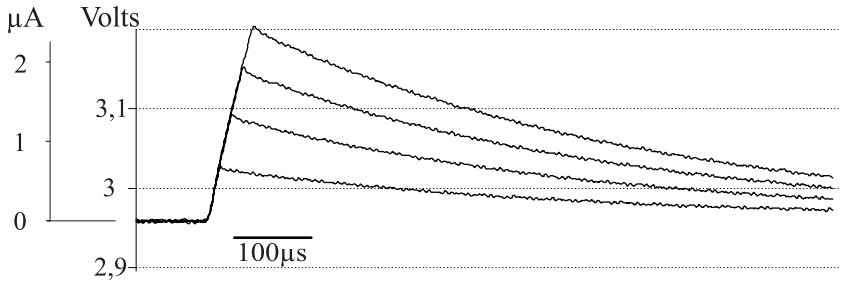

FIG. 4.12 – Courants synaptiques issus de stimulations de différents poids.

du canal $\bar{g}_{syn}$ et pour un poids constant. Dans ce cas, nous observons une réelle modulation du courant synaptique puisque la pente de montée varie elle aussi. À la différence de la mesure précédente, le paramètre exploré ($\bar{g}_{syn}$) ne dépend pas de la stimulation mais de la multisynapse. Il ne peut donc pas servir à exprimer une forme quelconque d'adaptation.

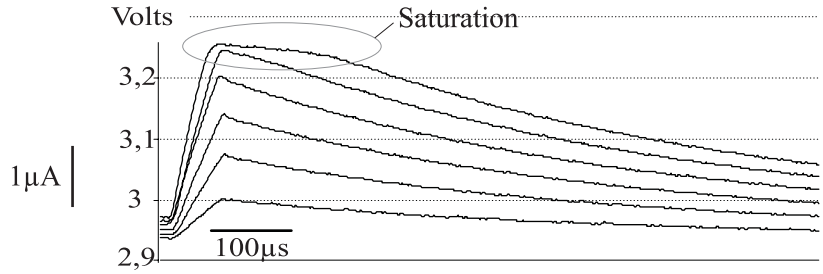

FIG. 4.13 – Courants synaptiques générés pour une stimulation donnée avec différentes forces de canal

La figure 4.14 montre que le réglage de la cinétique de décroissance fonctionne correctement. Les comportements les plus lents ne sont pas représentés car la courbure de la décroissance n'est pas observable à cette échelle. Il est cependant possible de monter à des constantes de temps de plusieurs centaines de millisecondes.

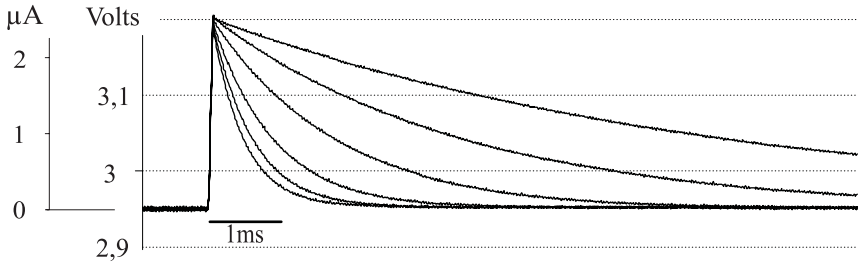

FIG. 4.14 – Différents comportements avec variation de la cinétique de descente

Une validation de la sommation des stimulations est présentée en figure 4.15. Elle représente le courant synaptique (B) obtenu en réponse à trois stimulations (A) de forces différentes. Pour cette figure, la constante de temps a été réglée sur une valeur

très lente (supérieure à 400 ms).

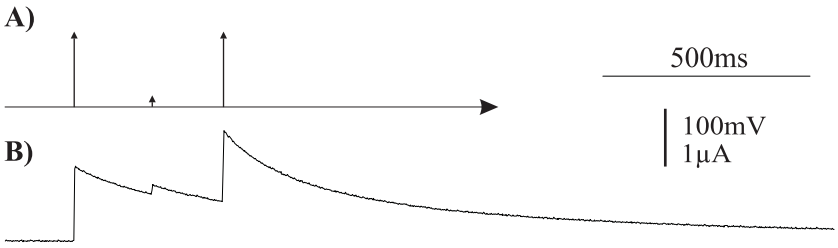

FIG. 4.15 – L'addition de stimulations synaptiques

La dernière figure (fig. 4.16) présente finalement l'association de la synapse analogique avec des unités numériques de prétraitement. C'est à dire que nous pouvons observer le résultat d'un système numérique de saturation synaptique placé entre la réception des stimulations et leur envoi effectif sur les composants analogiques.

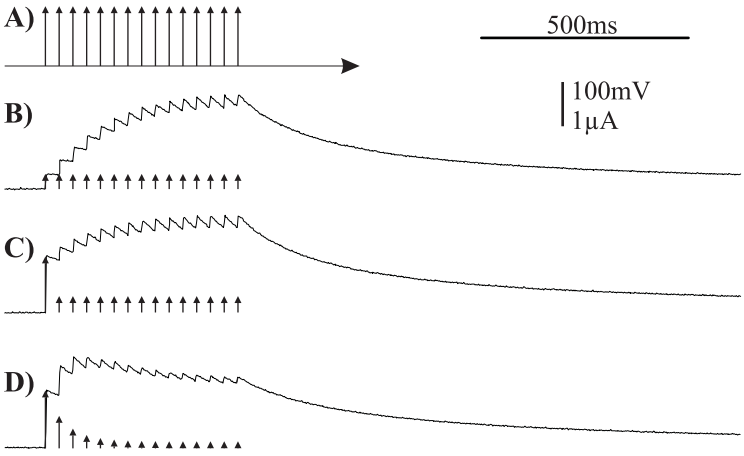

FIG. 4.16 – Comparaison de différents modèles de saturation synaptique

La stimulation correspond à un train de potentiels d'action présentés en (A). Le poids des stimulations avant saturation a été modifié dans chaque cas pour obtenir des courants comparables, il reste cependant constant pendant chaque série. il est observable sur la stimulation initiale de chaque train d'impulsions (synapse non saturée). Pour chaque série, la suite de stimulations est affichée avec le courant qui en découle en sortie de synapse analogique. La première mesure (B) présente la réponse sans phénomène de saturation synaptique : la série de stimulations correspondante conserve un poids constant. Le ralentissement observable ne provient pas d'une saturation mais d'un équilibre entre la décroissance continue et les échelons d'excitation. La mesure (C) traduit un effet de régénération de la synapse après une transmission. La première stimulation est forte, les suivantes plus faibles et dépendantes du temps entre les potentiels d'action. Finalement, la mesure (D) présente le phénomène de régénération des vésicules synaptiques, associé à une probabilité de déclenchement. Dans ce dernier cas, la population de vésicules disponibles lors d'une stimulation est constituée de celles qui se sont regénérées et de celles qui ne se sont pas déclenchées à la stimulation précédente. Cette deuxième catégorie est d'autant plus faible que le

nombre de stimulations passées est élevé.

Les constantes de temps des phénomènes de recouvrement synaptique ne reflètent pas les modèles biologiques, elles ont été choisies pour produire des graphes didactiques. Ces phénomènes sont très intéressants, mais également très gourmands en calcul au sein d'un système contenant un grand nombre de neurones. Ils sont associés à une connexion et leur nombre dépend donc du carré du nombre de neurones mis en jeu. Leur intégration dans le système final n'est pas à l'étude pour l'instant. Elle sera proposée s'il reste de la place dans les FPGAs de chaque sous-unité.

4.5.2 La cellule mémoire

Le changement des paramètres du noyau de calcul nous est permis par l'usage des cellules mémoires. Lors de la présentation de la structure du circuit, nous avons pourtant vu qu'il était nécessaire de les utiliser au-delà de leurs spécifications (205 paramètres au lieu de 200). Une caractérisation s'impose alors pour déterminer leur comportement en situation réelle. Cela nous permettra de déterminer le nombre maximal de cellules exploitables sur un seul circuit.

Tout d'abord, les valeurs mémorisées dans les cellules subissent une erreur de recopie de 3 mV, ce qui signifie qu'au delà de l'incertitude sur la valeur, un bruit est généré à l'intérieur même du circuit. La différence d'échelle de temps entre le rafraîchissement des paramètres et les phénomènes simulés suppriment cependant la répercussion de cette erreur sur le comportement du neurone.

La mesure de la cellule de test révèle alors une perte d'information négligeable. En effet, en adaptant le balayage pour observer une tenue de 10 ms, la perte d'information reste inférieure au milliVolt. Un circuit utilisant ces cellules est donc capable de gérer un grand nombre de paramètres.

Afin de limiter l'influence de l'incertitude de mémorisation, un essai a été réalisé pour augmenter la fréquence de rafraîchissement. Des mesures sur le comportement de la cellule, lorsque sa valeur est modifiée, montrent un temps de mémorisation inférieur à la microseconde. Ce temps d'ouverture a été testé et montre effectivement que la valeur mémorisée est suffisament précise après un seul cycle de chargement. Le circuit étant associé à un convertisseur numérique/analogique capable de réaliser la conversion en 1 µs, nous pouvons élever la fréquence de rafraîchissement des paramètres jusqu'à 350 kHz avec chargement en un cycle :

- 0,7 µs pour le chargement du convertisseur (interface série à 20 MHz)
- 1 µs pour l'établissement de la valeur analogique
- 1 µs pour la recopie du paramètre (temps d'ouverture de la cellule mémoire)

La période totale de cette trame est de 2,7 ms ce qui signifie une fréquence de 370 MHz pour le rafraîchissement, ainsi qu'une tenue de 553,5 µs pour les paramètres avec une perte négligeable. Il est également possible d'augmenter encore la fréquence en tolérant deux ou trois cycles pour le chargement ; cependant, il n'est pas intéressant d'aller au-delà de 400 kHz en ne jouant que sur le temps d'ouverture. L'usage d'un convertisseur ayant une interface parallèle supprime le temps de chargement et permet de monter à 500 kHz, mais nécessite beaucoup de broches d'interface. Bien que le chargement du convertisseur n'affecte pas sa valeur de sortie, il n'est pas mené pendant l'ouverture de la cellule mémoire à cause du bruit généré par l'électronique numérique.

4.5.3 Le convertisseur numérique/analogique

L'usage d'un convertisseur interne, associé à une mémorisation intégrée, permet de simplifier la mise en œuvre du circuit et réduit les besoins en composants externes. Bien que le convertisseur n'ait pas été developpé dans cette optique, sa caractérisation est liée à un développement pour cet usage.

Montage de test

Prévu pour un fonctionnement interne au circuit, le convertisseur n'est pas pourvu d'étage de sortie. Malgré une forte résistance de sortie, une charge capacitive de 2 pF n'est pas un réel problème. Il en est autrement pour un usage externe : la broche de sortie génère 30 pF de charge à laquelle il faut ajouter l'impédance d'entrée de l'appareil de mesure. Un montage suiveur a alors été placé en sortie pour isoler le convertisseur du système de mesure. Ce système permet au convertisseur de fonctionner normalement, mais les transitions resteront lentes à cause de l'effet capacitif des entrées/sorties des circuits.

