UNIVERSITÉ DE MONTRÉAL

CONCEPTION ET RÉALISATION D'UN PROTOTYPE DE LA PARTIE IMPLANTABLE D'UN STIMULATEUR VISUEL CORTICAL

MARTIN ROY

DÉPARTEMENT DE GÉNIE ÉLECTRIQUE ET DE GÉNIE INFORMATIQUE

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION

DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES

(GÉNIE ÉLECTRIQUE)

NOVEMBRE 1999

©Martin Roy, 1999.

National Library of Canada
Bibliothèque nationale du Canada

Acquisitions and Bibliographic Services
Acquisitions et services bibliographiques

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-48870-5

Canada

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

CONCEPTION ET RÉALISATION D'UN PROTOTYPE

DE LA PARTIE IMPLANTABLE D'UN STIMULATEUR VISUEL CORTICAL

présenté par : ROY Martin

en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées

a été dûment accepté par le jury d'examen constitué de :

M. SAVARIA Yvon, Ph.D., président

M. SAWAN Mohamad, Ph.D., membre et directeur de recherche

M. BRAULT Jean-Jules, Ph.D., membre

REMERCIEMENTS

Je tiens premièrement à remercier mon directeur de recherche, Mohamad Sawan, pour m'avoir proposé un sujet de recherche passionnant et ensuite me guider durant l'accomplissement de mes travaux.

Un grand merci à Gaétan Décarie pour sa contribution dans le projet et pour ses judicieux conseils.

Je tiens à saluer mes collègues de l'équipe de recherche PolySTIM pour leur bonne compagnie et pour leur aide.

Un merci spécial à ma compagne, Guylaine Renaud, pour m'avoir épaulé tous le long de mes études.

Finalement, un grand merci à tous mes amis, à ma famille et particulièrement à mes parents, Pierre Roy et Monique Denis pour leur aide et leur encouragement.

RÉSUMÉ

Les récents progrès dans le domaine de la microélectronique ont permis la réalisation de stimulateurs physiologiques de plus en plus complexes et performants. Parmi ces progrès, la miniaturisation des circuits intégrés et la diminution de la puissance requise pour leur fonctionnement laissent entrevoir la réalisation de nouvelles avenues dans le domaine de la stimulation électrique. L'une de ces avenues consiste à stimuler le cortex visuel d'un non-voyant afin de lui redonner une vision partielle. En effet, les expériences de plusieurs chercheurs ont permis de constater qu'il est possible de faire apparaître un petit point lumineux, appelé phosphène, dans le champ visuel d'un non-voyant en lui injectant des courants de stimulation dans son cortex visuel. Il semble donc possible de réaliser un système permettant de produire une sensation d'image pixelisée chez les non-voyants en stimulant plusieurs endroits de leur cortex visuel. Un tel système comportera un contrôleur externe pour l'acquisition et l'amélioration des images et un implant qui, selon les données transmises par le contrôleur externe, injectera les courants de stimulation.

Le but de ce mémoire de maîtrise est de réaliser un prototype d'implant visuel cortical (PIVC). Le prototype, qui possède la même architecture que l'implant cortical anticipé, est réalisé à partir de composants discrets et comporte une matrice de diodes électroluminescentes (DEL) permettant de visualiser les « images » obtenues suite à la réception de données du contrôleur externe. Le PIVC n'est donc pas utilisé pour stimuler le cortex visuel, mais employé pour donner une idée de l'image perçue par le non-voyant suite à une opération de stimulation. Le prototype permet donc de valider une architecture d'implant mais pourra également servir comme outil de développement car la principale composante du prototype est reconfigurable.

Deux architectures de prototype de l'implant furent développées au cours du projet mais seule la première fût utilisée pour réaliser un prototype complet du stimulateur. La différence entre les deux architectures se situe au niveau du circuit de conversion numérique analogique qui contrôle les intensités des courants de stimulation. La première architecture utilise un circuit intégré et des composants discrets commerciaux pour réaliser cette fonction alors que la deuxième architecture utilise un circuit intégré prédiffusé. La partie analogique de la deuxième architecture comprenant un convertisseur numérique analogique et des échantillonneurs-bloqueurs fonctionnant en mode courant est donc entièrement intégrée à l'intérieur d'une seule puce. Les résultats de simulation indiquent que la topologie du circuit prédiffusé est fonctionnelle. Toutefois, la technologie choisie ne permet pas une grande densité d'interconnexion (une seule couche de métal) ce qui a empêché le bon fonctionnement du circuit de la deuxième architecture.

Cependant, les résultats obtenus de la réalisation de la première architecture du prototype ont donnés lieu au fonctionnement attendu. Ainsi il est possible de faire afficher des images sur la matrice de diodes électroluminescentes suite à la réception des données provenant du contrôleur externe. Ces données correspondent aux paramètres de stimulation qui permettent de moduler l'intensité lumineuse des DEL. Ces paramètres sont les amplitudes des courants et les durées de phase et d'interphase des opérations de stimulation. Le protocole de communication établit entre le contrôleur externe et le prototype permet, suite à une étape de configuration, de définir un taux de transfert variable de l'information selon les besoins. Cette flexibilité est requise par cette application car les paramètres idéaux pour obtenir une stimulation efficace ne sont pas encore connues. Les images obtenues sur la matrice de DEL laissent croire qu'un implant visuel cortical comportant essentiellement la même architecture que le prototype permettrait à un non-voyant de lire et de reconnaître des objets facilement même si le nombre de pixels de l'image est relativement faible.

ABSTRACT

With the recent advances in microelectronics, it is now possible to fabricate physiological stimulator devices which are more complex and efficient than ever. Among theses advances, the miniaturisation and power consumption diminution of integrated circuit devices are some of the characteristics that allow to anticipate new avenues in electrical stimulations.

The stimulation of the visual cortex of a blind person in order to give him back a partial sight is one of these avenues. Several research experiments have shown that it is possible to induce the perception of a small spot of light called phosphene, into the visual field of the blind, following the injection of stimulation currents into his visual cortex. It seems that it is now possible to design a system that allows to produce pixelized image sensation to the blind following the stimulation of several sites into his visual cortex. Such a system will have an external controller for image acquisition and improvement and an implant to produce stimulation currents.

The main goal of the project is to design and fabricate a cortical visual implant prototype (CVIP). The prototype has the same architecture as the envisioned implant, which is realised from discrete electronics components and contains a LED array that allows the visualisation of obtained images following the reception of visual data from the external controller. The CVIP is not used for visual cortex stimulation but to give an idea of what a blind person would see after a stimulation operation. The prototype allows the validation of the anticipated implant architecture and can be used as a development tool because the main component is reconfigurable.

Two architectures have been designed but only the first one has been used to prototype a complete visual stimulator. The difference between the architectures is the digital to

analogue conversion circuit used to control the intensities of stimulation currents. The first architecture uses a commercial integrated circuit and several discrete components to make the conversion, while the second architecture uses a bipolar array integrated circuit. The analog part of the second architecture, composed of a digital to analog converter and several current mode samples and holds, is then entirely integrated into a single chip.

The simulation results indicates the full operation of the bipolar array integrated circuit topology, but the selected technology doesn't allow more than one interconnection layer which gave high integrated complexity and prevent the circuit of the second architecture of the prototype to operate properly.

However, results obtained following the fabrication of the first architecture of the prototype show the expected operation of the circuit. Thus, it is possible to display images on the LED array following the reception of data from the external controller. These data are the stimulation parameters that allow to modulate the light intensities of LEDs. The parameters are the amplitudes of the currents and the phase and interphase delays of the stimulation operations. The established communication protocol between the external controller and the prototype allows, following a configuration step, to define a variable transfer rate of the information. This flexibility is required by this application since the ideal parameters to obtain efficient stimulation results are not known.

The obtained images on the LED array let us believe that a cortical visual implant that has essentially the same architecture as the proposed prototype would allow, for a blind person, to read and recognize objects easily even if the number of image pixels is relatively low.

TABLE DES MATIÈRES

REMERCIEMENTS ..... **iv**

RÉSUMÉ ..... **v**

ABSTRACT ..... **vii**

TABLE DES MATIÈRES ..... **ix**

LISTE DES TABLEAUX ..... **xiii**

LISTE DES FIGURES ..... **xiv**

LISTE DES ANNEXES ..... **xix**

LISTE DES SIGLES ET ABRÉVIATIONS ..... **xx**

INTRODUCTION **1**

CHAPITRE 1 : ÉTUDE DES STIMULATEURS PHYSIOLOGIQUES **5**

1.1 Introduction 5

1.2 La cellule nerveuse 6

1.3 Classification des stimulateurs électriques 8

1.4 Les technologies 11

1.5 Les techniques analogiques 12

1.5.1 Transmission et réception de données et d'énergie 13

1.5.2 Convertisseur numérique analogique (CNA) 17

1.6 Electrodes et réseaux d'électrodes 19

1.6.1 Modélisation de l'interface électrode-tissu 20

1.7 Considérations d'ordre biomédicales 21

1.7.1 Biocompatibilité des implants 22

1.7.2 Excitabilité et lésion neuronale 22

1.8 Conclusion 23

CHAPITRE 2 : LES PROTHÈSES VISUELLES 24

2.1 Introduction 24

2.2 Le système visuel humain et les maladies de la vision 24

2.3 Les avenues de recherche pour récupérer partiellement la vision 27

2.3.1 L'implant rétinien 27

2.3.2 Le stimulateur du nerf optique 29

2.3.3 L'implant cortical 29

2.3.4 Comparaison entre les différentes avenues de recherche 30

2.4 Les connaissances actuelles sur la stimulation du cortex visuel 32

2.4.1 Bref historique 32

2.4.2 Les travaux de l'équipe de Hambretch 33

2.5 Architecture et travaux de pointe sur l'implant cortical 35

2.5.1 La première version du stimulateur visuel miniaturisé de PolySTIM 36

2.5.2 Le démultiplexeur physiologique de Jones et Normann 38

2.5.3 L'implant de Troyk 40

2.6 Efficacité d'une vision obtenue à l'aide d'un implant cortical 40

2.7 Conclusion 41

CHAPITRE 3 : CONCEPTION MATÉRIELLE DU PROTOTYPE **43**

3.1 Introduction 43

3.2 Élaboration de la structure de base des prototypes 44

3.3 Structure du prototype du stimulateur visuel cortical (PSVC) 46

3.4 Architecture de la première version du PIVC 47

3.4.1 Décodeur Manchester 48

3.4.2 Sélection d'une DEL et contrôle de l'intensité du courant 50

3.5 Architecture de la deuxième version du PIVC 52

3.5.1 Principe de fonctionnement du CEB 53

3.5.2 L'échantillonneur-bloqueur du CEB 55

3.5.3 Le commutateur de courant ITA 57

3.5.4 Le CNA du CEB 59

3.6 Conclusion 61

CHAPITRE 4 : LE CONTRÔLEUR DU PIVC **63**

4.1 Introduction 63

4.2 Les modes de fonctionnement 63

4.3 Le protocole de communication 65

4.3.1 Le mot de configuration 66

4.3.2 Le mot de paramètres 69

4.4 Architecture générale du contrôleur du PIVC 70

4.4.1 Le contrôleur principal (CPIVC) 72

4.4.2 Le module de formation des commandes de stimulation 74

4.4.3 Les canaux de stimulation 75

4.5 Conclusion 77

CHAPITRE 5 : RÉALISATION ET RÉSULTATS 78

5.1 Introduction 78

5.2 Réalisation du CEB 78

5.3 Simulation du CEB 80

5.3.1 L'amplificateur opérationnel 80

5.3.2 Le CNA et le miroir de courant 82

5.3.3 L'échantillonneur-bloqueur (EB) 83

5.3.4 Caractéristiques globales du CEB 85

5.3.5 Précision des simulations et causes de défectuosité 87

5.4 Réalisation du PIVC 87

5.5 Résultats de simulation du contrôleur du PIVC 90

5.6 Vérification de la fonctionnalité du prototype 94

5.7 Conclusion 95

CONCLUSION 97

BIBLIOGRAPHIE 101

ANNEXES 111

LISTE DES TABLEAUX

Tableau 2.1 : Historique de la stimulation du cortex visuel 32

Tableau 4.1 : Description des champs des registres de configuration 68

Tableau 5.1 : Caractéristiques simulées de l'amplificateur opérationnel 81

Tableau 5.2 : Caractéristiques simulées du CNA avec et sans le miroir de courant 82

Tableau 5.3 : Caractéristiques simulées d'un EB 85

Tableau 5.4 : Caractéristiques simulées du CEB 86

Tableau A.1 : Liste des broches spéciales et d'alimentation du CPLD 121

Tableau A.2 : Liste des entrées du contrôleur et broches associées 124

Tableau A.3 : Liste des sorties du contrôleur et broches associées 125

Tableau A.4 : Liste des points de test du contrôleur et broches associées 129

Tableau D.1 : Liste des fichiers de la description VHDL du contrôleur du PIVC 147

Tableau G.1 : Contenu des registres de configuration 213

Tableau G.2 : Description des types de test 215

LISTE DES FIGURES

Figure 1.1 : La cellule nerveuse 7

Figure 1.2 : Schéma bloc typique d'un stimulateur multicanaux 10

Figure 1.3 : Circuit simplifié d'un lien inductif 14

Figure 1.4 : Récepteur et modulateur LSK 15

Figure 1.5 : Codage Manchester 16

Figure 1.6 : CNA à miroirs de courant simple 17

Figure 1.7 : CNA en BiCMOS 18

Figure 1.8 : CNA proposé par St-Amand [62] 19

Figure 1.9 : Contrôle du sens du courant 19

Figure 1.10 : Réseau d'électrodes (tiré de [33]) 20

Figure 1.11 : Modèle de l'interface électrode-électrolyte 21

Figure 2.1 : Vue en coupe d'un oeil 25

Figure 2.2 : Le système visuel humain 26

Figure 2.3 : Implant rétinien (tiré de [8]) 28

Figure 2.4 : Stimulateur du nerf optique (tiré de [10]) 29

Figure 2.5 : Implant cortical (adapté de [14]) 30

Figure 2.6 : Formes d'onde utilisées pour la stimulation intracorticale 34

Figure 2.7 : Stimulateur visuel miniaturisé de Boyer [52] 37

Figure 2.8 : Structure générale du CI de Jones et Normann [39] 38

Figure 2.9 : Structure d'un canal de stimulation (DAC control element) 39

Figure 3.1 : Prototype du stimulateur visuel cortical 46

Figure 3.2 : Schéma bloc de la première architecture du PIVC 47

Figure 3.3 : Décodeur Manchester (tiré de [80]) 49

Figure 3.4 : Exemple de formes d'onde des E/S du décodeur Manchester 50

Figure 3.5 : Sélection d'une DEL et contrôle de l'intensité de son courant 51

Figure 3.6 : Schéma bloc de la deuxième architecture du PIVC 52

Figure 3.7 : Schéma bloc simplifié du CEB 54

Figure 3.8 : Miroir de courant dynamique 55

Figure 3.9 : Échantillonneur- bloqueur en mode courant 56

Figure 3.10 : Schéma bloc du commutateur ITA 58

Figure 3.11 : Circuit d'un commutateur de courant 58

Figure 3.12 : Schéma simplifié du convertisseur numérique-analogique 60

Figure 3.13 : Schéma d'un commutateur utilisé dans le CNA 61

Figure 4.1 : Format d'une trame de données 65

Figure 4.2 : Disposition des bits dans une trame de données 66

Figure 4.3 : Format des mots de configuration 67

Figure 4.4 : Contenu des registres de configuration 67

Figure 4.5 : Composition d'un mot de paramètres (MPMT) 69

Figure 4.6 : Schéma bloc du circuit numérique de contrôle du PIVC 71

Figure 4.7 : Schéma bloc simplifié du CPIVC 72

Figure 4.8: Schéma bloc du module de formation des commandes de stimulation 75

Figure 4.9 : Schéma bloc d'un canal de stimulation 76

Figure 5.1 : Photographie du CEB 79

Figure 5.2 : Opération (échantillonnage/maintient) d'un EB 83

Figure 5.3 :Dérive du courant d'un EB en mode maintient 84

Figure 5.4 : Réponses simulée et idéale du CEB (mode maintient) 86

Figure 5.5 : Principaux éléments de chaque carte du prototype 88

Figure 5.6 : Photographie du circuit imprimé de la carte maîtresse 89

Figure 5.7 : Photographie du circuit imprimé de la carte affichage 89

Figure 5.8 : Simulation du contrôleur (intervalle de temps : 0 - 76 us) 91

Figure 5.9 : Simulation du contrôleur (intervalle de temps : 71.2us – 80 us) 93

Figure A.1 : Circuit de la carte maîtresse (1/3) 113

Figure A.2 : Circuit de la carte maîtresse (2/3) 114

Figure A.3 : Circuit de la carte maîtresse (3/3) 115

Figure A.4 : Circuit de la carte affichage (1/5) 116

Figure A.5 : Circuit de la carte affichage (2/5) 117

Figure A.6 : Circuit de la carte affichage (3/5) 118

Figure A.7 : Circuit de la carte affichage (4/5) 119

Figure A.8 : Circuit de la carte affichage (5/5) 120

Figure B.1 : Schéma bloc du CEB 131

Figure B.2 : Structure interne d'un EB 132

Figure B.3 : Source de courant d'un EB 132

Figure B.4 : Échantillonneur-bloqueur du EB 133

Figure B.5 : Miroir de courant (type P) 134

Figure B.6 : Convertisseur numérique-analogique (CNA) 135

Figure B.7 : Amplificateur opérationnel utilisé par le CNA 136

Figure B.8 : Schéma bloc du commutateur de courant 137

Figure B.9 : Circuit de contrôle du commutateur de courant 138

Figure B.10 : Circuit du commutateur de courant 139

Figure B.11 : Dessin du masque de métal du CEB 140

Figure C.1 : Réponse en fréquence de l'amplificateur opérationnel 141

Figure C.2 : Gain en mode commun de l'amplificateur opérationnel 141

Figure C.3 : Gain Vsortie/Valim pour mesure du PSSR du OPAMP 142

Figure C.4 : Réponses théorique et simulée du CNA 142

Figure C.5 : Non-linéarité intégrale (INL) du CNA 143

Figure C.6 : Non-linéarité différentielle (DNL) du CNA 143

Figure C.7 : Réponse théorique (-) et simulé (..) du CNA avec miroir de courant 144

Figure C.8 : Non-linéarité intégrale du CNA avec miroir de courant 144

Figure C.9 : Non-linéarité différentielle du CNA avec miroir de courant 145

Figure C.10 : INL du CEB au début du mode maintient 145

Figure C.11 : DNL du CEB au début du mode maintient 146

Figure E.1 : Circuit imprimé de la carte maîtresse (1/6) 195

Figure E.2 : Circuit imprimé de la carte maîtresse (2/6) 196

Figure E.3 : Circuit imprimé de la carte maîtresse (3/6) 197

Figure E.4 : Circuit imprimé de la carte maîtresse (4/6) 198

Figure E.5 : Circuit imprimé de la carte maîtresse (5/6) 199

Figure E.6 : Circuit imprimé de la carte maîtresse (6/6) 200

Figure E.7 : Circuit imprimé de la carte affichage (1/3) 201

Figure E.8 : Circuit imprimé de la carte affichage (2/3) 202

Figure E.9 : Circuit imprimé de la carte affichage (3/3) 203

Figure F.1 : Simulation du correcteur de Hamming 205

Figure F.2 : Simulation du fonctionnement du CPIVC (1/4) 206

Figure F.3 : Simulation du fonctionnement du CPIVC (2/4) 207

Figure F.4 : Simulation du fonctionnement du CPIVC (3/4) 208

Figure F.5 : Simulation du fonctionnement du CPIVC (4/4) 209

Figure F.6 : Simulation du fonctionnement d'un canal de stimulation (1/2) 210

Figure F.7 : Simulation du fonctionnement d'un canal de stimulation (2/2) 211

LISTE DES ANNEXES

Annexe A : Détails de conception et circuit du PIVC 111

Annexe B : Circuit complet du CEB 130

Annexe C : Résultats de simulation du CEB 141

Annexe D : Description VHDL du contrôleur du PIVC 147

Annexe E : Schémas des circuits imprimés du prototype 194

Annexe F : Résultats de simulation du contrôleur 204

Annexe G : Le banc de test 212

LISTE DES SIGLES ET ABRÉVIATIONS

ACS	Adressage des canaux et des sites
AM	Amplitude Modulation
AMP	Amplitude
AMPC	Amplitude commune
ASK	Amplitude Shift Keying
BiCMOS	Bipolar and CMOS
BTHS	Base de temps pour l'horloge de stimulation
CCD	Charge Coupled Device
CE	Contrôleur externe
CEB	CNA et échantillonneur-bloqueur
CI	Circuit intégré
CMOS	Complementary Metal Oxide Semiconductor
CNA	Convertisseur numérique-analogique
CPLD	Complex Programmable Logic Device
DAC	Digital to analog converter
DEL	Diode électroluminescente
DIC	Durée d'interphase commune
DIP	Durée d'interphase
DIT	Durée inter-train
DNL	Differential Nonlinearity

DP	Durée de phase
DPC	Durée de phase commune
DT	Durée d'un train
EB	Échantillonneur-Bloqueur
E/S	Entrées/sorties
FA	Fanion amplitude
FCS	Fanion canal et site
FDI	Fanion durée d'interphase
FDP	Fanion durée de phase
FSK	Frequency Shift Keying
INL	Integral Nonlinearity
IVC	Implant visuel cortical
LCMSP	Longueur d'une chaîne de mot de stimulation partiel
LSK	Load Shift Keying
MOS	Metal Oxide Semiconductor
MSA	Machine séquentielle algorithmique
NBA	Nombre de bit d'amplitude
NBDI	Nombre de bit de durée d'interphase
NBDP	Nombre de bit de durée de phase
NMOS	N-channel Metal Oxide Semiconductor
PIVC	Prototype de l'implant visuel cortical
PSK	Phase Shift Keying

PSVC	Prototype du stimulateur visuel cortical
RAM	Random Access Memory
RC1	Registre de configuration du PIVC #1
RC2	Registre de configuration du PIVC #2
RC3	Registre de configuration du PIVC #3
RC4	Registre de configuration du PIVC #4
RF	Radio fréquence
SIDNE	Stimulation-Induced Depression of Neuronal Excitability
SCM	Société canadienne de microélectronique
SVC	Stimulateur visuel cortical

INTRODUCTION

La dernière étude de Statistique Canada concernant le nombre de personnes souffrant d'une déficience visuelle au Canada remonte à 1986. On estimait alors ce nombre à 581110 personnes dont 51005 étaient légalement aveugles [1]. Une personne est légalement aveugle lorsque l'acuité de la vision centrale de son meilleur œil est égale ou inférieure à 20/200. Aux États-Unis, le nombre d'hommes et de femmes légalement aveugles, âgés de 40 ans et plus fut estimé à 775796 en 1990 [2]. La plupart des personnes atteintes d'une déficience visuelle conservent cependant une vision partielle. Aux États-Unis, environ 10% des personnes légalement aveugles ont une cécité totale c'est-à-dire qu'elles ne perçoivent aucune sensation de vision [2].

La génétique moléculaire devrait permettre un jour de limiter l'expansion de certaines maladies et même de restaurer en partie la vision, mais actuellement, la plupart de ces maladies sont irréversibles, donc lorsqu'une personne devient aveugle, elle le demeure pour le reste de sa vie [3]. Plusieurs moyens, comme le Braille pour la lecture ou bien l'utilisation d'un chien guide pour les déplacements, sont à la disposition des personnes non-voyantes afin d'augmenter leur indépendance. Par ailleurs, des systèmes de réhabilitation par substitution développés au cours des dernières années permettent de donner une certaine assistance à ces personnes. Ces dispositifs captent l'information visuelle, la traite et finalement la dirige vers un organe sensoriel qui est intact comme le système auditoire par exemple [77]. Ces moyens et ces systèmes, bien qu'utiles, ne permettent cependant pas aux gens aveugles d'effectuer certaines fonctions aussi efficacement et rapidement que s'ils possédaient une certaine forme de vision. La prothèse visuelle (ou stimulateur visuel) constitue une nouvelle alternative qui devrait permettre dans un proche avenir à redonner une vision partielle à la majorité des non-voyants.

À ce jour, il existe essentiellement trois grandes avenues de recherche pour rétablir partiellement la vision chez un non-voyant : la stimulation de la rétine, la stimulation du nerf optique et la stimulation du cortex visuel. De façon très générale et quelle que soit l'approche utilisée, le système de stimulation doit comporter un module externe et un module interne (implanté dans le corps humain). Les principales tâches de la partie externe son de capter des images et d'effectuer un certain traitement sur celles-ci. L'information résultante est alors envoyée à l'implant. Celui-ci est chargé d'injecter des courants de stimulation à un endroit particulier du système visuel humain (rétine, nerf optique ou cortex visuel).

La stimulation du cortex visuel semble être l'approche la plus prometteuse à bien des égards. En effet, elle seule jusqu'à maintenant a fait l'objet d'expériences chez les humains. Les résultats sont encourageants et laissent entrevoir la possibilité que cette approche puisse être réalisable. L'injection de courants de stimulation dans le cortex visuel d'un non-voyant, produit de petits points lumineux (appelés phosphènes) dans son champ visuel. De plus, il semble y avoir un lien direct entre la position d'un phosphène dans le champ visuel et l'endroit où la stimulation est appliquée dans le cortex visuel [16,22,23]. L'activation de plusieurs phosphènes adjacents permettrait donc de créer une image. Chaque phosphène correspondant à un pixel de l'image.

Un des projets de l'équipe de recherche PolySTIM de l'École Polytechnique, consiste à concevoir et à réaliser un stimulateur visuel cortical (SVC). Le développement d'un système comme celui-ci demande un effort de recherche dans des directions qui touchent plusieurs disciplines comme le génie physique, le génie biomédical, le génie électrique et la médecine.

Quelques obstacles techniques doivent être franchis avant de pouvoir réaliser ce genre de système. Par exemple, pour des raisons de biocompatibilité, les liens physiques (fils) entre les deux modules doivent absolument être évités. Cette contrainte nécessite l'utilisation d'un lien inductif pour la transmission d'énergie et d'information du module externe vers l'implant, ce qui implique l'existence d'un seul et unique canal de

transmission sériel. L'implant doit alors être en mesure de récupérer l'information et ensuite d'effectuer des opérations d'aiguillages des données reçues vers les canaux de stimulation. Le circuit de l'implant possède donc implicitement une certaine complexité. De plus, le processus de développement de son architecture doit tenir compte de plusieurs autres éléments qui seront décrits dans les prochains chapitres. Il est ainsi préférable de passer par une étape de prototypage du système avant sa réalisation définitive [85].

Le but du présent projet de maîtrise est de concevoir et réaliser un prototype d'un implant visuel cortical (PIVC), donc de la partie interne du stimulateur. Jumelé avec le contrôleur externe, la fonctionnalité du stimulateur visuel cortical en entier pourra être testée et vérifiée avant son intégration finale. Le but du prototype n'est pas de stimuler des cellules nerveuses, mais bien de donner un aperçu de ce qu'une personne non-voyante peut voir avec un stimulateur cortical. Pour ce faire, le prototype comporte une matrice de diodes électroluminescentes (DEL) qui permettent de simuler les sensations des phosphènes perçues par un non-voyant, suite à des opérations de stimulation.

Un seul prototype est fabriqué mais nous proposons dans ce mémoire deux architectures permettant d'implanter la fonctionnalité d'un implant visuel cortical. La première (qui est fabriquée) est composée uniquement de composants et circuits intégrés commerciaux. La deuxième, semblable à la première, utilise également des composants et circuits intégrés commerciaux, mais possède en plus un circuit intégré prédiffusé qui contient une partie des circuits analogiques. La deuxième architecture n'a pu être fabriquée suite à un défaut de fabrication du circuit prédiffusé. Dans chaque cas, le circuit numérique de contrôle est implanté dans un circuit intégré programmable (CPLD). La partie numérique du prototype est donc facilement modifiable en partie ou en totalité.

La réalisation du prototype permet d'atteindre deux objectifs. Premièrement, il valide une nouvelle architecture d'implant visuel cortical qui favorise la flexibilité dans la communication des données avec le contrôleur externe. Deuxièmement, le prototype

peut servir d'outil de développement car sa partie numérique est entièrement reconfigurable. Ainsi, advenant une évolution des connaissances sur la stimulation corticale, un nouveau circuit numérique de contrôle pourra être testé rapidement grâce au prototype.

Le premier chapitre est une introduction aux stimulateurs physiologiques. On y retrouve principalement une description d'un système de classement des stimulateurs, ainsi que leurs principales caractéristiques physiques.

Le chapitre 2 traite des stimulateurs visuels et débute par une brève description du système visuel humain. Les trois approches de restauration de la vision sont ensuite approfondies. Finalement, on y présente les connaissances actuelles sur la stimulation du cortex visuel et de ses effets, ainsi que quelques architectures d'implant visuel cortical proposées par certains chercheurs.

Le chapitre 3 est consacré au développement des deux architectures. On y retrouve les éléments de leurs circuits analogiques et les interactions entre leurs différents modules. Les détails du circuit prédiffusé sont également traités.

Le chapitre 4 présente le circuit numérique de contrôle du prototype. Nous proposons d'abord un protocole de communication entre le contrôleur externe et le prototype. Ensuite nous développons le circuit numérique en entier.

Le chapitre 5 traite des résultats et de certains aspects par rapport aux réalisations du prototype et du prédiffusé. On approfondit notre compréhension du fonctionnement du prototype grâce notamment à des figures illustrant des résultats de simulation.

CHAPITRE 1

ÉTUDE DES STIMULATEURS PHYSIOLOGIQUES

1.1 Introduction

La stimulation physiologique est un sujet de recherche intéressant, car les progrès réalisés dans ce domaine ont un impact direct sur la qualité de vie de nombreuses personnes et les applications qui en découlent n'ont cessé de croître au fil des années. À ce jour, essentiellement trois méthodes existent pour créer un changement du comportement des cellules du système nerveux. La première méthode consiste à injecter une substance chimique à un groupe de cellules. Cette substance permet de modifier l'activité synaptique grâce à ses propriétés d'excitation ou d'inhibition [55]. La deuxième méthode utilise un stimulateur magnétique qui, par induction électromagnétique, permet de créer un champ électrique dans une fibre nerveuse dans le but de provoquer un potentiel d'action [27,28]. Un potentiel d'action correspond à une impulsion électrique qui se propage le long de l'axone d'une cellule nerveuse. La dernière méthode, la stimulation électrique, emploie des électrodes afin de transmettre un courant ionique à une fibre nerveuse et donc de produire un potentiel d'action [24].

Récemment, Chen et al. [55] ont développé et testé in vivo une sonde très compacte qui permet d'introduire une petite quantité de liquide à un endroit bien précis dans le corps. La méthode de stimulation recourant aux substances chimiques demeure au stade des activités de recherche dans le domaine de la neurobiologie. Aucune application permettant de récupérer une fonction chez l'humain n'utilise cette approche. Ceci est en partie dû au fait que l'injection d'un liquide est un processus plutôt lent.

La stimulation magnétique est surtout employée pour le diagnostic de maladie neuro-musculaire et pour la cartographie du cerveau [78]. Malgré le fait qu'au moins une étude

ait démontré des résultats encourageants, il n'y a, pour l'instant, aucune application qui permet de rétablir une fonction chez l'humain en utilisant cette approche [28]. Cette méthode offre comme avantage, par rapport à la stimulation électrique, une meilleure biocompatibilité, grâce à l'absence de contact métal–tissu (pas d'électrodes) [28]. La stimulation électrique reste néanmoins le meilleur moyen connu pour récupérer une quelconque fonction chez l'humain, car elle possède la précision requise par la majorité des applications.

Ce chapitre débute par une brève description de la cellule nerveuse, du phénomène du potentiel d'action et de l'interaction entre le stimulateur et la cellule. Dans un deuxième temps, on présente une classification des stimulateurs électriques permettant ainsi de cibler le type le plus convenable à la réalisation d'une prothèse visuelle. Ensuite, les technologies de fabrication de ces stimulateurs ainsi que quelques-unes de leurs caractéristiques techniques seront abordées. Les opérations de stimulation et leurs effets biologiques sont des facteurs qui doivent être absolument tenus en compte lors de la conception d'un implant. Ces particularités d'ordre biomédical seront donc considérées subséquemment.

1.2 La cellule nerveuse

Le travail effectué par une cellule nerveuse consiste à intégrer l'information provenant des cellules qui lui sont rattachées et ensuite, d'envoyer le résultat (une fois l'information traitée) vers d'autres cellules sous forme d'impulsions électriques. La figure 1.1 montre la structure simplifiée d'une cellule nerveuse. Celle-ci est principalement composée de dendrites qui reçoivent les impulsions des cellules avoisinantes, d'un corps cellulaire, d'un noyau et finalement de l'axone [4]. La jonction entre l'axone d'une cellule et les dendrites d'une autre est appelée synapse.

Une impulsion électrique appelée potentiel d'action peut être transmise à travers l'axone d'une cellule nerveuse, grâce à la propriété électrochimique de la membrane nerveuse. Celle-ci possède des pompes ioniques qui lui permettent de maintenir une différence de potentiel d'environ –65 mV entre le milieu intérieur et le milieu extérieur de la cellule.

Une dépolarisation de la membrane nerveuse se crée lors de l'arrivée de potentiels d'action de neurones voisins. Cette dépolarisation est obtenue grâce à un neuromédiateur (provenant des synapses) qui ouvre des canaux ioniques sur la membrane. Cette dépolarisation augmente la perméabilité de la membrane aux ions de sodium ce qui entraîne une rupture des équilibres ioniques et la création d'un potentiel d'action.

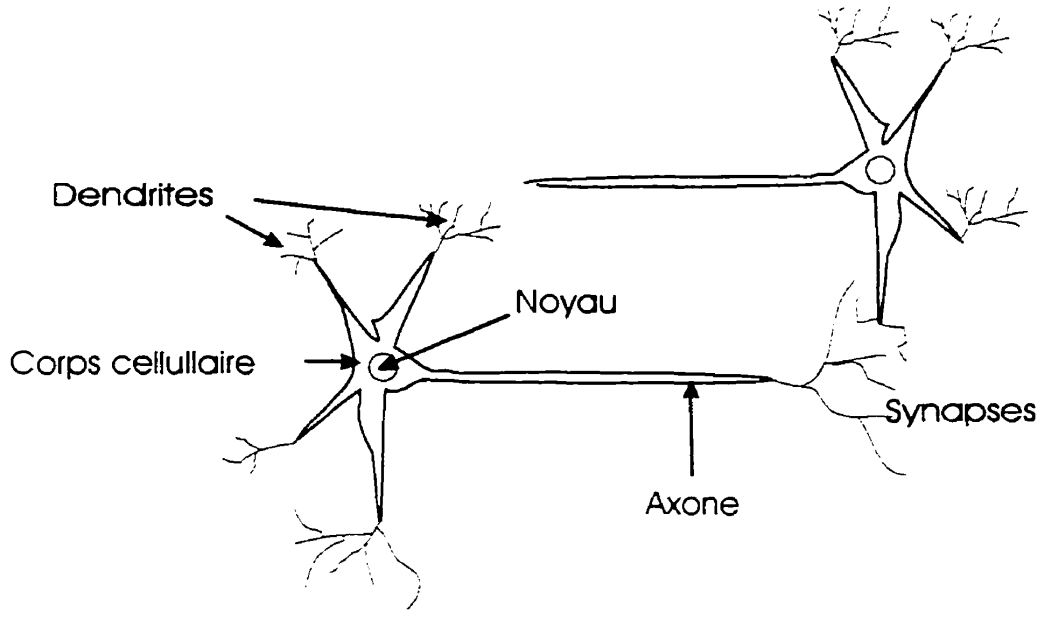


Figure 1.1 : La cellule nerveuse

La structure électrochimique des cellules des tissus humain peut donc répondre à l'application d'un courant ionique. Cette propriété permet d'influencer le comportement d'un groupe de cellules ou d'un muscle du corps à partir d'une source externe de courant. La stimulation électrique consiste donc à injecter un courant ionique sur une partie du corps humain, dans le but de lui « commander » un certain travail. Ce courant est injecté à l'aide d'électrodes ou d'un réseau d'électrodes selon l'application.

1.3 Classification des stimulateurs électriques

Des gammes complètes de stimulateurs électriques furent développées pour diverses applications durant les dernières décennies. Le stimulateur cardiaque est sans aucun doute le type le plus connu. Une classification possible des stimulateurs électriques est la suivante [24]:

1) les systèmes reliés par fils;

2) les systèmes externes;

3) les systèmes reliés par liens transcutanés ;

4) les systèmes totalement implantables.

Les stimulateurs de la classe #1 sont des systèmes où un implant est connecté à un contrôleur externe à l'aide de quelques fils qui traversent la peau (connexions percutanées). Ce genre de système est surtout utilisé pour effectuer des expériences en clinique, car il y a risque d'infection de la peau dû aux fils qui la traversent. Cependant, des applications non expérimentales entrent dans cette classe, comme par exemple le stimulateur cardiaque temporaire qui peut être introduit dans le cœur via une veine accessible et ensuite être enlevé lorsque le support cardiaque n'est plus requis [24]. Le professeur Wise et son équipe ont développé durant les dix dernières années une panoplie de stimulateurs et de sondes appartenants à cette classe [35,36,42,44,45].

Les stimulateurs de la classe #2 sont des systèmes où les stimulations sont appliquées directement sur la peau à l'aide d'électrodes de surface. Parmi les applications de cette classe, on retrouve le stimulateur électrique neuronal transcutanée (TENS) et la thérapie par choc électroconvulsif (ECT) [24]. Ces stimulateurs sont généralement employés dans un but thérapeutique et non comme moyen pour récupérer une fonction.

Les stimulateurs de la classe #3 utilisent un lien transcutané (aucune connexion physique) pour la transmission d'information et d'énergie entre un contrôleur externe et

un implant [37]. Le lien transcutané est généralement réalisé à l'aide d'un couplage inductif [24]. Plusieurs systèmes de stimulation de muscles paralysés [26], de la cochlée [43,54] et de la vessie [25] font partie de cette classe.

Les stimulateurs de la classe #4 ne comportent qu'un seul module qui est totalement implantable et indépendant. Ce genre de stimulateur doit donc contenir sa propre source d'énergie. Le stimulateur cardiaque appartient à la classe #4. Souvent, la pile utilisée dans cette application peut être rechargée à l'aide d'un lien inductif.

Généralement, c'est l'application qui détermine la classe de stimulateur à utiliser. Seuls les classes de stimulateur #1 et #3 peuvent être employées pour réaliser une prothèse visuelle. En effet, deux modules sont essentiels dans ce type d'application. Le module externe est nécessaire pour capter la lumière (les images) et pour la traiter. Le module interne (l'implant) est indispensable, car l'endroit à stimuler dans tous les cas (rétine, nerf optique ou cortex) est situé à l'intérieur du corps humain.

Deux sous-catégories existent dans les classes #1 et #3 : les systèmes possédant un seul canal de stimulation et les systèmes en possédant plusieurs. L'implant cortical comporte nécessairement plusieurs canaux car différents sites dans le cortex doivent être excités simultanément pour générer des patrons reconnaissables par le non-voyant. L'implant cortical n'est pas la seule application des stimulateurs multicanaux. L'implant rétinien [3], certains implants cochléaires [43], musculaires [26] et de la vessie [25] utilisent eux aussi plusieurs canaux de stimulation. Un schéma bloc typique d'un stimulateur multicanaux est montré à la figure 1.2.

La partie implantable du stimulateur possède généralement un premier bloc regroupant un récepteur, un émetteur et un régulateur. Ce bloc permet la réception de l'information et du signal d'horloge, ainsi que la transmission de données. C'est également ce module qui convertit l'énergie reçue du contrôleur externe en un ou plusieurs niveaux de tension utiles.

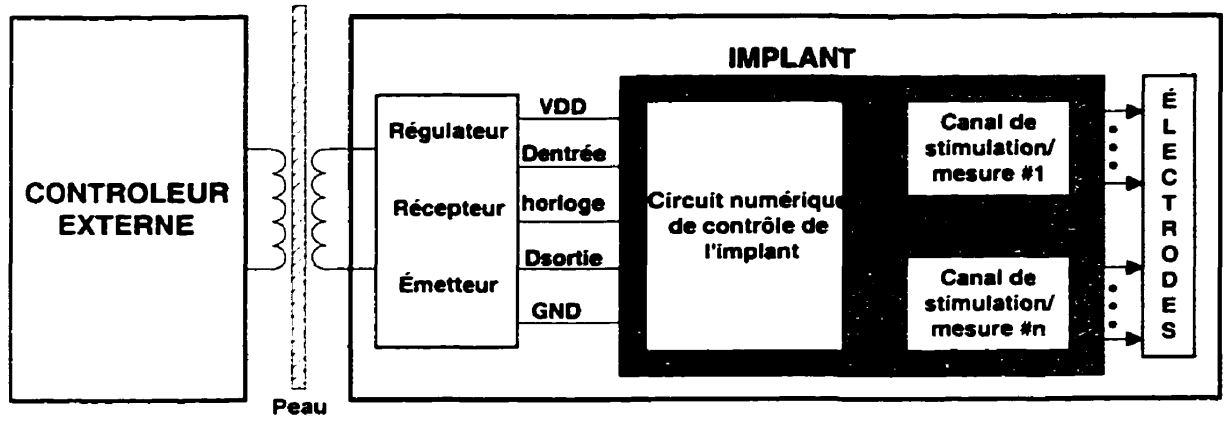


Figure 1.2 :Schéma bloc typique d'un stimulateur multicanaux

L'implant comporte aussi un circuit numérique qui selon les instructions du contrôleur externe, commande les opérations à effectuer à un ou plusieurs canaux de stimulation ou de mesure. Finalement une matrice d'électrodes est utilisée comme étage de sortie pour injecter les courants de stimulation aux cellules nerveuses.

Chaque canal de stimulation ou de mesure comprend un convertisseur numérique-analogique (CNA) et habituellement un système d'aiguillage des courants dans le cas où plusieurs sites (groupes de cellules) doivent être stimulés séquentiellement. Mise à part la stimulation de structure cellulaire, les implants peuvent également être utilisés pour enregistrer des signaux provenant de ces structures. Le système de mesure d'un canal est donc utile pour enregistrer ces signaux, mais aussi pour connaître l'état de l'interface tissu–électrode.

Les tâches effectuées par les implants peuvent être plus ou moins complexes. Certaines conceptions permettent la programmation de quelques paramètres de stimulation offrant la flexibilité requise pour les applications nouvelles ou pour celles qui risquent d'évoluer. Les prochaines sections donnent un aperçu des technologies utilisées ainsi que de quelques circuits existants permettant la réalisation de stimulateurs et autres implants de complexités diversifiées.

1.4 Les technologies

Les progrès réalisés dans le domaine des stimulateurs et autres implants en général sont étroitement reliés aux progrès effectués dans les procédés de fabrication des électrodes et réseaux d'électrodes ainsi que dans le domaine de la microélectronique. Généralement, le circuit numérique en entier et une bonne partie des circuits analogiques sont réalisés à l'intérieur d'un seul circuit intégré (CI) afin de réduire l'espace occupé par un implant.

De nos jours, la technologie CMOS est largement dominante en ce qui concerne le nombre des CI fabriqués. Un procédé de fabrication plus simple et moins coûteux est une première caractéristique qui avantage le transistor CMOS par rapport à son plus grand concurrent, le transistor bipolaire. Employés dans un circuit numérique, les transistors CMOS consomment très peu d'énergie lorsque la fréquence de commutation n'est pas trop élevée. Finalement, la grande impédance de grille de ces transistors en fait des composants idéaux pour l'étage d'entrée d'un circuit analogique.

Les transistors bipolaires ont cependant comme avantages de pouvoir fonctionner à des fréquences beaucoup plus élevées ainsi que de manipuler des courants plus importants que les transistors CMOS de même taille, ce qui est un atout certain lors de la conception de circuit analogique. Le procédé de fabrication BiCMOS intègre des transistors CMOS et bipolaires dans un seul et même CI. Cette technologie permet de prendre avantage de chacun des deux types de transistor. Toutefois, le procédé de fabrication d'un CI BiCMOS est considérablement plus coûteux car il nécessite plus de masques.

Une faible consommation de puissance et des petites dimensions sont les caractéristiques les plus souhaitables d'un CI incorporé dans un implant. De plus, la chaleur dissipée par l'implant et transmise au corps humain doit être réduite à son minimum. En effet, l'augmentation de température près des électrodes doit être inférieure à 1°C, sous peine d'endommager les tissus environnants [42]. De plus, dans le cas d'un stimulateur de classe #3, l'énergie fournie à l'implant par le contrôleur externe doit également être la

plus faible possible, afin de limiter les effets néfastes du champ magnétique créé par le lien inductif sur le cerveau.

Pour toutes les raisons mentionnées précédemment, la technologie CMOS est pour l'instant la technologie la plus utilisée afin de réaliser un CI incorporé dans un implant [25,26,35,36,45,54,57]. Toutefois, d'autres CI d'implants sont réalisés avec la technologie BiCMOS [42,56,59]. De meilleures performances, surtout avec les circuits analogiques, sont obtenues par ces implants.

L'équipe de Wise [35,36,42,44,45] a proposé quelques conceptions de stimulateurs et de sondes dont l'électronique est directement intégrée dans le même substrat utilisé pour fabriquer les électrodes de stimulation. Cette façon de faire à l'avantage de minimiser l'espace total occupé par l'implant. Par contre, des étapes additionnelles au procédé CMOS standard sont nécessaires pour fabriquer de tels implants.

1.5 Les techniques analogiques

Malgré le fait que les implants doivent dissiper le moins de puissance possible en plus d'être compacts, les nouvelles conceptions deviennent de plus en plus complexes et intègrent de plus en plus d'électronique au fil des ans. De nombreuses techniques analogiques développées au cours des dernières années permettent la réalisation de ces implants dont la complexité ne cesse d'augmenter.

Ces techniques sont axées principalement sur la réduction de la consommation de puissance et des dimensions du circuit plutôt que sur l'optimisation de leur vitesse et leur précision. Les circuits analogiques des implants se retrouvent la plupart du temps aux étages d'entrée et de sortie. Parmi ces circuits, on retrouve les circuits de transmission et de réception de données et d'énergie, les CNA, les circuits d'aiguillage des courants de stimulation et finalement les systèmes de mesure et de vérification de l'interface électrode–tissu. Les deux prochaines sections traitent de certains détails de conception des deux plus importants blocs analogiques, soit le système de transmission et de réception et les CNA.

1.5.1 Transmission et réception de données et d'énergie

Le type de système employé pour la transmission et la réception de données et d'énergie entre un implant et son contrôleur externe dépend largement de l'application. Le transfert peut être fait soit à l'aide de fils (connexion percutanées), d'un lien optique ou d'un lien inductif. Le transfert à l'aide de fils comporte un inconvénient majeur soit le risque d'infection de la peau par ces fils [58]. Le lien optique doit posséder un canal de transmission transparent, ce qui est difficilement réalisable dans le cas de l'implant cortical. Le lien inductif reste donc pour le moment le meilleur moyen pour effectuer ce travail.

Quatre fonctions doivent être accomplies par le bloc regroupant le transmetteur, le récepteur et le régulateur de l'implant cortical. Parmi ces fonctions, on retrouve :

- la régulation d'une ou plusieurs tensions pour alimenter le reste du circuit de l'implant;
- la détection des données (commandes de stimulation);
- la transmission de données (mesure ou état de l'implant);
- le recouvrement et la génération de l'horloge.

Quelques auteurs proposent divers moyens pour accomplir ces tâches à l'aide d'un implant comportant une seule [25,26,38,56-59,66] ou deux [65] bobines. La transmission d'énergie est rendue possible grâce au couplage inductif entre la bobine du contrôleur externe et celle de l'implant. Une configuration possible pour implanter un lien inductif est présentée à la figure 1.3.

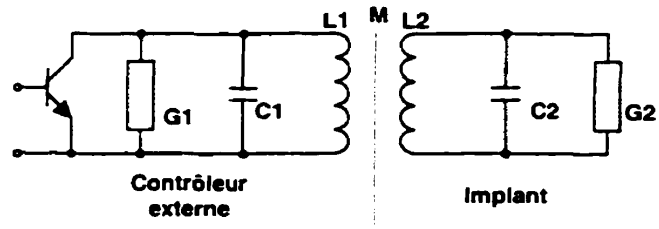


Figure 1.3 : Circuit simplifié d'un lien inductif

L'amplificateur de puissance du contrôleur externe et le circuit récepteur de l'implant forment tous les deux des filtres passe bande centrés sur une fréquence commune ω_0 où :

$$\omega_0 = \frac{1}{\sqrt{L_1 C_1}} = \frac{1}{\sqrt{L_2 C_2}} \tag{1.1}$$

Le choix de la fréquence ω_0 de la porteuse du signal dépend principalement de la capacité d'absorption des tissus et des dimensions de l'implant. Plus la fréquence est élevée, plus les grandeurs des bobines peuvent être réduites. Par contre, selon la puissance transmise par le contrôleur externe, la fréquence de transmission peut être limitée, car l'absorption des ondes électromagnétiques par les tissus humains augmente elle aussi avec la fréquence [69]. Plusieurs autres facteurs entrent en ligne de compte dans la conception d'un lien inductif comme par exemple la forme des bobines, la position de l'implant par rapport au contrôleur externe, la tolérance au déplacement et finalement les besoins en énergie, en tension régulée et en bande passante [60,65]. Le rendement du lien inductif, soit le rapport de la puissance consommée par l'implant sur la puissance totale qui lui est transmise est surtout fonction de la fréquence de la porteuse, du coefficient de couplage k entre les deux bobines et du type de régulateur utilisé.

La régulation de tension se fait à partir d'un redresseur à simple [54] ou double alternance [26]. Des condensateurs sont utilisés dans ces circuits pour emmagasiner l'énergie et réduire les rebonds. Une diode zener peut être employée pour fournir une meilleure régulation [43].

La plupart des systèmes implantables de biotélémétrie utilisent la modulation de fréquence pour la transmission des signaux [61]. Dans les systèmes dont la distance entre les bobines émettrice et réceptrice est petite, les signaux FM ne sont pas affectés par le mouvement continu de ces bobines. Ceci fait du FM la solution de choix pour la transmission de signal analogique. Toutefois, en ce qui concerne la transmission de signaux numériques, plusieurs types de modulation peuvent être employés comme le ASK (Amplitude Shift Keying), le FSK (Frequency Shift Keing) et le PSK (Phase Shift Keying). Le ASK a l'avantage d'avoir le circuit de modulation/démodulation le plus simple. Le LSK (Load-Shift Keying) est une nouvelle méthode qui semble très prometteuse, car elle permet une transmission de données de l'implant vers le module externe très efficace en terme de consommation de puissance. La figure 1.4 montre un exemple de circuit pour implanter la modulation LSK.

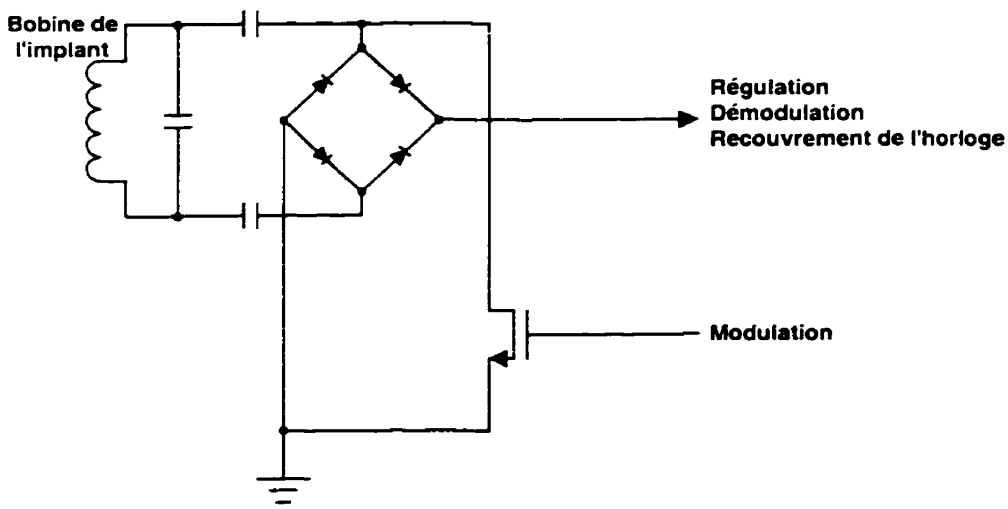


Figure 1.4 : Récepteur et modulateur LSK

Le LSK utilise la propriété du couplage inductif qui implique qu'un changement dans la charge de la bobine secondaire se reflète par un changement dans l'impédance de la bobine primaire [61]. Ce changement d'impédance peut facilement être détecté par le contrôleur externe. Dans le circuit de la figure 1.4, c'est en activant le transistor NMOS que l'impédance peut être changée. Ce type de modulation consomme peu de puissance

par rapport à la plupart des autres méthodes de transmission où la génération d'une porteuse est nécessaire. Au moins deux systèmes utilisent ce type de modulation pour leur transfert de données [26,58].

La fonction de recouvrement des données et la génération de l'horloge peuvent être accomplies de plusieurs façons. Une première méthode consiste à utiliser le code Manchester de façon à intégrer le signal d'horloge dans les données transmises du contrôleur externe vers l'implant. Avec ce type de codage, il y a toujours au moins une transition du signal par bit transmis ce qui permet de fournir un signal de synchronisation pour le générateur d'horloge. Comme illustré à la figure 1.5, un front montant indique qu'un « 0 » logique vient d'être transmis alors que les fronts descendant sont utilisés pour signaler les « 1 » logiques.

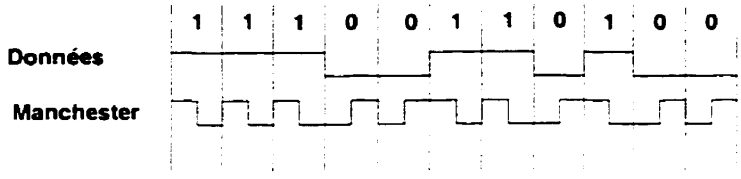


Figure 1.5 : Codage Manchester

Cette méthode à l'avantage d'être simple à implanter. Cependant, la fréquence de transmission des données est deux fois moins élevée que la fréquence de l'horloge [25,52]. Une autre approche consiste à utiliser la porteuse du signal transmis pour générer une horloge synchronisé avec le contrôleur externe [58].

La plupart des implants emploient des composants discrets dans une bonne partie de la circuitrie du système de réception, de transmission et de régulation. Ceci est nécessaire car plusieurs composants comme les diodes zener et les bobines par exemple sont plus difficiles à intégrer. Récemment, Akin et al. [58] ont développé un implant dont la circuitrie de réception et de transmission, à l'exception de la bobine, est entièrement intégrée dans un seul et unique CI.

1.5.2 Convertisseur numérique analogique (CNA)

Le convertisseur numérique analogique est un étage critique dans les stimulateurs implantables. En effet, ce module doit produire un courant avec une bonne précision (jusqu'à 8 bits quelquefois), être compact et consommer peu d'énergie. Les deux dernières caractéristiques sont très importantes, car il y a le même nombre de convertisseurs que de canaux dans un stimulateur multicanaux. Par exemple, l'implant visuel proposé par Boyer [52] utilise 25 CNA. Quelques auteurs ont proposé des architectures intéressantes de CNA utilisés dans des implants [36,39,42,45,52,62].

Un convertisseur simple employé par Kim [45] est présenté à la figure 1.6. Elle utilise des miroirs de courants dont le rapport W/L augmente d'un facteur deux pour chaque bit du mot d'entrée. Chaque bit est associé à un miroir de type N pour tirer un courant du nerf vers la source et un autre de type P pour envoyer un courant de la source vers le nerf. L'utilisation des deux types de miroir donne la possibilité d'effectuer des stimulations biphasiques.

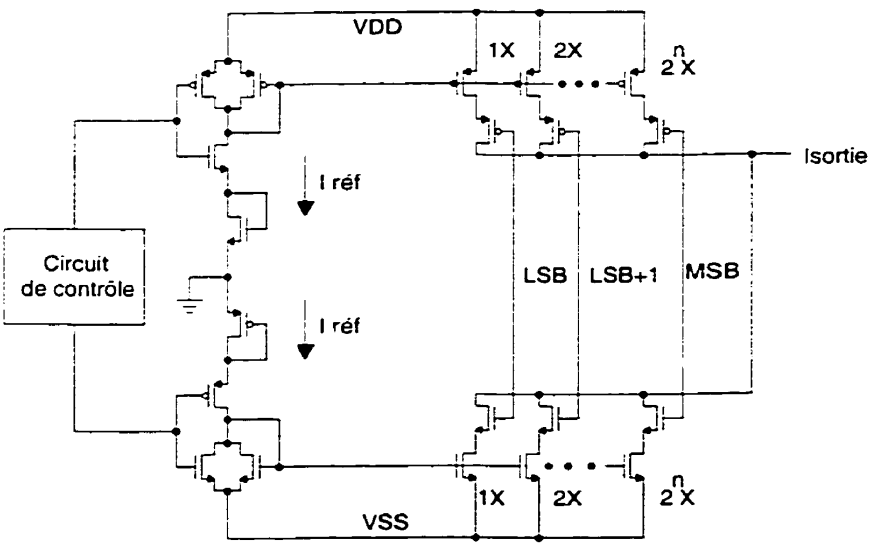


Figure 1.6 : CNA à miroirs de courant simple

L'impédance de sortie de ce convertisseur peut être augmentée en utilisant des miroirs de courant de type cascode, tels qu'employés dans le CNA utilisé par Jones et Normann

[39]. Cependant, cette dernière source occupe environ le double de l'espace par rapport au CNA de Kim.

Une seconde topologie proposée par Kim [42] et réalisée à l'aide d'un procédé BiCMOS, utilise des interrupteurs MOS associés à chacun des bits pour contrôler le courant de base d'un transistor bipolaire et donc son courant au collecteur (figure 1.7).

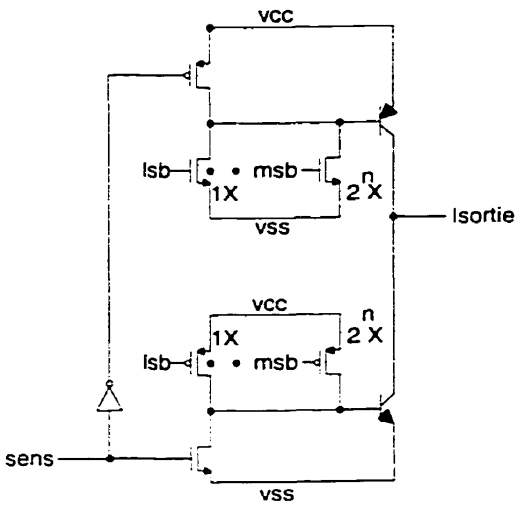


Figure 1.7 : CNA en BiCMOS

Les dimensions de ce circuit sont réduites par rapport aux deux autres convertisseurs, car ici les transistors utilisés comme interrupteurs sont également employés pour contrôler le courant dans la base du transistor bipolaire.

La figure 1.8 montre une dernière topologie dont la principale caractéristique est d'être très compacte. Le courant de référence généré par le CNA est réalisé à l'aide de transistors identiques placés en série ou en parallèle [62,63]. La direction du courant dans le nerf est contrôlée à l'aide d'une configuration en H, telle que montrée à la figure 1.9.

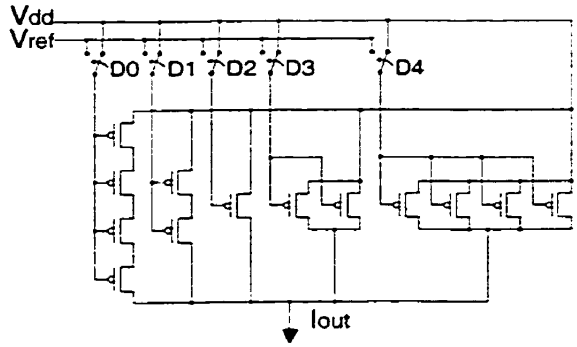


Figure 1.8 : CNA proposé par St-Amand [62]

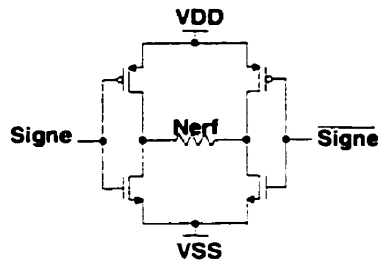


Figure 1.9 :Contrôle du sens du courant

1.6 Électrodes et réseaux d'électrodes

Les électrodes et réseaux d'électrodes servent d'intermédiaire entre l'étage de sortie de l'électronique et les cellules nerveuses lors de l'injection des courants de stimulation. Depuis environ trois décennies, des techniques basées sur la lithographie ont fait leur apparition, permettant de réaliser des réseaux d'électrodes de faibles dimensions [29-33,64]. Le silicium s'est avéré comme étant un substrat très avantageux pour servir de base à une telle structure, car il offre une bonne précision, il est biocompatible et il permet l'intégration de circuits électroniques [32]. Ces progrès ont amenés la réalisation de réseaux d'électrodes de grande densité (figure 1.10) qui pourraient éventuellement être utilisés dans des systèmes de stimulation corticale [32, 33].

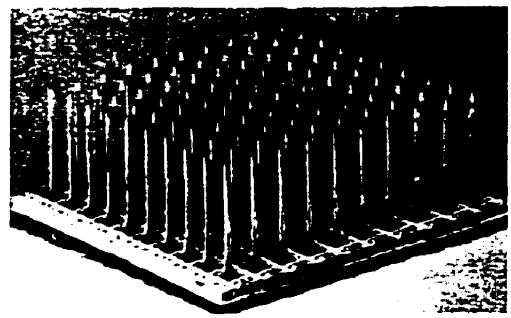


Figure 1.10 : Réseau d'électrodes (tiré de [33])

Le choix du matériau qui recouvre la surface de stimulation des électrodes doit tenir compte de la biocompatibilité et de certains effets nuisibles comme la polarisation des électrodes. Le matériau doit être le plus inerte possible afin de prévenir son oxydation. Par contre, ce même type de matériau a généralement la propriété d'être facilement polarisable (accumulation de charge à l'interface électrode-tissu). Ce phénomène peut cependant être éliminé en effectuant des stimulations biphasiques.

1.6.1 Modélisation de l'interface électrode-tissu

Les recherches effectuées au niveau du comportement électrochimique de l'interface entre les électrodes de stimulation et les tissus humains ont démontré la grande complexité à modéliser cette interface [46,47,67,68]. Une bonne connaissance de ce phénomène est néanmoins nécessaire pour pouvoir garantir le fonctionnement efficace et sécuritaire de l'implant. Par exemple, la caractérisation de cette interface permet de connaître son impédance, un élément important dans la conception des sources de courant de l'implant. Un modèle très utilisé de cette interface est présenté à la figure 1.11. La source de tension $E_{1/2}$ modélise le potentiel demi-cellule dont la valeur dépend principalement du type de métal de l'électrode, de la concentration de ses ions dans l'électrolyte et de la température [70]. Ce potentiel est dû à une séparation des charges à l'interface, produisant une double couche électrique où un type de charge est dominant sur la couche de l'électrode et le type de charge opposé est dominant sur l'autre couche. Ce phénomène est appelé la double couche de Helmholtz. Lorsque la densité de courant n'est pas trop élevée, les valeurs de la résistance Rw et du condensateur Cw (impédance

de Warburg) varient proportionnellement avec l'inverse de la racine carré de la fréquence. La valeur de Rw diminue et celle de Cw augmente de façon non-linéaire avec une augmentation dans la densité de courant.

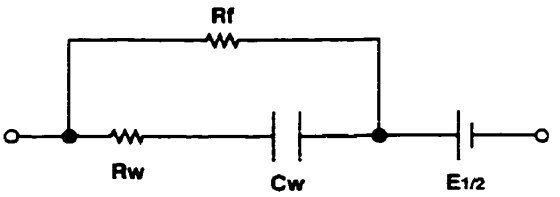


Figure 1.11 : Modèle de l'interface électrode-électrolyte

La résistance de Faraday Rf doit nécessairement être incluse dans le modèle, car il est possible de faire passer un courant continu à travers l'interface. Cette résistance est composée en réalité de trois éléments, soit la résistance de transfert, la résistance de diffusion et finalement de la résistance de réaction [67]. Les détails de cette interface dépassent le cadre du présent projet maîtrise. Des explications plus approfondies sur ce sujet peuvent par contre être trouvées dans divers articles [46,47,67,68].

1.7 Considérations d'ordre biomédicales

Seuls quelques aspects techniques furent abordés jusqu'à présent dans ce chapitre. Toutefois, d'autres considérations doivent être tenues en compte lors de la conception d'un implant. Selon l'application, le stimulateur peut demeurer à l'intérieur du corps humain pendant plusieurs années. Celui-ci possède cependant des mécanismes qui le protègent contre l'intrusion de corps étrangers. L'implant doit donc être biocompatible. Par ailleurs, l'objectif premier d'un stimulateur est d'injecter un courant dans le tissu nerveux afin de provoquer une certaine réaction. L'injection de charges peut endommager les tissus nerveux. Des précautions doivent donc être prises pour éviter cela.

1.7.1 Biocompatibilité des implants

Les implants sont habituellement recouverts d'un polymère ou d'un verre quelconque qui ont comme fonction de bien protéger l'implant sans causer d'effets néfastes sur le corps humain. La biocompatibilité des implants est surtout critique au niveau des électrodes de stimulation. Celles-ci doivent être conçues de façon à minimiser les blessures causées aux tissus nerveux car ces blessures peuvent provoquer des inflammations [50]. L'injection d'un courant dans le corps à l'aide d'électrodes produit des réactions électrochimiques. Ces réactions peuvent changer de façon importante l'environnement des électrodes en changeant les concentrations d'ions et le PH et ainsi provoquer leur corrosion [50]. Le platine ou l'iridium sont couramment utilisés comme surface de stimulation [36,45,57,64] car ces matériaux sont presque inertes chimiquement. Outre les matériaux, la biocompatibilité des électrodes est fonction de leur géométrie, leur stérilisation, des paramètres de stimulation et finalement de la technique utilisée pour les insérer dans le cerveau [49,50,71].

1.7.2 Excitabilité et lésion neuronale

La stimulation prolongée d'une structure dans le système nerveux central ou sur une fibre nerveuse peut causer des lésions neuronales irréversibles. Habituellement, des stimulations biphasiques sont utilisées pour exciter un groupe de cellules nerveuses, c'est-à-dire qu'une opération de stimulation comporte deux phases, l'une positive et l'autre négative. Le courant de stimulation dans les cellules passe donc dans les deux directions.

Selon McCreery et al. [48], deux facteurs, soit la charge par phase et la densité de charge contribuent conjointement aux risques d'induire des lésions aux neurones. La charge par phase est définie comme étant l'intégrale du courant d'une phase (demi-cycle) de stimulation. La densité de charge est définie comme étant l'intégrale de la densité du courant de l'une ou l'autre des phases du stimulus. La densité de charge correspond également à la charge par phase divisée par la surface active des électrodes. Toujours selon McCreery et al., la densité de charge déterminerait la proportion des neurones près

de l'électrode qui sont excités et la charge par phase déterminerait le volume total de tissu dans lequel les neurones sont excités. Les risques de lésions peuvent donc être réduits en augmentant la surface active des électrodes et en diminuant la quantité de charge (donc le courant) injectée par phase. La surface de l'électrode doit par contre être limitée, car plus celle-ci est grande, plus les lésions « mécaniques » causées lors de son insertion sont importantes.

Une autre conséquence d'une stimulation prolongée d'un groupe de neurones consiste en une réduction de leur excitabilité [51]. Ce phénomène nommé SIDNE (Stimulation-Induced Depression of Neuronal Excitability) est beaucoup moins néfaste qu'une lésion, car il est réversible et serait dû à un mécanisme de protection contre les lésions neuronales [51]. La sévérité d'un SIDNE s'accentue avec l'augmentation de la durée et de la fréquence de la stimulation. Augmenter l'amplitude pour contrer un SIDNE ne fait qu'aggraver ce phénomène. Une façon possible pour éviter ce phénomène est de distribuer la stimulation sur plusieurs électrodes de façon séquentielle, pour ainsi réduire la durée d'activation par cycle de chaque électrode.