La plage de fonctionnement est fixée entre 0 V (imposé par le circuit) et 2, 56 V (source externe). Le parallèle est donc effectué rapidement entre la valeur théorique de la sortie et celle observée.

Conformité des valeurs restituées

La sortie du convertisseur a été étudiée sur deux circuits pour pouvoir discerner le comportement propre au circuit de la dispersion des composants. La figure 4.17 représente la non-linéarité intégrale, c'est-à-dire, l'erreur en LSB observée entre les tensions attendue et mesurée.

FIG. 4.17 – Non-linéarité intégrale du convertisseur

Bien que l'on puisse observer une grande erreur pour les faibles valeurs (3 LSB), ces courbes révèlent un comportement linéaire. Étant donné qu'il n'a pas été possible de descendre en dessous de 30 mV, nous considèrerons qu'il s'agit de la tension minimale du convertisseur et que cette erreur provient du *design* du circuit lui-même. La non-linéarité différentielle qui en découle est présentée en figure 4.18.

Cette fois nous observons une faible différence en sortie par rapport aux valeurs attendues. L'erreur maximale mesurée sur ces deux circuits est de 0,35 LSB. La pré-

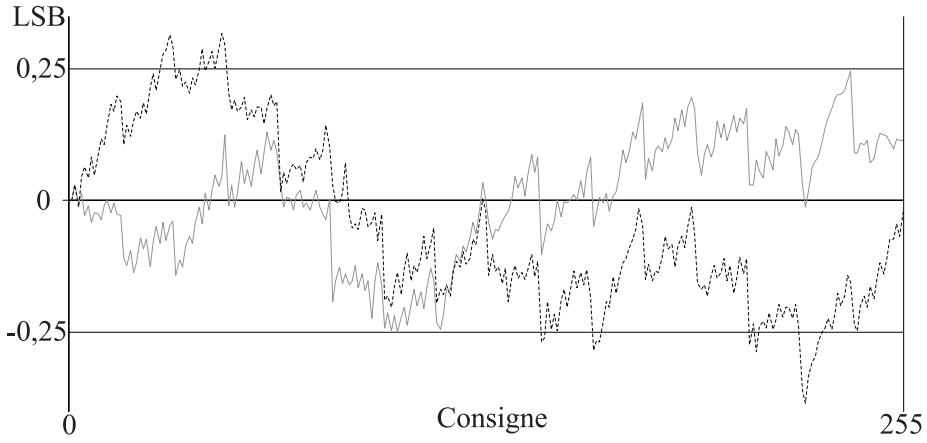

FIG. 4.18 – Non-linéarité différentielle du convertisseur

cision de cette valeur est plus grande que nécessaire. Une prochaine évolution du composant pourrait voir ses dimensions réduites en tolérant une légère dégradation de la précision.

La tolérance utile est d'autant plus grande que pour gérer les paramètres, seule la monotonie est réellement nécessaire. En effet, pour les algorithmes de recherche numérique de solution, utilisés ici pour régler le circuit, la linéarité de la conversion n'est pas requise. Voyons cependant la linéarité du convertisseur présentée figure 4.19.

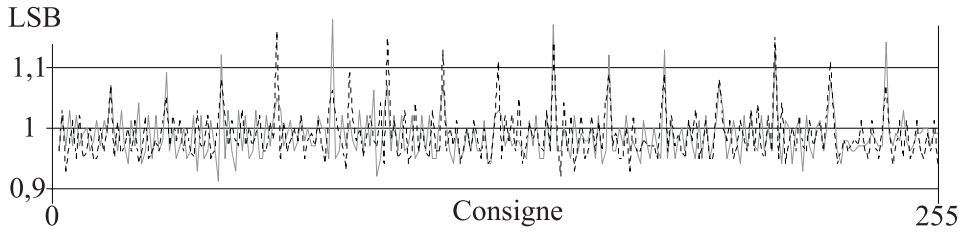

FIG. 4.19 – Différences entre les valeurs successives des sorties du convertisseur

Ces courbes nous montrent une très bonne linéarité puisque la différence est toujours proche de 1 LSB. Cette linéarité est très intéressante pour l'usage premier du convertisseur, à savoir, la génération de la forme des courants synaptiques. Par contre, pour un usage au niveau des paramètres analogiques, il aurait suffi de ne pas avoir de valeur négative ou inférieure à quelques dixièmes de LSB.

Temps d'établissement

Le temps d'établissement du convertisseur est de 250 ms. Cette lenteur, comme nous l'avons déjà évoqué, vient de l'absence d'étage de sortie pour un signal prélevé à l'extérieur du circuit. La charge imposée sur ce signal est une capacité d'environ 45 pF (30 pF + 15 pF pour le suiveur), alors que l'entrée des cellules mémoires est de 2 pF. Il faut donc s'attendre à un temps de conversion de l'ordre 11 µs, ce qui reste trop long. Bien qu'il soit possible de fonctionner à une fréquence de rafraîchissement de 50 kHz, cette solution n'est pas intéressante dans une optique de réduction du

bruit.

Pour améliorer le temps de conversion, deux solutions se présentent : l'usage d'un étage de sortie, ou une révision des caractéristiques des composants impliqués dans la gestion des paramètres. Le principal écueil de la conception d'un étage de sortie est l'étendue des valeurs des paramètres : entre 1 V et le rail d'alimentation 5 V. Il reste aussi possible de revoir le dimensionnement de l'ensemble du dispositif. En effet, la bonne tenue des cellules mémoires autorise une réduction des condensateurs qui conservent les valeurs ; en ce qui concerne le convertisseur, une réduction des longueurs de résistances aura le double effet de réduire la surface occupée et l'impédance de sortie. Les contreparties seront une augmentation de l'erreur et de la consommation, mais nous possédons une certaine marge de manœuvre puisqu'une précision de 0,35 LSB est inutile et que la consommation est de 100 µA pour les 5 V du circuit.

Par rapport aux caractéristiques de conceptions, nous retrouvons un temps d'établissement de l'ordre de 500 ns sur 100 fF. Cette grandeur montre que le réalisation est en deça des prévisions théoriques, mais reste intéressante.

4.5.4 Mesures planifiées

Les mesures réalisées jusqu'ici sont essentiellement destinées à démontrer le bon fonctionnement du circuit. Un exemplaire a été étudié en détail pour fournir l'ensemble des figures qui précèdent, deux autres ont été utilisés au titre de comparaison. Aucun de ces trois exemplaires n'a encore été pris en défaut.

Les mesures restantes concernent, dans un premier temps, une caractérisation détaillée des 25 exemplaires reçus, avec notamment une évaluation de la dispersion observable parmi les paramètres. Au-delà du circuit *Galway*, ces caractérisations seront recoupées avec celles du circuit *Pamina* pour fiabiliser la connaissance de la bibliothèque analogique.

Un réglage fin de chaque circuit, sur son emplacement d'accueil, sera ensuite mené pour que le système final puisse être utilisé en condition de simulation. Pour chaque neurone et pour chaque modèle désiré sur ce neurone, un jeu de paramètres optimaux sera établi. Ce travail est considérable, d'où la nécessité de l'automatiser au mieux. Si les simulations conduites sur PAX3 tendent vers une modulation des modèles selon l'activité du réseau, il sera en outre nécessaire de caractériser plus finement les valeurs de chaque paramètre en fonction de la tension de consigne. Seule une méthode automatisée permettra d'achever une telle quantité de travail.

4.6 Bilan

4.6.1 Résumé

Nous avons abordé dans ce chapitre les bases de PAX3. La structure repose sur un bus commun dédié à la circulation des potentiels d'action. L'absence d'échange de données entre les unités de calcul sera compensée par un calcul redondant. Un algorithme de STDP sera intégré au plus près des circuits, dans le FPGA qui les contrôle. Cette plasticité sera calculée en temps réel en utilisant des techniques de virgule fixe associées à une stratégie de calcul permanent mais peu gourmand en ressources.

Le premier circuit destiné à cette plateforme est pleinement fonctionnel. Il contient 5 neurones de complexité variable et, est entièrement paramétrable. Il contient également des éléments en développement, susceptibles, à terme, de le doter d'une interface

numérique unique et d'éléments d'adaptation pour faciliter son usage en réseaux de neurones de grande taille.

4.6.2 Une évolutivité sous conditions

La conception d'un système simulant un grand nombre de neurones nécessite beaucoup de remises en question et d'adaptation des modèles. En travaillant sur la définition des pires cas, par exemple, il est possible d'étendre l'architecture actuelle pour atteindre un potentiel dépassant le millier de neurones.

Premièrement, la vision d'un réseau au sein duquel chaque neurone est directement connecté à tous les autres n'est pas réaliste. Le système de troisième génération rassemblera jusqu'à 500 neurones, et, à ce stade, seul un pourcentage limité des connexions est réellement exploité. Bien qu'il soit impératif d'offrir la possiblité de configurer toutes les connexions, il n'est plus nécessaire de les activer en même temps. La puissance de calcul requise est alors réduite, ou, préférablement, la limite de la taille du réseau est repoussée.

De même, nous avons défini le temps minimal de transmission d'un potentiel d'action par rapport à une activité simultanée (de l'ordre de la microseconde) de l'ensemble des neurones. Ce cas de figure est improbable, nous pouvons alors nous contenter d'un pire cas probable, moins exigeant sur les contraintes mais pourtant suffisant pour toujours garantir une excellente fluidité de communication. En se basant sur la contrainte (très forte) de 50% des neurones qui puissent s'activer pendant un intervalle de 100 µs, l'architecture actuelle permet de garantir une gestion d'un réseau de 3 000 neurones.

Une autre piste à explorer est au niveau de l'aspect temporel des transmissions synaptiques. En effet, les systèmes matériels en temps réel cherchent à produire des connexions synaptiques dites *instantanées*, c'est-à-dire, des connexions dont le temps de propagation est négligeable à l'échelle du neurone. Cette vision provient des simulateurs logiciels pour qui le temps simulé n'est pas une contrainte. Les contraintes en communications allant de pair avec la taille des réseaux simulés, les délais de traitement deviennent observables si l'on traite de grands réseaux sur des simulateurs matériels. De ce fait, ils remettent en question l'aspect temps réel du simulateur. Il convient alors de faire coïncider les latences matérielles avec les temps de réaction des éléments biologiques pour retrouver une pertinence temporelle. Les délais observables lors de transmissions synaptiques sont hélas peu modélisés ; la puissance des simulations matérielles pourrait d'ailleurs contribuer à leur détermination. Nous ne possédons pour l'instant qu'une plage de valeurs relativement floue, située entre quelques dizaines de microsecondes et la milliseconde.

4.6.3 Perspectives

Le système de troisième génération a pour objectif de se substituer aux simulateurs logiciels pour l'étude de réseaux de l'ordre de la centaine de neurones. La structure implantée dans le circuit *Galway* permet la mise en œuvre d'une multitude de jeux de paramètres pour le formalisme d'Hodgkin et Huxley.