1.8 Conclusion

Ce chapitre a présenté une introduction aux stimulateurs sensoriels. Nous avons décrit et expliqué les différentes classes de stimulateurs, ainsi que les principaux aspects à considérer par rapport aux technologies, aux techniques analogiques et aux électrodes de stimulation. Finalement, nous avons fait état de certaines considérations d'ordre biomédical dont on doit tenir compte.

Un élément qui se démarque dans l'étude des implants c'est qu'ils doivent tous être optimisés en fonction de leur consommation d'énergie et de leur dimension. Ces caractéristiques sont également essentielles pour n'importe quel type d'implant visuel.

Le prochain chapitre traite des prothèses visuelles. C'est-à-dire des stimulateurs visuels en général (rétine, nerf optique et cortex) et de l'implant cortical en particulier.

CHAPITRE 2

LES PROTHÈSES VISUELLES

2.1 Introduction

Le chapitre précédent a exposé les principaux points touchant les stimulateurs physiologiques en général. Le présent chapitre approfondit le sujet de la récupération partielle de la vision.

La première section décrit le système visuel humain et les principales maladies de la vision. Ensuite, on présente et compare brièvement les trois grandes avenues de recherche pour rétablir la vision soit : la stimulation de la rétine, la stimulation du nerf optique et finalement la stimulation du cortex visuel.

Le type de prothèse visuelle en cours de réalisation par notre équipe de recherche (PolySTIM) est un stimulateur cortical. Le prototype développé dans le cadre de ce projet de maîtrise doit donc avoir la même fonctionnalité qu'un implant cortical. C'est pourquoi la section suivante traite des connaissances actuelles sur la stimulation du cortex visuel et de ses effets.

Finalement, nous présentons les principaux travaux de recherche sur le développement de l'implant cortical et sur l'efficacité de la vision que l'on projette obtenir à l'aide de celui-ci.

2.2 Le système visuel humain et les maladies de la vision

La figure 2.1 montre la structure de l'œil. La principale tâche de l'œil est de focaliser la lumière sur sa rétine, premier étage cellulaire du système visuel humain. La rétine est constituée entre autres de deux types de cellules photoréceptrices (qui détectent la lumière) soient les cônes et les bâtonnets. Les cônes, surtout concentrés près de la fovéa,

sont sensibles à de hauts niveaux d'intensité lumineuse et aussi à la couleur. Les 5.5 millions de ce type de cellule occupent une surface circulaire d'environ 0.5 mm de diamètre sur la rétine de chaque œil. Les bâtonnets, répartis sur toute la surface de la rétine, sont sensibles à de faibles niveaux d'intensité lumineuse et sont insensibles à la couleur. Il y a environ 120 millions bâtonnets dans la rétine de chaque œil. La rétine possède principalement deux autres étages cellulaires, soit les cellules bipolaires et horizontales et les cellules ganglionnaires. Ces étages de cellules effectuent quelques traitements avant d'envoyer l'information aux autres étages de cellules dans le système nerveux.

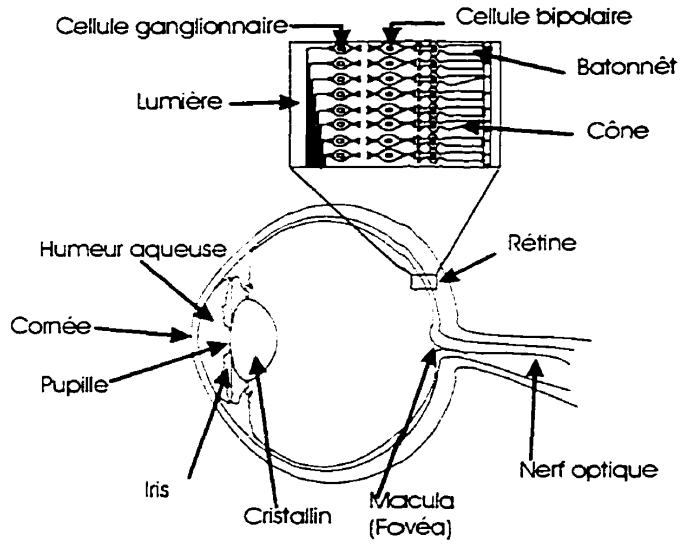


Figure 2.1 : Vue en coupe d'un oeil

La figure 2.2 montre la structure du système visuel humain aussi appelé « voies visuelles ». Les deux nerfs optiques amènent l'information des cellules ganglionnaires de chaque rétine à un point de croisement appelé Chiasma optique. Ensuite, l'information passe par les tractus optiques vers une couche de cellules située de chaque côté du cerveau appelé corps genouillé latéral qui passera à son tour l'information au cortex visuel. Les nerfs optiques, le chiasma optique et les tractus optiques ne font pas ou font peu de traitement. Leur principale utilité est de joindre l'information des deux côtés du champ visuel provenant de chaque œil.

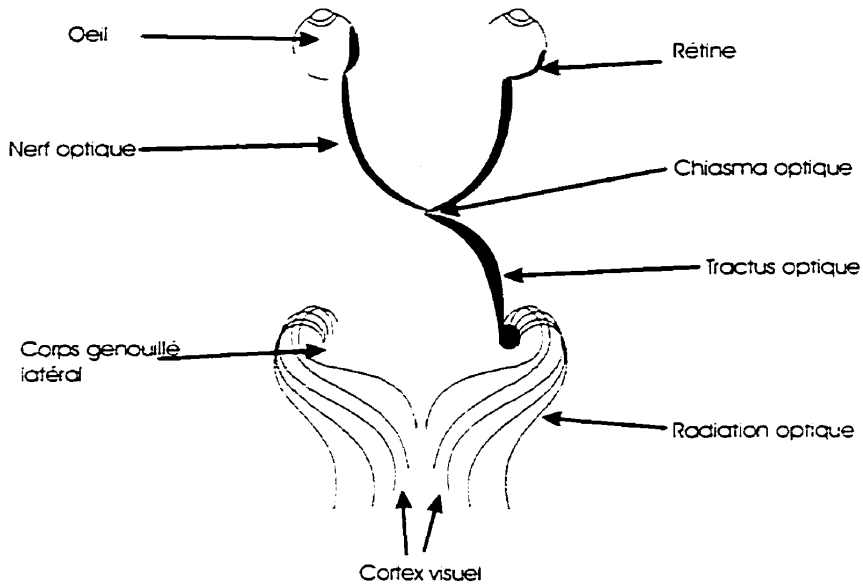


Figure 2.2 : Le système visuel humain

Les cellules du corps genouillé latéral et du cortex effectuent d'autres traitements pour produire à la fin une sensation de vision chez l'humain. L'information entre les diverses couches de cellules voyagent dans les deux directions, soit de la rétine au cortex visuel et vice-versa.

Les maladies de la vision les plus répandues aux États-Unis sont, dans l'ordre, la dégénérescence maculaire, le glaucome, la rétinopathie diabétique et finalement la cataracte [1,5]. La rétinite pigmentaire est une autre maladie bien répandue au Canada [1] et aux États-Unis. La dégénérescence maculaire est un trouble où la macula (centre de la rétine) cesse de répondre à la lumière. Le glaucome est une complication liée à l'humeur aqueuse dans l'œil qui ne s'écoule pas normalement et où une forte pression se développe causant des dommages à la délicate structure de l'œil. La rétinopathie diabétique est due au fait que des vaisseaux sanguins formés anormalement éclatent, endommageant la rétine. La cataracte est un problème au niveau du cristallin de l'œil qui s'épaissit, se durcit et devient opaque bloquant le passage de la lumière. Finalement, la rétinite pigmentaire est causée par la dégénérescence de la rétine. La déficience visuelle est quelque fois causée par des lésions au niveau du nerf optique, ce qui entraîne alors

des pertes unilatérales de champ visuel ou par des lésions au niveau du chiasma optique ou plus loin, ce qui entraîne généralement des pertes bilatérales.

2.3 Les avenues de recherche pour récupérer partiellement la vision

Jusqu'à maintenant, il y a principalement trois types de prothèses visuelles (ou stimulateur visuel) qui font l'objet de travaux de recherche pour rétablir artificiellement la vision, soit : l'implant rétinien ou oculaire, le stimulateur du nerf optique et finalement l'implant cortical. Un point commun à ces prothèses, elles possèdent toutes les trois un module externe et un module interne. Dans chaque cas, le module externe est responsable de capter la lumière (les images), d'effectuer un certain traitement sur ces images (le traitement varie selon le site de stimulation) et finalement d'envoyer l'information et l'énergie requise au module interne (implant). Le module interne reçoit l'information du module externe et applique des courants de stimulation à une certaine couche de cellules selon le type de stimulateur. Ces courants de stimulation ont la propriété de rétablir une certaine vision chez le non-voyant. Au moins un autre site d'implantation, le corps genouillé latéral, fut considéré pour rétablir la vision, mais pour le moment, cette option est rejetée, car il est situé très profondément dans le cerveau et il est difficile d'y accéder par chirurgie [3,6,7].

2.3.1 L'implant rétinien

Bien que récent, l'implant rétinien est très populaire et il existe plusieurs groupes de recherche qui étudient cette avenue, afin de récupérer partiellement la vision. Les implants rétiniens du MIT (Massachusetts Institute of Technology) [3,7] et des partenariats allemands EPI-RET [8] et SUB-RET [9] sont parmi les projets [41,53] les plus avancés dans le développement de ce type de prothèse visuelle. La figure 2.3 montre le schéma d'un implant rétinien typique. Le module externe comprend une caméra CCD, un circuit intégré (CI) de traitement de signal et d'une diode électroluminescente laser, tous montés sur une paire de lunettes que portera le non-voyant. Le CCD capte les images, le CI effectue des traitements sur l'image et la diode laser envoie l'information (les commandes de stimulation) et l'énergie à l'implant.

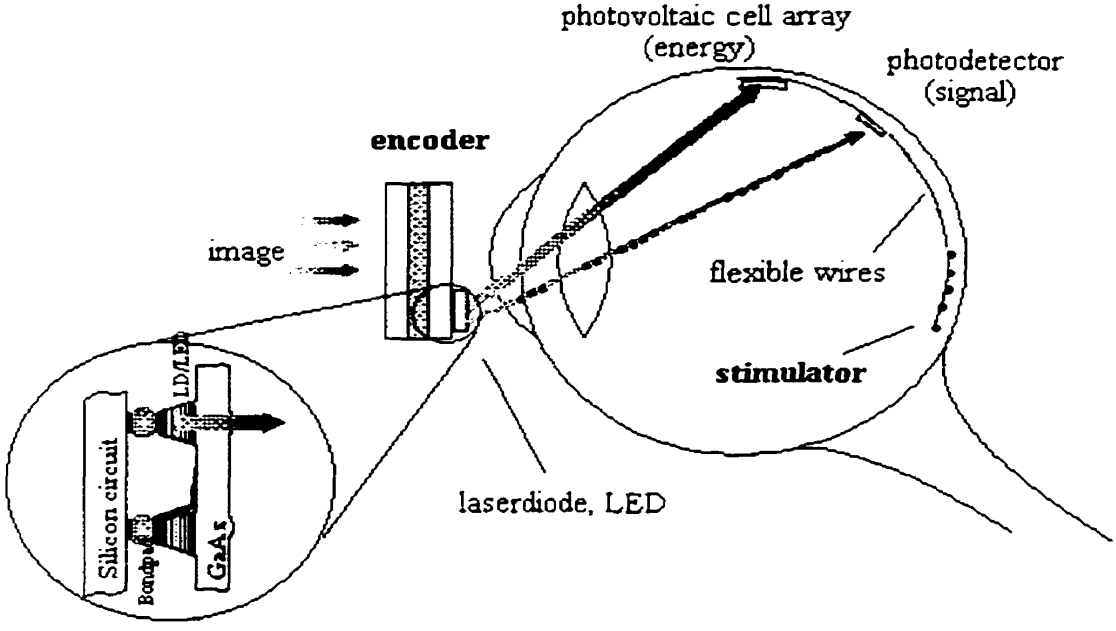


Figure 2.3 : Implant rétinien (tiré de [8])

L'implant, monté sur un circuit imprimé flexible est fixé sur la couche interne de la rétine et comporte une matrice de photodiodes récepteurs et un stimulateur. Les photodiodes captent l'énergie et l'information nécessaires au stimulateur qui envoie alors des courants de stimulation sur les cellules ganglionnaires de la rétine.

Quelques différences de conception et lieux d'implantation existent parmi les groupes de recherche. Ainsi, la plupart de ces groupes utilisent un lien RF au lieu d'une diode laser pour véhiculer l'énergie et l'information du module externe au module interne. De plus, le choix de l'endroit précis d'implantation sur la rétine, soit à l'avant (implant épi-rétinal) [3,7,8] ou à l'arrière (implant sous-rétinal) [9] de la rétine, diffère selon les groupes. Le CI de traitement est un module qui doit émuler la fonctionnalité de la rétine, donc sa fonctionnalité est largement dépendante de l'endroit d'implantation du stimulateur.

2.3.2 Le stimulateur du nerf optique

À notre connaissance, un seul groupe [10] développe ce genre de système. Le projet MIVIP (Microsystems based Visual Prosthesis) consiste à réaliser un prototype comprenant un œil artificiel, un processeur externe, un lien RF pour la transmission de données et d'énergie transcutanée, un neurostimulateur et une électrode en spirale pour stimuler le nerf optique. Peu d'information et de références sont actuellement disponibles sur cette approche (figure 2.4).

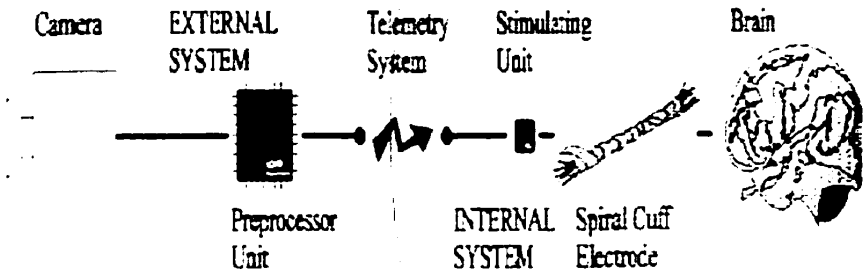


Figure 2.4 : Stimulateur du nerf optique (tiré de [10])

2.3.3 L'implant cortical

La perspective de développer une prothèse visuelle en utilisant la stimulation corticale existe depuis une trentaine d'années grâce aux recherches de Brindley et Lewin [11] et celles de Dobelle et Mladejovsky [6,12]. Ces chercheurs examinaient le moyen de réaliser une prothèse visuelle basée sur le fait que de petits points lumineux appelés phosphènes apparaissent dans le champ visuel d'un non-voyant lorsqu'on lui applique des courants de stimulation à la surface de son cortex visuel.

L'intensité des courants de stimulation nécessaires pour produire des phosphènes était alors de l'ordre du milliampère. À l'époque, l'énergie requise d'un système de stimulation comme celui-là était trop importante pour être physiquement réalisable et l'avancement de ce type d'implant fut donc arrêtée pendant quelques années.

Les découvertes sur les effets de la stimulation à l'intérieur du cortex plutôt qu'à sa surface ont cependant donné un regain de vie à cette approche. De plus, les récents progrès en microfabrication ont permis le développement de matrices d'électrodes [33] essentielles à ce genre de prothèse. La figure 2.5 montre le principe de base d'un stimulateur cortical.

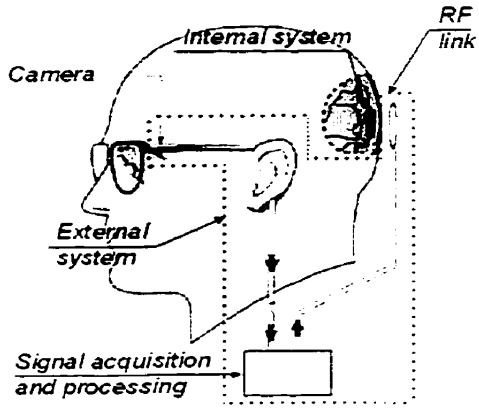


Figure 2.5 : Implant cortical (adapté de [14])

Premièrement, une caméra capte les images et envoie l'information à une unité de traitement [85]. Celle-ci améliore les contrastes et réduit la résolution des images. L'information est alors envoyée sous forme de mots de commande à l'implant. Celui-ci effectue les opérations de stimulation selon l'information reçue. Ces opérations consistent à injecter des courants de stimulation à plusieurs sites du cortex visuel. L'intensité et la stabilité des phosphènes sont contrôlées en modifiant les paramètres (amplitude, durée de phase et d'interphase, etc) des courants de stimulation. L'implant utilise une matrice d'électrodes comme celui de la figure 1.10 du premier chapitre pour injecter les courants dans le cortex. Plus de détails seront donnés sur la stimulation du cortex et sur l'implant cortical dans les prochaines sections.

2.3.4 Comparaison entre les différentes avenues de recherche

L'implant rétinien stimule les cellules situées dans la première couche du système visuel humain. Cet implant a l'avantage qu'à l'exception de certaines couches de cellules de la

rétine, toutes les autres cellules du système visuel peuvent être utilisées permettant une meilleure récupération de la vision. À ce jour, les connaissances sur la fonctionnalité des cellules de la rétine sont plus importantes que celles des autres couches du système visuel. La technique de traitement d'image de l'implant rétinien peut donc être plus efficace que celles des autres approches. De plus, l'opération chirurgicale pour installer l'implant peut être plus simple et respecte les techniques connues [3,7], ce qui n'est pas le cas avec les autres types d'implants. Cependant, cette approche ne peut être utilisée si la maladie affecte la couche des cellules ganglionnaires (comme par exemple le glaucome) ou tout autre cellule située plus loin dans le système visuel humain. Sur ce point, l'implant cortical est le meilleur choix des trois car il est situé dans les derniers étages du système visuel.

Le nerf optique mesure environ 50 mm et a un diamètre d'environ 3mm [3]. Il contient environ 1 million de fibres qui sont regroupées dans une membrane difficile à enlever sans endommager les tissus nerveux. Le grand désavantage de cette approche, c'est que les fibres du nerf optique ne sont pas organisées de façon stricte comme peuvent l'être les cellules de la rétine. La stimulation isolée d'une sélection de fibres est donc difficile à obtenir. Par contre, cette méthode a l'avantage de pouvoir être utilisée pour n'importe quel genre de maladie de la vision affectant l'œil et la rétine (donc la majorité des causes de cécité). Selon Wyatt [3], cette approche est la moins encourageante en ce qui concerne la possibilité de redonner une certaine forme de vision.

L'implant cortical nécessite une intervention chirurgicale majeure car l'implant doit être placé assez profondément dans le cortex visuel du patient. Cependant, des expériences encourageantes ont démontré qu'une personne aveugle depuis 20 ans a su reconnaître de simples patrons à l'aide de cette méthode. En admettant que les trois types d'implants soient disponibles dans un futur proche, c'est le genre de maladie qui indiquera le type d'implant à utiliser. L'implant cortical pourra cependant être utilisé comme moyen de réhabilitation pour presque tous les types de maladies.

2.4 Les connaissances actuelles sur la stimulation du cortex visuel

2.4.1 Bref historique

Le tableau 2.1 présente une synthèse des travaux de recherche qui ont contribué à l'évolution des connaissances sur la stimulation du cortex visuel.

Tableau 2.1 : Historique de la stimulation du cortex visuel

Date	Chercheur(s)	Contribution	Référence
1929	Foester	Découverte du phosphène.	[15]
1968	Brindley, Lewin	Expériences sur la stimulation de la surface du cortex visuel des humains.	[11]
1974	Dobelle, Mladejovsky	Expériences sur la stimulation de la surface du cortex visuel des humains.	[6,12,16]
1980	Barlett, Doty	Stimulation intracorticale sur des primates.	[17]
1983	Ronner, Lee	Stimulation intracorticale sur des chats.	[18]
1990-1996	Hambretch et al.	Stimulation intracorticale sur des humains.	[19,20,21, 22]
1999	Rousche, Normann	Stimulation intracorticale sur des chats à l'aide d'une matrice d'électrodes rigides.	[79]

Foester est le premier chercheur à avoir découvert le phénomène des phosphènes, c'est-à-dire l'apparition de points lumineux dans le champ visuel d'un non-voyant suite à l'application de courant de stimulation sur le cortex visuel. Ce phénomène fut par la suite observé de façon concluante par Brindley [11] et Dobelle [6,12,16]. Ces derniers utilisaient des électrodes placées à la surface du cortex visuel pour provoquer des phosphènes. Bien que les résultats obtenus alors semblaient encourageants, la stimulation de surface comporte néanmoins quelques limites. Notamment, la valeur du courant de stimulation pour produire un phosphène est de l'ordre du milliampère et les phosphènes produits par les stimulations ne sont pas parfaitement stables.

Les premières expériences de stimulation intracorticale (stimulation à l'intérieur du cortex et non pas sur sa surface) furent réalisées par Barlett et Doty sur des macaques [17]. Les résultats de ces expériences ont démontré que le seuil du courant nécessaire pour faire apparaître un phosphène peut être aussi bas que 2μA et que ce seuil dépend largement de la forme d'onde du stimulus. Hambretch et son équipe furent les premiers à apporter des contributions dans le domaine de la stimulation intracorticale du cortex visuel chez les humains. Les résultats qu'ils ont obtenu donnent une bonne idée de la fonctionnalité que doit posséder l'implant cortical et sont présentés à la prochaine section.

2.4.2 Les travaux de l'équipe de Hambretch

Depuis la fin des années 80, Hambretch et son équipe ont mené plusieurs expériences sur la stimulation intracorticale chez des humains [19,20,21,22]. Les résultats obtenus sont très encourageants en ce qui concerne la faisabilité d'une prothèse visuelle basée sur la stimulation intracorticale. La dernière publication de cette équipe a démontré qu'une personne totalement aveugle depuis plus de 20 ans a su reconnaître de simples patrons suite à l'application de courant de stimulation à l'intérieur du cortex [22]. Cette expérience a duré en tout quatre mois. La figure 2.6 montre les formes d'onde utilisées dans cette expérience. Des trains d'impulsions furent donc employés pour provoquer des sensations de phosphène. Les différents paramètres des trains ont une certaine influence sur l'apparition et l'aspect du phosphène. Parmi ces paramètres, on retrouve l'amplitude (AMP) du courant, les durées de phase (DP) et d'interphase (DIP) d'une impulsion et finalement les durées d'un train d'impulsion (DT) et le délai inter-train (DIT) entre les impulsions.

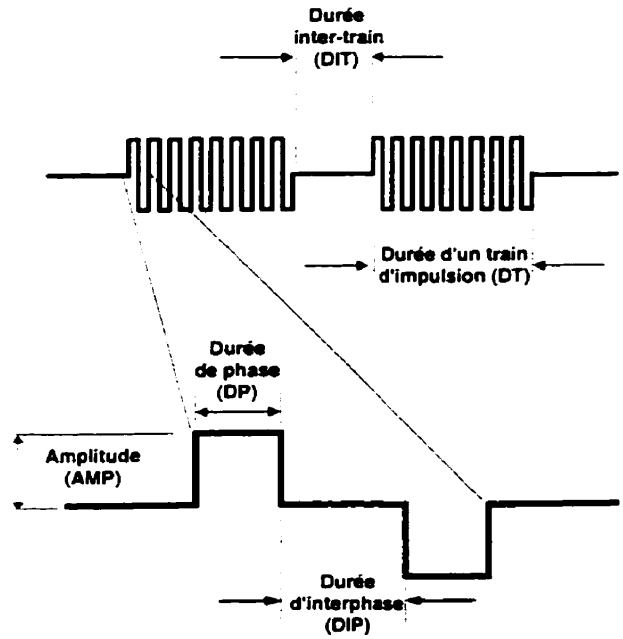


Figure 2.6 : Formes d'onde utilisées pour la stimulation intracorticale

Les résultats des expériences effectuées ont permis de dégager certaines informations par rapport à l'aspect et autres caractéristiques des phosphènes [21,22]. Ces informations donnent des indications notamment sur l'intensité, la durée, la taille et l'emplacement des phosphènes dans le champ visuel.

Ainsi l'intensité d'un phosphène augmente avec l'amplitude, la durée de phase, la fréquence et la durée du train d'impulsion. Les phosphènes sont plus stables et ont une meilleure « qualité visuelle » lorsque la durée de phase est supérieure à 400us. Le seuil de l'intensité du courant pour produire un phosphène varie entre 1.9uA et 80uA et dépend largement de la profondeur de pénétration des électrodes dans le cortex. Le seuil diminue en augmentant la durée de phase, la fréquence ou la durée du train d'impulsion.

L'intervalle de temps entre le début de la stimulation et l'apparition d'un phosphène varie généralement entre 395ms et 452 ms. La durée d'un phosphène est approximativement égale à la durée du train d'impulsion mais limitée à 930 ms. La

durée inter-train doit être inférieure à 25ms pour produire des phosphènes stables (qui n'oscillent pas).

Par rapport à la taille du phosphène, celle-ci diminue en augmentant l'amplitude ou la durée de phase. Par contre elle augmente avec la durée du train. Généralement, deux phosphènes distincts peuvent être générés simultanément lorsque l'espace entre deux sites de stimulation est d'au moins 500um. Il y a cependant interaction lorsque la distance entre les sites est de 250 um. Les interactions entre phosphènes peuvent être réduites et même éliminées en ajustant la durée inter-trains.

L'emplacement d'un phosphène dépend surtout de l'emplacement des électrodes de stimulation. Les phosphènes bougent avec le mouvement des yeux et lorsque plus d'un phosphène est généré, tous les phosphènes bougent en gardant la même distance relative.

Finalement, des phénomènes d'accoutumance furent observé lors de stimulation répétées. À ce moment, le seuil de l'intensité du courant requis pour l'apparition d'un phosphène augmente. L'augmentation de la durée de phase semble réduire les effets d'accoutumance.

2.5 Architecture et travaux de pointe sur l'implant cortical

Quelques auteurs ont proposé des conceptions de stimulateur implantable multicanaux possédant des caractéristiques souhaitables pour l'implant cortical. L'implant cochléaire de Mcdermott [43,54] et la prothèse urinaire de Sawan et al. [66] en sont des exemples. Ces travaux sont très intéressants car les stimulateurs proposés furent testés in vivo avec succès. Le nombre de canaux de stimulation est cependant faible dans chacun des cas. Wise et son équipe ont par ailleurs proposé quelques architectures de stimulateur et de sonde d'enregistrement comportant un grand nombre de canaux [35,36,42,44,45]. Mais ces implants appartiennent à la classe #1 (liens percutannés) un désavantage majeur par rapport aux stimulateurs de classe #3 (lien inductif transcutané). À ce jour, il n'existe aucun système complet de récupération de la vision basé sur la stimulation corticale.

Cependant, certains auteurs ont proposé des architectures et quelques projets sont en cours de réalisation.

2.5.1 La première version du stimulateur visuel miniaturisé de PolySTIM

Dans le cadre d'un mémoire de maîtrise, Alain Boyer de notre équipe de recherche (PolySTIM) a développé une architecture complète d'un implant cortical [52,86]. La figure 2.7 présente la structure globale du stimulateur. Cet implant possède 25 canaux dont chacun doit stimuler 25 sites de façon séquentielle. Un réseau de 650 électrodes est donc nécessaire à cette architecture. La structure de l'implant regroupe un module de récupération et préparation des données, un bloc d'aiguillage de l'information et 25 canaux de stimulation. Chaque canal possède un générateur de stimulus et un bloc de production et distribution du courant. Un seul mode d'opération existe et consiste à l'envoi de trames d'information (11 bits de données + 4 bits de Hamming) du contrôleur externe vers l'implant par le biais d'un lien RF inductif AM qui fournit en plus l'énergie nécessaire à l'implant.

Un décodeur Manchester est utilisé pour extraire les données et l'horloge du signal numérique d'entrée. Un accumulateur convertit ensuite les données sérielles reçues en mots de 15 bits. Chaque mot passe par le module de Hamming pour corriger une éventuelle erreur de transmission. Un mot de commande complet est constitué de 16 bits mais les données sont transmises par trame de 15 bits dont 4 sont des bits de parité. Seulement 11 bits de données utiles sont donc transmises par trame. Les mots de commande spécifient :

- l'amplitude de la stimulation (2 bits pour la plage choisi et 4 bits pour l'amplitude) ;
- la durée des phases (4 bits) ;
- la durée d'interphase (4 bits);
- la forme d'onde désirée (2 bits).

À l'aide du synchroniseur, le regénérateur de mot de commande détecte le moment où un mot de commande complet est reçu et envoie alors l'information au bon canal de stimulation. Le module d'aiguillage de l'information s'occupe de l'adressage des sites de stimulation. Le balayage des sites est donc fixe et ne change jamais. Ceci a permis de diminuer la longueur des mots de commande.

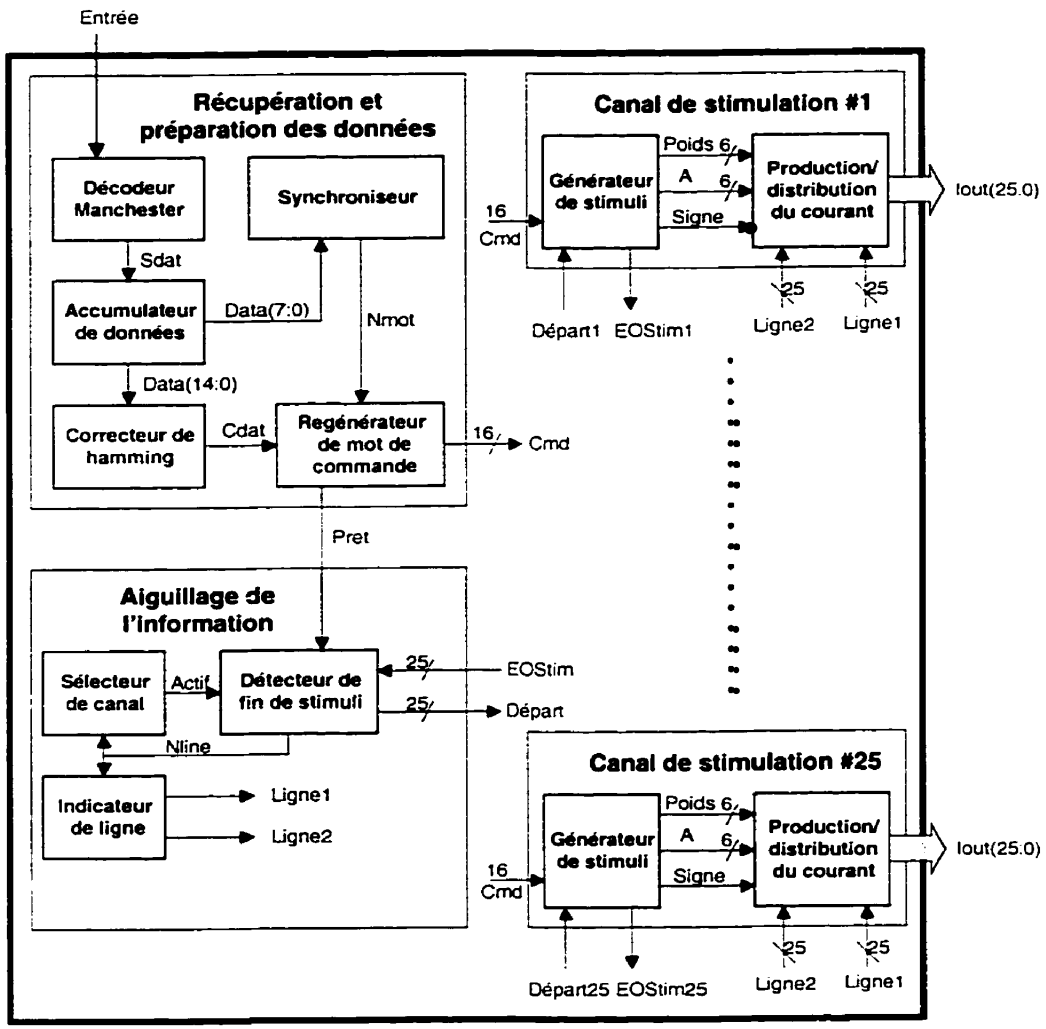


Figure 2.7 :Stimulateur visuel miniaturisé de Boyer [52]

Suite à la réception d'un mot de commande par un des 25 canaux, celui-ci débute alors les opérations de stimulation. Le générateur de stimuli du canal est une MSA qui permet

de générer une de trois formes d'onde de stimulation à l'aide du bloc de production et distribution du courant. Chaque stimulation est biphasique, c'est-à-dire que le courant injecté dans la structure cellulaire voyage dans les deux directions.

2.5.2 Le démultiplexeur physiologique de Jones et Normann

Jones et Normann [39] ont réalisé un CI prototype d'un implant multicanaux cortical. La structure globale de l'implant peut être divisée en deux sections : la section d'entrée présenté à la figure 2.8 et le sous-sytème de génération de stimuli (DAC control element) illustré à la figure 2.9.

La section d'entrée est un contrôleur qui reçoit les données sérielles provenant d'un module externe et les transforme en données parallèles qui sont ensuite envoyées sur différents bus pour être interceptées par le bon canal de stimulation (générateur de stimuli).

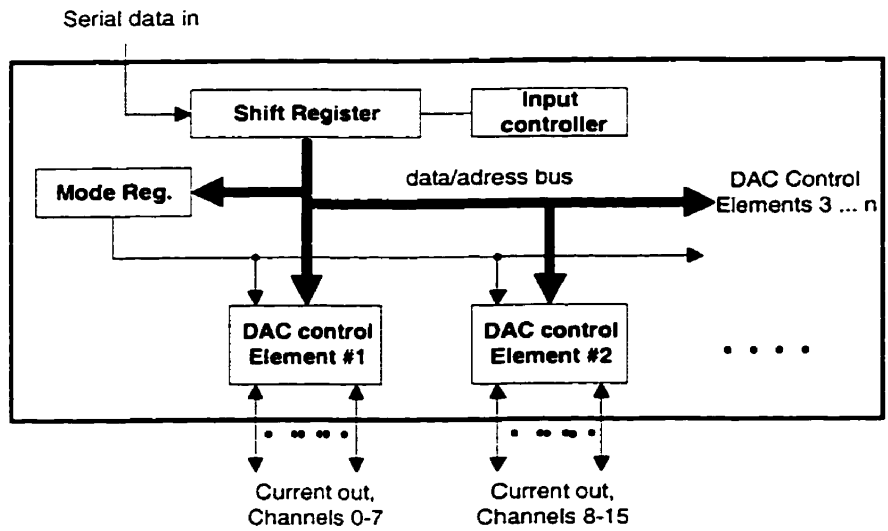


Figure 2.8 : Structure générale du CI de Jones et Normann [39]

Dans le but de réduire le taux moyen de transmission de données entre le contrôleur externe et l'implant, les auteurs proposent d'intégrer deux RAM dans chaque canal représentants une banque de pixels. Dans ces RAM, l'amplitude (1ère RAM) et la largeur

de l'impulsion (2ième RAM) de chaque site de stimulation y sont sauvegardées. Le taux de transmission des données entre la partie externe et l'implant est alors réduit car seulement les changements entre deux images successives ont besoin d'être envoyés aux RAMs.

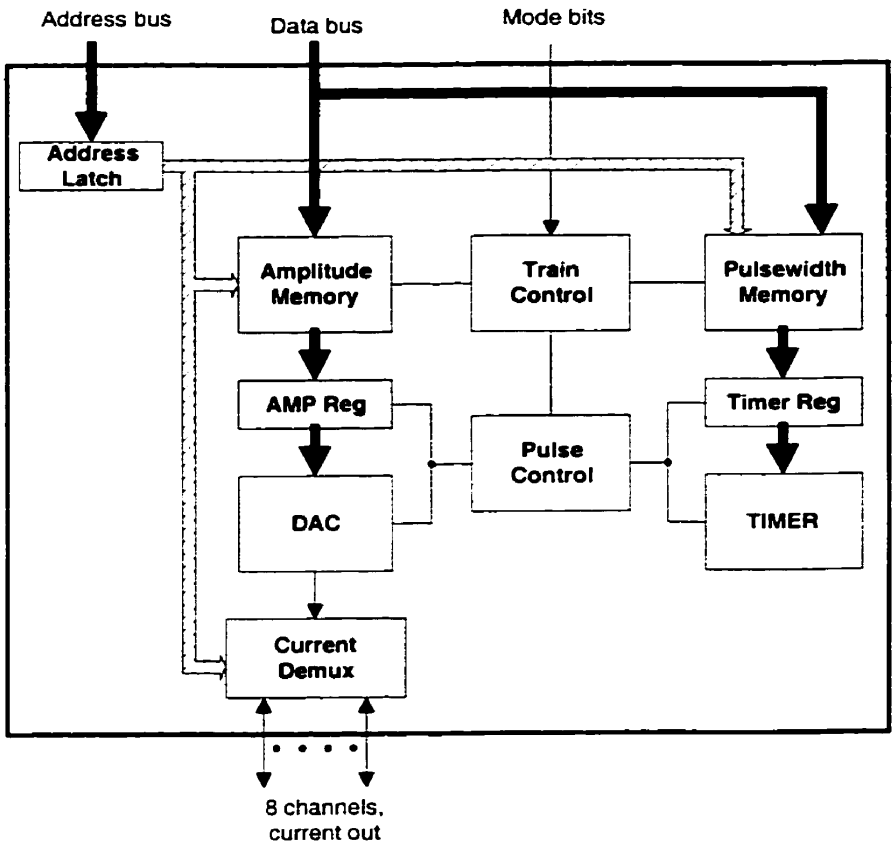


Figure 2.9 : Structure d'un canal de stimulation (DAC control element)

Le générateur de stimuli possède un CNA en mode courant, les deux RAM qui emmagasinent les paramètres de stimulation (amplitude et durée d'impulsion) et un circuit pour activer et désactiver la stimulation. Ici seule la forme d'onde carré est utilisée pour stimuler.

Au départ en opération normale, les données (amplitude et durée de l'impulsion) sont emmagasinées dans les RAM et les opérations de stimulation commencent. Chaque

canal stimule chacun des sites qui lui est associé à tour de rôle en prenant l'information dans les deux RAM. S'il n'existe aucune différence entre deux images consécutives, alors aucune information n'est envoyée à l'implant. Si une partie ou la totalité de l'image change, alors seulement les points de l'image qui changent sont envoyés via des mots de commande spécifiant entre autre le site où il y a changement, la nouvelle amplitude et finalement la nouvelle durée d'impulsion. Avec cette architecture, les opérations de stimulation peuvent être effectuées de façon continue car les paramètres de stimulation sont emmagasinées dans les RAM.

2.5.3 L'implant de Troyk

À ce jour, le projet le plus ambitieux en cours de réalisation est dirigé par Troyk [40] et consiste à développer un stimulateur cortical multi-canaux transcutané pour le National Institute of Neurological Disorders and Stroke affilié au National Institute of Health; un organisme gouvernemental de recherche dans le domaine de la santé aux États-Unis.

Une fois complété, le système comprendra un implant constitué d'un récepteur et transmetteur RF, de 256 canaux de stimulation et d'un connecteur de très grande densité (connexions avec 256 microelectrodes). Le module externe aura pour rôle de servir d'intermédiaire entre l'implant et un ordinateur de contrôle. Le système possédera en plus un système inverse de télémétrie. Un lien inductif sera utilisé pour le transfert d'information entre les deux modules ainsi que pour le transfert d'énergie du contrôleur externe à l'implant.

2.6 Efficacité d'une vision obtenue à l'aide d'un implant cortical

Essentiellement trois paramètres déterminent la qualité d'une image fragmentée en pixels : le nombre de pixels, leur densité et leur plage d'intensité [23]. Le nombre de pixels et leur densité obtenus à partir de l'implant cortical est directement lié au nombre d'électrodes insérées dans le cortex. Ce nombre est cependant limité par deux facteurs. Le premier est lié au fait qu'il y a des interactions entre deux phosphènes lorsque l'espacement entre deux sites de stimulation actifs est égale ou inférieure à 250 um [22].

Cette contrainte peut cependant être contournée en assurant un fonctionnement de l'implant où en aucun moment deux sites adjacents soient activés simultanément. Le deuxième facteur est d'ordre technologique, c'est-à-dire qu'il peut devenir difficile de réaliser une matrice d'électrodes de très grande densité à l'intérieur d'une certaine surface surtout si les électrodes de la matrice doivent avoir la robustesse nécessaire lors de l'implantation.

La topographie du cortex visuel est un autre élément qui affecte la qualité de la vision produite par une prothèse corticale. En effet, la localisation des phosphènes dans le champ visuel n'est pas seulement fonction de l'emplacement des électrodes mais aussi de plusieurs facteurs qui sont encore inconnus. Cependant, il semble y avoir un endroit précis dans le cortex où une relation linéaire existe entre la position d'un phosphène dans le champ visuel et la position d'un site de stimulation sur le cortex [16,22,23].

Les travaux de Cha, Horch et Normann [23] ont permis de conclure qu'un implant offrant 625 sites de stimulation (25x25) sur une surface totale de $1cm^2$ (1cmx1cm - correspond à 1.7 degrés du champs visuel) permettrait à un non-voyant de retrouver une vision de 20/30. L'espacement entre les sites de stimulation serait environ de 400um pour couvrir la surface de $1cm^2$.

2.7 Conclusion

Ce chapitre a premièrement donné une brève description du système visuel humain pour ensuite présenter les trois principales avenues de recherche dans le domaine des prothèses visuelles soit l'implant rétinien, le stimulateur du nerf optique et finalement l'implant cortical.

On a par la suite traité de la stimulation du cortex visuel et d'architecture d'implant cortical car la stimulation du cortex visuel semble être l'approche la plus prometteuse des trois méthodes présentées et c'est le type de système qui est actuellement en préparation par notre équipe de recherche (PolySTIM).

Le prochain chapitre traite de la conception matérielle des deux versions de prototype d'implant visuel cortical. On traitera de la structure générale de chaque version et des circuits analogiques qui les composent, alors que le chapitre 4 abordera la partie numérique.

CHAPITRE 3

CONCEPTION MATÉRIELLE DU PROTOTYPE

3.1 Introduction

Les deux premiers chapitres ont permis d'établir les principales tâches que doit accomplir la partie implantable d'une prothèse visuelle basée sur la stimulation du cortex visuel. Le présent chapitre propose donc deux architectures pour réaliser le prototype d'un implant visuel cortical (PIVC). Chaque conception possède une partie numérique et une partie mixte (analogique et numérique). Ici, l'objectif principal est de présenter les structures et d'expliquer les fonctionnements des deux topologies et de leurs sous-modules comportants des circuits mixtes. Le chapitre 4 présentera, par la suite, l'architecture de la partie numérique qui réalise le contrôleur principal du prototype. Chaque architecture permet d'effectuer les mêmes tâches qu'un éventuel implant visuel cortical (IVC). Ces tâches consistent principalement à recevoir des données d'un contrôleur externe et d'interpréter ces données pour exécuter des opérations de stimulation. Le système de réception RF (lien inductif) n'est cependant pas inclus dans chacun des prototypes, car sa conception pose de grands défis et fait l'objet de recherches de quelques membres de l'équipe PolySTIM. La deuxième architecture est un raffinement de la première, où une bonne partie des circuits mixtes sont intégrés d'une façon alternative à l'intérieur d'un seul et unique CI. Ceci permet de valider une technique analogique propice aux implants multicanaux.

Le but du prototype n'est pas de stimuler des cellules nerveuses, mais bien de donner un aperçu de ce que pourra voir une personne non-voyante à l'aide d'un implant visuel cortical. C'est pourquoi les opérations de stimulation dans le cas du prototype consistent à allumer et à éteindre des diodes électroluminescentes (DEL). Celles-ci sont disposées d'une façon à former une matrice carrée de 16 x 16 éléments. Cette disposition est

approximativement la même que celle des phosphènes générés dans le champ visuel d'un non-voyant à l'aide d'un réseau de 16x17 électrodes implanté dans le cortex visuel au niveau où est traité l'information provenant de la fovéa. Chacune des diodes simule donc l'apparition (diode allumée) et la disparition (diode éteinte) d'un phosphène associé à une paire d'électrodes. Précisons que le choix des dimensions de la matrice du prototype est arbitraire et basé essentiellement sur les caractéristiques et disponibilités des composants discrets commerciaux.

La prochaine section définit la structure commune que possède les deux prototypes. Le PIVC constitue le module interne d'un prototype complet du stimulateur visuel cortical (PSVC). La section suivante décrit la structure globale du PSVC et surtout des liens physiques de communication entre les deux modules qui le composent. Ensuite sont présentés tour à tour l'architecture et les détails des éléments analogiques et mixtes de chaque prototype. Seule la première architecture proposée fut fabriquée. Une conclusion résume les principales caractéristiques du prototype retenu pour fabrication, ainsi que ses différences architecturales par rapport à un éventuel IVC.

3.2 Élaboration de la structure de base des prototypes

Les connaissances actuelles sur la stimulation du cortex visuel et ses effets établissent en quelque sorte les fonctions que doivent accomplir l'implant et donc les fonctions du prototype. La principale tâche de l'implant est d'injecter un courant dans un certain site de stimulation à un instant donné. Habituellement, le courant est injecté sous forme de trains d'impulsions. Les différents paramètres d'un train agissent sur les caractéristiques (intensité, stabilité, ...etc.) d'un phosphène. Parmi ces paramètres, on retrouve l'amplitude du courant (AMP), la durée de phase (DP) et d'interphase (DIP) d'une impulsion, la durée du train (DT) et finalement, la durée inter-train (DIT). Pour réaliser ce genre de système, l'implant doit donc comporter au minimum un module de réception de l'information, un module qui interprète les commandes de stimulation reçues et finalement plusieurs canaux de stimulation. Les paragraphes qui suivent expliquent la raison d'être des canaux de stimulation. Le prochain chapitre traitera des autres modules.

Idéalement, l'implant devrait être en mesure de contrôler l'injection du courant dans chacun des sites indépendamment des opérations de stimulation en cours dans les autres sites. Des restrictions d'ordre techniques et biologiques empêchent toutefois de réaliser un tel système. En effet, un canal de stimulation comportant une source de courant et un circuit de contrôle serait nécessaire pour chaque site à stimuler. Un implant comportant un réseau de 25 x 26 électrodes aurait besoin alors de 625 canaux de stimulation, donc un circuit occupant une surface énorme et consommant beaucoup trop d'énergie.

Pour palier à ce problème, il faut réduire le nombre de canaux de stimulation et associer à chaque canal un certain nombre de sites de stimulation, qui peuvent être accédés séparément à l'aide d'un sélecteur. Le nombre de canaux de stimulation dépend de plusieurs facteurs, comme le nombre de sites de stimulation (nombre de paires d'électrodes), du temps de stimulation requis par les sites pour produire des phosphènes stables (DP et DIP) et du nombre d'impulsions par site par image. Par exemple, l'architecture du stimulateur visuel miniaturisé de Boyer [52] possède 25 canaux de stimulation, permettant de réaliser le balayage complet des sites au moins 23 fois par seconde pour un réseau de 650 électrodes (dans le cas où les durées d'impulsion sont maximales). Évidemment, la fréquence de balayage des sites peut être augmentée en diminuant les durées des impulsions de stimulation.

Les informations sur les expériences de stimulation du cortex visuel chez l'humain ne permettent pas pour l'instant de définir de façon précise les paramètres idéaux pour générer des phosphènes stables et clairs. Il est donc difficile de déterminer le nombre optimal de canaux de stimulation requis par l'implant. Le nombre minimal de canaux peut cependant être établi en supposant un temps de stimulation maximal (DP et DIP maximum) pour chaque site.

En ce qui concerne les deux prototypes proposés, le nombre de canaux fut fixé à 16 dans le cas de la première architecture et à 12 pour la seconde. Ici, le choix fut déterminé par les caractéristiques des éléments discrets permettant d'obtenir une bonne symétrie dans le circuit, facilitant sa conception et sa réalisation.

3.3 Structure du prototype du stimulateur visuel cortical (PSVC)

La figure 3.1 illustre les liens qui existent entre le contrôleur externe (CE) et le PIVC. Le contrôleur externe a comme tâche, dans un premier temps, l'acquisition et le traitement des images. Le traitement consiste principalement à améliorer les contrastes et à réduire la résolution des images, de façon à ce que chaque pixel de celles-ci corresponde à une diode électroluminescente de la matrice d'affichage du PSVC.

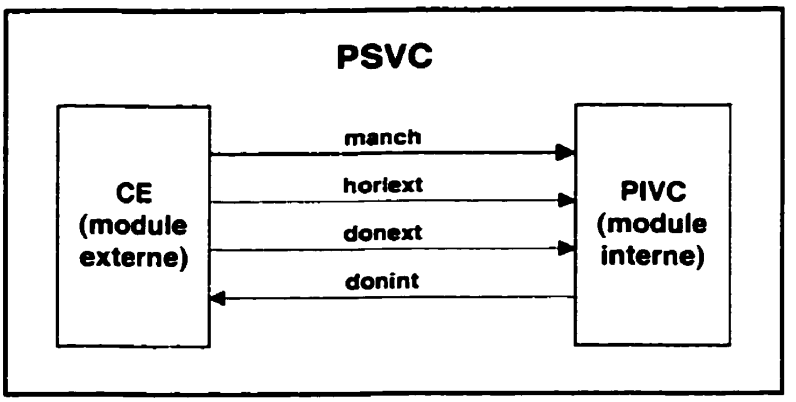


Figure 3.1 : Prototype du stimulateur visuel cortical

Le CE doit ensuite envoyer l'information visuelle au PIVC. Cette information est transmise à l'aide de mots de commande contenant des paramètres de stimulation. Les valeurs de ces paramètres sont déduites des valeurs des pixels des images captées, permettant ainsi de recréer ces dernières sur la matrice de diode. Grâce à l'utilisation du code Manchester, l'horloge et les données peuvent être transmis du CE au PIVC à l'aide d'un câble coaxial (une seule ligne ***manch***). Le fonctionnement du code est expliqué à la prochaine section. Le PIVC possède deux autres ports d'entrées pour d'éventuelles modifications, où il serait nécessaire d'avoir un signal d'horloge (ligne ***horlext***) sur un câble différent de celui des données (ligne ***donext***). Le PIVC est également muni d'un port de sortie (ligne ***donint***), qui pourrait être utilisé dans les prochaines conceptions. Le protocole de communication entre les deux modules est traité dans le prochain chapitre.

3.4 Architecture de la première version du PIVC

La figure 3.2 illustre le schéma bloc d'une première architecture d'un PIVC. Les figures A.1 à A.8 de l'annexe A présentent le circuit complet du prototype. Le PIVC est entièrement réalisé à partir de composants discrets commerciaux.

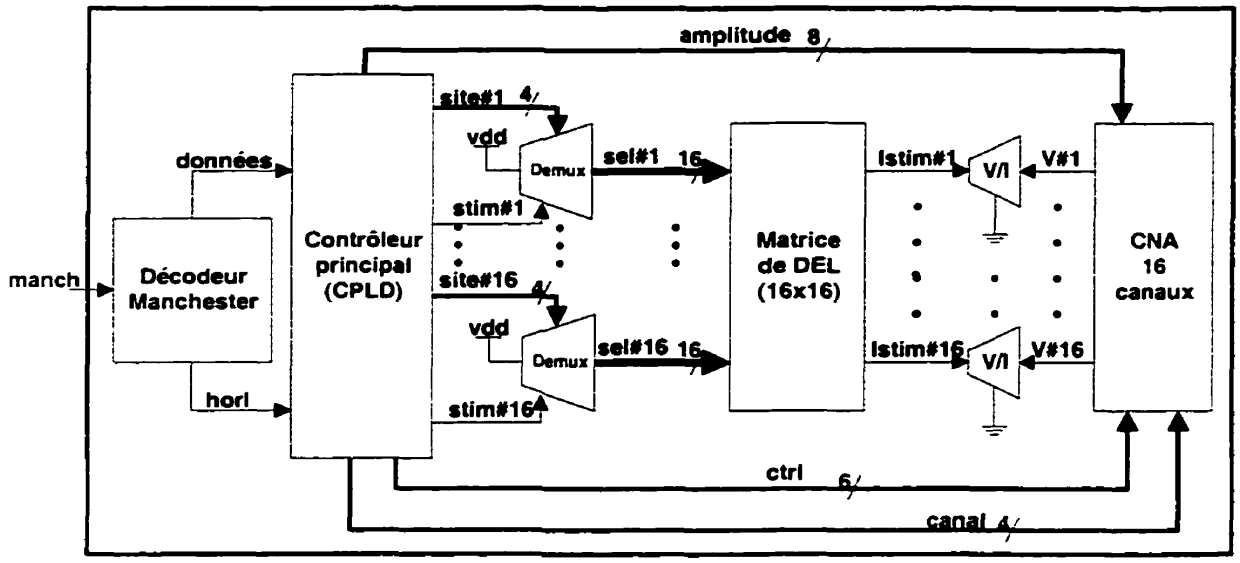


Figure 3.2 : Schéma bloc de la première architecture du PIVC

En mode d'opération normal, les données et le signal d'horloge provenant du CE sont envoyés sur la même ligne, ***manch***, à l'entrée du décodeur Manchester. Celui-ci produit deux signaux, soit un signal d'horloge et un signal de données qui sont appliqués à l'entrée du contrôleur principal. Ce dernier est responsable de la détection de trames de données contenant des mots de commande, de la détection et correction d'une éventuelle erreur de transmission par trame et finalement de gérer les opérations de stimulation (activation, désactivation des DEL). Le contrôleur est entièrement intégré dans un CI reprogrammable permettant de modifier et même de changer complètement sa fonctionnalité dans le futur.

Le prototype possède 16 canaux de stimulation, c'est-à-dire que 16 diodes électroluminescentes peuvent être activées simultanément. Chaque canal a la possibilité d'injecter un courant dans une seule DEL sur un choix de 16 qui lui sont réservés. Un

canal de stimulation comprend un contrôleur de canal (intégré dans le CPLD), un démultiplexeur pour la sélection d'une des 16 DEL, un CNA en mode tension et finalement un convertisseur tension-courant. Ce dernier permet de moduler l'intensité lumineuse de la DEL active. Tous les canaux partagent une seule puce regroupant 16 CNA. Chacune des sorties de la puce contrôle donc le courant d'un canal du PIVC.

Pour allumer une DEL en particulier, le contrôleur de canal envoie une requête de chargement d'amplitude au contrôleur de CNA (à l'intérieur du CPLD). Une fois l'amplitude chargée dans le CNA, celui-ci produit à la sortie associée au canal, une tension qui est alors convertie en courant. Ensuite, le contrôleur de canal envoie l'adresse du site (bus ***site#x***) et un signal de contrôle (***stim#x***) au démultiplexeur du canal sélectionné activant la DEL. Les 16 diodes électroluminescentes d'un canal sont disposées dans une configuration en cathodes communes. Les cathodes sont connectées à la sortie du convertisseur tension-courant. Chaque sortie du démultiplexeur est branchée à l'anode d'une seule DEL (lignes **sel#x**). Lorsqu'une sortie du démultiplexeur est activée, celle-ci passe d'un niveau de tension nul à un niveau suffisant pour faire passer un courant dans la DEL choisie.

Chaque bloc du schéma de la figure 3.2 correspond à un circuit discret commercial à l'exception du décodeur Manchester et des convertisseurs tension-courant. Les deux prochaines sections présentent donc et expliquent le fonctionnement des circuits du décodeur Manchester et du convertisseur tension-courant ainsi que du système de sélection d'une DEL à l'intérieur d'un canal.

3.4.1 Décodeur Manchester

Le circuit du décodeur Manchester est illustré à la figure 3.3. Ce circuit génère un signal d'horloge (***horl***) et un signal de données (***données***) à partir du signal numérique d'entrée ***manch***. Un exemple des formes d'onde de l'entrée et des sorties est montré à la figure 3.4.

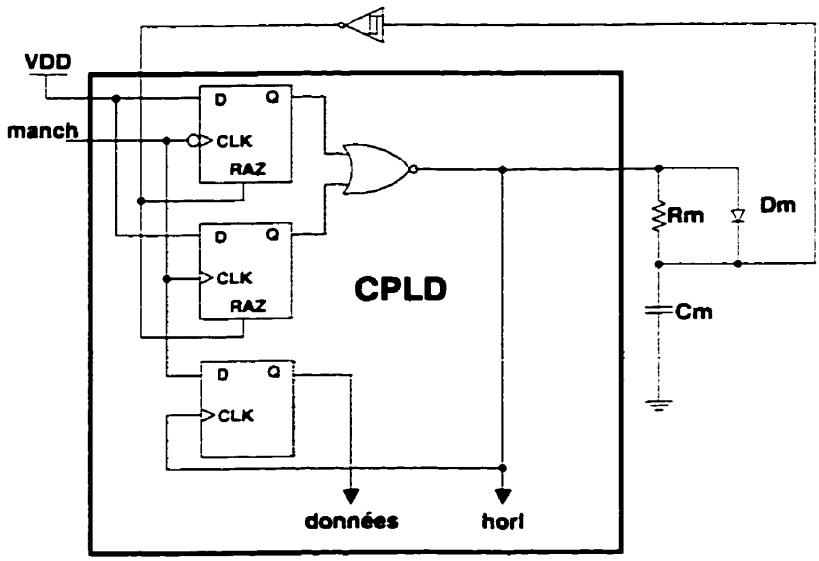


Figure 3.3 : Décodeur Manchester (tiré de [80])

Un signal Manchester utilise la deuxième moitié d'un cycle d'horloge pour représenter une valeur binaire. Lorsqu'il y a une transition dans le signal pendant un cycle complet, la valeur de la donnée ne change pas. Lorsqu'il n'y a pas de transition, cette valeur change.

Le décodeur utilise une première transition du signal ***manch*** pour amener la ligne ***horl*** à zéro (une transition amène un « 1 » à l'une des deux entrées de la porte NOR donc un « 0 » à sa sortie). À ce moment, le condensateur Cm se décharge dans la résistance Rm. Lorsque la tension aux bornes de Cm arrive au seuil du niveau bas, la sortie de l'inverseur de Schmitt est amenée à « 1 », ce qui amène un « 0 » aux deux entrées de la porte NOR et ramène donc la ligne d'horloge à « 1 ».

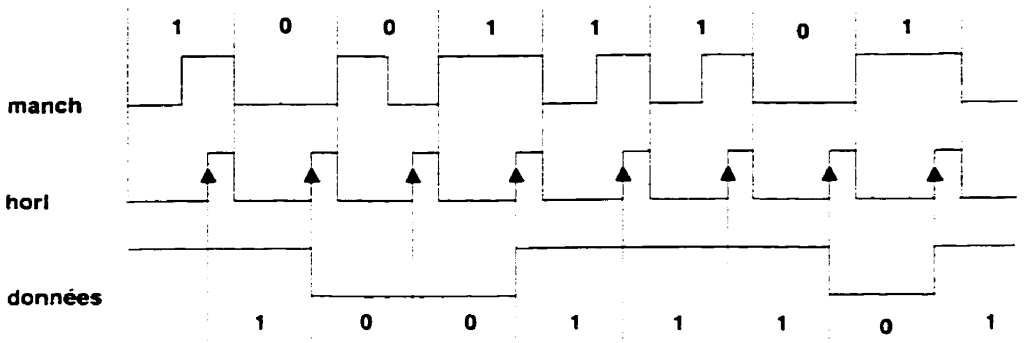


Figure 3.4 : Exemple de formes d'onde des E/S du décodeur Manchester

Les valeurs de Cm et Rm sont choisies de façon à ce que la tension à Cm atteigne le seuil du niveau bas à environ 3/4 de la période de l'horloge. Ceci permet de faire l'échantillonnage de la donnée présente dans le signal ***manch*** (l'échantillonnage est effectué au front montant du signal ***horl***) dans la deuxième moitié de la période d'un cycle d'horloge. La diode Dm permet de charger Cm rapidement lorsque le signal d'horloge est mis à « 1 ». Les signaux ***données*** et ***horl*** sont acheminés au contrôleur principal (à l'intérieur du CPLD). Précisons que les composants numériques du décodeur sont implantés à l'intérieur du même CPLD.

3.4.2 Sélection d'une DEL et contrôle de l'intensité du courant

La figure 3.5 présente le circuit utilisé pour la sélection d'une DEL et pour modifier au besoin l'amplitude de son courant. Chacune des sorties du démultiplexeur est connectée à l'anode d'une seule diode électroluminescente. Lors de l'activation d'une DEL en particulier, le démultiplexeur amène l'une de ses 16 sorties au niveau haut qui correspond à la valeur présente sur le bus ***site***. Le signal ***stim*** permet de laisser passer ou d'empêcher un courant dans la DEL sélectionnée en activant ou désactivant le démultiplexeur.

Le courant dans la DEL active est contrôlé à l'aide du convertisseur tension-courant formé à partir d'un amplificateur opérationnel (A0), un transistor JFET (Q1), un transistor NPN (Q2) et deux résistances (R1 et R2). Admettant une grande impédance

d'entrée de l'ampli opérationnel, tout le courant ***Id*** de la DEL active doit passer par la résistance R1. La tension aux bornes de R1 est donc proportionnelle au courant ***Id***. L'amplificateur opérationnel est utilisé dans une boucle de contre-réaction avec les autres composants du convertisseur V/I pour justement permettre de modifier la valeur de ***Id*** selon la référence ***V*** (provenant d'une des 16 sorties du CNA).

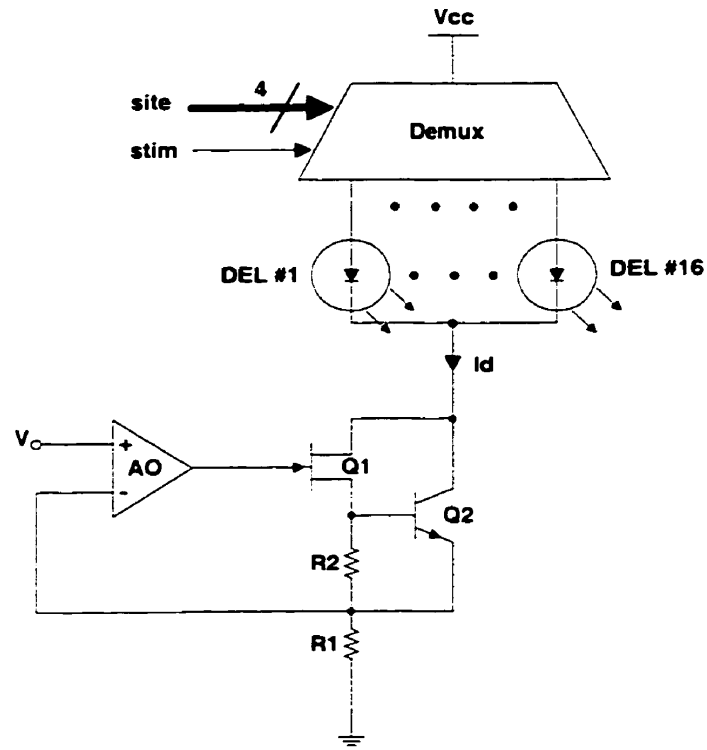


Figure 3.5 : Sélection d'une DEL et contrôle de l'intensité de son courant

Une fois le système en équilibre, la tension à l'entrée négative de l'ampli (donc aux bornes de R1) sera approximativement égale à la tension ***V*** présente à son entrée non-inverseur. Le rapport de conversion est donné par l'équation suivante:

$$Id = \frac{V}{R1} \tag{3.1}$$

La résistance R2 limite le courant qui passe dans le transistor JFET Q1, car le courant de drain maximal de celui-ci ne doit pas dépasser une certaine valeur. Ainsi, à partir d'une certaine valeur, le courant Id passe par Q2 plutôt que par Q1 [72].

3.5 Architecture de la deuxième version du PIVC

La figure 3.6 illustre le schéma bloc de la deuxième architecture. Ici le nombre de canaux ainsi que le nombre de diodes électroluminescentes par canal passent de 16 à 12. Cette architecture ressemble à la première à l'exception que cette fois-ci, la majorité des fonctions analogiques sont intégrées dans une seule puce nommée CEB (CNA et échantillonneur-bloqueur) réalisée en technologie prédiffusée. Le CEB fut réalisé à partir d'un procédé de fabrication bipolaire prédiffusé comportant une seule couche de métal. La capacité de manipuler des courants importants et la rapidité de fabrication sont les principales raisons qui nous ont amenés à choisir cette technologie pour implanter la fonctionnalité voulue.

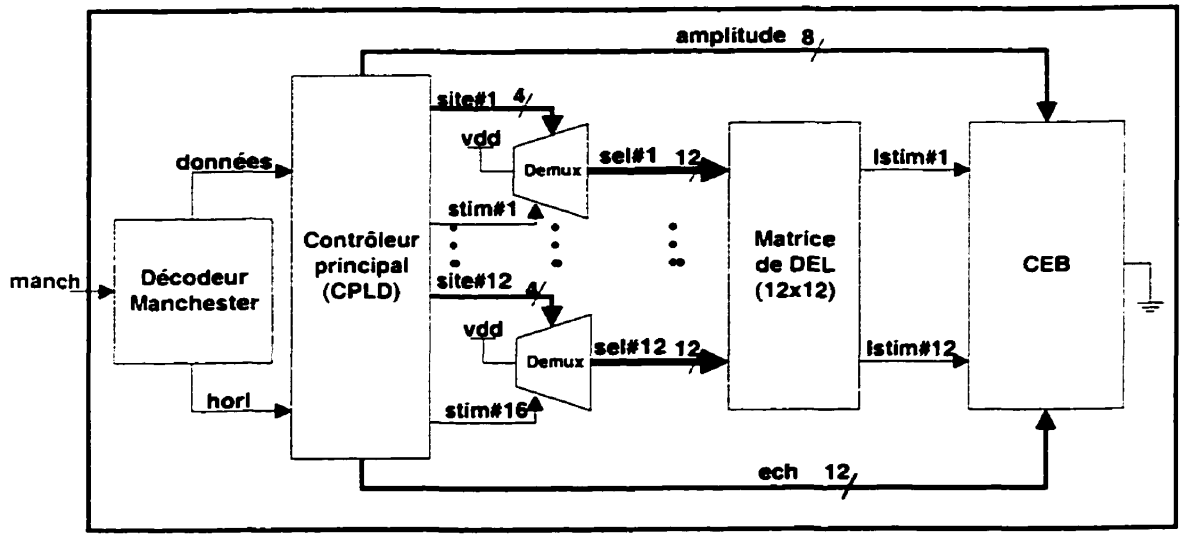


Figure 3.6 : Schéma bloc de la deuxième architecture du PIVC

Le CEB comprend un CNA et 12 échantillonneurs-bloqueurs (EB) en mode courant. Le fonctionnement général de cette seconde topologie est le même que la première à l'exception qu'une étape d'échantillonnage est ajoutée dans le processus de contrôle des

courants des diodes électroluminescentes. Ainsi le CPLD dans un premier temps présente au CNA un mot de 8 bits correspondant à l'amplitude du courant de stimulation et qui produit à l'aide du bloc V/I le courant recherché. Une des 12 lignes de commande d'échantillonnage (***ech***) devient alors active permettant à un seul des 12 EB de mémoriser ce courant. Lorsque le signal est désactivé, le courant de sortie ***Istim*** du canal est alors égal à la valeur du courant précédemment échantillonné.

La prochaine section explique le principe de fonctionnement général du CEB. Les sections suivantes décrivent les principaux circuits qui le composent ainsi que leur fonctionnement. Les figure B.1 à B.10 de l'annexe B présentent les schémas détaillés de tous les circuits constituant le CEB.

3.5.1 Principe de fonctionnement du CEB

La figure 3.7 illustre le schéma bloc simplifié du CEB. Le CNA produit à sa sortie un courant ***Icna*** proportionnel au mot de 8 bits ***b0-b7*** présent à son entrée. Une référence en tension externe (***Vref***) permet également de fixer la valeur maximale de ce courant.

Un miroir de courant copie le courant ***Icna*** dans ***Im*** qui est ensuite acheminé à un des 12 EB. En mode échantillonnage (signal ***ech#1*** à « 0 »), l'EB mémorise alors le courant ***Im***. Lorsque ***ech*** passe à « 1 », l'EB revient en mode maintien et tire un courant ***Istim#1*** égal à ***Im***.