Les éléments testés sur le circuit, ainsi que les fonctions implantées en électronique numérique de bas niveau, permettent également d'ouvrir la voie d'un *élément neuromimétique* mixte. Cet élément regrouperait un noyau de calcul analogique, déjà éprouvé, accompagné des fonctions numériques permettant de transformer l'entrée

de chaque multisynapse en un grand nombre de monosynapses plastiques soumises aux phénomènes de saturation et de STDP. Son interface serait une connexion série bas débit pour la configuration, associée à une liaison à haut débit pour la circulation en temps réel des évènements du réseau.

Finalement, l'ordinateur de contrôle n'est plus nécessaire pour le calcul de la STDP, mais reste impliqué dans le fonctionnement en temps réel du système. Il peut être utilisé pour calculer une plasticité plus complexe sur quelques éléments donnés, mais aussi pour simuler un environnement virtuel au réseau. Il peut ensuite être associé au caractère reconfigurable des neurones pour moduler leurs paramètres, et ainsi jouer le rôle d'une glande hormonale contrôlée par le réseau lui-même.

Conclusion

Au terme de cet ouvrage, nous allons réaliser un bilan des travaux effectués. Ces derniers seront replacés dans le contexte des neurosciences computationnelles, nous poserons ensuite les questions qui permettront d'aller plus loin encore dans la mise en place et l'exploitation de simulations biologiquement réalistes de réseaux de neurones. Nous verrons finalement les perspectives qu'ouvrent les systèmes réalisés.

Bilan des travaux

Au fil de ces travaux, nous sommes montés d'un niveau de complexité dans la simulation neuronale biologiquement réaliste. L'équipe *Ingénierie des Systèmes Neuromorphiques* avait préparé cette transition grâce aux travaux de L. Alvado. Les travaux de S. Saïghi permettent d'augmenter le potentiel du système en repoussant les capacités d'intégration et en autorisant une évolution des modèles eux-mêmes, améliorant d'autant la flexibilité du système. Le circuit *Claudia* était destiné à augmenter encore la densité d'integration mais s'est révélé défectueux. Dans ce contexte, les réalisations de l'auteur ont été les suivantes :

- la conception de la carte PCI PAX qui accueille les circuits *Trieste* ;
- la configuration du FPGA de la carte PAX pour le pilotage des circuits et les transferts PCI ;
- l'établissement de l'environnement logiciel du système PAX (système, pilote de périphérique, bibliothèque d'interface) ;
- la conception des cartes du système PAX2 destinées, entre autres, à accueillir le circuit *Claudia* ainsi que les synapses pour réseau de neurones hybrides ;
- l'élaboration de la structure numérique de PAX2, et la réalisation des modules de communication et d'accès aux circuits ;
- la rédaction d'une nouvelle bibiothèque d'interface adaptée à PAX2 ;
- la conception du circuit intégré *Galway* ;
- la spécification détaillée de l'architecture de PAX3 ;
- l'élaboration de la structure de calcul embarqué de la plasticité sur PAX3.

L'évolution de la structure de gestion du réseau aboutit à un calcul mixte, au plus près du circuit, de synapses plastiques selon la STDP. Nous disposons alors des outils théoriques et descriptifs suffisants pour résoudre ces algorithmes au sein même du circuit. Il est également à notre portée de réaliser un canal postsynaptique complet. C'est-à-dire un canal analogique unique pour chaque type de synapse, mais avec une gestion numérique capable de prendre en charge les phénomènes de saturation synaptique.

Les systèmes permettent également la réalisation d'expériences sur des réseaux de neurones hybrides. La connexion à des neurones vivants isolés à l'aide de microélectrodes intracellulaires était déjà exploitée par l'équipe et ses partenaires. La modularité du système de troisième génération sera associée aux travaux des projets NEUROBIT (IST-2001-33564) et IDEA (NEST-2003-1 ADVENTURE, 516432). Ce dernier mène, entre autres, à la configuration sur FPGA du traitement de signal nécessaire à l'exploitation d'un MEA. Il sera alors possible d'interfacer un MEA avec le système PAX3 permettant l'établissement de réseaux hybrides de grande envergure, et mêlant un nombre significatif de chaque type de neurones : vivants et artificiels.

Contribution aux neurosciences computationnelles

Nous avons vu, dans l'état de l'art, que de nombreuses approches existaient pour simuler les réseaux de neurones ou les interfacer aux cellules vivantes. Contrairement aux réseaux de neurones formels, le nombre de neurones dans une structure à vocation de réalisme n'est qu'un aspect de l'intérêt qu'elle présente. La figure 1.15 en page 40 nous a déjà montré un éventail des solutions disponibles. La figure 4.20 nous montre, elle, les possibilités de chaque système, ainsi que le support de simulation utilisé. Le système référencé *Renaud (2007)* correspond aux possibilités des deux systèmes PAX2 et PAX3, la répartition entre les supports analogiques, numériques matériel et logiciel correspond au système PAX3. Les simulateurs numériques ([HIN89], [BLA98]) sont présentés comme résolvant le modèle de Hodgkin et Huxley, mais sont évidemment capable de résoudre des modèles plus simples.

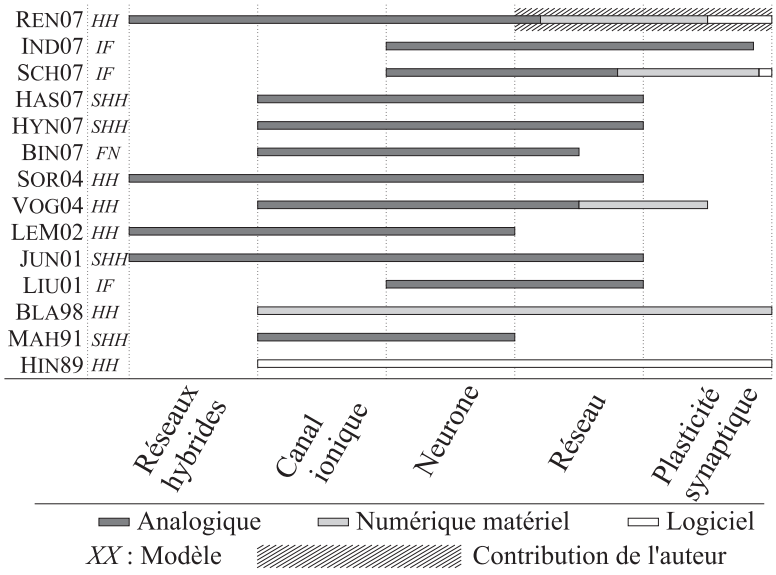

FIG. 4.20 – Positionnement des travaux par rapport à l'état de l'art

Des principes de simulation n'utilisant pas de circuits intégrés sont également présentés puisque l'usage d'ASICs n'est pas une fin en soit pour les neurosciences computationnelles. Le logiciel NEURON a d'ailleurs permis la simulation de réseaux plastiques bien avant les simulateurs matériels. Or, les propriétés intéressantes pour les neurosciences sont précisément les aspects d'adaptation et de diversité, jusqu'ici

sous-considérés par les concepteurs car ils ne mettent pas l'accent sur la densité d'intégration. Les systèmes de simulation de réseaux plastiques sont encore peu nombreux. De par leur approche unique, les travaux ici présentés permettent, pour la première fois, l'usage de modèles aussi souples et précis que ceux des solutions logicielles. En effet, les travaux de J. Schemmel privilégient le nombre de neurones, alors que ceux de G. Indiveri, implantant une plasticité simplifiée pour répondre à des contraintes technologiques d'intégration, s'attachent plus au développement technique qu'à la plasticité biologique. Les autres méthodes de simulation de réseaux plastiques ne sont pas capables de garantir une quelconque échelle de temps.

L'augmentation de la capacité de calcul du système en se fixant une échelle de temps supérieure à 1 présente un intérêt pour la reproduction de fonctions biologiques complexes et lentes. Cependant, la perte du temps réel signifie l'abandon de réseaux hybrides. Les systèmes présentés dans ces travaux sont les premiers à avoir la structure et la capacité de combiner réseaux artificiels plastiques et réseaux hybrides. Cette combinaison ouvre la voie à de nombreuses expériences en vue d'une meilleure compréhension du comportement des cellules nerveuses.

Un dialogue avec les modélisateurs

Ces modèles, déjà précis, peuvent être affinés par l'ajout de nouvelles propriétés. Peu de modèles s'intéressent aux parcours des informations à l'extérieur du soma. Plus particulièrement, les aspects temporels d'une propagation synaptique sont sujets à controverse.

Pour simuler des réseaux plus importants mais biologiquement réalistes, les phénomènes de propagation doivent être pris en compte. Les systèmes réalisés sont à la fois le révélateur du problème (puisque les délais ne sont pas une contrainte sur les simulations numériques), des outils de résolution (puisqu'ils offrent une puissance suffisante pour comparer un grand nombre de propagations différentes) et une référence pour la validation d'hypothèses.

En restant dans le domaine de la description temporelle, il est nécessaire de définir plus précisément la cinétique de réaction des potentiels de membrane. Dans le modèle de synapse que nous utilisons, la somme de deux stimulations s'effectue par l'équivalent d'un délai pour l'une d'entre elles. Bien que tout soit fait pour réduire ce temps de décalage, et que cela n'entrave pas le fonctionnement du réseau, il reste à établir les conséquences générées sur la plasticité, ainsi que sur l'ensemble des phénomènes basés sur la coïncidence des évènements.

Un dernier point important se situe au niveau de la variabilité des modèles. Le formalisme de Hodgkin et Huxley permet des configurations de paramètres qui ne traduisent pas de comportement neuronal. Cependant, la variabilité observable en biologie a pour conséquence de rendre plausibles un grand nombre de jeux de paramètres. La grande difficulté est de différencier un comportement de neurone artificiel par rapport à un autre qui, certes, suit le formalisme, mais qui ne présente pas les caractéristiques d'une cellule nerveuse. Cette différence peut d'ailleurs ne pas être observable sur un tracé de potentiel d'action. Une définition de l'espace de validité des paramètres du modèle éviterait que ne se reproduise la mauvaise interprétation du comportement des circuits *Claudia*. Dans quelle mesure une telle restriction est-elle valable ? Quand faut-il dire que tel circuit ne reproduit pas un comportement de neurone ? Comment distinguer un comportement anormal d'une dérive observable

dans le cadre de la dispersion naturelle des cellules ? Ce travail de longue haleine nécessite de grouper les efforts de plusieurs spécialités (modélisateurs, biologistes...) pour produire des simulateurs plus robustes aux dispersions.

Perspectives

Les développements qui viennent d'être présentés ouvrent la voie à de nombreux axes de développement et de recherche qui concernent chacun des aspects abordés dans cette thèse pluridisciplinaire.

Intégration des fonctions de calcul numérique et d'interface

Sur le plan de l'électronique intégrée, ces travaux utilisent l'ensemble des outils nécessaires à l'intégration de fonctions adaptatives de la synapse au niveau du circuit : structure, théorie et description VHDL. En les combinant à la bibliothèque analogique, nous disposons des techniques et du recul suffisants pour la conception future de neurones artificiels sur circuits intégrés fonctionnant de manière autonome et disposant de nombreuses entrées synaptiques. Ces circuits, intégrant calculs analogique et numérique, ainsi que des convertisseurs et de la mémoire, réduiraient les besoins en interface et en connexion externes. L'usage de mémoires non volatiles les destinerait à un fonctionnement en prothèses ou implants.

Recherche méthodologique d'auto-calibration

Un travail sur la méthodologie reste nécessaire autour du circuit réalisé et des suivants qui utiliseront la bilbiothèque analogique. En effet, chaque paramètre doit actuellement être réglé pour s'assurer que le modèle implanté est correct. Il peut s'agir de travail logiciel visant à analyser le comportement du circuit pour en affiner la fidélité. Ce travail peut également être conduit par l'ajout de circuits analogiques spécifiques, capables de rendre compte des valeurs des paramètres tels qu'ils sont perçus par les équations. Optimisée, l'autocalibration sera rapportée au niveau du FPGA.

Une étude des contraintes sur les modèles réseau

À l'échelle du système, un travail sur les capacités réalistes requises en simulation d'un réseau biologique, plutôt que sur un pire cas théorique, permettra d'augmenter considérablement la taille des réseaux simulés. En effet, nous avons vu que la synchronisation d'un réseau de neurones sur une fenêtre temporelle de quelques dizaines de microsecondes n'est pas réaliste. Quelle est alors la taille du réseau nécessaire pour qu'une activité de dix potentiels d'action par microseconde soit probable ? Certainement quelques milliers de neurones. Dans ce cas, la possibilité de connecter chaque neurone à tous les autres est-elle encore intéressante ? L'est-elle déjà à l'échelle de la centaine de neurones ? Le pas à franchir pour obtenir un système dont la taille même allège les contraintes est faible ; et la structure de PAX3 est peut-être valable pour des structures bien plus étendues. La prochaine évolution de réseau devra tenir compte de toutes ces questions.

Une optimisation automatisée des ressources

Pour un système plus étendu, la gestion ne peut pas s'effectuer en s'adressant directement aux circuits. Seul est intéressant le modèle utilisé, le numéro de circuit importe peu du point de vue de l'utilisateur. Il y a donc tout un travail d'abstraction à effectuer au niveau des outils informatiques de configuration. Une gestion des fichiers de description issus des logiciels de simulation les plus répandus dans la communauté des neurosciences augmente considérablement le portage d'expérimentations. Dans un tel système au sein duquel toutes les configurations ne sont pas envisageables pour un neurone, l'outil logiciel effectue au mieux la répartition des unités nécessaires pour obtenir le réseau désiré. Les restrictions à l'échelle d'un circuit doivent être dosées pour avoir peu ou pas d'incidences du point de vue de l'utilisateur.

Augmentation de la complexité descriptive du réseau

L'augmentation des capacités du système peut également être observée du point de vue des capacités adaptatives du réseau. Plusieurs lois de STDP peuvent être exploitées au sein du même réseau pour combiner diversité de neurones et diversité de plasticités synaptiques. En outre, le système peut être équipé d'un processeur embarqué ainsi que d'une connexion temps réel vers un ordinateur, ce qui laisse bien des possibilités de calcul de phénomènes d'adaptation. La reconfigurabilité des circuits permet également de reproduire des phénomènes chimiques qui altèrent les neurones eux-mêmes. Cette particularité permise par le modèle Hodgkin-Huxley ouvre la voie d'un niveau supplémentaire de régulation de l'activité nerveuse.

Un lien entre activité et comportement

Un réseau de neurones de taille suffisante doit permettre le contrôle d'un robot réel ou virtuel. Une telle plateforme permet d'observer le réseau pendant une phase d'apprentissage en associant des stimuli consécutifs à l'évolution du robot. La différenciation entre punition et récompense, ou encore l'évolution d'un réseau en fonction de l'environnement extérieur constituent une contribution importante aux neurosciences.

L'ensemble de ces pistes montre à quel point de tels systèmes participent à l'effort de recherche dans différents domaines. Les moyens de l'équipe *Ingénierie des Systèmes Neuromorphiques* ne lui permettent pas de tous les suivre, d'autant qu'ils font appel à de nombreuses compétences, au-delà de l'électronique. Il sera donc nécessaire d'effectuer des choix. A ce titre, la contribution des projets européens est très importante, puisqu'elle apporte des ressources humaines et renforce les liens de l'équipe avec les modélisateurs. Les neurosciences forment une discipline avec un grand potentiel d'explorations dans un cadre de recherche pluridisciplinaire auxquelles l'auteur ambitionne d'encore contribuer.

Bibliographie

[Abb94] L. F. Abbott. *Single neuron dynamics : an introduction, in neural modeling an neural networks.* Pergamon Press, Oxford, 1994.

[Alv03a] L. Alvado. *Neurones artificiels sur Silicium : une évolution vers les réseaux.* Thèse de doctorat, Université Bordeaux 1, N° d'ordre 2674, 2003.

[Alv03b] L. Alvado, S. Saïghi, J. Tomas, et S. Renaud. An exponential-decay synapse integrated circuit for bio-inspired neural network. In *7th International Work Conference on Artificial et Natural Neural Networks*, volume 1, pages 670–677, 2003.

[Alv04] L. Alvado, J. Tomas, S. Saïghi, S. Renaud, T. Bal, A. Destexhe, et G. Le Masson. Hardware computation of conductance-based neuron models. *Neurocomputing*, Vol 58-60 :109–115, 2004.

[Bad06] M. Badoual, Q. Zou, A. P. Davison, M. Rudolph, T. Bal, Y. Fregnac, et A. Destexhe. A biophysical et phenomenological model of multiple spike interactions in spike-timing dependant plasticity. *Int. J. Neural Systems*, in press, 2006.

[Bin06] S. Binczak, S. Jacquir, J. M. Bilbault, V. B. Kazantsev, et V. I. Nekorkin. Experimental study of electrical fitzhugh-nagumo neurons with modified excitability. *Neural Networks*, Vol. 19 :684–693, 2006.

[Bin03] S. Binczak, V. B. Kazantsev, V. I. Nekorkin, et J. M. Bilbault. Experimental study of bifurcations in modified fitzhugh-nagumo cell. *Electronic Letters*, Vol. 39 :961–962, 2003.

[Bla98] J.J. Blake, L.P. Maguire, T.M. McGinnity, R. Roche, et L.J. McDaid. The implementation of fuzzy systems, neural networks and fuzzy neural networks using FPGAs *Information Science*, Vol. 112, N°1-4 :151–168, 1998.

[Boa00] K. Boahen. Point-to-point connectivity between neuromorphic chips using address events. *IEEE trans. Circuits et Systems II : Analog and Digital Signal Processing*, Vol. 47 :416–434, 2000.

[Bor05] Y. Bornat, J. Tomas, S. Saïghi, et S. Renaud. Bicmos analog integrated circuits for embedded spiking neural networks. In *XX Conference on Design of Circuits et Integrated Systems, DCIS 2005, Lisbon*, 2005. ISBN 972-99387-2-5.

[Buc05] D. Bucher, A. A. Prinz, et E. Marder. Animal-to-animal variability in motor pattern production in adults et during growth. *J. Neuroscience*, Vol 25(7) :1611–1619, 2005.

[Cha99] J. K. Chapin, K. A. Moxon, et R. S. Markowitz et M. A. L. Nicolelis. Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex. *Nature Neuroscience*, Vol. 2 :664–670, 1999.

[DeM01] T. B. DeMarse, D. A. Wagenaar, A. W. Blau, et S. M. Potter. The neurally controlled animat : Biological brains acting with simulated bodies. *Autonomous Robots*, Vol. 11 :305–310, 2001.

[Des94] A. Destexhe, Z. F. Mainen, et T. J. Sejnowski. An efficient method for computing synaptic conductances based on a kinetic model of receptor binding. *Neural Computation*, Vol. 6 :14–18, 1994.

[Des99a] A. Destexhe, Z. F. Mainen, et T. J. Sejnowski. *Kinetic models of synaptic transmission.* in Methods in Neuronal Modeling, Eds. C. Koch et I. Segev, MIT Press, Cambridge ma, usa, 1999.

[Des99b] A. Destexhe et D. Paré. Impact of network activity in the integrative properties of neocortical pyramidal neurons in vivo. *J. Physiology*, Vol. 81 :1531–1547, 1999.

[Dou00] V. Douence. *Circuits et systèmes de modélisation analogique de neurones biologiques.* Thèse de doctorat, Université Bordeaux 1, N° d'ordre 2324, 2000.

[Dup98] D. Dupeyron. *Contribution à l'intégration sur silicium de modèles analogiques de neurones biologiques.* Thèse de doctorat, Université Bordeaux 1, N° d'ordre 1967, 1998.

[Eli93] J. G. Elias. Artificial dendritic trees. *Neural computation*, Vol. 5 :648–663, 1993.

[Fie04] J. Fieres, A. Grübl, S. Philipp, K. Meier, J. Schemmel, et F. Schürmann. A platform for parallel operation of vlsi neural network. In *Proceedings of Brain Inspired Cognitive Systems.* University of Stirling, UK, 2004. ISBN 1 85769 199 7.

[Fit61] R. FitzHugh. Impulses and physiological states in models of nerve membrane. *J. Biophys.*, Vol. 1 :445–466, 1961.

[Fus02] S. Fusi. Hebbian spike-driven synaptic plasticity for learning patterns of mean firing rates. *Bio. Cybernetics*, Vol. 87 :459–470, 2002.

[Ger02] W. Gerstner et W. M. Kistler. *Spiking neuron models.* Cambridge Univ. Press, 2002.

[Gol46] H. H. Goldstine et A. Goldstine. The electronic numerical integrator et computer (eniac). (reprinted in *The Origins of Digital Computers : Selected Papers*, Springer-Verlag, New York, 1982, 359-373), 1946.

[Has07] P. Hasler, S. Koziol, et E. Farquhar. Transistor channel dendrites implementing hmm classifiers. In iscas *'2007, submitted*, 2007.

[Heb49] D. O. Hebb. *The Organization of Behavior.* Jon Wesley et Sons, New York, 1949.

[Hin97] M. L. Hines et N. T. Carnevale. The neuron simulation environment. *Neural Comp.*, Vol. 9 :1179–1209, 1997.

[Hin89] M. Hines. A program for simulation of nerve equations with branching geometries. *Int. J. Bio-Med. Comput.*, Vol. 24 :55–68, 1989.

[Hin07] K. M. Hinna et K. Boahen. Silicon neurons that burst when primed. In iscas*'2007, submitted*, 2007.

[Hod52] A. L. Hodgkin et A. F. Huxley. A quantitative description of membrane current and its application to conduction et excitation in nerve. *Journal of Physiology*, Vol. 116 :500–544, 1952.