Le circuit de mémorisation du courant est basé sur la technique du miroir de courant dynamique largement utilisé en CMOS [73]. Dans cette technique, il est possible de mémoriser un courant en chargeant avec une tension correspondante une capacité placée à la grille d'un transistor NMOS, dont la source est branchée à la masse. Le circuit de la figure 3.8 est un exemple simple de miroir de courant dynamique réalisé à l'aide d'un transistor NMOS. En mode échantillonnage, les commutateurs INT#1 et INT#2 sont fermés et INTM est ouvert.

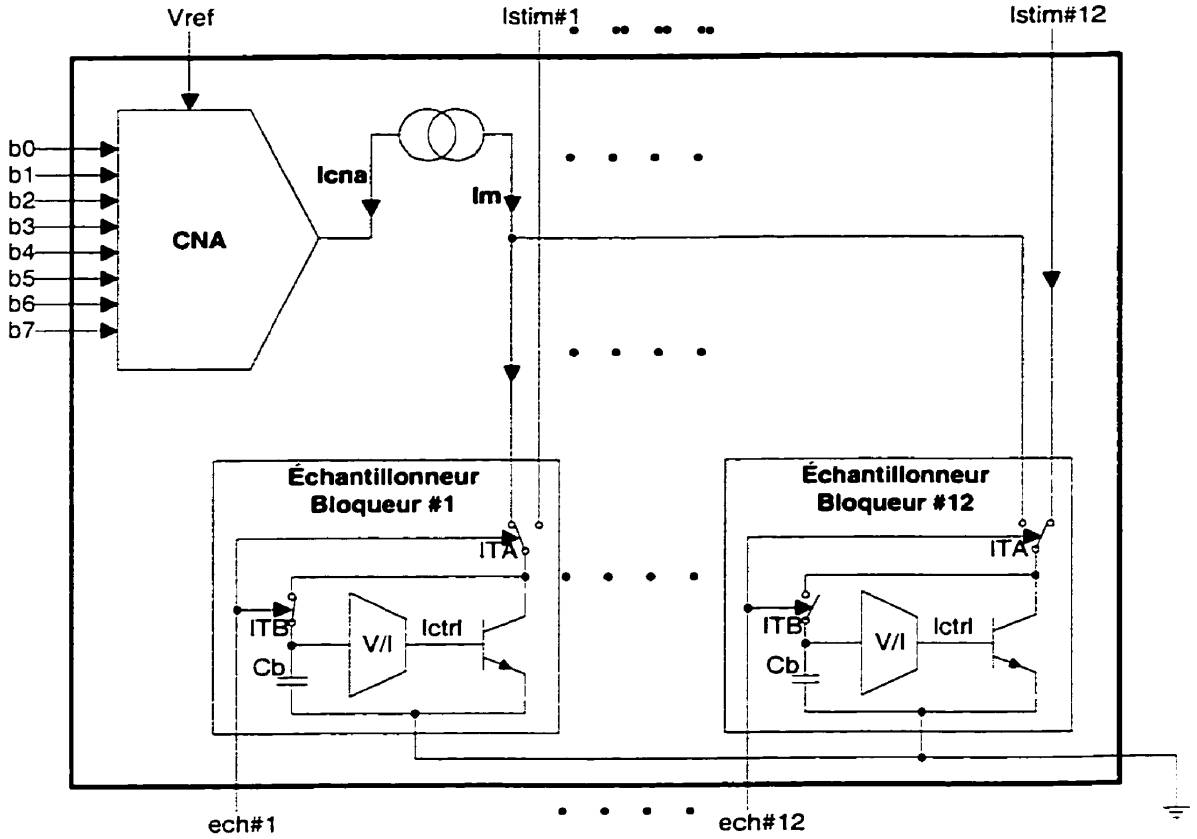


Figure 3.7 : Schéma bloc simplifié du CEB

La source de courant ***Iref*** force alors la tension V_{gs} à la grille du transistor NMOS de prendre une valeur qui permet de laisser passer la totalité du courant (***Iref***) dans le transistor. Cette tension est emmagasinée dans Cm. Lorsqu'un transistor MOS fonctionne dans la région active, la valeur du courant de drain est donné par l'équation suivante :

$$I_d = \frac{\mu_n C_{ox}}{2}\left(\frac{W}{L}\right)\left(V_{gs} - V_{tn}\right)^2 \tag{3.2}$$

où μ_n, C_{ox}, W, L et V_{tn} sont des constantes pour un transistor donné. Le courant Id est donc fixé uniquement par V_{gs}. En mode maintien, les interrupteurs INT#1 et INT#2 deviennent ouverts et l'interrupteur INTM se ferme. En admettant que le transistor

NMOS fonctionne dans la région active, le courant ***Isortie*** dépend alors seulement de Vgs (précédemment emmagasinée dans Cm). Le courant ***Isortie*** est donc égal à ***Iref***.

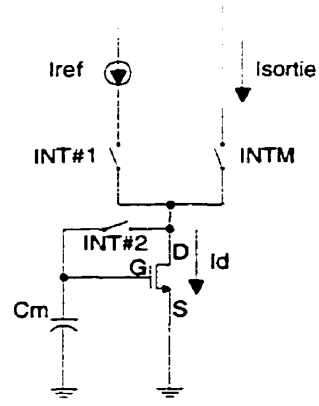


Figure 3.8 : Miroir de courant dynamique

L'EB fonctionne suivant le même principe. Dans le circuit de la figure 3.7, le courant échantillonné résulte d'une tension développée aux bornes de la capacité Cb. Cette tension, convertie en un courant ***Ictrl***, contrôle le courant recherché du EB. Une fois l'échantillonnage effectué, le commutateur ITB s'ouvre et ITA branche la sortie du EB au reste du circuit (les cathodes des DEL d'un canal). Le courant de sortie est pratiquement le même que le courant à l'échantillonnage, car la tension de contrôle est emmagasinée dans le condensateur Cb.

3.5.2 L'échantillonneur-bloqueur du CEB

Le circuit de l'échantillonneur-bloqueur est illustré à la figure 3.9. Le commutateur ITB (de la figure 3.7) est formé à partir des transistors Q1 à Q5. Lorsque ITB est fermé, un courant provenant de la ligne ***ctrl*** active Q1. Ceci a pour effet de désactiver Q2. Les transistors Q3, Q4 et Q5 et la source de courant ***Ipd*** forment un suiveur avec tension de décalage Vbe (Q3) compensée [74]. La tension à la base de Q5 est alors la même que celle présente à la base de Q3. Pour ouvrir l'interrupteur dans le but de passer dans le mode de maintien, il suffit simplement d'interrompre le courant de polarisation à la base de Q1. À ce moment, Q2 devient actif et amène la tension présente aux émetteurs de Q3

et Q5 à un niveau supérieur à leur tension de base. Ces deux transistors sont à ce moment ouverts et la base de Q5 de même que la capacité Cb sont alors isolées.

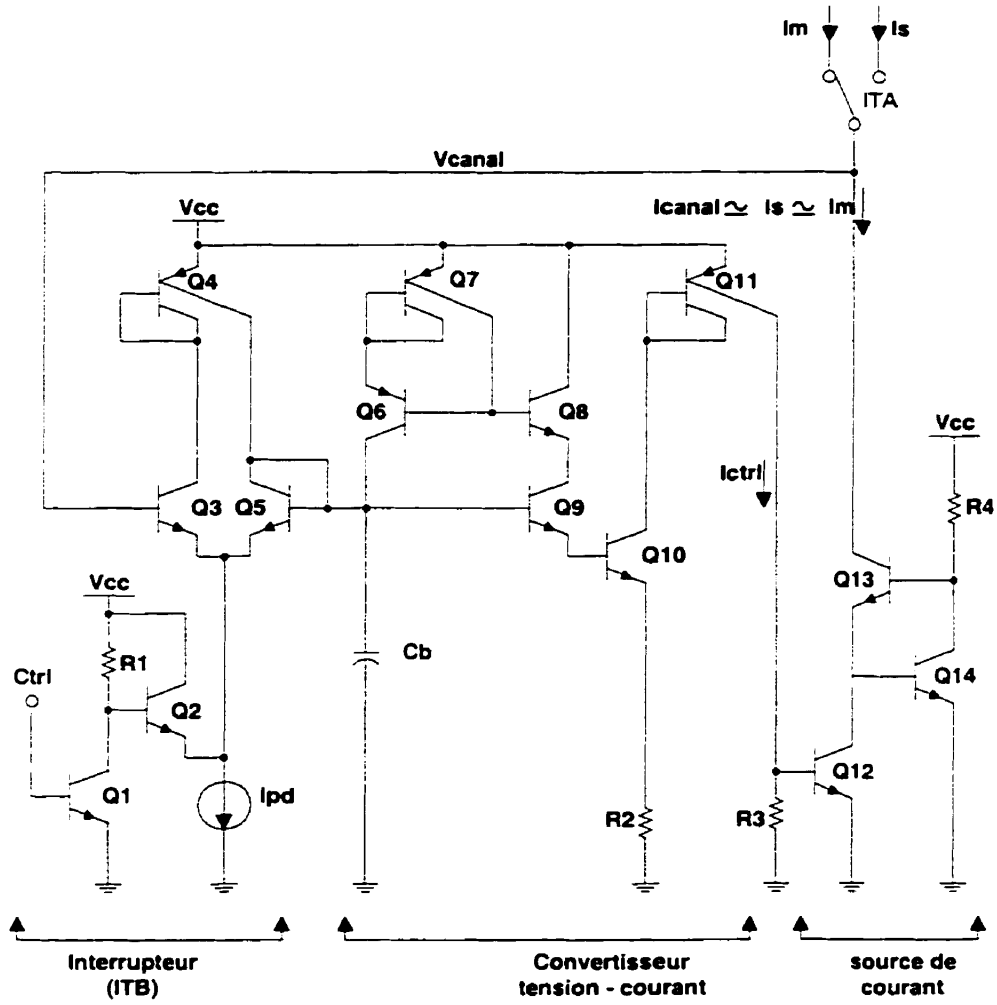


Figure 3.9 : Échantillonneur- bloqueur en mode courant

Pendant la période d'échantillonnage, la tension de Q5 est donc approximativement la même que celle présente à l'entrée de l'échantillonneur. Cette tension, emmagasinée dans Cb, est convertie en un courant à l'aide notamment des transistors Q9 et Q10. Ces transistors forment un suiveur dont la tension de sortie (émetteur de Q10) appliquée à la résistance R2 donne un courant qui est alors copié par Q11. Le courant résultant ***Ictrl***

pilote la base de Q12, qui une fois le système stabilisé, laisse passer presque la totalité du courant ***Im***.

Les transistors Q6 à Q8 forment un circuit de cancellation du courant de polarisation du suiveur permettant de réduire considérablement le courant nécessaire pour piloter Q9 et donc de maintenir à un niveau acceptable la tension emmagasinée dans Cb pour une période relativement longue. Ce type de circuit est souvent utilisé à l'étage d'entrée des amplificateurs opérationnels bipolaires [75]. Le courant d'émetteur de Q8 étant approximativement égal au courant d'émetteur de Q9, les deux transistors ont donc approximativement le même courant de base. Le courant de base de Q8 est copié à l'aide du miroir de courant formé à partir de Q7 et Q6 et finalement appliqué à la base de Q9 permettant de réduire de façon importante le taux de décharge de Cb.

Les transistors Q13 et Q14 ont pour fonction d'augmenter l'impédance de sortie de la source de courant. Sans ces transistors, le courant qui circulerait dans Q12 serait fonction de ***Ictrl*** mais aussi de la tension Vce de Q12.

Avec les deux transistors ajoutés, la tension Vce de Q12 est maintenue approximativement constante et égale à la tension Vbe de Q14. Cette technique est couramment utilisée en technologie CMOS [76] pour augmenter l'impédance de sortie d'un miroir de courant. Elle comporte cependant un inconvénient si utilisé en bipolaire car un léger courant est nécessaire à la polarisation de Q14. Dans la présente application, la valeur de R4 fut choisie de façon à ce que le courant de polarisation de Q14 soit négligeable par rapport au courant ***Im***.

3.5.3 Le commutateur de courant ITA

Le schéma bloc du commutateur ITA est illustré à la figure 3.10. Il est constitué principalement de deux commutateurs de courant (int#1 et int #2) qui partagent le même point de sortie. Un circuit en logique TTL commande leur ouverture et leur fermeture. La difficulté dans la réalisation d'un commutateur de courant formé à partir de transistors bipolaires vient du fait que ceux-ci ont besoin d'un courant de polarisation

pour devenir passant. Ce courant est injecté dans la sortie du commutateur ce qui « fausse » la valeur du courant de sortie.

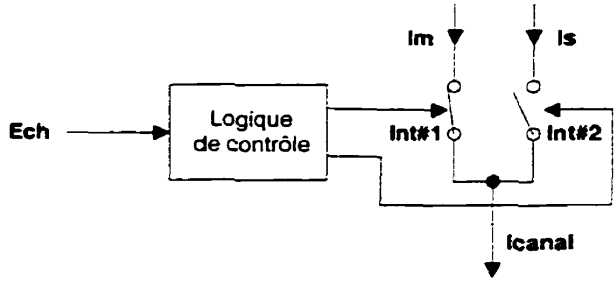


Figure 3.10 : Schéma bloc du commutateur ITA

La figure 3.11 montre le circuit qui vient à bout de cette difficulté. Lorsqu'un courant est injecté dans Q1 et Q2, la base de Q6 est amenée à une tension inférieure à son émetteur et le commutateur est à ce moment non actif. Le transistor Q3 alors polarisé, laisse passer un courant dans R4.

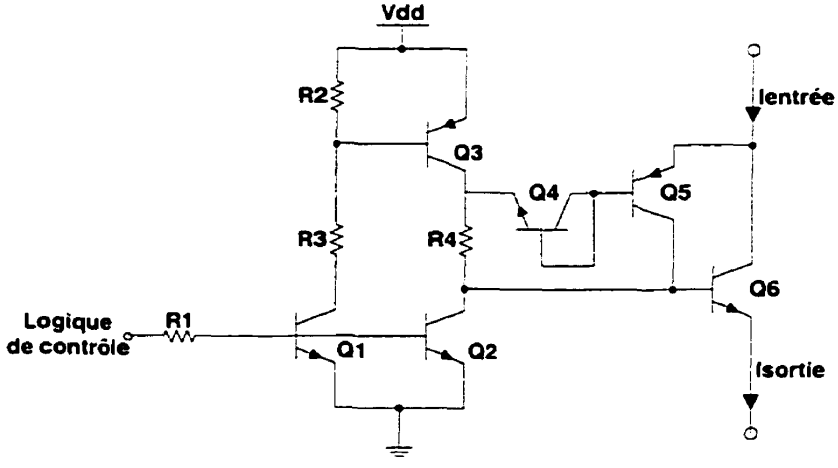


Figure 3.11 : Circuit d'un commutateur de courant

La tension développée aux bornes de R4 désactive Q4 et Q5; il n'y a donc aucune fuite possible d'un courant à l'entrée de l'interrupteur. Lorsque la tension aux bases de Q1 et Q2 est amenée à zéro, ceux-ci et Q3 deviennent bloqués et l'interrupteur laisse passer le

courant. Les transistors Q4,Q5 et Q6 fonctionnent alors comme un simple transistor branché en diode.

3.5.4 Le CNA du CEB

La figure 3.12 montre le circuit simplifié du CNA en mode courant. Celui-ci réalisé à partir d'une architecture maître-esclave possède deux réseaux de résistances R-2R, soit un réseau primaire utilisé pour les quatre bits les plus significatifs et un réseau secondaire utilisé pour les quatre bits les moins significatifs.

Les transistors Q2 à Q11 forment des sources de courant très précises. La source formée à partir de Q2 fournit le courant le plus important (MSB) et celle formée à partir de Q10 fournit le courant le plus faible (LSB). Le courant ***Icna*** résultant vient de l'addition des courants de chaque source sélectionnée par les commutateurs ***int1*** à ***int8*** à l'aide du bus d'amplitude ***b7-b0***.

La taille du transistor de chaque source doit être proportionnelle au courant qui le traverse pour maintenir une densité de courant égale à l'émetteur de chaque transistor (source de courant) et ainsi éviter les variations de Vbe entre eux. Une architecture de CNA similaire mais avec un seul réseau R-2R nécessiterait que le transistor MSB occupe une surface 128 fois plus élevée que la surface occupée par le transistor LSB. L'utilisation d'une architecture maître-esclave permet justement de réduire les dimensions totales du circuit car le rapport des dimensions des transistors n'est important qu'à l'intérieur d'un seul et même réseau. La taille du transistor Q7 est donc huit fois plus élevée que celle de Q10 et la taille de Q2 huit fois plus élevé que celle de Q5.

La tension Vpol est générée à l'intérieur du CI. L'ampli opérationnel, le transistor Q1 et les résistance Rref, R1 et R2 permettent de contrôler l'amplitude des sources de courant à partir de la référence en tension ***Vref***.

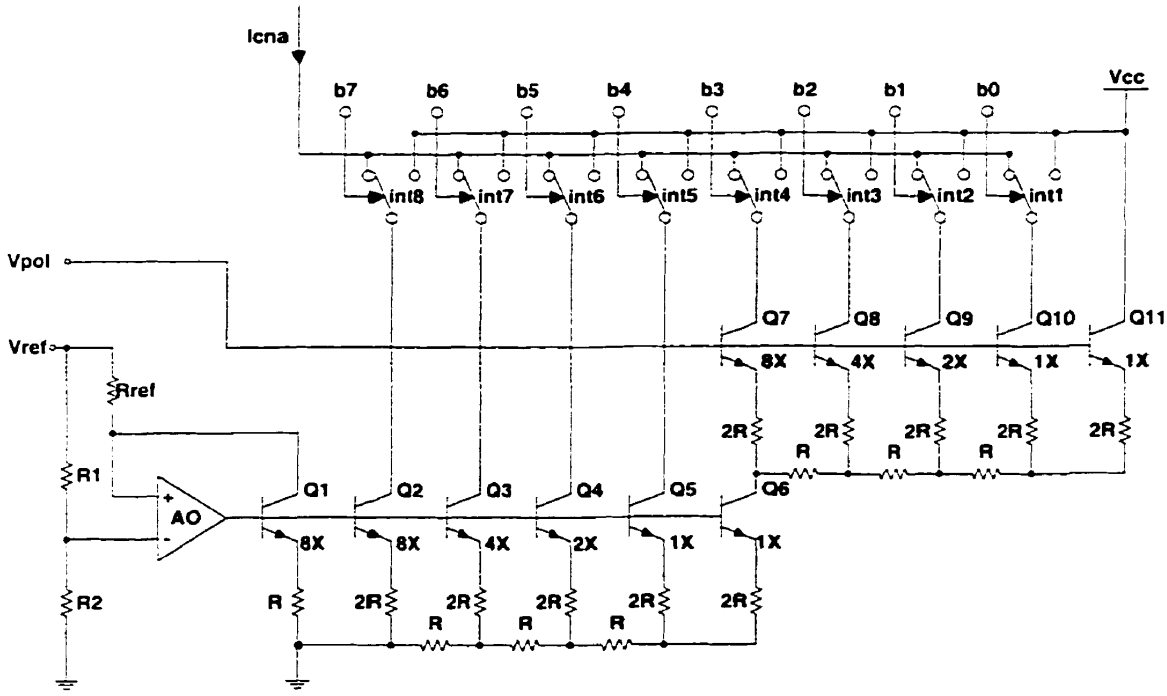


Figure 3.12 : Schéma simplifié du convertisseur numérique-analogique

Ces éléments forment une boucle de contre-réaction qui maintient l'entrée négative de l'ampli opérationnel à celle présente à son entrée positive. Ceci permet de contrôler le courant qui circule dans Q1 selon la relation suivante:

$$I_{Q1} = \frac{V_{ref}\left(1 - \frac{R2}{R2 + R1}\right)}{R_{ref}} \tag{3.3}$$

et donc de contrôler les courants de chaque source du réseau primaire. La valeur du courant de sortie ***Icna*** est donnée par l'équation suivante :

$$I_{cna} = \frac{V_{ref}\left(1 - \frac{R2}{R2 + R1}\right)}{R_{ref}}\left(\frac{b_7}{2} + \frac{b_6}{2^2} + \frac{b_5}{2^3} + \frac{b_4}{2^4} + \frac{b_3}{2^5} + \frac{b_{21}}{2^6} + \frac{b_1}{2^7} + \frac{b_0}{2^8}\right) \tag{3.4}$$

Le circuit d'un commutateur utilisé dans le CNA est présenté à la figure 3.13. Puisque le convertisseur utilise seulement une alimentation positive, les niveaux de tension requis pour contrôler les commutateurs doivent être augmentés. Ceci est réalisé à l'aide du transistor Q1.

Le commutateur de courant est formé à partir de deux paires de transistors branchées dans une configuration darlington. Lorsque la tension à la base de Q2 (donc lorsque bx = 0) est supérieure à la tension Vlogique présente à la base de Q5, les transistors Q2 et Q3 deviennent passant et Q4 et Q5 deviennent bloqués. La source de courant associée à l'interrupteur est alors connectée à la sortie du CNA. Lorsque la tension à la base de Q2 (bx = 1) est inférieure à Vlogique, la sortie du CNA est alors déconnectée de la source de courant.

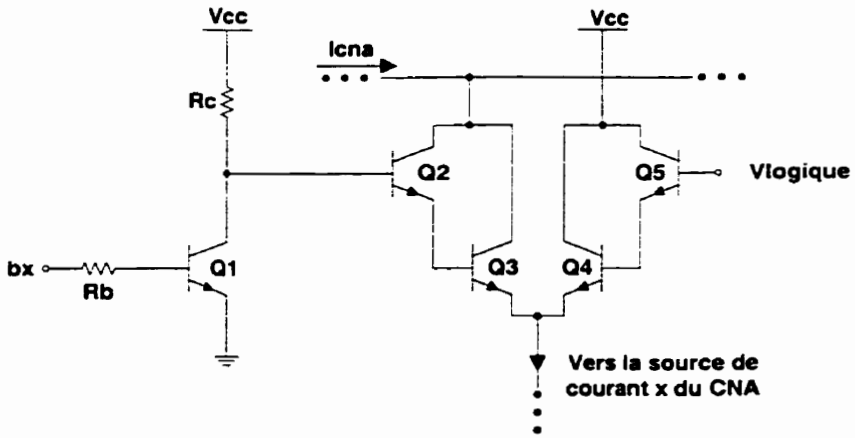


Figure 3.13 : Schéma d'un commutateur utilisé dans le CNA

3.6 Conclusion

Nous avons présenté dans ce chapitre deux conceptions pour réaliser le prototype d'un implant visuel cortical. Les tâches des prototypes consistent à recevoir des commandes d'un contrôleur externe et ensuite d'injecter des courants à des diodes électroluminescentes par l'entremise de canaux de stimulation. La fonction de chaque DEL est d'émuler la sensation de phosphène perçu par un non-voyant suite à l'injection

d'un courant dans son cortex visuel. Les deux prototypes possèdent chacun une partie numérique (un contrôleur) et une partie analogique-mixte réalisée à partir de circuits discrets commerciaux et d'un CI prédiffusé pour la deuxième architecture.

Compte tenu de la complexité du deuxième prototype et des résultats obtenus avec le prédiffusé, seule la première version fut complètement réalisée. Le prototype retenu pour fabrication possède approximativement les mêmes modules que l'éventuel implant cortical à l'exception du lien inductif et du système de permutation du courant, ici inutile car le courant ne peut voyager que dans un seul sens à l'intérieur d'une diode électroluminescente.

Le prochain chapitre traite de l'architecture du contrôleur (partie numérique) du prototype retenu pour fabrication.

CHAPITRE 4

LE CONTRÔLEUR DU PIVC

4.1 Introduction

Les deux architectures de PIVC furent décrites au cours du chapitre précédent. Le prototype fabriqué possède deux parties, soit l'ensemble des circuits mixtes (analogique et numérique) traités précédemment et le circuit numérique de contrôle entièrement intégré dans un CPLD. Le présent chapitre aborde donc l'architecture du circuit numérique. Ce circuit contrôle les opérations des 16 canaux de stimulation selon les commandes provenant du contrôleur externe (CE). Puisque le CPLD est facilement reprogrammable, le circuit numérique pourra être modifié ou complètement changé dans l'avenir pour satisfaire à de nouvelles exigences.

Dans les prochaines pages, nous décrivons les modes de fonctionnement du PIVC, le protocole de communication établi entre les deux modules du PSVM (externe et interne) et l'architecture générale ainsi que des modules importants du circuit numérique.

4.2 Les modes de fonctionnement

De nos jours, les stimulateurs deviennent de plus en plus complexes et permettent d'effectuer plusieurs fonctions. Parmi ces fonctions, on retrouve divers opérations de stimulation, de mesure de signaux biologiques ainsi que le test de l'intégrité physique de l'implant. De plus, il peut être nécessaire de configurer l'implant pour optimiser son fonctionnement et pour lui donner plus de flexibilité dans ses opérations de stimulation.

Pour permettre à un implant d'accomplir ces différentes tâches, il peut être judicieux de lui définir des modes de fonctionnement. Chaque mode correspond à une tâche bien précise exécutée par l'implant. Une fois les modes de fonctionnement définis, un

protocole de communication entre les deux parties du stimulateur peut être établi. Le protocole sera alors optimisé en fonction des exigences du système. L'élaboration d'un protocole de communication est un aspect très critique lors de la conception d'un tel système. En effet, la communication doit être efficace pour limiter la quantité d'information à transférer et ainsi diminuer la fréquence d'opération de l'implant et donc la quantité d'énergie irradiée dans les tissus biologiques. Cependant, les modes de fonctionnement et le protocole de communication doivent également offrir la possibilité de modifier rapidement les paramètres de stimulation pour assurer une stimulation efficace et sécuritaire.

Le PIVC doit idéalement effectuer les mêmes fonctions que la version miniaturisée de l'implant. La principale tâche qui fait l'objet du présent projet consiste à commander des opérations de stimulation (injection de courant) aux 16 canaux. D'autres fonctions comme le test de l'intégrité de l'implant et de l'interface électrode-électrolyte font l'objet de travaux d'un autre membre de notre équipe de recherche.

La philosophie de base dans l'élaboration de la structure du système est de maximiser le traitement réalisé par le contrôleur externe pour ainsi réduire au minimum le travail à être effectué par le PIVC. Ceci permet de simplifier l'architecture du prototype et donc de réduire sa consommation. De plus, lorsqu'un contrôle maximal est requis de la part du CE, celui-ci peut adresser directement chaque site et envoyer tous les paramètres pour commander une opération de stimulation. Dans ce cas, la quantité d'information à transférer est maximale. Il sera toutefois possible de réduire le taux de transfert d'information, en configurant le PIVC de façon à définir quelques paramètres de stimulation communs à tous les sites tout en précisant la séquence d'accès aux sites. De cette façon, seuls les paramètres spécifiques à chaque site doivent être transmis.

Le PIVC a donc deux modes de fonctionnement : configuration et stimulation. Le mode configuration permet de définir quels paramètres de stimulation seront communs à tous les sites, lesquels seront spécifiques, et permet de définir aussi la façon d'adresser ces sites (prédéfinis ou spécifiques). Ce mode permet également de définir le nombre de bits

utilisés pour spécifier chacun des paramètres et de charger en mémoire la séquence d'accès (de balayage) des sites de stimulation. Le mode stimulation initie les opérations de stimulations suite à la réception d'un ou plusieurs paramètres spécifiques.

4.3 Le protocole de communication

Les données sont envoyées du contrôleur externe au prototype de l'implant de façon sérielle par trame de 31 bits comme illustré à la figure 4.1. Chaque trame est précédée d'un bit de départ « 0 » et suivi par un ou plusieurs bits d'arrêts « 1 ».

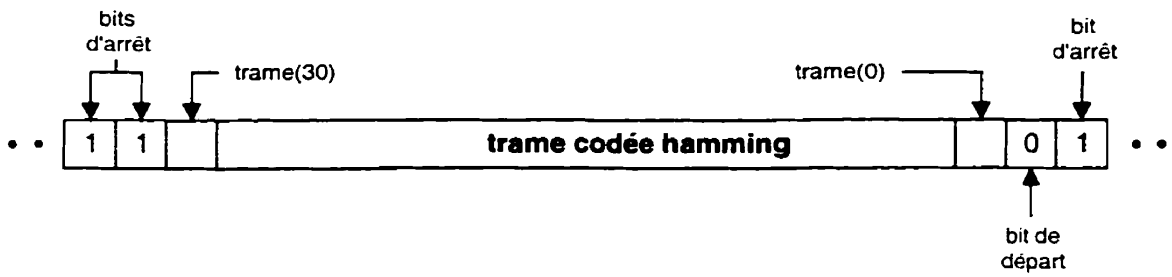


Figure 4.1 : Format d'une trame de données

La trame contient 26 bits de données formant une commande et 5 bits de parité. Les bits de parité sont disposés dans la trame de façon à ce que celle-ci forme un mot de 31 bits codé Hamming (figure 4.2). Le code de Hamming fut choisi comme moyen de détection et de correction d'erreur car il est simple à réaliser et le nombre de bits de redondance qui doivent être ajoutés est minimal comparativement aux autres méthodes de détection et correction d'erreur.

Un mot de commande (MCMD) comporte deux champs : le champ mode (1 bit) et le champ information (25 bits). Selon la valeur du bit de mode (1er bit du mot de commande), le MCMD est défini comme étant un mot de configuration (MCFG) ou un mot de paramètre (MPMT).

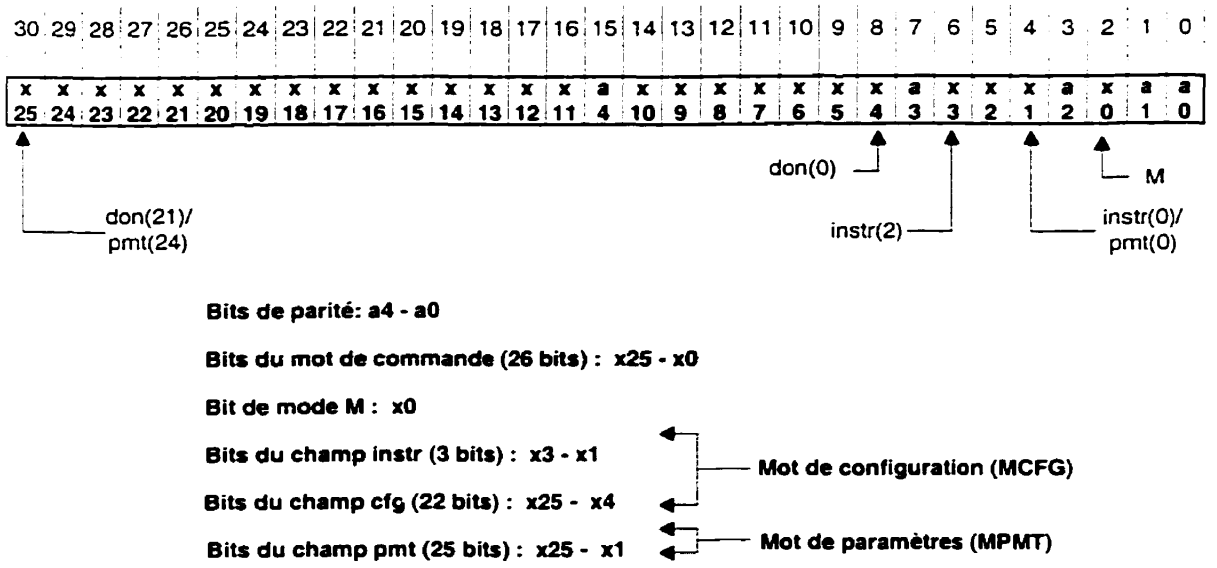


Figure 4.2 : Disposition des bits dans une trame de données

4.3.1 Le mot de configuration

À part du bit de mode, un mot de configuration comporte en plus un champ instruction ***instr*** (3 bits) et un champ de données ***don*** (22 bits) comme illustré à la figure 4.3. Selon l'instruction, les données ***don*** seront utilisées comme code de démarrage, code d'arrêt, commande d'initialisation ou bien aiguillées dans un des quatre registres de configuration (RC1 à RC4). Ces registres comportent plusieurs champs qui déterminent la configuration générale du PIVC. La commande d'initialisation sera traitée un peu plus loin dans cette section. La figure 4.4 illustre la position des champs dans chacun des registres. De plus, le tableau 4.1 indique la fonction de chaque champ pour tous ces registres.

Quatre fanions (FA, FDP, FDI, FCS) définissent lesquels des paramètres sont spécifiques, lesquels sont communs et le mode d'accès aux sites de stimulation. Trois champs (NBA, NBDP et NBDI) déterminent le nombre de bits utilisés pour spécifier chacun des paramètres. Ce nombre peut varier de 1 à 8. Ici, le code « 000 » indique un mot de 1 bit et le code « 111 » indique un mot de 8 bits. Trois autres champs (AMPC, DPC et DIC) indiquent la valeur de chacun des paramètres communs.

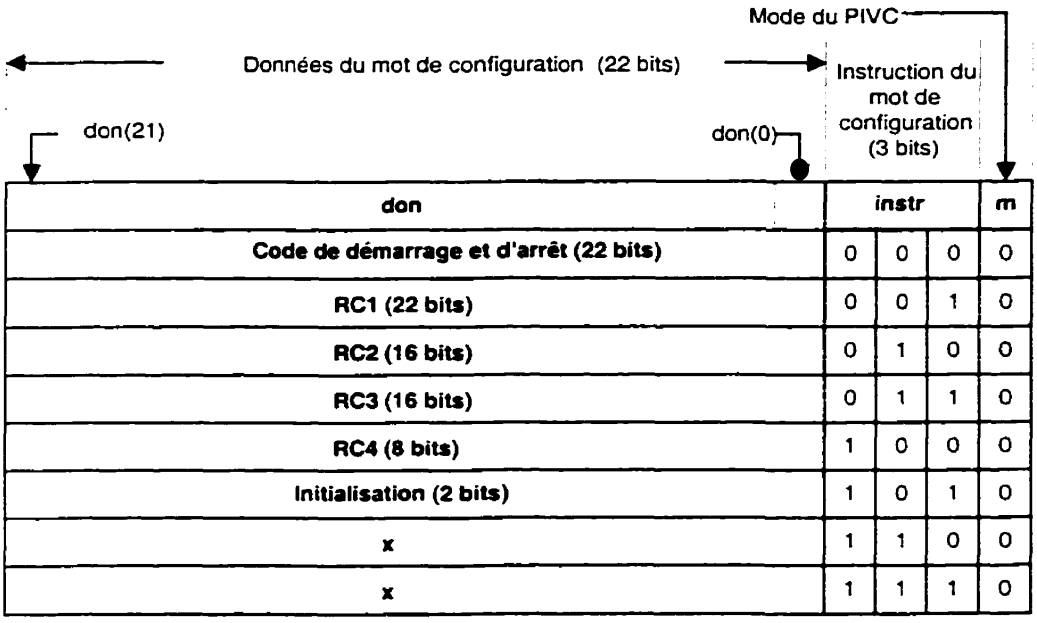


Figure 4.3 : Format des mots de configuration

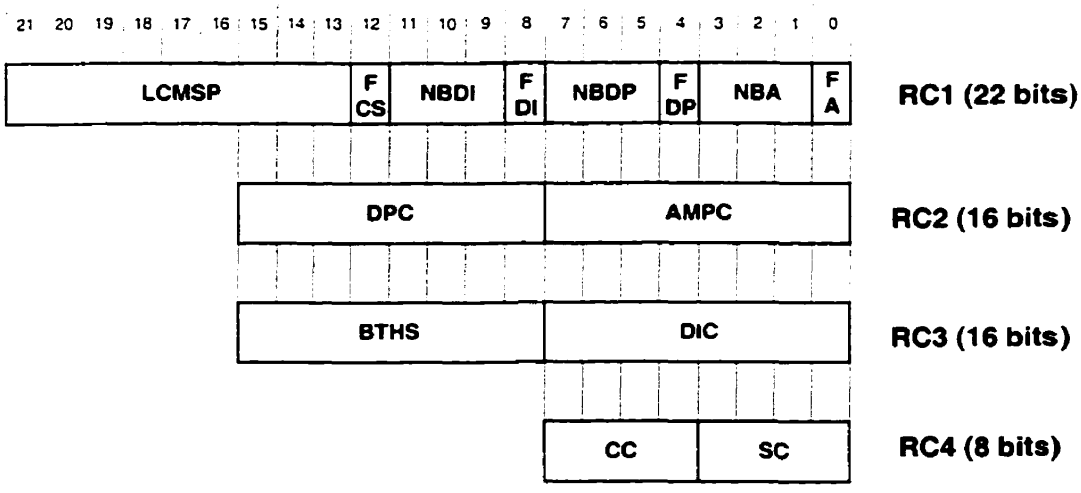


Figure 4.4 : Contenu des registres de configuration

Tableau 4.1 : Description des champs des registres de configuration

Champ	Registre	Nb. de bit	Description /fonction
FA	RC1	1	Fanion amplitude. Indique si le paramètre amplitude est commun (0) ou spécifique (1).
NBA	RC1	3	Nombre de bits utilisés pour spécifier l'amplitude (1 à 8).
FDP	RC1	1	Fanion durée de phase. Indique si le paramètre durée de phase est commun (0) ou spécifique (1).
NBDP	RC1	3	Nombre de bits utilisés pour spécifier la durée de phase (1 à 8).
FDI	RC1	1	Fanion durée d'interphase. Indique si le paramètre durée d'interphase est commun (0) ou spécifique (1).
NBDI	RC1	3	Nombre de bits utilisés pour spécifier la durée d'interphase (1 à 8).
FCS	RC1	1	Fanion d'adressage des canaux et sites. L'adressage est prédéterminée (0) ou spécifique (1).
LCMSP	RC1	9	Longueur d'une chaîne continue de mot de stimulation partiel.
AMPC	RC2	8	Valeur de l'amplitude commune.
DPC	RC2	8	Valeur de la durée de phase commune.
DIC	RC3	8	Valeur de la durée d'interphase commune.
BTHS	RC3	8	Base de temps pour l'horloge de stimulation
CC	RC4	4	Numéro du canal (0 à 15) à être enregistré dans la RAM
SC	RC4	4	Numéro du site (0 à 15) à être enregistré dans la RAM

Le champ BTHS est utilisé comme base de temps pour l'horloge de stimulation. Ce champ contient une valeur qui indique le nombre de cycles du système correspondant à un cycle de l'horloge de stimulation.

La séquence de balayage des sites de stimulation est établie à l'aide de l'envoi de 256 mots de configuration consécutifs (16 x 16 DEL) tous destinés au registre RC4. Le contenu de ce registre consiste en une adresse de stimulation qui est par la suite emmagasinée dans une RAM nommée ACS (adressage des canaux et des sites). Le champ CC du MCFG identifie le canal et le champ SC identifie le site. La séquence de balayage des sites de stimulation est donc stockée dans la RAM. La commande

d'initialisation permet de démarrer l'horloge de stimulation (don(0) = 1) et de remettre à zéro l'adresse de la RAM ACS (don(1) = 1).

4.3.2 Le mot de paramètres

À part le bit de mode, un mot de paramètre (MPMT) comporte en plus un champs ***pmt*** contenant un, une partie ou plusieurs mots de stimulation partiels (MSP) comme illustré à la figure 4.5. Chaque MSP contient un ou plusieurs paramètres de stimulation et potentiellement l'adresse du site à stimuler. La position des paramètres dans un MSP est toujours la même et commence par l'amplitude, suivie par la durée de phase, la durée d'interphase, le canal et finalement le site. Le nombre de bits pour chacun des paramètres est définis lors de la configuration. Naturellement, un paramètre définis comme étant commun sera absent du MSP. De même si le mode d'adressage des sites de stimulation est prédéfini, les numéros de site et de canal seront absent du MSP.

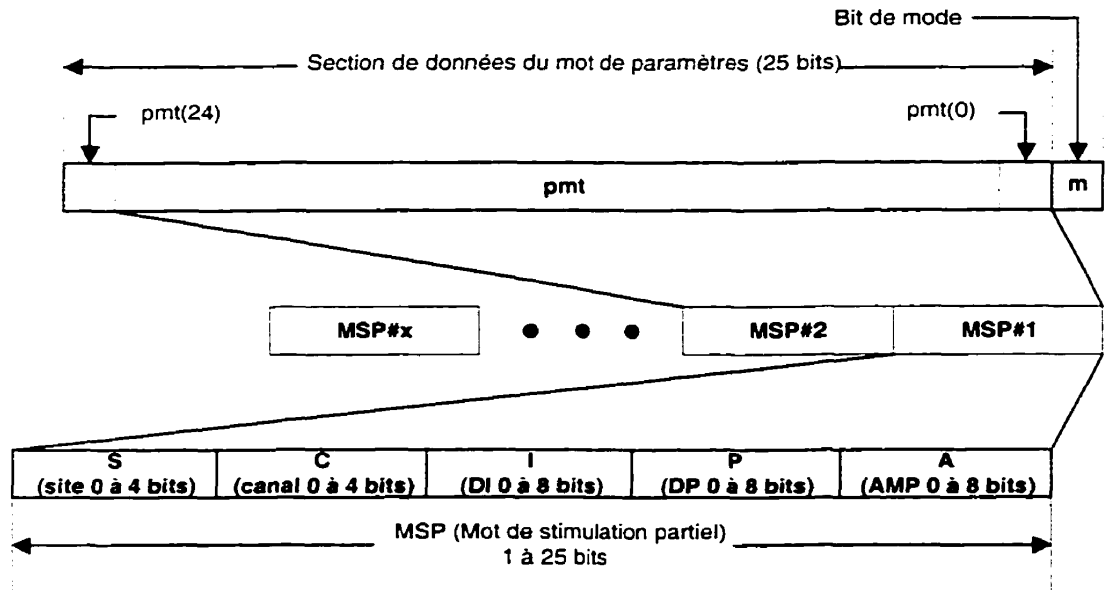


Figure 4.5 : Composition d'un mot de paramètres (MPMT)

Les MSP sont disposés de façon continue dans les MPMT pour permettre un taux de transfert efficace. Lorsque le PIVC est en mode stimulation, celui-ci reçoit alors des MPMT. Chaque MSP présent dans ces mots est alors joint aux paramètres communs et

possiblement à une adresse prédéterminée de site pour former une commande de stimulation (CSTIM). Cette commande est alors envoyé à l'un des 16 canaux de stimulation. Les MSP sont transmis jusqu'à ce que le nombre corresponde à la valeur présente dans le champ LCMSP du registre de configuration RC1. Suite à la réception d'une chaîne complète de MSP, le PIVC sera en mesure d'en accepter une autre immédiatement. La chaîne commencera alors par le premier bit du champ ***pmt*** du prochain MPMT.

4.4 Architecture générale du contrôleur du PIVC

Le schéma bloc du circuit numérique de contrôle du PIVC est illustré à la figure 4.6. Celui-ci comporte un contrôleur principal (CPIVC), un module de génération d'horloge de stimulation (basse fréquence), un contrôleur du CNA (CCNA) et 16 canaux de stimulation.

Le contrôleur principal coordonne la réception des données sérielles provenant du décodeur Manchester, la détection et correction d'une éventuelle erreur de transmission dans chacune des trames (décodeur de Hamming), la formation des mots de commande pour configurer le PIVC ou bien pour produire des commandes de stimulation qui sont alors transmises au bus des paramètres (comprenant les bus ***bsite, bcanal, bamp, bdp et bdi***).

Les données présentes sur le bus des paramètres sont acheminées au bon canal grâce au bus de canal ***bcanal***. Le signal ***bpv*** indique le moment où les données sont valides. Lorsqu'un canal reçoit une CSTIM, il sauvegarde les paramètres dans des registres temporaires. Une fois les opérations d'une potentielle stimulation courante terminées, le canal transfère les données de stimulation contenu dans ses registres temporaires à ses registres principaux et initie une nouvelle opération de stimulation.

Le module d'horloge de stimulation fournit une base de temps (basse fréquence) aux canaux. Ce module produit une série d'impulsions périodiques. Le délai entre chaque impulsion est un multiple de la période d'horloge. La valeur de ce multiple est stockée

dans le champ BTHS du registre de configuration RC3. Une impulsion sur la ligne ***sync*** redémarre l'horloge de stimulation.

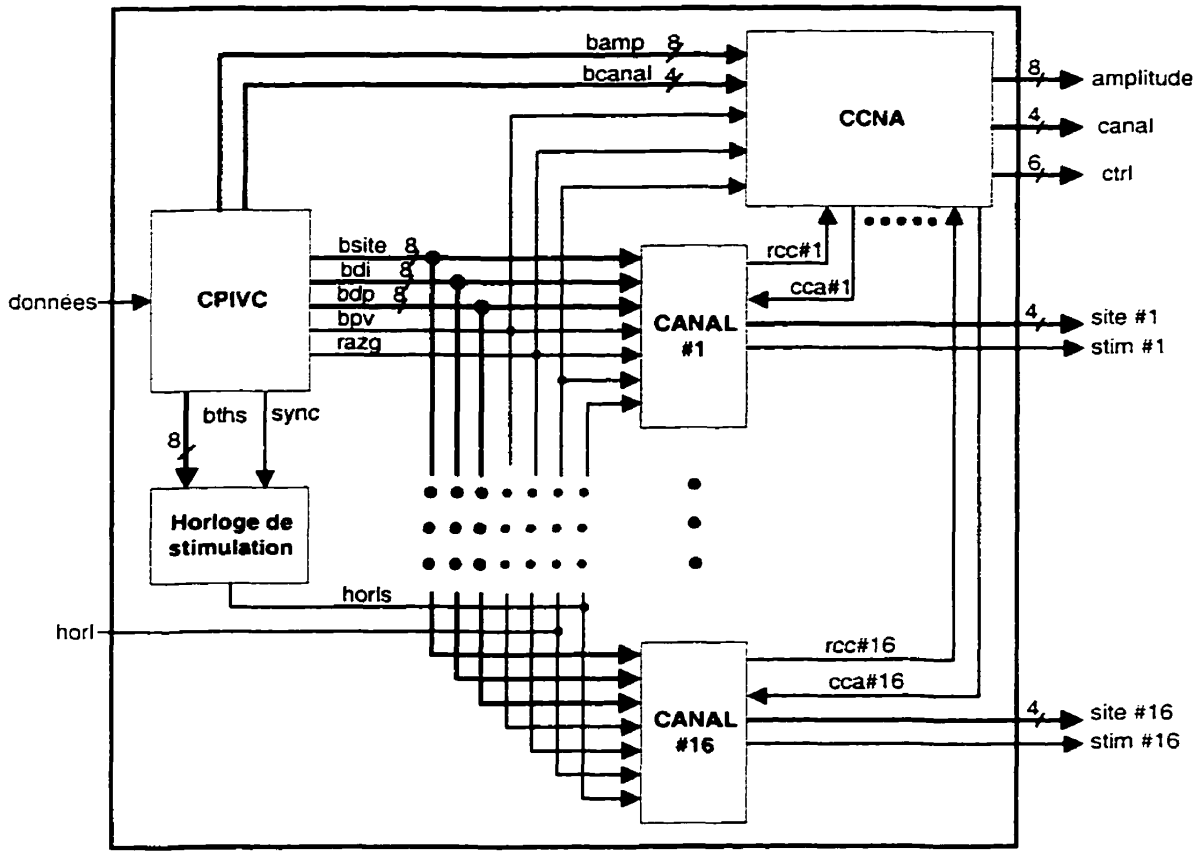


Figure 4.6 : Schéma bloc du circuit numérique de contrôle du PIVC

Le CCNA commande la conversion d'une amplitude (mot de 8 bits) en une tension au CNA à 16 canaux. Il reçoit les amplitudes du contrôleur principale et les emmagasine dans une RAM. Suite à une requête de conversion de la part d'un canal (signal ***rcc#x***), le CCNA initie la conversion en utilisant l'amplitude correspondant au canal. Le CCNA indique que la conversion est terminé à l'aide du signal ***cca#x***.

4.4.1 Le contrôleur principal (CPIVC)

Le schéma bloc simplifié du CPIVC est montré à la figure 4.7. Le module d'entrée est un convertisseur de données sérielles en données parallèles et détecteur de trame. Lorsqu'un bit de départ (0) est suivi d'une trame de 31 bits et finalement d'un bit d'arrêt, le module prépare les données sous format parallèle et les transmet au correcteur de Hamming en même temps que le signal ***trmdtc***. Dans le cas où le bit d'arrêt n'arrive pas, la trame est rejetée.

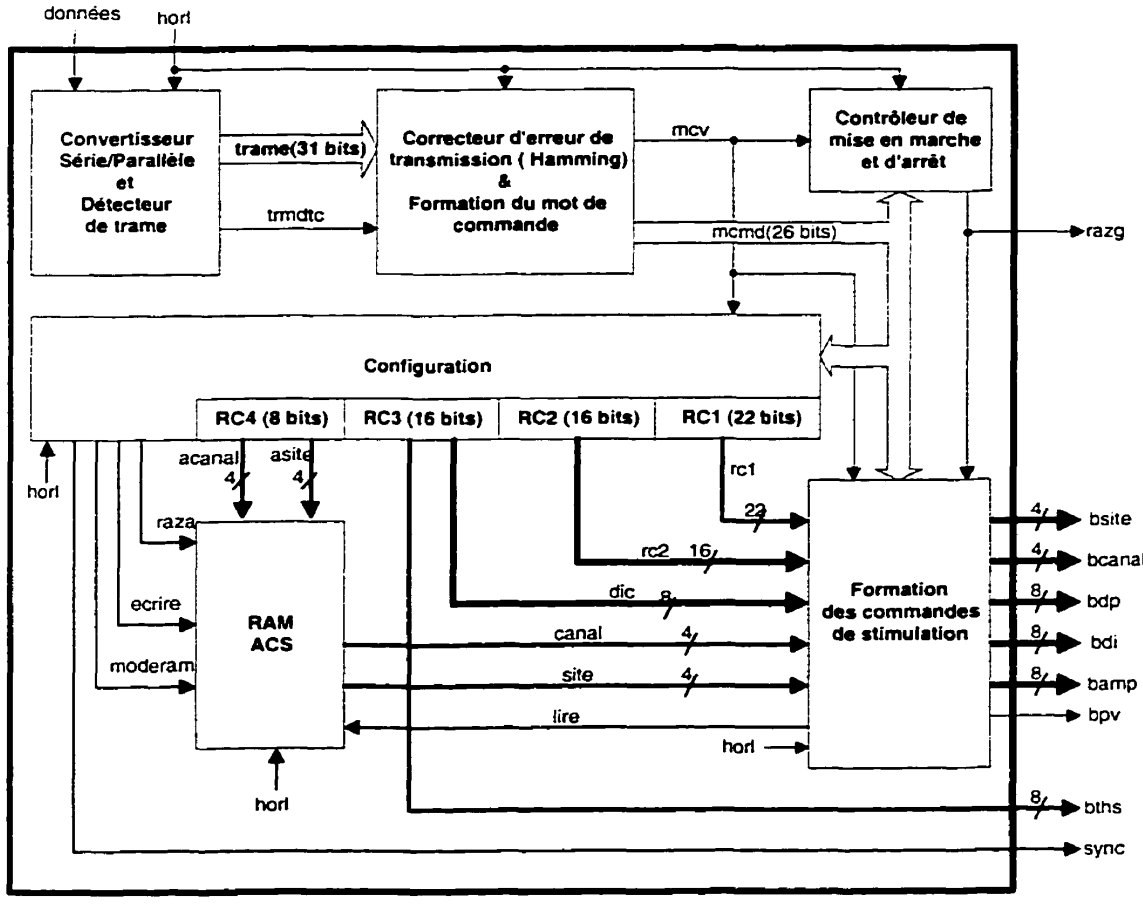


Figure 4.7 : Schéma bloc simplifié du CPIVC

Le correcteur de Hamming corrige une éventuelle erreur de transmission dans la trame et dispose les données dans le format original des mots de commande. Le signal ***mcv***

indique le moment où les données à la sortie du correcteur (le mot de commande de 26 bits) sont valides. Ces données sont acheminées à trois modules soit le contrôleur de mise en marche et d'arrêt, le module de configuration et finalement le module de formation de mot de stimulation complet.

Le module de mise en marche et d'arrêt contrôle l'état du circuit numérique (CPLD) entier à l'exception des deux modules d'entrées (détecteur de trame et correcteur de Hamming) via le signal de remise à zéro ***razg***. Le circuit numérique est mis en marche une fois le code de démarrage reçu. Il est également possible d'arrêter rapidement le fonctionnement du circuit en transmettant un code d'arrêt.

Une fois le système en marche, les mots de commandes sont acceptés soit par le module de configuration ou par le module de formation de mot de commande selon l'état du bit de mode (premier bit du MCMD).

Lorsque le bit de mode est à zéro, le MCMD est un mot de configuration. Selon les valeurs des bits du champ instruction, les données du champ ***don*** sont généralement sauvegardées dans un des quatre registres de configuration. Les données des registres RC1 à RC3 sont présentées au module de formation des mots de stimulation. Les données de RC4 sont automatiquement emmagasinées dans la RAM ACS.

La RAM ACS reçoit des commandes de deux modules. Lors de l'emmagasinage de la séquence de balayage, la RAM est contrôlée uniquement par le module de configuration. Celui-ci utilise le signal ***moderam*** pour mettre la RAM en état d'écriture. Une impulsion sur la ligne ***écrire*** indique à la RAM ACS que les données présentes sur les bus ***acanal*** et ***asite*** doivent être mémorisées. De plus, la même impulsion incrémente l'adresse active de la RAM. Cette adresse est mise à zéro à l'aide du signal ***raza*** suite à la réception d'un mot de configuration de type initialisation. Les 256 MCFG dédiés à RC4 doivent donc être transmis consécutivement pour l'enregistrement complet de la séquence de balayage. Le signal ***lire*** provenant du module de formation des mots de stimulation commande à la RAM ACS de fournir les données correspondant à l'adresse active. Cette adresse est incrémentée à chaque cycle d'horloge où le signal ***lire*** est actif.

4.4.2 Le module de formation des commandes de stimulation

La figure 4.8 montre le schéma du module de formation des commandes de stimulation. Lorsque le bit de mode d'un MCMD est à « 1 », ce module prend les données du champ ***pmt*** et forme une commande de stimulation présente sur les bus des paramètres (***bamp, bdp,bdi,bsite,bcanal***) et qui est envoyée à un canal.

Une commande de stimulation comprend trois paramètres soient l'amplitude (***bamp***) d'une impulsion de courant, sa durée de phase (***bdp***) et finalement sa durée d'interphase (***bdi***). Ce mot comprend en plus l'adresse (***bcanal*** et ***bsite***) du site de la stimulation donc l'endroit où la stimulation doit être appliquée.

Ce module possède cinq multiplexeurs qui effectuent la sélection des paramètres selon leur types définis soit commun ou spécifique et selon que l'adressage des sites de stimulation est prédéterminée ou spécifique. Les quatre fanions du registre RC1 sont employés pour indiquer la sélection à réaliser. Le bloc séquenceur aiguille des données du champ ***pmt*** de façon sérielle dans un ou plusieurs registres à décalage (AS, DPS, DIS, CS, SS). Ces registres servent à charger les valeurs des paramètres spécifiques contenus dans un MSP. Lorsque toutes les données d'un MSP sont chargées dans un ou plusieurs de ces registres, le séquenceur active le signal ***bpv*** pour indiquer que les données présentes sur le bus des paramètres sont valides. Le séquenceur utilise la ligne ***lire*** pour commander la lecture d'une nouvelle adresse de site et l'incrément de l'adresse active de la RAM ACS.

La valeur présente sur LCMSP permet au séquenceur de connaître le moment où une chaîne de mots de stimulation prend fin donc lorsque les données d'un MPMT ne sont plus valides.

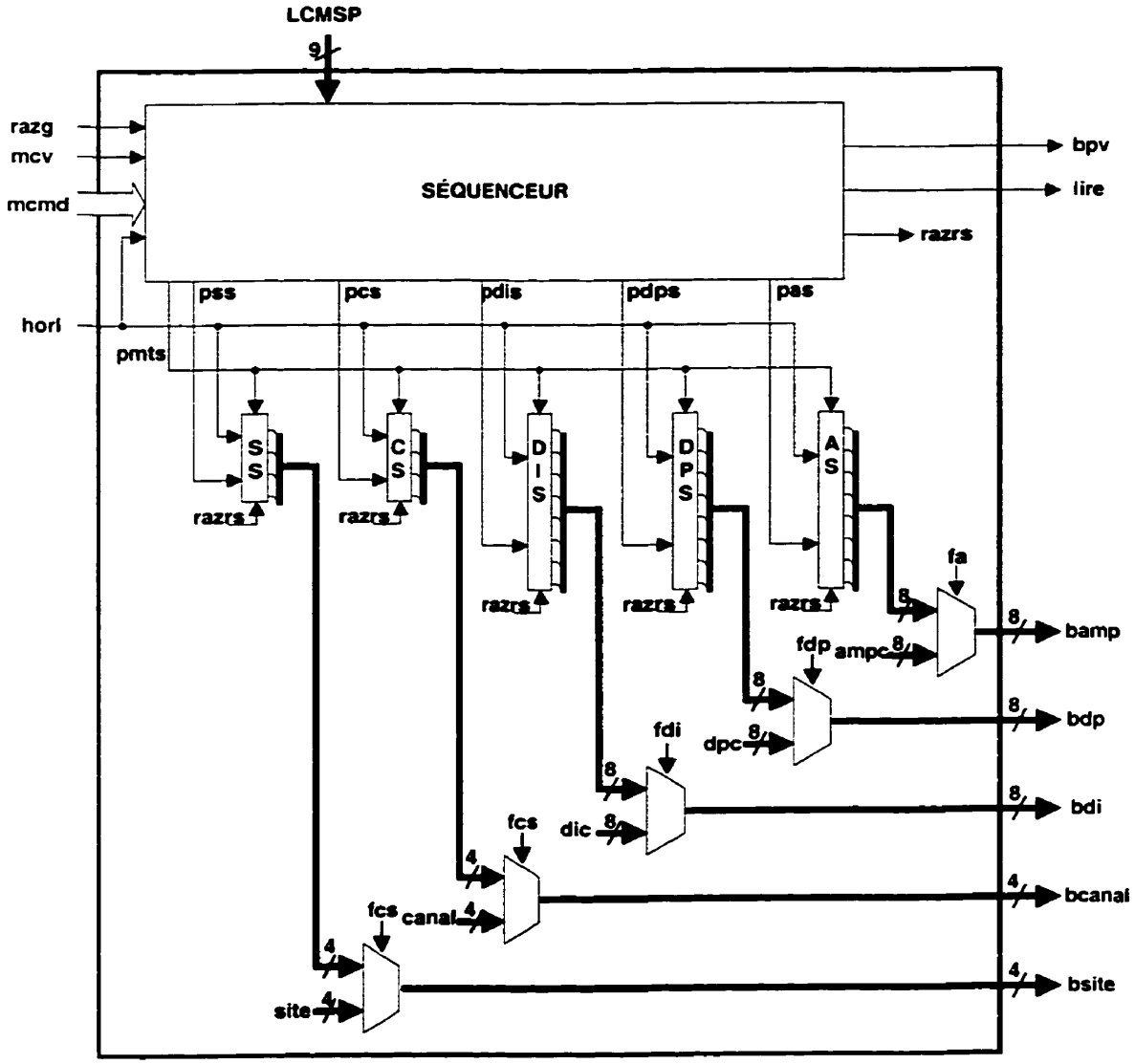


Figure 4.8: Schéma bloc du module de formation des commandes de stimulation

4.4.3 Les canaux de stimulation

Les canaux de stimulation sont des modules qui servent à synchroniser les durées de phase et d'interphase des stimulations (figure 4.9). Lorsqu'un canal reçoit une impulsion sur la ligne ***bpv*** et que la valeur présente sur le bus ***bcanal*** correspond au numéro du canal, les données présentes sur les bus ***bdi***, ***bdp*** et ***bsite*** sont alors mémorisées dans des registres temporaires (RIS,RPS et RSS).

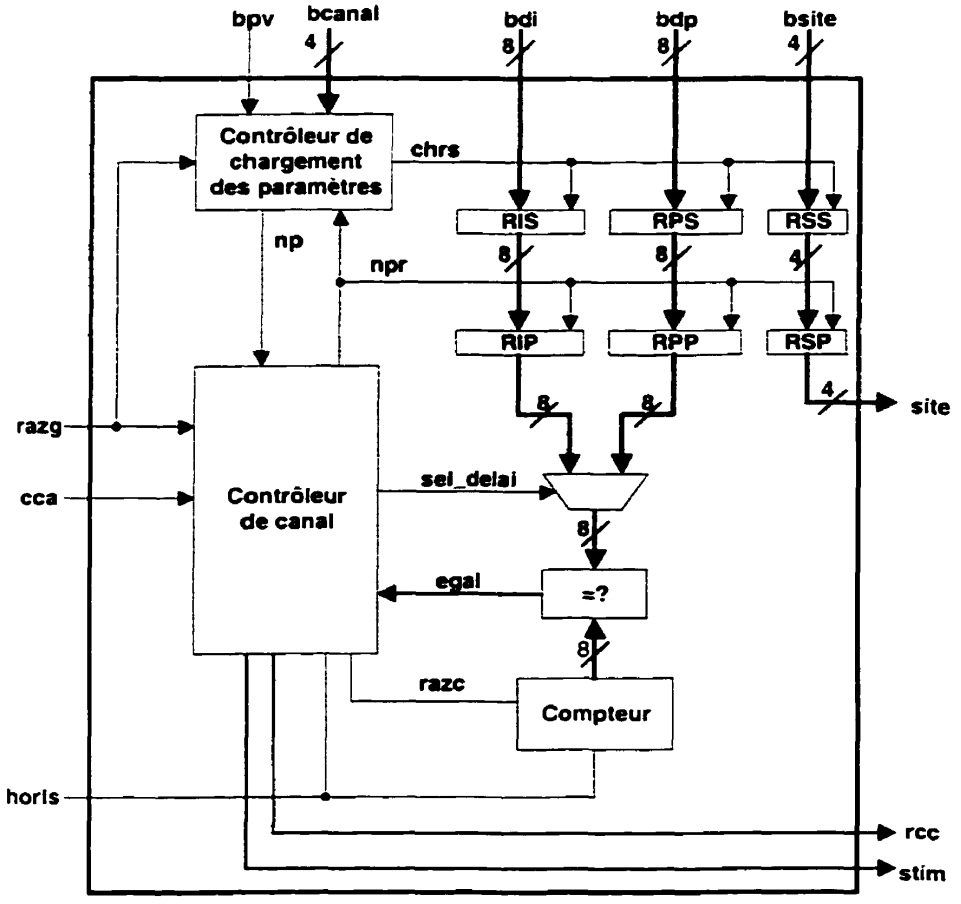


Figure 4.9 : Schéma bloc d'un canal de stimulation

Lorsqu'une actuelle stimulation prend fin, les données des registres temporaires sont alors transférées dans les registres courant (RIP, RPP et RSP). Le contrôleur de canal envoie alors une requête de chargement de l'amplitude au CCNA à l'aide du signal ***rcc***. Celui-ci répond par une impulsion sur la ligne ***cca*** , une fois la requête accomplie.

Le contrôleur de canal actionne alors les opérations de stimulation à l'aide de la ligne ***stim***. Cette ligne est connectée au multiplexeur analogique du canal et contrôle donc son état (ouvert ou fermé). Les délais (DP et DIP) sont calculés à l'aide notamment d'un compteur dont la base de temps vient de l'horloge de stimulation (signal ***horls***). La valeur du compteur est comparée à celle du délai en cours (DP ou DIP). Lorsque les deux valeurs sont égales, le délai est alors expiré et une action est requise.

L'implant cortical doit appliquer des stimulations biphasiques, c'est-à-dire que le courant dans la première phase voyage dans un certain sens et dans la phase suivante voyage dans le sens contraire. Il peut y avoir aussi un délai entre les phases (interphase). Dans le cas du prototype, les deux durées de phases sont respectées. Par contre le courant voyage toujours dans la même direction.

Une opération de stimulation (d'activation d'une DEL) comprend donc trois étapes. Une première phase de stimulation d'une certaine durée, un délai interphase et finalement une seconde phase de stimulation. Pendant les durées de phase, le signal ***stim*** est actif et le bus ***site*** est branché au port de sélection du multiplexeur analogique du canal.

4.5 Conclusion

Nous avons présenté dans ce chapitre la structure du circuit numérique du CPLD qui sert de contrôleur du PIVC. Ce module a pour fonction de recevoir des mots de commande d'un contrôleur externe et de commander les opérations de stimulation. L'architecture du circuit numérique définit deux modes de fonctionnement : configuration et stimulation.

Le mode configuration permet de définir les types de paramètres de stimulation qui sont communs à tous les sites, les types qui sont spécifiques ainsi que le nombre de bits nécessaires pour spécifier chacun des paramètres. Ce mode permet en plus de définir une séquence de balayage des sites de stimulation.

En mode stimulation, les données nécessaires pour initier une opération de stimulation ainsi que le format des mots de paramètres sont fonction de la configuration du PIVC. Ainsi, un seul mot de paramètre peut contenir un seul, une partie ou plusieurs mots de stimulation partiels. La réception d'un mot de stimulation partiel initie une opération de stimulation.

Le prochain chapitre traite des aspects relatifs à la fabrication et à la simulation du prototype et du CEB.

CHAPITRE 5

RÉALISATION ET RÉSULTATS

5.1 Introduction

Nous avons proposé dans les deux chapitres précédents deux architectures permettant l'implantation de la fonctionnalité d'un implant visuel cortical. Seule la première fut fabriquée. La deuxième architecture n'a pu être réalisée physiquement car le circuit intégré prédiffusé soit le CNA avec échantillonneurs-bloqueurs (CEB), après réception, a révélé des défectuosités de fabrication. Ce chapitre a deux objectifs principaux. Le premier est de présenter quelques aspects concernant la réalisation ainsi que les résultats de simulation du CEB. Le deuxième objectif du chapitre est de présenter les principaux aspects par rapport à la réalisation du prototype (PIVC) ainsi que certains résultats de simulation qui démontrent sa fonctionnalité.

5.2 Réalisation du CEB

Le CEB fut réalisé (figure 5.1) à partir d'un procédé de fabrication bipolaire prédiffusé de la compagnie Gennum par le biais de la Société Canadienne de Microélectronique (SCM). Dans cette technologie, un réseau de transistors bipolaires et autres composants sont disposés de façon régulière sur une puce de silicium. Une seule couche de métal permet au concepteur de relier électriquement les divers composants afin de compléter la réalisation d'un circuit électronique. Ces réseaux, offerts en plusieurs grandeurs, regroupent deux types de module : type ***bloc*** (block tiles) et type ***rue*** (street tiles). Chaque type de module comporte un certain nombre de composants.

Puisque le CEB est un circuit d'une bonne complexité, le réseau avec la superficie maximale disponible fut choisi pour implanter la fonctionnalité du circuit. Les

dimensions du réseau sont de 200milx200mil aussi nommé réseau 4x4. Ce réseau comporte 16 modules de type ***bloc*** et 24 modules de type ***rue***.