[Ind07] G. Indiveri, S. Mitra, et S. Fusi. Spike-based vlsi learning circuits for classifying complex patterns in networks of integrate-and-fire neurons. In iscas*'2007, submitted*, 2007.

[Ind06] Giacomo Indiveri, Elisabetta Chicca, et Rodney Douglas. A vlsi array of low-power spiking neurons and bistable synapses with spike-timing dependent plasticity. *IEEE Trans. Neural Networks*, Vol. 17 :211–221, 2006.

[Ito94] M. Ito. La plasticité des synapses. *La recherche*, Vol. 25, N°267 :778–785, 1994.

[Izh04] Eugene M. Izhikevich. Which model to use for cortical spiking neurons ? *IEEE Trans. Neural Networks*, Vol. 15 :1063–1070, 2004.

[Jun01] R. Jung, E. J. Brauer, et J. J. Abbas. Real-time interaction between a neuromorphic electronic circuit and the spinal cord *IEEE Trans. on Neural Systems and Rehab. Eng.*, Vol. 9(3) :319–326, 2001.

[Laf98] A. Laflaquière. *Neurones artificiels sur silicium : conception analogique et construction de réseaux hybrides.* Thèse de doctorat, Université Bordeaux 1, N° d'ordre 1801, 1998.

[LeM95] G. Le Masson, S. Le Masson, et M. Moulins. From conductances to neural network properties : analysis of simple circuits using the hybrid network method. *Prog. Biophysics*, Vol. 64, N° 2/3 :201–220, 1995.

[LeM02] G. Le Masson, S. Renaud-Le Masson, et D. Debay. Feedback inhibition controls spike transfer in hybrid thalamic circuits. *Nature*, Vol. 417 :854–858, 2002.

[Lev06] T. Levi, N. Lewis, J. Tomas, et P. Fouillat. Scaling rules for mos analog design reuse. In mixdes *2006, International Conference Mixed Design of Integrated Circuits et Systems, Gdansk*, 2006.

[Lev85] W. B. Levy et N. L. Desmond. *Synaptic Modification, Neuron-Selectivity, et Nervous System Organization*, Chap. 6, pages 105–121. The rules of elemental synaptic plasticity, Lawrence Erlbaum Assoc., Hillsdale, New Jersey, 1985.

[Liu04] S.-C. Liu et R. Douglas. Temporal coding in a silicon network of integrate-and-fire neurons. *IEEE, Trans. Neural Networks*, Vol 15 :1305–1314, 2004.

[Liu01] S.-C. Liu, J. Kramer, G. Indiveri, T. Delbrück, T. Burg et R. Douglas. Orientation-selective avlsi spiking neurons. *Neural Networks*, Vol 14 :629–643, 2001.

[Mah89] M. A. Maher, S. P. DeWeerth, et M. A. Mahowald. Implementing neural architectures using analog vlsi circuits. *IEEE Trans. on Circuits et Systems*, Vol. 5, N°36 :643–652, 1989.

[Mah91] M. Mahowald et R. Douglas. A silicon neuron. *Nature*, Vol 354 :515–518, 1991.

[Mal33] R. R. M. Mallock. An electrical calculating machine. In *Proceedings of the Royal Society of London*, pages 457–483, 1933.

[Mar04] S. Martinoia, V. Sanguineti, L. Cozzi, L. Berdondini, J. van Peltd, J. Tomas, G. Le Masson, et F. Davide. Towards an embodied in vitro electrophysiology : the NeuroBIT project. *Neurocomputing*, Vol. 58-60 :1065–1072, 2004.

[Mil94] W. T. miller. Real-time neural network control of a biped walking robot. *IEEE, Control Systems Magazine*, Vol. 14 :41–48, 1994.

[Mit06] S. Mitra, S. Fusi, et G. Indiveri. A vlsi spike-driven dynamic synapses which learns only when necessary. In *ISCAS'2006, Greece, 2006*, page trouver dans le proceedings, 2006.

[Mor81] C. Morris et H. Lecar. Voltage oscillations in the barnacle giant muscle fiber. *Biophys. J.*, Vol 35 :193–213, 1981.

[Nis81] Takao Nishitani, Rikio Maruta, Yuichi Kawakami, et Hideto Goto. A single-chip digital signal processor for telecommunication applications. *IEEE journal of Solid-State Circuits*, SC-24 :372–376, 1981.

[Now03] T. Nowotny, V. P. Zhigulin, A. I. Selverston, H. D. I. Abarbanel, et M. I. Rabinovich. Enhancement of synchronization in a hybrid neural circuit by spike-timing dependent plasticity. *J. Neuroscience*, Vol. 23(30) :9776–9785, 2003.

[Opr04] S. A. Oprisan, A. A. Prinz, et C. C. Canavier. Phase resetting and phase locking in hybrid circuits of one model and one biological neuron. *Biophysical Journal*, Vol. 87 :2283–2298, 2004.

[Pak05] K. Pak. Contribution à la conception d'une synapse artificielle en technologie $0,35\,\mu m$. Rapport de stage, dea d'électronique, Université Bordeaux 1, 2005. Laboratoire ixl.

[Pel76] A. Peled. On the hardware implementation of digital signal processors. *IEEE Trans., Acoust., Speech, Signal Processing*, ASSP-24 :76–86, 1976.

[Ren07] S. Renaud, J. Tomas, Y. Bornat, A. Daouzli, et S. Saïghi. Neuromimetic ics with analog cores : an alternative for designing spiking neural networks. In iscas*'2007, submitted*, 2007.

[Ren04] S. Renaud-Le Masson, G. Le Masson, L. Alvado, S. Saïghi, et J. Tomas. A neural simulation system based on biologically-realistic electronic neurons. *Information Science*, Vol. 161, N°1-2 :57–69, 2004.

[Saï04] S. Saïghi. *Circuits et systèmes de modélisation analogique de réseaux de neurones biologiques : Application au développement d'outils pour les neurosciences computationnelles.* Thèse de doctorat, Université Bordeaux 1, N° d'ordre 2891, 2004.

[Sch07] J. Schemmel, D. Brüderle, A. Grübl, K. Meier, et E. Müller. Modelling synaptic plasticity within networks of highly accelerated I&F neurons. In iscas*'2007, submitted*, 2007.

[Sch04] J. Schemmel, K. Meier, et E. Muller. A new vlsi model of neural microcircuits including spike time dependent plasticity. In *Neural Networks,*

2004. Proceedings. 2004 IEEE International Joint Conference on, volume 3, pages 1711–1716, 2004.

[Sha93] Andrew A. Sharp, Michael B. O'neil, L. F. Abbott, et Eve Marder. The dynamic clamp : artificial conductances in biological neurons. *Trends. in Neurosciences*, Vol. 16 :389–394, 1993.

[Sor04] M. Sorensen, S. De Weerth, G. Cymbalyuk, et R. L. Calabrese. Using a hybrid neural system to reveal regulation of neuronal network activity by an intrinsic current *J. Neurosci.*, Vol. 24 :5427–5438, 2004.

[Tri98] D. Tritsch, D. Chesnoy-Marchais, et A. Feltz. *Physiologie du neurone*. Doin, Paris, 1998. isbn 2-7040-0872-8.

[Vog04] R.J. Vogelstein, U. Malik, et G. Gauwenberghs. Silicon spike-based synaptic array and address-event transceiver In *Proceedings of* iscas*'04, submitted*, vol. 5 :385–388, 2007.

[Zec01] G. Zeck et P. Fromherz. Noninvasive neuroelectronic interfacing with synaptically connected snail neurons immobilized on a semiconductor chip. In *Proceedings of the National Academy of Sciences*, pages 10457–10462, 2001.

[Zou06a] Q. Zou. *Modèles computationnels de la plasticité impulsionnelle : synapses, neurones et circuits*. Thèse de doctorat, Université Paris VI, 2006.

[Zou06b] Q. Zou, Y. Bornat, S. Saïghi, J. Tomas, S. Renaud, et A. Destexhe. Analog-digital simulations of full conductance-based networks of spiking neurons with spike timing dependant plasticity. *Network : Computation in Neural Systems*, In Press, 2006.

[Zou06c] Q. Zou, Y. Bornat, J. Tomas, S. Renaud, et A. Destexhe. Real-time simulations of networks of Hodgkin-Huxley neurons using analog circuits. *Neurocomputing*, volume 69, pages 1137–1140, 2006.

Publications de l'auteur

Revues internationales avec comité de lecture

[A1] Q. Zou, Y. Bornat, S. Saïghi, J. Tomas, S. Renaud, et A. Destexhe. Analog-digital simulations of full conductance-based networks of spiking neurons with spike timing dependant plasticity. *Network : Computation in Neural Systems*, In Press, 2006.

[A2] Q. Zou, Y. Bornat, J. Tomas, S. Renaud, et A. Destexhe. Real-time simulations of networks of hodgkin-huxley neurons using analog circuits. *Neurocomputing*, 69 :1137–1140, 2006.

Conférences internationales avec comité de lecture et acte

[B1] S. Renaud, J. Tomas, Y. Bornat, A. Daouzli, et S. Saïghi. Neuromimetic ICs with analog cores : an alternative for designing spiking neural networks. *In* ISCAS'2007, submitted, *2007.*

[B2] Y. Bornat, S. Renaud, J. Tomas, S. Saïghi, Q. Zou, et A. Destexhe. An analog/digital simulation system for biomimetic neural networks. *I*n EPFL *Latsis Symposium 2006, Dynamical principles for neuroscience and intelligent biomimetic devices, Lausanne (Switzerland)*, 2006.

[B3] S. Saïghi, Y. Bornat, J. Tomas, et S. Renaud. Neuromimetic ICs and system for parameters extraction in biological neuron models. *In* InternationaI Symposium on Circuits And Systems 2006, ISCAS06, Island of Kos, Greece, *pages 4207–4210, 2006.*

[B4] Y. Bornat, J. Tomas, S. Saïghi, et S. Renaud. BiCMOS analog integrated circuits for embedded spiking neural networks. *In* XX Conference on Design of Circuits and Integrated Systems, DCIS 2005, Lisbon, *2005.*

[B5] S. Saïghi, J. Tomas, Y. Bornat, et S. Renaud. A conductance-based silicon neuron with dynamically tunable model parameters. *In* 2nd International IEEE EMBS Conference on Neural Engineering, Arlington (VA), USA, *pages 285–288, 2005.*

[B6] Q. Zou, Y. Bornat, J. Tomas, S. Renaud, et A. Destexhe. Real time simulations of networks of hodgkin-huxley neurons using analog circuits. In *14th Annual Computational Neuroscience Meeting,* CNS 2005*, Madison,* USA, pages 50–57, 2005.

[B7] S. Saïghi, J. Tomas, Y. Bornat, et S. Renaud. A neuromimetic integrated circuit for interactive real-time simulation. *In* 1st International Work-Conference on the Interplay Between Natural and Artificial Computation, IWINAC 2005*, pages 338–346, 2005.*

[B8] S. Saïghi, J. Tomas, Y. Bornat, et S. Renaud-Le Masson. A mixed neuromorphic ASIC for computional neurosciences. In XIX *Conference on Design of Circuits and Integrated Systems,* DCIS 2004*, Bordeaux*, pages 299–303, 2004.