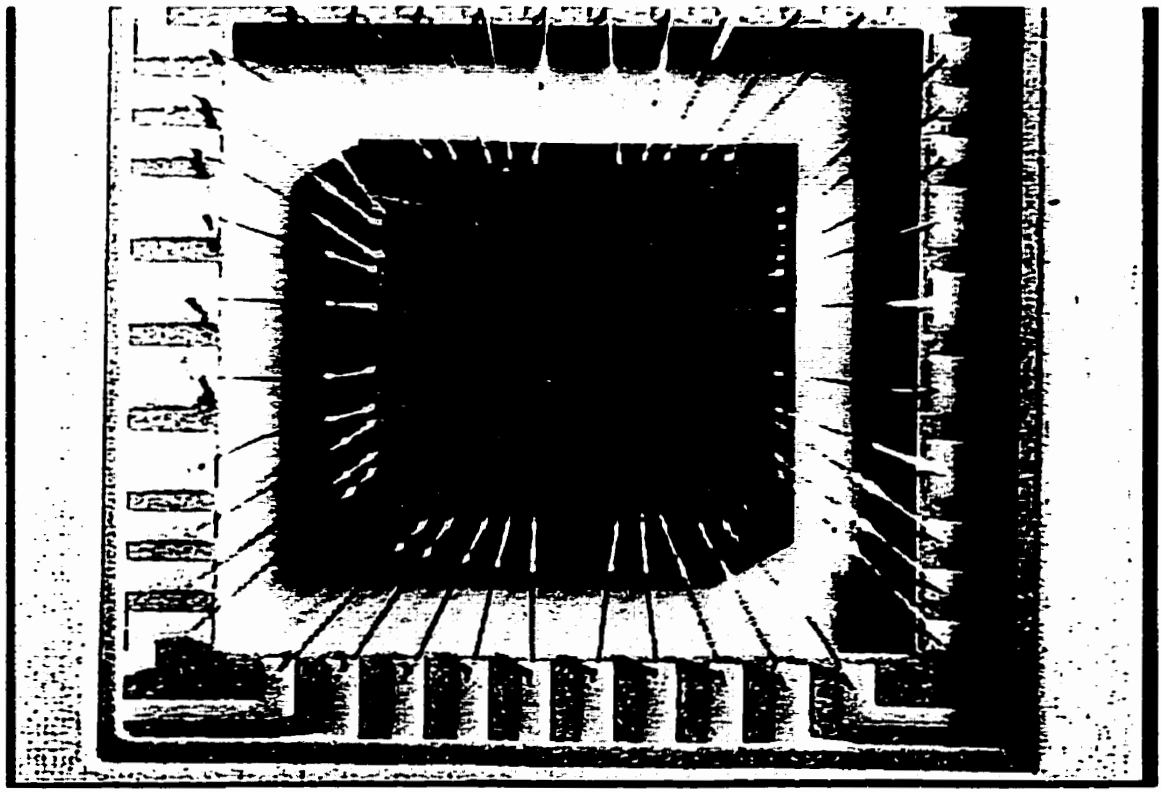


Figure 5.1 : Photographie du CEB

La plupart des composants possèdent un ou plusieurs points de croisement (cross-unders) permettant de croiser des signaux. Ils sont essentiels car, il n'y a qu'une seule couche de métal disponible pour relier électriquement les composants d'un circuit donné. Ces points possèdent cependant une certaine résistance et une certaine capacité qui peuvent selon le cas nuire au comportement du circuit. Le concepteur doit donc tenir compte de ces effets parasites dans le choix des endroits de croisement des signaux. Quelques éléments importants concernant l'utilisation de cette technologie :

- la tension maximale entre ***vcc*** et ***vee*** est de 20V;

- la tension Vbe seuil inverse avant claquage des transistors est d'environ 6V;
- la largeur minimale d'une ligne de métal est de 6um;
- pour éviter les effets d'électromigration, la densité de courant ne doit pas dépasser 5mA par largeur minimal (6um) de trace.

5.3 Simulation du CEB

Cette section présente les résultats des simulations du CEB et de ses principaux éléments constitutifs. Le but du CEB est de produire des courants de sortie stables à chaque canal de stimulation pour une période maximale d'environ 2ms. Ces courants sont utilisés pour activer des diodes électroluminescentes. La valeur maximale des courants de sortie du CEB est fixée à 10mA. Cette valeur est suffisante pour l'activation des DEL et évite dans une certaine mesure, l'encombrement des traces de métal dans le prédiffusé (deux traces de métal de largeur minimale doivent être juxtaposées pour toutes les sorties du CEB afin d'éviter les effets d'électromigration). Le CEB est donc optimisé pour fournir un courant de 10mA maximum et les simulations sont effectuées autour de ce point d'opération.

Les prochaines sections présentent les caractéristiques des principaux modules du CEB en commençant par l'amplificateur opérationnel, suivit du CNA (avec et sans le miroir de courant de type P) et finalement l'échantillonneur-bloqueur (EB) en mode courant. La dernière section présente les caractéristiques globales du CEB.

5.3.1 L'amplificateur opérationnel

L'amplificateur opérationnel (annexe B, figure B.7) possède deux étages d'amplification (à gain élevé) et un étage de sortie. La paire différentielle d'entrée est formée à partir de transistors en configuration Darlington ce qui donne une bonne impédance d'entrée et minimise le courant de polarisation d'entrée. Parmi ses caractéristiques listées dans le tableau 5.1, on retrouve entre autres une marge de phase de 45° qui assure la stabilité de l'amplificateur.

Tableau 5.1 : Caractéristiques simulées de l'amplificateur opérationnel

Caractéristiques	Valeur
Gain DC	86 dB
Bande passante (gain unitaire)	15.4MHz
Marge de phase	45°
Marge de gain	17 dB
Tension de faux zéro d'entrée	30uV
Courant de polarisation des entrées	6 nA
Impédance d'entrée différentielle (Rin)	2.6 MΩ
Impédance de sortie (Rout)	120Ω
Slew rate	31V/us
Tension d'alimentation Valim	12V
Tension d'entrée minimale Vmin (Valim = 12V)	3.1V
Tension d'entrée maximale Vmax (Valim = 12V)	10.8V
Consommation de courant (Valim = 12V)	1.234 mA
Consommation de puissance (Valim = 12V)	14.8 mW
CMRR (taux de rejet en mode commu)	110 dB
PSRR (taux de rejet des alimentations)	89dB

Les figure C.1 à C.3 de l'annexe C montrent les courbes de la réponse en fréquence, du gain en mode commun et du gain de l'alimentation. Les valeurs des autres paramètres de l'amplificateur sont adéquates compte tenu de sa tâche qui est de stabiliser le courant de référence du CNA à l'aide d'une boucle de contre-réaction. L'amplificateur est utilisé dans une topologie un peu inhabituelle du fait que son alimentation négative est branchée à la masse. Ceci implique une tension minimale d'entrée par rapport à la masse pour pouvoir utiliser correctement ce circuit.

5.3.2 Le CNA et le miroir de courant

Le CNA (annexe B, figure B.6) et le miroir de courant de type P (annexe B, figure B.5) ont pour tâche de produire le courant de référence utilisé par les EB. Le tableau 5.2 présente les principales caractéristiques du CNA avec et sans le miroir de courant.

Tableau 5.2 : Caractéristiques simulées du CNA avec et sans le miroir de courant

Caractéristiques	Valeur (CNA seul)	Valeur (CNA et miroir)
LSB (opération à 10 mA)	9uA	39uA
Valeur du courant maximal (opération à 10mA)	2.298 mA	10mA
Vref pour opération à 10 mA	5.767V	5.767V
Erreur de décalage de tension (offset)	0.004 LSB	0.121 LSB
Erreur de gain	0.020 LSB	0.878 LSB
Non-linéarité intégral max. (INL)	1.189 LSB	1.869 LSB
Non-linéarité différentielle max. (DNL)	0.254 LSB	0.305 LSB
Temps de stabilisation (settling time)	500 ns	2.75 us

Le courant de référence doit idéalement être proportionnel au mot binaire correspondant aux entrées b0 (LSB) à b7 (MSB). Chacune des entrées envoie une tension (basse inférieure à 0.6V) pour indiquer le niveau logique « 1 » et une tension haute (supérieure à 0.7 V) pour indiquer le niveau logique « 0 ». La conversion effectuée par la combinaison de ces deux modules doit être la plus précise possible. Une fois Vref ajusté, le CNA produit un courant d'une valeur maximale de 2.3 mA.

Le miroir a deux fonctions ; la première est de transformer un courant « tiré » par le CNA en un courant « poussé » vers les EB et la deuxième consiste à amplifier ce courant par un facteur d'environ 4.25 pour avoir un courant de sortie maximum de 10 mA. Les figures C.4 à C.9 (annexe C) montrent les courbes de certaines caractéristiques simulées du CNA avec et sans miroir de courant.

5.3.3 L'échantillonneur-bloqueur (EB)

Les figures B.2 à B.4 de l'annexe B présentent le circuit de l'échantillonneur-bloqueur. La figure 5.2 montre le comportement (simulé) de l'entrée/sortie en courant du CEB lorsque le courant de référence est 10 mA. La tâche de l'EB est premièrement d'échantillonner un courant provenant du CNA via le miroir de courant. Dans un second temps, l'EB doit reproduire le courant échantillonné, donc agir comme une source de courant pour l'une des 12 sorties du CEB à laquelle il est branchée. Sur la figure 5.2, la période d'échantillonnage s'étend de 0 à 1ms. Ensuite l'EB change d'état et maintient le courant échantillonné (la période de maintient s'étend de 1ms à 4ms sur la figure).

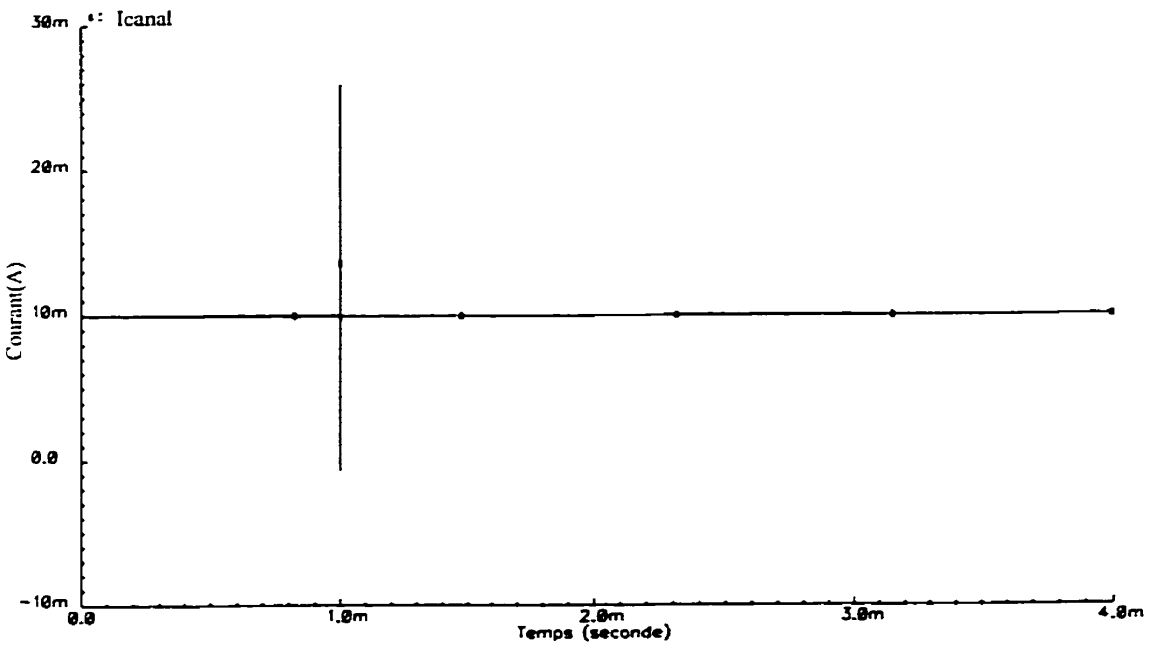


Figure 5.2 : Opération (échantillonnage/maintient) d'un EB

Le commutateur débranche la sortie du miroir de courant (référence) de l'entrée/sortie de l'EB qui est alors aiguillé vers une des 12 sorties du CEB. La figure 5.3 est un agrandissement de la figure 5.2 où on peut voir clairement un effet indésirable lors du passage de l'état d'échantillonnage à l'état de maintient. En effet le courant subit une petite chute d'intensité lors du changement d'état. Cette chute provient principalement du manque de rapidité du commutateur à l'intérieur de l'EB.

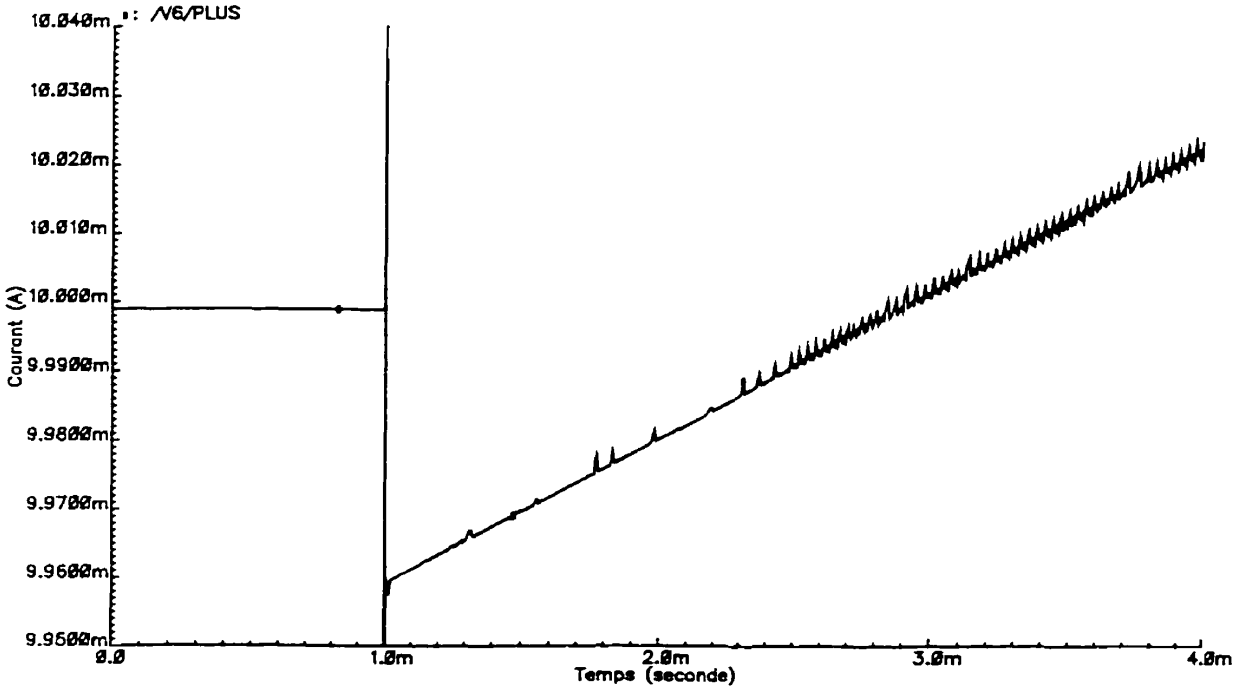


Figure 5.3 : Dérive du courant d'un EB en mode maintien

Un autre effet indésirable provient de l'EB et concerne la valeur de l'intensité du courant. Celle-ci ne reste pas constante mais grimpe à un certain taux. Cette dérive du courant est due au fait que l'impédance d'entrée du convertisseur tension-courant n'est pas assez élevée, et un courant « de fuite » vient charger le condensateur de maintient ce qui entraîne l'augmentation graduelle du courant à la sortie du CEB. En ce qui concerne la présente application (PIVC), ce phénomène n'est pas trop néfaste car le taux d'augmentation est d'environ de 40 uA/2ms donc équivalent à un LSB. Le tableau 5.3 résume les principales caractéristiques d'un EB.

Tableau 5.3 : Caractéristiques simulées d'un EB

Caractéristiques	Valeur
Chute d'intensité maximale du courant de sortie	40uA
Taux d'augmentation du courant de sortie	40uA/2ms
Temps de stabilisation (settling time)	5us
Tension à l'entrée/sortie du EB en mode échantillonnage et lorsque le courant = 10 mA.	4.76V
Tension à l'entrée/sortie du EB en mode échantillonnage et lorsque le courant = 40uA (1LSB).	1.959V
Tension minimale de la sortie en mode maintient	5V
Tension maximale de la sortie en mode maintient	6V
Impédance de sortie (mode maintient)	2 MΩ

5.3.4 Caractéristiques globales du CEB

La figure 5.4 montre les courbes idéales et simulées de la réponse du CEB pour chaque niveau du courant de sortie de l'EB une fois qu'il est en mode de maintient. Les figures C.10 et C.11 (annexe C) donnent les courbes des non-linéarités intégrale et différentielle du système complet.

La courbe simulée correspond amplement à la courbe théorique. Le tableau 5.4 donne les caractéristiques de l'ensemble du CEB lorsque tous les modules sont regroupés. Une certaine différence existe entre les performances idéales et simulées du système. Ces différences sont dues aux imperfections de chacune des parties qui s'accumulent. On remarque qu'une bonne part de l'erreur est due aux opérations de l'EB. Toutefois, les résultats simulés donnent la précision requise pour la présente application.

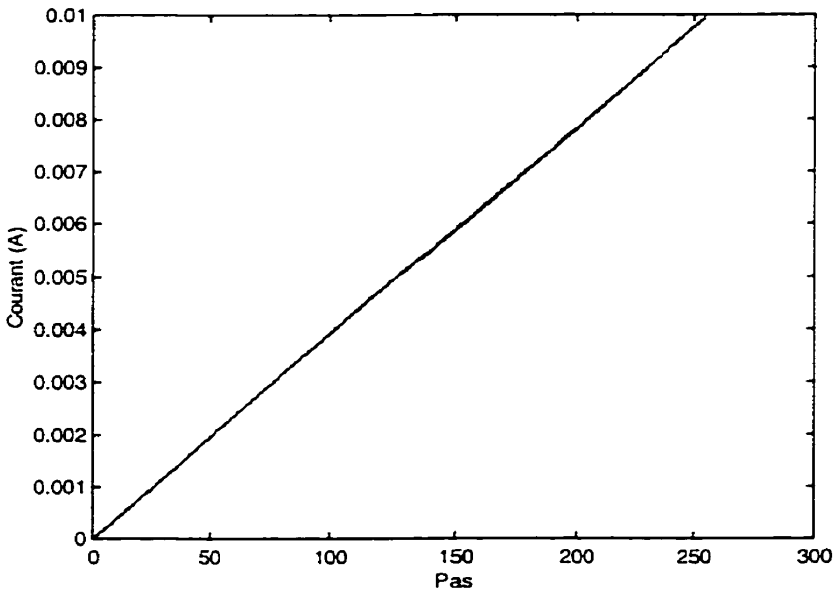


Figure 5.4 : Réponses simulée et idéale du CEB (mode maintient)

Tableau 5.4 : Caractéristiques simulées du CEB

Caractéristiques	Valeur
LSB (opération à 10 mA)	39 uA
Erreur de décalage (« offset »)	0.263 LSB
Erreur de gain	0.818 LSB
Non-linéarité intégral max. (INL)	2.672 LSB
Non-linéarité différentielle max. (DNL)	0.940 LSB
Temps de stabilisation (settling time) – échantillonnage	500 ns
Temps de stabilisation – maintient	5 us
Impédance de sortie (Rout)	2 MΩ
Chute d'intensité maximale du courant de sortie	40uA
Taux d'augmentation du courant de sortie	40uA/2ms
Tension minimale de la sortie en mode maintient	5V
Tension maximale de la sortie en mode maintient	6V

5.3.5 Précision des simulations et les causes de défectuosité

L'édition du schéma électrique et du dessin des masques fut réalisée à l'aide du logiciel Cadence et le logiciel Hspice fut utilisé pour simuler le circuit. Les résultats de simulation de tous les canaux (EB) du système sont similaires mais comportent quelques différences. Ces différences sont inévitables car le processus du dessin des masques avec une seule couche de métal s'est avéré fort complexe. Ainsi, selon les composants disponibles, certain canaux (EB) ont eu des impédances d'entrées plus élevées que d'autres par exemple.

Le processus d'extraction par Cadence dans cette technologie, c'est-à-dire la conversion d'un dessin de circuit en un « netlist » qui peut être lu par un simulateur, comporte quelques limites. En effet, les capacités parasites induites par les traces de métaux ne sont pas incluses dans le « netlist ». De plus, les résistances des points de croisement, dont chacune peut atteindre jusqu'à environ 44 ohms, sont également omis dans la génération du « netlist ». Ces omissions peuvent changer de façon importante le comportement du circuit mais ils ne constituent pas un obstacle pour empêcher le circuit une fois fabriqué de fonctionner convenablement.

En tout, 5 circuits intégrés du CEB furent fabriqués. Selon les tests effectués sur les CI, il existe plusieurs courts-circuits. Ainsi sur certaines entrées ***ech*** du CEB, on retrouve une tension non nulle ce qui est théoriquement impossible sans courts-circuits. Bien que le dessin du masque de métal respecte toutes les règles, le circuit reste très dense (annexe B, figure B.11) et il est bien possible qu'en réalité certaines traces de métal se touchent formant ainsi quelques courts-circuits. Ces courts-circuits empêchent le bon fonctionnement du CEB.

5.4 Réalisation du PIVC

Le prototype comprend deux cartes (circuits imprimés), la deuxième étant superposée à la première. L'architecture du PIVC fabriqué correspond à la première architecture développée au chapitre 3 (figure 3.2). La figure 5.5 montre l'ensemble des principaux

éléments de chaque carte. La première carte, la carte maîtresse, comprend entre autre le CPLD et le CNA. La deuxième carte, la carte affichage, comprend les convertisseur tension-courant, les démultiplexeurs et la matrice de diodes électroluminescentes.

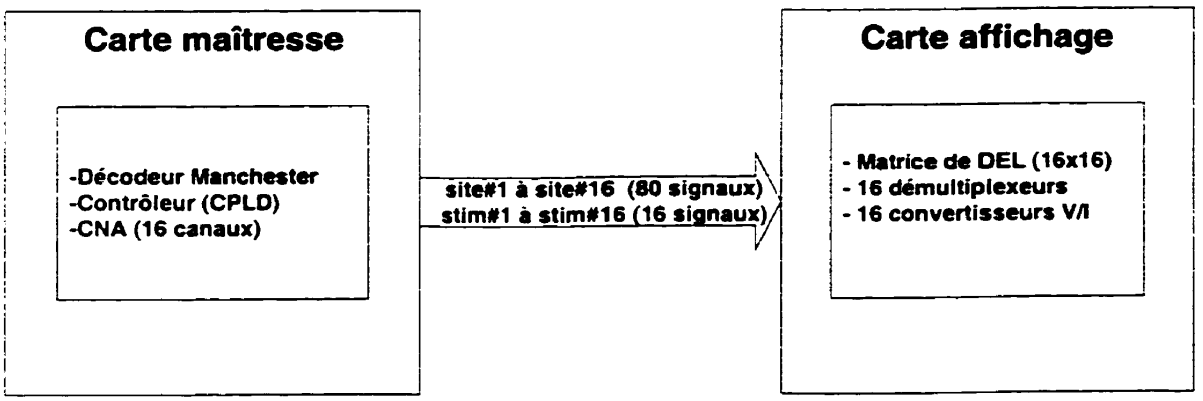


Figure 5.5 : Principaux éléments de chaque carte du prototype

Les raisons pour lesquelles deux cartes furent utilisées plutôt qu'une seule sont les suivantes :

- les dimensions d'une seule carte permettant de disposer tous les composants sont trop grandes;
- le routage des fils est grandement simplifié grâce à l'utilisation de connecteurs donc le nombre de couches nécessaires pour chaque carte est réduit;
- la superficie totale des deux cartes est moindre qu'une seule;
- la première carte pourra être utilisée pour d'autres applications.

La figure 5.6 montre une photographie du circuit imprimé de la carte maîtresse et la figure 5.7 une photographie du circuit imprimé de la carte affichage.

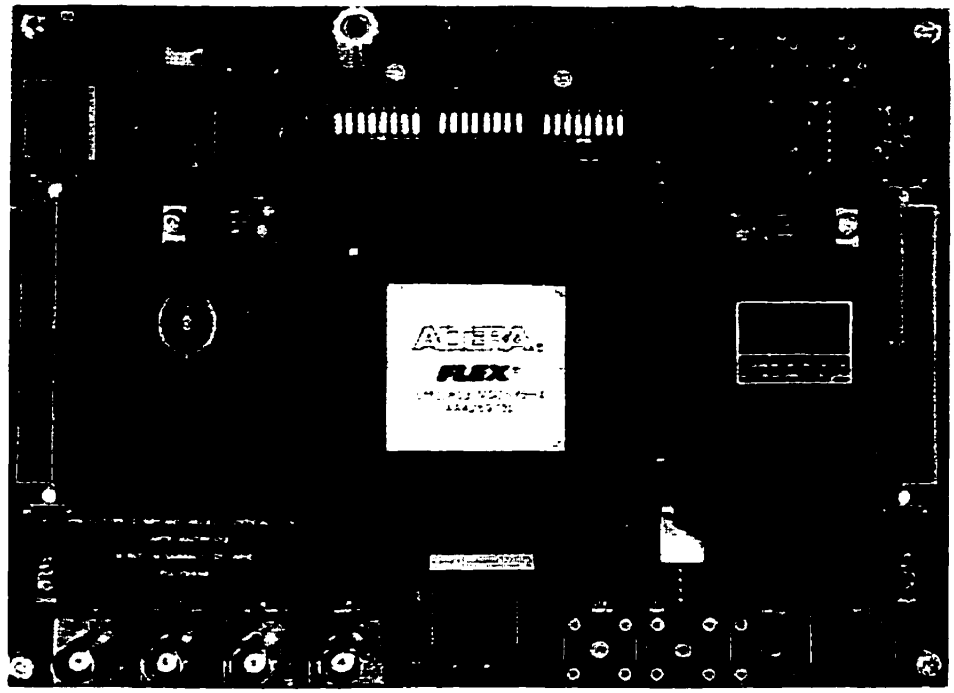


Figure 5.6 : Photographie du circuit imprimé de la carte maîtresse

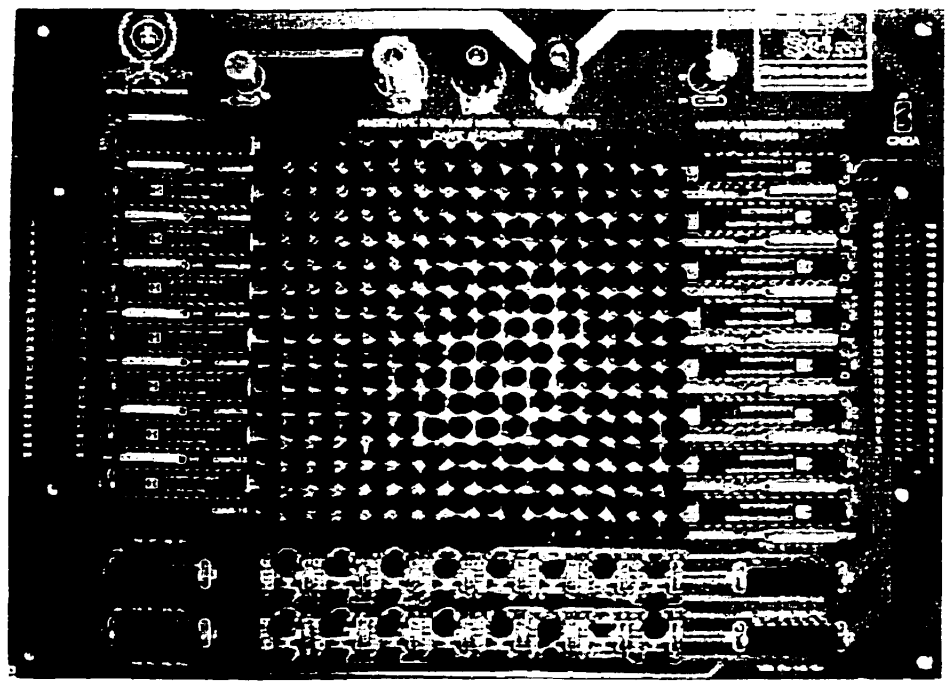


Figure 5.7 : Photographie du circuit imprimé de la carte affichage

L'édition des schémas électroniques (annexe A) et des circuits imprimés (annexe E) fut réalisée à partir du logiciel Accel. Le contrôleur du PIVC est implanté dans un CPLD de type EPF10K130VGC599 de la compagnie Altera. Le contrôleur utilise 2809 éléments logiques (logic elements) soit 42% du nombre total d'éléments logiques disponible dans un CPLD de ce type [82]. La prochaine section montre des résultats de simulation du contrôleur du prototype (PIVC) qui aident à comprendre son fonctionnement interne.

5.5 Résultats de simulation du contrôleur du PIVC

Le logiciel Maxplus II (Altera) fut employé pour la conception, la compilation et la simulation du circuit numérique du CPLD et un programme en C++ fût écrit pour générer les vecteurs de test utilisés pour la simulation. Ici on présente un exemple de simulation du contrôleur. La figure 5.8 montre le diagramme de temps complet de cette simulation.

Les entrées ***HORL*** et ***NNEES_DIRECT*** du diagramme correspondent au signaux ***horl*** et ***données*** de la figure 3.2 (chapitre 3). Tous les autres entrées (***TX_MODE***, ***HORL_DIRECT***, ***SIGNAL_MANCH*** et ***RAZ_MANCH***) montrées sur le diagramme servent à une vérification approfondie du design.

Dans cet exemple, nous avons fixé arbitrairement la longueur des mots de stimulation partiels (MSP) à 15 bits. Deux paramètres sont définis spécifiques soit l'amplitude qui est spécifiée à l'aide de 8 bits et la durée de phase spécifiée à l'aide de 7 bits. La durée d'interphase est fixée à 1 pour tous les sites donc définis comme paramètre commun. Le mode d'adressage choisi est le mode prédéterminé. De plus, le nombre de MSP à transférer de façon continue au PIVC est fixé à 2 (LCMSP = 2). Finalement on définit la base de temps (BTHS) à 3, c'est-à-dire que l'horloge de stimulation est trois fois moins rapide que l'horloge du contrôleur.La première étape consiste à configurer le PIVC en lui envoyant des mots de configuration (MCFG). Le signal ***MCV*** sur le diagramme de la figure 5.8 indique qu'un nouveau mot de commande est reçu.

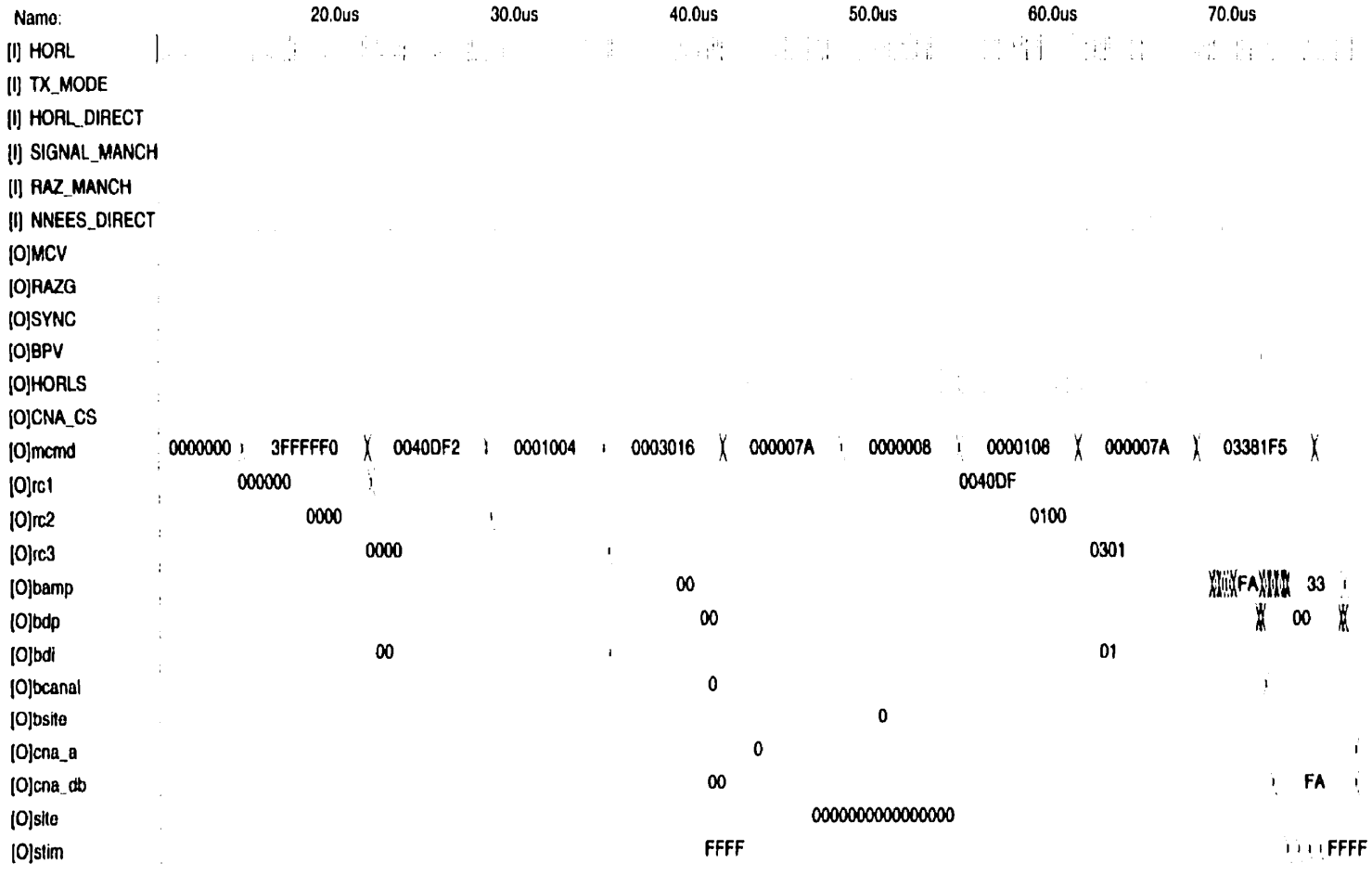


Figure 5.8: Simulation du contrôleur (intervalle de temps: 0 - 76us)

Ce mot correspond au signal ***mcmd*** du diagramme. Ainsi, le premier MCMD (3FFFFF0) contient le code de démarrage du contrôleur et amène le signal ***RAZG*** à « 1 ». Les trois mots suivants sont employés pour charger les registres de configuration (***rc1, rc2, rc3*** sur le diagramme) avec la configuration établie. À ce moment, on remarque que l'horloge de stimulation ***HORLS*** démarre (sa période étant trois fois plus grande que celle de l'horloge du système).

Une commande d'initialisation (***mcmd*** = 000007A) amène l'adresse de la RAM ACS à zéro. On charge par la suite les deux premières adresses de stimulation (1ere adresse : canal=0 et site=0, 2ième adresse : canal=1 et site=0). Les opérations de configuration se terminent avec l'envoi d'une commande d'initialisation dans le but de ramener l'adresse de la RAM ACS à zéro.

Les opérations de stimulation peuvent maintenant être initiées. La figure 5.9 est un agrandissement de la zone d'intérêt. Ici on se limite à l'envoie de 2 MSP seulement. Les valeurs de l'amplitude et de la durée de phase inscrites dans le premier MSP sont « FA » et « 01 » respectivement.

Les valeurs de l'amplitude et de la durée de phase inscrites dans le deuxième MSP sont « 33 » et « 01 » respectivement. Le chargement du deuxième MSP commence suite à la réception du premier mot de paramètre et se termine suite à la réception et au traitement du second.

Le premier MPMT est détecté à t = 68 us. Après 15 cycles d'horloge, le module de formation des commandes de stimulation produit un mot comprenant tous les paramètres spécifiques et communs (***bamp,bdp,bdi***) ainsi que l'adresse (***bcanal,bsite***) de la stimulation. Le signal ***BPV*** indique aux canaux de stimulation qu'une nouvelle commande est disponible. Le canal qui correspond à ***bcanal*** (ici le canal 0) prend alors les données (paramètres et site) et les sauvegarde dans ses registres temporaires. Puisqu'à ce moment il n'y a aucune stimulation en cours, le canal est en mesure de transférer les données de ses registres temporaires à ses registres principaux.

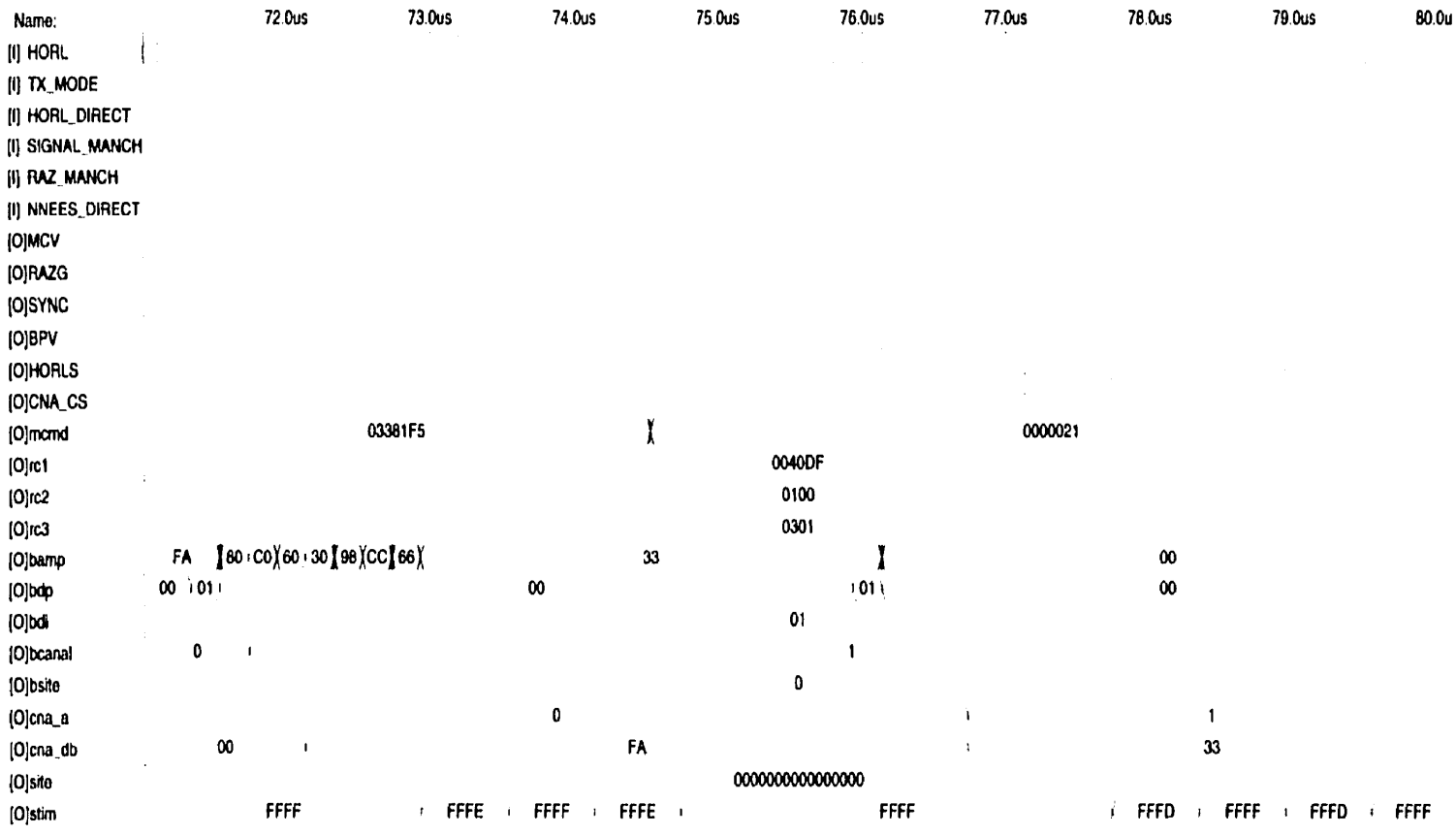


Figure 5.9 : Simulation du contrôleur (intervalle de temps : 71.2us - 80us)

L'amplitude (***cna_db***) est alors enregistrée dans le bon canal (***cna_a***) du CNA. Le signal ***cna_cs*** indique que les données sur les bus ***cna_db*** et ***cna_a*** sont valides.

Les opérations de stimulation peuvent alors être effectuées (bus ***site*** et ***stim***). Le bus ***site*** des diagrammes de temps regroupe les bus site#x des 16 canaux de la figure 3.2. Donc le bus ***site #1*** (figure 3.2) correspond au 4 premiers bit du bus ***site*** des diagrammes de temps, le bus ***site #2*** au 4 autres bits suivants et ainsi de suite. Le bus ***stim*** des diagrammes de temps regroupe tous les signaux ***stim#x*** (pour chaque canal) de la figure 3.2. L'annexe F présente d'autres résultats de simulation du contrôleur.

5.6 Vérification de la fonctionnalité du prototype

Un banc de test fut développé dans le but de tester le prototype. Ce banc de test (ou testeur), réalisé à partir d'un CPLD (EPF10K20RC240-4) installé sur une carte de développement d'Altera, agit comme contrôleur externe et envoie des commandes au PIVC.

L'annexe G montre le fonctionnement du testeur. Le testeur permet entre autre de stimuler un seul et unique site ou bien tous les sites dans le but d'afficher des images. Il permet donc de vérifier le bon fonctionnement de tous les modules du PIVC et surtout de son contrôleur. Ainsi toutes les étapes nécessaires pour arriver à l'affichage d'une image soit la réception des mots de commande, la correction d'une erreur de transmission, le décodage des mots de configuration et des mots de paramètres et finalement les opérations de minutages des canaux de stimulation peuvent être vérifiées.

Grâce au testeur, tous les formats possibles de MSP peuvent être validés pour afficher les images sur la matrice de DEL du PIVC. Il est donc possible d'effectuer des opérations de stimulation grâce à l'envoi d'information avec un débit variable et surtout de changer les paramètres de stimulation très facilement.

5.7 Conclusion

Nous avons présenté dans ce chapitre les résultats de simulation du CEB et du contrôleur du PIVC ainsi que certains détails relatifs à leur réalisation. Dans le cas du CEB, les simulations ont permis de montrer les détails de son fonctionnement et ses performances théoriques.

L'architecture du prototype proposé peut donc être considérée comme base pour la conception d'un implant visuel cortical. Il permet en plus de donner une idée de ce que les personnes percevront avec un implant visuel cortical. Il faut toutefois souligner que des différences existent entre la perception donnée par la matrice de DEL et la perception des phosphènes. Ainsi, l'activation et la désactivation des DEL indiquent le moment précis où une stimulation a lieu. Nous savons par contre que dans le cas de l'implant, il y a un délai entre le début d'une stimulation et l'apparition d'un phosphène. Toutefois, malgré les différences entre les DEL et les phosphènes, le prototype permet de valider très rapidement la fonctionnalité de l'implant.

La caractéristique qui démarque ce circuit des autres concerne surtout la flexibilité et la programmabilité de l'implant. Avec cette topologie, il est possible de réduire le taux de transfert d'information considérablement. Ainsi, il est possible que notre application (prothèse visuelle) nécessite l'adressage des sites de façon spécifique et qu'aucun des paramètres ne soit commun. Avec le présent système, un balayage complet des 256 sites (DEL) de la matrice se fait alors avec un minimum de 10814 cycles d'horloge (250 x 32 bits par MSP x 33/25) du système (PIVC).

Il est également possible que les recherches sur la stimulation du cortex visuel démontrent qu'un ou plusieurs paramètres soient communs à tous les sites et que l'adressage puisse être prédéterminé. Par exemple, en supposant qu'un seul bit est nécessaire pour spécifier l'amplitude du courant de stimulation et que tous les autres paramètres sont communs, il est possible d'effectuer un balayage complet des 256 sites à l'aide de seulement 338 cycles d'horloge (256 x 1 bit par MSP x 33/25).

Cette dernière configuration est environ 30 fois plus rapide que la première. Il serait étonnant qu'un seul bit soit suffisant pour spécifier l'amplitude du courant, mais dans tous les cas, la topologie proposée ici permet un transfert d'information très performant.

CONCLUSION

Parmi les trois avenues de recherche (stimulation de la rétine, du nerf optique ou du cortex visuel) qui permettraient de récupérer une certaine vision chez un non-voyant, le stimulateur cortical est la seule approche jusqu'à maintenant expérimentée chez les humains et les résultats obtenus sont considérés encourageants par plusieurs chercheurs. Il semble donc raisonnable d'anticiper la réalisation d'une prothèse visuelle corticale d'ici quelques années.

Le présent projet s'inscrit donc dans l'effort de recherche nécessaire pour parvenir à la réalisation d'un stimulateur visuel cortical. Dans le but de mieux cerner d'éventuels problèmes de conception, il est souvent judicieux de passer par une étape de prototypage du système. Le but de ce projet était donc de proposer une architecture de la partie implantable du stimulateur visuel cortical et ensuite de réaliser un prototype (PIVC) fabriqué à partir de composants discrets commerciaux qui réalise la fonctionnalité de l'implant. Une matrice de diodes électroluminescentes simule les phosphènes.

La première étape du projet fut d'étudier les stimulateurs en général. Cette étude a permis de faire ressortir la seule classe de stimulateur apte à être utilisée comme prothèse visuelle, c'est-à-dire celle qui est composée d'un contrôleur externe et d'un implant. Le contrôleur externe est utilisé pour capter et améliorer des images. L'implant injecte des courants de stimulation dans le cortex visuel du non-voyant selon les données reçues du contrôleur externe.

Une description des connaissances actuelles sur la stimulation du cortex visuel et des architectures de stimulateurs corticaux nous a permis, par la suite, de comprendre que l'implant cortical doit nécessairement posséder plusieurs canaux de stimulation de façon à ce que plusieurs sites puissent être activés simultanément.

Nous avons ensuite procédé à la mise au point de l'architecture générale de deux prototypes. La première, fabriquée, possède seulement des composants et circuits intégrés commerciaux. La deuxième possède en plus un circuit intégré prédiffusé (CEB) contenant une bonne partie des circuits analogiques du PIVC.

Le protocole de communication entre les deux modules du stimulateur et la structure du circuit numérique du contrôleur du PIVC furent alors décrits et expliqués. Finalement nous avons traité, dans le dernier chapitre, quelques aspects concernant la réalisation du CEB et du prototype, ainsi que de leurs résultats de simulation.

Les données relatives aux expériences effectuées sur la stimulation démontrent clairement qu'il est relativement facile de générer des phosphènes stables suite à l'injection d'un courant dans le cortex visuel. Cependant, certaines caractéristiques concernant l'apparence et le temps d'activation des phosphènes laissent présager quelques difficultés à produire des images à partir des phosphènes. En effet, plusieurs phénomènes comme l'accoutumance, ou bien le laps de temps requis avant l'apparition d'un phosphène suite à une stimulation, viennent compliquer un éventuel processus artificiel de récupération de la vision.

Pour ces raisons, le prototype proposé (donc éventuellement de l'implant) fut réalisé avec une grande flexibilité d'opération. Cette flexibilité est nécessaire car les connaissances actuelles sur la stimulation du cortex visuel sont limitées. L'implant doit donc avoir la capacité de fonctionner différemment selon les résultats obtenus.

La flexibilité du système permet en plus de modifier la quantité d'information à transférer du contrôleur externe à l'implant. Cette caractéristique sera avantageuse surtout lors des premières expériences qui démontreront les meilleures façons de stimuler.

Des deux versions du prototype proposées dans ce mémoire, seule la première fut fabriquée. La deuxième comporte cependant un module qui intègre une bonne partie des circuits analogiques à l'aide d'une technique jusqu'à maintenant peu utilisée dans ce

genre d'application. Cette technique consiste à employer un seul convertisseur numérique-analogique et à mémoriser les courants de stimulation grâce à des mémoires de courant. Utilisée dans un implant multicanaux, cette méthode comporte selon nous des caractéristiques avantageuses, comme par exemple une réduction dans les dimensions du circuit et dans la consommation d'énergie. Habituellement réalisée à partir d'un circuit CMOS, nous avons adapté cette technique à la technologie bipolaire, car celle-ci nous permettait de manipuler des courants importants, nécessaires à l'activation des diodes électroluminescentes. Les simulations ont démontré le bon fonctionnement du circuit. Cependant, l'utilisation d'un prédiffusé à une seule couche de métal pour implanter ce circuit a amené des complications supplémentaires au projet.

Une bonne partie de l'architecture du prototype est reconfigurable. Cette particularité permettra éventuellement de modifier le design actuel, afin de le rendre plus performant. Deux parties ne peuvent être changées dans le design : le nombre de canaux et le nombre de sites par canal. Quelques éléments devront nécessairement être ajoutés à cette topologie avant l'intégration finale de l'implant : par exemple, un système de test et de vérification de l'intégrité de l'implant et de l'interface bioélectronique. Pour ce faire, le protocole de communication entre les deux modules du stimulateur devra être revu car l'information voyagera alors dans les deux directions.

Il est bien possible que certaines parties du contrôleur du prototype devront être modifiées suite à l'acquisition de nouvelles connaissances dans le domaine de la stimulation du cortex visuel. Par exemple, une fois connues, les performances réelles du système RF de transmission de données et d'énergie peuvent imposer des changements au système de correction d'erreur de transmission. En effet, ce système est limité par le fait qu'une seule erreur par trame peut être détectée, ce qui pourrait être insuffisant.

Le prototype comporte deux cartes. La première, la carte maîtresse, contient principalement le CPLD et le convertisseur numérique-analogique à 16 canaux. La deuxième, la carte affichage, contient principalement la matrice de diodes électroluminescentes, les démultiplexeurs et les convertisseurs tension-courant. Les

connexions entre les deux cartes se font à l'aide de deux connecteurs sur chaque carte comportant 50 broches chacune. La première carte, comportant le CPLD reconfigurable, pourra éventuellement servir à d'autres applications.

Nous recommandons l'utilisation du prototype pour premièrement tester à fond les performances du PIVC et du contrôleur externe. La partie numérique (possiblement modifiée) décrite en VHDL sera alors prête pour son intégration dans une puce dédiée. D'autre étapes sont cependant nécessaires avant d'arriver à un implant cortical prêt à être testé in vivo. Parmi ces étapes, il y a la conception des différents modules analogiques parmi lesquels le lien RF, le convertisseur numérique analogique, le sélecteur de couples d'électrodes, ainsi que le système de mesure de l'état de l'interface électrode-tissu. Plusieurs de ces modules sont présentement en voie de développement par des membre de l'équipe PolySTIM. La dernière étape consistera à intégrer et tester tous les modules (numérique et analogique).

BIBLIOGRAPHIE

[1] NAEYAERT, K. (1990). La cécité et la déficience visuelle au Canada, 3, no. cat.82-615, Ottawa.

[2] PREVENT BLINDNESS AMERICA. (Page consulté le 1er mars 1999). Site de "Prevent Blindness America",[En ligne]. Adresse URL : http://www.prevent-blindness.org/

[3] WYATT, J.L. et RIZZO, J.F. (mai1996). Ocular implants for the blind. IEEE Spectrum, 33, 47-53.

[4] HUBEL, D.H. (1995). Eye, Brain, and Vision. Scientific American Library, New York, 240 pages.

[5] BRAILLE INSTITUTE. (Page consulté le 1er mars 1999). Site du "Braille institute",[En ligne]. Adresse URL : http//www.brailleinstitute.org/

[6] DOBELLE, W.H. et MLADEJOVSKY, M.G. (1974). Phosphene produced by electrical stimulation of human occipital cortex and their application to the development of a prosthesis for the blind. J. Physiol., 243, 553-576.

[7] WYATT, J.L. et RIZZO, J.F. (1997). Prospect for a visual Prosthesis. The Neuroscientist, no.4, 251-262.

[8] CENTER FOR SOLID-STATE ELECTRONICS AND OPTOELECTRONICS. (Page consulté le 1er février 1999). Site du "Retina Implant",[En ligne]. Adresse URL : http//www.uni-duisburg.de/reports/jb95/

[9] UNIVERSITÉ DE TUEBINGEN – PROF. E.ZRENNER. (Page consulté le 1er février 1999). Site du "Sub Retinal Implant Project",[En ligne]. Adresse URL : http//www.tuebingen.de/subret/

[10] UCL-DICE (Université catholique de Louvain). (Page consulté le 1er février 1999). Site du MIVIP (Microsystems Based Visual Prostheses),[En ligne].

Adresse URL : http//www.dice.ucl.ac.be/Mivip/

[11] BRINDLEY, G.S. et LEWIN, W. (1968). The sensations produced by electrical stimulation of the visual cortex. J. Physio, 196, 479-493.

[12] DOBELLE, W.H., MLADEJOVSKY, M.G. et GIRVIN, J.P. (1974). Artificial Vision for the blind : electrical stimulation of visual cortex offers hope for a functional prosthesis. Science, 183, 440-444.

[13] NORMANN, R.A. (1995). Visual Neuroprosthetics – Functional Vision for the Blind, IEEE Engineering in Medicine and Biology, 77-83.

[14] NORMANN, R.A., MAYNARD, E.D., GUILLORY,K.S et WARREN,D.J.I (1996). Cortical implants for the blind, IEEE Spectrum, 33, 54-59.

[15] FOERSTER, O. (1929). Beiträge zur pathophysiologie der sehbahn und der sehsphäre, J. Psychol. Neurol., 39, 463-485.

[16] DOBELLE, W.H., TURKEL, J., HENDERSON, D.C. et EVANS,J.R. (1979). Mapping the representation of the visual field by electrical stimulation of human visuel cortex. American Journal of Ophthalmology, 88, 727-735.

[17] BARTLETT, J.R. et DOTY, R.W. (1980). An exploration of the ability of macaques to detect microstimulation of striate cortex, Acta Neurobio. Exp., 40, 713-727.

[18] RONNER, S.F. et LEE, B.G. (1983). Excitation of visual cortex neuron by local intracortical microstimulation, Experimental Neurology, 81, 376-395.

[19] HAMBRETCH, F.T., BAK, M., GIRVIN, J.P., KUFFA, C.V., LOEB,G.E. et SCHMIDT,E.M. (1990). Visual sensations produced by intracortical microstimulation of the human occipital cortex. Med. & Bio. Eng. & Comput, 28, 257-259.

[20] HAMBRETCH, F.T., BAK, M., KUFFA, C.V., O'ROURKE, D.K., SCHMIDT, E.M. et VALLABHANATH, P. (1992). Feasibility of a visual prosthesis for the blind utilizing intracortical microstimulation, 4th Vienna Int. Workshop on

Functional Electrostimulation, 1-8.

[21] HAMBRETCH, F.T. (1995). Visual prostheses based on direct interfaces with the visual system, Baillière's Clinical Neurology, 4, no.1, 147-164

[22] HAMBRETCH, F.T., BAK, M., KUFFA, C.V., O'ROURKE, D.K., SCHMIDT, E.M. et VALLABHANATH, P. (1996). Feasibility of a visual prosthesis for the blind based on intracortical microstimulation of the visual cortex. Brain, 119, 507-522.

[23] CHA, K., HORCH, K. et NORMANN, R.A. (1992). Simulation of a Phosphene-Based Visual Field : Visual Acuity in a Pixelized Vision System, Annals of Biomedical Engineering, 20, no.4, 439-449.

[24] SELIGMAN, L.J. (1982). Physiological Stimulators : From Electric Fish to Programmable Implants. IEEE Transactions on Biomedical Engineering, BME-29, no 4, 270-284.

[25] ARABI, K. et SAWAN, M. (Janvier 1996). Implantable multiprogrammable microstimulator dedicated to bladder control. Med. & Biol. & Comput., 34, 9-12.

[26] SMITH, B., TANG, Z., JOHNSON, M.W., POURMEDHI, S., GAZDIK, M.M., BUCKETT, J.R. et PECKHAM, P.H. (1998). An Externally Powered, Multichannel, Implantable Stimulator-Telemeter for Control of Paralyzed Muscle. IEEE Transactions on Biomedical Engineering, 45, no.4, 463-475.

[27] SAYPOL, J.M, ROTH, B.J., COHEN, L.G. et HALLETT, M. (1991). A theoretical Comparaison of Electric and Magnetic Stimulation of the Brain Annals of Biomedical Engineering, 19, 317-328.

[28] DAVEY, K., LUO, L. et ROSS, D.A. (1994). Toward Functional Magnetic Stimulation (FMS) Theory and Experiment. . IEEE Transactions on Biomedical Engineering, 41, no.11, 1024-1030..

[29] WISE, K.D., ANGELL, J.B. et STARR, A. (juillet 1970). An integrated circuit

approach to extracellular microelectrodes. IEEE Trans. Biomed. Eng., BME-17, 238-247.

[30] BLUM, N.A., CARKHUFF, B.G., CHARLES, H.K., JR.,EDWARDS,R.L. et MEYER,R.L. (1991 janvier). Multisite microprobes for neural recordings. IEEE Trans. Biomed. Eng., 38, no.1, 68-74.

[31] MASTROTOTARO,J.J., MASSOUD, H.Z., PILKINGTON,T.C. et IDEKER,R.E. (1992 mars). Rigid and flexible thin-film multielectrode arrays for transmural cardiac recording. IEEE trans. Biomed. Eng., 39, no.3, 271-279.

[32] HOOGERWERF, A.C. et WISE, K.D. (1994). A Three-Dimensional Microelectrode Array for Chronic Neural Recording. IEEE Trans. Biomed. Eng., 41, no.12,1136-1145.

[33] JONES, K.E., CAMPBELL, P.K. et NORMANN, R.A. (1992). A Glass/Silicon Composite Intracortical Electrode Array. Annals of Biomedical Engineering, 20, 423-437.

[34] NAJAFI, K. et WISE, K.D. (1986). An implantable multielectrode array with on-chip signal processing. IEEE J. Solid-State Circuits, 21, no.6, 1035-1045.

[35] JI, J. et WISE, K.D. (1992). An Implantable CMOS Circuit Interface for Multiplexed Microelectrode Recording Arrays. IEEE J. Solid-State Circuits, 27, 433-443.

[36] TANGHE, S.J. et WISE, K.D. (1992). A 16-channel CMOS Neural Stimulating array. IEEE J. Solid-State Circuits, 27, 1819-1825.

[37] TROYK, P.R. et SCHWAN, M.A. (1992). Class E driver for transcutaneous power and data link for implanted electronic devices. Med. Biol. Eng. Comput., 30, 69-75.

[38] SHAH, M.R., PHILLIPS, R.P. et NORMANN, R.A. (1998). A Study of Printed Spiral Coils for Neuroprosthetic Transcranial Telemtry Application. IEEE Trans. Biomed. Eng., 45, no.7, 867-876.

[39] JONES, K.E. et NORMANN, R.A. (1997) An Advanced Demultiplexing System for Physiological Stimulation.IEEE Trans. Biomed. Eng., 44, no.12,1210-1220.

[40] TROYK, P.R. Multi-Channel Transcutaneous Cortical Stimulation System. Progress 99 contract #N01-NS-7-2365, report #7 for the contract period 10/31/98 – 1/31/, Neural Prosthesis Program (NPP), National Institute of Disorder and Stroke, National Institude of Healt (NIH).

[41] YAGI, T., WATANABE, M., UCHIKAWA, Y. ITO, N. et MATSUSHIMA, T. (1998). A Study on Hybrid Artificial Retina with Cultured Neural Cells and Semi-conductor Micro-device, Proc. Of 1998 IEEE International Joint Conf. On Neural Networks (IJCNN'98), 780-783

[42] KIM, C. et WISE, K.D. (1997). Low-Voltage Electronics for the stimulation of Biological Neural Networks Using Fully Complementary BiCMOS Circuits. IEEE Journal of Solid-State Circuits., 32, no.10, 1483-1490.

[43] McDERMOTT, H. (1989). An Advanced Multiple Channel Cochlear Implant. IEEE Trans. Biomed. Eng., 36, no.7, 789-797.

[44] JI, J., NAJAFI, K. et WISE, K.D. (1991) A Low-Noise Demultiplexing System for Active Multichannel Microelectrode Arrays. IEEE Trans. Biomed. Eng., 38, no.1, 75-81.

[45] KIM, C. et WISE. K.D. (1996). A 64-Site Multishank CMOS Low-Profile Neural Stimulating Probe. IEEE Journal of Solid-State Circuits., 31, no.9, 1230-1238.

[46] DYMOND, M.A. (1976) Characteristics of the Metal-Tissue Interface of Stimulation Electrodes. IEEE Trans. Biomed. Eng., BME-23, no.4, 274-280.

[47] GEDDES, L.A. (1997). Historical Evolution of Circuit Models for the Electrode-Electrolyte Interface. Annals of Biomedical Engineering, 25,1-14.

[48] MCREERY, D.B., AGNEW, W.F., YUEN, T.G.H. et BULLARA, L. (1990

octobre) Charge Density and Charge Per Phase as Cofactors in Neural Injury Induced by Electrical Stimulation. IEEE Trans. Biomed. Eng., 37, no.10, 996-1001.

[49] EDELL, D.J., TOI, V.V., McNEIL, M. et CLARK, L.D. (1992). Factors Influencing the Biocompatibility of Insertable Silicon Microshafts in Cerebral Cortex. IEEE Trans. Biomed. Eng., 39, no.6, 635-643.

[50] BEARD, R.B., HUNG, B.N. et SCHMUKLER, R. (1992). Biocompatibility Considerations at Stimulating Electrodes Interfaces. Annals of Biomedical Engineering, 20, 395-410.

[51] McCREERY, D.B., YUEN, T.G.H., AGNEW, W.F. et BULLARA, L.A. (1997). A Characterization of the Effects on Neuronal Excitability Due to Prolonged Microstimulation with Chronically Implanted Microelectrodes. IEEE Trans. Biomed. Eng., 44, no.10, 931-939.

[52] BOYER, A. (1995). Développement d'un stimulateur miniaturisé dédié à un implant visuel. Mémoire de maîtrise, École polytechnique, Canada.

[53] SUANING, G.J, LOVELL, N.H (octobre 1998). A 100 Channel Neural Stimulator for Excitation of Retinal Ganglion Cells, Proceedings IEEE EMBS, Hong Kong.

[54] McDERMOTT, H. (1991). A Custom-Designed Receiver-Stimulator Chip for an Advanced Multiple-Channel Hearing Prosthesis, IEEE Journal of Solid-State Circuits, 28, no.8, 1161-1164.

[55] CHEN, J., WISE, K.D., HETKE, J.F. et BLEDSOE, S.C.Jr. (1997) A Multichannel Neural Probe for Selective Chemical Delivery at the Cellular Level. IEEE Trans. Biomed. Eng., 44, no.8, 760-769.

[56] ZIAIE, B., NARDIN, M.D. COGHLAN, A.R. et NAJAFI, K. (1997). A Single-Channel Implantable Microstimulator for Functional Neuromuscular Stimulation. IEEE Trans. Biomed. Eng., 44, no.10, 909-919.

[57] CAMERON, T., LOEB, G.E., PECK, R.A., SCHULMAN, J.H., STROJNIK, P. et TROYK, P.R. (1997). Micromodular Implants to Provide Electrical Stimulation of Paralyzed Muscles and Limbs. IEEE Trans. Biomed. Eng., 44, no.9, 781-789.

[58] AKIN, T., NAJAFI, K. et BRADLEY, R.M. (1998). A Wireless Implantable Multichannel Digital Neural Recording System for a Micromachined Sieve Electrode. IEEE Journal of Solid-State Circuits, 33, no 1, 109-118.

[59] HUANG, Q. et OBERLE, M. (1998). A 0.5-mW Passive Telemetry IC for Biomedical Applications. . IEEE Journal of Solid-State Circuits, 33, no 7, 937-945.

[60] TROYK, P.R. et SCHAWN, M.A.K. (1992). Closed-loop Class E Transcutaneous Power and Data Link for MicroImplants. IEEE Trans. Biomed. Eng., 39, no.6, 589-599.

[61] TANG, Z., SMITH, B., SCHILD, J.H. et PECKHAM, P.H. (1995). Data Transmission from an Implantable Biotelemeter by Load-Shift Keying Using Circuit Configuration Modulator. IEEE Trans. Biomed. Eng., 42, no.5, 524-528.

[62] ST-AMAND, R. (1995). Conception d'une source de courant servant d'étage de sortie à un stimulateur neuromusculaire. Mémoire de maîtrise, École polytechnique, Canada.

[63] VOGHELL, J-C., SAWAN, M., ROY, M. et BOURRET, S. (décembre 1998) Programmable Current Source Dedicated to Implantable Microstimulators. Int. Conf. on Microelectronics, Monastir.

[64] RUTTEN, L.C. VAN WIER, H.J. et PUT, J.H.M. (1991). Sensitivity and Selectivity of Intraneural Stimulation Using a Silicon Electrode Array. . IEEE Trans. Biomed. Eng., 38, no.2,192-198.

[65] FORSTER, I.C. (avril 1981). Theorical Design and Implementation of a Transcutaneous, Multichannel Stimulator for Neural Prosthesis Applications.

Journal of Biomedical Engineering, 3, 107-120.

[66] SAWAN, M., DUVAL, F., HASSOUNA, M.M., LI,J-S. ELHILALI,M.M., LACHANCE,J., LECLAIRE,M., POURMEHDI,S. et MOUïNE,J. (1992). Computerized Transcutaneous Control of a Multichannel Implantable Urinary Prosthesis. IEEE Trans. Biomed. Eng., 39, no.6, 600-609.

[67] MAYER, S. GEDDES, L.A. BOURLAND, J.D. et OGBORN, L. (1992). Faradic resistance of the electrode/electrolyte interface. Medical & Biological Engineering & Computing, 30, 538-542.

[68] SCHWAN, H.P. (1992). Linear and Nonlinear Electrode Polarization and Biological Materials. Annals of Biomedical Engineering, 20, 269-288.

[69] HOCHMAIR, E. (1984). System Optimization for Improved Accuracy in Transcutaneous Signal and Power Transmission. IEEE Trans. Biomed. Eng., BME-31, no.2, 177-186.

[70] COBBOLD, R.S.C. (1974) Transducers for Biomedical Measurements : Principles and Applications. John Wiley & Sons, Toronto, 486 pages.

[71] ROUSCHE, P. et NORMANN R.A (1992). A Method for Pneumatically Inserting an Array of Penetrating Electrodes into Cortical Tissue. Annals of Biomedical Engineering, 20, 413-422.

[72] CIROVIC, M.M (1977). Integrated Circuits.Reston Publishing, Etats-Unis, 304 pages.

[73] DAUBERT, S.J. et VALLANCOURT, D. (1990 avril). Operation and Analysis of current copier circuits. IEE Proceedings, 137, no.2, 109-115.

[74] KLAASSEN, K.B. (1971 juin). Offset-Compensated Emitter-Follower. IEEE Journal of Solid-State Circuits,127-128.

[75] GREBENE, A.B. (1984). Bipolar And Mos Analog Integrated Circuit Design, John Wiley & Son, Toronto, 894 pages.

[76] JOHNS, D.A. et MARTIN,K. (1997) Analog Integrated Circuit Design, John Wiley & Son, Toronto, 706 pages.

[77] CAPELLE, C. TRULLEMANS, C. ARNO, P. VERAART, C. (1998). A Real-Time Experimental Prototype for Enhancement of Vision Rehabilitation Using Auditory Substitution. IEEE Transactions on Biomedical Engineering, 45, no.10, 1279-1293.

[78] ESSELLE, K.P. et STUCHLY, M.A. (1992). Neural Stimulation with Magnetic Fields : Analysis of Induced Electric Fields. IEEE Transactions on Biomedical Engineering, 39, no.7, 693-700.

[79] ROUSCHE, P.J. et NORMANN, R.A. (1999). Chronic Intracortical Microstimulation (ICMS) of Cat Sensory Cortex Using the Utah Intracortical Electrode Array. IEEE Transactions on Rehabilitation Engineering, 7, no.1, 56-68.

[80] ROBIN, S. (1998). Réalisation et tests d'un système implantable dédié à la stimulation neurale sélective. Mémoire de maîtrise, École polytechnique, Canada.

[81] ALTERA. (janvier 1998). Evaluating Power for Altera Devices. Application Note 74.

[82] ALTERA. (octobre 1998). FLEX 10K Embedded Programmable Logic Family. Data Sheet.

[83] ALTERA. (août 1998). Configuring FLEX 10K Devices. Application Note 59.

[84] HARVEY, J.F. ROY, M. SAWAN,M. (décembre 1999) Prototyping of a complete and functional visual stimulator. En préparation pour publication dans IEEE Transactions on Biomedical Engineering.

[85] HARVEY, J.-F., VAILLANCOURT, P., SAWAN, M. (octobre 1997). Image Acquisition and Reduction Dedicated to a Visual Implant. 1st Symposium on Advanced Biomaterials (ISAB), Montreal.

[86] BOYER, A., SAWAN, M. Stimuli Generator Dedicated for an Implantable Visual Miniaturized Stimulator. IEEE-EMBS 17th Int. Conf., Montréal, septembre 1995.

ANNEXE A

DÉTAILS DE CONCEPTION ET CIRCUIT DU PIVC

A.1 Introduction

Cette annexe présente certains détails de conception ainsi que les schémas du prototype retenu pour fabrication. La prochaine section présente une vue d'ensemble du circuit. Ensuite, on retrouve les schémas complet du circuit. Finalement, les autres sections présentent des listes des divers types d'E/S du CPLD.

A.2 Vue d'ensemble

Le PIVC comprend les sous-modules suivant : un décodeur Manchester; un contrôleur (CPLD) avec circuit de configuration, un CNA (16 canaux) avec circuit de référence en tension, 16 canaux de stimulations (convertisseurs tension–courant et décodeurs) et une matrice de DEL.