[B9] S. Saïghi, J. Tomas, Y. Bornat, et S. Renaud-Le Masson. Silicon integration of biological neurons models. *In* XVIII Conference on Design of Circuits and Integrated Systems, DCIS 2003, Ciudad Real, Espagne*, pages 597–602, 2003.*

Conférences nationales avec comité de lecture et acte

[C1] N. Lewis, Y. Bornat, L. Alvado, C. Lopez, A. Daouzli, T. Levi, J. Tomas, S. Saïghi, et S. Renaud. PAX : un outil logiciel / matériel d'investigation pour les neurosciences computationnelles. *In* 1ère conférence française de Neurosciences Computationnelles, Pont-à-Mousson, 23-24 octobre 2006*, pages 171–174, 2006.*

[C2] Y. Bornat. Un asic mixte neuromimétique paramétrable dédié aux neurosciences computationnelles. *I*n *Journées Nationales du Réseau Doctoral de Microélectronique* jnrdm 2005*, Paris*, pp. 218-221 2005.

Rapports d'avancement et/ou techniques

[D1] Y. Bornat, L. Alvado, C. Lopez, S. Renaud, S. Saïghi, et J. Tomas. The mixed A/D simulation system of biomimetic neurons : pax2. *T*echnical report, SenseMaker, 2005. Deliverable 21.

[D2] S. Renaud, J. Tomas, L. Alvado, C. Lopez, Y. Bornat, et S. Saïghi. Technical report of the final closed-loop set up prototype. *Technical report, NeuroBit, 2005. Deliverable 5.2.*

[D3] J. Tomas et Y. Bornat. Development of the bio-interface dedicated to smu. *Technical report, SenseMaker, 2005. Deliverable 32.*

[D4] J. Tomas, S. Renaud, et Y. Bornat. Translation biogical-like and formal plasticty algorithms. *Technical report, SenseMaker, 2004. Deliverable 14.*

[D5] S. Renaud, J. Tomas, et Y. Bornat. Translation rules between biogical-like and smu artificial neurons. *Technical report, SenseMaker, 2003. Deliverable 11.*

Annexe A

Modèle du neurone intégré dans les circuits *Trieste* et *Claudia*

Le modèle intégré dans les circuits *Trieste* correspond aux modèles inhibiteur et excitateur les plus courants au sein des neurones corticaux. Pour la cellule excitatrice, il s'agit de la *cellule à décharge régulière* (*regular spiking cell* ou RS). Dans les neurones inhibiteurs, il s'agit de la *cellule à décharge rapide* (*fast spiking cell* ou FS) dépourvue de phénomène de modulation lente.

Les deux cellules utilisent les mêmes courants principaux. Le modèle est adapté selon les entrées numériques du circuit : I/E, V_{cmd}, CMD0 et CMD1. La seule approximation réalisée se situe sur les cinétiques des canaux ioniques : seul la cinétique d'inactivation du canal sodium est dépendante du potentiel de membrane, il s'agit d'une dépendance binaire.

A.1 Le circuit *Trieste*

$$C_{mem} \cdot \frac{\partial V_{mem}}{\partial t} = -I_{Na} - I_K - I_{Mod} - I_{fuite}$$

A.1.1 Courant de fuite

$$I_{fuite} = g_{fuite} \cdot (V_{mem} - V_{equi})$$

$$\begin{cases} g_{fuite} = 22\,\text{nS} \text{ si I/E} = 0 \\ g_{fuite} = 33\,\text{nS} \text{ si I/E} = 1 \end{cases}$$

$$\begin{cases} V_{equi} = -70\,\text{mV} \text{ si } V_{cmd} = 0 \\ V_{equi} = -80\,\text{mV} \text{ si } V_{cmd} = 1 \end{cases}$$

A.1.2 Courant sodium

$$I_{Na} = g_{Na} \cdot m^3 \cdot h \cdot (V_{mem} - V_{Na})$$

où $g_{Na} = 11\,\mu\text{S}$ et $V_{Na} = 50\,\text{mV}$.

pour l'activation

m est solution de l'équation :

$$\tau_m \cdot \frac{\partial m}{\partial t}(t) = m_\infty(V_{mem}) - m(t)$$

avec

$$m_\infty(V_{mem}) = \frac{1}{1 + e^{-\frac{V_{mem} - V_{offset}}{V_{pente}}}}$$

sachant que $\tau_m = 0,03\,\text{ms}$, $V_{offset} = -37\,\text{mV}$ et $V_{pente} = 7,2\,\text{mV}$.

pour l'inactivation

h est solution de l'équation :

$$\tau_h(V_{mem}) \cdot \frac{\partial h}{\partial t}(t) = h_\infty(V_{mem}) - h(t))$$

avec

$$\begin{cases} \tau_h = 3\,\text{ms} & \text{si } V_{mem} > 0 \\ \tau_h = 0,25\,\text{ms} & \text{si } V_{mem} < 0 \end{cases}$$

$$h_\infty(V_{mem}) = \frac{1}{1 + e^{\frac{V_{mem} - V_{offset}}{V_{pente}}}}$$

sachant que $V_{offset} = -42\,\text{mV}$ et $V_{pente} = 4,6\,\text{mV}$.

A.1.3 Courant potassium

$$I_K = g_K \cdot n \cdot (V_{mem} - V_K)$$

où $V_K = -100\,\text{mV}$ et

$$\begin{cases} g_K = 2,2\,\mu\text{S} & \text{si I/E} = 0 \\ g_K = 1,1\,\mu\text{S} & \text{si I/E} = 0 \end{cases}$$

pour l'activation

n est solution de l'équation :

$$\tau_n \cdot \frac{\partial n}{\partial t}(t) = n_\infty(V_{mem}) - n(t)$$

avec

$$n_\infty(V_{mem}) = \frac{1}{1 + e^{-\frac{V_{mem} - V_{offset}}{V_{pente}}}}$$

sachant que $\tau_n = 3\,\text{ms}$, $V_{offset} = -37\,\text{mV}$ et $V_{pente} = 11,38\,\text{mV}$.

A.1.4 Courant modulateur lent

Ce courant n'est présent que pour le neurone excitateur.

$$I_{Mod} = g_{Mod} \cdot m \cdot (V_{mem} - V_{Mod})$$

où $V_{Mod} = -100\,\text{mV}$ et

$$\begin{cases} g_{Mod} = 0\,\text{nS} & \text{si I/E} = 0 \\ g_{Mod} = 25\,\text{nS} & \text{si (I/E, CMD1, CMD0)} = (1,0,0) \\ g_{Mod} = 50\,\text{nS} & \text{si (I/E, CMD1, CMD0)} = (1,0,1) \\ g_{Mod} = 100\,\text{nS} & \text{si (I/E, CMD1, CMD0)} = (1,1,0) \\ g_{Mod} = 200\,\text{nS} & \text{si (I/E, CMD1, CMD0)} = (1,1,1) \end{cases}$$

pour l'activation

m est solution de l'équation :

$$\tau_m(V_{mem}) \cdot \frac{\partial m}{\partial t}(t) = m_\infty(V_{mem}) - m(t)$$

avec

$$m_\infty(V_{mem}) = \frac{1}{1 + e^{-\frac{V_{mem} - V_{offset}}{V_{pente}}}}$$

sachant que $V_{offset} = -35\,\text{mV}$, $V_{pente} = 11,4\,\text{mV}$ et

$$\begin{cases} \tau_m = 300\,\text{ms} & \text{si } V_{mem} > 0 \\ \tau_m = 8\,\text{ms} & \text{si } V_{mem} < 0 \end{cases}$$

A.2 Le circuit *Claudia*

Le circuit *Claudia* regroupe quatre neurones construits sur un modèle analogue à celui qui précède. Les trois premiers sont de nature excitatrice, le dernier de nature inhibitrice.

L'entrée I/E n'est donc plus utilisée. De même, les valeurs utilisées pour V_{equi} supposent $V_{cmd} = \text{I/E}$, ce qui signifie :

$$\begin{cases} V_{equi} = -70\,\text{mV} & \text{pour un neurone inhibiteur} \\ V_{equi} = -80\,\text{mV} & \text{pour un neurone excitateur} \end{cases}$$

Pour les neurones excitateurs, les valeurs de g_{Mod} ont également été revues. Elles sont désormais au nombre de 8 et dépendent des trois entrées CMD0, CMD1 et CMD2. Les valeurs correspondant à chaque entrée sont données dans le tableau A.1

Ces valeurs sont valables pour l'ensemble des neurones du circuit.

CMD2	CMD1	CMD0	g_{Mod}
0	0	0	33 pS
0	0	1	80 nS
0	1	0	158 nS
0	1	1	233 nS
1	0	0	308 nS
1	0	1	381 nS
1	1	0	456 nS
1	1	1	526 nS

TAB. A.1 – Les valeurs de g_{Mod} sur *Claudia*

Annexe B

Répartition des accès sur la carte PAX

Les fonctions de PAX sont accessibles en écrivant ou en lisant une sérire d'adresses mémoires. Le tableau B.1 effectue le lien entre chaque adresse et la fonction qui lui est associée.

B.1 Tableau de Répartition par adresses

Adresse	+03h	+02h	+01h	+00h
000h	Registre en-tête			
004h	États		Diodes	
008h	Valeur de l'horloge			
00Ch	Période d'horloge			
020h	Vstim0		Vcomp0	
024h	Vstim1		Vcomp1	
028h	Vstim2		Vcomp2	
02Ch	Vstim3		Vcomp3	
030h	Vstim4		Vcomp4	
034h	Vstim5		Vcomp5	
038h	Vstim6		Vcomp6	
03Ch	Vstim7		Vcomp7	
040h	I/E	Scmd	Cmd1	Cmd0
044h	-	-	-	Vlcmd
050h	Compteur d'interruptions			
060h	SynI1	SynE1	SynI0	SynE0
064h	SynI3	SynE3	SynI2	SynE2
068h	SynI5	SynE5	SynI4	SynE4
06Ch	SynI7	SynE7	SynI6	SynE6
070h	Quant. Stim.		Cycle Stim.	
080h	-	-	-	Spikes

TAB. B.1 – Répartition des fonctions sur PAX par adresse

B.2 Signification des registres

Voici maintenant la signification de chaque fonction.

Registre en-tête : En lecture, le registre en-tête fourni un entier révélateur de la version du système. Il est ainsi possible de érifier la configuration de FPGA. En écriture, tout accès à cette adresse signifie que le logiciel a pris en compte la dernière interruption. Cette fonction ne doit être utilisée qu'au début du gestionnaire d'interruption pour éviter toute perte de données.

Diodes : En écriture, envoie la valeur bit à bit vers les 16 diodes de la carte. Cette fonction permet le déverminage rapide matériel et logiciel, même si le système d'exploitation paraît bloqué à cause des requêtes temps réel. Peut être relu.