Le PIVC comporte deux cartes (circuits imprimés). La première carte (***carte maîtresse***) contient le décodeur Manchester, le contrôleur (CPLD-EPF10K130VGC599 de Altera) et le CNA (AD8600 de Analog Devices). La deuxième carte (***carte affichage***) contient les décodeurs et la matrice de DEL. Les deux cartes doivent être alimentées avec deux niveaux (+5V et –5V) de tension et une masse.

La carte maîtresse contient deux régulateurs de tension à 3.3V (VCCINT et VCCIO utilisé par le CPLD). Trois plans de masse (GNDDIO, GNDINT, GNA) et 4 plans d'alimentation (+5,-5, VCCINT et VCCIO) sont utilisés pour alimenter les divers sous-modules du système. Ces différents plans permettent de minimiser le bruit (provenant principalement du CPLD) induit dans la partie analogique du système. Des résistances

de Pull-up doivent être utilisées de façon à ce que le CPLD puisse commander le CNA et les démultiplexeurs qui fonctionnent à partir d'une alimentation de +5V.

La configuration du CPLD peut être accomplie de deux façons soit en mode JTAG à partir d'un ordinateur via un câble ***Byteblaster*** ou bien en mode passif sériel par des EPROM (U1,U6 – EPC1). La procédure pour configurer le FPGA est décrite dans [83].

Le courant maximal qui peut être fourni par chacune des sorties du démultiplexeurs (74HC4514) est de 25 mA. Ce courant de « stimulation » (qui passe par une des 16 DEL d'un canal) est généré lorsque la tension à l'entrée du convertisseur tension-courant est de 1.25V. Cette tension ne doit pas être dépassée.

Le CNA (16 canaux) utiiise une tension de référence pour générer les différents niveaux analogiques. Cette référence en tension est générée à partir d'un circuit comprenant un CI de référence 2.5 V (U2 – LT1009) et de trois amplificateurs opérationnels (U3 A,B et C – TLO74). La tension de sortie appliquée au CNA est variable et peut être ajustée à l'aide de R10. Cette tension ne doit pas dépasser 1.25V.

Des composants additionnels furent ajoutés à la carte maîtresse afin de la rendre un peu plus universelle et ainsi être utilisée pour d'autres applications. Parmi ces composants, on retrouve un oscillateur (U4), trois séries de huit interrupteurs, 3 boutons poussoirs ainsi que quatre ampli OP utilisés en suiveur pour amener chacune des quatre premières sorties du CNA à un connecteur BNC.

Le décodeur Manchester est utilisé pour séparer l'horloge des données. Les valeurs du condensateur C13 et de la résistance R7 doivent être choisies en fonction de la fréquence de horloge encodé dans le signal d'entrée. Le testeur développé dans le cadre de ce projet de maîtrise produit un signal d'horloge dont la fréquence est égale à 1.5734 Mhz donc une période de 635.5511 ns. Les valeurs de R7 (2.7K) et C13 (100pF) furent choisies pour que la constante de temps résultante permettent la décharge de C13 en un temps équivalent à un peu plus de la moitié de cette période.

A.3 Schéma complet du circuit du prototype

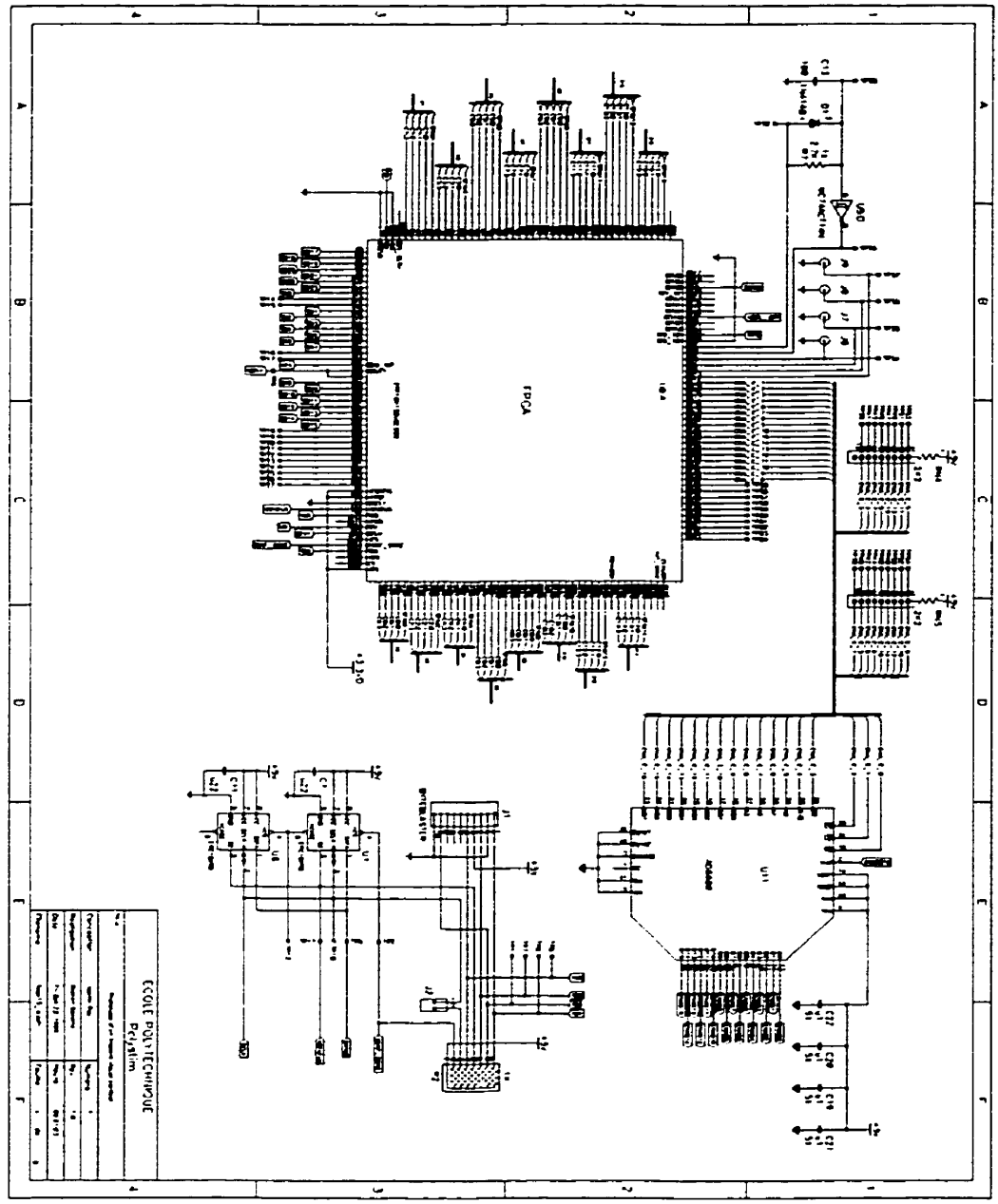


Figure A.1 : Circuit de la carte maîtresse (1/3)

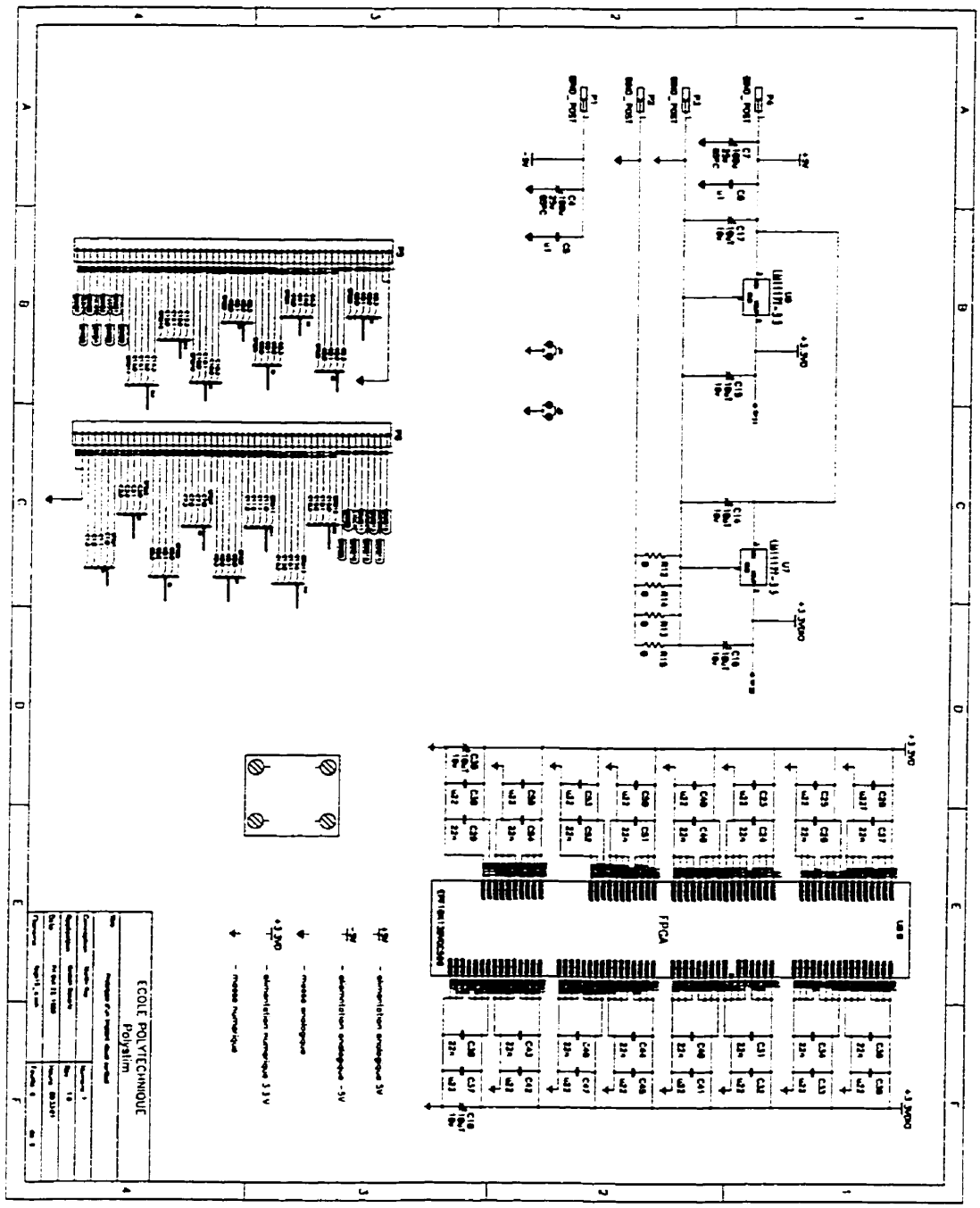


Figure A.2 : Circuit de la carte maîtresse (2/3)

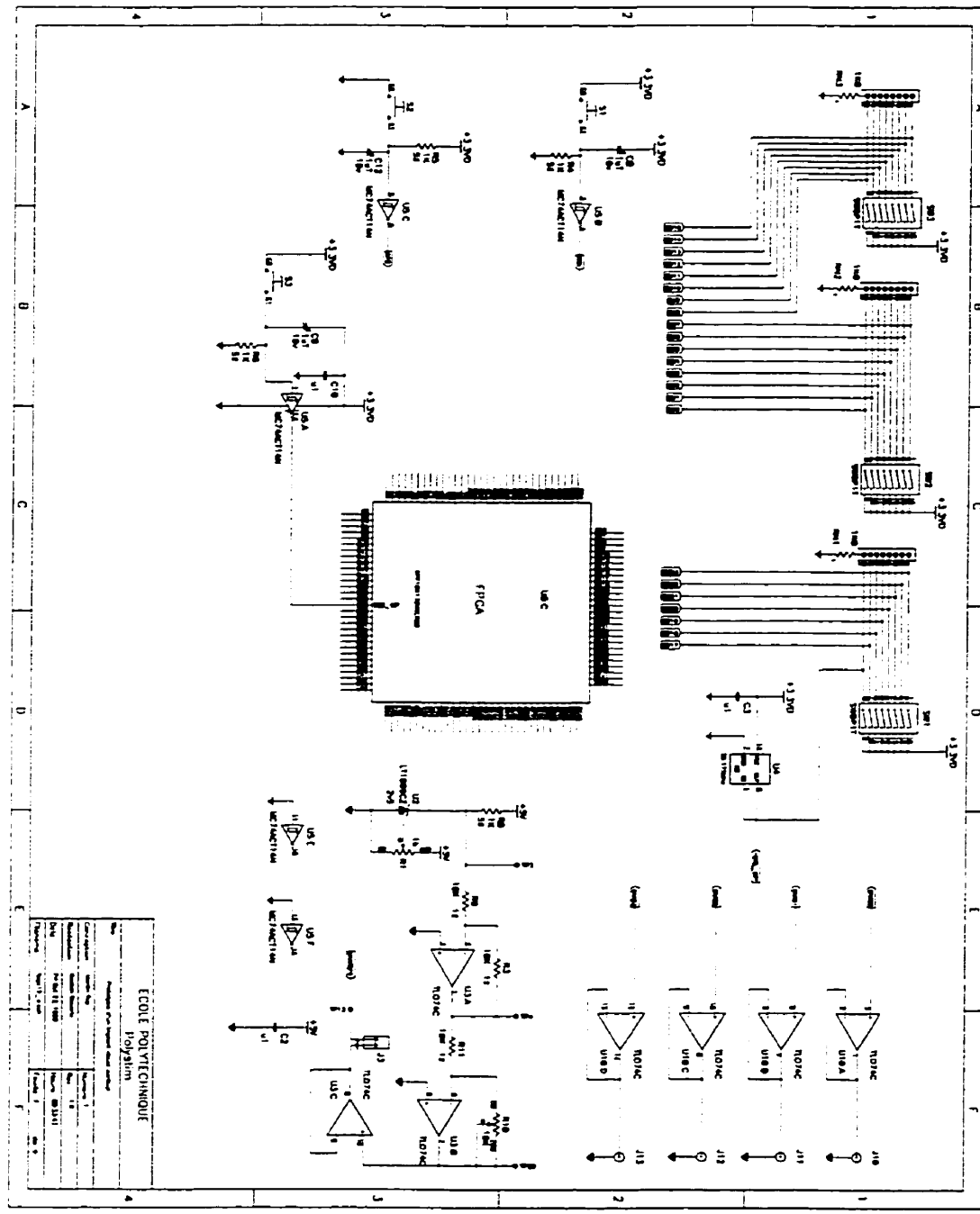


Figure A.3 : Circuit de la carte maîtresse (3/3)

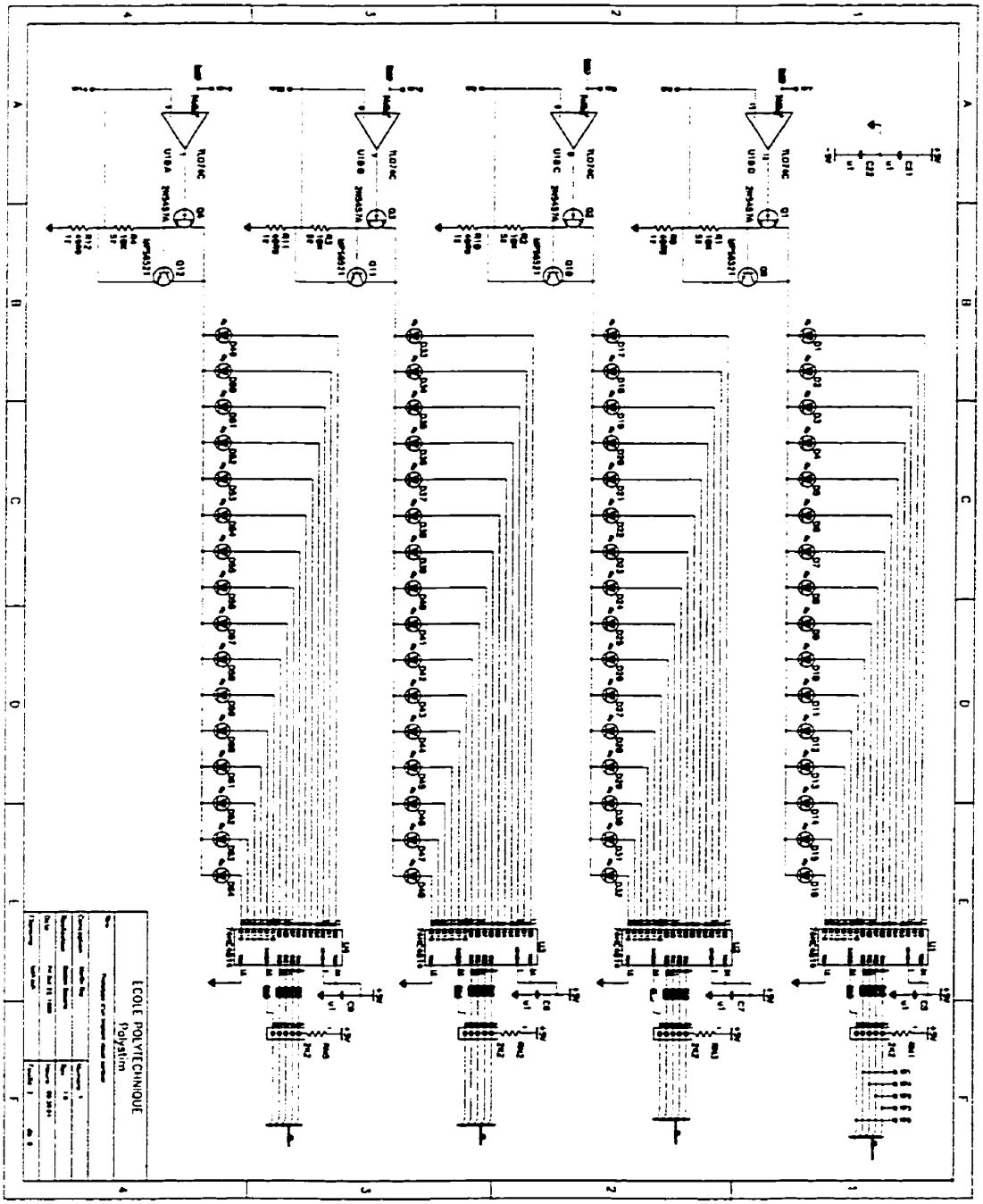


Figure A.4 : Circuit de la carte affichage (1/5)

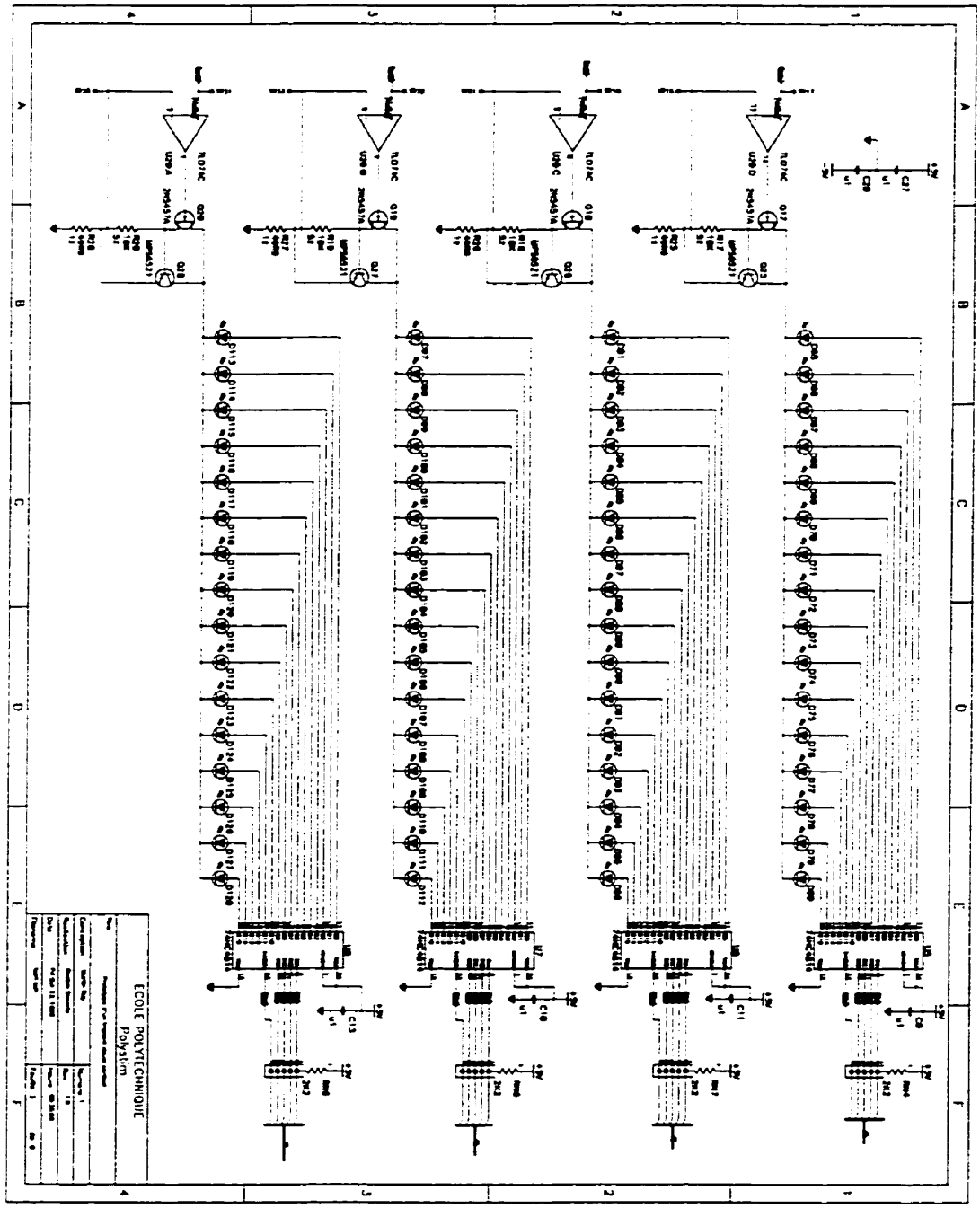


Figure A.5 : Circuit de la carte affichage (2/5)

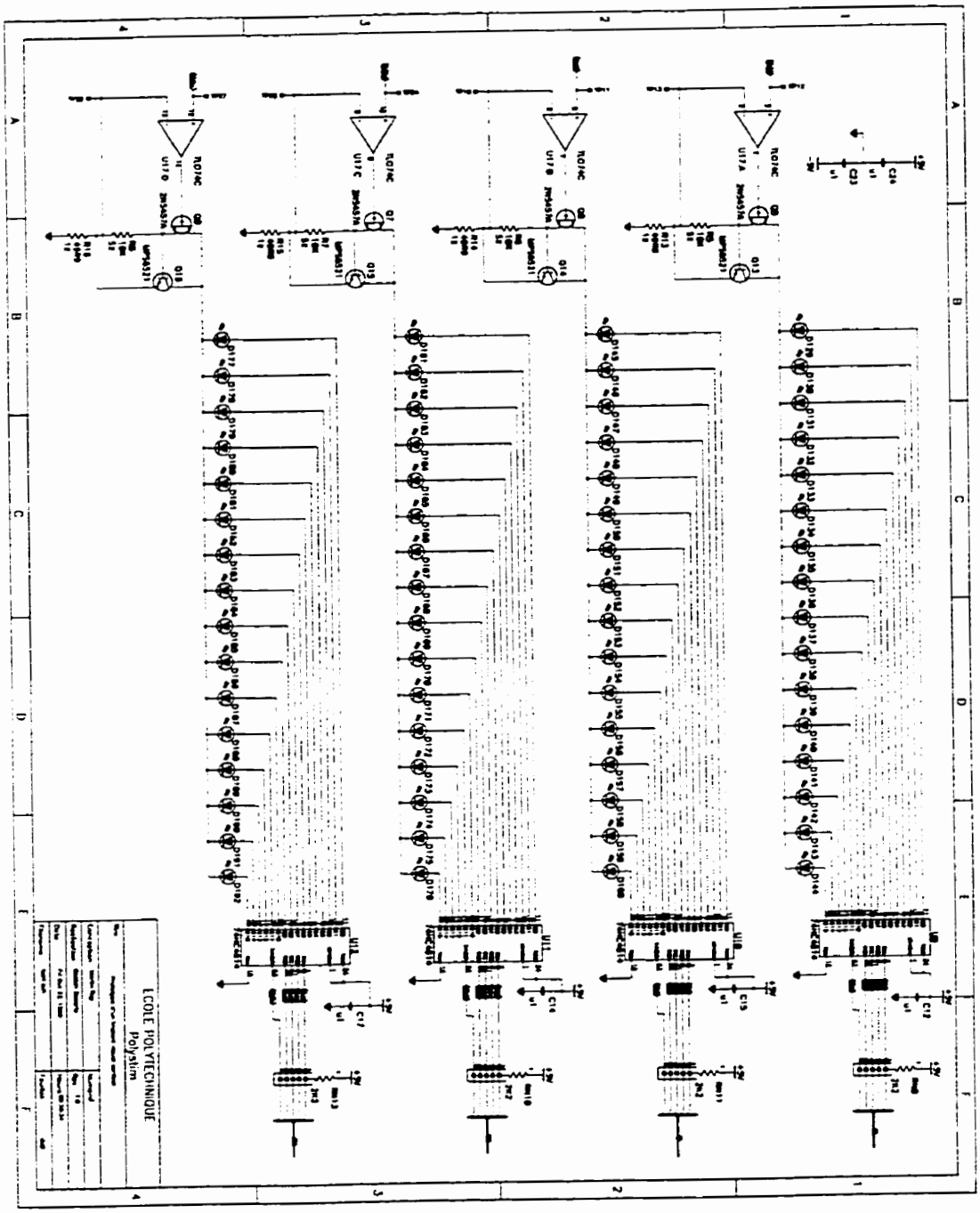


Figure A.6 : Circuit de la carte affichage (3/5)

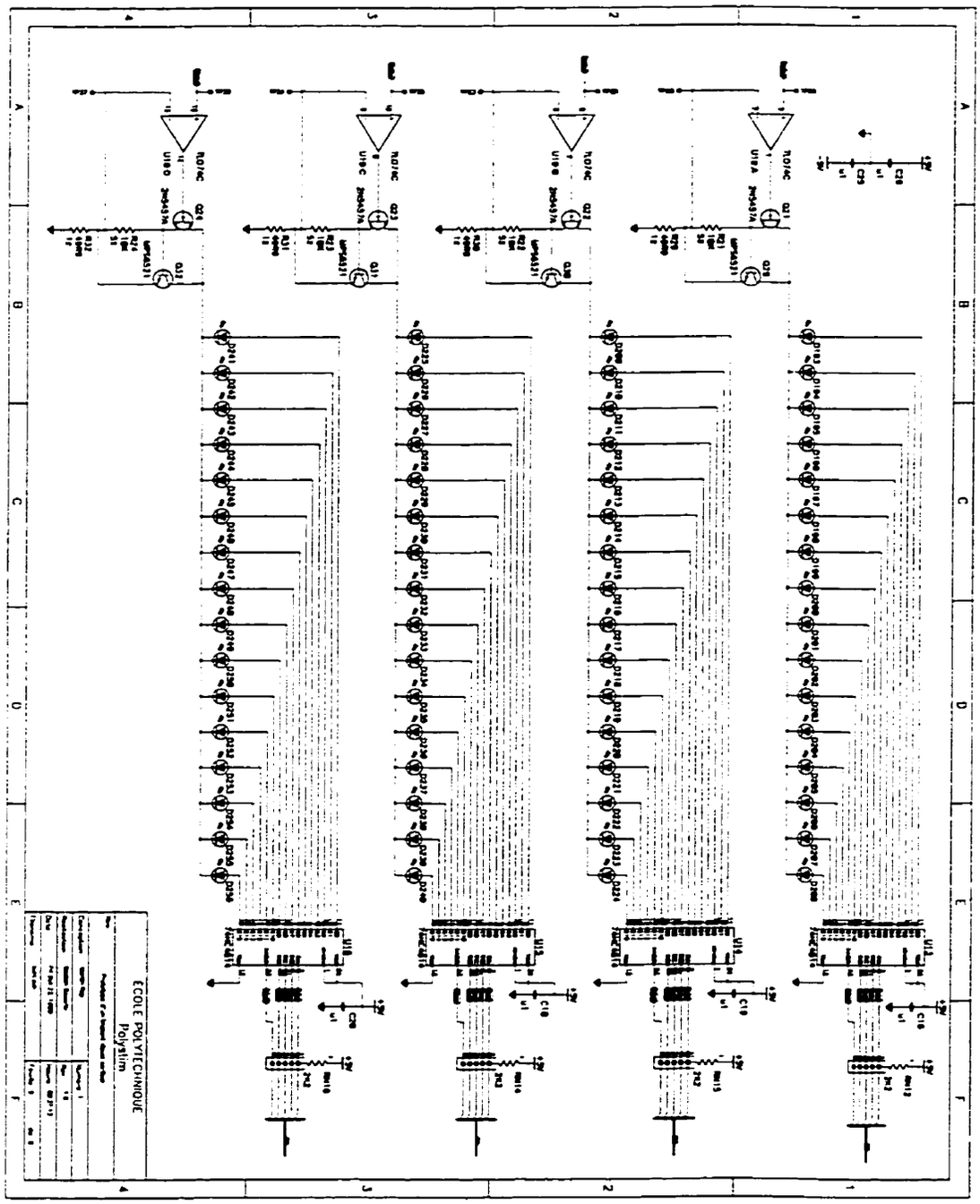


Figure A.7 : Circuit de la carte affichage (4/5)

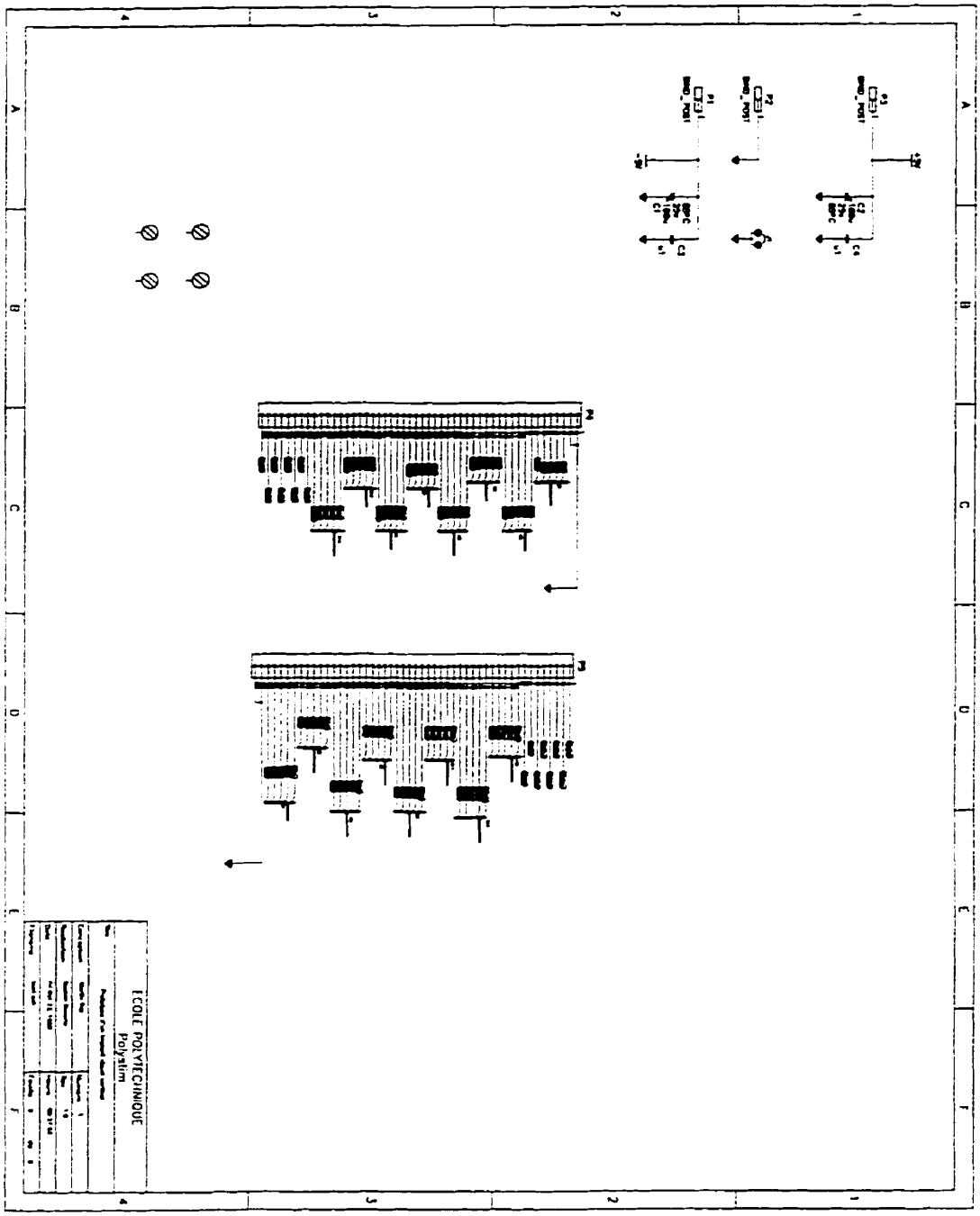


Figure A.8 : Circuit de la carte affichage (5/5)

A.4 Identification des broches spéciales et d'alimentation du CPLD

Cette section présente un tableau qui liste les broches spéciales utilisées pour la configuration du CPLD [82,83].

Tableau A.1 Liste des broches spéciales et d'alimentation du CPLD

Nom de la PIN	# de broche	Type	Connexion
MSEL0	F6	E(C)	GND
MSEL1	C3	E(C)	GND
NSTATUS	E43	B(C)	EPC1
NCONFIG	B4	E(C)	VDD
DCLK	BE5	E(C)	EPC1
CONF_DONE	BC43	B(C)	EPC1
INIT_DONE	AM40	S(C)	NC
nCE	BB6	E(C)	GND
nCEO	BF44	S(C)	NC
nWS	BB40	E(C)	NC
nRS	BA37	E(C)	VCCIO
nCS	AY38	E(C)	NC

CS	BA39	E(C)	NC
RDYnBSY	AW47	S(C)	NC
CLKUSR	AY42	E(C)	NC
DATA7	BD14	E(C)	NC
DATA6	BA17	E(C)	NC
DATA5	BB16	E(C)	NC
DATA4	BF12	E(C)	NC
DATA3	BG11	E(C)	NC
DATA2	BG9	E(C)	NC
DATA1	BF10	E(C)	NC
DATA0	BC5	E(C)	NC
TDI	BF4	E(CT)	BB
TDO	BB42	S(CT)	BB
TCK	BE43	E(CT)	BB
TMS	F42	E(CT)	BB
TRST	B46	E(CT)	VCCIO
Entrées dédiées	B24,C25, BG25, BG23	E	GNDIO si

			non utilisées
Pin d'horloge dédié	BF24, A25	E	horloge du système
DEV_CLRn	BE23	E	NC
DEV_OE	BC25	E	NC
VCCINT	E5, A3, A45, C1, C11, C19, C29, C37, C47, G25, L3, L45, W3, W45, AJ3, AJ45, AU3, AU45, BE1, BE11, BE19, BE29, BE37, BE47, BG3, BG45	VCCINT	3.3
VCCIO	D24, E9, E15, E21, E27, E33, E39, G7, G41, J5, J43, R5, R43, AA5, AA43, AD4, AD44, AG5, AG43, AN5, AN43, AW5, AW43, BA7, BA41, BC9, BC15, BC21, BC27, BC33, BC39, BD24	VCCIO	3.3
GNDINT	A47, B2, C13, C21, C27, C35, C45, D4, G23, N3, N45, AA3, AA45, AG3, AG45, AR3, AR45, BD44, BE3, BE13, BE21, BE27, BE35, BE45, BG1, BG47	GNDINT	0
GNDIO	E7, E13, E19, E29, E35, E41, F24, G5, G43, H40, N5, W5, W43, AD6, AD42, AJ5, AJ43, AR5, AR43, AY8, AY40, BA5, BA43, BB24, BC7,	GNDIO	0

	BC13, BC19, BC29, BC35, BC41, N43		

Légende :

B : broche bidirectionnel.

B(C) : broche bidirectionnel utilisé comme entrée de configuration.

E : Entrée dédié

E(C) : Entrée de configuration.

S(C) : Sortie de configuration

NC : Non connecté.

A.5 Liste des E/S du CPLD

ENTRÉES

Tableau A.2 Liste des entrées du contrôleur et broches associées

Nom du signal (intérieur du CPLD)	# broche du CPLD	Connexion/explication
Donnees_direct	BG7	J7/ données (non codées) du cont. Ext.
Horl	A25	CPLD(B26)/ signal d'horloge sélectionné
Horl_direct	BF8	J8/signal d'horloge (non codé)
Raz_manch	BG5	Sortie du Schmitt trigger (U5 :D)
Signal_manch	BG13	J9/signal codé Manchester
Tx_mode	A11	SW3(broche 1)/sélection du mode de TX

SORTIE

Tableau A.3 Liste des sorties du contrôleur et broches associées

Nom du signal (intérieur du CPLD)	# broche du CPLD	Type	Connexion/explication
cna_a0	BG21	S-DO	CNA
Cna_a1	BF20	S-DO	CNA
Cna_a2	BG19	S-DO	CNA
Cna_a3	BF18	S-DO	CNA
Cna_cs	BG15	S-DO	CNA
Cna_db0	BF32	S-DO	CNA
Cna_db1	BG31	S-DO	CNA
Cna_db2	BF30	S-DO	CNA
Cna_db3	BG29	S-DO	CNA
Cna_db4	BF28	S-DO	CNA
Cna_db5	BG27	S-DO	CNA
Cna_db6	BF26	S-DO	CNA
Cna_db7	BF22	S-DO	CNA
Cna_en	BF16	S-DO	CNA
Cna_ls	BF14	S-DO	CNA
Cna_rs	BG33	S-DO	CNA
Cna_rw	BG17	S-DO	CNA
Horl_manch	BF2	S	Anode D1 / Horloge décodé (du signal manchester)

horl_sortie	B26	S	-
site3	D46	S-DO	-
site2	E47	S-DO	-
site1	F46	S-DO	-
site0	G47	S-DO	-
stim0	H46	S-DO	-
site7	D2	S-DO	-
site6	E1	S-DO	-
site5	F2	S-DO	-
site4	G1	S-DO	-
stim1	H2	S-DO	-
site11	J47	S-DO	-
site10	K46	S-DO	-
site9	L47	S-DO	-
site8	M46	S-DO	-
stim2	N47	S-DO	-
site15	J1	S-DO	-
site14	K2	S-DO	-
site13	L1	S-DO	-
site12	M2	S-DO	-
stim3	N1	S-DO	-
site19	P46	S-DO	-
site18	R47	S-DO	-
site17	T46	S-DO	-
site16	U47	S-DO	-
stim4	V46	S-DO	-
site23	P2	S-DO	-
site22	R1	S-DO	-
site21	T2	S-DO	-

site20	U1	S-DO	-
stim5	V2	S-DO	-
site27	W47	S-DO	-
site26	Y46	S-DO	-
site25	AA47	S-DO	-
site24	AB46	S-DO	-
stim6	AC47	S-DO	-
site31	W1	S-DO	-
site30	Y2	S-DO	-
site29	AA1	S-DO	-
site28	AB2	S-DO	-
stim7	AC1	S-DO	-
site35	AD46	S-DO	-
site34	AE47	S-DO	-
site33	AF46	S-DO	-
site32	AG47	S-DO	-
stim8	AH46	S-DO	-
site39	AD2	S-DO	-
site38	AE1	S-DO	-
site37	AF2	S-DO	-
site36	AG1	S-DO	-
stim9	AH2	S-DO	-
site43	AJ47	S-DO	-
site42	AK46	S-DO	-
site41	AL47	S-DO	-
site40	AM46	S-DO	-
stim10	AN47	S-DO	-
site47	AJ1	S-DO	-
site46	AK2	S-DO	-

site45	AL1	S-DO	-
site44	AM2	S-DO	-
stim11	AN1	S-DO	-
site51	AP46	S-DO	-
site50	AR47	S-DO	-
site49	AT46	S-DO	-
site48	AU47	S-DO	-
stim12	AV46	S-DO	-
site55	AP2	S-DO	-
site54	AR1	S-DO	-
site53	AT2	S-DO	-
site52	AU1	S-DO	-
stim13	AV2	S-DO	-
Site59	AY46	S-DO	-
site58	BA47	S-DO	-
site57	BB46	S-DO	-
site56	BC47	S-DO	-
stim14	BD46	S-DO	-
site63	AW1	S-DO	-
site62	AY2	S-DO	-
site61	BA1	S-DO	-
site60	BB2	S-DO	-
stim15	BC1	S-DO	-

Légende :

S : sortie

S-DO : sortie drain ouvert

POINTS DE TEST

Tableau A.4 Liste des points de test du contrôleur et broches associées

Nom du signal (intérieur du CPLD)	# broche du CPLD	Explication
razg	BF34	signal interne
pt_q1	BG35	signal interne
pt_q2	BF36	signal interne
bpv	BG37	signal interne
trmdtc_sort	BF38	signal interne
mcv_sort	BG39	signal interne
moderam	BF40	signal interne
raz_adresse	BG41	signal interne
horls	BF42	signal interne
pt_donnees	BG43	signal interne
donnees_manch	A35	signal interne
lire	B36	signal interne
ecrire	A37	signal interne
donnees_sort	B38	signal interne
trame_ech_sort0	A39	signal interne qui correspond à trame_don(0)
trame_ech_sort1	B40	signal interne qui correspond à trame_don(15)
trame_ech_sort2	A41	signal interne qui correspond à trame_don(30)
mcmd_ech_sort0	B42	signal interne qui correspond à mcmd(0)
mcmd_ech_sort1	A43	signal interne qui correspond à mcmd(15)
mcmd_ech_sort2	B44	signal interne qui correspond à mcmd(25)

ANNEXE B

CIRCUIT COMPLET DU CEB

Cette annexe présente les schémas des circuits de tous les modules du CEB. L'édition des schémas fut réalisée à l'aide du logiciel Cadence dans le but d'en faire la simulation avec Hspice. La figure B.1 montre le schéma bloc général du CEB comprenant notamment le CNA (DAC sur le schéma), le miroir de courant (type P), les 12 EB ainsi que chacun de leur commutateur de courant (ITA de la figure 3.7).

La figure B.2 montre la structure interne d'un EB comprenant l'échantillonneur-bloqueur (sampler) en mode courant qui commande une source de courant (current sink). La figure B.3 présente le circuit de la source de courant faisant partie du EB et la figure B.4 le circuit de l'échantillonneur-bloqueur de l'EB.

Le courant qui doit être échantillonné par un des EB provient du miroir de courant de type P dont le circuit est montré à la figure B.5. Le miroir amplifie d'un certain multiple le courant de sortie du CNA en mode courant (figure B.6). Celui-ci utilise un amplificateur opérationnel dont le circuit est présenté à la figure B.7.

La figure B.8 est un schéma de la structure du commutateur de courant (ITA de la figure 3.7) qui a pour tâche d'aiguiller l'entrée/sortie de l'EB soit vers le miroir de courant ou vers la sortie ***Istim***. Ce commutateur comprend un circuit de commande illustré à la figure B.9 et deux commutateurs de courant placés en parallèle. Le circuit de ces commutateurs est montré à la figure B.10. La figure B.11 montre le dessin du masque utilisé pour réaliser le CEB avec la technologie bipolaire prédiffusé GA911.

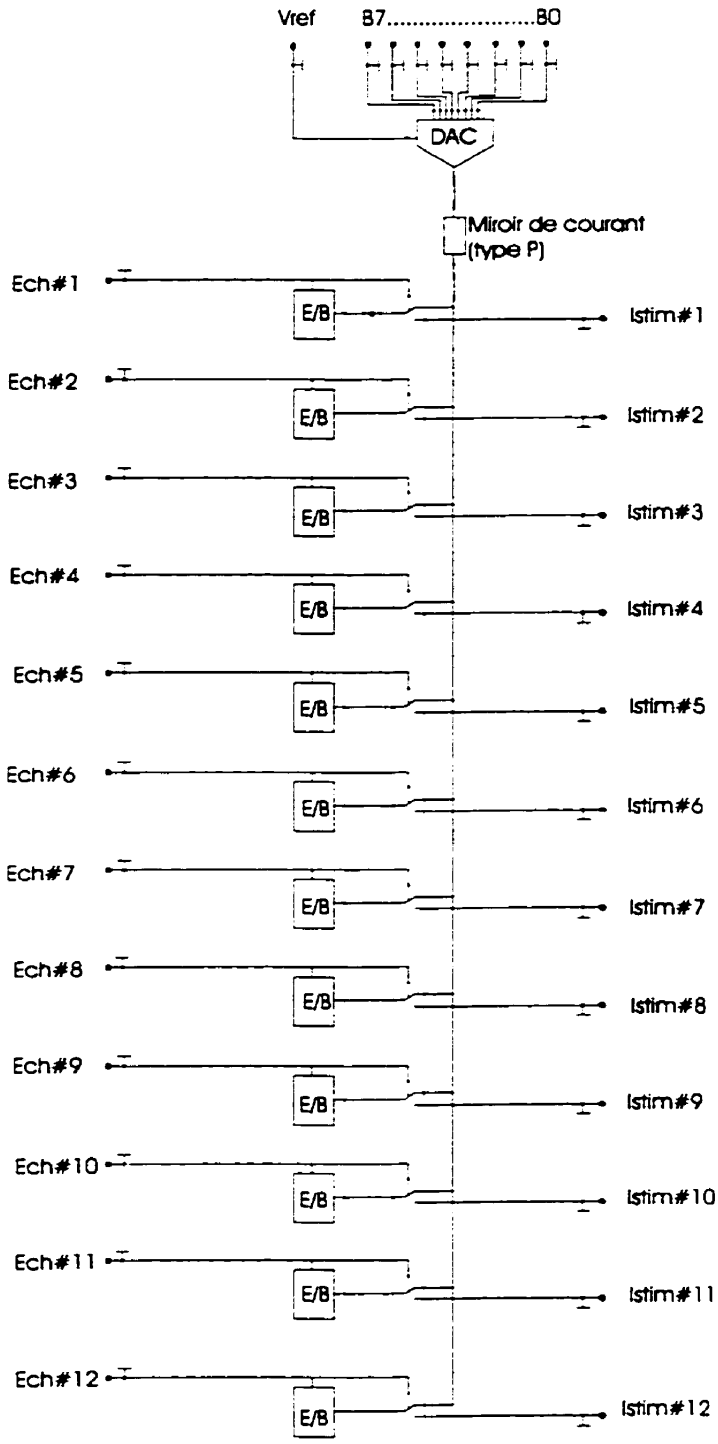


Figure B.1: Schéma bloc du CEB

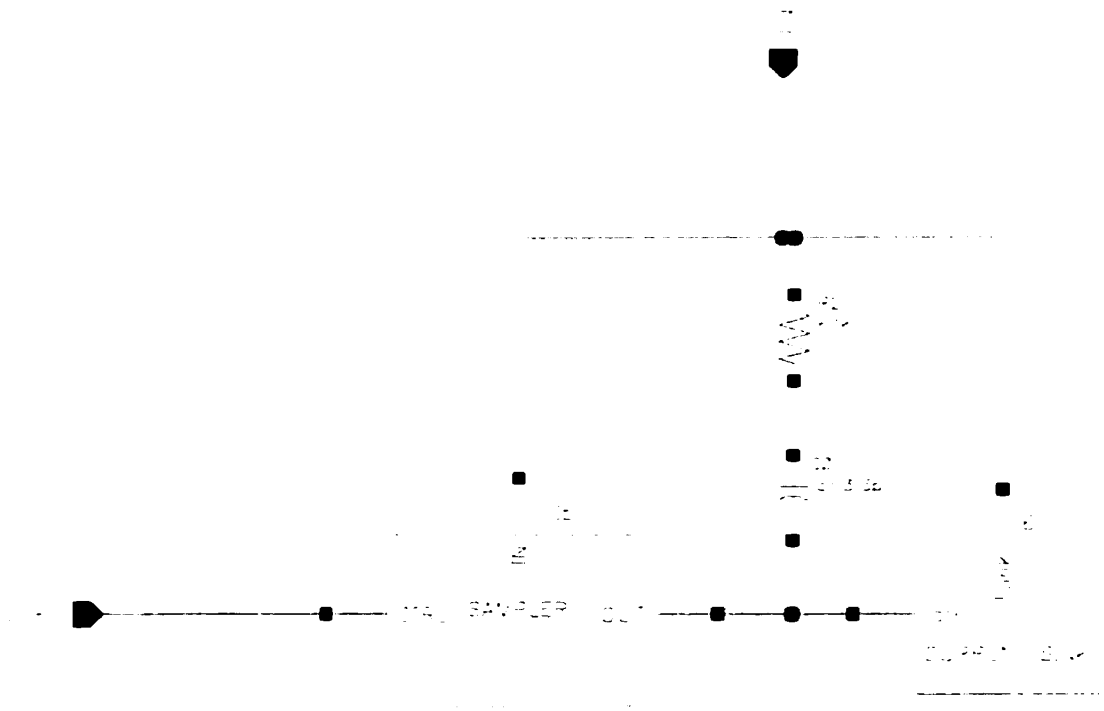


Figure B.2: Structure interne d'un EB

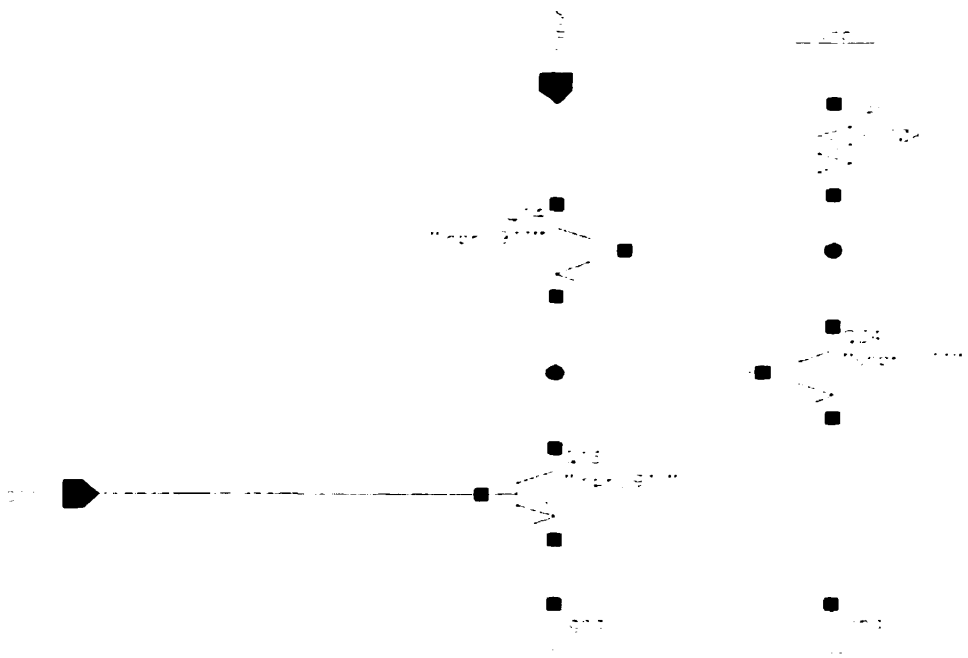


Figure B.3: Source de courant d'un EB

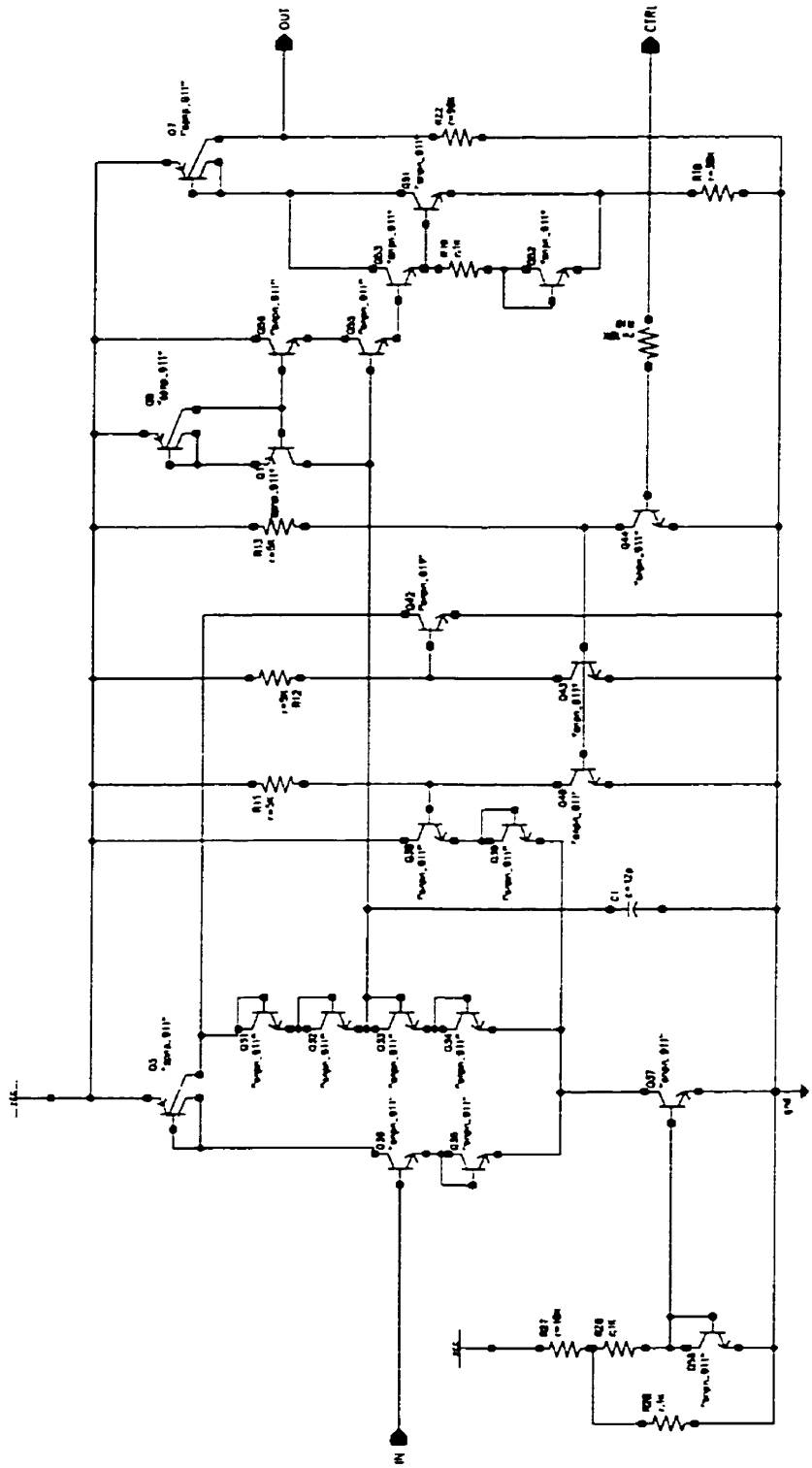


Figure B.4 : Échantillonneur-bloqueur du EB

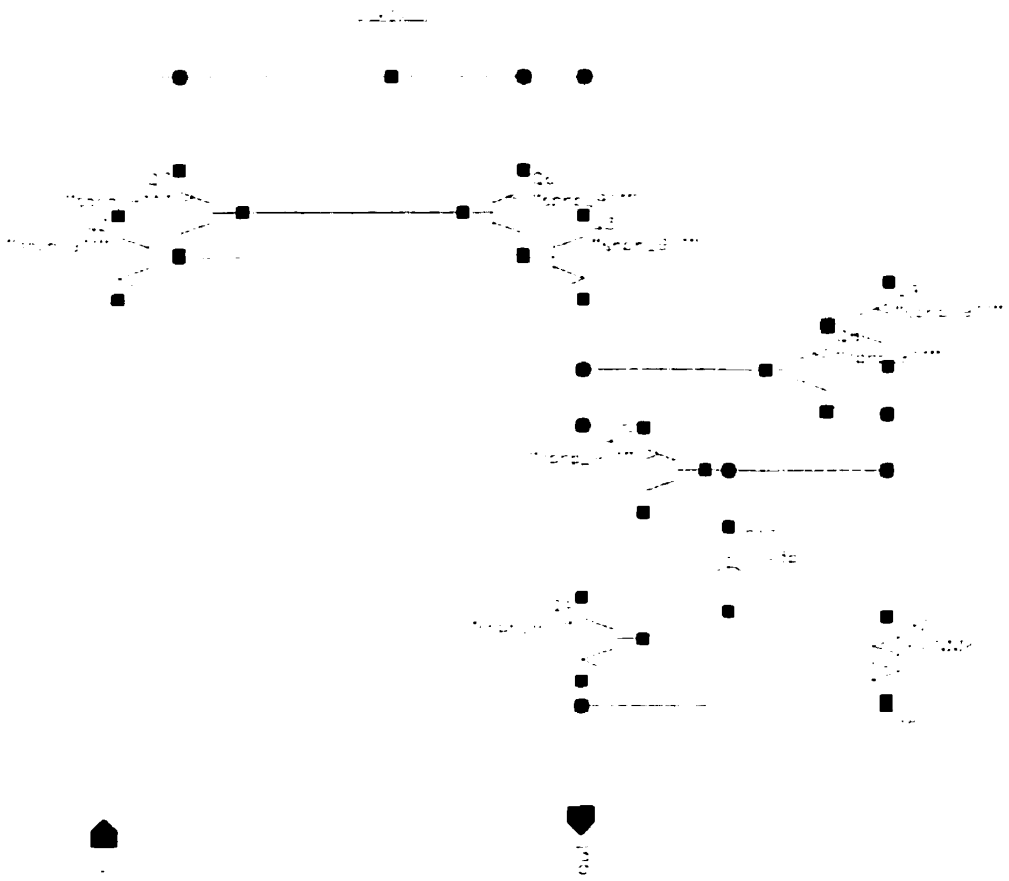


Figure B.5: Miroir de courant (type P)

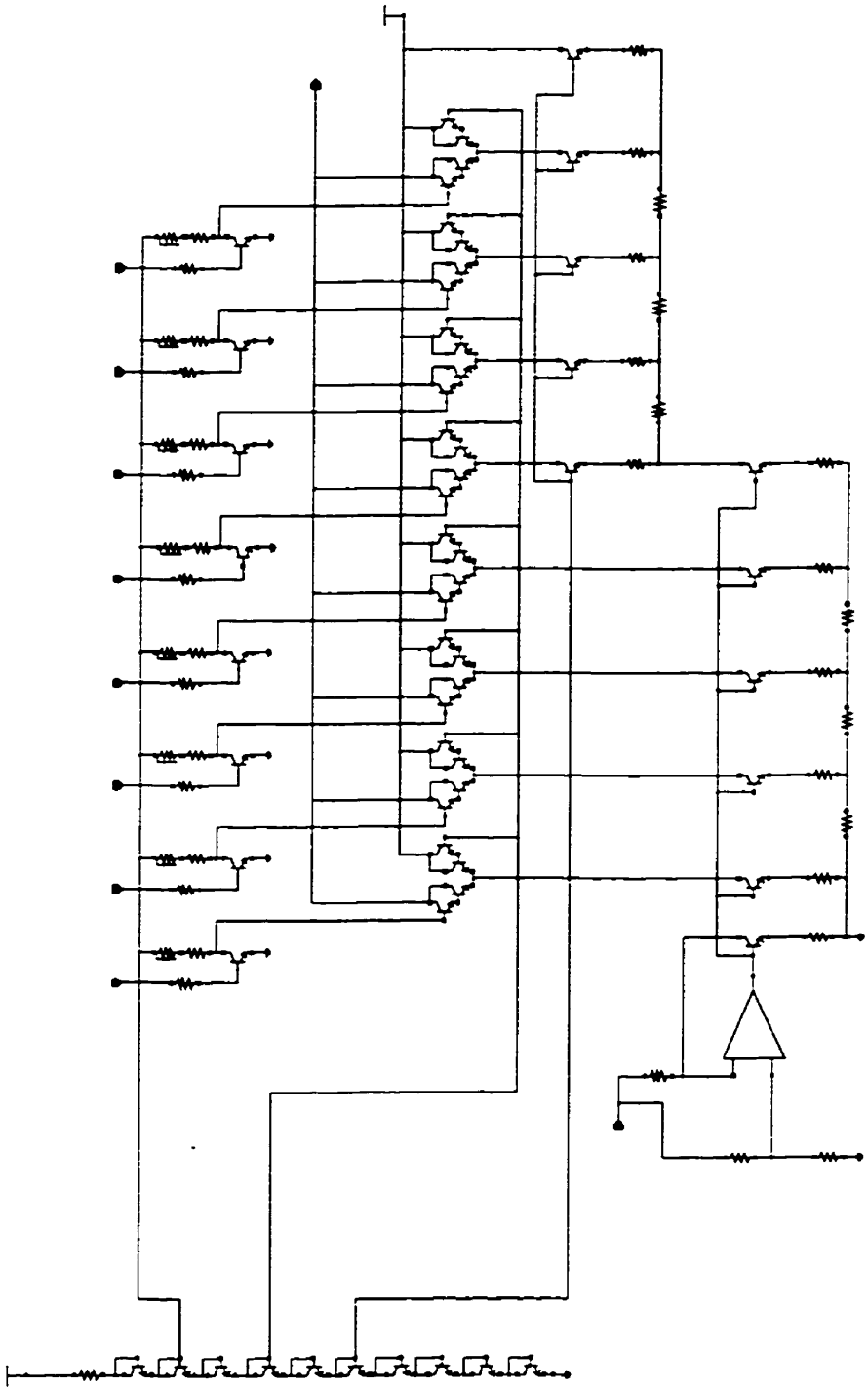


Figure B.6: Convertisseur numérique-analogique (CNA)

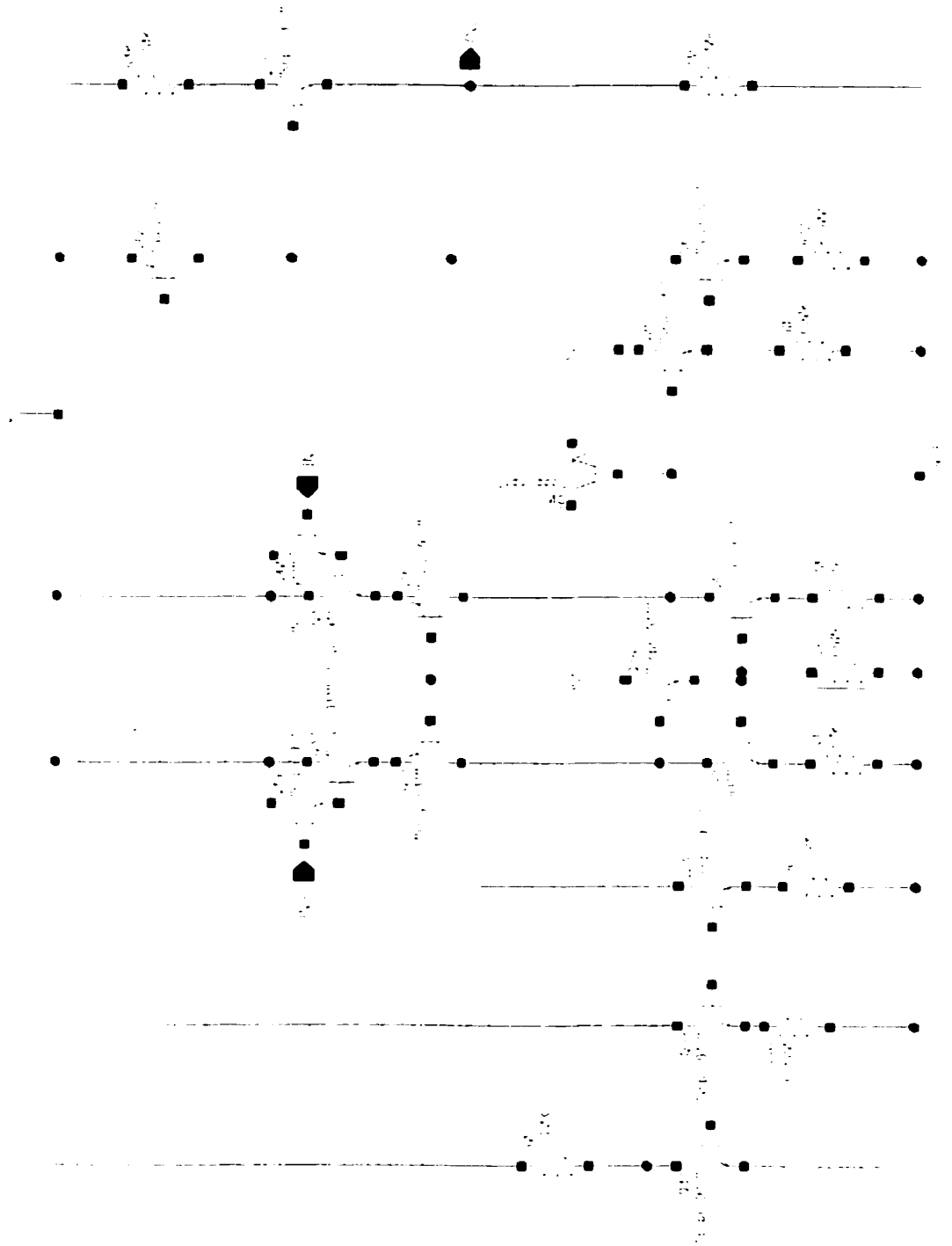


Figure B.7: Amplificateur opérationnel utilisé par le CNA

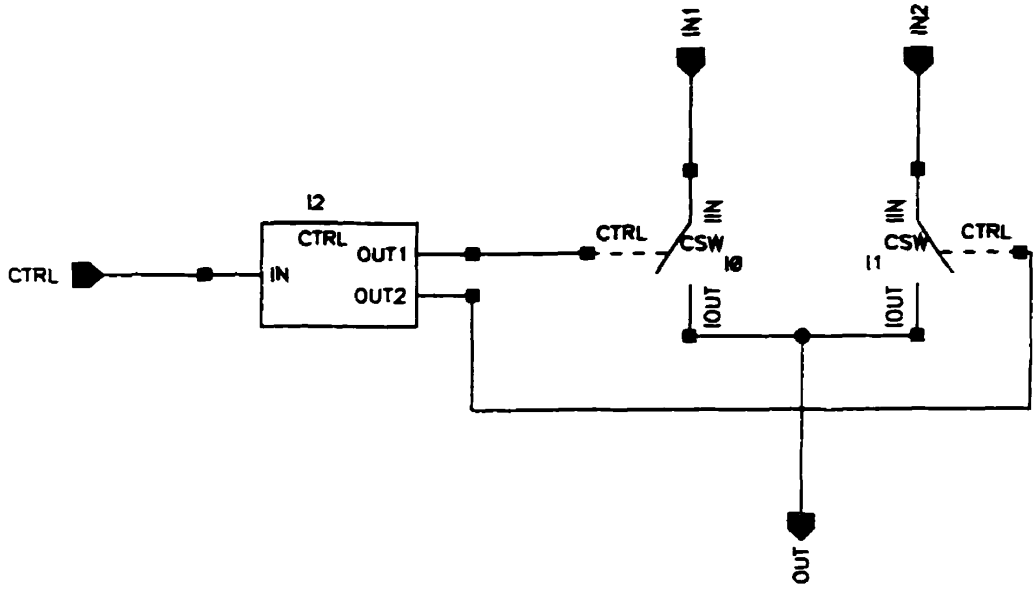


Figure B.8 : Schéma bloc du commutateur de courant

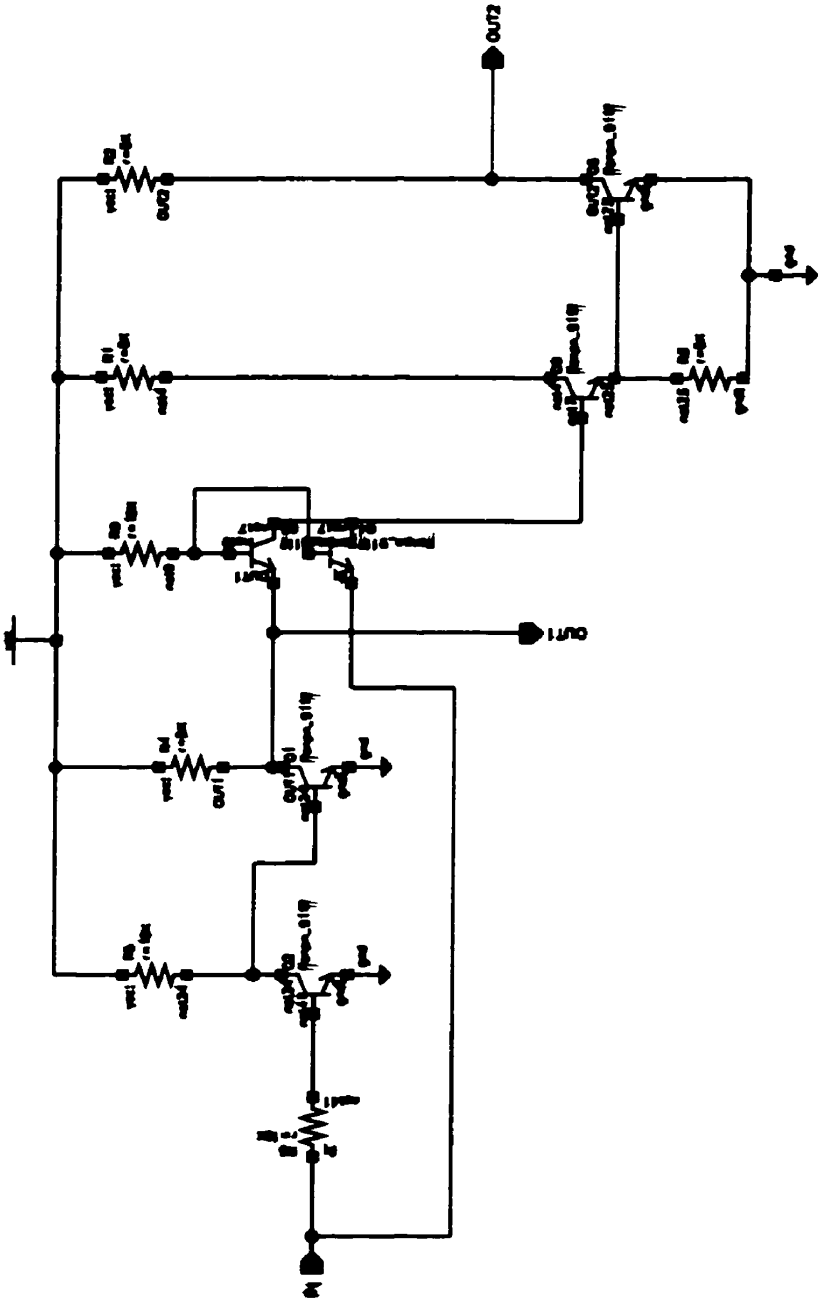


Figure B.9 : Circuit de contrôle du commutateur de courant

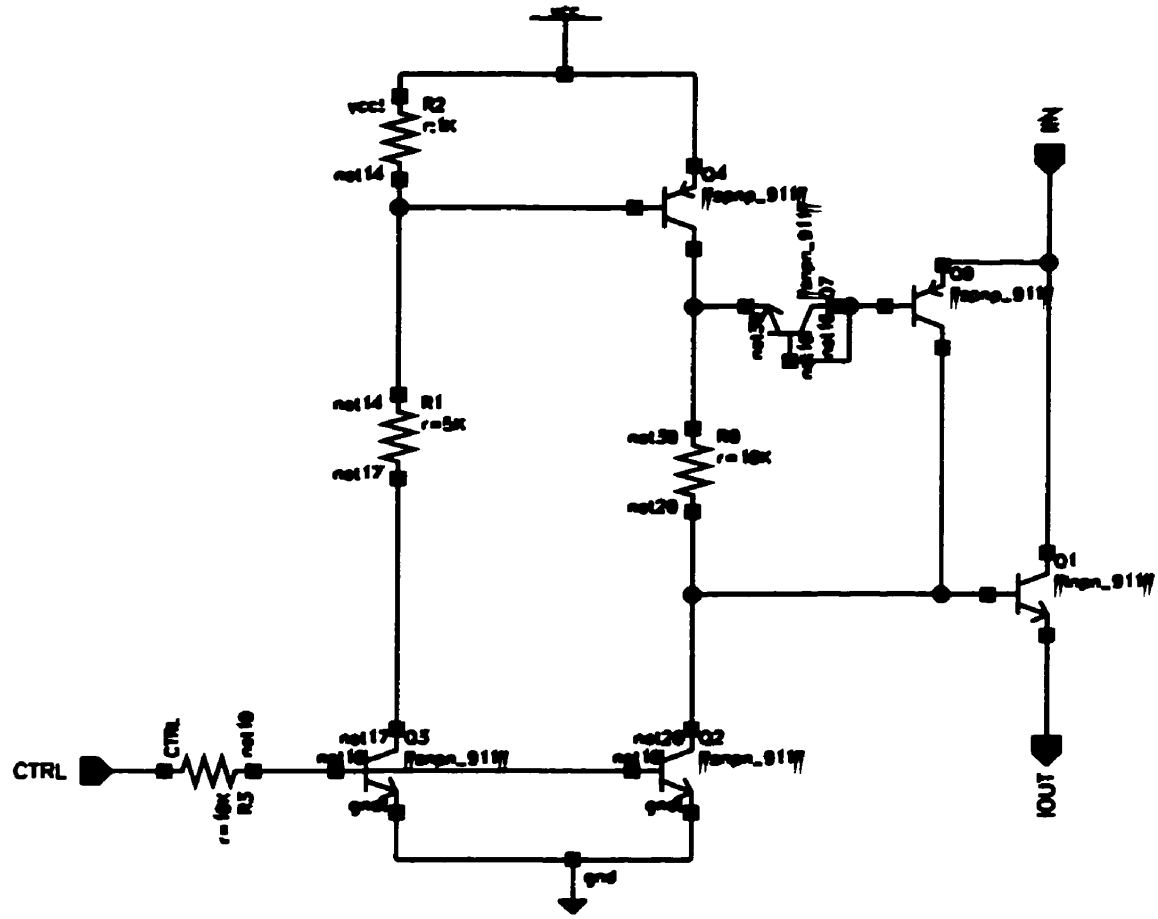


Figure B.10 : Circuit du commutateur de courant

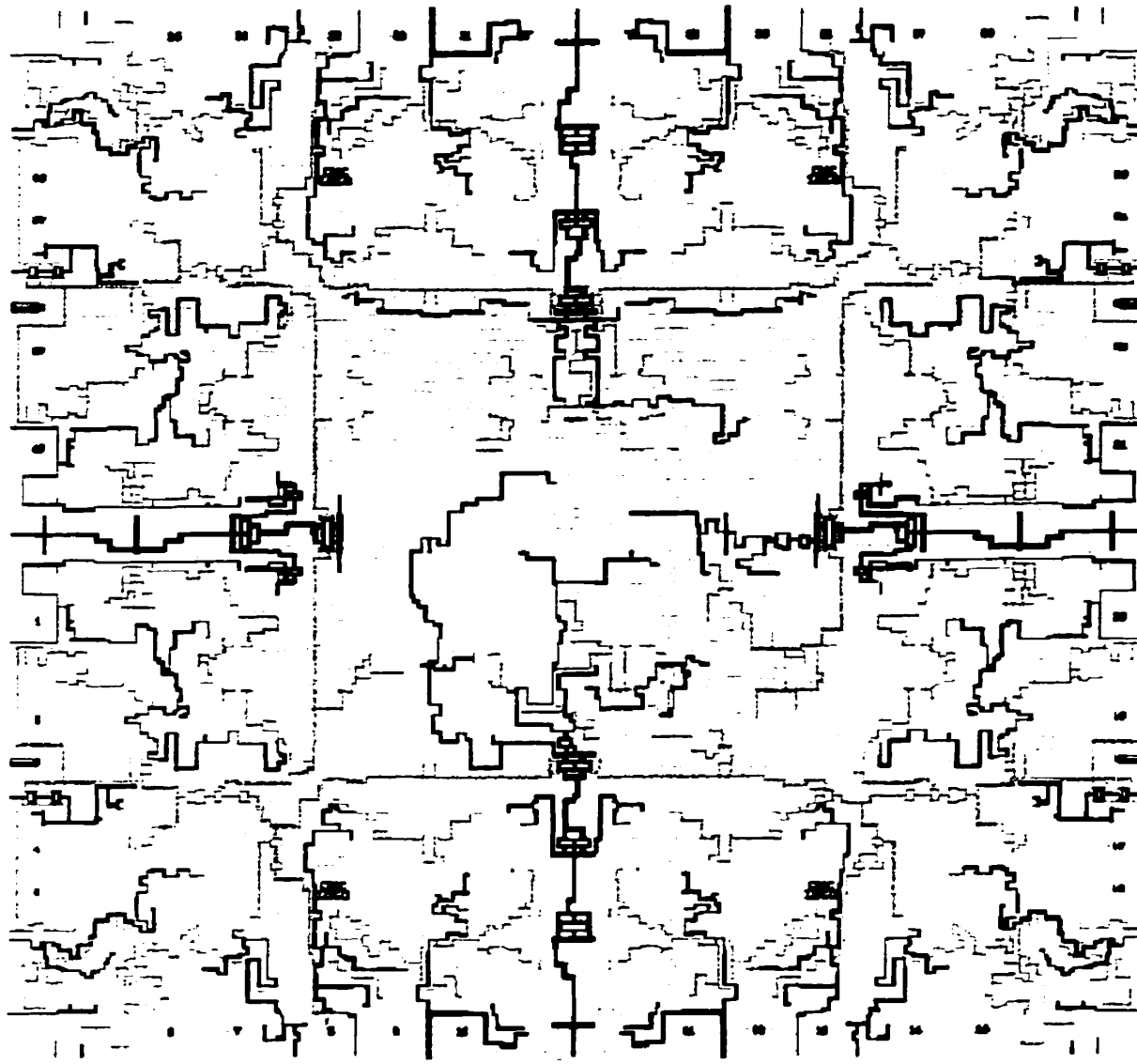


Figure B.11 : Dessin du masque de métal du CEB

ANNEXE C

RÉSULTATS DE SIMULATION DU CEB

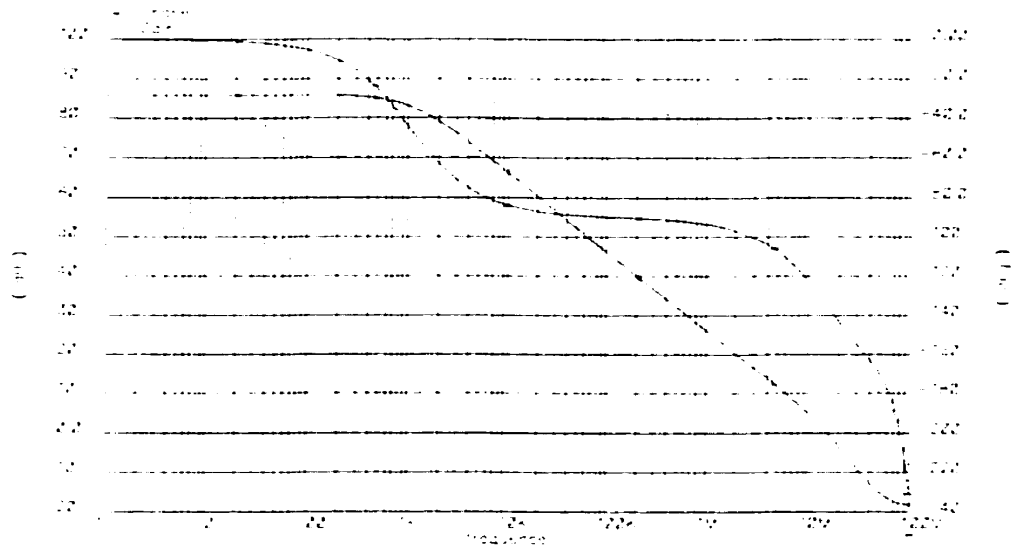


Figure C.1 : Réponse en fréquence de l'amplificateur opérationnel

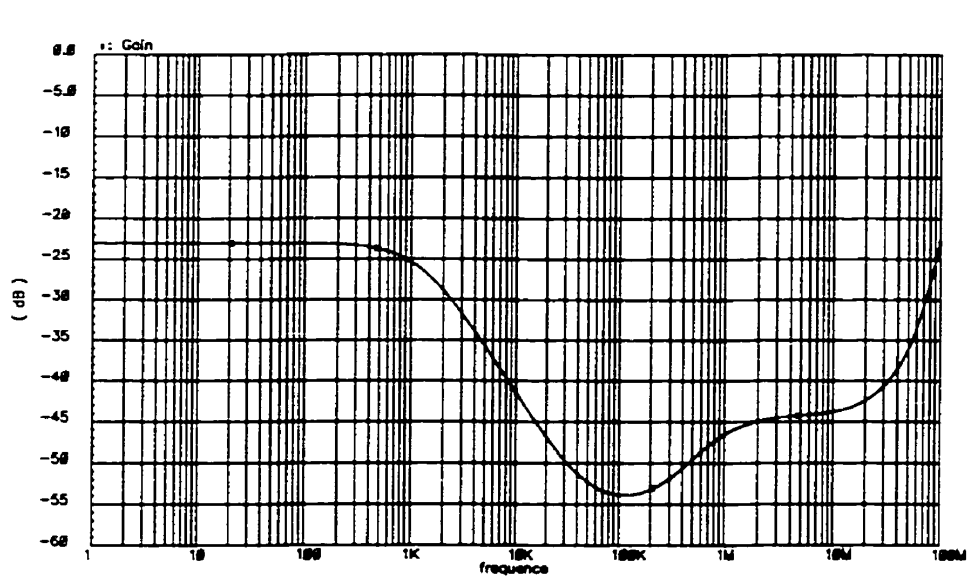


Figure C2: Gain en mode commun de l'amplificateur opérationnel

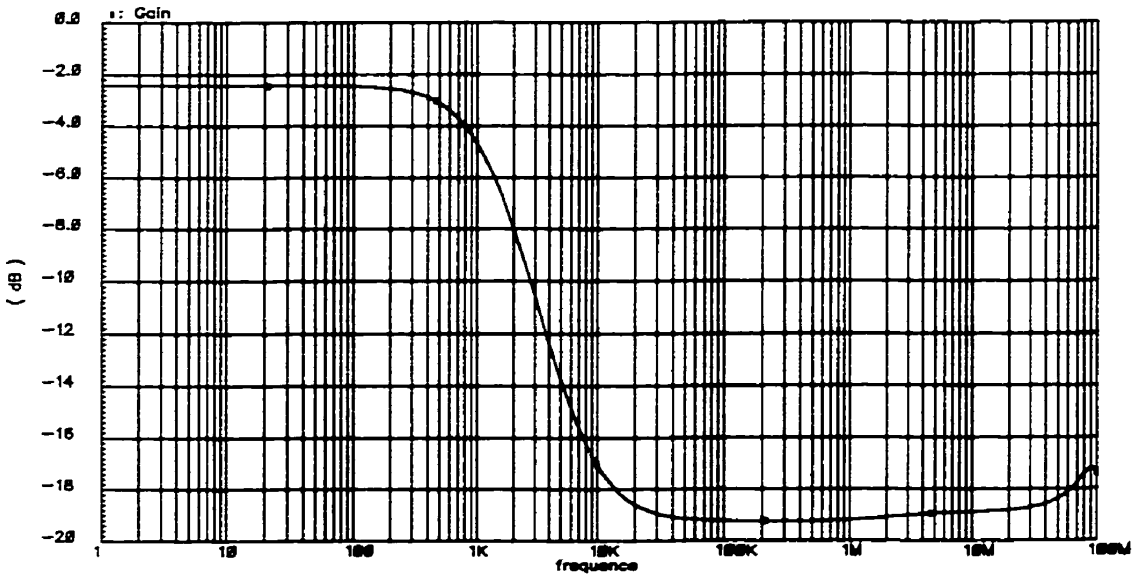


Figure C.3: Gain Vsortie/Valim pour mesure du PSRR du OPAMP

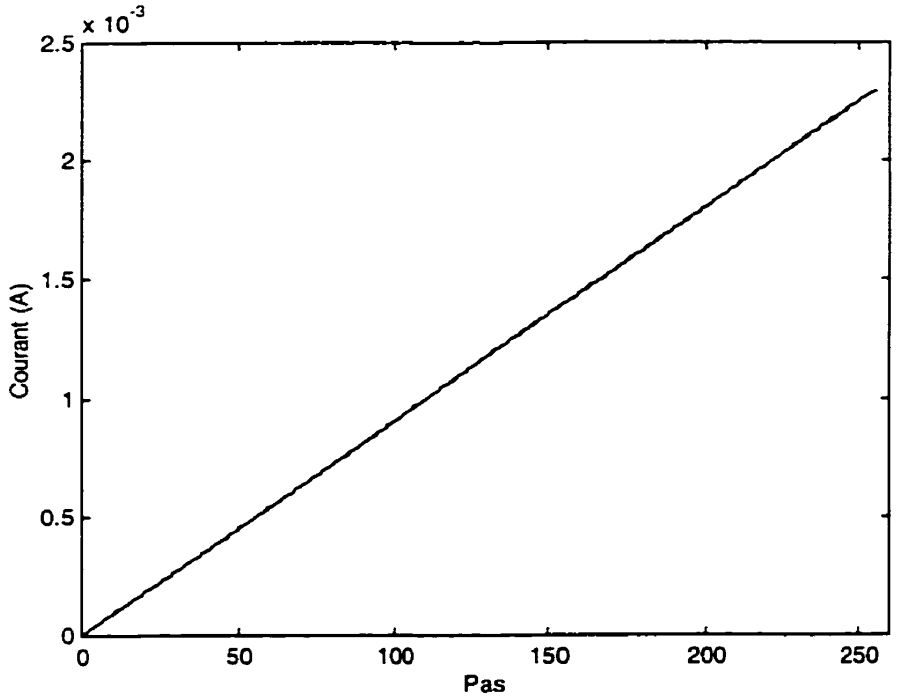


Figure C.4 : Réponses théorique et simulée du CNA

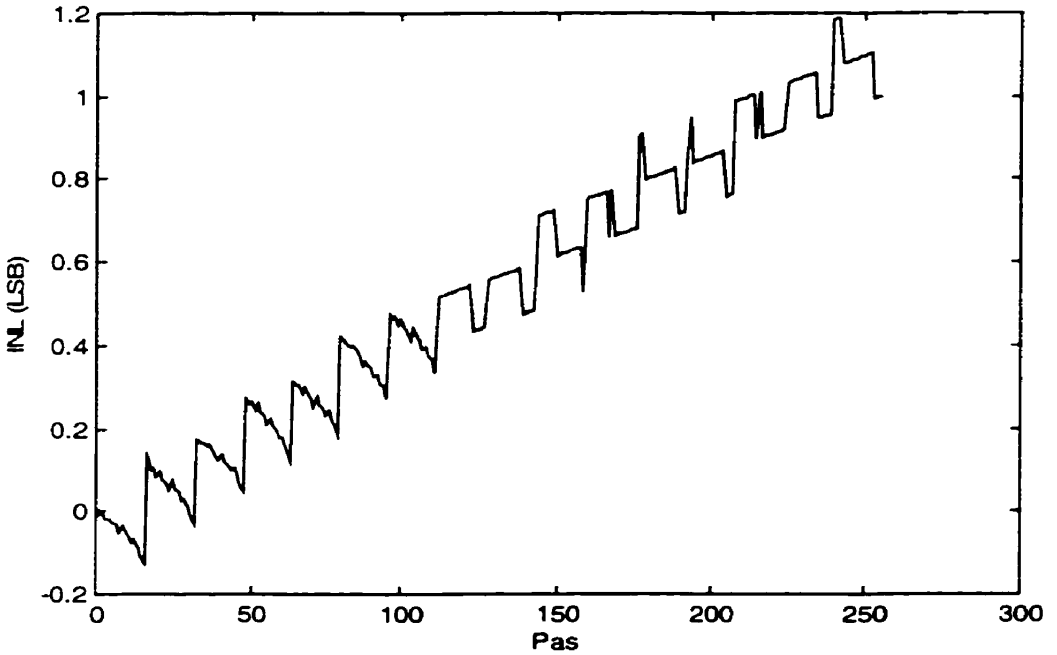


Figure C.5 : Non-linéarité intégrale (INL) du CNA

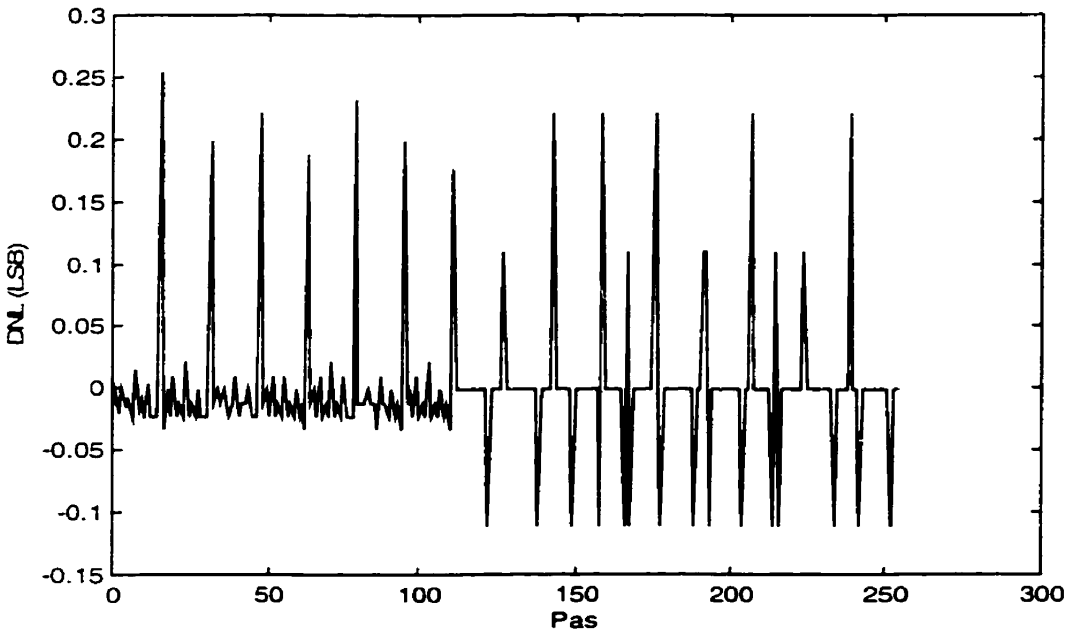


Figure C6 : Non-linéarité différentielle (DNL) du CNA

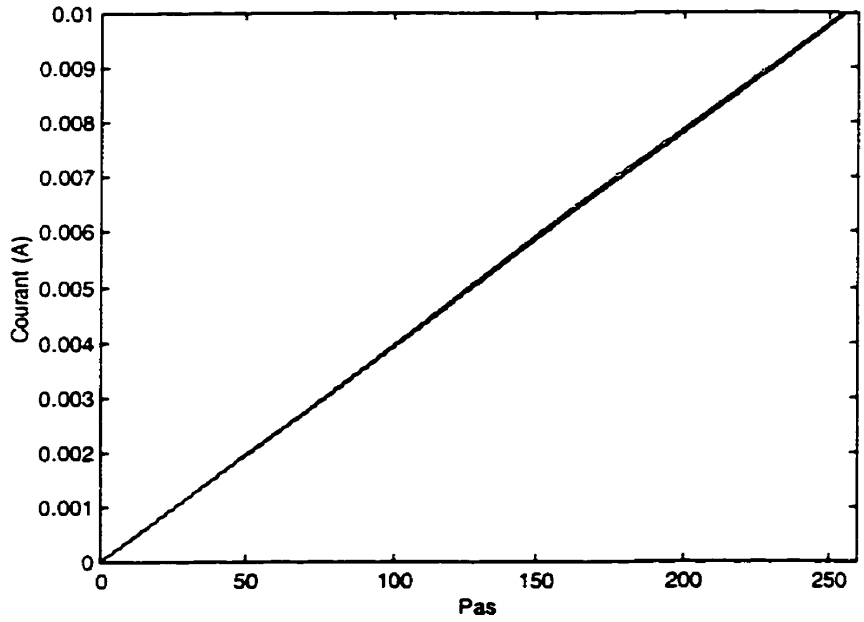


Figure C.7 : Réponse théorique (-) et simulée (..) du CNA avec miroir de courant

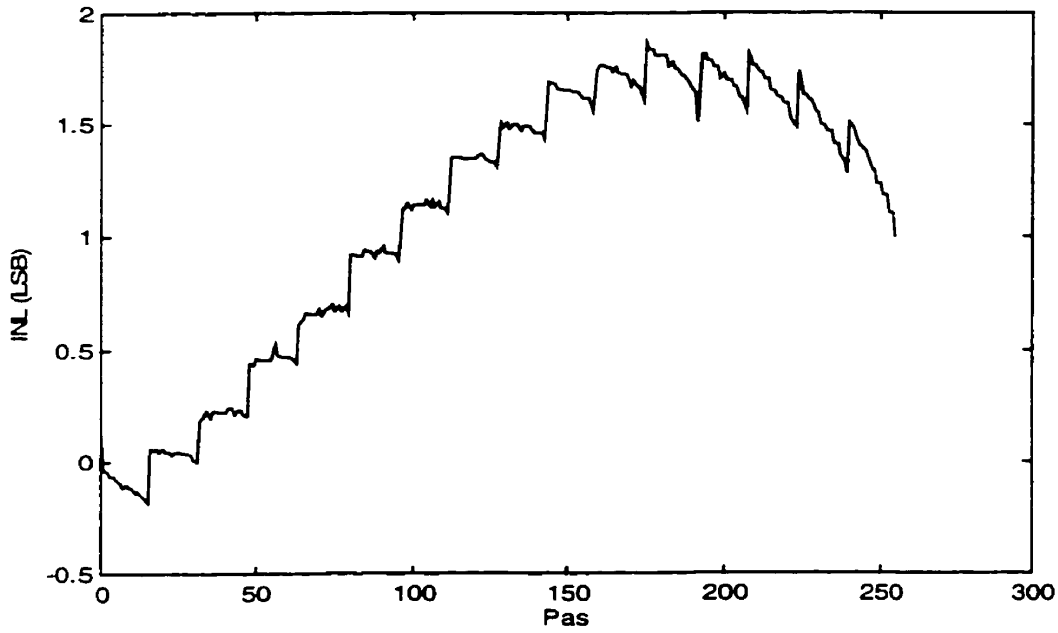


Figure C.8 : Non-linéarité intégrale du CNA avec miroir de courant

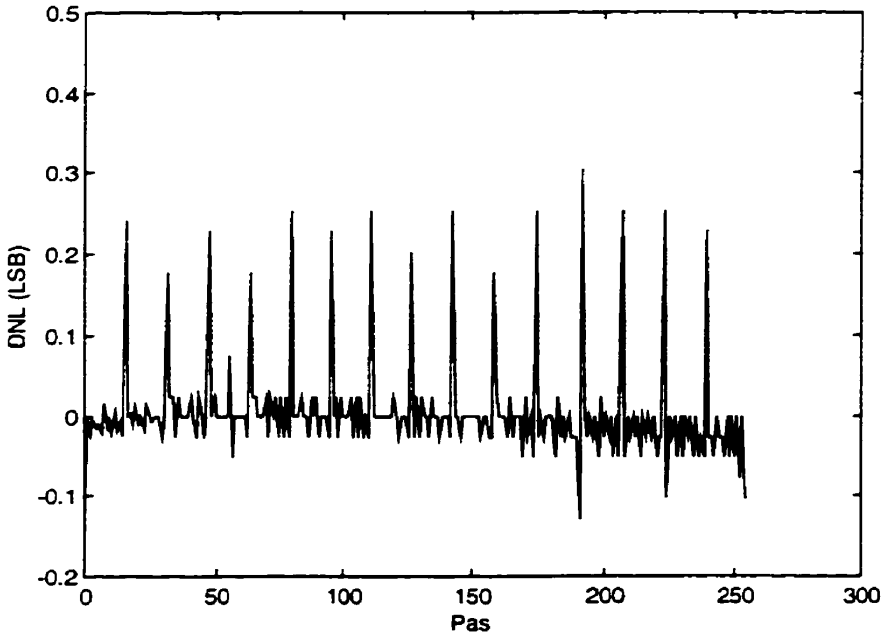


Figure C.9: Non-linéarité différentielle du CNA et du miroir de courant

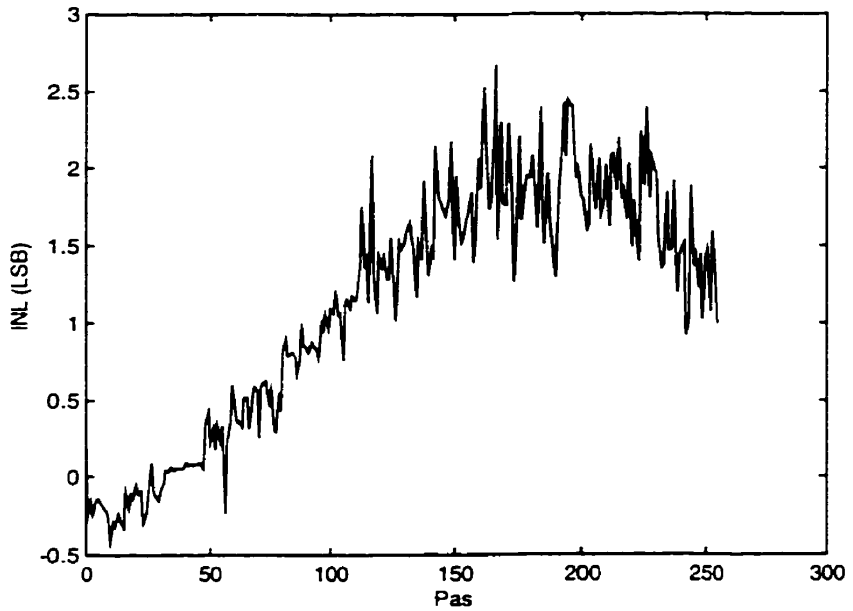


Figure C.10 : INL du CEB au début du mode maintient

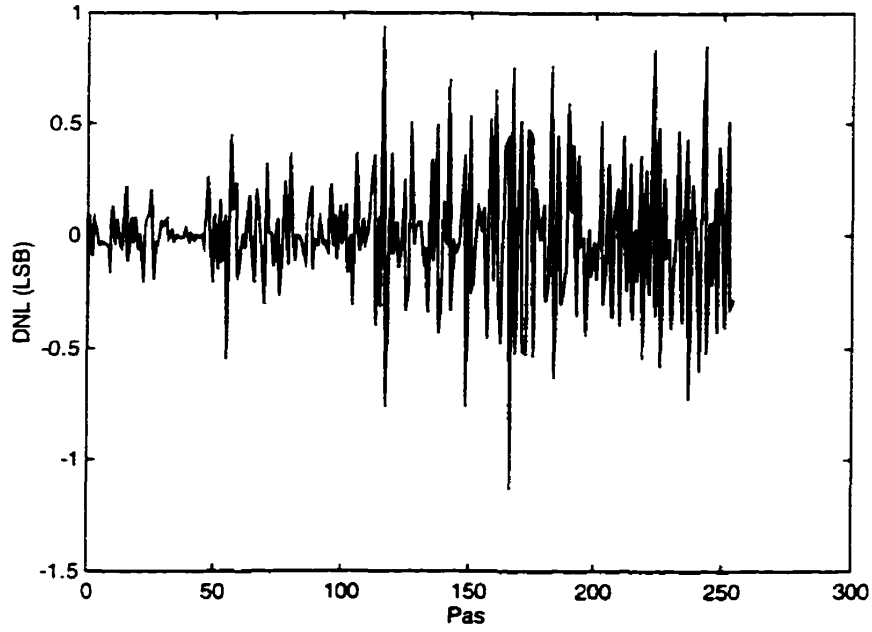


Figure C.11 : DNL du CEB au début du mode maintient

ANNEXE D

DESCRIPTION VHDL DU CONTRÔLEUR DU PIVC

D.1 Compilation de la description VHDL du PIVC

La compilation du code VHDL nécessaire à la configuration du FPLD est réalisée à l'aide du logiciel MAXPLUS II de Altera. Le tableau C.1 liste tous les fichiers VHDL contenant la description du contrôleur. Ces fichiers doivent tous être placés dans le même répertoire et ensuite compilés séparément dans l'ordre dans lequel ils apparaissent dans le tableau.

Tableau D.1 : Liste des fichiers de la description VHDL du contrôleur du PIVC

Ordre comp.	Fichier	Ordre comp.	Fichier
1	Pivc_def.vhd	9	Gen.vhd
2	Manch.vhd	10	Horlstim.vhd
3	Trame.vhd	11	Cpivc.vhd
4	Hamming.vhd	12	Ampfifo.vhd
5	Cmma.vhd	13	Canfifo.vhd
6	Ccfg.vhd	14	Ccna.vhd
7	Ramacs.vhd	15	Canal.vhd
8	Acs.vhd	16	Pivc.vhd

D.2 Description VHDL du PIVC

PIVC_DEF.VHD

```
LIBRARY IEEE;
  USE IEEE.std_logic_1164.all;
  USE IEEE.std_logic_arith.all;
```

```
PACKAGE pivc_def IS
-----------------------------------------------------------------------
-- Architecture générale
-----------------------------------------------------------------------
CONSTANT  NCANAL    :integer       :=16;
-----------------------------------------------------------------------
-- Formats des donnees
-----------------------------------------------------------------------
CONSTANT  NB_MODE   :integer    :=1;
CONSTANT  NB_INSTR  :integer    :=3;
CONSTANT  NB_CFG    :integer    :=22;
CONSTANT  NB_PMT    :integer    :=25;
CONSTANT  NB_PMTCFG :integer    :=25;
CONSTANT  NB_MCMD   :integer    :=NB_MODE+NB_PMTCFG;
CONSTANT  NB_PARITE :integer    :=5;
CONSTANT  NB_TRAME  :integer    :=NB_MCMD+NB_PARITE;
CONSTANT  NB_SITE   :integer    :=4;
CONSTANT  NB_CANAL  :integer    :=4;
CONSTANT  NB_RC1    :integer    :=22;
CONSTANT  NB_RC2    :integer    :=16;
CONSTANT  NB_RC3    :integer    :=16;
CONSTANT  NB_RACS   :integer    :=8;
CONSTANT  NB_LCMSP  :integer    :=9;
CONSTANT  NB_LMSP   :integer    :=6;
CONSTANT  NB_NBA    :integer    :=3;
CONSTANT  NB_NBDP   :integer    :=3;
CONSTANT  NB_NBDI   :integer    :=3;
CONSTANT  NB_AMP    :integer    :=8;
CONSTANT  NB_DP     :integer    :=8;
CONSTANT  NB_DI     :integer    :=8;
CONSTANT  NB_BTHS   :integer    :=8;
-----------------------------------------------------------------------
-- Horloge de stimulation
-----------------------------------------------------------------------
CONSTANT  NB_HORLS    :integer    :=8;
CONSTANT  ZERO_COMPT  :unsigned   :="00000000";
CONSTANT  UN_COMPT    :unsigned   :="00000001";
-----------------------------------------------------------------------
-- Index des champs d'un mot de configuration
-----------------------------------------------------------------------
CONSTANT  ID_MCFG_MODE        :integer    :=0;
CONSTANT  ID_MCFG_INSTR_00    :integer    :=1;
CONSTANT  ID_MCFG_INSTR_02    :integer    :=3;
CONSTANT  ID_MCFG_CFG_00      :integer    :=4;
CONSTANT  ID_MCFG_CFG_21      :integer    :=25;
-----------------------------------------------------------------------
-- Index des champs d'un mot de stimulation
-----------------------------------------------------------------------
CONSTANT  ID_MSTIM_MODE    :integer    :=0;
CONSTANT  ID_MSTIM_PMT_00  :integer    :=1;
CONSTANT  ID_MSTIM_PMT_24  :integer    :=25;
-----------------------------------------------------------------------
-- Mode du PIVC
-----------------------------------------------------------------------
CONSTANT  MODE_CONFIGURATION :std_logic     :='0';
CONSTANT  MODE_STIMULATION   :std_logic     :='1';
-----------------------------------------------------------------------
```

```
-- Instructions de configuration du PIVC
-----------------------------------------------------------------------
CONSTANT  INSTR_DEMAR_ARRET   :std_logic_vector   :="000";
CONSTANT  INSTR_RC1           :std_logic_vector   :="001";
CONSTANT  INSTR_RC2           :std_logic_vector   :="010";
CONSTANT  INSTR_RC3           :std_logic_vector   :="011";
CONSTANT  INSTR_ACS           :std_logic_vector   :="100";
CONSTANT  INSTR_SYNC          :std_logic_vector   :="101";
-----------------------------------------------------------------------
-- Codes de demarrage et d'arret
-----------------------------------------------------------------------
CONSTANT  CODE_ARRET  :std_logic_vector  :="00000000000000000000";
CONSTANT  CODE_DEMAR  :std_logic_vector  :="11111111111111111111";
-----------------------------------------------------------------------
-- Paramètres de la RAM ACS
-----------------------------------------------------------------------
CONSTANT  NB_DON_RAMACS       :integer       :=8;
CONSTANT  NB_ADR_RAMACS       :integer       :=8;
CONSTANT  VAL_MAX_ADR_RAMACS  :unsigned      :="11111111";
CONSTANT  ID_RAMACS_SITE_00   :integer       :=0;
CONSTANT  ID_RAMACS_SITE_03   :integer       :=3;
CONSTANT  ID_RAMACS_CANAL_04  :integer       :=4;
CONSTANT  ID_RAMACS_CANAL_07  :integer       :=7;
-----------------------------------------------------------------------
-- Index du registre de configuration RC1
-----------------------------------------------------------------------
CONSTANT  ID_FA          :integer       :=0;
CONSTANT  ID_NBA_00      :integer       :=1;
CONSTANT  ID_NBA_02      :integer       :=3;
CONSTANT  ID_FDP         :integer       :=4;
CONSTANT  ID_NBDP_00     :integer       :=5;
CONSTANT  ID_NBDP_02     :integer       :=7;
CONSTANT  ID_FDI         :integer       :=8;
CONSTANT  ID_NBDI_00     :integer       :=9;
CONSTANT  ID_NBDI_02     :integer       :=11;
CONSTANT  ID_FCS         :integer       :=12;
CONSTANT  ID_LCMSP_00    :integer       :=13;
CONSTANT  ID_LCMSP_08    :integer       :=21;
-----------------------------------------------------------------------
-- Index du registre de configuration RC2
-----------------------------------------------------------------------
CONSTANT  ID_AMPC_00     :integer       :=0;
CONSTANT  ID_AMPC_07     :integer       :=7;
CONSTANT  ID_DPC_00      :integer       :=8;
CONSTANT  ID_DPC_07      :integer       :=15;
-----------------------------------------------------------------------
-- Index du registre de configuration RC3
-----------------------------------------------------------------------
CONSTANT  ID_DIC_00      :integer       :=0;
CONSTANT  ID_DIC_07      :integer       :=7;
CONSTANT  ID_BTHS_00     :integer       :=8;
CONSTANT  ID_BTHS_07     :integer       :=15;
-----------------------------------------------------------------------
-- Index d'un mot de synchronisation
-----------------------------------------------------------------------
CONSTANT  ID_SYNC_HS       :integer       :=0;
CONSTANT  ID_SYNC_RAZADR   :integer       :=1;
```

```
-----------------------------------------------------------------
---
-- Séquenceur
-----------------------------------------------------------------
CONSTANT  VAL_MAX_COMPTEUR_BITS :unsigned   :="11010";
CONSTANT  VAL_OFF_COMPTEUR_BITS :unsigned   :="11001";
CONSTANT  ZERO_6BITS            :unsigned   :="000000";
CONSTANT  VAL_MAX_NB_CANAL      :unsigned   :="11";
CONSTANT  VAL_MAX_NB_SITE       :unsigned   :="11";
CONSTANT  BITS_ZERO             :unsigned   :="000000";
CONSTANT  MSP_ZERO              :unsigned   :="000000";
CONSTANT  CH_ZERO               :unsigned   :="000000000";
CONSTANT  VAL_NB_MCMD           :unsigned   :="11001";
CONSTANT  NBA_ZERO              :unsigned   :="000";
CONSTANT  NBDP_ZERO             :unsigned   :="000";
CONSTANT  NBDI_ZERO             :unsigned   :="000";
-----------------------------------------------------------------
-- FIFO
-----------------------------------------------------------------
CONSTANT  NB_COMPT_ELEMENT    :integer      :=6;
CONSTANT  N_ELEMENT           :integer      :=32;
-----------------------------------------------------------------
-- Canal de stimulation
-----------------------------------------------------------------
CONSTANT  ZERO_DELAI  :std_logic_vector  :="00000000";
CONSTANT  UN_DELAI    :std_logic_vector  :="00000001";
CONSTANT  canal_ident :std_logic_vector
   :="1111111011011100101110101001100001110110010101000011001000010000"
;
END PACKAGE pivc_def;
```

MANCH.VHD

```
LIBRARY IEEE;
  USE IEEE.std_logic_1164.all;
  USE IEEE.std_logic_arith.all;
  USE work.pivc_def.all;

ENTITY manch IS
  PORT( tx_mode       : IN  std_logic;
    horl_direct       : IN  std_logic;
    donnees_direct    : IN  std_logic;
    signal_manch      : IN  std_logic;
    raz_manch         : IN  std_logic;
    horl_manch        : OUT std_logic;
    donnees_manch     : OUT std_logic;
    horl            : OUT std_logic;
    donnees         : OUT std_logic;
    pt_q1           : OUT std_logic;
    pt_q2           : OUT std_logic
    );
END manch;

ARCHITECTURE rtl OF manch IS

SIGNAL q1                : std_logic;
```

```
SIGNAL q2                 : std_logic;
SIGNAL horl_manch_i       : std_logic;
SIGNAL donnees_manch_i    : std_logic;

BEGIN
-----------------------------------------------------------------------
-- Mode
-----------------------------------------------------------------------
   tx_mode_proc:PROCESS(tx_mode)
   BEGIN
      CASE tx_mode IS
         WHEN '1' =>
            horl    <= horl_direct;
            donnees<= donnees_direct;
         WHEN '0' =>
            horl    <= horl_manch_i;
            donnees<= donnees_manch_i;
         WHEN OTHERS =>
            horl    <= horl_manch_i;
            donnees<= donnees_manch_i;
      END CASE;
   END PROCESS tx_mode_proc;
-----------------------------------------------------------------------
-- Bascule D1
-----------------------------------------------------------------------
basc_d1:PROCESS(signal_manch,raz_manch)
   BEGIN
      IF (raz_manch = '0') THEN
         q1 <= '0';
      ELSIF (signal_manch'EVENT AND signal_manch = '0') THEN
         q1 <= '1';
      END IF;
   END PROCESS basc_d1;
-----------------------------------------------------------------------
-- Bascule D2
-----------------------------------------------------------------------
basc_d2:PROCESS(signal_manch,raz_manch)
   BEGIN
      IF (raz_manch = '0') THEN
         q2 <= '0';
      ELSIF (signal_manch'EVENT AND signal_manch = '1') THEN
         q2 <= '1';
      END IF;
   END PROCESS basc_d2;
-----------------------------------------------------------------------
-- Bascule D3
-----------------------------------------------------------------------
basc_d3:PROCESS(horl_manch_i)
   BEGIN
      IF (horl_manch_i'EVENT AND horl_manch_i = '1') THEN
         donnees_manch_i  <= signal_manch;
      END IF;
   END PROCESS basc_d3;
-----------------------------------------------------------------------
-- Porte NON OU
-----------------------------------------------------------------------
non_ou:PROCESS(q1,q2)
```

```
  BEGIN
    IF (q1 = '0' AND q2 = '0') THEN
      horl_manch_i   <= '1';
    ELSE
      horl_manch_i   <= '0';
    END IF;
  END PROCESS non_ou;
-----------------------------------------------------------------------
-- Horl_manch et points de mesure Q1 cet Q2
-----------------------------------------------------------------------
 hmanch:PROCESS(horl_manch_i,q1,q2)
  BEGIN
    donnees_manch <= donnees_manch_i;
    horl_manch  <= horl_manch_i;
    pt_q1     <= q1;
    pt_q2     <= q2;
  END PROCESS hmanch;
END rtl;
```

TRAME.VHD

```
LIBRARY IEEE;
  USE IEEE.std_logic_1164.all;
  USE IEEE.std_logic_arith.all;
  USE work.pivc_def.all;

ENTITY trame IS
  PORT(horl : IN  std_logic;
    dcts    : IN  std_logic;
    trmdtc  : OUT std_logic;
    trame   : OUT std_logic_vector(NB_TRAME-1 DOWNTO 0)
      );
END trame;

ARCHITECTURE rtl OF trame IS

SIGNAL bit_arret    : std_logic;
SIGNAL bit_depart   : std_logic;
SIGNAL trame_int    : std_logic_vector(NB_TRAME-1 DOWNTO 0);
SIGNAL trmdtc_int   : std_logic;
SIGNAL trm_rejetee  : std_logic;

BEGIN
-----------------------------------------------------------------------
--   Registre à décalage à l'entrée du PIVC.
--   Remise à 1 de tous les bits du registre et du bit de départ
--   lorsqu'une trame fût détectée ou rejetee au cycle d'horloge
--   précédent.
-----------------------------------------------------------------------
  registre_decalage:PROCESS(horl)
  BEGIN
    IF (horl'EVENT AND horl = '1') THEN
      IF (trmdtc_int = '1' OR trm_rejetee = '1') THEN
        trame_int <= (OTHERS => '1');
        bit_depart  <= '1';
```

```
        ELSE
          bit_depart  <= trame_int(0);
          FOR i IN 1 TO (NB_TRAME-1) LOOP
            trame_int(i-1) <= trame_int(i);
          END LOOP;
          trame_int(NB_TRAME-1) <= bit_arret;
        END IF;
        IF (trm_rejetee = '1') THEN
          bit_arret   <= '1';
        ELSE
          bit_arret   <= dcts;
        END IF;
      END IF;
    END PROCESS registre_decalage;
-----------------------------------------------------------------------
--   Détection d'une trame
-----------------------------------------------------------------------
    detection_trame:PROCESS(horl)
    BEGIN
      IF (horl'EVENT AND horl = '1') THEN
        IF (trame_int(0) = '0'
                          AND dcts = '1' AND trmdtc_int = '0'
          AND trm_rejetee = '0') THEN
          trmdtc_int       <= '1';
          trm_rejetee      <= '0';
        ELSIF(trame_int(0) = '0' AND dcts = '0'
                          AND trmdtc_int= '0'
            AND trm_rejetee = '0') THEN
          trmdtc_int       <= '0';
          trm_rejetee      <= '1';
        ELSE
          trmdtc_int       <= '0';
          trm_rejetee      <= '0';
        END IF;
      END IF;
    END PROCESS detection_trame;
-----------------------------------------------------------------------
--  Signaux et données de sortie
-----------------------------------------------------------------------
    PROCESS(trame_int,trmdtc_int)
    BEGIN
      trame     <= trame_int;
      trmdtc <= trmdtc_int;
    END PROCESS;
END rtl;
```

HAMMING .VHD

```
LIBRARY IEEE;
  USE IEEE.std_logic_1164.all;
  USE IEEE.std_logic_arith.all;
  USE work.pivc_def.all;

ENTITY hamming IS
  PORT( horl   : IN  std_logic;
    trmdtc    : IN  std_logic;
```

```
     trame      : IN  std_logic_vector(NB_TRAME-1 DOWNTO 0);
     mcmd       : OUT std_logic_vector(NB_MCMD-1 DOWNTO 0);
     mcv        : OUT std_logic
     );
END hamming;

ARCHITECTURE rtl OF hamming IS

SIGNAL parite : unsigned(NB_PARITE DOWNTO 0);
SIGNAL erreur : std_logic_vector(NB_TRAME-1 DOWNTO 0);

BEGIN
-----------------------------------------------------------------------
-- Calcul des bits de parités
-----------------------------------------------------------------------
calcul_parite: PROCESS(trame)
 BEGIN
  parite(0) <= trame(0) XOR trame(2) XOR trame(4) XOR trame(6)
            XOR  trame(8) XOR trame(10) XOR trame(12) XOR trame(14)
            XOR  trame(16) XOR trame(18) XOR trame(20) XOR trame(22)
            XOR trame(24) XOR trame(26) XOR trame(28) XOR trame(30);
  parite(1) <= trame(1) XOR trame(2) XOR trame(5)XOR trame(6)
            XOR trame(9) XOR trame(10) XOR trame(13) XOR trame(14)
            XOR trame(17) XOR trame(18) XOR trame(21) XOR trame(22)
            XOR trame(25) XOR trame(26) XOR trame(29) XOR trame(30);
  parite(2) <= trame(3) XOR trame(4) XOR trame(5) XOR trame(6)
            XOR trame(11) XOR trame(12) XOR trame(13) XOR trame(14)
            XOR trame(19) XOR trame(20) XOR trame(21) XOR trame(22)
            XOR trame(27) XOR trame(28) XOR trame(29) XOR trame(30);
  parite(3) <= trame(7) XOR trame(8) XOR trame(9) XOR trame(10)
            XOR trame(11) XOR trame(12) XOR trame(13) XOR trame(14)
            XOR trame(23) XOR trame(24) XOR trame(25) XOR trame(26)
            XOR trame(27) XOR trame(28) XOR trame(29) XOR trame(30);
  parite(4) <= trame(15) XOR trame(16) XOR trame(17) XOR trame(18)
            XOR trame(19) XOR trame(20) XOR trame(21) XOR trame(22)
            XOR trame(23) XOR trame(24) XOR trame(25) XOR trame(26)
            XOR trame(27) XOR trame(28) XOR trame(29) XOR trame(30);
parite(5) <= '0';
  END PROCESS calcul_parite;
-----------------------------------------------------------------------
-- Détection d'une éventuelle erreur de transmission
-----------------------------------------------------------------------
  detection_erreur: PROCESS(parite)
  BEGIN
  FOR i IN 1 TO NB_TRAME LOOP
    IF (i = parite) THEN
      erreur(i-1) <= '1';
    ELSE
      erreur(i-1) <= '0';
    END IF;
  END LOOP;
  END PROCESS detection_erreur;
-----------------------------------------------------------------------
-- Formation du mot de commande et correction d'une éventuelle erreur
-- de transmission
-----------------------------------------------------------------------
  correcteur:PROCESS(horl)
```

```
    BEGIN
      IF (horl'EVENT AND horl = '1') THEN
        IF (trmdtc = '1') THEN
          mcmd(0) <= trame(2) XOR erreur(2);
          mcmd(1) <= trame(4) XOR erreur(4);
          mcmd(2) <= trame(5) XOR erreur(5);
          mcmd(3) <= trame(6) XOR erreur(6);
          mcmd(4) <= trame(8) XOR erreur(8);
          mcmd(5) <= trame(9) XOR erreur(9);
          mcmd(6) <= trame(10) XOR erreur(10);
          mcmd(7) <= trame(11) XOR erreur(11);
          mcmd(8) <= trame(12) XOR erreur(12);
          mcmd(9) <= trame(13) XOR erreur(13);
          mcmd(10) <= trame(14) XOR erreur(14);
          mcmd(11) <= trame(16) XOR erreur(16);
          mcmd(12) <= trame(17) XOR erreur(17);
          mcmd(13) <= trame(18) XOR erreur(18);
          mcmd(14) <= trame(19) XOR erreur(19);
          mcmd(15) <= trame(20) XOR erreur(20);
          mcmd(16) <= trame(21) XOR erreur(21);
          mcmd(17) <= trame(22) XOR erreur(22);
          mcmd(18) <= trame(23) XOR erreur(23);
          mcmd(19) <= trame(24) XOR erreur(24);
          mcmd(20) <= trame(25) XOR erreur(25);
          mcmd(21) <= trame(26) XOR erreur(26);
          mcmd(22) <= trame(27) XOR erreur(27);
          mcmd(23) <= trame(28) XOR erreur(28);
          mcmd(24) <= trame(29) XOR erreur(29);
          mcmd(25) <= trame(30) XOR erreur(30);
          mcv <= '1';
        ELSE
          mcv <= '0';
        END IF;
      END IF;
    END PROCESS correcteur;
END rtl;
```

CMMA.VHD

```
LIBRARY IEEE;
  USE IEEE.std_logic_1164.all;
  USE IEEE.std_logic_arith.all;
  USE work.pivc_def.all;

ENTITY cmma IS
  PORT( horl     : IN  std_logic;
    mcv          : IN  std_logic;
    mode         : IN  std_logic;
    instr        : IN  std_logic_vector(NB_INSTR-1 DOWNTO 0);
    cfg          : IN  std_logic_vector(NB_CFG-1 DOWNTO 0);
    razg         : OUT std_logic
    );
END cmma;

ARCHITECTURE rtl OF cmma IS
```

```
BEGIN
-----------------------------------------------------------------------
-- Détecteur des codes de démarrage et d'arrêt du PIVC.
-----------------------------------------------------------------------
   detecteur:PROCESS(horl)
   BEGIN
      IF (horl'EVENT AND horl = '1') THEN
         IF (mode = MODE_CONFIGURATION AND
            instr = INSTR_DEMAR_ARRET AND
            mcv = '1') THEN
            IF (cfg = CODE_DEMAR) THEN
               razg <= '1';
            ELSIF(cfg = CODE_ARRET) THEN
               razg <= '0';
            END IF;
         END IF;
      END IF;
   END PROCESS detecteur;
END rtl;
```

CCFG.VHD

```
LIBRARY IEEE;
   USE IEEE.std_logic_1164.all;
   USE IEEE.std_logic_arith.all;
   USE work.pivc_def.all;

ENTITY ccfg IS
    PORT(  razg     :IN std_logic;
      horl          :IN std_logic;
   mode             :IN std_logic;
   instr            :IN std_logic_vector(NB_INSTR-1 DOWNTO 0);
   cfg              :IN std_logic_vector(NB_CFG-1 DOWNTO 0);
   mcv              :IN std_logic;
   rc1              :OUT std_logic_vector(NB_RC1-1 DOWNTO 0);
   rc2              :OUT std_logic_vector(NB_RC2-1 DOWNTO 0);
   rc3              :OUT std_logic_vector(NB_RC3-1 DOWNTO 0);
   sync             :OUT std_logic;
   ram_canal        :OUT std_logic_vector(NB_CANAL-1 DOWNTO 0);
   ram_site         :OUT std_logic_vector(NB_SITE-1 DOWNTO 0);
   moderam          :OUT std_logic;
   ecrire           :OUT std_logic;
   raz_adresse      :OUT std_logic
      );
END ccfg;

ARCHITECTURE rtl OF ccfg IS

TYPE type_etat_ramacs IS(LECTURE,ECRITURE,RAZADR);
SIGNAL etat_ramacs    :type_etat_ramacs;
SIGNAL racs           :std_logic_vector(NB_RACS-1 DOWNTO 0);

BEGIN
-----------------------------------------------------------------------
-- Décodeur d'instruction
-----------------------------------------------------------------------
```

```
proc_instr:PROCESS(horl,razg)
  BEGIN
    IF(razg = '0') THEN
      rc1          <= (OTHERS => '0');
      rc2          <= (OTHERS => '0');
      rc3          <= (OTHERS => '0');
      etat_ramacs <= LECTURE;
      moderam      <= '0';
    ELSIF (horl'EVENT AND horl = '1') THEN
      IF(mcv = '1' AND mode = MODE_STIMULATION) THEN
        etat_ramacs <= LECTURE;
        moderam   <= '0';
      ELSIF (mcv = '1' AND mode = MODE_CONFIGURATION) THEN
        CASE instr IS
          WHEN  INSTR_RC1  =>
            rc1     <= cfg;
            etat_ramacs <= RAZADR;
            moderam   <= '0';
          WHEN  INSTR_RC2  =>
            rc2  <= cfg(NB_RC2-1 DOWNTO 0);
            etat_ramacs <= RAZADR;
            moderam   <= '0';
          WHEN  INSTR_RC3   =>
            rc3  <= cfg(NB_RC3-1 DOWNTO 0);
            etat_ramacs <= RAZADR;
            moderam   <= '0';
          WHEN  INSTR_ACS  =>
            racs    <= cfg(NB_RACS-1 DOWNTO 0);
            etat_ramacs <= ECRITURE;
            moderam<= '1';
          WHEN  INSTR_SYNC =>
            IF (cfg(ID_SYNC_HS) = '1') THEN
              sync     <= '1';
            END IF;
            IF (cfg(ID_SYNC_RAZADR) = '1') THEN
              etat_ramacs <= RAZADR;
            END IF;
            moderam   <= '0';
          WHEN OTHERS =>
            etat_ramacs <= LECTURE;
            moderam   <= '0';
        END CASE;
      ELSE
        etat_ramacs <= LECTURE;
        sync      <= '0';
      END IF;
    END IF;
  END PROCESS proc_instr;
-----------------------------------------------------------------------
-- Contrôleur d'écriture dans la RAMACS
-----------------------------------------------------------------------
ram:PROCESS(horl,razg)
  BEGIN
    IF(razg = '0') THEN
      ecrire    <= '0';
      raz_adresse <= '0';
    ELSIF (horl'EVENT AND horl = '1') THEN
```

```
        CASE etat_ramacs IS
          WHEN LECTURE  =>
            ecrire    <= '0';
            raz_adresse <= '1';
          WHEN ECRITURE =>
            ecrire    <= '1';
            raz_adresse <= '1';
            ram_site <= racs(NB_SITE-1 DOWNTO 0);
            ram_canal <= racs(NB_CANAL+NB_SITE-1
                        DOWNTO NB_SITE);
          WHEN RAZADR   =>
            ecrire    <= '0';
            raz_adresse <= '0';
          WHEN OTHERS =>
            ecrire    <= '0';
            raz_adresse <= '1';
        END CASE;
      END IF;
    END PROCESS ram;
END rtl;
```

RAMACS.VHD

```
LIBRARY IEEE;
  USE IEEE.std_logic_1164.all;
  USE IEEE.std_logic_arith.all;
  USE work.pivc_def.all;
LIBRARY lpm;
USE lpm.lpm_components.ALL;

ENTITY ramacs IS
   PORT(dent:  IN STD_LOGIC_VECTOR (NB_DON_RAMACS-1 DOWNTO 0);
      adresse:   IN STD_LOGIC_VECTOR (NB_ADR_RAMACS-1 DOWNTO 0);
      el:        IN STD_LOGIC;
  horl:          IN STD_LOGIC;
      dsort:     OUT STD_LOGIC_VECTOR (NB_DON_RAMACS- 1 DOWNTO 0));
END ramacs;

ARCHITECTURE rtl OF ramacs IS

BEGIN
  inst_1: lpm_ram_dq
        GENERIC MAP(lpm_widthad    => NB_ADR_RAMACS,
          lpm_address_control      => "UNREGISTERED",
          lpm_outdata              => "UNREGISTERED",
          lpm_width                => NB_DON_RAMACS)
        PORT MAP   (data           => dent,
          address                  => adresse,
          we                       => el,
          inclock                  => horl,
          q                        => dsort
          );
END rtl;
```

ACS.VHD

```
LIBRARY IEEE;
   USE IEEE.std_logic_1164.all;
   USE IEEE.std_logic_arith.all;
   USE work.pivc_def.all;

ENTITY acs IS
    PORT(canal_ent:     IN std_logic_vector(NB_CANAL-1 DOWNTO 0);
   site_ent:            IN std_logic_vector(NB_SITE-1 DOWNTO 0);
   moderam:             IN std_logic;
   horl:                IN std_logic;
  ecrire:               IN std_logic;
   lire:                IN std_logic;
   raz_adresse:         IN std_logic;
   razg:                IN std_logic;
   canal_sort:          OUT std_logic_vector(NB_CANAL-1 DOWNTO 0);
   site_sort:           OUT std_logic_vector(NB_SITE-1 DOWNTO 0);
   adresse_ramacs:      OUT std_logic_vector(NB_ADR_RAMACS-1 DOWNTO 0)
   );
END acs;

ARCHITECTURE rtl OF acs IS

-----------------------------------------------------------------------
-- Declaration de La RAM ACS
-----------------------------------------------------------------------
COMPONENT ramacs
  PORT(dent       :IN std_logic_vector(NB_DON_RAMACS-1 DOWNTO 0);
       adresse    :IN std_logic_vector(NB_ADR_RAMACS-1 DOWNTO 0);
       el         :IN std_logic;
       horl       :IN std_logic;
       dsort      :OUT std_logic_vector(NB_DON_RAMACS- 1 DOWNTO 0)
      );
  END COMPONENT;

-----------------------------------------------------------------------
-- Declaration des signaux
-----------------------------------------------------------------------
SIGNAL adresse   :std_logic_vector(NB_ADR_RAMACS-1 DOWNTO 0);
SIGNAL flect     :std_logic;
SIGNAL fecr      :std_logic;
SIGNAL horl_prec :std_logic;
SIGNAL dent      :std_logic_vector(NB_DON_RAMACS-1 DOWNTO 0);
SIGNAL dsort     :std_logic_vector(NB_DON_RAMACS-1 DOWNTO 0);

BEGIN
-----------------------------------------------------------------------
-- Connexions de la RAM ACS
-----------------------------------------------------------------------
ramacs_inst: ramacs
  PORT MAP(dent     => dent,
       adresse      => adresse,
       el           => ecrire,
       horl         => horl,
       dsort        => dsort
      );
-----------------------------------------------------------------------
-- Aiguillage des données d'entrée (#canal et #site)
```

```
-----------------------------------------------------------------
---
format:PROCESS(canal_ent,site_ent)
  BEGIN
    dent(ID_RAMACS_SITE_03 DOWNTO ID_RAMACS_SITE_00)
      <= site_ent(NB_SITE-1 DOWNTO 0);
    dent(ID_RAMACS_CANAL_07 DOWNTO ID_RAMACS_CANAL_04)
      <= canal_ent(NB_CANAL-1 DOWNTO 0);
  END PROCESS format;
-----------------------------------------------------------------
-- Fanion ecriture
-----------------------------------------------------------------
ecrit_proc:PROCESS(horl,razg)
  BEGIN
    IF(razg = '0') THEN
      fecr <= '0';
    ELSIF (horl'EVENT AND horl = '1') THEN
      IF (moderam = '1') THEN
        IF(ecrire = '1') THEN
          fecr <= '1';
        ELSE
          fecr <= '0';
        END IF;
      END IF;
    END IF;
  END PROCESS ecrit_proc;
-----------------------------------------------------------------
-- Fanion lecture
-----------------------------------------------------------------
ctrl_flect:PROCESS(horl,razg)
  BEGIN
    IF(razg = '0') THEN
      flect  <= '0';
    ELSIF (horl'EVENT AND horl = '1') THEN
      IF(moderam = '0') THEN
        IF(lire = '1') THEN
          flect <= '1';
        ELSE
          flect <= '0';
        END IF;
      END IF;
    END IF;
  END PROCESS ctrl_flect;
-----------------------------------------------------------------
-- Adresse active de la RAM ACS et contrôle des données de sortie
-- (# de canal et # de site)
-----------------------------------------------------------------
proch_adr:PROCESS(horl,razg)
  BEGIN
    IF(razg = '0') THEN
      adresse<= (OTHERS => '0');
      site_sort <= (OTHERS => '0');
      canal_sort  <= (OTHERS => '0');
    ELSIF (horl'EVENT AND horl = '1') THEN
      IF (raz_adresse = '0') THEN
        adresse <= (OTHERS => '0');
      ELSIF (flect = '1' OR (fecr = '1'
```

```
                        AND moderam = '1')) THEN
                        IF (unsigned(adresse) = VAL_MAX_ADR_RAMACS)
                        THEN
            adresse <= (OTHERS => '0');
          ELSE
            adresse <= unsigned(adresse) + '1';
          END IF;
          IF (flect = '1') THEN
            site_sort(NB_SITE-1 DOWNTO 0) <= dsort
            (ID_RAMACS_SITE_03 DOWNTO
            ID_RAMACS_SITE_00);
            canal_sort(NB_CANAL-1 DOWNTO 0) <= dsort
            (ID_RAMACS_CANAL_07 DOWNTO
                              ID_RAMACS_CANAL_04);
          END IF;
        END IF;
      END IF;
   END PROCESS proch_adr;
-----------------------------------------------------------------------
-- Données de sortie
-----------------------------------------------------------------------
sortie:PROCESS(dsort,adresse)
   BEGIN
      adresse_ramacs <= adresse;
   END PROCESS sortie;
END rtl;
```