États : En lecture, ce registre fournit une série de messages d'erreur. Ils sont donnés dans le tableau B.2, certains bits peuvent être remis à zero en écrivant dessus.

Bit	Signification d'un 1	Effaçable
0	Interruption demandée	non
1	Interruption ratée	oui
2	Écriture synaptique non reçue	oui

TAB. B.2 – Signification du registre d'états

Ces registre ne servent qu'à informer l'utilisateur du déroulement de la simulation.

Valeur de l'horloge : En lecture, nombre de cycles à 64 MHz avant la prochaine requête d'interruption. Peut être forcé par l'écritrue.

Période d'horloge : En écriture, défini le nombre de cycles à 64 MHz qui séparent deux requêtes d'interruption. Peut être relu.

Vcomp$x$: En écriture, les 12 bits de poids fort sont envoyés vers un convertisseur numérique/analogique. Le signal est présenté sur l'entrée Vcomp du circuit x, il définit le seuil de tension au delà duquel le circuit effectue un potentiel d'action. Peut être relu.

Vstim$x$: En écriture, les 12 bits de poids fort sont envoyés vers un convertisseur numérique/analogique. Le signal est présenté sur l'entrée Vstim du circuit x, il définit le courant de compensation. Peut être relu.

Cmd$n$: En écriture, défini les entrées Cmd0 et Cmd1 de chaque circuit, selon le bit de l'octet. Ces entrées définissent g_{Mod} pour le circuit. Peuvent être relus.

Scmd : En écriture, défini l'entrée Scmd de chaque circut selon le bit de l'octet. Cette entrée défini la plage de valeurs à utiliser pour le courant de compensation. Peut être relu.

I/E : En écriture, défini l'entrée I/E de chaque circut selon le bit de l'octet. Cette entrée sélectionne le modèle inhibiteur ou excitateur du circuit. Peut être relu.

Vlcmd : En écriture, défini l'entrée Vlcmd de chaque circut selon le bit de l'octet. Cette entrée se réfère au canal de fuite du circuit. Peut être relu.

Compteur d'interruptions : En lecture, compte le nombre d'interruptions non prises en compte pour vérifier le comportement du système d'exploitation. Peut être forcé ou initialisé par l'écriture.

Syn*Xn* : En écriture, défini le temps pendant lequel chaque synapse devra être stimulée au cours de la période suivante (voir Quant. Stim.). Cette valeur est prise en compte et réinitialisée à chaque interruption (début de cycle). Peut être relue.

Quant. Stim. : En écriture, défini la correspondance entre un quantum de Syn*Xn* et le nombre de cycles d'horloge à 64 MHz. Si la valeur est trop élevée, le temps restant n'est pas reporté sur le cycle suivant. Peut être relu.

Cycle Stim. : En lecture seule, informe sur la progression de la stimulation synaptique en donnant le nombre de quanta restant à envoyer.

Spikes : En lecture seule, reflette l'activité des circuit correspondant à chaque bit. Chaque bit vaut 1 si le circuit correspondant a émis un potentiel d'action lors du cycle précédent. Le registre n'est remis à zéro qu'à la fin d'un cycle lors duquel il a été consulté.

Annexe C

Codage des mots de communication sur PAX2

C.1 Tableau de codage des instructions

on présente le truc, le MSB (E) en premier... codages identiques pour STIM et P_A

Codage															Fonction
E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
1	*temps*								*s*	*neurone*					STIM
1	instant									neurone					P_A
0	1	1	*periode*												DEF_PER
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	RAZ
0	1	0	1	0	0	0	0	0	1	neurone					DEF_NUM
0	0	1	x	demi-mot						neurone					DEF_COMP
0	0	0	1	0	0	1	g_{max}			neurone					DEF_COND
0	0	0	1	0	0	0	0	0	0	0	*Id.*				DEF_FPGA
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	mot vide

TAB. C.1 – Instructions de communication sur PAX2

C.2 Signification des instructions

STIM Cette instruction provoque une stimulation, dont la force est codée par *temps*, sur la synapse *s* du neurone *neurone*. Le codage de la force est codé en nombre de cycles de l'horloge de base du système. Le bit *s* code la synapse utilisée :

- 0 : Inhibitrice ;
- 1 : Excitatrice.

Cette instruction ne circule qu'entre le maître de la boucle et le FPGA qui contôle le neurone concerné.

P_A Ce mot signale un potentiel d'action émit par le neurone *neurone* et survenu au moment *instant*. Cette dernière valeur dépend du nombre de cycles de l'horloge de base écoulés depuis la dernière remise à zéro. Cette instruction ne circule qu'entre le FPGA qui contrôle le neurone concerné et le maître de la boucle.

DEF_PER Cette instruction définit la période de l'horloge de base du système. Elle est définie à $periode \times 0,5\,\mu s$.

RAZ Déclenche une procédure de remise à zéro du système. Cette initialisation est effective au bout d'un certain délai qui dépend du numéro du FPGA, de sorte qu'elle est instantannée sur l'ensemble du système malgré le temps de propagation des instructions.

DEF_NUM Attribue le numéro *neurone* au premier neurone géré par le FPGA. L'ensemble des neurones locaux est ensuite numéroté localement. Cette instruction est renvoyée à l'élément suivant de la boucle avec la valeur *neurone* augmentée du nombre de neurones locaux.

Lorsque l'instruction revient au maître de la boucle, tous les neurones ont été numérotés et l'instruction en contient le nombre.

DEF_COMP Détermine le courant de compensation du neurone *neurone*. Pour réduire la taille des instructions, ce courant est divisé en deux demi-mots identifiés par x :

- 0 : poids faible
- 1 : poids fort et validation de la donnée transmise

Si un demi-mot de poids faible est envoyé seul, il sera mémorisé sans être pris en compte. Si un demi-mot de poids fort est envoyé seul, il sera pris en compte et les 6 bits de poids faible seront inchangés.

DEF_COND Modifie la force de la conducance du canal modulant pour tout le circuit contenant le neurone *neurone*. Cette fonction est prise en compte pour l'ensemble du circuit de destination, même si le neurone *neurone* est de nature inhibitrice (sans canal modulant).

DEF_FPGA Attribue le numéro $Id.$ au FPGA, cette instruction est renvoyée dans la chaine avec $Id. + 1$ en argument. Lorsque le mâitre de la boucle reçoit cette instruction, il détermine le nombre de nœuds que contient la boucle.

Annexe D

Répartition des accès sur la carte PAX2

D.1 Tableau de Répartition par adresses

Dans PAX2, les accès s'effectuent uniquement en 32 bits, les adresses 000Ch et inférieures ont en lecture uniquement, les autres en écriture seule. Le tableau D.1 donne la répartition des fonctions par adresse.

Adresse	Fonction	Bits
00000h	Registre principal	5 / - [1]
00004h	Mémoires de bruit	32
00008h		32
0000Ch	Lecture de potentiels d'action	32
00010h	Mode de fonctionnement	3
00014h	Poids du bruit	8
00018h	Période du système	12
00020h	Échantillonnage du bruit	8
00024h	Masque de requêtes de bruit	32
00028h		32
08000h + $4 \times n$	Compensation	12/7 [2]
10000h + $4 \times n$	Paramètres numériques	5/3 [3]
14000h	Envoi direct (test)	15
18000h + $4 \times s$	Mise à jour du bruit	10
1C000h ...	Matrice de connections	2048×32

TAB. D.1 – Répartition des fonctions sur PAX2 par adresse

D.2 Détail des registres

Registre Principal : Ce registre indique, bit à bit, l'état de fonctionnement de chaque partie du FPGA. Il est, entre autres, utilisé pour connaitre la cause d'une interruption et le bon déroulement de la simulation. Son détail est donné par le tableau D.2.

Bit	Signification
0	
1	Présence d'un potentiel d'action en mémoire
2	
3	Nécessité de mise à jour du bruit synaptique
4	
5	
6	

TAB. D.2 – Signification bit à bit du registre principal

Les accès en écriture à cette adresse ont une signification particulière : un entier null provoque une réinitialisation du système ; un entier non nul confirme la prise en compte d'une interruption.

Mémoires de bruit : Les mémoires de bruit indiquent, bit à bit, si une série doit être, ou non, alimentée en données. Les bits sont organisés selon la table D.3.

Bit	Neurone	Synapse
0	0	inhibitrce
1		excitatrice
2	1	inhibitrce
3		excitatrice
...	...	...
62	31	inhibitrce
63		excitatrice

TAB. D.3 – Organisation des descripteurs de mémoire de bruit

Lecture des potentiels d'action : Cette adresse est la sortie de la file d'attente des évènements de potentiel d'action. Chaque lecture renvoie un codage de l'évènement le plus ancien et l'efface de la mémoire du FPGA. Les évènement sont codés sur 32 bits, les 5 bits de poids faible donne le numéro du neurone qui s'est activé, les 27 bits de poids fort contiennent, eux, le nombre de cycles système écoulés depuis la dernière remise à zéro au moment où le potentiel d'action a été détecté.

Mode de fonctionnement : Ce registre contient deux bits significatifs :

- le bit 0 détermine quelle information doit être prioritaire sur le système (0 : mode configuration / 1 : simulation de réseau)
- le bit 1, s'il est actif, déclenche l'envoi du bruit synaptique, et commence donc à vider la mémoire qui lui est attribuée.

Poids du bruit : L'ensemble des stimulations de bruit synaptique possèdent le même poids. Ce registre sert à le définir. s'il est modifié en cours de simulation, ses

effets sont immédiats. Sa valeur est sur 8 bits et code le nombre de cycles d'horloge système que doit durer le pulse de stimulation synaptique.

Période du système : L'horloge de base du système est réglable selon les besoins de la simulation. Ce registre en détermine la période sur 32 bits. La période est codée en demi-microsecondes. Une valeur de 20 donnera donc au système une granularité temporelle de 10 µs. Cette valeur n'influe pas sur la fréquence à laquelle les FPGAs sont cadencés.

Échantillonnage du bruit : Ce registre contrôle également une horloge. Cette dernière agit sur l'envoi des stimulations de bruit. Une série de stimulations est constituée d'une succession de temps d'attente entre chaque envoi. Ces temps d'attente sont des entiers qui codent un nombre de périodes à attendre. Le registre dont il est ici question code cette période. Il s'exprime en nombre de périodes du système.

Masque de requêtes de bruit : Ce registre code, bit à bit, quelles synapses seront soumise au bruit synaptique. L'écriture d'un bit à 1 signifie que la synapse recevra du bruit. La répartition des bits est donnée par le tableau D.3

description des zones mémoires

Nous abordons ici la description des zones mémoires qui recueillent les données brutes de simulation, contrairement aux registres qui la paramètrent.

Compensation : Cette zone permet de définir le courant de compensation de chaque neurone. l'adresse réelle à utiliser est 08000h$+4 \times n$ où n est le numéro du neurone concerné. Selon le circuit, l'accès s'effectue en une écriture de 12 bits (*Trieste*, $n < 8$) ou en deux accès de 7 bits (*Claudia*, $7 < n < 32$). Pour plus d'informations, voir DEF_COMP en page 148.

Paramètres numériques : Toute écriture dans cette zone détermine les paramètre numérique qui règlent le modèle de simulation du neurone. Bien que les paramètres concernent un circuit, il sont référencés par un numéro de neurone. Les paramètres sont alors envoyés vers le circuit qui contient le neurone concerné, et affecteront *tous* les neurones du circuit. Pour les circuits *Claudia*, il s'agit d'un entier codé sur 3 bits. Pour les circuits *Trieste*, le codage s'effectue sur 5 bits qui sont (respectivement, poids fort en premier) Vlcmd, I/E, Scmd, Cmd1, Cmd0.

Envoi direct : Toute donnée sur 15 bits écrite dans cette zone mémoire est envoyée directement sur la structure chaînée. Cette fonction est essentiellement utile pour le déverminage du système. La description des trammes est donnée en annexe C.

Mise à jour du bruit : Cette zone permet de remplir les trames de bruit synaptique à envoyer au circuit. Les trames sont codées comme une succession d'intervalles entre les envois de stimulation. L'adresse se code comme suit : 18000h $+8 \times n + 4 \times s$. n représente le numéro du neurone concerné et s vaut 0 pour une synapse inhibitrice, et 1 pour une synapse excitatrice.

Matrice de connections : Cette zone, la plus grande, permet de définir les poids synaptiques des différentes connections. L'adresse est 1C000h+$(64 \times pre + 2 \times post + syn) \times 4$ où *pre* et *post* sont les numéros respectifs de neurones présynaptique et postsynaptique, et *syn* est la nature de la connection : 0 pour inibitrice, 1 pour excitatrice.

Annexe E

Paramètres de configuration du circuit *Galway*

Les différents tableaux qui suivent commentent les différents paramètres nécéssaires pour configurer le circuit *Galway*. Nous commencerons par les paramètres numériques qui, principalement, autorisent, ou non, les branchement de chaque conductance sur les neurones ou la broche de visualisation.

E.1 Les paramètres numériques

Les paramètres numériques se présentent sous la forme d'un mot de 14 bits envoyé en série. L'interface série est décrite en page 111. Le premier décodage s'effectue sur les trois bits de poids faible. Il détermine vers quelle noyau de simulation doit être envoyé le mot. La répartition est présentée par le tableau E.1.

0	1	2	Unité concernée
0	0	0	Neurone 0 (FS)
0	0	1	Neurone 1 (RS)
0	1	0	Neurone 2 (RS)
0	1	1	Neurone 3 (RS)
1	0	0	Neurone 4 (5 cond.)
1	0	1	Conductance supplémentaire
1	1	x	Convertisseur numérique/analogique

TAB. E.1 – Description des bits de destination

Dans le cas du convertisseur analogique numérique, les données sont directement exploitées ; les bits 2 à 9 sont envoyés au registre de sortie qui commande la conversion.

Pour la conductance supplémentaire, les bits suivants suffisent pour définir l'ensemble des modes de fonctionnement. Ils sont présentés dans le tableau E.2.

Dans le cas d'une configuration où l'on demande un branchement à la fois vers les neurones 3 et 4, le neurone 4 est prioritaire. Outre le fait que cela ne signifie rien au niveau des modèles, il n'est pas possible d'associer une conductance à deux neurones à cause de la dépendance au potentiel de membrane.

Bit	Mode	si 0	si 1
5	Neurone 3	non reliée	reliée
6	Neurone 4	non reliée	reliée (prioritaire)
7	Visualisation		
8	Forme	$m \cdot h$	$m^2 \cdot h$

TAB. E.2 – Registre de la conductance supplémentaire

Le bit 8 permet de choisir le comportement temporel de la conductance. La cinétique d'activiation peut en effet être élevée au carré. Cela est nécessaire pour pouvoir modéliser un grande nombre de canaux.

E.1.1 Paramètres propres aux neurones

Dans le cas où les trois premiers bits du mot de configuration codent un neurone, les deux bits suivants (4 et 5) déterminent le registre à modifier. Le tableau E.3 donnent la correspondance entre le codage et le registre adressé.

3	4	Registre concerné
0	0	Conductances connectées sur la membrane
0	1	Conductances connectées sur la surtie de visualisation
1	0	Mode de calcul
1	1	*Inutilisé*

TAB. E.3 – Description des bits de registre relatif aux neurones

Deux registres sont utilisés pour déterminer l'usage qui est fait de chaque conductance (utilisée ou visualisée). Un dernier mot de configuration est nécessaire pour définir le mode de simulation du neurone. Le tableau E.4 détermine le canal associé à chaque bit de configuration.

Bit	conductance
5	Sodium
6	Potassium
7	Fuite
8	Stimulation
9	Modulation
A	Calcium
B	Synapse A
C	Synapse B
D	Synapse C

TAB. E.4 – Registre de conductance

Seul le neurone 4 possède l'ensemble des conductances présentées par le tableau E.4. Dans les autres cas, le bit considéré est sans effet si la conductance associée n'existe pas.

Finalement, le dernier registre à documenter est celui qui concerne le mode de simulation propre au neurone. Le tableau E.5 nous montre comment régler la puissance de l'activiation des conductances lentes.

Bit	Mode	si 0	si 1
5	Voltage clamp	activé	simulation normale
6	Forme modulation	$m \cdot h$	$m^2 \cdot h$
7	Forme Calcium	$m \cdot h$	$m^2 \cdot h$

TAB. E.5 – Registre de mode de calcul

La dernière fonction, le *voltage-clamp*, permet d'ouvrir la boucle de simulation du neurone. Cela signifie que le potentiel de membrane V_{mem} utilisé par les contuctances est défini par une entrée extérieure au lieu d'être mesuré aux bornes de la capacité de membrane (C_{mem}). Cette configuration permet d'extraire les paramètres du neurone. Elle peut également être employée pour effectuer des simulations avec des composants externes.

E.2 Les paramètres analogiques

L'interface permettant d'introduire les paramètres analogiques est décrite en page 112. Le tableau E.6 présente la signification de chacun des 205 paramètres analogiques du circuit. Ils sont numérotés de 0 à 204 sous forme hexadécimale, ce qui permet d'effectuer directement le lien entre le paramètre et une adresse mémoire qui lui correspond.

Signification du paramètre	Neurone 0	1	2	3	4	Cond. libre	
Na^+ -act- V_{pente}	00	1E	45	6C	93		
Na^+ -act- V_{offset}	01	1F	46	6D	94		
Na^+ -act- τ	02	20	47	6E	95		
Na^+ -inact- V_{pente}	03	21	48	6F	96		
Na^+ -inact- V_{offset}	04	22	49	70	97		
Na^+ -inact- τ	05	23	4A	71	98		
Na^+ - g_{max}	06	24	4B	72	99		
Na^+ - V_{equi}	07	25	4C	73	9A		
K^+ - V_{pente}	08	26	4D	74	9B		
K^+ - V_{offset}	09	27	4E	75	9C		
K^+ - τ	0A	28	4F	76	9D		
K^+ - g_{max}	0B	29	50	77	9E		
K^+ - V_{equi}	0C	2A	51	78	9F		
Fuite - g_{max} (1/2)	0D	2B	52	79	A0		
Fuite - g_{max} (2/2)	0E	2C	53	7A	A1		
Fuite - V_{equi}	0F	2D	54	7B	A2		
Ca^{2+} -act- V_{pente}		2E	55	7C	A3		
Ca^{2+} -act- V_{offset}		2F	56	7D	A4		

TAB. E.6: Correspondance entre les paramètres et leur numéro d'ordre (continué page suivante)

Signification du paramètre	Neurone 0	1	2	3	4	Cond. libre	
Ca^{2+} -act- τ		30	57	7E	A5		
Ca^{2+} -inact- V_{pente}		31	58	7F	A6		
Ca^{2+} -inact- V_{offset}		32	59	80	A7		
Ca^{2+} -inact- τ		33	5A	81	A8		
Ca^{2+} - g_{max} (1/2)		34	5B	82	A9		
Ca^{2+} - g_{max} (2/2)		35	5C	83	AA		
Ca^{2+} - V_{equi}		36	5D	84	AB		
mod. -act- V_{pente}					AC	C3	
mod. -act- V_{offset}					AD	C4	
mod. -act- τ					AE	C5	
mod. -inact- V_{pente}					AF	C6	
mod. -inact- V_{offset}					B0	C7	
mod. -inact- τ					B1	C8	
mod. - g_{max} (1/2)					B2	C9	
mod. - g_{max} (2/2)					B3	CA	
mod. - V_{equi}					B4	CB	
Syn. A - τ	10	37	5E	85	B5		
Syn. A - g_{max} (1/2)	11	38	5F	86	B6		
Syn. A - g_{max} (2/2)	12	39	60	87	B7		
Syn. A - V_{equi}	13	3A	61	88	B8		
Syn. B - τ	14	3B	62	89	B9		
Syn. B - g_{max} (1/2)	15	3C	63	8A	BA		
Syn. B - g_{max} (2/2)	16	3D	64	8B	BB		
Syn. B - V_{equi}	17	3E	65	8C	BC		
Syn. C - τ	18	3F	66	8D	BD		
Syn. C - g_{max} (1/2)	19	40	67	8E	BE		
Syn. C - g_{max} (2/2)	1A	41	68	8F	BF		
Syn. C - V_{equi}	1B	42	69	90	C0		
Stimulation (1/2)	1C	43	6A	91	C1		
Stimulation (2/2)	1D	44	6B	92	C2		
Caractérisation							CC

TAB. E.6: Correspondance entre les paramètres et leur numéro d'ordre

Résumé

Dans un contexte où l'usage de circuits neuromimétiques se généralise au sein des neurosciences, nous étudions ici leur intégration au sein de réseaux adaptatifs. Les circuits mis en oeuvre se basent sur un modèle proche de la biologie résolu en continu et en temps réel. Les calculs relatifs à l'adaptation du réseau sont réalisés en numérique temps réel, logiciel et/ou matériel. La partie logicielle est assurée par un ordinateur interfacé à travers le bus PCI, tandis que la partie matérielle utilise des FPGAs. Trois générations sont présentées avec une analyse critique sur leur utilisation comme système de simulation de réseau neuronal.

Mots-clés

Réseaux de neurones, plasticité synaptique, circuits intégrés neuromimétiques, circuits intégrés mixtes analogiques-numériques, neurones artificiels, modèle Hodgkin-Huxley.

Abstract

This work addresses the integration issues for neuromimetic integrated circuits dedicated to computationnal neuroscience, in the context of adaptive neural networks. Those circuits implement a biologically realistic model. This model is continuously solved in real time. The network adaptation is also computed in real time by digital means using software and/or hardware material. The software part of the system is handled by a computer, interfaced to the system through the PCI bus, whereas the hardware part uses FPGAs. Three generations of the simulation system are presented, followed by a critical analysis on their performance as analog neural network simulators.

Keywords

Neural networks, synaptic plasticity, neuromimetic integrated circuits, mixed analog-digital integrated circuits, artificial neurons, Hodgkin-Huxley models.