GEN.VHD

```
LIBRARY IEEE;
   USE IEEE.std_logic_1164.all;
   USE IEEE.std_logic_arith.all;
   USE work.pivc_def.all;

ENTITY gen IS
    PORT(razg    :IN std_logic;
   horl          :IN std_logic;
   mcv           :IN std_logic;
   mode          :IN std_logic;
   fa            :IN std_logic;
   fdp           :IN std_logic;
   pmt           :IN std_logic_vector(NB_PMT-1 DOWNTO 0);
   lchaine       :IN std_logic_vector(NB_LCMSP-1 DOWNTO 0);
   nba           :IN std_logic_vector(NB_NBA-1 DOWNTO 0);
   nbdp          :IN std_logic_vector(NB_NBDP-1 DOWNTO 0);
   fdi           :IN std_logic;
   nbdi          :IN std_logic_vector(NB_NBDI-1 DOWNTO 0);
   fcs           :IN std_logic;
   ampc          :IN std_logic_vector(NB_AMP-1 DOWNTO 0);
   dpc           :IN std_logic_vector(NB_DP-1 DOWNTO 0);
   dic           :IN std_logic_vector(NB_DI-1 DOWNTO 0);
   ram_site      :IN std_logic_vector(NB_SITE-1 DOWNTO 0);
   ram_canal     :IN std_logic_vector(NB_CANAL-1 DOWNTO 0);
   ram_lire      :OUT std_logic;
   bamp          :OUT std_logic_vector(NB_AMP-1 DOWNTO 0);
   bdp           :OUT std_logic_vector(NB_DP-1 DOWNTO 0);
```

```
   bdi          :OUT std_logic_vector(NB_DI-1 DOWNTO 0);
   bsite        :OUT std_logic_vector(NB_SITE-1 DOWNTO 0);
   bcanal       :OUT std_logic_vector(NB_CANAL-1 DOWNTO 0);
   bpv          :OUT std_logic
     );
END gen;

ARCHITECTURE rtl OF gen IS

-----------------------------------------------------------------------
--  Declaration des signaux
-----------------------------------------------------------------------
TYPE   type_etat_gen  IS (REPOS,FORMATION,ATTENTE);
SIGNAL etat_gen       :type_etat_gen;
SIGNAL p_etat_gen     :type_etat_gen;
SIGNAL pmtreg         :std_logic_vector(NB_PMT-1 DOWNTO 0);
SIGNAL as             :std_logic_vector(NB_AMP-1 DOWNTO 0);
SIGNAL dps            :std_logic_vector(NB_DP-1 DOWNTO 0);
SIGNAL dis            :std_logic_vector(NB_DI-1 DOWNTO 0);
SIGNAL cs             :std_logic_vector(NB_CANAL-1 DOWNTO 0);
SIGNAL ss             :std_logic_vector(NB_SITE-1 DOWNTO 0);
SIGNAL pas            :std_logic;
SIGNAL pdps           :std_logic;
SIGNAL pdis           :std_logic;
SIGNAL pcs            :std_logic;
SIGNAL pss            :std_logic;
SIGNAL fmsp           :std_logic;
SIGNAL pas_fin        :std_logic;
SIGNAL pdps_fin       :std_logic;
SIGNAL pdis_fin       :std_logic;
SIGNAL pcs_fin        :std_logic;
SIGNAL pmts           :std_logic;
SIGNAL razrs          :std_logic;
SIGNAL decal          :std_logic;
SIGNAL ch_compte      :unsigned(NB_LCMSP-1 DOWNTO 0);
SIGNAL msp_compte     :unsigned(NB_LMSP-1 DOWNTO 0);
SIGNAL bits_compte    :unsigned(NB_LMSP-1 DOWNTO 0);
SIGNAL pas_compte     :unsigned(NB_NBA-1 DOWNTO 0);
SIGNAL pdps_compte    :unsigned(NB_NBDP-1 DOWNTO 0);
SIGNAL pdis_compte    :unsigned(NB_NBDI-1 DOWNTO 0);
SIGNAL pcs_compte     :unsigned(NB_CANAL-1 DOWNTO 0);
SIGNAL pss_compte     :unsigned(NB_SITE-1 DOWNTO 0);
SIGNAL lire           :std_logic;
SIGNAL   bpv_int      :std_logic;

BEGIN
-----------------------------------------------------------------------
-- Signaux de sortie
-----------------------------------------------------------------------
sortie_proc:PROCESS(bpv_int,lire)
  BEGIN
    bpv <= bpv_int;
    ram_lire  <= lire;
  END PROCESS sortie_proc;
-----------------------------------------------------------------------
-
```

```
--  Description des démultiplexeurs de paramètre (choix entre
paramètres
-- commun ou spécifique) et de source d'adresse (canal et site).
-----------------------------------------------------------------------
-
-- Amplitude (AMP)
mux_amp:PROCESS(ampc,as,fa)
   BEGIN
      IF (fa = '1') THEN
         bamp <= as;
      ELSE
         bamp <= ampc;
      END IF;
   END PROCESS mux_amp;
-- Durée de phase (DP)
mux_dp:PROCESS(dpc,dps,fdp)
   BEGIN
      IF (fdp = '1') THEN
         bdp <= dps;
      ELSE
         bdp <= dpc;
      END IF;
   END PROCESS mux_dp;
-- Durée d'interphase (DI)
mux_di:PROCESS(dic,dis,fdi)
   BEGIN
      IF (fdi = '1') THEN
         bdi <= dis;
      ELSE
         bdi <= dic;
      END IF;
   END PROCESS mux_di;
-- Adressage de canal
mux_canal:PROCESS(ram_canal,cs,fcs)
   BEGIN
      IF (fcs = '1') THEN
         bcanal <= cs;
      ELSE
         bcanal <= ram_canal;
      END IF;
   END PROCESS mux_canal;
--  Adressage de site
mux_site:PROCESS(ram_site,ss,fcs)
   BEGIN
      IF (fcs = '1') THEN
         bsite <= ss;
      ELSE
         bsite <= ram_site;
      END IF;
   END PROCESS mux_site;
-----------------------------------------------------------------------
--  Description des registre à décalage des paramètres spécifiques et
-- d'adresse des canaux et sites.
-----------------------------------------------------------------------
-- Amplitude spécifique (AS)
reg_amp:PROCESS(horl,razg)
   BEGIN
```

```
      IF(razg = '0') THEN
         as <= (OTHERS => '0');
      ELSIF (horl'EVENT AND horl = '1') THEN
         IF (pas = '1') THEN
            IF(razrs = '1' OR bpv_int = '1') THEN
               as(NB_AMP-2 DOWNTO 0) <= "0000000";
            ELSE
               FOR i IN 0 TO (NB_AMP-2) LOOP
                  as(i) <= as(i+1);
               END LOOP;
            END IF;
            as(NB_AMP-1)   <= pmts;
         ELSE
            IF (razrs = '1' OR bpv_int = '1') THEN
               as <= (OTHERS => '0');
            END IF;
         END IF;
      END IF;
   END PROCESS reg_amp;
-- Durée de phase spécifique (DPS)
reg_dp:PROCESS(horl,razg)
   BEGIN
      IF(razg = '0') THEN
         dps <= (OTHERS => '0');
      ELSIF (horl'EVENT AND horl = '1') THEN
         IF (pdps = '1') THEN
            IF(razrs = '1' OR bpv_int = '1') THEN
               dps(NB_DP-1 DOWNTO 1) <= "0000000";
            ELSE
               FOR i IN (NB_DP-1) DOWNTO 1 LOOP
                  dps(i) <= dps(i-1);
               END LOOP;
            END IF;
            dps(0) <= pmts;
         ELSE
            IF (razrs = '1' OR bpv_int = '1') THEN
               dps <= (OTHERS => '0');
            END IF;
         END IF;
      END IF;
   END PROCESS reg_dp;
-- Durée d'inter-phase spécifique (DPI)
reg_di:PROCESS(horl,razg)
   BEGIN
      IF(razg = '0') THEN
         dis <= (OTHERS => '0');
      ELSIF (horl'EVENT AND horl = '1') THEN
         IF (pdis = '1') THEN
            IF(razrs = '1' OR bpv_int = '1') THEN
               dis(NB_DI-1 DOWNTO 1) <= "0000000";
            ELSE
               FOR i IN (NB_DI-1) DOWNTO 1 LOOP
                  dis(i) <= dis(i-1);
               END LOOP;
            END IF;
            dis(0) <= pmts;
         ELSE
```

```
            IF (razrs = '1' OR bpv_int = '1') THEN
               dis <= (OTHERS => '0');
            END IF;
         END IF;
      END IF;
   END PROCESS reg_di;
-- Canal spécifique (CS)
reg_cs:PROCESS(horl,razg)
   BEGIN
      IF(razg = '0') THEN
         cs <= (OTHERS => '0');
      ELSIF (horl'EVENT AND horl = '1') THEN
         IF (pcs = '1') THEN
            IF(razrs = '1' OR bpv_int = '1') THEN
               cs(NB_CANAL-2 DOWNTO 0) <= "000";
            ELSE
               FOR i IN 0 TO (NB_CANAL-2) LOOP
                  cs(i)   <= cs(i+1);
               END LOOP;
            END IF;
            cs(NB_CANAL-1) <= pmts;
         ELSE
            IF (razrs = '1' OR bpv_int = '1') THEN
               cs <= (OTHERS => '0');
            END IF;
         END IF;
      END IF;
   END PROCESS reg_cs;
-- Site spécifique (SS)
reg_ss:PROCESS(horl,razg)
   BEGIN
      IF(razg = '0') THEN
         ss <= (OTHERS => '0');
      ELSIF (horl'EVENT AND horl = '1') THEN
         IF (pss = '1') THEN
            IF(razrs = '1' OR bpv_int = '1') THEN
               ss(NB_SITE-2 DOWNTO 0)  <= "000";
            ELSE
               FOR i IN 0 TO (NB_SITE-2) LOOP
                  ss(i)   <= ss(i+1);
               END LOOP;
            END IF;
            ss(NB_SITE-1)  <= pmts;
         ELSE
            IF (razrs = '1' OR bpv_int = '1') THEN
               ss <= (OTHERS => '0');
            END IF;
         END IF;
      END IF;
   END PROCESS reg_ss;
-------------------------------------------------------------------------
-- Convertisseur parallèle-série (les MSP)
-------------------------------------------------------------------------
reg_pmt:PROCESS(horl,razg)
   BEGIN
      IF(razg = '0') THEN
         pmtreg <= (OTHERS => '0');
```

```
      ELSIF (horl'EVENT AND horl = '1') THEN
         IF (mcv = '1' AND mode = MODE_STIMULATION) THEN
            pmtreg <= pmt;
            pmts <= pmt(0);
         ELSIF (decal = '1') THEN
            pmts <= pmtreg(1);
            FOR i IN 0 TO (NB_PMT-2) LOOP
               pmtreg(i) <= pmtreg(i+1);
            END LOOP;
            pmtreg(NB_PMT-1) <= '0';
         END IF;
      END IF;
   END PROCESS reg_pmt;
-----------------------------------------------------------------------
-- Machine à état du module de formation de mot de stimulation complet
-----------------------------------------------------------------------
-- Etat présent
etat_generateur:PROCESS(horl,razg)
   BEGIN
      IF (razg = '0') THEN
         etat_gen    <= REPOS;
      ELSIF (horl'EVENT AND horl = '1') THEN
         etat_gen    <= p_etat_gen;
      END IF;
   END PROCESS etat_generateur;
-- État suivant
proch_etat_generateur:PROCESS(etat_gen)
   BEGIN
      CASE etat_gen IS
         WHEN REPOS =>
            IF (mcv = '1'AND mode = MODE_STIMULATION)   THEN
               p_etat_gen  <= FORMATION;
            ELSE
               p_etat_gen  <= REPOS;
            END IF;
         WHEN FORMATION =>
            IF (ch_compte = (unsigned(lchaine)-'1') AND fmsp = '1'
               AND msp_compte /= MSP_ZERO) THEN
               p_etat_gen <= REPOS;
            ELSIF (ch_compte = CH_ZERO AND msp_compte = MSP_ZERO
               AND bits_compte /= BITS_ZERO) THEN
               p_etat_gen <= REPOS;
            ELSIF (bits_compte = VAL_NB_MCMD AND mcv = '0') THEN
               p_etat_gen  <= ATTENTE;
            ELSE
               p_etat_gen  <= FORMATION;
            END IF;
         WHEN ATTENTE =>
            IF (mcv = '1' AND mode = MODE_CONFIGURATION) THEN
               p_etat_gen  <= REPOS;
            ELSIF (mcv = '1' AND mode = MODE_STIMULATION) THEN
               p_etat_gen  <= FORMATION;
            ELSE
               p_etat_gen  <= ATTENTE;
            END IF;
         WHEN OTHERS =>
            p_etat_gen <= REPOS;
```

```
    END CASE;
  END PROCESS proch_etat_generateur;
-----------------------------------------------------------------------
-- Compteur des bits d'un MSP.
-- Compteur d'une chaîne de MSP.
-- Contrôle des signaux lire et razrs.
-----------------------------------------------------------------------
compteur_msp:PROCESS(razg,horl)
  BEGIN
    IF (razg = '0') THEN
      msp_compte  <= (OTHERS => '0');
      ch_compte <= (OTHERS => '0');
      lire    <= '0';
      razrs     <= '1';
    ELSIF (horl'EVENT AND horl = '1') THEN
      IF (((mcv = '1' AND mode = MODE_STIMULATION)
             OR etat_gen = FORMATION)
             AND (bits_compte <= VAL_OFF_COMPTEUR_BITS)) THEN
          razrs <= '0';
        IF (msp_compte = MSP_ZERO) THEN
          IF((fa = '1' AND fdp = '0' AND fdi = '0' AND fcs = '0'
               AND unsigned(nba) = NBA_ZERO) OR(fa = '0' AND fdp = '1
               AND fdi = '0' AND fcs = '0'
               AND unsigned(nbdp) = NBDP_ZERO) OR (fa = '0'
               AND fdp = '0' AND fdi = '1' AND fcs = '0'
               AND unsigned(nbdi) = NBDI_ZERO)) THEN
            msp_compte <= (OTHERS => '0');
            IF(ch_compte = CH_ZERO AND bits_compte = BITS_ZERO
              AND lire = '0') THEN
              lire <= '1';
              IF(std_logic_vector(ch_compte)=lchaine)THEN
                ch_compte <= (OTHERS => '0');
              ELSE
                ch_compte <= ch_compte + '1';
              END IF;
            ELSIF (ch_compte = CH_ZERO AND bits_compte = BITS_ZERO
                   AND lire = '1' AND lchaine = (CH_ZERO + '1'))
                THEN
                lire <= '0';
                IF(std_logic_vector(ch_compte)=lchaine)THEN
                   ch_compte <= (OTHERS => '0');
                ELSE
                   ch_compte <= ch_compte + '1';
                END IF;
            ELSIF (bits_compte = BITS_ZERO AND
                ch_compte < unsigned(lchaine)) THEN
                lire <= '1';
                IF (std_logic_vector(ch_compte) = lchaine) THEN
                   ch_compte <= (OTHERS => '0');
                ELSE
                   ch_compte <= ch_compte + '1';
                END IF;
            ELSIF (ch_compte < unsigned(lchaine) AND lire = '1'
                AND bits_compte < (VAL_OFF_COMPTEUR_BITS-'1')) THEN
                lire <= '1';
                IF (std_logic_vector(ch_compte) = lchaine) THEN
                   ch_compte <= (OTHERS => '0');
```

```
                    ELSE
                       ch_compte <= ch_compte + '1';
                    END IF;
                ELSIF (ch_compte = unsigned(lchaine)) THEN
                    lire <= '0';
                    IF (std_logic_vector(ch_compte) = lchaine) THEN
                       ch_compte <= (OTHERS => '0');
                    ELSE
                       ch_compte <= ch_compte + '1';
                    END IF;
                ELSE
                   lire <= '0';
                END IF;
        ELSE
        IF (mcv = '1' AND mode = MODE_STIMULATION
             AND ch_compte = CH_ZERO) THEN
           lire <= '1';
        ELSE
           lire <= '0';
        END IF;
           msp_compte <= msp_compte + '1';
        END IF;
      ELSIF (fmsp = '1'AND ch_compte < (unsigned(lchaine)-'1')) THEN
           msp_compte <= (OTHERS => '0');
        lire <= '1';
        IF (std_logic_vector(ch_compte) = lchaine) THEN
                ch_compte <= (OTHERS => '0');
        ELSE
           ch_compte <= ch_compte + '1';
        END IF;
      ELSIF (fmsp = '1'
        AND ch_compte < unsigned(lchaine)) THEN
        msp_compte <= (OTHERS => '0');
        lire <= '0';
        IF (std_logic_vector(ch_compte) = lchaine) THEN
           ch_compte <= (OTHERS => '0');
        ELSE
           ch_compte <= ch_compte + '1';
        END IF;
      ELSE
        msp_compte <= msp_compte + '1';
        lire <= '0';
      END IF;
   ELSIF (etat_gen = REPOS) THEN
          msp_compte  <= (OTHERS => '0');
          ch_compte <= (OTHERS => '0');
          lire <= '0';
          razrs <= '1';
   ELSE
          lire <= '0';
          razrs <= '0';
   END IF;
END IF;
   END PROCESS compteur_msp;
-------------------------------------------------------------------------
-- Signal bpv_int
-------------------------------------------------------------------------
```

```
bpv_int_proc:PROCESS(razg,horl)
  BEGIN
    IF (razg = '0') THEN
      bpv_int <= '0';
    ELSIF (horl'EVENT AND horl = '1') THEN
      IF(decal = '1') THEN
        IF (fmsp = '1' AND etat_gen = FORMATION) THEN
          bpv_int<= '1';
        ELSE
          bpv_int <= '0';
        END IF;
      ELSE
        bpv_int <= '0';
      END IF;
    END IF;
  END PROCESS bpv_int_proc;
-----------------------------------------------------------------------
-- Compteur des bits d'un mot de parametre (MPMT)
-----------------------------------------------------------------------
compteur_bits:PROCESS(razg,horl)
  BEGIN
    IF (razg = '0') THEN
      bits_compte <= (OTHERS => '0');
      decal     <= '0';
    ELSIF (horl'EVENT AND horl = '1') THEN
      IF(etat_gen = FORMATION) THEN
        IF (bits_compte = VAL_MAX_COMPTEUR_BITS) THEN
          bits_compte    <= (OTHERS => '0');
          decal       <= '0';
        ELSE
          bits_compte    <= bits_compte + '1';
          decal       <= '1';
        END IF;
      ELSE
        bits_compte <= (OTHERS => '0');
        decal     <= '0';
      END IF;
    END IF;
  END PROCESS compteur_bits;
-----------------------------------------------------------------------
-- Séquenceur
-----------------------------------------------------------------------
sequence:PROCESS(razg,horl)
  BEGIN
    IF (razg = '0') THEN
      pas       <= '0';
      pdps    <= '0';
      pdis    <= '0';
      pcs       <= '0';
      pss       <= '0';
      fmsp    <= '0';
      pas_fin   <= '0';
      pdps_fin  <= '0';
      pdis_fin  <= '0';
      pcs_fin   <= '0';
      pas_compte  <= (OTHERS => '0');
      pdps_compte <= (OTHERS => '0');
```

```
   pdis_compte <= (OTHERS => '0');
   pcs_compte  <= (OTHERS => '0');
   pss_compte  <= (OTHERS => '0');
ELSIF (horl'EVENT AND horl = '1') THEN
   IF (etat_gen = FORMATION AND bits_compte
        < (VAL_MAX_COMPTEUR_BITS-'1') AND
        (ch_compte < unsigned(lchaine) OR (msp_compte = MSP_ZERO
        AND ch_compte = unsigned(lchaine)))) THEN
      IF(fa = '1' AND pas_compte <= unsigned(nba) AND
        pas_fin = '0') THEN
        pas     <= '1';
        pdps    <= '0';
        pdis    <= '0';
        pcs        <= '0';
        pss        <= '0';
        IF (pas_compte = unsigned(nba) AND fdp = '0' AND fdi = '0'
             AND fcs = '0') THEN
          pas_compte  <= (OTHERS => '0');
          fmsp       <= '1';
          pas_fin    <= '0';
        ELSIF(pas_compte = unsigned(nba)) THEN
          pas_compte  <= (OTHERS => '0');
          fmsp    <= '0';
          pas_fin    <= '1';
        ELSE
          pas_compte  <= pas_compte + '1';
          fmsp    <= '0';
          pas_fin    <= '0';
        END IF;
ELSIF (fdp = '1' AND pdps_compte <= unsigned(nbdp) AND
        pdps_fin = '0') THEN
        pas     <= '0';
        pdps       <= '1';
        pdis    <= '0';
        pcs        <= '0';
        pss        <= '0';
   IF(pdps_compte = unsigned(nbdp) AND fdi = '0' AND
            fcs = '0') THEN
          pdps_compte <= (OTHERS => '0');
          fmsp    <= '1';
          pas_fin    <= '0';
          pdps_fin  <= '0';
        ELSIF (pdps_compte = unsigned(nbdp)) THEN
          pdps_compte <= (OTHERS => '0');
          fmsp    <= '0';
          pdps_fin  <= '1';
        ELSE
          pdps_compte <= pdps_compte + '1';
          fmsp    <= '0';
          pdps_fin  <= '0';
        END IF;
   ELSIF(fdi = '1' AND pdis_compte <= unsigned(nbdi) AND
          pdis_fin = '0') THEN
        pas        <= '0';
        pdps    <= '0';
        pdis    <= '1';
        pcs        <= '0';
```

```
        pss       <= '0';
   IF(pdis_compte = unsigned(nbdi) AND fcs = '0') THEN
          pdis_compte <= (OTHERS => '0');
          fmsp    <= '1';
          pas_fin   <= '0';
          pdps_fin  <= '0';
          pdis_fin  <= '0';
        ELSIF (pdis_compte = unsigned(nbdi)) THEN
          pdis_compte <= (OTHERS => '0');
          fmsp    <= '0';
          pdis_fin  <= '1';
        ELSE
          pdis_compte <= pdis_compte + '1';
          fmsp    <= '0';
          pdis_fin  <= '0';
        END IF;
ELSIF(fcs = '1' AND pcs_compte <= VAL_MAX_NB_CANAL AND
        pcs_fin = '0') THEN
        pas       <= '0';
        pdps    <= '0';
        pdis    <= '0';
        pcs       <= '1';
        pss       <= '0';
        IF(pcs_compte = VAL_MAX_NB_CANAL) THEN
          pcs_compte  <= (OTHERS => '0');
          pcs_fin<= '1';
        ELSE
          pcs_compte  <= pcs_compte + '1';
          pcs_fin<= '0';
        END IF;
   ELSIF(fcs = '1' AND pss_compte <= VAL_MAX_NB_SITE AND
        fmsp = '0') THEN
        pas       <= '0';
        pdps      <= '0';
        pdis      <= '0';
        pcs       <= '0';
        pss       <= '1';
        IF(pss_compte = VAL_MAX_NB_SITE) THEN
          pss_compte  <= (OTHERS => '0');
          fmsp      <= '1';
          pas_fin   <= '0';
          pdps_fin  <= '0';
          pdis_fin  <= '0';
          pcs_fin   <= '0';
        ELSE
          pss_compte  <= pss_compte + '1';
          fmsp      <= '0';
        END IF;
      END IF;
ELSIF (etat_gen = FORMATION OR etat_gen = ATTENTE) THEN
     pas       <= '0';
     pdps      <= '0';
     pdis      <= '0';
     pcs       <= '0';
     pss       <= '0';
     fmsp      <= '0';
   ELSIF (etat_gen = REPOS) THEN
```

```
          pas       <= '0';
          pdps      <= '0';
          pdis      <= '0';
          pcs       <= '0';
          pss       <= '0';
          fmsp      <= '0';
          pas_fin   <= '0';
          pdps_fin  <= '0';
          pdis_fin  <= '0';
          pcs_fin     <= '0';
          pas_compte  <= (OTHERS => '0');
          pdps_compte <= (OTHERS => '0');
          pdis_compte <= (OTHERS => '0');
          pcs_compte  <= (OTHERS => '0');
          pss_compte  <= (OTHERS => '0');
        END IF;
      END IF;
   END PROCESS sequence;
END rtl;
```

HORLSTIM.VHD

```
LIBRARY IEEE;
  USE IEEE.std_logic_1164.all;
  USE IEEE.std_logic_arith.all;
  USE work.pivc_def.all;

ENTITY horlstim IS
  PORT( horl   : IN  std_logic;
     razg      : IN  std_logic;
     sync      : IN  std_logic;
     rhs       : IN  std_logic_vector(NB_HORLS-1 DOWNTO 0);
     horls     : OUT std_logic
     );
END horlstim;

ARCHITECTURE rtl OF horlstim IS

signal compt_horl     : unsigned(NB_HORLS-1 DOWNTO 0);

BEGIN
-----------------------------------------------------------------------
-- Horloge de stimulation
-----------------------------------------------------------------------
horloge:PROCESS(horl,razg)
  BEGIN
     IF (razg = '0') THEN
       compt_horl  <= (OTHERS => '0');
       horls     <= '0';
     ELSIF (horl'EVENT AND horl = '1') THEN
       IF (unsigned(rhs) = ZERO_COMPT OR sync = '1') THEN
         compt_horl  <= (OTHERS => '0');
         horls     <= '0';
       ELSIF (sync = '0') THEN
         IF (unsigned(rhs) = UN_COMPT) THEN
           compt_horl  <= (OTHERS => '0');
```

```
            horls     <= '1';
          ELSIF(compt_horl = ZERO_COMPT) THEN
            compt_horl  <= compt_horl + '1';
            horls     <= '1';
          ELSIF(compt_horl = (unsigned(rhs)-'1')) THEN
            compt_horl  <= (OTHERS => '0');
            horls     <= '0';
          ELSE
            compt_horl  <= compt_horl + '1';
            horls     <= '0';
          END IF;
        END IF;
      END IF;
  END PROCESS horloge;
END rtl;
```

CPIVC.VHD

```
LIBRARY IEEE;
  USE IEEE.std_logic_1164.all;
  USE IEEE.std_logic_arith.all;
  USE work.pivc_def.all;

ENTITY cpivc IS
  PORT(horl :IN  std_logic;
  dcts    :IN  std_logic;
  razg    :OUT std_logic;
  sync    :OUT std_logic;
  bamp    :OUT std_logic_vector(NB_AMP-1 DOWNTO 0);
  bdp     :OUT std_logic_vector(NB_DP-1 DOWNTO 0);
  bdi     :OUT std_logic_vector(NB_DI-1 DOWNTO 0);
  bsite   :OUT std_logic_vector(NB_SITE-1 DOWNTO 0);
  bcanal  :OUT std_logic_vector(NB_CANAL-1 DOWNTO 0);
  bpv     :OUT std_logic;
  bths    :OUT std_logic_vector(NB_BTHS-1 DOWNTO 0);
  -- Signaux et donnees de verification
  trame_don : OUT std_logic_vector(NB_TRAME-1 DOWNTO 0);
  trmdtc  : OUT std_logic;
  mcmd    : OUT std_logic_vector(NB_MCMD-1 DOWNTO 0);
  mcv     : OUT std_logic;
  rc1     : OUT std_logic_vector(NB_RC1-1 DOWNTO 0);
  rc2     : OUT std_logic_vector(NB_RC2-1 DOWNTO 0);
  rc3     : OUT std_logic_vector(NB_RC3-1 DOWNTO 0);
  canal_ent : OUT std_logic_vector(NB_CANAL-1 DOWNTO 0);
  site_ent  : OUT std_logic_vector(NB_SITE-1 DOWNTO 0);
  moderam   : OUT std_logic;
  ecrire    : OUT std_logic;
  raz_adresse      : OUT std_logic;
  adresse_ramacs   : OUT std_logic_vector(NB_ADR_RAMACS-1 DOWNTO 0);
  lire             : OUT std_logic;
  canal_sort       : OUT std_logic_vector(NB_CANAL-1 DOWNTO 0);
  site_sort        : OUT std_logic_vector(NB_SITE-1 DOWNTO 0)
  );
END cpivc;

ARCHITECTURE rtl OF cpivc IS
```

```
-----------------------------------------------------------------------
--  Declaration des modules
-----------------------------------------------------------------------

COMPONENT trame
  PORT(horl    : IN  std_logic;
     dcts      : IN  std_logic;
     trmdtc    : OUT std_logic;
     trame     : OUT std_logic_vector(NB_TRAME-1 DOWNTO 0)
       );
  END COMPONENT;

COMPONENT hamming
 PORT( horl    : IN  std_logic;
     trmdtc    : IN  std_logic;
     trame     : IN  std_logic_vector(NB_TRAME-1 DOWNTO 0);
     mcmd      : OUT std_logic_vector(NB_MCMD-1 DOWNTO 0);
     mcv       : OUT std_logic
    );
END COMPONENT;

COMPONENT cmma
 PORT( horl    : IN  std_logic;
     mcv       : IN  std_logic;
     mode      : IN  std_logic;
     instr     : IN  std_logic_vector(NB_INSTR-1 DOWNTO 0);
     cfg       : IN  std_logic_vector(NB_CFG-1 DOWNTO 0);
     razg      : OUT std_logic
    );
END COMPONENT;

COMPONENT ccfg
 PORT( razg :IN std_logic;
     horl   :IN std_logic;
     mode   :IN std_logic;
     instr  :IN std_logic_vector(NB_INSTR-1 DOWNTO 0);
     cfg    :IN std_logic_vector(NB_CFG-1 DOWNTO 0);
     mcv    :IN std_logic;
     rc1    :OUT std_logic_vector(NB_RC1-1 DOWNTO 0);
     rc2    :OUT std_logic_vector(NB_RC2-1 DOWNTO 0);
     rc3    :OUT std_logic_vector(NB_RC3-1 DOWNTO 0);
     sync   :OUT std_logic;
     ram_canal   :OUT std_logic_vector(NB_CANAL-1 DOWNTO 0);
     ram_site    :OUT std_logic_vector(NB_SITE-1 DOWNTO 0);
     moderam     :OUT std_logic;
     ecrire      :OUT std_logic;
     raz_adresse :OUT std_logic
    );
 END COMPONENT;

COMPONENT acs
 PORT(canal_ent  :  IN std_logic_vector(NB_CANAL-1 DOWNTO 0);
  site_ent       :  IN std_logic_vector(NB_SITE-1 DOWNTO 0);
  moderam        :  IN std_logic;
  horl           :  IN std_logic;
  ecrire         :  IN std_logic;
```

```
   lire            :  IN std_logic;
   raz_adresse     :  IN std_logic;
   razg            :  IN std_logic;
    canal_sort     : OUT std_logic_vector(NB_CANAL-1 DOWNTO 0);
   site_sort       :  OUT std_logic_vector(NB_SITE-1 DOWNTO 0);
   adresse_ramacs    :OUT std_logic_vector(NB_ADR_RAMACS-1 DOWNTO 0)
    );
END COMPONENT;

COMPONENT gen
 PORT( razg :IN std_logic;
     horl    :IN std_logic;
     mcv     :IN std_logic;
     mode    :IN std_logic;
     fa      :IN std_logic;
     fdp     :IN std_logic;
     pmt     :IN std_logic_vector(NB_PMT-1 DOWNTO 0);
     lchaine :IN std_logic_vector(NB_LCMSP-1 DOWNTO 0);
     nba     :IN std_logic_vector(NB_NBA-1 DOWNTO 0);
     nbdp    :IN std_logic_vector(NB_NBDP-1 DOWNTO 0);
     fdi     :IN std_logic;
     nbdi    :IN std_logic_vector(NB_NBDI-1 DOWNTO 0);
     fcs     :IN std_logic;
     ampc    :IN std_logic_vector(NB_AMP-1 DOWNTO 0);
     dpc     :IN std_logic_vector(NB_DP-1 DOWNTO 0);
     dic     :IN std_logic_vector(NB_DI-1 DOWNTO 0);
     ram_site  :IN std_logic_vector(NB_SITE-1 DOWNTO 0);
     ram_canal :IN std_logic_vector(NB_CANAL-1 DOWNTO 0);
     ram_lire  :OUT std_logic;
     bamp    :OUT std_logic_vector(NB_AMP-1 DOWNTO 0);
     bdp     :OUT std_logic_vector(NB_DP-1 DOWNTO 0);
     bdi     :OUT std_logic_vector(NB_DI-1 DOWNTO 0);
     bsite   :OUT std_logic_vector(NB_SITE-1 DOWNTO 0);
     bcanal  :OUT std_logic_vector(NB_CANAL-1 DOWNTO 0);
     bpv     :OUT std_logic
    );
END COMPONENT;

COMPONENT horlstim
 PORT( horl    : IN  std_logic;
     razg      : IN  std_logic;
     sync      : IN  std_logic;
     rhs       : IN  std_logic_vector(NB_HORLS-1 DOWNTO 0);
     horls     : OUT std_logic
    );
END COMPONENT;

-----------------------------------------------------------------------
--  Declaration des signaux
-----------------------------------------------------------------------

SIGNAL trame_cpivc    :std_logic_vector(NB_TRAME-1 DOWNTO 0);
SIGNAL trmdtc_cpivc:std_logic;
SIGNAL mcmd_cpivc     :std_logic_vector(NB_MCMD-1 DOWNTO 0);
SIGNAL mcv_cpivc      :std_logic;
SIGNAL raz_cpivc      :std_logic;
SIGNAL rc1_cpivc      :std_logic_vector(NB_RC1-1 DOWNTO 0);
```

```
SIGNAL rc2_cpivc      :std_logic_vector(NB_RC2-1 DOWNTO 0);
SIGNAL rc3_cpivc      :std_logic_vector(NB_RC3-1 DOWNTO 0);
SIGNAL racs_canal_ent_cpivc  :std_logic_vector(NB_CANAL-1 DOWNTO 0);
SIGNAL racs_site_ent_cpivc :std_logic_vector(NB_SITE-1 DOWNTO 0);
SIGNAL moderam_cpivc         :std_logic;
SIGNAL ecrire_cpivc        :std_logic;
SIGNAL raz_adresse_cpivc     :std_logic;
SIGNAL racs_canal_sort_cpivc :std_logic_vector(NB_CANAL-1 DOWNTO 0);
SIGNAL racs_site_sort_cpivc  :std_logic_vector(NB_SITE-1 DOWNTO 0);
SIGNAL lire_cpivc            :std_logic;

-----------------------------------------------------------------------
--  Definition des interconnexions
-----------------------------------------------------------------------
BEGIN
trame_inst: trame
PORT MAP(horl    => horl,
  dcts      => dcts,
  trame       => trame_cpivc,
  trmdtc    => trmdtc_cpivc
  );

hamming_inst: hamming
PORT MAP(horl    => horl,
      trame => trame_cpivc,
      trmdtc=> trmdtc_cpivc,
      mcmd     => mcmd_cpivc,
      mcv    => mcv_cpivc
     );

cmma_inst: cmma
PORT MAP(horl    => horl,
      mcv        => mcv_cpivc,
      mode       => mcmd_cpivc(0),
      instr(NB_INSTR-1 DOWNTO 0) => mcmd_cpivc(ID_MCFG_INSTR_02
            DOWNTO ID_MCFG_INSTR_00),
  cfg(NB_CFG-1 DOWNTO 0)       => mcmd_cpivc(ID_MCFG_CFG_21
            DOWNTO ID_MCFG_CFG_00),
  razg           => raz_cpivc
  );

ccfg_inst: ccfg
PORT MAP(razg    => raz_cpivc,
  horl           => horl,
  mode           => mcmd_cpivc(ID_MCFG_MODE),
  instr(NB_INSTR-1 DOWNTO 0)=> mcmd_cpivc(ID_MCFG_INSTR_02
         DOWNTO ID_MCFG_INSTR_00),
  cfg(NB_CFG-1 DOWNTO 0)     => mcmd_cpivc(ID_MCFG_CFG_21
         DOWNTO ID_MCFG_CFG_00),
  mcv          => mcv_cpivc,
  rc1          => rc1_cpivc,
  rc2          => rc2_cpivc,
  rc3          => rc3_cpivc,
  sync         => sync,
  ram_canal    => racs_canal_ent_cpivc,
  ram_site     => racs_site_ent_cpivc,
  moderam      => moderam_cpivc,
```

```
   ecrire       => ecrire_cpivc,
   raz_adresse => raz_adresse_cpivc
   );

acs_inst: acs
PORT MAP(canal_ent => racs_canal_ent_cpivc,
      site_ent      => racs_site_ent_cpivc,
      moderam       => moderam_cpivc,
      horl          => horl,
      ecrire        => ecrire_cpivc,
      lire          => lire_cpivc,
      raz_adresse   => raz_adresse_cpivc,
      razg          => raz_cpivc,
        canal_sort => racs_canal_sort_cpivc,
      site_sort     => racs_site_sort_cpivc,
      adresse_ramacs => adresse_ramacs
     );

gen_inst: gen
PORT MAP(razg     => raz_cpivc,
      horl        => horl,
      mcv         => mcv_cpivc,
      mode        => mcmd_cpivc(ID_MSTIM_MODE),
      fa          => rc1_cpivc(ID_FA),
      fdp         => rc1_cpivc(ID_FDP),
      pmt         => mcmd_cpivc(ID_MSTIM_PMT_24 DOWNTO
                              ID_MSTIM_PMT_00),
      lchaine     => rc1_cpivc(ID_LCMSP_08 DOWNTO ID_LCMSP_00),
      nba         => rc1_cpivc(ID_NBA_02 DOWNTO ID_NBA_00),
      nbdp        => rc1_cpivc(ID_NBDP_02 DOWNTO ID_NBDP_00),
      fdi         => rc1_cpivc(ID_FDI),
      nbdi        => rc1_cpivc(ID_NBDI_02 DOWNTO ID_NBDI_00),
      fcs         => rc1_cpivc(ID_FCS),
      ampc        => rc2_cpivc(ID_AMPC_07 DOWNTO ID_AMPC_00),
      dpc         => rc2_cpivc(ID_DPC_07 DOWNTO ID_DPC_00),
      dic         => rc3_cpivc(ID_DIC_07 DOWNTO ID_DIC_00),
      ram_site    => racs_site_sort_cpivc,
      ram_canal   => racs_canal_sort_cpivc,
      ram_lire    => lire_cpivc,
      bamp        => bamp,
      bdp         => bdp,
      bdi         => bdi,
      bsite       => bsite,
      bcanal      => bcanal,
      bpv         => bpv
     );
-----------------------------------------------------------------------
--  Signaux et données de sortie
-----------------------------------------------------------------------
verification: PROCESS(trame_cpivc,trmdtc_cpivc,mcmd_cpivc,mcv_cpivc)
  BEGIN
     razg      <= raz_cpivc;
     bths      <= rc3_cpivc(ID_BTHS_07 DOWNTO ID_BTHS_00);
     -- Signaux et données de vérification
     trame_don <= trame_cpivc;
     trmdtc    <= trmdtc_cpivc;
     mcmd    <= mcmd_cpivc;
```

```
      mcv     <= mcv_cpivc;
      rc1     <= rc1_cpivc;
      rc2     <= rc2_cpivc;
      rc3     <= rc3_cpivc;
      canal_ent    <= racs_canal_ent_cpivc;
      site_ent     <= racs_site_ent_cpivc;
      moderam      <= moderam_cpivc;
      ecrire       <= ecrire_cpivc;
      raz_adresse <= raz_adresse_cpivc;
      lire         <= lire_cpivc;
      canal_sort   <= racs_canal_sort_cpivc;
      site_sort    <= racs_site_sort_cpivc;
   END PROCESS verification;
END rtl;
```

AMPFIFO.VHD

```
LIBRARY IEEE;
   USE IEEE.std_logic_1164.all;
   USE work.pivc_def.all;
LIBRARY lpm;
   USE lpm.lpm_components.ALL;

ENTITY ampfifo IS
    PORT( amp_ent : IN  std_logic_vector(NB_AMP-1 DOWNTO 0);
   horl           : IN  std_logic;
   ecritf         : IN  std_logic;
   lectf          : IN  std_logic;
   vide           : OUT  std_logic;
   amp_sort       : OUT std_logic_vector(NB_AMP-1 DOWNTO 0)
   );
END ampfifo;

ARCHITECTURE rtl OF ampfifo IS

COMPONENT scfifo
   GENERIC (LPM_WIDTH: POSITIVE;
      LPM_WIDTHU: POSITIVE;
      LPM_NUMWORDS: POSITIVE;
      LPM_SHOWAHEAD: STRING := "OFF";
      ALMOST_FULL_VALUE: POSITIVE:= 0;
      ALMOST_EMPTY_VALUE: POSITIVE:= 0;
      ALLOW_RWCYCLE_WHEN_FULL: STRING := "OFF";
      MAXIMIZE_SPEED: POSITIVE:= 5;
      OVERFLOW_CHECKING: STRING:= "ON";
      UNDERFLOW_CHECKING: STRING:= "ON";
  USE_EAB: STRING:="ON");
   PORT (data: IN STD_LOGIC_VECTOR(LPM_WIDTH-1 DOWNTO 0);
      clock, wrreq, rdreq: IN STD_LOGIC; empty: OUT STD_LOGIC;
      q: OUT STD_LOGIC_VECTOR(LPM_WIDTH-1 DOWNTO 0)
      );
END COMPONENT;

BEGIN
-----------------------------------------------------------------------
-- FIFO
```

```
-----------------------------------------------------------------
---
fifo_inst: scfifo
   GENERIC MAP (use_eab    => "ON",
       lpm_width           => NB_AMP,
       lpm_widthu          => NB_COMPT_ELEMENT,
       lpm_numwords        => N_ELEMENT,
       overflow_checking   => "OFF",
       underflow_checking  => "OFF")
   PORT MAP(data           => amp_ent,
     clock                 => horl,
     wrreq                 => ecritf,
     rdreq                 => lectf,
     empty                 => vide,
     q                     => amp_sort
      );
END rtl;
```

CANFIFO.VHD

```
LIBRARY IEEE;
  USE IEEE.std_logic_1164.all;
  USE work.pivc_def.all;
LIBRARY lpm;
  USE lpm.lpm_components.ALL;

ENTITY canfifo IS
   PORT( can_ent    : IN  std_logic_vector(NB_CANAL-1 DOWNTO 0);
  horl              : IN  std_logic;
  ecritf            : IN  std_logic;
  lectf             : IN  std_logic;
  vide              : OUT  std_logic;
  can_sort          : OUT std_logic_vector(NB_CANAL-1 DOWNTO 0)
  );
END canfifo;

ARCHITECTURE rtl OF canfifo IS

COMPONENT scfifo
   GENERIC (LPM_WIDTH: POSITIVE;
      LPM_WIDTHU: POSITIVE;
      LPM_NUMWORDS: POSITIVE;
      LPM_SHOWAHEAD: STRING := "OFF";
      ALMOST_FULL_VALUE: POSITIVE:= 0;
      ALMOST_EMPTY_VALUE: POSITIVE:= 0;
      ALLOW_RWCYCLE_WHEN_FULL: STRING := "OFF";
      MAXIMIZE_SPEED: POSITIVE:= 5;
      OVERFLOW_CHECKING: STRING:= "ON";
      UNDERFLOW_CHECKING: STRING:= "ON";
  USE_EAB: STRING:="ON");
   PORT (data: IN STD_LOGIC_VECTOR(LPM_WIDTH-1 DOWNTO 0);
      clock, wrreq, rdreq: IN STD_LOGIC; empty: OUT STD_LOGIC;
      q: OUT STD_LOGIC_VECTOR(LPM_WIDTH-1 DOWNTO 0)
      );
END COMPONENT;
```

```
BEGIN
-----------------------------------------------------------------------
-- FIFO
-----------------------------------------------------------------------
fifo_inst: scfifo
   GENERIC MAP (use_eab => "ON",
     lpm_width          => NB_CANAL,
       lpm_widthu       => NB_COMPT_ELEMENT,
       lpm_numwords     => N_ELEMENT,
       overflow_checking     => "OFF",
       underflow_checking    => "OFF")
   PORT MAP(data => can_ent,
     clock   => horl,
     wrreq   => ecritf,
     rdreq   => lectf,
     empty   => vide,
     q    => can_sort
      );
END rtl;
```

CCNA.VHD

```
LIBRARY IEEE;
  USE IEEE.std_logic_1164.all;
  USE IEEE.std_logic_arith.all;
  USE work.pivc_def.all;

ENTITY ccna IS
  PORT( horl   : IN  std_logic;
    razg       : IN  std_logic;
    bcanal     : IN  std_logic_vector(NB_CANAL-1 DOWNTO 0);
    bamp       : IN  std_logic_vector(NB_AMP-1 DOWNTO 0);
    bpv        : IN std_logic;
    rcc        : IN  std_logic_vector(NCANAL-1 DOWNTO 0);
    cca        : OUT std_logic_vector(NCANAL-1 DOWNTO 0);
    cna_a      : OUT std_logic_vector(NB_CANAL-1 DOWNTO 0);
    cna_rw     : OUT std_logic;
    cna_rs     : OUT std_logic;
    cna_en     : OUT std_logic;
    cna_cs     : OUT std_logic;
    cna_ld     : OUT std_logic;
    cna_db     : OUT std_logic_vector(NB_AMP-1 DOWNTO 0)
    );
END ccna;

ARCHITECTURE rtl OF ccna IS

-----------------------------------------------------------------------
-- Déclaration des modules
-----------------------------------------------------------------------
COMPONENT ampfifo
  PORT (amp_ent  : IN  std_logic_vector(NB_AMP-1 DOWNTO 0);
  horl           : IN  std_logic;
  ecritf         : IN  std_logic;
  lectf          : IN  std_logic;
  vide           : OUT std_logic;
```

```
   amp_sort       : OUT std_logic_vector(NB_AMP-1 DOWNTO 0)
        );
  END COMPONENT;

COMPONENT canfifo
  PORT (can_ent  : IN  std_logic_vector(NB_CANAL-1 DOWNTO 0);
   horl          : IN  std_logic;
   ecritf        : IN  std_logic;
   lectf         : IN  std_logic;
   vide          : OUT std_logic;
   can_sort      : OUT std_logic_vector(NB_CANAL-1 DOWNTO 0)
        );
  END COMPONENT;

TYPE   type_etat_ccna IS (REPOS,FIFO,REQ,CS_BAS,CS_HAUT);
SIGNAL etat_ccna :type_etat_ccna;
SIGNAL vide      :std_logic;
SIGNAL vide_nc   :std_logic;
SIGNAL lectf     :std_logic;
SIGNAL canal     :std_logic_vector(NB_CANAL-1 DOWNTO 0);
SIGNAL amplitude :std_logic_vector(NB_AMP-1 DOWNTO 0);
SIGNAL proch_cca :std_logic_vector(NCANAL-1 DOWNTO 0);

BEGIN
-----------------------------------------------------------------------
-- Connexions des modules
-----------------------------------------------------------------------
ampfifo_inst: ampfifo
  PORT MAP( amp_ent   => bamp,
       horl           => horl,
       ecritf         => bpv,
       lectf          => lectf,
       vide           => vide,
       amp_sort       => amplitude
         );

canfifo_inst: canfifo
  PORT MAP( can_ent   => bcanal,
       horl           => horl,
       ecritf         => bpv,
       lectf          => lectf,
       vide           => vide_nc,
       can_sort       => canal
         );
-----------------------------------------------------------------------
-- Signaux de contrôle et données destinées au CNA
-----------------------------------------------------------------------
sign_proc:PROCESS(razg,horl)
  BEGIN
    IF (razg = '0') THEN
      cna_rs <= '0';
      cna_rw <= '1';
      cna_ld <= '1';
      cna_en <= '1';
      cna_a  <= (OTHERS => '0');
      cna_db <= (OTHERS => '0');
    ELSIF (horl'EVENT AND HORL = '1') THEN
```

```
        cna_rs <= '1';
        cna_rw <= '0';
        cna_ld <= '0';
        cna_en <= '0';
        cna_a  <= canal;
        cna_db <= amplitude;
      END IF;
    END PROCESS sign_proc;
-----------------------------------------------------------------------
-- Machine à état du CCNA
-----------------------------------------------------------------------
-- État present
proch_etat_proc:PROCESS(razg,horl)
  VARIABLE dtc     :std_logic;
  VARIABLE canal_pres:std_logic_vector(NB_CANAL
                          DOWNTO 0) :="00000";
  BEGIN
    IF (razg = '0') THEN
    ELSIF (horl'EVENT AND horl = '1') THEN
      CASE etat_ccna IS
        WHEN REPOS =>
          IF (vide = '0') THEN
            etat_ccna <= FIFO;
            lectf <= '1';
          ELSE
            etat_ccna <= REPOS;
            lectf <= '0';
          END IF;
          cna_cs <= '1';
          cca <= (OTHERS => '0');
        WHEN FIFO =>
          etat_ccna <= REQ;
          lectf     <= '0';
          cna_cs <= '1';
          cca <= (OTHERS => '0');
        WHEN REQ =>
          dtc := '0';
          canal_pres(NB_CANAL-1 DOWNTO 0) :=
               canal(NB_CANAL-1 DOWNTO 0);
          FOR i IN 0 TO NCANAL-1 LOOP
          IF (i = unsigned(canal_pres) AND rcc(i) = '1') THEN
               proch_cca(i) <= '1';
               dtc  := '1';
            ELSE
               proch_cca(i) <= '0';
            END IF;
          END LOOP;
          IF (dtc = '1') THEN
            etat_ccna <= CS_BAS;
          ELSE
            etat_ccna <= REQ;
          END IF;
          lectf  <= '0';
          cna_cs <= '1';
          cca <= (OTHERS => '0');
        WHEN CS_BAS =>
          etat_ccna <= CS_HAUT;
```

```
            lectf   <='0';
            cna_cs <= '0';
            cca <= proch_cca;
         WHEN CS_HAUT =>
            IF (vide = '0') THEN
               etat_ccna <= FIFO;
               lectf     <= '1';
            ELSE
               etat_ccna <= REPOS;
               lectf     <= '0';
            END IF;
            cna_cs <= '1';
            cca <= (OTHERS => '0');
         WHEN OTHERS =>
            etat_ccna <= REPOS;
            lectf   <= '0';
            cna_cs <= '1';
            cca <= (OTHERS => '0');
      END CASE;
    END IF;
  END PROCESS proch_etat_proc;
END rtl;
```

CANAL.VHD

```
LIBRARY IEEE;
  USE IEEE.std_logic_1164.all;
  USE IEEE.std_logic_arith.all;
  USE work.pivc_def.all;

ENTITY canal IS
  PORT( horl      : IN  std_logic;
    horls      : IN  std_logic;
    razg     : IN  std_logic;
    cca      : IN  std_logic;
    bpv      : IN  std_logic;
    bcanal   : IN  std_logic_vector(NB_CANAL-1 DOWNTO 0);
    bsite    : IN  std_logic_vector(NB_SITE-1 DOWNTO 0);
    bdi      : IN  std_logic_vector(NB_DI-1 DOWNTO 0);
    bdp      : IN  std_logic_vector(NB_DP-1 DOWNTO 0);
    canal_id  : IN  std_logic_vector(NB_CANAL-1 DOWNTO 0);
    site     : OUT std_logic_vector(NB_SITE-1 DOWNTO 0);
    rcc      : OUT std_logic;
    stim     : OUT std_logic
      );
END canal;

ARCHITECTURE rtl OF canal IS

TYPE etat_canal  IS (REPOS,REQCA,DEBUT_STIM,PHASE_A,
       INTER_PHASE,PHASE_B);
SIGNAL etat : etat_canal;
SIGNAL ris  : std_logic_vector(NB_DI-1 DOWNTO 0);
SIGNAL rps  : std_logic_vector(NB_DP-1  DOWNTO 0);
SIGNAL rss  : std_logic_vector(NB_SITE-1 DOWNTO 0);
SIGNAL rip  : std_logic_vector(NB_DI-1 DOWNTO 0);
```

```
SIGNAL rpp  : std_logic_vector(NB_DP-1  DOWNTO 0);
SIGNAL reg_compt        : std_logic_vector(NB_DP-1 DOWNTO 0);
SIGNAL req_nouv_stim    : std_logic;
SIGNAL   acc_nouv_stim  : std_logic;

BEGIN
-----------------------------------------------------------------------
-- Contrôleur de chargement des paramètres et du site de stimulation
-----------------------------------------------------------------------
ccp:PROCESS(horl,razg)
  BEGIN
    IF (razg = '0') THEN
      req_nouv_stim <= '0';
      ris          <= (OTHERS => '0');
      rps          <= (OTHERS => '0');
      rss          <= (OTHERS => '0');
    ELSIF (horl'EVENT AND horl = '1') THEN
      IF (bpv = '1' AND  bcanal = canal_id) THEN
        req_nouv_stim <= '1';
        ris          <= bdi;
        rps          <= bdp;
        rss          <= bsite;
      ELSIF (acc_nouv_stim = '1') THEN
        req_nouv_stim <= '0';
      END IF;
    END IF;
  END PROCESS ccp;
-----------------------------------------------------------------------
-- Chargement des registres "présent" avec le contenu des
-- registres "suivant"
-----------------------------------------------------------------------
reg_p:PROCESS(horl,razg)
  BEGIN
    IF (razg = '0') THEN
      rip    <= (OTHERS => '0');
      rpp    <= (OTHERS => '0');
      site <= (OTHERS => '0');
    ELSIF (horl'EVENT AND horl = '1') THEN
      IF (acc_nouv_stim = '1') THEN
        rip  <= ris;
        rpp  <= rps;
        site <= rss;
      END IF;
    END IF;
  END PROCESS reg_p;

-----------------------------------------------------------------------
-- Machine à état du contrôleur de canal
-----------------------------------------------------------------------
-- État présent
machine:PROCESS(horl,razg)
  BEGIN
    IF (razg = '0') THEN
      etat        <= REPOS;
      acc_nouv_stim <= '0';
      rcc         <= '0';
      stim      <= '1';
```

```
   reg_compt   <= (OTHERS => '0');
ELSIF (horl'EVENT AND horl = '1') THEN
   CASE etat IS
      WHEN REPOS =>
         IF (req_nouv_stim = '1') THEN
            etat <= REQCA;
            acc_nouv_stim <= '1';
         ELSE
            etat <= REPOS;
            acc_nouv_stim <= '0';
         END IF;
         rcc          <= '0';
         stim    <= '1';
         reg_compt   <= (OTHERS => '0');
      WHEN REQCA  =>
         IF (cca = '1') THEN
            etat    <= DEBUT_STIM;
            rcc          <= '0';
         ELSE
            etat <= REQCA;
            rcc          <= '1';
         END IF;
         acc_nouv_stim <= '0';
         stim      <= '1';
         reg_compt   <= (OTHERS => '0');
      WHEN DEBUT_STIM =>
         IF (rpp = ZERO_DELAI) THEN
            etat      <= REPOS;
            stim      <= '1';
         ELSIF (horls = '1') THEN
            etat    <= PHASE_A;
            stim      <= '0';
         ELSE
            etat    <= DEBUT_STIM;
            stim    <= '1';
         END IF;
         acc_nouv_stim <= '0';
         rcc       <= '0';
         reg_compt   <= (OTHERS => '0');
      WHEN PHASE_A =>
         IF (horls = '1'
                                 AND rpp = (unsigned(reg_compt)+'1')
            AND rip /= ZERO_DELAI) THEN
            etat    <= INTER_PHASE;
            stim    <= '1';
            reg_compt <= (OTHERS => '0');
         ELSIF (horls = '1'AND rpp = (unsigned(reg_compt)+'1')
            AND rip = ZERO_DELAI) THEN
            etat    <= PHASE_B;
            stim    <= '0';
            reg_compt <= (OTHERS => '0');
         ELSE
            IF(horls = '1') THEN
               reg_compt <= unsigned(reg_compt) + '1';
            END IF;
            etat      <= PHASE_A;
            stim      <= '0';
```

```
            END IF;
            acc_nouv_stim    <= '0';
            rcc          <= '0';
         WHEN INTER_PHASE  =>
            IF (horls = '1' AND
               rip = (unsigned(reg_compt)+'1')) THEN
               etat      <= PHASE_B;
               stim      <= '0';
               reg_compt   <= (OTHERS => '0');
            ELSE
               IF(horls = '1') THEN
                  reg_compt <= unsigned(reg_compt) + '1';
               END IF;
                  etat      <= INTER_PHASE;
                  stim      <= '1';
            END IF;
            acc_nouv_stim    <= '0';
            rcc            <= '0';
         WHEN PHASE_B =>
         IF (horls = '1' AND
            rpp = (unsigned(reg_compt) + '1')) THEN
               etat    <= REPOS;
               stim    <= '1';
               reg_compt <= (OTHERS => '0');
         ELSE
            IF(horls = '1') THEN
               reg_compt <= unsigned(reg_compt) + '1';
            END IF;
               etat      <= PHASE_B;
               stim      <= '0';
            END IF;
            acc_nouv_stim    <= '0';
            rcc            <= '0';
         WHEN OTHERS =>
            etat        <= REPOS;
            acc_nouv_stim <= '0';
            rcc          <= '0';
            stim      <= '1';
            reg_compt   <= (OTHERS => '0');
      END CASE;
   END IF;
  END PROCESS machine;
END rtl;
```

PIVC.VHD

```
LIBRARY IEEE;
  USE IEEE.std_logic_1164.all;
  USE IEEE.std_logic_arith.all;
  USE work.pivc_def.all;

ENTITY pivc IS
  PORT(tx_mode      : IN  std_logic;
    horl_direct     : IN  std_logic;
    horl            : IN  std_logic;
    donnees_direct : IN  std_logic;
```

```
      signal_manch   : IN  std_logic;
      raz_manch      : IN  std_logic;
      -- Sorties du FPLD
      horl_manch     : OUT std_logic;
      horl_sortie    : OUT std_logic;
      cna_a          : OUT std_logic_vector(NB_CANAL-1 DOWNTO 0);
      cna_rw         : OUT std_logic;
      cna_rs         : OUT std_logic;
      cna_en         : OUT std_logic;
      cna_cs         : OUT std_logic;
      cna_ld         : OUT std_logic;
      cna_db         : OUT std_logic_vector(NB_AMP-1 DOWNTO 0);
      stim           : OUT std_logic_vector(NCANAL-1 DOWNTO 0);
      site         : OUT std_logic_vector((NCANAL*NB_CANAL)-1 DOWNTO 0);
      -- Points de test
      razg           : OUT std_logic;
      pt_q1          : OUT std_logic;
      pt_q2          : OUT std_logic;
      bpv            : OUT std_logic;
      trmdtc_sort    : OUT std_logic;
      mcv_sort       : OUT std_logic;
      moderam        : OUT std_logic;
      raz_adresse    : OUT std_logic;
      horls          : OUT std_logic;
      pt_donnees     : OUT std_logic;
      donnees_manch  : OUT std_logic;
      lire           : OUT std_logic;
      ecrire         : OUT std_logic;
      donnees_sort   : OUT std_logic;
      trame_ech_sort : OUT std_logic_vector(2 DOWNTO 0);
      mcmd_ech_sort  : OUT std_logic_vector(2 DOWNTO 0)
          );
END pivc;

ARCHITECTURE rtl OF pivc IS

-----------------------------------------------------------------------
--  Declaration des modules
-----------------------------------------------------------------------
COMPONENT OPNDRN
   PORT (a_in : IN STD_LOGIC;
       a_out: OUT STD_LOGIC);
END COMPONENT;

COMPONENT manch
PORT(tx_mode        : IN  std_logic;
  horl_direct       : IN  std_logic;
  donnees_direct    : IN  std_logic;
  signal_manch      : IN  std_logic;
  raz_manch         : IN  std_logic;
  horl_manch        : OUT std_logic;
  donnees_manch     : OUT std_logic;
  horl              : OUT std_logic;
  donnees           : OUT std_logic;
  pt_q1             : OUT std_logic;
  pt_q2             : OUT std_logic
  );
```

```
END COMPONENT;

COMPONENT cpivc
 PORT(horl        :IN  std_logic;
  dcts            :IN  std_logic;
  razg            :OUT std_logic;
  sync            :OUT std_logic;
  bamp            :OUT std_logic_vector(NB_AMP-1 DOWNTO 0);
  bdp             :OUT std_logic_vector(NB_DP-1 DOWNTO 0);
  bdi             :OUT std_logic_vector(NB_DI-1 DOWNTO 0);
  bsite           :OUT std_logic_vector(NB_SITE-1 DOWNTO 0);
  bcanal          :OUT std_logic_vector(NB_CANAL-1 DOWNTO 0);
  bpv             :OUT std_logic;
  bths            :OUT std_logic_vector(NB_BTHS-1 DOWNTO 0);
  -- Signaux et donnees de verification
  trame_don       : OUT std_logic_vector(NB_TRAME-1 DOWNTO 0);
  trmdtc          : OUT std_logic;
  mcmd            : OUT std_logic_vector(NB_MCMD-1 DOWNTO 0);
  mcv             : OUT std_logic;
  rc1             : OUT std_logic_vector(NB_RC1-1 DOWNTO 0);
  rc2             : OUT std_logic_vector(NB_RC2-1 DOWNTO 0);
  rc3             : OUT std_logic_vector(NB_RC3-1 DOWNTO 0);
  canal_ent       : OUT std_logic_vector(NB_CANAL-1 DOWNTO 0);
  site_ent        : OUT std_logic_vector(NB_SITE-1 DOWNTO 0);
  moderam         : OUT std_logic;
  ecrire          : OUT std_logic;
  raz_adresse     : OUT std_logic;
  adresse_ramacs : OUT std_logic_vector(NB_ADR_RAMACS-1 DOWNTO 0);
  lire            : OUT std_logic;
  canal_sort      : OUT std_logic_vector(NB_CANAL-1 DOWNTO 0);
  site_sort       : OUT std_logic_vector(NB_SITE-1 DOWNTO 0)
   );
END COMPONENT;

COMPONENT horlstim
 PORT(horl      : IN  std_logic;
  razg          : IN  std_logic;
  sync          : IN  std_logic;
  rhs           : IN  std_logic_vector(NB_HORLS-1 DOWNTO 0);
  horls         : OUT std_logic
   );
END COMPONENT;

COMPONENT ccna
 PORT(horl      : IN  std_logic;
  razg          : IN  std_logic;
  bcanal        : IN  std_logic_vector(NB_CANAL-1 DOWNTO 0);
  bamp          : IN  std_logic_vector(NB_AMP-1 DOWNTO 0);
  bpv        : IN std_logic;
  rcc        : IN  std_logic_vector(NCANAL-1 DOWNTO 0);
  cca        : OUT std_logic_vector(NCANAL-1 DOWNTO 0);
  cna_a      : OUT std_logic_vector(NB_CANAL-1 DOWNTO 0);
  cna_rw     : OUT std_logic;
  cna_rs     : OUT std_logic;
  cna_en     : OUT std_logic;
  cna_cs     : OUT std_logic;
  cna_ld     : OUT std_logic;
```

```
  cna_db    : OUT std_logic_vector(NB_AMP-1 DOWNTO 0)
   );
END COMPONENT;

COMPONENT canal
 PORT(horl   : IN  std_logic;
  horls      : IN  std_logic;
  razg       : IN  std_logic;
  cca        : IN  std_logic;
  bpv        : IN  std_logic;
  bcanal     : IN  std_logic_vector(NB_CANAL-1 DOWNTO 0);
  bsite      : IN  std_logic_vector(NB_SITE-1 DOWNTO 0);
  bdi        : IN  std_logic_vector(NB_DI-1 DOWNTO 0);
  bdp        : IN  std_logic_vector(NB_DP-1 DOWNTO 0);
  canal_id   : IN  std_logic_vector(NB_CANAL-1 DOWNTO 0);
  site       : OUT std_logic_vector(NB_SITE-1 DOWNTO 0);
  rcc        : OUT std_logic;
  stim       : OUT std_logic
   );
END COMPONENT;

-----------------------------------------------------------------------
--  Declaration des signaux
-----------------------------------------------------------------------
SIGNAL donnees        :std_logic;
SIGNAL pivc_razg      :std_logic;
SIGNAL pivc_sync      :std_logic;
SIGNAL pivc_bamp      :std_logic_vector(NB_AMP-1 DOWNTO 0);
SIGNAL pivc_bdp       :std_logic_vector(NB_DP-1 DOWNTO 0);
SIGNAL pivc_bdi       :std_logic_vector(NB_DI-1 DOWNTO 0);
SIGNAL pivc_bsite     :std_logic_vector(NB_SITE-1 DOWNTO 0);
SIGNAL pivc_bcanal    :std_logic_vector(NB_CANAL-1 DOWNTO 0);
SIGNAL pivc_bpv       :std_logic;
SIGNAL pivc_bths      :std_logic_vector(NB_DP-1 DOWNTO 0);
SIGNAL pivc_horls     :std_logic;
SIGNAL pivc_rcc       :std_logic_vector(NCANAL-1 DOWNTO 0);
SIGNAL pivc_cca       :std_logic_vector(NCANAL-1 DOWNTO 0);
SIGNAL canal_no       :std_logic_vector((NCANAL*NB_CANAL)-1 DOWNTO 0);


SIGNAL cna_a_int      :std_logic_vector(NB_CANAL-1 DOWNTO 0);
SIGNAL cna_rw_int     :std_logic;
SIGNAL cna_rs_int     :std_logic;
SIGNAL cna_en_int     :std_logic;
SIGNAL cna_cs_int     :std_logic;
SIGNAL cna_ld_int     :std_logic;
SIGNAL cna_db_int     :std_logic_vector(NB_AMP-1 DOWNTO 0);
SIGNAL stim_int       :std_logic_vector(NCANAL-1 DOWNTO 0);
SIGNAL site_int       :std_logic_vector((NCANAL*NB_CANAL)-1 DOWNTO 0);
SIGNAL trmdtc         :std_logic;
SIGNAL mcv            :std_logic;
SIGNAL sync           :std_logic;
SIGNAL    trame_don :std_logic_vector(NB_TRAME-1 DOWNTO 0);
SIGNAL    mcmd    :std_logic_vector(NB_MCMD-1 DOWNTO 0);
SIGNAL    adresse_ramacs :std_logic_vector(NB_ADR_RAMACS-1 DOWNTO 0);
SIGNAL    bths    :std_logic_vector(NB_BTHS-1 DOWNTO 0);
SIGNAL    rc1       :std_logic_vector(NB_RC1-1 DOWNTO 0);
```

```
SIGNAL    rc2       :std_logic_vector(NB_RC2-1 DOWNTO 0);
SIGNAL    rc3       :std_logic_vector(NB_RC3-1 DOWNTO 0);
SIGNAL    canal_ent :std_logic_vector(NB_CANAL-1 DOWNTO 0);
SIGNAL    site_ent  :std_logic_vector(NB_SITE-1 DOWNTO 0);
SIGNAL    canal_sort  :std_logic_vector(NB_CANAL-1 DOWNTO 0);
SIGNAL    site_sort :std_logic_vector(NB_SITE-1 DOWNTO 0);

-----------------------------------------------------------------------
--  Definition des interconnexions
-----------------------------------------------------------------------
BEGIN
manch_inst: manch
PORT MAP(tx_mode    => tx_mode,
  horl_direct       => horl_direct,
  donnees_direct    => donnees_direct,
  signal_manch      => signal_manch,
  raz_manch         => raz_manch,
  horl_manch        => horl_manch,
  donnees_manch     => donnees_manch,
  horl              => horl_sortie,
  donnees           => donnees,
  pt_q1             => pt_q1,
  pt_q2             => pt_q2
  );

cpivc_inst: cpivc
PORT MAP(
  horl         =>horl,
  dcts         =>donnees,
  razg         =>pivc_razg,
  sync         =>pivc_sync,
  bamp         =>pivc_bamp,
  bdp          =>pivc_bdp,
  bdi          =>pivc_bdi,
  bsite        =>pivc_bsite,
  bcanal       =>pivc_bcanal,
  bpv          =>pivc_bpv,
  bths         =>pivc_bths,
  -- Signaux et donnees de verification
  trame_don    => trame_don,
  trmdtc       => trmdtc,
  mcmd         => mcmd,
  mcv          => mcv,
  rc1          => rc1,
  rc2            => rc2,
  rc3            => rc3,
  canal_ent      => canal_ent,
  site_ent       => site_ent,
  moderam        => moderam,
  ecrire         => ecrire,
  raz_adresse    => raz_adresse,
  adresse_ramacs => adresse_ramacs,
  lire           => lire,
  canal_sort     => canal_sort,
  site_sort      => site_sort
    );
```

```
horlstim_inst: horlstim
 PORT MAP (
  horl          => horl,
  razg          => pivc_razg,
  sync          => pivc_sync,
  rhs           => pivc_bths,
  horls         => pivc_horls
   );

ccna_inst: ccna
 PORT MAP (
  horl       => horl,
  razg       => pivc_razg,
  bcanal     => pivc_bcanal,
  bamp       => pivc_bamp,
  bpv        => pivc_bpv,
  rcc        => pivc_rcc,
  cca        => pivc_cca,
  cna_a      => cna_a_int,
  cna_rw     => cna_rw_int,
  cna_rs     => cna_rs_int,
  cna_en     => cna_en_int,
  cna_cs     => cna_cs_int,
  cna_ld     => cna_ld_int,
  cna_db     => cna_db_int
   );

canal_identite:PROCESS IS
  BEGIN
    canal_no  <= CANAL_IDENT;
  END PROCESS canal_identite;

  vect_canal: FOR canal_index IN 0 TO 15 GENERATE
    BEGIN
      canal_inst: canal
      PORT MAP (horl => horl,
        horls      => pivc_horls,
        razg       => pivc_razg,
        cca        => pivc_cca(canal_index),
        bpv        => pivc_bpv,
        bcanal     => pivc_bcanal,
        bsite      => pivc_bsite,
        bdi        => pivc_bdi,
        bdp        => pivc_bdp,
        canal_id   => canal_no(((canal_index+1)*4)-1
                      DOWNTO (canal_index*4)),
        site    => site_int(((canal_index+1)*4)-1
                      DOWNTO (canal_index*4)),
        rcc        => pivc_rcc(canal_index),
        stim    => stim_int(canal_index)
        );
    END GENERATE vect_canal;

cna_rw_inst: opndrn
 PORT MAP (
  a_in    => cna_rw_int,
  a_out   => cna_rw
```

```
      );
cna_rs_inst: opndrn
 PORT MAP (
  a_in    => cna_rs_int,
  a_out   => cna_rs
      );
cna_en_inst: opndrn
 PORT MAP (
  a_in    => cna_en_int,
  a_out   => cna_en
      );
cna_cs_inst: opndrn
 PORT MAP (
  a_in    => cna_cs_int,
  a_out   => cna_cs
      );
cna_ld_inst: opndrn
 PORT MAP (
  a_in    => cna_ld_int,
  a_out   => cna_ld
      );

opdr_adr: FOR adr_index IN 0 TO 3 GENERATE
     BEGIN
        cna_a_inst: opndrn
        PORT MAP (
          a_in    => cna_a_int(adr_index),
          a_out   => cna_a(adr_index)
          );
     END GENERATE opdr_adr;

opdr_db: FOR db_index IN 0 TO 7 GENERATE
     BEGIN
        cna_db_inst: opndrn
        PORT MAP (
          a_in    => cna_db_int(db_index),
          a_out   => cna_db(db_index)
          );
     END GENERATE opdr_db;


opdr_stim: FOR canal_index IN 0 TO 15 GENERATE
     BEGIN
        canl_opdrn_inst: opndrn
        PORT MAP (
          a_in    => stim_int(canal_index),
          a_out   => stim(canal_index)
          );
     END GENERATE opdr_stim;

opdr_site: FOR index IN 0 TO 63 GENERATE
     BEGIN
        canl_opdrn_inst: opndrn
        PORT MAP (
          a_in    => site_int(index),
          a_out   => site(index)
          );
```

```
    END GENERATE opdr_site;

-----------------------------------------------------------------------
-- Signaux de vérification et de sortie
-----------------------------------------------------------------------
sortie:PROCESS(horl,pivc_razg)
  BEGIN
    razg         <= pivc_razg;
    --bamp       <= pivc_bamp;
    --bdp        <= pivc_bdp;
    --bdi        <= pivc_bdi;
    --bsite      <= pivc_bsite;
    --bcanal     <= pivc_bcanal;
    bpv          <= pivc_bpv;
    --bths       <= pivc_bths;
    horls        <= pivc_horls;
    --rcc        <= pivc_rcc;
    --cca        <= pivc_cca;
    pt_donnees   <= donnees;
    sync         <= pivc_sync;
    trmdtc_sort  <= trmdtc;
    mcv_sort     <= mcv;
    donnees_sort        <= donnees;
    trame_ech_sort(0)   <= trame_don(0);
    trame_ech_sort(1)   <= trame_don(15);
    trame_ech_sort(2)   <= trame_don(30);
    mcmd_ech_sort(0)    <= mcmd(0);
    mcmd_ech_sort(1)    <= mcmd(15);
    mcmd_ech_sort(2)    <= mcmd(25);
  END PROCESS sortie;
END rtl;
```

ANNEXE E

SCHÉMAS DES CIRCUITS IMPRIMÉS DU PROTOTYPE

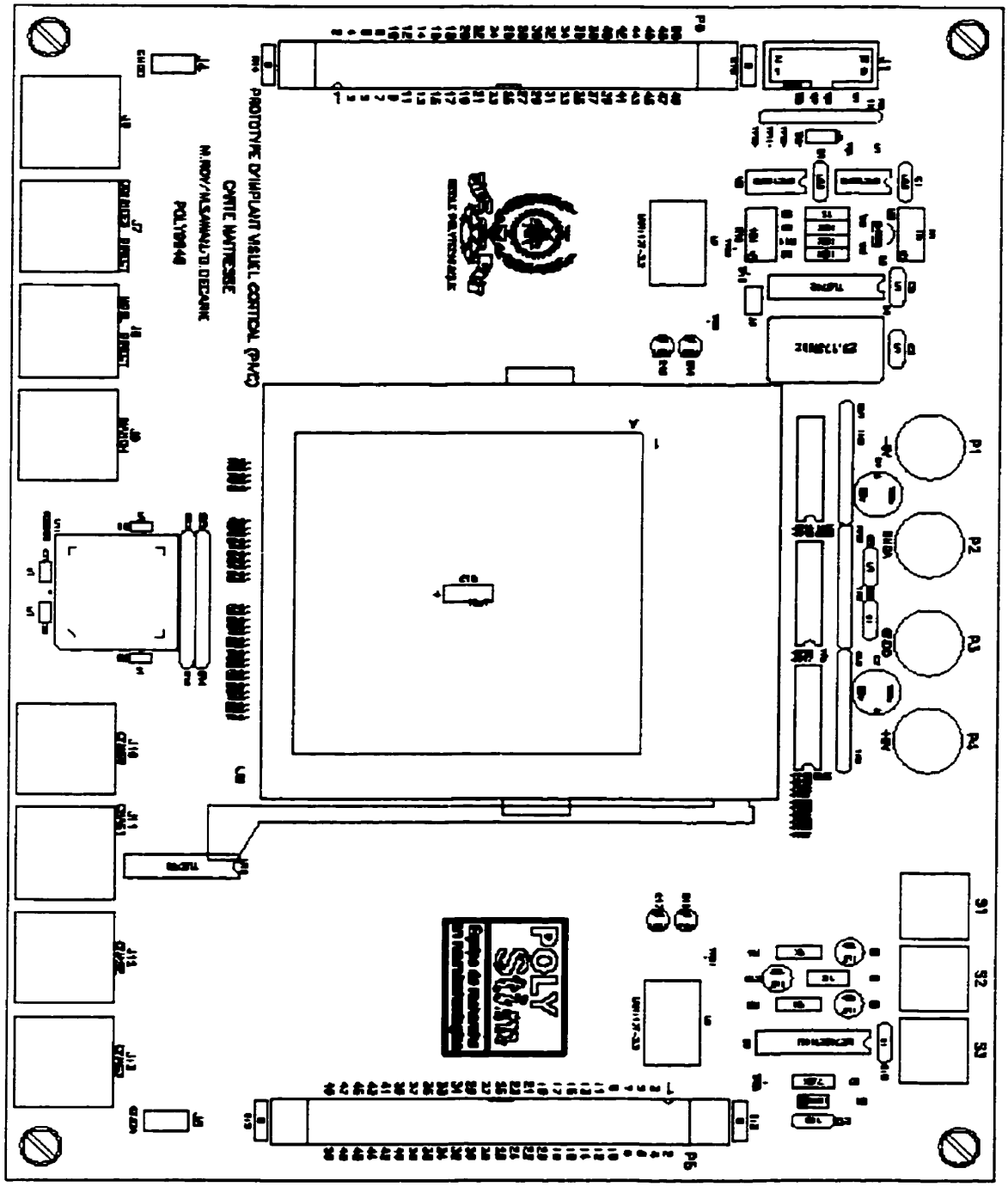


Figure E.1 : Circuit imprimé de la carte maîtresse (1/6)

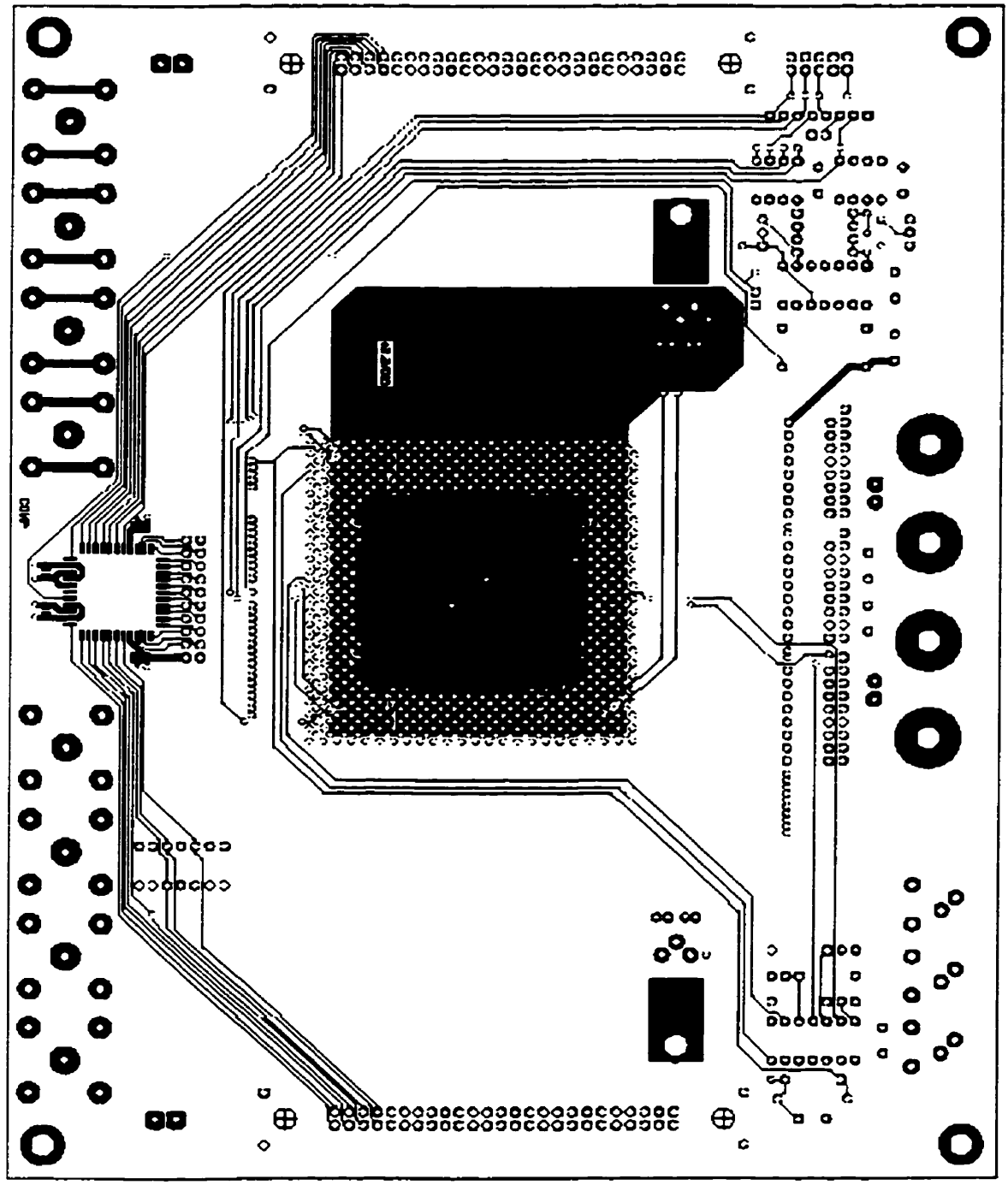


Figure E.2 : Circuit imprimé de la carte maîtresse (2/6)

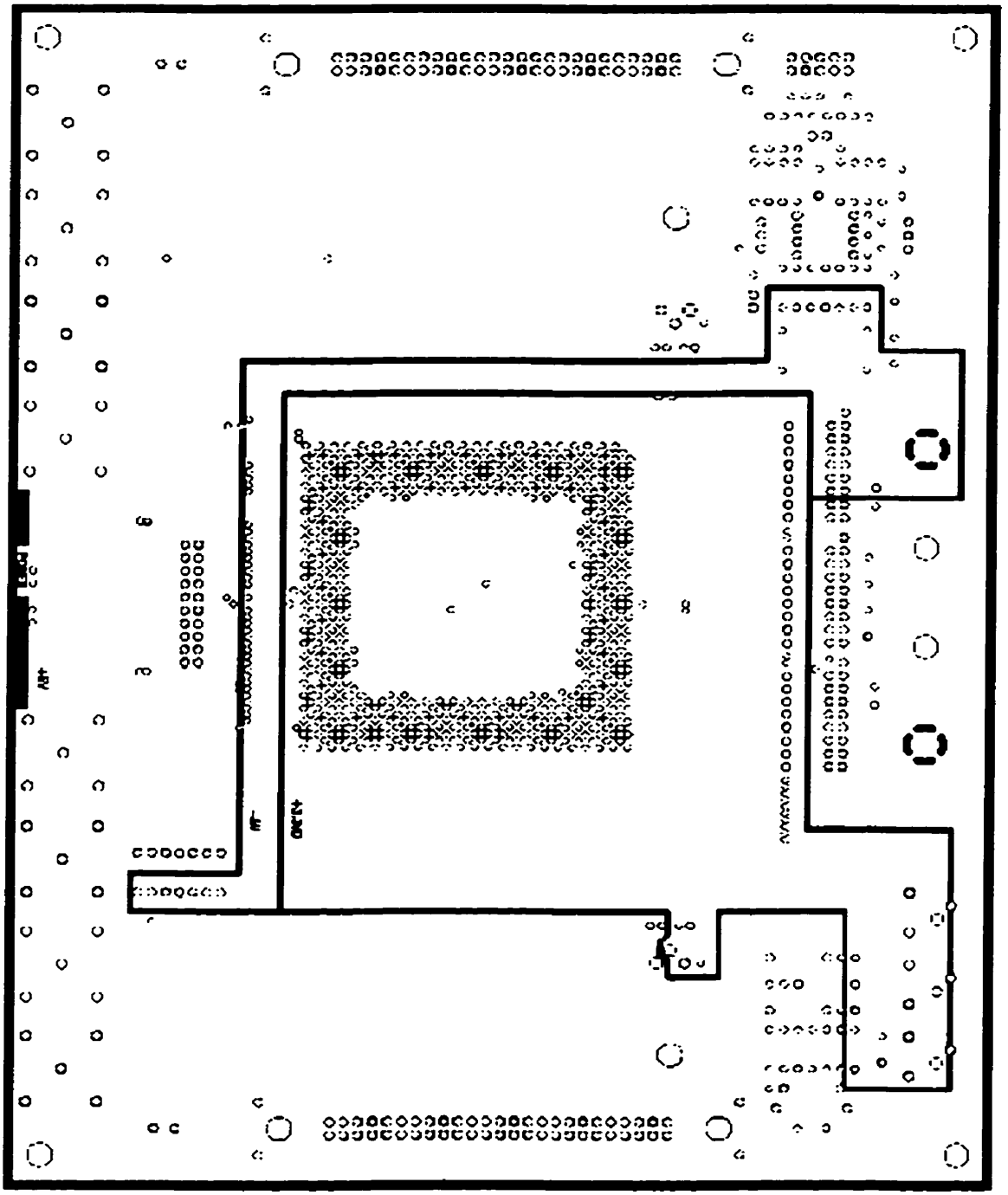


Figure E.3 : Circuit imprimé de la carte maîtresse (3/6)

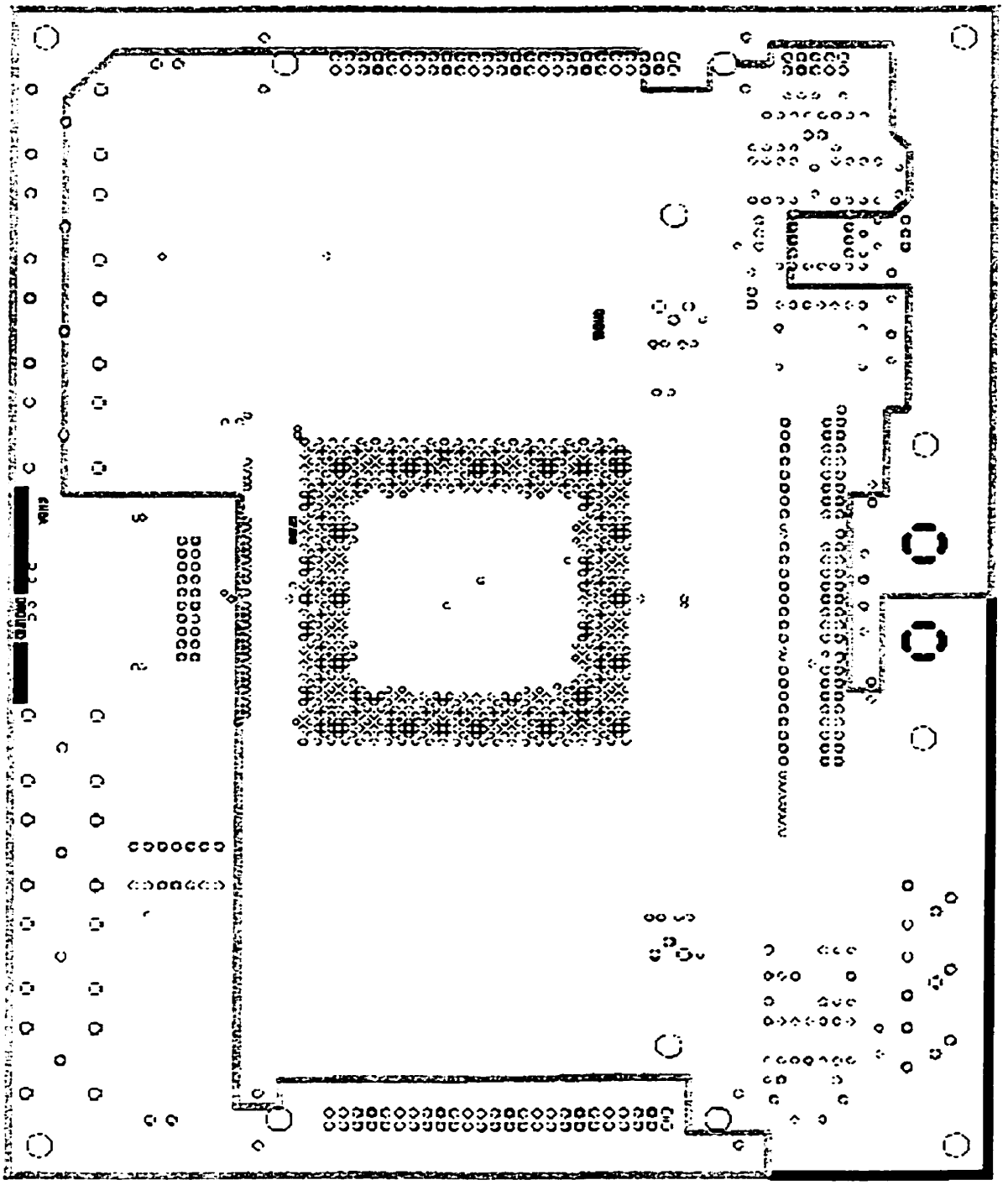


Figure E.4 : Circuit imprimé de la carte maîtresse (4/6)

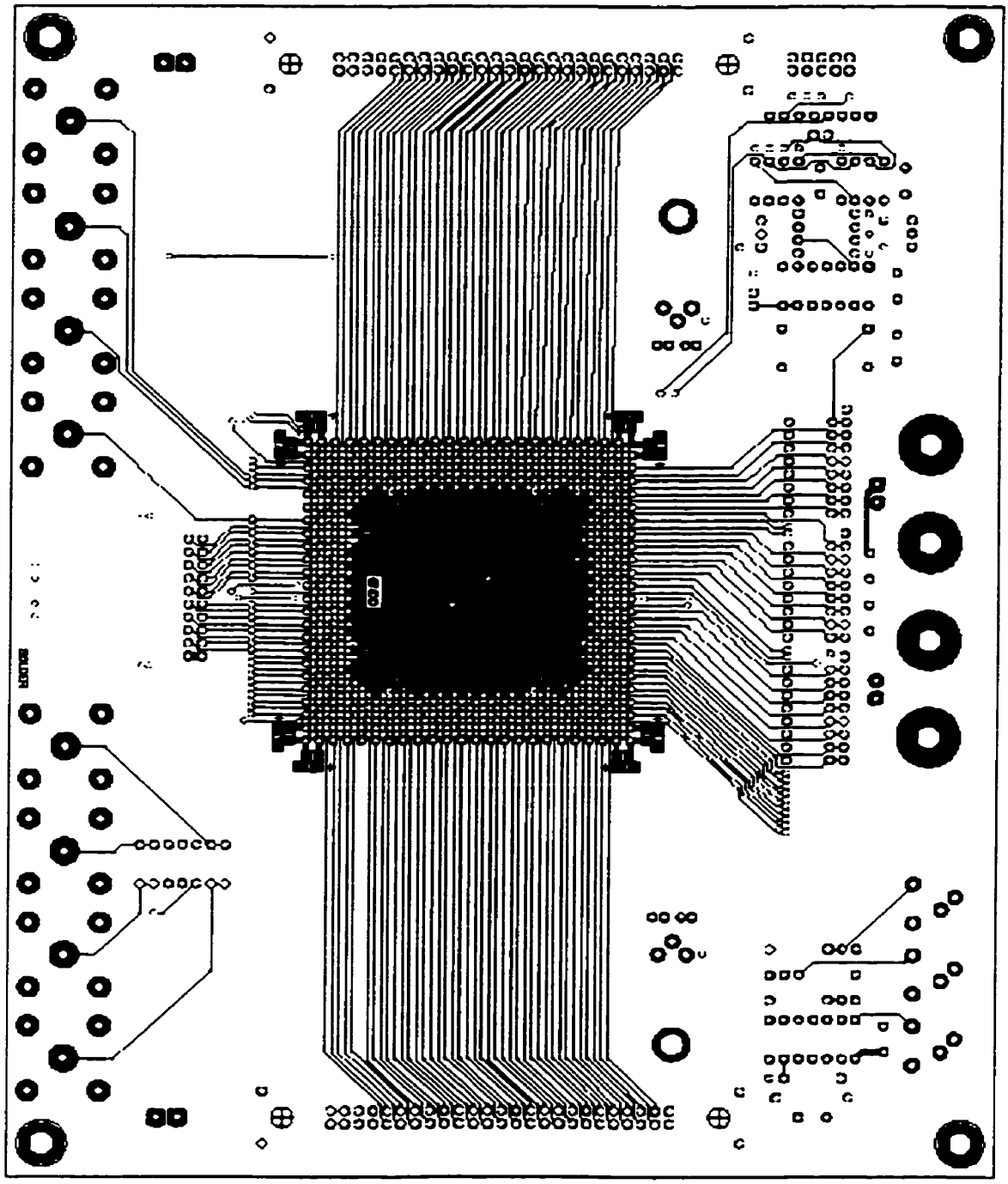


Figure E.5 : Circuit imprimé de la carte maîtresse (5/6)

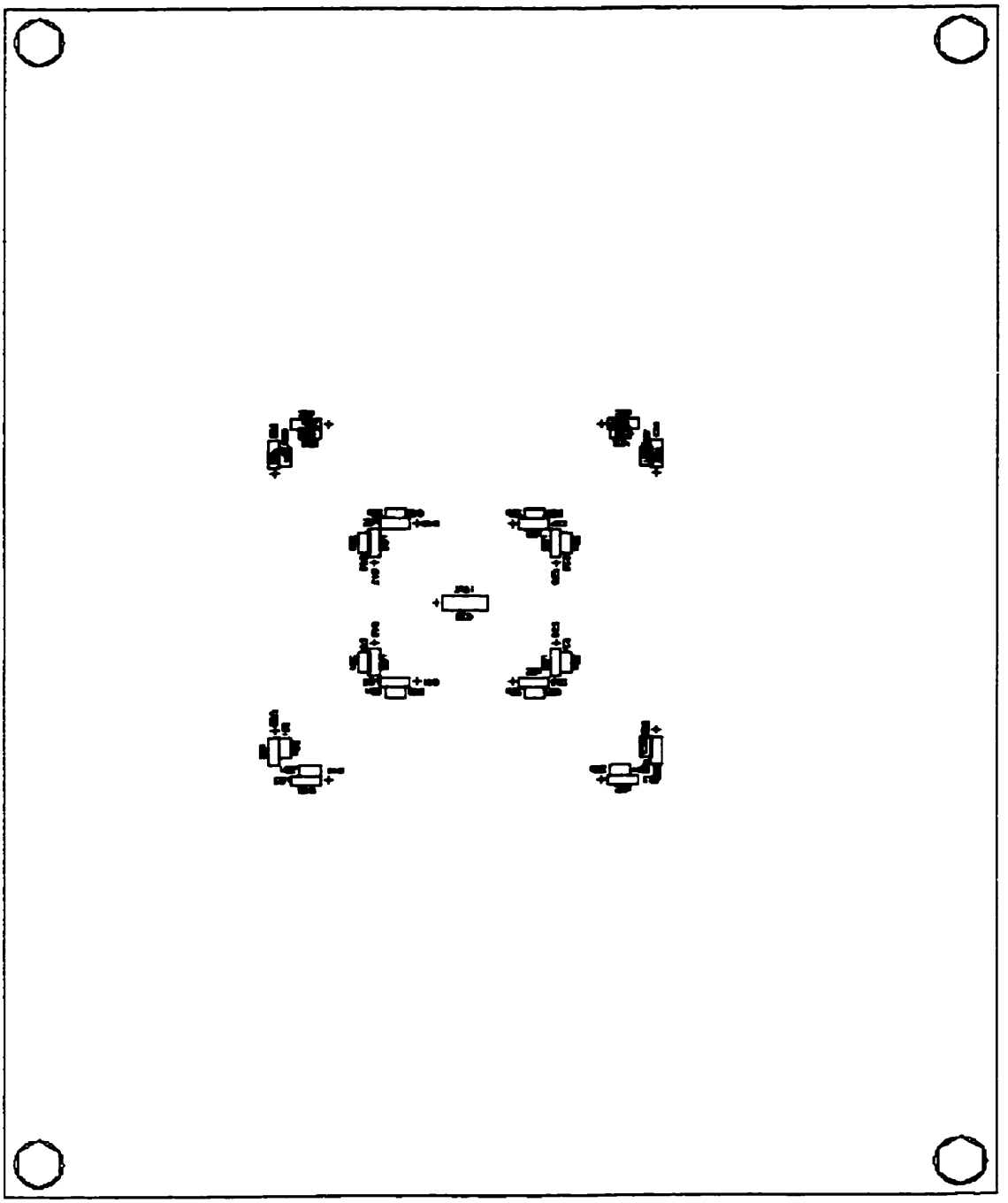


Figure E.6 : Circuit imprimé de la carte maîtresse (6/6)

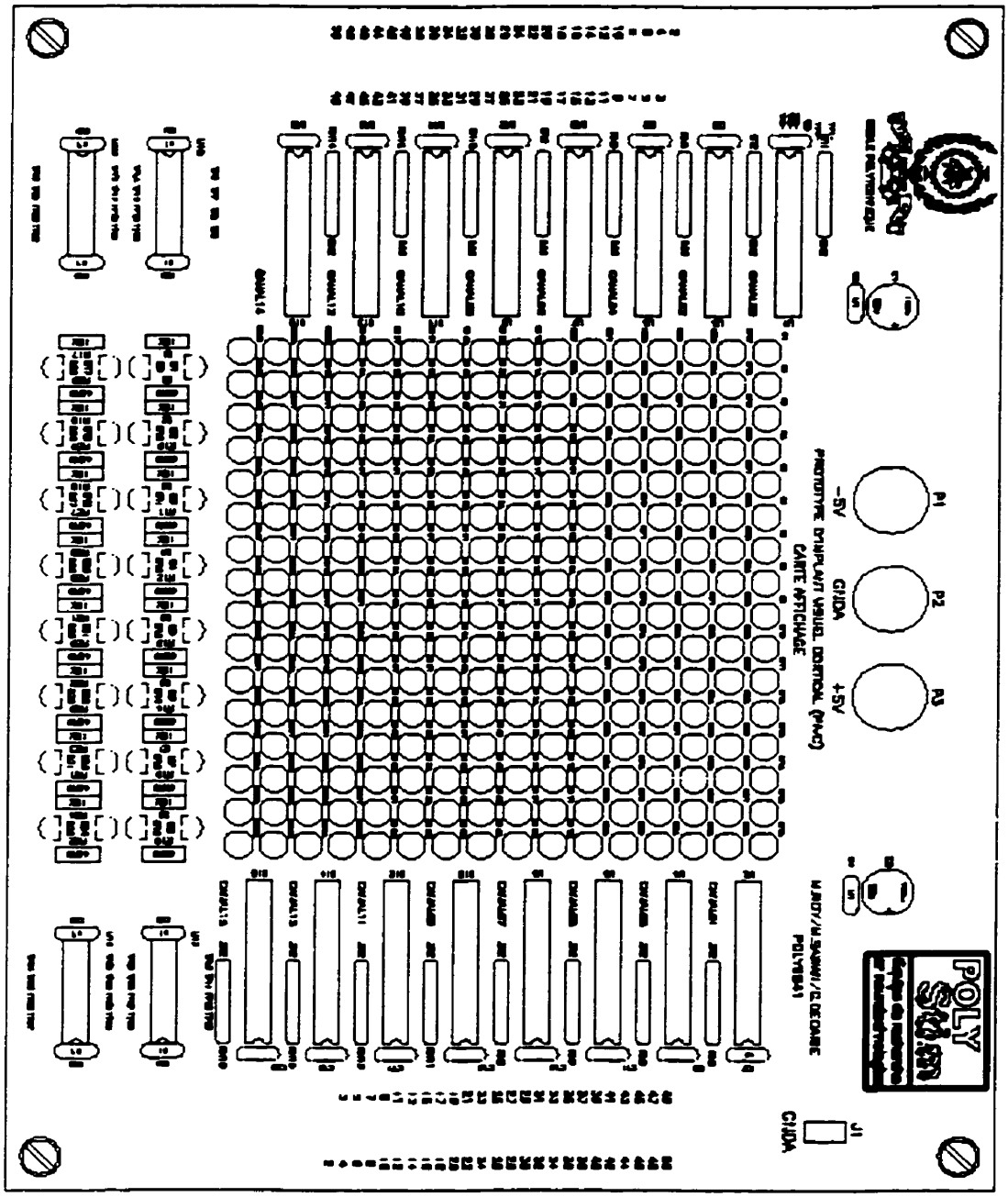


Figure E.7 : Circuit imprimé de la carte affichage (1/3)

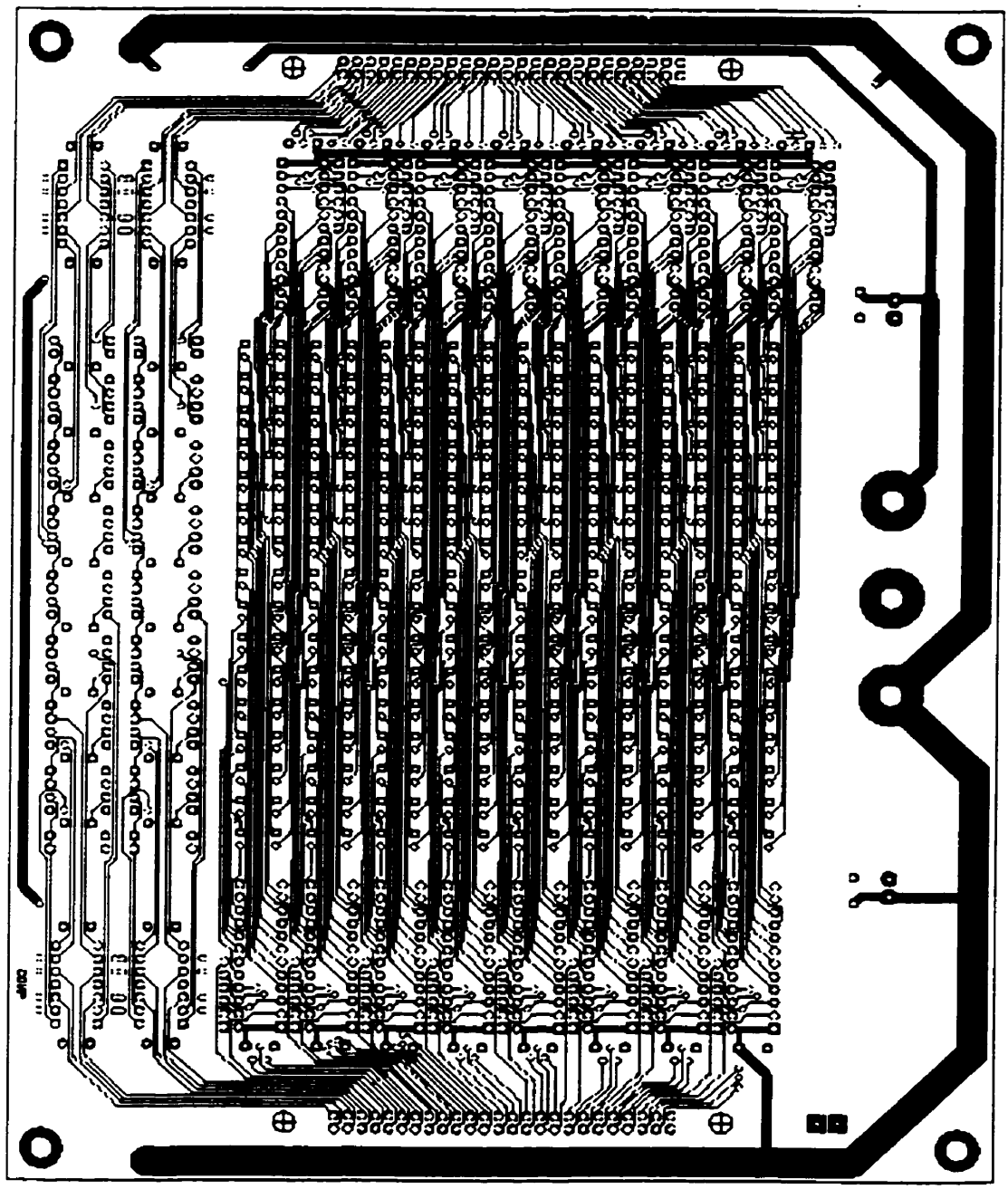


Figure E.8 : Circuit imprimé de la carte affichage (2/3)

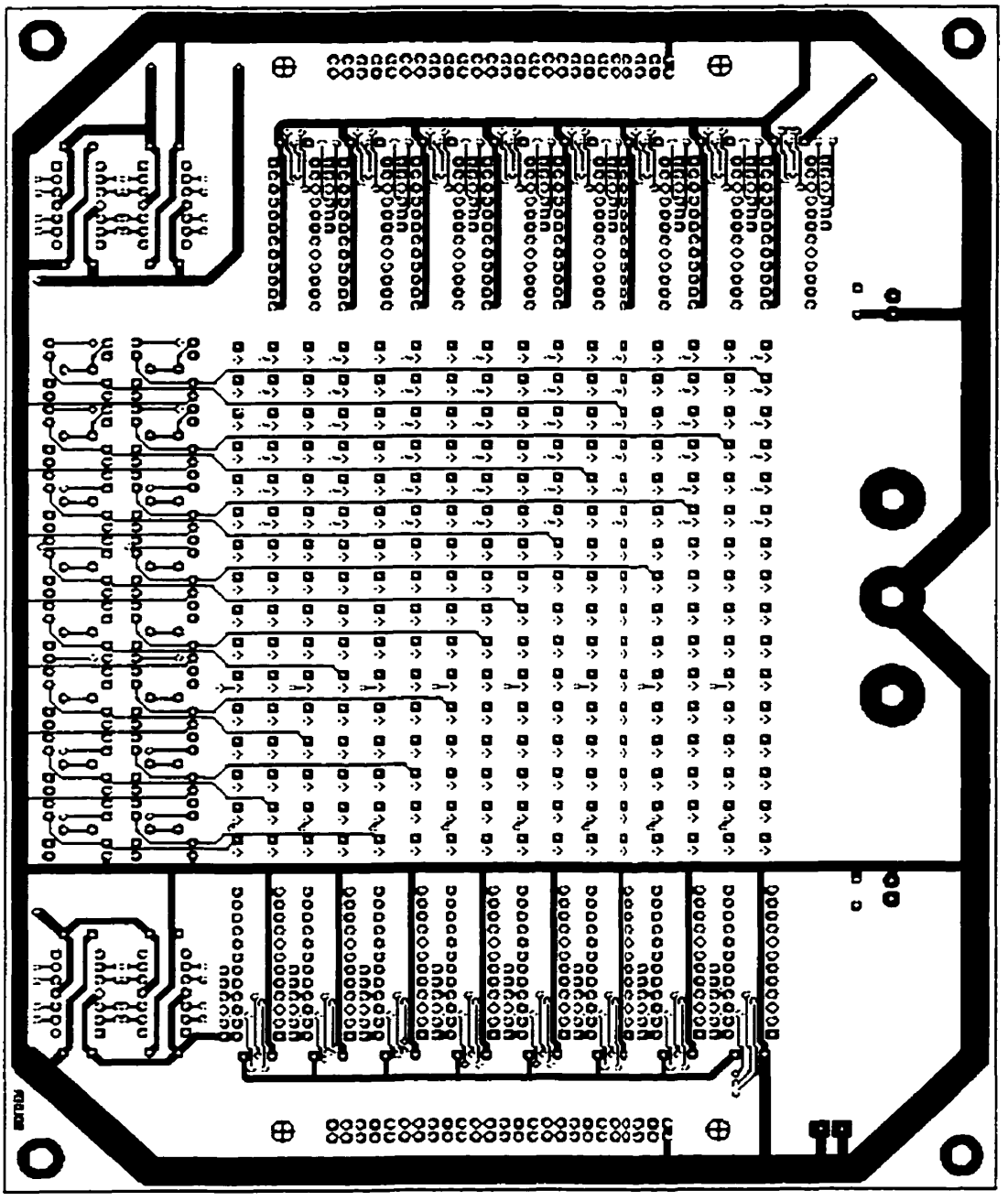
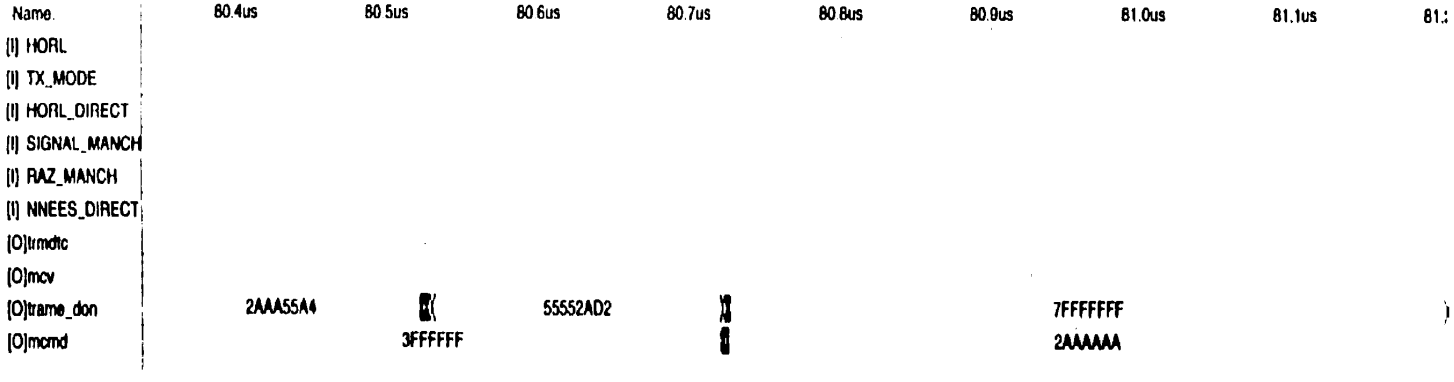


Figure E.9 : Circuit imprimé de la carte affichage (3/3)

ANNEXE F

RÉSULTATS DE SIMULATION DU CONTRÔLEUR



MAX+plus II 9.1 File: U:\ROY\MAITRISE\VHDL\DESIGN_02_A\PIVC.SCF Date: 09/14/1999 14:57:39 Page: 1

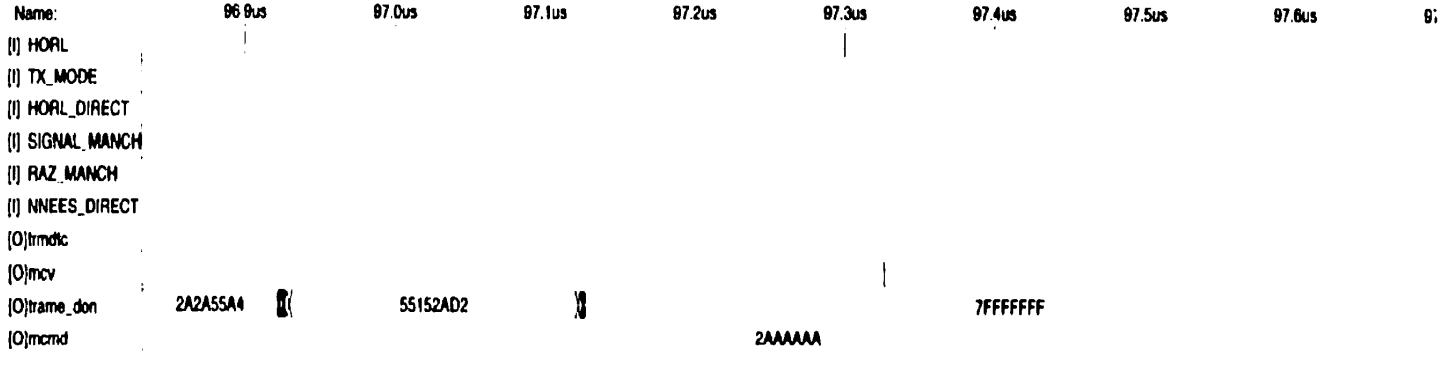


Figure F.1 : Simulation du correcteur de Hamming

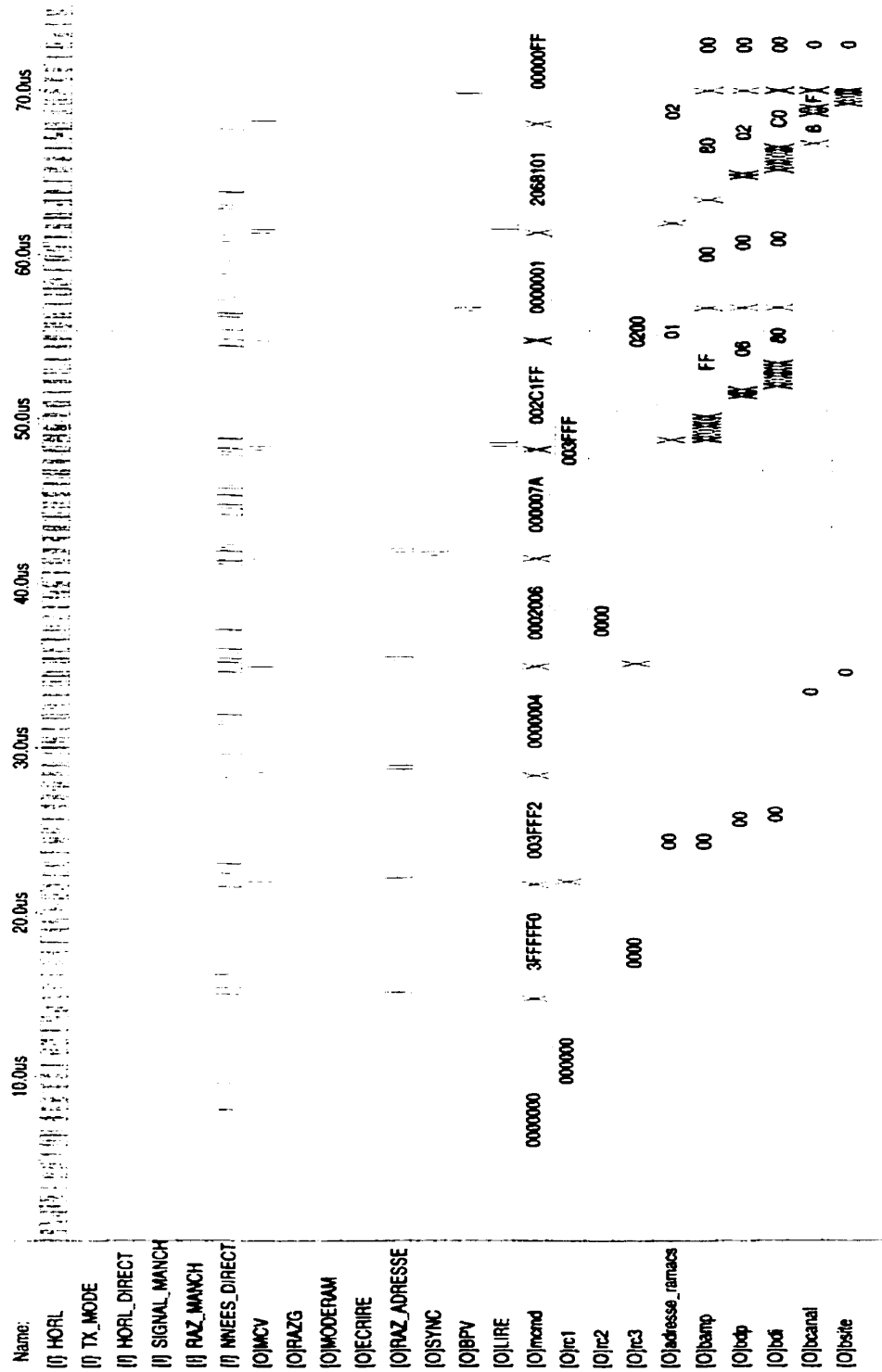


Figure F.2 : Simulation du fonctionnement du CPIVC (1/4)

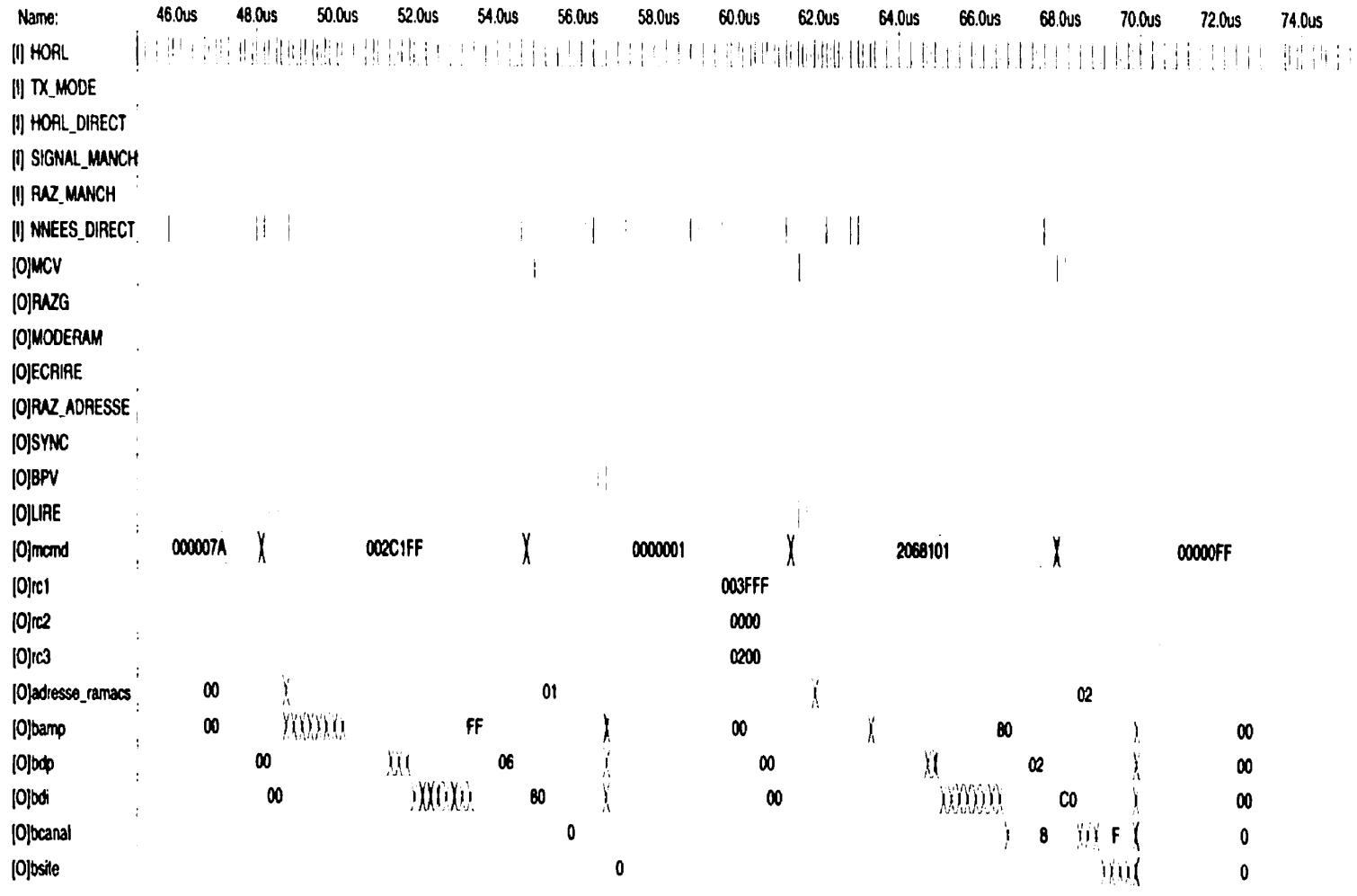


Figure F.3 : Simulation du fonctionnement du CPIVC (2/4)

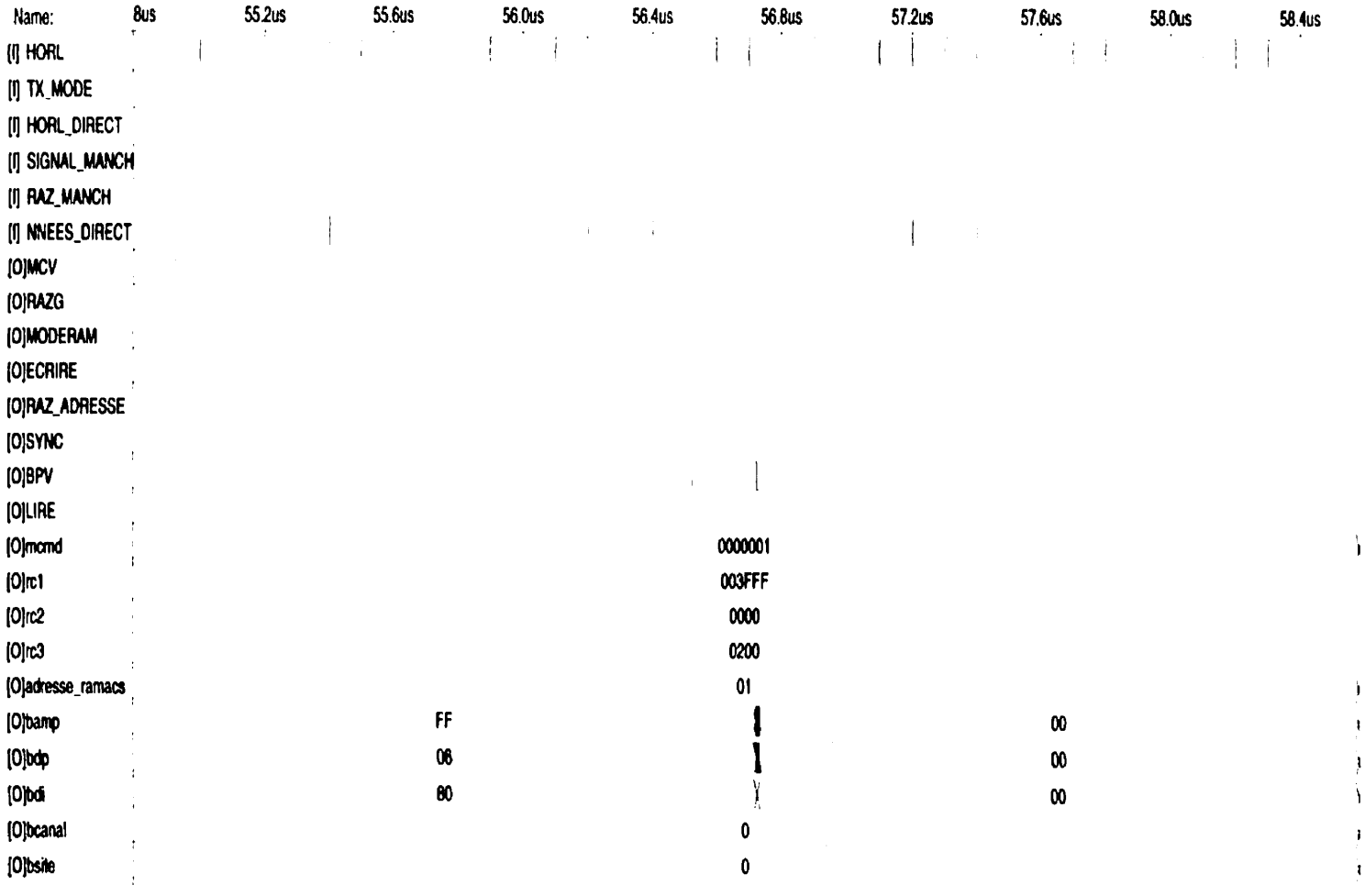


Figure F.4 : Simulation du fonctionnement du CPIVC (3/4)

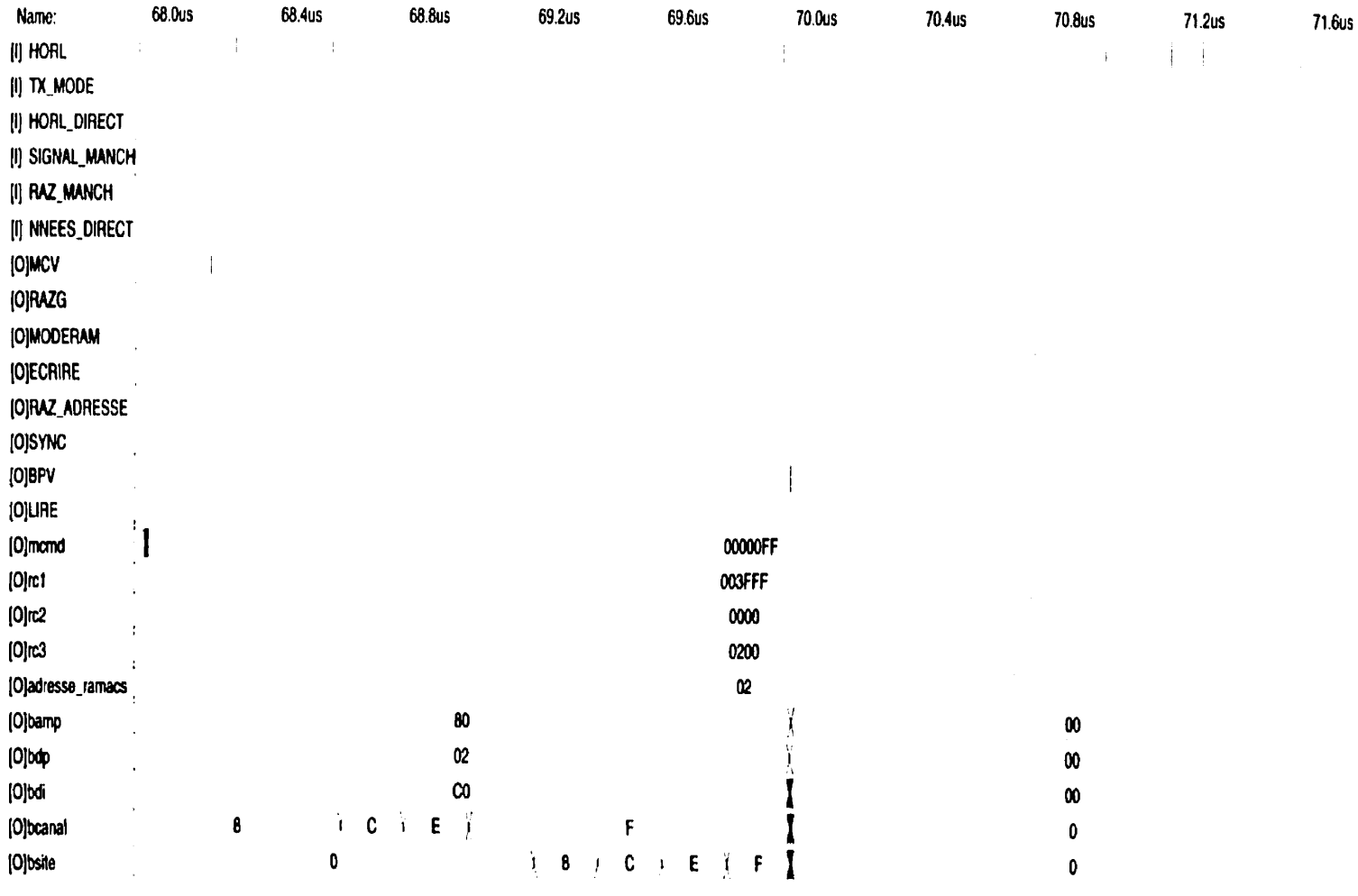


Figure F.5 : Simulation du fonctionnement du CPIVC (4/4)

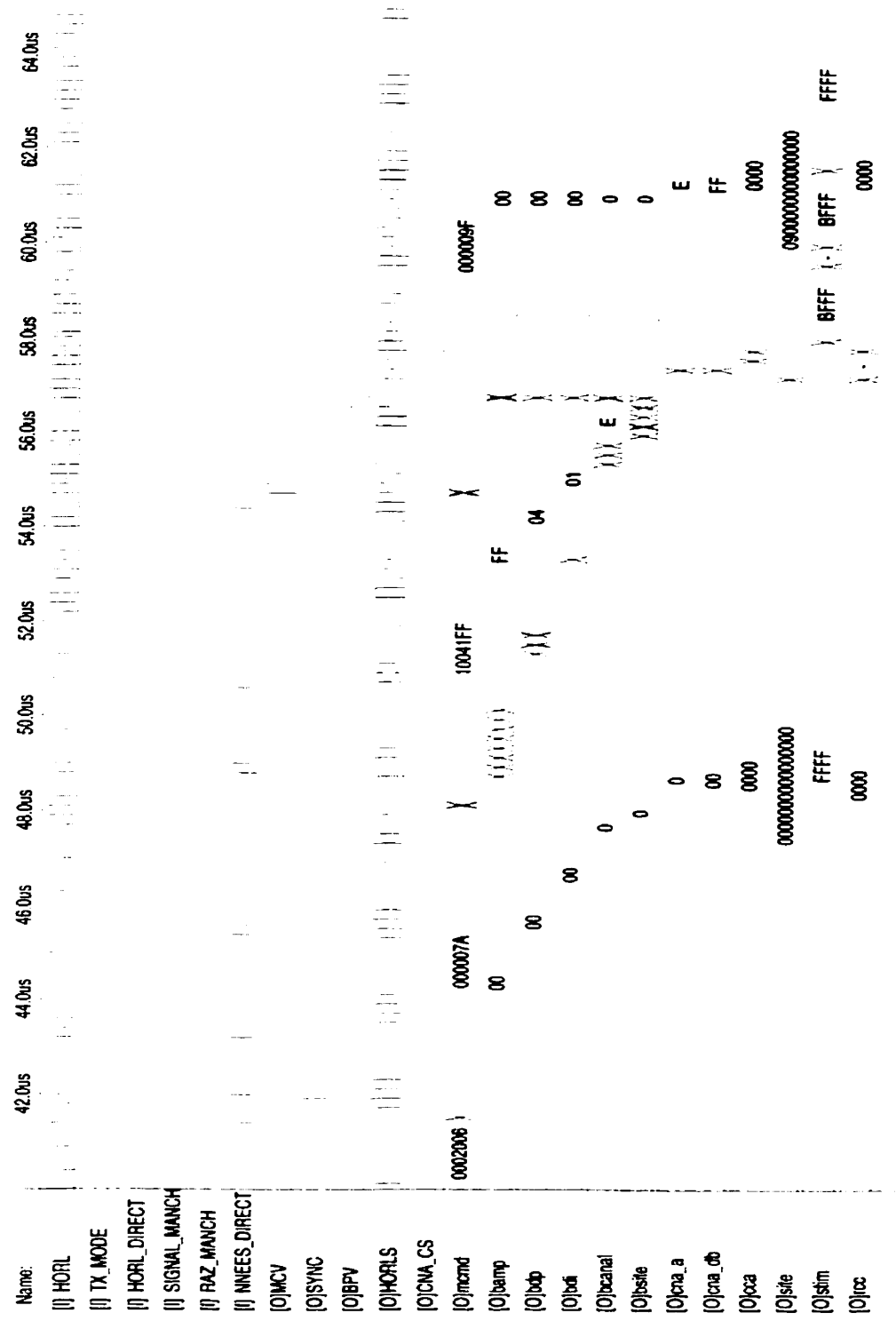


Figure F.6 : Simulation du fonctionnement d'un canal de stimulation (1/2)

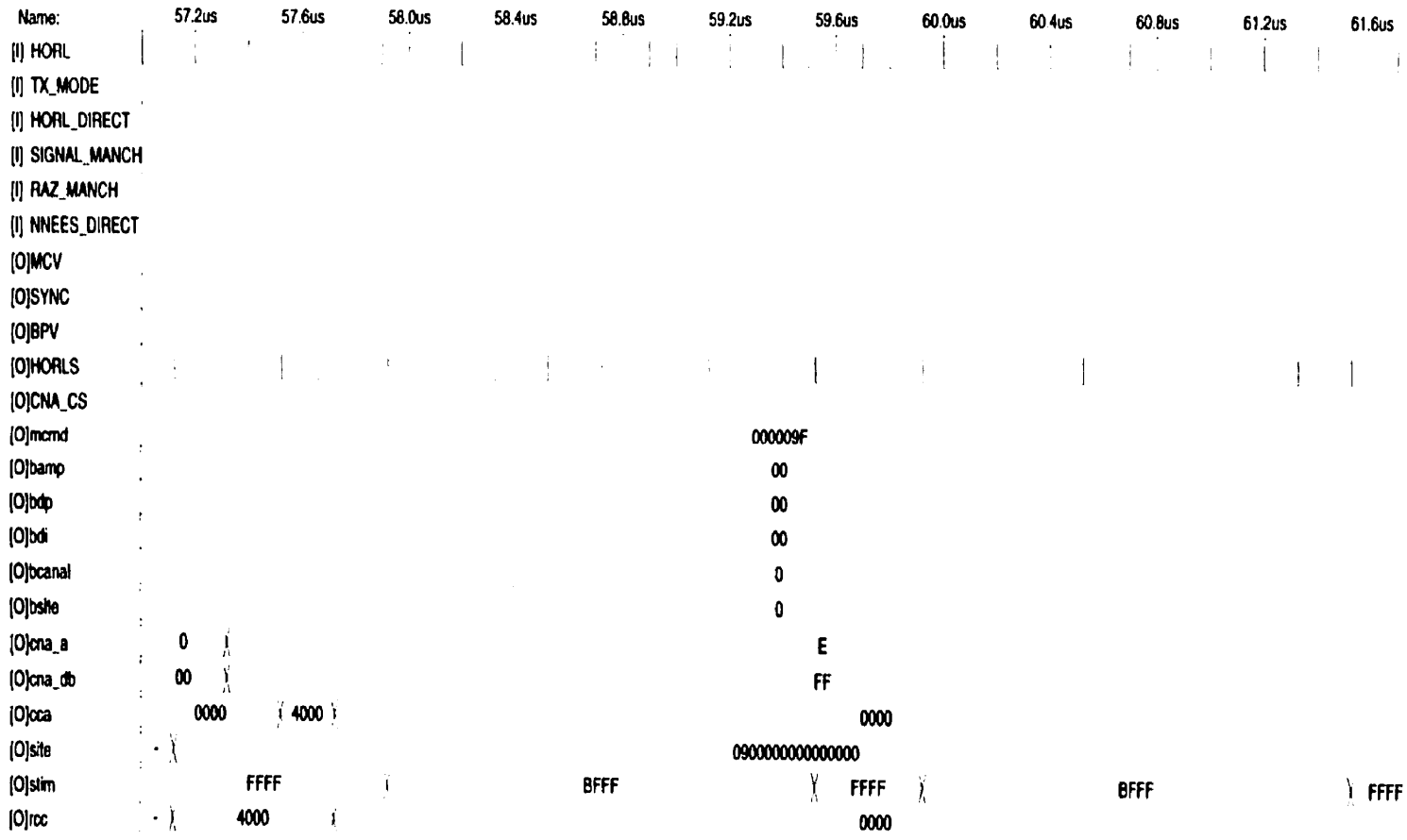


Figure F.7 : Simulation du fonctionnement d'un canal de stimulation (2/2)

ANNEXE G

LE BANC DE TEST

G.1 Introduction

Cette annexe décrit principalement le fonctionnement du banc de test utilisé pour tester le PIVC et décrit brièvement sa structure interne. Le banc de test est réalisé à partir de la carte de développement UP1 de Altera qui possède notamment un FPLD (EPF10K20). Celui-ci est donc employé pour stimuler le PIVC à l'aide de la ligne codé Manchester ainsi que des lignes de données et d'horloge directes. Plusieurs signaux internes du testeur sont rendus disponible via les E/S du FPLD pour permettre de vérifier le bon fonctionnement du testeur.

Le testeur offre sept différents types de test (T01 à T07) dont les descriptions figurent au tableau F.2. Ces types permettent notamment de stimuler un seul et unique site, de stimuler tous les sites séquentiellement afin d'afficher une image (sur un choix de six) sur la matrice de diodes électroluminescentes, et finalement de créer un effet de mouvement de l'image sur cette matrice. L'ensemble de ces tests permet de valider la fonctionnalité du PIVC. Le testeur possède donc une bonne partie de la fonctionnalité du contrôleur externe requis par le système de restauration de la vision.

La prochaine section de l'annexe décrit le fonctionnement et le mode d'emploi du testeur et la section suivante présente une brève description de sa structure.

G.2 Fonctionnement de la carte de test (carte de développement)

Une fois le FPLD programmé, une étape de configuration du testeur est nécessaire avant le début des opérations de test. Une série de huit interrupteurs (FLEX_SW1) ainsi que

deux boutons poussoirs (FLEX_PB1 et FLEX_PB2) sont disponibles sur la carte UP1 et permettent de choisir le test à effectuer, déterminer la configuration du PIVC et commander les opérations de stimulation. Le tableau F.1 liste le contenu des registres de configuration du testeur et leur numéro d'identification.

Tableau G.1 Contenu des registres de configuration

#Identification	Contenu / type de test
00	RC1 : NBDP-FDP-NBA-FA
01	RC1 :LCMSP(2-0)-FCS-NBDI-FDI
02	RC1 :LCMSP(7-3)
03	RC2 : AMP
04	RC2 : DPC
05	RC3 : DIC
06	RC3 : BTHS
07	DII : Délai inter-image. Délai d'attente entre l'envoi de deux images consécutives.
08	NIM : Nombre d'image à envoyer par scène. Applicable pour les types de test T03 à T7 seulement (image en mouvement). Pour l'envoi d'une image par scène, la valeur de NIM doit être égale à 00 (la valeur 01 est interdite). Pour l'envoi de deux images par scène, NIM = 2. Pour 3, NIM = 3 ...etc.
09	Adresse de la LED à activer (site(7-4) et canal(3-0)) de stimulation (type T01 seulement).
0A	NRAM : Numéro de la RAM contenant une image (de 00 à 05).
0B	BTMT : Base de temps pour divers temporisateurs du testeur.
0C	DICMSP : Délai inter-chaîne de MSP. Plus précisément, la durée entre l'envoi du dernier MSP d'une chaîne et le début de chargement du premier MSP de la chaîne suivante.

Le registre qui correspond au numéro d'identification 00 par exemple permet de charger les valeurs destinées à la première partie (8 premiers bits) du registre de configuration de RC1 du PIVC. Les 13 registres de configurations doivent tous être chargés avec une certaine valeur préalablement aux opérations de test. La procédure pour entrer les données de configuration est la suivante :

1 à l'aide de FLEX_SW1, entrer le numéro d'identification correspondant au registre de configuration voulu;

2 pousser et relâcher FLEX_PB1 pour enregistrer le numéro d'identification;

3 à l'aide de FLEX_SW1, entrer la valeur voulue dans le registre dont le numéro d'identification vient d'être enregistré

4 pousser et relâcher FLEX_PB2 pour enregistrer cette valeur

5 recommencer pour tous les registres de configuration.

Le tableau F.2 décrit chaque type de test et liste leur numéro d'identification. Dans tous les modes, lorsque le mode d'adressage des sites est prédéterminé, la séquence des adresses est simplement incrémentale.

Dans le cas de T01, la même adresse est enregistrée dans toutes la RAM ACS. Cette adresse doit figurer dans le registre ADRS. La procédure permettant d'activer les opérations du testeur est la suivante :

1 à l'aide de FLEX_SW1, entrer le numéro d'identification correspondant au type de test voulu;

2 pousser et relâcher FLEX_PB1;

3 pousser et relâcher FLEX_PB2.

Lorsque FLEX_PB1 et FLEX_PB2 sont poussés et relâchés en même temps, le testeur est alors réinitialisé.

Tableau G.2 : Description des types de test

Type	# Ident.	Description
T01	10	Mesure des paramètres de stimulations utilisée pour activer une LED de la matrice sélectionnée par l'utilisateur du testeur. Commande d'activation simple, c-à-d envoi d'un seul MSP. Implique de définir AMPC, DPC, DIC et DICMSP. Les mots de stimulation sont constitués selon la configuration définie.
T02	11	Affichage d'image suite à une configuration définie par l'utilisateur du testeur. Un seul paramètre est spécifique et les valeurs sont stockées dans les six RAMS. Il y a donc un choix de six différentes images.
T03	12	Image en mouvement horizontal vers la droite.
T04	13	Image en mouvement horizontal vers la gauche.
T05	14	Image en mouvement vertical vers le haut.
T06	15	Image en mouvement vertical vers le bas.
T07	16	Balayage des 6 images.
TFIN	1F	Arrêt de la stimulation

En ce qui concerne la synchronisation du système, la fréquence du cristal à l'entrée du FLPD du testeur est Fcristal = 25.175 MHz (la période Pcristal = 39.72195ns). La fréquence d'horloge du testeur (réellement utilisée par les modules du testeur) est un multiple (8) de la fréquence d'horloge à l'entrée du Donc, Fhorl = Fcristal/multiple = 25.175MHz/8 = 3.146875 MHz (Période Phorl = 317.7756ns). La fréquence à laquelle les données sont transmises au PIVC est un multiple (2) de la fréquence d'horloge du testeur. Alors Ftrans = 1.573438 MHz (Période Ptrans = 635.5511ns).

Le nombre de cycle d'horloge (horl_direct) entre le bit d'arrêt d'une trame précédente et le bit de départ de la trame suivante est déterminé par la valeur de DICMSP. Si cette valeur est inférieure à 18 et que BTMT = 02, alors il n'y a aucun délai (ou cycle d'horloge) entre les deux trames. Si la valeur est égale ou supérieure à 18, alors le nombre de cycle d'horloge est égal à DICMSP – 17. En ce qui concerne le délai inter-image, la valeur minimale est de 21. En respectant les conditions de départ (valeurs de BTHS et BTMT), il est donc possible de déterminer avec précision le moment d'envoi de chaque trame et donc de prévoir avec certitude le comportement du PIVC.